

Matthias MEUSBURGER
Année 2000-2001
IUT INFORMATIQUE BELFORT

Rapport technique



Responsable de stage :
David Laiymani

I. INTRODUCTION

Ce rapport a pour but d'expliciter et d'approfondir les différentes manières d'intégrer un état Crystal Reports dans Visual Basic. En effet, bien que ces notions aient été abordées dans le rapport de stage, les aspects techniques restent à définir. Cependant, pour ne pas le surcharger inutilement, ce rapport ne traite que des principes régissant les notions abordées. De fait, l'utilisation concrète de ces méthodes se trouvent dans les documents produits à l'intention de MSG Software.

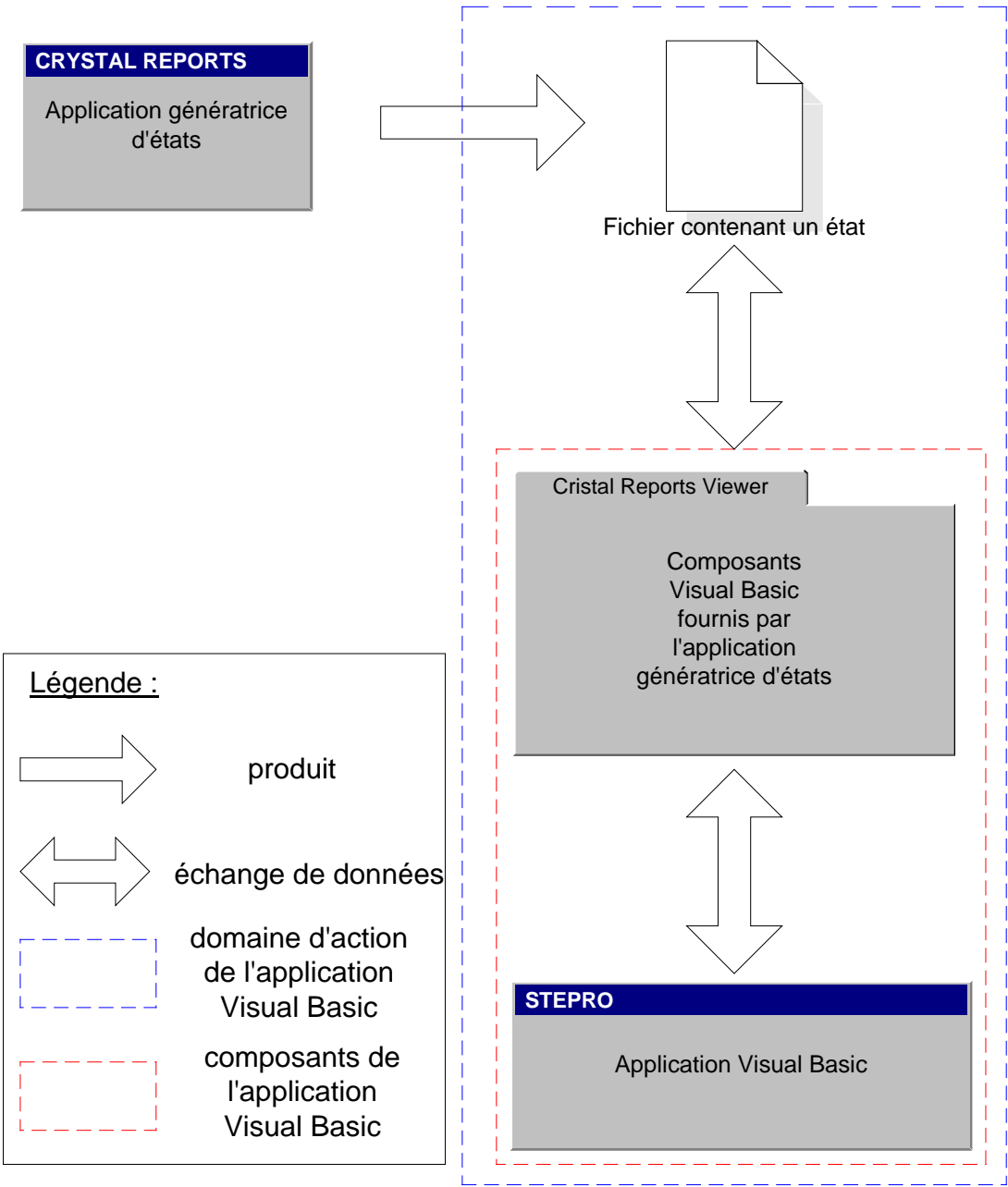
Pour chaque type d'intégration, un schéma illustrera le principe de fonctionnement et les conditions d'utilisations seront précisées.

