

FileBench

A Prototype Model Based Workload for File Systems

Work In Progress

Richard McDougall

Sun Microsystems



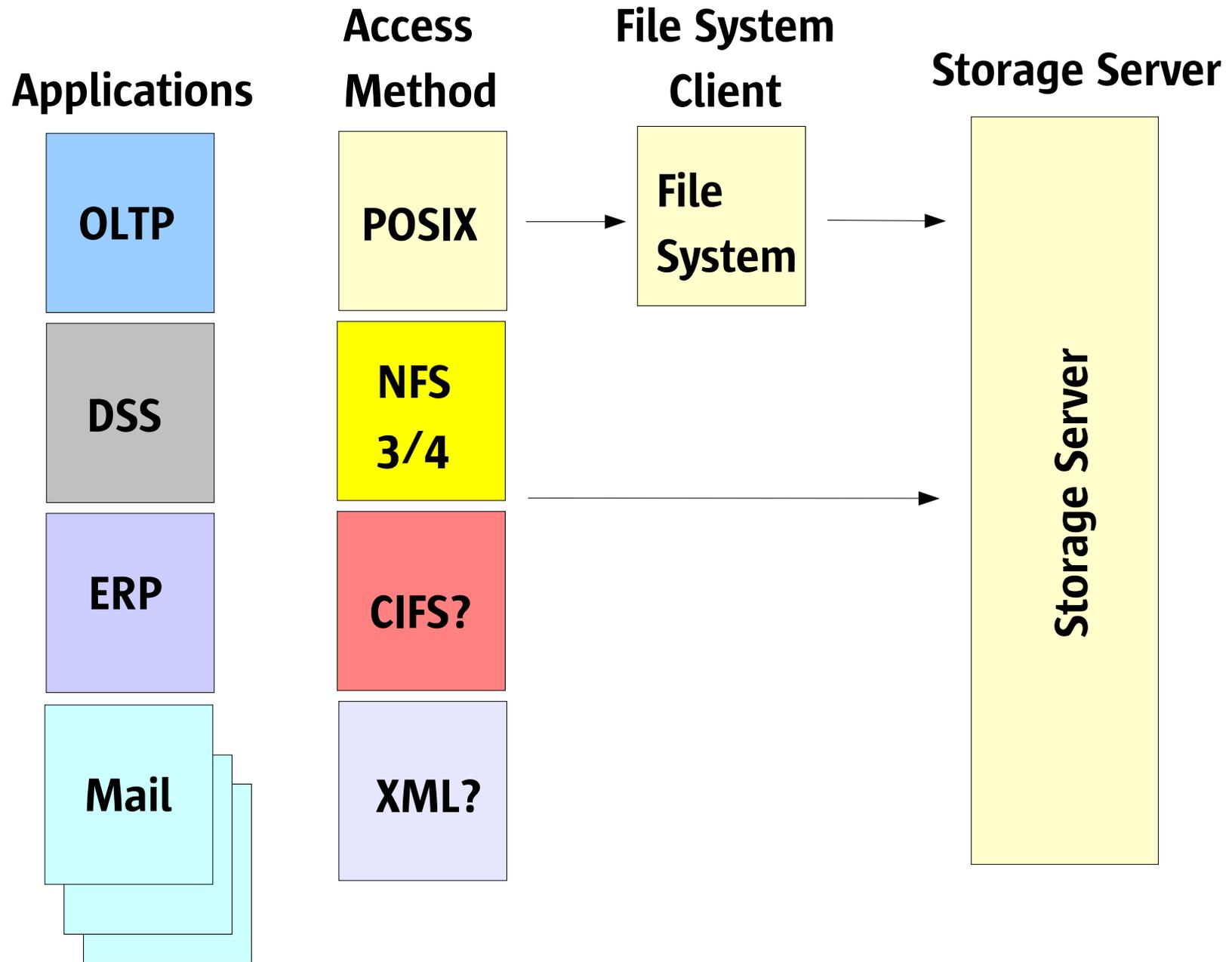
Benchmarks?

- For Vendors
 - Product characterization
 - Product design goaling
 - Benchmarketing
- For Customers
 - Purchasing Guide
 - Configuration characterization/tuning/verification

Requirements for file-level benchmarking

- Represent Apps rather than I/Os
- Trace-derived synthesis
- Thread-level representation
- Inter-thread dependency/sync.
- Forward Path
- Extensible to new protocols
- Modular to include test of client:
 - process/thread model,
 - cpu efficiency etc...
- Pre-structuring/aging of file sets
- Scalable
 - Throughput, #Users
 - #Files/Directories
 - Working set size
 - #Clients
 - Client resources (mem/cpu)

What do we want to characterize?



Characterization Strategies

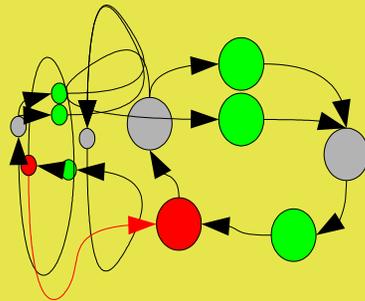
- I/O Microbenchmarking
 - Pros: Easy to run
 - Cons: Small test coverage, Hard to correlate to real apps
- Trace Capture/Replay
 - I/O Trace, NFS Trace, Application Trace
 - Pros: Accurate reconstruction of real application I/O mix
 - Cons: Large traces, difficult to reconstruct I/O dependencies
- Model Based
 - Distillation of trace into representative model
 - Probability based, Simulation based
 - Pros: Easy to run, Scalable in multiple dimensions
 - Cons: Care required to ensure accurate real-world representation

