

# Je développe avec Visual Web Developer 2005 Express



*Vous découvrirez sur le CD Rom de ce numéro, la version bêta 2 du logiciel Microsoft Visual Web Developer 2005 Express (VWD), c'est le remplaçant de Web Matrix.*

*Bien que payant, ce nouvel IDE constitue une heureuse surprise et surtout un outil bien plus puissant que Web Matrix, tout en étant compatible avec la gamme Visual Studio.*

## Installation

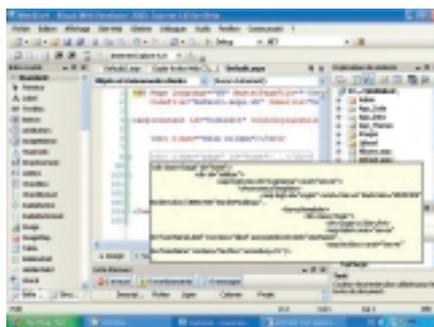
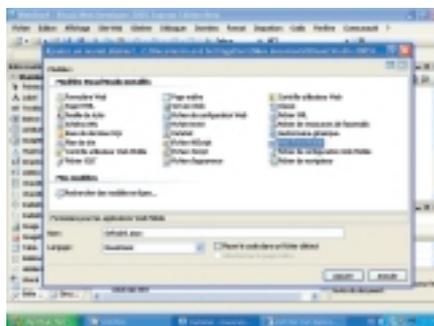
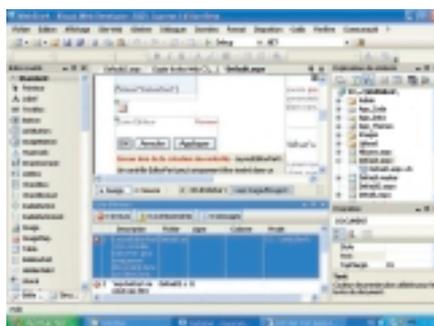
L'installation se déroule sans souci, que ce soit à partir de notre CD Rom ou du site MSDN (compter une centaine de mégas octets). Si vous êtes en Windows XP, il faut le SP 2 d'installé. Sur certaines configurations, la préversion de SQL Server 2005 Express peut échouer (ce fut le cas lors de nos tests).

## À quoi ça sert cet outil ?

Avant toute chose, VWD sert à créer des applications web de type ASP.NET 2.0. Cependant, .NET oblige, on peut aussi coder en C#, VB et J#. Il permet de créer et de gérer des sites web dynamiques, de créer des pages web dynamiques ou non. Comme un véritable IDE, il inclut un RAD pour construire l'interface et possède un puissant débogueur. À noter que vous n'avez pas besoin d'avoir IIS, (le serveur web de Microsoft) pour tester. L'outil incorpore un serveur web de tests (durant nos tests, nous avons rencontré quelques dysfonctionnements en mode exécution - debug). Le côté XML n'est pas oublié, avec la possibilité de créer des services web. Pour faciliter la vie du développeur novice en technologie .NET et dans le développement complexe, on dispose de starter kit. Ce sont des exemples clé en main, facilement adaptables avec l'ensemble des écrans et codes.

## Un démarrage en douceur

Au premier lancement de l'outil, on se retrouve dans un IDE assez complet, avec trois grandes zones de travail : la boîte à outil, l'éditeur et les palettes flottantes de projet et de propriétés. La page de démarrage est des plus simples. La section mise en route vous facilitera la prise en main de l'IDE. Pour créer un site, rien de plus simple : mise en route, option créer un site web. Vous avez le choix entre un site web et le site web personnel. L'option personnelle génère un site clé en



main, avec l'ensemble des dossiers et pages ASP.NET nécessaires. À vous ensuite de les personnaliser avec le contenu adéquat. Très pratique. Vous noterez que les pages web créées par VWD ne sont pas en HTML mais en XHTML (plusieurs déclinaisons disponibles). Si vous lancez le test de votre site avec l'option debug, le serveur de tests prendra un peu de

temps à démarrer. VWD n'est pas multi projet. Vous n'ouvrez donc qu'un projet à la fois.

## Des aides à la conception

Comme dans d'autres IDE Web, on dispose d'une vue Design, d'une vue source. Dommage que l'on ne puisse pas avoir les deux simultanément comme dans Dreamweaver. Cependant, il est très simple à partir de la vue Design d'accéder au code de tel champ, de tel bloc, de le personnaliser. La structure du code reprend l'affichage de Visual Studio. Si un bloc est " fermé ", en plaçant le curseur dessus, le code apparaît. Sur la validation du code, il est possible de choisir le type de validation selon la, ou les cibles. Dommage qu'il n'y ait pas plus de navigateurs supportés. Par contre, la personnalisation de l'éditeur est très complète, et ce, quel que soit le langage, la technologie. On notera avec plaisir les options d'accessibilité disponibles en conception des pages. Sur les bases de données, on disposera de SQL Server 2005 Express (version gratuite) qui permettra de créer rapidement et simplement des bases de données relationnelles que l'on pourra ensuite mapper directement avec VWD dans une page dynamique. On dispose de tous les éléments nécessaires pour le faire et explorer la base sélectionnée.

On apprécie beaucoup l'outil pour ses capacités, sa richesse fonctionnelle, son petit prix, les bonnes performances, même sur les petites configurations. L'assistance au développement s'avère plutôt bonne, notamment avec l'éditeur de code personnalisable et l'intellisense, et l'accès aux codes snippets. Malgré tout, il n'est pas certain qu'un utilisateur lambda souhaitant faire son site web personnel soit totalement à l'aise avec un tel outil, certes très riche, mais sans doute un peu trop compliqué à comprendre sans effort d'apprentissage.

Installez-le, testez-le !

■ François Tonic

## Report Sharp-Shooter : les rapports faciles

C'est un moteur commercial de création de rapports pour .NET, capable de créer des rapports à partir de plusieurs sources de données et comprenant de nombreuses options d'exportation comme le PDF, le HTML ou le JPG.

**F**orce est de constater que les bonnes solutions d'édition et de génération d'états pour C# disponibles en français ne sont pas légion ! Mis à part Crystal Reports, leader du marché, vous n'avez pas grand choix. Il y a bien ReportMaker, peu répandu, mais c'est à peu près tout. ReportBuilder, QuickReport, ou Rave FastReport, c'est plutôt en anglais et destiné à un public de développeurs Borland. Bref, Report Sharp-Shooter, solution complète 100% .net et francisée, se démarque du lot. Notez bien que le runtime créé par Report Sharp-Shooter est libre de droits et que par conséquent vous pouvez commercialiser une application comprenant ce runtime sans payer un euro de royalties à l'éditeur.

### 1 • Installation et configuration

Report Sharp-Shooter est livré sous forme de package MSI (Microsoft Installer). Entre parenthèses, vous pouvez télécharger une version d'essai. Pour pouvoir l'utiliser sous Visual Studio vous devez ajouter les composants Report Sharp-Shooter à la boîte à outils. Pour faire ceci, vous devez démarrer Visual Studio, créer une application C# fenêtrée, cliquer (bouton droit) sur la boîte à outils pour ajouter une nouvelle séparation (Add Tab) en lui affectant un nom (Report Sharp-Shooter), puis enfin ajouter les composants. Pour créer un état vous avez plusieurs possi-



bilités : vous pouvez passer par un assistant, créer un modèle avec le designer, ou encore le programmer. La version testée est l'édition standard 1.9 compatible avec le framework 1.1.

### 2 • Création d'une source de données

Commençons par créer une source de données en glissant-déposant un objet System.Data.DataSet sur notre formulaire. Sélectionnons les propriétés du dataSet1 et ajoutons-y une table à la Collection (add). Puis ajoutons deux colonnes (avec les noms "Nom" et "Téléphone"), et insérons le code suivant au chargement du formulaire :

```
private void Form1_Load(object sender, System
.EventArgs e)
{
    DataRow row = dataTable1.NewRow();
    row["Nom"] = "Tintin";
    row["Téléphone"] = "01-327284428";
    dataTable1.Rows.Add(row);

    row = dataTable1.NewRow();
    row["Nom"] = "Milou";
    row["Téléphone"] = "01-327284429";
    dataTable1.Rows.Add(row);
}
```

### 3 • Lier la source de données et lancer le Designer

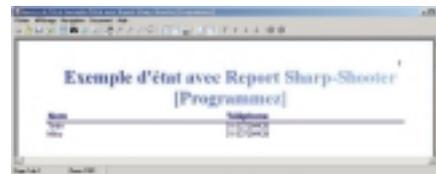
Faites maintenant glisser-déposer le composant "ReportGenerator" sur le formulaire. Nous devons lier la source de données à l'état. Sélectionnons l'objet ReportGenerator1, puis éditons sa propriété DataSources. Cliquons sur "add", puis tapons "SourcesDonnées", et sélectionnons dataSet1 dans le menu déroulant. Lançons le designer (un lien se situe en bas de la fenêtre propriétés de l'objet ReportGenerator1, ou bien cliquez deux fois sur cet objet). Choisissons Fichier/Nouveau puis "état standard". Une boîte de dialogue assistant se présente. Indiquons comme titre "Exemple d'état avec Report Sharp-Shooter [Programmez]", le style standard et le langage cible (C#). Puis cliquons sur le bouton bleu "+" en haut à gauche pour y ajouter SourcesDonnées.Table1. Ajoutons à l'état, les champs que vous désirez afficher (ici "Nom" et "Téléphone"), et sauvons notre modèle d'état.

### 4 • Ajoutons un bouton au formulaire pour générer le rapport

Ajoutons un bouton au formulaire et le code suivant qui reliera automatiquement l'état ReportGenerator1 au formulaire sauvé :

```
private void button1_Click(object sender, System
.EventArgs e)
{
    if (reportGenerator1.Template!=null)
    try
    {
        reportGenerator1.Prepare();
        using(NineRays.Reporting.View.Preview
Form previewForm = new NineRays.Reporting.
View.PreviewForm(reportGenerator1))
        {
            previewForm.WindowState = Form
WindowState.Maximized;
            previewForm.ShowDialog(this);
        }
    }
    catch(Exception ee)
    {
        MessageBox.Show(ee.Message, "Report
Sharp-Shooter", MessageBoxButtons.OK,
MessageBoxIcon.Error);
    }
}
```

En conclusion : après ce bref test nous sommes impressionnés par la qualité de cette solution commerciale. Le Designer est en fran-



çais, et il est très facile de générer un rapport à partir de n'importe quelle source de données. L'intégration à Visual Studio est parfaite. Rien à redire.

**Lien utile :** Version d'évaluation

[http://grays.net/cgi-bin/components\\_downloads.cgi?act=50&cid=93](http://grays.net/cgi-bin/components_downloads.cgi?act=50&cid=93)

■ Xavier Leclercq - [Xavier.Leclercq@programmez.com](mailto:Xavier.Leclercq@programmez.com)

# Réaliser un site Web de A (comme ASP.Net) à Z (comme Zope)

Le site web a bien changé. À chaque besoin, son site web. Cette segmentation provoque une complexité de plus en plus importante, rendant parfois un site web aussi peu contrôlable qu'un projet informatique classique. Et choisir tel ou tel langage, tel ou tel outil se résume en un parcours du combattant, donnant une bonne migraine en fin de journée.

Depuis le début du Web, les langages, les outils et les technologies se sont transformés. L'offre est devenue tellement pléthorique qu'il est parfois difficile de choisir. Les nouvelles tendances concernent les langages s'appuyant largement sur XML et l'effacement du HTML. Le Rich Media, revient en force depuis quelque temps, ainsi que le client lourd / riche. L'autre grande tendance concerne les techniques de développement, avec la séparation des couches présentation (l'interface) et du code fonctionnel, afin d'être plus souple dans les évolutions du site et de fournir une application web, la plus indépendante possible pour le navigateur. L'émergence du langage de description répond à cette demande. Un seul code, plusieurs interfaces adaptées à chaque situation, chaque terminal. L'emploi massif de CSS couplé ou non au XHTML répond aussi à ce besoin de rationaliser l'interface, le développement et surtout le code qui devient moins permissif qu'avant. Dans ce dossier, nous avons décidé d'étudier les différentes étapes de la réalisation d'un site Web, jusqu'à l'hébergement. Nous avons réuni un panel de spécialistes qui chacun dans leur domaine, vous livrent des pistes, des réflexions, et des idées concrètes pour votre futur site.

■ François Tonic

## Ressources complémentaires sur : [www.programmez.com](http://www.programmez.com)

Exemples, sources, listes d'outils. Vous pourrez aussi poser vos questions aux auteurs. Ce dossier n'est pas figé mais interactif, à l'image du web d'aujourd'hui.



## Notre panel d'experts



### Grégory Renard

Grégory Renard, aussi connu sous son pseudo Rédo, travaille depuis plus de 5 ans sur la plate-forme .NET. Il a co-fondé la société Wygwam, spécialisée dans les technologies .NET et notamment ASP.NET, au sein de laquelle il est Directeur développement.



### Xavier Leclercq

Xavier Leclercq est administrateur système, programmeur et consultant indépendant. Adolescent il développa en assembleur un logiciel anti-virus pour l'Amiga (un micro-ordinateur de la marque Commodore). Il rédige depuis plus de 10 ans des articles techniques pour la presse informatique. Il a aujourd'hui 35 ans et vit avec sa femme et ses deux enfants à Bilton en Belgique.



### Xavier Vanneste

Xavier Vanneste, a 28 ans il est développeur/formateur chez ACCESS IT, un gold partner Microsoft. Il est tombé dans l'informatique petit, mais a surtout accroché en BTS (Mécanique et Automatisation Industrielle rien à voir avec l'informatique). Il s'est mis à fond dedans et maintenant, après 5 ans d'expérience et 10 certifications Microsoft il a réussi à faire ce qu'il aime, ce dont il est assez fier.



### Gauthier Delamarre

Développeur web spécialisé sur la plate-forme Linux/Apache/PHP/MySQL depuis 5 ans, Gauthier Delamarre est également responsable du site [programmez.com](http://programmez.com)

# 1 ► DÉFINIR LES LANGAGES ET LES OUTILS

*Construire un site n'est plus aujourd'hui un exercice réservé aux seuls pros du web. Même sans aucune connaissance en langage HTML, XML, c'est possible en quelques clics. Pour ceux qui veulent faire les choses bien, la situation se complique rapidement. Comment choisir la bonne technologie ? Comment définir son site ?*

Le site est au centre d'un écosystème très vaste : la conception, l'hébergement, le contenu et sa gestion, la maintenance, l'évolution technique et technologique, la sécurité, etc. Pour un nouveau site, il s'agira tout d'abord de définir très précisément la ou les cibles d'internautes, l'environnement fonctionnel, le plan du site. C'est à partir du fonctionnel (qui doit être le plus précis possible) que vous pourrez définir les techniques, les langages et les outils à utiliser. Ce choix se fera, souvent, avec différentes contraintes : le coût (licence, développement), l'environnement serveur déjà installé ou envisagé, la formation des développeurs (langages, outils).

Pour chaque fonction de votre site, vous devez lister les technologies disponibles et les outils. C'est à partir de cette liste, et en la confrontant à vos contraintes de départ, que vous pourrez définir les langages, les technologies et les outils pour réaliser votre site. Ainsi, prenons l'exemple des pages dynamiques :

- je dois réaliser des pages dynamiques sur le site
- j'ai à ma disposition : ASP.NET, JSP, PHP, ColdFusion...

- les outils sont les IDE Web, les outils Open source pour JSP / PHP

- mes contraintes : mon serveur est en Linux

De facto, vous pouvez éliminer les outils Microsoft et ASP.NET. Vous aurez principalement le choix entre JSP et PHP. Si vous passez par un hébergeur, supporte-t-il les deux technologies ? Connaissez-vous Java ? Si vous ne souhaitez pas payer les outils, quels langages possèdent le plus d'outils open source (ayant les fonctions voulues pour le développement) ? Dans ce choix, qui doit être pragmatique avant tout et lié à chaque fonction de votre site, partez du plus général, au plus fin.

N'oubliez pas de choisir la catégorie d'applications web que vous allez faire : un site, un client riche, un client léger, un portail, un intra ou extranet, un rich media, etc. Selon la catégorie

## Les outils commerciaux pour développer un site web

**Macromedia Dreamweaver** : considéré comme la référence. Puissant IDE Web pour concevoir des sites statiques, dynamiques, clients légers ou riches. Supporte l'ensemble des langages et des standards actuels. Panoplie fonctionnelle impressionnante. Dédié aussi bien aux créateurs de sites qu'aux développeurs. Bonne intégration avec les autres outils de l'éditeur.

**Adobe GoLive** : pérennité incertaine suite au rachat de Macromedia. Moins riche que Dreamweaver, GoLive permet de développer des sites complexes, supporte l'ensemble des standards. Bonne intégration avec les autres outils de l'éditeur.

**CodeCharge Studio** : un IDE RAD Web, idéal pour développer des sites dynamiques. Supporte PHP, ASP.NET, JSP et se connecte aux principales bases de données du marché. Permet de créer des pages HTML. Dispose d'un module d'intégration à FrontPage.

**Macromedia Contribute** : version (très) light de Dreamweaver. Permet de gérer un site web, d'ajouter des pages ou du contenu.

**WebDev** : un atelier logiciel totalement intégré pour créer des sites Web (statiques ou dynamiques) et des applications web en J2EE ou .NET. Intègre l'ensemble des fonctions (tests, déploiement, paiement sécurisé...). Edité par le français PC Soft.

**Microsoft Visual Studio Web Developer** : successeur de Web Matrix, IDE Web peu cher. Dédié à ASP.NET 2.

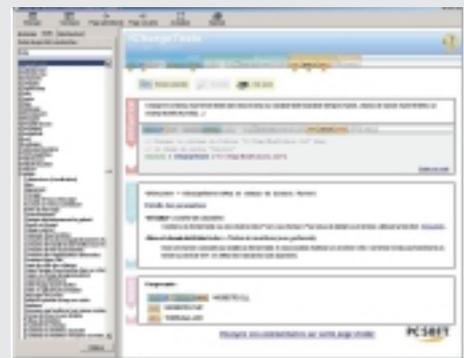
**RapidWeaver** : voici un IDE tout simple, peu coûteux (- 25 \$ US). Il permet en quelques clics de créer son premier site Internet, sans code, en cliquant sur des boutons et onglets. Il utilise le mode skin pour l'interface. Rapide et ergonomique, il plaira à tous ceux qui ne connaissent pas HTML et qui veulent un outil le plus simple possible. Petit détail : disponible uniquement sous MacOS X.

**NaMo Web Editor** : IDE Web accessible. Inclut un nettoyeur de code HTML. Possède un module RAD et des assistants pour créer des sites ASP, PHP, JSP. Module de déploiement et de vérification de compatibles avec les navigateurs.

**Front Page** : l'IDE Web le plus connu sous Windows, édité par Microsoft. Simple, il permet de créer rapidement un site Web en technologies Microsoft.

**Rational Web Developer** : IDE dédié aux applications J2EE, SOA. Inclut un RAD et des assistants de création. Existe aussi l'IDE Rational Application Developer. Totalement intégré à la nouvelle plate-forme de développement Atlantic. Dédié aux applications J2EE, Java, Web Services, SOA

■ F.T.



## 1 » DÉFINIR LES LANGAGES ET LES OUTILS

de l'application, les contraintes technologiques et d'outils se modifient !

### Les grandes étapes à respecter

- 1 • Je définis les besoins et les fonctions du site, les objectifs, les cibles et le contenu.
- 2 • Je dessine le plan du site avec l'ensemble des fonctions, des interactions. Je suis le plus précis possible. Je n'oublie pas de préciser l'architecture.
- 3 • Je choisis les technologies, langages et outils à partir de la définition du site
- 4 • Je conçois les pages et le code
- 5 • Je teste le site
- 6 • Je publie le site.

### Que faire d'un site existant ?

Vous souhaitez migrer votre site vers de nouvelles technologies ? L'opération est plus ardue que de créer de zéro. Il faudra tout d'abord recenser les pages du site, les technologies, les outils et les langages utilisés, tout en définissant les nouveaux objectifs du site, avec en correspondance le besoin ou non de changer de technologie. Si vous êtes en ASP.NET, mais que vous souhaitez passer en PHP (ou l'inverse), comment faire ? Faut-il tout réécrire ? Je suis en HTML, mais je veux passer en XHTML comment faire ? Je veux refaire l'interface ? Est-elle dynamique ou statique ? Est-elle couplée au code fonctionnel ? Si je veux utiliser CSS, ai-je les compétences, les outils nécessaires ? Quel impact sur mon code actuel ? Quand on passe, par exemple, d'ASP à ASP.NET, il faut souvent tout réécrire, quand je passe d'ASP à PHP, il existe des convertisseurs, mais la réécriture demeure. Je change de base de données, quels impacts sur mon code, les performances. Je change d'outils web, la migration des pages et du code posent-ils problème ? (il peut y avoir des différences d'implémentations de standards).

■ François Tonic

Abonnement  
en ligne

[www.programmez.com](http://www.programmez.com)

## Les outils libres pour développer un site Web

### IDE PHP

Pour programmer en PHP sous Windows, il existe un très bon outil développé en Delphi, du nom de Dev-PHP. Cet outil en est à sa version 2.0.12 et est distribué sous licence GPL. Il s'agit d'un IDE, c'est à dire d'un environnement de développement intégré, qui permet de créer des applications en PHP. Vous pouvez exécuter ou vérifier la syntaxe de vos codes sources PHP à partir de l'interface. Ce logiciel est très populaire et cela se comprend : il est en effet rapide, léger et stable. Le visuel de l'interface est agréable et, cerise sur la gâteau, il sera en français pour peu que vous ayez choisi cette option.

DEV-PHP : <http://devphp.sourceforge.net/>  
Fig. 1

### Éditeur à la frontpage

Le 14 avril 2005 était annoncée la version 1.0 (preview) de NVU (sous licence Mozilla). Il s'agit d'un éditeur de page web dérivé du module "composer" de Mozilla. Vous connaissez tous le succès de Firefox et de Thunderbird (tous deux également issus de Mozilla), et certains placent déjà NVU comme concurrent sérieux à Frontpage de Microsoft. NVU tourne sous Windows, Linux et Mac OS X. NVU est un outil WYSIWYG (What You See Is What You Get) fort complet et propose même un support de XHTML, de PHP, des commentaires d'édition, la numérotation des lignes dans l'aperçu du code source, etc.

NVU: <http://frenchmozilla.sourceforge.net/nvu/>

### Éditeur CSS

Comment développer facilement un site compatible CSS (feuilles de style en cascade) ? En fait, vous n'avez pas besoin d'un éditeur spécial mais tout comme le HTML, il est plus faci-

le de créer un site CSS avec un éditeur spécialisé. CSSED est multi-plate-forme (Windows, Linux, Mac OS X), et sous licence GPL. Vous avez à votre disposition l'auto-complétion, des assistants et un aide mémoire des propriétés et attributs CSS associés (qui sont fort nombreux !). Vous pouvez aussi valider votre code CSS, et commenter votre code.

CSSED : <http://cssed.sourceforge.net/>  
Fig. 2

### Éditeur HTML

Comment ne pas parler de Bluefish 1.0 dans le cadre d'outils libres pour le développement de sites ? Il s'agit de l'éditeur HTML WYSIWYG sous Linux le plus populaire (GPL). Son interface intuitive et puissante est en français. Vous pouvez créer un projet couvrant l'ensemble d'un site Web, votre code sera mis en couleur et enfin, l'outil est parfaitement intégré à GNOME ou KDE. Malheureusement, BlueFish ne tourne que sous Linux...

Bluefish : <http://bluefish.openoffice.nl/>  
Fig. 3

### Éditeur XML

Bitflux Editor est non pas un éditeur WYSIWYG HTML, mais bien XML (GPL). Il s'emploie, via votre navigateur Firefox et sera par conséquent multi-plate-forme (Windows, Linux, Mac OS X). Il a été écrit en JavaScript et utilise les technologies XML, XSLT, et CSS pour le rendu visuel. Vous pouvez l'employer avec n'importe quel type de document XML, mais il est préférable de posséder de solides notions en la matière pour s'en servir.

Bitflux Editor : <http://bitfluxeditor.org/>  
Fig. 4

■ Xavier Leclercq

[Xavier.Leclercq@programmez.com](mailto:Xavier.Leclercq@programmez.com)

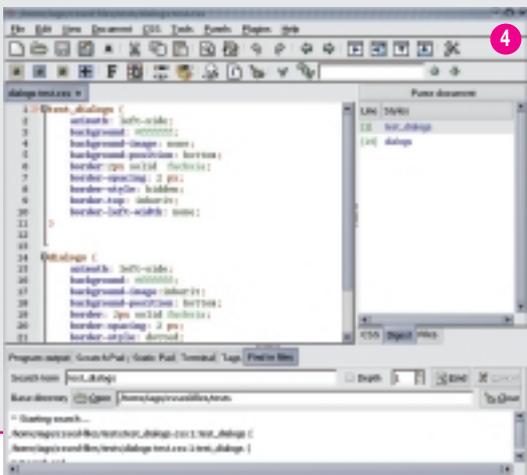
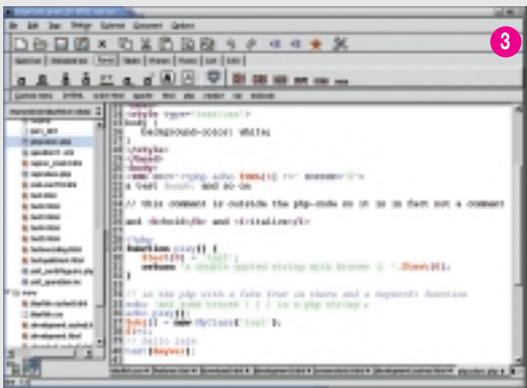
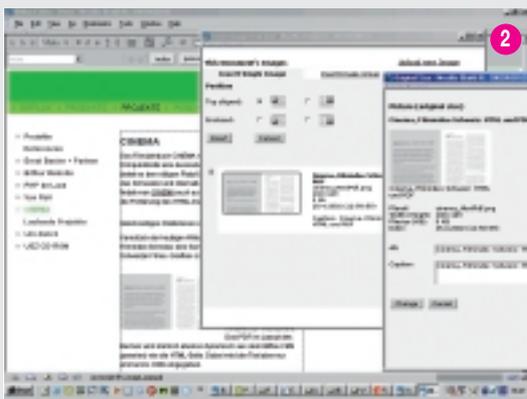
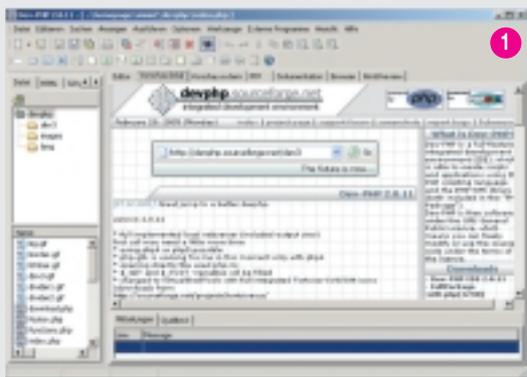
## TABLEAU DES TECHNOLOGIES

Nom	Type	Commentaire
<b>ActiveX</b> Microsoft <a href="http://www.microsoft.com/com/activex.htm">http://www.microsoft.com/com/activex.htm</a>	Client riche	API d'extension de l'architecture Microsoft Windows COM avec de grosses lacunes sécuritaires. Windows uniquement
<b>AJAX</b>	Client riche	DHTML avec XMLHttpRequest
<b>DHTML</b>	Client fin	HTML, Javascript et CSS
<b>HTML</b>	Client fin	Site statique sans feuille de style ni javascript
<b>Flash</b> Macromedia <a href="http://www.macromedia.com/fr/software/flash/">http://www.macromedia.com/fr/software/flash/</a>	Client riche / Serveur	Permet la création d'applications Web comprenant des animations vectorielles interactives
<b>JSP</b> Sun <a href="http://java.sun.com/products/jsp/tags/11/tags11.html">http://java.sun.com/products/jsp/tags/11/tags11.html</a>	Client / Serveur 3 tier	JSP est une technologie basée Java de création dynamique de code HTML/XML
<b>JWS</b> Sun <a href="http://java.sun.com/products/javawebstart/">http://java.sun.com/products/javawebstart/</a>	Client riche	Java Web Start (nécessite l'installation manuelle sur chaque poste client)
<b>Laszlo</b> Laszlo <a href="http://www.laszlo.com">http://www.laszlo.com</a>	Client riche	Partie cliente et serveur Open Source en vue de créer une application RIA multimédia
<b>Nexaweb</b> Nexaweb <a href="http://www.nexaweb.com">http://www.nexaweb.com</a>	Client / Serveur 3 tier	Plate-forme de développement J2EE avec un IDE WYSIWYG
<b>PDF&amp;XDP</b> Adobe <a href="http://www.adobe.com">http://www.adobe.com</a>	Client riche	Sur-ensemble de PDF en combinaison avec du Javascript
<b>ROR</b> <a href="http://www.rubyonrails.org/">http://www.rubyonrails.org/</a>	Client / Serveur 3 tier	Framework client / serveur Ruby On Rails pour des développements ultra rapide
<b>SMIL</b>	Client riche	Synchronized Multimedia Integration Language. HTML de haut niveau pour réaliser des présentations audiovisuelles
<b>ULC</b> Canoo <a href="http://www.canoo.com/ulc">http://www.canoo.com/ulc</a>	Client riche / Serveur	Canoo Ultra Light Client. Déploiement via JWS. ULC a pour but de développer des RIA pour les applications J2EE
<b>WSC</b> Microsoft <a href="http://msdn.microsoft.com/smartclient/">http://msdn.microsoft.com/smartclient/</a>	Client riche	Windows Smart Client (application développée sous Visual Studio). Limité à la plate-forme .NET
<b>XAML</b> Microsoft <a href="http://www.xaml.net/">http://www.xaml.net/</a>	Client riche	Langage de description d'interface pour la future version de Windows
<b>Xforms</b> W3C <a href="http://www.w3.org">http://www.w3.org</a>	Client riche	Découpe un formulaire XHTML en trois parties, le modèle XForms, les données d'instance et l'interface d'utilisateur
<b>XUI</b> <a href="http://xui.sourceforge.net/">http://xui.sourceforge.net/</a>	Client riche	Framework de développement Java et XML
<b>XUL</b> Mozilla <a href="http://xul-fr.org/">http://xul-fr.org/</a>	Thin Client si base Firefox	XML User Interface (zool), fonctionnant avec un navigateur Mozilla ou Firefox ou avec XulRunner

■ Xavier Leclercq - [Xavier.Leclercq@programmez.com](mailto:Xavier.Leclercq@programmez.com)

### Exemples de configurations matérielles pour des serveurs

	GNU Linux	Windows Server	MacOS X Server
<b>Serveur Web</b>	Apache Serveur dédié 1 Go ram (mini) 1 DD de sauvegarde	IIS ou Apache Serveur dédié 1 Go ram (mini) 1 DD de sauvegarde	Apache Serveur dédié 1 Go ram (mini) 1 DD de sauvegarde
<b>Serveur audio / video (streaming)</b>	Serveur dédié 2 Go ram (mini) 1 DD de sauvegarde	Serveur dédié 2 Go ram (mini) 1 DD de sauvegarde	Xserve G5 2 Go ram (mini) 1 DD de sauvegarde
<b>Serveur de données</b>	MySQL, Oracle, DB2...	MySQL, Oracle, DB2, Sybase, 4D, FileMaker, SQL Server...	MySQL, Oracle, 4D, FileMaker...



# 2 DÉVELOPPER SON SITE WEB

*Page dynamique, page statique, PHP, JSP, ASP, HTML, voilà ce qui revient souvent. Aujourd'hui, les sites web ont besoin d'être flexibles, réactifs. Il n'est plus possible de coupler fortement, présentation, données et fonctionnel. Chaque couche doit être découplée afin de rendre l'application web la plus indépendante possible et en faciliter la maintenance et l'évolution.*

L'adoption d'une architecture de développement multi couche est un élément important. On retrouve le modèle de conception des applications desktop ou client-serveur. De plus, le fait d'avoir une couche de présentation indépendante permet de pouvoir s'adapter aux utilisateurs, aux terminaux. Cependant, il ne faut pas pécher par excès. Si précédemment on avait recours à des versions de sites spécifiques, aujourd'hui, on cherche plutôt à éviter cela. Question de rationalité du développement et de la maintenance. Une des questions essentielles est de savoir comment mon site doit prendre en compte les nouvelles technologies et les nouveaux terminaux, sans pour autant tout casser ou réécrire une grande partie du code.

## Utiliser les vrais standards

Cela passe par le refus de faire des versions spécifiques et de coder les pages spécifiquement pour des navigateurs. Utiliser au maximum les standards. Bien entendu, on ne résout pas tout, mais le standard offre un minimum de garanties. Et surtout, vous savez que les navigateurs s'appuient sur les mêmes technologies. Par contre, pour un intranet, voire un extranet, si vous maîtrisez le poste client, vous pouvez, à la rigueur, opter pour du spécifique, ou du client riche.

