Manuel d'initiation

CSPRO

Août 2009

SOMMAIRE

1. Revue du processus de traitement des enquêtes

1) Description générale

Le traitement des questionnaires d'une enquête débute par la réception et le stockage des questionnaires et se termine par la tabulation, suivie de la création d'une base de données. Ainsi, il faut distinguer au cours de ce processus les étapes préparatoires et les étapes opérationnelles.

Les étapes préparatoires du traitement consistent en une participation active des personnes chargées du traitement informatique à l'élaboration du document de projet, du plan de travail, de la méthodologie, du questionnaire et du plan de tabulation, lui-même axé sur le plan d'analyse. Pendant cette étape, sont initiés aussi l'aménagement des salles servant au stockage, la codification lorsque cette phase s'avère nécessaire, la saisie et le traitement des données ainsi que l'acquisition des équipements informatiques.

Les étapes opérationnelles du traitement sont:

- L'élaboration du questionnaire
- La finalisation d'un système efficace de codification
- L'élaboration des manuels de vérification et de codification
- Le traitement d'une enquête pilote
- L'archivage et l'organisation de l'arrivée des questionnaires
- L'élaboration des manuels de saisie
- La vérification manuelle des questionnaires
- Le recrutement et la formation des agents de codification
- Les opérations de codification
- L'élaboration des programmes de saisie et de contrôle
- Le recrutement et la formation des agents de saisie
- Les opérations de saisie et de vérification (double saisie)
- L'élaboration des tests de contrôles de qualité
- L'élaboration des spécifications de redressement
- L'écriture des programmes de redressement
- L'apurement et la fusion des données
- L'élaboration des maquettes de tableaux
- L'élaboration des programmes de tabulation
- La production des tableaux statistiques
- La cartographie thématique
- La dissémination
- L'élaboration d'une base de données

2) Objectifs pour le traitement informatique des données

Le principal objectif est de boucler les opérations de traitement dans les délais impartis après la collecte. Une durée de tout le traitement notamment les phases de codification, de saisie et de contrôle des données doit être estimée à partir du contenu du questionnaire (nombre de variables) et de la taille de l'échantillon. La codification devrait toujours commencer à la fin de la collecte pour donner le temps aux questionnaires d'être reçus, vérifiés et archivés. La formation des agents de saisie et les opérations de saisie doivent aussi être décalée par rapport à la codification et ne pourront commencer qu'après un certain nombre de questionnaires soient déjà préparés (codifiés), ensuite l'apurement se fera après au fur et à mesure que durera la saisie et enfin devront suivre la création du fichier final ainsi que la production des tableaux statistiques.

3) Participation à l'élaboration du questionnaire

L'élaboration d'un questionnaire se fera en plusieurs étapes auxquelles seront étroitement associés les spécialistes responsables du traitement informatique. Les informaticiens doivent apporter leur contribution à la confection du questionnaire et au choix des différents codes (système de codification). Un questionnaire doit être conçu pour comporter un maximum de champs pré-codés, ne laissant que peu de champs à codifier pendant la codification ou mieux encore d'ignorer cette étape.

Une attention toute particulière doit être accordée au système de codification des identifiants géographiques car sa structure hiérarchique doit répondre aux normes du logiciel de traitement utilisé sous peine de ne pouvoir produire certains tableaux.

Le questionnaire doit toujours être soumis aux utilisateurs pour recueillir leurs remarques et l'organisation d'une enquête pilote est aussi toujours souhaitable pour permettre sa finalisation.

4) Disponibilités en personnel, équipements et salles

4-1) Personnel

La structure en charge de l'enquête doit avoir à sa disposition un personnel informaticien capable de prendre en charge les travaux informatiques liés à l'exploitation des données ou à défaut recourir à une collaboration extérieure sous la forme d'une consultation.

4-2) Équipements informatiques

Un besoin de machines doit être exprimé pour l'exploitation informatique en fonction de la durée du traitement notamment de la saisie.

4-3) Salles de stockage, de codification et de saisie

Des espaces doivent être prévus pour servir au stockage des questionnaires, à la codification et la saisie. Ils devront être réfectionnés si nécessaire.

Il est très important aussi que toutes les salles destinées à abriter les activités de traitement (archivage, codification, saisie, traitement et bureaux des informaticiens) soient localisées sur un même site ou dans un même bâtiment afin d'éviter le transport des questionnaires, source potentielle de perte ou d'égarement.

5) Charge de travail

5-1) Codification

La plupart des questions sont en général pré-codées sur le questionnaire. On peut donc raisonnablement estimer la charge de travail de codification par rapport aux questions à coder.

5-2) Saisie

La quantité de postes de travail nécessaire pour les activités de saisie et de vérification est fonction du nombre d'agents affectés à cette tâche, lui-même fonction du volume de données et de la durée prévue pour l'opération.

Un agent de saisie doit avoir un volume de travail de 5 heures par jour, ce qui fait que les agents sont toujours organisés en équipes (brigades) pour rentabiliser l'utilisation des machines.

6) Élaboration des manuels

Les manuels de vérification des questionnaires, de codification et de saisie/vérification) doivent être préparés avant la phase de traitement et peuvent être revus à la lumière de l'évaluation faite à l'issue du traitement de l'enquête pilote.

7) Réception des questionnaires

Cette phase doit être organisée pour que les questionnaires organisés en lots (zone d'enquête) puissent être réceptionnés au fur et à mesure que se déroule la collecte; il sera alors procédé à un contrôle de présence de toutes les zones d'enquête, Il sera vérifié que chaque zone contient effectivement l'ensemble de tous les questionnaires le concernant; ils seront ensuite stockés en salle d'archive dans les rayonnages prévus à cet effet. C'est à partir de là que commence le travail de traitement proprement dit.

8) Enregistrement et Archivage des questionnaires

Des registres de contrôle (archivage, codification, saisie) doivent être tenus en salle d'archivage, de codification et de saisie. Un circuit de déplacement des questionnaires doit aussi être mis en place. Une application en Excel déjà disponible à l'ANSD serait encore mieux.

En ce qui concerne la codification, les dates d'arrivée et de retour aux archives, les travaux de vérification, de codification ainsi que le nom des agents les ayant effectués seront indiqués pour chaque zone traitée.

Il en sera de même pour la saisie où les dates d'arrivée et de retour aux archives, les dates et les noms des agents ayant procédé à la saisie et à la vérification seront aussi indiqués.

9) Codification

La description des procédures et des tâches du personnel de codification sera précisée dans le manuel de l'agent de codification.

10) Saisie et vérification

Les opérateurs de saisie devront travailler en équipes et en horaires décalés. Chaque équipe (matin ou après-midi) sera répartie dans des salles dirigées par des superviseurs.

Un système de contrôle de la qualité et de la quantité de travail effectué par chaque agent doit âtre mis en place (statistiques des opérateurs de saisie) ainsi qu'un système de motivation.

La description des procédures et des tâches des agents de saisie sera faite dans le manuel des agents de saisie.

Le programme de saisie prévoira tous les contrôles possibles sur les variables et la structure des questionnaires. Il est recommandé que la double saisie soit aussi envisagée pour corriger les erreurs de frappe.

Une procédure et un programme d'exploitation des statistiques des prestations des opérateurs de saisie soient mis en place pour permettre de suivre l'évolution de la quantité et de la qualité du travail de chaque opérateur de saisie; on peut, dès lors, envisager divers moyens de stimuler les opérateurs à produire plus et mieux.

11) Contrôles et Corrections

Trois niveaux de contrôles et corrections sont prévus:

11-1) Contrôles et corrections interactifs en saisie

Durant la saisie, une attention particulière doit être accordée au contrôle des identifications afin de s'assurer qu'aucune erreur ne puisse s'y glisser qui pourrait conduire à la disparition de zones géographiques entières et au glissement de l'effectif de ces zones vers d'autres zones impossibles à identifier ultérieurement.

D'autres contrôles très légers pourraient aussi être envisagés sur la présence, l'exhaustivité de certaines données ainsi que les sauts.

A ce niveau, des corrections seront apportées au fur et à mesure que seront signalées les erreurs pendant la saisie.

11-2). Contrôles de cohérence

Les tests de contrôle de cohérence doivent être élaborés en étroite collaboration entre informaticiens, statisticiens, démographes et d'autres personnes susceptible d'y contribuer. Un programme de contrôle est alors élaboré pour relever ces incohérences. Ceci permet de connaître le nombre de cas rencontrés pour chacun des types d'erreur, d'évaluer l'ampleur du problème, de prendre les décisions adéquates pour les corrections automatiques de ces erreurs et surtout de faire des recommandations au niveau de la collecte.

11-3). Corrections automatiques

En fonction donc de l'ampleur des problèmes (erreurs) rencontrés, des spécifications de corrections automatiques seront élaborées et traduites en langages de programmation.

Ce programme peut être exécuté sur les données brutes issues des opérations de saisie en vue redresser des erreurs par la technique du cold ou hot deck.

Ce fichier servira à sortir tous les tableaux destinés à l'analyse et pourra servir ultérieurement pour la constitution d'une base de données.

12) Plans d'analyse et de tabulation

Le plan d'analyse doit être développé très tôt dans le processus car c'est lui qui permettra d'élaborer le plan de tabulation; les plans d'analyse et de tabulation permettront de mieux cerner le contenu du questionnaire. Une fois le plan disponible, les maquettes de tableaux et les programmes de tabulation pourront être élaborées par les statisticiens/démographes. La maquette, le contenu et le niveau géographique de production de chaque tableau doivent être défini.

13) Tabulation

Les maquettes de l'ensemble des tableaux du plan de tabulation seront produites par les statisticiens, démographes et analystes ainsi que les programmes de tabulation.

Dès que le fichier final de données épurées est obtenu, la production des tableaux statistiques peut commencer.

14) Mise à disposition de l'outil informatique pour les analystes

Dès la fin de la production des résultats, les tableaux statistiques seront constitués en une petite base de données (TRS de CSPro).

Il sera également nécessaire aux analystes de faire tourner les logiciels d'analyse démographique, d'élaborer des graphiques (EXCEL etc...) et de rédiger les rapports d'analyse (traitement de texte). Les informaticiens se mobiliseront pour fournir aux analystes toute l'assistance technique requise pour le bon déroulement des activités d'analyse, y compris l'initiation ou la formation des analystes aux différents logiciels.

15) Archivage, dissémination, base de données, système intégré d'information et système géographique d'information (SIG/GIS)

Les données d'une enquête devront être soigneusement archivées sur CD_ROM afin de permettre à tout moment la production de tableaux complémentaires nécessaires aux utilisateurs, notamment pour les analyses approfondies.

Il sera un plus de mener à bien les activités de dissémination(WEB), notamment la création d'une base de données, la constitution d'un système intégré d'information et l'élaboration d'un Système d'Information Géographique, fusionnant les résultats du traitement de données et le fruit des activités cartographiques.

16) Logiciels de traitement.

On en distingue deux types :

- les logiciels d'exploitation
- les logiciels d'analyse
- les logiciels d'exploitation et d'analyse.

1. Les logiciels d'exploitation.

Ces logiciels permettent de faire un traitement d'une enquête depuis la confection du programme de saisie, les contrôles de qualité jusqu'à la production de tableaux. On citera en particulier les logiciels suivants :

- CSPRO sur Windows produit par le Census Bureau. Il a remplacé les logiciels IMPS et ISSA et résulte de la fusion de ces deux logiciels. Il a des avantages certains notamment sur les contrôles de structure, de cohérence en temps réel à l'entrée des données et permet aussi de transférer de façon très conviviale les données vers d'autres logiciels d'analyse de données (SPSS, STATA).

- SAS permet aussi de faire des applications de traitement des enquêtes et présente les mêmes avantages que CSPRO à la différence qu'il gère directement des bases de données.

2. Les logiciels d'analyse.

- SPSS sous Windows. Ce logiciel permet de faire des analyses statistiques très poussées et possède des fonctions de création de données statistiques et de tabulation très conviviale.

•STATA sous Windows, comme SPSS, il permet de permettre de faire des analyses statistiques très poussées et possède des fonctions de création de données statistiques et de tabulation très conviviale. L'avantage qu'il présente par rapport SPSS est que ses programmes s'exécutent en mémoire donc très rapidement. Ceci présente un inconvénient qui est qu'il demande trop d'espace mémoire.

Remarques : Il arrive très souvent que lors d'une exploitation d'enquête que plusieurs logiciels soient utilisés selon leur degré de performance des modules qui le composent. Par exemple on peut utiliser CSPRO pour la saisie et les contrôles et SPSS ou STATA pour la tabulation ou bien CSPRO pour la saisie et les contrôles et SPSS ou STATA pour la tabulation ou encore CSPRO pour la saisie, le traitement et la tabulation.

3. Les logiciels d'exploitation et d'analyse.

•SPSS sous Windows. Ce logiciel peut être utilisé aussi bien pour l'exploitation que l'analyse car possédant un module de saisie appelé DATA ENTRY pas toujours convivial mais quand même permettant de faire une exploitation sans avoir à recourir à un autre logiciel pour la saisie et les contrôles.

2. Vue d'ensemble de CSPRO

CSPRO puise ses origines dans IMPS qui intégrait trois logiciels à savoir : le CENTRY (saisie des données), le CONCOR (Correction des données de Recensements) et le CENTS (Tabulation). Plusieurs améliorations vont être faites sur ce logiciel qui donnera naissance au CSPRO issu d'une association entre les développeurs de IMPS (U.S Census Bureau) et de ISSA (Macro International). Il remplace les deux applications sur le marché. Le principal avantage est le traitement des fichiers hiérarchiques.

