Laboratory Setup Status Report
January 2002

Roberto De Leo
deleo@crs4.it
Laboratory for Advanced Planning and Simulations
GEMMS Area
Energy and Industrial Processes
CRS4
(Center for Advanced Studies, Research
and Development in Sardinia)

January 2002

Abstract

In this document is presented the activity of Dr. Roberto De Leo for the year 2001. Its activities regarded the hardware and software upgrade and mantainance of the Laboratory for Advanced Planning and Simulation project (LAPS) and the software development activities in the framework of the development of tools for geometric reconstruction of carotid arteries.

1 Introduction

The activity of the author within the project LAPS has been four-folded: setting up (form scratch) of the machines (hardware and software), conversion of several projects to the autoconf/automake standard, support on the work relative to the geometrical modelling and geometrica reconstruction of arterial.

2 Machines Setup

At the beginning of 2001, to be able to cope with higher number of researcher working in the project, we incresead by three untis the number of workstation available in the project.

The OS the best suited our working needs has been recognized without any doutb as Linux: it is totally free, it comes with all tools needed for working in a network environment (because it is actually born abd developed on the net) and all tools for programming in all major languages, in particular C, C++ and Perl, and it is natively multitasking and multiuser.

Several reasons lead us then to buy PCs instead of workstations: the much lower prices, the higher compatitibility with Linux, the lower cost of hardware upgrades, or equivalently the higher degree of components recycling, the availability of a huge range of different video cards with "high end" 3D hardware support and the amazing speed of their present development: for example the newly released GeForce4, according to benchmarks publicly available on the net, for example at http://www.tomshardware.com/, has a speed that is double respect to the GeForce3 released in the first quarter of last year.

Given the funds availability, five PCs have been bought. In order to check the reliability of the AMD CPU brand, whose present-time in performance/price ratio is much higher respect to Intel, two of these five have a 1GHz AMD CPU. The others, two double CPU and a single one, have all 1GHz Intel PIII CPUs.

Up to now the AMD CPUs, much cheaper then Intel's, proved to be same quailty, if not better, then Intel ones.

Also with video cards, the other crucial ingredients of the machines, we made some experiment, buying two ATI Radeon 8550 cards, two NVidia GeForce2 GTS cards and a NVidia GeForce3 card. The first kind appears to be more suitable on linux, as ATI gives full support to GNU Linux developers and so ATI drivers are open source and included in all latest XFree86 releases, while NVidia still refuses to do so and develops on its own their own Linux drivers, so that no open source drivers are available for that brand and so the future development of such drivers depends fully on the company future decisions.

At this stage we did not stress video cards enough to be able to say which one fits better for our needs.

As Linux distribution we decided to adopt Slackware, in part because a good expertise was available within the group and in part because it is one of the soundest distros available and the policy of building on the machine all major software needed made totally unuseful any mechanism rpm-like of pre-packaged binaries, for which more structured distros such as RedHat or SuSE are famous among Linux distros.

As kernel we decided to use on all machines version 2.4.5, to avoid any possible problem coming from minor kernel differences. All kernels have been compiled on the machine tayloring its configuration to match the exact hardware installed on the machine (including optimization for AMD or Intel CPU). No kernel problem has been found up to now on 2.4.5 version.

On the machines with the Radeon video card it has been necessary to compile the DRI version of X directly from source as no support for Radeon 3D acceleration was available at the beginning of 2001. From simple Mesa-OpenGL benchmarking programs it turned out that the Radeon card is more or less equivalent to the NVidia GeForce2 and so the choice between the two should be dictated mainly by drivers availability reasons.

On every machine has been set up an automount service, allowing users to access easily each other's machine's home, that does not belong to the main Crs4 network and can be accessed only within the group microcluster.

In order to be able to access to MS Visual C++ and MS Office applications without having to reboot the machine, on all of them has been installed a trial version of VMWare virtual machine where on turn have been installed WinNT and Win2k and the relative application packages we needed.

