OFPPT

ROYAUME DU MAROC

مكتب التكوين المهني وإنعاش الشغل

Office de la Formation Professionnelle et de la Promotion du Travail

RESUME THEORIQUE
&
GUIDE DES TRAVAUX PRATIQUES

MODULE N° : 15

TITRE DU MODULE : Empaquetage et déploiement d'une application Client/serveur

SECTEUR : NTIC

SPECIALITE : TSDI

Niveau : TECHNICIEN SPECIALISE

Version Juillet 2005

REMERCIEMENT

La DRIF remercie les personnes qui ont contribué à l'élaboration du présent document.

Pour la supervision :

MME.BENNANI WAFAE	DIRECTRICE CDC TERTIAIRE & TIC
M. ESSABKI NOURDDINE	CHEF DE DIVISION CCFF

Pour la conception :

- JELLAL ABDELILAH	Formateur animateur au CDC Tertiaire & TIC

Pour la validation :

Les utilisateurs de ce document sont invités à communiquer à la DRIF toutes les remarques et suggestions afin de les prendre en considération pour l'enrichissement et l'amélioration de ce programme.

Said Slaoui
DRIF

MODULE : Empaquetage et déploiement d'une application Client/serveur

Durée : 36 **h**

OBJECTIF OPERATIONNEL DE PREMIER NIVEAU DE **COMPORTEMENT**

COMPORTEMENT ATTENDU

Pour démontrer sa compétence, le stagiaire doit **empaqueter et déployer une application sur une architecture client serveur** selon les conditions, les critères et les précisions qu i suivent :

CONDITIONS D'EVALUATION

- Epreuve pratique

CRITERES GENERAUX DE PERFORMANCE

- Utilisation des commandes appropriées.
- Respect du temps alloué.
- Respect des règles d'utilisation du matériel et logiciel Informatique.

PRECISIONS SUR LE COMPORTEMENT ATTENDU	**CRITERES PARTICULIERS DE PERFORMANCE**
A. Empaqueter l'application	• Justesse de génération du .exe de l'application • Création adéquate de l'assistant de l'installation
B. Déployer l'application au niveau client/serveur	• Réalisation et test adéquat de la procédure de déploiement • Installation et paramétrage de l'application au niveau s erveur • Installation et paramétrage de l'application au niveau client
C. Définir les normes de l'exploitation et de la sécurité	• Création judicieuse d'un dossier d'utilisation ou d'exploitation • Implémentation appropriée d'une politique de sauvegarde et de sécurité

MODULE 15 : Empaquetage et déploiement d'une application Client/serveur

RESUME THEORIQUE

Dans ce chapitre, vous allez apprendre à empaqueter et déployer une application sur une architecture client/serveur.
Le client/serveur n'est pas un des phénomènes de mode, c'est une étape fondamentale dans l'évolution des architectures informationnelles et la mise en place d'une informatique répartie répondant aux besoins de l'utilisateur final.
Les systèmes de gestion de bases de données relationnels (SGBD -R) est au centre des architectures client/serveur qui appréhende ces trois besoins dans un cadre uniforme et cohérent.
Les environnements logiciels et matériels, ouvert grâces à ce standard, permet de pérenniser les investissements applicatifs en termes de flexibilité, portabilité, interopérabilité et sécurité.
Les architectures client/serveurs constituent une première réponse globale satisfaisante pour l'utilisateur qui veut toujours plus (fonctionnellement), plus vite (informationnellement) moins cher, ce que la chute des coûts du matériel et du logiciel l'autorise à penser !

I. Principes de base

. Processus d'empaquetage

Cette opération consiste à créer un empaquetage permettant d'installer votre application sur l'ordinateur d'un utilisateur. Un *empaquetage* se compose d'un ou de plusieurs fichiers .cab contenant vos fichiers projet compressés, ainsi que tous les autres fichiers qu'il est nécessaire d'installer pour que l'application fonctionne. En l'occurrence, il peut s'agir de programmes d'installation, de fichiers .cab secondaires, ou autres.
Vous pouvez librement distribuer toutes applications ou tous composants redistribuables créés avec **Microsoft® Office Developer**. Notez que comme Microsoft® FrontPage® et Microsoft® Outlook® gèrent uniquement des projets fondés sur des profils utilisateur, l'Assistant Empaquetage ne peut pas créer d'empaquetage en utilisant ces produits. Vous pouvez cependant créer un empaquetage et déployer des projets autonomes comme les projets compléments, créés dans l'environnement Microsoft® Visual Basic® pour Applications (VBA) indépendamment de l'application concernée. De plus, l'Assistant Empaquetage ne peut pas empaqueter des applications de workflow pour les projets Microsoft® Exchange Server ou de tableau de bord interactif.
En plus de documents, de feuilles de calcul, de classeurs ou d'autres fichiers Office, il se peut que votre application nécessite d'autres fichiers créés par vous, comme des DLL, des contrôles Microsoft® ActiveX® (fichiers .ocx) ou des bitmaps (fichiers .bmp). Le cas échéant, vous pouvez aussi inclure Microsoft Access RunTime et Graph9.exe dans l'empaquetage de votre application. L' Assistant Empaquetage facilite l'empaquetage et la distribution de tous ces fichiers. Pour plus d'informations sur Access Runtime, voir : Déploiement des applications Microsoft Access à l'aide d'Access Runtime.

Remarque Vous avez également la possibilité de distribuer des contrôles ActiveX, des fichiers .exe et des DLL achetés dans le commerce. Néanmoins, pour chacun d'eux, consultez l'accord de licence de l'éditeur pour savoir si vous êtes autorisé à le faire.

Vous pouvez également utiliser l'Assistant Empaquetage pour ajouter des fichiers de dépendances. Ces fichiers répertorient les composants d'exécution qui doivent être distribués avec les fichiers projet de votre application.

g. Les différentes étapes du processus d'empaquetage :

geD. Identification des fichiers à distribuer

Avant de pouvoir créer un empaquetage, l'Assistant doit identifier les fichiers projet et les fichiers dépendants nécessaires à l'application. Les fichiers projet sont les fi chiers inclus dans l'application proprement dite — par exemple, le fichier. vbo et son contenu. Les fichiers dépendants sont les fichiers ou composants d'exécution nécessaires au fonctionnement de l'application. Les informations relatives aux dépendances sont stockées dans différents fichiers .dep correspondant aux différents composants de votre projet.

eD. Création, le cas échéant, de fichiers de dépendances pour les composants de l'application

Si, au cours de l'étape précédente, il est apparu que votre ap plication nécessitait des fichiers de dépendances, créez ces fichiers avant de créer l'empaquetage proprement dit. Vous pouvez ensuite inclure ces fichiers dans l'empaquetage.

eD. Choix de l'emplacement d'installation des fichiers sur l'ordinateur de l'utilisateur

En règle générale, les fichiers programme et les fichiers d'installation sont installés dans un sous - répertoire du répertoire Program Files, tandis que les fichiers système et les fichiers dépendants sont installés dans le répertoire \Windows\System ou \Winnt\System32. Votre programme d'installation doit en tenir compte lorsqu'il détermine où installer chaque fichier.

innt . Création de l'empaquetage

L'Assistant crée l'empaquetage et le programme d'installation (setup1.exe) correspondant, en référençant tous les fichiers requis. A l'issue de cette étape, un ou plusieurs fichiers .cab sont créés, ainsi que les fichiers d'installation nécessaires.

. Assistant Empaquetage « Microsoft® Office XP Developer »

L'Assistant Empaquetage procure une méthode perme ttant d'empaqueter et d'ajouter facilement des routines d'installation professionnelles dans les applications. Il vous guide pas à pas dans les opérations qui consistent à créer les fichiers .cab de votre application, à regrouper ces fichiers dans un empaquetage contenant toutes les informations nécessaires à l'installation, et à distribuer ces empaquetages aux utilisateurs avec un programme d'installation.
L'Assistant Empaquetage peut être ouvert de deux façons différentes : depuis le menu **Démarrer** de Windows ou depuis une application Office.
Pour lancer l'Assistant Empaquetage, dans le menu **Démarrer**, sélectionnez **Programmes**, puis **Microsoft® Office XP Developer** et enfin **Assistant Empaquetage**.
Pour lancer l'Assistant Empaquetage depuis une application Offi ce, ouvrez Microsoft® Visual Basic Editor (ALT+F11) et dans le menu **Compléments**, sélectionnez **Assistant Empaquetage**.
Remarque Si l'Assistant Empaquetage n'est pas accessible à partir du menu, utilisez le Gestionnaire de compléments accessible à partir d u menu pour le charger.

Définissez toutes les options nécessaires pour votre fichier, puis cliquez sur **Terminer**. Pour plus d'informations sur les écrans de l'Assistant, cliquez sur **Aide** ou sélectionnez une rubrique dans la liste ci-dessous.

id. Création d'un programme d'installation avec l'Assistant Empaquetage

L'Assistant Empaquetage facilite la création et le déploiement des programmes d'installation pour les applications Microsoft® Office XP Developer. L'Assistant vous guide à travers les étapes de la création d'un programme d'installation contenant toutes les informations nécessaires à l'installation.

Remarque Vous pouvez également créer plusieurs empaquetages pour une même application afin de faciliter l'empaquetage de votre application sous différentes formes, par exemple, différents empaquetages peuvent être utilisés pour chaque langue dans laquelle l'application est déployée.

Remarque Si votre application est un modèle ou un complément propre à une application, il est préférable d'éviter de créer un programme d'installation avec l'Assistant Empaquetage car ce dernier ne permet pas d'installer le modèle ou le complément dans les dossiers spéciaux qui leur sont réservés sur les ordinateurs des utilisateurs.

é. Pour créer un programme d'installation

- Ouvrez Visual Basic Editor (Alt+F11) à partir de l'application Office dans laquelle vous avez créé l'application.
- Dans le menu **Compléments**, choisissez **Assistant Empaquetage**.

Remarque Si l'Assistant Empaquetage ne figure pas dans le menu, vous devez le charger en utilisant le Gestionnaire de compléments accessible à partir du menu.

- Dans l'écran **Identifiez l'application et l'empaquetage**, sélectionnez l'empaquetage à construire pour le fichier principal sélectionné.
- Dans l'écran **Informations sur l'application**, indiquez les informations appropriées, relatives à votre application. Dans le champ **Langue d'installation**, sélectionnez la langue que le programme d'installation doit utiliser.
- Dans l'écran **Liens de dépendance**, vérifiez que tous les fichiers requis, à inclure dans l'empaquetage, sont sélectionnés. Ajoutez les fichiers supplémentaires de votre choix dans l'empaquetage.

Les écrans suivants qui apparaissent dépendent du type d'application que vous empaquetez. Fournissez les informations propres à l'application dans les écrans de l'Assistant.

Dans l'écran **Modifier les emplacements d'installation**, cliquez sur **Suivant** pour sélectionner les emplacements d'installation par défaut ou modifiez les emplacements.

Dans l'écran **Définir les raccourcis du menu Démarrer**, cliquez sur **Suivant** pour accepter les raccourcis par défaut. Pour plus d'informations sur la personnalisation des raccourcis du menu **Démarrer**.

Dans l'écran **Exécuter lorsque l'installation est terminée**, cochez la case **Exécuter cette commande lorsque l'installation est terminée** pour faire en sorte qu'un fichier soit démarré à la fin de l'installation. Sélectionnez ou indiquez un nom de fichier dans le champ requis (ou toute commande

pouvant être exécutée à partir de la boîte de dialogue **Exécuter** accessible à partir du menu **Démarrer**).

Remarque Le fichier que vous choisissez d'exécuter en fin d'installation doit être ajouté à l'empaquetage s'il ne doit pas être installé sur l'ordinateur de l'utilisateur.

- Dans l'écran **Créer l'empaquetage de l'application**, choisissez l'option **Créer le programme d'installation** pour créer l'empaquetage dès maintenant ou bien, choisissez l'option **Enregistrer le script d'empaquetage sans le créer** pour enregistrer la définition de l'empaquetage pour une création ultérieure.
- Cliquez sur **Terminer** pour fermer l'Assistant Empaquetage. L'Assistant démarre l'empaquetage de l'application.
- Dans la boîte de dialogue **Rechercher un dossier**, sélectionnez le dossier dans lequel enregistrer l'application empaquetée.

Remarque Pour l'enregistrer sur le réseau, vous pouvez copier puis coller le chemin d'accès au réseau dans le champ réservé à l'indication du dossier.

- Si votre empaquetage contient les composants Access Runtime, vous êtes invité à enregistrer les fichiers Access Runtime dans un dossier local et à insérer le CD-ROM Microsoft Office (il ne s'agit pas du CD-ROM Office Developer).

Dès que l'Assistant Empaquetage a terminé de créer l'empaquetage, vous pouvez trouver le programme d'installation dans le dossier sélectionné au cours de la dernière étape de l'Assistant. Vous pouvez alors tester votre programme d'installation.

Pour créer de nouveau l'empaquetage, ouvrez l'Assistant Empaquetage, sélectionnez le fichier principal de l'application, puis sélectionnez l'empaquetage que vous voulez créer. Si vous ne souhaitez modifier aucun paramètre, cliquez sur Terminer pour créer l'empaquetage.

Pour modifier un empaquetage, ouvrez l'Assistant Empaquetage, sélectionnez le fichier principal de l'application, puis sélectionnez l'empaquetage que vous voulez modifier. Poursuivez les étapes de l'Assistant en apportant les modifications souhaitées. Une fois les modifications apportées, cliquez sur Terminer pour créer l'empaquetage avec les nouveaux paramètres. Notez que l'Assistant Empaquetage enregistre les paramètres de l'empaquetage.

. Déploiement d'une application

Cette opération consiste à créer un empaquetage permettant d'installer votre application sur l'ordinateur d'un utilisateur. Un empaquetage se compose d'un ou de plusieurs fichiers .cab contena nt vos fichiers projet compressés, ainsi que tous les autres fichiers qu'il est nécessaire d'installer pour que l'application fonctionne. En l'occurrence, il peut s'agir de programmes d'installation, de fichiers .cab secondaires, ou autres.

Vous pouvez librement distribuer toutes applications ou tous composants redistribuables créés avec Microsoft® Office Developer. Notez que comme Microsoft® FrontPage® et Microsoft® Outlook® gèrent uniquement des projets fondés sur des profils utilisateur, l'Assistant Empa quetage ne peut pas créer d'empaquetage en utilisant ces produits. Vous pouvez cependant créer un empaquetage et déployer des projets autonomes comme les projets compléments, créés dans l'environnement Microsoft® Visual Basic® pour Applications (VBA) indép endamment de l'application concernée. De plus, l'Assistant Empaquetage ne peut pas empaqueter des applications de workflow pour les projets Microsoft® Exchange Server ou de tableau de bord interactif.

En plus de documents, de feuilles de calcul, de classeu rs ou d'autres fichiers Office, il se peut que votre application nécessite d'autres fichiers créés par vous, comme des DLL, des contrôles Microsoft® ActiveX® (fichiers .ocx) ou des bitmaps (fichiers .bmp). Le cas échéant, vous pouvez aussi inclure Microsoft Access RunTime et Graph9.exe dans l'empaquetage de votre application. L' Assistant Empaquetage facilite l'empaquetage et la distribution de tous ces fichiers.

Remarque Vous avez également la possibilité de distribuer des contrôles ActiveX, des fichiers .exe et des DLL achetés dans le commerce. Néanmoins, pour chacun d'eux, consultez l'accord de licence de l'éditeur pour savoir si vous êtes autorisé à le faire.

Vous pouvez également utiliser l'Assistant Empaquetage pour ajouter des fichiers de dépendances. Ces fichiers répertorient les composants d'exécution qui doivent être distribués avec les fichiers projet de votre application.

