

# Directory Services Markup Language v2.0

## draft September 19, 2001

Deleted: Microsoft's Proposal for ↵

## 1. Introduction

The Directory Services Markup Language v1.0 (DSMLv1) provides a means for representing directory structural information as an XML document.<sup>1</sup>

DSMLv2 goes further, providing a method for expressing directory queries and updates (and the results of these operations) as XML documents. DSMLv2 documents can be used in a variety of ways. For instance, they can be written to files in order to be consumed and produced by programs, or they can be transported over HTTP to and from a server that interprets and generates them.

DSMLv2 functionality is motivated by scenarios including:

- A smart cell phone or PDA needs to access directory information but does not contain an LDAP client.
- A program needs to access a directory through a firewall, but the firewall is not allowed to pass LDAP protocol traffic because it isn't capable of auditing such traffic.
- A programmer is writing an application using XML programming tools and techniques, and the application needs to access a directory.

In short, DSMLv2 is needed to extend the reach of directories.

DSMLv2 is not required to be a strict superset of DSMLv1, which was not designed for upward-compatible extension to meet new requirements. However it is desirable for DSMLv2 to follow the design of DSMLv1 where possible.

## 2. Design Approach

DSMLv2 focuses on extending the reach of LDAP directories. Therefore, as in DSMLv1, the design approach is not to abstract the capabilities of LDAP directories as they exist today, but instead to faithfully represent LDAP directories in XML. The difference is that DSMLv1 represented the *state* of a directory while DSMLv2 represents the *operations* that an LDAP directory can perform and the results of such operations.

Therefore the design approach for DSMLv2 is to express LDAP requests and responses as XML document fragments. For the most part DSMLv2 is a systematic translation of LDAP's ASN.1 grammar (defined by RFC 2251) into XML-Schema. Thus, when a DSMLv2 element name matches an identifier in LDAP's ASN.1 grammar, the named element means the same thing in DSMLv2 and in LDAP.

Deleted: s

<sup>1</sup> DSMLv1.0 refers to the first DSML specification (<http://www.dsml.org/1.0/dsml.html>) and its XML name space (xmlns=http://www.dsml.org/dsml).

DSMLv2 is defined in terms of a set of XML fragments that are used as payloads in a transport binding. A transport binding defines how the DSMLv2 XML fragments are sent as requests and responses in the context of a specific transport such as SOAP, SMTP, or a simple data file.

DSMLv2 defines two normative transport bindings: 1) a SOAP binding is defined in section 6; and 2) a file binding that serves as the DSMLv2 analog of LDIF is defined in section 7. The rules for defining other DSMLv2 compliant transport bindings are found in section 8.

The simple correspondence between LDAP and DSMLv2 has compelling advantages. However there are a few places where it makes sense for DSMLv2 to diverge from LDAP:

1. An LDAP application associates a security principal with an LDAP connection by issuing a Bind request – or, in the SASL Bind case, by issuing as many successive Bind requests as needed to complete the authentication. A DSMLv2 document can be transported via a variety of mechanisms, so the document itself is not used to authenticate the requestor. DSMLv2 includes a limited Bind request that may be used to associate a security principal with a collection of DSMLv2 operations.
2. LDAP does not include a method of grouping operations to be expressed in a single request. DSMLv2 allows multiple LDAP operations to be expressed in one request document, and by specifying a simple positional correspondence between individual requests within a request document and individual responses within a response document. This change is explained in Section 4 below.
3. In LDAP, a single search request typically generates multiple responses, closed by a *searchResDone* response. To enable the positional correspondence between requests and responses mentioned above, DSMLv2 provides for a transport binding to wrap the complete set of related search responses into a single *searchResponse* element containing the individual LDAP responses to a search request.
4. DSMLv1 specifies an encoding of directory data. There's value in following DSMLv1 where possible, so DSMLv2 uses DSMLv1's encoding of directory data within search responses, add requests, etc.
5. The systematic translation of RFC 2251 results in a redundant level of nested element, the *LDAPMessage*. DSMLv2 eliminates this extra level.
6. Defaulting works more naturally in XML documents than in ASN.1 structures, so DSMLv2 uses defaulting in a few places where LDAP doesn't. In DSMLv2 the string-valued elements *matchedDN* and *errorMessage* (from *LDAPResult* in LDAP) and *attributes* (from *SearchRequest* in LDAP) are optional, and when absent are treated as the empty string. The *sizeLimit*, *timeLimit*, and *typesOnly* elements (from *SearchRequest* in LDAP) default to 0, 0, and FALSE respectively.

In the following the term *provider* is used to refer to that element of the client that implements a transport binding on behalf of the client. Certain error conditions are detected by the provider without any necessary interaction with a server.

**Deleted:** DSMLv2 does not include a Bind request.

**Deleted:** is

**Deleted:** a request-response protocol, but DSMLv2 will be used in a request-response manner, whether by producing and consuming files, sending HTTP requests or responses, or whatever.

**Deleted:** takes this difference into account by

**Deleted:** ing

**Deleted:** from a connection-based asynchronous protocol to a simple pairing of request and response documents is the most significant difference between LDAP and DSMLv2; its effects are

**Deleted:** Going to a request-response model makes the LDAP *messageID* field and *Abandon* request redundant, so DSMLv2 drops them.

**Deleted:** s

**Deleted:** <#> LDAP's encoding of search filters (*Filter*) and of the list of attributes returned by a search (*AttributeDescriptionList*) are quite unfriendly. Fortunately, RFC 2255 (LDAP URL format) specifies alternative representations that are friendlier, so DSMLv2 uses those instead of following RFC 2251. ¶

**Formatted:** Bullets and Numbering

**Formatted:** Font: Georgia, Font color: Teal

**Formatted:** Font: Italic

### 3. DSMLv2 URN

The base URN for DSMLv2 is:

`urn:oasis:dsml:names:tc:DSML:2:0`

There are two namespaces: the core namespace consisting of the individual operations and the batch namespace including definitions of a request envelope, a response envelope and an envelope grouping the entries, references and result of a search operation.

Example of using the DSMLv2 URN:

```
<batch request xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns:dsml="urn:oasis:dsml:names:tc:DSML:2:0:core">
```

### 4. Top-Level Structure

There are two types of DSMLv2 document: the *request* document and the *response* document. In a DSMLv2-based interaction between a client and a server<sup>2</sup> there is a pairing of requests and responses: For each request document submitted by the client there is one response document produced by the server.

The top-level element of a request fragment may be a *DsmlRequest* or a *BatchRequest* and the top-level element of a response fragment may be a *DsmlResponse* or a *BatchResponse*.

A *BatchRequest* contains zero, one, or many individual request elements, while a DSMLv2 *BatchResponse* consists of zero, one or many individual response elements.

In a valid *batch* request-response pair, there is a straightforward positional correspondence between individual requests within a request document and individual responses within a response document: The  $n^{\text{th}}$  response element corresponds to the  $n^{\text{th}}$  request element. For instance, if the third response element is a delete response, then it corresponds to the third request element, a delete request.

A valid *batch* request-response pair can have fewer responses in the response document than requests in the request document. This can happen due to a syntax error in the request document or due to a failure while processing a request (more on these conditions below.) The correspondence stated above still holds: The  $n^{\text{th}}$  response element corresponds to the  $n^{\text{th}}$  request element.

<sup>2</sup> Clients and programs (not servers) might also perform DSMLv2-based interactions, e.g. in which the client provides a file holding a document as input and the program produces a file containing a document as output. But it is clumsy to say "program or server" throughout the document, so we say "server".

Deleted: URI

Deleted: URI

Deleted: http://www.dsmlorg/DSML/v2

Formatted: Default Paragraph Font

Deleted: URI

Deleted: dsmlEnvelopeR

Deleted: http://www.dsml.org/DSML/v2

Deleted: ¶

Deleted: document

Deleted: is

Formatted: Font: Italic

Deleted: dsmlEnvelopeRequest

Deleted: document

Deleted: is

Deleted: d

Deleted: smlEnvelope

Formatted: Font: Italic

Deleted: dsmlEnvelopeRequest

Deleted: response

Formatted: Font: Italic

Deleted: document

Here is an example of a valid batch request-response pair<sup>3</sup>:

#### DSML Request Document:

```
<batch:request xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns:dsml="urn:oasis:dsml:names:tc:DSML:2:0:core">
  <modifyRequest>...</modifyRequest>
  <addRequest>...</addRequest>
  <delRequest>...</delRequest>
  <addRequest>...</addRequest>
</batch:request>
```

Deleted: <dsmlEnvelopeRequest xmlns="http://www.dsml.org/DSML/v2">

Deleted: dsmlEnvelopeRequest

#### DSML Response Document:

```
<batch:response xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns:dsml="urn:oasis:dsml:names:tc:DSML:2:0:core">
  <modifyResponse>...</modifyResponse>
  <addResponse>...</addResponse>
  <delResponse>...</delResponse>
  <addResponse>...</addResponse>
</batch:response>
```

Deleted: <dsmlEnvelopeResponse xmlns="http://www.dsml.org/DSML/v2">

Deleted: dsmlEnvelopeResponse

XML-Schema for BatchRequest is [here](#) and for BatchResponse is [here](#).

Deleted: dsmlEnvelopeRequest

Deleted: dsmlEnvelopeResponse

A BatchRequest containing zero request elements is a valid request; the valid response is a BatchResponse containing zero response elements. Such a batch request-response pair can be used to verify that a server is capable of processing DSMLv2 documents.

Formatted: Font: Italic

Deleted: dsmlEnvelopeRequest

Deleted: dsmlEnvelopeResponse

## Syntax Errors

A client may produce a request document that is syntactically incorrect, i.e. does not conform to the XML-Schema for a top-level XML fragment as defined above. In this case the DSMLv2 provider produces a response document, to aid in debugging the client. If the server detects the syntax error before performing any directory operations on behalf of the client, the response has the form:

Deleted: dsmlEnvelopeRequest.

