DRAFT

Comparison of Present Directory Services XML Definitions

DAML, DirXML, iXDGW

Submitted: April 6 2001

Bill Curtin
DISA

1 Introduction

DSML version 1 is a XML definition of LDAP schema for use with Directory Service products. Version 2 of DSML will extend the DSML specification to include directory operations as well as schema.

The intent of this document is to provide a comparison of XML already defined for directory service operations to help guide the DSML working group in designing version 2 DSML. This comparison includes Novell's DIRXML, iPlanet's eXtensible Template Language (XTL), and Access360's Directory Access Markup Language (DAML).

Note that this comparison of uses the definition of the XTL defined in Chapter 5 of the user guide (which included attributes associated with the operation elements). This definition of XTL was different than the DTD in Annex B of that document (which did not contain an ATTLIST for the operation elements).

1.1 General Observation

The three XML definitions compared in this document were each designed for different purposes.

The DAML appears to be designed as a direct access to a LDAP server. The XML mimics very closely the LDAP PDU structure defined in RFC 2251.

Novell's DirXML is designed primarily to support meta-directory functions to synchronize Novell's NDS with external applications and directories. As such elements and attributes have been defined to support functions necessary to perform meta-directory functions (such as schema mapping, associating NDS and external entries, etc). Novell's XML definition is referred to as XDS.

iPlanet's XTL is designed to support the iPlanet XMLDAP Directory Gateway (aka iXDGW). This is a middleware / Directory Portal product to interface with LDAP enabled directory services. XTL is used in conjunction with JAVA Servlets to perform directory operations.

iXDGW uses the concept of "spaces" organized into global, session, or local "levels" (depending on lifetime and visibility). This space is used for temporary storage, variable definitions, etc. Additionally, the XML is divided into three logical sections: a general tags used generally within a XML document, servlet tags used in the servlet – HTTP interface, and LDAP tags which define the interface to LDAPv3 directories.

Probably the first thing that needs to be done prior to defining DSML version 2 is to bound the scope of the standard. Is the working group defining XML to support one or more of the purposes associated with the above XML definitions? If the scope of DSML version 2 is defined, a more thorough examination of the three XML definitions will be possible.

1.2 Caveats

- This comparison was performed via a paper study of documentation associated with a product or the DTD of an XML definition. Findings are based on the authors understanding of this documentation. Author notes / questions are included in red text.
- The comparison below includes the XML elements and attributes associated with each operation. It does not, however, duplicate the detail defined in each XML DTD.
- Although commas are used below to separate elements and attributes, this was done for readability and no ordering should be implied.
- Less detail was given on the functionality of DAML elements and attributes as this is fully detailed in the LDAP v3 standards.

2 Operations

2.1 Bind / Connect to Server

DAML: *BindRequest (SimpleAuthentication | SASLCredentials) ATTLIST Name, Version* – used to bind to a LDAP server. The Name attribute contains the distinguished name (as defined in DSML version 1 schema) used to authenticate to the LDAP server, and the Version attribute defines the LDAP version and is set to version 3. SimpleAuthentication allows for a password attribute. SASLCredentials allows for method and credentials attributes.

DirXML: *authentication-info (server?, user?, password?)* - used to connect and log into an application. Child elements are all defined as #PCDATA. The server element specifies the server hosting the external application. User element specifies the username used to log into the external application. The password element specifies the password used for logging into the external password.

iXDGW: Does not have a specific element for connecting and authenticating to a server. Each element associated with an operation contains attributes to identify the directory server host and port, and the DN of the user and password to bind as. A session is created with the *servlet:session* element.

Comments: The DAML allows for methods of authentication contained in RFC 2251. Both DirXML and iXDGW support the simple authentication method of LDAP authentication however they do not support SASL based authentication.

DirXML and iXDGW do support XML to define connection to a LDAP server. The IXDGW contains a separate attribute for the LDAP port, although DirXML could support the port number with standard addressing (e.g. foobar.org:389). DAML does not define XML for TCP/IP connection.

2.2 Bind / Connect Response from Server

DAML: *BindResponse (LDAPResult, ServerSASLCredentials?)* Similar to RFC 2251, a bind response is returned for a bind request *LDAPResult* is either a referral element or an enumerated attribute list that mimics the result status defined in the RFC. Note that the LDAPResult does not address the reserved or unused parts of the RFC list. Additionally, only a subset of the *LDAPResult* attributes is valid as a response.

