Administrative Health Data Mining using Debian GNU/Linux

Graham J Williams
CSIRO Data Mining
GPO Box 664
Canberra, ACT 2601, Australia
Graham.Williams@csiro.au

Abstract

Government departments are increasingly turning to GNU/Linux for their desktops, in addition to their servers. In this paper we look into using commodity hardware and software to manage and analyse large collections of data. CSIRO Data Mining has been using Debian GNU/Linux as a key platform for research and development in Data Mining for several years. During that time we have developed tools for rapidly accessing and analysing data and an environment for the analysis of very large datasets with delivery of results over secure web connections to clients as web services. All work has been performed on Debian GNU/Linux platforms with a variety of open source software including MySQL, Python, R, LaTeX, Apache, and locally developed wajig. In this paper we relate our successful experiences (and lessons learnt) with deploying Debian GNU/Linux, particularly in collaboration with a government department.

1 Introduction

CSIRO Data Mining has been using Debian GNU/Linux for many years for the management and analysis of very large collections of data. Datasets ranging in sizes of up to 10 GB are often processed, massaged into various shapes, and analysed using many different tools. Sophisticated tools and environments are required in order to manage data of such magnitude and we have worked toward a computing environment based on GNU/Linux running on standard PCs.

Data Mining brings together technology from databases, artificial intelligence, and statistics, amongst others. The aim of data mining is:

> The non-trivial extraction of novel, implicit, and actionable knowledge from extremely large databases, in a timely manner.

Data mining has been used in applications from fraud detection in credit card transactions, through web log analysis for intrusion detection, to identification of adverse outcomes in the use of combinations of prescription drugs.

By its very nature, Data Mining requires significant computational resources. But processor speed alone is not enough. Considerable data storage with fast access is required so that tens and hundreds of gigabytes can be processed quickly.

The original environment set up by CSIRO Data Mining in 1995 included proprietary hardware consisting of a 12 processor machine with a 500GB RAID array and 6GB main memory, running a proprietary Unix operating system. At the time this was quite a large and powerful computing resource, valued at approximately $500,000, and used for the development and deployment of software for analysing data, and data mining of data from clients including NRMA Insurance, Health Insurance Commission, Australian Taxation Office, and Mount Stromlo and Siding Springs Observatories.

Over the years we have moved from an expensive, specialised platform to commodity hardware/software in delivering an experimental network for a government department. The network provides a data server for managing several hundred gigabytes of data. Analysts process the data in various ways to provide specific datasets for policy research. The solution provides an efficient and extremely cost effective alternative to traditional proprietary solutions. A data server with 500GB RAID disk, dual 3GHz processors, and 4GB of main memory is quite adequate for significant data processing tasks, and today costs less than $15,000. Coupled with desktop machines with 3GHz processors, 1GB memory and 60GB hard drives at about $2,000 each, an environment for sophisticated data analyses of very large datasets is easily delivered.

In the following we identify Debian GNU/Linux and open source software as providing cost-effective solutions for accessing enormous collections of administrative data for the purpose of data analysis and data mining. The aim is to improve business processes and, for example, to improve the delivery of health care in Australia. The systems described here have been implemented within CSIRO Data Mining for our own use and also delivered in collaboration with a Government Department (Health and Ageing) for managing and analysing data. The systems provide

an environment for managing data at a fraction of the cost and with improved performance over competing solutions. We present our experiences in delivering state-of-the-art hardware/software services using Debian GNU/Linux. We record some of the software tools of particular use for the projects we are involved in and report on some of the outcomes of the work.

2 Linked Administrative Health Data

The Medicare Benefits Scheme (MBS) and Pharmaceutical Benefits Scheme (PBS) provide universal health care insurance for Australians. The schemes cost several tens of billions of dollars each year. Data relating to virtually every interaction anyone in Australia has had with the health care system outside of hospitals, since 1975, has been collected. The data is essentially for the administration of the insurance scheme (i.e., for paying for the medical expenses) rather than collected for analysis, yet there is a wealth of information contained in the data, about the delivery of health care in Australia.

Unfortunately, in terms of obtaining a complete picture of the health care of a patient, the administration of hospitals in Australia is the responsibility of the state governments. They each maintain their own databases of admissions to hospital, and rarely has this been made available, at the unit level (i.e., at the level of individual patient transactions), for linking with the MBS/PBS data.

