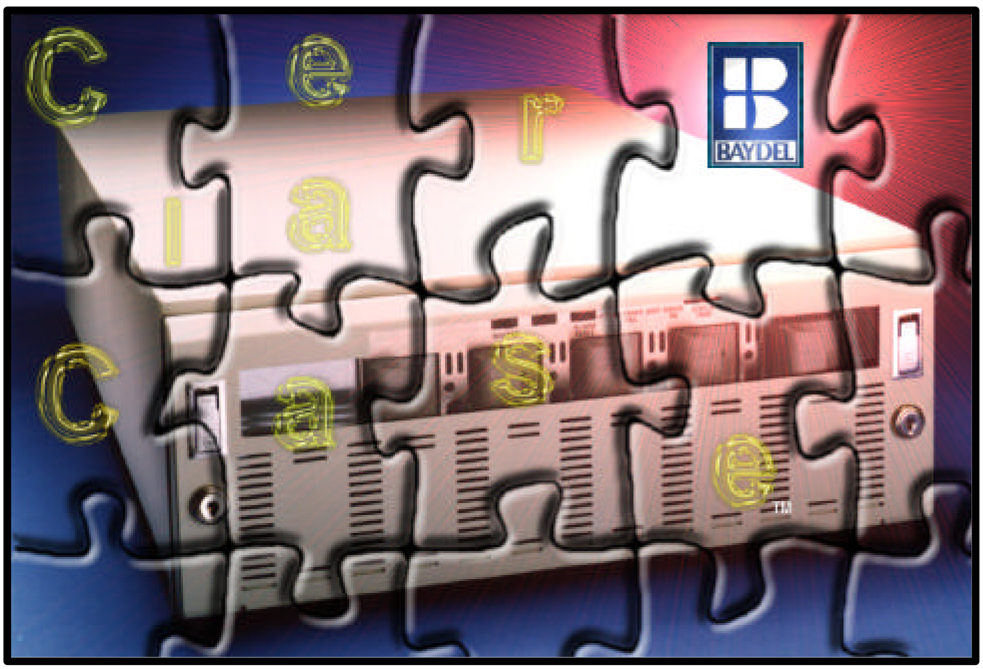
Understanding ClearCase I/O Performance and It's Demands on Storage



Baydel Ltd.
Brook Way, Leatherhead
Surrey KT22 7NA
United Kingdom

Tel: 44 (0) 1372 378811
FAX: 44 (0) 1372 386960

Baydel North America, Inc.
2370 Bering Drive
San Jose, CA   95131
U.S.A.

Tel: (408) 383 7474
FAX: (408) 383 9501

http:\\www.baydel.com

TABLE OF CONTENTS

Introduction

Statement of Purpose

The area of High Performance and ClearCase are not often associated together. However, I/O Subsystem performance can play a major role in overall ClearCase application performance. This paper examines the I/O workload ClearCase puts on an actual disk subsystem (from Production ClearCase installations) and uses this information to outline what I/O characteristics will yield the best ClearCase application performance. ClearCase Versioned Object Base (VOB), View, and Distributed Defect Tracking System (DDTS) will be examined.

Brief Overview of the ClearCase Application

ClearCase is a Software Configuration Management System. ClearCase provides a mechanism for managing change in a software development environment to ensure quality and maintainability. It keeps track of which versions of which files were used for each release and which combinations were used in software builds.

The heart of ClearCase is a permanent, secure Data Repository. Developer work environments are created by extracting, or “Check-Out”, source file versions from a central database and placing them in a private work area. This work area is unique to a defined configuration (or view). All enhancements and bug fixes are made in this work area which is outside of the tools control. Once the changes are made, the developer “checks in” the affected files.

ClearCase becomes part of the operating system, so it has complete control over all access to files it manages. ClearCase managed directory structures control which files are seen based on rules and a point in time (with the current moment being the default). One can go back in time and see different files, or one can use different rules and see different software releases.

ClearCase Data Storage

ClearCase data is stored in VOB’s and VIEW’s which can be located on any, or all, hosts where ClearCase is installed. A VOB or View can have auxiliary data storage on non-ClearCase hosts; such storage is accessed through standard symbolic links.

The networks permanent data repository (or VOB) is conceptually a central resource. Typically the repository is distributed among multiple VOBs on multiple hosts.

ClearCase Application Architecture

ClearCase is a distributed application with a client-server architecture. A ClearCase environment typically has the following host classifications:

- Network Wide Release Host
- License Server Host(s)
- Registry Server Host
- Client Hosts
- Server Hosts
- Non-ClearCase Hosts

For the purposes of this discussion the following are further defined.

***<u>Server Hosts</u>*:**

Some hosts may only be used as data repositories for VOB storage directories (i.e. a VOB Server) and/or view storage directories (i.e. a View Server). Such hosts run ClearCase server programs only***:*** *albd_server, vob_server, view_server* and other programs installed in: /usr/atria/etc.

<u>Non ClearCase Hosts:</u>

ClearCase may not be installed on every host. In fact, dedicated NFS file servers, or filers, may not even be possible to install ClearCase (As these hosts are not supported by ClearCase). Although such hosts do not run

ClearCase programs, they can access ClearCase data via standard UNIX network file system facilities "EXPORT" or "SHARE". Mechanisms of these non-ClearCase hosts are accomplished via standard UNIX tools.

Versioned Object Base (VOB) Storage

Brief Overview of the VOB

The permanent data vault of ClearCase is termed a VOB. The VOB contains data shared by all developers, including:

1. Current and historical versions of source files.
2. Derived objects built from the sources by compilers, linkers and others.
3. Detailed accounting information on the development process itself, such as who created a particular version or what versions of source went into a particular release.

A VOB is a data structure that can be mounted as a file system of type MVFS (Multi Version File System) and which is used in ClearCase to store all source code, binary objects and other versioned objects under configuration management.

VOB Server Host Memory

A host on which one or more VOB storage directories reside is termed a: VOB host/server. VOB server performance is very dependant on host memory and disk write speeds.