Il contient plusieurs modules exécutables dont :

- Application designer (CSPRO.EXE)- création des masques de saisie et production des tableaux croisés ;
- Data Entry System (CSENTRY.EXE).

2-1. Configuration requise

Le logiciel CSPRO tourne sur un minimum de matériels. Il faut au moins :

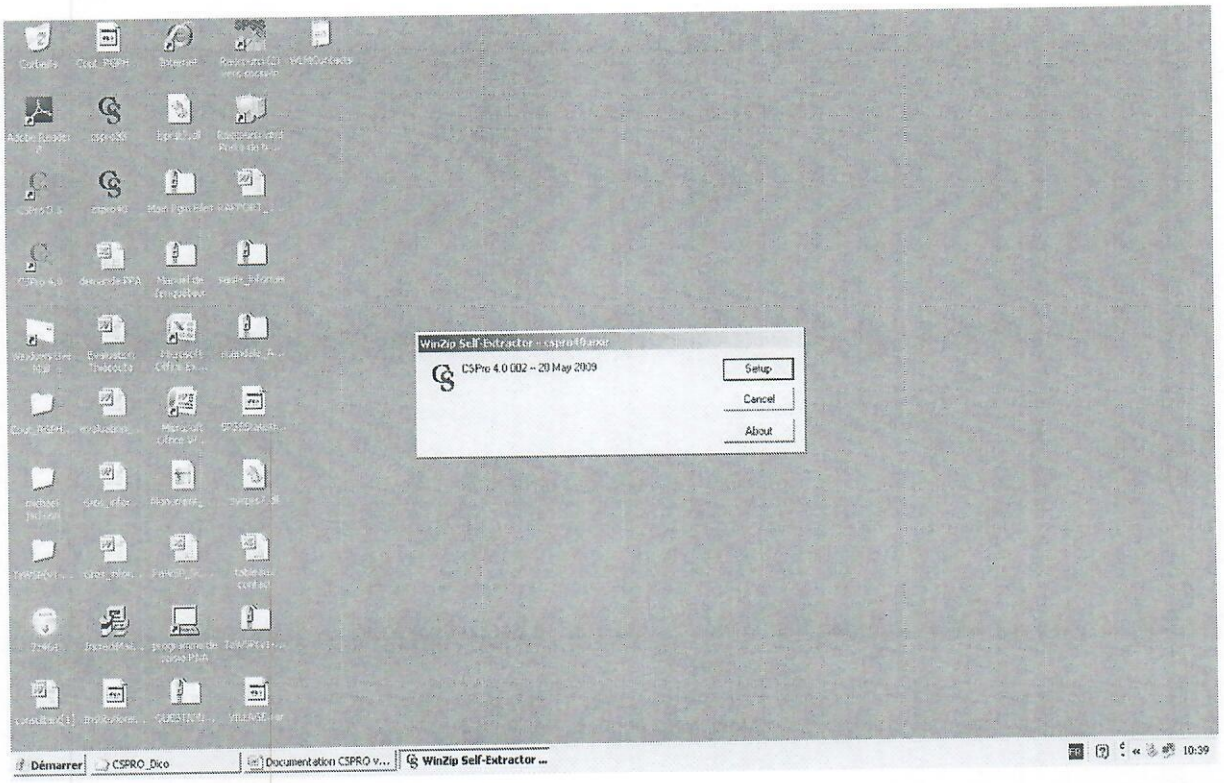
-Windows version 3 ou plus
- Un processeur 80386 ou supérieur
-4 méga octets minimums de mémoire vive (RAM)
-Disque dur avec au moins 23 méga octets d'espace disque disponible
-Un lecteur de disquette de haute densité 3,5
-Un adaptateur graphique avec une résolution 640x480 (VGA) ou supérieur.

Il existe aussi une version client serveur pas très connue.

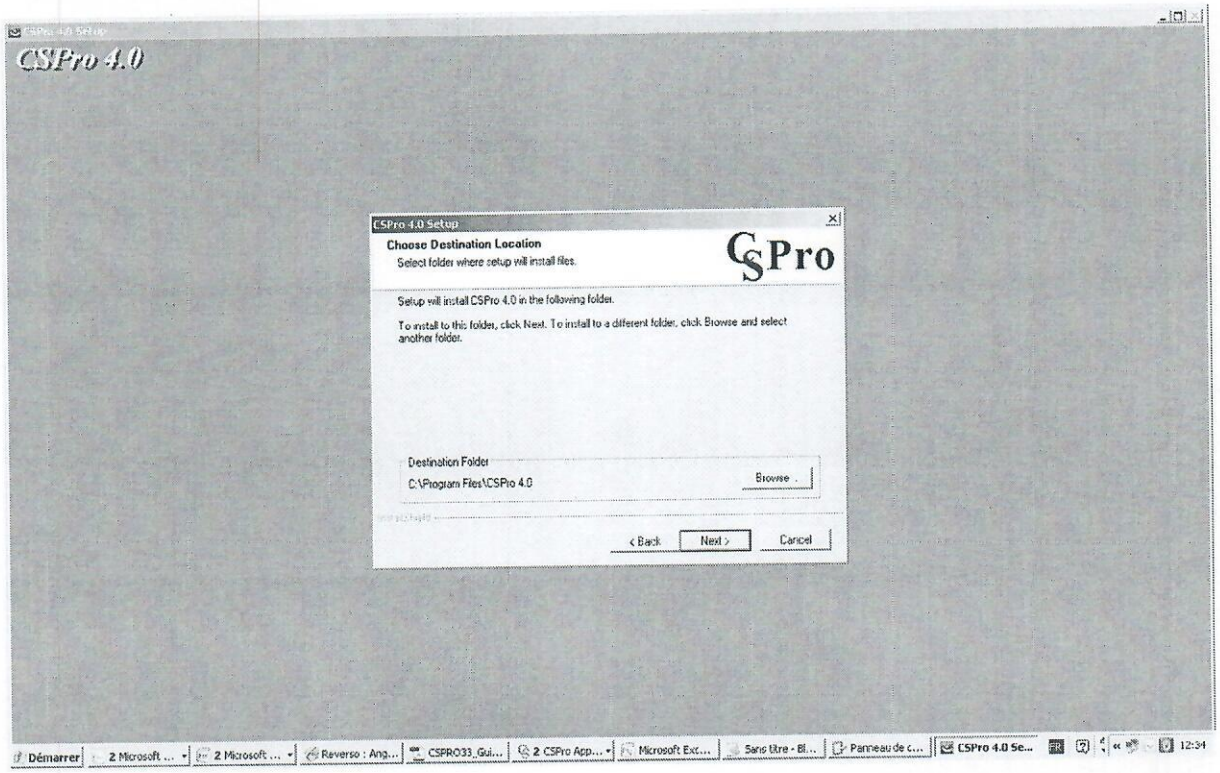
2-2. Installation

CSPRO version 3.3 est disponible sur Internet sous le site www.census.gov/ipc/www/cspro et peut être téléchargé comme une application exécutable. Il est installé en lançant cspro33.exe et en suivant les instructions. Il est techniquement distribué gratuitement par le département International Programs Center du Bureau of Census U.S.

En double cliquant sur l'application cspro40, la procédure d'installation est lancée comme pour tout autre programme. Patientez pendant que le programme d'installation se charge en mémoire. Cela peut prendre quelques minutes. Un premier écran apparaît avec la commande setup sur lequel il faut cliquer pour arriver à l'écran qui suit:



Cliquez sur le bouton **Setup** pour poursuivre la procédure d'installation. **Cancel** si on veut arrêter l'installation et puis **Back** si pour une raison quelconque on veut retourner à l'écran précédent.

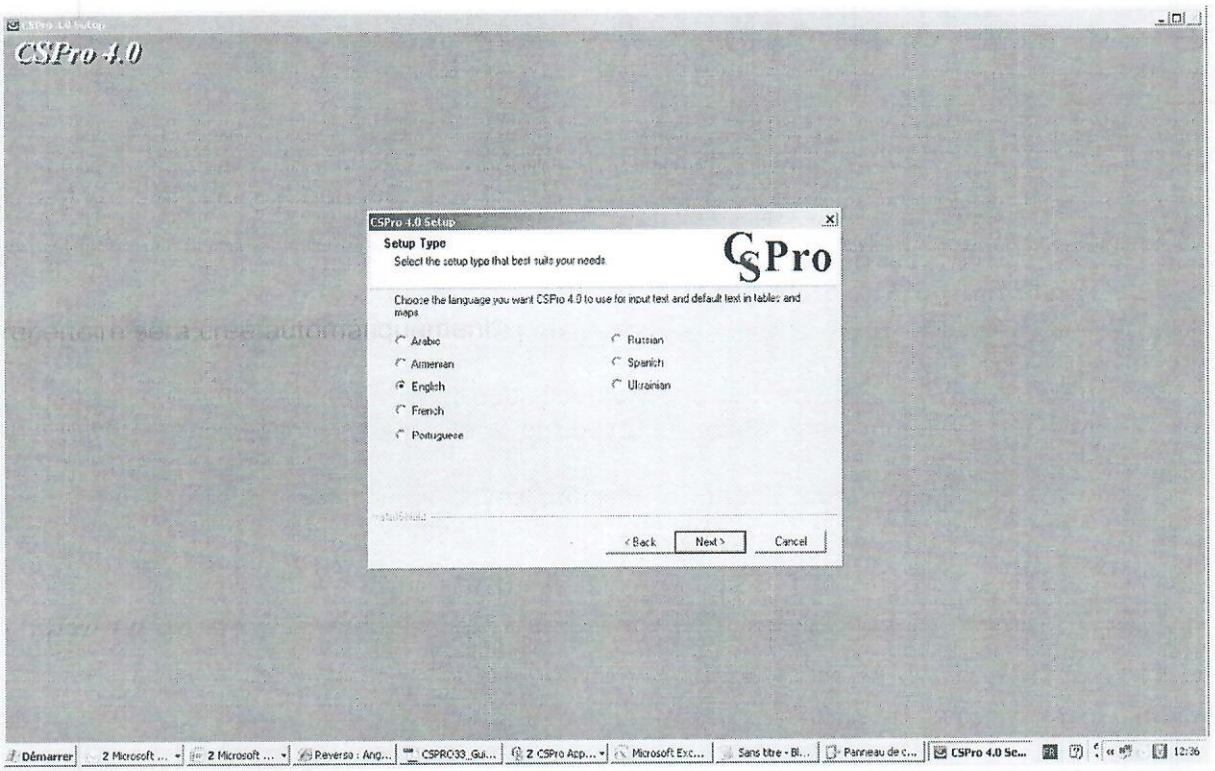


On a la possibilité de choisir à ce niveau le répertoire devant contenir le logiciel.

Pour les modifications, il faudra cliquer sur **Browse**, alors la possibilité de modifier le répertoire d'accueil est donnée.

Dans notre cas, il est conseillé de poursuivre l'installation avec les options de base. Le logiciel s'installe dans le sous-répertoire **CSPro4.0** dans le dossier **Program files**. Ce sous-répertoire sera créé automatiquement.

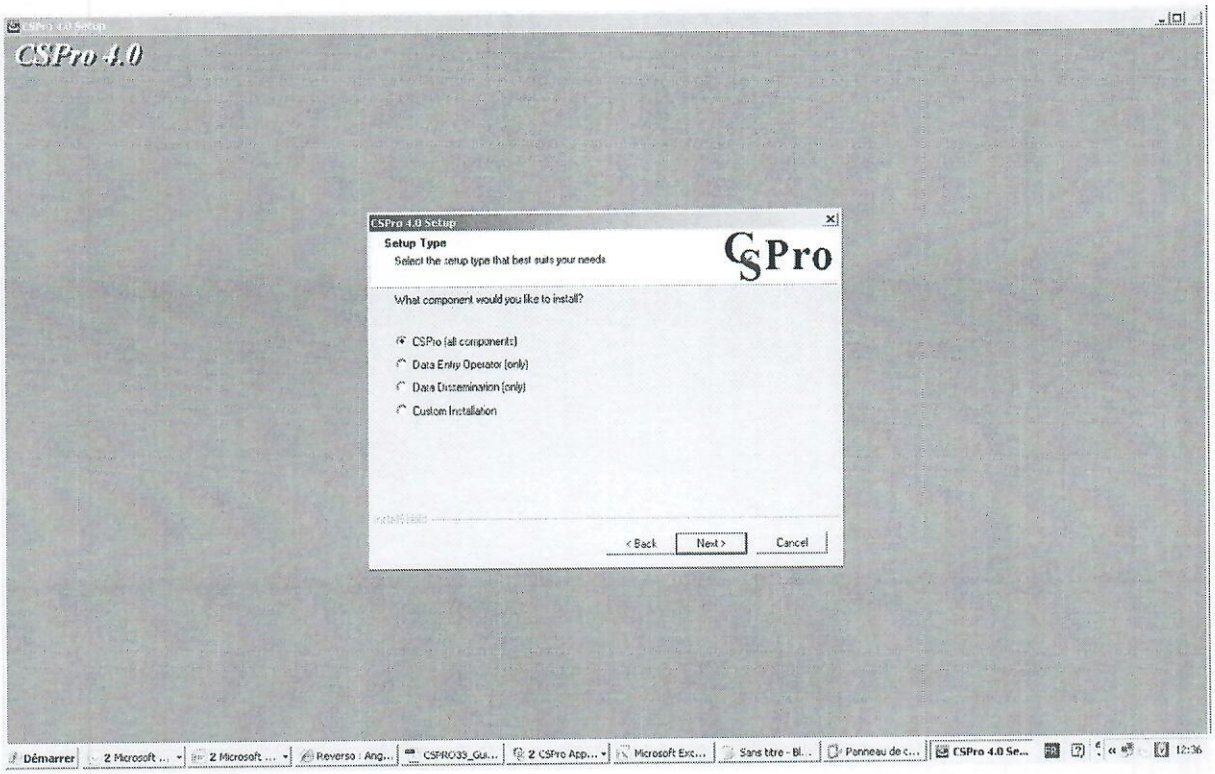
Cliquez ensuite sur le bouton **Next** pour poursuivre la procédure d'installation. **Cancel** si on veut arrêter l'installation et puis **Back** si pour une raison quelconque on veut retourner à l'écran précédent.
L'écran suivant apparaît et vous devez préciser la langue dans laquelle seront entrés les textes des tables; prendre le français (French) :



