A Parallel Architecture Using HDF for Storing DICOM Medical Images on Distributed File Systems

Tiago Steinmetz Soares
Informatics and Statistic Department
Federal University of Santa Catarina
Florianpolis, Brazil
Email: steinmetz@telemedicina.inf.ufsc.br

Douglas D.J. de Macedo
Post-Graduate Program of Knowledge Engineering and Management
Federal University of Santa Catarina
Florianpolis, Brazil
macedo@inf.ufsc.br

Michael A. Bauer
Department of Computer Science
University of Western Ontario, UWO
London, Canada
bauer@uwo.ca

M.A.R Dantas
Informatics and Statistic Department
Federal University of Santa Catarina
Florianpolis, Brazil
mario@inf.ufsc.br

***Abstract*—The Hierarchical Data Format (HDF) is an interesting approach for developing scientific applications where a large amount of data must be stored and accessed. A telemedicine project underway in the State of Santa Catarina (SC), in Brazil, has developed a server called the CyclopsDCM-Server, which adopts the HDF for the manipulation of medical images (DICOM). This paper proposes a new approach for the parallel implementation of I/O operations for the medical images stored on this server. This effort was based upon the MPI paradigm that is supported by the version 5 of the HDF. Early experiments indicate that the proposed approach can achieve very good performance when compared to the standard HDF implemented in the CyclopsDCM-Server.**

***Keywords*-Parallel I/O; HDF5; DICOM; Telemedicine; PVFS; MPI;**

I. INTRODUCTION

The success of an interactive telemedicine prototype experiment veried in the 1960s [1], between the Massachusetts General Hospital and a medical station at Bostons Logan International Airport, led to a dissemination of the idea of telemedicine throughout several countries. The term telemedicine is commonly used to refer to the remote delivery of health care, basically providing specialized health care, medical diagnosis and monitoring through telecommunications technology to people who cannot access to a medical system directly [2]. However, the capabilities of the equipment must be transmitted at least equal in quality to the information transmitted in the traditional setting. Indeed, the capabilities of the technology are expanding rapidly, becoming faster, more efficient and cheaper, enabling lower costs for the implementation and growth of telemedicine systems.

Researchers from the Telemedicine Laboratory at UFSC [3], adopting the telemedicine approach, created a telemedicine network project called Rede Catarinense de Telemedicina (RCTM). This project aims to provide connections among different hospitals and different cities within the State of Santa Catarina to provide access to exams, electrocardiograms (EKG), and imgaes from magnetic resonance, computed tomography, X-ray angiography and nuclear medicine [4]. All information acquired of a patient is sent online as DICOM images (Digital Image COmmunications in Medicine) to a developed PACS (Picture Archiving and Communication Systems), the CyclopsDCM-Server [5] and could be retrieved anytime where the system is deployed.

CyclopsDCMServer is a DICOM medical image facility that was conceived by the Cyclops Group [6] to provide DICOM image storage and wide area network (WAN) access. The CyclopsDCMServer stores all information in a ordinary data base PostgresSQL and can handle around 8 terabytes [4]. Summarizing, this server provides segmentation service for the incoming information, processing and storing the images in a centralized database.

A new DCMServer architecture was proposed to circumvent some of the issues of ordinary relational databases and it has been improve since then. This architecture has two basic applications, PVFS and HDF5. PVFS (Parallel virtual le system) is a distributed le systems designed to scale to petabytes of storage and provide high access rates [7]. THe Hierarchical Data Format 5 (HDF5) is a data model for high volume and complex data.

This paper is organized as follows. We start by describing DICOM images (Section 2) and some HDF5 definitions (Section 3), followed by some background about from previous work done on the system in Section 4; this describes several important aspects of the project. In the Section 5 we present some related work and in Section 6 the proposed architecture. In Section 7 we present some experimental

results, with subsections related to the environment and experiments. Finally in the Section 8 we present the conclusions and future work.