DirXML: *status (ANY) ATTLIST level, event-id* – is not used specifically as a response mechanism for the connection, but is used for other operations or to give the status of a DirXML driver. The level attribute is required and can be one of 5 possible states ranging from fatal error requiring shutting down the driver, to success. In the event of a warning of error level, the *event-id* contains the id of the connection command and the *status* element will contain explanatory text.

iXDGW: As noted earlier there is no specific XML element to connect and authenticate. Each operation includes this capability. Operations also contain a *response-id* attribute to define specific "space" where responses are conveyed. The response is returned as part of a *servlet:response* element and can be success or failure.

Comments: Each definition has the capability of returning a status responses to a bind / connection request. The DirXML and iXDGW implementations also seems to include specific attributes to associate the session / command.

2.3 Unbind / Disconnect from Server

DAML: *UnbindRequest (EMPTY)* – used as defined in RFC 2251

DirXML: No specific XML for ending the session.

IXDGW: No specific XML for ending the session, however, implementation specific setting for session timeout.

Comments: Depending on the usage of DSML, an unbind type operation may or may not be needed. If used as direct access to a LDAP server there may be a need to support an unbind element. If on the other hand it is being used via an application acting as an interface between the client and LDAP server the disconnect or unbind could be inherent as part of the interface.

2.4 Add Entry

DAML: *AddRequest (entry)*– used to define an add entry as defined by RFC 2251. The *entry* child element is from DSML version 1. This includes a DN attribute (which is required), and object class and attribute name and values.

DirXML: *add (association?, add-attr*, password?) ATTLIST src-dn, src-entry-id, dest-dn, dest-entry-id, class-name, template-dn, event-id* –used to create a new entry. The *association* child element is optional and links an NDS entry to an entry in

an external application (useful when acting as a meta-directory synchronization agent). The *add-attr* element conveys the attribute name and values to be added to the entry. The *password* child element may contain an authentication password for the added object (editors note: it is unclear to me if this password would be different than one included in a password attribute defined by the *add-attr* element). The class-name is a required attribute that defines the base object class for the entry. The *src-** and *dest-** attributes are used to link the entry in the NDS and external application (remember that DirXML is primarily used for meta-directory purposes so the source distinguished name and destination distinguished name may be different). The *template-dn* is used to identify a template to be used when creating the entry in the receivers name space. The *entry-id* is used to identify this add event.

IXDGW: *ldap:add (ldap:attr) ATTLIST host, port, user, password, dn, on-success, on-failure, response-id, id, attrs* (Note: virtually every element defined within XTL may also be child elements, however, ldap:attr is the more likely LDAP child element) – used to add an entry to the directory. The *ldap:attr* element associates a set of attribute name and values to be added when creating the entry. As noted earlier, the *host, port, user,* and *password* attributes are used to authenticate to the LDAP server. The *dn* attribute contains the distinguished name of the new entry. The *on-success, on-failure, and response-id* are used to convey a response for the operation and to define a template to execute based on the operation's success or failure. The id and attrs is an alternative means of defining attribute and value(s) sets to be added at entry creation.

Comments: Common to each of these XML definitions is the ability to define a Distinguished Name of the new entry and a list of attribute name and values to be included when creating the new entry. Both DAML and DirXML also define the base object class of the entry (not sure how iXDGW determines the object class of the new entry). Depending on whether a session bind is maintained, the host, port, user, and password may be needed by any operation similar to the iXDGW implementation. Other elements and attributes seem to be environment and implementation specific.

2.5 Add Response

DAML: *AddResponse (LDAPResult)*– Similar to RFC 2251, a add response is returned for a add request. *LDAPResult* is either a referral element or an enumerated attribute list that mimics the result status defined in the RFC.

DirXML: *add-association (#PCDATA) ATTLIST dest-dn, dest-entry-id, event-id* – the add-association element is used to return a unique key of the entry added as a result of the add command. The *dest-dn* attribute is required, however, because DirXML is primarily a meta-directory it should reflect the *src-dn* used in the *add* element in the directory from which the synchronization took place (this might be different than the DN actually added). Likewise the *dest-entry-id* attribute is the *src-entry-id* from the original add command. The event-id is the same as the add command.