α. Déploiement des modèles Office et des compléments propres aux applications

L'Assistant Empaquetage est un outil qui convient à la création des programmes d'installation pour un grand nombre d'applications Microsoft® Office. Toutefois, il existe une exception à cette règle pour les modèles Office et les compléments propres aux applications (et non pour les suppléments COM) qui doivent être installés dans des dossiers particuliers sur l'ordinateur de l'utilisateur, car l'Assistant Empaquetage ne permet pas d'effectuer cette installation.
Sur un ordinateur équipé de Microsoft® Windows® et doté de profils utilisateur désactivés, les modèles et les compléments Office doivent être respectivement installés dans les dossiers suivants :
C:\Windows\ApplicationData\Microsoft\Templates
C:\Windows\Application Data\Microsoft\Addins
Si les profils utilisateur sont utilisés, les modèles et les compléments Office doivent être installés dans les dossiers suivants :
Si votre système d'exploitation est Windows 98, Windows Me ou Microsoft® Windows NT® 4 :

C:\Dossier Windows\Profiles\Nom_utilisateur\Application Data\Microsoft\Templates
C:\Dossier Windows\Profiles\Nom_utilisateur\Application Data\Microsoft\Addins
Si votre système d'exploitation est Windows 2000 ou une version ultérieure :
C:\Documents and Settings\Nom_utilisateur\Application Data\Microsoft\Templates
C:\Documents and Settings\Nom_utilisateur\Application Data\Microsoft\Addins
L'installation d'un modèle ou d'un complément dans l'un de ces dossiers n'est pas absolument nécessaire, mais elle présente cependant les avantages suivants :

- Si un modèle personnalisé de Microsoft® Word, Microsoft® Excel ou Microsoft® PowerPoint® est stocké sur les ordinateurs des utilisateurs, il doit être installé dans le dossier Templates pour qu'il apparaisse dans la boîte de dialogue Nouveau accessible à partir du menu Fichier.
- Comme les modèles et les compléments sont contenus par défaut dans les dossiers Templates et AddIns, leur installation à cet emplacement facilite la tâche des utilisateurs qui doivent localiser un modèle ou un complément. Par exemple, les compléments Excel (fichiers .xla) qui sont installés dans le dossier AddIns s'affichent automatiquement dans la liste des macros complémentaires disponibles de la boîte de dialogue Macro complémentaire (menu Outils), de sorte que les utilisateurs n'ont pas à rechercher le fichier correct.
- Les dossiers Templates et AddIns sont des fichiers sécurisés ; cela signifie qu'un complément ou un modèle contenu dans un de ces dossiers ne doit pas être signé numériquement par une source sécurisée pour exécuter du code quand la sécurité a la valeur haute. Ces dossiers sont sécurisés par défaut, mais il est possible d'annuler ce paramétrage en désactivant la case à cocher Faire confiance à tous les modèles et compléments installés dans l'onglet Sources fiables de la boîte de dialogue Sécurité.

Pour ouvrir cette boîte de dialogue dans les applications Office, pointez sur Macro dans le menu Outils, puis choisissez Sécurité. Si vous modifiez le dossier par défaut des modèles et des compléments, le nouveau dossier que vous indiquerez sera sécurisé.

Il existe des façons simples de déployer des modèles et des compléments vers les dossiers Templates et AddIns. Vous pouvez utiliser l'une ou l'autre de ces stratégies ou créer un simple programme d'installation personnalisé. Pour plus d'informations, voir Déploiement des compléments Office propres aux applications.

anne. Définition des profils utilisateur

Microsoft Windows fournit une fonction de profils utilisateur. Un profil utilisateur est un compte maintenu par le système d'exploitation qui assure le suivi des fichiers d'un utilisateur en particulier ainsi que de la configuration du système.
Dans Microsoft® Windows NT® Workstation, Windows NT Server et Windows 2000, les profils utilisateur sont automatiquement activés en permanence. A chaque fois qu'un nouvel utilisateur se connecte, son profil utilisateur est créé. Sous Windows NT, le profil est créé sous le dossier C:\Winnt\Profiles\Nom_utilisateur. Sous Windows 2000, il est créé sous C:\Documents and Settings\NomUtilisateur. Dans d'autres versions de Microsoft Windows, les profils utilisateur sont facultatifs.
Lorsque vous vous connectez à un ordinateur comme utilisateur disposant d'un profil, Windows vérifie les données stockées pour ce profil et charge les paramètres correspondants. De plus, Windows gère

un dossier qui contient les fichiers créés pour ce profil utilisateur :
C:\Windows\Profiles\Nom_utilisateur.
Sous les systèmes d'exploitation mentionnés plus haut, il est possible de sécuriser le système de façon que chaque utilisateur ait seulement accès aux fichiers, aux applications et à la configuration du système définis pour son profil. Par exemple, les fichiers installés par un utilisateur détenant des privilèges administratifs seront interdits aux utilisateurs dépourvus de ces privilèges.

sn. Déploiement des modèles Office personnalisés

Lors de la création d'un nouveau document dans Word, Excel ou PowerPoint, la boîte de dialogue Nouveau (menu Fichier) affiche la liste des modèles parmi lesquels vous pouvez faire votre sélection pour créer un nouveau document. Cette boîte de dialogue peut afficher trois types de modèles : les modèles intégrés inclus dans Microsoft Office, les modèles créés par les utilisateurs et stockés sur leurs ordinateurs et enfin, les modèles de groupe de travail, qui sont stockés sur un partage réseau. Seuls les modèles provenant de certains dossiers sont affichés ; mais si vous créez un nouveau modèle, vous devez l'enregistrer dans l'emplacement approprié pour qu'il apparaisse dans la boîte de dialogue. Le tableau suivant indique les emplacements où la boîte de dialogue recherche chaque type de modèle .

Type de modèle	Emplacement par défaut	Remarques
Intégré	C:\Program Files \Microsoft Office\Templates \LanguageID	L'identificateur de langue est un nombre indiquant la langue du produit. Pour l'anglais américain, ce nombre est 1033. Les modèles créés par l'utilisateur et enregistrés ici ne figureront pas dans la boîte de dialogue Nouveau.
Créé par l'utilisateur	C:\Windows\Application Data \Microsoft\Templates - ou - C:\Windows\Profiles\Nom_utilisateur \Application Data\Microsoft \Templates - ou - C:\Documents and Settings \Nom_utilisateur\Application Data \Microsoft\Templates - ou - C:\Winnt\Profiles\Nom_utilisateur \Application Data \Microsoft\Templates	Dans Word ou Excel, vous pouvez modifier cet emplacement dans la boîte de dialogue Options du menu Outils. Pour qu'ils s'affichent dans la boîte de dialogue Nouveau, les modèles créés par l'utilisateur doivent être stockés dans l'emplacement spécifié dans la boîte de dialogue Options.
Groupe de travail	Non spécifié	Vous pouvez indiquer l'emplacement des modèles de groupes de travail dans la boîte de dialogue Options de Word ou en utilisant le code VBA d'Excel ou de Word. Lorsque vous effectuez ce paramétrage, les modèles de l'emplacement spécifié s'affichent dans la boîte de dialogue Nouveau.

Comme vous pourrez le constater, deux possibilités s'offrent à vous pour installer un modèle personnalisé. Vous pouvez soit l'installer en local sur les ordinateurs des utilisateurs, soit le copier dans un dossier partagé sur un serveur du réseau.

. **Installation d'un modèle personnalisé en local**

Pour installer un modèle personnalisé sur les ordinateurs des utilisateurs, vous disposez de deux solutions :

- Ø Vous pouvez envoyer le modèle aux utilisateurs par courrier électronique accompagné des instructions leur expliquant où le copier sur leur ordinateur ou bien, leur demander de le copier à partir d'un partage réseau.
- Ø Vous pouvez créer un programme d'installation personnalisé pour copier le modèle dans le dossier approprié.

. **Déploiement d'un modèle de groupe de travail vers un partage réseau**

Si vous avez devez déployer un modèle personnalisé vers un grand nombre d'utilisateurs, vous pouvez le copier dans un dossier partagé sur le serveur du réseau pour qu'il soit disponible pour tous ceux qui ont accès au partage.
Après avoir copié le modèle dans le dossier partagé, vous devez vérifier que chaque utilisateur a bien indiqué le chemin d'accès aux modèles du groupe de travail. Word et Excel proposent une option qui remplit cette fonction. Cette option, accessible aux utilisateurs ainsi qu'à vous-même, permet de pointer vers le dossier partagé contenant l'application de chaque utilisateur.
Il existe plusieurs façons de définir l'option des modèles du groupe de travail :

- Ø Vous pouvez donner des instructions à chaque utilisateur pour qu' il accomplisse cette tâche.
- Ø Vous pouvez écrire du code pour définir cette option et exécuter le code sur les ordinateurs de tous les utilisateurs.
- Ø Pour les modèles personnalisés de Word, si le fichier Normal.dot est sous contrôle administratif, vous pouvez définir l'option ici. Dans ce cas de figure, le fichier Normal.dot est probablement en lecture seule de façon à ce que les utilisateurs ne puissent pas le modifier.

Dans Word, vous pouvez modifier l'option des modèles de groupe de travail en utilisant du code VBA (Microsoft Visual Basic pour Applications), en définissant la propriété DefaultFilePath de l'objet Options et en passant la constante wdWorkgroupTemplatesPath pour l'argument path. C'est ce que montre le fragment de code suivant :

```
Options.DefaultFilePath(wdWorkgroupTemplatesPath)=
"\\Server\Share\WorkgroupTemplates"
```

Dans Excel, vous pouvez modifier l'option des modèles de groupe de travail en utilisant le code VBA pour définir la propriété NetworkTemplatesPath de l'objet Application :

```
Application.NetworkTemplatesPath = "\\Server\Share\WorkgroupTemplates"
```

Si vous disposez de Visual Basic, vous pouvez, par exemple, écrire un simple programme qui vous permet de définir cette option en lançant Word ou Excel par l'intermédiaire d'Automation ; vous pouvez ensuite compiler le programme en un fichier .exe et le distribuer à tous vos utilisateurs par courrier électronique. Le programme définira correctement l'option des modèles de groupe de travail quand il sera exécuté par les utilisateurs.

Il est utile, sur le plan de la maintenance, de stocker un modèle de groupe de travail sur un partage réseau commun car vous pouvez en effet modifier le modèle sans avoir à le redistribuer à tous les utilisateurs. Toutefois, si vous devez déployer le modèle vers les utilisateurs qui n'ont pas accès au partage réseau commun, vous pouvez créer un programme d'installation personnalisé pour déployer un modèle.

. **Distribution de suppléments COM**

Si vous avez créé un complément COM avec Microsoft® Office XP Developer, vous pouvez créer un programme d'installation pour ce complément en utilisant l'Assistant Empaquetage. Le programme d'installation doit déployer la DLL du supplément COM, ainsi que tous les fichiers dont elle est dépendante comme les bibliothèques de types qui peuvent être absentes des ordinateurs des utilisateurs.

Une DLL de supplément COM créée avec le concepteur de compléments de Microsoft Office Developer ou Visual Basic est auto-enregistrée. Cela signifie qu'elle s'enregistre correctement à l'exécution du programme. A l'issue de cette opération, le supplément COM apparaît dans la liste des suppléments COM disponibles dans la boîte de dialogue Suppléments COM. Par conséquent, peu importe où vous installez la DLL. Cependant, par souci de cohérence, il est préférable de choisir un seul dossier pour installer tous les suppléments COM. Il peut s'agir par exemple du dossier AddIns dont le chemin d'accès est C:\Windows\Application Data\Microsoft, C:\Windows\Profiles\Nom_utilisateur\Application Data\Microsoft, C:\Winnt\Profiles\Nom_utilisateur\Application Data\Microsoft or C:\Documents and Settings\Nom_utilisateur\Application Data\Microsoft

II) Déploiement d'application Windows avec Visual Studio.NET

1) Généralités et méthodes de déploiement en .NET.

Le déploiement est le procédé qui consiste à distribuer une application ou un composant afin de l'installer sur une machine différente. Pour qu'une application fonctionne, il faut que tous les composants qu'elle utilise soient présents dans la version attendue sur le poste cible.

Visual Studio.NET utilise la technologie Windows Installer pour redistribuer les applications Windows. En VB6, nous avions l'habitude d'utiliser soit l'assistant d'empaquetage et de déploiement (Package and Deployment Wizard), soit Visual Studio Installer (VSI).

Attention : Bien que VSI utilise la technologie Windows Installer, les projets Windows Installer (.wip) créés avec VSI ne peuvent pas être ouverts avec Visual Studio .NET.

Voici une ressource sur Windows Installer :
http://www.microsoft.com/windows2000/techinfo/howitworks/management/installer.asp

Avec les nouvelles fonctionnalités apportées par .NET (assemblies auto descriptives, CLR commune à tous les langages managés, etc…), la manière de déployer des applications Windows est impactée. Il existe maintenant plusieurs méthodes pour installer son application .NET sur une machine cible.

Nous allons rapidement les présenter ci-dessous.

a) Déploiement à l'aide de fichiers Windows Installer (.msi)

Traditionnellement, les applications Windows ont été installées à l'aide de programmes exécutables et plus récemment à l'aide de fichiers Windows Installer (.msi).

Il ne faut pas perdre de vue que pour installer une application Windows avec Windows Installer, l'utilisateur doit avoir les privilèges suffisants. Ces derniers dépendent des actions effectuées par Windows Installer ainsi que de la plateforme cible. Par exemple, sous Windows 95/98/Me, aucun privilège n'est nécessaire alors que sous Windows 2000/XP, pour faire une action aussi simple que créer un dossier sous « Program Files », il faut appartenir à un groupe ayant des privilèges élevés comme « Utilisateurs avec pouvoir » ou « Administrateurs ».
Si un utilisateur, non membre du groupe Administrateur local, essaie d'installer un fichier .msi sur Windows 2000, il peut obtenir un message lui disant qu'il n'a pas les droits nécessaires pour effectuer l'installation. Il lui sera alors possible d'effectuer l'installation sous un autre profil utilisateur ayant les permissions nécessaires (administrateur par défaut). L'utilisateur devra alors fournir le mot de passe de ce compte.
On peut également utiliser des outils comme SMS pour installer une application avec les droits administrateurs sur tous les postes.

b) Déploiement par simple copie

La nouvelle architecture de .NET permet de déployer par simple copie des fichiers :

- ? Distribution à l'aide de XCOPY via la console
- ? Distribution à l'aide du copier/coller dans l'explorateur Windows
- ? Distribution à l'aide de commandes FTP
- ? Distribution aux utilisateurs via email.

Il faut quand même noter que certaines fonctionnalités attendues lors d'un déploiement ne sont pas présentes lors de ce type d'opérations (créer un raccourci, vérifier que tous les composants sont présents,…).

De plus, il faut noter que ce type de déploiement ne peut pas être utilisé si l'on fait de l'interopérabilité COM.

c) Utilisation du déploiement Internet

Une nouvelle fonctionnalité d'empaquetage d'application Windows Forms est née avec .NET. Cette approche est appelée « déploiement d'applications .NET par Internet ». Elle fonctionne de la manière suivante :

- . Vous stockez vos fichiers (assemblies exe ou dlls) sur un serveur.
- . Les utilisateurs se connectent à l'application à travers leur navigateur (via http) ce qui provoque l'apparition d'une boîte de dialogue « Exécuter ».
- . Les fichiers initiaux et les assemblies immédiatement nécessaires sont téléchargées dans le répertoire « <windir>\assembly\download\ » et le répertoire « Temporary Internet Files ».
- . Chaque ressource supplémentaire nécessaire est alors téléchargée dans ces mêmes répertoires au cours de l'utilisation de l'application.

Vous trouverez de plus amples informations sur cette nouvelle fonctionnalité aux adresses suivantes :

- Death of the Browser? :
  http://msdn.microsoft.com/library/en-us/dnadvnet/html/vbnet10142001.asp
- Security for Downloaded Code :
  http://msdn.microsoft.com/library/en-us/dnadvnet/html/vbnet12112001.asp

Dans ce document, nous nous concentrerons sur la technologie Windows Installer proposée par Visual Studio.NET.