Main memory in the VOB host is one of the most important factors in VOB performance. Increasing the size of the VOB servers main memory can be the easiest (And most cost effective) way to make VOB access faster and/or to increase the number of concurrent users without degrading performance.

ClearCase is based on the dbVISTA database engine (From RAIMA Corporation) which can be the Achilles heel in terms of performance. The dbVISTA database files are cached or accessed using the File System. Therefore, it is generally recommended that the VOB server be on a dedicated machine (often times requiring multiple VOB servers) as a general rule of thumb, host memory should equal ½ of the
Database size. One can simply perform:

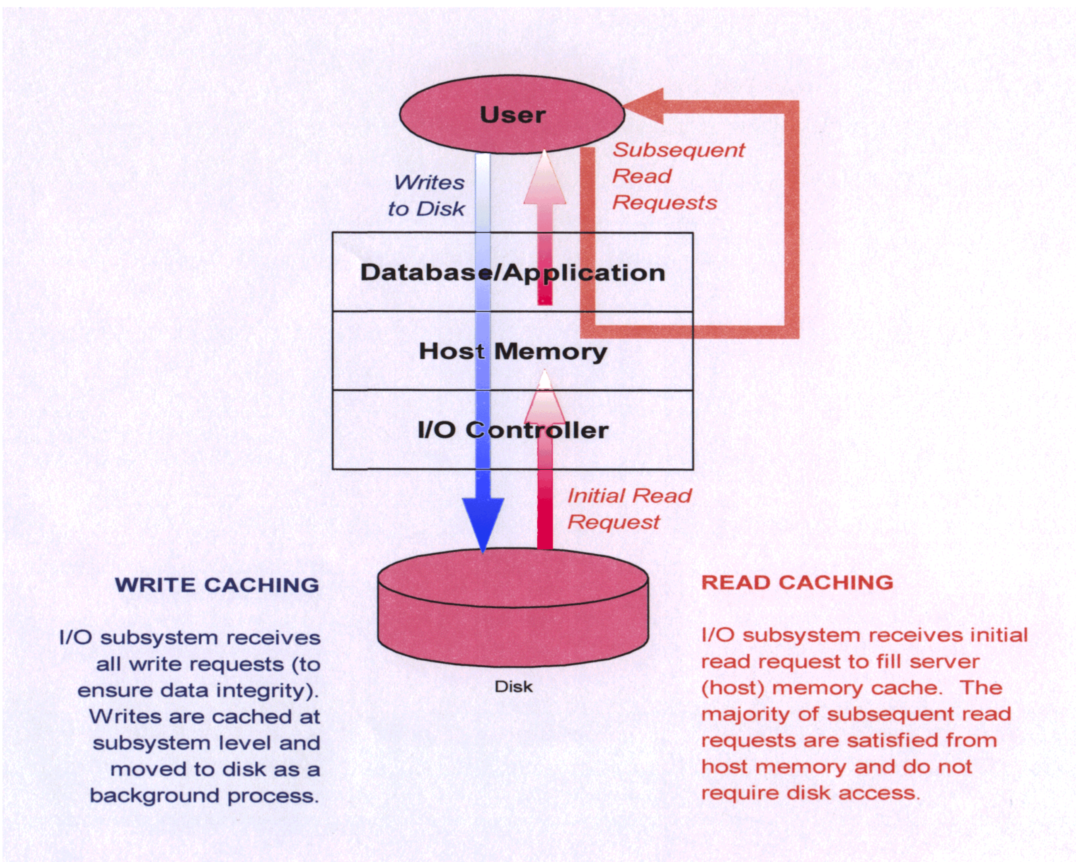
(du –s on the directory) / 2 = amount of memory for machine

In A Perfect World

It is not uncommon in today's server configurations to see memory in excess of 1 (one) Gigabyte. In a simplistic sense, this memory performs one function: **READ CACHING.** In a "Perfect World," the server memory will do all of the read caching and the I/O subsystem will do all of the write caching.

Why read cache at the host level? Simple: Read performance from server memory cache will be better than read performance from disk subsystem cache because of the additional latencies imposed on a transaction to and from a disk subsystem. In short, to put a data packet onto an I/O bus there is additional protocol processing (i.e. SCSI) that is not required if that same data request can be satisfied directly from the server memory.

Why implement write cache at the disk level? For recoverability and exception handling. If write data is being cached in server memory, it becomes difficult to recover the data in the event of a system crash, etc. Furthermore, most server configurations do not have built in resiliency mechanisms for power failures or cache memory failures.



Interaction of Host Cache with Disk Cache

Given the "Perfect World" scenario (see diagram), when the host server is doing effective read caching the percentage of read operations on the disk subsystem will be lower than the percentage of write operations. Poor read caching or lack of sufficient cache resources in the server can cause unnecessary disk read operations. Indicators of poor host read cache efficiency can be derived by monitoring and reporting physical activity from the actual disk subsystem.

