

# Release Notes:

## Enterprise Objects Framework 2.1

This document contains the most up-to-date release notes for release 2.1 of Enterprise Objects Framework.

The MacOS version of this release is 3-way fat; it can be used to develop software for NeXT, Intel, and SPARC. The OpenStep for Windows version of this release can be used to develop software for Intel machines running Windows NT. The PDO version of this release can be used to develop software for Solaris and HPUX.

These release notes are organized into the following sections:

- Information About Other Releases
- Notes specific to Windows NT Developers
- Database Client Libraries
- New Features in Release 2.1
- Installing the Examples
- Enterprise Objects Framework User Defaults
- Locating the Documentation for this Release
- Known Problems in Release 2.1

**Note:** Because release 2.1 of Enterprise Objects Framework is based on the same architecture as release 2.0, many of the subjects in this document are relevant to both

releases. Consequently, the term "release 2.x" is used where a discussion applies equally to releases 2.0 and 2.1.

## Information About Other Releases

This document describes the features introduced, the bugs fixed, and the known problems in Enterprise Objects Framework release 2.1.

For a description of the features introduced in Enterprise Objects Framework release 2.0, see the 2.0 release notes in NeXTAnswers. They contain a lot of information that's still relevant for 2.1, including information about converting release 1.x applications to 2.0 and a list of features shared with Enterprise Objects Framework release 1.2.

For a description of how Enterprise Objects Framework 2.0 and 2.1 differ from the preceding releases, see the document *Differences Between Enterprise Objects Framework 1x and 2.0* in **/NextLibrary/Documentation/NextDev/EnterpriseObjects/1x\_To\_2.**

## Notes Specific to Windows NT Developers

References to filenames in this document use UNIX format. On Windows NT, you can interpret these by reading backslashes for the slashes and adding the installation directory (**c:\NeXT\** by default) to the beginning. For example, **/NextLibrary/Documentation** on Windows NT is **c:\NeXT\NextLibrary\Documentation.**

To use Enterprise Objects Framework on Windows NT, you must have the appropriate database client libraries. The Sybase client libraries are provided on the OpenStep Enterprise 4.2 CD as an optional package. To install the Sybase client libraries, you must do a custom installation and explicitly specify that you want to install the package. To use Enterprise Objects Framework with Oracle or Informix, you must purchase the appropriate client libraries from your database vendor.

Here's what you need:

**Oracle**

Phone: (800) 542-1170

Ask for: SQLNET v2.2 for PC/Windows NT

The Oracle adaptor on NT requires the Oracle 7.3 or 7.2 Client Library. It won't work with the 7.1 libraries.

**Informix**

Phone: (800) 331-1763

Ask for: ESQL/C version 7.2 for Win32

## Notes Specific to PDO Developers

To use Enterprise Objects Framework on PDO, you must have the appropriate database client libraries.

Here's what you need:

**Oracle**

Phone: (800) 542-1170

Ask for: 7.3.2 SQLNet V2 TCP/IP Client libraries

### **Informix**

Phone: (800) 331-1763

Ask for: ESQL/C Version 7.20.UC2

### **Sybase**

Phone: (800) 685-8225

Ask for: OpenClient/C Version 10.0.4

On PDO applications must explicitly link against the adaptor framework and the client libraries. For an example of how to do this, see the makefiles in the example "SimpleFetch" in **NextDeveloper/Examples/EnterpriseObjects**.

## Database Client Libraries

This section includes some tips on using database client libraries with Enterprise Objects Frameworks. It is organized by database vendor.

### **Oracle**

On Windows NT, using the latest release of the client library (7.3) requires you to use SQL\*Net v2, which requires a **tnsnames.ora** file. **tnsnames.ora** is a file that you put on client machines, generally in the directory **Orant/Network/Admin**. The file contains information needed to connect to a server over the network. Entries in **tnsnames.ora** are keyed off of a server ID alias, and they include information such as the server ID, the host machine name, and the network protocol used by the client library to resolve the server ID alias. An entry in **tnsnames.ora** might resemble the following:

```
myServerAlias = (DESCRIPTION= (ADDRESS= (PROTOCOL=tcp)
```

```
(HOST=myMachine) (PORT=1521)) (CONNECT_DATA=(SID=eof)))
```

Oracle provides tools you can use to create **tnsnames.ora** files. Refer to your Oracle documentation for more information on **tnsnames.ora** files and the tools you can use to create them.

If you're using the 7.2 version of the Oracle client libraries on Windows NT or if you're using Mach clients (the Oracle client included in the Mach release is 7.2), you can use either SQL\*Net v1 or SQL\*Net v2. To use SQL\*Net v1, you should set your adaptor's connectionDictionary **serverId** entry to "T:<host-machine>:<server-name>".

To use SQL\*Net V2 on Mach clients, you should create a **tnsnames.ora** file and put it in the **/etc** directory.

### Informix

If you get the error "INFORMIXSERVER not in sqlhosts file (25596)" but can connect to your database server using the Informix ilogin program, you may need to run SetNet32 to update the environment variables used by Informix.

The Informix client libraries appear to have redundant sources of server information. They use the **sqlhosts** file (**\$INFORMIXDIR/etc/sqlhosts**) as well as a collection of environment variables managed by the Setnet32 program.

See your Informix documentation for more information on the sqlhosts file and the Setnet32 program.

## New Features for 2.1

### Integration with Project Builder and Interface Builder

- There is a new Enterprise Objects Framework Wizard that automates the creation of simple Enterprise Objects Framework applications. When you create an application in Project Builder, you have the option of either creating an empty Enterprise Objects Framework application or using the wizard to automate the creation of the user interface.
- Interface Builder now has a "New Database Interface" command that invokes the Wizard to add a new interface to an existing project.
- You can display totals, averages, and other aggregate computations without writing any code. Interface Builder now allows you to form connections using special keys. For example, you can associate the maximum salary to a text field using the key **@max.salary**.
- You can create "query by example" user interfaces without writing any code. Using Interface Builder, you can connect user interface control objects to special keys such as **@query=.departmentName** or **@query>.budget**, and Enterprise Objects Framework constructs and applies qualifiers for them.
- In Interface Builder, you can specify a qualifier for an EODisplayGroup's EODatabaseDataSource using the Data Source Inspector.
- Interface Builder allows you to set a fetch limit for an EODisplayGroup's EODatabaseDataSource.
- You no longer have to manually add key paths to an EODisplayGroup in Interface Builder. Instead, you can use the Interface Builder's Inspector to traverse relationships and connect to the desired property.
- When you connect a user interface control to an EODisplayGroup in Interface Builder, a formatter based on the data type of the associated property is automatically added to the control.
- There is a new EOComboBoxAssociation.
- There is a new EOArrayDataSource class.

## **EOModeler**

- EOModeler has a new "Explorer" outline interface that makes navigating the model easier.
- EOModeler provides an Inspector for editing the connection dictionary.
- EOModeler has a new Diagram View that helps you visualize relationships between entities.
- EOModeler supports cutting and pasting model objects both within and between models. You can also drag and drop model objects within and between models.
- EOModeler supports quicker creation of EOModels by reading only the tables you specify.

## **Adaptors**

- The Oracle and Informix adaptors link with the newest client libraries.
- The new Oracle login panel is designed to work more naturally with SQL\*Netv2.

## **Framework**

- The Framework supports whole-table caching for read-only entities, thereby avoiding redundant fetches of reference data.
- Exceptions raised by the framework now contain more information. For example, validation exceptions include the object and property that failed validation, and update exceptions indicate the object and database operation that were involved in the failure.

## **Examples**

- The Enterprise Objects Framework examples have been updated.

# Installing the Examples

This release provides on-line examples to help familiarize you with Enterprise Objects Framework 2.x. These examples are located in **/NextDeveloper/Examples/EnterpriseObjects**. Installing the examples involves these steps:

- Setting up users and databases on your database server for the example databases.
- Installing the example directory.
- Populating your database server with example data.

### **Setting up Database Accounts**

The Enterprise Objects Framework 2.x examples use two sets of tables: Movies and Rentals. Some examples use just one of these databases, while others use both. The multi-database support in Enterprise Objects Framework 2.x makes it possible for you to install these databases in three different configurations:

- Both sets of tables together in a single user/database.
- Each set of tables in its own user/database on the same database server.
- Each set of tables on its own database server (for example, Movies on Informix, Rentals on Oracle).

Depending on your desired setup, you use the tools available with your database server to set up one or two new user/databases. For example, on Sybase you might create a new database on your server called <sup>a</sup>Movies<sup>o</sup> and login with the user <sup>a</sup>sa<sup>o</sup>. On Oracle you might create a new user with the name <sup>a</sup>Movies<sup>o</sup>. Once you have set up these accounts, you're ready to install the examples.