2) Les types de projets de déploiement proposés par VS.NET

Nous avons la possibilité avec Visual Studio .NET de créer 4 types de projet d'installation :

Type de projet	Utilisation

Type de projet	Utilisation
Projet de configuration (Setup)	Construit un programme d'installation pour des applications Windows.
Projet de configuration Web (Web Setup)	Construit un programme d'installation pour des applications Web.
Projet de module de fusion (Merge Module)	Empaquette un composant qui doit être utilisé par plusieurs applications et qui peut ainsi être ajouté à un package msi.
Projet Cab	Crée un fichier Cabinet téléchargeable à travers un navigateur Internet.

Attention : le type de projet ne peut pas être changé après avoir été créé. Ainsi, si on a choisi de créer un projet de configuration pour une application Windows, on ne peut pas, par la suite, l'utiliser pour le déployer sur le Web. Il faudra recréer un projet de configuration Web.

La différence entre un projet « Setup » et un projet « Web Setup » réside dans le choix du répertoire destination de l'installation :

- un projet « Setup » installera, par défaut, les fichiers de l'application Windows dans le dos sier « Program Files » de l'ordinateur cible.
- Un projet « Web Setup » installera les fichiers de l'application Web dans un répertoire virtuel racine sur le serveur Web.

Dans ce document, nous nous intéresserons uniquement aux projets « Setup » permettant de déployer des applications Windows.

3) Les atouts de Windows Installer

On utilise en général la technologie Windows Installer pour créer des programmes d'installation traditionnels.
Quand une application Windows nécessite des fonctionnalités d'installati on traditionnelles, la technologie Windows Installer semble la plus indiquée pour l'empaquetage et le déploiement.
En effet, cela permet :

- ? D'obtenir un programme d'installation ayant une interface graphique facile à utiliser
- ? Une intégration dans le « Panneau de Configuration » pour :
  - ? Installer
  - ? Désinstaller
  - ? Ajouter ou retirer des fonctionnalités de l'application
  - ? Réparer une installation corrompue
- ? D'obtenir un programme d'installation qui :
  - ? Restore le système dans son état initial en cas d'échec d'une partie du setup.
  - ? Restore le système dans son état initial en cas d'arrêt du setup par l'utilisateur (cancel par exemple).
- ? De prendre en charge les fonctionnalités suivantes :
  - ? Manipulation du Registre.

- ? Association de types de fichiers à l'application.
- ? Installation d'assemblies dans le Global Assembly Cache (GAC).
- ? Personnalisation de tâches à effectuer après le succès de l'installation
- ? Vérification des prérequis logiciels et matériels avant l'installation.

Du point de vue du développeur, cela permet également d'utiliser l'architecture de gestion des versions d'applications afin de mettre à jour ou patcher une application dans le bon ordre.

Pour plus d'information sur les mises à jour d'application avec Windows Installer, vous pourrez consulter le document suivant :
Small update et minor upgrade avec Windows Installer 2.0 :
http://www.microsoft.com/france/msdn/support/colones/default.asp

Du point de vue de l'administrateur système, les applications empaquetées au format .msi offrent les fonctionnalités suivantes :

- L'intégration avec des outils de distribution électronique de logiciels, comme System Management Server (SMS) ou Microsoft Active Directory™ « Directory Service Group Policies for software distribution », permettant un déploiement simplifié sur tout un parc machine.
- La possibilité d'installation en mode silencieux c'est-à-dire sans interaction avec l'utilisateur.

De plus, lorsque notre application fait de l'interopérabilité COM (par exemple, on utilise un .ocx COM depuis .NET), la seule solution pour installer l'application est un projet Windows Installer.

4) Comment redistribuer Windows Installer en cas d'absence sur le poste cible.

Les packages de Visual Studio Installer étant des packages msi, on sent bien qu'un premier problème va se poser : Quid si Windows Installer n'est pas présent sur la machine cible ?
Dans ce cas, il faudra l'installer ! On a la possibilité, dans les projets « Setup » de joindre le redistribuable Windows Installer à notre programme d'installation qui, dès lors, vérifiera la présence de Windows Installer sur la machine cible et l'installera en cas d'absence.

Pour ce faire, voici la démarche :

. clic droit sur notre projet « Setup » dans le Solution Explorer de VS.NET puis sélectionner « Propriétés ».
. dans la liste « Programme d'amorçage » sélectionner « Programme d'amorçage de Windows Installer »
. cliquer « OK ».

Ceci a pour conséquence, lors de la génération de la solution, de voir apparaître en plus de notre fichier .msi, 4 fichiers supplémentaires :

- Setup.exe: le fichier qui va déterminer si Windows Installer est présent sur la machine cible
- Setup.ini: le fichier qui indique à Setup.exe le nom de votre fichier msi à installer
- Instmsiw.exe: Windows Installer pour les PC avec Windows NT.
- Instmsia.exe: Windows Installer pour les PC avec Windows 95 ou Windows 98.

Ces fichiers permettront d'installer Windows installer en cas d'absence et ceci quel que soit le système d'exploitation Windows. Il faut noter que Windows Installer est déjà présent sur Windows 2000 et Windows XP.

III) Comment redistribuer le Framework .NET en cas d'absence sur le poste cible ?

Une application .NET nécessite que le Framework.NET soit installé sur la machine cible. Ce dernier ne s'installe pas sur toutes les plateformes. De plus, il ne s'installe que si une version **5.01 ou supérieure** de Internet Explorer est présente sur le poste.

1) Prérequis à l'installation du Framework.NET

Voici la liste des plateformes où le framework.NET peut être installé :

- Microsoft® Windows® 98
- Microsoft Windows 98 Second Edition
- Microsoft Windows Millennium Edition (Windows Me)
- Microsoft Windows NT® 4 (Workstation or Server) si le service pack 6a est installé
- Microsoft Windows 2000 (Professional, Server, ou Advanced Server) avec le dernier service pack ainsi que les dernières mises à jour critiques (téléchargeable sur "Microsoft Security Web site" : http://www.microsoft.com/security/)
- Microsoft Windows XP (Home ou Professional)
- Microsoft Windows .NET Server family

De plus, il est conseillé d'installer les composants suivants (en fonction des besoins de votre application) :

- MDAC version 2.6 et supérieures pour l'accès aux données, disponible ici : Universal Data Access Web site. MDAC 2.7 est conseillé.
- WMI, disponible ici : http://msdn.microsoft.com/library/default.asp?url=/downloads/list/wmi.asp .

Voici un lien en anglais où toutes les précisions sur les prérequis à installer sont données :
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnnetdep/html/dotnetfxref.asp

Remarque : l'environnement de développement Visual Studio .NET a des prérequis différents en terme de plateforme :
http://msdn.microsoft.com/vstudio/productinfo/sysreqs/default.asp.

Il y a plusieurs méthodes pour redistribuer le Framework.NET. Le package de distribution du Framework.NET est disponible sous la forme d'un exécutable nommé Dotnetfx.exe. Nous allons voir dans la partie suivante comment redistribuer ce dernier sur les postes non munis du Framework.

2) Mécanismes possibles pour le redéploiement du Framework

Lorsque l'on déploie une application Visual Studio.NET, le Framework doit être installé sur la machine cible. Le Framework ne peut pas être inclus dans le package .msi sous forme de module de fusion (merge module). Il doit être installé séparément du fichier .msi à l'aide de Dotnetfx.exe.

Il y a trois possibilités lorsque l'on distribue une application dépendante du Framework :

. Demander à l'utilisateur de lancer Dotnetfx.exe sur sa machine (par un fichier readme.txt par exemple).

- Pour les administrateurs, dans les entreprises, utiliser un outil de distribution de logiciels type SMS. Ceci permet en effet de redistribuer le Framework.NET une fois pour tout e sur tous les postes de l'entreprise.
- Utiliser un programme d'amorçage fourni par Microsoft (« setup.exe bootstrapper sample ») qui va lancer une installation silencieuse de Dotnetfx.exe puis lancer l'installation de votre package .msi.

La première possibilité est certainement la plus simple à mettre en œuvre. Sachant que si le Framework.NET n'est pas installé, l'application ne s'installera pas, le fichier readme.txt aura un poids plus fort.

La deuxième solution est certainement la meilleure pour les grandes entreprises bénéficiant des outils d'administration de parc machine adéquats.

La troisième possibilité ravira les personnes qui veulent retrouver un déploiement installant tous les fichiers (Framework, package etc,…) à l'aide d'un setup (même si dans la réalité, les divers constituants sont juste installés les uns après les autres).

Le programme d'amorçage effectue les opérations suivantes :

- Vérification de la présence du Framework.NET sur le poste cible.
- Lancement d'une installation silencieuse de Dotnetfx.exe si la version spécifiée du Framework n'est pas présente, et, si nécessaire, installation ou mise à jour de Windows Installer en version 2.0. Un redémarrage de la machine peut être nécessaire à ce moment là.
- Installation de votre application. Si un redémarrage est nécessaire, il est reporté après la fin de l'installation de votre application.

Vous pourrez télécharger ce programme d'amorçage « Framework setup.exe bootstrapper sample » à l'adresse suivante :
http://www.microsoft.com/downloads/details.aspx?FamilyId=BF253CFD-1EFC-4FC5-BA7E-6A6F21403495&displaylang=en

Pour l'utiliser, il suffit de télécharger les fichiers compilés et de suivre les instructions fournies. Un fois téléchargés, vous aurez :

- un fichier settings.ini qui vous permettra de paramétrer votre installation,
- un fichier setup.exe qui lancera l'installation et fera les actions décrites précédemment.

Si vous voulez en savoir plus, les sources de ce programme sont fournies.

Nous allons montrer un tel déploiement plus loin dans ce document (cf. Mise en pratique).

3) Quelques points à éclaircir

a) Installer une version localisée du Framework.NET

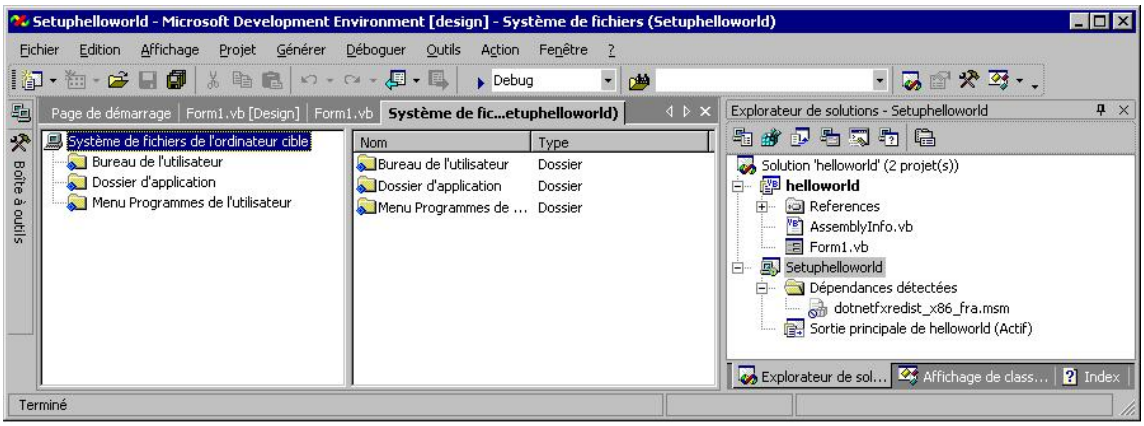
Dotnetfx.exe a été localisé en différentes langues. Le Framework.NET s'installe dans toutes les langues sur les systèmes d'exploitation Windows Me, Windows 2000, Windows NT 4.0, Windows XP (et les Windows .NET à venir). On peut très bien mettre un Framework français sur une machine US et inversement. D'ailleurs, on peut mettre plusieurs Framework dans des langues différentes sur les plateformes citées plus haut. Par contre, comme toute règle a ses exceptions, sur un système Windows 98, il vous faudra installer la version du Framework.NET correspondant à la langue du Windows 98 installé sur la machine. Par exemple, il vous faudra installer la version française de Dotnetfx.exe sur un Windows 98 français.

Cette limitation concerne seulement Windows 98.

Les versions localisées du Framework.NET sont téléchargeables à partir du site
http://msdn.microsoft.com/downloads

b) Comportement de Visual Studio.NET

Lorsque l'on ajoute un projet de configuration afin de déployer une application, Visual Studio.NET ajoute dans les dépendances du projet un module de fusion (merge module) appelé dotnetfxredist_x86_enu.msm.



Par défaut, ce module de fusion est exclu du fichier .msi. En effet, il ne peut pas être redistribué. Il existe pour un usage interne du projet, afin d'empêcher que chaque assembly du Framework.NET utilisée par l'application ne soit listée dans les dépendances du projet.

Si vous essayez d'inclure dotnetfxredist_x86_enu.msm à votre package msi, vous recevrez une er reur de compilation :

ERREUR : dotnetfxredist_x86_enu.msm ne doit pas être utilisé pour redistribuer le .NET Framework. Veuillez exclure ce module de fusion.

Il vous faudra donc toujours exclure ce module de fusion.

Par contre, s'il est exclu, vous obtiendrez quand même un Avertissement :
AVERTISSEMENT : Ce programme d'installation ne contient pas le .NET Framework qui doit être installé sur l'ordinateur cible en exécutant Dotnetfx.exe avant cette installation. Le fichier Dotnetfx.exe se trouve sur le média Visual Studio .NET 'Windows Components Update'. Dotnetfx.exe peut être redistribué avec votre programme d'installation.

Cet avertissement vous indique que le moyen approprié de redéployer le Framework.NET est d'utiliser Dotnetfx.exe.

IV) Mise en pratique

Nous allons maintenant appliquer ce que nous avons vu à des cas simples et concrets :

- Redistribution d'une application .NET pure
- Redistribution d'une application d'accès aux données avec ADO classique.

Le premier exemple va nous permettre de regarder comment redéployer le Framework .NET avec notre application. Le deuxième montrera comment redéployer MDAC avec notre package msi.

1) Exemple de redistribution simple

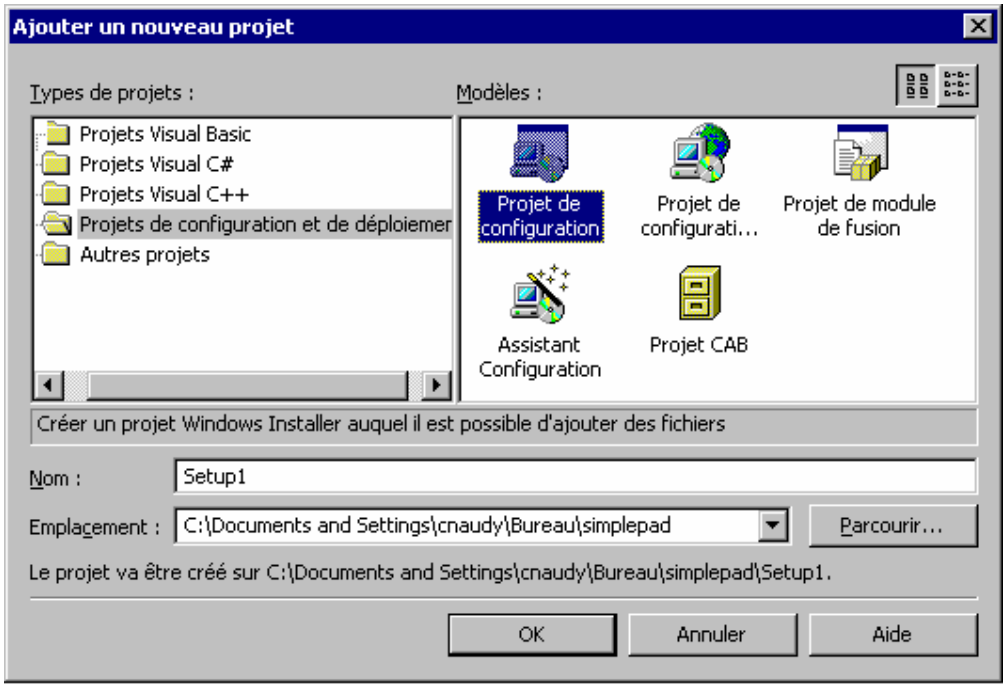
Pour fixer les idées, rien de tel que de créer son premier package de déploiement.

