Passerelle - Development #39416

cmis : ajout des métadonnés

30 janvier 2020 14:11 - Emmanuel Cazenave

Statut:	Fermé	**Début:**	30 janvier 2020
Priorité:	Normal	**Echéance:**	
Assigné à:	Valentin Deniaud	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Au niveau connecteur, le endpoint pourrait gagner les paramètres suivants :

object_type : pour passer un type de doc, (ex : 'object_type': 'D:dui:type')

Et ensuite un nombre arbitraire de paramètres du type :

```
meta_xxx : valeur (ex: 'meta_dui:tnumDossier': 'ABCD')
```

A voir aussi les possibilités d'introspection pour :

- vérifier/convertir les métadonnés avant de les balancer vers le serveur cmis
- afficher sur la page d'accueil du connecteur les types d'objets/types de métadonnées acceptables

Voir #39357 pour la petit expérience préparatoire.

Révisions associées

Révision 45428e94 - 17 février 2020 15:23 - Valentin Deniaud

cmis: add object type and property support (#39416)

Révision b2fe74db - 17 février 2020 15:23 - Valentin Deniaud

cmis: add view to list available object types (#39416)

Historique

#1 - 04 février 2020 16:54 - Valentin Deniaud

- Assigné à mis à Valentin Deniaud

#2 - 04 février 2020 18:23 - Valentin Deniaud

J'ai quelques remarques/questions avant de patcher :

Emmanuel Cazenave a écrit :

> Et ensuite un nombre arbitraire de paramètres du type :
> meta_xxx : valeur (ex: 'meta_dui:tnumDossier': 'ABCD')

Ça me paraît bien étrange et compliqué à faire comprendre à JSON Schema. Pourquoi pas simplement un paramètre metadata de type objet, genre metadata: {'dui:tnumDossier': 'ABCD', ...} ?

- vérifier/convertir les métadonnés avant de les balancer vers le serveur cmis

Quel est le besoin ? Ça risque de faire beaucoup de code pour la beauté de ne pas balancer n'importe quoi au serveur, alors que j'ai plutôt l'impression que c'est son boulot de faire la validation et de nous retourner des erreurs, qu'on affiche ensuite. L'un ou l'autre, pour l'utilisateur, c'est du pareil au même, donc autant faire simple.

- afficher sur la page d'accueil du connecteur les types d'objets/types de métadonnées acceptables

Bon ici je soulève un problème de terminologie qui m'a compliqué la lecture des discussions précédentes, ce qui est entendu par métadonnée ici se nomme propriété dans le vocable CMIS[1]. Je serais pour partir sur le terme de la spec et oublier « métadonnées » (donc dans l'exemple plus haut,

remplacer metadata par properties).
Ensuite j'émets de nouveau des doutes sur l'utilité vs la difficulté d'afficher ces infos :

- On récupère facilement tous les types avec repo.getTypeDescendants().
  - Mais on a pas de contrôle sur ce que ça sort, il peut très bien y en avoir 100.
- Si on récupère en plus les propriétés, le temps de chargement de la page du connecteur sera beaucoup trop long.
  - Et ce sera compliqué d'avoir un affichage intelligible.
- À côté de ça, la personne qui utilise l'API a tout ça mieux présenté dans le GUI du serveur.

Donc je serais pour zapper aussi.

[1] Le bon sens de métadonnées ça a plutôt l'air d'être type d'objet + propriétés.

#3 - 04 février 2020 19:27 - Emmanuel Cazenave

Valentin Deniaud a écrit :

> J'ai quelques remarques/questions avant de patcher :
>
> Ça me paraît bien étrange et compliqué à faire comprendre à JSON Schema. Pourquoi pas simplement un paramètre metadata de type objet, genre metadata: {'dui:tnumDossier': 'ABCD', ...} ?

Parce que dans l'interface d'appels WS de wcs, on a pas ce qu'il faut pour spécifier des sous objets JSON.

> - vérifier/convertir les métadonnés avant de les balancer vers le serveur cmis
>
> Quel est le besoin ?

Ok je pense qu'on peut tout à fait zapper ça pour l'instant, on verra à l'usage.

> - afficher sur la page d'accueil du connecteur les types d'objets/types de métadonnées acceptables
>
> Bon ici je soulève un problème de terminologie qui m'a compliqué la lecture des discussions précédentes, ce qui est entendu par métadonnée ici se nomme propriété dans le vocable CMIS[1]. Je serais pour partir sur le terme de la spec et oublier « métadonnées » (donc dans l'exemple plus haut, remplacer metadata par properties).

Ok.