Satisfying Write Requests at Memory Speeds Without Buying All That Memory

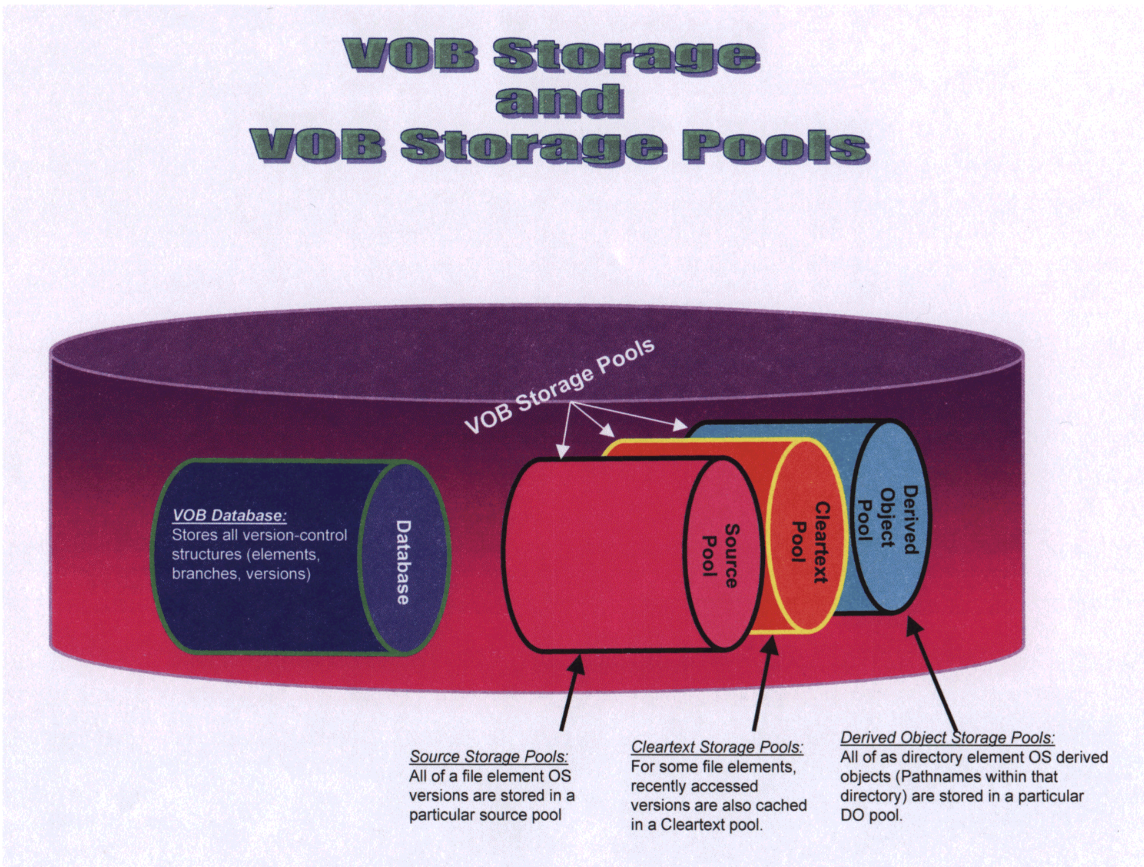
In order for a disk subsystem to consistently satisfy and sustain memory speed response times, a simple rule applies: *The data structure on the disk devices must be able to pull data from the back of the cache (i.e., flush the cache) and must be faster than the input structure*. (Given two hoses in a bucket, one putting water into the bucket and the other taking water out, as long as the hose taking water out is larger than the hose putting water in, the bucket will not overflow.)

If the host input channel into the cache is Ultra SCSI bus, then the best case (spec) input rate would be approximately 40 MB/sec (assuming very large application I/O size, i.e., a 1 MB I/O). For the sake of argument we will use 40 MB/sec as our input channel capabilities. Now the transfer rate of a single disk drive is dictated by the rate at which data actually travels by the disk heads (the "Head To Media Rate"). With a single Seagate Cheetah 18, the head to media rate is approximately 10 to 12 MB/sec. Therefore, a single Cheetah 18 cannot provide an adequate output channel to sustain data flow faster than the host input channel. This structure would only satisfy writes at memory speeds up until the point of memory full.

If four, Cheetah 18 drives were synchronized, the head to media rate for the set of four drives becomes 40 to 48 MB/sec. Four synchronized Cheetah 18 drives can thus provide a faster output channel than the host input channel.

VOB Server Storage

There are four major components of a VOB Storage:
- The Database
- The source pool(s)
- Cleartext Pool(s)
- Derived Object Pool(s)



The above components are referred to as the "storage pools" within ClearCase.
Each VOB has its own database, there is no database that encompasses all VOBs.

ClearCase Data Integrity Requirements

- Source pool(s) are very important data – therefore data integrity requirements are high
- Cleartext pools contain the least critical data – therefore data integrity requirements might be lower. Cleartext pools generally have the heaviest traffic of all the pools
- Derived Object Pool(s) – Data integrity requirements are high due to loss of a pool can cause a loss of synchronization with the Derived Object Catalog in the VOB database.

The Impact of Disk Write Speed

ClearCase tends to be very write intensive at the disk level… it is not uncommon to see 1 to 5 or 1 to 10 ratio of reads to writes. Operations that write to the VOB database include, but not limited to:

1. Checkin
2. Checkout
3. Clearmake and Omake
4. Creating meta-data
5. VOB scrubbing

Lessons Learned Locating ClearCase VOB Storage Pools on the Baydel Raid

Production VOB's With Journaled (extent-based) Filesystems

Baydel Client name:	**Cisco - Sun with Veritas VXFS Filesystem**									***DATE:***	**September 8, 1999**
Hardware	Applications	Company	Device "given" name of Raid	HOST I/O Average Read size	HOST I/O Average write size	I/O Profile:Read I/O's	I/O Profile: Write I/O's	DERIVED INDICATORS: Read Cache Efficiency	DERIVED INDICATORS: Write commands Cached	**Total Overall Cache Efficiency in %**	
Sun Ultra2	ClearCase	Cisco	bfr-vob2-raid	10.00	16.00	0.79%	99.21%	31.77%	100.00%	99.46%	
Sun E450	ClearCase	Cisco	cor-vob5	17.00	12.00	48.35%	51.65%	23.03%	99.96%	62.76%	
Sun Ultra2	ClearCase	Cisco	dori	58.00	6.00	14.08%	85.92%	50.18%	99.65%	92.68%	
Sun	ClearCase	Cisco	hubbel	10.00	8.00	19.23%	80.77%	30.12%	100.00%	86.56%	
Sun E6000	ClearCase	Cisco	lrp-vob1	29.00	13.00	29.06%	70.94%	27.35%	99.88%	78.80%	
Sun E4000	ClearCase	Cisco	itech-vob1	27.00	13.00	27.47%	72.53%	23.66%	99.96%	79.00%	
Sun E4500	ClearCase	Cisco	nogir	25.00	7.00	12.59%	87.41%	43.40%	99.42%	92.37%	
Sun Ultra2	ClearCase	Cisco	nori	45.00	6.00	17.81%	82.19%	41.57%	99.80%	89.43%	
Sun Ultra2	ClearCase	Cisco	ori	77.00	7.00	9.67%	90.33%	65.72%	99.84%	96.54%	
Sun	ClearCase	Cisco	splob-vob1	27.00	13.00	22.88%	77.12%	23.95%	99.19%	81.98%	
	Overall Averages			**32.50**	**10.10**	**20.19%**	**79.81%**	**36.08%**	**99.77%**	**85.96%**	

