Présentation générale: Upsizing pour SQL Server

Auteur: Doug Hennig

Date: Le 3 avril, 2006

Table des matières

© Microsoft confidentiel

1 Présentation et cadre

1.1 Introduction

L'Assistant Upsizing fourni avec Foxpro Visual nécessite une mise à jour significative pour corriger quelques déficiences et fournir de nouvelles fonctionnalités.

1.2 Arrière plan, besoins et justification

Il y a plusieurs insuffisances dans l'Assistant Upsizing, comme:

- Les champs Date et DateTime vides ne sont pas gérés correctement parce que le concept d'une DateTime vide n'existe pas sur SQL Server ni sur les autres bases de données.
- Dans certaines conditions, le chargement des tables migrées avec les données peut être assez lent.
- Les champs dont les noms sont des mots SQL réservés sont renommés lors de leur migration (par exemple, un champ nommé "Order" est renommée "Order_"). Cela peut être une source de problèmes avec les programmes préexistants, rapports, formulaires, et autres.

Plusieurs nouvelles caractéristiques sont prévues:

- L'assistant pourra fonctionner comme un moteur sans interface utilisateur. Cela permettrait aux développeurs d'automatiser programmatiquement des tâches d'upsizing plutôt que d'avoir à passer manuellement par les étapes de l'interface de l'assistant.
- Le nouveau moteur pourra être appelé depuis l'Explorateur des Données. Idéalement, l'utilisateur devrait pouvoir cliquer droit sur une base de données VFP, répondre à quelques questions minimes (le serveur et nom de la base de données, par exemple) et avoir le moteur de l'assistant migration qui crée un Serveur SQL ou une autre base de données avec les options les plus courantes.
- Il supportera spécifiquement Serveur SQL 2005 et Serveur SQL 2005 Express.
- Il aura des crochets (des hooks) de façon à être amélioré par les développeurs qui souhaitent personnaliser son comportement à différentes étapes du processus de l'upsizing.
- Des événements seront implémentés à plusieurs étapes dans le process d'upsizing, pour permettre à un composant de l'interface utilisateur d'utiliser BINDEVENT () pour afficher la progression.

1.3 Objectifs

Les objectifs de la mise à jour de l'Assistant Upsizing pour Sedna sont les suivants:

- Réutiliser le maximum de l'assistant upsizing actuel en l'améliorant dans le sens des autres objectifs.
- Supprimer dans le moteur toute dépendance aux éléments de l'interface visuelle, y compris aux contrôles du formulaire de l'assistant et aux boîtes de dialogue.
- Faire en sorte qu'on puisse l'appeler depuis différentes origines, y compris l'Explorateur des Données.
- Définir des champs Date et DateTime dans la table cible qui acceptent les valeurs nulles si les champs Date et DateTime de la table source ont des valeurs vides (blank), même si les champs source n'acceptent pas de valeurs nulles, et charger des valeurs nulles depuis la table source au lieu de valeurs vides (blank).
- Améliorer la performance du chargement des données là où on le peut.

© Microsoft confidentiel

- Gérer les champs nommé avec des mots SQL réservés en utilisant des délimiteurs de nom (par exemple," [Order]").
- Supporter toutes les nouvelles caractéristiques de SQL Server 2005 et SQL Server 2005 Express.
- Ajouter le support pour une extension objet d'une façon similaire à l'aperçu dans VFP 9. Cela permet à un développeur d'étendre ou même de modifier le comportement du moteur à plusieurs étapes du processus de l'upsizing.
- Implémenter des événements à plusieurs étapes du processus de l'upsizing.

1.4 Dépendances / considérations

Pour que l'upsize puisse être appelé depuis l'Explorateur des Données, il faudra que l'application de l'Explorateur des Données soit actualisée pour appeler le moteur de l'upsizing sur demande de l'utilisateur.

2 Scénarios

2.1 Migration d'une base de données VFP depuis l'Explorateur de Données en utilisant l'assistant

Bob veut migrer une base de données VFP vers SQL Server. Il affiche son Centre d'Informations (Task Pane), et choisit l'Explorateur de Données. Il clique sur le bouton "Ajouter une Connexion" et sélectionne sa base de données VFP. Quand sa base de données figure dans l'Explorateur de Données, il clique droit dessus et choisit l'Assistant Migration (Upsizing) dans le menu contextuel. Il aurait pu également faire glisser sa base de données VFP (par un drag-and-drop) sur le serveur SQL Server 2005 visible dans l'Explorateur de Données.

