

# FileBench Basics

presentation version 3.0

**Eric Kustarz**

**Richard McDougall**

**Spencer Shepler**

**Andrew Wilson**

# FileBench Discussion

- FileBench motivation
- FileBench description
- Preparation for inclusion in OpenSolaris
- Issues
- What next?

# Testing File System Performance

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

# Why Invest in a File System Performance Framework?

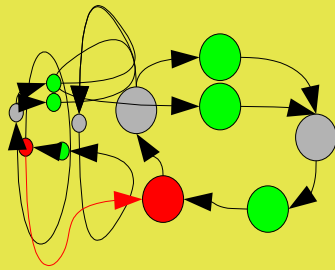
- Need complete test coverage for file level applications
  - Current coverage is mostly micro benchmarks:
    - bonnie, iozone, mongo
  - Coverage was very limited (less than 10% of important application cases covered)
  - Current approach is to use benchmark full application suites: e.g. Oracle using 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
  - NFSv3 workload only represents “home directory servers”
  - upcoming addition of CIFS will be limited in workload

# Model Based Methodology Study

## Application Level Trace

- Thread
- File/Dir
- Attrs etc...

## Workload Model



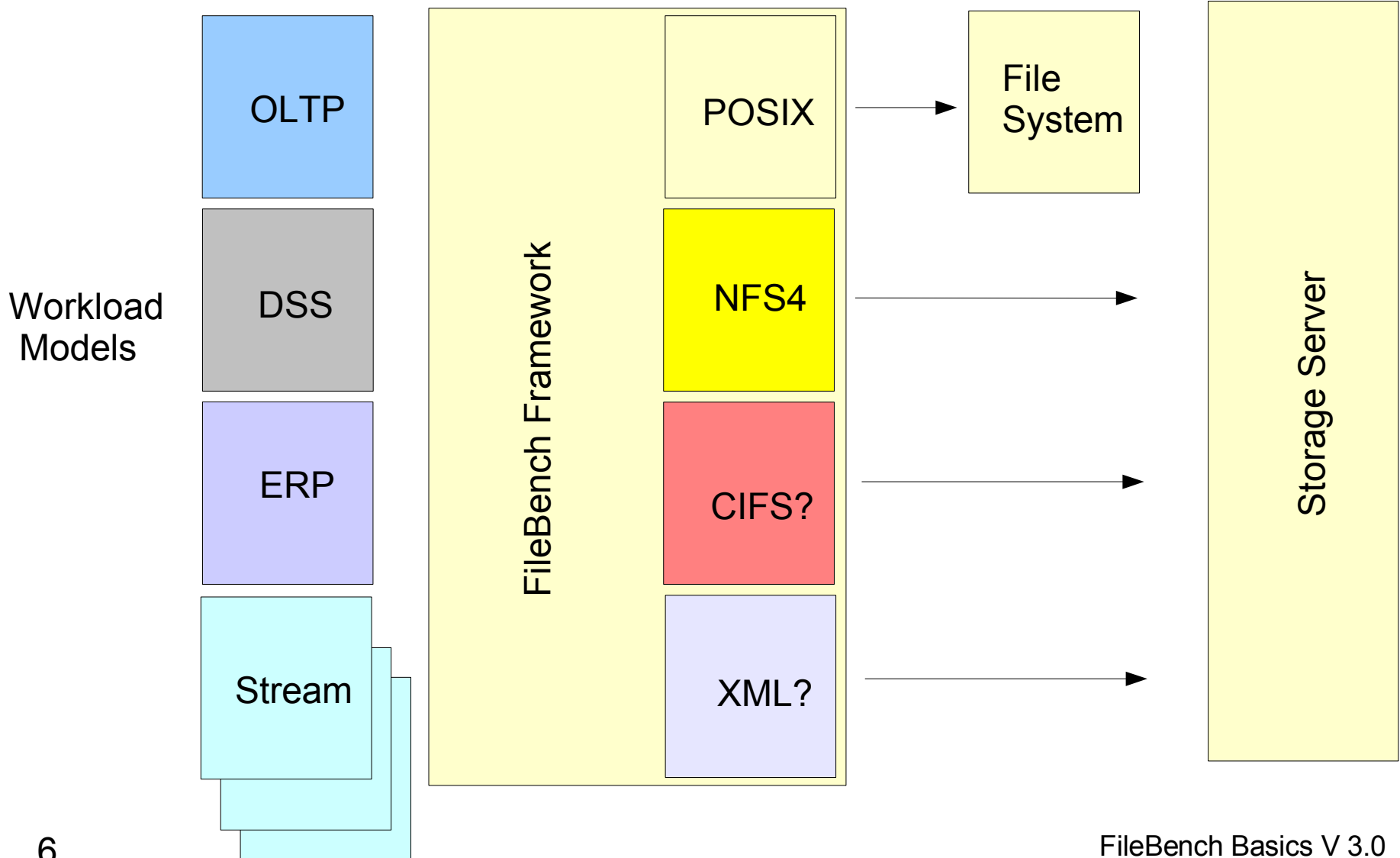
## Workload Replay

- Scale Factors

## Measurement Target

- FS, Client, Server etc
- Measurement Attrs

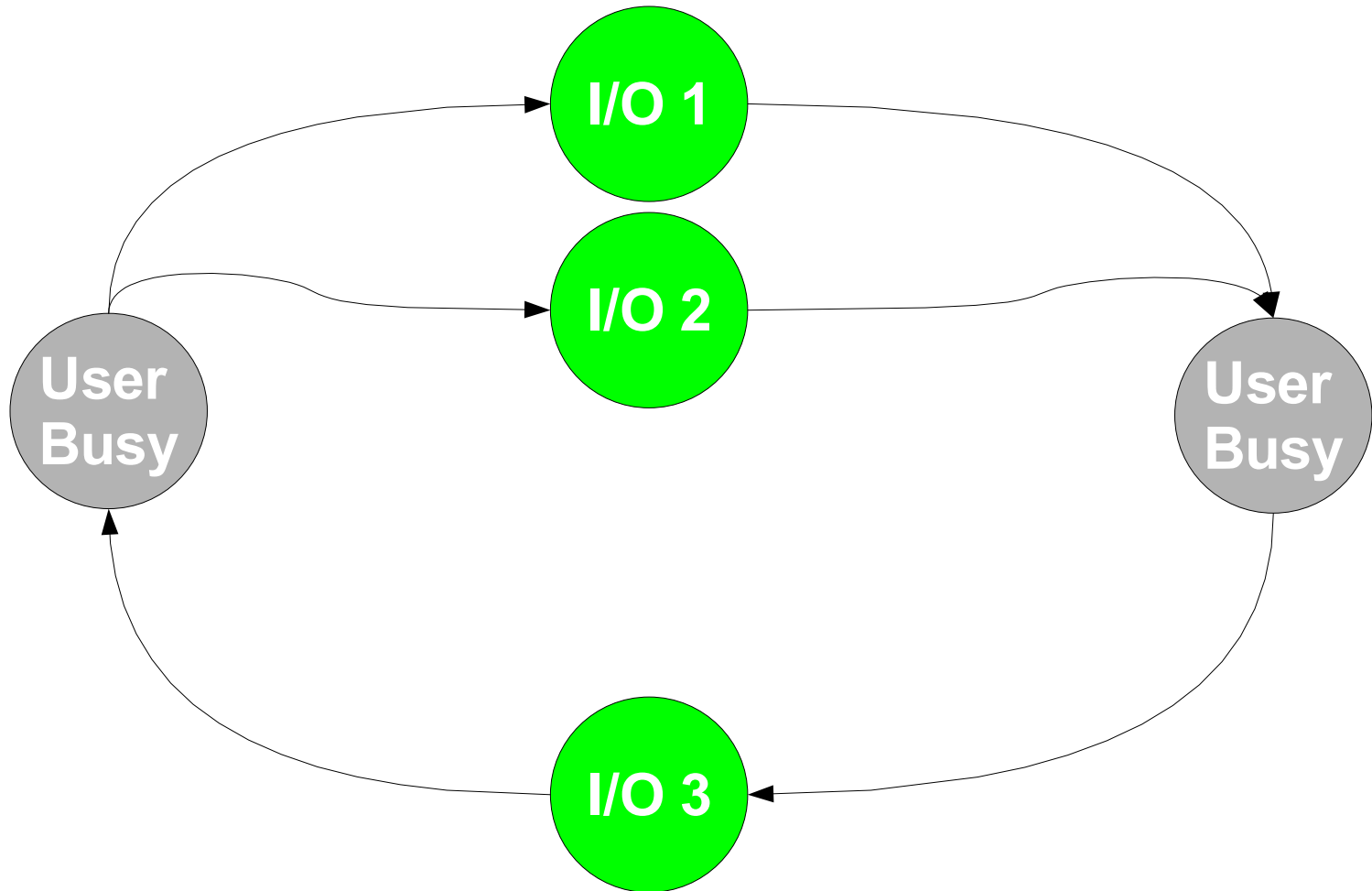
# FileBench Architecture



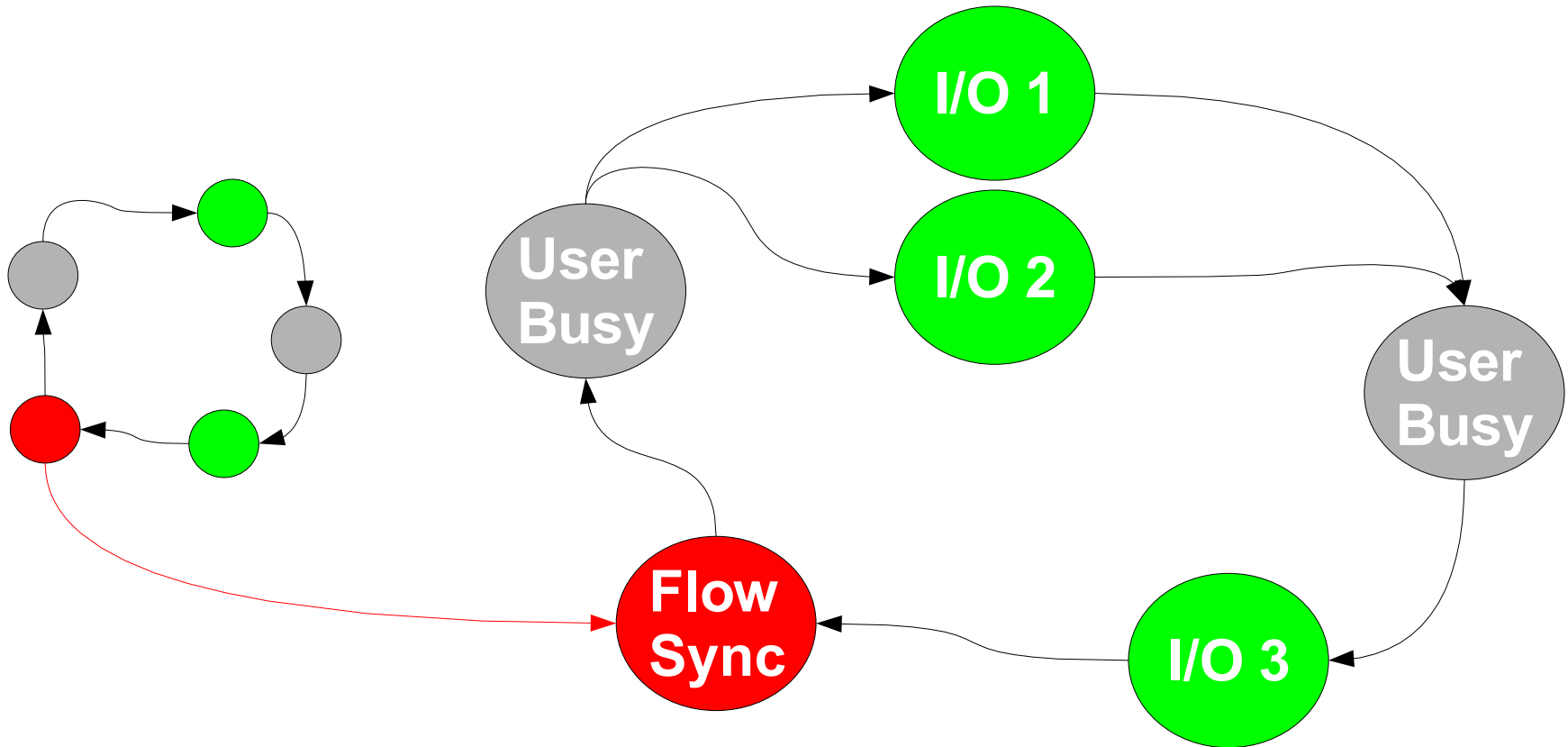
# Model Allows Complex/Important Scaling Curves

