

Building a Community Grid for Medical Image Analysis inside a Hospital, a Case Study

Marko Niinimäki¹, Xin Zhou¹, Adrien Depeursinge¹, Antoine Geissbühler¹, and Henning Müller^{1, 2}

¹Medical Informatics Service, Geneva University Hospitals and University of Geneva, Geneva, Switzerland

²Business Information System, University of Applied Sciences Sierre, Switzerland

Abstract

This paper describes steps taken to create an internal Grid network to support medical image analysis at the Geneva University Hospitals. While computing resources today are inexpensive, the real bottleneck of their usage in many institutes is the difficulty to harness them for a particular application and to maintain an infrastructure. To overcome this problem we have created a community Grid that provides the researchers a simple interface to a computing cluster inside the hospitals. The fact that no data has to leave the institution for the computation makes the access to the resources easier for researchers and removes some of the security constraints. Reusing existing computers on the hospital network, an automatic setup to distribute the software, using conceptualization techniques, and using a robust Grid solution based on Linux and ARC (Advanced Resource Connector) all limit the required maintenance and can quickly give researchers access to computing power.

We describe the components of our setup, its usage with a real application, and demonstrate its performance.

Contents

1	Introduction	3
2	Challenges for creating a computing infrastructure inside a hospital	5
3	Solution guidelines	6
4	An application	8
5	Conclusions	10

1 Introduction

Modern hospitals produce enormous amounts of data in all departments, from images, to lab results, medication use, and release letters. Since several years these data are most often produced in digital form, making

them accessible for researchers to optimize the outcome of the care process and analyze all the available data across patients. The Geneva University Hospitals (HUG) are no exception to this with its daily radiology department's output of over 70'000 images in 2007, majority of them tomographic slices, sometimes in temporal series with a relatively small size (around 512 kB). Other departments such as cardiology, dermatology, or pathology, likewise produce large amounts of visual data that need to be stored and could be analyzed for diagnostic aid tools [4].

Content-based medical image retrieval in general has the goal to allow for retrieval of similar images or cases over very heterogeneous image collections to help the diagnostic process [15, 20]. For large collections, this is a time consuming process, even if for single images feature extraction is often fast [21]. Parallelization of the processing is needed to keep up with the flood of images.

Different technologies can be applied to parallization. In [14], we utilized the Parallel Virtual Machine software tool to embed in the program code how to distribute the processing in a few (locally connected) computers. Cluster computing systems, or Local Resource Management Systems (LRMSes), like Condor [11] provide a higher level command language in order to, typically, send a program and its input to be run in any computer of the cluster. Grid technologies enable us to connect several clusters, even if they run different kinds of LRMSes.

With their security infrastructure, standardized job execution and data access interfaces, Grid technologies provide tools for complicated distributed tasks. "Gridified" image retrieval has been proposed several times, most often using external clusters made available by large-scale research projects in the domain [12, 16]. Sustainability of these solutions and a business model still need to be developed.

Most approaches for image retrieval on Grids concentrate on the visual feature extraction as this is generally the most time-consuming part. Projects such as caBIG¹ also allow for image retrieval in Grid environments but rather relate to text-based retrieval, which is less computationally expensive.

Previously, the medGIFT² group at the Medical Informatics Service at the HUG successfully gridified their GNU Image Finding Tool (GIFT³)-based medical image retrieval application [17, 8]. However, the actual usage of this gridified version was limited to a single quad-core server, or external Grid resources, as described in [21], made available by the KnowARC⁴ EU research project.

While Grids in general can be characterized as resource sharing in multi-institutional organizations [7], Grid technologies can be applied to access resources inside just one organization, as in our case. The benefits of this are (1) conformance with the controlled and strict access policies needed in organizations that handle confidential data, (2) the possibility to connect resources at different sites within the organization — for instance in different subnetworks in buildings that can be many kilometers apart, and (3) the possibility of later expanding the analysis to an inter-organizational level while using the same interfaces.