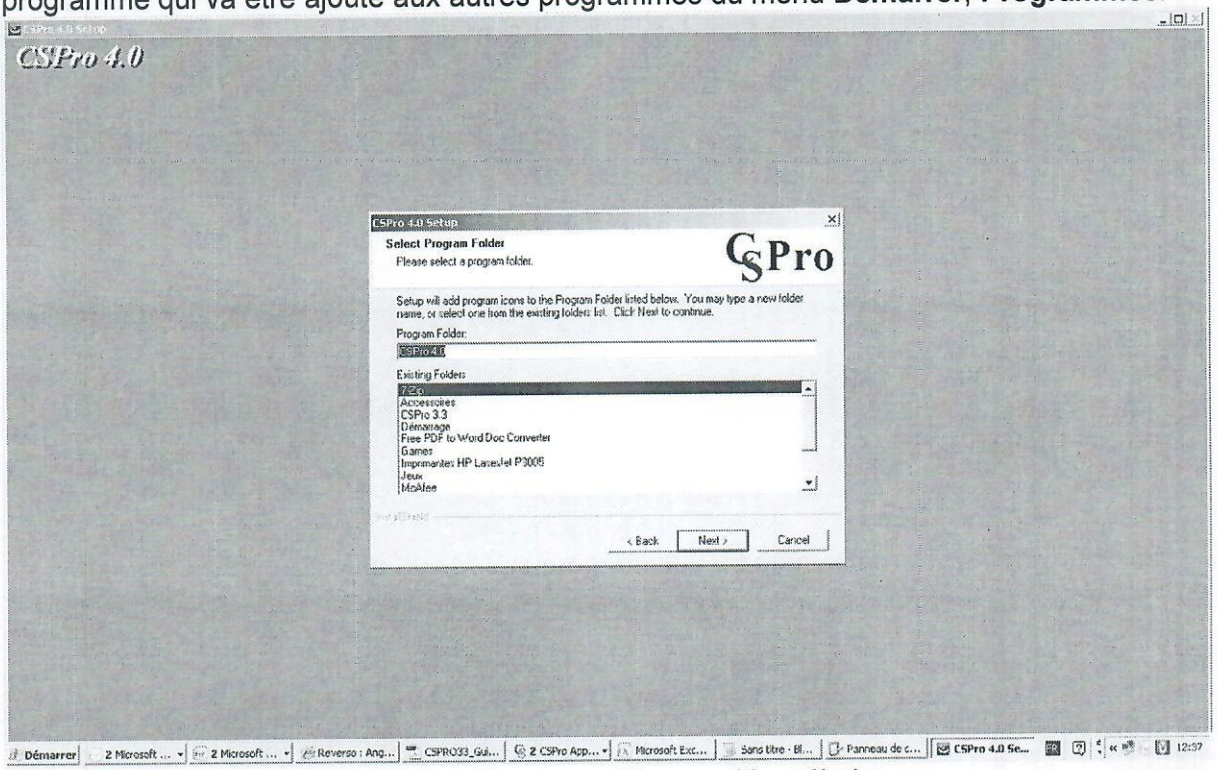
Cliquez sur le bouton **Next** pour poursuivre la procédure d'installation. **Cancel** si on veut arrêter l'installation et puis **Back** si pour une raison quelconque on veut retourner à l'écran précédent.

L'écran suivant apparaît et permet à l'utilisateur de choisir les modules qu'il veut installer:

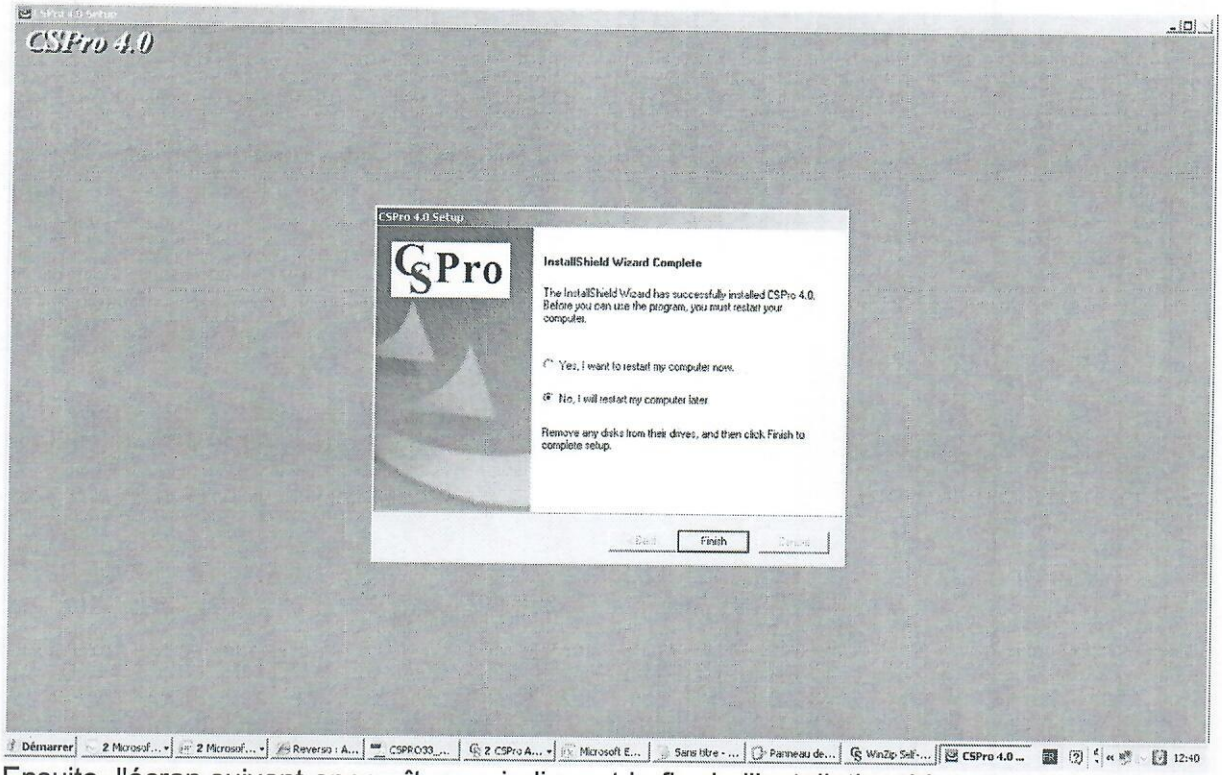
Ici, il faut choisir les composantes à installer; prenez la première option Cspro (all components).



Faites **Next** pour continuer. L'écran suivant vous indique que **CSPro4.0** est le nom du programme qui va être ajouté aux autres programmes du menu **Démarrer**, **Programmes**.



Continuez en cliquant sur le bouton **Next** pour démarrer l'installation



Ensuite, l'écran suivant apparaît vous indiquant la fin de l'installation. Vous pouvez décocher le [**Look at the README me file**] pour ne pas lire ce fichier.

Cliquez sur **Finish** pour terminer.

CSPro est installé avec succès; une icône est mise sur votre bureau Windows.

Pour démarrer CSPro, vous pouvez, à votre convenance, cliquer sur l'icône ou passer par la procédure Démarrer, Programmes, CSProX, CSProX.

3. Création d'un masque de saisie

La création d'un masque de saisie sous CSPRO implique l'exécution de 4 étapes précises qui sont :

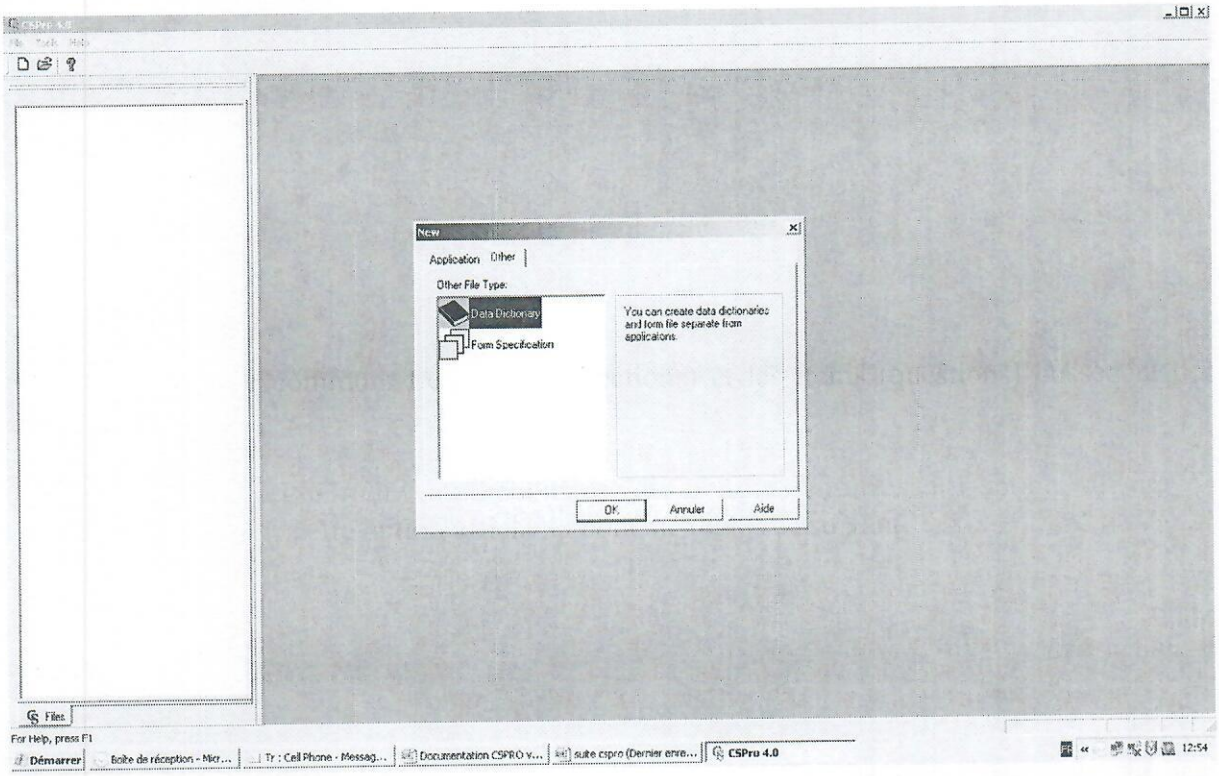
- La définition du dictionnaire des données
- La génération du masque de saisie
- Les instructions de contrôle
- La création de l'application de saisie

3-1. La définition du dictionnaire des données

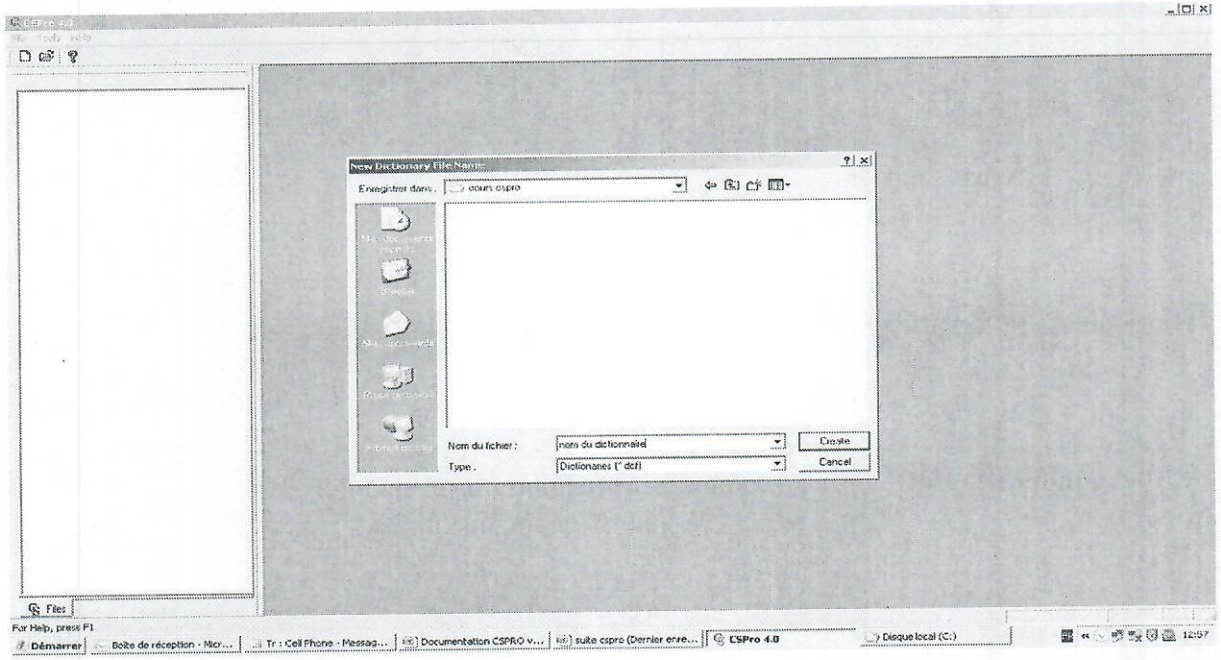
Pour créer le masque de saisie sous CSPRO, il faut, sur la base du questionnaire, décrire la structure des données en question. C'est pour cette raison que la conception du questionnaire et du masque de saisie doivent être des opérations intégrées. Le dictionnaire des données décrit l'organisation du fichier de données. Il permet ainsi de préciser la nature, le type et la taille des records, des variables, des modalités (selon le cas). **Un record** correspond à une section du questionnaire. Il permet d'enregistrer les réponses relatives aux questions (**item**)

traitant d'un thème commun. Les records (sections) et les items (questions) possèdent des propriétés (type, libellé, taille, etc.).