De plus, un algorithme sera proposé pour résumer la procédure.

Les exemples de codes source commentés correspondants à ces méthodes se trouvent dans l'annexe C : « Documentation interne concernant Crystal Reports » partie 2 : « manuel d'intégration à Visual Basic ».

II. CARACTERISTIQUES COMMUNES A TOUS LES TYPES D'INTEGRATION D'UN ETAT

Interactions entre les différents objets entrant en jeu dans l'édition d'un état au sein de Visual Basic et domaines d'activité



On voit, d'après le schéma ci-contre, que quelle que soit la méthode d'intégration employée, un ou plusieurs composants Visual Basic fournis par le générateur d'états sont à inclure dans l'application.

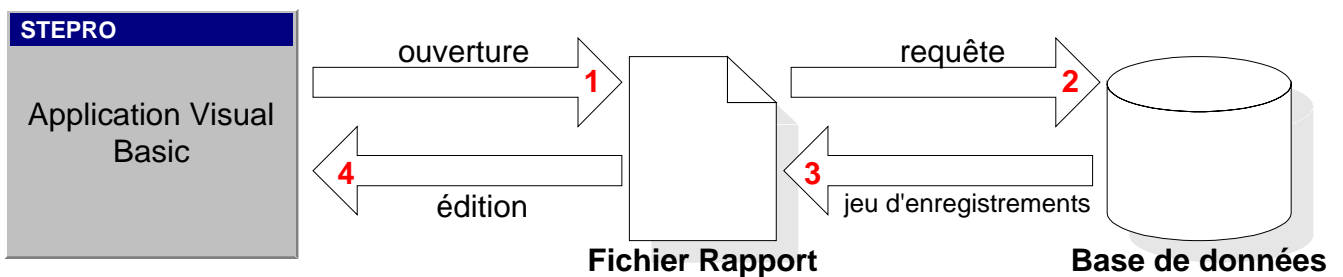
Cependant, les fichiers rapports sont présents dans un répertoire quelconque sous leur forme habituelle. Ainsi, le client peut à loisir personnaliser les états utilisés dans l'application, à la condition qu'il possède une licence d'utilisation du logiciel générateur d'états.



III. INTEGRATION BRUTE

Comme cela a déjà été expliqué dans le rapport de stage, l'intégration brute désigne le fait d'insérer dans une application Visual Basic un rapport tel qu'il a été conçu dans le logiciel propriétaire de ce rapport, sans y apporter aucune modification. Il s'agit de la méthode la plus simple à mettre en œuvre, mais également de la moins souple.

III.1 Principe de fonctionnement



Le schéma ci-dessous montre le principe de l'édition brute sous Visual Basic. On y voit que :

- 1) L'application ouvre le fichier rapport contenant l'état à éditer
- 2) Le fichier rapport envoie sa requête à la base de données
- 3) La base de données, suite à cette requête, renvoie un jeu d'enregistrements au fichier rapport
- 4) L'édition se fait dans l'application avec le fichier rapport comme intermédiaire entre l'application et la base de données.

Note : les composants fournis par l'application génératrice d'état ne sont pas représentés ici. En effet, on considère qu'ils font partie de l'application Visual Basic.

III.2 Conditions d'utilisation

La source de données à utiliser est définie directement dans le fichier rapport.

Si la connexion à la base de données nécessite un mot de passe et/ou un nom d'utilisateur, ceux-ci seront définis dans le code Visual Basic.

N'importe quel connecteur de base de données peut être utilisé. Dans le contexte de STEPPO, nous utilisons **ODBC**.