Nous allons créer un projet de configuration pour une application écrite avec le SDK du Framework .NET.

Cette application se nomme « SimplePad » et vous la trouverez dans le dossier FrameworkSDK sous l'arborescence suivante :
Samples\QuickStart\winforms\samples\printing\simplepad\cs si vous avez installé les exemples du Framework SDK.

Ouvrez la solution SimplePad.sln. Cette application est un Bloc -notes permettant d'afficher, de créer, d'imprimer des fichiers texte.

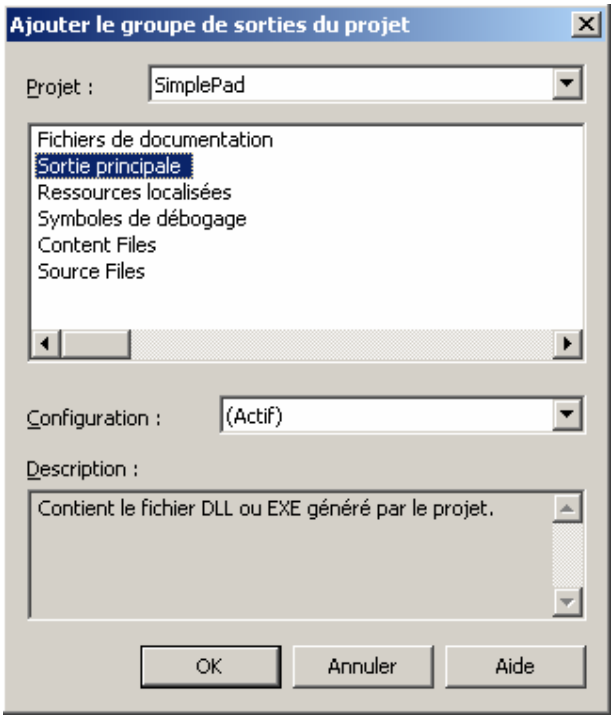
La première chose à faire est d'ajouter un projet de configuration à cette solution.
Pour ceci, allez dans Menu « Fichier », « Ajouter un projet » puis « Nouveau projet ».



On a alors le choix entre les différents types de projet de déploiement discutés dans la première partie.

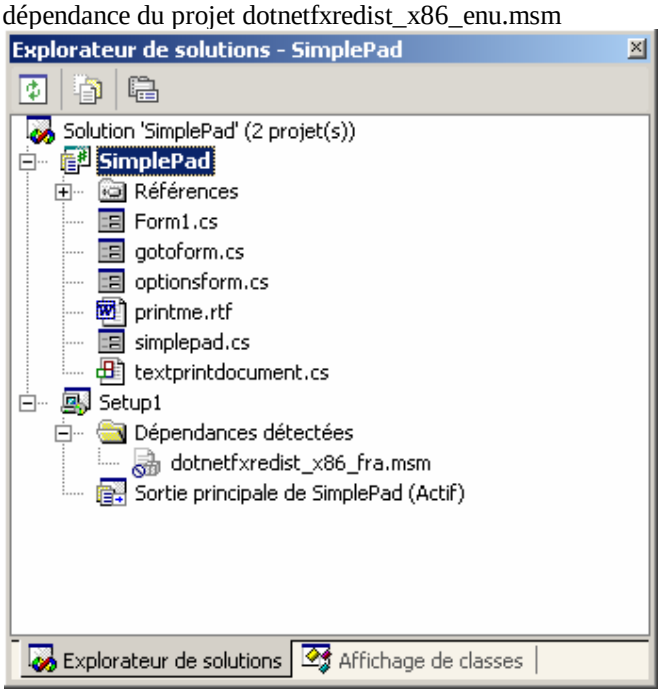
Nous choisissons un « projet de configuration » (Setup Project).
Une fois ce projet ajouté à notre solution, nous allons commencer à le configurer.

La première chose à faire est de sélectionner les « sorties du projet ». Pour cela, cliquez avec le bouton droit sur le projet «Setup1», sélectionnez « Ajouter » puis « sortie du projet ».

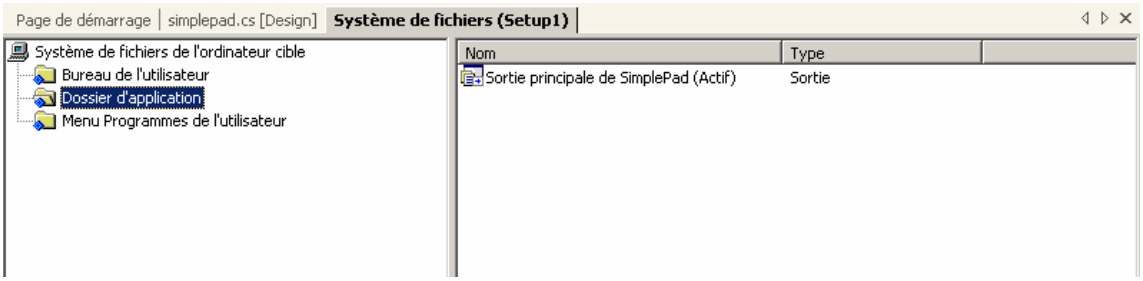


Sélectionnez SimplePad dans la zone de liste déroulante pour « sortie principale » et appuyez sur "OK".

On voit apparaître en dépendance du projet dotnetfxredist_x86_enu.msm



De plus, la sortie principale de l'application apparaît dans le dossier de l'application.



On peut alors créer un raccourci sur le bureau en cliquant sur la sortie principale avec le bouton droit et en sélectionnant « créer un raccourci ». On déplace ensuite ce raccourci dans le dossier Bureau de l'utilisateur
(cf. fiche technique : Q307358 HOW TO: Create Shortcuts for a .NET Deployment Project
http://support.microsoft.com/default.aspx?scid=kb;EN-US;Q307358)

Nous avons ainsi créé un projet de configuration minimal.

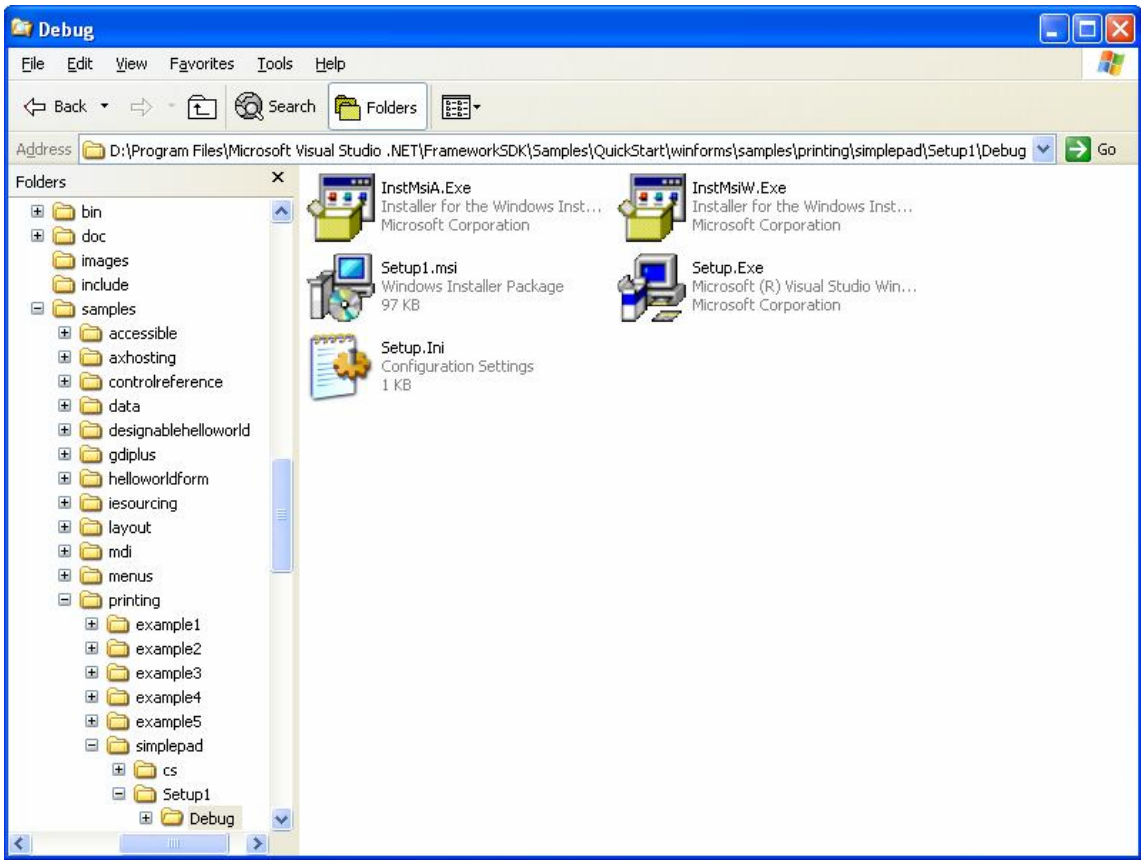
On peut alors générer le package msi (menu « générer » puis « générer setup1 »).

Le premier test que nous allons faire est d'installer ce package sur la machine de développement. Pour cela, cliquer avec le bouton droit sur le projet setup1 dans l'explorateur de solution et choisir « Installer ».
Il est évident qu'un package aussi simple va s'installer sans problème sur la machine de développement qui contient déjà tous les composants nécessaires au fonctionnement de l'application. Par contre, cela peut être utile si l'on veut vérifier que d'éventuels fichiers ajoutés au package sont bien redistribués au bon endroit (dans notre cas, que le raccourci est bien créé sur le bureau).

De même, après avoir installé et vérifié que l'on a bien un raccourci sur le bureau, on peut désinstaller l'application en faisant la même opération et en choisissant « désinstaller ».
Il est aussi possible de choisir de désinstaller à partir de l'icône "ajout et suppression de programmes" situé dans le panneau de configuration.

Maintenant que nous avons vérifié que notre package installe l'application, nous allons essayer de le déployer sur une machine de test.

Si on regarde dans le dossier de Setup1, on a notre package généré sous le dossier Debug.



On voit bien que le fichier msi ainsi que les fichiers d'installation de Windows Installer ont été générés. Ceci est activé, par défaut, dans Visual Studio.NET.

Ce package suffirait si nous voulions déployer sur une plateforme où le Framework .NET est déjà installé. Par contre, s'il est absent, l'application refusera de s'installer car, par défaut, il y a une condition de lancement pour les package msi généré par Visual Studio.NET qui vérifie la présence du Framework et annule l'installation en cas d'absence.

Si nous voulons que le Framework soit installé avant que l'installation du package msi commence, nous pouvons utiliser le projet d'amorçage (Microsoft .NET Framework setup.exe bootstrappe r sample) fourni sur le site Microsoft à l'adresse suivante :
http://www.microsoft.com/downloads/details.aspx? FamilyId=BF253CFD -1EFC-4FC5-BA7E-6A6F21403495&displaylang=en

Cet utilitaire vient en complément du package msi que nous venons de créer.
Son rôle consiste à effectuer une vérification de la présence du Framework .NET. En cas d'absence, il cherchera à lancer le package de re distribution du Framework : Dotnetfx.exe.

Téléchargez la version compilée de ce projet d'amorçage. Elle consiste en deux fichiers :

- Setup.exe : c'est ce fichier qui lancera la vérification de la présence du Framework .NET et qui l'installera en cas d'abse nce.
- Settings.ini : ce fichier sert à configurer l'installation (proposer un partage réseau où trouver Dotnetfx.exe par exemple)

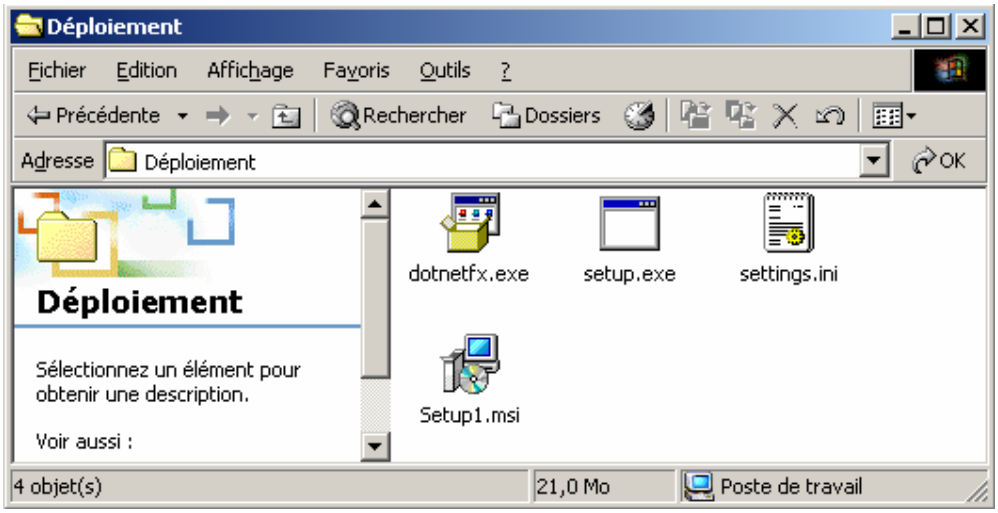
Pour modifier un projet de déploiement afin d'utiliser l'utilitaire d'amorçage, il faut :

. Sélectionner le projet de configurat ion dans l'explorateur de solution
. Cliquer avec le bouton droit afin d'obtenir les propriétés

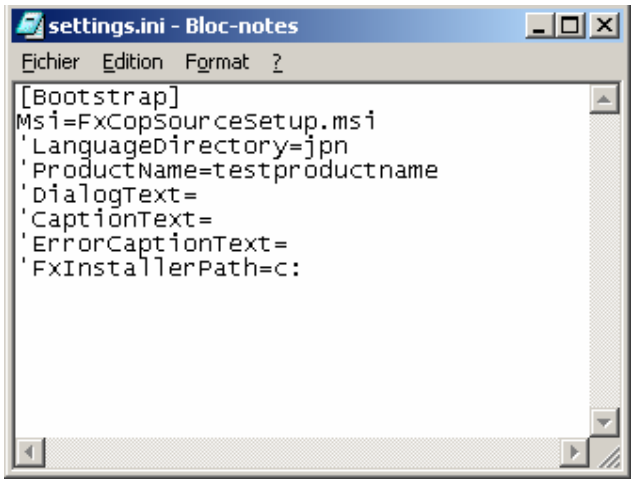
. Dans la page de propriété, mettre la propriété « programme d'amorçage » sur « Aucun »

Après avoir regénéré le projet de configuration, on obtient un seul fichier en sortie : setup1.msi (les différents fichiers pour l'installation de Windows Installer ne sont plus présents).

Le plus simple est alors de copier ce fichier msi dans un dossier de déploiement avec les fichiers setup.exe, settings.ini et Dotnetfx.exe :



Il ne nous reste plus qu'à configurer le fichier settings.ini. Si on l'ouvre avec le bloc -notes, il apparaît sous la forme suivante :



Nous avons à notre disposition 6 paramètres dont 5 sont en commentaire.

Pour notre déploiement, la première chose à faire est de remplacer la ligne
Msi=FxCopSourceSetup.msi
par :
Msi=Setup1.msi

On a également la possibilité de spécifier un partage réseau, Msi=\\monpartage\mondossier\Setup1.msi, ou un sous répertoire, par exemple Msi=mondossier/Setup1.msi.

Il en est de même pour la localisation de Dotnetfx.exe que l'on peut personnaliser à l'aide du paramètre FxInstallerPath. Etant donné que notre Dotnetfx.exe se situe dans le même répertoire que setup.exe et settings.ini, il n'est pas nécessaire d'utiliser ce par amètre.

Sauvegardons donc les modifications apportées au fichier settings.ini.

