

MDFS Web Service

The methods of the MDFS are made rather for complex than simple operations due to the efficiency of the service because a lot of small requests cause big overhead. The service offers the following methods which will be described later:

- Create
- Update
- Delete
- Select
- Link
- Unlink

The first four methods are similar to the SQL's data manipulation language (DML) and the last two are used for a feature of MDFS which enables to create shortcuts in the data tree. These links are then treated rather on the system level than application level.

The nodes are identified by a node id, a numeric value which can be used when in the methods when one needs to refer to a specific node.

Create

This method saves a tree into the MDFS. The data to store (the tree itself) can be specified by one of the two formats:

- direct
- indirect

The direct format is that the data is represented natively in XML. This format is usable in the most cases. An example:

```
<myNode xmlns="myNamespace" aName="aValue" />
```

The second format, indirect specification, consists of the double marked XML file. This means, that the data in a form of tree is stored in another tree, an example which represents the same semantics as the above one:

```
<node>
  <namespace>myNamespace</namespace>
  <name>myNode</name>
  <attribute>
    <name>aName</name>
    <value>aValue</value>
  </attribute>
</node>
```

Furthermore, the create operation can save one request, as it is possible to mount (append) the just created tree under some other node.

Update

Takes the tree from MDFS and from the request and synchronizes them. This method could require two special meta-nodes which are:

- synchronization boundary marker
- leaf node marker

The first meta-node's meaning is that when the (recursive) synchronization algorithm reaches it, the synchronization will be not continued with iterating over the child nodes. In other words, the meaning is that there is no difference deeper from this point.

The second meta-node marks the leaf node and when the synchronization algorithm reaches a node with this mark, all its child nodes will be deleted (recursively). Again in other words - the mark identifies the new end of the tree and what there was before must be removed.

Possibility of acceleration: The synchronization can be accelerated by using a hash function which projects the new node and its data into a single (scalar) value which is then compared against the same value stored in MDFS.

Delete

This operation is simpler, comparing to create or update. Its purpose is to remove the specified node (or nodes) with all of the descendants which they have.

Select

The select method executes a query in the MDFS. As the MDFS can be treated also as an XML database, the best choice for the query language is XPath. The result of the query can be in various types:

- scalar number
- vector of numbers
- vector of strings
- vector of nodes

This set of types is a subset of the full {scalar, vector} x {number, string, node} combinations because we haven't determined yet how to get a scalar data or node - usually the nodes and theirs components are in an array (set). The single scalar value refers to the count() function.

Link, Unlink

These methods are used for one specific feature of MDFS. The link method expects three parameters, nodes to be linked and nodes which are the others linked to, however usually there will be a single node in both sets. The third parameter specifies the link type - strong or weak.

Links between the nodes represent other type of relation than the parent-child one, because child nodes are ordered and linked nodes are not ordered - we store only the information needed to link the nodes. When we delete the nodes, we must delete also the nodes which are linked in a weak relation to the deleted one. When a node gets deleted its relations will be removed and to remove the relations sooner we can use the Unlink method.