Journal de l'OSGeo

Le Journal de la Fondation Open Source Geospatial Volume 1 / Mai 2007

Dans ce volume

OSGeo
Your Open Source Compass

FOSS4G - Ouverture des Inscriptions à la Conférence

Nous sommes heureux de vous annoncer l'ouverture des inscriptions en ligne à la Conférence Free and Open Source Software for Geospatial 2007 (FOSS4G 2007). FOSS4G est l'évènement annuel qui réunit les personnes et les sociétés qui créent, utilisent, et gèrent des logiciels géospatiaux open source. Inscrivez-vous dès maintenant en ligne.[1]

Inscrivez-vous avant la date limite du 27 Juillet, pour économiser sur les frais d'inscription ! Tirez profit de l'opportunité que FOSS4G 2007 vous offre, de construire un réseau avec les autres professionnels des données géospatiales, de renouveler d'anciennes relations, et d'en créer de nouvelles.

Pour les dernières mises à jour, l'inscription et/ou la soumission d'une présentation, visitez le site web de la conférence.[2]

OPPORTUNITES D'EXPOSITION & DE SPONSORING

Concernant les opportunités d'exposition et de sponsoring, lisez la page des partenaires [3] ou contactez Paul Ramsey, Président de la Conférence par email.[4]

SOUMETTRE UNE PRESENTATION

Vous pouvez soumettre une présentation en ligne.[5] La date limite pour les soumissions est le 29 Juin 2007.

Les présentations FOSS4G durent 25 minutes, avec 5 minutes de questions/réponses à la fin. Les présentations concernent l'utilisation ou le développement de logiciels géospatiaux opensource. Tout le monde peut soumettre une proposition de présentation et participer à la conférence comme présentateur. Plus d'informations sont disponibles sur la page des présentations sur le site web.

Nous espérons vous voir à Victoria, au Canada en Septembre !

[1] Inscription en ligne : http://www.foss4g2007.org/register/
[2] Site web de la conférence : http://www.foss4g2007.org/
[3] Page des partenaires : http://foss4g2007.org/sponsors
[4] Email Paul Ramsey : pramsey@foss4g2007.org
[5] Soumettez une présentation sur http://www.foss4g2007.org/presentations/

Étude thématique

Évaluation du Web Processing Service de l'OGC ...

... pour une utilisation dans un SIG côté client

par Christopher Michael et Daniel P. Ames, traduit par Yves Jacolin

Résumé

La spécification de Web Processing Service de l'Open Geospatial Consortium a été proposée pour répondre aux besoins de développements de modules de géotraitement par Internet, et de facilité de partage des algorithmes et des fonctionnalités de géotraitement. Cet article cherche à évaluer la proposition de WPS en respect de la faisabilité et de l'utilité potentielle, et d'identifier les zones à améliorer.

Les questions que posent la proposition du WPS sont discutés ensemble avec des solutions posibles. Plusieurs améliorations potentielles à cette proposition sont présentées et considérées, incluant un mécanisme pour guider les applications clientes dans la demande correctes des données et dans le moyen de lister les données disponibles sur un serveur.

Introduction

L'Open Geospatial Consortium (OGC, ou OpenGIS) est une organisation de normalisation par consensus d'abord concerné par la diffusion de spécifications libres (c.a.d. non propriétaire) pour unifier les logiciels d'information géographique, en apportant à la fois une variété de formats disjoints et des mécanismes de communication pour permettre une interoperabilité (Open Geospatial Consortium, 2006). Plutôt que d'éviter cette normalisation et la laisser entièrement propriétaire, plusieurs développeurs de système SIG "ont montrés une extraordinaire coopération en équipe pour soumettre des Spécifications de l'OpenGIS" (Information Today, 1997) et ont activement fait leur ces normes, certains en participant eux-même à leur développement. Comme l'OGC est composé de professionnels dans de nombreux champs d'application, plutôt qu'un simple comité dans un seule environnement corporate, les normes sont typiquement de grande qualité et sont disponible pour n'importe quel nombre de tâche scientifiques différentes.

Le 17 novembre 2005, l'OGC a publié la quatrième révision de la proposition pour la spécifica-

tion appelé Web Processing Service (WPS) (Open Geospatial Consortium, 2005). Cette spécification proposée décrit un mécanisme par lequel un géotraitement peut être réalisé sur des serveurs distant, en utilisant principalement du XML (Extensible Markup Language) pour la communication par Internet. La spécification est faite de telle sorte qu'elle doit être complétement indépendant du langage et de la plateforme. L'OGC a demandé au lecteur de commenter pendant une certaine durée, avec une date butoir au 4 février 2006. Bien que le forum pour les commentaires des lecteurs soit déjà fermé, il y en a eut, jusqu'ici, peu si toutes les études réelles sur la faisabilité et l'utilité de la proposition du point de vue du SIG côté client.

