Créer un modèle F# pour AutoCAD avec Visual Studio Express

L'objectif de ce tutoriel est de montrer comment créer un modèle de démarrage d'un nouveau projet F# pour AutoCAD dans Visual Studio Express, modèle permettant de lancer automatiquement AutoCAD en chargeant la DLL depuis Visual Studio en mode Debug.

L'exemple montré utilise Visual Studio Express 2013 pour Windows Desktop qui est la première version Express à supporter F#. Toutefois, F# n'est pas installé d'origine, il faut ajouter le module Visual F# Tools 3.1.1 téléchargeable à cette adresse : http://fsharp.org/use/windows/

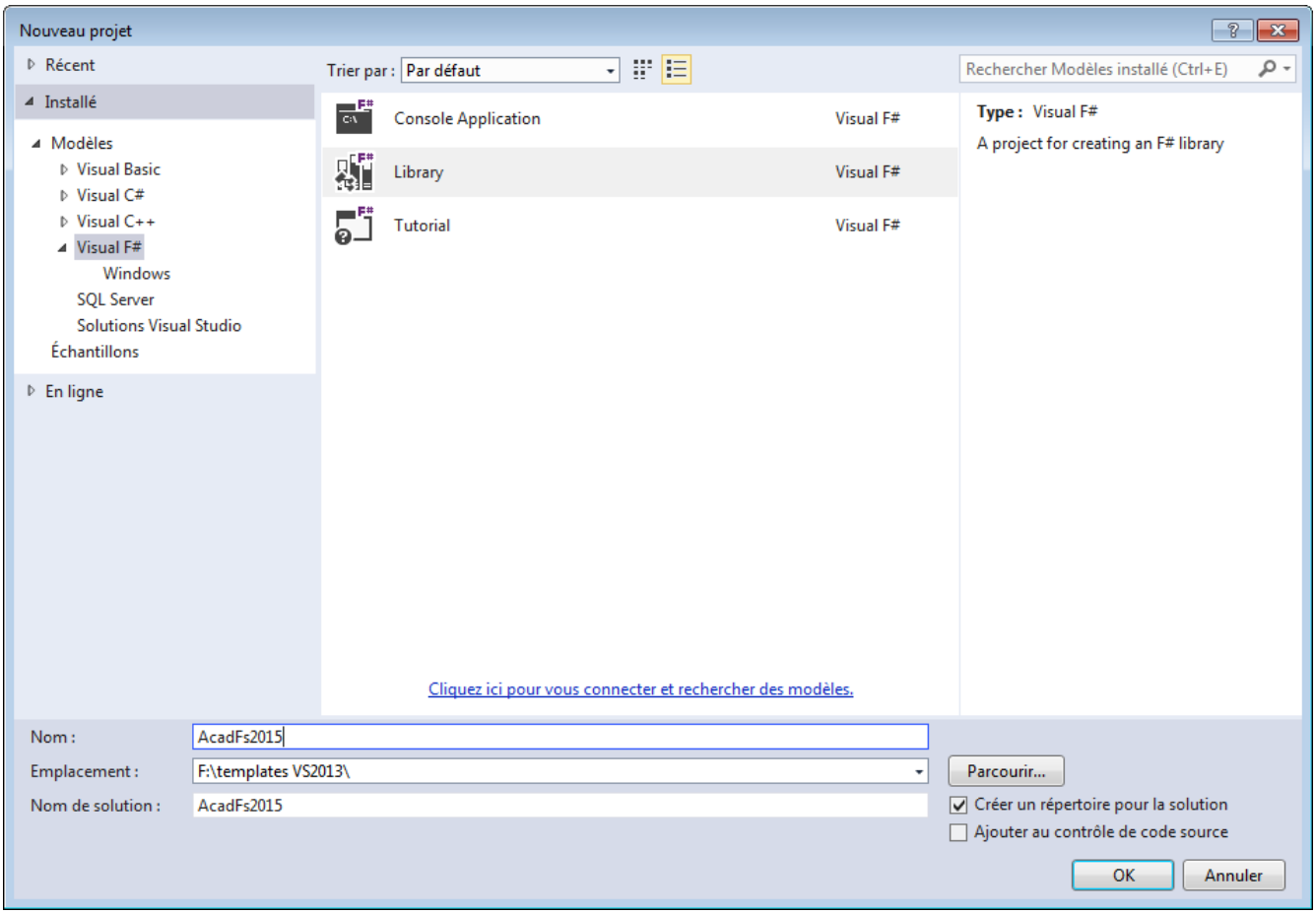
Il y sera créé un modèle pour AutoCAD 2015 mais là encore il est possible de transposer pour cibler d'autres versions d'AutoCAD.