Model based methodology study

Application Level Trace

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

Workload Model



Workload Replay

- Scale Factors

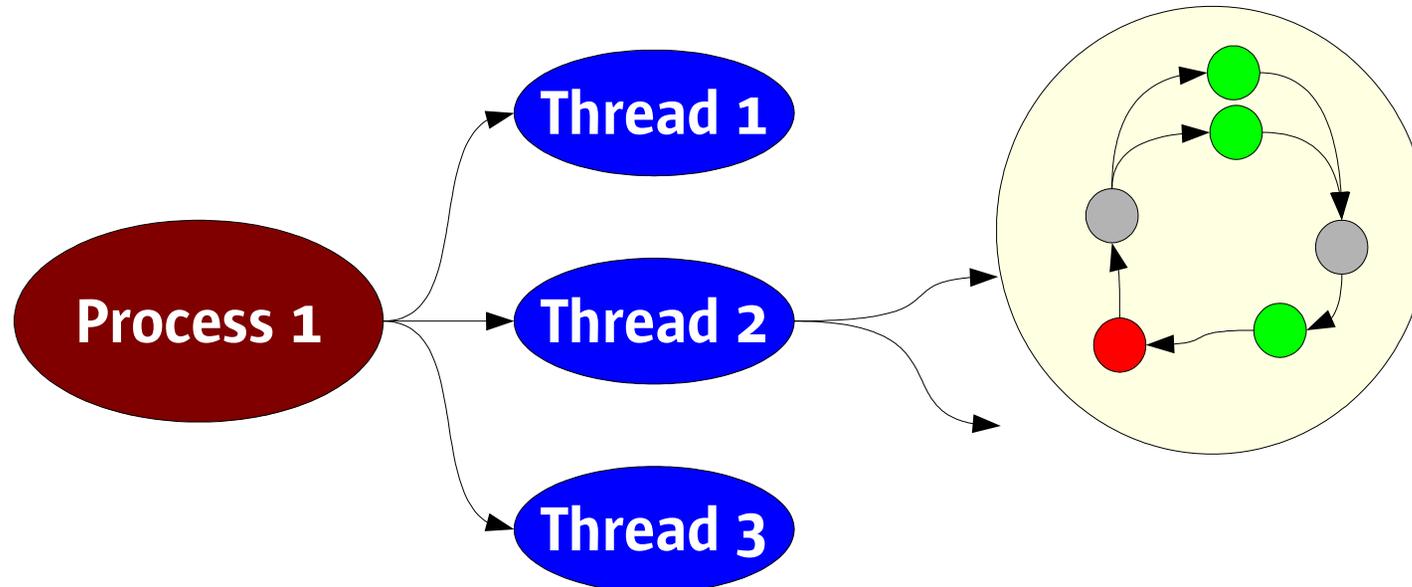
Measurement Target

- FS, Client, Server etc
- Measurement Attrs

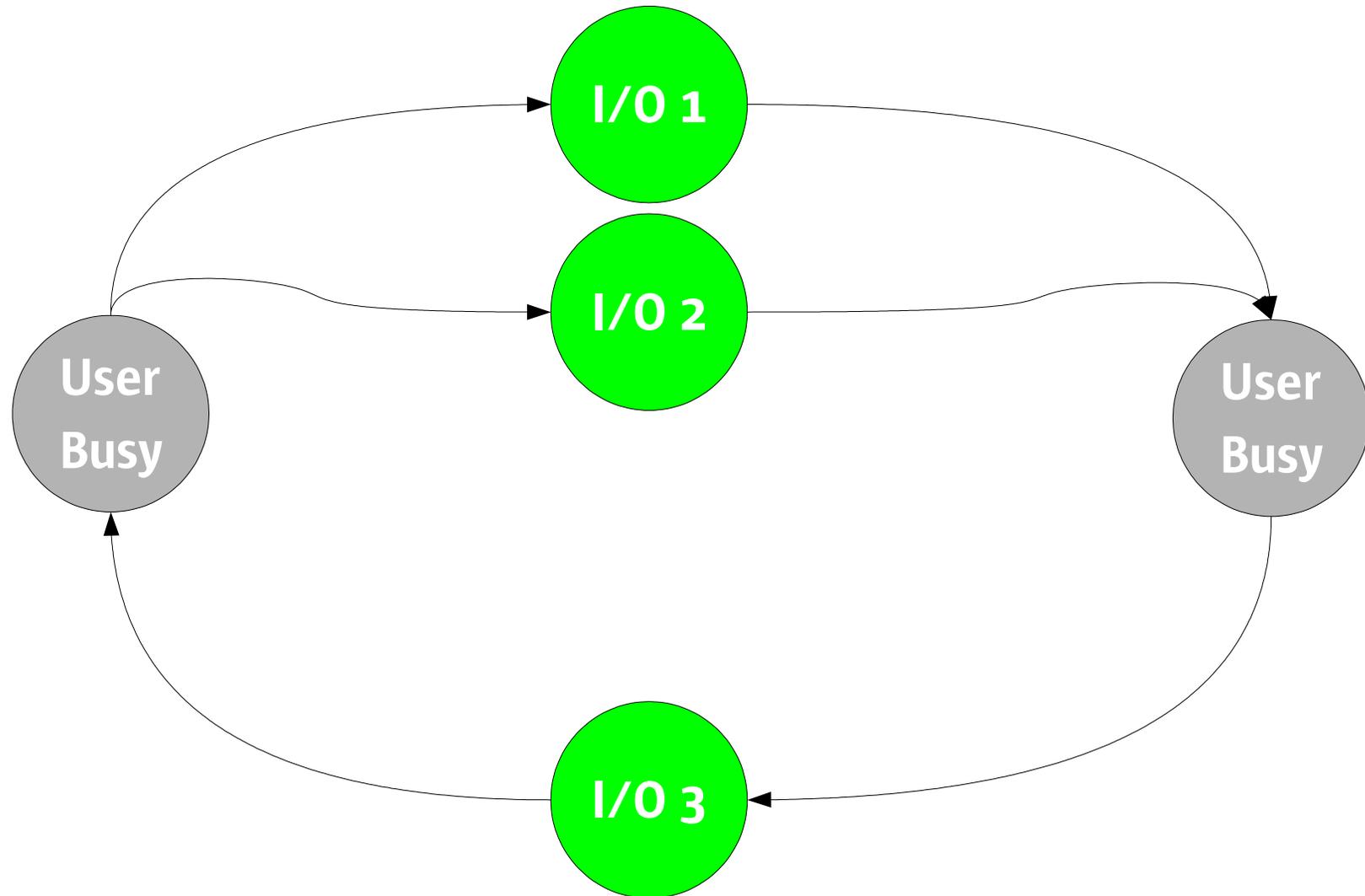
Model Allows Complex/Important Scaling Curves

- e.g.
 - Throughput/Latency vs. Working set size
 - Throughput/Latency vs. #users
 - CPU Efficiency vs. Throughput
 - Caching efficiency vs. Workingset size/Memsize