Il faut aussi se méfier, ou tout du moins être prudent, face aux modes et aux technologies que l'on présente comme nouvelles ou incontournables. Ainsi, prenons XUL, cette technologie permet de découpler la présentation du code web proprement dit. Un langage intéressant et puissant, mais malheureusement limité à quelques navigateurs. Internet Explorer ne le supporte pas.

## Ayez un code strict, structuré et propre

Comme dans toute application, la qualité du code web conditionne beaucoup de choses : le

bon fonctionnement du site, les performances, la compatibilité avec les navigateurs, les évolutions futures, la charge serveur, etc. Le navigateur est souvent permissif et un code même mal écrit fonctionnera, c'est aussi la faute de l'IDE. Vous pouvez opter pour un codage strict, possible en HTML ou en XHTML. Dans ce cas, l'interface se déporte dans le CSS. De plus, en HTML Strict, les contenus se placent dans les

éléments de blocs. De nombreuses balises sont ainsi interdites d'usage.

## Pour l'interface

Comme vu plus haut, il faut, au maximum découpler interface et fonctionnel. En utilisant XHTML, pour l'interface, il faut passer par le CSS. Standard du Web, il s'impose dans la conception d'un site. Il évite d'occuper trop de

## JSF : un complément aux pages JSP

JSF est un framework technique côté serveur en Java. Il se dédie à la conception d'applications web dynamiques. Il se compose de deux éléments distincts : un ensemble d'API pour représenter les composants d'interface et gérer les états, les événements et la validation des entrées, définir la page de navigation et le support de la location et de l'accessibilité et un custom tag library de type JSP pour exprimer une interface JSF dans une page JSP. JSF passe par une page JSP pour être vu côté client. La qualité de JSF est de pouvoir réutiliser les composants d'interface que les outils proposent et qui sont en standard dans ces spécifications.

JSF a pour but de simplifier le développement, la gestion et le déploiement des applications web dynamiques, tout en offrant une bonne indépendance des pages par rapport au poste client. Contrairement à Struts, JSF est un standard du JCP. Comme Struts, JSF se base sur le modèle MVC. Struts est à la base plus complexe et plus complet que JSF. On ne retrouve pas en JSF par exemple le framework de gestion de couches, Tiles. De plus, sur la partie validation, le framework struts est plus complet et évolué. Cependant, JSF est plus flexible et extensible, avec un modèle de réutilisation.

JSF sépare strictement la définition du composant, du rendu. Il est possible d'afficher les composants de différentes manières et ce, sur différents clients. Struts n'a pas de notion de composant côté serveur. Avec JSF, il est possible de créer des interfaces riches (de type rich client). Les principaux IDE du marché supportent désormais les JSF (Jbuilder, Sun Java Studio Creator, etc.). En conception RAD, JSF simplifie largement la conception de pages JSP, ce qui n'était pas toujours chose aisée. Une application JSF fonctionne dans une servlet (bref sur le serveur). Et contient : les composants javabeans (contenant les données et les fonctions de l'application), la liste des événements, les pages JSP, les custom tag libraries, le fichier de configuration de l'application. Si vous utilisez déjà Struts, il n'est pas utile de le remplacer. Pour un nouveau projet web, en revanche, la question peut se poser. Les éditeurs poussent fortement JSF en lieu et place de Struts. Le fichier de configuration JSF est un fichier XML (ex. : faces-config.xml). Cela fait une technologie de plus à apprendre, mais si vous restez en environnement Java...



■ F.T.

bande passante. Un seul fichier peut contenir la mise en page d'un site entier. CSS remplace donc la présentation par HTML. Le CSS évite les mises à jour lourdes et peu pratiques. Ainsi, quand on ajoute de nouvelles pages, inutile de refaire tout le balisage. La maintenance et l'évolution du site s'en trouvent simplifiées. On sait où aller pour modifier, supprimer, ajouter des éléments, une gestion, du contenu. Il est plus facile de structurer son site ainsi et des pages, en ayant deux parties bien claires et séparées.

Le DOM (Document Object Model) permet de manipuler le contenu d'une page (le document). Il donne une représentation structurée (orienté objet) des éléments et du contenu, avec indication des méthodes pour modifier les objets. On peut ainsi créer une interactivité entre les éléments (ex : faire des tris), sans que le serveur soit sollicité. Les navigateurs récents le supportent. Et il facilite l'utilisation de JavaScript, on se base sur DOM et non sur l'implémentation JavaScript des navigateurs. DOM est disponible en deux niveaux. Le niveau 2 apporte notamment les namespaces XML. Le niveau 3 est en cours de spécifications. DOM se pilote et se développe avec Java, JavaScript, VBScript.

## Prévoir l'accessibilité de son site

L'accessibilité devient une fonction indispensable dans les sites. Il s'agit de rendre disponibles, visibles les pages et les contenus, à tout le monde. L'accessibilité d'un site ne nécessite pas de créer une version spécifique (excepté pour des technologies très spécifiques). Cet accès doit être pris en compte dès le début de la conception. Idéalement, il faut considérer les recommandations WAI du W3C.

Vous pouvez ainsi utiliser l'attribut alt, à condition de remplir celui-ci manuellement pour être explicite. Alt permet de mettre en place des textes de remplacements (à des images). Pour les vidéos, vous pouvez inclure des sous-titres dans les séquences (là, il faut travailler à la source du fichier). Flash MX inclut par défaut certaines fonctions d'accessibilité. Prévoyez une navigation et une validation sans la souris.

■ François Tonic

## xHTML : le langage du web nouveau

**S**i HTML demeure la base du Web, le XHTML est préférable à bien des égards. XHTML est du HTML à la sauce XML. L'avantage est la cohérence et la structuration du code et des balises, ce que HTML ne permet pas. Le fait d'utiliser XML favorise à terme une interaction poussée entre les différents dialectes web du XML (en théorie). Le XHTML remplace HTML 4. Et il travaille sans problème avec DOM et CSS. Vous avez plusieurs niveaux de XHTML :

- XHTML Transitional : DTD un peu rigide mais souple, idéal quand on migre de HTML à XHTML
- XHTML Strict : DTD rigide, structuration du code stricte
- XHTML Frameset : permet d'inclure des cadres dans ses pages.



Exemple XHTML :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>XHTML Quicktime Object</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<style type="text/css">
<!--
/* object stuff here */

/* hides the second object from all versions of IE */
* html object.mov {
    display: none;
}

/* displays the second object in all versions of IE apart from 5 on PC */
* html object.mov/**/ {
    display: inline;
}

/* hides the second object from all versions of IE >= 5.5 */
* html object.mov {
    display/**/: none;
}

/* just stuff to make the page pretty */
body, h1, h2, h3, h4, h5, h6, p, ol, ul, dl, input {
    font-family: Verdana, Tahoma, sans-serif;
}

```

Attention : XHTML hérite du XML pour sa rigueur de déclaration. Soyez attentif à ce que vous écrivez, dans le cas contraire, le document ne sera pas valide. Le balisage est strict, ainsi en XHTML, toute balise ouverte doit être fermée, ce qui n'est pas forcément le cas en HTML. Dans le cadre d'un site web en XHTML, déporter le plus possible la couche de présentation dans le CSS. XHTML se concentre sur le document.

Si HTML continue à être supporté et largement utilisé, pourquoi passer à XHTML ?

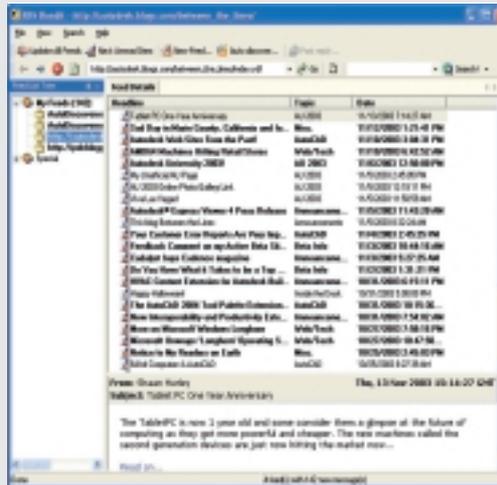
- il remplace HTML 4 et possède une structure plus moderne, plus cohérente,
- la v1 est une version de transition vers les futures versions
- les navigateurs anciens ou modernes fonctionnent avec XHTML.
- il offre un comportement plus prévisible que le HTML.
- on pousse à une meilleure structuration du code Web et à l'emploi du CSS.
- utilisation d'outils de validation par des services dédiés sur le web, réduisant ainsi le temps de tests.

■ F.T.

## Le flux RSS : la syndication de contenu



**F**lux, fil, canal, on utilise toutes sortes de termes pour désigner l'utilisation de RSS. RSS, pour faire simple, permet de faire de la syndication de contenu entre sites web. Il repose sur le langage XML. On peut créer ou intégrer un flux RSS manuellement ou automatiquement via des outils. Préférez l'approche outil afin de vous faciliter la vie. Manipuler des données et balises XML n'a jamais été simple et c'est une perte de temps. Les sites de blogs proposent souvent ce genre de fonction. Quand on insère un flux RSS dans son site, on utilise un lecteur RSS. Les dernières générations de navigateurs l'intègrent. Un flux RSS est



un simple fichier texte au format XML. Comme on est dans un contexte XML, pour pouvoir afficher correctement le flux sur un site, on doit disposer d'un parseur qui s'occupera de réaliser la transformation. On passe alors le plus souvent par un script pour opérer la transformation, par exemple avec MagPie RSS. Si un flux RSS est avant tout du texte, il est possible d'enrichir le contenu avec du multimédia. Pour ce faire, on utilisera la balise <enclosure> (en RSS 2). Cependant cette balise n'est pas supportée par tous les lecteurs. Pour aller plus loin :

<http://blogs.law.harvard.edu/tech/rss>

■ F.T.

## DHTML

**L**e DHTML (XHTML + JavaScript + CSS) convenablement programmé est déjà une très bonne solution pour développer une application Web, le développeur peut se concentrer dans un premier temps sur la partie XHTML (xhtml-strict) sans tenir compte de l'aspect visuel. Le XHTML est aussi plus rigoureux (les balises doivent être fermées, sensibilité à la casse, imbrication correcte, etc.).



Exemple de code :

Définition de la partie XHTML (la structure) avec une dtd selon le modèle XHTML-strict, un encodage en langue française, un menu et une définition de style "stylesheet" :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr"
lang="fr">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-
8859-1" />
<title>Cette page est en xhtml-strict</title>
<link rel="stylesheet" type="text/css" href="style.css" media="screen"
/>
</head>
<body>
<h1>
<img src="" alt="Titre du site"/>
</h1>
<div id="menu">
<h3>rubrique 1</h3>
<ul>
<li><a href="#" title="">lien</a></li>
<li><a href="#" title="">lien</a></li>
</ul>
```

```
<h3>rubrique 2</h3>
<ul>
<li><a href="#" title="">lien</a></li>
<li><a href="#" title="">lien</a></li>
<li><a href="#" title="">lien</a></li>
</ul>
</div>
<div id="contenu">
<h2>blaba blaba titre</h2>
<p>blaba blaba texte</p>
</div>
<div id="pied">
<ul>
<li><a href="#" title="">lien</a></li>
</ul>
</div>
</body>
</html>
```

### LES PLUS

- Fonctionne avec tous les navigateurs récents
- Réduit la charge côté serveur
- Contenu indexable par les moteurs de recherches si DHTML, contenu non indexable si AJAX (XMLHttpRequest)

### LES MOINS

- Difficile d'intégrer du contenu vidéo ou audio
- Le JavaScript est parfois désactivé sur les navigateurs, mais l'application doit pouvoir toujours tourner en raison du critère d'accessibilité
- Difficulté de programmer des effets complexes en tout genre
- Difficulté de sécuriser l'application car la logique d'une application DHTML est accessible par le navigateur

■ Xavier Leclercq

## JSP (JavaServer Pages) : site Web dynamique avec connexion à un serveur de données

JSP a été conçu avec, comme cahier des charges, la portabilité et l'extensibilité (taglib). JSP n'exige ni système d'exploitation ni serveur http particulier. Si vous devez à la fois développer un site Web dynamique sous Linux ou Macintosh et sous Windows, le JSP est un choix logique. Le serveur JSP crée et compile une servlet en arrière plan. Celle-ci est chargée et exécutée automatiquement dans le but de générer le contenu d'une page HTML/XML, qui est alors renvoyée au client. JSP embarque des morceaux de code, et cette insertion directe de code permet d'obtenir une page dynamique. Chaque bloc, appelé "scriptlet" débute par une balise d'ouverture et une balise de fermeture avec une série de variables prédéfinies comme request (un objet HttpServletRequest), reponse (un objet HttpServletResponse), out (un objet PrintWriter pour les sorties), et in (un objet BufferedReader pour les entrées). Il y a aussi page (la servlet elle-même), pageContext (une instance PageContext qui contient les données associées à la page entière) et session (la session HTTP, qui peut conserver de l'information à propos d'un utilisateur par exemple). Les API JSP séparent la génération du contenu (logique applicative) de la présentation de celui-ci, ce qui permet à une servlet de générer du contenu et de le stocker dans le contexte d'une requête.



Voici comment peut se présenter le classique "Hello World" :

```
Hello.jsp
<HTML>
<HEAD><TITLE> Hello World !</TITLE></HEAD>
<BODY>
<H1>
<%
if request.getParameter("name") == null {
    out.println("Hello World !");
}
else {
    out.println("Hello " + request.getParameter("name"));
}
%>
</H1>
</BODY>
</HTML>
```

Le deuxième accès sera plus rapide, car le serveur aura déjà compilé une première fois la servlet. Mais si le contenu de votre page .jsp a été modifié, le serveur devra recréer et recompiler celle-ci. En effet le moteur de JSP vérifie si la date du fichier .jsp correspond à celle du fichier .class. Le moteur de JSP ne transforme et compile la classe que dans le cas où le script JSP a été mis à jour. La création de la servlet consiste à passer du code mixte HTML/XML du fichier d'extension ".JSP" à du code java. JSP peut faire partie d'une architecture 3-tiers, ce qui signifie qu'un serveur applicatif supportant les Java Server Pages pourra servir d'intermédiaire entre le navigateur du client et une base de données (le serveur de données proprement dit). JSP fournit la syntaxe nécessaire à la connexion au serveur de données, ainsi qu'à la manipulation de celles-ci via le langage SQL. Retenons que les JSP sont aussi multithreadées, portables, orientées objet et sûres.

■ Xavier Leclercq - [Xavier.Leclercq@programmez.com](mailto:Xavier.Leclercq@programmez.com)

## Choisir sa base de données

Deux bases se distinguent : les SGBDR "serveur", permettant généralement des connexions par un quelconque protocole réseau : Microsoft SQL Server, Firebird, PostgreSQL, MySQL et Oracle sont parmi les SGBDR "serveurs" les plus utilisés. En matière d'architecture, ces SGBDR permettent de séparer concrètement le stockage des données du reste de l'application (serveur d'application, clients). La source de données n'est donc pas forcément sur la même machine que l'application cliente. Les SGBDR reposant sur un système de fichiers embarquant le moteur de base de données, comme Microsoft Access ou SQLite, ne permettent pas cette séparation. Ils conviennent tout à fait à des applications autonomes, pour lesquelles la source de données est déployée sur la même machine que l'application. On peut se servir du SGBDR comme d'un simple conteneur de données auquel l'application adressera des requêtes pour extraire, supprimer, modifier ou insérer des enregistrements. On peut utiliser les fonctionnalités des procédures stockées et des triggers disponibles dans certains SGBDR pour assurer une partie du code métier de l'application. Il s'agit alors de développer des fonctions dans un langage supporté par le SGBDR, dont l'exécution sera prise en charge par le SGBDR lui-même et non par l'application. Les triggers sont conçus de la même façon et sont déclenchés automatiquement sur certains événements. Tous les SGBDR n'offrent pas ces fonctionnalités. A noter également que certains produits, s'ils supportent tous le langage de requête ANSI SQL 92, enrichissent ce dernier de leurs propres fonctionnalités. Les puissances de ces systèmes de bases de données ne sont pas équivalentes. Deux critères de performances sont à prendre en compte : la vitesse d'exécution des requêtes et la capacité du serveur de base de données à supporter des connexions simultanées. Certains SGBDR voient leur performance s'amenuiser au fur et à mesure de l'augmentation du volume stocké, d'autres sont robustes et performants et sont donc utilisables au sein de projets complexes, traitant de grandes quantités de données. De tels SGBDR sont utilisés, par exemple, dans des applications de business intelligence et servent de support à d'imposants Datawarehouses ou à d'agiles Datamarts. Si l'optimisation des performances est d'abord à la charge du DBA qui devra configurer correctement le SGBDR, il demeure que le développeur devra prendre soin de ne pas le solliciter inutilement.

■ Gauthier Delamarre

EXEMPLE DE CODE SUR  
WWW.PROGRAMMEZ.COM

## ASP.NET : un bon choix ?

La création d'un projet web, comme tout autre type de projet, demande une analyse préalable de l'existant et des technologies disponibles. Après un rapide passage en revue, le choix s'avère ample et délicat ! ASP.NET, comparativement à d'autres technologies du marché telles qu'ASP 3.0, PHP 4.0 ou PHP 5.0, voir JSP, offre de nombreux avantages non négligeables tant en terme de fonctionnalités disponibles nativement au sein de son Framework .NET qu'en terme de stabilité et de performances.



Les avantages de la technologie ASP.NET, peuvent se représenter au travers de ces premiers points :

- Le choix du langage : VB.NET, C#, J#,
- Le mode compilé : Intermédiaire Language (IL). Augmentant les performances et garantissant une stabilité en production de votre code au travers de la vérification au moment de la compilation.
- Le mode dual de gestion de code : Code Behind ou Code Inline
- L'architecture native du .NET Framework au travers des classes unifiées
- Des contrôles clients riches de validation de données (RequiredFieldValidator, RegularExpressionValidator, ...) et de binding de celles-ci (DataGrid, DataList, DataRepeater, ...)
- Un IDE riche, performant et intuitif qu'est VS.NET 2003 et 2005, mais aussi l'existence d'autres IDE libres tout aussi simples d'utilisation
- Un mode Debug non négligeable
- Une compatibilité ascendante ASP.NET vers ASP.NET 2.0

Comparativement, les défauts seront plutôt qualifiés de défauts de jeunesse et ont été corrigés par le Service Pack ou seront palliés dans la version 2.0 d'ASP.NET. Nous pourrions mettre en avant la faille de sécurité du mode d'authentification par formulaire Web du Framework .Net qui a été rapidement corrigée par Microsoft, ou encore, ce faux problème qu'est la mauvaise gestion des contrôles Validator pour le navigateur FireFox qui, en définitive, n'est pas géré par le Framework .NET au travers de sa signature puisque ce navigateur était non existant au moment de la sortie de la version 1.1. L'adaptation a quant à elle aussi été fournie très rapidement par les équipes de Redmond. Enfin, et si le choix ne semblait pas encore intuitif, ASP.NET se retrouve depuis sa sortie entouré d'un écosystème communautaire foisonnant de tutoriaux, codes ou projets complets utilisables complètement ou partiellement (StarterKits Microsoft, Blogs, Communautés Francophones, ...).

■ **Grégory Renard**  
Directeur Développement  
Wygwam sarl



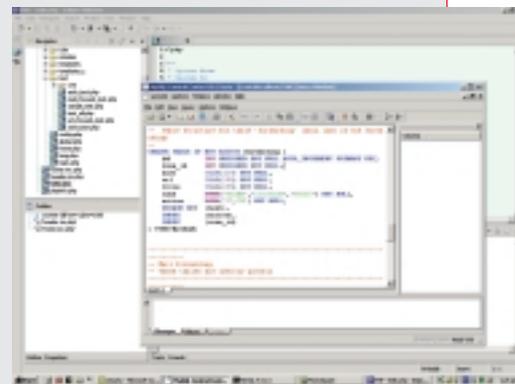
## Pourquoi utiliser PHP sur son site Web ?

La question peut paraître saugrenue, tant PHP semble aujourd'hui s'être imposé sur une grande majorité des serveurs web, comme le langage incontournable pour la réalisation de sites dynamiques. Mais pour autant, d'autres solutions existent, et selon les circonstances, la suprématie de PHP peut être remise en cause.



### Les +

Parmi les avantages incontestablement nombreux de l'utilisation de PHP, on peut commencer par citer sa facilité d'apprentissage. En effet, quelques heures de lectures sur le web, et vous serez prêts à rédiger vos premiers scripts. De plus, l'interpréteur PHP est généralement disponible d'emblée sur les hébergements personnels comme professionnels, ce qui permet de ne pas se préoccuper de sa mise en place (il faut préciser que MySQL est également présent de manière systématique). Ensuite, il y a l'impressionnante importance de la communauté d'utilisateurs, qui permet le plus souvent de trouver des solutions rapides et efficaces aux problèmes que l'on peut rencontrer lors de ses développements. Dans sa dernière série, la 5.x, PHP s'est vu doté d'un tout nouveau moteur offrant un support avancé de la programmation objet, ce qui était l'une des plus grosses lacunes des versions antérieures. Entre autres nouveautés, on ne peut manquer non plus de parler de l'intégration du moteur SQLite, qui tout en travaillant sur des fichiers plats et autonomes (i.e. sans serveur), offre des performances spectaculaires (pour peu que le nombre de connexions concurrentes reste limité). Enfin, le support de la librairie SimpleXML a, comme son nom le laisse entendre, grandement facilité la manipulation de sources de données XML (comme les fameux flux RSS).



### Les -

Ceci étant, PHP n'est pas exempt de défauts, et souffre de certaines faiblesses, comme tout autre langage. Pour un point notamment, PHP a les défauts de ses qualités : le framework de PHP, pour complet qu'il soit, n'est pas fortement structuré. Si la liberté offerte au programmeur expérimenté par ce modèle très souple est généralement très appréciée, elle peut en revanche désorienter le débutant.

L'un des autres points rébarbatifs, généralement constaté auprès des développeurs, concerne le développement en équipe. Il faut reconnaître que les environnements dédiés à ce langage ne sont pas légion. Et parmi ceux-ci, les seuls à proposer une forme de travail collaboratif se reposent sur CVS, qui impose un certain nombre de contraintes qui peuvent à la longue devenir franchement contre-productives.

Enfin, il y a les limites imposées par un existant qui peuvent également écarter PHP du champ des choix technologiques possibles, notamment sur des projets d'importance.

■ **Gauthier Delamarre**

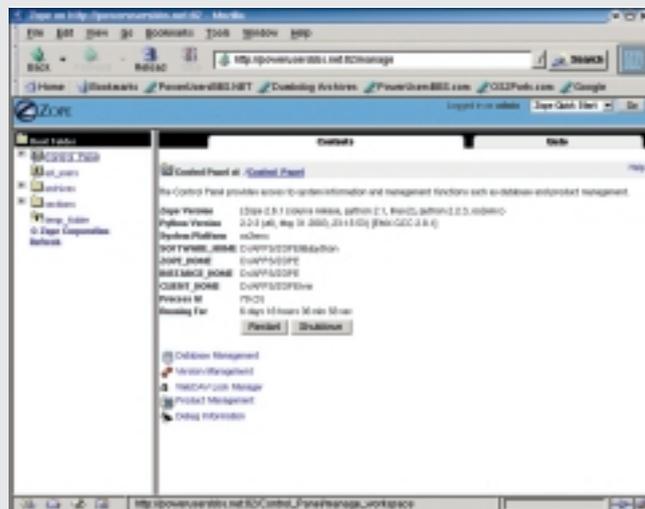
## Zope/Plone, une plate-forme pour les systèmes de gestion de contenu

**P**ourquoi utiliser un CMS (Système de gestion de contenu, SGC en français) ? Pour améliorer la communication commerciale et marketing, pour rationaliser la documentation technique, pour réaliser des économies. L'offre aujourd'hui, aussi bien commerciale que libre, est arrivée à maturité. Certains fournisseurs vendent des licences mais proposent aussi des formule ASP (Application Service Provider) (plus de la moitié des systèmes CMS aux Pays-Bas seraient déjà vendus via cette formule). Un logiciel CMS séparera le fond de la forme et de cette manière, l'utilisateur final se concentrera uniquement sur la rédaction d'un contenu, tandis que le logiciel CMS mettra en page ces informations et les publiera sur Internet ou Intranet. Une entreprise pourra de cette manière réaliser l'économie d'un imprimeur en arrêtant de publier un catalogue annuel "papier et figé" de ses produits. Mieux : le catalogue électronique mis en place par le CMS sera à jour en permanence, et accessible à tous à partir d'un simple navigateur ou gravé sur cd-rom. Il existe aussi des solutions gratuites GPL comme SPIP ou Zope/Plone. Il faut néanmoins distinguer les CMS ne nécessitant pas de base de données (comme Guppy), des portails (PHP-Nuke, XOOPS) et les systèmes de publications (plus orientées entreprises).

### Les avantages du CMS

- Le pouvoir de l'information revient aux détenteurs de celles-ci, et non aux informaticiens, n'importe quel utilisateur sans connaissance technique est capable de mettre en ligne de l'information
- Gestion et encodage automatisé
- Uniformisation du rendu, haute qualité de l'information publiée
- Suivi des versions de chaque document
- Validation, vérification du contenu par des tierces personnes

Zope/Plone permet de construire des sites collaboratifs, c'est-à-dire qu'un utilisateur peut s'identifier et mettre à jour son contenu dans un espace privé. Une fois ce travail de composition achevé, il peut ou non (suivant les règles en vigueur) décider de le partager avec la communauté. Un modérateur pourra aussi éventuellement intervenir pour approuver les changements. Les documents seront automatiquement stockés et indexés dans un catalogue (avec moteur de recherche objet intégré). Contrairement à d'autres CMS qui utilisent PHP, Plone s'appuie sur Python et Zope. Avec Zope le processus de



publication de contenu se déroule en quatre étapes : Identification, Composition, Modération et Publication. Le chemin que doit suivre le document pour être validé est baptisé "workflow", tandis que le "Content Management Framework" (CMF) constitue une boîte à outils de services logiciels. De plus, il respecte les normes d'ergonomie et d'accessibilité (XHTML et CSS).

### Quelques systèmes CMS

**SPIP** : Système de Publication pour l'Internet Partagé (<http://www.spip.net/>)

**SPIP-Agora** : Version enrichie (XHTML) de SPIP développée par le Service d'Information du Gouvernement français (<http://www.agora.gouv.fr/>)

**Plume-CMS** : Système CMS sous licence GNU GPL, rapide, simple et conforme aux standards W3C (<http://pxsystem.sourceforge.net/fr/>)

**My-kiwy** : CMS français pour entreprises (gère les fonds documentaires, mais aussi module de commerce électronique) (<http://www.my-kiwy.com>)

■ **Xavier Leclercq**

[Xavier.Leclercq@programmez.com](mailto:Xavier.Leclercq@programmez.com)

# 3 LE CLIENT RICHE

## Les applications ont trouvé leur modèle

www.programmez.com

- liens
- codes
- dialogue avec les auteurs

*Délaissé par les éditeurs et les utilisateurs au profit du client léger, c'est à dire un pur site Web, le client riche (ou client lourd) revient en force. Et ce, pour plusieurs raisons : besoins d'interactivité, nécessité de traitement et d'une interface complexe, et de tirer avantage des nouveaux langages et environnements. Le client riche s'oppose au client léger en HTML, en dépassant les limites du langage web par excellence.*

Le client riche n'a de sens que dans le cadre d'un intra ou extranet, il concerne uniquement les applications d'entreprises et non le site Internet proprement dit que tout utilisateur peut aller visiter. Dans ce cas, on utilisera plutôt du Rich Media, avec un soupçon de client riche au niveau interface et interactivité Client - Serveur. Le risque du client lourd est de fournir une application difficile à utiliser, d'un coût élevé et finalement, inefficace.

XML et les dérivés ont su dépasser les clients légers basés sur HTML. Même si le client HTML a l'avantage d'être déployable rapidement et partout, il demeure peu convivial et limité. À

l'inverse, le client riche traîne depuis longtemps un coût de déploiement élevé et un besoin en bande passante conséquente. Quand on fait du client riche (et même du Rich Media) mieux vaut s'appuyer sur les players et runtime clients en principe présents sur le poste de l'utilisateur, ou alors qui nécessitent le minimum de téléchargement. Ce socle technique permet de réduire le poids du client riche car, dès lors, les composants du poste client possèdent les fonctions de base du client riche. Le problème est de choisir sa ou ses technologies. Ainsi, pour la description d'interface, les dérivés XML propriétaires ne manquent pas : XUL, XAML, MXML, XDP, etc.

Pour le moment, XUL et MXML se détachent, car ils sont disponibles. XUL est issu de Mozilla et sert déjà dans Firefox. Il peut travailler avec ASP, PHP, Perl, JSP, etc. Dans le monde Java, on dispose désormais d'un socle client riche fiable et reconnu : Eclipse Rich Client Platform. Tout est conçu pour être le plus léger possible. Il s'installe d'un clic sur le poste de l'utilisateur. Attention tout de même à la dépendance technologique et aux incompatibilités du client riche, si vous êtes en environnement hétérogène. Dans ce cas, utilisez une technologie de type Flash / Flex, Java, ou purement XML (CSS, xHTML, Xforms, etc.).

■ François Tonic

### Java web start



Il s'agit d'une plate-forme Java facilitant le déploiement d'applications, basée sur les technologies serveurs Web Java. Il fonctionne avec l'ensemble des navigateurs du marché et permet aux utilisateurs de lancer des applications en toute transparence. Cependant, quand on utilise une nouvelle application, on clique sur un lien URL d'une page permettant de déployer l'application sur son poste (l'application se télécharge, ex : <a href="/products/javawebstart/docs/MyApp.jnlp">Launch My Application</a>). Java web start vérifie à chaque lancement si l'application a été mise à jour. Il s'appuie sur JNPL (attention à la configuration du serveur web pour le fichier .jnpl, le type MIME est : application/x-java-jnlp-file, le serveur renvoyant le MIME adéquat afin de reconnaître les fichiers jnpl). Pour pouvoir créer un client riche Java Web start, vous devez implémenter JNPL. Sur le poste client, Java Web Start joue le rôle d'un lanceur d'applications. Une application de ce type doit être dans un JAR et contenir l'ensemble des ressources. Java web start transfère uniquement un fichier Jar du serveur au client. Grâce à un fonctionnement sandbox (= exécution limitée), l'application aura un accès limité au poste client. Bref, le modèle de développement Java ne change pas, mais l'utilisation d'une application Java est facilitée pour le client. Il est possible de vérifier si le lanceur est déjà présent ou non. Il suffit de passer par un script JavaScript ou VbScript. Exemple d'une détection Java Web Start dans une page HTML :

```
<SCRIPT LANGUAGE="Javascript">
var javawsInstalled = 0;
isIE = "false";

if (navigator.mimeTypes && navigator.mimeTypes.length) {
  x = navigator.mimeTypes['application/x-java-jnlp-file'];
  if (x) javawsInstalled = 1;
} else {
  isIE = "true";
}

function insertLink(url, name) {
  if (javawsInstalled) {
    document.write("<a href=" + url + ">" + name + "</a>");
  } else {
    document.write("Need to install Java Web Start");
  }
}
</SCRIPT>
```

S'il convient aux intra, extranet, applications C/S d'entreprise, Java Web Start n'est guère approprié aux utilisateurs lambda, mieux vaut passer par des solutions transparentes de type Flash. Si vous utilisez Java Web Start en environnement hétérogène sur le poste client, n'oubliez pas de tester le fonctionnement avant tout déploiement. À noter qu'il est possible d'utiliser JNLP dans une archive de type WAR.

En complément : <http://java.sun.com/products/javawebstart/docs/developersguide.html> ■ F.T.