Deleted: -based

Deleted: server

Deleted: still

Deleted: Document

#### DSML Response – Syntax error in request:

```
<batch:response xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core">
  <errorResponse type="notAttempted">
    <message>Unknown element 'bogusRequest' line 87 column 4 </message>
  </errorResponse>
</batch:response>
```

Deleted: <dsmlEnvelopeResponse xmlns="http://www.dsml.org/DSML/v2">

Deleted: dsmlEnvelopeResponse

The *errorResponse* element contains details about the error.

**SHOULD THIS BE PERMITTED? IT WOULD SEEM THAT A VALID BatchRequest should be a requirement for a provider to perform any of the individual operations.**

Formatted: Font: Italic

<sup>3</sup> Section 5 below gives details of the contents of request and response elements such as *modifyRequest* and *addResponse*.

If the server performs one or more directory operations on behalf of the client before detecting the syntax error, the server's response contains the response element for each operation that it performed, followed by an *errorResponse* element. For instance,

**DSML Request containing syntax error:**

```
<batch:request xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core">
  <modifyRequest>...</modifyRequest>
  <addRequest>...</addRequest>
  <bogusRequest>...</bogusRequest>
  <addRequest>...</addRequest>
  ...
</batch:request>
```

- Deleted: Document
- Deleted: <dsmlEnvelopeRequest xmlns="http://www.dsml.org/DSML/v2">
- Deleted: dsmlEnvelopeRequest

**DSML Response Document – Syntax error in request:**

```
<batch:response xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core">
  <modifyResponse>...</modifyResponse>
  <addResponse>...</addResponse>
  <errorResponse type="notAttempted">
    <message>Unknown element 'bogusRequest' line 87 column 4</message>
  </errorResponse>
</batch:response>
```

- Deleted: <dsmlEnvelopeResponse xmlns="http://www.dsml.org/DSML/v2">
- Deleted: <errorResponse>Unknown element 'bogusRequest' line 4 column 3</errorResponse>
- Deleted: </dsmlEnvelopeResponse>

**Failures**

A client may produce a request document that is syntactically correct but that contains a request that *fails* when the provider executes it. Failure is defined as follows:

- The DSMLv2 provider was unable to connect to a server (represented as an errorResponse with a type="couldNotConnect".)
- The DSMLv2 provider connected to a server, but the server closed the connection without responding to the request<sup>4</sup> (represented as an errorResponse with a type="connectionClosed".)
- The server returned a LDAPErrorCode other than 0 ("success"), 6 ("compareTrue"), 5 ("compareFalse"), or 10 ("referral").

- Deleted: server
- Deleted: server
- Deleted: the
- Deleted: LDAP
- Deleted: of -2, descr
- Formatted: Font: Italic
- Deleted: r
- Deleted: resultCode
- Formatted: Font: Not Italic
- Formatted: Font: Not Italic
- Deleted: server
- Deleted: the LDAP
- Deleted: LDAP
- Formatted: Font: Italic
- Deleted: resultCode of -3, descr
- Deleted: LDAP
- Deleted: resultCode
- Deleted: Document
- Deleted: <dsmlEnvelopeRequest xmlns="http://www.dsml.org/DSML/v2">

When a request execution fails, the server does not attempt to execute later requests within the document. The server produces a response element for each request element that was attempted, including the one that failed.

**DSML Request containing a request that fails**

```
<batch:request xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core">
```

<sup>4</sup> Obviously neither this nor the previous condition can happen if the DSMLv2 server is tightly integrated with a directory server, rather than communicating with a directory server via LDAP. On the other hand these conditions may certainly occur within the provider trying to interact with a tightly integrated server.

Deleted: server

Deleted: a server

```
<modifyRequest>...</modifyRequest>
<addRequest>...</addRequest>
<delRequest>...</delRequest>
<addRequest>...</addRequest>
</batch:request>
```

Deleted: </dsmlEnvelopeRequest>

### DSML Response – One request not attempted

```
<batch:response xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core">
  <modifyResponse>...</modifyResponse>
  <addResponse>...</addResponse>
  <errorResponse type="connectionClosed"/>
</batch:response>
```

Deleted: Document

Deleted: <dsmlEnvelopeResponse xmlns="http://www.dsml.org/DSML/v2">

Deleted: delResponse

Deleted: ¶  
<resultCode descr="connectionClosed">3</resultCode>¶  
</delResponse>

Deleted: </dsmlEnvelopeResponse>

Deleted: dsmlEnvelopeRequest

## Parallel processing

A BatchRequest element may contain the optional XML-attribute *processing*, which influences how the server can process the request elements. The valid values are: *sequential* and *parallel*. If this attribute is omitted, the default value is *sequential*.

Example:

```
<batch:request xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core"
  processing="parallel" >
  ...
</batch:request>
```

Deleted: <dsmlEnvelopeRequest xmlns="http://www.dsml.org/DSML/v2">

Deleted: </dsmlEnvelopeRequest>

In a BatchRequest with *processing*="sequential", the server must preserve sequential semantics, i.e. it behaves as already described. The effect of processing the BatchRequest must be as if the request elements were executed in the order they occur within the envelope<sup>5</sup>.

Deleted: dsmlEnvelopeRequest

Deleted: dsmlEnvelopeRequest

In a BatchRequest with *processing*="parallel", the server may execute the request elements in any order. This form of processing is useful when a request contains multiple updates and the client knows that the updates are independent, as might be the case when DSMLv2 is used to bulk-load a directory. It is also useful when a request contains multiple queries and no updates.

Deleted: dsmlEnvelopeRequest

The *processing*="parallel" option does not change the positional correspondence between request elements and response elements. Response elements are ordered according to the request document, not the order of execution.

## Resuming on error

A BatchRequest element may contain a second optional XML-attribute *onError*, which influences how the server responds to failures while processing request elements. The valid values are: *exit* and *resume*. If this attribute is omitted, the default value is *exit*.

Deleted: dsmlEnvelopeRequest

<sup>5</sup> A provider or server could detect that the document contains no updates and execute the queries in parallel without the client providing explicit directions, but is not required to do so.

Example:

```

<batch:request xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core"
  onError="resume" >
  ...
</batch:request>

```

Deleted: <dsmlEnvelopeRequest xmlns="http://www.dsml.org/DSML/v2">

Deleted: </dsmlEnvelopeRequest>

Deleted: dsmlEnvelopeRequest

In a *BatchRequest* with *onError*="exit", the server stops executing request elements as soon as one request element fails, i.e. when *processing*="sequential" it behaves as already described.

If *processing*="parallel" and *onError*="exit", the server stops initiating execution of new request elements as soon as one request element fails. Because of the parallelism, several executions might fail, and a request that the server did not attempt might precede a request that the server executed (with success or failure.) Because of the positional correspondence between requests and responses, the provider/server may need to return responses for requests that it did not attempt. If the provider does not attempt to execute a request element, but needs to provide a response in order to maintain positional correspondence, it generates an *errorResponse* element with a *type*="notAttempted".

Deleted: server

Deleted: resultCode of -1 (descr  
Deleted: )

Formatted: Font: Not Italic

Formatted: Font: Not Italic

Deleted: Document

DSML Request with parallel execution containing a request that fails:

```

<batch:request xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core"
  processing="parallel" onError="resume" >
  <modifyRequest>...</modifyRequest>
  <addRequest>...</addRequest>
  <delRequest>...</delRequest>
  <addRequest>...</addRequest>
</batch:request>

```

Deleted: <dsmlEnvelopeRequest xmlns="http://www.dsml.org/DSML/v2">

Deleted: </dsmlEnvelopeRequest>

DSML Response – two requests not successful

```

<batch:response xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core" >
  <modifyResponse>...</modifyResponse>
  <errorResponse type="notAttempted"/>
  <delResponse>
    <resultCode descr="noSuchObject">32</resultCode>
  </delResponse>
</batch:response>

```

Deleted: Document

Deleted: attempted

Deleted: <dsmlEnvelopeResponse xmlns="http://www.dsml.org/DSML/v2">

Deleted: add

Deleted: ¶  
<resultCode descr="notAttempted">-1</resultCode>¶  
</addResponse>

Deleted: </dsmlEnvelopeResponse>

Deleted: dsmlEnvelopeRequest

Deleted: server

In a *BatchRequest* with *onError*="resume", the server executes the remaining request elements even though one or more requests have failed. This form of processing is most useful when *processing*="parallel".

Even when *processing*="parallel", the syntax checking of a request document is performed sequentially. The provider does not attempt any requests that follow the first syntax error in the document. Therefore if a response document contains an *errorResponse* element, it is the final element in the document.

## 5. LDAP Operations

With the exception of *extendedRequest*, each *DsmlRequest* element contains:

- A *dn* attribute (as in DSMLv1) containing a distinguished name.
- Zero or more *control* elements representing LDAP Controls.

Deleted: *LDAP request*

Formatted: Font: Italic

Here are few examples of LDAP request elements:

```
<batch:response xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core">
  <modifyRequest dn="CN=Joe Smith, OU=Dev, DC=Example, DC=Com">
    ...
  </modifyRequest>
  <addRequest dn="OU=Sales,DC=Example, DC=Com">
    ...
  </addRequest>
  <delRequest dn="CN=Alice,OU=HR,DC=Example,DC=Com">
    <control>...</control>
    <control>...</control>
  </delRequest>
  <searchRequest>
    ...
    <control>...</control>
  </searchRequest>
</batch:response>
```

Deleted: <dsmlEnvelopeRequest xmlns="http://www.dsml.org/DSML/v2">

Deleted: </dsmlEnvelopeRequest>

Here is an example of an LDAP Control:

```
<control type="1.2.840.113556.1.4.619" criticality="true">
  <controlValue xsd:type="base64Binary">RFNNTHYyLjAgcm9ja3MhIQ==</controlValue>
</control>
```

Deleted: >␣<controlType>

Deleted: </controlType>␣<criticality>

Deleted: </criticality>

The *controlValue* element is base64 encoded.

XML-Schema for *control* is [here](#).