- 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

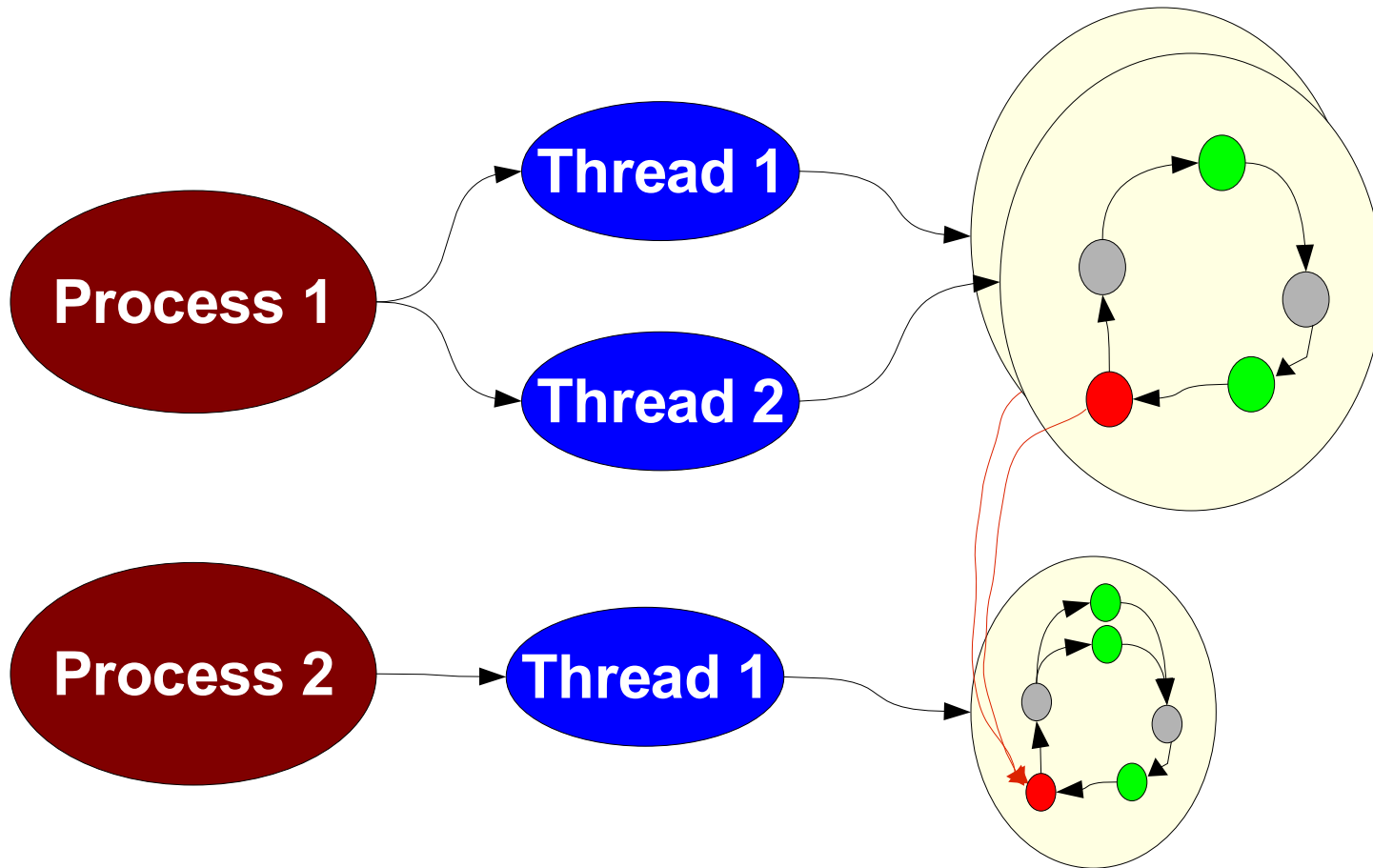


# Flow States: Synchronized Flow



Flow blocks until  
completion of other flow

# Characterize and Simulate via Cascades of Workload Flows:



# 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.

# Files and Filesets

- Files: a definition of a single file
  - Will soon be folded into 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

# Simple Random I/O Workload Description

Workload Model File: randomread.f:

```
set $dir=/tmp
set $nthreads=1
set $iosize=8k
set $filesize=1m

define file name=largefile1,path=$dir,size=$filesize,prealloc,reuse,paralloc

define process name=rand-read,instances=1
{
  thread name=rand-thread,memsize=5m,instances=$nthreads
  {
    flowop read name=rand-read1,filename=largefile1,iosize=$iosize,random
    flowop eventlimit name=rand-rate
  }
}
```

# Example FileBench Run

```

myhost% filebench randomread
parsing profile for config: randomread2k
Running /tmp/myhost-tmpfs-randomread-Sep_28_2007-07h_53m_00s/randomread2k/thisrun.f
FileBench Version 1.0.0
 4599: 0.009: Random Read Version 1.12 2005/06/25 01:03:16 IO personality successfully loaded
 4599: 0.010: Creating/pre-allocating filesets
 4599: 0.011: File largefile1: mbytes=160
 4599: 0.012: Re-using file largefile1 on tmpfs file system.
 4599: 0.012: Creating file largefile1...
 4599: 0.012: Preallocated 1 of 1 of fileset largefile1 in 1 seconds
 4599: 0.012: waiting for fileset pre-allocation to finish
 4599: 19.186: Change dir to /tmp/myhost-tmpfs-randomread-Sep_28_2007-07h_53m_00s/randomread2k
 4599: 19.186: Starting 1 rand-read instances
 4602: 20.196: Starting 1 rand-thread threads
 4599: 23.206: Running...
 4599: 83.636: Run took 60 seconds...
 4599: 83.637: Per-Operation Breakdown
rand-rate           0ops/s   0.0mb/s   0.0ms/op   0us/op-cpu
rand-read1         329ops/s  0.6mb/s   3.0ms/op   48us/op-cpu

 4599: 83.637:
IO Summary:      19891 ops 329.2 ops/s, (329/0 r/w)  0.6mb/s,   3108us cpu/op,   3.0ms latency
 4599: 83.637: Stats dump to file 'stats.randomread2k.out'
 4599: 83.637: in statsdump stats.randomread2k.out
 4599: 83.637: Shutting down processes
Generating html for /tmp/myhost-tmpfs-randomread-Sep_28_2007-07h_53m_00s
file = /tmp/myhost-tmpfs-randomread-Sep_28_2007-07h_53m_00s/randomread2k/stats.randomread2k.out
myhost%

```

# Running FileBench, Cont.

## Batch Mode

- Uses batch mode file with .prof extension
- Invoked with: **filebench <profname>**
- Example .prof used to run randomread.f from filebench in previous slide shown at right
- Distribution includes .prof files that execute suites of tests:
  - fileio.prof
  - filemicro.prof
  - filemacro.prof

## Configuration: randomread.prof:

```

DEFAULTS {
    runtime = 60;
    dir = /tmp;
    stats = /tmp;
    filesystem = tmpfs;
    description = "randomread tmpfs";
}

CONFIG randomread2k {
    function = generic;
    personality = randomread;
    filesize = 160m;
    iosize = 2k;
    nthreads = 1;
}

