Identify and Describe eDirectory Components

Before you dive into the deep end of the pool, you need to understand a term that governs eDirectory: schema. This is much like understanding the philosophy of an organization. What is the underlying principle that governs how a company is organized and functions? In eDirectory, this is the schema.

The eDirectory schema is the set of rules for object creation and placement. The schema dictates what properties must be assigned values when an object is created and what properties are optional. The schema is composed of two sets of definitions: object class, which defines which objects can be created and what attributes or properties are associated with each object type; and attribute definition, which defines the structure, syntax, and limitations of an attribute. The schema in NetWare 6.5 is extensible, which means that you can introduce new objects and properties into the schema to accomodate other network services and resources. Examples of this include the installation of ZENworks or GroupWise, which each extend the schema and introduce a host of new objects to the base schema that ships with NetWare 6.5. You can extend the schema in NetWare 6.5 using either Schema Manager, found in ConsoleOne under the Tools menu, or iManager. Novell prefers iManager in this version of NetWare.

The three major components of eDirectory are objects, properties, and values. The following list provides definitions for each type of component:

- An *object* is a fundamental component of eDirectory that contains information about a network resource. If you compare a traditional database or spreadsheet to eDirectory, a record in a database or a row in a spreadsheet is comparable to an object in eDirectory.
- An object has properties. *Properties* are fundamental components of eDirectory. A property is an attribute of an object. If you compare a traditional database or spreadsheet to eDirectory, an object's property is a field in a database or a column in a spreadsheet. Some properties are required, depending on the object, whereas others are optional. Some properties are single-valued, whereas others are multivalued. For example, to create a user object, depending on the utility that you use, you must give the user object a username and a last name. In ConsoleOne, you highlight the object's context to accomplish this, whereas in iManager, you also must provide a context for the user object. All other properties are optional. After you create the user object, you can give

the user a description, location, phone number, fax number, and password. The phone number and fax number are multivalued properties to accommodate the many numbers that a user can have. Password, description, and location are single-valued properties. A user can have only one. In the same way, two user objects have the same properties, which are different from the properties that a printer uses. The printer object has a different set of properties that the schema governs. Even though two users have the same properties, the values that are assigned to those properties are different in many cases.

- The data that you assign to an object's property is a *value*. If you compare a spreadsheet to eDirectory, a property value is similar to a cell in a spreadsheet.

Know the difference between a single-valued property and a multivalued property, in addition to those properties that are required and those that are optional. If a property is optional, you can create an object without defining a value for the property. If a property is required, you must define a value for it.

In Table 5.1, you see how a user object with a UserName of Warren, LastName of Wyrostek, and Description of Trainer correlate to records, fields and cells.

Table 5.1 Objects, Properties and Values

Object	UserName (Property 1)	LastName (Property 2)	Description (Property 3)
User	Warren (Value 1)	Wyrostek (Value 2)	Trainer (Value 3)
Records	**Field 1**	**Field 2**	**Field 3**
Record 1	Value 1	Value 2	Value 3

eDirectory organization is made up of three 3s. The first set of threes is the three major components of eDirectory: objects, properties, and values. You need to learn the second and third sets of three.

- The three classes of objects
- The three major types of container objects

Identify and Describe eDirectory Object Classes

The three classes of eDirectory objects include the following:

- Tree object
- Container objects
- Leaf objects

The Tree object is the top of the eDirectory tree. In some management utilities, the Tree object is called the [Root] of the tree.

A Container object is a holder of other eDirectory objects. In a sense, a Container logically groups eDirectory objects. Examples of Container objects are the Country object, the Organization object, the Organizational Unit object, the Domain object, the Security container, Role Based Services, and the License container. Some Novell documents also classify the Tree object as a container object. For this exam, consider the Tree object as both a single entity at the top of the tree and as a Container object.

Leaf objects are one of the three major types of eDirectory objects. Network resources are represented in the eDirectory tree by Leaf objects. Examples of Leaf objects are the User object, the Group object, the Server object, the Volume object, the Printer object, and the Organizational Role object.