CSIRO Data Mining, with the Commonwealth Department of Health and Ageing and the Queensland Department of Health, has brought together the MBS/PBS data with Queensland hospital admissions data, to provide a whole of patient view of the delivery of health care. This dataset has provided an experimental base with which to demonstrate the potential of data mining of administrative health data, particularly on a GNU/Linux platform.

The data consists of 5 years of MBS and PBS transactions and 4 years of Queensland hospital admission for all patients in Queensland, de-identified so as to preserve patient privacy and confidentiality. During the period that is covered, Queensland's estimated resident population was between 3.2 million (end of 1994-1995 financial year) and 3.5 million (end of 1998-1999 financial year). The dataset contains records for 1.1 million individuals who were hospitalised in Queensland between 1995 and 1999. There are 3 million hospital records in the data. For these patients there are 100 million MBS transactions and 60 million PBS transactions. For hospital records there are nearly 60 variables recorded, nearly 20 for MBS and 15 for PBS. Overall as comma separated text files these account for 500MB, 8GB,

and 4GB respectively (and slightly larger as MySQL database files).

On the issue of privacy, considerable effort has been expended to ensure patients can not be identified from the data whilst still maintaining a dataset suitable for research. All hospital records have exact dates removed and contain only days since first admission, with only a month and year of first admission being recorded. The dataset contains only encrypted identifiers and no names or addresses. Additional protection is provided by physical security for holding the anonymised datasets with all access being monitored and through encrypted connections (using **ssh**). All researchers with access to the dataset are also subject to criminal provisions of the Privacy Act for any deliberate attempt to identify persons in the data set! Maintenance of confidentiality is an important issue!

3 Administering GNU/Linux

Although hardware vendors have toyed with delivering GNU/Linux pre-installed, it remains the exception. While this situation continues to improve, GNU/Linux will for some time still require user installation on new hardware.

We took the approach of using inexpensive, off-the-shelf, PCs that are otherwise supplied with Microsoft Windows. Generally we have used new hardware and this has been a regular cause of "issues." Essentially, on every platform except the very basic, stock standard, we have had to deal with Linux kernels or XServers that were not quite there yet in their support of the hardware. This is an issue only when GNU/Linux does not come pre-installed.

Examples of "issues" include installing Debian GNU/Linux on a RAID/SCSI only machine before the default kernel supported the particular device. In this case the simplest solution, at that time, was to also install a standard IDE drive and, in fact, install GNU/Linux to that drive, then upgrade to a newer kernel. Similar problems arise with newer Ethernet cards, requiring a standalone install of GNU/Linux then installing a newer kernel from CD-ROM before network connectivity could be established. With the current release of Debian these are no longer specific problems, although even newer hardware could be!

Support of graphics chips is then the next problem that one comes across. While the problem is being addressed and only affects very few chips, it remains an issue. For example, recent desktop machines have a newer graphic chip that falls into the family of *i810* drivers under XFree86. The specific chip (82865G) is supported in 4.3 but not 4.2 of XFree86, and some action is needed, at present, to specially

obtain a copy of the xserver-xfree86 package, version 4.3 for Debian.

A key aspect of delivering GNU/Linux solutions is the administration overhead required. Debian GNU/Linux has always delivered a package management system, **dpkg**, that simplifies the task of the system administrator in terms of installing and upgrading packages. The **apt-get** framework that has been developed on top of dpkg has further provided sophisticated tools for managing a Debian-based system. The power of the approach is illustrated by the fact that the apt-get framework has now been ported to Red Hat to provide many of the same advantages for that distribution.

Nonetheless, remembering all the different commands to obtain different information about different aspects of Debian package management and system administration is still quite an exercise. Swapping between the many associated tools, such as *dselect*, *aptitude*, *apt-get*, *dpkg*, *synaptic*, *apt-cache*, and so on, is perhaps interesting but particularly cumbersome and annoying for the system administrator. Out of this plethora of utilities grew the **wajig** package for system administration.

