Wide-field Infrared Survey Explorer (WISE)

WSDC Hardware System Design Peer Review Report

3-April-2009

Prepared by: T. Conrow, H. Brandenburg, R. Cutri

Infrared Processing and Analysis Center
California Institute of Technology

WSDC D-A005
-

1 INTRODUCTION

A peer review of the status of the compute hardware design was conducted by the WISE Science Data Center (WSDC) on March 19, 2009. The focus of this review was the planned design of the computing and storage hardware the are responsible for ingestion of raw WISE science and engineering telemetry and ancillary navigation files, pipeline data processing that converts raw WISE telemetry to calibration images, extracted source tables and metadata, and storage, management and back-up of the large volume of WISE data products. This review did not consider design of the long-term WISE data archive within the IPAC Infrared Science Archive from which final WISE data products will be served to the community.

This peer review was held as a result of a recommendation made by the review panel at the WSDC Critical Design Review in January 2008.

1.1 Review Panel Members

Wendy Burt (IPAC/Caltech)
Steve Groom (IPAC/Caltech)
Ingolf Heinrichsen (JPL/WISE)
Jason La Pointe (JPL/WISE)

1.2 Instructions for Review Panel

The peer review panel was asked to comment on the following specific questions*:*

- Is the WSDC processing system capable of handling our expected data volume and throughput?
- Are there any fragilities that might impede our ability to smoothly manage the ops system for the mission duration?
- Are the backup plans sufficient to recover from hardware outages?
- Please share any cluster or high-volume disk server lessons learned

In addition, comments on other aspects of the design were welcomed.

Written comments were received from Burt, Groom, Heinrichsen and LaPointe. These comments are summarized in Sections 2.1-2.4, below. WSDC responses follow each question or comment, and are shown in italicized blue text.

1.3 Applicable Documents

WSDC Functional Requirements Document (WSDC D-R001)
WSDS Functional Design Document (WSDC D-D001)

2 PANEL REPORTS

2.1 Wendy Burt

2.1.1 Backup:

The WISE backup plan is dependent on the IPAC shared backup infrastructure to complete the backup within a 6-hour window. As discussed at the peer review, testing needs to done to ensure the backup could be completed within the required window. The test should focus primarily on backing up (scanning) the number of files that need to be backed up, and not so much of the data volume.

Steve Groom suggested writing the files between backup cycles into a single logical file partition. This would greatly reduce the backup process time since it would eliminate the need to scan for files to backup; the backup process would simply back up the entire file partition. This would ensure the backup process is completed within the required 6-hour window.

2.1.2 Cross-mount:

Typically, when a system loses connectivity to one of its mounted filesystems, the system performance degrades because the system continually attempts to reconnect to the lost filesystem. The effect varies depending on the configuration. The best case scenario is there would be no performance impact. The worst-case scenario is the system might hang (freeze).

Each of the WISE cluster nodes mounts the local disks off all the other cluster nodes. When one of the nodes has a hardware failure, the overall cluster performance might degrade while the other node keeps trying to talk to the failed node. If testing has not already been done, it should be in order to determine whether performance will degrade (if any) on the remaining nodes.

2.1.3 Web server:

The current two-tier QA Web architecture could potentially expose the entire WISE data archive to malicious activity by exposing the entire data set to the Web server. It might be beneficial for the WISE project to look into a three-tier architecture, where a separate layer from the Web server has access to the data instead of the Web server. For example, the proposed SSC Heritage Archive architecture uses an application layer that sits in the INS, which the Web server (in PNS) talks to. This limits overall vulnerability of the design.

2.1.4 System utilization monitoring:

IPAC Systems Group is deploying Zenoss as its new monitoring tool, which also collects system utilization information. It might be beneficial for the WISE project to use Zenoss for utilization trending instead of maintaining its own tool.

2.1.5 Administration overhead:

The cluster administration was noted to be complicated to manage. It would be beneficial to document the configuration management process that is used for cluster administration.

2.2 Steve Groom

Below is a summary of questions and comments came up during the review. They are in no particular order. Some of the questions listed here were discussed or answered at the review, I include them here for reference and possible further consideration.