### **Copying the Example Directory**

To configure and build the examples you need to copy the example directory to a writable area in your file system. You can do this by copying the **/NextDeveloper/Examples/EnterpriseObjects** folder into your home directory (or any other directory writable by you).



## Configuring the Example Models

The model files used by the examples must be configured to use your adaptor and server. To configure the examples, run the **configure\_examples** program in your copy of the examples directory. It will ask you for the name of the adaptor you wish to use (Informix, Oracle, Sybase, and so on) and for the login information for your database. It will then convert the example models for your server.

## Populating the Databases

Now that the examples are configured, you can fill your example databases with sample data. The **install\_database** tool in the **DatabaseSetUp** directory will connect to your databases, add the example tables, and fill them with data. If you later wish to remove the data, simply run the **drop\_database** tool.

## Building the Example Programs

With your example projects installed and your database filled with data, you are ready to build and run the examples. To do this, in a command shell **cd** to your example directory and type **make all**. This performs a **make install** on **BusinessLogic.framework** and **EOExtensions.framework** to put them in **/LocalDeveloper/Frameworks**, where they are shared by many of the other examples. It then makes all of the examples applications.

**Note:** The **/LocalDeveloper/Frameworks** directory must be created and writable by you in order to build the examples. On Mach this can be accomplished by using **su** to become the superuser and then executing the following commands:

```
mkdirs /LocalDeveloper/Frameworks
chmod a+w /LocalDeveloper/Frameworks
```

# Enterprise Objects Framework Defaults

This section lists the user defaults for which Enterprise Objects Framework checks and changes its behavior.

### **Defaults in the NSGlobalDomain**

To change these defaults, issue a command such as the following from a shell:

```
defaults write NSGlobalDomain EOAdaptorUseBindVariables NO
```

#### *EOFDebugEditingContext*

Default is NO.

EOControl Layer will log every time an object is changed.

#### *EOFDebugUndo*

Default is NO.

Logs each time something is pushed on or popped off an undo list.

Also logs in `[_EOUndoStack dealloc]`.

#### *EOAdaptorDebugEnabled*

Default is NO.

Logs for connection attempts, all transaction activity (begin, rollback, commit), and SQL statements (select, update, etc.).

#### *EOProjectSourceSearchPath*

An optional array of paths in which EOAccess searches for framework project directories that contain models.

#### *EOAdaptorUseBindVariables*

Default is YES.

Controls whether or not the SQL generation process will use bind variables (for Oracle and Informix).

### *EOOracleTableNamesSQL*

SQL statement used to identify tables for the model reverse-engineering process.

Default is:

```
SELECT TABLE_NAME FROM USER_TABLES ORDER BY TABLE_NAME")
```

If you would like the model reverse-engineering to look at a different set of the tables, you can provide your own SQL statement, for example:

```
SELECT TABLE_NAME FROM USER_TABLES ORDER BY TABLE_NAME  
WHERE TABLE_NAME not in ('MY_USELESS', 'FOO')
```

### *EOOracleUseNoWaitLocks*

Default is NO.

Controls whether Oracle uses NOWAIT locks.

### *EOSybaseInterfacesFile*

Default is unset.

If the default is set, the SybaseAdaptor initializes the client library to run with this interface the specified interfaces file. This default is not applicable on Windows NT.

### *EOSybaseTableNamesSQL*

This default can be used to change the default SQL used to determine the tables that are analyzed in creating a new model.

Default is

```
select name from sysobjects where type = 'U' or type = 'V'
```

### *EOSybaseAttributeSQL*

SQL statement used to get the attribute information for a given table.

Default is

```
select a.name attr_name, b.type attr_type, b.name type_name,  
b.allownulls allownulls, c.type primary_key, a.length width, a.prec
```

```
prec, a.scale scale from syscolumns a, systypes b, syskeys c where a.id = object_id('%@') and a.id != c.id and (a.colid != c.key1 or a.colid != c.key2 or a.colid != c.key3 or a.colid != c.key4) and (c.type = 1 or c.type is null) and a.usertype = b.usertype and datalength(a.name) > 0 order by a.name
```

### *EOSybaseRelationshipSQL*

Default is

```
select a.keycnt, b.name tname, c.name cname, d.name dtname, e.name dcname from syskeys a, sysobjects b, syscolumns c, sysobjects d, syscolumns e where (b.type = 'U' or b.type = 'V') and a.depid is not null and a.depid > 0 and b.id = a.id and b.id = c.id and ((c.colid = a.key1 and e.colid = a.depkey1) or (c.colid = a.key2 and e.colid = a.depkey2) or (c.colid = a.key3 and e.colid = a.depkey3) or (c.colid = a.key4 and e.colid = a.depkey4)) and d.id = a.depid and d.id = e.id order by b.name, d.name
```

### *EOSybaseStoreProcedureSQL*

Default is

```
select name, id from sysobjects where type = 'P'
```

### *EOSybaseStoredProcedureDetailsSQL*

Default is

```
sp_sproc_columns %@
```

### *InformixINFORMIXSERVER*

No default value.

The name of the Informix server.

### *InformixInformixTableNamesSQL*

This default can be used to change the default SQL used to determine the tables that will be analyzed in creating a new model.

Default is

```
select tabname, owner, tabid from informix.systables where (tabid > 99)
and (tabname <> 'ANSI') order by tabname
```

### **Defaults in the EOModeler Domain**

To change these defaults, issue a command such as the following from a shell:

```
defaults write EOModeler DisableInheritanceCheck YES
```

### *BundlesToLoad*

If you write extensions to EOModeler, you can get EOModeler to load them by setting this default. This default is an array, so remember to include parentheses when you set it. For example:

```
defaults write EOModeler BundlesToLoad
"(/LocalDeveloper/ModelerBundles/MyModelerBundle.bundle,
$HOME/eoexamples/ModelerBundle/ModelerBundle.bundle) "
```

### *RECORD\_FETCH\_LIMIT*

Default is 100.

For the Data Browser in EOModeler. This controls how many rows to fetch before asking users if they want to fetch all, fetch another n rows, or just stop now.

### *Consistency Checks*

All of the following defaults have the default NO:

- DisableConsistencyCheckOnSave
- DisableExternalNameCheck
- DisableInheritanceCheck
- DisablePrimaryKeyCheck
- DisableRelationshipCheck
- DisableStoredProcedureCheck

### *SkipBeautifyNamesOnModelCreation*

Default is NO.

Setting this default to YES will cause EOModeler to leave the entity names and attribute names matching the exact names of their corresponding database objects (including the case).

### *DisableAdvancedOptions*

Default is NO.

If set to YES, the EOModeler runs in "Simplified mode". The advanced inspectors don't appear, the choices of table view columns is limited, and so on.

## Locating the Documentation for this Release

The documentation for this release is provided in PDF and Windows Help formats. On Windows NT, you can view documentation by choosing OPENSTEP Books On-Line from the OPENSTEP Enterprise program group. You can also find the documentation in the following on-line locations:

- The *Differences Between Enterprise Objects Framework 1x and 2.0* document describes how the product has changed between the 1x and 2.0 releases. See **/NextLibrary/Documentation/NextDev/EnterpriseObjects/1x\_To\_2**.
- *Using Enterprise Objects Framework with OPENSTEP* provides a tutorial and information on creating an Enterprise Objects Framework application on OpenStep. In addition to the hard copy book, a version of the book is provided on-line in PDF format. See **/NextLibrary/Documentation/NextDev/EnterpriseObjects/UsingEOFwithOS**.
- The *Enterprise Objects Framework Developer's Guide* provides a conceptual

overview of the product. In addition to the hard copy book, a version of the book is provided on-line in PDF format. See

**/NextLibrary/Documentation/NextDev/EnterpriseObjects/Guide.**

- The *Enterprise Objects Framework Reference* includes class specifications for the Enterprise Objects Framework classes. Reference documentation is located in **/NextLibrary/Frameworks** in one of these locations:

/EOAccess.framework/Resources/English.lproj/Documentation/Reference  
/EOInterface.framework/Resources/English.lproj/Documentation/Reference  
/EOControl.framework/Resources/English.lproj/Documentation/Reference  
/InformixEOAdaptor.framework/Resources/English.lproj/Documentation/Reference  
/ODBCCEOAdaptor.framework/Resources/English.lproj/Documentation/Reference  
/OracleEOAdaptor.framework/Resources/English.lproj/Documentation/Reference  
/SybaseEOAdaptor.framework/Resources/English.lproj/Documentation/Reference

#### **If You Need a PDF Reader...**

PDF readers for various platforms can be found on the OPENSTEP Enterprise CD-ROM in the /3RD\_PRTY directory. Or, you can download one from one of the following web locations:

- For Windows or Solaris: **<http://www.adobe.com/acrobat/>**
- For NEXTSTEP: **<http://www.BITart.com/PDFView/>**

**Note:** If you're using the Acrobat Reader to view PDF files on Windows NT, some text may initially appear as gray lines. If this happens, you can fix it by choosing File -> Preferences -> General. In the General Preferences panel, uncheck the Greek Text option.

