

DirXML

Share Data Across Applications

Have you counted the number of databases your company maintains? Most companies maintain a minimum of five databases: an employee address book, a payroll database, a human resources database, an e-mail address book, and a network database. For many companies, this list is just the tip of the database iceberg; these companies also have several off-the-shelf or customized databases. Regardless of their purpose, most of these databases contain an element of common data, such as usernames or e-mail addresses.

In July, Novell announced a new product called *DirXML*, which will enable you to use Novell Directory Services (NDS) to link critical business data across your company. DirXML is currently under development; Novell plans to release this product sometime in 2000 and to make the beta version available by the end of this year.

DATA NETWORKING

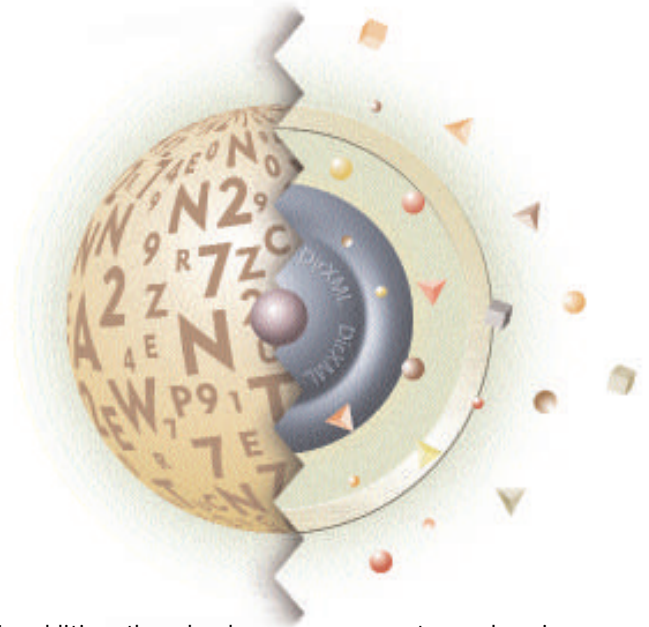
Companies and people rely on networks to share information, but until now most applications have been unable to share common information, such as e-mail addresses. If two or more applications require the same information, companies must enter this information in multiple databases.

DirXML enables applications to use NDS to store common data. As a result, companies need to enter this common information only once.

Using NDS or another directory to consolidate databases (thereby creating a meta-directory) is not a new idea. In fact, since the introduction of NDS, Novell has listed database consolidation as one of the many benefits NDS provides. For example, NDS enables you to consolidate network databases such as NetWare server databases and the databases of non-NetWare platforms such as Windows NT Server and Solaris.

However, Novell has not been quite as successful in consolidating application databases. There are several reasons why consolidating application databases has been more difficult: First, many companies use special-purpose, or legacy, applications. Redirecting these applications to use NDS would require a major, if not impossible, coding effort.

Using synchronization to create a common data store is a less labor-intensive solution, although coding is still required. Synchronization also introduces the problem of "dirty," or unsynchronized, data. If unsynchronized data is found, which database is the authoritative source?



In addition, there has been no easy way to synchronize common data. Most synchronization methods take an all-or-nothing approach—either synchronize all data or synchronize no data.

Second, developing NDS-integrated products requires detailed knowledge of the NDS architecture and application programming interfaces (APIs). A number of Independent Software Vendors (ISVs) have embraced NDS and have provided some degree of NDS integration (such as redirection or synchronization) for their applications. However, some ISVs do not have enough resources to send programmers to NDS development courses. As a result, these ISVs have not integrated their applications with NDS.

Finally, the issue of data ownership has prevented companies from using NDS to consolidate database applications. In many companies, separate departments maintain individual databases and do not want to give up this control. For example, how do you think the Telecommunications department would respond if you told them that the IT department was going to maintain the PBX database?

TAKING THE DIRECTORY TO THE APPLICATION

With these barriers in mind, Novell has designed a product that allows applications to share common data and requires almost no change to the applications. Rather than requiring developers to retrofit their applications or to write special code, Novell provides tools to allow applications to use information stored in NDS. In addition, each department that owns a database can maintain its own database, eliminating political battles over data ownership.

How DirXML Works

To provide NDS integration without requiring applications to be changed, Novell had to modify NDS. Since most applications need to access only a small subset of the information stored in NDS, DirXML will enable you to partition

the NDS tree in a way that makes sense for each application. You can then create custom replicas for applications, and these replicas will serve as connectors between NDS and the applications' native database.

Because NDS supports multiple writable replicas and NDS replication is bi-directional, both applications and network administrators can change the data stored in NDS. The normal NDS replication process will automatically update these changes throughout the network.