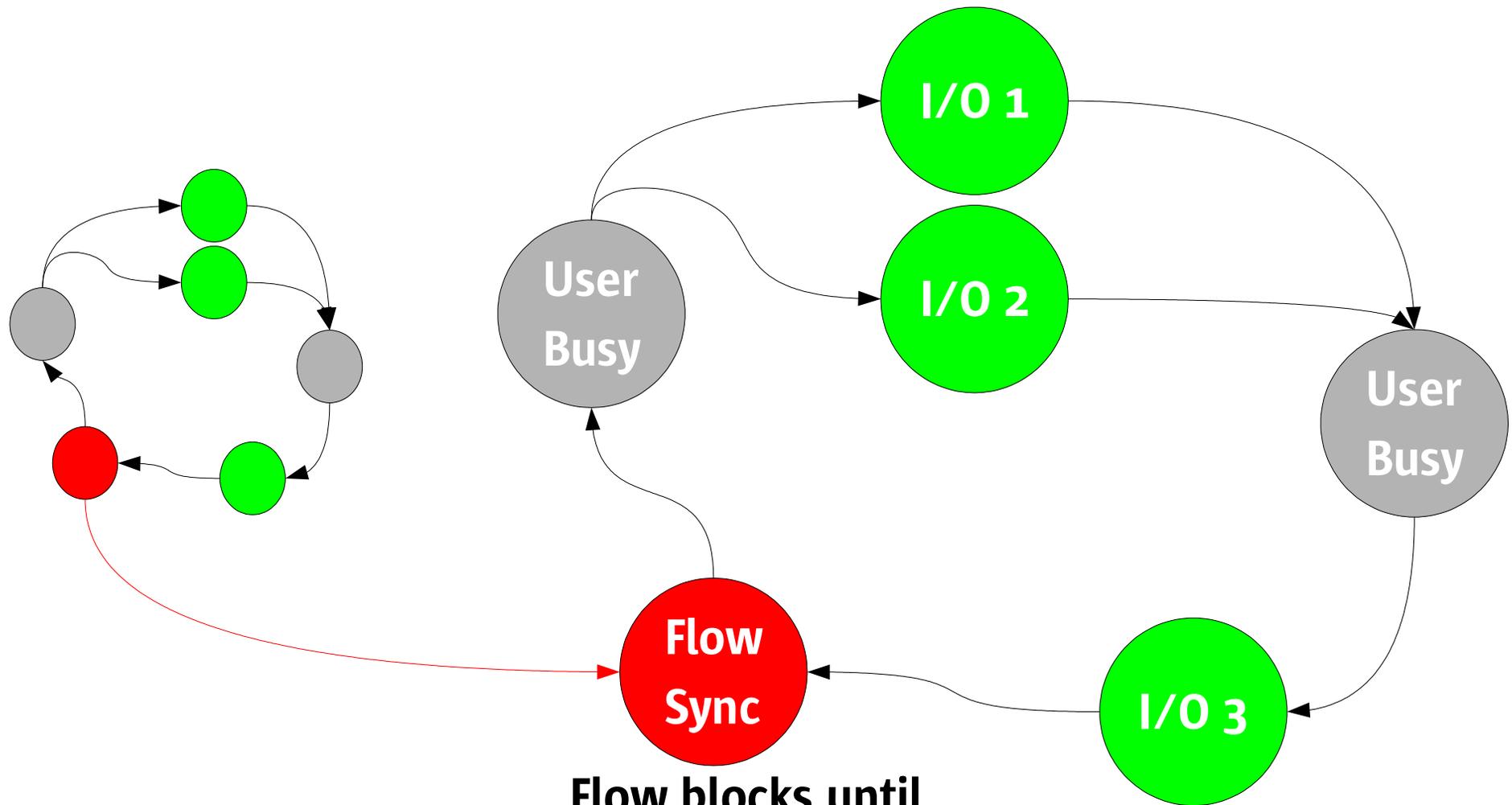
Characterize and Simulate via Cascades of Workload Flows:



Flow States: Open Ended Flow



Flow States: Synchronized Flow



**Flow blocks until
completion of other 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
 - I/O Pattern, probabilities
 - Working set size
 - Etc...