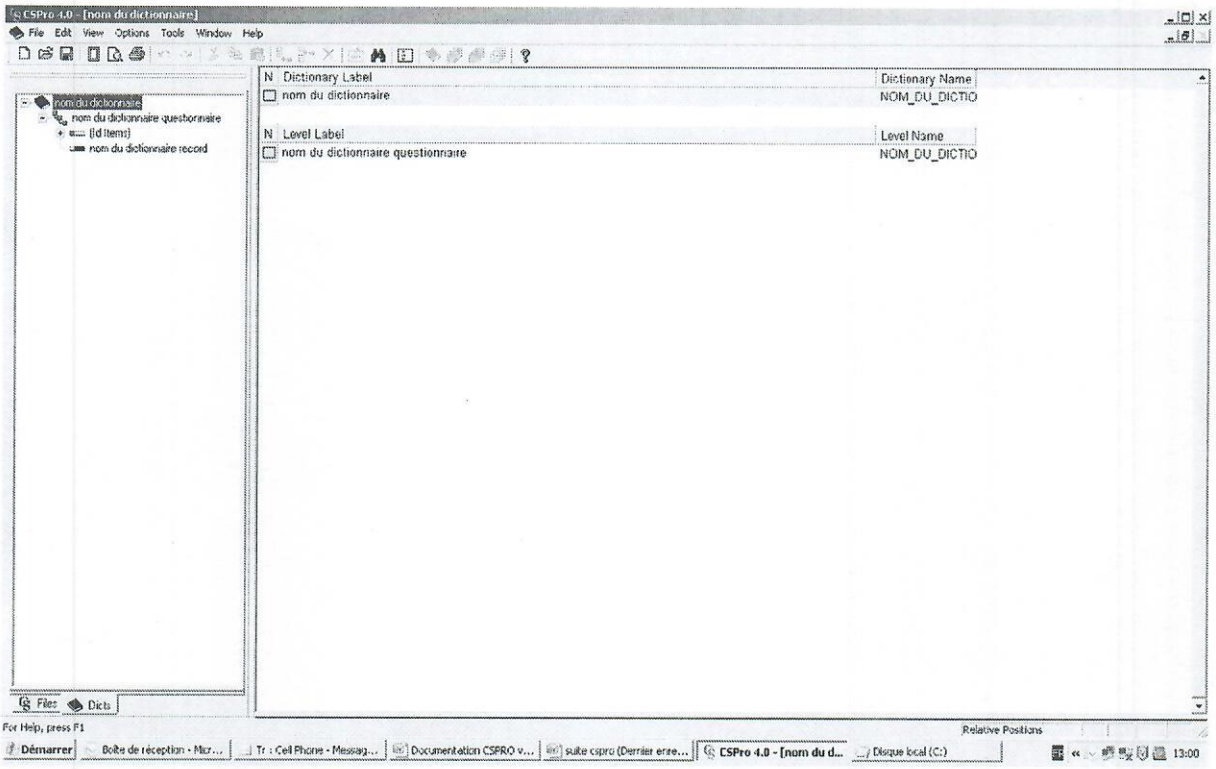
Sous-CSPRO, on crée un nouveau dictionnaire des données en utilisant l'option **File/new…** du menu. Le choix de cette option affiche la boîte de dialogue suivante :



Le type d'objet correspondant au dictionnaire de données c'est « **Data Dictionnary** ». En sélectionnant cette option, l'écran suivant apparait :



Il reste donc à donner un nom au fichier, ainsi que le chemin, puis cliquer sur **create**. Si les informations données sont conforment à celles affichées, faire terminer. A défaut, faire précédent pour modifier les erreurs éventuelles. On obtient alors l'environnement de travail suivant :



Le dictionnaire initial comporte deux records : un record « **Id items** » destiné à contenir l'identifiant (composé le cas échéant de plusieurs variables) et un record vide. La première chose à faire, c'est (en fonction de la structure du questionnaire) d'insérer et de modifier le nom des records pour chaque section du questionnaire.

Pour ajouter un record, il est conseillé de cliquer sur le bouton questionnaire de la fenêtre de gauche. L'environnement de travail se présente alors comme suit :

En utilisant les menus contextuels dans la fenêtre de droite, on a la possibilité d'ajouter, de modifier ou de supprimer un record. Les propriétés d'un **record** sont les suivantes :

Record Label : il s'agit du libellé que l'on veut donner au record

Record Name : correspond au nom logique du record, c'est lui qui sera utilisé le cas échéant dans les procédures de contrôle pour faire référence au record. Il est composé d'au plus 32 caractères. Les caractères possibles sont : A-Z, 0-9, et le _. **Rmq** : la première position est réservée pour les lettres de A à Z. Par ailleurs, la dernière position ne saurait être le _. Il faut

noter pour finir que CSPRO possède des noms réservés[1] qui ne peuvent être utilisés au niveau du dictionnaire.

Type value : C'est un alphanumérique qui permet d'identifier les lignes du fichier des données qui représentent un record donné.

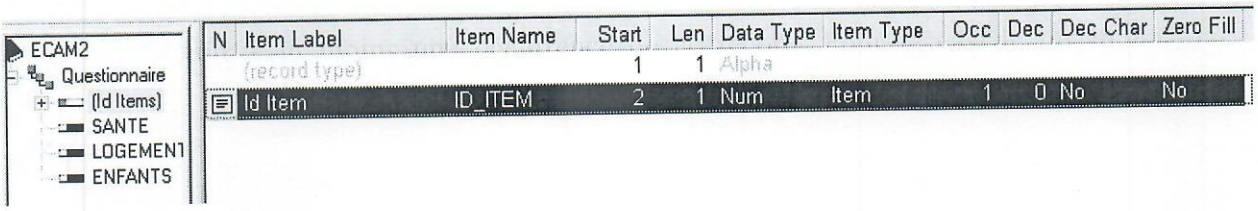
Required : prend deux valeurs possibles : Yes/No. Il s'agit de dire si pour chaque questionnaire, on a au moins une occurrence de ce record. Pour le cas du questionnaire ECAM, les sections où il y a des conditions d'éligibilité prennent la valeur No pour cet attribut.

Max: Précise le nombre maximum d'occurrences possibles pour une section donnée. Si ce nombre est supérieur à 1, on pourra générer des **rosters** au niveau du masque de saisie.

Remarque : Avant les variables de chaque record, on a ne répétition des identifiants.

A cette phase, il ne reste plus qu'à ajouter les **items** (questions) à l'intérieur de chaque **record** (section). Pour ajouter un **item**, le processus est le même que lorsqu'on veut ajouter un record.

Dans la fenêtre de gauche, il faut cliquer sur le record à l'intérieur duquel on aimerait ajouter des items. La fenêtre de droite prend alors l'apparence suivante :



Utiliser le menu contextuel dans la fenêtre de droite pour ajouter, supprimer ou modifier un item. Les propriétés des items sont différentes de celles des records. On ne s'étendra pas ici sur les propriétés **Item label** et **Item Name** qui rejoignent le cas record label et record Name.

Start : Champ très important, précise la position du fichier à partir de laquelle la réponse à la question sera stockée. Il est conseillé de laisser CSPRO gérer automatiquement ce champ, sauf pour le cas des **SubItems** (à voir plus loin).

Len : permet de donner la taille du champ en terme de nombre de positions.

Data Type : permet de préciser le type de donnée du champ en cours. Les types possibles sont : numériques et alphanumériques.

Item Type : Par défaut, ce champ prend la valeur Item. Il existe des questions dont les réponses peuvent être subdivisées en sous réponses. L'exemple typique c'est celui de la date qui est composé du jour, du mois et de l'année. On peut demander à CSPRO de gérer le jour, le mois et l'année comme des Subitems de la variable date. Lorsqu'on choisit Subitem, il faut

---

modifier les propriétés Start et Len de telle sorte que la position du Subitem coïncide avec la partie de l'item qu'il représente.

Occ : permet de préciser le nombre d'occurrence de l'item dans le record. Si on décide par exemple que cette valeur vaut 5 pour une variable qui tient sur deux (02) positions (Len), CSPRO va réserver 10 positions pour la saisie des informations sur ce champ.

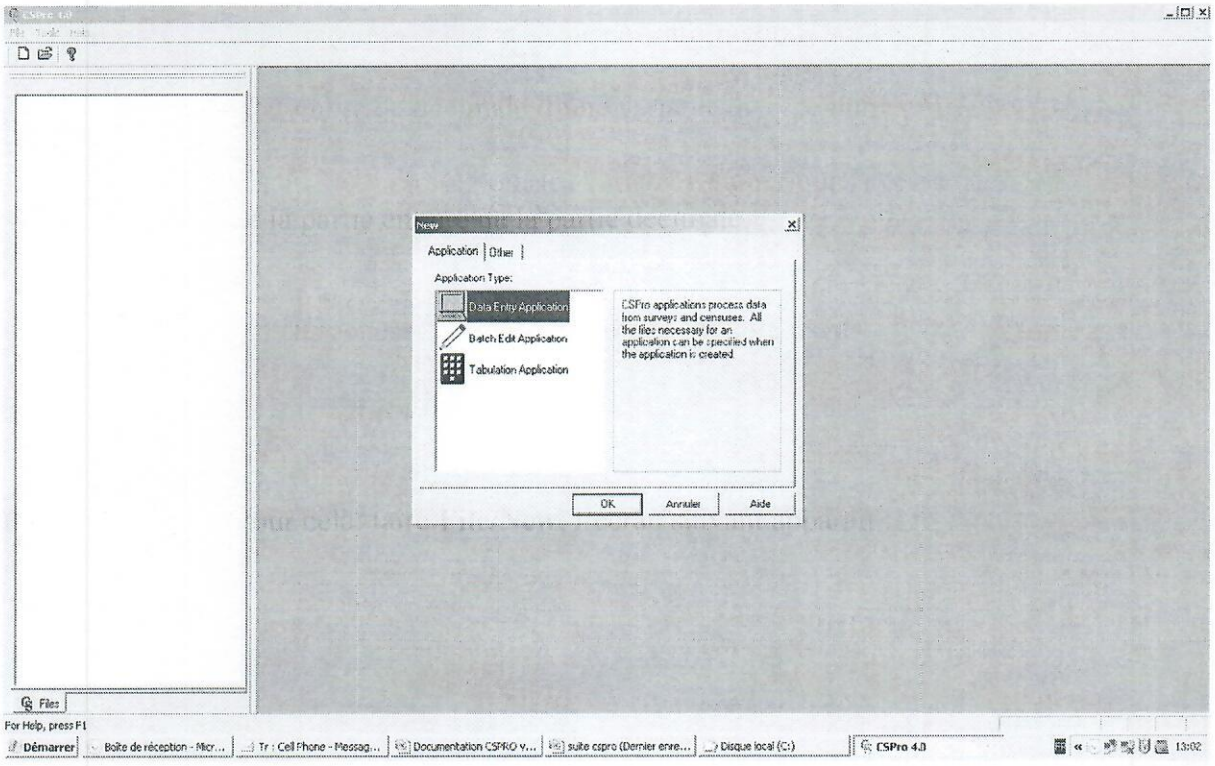
Dec : correspond au nombre de décimale d'une variable de type numérique.

Dec char : Vaut Yes ou No, précise pour les variables numériques avec décimale s'il faut une virgule pour séparer la partie entière de la partie décimale.