Here are few examples of LDAP response elements:

```
<batch:response xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core">
  ...
  <modifyResponse>
    <resultCode descr="unwillingToPerform">53</resultCode>
    <errorMessage>System Attribute may not be modified</errorMessage>
  </modifyResponse>

  <addResponse>
    <resultCode descr="success">0</resultCode>
  </addResponse>

  <addResponse>
    <resultCode descr="success">0</resultCode>
    <control>...</control>
    <control>...</control>
  </addResponse>

  ...
</batch:response>
```

Deleted: <dsmlEnvelopeResponse xmlns="http://www.dsml.org/DSML/v2">␣

Deleted: </dsmlEnvelopeResponse>␣



The *matchedDN* and *errorMessage* elements are optional and default to the empty string.

The *resultCode* element has an optional *descr* attribute. If the server supplies this attribute, its value is the RFC 2251 text representation of the result code (“success”, “operationsError”, etc.), supplied for debugging convenience.

Like LDAP Request elements, LDAP response elements may contain zero or more controls<sup>6</sup>.

The remainder of this section describes the encoding of each LDAP operation. Refer to RFC 2251 for the semantics of LDAP operations.

## 5.1 Modify

DSMLv2 specifies each attribute modification by attaching an *operation* attribute to a DSMLv1 *attr* element. As in LDAP, an *operation* can be *add*, *delete*, or *replace*.

Example of *modifyRequest*:

```
<batch:response xmlns:batch="urn:oasis:dsml:names:tc:DSML:2:0:batch"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="urn:oasis:dsml:names:tc:DSML:2:0:core">
  ...
  <modifyRequest dn="CN=Bob Rush,OU=Dev,DC=Example,DC=COM">
    <attr name="telephoneNumber" operation="replace">
      <value>536 354 2343</value>
      <value>234 212 4534</value>
    </attr>
    <attr name="sn" operation="replace">
      <value>Rush</value>
    </attr>
    <attr name="directReport" operation="add">
      <value>CN=John Smith, DC=microsoft, DC=com</value>
    </attr>
  </modifyRequest>
</batch:request>
```

Deleted: <dsmlEnvelopeRequest xmlns="http://www.dsml.org/DSML/v2">

Deleted: </dsmlEnvelopeRequest>

XML-Schema for *modifyRequest* is [here](#).

Example of *modifyResponse*:

```
<modifyResponse>
  <resultCode code="53" descr="unwillingToPerform" />
  <errorMessage>System Attribute may not be modified</errorMessage>
</modifyResponse>
```

Deleted: >53</resultCode

*modifyResponse* is an [LDAPResult](#).

<sup>6</sup> With the exception of *searchResponse*, for reasons that will become clear in Section 5.2 .

## 5.2 Search

The DSMLv2 search encoding is based on the LDAP search encoding, but with some changes as described in Section 2. In the *searchRequest* encoding:

- *baseObject*. Following DSMLv1 conventions, the distinguished name of the search base is expressed as the XML attribute *dn*.

Example:

```
<searchRequest dn="OU=Marketing,DC=Example,DC=COM" />
```

- *sizeLimit*, *timeLimit*, *typesOnly*. These elements default to 0, 0, and FALSE respectively.
- *filter*. Expressing an RFC 2251 filter as nested XML elements would be complex. So the DSMLv2 *filter* element uses RFC 2255 (LDAP URL) encoding.

Example:

```
<filter>(objectClass=organizationalUnit)</filter>
```

- *attributes*. In RFC 2251, *attributes* is a sequence of attribute names, which would translate into a sequence of elements containing attribute names. So the DSMLv2 *attributes* element uses RFC 2255 encoding: a comma-separated list of attribute names. *attributes* defaults to the empty string, which is the RFC 2255 encoding of an empty LDAP *attributes* sequence<sup>7</sup>.

Example:

```
<attributes>sn,givenName,title</attributes>
```

*SearchRequest* example:

```
<searchRequest dn="ou=Marketing,dc=microsoft,dc=com"
  scope="singleLevel"
  derefAliases="neverDerefAliases"
  sizeLimit="1000">
  <filter><substrings><final_desc="sn">john</final></substrings></filter>
  <control controlType="1.2.840.113556.1.4.612" criticality="true">
    <controlValue xsd:type="base64Binary">U2VhcmNoIFJlcXVlc3QgRXhhbXBsZQ== </controlValue>
  </control>
  <control controlType="1.2.840.113556.1.4.643" criticality="true">
    <controlValue xsd:type="base64Binary">TWljcm9zb2Z0IEFjdGkiZSBEaXJ3Rvcnk=
  </controlValue>
  </control>
</searchRequest>
```

XML-Schema for *searchRequest* is [here](#).

<sup>7</sup> An empty LDAP *attributes* sequence requests all user attributes, so an empty DSMLv2 *attributes* string does the same thing.



The response to a *searchRequest* is logically called a *searchResponse*. According to RFC 2251, a search response contains a) zero to many *searchResultEntry*, b) zero to many *searchResultRef* and c) one *searchResultDone*. DSMLv2.0 permits wrapping all of these related elements into one *searchResponse* envelope.

Deleted: s

In the *searchResultEntry* encoding:

- DSMLv2 borrows the DSMLv1 directory-entry encoding for *searchResEntry*.
- *searchResEntry* may have zero or more LDAP controls, consistent with RFC 2251.

*searchResultEntry* (with terminating *searchResultDone*) example:

```
<searchResponse>
  <searchResultEntry dn="OU=Development,DC=Example,DC=COM" >
    <attr name="allowedAttributeEffective" ref="urn:active-directory-schema" >
      <value>description</value>
      <value>ntSecurityDescriptor</value>
      <value>wwwHomepage</value>
    </attr>
  </searchResultEntry >
  <searchResultEntry dn="CN=David,OU=HR,DC=Example,DC=COM" >
    <attr desc="objectclass"><value>person</value></attr>
    <attr desc="sn"><value>Johnson</value></attr>
    <attr desc="givenName"><value>David</value></attr>
    <attr desc="title"><value>Program Manager</value></attr>
  </searchResultEntry >
  <searchResultEntry dn="CN=JSmith,OU=Finance,DC=Example,DC=COM" >
    <attr desc="objectclass"><value>top</value></attr>
    <attr desc="objectclass"><value>person</value></attr>
    <attr desc="objectclass"><value>organizationalPerson</value></attr>
    <attr name="sn"><value>Smith</value></attr>
  </searchResultEntry >
  <searchResultDone >
    <resultCode code="0">
      <control controlType="1.2.840.113556.1.4.621" criticality="false">
        <controlValue xsd:type="base64Binary">U2VhcmNoIFJlcXVlc3QgRXhhbXBsZQ==</controlValue>
      </control>
    </searchResultDone >
  </searchResponse >
```

- Deleted: name
- Deleted: name
- Deleted: name
- Deleted: <objectclass>
- Deleted:
- Deleted: <oc-value>top</oc-value>
- Deleted: <oc-value>person</oc-value>
- Deleted: <oc-value>organizationalPerson</oc-value>¶<br></objectclass>¶
- Deleted: >¶<br><r
- Deleted: >
- Deleted: </resultCode
- Deleted: >¶<br><
- Deleted: >
- Deleted: </controlType>¶<br><
- Deleted: >
- Deleted: </criticality>
- Deleted: ldapURL
- Deleted: ldapURL
- Deleted: ldapURL
- Deleted: ldapURL

XML-Schema for *searchResultEntry* is [here](#).

*searchResultReference* example:

```
<searchResponse>
  <searchResultReference>
    <ref>ldap://srv01.example.com/OU=Marketing,DC=Example,DC=COM</ref>
    <ref>ldap://srv05.fabrikam.com/DC=Fabrikam,DC=COM</ref>
  </searchResRef>
  ...
</searchResponse >
```

XML-Schema for *searchResultReference* is [here](#).

*searchResultDone* is illustrated above in the *searchResultEntry* example. *searchResultDone* is an *LDAPResult*.

### 5.3 Add

A DSMLv2 *addRequest* uses the DSMLv1 encoding style to describe the object to be added.

Example of *addRequest*:

```
<addRequest dn="CN=Alice,OU=HR,DC=Example,DC=COM">
  <attr desc="objectclass"><value>top</value></attr>
  <attr desc="objectclass"><value>person</value></attr>
  <attr desc="objectclass"><value>organizationalPerson</value></attr>
  <attr name="sn"><value>Johnson</value></attr>
  <attr name="givenName"><value>Alice</value></attr>
  <attr name="title"><value>Software Design Engineer</value></attr>
</addRequest>
```

Deleted: <objectclass>␣  
 <oc-value>top</oc-  
 value>␣  
 <oc-value>person</oc-  
 value>␣  
 <oc-  
 value>organizationalPerson</  
 oc-value>␣  
 </objectclass>␣

XML-Schema for *addRequest* is [here](#).

Example of *addResponse*:

```
<addResponse>
  <resultCode code="0">
    <errorMessage>completed</errorMessage>
  </addResponse>
```

Deleted: >␣  
 <  
 Deleted: >  
 Deleted: </resultCode

*addResponse* is an [LDAPResult](#).

### 5.4 Delete

Examples of *delRequest*:

```
<delRequest dn="CN=Bob,OU=HR,DC=Example,DC=COM" />
<delRequest dn="OU=HR,DC=Example,DC=COM" >
  <control controlType="1.2.840.113556.1.4.805" />
</delRequest>
```

Deleted: >␣  
 <  
 Deleted: >  
 Deleted: </controlType  
 Deleted: e  
 Deleted: </control>␣

XML-Schema for *delRequest* is [here](#).

Example of *delResponse*:

```
<delResponse matchedDN="OU=HR,DC=Example,DC=COM">
  <resultCode code="32" descr="noSuchObject" />
  <errorMessage>DSEDL::230234</errorMessage>
</delResponse>
```

Deleted: >␣  
 <  
 Deleted: 32</resultCode>␣  
 < matchedDN>OU=HR,DC=Ex  
 ample,DC=COM</matchedDN  
 >

*delResponse* is an [LDAPResult](#).