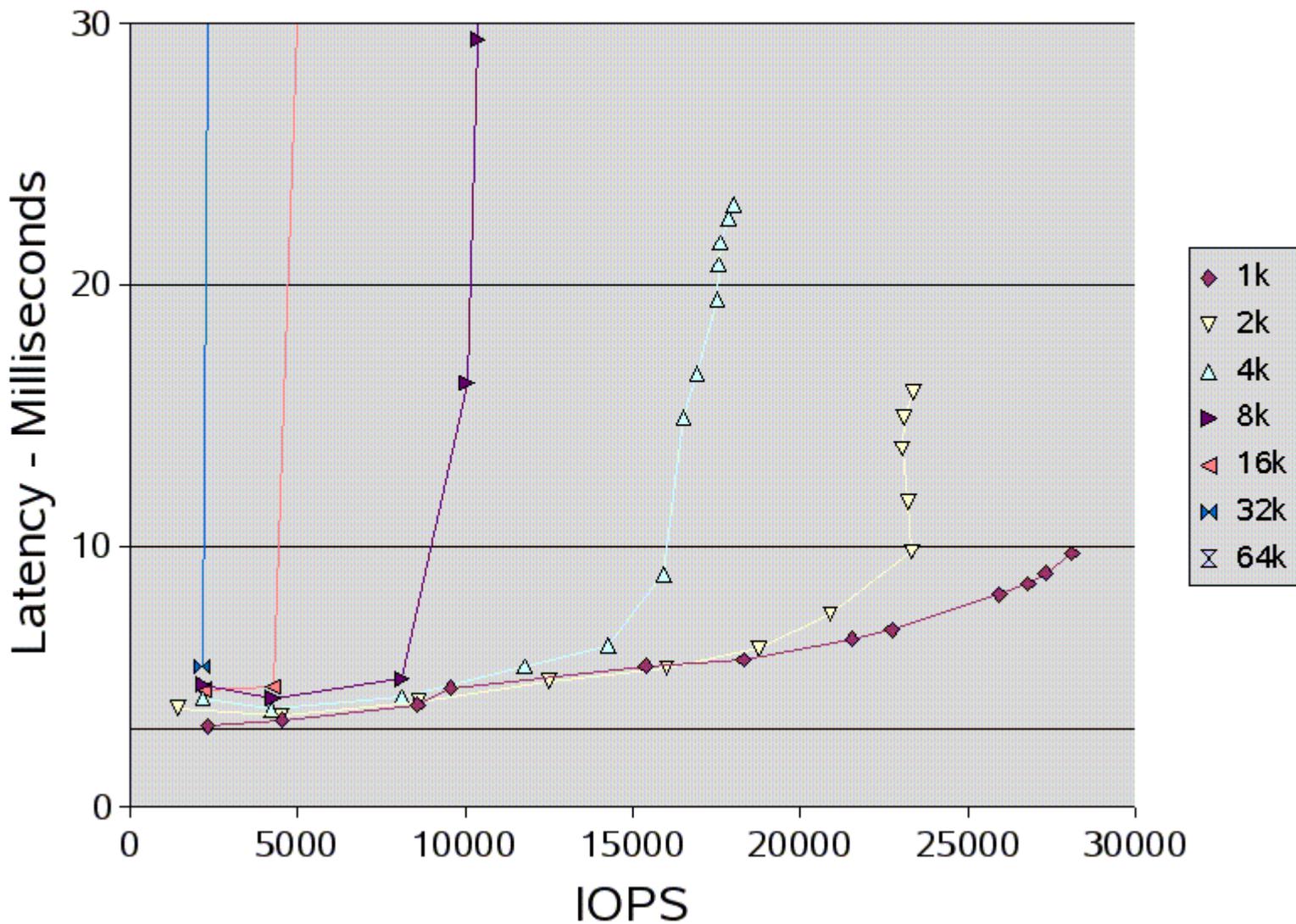
Simple Random I/O Workload Description

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

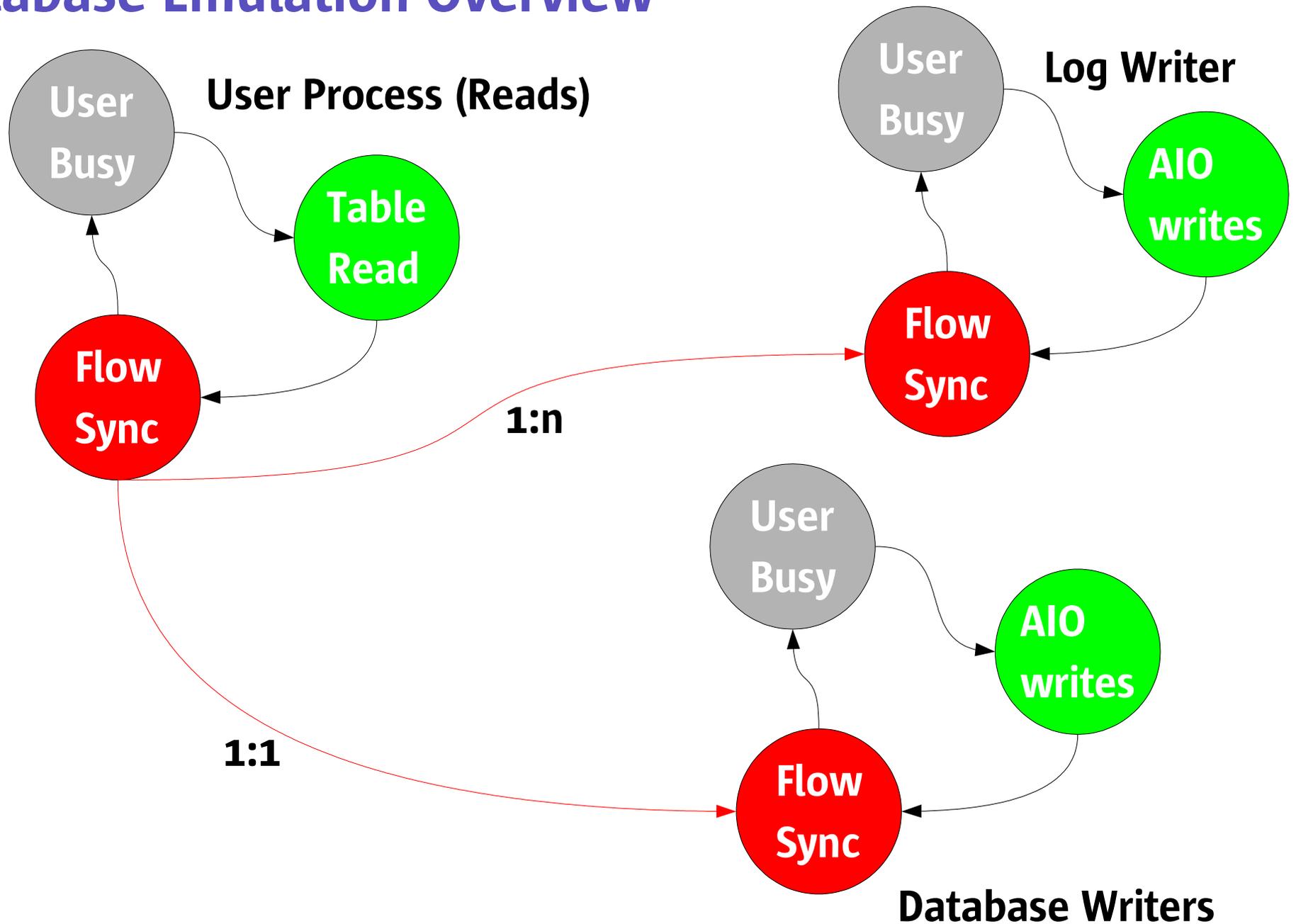
define process name=rand-read,instances=1
{
  thread name=rand-thread,memsize=5m,instances=$nthreads
  {
    flowop read name=rand-read1,filename=bigfile0,iosize=$iosize,random
    flowop eventlimit name=rand-rate
  }
}
```

Random I/O – NFS V3

Random I/O Latency

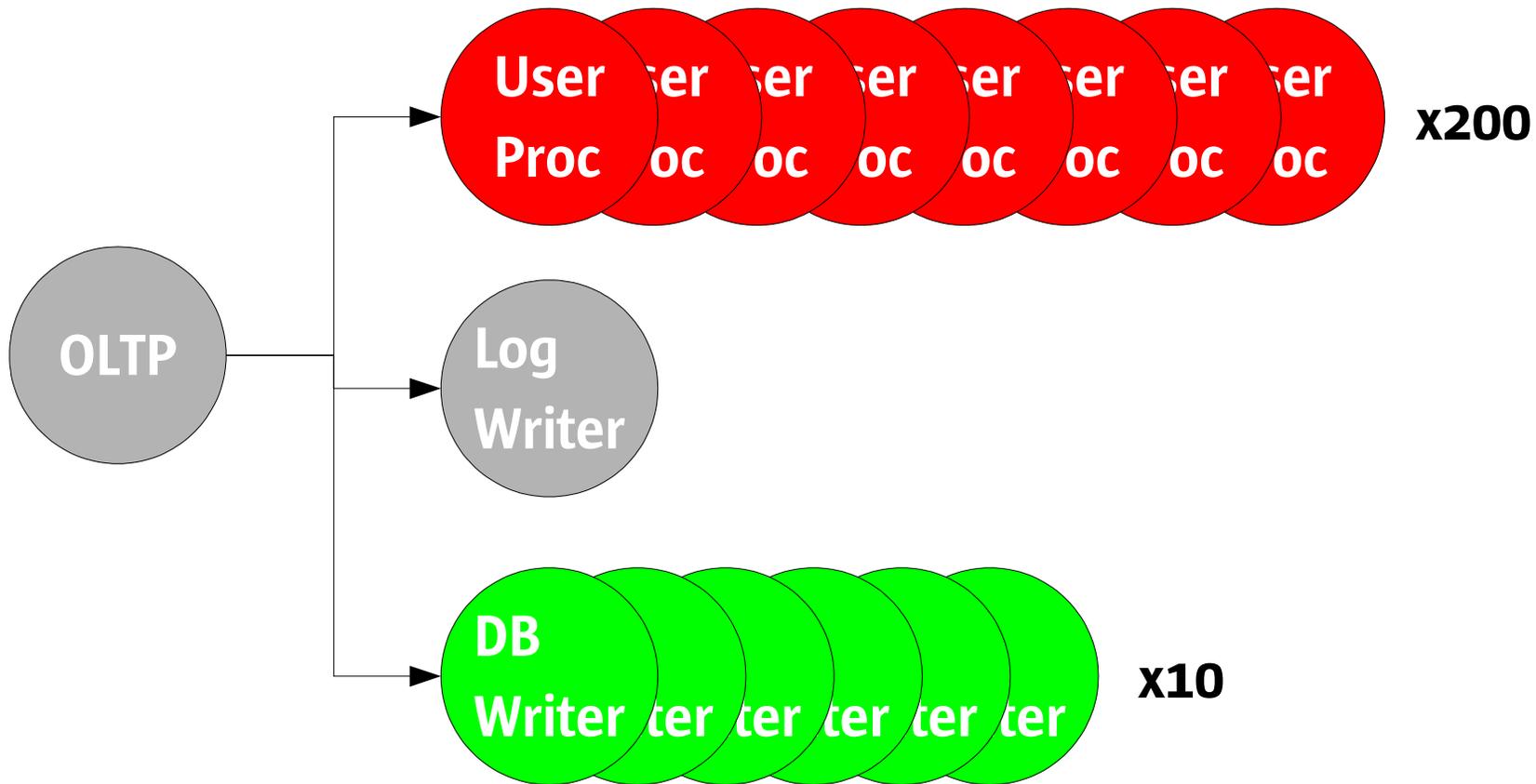


Database Emulation Overview



Database Emulation Process Tree

User Process (Reads)



Simplified OLTP Database Program