Like most hospitals, the HUG do not have any central computing infrastructure at the moment that would allow researchers to run computationally intensive applications, for instance in a cluster of dedicated computers or a mainframe computer. However, over 6'000 desktop computers are available on the network of the hospital for computation, and another 5'000 on the University of Geneva network. Even a partial use of these resources, for example at night or on a voluntary basis, could help to fulfill the most urgent computational needs for medical image analysis that would allow much more advanced algorithms. Currently, most of these computers are mainly used to access patient records from central databases, an activity that consumes only very little of the resources of the computers. At the same time, care must be taken that the

¹<http://cabig.nci.nih.gov/>

²<http://www.sim.hcuge.ch/medgift/>

³<http://www.gnu.org/software/gift/>

⁴<http://www.knowarc.eu/>

analysis tasks do not exhaust the hospital network resources.

In this paper we demonstrate that the creation of a computing Grid within a hospital organization (such as ours) is possible, despite the many technical and particularly political concerns. We have built a test bed consisting of standard hospital desktop PCs running a LRMS (Local Resource Management System) that is capable of accepting jobs from the ARC (Advanced Resource Connector) Grid middleware [5]. The LRMS runs inside virtual machines, so that the software needed for distributed computing tasks can be run in an isolated environment. Our test bed setup was designed so that it can be controlled by the hospital IT administrators and requires minimum intervention from the part of the researchers. Existing infrastructures for software distribution as well as security policies were all taken into account.

The rest of the paper is organized as follows. In Section 2 we describe the challenges met for harnessing the desktop computing resources of a large hospital with a sometimes complicated security and political structure. In Section 3 we present our solution based on the given constraints. In Section 4, we discuss our GIFT-based application and demonstrate its performance in our internal hospital Grid. Conclusions are drawn in Section 5.

2 Challenges for creating a computing infrastructure inside a hospital

Several problems for creating Grid networks inside medical institutions were already known before the beginning of the project [13, 17]. These concern mainly the little flexibility of a large institution towards research projects that are clearly not the main objectives of the teams managing the infrastructure. This is quite understandable since the smooth running of the hospital network, networked appliances and applications is certainly a priority in an organization where people's lives can depend on them. Data security and confidentiality are similarly important and our applications can not compromise them.

On the other hand, Grid software is often relatively demanding to its environment, often requiring e.g. computers with static IP (Internet Protocol) addresses and access to specific TCP (Transmission Control Protocol) ports. While these are not difficult to be provided within hospital local area networks (LANs), they may still contradict hospital policies. Grid and LRMS systems tend to be fairly intrusive as well, requiring dedicated computing resources, whereas we would favor a setup where computing can use only a part of the resources of a normal user's desktop PC.

Thus the main problems are related to infrastructure (I) and Grid software (G). Political problems (P) form an additional factor. We can summarize these three categories and the requirements that they pose to our solution as follows:

- I1 All applications within the hospital are behind a rigid firewall that does not allow us to have direct connections outside of the hospitals: all TCP ports other than 80 (http - Hypertext Transfer Protocol) and 443 (https - Hypertext Transfer Protocol over Secure Socket Layer) are blocked, and access to http/https is only possible through a proxy. The only way for us to get access to the University network was via a VPN (Virtual Private Network). Requirement: our Grid setup should not rely on external resources.
- I2 Research applications such as GIFT (GNU Image Finding Tool) and also Grid middlewares are often developed under Linux, whereas Linux was not allowed to be installed on the hospital desktops at the beginning of our project. Requirement: do not assume a Linux environment.
- G1 For an inexperienced programmer and system administrator, Grid systems are hard to install, maintain, and interface. Grid certificates can be difficult to obtain. Requirement: allow sufficient time for

learning, use external expertise, use a temporary certificate authority, if needed.

- P1 Grids are relatively unknown in hospital network departments. The lack of knowledge creates a rejection of technology for the responsible persons as they are afraid to not be able to control the infrastructure. Requirement: involve hospital IT, create a test bed for such new technology.
- P2 The fragmentation of the structure of the hospital into very independent entities results often in very strict decision hierarchies and rather a tendency to reject technology and remain with an existing, restrictive structure; it is sometimes hard to find the responsible person for a particular decision and often response times for requests can be long due to the fragmentation. Requirement: be patient. Try to employ a solution that does not need much of the IT personnel's involvement.

