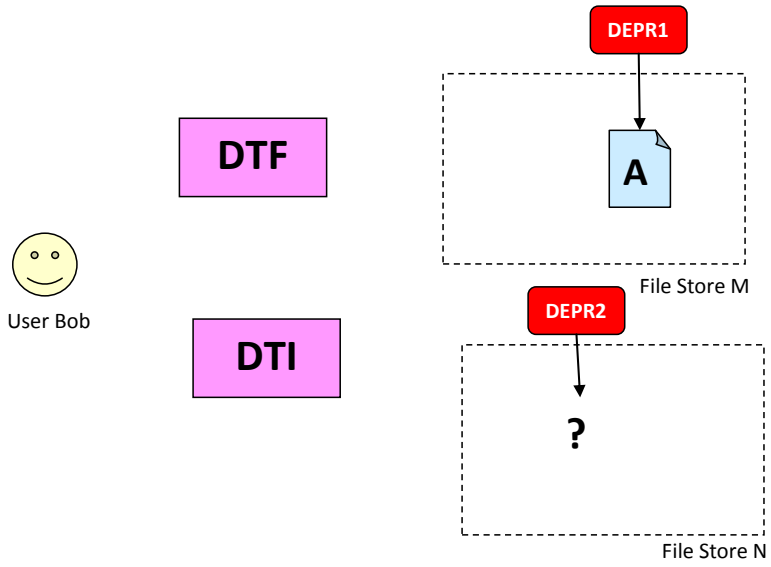
OGSA-DMI Use Case Example

NB: Eventually to go into main OGSA-DMI specification document.

Scenario A

This use case illustrates the operations that are invoked on the OGSA-DMI port types and the approximate XML documents passed between the various entities that are needed to transfer a file A from file store M to file store N.



Reference should be made to the current draft (22?) of the OGSA-DMI specification. This example uses the proposed security model that is not in the current draft but which was proposed by email and discussed on the last telecom. The security elements used below are indicative and should be replaced by more normative text by those that know what it should be!

Stages

1. Obtain a source DEPR (DEPR1) from a service K that knows about file store M, the location of file A and the user Bob requesting access to the file.

Comment [MA1]: This bit is out of scope of DMI right?

```
<tns:DataEPR>
    <wsa:Address> What is this the address of? </wsa:Address>
    <wsa:Metadata>
        <tns:Data
```

```
  protocol="http://www.ogf.org/ogsa-dmi/2006/03/im/protocol/gridftp-v20"
  url="gridftp://filestore-m.my.org/homes/bob/public_html/A.txt" >
         <tns:SerialisedX509GSIProxy>
         DER Encoded Text Stream
         </tns:SerialisedX509GSIProxy>
       </tns:Data>
       <tns:Data
  protocol="http://www.ogf.org/ogsa-dmi/2006/03/im/protocol/scp-v20"
  url="scp://filestore-m.my.org/homes/bob/public_html/A.txt" >
        <tns:UserNamePassword user="bob" password="secret"> />
       </tns:Data>
       <tns:Data
  Protocol="http://www.ogf.org/ogsa-dmi/2006/03/im/protocol/http/v11"
  url="http://filestore-m.my.org/~bob/A.txt" >
       </tns:Data>
    </wsa:Metadata>
</tns:DataEPR>
```

Comment [MA2]: I'm ok with the DEPR idea. However, the transport stuff is something that should probably not come from third parties. Service k, as you call it , should supply the raw Data addressing stuff. The transport stuff should be supplied by the DTF which decorates the bare DEPR or by a DMI related service.

This DEPR minting service sitting near file store M is saying that the file A.txt can be extracted from the file store by:

- Using a GSI proxy to access a GridFTP V2.0 server. The proxy will be of limited life and may have been provided by the Bob when he connected to the DEPR minting service. It could also be a proxy generated on demand by the minting service with a specific short-term life or usage restriction that only has rights (either in the proxy or through the access mapping on the GridFTP server) to retrieve the specific file. The GSI proxy is inserted into the DEPR as a text stream – similar to its storage on disc.
- Using a normal SCP server the file can be retrieved through the use of a username and password. These are presented in clear text in the file. Even if passed over TLS this is not ideal but an illustrative example. As this username and password is being used to access a specific file on a specific server then this could be a one-time username and password for this specific localized access.
- The data file is also accessible through a web server at an unprotected location.

