Une gestion paramétrable des éditions iSeries

- Les articles -

Date de mise en ligne : mardi 10 janvier 2006

Description :

Cet article a pour objectif de vous apporter des éléments de réflexion sur la manière de paramétrer les associations programmes-imprimantes.

Environnement iSeries

Comment paramétrer les associations entre programmes et imprimantes, dans un contexte applicatif donné ? Si vous ne disposez d'aucun outil en ce sens, il est essentiel de constituer cette brique applicative, qui vous garantira pérennité pour vos applications, simplicité de mise en oeuvre, et reconnaissance de la part de votre équipe exploitation. Si la solution présentée ici est développée en Adelia, il faut souligner qu'elle est tout à fait transposable au développement RPG.

Si vous développez des applications AS/400 (pardon "iSeries") pour votre société, vous vous êtes vraisemblablement déjà heurté au problème du paramétrage des éditions. Vous aviez développé un programme d'édition tout simple, mais très vite, vous vous êtes aperçus que vous aviez besoin d'éditer ce document tantôt sur une imprimante, tantôt sur une autre, en fonction du destinataire du document concerné. Dans l'urgence, et pour répondre rapidement à ce besoin, vous avez alors écrit des bidouilles dans vos programmes CL, du genre (je vous le fais façon "algo" car j'ai la flemme de vous l'écrire en CL) :

```
SI
 OVRPRTF vers <l'imprimante de machin>
 SINON
 SI
 OVRPRTF vers <l'imprimante de bidule>
 SINON
 ETC...
 FIN
 FIN
```

L'horreur totale...

Pire, on vous demande par la suite d'installer votre application sur un autre AS/400, sur un site distant, et vous repartez pour un tour, en ajoutant cette fois des tests supplémentaires par rapport à une notion de société, ou que sais-je encore.

Bon, j'arrête là, voyons maintenant comment on peut traiter intelligemment nos éditions "iSeries" (ça y est, j'l'ai dit !). Attention, la solution que je vais vous proposer n'est pas la meilleure, la seule et l'unique. Elle n'a pour autre vocation que de vous apporter des idées, pour vous aider à créer ou améliorer vos propres outils. Cette méthode est pratiquée par différentes sociétés sous une forme plus ou moins élaborée. Elle s'est révélée bien adaptée au contexte applicatif dans lequel j'évolue depuis quelques temps, à vous de juger si elle peut s'adapter au vôtre.

Pour bien comprendre dans quel esprit cet outil a été développé, il faut savoir que nous avions besoin de développer un outil de gestion des impressions dans un contexte multi sociétés, multi-établissements, et multi-dépôts. C'est la raison pour laquelle vous trouverez ces notions (sociétés, établissements et dépôts) dans les fichiers et les programmes présentés par la suite. Si vous travaillez dans un contexte mono-site, vous ne verrez peut être pas l'utilité de prévoir un paramétrage des éditions multi-sites. Je vous recommande tout de même d'y réfléchir, et d'essayer d'anticiper autant que possible sur une évolution possible de votre société, pour ne pas vous retrouver obligés de tout "casser" par la suite.

Pour pouvoir faire de la bonne cuisine, il nous faut les ingrédients suivants :

- un fichier des maquettes prénommé MAQPGM
- quelques tables paramètres contenant la liste des Outq, les différentes qualités d'impression, densités d'impression, etc...
- un programme de gestion du fichier MAQPGM (avec sélection, suppression, etc...), et un programme de mise à jour de ce même fichier
- une règle de gestion prénommée MAQ_OVRPRTF, qui sera utilisée dans tous les programmes batch d'édition
- un programme batch, utilisé exclusivement par la règle de gestion MAQ_OVRPRTF, et destiné à créer la commande d'OVRPRTF. Ce programme s'appellera MMAQ3B.

Vous l'avez compris, la pièce maîtresse du puzzle est la règle de gestion MAQ_OVRPRTF (et son petit copain le programme MMAQ3B). Grâce à elle, vous n'aurez plus besoin de faire des OVRPRTF dans les programmes CL en amont de vos programmes d'édition.