To be able to have access to the machine's files from VMWare in the mor euser-friendly way, on every machine has been installed from the latest source package the samba package, allowing to access Unix directories and printers through the MS Network protocols.

After a severe virus attack to all Crs4 machines communicating through the MS protocol, every samba server has been brought to Security Level equal to "user" from the previous, too friendly, "share" setting. This way, every user needs to authenticate himself before accessing a Unix directory through samba, so that all viruses that tend to fill shared folders are ruled out.

The other fundamental packages installed from sources are all directly related with the research activity: DOC++, AliRoot, Qt and OpenCascade.

The first one is the package we decided to use as automatic documentation tool: it allows easily to include the documentation inside comments in a text files, for example in C or C++ header files, and put the comments in HTML files keeping track of hierarchy class structures (if any).

AliRoot is a complex Cern package oriented to the reconstruction and analysis of particle physics data. Its relevance is due to the fact that the RayCastcsg library, currently developed in the project. Few efforts have been put in making user-friendly AliRoot's compilation process. In particular its code has been developed on RedHat machines, famous for not respecting GNU standars and for including in their distribution software not yet stable that ususally makes not immediate the porting from RedHat machine to non RedHat ones.

The main problem we have found and solved after many efforts is that AliRoot package contained a call to the system function "__setfpucw", not accessible anymore to users in GNU standards, rigorously followed by slackware, so that it was impossible to compile the package on our GNU-compliant Linux machines. After a thorough research over the net and on the machine man pages about this system call, we were able to fix the problem and to submit the patch to the author, so that now AliRoot distribution should compile smoothly also on GNU compliant systems.

The Qt package is one of the two main open source widget C++ cross-platform libraries (the other being GTK), and by far the one with the better documentation. After going over all possibilities available (forms, OpenInventor, GTK, Qt) was clear that this package was the one that should be used for building the user-friendly interfaces to our future medical imagin packages.

After a few encouraging experiments, we got in touch with an other researchers group working with OpenCascade and using Qt as widget interface and we were able to get nice code examples so that now we have a QtCascade C++ module that enable us to embed a Cascade application inside a Qt canvas. Not all desired functions are still implemented but the most difficult techinical difficulties are now over.

The last package is for sure the most complex we use. It is a probably the best geometric modelling C++ library available and is open source, so that is possible to enter its code in debugging and when is not clear what a class or a function exactly does. The drawback is that the documentation is very scarce and so you actually have almost no other choice than open its header files if you want really to understand functions behaviour.

The compilation scritps bundled in the official release are very far from standard and to customize compilation flags and parameters is far from easy. On the other side the size of the package is really huge (the compilation process takes more than five hours on our machines) and to undertake the quest of generating autoconf/automake scripts appeared discouraging. Luckily we found out soon that it was already active a project to make this and so we were able soon to download a beta and then a stable package with the standard GNU autoconfiguration mechanism, that makes much easier to customize configuration options to our hardware.

To conclude the topic about software installation, we just mention that were also compiled on the group machines a few important packages as the latest versions of Xemacs, Perl, Apache Httpd and Kde.

3 Self configuration tools

One of the most important software packages for the group activity is the Slicetool package written by A. Giachetti. The object of the tool is to extract the contours from TAC data slices, so that it is possible from those contour to build the 3D models needed.

The tool had been written years ago and had an extremely complicated Makefile structure, having several Makefiles containing machine-dependent configuration parameters and several Makefiles for every possible machine architecture, makeing a very difficult task to update the files and to build new Makefiles for different architectures.

All linux Makefiles of the original package were aimed at RedHat machines, so it was needed to produce new versione of the Makefiles to make them work under the Slackware distribution. It became immediately clear that it would had been not too smart to spend lot of time trying to understand the complicated totally non standard Makefile structure just to produce a single new Makefile and have to do everything again once a new configuration would have appeared.

Instead, we decided to invest time in learning the "de facto" standard autoconfiguration mechanism provided by GNU with the packages autoconf and automake. The beauty of this mechanism is that, through the execution of the single script "configure", it checks by its own all machine-dependent compilation parameters, so that are not anymore needed different Makefiles for different architectures, and provides an easy way of setting compilation parameters through command line options of the configuring script.