## 3» LE CLIENT RICHE

### EXEMPLE DE CODE SUR WWW.PROGRAMMEZ.COM

## Flex

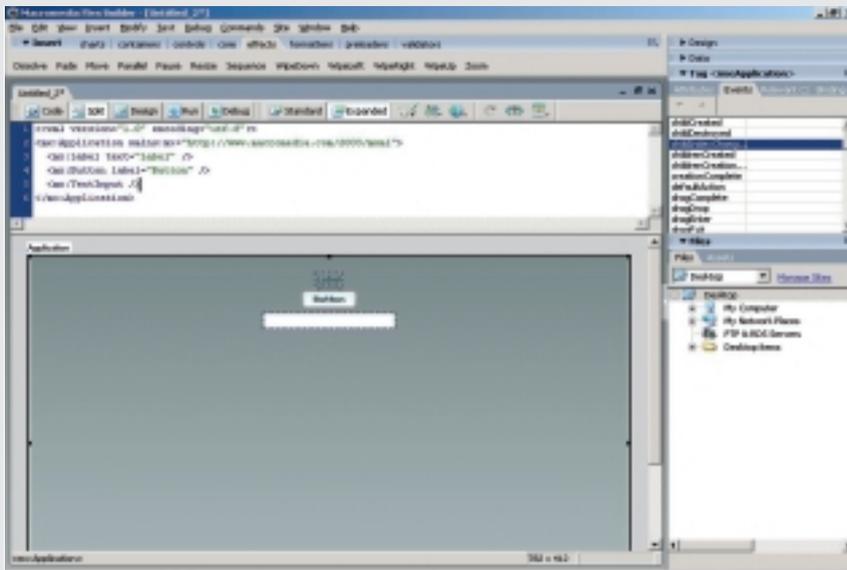


**F**lex est un serveur de présentation, cela ne veut pas dire qu'il permet de faire des présentations à la PowerPoint, mais plutôt qu'il se situe dans la couche présentation d'une architecture N-Tiers. En fait, il reprend le modèle d'exécution des pages ASP ou des pages JSP, c'est-à-dire qu'il apparaît en front, afin de mettre en forme des informations venant des composants métiers se trouvant en arrière de l'application. Seulement, contrairement aux pages JSP ou ASP, Flex intervient sur la mise en forme des informations afin de les envoyer au client au format Flash. Il se positionne en complément d'un serveur d'application J2EE ou .Net qui, eux, sont présents pour faire la couche Métier de l'application, et permet d'ajouter une couche applicative pour communiquer avec le Flash envoyé au client. Ainsi le serveur d'application .Net ou J2EE envoie les informations à Flex qui les transforme et les envoie à Flash. La programma-

tion de Flex se fait en utilisant un nouveau langage, le MXML. Sorti avant le XAML de Microsoft, le MXML permet de faire une description de l'interface utilisateur au format XML à l'aide de balises spéciales qui font partie du schéma MXML. Ainsi, la séparation de code client riche/serveur se fait exactement de la même manière qu'elle se ferait en ASP.Net avec le HTML et le code C#, ici

aurait deux fichiers, un MXML représentant l'interface et un fichier représentant la mise en forme des informations. Se basant sur Action Script 2.0 et MXML, le développement de Flex reprend des standards et des modèles de développement connus par les programmeurs Flash. Flex Builder permet d'améliorer grandement la productivité. De plus, à ce serveur de présentation vient s'ajouter un framework d'utilisation de la plate-forme. Ce framework est composé d'un ensemble de classes permettant de faciliter le développement des applications. Parmi ces classes, on retrouve des classes permettant de mettre en place des composants sur l'interface utilisateur, mais aussi quelques classes permettant de calculer des règles métier.

Les avantages de Flex sont les suivants : il tourne sur un grand nombre de configurations logicielles (Linux, Windows, Unix), il permet la mise en place rapide d'interfaces utilisateur puissantes au format Flash, et le modèle de développement est simple et rapide à mettre en place.



L'inconvénient majeur est qu'il faut que le client ait au moins la version 7 du player Flash, ce qui restreint la pénétration côté client, même si quelque soit le client, IE6, Firefox ou Netscape, le résultat sera identique, sans décalage dans la présentation. En conclusion, Flex est à Flash ce qu'ASP.Net et J2EE sont au HTML.

■ **Xavier Vanneste**

## La nouvelle génération d'interfaces : exemple de XUL

**S**i le navigateur est la plate-forme (en l'occurrence Firefox) il est par exemple possible d'exécuter une application complexe client / serveur sans rien installer. Cette application, qui pourrait être qualifiée de "Thin Client" (si tous les utilisateurs étaient équipés de Firefox) est exécutable telle quelle, de suite. Et ce miracle est dû à XUL (XML User Interface Language, prononcez "zool"). Nous invitons les sceptiques à exécuter l'application de démonstration "Mozilla Amazon Browser" sous Firefox ou Mozilla. De plus en plus de développeurs choisissent, soit le binôme client / serveur Mozilla-XUL/PHP, soit Mozilla-XUL/Java.

L'inconvénient majeur de XUL est qu'il ne peut s'exécuter sous Internet Explorer, mais depuis peu il est devenu possible de télécharger des exécutables autonomes via XulRunner. Il s'agit d'une plate-forme "standalone" d'exécution d'applications, avec laquelle l'utilisateur peut se passer de Firefox pour exécuter un logiciel XUL ! Mieux : des versions ont été compilées avec le support activé de SVG (Scalable Vector Graphics) ! Des projets sont en développement pour créer des IDE XUL, comme XulMaker, XCUBE ou MozCreator.

Concrètement cela donnera quelque chose comme ceci :

```
#xulrunner MonAppli.ini
```

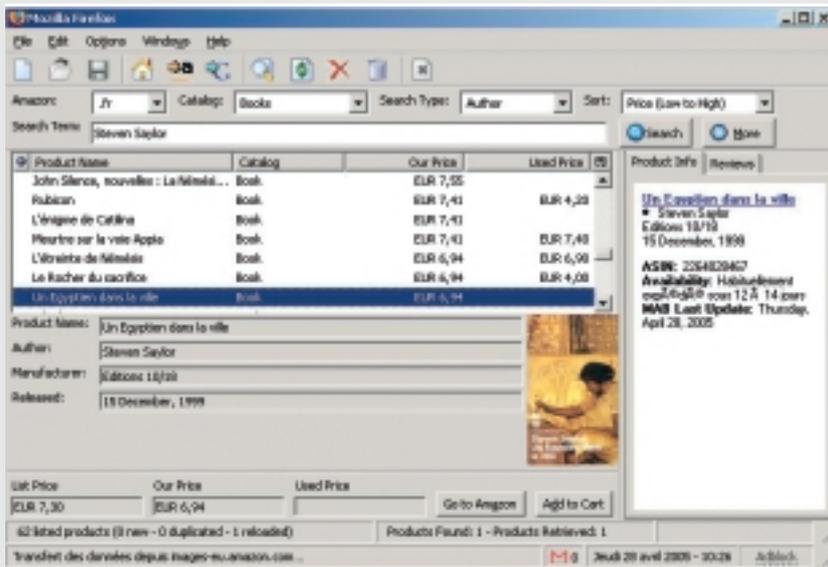
Vous trouverez sur le site de XulRunner le code source complet d'une application nommée

### LES PLUS

- Sous licence Mozilla, compatible avec HTML, XHTML, XSLT, CSS2, DOM2
- Pas besoin d'IDE lourds
- Documentation développeur traduite en français
- Multi-plate-forme
- Si l'application est livrée sous forme de package Xpi (qui peut-être signé pour plus de sécurité), son installation se réalise d'un simple clic (mais avec un redémarrage du navigateur nécessaire)

### LES MOINS

- Incompatible avec XAML de Microsoft
- Courbe d'apprentissage lente si développeur débutant
- Pas d'IDE de développement stable et fiable pour créer rapidement des écrans XUL



"mybrowser", qui comme son nom l'indique permet de naviguer. En gros le code source principal est le suivant :

```
<?xml version="1.0"?>
<?xml-stylesheet href="chrome://global/skin/" type="text/css"?>
<!DOCTYPE window SYSTEM "chrome://mybrowser/locale/mybrowser.dtd">
<window
  id = "mybrowser"
  title = "&mybrowser.title;"
  width = "800"
  height = "600"
  xmlns = "http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">
<script src="mybrowser.js"/>
<hbox>
<button id="back" label="&mybrowser.back;" oncommand="back();" disabled="true"/>
<button id="forward" label="&mybrowser.forward;" oncommand="forward();" disabled="true"/>
<button id="reload" label="&mybrowser.reload;" oncommand="reload();"/>
<button id="stop" label="&mybrowser.stop;" oncommand="stop();" disabled="true"/>
<textbox id="urlbar" value="" flex="1" onchange="go();"/>
<button id="go" label="&mybrowser.go;" oncommand="go();"/>
</hbox>
<browser flex="1" id="browser" src="" type="content-primary"/>
<statusbar>
<statusbarpanel id="status" label="" crop="end" flex="1"/>
<progressmeter id="progress" mode="determined" value="0%" style="display: none"/>
<statusbarpanel id="security" label="" style="display: none"/>
</statusbar>
</window>
```

Avec les fonctions supplémentaires en JavaScript qui servent à réagir aux boutons, comme la fonction "go" :

```
function go() {
  var urlbar = document.getElementById("urlbar");
  var browser = document.getElementById("browser");

  browser.loadURI(urlbar.value, null, null);
}
```

■ Xavier Leclercq

## ASP.Net



L'apparition de contrôles serveurs tels les RequiredFieldValidator, RegularExpressionValidator, embarquant leurs propres scripts de validation, permet maintenant au développeur de se détacher de ces tâches auparavant fastidieuses et souvent laborieuses.

L'apparition de .NET 1.0 a donc été un soulagement pour bon nombre de développeurs et de sociétés pour qui l'interfaçage client riche n'était pas obligatoirement productif et rentable. Force est de constater que ce n'est pas à l'évidence la bonne solution, car une grande partie des processus qui sont maintenant déportés côté serveur s'effectuait auparavant avec quelques lignes de scripts côté client et ce, sans rechargement de la page. L'arrivée du Framework .Net a donc été annoncée par certains comme la mort du JavaScript et des interfaçages clients riches. Cependant, on est bien loin de la fin du ClientSide qui est la clé de la réussite d'une bonne interaction avec l'utilisateur final. Bon nombre de sociétés l'ont compris et se sont spécialisées dans la création de contrôles serveurs riches, embarquant css, javascript et .Net, afin de mettre à la disposition des développeurs des outils paramétrables ne nécessitant aucune connaissance particulière dans le scripting client.

Microsoft est sûrement l'une de celles-ci lorsque l'on voit des applications web du type WebOutlook sous Exchange, WSS (Windows Sharepoint Services) et SPS (Sharepoint Portal Server), ou encore les Blogs MSN Space. Preuve que le ClientSide est bien une des clés de la réussite de votre projet.

L'arrivée de .NET 2.0 est une avancée quant à la mise en avant des clients riches. Cette nouvelle mouture inclut de nouveaux contrôles serveurs intégrant leurs propres fonctionnalités clientes, comme le Menu dynamique, le WebPart Catalog qui permettra à l'utilisateur final de paramétrer sa mise en page par simple " glisser-déplacer ", le gestionnaire de Skins qui permettra la modification visuelle d'une interface en quelques clics, ou encore le ClientCallback qui permettra de ne recharger qu'une partie d'une page. Ceci est donc prometteur pour l'avenir et l'apparition de définitions comme XAML nous laisse présager l'arrivée d'une plateforme .Net ne se souciant plus du type de présentation de votre projet (webform ou winform), mais plutôt de l'interaction riche avec l'utilisateur final.

■ Aurélien Verla - *Chef Projet Wygwam sarl* [[www.wygwam.com](http://www.wygwam.com)] Microsoft MVP Visual Developer ASP - ASP.NET



Blog : <http://blogs.developpeur.org/aurelien> - Auteur de nombreux articles .NET sur [www.asp-php.net](http://www.asp-php.net)

- liens
- codes
- dialogue avec les auteurs

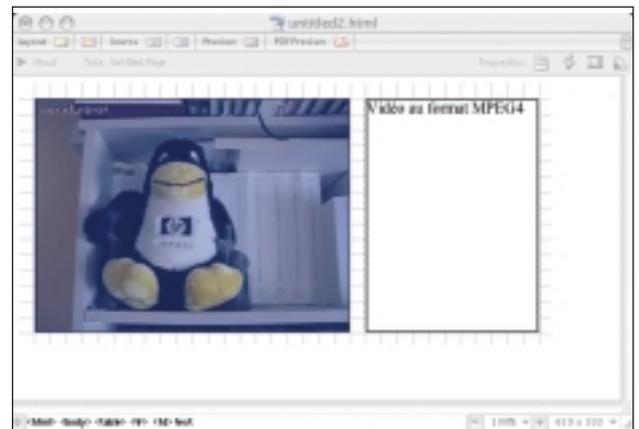
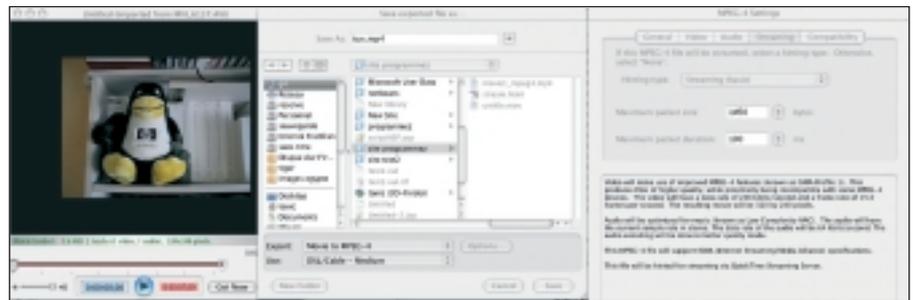
# 4 LE " RICH MEDIA "

## Quand le Web devient réellement interactif

*Rich media? Sous ce vocable barbare se cache un site web plus visuel, plus riche. Il existe de nombreuses possibilités pour enrichir un site et ajouter des éléments multimédia. La technologie Flash est sans doute la plus connue dans ce domaine.*

Le Rich media répond à un besoin croissant d'une interface web de plus en plus riche, avec du multimédia (animation, streaming, fichiers audio ou vidéo). Le Rich media est en quelque sorte un client lourd, orienté utilisateur. Faire du Rich media nécessite des développements spécifiques. Actuellement, la technologie Flash représente le standard pour réaliser un site Rich media.

Bien que relativement exigeant en bande passante et en programmation (langage propre), ses possibilités sont sans limite ou presque. Cependant, d'autres solutions moins "lourdes" sont possibles : xHTML avec CSS, ASP.NET 2, XUL, JSF. Il faut tout de même rester le plus standard possible et savoir quelles technologies sont déjà installées sur une majorité de navigateurs.



### Streaming Audio – Video

Le streaming audio / video nécessite une infrastructure matérielle et logicielle adéquate. À la différence d'un fichier de chargement d'un fichier ou d'un Fast Start, le pur streaming utilise un flux serveur sur une vidéo qui ne se charge pas sur le poste client, mais est exposée par le serveur.

Dans le cas d'un streaming audio – video, il faut disposer d'un serveur dédié configuré. Il s'agit de technologies lourdes, nécessitant un serveur vélocité et une bande passante dimensionnée au nombre de connexions. Il existe trois grandes technologies de streaming : QuickTime d'Apple, Windows Media de Microsoft et REAL Audio – Video de REAL Network. À cela s'ajoutent des solutions open source ou non, telles que VideoLAN Server, ou encore le DivX. VideoLAN Server fera du MPEG

1, 2 et 4. QuickTime Streaming Server fera du QuickTime et du MPEG-4. Bien entendu, on ne broadcaste pas une forme vidéo ou audio sans traitement. Il faut d'abord encoder la séquence dans le bon format et le débit voulu. Chaque format a ses impératifs de flux, afin d'obtenir une lecture fluide sur le poste client. On utilisera alors tel ou tel algorithme d'encodage ou de compression, mais dans ce cas, il faut que l'utilisateur possède le bon player. Une fois la séquence calibrée, il faut la publier sur son serveur vidéo. Dans le streaming, il existe plusieurs protocoles de transport, les principaux étant RTP et RTSP. Vous pouvez aussi réaliser du streaming live. C'est à dire, publier sur le Web une vidéo en direct, d'un événement. L'architecture demeure la même que pour une séquence locale streamée, mais on doit utiliser un outil spécifique pour diffu-

ser du live (ex. : QuickTime Broadcaster). Il faut ensuite définir au niveau administration serveur, le nombre de connexions et les flux autorisés. Attention, si vous autorisez trop de broadcast simultané avec un serveur non adapté, vous risquez une sévère perte de performance, ou carrément un effondrement des serveurs vidéo et Web, incapables de gérer les délais.

Intégrer une séquence non streamée est simplissime. Il suffit de créer un objet audio / video, de choisir le type de lecteur et le format, puis d'indiquer le fichier (préalablement déposé dans le site). C'est tout ! Le plus difficile est donc l'encodage / conversion. Pour les sites de petite taille, privilégiez le MPEG.

■ F.T.

## Laszlo : le Flash open source

Avec Flash, nous entrons de plain pied dans le concept de client riche. Et l'Open Source dans tout ça ? Du côté d'Adobe, il existe bien l'initiative Open Source "Adam" et "Eve", deux bibliothèques qui facilitent la spécification des processus d'interface homme-machine, sinon il y a le framework dont on parle énormément en ce moment : Laszlo. L'éditeur de solutions Laszlo Systems a en effet lancé un environnement de développement Flash (Flash 5, Flash 6 en bêta) pour client riche en open source : OpenLaszlo. Concrètement, il

```
6-7855.
<canvas>
  <dataset name="dset">
    <employee>
      <firstName>John</firstName>
      <lastName>Smith</lastName>
      <phone>617-536-7855</phone>
    </employee>
  </dataset>

  <text datapath="dset:/employee/firstName/text()" />
  <text datapath="dset:/employee/lastName/text()" />
  <text datapath="dset:/employee/phone/text()" />

  <simplelayout axis="x" />
</canvas>
```

Vous pouvez aussi évidemment réaliser des animations, visionner une vidéo, écouter une bande sonore etc.

### LES PLUS

- 96% des navigateurs clients en Europe auraient un plug-in Flash et Flash s'affiche de façon identique sur les différentes plateformes
- Laszlo est Open Source
- Flash manipule aussi le XML, ce qui le rend plus inter-opérable
- Intégration avec les services Web possible, ainsi qu'avec Java, par exemple, (JSPn PHP, ASP, etc.)
- Le code MXML de Flex peut être stylisé avec CSS
- Il est simple d'intégrer du contenu audio et vidéo
- Le langage ActionScript qui accompagne Flash est un langage de scripts basé sur EcmaScript/Javascript.

### LES MOINS

- Le serveur Laszlo n'est pas encore ultra performant
- Le contenu Flash n'est pas indexable par un moteur de recherche

■ Xavier Leclercq

s'agit d'un framework de développement (constitué du langage et du serveur) qui permet à un développeur de réaliser des interfaces riches sans avoir à maîtriser des technologies comme Flash ou Shockwave. Laszlo repose sur une technologie baptisée LZX (elle-même fondée sur XML et Javascript) assez proche de XUL. Comme des codes sources valent mieux que des grands discours, en voici :

```
<canvas>
  <text>Hello World!</text>
</canvas>
```

Affichera "Hello World!"

```
<canvas>
  <viewresource="cd_cover.jpg" />
</canvas>
```

Affichera l'image cd\_cover.jp. Enfin le code source

suivant affichera John Smith 617-53

## Vidéo dans Flash

Flash est un bon outil d'animation et certainement le seul qui soit performant sur un réseau à bande passante limitée comme Internet. Le problème, c'est que pour atteindre cette performance il utilise le vectoriel. Or, imaginez une annonce publicitaire, un film, ou une visite virtuelle en vectoriel ! Pour cela, il faut se tourner vers la vidéo comme Avi, Mpeg, Mov, WMV ou RM. Flash peut intégrer tous ces formats en utilisant le lecteur et les codecs qui se trouvent sur le poste client de l'utilisateur. Ainsi le problème de la multi diffusion sur Internet et de pouvoir toucher tout le monde fait qu'on doit avoir plusieurs formats de vidéo et laisser l'utilisateur choisir, car il n'a pas forcément Windows media player ou Quicktime, et même s'il a tous les lecteurs qui existent, il n'a pas forcément installé DivX. Avec Flash, on dispose d'un format vidéo adapté à la multi diffusion, le format FLV (Flash Vidéo). L'avantage de ce format est que le lecteur est Flash et le codec, Sorenson MP4. Ainsi, le seul logiciel dont le client a besoin c'est Flash Player. Reste le problème du streaming. Lorsqu'on diffuse une vidéo sans passer par un serveur de streaming, le client doit en télécharger



une grosse partie sur son poste pour la lire (buffering), la vidéo qui lui est envoyée n'est pas forcément adaptée à sa bande passante (ce peut être une vidéo trop grosse, mais de bonne qualité pour la bande passante du client, ou au contraire de qualité médiocre mais de faible poids alors que le client est en ADSL), de plus, la vidéo se trouve entièrement sur le poste du client, dans le cache du navigateur Internet, lorsque celui-ci a fini de la visionner, ce qui n'est pas forcément préférable pour des sites de diffusion avec restriction sur la propriété morale. Pour cela, il y a la possibilité d'utiliser le serveur "Flash Communication Serveur" afin de faire du streaming. Les avantages du FLV couplé à Flash Communication Serveur résident dans la possibilité de faire de la vidéo interactive. On peut facilement imaginer faire, par exemple, une visite filmée d'un musée, et lorsque le client clique sur un des tableaux du film, toutes les informations du tableau apparaissent sur un encart à côté, seule l'intégration du format FLV et Flash permettent de faire cela.

### Code pour afficher une vidéo

```
var connection_nc:NetConnection =
new NetConnection();
connection_nc.connect(null);
```

```
var stream_ns:NetStream = new
NetStream(connection_nc);

my_vid.attachVideo(stream_ns);
stream_ns.play("tom_greeting.flv");
```

```
No_btn.onRelease = function() {
  stream_ns.play("tom_goodbye.flv");
}
Yes_btn.onRelease = function() {
  stream_ns.play("Crockard.flv");
}
```

### Code pour réaliser du streaming video

```
var connection_nc:NetConnection =
new NetConnection();
connection_nc.connect(rtmp://fcs_
server/app_name/instance);
```

```
var stream_ns:NetStream = new
NetStream(connection_nc);
```

```
my_vid.attachVideo(stream_ns);
stream_ns.play("tom_greeting");
```

```
No_btn.onRelease = function() {
  stream_ns.play("tom_goodbye");
}
Yes_btn.onRelease = function() {
  stream_ns.play("Crockard");
}
```

Exemples d'après Macromedia

■ Xavier Vanneste

# 5 TESTS ET OPTIMISATIONS : RENDRE UN SITE UTILISABLE

www.programmez.com  
 • liens  
 • codes  
 • dialogue avec les auteurs

*Avant de penser à publier le site, il faut passer par une étape incontournable : les tests et les optimisations. Pour éviter les blocages, les pages d'erreurs, les problèmes serveurs, il faut se résigner à tester.*

Le débogage des pages Web ne se réalise pas à la fin du projet. Si vous utilisez des IDE Web, vous avez à votre disposition des modules de débogage selon le niveau et le type de langage et selon les spécifications des navigateurs. Cette première étape se met en place dès les premières pages conçues. Si vous utilisez CSS, vous devez disposer aussi d'un débogueur. N'hésitez jamais à utiliser des outils de validation officiels. Le W3C propose en ligne un validateur de code HTML / XHTML. Ces outils permettent de tester rapidement la conformité du code d'une page. En cas d'erreurs, le validator indique l'ensemble des erreurs et avertissements.

La permissivité de certains langages et IDE, notamment sur HTML, donne un code surchargé, mal structuré, des problèmes de respect du standard, etc. Si les IDE Web permettent de générer un code automatiquement, il faut s'en méfier. Respecter au plus près les balises "modernes", supprimer les anciennes balises. Il faut donc : rationaliser le code, éviter la redondance, séparer les couches. L'usage de CSS permet d'avoir des interfaces types réuti-

lisables à volonté dans des pages. Dans le cas de pages PHP, JSP, ASP.NET, XHTML, vous pouvez opter pour des outils très complets intégrant du tuning, du profiling, du monitoring, du debug, etc. Le choix est vaste : Zend, PHPEdit, les outils XML / XSLT, Visual Studio Web Developer, Sun Java Studio Creator, etc. Les outils de tests ne manquent pas ! Quelques outils du marché :

- WebKing (Parasoft) : outil de test automatique pour les applications web critiques avec analyseur des statistiques, test de régression et fonctionnel.
- Web Performance Trainer : outil de montée en charge et de stress, avec analyse http / https.
- E-test (empirix) : gamme d'outils de tests pour les applications web (gestion de tests, tests de régression et fonctionnels, de stress, de montée en charge...).

Dans certains cas, optez pour la compression des pages. Car, plus les pages sont réduites, plus elles sont performantes et moins le serveur se surcharge durant les transferts. Très utile pour un site mobile. N'hésitez pas à faire tester l'interface de votre site par des utilisateurs - testeurs. Ainsi, vous pourrez optimiser la navigation et mieux adapter les pages. Pour restructurer le code plus logiquement, il est possible d'utiliser les éléments div et/ou span. Par exemple, div divise une page en blocs. Pour structurer encore plus le code HTML et XHTML, utilisez l'attribut id. Celui-ci vous imposera des règles de déclaration et d'utilisation très strictes. De plus, vous pouvez combiner éléments div / span et attribut id.

Sur les pages	Conformités du code Comportement sur les navigateurs Affichage sur les différents terminaux Génération et affichage page dynamique Charge réseau Régression et bugs Optimisation des images Temps de réponses
Sécurité	Sécurité des pages et données sensibles Utilisation des protocoles sécurisés
Serveur	Montée en charge Charge réseau Répartition de la charge Fonctionnement du code behind Comportement des serveurs (web, données...)

■ F.T.

## Gérer le cache de ses pages en PHP



Il y a quelques années, le concepteur moyen pensait à tort que plus son site serait riche visuellement, plus il recevrait de visiteurs. Aujourd'hui, si le poids des pages continue d'augmenter en même temps que la rapidité des transferts, on pense heureusement plus en terme d'accessibilité pour tous qu'en terme de richesse visuelle.



La preuve a été faite que l'on déserte très vite un site jugé trop lourd et trop lent. Bref, lorsque les pages d'un site sont lourdes, gérer le cache devient vite une nécessité.

En ce qui concerne la page statique HTML le développeur peut jouer directement sur le mécanisme d'échange HTTP. Lorsqu'un navigateur demande une page avec GET, le ser-

veur lui renvoie la date courante, puis la date de dernière modification.

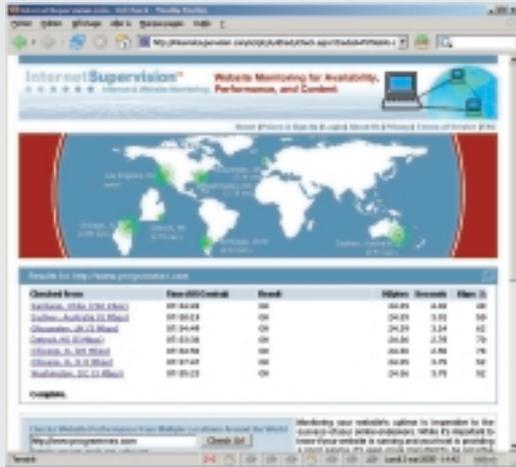
Ce qui donne en résumé (extrait) :

```
#curl -verbose http://www.programmez.com
GET http://www.programmez.com HTTP/1.1
User-Agent: curl/7.12.0 (i686-suse-linux) libcurl/7.12.0 OpenSSL/0.9.7d ipv6 zlib/1.2.1
Host: www.programmez.com
...
HTTP/1.0 200 OK
Date: Mon, 02 May 2005 11:26:49 GMT
Server: Apache/1.3.20 Sun Cobalt (Unix) PHP/5.0.3 mod_ssl/2.8.4 OpenSSL/0.9.6
mod_auth_pam_external/0.1 FrontPage/5.0.2.2510 mod_perl/1.2.6
X-Powered-By: PHP/5.0.3
...
```

D'autres renseignements sont renvoyés comme la date de dernière modification du document. Or, lorsque le navigateur client redemande

## Tests et montées en charge

En tant que concepteur de site Web, vous devez veiller à répondre aux besoins immédiats des utilisateurs (tester le site auprès d'utilisateurs finaux), optimiser les flux de navigation (tout en visant l'accessibilité maximale), et enfin, tester le site pour s'assurer qu'il respecte les standards. Comme en programmation, plus un problème sera détecté en amont, moins le coût de correction sera élevé. Il y a d'abord les classiques : le validateur syntaxique du W3C (<http://validator.w3.org/>), l'analyseur de liens (<http://validator.w3.org/checklink/>) à la recherche d'un lien cassé, ainsi que le service de validation CSS



du W3C qui vérifie la conformité des feuilles de style en cascade (CSS) autonomes ou intégrées aux documents (X)HTML (<http://jigsaw.w3.org/css-validator/>), ainsi qu'un test de vérification orthographique du français ou de l'anglais (<http://www.w3.org/2002/01/spell-checker>).

Il existe aussi des batteries de tests plus originaux : comme un test de temps de chargement (<http://www.1-hit.com/all-in-one/tool.load-time-checker.htm>), un test de lisibilité (<http://www.juicystudio.com/fog/>) ainsi que d'accessibilité ([http://www.usable-net.com/products\\_services/lift\\_online\\_free\\_trial/lift\\_online\\_free\\_trial.html](http://www.usable-net.com/products_services/lift_online_free_trial/lift_online_free_trial.html)), un test de qualité (<http://webxact.watchfire.com/>), et un test de simulation de Google (<http://www.gritechnologies.com/tools/spider.go>). Il existe aussi un test intéressant de performance depuis plusieurs localisations différentes (<http://internetsupervision.com/scripts/urlcheck/check.aspx>) et un test de validité des dates d'expiration des pages dans le cache (<http://www.web-caching.com>).

Les logiciels de montée en charge sont nombreux (aussi bien libres que commerciaux) et doivent s'effectuer en local. Par exemple avec httpperf vous pouvez exiger de lancer x sessions à la seconde (--rate), avec x sessions à lancer se composant chacune de x requêtes. Ainsi vous simulerez de nombreux utilisateurs arrivant sur le serveur par seconde.

```
httpperf -wss=100,50,0 -rate=150 -server=192.168.0.39 -uri=Programmez.html
Total: connections 4863 requests 9626 replies 4860 test-duration 3.508 s
Connection rate: 1386.4 conn/s (0.7 ms/conn, <=98 concurrent connections)
Request rate: 2744.3 req/s (0.4 ms/req)
CPU time [s]: user 0.46 system 1.40 (user 13.0% system 39.8% total 52.8%)
```

Maintenant, si vous devez tester la montée en charge d'une application basée J2EE ou Visual Studio .Net ou même ColdFusion, vous aurez besoin d'écrire des scénarii de tests. Par exemple, en France, le distributeur Ideo Technologies propose OpenLoad de l'éditeur OpenDemand. Avec cet outil, vous avez accès à une configuration très fine (timeouts, type de navigateur, plate-forme, la vitesse de connexion, etc.), et vous pouvez le dimensionner pour générer des dizaines de milliers d'utilisateurs virtuels et répondre à des demandes de très grosses montées en charge.

■ **Xavier Leclercq** - [Xavier.Leclercq@programmez.com](mailto:Xavier.Leclercq@programmez.com)

ce document, il envoie une date et un identifiant unique de la version dont il dispose. Par conséquent le serveur transmettra à nouveau le document uniquement si celui-ci a été modifié.

Alexandre Alapetite a développé une librairie qui sera utilisée conjointement aux sessions. Celle-ci vérifie que les données contenues dans \$\_SESSION ont été modifiées depuis la dernière génération du document par le biais d'une somme de contrôle MD5 stockée dans l'en-tête HTTP Etag. Voici concrètement ce que cela donne :

```
Exemple session.php
<?php
session_cache_limiter(""); //Désactive la génération automatique d'entêtes par la session
session_start(); //Démarrage la session
...
require_once("http-conditional.php");
//Récupère la date de la dernière modification du contenu (format Timestamp Unix)
//par le biais par exemple d'une requête à une base de données (ou date "en dur"...)
$dateDerniereModification=...;
if (httpConditional($dateDerniereModification,0,0,false,true,true))
{//Aucune modification depuis le dernier passage du client
... //Ferme la base de données, et autres nettoyages
exit(); //Pas besoin d'envoyer autre chose
}
//!\ Ne pas envoyer de texte au client avant cette ligne
... //La suite du script, comme s'il n'y avait pas cette première partie
?>
```

```


```

Vous devez inclure la bibliothèque et effectuer un test avant tout envoi de données au client. Et tout le bénéfice de cette technique ira à la bande passante...

Module http-conditional : <http://alexandre.alapetite.net/doc-alex/php-http-304/http-conditional.php.txt>

■ **Xavier Leclercq** - [Xavier.Leclercq@programmez.com](mailto:Xavier.Leclercq@programmez.com)



# Nvu 1.0 : un éditeur Web universel et gratuit, enfin viable sous Linux !

*Issu du projet Mozilla, Nvu est la refonte totale du module d'édition HTML de Mozilla. Depuis l'origine, Nvu est mené par le Français Daniel Glazman et activement soutenu par Linspire. Présenté comme le premier vrai IDE Web en Open Source "à la Front Page", il a l'avantage de fonctionner sous Windows, Linux et MacOS X, et est gratuit. Petite présentation.*

À la base, Nvu utilise le moteur de rendu Gecko. Il supporte CSS, XML, XUL, JavaScript, HTML 4.01 et XHTML. L'environnement reprend le fonctionnement par Tabs (Firefox) pour la navigation entre les pages, ce qui est très pratique. Il est adapté aux petits sites statiques. Les linuxiens ont enfin leur IDE Web graphique, en attendant un hypothétique portage de GoLive ou Dreamweaver.