Avant le WPS, des systèmes de géotraitement basé sur le web et des approches similaires au WPS ont été développés par diverses entités. Le plus remarquable, le produit ArcInfo 8.3 (ESRI 2003) de l'Institut de Recherche de Système Environnementale (ESRI) contient une fonctionnalité appelé le Serveur de Géotraitement [NdT : Geoprocessing Server], lequel fonctionne sur des serveurs UNIX pour réaliser des géotraitement derrière des logiciels client ESRI qui envoit un travail à réaliser. Le protocole du Seveur de Géotraitement d'ESRI est propriétaire et fermé ce qui implique que seul les logiciels d'ESRI sont capable d'utiliser les possibilités de traitement distant. D'une manière intéressante, cette fonctionnalité a été retirée de la vesion suivante (ArcGIS 9.0). Une fonctionalité similaire mais subtilement différente a été introduite dans ArcGIS Server 9.2, où un outil appelé "Model Builder" construit à partir de composants d'ESRI plus simple peut servir des clients ArcGIS Desktop et ArcExplorer (Environmental Systems Research Institute, 2006). Contrairement au WPS, le développement d'ESRI n'est pas compatible avec les produits qui ne proviennent pas d'ESRI et est un protocole de communication fermé, propriétaire l'empéchant d'être adopté largement ou étudié dans un environnement non-ESRI.

Qu'est ce qu'est le Web Processing Service (WPS) ?

Le Web Processing Service définie un mécanisme par lequel ubn client peut soumettre un traitement à un serveur. Le service définie une "instance serveur", ou un serveur, en tant qu'entité qui peut fournir un ou plusieurs traitements, ou des tâches de traitement individuel (par exemple, ajouter deux jeu de rasters ensemble peut être un traitement). De cette manière, n'importe quel serveur peut être capable de réaliser des traitements multiples et différent, et pas nécessairement lié.

La spécification indique que le XML (eXtensible Markup Language) doit être utilisé pour toutes les communications. Les documents XMLS sont fait d'éléments individuels, qui sont des conteneurs logiques pour des données liés. Un élément peut contenir d'autres éléments, et n'importe quel élément peut contenir des attributs qui décrivent cette élément. Un exemple simple d'un document XML pourrait être :

```
<landscape name="Smithsonian Park">
<tree type="Elm" height="8" />
</landscape>
```

Ce document XMS décrit un paysage, lequel est indiqué par l'élément nommé "landscape". Un attribut nommé "name" indique qu'il s'agit du Parc Smithsonian. Un sous-élément nommé "Tree" implique que le paysage contient ou possède cet arbre, et l'arbre a deux attributs supplémentaire qui décrivent le type de l'arbre et sa hauteur. L'élément est fermé, ou terminé, en répétant le nom de l'élément avec un slash devant.

XML a été conçue pour être "directement utilisable sur Internet", "lisible par un être humain et raisonnablement clair", "formel et concis", et "facile à écrire" (W3C, 2000). Utiliser du XML est bénéfique principalement par sa lisibilité par un être humain ce qui aide grandement dans la conception et le débogage d'application l'utilisant. Les documents XML peuvent être validé pour s'assurer qu'ils contiennent tous les éléments et attributs nécessaires. La validation est faîtes par un schéma XML, qui est une forme spéciale de document XML qui décrivent la structure qu'un document XML doit suivre.

Le but principal du Web Processing Service est de définir comment communiquer pour réaliser un traitement distant. À cette fin, il y a trois requêtes clés qui peut être faites à un serveur WPS : GetCapabilities, DescribeProcess, et Execute. La première de ces requêtes demande au serveur de lister les traitements individuels qui sont disponible sur le serveur, accompagné d'un court résumé et de mots-clé.La requête ne nécessite pas de paramètres. Une fois qu'un traitement a été identifié à partir de la réponse, une requête "DescribeProcess" peut être envoyée, en précisant le traitement en question comme paramètre. La réponse à cette requête inclus les mêmes informations que la réponse GetCapabilities, avec des informations plus détaillées sur les paramètres en entrée nécessaire pour le traitement et si l'entrée est simple (par exemple un nombre simple comme 23) ou com-

plexe (par exemple un fichier de données). Les sorties complexes sont typiquement encodé en XML, par exemple en utilisant GML (Geographic Markup Language, un parent du XML) pour les données vecteurs.

Si la réponse DescribeProcess indique que c'est le traitement que l'utilisateur ou le client souhaite éxécuter, la troisième requête (Execute) peut être invoquée. C'est avec cette requête que le serveur réalisera en réalité le traitement. Les paramètres obligatoires incluent le nom du traitement ainsi que tout entrée nécessaire pour ce traitement particulier. La réponse à la requête Execute est un document ExecuteResponse, un autre document XML qui indique un status du traitement, qui retourne les entrées qui ont été utilisées, et fournit soit une sortie littérale simple ou des liens vers une sortie complexe. Le status du traitement peut être "ProcessAccepted", informant que le traitement a été reçu et est mis dans la pile des traitements ; "ProcessStarted", informant que le traitement est en cours ; "ProcessSucceeded", signifiant que le traitement est terminé ; ou "ProcessFailed", indiquant que un problème est apparut. Si le status est ProcessAccepted ou ProcessStarted, le status est accompagné par un attribut qui indique où le document ExecuteResponse peut être trouvé. De cette manière le client peut vérifier le status du traitement par une requête sur le prochain document ExecuteResponse. Dans le cas d'un retour ProcessStarted, un message du status et un pourcentage de progression peut être fournit.