Vous trouverez son source, ainsi que tous les autres, dans un fichier texte joint à cet article. Mais je vous donne ci-dessous la documentation de cette règle de gestion, qui devrait vous éclairer sur la manière dont on peut l'utiliser.

```
4 exemples d'appel de la règle de gestion :
 ==========================================
 exemple 1
 INSERER_RG MAQ_OVRPRTF(*NOM_PGM;0;0;'*ETABUSER';
 ' ';0;' ';*)

 exemple 2
 INSERER_RG MAQ_OVRPRTF(*NOM_PGM;0;0;'*USER';
 ' ';2;'PRT07';W_OVR)

 exemple 3
 INSERER_RG MAQ_OVRPRTF(*NOM_PGM;W_COD_ETA;W_COD_DEP;' ';
 ' ';2;'PRT07';*)

 exemple 4
 INSERER_RG MAQ_OVRPRTF(*NOM_PGM;W_COD_ETA;W_COD_DEP;'*USER';
 ' ';2;'PRT07';*)
```

Explications :

N.B. : dans les exemples expliqués ci-dessous, vous allez entendre parler d'établissements et de dépôts, associés ou non à un profil utilisateur. Bien évidemment, ces notions d'établissement et de dépôt sont spécifiques à un contexte applicatif qui n'est peut être pas le vôtre. Retenez simplement l'idée et voyez si vous pouvez l'adapter à vos besoins.

L'exemple 1 est préconisé dans le cas où l'écran de lancement du traitement ne permet pas de saisir le code établissement et le code dépôt. Dans ce cas on recherchera le code établissement et le code dépôt associé à l'utilisateur .

L'exemple 2 est préconisé si la notion d'établissement et la notion de dépôt n'ont aucune utilité pour le traitement considéré.

L'exemple 3 est à utiliser de préférence si l'écran de lancement du traitement permet de saisir le code établissement et le code dépôt. On préfèrera cependant utiliser l'exemple 4 qui offre un niveau de recherche plus large.

L'exemple 4 est à utiliser de préférence si l'écran de lancement du traitement permet de saisir le code établissement et le code dépôt. Le paramètre "*USER" permettra cependant de faire une recherche préalable au niveau de l'utilisateur avant de rechercher les paramètres liés à l'établissement et au dépôt considéré. Ce type d'utilisation pourra aussi être pratiquée au sein d'un programme traitant alternativement plusieurs dépôts. La RG sera alors placée avant l'ordre d'ouverture du PRTF, dans une rupture entête sur code dépôt.

Liste des paramètres :

1. code maquette : s'il est passé à blanc, il est alimenté automatiquement avec *NOM_PGM, peut également être alimenté avec *NOM_PGM ou forcé avec un autre nom de programme (cas d'un programme batch utilisant plusieurs maquettes).
2. code établissement : il peut être forcé avec un code établissement ou passé à zéro
3. code dépôt : il peut être forcé avec un code dépôt ou passé à zéro
4. code utilisateur : il peut être passé à blanc ou forcé avec les valeurs suivantes (valeurs obligatoirement saisies entre cotes, à l'exception de *USER qui peut être saisi sans cotes car c'est un mot clé reconnu par ADELIA)

```
 '*USER     ' : alimenté avec le profil de l'utilisateur par défaut
 '*NOUSER   ' : pas de recherche sur le profil utilisateur
 '*ETABUSER ' : utilisation de la règle RECUP_ETAB_USER pour récupérer l'établissement et le dépôt de
l'utilisateur et les substituer aux valeurs des paramètres 2 et 3
 ' ' : équivaut à *NOUSER si à blanc
```