The *status* element (see 2.2) is used to return the status of the add operation.

IXDGW: There is no new XML defined to return the response from an add operation. The *on-success, on-failure* attributes cause post processing after the operation. The response-id defines the "space" where response information is stored for use by the templates associated with these attributes.

Comments: There are no common means of returning a response from the add operation. DAML returns a LDAPResult as defined by RFC2251. DirXML's use of the status element is a similar solution but includes a unique id associated with the original add operation. It also returns an add association that is tied to the meta-directory functionality. iXDGW uses a solution which is very much tied to their gateway implementation where by the response is returned into their notion of "space" and templates are run depending on success or failure.

2.6 Delete an Entry

DAML: *DeleteRequest (EMPTY) ATTLIST DN* – Similar to RFC 2251, a delete request is used to remove an entry from a LDAP server. The DN attribute is required and used to define the distinguished name of the entry to be removed.

DirXML: *delete (association?) ATTLIST src-dn, src-entry-id, dest-dn, dest-entry-id, class-name, event-id* – can be used to remove an entry. The *association* child element is optional and links an NDS entry to an entry in an external application. The *src-** and *dest-** attributes are used to link the entry in the NDS and external application. The class-name attribute defines the base object class of the entry (not sure why this would be used). The *entry-id* is used to identify this delete event.

IXDGW: *ldap:del (*) ATTLIST host, port, user, password, dn, on-success, on-failure, response-id* (Note: virtually every element defined within XTL may also be child elements according to the DTD, however, another part of the iXDGW document states that *ldap:del* has no contents) – used to remove an entry to the directory. As noted earlier, the *host, port, user,* and *password* attributes are used to authenticate to the LDAP server. The *dn* attribute contains the distinguished name of the entry to be removed. The *on-success, on-failure, and response-id* are used to convey a response for the operation and to define a template to execute based on the operation's success or failure.

Comments: Common to each of these XML definitions is the use of a Distinguished Name to identify the entry to be removed. Depending on whether a session bind is maintained, the host, port, user, and password may be needed by any operation similar to the iXDGW implementation. Other elements and attributes seem to be environment and implementation specific.

2.7 Delete Response

DAML: *DeleteResponse (LDAPResult)*– Similar to RFC 2251, a delete response is returned for a delete request. *LDAPResult* is either a referral element or an enumerated attribute list that mimics the result status defined in the RFC.

DirXML: *remove-association (#PCDATA) ATTLIST event-id* – the *remove-association* element is used to remove a unique key associated with the deleted entry. This would be transformed from the external application's delete command in the case where the synchronized entry is to be maintained in NDS but only the association with the external deleted entry is to be removed. The *event-id* identifies the *remove-association* command.

The *status* element (see 2.2) is used to return the status of the delete or remove-association operation.

IXDGW: There is no new XML defined to return the response from a delete operation. The *on-success, on-failure* attributes cause post processing after the operation. The response-id defines the "space" where response information is stored for use by the templates associated with these attributes.

Comments: There are no common means of returning a response from the delete operation. DAML returns a LDAPResult as defined by RFC2251. DirXML's use of the status element is a similar solution but includes a unique id associated with the operation. It may also return remove-association that is tied to the meta-directory functionality. iXDGW uses a solution which is tied to their gateway implementation where by the response is returned into their notion of "space" and templates run depending on success or failure.

2.8 Modify an Entry

DAML: *ModifyRequest (Modification*) ATTLIST DN* – Similar to RFC 2251, a modification request is used to add, delete, or replace attribute – values from an entry. The modification element associates an attribute list with an operation (add, delete, replace). The DN attribute is required and used to define the distinguished name of the entry to be modified.

DirXML: *modify (association?, modify-attr+) ATTLIST src-dn, src-entry-id, dest-dn, dest-entry-id, class-name, event-id* –used to modify an entry's attributes. The *association* child element links an NDS entry to an entry in an external application. The *modify-attr* child element specifies the attribute – values associated with the modify operation, defines value syntax, and allows add-value, remove-value, and remove-all-values (and presumably the entry attribute). The *src-** and *dest-** attributes are used to link the entry in the NDS and external application. The class-name attribute defines the base object class of the entry. The *entry-id* is used to identify this event.