L'Assistant Upsizing apparaît. Bob suit les différentes étapes de l'assistant, en sélectionnant les options qu'il souhaite utiliser (certaines sont préétablies, comme le choix de la base de données VFP à migrer, en se basant sur le contexte de l'Explorateur des Données). Il choisit ensuite "Finir" pour migrer sa base de données; une fois le processus terminé, l'Explorateur des Données est actualisé pour montrer la base de données migrée.

Ce scénario est vraiment seulement un raccourci pour lancer l'Assistant Upsizing depuis le menu Outils, Assistants.

2.2 Migration d'une base de données VFP depuis l'Explorateur de Données en utilisant le moteur

Jane veut migrer une base de données VFP vers SQL Server. Elle affiche son Centre d'Informations (Task Pane), et choisit l'Explorateur de Données. Elle clique sur le bouton "Ajouter une Connexion" et sélectionne sa base de données VFP. Quand sa base de données figure dans l'Explorateur de Données, elle clique droit dessus et choisit "Migrer la Base de Données" dans le menu contextuel.

On lui demande alors de sélectionner la Base de Données à migrer ; elle peut choisir une connexion dans la base de données, une DSN, ou spécifier une chaîne de connexion. Maintenant elle spécifie si elle veut créer une nouvelle base de données ou si elle veut migrer vers une base déjà existante. Dans le premier cas, elle saisit le nom de la nouvelle base de données ; dans le deuxième cas, elle choisit la base de données de destination dans une liste des bases de données existantes. Elle clique sur le bouton "Migrer" pour commencer le processus. Une barre de progression lui montre l'avancement de la migration. Quand le processus est terminé, la boite de dialogue se ferme et l'Explorateur des Données est actualisé pour montrer la base de données migrée.

© Microsoft confidentiel

2.3 Migration d'une base de données VFP par programme

Mary a plusieurs bases de données VFP qu'elle souhaite migrer vers SQL Server. Comme elles sont assez grosses, elle ne veut pas le faire interactivement, mais préfère exécuter le processus la nuit.

Elle écrit une routine qui ouvre une base de données, instancie le moteur de migration, ajuste les propriétés du moteur aux options qu'elle souhaite (information sur la connexion, nom de la base de données nouvelle ou existante, correspondance des champs, etc…) et qui migre la base de données. Elle teste cette routine sur une petite base de données pour vérifier que tout fonctionne sans erreur, en dehors de toute interface utilisateur. Quand tout fonctionne bien, elle écrit un programme "pilote" qui appelle cette routine pour chaque base de données qu'elle veut migrer, et elle lance ce programme le soir avant de partir.

3 Description fonctionnelle

3.1 Moins d'Interface Utilisateur dans le moteur d'Upsizing

Le moteur d'upsizing actuel est une classe appellée UpsizeEngine dans le WizUsz.PRG. Cette classe contient plusieurs dépendances à l'interface utilisateur. On trouve par exemple le code suivant dans la méthode AnalCleanUp :

```
*Turn off the recordsource of grid and columns to prevent errors when source table is
closed

OWizard.Form1.PageFrame1.Page1.PageFrame1.Page7.grdTypeMap.RECORDSOURCE=""
OWizard.Form1.PageFrame1.Page1.PageFrame1.Page7.grdTypeMap.RECORDSOURCE=""
```

Un autre type d'interface utilisateur est que certaines méthodes utilisent les MESSAGEBOX() pour afficher des messages d'erreur à l'utilisateur. Un troisième type est l'usage de WAIT WINDOW et d'une barre thermomètre pour afficher la progression.