```

## Interactive Mode

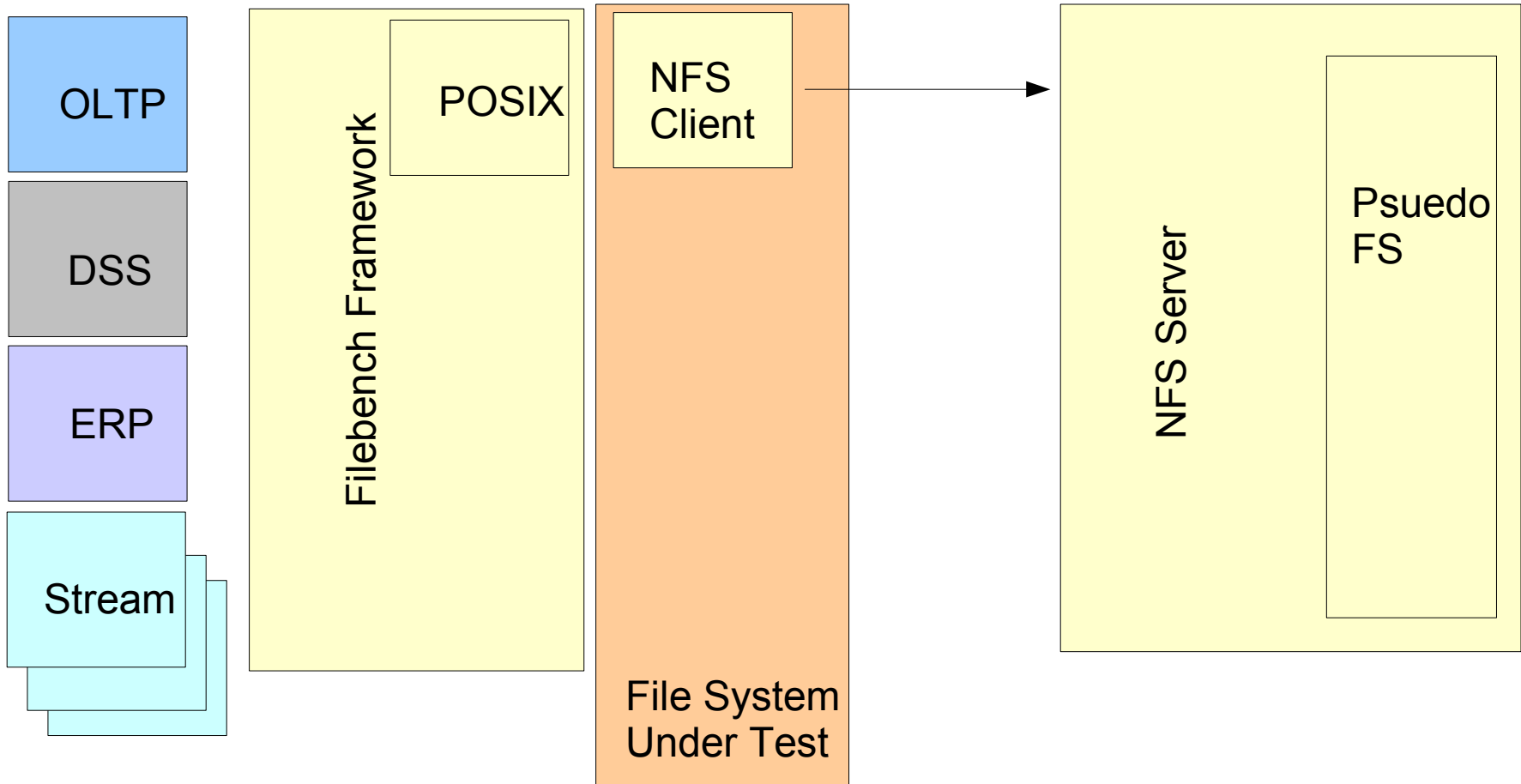
- Can directly run individual .f workload files
- Invoked with: **go\_filebench**
- Load workloads and set parameters from filebench prompt: filebench>

# Running a Single FileBench Workload

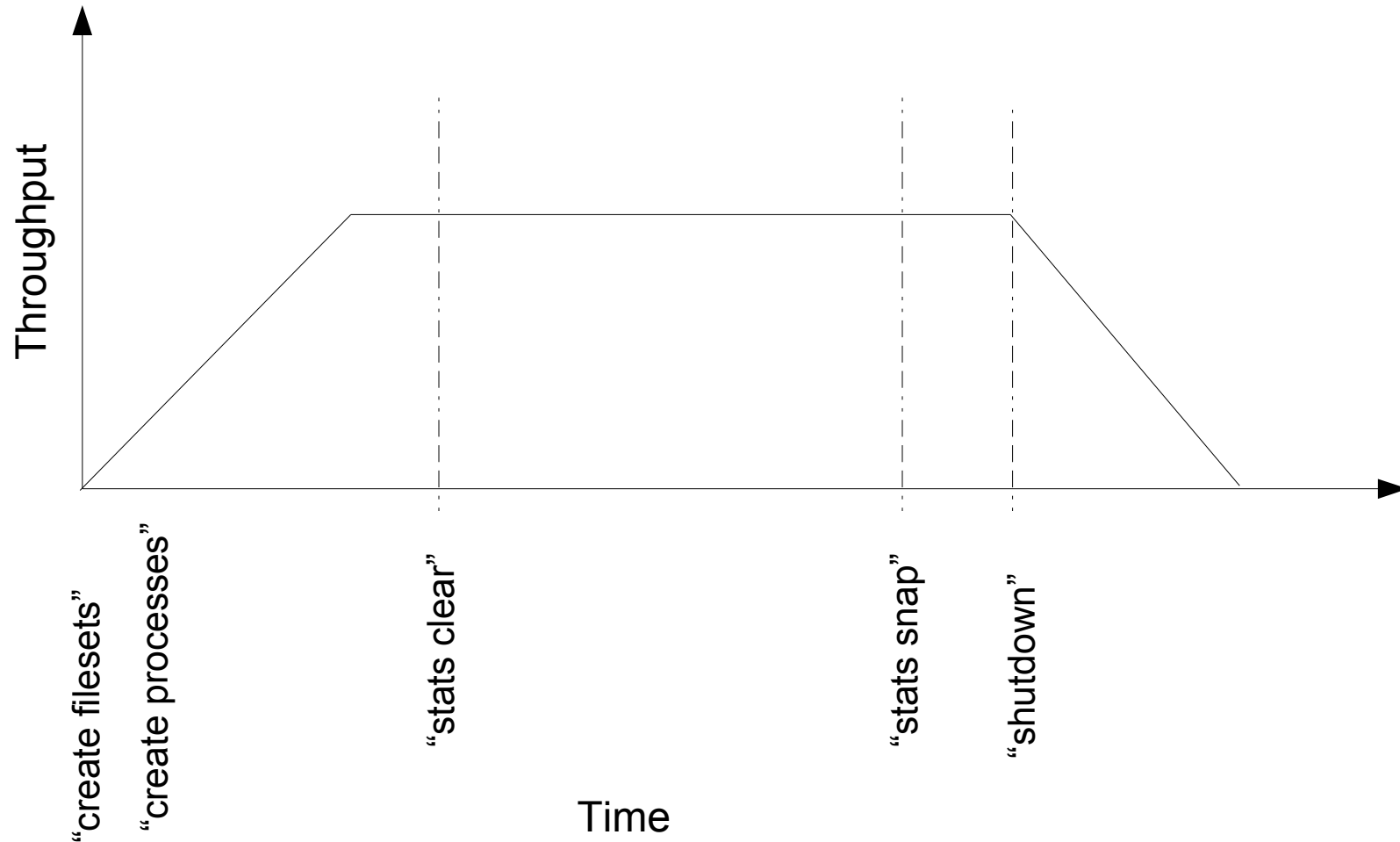
Example varmail run:

```
myhost% go_filebench
filebench> load varmail
Varmail Version 1.24 2005/06/22 08:08:30 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> (e.g. run 60)
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
filebench> stats dump "stats.varmail"
filebench> quit
```