```
define file name=logfile,path=$dir,size=1g,reuse,prealloc,paralloc
define file name=datafilea,path=$dir,size=$filesize,reuse,prealloc,paralloc
define process name=dbwr,instances=$ndbwriters
{
  thread name=dbwr,memsize=$memperthread,useism
  {
    flowop aiowrite name=dbaiowrite-a,filename=datafilea,
      iosize=$iosize,workingset=10g,random,dsync,directio,itiers=10
    flowop hog name=dbwr-hog,value=10000
    flowop semblock name=dbwr-block,value=100,highwater=10000
    flowop aiowait name=dbwr-aiowait
  }
}

define process name=lgwr,instances=1
{
  thread name=lgwr,memsize=$memperthread,useism
  {
    flowop write name=lg-write,filename=logfile,
      iosize=256k,workingset=1g,random,dsync,directio
    flowop semblock name=lg-block,value=320,highwater=1000
  }
}

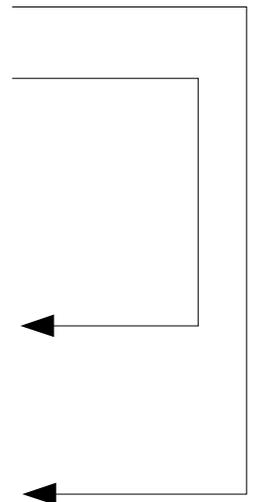
define process name=shadow,instances=$nshadows
{
  thread name=shadow,memsize=$memperthread,useism
  {
    flowop read name=shadowread-a,filename=datafilea,
      iosize=$iosize,workingset=10g,random,dsync,directio
    flowop hog name=shadowhog,value=$usermode
    flowop sempost name=shadow-post-lg,value=1,target=lg-block,blocking
    flowop sempost name=shadow-post-dbwr,value=1,target=dbwr-block,blocking
