

There has been substantial interest in the difficult task of finding and fixing leaks in EOF, OpenStep and WOF applications. As the EOF team was preparing for the final release of WebObjects 3.5, they wrote a small test app to help them find leaks in the framework. This example will help you learn how to write an application that should be able to run indefinitely without running out of memory. Also, if you think you have encountered a serious leak in the framework, you can use this app as a minimal test case for illuminating the problem.

The app takes a few input arguments:

- batchsize m [number, default 1]
- runs n [number, default 1]
- trials t [number, default 1]
- hang yn [YES/NO, default NO]

For each trial, at runtime the app will, using the Movies EOmodel:

- Empty the tables

- Insert movies (insert  $m * n$  movies)
- Add directors and actors ( $m*n$  objects fetched,  $m*n$  to-one faults fired,  $m*n$  to-many faults fired,  $m*n+m^2*n$  objects inserted)
- Modify objects ( $m*n$  objects fetched,  $m*n$  to-one faults fired,  $m*n$  to-many faults fired,  $m*n+m^2*n$  objects faulted in,  $m*n$  objects deleted,  $m*n$  objects inserted,  $m*n+m^2*n$  objects mutated)

The -hang option makes the app pause before quitting. This allows you to use pview or ps after the app finishes, rather than monitoring the app during its entire run.

The following results were obtained on Windows NT, using pview.exe to examine the image at the end of the run. This app has only been tested using EOF 2.2, the EOF version that ships with WebObjects 3.5. You might notice that the results aren't quite as deterministic as one might hope, but this is the raw untouched data we got.

Oracle...

Batch	Runs	Trials	Virtual-Size
6 4	1	27980	
6 4	10	27980	

6	4	25	27980
10	10	10	27980
--			
10	10	100	28740
10	10	10	28740

Sybase...

Batch	Runs		Trials	Virtual-Size
6	4	10	32860	
6	4	1	32792	
6	4	25	32860	
--				
10	10	1	33520	
10	10	10	33588	
10	10	100	33588	

Informix...

Batch		Runs		Trials	Virtual-Size
10	10	10		25492	
10	10	1		23444	
6	4	10		23444	
6	4	25		23444	
6	4	5		23460	
--					
10	10	10		25508	
10	10	1		23460	
--					
6	4	100		25508	
10	10	(37+)		30352	[This test was aborted after 37 trials due to server failure.]

Here is the source code. You'll need to drop the Movies EOmodel into the app before compiling.

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

```
#import "Movie.h"
#import "Actor.h"
#import "Director.h"

void usage(void)
{
    NSLog(@"Usage: LeakTester -batchsize <batchsize> -runs <number of
runs> -trials <number of trials> -hang <BOOL>
WARNING!!!! THIS PROGRAM
REMOVES ALL EXISTING DATA IN THE TABLES USED");
}

void logReport(NSString *modelPath, int batchsize, int runs, int trials)
{
    NSAutoreleasePool *pool = [[NSAutoreleasePool alloc] init];
    EOModel *model;

    model = [[[EOModel alloc] initWithContentsOfFile:modelPath]
autorelease];
```

```
    NSLog(@"%@, batchsize = %i, runs = %i, trials = %i", [model  
adaptorName], batchsize, runs, trials);
```

```
    [pool release];
```

```
}
```

```
void clearTables(NSString *modelPath)
```

```
{
```

```
    NSAutoreleasePool *pool = [[NSAutoreleasePool alloc] init];
```

```
    EOModel *model;
```

```
    EOAdaptor *adaptor;
```

```
    EOAdaptorContext *context;
```

```
    EOAdaptorChannel *channel;
```

```
    EOSQLExpression *expression;
```

```
    model = [[[EOModel alloc] initWithContentsOfFile:modelPath]
```

```
autorelease];
    adaptor = [EOAdaptor adaptorWithModel:model];
    context = [adaptor createAdaptorContext];
    channel = [context createAdaptorChannel];

    NSLog(@"Dumping old data from tables...");

    [channel openChannel];

    expression = [[adaptor expressionClass] expressionForString:@"delete
from LKMOVIE"];
    [channel evaluateExpression:expression];

    expression = [[adaptor expressionClass] expressionForString:@"delete
from LKACTOR"];
    [channel evaluateExpression:expression];

    expression = [[adaptor expressionClass] expressionForString:@"delete
from LKDIRECTOR"];
```

```
[channel evaluateExpression:expression];
```

```
// Informix complains if we do a COMMIT here; OK for Sybase and  
Oracle.
```

```
// expression = [[adaptor expressionClass]  
expressionForString:@"commit"];  
// [channel evaluateExpression:expression];
```

```
[channel closeChannel];
```

```
[pool release];
```

```
}
```

```
void doit(NSString *modelPath, int batchsize, int runs)
```

```
{
```

```
    NSAutoreleasePool *pool, *pool0, *pool1, *pool2;
```

```
    EOEditingContext *editingContext = [EOEditingContext new]; // context  
    with defaultCoordinator
```



```
Movie *movie;  
Movie *lastMovie;  
NSArray *movies;  
NSArray *actors;  
NSArray *lastActors;  
Actor *actor;  
Director *director;  
int i;  
int j;  
int priKey;  
int m;
```

```
// For apps with models stored in the main bundle, the following is  
// unnecessary, since the defaultGroup will be automatically built
```