II. DICOM IMAGES

The Digital Imaging and Communications in Medicine standard is one of the most universal and fundamental standards in digital medical imaging. The DICOM standard was defined in 1992, and was the third version of the ACR-NEMA Standards Publication PS3. Before this standard was established, each manufacture created their own solution for visualization, storage and impression of digital images. The rst ACR-NEMA standard was conceived in 1983 by the American College of Radiology, with main principle to make digital medical images independent of device manufactures, creating a unique standard for medical devices and facilitating the expansion of digital images [8].

Taking many important features from earlier and other standards, the early versions of focused on the improvement and correction of some issues, and where those publications provided specifications related to hardware interfaces, it introduced a set of data format and commands for software packages. Completed in September 1992, the third version came with major revision, supplying increasing variety of digital devices and their communications protocols. This version was called DICOM 3.0, as it followed two earlier ACR-NEMA editions; the standard is reviewed annually and updated with new supplements if necessary [8].

Another important subject relative to DICOM is the Picture Archiving and Communication Systems. PACS consists in hardware and software medical systems designed to run digital medical imaging and is supported by major medical imaging equipment manufacturers. It embraces digital image acquisition devices, digital image archives and workstations.

The CyclopsDCMServer is both a digital image archive system and workstation, which was develop by the Cyclops Group. Created to work with PACS equipment as hospitals and radiology clinics, the purpose of the server is to store and retrieve DICOM index les from an ordinary data base managed by a relational DBMS, such as PostgreSQL. All communication between the server and medical equipment is performed through TCP/IP.

Nowdays, the server supports eight of the several DICOM modalities, namely: computed radiography (CR); computed tomography (CT); magnetic resonance (MR); nuclear medicine (NM); ultrasound (US); X-ray angiography (XA); electrocardiograms (DICOM waveform); DICOM structured reporting (SR) [5].

III. HIERARCHICAL DATA FORMAT (HDF)

Developed by the HDF group at the University of Illinois, initially in the 90s, the goal of HDF is to support large scientific data; the current version is HDF5. One of the main feature of HDF5 is that files can contain binary data as multi-dimensional arrays and allow direct access to parts of the file without first parsing the entire contents [9]. HDF5 is designed for storing large scientific data, including high performance data manipulation supporting random access, number encoding in native format, data compression, individual data set encryption, and storage strategies for parallel I/O and multidimensional data structures.

There are two essential structures in HDF5 which forms the base for the library: dataset and group. Dataset is a multi-dimensional array of datatype; HDF stores and organize all kinds of data from atomic to composed types, similar to the C struct construct. Other special array operations, such as chunks, compression and extendability, are available through the HDF library and can be applied to a dataset. The group is similar to UNIX directories, though cycles are allowed. Every file is started with a root group, represented as /, and could be followed by the name of another group or a dataset.

An important feature of HDF5 is support for standard parallel I/O interfaces. The Parallel Hierarchical Data Format 5 (Parallel HDF5) required MPI/IO interface through MPICH ROMIO [10] or a vendors MPI-IO, but it does not offer compatibility with shared memory programming. Implemented to get better performance in I/O procedures, the Parallel HDF5 uses distributed le system, such as Parallel Virtual File System (PVFS), Lustre, GPFS and specially configured NFS.

The idea of Parallel HDF5 is to make it easy for users to use the library and provide compatibility with serial HDF5 file. One approach is to read and write data by hyperslab [9], i.e., a multidimensional array that can be spread by rows, columns, patterns and chunks, and a hyperslab selection could be a logically contiguous collection of points, or it can be a regular pattern of points or blocks, depending on the type used. Other important structure is the dataspace. Through a dataspace required components of dataset or even a attributes are defined, as well as array ranks, sizes and types. The difference between the types of hyperslabs is the way that each process will access data of the datasets.

IV. BACKGROUND

One project that has been underway at the Telemedicina Laboratory at UFSC since 2008 is to explore new architectures for DICOM images using distributed file systems. The purpose of this research is to address issues of telemedicine environments based on ordinary database systems. These include addressing issues such as scalability, information distribution, ability to use high performance system techniques and operational costs. Among some of the procedures used to avoid the scalability issue, the project design was to use high performance distributed systems, like clusters or grids [4]. The first approach taken apart from the usual system was to store all information hierarchically, namely, organize and store in HDF5 data format. The second step was to use PVFS as a distributed file system.