Le chemin du répertoire de sortie pour le mode Debug utilisé dans l'exemple est celui par défaut (.\bin\debug).

Démarrer un nouveau projet

Dans Visual Studio, choisir le langage *Visual F#* et le type de projet : *Library* (Bibliothèque de classe).

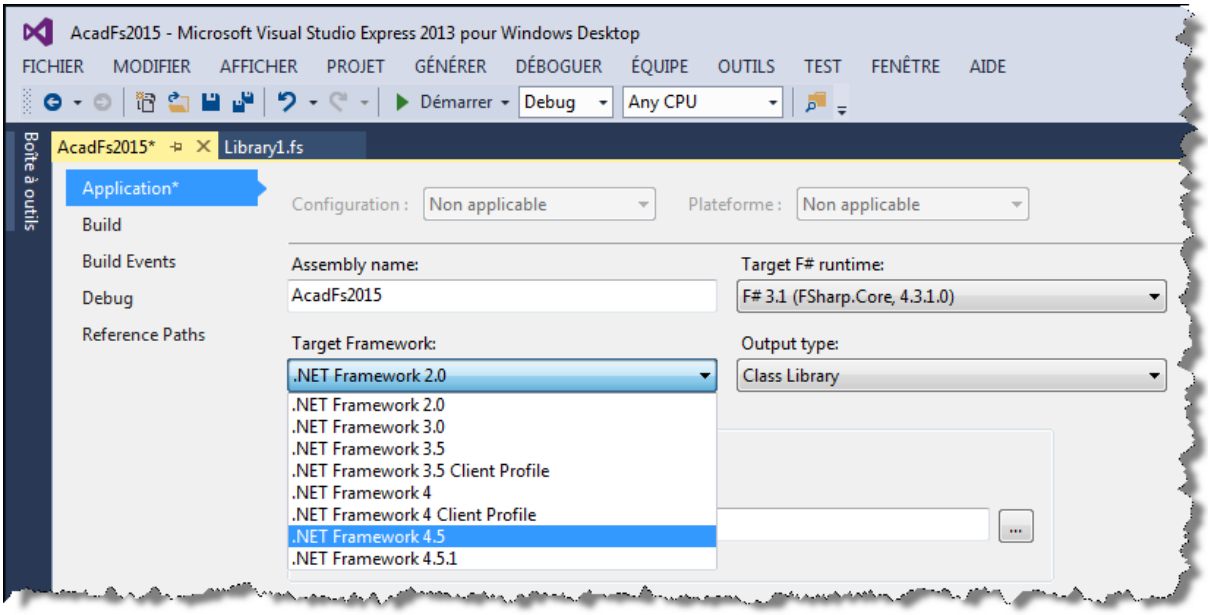
Renommer le projet : AcadFs2015 par exemple, et changer le chemin de l'emplacement comme désiré.



Spécifier le Framework ciblé

Ouvrir la fenêtre propriété du projet : dans l'explorateur de solutions, sélectionner le projet AcadFs2015, puis clic droit et *Propriétés* ou cliquer sur l'icône en forme de clé plate.

Dans la fenêtre de *Propriétés*, onglet *Application*, choisir le Framework cible dans la liste déroulante.



Il est préférable de cibler la version du Framework correspondant à celle installée par la version d'AutoCAD pour laquelle le modèle est créé :

- AutoCAD 2010 et 2011 : .NET Framework 3.5
- AutoCAD 2012 à 2014 : .NET Framework 4.0
- AutoCAD 2015 : .NET Framework 4.5