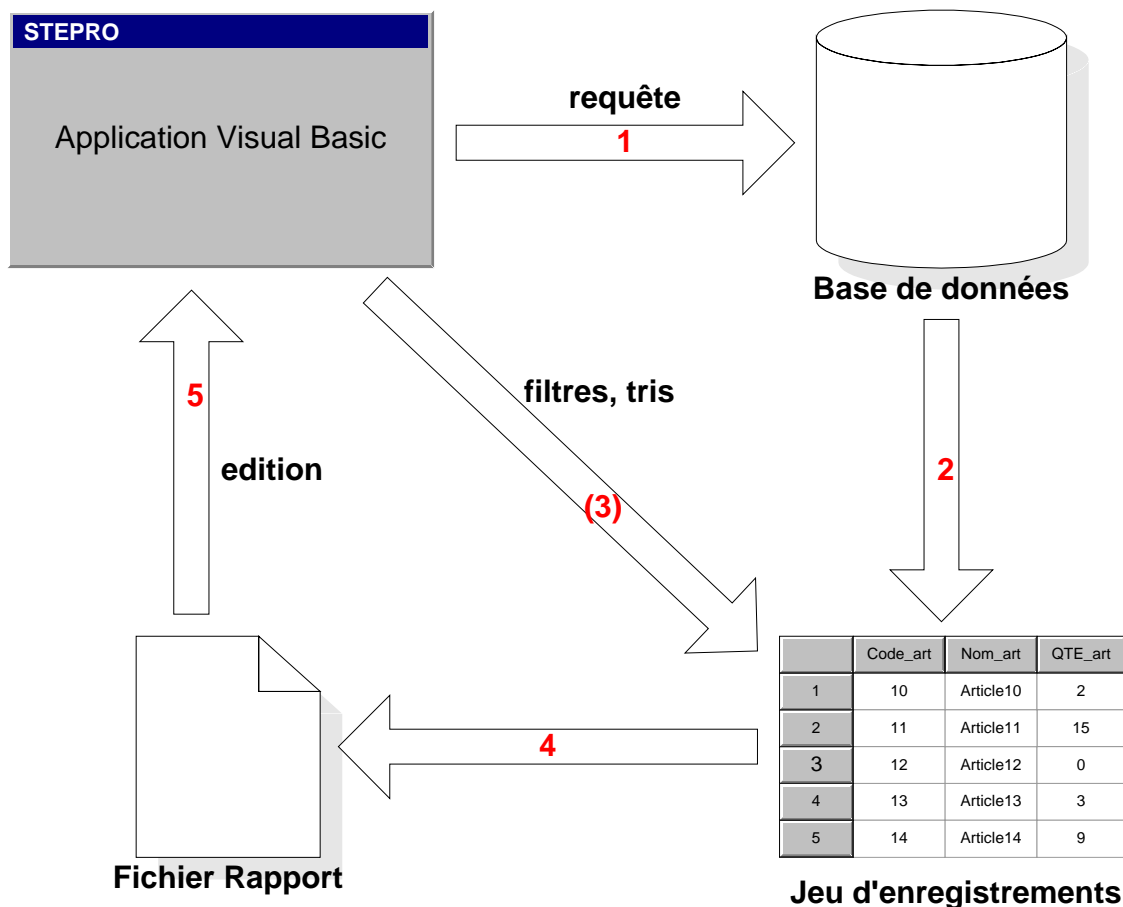
III.3 Procédure Visual Basic

- Ouverture du fichier rapport
- Définition éventuelle des paramètres de connexion à la base de données

IV. INTEGRATION AVEC PASSAGE D'UN JEU D'ENREGISTREMENTS

Cette technique consiste à changer la source de données du rapport, à l'origine la base de données, au profit d'un jeu d'enregistrements créé dans Visual Basic. L'intérêt d'utiliser cette méthode réside dans le fait de pouvoir moduler l'édition d'un état en fonction des différents filtres et tris que l'utilisateur aura appliqués aux données.

IV.1 Principe de fonctionnement



Le schéma ci-dessus montre le principe d'édition d'un état au sein d'une application Visual Basic avec passage d'un jeu d'enregistrement au rapport.

On y voit que :

- 1) L'application envoie une requête à la base de données.
- 2) La base de données renvoie ces informations sous la forme d'un jeu d'enregistrements.
- 3) L'application peut alors traiter ce jeu d'enregistrement en lui appliquant éventuellement des tris ou des filtres.
- 4) Le jeu d'enregistrement est passé au fichier rapport contenant l'état à éditer.
- 5) L'édition de l'état se fait avec le jeu d'enregistrements comme source de données.

IV.2 Conditions d'utilisation

Le type de connecteur de base de données à utiliser doit être de type Active Data, c'est-à-dire ADO (**ActiveX** Data Objects), DAO (Data Access Objects) ou RDO (Remote Data Access).

Le fichier rapport doit être construit d'une seule traite. En effet, lors de la création de la requête, les champs des différentes tables de la source de données sont réunis dans une seule table. Une fois la création de cette table achevée, il devient impossible d'en modifier la composition, ni en utilisant l'interface graphique, ni en éditant le code SQL. Ainsi il est préférable de mettre au point un état en utilisant une connexion de type ODBC puis de recréer celui-ci en Active Date une fois la version définitive achevée.