The "configure" file itself is pretty complicated but can be easily generated thanks to a set of macros from a "configure.in" script, human readable, whose sintax is described in the autoconfig docs. There is no need to spend many words on the huge semplification that this way of generating Makefiles implies. For example it is possible to tell from command line whether we want or not to compile with the debugging flag on or we can set from command line the path of external includes or libraries needed for compiling, besides being able to tell the configure script to check the most likely locations for such external components.

To generate all Makefiles, the configure script needs a "Makefile.in" macro file for every Makefile to be generated. This kind of file can be again pretty complicated, but again we can simplify them letting the "automake"script generating them though other much simpler scripts called "Makefile.am", that basically contains a list of files involved in the package directory in which it stays and a few information about the kind of operation to perform. Also in this case an extensive documentation rich of examples is available online.

Among the many advantages of these approach to the mantaining of complex software packages, we like to cite the good number of "default" make options generated automatically by the configure file, among which an accurate "clean" method that remove all files generated during the compilation and a very useful "dist" method that automatically produces a "tgz" containing the whole source package.

The updated "Makefile" generating system of Slicetool this way worked so good that we decided to update in the same direction also the fairly complex package "Raycastcsg", written by P. Pili.

4 Bifurcations

The most complex operation in the reconstruction of a surface from its plane sections is by far the reconstruction of the bifurcations, i.e. the reconstruction of those components of the surface containing saddle points.

No satisfactory general algorithm has been found up to now, and indeed, while there are standard calls in OpenCascade to build a cylinder from two parallel plane loops, there is no standard function to build a pair of topological "pants" from a pair of non intersecting loops on a plane to a parallel loop.

Once OpenCascade has been satisfactorily installed on both WinNT and Linux a big effort has been directed at solving this problem. As it become clear the advantage of working under Linux and compiling with gcc rather than working on WinNT and using MS Visual C++, it became very important to develop a Qt interface to OpenCascade to be able to manipulate easily the objects, at least to be able to translate and rotate the 3D objects with the mouse.

After Solving the Qt-Cascade bindings problem as explained above, we were able to check the problems present in the first try of producing a surface given three loops on two parallel planes. This method is probably the simplest and consists just in generating the two cylinders connecting the two cylinders belonging to the same plane with the one in the other plane, making them solid, evaluating their union and extracting the lateral surface of the solid. Its problem is the fact that generically the surface obtained this way is not doubly continuous and there is no systematic way to smooth it.

An alternative way, still being worked on, is to build the pants as the union of four components, each of which topologically trivial: two components are the lateral pieces of the pants, that are basically half cylinders, and the other two are half each of the component left once we take out those lateral components, namely the component containing the saddle point.

Also this method has problems, mainly due to the impossibility of finding a generic algorithm to determine the position of a "canonical" saddle point, as the relative positions of the three loops can be such that the chosen point falls inside one of the two loops belonging to the same plane, which is a pretty undesirable condition, and also due to the difficulties in finding a "canonical" way to determine the boundaries of the components cited above.

On the other side data coming from TACs are not "generic" and it is reasonable to expect that data coming, for example, from arterial bifurcation behaves nicely enough, especially if the sampling is taken with a small enough Δz. It would be interesting to check on several cases the success of this kind of approach to understand how much it can be trusted.

5 Conclusions

A big effort has been made to build a robust 3D-geometry oriented GNU-compliant Linux microcluster for the LAPS project. Such microcluster has been now extensively checked out and it is properly working.

Important LASP's packages configuration systems have been ported to the standard GNU autoconf/automake style, makeing much easier the installation of those packages on the many diverse architectures present in the center.

Lot of work has been made to integrate the very complex OpenCascade geometric modeller OpenCascade with the cross-platform widgets library Qt and to produce Perl bindings for both of them. This last activity is not completed yet but is giving promising signs.