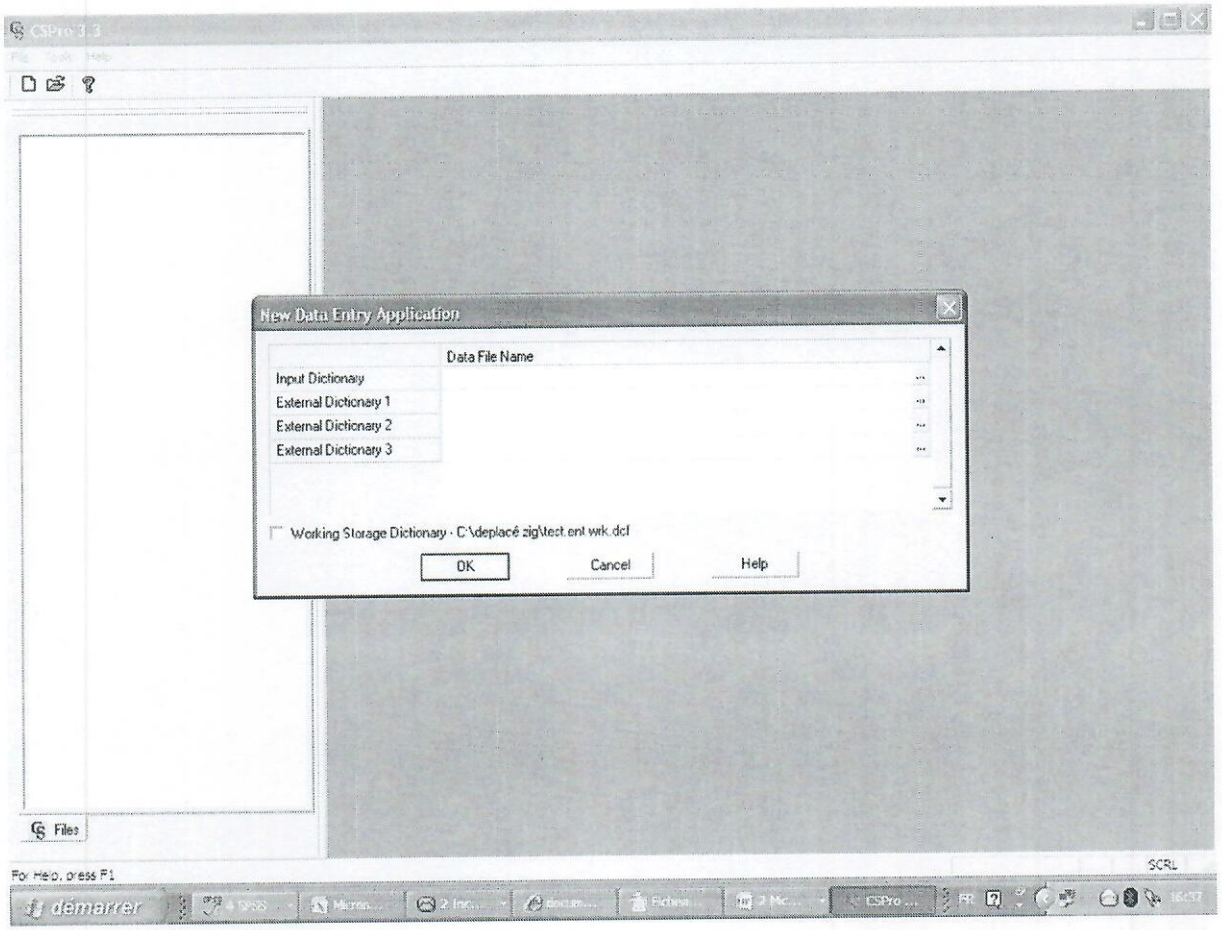
Zéro Fill : Prend deux valeurs possibles : Yes ou No. Lorsque la valeur choisie est Yes, CSPRO complète (le cas échéant) les saisies par des zéro à gauche.

3-2. La génération du masque de saisie

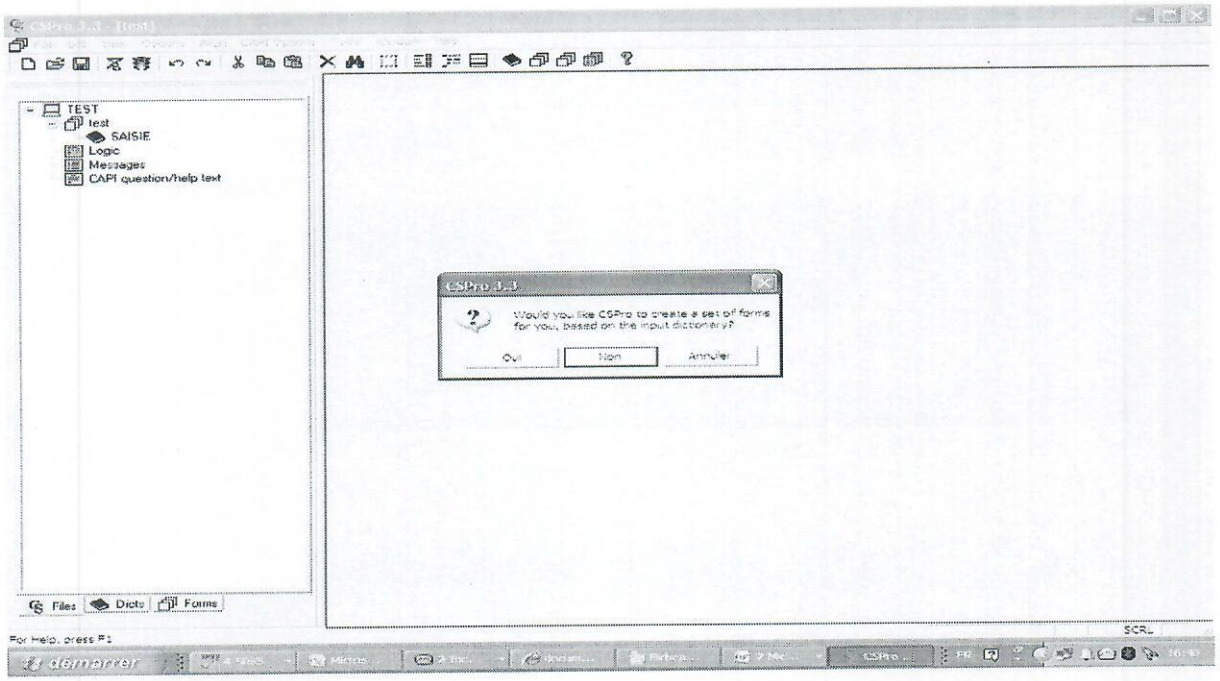
Le dictionnaire des données étant achevé, la prochaine étape consiste à générer le masque de saisie. Pour cela, faire **File/New**, l'écran suivant apparait :



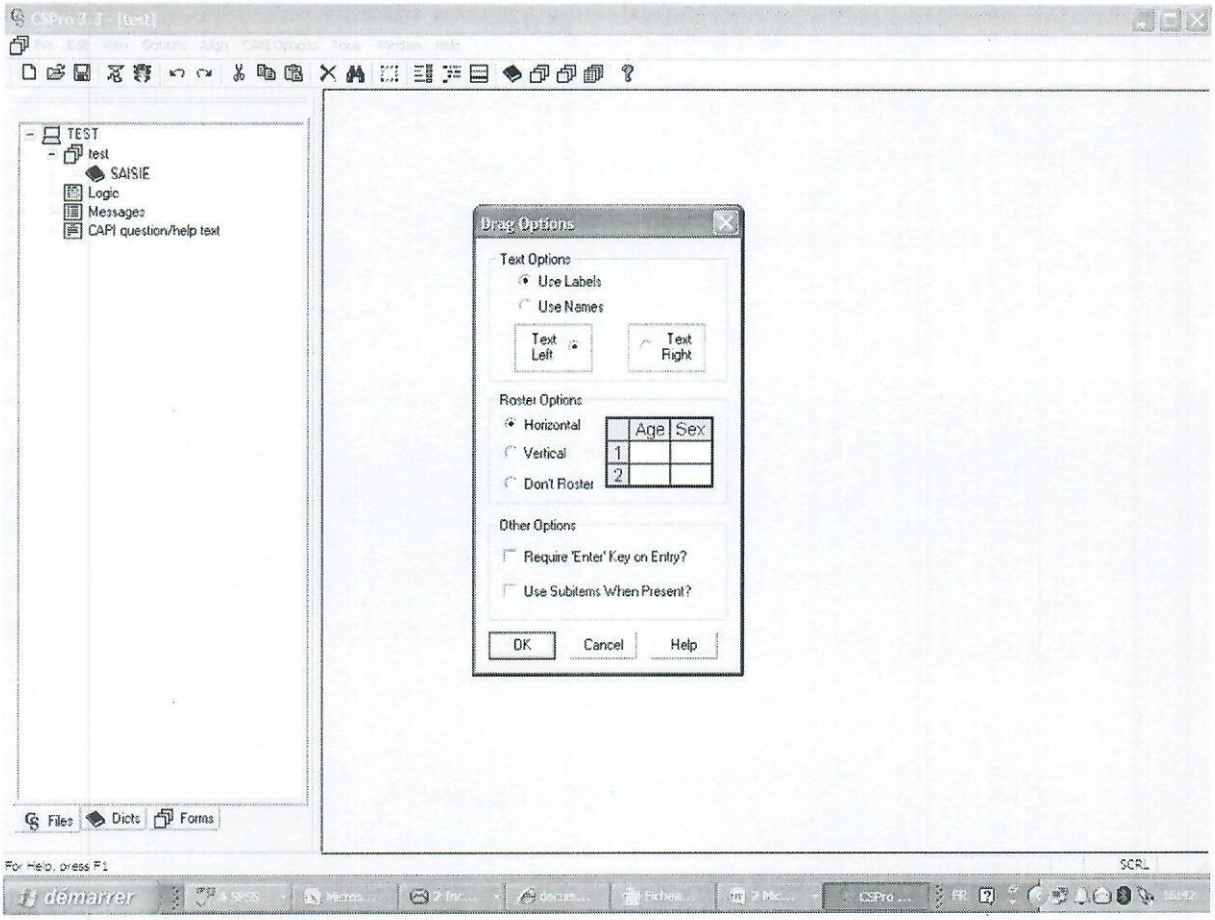
Au niveau de la boîte de dialogue qui s'affiche, choisir comme type d'objet « **Data Entry Application** ». Puis cliquer sur **ok**. A cette étape, il faut donner le chemin et le nom de l'application de saisie avec l'extension « **.ent** », cliquer sur « **Create** ». L'écran suivant apparait :



Indiquer au niveau de Input Dictionary le chemin et le nom du fichier dictionnaire des données puis cliquer sur « **ok** ».

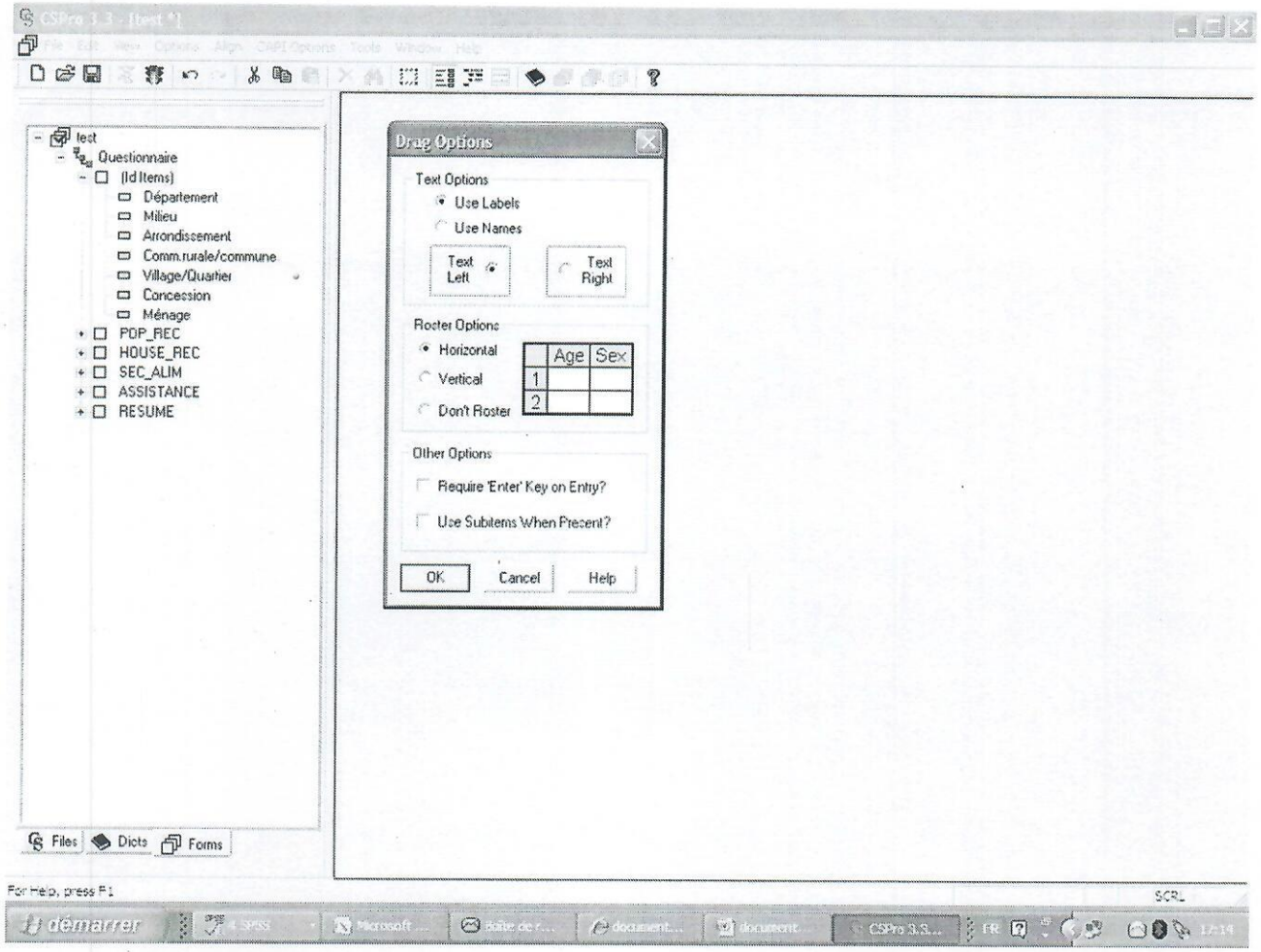


Cliquer sur « **oui** », l'écran suivant apparait :



Selon la présentation du masque de saisie que l'on veut voir apparaître(étiquettes de variables, enregistrements en ligne ou colonne), on procède à des choix au niveau de l'écran apparu. Ensuite cliquer sur « **ok** », CSPRO génère un masque de saisie.

L'expérience montre que très souvent, le premier masque de saisie généré est loin d'être acceptable. Utiliser le **Menu Edit** et l'option **Generate form** pour régénérer le masque. Cette fois-ci on obtient la boîte de dialogue suivante :

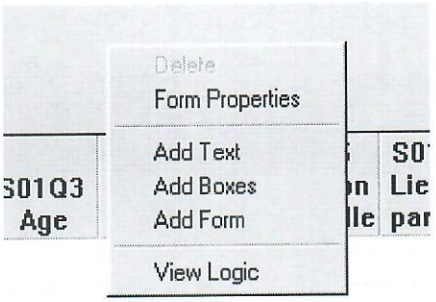


L'option **Require « Enter » Key on Entry** impose que la saisie de chaque variable soit validée avant qu'on ne passe d'un champ à l'autre. Il est conseillé de ne pas cocher cette case.