## 5.5 ModifyDN

Example of *modDNRequest*.

```
<modDNRequest dn="CN=Alice Johnson,DC=Example,DC=COM"
  newrdn="Alice Weiss"
  deleteoldrdn="true"
  newSuperior="OU=Marketing,DC=Example,DC=COM"/>
```

XML-Schema for *modDNRequest* is [here](#).

Example of *modDNResponse*.

```
<modDNResponse>
  <resultCode code="0" descr="success"/>
</modDNResponse>
```

*modDNResponse* is an [LDAPResult](#).

Deleted: >|  
<  
Deleted: >  
Deleted: </newrdn>  
Deleted: <  
Deleted: >  
Deleted: </deleteoldrdn>  
Deleted: <  
Deleted: >  
Deleted: </newSuperior  
Deleted: </modDNRequest>|  
Deleted: >0</resultCode

## 5.6 Compare

Example of *compareRequest*:

```
<compareRequest dn="CN=Johnson,OU=HR, DC=Example,DC=COM">
  <attr name="sn"><value>Johnson</value></attr>
</compareRequest>
```

XML-Schema for *compareRequest* is [here](#).

Example of *compareResponse*:

```
<compareResponse>
  <resultCode code="6" descr="compareTrue"/>
</compareResponse>
```

Deleted: >6</resultCode

The *compareResponse* is an [LDAPResult](#).

## 5.7 Extended Operation

Example of *extendedReq*:

```
<extendedReq>
  <requestName>1.3.563.52.424</requestName>
  <requestValue xsd:type="base64Binary">TFNNTHYyLjAgcm9ja3MhIQ==</requestValue>
</extendedReq>
```

The *requestValue* element uses base64 encoding.

XML-Schema for *extendedReq* is [here](#).

Example of *extendedResp*:

```
<extendedResp>
  <resultCode>0</resultCode>
  <response>RFNNTHyLjAgcm9ja3MhIQ==</response>
</extendedResp>
```

The *response* element uses base64 encoding.

XML-Schema for *extendedResp* is [here](#).

Deleted: Process

## 6. SOAP Transport Binding

The following describes a method of using SOAP over HTTP/1.1 and HTTPS/1.1 as a transport for DSML v2 requests and responses. The version of SOAP for this binding is SOAP 1.1. The use of other transports for SOAP, e.g., SMTP, would be subject to additional considerations not addressed here.

The namespace for the core DSMLv2 is "urn:oasis-open:dsml:2" and for the Batch elements is "urn:oasis-open:dsml:2:batch." These namespaces are used at the top-level element of the <body> of each SOAP request and response. Default namespace designations MAY be used.

All SOAP requests and responses in this binding MUST use the xml encoding "UTF-8".

Each SOAP request is either a <batch:request> as defined in DSML\_Batch.xsd or a DSMLRequest as defined in the core DSMLv2.xsd. Any DSMLv2/SOAP compliant server MUST accept all defined forms of request. A SOAP node SHOULD indicate in the 'SOAPAction' header field the element name of the top-level element in the <body> of the SOAP request. Note that sending a single DSMLRequest (i.e., one that is not contained within a <batch:request>) is most effective if "Connection: Keep-Alive" is used on the underlying HTTP/HTTPS connection.

Each SOAP response is either a <batch:response> or a <batch:searchResponse> as defined in DSML\_Batch.xsd or a DSMLResponse as defined in the core DSMLv2.xsd. The response to a <dsml:searchRequest> is always a <batch:searchResponse>

If a DSML v2 response contains an <dsml:errorResponse> then a SOAP fault element with the faultcode: 'Client' is returned and the DSML response is contained in the 'detail' of the fault element.

For example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<se:Envelope xmlns:se='http://schemas.xmlsoap.org/soap/envelope'>
  <se:Header/>
  <se:Body>
    <se:Fault>
```

Deleted: DSMLv2 does not define a protocol to transport DSML documents. A DSMLv2 document can be transported via a variety of mechanisms. If a network transport is needed, one recommended approach is SOAP. As defined in the SOAP 1.1 specification, SOAP can use many different protocols to transport SOAP messages, such as http and others. ↩  
↩  
Example of DSMLv2 document in a SOAP Request, Response and a SOAP Fault.

```

<faultcode>se:Client</faultcode>
<faultstring>DSML Client error</faultstring>
<detail>
  <dsml:errorResponse xmlns:dsml="urn:oasis-open:dsml:2"
    type="couldNotConnect">
    <dsml:message>Cannot connect to a DSML server</dsml:message>
  </dsml:errorResponse>
</detail>
</se:Fault>
</se:Body>
</se:Envelope>

```

```

Deleted: <!-- **** DSML
Request ***** -->
<SOAP-ENV:Envelope
xmlns:SOAP-
ENV="http://schemas.xmlsoap
.org/soap/envelope/" SOAP-
ENV:encodingStyle="http://sc
hemas.xmlsoap.org/soap/enco
ding/">
  <SOAP-ENV:Body
xmlns:dsml="http://www.dsml
.org/dsml/v2">
    <dsml:dsmlEnvelopeRequest>
      <dsml:modifyRequest>...</ds
ml:modifyRequest>
      <dsml:addRequest>...</dsml:
addRequest>
    </dsml:dsmlEnvelopeRequest
>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

If a DSML v2 response contains an LDAPResult with a ResultCode corresponding to any LDAPErrorCode other than: 'success', 'compareFalse' or 'compareTrue' then a SOAP fault element with the faultcode: 'Server' is returned and the DSML response is contained in the 'detail' of the fault element. For example:

```

<?xml version="1.0" encoding="UTF-8" ?>
<se:Envelope xmlns:se="http://schemas.xmlsoap.org/soap/envelope">
  <se:Header/>
  <se:Body>
    <se:Fault>
      <faultcode>se:Server</faultcode>
      <faultstring>DSML Server error</faultstring>
      <detail>
        <batch:response xmlns:dsml="urn:oasis-open:dsml:2"
          xmlns:batch="urn:oasis-open:dsml:2:batch">
          <dsml:modifyResponse>
            <resultCode code="53" descr="unwillingToPerform"/>
            <errorMessage>System Attribute may not be modified</errorMessage>
          </dsml:modifyResponse>
          <dsml:addResponse>
            <resultCode code="0"/>
          </dsml:addResponse>
          <dsml:naddResponse>
            <resultCode code="0" descr="success"/>
          </dsml:naddResponse>
          <control>...</control>
          <control>...</control>
        </batch:response>
      </detail>
    </se:Fault>
  </se:Body>
</se:Envelope>

```

```

Deleted:
<!-- **** DSML Response
***** -->
<SOAP-ENV:Envelope
xmlns:SOAP-
ENV="http://schemas.xmlsoap
.org/soap/envelope/" SOAP-
ENV:encodingStyle="http://sc
hemas.xmlsoap.org/soap/enco
ding/">
  <SOAP-ENV:Body
xmlns:dsml="http://www.dsml
.org/dsml/v2">
    <dsml:dsmlEnvelopeResponse
>
      <dsml:modifyResponse>
        ...</dsml:modifyResponse>
      <dsml:addResponse>...</dsml:
addResponse>
    </dsml:dsmlEnvelopeResponse
>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

```

</se:Envelope>

```

Deleted: 6.1 SOAP Header  
 DSMLv2 does not mandate any SOAP header elements, such as routing, reliable messaging, signing, and encryption.  
 6.2 SOAP Body  
 The maximum number of [ ]

SOAP headers SHOULD not be sent in requests and MUST be ignored by the recipient except in the case of a MustUnderstand. In the event that a recipient receives a request with a MustUnderstand header then a SOAP Fault of type MustUnderstand is returned.

Formatted: Border: Top: (Single solid line, Auto, 0.5 pt Line width, From text: 31 pt Border spacing: )

## 7. File Transport Binding

tbd

Deleted: ¶

Formatted: Bullets and Numbering

Formatted: Normal

## 8. Compliant Transport Bindings

Formatted: Bullets and Numbering

(1) A DSMLv2-compliant top-level entity is an element of type `dsmlEnvelopeRequest`, `dsmlEnvelopeResponse`, `dsmlRequest`, `dsmlResponse`, or `searchResponse`.

(2) A DSMLv2-compliant transport binding defines one or more transport document types.

(3) One or more DSMLv2-compliant entities may be embedded in a transport document.

Formatted: Bullets and Numbering

## 9. Appendix

### DSMLv2.0 XML Schema

```
<xsd:schematargetNamespace="urn:oasis:names:tc:DSML:2:0:core"
  xmlns="urn:oasis:names:tc:DSML:2:0:core"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <!-- DSML Requests -->
  <xsd:group name="DSMLRequest">
    <xsd:choice>
      <xsd:element name="bindRequest" type="BindRequest"/>
      <xsd:element name="searchRequest" type="SearchRequest"/>
      <xsd:element name="modifyRequest" type="ModifyRequest"/>
      <xsd:element name="addRequest" type="AddRequest"/>
      <xsd:element name="delRequest" type="DelRequest"/>
      <xsd:element name="modDNRequest" type="ModifyDNRequest"/>
      <xsd:element name="compareRequest" type="CompareRequest"/>
      <xsd:element name="abandonRequest" type="AbandonRequest"/>
      <xsd:element name="extendedReq" type="ExtendedRequest"/>
    </xsd:choice>
  </xsd:group>
  <!-- DSML Responses -->
  <xsd:group name="DSMLResponse">
    <xsd:choice>
      <xsd:element name="bindResponse" type="LDAPResult"/>
      <xsd:element name="searchResultEntry" type="SearchResultEntry"/>
      <xsd:element name="searchResultReference" type="SearchResultReference"/>
      <xsd:element name="searchResultDone" type="LDAPResult"/>
      <xsd:element name="modifyResponse" type="LDAPResult"/>
      <xsd:element name="addResponse" type="LDAPResult"/>
      <xsd:element name="delResponse" type="LDAPResult"/>
      <xsd:element name="modDNResponse" type="LDAPResult"/>
      <xsd:element name="compareResponse" type="LDAPResult"/>
      <xsd:element name="extendedResp" type="ExtendedResponse"/>
      <xsd:element name="errorResponse" type="ErrorResponse"/>
    </xsd:choice>
  </xsd:group>
  <!-- ***** DsmIDN ***** -->
  <xsd:simpleType name="DsmIDN">
    <xsd:restriction base="xsd:string"/>
  </xsd:simpleType>
  <!-- ***** DsmIRDN ***** -->
  <xsd:simpleType name="DsmIRDN">
    <xsd:restriction base="xsd:string"/>
  </xsd:simpleType>
  <!-- ***** Request ID ***** -->
  <xsd:simpleType name="RequestID">
    <xsd:restriction base="xsd:string"/>
  </xsd:simpleType>

```