#### **Other Documentation**

Depending upon your application, the *OpenStep Conversion Guide* may also provide useful information to you. It discusses how to use the automated conversion process

to convert applications to OpenStep. See  
**/NextLibrary/Documentation/NextDev/Conversion/ConversionGuide.**

### **Documentation Feedback**

Your comments on our documentation are especially valuable. Please send electronic mail with your comments and suggestions to **techpubs\_feedback@next.com**. Please include **EODoc:** as the first part of your message's subject.

## Known Problems in Release 2.1

This section describes known problems with release 2.1 of Enterprise Objects Framework. It is organized into the following sections:

- Access Layer
- Control Layer
- Interface Layer
- Miscellaneous Framework
- Informix Adaptor
- Oracle Adaptor
- Sybase Adaptor
- EOModeler
- On-line Examples
- Documentation

### **Access Layer**

These problems exist in the access layer of this Enterprise Objects Framework release:

Reference: 77935



Problem: EOModelGroup's **setDelegate** method doesn't work correctly.

Description: **setDelegate** builds a lookup list to determine which methods the delegate implements. Unfortunately in the case of **subEntityForEntity:primaryKey:isFinal:** it checks the selector **subEntityForEntity:primaryKey:**.

Workaround: Have the delegate implement a dummy method **subEntityForEntity:primaryKey:**.

Reference: 77886

Problem: There are severe performance problems when you perform prefetching using hints or batch faulting

Description: The algorithm used for batch faulting and relationship prefetching for to-many relationships is  $O(n^2)$ . This means that fetching 1000 records takes 100 times longer than fetching one tenth as many, rather than  $O(1) = 10x$ .

Workaround: You can use the statement [EODatabaseContext \_setUseFastBatchAlgorithm:YES] to enable an alternative algorithm. Due to time constraints, this new algorithm has not been qualified as production quality.

Reference: 77721

Problem: Can't update rows containing string attributes with trailing spaces.

Description: The Enterprise Objects Framework adaptors automatically strip trailing spaces from string values fetched from the database. They also strip spaces on newly inserted strings before sending them to the database. This works fine until you fetch data that a non-Enterprise Objects Framework application inserted that contains trailing spaces. If that attribute is marked

as "used for locking," Enterprise Objects Framework will be unable to update the row (because the version in the database is different from the space-stripped version in the snapshot).

Workaround: Do not mark columns that could contain trailing spaces as used for locking. Alternately, strip the spaces from the data in your server.

Reference: 77656

Problem: Firing a snapshot fault after replacing a to-many relationship locks up the application.

Description: if you replace a to-many property fault (with **nil** or with another array) and then invoke **saveChanges**, Enterprise Objects Framework goes into an infinite fault firing loop.

Workaround: Update a to-many relationship using the methods **addObject** and **removeObject**; do not replace it with **nil** or another array.

Reference: 77546

Problem: Inclusion of framework models in defaultManagerGroup doesn't happen automatically on PDO.

Description: Applications on PDO that link with frameworks that include models don't automatically get the models put in the applications.

Workaround: Manually construct the the model group yourself at application initialization. For example:

```
EOModelGroup *group = [EOModelGroup new];
```

```
// repeat for each model containing framework used by app  
NSBundle *bundle = [NSBundle bundleForClass:[SomeClassInFramework class]]
```

```
NSEnumerator *enumerator = [[bundle pathsForResourceOfType:@"eomodeld" inDirectory:nil]
objectEnumerator];
NSString *modelPath;
while (modelPath = [enumerator nextObject])
    [group addModelWithFile:modelPath];

[EOModelGroup setDefaultGroup:group];
[group release];
```

Reference: 77631

Problem: If you're editing a model (in code or with EOModeler), sometimes an EORelationship will return **nil** from **inverseRelationship**, even though you just added the inverse relationship.

Description: The first time you ask an EORelationship for its inverse relationship, it searches all the relationships in its **destinationEntity** looking for an inverse. It caches the result of this search, and this cache does not always get invalidated when the relationships of the **destinationEntity** change.

Workaround: See the category on EORelationship in the **ModelerBundle/RelationshipExtras.m** example.

Reference: 77354

Problem: Problems saving changes to attributes with mutable custom value objects.

Description: Suppose you have a mutable custom value type, PhoneNumber, that implements methods such as **setAreaCode:** and **setPrefix:** to change an enterprise object's values. If you use such methods to modify an enterprise object's values and save changes, the enterprise object is not saved to the database.

Workaround: To use mutable value classes you must do three things:

1. In the PhoneNumber value class object, implement **isEqual:** to appropriately compare two instances.
2. If the PhoneNumber value is about to be modified, its owning enterprise object must invoke `[self willChange]` before the modification.
3. In your enterprise object class, you must implement a "set" method that copies rather than retains the object passed to it. For example:

```
- (void)setDayTimePhone:(PhoneNumber *)number {  
    [dayTimePhone autorelease]  
    dayTimePhone = [number copy];  
}
```

#1 would be required, even in a perfect world. #2 is a requirement of the basic Enterprise Objects Framework architecture. #3 is a bug in Enterprise Objects Framework -- Enterprise Objects Framework should be passing enterprise objects a copy of their values, but instead it's passing the the same instance that's shared in the snapshot.

In general, the simplest workaround is to use **immutable** custom value objects.

Reference: 72619

Problem: EOAdaptorChannel does not inherit EOAdaptor's delegate.

Description: According to the EOAdaptorChannel class specification and header file, the EOAdaptorChannel's delegate should automatically be kept in synch with its adaptor. Indeed, if a new delegate is assigned to an EOAdaptor, the new delegate is propagated to any of that adaptor's existing contexts and channels. However, if you create an EOAdaptor, assign a delegate to it, and

then create EOAdaptorContexts and EOAdaptorChannels, the newly-created contexts and channels don't have the delegate assigned to them.

Workaround: Wait until all of the channels and contexts are instantiated before assigning the adaptor's delegate, or reassign the adaptor's delegate each time a new channel or context is created.

Reference: 61475

Problem: Enterprise Objects Framework performs less efficient deep fetches for single-table inheritance mappings than it should.

Description: To perform a deep fetch, Enterprise Objects Framework performs a fetch for each concrete class in an inheritance hierarchy. For a single-table inheritance mapping, Enterprise Objects Framework should perform only one fetch and then sort the results in memory. For example, if Person, Employee, and Customer objects are stored in one table—the PERSON table—the Framework should perform one fetch on the PERSON table to fulfill a deep fetch request of Person, Employee, and Customer objects. Instead, it performs three fetches: one to get Person objects, one to get Employee objects, and another to get Customer objects.

Workaround: Define only one entity for the entire inheritance hierarchy, and use the EOModelGroup delegate methods **subEntityForEntity:primaryKey:isFinal:**, and **entity:classForObjectWithGlobalID:** to create instances of the proper subclasses from database rows.

Reference: 74251

Problem: Changes made during **saveChanges** are silently lost.

Description: If you change an enterprise object while its editing context is in its **saveChanges** method (for example, if you change an enterprise object in an EODataContext delegate method), the changes may be silently lost.

Workaround: Don't make changes to objects during the save process. Instead, make the changes from the EOEditingContext delegate method **editingContextWillSaveChanges:**.

Reference: 74345

Problem: You can't update to-many relationship if the foreign key isn't marked as a class property or as used for locking.

Description: Suppose that an entity's foreign key attribute isn't marked as a class property or as used for locking. If you designate the foreign key attribute as a to-many relationship's destination key, the foreign key value isn't always updated. This occurs because the destination entity doesn't know that the attribute participates in a relationship. Therefore, the destination entity doesn't fetch the foreign key from the database or update it.

Workaround: Mark attributes that are destination keys of a to-many relationship so they are fetched. For example, you could:

- Mark them as class properties.
- Mark them as used for locking.
- Use them in inverse relationships to the problematic to-many's source entity.

Reference: 74379

Problem: EODatabaseContext can't be the direct parentObjectStore of an EOEditingContext.

Description: The EODatabaseContext requires that an EOObjectStoreCoordinator sit between it and any EOEditingContexts that it serves. This is the default configuration set up by the framework, so you shouldn't normally run into this problem. Just a reminder, you can set up an editingContext and be ready to go with this one line:

```
EOEditingContext *editingContext = [EOEditingContext new];  
    // automatically uses [EOObjectStoreCoordinator defaultCoordinator]  
    // as parentObjectStore
```

Any necessary EODatabaseContexts are created and registered automatically.