Wajig was written as a tool to simply collect together all of the tricks and incarnations of various other tools that one learns from more than 10 years of using GNU/Linux. The aim of *wajig* was to capture, for example, how to reconfigure packages, how to stop and start daemons, how to update the Debian *alternatives* (like `editor`), how to find which package a particular file belongs to, how to list the available versions of a package, and so on! It has grown, over the years, into a comprehensive set of commands for managing a Debian GNU/Linux system, essentially from a normal user account, without explicitly requiring root login.

Wajig provides a single, comprehensive interface to many of the actions required to be performed by a system administrator. It currently includes nearly 100 commands. An indication of some of the supported actions include:

auto-clean	Remove superseded deb files from the download cache
auto-download	Do an update followed by a download of all updated packages
available	List versions of packages available for installation
bug	Check reported bugs in package using the Debian Bug Tacker
build-depend	Retrieve packages required to build listed packages
changelog	Retrieve latest changelog for the package
daily-upgrade	Perform an update then a dist-upgrade
dependees	List of packages which depend on the specified package
describe	One line description of packages
describe-new	One line description of new packages
detail	Provide a detailed description of package
dist-upgrade	Upgrade to new distribution

download	Download package files ready for an install
edit-sources	Edit the sources.list file which locates Debian archives
find-file	Search for a file within installed packages
find-pkg	Search for an unofficial Debian package at apt-get.org
install/dist	Install packages from specified distribution
integrity	Check the integrity of installed packages
large	List size of all large (>10MB) installed packages
list-alts	List the objects that can have alternatives configured
list-daemons	List the daemons that wajig can start/stop/restart
list-files	List the files that are supplied by the named package
list-orphans	List libraries not required by any installed package
new	List packages that became available since last update
news	Obtain the latest news about the package
orphans	List libraries not required by any installed package
policy	From preferences file show priorities and policy
purge	Remove one or more packages and configuration files
recommended	Install package and associated recommended packages
reconfigure	Reconfigure the named installed packages
remove	Remove one or more packages (see also purge)
remove-depend	Remove package and those it depend on and not required
repackage	Generate a .deb file for an installed package
rpm2deb	Convert a RedHat .rpm file to a Debian .deb file
rpminstall	Install a RedHat .rpm package
showinstall	Trace the steps that an install would perform
source	Retrieve and unpack sources for the named packages
start	Start a daemon, e.g., gdm, apache (see list-daemons)
status	Show the version and available version of packages
update	Update the list of down-loadable packages
upgrade	Upgrade all of the installed packages or just those listed
whichpkg	Find the package that supplies the given command or file

4 Debian Installation

Debian provides a rock solid, stable, distribution which is highly recommended for those running servers that must be reliable. This distribution has been thoroughly tested and there are many, and a growing number, of servers around the world using Debian. Software included in this distribution is, significantly, not generally leading edge, but instead mature software unlikely to ever crash! For the most conservative, this is the suggested installation, but not my personal choice.

Many people like to stay up-to-date with the "latest and greatest" software. This is particularly important for developers in the ever changing world of software. An alternative Debian distribution, the *unstable* distribution, is where you will find the "latest and greatest." Despite its name this distribution is in fact also quite solid. Certainly every now and again a package will be updated with unintended consequences, that rarely, but all the same occasionally do, render the system somewhat crippled. Serious problems, it should be stressed, are rare but can happen at any time. The saving grace is that many of the very large team of almost 2000 Debian developers world wide (as well as a very large population of Debian users) update their "unstable" systems daily so that any problems generally get fixed quite quickly, and usually overnight.

Whenever problems arise with Debian the first place to turn to is the Debian bug tracking system. Here, problems will be reported quite promptly, and temporary workarounds identified while the problem is fixed and the packages are updated. *Wajig* provides simple access to the bug reports for any package:

```
$ wajig bugs gnome-applets
```

A third distribution, in between the stable and unstable distributions, is the testing distribution. This provides a compromise between stability and bleeding edge. Packages in this distribution have not been through the rigorous testing of the stable distribution, but are generally stable enough to not have serious bugs reported against them. In fact, packages automatically migrate from the unstable distribution to the testing distribution after a certain period of time in unstable, and there being no serious bugs reported against the package. The testing distribution eventually works its way to be the next stable release for Debian GNU/Linux.