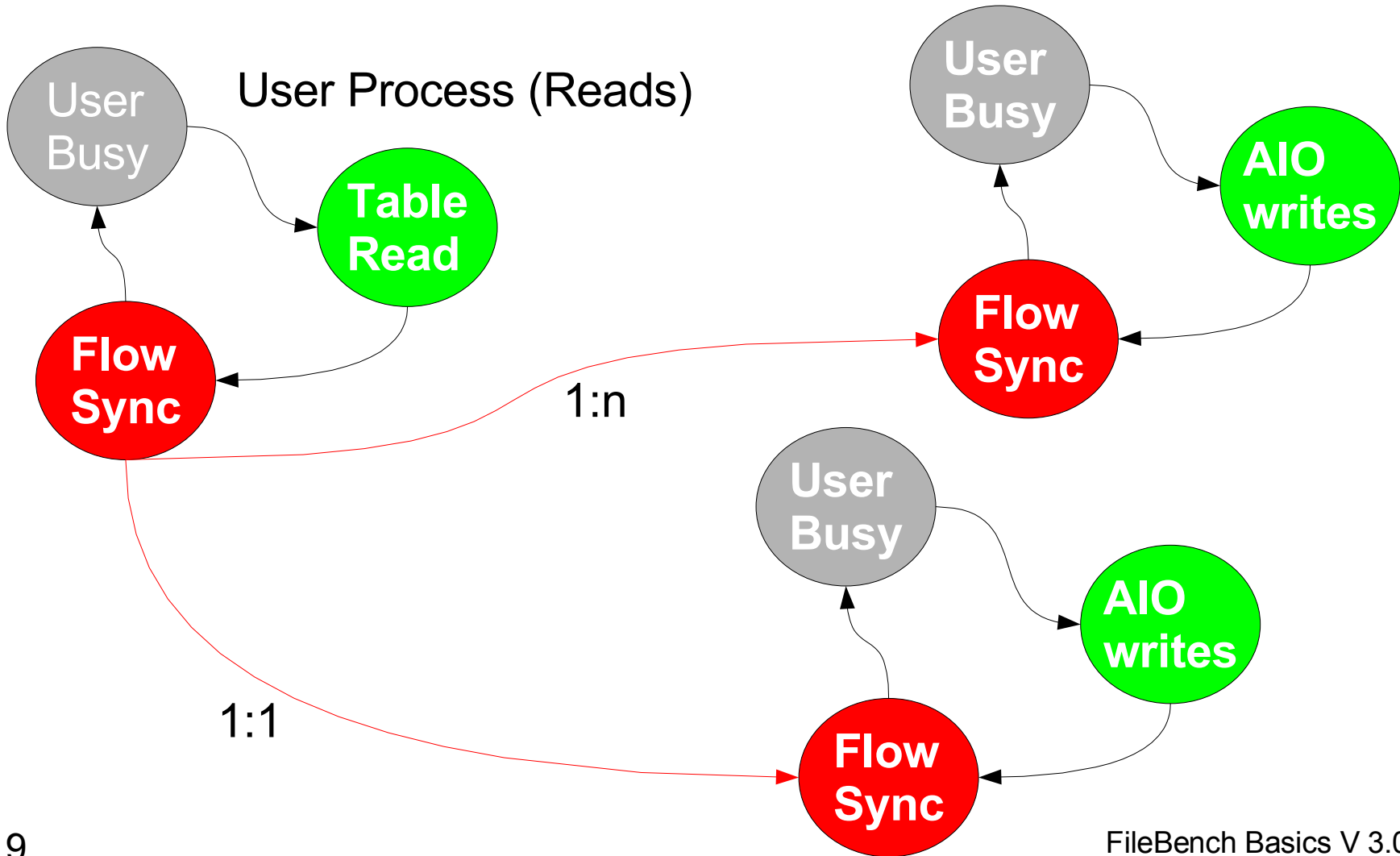
# NFS Client testing: POSIX level workload + NFS Server