3 Solution guidelines

Despite the problems, we were able to establish a test bed and based on it demonstrate the benefits of Grids in the domain of the hospital. A wider distribution inside the hospital is planned to be deployed for research projects in the future.

Briefly, our solution can be described as running a computing node inside Virtual Machines that, in turn, are run in desktop computers. Job submission to these computing nodes is managed by the ARC middleware. Currently, users who want to submit jobs, log in to a Linux computer that has the both the ARC server and its job submission interface. More computers can be added later both for job submission and running ARC, removing the "single point of failure". Virtualization was chosen as the processes running in the virtual machines can be in another operating system than the host (in our case Linux) and the processes running in a virtual machine have no access to potentially confidential files on the host operating system.⁵

The requirements were considered in the solution as follows:

- I1 A local Grid information system was employed inside the hospital network to control the information of the cluster. No external connectivity is required. However, the solution can support job submission to both internal and external resources, when available. The internal computing resources do not need any external connectivity.
- I2 The desktops run the hospital standard issue Windows XP, completely controlled by the hospital IT (by standardized distributed installation images). A specific distribution package was created for these computers, containing VMPlayer⁶ and the Virtual Machine image. The image contains a Debian Linux, with all the software packages required for image analysis, and the LRMS for receiving packets to be computed and sending back the results. The Linux image was to be kept as small as possible to limit its influence on the network traffic when installing the images on a larger number of desktops.
- G1 Our decision to use ARC and the Condor LRMS was based on the project personnel's expertise in these tools, and enabled us to create a working Grid and cluster setup quickly. Our own certificate authority (CA) was used when getting certificates from official Grid CA's took too much time.
- P1 Before our Grid system could potentially be deployed in the hospital at a larger scale, a smaller set of desktop PCs was assigned for this purpose. The network administrators wanted to assure thorough testing before any larger-scale use. For our test bed, 20 desktop PCs of the type Compaq Evo 510

⁵These are often cited as benefits of a Virtual Machine Grid, see e.g. [6]. For an overview of virtualization techniques see [1].

⁶<http://www.vmware.com/>

(Pentium 4, 2.8 GHz) were made available to us.⁷ A standard hospital installation of Windows XP with antivirus software was used as the operating system. The network connectivity of these computers was, likewise, the same as for any hospital desktop, i.e. they belong to the same sub network as other desktops in the main hospital area, and gain their IP addresses from the same address pool.

- P2 Originally, we wanted to run Linux in the Virtual Machine images with Network Address Translation (NAT); they were able to initiate connections to the network through the host computer, but incoming connections would be impossible. Therefore, the Virtual Machine images would not have needed to have site-wide IP addresses managed by the hospital IT.

The LRMS, in our case Condor, is run inside the virtual machines and communicates with the Manager node. In a setup called Bridged Model, virtual machines can have IP addresses that are accessible from the other nodes. However, this means that the Virtual machines need to gain their IP addresses from the hospital's DHCP (Dynamic Host Configuration Protocol) server, and that is strongly discouraged for administrative reasons. Some workarounds exist, including a VPN (Virtual Private Network) tunnel from the execution nodes to the manager, and Condor GCB (Generic Connection Broker) [19]. We originally relied on GCB in the following setup. The worker nodes were configured to establish connections to a computer running GCB (actually this is the same computer as our Condor Manager but this is not important in this context). GCB then forwards the connections to Condor Manager, letting the execution node and the Manager communicate as if there was no NAT. However, this approach did not prove to be reliable enough. If the connection was dropped due to network lags or other minor problems, it failed to re-establish. Our current model forces a MAC (Medium Access Control) address to the Virtual Machine at startup, and the DHCP (Dynamic Host Configuration Protocol) administrator has allocated IP addresses for these specific MACs. However, this adds unnecessary administration overhead to the system and another solution is envisioned for the future to allow for a better scalability.