Si le status du traitement est ProcessFailed, le document ExecuteResponse contient aussi un code d'erreur dans l'élément XML ExceptionReport, qui peut être un des 5 codes d'erreurs (MissingParameterValue, InvalidParameterValue, NoApplicableCode, ServerBusy ou FileSizeExceeded). Si le traitement réussit, le document en réponse incluera soit les sorties (dans le cs d'une simple valeur littérale) ou des liens URL vers les sorties compelxes (tels qu'un fichier avec des données raster). Si une seule sortie complexe est produit, cette sortie peut être retournée directement à la place du document ExecuteResponse. Ensemble, ces trois requêtes d'opération et leurs réponses constituent la majorité de la proposition du Web Processing Service.

Pourquoi et quand un Web Processing Service doit être utilisé ?

Parce que les fonctionnalités de géotraitement proposées sont illimitées de capacités ou de nature (Open Geospatial Consortium, 2005), la proposition contient de grandes promesses dans l'utilisation d'outils de calcul sans inquiétude tradionnelle telle que la distribution de corection de bug ou la vérification de la version la plus à jour. Bien que l'étendue de ce qui peut être accomplit est illimité, plusieurs opérations peuvent être complétés plus rapidement localement (c'est à dire, sur un PC de bureau d'utilisateur) qu'à distance (c'est à dire sur un serveur central), spécialement après avoir pris en compte le temps de téléchargement des données en entrée et renvoyé les données resultantes en sorties par la suite. Lors du choix entre le traitement des données en local ou distante, un certain nombre de facteur doit être considéré en plus de la taille brute des jeux de données impliquées. La complexité du calcul a une grande importance. Si le calcul prend plusieurs heures pour se terminer même pour un petit jeu de données, il peut être préférable de réaliser les calculs à distance. Si la tâche n'est pas complexe et la plus grosse partie du travaille repose sur l'urgence de gros volumes de données stockées localement, il peut être plus efficasse de réaliser le calcul localement. D'où, avant de choisir d'utiliser seulement le calcul de données distant, un choix intelligent doit être fait entre mettre en place un traitement local ou distant. Comme illustré dans le graphique de la Figure 1, le traitement distant a une place utileans le traitement moderne. De plus grandes capacités de puissance de calcul dans une ferme de serveurs peut facilement réduire le coût des tâches qui demandent du temps et qui sont hautement complexe, spécialement lorsque elles sont combinées à un réseau à haut-débit tel que les fibres optiques ou l'éthernet Gigabit. Par conséquent, les données doivent être envoyées aux serveurs ayant typiquement de grande capacité de calcul, au lieu d'utiliser l'ordinateur local plus lent, lorsque le temp de calcul des données localement serait plus important que le temps combiné de trasmission des données, réaliser le calcul à distance, et les récupérer.

Le calcul distant est également un choix idéal si l'algorithme est relativement nouveau, et est encore en développement actif. Dans ce scénario, les nouvelles publications des versions ne nécessite pas des mis à jour logiciels par l'utilisateur final. Plutôt, seul le serveur nécessite une mise à jour pour refleter le nouveau code ou algorithme. Par conséquence tous les utilisateurs automatiquement gagne un accès à la version la plus récente et la plus précise du calcul simplement en utilisant le service de calcul basé sur le serveur. Ce seul point de contrôle sur le calcul crée également la possiblité de charger le serveur par son

utilisation du calcul, si cela est voulut.

La vue traditionnelle des mandats de traitement distant dont la donnée est téléchargée, tel que dans un Appel de Prrocédure Distant [NdT : Remote Procedure Calls] (RPC) où la donnée en entrée et les paramètres sont envoyé ensemble avec un appel de fonction (Bloomer, 1992). La donnée en entrée peut être également stockée sur le serveur, nécessitant que le client définisse seulement l'entrée particulière qui est nécessaire. Cela crée l'opportunité pour un serveur de traitement de fournir également la donnée : la donnée brute ne sera pas nécessairement fournie, mais la donnée du résultat calculé peut être renvoyé sans nécessairement télécharger la donnée en entrée. Cela prend tout son sens particulièrement lors de traitement nécessitant des données en temps réel tel que les observations des stations météorologiques, les observations du trafic en temps réel, etc. Ces calculs peuvent être fourni par la même agence qui collecte la donnée, autorisant le traitement a avoir accès aux dernières données disponibles. Les motivations pour utiliser un serveur de traitement distant sont multiples, mais la décision ultime doit reposer sur l'utilisateur si le traitement distant est approprié pour la tâche ou non.