```

</xsd:simpleType>
<!-- ***** OID ***** -->
<xsd:simpleType name="OID">
  <xsd:union memberTypes="NumericOID xsd:NMTOKEN"/>
</xsd:simpleType>
<xsd:simpleType name="NumericOID">
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="[0-2]\.[0-9]+\.[0-9+]" />
  </xsd:restriction>
</xsd:simpleType>
<!-- ***** ExtensionTypeID ***** -->
<xsd:simpleType name="ExtensionTypeID">
  <xsd:union memberTypes="NumericOID xsd:anyURI"/>
</xsd:simpleType>
<!-- ***** MAX Integer ***** -->
<xsd:simpleType name="MAXINT">
  <xsd:restriction base="xsd:unsignedInt">
    <xsd:maxInclusive value="2147483647"/>
  </xsd:restriction>
</xsd:simpleType>
<!-- ***** DSML Value **** -->
<xsd:simpleType name="DsmlValue">
  <xsd:union memberTypes="xsd:string xsd:base64Binary xsd:anyURI"/>
</xsd:simpleType>
<!-- ***** DSML Any external XML structure **** -->
<xsd:complexType name="DsmlANY">
  <xsd:sequence>
    <xsd:any />
  </xsd:sequence>
</xsd:complexType>
<!-- ***** DSML Control **** -->
<xsd:complexType name="Control">
  <xsd:sequence>
    <xsd:element name="controlValue" type="DsmlANY" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="type" type="ExtensionTypeID" use="required"/>
  <xsd:attribute name="criticality" type="xsd:boolean" use="optional" default="false"/>
</xsd:complexType>
<!-- ***** DSML Filter **** -->
<xsd:complexType name="Filter">
  <xsd:group ref="FilterGroup"/>
</xsd:complexType>
<xsd:group name="FilterGroup">
  <xsd:sequence>
    <xsd:choice>
      <xsd:element name="and" type="FilterSet"/>
      <xsd:element name="or" type="FilterSet"/>
      <xsd:element name="not" type="Filter"/>
      <xsd:element name="equalityMatch" type="AttributeValueAssertion"/>
      <xsd:element name="substrings" type="SubstringFilter"/>
      <xsd:element name="greaterOrEqual" type="AttributeValueAssertion"/>
      <xsd:element name="lessOrEqual" type="AttributeValueAssertion"/>
      <xsd:element name="present" type="AttributeDescription"/>
      <xsd:element name="approxMatch" type="AttributeValueAssertion"/>
      <xsd:element name="extensibleMatch" type="MatchingRuleAssertion"/>
    </xsd:choice>
  </xsd:sequence>
</xsd:group>
<xsd:complexType name="FilterSet">
  <xsd:sequence>
    <xsd:group ref="FilterGroup" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="AttributeValueAssertion">
  <xsd:sequence>
    <xsd:element name="value" type="DsmlValue"/>
  </xsd:sequence>
</xsd:complexType>

```

```

</xsd:sequence>
<xsd:attribute name="desc" type="OID"/>
</xsd:complexType>
<xsd:complexType name="AttributeDescription">
<xsd:attribute name="desc" type="OID"/>
</xsd:complexType>
<xsd:complexType name="SubstringFilter">
<xsd:sequence>
<xsd:element name="initial" type="DsmIValue" minOccurs="0"/>
<xsd:element name="any" type="DsmIValue" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element name="final" type="DsmIValue" minOccurs="0"/>
</xsd:sequence>
<xsd:attribute name="desc" type="OID"/>
</xsd:complexType>
<xsd:complexType name="MatchingRuleAssertion">
<xsd:sequence>
<xsd:element name="value" type="DsmIValue"/>
</xsd:sequence>
<xsd:attribute name="dnAttributes" type="xsd:boolean" use="optional" default="false"/>
<xsd:attribute name="matchingRule" type="OID" use="optional"/>
<xsd:attribute name="desc" type="OID" use="optional"/>
</xsd:complexType>
<!-- ***** DSML MESSAGE ***** -->
<xsd:complexType name="DsmIMessage">
<xsd:sequence>
<xsd:element name="control" type="Control" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="requestID" type="RequestID" use="optional"/>
</xsd:complexType>
<!-- ***** LDAP RESULT ***** -->
<xsd:simpleType name="LDAPErrorcode">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="success"/>
<xsd:enumeration value="operationsError"/>
<xsd:enumeration value="protocolError"/>
<xsd:enumeration value="timeLimitExceeded"/>
<xsd:enumeration value="sizeLimitExceeded"/>
<xsd:enumeration value="compareFalse"/>
<xsd:enumeration value="compareTrue"/>
<xsd:enumeration value="authMethodNotSupported"/>
<xsd:enumeration value="strongAuthRequired"/>
<xsd:enumeration value="referral"/>
<xsd:enumeration value="adminLimitExceeded"/>
<xsd:enumeration value="unavailableCriticalExtension"/>
<xsd:enumeration value="confidentialityRequired"/>
<xsd:enumeration value="saslBindInProgress"/>
<xsd:enumeration value="noSuchAttribute"/>
<xsd:enumeration value="undefinedAttributeType"/>
<xsd:enumeration value="inappropriateMatching"/>
<xsd:enumeration value="constraintViolation"/>
<xsd:enumeration value="attributeOrValueExists"/>
<xsd:enumeration value="invalidAttributeSyntax"/>
<xsd:enumeration value="noSuchObject"/>
<xsd:enumeration value="aliasProblem"/>
<xsd:enumeration value="invalidDNsyntax"/>
<xsd:enumeration value="aliasDereferencingProblem"/>
<xsd:enumeration value="inappropriateAuthentication"/>
<xsd:enumeration value="invalidCredentials"/>
<xsd:enumeration value="insufficientAccessRights"/>
<xsd:enumeration value="busy"/>
<xsd:enumeration value="unavailable"/>
<xsd:enumeration value="unwillingToPerform"/>
<xsd:enumeration value="loopDetect"/>
<xsd:enumeration value="namingViolation"/>
<xsd:enumeration value="objectClassViolation"/>
<xsd:enumeration value="notAllowedOnNonLeaf"/>

```

```

<xsd:enumeration value="notAllowedOnRDN"/>
<xsd:enumeration value="entryAlreadyExists"/>
<xsd:enumeration value="objectClassModsProhibited"/>
<xsd:enumeration value="affectMultipleDSAs"/>
<xsd:enumeration value="other"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="resultCode">
<xsd:attribute name="code" type="xsd:int"/>
<xsd:attribute name="descr" type="LDAPErrorCode" use="optional"/>
</xsd:complexType>
<xsd:complexType name="LDAPResult">
<xsd:complexContent>
<xsd:extension base="DsmIMessage">
<xsd:sequence>
<xsd:element name="resultCode" type="resultCode"/>
<xsd:element name="errorMessage" type="xsd:string" minOccurs="0"/>
<xsd:element name="referral" type="xsd:anyURI" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="matchedDN" type="DsmIDN" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ErrorResponse">
<xsd:sequence>
<xsd:element name="message" type="xsd:string" minOccurs="0"/>
<xsd:element name="detail" minOccurs="0">
<xsd:complexType>
<xsd:sequence>
<xsd:any/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
<xsd:attribute name="requestID" type="RequestID" use="optional"/>
<xsd:attribute name="type">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:enumeration value="notAttempted"/>
<xsd:enumeration value="couldNotConnect"/>
<xsd:enumeration value="connectionClosed"/>
<xsd:enumeration value="other"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
</xsd:complexType>
<!-- ***** Bind ***** -->
<xsd:complexType name="BindRequest">
<xsd:complexContent>
<xsd:extension base="DsmIMessage">
<xsd:attribute name="principal" type="DsmIDN" use="required"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- ***** Search ***** -->
<xsd:complexType name="AttributeDescriptions">
<xsd:sequence minOccurs="0" maxOccurs="unbounded">
<xsd:element name="attribute" type="AttributeDescription"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="SearchRequest">
<xsd:complexContent>
<xsd:extension base="DsmIMessage">
<xsd:sequence>
<xsd:element name="filter" type="Filter"/>
<xsd:element name="attributes" type="AttributeDescriptions" minOccurs="0"/>

```