In practice, the setup of our environment is shown in Figure 1. One local ARC server is currently in production. The same computer acts as a manager node for the Condor LRMS. By installing more ARC/Condor servers in the hospital, we will supply redundancy so that the system will work even if one of the servers fails. A Grid Index Information Service (GIIS) contains the details of the ARC servers, including number of active nodes, and the runtime environments (like Java) provided by these servers.

A grid job description contains the input images for further feature extraction, the GIFT system as source code, and a small program to compile and run GIFT with the images. ARC's GridManager receives the requests and pushes them to Condor. ARC then supervises their execution and receives their output. The output is stored at the ARC server until the user issues a download command.

Condor execution nodes are run in Virtual Machines (VMs), as shown on the left side of the figure. The Linux system inside the VM contains a C compiler and other tools needed to compile and execute GIFT feature extraction.

Since the execution nodes are run in desktop PC's, the system must be resistant to failures – after all, PC users regularly re-boot their computers. Here, we have used a job manager called GridJM⁸. In the case of failure, the job manager simply re-submits the job until it succeeds or a re-submit limit has been reached.

Grid users log in to the same Linux computers where we run the ARC servers, i.e. job submission is done in those computers, too. This is not an optimal solution, but needed since the user's in general have only Windows desktop computers and job submission is easier using Linux.

⁷It should be noticed that the processor of these computers does not support virtualization extensions thus making the virtual machines run slower than more recent hardware; for details about hardware level virtualization support see e.g. [9].

⁸<http://www.tcs.hut.fi/aehyvari/gridjm/>

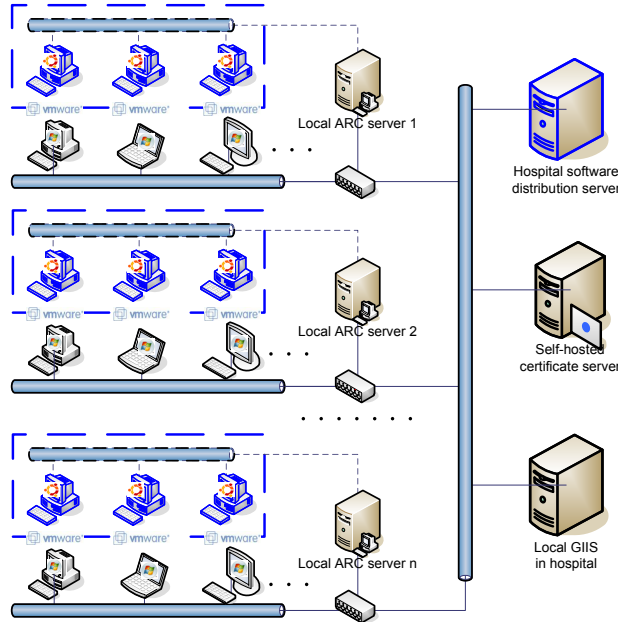


Figure 1: The main components of our setup.

Finally, the hospital IT administration controls the Windows operating system environment completely. The VMware installation and Linux VMware images are automatically installed in the computers using an automated installation system provided by the hospital.

4 An application

Our setup was created for the needs of the MedGIFT group of the Medical Informatics Service and the application uses a gridified version of the GNU Image Finding Tool (GIFT) created in a research project at Geneva University several years ago. In practice this means the data from a large image database is re-packaged in chunks of N images, the packages and their processing instructions are sent to computing nodes where they are processed in parallel, and the results are recovered from the nodes. A job description language, in our case ARC's XRSL (Extended Resource Specification Language, see [18]), is used to request the computing.

In line with our previous research ([21]), we have used the ImageCLEF (see [3]) database of almost 50'000 images with a package size of 1'000 (except for the last package, naturally). The flow of a single job is as follows:

- the user submits the job to the ARC node;
- the ARC middleware pushes the job to the LRMS;
- execution in the LRMS takes place in the virtual machines; the virtual machines are waiting for jobs since they have been deployed by the hospital installation system;
- the user runs a command to receive the output of the job.