IXDGW: *ldap:modify (ldap:mod)* ATTLIST host, port, user, password, dn, on-success, on-failure, response-id*, id, mods (Note: virtually every element defined within

XTL may also be child elements, however, ldap:mod is the more likely LDAP child element) – used to modify an entry's attribute - values. The ldap:mod child element defines an operation (add, delete, or replace), syntax (text or binary) the attribute to be modified and a modification list). As noted earlier, the *host, port, user,* and *password* attributes are used to authenticate to the LDAP server. The *dn* attribute contains the distinguished name of the entry to be modified. The *on-success, on-failure, and response-id* are used to convey a response for the operation and to define a template to execute based on the operation's success or failure. The *id* and *mods* attributes are alternative means from *ldap:mod* of defining attribute and value(s) sets to modify.

Comments: Common to each of these XML definitions is the ability to define a Distinguished Name of the entry and a list of attribute name and values to modify. Both DAML and iXDGW supports add, replace, and delete operation. DirXML supports add, remove value (presumably equivalent to replace), and remove all values operations (presumably remove all values is equivalent to delete).

2.9 Modify Response

DAML: *ModifyResponse (LDAPResult)*– Similar to RFC 2251, a modify response is returned for a modify request. *LDAPResult* is either a referral element or an enumerated attribute list that mimics the result status defined in the RFC.

DirXML: The *status* element (see 2.2) is used to return the status of the modify operation.

IXDGW: There is no new XML defined to return the response from a modify operation. The *on-success, on-failure* attributes cause post processing after the operation. The response-id defines the "space" where response information is stored for use by the templates associated with these attributes.

Comments: There are no common means of returning a response from the modify operation. DAML returns a LDAPResult as defined by RFC2251. DirXML's use of the status element is a similar solution but includes a unique id associated with the operation. iXDGW uses a solution which is tied to their gateway implementation where by the response is returned into their notion of "space" and templates run depending on success or failure.

2.10 Modify Distinguished Name

DAML: *ModifyDNRequest (EMPTY)* ATTLIST *DN, NewRDN, NewSuperior, DeleteOld* – Similar to RFC 2251, a modification distinguished name request is used to change an entry's DN. The DN attribute reflects the current DN of the entry. The NewRDN is the new relative distinguished name of the entry. The NewSuperior defines the distinguished name of a new superior entry (used to move a leaf or subtree to a new superior). The DeleteOld attribute determines if the old RDN attribute is retained in the entry.

DirXML: *rename (association?, new-name)* ATTLIST *src-dn, src-entry-id, dest-dn, dest-entry-id, old-src-dn, remove-old-name, class-name, event-id* –used to modify an entry's relative distinguished name. The *association* child element links an NDS entry to an entry in an external application. The *new-name* child element specifies the new RDN for the entry. The *src-** and *dest-** attributes are used to link the entry in the NDS and external application. The *old-src-dn* specifies the current DN of the entry (before the change). The *remove-old-name* attribute specifies whether the old RDN is retained or deleted. The The *class-name* attribute defines the base object class of the entry. The *entry-id* is used to identify this event.

move (association?, parent) ATTLIST *src-dn, src-entry-id, dest-dn, dest-entry-id, old-src-dn, class-name, event-id* – used to move an entry to a new "container" (superior). The *association* child element links an NDS entry to an entry in an external application. The *parent* child element specifies the new "parent container" (superior) for the entry. The *src-** and *dest-** attributes are used to link the entry in the NDS and external application. The *old-src-dn* specifies the current DN of the entry (before the change). The The *class-name* attribute defines the base object class of the entry. The *entry-id* is used to identify this event.

IXDGW: There is no XML defined to modify the Distinguished Name.

Comments: DAML ModifyDN operation is defined according to RFC 2251. DirXML defines two separate elements to perform the two major functions of an LDAP ModifyDN operation. Additionally, DAML performs the operation by using attributes, while DirXML has defined new elements to express the new RDN and parent (superior). At the core of both approaches however is the ability to identify the entry using its present distinguished name, and define new RDN, new superior, and keep the old RDN as part of the entry.

2.11 ModifyDN Response

DAML: *ModifyDNResponse (LDAPResult)*– Similar to RFC 2251, a modify distinguished name response is returned for a modify DN request. *LDAPResult* is either a referral element or an enumerated attribute list that mimics the result status defined in the RFC.

