

Make your Documentum REST clients hypermedia controlled

It has been more than one year since EMC released the 1st official [Documentum Platform REST Services](#). More developers nowadays are transiting their services over to the REST paradigm. There have been a lot of discussions on the web about the definition & benefits of the REST software development architecture. Among all advantages of the REST, hypermedia is almost at the heart (seeing Roy Fielding's post "[REST APIs must be hypertext-driven](#)"). It's not a requirement only for the REST services implementation, but also for the REST clients which consume a real RESTful services. Mike Amundsen in his talk "[Implementing Hypermedia Clients: It's Not Rocket Science - Programming - O'Reilly Media](#)" said "*Faithful hypermedia clients only need a starting URL and a (human) driver.*" This is where we start in this document -- Building your hypermedia controlled Documentum REST clients.

EMC REST architect [Jonathan Robie](#) in his blog "[EMC Documentum Platform REST Services Tutorial](#)" has introduced how to consume Documentum REST Services following the hypermedia driven model. This document much on the same is emphasizing the hypemedia.

URI Is The Implementation Thing

As you may have noticed, the new Documentum REST Services did not publish a [WADL](#) like description language to describe the hierarchy of implemented resources (we did so in [Content Management Interoperability Services \(CMIS\)](#) implementation). It's because the resource URI is a server implementation thing for the hypermedia REST APIs. In the other hand, by taking the advantage of REST loose coupling, the client implementation should not have the necessary to bookmark or construct the resource URIs on the client side. Hypermedia gives clients the wheel to design a workflow to consume resources in an independent way.

The Entry URI for Home Document

The root entry for Documentum REST Services is the Home Document Resource (Home Documents for HTTP APIs, IETF draft). Ideally, this is the **only** URI that clients should

Make your Documentum REST clients hypermedia controlled

remember. The home document resource lists all topmost resources available in the service implementation.

```
<resources xmlns="http://identifiers.emc.com/vocab/documentum"    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

Starting from this resource representation on, clients should follow the self-descriptive resource messages to discover the further resource capability. For example, following the resource relation "<http://identifiers.emc.com/linkrel/repositories>", clients can get to the repositories feed resource.

```
<link href="http://localhost/dctm-rest/repositories"/>

==>

GET http://localhost/dctm-rest/repositories HTTP/1.1

[customized HTTP headers]
```

Honoring Hypermedia

On each resource representation, there are one "links" node or multiple "link" nodes (for XML atom feeds and entries) telling clients what to do with the presented resource. These are the link relations for the specific resource. Clients should follow the link relation on the resource to do further operations.

Example 1 - From User To Home Cabinet

On the current user resource, clients look for a link relation "<http://identifiers.emc.com/linkrel/default-folder>" to find the link to the user's home cabinet resource.

```
<user xmlns="http://identifiers.emc.com/vocab/documentum"    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"    xsi:type="user">
```

Clients then invoke a HTTP GET method to the target URI to get the home cabinet.

```
<link rel="http://identifiers.emc.com/linkrel/default-folder"
      href="http://localhost/dctm-rest/repositories/sample/users/dmadmin/home"/>

==>

GET http://localhost/dctm-rest/repositories/sample/users/dmadmin/home
HTTP/1.1

[customized HTTP headers]
```

Example 2 - Get Thumbnails of Folder Documents

On the folder documents feed, clients look for a link relation "icon" in each entry to find the link to the document's thumbnail source.

```
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:dm="http://identifiers.emc.com/linkrel/default-folder" ...>
```

Clients then invoke a HTTP GET method to the target URI to download the thumbnail.

```
<link rel="icon"
      href="http://thumbnailserver/thumbsrv/getThumbnail?
path=000004d2\80\00\04\ce.jpg&store=thumbnail_store_01"/>

==>

GET http://thumbnailserver/thumbsrv/getThumbnail?
path=000004d2\80\00\04\ce.jpg&store=thumbnail_store_01 HTTP/1.1

[customized HTTP headers]
```

Example 3 - Get Created Resource

On the response of a resource creation operation, the server returns a "Location" header to indicate the link of the newly create resource. Clients should honor the status code and the "Location" header to retrieve the resource.

```
POST /documents HTTP/1.1

[customized headers]


<document type="dm_document">

  <properties>

    <object_name>readme.txt</object_name/>

  </properties>

</document>


HTTP/1.1 201 Created

Location: http://localhost/dctm-rest/repositories/sample/
documents/090004d28000511d

[various headers]
```

URI Template

The only exceptional case that clients need to manipulate a resource URI is the URI template. A URI Template is a compact sequence of characters for describing a range of Uniform Resource Identifiers through variable expansion (RFC6570). By so far, there is only one link relation in Documentum REST Services exposing the URI as the template, the link relation "<http://identifiers.emc.com/linkrel/dqi>" on a repository resource.

```
<repository xmlns="http://identifiers.emc.com/vocab/documentum" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

Make your Documentum REST clients hypermedia controlled

Following this link relation, clients can construct an actual DQL query resource URI.

```
<link rel="http://identifiers.emc.com/linkrel/dql"
      hreftemplate="http://localhost/dctm-rest/repositories/sample{?
dql,page,items-per-page}"/>

==>

GET http://localhost/dctm-rest/reposotiries/sample?dql=select%20*%20from
%20dm_cabinet&items-per-page=20 HTTP/1.1

[customized HTTP headers]
```

There have been several outstanding URI template libraries outside that can help you to parse and construct the URIs from a template.

Summary

Of course there are also disadvantages of hypermedia (for instance, clients need to plan for changes). But with the hypermedia driven development mode, both the client applications and resources will evolve and scale gracefully for a long-term investigation. You'll find more interesting articles about the REST & hypermedia style architecture on the web.

[Learn more about Documentum REST Services >>](#)