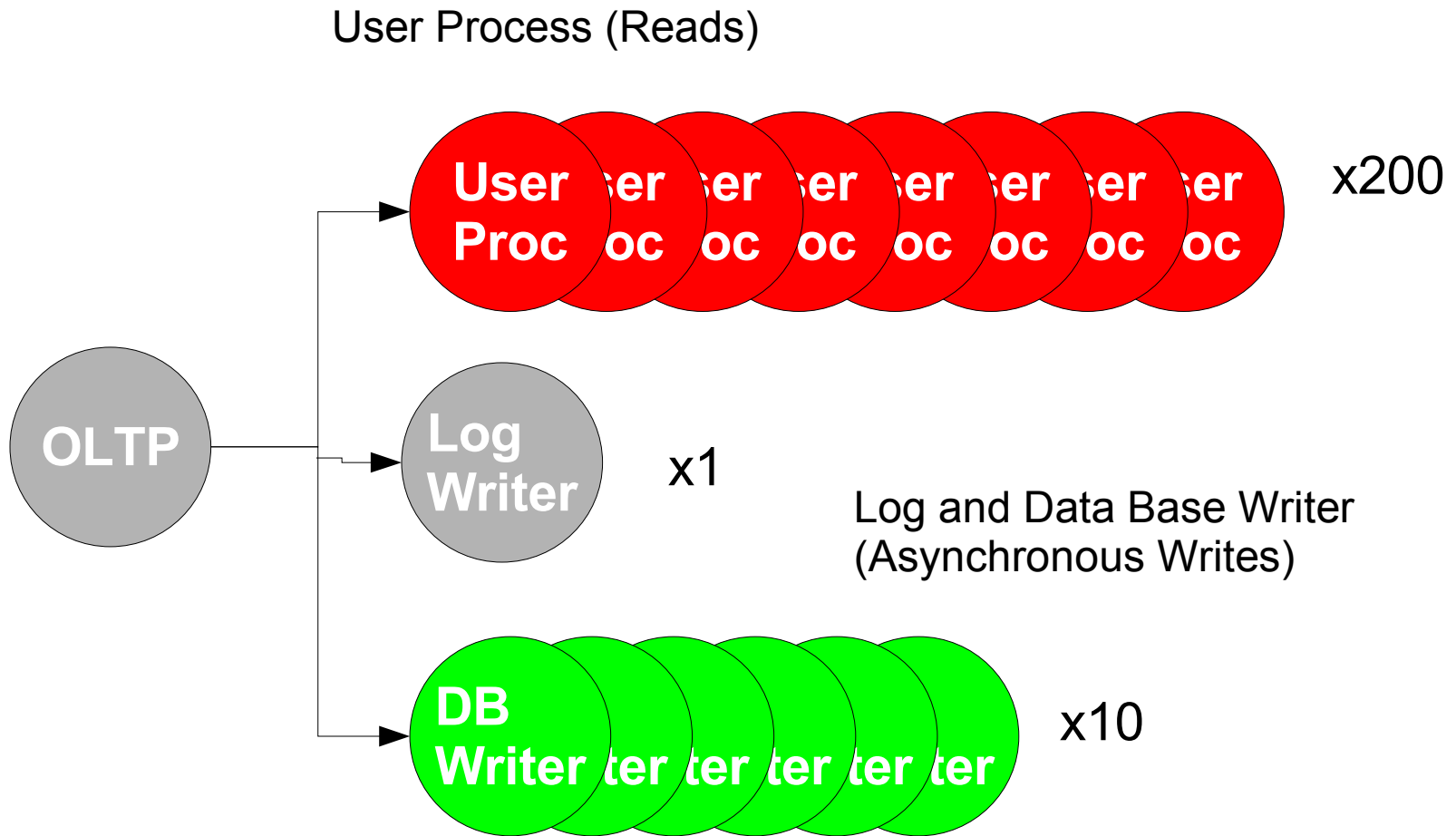
# The steps behind the “run” command



# Database Emulation Overview



# Database Emulation Process Tree





# 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

# Preparation for OpenSolaris

- Code cleanup
  - cstyle and lint clean
  - Remove unused code
  - Lots of additional comments
  - Full 64bit version for amd64
  - Additional error path handling
  - Linux cleanup
- Versioning of filebench and workloads
- filebench command now the perl “wrapper”
- go\_filebench is the “real” executable
- go\_filebench now offers command completion

