

Utilisation de SAS dans un environnement intranet multiplateforme

- Club SAS de Québec, 16 novembre 2011



*Affaires municipales,
Régions et Occupation
du territoire*

Québec



Plan de match



- Contexte d'ensemble
- Choix de l'outil de diffusion
- Exemple concret d'intégration du serveur SAS
 - Codes et cas pratique
- Questions

Ministère des Affaires municipales, des Régions et de l'Occupation du territoire

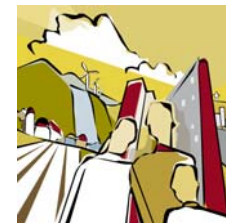


- Mission : appuyer l'administration et le développement des municipalités, des régions et de la métropole en favorisant une approche durable et intégrée pour le bénéfice des citoyens
- Environ 500 employés
 - Québec, Montréal et en région
- Plus de 1200 partenaires
 - Municipalités, MRC, conférences régionales des élus, ...

*Affaires municipales,
Régions et Occupation
du territoire*

Québec 

Quelques domaines d'affaires

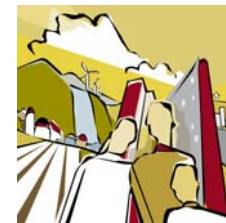


- Organisation municipale
- Infrastructures
- Évaluation foncière
- Aménagement du territoire
- Finances municipales
- Développement régional et rural

*Affaires municipales,
Régions et Occupation
du territoire*

Québec 

À l'été 2009



Révision de l'offre service de notre direction

1. Améliorer la connaissance du territoire
2. Partager l'information à l'intérieur du Ministère
3. Mixer la géomatique et la statistique

❖ Optimiser l'utilisation du serveur SAS

*Affaires municipales,
Régions et Occupation
du territoire*

Québec 

Premiers constats sur le serveur SAS



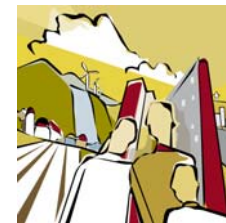
- Peu utilisé
 - Quelques personnes formées avec EG 4.1 ou SAS 9.1 local
- Administré à l'extérieur du Ministère
- Pas de ressources pour le développement
- « Entrepôt de données » plus ou moins tenu à jour

D'où viennent les données, l'information?



- Serveurs de bases de données Oracle
 - Données « classiques » et à références spatiales
- Environnement de diffusion géomatique ArcGIS Server
- « Entrepôt de données » sur le serveur SAS
- Données dans des fichiers sur le réseau
 - Structurées (ex. .csv)
 - Formatées pour impression

Contraintes dans le choix d'un outil de diffusion



- Limiter les coûts au minimum
 - Créer à l'interne, mais sans équipe informatique
 - Utiliser au maximum l'environnement en place
- Ne pas forcer l'installation de quoi que ce soit sur les postes
 - Tenir compte des règles de sécurité en place
- Intégrer la cartographie et les représentations classiques de données
- Fournir rapidement une application efficiente

Choix de l'outil de diffusion



*Affaires municipales,
Régions et Occupation
du territoire*

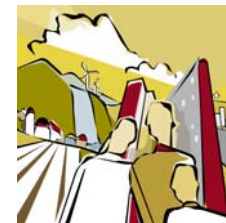
Québec 

Outil choisi



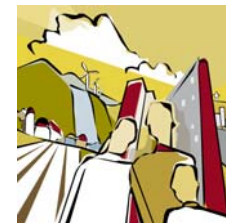
- Solution Web en Flash
 - Un gabarit d'application modulable
 - API Flex contenant des composants graphiques et tableaux
 - API d'ESRI pour se connecter à des services cartographiques
- Aucun pilote pour se connecter à nos formats de BD
- Chargement de fichiers « texte » structurés (ex. XML)
- Connexion à des services Web pour l'accès à des bases de données et des traitements spécifiques

Service Web



- Série de fonctions diffusées par un serveur Web
 - Permettre des calculs
 - Extraire des jeux de données
 - Envoi d'une requête SQL à une BD Oracle
 - Soumettre un programme SAS avec des paramètres à un SAS Stored Process Logical Server
 - Soumettre du code SAS à un SAS Workspace Logical Server
 - ...
- Paramètres à l'entrée et objet à la sortie en option

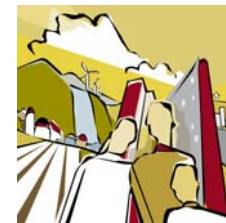
Exemple de code SAS pour utilisation dans un service Web



```
*ProcessBody;          /*Commentaire pour utiliser en service Web*/
*%let code=22015; /*Code géo. pour les essais*/

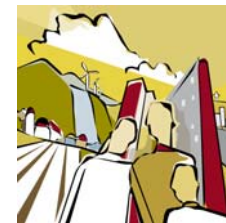
libname lib oracle user=usager password=*** path=chemin;
proc sql;
  create table temp as
    select distinct a.AN, a.CODE_GEO, a.POSTE, b.DESC, a.VALEUR
    from lib.table1 a join lib.table2 b on a.POSTE = b.POSTE
    where a.CODE_GEO = "&code." and a.AN = "2004";
quit;
...
data PF_DETAIL;
  set PF_DETAIL_2004 PF_DETAIL_2005 PF_DETAIL_2006;
run;
```