L'option **Use Subitems When present** signifie lorsqu'elle est cochée qu'en cas d'existence de Subitem, ce sont ces derniers qui seront saisies, et l'item pourra être déduit comme étant la concaténation de ces Subitems.

L'option **Roster When** conduit à une présentation très conviviale des **records** sous forme de tableau lorsque ceux-ci ont plus d'une occurrence.

Le masque de saisie pour être complet, doit faire l'objet d'une certaine mise en forme. On devrait par exemple ajouter des **champs miroirs**. Il s'agit de variables qui sont présents dans un écran sans qu'on puisse les modifier. Les identifiants sont très souvent placés comme champs-miroirs. Pour créer un champ miroir, on fait glisser les identifiants du dictionnaire des données vers l'emplacement souhaité à l'écran de saisie. On peut également modifier la disposition des **Rosters** et des différents champs. Un menu contextuel sur l'écran de saisie affiche la boîte de dialogue suivante :

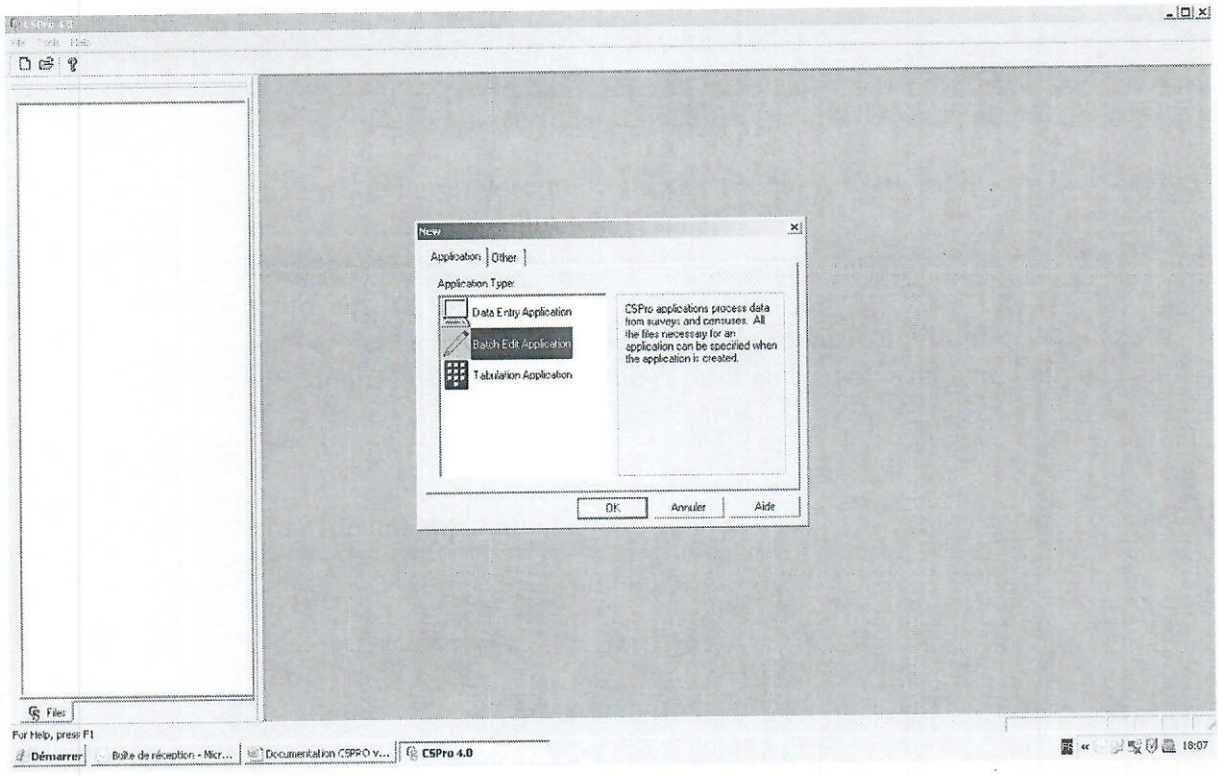


A l'aide ce menu, on peut ajouter et encadrer du texte. L'option **View logic** affiche le module de programmation des procédures de contrôle à la saisie.

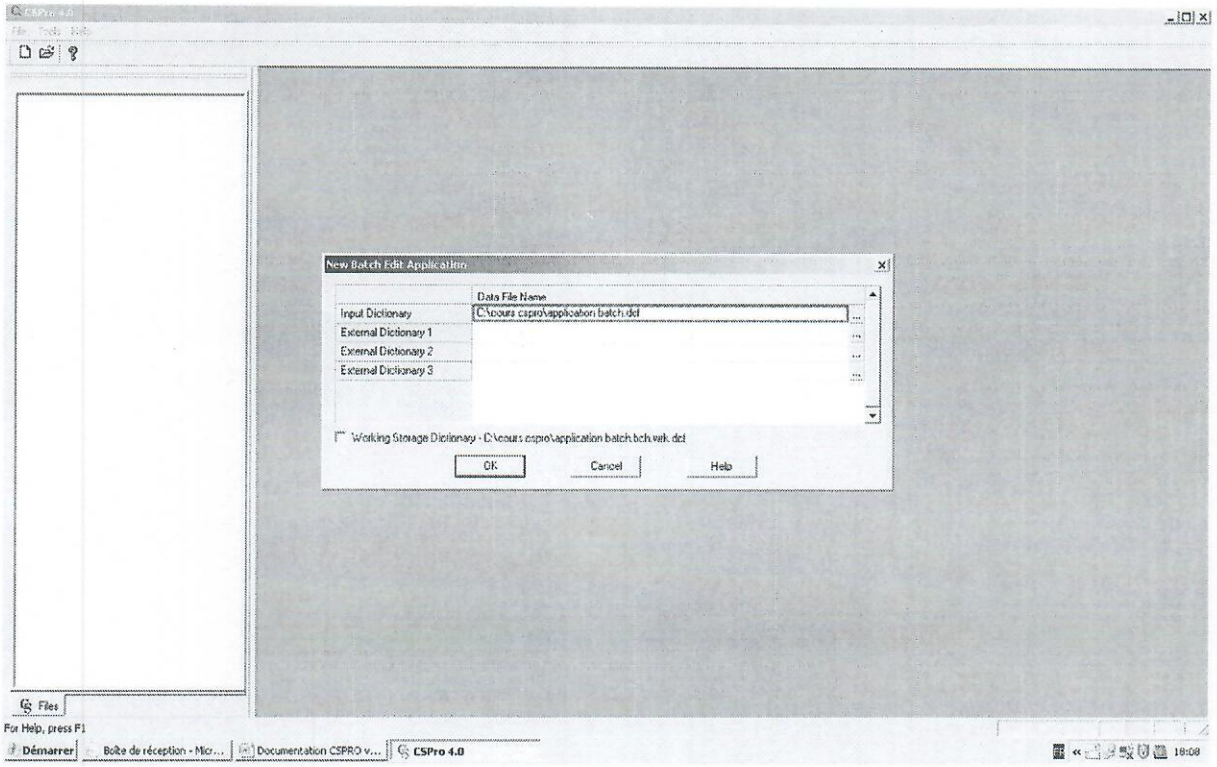
3-3. La réalisation d'une application batch

Une application batch permet de lancer des contrôles après la saisie des données ou de faire des imputations sur les données ainsi que de réaliser des menus de lancement d'applications.

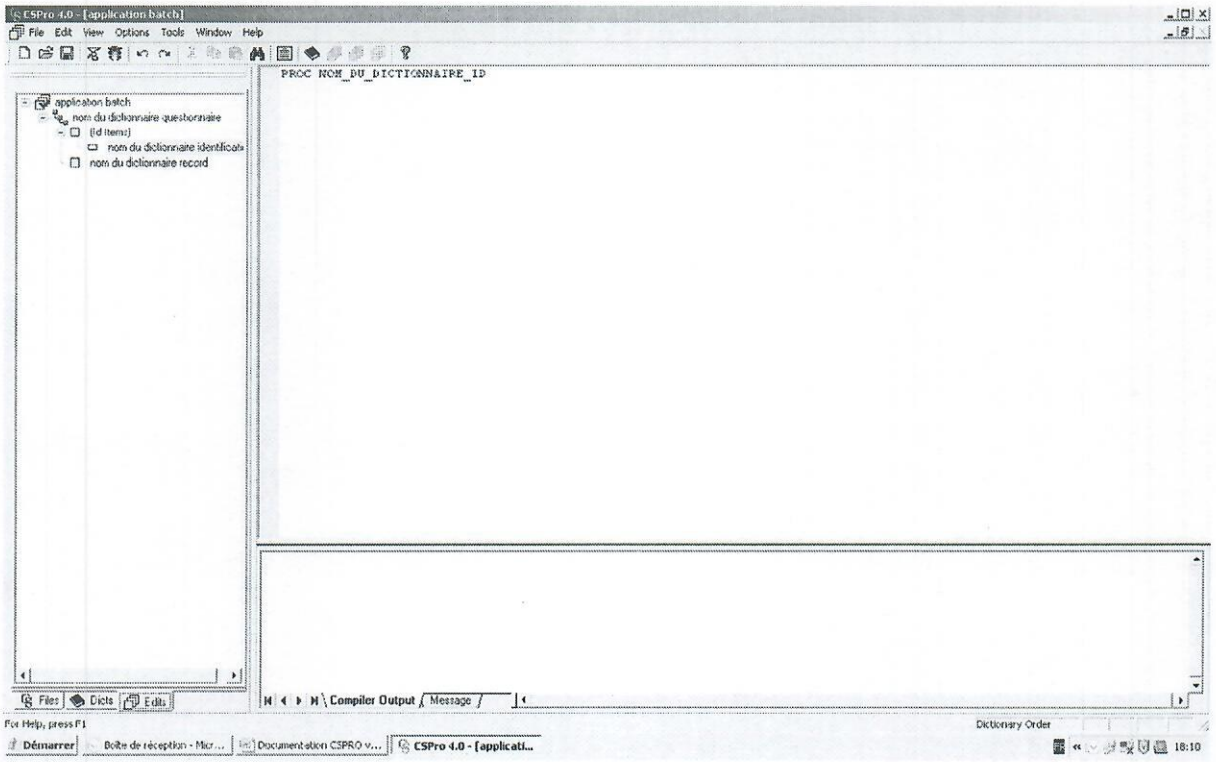
Faire **File/New**, l'écran suivant apparait :



Au niveau de la boîte de dialogue qui s'affiche, choisir comme type d'objet « **Batch Edit Application** ». Puis cliquer sur **ok**. A cette étape, il faut donner le chemin et le nom de l'application de saisie avec l'extension « **.bch** », cliquer sur « **Create** ». L'écran suivant apparait :



Indiquer au niveau de Input Dictionary le chemin et le nom du fichier dictionnaire des données puis cliquer sur « **ok** ».



On peut commencer à introduire les codes commande.

3-4. Les commandes de contrôle CSpro

3-4.1. Généralités.

Le langage de programmation sous CSPRO est assez proche du Pascal. Chaque objet (record, roster, item, etc.) sur lequel on veut écrire une procédure de contrôle doit avoir une **preproc** et/ou une **postproc**. La preproc permet de rédiger les contrôles que l'on souhaite effectuer avant la saisie de toute information concernant l'objet. La preproc est exécutée avant l'apparition du curseur dans le champ. La postproc quant à elle permet de spécifier les actions à faire après la saisie dans le champ.

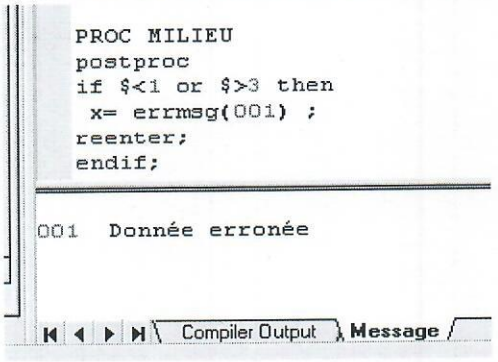
Toute procédure commence par le mot clé PROC suivi du nom de l'objet sur lequel il s'applique. Exemple :

```
PROC AGE
   Préproc
   Instructions A
   Postproc
   Instructions B
```

C'est à travers quelques exemples que l'on va asseoir les notions de programmation sous CSPRO.

Exemple 1 : Voici une procédure qui affiche le message « donnée erronée » et impose une nouvelle saisie si la valeur saisie ne respecte pas la plage (le milieu de résidence est compris entre 1 et 3).

```
PROC MILIEU
    postproc
    if $<1 or $>3 then
      x= errmsg(001) ;
      reenter ;
    endif ;
```



Dans la fenêtre message, taper : **001 Donnée erronée**

Le caractère $ fait référence à l'information qui a été saisie. La fonction **reenter** maintien le curseur dans le champ en cours. On remarque par ailleurs que chaque instruction se termine par un « ; ».

