Filebench

Spencer Shepler
Eric Kustarz
Andrew Wilson
[Richard McDougall]

Filebench Discussion

- Filebench motivation
- Filebench description
- Issues
- What next?

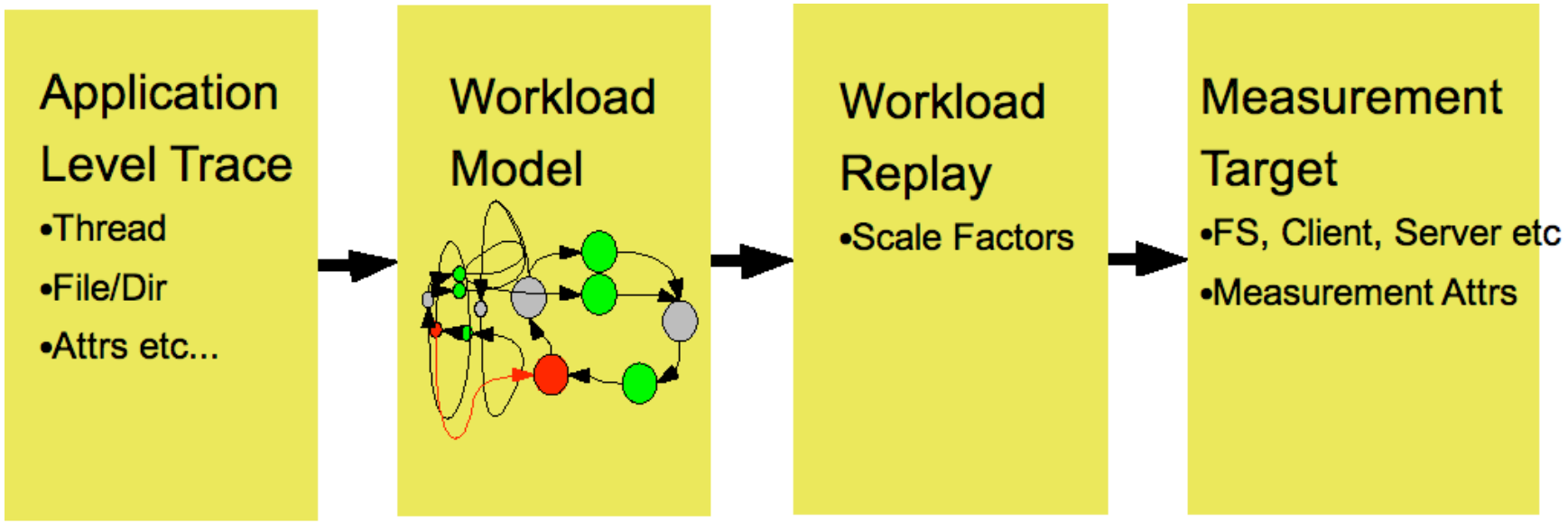
Testing filesystem performance

- Dd
- Tar
- mkfile
- Bonnie
- Iozone
- And on and on…
  - fsstress, ffsb, fsrandom, mongo, iometer