Since the DICOM server normally supports drivers only for standard DBMSs, it was necessary to create something similar to these drivers. The HDF5 Wrapper Library (H5WL) was created for this purpose. As the name suggests, this library contains a wrapper object which is used to create, locate, collect and store information related to DICOM images using HDF5 files. When the CyclopsDCMServer requires a creation of new HDF5 file for H5WL, this is created using PVFS.

Two important entities were introduced to help reading data. The first entity consists of information related to the image, such as name of the patient, dimension of the image and other characteristics. The second is the image entity which represents the binary information created by PACS equipment (e.g., computed tomography (CT) or magnetic resonance (MR) images.

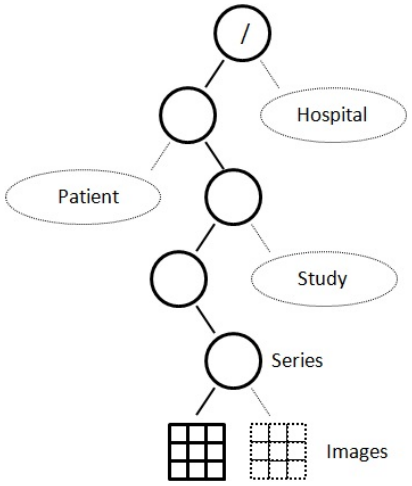

Figure 1. Hierarchical data sctrucuted [4]

Another feature is how the hierarchical data structured is organized. An example of the structure is shown in Figure 1, where the data is organized around six layers: root, hospital, patient, study, series and image.

When a client sends an image to the CyclopsDCMServer, the image is captured and the hierarchy is created through calls to the H5WL methods, creating groups that provide information that is used to identify the image inside the file. The biggest component of a DICOM file is the image, which is the leaf of the structure. Basically, the structure image is a compressed image (JPEG) [8] and is responsible for representing the DICOM image.

This architecture, proposed by Macedo [1], has virtues and weakness. Based on 25 experiences with the system, it showed an average improvement in storage of about 16% when compared with the usual system using DBMSs.

However, in term of retrieval operations, there was a drop in performance, with average decrease in performance of around 21%. This was due to mechanisms used to provide similar behavior to that of standard DBMSs when retrieving information. This paper addresses this problem by proposing an extension to the architecture to take more advantage of a parallel environment. This architecture is detailed in Section VI.

V. RELATED WORK

There is little published work in the telemedicine field which uses HDF5 to store images, as most medical image servers come with drivers only for ordinary data bases. The three works below are similar to the current work in that they use parallel I/O as a solution for I/O bottlenecks access for large amounts of stored data.

The research work presented in Nikhil Laghave [11], is very similar to our work. This work is focused on the use of a parallel I/O library for scalability issues involving fermion dynamics for nuclear structure (MFDn). This work used the HDF5 parallel version for parallel I/O, testing with collective and independent models. As result, there was a gain in the efficiency of input/output of large datasets and the cost of using parallel I/O was less than sequential I/O for sufficiently large datasets.

In particle-based accelerator simulation groups, it is possible to find some HDF5 work. The work of A. Adelmann [12] focused on using parallel I/O for particle simulations which involved vast quantities of data and dimensional arrays. He used parallel I/O performance for MPI code as well parallel HDF5. He compared read and write performance in simulations between Parallel HDF5, mpi-io and one file per process. HDF5 showed good performance in writing, though mpi-io showed better results.

H. Yu [13] presented interesting work, though he did not use parallel HDF as solution for his problem, but rather a similar paradigm. His works dealt with large earthquake simulations which require terabytes of storage space and encountered I/O bottleneck issues. He developed his own parallel I/O strategies through MPI I/O to address his needs and was able to remove the I/O bottleneck and also hide pre-processing costs.