    flowop eventlimit name=random-rate
  }
}
```



OLTP Program – Benchmark Result Detail

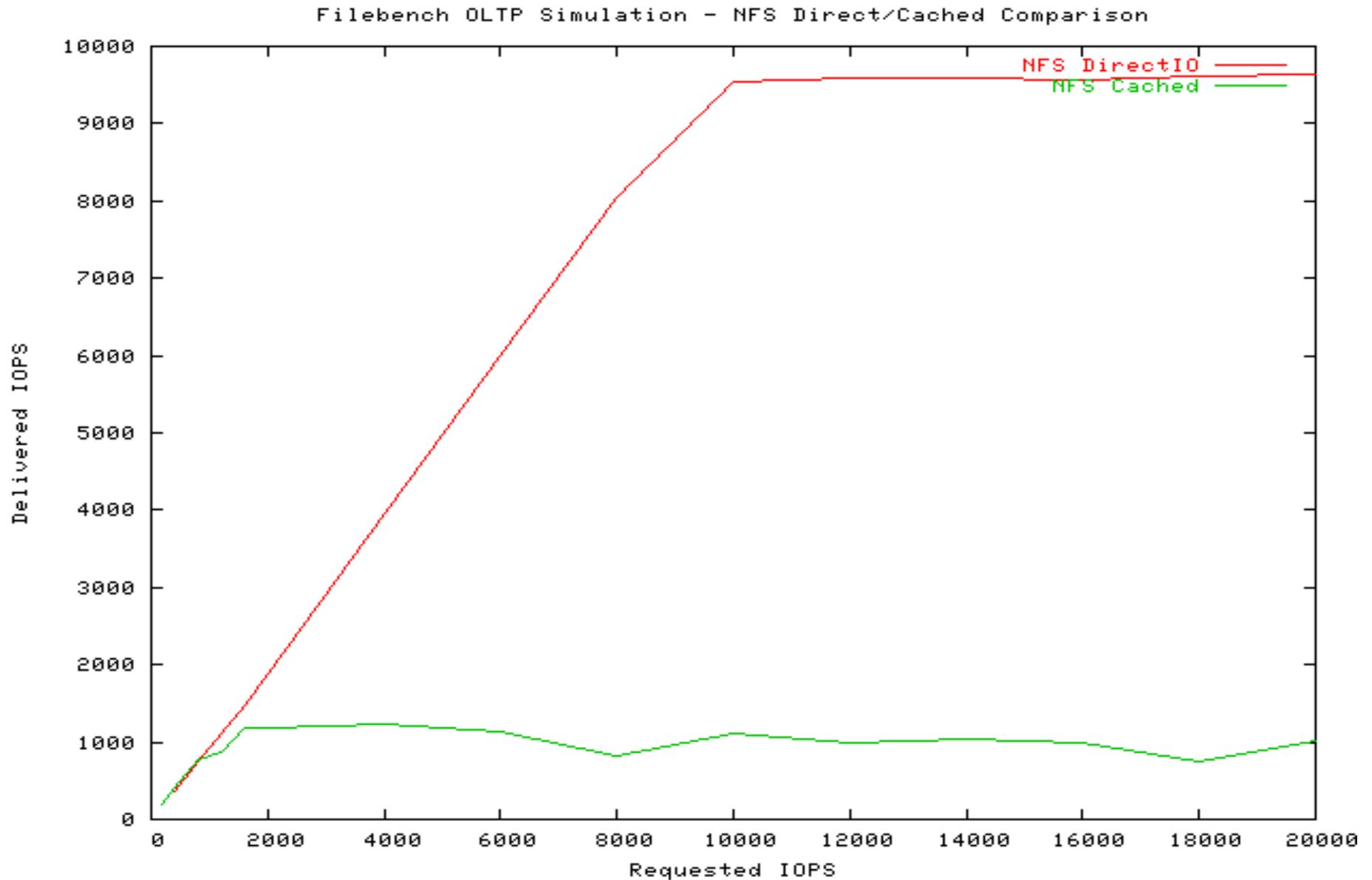
Flowop totals:

shadow-post-dbwr	4554ops/s	0.0mb/s	215.7ms/op	91us/op-cpu
shadow-post-lg	4555ops/s	0.0mb/s	0.7ms/op	21us/op-cpu
shadowhog	4546ops/s	0.0mb/s	2.5ms/op	111us/op-cpu
shadowread	4455ops/s	0.9mb/s	23.2ms/op	89us/op-cpu
lg-block	100ops/s	0.0mb/s	605.2ms/op	305us/op-cpu
lg-write	100ops/s	0.4mb/s	96.2ms/op	1962us/op-cpu
dbwr-aiowait	4445ops/s	0.0mb/s	144.0ms/op	242us/op-cpu
dbwr-block	4445ops/s	0.0mb/s	9.6ms/op	44us/op-cpu
dbwr-hog	4445ops/s	0.0mb/s	1.1ms/op	50us/op-cpu
dbaiowrite	4449ops/s	0.9mb/s	0.2ms/op	17us/op-cpu



IO Summary: 9087.7 ops/s, 4547/4496 r/w 18.0mb/s, 129uscput/op

NFS OLTP – IOPS Scaling



Workload Discussion



File Access

Workload	File Size	# files	#Streams	Sharing	I/O Mix	Seek Mode Random Read/10% Sequential Write	Access type mmap/posix
Web Server	Small	Large	Large	Low	<5% 50r/50w, 1% large		Both
Small DB	Large	Small	~100	High	sequential 50r/50w, 1% large	99% Random	POSIX
Large DB	Large	Small	~1000	High	sequential	99% Random	POSIX
DB Mail Server	Large	Small	>1000	High	?		
NFS Mail Server	Moderate	Moderate	>10k	Low	?	Sequential	POSIX