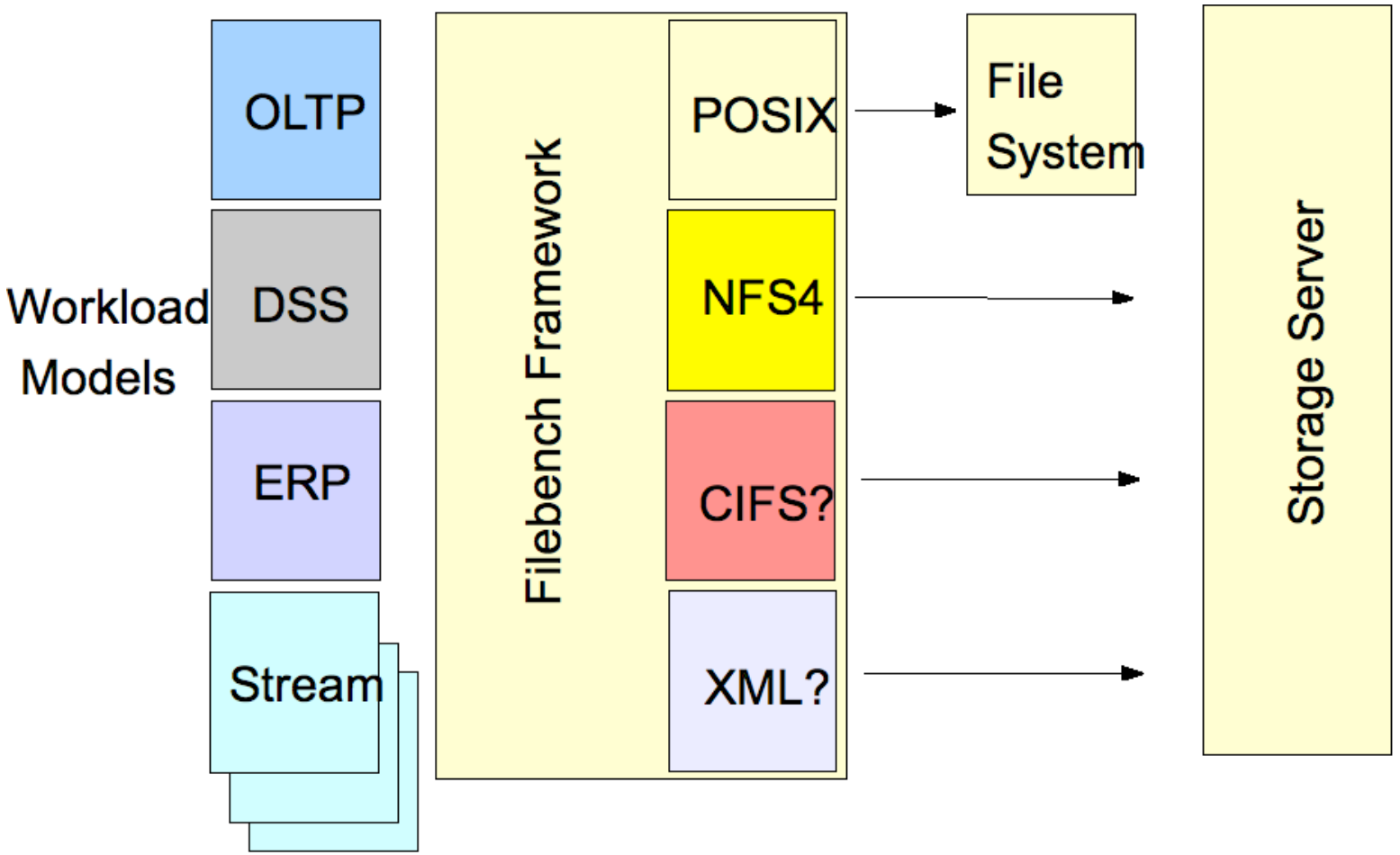
Why invest in a File System Perf Framework?

- Need complete test coverage for file level applications
  - Current coverage is mostly micro benchmarks:
    - bonnie, iozone, mongo
  - Coverage was very limited (less than 10% of important application cases covered)
  - Current approach is to use benchmark full application suites: e.g. Oracle using TPC-C: expensive, labor intensive
  - Up to 100 different benchmarks are required to accurately report on filesystem performance today
- For NFS use, SPECsfs is limited to NFS Version 3