VI. PARALLEL ARCHITECTURE

One of the great features available in HDF5 and was not considered in Macedos approach, is the support for MPI communication for parallel processing. As a possible solution to the bottleneck of retrieving data, we provide additional features using the Parallel HDF5 library. The main propose is to get better performance using parallel data access to HDF files stored in the PVFS distributed file system. The Parallel HDF5 library requires a parallel MPI/IO interface and, when working with MPI, it is necessary to design it to be used in a cluster environment. Another important requisite is the necessity to use the mpirun shell script to run any MPI application, which attempts to hide the differences in starting jobs for various devices from the user

[14]. For this, it is necessary to create a additional procedure to work with the CyclopsDCMServer. This procedure should be called every time when is required to retrieve or store some medical information.

It is noteworthy that Parallel HDF5 has support for PVFS through MPICH ROMIO. In this case, our first task is to build the environment using MPICH2 [12] with ROMIO for PVFS. With the environment built, the second task is to create the application which it will be responsible for reading and writing a dataset into a file.

Figure 2 illustrates how the architecture works. The functionality, basically, is the same as described in the previous section, the difference is inside of the H5WL. Instead of having the H5WL responsible for reading and writing the binary information created by PACS, it will be treated as a new parallel application. The parallel application will be initiated by H5WL calling mpirun shell script.

Independent of a read or write function, when H5WL calls the MPI application, all communication between them will be made by socket connections. The communication is done by the master process (represented by MPI process zero) and H5WL for passing function parameters like the location that is the target of an operation (group path), the image buffer and the number of MPI processes. The difference between the functions is the way that the server and MPI applications will communicate. For write functions, the H5WL will first receive the DICOM file, create a new hierarchy of the image based on DICOM file layers (Figure 1), get the path location for new image (JPEG image) and then call mpirun procedure to start the MPI application. The MPI application has functionally to read and write a buffer, without being concerned whether the image exists or not, as that is the responsibility of H5WL. The master process will first communicate with H5WL to retrieve the function to perform, get the location (group) of image in the HDF5 structure and the arguments for the job. If it is a write function, it will need the stream of images to be stored. In case of a read function, the application will only need the path group as parameters, and it will return to server all buffers read from the HDF file.

Independent of the job, the master process has to define the access properties, model and size for each process. The Parallel HDF5 library has available two types of properties (collective and independent data access) and four hyperslab model (Contiguous Hyperslab, Regularly Spaced Data, Pattern and Chunk) [9]. Then the master node has to distribute the memory buffer (write function) or file location to each process. Finally, once the jobs have executed, the main process will return to the wrapper the status of reading or writing the buffer.

VII. EXPERIMENTAL RESULTS

Our experiments are based on the Parallel HDF5 architecture, adapted to use PVFS, and sequential CyclopsDCMServe. The Parallel HDF5 properties used for read and write on the MPI application is: independent data access model and Contiguous Hyperlasb, which entail distributing the buffer by rows as show in the Figure 3.

It is important to note that our results do not take into consideration external factors, like computers using the same network

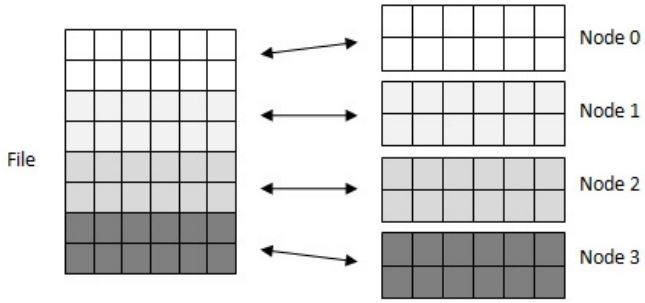

Figure 3. Contiguous Hyperlasb

A. Environment

The environment used for the experiments consist of a four node cluster, as specified in Table 1. The cluster is non-dedicated and is used just for experiments and belongs to Telemedicine Laboratory. The connection network is 100 Mbs Ethernet. The operating system installed on all nodes is CentOS with kernel 2.6.18. PVFS is used on only one metadata node. Each has a PVFS client for access to the PVFS file system and each node is also a MPI executioner and has an MPI application in its own file system for access to HDF file found in Parallel Virtual File System.