HPTC	Huge	Small	Small	Low	50r/50w	Sequential	POSIX
SW Development	Small	Large	>1000	Low	5r/5w/90a	Sequential	POSIX
Video Streaming							

I/O Characteristics

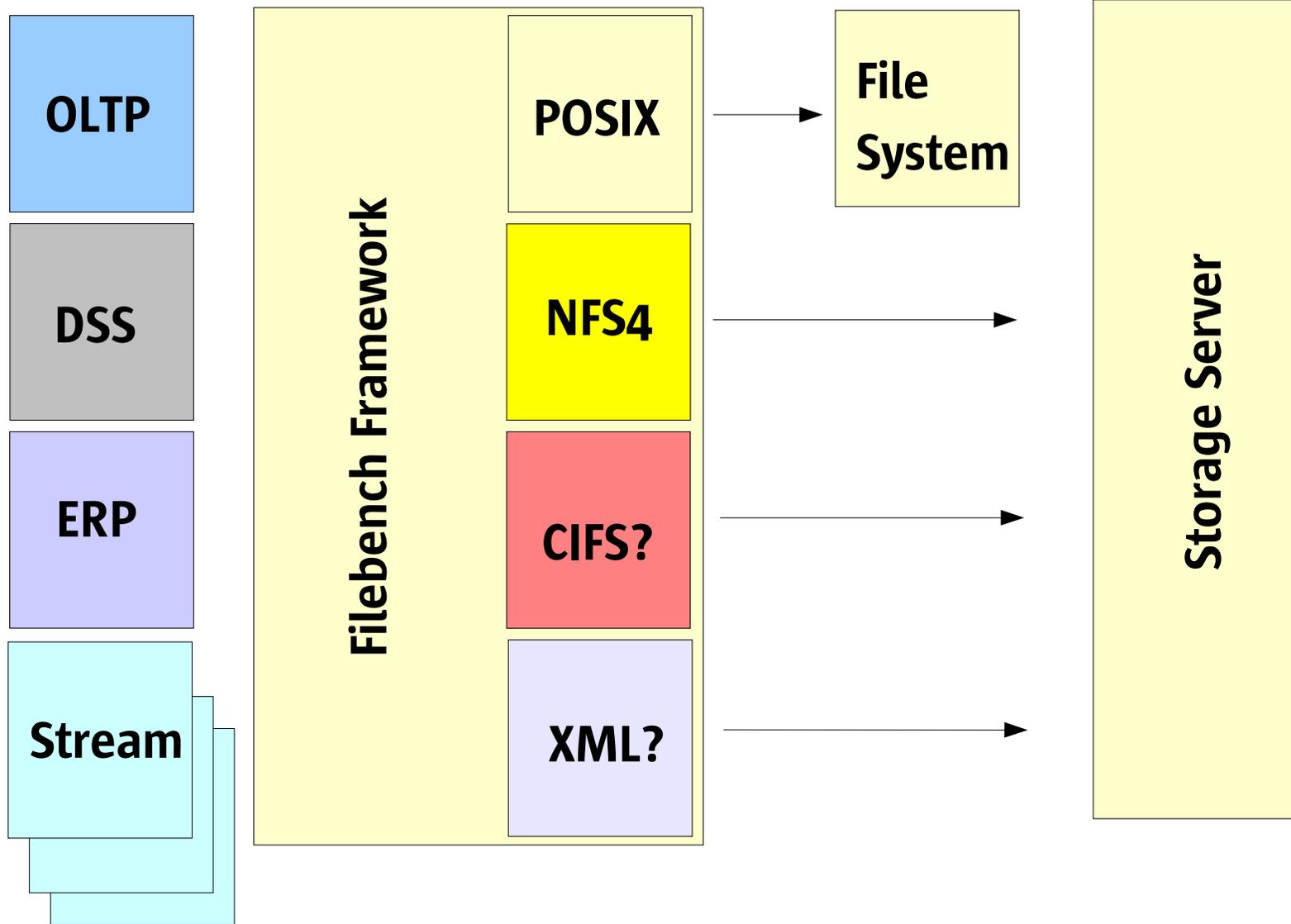
Workload	App/I/O CPU Content	Typical IOPS cient	Data Set Size	Working Set Size	Typical I/O Size	Typical Bandwidth
Web Server	99/1	<1000 per cient			<64k Random 2- 8k, 128k	<1MB/s
Small DB	90/10	~1000	1-10GB	50.00%	sequential Random 2- 8k, 128k	~10MB/s
Large DB	80/20	>10000	10GB-1TB	30.00%	sequential	50MB/s
DB Mail Server	90/10?				Small?	?
NFS Mail Server	90/10?	Low			Large reads, small writes	1-10MB/s >100MBs Client, 1GB/s Server
HPTC	80/20?	~1000?			~1MB	
SW Development	95/5?	~1000			~32k	~100mb/s

Example Composite

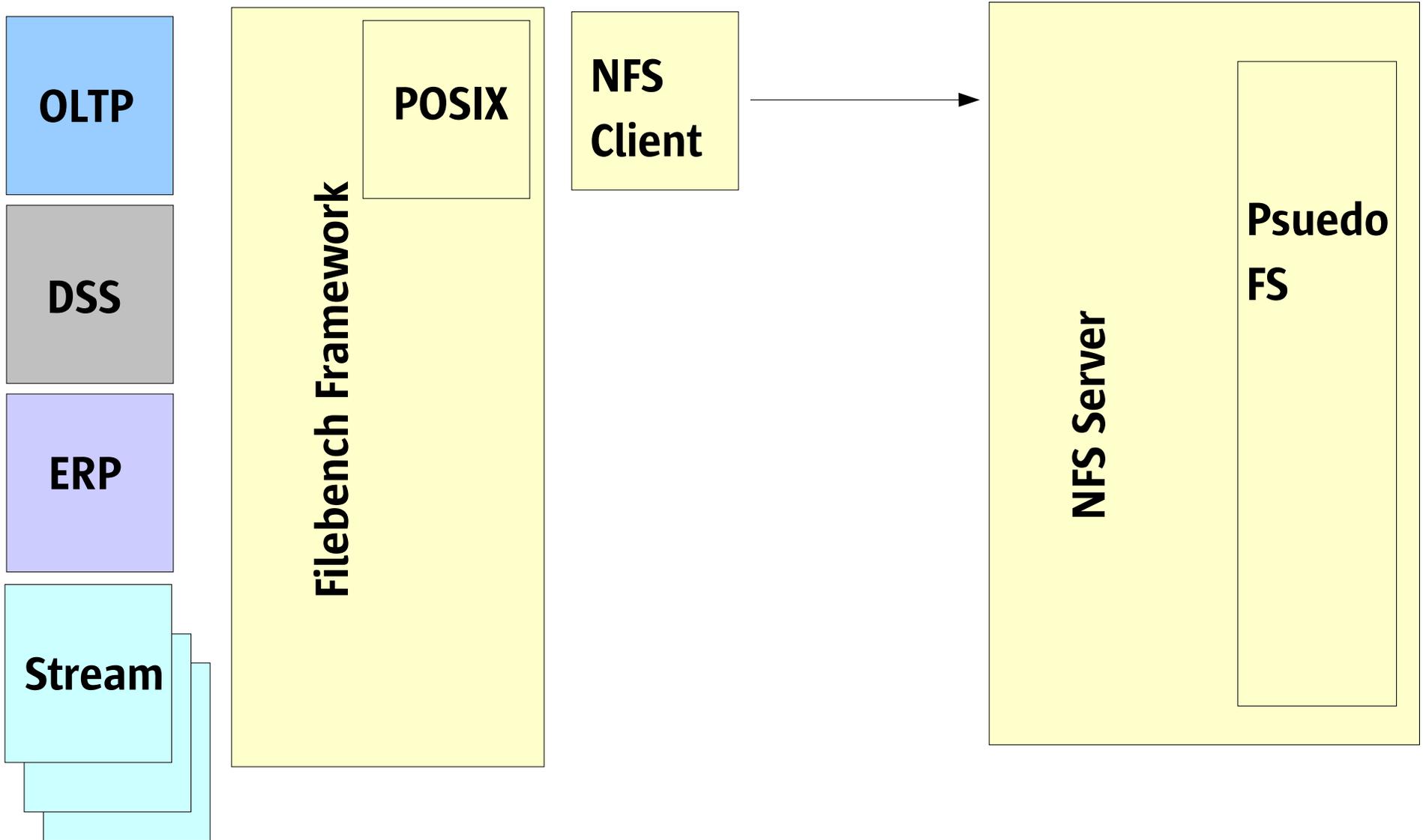
NAS Filer

Workload	IOPS	BW	Weight
OLTP-Small	5123	40	30
OLTP-Large	2532	21	10
ERP	4928	25	10
Web Serving	3241	3.5	5
Data Warehouse	75	75	5
HPTC – Single Stream	89	89	10
HPTC – Multi Stream	120	120	5
Mail-DB	2132	2	5
Mail-NFS	781	50	5
SW Development	4123	10	10
Video Streaming	120	120	5

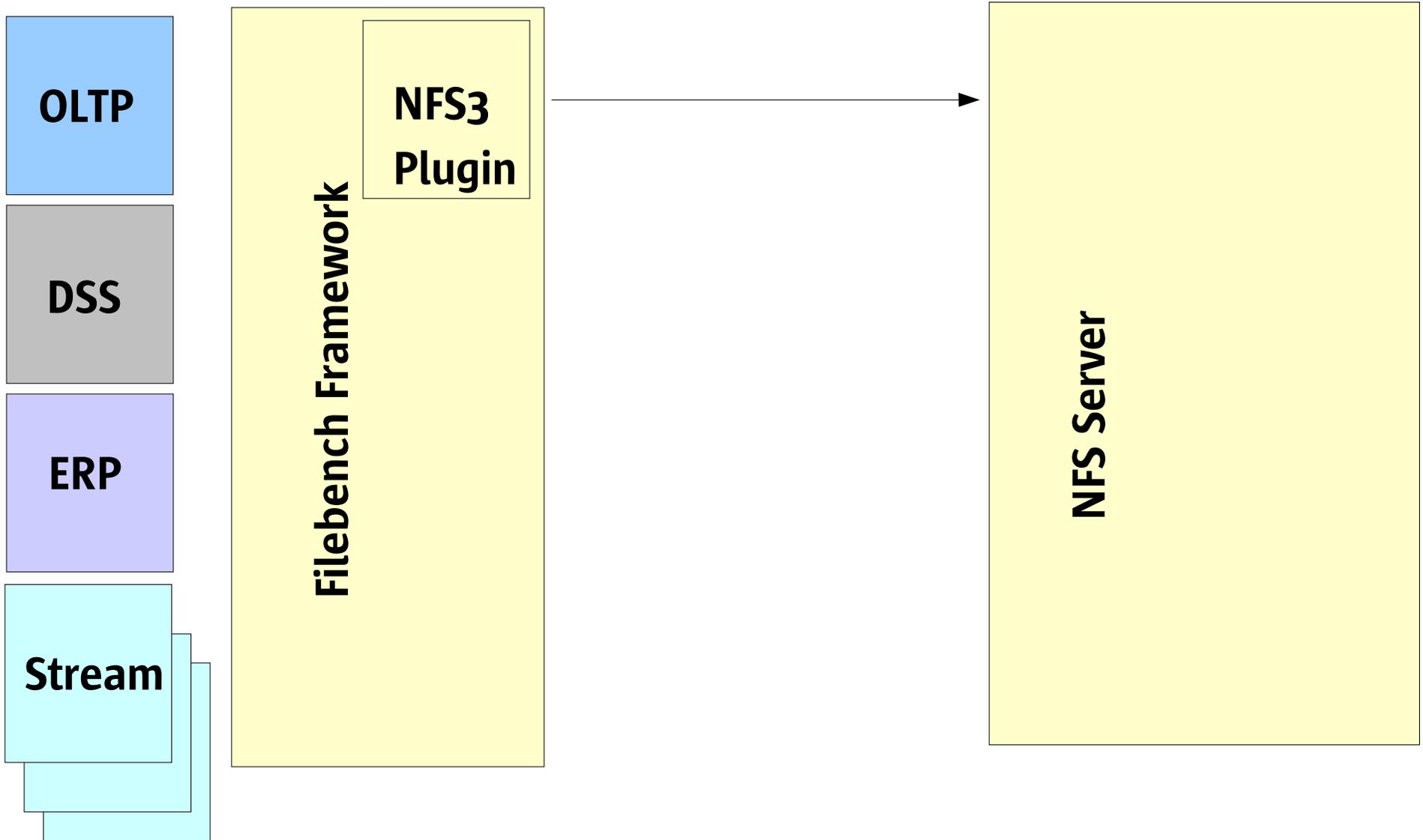
Filebench Achitecture



NFS Client Testing



NFS Client Testing



Running filebench...



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
```