Model based methodology study



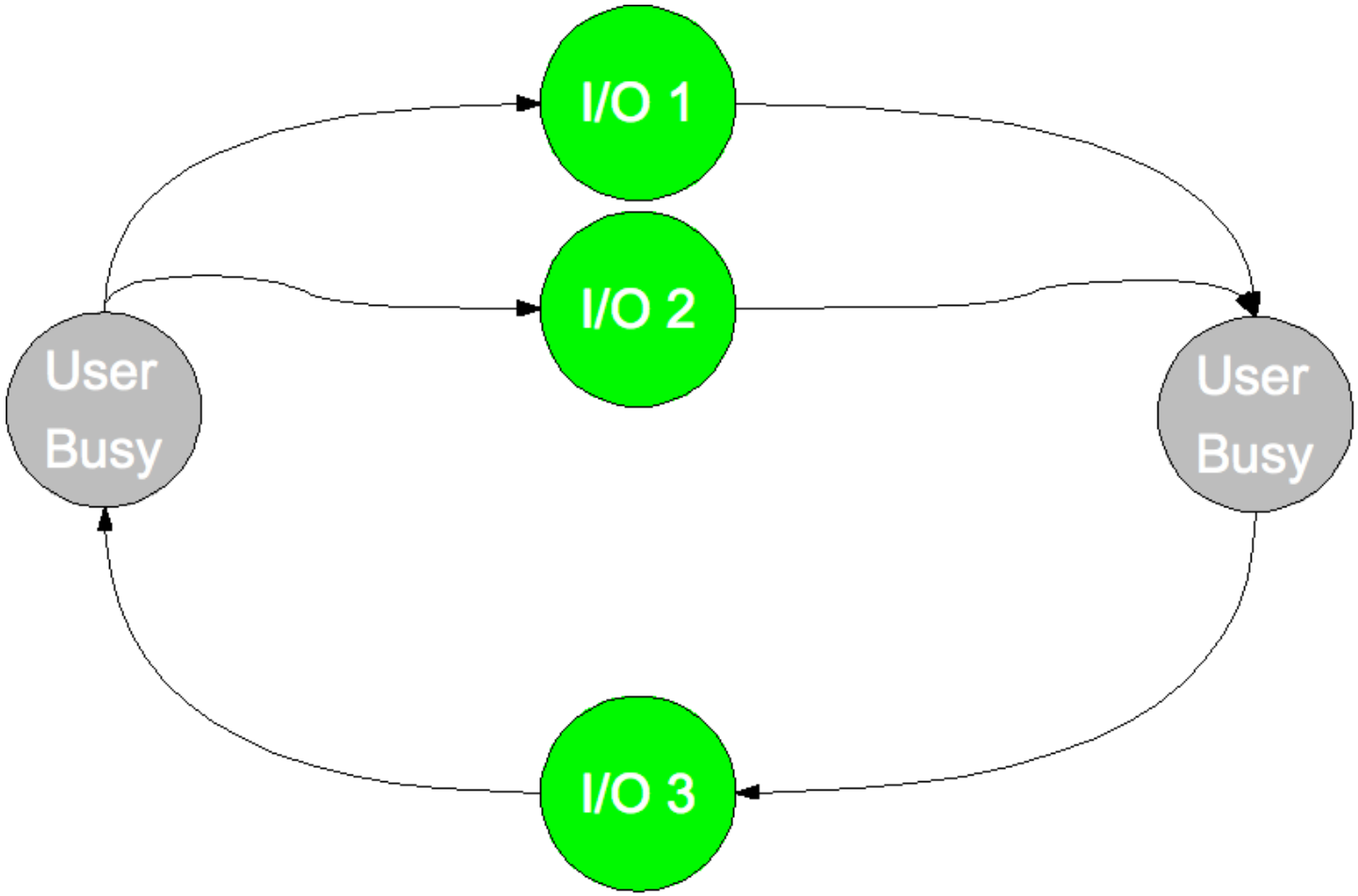
Filebench Architecture



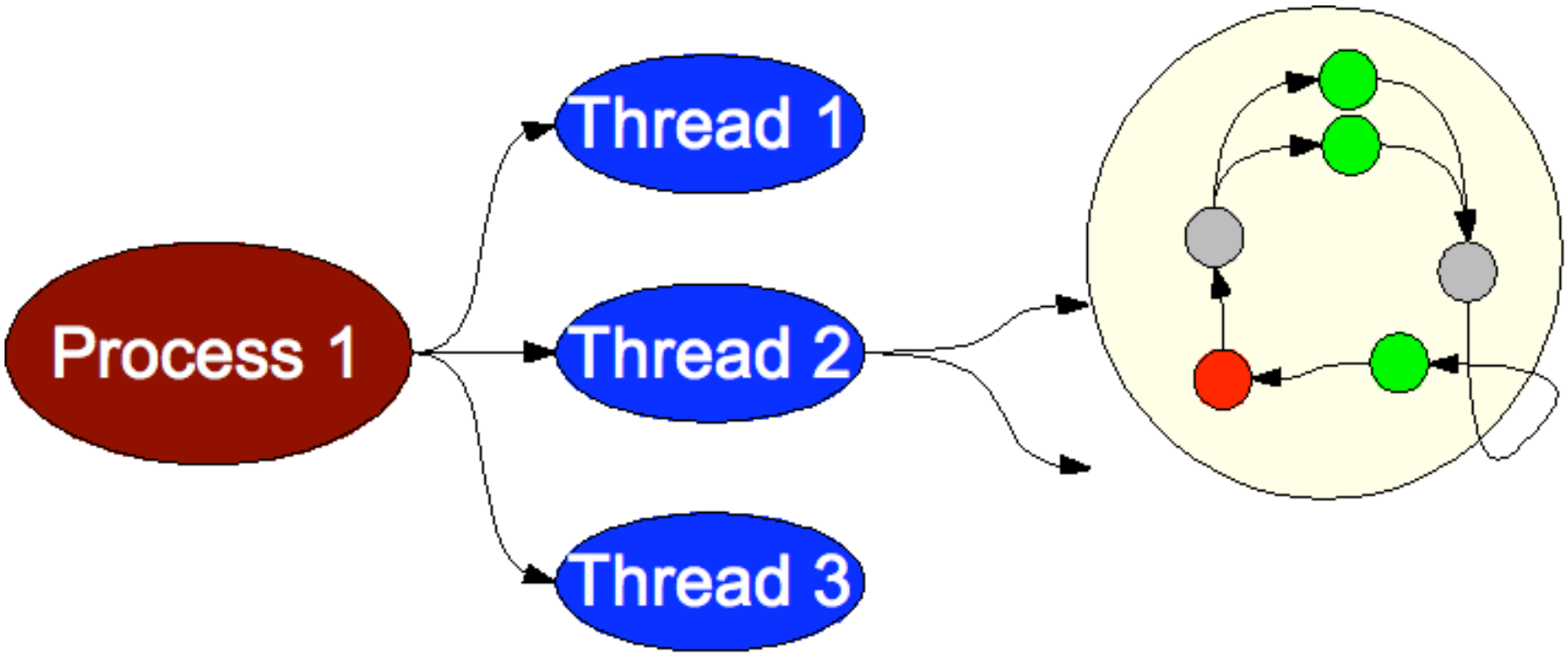
Model Allows Complex/Important Scaling Curves

- For example:
  - Throughput/latency vs. working set size
  - Throughput/latency vs. # of users
  - CPU efficiency vs. throughput
  - Caching efficiency vs. working set size/memsize