Name	CPU	Memory	HD
Node1	AMD athlon x2 2.1 GHz	2 Gb	20 Gb
Node2	AMD athlon x2 2.8 GHz	3 Gb	20 Gb
Node3	Intel PentiumR Dual 1.80 GHz	1 Gb	20 Gb
Node4	Intel Core i5 3.2 GHz	3 Gb	20 Gb

Table I
ENVIRONMENT

B. Experiments

Experiments were conducted with CyclopsDCMServer sequential and parallel architecture. The experiments involve only comparison of writing a new DICOM file in HDF file; future comparisons will compare file retrieval. This experiment measures the time spent write an image buffer into a HDF data set and was done 25 times with different DICOM files for each test. The selection of the files used was random, but the same files were used for the parallel process. The time collected is the time required to write an image, ignoring other information, like patient name,

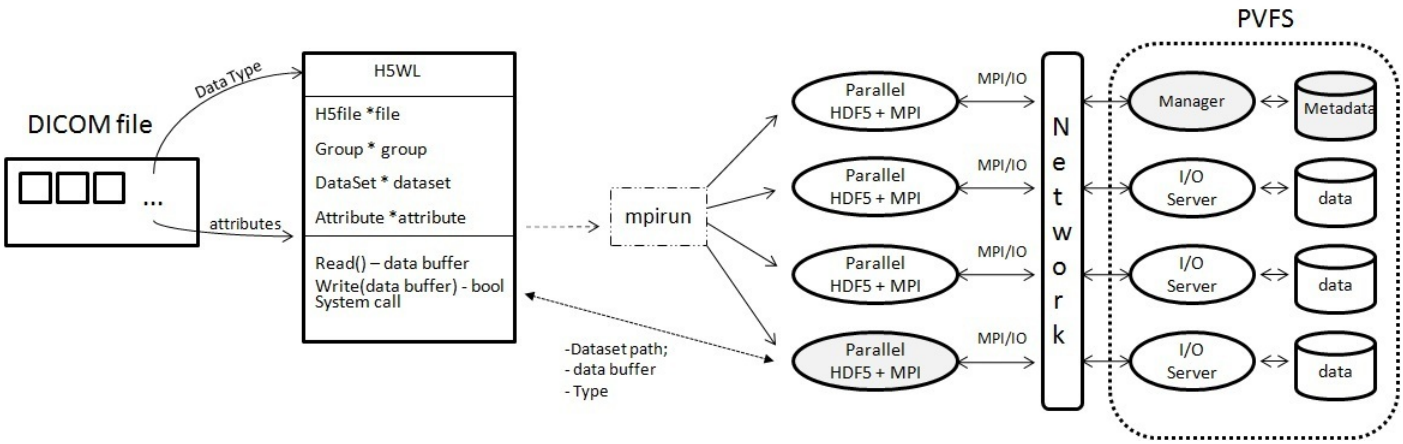

Figure 2. The architecture proposed

hospital and etc. The reason that this measure was chosen is because an image represents most of a DICOM file, i.e., could be over ninety percent and is normally nearly 50 Mb. These images were created by CT equipment that generates monochromatic images with 512 512 pixels with 16 bits per pixel.

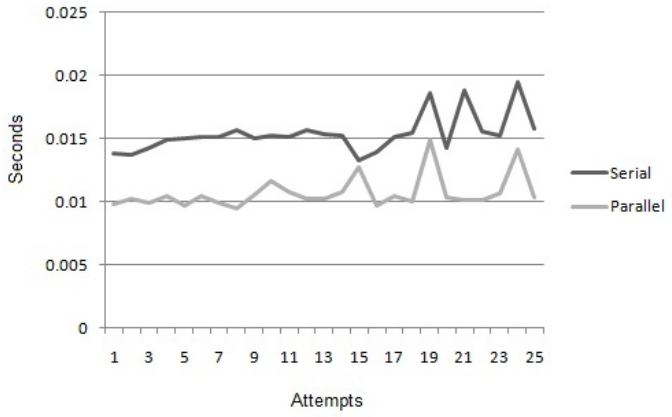

Figure 4. Contiguous Hyperlasb

Figure 4 compares the performance between serial CyclosDCMServer and our integrated parallel architecture. Our architecture shows better performance than the serial with the average writing time for parallel method being 0.0107176 seconds, while the average writing time for the serial was 0.01539956 seconds an improvement of around 30 percent. The minimal elapsed times were 0.013812s and 0.009815s for serial and parallel respectively, while the maximal elapsed times were 0.019435s and 0.012735s for serial and parallel.