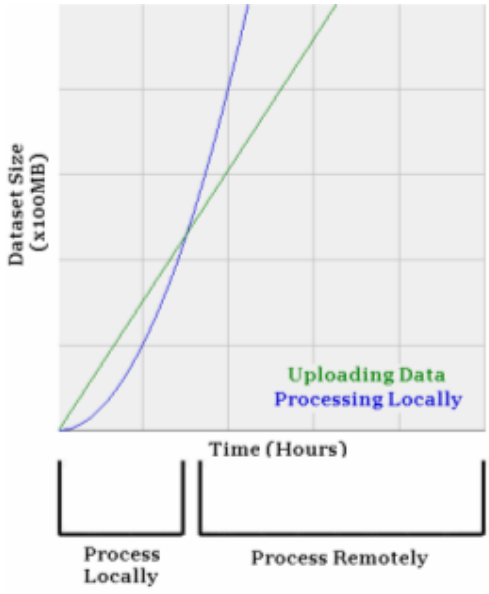

FIG. 1 – Traitement local et traitement distant ont chacun leur temps approprié.

Considérations sur le développement d'un WPS

La proposition du WPS décrit un mécanisme par lequel un ordinateur client peut soumettre un travail à réaliser sur un ordinateur distant. C'est une architecture client/serveur classique, signifiant qu'à la fois un composant client et un composant serveur sont nécessaires. Dans un but de développement et de test il est utile de compiler le composant côté client sur un système d'information géographique pré-existant pour profiter des fonctionnalités de visualisation existantes des données géospatiales à traiter et retourner, cependant, cela n'est pas toujours nécessaire et les tests initiaux peuvent être facilement réalisé en ligne de commande ou avec des outils de test ignorant l'aspect géographique. Ce composant côté client est la portion qui prend en charge la communication XML à travers Internet avec le serveur, idéalement sans que l'utilisateur n'est besoin de voir directement ou d'écrire le WML dans le but de voir les traitements disponibles ou pour lancer leurs requêtes et récupérer les résultats. Il est préférable de compiler ce composant côté client de telle manière qu'il ne repose pas sur un logiciel particulier, propriétaire ou autre.

Le composant côté server lequel fournit à l'implémentation cliente (et à d'autres implémentations clientes) de véritables services doit être également indépendant (c'est à dire non relié à un algorithme de géotraitement ou un calcul qui doit être fourni). Une approche à ce problème est de placer une couche d'interface autour de routines de géotraitement existantes ou non écrites comme des utilitaires en ligne de commande basé sous linux, des services Windows ou des applications windows. Aussi longtemps que le développement de l'application ou de l'algorithme en question peut être éxécuté sans interacion avec l'utilisateur (par exemple, à travers des arguments en ligne de commande, communication TCP/IP, ou par un transfer de données OLE), cette fine couche de communications est une bonne option. Il peut être placé autour de fonctionnalité de géotraitement existant et d'outils existant (ou de nouveaux outils développés) pour leur permetre d'être servi par du WPS, en fournissant un XML qui répond aux nécessités minimales du schéma de communication WPS de l'interface. Plutôt que de nécessiter une re-écriture complète d'un logiciel existant, celui-ci nécessitera certainement de faibles modifications pour le faire fonctionner d'une manière innattendue, idéalement via des arguments en ligne de commande et des fichiers de log générés. Les commandes de GRASS (Neteler, 2006) sont un bon exemple d'outils en ligne de commande existants lesquels peuvent être utilisé (wrapper) d'une manière telle que celle-la.

Ce wrapper peut alors initialiser l'outils en fonction de son besoin, et contrôler un fichier de log ou

renvoyer un message sur le status pour indiquer si le traitement a réussit, est encore en train de tourner, est en échec ou est dans la liste des traitements à faire. Les pourcentages du status et des messages d'erreur pertinent peuvent également être récupéré des fichiers de log, pour être retourné à l'utilisateur final à travers des communications appropriées d'une manière automatique, avec cette technique de wrapping. En plançant ce petit wrapper autour de cet outils indépendant leur permet d'être utilisé dans plusieurs endroits et pour de nombreux objectifs avec peu d'effort. Au fond ce wrapper devient le "Serveur WPS", parce qu'il prend en charge toutes les communications nécessaire par le WPS. Ce wrapper, constituant le serveur WPS, peut être développé comme une page PHP, comme une page en ASP .NET, comme une application indépendante ou en utilisant une autre technologie de serveur.

La plupart des opérations nécessaire à un serveur WPS sont simplement des opérations de métadonnées : fournir des informations sur les traitements individuels (c'est à dire, les entrées requises) et lister les traitements disponibles sur un serveur. Le serveur WPS chargera idéalement les informations sur les traitements disponibles à partir d'un fichier de configuration (peut-être un par traitement) ou d'une base de données, permettant alors au codé écrit pour le serveur WPS d'être réutilisé simplement en ajoutant des traitements additionnels aux fichiers de configuration ou à la base de données. Ces fichiers de configuration ou cette base de donées peut également indiquer au serveur WPS comment lancer le traitement et comment lui envoyé les fichiers en sorties.