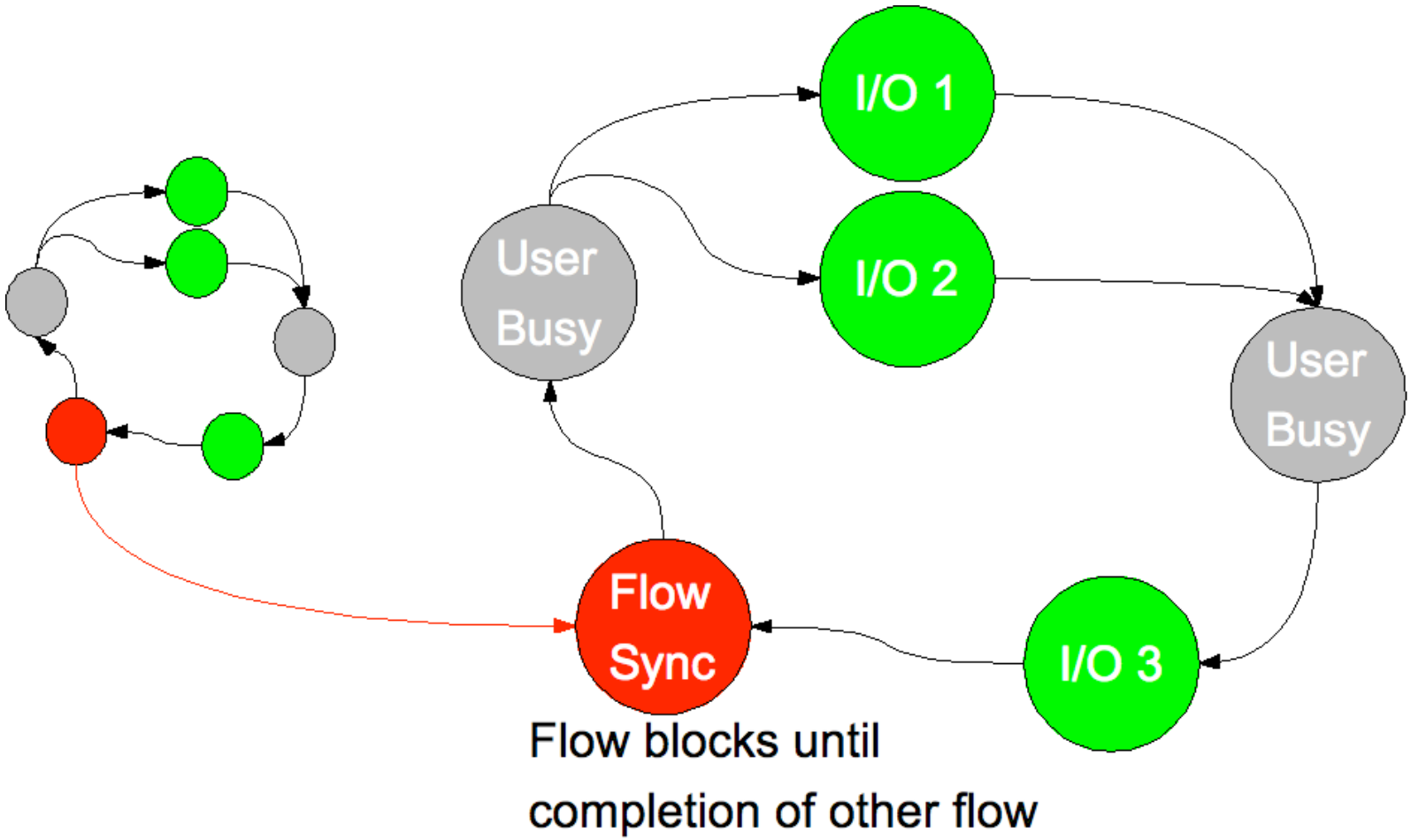
Flow States: Open Ended Flow



Characterize and Simulate via Cascades of Workload Flows:



Flow States: Synchronized Flow



Examples of Per-flow Operations

- Types
  - Read
  - Write
  - Create
  - Delete
  - Append
  - Getattr
  - Setattr
  - Readdir
  - Semaphore block/post
  - Rate limit
  - Throughput limit
- Attributes
  - Sync_data
  - Sync_metadata
  - IO Size
  - IO Pattern, probabilities
  - Working set size
  - Etc.