2. Obtain a sink DEPR (DEPR2) from a service L that knows about file store N, the destination of file A into this file store and user Bob.

```
<tns:DataEPR>
    <wsa:Address> What is this the address of? </wsa:Address>
    <wsa:Metadata>
        <tns:Data
  protocol="http://www.ogf.org/ogsa-dmi/2006/03/im/protocol/gridftp-v20"
  url="gridftp://filestore-n.my.org/homes/users/bob21/A.txt" >
         <tns:SerialisedX509GSIProxy>
         DER Encoded Text Stream
         </tns:SerialisedX509GSIProxy>
       </tns:Data>
    </wsa:Metadata>
```

Comment [MA3]: Issue as to what this points to. In effect it's just a bit of space with nothing in it – not sure if that is a good idea.

Comment [MA4]: Again I feel this info should not come from third parties.

```
</tns:DataEPR>
```

This DEPR minting service sitting near file store N is saying that the file A.txt can only be placed into the file store at the specified location by:

- Using a GSI proxy to access a GridFTP V2.0 server. The proxy will be of limited life and may have been provided by the Bob when he connected to the DEPR minting service. It could also be a proxy generated on demand by the minting service with a specific short-term life or usage restriction that only has rights (either in the proxy or through the access mapping on the GridFTP server) to retrieve the specific file. The GSI proxy is inserted into the DEPR as a text stream – similar to its storage on disc.

3. Use the DTF to generate a DTI for me to manage the transfer. Now that I have my source and sink DEPRs I need to find a service implementing the DTF port type to undertake the transfer for me. The exact mechanisms used to discover this (and other services) is outside the scope of OGSA-DMI but could include mechanisms such as UDDI registries. I could examine the DEPRs to see what protocols such a DTF would need to understand to complete successfully… but I know that if does not understand the protocols or fails to find a match then it will throw a fault and not generate a DTI for the data transfer.

   To set up the data transfer I need to specify any constraints that I have. In this case not to start before 6pm on Friday 8th June (I'm submitting this at 5.30pm as I still have some other things to set up) and that once completed I want the DTI to remain in existence for at least 30s. I will be polling every 10s to check for completion so I will be able to explicitly observe the transfer coming to an end.

```
<tns:TransferRequirements>
    <tns:StartNotBefore>08Fri2007 1800</tns:StartNotBefore>
    <tns:StayAliveTime time="30"/>
</tns:TransferRequirements>
```

   With the source DEPR, the sink DEPR and the transfer requirements XML I invoke the **requestDataTransferInstance** operation on the DTF. On success I will get back the EPR of the DTI – on failure I will get a **NoMatchingProtocolFault** which will occur if there is no compatible protocol as reported in the sink and source DEPRs. If there is a compatible protocol in the DEPRs but it is not supported by the DTF & DTI then a **ProtocolNot SupportedFault** (specifying which protocol is not supported) is returned to the client.

   Implementation wise you could imagine the DTF looping over the protocols in the source DEPR trying to find compatible (i.e. the same) protocols in the sink DEPR. A plugin mechanism could be easily used allowing the server to iterate over known solutions that it found in a dynamic library path. On finding a valid match the plugin could generate a service instance (the DTI) for future interaction.

NB: One could imagine situations where intelligent DTF's could download a file using http to scratch space from a source and then upload it to the sink using GridFTP (say). Such behavior would be encapsulated in the DTF and the DTI implementations and would not be observed by the end user.

4. Assuming success, I now have a valid DTI which I can reference through the returned EPR. My data transfer will start at 6pm unless I specify otherwise. As I'm ready to go I invoke the **start** operation on the DTI.

5. I monitor the progress of the DTI by polling the service by calling the **getState** operation. It returns a URI and I loop waiting it to enters the **Done** state. Once I detect that it is **Done** I retrieve the state qualifiers (success/failure) and if it has failed I can see the cleanup status (clean/unclean/unknown).