Production VOB's With UFS (Indirect-Block Based) Filesystems

Baydel Client name:	**ClearCase - Sun with UFS Filesystem**								***DATE:***	**Sept. 8, 1999**
Hardware	Applications	Company	Device "given" name of Raid	HOST I/O Average Read size	HOST I/O Average write size	I/O Profile:Read I/O's	I/O Profile: Write I/O's	DERIVED INDICATORS: Read Cache Efficiency	DERIVED INDICATORS: Write commands Cached	**Total Overall Cache Efficiency in %**
Sun SS 20	ClearCase	Aspect		12.00	9.00	48.00%	52.00%	16.83%	99.98%	60.07%
Sun E450	ClearCase	Escalade	ttya	7.00	10.00	43.99%	56.01%	62.65%	99.45%	83.26%
Sun E450	ClearCase	Escalade	ttyb	7.00	9.00	27.26%	72.74%	60.37%	99.50%	88.83%
Sun SS20	ClearCase	Scopus		20.00	8.00	1.65%	98.35%	54.16%	99.96%	99.20%
Sun SS20	ClearCase	Scopus		9.00	11.00	8.79%	91.21%	58.93%	100.00%	96.39%
Sun SS20	ClearCase	Scopus		10.00	13.00	27.84%	72.16%	21.46%	99.90%	78.06%
HP	ClearCase	Wells Fargo	VOB	8.00	8.00	78.40%	21.60%	17.43%	99.97%	35.26%
Sun E4000	ClearCase	Xerox	term0	11.89	13.00	51.00%	49.00%	26.00%	100.00%	62.26%
Sun E4000	ClearCase	Xerox	term1	13.07	10.21	56.00%	44.00%	32.00%	100.00%	61.92%
Sun E4000	ClearCase	Xerox	term2	13.38	10.72	47.00%	53.00%	31.00%	100.00%	67.57%
Sun E4000	ClearCase	Xerox	term3	9.45	9.87	47.00%	53.00%	25.00%	100.00%	64.75%
Sun E4000	ClearCase	Xerox	term4	13.87	10.16	48.00%	52.00%	28.00%	100.00%	65.44%
Sun E4000	ClearCase	Xerox	term7	16.29	9.80	46.00%	54.00%	37.00%	100.00%	71.02%
Sun	ClearCase	Xerox	Wizard 3	14.00	12.00	35.54%	64.46%	20.55%	98.63%	70.88%
Sun	ClearCase	Xerox	Wizard 4	15.00	12.00	34.66%	65.34%	29.17%	99.12%	74.88%
	Overall Averages			**12.00**	**10.38**	**40.08%**	**59.92%**	**34.70%**	**99.77%**	**71.99%**

Conclusions – VOB Storage

VOB Server Memory and the Underlying Filesystem

There is ample evidence that the overwhelmingly most important factor in VOB server performance is sufficient RAM. However, not only is the amount of RAM important, but also is the underlying filesystem. Some filesystems, like the Veritas VXFS file system, use the available RAM more efficiently therefore yielding overall higher read cache effectiveness in the server. Higher effectiveness of reads satisfied from file system buffer cache results in fewer disk read I/O operations and therefore higher percentages of write activity on the disk subsystem.

Real World Observation:
The above data shows that production VOB I/O requests show higher percentage of overall I/O read requests for the UFS filesystem than for the Veritas VXFS journalled filesystem:

Filesystem Type	**Statistics From VOB Servers In Production** **% Of Overall I/O Requests Which Are Reads**
UFS Filesystem	**40.08%**
Veritas VXFS Journalled Filesystem	**20.19%**

Journaled File Systems (extent-based)

Journaled file systems like Veritas, VXFS (Sun & HP), JFS (HP), XFS (SGI) can provide significantly better NFS file server performance and the fastest filesystem recovery on power loss (no FSCK required).

From a disk subsystems view journaled (extent-based) filesystems reduce the number of physical disk read requests and therefore further increase the requirement for high speed writes. Indirect file systems (Such as Sun's UFS) typically breaks files into 8k byte blocks that can be spread all over the disk. Additional 8k byte indirect blocks keep track of the location of the actual data blocks. In subsequent sequential read of the file, the disk system has to keep seeking (Reads) to pick up indirect blocks and scattered data blocks. This seeking procedure increases the number of physical disk reads.

In contrast, extent-based file systems keep track of data using extents. Each extent contains a starting point and size. If a 2 gigabyte file is written to an empty disk, it can be allocated as a single 2 gigabyte extent. There are no indirect blocks (To be read) and a sequential read of the file reads an extent record, then reads the data for the complete extent.

Pool Splitting and Disk Tuning

There exists a school of thought that suggests there may be performance benefits by placing pools on separate disks (or on separate servers, such as Network Appliance) tuned to optimize for different access patterns. Through the above data collected, the access patterns are as follows:

Database

The VOB database accesses are overwhelmingly writes, and 4 Kbytes in size. There is some thought that the distribution of these writes are randomly placed. A random write distribution would cause the Baydel write cache statistics to be substantially < 100%. Based on above observed production data, there appears to be a higher locality of reference.

Pools

The VOB pools are strongly write-biased at the disk level (though at the user level they are strongly read-biased, most of the reads are satisfied from the file system buffer cache). I/O request sizes vary from 512byte, 8 Kbyte, 16 Kbyte, and 64 Kbytes sizes.

Observation:
The above evidence suggests that performance improvements arise from the combination of a journaled based filesystem coupled with fast disk I/O (or write cache). The Network Appliance is architected with the WAFL (write anywhere file layout) journalled file system combined with the use of NVRAM for write acceleration.

