FINAL IMPLEMENTATION OF AN IMPROVED OPC DATA LOGGING SYSTEM IN AN IN A AUTOMATION ENVIRONMENT

B.C. BOTHMA AND H.J. VERMAAK

Abstract

This paper will discuss the final implementation of an Improved OPC data logging system and its improvements over the original. The improved solution focused on the hardware, software and administrative components of the system; taking the reliability and performance of each component into consideration. The software components include the database, the data acquisition and logging client application (DALC) and the various OPC servers; the hardware component includes the servers that will run the software components, power management and Redundant Array of Independents Disks (RAID) technologies; and the administrative component includes implementing automated routines to backup the important data and archive old logs.

Keywords: Redundant Array of Independents Disks, power management, data acquisition and logging client.

1. INTRODUCTION

This paper discusses the results form a study that aimed to improve the reliability and performance of an OPC data logging system. The system was designed to capture the information from all the OPC compliant devices in an automated component handling system. The main focus was firstly the speed at which the data was recorded (performance), and secondly the integrity of the data and the reliability of the storage medium and related components. The following sections shortly discuss the initial shortcomings of the old system and the actions taken to address them. The main focus however will be on the final configuration and implementation of the system.

2. SHORTCOMINGS OF THE ORIGINAL SYSTEM

The original system had several shortcomings. Following is a short discussion of the main shortcomings. The servers were all mid-range desktop computers[1]; this meant that resources were very limited. These servers also offered no data security, as they operated from one hard disk. If this data on the disk would become corrupted or the disk itself was to fail, not only would the system stop functioning, but all data would be lost. The lack of a proper backup system only amplified this problem, as any corrupted or lost data would not be recoverable.

A secondary problem was that one these computes were used as both an application and database server. This resulted in the already limited recourses being shared between the applications and the database management system (DBMS).

Concerning the database schema and DBMS there were also several problems. The most prominent being that the DBMS was not properly configured and would not be able to handle user loads of more than 10 users; It also didn't make proper use of the available resources. The database schema itself was also lacking. The original schema mostly made use of large string type columns to store the data. This resulted in unnecessarily large records. These large records not only wasted the available storage space, but were also slow to read and write.

Lastly the data acquisition and logging client (DALC) lacked the functionality to browse for OPC servers and items; making it impractical in cases with many OPC servers and items.

3. ADDRESSING THE SYSTEM'S SHORTCOMINGS

To address these shortcomings the following was suggested:

3.1 Implementation of Dedicated Servers

It was suggested that three new server were installed. Due to the limitations of the study these servers would be built using high end desktop hardware, rather that the much more expensive server platform hardware. Each of these severs would then be used for a dedicated purpose. The one would be used as a web server, another as an application server, and the last as a database server.

3.2 Improve the Reliability and Performance of the Primary Storage Medium.

Redundant Array of independent Disks (RAID)[2] would also be reviewed. RAID offers increased reliability and performance over single disk configuration.

3.3 Implement proper power management and backup systems

It was also suggested that UPS be added to the system, and that it should be configured to shutdown the servers in case of extended power failures. A backup scheme would also be designed and implemented to ensure the corrupted and lost data would be recoverable.

3.4 Proper configuration of the DBMS

With a new dedicated database server it would also be possible to properly configure the MySQL DBMS to make proper use of the available resources. The most vital system variables would be identified; they would then be repeatedly adjusted benchmarked to find the optimum configuration.

3.5 Redesign the database schema to be more compact.

The database schema would also be redesigned to be as compact as possible. MyQLS's features, like data types, storage engines, partitioning, etc. would be evaluated to ensure that the resulting schema would be compact and fast.

3.6 Add OPC server and Item Browsing to the DALC

To add OPC server browsing and item browsing the DALC would either be modified or in the worst case rewritten. Factors like ease of use and its performance would also be taken into consideration.