Ajouter les bibliothèques AutoCAD

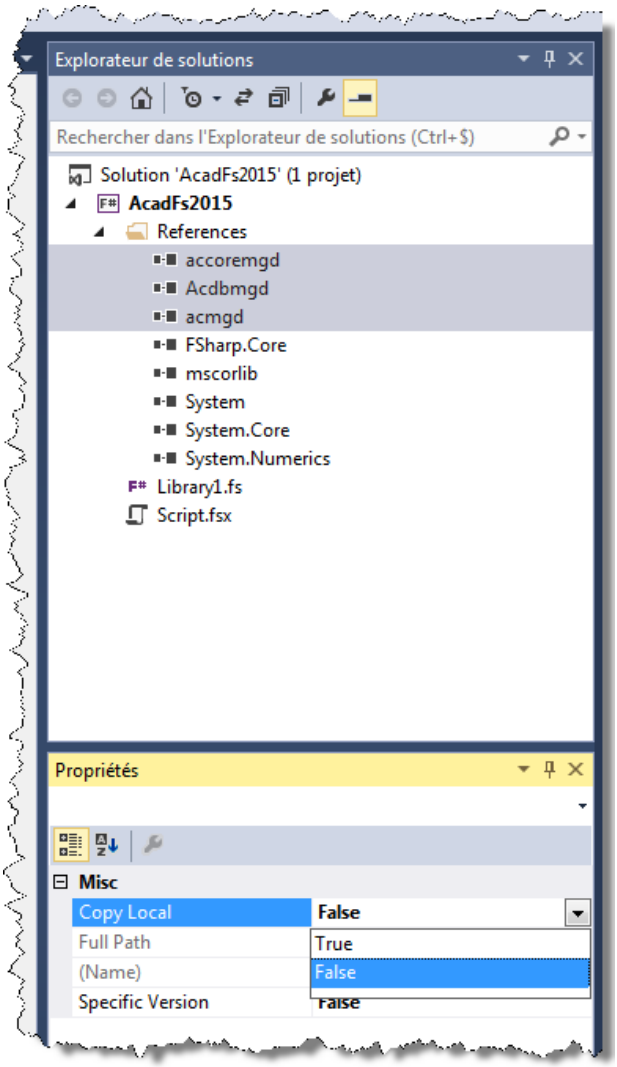
Dans l'explorateur de solutions, sélectionner *Références* puis clic droit et *Ajouter une référence...*
Dans la boite de dialogue *Reference Manager*, cliquer sur *Parcourir...*

Les références à ajouter se trouvent dans le répertoire d'installation de la version ciblée d'AutoCAD ou, mieux, dans le dossier ObjectARX20##\inc correspondant à la version d'AutoCAD ciblée (inc-win32 ou inc-x64 suivant les versions d'AutoCAD et la plateforme du poste utilisé).
On peut télécharger les dernières versions d'ObjectARX sur cette page :
http://usa.autodesk.com/adsk/servlet/index?siteID=123112&id=773204

Choisir les bibliothèques les plus couramment utilisées (il sera facile d'en ajouter d'autres de la même manière si la création du nouveau projet le requiert).

- AutoCAD 2010 à 2012 : AcDbMgd.dll et AcMgd.dll
- AutoCAD 2013 à 2015 : AcCoreMgd.dll, AcDbMgd.dll et AcMgd.dll

Mettre la propriété *Copy Local* de ces DLLs sur *False* :



Ajouter une ébauche de code

Renommer la classe créée par Visual Studio : *Library1* avec un nom plus explicite, *Commands* par exemple, depuis l'explorateur de solution.

Ouvrir la classe *Commands* dans l'éditeur de code par double clic sur *Commands* dans l'explorateur de solution.

Supprimer l'ébauche de code créée par Visual Studio.
Ajouter un nom de module, optionnellement précédé d'un espace de nom.
Ajouter les instructions *open* pour importer les espaces de nom AutoCAD les plus couramment utilisés et, éventuellement, un alias pour la classe Autodesk.AutoCAD.ApplicationServices.Application.

Ajouter quelques lignes de codes usuelles pour définir une commande AutoCAD :

```
open System
open Autodesk.AutoCAD.ApplicationServices
open Autodesk.AutoCAD.DatabaseServices
open Autodesk.AutoCAD.EditorInput
open Autodesk.AutoCAD.Geometry
open Autodesk.AutoCAD.Runtime

type AcAp = Autodesk.AutoCAD.ApplicationServices.Application

let getObject<'T when 'T :> DBObject> (oid : ObjectId) =
    oid.GetObject(OpenMode.ForRead) :?> 'T

let getObjects<'T when 'T :> DBObject> source =
    let rxc = RXClass.GetClass typeof<'T>
    source
    |> Seq.cast<ObjectId>
    |> Seq.filter (fun id -> id.ObjectClass.IsDerivedFrom rxc)
    |> Seq.map getObject<'T>

[<CommandMethod("Test")>]
let test() =
    let doc = AcAp.DocumentManager.MdiActiveDocument
    let db = doc.Database
    let ed = doc.Editor

    use tr = db.TransactionManager.StartTransaction()

    tr.Commit()
```

Générer la solution pour contrôler l'absence d'erreur (F6 ou F7).