Workaround: Don't assign an EODatabaseContext as the parentObjectStore of an EOEditingContext. There's no benefit to doing so anyway.

Reference: 76526

Problem: Applying a qualifier with key path to top of horizontally mapped inheritance hierarchy generates invalid SQL.

Description: Enterprise Objects Framework's query building mechanism doesn't handle relationships to inheritance hierarchies. For example, suppose that you are attempting to qualify a fetch through a to-many relationship (**planes**) that points to the top of a horizontally mapped inheritance hierarchy (for the entities Plane, FighterPlane, and TrainerPlane). If you want the query to test against all tables, you'd expect Enterprise Objects Framework to generate SQL similar to the following:

```

SELECT  t0.AIRPORT_ID
FROM PLANE t1, FIGHTER t2, TRAINER t3, AIRPORT t0
WHERE
    (t1.LENGTH <= 1000 AND t0.AIRPORT_ID = t1.AIRPORT_FK) OR
    (t2.LENGTH <= 1000 AND t0.AIRPORT_ID = t1.AIRPORT_FK) OR
    (t3.LENGTH <= 1000 AND t0.AIRPORT_ID = t1.AIRPORT_FK)

```

Instead, Enterprise Objects Framework generates the following SQL:

```

SELECT  t0.AIRPORT_ID
FROM PLANE t1, AIRPORT t0
WHERE
    (t1.LENGTH <= 1000) AND
    t0.AIRPORT_ID = t1.AIRPORT_FK

```

In other words, only the table for the root of the hierarchy is queried.

Workaround: You can create a qualifier that generates the correct SQL by:

1. Adding relationships in the source entity to all the tables in the inheritance hierarchy. For example, to the Airport entity, you'd add the relationships **toFighters** and **toTrainers** to the destination entities FighterPlane and TrainerPlane, respectively. Mark the relationships so they aren't class properties.
2. When building your query, explicitly list these extra relationships. In the Planes example, you'd fetch from Airport where "planes.length < 1000 OR toFigtherPlanes.length < 1000 OR toTrainerPlanes.length < 1000"

Alternatively, you might be able to solve this problem more generally by writing a post processor for EOQualifiers that splits up clauses that perform inheritance tests. The post processor could even programmatically generate the additional relationships on demand and register them with the model



using names like "plane\_Subclass\_Fighter".

A generic EOQualifier post processor could be wired into Enterprise Objects Framework so that application writers don't have to know it exists. The right place for such a mechanism is probably in EOKeyValueQualifier's **schemaBasedQualifierWithRootEntity:** method (see EOSQLQualifier.h). You could put the post processor code in a subclass of EOKeyValueQualifier (with an appropriate call to **super** after the transformation, if any, is performed) and have your subclass pose as EOKeyValueQualifier.

Reference: 47832

Problem: Enterprise Objects Framework can't update attributes whose internal types are custom (such as NSImages).

Description: A custom value class must implement **isEqual:** to be used for attributes marked as used for locking.

Workaround: Implement **isEqual:** in the custom value class or don't mark the attribute as used for locking.

Reference: 65078

Problem: Seemingly innocuous qualified fetch always causes exception.

Description: Some qualified fetches raise exceptions because values in the SQL have been formatted as strings when they should have been formatted as some other type. This can happen when you enter an invalid external type or when a Sybase model doesn't contain information about user-defined types that are used in the model.

Workaround: In the case of an invalid external type, simply correct it. In the case of a user defined type, create a new model by reverse engineering the database. The new model's connection dictionary contains information about user-defined types. Copy the connection dictionary from the new model to the original one.

Reference: 69039

Problem: Alert panel displaying adaptor error is never dismissed.

Description: This occurs whenever an adaptor operation that was invoked from an **awakeFromNib:** method displays an alert panel. The problem is that the entire object graph is not yet instantiated when **awakeFromNib:** is invoked.

Workaround: Database operations should be begun from the **applicationDidFinishLaunching:** method rather than the **awakeFromNib:** method. Of course, this means that any methods that indirectly cause database communication should also be invoked from the **applicationDidFinishLaunching:** method.

Reference: 76885

Problem: The set of valid values for the **databaseEncoding** entry of a connection dictionary are not documented, and the set can vary in different locales.

Description: A connection dictionary's **databaseEncoding** entry contains the localized name for the string encoding. The localized names are not documented. Furthermore, the localized names for string encodings can vary with the user's locale. Consequently, the specified encoding for an application might not work for users in different locales.

Workaround: You can find the localized name for string encodings in **`/NextLibrary/Frameworks/Foundation.framework/Resources/language.lproj/EncodingNames.strings`**.

Reference: 59472

Problem: Derived attributes are limited and don't offer full SQL as advertised.

Description: Placing a string or a numeric constant in the definition field of a derived attribute generates invalid SQL. Definitions such as `^title^` and `^0.0^` don't work correctly. However, definitions such as `^att1 + 5^` should work correctly when `^att1^` specifies another attribute.

Workaround: None.

Reference: 70049

Problem: `EOModelGroup` doesn't raise an exception when more than one entity has the same name.

Description: Although it is illegal to have the same entity name in two different models in a model group, `EOModelGroup` doesn't check to see if this is the case when adding a model.

Workaround: Manually verify that no two models have entities with the same name or write a method to perform the check.

Reference: 70260

Problem: Some error messages returned by the Framework's default validation methods (such as methods that check that a value doesn't exceed the

maximum width specified in its attribute) aren't localized.

Workaround: Implement an exception handler for use with the EOEditingContext, and have the handler replace the error messages with localized strings.

Reference: 76152

Problem: Inserting and deleting objects involved in inverse, to-many relationships can be very slow.

Description: Suppose that a ServiceRequest has a to-one relationship to its CustomerServiceRepresentative, and that the CustomerServiceRepresentative has an inverse to-many relationship to its ServiceRequests. When you assign a request to a representative using **addObject:toBothSidesOfRelationshipWithKey:**, you fire the fault for the **CustomerServiceRepresentative.serviceRequests**. So, if a representative has a large number of requests, assigning a new request to a representative can be very slow. Correspondingly, when you delete a request, Enterprise Objects Framework fires the corresponding representative's **serviceRequests** fault so it can remove the request from the array.

Workaround: Set the inverse, to-many relationship so it isn't a class property. For example, in the above request-representative scenario, you would remove the **serviceRequests** relationship from the class properties of the CustomerServerRepresentative entity.

## Control Layer

These problems exist in the control layer of this Enterprise Objects Framework release:

Reference: 77714

Problem: Qualifier for **nil** to-one relationship always returns no matches.

Description: If you want to fetch all employees without a department, you'd expect to be able to use the qualifier:

`"toDepartment = nil"`

Currently, this will always (incorrectly) return no matches because it's testing the primary key on the other side of the relationship rather than the foreign key on the employee side. This is a bug in the EOQualifier method **schemaBasedQualifierWithRootEntity**: when it translates object references to foreign key references.

Workaround: Query explicitly on the foreign key in the Employee object. For example, `"deptID = nil"`. This works even if deptID is not a class property of Employee. Note, however, that in this case the qualifier only works when run against the database, not for in-memory evaluation.

Reference: 77668

Problem: **EOArrayDataSource.h** isn't imported by **EOControl.h**.

Workaround: Explicitly import **EOControl/EOArrayDataSource.h** in your code.

Reference: 68146

Problem: Fault failure leaves EOInterface layer unstable.

Description: If a displayed enterprise object has a to-one relationship to a non-existent destination row, attempting to access the destination object raises an exception and corrupts the state of user interface objects. (Enterprise Objects Framework raises an exception when a to-one relationship cannot be

resolved due to a referential integrity problem in the database).

Workaround: See the chapter "Advanced Enterprise Object Modeling" in the *Enterprise Objects Framework Developer's Guide* for information on handling optional to-one relationships.

Reference: 64084

Problem: Inserted enterprise objects don't get removed from EODisplayGroup after sending **revert** to EOEditingContext.

Description: If you fetch objects into an EODisplayGroup, insert a few objects, delete a few objects, and then update a few objects; telling the EOEditingContext to revert backs out the updates, but not insertions or deletions. The actual insertions and deletions have been reverted, but the EODisplayGroup doesn't know how to revert its object list (since it doesn't keep track of what its original object list was before the insertions and deletions).

Workaround: Programmatically tell all affected EODisplayGroups to refetch after telling the EOEditingContext to revert, or use **refetch: (invalidateAllObjects:)** instead of revert.