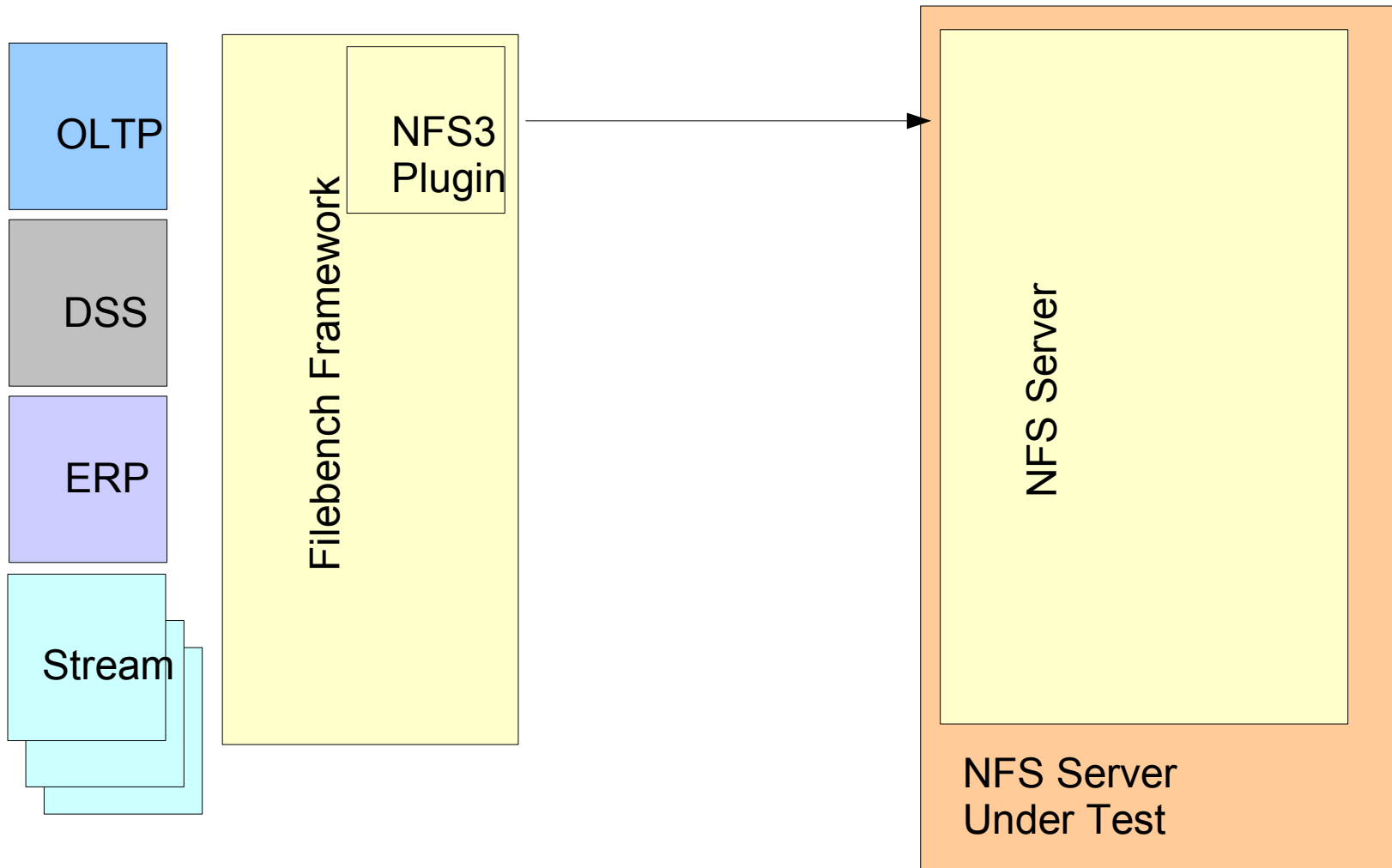
# FileBench in the wild

- NFS protocol verification
  - <http://nasconf.com/pres05/kustarz.pdf>
- Local file system improvement
  - [http://blogs.sun.com/erickustarz/entry/vdev\\_cache\\_improvements\\_to\\_help](http://blogs.sun.com/erickustarz/entry/vdev_cache_improvements_to_help)
- SATA/IO layer analysis
  - [http://blogs.sun.com/erickustarz/entry/ncq\\_performance\\_analysis](http://blogs.sun.com/erickustarz/entry/ncq_performance_analysis)
- (SCSI) protocol analysis
  - Non-volatile cache flushing with varmail workload

# FileBench Issues

- FileBench can not deal with “no work to do”
  - Work around is to increase “nfiles”
- No method to tell workloads “run to completion”
  - This is for varmail/postmark type workloads
  - Potential solution may be to set “runtime” to 0
- Linux / OSX / \*BSD support
  - Mostly works
  - Need additional use and verification
- Ease of use
  - .prof syntax is very picky
  - Error messages should provide pointers to solutions
- Integration with multi-client framework

# Future: NFS Plugin



# 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_for_Programmers)
- [http://www.solarisinternals.com/wiki/index.php/FileBench\\_Workload\\_Language](http://www.solarisinternals.com/wiki/index.php/FileBench_Workload_Language)
- [http://www.solarisinternals.com/wiki/index.php/FileBench\\_Workload\\_Modeling\\_Guide](http://www.solarisinternals.com/wiki/index.php/FileBench_Workload_Modeling_Guide)

# FileBench

Spencer Shepler  
([blogs.sun.com/shepler](http://blogs.sun.com/shepler))

Eric Kustarz  
([blogs.sun.com/erickustarz](http://blogs.sun.com/erickustarz))

Andrew Wilson  
([Andrew.W.Wilson@sun.com](mailto:Andrew.W.Wilson@sun.com))