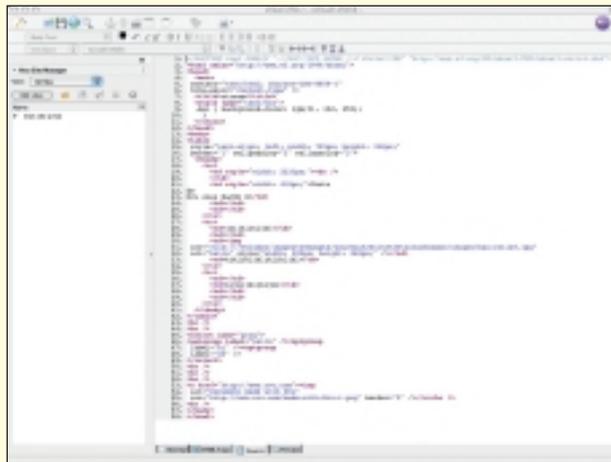
## L'éditeur

L'éditeur se découpe entre 4 onglets : Normal (pour le design de la page), HTML Tags (visualise la page sous forme de tags), Source et enfin, Preview. Il manque tout de même à l'outil un véritable gestionnaire de projets, afin de faciliter les créations, la gestion et les modifications de projets. Il n'y a pas comme sur d'autres IDE Web, de créateurs graphiques de site et sa représentation visuelle.

Pour créer des pages, rien de bien difficile. Par défaut, on utilise HTML, mais on peut cocher l'option XHTML (avec ou non DTD Strict). Si on peut regretter le minimum syndical sur les objets / composants d'interface, on apprécie les possibilités de personnalisation et de paramétrage de certains objets. L'objet tableau est très souple.

Pour la validation du code, on dispose d'un accès au Markup Validation Service, très utile pour traquer les problèmes de codage par rapport aux spécifications. Il n'y a pas de débogueur intégré. On dispose d'un éditeur CSS. Il manque encore la conception visuelle (impossible de faire du multi colonne) et surtout d'un validateur CSS. Si Nvu fonctionne très bien avec Firefox, avec Safari, on a observé quelques soucis d'affichages (par exemple avec le Selection List). Sous Internet Explorer,

même souci d'affichage. La récupération de pages Web d'un autre éditeur ne pose pas de réels soucis, même pour des pages complexes contenant du JavaScript (par contre, si l'objet est inconnu de Nvu, difficile d'utiliser la page).



Quand on importe une page Nvu sur un autre IDE Web, des différences de formatage apparaissent.

La publication de son site se fait très simplement. Il suffit de renseigner la partie "edit site" avec l'adresse FTP. La partie site permet d'afficher l'ensemble du site d'hébergement. Il peut afficher dans l'éditeur une page hébergée, la modifier puis la republier. Il est possible d'utiliser des templates, ce qui est pratique quand on crée un gabarit que l'on reprend ensuite pour l'ensemble des pages.

## Ergonomie

À l'usage, l'outil se révèle pratique et bien pensé. Les fonctions sont rapidement accessibles. Même si quelques fois, des petits soucis d'affichage existent sur les fenêtres de travail (ex. : préférences sous MacOS X). On apprécie que les fonctions d'accessibilité et de localisation soient proposées aux concepteurs.

Le dépouillement de l'interface favorise cela. La prise en main est rapide, quelques minutes suffisent pour comprendre l'éditeur et l'utilité de chaque section. Il est très simple d'aller d'une page à une autre grâce à la fonction tabs. Selon le type d'éditeur affiché, les fonctions sont ou non accessibles. On peut aussi changer l'interface via les thèmes (ici aussi, faites attention aux petits défauts d'affichage). Les performances de Nvu sont plutôt bonnes.

Comme dit plus haut, Nvu est un éditeur de pages web. On le présente volontiers comme un logiciel à la Dreamweaver, cependant n'allons pas trop vite en besogne. Il convient parfaitement aux petits sites et sites personnels. Sur des sites critiques, les pages mobiles, le debug, les pages dynamiques, Nvu ne peut rivaliser.

Cette version 1.0 montre tout le potentiel de l'outil, espérons qu'il saura se compléter au fur et à mesure, sans perdre sa simplicité. Il ne reste plus qu'à attendre les futurs projets.

■ François Tonic

## Fiche technique

### Nvu 1.0 pré-version

**Configuration :** Linux, MacOS X, Windows.

256 Mo minimum de mémoire vive

**Site :** <http://www.nvu.com>

#### Les +

- Les thèmes
- L'objet tableau
- Le paramétrage des objets
- La prise en main

#### Les -

- Des instabilités
- Compatibilité avec les navigateurs
- Objet vidéo
- Support CSS

# “NVU concurrence déjà Dreamweaver”

Dans le monde du Web et de l'Open Source, Daniel Glazman tient une place particulière. Il a été un acteur actif de Mozilla et l'est aujourd'hui de Nvu. Il pose un regard critique sur l'évolution actuelle du Web. Entretien avec un Français très impliqué dans le Web et dans l'Open Source.

**Programmez! : comment, aujourd'hui, un développeur web, doit-il choisir les langages, les technologies et les outils pour concevoir un site web, tout en respectant les contraintes imposées ?**



**Daniel Glazman :** Il n'y a que deux contraintes. Elles étaient déjà là en 1994 : faire en sorte que tous les visiteurs trouvent ce qu'ils cherchent. Cela implique de prendre en compte l'accessibilité, la lisibilité, la qualité de navigation, l'indexation, etc. La qualité de l'interface utilisateur est un élément majeur, son implémentation "propre" pour être accessible aux mal-voyants et handicapés est cruciale. Il faut aussi éviter à tout prix les solutions spécifiques à un navigateur donné, genre "ce site est optimisé pour tel navigateur". Faire en sorte que les évolutions et la maintenance du site soient aisées. Il faut utiliser les standards, éviter tout ce qui est propriétaire, et également tout ce qui est "hack" pour améliorer un site. Le "Best viewed by any browser" me semble la seule bonne approche. En la matière, je n'aime pas vraiment Flash. Flash est indispensable à certains, mais il n'est pas indexable, est difficile à maintenir et presque toujours inaccessible aux handicapés. Une plate-forme serveur Microsoft est une erreur en soi. Windows n'est pas assez stable, et les applications développées sous Windows non plus, pour assurer la stabilité nécessaire à des services sérieux. Bien entendu, d'autres auront une opinion totalement contraire... S'il y a des amateurs du reboot nocturne chaque nuit,

toutes les nuits de l'année, en guise de Garbage Collecting, ce n'est pas mon cas.

**PI :** *L'outil Nvu est un début de réponse face à des outils comme FrontPage, Contribute. Peut-il un jour véritablement concurrencer un poids-lourd comme Dreamweaver ? Quelle est la finalité de Nvu pour le développeur web ?*

**DG :** Le but n'est pas de concurrencer Dreamweaver. En tous cas, ce n'est pas le mien. Mon but est de fournir aux usagers un bon éditeur Web, moderne, extensible, conforme aux standards, gratuit, dont le source est libre. Nvu concurrence déjà Dreamweaver. Vu le coût de la licence de Dreamweaver, le segment bas du marché, celui qui utilise 10% des fonctionnalités de Dreamweaver, passe facilement à Nvu. Mais Nvu commence également à grignoter le segment au-dessus, celui des industriels et des auteurs Web plus avancés, mais qui comparent fonctionnalités et coût. Je ne dis pas que Nvu est au niveau de Dreamweaver, il faut comparer ce qui est comparable.

**PI :** *On parle désormais de découpler les couches du site web en rendant le plus indépendant possible la présentation, du fonctionnel. Est-ce selon vous une bonne solution ? Quelles en sont les limites ?*

**DG :** C'est le rêve inassouvi des experts du monde SGML et XML depuis vingt ans déjà. Je viens également de ce monde-là, mais je suis assez partagé. Je pense, par exemple, que les feuilles de styles remplacent difficilement les tableaux. Avant de crier au scandale sur cette opinion, je tiens à préciser que je travaille sur les feuilles de styles depuis 1989 avec le langage P de Vincent Quint, que je suis membre du CSS Working Group du W3C depuis 1997, que j'ai participé à l'élaboration des CSS2 et continue à travailler sur les CSS3.

**PI :** *xHTML s'impose comme le successeur du HTML, et avec son couplage avec CSS, on dispose d'un modèle de programmation intéressant et flexible. Est-il temps d'abandonner*

*HTML pour un langage moins permissif, plus structuré ?*

**DG :** Non, pas du tout. Je pense que le W3C a raté le coche avec son activité actuelle autour de XHTML. XHTML 1.0 ne pose pas de problème mais le type MIME des données est un vrai souci. On ne peut plus utiliser text/html et text/xml est insuffisant. C'est un peu chaotique. On va y arriver, c'est juste du HTML4 à la sauce XML, il n'y a pas de rupture technologique. Au contraire, XHTML2 me semble une erreur stratégique grave, une rupture volontaire bien trop grande avec HTML4. Adoption du W3C qui risque d'ailleurs ne pas être trop facile puisque deux implémentations intéropérables indépendantes seront nécessaires. On aura peut-être deux outils implémentant XHTML2, mais à quoi cela servira-t-il si aucun NAVIGATEUR n'implémente XHTML2 ? Je suis très sceptique, et je pense que l'initiative du WHAT Working Group (<http://www.whatwg.org/>) est beaucoup plus intéressante car elle prend en compte l'existant HTML4 sans l'abandonner et elle a beaucoup plus de liens avec les usagers et les bugs des navigateurs eux-mêmes. Je sais que Microsoft est extrêmement sceptique quant à XHTML2, et c'est un signal d'alarme qui devrait suffire...

**PI :** *Quels sont pour vous les remèdes, s'ils existent, pour rendre un site le plus générique possible malgré la diversité des navigateurs et des implémentations des standards plus ou moins rigoureuses ?*

**DG :** Sachant que seuls les geeks codent eux-mêmes leur site, je pense que la réponse à cette question doit se voir sur deux tableaux : les geeks doivent se former aux standards. C'est facile de comprendre XML "en gros", mais c'est différent de le comprendre et de le maîtriser. Combien de geeks par exemple comprennent comment et pourquoi PHP est totalement en violation de HTML 4 et XML 1.0 ? On ne devient pas "conforme aux standards" en deux coups de cuiller à pot.

■ *Propos recueillis par François Tonic*

# 6 HÉBERGER ET PUBLIER

## La vie du site commence...

[www.programmez.com](http://www.programmez.com)

- liens
- codes
- dialogue avec les auteurs

La phase finale du développement d'un site est sa publication sur le réseau et/ou Internet. Les outils de création possèdent des fonctions WebDav et FTP. La publication ne comporte aucune difficulté si la structure du site est bien pensée et respectée lors de la publication. Vous aurez bien entendu au préalable réservé un nom de domaine et une place chez un hébergeur si vous ne le faites pas sur une machine louée ou chez vous.

### Déployer son site via FTP

Votre site est prêt, mais comment le transférer ? Par le biais du protocole de transfert de fichiers FTP. Aujourd'hui, avec une formule d'hébergement de base comme celle proposée, par exemple, par le fournisseur Haisoft, vous disposez pour moins de 25 EUR par an d'un accès illimité au FTP (24h/24) pour mettre à jour votre site.



Mieux, via le logiciel de configuration Plesk 7.5, vous avez la possibilité de définir vos adresses email, mais aussi le nom de l'utilisateur FTP et son mot de passe associé. Pour

déposer vos fichiers php ou autres, vous devez ensuite utiliser un logiciel client FTP. En fait, de nombreux outils d'édition de sites Web utilisent intensivement le FTP, pour poster l'intégralité du site ou pour réaliser une sauvegarde. Il suffit bien souvent de les configurer convenablement avec le trinôme adresse IP du serveur FTP, utilisateur et mot de passe. Avec SPIP vous pouvez réaliser une sauvegarde automatique, via FTP, sous format XML, avant la mise à jour du logiciel serveur. Sous Max OS X, Linux ou Windows, vous pouvez manipuler vos fichiers à distance avec FireFTP l'extension cliente FTP de FireFox.

Certaines précautions sont quand même à prendre : d'abord, si vous développez sous Windows, mais que votre site est hébergé sous Linux, vous devrez prendre garde à la casse (Windows est plus tolérant), ensuite, il faudra faire attention à ne conserver que les droits minimaux de propriété et d'exécution des fichiers (colonne "attributs" sous FireFTP). Enfin, bannissez le FTP anonyme et placez si possible un quota d'utilisation, ou surveillez régulièrement l'espace disque (en effet si un hacker prend le contrôle de votre serveur FTP, il pourra le "tager" de manière à masquer les noms de fichiers qu'il y dépose).

- Haisoft : <http://www.haisoft.fr/>
- Plesk : <http://www.sw-soft.com/en/products/plesk75reloaded/>
- FireFTP : <http://extensions.geckozone.org/FireFTP/>
- RFC 959 (en français) : <http://abcdrfc.free.fr/rfc-vf/rfc959.html>
- Xavier Leclercq - [Xavier.Leclercq@programmez.com](mailto:Xavier.Leclercq@programmez.com)

### Comment choisir son hébergeur ?

#### Votre grille de sélection

Commençons par quelques évidences. Il n'existe pas d'hébergeur ou de formule d'hébergement idéale, tant le contenu des prestations peut varier. La première étape consiste donc à se poser les bonnes questions : S'agit-il d'un site professionnel ? Quels types de fonctionnalités votre site est-il destiné à supporter ? Quels langages sont utilisés sur votre site ? Quels média devront être hébergés ? Quelle fréquentation estimez-vous ?

A partir de ces questions, vous pourrez fixer les principaux points techniques nécessaires à votre grille de sélection : espace disque, trafic, technologies nécessaires, bases de données, taux de service, débit, système d'accès au serveur (SSH, Telnet), présence d'un panneau de contrôle, cgi, possibilité de compression Gzip et bien entendu type d'hébergement (mutualisé, serveur privé, serveur dédié).

#### Coût total d'hébergement et économies de bouts de chandelles

Il est probable que vous trouviez un bon nombre d'hébergeurs répondant à l'ensemble de vos exigences techniques, et à partir de là, la tentation de prendre le moins cher peut être grande... Cependant, le tarif ne doit pas forcément être le critère principal. L'hébergement recèle des coûts cachés : les temps d'administration et les temps d'indisponibilité principalement. Un serveur stable, une équipe technique joignable et réactive, une administration facilitée valent sûrement plus que quelques euros annuels. Prenez aussi garde à lire complètement les contrats soumis par les hébergeurs. Certaines clauses particulières peuvent s'avérer très pénibles a posteriori (Comment ? Ma base MySQL ne peut pas supporter plus de quatre connexions simultanées ?).

#### Histoires de confiance et de garantie

L'hébergement, comme tout les métiers, comporte des sociétés allant de la quasi-escroquerie aux spécialistes du service : il s'agit donc de tirer le bon numéro et si possible autrement qu'au hasard. Un avis positif au détour d'un forum n'est sûrement pas suffisant, pas plus d'ailleurs que le " Untel est très bien " entendu près de la machine à café. Il vaut donc la peine de passer un peu de temps à traîner sur la toile pour recueillir quelques échantillons d'expériences. Trouvez des sources récentes : les choses évoluent très vite dans le petit monde de l'hébergement et la vérité d'aujourd'hui n'est pas forcément celle de demain. C'est pourquoi rester au fait de l'état du marché après votre choix est loin d'être inutile...

■ Gauthier Delamarre

# 7 USER-AGENT



*Nous allons décrire pas à pas, comment proposer un langage de développement simplifié. Nous étudierons le modèle objet, et nous proposerons une syntaxe au format XML. Nous utiliserons plusieurs techniques pour optimiser le code. Le langage que nous allons réaliser permet de simplifier l'analyse des différentes versions des navigateurs, et permet d'intégrer rapidement de nouvelles versions.*

Proposer un site Internet qui fonctionne pour toutes les versions des navigateurs est un challenge ambitieux. Chaque version possède ses particularités. Par exemple, définir la largeur d'un champ texte pour accueillir tous les caractères voulus, n'est pas chose aisée. Une taille conforme sous IE Windows, n'est pas valide sous IE Macintosh ou Firefox sous Linux.

De nouvelles versions sortent régulièrement. Il faut pouvoir adapter l'application aux évolutions, sans devoir revoir toute l'application. Nous vous proposons de rédiger un script en XML, permettant d'adapter l'application aux différentes versions des navigateurs. Cela sera l'occasion d'étudier la modélisation d'un langage de développement simple, d'utiliser un analyseur XML pour le compiler, et d'exploiter le polymorphisme des langages objets pour exécuter le scénario.

Voici un extrait d'un script que nous souhaitons pouvoir exécuter.

```
<if match="^[Ww]in|Microsoft Internet Explorer">
  <set key="family.os" value="Windows"/>
</if>
```

Chaque navigateur s'identifie auprès du serveur par l'en-tête User-Agent. Celui-ci peut indiquer de nombreuses informations comme la version du navigateur, la version qu'il simule, les extensions qu'il possède, la langue utilisée, etc.

Voici par exemple, les chaînes retournées par certaines versions d'Internet Explorer et Firefox sous Windows :

```
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)
Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.5) Gecko/2004 1107 Firefox/0.10.1
```

Vous trouverez d'autres possibilités ici : <http://www.psychedelix.com/agents.html>

La première chose à faire est d'analyser le format de l'en-tête User-Agent, émis par les navigateurs. Celui-ci respecte une syntaxe ouverte, composée de trois champs principaux. La première partie indique le nom de l'agent. Pour des raisons historique et de compatibilité, la chaîne de caractère commence généralement par Mozilla suivi éventuellement d'un numéro de version. Cela correspond au nom de code de la toute première version du navigateur de Netscape. Ensuite, entre parenthèses, de nombreuses précisions sont indiquées, séparées par des points-virgules. Après la parenthèse fermante, on trouve également des informations complémentaires.

Nous allons découper ces chaînes en champs. Le premier correspond à la première partie avant la parenthèse. Ensuite, nous trouvons les différentes valeurs entre les paramètres, à raison d'un champ par point virgule. La dernière partie constitue également un champ.

```
ArrayList items = new ArrayList(7);
int idx = userAgent.indexOf('(');
if (idx != -1)
{
  items.add(userAgent.substring(0, idx - 1));
  for (StringTokenizer tokens = new StringTokenizer(userAgent
    .substring(idx + 1), ";");
    tokens.hasMoreTokens();)
    items.add(tokens.nextToken().trim());
}
else
  items.add(userAgent);
```

Ainsi, il devient plus facile d'analyser les champs. Des expressions régulières vont nous aider à rechercher des valeurs dans les champs, afin de valoriser des variables.

Nous souhaitons permettre la rédaction d'un script d'analyse à l'aide d'un langage XML. Celui-ci doit nous permettre d'exprimer l'algorithme à utiliser sur l'en-tête User-Agent, afin

d'identifier les valeurs à indiquer pour différentes variables. Celles-ci seront exploitables par l'application. Elles pourront valoriser les tailles des champs, les polices de caractères à utiliser, les modifications des scripts, les largeurs des colonnes, etc.

Nous choisissons de rédiger un langage impératif. C'est-à-dire que celui-ci nous permet d'exprimer des actions à effectuer, et non des règles à analyser. Java est un langage impératif. Nous avons besoin de deux structures principales : les instructions et les expressions. Les instructions permettent d'indiquer une action à effectuer. Une expression permet d'exprimer un calcul, dont le résultat aura une influence sur l'instruction à exécuter.

Une interface Statement permet de porter les différentes instructions. Elle propose une méthode execute(). Celle-ci attend deux paramètres : une liste de champs (ceux déduit de la version du navigateur) et une liste de propriétés pour y stocker les résultats du traitement.

```
/**
 * A statement.
 *
 * @since 1.0
 * @version 1.0
 */
interface Statement
{
  /**
   * Execute the statement.
   *
   * @param items The items presents in the user agent.
   * @param prop The properties to set.
   *
   * @since 1.0
   *
   * @pre items!=null
   * @pre prop!=null
   */
}
```

```
public void execute(AbstractCollection items,
Properties prop);
}
```

Nous utilisons un paramètre de type `AbstractCollection` pour pouvoir accueillir tout type de conteneur. Il est préférable d'utiliser le type le plus générique possible, afin de laisser plus de liberté au développeur. Les variables à valoriser lors de l'analyse sont mémorisées dans un objet `Properties` sous forme de caractère. Cette classe sert à cela. Elle propose des méthodes d'accès au format caractère et n'a pas l'inconvénient d'un `RessourceBundle`. En effet, il n'est pas nécessaire d'exploiter inutilement les capacités multi langues du `RessourceBundle`.

Quelles sont les instructions dont nous avons besoin ? Il nous faut une instruction se chargeant d'exécuter une liste d'instructions. Cela correspond aux accolades des langages comme Java ou C++ ou au `begin/end` du Pascal. La classe `Body` représente un bloc d'instruction. Elle possède un tableau d'instruction qu'elle se propose d'invoquer dans l'ordre.

```
class Body implements Statement
{
private Statement[] body_;

Body(AbstractCollection body)
{
body_ = new Statement[body.size()];
body.toArray(body_);
}

public void execute(AbstractCollection items,
Properties prop)
{
final int s = body_.length;
for (int i = 0; i < s; ++i)
{
body_[i].execute(
items, prop);
}
}
}
```

Après la compilation du programme, une seule instance `Body` est nécessaire. Celle-ci possède alors toutes les autres instructions et expressions du programme. Compiler un programme consiste à construire un arbre d'objet, afin d'obtenir une instance `Body` racine. Nous avons également besoin d'une instruc-

tion `SetProperty` pour valoriser les fameuses variables, objectifs de l'exécution du programme. La classe `SetProperty` s'occupe de cela. Le constructeur attend une clef et une valeur. L'exécution de cette instruction ne fait qu'ajouter la valeur indiquée sous la clef, dans l'instance `Properties`.

```
class SetProperty implements Statement
{
private String key_;
private String value_;

SetProperty(String key, String value)
{
key_ = key;
value_ = value;
}

public void execute(AbstractCollection items,
Properties prop)
{
prop.put(key_, value_);
}
}
```

Si le langage ne propose que la valorisation de variables, il ne peut pas faire grand-chose. Il nous manque une instruction de comparaison. Un `If`. Cette instruction regroupe une expression et une instruction à exécuter si celle-ci retourne `true`. La classe `If` exprime cela.

```
class If implements Statement
{
private Expression expr_;
private Statement body_;

If(Expression expr, Statement body)
{
expr_ = expr;
body_ = body;
}
```

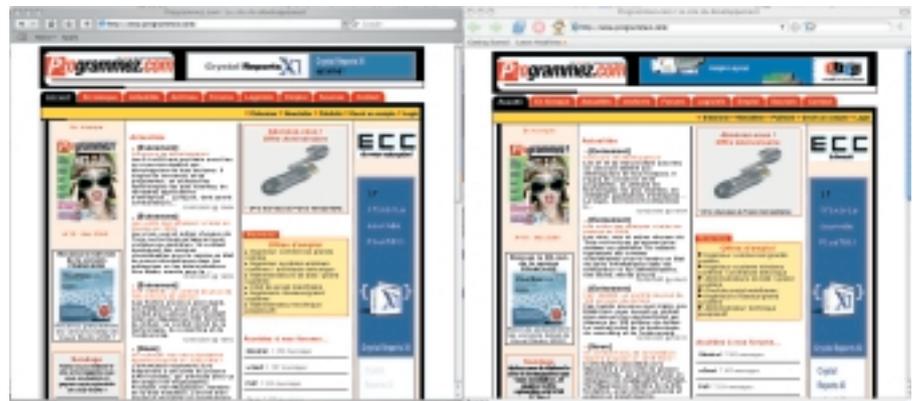
```
}

public void execute(AbstractCollection items,
Properties prop)
{
if (expr_.match(items))
body_.execute(items, prop);
}
}
```

La simplicité de notre langage n'exige pas de structures plus complexes comme les boucles, les déclarations de fonctions ou de procédures, de variables, etc. La même approche serait utilisée, en implémentant l'interface `Statement`.

Il nous faut maintenant être capables de proposer des expressions logiques. L'interface `Expression` propose une méthode `match()` retournant un booléen. En effet, dans le cas qui nous concerne, seules les expressions booléennes nous importent. Nous souhaitons savoir si l'en-tête `User-Agent` correspond à un modèle particulier. Si c'est le cas, nous valorisons des variables.

```
/**
* An expression.
*
* @since 1.0
* @version 1.0
*/
interface Expression
{
/**
* Check if the items match the expression.
*
* @param items The items presents in the user
agent.
* @return true if an items match the expression.
*
* @since 1.0
*/
```



```

*
* @pre items!=null
*/
public boolean match(AbstractCollection items);
}

```

L'expression la plus importante pour ce langage consiste à rechercher un modèle dans les champs de l'en-tête User-Agent à l'aide d'une expression régulière. La classe MatchRE implémente l'interface Expression pour cela.

```

class MatchRE implements Expression
{
    private Pattern re_;

    MatchRE(Pattern re)
    {
        re_ = re;
    }

    public boolean match(AbstractCollection items)
    {
        for (Iterator i = items.iterator(); i.hasNext(); )
        {
            if (re_.matcher((String) i.next()).find())
                return true;
        }
        return false;
    }
}

```

Ainsi, une instruction If peut rechercher un modèle de chaîne de caractère, et valoriser ainsi des propriétés, à l'aide d'instructions SetProperty. Les expressions sont booléennes. Nous pouvons alors offrir des expressions logiques afin de faciliter la rédaction d'expressions complexes. La classe Not permet d'inverser le sens du test. Il est difficile de faire plus simple. Elle attend une autre expression pour inverser le résultat.

```

class Not implements Expression
{
    private Expression expr_;

    Not(Expression expr)
    {
        expr_ = expr;
    }

    public boolean match(AbstractCollection items)
    {

```

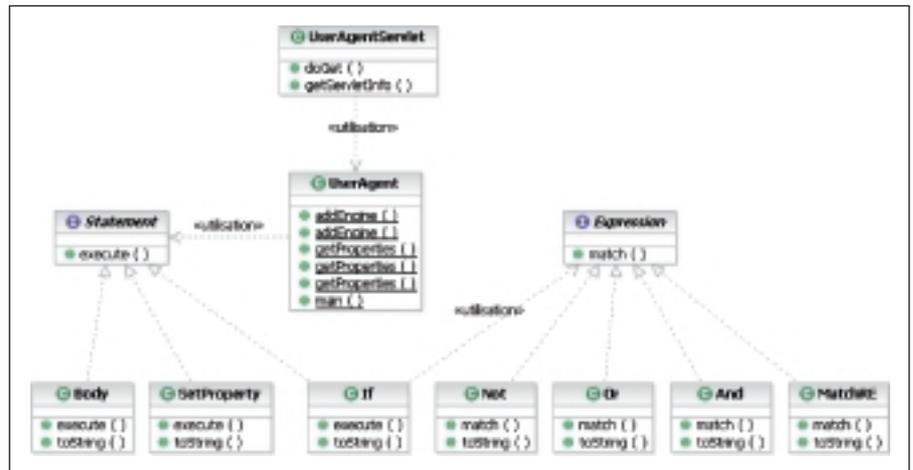


Figure 1

```

return lexpr_.match(items);
}
}

```

Les classes Or et And permettent de combiner les expressions booléennes. Pour des raisons d'optimisation, les instances mémorisent des tableaux d'expressions et non un simple couple d'expression.

```

class Or implements Expression
{
    private Expression[] nodes_;

    Or(AbstractCollection nodes)
    {
        nodes_ = new Expression[nodes.size()];
        nodes.toArray(nodes_);
    }

    public boolean match(AbstractCollection items)
    {
        for (int i = nodes_.length - 1; i >= 0; -i)
            if (nodes_[i].match(items))
                return true;
        return false;
    }
}

```

```

class And implements Expression
{
    private Expression[] nodes_;

    And(AbstractCollection nodes)
    {
        nodes_ = new Expression[nodes.size()];
        nodes.toArray(nodes_);
    }
}

```

```

public boolean match(AbstractCollection items)
{
    for (int i = nodes_.length - 1; i >= 0; -i)
        if (!nodes_[i].match(items))
            return false;
    return true;
}
}

```

Notez les différences entre les deux méthodes match(). Dans la version de la classe Or, il suffit d'avoir une expression valide pour interrompre l'analyse. Dans la version de la classe And, c'est l'inverse. Il faut avoir une expression non valide pour interrompre l'analyse.

Il nous reste à proposer une classe permettant de compiler un fichier XML pour produire des instances Statement et Expression. Ensuite, lors de l'analyse d'un en-tête User-Agent, il suffit d'invoquer le moteur avec les champs extraits.

```

result = new Properties();
engine_.execute(items, result);

```

Le modèle objet est représenté Figure 1. La classe UserAgent se charge d'analyser tout cela et d'exécuter le script.

Dans la 2e partie, nous regarderons comment proposer une syntaxe XML à notre langage, et nous proposerons une servlet pour exploiter l'analyse de la version de l'agent côté client. Nous en profiterons pour exploiter toutes les possibilités avancées de gestion du cache du protocole http.

**Philippe PRADOS**  
 Senior IT Architecte chez IBM  
 Sécurité et technologies GRID  
 press@philippe.prados.name

# 8 → AJAX : LE RENOUVEAU DU CLIENT LÉGER RICHE

Les applications client léger, reposant sur un navigateur web, connaissent une ère de stabilité avec un ensemble normalisé de frameworks reposant sur le pattern MVC2 (Model, View, Controller) tel que Struts, JavaServerfaces ou Copix (php). Ces frameworks prennent en charge la génération des écrans et simplifient les enchaînements d'une vue à l'autre, tout en séparant bien la partie " données " des éléments de " présentation ".

Jusqu'à présent, chaque action était généralement traduite par un aller-retour sur le serveur, avec la régénération de la vue en cours, pour le navigateur. Cette opération, à priori fastidieuse, et pas forcément justifiée, représente un coût non négligeable pour le serveur qui génère des écrans et pour le réseau qui fait transiter des éléments de présentation, alors que l'application pourrait se contenter de véhiculer que des données. Les applications de type client riche prennent en charge toute la génération de la présentation et se contentent d'échanger avec le serveur des objets valeurs qui contiennent les données utiles. Cette démarche bien connue et pratiquée massivement avant l'avènement des navigateurs web, est simple à mettre en œuvre et décharge le serveur de tous les traitements visuels. D'un autre côté, les navigateurs relativement récents comme IE 5.5, Mozilla 1.4/Firefox 0.8, ou Safari 1.2 (qui représentent plus de 98% du parc), intègrent un potentiel sous-exploité, à savoir, la capacité de modifier dynamiquement la vue à l'aide de scripts, et la possibilité d'effectuer des requêtes asynchrones au serveur, sans avoir à recharger la page.

Ce type de fonctionnalités était exploité de façon anecdotique par quelques développeurs, jusqu'à ce que Google lance son service d'email, GMail, suivi de Google Suggest (pour l'aide à la recherche), et surtout Google Maps, qui a prouvé qu'il était possible de concilier une interface riche avec un client léger.

## Présentation d'AJAX

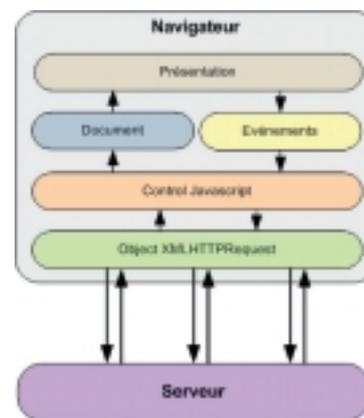
Le concept " AJAX " (Asynchronous Javascript and XML) introduit à travers l'article de Jesse James Garrett, présente une nouvelle approche pour la conception d'applications web s'appuyant sur l'utilisation de Javascript et d'une librairie de communication client/serveur asynchrone reposant sur le protocole HTTP et la norme XML.

D'un côté, la présentation s'appuie sur la capacité des navigateurs à prendre en compte les modifications dynamiques de l'arbre de présentation. Dans les navigateurs récents, une page HTML est en réalité montée en mémoire sous forme d'arbre XML : le navigateur se charge de corriger et d'interpréter le document pour obtenir une structure cohérente. Cet arbre est manipulable à l'aide des fonctions DOM (Document Object Model normalisé par le W3C) accessibles en Javascript et implémentées dans les navigateurs supportant la norme.

De plus, en tirant profit des fonctionnalités avancées des CSS (Cascading Style Sheet) implémentées dans les navigateurs, le niveau de présentation atteint un degré d'ergonomie proche de celui des interfaces riches, quand il ne le dépasse pas.