```

</xsd:sequence>
<xsd:attribute name="dn" type="DsmIDN" use="required"/>
<xsd:attribute name="scope" use="required">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="baseObject"/>
      <xsd:enumeration value="singleLevel"/>
      <xsd:enumeration value="wholeSubtree"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="derefAliases" use="required">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="neverDerefAliases"/>
      <xsd:enumeration value="derefInSearching"/>
      <xsd:enumeration value="derefFindingBaseObj"/>
      <xsd:enumeration value="derefAlways"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="sizeLimit" type="MAXINT" use="optional" default="0"/>
<xsd:attribute name="timeLimit" type="MAXINT" use="optional" default="0"/>
<xsd:attribute name="typesOnly" type="xsd:boolean" use="optional" default="false"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- ***** Search Result Entry ***** -->
<xsd:complexType name="SearchResultEntry">
  <xsd:complexContent>
    <xsd:extension base="DsmIMessage">
      <xsd:sequence>
        <xsd:element name="attr" type="DsmIAttr" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="dn" type="DsmIDN" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="DsmIAttr">
  <xsd:sequence>
    <xsd:element name="value" type="DsmIValue" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="desc" type="OID" use="required"/>
</xsd:complexType>
<xsd:complexType name="DsmIModification">
  <xsd:sequence>
    <xsd:element name="value" type="DsmIValue" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="desc" type="OID" use="required"/>
  <xsd:attribute name="operation" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="add"/>
        <xsd:enumeration value="delete"/>
        <xsd:enumeration value="replace"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>
<!-- ***** Search Result Reference ***** -->
<xsd:complexType name="SearchResultReference">
  <xsd:complexContent>
    <xsd:extension base="DsmIMessage">
      <xsd:sequence>
        <xsd:element name="ref" type="xsd:anyURI" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- ***** MODIFY ***** -->
<xsd:complexType name="ModifyRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmIMessage">
      <xsd:sequence>
        <xsd:element name="attr" type="DsmIModification" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="dn" type="DsmIDN" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!-- ***** ADD ***** -->
<xsd:complexType name="AddRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmIMessage">
      <xsd:sequence>
        <xsd:element name="attr" type="DsmIAttr" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="dn" type="DsmIDN" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!-- ***** DELETE ***** -->
<xsd:complexType name="DelRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmIMessage">
      <xsd:attribute name="dn" type="DsmIDN" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!-- ***** MODIFY DN ***** -->
<xsd:complexType name="ModifyDNRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmIMessage">
      <xsd:attribute name="dn" type="DsmIDN" use="required"/>
      <xsd:attribute name="newrdn" type="DsmIRDN" use="required"/>
      <xsd:attribute name="deleteoldrdn" type="xsd:boolean" use="optional" default="true"/>
      <xsd:attribute name="newSuperior" type="DsmIDN" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!-- ***** COMPARE ***** -->
<xsd:complexType name="CompareRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmIMessage">
      <xsd:sequence>
        <xsd:element name="attr" type="AttributeValueAssertion"/>
      </xsd:sequence>
      <xsd:attribute name="dn" type="DsmIDN" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!-- ***** ABANDON ***** -->
<xsd:complexType name="AbandonRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmIMessage">
      <xsd:attribute name="abandonID" type="RequestID" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!-- ***** EXTENDED OPERATION ***** -->
<xsd:complexType name="ExtendedRequest">
  <xsd:complexContent>

```

```

<xsd:extension base="DsmIMessage">
  <xsd:sequence>
    <xsd:element name="requestName" type="ExtensionTypeID"/>
    <xsd:element name="requestValue" type="DsmIANY" minOccurs="0"/>
  </xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ExtendedResponse">
  <xsd:complexContent>
    <xsd:extension base="LDAPResult">
      <xsd:sequence>
        <xsd:element name="responseName" type="ExtensionTypeID" minOccurs="0"/>
        <xsd:element name="response" type="DsmIANY" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!-- *****END base SCHEMA ***** -->
</xsd:schema>

```

ⓄASIS statements?

```

Deleted: <xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema" ¶
xmlns="http://www.dsml.org/DSML/v2" ¶
elementFormDefault="qualified" ¶
targetNamespace="http://www.dsml.org/DSML/v2">¶
¶
<xsd:element
name="dsmlEnvelopeRequest"
type="DSMLEnvelopeRequest"
/>¶
<xsd:element
name="dsmlEnvelopeResponse"
type="DSMLEnvelopeResponse"
/>¶
¶
<!-- *****
DSMLv2.0 Envelopes
***** -->¶
¶
<!-- ***DSML Request*** ¶

```

**Deleted: ¶**

¶  
The presentation, distribution or other dissemination of the information contained herein by Microsoft is not a license, either expressly or impliedly, to any intellectual property owned or controlled by Microsoft. ¶

¶  
This document and the information contained herein is provided on an "AS IS" basis and MICROSOFT DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. ¶

¶  
IN NO EVENT WILL MICROSOFT BE LIABLE TO ANY OTHER PARTY FOR THE COST OF PROCURING SUBSTITUTE GOODS OR SERVICES, LOST PROFITS, LOSS OF USE, LOSS OF DATA, OR ANY INCIDENTAL, CONSEQUENTIAL, DIRECT, INDIRECT, OR SPECIAL DAMAGES WHETHER UNDER CONTRACT, TORT, WARRANTY, OR OTHERWISE, ARISING IN ANY MANNER RELATING TO THIS ¶ [4]

```

<!-- ***** DSML Response ***** -->
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/" SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body xmlns:dsml="http://www.dsml.org/dsml/v2">
    <dsml:dsmlEnvelopeResponse>
      <dsml:modifyResponse> ...</dsml:modifyResponse>
      <dsml:addResponse>...</dsml:modifyResponse>
      ...
    </dsml:dsmlEnvelopeResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

```

<!-- ***** SOAP Fault ***** -->
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/" SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>SOAP-ENV:Server</faultcode>
      <faultstring>Server Error</faultstring>
      <detail>
        Cannot connect to a DSML server
      </detail>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

## 6.1 SOAP Header

DSMLv2 does not mandate any SOAP header elements, such as routing, reliable messaging, signing, and encryption.

## 6.2 SOAP Body

The maximum number of DSML Envelopes that can be contained in a SOAP Body is one. In the SOAP request message a *dsmlEnvelopeRequest* is expected; in the SOAP response message a *dsmlEnvelopeResponse* is expected.

## 6.3 SOAP Fault

SOAP Fault should be used only when an error occurs outside the scope of DSML processing. For example, the SOAP Server is not able to find or connect to a DSML server to process a DSMLv2 document. On the other hand, if DSML errors happen

during DSML processing, then they should be carried as a DSML response document in the SOAP response message.

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="http://www.dsml.org/DSML/v2"
  elementFormDefault="qualified"
  targetNamespace="http://www.dsml.org/DSML/v2">

  <xsd:element name="dsmlEnvelopeRequest" type="DSMLEnvelopeRequest" />
  <xsd:element name="dsmlEnvelopeResponse" type="DSMLEnvelopeResponse" />

  <!-- ***** DSMLv2.0 Envelopes ***** -->

  <!-- ***DSML Request*** -->
  <xsd:complexType name="DSMLEnvelopeRequest">
    <xsd:sequence>
      <xsd:choice minOccurs="0" maxOccurs="unbounded" >
        <xsd:element name="searchRequest" type="SearchRequest" />
        <xsd:element name="modifyRequest" type="ModifyRequest" />
        <xsd:element name="addRequest" type="AddRequest" />
        <xsd:element name="delRequest" type="DelRequest" />
        <xsd:element name="modDNRequest" type="ModifyDNRequest" />
        <xsd:element name="compareRequest" type="CompareRequest" />
        <xsd:element name="extendedReq" type="ExtendedRequest" />
      </xsd:choice>
    </xsd:sequence>

    <xsd:attribute name="processing" use="optional" default="sequential">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="sequential" />
          <xsd:enumeration value="parallel" />
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:attribute>

    <xsd:attribute name="onError" use="optional" default="exit">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="resume" />
          <xsd:enumeration value="exit" />
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:attribute>
  </xsd:complexType>
```



```

<!-- **** DSML Response **** -->
<xsd:complexType name="DSMLEnvelopeResponse">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="unbounded" >
      <xsd:element name="searchResponse" type="SearchResponse"/>
      <xsd:element name="modifyResponse" type="LDAPResult" />
      <xsd:element name="addResponse" type="LDAPResult" />
      <xsd:element name="delResponse" type="LDAPResult" />
      <xsd:element name="modDNResponse" type="LDAPResult" />
      <xsd:element name="compareResponse" type="LDAPResult" />
      <xsd:element name="extendedResp" type="ExtendedResponse" />
      <xsd:element name="errorResponse" type="xsd:string" />
    </xsd:choice>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="Control">
  <xsd:sequence>
    <xsd:element name="controlType" minOccurs="1" maxOccurs="1" type="xsd:string"
  />
    <xsd:element name="criticality" minOccurs="0" default="false" type="xsd:boolean"
  />
    <xsd:element name="controlValue" minOccurs="0" maxOccurs="1" type="xsd:string"
  />
  </xsd:sequence>
</xsd:complexType>

<!-- ***** MAX Integer ***** -->
<xsd:simpleType name="MAXINT">
  <xsd:restriction base="xsd:unsignedInt">
    <xsd:maxInclusive value="2147483647" />
  </xsd:restriction>
</xsd:simpleType>

<!-- ***** LDAP RESULT ***** -->

<xsd:simpleType name="LDAPErrorCode">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="success" />
    <xsd:enumeration value="operationsError" />
    <xsd:enumeration value="protocolError" />
    <xsd:enumeration value="timeLimitExceeded" />
    <xsd:enumeration value="sizeLimitExceeded" />
  </xsd:restriction>
</xsd:simpleType>