Exemple 2 : traitons d'un cas où on fait un saut à la question S03Q9 si la réponse est 2 (non).

```
PROC S03Q3
    postproc
    if $=2 then
      skip to S03Q9 ;
    endif ;
```

Exemple 3 : Cas de la question Q1 de la sous-section 05.1 du questionnaire de ECAM 2 (Cette sous-section est un record au niveau du dictionnaire). Si le ménage n'a eu aucune naissance (S05Q1=2=) alors on doit changer de section.

```
PROC S05Q1
    postproc
    if $=2 then
      endsect ;
    endif ;
```

Exemple 4 : Cas de la question Q15 du questionnaire de ECAM2. le code de l'équipement correspond au numéro de ligne de l'enregistrement. Pour gagner du temps à la saisie, il sera « pré imprimé ».

```
PROC S05Q1
    preproc
    S07Q15(noccurs(LOGE_B)+1)= noccurs(LOGE_B)+1;
    noinput ;
```

LOGE_B est le nom logique du record correspondant à la sous-section 07.2. l'instruction **noinput** fait passer de la preproc à la postproc. Ceci interdit toute saisie dans le champ concerné.

3-4.2. Section de déclaration (PROC GLOBAL)

Les déclarations et les définitions sont définies dans la procédure globale. Dans cette section vous déclarez le mode de fonctionnement (implicite ou explicite), des variables, des tableaux et des fonctions définies par l'utilisateur. La procédure globale apparaît toujours au début du fichier de logique et commence par la ligne "PROC GLOBAL". À part les fonctions définies par l'utilisateur, il n'y a aucune déclaration exécutable dans cette section. Vous pouvez éditer la section PROC GLOBAL en cliquant sur la plus haute entrée de l'entrée de données édite l'arbre ou le lot édite l'arbre.

Exemple :

```
PROC GLOBAL

 set explicit; {mode}
 numeric x, xage;  (numeric variables)
 alpha flag; (alphanumeric variable)
 array Relly(5); (numeric array)
```

3-4.3. Les fonctions

Des fonctions définies par l'utilisateur sont codées dans la partie de déclaration (PROC GLOBAL) d'une application. Une fois défini, ils peuvent être utilisés n'importe où dans une application. Les fonctions sont utilisées pour exécuter les opérations qui sont utilisées en plusieurs endroits différents dans une application.

Le format de la commande :
Valeur de retour = nom de fonction (liste de paramètre)

Exemple :

```
function InitRellyArray ();  {user-defined function}
  Relly (1) = 3; { enfant du CM }
```

```
  Relly (2) = 4; { parent du CM }
  Relly (3) = 9; { petit-fils du CM }
  Relly (4) = 8; { grandparent du CM }
end;
```

3-4.3. Les opérateurs

Not equal to	<>
Less than	<
Less than or equal to	<=
Greater than	>
Greater than or equal to	>=
In range	in

Operation	Symbol	Keyword
Negation	!	not
Conjunction	&	and
Disjunction	|	or

3-4.4. Liste des commandes

1. **Accept** Rend le numéro d'un choix sur une liste proposée à l'opérateur de saisie de données.

Exemple:

```
PROC UR
preproc
  I = 0;
  do until I in 1:2
    I = accept("Area Designation?", "Urban", "Rural");
  enddo;
  $ = I;
  noinput;
```

2. **Advance** Avance à un champ indiqué durant l'entrée de données.

Format 1:

```
advance [to] field-name;
```

Format 2:

```
advance [to] alpha-variable;
```

3. **Alpha** Déclare des variables alphanumériques utilisées dans l'application.

Format:

```
alpha [(len)] var-1[, var-2[..., var-n]]);
```

Si la valeur de **len** n'est pas précisée, elle prend par défaut la valeur 16.

Exemple 1:

```
PROC GLOBAL
        alpha a,b,c;
        alpha(10) x,y;
```

```
PROC A1
      x = "hi mom";

      x sera égal à "hi mom     "
                     1234567890

      x = "good night, mom";

      x sera égal "good night"
                   1234567890
```

Exemple 2:

```
PROC GLOBAL
      alpha (3) reply;
      alpha flag;

PROC Q5
      if q5 = 1 then
        reply = "Yes";
        flag = "Y";
      endif;
```

4. **Array** Déclare un tableau (ou matrice) de 1 à 3 dimensions.

Format:

```
array [alpha[(len)]] array-name(dim-1[,dim-2[,dim-3]]) [save];
```

L'option **save** permet d'initialiser la matrice avec les données en ligne.

exemple:

```
PROC GLOBAL
  array age_hd (2,8); {sexe par lien de parenté}
```

exemple:

```
PROC MY_PROGRAM
preproc
  homme = 1;
  femme = 2;

  age_hd (homme,1) = 20; { CM homme }
  age_hd (homme,2) = 24; { époux }
  age_hd (homme,3) = 8; { enfant de sexe masculin}
  age_hd (femme,1) = 26; { CM femme }
  age_hd (femme,2) = 32; { épouse }
  age_hd (femme,3) = 5; { enfant de sexe féminin}
```

5. **Break** Sortie d'une boucle, et continue l'exécution après la commande enddo spécifiant la fin de la boucle.

Exemple :

```
PROC QUEST
  spouse = 0;
  for i in PERSON_EDT do
    if relationship = 2 then
      spouse = i;
      break;
    endif;
  enddo;
```

6. **Clear** Initialise les valeurs en mémoire de données définies dans des fichiers externes à zéro ou blanc.

Format:

```
b = clear(ext-dict);
```

7. **Close** Ferme un fichier externe précédemment ouvert.

Format:

```
b = close(ext-dict-name | file-name);
```

8. **Cmcode** Rend le nombre de mois depuis l'année 1900 d'une date donnée en mois et année.

Format:

```
i = cmcode(month,year);
```

Exemple 1:

```
XMONTH = 06;
XYEAR = 81;
DATE = cmcode(XMONTH,XYEAR);
```

La valeur de DATE avec les paramètres donnés [Juin 1981], sera (81 x 12) + 6 = 978.

Exemple 2:

```
XMONTH = 2;
XYEAR = 2000;
DATE = cmcode(XMONTH,XYEAR);
```

La valeur de DATE avec les paramètres donnés [Février 2000], sera ((2000 – 1900) * 12) + 2, ou 1202.

9. **concat** Joint deux ou plusieurs caractères alpha.

Format:

```
s = concat(string-2,string-2[,..,string-n]);
```

Exemple:

```
PROC GLOBAL
  alpha 30 FIRST_NAME, LAST_NAME, FULL_NAME;

PROC ABC
  FIRST_NAME = "John"
  LAST_NAME = "Henry"
  FULL_NAME = concat(strip(FIRST_NAME)," ", strip(LAST_NAME));
```

Le résultat obtenu :

```
  FIRST_NAME = "John "
  LAST_NAME = "Henry "
  FULL_NAME = "John Henry "
```

10. **Count** Rend le nombre d'occurrences d'un tableau d'enregistrements répétés ou bien le nombre de fois que la condition est trouvée.

Format:

```
i = count(multiple-item [where condition]);
```

Exemple:

```
NBR_ENFANT = count(PERSONS where LIEN = 3);
```

11. **Curocc** Rend le numéro de l'enregistrement courant dans un tableau d'enregistrements répétés.

Format:
```
i = curocc([group]);
```

Exemple :

```
PROC RELATION
  if curocc(PERSON_REC) = 1 then
    if (LIEN <> 1) then
      errmsg("La première personne doit être le CM");
    endif;
  endif;
```

12. **Delcase** Marque un enregistrement pour sa suppression dans un fichier externe basé sur une clé.

Format:
```
b = delcase(ext-dict-name[,var-list]);
```

var-list permet de définir la clé

13. **Delete** Supprime un enregistrement ou plusieurs enregistrements répétés dans un questionnaire en cours de lecture.

Format:
```
b = delete(group[occ]);
```

Exemple (enregistrements multiples):
```
do varying i = totocc(PERSON_REC) until i <= 0 by (-1)
  if rel(i) = notappl and
    sex(i) = notappl and
    age(i) = notappl then
    delete (PERSON_REC(i)); {permet d'enlever les enregistrements vides }
  endif;
enddo;
```

14. **Demode** Rend le mode d'entrée de données actuel.

Format:
```
i = demode();
```

Il y a trois modes de saisie de données:

- add, pour entrer de nouveaux enregistrements; la valeur retournée par Demode est '1'
- modify, pour modifier un enregistrement déjà saisi; la valeur retournée par Demode est '2'.
- verify, pour introduire à nouveau les données et verifier qu'il y a pas de différences entre les données de la première et deuxième saisie; la valeur retournée par Demode est '3'.

Exemple:
```
if demode() = add then
  V103 = 3;
```

```
endif;
```

15. **Do** Exécute une ou plusieurs déclarations à plusieurs reprises tandis qu'une condition logique reste vraie ou bien jusqu'à ce qu'une condition logique ne soit plus vraie.

Format:
```
do [[varying] var = expression] while/until condition [by expression]
  instructions;
enddo;
```

Exemple:
```
HEAD = 0;
do varying i = 1 until HEAD > 0 or i > totocc(PERSON)
  if RELATIONSHIP(i) = 1 then
    HEAD = i;
  endif;
enddo;
```

Le même exemple peut être réécrit en utilisant la condition « while» comme indiqué ci-dessous:

```
HEAD = 0;
do varying i = 1 while HEAD = 0 and i <= totocc(PERSON)
  if RELATIONSHIP(i) = 1 then
    HEAD = i;
  endif;
enddo;
```

16. **Edit** Convertit un nombre en caractère.

Format:
```
  s = edit(edit-pattern,numeric-expression);
```
Exemple 1:
```
  X = 87;
  A1 = edit("ZZZ9",X); yields A1 = " 87"
  A2 = edit("9999",X); yields A2 = "0087"
  A3 = edit("Z999",X); yields A3 = " 087"
```

Exemple 2:
```
  Y = 0;
  A4 = edit("ZZ9",Y); yields A4 = " 0"
  A5 = edit("999",Y); yields A5 = "000"
  A6 = edit("ZZZ",Y); yields A6 = " "
```

Exemple 3:
```
  A = edit("99:99:99",sysdate());
```

Exemple 4:
```
  A = edit("99/99/99",sysdate("DDMMYY"));
```

Exemple 5:
```
  A = edit("ZZZ,ZZZ,ZZ9",MONTANT);
```

17. **Endgroup** Finit l'entrée de données pour l'enregistrement courant ou le groupe d'enregistrements.

Exemple:
```
  if KIDSBORN = 0 then
    endgroup;
```

```
endif;
```

18. **Endlevel** Finit l'entrée de données pour le niveau actuel.

Exemple:
```
if MORE_WOMEN = 0 then
  endlevel;
endif;
```

19. **Enter** Permet d'entrer des données à partir d'un autre formulaire.

Format:
```
enter form-file-name
```

20. **Errmsg** Montre ou écrit un message.

Format 1 Exemples:

Exemple 1:
```
errmsg("Le CM est âgé de %d", AGE);
```

Exemple 2:
```
errmsg("Plus d'un CM dans un ménage") denom = PERSON_COUNT summary;
```

Exemple 3:
```
errmsg("Le CM est âgé de %d.  Son âge doit être >= 12", AGE)
```

Format 2 Exemple:

```
OK = errmsg (1,"June"30,31);
```

Le fichier message va contenir le texte suivant:

```
1 %s est de %d jours. Vous avez introduit %d!
```

21. **Exit** Finit une procédure avant que le traitement normal finisse.

Exemple:
```
function FIRST_WOMAN();
  FIRST_WOMAN = 0;
  do i = 1 while i <= HH_MEMBERS
    if SEX(i) = 2 then
      FIRST_WOMAN = i;
      exit;
    endif;
  enddo;
end;
```

22. **File** Déclare un ou plusieurs fichiers utilisés dans l'application.

Exemple:
```
PROC GLOBAL
  File FILE_PERSON, FILE_HOUSEHOLD;
```

23. **fileconcat** Cumule plusieurs fichiers définis dans le même format.

Format:
```
b = fileconcat(result-file-name, file1[, file2[, ...]]);
```

Exemple:

```
fileconcat("c:\prov1\prov1.dat", "c:\prov1\01*.dat");
```

24. **Filecopy** Copie un fichier dans un autre fichier.

Format:

```
b = filecopy(file-name, result-file-name);
```

Exemple:

```
filecopy(DATA, DATACOPY);
```

25. **impute** Assigne une valeur à une variable et enregistre la fréquence d'assignation.

Format:

```
impute (item-name, expression)
    [stat (item-name1, item-name2,....,item-nameN)]
    [title (alpha-expression)]
    [vset (vset-number)]
    [specific];
```

Exemple:

```
impute(P04_AGE, TEMPAGE) title("Age updated via TempAge") vset(2);
```

26. **int** Rend la partie entière d'une expression numérique.

Format:

```
i = int(numeric-expression);
```

Exemple:

```
x = int(5 / 3);
```

La valeur de x sera 1.

27. **invalueset** Détermine si une valeur d'une variable est dans une plage de valeurs.

Format:

```
b = invalueset(item-name[, valueset-name]);
```

Exemple 1:

```
if not invalueset(P03_SEX) then
  errmsg("Sexe est invalide. La valeur est %d", P03_SEX);
endif;
```

28. **Loadcase** Charge un enregistrement d'un fichier externe dans la mémoire à partir de variables servant de clé.

Format:

```
b = loadcase(ext-dict-name[, var-list]);
```

Exemple:

```
OK = loadcase(SAMPDICT,CLUSTER,HH);
```

La fonction retourne la valeur 1 (vrai) si l'enregistrement est trouvé et chargé, 0 (faux) dans le cas contraire.

29. **next** Finit une boucle, une itération et continue l'exécution avec l'itération suivante dans la boucle.

Format:
```
skip [to [next]] field-name;
```

Exemple:
```
if Q202 <> 1 then
  skip to next Q201;
endif;
```

30. **noccurs** Rend le nombre de présences (d'occurrences) pour un formulaire se répétant ou une liste.

Exemple:
```
TOTAL_PERSONS = noccurs(PERSON);
```

31. **noinput** Empêche la saisie pour le champ actuel pendant l'entrée de données.