De l'autre côté, la communication entre le client et le serveur repose sur l'utilisation de l'objet XMLHttpRequest [8], implémenté à l'origine par Internet Explorer sous forme d'ActiveX (disponible depuis dans Mozilla 1.4, Safari 1.2, Opera 8). XMLHttpRequest offre la possibilité d'effectuer des requêtes HTTP à partir d'appels Javascript et de récupérer le résultat sous forme d'un document XML. Le traitement du résultat peut être effectué de façon asynchrone, ce qui augmente le confort de l'utilisateur qui n'est pas bloqué par l'aller-retour serveur pour un rafraîchissement de son écran. En combinant les deux points évoqués précédemment, le



développement d'une application web ne se conçoit plus sous forme d'enchaînement de pages, mais plus comme une application riche (à l'instar d'une applet par exemple), ou tout ou partie de l'interface est chargée une fois en mémoire, et les différentes actions effectuées sur l'interface engendrent des appels au serveur, via l'objet XMLHttpRequest. La partie "contrôle" est

entièrement prise en charge par du code Javascript et l'échange avec la couche service/métier repose sur le transfert d'objets valeurs à l'aide d'appels en Javascript.

## La communication client/serveur asynchrone

Comme le présente très bien l'article (<http://developer.apple.com/internet/webcontent/xmlhttpreq.html>) de " Apple Developer Connection ", les communications client/serveur s'appuient sur l'utilisation de l'objet XMLHttpRequest à travers du code Javascript.

Le principe de fonctionnement est simple :

- L'objet est instancié
- Une fonction de 'callback' est associée à l'instance

- Une connexion est ouverte sur le serveur web
- Les paramètres sont envoyés au serveur
- Pour un changement d'état de l'objet, la fonction de 'callback' est appelée pour éventuellement traiter la réception des données (dans le cas où le transfert est accompli)

Le code Javascript correspondant est le suivant :

```
// L'objet est instancié
var req = new XMLHttpRequest();
// Sur Internet explorer l'instanciation utilise un objet ActiveX
//var req = new ActiveXObject("Microsoft.XMLHTTP");

// Une fonction de 'callback' est associée à l'instance
req.onreadystatechange = processReqChange;

// Une connexion est ouverte sur le serveur web
// le 3eme paramètre indique s'il s'agit d'une connexion asynchrone
req.open("GET", url, true);

// Les paramètres sont envoyés
req.send( "MesDonnées..." );

// Pour un changement d'état de l'objet
// la fonction de 'callback' est appelée
function processReqChange()
{
    // si l'état de la requête est de type 'complete'
    // alors on effectue le traitement
    if( req.readyState == 4 )
    {
        // Le code d'erreur HTTP est vérifié
        if( req.status == 200 )
        {
            // Pas d'erreur : la réponse peut être prise en compte
            reponse = req.responseXML;
            // Traitement sur la réponse
        }
        else
        {
            // Erreur : une alerte est générée
            alert("Erreur de traitement serveur :\n" + req.statusText);
        }
    }
}
```

Le format des données échangées n'est pas imposé : au serveur d'interpréter le contenu. Le programmeur est libre de construire un message SOAP, XMLRPC ou encore REST (ou autre), en fonction des exigences du serveur.

## La présentation avec Javascript et DOM

Les navigateurs web évolués maintiennent en mémoire un arbre de données qui représentent la page HTML. Cet arbre (ou document) est acces-

sible à l'aide de l'interface DOM implémentée en Javascript. A l'aide de fonctions standards, le programmeur parcourt la hiérarchie de nœuds et a la possibilité de créer, mettre à jour ou supprimer dynamiquement certains fragments. Une modification du document impacte directement le rendu dans le navigateur qui tient immédiatement compte des nouvelles valeurs. Pour un fragment HTML suivant :

```
<p id="message">Hello</p>
```

Le code Javascript s'appuyant sur l'interface DOM pour modifier directement le contenu s'écrit :

```
<script type="text/javascript">
document.getElementById( "message" ).firstChild.data =
"Hello world";
</script>
```

Après exécution du code, le texte contenu dans l'élément <p> devient "Hello world" et le navigateur met à jour l'écran.

Par extension, toute la page peut être complètement mise à jour. L'utilisation de l'attribut (hors norme) 'innerHTML' simplifie énormément ce genre d'opération en assurant la création des éléments à partir d'une simple chaîne de caractères.

Si nous reprenons notre exemple précédent :

```
<script type="text/javascript">
document.getElementById( "message" ).innerHTML =
"<table id='table'><tr><td> Hello world </td></tr></table> ";
</script>
```

Nous créons une table dans l'élément <p> à partir de la chaîne de caractères, et ce, sans passer par une succession fastidieuse d'appels de méthodes de création, d'ajout et d'organisation de nœuds de l'API DOM.

## Utilisation avec des WebServices

### Exemple de mise en œuvre

Nous allons illustrer la combinaison des deux notions à travers un appel simple à un Webservice.

Nous allons appeler la méthode distante à l'aide d'une enveloppe SOAP (RPC : Remote Procedure Call) :

```
String getData( int id ) ;
```

Le code JavaScript correspondant à l'appel s'écrit :

```
req.onreadystatechange = processReqChange;
req.open( "GET", url, true );

// Dans le cadre d'une requête SOAP nous ajoutons les entête HTTP suivants
req.setRequestHeader("Man", "POST "+ url + " HTTP/1.1");
req.setRequestHeader("MessageType", "CALL");
req.setRequestHeader("Content-Type", "text/xml; charset=UTF-8");
req.setRequestHeader("SOAPAction", "urn:test#getData" );

// Nous construisons l'enveloppe SOAP à la main
var id = 10;
var request = "<?xml version='1.0' encoding='UTF-8'?>" +
"<SOAP-ENV:Envelope"+
    ' xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"'+
    ' xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"'+
```

```
' xmlns:xsd="http://www.w3.org/1999/XMLSchema" >'+
'<SOAP-ENV:Body>'+
'  <ns1:getData xmlns:ns1="urn:test" SOAP-ENV:encodingStyle=
"http://schemas.xmlsoap.org/soap/encoding/">'+
'    <id xsi:type="xsd:int">'+id+'</id>
'  </ns1:getData>'+
'</SOAP-ENV:Body>'+
'</SOAP-ENV:Envelope>';

// L'enveloppe SOAP est envoyée au serveur
req.send( request );
```

Le traitement de la réponse est effectué dans la méthode 'processReqChange' prévue à cet effet. En cas de succès, l'élément <p> doit contenir la valeur de retour de la méthode web 'getData'.

```
function processReqChange()
{
  // si l'état de la requête est de type 'complete'
  // alors on effectue le traitement
  if( req.readyState == 4 )
  {
    // Le code d'erreur HTTP est vérifié
    if( req.status == 200 )
    {
      // Pas d'erreur : la réponse peut être prise en compte
      reponse = req.responseXML.getElementsByTagName("return")[0];
      // Traitement sur la réponse
      var returnValue = reponse.firstChild.nodeValue;
      document.getElementById( "message" ).innerHTML = returnValue;
    }
    else
    {
      // Erreur : une alerte est générée
      alert("Erreur de traitement serveur :\n" + req.statusText);
    }
  }
}
```

### Limitations et solutions

L'exemple ci dessus illustre un appel RPC assez simple à une méthode, à l'aide d'une enveloppe SOAP et déjà, nous constatons une certaine lourdeur dans la construction du message et la récupération du résultat. Pour des structures de données complexes, cette opération s'avère très fastidieuse à implémenter en Javascript. Pour quelques cas particuliers, cette approche peut suffire, mais lorsqu'il est question d'effectuer un grand nombre d'appels à différentes méthodes pour échanger des données hétérogènes, cette technique atteint ses limites. La solution simple consiste à utiliser un protocole plus léger (REST, XMLRPC) pour les échanges de données. Une autre solution revient à s'appuyer sur les extensions intégrées aux navigateurs web pour effectuer des appels aux services web. Les navigateurs basés sur Mozilla disposent d'une API pour s'interfacer avec des WebServices. Dans ce cas de figure, un appel en SOAP (synchrone) se présente sous la forme suivante :

```
var id = 10 ;
```

```
// Creation de l'instance
var soapCall = new SOAPCall();
soapCall.transportURI = this.soapURL;

// Nous construisons un tableau de paramètres
var params = new Array();
var i = 0;
params[i++] = new SOAPParameter( id , "id" );

// Les paramètres sont encodés
soapCall.encode(0, "getData", "urn:test", 0, null, params.length, params);

// Appel de la méthode web
var soapResponse = soapCall.invoke();

// Traitement de la réponse
var num = new Object();
var params = soapResponse.getParameters( false, num );
var data = params[0].value;
```

Mozilla est également capable d'utiliser directement un fichier de description de WebServices (WSDL) pour simplifier la programmation de l'appel. Internet Explorer dispose également de mécanismes similaires (mais avec une API différente) se basant sur un composant ActiveX. Dans tous les cas, l'utilisation de briques logicielles non normalisées force à écrire un code spécifique pour chaque type de navigateur. L'utilisation d'XMLHttpRequest reste quand même le plus petit dénominateur commun à tous les navigateurs évolués.

### Vers la définition d'un framework homogène

L'utilisation combinée de XMLHttpRequest et de Javascript/DOM connaît un essor depuis sa mise en œuvre à travers Google Mail. Cette technique relativement nouvelle de développement d'interface graphique légère ne bénéficie encore pas de framework avancé.

Quelques tentatives commencent à émerger comme Echo2 ou JST qui proposent un début de framework intéressant. De même, la norme Java Server Faces 2.0 semble s'orienter également vers une adoption de l'approche AJAX. Une autre piste consiste à s'appuyer sur les fonctions de transformation XSL intégrées dans les navigateurs évolués pour régénérer l'écran à partir des valeurs issues du serveur.

L'adoption d'un framework stable et efficace est d'autant plus nécessaire que le développement sans outil et le débogage d'une application AJAX est relativement difficile. La plate-forme Mozilla propose notamment des outils comme le débogueur Javascript Venkman qui simplifie la tâche du développeur.

### ■ Alexis AGAHI

Expert XML/J2EE - Homsys-  
Aston, filiale de Homsys Group



Homsys-Aston et Aston-Education, filiales de Homsys Group, conseillent, mettent en œuvre et proposent des formations sur des projets d'envergures orientés autour de 3 axes : Pilotage de l'entreprise; Performance du Système d'Information et Performance des Infrastructures. Homsys Group est implanté nationalement et fédère plus de 500 collaborateurs, et est coté en Bourse (Code ISIN : FR0010000869). [www.homsysgroup.com](http://www.homsysgroup.com)

# Octo Technology, un concentré d'architecture

*Créée en 1998, Octo Technology s'est, dès l'origine, spécialisée sur l'architecture. L'an dernier, cette société qui compte une cinquantaine de consultants a réalisé un chiffre d'affaires de 5,7 M€, en progression de 35%.*

**A**uparavant architecte chez Atos, François Hisquin, fondateur d'Octo Technology était conscient de la rareté de cette ressource dans les grandes SSII, encore davantage en 1998, souligne-t-il. Son idée a donc été de fédérer les compétences d'architecte au sein d'un cabinet assez semblable à ceux pouvant exister dans le bâtiment. François Hisquin évoque le concept "d'intelligence collective" et insiste sur les processus internes qui favorisent les échanges. Une des spécificités d'Octo réside dans cette approche très collaborative des projets, via notamment des "comités challenging", où l'architecte responsable d'un projet est confronté à ses pairs dans un objectif d'amélioration de la qualité. Le client, souligne François Hisquin, doit avoir accès à l'architecture Octo, pas à celle d'un consultant spécifique.

Les livres blancs régulièrement publiés par Octo (et téléchargeables sur le site [www.octo.fr](http://www.octo.fr)) sont le fruit de ces travaux collectifs. Derniers parus : sur l'architecture orientée services (SOA), sur l'architecture de systèmes d'information.

Société du groupe Aubay depuis 1999, un lien capitalistique et non fonctionnel, Octo Technology compte quatre activités dans l'architecture : l'architecture d'application, qui touche au développement en J2EE et .net, l'ar-



François Hisquin, Directeur Général d'Octo Technology

chitecture d'intégration, avec la mise en place de socles EAI, l'architecture dans le domaine de la sécurité (gestion des identités, SSO) et une activité plus transverse sur l'architecture des systèmes d'information, qui consiste en des missions sur l'analyse d'impact de nouveaux projets, la pérennité du système. Ces dernières missions sont essentiellement réali-

sées dans le domaine bancaire, confronté explique François Hisquin, à de vrais problèmes stratégiques de découpage entre production et distribution.

## Un fort niveau d'exigence

La croissance d'Octo est étroitement liée au recrutement : la société prévoit de recruter une quinzaine d'architectes en 2005 et souligne qu'elle a un "niveau d'exigence fort". François Hisquin indique que l'essentiel des recrutements provient de candidatures spontanées, Octo technology étant désormais "vu comme un des meilleurs cabinets d'architectes de la place" et reconnu pour ses projets dans le monde Open Source (y compris en .net !)

Les profils recherchés sont ceux de juniors (0 à 2 ans d'expérience) passionnés "qui aiment la matière première qu'est l'informatique", investis dans le monde Open Source, signe, selon François Hisquin, des vrais mordus, et qui doivent bien sûr avoir fait de la programmation. Deuxième profil, des seniors, plus expérimentés donc, cinq à dix ans d'expérience, "avoir codé, aimer ça, avoir vu un certain nombre de choses dans les systèmes d'information, de l'application à l'intégration". François Hisquin précise rechercher "avant tout un état d'esprit" : les nouvelles recrues doivent aimer construire, mais en respectant l'existant. L'objectif de François Hisquin est de compter une centaine de collaborateurs d'ici fin 2007, ou 2008 et 150 à terme ; un petit pôle d'experts en développement pourrait être créé.

■ Carole Pitrás

## Une filiale dédiée aux prestations offshore

**O**cto Technology a créé il y a deux ans Pivolis, une filiale dirigée par Vincent Massol, dédiée au développement offshore en Inde et en Roumanie. François Hisquin explique que lorsque Octo Technology intervenait sur des projets d'architecture, des prestations de développement étaient régulièrement demandées. Considérant que ce n'était pas le métier d'Octo et ne voulant pas devenir une SSII comme les autres, il a donc décidé de passer des accords de partenariat avec des sociétés roumaines et indiennes, de mettre en place une approche modulaire de la

construction d'application et un pilotage de la réalisation. A donc été développée une méthodologie spécifique : AOSD (Agile Offshore Software Development) qui permet, explique la société, "de développer de façon collaborative entre plusieurs équipes réparties géographiquement. L'AOSD met en œuvre les meilleures pratiques et outils issus de l'Open Source et des méthodologies Agiles (RUP et XP notamment) : itérations courtes, intégration continue, automatisation des tests, garantissant la stabilité et la qualité de l'application finale."

# Le webmaster existe-il encore ?

*Point de départ de cette enquête : qu'est devenu le webmaster ? Enfant chéri de la bulle internet, à la croisée des chemins de la technique, du design et du contenu, cet être protéiforme semble avoir disparu ou désigner désormais un simple animateur de site. En revanche, compte tenu de la prolifération des applications web au sein des entreprises, les développeurs, chefs de projet et autres directeurs de projets spécialisés dans les technologies web, ont le vent en poupe.*

Un ouvrage intitulé : les métiers de la net Economie\*, paru en 2001, répertoriait plus d'une trentaine de métiers, liés à la technologie (développeur web, chef de projet web, architecte réseau, ingénieur systèmes), au graphisme (graphiste multimedia, webdesigner), à la gestion de contenu (responsable éditorial, animateur de forum, cyberdocumentaliste), ou à des métiers aussi divers que juriste internet, chef de publicité en ligne, etc. Premier constat : le webmaster n'existe plus...en supposant qu'il ait réellement existé un jour, tant le terme recouvrait des réalités diverses. L'ouvrage suscitait déjà que "selon l'organisation et la taille du site sur lequel il travaille, un webmaster peut être chargé de l'éditorial ou de la technique. Voire des deux..." aujourd'hui, une recherche sur les sites d'offres d'emploi (les jeudis.com, Monster ou encore Cadremploi), avec comme mot clé webmaster se solde par un nombre de réponse oscillant entre zéro et cinq. Quelques postes de webmaster subsistent, souvent au sein de grandes entreprises, et il s'agit le plus souvent d'un rôle d'animation plutôt que de savoir-faire technique.

Manager conseil chez Micropole Univer,



**Thierry Guénard** est issu du monde de la communication et travaille sur les impacts de la mise en place de portails, sur différents aspects : interface d'utilisation, stratégie

éditoriale, accompagnement de projet, mais aussi conduite du changement. Il lie l'évolution des métiers à celle de l'utilisation des technologies Web au sein des entreprises. Dans un premier temps, elles étaient cantonnées à un support " pour améliorer la visibili-

té et la communication d'une entreprise ", et " en dehors de son système d'information ". "Le webmaster était alors le roi, il pouvait construire un site sans contraintes techniques". S'il était plutôt orienté sur la technique on pouvait néanmoins se poser des problèmes de pertinence sur les aspects graphiques. Ces conditions ont sensiblement évolué au fur et à mesure de l'introduction des technologies web dans les systèmes d'information et les stratégies d'entreprises, avec par exemple les intranet, puis les portails d'entreprise. Aujourd'hui, un projet web " est un projet informatique à part entière ". Selon Thierry Guénard, ce monde se progicalise autour des offres portails. Entre temps " le webmaster a été largué " et de nouveaux métiers apparaissent, liés à l'expertise sur ces outils, sur l'entretien et l'animation de site et de portails. Auparavant, le webmaster était le passage obligé pour mettre l'information en ligne, désormais, les outils de gestion de contenu, comme Vignette, rapprochent la mise en ligne de l'information du propriétaire de cette information. Concrètement, les fonctionnels mettent l'information en ligne, avec des systèmes de workflow. Ce qui peut faire naître une nouvelle problématique : comment créer dans ces différents services fonctionnels de l'entreprise, les réflexes de consultation et de dépôt de l'information. D'où, indique Thierry Guénard, le rôle d'animateurs, de responsables éditoriaux, de responsables qualité, qui peuvent veiller à une bonne alimentation d'un site, à sa pertinence, à la hiérarchisation des informations. D'où des projets autour de notions de travail collaboratif, et la mise en place d'un état d'esprit collaboratif, qui requiert des profils plus fonctionnels que purement techniciens.

A la tête d'une société spécialisée dans le

développement d'applications (et non de sites) web, Pierre Cyril de Geyer remarque que "le terme de webmaster est très général. Cela peut aller de la personne qui met à jour un site internet sans rien connaître de l'html, et encore moins de la programmation, à un informaticien qui développe cet outil de gestion de site. "

Il distingue trois catégories : le webmaster éditorial, " un littéraire, qui écrit sur Internet mais pourrait tout aussi bien écrire sur d'autres supports ", le webmaster graphiste, " un artiste : il crée des visuels et les met en place. C'est généralement un transfuge du monde de la PAO" et enfin, le webmaster technique : "C'est un informaticien, il développe des logiciels et veille sur eux. Il développe des applications pour le Web mais pourrait tout aussi bien développer la même chose pour des applications locales. "

## Les technologies web sont partout !

Deuxième constat : les développeurs web sont donc des développeurs comme les autres, qui doivent désormais justifier de leur compétence, observe Nicolas Sorel, développeur indépendant des sites codes-sources.com. " en 2000, il était assez facile de se dire compétent et de s'improviser développeur de sites web. C'est devenu un vrai métier ". Et il faut souvent justifier de diplômes, ce que regrette un peu cet autodidacte, qui préfère la reconnaissance technique obtenue, par exemple, auprès de communautés. Les entreprises peuvent toutefois préférer une évolution qui va de pair avec l'évolution des sites et des technologies, ainsi que la prise en compte, par exemple, de la sécurité des sites, notion désormais impossible à négliger.

S'adressant à un public qui prend un malin plaisir à chercher la moindre faille en la matiè-

re, Nicolas Sorel souligne qu'il a dû s'adapter et avoir toujours une vision d'avance sur ce que certains "bidouilleurs" peuvent essayer de faire.

Toutefois, les sites ne sont finalement qu'une petite partie des développements web, qui via l'intranet et les portails, notamment, touchent désormais beaucoup d'applications au sein des entreprises. "Je pense que l'évolution va dans le sens d'une complexification des applications développées", souligne Cyril Pierre de Geyer. " Le webmaster technique ne développe plus seulement des applications pour Internet, mais il développe des logiciels à part entière pour l'entreprise. Ces logiciels se basent sur les outils internet mais sont aussi complexes que tout autre logiciel basé sur une autre plate-forme "

Responsable des ressources humaines au sein de SQLI, SSII qui dispose également d'un "studio" pour les aspects graphiques des projets web, **Alexandra Lecordier** indique que tous les types de profils peuvent intervenir sur un projet web : consultant fonctionnel, architecte, ingénieur en développement, chef de projet, directeur de projet et, pour la partie studio des web designers, directeurs artistiques ou autres développeurs Flash. Peuvent également intervenir un ergonomiste ou des consultants éditoriaux.

Le Studio SQLI compte une quarantaine de collaborateurs, sur un effectif total de plus de 600 personnes. Un " monde à part ", selon Alexandra Lecordier, composé notamment de créatifs issus d'écoles de design, alors que la plupart des collaborateurs sont des ingénieurs.



web, **Alexandra Lecordier** indique que tous les types de profils peuvent intervenir sur un projet web : consultant fonctionnel, architecte, ingénieur en développement, chef de projet, directeur de projet et, pour la partie studio des web designers, directeurs artistiques ou autres développeurs Flash. Peuvent également intervenir un ergonomiste ou des consultants éditoriaux.

Le Studio SQLI compte une quarantaine de collaborateurs, sur un effectif total de plus de 600 personnes. Un " monde à part ", selon Alexandra Lecordier, composé notamment de créatifs issus d'écoles de design, alors que la plupart des collaborateurs sont des ingénieurs.



Selon la définition de Syntec Informatique " un webdesigner conçoit l'identité visuelle des sites internet. Sous la direction du webmaster, ce professionnel de l'internet définit une charte graphique, crée des boutons, des pictogrammes ou encore des bannières. S'il n'est pas forcément informaticien, il doit cependant maîtriser parfaitement le développement dans le domaine du web, les outils de programmation et les logiciels spécifiques (HTML, Flash, Java, Dreamweaver, ou encore Photoshop). Responsable du confort de navigation, il doit également déterminer l'ergonomie du site et bannir les solutions impliquant des temps de chargement trop longs, qui se soldent par la fuite des internautes... ". Ces profils orientés graphistes sont les plus nombreux sur le marché...mais la qualité n'est pas toujours au rendez-vous note Alexandra Lecordier ; ce type de profil est également moins bien rémunéré : parfois moins de 20 KE annuel, cela peut aller jusqu'à 30, voire 35 KE avec de l'expérience, ou si des fonctions plus proches de celles du webmaster y sont associées, relevant donc qu'une quasi double compétence. Assurant des formations, principalement sur PHP, Cyril Pierre de Geyer remarque qu'il y a deux types de profils qui s'y intéressent : des graphistes voulant ajouter une nouvelle corde à leur arc et, paral-

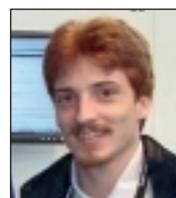
lèlement à ce qu'ils réalisent en Html, veulent se former aux technologies web que sont ASP.Net et PHP pour créer des sites dynamiques. Deuxième population : des informaticiens, qui maîtrisent C, Java ou ASP et veulent se former aux JSP ou à PHP.

### Trois technologies à l'honneur

Sans surprise, et par ordre alphabétique pour qu'on ne me soupçonne pas de partialité, Asp.net, Java (J2EE, JSP, etc.) et PHP... Alexandra Lecordier relève que, sur la partie technique, l'évolution des profils suit l'évolution technologique, d'où davantage de recrutements sur .net ces derniers mois, au côté des traditionnels PHP et J2EE. Les salaires proposés varient selon l'expérience : entre 28 et 33K€, en moyenne pour des profils débutant, moins de trois ans d'expérience, et peuvent aller jusqu'à 55 K€ pour des experts sur l'une ou l'autre des technologies suscitées.



◀ **Cyril Pierre de Geyer et Nicolas Sorel**, respectivement spécialisés sur PHP et ASP.net soulignent la capacité des nouvelles versions à permettre aux graphistes ou designers, de modifier leur partie... sans toucher au code. Nicolas Sorel, qui teste .Net 2 et Visual Studio 2005 souligne que la nouvelle plate-forme proposée par Microsoft " per-



mettra à un designer de toucher à des fichiers de templates sans toucher au code ". Ce qui fait gagner du temps au développeur aussi, qui n'est plus tenu d'interpréter, souvent péniblement, les indications des graphistes. Sans parler de ceux qui faisaient cauchemarder les développeurs en utilisant des Dreamweaver ou autres logiciels qui " saccageaient le code ". Grand défi des feu web agences au début des années 2000, la collaboration entre ingénieurs et créatifs semble toujours aussi problématique...Autre évolution de cette version explique Nicolas Sorel : une plus grande facilité à travailler en équipe.

#### Pour en savoir plus

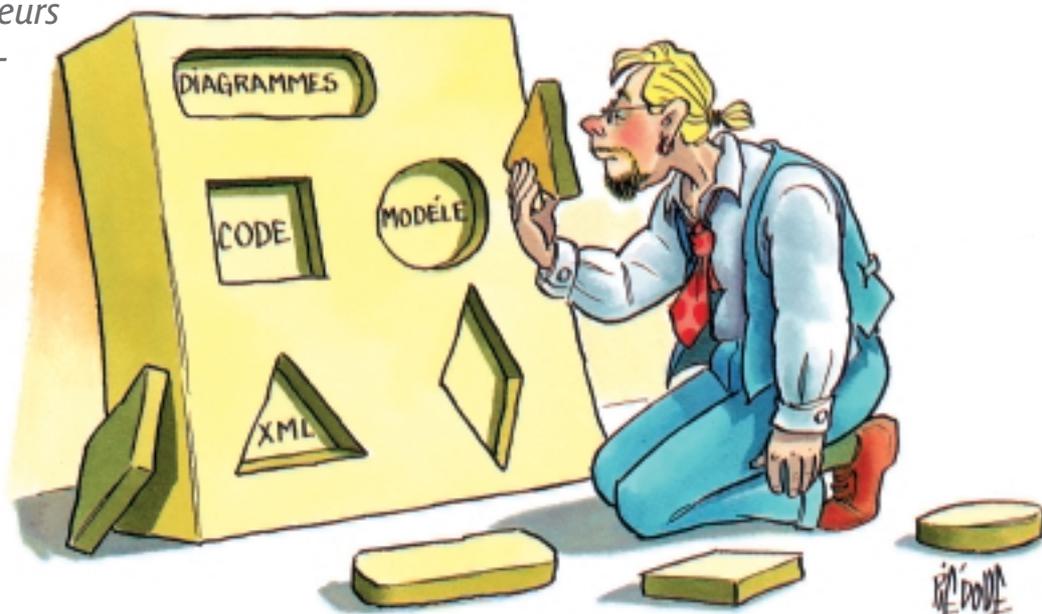
- Les métiers de la net Economie par Jean-Marc Engelhard
  - [www.passinformatique.com](http://www.passinformatique.com)
- Carole Pitras

## La passion du Web

**X**avier Vanneste est tombé dans l'informatique et le web... en faisant un BTS automatisé. Devenu très vite passionné, il a créé son propre site et s'est fait remarquer par une web agency qu'il a intégrée en 2000. Il a ensuite passé des certifications Microsoft qui ont validé son expérience, preuve qu'un passionné peut y arriver, même sans diplôme souligne-t-il. Depuis début 2005, il travaille chez Access It, société de 50 personnes basée à Villeneuve D'Ascq, comme développeur, sur des projets de portail internet et intranet, en utilisant essentiellement les technologies Microsoft et Flash. Il indique intervenir sur l'ensemble du processus du développement : analyse fonctionnelle, analyse technique, développement. Il a également des missions d'avant-vente, de conseil en choix des technologies et de formation. Son avenir, il le voit plutôt dans cette branche conseil.

# UML : à chacun son diagramme

*Depuis plus d'un an, les éditeurs spécialisés dans la modélisation s'emparaient de la seconde version d'UML, après une décade d'existence de la première mouture (une belle preuve de longévité). Fin 2004, les spécifications d'UML 2 étaient enfin disponibles. Depuis, les outils s'adaptent rapidement. Pourrait-il convaincre, là où UML n'a jamais réellement su le faire ?*



La modélisation, grâce à UML, a trouvé une harmonisation et une standardisation qui lui faisaient défaut. Aujourd'hui, le formalisme UML a définitivement conquis le monde informatique. Il fournit du formalisme et un vocabulaire commun au projet et valable pour l'ensemble des intervenants (le glossaire). UML est issu de la fusion de plusieurs méthodes durant les années 1990. UML sert avant tout à modéliser un système grâce à des diagrammes, le tout dans un contexte objet. L'intérêt d'UML est son abstraction aux langages et aux plate-formes. UML 1.x a été souvent cantonné à un rôle d'analyse et de pré-conception, même si des outils le liaient au code, notamment à la génération automatique. Il était perçu comme un outil d'architecture. Bref, il n'y avait que peu d'interaction entre la partie conception et l'analyse. On demeurait donc trop code centric. Or, le but de la modélisation est de passer à un niveau au-delà : le pilotage par modèle, ou model-driven. Le modèle UML 2 sert à chaque étape du projet, de l'analyse, au développement, aux tests, au déploiement.

UML 2 doit répondre à ce défi en introduisant de nouveaux diagrammes. Une des nouveautés importantes d'UML 2 concerne la sémantique d'action. Cela permet de visualiser / créer /

observer les algorithmes directement dans son modèle. UML 2 facilite donc la conceptualisation globale de son projet de bout en bout, d'une manière totalement standard et sur un modèle. À partir de ce modèle, vous pouvez générer des maquettes, des scénarios de tests et déploiements, etc. Si on pouvait déjà réaliser de la génération de code avec UML 1.x (avec ou sans modèle MDA), UML 2 va bien plus loin et doit permettre, en théorie du moins, d'augmenter le pourcentage de code généré.

## Au départ

UML nécessite obligatoirement une phase de formation, de pratique. Il ne faut jamais sous-estimer l'investissement à réaliser en temps et en argent. UML demeure compliqué, UML 2 l'est encore plus. On apprend de nouveaux réflexes et surtout une nouvelle manière de

penser le projet, la conception, l'architecture. La fabrication d'un modèle UML n'est possible que si on dispose d'un cahier des charges clair et précis. Le modèle peut mettre en évidence les carences de conception ou un manque fonctionnel. Le modèle, pour aboutir à un projet viable lors du codage, doit être le plus stable, le plus propre possible. On peut désormais prévoir le comportement du projet à partir du modèle (idéal pour la maquette).

UML peut apporter une réelle plus value à un projet si on détermine précisément les diagrammes et les ensembles adaptés à son projet. Tout n'est pas utile dans UML. C'est pour cela qu'il faut opter pour des outils de modélisation spécifiques à son métier. Si vous êtes développeur, prendre une solution orientée développeur. Éviter à tout prix les outils généralistes. L'orienté rôle est une notion très

## Objecteering : Merise toujours là

Pour Objecteering, UML gagne du terrain. Les entreprises et utilisateurs ne se posent plus de questions. Cependant Merise demeure présent et ceux l'utilisant encore ne veulent souvent pas changer. La croissance est directement liée au marché et surtout aux nouveaux projets. Depuis 2004, la société constate une forte croissance. L'éditeur fait évoluer ses outils régulièrement. Il dispose maintenant d'un support Java, C# et d'un reverse C++. Tout récemment, un modeleur MDA a été présenté afin d'en simplifier l'approche.

importante à ne jamais perdre de vue. UML est suffisamment souple pour s'adapter aux différents intervenants du projet et répondre aux attentes de telle catégorie.

UML / UML 2 dans le cadre de petits / moyens projets met en place une qualité logicielle, la possibilité de pouvoir qualifier le livrable, ou encore de fournir les métriques nécessaires lors de la génération du code, du codage. UML n'est pas qu'un simple outil à créer des modèles de projets mais aussi (et surtout ?) il permet de garantir une qualité logicielle dans la conception, le code, les tests, le déploiement, la maintenance.

■ **François Tonic**

## Telelogic : UML, un choix pérenne

Philippe Leblanc, spécialiste UML chez Telelogic nous livre son analyse et réflexion sur UML et le marché.

" On trouve beaucoup d'intérêt à UML. Un grand nombre d'entreprises possède une ou deux licences mais il n'y a pas de généralisation sur leurs projets. Il y a un coût de formation, d'appropriation. Aujourd'hui, les utilisateurs d'UML voient souvent cela comme une solution miracle, sans y mettre l'investissement nécessaire. UML a un succès mitigé à cause de ce manque d'investissement, de formations. Dans un outil de développement, par exemple, on ne peut pas se passer du compilateur. La modélisation, on peut s'en passer. Chez nous, l'outil le plus vendu actuellement concerne la gestion des exigences (DOORS). Les gens semblent plus convaincus par cette catégorie d'outils. Un outil UML exige un certain niveau de compétence. UML 2 est plus riche que l'UML 1.x, son spectre d'utilisation s'étend. Le plus difficile est de définir le, ou les sous-ensembles à utiliser dans son projet. Telelogic ne propose pas (comme d'autres éditeurs) différentes éditions selon les rôles, cependant, c'est une tendance que l'on suit. On regarde aussi les travaux autour d'Eclipse EMF. Il n'est pas encore implémenté dans notre outil. Son intégration se fera sur le long terme. UML est un choix pérenne. La première génération d'UML a duré presque 10 ans. Aujourd'hui, on est sur UML 2. La prochaine version arrivera dans un ou deux ans. Il faut savoir que UML 2 a été beaucoup poussé par les utilisateurs. Il répond bien à leurs demandes, à leurs attentes."