Un aperçu du flux de communication suggérée entre une application cliente et une application serveur est montré Figure 2. Au début, l'application cliente envoie une requête au serveur, fournissant une requête GetCapabilities. Le serveur WPS renvoit alors un document XML correspondant au schéma WPS (flèche 1 dans la Fig. 2). Le client présente alors à l'utilisatuer une liste des traitements disponible sur le serveur (flèche 2 dans la Fig. 2). L'utilisateur sélectionne un des ces traitements, entrainant l'envoie par le client d'une requête sur les informations additionnelles sur ce traitement (flèche 3) en envoyant une requête DescribeProcess. L'application cliente aide l'utilisateur à collecter et à entrer les paramètres en entrés nécessaire pour que le traitement soit éxécuté ; il initialise alors le traitement sur le serveur (flèche 3) en envoyant une requête Execute. Le serveur à son tour déclenchera la commande de lancement du traitement spécialisé (flèche 4).

Remarquez que le serveur WPS ne doit pas éxécuter le traitement demandé directement, parce que cela pourrait impliquer un délais : le serveur aurait besoin de terminer le traitement avant qu'une réponse soit envoyé au client. À la place, il ets préférable d'avoir le serveur WPS initialisez un traitement dans différent thread, ou dans différents processus, ainsi une ExecuteResponse initial peut être immédiatement retourné vers le client. De cette manière, les opérations de géotraitement qui nécessitent beaucoup de temps permettra à l'utilisateur de travailler sur son ordinateur. Pour une meilleur propreté et sécurité, il est plus initéressant de démarrer un processus complètement nouveau, ainsi si un processus se termine anormalement il n'arrêtera pas le serveur WPS entièrement également : uen commande de lancement de procéssus spécialisé est alors une bonne solution.

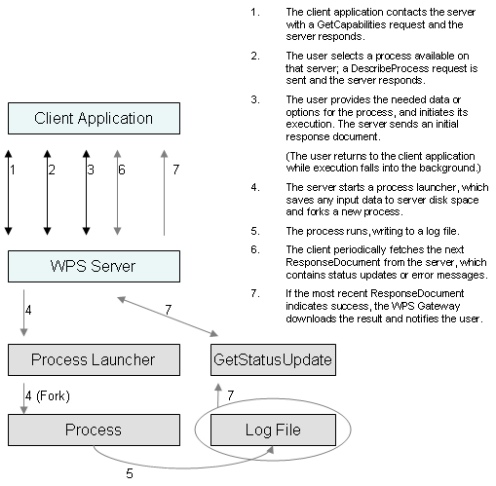

FIG. 2 – Suggested communications flow between the client and the server.

Comme expliqué à la section 2, ce document ExecuteResponse initial informe où l'application cliente peut trouver les futures mises à jours du status (c'est à dire, la prochaine ExecuteResponse) à propos de l'opération demandée, ainsi que la notification du succès, de l'échec, des travaux acceptés ou refusés, et fournit des mises à jours du status. L'application cliente doit questionner le document ExecuteResponse suivant autant de fois que l'utilisateur le désire (flèche 6 à la Fig. 2). Idéalement le serveur WPS doit être conçu de telle manière qu'un document ExecuteResponse peut être trouvé à un endroit logique, peut être en utilisant toujours la même URL pour les réponses les plus récentes. Une bonne approche pour adresser des générations de documents de réponse serait la création de page web spécialisé

ou de composant serveur qui lirait simplement un fichier de log qui serait écrit par un processus éxécuté pour déterminé le status en cours ou les erreurs.

Avoir la possibilité de fournir la mise à jour des status nécessite que le composant du serveur soit capable d'obtenir de tel mise à jour du processus qui est éxécuté. Cela implique qu'un fichier de log est une excellent solution : le processus doit être capable d'écrire un fichier de log ou de rediriger sa sortie standard vers un fichier texte. De cette manière, le fichier peut être automatiquement lu par le serveur WPS, permettant une bonne connectivité entre les composants du système.

Une fois que le document ExecuteResponse indique que le traitement s'est terminé, le serveur doit retourner un message d'erreur applicable ou retourner l'endroit où la donnée générée peut être téléchargée ou récupérée. Le logiciel client doit alors autoriser l'utilisateur pour sauver ou voir la données, puis compléter le rôle du WPS.

Améliorations proposées, problèmes et solutions potentiels

Bien que la proposition du WPS soit capable d'accomplir ses buts initiaux sosu sa forme courante, la proposition peut être améliorée par 6 changements clés. Ces changements incluent deux éléments additionnels dans la réponse DescribeProcess fournit par le serveur, lequel décrit l'entrée et la sortie d'un traitement données, ainsi qu'un mécanisme par lequel un client peut annuler une requête qui est en attente ou en cours. Les changements potentiels incluent également des corrections dans des comportements inconsistant, la fourniture de types d'erreur additionnels pour la prise en charge d'erreur, et la possibilité d'avoir un seul point d'entré pour chaque traitement et peut être un seul point d'entré pour chaque serveur.