VIII. CONCLUSION AND FUTURE WORKS

In this paper, was introduced an extension to the architecture for the CyclopsDCMServer that was introduced by Macedo et al. [1]. The focus of this paper was to introduce a new parallel architecture designed to reduce the bottleneck I/O issues in the serial architecture. Experiments involving writing compared the serial and parallel I/O in the same environment and show that the parallel architecture resulted in a thirty percent improvement.

As future work, there is a need to compare the retrieval of a DICOM image from an HDF file using the parallel architecture. It is not clear how this will perform since this uses H5WL which will not perform as well as a standard DBMS.

Another important experiment is to measure the performance of the complete operations of receiving a DICOM file, wrapping and storing it, and the reverse operation of retrieving and unwrapping it. Others future work includes analyzing the significance of the number of MPI nodes on reading and writing and to measure the communication between H5WL and master mpi node.

As seen in the Section 6, many researchers that have similar issues have used parellel I/O to avoid bottleneck I/O problems and have obtained similar results. Given this and considering the features available in HDF, one can expect to see gains in other future work involving the retrieval of stored information.

REFERENCES

[1] D. de Macedo, H. Perantunes, L. Maia, E. Comunello, A. von Wangenheim, and M. Dantas, “An interoperability approach based on asynchronous replication among distributed internet databases,” in *Computers and Communications, 2008. ISCC 2008. IEEE Symposium on*, 2008, pp. 658 –663.

[2] W. Hersh, U. S. A. for Healthcare Research, Quality, and O. H. S. U. E. based Practice Center, *Telemedicine for the Medicare population: Update*. Citeseer, 2006.

[3] "Laboratorio de telemedicina," Access:, January 2011. [Online]. Available: http://www.telemedicina.ufsc.br

[4] D. De Macedo, A. Von Wangenheim, M. Dantas, and H. Perantunes, "An architecture for dicom medical images storage and retrieval adopting distributed file systems," *International Journal of High Performance Systems Architecture*, vol. 2, no. 2, pp. 99–106, 2009.

[5] "Cyclops project," Available at:http://www.cyclops.ufsc.br . Access: 2011., January 2011. [Online]. Available: http://www.cyclops.ufsc.br

[6] "Cyclops group," Access:, December 2011. [Online]. Available: http://cyclops.telemedicina.ufsc.br

[7] "Pvfs," Access:, January 2011. [Online]. Available: http://www.pvfs.org

[8] O. Pianykh, *Digital Imaging and Communications in Medicine (DICOM): A practical introduction and survival guide*. Springer Verlag, 2008.

[9] "Hdfgroup," Available at:http://www.hdfgroup.org . Access: 2011., January 2011. [Online]. Available: http://www.hdfgroup.org

[10] "Romio," Access:, March 2011. [Online]. Available: http://www.mcs.anl.gov/research/projects/romio/

[11] N. Laghave, M. Sosonkina, P. Maris, and J. Vary, "Benefits of parallel i/o in ab initio nuclear physics calculations," *Computational Science–ICCS 2009*, pp. 84–93, 2009.

[12] A. Adelmann, R. Ryne, J. Shalf, and C. Siegerist, "H5part: A portable high performance parallel data interface for particle simulations," in *Particle Accelerator Conference, 2005. PAC 2005. Proceedings of the*. IEEE, 2006, pp. 4129–4131.

[13] H. Yu, K. Ma, and J. Welling, "A parallel visualization pipeline for terascale earthquake simulations," in *Supercomputing, 2004. Proceedings of the ACM/IEEE SC2004 Conference*. IEEE, 2005, p. 49.

[14] "Mpirun," Access:, March 2011. [Online]. Available: http://www.mcs.anl.gov/research/projects/mpi/www/www1/mpirun.html