1. code libre : il peut être passé à blanc ou forcé avec une valeur quelconque (adresse logistique, tournée de livraison, etc...) pour traiter des cas particuliers non prévus dans le paramétrage standard
2. nombre d'exemplaires forcé : permet de substituer un nombre d'exemplaires différent de celui paramétré dans le fichier des maquettes. Si ce paramètre est à zéro, c'est le paramètre du fichier de maquette qui est pris en compte.
3. imprimante forcée : permet de substituer une imprimante différente de celle définie dans le fichier des maquettes. Si ce paramètre est à blanc, c'est le paramètre du fichier de maquette qui est pris en compte.
4. overflow : peut être passé avec un astérisque si le programme utilisant la règle de gestion gère ses sauts de page en automatique en tenant compte du paramétrage du fichier de maquette. Par contre, si le programme nécessite une gestion de sauts de page manuelle, il pourra utiliser le paramètre 8 retourné par la règle de gestion. Attention : si le fichier des maquettes n'est pas correctement paramétré alors le paramètre 8 peut être retourné à zéro. Le développeur devra apprécier quelle valeur attribuer par défaut dans ce cas.

PRTF en ouverture manuelle

Vous l'aurez peut être deviné en lisant les lignes qui précèdent, avec la règle de gestion MAQ_OVRPTRF, l'OVRPRTF ne se fait pas en amont de l'appel du programme d'édition, mais à l'intérieur même du programme d'édition. Or, vous le savez sans doute, pour que l'OVRPRTF soit pris en compte, il faut qu'il soit effectué avant l'ouverture du PRTF concerné. Donc il faut que le PRTF soit paramétré en "ouverture manuelle" au moment de la compilation du programme. Si vous ne l'avez jamais fait, sachez que vous devez demander l'accès aux paramètres de compilation, et presser la touche F18 pour accéder aux paramètres supplémentaires. Enfin, vous indiquez un "O" dans la colonne "Ouv", en face du programme, comme dans l'exemple ci-dessous) :

```
Fichier     Désignation                    Ouv.
 **AADXXL    Liste des produits catalogue     O
```

A l'intérieur du programme d'édition, on aura donc successivement 2 lignes de code :

- une ligne pour appeler la règle de gestion MAQ_OVRPTRF avec les paramètres adéquats,
- une ligne pour demander l'ouverture du PRTF concerné avec l'ordre Adelia "OUVRIR".

Si un peu plus tard, dans le même programme, vous avez besoin de modifier l'OVRPRTF, il vous suffira de fermer le PRTF avec l'ordre Adelia "FERMER", puis de faire un nouvel appel à la règle de gestion MAQ_OVRPRTF, suivi d'un nouvel ordre d'ouverture, et le tour est joué. J'ai notamment exploité cette possibilité au travers de ruptures, basées par exemple sur la notion d'adresse logistique. Chaque fois que mon programme détectait un changement d'adresse logistique, il refermait le PRTF, effectuait une nouvelle substitution (OVRPRTF), et une réouverture du PRTF. En passant cette notion d'adresse logistique dans le paramètre "code libre" de la règle de gestion, je pouvais ainsi très facilement paramétrer, et ventiler, les spoules sur les bonnes imprimantes. Bref, je pilotais complètement mon processus d'édition, sans jamais avoir besoin de "sortir" de mon programme batch, difficile de faire plus confortable.

Peut être mes explications vous paraissent-elles un peu confuses, aussi je vous donne ci-dessous un petit exemple, qui consiste à lire un fichier de commandes clients (fichier CDECLI), et à chaque changement de dépôt, à orienter l'impression vers l'imprimante adéquate. On considèrera que le fichier CDECLI est trié sur les zones suivantes :

- KW_COD_ETA : code établissement
- KW_COD_DEP : code dépôt

Ces zones serviront à déclencher la rupture et donc l'orientation du spoule sur l'imprimante correspondante (cf. rupture CHG_DEPOT dans l'exemple ci-dessous).