> Ensuite j'émets de nouveau des doutes sur l'utilité vs la difficulté d'afficher ces infos :
>
> - À côté de ça, la personne qui utilise l'API a tout ça mieux présenté dans le GUI du serveur.

Hmmm la personne qui écrit l'appel webservice dans un workflow est pas forcément la même que celle qui a la main sur le serveur de GED et quand bien même, voir sur la page d'un connecteur quels paramètres sont attendus, c'est toujours mieux que de devoir aller chercher l'info ailleurs.

> - On récupère facilement tous les types avec repo.getTypeDescendants() ...

repo.getTypeChildren(typeId='cmis:document') a l'air de donner un truc exploitable mais je n'ai pas creusé.

Si ça ne marche d'y aller à l'aveugle, on pourrait imaginer que le connecteur gagne dans son modèle un paramètre optionnel 'document_types', qui permettrait de déclarer le nom des types de documents qui nous intéressent (ici 'cmis:document,D:dui:type'), et qui guiderait l'introspection.

Bref ça me semble bon à creuser un peu (mais pas des jours non plus), la plus-value serait pas mal. Mais effectivement si on arrive pas à obtenir un affichage concis en un temps raisonnable, on laisse tomber.

#5 - 04 février 2020 19:56 - Benjamin Dauvergne

Emmanuel Cazenave a écrit :

> Valentin Deniaud a écrit :
>
> > J'ai quelques remarques/questions avant de patcher :
> >
> > Ça me paraît bien étrange et compliqué à faire comprendre à JSON Schema. Pourquoi pas simplement un paramètre metadata de type objet, genre metadata: {'dui:tnumDossier': 'ABCD', ...} ?

> Parce que dans l'interface d'appels WS de wcs, on a pas ce qu'il faut pour spécifier des sous objets JSON.

Il faut utiliser flatten/unflatten qui sont prévus pour ça (ça convertit automatiquement une clé metadata/dui:tnumDossieer en ce qu'on souhaite et c'est plus agréable à lire coté w.c.s.

#6 - 04 février 2020 20:04 - Benjamin Dauvergne

Valentin Deniaud a écrit :

- Si on récupère en plus les propriétés, le temps de chargement de la page du connecteur sera beaucoup trop long.
  - Et ce sera compliqué d'avoir un affichage intelligible.

Présenter l'arbre des types puis des liens vers des sous-pages qui feront la récupération des propriétés, ou un bout d'ajax (vers un endpoint local), un truc pour explorer.

- À côté de ça, la personne qui utilise l'API a tout ça mieux présenté dans le GUI du serveur.

La personne qui utilise l'API ce sera souvent nous et on aura pas du tout accès à la GED en face à part via CMIS.

#8 - 04 février 2020 21:16 - Benjamin Dauvergne

Faut arrêter avec les notes privées :)

Emmanuel Cazenave a écrit :

> Se lancer là dedans va éclater le financement Toulousain (sauf si Valentin arrive faire ça en une demi journée top chrono), ce qui ne veux pas dire que c'est une mauvaise idée (bon enfin perso je trouve ça overkill quand même, l'arbre des types a l'air maouse costaud et tout à fait bordélique) mais je pose ça là.

Je ne sais pas je n'ai pas vu l'arbre, c'est difficile de donner son avis sans voir de quoi on parle, mais bon j'imaginais un truc tout simple :

```
<ul>
{% for type in repo.getTypeDescendants %}
<li><a href="{% url "cmis-type" type=type.typeId %}">{{ type.typeId }}</a></li>
{% endfor %}
</ul>
```

ça peut même aller sur une autre page "types de document" si ça prend trop de place sur la page principale.

Puis dans la vue cmis-type le même genre de boucle toute con sur les propriétés, avec juste leur nom, leur type et leur description ou label.

#9 - 04 février 2020 21:33 - Benjamin Dauvergne

Pour ne pas mourir bête, je suis allé voir et effectivement c'est plein de **bip**.

On pourrait simplement afficher les types issus de repo.getTypeDefinitions() dans un <ul> et faire pointer les liens vers une page qui, étant donné un type affiche la liste des propriétés et des sous-types (et ça repointe vers la même page), en sachant qu'en général il n'y a qu'un ou deux types issus de cmis:document qui nous intéresseront.

À coder y en a pour max 1h.

#10 - 05 février 2020 16:47 - Valentin Deniaud

- Fichier 0001-cmis-add-object-type-and-property-support-39416.patch ajouté

- Fichier 0002-cmis-add-view-to-list-available-object-types-39416.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

Je me mets à l'écriture des tests, voilà le code en attendant.