Un code comme celui-ci ne fonctionnera pas dans l'environnement du moteur seul, ou bien s'il est appelé depuis d'autres sources telles que l'Explorateur de Données. Ces problèmes seront résolus comme suit :

- Les dépendances du formulaire de l'assistant Upsizing seront redéveloppées dans une sous-classe d'UpsizeEngine, et cette classe sera utilisée par l'Assistant Upsizing au lieu d'UpsizeEngine.
- Les messages d'erreur seront stockés dans un tableau aErrorMessages au lieu d'être affichés dans un MESSAGEBOX() ; une interface utilisateur comme l'Assistant Upsizing pourra alors afficher le contenu de ce tableau pour montrer à l'utilisateur tous problèmes qui se seront produits.
- Au lieu d'utiliser WAIT WINDOW ou d'autres moyens de montrer l'avancement de l'opération, une méthode vide nommée UpdateStatus sera ajoutée à la classe et RAISEEVENT ('UpdateStatus', UpdateMessage) sera utilisé. Les interfaces utilisateur (y compris l'Assistant Upsizing) souhaitant être informées de la progression pourront faire un BINDEVENT() sur UpsizeEngine.UpdateStatus.

3.2 Possibilité d'être appelé depuis l'Explorateur des Données

Les autres changements au moteur de l'upsizing décrit dans cette section devraient suffire pour autoriser le Moteur Upsizing à être appelé depuis d'autres sources que l'Assistant Upsizing, y compris depuis l'Explorateur des Données. Cependant, cette capacité sera testée spécifiquement pour s'assurer de l'absence de problèmes particuliers.

© Microsoft confidentiel

3.3 Gestion correcte des champs Date et DateTime

Il y a rois routines dans UpsizeEngine qui nécessitent des modifications pour gérer correctement les champs VFP Date et Datetime. La première est AnalyzeFields, chargée de la récupération de la structure des tables à migrer. Pendant l'examen de la structure d'une table, cette méthode doit aussi déterminer s'il y a des enregistrements avec des champs Date ou DateTime vides, et si c'est le cas, marquer ces champs comme devant accepter les valeurs nulles, même si ces champs n'acceptent pas les null dans la version VFP de cette table.

Les autres routines sont JimExport et ExecuteSProc, qui sont utilisées pour migrer les données dans une table particulière. Ces méthodes doivent remplacer les Date et DateTime vides par des null.

3.4 Amélioration des performances de chargement des données

La commande SQL Server BULK INSERT est une façon de charger rapidement des données dans une base de données. Cependant, la méthode SendData de l'UpsizeEngine utilise utilise un mécanisme de chargement des données plus lent, si un champ non-caratère accepte les valeurs nulles. La raison en est que quand COPY TO est utilisé pour créer un fichier texte, les valeurs nulles sont perdues et des blancs sont générés à la place. Pour améliore le chargement des données, le fichier texte généré par la méthode GenBulkInsertTextFile sera traité pour enlever les blancs, et SendData sera modifiée pour permettre l'utilisation de BULK INSERT dans ces conditions.

3.5 Gestion des noms d'objets réservés

La méthode RemotizeName de l'UpsizeEngine recherche des noms d'objet dans une table des mots-clés, et quand elle en trouve un, elle renome l'objet en lui ajoutant un suffixe "_". Au lieu de renommer cet objet, on ajoutera des délimiteurs tels que "[" et "]".

3.6 Gestion de SQL Server 2005 et de SQL Server 2005 Express

Un problème de l'Assistant Upsizing avec SQL Server 2005 est qu'il n'affiche pas les sources de données utilisant le pilote ODBC SQL Native Client dans la liste des sources de données proposées à l'utilisateur, tant que l'option Toutes n'est pas sélectionnée, parce qu'il recherche spécifiquement les pilotes SQL Server.il y a trois méthodes à modifier dans la classe ChooseDataSource (dans MyCtrls.VCX) pour supporter le pilote SQL Native Client : GetServerType, InsDataSource, et ServerType.

3.7 Extensibilité

L'application aperçu de VFP 9 propose une extensibilité avec l'utilisation d'un objet extension de l'aperçu. Si un tel objet existe, beaucoup des méthodes dans l'application appellent des méthodes de l'objet. Cela permet à un développeur d'étendre le comportement de la fenêtre de l'aperçu du rapport. D'une manière semblable, l'UpsizeEngine supportera un objet extension de l'Upsizing. Une nouvelle propriété, oExtension, sera ajoutée à la classe. La plupart des méthodes dans la classe vérifiera si l'oExtension contient un objet et si cet objet a une méthode de l'extension pour la méthode courante. Si c'est le cas, la méthode de l'objet sera appelée, en lui passant une référence à l'objet UpsizingEngine pour le cas où la méthode aurait besoin d'une référence à une propriété ou d'un appel à une méthode du moteur d'upsizing.

© Microsoft confidentiel