Le premier changement suggéré est d'ajouter un élément au document XML (eXtensible Markup Language) lequel est retourné après une requête DescribeProcess. À ce jour, les entrées nécessaire sont listées par ce document, mais aucune description de la manière dont le client doit questionner l'utilisateur pour cet entrée n'est fournis. La données nécessaire peut arriver sous la forme d'une sélection d'une forme sur une carte, d'une valeur littérale tel que "23", rechercher un fichier d'un type données, ou une douzaine d'autres méthodes pour récupérer des données. Dans nos tests de développement, nous avobns introduit un nouvel élément XML appelé "PromptMethod" pour résoudre ce problème. Cet élément peut contenir les valeurs "browseforvector", "browseforraster", "getboundingbox", ou "getmatchingregex". Cela impliquera que l'application cliente devra demander respectivement un fichier vecteur, un fichier raster, récupérer une boite englobante (par exemple, en demandant à l'utilisateur de la dessiner) ou récupérer un certain nombre d'information correspondant à un motif particulier. Les trois premiers ne nécessite aucune explication ; la dernière getmatchingregex, acceptera seulement une valeur entré par l'utilisateur laquelle correspondra à l'expression réguilière fournit.

Une expression régulière est une règle définie par des caractères spéciaux, tels que "^[a-zA-Z][a-zA-Z]$". Cette expression régulière cherchera au début de l'entrée (symbolisé par ^), suivit de deux caractères, de A à Z, indépendemment de la casse, suivie par la fin de l'entrée (symbolisé par le symbol dollar). Cela fournit un filtre en entrée flexible et rapidement configurable afin de s'assurer que l'utilisateur entre bien l'entré qui est nécessaire. Il existe certaines variations de la syntaxe dans les expressions régulière à cause de la diversité des moteurs de traitement des expressions réguilières (Goyvaerts, 2006), singifiant qu'un soin particulier doit être pris afin que les expressions sont conçu de telle manière qu'elles seront interprétées correctement par tous les clients et le serveur. Un exemple de l'addition suggérée au WPS peut être vue à la Figure 3, où l'expression régulière a été définie par "^\d{8}$". Cette expression indique que le début de la chaîne (^) doit être suivit par 8 chiffres (d{8}) puis par la fin de la chaîne ($). Cette expression assume que le traitement des expressions régulière de Microsoft sera utilisé.

Le second changement suggéré est une autre addition à la réponse DescribeProcess. À ce jour il n'existe pas de mécanisme par lequel un serveur peut lister les données disponibles sur le serveur pour une utilisation pour un traitement données. Un très bon exemple où cela peut être utile est dans le calcul des limites des bassins versant - définissant un réseau de flux et de bassin basé sur un jeu de données de modèle numérique de terrain (Savant et al., 2002). Les raster d'élévation peuvent typiquement être très important, il est donc indésirable et souvent impossible de télécharger complétement le fichier en entrée sur le serveur. Les jeux de données tels que les données d'élévation ne change pas souvent, et peuvent être stockées sur le serveur pour sauvegarder du temps. Dans nos tests de développement, nous résolvons cela en introduisant un élément XML nommé "AvailableData",avec un élément

enfant pour chaque données contenant un nom et une brève descriptions comme montré Figure 4. Non seulement cela augmente la vitesse du traitement en enlevant le besoin de télécharger les données en entrée, mais cela crée également un moyen pour lequel un serveur WPS peut agir comme dépôt de données ou les traiter et renvoyer les données calculées.

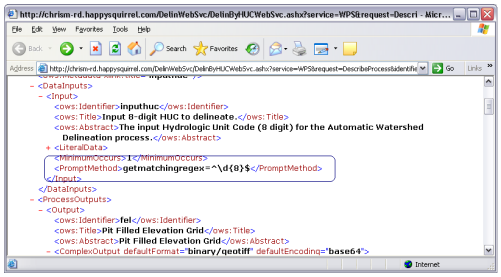

FIG. 3 – Élément PromptMethod suggéré dans la réponse DescribeProcess.

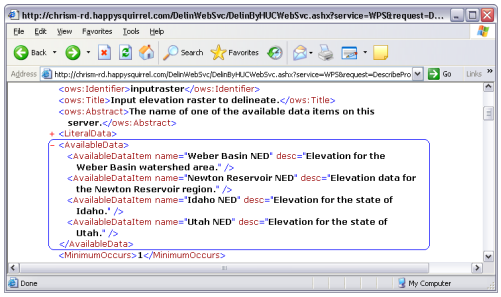

FIG. 4 – Élément AvailableData suggéré dans la réponse DescribeProcess.

Dans la proposition actuel du WPS il i y a une incohérence après avoir soumit un travail au serveur en regards de ce qu'il devrait arriver par la suite. Si le traitement résultera en une seule valeur retourné et que la requête Execute a été réalisé avec le paramètre "store" à false, le WPS autorisera le traitement à retourner immédiatement la sortie, plutôt qu'un document ExecuteResponse XML. S'il y a plus d'un résultat, ou si on a demandé au traitement de stocker les résultats, alors une document ExecuteResponse est généré. Ce comportement est incohérent , puisque n'importe quel traitement pourrait retourner soit de multiples résultat en sortie ou un seul, en fonction des paramètres fournis au triatement. Au lieu de cela, il est plus simple de toujours retourner un document ExecuteResponse, en stockant n'importe quelle donnée en sortie (par exemple des fichiers vecteurs ou rasters) sur le serveur jusqu'au téléchargement par le client. De simple valeur en sortie (par exemple un nombre unique) peut être retourné directement inclus dans le document ExecuteResponse, avec des liens pointant vers les sorties complexes. Notre troisième suggestion est de toujours demander de retourner un document ExecuteResponse pour fournir une meilleur cohérence et une simplification du développement du client.

