

Passerelle - Development #62012

toulouse_smart: pouvoir relancer la création d'une intervention de façon idempotente.

21 février 2022 12:27 - Nicolas Roche

Statut:	Fermé	Début:	21 février 2022
Priorité:	Normal	Echéance:	
Assigné à:	Nicolas Roche	% réalisé:	0%
Catégorie:		Temps estimé:	0:00 heure
Version cible:		Planning:	Non
Patch proposed:	Oui		

Description

Pour prévoir les cas où wcs ne reçoit pas la réponse du connecteur, il faudrait que wcs puisse relancer une demande de création d'intervention.

Passerelle devrait alors ignorer la même requête pour une même demande faite une deuxième fois, s'il a déjà créé un job pour celle-ci.

Éventuellement, prévoir de modifier le payload s'il faut rejouer la requête parce qu'elle a vraiment échouée (au bout de 5 essais asynchrones).

Révisions associées

Révision 4543200d - 26 août 2022 09:52 - Nicolas Roche

toulouse_smart: check response error on tests (#62012)

Révision 5f695ccd - 26 août 2022 09:52 - Nicolas Roche

toulouse_smart: check media path on tests (#62012)

Révision 5657d289 - 26 août 2022 09:52 - Nicolas Roche

toulouse_smart: factorize wcs_request response (#62012)

Révision a8530e28 - 26 août 2022 09:52 - Nicolas Roche

toulouse_smart: allow to redo a failed intervention (#62012)

Historique

#2 - 21 février 2022 12:53 - Nicolas Roche

- Fichier 0001-toulouse_smart-allow-to-redo-a-failed-intervention-6.patch ajouté
- Statut changé de Nouveau à Solution proposée
- Patch proposed changé de Non à Oui

#3 - 04 juillet 2022 11:31 - Benjamin Dauvergne

- Statut changé de Solution proposée à En cours

Comme tu modifies une demandes existante, pour éviter de te marcher dessus avec le job d'envoi il faudrait enfermer tout ça dans une transaction (transaction.atomic) et faire un select_for_update() si le .get() réussit.

Il faudrait faire pareil dans le job (pour que les deux méthodes verrouillent l'objet WcsRequest en base).

#4 - 23 juillet 2022 19:54 - Nicolas Roche

- Fichier 0004-toulouse_smart-allow-to-redo-a-failed-intervention-6.patch ajouté
- Fichier 0003-toulouse_smart-factorize-wcs_request-response-62012.patch ajouté
- Fichier 0002-toulouse_smart-check-media-path-on-tests-62012.patch ajouté
- Fichier 0001-toulouse_smart-check-response-error-on-tests-62012.patch ajouté
- Statut changé de En cours à Solution proposée

il faudrait enfermer tout ça dans une transaction (`transaction.atomic`) et faire un `select_for_update()` si le `.get()` réussit.

J'ai fait ça (`create_intervention` est déjà en `atomic`, via un décorateur).

Il faudrait faire pareil dans le job (pour que les deux méthodes verrouillent l'objet `WcsRequest` en base).

Fait.

#5 - 25 juillet 2022 15:09 - Benjamin Dauvergne

- *Statut changé de Solution proposée à En cours*

Tu ne peux pas faire de `.select_for_update()` en dehors d'une transaction :

```
wcs_request = self.wcs_requests.select_for_update().get(pk=kwargs['pk'])
with atomic():
```

ici ça passe dans les tests parce qu'ils sont joués dans une transaction aussi (c'est moche, ça cache toutes les erreurs de ce genre).

#6 - 25 juillet 2022 16:47 - Nicolas Roche

- *Fichier 0004-toulouse_smart-allow-to-redo-a-failed-intervention-6.patch ajouté*

- *Statut changé de En cours à Solution proposée*

Tu ne peux pas faire de `.select_for_update()` en dehors d'une transaction

Compris, c'était bien indiqué dans la doc django, mais l'exemple n'était pas clair <https://docs.djangoproject.com/fr/4.0/ref/models/querysets/#select-for-update> (c'est pas évident ce patch, parce que je ne saurais pas écrire un test de non régression).