In setting up a network of Debian GNU/Linux servers and desktop machines a suggested philosophy is to cater for the variety of needs of all users, while maintaining a stable and productive environment. A key issue is that, while a shared server must be carefully managed to ensure stable and reliable delivery of services, and while some users are quite happy to stay with a stable desktop environment, others require, or perhaps just enjoy, the delights of living at the cutting edge. *Wajig* is an effective system administration tool providing for these extremes.

For desktop workstations the general installation allows the *owner* of the desktop (the person who sits in front of the machine) full administrative power through *wajig* and its use of *sudo* for particular privileged operations. In this way the user who likes to remain up-to-date can choose the *unstable* distribution and daily upgrade their system, perhaps with:

```
$ wajig update
$ wajig dist-upgrade
```

Or to simply download the newly upgraded packages unattended and then to later upgrade the distribution when they can attend to it:

```
$ wajig auto-download
```

Other desktop users will choose to only upgrade individual packages as required with the traditional *install* command:

```
$ wajig install python
```

For shared servers, stability and reliability of service are crucial. It is suggested that these machines only ever have specific packages upgraded, as needed. Most often these are specific security updates, while the rest of the system remains untouched.

Generally, one server in the network will also archive any Debian packages that any of the other hosts have downloaded for installation. This archive is then made available to all other hosts on the network so that any upgrades will first look there for the available packages.

Overall, Debian GNU/Linux provides distributions that suit all needs, while simplifying the administrative tasks, with a tool like *wajig* sitting on top of the underlying system to simplify the system administrator's job.

5 Data Analysis and Management

Data mining was introduced above as the non-trivial extraction of novel, implicit, and actionable knowledge from extremely large databases, in a timely manner. Indeed, data mining works with very large datasets possibly drawn together from many sources. Data management is a significant issue as is the reporting of results, and automated tools for generating summaries of large data sets. Data analyses are also performed over the very large data sets. We briefly describe here the

collection of software tools used for these tasks. We finish with a summary of outcomes from projects using these tools.

Python has turned out to be an extremely flexible, efficient, and clean language on which to base our environment for data management and data mining. Having started out with Java in the early days of our data mining activities, Python was a pleasant surprise. While Java originally offered many benefits, including portability, rapid development, networking, it has become quite cumbersome, complex, and remains less efficient than other options. Python provides many of the same advantages (including portability and object-oriented), while remaining much more like a scripting and rapid prototyping language. Access to many standard C and C++ libraries and toolkits gives it considerable flexibility. And software development time using Python is dramatically reduced.

Consider the task of generating some graphic displays. The standalone R package provides sophisticated statistical processing and graphing functionality and is easily accessible from within a Python program. The follong Python program produces the plot shown in Figure 1.

```
from rpy import *
r.pdf('rplot-dot.pdf')
dates = ('Jul-98', 'Aug-98', 'Sep-98', 'Oct-98', 'Nov-98', 'Dec-98',
         'Jan-99', 'Feb-99', 'Mar-99', 'Apr-99', 'May-99', 'Jun-99')
death30 = (2.02, 1.53, 2.73, 3.09, 2.37, 2.60,
           3.87, 6.11, 3.23, 4.52, 4.27, 1.40)
death6m = (1.52, 2.55, 3.28, 1.55, 0.95, 3.65,
           4.42, 5.68, 8.29, 15.08, 32.70, 75.52)
r.par(cex=1.5)
r.plot(death6m, type='b', xlab='Month', ylab='Percentage',
       lty=1, pch=0, axes=False)
r.lines(death30, type='b', lty=2, pch=1)
r.box()
r.axis(1, at=range(1,len(dates)+1), labels=dates)
r.axis(2, at=range(0,100,10))
r.legend(1, 60,
         ('Deaths before 30 days', 'Deaths before 6 months'),
         lty=(2, 1), pch=(1, 0))
r.dev_off()
```

Listing 1: Simple dot plot.

The full power of the model building and statistical testing of R is, for example, available within the familiarity of Python. Graphical interfaces, Database connectivity, XML, SOAP, web services, and much more can all be dealt with within Python.