Virtual Replicas

DirXML includes virtual replicas, a new type of replica which enables you to partition NDS trees by object class, attribute, or even data value rather than by container object. For example, you can create a replica that contains only User objects and specific attributes such as telephone numbers and e-mail addresses.

To define which object classes and attributes are included in a virtual replica, you create a replica filter, which is stored as an object in the NDS tree. Servers that are aware of this new replica type use the replica filters to deliver the appropriate data to the virtual replica.

After you define what the virtual replica should contain, virtual replica servers filter out all other NDS objects and attributes. As a result, DirXML delivers only the data that applications need to access.

Virtual replica servers read in the replica filter information as they load. These servers send only updates to information contained in the virtual replica, thereby making better use of network bandwidth. If a network contains servers running older versions of NDS (pre-NDS 8), these servers will send all synchronization information. However, the virtual replica servers will simply filter out any information that is not needed.

Like other servers that store replicas, a virtual replica server can hold multiple virtual replicas, thereby supporting multiple applications.

Application Access

Applications can access virtual replica data by subscribing to replication events or by performing Lightweight Directory Access Protocol (LDAP) searches on the NDS data. If an application subscribes to replication events, the application receives updates to NDS data in NDS's

native format using its native APIs. If an application uses LDAP to search virtual replica data, the application attaches to the virtual replica server and searches an application-specific index, which is maintained by the virtual replica server.

Virtual replica servers automatically build application-specific indexes. These indexes speed up LDAP searches by eliminating the need to search the entire

NDS tree. These indexes also eliminate the need for LDAP catalogs, which require the use of a special catalog API.

XML Integration Drivers

To deliver NDS data to applications in their native format, DirXML uses integration drivers. Using the Extensible Markup Language (XML), the integration drivers apply the appropriate filtering,

Eicon 1/2 Page Island AD

4 7/8" x 7 3/8"
(4.875" x 7.375")

What Are XML, XSL, and XSLT?

Extensible Markup Language (XML) enables disparate directories to share data. Like HTML, XML is a subset of the Structured Generalized Markup Language (SGML). However, XML is much easier to learn and implement than SGML.

The World Wide Web Consortium (W3C) designed XML to enable developers to serve, receive, and process generic SGML on the web. Unlike HTML, XML retains the key SGML advantages of extensibility, structure, and validation. XML also differs from HTML in that it is more flexible and data-oriented than HTML. As a result, XML enables you to do the following:

- Define new tag and attribute names at will. With HTML, you cannot specify your own tags or attributes, making it difficult to systematically qualify data.
- Create complex nested document structures. HTML does not support the deep structures needed to represent database schemas or object-oriented hierarchies.
- Create an XML document that contains an optional description of its grammar for applications to use when performing structural validation. HTML does not support a language

specification that allows applications to check data for structural validity.

XSL

Extensible Style Sheet Language (XSL) is a language for creating style sheets. Using an XSL style sheet, you can specify how a program should transform and format XML data. An XSL style sheet processor applies the XSL style sheet to XML source content. The content is then presented in the manner defined in the style sheet.

XSLT

The W3C designed Extensible Style Sheet Language Transformations (XSLT) to be used with XSL. XSLT is a language that is used to transform XML documents into other XML documents. As mentioned above, XSL includes an XML vocabulary for specifying formatting. When you use XSLT with XSL, XSL uses XSLT to determine how an XML document should be transformed into another XML document that uses the formatting vocabulary.

Although you can use XSLT independently of XSL, the W3C designed XSLT primarily for use with XSL rather than as a completely general-purpose XML transformation language. ●

data transformation, object mapping, and event mapping to NDS data before delivering the data to the application. (For more information about XML, see "What Are XML, XSL, and XSLT?") Integration drivers then receive data from applications and use XML to convert the data to the NDS format. Using XML eliminates the need to do any NDS-specific coding.

Because the integration driver interface is simple, developers can synchronize applications with NDS without performing a lot of coding. The integration driver interface consists of four API calls written in the Java or C language and one or two style sheets written in Extensible Style Sheet Language (XSL).

XSL style sheets are simple text files that XML uses to export data from NDS or to import data into NDS. When you use DirXML to create XSL style sheets, they are stored as NDS objects.

XSL style sheets describe the rules that integration drivers use to transform data. The following are examples of these rules:

- Map attributes from one directory to another. For example, you can associate the e-mail attribute in NDS with the e-mail address in the Human Resources database.
- Map event types between directories. For example, when you delete user

JSmith from the Human Resources database, the JSmith user account will be locked in NDS.