6. Once I am happy that the DTI has completed and I have extracted any failure state that I need I can removed it with the destroy operation. If I do not destroy it explicitly it MAY get cleaned up automatically 30s after entering the done state (as specified in the transfer requirements).

Scenario B

Consider a scenario where OGSA-DMI is being used to support a simple file movement using a command line tool – dmi-cp – to copy a file from a source to a destination. In this case the file in question will use the GridFTP protocol and naming to specify the source and destination location.

1. The end-user executes the command to transfer the file from its current source location to the destination location:

   dmi-cp gridftp://source.server.org:/homes/bob/files/A.txt gridftp://sink.server.org/bob/B.txt

2. Before invoking the **requestDataTransferInstance** on the DTF I need to create the source and sink DEPRs. With the explicit file locations specified in the command line invocation, and by using the user's X.509 proxy certificate that exists in the client environment. The source DEPR would be:

```
<tns:DataEPR>
    <wsa:Address> What is this the address of? </wsa:Address>
    <wsa:Metadata>
        <tns:Data
  protocol="http://www.ogf.org/ogsa-dmi/2006/03/im/protocol/gridftp-v20"
  url="gridftp://source.server.org:/homes/bob/files/A.txt" >
         <tns:SerialisedX509GSIProxy>
         DER Encoded Text Stream
         </tns:SerialisedX509GSIProxy>
       </tns:Data>
    </wsa:Metadata>
```

```
</tns:DataEPR>
```

The sink DEPR would be:

```
<tns:DataEPR>
    <wsa:Address> What is this the address of? </wsa:Address>
    <wsa:Metadata>
        <tns:Data
  protocol="http://www.ogf.org/ogsa-dmi/2006/03/im/protocol/gridftp-v20"
  url="gridftp://sink.server.org:/bob/B.txt" >
         <SerialisedX509GSIProxy>
         DER Encoded Text Stream
         </SerialisedX509GSIProxy>
       </tns:Data>
    </wsa:Metadata>
</tns:DataEPR>
```

3. With the source and sink DEPRs in place the activity follows on from step 3 in Scenario A. Additional command line options (not shown) could be used to pass in values to the transfer requirements identified in step 3 of Scenario A.
4. The dmi-cp tool may have the capability to monitor and report on the transfer as it proceeds through the use of a progress bar or regular percentage updates. The tool would also report on success or failure of the DTI before exiting.

WS-Naming and the Data EPR

The role of the Data EPR is becoming increasingly confused. Its current use has evolved to capture information relating to the location of data and how it can be accessed. As such it implicitly represents a mapping between a logical data name and the real representation (i.e. location and authentication token to gain access) of the data. This information may be retrieved from some form of logical to real data location resolving service (part of the un-described activity in Scenario A) or provided explicitly (Scenario B). When used inside the DTF it does not seem to need the EPR wrapping.

Proposal: Remove the EPR wrapping from the DEPR and define a specific structure for encapsulates the mapping between a logical data name and its real locations. This data structure could still be retrieved as part of an EPR from (as yet undefined) mapping/resolver service but need not be.

```
<tns:LogicalData name="xsd:any" >
    <tns:Data protocol="xsd:any" url="xsd:any" />
        <tns:SerialisedX509GSIProxy>?
        <tns:UserNamePassword user="xsd:any" password="xsd:any">?
    </tns:Data>*
```

```
<tns:LogicalData/>
```

It is not clear what role WS-Naming might have in DMI. The focus of WS-Naming seems to be much more about detecting the equivalence of web service endpoints and obtaining new EPR's if the old ones are no longer available through a resolver port type. While DMI could work with WS-Naming endpoints there seems to be no requirement for them within DMI as all we do is populate any EPR using the metadata elements – in a similar manner to the WS-Naming specification.

Issues

- We have ways of saying the transfer should start now, at a specified time in the future but not 'when I tell you to start'.
- A lot of intelligence is being placed in the DEPR generation… not sure if we don't need to define this.
- DTF throws a fault and fails if protocols not understood.
- When the DTF finds from the DEPRs that multiple protocol solutions are available to undertake the data transfer there is an implementation dependent mechanism (encapsulated within the DTF) for selecting one of the viable protocols.