I have included some references to the presentation slides as HPR=”Hardware Peer Review” presentation package, HA=”Hardware Architecture” package.

2.2.1 Has ISG properly accounted for the network bandwidth consumed by the operational flow of WISE data through IPAC networks into the WSDC, and budgeted that usage against the usage by other IPAC projects during the WISE mission? Of particular interest here are the external network links from IPAC to the outside, where upgrades to network capacity might take more planning and cost to implement if needed. The answer seems to be that WISE demands on IPAC networks are well within the capacity of existing networks.

2.1.2 (HPR #12) – Derivation of throughput requirements shows daily time allotted for each activity. Doing this is a good idea to make sure the activities are doable. You should also recognize that it may be a “worst case” sort of time budgeting, possibly not taking advantage of opportunities to have some of these activities overlapped in time. “Pipelining” (overlapping) some of this activity, if practical and resources permit, may be a way to reduce the end-to-end time required to complete all processing steps.

2.2.3 Backup/Restore operations and filesystem organization – This is an area where I think more thought is needed, for several reasons, in subsections below:

2.2.3a Filesystems containing lots of "small" files pose a challenge for even well-designed backup systems. The time spent scanning the filesystem during the backup process tends to go way up as the number of files is increased, even for incremental backup cases when no files need to be backed up. Thus you don't want to have to backup such filesystems frequently, so it is helpful when filesystems with large numbers of small files are fairly static and don't require frequent backups.

2.2.3b It has been my experience that backup and restore times for a given filesystem can be highly asymmetric, namely it can take quite a bit longer to restore a filesystem than to back it up. This is contrary to a comment that was made during the review, where backups and restores were presumed to require about the same amount of time. The main difference between backups and restores lies in the extra time required to create each file during a restore operation, independent of the size of the file. Depending on the number and size of files being restored, the amount of time spent just creating files can overwhelm the time required to actually transfer the data from the restore medium back to disk. Estimations of time to recover from backups should expect this asymmetric behavior depending on file size characteristics.

2.2.3c If I understood correctly, WISE is presenting a design involving many filesystems, many or most of which change in some way or receive new data every day. This is an attempt to spread the I/O load over all available devices. However, this optimization of load balancing results in a layout that appears quite undesirable from a data management perspective, especially for backups. It means that every filesystem must be scanned for backups every day. While it spreads the load, it also spreads the exposure to changing data, and therefore the overhead and cost of doing backups, as well as the criticality of keeping them current.

It may be cleaner to consider having a selected number of filesystems that are growing at any one time, and once they fill, move on to filling other filesystems. Once "full", filesystems can be treated as read-only (or read-mostly), and backed up on a less frequent schedule according to the reduced rate of change. Another advantage of this model is that it allows a more incremental commitment of storage hardware, and makes it more seamless to add more filesystems&storage later should that be necessary.

It would seem that some compromise between spreading the I/O load (a good thing) and spreading the changing data widely (not a good thing) is needed. For example I can imagine a model with multiple logical filesystems per physical device, with only some of those filesystems actively filling/changing, while the others are stable/static. I/O load balancing is still achieved because the physical devices themselves are in use, but without the widespread data changes which require everything to be backed up frequently.

2.2.3d One last thought on backups, WISE should consider the level of staffing demands implied upon ISG to implement, operate, and monitor whatever backup scenario is envisioned.

2.2.4 The QA web application needs read-only NFS access to the main file store, to support QA functions. However, that QA data is sprinkled across all the operational filesystems, and thus the QA system will require access to every filesystem, and thus the entire set of WISE filesystems must be exposed to this web server, albeit read-only. Depending on how the QA web app and server are configured, this may present a risk of unwanted access to other contents of those fileystems besides just the QA data. For example if the QA web app provides some kind URL-based access to data files, that mechanism could potentially be exploited to access data other than just QA-related data. The risk here should be considered during the design of this function, and you might want to influence the organization/layout of QA data on those filesystems accordingly. I'm not sure how much of a risk or concern this presents, suffice to say that the entire filestore is being made visible to a web server which really only needs some of that data, and there should be some consideration of the tradeoffs in that decision.