Reference: 72177

Problem: Deleted objects are still registered in the EOEditingContext after a **saveChanges** operation.

Description: If an object is deleted in the EOEditingContext and then the EOEditingContext is saved, the deleted object isn't forgotten by the EOEditingContext (i.e., sending the EOEditingContext the message **objectForGlobalID:** should return **nil**, but it doesn't).

Workaround: Fortunately, this should have no affect on most applications. However, if you need to work around this problem, remember the deleted objects in the **editingContextWillSaveChanges:** delegate, and then invoke **forgetObject:** for each object after a successful save.

Reference: 72269

Problem: You can't tell Enterprise Objects Framework to not undo your changes when delete propagation fails.

Description: The EOEditingContext delegate method **editingContextShouldUndoUserActionsAfterFailure:** is supposed to allow programs to indicate that they do not want user actions undone if a validation or delete propagation error occurs. However, the EOEditingContext undoes the user action regardless of the return value from the method.

Workaround: None.

Reference: 72903

Problem: Aborted deletions are incorrectly recorded in the undo stack as empty undos.

Description: When the deletion of an object fails due to a deny rule, the undo manager records an empty undo group. This means that although nothing actually happened during the operation, the operation still needs to be undone before previous operations can be undone. For interactive programs this is not normally a problem since users rely on visual cues to determine how many times to undo. However, this may be a problem for code that programmatically performs undo operations.

Workaround: None.

Reference: 74965

Problem: The EOEditingContext delegate method **editingContext:shouldPresentException:** isn't called when **saveChanges** is invoked programmatically.

Description: If you use the following code:

```
eo = [[[Movie alloc] init] autorelease];  
[eo setTitle:nil]; // Assume that the TITLE column doesn't allow NULLs  
[editingContext insertObject:eo];  
  
[editingContext saveChanges];
```

the **editingContext:shouldPresentException:** delegate method isn't called. Instead, an exception is raised.

However, if you instead set a user interface control object as the target for the editing context's **saveChanges:** action, the delegate method is invoked.

Workaround: Always use **saveChanges:** instead of **saveChanges**. If you don't have an argument to pass, just use

```
[editingContext saveChanges:nil];
```

Reference: 76466

Problem: EOEditingContexts don't propagate uninserts.

Description: Propagate delete works fine for an object that that is fetched from the database. However, if an object is created and inserted in an editing context



and then deleted (thus never being saved to the database), the delete isn't propagated to the destinations of the object's relationships. For example, suppose you create an ExpenseReport object and several LineItems for it. If you then delete the ExpenseReport, the delete doesn't propagate to the LineItems. If the LineItems can't exist without an owning ExpenseReport, EOEditingContext's **saveChanges** fails when the delete is performed within the same event as the insert.

Workaround: Call [editingContext processRecentChanges] before deleting the previously inserted object.

Reference: 76901

Problem: Associations are refreshed too early when the fetch limit panel is raised, possibly causing an exception.

Description: Suppose you have two EODisplayGroups that are set to fetch on load. If the second display group to fetch exceeds the fetch limit, an alert panel inquiring whether to continue the fetch pops up. Presenting the panel causes the run loop to flush the first display group's user interface drawing. If the first display group's user interface displays enterprise object properties via key paths, EOFaults may be fired. Since the default EODatabaseChannel is busy with the fetch for the second display group, an exception is raised because there isn't an available channel.

Workaround: There are three workarounds:

1. Don't use the fetch "prompt on limit" feature.
2. Implement your own fetch limit panel that doesn't invoke a modal event loop.
3. Register an additional EODatabaseChannel with the EODatabaseContext so the fetch of the faults can occur while the

second EODisplayGroup fetch is paused.

Reference: 77181

Problem: The EOQualifier contains operator doesn't do SQL generation.

Description: The contains operator is meant to do to-many array comparison. For example, you might query on the Studio entity using the following qualifier format:

```
"movies contains %@", someMovieObj
```

The resulting qualifier works for an in-memory search, but doesn't generate SQL for a database search.

On the other hand, using the equality operator for to-many array comparison kind of works in SQL, but not in memory. For example, you might query on the Studio entity using the following qualifier format:

```
"movies = %@", someMovieObj
```

The resulting qualifier returns no matches when evaluated in memory because the array property is not equal to the movie object as determined by **isEqual:**. However, when it's evaluated in the database, it returns the same studio N times (where N is the number of movies in the Studio's movies array), even though the qualifier is attempting to match only one movie.

Workaround: Use the contains operator for qualifiers in-memory evaluation and the equality (=) operator (with **usesDistinct** on the fetch specification) for evaluation in the database.

Reference: 69934

Problem: EOQualifier needs a case insensitive 'like' operator that works both in memory and in SQL.

Workaround: You can use a pattern like `*[Hh][Ee][Ll][Ll][Oo]*` or you can create a method on the enterprise object that always returns a version in uppercase and use that method as a key in the qualifier.

## Interface Layer

These problems exist in the interface layer of this Enterprise Objects Framework release:

Reference: 77378

Problem: Undo of uncommitted edit in text field doesn't undo.

Description: Make a change in a text field, and then, without exiting the field, press undo. The change that you made in the text field is not undone.

This is happening because the editing context only knows about changes to enterprise objects. An uncommitted edit in a text field is not known to the enterprise object and thus not to the editing context or undo manager. Consequently, it won't be undone.

Workaround: Users can leave the field (tab or hit return) and then undo.

To make this work correctly, you'd need to subclass the `EOGenericControlAssociation` to register an undo record with the undo manager in **`control:textShouldBeginEditing:`**. The association would need to be smart enough to know to ignore callbacks from the undo manager when

it has already committed the edit (since at that point the edit would be undone by the editing context).

Reference: 71553

Problem: EOPickTextAssociation displays an alert panel whenever the '[' character is entered.

Description: EOPickTextAssociation should either escape all substitution characters or wait until a complete expression is typed in before executing. Since it does not, EOQualifier thinks that it has found an illegal pattern and raises an exception, which the EOPickTextAssociation displays in an alert panel.

Workaround: None.

Reference: 76902

Problem: Query by example does not work correctly for to-many relationship properties.

Description: If a query interface's user interface control is bound to a to-many relationship property (using the EODetailSelection association, for example) the resulting query searches on only the first object added to the to-many relationship array (instead of searching on all the objects in the array).

Workaround: Don't use Query By Example for to-many relationship properties. Implement your own query mechanism instead.

Reference: 57842

Problem: An EODisplayGroup doesn't show its keys in Interface Builder's Inspector.

Description: This occurs when Enterprise Objects Framework is unable to find the model file for an EODisplayGroup's entity. To verify that this is the problem, inspect the EODisplayGroup's data source (click on the EODisplayGroup, open the Inspector panel, and select the DataSource panel from the Inspector popup). If the Model field is blank, then Enterprise Objects Framework is unable to find your model.

Workaround: Quit Interface Builder and verify that the model containing the entity referenced in the EODisplayGroup is added as a resource in the project containing the nib file.

Reference: 65061

Problem: Interface Builder connections are broken when you make changes to a model.

Description: When the name of an EOAttribute or EORelationship is changed in a model, any EOAssociations that use the old name become invalid because EOAssociations' keys aren't automatically updated to the new name. Unfortunately, Interface Builder's Connection Inspector doesn't directly support breaking the connection.

Workaround: To repair broken connections:

1. Select the display object that's bound to the obsolete key.
2. Use the associations popup to select "Outlets."  
This removes the old association connectors.
3. Reselect the appropriate association type and reform the association to the new key.

Reference: 67804

Problem: EOModeler can't find model from another framework.

Description: This problem has two symptoms:

- When you load a nib file into your application, it can't find the model you have in your framework.
- When you save changes to a model file in your framework, the change isn't reflected in open nib files.

Enterprise Objects Framework 2.x finds models for an application by looking for all models in the application and in any frameworks used by the application. By default Enterprise Objects Framework looks for referenced framework in their *installation* location. Thus if you don't have the framework that contains the model installed, Enterprise Objects Framework won't find the model. Similarly, a change saved to a model in the framework project isn't noticed by Interface Builder since it's looking at the *installed* version, not the source version.

Workaround: You can tell Enterprise Objects Framework to look at the source for your framework projects by using the following defaults command (executed in a shell):

```
defaults write NSGlobalDomain EOProjectSourceSearchPath "($  
(HOME)/myProjectsDirectory1, /myOtherProjectsDirectory)"
```

Then, when Interface Builder or EOModeler is looking for models contained in one of your frameworks, it will first search all project directories within **\$(HOME)/myProjectsDirectory1** and **/myOtherProjectsDirectory** before searching for the built versions.

Alternatively, if you update a model in a framework, reinstall the framework (via **make install**), restart Interface Builder, and reopen the application nib file.