4 CPU server	709 min
37 CPU remote clusters	537 min
20 CPU local VM cluster	240 min

Table 1: Comparison of running jobs in a local 4 CPU cluster, remote clusters of the KnowARC project, and our hospital VM cluster.

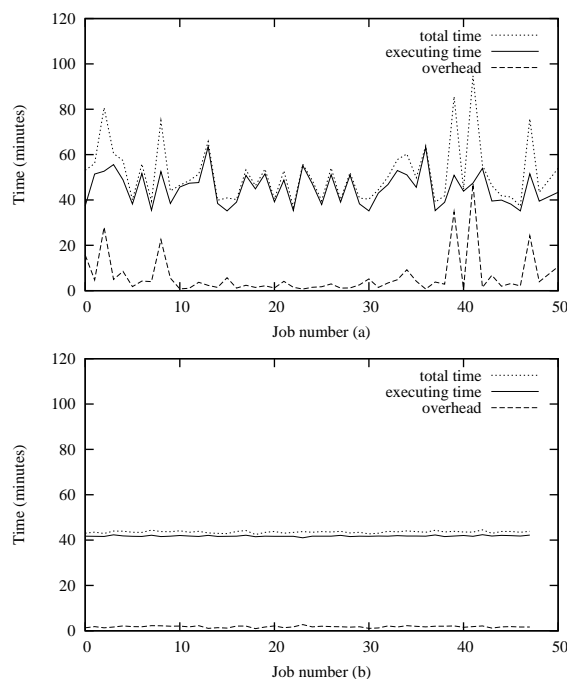


Figure 2: Comparison of the time consumption with remote (a) and local (b) infrastructures.

In the case of a single job it is important to evaluate (a) the performance of a VM-based Linux installation compared to a native one in an identical computer and (b) delays imposed by job preparation by the middleware and the LRMS, and the transfer of the package to the execution node. The exact figures (32 minutes in native Linux, 41 minutes in the virtual machine, 45 minutes per average by remote ARC resources) show that the overheads caused by the virtual machine, ARC, and the LRMS are negligible compared to actual computing time.

A more concise result is shown in Table 1. It consists of running a large set of image analysis jobs under the control of a GridJM, as describe in our previous paper [21]. In practice, a job generator creates the packages and the XRSL job description files. They are passed to GridJM that submits them to the ARC resources that are available to the user, resubmits in the case of a failure, and recovers the results.

In Table 1, we compare feature extraction in a quad-core server and a small Grid of 37 CPUs in remote locations in several countries. In all these cases, the number of image packages is 50, containing 1'000 images each (the last package slightly less), each package about 190 MB. Here, we compare the results of that paper in a situation where we have the same data sets executed in our Virtual Machine cluster.

In [21], we presented the execution with remote Grid resources, and predicted that using a local cluster will largely reduce the overall execution time. Figure 2 shows a comparison between using remote Grid resources and the local cluster, which proves this assumption. The local cluster used in this test consists of

20 desktops (five-year-old Compaq Evo desktop computers with Pentium 4–2.8 GHz, 768MB memory). However, since our package generator could not supply packages fast enough, only a maximum of 10 jobs run in parallel.

An identical Virtual Machine image was used in each of the desktops used taking 300MB of memory. From Figure 2 we can see that the time spent on execution is quite uniform (“flat lines”), as each computing node is set up in the same way. The main benefit comes from the time gained from scheduling, queuing, and transfer of data. This overhead is reduced from 10 minutes on average per package (in the case of remote resources) to only 1 minute. With only 10 desktops employed in the local cluster for testing, the total time is reduced to less than half. No jobs were required to be resubmitted, either.

The results can be improved easily by using a smaller package size, so that package generation does not become a bottleneck.