Il existe cependant une fonction de conversion d'un état de type ODBC en un état de type ADO, mais son utilisation est limitée par deux conditions très restrictives. En effet, si le rapport comporte plusieurs tables ou s'il comporte un sous-état, la conversion s'avère impossible. Il sera donc nécessaire, dans la majorité des cas, de recréer le rapport.

IV.3 Procédure Visual Basic

IV.3.a Dans le cas où le rapport ne comporte pas de sous-état

Cette opération est relativement simple. En effet, elle se décompose en trois parties :

- création ou récupération d'un jeu d'enregistrement.
- ouverture du rapport existant
- redirection de la source de données du rapport au profit du jeu d'enregistrement.

IV.3.b Dans le cas où le rapport comporte un sous-état

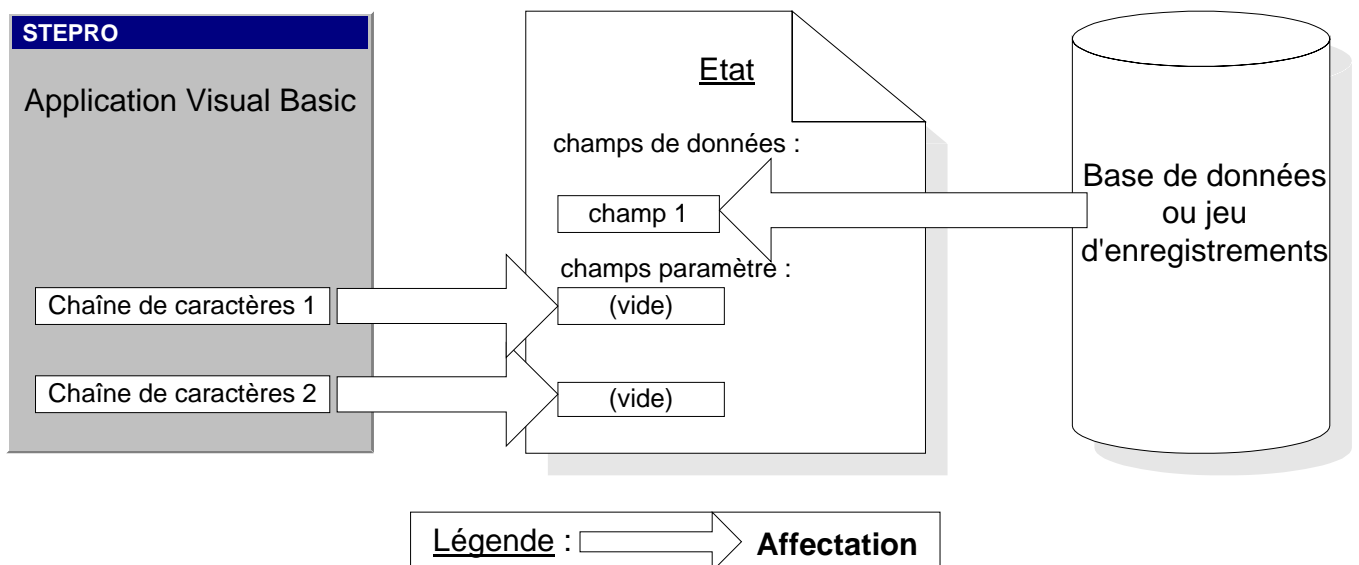
Dans ce cas, deux jeux d'enregistrements sont nécessaires. En effet, il faut en créer un pour l'état principal et un pour le sous-état.

- création ou récupération du jeu d'enregistrements pour l'état principal.
- ouverture du rapport existant
- redirection de la source de données du rapport au profit du jeu d'enregistrements
- création ou récupération du jeu d'enregistrements pour le sous-état
- ouverture du sous-état
- redirection de la source de données du sous-état au profit du jeu d'enregistrements

V. PASSAGE DE CHAMPS DYNAMIQUES

Cette technique permet d'affecter au rapport des champs dont les valeurs sont définies au moment de l'exécution de l'application Visual Basic. Cela permet à l'utilisateur de personnaliser son édition d'état. Il est possible de passer une ou plusieurs valeurs pour un même champ.

V.1 Principe de fonctionnement



V.2 Conditions d'utilisation

Dans le cas où l'on veut passer un ou plusieurs champs comportant chacun une seule valeur, il faut créer dans le rapport autant de champs paramètres dont les formules seront laissées vides que de champs que l'on veut transmettre.

Dans le cas où l'on veut passer un ou plusieurs champs comportant chacun plusieurs valeurs, il faut également créer dans le rapport un champ paramètre par champ dynamique à passer. Cependant, dans ce cas, il est nécessaire de conditionner l'affichage des valeurs par une formule insérée dans le champ paramètre.