```

```
<xsd:enumeration value="compareFalse" />
<xsd:enumeration value="compareTrue" />
<xsd:enumeration value="authMethodNotSupported" />
<xsd:enumeration value="strongAuthRequired" />
<xsd:enumeration value="referral" />
<xsd:enumeration value="adminLimitExceeded" />
<xsd:enumeration value="unavailableCriticalExtension" />
<xsd:enumeration value="confidentialityRequired" />
<xsd:enumeration value="saslBindInProgress" />
<xsd:enumeration value="noSuchAttribute" />
<xsd:enumeration value="undefinedAttributeType" />
<xsd:enumeration value="inappropriateMatching" />
<xsd:enumeration value="constraintViolation" />
<xsd:enumeration value="attributeOrValueExists" />
<xsd:enumeration value="invalidAttributeSyntax" />
<xsd:enumeration value="noSuchObject" />
<xsd:enumeration value="aliasProblem" />
<xsd:enumeration value="invalidDNyntax" />
<xsd:enumeration value="aliasDereferencingProblem" />
<xsd:enumeration value="inappropriateAuthentication" />
<xsd:enumeration value="invalidCredentials" />
<xsd:enumeration value="insufficientAccessRights" />
<xsd:enumeration value="busy" />
<xsd:enumeration value="unavailable" />
<xsd:enumeration value="unwillingToPerform" />
<xsd:enumeration value="loopDetect" />
<xsd:enumeration value="namingViolation" />
<xsd:enumeration value="objectClassViolation" />
<xsd:enumeration value="notAllowedOnNonLeaf" />
<xsd:enumeration value="notAllowedOnRDN" />
<xsd:enumeration value="entryAlreadyExists" />
<xsd:enumeration value="objectClassModsProhibited" />
<xsd:enumeration value="affectMultipleDSAs" />
<xsd:enumeration value="other" />
<xsd:enumeration value="notAttempted" />
<xsd:enumeration value="couldNotConnect" />
<xsd:enumeration value="connectionClosed" />
</xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="LDAPResult">
  <xsd:sequence>
    <xsd:element name="resultCode" minOccurs="1" maxOccurs="1">
      <xsd:complexType>
        <xsd:simpleContent>
          <xsd:extension base="xsd:int">
```

```

        <xsd:attribute name="descr" use="optional" type="LDAPErrorCode" />
    </xsd:extension>
</xsd:simpleContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="matchedDN" minOccurs="0" maxOccurs="1" default=""
type="xsd:string" />
<xsd:element name="errorMessage" minOccurs="0" maxOccurs="1" default=""
type="xsd:string" />
<xsd:element name="referral" minOccurs="0" maxOccurs="unbounded" default=""
type="xsd:string" />
</xsd:sequence>
</xsd:complexType>

```

```

<!-- ***** Search ***** -->

```

```

<xsd:complexType name="SearchRequest">
<xsd:sequence>
    <xsd:element name="scope" minOccurs="1" maxOccurs="1">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="baseObject" />
                <xsd:enumeration value="singleLevel" />
                <xsd:enumeration value="wholeSubtree" />
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element name="derefAliases" minOccurs="1" maxOccurs="1">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="neverDerefAliases" />
                <xsd:enumeration value="derefInSearching" />
                <xsd:enumeration value="derefFindingBaseObj" />
                <xsd:enumeration value="derefAlways" />
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element name="sizeLimit" minOccurs="0" maxOccurs="1" default="0"
type="MAXINT" />
    <xsd:element name="timeLimit" minOccurs="0" maxOccurs="1" default="0"
type="MAXINT" />
    <xsd:element name="typesOnly" minOccurs="0" maxOccurs="1" default="false"
type="xsd:boolean" />

```

```
<xsd:element name="filter" minOccurs="1" maxOccurs="1" type="xsd:string" />
<xsd:element name="attributes" minOccurs="0" maxOccurs="1" type="xsd:string" />
<xsd:element name="control" minOccurs="0" maxOccurs="unbounded"
type="Control"/>
</xsd:sequence>
<xsd:attribute name="dn" use="required" type="xsd:string" />
</xsd:complexType>
```

```
<xsd:complexType name="SearchResponse">
<xsd:sequence>
<xsd:element name="searchResEntry" minOccurs="0" maxOccurs="unbounded"
type="SearchResultEntry" />
<xsd:element name="searchResRef" minOccurs="0" maxOccurs="unbounded"
type="SearchResultRef"/>
<xsd:element name="searchResDone" minOccurs="1" maxOccurs="1"
type="LDAPResult" />
</xsd:sequence>
</xsd:complexType>
```

```
<xsd:complexType name="SearchResultEntry">
<xsd:sequence>
<xsd:element name="objectclass" minOccurs="0" maxOccurs="1"
type="ObjectClass" />
<xsd:element name="attr" minOccurs="0" maxOccurs="unbounded"
type="Dsm1Attr" />
<xsd:element name="control" minOccurs="0" maxOccurs="unbounded"
type="Control"/>
</xsd:sequence>
<xsd:attribute name="dn" use="required" type="xsd:string" />
</xsd:complexType>
```

```
<xsd:complexType name="Dsm1Attr">
<xsd:sequence>
<xsd:element name="value" minOccurs="0" maxOccurs="unbounded">
<xsd:complexType>
<xsd:simpleContent>
<xsd:extension base="xsd:string">
<xsd:attribute name="encoding" use="optional" >
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:enumeration value="base64" />
```

```

        </xsd:restriction>
        </xsd:simpleType>
        </xsd:attribute>
        </xsd:extension>
        </xsd:simpleContent>
        </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="name" use="required" type="xsd:string" />
    <xsd:attribute name="ref" use="optional" type="xsd:anyURI" />
</xsd:complexType>

```

```

<xsd:complexType name="DsmmlModifyAttr">
    <xsd:sequence>
        <xsd:element name="value" maxOccurs="unbounded">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute name="encoding" use="optional" >
                            <xsd:simpleType>
                                <xsd:restriction base="xsd:string">
                                    <xsd:enumeration value="base64" />
                                </xsd:restriction>
                            </xsd:simpleType>
                        </xsd:attribute>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="name" use="required" type="xsd:string" />
    <xsd:attribute name="operation" use="required" >
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="add" />
                <xsd:enumeration value="delete" />
                <xsd:enumeration value="replace" />
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:complexType>

```

```

<xsd:complexType name="ObjectClass">
    <xsd:sequence>
        <xsd:element name="oc-value" minOccurs="1" maxOccurs="unbounded" >

```

```
<xsd:complexType>
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
      <xsd:attribute name="ref" use="optional" type="xsd:anyURI" />
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
```

```
<xsd:complexType name="SearchResultRef">
  <xsd:sequence>
    <xsd:element name="ldapURL" minOccurs="1" maxOccurs="unbounded"
type="xsd:string" />
    <xsd:element name="control" minOccurs="0" maxOccurs="unbounded"
type="Control"/>
  </xsd:sequence>
</xsd:complexType>
```

```
<!-- ***** MODIFY ***** -->
```

```
<xsd:complexType name="ModifyRequest">
  <xsd:sequence>
    <xsd:element name="attr" minOccurs="1" maxOccurs="unbounded"
type="DsmlModifyAttr" />
    <xsd:element name="control" minOccurs="0" maxOccurs="unbounded"
type="Control"/>
  </xsd:sequence>
  <xsd:attribute name="dn" use="required" type="xsd:string" />
</xsd:complexType>
```

```
<!-- ***** ADD ***** -->
```

```
<xsd:complexType name="AddRequest">
  <xsd:sequence>
    <xsd:element name="objectclass" minOccurs="0" maxOccurs="1"
type="ObjectClass" />
    <xsd:element name="attr" minOccurs="0" maxOccurs="unbounded"
type="DsmlAttr" />
```

```
    <xsd:element name="control" minOccurs="0" maxOccurs="unbounded"
type="Control"/>
  </xsd:sequence>
  <xsd:attribute name="dn" use="required" type="xsd:string"/>
</xsd:complexType>
```

```
<!-- ***** DELETE ***** -->
<xsd:complexType name="DelRequest">
  <xsd:sequence>
    <xsd:element name="control" minOccurs="0" maxOccurs="unbounded"
type="Control"/>
  </xsd:sequence>
  <xsd:attribute name="dn" use="required" type="xsd:string"/>
</xsd:complexType>
```

```
<!-- ***** MODIFY DN ***** -->
<xsd:complexType name="ModifyDNRequest">
  <xsd:sequence>
    <xsd:element name="newrdn" minOccurs="1" maxOccurs="1" type="xsd:string" />
    <xsd:element name="deleteoldrdn" minOccurs="1" maxOccurs="1"
type="xsd:boolean" />
    <xsd:element name="newSuperior" minOccurs="0" maxOccurs="1"
type="xsd:string" />
    <xsd:element name="control" minOccurs="0" maxOccurs="unbounded"
type="Control"/>
  </xsd:sequence>
  <xsd:attribute name="dn" use="required" type="xsd:string"/>
</xsd:complexType>
```

```
<!-- ***** COMPARE ***** -->
<xsd:complexType name="CompareRequest">
  <xsd:sequence>
    <xsd:element name="attr" minOccurs="1" maxOccurs="1" type="DsmlAttr" />
    <xsd:element name="control" minOccurs="0" maxOccurs="unbounded"
type="Control"/>
  </xsd:sequence>
  <xsd:attribute name="dn" use="required" type="xsd:string"/>
</xsd:complexType>
```

```

<!-- ***** EXTENDED OPERATION ***** -->

<xsd:complexType name="ExtendedRequest">
  <xsd:sequence>
    <xsd:element name="requestName" minOccurs="1" maxOccurs="1"
type="xsd:string" />
    <xsd:element name="requestValue" minOccurs="0" maxOccurs="1"
type="xsd:string" />
    <xsd:element name="control" minOccurs="0" maxOccurs="unbounded"
type="Control"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ExtendedResponse">
  <xsd:sequence>
    <xsd:element name="resultCode" minOccurs="1" maxOccurs="1">
      <xsd:complexType>
        <xsd:simpleContent>
          <xsd:extension base="xsd:int">
            <xsd:attribute name="descr" use="optional" type="LDAPErrorCode" />
          </xsd:extension>
        </xsd:simpleContent>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="matchedDN" minOccurs="0" maxOccurs="1" default=""
type="xsd:string" />
    <xsd:element name="errorMessage" minOccurs="0" maxOccurs="1" default=""
type="xsd:string" />
    <xsd:element name="referral" minOccurs="0" maxOccurs="unbounded" default=""
type="xsd:string" />
    <xsd:element name="responseName" minOccurs="0" maxOccurs="1"
type="xsd:string" />
    <xsd:element name="response" minOccurs="0" maxOccurs="1" type="xsd:string" />
    <xsd:element name="control" minOccurs="0" maxOccurs="unbounded"
type="Control"/>
  </xsd:sequence>
  <xsd:attribute name="dn" use="optional" type="xsd:string" />
</xsd:complexType>

<!-- *****END OF SCHEMA ***** -->
</xsd:schema>