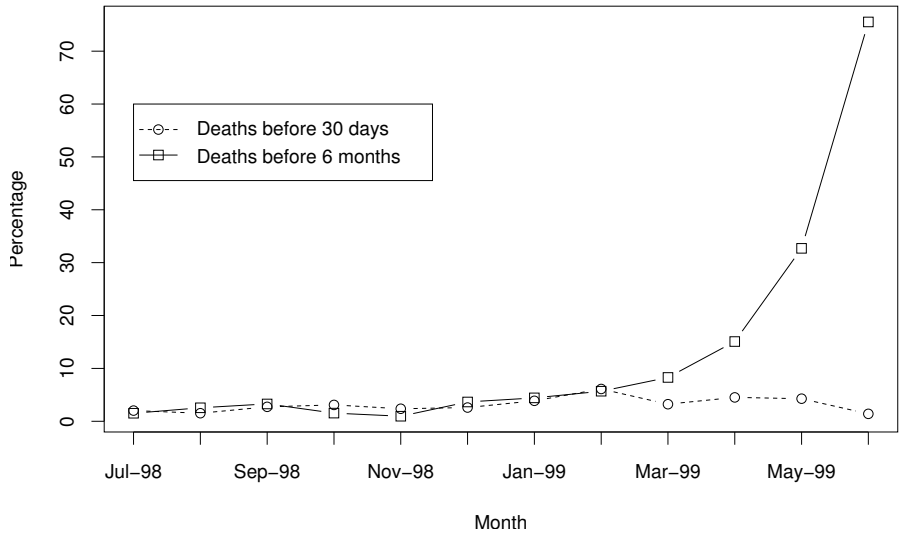

Figure 1: Simple dot plot.

For a project with the Health Insurance Commission a *Pathology Explorer* was developed. The tool provides a system for graphically interacting with large collections of administrative Medicare data. The user can explore characteristics of different patterns of behaviour exhibited by, for example, different pathology laboratories. Surprising observations are revealed through graphic presentations of temporal patterns of behaviour. The Pathology Explorer delivers its results over the Internet as a web service using XML/SOAP. The data server is a Debian GNU/Linux host running apache and MySQL to access the data. Delivery over the Internet is to any browser or Web Services enabled application, using secure/encrypted connections, driven by **apache-ssl**. Figure 2 shows an example interface to the system.

Another study has identified adverse reactions to drugs that are exhibited within the data. Systematic monitoring of adverse drug reactions is important for both financial and social reasons. In general, the early detection of unexpected adverse reactions relies on a local voluntary reporting system and through monitoring of overseas experience. However, the availability of the administrative data provides a unique opportunity to detect common and rare adverse reactions early. Statistical analysis and data mining of such transaction data provides an alternative way of identifying the adverse drug reaction. We demonstrated this using a specific class of drugs known as ACE inhibitors (used to treat congestive heart failure and high blood pressure), and identified their association with a specific hospitalisation for angioedema (a swelling beneath the skin). Figure 3 illustrates sample distribution and model generated from the data. All data management and analyses were

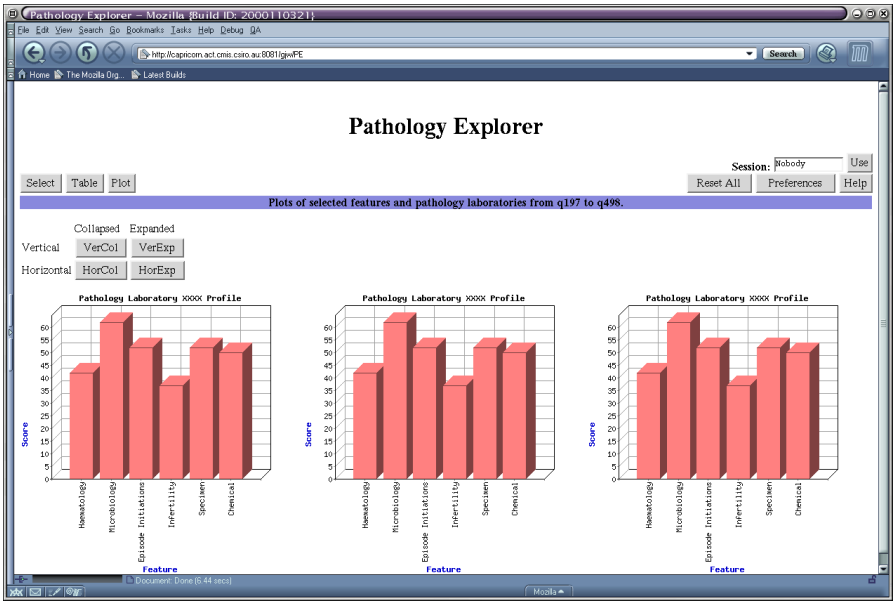

Figure 2: Sample interface to Pathology Explorer.

performed using Python and R, with specific model building tools developed for GNU/Linux.