To further illustrate the impact of the underlying filesystem on performance the following tests were performed.

Configuration
Host: Sun Ultra 10 – 333 Mhz
Memory: 512 MB
Operating System: Solaris 2.6
Disk: Baydel 68GB Ultra SCSI Disk Array with 64MB cache
Filesystems: Standard Sun UFS file system, Veritas VXFS filesystem

The Baydel IOP2 program was used to generate completely random 8K write requests to both the standard Sun UFS filesystem and the Veritas VXFS filesystem.

The following results were measured

Sun UFS

I/O's Per Second	68
MB Per Second	0.563636

Veritas VXFS

I/O's Per Second	286
MB Per Second	2.349015

The above illustrates that even with the worst case scenario (completely random writes with no locality of reference) the underlying filesystem impacts write performance.

System Tools for Gathering Profile Data - The Solaris 2.5 Trace Capability

The performance and statistical data in the previous sections was provided from Baydel Disk Arrays in production VOB environments. Since this level of I/O granularity is not available to all environments, the following discussion is provided to give some level of visibility into the I/O profile in Sun Solaris environments. Clearcase VOBs in Sun Solaris environments can gather some profile information using the *prex* and *tnf* kernel trace capabilities. For a detailed discussion on these features, see: Sun Performance and Tuning, Second Edition by Adrian Cockcroft and Richard Pettit Pages 188-194. This following discussion is intended to provide a starting point.

The trace functionality provides the ability to trace I/O. The following excerpts are from the book "Sun Performance and Tuning." The trace functionality was introduced in Solaris 2.5 and consists of the following features:

- A program **prex(1)** controls probe execution for user and kernal traces.
- A program **tnfxtract(1)** reads out the kernal trace buffer, and **tnfdump(1)** displays the TNF data in human-readable ASCII

The command sequence to initiate an I/O trace is quite simple

\# **prex –k**
Type "help" for help …
prex> **buffer alloc**
Buffer of size 393216 bytes allocated
prex> **enable io**
prex> **trace io**
prex> **ktrace on**

Now, wait awhile or run the program you want to trace

prex> **ktrace off**

Now the data can be extracted and dumped for viewing

\# **mkdir /tmp/tnf**
\# **cd /tmp/tnf**
\# **tnfxtract io.tnf**
\# **tnfdump io.tnf | more**

Here are the simple interpretations:
The “strategy” probe records an I/O being issued
The “biodone” records an I/O completing
The “pageout” the pageout scanner may also cause an I/O

The following provides sample output, this will provide some visibility into read / write ratio and size of the individual I/O requests.

```
probe       tnf_name: "biodone" tnf_string: "keys io blockio;file ../../common/os/bio.c;line 1010;"
probe       tnf_name: "pageout" tnf_string: "keys vm pageio io;file ../../common/vm/vm_pvn.c;line 516;"
probe       tnf_name: "strategy" tnf_string: "keys io blockio;file ../../common/os/driver.c;line 377;"
---------------- ---------------- ----- ----- ---------- --- ------------------------ ----------------------
    Elapsed (ms)       Delta (ms)   PID LWPID    TID     CPU Probe Name               Data / Description . .
---------------- ---------------- ----- ----- ---------- --- ------------------------ ----------------------
        0.000000         0.000000     0     0 0xf61a1d80   0 strategy                 device: 8388632
                                                                                       block: 795344
                                                                                       size: 8192
                                                                                       buf: 0x5afdb40
                                                                                       flags: 9
        0.166472         0.166472     0     0 0xfbf6aec0   0 biodone                  device: 8388632
                                                                                       block: 795344
                                                                                       buf: 0xf5afdb40
```

The first strategy routine is an 8-Kbyte write (flags bit 0x40 is set for reads) to block 795344. To decode the device, 8388632 = 0x800018, 0x18 = 24, and running ls –lL on /dev/dsk shows that the c0t3d0s0 is minor device 24, which is mounted as /export/home.

VIEW SERVER

Brief Overview of the VIEW

A ClearCase development environment can include any number of views.   Each view represents a virtual workspace to the developer.   The view is intended to be where actual work takes place.

Short term storage for data created during the development process is provided by ClearCase Views.   A view stores checked-out versions of file elements, view private files, and newly derived objects.

Developers may consider views and VOBS as very different; A VOB is where the data resides, the view is a “Lens” through which a developer sees data. From an I/O workload, VOBS and Views have similar characteristics.

Views maintain their own caches. The goal with view caches is to reduce physical VOB (Network Traffic and fileserver accesses) and hence, read disk I/O accesses.   The more efficient the view cache, the more write intensive the drive workload.   Views have both a database portion and a storage portion called the “.S” pool.

Lessons Learned Locating ClearCase View Storage Pools on the Baydel Raid

Production VOB's With Journaled (extent-based) Filesystems

Baydel Client name:	**Cisco**									**DATE:**	**Feb, 10 2000**
Hardware	Applications	Host Memory in MB	Device "given" name of Raid	HOST I/O Average Read size	HOST I/O Average write size	I/O Profile:Read I/O's	I/O Profile: Write I/O's	DERIVED INDICATORS: Read Cache Efficiency	DERIVED INDICATORS: Write commands Cached	**Total Overall Cache Efficiency in %**	
Sun	View Server		sagan-stats	25.00	18.00	34.20%	65.80%	55.44%	99.99%	84.75%	
Sun	View Server		sagan-stats2	32.00	24.00	8.37%	91.63%	70.88%	100.00%	97.56%	
Sun	View Server		sagan-stats3	26.00	18.00	39.05%	60.95%	63.31%	99.99%	85.67%	
Sun	View Server		sagan-stats3a	22.00	16.00	17.48%	82.52%	62.12%	100.00%	93.38%	
Sun	View Server		sagan-bottom	23.00	17.00	20.32%	79.68%	59.21%	99.99%	91.70%	
Sun	View Server		sagan-top	22.00	15.00	22.95%	77.05%	58.75%	100.00%	90.53%	
Sun E6000	View Server		itech-view3	8.00	10.00	38.94%	61.06%	19.51%	99.91%	68.60%	
							Overall Raid Cache Effectiveness			**90.60%**	