Il est également possible de demander au projet d'amorçage de vérifier, en plus de la présence du Framework et de sa version, la langue du Framework installé. Ceci est fait gr âce au paramètre LanguageDirectory du fichier settings.ini. Nous ne nous servirons pas ici de cette possibilité.

Les différentes entrées de ce fichier settings.ini sont décrites à l'adresse suivante :
http://msdn.microsoft.com/library/default.asp? url=/library/en -us/dnnetdep/html/redistdeploy.asp
(dans la partie : " Creating the Settings.ini file")

Nous pouvons maintenant tester notre package d e déploiement sur une machine n'ayant pas le Framework.

Il ne faut pas perdre de vue que pour s'installer, le Framework a besoin de certains composants (IE 5.01 minimum, etc…). Avant d'essayer d'installer l'application il est utile de s'assurer que la mac hine cible remplie bien les conditions requises :
http://msdn.microsoft.com/library/default.asp? url=/library/en -us/dnnetdep/html/dotnetfxref.asp

Si ce n'est pas le cas, le programme Dotnetfx.exe remontera une erreur et l'installation ne pourra pas continuer.

Un fois notre application installée et fonctionnant, il est toujours possible de la désinstaller, ainsi que le Framework, à l'aide d'ajout et suppression de programmes dans le panneau de configuration.

2) Exemple de redistribution avec de l'ADO classique

Il est possible avec VB.NET d'utiliser notre bon vieil ADO par interopérabilité. Nous allons maintenant effectuer le déploiement d'une app lication VB.NET simple utilisant ADO classique.

Pour cela, nous allons utiliser un code très simple se connectant à la base exemple SQL server nommée « Pubs ».

- Ajouter une référence à Microsoft ActiveX Data Object Library 2.7 (clic avec le bouton droit sur Référence dans l'explorateur de projet => « ajouter une référence » => choisir Microsoft ActiveX Data Object Library 2.7 dans l'onglet COM
- Ajouter 2 contrôles textbox sur votre formulaire
- Coller les déclarations suivantes dans la section de déclaration générale de votre formulaire :

```
    Dim cn As New ADODB.Connection()
Dim rs As New ADODB.Recordset()
Dim cnStr As String
  Dim cmd As New ADODB.Command()
```

- Coller le code suivant dans l'évènement Form_Load :

```
    cnStr = "Provider=SQLOLEDB;Initial Catalog=Pubs;Data Source=nomserveur;User
    ID=sa;Password=motdepasse;"
cn.Open(cnStr)
```

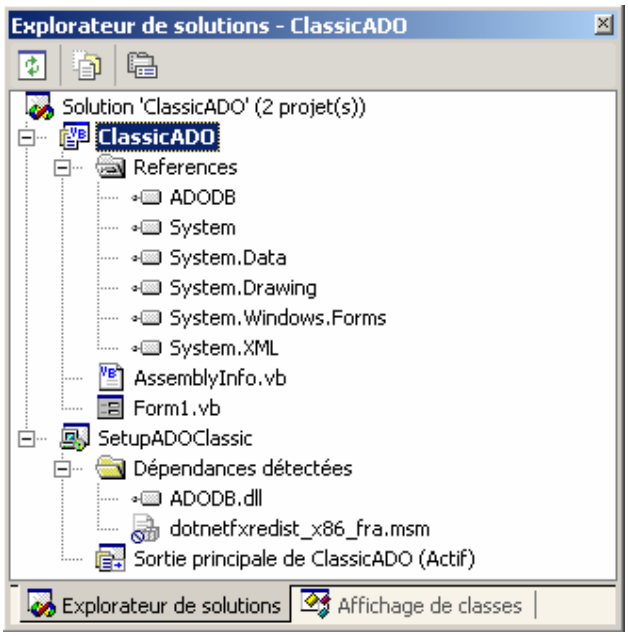
```
rs = cn.Execute("Select * from Authors")

TextBox1.Text = rs.Fields(0).Value.ToString
TextBox2.Text = rs.Fields(1).Value.ToString

rs.Close()
cn.Close()

rs = Nothing
cn = Nothing
```

Lorsque vous lancez l'application, les deux boîtes de texte sont remplies au chargement du formulaire

C'est cette application simpliste que nous allons redistribuer maintenant.

Ajoutons un projet de configuration à notre solution et configurons les sorties du projet.

On voit bien que la « Primary Interop Assembly » ADODB.dll est détectée comme dépendance de notre application :



Bien sûr, étant donné que l'on référence ADO, nous allons devoir redéploy er MDAC sur les machines cibles si la version n'est pas au moins MDAC 2.7.

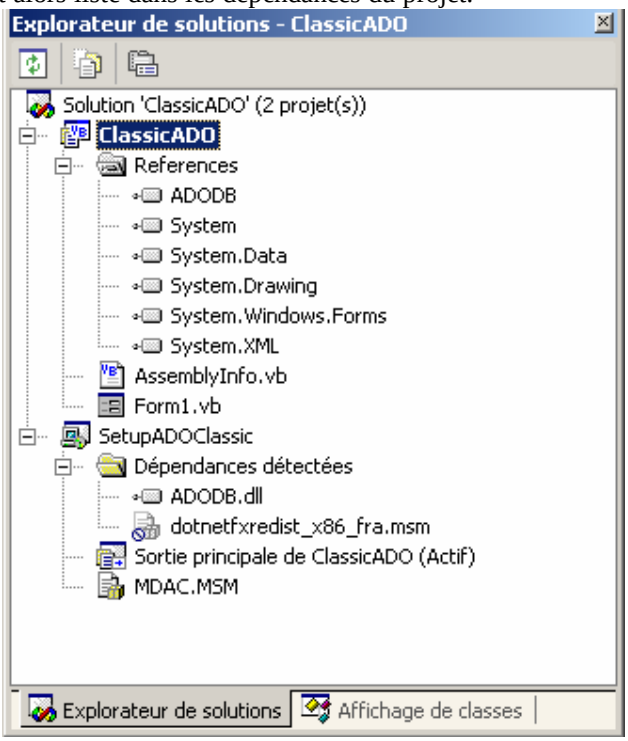
Pour cela, un White Paper est disponible. Il traite de la redistribution de MDAC dans un package Windows Installer. Vous le trouverez ici :
http://support.microsoft.com/default.aspx? scid=kb;FR;q320788
Après avoir téléchargé ce White Paper, vous trouverez un lien à l'intérieur permettant de télécharger le module de fusion permettant de redistribuer MDAC (mdac.msm). Ce module de fusion permet le redéploiement de mdac_typ.exe qui reste la seule manière supportée d'installer MDAC.
Après lecture du White Paper, vous verrez que ce module inclut mdac_typ.exe, ce qui permet de contourner la protection des fichiers systèmes de Windows sous les plateformes Windows 2000, Windows XP et suivantes.

Une fois ce module de fusion téléchargé, il suffit de l'ajouter à notre projet de configuration :

- Clic droit sur le projet de configuration dans l'explorateur de solution
- Selectionner Ajouter

. Choisir module de fusion
. Sélectionner mdac.msm

Le module de fusion est alors listé dans les dépendances du projet.



Il suffit alors de générer l'application et d'installer le package msi sur un poste cible.

Lors de l'installation, une fois l'application installée, le merge module est lancé et installe, en cas de besoin, MDAC à l'aide de mdac_typ.exe.

On peut noter que même si MDAC 2.7 est présent sur la machine, le module de fusion lance mdac_typ.exe. Ceci est sans conséquence car mdac _typ.exe saura gérer la mise à jour et ne fera pas de changements s'il détecte que MDAC est déjà présent dans une version égale ou supérieure.

Ceci nous permet d'utiliser, si on le désire, le programme d'amorçage, permettant de vérifier que le Framework est présent, avec le package msi contenant le module de fusion qui déploie MDAC 2.7.

V. Conclusion

Visual Studio.NET nous offre la possibilité de créer des projets de configuration permettant d'effectuer les tâches de bases de tout programme d'installation. Il utilise la technologie Windows Installer avec tous ses avantages.
Nous ne sommes plus maintenant dépendants des différentes runtimes de nos outils de développement (runtime VB, runtime C, MFC, etc…). La seule chose dont nos applications écrites en code .NET pur ont besoin c'est le Framework.NET. Nous avons vu différentes manières de le redistribuer. L'avantage de celui-ci c'est qu'une fois installé sur une machine, on n'aura plus à se soucier de lui. Pour l'instant, il nous faut encore vérifier sa présence. Ceci est fait, soit par l'utilisateur, soit par le programme d'amorçage dont nous avons parlé plus haut. Ce programme d'amorçage peut également être utilisé dans le cas d'une application d'accès aux données utilisant de l'ADO classique. Il faudra alors compléter notre package de déploiement avec le module de fusion permettant de redéployer MDAC 2.7.

Il ne faut pas oublier qu'à terme, le Framework devrait être présent dans les systèmes d'exploitation Microsoft ce qui facilitera le déploiement des applications .NET.

TRAVAUX PRATIQUES

. 1. Presentation de ClickOnce

Disponible depuis la version 2.0 du framework .NET, **ClickOnce** est la nouvelle technologie de déploiement (et de mise à jour) d'applications.
Elle se base sur le protoc ole HTTP pour effectuer les installations ou les mises à jour.
Le déploiement d'applications, au moyen d'HTTP, est diponible depuis la version 1.0 du framework .NET, mais il a subi de nombreuses modifications qui l'ont rendu plus performant. Pour mettre en place ClickOnce, il vous suffit de posséder:

- Visual Studio 2005 (actuellement en BETA 2) pour développer votre application
- un serveur Web IIS, pour publier votre application

Avec les précédentes versions du framework .NET, exécuter une application pou vait se faire via un simple lien dans une page Web:

Cliquer ici pour télécharger et exécuter le logiciel MonProjet.exe

Bien qu'efficace, cette technique présentait tout de même des inconvénients:

- le framework .NET deva it être installé sur les machines cibles
- l'application est disponible "**Offline**" seulement si l'utilisateur choisit "Travailler en mode déconnecté" dans son navigateur Web
- le fichier **.config** de l'application n'est pas disponible par défaut
- l'application ne possède pas de raccourci sur le bureau ou dans le menu "**Démarrer**"
- etc.

Si vous aviez besoin de plus de contrôle sur le déploiement ou la mise à jour de votre application, vous pouviez utiliser l'"**Updater Application Block –Version 2.0**": il s'agit d'un composant, développé par Microsoft, qui peut détecter, télécharger, et mettre à jour votre application.
Cette technique, très efficace mais possédant malgré tout quelques "failles" (par exemple, le code exécuté possède les droits "Full Trust", ce qui est potentiellement dangereux, d'un point de vue sécurité, etc.) existe toujours (vous pouvez télécharger et avoir plus d'informations sur l'UAB : Voir MSDN) mais laisse sa place, dans la version 2.0 du Framework .NET, à ClickOnce.

Pour faire simple, on peut dire que ClickOnce possède tous les avantages de l'UAB (plus quelques autres), sans les inconvénients.
Voici une liste des caractéristiques (non exhaustive) de ClickOnce:

- les mises à jour sont transactionnelles (c'est -à-dire qu'elles sont faites entièreme nt ou pas du tout)
- bonne intégration avec Visual Studio
- il est possible de proposer le téléchargement de pré -requis, nécessaires au bon fonctionnement de l'application
- il est possible d'avoir un raccourci dans le menu "Démarrer"
- etc.

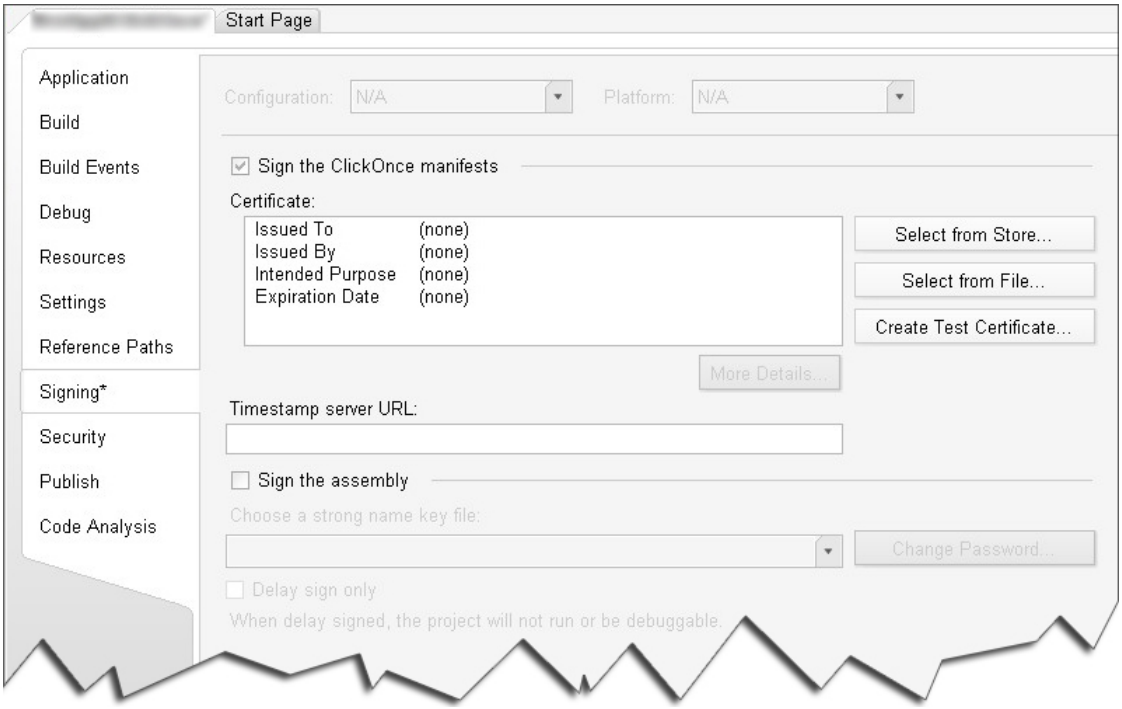
Maintenant que n ous avons vu la théorie, passons un peu à la pratique: rien de tel pour apprendre ;)

. 2. Déploiement d'une application

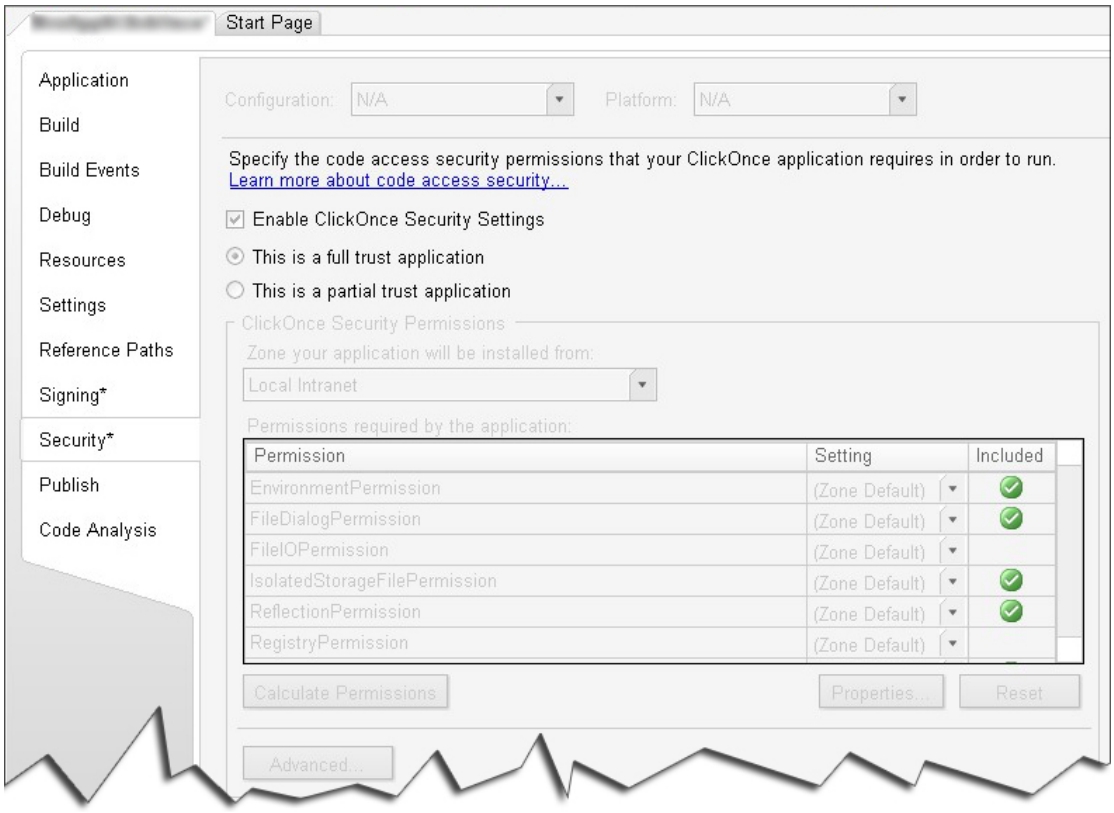
. 2.1. Premier déploiement

Au cours de cette démonstration, et pour que l'explication soit la plus claire possible, nous prendrons un c as pratique réel, plus précisément, nous déploierons une application et ses différentes mises à jour.