Exemple:
```
PROC Q102
preproc
 if Q101 <> 1 then
   noinput;
 endif;
```

32. **numeric** Déclare des variables numériques utilisées dans l'application.

```
PROC GLOBAL

  numeric X, Y, male, female;
```

33. **pos** Rend la position d'un caractère dans une série de caractères.

Format:
```
i = pos (substring, source);
```

Exemple 1:
```
X = pos("L","FOR THE CHILDREN");
```

La valeur de X sera 12;

Exemple 2:
```
X = pos("DRE","CHILDREN");
```

La valeur de X sera 5;

Exemple 3:
```
X = pos("DCN","CHILDREN");
```

La valeur de X sera 0.

34. **postproc** Déclare que les instructions qui suivent sont exécutées à la fin d'un bloc.

```
PROC SEX

postproc
 if ($ = 2 and AGE < 5) then
   reenter;
 endif;
```

35. **preproc** Déclare que les instructions qui suivent sont exécutées au début d'un bloc.

```
PROC DATE

 preproc
   DATE = sysdate("DDMMYYYY");
```

36. **Recode** Assigne une valeur à une variable basée sur la valeur d'une ou plusieurs autres variables.

Format:

```
recode var-1 [:var-2 [:var-n]]    => var-out;
      [range-1] [:range-2 [:range-n]]  => exp;
      [range-1] [:range-2 [:range-n]]  => exp;
          : : :
                  [: [:]]    => other-exp;
endrecode;
```

Exemple 1:

```
recode AGE => AGE_GROUP;
      0-19 => 1;
     20-29 => 2;
     30-39 => 3;
     40-49 => 4;
     >= 50 => 5;
           => 9;
endrecode;
```

Exemple 2:

```
recode ATTEND : ED_LEVEL => EDUC;
    2,notappl :          => 1;
            1 : 1        => 2;
            1 : 2,3      => 3;
              :          => 9;
endrecode;
```

37. **reenter** Force l'agent de saisie à ré-entrer le champ actuel ou précédent.

Format :

```
reenter [field-name];
```

Exemple:

```
if KIDS = 1 & BOYS = 0 & GIRLS = 0 then
  reenter KIDS;
endif;
```

38. **setfile** Assigne un fichier de données à un dictionnaire ou à un fichier déclaré.

Format:

```
b = setfile(ext-dict-name | file-name, alpha-exp
            [, update | append | create]);
```

Si les options update, append ou create qui sont optionnels, ne sont citées, le fichier est ouvert en mode update.

Exemple 1:

```
OK = setfile(LOOKUP, "c:\My Lookup File.dat");
```

Exemple 2:

```
OK = setfile(REPORT, REPORT_FILE_NAME, create);
```

39. **skip** Permet de sauter jusqu'à la variable spécifiée au moment de la saisie.

Format :

```
skip [to [next]] field-name;
```

Exemple :

```
if Q305 <> 2 then
  skip to Q307;
endif;
```

40. **soccurs** Retourne le nombre d'occurrences d'un enregistrement.

Format:

```
i = soccurs(record-name);
```

Exemple:

```
NUM_HH_MEMBERS = soccurs(PERSON_REC);
```

41. **sort** Fait le tri des occurrences d'un enregistrement avec comme critère la valeur d'une variable.

Format:

```
b = sort(group using item);
```

Exemple:

```
Sort(PERSON using LINE_NUM);
```

42. **special** Détermine si la valeur d'une variable est MISSING, NOTAPPL, or DEFAULT.

Format:

```
b = special(numeric-exp);
```

le *numeric-exp* peut être une variable, un champ ou une expression numérique.

43. **sum** Retourne la somme d'une variable répétée plusieurs fois.

Format:

```
d = sum(multiple-item [where condition]);
```

Exemple:

```
TOTAL_INCOME = sum(INCOME);
TOTAL_FEMALE_INCOME = sum(INCOME where SEX = 2);
```

44. **sysdate** Retourne la date du système comme un entier.

Format:

```
i = sysdate([date-format]);
```

[] indique que cette partie est optionnelle.

La date courante sera retournée comme une chaine de caractères en utilisant la fonction edit comme suit:

```
edit("99/99/99",sysdate("DDMMYY"));
```

Exemple :

Si la date courante est 17 Décembre 1999, on aura les résultats suivants:

49. **write** Ecrit dans un fichier texte.

Format:

```
[b =] write(alpha-exp[,p1[,p2[,...,pn]]]);
```

In the string expression

%[n]d = insérer un nombre et l'afficher comme un entier
%[n.d]f = insérer un nombre et l'afficher comme une valeur décimale
%[n.d]s = insérer du caractère texte

"n" est la taille du champ et "d" représente la valeur à afficher.

Exemple:

```
write("Sex = %d", SEX);
```

50. **writecase** Ecrit un enregistrement de la mémoire dans un fichier externe.

Format:

```
b = writecase(ext-dict-name[,var-list]);
```

Exemple:

```
OK = writecase(KIDS,CLUSNUM,HHNUM,LINE);
```

3-5. Création de l'icône de saisie

Sur le bureau Windows,

Faites apparaître le menu contextuel en cliquant sur la touche droite de votre souris. Choisissez "**Nouveau**" et ensuite "**Raccourci**".

Il vous est alors demandé d'indiquer la ligne de commande. Faites **Parcourir** pour aller chercher l'application par exemple phase1.ent dans le répertoire de travail

Il vous est alors demandé le nom de l'icône. Tapez "**nom du programme de saisie(Saisie Phase 1)**" et validez.

L'icône suivant apparaît sur le bureau. Il suffira plus tard de cliquer dessus pour lancer la saisie.

Saisie Phase 1.lnk

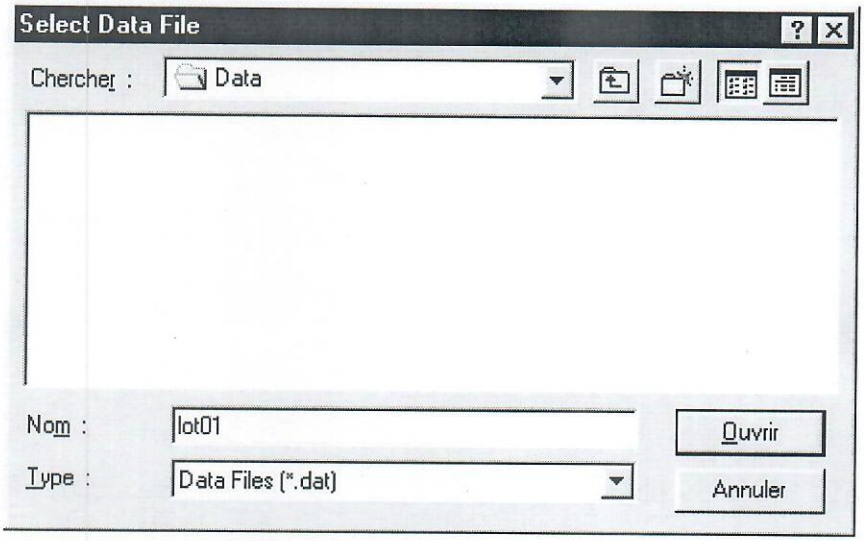
3-6. Instructions de démarrage de la saisie

- Le lancement du programme de saisie de la phase 1 se fait en double-cliquant sur l'icône **[Saisie Phase1]** se trouvant sur votre bureau.
- Cliquer ensuite sur **le feu vert** situé sur la barre de menu, pour lancer l'application de saisie.
- Ici, il faut sélectionner le fichier des données. Pour la première fois, saisir le nom du fichier. Pour les prochaines fois, il suffira de cliquer sur **Ouvrir**.

Nom des fichiers de données et emplacement.

Le fichier de données, par exemple LOT01, doit se placer dans le dossier créé à cet effet, C:\\Data

Sélectionnez le et indiquez le nom du fichier de données ; l'écran aura ensuite l'aspect suivant :



Faites **Ouvrir**.

Pour la première fois, répondez **Oui** à la question "LOT01 - ce fichier n'existe pas, souhaitez-vous le créer ?".

Operator ID : Il s'agit d'un code enquêteur nécessaire au suivi des évènements pouvant se produire lors de la saisie. Mettez votre code agent, à 2 digits, qui vous a été attribué par l'équipe centrale d'encadrement. Exemple : "11".

Chaque personne intervenant sur l'enquête doit être identifiable (agent de saisie, contrôleur de saisie, superviseur, cadre chargé du traitement). A cet effet, l'équipe centrale d'encadrement devra affecter à chacun des intervenants un code agent.

Après avoir mis le code opérateur, cliquez sur **OK.**

Ensuite, vous avez à choisir entre trois modes opératoires: add case, modify case et Verify.

Le mode "**add case**" est celui qu'il faut choisir pour un nouvel enregistrement.
Tandis que si vous devez modifier un enregistrement préalablement saisi, vous devez passer en mode "**modify case**".
Le mode "**verify case**" est à utiliser dans le cas d'une double saisie pour la vérification de la saisie. Nous n'insistons pas ici sur ce dernier mode.

4. Traitement des données

4-1.Le tri à plat ou fréquences des variables

1. Lancer CSPro
2. Aller dans TOOLS
3. Choisir Tabulate Frequencies
4. Sélectionner le dictionnaire de données de l'application au niveau de son répertoire
5. Cliquer sur Open

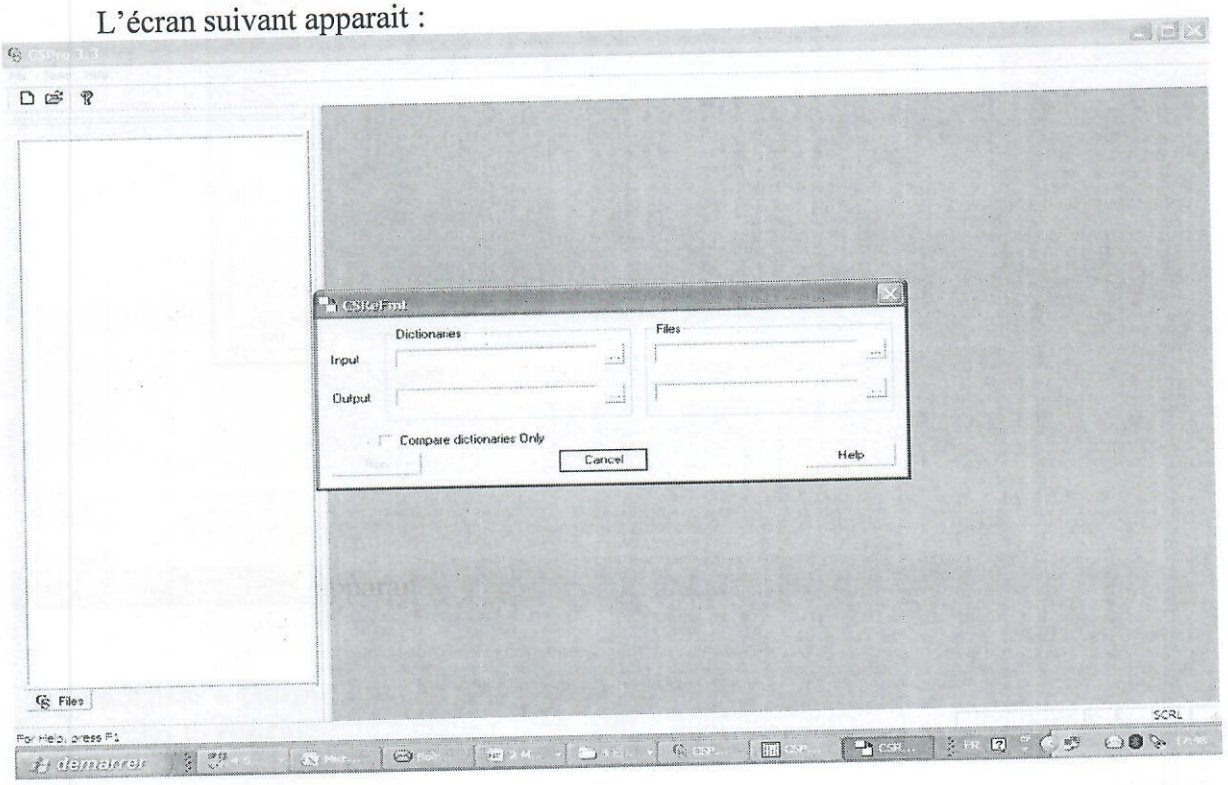
6. Sélectionner les variables sur lesquelles on veut obtenir les fréquences
7. Cliquer sur le feu vert
8. Sélectionner le nom du fichier de données au niveau de son répertoire
9. Cliquer sur Open
10. Le résultat apparait à l'écran

4-2.Le reformatage des données

Le principe est le suivant :
Une opération de saisie a déjà débuté et le dictionnaire de données a évolué, donc il faut recadrer les données déjà saisies sur le nouveau dictionnaire.

1. Lancer CSPro
2. Aller dans TOOLS
3. Lancer Reformat Data

L'écran suivant apparait :



Introduire à gauche dans **Dictionaries** au niveau de **Input** le nom du premier dictionnaire et dans **Output** le nom du dictionnaire modifié.
A gauche de l'écran dans **Files**, introduire le nom du premier fichier dans **Input** et dans **Ouput** le nom du fichier où seront transférées les données dans la nouvelle structure et cliquer sur **Run**. Continuer alors à travailler sur ce nouveau fichier obtenu correspondant à la nouvelle structure du dictionnaire.
Ne pas oublier que tous ces fichiers de données et dictionnaires sont logés dans des répertoires qu'il faut spécifier.

4-3.La concaténation de fichiers de données

On veut fusionner plusieurs fichiers de données saisies dans la même application en un seul.