Baydel Client name:	**Customer**	**Wells Fargo - VIEW**							***DATE:***	**11-May-99**
Hardware	Applications	Host Memory in MB	Device "given" name of Raid	HOST I/O Average Read size	HOST I/O Average write size	I/O Profile:Read I/O's	I/O Profile: Write I/O's	DERIVED INDICATORS: Read Cache Efficiency	DERIVED INDICATORS: Write commands Cached	**Total Overall Cache Efficiency in %**
HP	VIEW			11.00	8.00	40.85%	59.15%	21.32%	100.00%	67.86%
							Overall Raid Cache Effectiveness			**67.86%**

Build Servers

Build Avoidance

The best build performance comes from avoiding a build all together. ClearCase provides a make-compatible build system called ClearMake. ClearMake can determine that the derived object from one engineer or a nightly build depends upon all the same versions of files as another engineer is using. When that is the case, no compile is necessary, and the derived object is made available to both engineers. It is made available into the workspace as though a compile had happened.

Baydel Client name:	**Cisco**								***DATE:***	**September, 8 1999**
Hardware	Applications	Host Memory in MB	Device "given" name of Raid	HOST I/O Average Read size	HOST I/O Average write size	I/O Profile:Read I/O's	I/O Profile: Write I/O's	DERIVED INDICATORS: Read Cache Efficiency	DERIVED INDICATORS: Write commands Cached	**Total Overall Cache Efficiency in %**
Sun E6000	ClearCase		build5-1	7.00	3.00	75.56%	24.44%	16.77%	99.93%	37.09%
Sun E6000	ClearCase		build5-2	5.00	13.00	73.99%	26.01%	68.81%	99.82%	76.88%
Sun	ClearCase		build6-1	24.00	17.00	38.77%	61.23%	47.76%	99.97%	79.73%
Sun	ClearCase		build6-2	24.00	15.00	27.39%	72.61%	48.64%	99.98%	85.92%
Sun	ClearCase		build6-3	21.00	4.00	19.48%	80.52%	43.34%	99.97%	88.94%
Sun	ClearCase		nm-build2	85.00	21.00	22.55%	77.45%	72.04%	99.99%	93.69%
Sun	ClearCase		rtp-build1	29.00	12.00	17.08%	82.92%	40.29%	100.00%	89.80%
							Overall Raid Cache Effectiveness			**78.86%**

The following performance information was gathered **only during the Build Process**

Baydel Client name:	**Customer**	**Network Equipment Technology - View Server Builds**							***DATE:***		**March, 24 1999**
Hardware	Applications	Host Memory in MB	Device "given" name of Raid	HOST I/O Average Read size	HOST I/O Average write size	I/O Profile:Read I/O's	I/O Profile: Write I/O's	DERIVED INDICATORS: Read Cache Efficiency	DERIVED INDICATORS: Write commands Cached	**Total Overall Cache Efficiency in %**	
Sun Ultra 1	View Server - Clearmake	128MB		7.83	8.73	0.0011%	99.89%	75.69%	100.00%	99.89%	
							Overall Raid Cache Effectiveness			**99.89%**	

Baydel Client name:	**Voice Mail Company - Clearcase Builds**								***DATE:***	**27-Apr-98**
Hardware	Applications	Host Memory in MB	Device "given" name of Raid	HOST I/O Average Read size	HOST I/O Average write size	I/O Profile:Read I/O's	I/O Profile: Write I/O's	DERIVED INDICATORS: Read Cache Efficiency	DERIVED INDICATORS: Write commands Cached	**Total Overall Cache Efficiency in %**
Sun E4000	Clearcase			9.00	11.00	1.71%	98.29%	51.61%	99.98%	99.15%
Sun E4000	Clearcase			9.00	12.00	1.43%	98.57%	45.81%	99.98%	99.21%
							Overall Raid Cache Effectiveness			**99.18%**

DDTS (Distributed Defect Tracking System)

Rational Software's bug tracking software companion to ClearCase is called DDTS.

Baydel Client name:	**Customer**	**Cisco - DDTS Bug Tracker (Atria)**							***DATE:***	**Oct, 16 1998**
Hardware	Applications	Host Memory in MB	Device "given" name of Raid	HOST I/O Average Read size	HOST I/O Average write size	I/O Profile:Read I/O's	I/O Profile: Write I/O's	DERIVED INDICATORS: Read Cache Efficiency	DERIVED INDICATORS: Write commands Cached	**Total Overall Cache Efficiency in %**
Sun E4000	DDTS	4GB		12.00	8.00	42.35%	57.65%	35.10%	100.00%	72.51%
							Overall Raid Cache Effectiveness			**72.51%**

ClearCase Backup Methodologies in use on the Baydel RAID

Three basic schemes are in use for ClearCase Backup on the Baydel RAID:

1. No Mirroring
2. Mirroring
3. Veritas Snapshot

1. No Mirroring

In this case the VOB database and storage pools are stored on the Baydel RAID. The steps required are:

- Lock the VOB
- Backup VOB Database, Source Pool and Derived Object Pool as a single, consistent data set.
- Unlock the VOB

2. Mirroring

In this case the VOB Database and Storage Pools are located on the Baydel RAID. Standard drives (No reduncancy or RAID) are used as the "Mirror Volume". The steps required are:

- Establish mirror to standard drive volume
- Lock the VOB
- Split off mirror volume
- Unlock the VOB
- Backup from Detached Mirror Volume on standard drives.

