|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| **Title\*:** | Returning announced resources from collections | | |
|  |  | | |
| from **Source**\*: | Telefon AB LM Ericsson | | |
| Contact: | Erik Van der Velden | | |
|  |  | | |
| input for **Committee**\***:** | M2M WG2 | | |
|  |  | | |
| Contribution **For\*:** | Decision | **X** |  |
|  | Discussion |  |  |
|  | Information |  |  |
|  |  | | |
| Submission date**\***: | 2011-08-18 | | |
|  |  | | |
| Meeting & Allocation: | **M2MWG2-Tuesday-G2M-Architectur** - | | |
| Relevant WI(s), or deliverable(s): |  | | |
|  | | | |

**Decision/action requested:** include in TS

**ABSTRACT:** *This contribution is based on an email discussion on the list.*

*The question was posed like this:*

> Assume we have a dscl with an sclbase of <http://dscl1.operator.org> and

> an id of dscl1 It has a locally registered app with ID app1 (globally

> unique ;-) So it has a URI of

> <http://dscl1.operator.org/applications/app1>

>

> The DSCL registers with an NSCL with base <http://nscl.operator.org/> So

> there is now an scl resource with URI

> <http://nscl.operator.org/scls/dscl1> on the nscl (and one with

> <http://dscl1.operator.org/scls/nsclid> on the dscl) The DSCL announces the app1 to the NSCL.

> This will create a resource with the URI:

> <http://nscl.operator.org/scls/dscl1/applications/app1Annc>

> (note that there are currently no rules on how the annc resource should be called, it might also be called app1, if you like, which I prefer actually).

>

> The application can now decide to create a local container in the dscl and one in the nscl.

> The one in the dscl is called localContainer and is created at this URI:

> <http://dscl1.operator.org/applications/app1/containers/localContainer>

>

> The one in the nscl is called remoteContainer and is created at this URI:

> <http://nscl.operator.org/scls/dscl1/applications/app1Annc/containers/r>

> emoteContainer (depending on the naming rules for announced resources,

> it may be prohibited to call this localContainer as well, if the

> localContainer was already announced)

>

> The app creates the remote container at that location (under the announcement) in order to automatically remove that container when deannouncement takes place.

> (note that if it wanted to create the remote container with a different lifecycle from the announcement it could have been created at a different URI like <http://nscl.operator.org/containers/remoteContainer> or <http://nscl.operator.org/scls/dscl1/containers/remoteContainer>).

>

> The app also decides to announce the localContainer in the NSCL.

> So there is a new resource announced resouce is created here

> <http://nscl.operator.org/scls/dscl1/applications/app1Annc/containers/l>

> ocalContainerAnnc

>

> Now an NA does a discovery action on the NSCL.

> I provides the correct criteria (we've not discussed the searchStrings etc, but they are there).

> As a result of discovery the following resource URLs can be returned, provided they match the criteria:

> <http://nscl.operator.org/scls/dscl1> (note that discovery will NOT

> return <http://dscl1.operator.org> !)

> <http://dscl1.operator.org/applications/app1/containers/localContainer>

> (since it is announced)

> <http://dscl1.operator.org/applications/app1> (since it is announced)

> <http://nscl.operator.org/scls/dscl1/applications/app1Annc/containers/r>

> emoteContainer (since this lives on the NSCL)

>

>

> So far so good.

> Nothing uncontroversial. We decided that the discovery would NOT reveal the announced resources themselves.

>

> In contrast

> A discovery on DSCL1 could return the following URLs

> <http://dscl1.operator.org/applications/app1>

> <http://dscl1.operator.org/applications/app1/containers/localContainer>

> <http://dscl1.operator.org/scls/nsclid>

>

> I.e., discovery on the NSCL will never give back URL of remote resources on the NSCL.

>

> Now for the actual questions.

>

> Question

> When not using discovery but just RETRIEVE on the collection resource, what should be returned?

> 1a) return the announced resource URL.

> 1b) return the same URL as in case of discovery , i.e., the announcing resource URL?

>

> I.e, if I RETRIEVE/GET

> <http://nscl.operator.org/scls/dscl1/applications>

> Will I get

> 1a) <http://nscl.operator.org/scls/dscl1/applications/app1Annc>

> Or

> 1b) <http://dscl1.operator.org/applications/app1>

>

> 1b would be more inline with the discovery. But this would mean that the remains unaware of the announced resource child.

> And since the announced resource can have children (like <http://nscl.operator.org/scls/dscl1/applications/app1Annc/containers/remoteContainer>) I think it makes more sense to give back the announced resource and let the issuer do the de-referencing (i.e., use the link attribute in the announced resource).

>

> It could be argued that for discovery we could also give back the announced resource and let the indirection be done by the issuer, to be consistent.

*The answer from Yongjing was this:*

> As for question 1, I also prefer 1a). The hosting SCL doesn't have to have more intelligence than doing a normal RETRIEVE, i.e. returning the reference of the sub-resource as it is.

#### Announced Resource

An announced resource shall point to the original resource hosted in another SCL. The announced resource is an actual resource which consists of only a limited set of attributes, which are the searchStrings, the link to the original resource and the access right. The purpose of the announced resource is to facilitate a discovery of the original resource when querying the announced-to SCL, so that the issuer of the discovery does not have to contact all SCLs in order to find the resources. An announced resource itself shall only be visible when it is directly accessed via its full URI. During discovery a direct reference to the original resource shall be returned. Only locally created resources can be announced.

Removing an announced resource, for example due to deregistration of an SCL (which correspond to a removal of the parent SC L resource), does not remove the original resource, but does remove all the children of announced resource Reversely, when the original resource is removed, it is the responsibility of the original SCL, where the original resource is hosted, to remove the announced resource. If this is not done (e.g. because the original SCL is offline), the announcement resource shall be removed when it expires.

*Editor notes: the above means that an announced resource cannot be dynamically discovered. We can consider doing a discover with resource type set to announced resource type.*

NOTE: There are collections with:

1) only real resources,

2) a mix of real resources and announced resources

3) only announced resources.

A resource of the same type (e.g. *<application>* resources) might have different content depending on where in the tree it is located. There are different ways of representing this, i.e. either by defining different resources or by defining one collection resource that contain both types of children. The latter solution is chosen, but it is indicated in each case whether which child resources are allowed.

#### NotificationChannel Resource

A NotificationChannel resource shall be used by non-server capable client to receive asynchronous notifications. The notification channel is prepared to handle several mechanisms on how to receive these asynchronous notifications. However, currently only one mechanism is fully specified, which is the so-called “long polling” mechanism. This method is based on the server not responding to requests until a notification needs to be sent (or until a timeout occurs).

#### Discovery Resource

A discovery resource shall be used to allow discovery. It is used to retrieve the list of URI of resources matching a discovery filter criteria. It does not represent a real resource in the sense that it does not have a representation and it shall never be cached.

#### Collection Resource

This resource represents an abstract concept that is applicable to various resources in the resource structure. For details on the collection resources, see [6.2.2].

A collection resource normally has its own associated accessRightID and allows subscriptions on modification in the collection resource.

I.e. when resources contain a collection of similar sub-resources, this is modelled as a collection resource. There are several collection resources identified, e.g. the SCL resource mentioned above contains collection resources for *<group>* resources, for *<container>* and <locationContainer> resources, for *<application>* resources, for *<accessRight>* resources and for <mgmtObj> and <mgmtCmd> resources. A collection can container local resources and/or the corresponding announced resources. A collection resource representation contains the sub-resources by reference and it may also contain attributes.