from

```
// the contents of the main bundle and any referenced frameworks.  
pool = [[NSAutoreleasePool alloc] init];  
[EOModelGroup setDefaultGroup:[[EOModelGroup new] autorelease]];  
[[EOModelGroup defaultManager] addModelWithFile:modelPath];
```

```
pool0 = [[NSAutoreleasePool alloc] init];
// Create and insert root set
NSLog(@"Inserting root set...");
priKey = 0;
for ( i = 0 ; i < runs ; i++ ) {
    pool1 = [[NSAutoreleasePool alloc] init];
    for ( j = 0 ; j < batchsize ; j++ ) {
        priKey++;
        movie = [[[Movie alloc] init] autorelease];
        [movie setMovieId:[NSDecimalNumber
decimalNumberWithMantissa:priKey exponent:0 isNegative:NO]];
        [movie setTitle:[NSString
stringWithFormat:@"Movie_%i",priKey]];
        [editingContext insertObject:movie];
    }
    [editingContext saveChanges];
    [[editingContext rootObjectStore] invalidateAllObjects];
    [[editingContext undoManager] forgetAll];
}
```

```
    [pool1 release];  
}  
[pool0 release];
```

```
priKey = 0;  
for ( i = 0 ; i < runs ; i++ ) {  
    pool1 = [[NSAutoreleasePool alloc] init];  
    pool2 = [[NSAutoreleasePool alloc] init];  
    NSLog(@"Fetching Movies batch and adding objects for run
```

```
#%i...",i);
```

```
    movies = [editingContext  
objectsWithFetchSpecification:[EOFetchSpecification  
fetchSpecificationWithEntityName:@"Movie" qualifier:[EOQualifier  
qualifierWithQualifierFormat:@"(movielfd > %d) AND (movielfd <= %d)",  
i*batchsize, (i+1)*batchsize] sortOrderings:nil]];
```

```
    for ( j = 0 ; j < [movies count] ; j++ ) {  
        movie = [movies objectAtIndex:j];  
        priKey++;
```

```
director = [[movie director] self]; // Just to tickle the
fault
director = [[Director alloc] init];
[editingContext insertObject:director];
[director setDirectorId:[NSDecimalNumber
decimalNumberWithMantissa:priKey exponent:0 isNegative:NO]];
[director setName:[NSString
stringWithFormat:@"First_%i",priKey]];
[movie addObject:director
toBothSidesOfRelationshipWithKey:@"director"];
[director release];
for ( m = 0 ; m < batchsize ; m++ ) {
    priKey++;
    actor = [[Actor alloc] init];
    [editingContext insertObject:actor];
    [actor setActorId:[NSDecimalNumber
decimalNumberWithMantissa:priKey exponent:0 isNegative:NO]];
    [actor setName:[NSString
stringWithFormat:@"First_%i",priKey]];
```

```
        [movie addObject:actor  
toBothSidesOfRelationshipWithKey:@"actors"];  
        [actor release];  
    }  
}
```

```
NSLog(@"Saving...");  
[editingContext saveChanges];
```

```
[editingContext invalidateAllObjects];  
[[editingContext rootObjectStore] invalidateAllObjects];  
[[editingContext undoManager] forgetAll];  
[pool2 release];  
[pool1 release];  
}
```

```
for ( i = 0 ; i < runs ; i++ ) {  
    pool1 = [[NSAutoreleasePool alloc] init];
```

```
pool2 = [[NSAutoreleasePool alloc] init];
NSLog(@"Fetching Movies batch and mutating for run #%"i...",i);
movies = [editingContext
```

```
objectsFetchSpecification:[EOFetchSpecification
fetchSpecificationWithEntityName:@"Movie" qualifier:[EOQualifier
qualifierWithQualifierFormat:@"(movied > %d) AND (movied <= %d)",
i*batchsize, (i+1)*batchsize] sortOrderings:nil]];
```

```
for ( j = 0 ; j < [movies count] ; j++ ) {
    movie = [movies objectAtIndex:j];
    priKey++;
    director = [[movie director] self]; // Just to tickle the
```