4. FINAL IMPLEMENTATION AND CONFIGURATION OF THE SERVERS

The three new servers were built and setup to act as dedicated application, database and web servers. The servers were built to identical specification, except for the database server which had some additional components. The specifications of the server are as follows:

- An Intel Q6600 quad core CPU [3] (latest technology at the time)
- 8 GB of high speed low latency memory
- 3 x 250 GB hard disks.
- A mother board based on Intel's G45 chipset.
- Windows Server 2008 R2 was used as the final operating system for the servers.

In addition to these the database server had the following components:

- A Adaptec 3405 RAID controller [4](with a backup battery)
- 4 x 250 GB hard disk

The additional four [5] hard disks were used in conjunction the RAID controller to create a RAID 10 array. RAID 10 was selected since it proved too offer a very good balance between performance, reliability and effective capacity.
Each server's components were installed in a rack mountable chassis.

Figure 1 shows the components of the database server installed in its cassis. These chassis were then mounted in a 42U cabinet, as in.

A Gigabit network switch was also added to the system and mounted in the cabinet. The switch allowed ensured fast data transfers between servers.

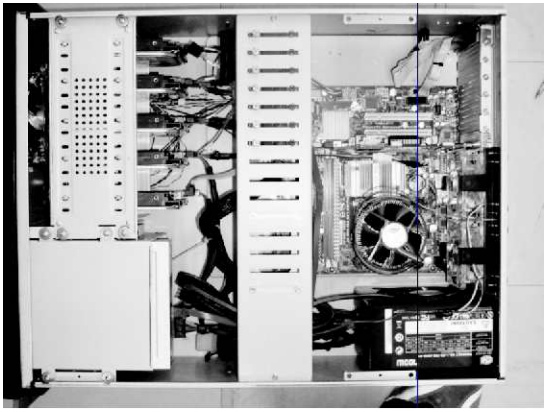

Figure 1. The Database Server's Hardware Installed into its Chassis

Figure 2. The Server Cabinet with all the Equipment Installed

4.1 Power Management

To protect against power failure or fluctuations a 3KV UPS was added to the system.

The UPS's serial interface was connected to the database server. The UPSMON software, included with the UPS, was installed configured to first shutdown the servers and then itself, in case of extended power failure.

Additionally the servers were also configured to start up again once the power is restored; the UPS automatically starts up once power is restored shows a timeline of how the system behaves in a verity of conditions.

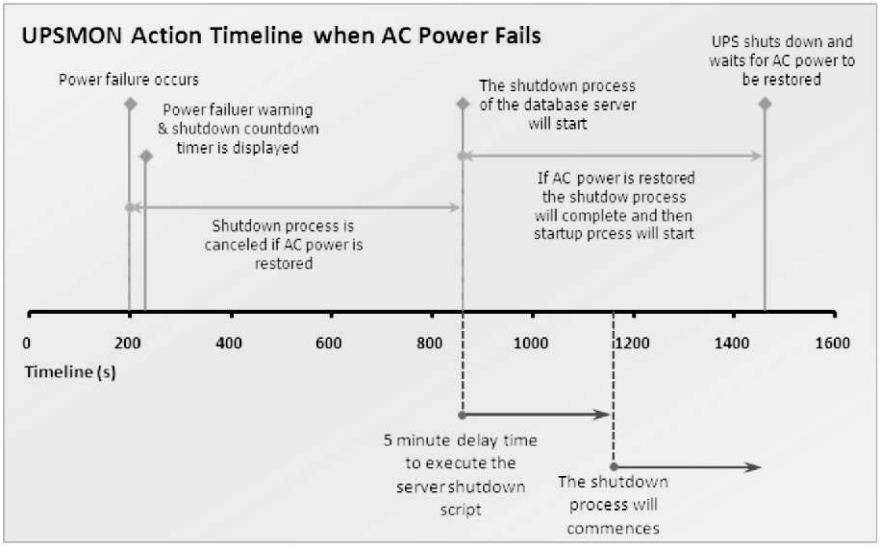

Figure 3. Timeline of the Actions Taken by UPSMON When AC Power Fails

4.2 The Backup System

The backups system was designed in such a way that individual files or the entire system disk could be recovered. It make use of a Daily, Weekly, Monthly scheme.