Finally, we present some technical details about the Virtual Machines. The VM “image” that contains the Linux system runnable inside the VM is distributed as a single, compressed 317 MB file. The Linux system in the Virtual Machine image is based on Debian 3.1. The principal software packages and their sizes are as follows: GNU compiler, linker and header files – ca. 30 MB; ImageMagick image conversion software – ca. 27 MB; the Java environment – ca. 170 MB; and Condor – ca. 100 MB.

The impact of running the analysis was not significant for the network, since only 50 files of size 190 MB (and small result files) were transmitted. The impact for the desktop users will be more visible: when the Virtual Machine is running, half of the PC’s memory is used by it. Moreover, we measured the CPU performance using NovaBench software, first when no other software was running in the PC, and later during the analysis. The results indicate that the analysis used about 50% of the PC’s CPU capacity, too.⁹

5 Conclusions

In this paper we have demonstrated the feasibility of an internal hospital Grid. We discussed the challenges imposed by strict requirements of the organization, and how to find solutions to the various problems. An application in content-based medical image retrieval using our Grid cluster is described and performance measurements are shown.

This article solves several problems that exist in many medical institutions. Data are produced in quickly increasing quantities and analyzing these data is required to improve care process. Unfortunately, very few central computing resources exist in hospitals for research projects to analyze such large amounts of data using modern, complex algorithms. Desktop computers on the other hand exist in large quantities in most medical institutions and reusing them for computation seems a good idea. Our setup using virtualization for the Grid solves several problems:

- the Linux operating system can be used for the image analysis;
- security is improved as the virtual machine does not have access to the host operating system;
- the user remains in control and can stop a virtual machine in case the processing is slowing down the computer or he needs the computing power for other tasks.

After all, our experiences with these Grid technologies are very positive. The setup of the Grid using the standard automatic software distribution system of the hospitals was very quick. Maintenance of such a

⁹Without the VM: 35 Mflops/s, 6.4 M integer operations/s, when analysis running in the VM: 18 Mflops/s, 2.3 M integer operations/s.

system is also relatively easy as a maximum of steps for setup and maintenance are automated. Maintaining the software environment in the virtual machines, too, has been easier than expected since we have been able to provide a single virtual machine image, containing all the components needed for image analysis jobs. It should be foreseen that the maintenance will become more demanding if completely different type of analysis, with different software components, will be needed in the future. However, so-called dynamic runtime environments (see [2]) for Grid middleware will facilitate such tasks.

Overhead when using external Grid resources caused by job scheduling and transmissions of large datasets are also reduced significantly. The system is thus ready for a larger deployment and the use for other research projects than only image retrieval. For the time being, we have only tested our system with tasks containing a single step (e.g. feature extraction from a set of images) running in parallel. In the future, more complicated workflows with jobs producing output for other jobs are likely to enter the scene. In order to execute these kinds of work flows in our Grid, ARC integrated with Taverna workflow manager (see [10]) will be tested.

Acknowledgements

This work was partially funded by the European Union 6th Framework program in the context of the KnowARC research project (contract number 032691). We would also like to thank the network group of the HUG for their support in the project and the quick responses we often received.