fault

```
    director = [[Director alloc] init];
    [editingContext insertObject:director];
    [director setDirectorId:[NSNumber
decimalNumberWithMantissa:priKey exponent:0 isNegative:NO]];
    [director setName:[NSString
stringWithFormat:@"First_%"i",priKey]];
```

```

    [movie addObject:director
toBothSidesOfRelationshipWithKey:@"director"];
    [director release];
    if ( j % 2 ) {
        actors = [[movie actors] shallowCopy];
        lastMovie = [movies objectAtIndex:j-1];
        lastActors = [[lastMovie actors] shallowCopy];
        for ( m = 0 ; m < [actors count] ; m++ ) {
            [movie removeFromActors:[actors objectAtIndex:m]];
        }
        for ( m = 0 ; m < [lastActors count] ; m++ ) {
            [movie addToActors:[lastActors objectAtIndex:m]];
        }
        for ( m = 0 ; m < [lastActors count] ; m++ ) {
            [lastMovie removeFromActors:[lastActors
objectAtIndex:m]];
        }
        for ( m = 0 ; m < [actors count] ; m++ ) {
            [lastMovie addToActors:[actors objectAtIndex:m]];

```

```
    }  
    [actors release];  
    [lastActors release];  
  }  
}
```

```
NSLog(@"Saving mutations...");  
[editingContext saveChanges];
```

```
[editingContext invalidateAllObjects];  
[[editingContext rootObjectStore] invalidateAllObjects];  
[[editingContext undoManager] forgetAll];  
[pool2 release];  
[pool1 release];
```

```
}  
[EOModelGroup setDefaultGroup: nil];  
[pool release];  
[editingContext release];
```

```
}
```



```
void main (int argc, const char *argv[])
{
    NSAutoreleasePool * pool;
    BOOL hang;
    int batchsize, runs, trials;
    NSString *modelPath;
    int trial;

    pool = [[NSAutoreleasePool alloc] init];
    hang = [[NSUserDefaults standardUserDefaults] boolForKey:@"hang"];
    batchsize = [[NSUserDefaults standardUserDefaults]
integerForKey:@"batchsize"];
    if (!batchsize) batchsize = 1;
    runs = [[NSUserDefaults standardUserDefaults] integerForKey:@"runs"];
    if (!runs) runs = 1;
    trials = [[NSUserDefaults standardUserDefaults]
integerForKey:@"trials"];
```

```
if (!trials) trials = 1;  
modelPath = [[[NSBundle mainBundle] bundlePath]  
stringByAppendingPathComponent: @"LeakTester.eomodeld"];
```

```
logReport(modelPath, batchsize, runs, trials);
```

```
for (trial = 0; trial < trials; trial++) {  
    NSLog(@"Trial %i beginning...", trial);  
    clearTables(modelPath);  
    doit(modelPath, batchsize, runs);  
    NSLog(@"Trial %i done.", trial);  
}
```

```
logReport(modelPath, batchsize, runs, trials);
```

```
[pool release];
```

```
if (hang) {  
    NSLog(@"Hit RETURN to exit...");
```

```
    getchar();  
}
```

```
NSLog(@"Done.");  
exit(0);          // insure the process exit status is 0
```

```
}
```