Figure 5.1 shows examples of each of these object types.

Be familiar with the icons that represent each type of Container and Leaf object.

Characteristics of the Tree object include the following:

- Only one Tree object can exist in an eDirectory tree.
- The Tree object is a mandatory object in an eDirectory tree. You cannot have an eDirectory tree without a Tree object.
- The Tree object is created when the first NetWare server is installed into the tree.
- You cannot move, delete, or rename the Tree object. The name value that you give to an eDirectory tree can be changed, but you cannot change the name of the Tree object.

- The Tree object can contain Country and Organization objects or Alias objects to the Country or Organization. The Tree object might also have a Security container.

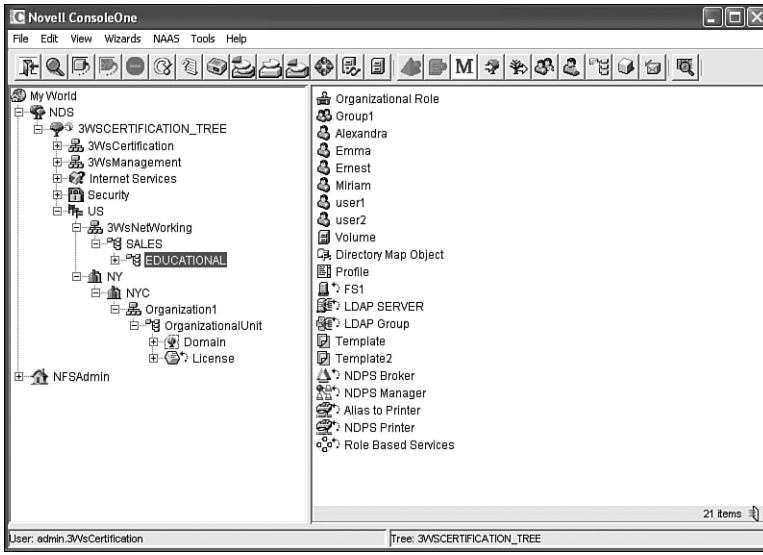

Figure 5.1 eDirectory objects.

When it comes to Container objects, even though NetWare 6.5 has more than three, Novell defines three major types of containers: the Country object, the Organization object, and the Organizational Unit. Following are characteristics of the Country object:

- A Country object is an optional object. It is generally used for global enterprises, in which the top level of the tree is divided by country.
- If you are working in a multiplatform directory services environment that depends on the X.500 standard, you need the Country object. This object is not required on a strictly NetWare-oriented eDirectory network.
- A Country object is represented by a two-character country code, such as DE, US, or UK.
- The Country object can exist only in the Tree object.
- The Country object can contain the Organization object or an Alias object to an Organization. Depending on the version of eDirectory that you are using, you can create other objects under the Country object, but those are not relevant for the exam.

Following are characteristics of the Organization object:

- An eDirectory requires at least one Organization object. In the past, Novell recommended that the eDirectory tree have a single Organization object. With the newer versions of eDirectory and NetWare 6.5, that is no longer the case. Novell now suggests that an eDirectory tree can have multiple Organization objects where appropriate.
- You use the Organization object to organize the tree into major groups, such as companies, universities, or divisions.
- The Organization object can be contained in the Tree object or the Country object.
- An Organization object cannot be contained in another Organization object.
- An Organization object can contain Organizational Unit objects and Leaf objects.

Following are characteristics of the Organizational Unit object:

- An Organizational Unit object is an optional object in eDirectory. Although it is not required, Novell recommends the use of Organizational Units for partitioning the tree.
- An Organizational Unit object further subdivides a tree under the Organization object into workgroups, departments, teams, and in some designs, geographical locations.
- An Organizational Unit object can be contained in an Organization object or an Organizational Unit object. It cannot be contained in the Country object or the Tree object.
- An Organizational Unit object can contain other Organizational Unit objects and all Leaf objects.

Following are the four special types of Container objects:

- License Container objects hold license certificate objects and are created when you install a license that is compatible with Novell Licensing Services (NLS).
- DNS components are represented by the Domain object, which can exist in an Organization object, Organizational Unit object, Country object, or Locality object.