Ajouter un script pour charger l'application au démarrage d'AutoCAD

Depuis l'explorateur de solutions, faire un clic droit sur le projet AcadCs2013 puis *Ajouter* et *Nouvel élément...* (Ctrl+Maj+A).

Dans la boite dialogue *Ajouter un nouvel élément*, choisir : *Text File*. Renommer le fichier : start.scr et cliquer sur *Ajouter*.

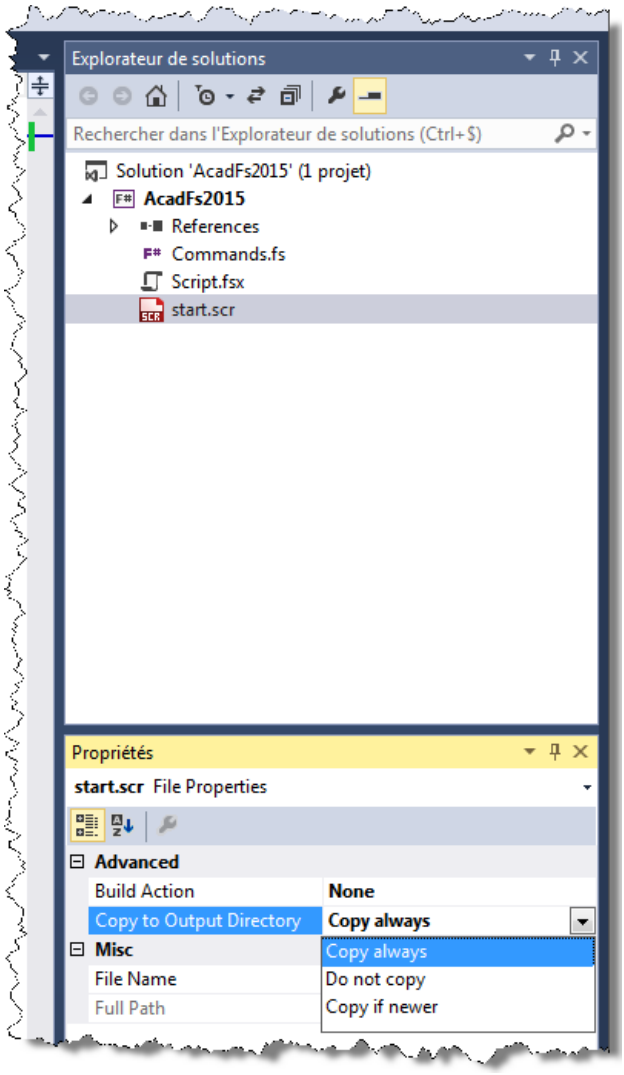
Le fichier s'ouvre dans la fenêtre d'édition, ajouter l'instruction suivante sans oublier une espace à la fin.

```
netload "..\..\bin\debug\AcadFs2015.dll"
```

Si le répertoire de sortie pour le mode Debug a été modifié, changer le chemin de la DLL en conséquence.

Enregistrer le fichier start.scr.

Dans les propriétés de start.scr (sélectionner start.scr dans l'explorateur de solutions), mettre la propriété *Copy to Output Directory* sur *Copy always*.



Modifier le fichier de projet MSBuild pour lancer AutoCAD au débogage

Les fichiers de projet MSBuild (.fsproj) sont des fichiers xml qui décrivent et contrôlent le processus de génération des applications.
C'est dans ce fichier que doivent être ajoutées les instructions qui permettront de lancer AutoCAD depuis Visual Studio en mode débogage et de charger la DLL au démarrage d'AutoCAD.

Générer la solution et fermer Visual Studio pour pouvoir modifier le fichier AcadFs2015.fsproj.

Ouvrir le fichier .\AcadFs2015\AcadFs2015\AcadFs2015.fsproj avec un éditeur de texte (bloc-note, notepad++, etc.).

Ajouter les nœuds suivants à la fin du nœud *PropertyGroup* correspondant à la génération en mode Debug (le deuxième nœud *PropertyGroup* du fichier), après avoir modifié, si nécessaire, le chemin du fichier acad.exe correspondant à la version d'AutoCAD ciblée et celui du fichier start.scr (le répertoire de sortie pour le mode Debug).

```
<StartAction>Program</StartAction>
<StartProgram>C:\Program Files\Autodesk\AutoCAD 2015\acad.exe</StartProgram>
<StartArguments>/nologo /b "..\..\bin\debug\start.scr"</StartArguments>
```

StartAction indique que la génération du projet doit lancer un programme (ce qui n'est généralement pas le cas avec une DLL).
StartProgram spécifie le programme à démarrer, ici AutoCAD.
StartArguments contient les arguments qui vont lancer le script qui chargera la DLL.