## LES PRINCIPAUX OUTILS DU MARCHÉ

### Outils Open Source

Nom	Remarques
<b>UML Object Modeller for Linux</b> <a href="http://uml.sourceforge.net/">http://uml.sourceforge.net/</a>	Un des meilleurs outils Linux en Open Source (fonctionne sous KDE)
<b>Omondo</b> <a href="http://www.omondo.com">http://www.omondo.com</a>	Plug-in de modélisation UML pour l'IDE Eclipse. Version entreprise prochainement disponible
<b>ArgoUML</b> <a href="http://argouml.tigris.org/">http://argouml.tigris.org/</a>	Modèleur en licence BSD. Implémentation UML presque complète.

### Outils commerciaux

Nom	Remarques
<b>Real-time Modeler</b> Artisan <a href="http://www.artisansw.com">http://www.artisansw.com</a>	Modèleur temps réel avec support multi-utilisateur
<b>AllFusion Component Modeler</b> Computer Associates <a href="http://www.ca.com">http://www.ca.com</a>	Modèleur UML dédié au eBusiness en entreprise. Un des outils les plus complets du marché
<b>Describe</b> Embarcadero <a href="http://www.embarcadero.com/">http://www.embarcadero.com/</a>	S'intègre aux IDE Java. Supporte le développement en groupe.
<b>Rhapsody</b> I-Logix <a href="http://www.ilogix.com">http://www.ilogix.com</a>	La gamme Rhapsody permet d'utiliser UML dans le cadre de développement d'applications embarquées
<b>Rational ROSE / XDE et Modeler</b> <a href="http://www.rational.com">http://www.rational.com</a>	L'éditeur propose toujours la modélisation dans les outils ROSE et XDE, fonctionnant en Java et .NET. La suite Atlantic possède son propre modèleur Rational Software Modeler. Supporte UML 2
<b>Objecteering UML Modeler</b> <a href="http://www.objecteering.com">http://www.objecteering.com</a>	Se décline en plusieurs versions. La version personal est gratuite.
<b>PowerDesigner</b> Sybase <a href="http://www.sybase.com">http://www.sybase.com</a>	Outil de conception UML orienté entreprise. Support du ebXML
<b>Together Solo</b> <a href="http://www.borland.com">http://www.borland.com</a>	Modèleur UML pour les développeurs indépendants et les petites équipes. Existe en différentes déclinaisons.
<b>Visual Paradigm</b> <a href="http://www.visual-paradigm.com/">http://www.visual-paradigm.com/</a>	Outil visuel UML. Il incorpore un IDE en plus du modèleur UML.
<b>Visual UML Standard Edition</b> Visual Object Modelers <a href="http://www.visualuml.com">http://www.visualuml.com</a>	Génération de diagrammes et reverse engineering pour C++, C# et Java
<b>Poseidon for UML</b> <a href="http://www.gentleware.com/">http://www.gentleware.com/</a>	Environnement UML bien connu. Prix accessible, existe en plusieurs versions.
<b>TAU</b> <a href="http://www.telelogic.com">http://www.telelogic.com</a>	Environnement UML 2.0. Existe en 2 déclinaisons, possibilité de temps réel

## Borland : prendre en compte les rôles de chacun

Bruno de Combiens de Borland France commente l'approche de l'éditeur dans le monde UML " Avant on faisait de l'UML pour faire de l'UML. Maintenant, c'est pour faire autre chose. On peut générer du code. Il est possible de réaliser du reverse engineering pour en déduire l'architecture, des métriques ou encore la documentation. Il est possible d'analyser en profondeur, par exemple, en voyant la profondeur de l'héritage des classes. Utile dans le cadre d'une maintenance. Avec UML, on peut qualifier le livrable. De plus, il est possible de réaliser du refactoring visuel. Je pense qu'il y a un usage à la carte d'UML. Il faut avoir une approche orientée rôle. Un développeur, un architecte n'ont pas la même problématique, n'ont pas besoin des mêmes diagrammes. "

## Et l'approche Microsoft ?

Microsoft a implémenté sa propre solution de modélisation dans Team System. L'éditeur critique la complexité d'UML 2 pour les concepteurs, les développeurs. Les deux modèles sont donc incompatibles. Microsoft avait-il réellement intérêt à utiliser UML dans son outil ? Pas sûr. Le format propriétaire peut parfaitement fonctionner. De plus, il est à prévoir que les deux formats communiqueront via des outils de conversion ou des passerelles. Mais il est encore bien trop tôt pour mesurer l'impact de la méthode Microsoft sur le marché.

# “UML améliore la production et la fiabilité des applications”



*Connu pour son travail sur UML et UML 2.0, Bran Selic a beaucoup travaillé avec l'OMG, à l'origine de ce langage de modélisation, et activement œuvré pour l'architecture MDA. Il travaille aujourd'hui à l'implémentation d'UML 2 dans les futurs outils d'IBM Rational et donne son avis sur différents aspects d'UML et du marché.*

**Programmez ! : De quelle manière UML s'est-il imposé sur le marché ces dernières années ?**

**Bran Selic :** Lorsque UML fut adopté comme standard en 1997, il a été rapidement utilisé, notamment par les chercheurs. De par ma position au sein de l'équipe responsable



d'UML 2, j'ai pu voir que le taux d'adoption s'est rapidement accéléré. J'ai visité de nombreux clients qui utilisaient

déjà UML d'une manière ou d'une autre, même si je n'ai pas de données disponibles sur le degré de pénétration d'UML. Cependant, j'estime qu'il y a de nombreux professionnels du logiciel possédant un certain degré de connaissance d'UML. Je collabore également avec de nombreuses universités à travers le monde et je peux confirmer que pratiquement tous les cursus informatique que j'ai vus, proposent UML. Selon moi, il existe un véritable besoin pour une forme d'expression partagée, qui tranche avec la confusion et les changements perpétuels induits par des technologies des outils et des méthodes pléthoriques. Il convient de noter que tous les utilisateurs d'UML ne s'en servent pas de la même manière. Martin Fowler avait identifié trois types d'utilisateurs : les sketchers utilisant UML pour la formulation et la communication des idées, les blueprinters, pour les caractéristiques détaillées sur la conception aux codeurs et enfin les programmeurs qui utilisent actuellement UML dans les outils de très haut niveau. Cependant, indépendamment de la manière dont ils s'en servent, l'expérience générale avec UML a été qu'il améliore la productivité et la fiabilité des applications. Dans certains cas, les améliorations ont un facteur de 2 à 3.

Cependant, nous devons faire attention à ne pas répéter les erreurs de l'histoire. UML est arrivé, parce qu'au milieu des années 1990, il y avait trop de langages de modélisation, plus de 100 selon un décompte, ayant pour résultante une calamiteuse dispersion de l'expertise et des outils. Je ne dis pas qu'UML est la panacée. Il y a bien entendu des applications pour lesquelles il n'est pas la meilleure solution.

**P ! : Il y a le positionnement " entreprise " et le positionnement " développeur " des outils UML. Or, pour le développeur lambda, UML demeure compliqué et complexe à utiliser. Qu'en pensez-vous ?**

**BS :** Je crois que beaucoup de ceux qui pensent cela répètent simplement ce qu'ils ont entendu auprès d'autres personnes, ou qu'ils n'ont pas pris le temps de voir qu'UML est très modulaire. C'est comme les langues. Vous n'avez pas besoin de connaître tout le Français ou l'Anglais. Nous sommes nombreux à utiliser un pourcentage limité du vocabulaire de notre langue et on peut être cependant très efficace. UML est structuré de sorte qu'on puisse, non seulement prendre les éléments qui nous intéressent, mais aussi ignorer le reste en toute sécurité. Et comme pour sa propre langue, à mesure que l'expérience avec UML augmente, nous ressentons le besoin d'être plus complets, rajoutant de nouveaux mots de vocabulaire. Par exemple, pour modéliser un process business on utilisera les capacités de modélisation d'activité, tout en ignorant le reste. Cette modularité du langage a été accrue dans UML 2. Les problèmes complexes nécessitent les bons outils. Par exemple, il serait peu réaliste de construire un gratte-ciel

sans les machines appropriées. Aujourd'hui, la construction d'applications sophistiquées requiert de puissants outils. Regardez Java. À ses débuts, il s'agissait d'un langage très simple. Mais depuis, il s'est beaucoup étoffé.

**P ! : Pour que le développeur l'utilise plus intensivement, ne faudrait-il pas le simplifier et proposer un " langage " plus proche de la réalité ?**

**BS :** UML a été conçu directement à partir de la réalité des développeurs. Naturellement, il existe diverses réalités de développeurs. Pour répondre à cette question, un des fondamentaux à retenir, est qu'UML fournit une représentation visuelle du code du programme. Par exemple, un outil comme IBM Rational System Architect peut automatiquement extraire les diagrammes de classes de Java ou d'un code similaire, qui montre des relations entre les différents éléments de la structure du programme, ou les interactions complexes du code en des diagrammes UML de séquences. Cependant, notre expérience à IBM est qu'une fois que les avantages de modelage deviennent évidents aux développeurs, ils sont souvent prêts à exploiter les possibilités additionnelles d'abstraction et d'automation que les outils et le langage fournissent.



**P ! : UML2 va-t-il pouvoir imposer définitivement UML ?**

**BS :** Je suis absolument convaincu qu'UML 2 augmentera grandement le niveau d'usage d'UML. Il y a de nombreuses raisons à ceci, une meilleure montée en charge, une précision accrue, une modularité améliorée, etc.

■ *propos recueillis par François Tonic*

# UML 2 : un pas vers une conception d'applications par la construction de modèles

*Conçu à l'origine comme une synthèse de différentes notations existantes, UML subit aujourd'hui son premier lifting avec la version UML 2.0.*

L'objectif principal d'UML est double. D'un part il s'agit de favoriser la modélisation des systèmes. Cette étape de conception et de définition d'une architecture est désormais devenue incontournable face à la complexité croissante des systèmes informatiques, notamment dans les domaines des applications web et des applications temps réel dans le monde des télécoms. Pour faire face à cette complexité qui tend à rendre les systèmes informatiques inintelligibles, le seul outil dont dispose le cerveau humain est l'abstraction. D'autre part UML a pour objectif essentiel de faciliter la communication entre intervenants sur un projet, qu'ils soient techniciens ou spécialistes fonctionnels. Les 9 diagrammes d'UML 1.0 existent toujours dans UML 2.0 (l'un d'entre eux ayant été renommé) sont regroupés en deux grandes catégories :

## 1. Les diagrammes de structure

- Les diagramme de classes : utilisés pour la représentation des classes, interfaces et leurs relations d'association, d'héritage, d'implémentation ou d'utilisation. Ce sont les diagrammes les plus utilisés. Ils sont susceptibles d'intervenir à tous les niveaux d'abstraction, aussi bien en phase d'analyse que de conception et de génération de code.
- Les diagrammes d'objets : représentent une photographie des relations entre instances de plusieurs classes à un instant donné.
- Les diagrammes de composants : représentent typiquement les relations et les contrats entre composants exécutables d'un système.
- Les diagrammes de déploiement : représentent la topologie d'un réseau informatique et la répartition de divers composants sur les différents nœuds de ce réseau.

## 2. Les diagrammes dynamiques

- Les diagrammes des cas d'utilisation : per-

mettent de structurer la formulation des besoins des utilisateurs vis-à-vis d'un système en unités réutilisables.

- Les diagrammes d'activité : correspondent dans les grandes lignes à la bonne vieille notion d'organigramme. Ils permettent en particulier de modéliser l'exécution de tâches parallèles.
- Les diagrammes d'état : représentent le comportement dynamique d'un seul objet. Ils spécifient le cycle de vie d'un objet et la manière dont celui-ci réagit à la survenue de différents événements.
- Les diagrammes de séquence : représentent une succession chronologique d'échanges de messages entre une collection d'objets. Ils sont souvent utilisés pour documenter le déroulement d'un scénario d'un cas d'utilisation.
- Les diagrammes de communication (de collaboration dans UML 1.0) : contiennent la même information que les diagrammes de séquence, mais en insistant sur l'aspect structurel des objets en interaction plutôt que sur l'aspect chronologique.

On estime actuellement que le taux de pénétration global d'UML dans l'industrie informatique est de l'ordre de 10%. Deux causes principales à la relative faiblesse de ce chiffre peuvent être invoquées.

Les aspects culturels et l'investissement dans d'autres technologies existantes d'une part (Merise en France par exemple), et d'autre part, la complexité des outils de modélisation. Leur maîtrise requiert souvent plusieurs mois de pratique quotidienne pour être véritablement opérationnelle. Ce chiffre de 10% pourrait être significativement revu à la hausse si l'on ne considérait que la population qui pratique effectivement la modélisation. Dans un domaine spécialisé comme celui des modèles de conception (design patterns) UML fait aujourd'hui l'unanimité.

## Quelles motivations derrière UML 2 ?

Trois idées forces sont à l'origine de la spécification UML 2.

### L'approche MDA

On désigne par le sigle MDA (Model Driven Architecture) une réorientation profonde de la manière dont nous construisons aujourd'hui des systèmes informatiques. Il s'agit d'une initiative ambitieuse de l'OMG (Object Management Group) qui vise à promouvoir la construction d'applications par modélisation plutôt que par codage. Pour faire face à la pléthore de plateformes de middleware actuelles, l'OMG propose dans son approche MDA d'utiliser deux types de modèles UML. Des modèles UML indépendants de la plate-forme, les PIM (Platform Independent Model), qui définissent de manière très rigoureuse les fonctionnalités métiers et le comportement d'une application, sans toutefois comporter aucun aspect technique. Ces modèles PIM seront alors traduits en modèles dépendant de la plate-forme, les PSM (Platform Specific Models) par des outils qui utilisent des mappings standardisés par l'OMG. Tout l'intérêt est là : la standardisation du mapping !

### L'automatisation

L'automatisation dans la production de code, on le voit, est étroitement liée à l'approche MDA, elle doit permettre de gagner en fiabilité et en robustesse. Toutefois l'approche MDA ne se réduit pas à une simple génération de code. Un accent tout particulier est mis sur la possibilité de pouvoir littéralement exécuter des modèles abstraits et incomplets dans une phase précoce, pour valider des choix d'architecture. Des tests unitaires et de spécification doivent être également générés. Cette étape de conversion n'est possible qu'au prix d'une plus grande rigueur sémantique du langage, ce à quoi UML 2 s'attache. On assiste donc avec UML 2 à un passa-

ge progressif d'un langage descriptif à un langage prescriptif. La précision de la sémantique est obtenue grâce à une simplification de la notion de profil UML. Rappelons qu'un profil UML constitue un mécanisme générique d'extension d'UML. Les sages de l'OMG demeurent toutefois réalistes et considèrent que, dans un premier temps, une phase manuelle de " fine-tuning " restera nécessaire sur les PSM générés automatiquement ! L'espoir étant qu'à long terme (~10-20 ans) cette phase se réduise avec la maturité des outils et des algorithmes. UML 2 peut donc être considéré comme un ingrédient clé de la démarche MDA de l'OMG.

**Elever le niveau d'abstraction**

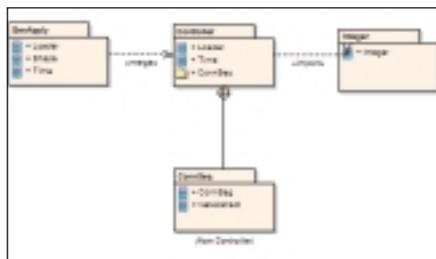
Enfin UML 2 renforce la capacité d'abstraction du langage pour permettre la modélisation de très grands systèmes. Le nouveau diagramme de modules et le nouveau diagramme global d'activité en sont l'illustration.

**Les nouveaux diagrammes dans UML 2**

Précisons d'emblée qu'UML 2 ne nécessite pas un long réapprentissage, si l'on connaît déjà UML. UML 2 apporte quatre nouveaux diagrammes et en enrichit certains, mais rien n'oblige à connaître les nouvelles caractéristiques. Les projets de très grande taille qui impliquent des environnements hétérogènes et des architectures distribuées pourront immédiatement bénéficier de la plus grande puissance d'abstraction d'UML 2, ceci, indépendamment de la stratégie à long terme MDA décrite ci-dessus. Voici une liste des principaux éléments des nouveaux diagrammes d'UML 2 :

**1. Les nouveaux diagrammes de structure**

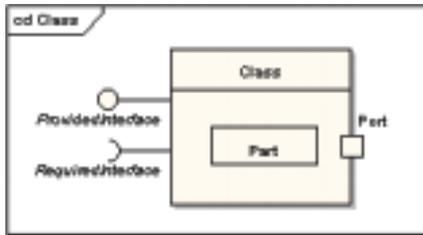
- Les diagrammes de module : Ils permettent de représenter la hiérarchie d'un projet et leurs dépendances mutuelles. Voici un exemple : *Figure 1. Un diagramme de module.*



Les packages peuvent contenir n'importe quel type d'éléments UML. La relation " merge " signifie qu'on rajoute récursivement tout le

contenu du package cible au package source. La relation " import " signifie que l'élément désigné dans le package cible est importé dans le package source.

- Les diagrammes de structure composite : Ces diagrammes montrent la structure interne d'un classificateur (typiquement une classe), y compris ses points d'interaction avec le reste du système. Voici un exemple générique :



*Figure 2 : Une classe dans un diagramme de structure composite.*

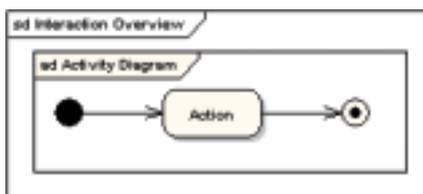
Une partie représente une instance possédée par le classificateur conteneur. Un port est un élément typé qui représente une partie visible du classificateur. On distingue la notion d'interface requise de celle d'interface fournie.

**2. Les nouveaux diagrammes dynamiques**

Les diagrammes globaux d'interaction : Il s'agit d'une nouvelle forme de diagrammes d'activités dans lesquels les nœuds sont des diagrammes d'interaction. Par diagramme d'interaction on désigne la catégorie des diagrammes constituée des diagrammes de séquence, de communication, de temps, et des diagrammes globaux d'interaction. La notation demeure essentiellement la même que pour les diagrammes d'activité classique, à l'exception de deux nouveaux éléments. La Figure 3 représente un nœud faisant référence à un diagramme d'activité, la Figure 4 montre un sous-diagramme d'activité explicite.

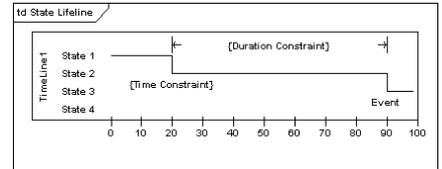


*Figure 3 : Référence à un diagramme d'activité.*

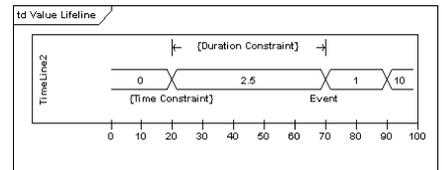


*Figure 4 : Un sous-diagramme d'activité.*

- Les diagrammes de temps : Ils sont conçus pour représenter l'évolution au cours du temps de la valeur ou de l'état d'un ou plusieurs éléments.



*Figure 5 : Evolution de l'état d'un objet au cours du temps.*



*Figure 6 : Evolution d'une valeur au cours du temps.*

L'intérêt de ces nouveaux diagrammes, à l'exception du dernier, ne sautera pas forcément aux yeux du profane. Il paraîtra même discutable aux plus sceptiques. Répétons cependant que le principal intérêt d'UML 2 réside probablement moins dans son expressivité accrue que dans la stratégie à long terme visant à en faire l'élément clé de l'approche MDA.

Mentionnons encore qu' UML 2 a été subdivisé en " Language Units ", chaque unité regroupant des concepts reliés, destinés à en faciliter l'apprentissage et la mise en œuvre. Le "State Machine Language" et le " Activities Language " en sont des exemples.

**Les outils compatibles UML 2**

Voici une liste non exhaustive d'outils compatibles UML 2 :

- Rational Software Architect d'IBM : Un outil de conception et de construction qui utilise les principes MDA et UML.
- Together Designer de Borland : Un outil destiné aussi bien aux développeurs qu'aux architectes, pour la construction d'applications à partir de modèles UML.
- Enterprise Architect de Sparx Systems : Un outil bon marché, compatible, permettant la construction des 13 diagrammes UML 2. Disponible aussi en temps que plug-in Eclipse.
- ARTISAN's Real-time Studio : Modélisation UML pour des systèmes temps réel embarqués. Une liste plus complète est disponible sur le site UML de l'OMG <http://www.uml.org/>.

**Conclusion**

La nécessité de modéliser les applications fait actuellement l'objet d'un large consensus. UML est sans aucun doute l'approche la plus répandue. La complexité des outils de modélisation qui demandent un apprentissage souvent long de plusieurs mois en freine cependant une plus large adoption. Entendons-nous : la complexité des outils n'est pas en cause ici, elle n'est que le simple reflet de la complexité des systèmes que nous construisons. L'adoption plus large d'UML et de MDA passe sans doute par un changement de mentalité, aussi bien chez les développeurs que chez les décideurs des DSI. Les développeurs devraient considérer UML

comme un bagage standard et non comme cela est encore trop souvent le cas comme une coquetterie académique, inutile dans la vraie vie. Les décideurs, quant à eux, doivent bien réaliser que la modélisation UML ne s'improvise pas en cinq minutes. Il s'agit d'un métier nécessitant une véritable formation et une approche nettement plus conceptuelle que la programmation traditionnelle. Investir dans la formation continue des équipes et le recrutement de profils conceptuels, comprendre que l'abstraction peut être rentable, voilà ce que UML 2 et MDA suggèrent.

UML 2 constitue la première évolution significative d'UML depuis sa normalisation en 1997.

Dans l'immédiat, cette évolution apportera surtout des réponses aux besoins de modélisations pour les très gros systèmes. A plus long terme, UML 2 s'inscrit dans le courant de la MDA qui se profile à l'horizon comme une nouvelle manière plus conceptuelle de construire des applications au moyen de modèles. L'OMG avec MDA et UML 2 nous promet des applications "future proof" (littéralement : "à l'abri des évolutions technologiques futures") concept révolutionnaire s'il en est. Faut-il y croire ? L'avenir le dira.

■ Pirmin Lemberger - Consultant chez SQLi

# Réussir la modélisation UML des phases amont

## Techniques de " prémodélisation " : un pont vers le modèle

*Selon le Standish Group(1) , développer une application logicielle, fait encourir un risque d'échec de 23%, un risque de dérapage (en dehors des prévisions fonctionnelles ou de délai/budget initiales) de 49%, et une chance de succès de 28%.*

Le succès ou l'échec d'un développement se jouent essentiellement lors des phases amont du développement logiciel. C'est là où les acteurs principaux qui déterminent les choix fondamentaux – les décideurs, les utilisateurs, les maîtres d'ouvrage – bâtissent l'objectif stratégique ; c'est là où le contour du domaine à traiter, et où les engagements fonctionnels, budgétaires, de planning, et architecturaux apparaissent.

A ce stade, l'enjeu est de définir des objectifs compris par tous (2), réalistes, recouvrant exhaustivement la vision initiatrice du projet. Face à une analyse préliminaire réussie, à une liste d'exigences bien formalisée et à une définition du domaine suffisamment complète, la réalisation du développement permet d'isoler et de réduire les risques.

### Objectif de la phase préliminaire au développement

Dans la phase préliminaire, un enjeu majeur consiste à utiliser des représentations permettant un dialogue entre les parties impliquées comme les utilisateurs, les analystes, les responsables hiérarchiques et les experts du domaine. L'emploi exagéré de techniques de modélisation, comme UML, constitue un obstacle à la compréhension pour certains intervenants, et ira à l'encontre de leur implication dans la modélisation ou les revues. A contrario, l'absence d'une méthode rigoureuse de représentation empêchera la collecte d'une information précise, cohérente et pertinente, que l'on peut abstraire ou détailler selon les niveaux de dialogue. On en viendra donc à combiner plusieurs techniques, pour adresser spécifique-

ment chaque type de problème, mais aussi pour fournir des représentations compréhensibles par les diverses catégories de personnes impliquées dans les phases amont.

La phase préliminaire fournira des représentations qui constitueront la base d'un contrat (3) pour les développeurs de l'application.

Le rappel "littéraire" de l'expression de besoin initial est une partie importante de ce contrat : il sera souvent nécessaire de s'y référer pour faire des choix, lorsque l'on sera accaparé par les difficultés techniques et pratiques de la réalisation.

### Techniques employées lors de la phase préliminaire

#### Vue générale

Il est dangereux de produire prématurément des modèles. La modélisation impose une

(1) - Chaos Report, Standish group, 2001

(2) - Dans certains cas, comme par exemple sur des domaines innovants en termes de produits et services, l'objectif n'est initialement pas totalement élucidé. Il s'avère alors nécessaire de définir les objectifs identifiés, et de procéder itérativement dans les étapes ultérieures jusqu'à obtention d'un cahier des charges complété.

(3) - Il ne s'agit que d'un des nombreux éléments du contrat, qui devra préciser par exemple le partage des rôles et responsabilités ainsi que les modalités d'organisation lors de la phase de réalisation.

vision " formelle " du problème, détermine des choix qui peuvent s'avérer prématurés, et produit facilement un assemblage de boîtes et de liens non représentatifs de la réalité du problème et des objectifs. Il est avant tout fondamental de répertorier une information pertinente sur la nature du problème à traiter et sur les objectifs. En revanche, il est très intéressant de réaliser des modèles, car ils formalisent le problème, structurent le cadre du développement, et permettent d'identifier des manques de l'analyse préalable en contraignant l'analyste à décrire l'implicite, et à considérer les aspects significatifs du processus. Ils permettent de rendre cohérente une base initiale trop informelle. Outre la formalisation nécessaire à la réalisation du logiciel, le modèle apporte aussi au métier la maîtrise de ses propres concepts et l'explicitation de ses procédures. Sans prétendre à l'exhaustivité, les techniques les plus fréquemment utilisées sont présentées ci-dessous. En pratique, elles sont employées selon un grand nombre de variantes, y compris dans leur dénomination. On assiste cependant, et pour chacune d'entre elles, à des mouvements d'uniformisation et de consolidation, se traduisant soit par des outillages ayant des fonctionnalités similaires, soit par des livres dédiés à ces pratiques, ou même des efforts de standardisation comme ceux de l'OMG(4) sur l'approche " system

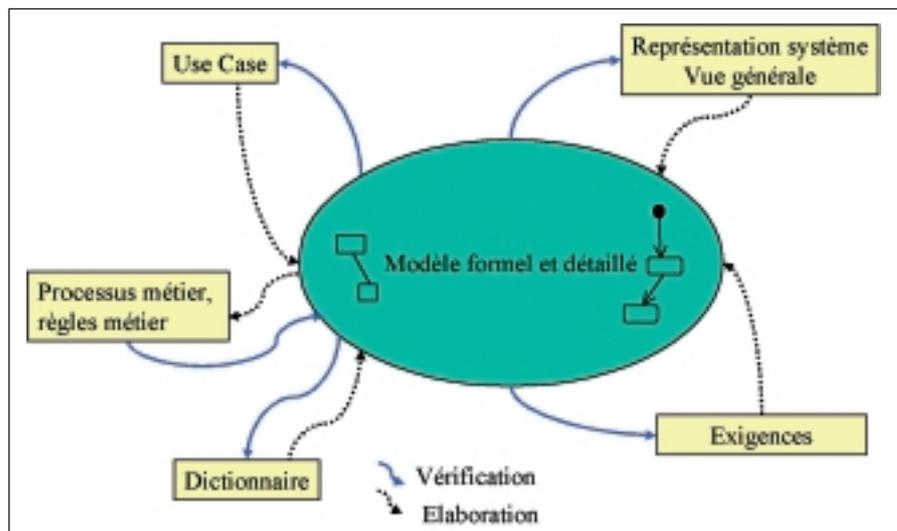


Figure 1 - Association de modèles formels et informels en phase préliminaire

engineering ", " business rules ", ou encore " Business Process Modeling " .

- Dictionnaire : termes et définitions pertinentes relatifs au système. Le dictionnaire est un moyen très simple et efficace pour décrire le domaine de l'application.
- Approche systémique : représentation globale du système présent et futur. Fournit les responsabilités du système. Derrière les préoccupations de cette approche, on retrouvera les techniques de l'analyse systémique, déjà connues dans l'approche Merise, les approches de " system engineering " pratiquées dans le

domaine technique, et les techniques de cartographie des applications utilisées dans le domaine des systèmes d'information.

- Analyse des besoins et des exigences : classifiés, formalisés, maintenus, tracés, on trouve des pratiques très similaires fournies par plusieurs ateliers et préconisées par plusieurs démarches.
- Modélisation des processus métier : Cette technique est essentielle pour la définition des procédures d'une organisation. Certains outils apportent une notation spécifique à cette technique, mais UML fournit un formalisme adapté, proche des modèles de workflow, appelé "Activity Diagram".
- Règles métier : description des règles fondamentales régissant le fonctionnement d'une organisation. Plusieurs familles d'outils supportent cette approche sous des formes très différentes. Le langage UML fournit une base pour les exprimer, grâce aux notions de contrainte, d'invariant, de pré ou post conditions.
- Cas d'utilisation : description des grands cas d'utilisation internes au système. Très utilisée par les praticiens de UML, cette technique est sous-tendue par des démarches de modélisation communément utilisées.
- Modèles conceptuels : UML permet de représenter les notions saillantes d'un domaine ou d'un système, et ainsi de formaliser les connaissances préliminaires du système, notamment par l'exploitation des diagrammes de classes.

L'ensemble des informations recueillies par le biais de ces techniques permet de constituer le référentiel du système. Le référentiel permet

Technique	Support UML	Support Outillé
Analyse des besoins	Néant	Outils dédiés indépendants ; Objecteering/UML supporte les exigences comme une extension UML
Dictionnaire	Néant	Support du dictionnaire comme une extension UML
Modélisation des processus métier (BPM)	Activity Diagrams L'OMG fournira des extensions standard pour ce besoin	Outils indépendants, ou outils UML (diagrammes d'activité avec des extensions dédiées (profils)(5))
Règles métier	Contraintes, Invariants, Pre/Post conditions	Outils dédiés, ou outils UML avec des extensions (profils) spécifiques
Approche systémique	Diagrammes de paquetages, "subsystem", Information flow (UML2.0), modèles d'assemblages (UML2.0) L'OMG fournira un standard dédié au " system engineering "	Outillages spécifiques pour la cartographie des SI, Outillages dédiés pour l'ingénierie des systèmes, Outillages UML. Ils peuvent être spécifiquement étendus (profils) pour la cartographie, ou pour le support du " system engineering " .
Cas d'utilisation	Cas d'utilisation	Outils UML. Objecteering/UML offre des extensions méthodologiques pour l'emploi des cas d'utilisation en phases amont.
Modèles conceptuels	Diagrammes de classe	Outils UML
Traçabilité	Notion de " dependency "	Techniques spécifiques à chaque outil. Quelques ateliers UML (dont Objecteering/UML) supportent la traçabilité.

(4) - OMG : Object Management Group, organisme de standardisation ayant notamment standardisé CORBA, et UML

(5) - Le " profil UML " est un mécanisme standard UML pour étendre UML pour un objectif spécifique (domaine d'application, technologie ciblée, méthodologie)

d'identifier les éléments présents dans le système d'information, ainsi que de fournir des nomenclatures et les identifiants qui seront constamment exploités et référés sur l'ensemble du cycle de vie des développements et évolutions du système.

Un effort particulier doit être entrepris pour obtenir de la part de l'ensemble des parties prenantes relatives au système et à ses évolutions, la validation et l'appropriation des modèles et informations recueillis. Des outils spécifiques sont nécessaires pour présenter le modèle aux divers niveaux des responsables hiérarchiques des métiers de l'entreprise, ainsi que pour le présenter aux utilisateurs finaux et leur permettre de comprendre l'organisation du processus, ainsi que le rôle que chacun des utilisateurs doit remplir

La gestion de la traçabilité est un élément important dans l'ensemble de l'approche. La mise en œuvre simultanée de techniques différentes impose en effet de gérer la cohérence globale, de garantir la complétude de chacune des facettes, et de permettre des vérifications croisées. De manière générale, le standard de modélisation UML apporte une assistance importante lors de la modélisation des phases amont. Cette assistance sera encore renforcée par les extensions du standard UML2.0. Cependant, UML ne couvre pas tous les besoins pour ces phases, où des extensions et des techniques complémentaires sont nécessaires. Le Tableau ci-dessous montre de manière synthétique le degré de support qu'UML fournit pour les techniques de modélisation des phases amont.