References

- [1] Keith Adams and Ole Agesen. A comparison of software and hardware techniques for x86 virtualization. In *ASPLOS-XII: Proceedings of the 12th international conference on Architectural support for programming languages and operating systems*, pages 2–13, New York, NY, USA, 2006. ACM. 5
- [2] D. Bayer, T. Bhindi, F. Orellana, A. Waananen, B. Konya, and S Moeller. Dynamic runtime environments for grid computing. In Marian Bubak and Kazimierz Wiatr Michal Turalad and, editors, *CGW'07 – Proceedings of Cracow'07 Grid Workshop*, pages 155–162, October 2007. 5
- [3] P. Clough, M. Grubinger, T. Deselaers, A. Hanbury, and H. Müller. Overview of the ImageCLEF 2006 photographic retrieval and object annotation tasks. In *Working Notes of the 2006 CLEF Workshop*, Alicante, Spain, September 2006. 4
- [4] Adrien Depeursinge, Henning Müller, Asmâa Hidki, Pierre-Alexandre Poletti, Alexandra Platon, and Antoine Geissbuhler. Image-based diagnostic aid for interstitial lung disease with secondary data integration. In *SPIE Medical Imaging*, San Diego, CA, USA, February 2007. 1
- [5] M. Ellert, M. Grønager, A. Konstantinov, B. Kónya, J. Lindemann, I. Livenson, J. Langgaard Nielsen, M. Niinimäki, O. Smirnova, and A. Wäänänen. Advanced resource connector middleware for lightweight computational grids. *Future Generation computer systems*, 23(2):219–240, 2007. 1
- [6] R. Figueiredo, P. Dinda, and J. Fortes. A case for grid computing on virtual machines. In *Proc. International Conference on Distributed Computing Systems (ICDCS)*, May 2003. 5
- [7] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *The International Journal of Supercomputer Applications*, 15(3), Summer 2001. 1
- [8] A. Hidki, A. Depeursinge, J. Iavindrasana, M. Pitkanen, X. Zhou, and H. Müller. The medgift project: perspective of a medical doctor. *Journal of Medical Imaging Technology*, 2007. 1

- [9] Intel. Intel virtualization technology. *Intel Technology Journal*, 10(3), 2006. [7](#)
- [10] Hajo Krabbenhöft, Steffen Möller, and Daniel Bayer. Integrating arc grid middleware with taverna workflows. *Bioinformatics Applications Note*, 2008. [5](#)
- [11] M. Litzkov, M. Livny, and M. Mutka. Condor — a hunter of idle workstations. In *Proceedings of the 8th international conference on distributed computing*, pages 104–111, San Jose, California, USA, June 1988. [1](#)
- [12] J. Montagnat, V. Breton, and I. E. Magnin. Partitioning medical image databases for content-based queries on a grid. *International Journal of Supercomputer Applications*, 44(2):154–160, 2005. [1](#)
- [13] H. Müller, A. Garcia, J. Vallée, and A. Geissbuhler. Grid Computing at the University Hospitals of Geneva. In *Proc. of the 1st HealthGrid conference*, pages 264–276, Lyon, France, January 2003. [2](#)
- [14] H. Müller, D. McG. Squire, W. Müller, and T. Pun. Efficient access methods for content-based image retrieval with inverted files. Technical Report 99.02, Computer Vision Group, Computing Centre, University of Geneva, rue Général Dufour, 24, CH-1211 Genève, Switzerland, July 1999. [1](#)
- [15] Henning Müller, Nicolas Michoux, David Bandon, and Antoine Geissbuhler. A review of content-based image retrieval systems in medicine – clinical benefits and future directions. *International Journal of Medical Informatics*, 73:1–23, 2004. [1](#)
- [16] M. Costa Oliveira, W. Cirne, and P. M. de Azevedo Marques. Towards applying content-based image retrieval in clinical routine. *Future Generation Computer Systems*, 23:466–474, 2007. [1](#)
- [17] Mikko Pitkanen, Xin Zhou, Miika Tuisku, Tapio Niemi, Ville Ryyanen, and Henning Müller. How grids are perceived in healthcare and the public service sector. In *HealthGrid*, Chigaco, U.S.A., June 2008. [1](#), [2](#)
- [18] O. Smirnova. *Extended Resource Specification Language (XRSL)*. The NorduGrid Collaboration. NORDUGRID-MANUAL-4. [4](#)
- [19] Sechang Son and Miron Livny. Recovering internet symmetry in distributed computing. In *Proc. Cluster Computing and the Grid (CCGrid 2003)*, May 2003. [3](#)
- [20] Lilian H. Y. Tang, R. Hanka, and H. H. S. Ip. A review of intelligent content-based indexing and browsing of medical images. *Health Informatics Journal*, 5:40–49, 1999. [1](#)
- [21] X. Zhou, M. Pitkanen, A. Depeursinge, and H. Müller. A medical image retrieval application using grid technologies to speed up feature extraction. In *ICT4Health*, Manila, Philippines, February 2008. [1](#), [4](#), [4](#)