Command line scripts using Windows Server Backup tool was used to configure the backup's process. These scripts were then scheduled for execution using the Task Scheduler tool. Both these tools are included with Server 2008 R2.

The backup scheme for the system disk of each server work as follows:

The daily incremental backups are rotated between two of the server's 250 GB hard disks. The weekly incremental backups are done to an external hard disk that is kept of site after each backup session. Lastly a full system backup is done, on the last Sunday of each month, to 4 TB Network Attached Storage (NAS) disk.

After every 14 incremental backups to a specific location a full system backup is automatically made. For the database server the backup scheme was expanded to ensure that the database's data would also backed-up.

MySQL was configured to create its binary log files on the system disk. These binary log files track all transactions executed against the database, and can therefore be used as incremental backups of the database. Besides these log files that will be backed-up with the system disk a monthly full backup is also made to the NAS.

5. FINAL DESIGN OF THE DATABASE SCHEMA

Figure 4 shows the ERD diagram of the final OPC data logging schema design. The schema consists of four tables. The three main tables of the schema are:

- tblopcservers: it stores the information relating to the OPC servers that the DALC connect to.
- tblopcitems: it stores the information relating to each of the items of the OPC servers.
- tblitemlog_1: this table stores the actual logs that track the changing values of each item.

The fourth table, tblarchivelog, will be used to store older log entries that are still of importance.

The three main tables are based on the InnoDB storage engine, since InnoDB offer the following advantages:

- It is the only storage engine distributed with MySQL to en force foreign key relations
- It has support for very large tables
- It caches both data and indexes in memory
- It supports row level locking, as opposed to table level locking of most other storage engines

tblarchivelog was based on the Archive storage engine, as it offer high compression and makes the data read only.

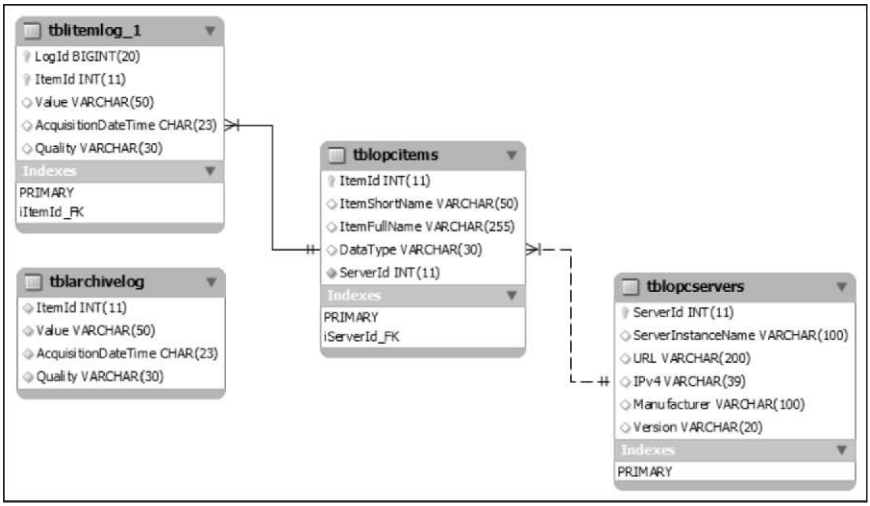

Figure 4. The OPC Data Logging Database Schema

6. Final Configuration of the DBMS

The MySQL DBMS has many system variables that influence its behaviour and performance [6]. While most of the variables affect the DBMS as a hole, the most important system variables proved to be those that affect the behaviour and performance of MySQLs two main storage engines: MyISAM and InnoDB [6].

Since InnoDB was selected as the primary storage engine its variables were adjusted accordingly. The variables relating to the MyISAM engine was left at their default values. The following variables were identified to have the greatest influence on the performance of InnoDB based tables:

- The *innodb_buffer_pool_size* variable was set to 6544 MB [6]. It determines how much memory space is reserved by the DBMS to cache table data and indexes, of tables based on the InnoDB storage engine.