Simple Random I/O Workload Description

```
define file name=bigfile0,path=$dir,size=$filesize,prealloc,reuse,paralloc

define process name=rand-read,instances=1
{
  thread name=rand-thread,memsize=5m,instances=$nthreads
  {
    flowop read name=rand-read1,filename=bigfile0,iosize=$iosize,random
    flowop eventlimit name=rand-rate
  }
}
```

Filesets

Filesets: a definition of a set of files

- A fractal tree of files
- A fileset has a depth and size, width of directories is computed from these
- Can also have a depth of 1 to make one large directory
- Can have uniform sizes, depths, widths or configured as a [gamma] distribution
- Filesets that mimic file servers typically use gamma distribution for size and depth

Running a single Filebench workload

```
Example varmail run:

filebench> load varmail

 Varmail personality successfully loaded
 Usage: set $dir=<dir>
        set $filesize=<size>    defaults to 16384
        set $nfiles=<value>     defaults to 1000
        set $dirwidth=<value>   defaults to 20
        set $nthreads=<value>   defaults to 1
        set $meaniosize=<value> defaults to 16384
        run <runtime>

filebench> set $dir=/tmp

filebench> run 10

 Fileset mailset: 1000 files, avg dir = 20, avg depth = 2.3,mbytes=15
 Preallocated fileset mailset in 1 seconds
 Starting 1 filereader instances
 Starting 1 filereaderthread threads
 Running for 10 seconds...
 IO Summary: 21272 iops 2126.0 iops/s, (1063/1063 r/w) 32.1mb/s,338us cpu/op, 0.3ms latency
```

OLTP Program - benchmark results
large_db_oltp_2k_cached

```
Flowop totals:
random-rate                0ops/s   0.0mb/s      0.0ms/op        0us/op-cpu
shadow-post-dbwr        9774ops/s   0.0mb/s     18.0ms/op       25us/op-cpu
shadow-post-lg          9775ops/s   0.0mb/s      0.1ms/op        5us/op-cpu
shadowhog               9775ops/s   0.0mb/s      0.1ms/op       27us/op-cpu
shadowread              9792ops/s  19.1mb/s      1.8ms/op       15us/op-cpu
dbwr-aiowait              98ops/s   0.0mb/s     27.8ms/op       93us/op-cpu
dbwr-block                98ops/s   0.0mb/s     64.2ms/op      163us/op-cpu
dbwr-hog                  98ops/s   0.0mb/s      0.0ms/op       16us/op-cpu
dbwrite-a               9774ops/s  19.1mb/s      0.1ms/op        9us/op-cpu
lg-block                   3ops/s   0.0mb/s    320.8ms/op      487us/op-cpu
lg-aiowait                 3ops/s   0.0mb/s      4.4ms/op       24us/op-cpu
lg-write                   3ops/s   0.8mb/s      0.2ms/op       23us/op-cpu

IO Summary:       2376841 ops  19669.9 ops/s,     9792/9777 r/w    39.0mb/s,
                        91uscpu/op
```

Filebench pre-defined workloads

- "File Macro"
  - Small database
  - Large database
  - Multi-threaded web server
  - Multi-threaded proxy server
  - Home directory server
  - NFS mail server
  - DB Mail server
  - Video server
- "File Micro"
  - Sequential read/write
  - Multistream read/write
  - Allocating writes
  - Reallocating writes
  - Random read/write
  - MT random read/write
  - File create/delete
  - File meta-data ops
  - I/O types: O_DSYNC, etc.
  - Directory size scaling

Filebench Features in Development

- Random Variables
- Composite Flowops
- NFS / CIFS Plugins
- Multi-client Framework
- Scalability Issues

Documentation / Discussion

- http://sourceforge.net/projects/filebench/
- http://opensolaris.org/os/community/performance/
- http://www.solarisinternals.com/wiki/index.php/FileBench
- http://www.solarisinternals.com/wiki/index.php/Filebench_for_Programmers
- http://www.solarisinternals.com/wiki/index.php/FileBench_Workload_Language