**Apport d'un outillage supportant la modélisation des phases amont**

L'ensemble des techniques pouvant être employées dans les phases amont, que celles ci relèvent de la description textuelle, ou de la modélisation, doit être coordonné à la fois par un processus de développement approprié, et par un outillage fédérateur. Un référentiel commun géré par l'outil est nécessaire. L'outil assure, par ailleurs, la coopération étroite entre les différentes formes de représentation, en maintenant la cohérence d'ensemble. Par exemple, la traçabilité doit être supportée et si possible automatisée par des assistants, afin de gérer les degrés de couverture de la modé-

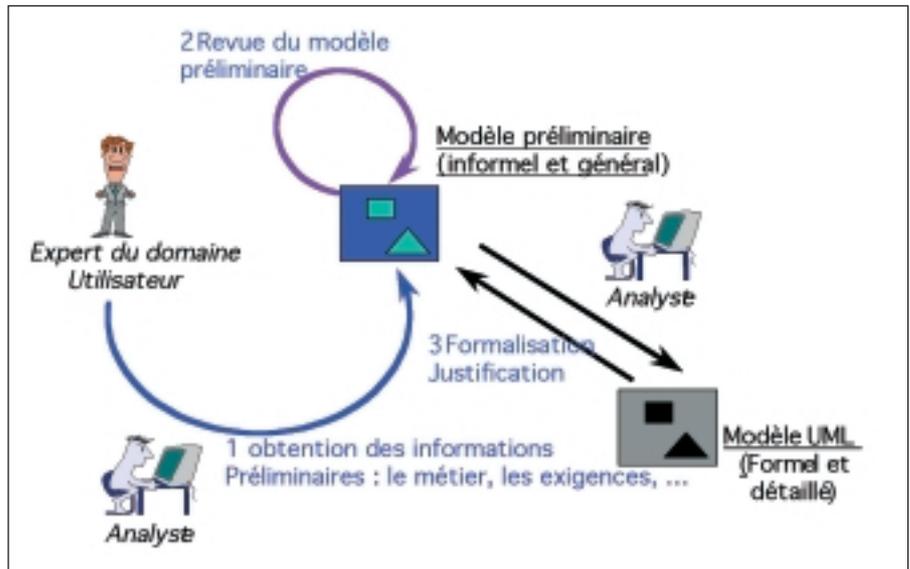


Figure 2 - Etapes de construction des modèles en analyse préliminaire

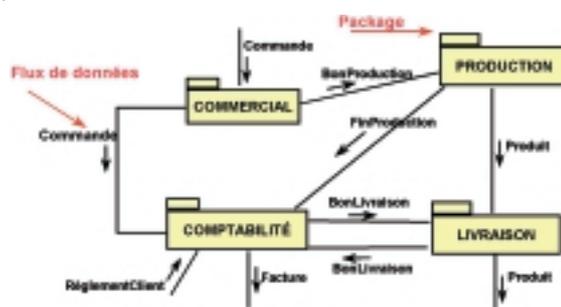


Figure 3 - Diagrammes de flux(Objecteering/UML) illustrant les échanges d'informations entre sous-systèmes

lisation, les correspondances entre différents niveaux de modélisation et les études d'impact des changements.

Un outillage approprié permet d'offrir des vues différentes d'un même modèle, apportant pour chaque type de technique une ergonomie dédiée, par exemple, des éditeurs graphiques UML, des tableurs pour saisir le dictionnaire ou les exigences. Les facultés d'extension et d'adaptation de UML (profils UML) permettent à l'utilisateur d'adapter UML à ses besoins et sa démarche, souvent conditionnée par l'entreprise et le secteur d'application.

Enfin, l'outillage doit apporter une productivité accrue des développeurs, ce qui, outre le gain d'efficacité assure une bonne acceptation par les développeurs de l'outillage et de la démarche sous-jacente. Les fonctions essentielles sont des fonctions de production de documentation automatique, permettant à partir des travaux sur des modèles ou des éléments textuels structurés de produire des documents, suivant un format propre à l'entre-

prise. L'outil assure une transition et une reprise d'information entre phases, apportant par là un fort gain de productivité et de qualité.

La production d'une documentation unifiée et cohérente à partir des modèles et des ajouts de commentaires en langage naturel permet également de relire et d'effectuer des revues globales, commentaires inclus, afin d'apporter une validation humaine toujours indispensable.

**Conclusion**

Le modèle correctement construit fournit aux producteurs du logiciel une spécification fonctionnelle sobre, stable et pertinente. En lui ajoutant la prise en compte des contraintes techniques propres à l'informatique de l'entreprise, ils pourront la prendre comme point de départ de leur travail. Par ailleurs, l'élucidation des processus du métier, et la mise à plat des informations exploitées qu'implique la modélisation apporte à l'entreprise la maîtrise de ses propres concepts, de son propre langage, ainsi que la clarté sur les priorités : la documentation du dictionnaire, du référentiel, des règles métier, leur appropriation par les utilisateurs, leur validation par les dirigeants, précisent et stabilisent l'informatisation de l'entreprise, améliorent la qualité des réflexions sur son rôle et son évolution. La modélisation est ainsi à la fois une étape utile à la réalisation technique du logiciel, et une phase nécessaire à la maturité de l'entreprise en matière de système d'information".

© Objecteering Software 2004

■ Philippe Desfray

# Ma rencontre avec .NET

TÉMOIGNAGE

En juin 2004, l'un de nos clients est venu nous trouver avec un sérieux problème. Cette société, leader sur son marché en France, installe et entretient des milliers de systèmes électromécaniques à travers tout le pays. Les appels de maintenance sont concentrés dans un centre qui organise les tournées des techniciens, envoie les pièces détachées et valide les interventions.

Cette société utilisait un logiciel pour la gestion de son centre d'appels, et un autre pour celle de son entrepôt (stock et envoi/réception d'articles) et sa comptabilité. Le problème venait du logiciel de gestion des appels, vieux de 5 ans, sur base Access, à l'ergonomie antédiluvienne et dont les arrêts non souhaités allaient de 3 par semaine à 3 par jour. D'autre part, on pouvait noter une instabilité structurelle de la passerelle entre les deux logiciels.

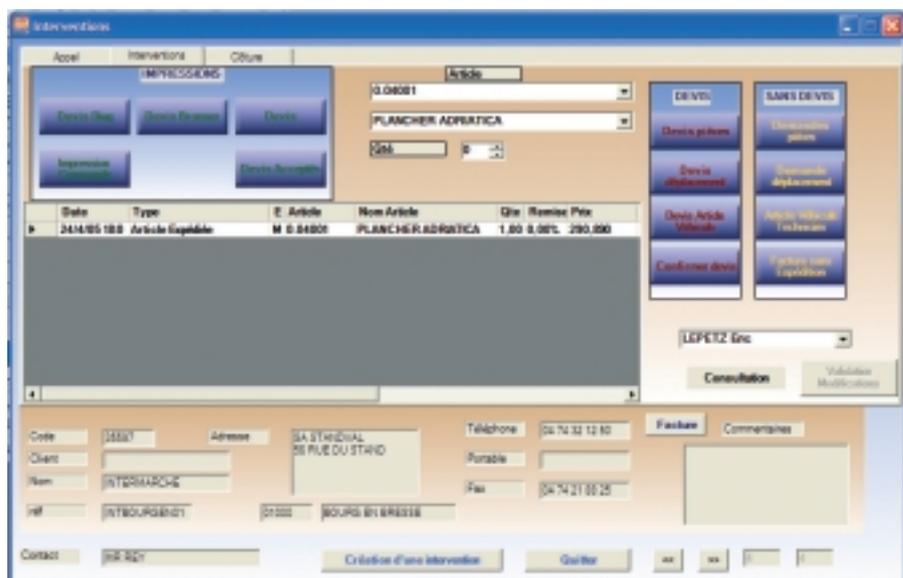
La période d'été arrivant, et avec elle la pointe annuelle du nombre d'appels, le risque était important d'assister à un blocage de longue durée, similaire à celui déjà enregistré pendant l'été 2003. Nous devions à la fois prévenir d'urgence le blocage et réfléchir à une solution plus pérenne.

## Migration VBA vers .NET

Jusque là, nos développements se faisaient en VBA client/serveur sur base SQL Server. Et tout naturellement c'est la direction que nous avons prise. En Juillet 2004 une maquette fût développée, avec deux objectifs :

- tester l'interface utilisateur et les fonctionnalités globales
- servir de " roue de secours " en cas d'arrêt brutal du système existant

Cette maquette réalisée, et la période cruciale passée, nous avons repris le projet en Septembre et nous sommes confrontés à un problème important : celui du déploiement. Impossible d'obtenir des performances acceptables sur les postes les plus anciens, et impossible, malgré deux mois de recherches de trouver le logiciel Microsoft permettant le déploiement (" run-time " Access). Que faire ? Ayant fait quelques tests préliminaires sur .NET pendant l'été, nous avons décidé de tenter la migration du prototype et le développement de l'application globale sur cette architecture.



Ce logiciel avait pour objectifs entre autres, la gestion du centre d'appels, l'organisation de la maintenance, du stock, des pièces détachées (voir encadré détaillé).

## Deux mois seulement de développement

Le planning de réalisation a été le suivant :

- Septembre et Octobre 2004 : développement et tests
- 2 Novembre 2004 : mise en production
- Mars 2005 : ajout de la fonctionnalité de gestion des Inventaires

Le système a identifié un bug en Février qui n'a entraîné ni perte de données, ni interruption, même partielle de l'utilisation.

Si ce planning peut être considéré comme serré, le succès de ce projet repose sur deux facteurs :

- l'utilisation de l'architecture Client/Serveur et de toutes les possibilités de SQL Server (TOUTES les règles de gestion et tous les processus métiers sont écrits en Transact\*SQL)

- l'architecture objet de .NET, permettant une réutilisation de code optimale, et une documentation générée (utilisation de NDoc) particulièrement claire

Les difficultés rencontrées sont venues :

- de la maîtrise de l'architecture de données à 3 niveaux
- de la pauvreté des Datagrid par rapport à leur homologue VBA

## L'accès aux données SQL

En premier lieu, il nous faut évoquer la différence entre VBA et .NET pour l'accès aux données SQL. Sous l'interface Access, on définissait un ordre SQL qui permettait la génération d'une grille d'accès aux données. Cette grille était paramétrable, avec des possibilités d'édition étendues comme l'ajout de liste déroulante, etc.

Rien de tel sous .NET. L'accès aux données nécessite trois objets indépendants que l'on doit ensuite lier :

- les ordres SQL

- un Dataset pour le stockage local des informations

- une Datagrid pour l'affichage

Les ordres SQL utilisent un éditeur identique à celui de VBA, mais contrairement à ce dernier, il ne sait pas générer automatiquement les UPDATE, INSERT ou autres DELETE si vous utilisez plusieurs tables. Il faut donc écrire un premier ordre SELECT ne concernant que les données modifiables pour générer les trois autres, puis le modifier afin d'obtenir les informations souhaitées.

Les ordres SQL de mise à jour génèrent un SELECT, afin de rafraîchir automatiquement les données. Oui, mais...je n'ai jamais réussi à les faire fonctionner avec des valeurs de Dataset calculées.

Le Dataset est une image locale de la structure de données et sans doute la structure nous ayant donné le plus de soucis. Il faut apprendre à le modifier, sous peine de perdre les liens avec les champs écran, et son maniement est assez délicat. La navigation dans un Dataset m'a rappelé le passage d'Ouessant par vent contre courant ! Quant au "currency manager", je ne souhaite même pas en parler.

Toute la navigation doit être programmée à la main, et même si des outils permettent de lier plusieurs tables dans un Dataset pour faciliter les déplacements, cela reste quand même un peu acrobatique. Je pense en fait que les problèmes viennent de l'universalité de l'architecture censée résoudre les développements web et client/serveur de façon identique.

Le Datagrid est d'une pauvreté épouvantable par rapport à son équivalent VBA, et il suffit pour s'en rendre compte de lire les messages le concernant sur les groupes de discussion. Il

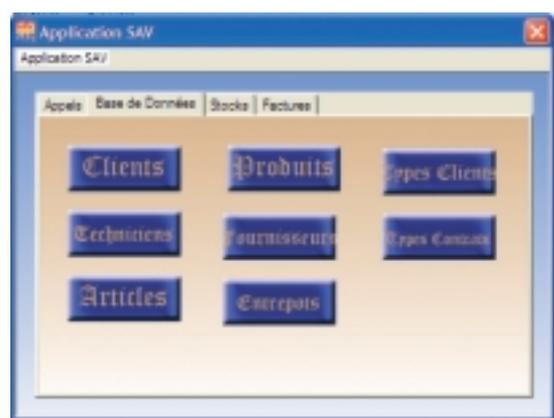
## FONCTIONNALITES de l'APPLICATION

- la gestion du centre d'appels
- la planification et le suivi des visites des techniciens sur les sites clients
- la gestion des stocks multi-entrepôts (entrepôt principal et véhicules techniciens)
- la gestion de la fabrication des machines
- la gestion de l'expédition des pièces détachées nécessaires aux réparations
- le réapprovisionnement des stocks et la gestion des fournisseurs
- la facturation des visites et des contrats de maintenance
- la gestion de règles complexes gérant le prix des pièces détachées en fonction de leur utilisation ou des clients destinataires
- Reprise et intégration de toutes les informations des deux logiciels remplacés (données ACCESS et ORACLE)
- Gérer l'accès par 30 à 40 utilisateurs en flux tendu, non interruptible.

existe, bien sûr, des développeurs ayant amélioré ses fonctionnalités, mais pour des raisons de Copyright et de maintenabilité, nous avons refusé d'utiliser des composants non Microsoft, libres de droits. Mais il paraît que la version 2005... C'est d'ailleurs ce que je dis à nos clients : vous verrez en V2 !

### La robustesse repose sur SQL Server

En fait, la robustesse de l'architecture et ses possibilités d'évolution reposent sur l'utilisation de SQL Server. Avec lui, pas de soucis de performances ou de stabilité, le nombre d'utilisateur ne dépendant que de la puissance du serveur...et de vos licences. En nombre de lignes de code, le développement TSQL doit représenter 40% du total et 75% des fonctionnalités de l'application. Les outils d'administration et de gestion



des données sont ergonomiques et performants, multi bases et serveurs.

Nous avons fait une utilisation systématique des Triggers et autres procédures stockées, et je pense que nous n'avons fait l'impasse sur aucune des fonctions du langage ou de l'architecture. Nous nous sommes tout de même arrêté avant l'utilisation de la réplication, mais un jour peut-être...

En conclusion, nous pensons avoir fait le bon choix, même si la lecture de MSDN à des heures avancées de la nuit ne vaut pas un bon Chandler. Le processus de déploiement, avec son intégration de CLUF (licence) et son ergonomie standard est tout à fait satisfaisant. Et le dernier point et non le moindre : en moins d'une semaine une version à interface en langue Anglaise a été développée et présentée aux USA. L'utilisation des règles Microsoft de localisation, tant au niveau du Client que du Serveur fonctionne exactement telle que décrite dans leur documentation.

■ Gilbert VIDAL

## PROBLÈME RENCONTRÉ

Problème soumis à Microsoft, dont la réponse ne fût pas vraiment satisfaisante.  
Dans un Dataset :

```
Me.bindingcontext(me.dataset.datatable).position = 0 toujours  
Me.bindingcontext(me.dataset, "datatable").position = position correcte
```

Nous avons demandé l'avis de Microsoft France et celui d'un expert Microsoft indépendant. En résumé, il semble que chacune des deux expressions est liée à son propre Currency Manager et donc à une position différente du Dataset.  
Malgré cela, ça ne marchait toujours pas, et je me dois de vous communiquer la dernière réponse reçue: "Puisque l'une des deux expressions fonctionne bien, pourquoi s'obstiner avec l'autre..." Cela pourrait servir de devise à notre métier.

# Commenter efficacement son code source en 5 étapes sous Visual Studio

Un développeur professionnel aime bien commenter son code, car il en est fier. Cependant, trop de commentaires risquent de nuire à la lisibilité de l'ensemble.

## 1 Apprenez à doser intelligemment vos commentaires

Comme le souligne très bien Kernighan et Plauger (\* 1), ce serait une erreur de documenter un mauvais code car dans ce cas, il vaut mieux le réécrire. Cela impliquerait-il qu'un code source convenablement rédigé n'a pas besoin de commentaires ? Cela dépend du nombre, de la nature et de la qualité des commentaires. Par exemple, les commentaires de répétitions du code sont préjudiciables. Un programmeur habitué à coder en C# trouvera inutile que l'on commente ce bout de code :

```
//Crée une nouvelle instance s'il n'en existe pas une autre
if(_InstanceUnique == null)
    return _InstanceUnique = new Singleton();
else
    return _InstanceUnique
```

En effet, le commentaire répète ce que le code explique très bien. Remarquez, qu'un commentaire littéral répétitif par rapport au code source est en plus astreignant à mettre à jour (car vous devrez mettre à jour à la fois votre code et votre commentaire). Si le développeur estime qu'il doit commenter largement pour expliquer son code, c'est que probablement celui-ci est tortueux et doit être réécrit. Sous Visual C# 2005, pour commenter un bloc, vous devez le sélectionner, puis taper CTRL + K, CTRL + C. Et pour le décommenter, il vous suffit d'appliquer les raccourcis CTRL + K, CTRL + U.

On peut toutefois considérer que le commentaire précédent est utile dans le sens où il résume parfaitement quelques lignes de code. Les commentaires de résumé sont productifs, à condition de pouvoir être parcourus plus rapidement que le code lui-même. D'après Steve McConnell (\* 2), des études menées par IBM ont prouvé qu'une ligne de commentaire pour 10 lignes de code était une proportion optimale pour augmenter la productivité. Avec moins, ou plus de lignes de code, la compréhension diminue.

### NOTES

\* Note 1 : Kernighan, B.W., Plauger, P.J., "Programming style for programmers and language designers", IEEE Symposium on Computer Software Reliability, New York.

\* Note 2 : Steve McConnell "Tout sur le code", Microsoft Press, ISBN 210-048753-1.

## 2 Utilisez à bon escient le commentaire aide-mémoire TODO

Souvent, vous êtes amenés à réaliser un commentaire temporaire. Du genre, "cette note pour signaler que le travail sur cette proportion de code n'est pas terminé". Si vous livrez votre code source avec votre logiciel, ou bien qu'un développeur externe soit obligé de lire votre

Système d'exploitation : Windows XP

Environnement : Visual Studio .NET 2003 et Visual Studio 2005

Langages : C#

code pour maintenance, il serait très malvenu de laisser ce genre de commentaire à la vue de tous (du genre "Attention : à corriger sinon plante une fois sur quatre..."). Sous Visual C# 2005, il existe une manière assez élégante de résoudre ce dilemme : vous marquez votre ligne de commentaire d'un "TODO" comme ceci :

```
{
    //TODO Attention : les deux lignes seront à réécrire
    NbrOctetsLus = GZipDeflateStream.Read(arrTamponFichier
Source, 0, increment);
    Console.WriteLine("de type 'Deflate'. ");
}
//TODO Attention : le catch est aussi à réécrire
catch (InvalidDataException)
```

Maintenant, vous pouvez naviguer dans les tâches View / Next Task ou View / Previous Task. Vous pouvez aussi faire apparaître la liste des tâches (CTRL + W, CTRL + T) et demander d'afficher la liste des TODO (Affichage / Afficher les tâches / Commentaires). En cliquant sur un des TODO, vous vous retrouvez sur la ligne en question. Simple, mais puissant. Vous pouvez considérer le commentaire TODO comme étant un aide mémoire pour ne pas oublier de contrôler une procédure, voire même de la terminer. Maintenant, si le commentaire TODO ne vous convient pas, vous pouvez aller plus loin dans la gestion de vos commentaires en les personnalisant. Ceci est facile à réaliser sous Visual C# 2005 via le menu : Outils / Options – Choisissez l'option Environnement / Liste des tâches (par exemple, vous pouvez modifier la balise TODO en une balise personnalisée URGENT avec une priorité différente).

## 3 Stockez vos commentaires comme des snippets

Si on part du principe qu'un commentaire n'est vraiment utile qu'à la condition qu'il ne soit pas déjà inclus au code, vous pouvez, par exemple, y inclure le type de licence. Prenons le cas de la découpe d'une classe principale en x fichiers (en utilisant les classes de type Partial), dans le but de "splitter" une même classe en plusieurs codes sources ".cs". C'est le compilateur qui se chargera de rapprocher l'ensemble au sein d'une même classe. Mais vous serez sans doute amenés à répéter certains commentaires comme le nom de l'auteur, le type de licence, etc.

```
/** Classe partielle : GigaClasse
/** Copyright 2005, Programmez
/** Auteur : moi
```

N'essayez pas de faire du style (par exemple, en créant un encadré de \*, ou en utilisant des "..." pour tabuler). Un commentaire avec un style simple sera en effet beaucoup plus facile à maintenir. Créez dans votre boîte à outils une nouvelle séparation (add tab), baptisez celle-ci, par exemple, "Commentaires", sélectionnez le bloc de commentaires et faites le glisser vers la boîte à outils pour le stoker. A propos du copier-coller, saviez-vous que vous pouvez effectuer une sélection partielle multi-lignes verticale ? Pour réaliser ce type de manipulation, il suffit d'effectuer une première sélection partielle sans relâcher le bouton droit de souris, puis d'appuyer sur la touche "ALT", tout en déplaçant verticalement le pointeur.

```

1) using System.Reflection;
2) using System.Reflection.Emit;
3) using System.Reflection.IntrospectionExtensions;
4
5 // General information about an assembly is controlled through the following
6 // set of attributes. Change these attribute values to modify the information
7 // associated with an assembly.
8 [assembly: AssemblyTitle("ExampleAssembly")]
9 [assembly: AssemblyDescription("")]
10 [assembly: AssemblyConfiguration("")]
11 [assembly: AssemblyCompany("")]
12 [assembly: AssemblyProduct("ExampleAssembly")]
13 [assembly: AssemblyCopyright("Copyright © 2005")]
14 [assembly: AssemblyTrademark("")]
15 [assembly: AssemblyCulture("")]

```

La sélection partielle multi-lignes verticale est une fonctionnalité déconcertante.

Revenons à nos moutons : vous pouvez maintenant faire glisser le commentaire stocké (ou le bout de code appelé snippet en anglais) présent dans la boîte à outils, pour le répéter en début de déclaration de vos classes partielles.

## 4 Apprenez à autodocumenter votre code avec les balises XML

Un programmeur professionnel doit être en mesure de produire une documentation technique professionnelle, et heureusement, Visual C# 2005 met à la disposition du développeur des tags spécifiques qui permettent de générer automatiquement une documentation technique. Visual C# 2005 créera un document XML de documentation qui sera ensuite traité par un outil externe pour produire un document dans un format lisible. NDoc créera une documentation compatible avec le format MSDN (Ndoc est un logiciel sous licence GPL).

C# supporte une série de tags XML préfixés par "///" qui devront être saisis avant la déclaration des entités de type namespace, classe ou autre (class, struct, interface, delegate, enum, constructor, property, method, event). Tous ces tags seront stockés dans un document XML qui sera généré par le compilateur (mais il faut le lui demander explicitement). En appliquant ensuite une feuille de style XSLT, il est possible de transformer ce fichier XML en fichier HTML. Il est concevable d'employer ses propres tags, mais Microsoft recommande l'utilisation de toute une liste de tags particuliers.

Pour bien comprendre, examinons quelques tags. Par exemple, le tag <c> permet d'afficher le texte qu'il contient sous la forme de

Syntaxe :

```
<c>texte</c>
```

Exemple :

```
/// résumé de <c>MaClasse</c>
```

Le tag <code> permet d'afficher le texte qu'il contient sous la forme de

code, mais sous la forme de plusieurs lignes (contrairement au tag <c>).

Syntaxe :

```
<code>
ligne de code 1
ligne de code 2
</code>
```

Exemple :

```
/// <code>
Stream fsSource = null;
Stream fsDest = null;
Stream GZipDeflateStream = null;
/// </code>
```

Le tag <example> permet de fournir un exemple d'utilisation.

```
/// <example>
/// Exemple d'appel :
/// <code>
/// MBigClasse m = new BigClasse();
/// m.AppelMethode(1,2,"Ludo");
/// </code>
/// </example>
```

Le tag <exception> fournit des informations sur la levée d'une exception par une méthode, tandis que le tag <list> crée une liste d'éléments avec puce, ou numérotés, ou sous forme de tableau.

Syntaxe :

```
<list type="bullet" | "number" | "table">
<listheader>
<term>term</term>
<description>description</description>
</listheader>
<item>
<term>term</term>
<description>description</description>
</item>
</list>
```

Le tag <param> donne des indications sur un paramètre d'une méthode.

```
/// <param name="secret">Code secret tapé par l'utilisateur</param>
```

Le tag <remarks> renseigne une information complémentaire.

```
/// <remarks>Le deuxième paramètre est optionnel.</remarks>
```

Le tag <return> commente la valeur de retour d'une méthode

```
/// <returns>Retourne une chaîne cryptée </returns>
```

Le tag <see> pointe vers un élément du code et si le compilateur ne le trouve pas, il émettra un avertissement.

```
/// <see cref="MaClasse2" />
```

Le tag <seealso> crée un lien vers un élément qui sera inclus dans la section see also.

```
/// <seealso cref="MaClasse3"/>
```

Le tag <value> indique des informations sur une propriété.

```
/// <value> valeur login utilisé pour authentification
/// </value>
```

Le tag <summary>, très commun, fournit un résumé

```
/// <summary>
/// Crée une nouvelle instance s'il n'en existe pas une autre
/// </summary>
```

Visual C# 2005 interagit avec certains de ces tags. Par exemple, nous avons vu que si un élément pointé par le tag see est introuvable le compilateur s'en plaindra. De même, Intellisense puisera dans le tag de <summary> pour afficher des informations sur l'entité. Concrètement, vous n'avez plus qu'à saisir les /// avant la déclaration d'une entité pour que Visual C# 2005 génère automatiquement un tag XML adapté.

```
using System;
using System.IO;

namespace Exemple_FrameWork2
{
    class ExempleReadAll
    {
        static void Main(string[] args)
        {
            string strContenuFichier;
            strContenuFichier = File.ReadAllText(@"C:\news.txt");
            Console.WriteLine(strContenuFichier);
            Console.ReadLine();
        }
    }
}
```

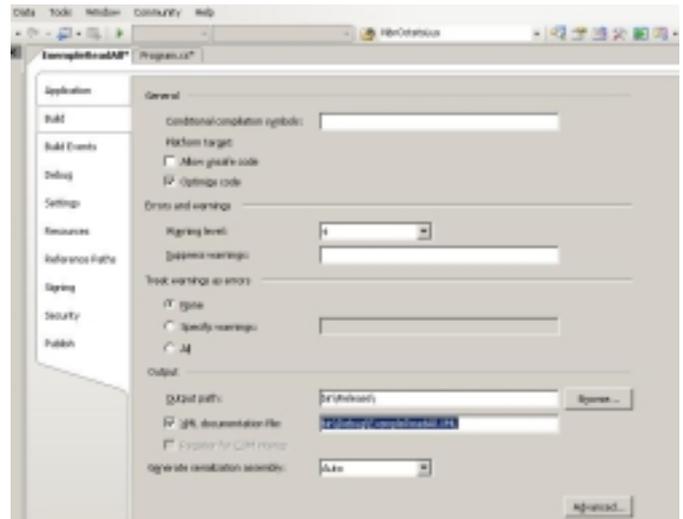
Si vous saisissez trois "/" avant la déclaration de la classe ExempleReadAll, Visual C# 2005 génère automatiquement le bloc :

```
/// <summary>
/// Mon commentaire tagé XML
/// Résumé : le fichier news.txt est lu d'une traite avec ReadAllText puis affiché
/// </summary>
class ExempleReadAll
```

Pour générer le XML, il faudra néanmoins l'indiquer explicitement en configurant votre projet Projet -> "Nom du projet" / Propriétés / build (générer) et modifier la propriété "Fichier de documentation XML" ("XML DOCUMENTATION FILE") en indiquant nom\_du\_projet.xml (c'est une convention).

Ou en ligne de commande csc GigaUtilitaire.cs /doc:MaDocGigaUtilitaire.xml .Voici ce que cela donne :

```
<?xml version="1.0"?>
<doc>
  <assembly>
    <name>ExempleReadAll</name>
  </assembly>
```



```
<members>
  <member name="T:Exemple_FrameWork2.ExempleReadAll">
    <summary>
      Mon commentaire tagé XML
      Résumé : le fichier news.txt est lu d'une traite avec ReadAllText
      puis affiché
    </summary>
  </member>
</members>
</doc>
```

Si quelque part dans votre code source vous tapez le nom de la classe pour laquelle il existe un résumé, un pop-up affichera ce résumé. Pour générer la documentation sous la forme d'une page HTML, vous devez utiliser l'option "Générer les pages web de documentation ..." du menu "Outils" (indisponible avec la version Express).

Visual Studio pourra souvent réaliser seul tout le travail, ou presque. Par exemple, voici ce que l'IDE génère automatiquement après avoir tapé seulement "///" avant la déclaration de public recherche() :

```
/// <summary>
///
/// </summary>
/// <param name="Contenu"></param>
/// <param name="indice"></param>
/// <param name="ChaineRecherchee"></param>
public recherche(string Contenu, int indice, int ChaineRecherchee)
{
    ...
    return ds;
}
```

## 5 Utilisez GhostDoc et Ndoc pour finaliser votre documentation professionnelle

Ghostdoc est un addin étonnant pour VS Studio 2003 qui tire ses informations du nom de vos entités. La documentation est là, camouflée pour ainsi dire dans le code, et Ghostdoc se charge de l'extraire et de la mettre en valeur. Une fois le plug-in installé, il suffit de faire appel au menu contextuel, tout en pointant au dessus de la méthode que vous

désirez autodocumenter (->"Document this"). GhostDoc analyse le texte, en repérant la différence entre les minuscules et majuscules, ainsi que certaines abréviations (anglaises), pour en extraire un commentaire cohérent. Voici un exemple parlant qui vaut mieux qu'un grand discours :

```
public bool CanRestart()
{
    return false;
}
```

Deviens :

```
/// <summary>
/// Determines whether this instance can restart.
/// </summary>
/// <returns>
/// <c>true</c> if this instance can restart; otherwise, <c>>false</c>.
/// </returns>
public bool CanRestart()
{
    return false;
}
```

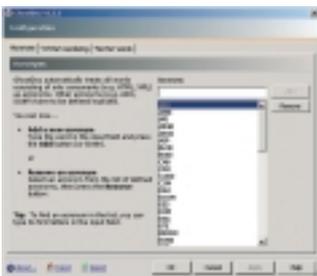
Ou encore, `public static bool UpdateSystemConfiguration(string keyWord, string configValue) {}`, se transforme en :

```
/// <summary>
/// Updates the system configuration.
/// </summary>
/// <param name="keyWord">Key word.</param>
/// <param name="configValue">Config value.</param>
/// <returns></returns>
public static bool UpdateSystemConfiguration(string keyWord, string configValue) {}
```

Nous avons laissé ces exemples en anglais, car Ghostdoc n'analyse pas aussi bien la langue de Molière... Pour preuve, voici ce que l'on obtient avec `public static bool MiseAJourDeLaConfigurationSystème(string MotClé, string ValeurDeConfiguration) {}`

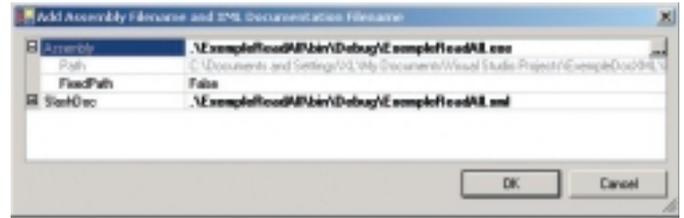
```
/// <summary>
/// Mises the A jour de la configuration système.
/// </summary>
/// <param name="MotClé">Mot CLÉ.</param>
/// <param name="ValeurDeConfiguration">Valeur de configuration.</param>
/// <returns></returns>
```

Il est néanmoins possible de configurer GhostDoc, pour lui demander de ne pas introduire "the" avant tel ou tel mot de la langue française, mais cela devient un peu lourd.



Ce genre d'outil doit vous aider à devenir plus productif et par conséquent, être fonctionnel au premier démarrage.

Un autre outil indispensable (un "must have") est Ndoc. Ndoc emploie différentes feuilles de style XSLT pour générer un document au format Html ou Javadoc



ou MSDN (ou encore Latex, mais ce format est encore en bêta avec la version 1.3.1 testée). Notez que Ndoc supporte Mono, ainsi que le framework 1.0 et 1.1, mais non le framework 2.x. Soit vous importez directement une solution de Visual Studio .Net 2003 en utilisant l'option "New from Visual Studio Solution", soit vous ajoutez vos assemblages en cliquant sur le bouton "add".