Le unflatten marche sans problème, c'est cool.

Je suis parti sur une vue séparée pour afficher les types, rien sur la page principale du connecteur si ce n'est un lien vers cette vue.
D'ailleurs pour l'instant ce lien n'apparaît pas : j'ai ajouté un template cmis_detail.html, mais il n'a pas l'air détecté, qu'est-ce que j'oublie ?

#11 - 05 février 2020 17:07 - Valentin Deniaud

Valentin Deniaud a écrit :

> D'ailleurs pour l'instant ce lien n'apparaît pas : j'ai ajouté un template cmis_detail.html, mais il n'a pas l'air détecté, qu'est-ce que j'oublie ?

C'est bon, en fait ça se base sur la détection automatique de DetailView, donc c'est cmisconnector_detail.html et pas juste cmis_. J'enverrai la nouvelle version des patches quand il y aura des tests, et un peu de CSS pour que ce soit moins moche.

#12 - 05 février 2020 17:37 - Benjamin Dauvergne

Valentin Deniaud a écrit :

> Je me mets à l'écriture des tests, voilà le code en attendant.

La validation des propriétés, je me dis que pour l'instant autant ne rien faire et laisser un commentaire :

```
FIXME: validate properties against object-type (default type being cmis:document)
```

#13 - 06 février 2020 12:22 - Valentin Deniaud

- Fichier 0001-cmis-add-object-type-and-property-support-39416.patch ajouté

- Fichier 0002-cmis-add-view-to-list-available-object-types-39416.patch ajouté

Voilà pour les tests.

Benjamin Dauvergne a écrit :

> La validation des propriétés, je me dis que pour l'instant autant ne rien faire et laisser un commentaire :

Tristesse, je pensais que le serveur allait renvoyer une jolie erreur en cas de type invalide et qu'il nous suffirait de l'afficher... En fait, il balance une 500. Je veux bien l'ajouter du coup, à voir.

#14 - 06 février 2020 12:32 - Benjamin Dauvergne

Valentin Deniaud a écrit :

> Voilà pour les tests.
>
> Benjamin Dauvergne a écrit :
>
> > La validation des propriétés, je me dis que pour l'instant autant ne rien faire et laisser un commentaire :
>
> Tristesse, je pensais que le serveur allait renvoyer une jolie erreur en cas de type invalide et qu'il nous suffirait de l'afficher... En fait, il balance une 500. Je veux bien l'ajouter du coup, à voir.

Mouais, il balancera une 500 du P:dui:dui qu'on va utiliser aussi, donc bon, on est même pas sûr d'utiliser beaucoup cmis:document c'est pour ça que je disais que c'était du code inutile pour l'instant.

#15 - 06 février 2020 14:42 - Valentin Deniaud

- Fichier 0001-cmis-add-object-type-and-property-support-39416.patch ajouté

- Fichier 0002-cmis-add-view-to-list-available-object-types-39416.patch ajouté

Bon pour review donc, les tests py3 passeront une fois que #39391 aura été mergé.

#16 - 10 février 2020 11:02 - Emmanuel Cazenave

- Statut changé de Solution proposée à Solution validée

Ok (à pousser prochain cycle).

#17 - 17 février 2020 15:25 - Valentin Deniaud

- Statut changé de Solution validée à Résolu (à déployer)

```
commit b2fe74db2fe62dd04165bbec95ce715bdddcc626
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Wed Feb 5 16:42:06 2020 +0100
```

```
    cmis: add view to list available object types (#39416)

commit 45428e944ac6743dd3909e0428f53ac0e4a9f112
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Wed Feb 5 16:23:04 2020 +0100

    cmis: add object type and property support (#39416)
```

#18 - 19 février 2020 18:15 - Frédéric Péters

- Statut changé de Résolu (à déployer) à Solution déployée

Fichiers

0001-cmis-add-object-type-and-property-support-39416.patch	2,5 ko	05 février 2020	Valentin Deniaud
0002-cmis-add-view-to-list-available-object-types-39416.patch	6,18 ko	05 février 2020	Valentin Deniaud
0001-cmis-add-object-type-and-property-support-39416.patch	6,97 ko	06 février 2020	Valentin Deniaud
0002-cmis-add-view-to-list-available-object-types-39416.patch	9,83 ko	06 février 2020	Valentin Deniaud
0001-cmis-add-object-type-and-property-support-39416.patch	6,97 ko	06 février 2020	Valentin Deniaud
0002-cmis-add-view-to-list-available-object-types-39416.patch	9,84 ko	06 février 2020	Valentin Deniaud