Reference: 68604

Problem: Repeated false alert that Enterprise Objects Framework 1x nib file is being converted to Enterprise Objects Framework 2.0.

Description: When you reopen a nib file after conversion you get the message: <sup>a</sup>This Enterprise Objects Framework 1x nib file has been translated to Enterprise Objects Framework 2.0. If you did not intend to convert to Enterprise Objects Framework 2.x, unload EOPalette, quit Interface Builder without saving this file, and reopen the nib.<sup>o</sup>

Workaround: Quit Interface Builder, then reopen the nib file.

Reference: 70847

Problem: After disconnecting all EOColumnAssociations from a table view, you can't reconnect the table view columns to a different EODisplayGroup.

Description: Suppose you form EOColumnAssociations between all of a table view's columns and an EODisplayGroup, and suppose you then disconnect them all. Subsequent attempts to connect a column to a different display group are not allowed. (All of the display group's properties are grayed out in Interface Builder's inspector.)

The reason for this is as follows. When the first EOColumnAssociation for a table view is created, an EOTableViewAssociation is also created and connected to the table view itself. The EOTableViewAssociation remains connected to the original display group even after all the columns have been disconnected, preventing columns from being connected to a different display group.

Workaround: To make the new connection, click on the table view and disconnect the EOTableViewAssociation.

Reference: 73688

Problem: EOPalette finds the wrong models for projects contained in another project's directory.

Description: When you open a nib whose project is in a subdirectory of another project, Interface Builder uses the model from the parent directory, instead of the model from the nib's own project.

Workaround: Do not store projects directly inside another project's directory. Instead, put at least one level of directories between the projects. For example, instead of putting "test" projects directly in a master project's directory, put them in a subdirectory called "Tests".

Reference: 74947

Problem: If you drag a symbolic link for a model into Interface Builder and answer YES to the "add Model to project" panel, Interface Builder crashes.

Workaround: Drag the actual model file instead of the symbolic link.

Reference: 76591

Problem: EOPalette tries to load model files twice in aggregate projects.

Description: Suppose you have an aggregate project that contains a framework for business logic (and associated model files) and that also contains an application that uses the business logic framework. If you try to open the



application's nib file, Interface Builder tries to add the model file to the EOModelGroup twice, raising an exception. This occurs because Interface Builder finds the model in the aggregate project and then finds it again in the framework linked by the application (which is part of the aggregate).

Workaround: Don't keep your application and framework projects in the same aggregate project.

Reference: 76892

Problem: Multiple fetch-on-load display groups associated with an invalid connection dictionary can cause an exception.

Description: When loading a nib file that has more than one display group set to fetch on load, an application can crash if the display groups' model has an invalid connection dictionary. A crash occurs because the second display group tries to run the adaptor's login panel while the first one already has the panel open. This confuses the Application Kit.

Workaround: Validate the connection dictionaries for all models in the `[EOModelGroup defaultGroup]` before opening the nib.

## Miscellaneous Framework

These problems exist in this Enterprise Objects Framework release:

Reference: 46679

Problem: Enterprise Objects Framework's private instance variables aren't declared **@private** and don't have names that begin with the underbar ('\_') character.

Description: All instance variables in Enterprise Objects Framework should be considered

private.

Workaround: Don't directly access Enterprise Objects Framework instance variables.

Reference: 69211

Problem: Link errors on Windows NT.

Description: Programs on Windows NT must add explicit references to at least one class in each framework in order to avoid link errors at run time.

Workaround: Add a function like that in the following code snippet, which refers to classes in each of Enterprise Objects Framework's layers. Though never invoked, it forces the appropriate linking to occur.

```
#ifndef WIN32
#import <EOControl/EOControl.h>
#import <EOAccess/EOAccess.h>
#import <EOInterface/EOInterface.h>

void _referenceAllEOFrameworks()
{
    [EODisplayGroup new];    // EOInterface
    [EOEntity new];          // EOAccess
    [EOEditingContext new];   // EOControl
}
#endif
```

If you create your project with the type "EOApplication," this code is automatically added to your project main file.

Reference: 72027

Problem: EOKeyValueCoding is broken for doubles on HPUX PDO.

Description: EOKeyValueCoding doesn't work correctly for accessor methods that set and return doubles on HPUX under PDO 4.1 (and previous releases). This means that users cannot correctly fetch enterprise objects that have class properties with accessor methods that use doubles.

Workaround: Change your enterprise objects' accessor methods to return and take as arguments NSNumbers, NSDecimalNumbers, or **ints**.

### **Informix Adaptor**

These problems exist in the Informix adaptor supplied with this release of Enterprise Objects Framework:

Reference: 64031

Problem: Informix adaptor user defaults for Enterprise Objects Framework 2.x are different than those for Enterprise Objects Framework 1.2.

Description: In Enterprise Objects Framework 1.2, the InformixAdaptor stores defaults in the EOFInformixAdaptor domain and uses the following keys: INFORMIXDIR, DBDATE, DBLANG, DBMONEY, InformixLogErrors, ShowSystables, ShowTableOwner, Beautify, DefaultIsolationLevel, GlobalOptimization, GlobalExplain, GlobalLockMode, GlobalPDQPriority, GlobalDataSkip, GlobalConstraints, DatabaseExclusive

Enterprise Objects Framework 2.x stores defaults in the standard NSGlobalDomain and prefixes all keys with <sup>a</sup>Informix<sup>o</sup>, that is: InformixINFORMIXDIR, InformixDBDATE, InformixDBLANG, InformixDBMONEY, InformixLogErrors, InformixShowSystables, InformixShowTableOwner, InformixBeautify, InformixDefaultIsolationLevel, InformixGlobalOptimization, InformixGlobalExplain, InformixGlobalLockMode, InformixGlobalPDQPriority, InformixGlobalDataSkip, InformixGlobalConstraints, InformixDatabaseExclusive.

Workaround: None.

Reference: 70232

Problem: Informix adaptor raises an exception when you try to sort on attributes that are not in the select list.

Description: Due to a restriction in the Informix adaptor, it's not possible to sort on attributes that aren't included in the select list. This means that it isn't possible to sort the results using an attribute that is not marked on the entity as either a primary key, used for locking, or a class property.

Workaround: If possible, add the attribute to the entity and mark it as used for locking. Otherwise, there is no workaround.

### **Oracle Adaptor**

These problems exist in the Oracle adaptor supplied with this release of Enterprise Objects Framework:

Reference: 78072

Problem: Using the "Set Adaptor Info" menu item in EOModeler to change the connection string results in a model that can't connect to the Oracle database.

Description: If you have a connection dictionary that includes both the "serverID" and "hostMachine" keys and you use the "Set Adaptor Info" menu item to set the "serverID" value to a something new, the old hostMachine key remains in your connection dictionary. Unfortunately, this remnant will lead the

OracleEOAdaptor to construct a connection string of the form "T:<hostMachine>:<serverID>" rather than just using "<serverID>".

This is likely to happen when you configure the examples to use a local database.

Workaround: You can remove the "hostMachine" key from the connection dictionary using the EOModeler ConnectionDictionary Inspector. Select the root EOModel, and display the Inspector. You can then select the row with the "hostMachine" key and click "Remove" to clear it from the connection dictionary.

Reference: 77517

Problem: If you try to update an enterprise object with an attribute that maps to a LONG RAW column while using on-demand locking, you get the following exception:

```
"fetchObject -- EODatabaseChannel 0x12345678: attempt  
to  
lock object that has out of date snapshot"
```

Description: It appears that the Oracle database sometimes returns wrong BLOB values when it's passed "SELECT ... <long-raw-column,... FOR UPDATE". So when the EODatabaseChannel attempts to acquire a lock on the row, the results of the SELECT don't match the results that were gotten on the last SELECT (without the FOR UPDATE clause).

Workaround: None. You can't use on-demand locking if you're going to be updating tables with LONG RAW columns.

Reference: 77366

Problem: If you're using pessimistic locking in combination with batch faulting or prefetching of relationships, you'll get the error "ORA-01786: FOR UPDATE of this query expression is not allowed" when you run your application.

Description: When you use pessimistic locking mode in combination with batch faulting or prefetching of relationships, Enterprise Objects Framework generates a SQL statement like the following:

```
<OracleSQLExpression: "SELECT DISTINCT t0.CATEGORY, t0.DATE_RELEASED, t0.LANGUAGE,
t0.MOVIE_ID, t0.RATING, t0.REVENUE, t0.STUDIO_ID, t0.TITLE FROM DIRECTOR t1, MOVIE t0 WHERE
t1.TALENT_ID = :talentID AND t0.MOVIE_ID = t1.MOVIE_ID FOR UPDATE" withBindings:{talentID =
87; }>
```

This statement fails with the ORA-01786 error. The Oracle RDBMS doesn't support the use of DISTINCT with the FOR UPDATE clause.