- The *innodb_log_file_size* vaiable was set to 512 MB [6]. These log file tracks changes made to InnoDB based tables. In general larger log file increases recovery times, but in normal operation they may reduce disk I/O.

- The *innodb_flush_log_at_trx_commit* variable was set to 2. This means that data is flushed to disk once every second, instead of being flushed after a commit. This reduced disk I/O significantly, resulting in much faster execution of INSERT statements. When the RIAD controller's battery backed cache was enabled it further increased the write performance [6].

In addition to variable were also adjusted. Unlike the above mentioned variables these do not directly influence the performance of the DBMS, but are still vital for it to function as desired. These variables include:

- The *datadir* variable [7]: It was set to point to the RAID array, so that all tables would be created there.

- The *innodb_file_per_table* variable: It was enabled so that the data for each table would be stored in separate files.

7. FINAL IMPLEMENTATION OF THE DALC

The DACL was completely rewritten to improve its ease of use and to incorporate the new features. It now consists of two programs: a logging client, as in Figure 5, and a configuration client, as in Figure 6 below.

The configuration client allows the user to browse for OPC servers and items on the network. After selecting which items to log, the information is saved to a configuration file that is used by the logging client.

Figure 5. The Data Logging and Acquisition Client

The once the logging client is started it will automatically read the configuration file and start to log the information to the database.

The new DALC also makes use of bulk record INSERT queries to write data to the database. Bulk record INSERT queries proved to be significantly faster than single records INSERT queries, in some cases more than 10 times faster.

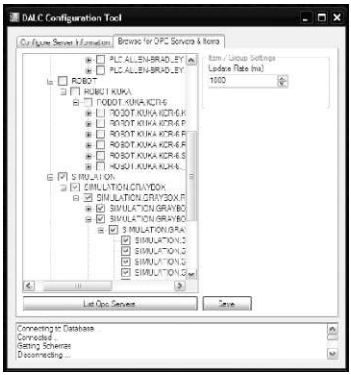

Figure 6. DALC Configuration Tool Browse for OPC Servers and Items

8. ACKNOWLEDGEMENTS

This forms part of research conducted by Research Group in Evolvable Manufacturing Systems (RGEMS) at the Central University of Technology, Free State. The Research was funded by the CUT and the Advanced Manufacturing Technology Strategy (AMTS) under project code AMTS-07-20-P; As well as the NRF Masters Grant Scheme.

9. BIBLIOGRAPHY

Bothma, B.C. and Vermaak, H.J., “Performance and Reliability Optimization of a Data Acquisition and Logging System in an Integrated Component-Handling Environment.,” in 2nd Robotics & Megatronics Symposium, Bloemfontein, South Africa, 2008, pp. 88-92.

Chen, P. M., Lee, E. K., Gibson, G. A., Katz, R. H. and Patterson, D. A., “RAID: high performance, reliable seconday storage,” ACM Comput. Surv., vol. 26, no. 2, pp. 145-185, June 1994.

Intel®. (August, 2007). Available: http://download.intel.com/design/processor/-datashts/31559205.pdf, last accessed in 05 2009.

Adaptec. (March, 2007). Available: http://www.adaptec.com/NR/rdonlyres-/94879AF5-C608-40AA-88AF-568826618162/0/4636_PCIeFamily_20.pdf, last accessed in 05 2009.

LaCie. (Mar., 2010). Available: http://www.lacie.com/download/whitepaper-/WP_RAID_EN.pdf.

Bothma, B.C. and Vermaak, H.J., “Configuring a MySQL database for a general purpose database in an Automation Environment,” in 12th Annual Research Seminar of the Faculty of Engineering and Information & Communications Technology, CUT , Bloemfontein, October 2009.

MySQL AB, MySQL Administrator's Guide and Language Reference, Second Edition ed., Seattle: MySQL Press, 2006.