Après avoir soumis un travail au serveur WPS, il peut être utile d'annuler l'opération demandée ; cette éventualité n'est pas prévue dans la proposition actuelle. Au cours de nos développement tests (et comme pour nos quatres changements proposés) nous avons introduit l'utilisation d'une "URL d'annulation de requête" dans le document ExecuteResponse, en compagnie de l'URL existante indiquant où trouver le prochain document ExecuteResponse. De cette manière un client souhaitant annuler un traitement a seulement besoin d'accèder à l'URL pour lancer l'annulation du traitement demandé, empéchant de gacher du temps de traitement du serveur pour des traitements indésirables.

Notre cinqième changement suggéré à la proposition du WPS est d'avoir un unique point d'entré pour chaque requête possible (GetCapabilities, DescribeProcess, et Execute) pour un traitement donné. Idéalement, un unique point d'entrée (par exemple une unique page PHP) pourrait être utilisé non seulement pour chaque requête sur un traitement, mais aussi pour chaque traitement disponible sur ce serveur. Pour l'instant, chacune des requêtes qu'un traitment gère peut avoir une URL différente pour réaliser cete requête, comme illustré à la figure 5. Dans cette figure, l'URL GetCapabilities est entouré en bleu, l'url DescribeProcess en vert et l'URL Execute en rouge. Ici chaque URL est la même, rendant la maintenance et le développement plus facile, bien qu'il soit acceptable d'avoir différent URL d'accès pourchaque opération. Des confusions ou des erreurs peuvent facilement arriver avec les différentes URL pour ces opérations. Cel apeut être corriger facilement en le rendant nécessaire pour utiliser une seul URL pour les trois requêtes qu'un traitement doit géré.

Enfin, notre sixième suggestion est de développer un système de gestion d'erreur plus structuré. Pour l'instant, il ya très peu de types d'erreur (Pour l'instant il y a tré peu d'erreur MissingParameterValue, InvalidParameterValue, NoApplicableCode, ServerBusy et FileSizeExceeded), qui sont limité - particulièrement lors de la tentative de lecture automatique d'une erreur. Avec un faible nombre de types d'erreur (codes d'erreur), il n'y a pas de manière générique pour prendre en charge les erreurs de la part de l'utilisateur. Une hiérarchie plus complexe et struc-

turée permettrait aux applications clientes de comprendre la nature de l'erreur qui est apparue, peut être en récupérant ou réessayant automatiquement si nécessaire. La possibilité de insonoriser l'utilisateur des erreurs et de récupérer élégament est une partie importante de n'importe quelle structure de standard, et cette capacité pourrait être rendu possible avec une hiérarchie des erreurs plus détaillée.

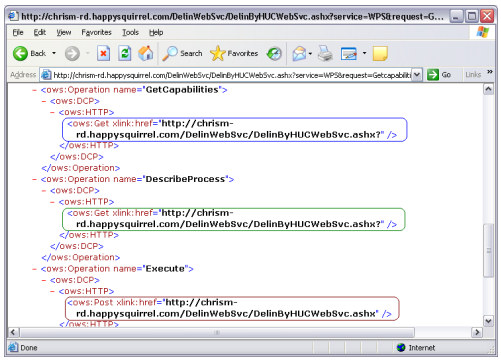

FIG. 5 – Possiblité d'avoir différente URL pour chaque requête pour le même traitement peut entrainer une confusion et une complexité à éviter.

Conclusions

Dans l'ensemble, la proposition du WPS est un moyen performant de normalisation d'un mécanisme de communications pour le géotraitement client/serveur. La proposition fonctionne comme elle est actuellement structuré et est en effet approprié pour plusieurs tâches SIG. Le développement de composant client et les traitements WPS du serveur a montré que la proposition du WPS est solide et efficasse. Néanmoins, il y a plusieurs opportunités d'amélioration à la proposition du WPS en incluabnt les six recommendations spécifiques fournit ici. Nous espérons que ces observations seront bénéfique à chacun pour développer les spécifications du WPS proposé sur diverses plateformes SIG.

N'importe quelle activité de géotraitement peut être présenté au format WPS. Cela permet aux développeurs d'algorithme pour continuellement améliorer l'algorithme tout en permettant aux utilisateurs n'importe où d'utiliser le code, sans devoir mettre à jour vers la dernière version. Cela assure également que le code et les algorithmes utilisés soient de grande qualité : si un bug est détecté dans une version publiée d'un algorithme qui est distribué sur un CD, aucun moyen n'existe pour s'assurer que tous les utilisateurs mettent à jours avec succès à la version corrigée. Avec une architecture de traitement basé sur Internet, ce problème est simplifié simplement en publiant le nouvel algorithme sur le serveur de traitement. Les géotraitement consommant du temps et intesément le processeur peut être réalisé sur un serveurdistant, libérant le bureau de l'utilisateur pour travailler sur d'autres tâches. Les gros jeux de données peuvent être stockés sur le serveur et traiter en fonction des paramètres particulier de l'utilisateur, en combinant les aspects de dépôt de données et de traitement de données dans un seul système.