- Perform data formatting conversions. For example, you can convert NDS data to the native format used by the Human Resources database or vice versa.

The following example shows how you can use a DirXML integration driver to link a Human Resources database with NDS. (See Figure 1.) When you create a new User object in NDS, the event is passed through the replication filter to the NDS event system, which sends it to the event handler. The event handler then sends the event to the XML generator, which converts the event into NDS-formatted XML data and sends the data to the XSL processor.

The XSL processor applies the appropriate XSL style sheet to the data and transforms the data into a format understood by the Human Resources database. For example, the database may use application-specific XML or LDAP Data Interchange Format (LDIF). If the application does not have an event interface or cannot accept XML input, DirXML uses an application interface to process the data into the application's native API calls. (An *event interface* is a process that allows an application to accept notification of changes to the database.)

If you create a user in the Human Resources database, the process described above is reversed. If the database requires an application-specific interface, this interface converts the event into an XML format native to the application. This XML datastream is submitted to the XML generator, which converts the datastream to NDS-formatted XML data.

The XML generator then sends the data to the XSL processor, which applies the appropriate XSL style sheet. Finally, the XSL processor sends the data to the XML receiver, which converts the NDS-formatted XML events into NetWare Core Protocol (NCP) requests.

WHAT NOVELL WILL DELIVER

Although DirXML is still in development, Novell has announced plans to deliver tools that will encourage the early adoption of the product. For example, Novell plans to provide the following tools:

- Prebuilt integration drivers for popular enterprise products, including groupware applications and Employee Resource Planning (ERP) applications, such as PeopleSoft
- A visual environment for creating and extending integration drivers
- A visual environment for extending the NDS schema and creating management snap-in modules

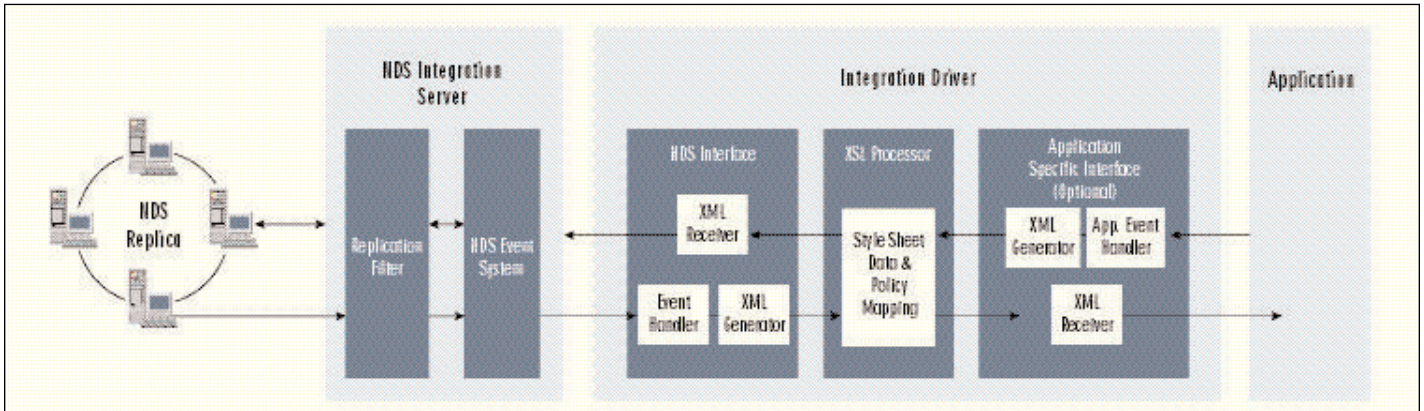


Figure 1. DirXML enables applications to share data that is stored in NDS.

- A visual environment for creating NDS-enabled forms for user input
- Tools for managing the data network

In addition, Novell is working with several companies to define a new language called the *Directory Services Markup Language* (DSML). DSML will standardize how developers use XML in directory interoperability. Through DSML, developers will be able to access and use directory

information without knowing the specific formats of each directory.

CONCLUSION

With DirXML, Novell will deliver a product that takes networking to a new level. For many years, companies have relied on the network to share files, e-mail messages, web pages, and so on. Until now, however, network applications could not share information with each other.

With DirXML, companies will be able to link all of their critical business data together while enabling the departments that own the data to control their own data. Best of all, DirXML will enable companies to link this data without modifying their existing applications.

Sandy Stevens is coauthor of Novell's Guide to BorderManager, Novell's Guide to Integrating NetWare and NT, and Novell's Guide to NetWare Printing.

Netoria
1/2 Page
AD

7 3/8" x 4 7/8"
(7.375" x 4.875")