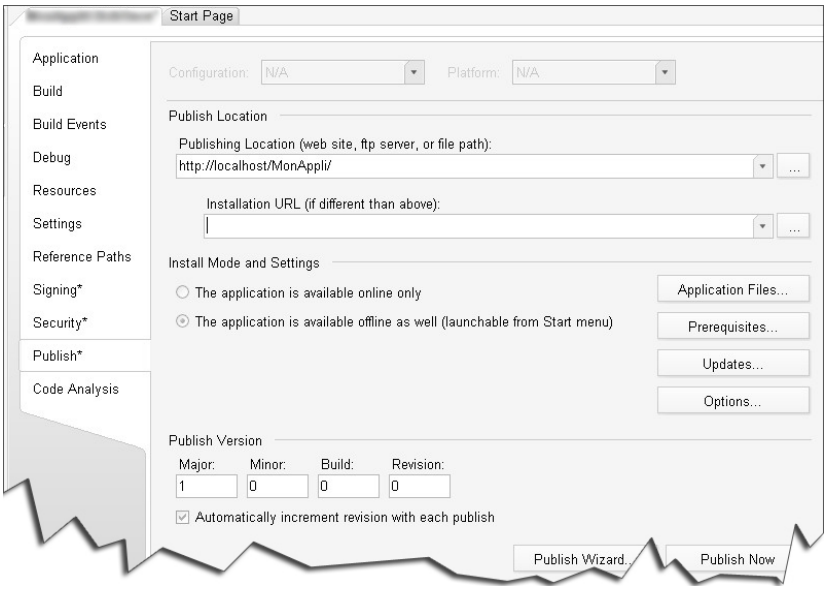
Créer une application Winform basique. Compilez la, pour que nous puissions la déployer. Pour cela, veuillez ouvrir le panneau des propriétés de votre p rojet (menu Projet > nom_du_projet propriétés). Cochez la case et signez le manifeste ClickOnce.



Nous devons maintenant définir les autorisations de séc urité. Pour le moment, cochez la case qui indique que notre application est "full trust" (on peut lui fait totalement confiance). Par mesure de sécurité, il faudra mieux éviter de choisir cette option; nous verrons donc un peu plus tard comment définir au mieux ces options de sécurité.

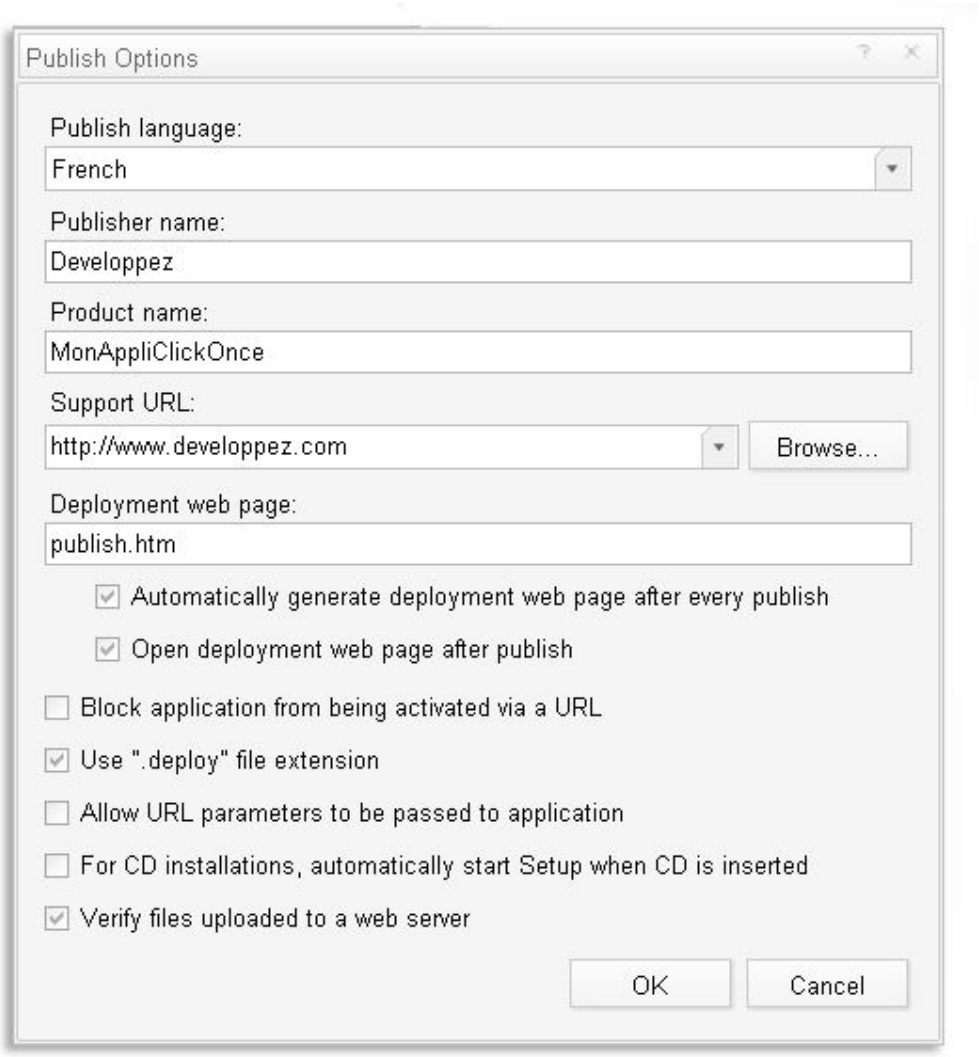


Passons maintenant à la partie la plus importante: la publication. Passons donc dans l'onglet publication, et définissez l'endroit où sera publié votre application ainsi que la page d'installation de votre application.
Sur cette page, vous allez définir si votre application sera publiée sur un ftp, un partage réseau ou un serveur IIS et vous pouvez également définir si l'a pplication ne sera accessible que de façon online (retélechargée à chaque fois) ou de façon offline, téléchargée, installée et accessible via le menu *Démarrer*.



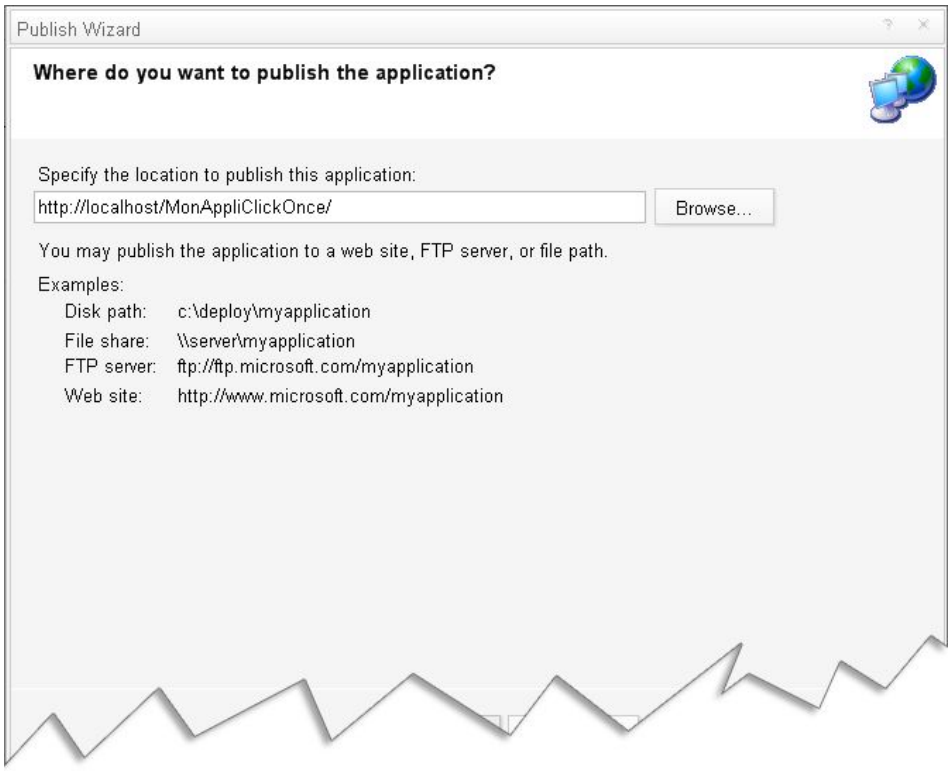
- **Publishing Location** indique le partage Web depuis lequel votre application sera déployé/mise à jour
- **Install Mode and Settings** permet de spécifier si votre application doit être disponible " **OnLine**" et "**OffLine**"
- **Prerequisites** vous permet de spécifi er les programmes pré -requis pour installer votre application

- **Updates** vous permet d'indiquer que votre application doit (ou non) vérifier la présence de mise à jour sur le serveur. Vous indiquez également à quel moment vous souhaitez faire cette vérificat ion (avant ou après le démarrage de l'application)
- Grâce au panneau **Options**, vous pouvez renseigner différentes informations

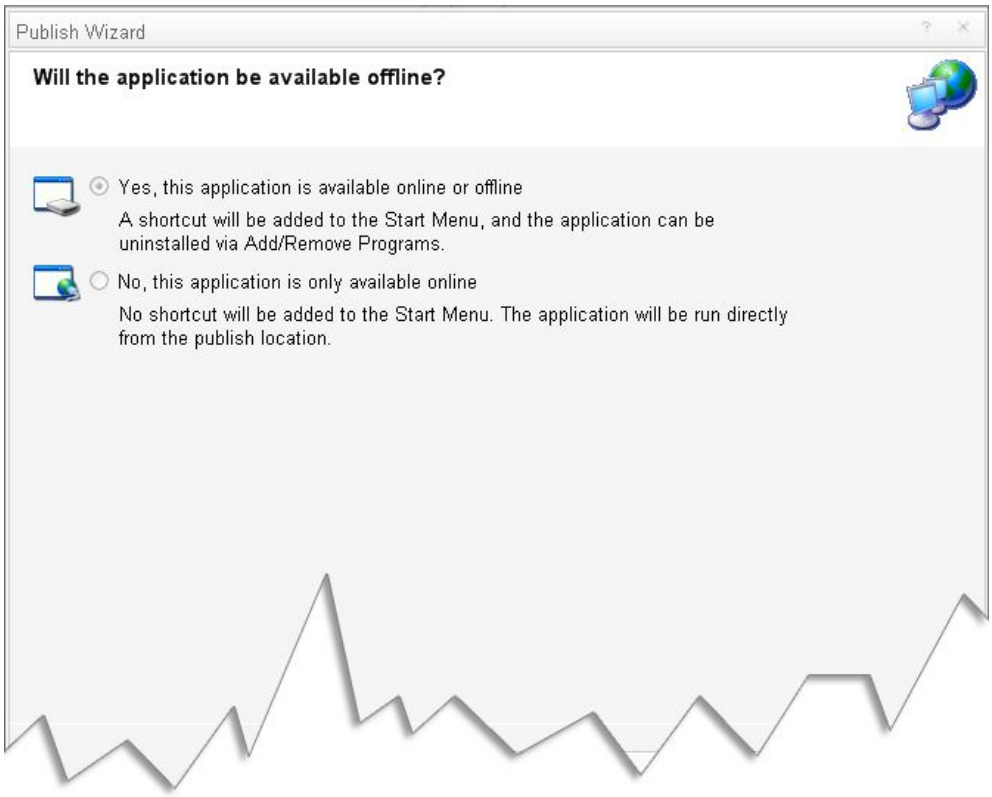


Une fois ces informations r enseignées, un clic sur le bouton "Publish Wizard" (également disponible depuis le menu "Build" ou bien, dans l'explorateur de solution, via un clic droit) lance l'assistant de configuration de l'installeur ClickOnce.

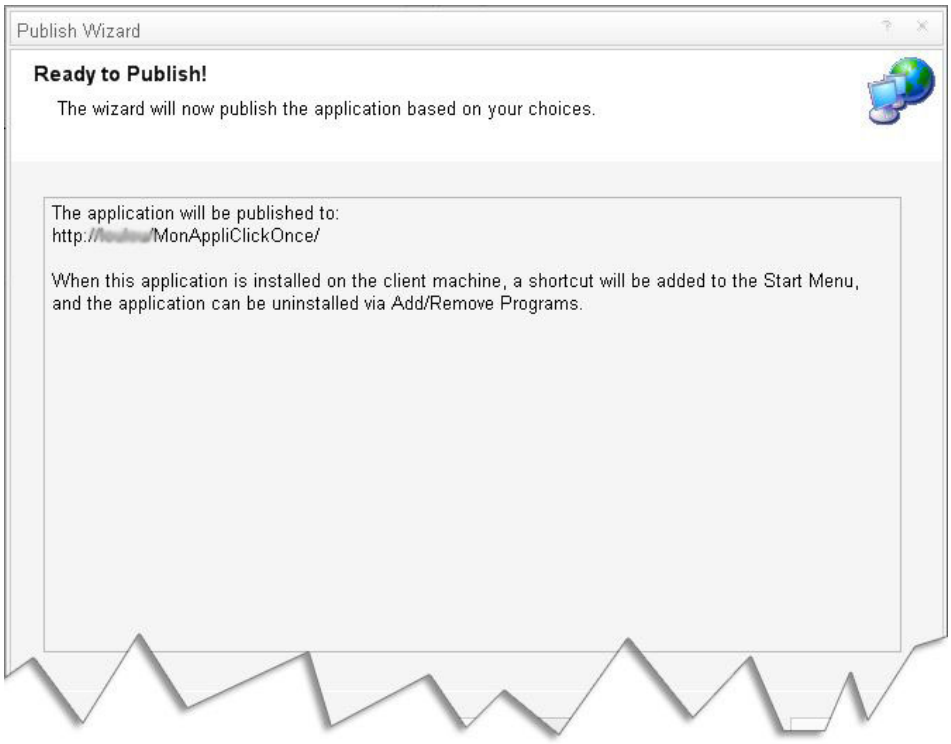
Le "**Publish Wizard**" vous permet de c onfigurer toutes les options nécessaires au déploiement de votre application.