Comment ça marche dans le programme à la source du service Web



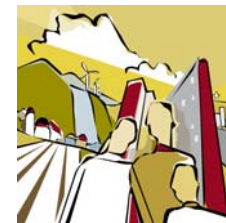
- Déclaration des librairies SAS
- Paramètres d'ouverture d'une session sur le serveur SAS
- Prévoir les paramètres fournis par l'utilisateur
- Préparation du message
 - Identification du programme et valeur des variables Macro
 - Construction des instructions SAS

Comment ça marche dans le programme à la source du service Web



- Envoi du message au serveur SAS
- Traitement des résultats
- Assignment d'une valeur de retour à la fonction
- Fermeture de la session sur le serveur SAS

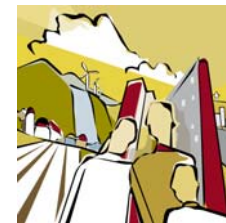
Exemple de code pour utiliser un programme SAS comme service Web



```
1. Function Get_C03_PF (byVal code as String) as DataSet
2. Dim obSAS As SAS.Workspace = GetSAS()
3. obSAS.LanguageService.StoredProcessService.Repository
   = "file:F:\WebServices"
4. obSAS.LanguageService.StoredProcessService.Execute
   ("C03_PF", "code=" & code)
5. Dim obAdapter As New OleDbDataAdapter("select * from " &
   "pf_detail", "provider=sas.iomprovider.1; SAS Workspace
   ID=" & obSAS.UniqueIdentifier)
6. obAdapter.Fill(Get_C03_PF, "pf_detail")
7. DoneWithSAS(obSAS)
```

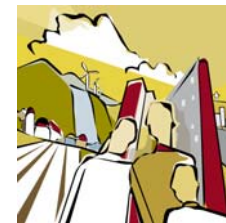
GetSAS() et DoneWithSAS() sont des fonctions définies préalablement qui ouvrent et ferment la session sur le serveur SAS

Comment ça marche dans l'application qui utilise le service Web



- Préparation du message
 - Identification du service et spécification des paramètres
- Envoi du message
- Écoute de la réponse
 - Traitement asynchrone
- Traitement de la réponse
 - Message xml structuré

Exemple de code dans l'application



- Déclaration du service

```
<mx:WebService id="wServicePF"
  wsdl="http://serveur/WebService/PFws.asmx?wsdl">

  <mx:operation name="Get_C03_PF" fault="erreur(event)"
    result="munPF(event)">

    <mx:request xmlns="">

      <code>01023</code>

    </mx:request>

  </mx:operation>

</mx:WebService>
```

- Appel du service

```
wServicePF.Get_C03_PF.request.code = CO_GEO;
wServicePF.Get_C03_PF.send();
```

Exemple de code dans l'application



- Traitement du résultat

- Succès

```
public function munPF(e:ResultEvent):void  
{pfAC = e.result.Tables.pf_detail.Rows;}
```

- Échec

```
public function erreur(e:FaultEvent):void  
{Alert.show(e.fault.faultString);}
```

Avantages des services Web



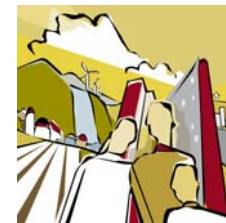
- Récupération des services dans plusieurs applications
 - Structure des messages et des réponses peut être diffusée
 - Structure construite selon des normes
- Traitement asynchrone permet de continuer à utiliser l'application
- Traitement effectué sur le serveur et non sur le poste client
- Pas nécessaire de publier de Stored Process
- Rend des fonctionnalités SAS accessibles à l'extérieur de SAS

Inconvénients des services Web



- Traitement asynchrone force la gestion des événements
 - Les résultats de plusieurs requêtes lancées une après l'autre dans un bloc n'arrivent pas nécessairement dans l'ordre
 - La réponse peut ne jamais arriver...
- Le développement des services demande la connaissance d'un autre langage, ex. : Java ou .Net
- Les formats retournés ne sont pas toujours ceux attendus
 - Une date Oracle est une date
 - Une date SAS est un nombre

Possibilités



- Pour l'instant, les services qu'on utilise ne retournent que des jeux de données
- Rien n'empêche de créer des services qui :
 - Ouvrent des sessions
 - Lancent des programmes
 - Retournent des identifiants de sessions à l'utilisateur qui peut alors accéder aux objets SAS dans un Workspace

Merci!

Des questions?



*Affaires municipales,
Régions et Occupation
du territoire*

Québec 