Le nœud *PropertyGroup* devrait ressembler à ça :

```
<PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Debug|AnyCPU' ">
  <DebugSymbols>true</DebugSymbols>
  <DebugType>full</DebugType>
  <Optimize>false</Optimize>
  <Tailcalls>false</Tailcalls>
  <OutputPath>bin\Debug\</OutputPath>
  <DefineConstants>DEBUG;TRACE</DefineConstants>
  <WarningLevel>3</WarningLevel>
  <StartAction>Program</StartAction>
  <StartProgram>C:\Program Files\Autodesk\AutoCAD 2013\acad.exe</StartProgram>
  <StartArguments>/nologo /b "..\..\bin\debug\start.scr"</StartArguments>
  <DocumentationFile>bin\Debug\AcadFs2015.XML</DocumentationFile>
</PropertyGroup>
```

Dans le nœud *ItemGroup* se trouvent les références ajoutées au projet.

Si le répertoire de la solution est sur le même lecteur que celui dans lequel se trouvent les DLLs référencées (le répertoire d'installation d'AutoCAD ou celui d'ObjectARX 20##), les chemins enregistrés sont relatifs.
Comme le modèle sera exporté dans un répertoire différent de celui de la solution, il faut remplacer ces chemins relatifs par les chemins absolus.

Par exemple, remplacer les chemins relatifs suivant :

```
<ReferenceInclude="AcCoreMgd">
  <HintPath>..\..\..\..\..\..\..\ObjectARX 2015\inc\AcCoreMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
<ReferenceInclude="AcDbMgd">
  <HintPath>..\..\..\..\..\..\..\ObjectARX 2015\inc\AcDbMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
<ReferenceInclude="AcMgd">
  <HintPath>..\..\..\..\..\..\..\ObjectARX 2015\inc\AcMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
```

Par ceux-ci :

```
<ReferenceInclude="AcCoreMgd">
  <HintPath>C:\ObjectARX 2015\inc\AcCoreMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
<ReferenceInclude="AcDbMgd">
  <HintPath>C:\ObjectARX 2015\inc\AcDbMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
<ReferenceInclude="AcMgd">
  <HintPath>C:\ObjectARX 2015\inc\AcMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
```

Enregister le fichier AcadFs2015.fsproj.

Exporter le modèle

Ré-ouvrir la solution AcadFs2015.sln dans Visual Studio.
Lancer un débogage (F5) pour contrôler le bon fonctionnement. AutoCAD devrait s'ouvrir et afficher en ligne de commande :

```
Commande: netload Nom du fichier d'assemblage: "..\..\bin\debug\AcadFs2015.dll"
```

Le message d'erreur ci-dessous indique soit un nom de DLL erroné dans le fichier start.scr ; soit que le fichier start.scr n'a pas été copié dans le répertoire de sortie (CF propriété *Copier dans le répertoire de sortie*).

```
Commande: netload Nom du fichier d'assemblage: "..\..\bin\debug\AcadFs2015.dll"
Impossible de charger l'assemblage. Détails de l'erreur:
System.IO.FileNotFoundException: Impossible de charger le fichier ou
l'assembly...
```