Wizard de publication

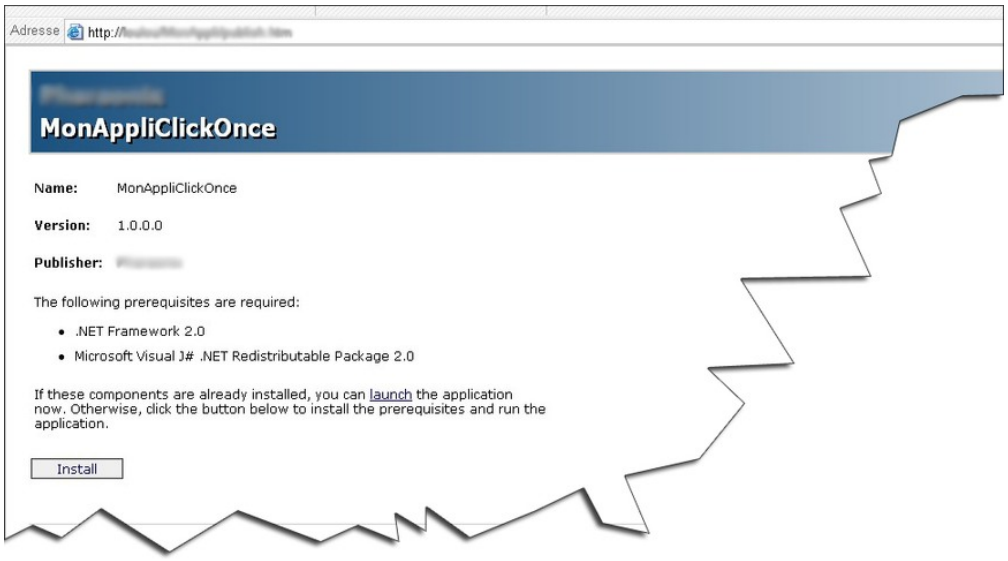


Wizard de publication (2)



Wizard de publication (3)

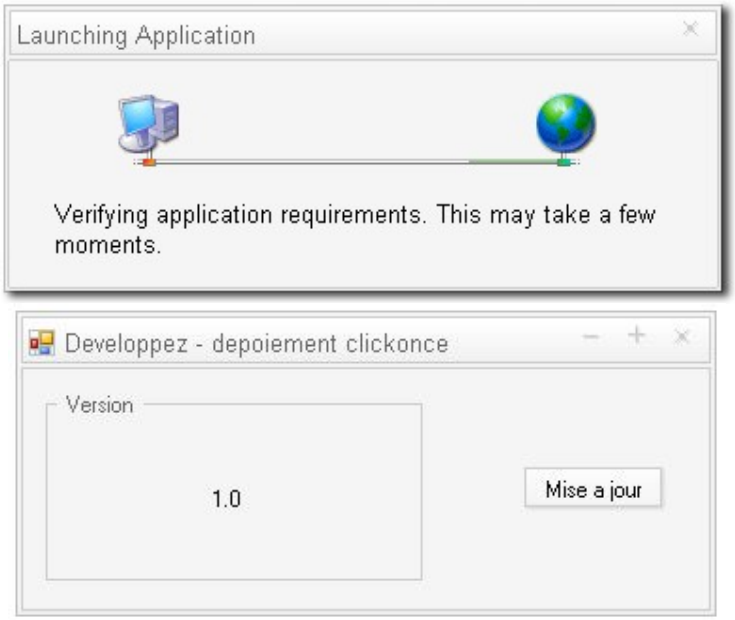
Un clic sur le bouton "**Finish**" vous permet de déployer votre application sur le serveur IIS et vous ouvre la page de publication, qu'il vous suffit d'envoyer à vos utilisateurs pour qu'ils installent votre application:



Application publiée

2.2. Installation

Il ne nous reste plus qu'à installer l'application sur le poste client. Cette installation se fait manuellement. Un clic sur le bouton "**Install**" vous permet de lancer l'installat ion de l'application sur le poste et on aperçoit bien la vérification de la présence de mise à jour, juste avant le lancement:

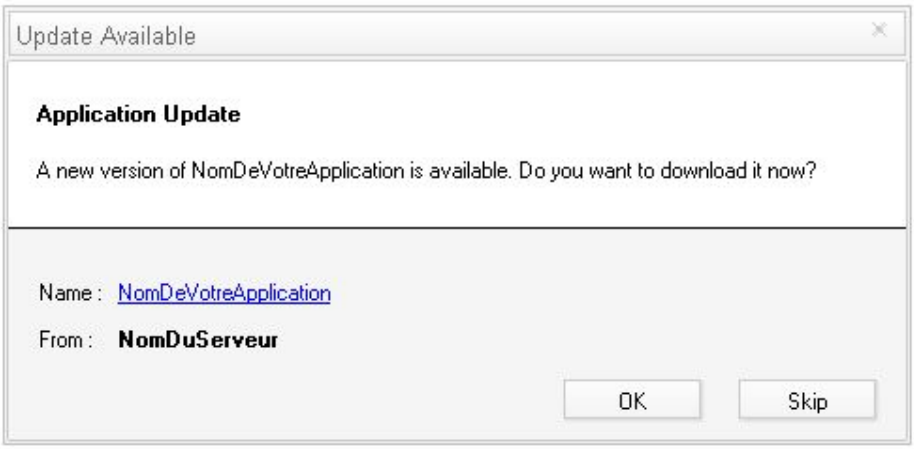


Vérification de mise à jour

6g. 2.3. Mise à jour

6geD. 2.3.1. Automatique

Pour faire une mise à jour de votre application, rien de plus simple: il vous suffit de faire vos modifications et de refaire un "**Publish**", dans Visual Studio.
Ensuite, exécuter votre application et dès le démarrage, celle -ci détectera la présence d'une nouvelle version, la téléchargera et relancera l'application pour prendre en compte les modifications.



Détection de mise à jour

Nouvelle version installée

7geD. 2.3.2. Manuelle

Vous avez également la possibilité de lancer une vérification manuelle de la mise à jour (utilisée généralement dans les menus "Mettre à jour" ou autre) en utilisant le code suivant:

Mise à jour manuelle

```
private void UpdateApp()
{
   ApplicationDeploy ment updater = ApplicationDeployment.CurrentDeployment;
   bool verDepServer = updater.CheckForUpdate();

   if (verDepServer) // Update disponible
   {
      DialogResult res = MessageBox.Show(
                  String.Format( "Une nouvelle version est disponible{0}Mettre à jour maintenant ? " ,
                        Environment.NewLine),
                  "Mise à jour",
                  MessageBoxButtons.YesNo);

      if (res == DialogResult.Yes)
      {
         updater.Update();

         MessageBox.Show( "Attention, l'application va redémarrer" , "Redémarrage",
                                    MessageBoxButtons.OK, MessageBoxIcon.Information);

         Application.Restart();
      }
   }
   else
   {
      MessageBox.Show( "Pas de nouvelle version disponible" );
   }
}
```

Cependant, il semblerait que dans sa version actuelle, le framework .NET 2.0 ait quelques problèmes à ce niveau là, sachez juste que plus tard (comprendre par là, lorsque le framework .NET 2.0 ne se ra plus en version BETA), l'utilisation du code précédant sera possible et ne posera pas de problème.

. 3. Utilisation avancée

8g. 3.1. Signature du manifest

Vous avez la possibilité de signer le manifest de votre application ClickOnce, ainsi que de signer v otre assembly .NET.
Pour rappel, dans la version 2003 de Visual Studio, pour signer une application, vous deviez utiliser l'utilitaire en ligne de commande **sn.exe** qui vous servait à générer un fichier de clé publique/privée (avec une extension **.snk**), et vous utilisiez le fichier **AssemblyInfo.cs** (ou .vb si vous développiez en VB.NET) pour ajouter un lien vers ce fichier .snk.

Sous Visual Studio 2005, le procédé est le même, hormis que vous n'avez plus besoin de passer par l'utilitaire en ligne de commande: dans l'onglet "**Signing**" de la page de propriétés de votre projet, cochez la case " **Sign the assembly**" et, via le menu déroulant, créer un nouveau fichier snk.

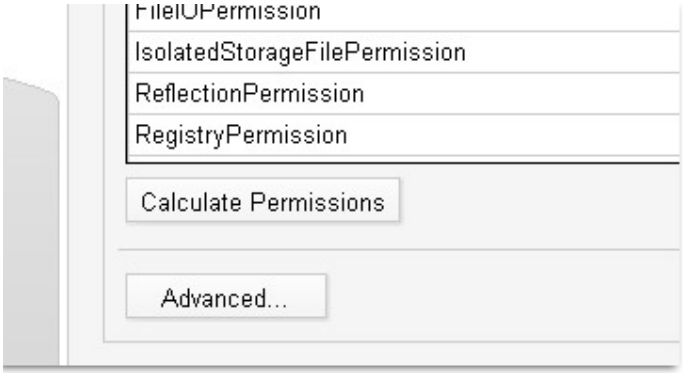
Rappelons que signer les assemblies .NET est une technique efficace et très pratique, dans le cas où vous seriez amené à les réutiliser dans d'autres projets, car cela vous permet de les inscrire dans la **GAC** (*Global Assembly Cache*) et vous évite, ainsi, de devoir redéployer la DLL de votre assembly à chaque fois que vous déployer un nouveau projet.

Vous avez également la possibilité de signer le manifest de votre application ClickOnce, en utilisant un certificat de sécurité.
Dans l'onglet "**Signing**", cocher la case "**Sign the ClickOnce manifests** ", puis choisissez le certificat que vous voulez utiliser. Vous pouvez:

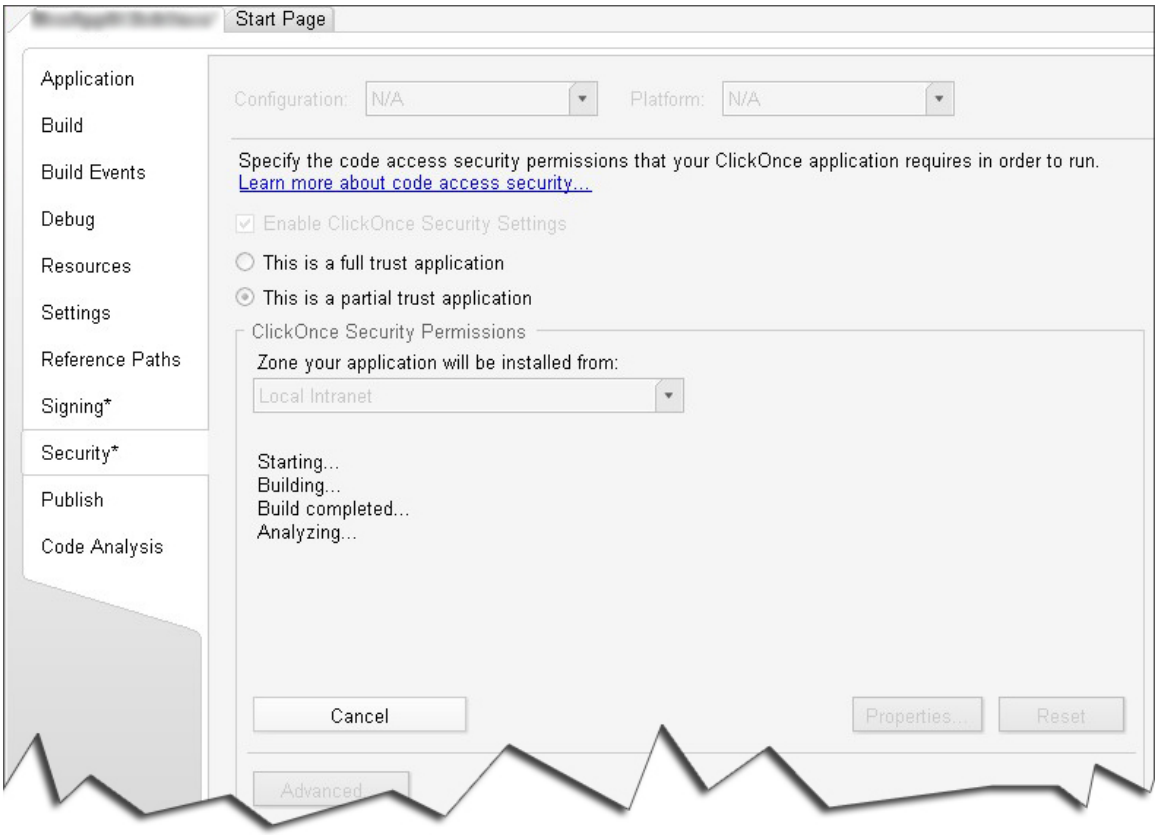
- utiliser un certificat déjà installé (bouton " **Select from Store** "),
- importer votre certificat (bouton " **Select from File** ")

) . 3.2. Définition des stratégies de sécurité

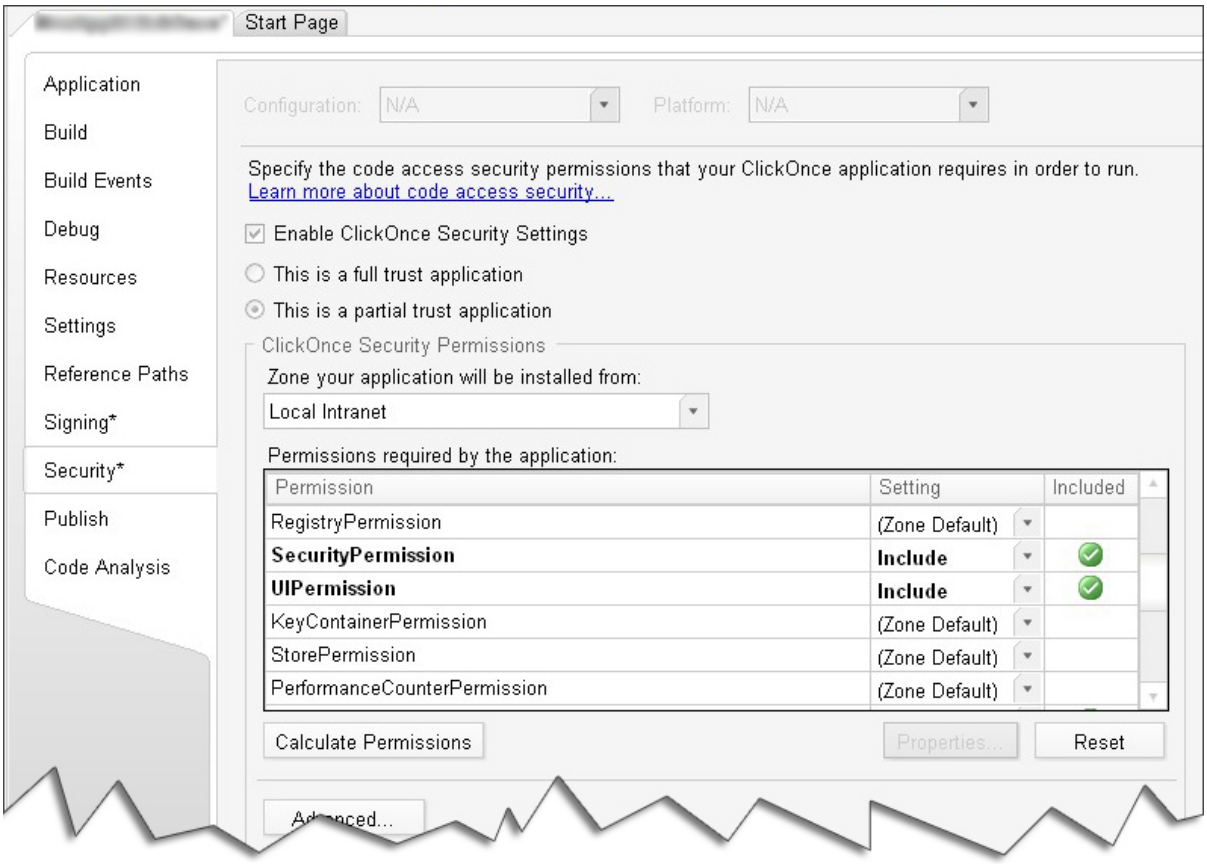
Vous devez savoir qu'il ne faut jamais sous -estimer la sécurité, surtout dans un environnement professionel. Il faut garder à l'esprit que l'application est souvent la passerelle qui permet à "l'attaquant" d'effectuer des actio ns non autorisées, c'est pourquoi il faut sécuriser l'application avec par exemple, "le principe du moindre privilège". Ainsi, une bonne sécurité est celle qui donne le moins de libertés possibles, plus précisément, celle qui donne simplement les droits né cessaires. Vous pouvez donc le faire manuellement en choisissant un après un les droits que vous accordez à votre application ou alors, vous pouvez laisser Visual Studio faire cela pour vous.
Cliquez alors sur "Calculer les permissions":