V.3 Procédure Visual Basic

V.3.a Pour un rapport ne comportant pas de sous-état

- Création d'une chaîne de caractères (entrée utilisateur, résultat d'une requête SQL, ...)
- Récupération de la collection de champs paramètres
- Identification du champ paramètre adéquat au sein de la collection grâce à un numéro d'index
- Affectation de la chaîne de caractères au champ paramètre en question

V.3.b Pour un rapport comportant un sous-état

V.3.b.i Si aucun des champs paramètres ne se trouve dans le sous-état

Dans ce cas, la procédure reste la même que dans le cas où le rapport ne comporte pas de sous-état. En effet, seuls les composants et fonctions à utiliser changent au niveau de la programmation.

V.3.b.ii Si au moins un des champs paramètres se trouve dans le sous-état

Dans ce cas, il n'est plus possible d'atteindre le champ désiré par un numéro d'index. Il faut parcourir les champs de la collection et trouver celui qui a le même nom que le champ recherché.

- Création d'une chaîne de caractères
- Récupération de la collection de champs paramètres
- Pour chaque champ dans la collection
 - Selon que le nom du champ est
 - "Nom_recherché1" :
 - création d'une chaîne de caractères
 - affectation de la chaîne de caractères au champ
 - "Nom_recherché2" :
 - ...
 - Fin Selon que
- Fin Pour

V.3.c Passage de valeurs multiples

Pour pouvoir passer plusieurs valeurs à un même champ, la méthode utilisée ne diffère pas grandement de celles vues précédemment.

- Création d'une chaîne de caractères
- Récupération de la collection de champs paramètres
- Autorisation du passage de valeurs multiples
- Identification du champ désiré
- Affectation de la chaîne de caractères à ce champ
- Création d'une autre chaîne de caractères
- Affectation de la chaîne de caractères au champ...

On peut affecter à un champ autant de valeurs qu'on le désire et on peut utiliser cette technique pour plusieurs champs de la collection.

Le plus grand changement en ce qui concerne le passage de valeurs multiples est le besoin de créer une formule dans le rapport qui conditionne l'affichage des valeurs. Un exemple de ce type de formule est donné dans l'annexe correspondante du manuel d'intégration à Visual Basic (page 56).

VI. CONCLUSION

Ces trois méthodes d'intégration d'un état à Visual Basic sont loin de représenter toutes les possibilités offertes par les outils Visual Basic de Crystal Reports.

Cependant, leur souplesse leur permet de couvrir la grande majorité des cas de figures se présentant lors de l'édition professionnelle d'états. En effet, il est possible de combiner sans problèmes ces méthodes entre elles. Ainsi, le passage de champs dynamiques peut fonctionner lors d'une édition brute ou avec passage d'un jeu d'enregistrements, qu'il y ait un sous-état ou non. Cette possibilité rend le recours à d'autres méthodes quasiment inutile.

TABLE DES MATIERES :

I. Introduction	2
II. Caractéristiques communes à tous les types d'intégration d'un état .	3
III. Intégration brute	5
III.1 Principe de fonctionnement.....	5
III.2 Conditions d'utilisation	6
III.3 Procédure Visual Basic	6
IV. Intégration avec passage d'un jeu d'enregistrements	7
IV.1 Principe de fonctionnement.....	7
IV.2 Conditions d'utilisation	8
IV.3 Procédure Visual Basic	9
IV.3.a Dans le cas où le rapport ne comporte pas de sous-état	9
IV.3.b Dans le cas où le rapport comporte un sous-état	9
V. Passage de champs dynamiques	10
V.1 Principe de fonctionnement.....	10
V.2 Conditions d'utilisation	10
V.3 Procédure Visual Basic	11
V.3.a Pour un rapport ne comportant pas de sous-état.....	11
V.3.b Pour un rapport comportant un sous-état.....	11
V.3.b.i Si aucun des champs paramètres ne se trouve dans le sous-état.....	11
V.3.b.ii Si au moins un des champs paramètres se trouve dans le sous-état	12
V.3.c Passage de valeurs multiples	12
VI. Conclusion	14
VII. Lexique	16

VII. LEXIQUE

Active X : Un *composant ActiveX* est une unité de code exécutable, un fichier .exe, .dll ou .ocx par exemple, qui suit la spécification ActiveX dans le but de fournir des objets. La technologie ActiveX permet aux programmeurs d'assembler ces composants logiciels réutilisables au sein d'applications et de services.

ODBC : Open Database Connectivity. Protocole standard d'accès aux bases de données relationnelles reposant sur SQL.