Nouveau patch 0004 qui modifie ça :

```
def create_intervention_job(self, *args, **kwargs):
-     wcs_request = self.wcs_requests.select_for_update().get(pk=kwargs['pk'])
-     with atomic():
+         wcs_request = self.wcs_requests.select_for_update().get(pk=kwargs['pk'])
+         after_timestamp = None
```

#7 - 25 juillet 2022 16:57 - Benjamin Dauvergne

- *Statut changé de Solution proposée à Solution validée*

Nicolas Roche a écrit :

Tu ne peux pas faire de `.select_for_update()` en dehors d'une transaction

Compris, c'était bien indiqué dans la doc django, mais l'exemple n'était pas clair <https://docs.djangoproject.com/fr/4.0/ref/models/querysets/#select-for-update> (c'est pas évident ce patch, parce que je ne saurais pas écrire un test de non régression).

Nouveau patch 0004 qui modifie ça :
[...]

Je ne sais pas ce que tu appelles non régression ici, si c'est testé l'exclusion du au verrou par `select_for_update()` ça peut se faire avec `transactional_db` et de `threads/processs` mais c'est pas vraiment utile. Si le code passe sur le `.select_for_update()` c'est verrouillé, on sait que ailleurs si on essaie de verrouiller ça ne passera pas, en général si tu n'as à poser qu'un verrou aucun souci n'est possible (à partir de deux verrous tu peux avoir des soucis).

Ici ton test est bien, il teste qu'on reprend bien une demande existante en cas de rejeu et c'est verrouillé, y a pas d'autre moyen.

Si on avait besoin de performance ici Pierre D. te dirait de limiter le `select_for_update()` au cas `created == False` mais je ne pense pas qu'on en ait besoin (dans le cas où l'objet est créé il est invisible en dehors de la transaction jusqu'au commit).

#8 - 26 août 2022 10:27 - Nicolas Roche

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

```
commit a8530e2806c0d31c454c4a80a1f8f912c1fbaaa8
Author: Nicolas ROCHE <nroche@entrouvert.com>
Date: Sat Jul 23 19:10:41 2022 +0200
```

```
toulouse_smart: allow to redo a failed intervention (#62012)
```

```
commit 5657d2896b7607ace4b70fd563d545ac7a5bd5e1
Author: Nicolas ROCHE <nroche@entrouvert.com>
Date: Sat Jul 23 19:09:20 2022 +0200
```

```
toulouse_smart: factorize wcs_request response (#62012)
```

```
commit 5f695ccd4aa516e84c2fda229cdea1b8094d13ce
Author: Nicolas ROCHE <nroche@entrouvert.com>
Date: Sat Jul 23 11:46:47 2022 +0200
```

```
toulouse_smart: check media path on tests (#62012)
```

```
commit 4543200ddf09fde0bc73aba0eb87632be06eb541
Author: Nicolas ROCHE <nroche@entrouvert.com>
Date: Sat Jul 23 11:40:20 2022 +0200
```

```
toulouse_smart: check response error on tests (#62012)
```

#9 - 01 septembre 2022 09:15 - Transition automatique

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

#10 - 06 novembre 2022 04:42 - Transition automatique

Automatic expiration

Fichiers

0001-toulouse_smart-allow-to-redo-a-failed-intervention-6.patch	8,06 ko	21 février 2022	Nicolas Roche
0003-toulouse_smart-factorize-wcs_request-response-62012.patch	3,71 ko	23 juillet 2022	Nicolas Roche
0004-toulouse_smart-allow-to-redo-a-failed-intervention-6.patch	7,25 ko	23 juillet 2022	Nicolas Roche
0002-toulouse_smart-check-media-path-on-tests-62012.patch	1,31 ko	23 juillet 2022	Nicolas Roche
0001-toulouse_smart-check-response-error-on-tests-62012.patch	3,13 ko	23 juillet 2022	Nicolas Roche
0004-toulouse_smart-allow-to-redo-a-failed-intervention-6.patch	7,25 ko	25 juillet 2022	Nicolas Roche