Visual Studio va alors analyser chaque ligne de votre code afin de détecter (en fonction de vos méthodes) quelles seront les autorisations à effectuer:



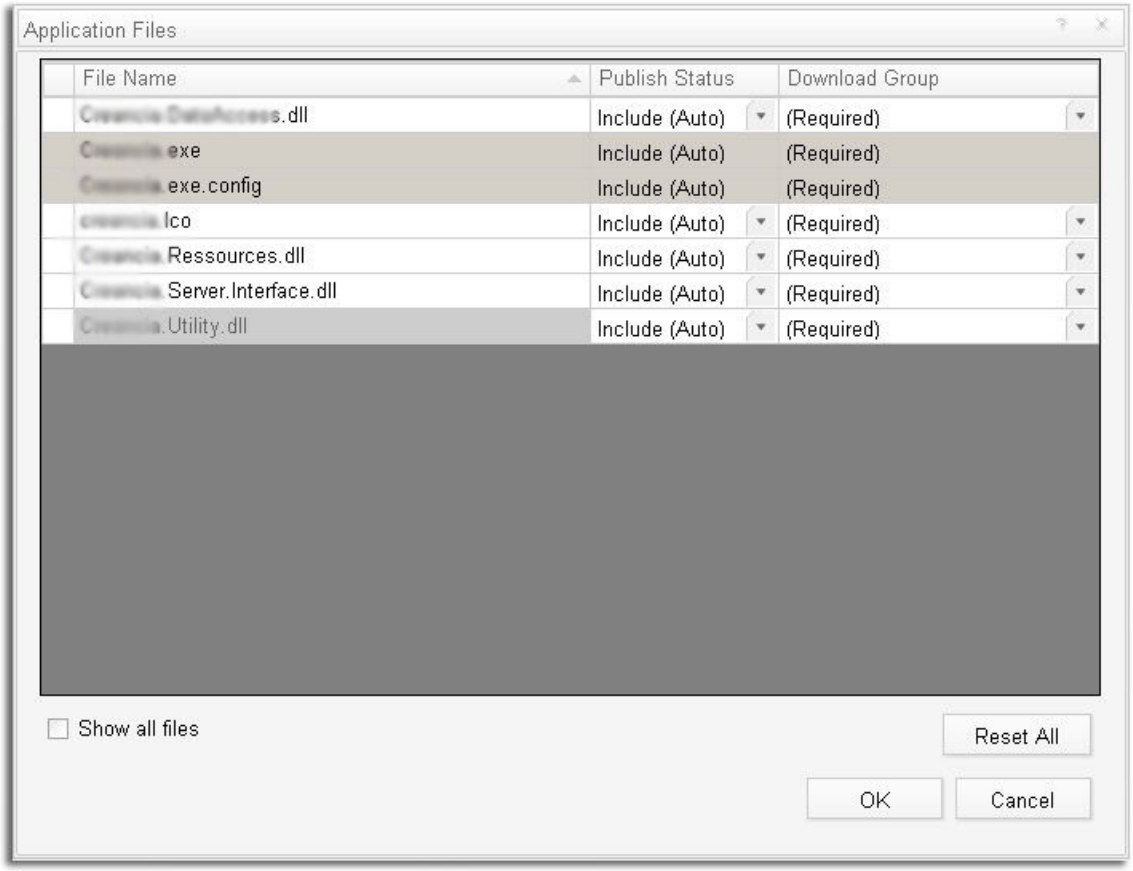
A la fin du scan, seules les autorisations nécessaires seront cochées et mises en gras. Vous pouvez ainsi ajouter des droits ou alors passer outre les conseils de Visual Studio.



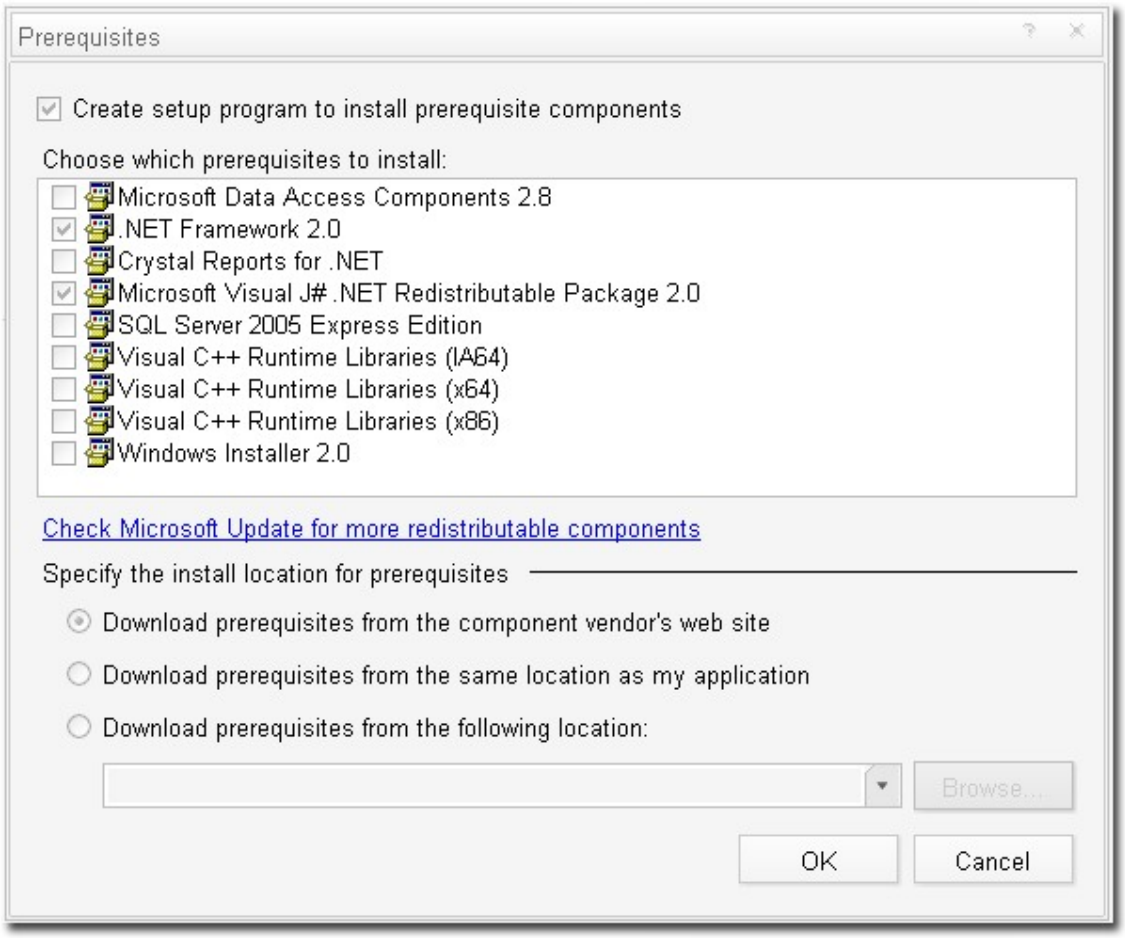
0g. 3.3. Déploiement avancé

Même si le déploiement via ClickOnce peut trouver certaines limitations, il possède néanmoins plusieurs fonctionnalités accessibles via de nouvelles fenêtres qui se trouvent sur l'onglet Publication des propriétés du projet.

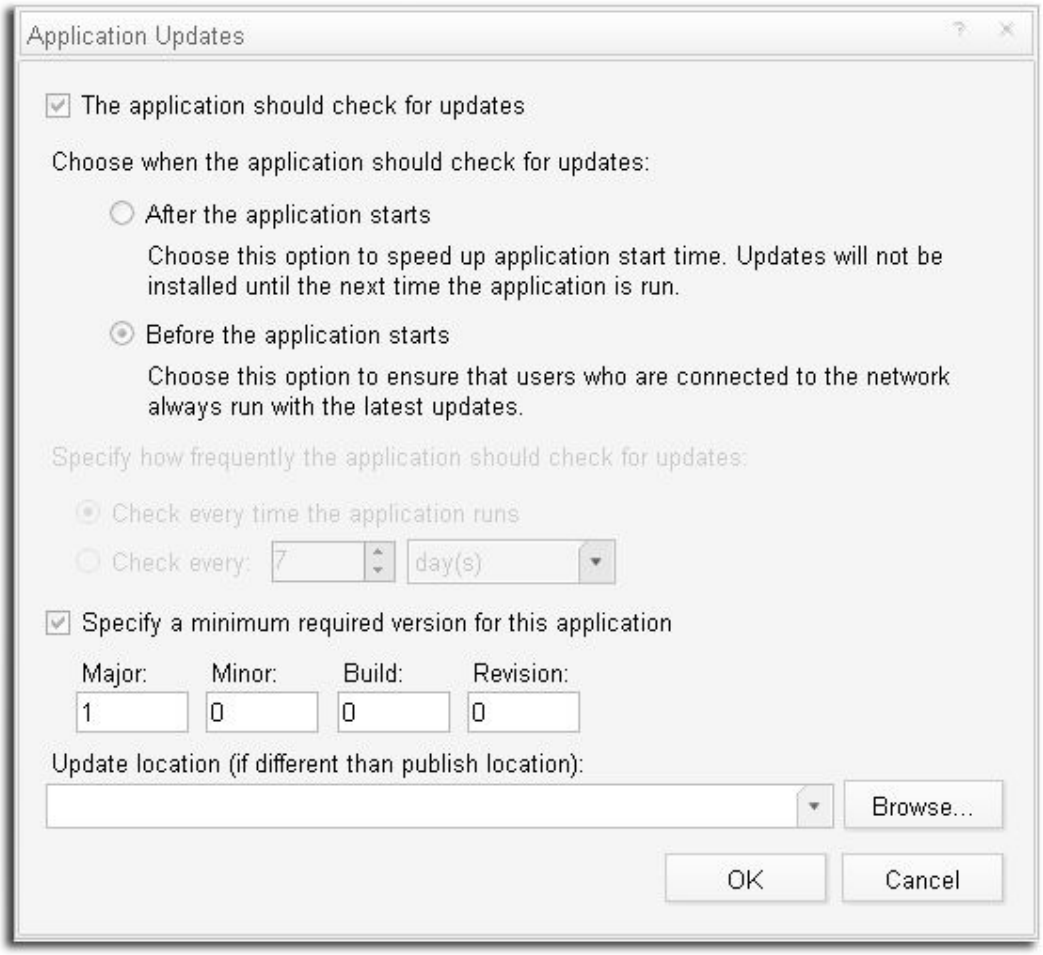
La seconde de ces fonctionnalités est de définir sur le projet, quels fichiers seront publiés puis déployer. En effet, votre projet peut contenir d ifférents fichiers ressources qui ne sont pas nécessaires à l'éxecution de l'application et ne necessitent donc pas d'être déployées.



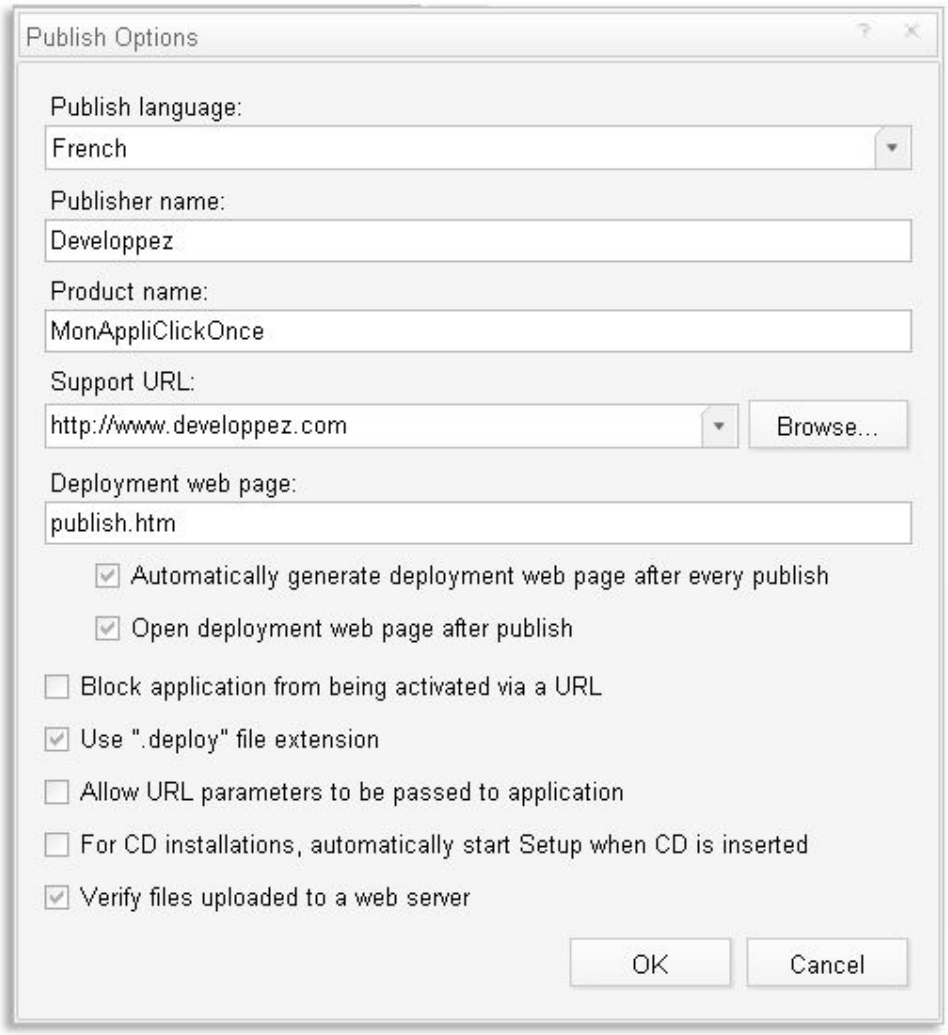
La seconde de ces fonctionnalités est la possibilité de définir des conditions d'installation. Vous pouvez par exemple exiger que certaines composants comme le framework 2.0 ou encore SQL Serveur soient installés sur l'ordinateur client, comme le monde la capture suivant e:



Une autre des possibilités est de définir le type de mise à jour de l'application déployée.
Vous pouvez par exemple définir si les mises à jour s on critiques et l'utilisateur doit absolument utiliser cette nouvelle dès qu'elle est téléchargée ou alors, telecharger la version en arrière plan et elle sera chargée à la prochaine utilisation.



Enfin, le dernier panneau d'options, permet de personnaliser la page de publication, allant du nom fichier, à l'url du support technique, en passant par la personnalisation du créateur de l'application.



4. Conclusion

ClickOnce est la nouvelle technologie de déploiement et de mise à jour qui possède de beaux jours devant elle: son utilisation simple et très agréab le nous permet d'affirmer que ce sera sans doute la technologie qui remplacera, à terme, la technologie des MSI.

Références :

Walkthrough: Deploying a Windows Application :
http://msdn.microsoft.com/library/default.asp? url=/library/en -us/vsintro7/html/vbtskCreatingInstallerForYourApplication.asp
Deployment Articles :
http://msdn.microsoft.com/library/default.asp? url=/library/en -us/dv_vstechart/html/vstchDeployment.asp
Deploying .NET Applications: Lifecycle Guide :
http://msdn.microsoft.com/library/default.asp? url=/library/en -us/dnbda/html/DALGRoadmap.asp
Microsoft .NET Framework Setup.exe Bootstrapper Sample :
http://www.microsoft.com/downloads/details.aspx? FamilyId=BF253CFD -1EFC-4FC5-BA7E-6A6F21403495&displaylang=en
Microsoft .NET Framework Redistributable :
http://www.microsoft.com/downloads/details.aspx? FamilyID=d7158dee -a83f-4e21-b05a-009d06457787&displaylang=en