DirXML: The *status* element (see 2.2) is used to return the status of the move and rename operation.

IXDGW: N/A

Comments: There are no common means of returning a response from the modifyDN operation. DAML returns a LDAPResult as defined by RFC2251. DirXML's use of the status element is a similar solution but includes a unique id associated with the operation.

2.12 Search

DAML: *SearchRequest (filter, AttributeName*)* ATTLIST *BaseObject, Scope, DerefAlias,SizeLimit, TimeLimit, TypesOnly* – used to return entries based on search criteria – similar to the search operation defined in RFC 2251. The *filter* child element is used to define the conditions by which an entry is matched. The *AttributeName* child element contains an attribute list that is returned from entries matching the search criteria. The *BaseObject* attribute defines the base entry to start the search. The *Scope* attribute defines the area to seach (base object only, one level below the base, or whole subtree). The *DerefAlias* attribute defines whether alias associated with the base object or matching objects are dereferenced. The *SizeLimit* and *TimeLimit* attributes control the number of entries to return and a timeout period for the search. The *TypesOnly* attribute defines whether the attribute type and value are returned or just the attribute types.

DirXML: *query (association?, (search-class | search-attr | read-att | read-parent)*) ATTLIST dest-dn, dest-entry-id, class-name, scope, event-id* – used to search for and return entries. The *association* child element is optional and links an NDS entry to an entry in an external application. The *search-class child* element specifies the filter of object classes to search for, if omitted all entries meeting the *search-attr* and *scope* criteria are returned. The *search-attr* child element specifies the filter for attributes matching. If more than one *search-attr* element is defined then the entry must match all attributes to be returned (similar to a Boolean "and"). Also, allows syntax to be defined for the search filter attribute value. The *read-att* child element specifies which attributes (and values) are returned with entries matching the query. Allows attributes and values to be returned as a default type or as XML (DSML version 1????). If absent all attributes are returned. If empty (<read-att/>) no attributes are returned. The *read-parent* child element specifies that the parent entry of entries matching the search filter criteria is returned. The *dest-dn* attribute defines the base entry's DN to start the search. If absent, the search starts from the root of the directory. The dest-dn-id is the unique id for the entry (used internally by DirXML). The class-name defines the base object class of the dest-dn (not sure why this is necessary). The scope attribute specifies the area of the search – entry, subordinates, or subtree (similar to base, onelevel, and subtree in RFC 2251). If absent, defaults to subtree. The *entry-id* is used to identify this search event.

IXDGW: *ldap:search (ANY) ATTLIST host, port, user, password, on-success, on-failure, response-id, list-id, space-id, scope, base, filter, attrs, pagesz, entries, resume, level* – used to search for and return entries. The *host, port, user,* and *password* attributes are used to authenticate to the LDAP server. The *on-success, on-failure, and response-id* are used to convey a response for the operation and to define a template to execute based on the operation's success or failure. The list-id attribute is a variable name to store / retrieve the search result listing. The *space-id* attribute defines the "space" name to store results. The *scope* attribute is base, onelevel, or subtree (default of subtree).

The base attribute contains the distinguished name of the base entry from which to start the search. The *filter* attribute is used to define the conditions by which an entry is matched. The *attrs* attribute is the list of attributes to return with matching entries (defaults to all). The *pagesz* attributes is used to return paged results. The *entry* attribute limits the number of entries returned. The resume attribute is used with paged results or to access persistent queries in a "space". The level attribute defines the "level" of the paged results "space".

Comments: Although defined differently (elements or attributes) between definitions, there are schema concepts that are common to each of these XML definitions. They are the ability to define a base entry distinguished name, a filter (DirXML includes a search-class which presumably could be mimicked by the other two XML definitions using a "objectclass=foobar" in the filter list), a list of attribute names and values to be returned, and scope (DirXML uses diffenent scope names that seem to map to base, onelevel, and subtree). Both DAML and iXDGW can control the sizelimit of the query. DAML supports creating full Boolean filters, while DirXML can only support "and" filters (not sure if iXDGW can support Boolean filters). Other than DAML which defines additional functionality which was defined in RFC 2251, other schema is specific to the environment in which the XML definition is defined.