Workaround: If you use the pessimistic locking mode against an Oracle database, you can't use the batch faulting or prefetching features.

Reference: 62425

Problem: Oracle Adaptor doesn't read stored procedures inside of packages.

Description: There is no way to get the database to tell you the components (procedures and functions) that are inside a package definition. Clients can still create stored procedures in the model that will call into packages, it's just that model description using EOModeler won't create these at connect time.

Workaround: You can use EOModeler to create the stored procedure definitions in the model. Just set the external name of the stored procedure to *package-name.procedure-name*.

Reference: 63348

Problem: Converted models have no width information→Oracle fetches empty strings.

Description: The Enterprise Objects Framework 2.x Oracle Adaptor has problems if you attempt to use an eomodel file from the Enterprise Objects Framework 1.0 release. These model files don't contain width information for string values, which results in all the strings being fetched with 0 length. If this happens for the column that is the primary key, you will also have problems with uniquing since every row will appear to have the same key.

Workaround: Use EOModeler to add width information for VARCHAR2 and RAW columns.

Reference: 73333

Problem: The Oracle adaptor has character set problems on Sparc and m68k.

Description: The Oracle adaptor on Mach does not properly support certain character set conversion on certain architectures:

- On SPARC ISO Latin1 works, but Japanese character sets do not.
- On m68k non-ASCII ISO Latin1 characters are stripped and Japanese character sets don't work at all.

On Intel all character sets work fine, including European and Japanese.

Workaround: To get non-ASCII character sets on SPARC and m68k, install the **EO2JOracleAdaptor.pkg** that's on the Enterprise Objects Framework 2.1 Mach CD.

**Note:** The EO2J Oracle Adaptor is built with the 7.0 version of the Oracle client libraries. Consequently, returning fetch sets from stored procedures is not supported.

### **Sybase Adaptor**

These problems exist in the Sybase adaptor supplied with this release of Enterprise Objects Framework:

Reference: 62634

Problem: The Sybase adaptor uses **CS\_CONVERT()** to convert numeric data into NSDecimalNumbers.

Description: Fetching decimal numbers in locales that use a characters other than `.` for the decimal may not work.

Workaround: None.

Reference: 63169

Problem: Models created for version 4.9 Sybase servers do not include stored procedures.

Description: EOModeler gets stored procedure information using functionality that isn't provided in old servers.

Workaround: Use EOModeler's Stored Procedure Editor to add EOStoredProcedure objects to a model.

Reference: 75024



Problem: The SQL generation for primary key support in the Sybase adaptor isn't sufficient.

Description: The SQL generated by EOModeler to support primary key generation in Sybase simply invokes the **sp\_primarykey** stored procedure. **sp\_primarykey** adds useful information to the syskeys table, but doesn't create constraints that enforce uniqueness (or NOT NULL) in the primary key columns.

Workaround: Add the constraints yourself using statements of the following form:

```
alter table <table-name> add constraint primary key (<column-name> {, <column-name> })
```

## EOModeler

These problems exist in EOModeler with this release of Enterprise Objects Framework:

Reference: 78497

Problem: If you try to save a 1.x **.eomodel** file in 2.1, you get the following error:

```
Apr 18 16:50:23 EOModeler[377] Unable to write file ~/projects/foo.eomodel/DiagramLayout
```

Description: In Enterprise Objects Framework 1.1, the model file was stored in one large property list file. When model files were put into a directory structure for 2.0, EOModeler was designed to open both **.eomodel** and **.eomodeld** model files. In 2.x releases EOModeler writes out any **.eomodel** file as one file.

The problem is that when you save models, the code in the Diagram View that saves layout information attempts to write into an **.eomodeld** directory. If it doesn't find one, it raises an exception.

This error can also occur if you perform a Save As operation and actually enter a name that ends in **.eomodel**.

The good news is that the Save of the file actually completes successfully, so any changes you made will not be lost. EOModeler is left in a potentially unstable state, but you can safely quit and restart it and the model will be fine.

Workaround: If you encounter this error, quit EOModeler and reopen the file. Choose "Save As" and provide a file name that ends in an **.eomodel** extension.

Reference: 78222

Problem: In diagram mode in EOModeler, dragging around an entity that is being edited can lead to strange behavior and exceptions.

Workaround: Finish editing before moving the entity.

Reference: 77880

Problem: EOModeler exhibits odd behavior if you try to paste model objects from a closed window.

Description: Copy a model object (or objects), close the window from which you copied, and then open a new window. The Paste menu item is enabled in the new window, but if you try to paste, nothing happens.

Workaround: Don't close the original window before pasting the model objects.

Reference: 77608

**Problem:** Existing Enterprise Objects Framework projects don't see the new Enterprise Objects Framework 2.1 source code generation templates (used when you issue the "Generate Source Files..." command in EOModeler).

**Description:** Enterprise Objects Framework 2.1 includes fixes and additions to EOInterfaceFile.template and EOImplementationFile.template. If a user has a project from Enterprise Objects Framework 2.0 that contains the old versions of these files, the old versions will continue to be used.

**Workaround:** If you didn't customize the templates, then you should just remove the old files from your project.

If you want to build a customized template based on the newest versions, copy /NextDeveloper/Apps/EOModeler.app/Resources/EOImplementationFile.template and /NextDeveloper/Apps/EOModeler.app/Resources/EOInterfaceFile.template into your project directory.

**Reference:** 77603

**Problem:** When you create a subclass in your model and generate source files for it, the resulting header file won't compile.

**Description:** For example, suppose you create a Salesman subclass of Employee and generate source files for it. The header for Salesman correctly inherits from Employee, but it doesn't include the **Employee.h** header, so it can't be compiled.

```
// Salesman.h
//
// Created on Mon Mar 10 10:19:41 PST 1997 by NeXT EOModeler Version 304

#import <EOControl/EOControl.h>
```

```
@class Employee;  
@class Department;  
  
@interface Salesman : Employee <NSCoding>  
{  
}  
  
@end
```

If you attempt to compile this code, you'll get this error:

Salesman.h:12: Cannot find interface declaration for `Employee', superclass of `Salesman'  
gnumake: \*\*\* [Salesman.o] Error 1

Workaround: **Change**

```
    @class Employee  
to  
    #import "Employee.h"
```

Reference: **77134**

Problem: If you open two different Diagram View windows for the same model file, the layout for only one of the windows is saved, and which layout is saved is indeterminate.

Workaround: Only open one Diagram View window for a model at a time.

Reference: **77191**

Problem: Pasted attributes do not retain their primary key, used for locking, or class property settings.

Description: If you copy and paste an attribute, the pasted attribute's primary key, used for locking, and class property bits revert to their default settings, regardless of

their state in the original. This occurs because the settings are actually maintained by the owning entity, which isn't incorporated into an attribute's pasteboard rendering.

Workaround: Manually set the bits after pasting.

Reference: 51250

Problem: You can't join on derived attributes.

Description: Invalid SQL is generated whenever a derived attribute is designated as a join attribute in a relationship.

Workaround: None.

Reference: 75801

Problem: Editing nested dictionaries with the UserInfo or ConnectionDictionary editors doesn't work.

Description: Suppose that a model has a userInfo or connection dictionary that contains another dictionary. If you use EOModeler's UserInfo or Connection Dictionary Inspector to edit the enclosed object, EOModeler converts it to a string. For example, if the userInfo dictionary contains the following dictionary:

```
myValue = {  
    hello = world;  
}
```

and you convert "hello" and "world" to uppercase, EOModeler saves the following string to the model file:

```
myValue = "{\n    HELLO = WORLD;\n}"
```

Similarly, EOModeler converts arrays to strings.

Workaround: If you need to store such compound data structures in your userInfo or connectionDictionary, you will need to edit it by hand using Edit, or write a your own custom Inspector.

## Wizard

These problems exist in the Wizard with this release of Enterprise Objects Framework:

Reference: 77987

Problem: The Wizard for creating new EOApplications can't use models that are accessed through symbolic links

Description: In Project Builder, if you attempt to create a new EOApplication that requires an EOModel and you specify a symbolic link to an EOModel, you will get the following error:

```
"write: Cannot create directory <project-location>/<model-name>"
```

Workaround: Specify a path that doesn't include a symbolic link.

## On-line Examples

These problems exist in the on-line examples included in this release of Enterprise Objects Framework:

Reference: 78215

Problem: The UsingStoredProcedures example doesn't work for Sybase or Oracle

Description: For Oracle, the insertMovie and deleteMovie stored procedures have arguments with bad columnNames. For Sybase, the nextPrimaryKey and fetchMovieWithPrimaryKey stored procedures have arguments with bad column names.