Example Performance Comparison

- Throughput:

	operations/s		
	FS-A	FS-B	
copyfiles	1403	1431	+2.0%
createfiles	2433	2438	+0.2%
deletefiles	778	833	+7.1%
fileserver	4264	2202	-48.4%
oltp	16840	866	-94.9%
randomread	78	37	-53.3%
singlstreamread	35	36	+2.9%
multistreamread	50	60	+20.0%
varmail	2231	5591	+150.6%
webproxy	7781	2255	-71.0%
webserver	1885	2901	+53.9%

Example Performance Comparison

- Client Microseconds per operation:

	uSec/op		
	FS-A	FS-B	
copyfiles	1076	2294	2.1x
createfiles	2131	8952	4.2x
deletefiles	1001	1999	2.0x
fileserver	3152	24994	7.9x
oltp	586	13557	23.1x
randomread	742	2329	3.1x
singlestreamread	16553	27372	1.7x
multistreamread	18001	25032	1.4x
varmail	1078	3168	2.9x
webproxy	4242	22418	5.3x
webserver	1247	10660	8.5x

Filebench Status

- Porting Status
 - S8, 10, x86, SPARC, Linux (2.6/Fedora)
- Early access workload models
 - Random Read/Write (Random block I/O)
 - Sequential I/O (single or multi-stream block I/O)
 - OLTP Database (Oracle Emulator)
 - File Server (Multi-user file intensive)
 - Varmail (Postmark style /var/mail emulation)
 - Webserver (Multi-threaded read + sequential weblog)
 - Webproxy (Multi-threaded read, create, write, delete)
 - Copyfiles (Copy a file tree)

From here...

- Develop more workloads
 - Validation is key
 - Collaborate with workload experts for validation
 - Open up framework
 - Further develop client/server separation/synthesis
- Sun + 1 other vendor collaborating
- Investigating community development
 - Framework development
 - Workload development



Comments?

r@sun.com