> NOTE:
>
> *With VOB Storage on the Baydel RAID the VOB still maintains redundancy during backups, even while the mirror is detached.*

3. Veritas Snapshot

In this case the VOB Database and Storage Pools are located on the Baydel RAID with the Veritas VXFS as the underlying file system. The steps required are:

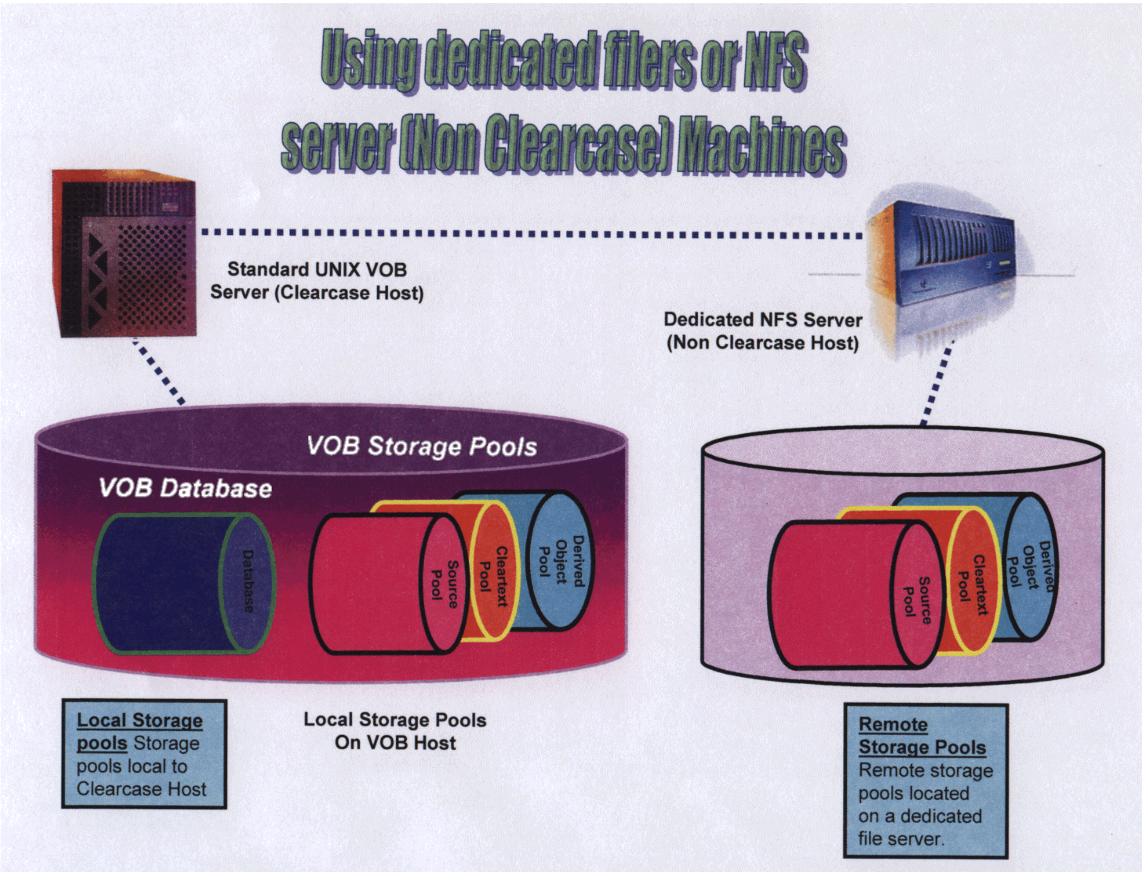
- Lock the VOB
- Create Veritas Filesystem snapshot. Snapshot destination device can be the Baydel RAID or standard drives.
- Unlock the VOB
- Backup from the snapshot.

ALTERNATE METHODS FOR IMPROVING ClearCase PERFORMANCE.

Two commonly recommended methods of increasing VOB performance are the use of dedicated NFS servers/appliances and the use of ASYNC filesystem I/O options.

Using Dedicated filers or NFS Server (non-ClearCase) Machines

The VOB database must be located on the machine in which the ClearCase database process run, this ensures the integrity of the database.



The VOB server process

Most access to VOB storage pools goes through a ClearCase server program, the vob_server. This process handles data access request from clients, forwarded to it by the VOB's db_server and vobrpc_server processes. As with other servers, a vob_server runs on the host where the corresponding VOB storage directory resides. Each VOB on a host has its own dedicated vob_server process.

Since dedicated NFS filers do not actually run any programs, ClearCase processes like vob_server and aldb_server cannot be executed on specialized appliances. All access to storage pools on the dedicated filer must go through the VOB server machine, therefore two machines are required to make up a VOB server.

Backups – Using Dedicated filers or NFS Server (Non-ClearCase) Machines.

As an organizations “Family Jewels”, VOB’s should be backed up reliably and frequently. ClearCase does not include data backup tools.

It is important to note that the VOB needs to be backed up as a single consistent data set, while the VOB is locked. System administrators must be able to accomplish backups across all machines among which a VOB is distributed. In particular, the database and source pools must be backed up together, while the VOB is locked. Coordinating the backup of the VOB database and Storage Pools is a critical data-integrity issue and strongly discourages off-host pool storage.

General rule… database and source pools should reside on the same machine.

NOTE:

The Cleartext pool has no backup requirements and is generally left out of the backups.

Using Asynchronous NFS writes

Some operating systems (i.e. HP-UX, with the ASYNC option) support asynchronous NFS writes to release the client sooner, prior to placing client data in buffer cache, the operating system immediately reports the data as written to the client. This functionality is achieved by exporting filesystems in which a VOB resides with an asynchronous option. It is often recommended that when using such asynchronous options that the VOB server be on an integrated UPS to protect against the loss of data should the server experience a power failure.

Having file blocks cache on the server during writes poses a problem if the server crashes. The client cannot determine which RPC write operating completed before the crash, violating the stateliness nature of NFS. Therefore, asynchronous options writes should be SYNCHRONOUS, that is, file blocks that are written to NFS servers (In this case, the VOB) should not be cached within the server and should be written to the disk subsystem before the clients RPC write call can complete.