- Security Container objects contain the security policy resources that are applicable to tree resources.
- Role-Based Service objects contain the roles and tasks that are assigned to users.

Leaf objects represent the available network resources. Leaf objects include the following:

- Alias objects, which are pointers to other objects, generally in a different container. The icon for an Alias object has a dotted quarter circle with a pointer on its right side.
- Application objects, which are network-distributable applications. This type of object is available when ZENworks is installed.
- Directory Map objects, which are eDirectory objects whose major property is the path to an application.
- Group objects, which represent a group of users.
- LDAP Groups, which represent one or more LDAP servers.
- LDAP Servers, which represent an LDAP server and its configuration information.
- NDPS Broker, NDPS Manager, and NDPS Printer, which are NDPS printing objects. These are discussed in detail in Chapter 11, "NetWare 6.5 Printing."
- Organizational Role objects, which represent a group of users who have a common task. These are task-oriented objects whose members are defined as occupants.
- Print Queue, Print Server, and Printer, which are legacy queue-based printing objects.
- Profile objects, which are objects whose major property is a profile login script that is used by groups of users who need the same environmental configuration upon login.
- Server objects, which represent a NetWare Core Protocol (NCP) server. These objects are created when a server is installed into the tree.
- Template objects, which are objects with properties that can be applied to users, when created, who have properties with the same values.
- User objects, which are the most fundamental objects in eDirectory. They represent a person who needs access to network resources. A User object must be created before a user can log in to the network.

- Volume objects, which represent a physical amount of storage space on a server. In a variety of NetWare management utilities, you can manage files and directories by opening the Volume object, even though the Volume object does not hold file and directory information.

You have now been introduced to the three 3s: objects, properties, and values; Tree objects, Container objects, and Leaf Objects; Country, Organization, and Organizational Unit. Based on which objects you decide to use in your tree, you can create a variety of tree designs. The simplest design has a Tree object, Organization object, and all Leaf objects in the Organization object. This design is also known as the Bamboo Tree. It looks like and acts like a NetWare 3 environment when only a single server is installed. You can also have a tree with the Tree Object, Organization object, Organizational Unit object, and Leaf objects in the Organizational Unit. This has long been the standard Novell sanctioned tree design. That being said, you can configure different designs based on your needs and the hierarchical rules of eDirectory.

Now that you know what the objects are and what designs they can produce, you need to know how to name objects in the tree.

Identify the Flow and Design of the eDirectory Tree

The eDirectory tree can be as small or as large as you want. The more objects and containers that you include in the design, the more that locating and naming those objects becomes a task.

Objects are named from the Leaf object up to but not including the Tree object. To identify an object in eDirectory, you must be familiar with six concepts:

- Context
- Distinguished Name
- Typeful-using Attribute types
- Typeless
- Current context
- Relative Distinguished Name

An object's context is where the object is located in the eDirectory tree. Context is the full path to an object in eDirectory. In Figure 5.2, Alexandra's context is (starting from the leaf object) Educational-Sales-3WsNetWorking-US, because Alexandra's user object exists in the Educational Organizational Unit, which exists in the Sales Organizational Unit, which exists in the 3WsNetWorking Organization, which exists in the US Country object.

