Portable Distributed Objects (PDO)

NeXT's Portable Distributed Objects (PDO) provides the framework for distributed computing in a heterogeneous environment via high-level objects that encapsulate the details of object messaging over the network. With PDO, developers create custom objects under NEXTSTEP that can be deployed in heterogeneous server environments and shared throughout the network. By presenting a consistent object-oriented model to the developer, client-server applications can be modelled quickly and easily via object services. Developers creating applications utilizing PDO enjoy the benefits of object-oriented programming, including fewer lines of code, higher reusability of software and greater maintainability and reliability of software systems.

PDO's compiler uses the server's native calling conventions and linker to build applications making it simple to integrate object software with existing programs and libraries on the server. For example, a server may execute existing software that reads and manipulates data from an outside feed. A PDO application can be developed which creates an object from that processed data which can then be made available to other NEXTSTEP and PDO applications on the network.

Customers who are doing client-server computing today should utilize NEXTSTEP and PDO as the enterprise framework for distributed, client-server computing. PDO is intended for non-OpenStep compliant platforms. In the

future, as OpenStep becomes available in other OS environments that PDO is available for, the same server objects that were deployed in the PDO environment will be deployable in the OpenStep environment, with no source changes. This gives customers the best bridge to distributed applications, utilizing NeXT's Distributed Object Framework, to future OpenStep environments.

What is a simple example of why you might use PDO?
An example of a specific client-server application is easily found in the financial market. There are many places where interest rates, or other monetary models are used. In a non-client-server environment, the data and data models are spread all over the place and are likely to be inconsistent between user or client applications. But in a client-server world, you can consolidate all of the calculations and different data models into one application that is deployed on one server. A PDO application on a server gives you the advantages of object-oriented programming, including inheritance which simplifies the modification of data models and calculations, and encapsulation which provides a single point of interaction with the information.

The server application would contain objects that can be as complex as necessary because the server could deal with the computational needs, and the objects wouldn't have to be updated in many places. These objects would

encapsulate the calculation models for entire organizations. For instance, one of the objects could do interest rate calculations. Therefore only the server object would have to be modified when the model or interest rates changed. In this way, the firm can enforce a singular discipline/calculation/model for any financial application existing or for the future that deals with interest rates. Maintenance of this one object is far less complex than maintaining the same lines of code in multiple applications, often at multiple sites. In fact, the object could be be tied to the one data feed that a firm might have with changing information about current financial information. On the client side, many different and far-flung user applications would message that one object on the server for the current rate for a given transaction and get accurate and consistent information throughout the corporation. For instance there might be an application which generates a financial report each quarter, and that would query the same object on the server for the interest rate info as does the interactive customer services applications on the desks of the customer reps.

How does the NeXT/SunSoft agreement affect the PDO product?
PDO is the portable subset of NeXT's Distributed Object Framework. This framework is part of the OpenStep spec. Our object model, and therefore NEXTSTEP, is based on transparent distribution and access of objects within a client-server environment and relies heavily on the Distributed Object Framework.

SunSoft and other OpenStep licensee's will support NeXT's Distributed Object Framework so applications that are developed in NEXTSTEP using Distributed Objects or Portable Distributed Objects will port directly across to Solaris or other OS's when they are OpenStep compliant.

PDO is intended for non-OpenStep compliant platforms. PDO adds NeXT's Distributed Object Framework to operating system environments that do not implement the OpenStep specification. There will be a class of server OS's that may not support OpenStep, and PDO will be the vehicle to allow customers to continue to deploy server objects in those environments, utilizing the homogenous object model that is presented in an OpenStep application development environment.

Customers who are doing client-server computing today should utilize NEXTSTEP and PDO as the enterprise framework for distributed, client-server computing. As OpenStep becomes available in other OS environments that PDO is available for, the same server objects that were deployed in the PDO environment will be deployable in the OpenStep environment, with no source changes. This gives customers the best bridge to distributed applications, utilizing NeXT's Distributed Object Framework, to future OpenStep environments.

Is PDO CORBA compliant?
NeXT's implementation of its Distributed Objects Framework, and therefore PDO, is compliant with OMG's CORBA specification, with the exception that NeXT does not supply an IDL compiler and interface. NeXT is currently working with SunSoft on a joint submission for an Objective C to IDL interface mapping. Our anticipation is that this submission will occur this year.