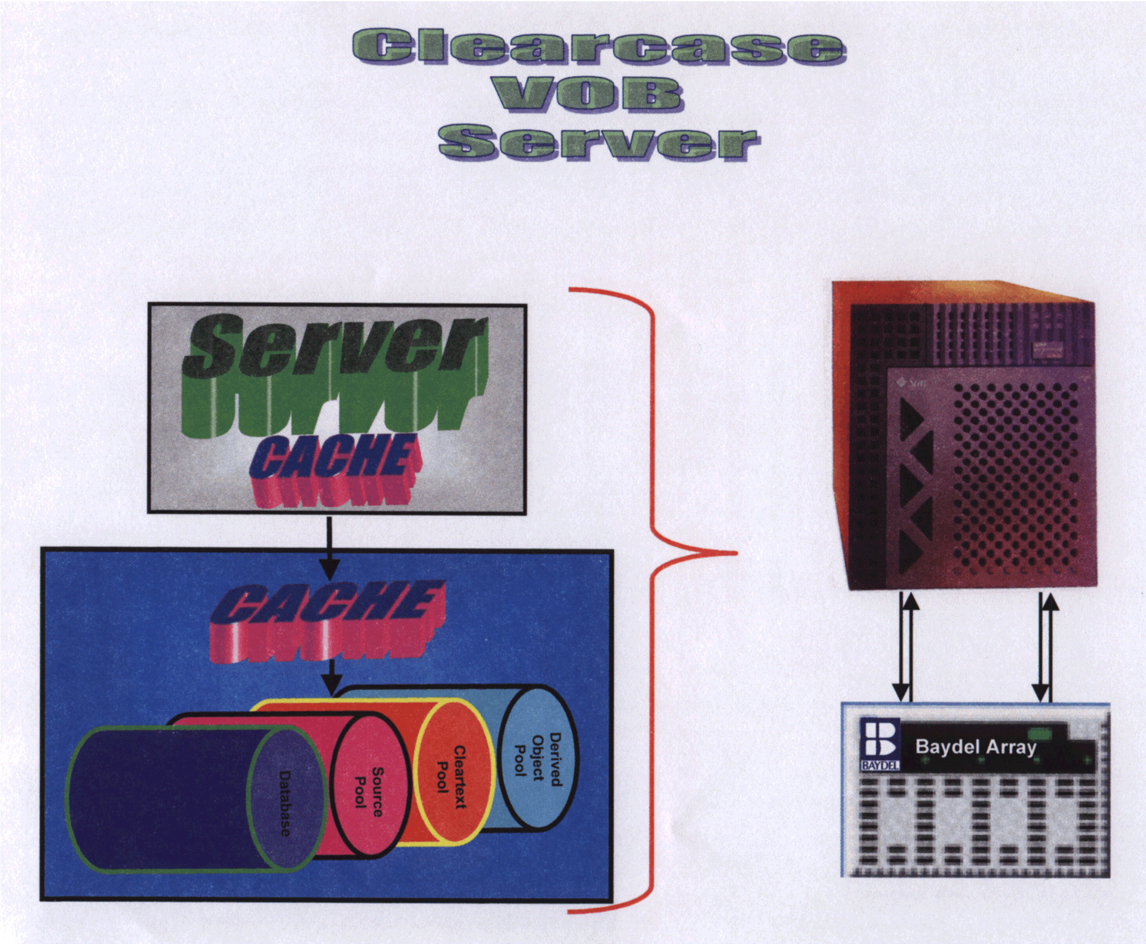
CONCLUSION
Benefits of locating VOB Storage Pools on the Baydel Raid

The following is summary of Baydel I/O performance in a ClearCase environment collected Baydel Raids in ClearCase environments. **The overall caching efficiency is 80.51%.** This is the overall percentage of ClearCase I/O operations that are satisfied from cache rather than disk regardless of: Host System, Host Memory, File System Type, or ClearCase Application (VOB, View, or DDTS server). The Baydel Raid represents a resilient, simple, effective mechanism for improving ClearCase I/O performance.

Hardware	*Applications*	*Overall Cache Efficiency (in %)*					*Customer Average*	*Customer Name*
SUN E450	VOB/View/DDTS	99.46%	62.76%	92.68%	86.56%	78.80%		
		79.00%	92.37%	89.43%	96.54%	81.98%		
		84.75%	97.56%	85.67%	93.38%	91.70%		
		90.53%	68.60%	37.09%	76.88%	79.73%		
		85.92%	88.94%	93.69%	89.80%	72.51%	***83.85%***	***Cisco***
Sun E4000	VOB/View	62.26%	61.92%	67.57%	64.75%	65.44%		
		71.02%	70.88%	74.88%			***67.34%***	***Xerox***
SUN SS20	VOB	60.07%					***60.07%***	***Aspect***
SUN SS20	VOB	99.20%	96.39%	78.06%			***91.22%***	***Scopus***
SUN E450	VOB	83.26%	88.83%				***86.05%***	***Escalade***
HP	VOB / VIEW	35.26%	67.86%				***51.56%***	***Wells Fargo***
SUN 4000	BUILDS	99.15%	99.21%				***99.18%***	***Voice Mail Co.***
SUN Ultra1	BUILDS	99.89%					***99.89%***	***NET***

Total Average **80.51%**

The following provides logical configuration outline for effective ClearCase performance with the Baydel Raid.



Topics for future research

1. Performance deltas.

- Timed Builds of source code
- "make clean" then "clearmake –v" of the source code.

For write intensive ClearCase operations like:

1. Clearmake
2. Make config
3. Builds
4. Relabel VOB
5. Reformat VOB

The Baydel storage will behave like a **solid state disk (SSD)**, in other words, write operations to the storage will take place at memory (Rather than disk) speeds within the storage.

2. Measure I/O impact of "Clean" (Recently Rebooted) systems or dirty systems.

3. Further refine I/O workload profiles for VOB database and the various storage pools (Derived Object Pool, Cleartext Pool, and Source Pool) by placing them on separate Baydel Arrays

4. Veritas VXFS file system tuning for optimal performance of the ClearCase Baydel combination.

5. Include section with actual performance test scripts and VOB Backup scripts to provide starting point templates for ClearCase Users.