La proposition du WPS introduit un standard qui permettra à différents développeurs dans des endroits différents de produire des offres de géotraitement et de les leur fournir facilement à différentes plateformes clientes. Dans ces tests, il a été démontré que la proposition du WPS est pratique et utiliasble dans sa définition actuel. Les améliorations additionnels proposé peuvent améliorer le WPS significativement, le rendant meilleur pour la tâche pour laquelle il a été crée.

Bibliographie

[1] Bloomer, John. 1992. **Power Programming with RPC. Cambridge, MA :** O'Reilly Media.

[2] Environmental Systems Research Institute (ESRI). 2003. **ArcGIS Desktop Products Data Sheet.** WWW document, http://www.esrichina-bj.cn/produce/esri/arcgisdesktopsheet.pdf

[3] Environmental Systems Research Institute (ESRI). (2006). **ArcGIS 9.2 Webinar – ArcGIS Server : Publishing a Geoprocessing Model.** WWW document, http://events.esri.com/info/index.cfm?fuseaction=seminarRegForm\&shownumber=9919

[4] **GIS Competitors Cooperate on OpenGIS Specs. 1997.** Information Today, 14(2), 15-15.

[5] Goyvaerts, Jan. 2006. **Regular Expression Tutorial.** WWW document, http://www.regular-expressions.info/tutorial.html

[6] Open Geospatial Consortium, Inc. 2006. **Vision and Mission.** WWW document, http://www.opengeospatial.org/about/?page=vision

[7] Open Geospatial Consortium, Inc. 2005. **OpenGIS® Web Processing Service (WPS) Discussion Paper.** WWW document, http://www.opengeospatial.org/

Christopher Michael
Research Assistant
Idaho State University Geospatial Software Lab

Daniel P. Ames, PhD
Assistant Professor
Department of Geosciences, Idaho State University

FOSS4G - Ouverture des Inscriptions à la Conférence

Nous sommes heureux de vous annoncer l'ouverture des inscriptions en ligne à la Conférence Free and Open Source Software for Geospatial 2007 (FOSS4G 2007). FOSS4G est l'évènement annuel qui réunit les personnes et les sociétés qui créent, utilisent, et gèrent des logiciels géospatiaux open source. Inscrivez-vous dès maintenant en ligne.[6]

Inscrivez-vous avant la date limite du 27 Juillet, pour économiser sur les frais d'inscription ! Tirez profit de l'opportunité que FOSS4G 2007 vous offre, de construire un réseau avec les autres professionnels des données géospatiales, de renouveler d'anciennes relations, et d'en créer de nouvelles.

Pour les dernières mises à jour, l'inscription et/ou la soumission d'une présentation, visitez le site web de la conférence.[7]

OPPORTUNITES D'EXPOSITION & DE SPONSORING

Concernant les opportunités d'exposition et de sponsoring, lisez la page des partenaires [8] ou contactez Paul Ramsey, Président de la Conférence par email.[9]

SOUMETTRE UNE PRESENTATION

Vous pouvez soumettre une présentation en ligne.[10] La date limite pour les soumissions est le 29 Juin 2007.

Les présentations FOSS4G durent 25 minutes, avec 5 minutes de questions/réponses à la fin. Les présentations concernent l'utilisation ou le développement de logiciels géospatiaux opensource. Tout le monde peut soumettre une proposition de présentation et participer à la conférence comme présentateur. Plus d'informations sont disponibles sur la page des présentations sur le site web.

Nous espérons vous voir à Victoria, au Canada en Septembre !

[6] Inscription en ligne : http://www.foss4g2007.org/register/
[7] Site web de la conférence : http://www.foss4g2007.org/
[8] Page des partenaires : http://foss4g2007.org/sponsors
[9] Email Paul Ramsey : pramsey@foss4g2007.org
[10] Soumettez une présentation sur http://www.foss4g2007.org/presentations/

Rédacteur en chef :
Tyler Mitchell - tmitchell AT osgeo.org

Rédacteur, Actualité :
Jason Fournier

Rédactrice, Étude de cas :
Micha Silver

Rédacteur, Zoom sur un projet :
Martin Wegmann

Rédacteur, Étude d'intégration :
Martin Wegmann

Rédacteur, Documents de programmation :
Landon Blake

Remerciements
Tous les relecteurs & le project Actualités de GRASS

Le *journal de l'OSGeo* est une publication de la *Fondation OSGeo*. La base de ce journal, les sources du style LaTeX 2ε ont été généreusement fournit par l'équipe éditoriale de l'actualité de GRASS et R.

Journal en ligne : http://www.osgeo.org/journal

Site de l'OSGeo : http://www.osgeo.org

Contact postal pour l'OSGeo, PO Box 4844, Williams Lake, British Columbia, Canada, V2G 2V8

ISSN 1994-1897