2.2.5 Disaster recovery scenarios were discussed. A comment was made that the project doesn't know what disaster might befall them and therefore could not devise recovery strategies in advance. I would advise that the project pick a couple of representative example "disaster" scenarios at various levels of severity, and use those as the basis to outline a recovery strategy for each. Scenarios might include more common but less disastrous failures, as well as more complete outages. Some examples would be loss of data due to device failure, loss of data or entire filesystem(s) due to software or human error, and environmental scenarios such as water/fire/smoke/physical damage to components or the facility itself. Consider approaches for (a) prevention, (b) mitigation of impacts, (c) recovery.

2.2.6 HA#23: Ganglia charts such as this can be useful though they're really only the beginning for analysis. You can see that the system is waiting for data to come over the network, but you can't see what the limiting factor is – the network itself, or how fast the other end can send it. Need to deploy this everywhere if you can.

2.2.7 Just as there is a recognition that spreading the load over devices is a good thing for I/O, it is also valuable to consider how and when load can be spread in time. When a bunch of compute processors all come online and demand some data at the same time, suddenly networking and I/O channels (especially at the servers) can become saturated. It is helpful to spread the load in time when possible. Watch out for processing models which assume that many jobs are launched or completing at once, since this is when compute jobs typically demand a lot of I/O. Although the compute cluster allows distribution of the compute load over many systems, the communications and server-side I/O load are often not as tolerant of load spiking. Imagine a large number of systems such as the one graphed in HA#23, where they all want to do I/O at the same time. Not only is "the network" going to be a limiting factor, but the server-side resource is going to be slammed by this "rush hour". Avoid rush hour by staggering the load over time when practical. See earlier note about pipeline overlapping of processing stages.

2.2.8 HA#26. I wholeheartedly agree with the planned move to Solaris for file servers having a more mature set of filesystem and NFS server implementations. ZFS is also very good and

helping to lessen the criticality of finding I/O size "sweet spots", in that it should do a better job of caching of filesystem data.

2.2.9 The WISE archive will rely on access to the WISE file store for access to and service of image products. To my knowledge there has been little or no consideration for what load the archive services might impose on the file store, which will be used for both operations and archive services at the same time.

2.3 Ingolf Heinrichsen

Given that the NEO-WISE code is going to run as an integral part of the WISE pipeline on the same h/w cluster I have a concern that the requirements for NEO-WISE have not been finalized and the actual computing and network need is not established at this late stage.

There is no middleware buffering the exposed services from the WISE operational network from access from the outside (slide 15). Make sure there is a security audit of those services passing through the IPAC perimeter to minimize the exposure to attack (exact port/IP matching etc.)

Need to establish a clear process for ensuring identical configurations, when new machines are added to the cluster at a later stage and how to deal with required operating system security patches.

The use of SQLlight as a database management system could lead to unexpected bottlenecks (no row only table lock). Need to be sure scaling is understood to the size and transaction numbers expected for the WISE system.

The fact that the cluster was limited by network rather than CPU during the second half of the test shown on page 23 indicates some uncertainty on how the system will scale to larger cluster sizes. Probably need some more investigation on how this congestion is caused and whether there is a way to avoid it.

I talked to Don and Bill about the location/requirement for a Disaster recovery cluster and I think we have set-up a meeting next week.

2.4 Jason La Pointe

2.4.1 The team did an excellent job of incorporating APIs to collect pertinent information relating to run time, memory usage, etc.

2.4.2 The presented material shows that the team has given adequate thought to providing a hardware system that will meet nominal mission requirements. Not much was discussed

regarding extended mission. It would be worthwhile to make sure that that the selected architecture is scalable to meet extended mission goals.

2.4.3 Don't forget to set aside some disk space and nodes for sandbox processing. I suspect some members of the science team will want you to run experimental runs with different parameter settings. You'll need to make sure that this won't interfere with ongoing ops processing. Consider setting up a special directory tree and one or two dedicated nodes to handle the task.