```



## DSMLv2.0 Envelope Request Example

```
<?xml version='1.0' ?>
```

```
<dsmlEnvelopeRequest xmlns="http://www.dsml.org/DSML/v2"  
processing="sequential" >
```

```
<!-- ***** MODIFY ***** -->
```

```
<modifyRequest dn="CN=Bob Rush,OU=Sales, DC=Example,DC=COM">  
  <attr name="telephoneNumber" operation="replace">  
    <value>536 354 2343</value>  
    <value>234 212 4534</value>  
  </attr>  
  <attr name="sn" operation="replace">  
    <value>Johnson</value>  
  </attr>  
  <attr name="directReport" operation="add">  
    <value>CN=John Smith, OU=Sales, DC=Example, DC=com</value>  
  </attr>  
</modifyRequest>
```

```
<modifyRequest dn="CN=Alice,OU=Development,DC=Example,DC=COM" >  
  <attr name="description" operation="replace">  
    <value>Group Assistant</value>  
  </attr>  
  <control>  
    <controlType>1.2.840.113556.1.4.841</controlType>  
    <criticality>>true</criticality>  
    <controlValue>TW9kaWZ5IFJlcXVlc3QgRXhhbXBsZQ==</controlValue>  
  </control>  
</modifyRequest>
```

```
<!-- ***** SEARCH ***** -->
```

```
<searchRequest dn="dc=example,dc=com">  
  <scope>wholeSubtree</scope>  
  <derefAliases>neverDerefAliases</derefAliases>  
  <sizeLimit>1</sizeLimit>  
  <timeLimit>1000</timeLimit>  
  <filter>(objectClass=user)</filter>  
  <attributes>sn,title,phoneNumber,description</attributes>  
</searchRequest>
```

```
<searchRequest dn="ou=Marketing,dc=example,dc=com">
  <scope>singleLevel</scope>
  <derefAliases>neverDerefAliases</derefAliases>
  <sizeLimit>1000</sizeLimit>
  <filter>(sn=john*)</filter>
  <control>
    <controlType>1.2.840.113556.1.4.843</controlType>
    <criticality>>true</criticality>

<controlValue>U2VhcmNoUmVxdWVzdCBFeGFtcGxIIENvbnRyb2wgTnVtYmVyIDE
=</controlValue>
  </control>
  <control>
    <controlType>1.2.840.113556.1.4.821</controlType>
    <criticality>true</criticality>

<controlValue>QW5vdGVvZXIgaU2VhcmNoUmVxdWVzdCBFeGFtcGxIIENvbnRyb2
w=</controlValue>
  </control>
</searchRequest>
```

```
<!-- ***** ADD ***** -->
```

```
<addRequest dn="CN=Alice,OU=HR,DC=Example, DC=COM">
  <objectclass>
    <oc-value>top</oc-value>
    <oc-value>person</oc-value>
    <oc-value>organizationalPerson</oc-value>
  </objectclass>
  <attr name="sn"><value>Johnson</value></attr>
  <attr name="givenName"><value>Alice</value></attr>
  <attr name="title"><value>Software Design Engineer</value></attr>
</addRequest>
```

```
<!-- ***** DELETE ***** -->
```

```
<delRequest dn="CN=Bob Salisbury, OU=HR,DC=Example,DC=COM" />
```

```
<!-- ***** MODIFY DN ***** -->
```

```
<modDNRequest dn="CN=Alice Johnson,DC=Example,DC=COM">
  <newrdn>Alice Weiss</newrdn>
  <deleteoldrdn>true</deleteoldrdn>
  <newSuperior>OU=Marketing,DC=Example,DC=COM</newSuperior>
```

</modDNRequest>

<!-- \*\*\*\*\* COMPARE \*\*\*\*\* -->

<compareRequest dn="CN=Johnson,OU=HR,DC=Example,DC=COM">  
 <attr name="sn"><value>Johnson</value></attr>  
</compareRequest>

<!-- \*\*\*\*\* EXTENDED OPERATION \*\*\*\*\* -->

<extendedReq>  
 <requestName>1.2.840.113556.1.4.417</requestName>  
  
<requestValue>QWN0aXZlIERpcmVjdG9yeSBpcyBzbyBjb29sISEh</requestValue>  
</extendedReq>

<!-- \*\*\*\* An example of an invalid request \*\*\*\*\* -->

<bogusRequest>  
 <firstname>Ryan</firstname>  
 <lastname>Rys</lastname>  
</bogusRequest>

</dsmlEnvelopeRequest>

## **DSMLv2.0 Envelope Response Example**

<?xml version='1.0' ?>

<dsmlEnvelopeResponse xmlns="http://www.dsml.org/DSML/v2">

<!-- \*\*\*\*\* MODIFY \*\*\*\*\* -->

<modifyResponse>  
 <resultCode descr="timeLimitExceeded">3</resultCode>  
 <errorMessage>Time Limit Exceeded:DSC20AC932</errorMessage>  
</modifyResponse>

<modifyResponse>  
 <resultCode descr="unwillingToPerform">53</resultCode>

```
<errorMessage>System Attribute may not be modified</errorMessage>
</modifyResponse>
```

```
<!-- ***** SEARCH ***** -->
```

```
<searchResponse>
  <searchResEntry dn="CN=Melissa Kennedy, OU=Development,OU=Platform,
DC=Example,DC=COM">
    <objectclass>
      <oc-value>top</oc-value>
      <oc-value>person</oc-value>
      <oc-value>organizationalPerson</oc-value>
      <oc-value>user</oc-value>
    </objectclass>
    <attr name="sn"><value>Kennedy</value></attr>
    <attr name="title"><value>Technical Writer</value></attr>
    <attr name="phoneNumber"><value>(425)999-8888</value></attr>
  </searchResEntry>
  <searchResEntry dn="CN=David Johnson,OU=HR,DC=Example,DC=COM">
    <objectclass>
      <oc-value>top</oc-value>
      <oc-value>person</oc-value>
      <oc-value>organizationalPerson</oc-value>
      <oc-value>user</oc-value>
    </objectclass>
    <attr name="sn"><value>Johnson</value></attr>
    <attr name="title"><value>Program Manager</value></attr>
    <attr name="phoneNumber"><value>(425)777-5555</value></attr>
  </searchResEntry>
  <searchResEntry dn="CN=John Smith, OU=Finance,DC=Example,DC=COM">
    <objectclass>
      <oc-value>top</oc-value>
      <oc-value>person</oc-value>
      <oc-value>organizationalPerson</oc-value>
      <oc-value>user</oc-value>
    </objectclass>
    <attr name="sn"><value>Smith</value></attr>
    <attr name="title"><value>Software Design Engineer</value></attr>
    <attr name="phoneNumber"><value>(425)666-3333</value></attr>
  </searchResEntry>
  <searchResRef>
    <ldapURL>ldap://srv05.fabrikam.com/DC=Fabrikam,DC=COM</ldapURL>
  </searchResRef>
  <searchResDone>
    <resultCode descr="success">0</resultCode>
```

```
</searchResDone>
</searchResponse>

<searchResponse>
  <searchResEntry dn="CN=Johnson Weber, OU=Marketing,
DC=Example,DC=COM">
    <attr name="sn"><value>Weber</value></attr>
    <attr name="givenName"><value>Johnson</value></attr>
    <attr name="title"><value>Vice President</value></attr>
    <attr name="phoneNumber"><value>(425)111-2222</value></attr>
    <attr name="memberOf">
      <value>CN=Administrators,OU=IT,DC=Example,DC=COM</value>
      <value>CN=Sales, OU=Groups, DC=Example,DC=COM</value>
    </attr>
  </searchResEntry>
  <searchResEntry dn="CN=John Kovack, OU=Marketing, DC=Example,DC=COM">
    <attr name="sn"><value>Kovack</value></attr>
    <attr name="givenName"><value>John</value></attr>
    <attr name="manager">
      <value>CN=Alice Gray,OU=Marketing,DC=Example,DC=COM</value>
    </attr>
    <attr name="title"><value>Group Manager</value></attr>
  </searchResEntry>
</searchResDone>
</searchResponse>
```

```
<!-- ***** ADD ***** -->
```

```
<addResponse>
  <resultCode>0</resultCode>
</addResponse>
```

```
<!-- ***** DELETE ***** -->
```

```
<delResponse>
  <resultCode descr="timeLimitExceeded">3</resultCode>
  <errorMessage>DSDEL::230234</errorMessage>
</delResponse>
```

```
<!-- ***** MODIFY DN ***** -->
```

```
<modDNResponse>
  <resultCode>0</resultCode>
</modDNResponse>
```

```
<!-- ***** COMPARE ***** -->
```

```
<compareResponse>
  <resultCode>0</resultCode>
</compareResponse>
```

```
<!-- ***** EXTENDED OPERATION ***** -->
```

```
<extendedResp>
  <resultCode descr="success">0</resultCode>
  <responseName>1.2.840.113556.1.4.417</responseName>
```

```
<response>RXh0ZW5kZWRSZXNwb25zZSBDb250cm9sIEV4YW1wbGU=  
</response>  
</extendedResp>
```

```
</dsmlEnvelopeResponse>
```

The presentation, distribution or other dissemination of the information contained herein by Microsoft is not a license, either expressly or impliedly, to any intellectual property owned or controlled by Microsoft.

This document and the information contained herein is provided on an "AS IS" basis and MICROSOFT DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IN NO EVENT WILL MICROSOFT BE LIABLE TO ANY OTHER PARTY FOR THE COST OF PROCURING SUBSTITUTE GOODS OR SERVICES, LOST PROFITS, LOSS OF USE, LOSS OF DATA, OR ANY INCIDENTAL, CONSEQUENTIAL, DIRECT, INDIRECT, OR SPECIAL DAMAGES WHETHER UNDER CONTRACT, TORT, WARRANTY, OR OTHERWISE, ARISING IN ANY MANNER RELATING TO THIS DOCUMENT, WHETHER OR NOT SUCH PARTY HAD ADVANCE NOTICE OF THE POSSIBILITY OF SUCH DAMAGES.