For further CORBA support, NeXT is following the CORBA 2.0 process (which will probably not be adopted until sometime in 1995). NeXT has declared its intention to support CORBA as the interoperability standard for differing object models. Given this goal, CORBA 2.0 is essential to providing global interoperability between vendors. As details of CORBA 2.0 become clear, NeXT will clarify any effects on its product offerings.

What are the new features in PDO 2.0?

- C++ integrated with ANSI C and Objective C compiler
- Adds libg++ 2.5.3-a library of useful core C++ objects
- New versions of the Development Tools
  gcc 2.5.8

- gdb 4.12
- gas 2.2.u7

- Portable make
  - Compatible with NEXTSTEP's make
    - Allows one set of makefiles for source code shared between NEXTSTEP and PDO platforms
    - Allows seamless use of Project Builder to create and maintain projects on PDO servers from a NEXTSTEP client
  - GNU make v3.69
- Added support for SunOS, Solaris and DEC OSF/1

What objects does PDO support?

Objective C Classes:

HashTable
List
NXStringTable
Object

Protocol
Storage
StreamTable

MachKit Classes:

NXData
NXNetNameServer
NXPort
NXInvalidationNotifier
NXProtocolChecker

Distributed Objects Classes:

NXConnection
NXProxy
DOEventLoop

Future releases of PDO will migrate additional non-GUI objects from NEXTSTEP.

Can I use my PDO server as a compile server for my NEXTSTEP applications?
No. PDO's compiler cannot generate Multi-Architecture Binary (MAB) files.

Can PDO applications link with libraries built with the server's native linker and compilers?
Yes. The PDO development tools use the native calling conventions of the host server to provide this functionality.

What about linking with other C++ libraries?
PDO's compiler is an integrated ANSI C/ObjC/C++compiler that allows you to write code which messages between Objective C and C++ code. PDO's compiler is not guaranteed to be compatible with other C++ compilers as is the case with most C++ compilers from other vendors.

Can you write PDO server applications which "serve" NeXTSTEP/PDO clients as well as X windows clients or other RPC or socket based applications?
Yes. There should be no problem with doing this, especially with PDO 2.0. One should use DOEventLoop to register PDO servers, and use the "addFileDescriptor" method to handle the sockets.
Also, sometimes packages require using a certain malloc. If this is the case, then malloc.o needs to be deleted from the

libpdo.a archive and replaced with their own zone package.

What network protocol is the PDO messaging based upon? And how reliable are the messages that get sent?
PDO uses the same network protocol as Mach does with NMSERVER. This protocol is layered over TCP/IP and UDP/IP. TCP/IP is used for delivery of messages, while UDP/IP is used for name lookups and port bookkeeping. Messages are delivered reliably in the same sense as TCP is reliable. That is, your message is either delivered to the remote end or you find out through port death notification that the connection has been broken.

How does a PDO client object resolve reference to a vended PDO server object? Broadcast Protocol? NM Server Caching? What happens if multiple servers attempt to vend with the same name?
Name lookup is handled by the nmserver. The nmserver supports host directed name lookup, or a broadcast lookup. Within one machine, only one server application can vend a name; however, in a network, multiple machines could vend the same name. If multiple machines are vending the same name, a lookup operation will find the correct server when using host directed lookup. If broadcast lookup is used, the first server to respond will be used and others ignored. Note that it is possible to build more elegant name lookup functionality on top of this basic set of name services---especially in conjunction with a network name database like NetInfo.

Can I get a list of hosts which vend or publish a given object?
No. It would be useful to get a list of vendors of a given object so that messages could be directed at a specific server. It is theoretically possible for a third party to write software to provide this service, but this has not been done yet.

What other information is available for PDO?

- New PDO 2.0 DatasheetÐ1-800-TRY-NEXT document #1M5222
- PDO 2.0 Server Compatibility GuideÐNeXTanswers #1593
- *Spreading the Wealth: DO and PDO* ÐNXApp (Vol 1, winter 1994)ÐNeXTanswers #1506
- PDO White PaperÐ*Interoperability through Distributed Objects*Ð1-800-TRY-NEXT document #1M5122.01
- Mini-Examples:
  SortingInActionDistributed
  DistributedCalculator-simple code example and demo
  These are not currently released, but are coming soon and may have different names when they do get released.
- This document and the published price list.