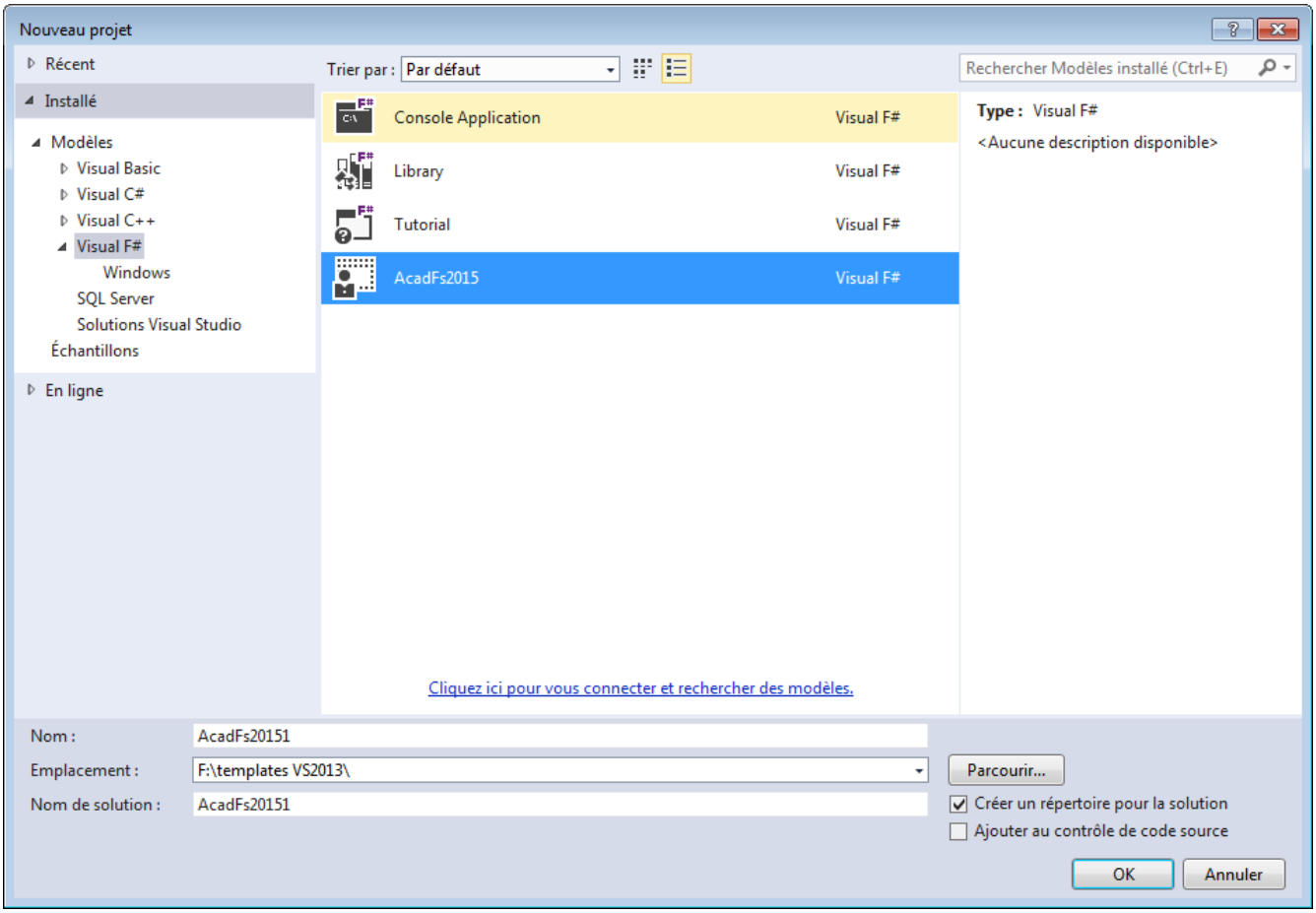
Si tout fonctionne correctement, il est temps de générer le modèle.

Dans le menu *FICHIER*, choisir Exporter le modèle...
Dans la boite de dialogue *Assistant Exportation de modèle*, laisser coché *Modèle de projet* et faire *Suivant*.
Changer le nom du modèle comme désiré, *Acad2015FsCommand*, par exemple et ajouter éventuellement une description *Modèle pour commande AutoCAD 2015*.
Terminer.

Avec les options par défaut, le modèle est exporté sous forme de fichier ZIP dans les dossiers :
Visual Studio 2013\Templates\ProjectTemplates
et
Visual Studio 2013\My Exported Templates
du répertoire Documents ou Mes documents.

Ce modèle sera désormais proposé par Visual Studio au démarrage d'un nouveau projet.

Penser à renommer le projet et la solution ainsi que le nom de la DLL dans le fichier start.scr du nouveau projet.



Conclusion

La même procédure peut être utilisée pour créer d'autres modèles, ciblant d'autres versions d'AutoCAD et/ou du Framework, ou encore pour d'autres types de projets (fonctions LISP, bibliothèques de classe, extensions d'application, ...).

Il est possible de rajouter des instructions dans le fichier start.scr comme le lancement d'une commande.

Il est aussi possible de spécifier un fichier DWG à ouvrir dans la balise *StartProgram* du fichier .fsproj avec un chemin complet ou relatif (la racine étant le répertoire de sortie).
Exemple avec un dessin Test.dwg dans le répertoire de la solution :

```
<StartProgram>
  C:\Program Files\Autodesk\AutoCAD 2015\acad.exe "..\..\..\Test.dwg"
</StartProgram>
```