Avant de générer la documentation (Ctrl+Maj+B), vous devez choisir son type (MSDN par exemple). Un certain nombre de paramètres peuvent être ensuite renseignés pour paramétrer la génération, par exemple, la langue du document, ainsi que les informations de version de l'assemblage. Ndoc est intuitif et ne demande aucune connaissance particulière pour construire votre documentation technique, la condition minimale étant d'avoir convenablement balisé votre code sources de tags XML.

## Dix règles d'or

Pour conclure en beauté, citons Steve McConnell, encore lui, qui nous recommande les règles d'or suivantes, qui tiennent plus du bon sens que d'autre chose :

- 1 Utilisez les commentaires pour préparer le lecteur à ce qui va suivre ;
- 2 Pas de commentaires inutiles ;
- 3 Documentez ce qui ne paraît pas évident (ce qu'il appelle les "surprises") ;
- 4 Ne mettez pas d'abréviations ;
- 5 Distinguez les commentaires principaux des mineurs ;
- 6 Commentez tout ce qui est non documenté par le langage ou l'environnement ;
- 7 Commentez les plages numériques autorisées et les limites des données entrées ;
- 8 Décrivez chaque classe ou sous-programme en une phrase ou deux en début de code ;
- 9 Documentez les données globales ;
- 10 Et enfin, le conseil le plus judicieux : ne commentez pas un code rusé, réécrivez-le !

### Liste des liens :

Ndoc : <http://ndoc.sourceforge.net>

Ghostdoc : <http://www.roland-weigelt.de/ghostdoc/>

■ Xavier Leclercq - [Xavier.Leclercq@programmez.com](mailto:Xavier.Leclercq@programmez.com)

# Programmez OpenGL avec Java

*Sun Microsystems n'en finit pas de soutenir des projets autour de son langage. Nous vous présentons aujourd'hui JOGL qui permet de programmer OpenGL très simplement avec Java.*

La programmation 3D fait partie des thèmes incontournables de l'informatique moderne. Sun Microsystems a bien compris cela et a doté son langage d'une API Java3D que j'ai eu le privilège de vous présenter dans *Programmez!* N° 60 et 61. Cette API, dont les sources sont désormais disponibles (<http://java.sun.com/products/java-media/3D/>), est de haut niveau et totalement objet, ce qui présente des avantages et des inconvénients. Un des inconvénients est que le programmeur C/OpenGL peut éprouver des difficultés à migrer à la fois vers Java et vers cette API, le code OpenGL C étant, à priori, d'une organisation très différente de celle requise à une application Java3D. Le Game Technology Group de Sun Microsystems a lancé un autre projet Open Source baptisé JOGL et qui est cette fois une simple enveloppe autour des APIs d'OpenGL. JOGL est disponible pour Solaris, Linux, Windows et MacOS X. La librairie JOGL a un intérêt double. D'abord la similitude du code C et du code Java fait qu'il est très simple de migrer son code le cas échéant et aussi qu'il est très simple, pour la même raison, d'apprendre OpenGL avec Java en se référant à un ouvrage dont les exemples seront classiquement écrits en C. Ensuite JOGL tire au mieux partie des capacités d'accélération graphiques de la plate-forme hôte (l'API Java3D utilise elle aussi l'accélération graphique) et des pilotes. Pour prendre un exemple, puisque JOGL est une simple enveloppe autour d'OpenGL, on construira des listes d'affichage aussi facilement qu'en C. Dans la suite de cet article, qui n'est pas un cours sur OpenGL, nous allons voir comment installer JOGL, comprendre l'organisation de la librairie et voir les similitudes entre un code Glut/OpenGL et son équivalent JOGL.

## 1 Installation et mise en œuvre de JOGL

JOGL est librement téléchargeable à <https://jogl.dev.java.net> et vient sous de multiples archives. Voyons qui sert à quoi :

- **jogl.jar** : contient toutes les classes de la librairie. Autrement dit, le runtime. Toute application JOGL doit avoir un CLASSPATH qui pointe sur cette archive.
- **jogl-natives-xxx.jar** : où xxx peut être Linux ou win32, par exemple. Il s'agit de librairies natives partagées (.dll, .so) interfaçant le runtime ci-dessus avec les librairies 3D (ou pilotes) du système hôte. Chaque plate-forme recevant une application JOGL devra recevoir le jeu de librairies partagées qui lui correspond et les applications devront pouvoir y accéder. Nous traiterons ce point plus loin dans cet article.
- **jogl-src.zip** : contient les sources des deux archives ci-dessus. Le téléchargement n'est pas indispensable, mais consulter de tels sources est extrêmement instructif quant à la manière de créer une enveloppe Java autour d'une librairie C de bas niveau comme OpenGL.
- **jogl-demos.jar** : un riche jeu de démonstrations compilées et prêtes à l'emploi (Fig. 1).
- **jogl-demos-data.jar** : des données (textures, etc) pour les démos.

Pour profiter pleinement de la lecture de cet article vous devez posséder :

- des notions de programmation OpenGL et GLUT en C/C++,
- Une JVM 1.4.2 minimum.

- **jogl-demos-utils.jar** : des classes utilitaires pour les démos. Cette archive et la précédente doivent être pointées par le CLASSPATH pour lancer ou compiler une démo.
- **jogl-demo-src.zip**. Les sources des démos.
- **javadoc\_public.zip**. La Javadoc de JOGL. Incontournable.

Tout venant sous forme d'archives, l'utilisation est simple. On voit souvent la recommandation de déposer de telles archives dans le répertoire lib/ext de la JRE. C'est certes la voie la plus directe, mais qui ne fonctionnera plus dès que l'on changera de JVM. Java étant un système sans cesse enrichi de librairie tierces, le mieux est sans doute de consacrer un répertoire à ces librairies sur votre système. Par exemple /opt/javalibs sous Linux ou c:\javalibs sous Windows. Dans ce répertoire, on placera les répertoires de chaque librairie et encore dans un sous-répertoire libs, les librairies partagées éventuelles. On configure alors l'outil de développement pour qu'il accède à tout cela, et on écrit un petit script Shell ou batch pour lancer les applications de manière autonome. Prenons pour l'exemple InfiniteShadowVolume.java parmi les sources de démos, et modifions en le package. Nous supposons vouloir compiler et lancer l'application sous Eclipse, les principes demeurant exactement les mêmes sous JBuilder ou avec un autre outil de développement. On ajoute au projet les librairies requises, c'est à dire les jars (Fig. 2). Ceci suffit pour la compilation, mais au lancement, la JVM devra trouver les librairies natives (.dll ou .so). Faute de les trouver, la JVM lèverait cette exception :

```
Exception in thread "main" java.lang.UnsatisfiedLinkError: no jogl in java.library.path
    at java.lang.ClassLoader.loadLibrary(Unknown Source)
    at java.lang.Runtime.loadLibrary0(Unknown Source)
    at java.lang.System.loadLibrary(Unknown Source)
```

Il sera possible de placer ces librairies dans le PATH système (Windows) ou dans le cache de ldconfig (cf. ld.so.conf sous Linux), mais le mieux est d'informer directement la JVM de leur emplacement, en lui définissant la propriété java.library.path. (Fig. 3) Sous Linux, on peut encore ajuster la variable d'environnement LD\_LIBRARY\_PATH à l'intention de l'application. Moyennant quoi, vous pouvez admirer la démo (Fig. 4). Voici un exemple de script Shell Linux pour lancer une application JOGL autonome :

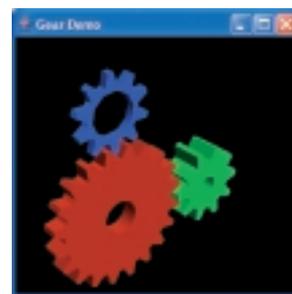


Fig. 1 : OpenGL à toute vitesse avec Java.

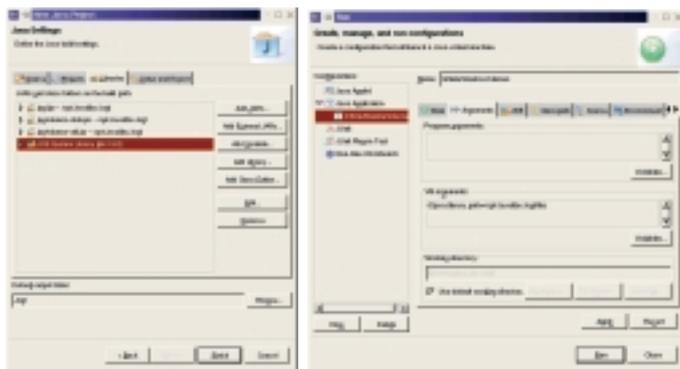


Fig. 2 : Votre outil de développement Fig. 3 : La JVM doit savoir où localiser les jars de JOGL. bibliothèques natives de JOGL.



Fig. 4 : Une des démonstrations de JOGL, Fig. 5 : Notre démonstration en JOGL ici sous Linux.



```
#!/bin/bash

JOGL=/opt/Javalibs/Jogl
CLASSPATH=$JOGL/jogl.jar
CLASSPATH=$JOGL/jogl-demos-data.jar:$CLASSPATH
CLASSPATH=$JOGL/jogl-demos-util.jar:$CLASSPATH

LD_LIBRARY_PATH=$JOGL/libs:$LD_LIBRARY_PATH

java -cp .:$CLASSPATH programmez.fred.jogl.InfiniteShadowVolumes
```

Ou bien un fichier batch pour Windows :

```
@echo off

set JOGL=c:\Javalibs\Jogl
set CLASSPATH=%JOGL%\jogl.jar
set CLASSPATH=%JOGL%\jogl-demos-data.jar;%CLASSPATH%
set CLASSPATH=%JOGL%\jogl-demos-util.jar;%CLASSPATH%

set LIBSPATH=%JOGL%\libs

REM ci-dessous, taper la commande sur une seule ligne
java -Djava.library.path=%LIBSPATH% -cp .;%CLASSPATH% programmez.fred.jogl.InfiniteShadowVolumes
```

## 2 Ecrire une application JOGL.

Partons d'un exemple GLUT écrit en C++ (du C déguisé en fait), dont vous trouverez le code sur le site, en compagnie des autres exemples. Nous faisons tourner un triangle multicolore autour de son axe vertical et nous le faisons aller et venir de l'arrière-plan vers l'avant-plan. Enfin, pour meubler la scène, un tore filaire tourne autour de deux de ses axes et autour du triangle (Fig. 5). L'organisation d'une application GLUT est toute simple. A l'initialisation, on définit des fonctions de rappel, principalement une pour l'affichage et une pour gérer le redimensionnement éventuel de la fenêtre, qui est gérée par GLUT. Ces deux fonctions de rappel usent et abusent alors des API OpenGL. Globalement, sous JOGL le principe est le même, sauf que la classe Java GLUT ne gère pas la fenêtre, mais seulement les API GLUT de haut niveau. C'est AWT qui gère la fenêtre, ce qui est cohérent dans l'univers Java. Ou encore Swing (voir exemple sur le site) mais AWT est le plus intéressant. Par ailleurs, sous Java, les fonctions de rappel n'existent pas. On emploie alors la même démarche que pour les événe-

ments. On implémente une interface contractuelle `GLEventListener`, dite écouteur d'événements et dont la JVM invoquera les méthodes qui tiendront lieu de fonction de rappel. C'est le point important, comme le montre l'exemple en fin d'article. L'usage est d'implémenter l'interface sous la forme d'une classe interne. Ceci posé, comment dessine-t-on ? La classe JOGL ne s'instancie normalement pas directement. On obtient une instance par l'intermédiaire d'une fabrique d'objet. Regardez le code du constructeur de notre exemple. On commence par obtenir cette fabrique qui nous construit un canevas, une sorte de super panneau, à partir des capacités matérielles du système hôte. A ce canevas on attache l'écouteur d'événement et finalement, on attache notre canevas à un objet `Animator` qui porte bien son nom. Dès que la méthode `start` de l'`Animator` est invoquée, la méthode de rappel `display` de l'écouteur sera invoquée régulièrement. Cette méthode reçoit notre canevas en argument sous le type `GLDrawable`. C'est ici que l'on peut enfin accéder au pipe-line d'affichage d'OpenGL, c'est à dire faire une programmation qui ressemble à s'y méprendre à du C. Une dernière remarque avant que nous ne nous quittons. La méthode `reshape` de l'écouteur est toujours invoquée au moins une fois par le système, au moment où la fenêtre apparaît sur votre écran.

### DemoTriangle

```
package programmez.fred.jogl;

import net.java.games.jogl.GLCanvas;
import net.java.games.jogl.GLCapabilities;
import net.java.games.jogl.GLDrawableFactory;
import net.java.games.jogl.Animator;
import net.java.games.jogl.GL;
import net.java.games.jogl.GLEventListener;
import net.java.games.jogl.GLDrawable;
import net.java.games.jogl.GLU;
import net.java.games.jogl.util.GLUT;

import java.awt.Frame;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;

public class DemoTriangle {
```

```

private final int Width = 300;
private final int Height = 300;
private float angle = 0;
private float eloignement = -20.0f;
private float pas = 0.2f;
private boolean sens = true;
private GLUT glut;

class DemoTriangleListener
    implements GLEventListener {

    public void init(GLDrawable drawable) {
        GL gl = drawable.getGL();
        gl.glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
        gl.glShadeModel(GL.GL_SMOOTH);
    }

    public void display(GLDrawable drawable) {
        GL gl = drawable.getGL();
        gl.glClear(GL.GL_COLOR_BUFFER_BIT);
        gl.glLoadIdentity();
        gl.glTranslatef(0.0f, 0.0f, eloignement);
        // ou
        /*
        GLU glu = drawable.getGLU();
        glu.gluLookAt( 0.0f, 0.0f, -eloignement, 0.0f,
                    0.0f, 0.0f, 0.0f, 1.0f, 0.0f);
        */
        gl.glRotatef(angle, 0.0f, 1.0f, 0.0f);
        gl.glBegin(GL.GL_TRIANGLES);
        gl.glColor3f(1.0f, 0.0f, 1.0f);
        gl.glVertex3f(0.0f, 0.5f, 0.0f);
        gl.glColor3f(1.0f, 1.0f, 0.0f);
        gl.glVertex3f(-0.75f, -0.5f, 0.0f);
        gl.glColor3f(0.0f, 1.0f, 1.0f);
        gl.glVertex3f(0.75f, -0.5f, 0.0f);
        gl.glEnd();

        gl.glRotatef(angle, 1.0f, 0.0f, 1.0f);
        glut.glutWireTorus(gl, 0.4, 1.5, 10, 30);

        angle += 1.0f;
        if(angle > 360.0f)
            angle = 0.0f;

        if(sens) {
            eloignement -= pas;
            if(eloignement < -49.0f)
                sens = false;
        }
        else {
            eloignement += pas;
            if(eloignement > -20.0f)
                sens = true;
        }
    }
}

```

```

}

public void reshape(GLDrawable drawable,
                    int x,
                    int y,
                    int width,
                    int height) {
    GL gl = drawable.getGL();
    gl.glViewport(0, 0, width, height);
    gl.glMatrixMode(GL.GL_PROJECTION);
    gl.glLoadIdentity();
    gl.glFrustum(-1.0, 1.0, -1.0, 1.0, 10.0, 50.0);
    gl.glMatrixMode(GL.GL_MODELVIEW);
}

public void displayChanged(GLDrawable drawable,
                           boolean modeChanged,
                           boolean deviceChanged) {
}

}

DemoTriangle() {
    glut = new GLUT();
    GLCapabilities capabilities = new GLCapabilities();
    GLDrawableFactory factory = GLDrawableFactory.getFactory();
    GLCanvas canvas = factory.createGLCanvas(capabilities);
    canvas.addGLEventListener(new DemoTriangleListener());
    final Animator anim = new Animator(canvas);
    Frame f = new Frame("Programmez! JOGL");
    f.setSize(Width, Height);
    f.add(canvas);
    f.setVisible(true);
    f.addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent e) {
            anim.stop();
            System.exit(0);
        }
    });

    f.addKeyListener(new KeyAdapter() {
        public void keyPressed(KeyEvent e) {
            if(e.getKeyCode() == KeyEvent.VK_ESCAPE) {
                anim.stop();
                System.exit(0);
            }
        }
    });

    anim.start();
}

public static void main(String[] args) {
    new DemoTriangle();
}
}

```

■ Frédéric Mazué - [fmazue@programmez.com](mailto:fmazue@programmez.com)

# Embarquez Python dans vos applications C++

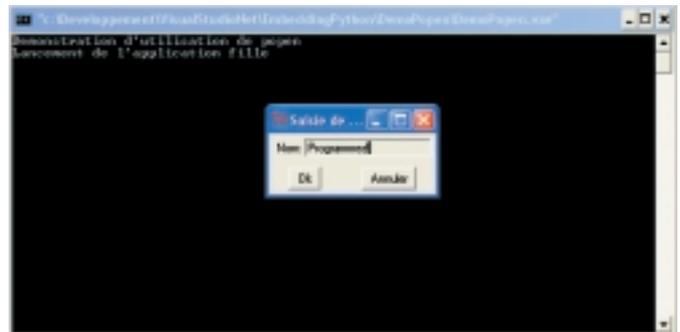
SOURCES DE L'ARTICLE  
WWW.PROGRAMMEZ.COM

*Si vous êtes paresseux comme une couleuvre, si vous devez livrer rapidement une application C++ extensible ou offrant une interface programmable, Python est exactement ce qu'il vous faut.*

Travailler avec C++ présente des avantages. On peut tout faire avec lui, et la rapidité à l'exécution est au rendez-vous. Cela présente aussi des inconvénients. C'est un langage difficile et les temps de développement sont souvent importants, le pire du cycle écriture/compilation étant atteint quand il s'agit de produire des interfaces utilisateurs graphiques et cela, malgré la qualité de toolkits tels que wxWidgets (cf. Programmez! 75). C++ présente aussi des lacunes. Par exemple, il n'offre rien si votre application doit présenter une interface programmable à l'utilisateur, à la manière d'un Autocad ou d'un Emacs. Si dans l'absolu on peut tout faire avec C++, le problème est qu'il faut souvent tout coder. Python, à l'opposé, sans compilation et avec une expressivité maximale, c'est le développement (très) rapide assuré, même pour les interfaces graphiques. Python est extensible à l'infini, et générer et exécuter du code Python avec Python n'est rien d'autre qu'un jeu d'enfant. Alors, faire collaborer C++ et Python est toujours une bonne idée. On réservera C++ au moteur d'une application et on prendra Python pour le reste, profitant ainsi du meilleur des deux mondes. C++ et Python peuvent collaborer de trois façons. Python peut invoquer du code écrit en C++, C++ et Python peuvent travailler en parallèle, ou C++ peut embarquer Python. J'ai déjà eu le privilège de vous parler de la première possibilité dans les pages de Programmez! et je vous propose de découvrir aujourd'hui les deux autres possibilités, que j'utilise personnellement régulièrement avec profit.

## 1 C++ et Python travaillant de concert

Nous supposons avoir une application C++, présentant éventuellement une fenêtre principale codée en C++. Nous ne voulons pas perdre de temps à coder des boîtes de dialogues, pas plus que nous voulons travailler avec C++ sur des fichiers de configuration enregistrant les résultats de ces dialogues. Nous déléguons cela à Python. L'idée est, d'un côté, d'écrire de petits scripts autonomes, Python écrivant les résultats des dialogues sur la sortie standard, et de l'autre côté, de lancer les scripts et d'en capturer la sortie avec la fonction Posix popen, ou à défaut, avec la fonction \_popen de la librairie C. Ceci posé, nous avons l'embarras du choix pour le toolkit graphique. J'en affectionne particulièrement deux, Tkinter et wxPython (<http://www.wxpython.org>). Tkinter, c'est l'extrême concision du code et de l'écriture, c'est aussi un déploiement simplifié, car Tkinter fait partie intégrante de Python, mais ce n'est pas un toolkit très riche, et dans certains cas, son look and feel ne s'accommodera pas avec le reste de l'application. wxPython est un toolkit extrêmement riche, mais moins rapide à coder que Tkinter, et qu'il faut déployer parallèlement à Python. En revanche, son look and feel se mariera parfaitement avec celui de wxWidgets. Pour



C++ et Python travaillant de concert.

l'exemple d'aujourd'hui, j'ai choisi Tkinter. Voici donc une application Python qui permet de saisir un texte depuis une boîte de dialogue. Le texte saisi sera injecté dans la sortie standard d'une part, et conservée dans un module Python auto-généré d'autre part.

### mon\_dialog.py

```
#!/usr/bin/env python
# -*- coding: iso-8859-1 -*-

# !! Attention code partiel

from Tkinter import *

class MonDialog(Frame):
    def __init__(self, parent):
        self.nom = None
        # etc,...

    def read_data(self):
        try:
            import data
            self.nom = data.nom
        except ImportError:
            self.nom = 'Programmez!'

    def write_data(self):
        try:
            fichier = open('data.py', 'w')
            fichier.write("nom = '" + self.nom + "'")
            fichier.close()
```

```

except IOError:
    import tkinter as tk
    tkMessageBox.showinfo("Problème!", "Erreur d'écriture des données")

def ok(self):
    self.nom = self.entry1.get()
    self.write_data()
    print self.nom
    self.quit()

def annuler(self):
    # si pas de sortie de données
    # mettre simplement un zéro de fin de chaîne
    # pour le istream de C++
    print "\x00"
    self.quit()

```

Les parties les plus intéressantes de ce script sont les fonctions `read_data` et `write_data`. La première essaie d'importer un script Python contenant des variables affectées. Ces variables sont les résultats du dialogue au précédent appel. Si le script n'existe pas, des valeurs par défaut sont utilisées. Si le script existe, le seul fait de l'importer donne vie aux variables dans son espace de nom, ce qui dispense d'écrire le moindre code devant parser un fichier de configuration. La méthode `write_data`, quant à elle, génère le script, tout simplement en écrivant dans un fichier à l'extension `.py`. Les méthodes `ok` et `annuler` du script sont triviales. On remarquera dans la dernière, comme il est facile avec Python d'injecter un caractère nul dans le flux. Cette opération est requise pour l'extraction côté C++. Voici maintenant le code C++. Il a été écrit sous Visual Studio .Net, autrement dit, sous Windows, ce qui entraîne l'emploi de `_popen` et `_pclose`. On les remplacera par `popen` et `pclose` sous Linux pour porter ce code.

### demopopen.cpp

```

#include <iostream>
#include <string>
#include <sstream>
#include <cstdio>

using namespace std;

const int buffer_size = 12;

int main()
{
    char buffer[buffer_size];
    FILE *tube;
    string nom;
    // Pour Windows. Pour Linux donner
    // string commande = "mon_dialog.py 2> /dev/null";
    string commande = "mon_dialog.py 2> nul";

    cout << "Demonstration d'utilisation de popen" << endl;
    cout << "Lancement de l'application fille" << endl << endl;

```

```

// _popen sous Windows
// popen sous Linux/UNIX
tube = _popen(commande.c_str(), "r");
if(tube == NULL)
{
    cout << "Probleme: la commande n'a pas pu être lancée" << endl;
    exit(1);
}
fgets(buffer, buffer_size, tube);
_pclose(tube); // pclose sous Linux/UNIX

istream flot(buffer, buffer_size);
flot >> nom;
cout << "Vous avez saisi le nom: " << nom << endl << endl;
cout << "Appuyez sur une touche" << endl;
cin.get();

return EXIT_SUCCESS;
}

```

Ce code est des plus simples. Le flux `stdout` capturé est copié dans un tampon, via `fgets`. Le tampon sert à construire un flux au sens C++ du terme et le contenu du flux est extrait jusqu'à ce qu'il soit vide. Pour utiliser cela en production, la seule chose à faire est d'affiner la gestion du flux afin, par exemple, que des espaces saisis dans un nom composé ne viennent perturber le mécanisme. Nous laissons ce point à la sagacité du lecteur. Enfin, on notera que le procédé ne conviendra pas dans tous les contextes, pour des raisons de sécurité. Sous Linux, on veillera, au minimum, à ce que le bit `SUID` de Python ne soit pas positionné, ainsi qu'on doit le vérifier pour tout exécutable lancé par `popen`.

## 2 Compiler une application embarquant l'interpréteur Python

Voici un code minimal qui embarque l'interpréteur Python et présente sa boucle interactive à l'utilisateur.

```

#include <iostream>

using namespace std;

#ifdef _DEBUG
#undef _DEBUG
#include <python.h>
#define _DEBUG
#else
#include <python.h>
#endif

int main(int argc, char **argv)
{
    Py_Initialize();

```

```
cout << "On embarque l'interpreteur" << endl;
Py_Main(argc, argv);

Py_Finalize();

cout << "Appuyez sur une touche" << endl;
cin.get();
return EXIT_SUCCESS;
}
```

On voit que tout ce qui se passe concernant l'interpréteur doit se situer entre les APIs `Py_Initialize` et `Py_Finalize`. Pour compiler un tel code, il faut que le compilateur puisse trouver l'en-tête `python.h`. L'édition de liens se fera avec `libpython23.so` sous Linux/Unix (pour un Python version 2.3 bien évidemment) et avec `python23.lib` sous Windows. Dans le code ci-dessus, vous pouvez remarquer que l'inclusion de l'en-tête `python.h` est placée dans un sandwich aux macros totalement indigestes. Vous devrez faire de même sous Visual Studio, si vous voulez compiler en mode DEBUG. Par défaut, l'éditeur de liens cherchera `python23_d.lib` et ne la trouvera pas, si vous avez seulement installé les binaires de Python. L'inclusion acrobatique ci-dessus est un moyen de contourner le problème, ce dernier ne se posant pas, si vous avez compilé votre Python à partir des sources. On met fin à l'interpréteur embarqué de façon classique soit `Ctrl-D` sous Linux et `Ctrl-Z` sous Windows.

### 3 Exécuter du code dans l'interpréteur

Si une chaîne de caractères est du code Python, on pourra faire exécuter cette chaîne dans l'interpréteur avec l'API `PyRun_SimpleString`. Reprenez l'exemple ci-dessus et remplacez l'appel à `Py_Main` par (`embedding2.cpp`) :

```
PyRun_SimpleString("import embedding2\n"
"embedding2.coucou(\n");
```

Vous trouverez tous les exemples complets sur le site du magazine, [www.programmez.com](http://www.programmez.com).

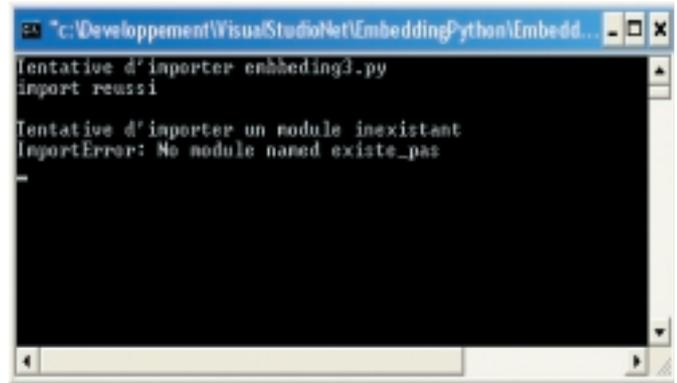
### 4 Gérer les erreurs

Embarquer Python n'est heureusement pas limité à l'exemple ci-dessus. Au contraire, Python nous offre un gigantesque jeu d'APIs très puissantes. Ces APIs, documentées, ont des noms parlants et sont préfixées de manière logique. Par exemple, `PyModule_xxxx` pour les API, agissant sur les module, `PyList_xxxx` pour les listes, `PyDict_xxxx` pour les dictionnaires, etc. Ces APIs retournent `NULL` ou `-1` en cas d'erreur et initialisent une exception. Le code `embedding3.cpp`, que vous trouverez sur le site, montre comment importer un module et comment traiter le cas d'une importation de module qui échouerait.

La capture ci-contre montre que le message d'erreur est le même que celui qu'émettrait l'interpréteur. D'autres APIs `PyErr_xxxx` sont, bien entendu, disponibles pour une gestion plus fine.

### 5 Dialoguer avec l'interpréteur

Ce que nous avons fait jusqu'ici est sans doute intéressant, mais d'une utilité assez limitée. Nous allons maintenant aborder quelque chose de beaucoup plus excitant, à savoir la collaboration d'une appli-



Gestion des erreurs survenues dans l'interpréteur Python depuis C++.

cation C++ avec l'interpréteur. Sur le site, vous trouverez un script baptisé `embedding4.py` dont voici le contenu :

```
#!/usr/bin/env python
# -*- coding: iso-8859-1 -*-

def echo(arg):
    print "Echo: " + arg

def get_liste():
    return ['fred', 1961]

if __name__ == '__main__':
    echo('Programmez!')
    print get_liste()
```

Nous nous proposons de charger ce script et d'invoquer ses deux fonctions. Nous illustrons ainsi, comment passer des paramètres à une fonction, ou extraire des valeurs d'une liste reçue comme valeur de retour. Voici : Encadré `embedding4.cpp`, le code qui fait ce travail.

#### `embedding4.cpp`

```
#include <iostream>
#include <string>

using namespace std;

#ifdef _DEBUG
#undef _DEBUG
#include <python.h>
#define _DEBUG
#else
#include <python.h>
#endif

int main()
{
    PyObject *module,
             *dictionnaire,
             *fonction,
```

```

        *tuple_param,
        *resultat;

Py_Initialize();

module = PyImport_ImportModule("embedding4");
if(!module)
{
    cout << "l'import a echoue. Placer embedding4.py dans le" << endl;
    cout << "meme repertoire que le present executable" << endl << endl;
    exit(3);
}
dictionnaire = PyModule_GetDict(module);

fonction = PyDict_GetItemString(dictionnaire, "echo");
if(fonction && PyCallable_Check(fonction))
{
    tuple_param = Py_BuildValue("(s)", "Programmez!");

    if(tuple_param)
    {
        resultat = PyObject_CallObject(fonction, tuple_param);
        if(resultat)
        {
            PyObject *result_as_Str = PyObject_Str(resultat);
            cout << PyString_AsString(result_as_Str) << endl;
            Py_DECREF(result_as_Str);
        }
        Py_DECREF(resultat);
    }
    Py_DECREF(tuple_param);
}
// pas de DECREF pour fonction

fonction = PyDict_GetItemString(dictionnaire, "get_liste");
if(fonction && PyCallable_Check(fonction))
{
    resultat = PyObject_CallObject(fonction, NULL);
    if(resultat)
    {
        PyObject *element;
        // On extrait le nom de la liste (rang 0)
        // On extrait la date de la liste (rang 1)
        element = PyList_GetItem(resultat, 0);
        cout << "Nom: " << PyString_AsString(element) << endl;
        element = PyList_GetItem(resultat, 1);
        cout << "Date: " << PyInt_AsLong(element) << endl;
        // La référence sur élément n'est jamais décrétementée
        // car élément appartient à la liste.
    }
    Py_DECREF(resultat);
    // Libérer la liste libérera les éléments qu'elle contient.
}

```

```

// pas de DECREF pour fonction
// pas de DECREF pour dictionnaire
Py_DECREF(module);
Py_Finalize();

cout << "Appuyez sur une touche" << endl;
cin.get();
return EXIT_SUCCESS;
}

```

Encore une fois, les choses sont simples, grâce à la variété des APIs dont nous disposons. Le point qu'il faut bien comprendre concerne le comptage de référence des objets Python, objet étant pris au sens large et non au sens d'instance de classe. Pour gérer et libérer la mémoire au cours de l'exécution, Python maintient un compteur de référence sur tout objet et quand le compteur tombe à zéro, l'objet est détruit. Côté C++ il est à la charge du programmeur de décrétement les références, au moyen d'une macro. `Py_DECREF`. La difficulté réside dans le fait de savoir quand décrétement ou pas. D'abord, lire la doc des APIs. S'il est



Une fonction sans valeur de retour retourne toujours l'objet `None` côté C++.

dit que l'API retourne une nouvelle référence, il faudra décrétement celle-ci après utilisation. S'il est dit que l'API retourne une référence empruntée (borrowed), il ne faudra pas décrétement. Une autre règle est que, quand on obtient une référence sur un objet déjà détenu, par exemple un élément dans une liste ou une fonction dans un module, la référence est empruntée et c'est le conteneur qui se chargera de la décrétement, ce qui est, somme toute, fort logique. Le code C++ ci-contre illustre bien ceci. Il convient de savoir encore une chose. Pour le programmeur Python, la fonction `echo` du script ne retourne rien, pour le programmeur C++ elle retourne une référence sur l'objet `None`, que nous imprimons sur la console pour illustrer ce phénomène. Cette référence est une nouvelle référence comme toute valeur de retour de fonction et en tant que telle, elle doit être libérée.

Le principal est dit quant à l'embarquement de Python. Rendre une application C++ extensible ou programmable, n'est plus qu'une affaire d'imagination, épaulée par un jeu d'APIs particulièrement riche.

■ Frédéric Mazué - [fmazue@programmez.com](mailto:fmazue@programmez.com)