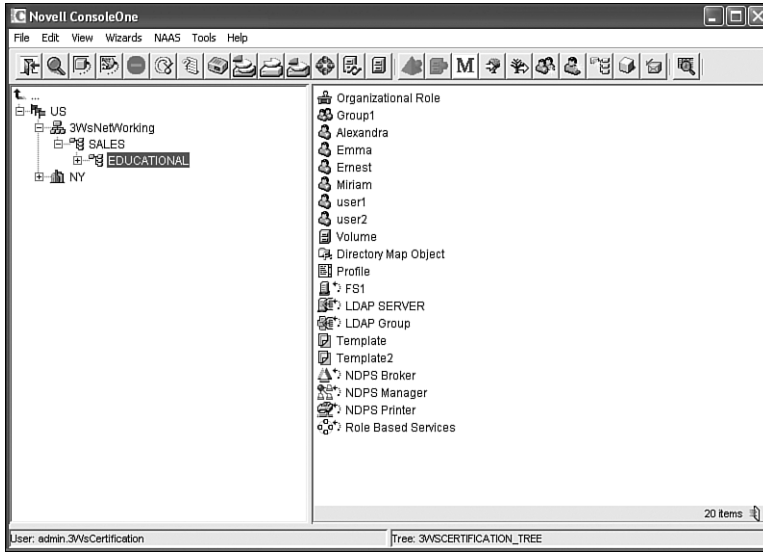

Figure 5.2 eDirectory naming.

A Distinguished Name is a combination of an object's common name and its context, beginning with the object and progressing up to, but not including, the Tree object. You can easily identify a Distinguished Name because it has a leading period, its contextual elements are separated by periods, and it does not have trailing periods.

A Distinguished Name is the best way for you to name objects in map statements, login scripts, and capture statements because you will always include the common name and the context of the object. If you are trying to locate or name an object in a multicontext environment that is not in your context, the best way to do so is with a Distinguished Name.

A Distinguished Name can be typeful or typeless, which means exactly that. In a typeful name, you type more, whereas in a typeless name, you type less. Actually, in a typeful name, you include an Attribute type followed by an equal sign to define what each naming element is. In a typeless name, you do not include the Attribute types.

Following are the four Attribute types:

- Country–C
- Organization–O
- Organizational Unit–Ou
- Leaf Object (Common Name)–CN

For the exam, these are the four Attribute types you need to know. Additional Attribute types are part of both the base schema and the schema when extended, but they are beyond the scope of this book.

Alexandra's typeful Distinguished Name is as follows:

.CN=Alexandra.Ou=Educational.Ou=Sales.O=3WsNetWorking.C=US

Following is Alexandra's typeless Distinguished Name:

.Alexandra.Educational.Sales.3WsNetWorking.US

Notice that each Distinguished Name begins with a leading period. That is the indicator of a Distinguished Name. Don't forget that. Also, note the lack of a trailing period at the end of the name. A trailing period indicates a Relative Distinguished Name.

Most NetWare utilities let you use either a typeful or typeless name to authenticate. Some utilities still maintain the need for a typeful Distinguished Name for authentication. The more that you use LDAP, the more you should become comfortable with typeful names.

In LDAP naming, contextual elements are separated by a comma, not a period. When you authenticate to Remote Manager or use ICE to import or export an LDIF file, you need to use LDAP naming conventions.

A workstation's current context is where the workstation is configured to look for objects in the eDirectory. This is called Name Context in some utilities. Current context reflects your current position, at your workstation, in the eDirectory tree.

When you try to authenticate to the tree, if you do not use a Distinguished Name, you have to be aware of your workstation's current context. Based on the container that the workstation is configured to look in, you have to adjust

your username to authenticate successfully. In Figure 5.2, if Alexandra's workstation had a current context of Sales.3WsNetWorking.US, and she tried to log in simply as Alexandra, she would fail to authenticate. Her user object is in Educational.Sales.3WsNetWorking.US. If her current context was Educational.Sales.3WsNetWorking.US and she tried to log in simply as Alexandra, she would succeed. Alexandra is her common name, or Relative Distinguished Name.

A Relative Distinguished Name is an object's common name relative to a workstation's current context. A Relative Distinguished Name never has a leading period, but it might have a trailing period. In addition, a Relative Distinguished Name's contextual elements are separated by periods. A trailing period (a period at the end of the name) is used as a shortcut to remove the leftmost such contextual element, as exists in the current context. If more than one trailing period is used, more than one contextual element is removed.

If Alexandra's current context is 3WsNetWorking.US, her typeful Relative Distinguished Name is as follows:

CN=Alexandra.Ou=Educational.Ou=Sales

Following is her typeless Relative Distinguished Name:

Alexandra.Educational.Sales

Neither of these examples has a leading period, which means each is a Relative Distinguished Name.