Workaround: You can either change the names of the arguments in the stored procedure definition (by editing **oraclesp.sql** or **sybasesp.sql**), or you can change the columnName of the problematic arguments. Here is a detailed list of the changes you need to make in EOModeler to the columnNames to make the EOModel stored procedures match those in the database:

```
OracleStoredProcedures.eomodeld/deleteMovie.storedProcedure
    movieID
        AMOVIE_ID => MOVIE_ID
OracleStoredProcedures.eomodeld/insert.storedProcedure
    category
        ACATEGORY => CATEGORY
    dateReleased
        ADATE_RELEASED => DATERELEASED
    language
        ALANGUAGE => LANGUAGE
    movieID
        AMOVIE_ID => MOVIE_ID
    rating
        ARATING => RATING
    revenue
        AREVENUE => REVENUE
    studiold
        ASTUDIO_ID => STUDIO_ID
    title
        ATITLE => TITLE
SybaseStoredProcedures.eomodeld/fetchMovieWithPrimaryKey.storedProcedure
    name
        MOVIE_ID => movie_id
SybaseStoredProcedures.eomodeld/nextMoviePrimaryKey.storedProcedure
    name
        MOVIE_ID => A_MOVIE_ID
```

Reference: 77990

Problem: Sometimes when you run the **install\_database** script or use the schema object creation window in EOModeler, you might get one of the following errors.

```
Mar 27 10:19:34 eoutil[1577] Exception running dump: ORA-00955: name is already used by an existing object
```

```
create table   !!! eo_temp_table as select max(MOVIE_ID) counter from MOVIE
```

```
Mar 27 10:27:54 eoutil[1591] Exception running dump: ORA-00955: name is already used by an existing object
```

```
create procedure   !!! eo_set_sequence is
```

```
    xxx number;
```

```
    yyy number;
```

```
begin
```

```
    select max(counter) into xxx from eo_temp_table;
```

```
    if xxx is not NULL then
```

```
        yyy := 0;
```

```
        while (yyy < xxx) loop
```

```
            select MOVIE_SEQ.nextval into yyy from dual;
```

```
        end loop;
```

```
    end if;
```

```
end;
```

Description: If the primary key support code dies unexpectedly, it can leave the **eo\_set\_sequence** stored procedure and the **eo\_temp\_table** table in the user's schema. This isn't a problem right away, but any subsequent attempt to run the **install\_database** script will fail because the statements that attempt to create these schema objects will fail and the data loading will stop before it is finished.

Workaround: Use the following commands from SQLPlus to remove these objects before you attempt to create the schema objects with the **install\_database** script.

```
SQL> drop table eo_temp_table;
```

```
SQL> drop procedure eo_set_sequence;
```



Reference: 64283

Problem: Enterprise Objects Framework 2.x examples merge with obsolete Enterprise Objects Framework 1.1 examples.

Description: If a system has Enterprise Objects Framework 1.1 Developer installed on it (EODeveloper.pkg) and then upgrades to 4.0, the EODeveloper package (incorrectly) stays installed, and the examples stay in **/NextDeveloper/Examples/EnterpriseObjects**.

When the user later installs EO2Developer, the Enterprise Objects Framework 2.x examples are merged with the Enterprise Objects Framework 1.1 example files already in **/NextDeveloper/Examples/EnterpriseObjects**.

Workaround: Uninstall 1.1 EODeveloper.pkg before installing EO2Developer.pkg.

Reference: 65165

Problem: **install\_database** doesn't work: Sybase: Can't find type 'decimal'.

Description: The **install\_database** script in the **EnterpriseObjects/DatabaseSetUp** directory doesn't work for Sybase 4.9 servers.

When the models in the examples directory were converted to Sybase, the server version wasn't taken into account. Some of the attributes were converted to 'decimal' data type and the Sybase 4.x server doesn't know about these.

Workaround: Use EOModeler to find and change all the external types from decimal to float in each of the models in your examples directory.

Reference: 68231

Problem: **make all** in the Enterprise Objects Framework examples fails sometimes.

Description: The Enterprise Objects Framework examples will fail to make if **\$ {NEXT\_ROOT}/LocalDeveloper** is present on your system and you don't have write access to it. The makefile tries to install the BusinessLogic framework in the **/LocalDeveloper/Frameworks** directory.

Workaround: Execute **make** as root or ask your system administrator to create a **/LocalDeveloper/Frameworks** directory and make it writable by you.

Reference: 70557

Problem: The Customer.app example doesn't propagate changes to CreditCard objects.

Description: The relationship between CreditCard and Member is not modeled correctly in the examples. CreditCard's primary key is composed of its **cardNumber** and **cardType** attributes. In the Customer application, you can modify the **cardNumber** and **cardType** of a Member's CreditCard, but these changes aren't propagated to the objects to which the CreditCard is related. That is, after saving the changes to the database, the CreditCard's **cardNumber** and **cardType** are updated, but foreign keys in the MEMBER table still have the old values.

CreditCard should have a separate primary key identifier, such as **cardID**, on which the CreditCard-Member relationships are based. This approach would keep CreditCards and Members in sync, and it would also be a more efficient implementation.

Workaround: **None.**

Reference: 72246

Problem: The ODBC adaptor source code provided as an example is only available on OpenStep for Windows NT.

Description: The source code for the ODBC adaptor is available on Windows NT as an example. It is not, however, included on Mach or PDO.

Workaround: Copy the source code from an NT machine.

**Note:** If you just want the ODBC adaptor to see an example of a concrete adaptor, you can look at the FlatFile adaptor that's included in all releases.

Reference: 76020

Problem: In the Studios.app, buying the movies associated with a Talent doesn't update the Movies' Studios.

Description: The Studio class's **buyAllMoviesStarring:** method doesn't take advantage of Enterprise Objects Framework's EOKeyRelationshipManipulation methods. Instead, it simply invokes [self addToMovies:...].

Workaround: Change **Studio.m's buyAllMoviesStarring:** implementation to the following:

```
- (void)buyAllMoviesStarring:(Talent *)talent
{
    NSEnumerator *movieEnumerator = [[talent moviesStarredIn] objectEnumerator];
    id movie;

    while ( (movie = [movieEnumerator nextObject]))
        [self addObject:movie toBothSidesOfRelationshipWithKey:@"movies"];
```

}

Reference: 69419

Problem: Example data for Movie and Rental models must be installed in different databases.

Description: When installing data for more than one model in a database, the **eo\_sequence\_table** is only properly initialized for the first model installed.

Workaround: You can either install the data for each model in a different database or change the last line in the **install\_database** script located in the DatabaseSetUp directory of your examples directory as follows:

**Change this:**

```
${EOUTIL} dump ../BusinessLogic/Rentals.eomodeld -source plist  
database -schemaCreate -postInstall < RentalData.plist
```

**To this:**

```
${EOUTIL} dump ../BusinessLogic/Rentals.eomodeld -source plist  
database -schemaCreate -postInstall -force < RentalData.plist
```

When you use the **-force** option as a workaround, exceptions are logged because the **eo\_sequence\_table** table and **eo\_pk\_for\_table** procedure already exist when you execute **install\_databases**. You can just ignore these exceptions.

## Documentation

This problem exists in the documentation included in this release of Enterprise Objects Framework:

Reference: 77170

Problem: Enterprise Objects Framework Developer's Guide sections about ordering database operations are out of date.

Description: The sections "How Do I Order Database Operations" and "Constraints for Enforcing Relational Integrity Rules" in the chapter "Answers to Common Design Questions" in the Enterprise Objects Framework Developer's Guide need to be updated to reflect a new operation ordering algorithm.

The new algorithm works as follows:

- If an entity (Movie) has a to-one relationship to a second entity (Studio) and the inverse relationship is a to-many, then the second entity (Studio) is considered the *master*.
- If an entity (Talent) has a to-one relationship to a second entity (TalentPhoto) and the inverse relationship is also a to-one, then the entities are peers.

Enterprise Objects Framework builds up an entity ordering based on the "master-ness" of the entities. This ordering is built up dynamically and is enhanced as your application touches more and more entities.

Before sending operations to the database, Enterprise Objects Framework orders the operations based on the entity ordering. For inserts, Enterprise Objects Framework orders master entities first; whereas for deletes, Enterprise Objects Framework orders master entities last. Order on updates doesn't generally matter, Enterprise Objects Framework orders masters first.

This new algorithm should reduce the number of scenarios in which you have to reorder operations. However, if your database has sophisticated referential integrity, if it uses triggers, or there are referential integrity constraints that are not modeled as

EORelationships, you may still need to reorder adaptor operations programmatically.