Code source du programme exemple KRX01B :

```
RECEVOIR ...paramètres du programme...
 *
 ... à compléter...
 *
 POSITIONNER_AV  CDECLI
 LIRE_AVANT      CDECLI
 TANT_QUE        CDECLI EXISTE
 TRAITER_CHG    CHG_DEPOT
 TRAITER_PROC   EDICDE
 LIRE_AVANT     CDECLI
 TRAITER_TOTAL  CHG_DEPOT
 REFAIRE
 *
 ... à compléter ...
 *
 TERMINER
 ************************
 TRT_CHANGEMENT CHG_DEPOT
 *
 INSERER_RG MAQ_OVRPRTF(*NOM_PGM; KW_COD_ETA; KW_COD_DEP; *USER; ' '; 0; ' '; *)
 OUVRIR KRX01B
 *
 FIN_TRAITEMENT
 ************************
 DEBUT_PROCEDURE EDICDE
 *
 *-- Procédure d'édition d'une commande
 ... à compléter ...
 *
 FIN_PROCEDURE
 ************************
```

```
TRT_TOTAL CHG_DEPOT
*
FERMER KRX01B
*
FIN_TRAITEMENT
```

J'espère que vous voyez maintenant l'intérêt de cette solution, car en ce qui me concerne, elle a apporté à mon équipe et à moi même une grande souplesse de développement, et nous a permis de répondre à toutes les demandes, des plus simples aux plus complexes.

Pour clarifier davantage mon propos, il faut savoir que chaque programme batch d'édition est déclaré impérativement dans le fichier MAQPGM (règle que j'ai imposée au travers de nos normes et standards). Il sera déclaré systématiquement au niveau société, mais il sera possible de le déclarer ausi à un niveau établissement, voire dépôt, pour affiner le paramétrage des files d'attente d'impression associées (OutQueue) aux différents sites de production. Exemple :

programme	Soc	Etab	Dépôt	OutQueue
KRX01B	XXX			IMP_PRT01
KRX01B	XXX	1		IMP_PRT02
KRX01B	XXX	1	2	IPT_PRT08
KRX01B	XXX	2	3	IPT_PRT10

Donc au moment d'associer le PRTF à une imprimante, le programme KRX01B va déterminer sur quel dépôt il "travaille", et appeler la règle de gestion MAQ_OVRPRTF. Cette dernière va rechercher le paramétrage associé à ce dépôt, et si elle ne le trouve pas, elle recherchera le paramétrage associé au niveau supérieur (soit le niveau "établissement"), et si elle ne le trouve pas, elle va le rechercher au niveau le plus élevé (soit le niveau "société"). Bien sûr, c'est un peu plus compliqué que ça, puisque le fichier MAQPGM contient dans sa clé, en plus des sociétés, établissements et dépôts, le code utilisateur et une zone intitulée "code libre". Mais je pense que vous avez compris l'idée générale, la lecture du code source de la règle de gestion complètera votre connaissance du système.

Le fichier texte zippé joint à cet article contient les éléments suivants :

- le source de la règle de gestion MAQ_OVRPRTF (avec une partie des commentaires que vous trouvez déjà dans cet article)
- le source du pgm MMAQ3B servant à concaténer la commande d'OVRPRTF.
- la structure du fichier MAQPGM, ainsi que la structure des fichiers indexs associés
- la structure du fichier IMPRIM, donnée ici à titre indicatif, est qui le fichier des imprimantes que l'on peut associer à chaque enregistrement du fichier MAQPGM. Ce fichier est très basique, pas grand chose à en dire.
- la maquette du programme MIM10S, programme de gestion des enregistrements du fichier MAQPGM : donnée ici à titre indicatif, elle a pour but de vous donner une idée de la manière dont vous pouvez écrire votre propre programme de gestion du fichier MAQPGM.
- la maquette du programme MIM11M, programme de mise à jour des enregistrements du fichier MAQPGM : donnée ici à titre indicatif, elle a pour but de vous donner une idée de la manière dont vous pouvez écrire votre propre programme de mise à jour du fichier MAQPGM.
- la liste des tables paramètres utilisées par le programme de mise à jour du fichier MAQPGM (programme MIM11M).

Bonne lecture, et bon courage.

Post-scriptum :

L'auteur de cet article insiste sur la nécessité de bien comprendre la philosophie du module présenté ici, avant de chercher à l'appliquer à votre propre organisation. Cet article constitue un ensemble de pistes, à suivre ou non, selon votre contexte organisationnel et technique.