3WsNetWorking.US (refer to Figure 5.2) has a user named Ronnie. If Ronnie's current context is Educational.Sales.3WsNetWorking.US, Ronnie could authenticate using either a Distinguished Name or a Relative Distinguished Name with trailing periods. Ronnie's typeless Relative Distinguished Name would be this:

Ronnie ..

The two trailing periods remove the two leftmost such contextual elements as exist in the current context. In this case, those two trailing periods remove Educational and Sales from the current context. That would leave the current context as 3WsNetworking.US, where Ronnie's User object exists.

From this discussion, you can see that the following is true:

Relative Distinguished Name + Current Context = Distinguished Name

There's one important point that you shouldn't overlook: Every object in eDirectory has a Relative Distinguished and Distinguished Name. This includes Volume objects, Server objects, and Container objects. The rules that apply to naming a User object also apply to all other objects. In addition, the simplest name that users can log in with is their username with no trailing or leading periods. That is their Relative Distinguished Name when their current context is pointing to their context.

You must know the difference between Distinguished and Relative Distinguished Names, typeful and typeless, and Context and Current context. Practice naming the objects you see in Figure 5.2. Be sure you know which has a leading period and which does not.

One cool utility that you can use with your current context is CX. CX is a NetWare command-line utility that allows you to view or change your current context.

To see all of the possible options that are available with CX, at a command prompt, enter the following (see Figure 5.3):

```
CX /?
```

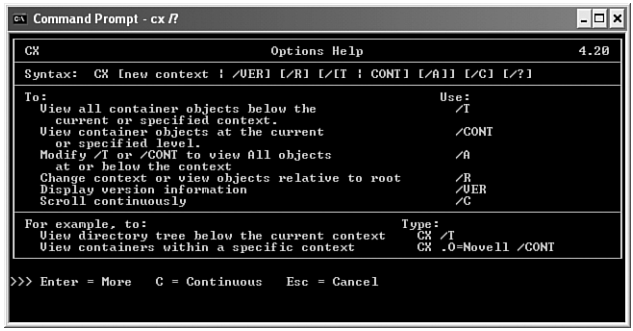

Figure 5.3 CX Help screen.

Following are the most widely used CX options:

- /R to change your current context to the Root of the tree
- /T to view all containers in your current context
- /A to view all objects in your current context

Enter the following:

```
CX /T /A /R
```

You will see all objects in all containers up to and including the Root of the tree.

You can also move from Container to Container either by entering a subcontainer's name or by using a trailing period to move up the tree. See Figure 5.4 for some examples of what you can do with CX.

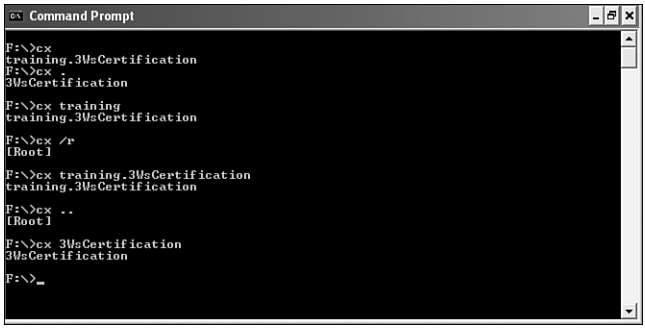

Figure 5.4 CX examples.

The last concept that is worth noting under this objective is inheritance. You will hear much more about inheritance in Chapters 8, "File System Security," and 9, "eDirectory Security," but because this objective deals with the flow and design of eDirectory, it's necessary to mention inheritance in passing.

When you design your eDirectory tree, with all of its Containers and Leaf objects, and you know how to name all of them, you need to take into consideration how the design affects both eDirectory security and file system security. Inheritance in this context is a security concept. It's the process of rights flowing down from a higher level to a lower level in both the eDirectory tree and the NetWare 6.5 file system. Consider how inheritance impacts security as you design your tree.

Now that you know how to name and locate eDirectory objects, you need to be able to identify four tools that will become your best friends when it comes to managing eDirectory.