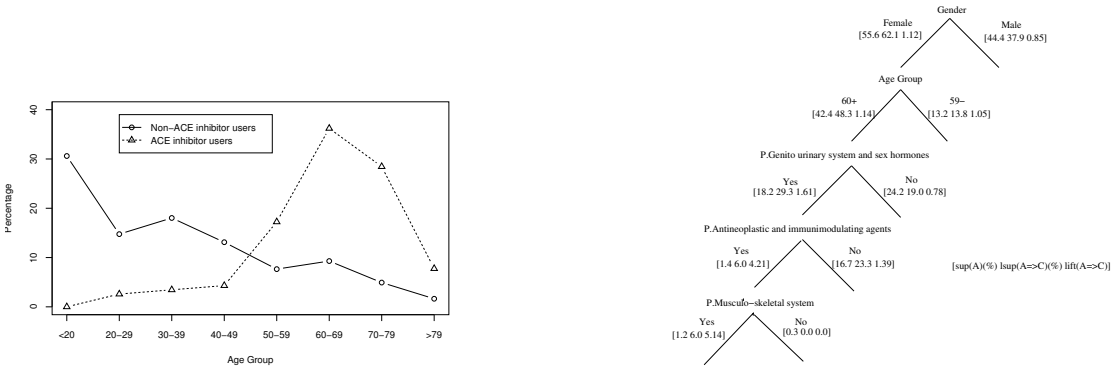

Figure 3: Sample outputs of adverse drug reactions study.

Finally, to illustrate further Python functionality and a common task, we present a screen shot from a growing collection of GUIs for traditional command line tools. In this case a GUI has been developed using **Glade** to generate a Gnome-based interface with the underlying functionality implemented in Python. Glade directly generates Python code. The functionality simply extracts the information from the GUI to build a command line that is then executed, in order to generate, in this case, a decision tree model of the data. A simple exercise in building such

a GUI, as it is using Glade and Python, delivers considerable benefit for users of the underlying command-line based application. The interface is self documenting, using tooltips, and provides immediate access to all options available, as illustrated in Figure 4.

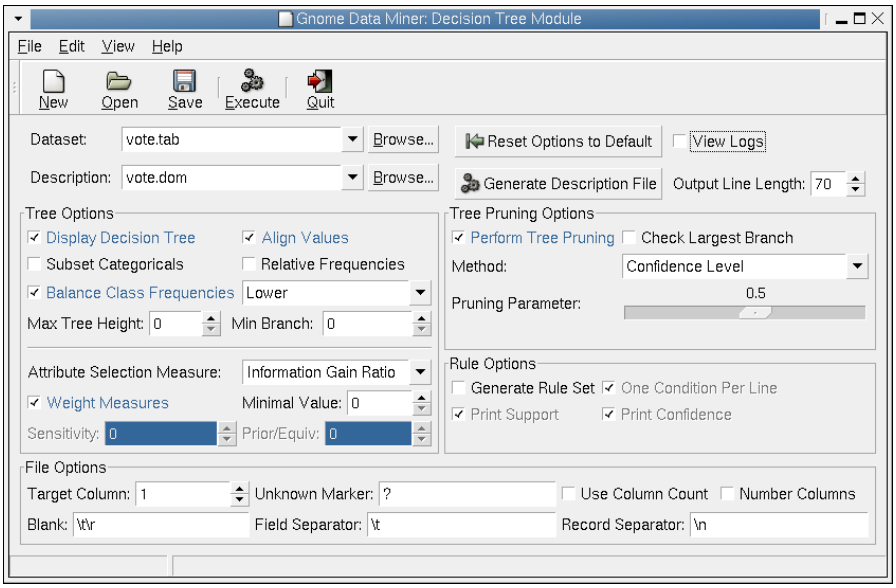

Figure 4: Sample interface to command line tools.

6 Conclusions

With many governments and organisations world wide re-assessing their information technology needs and looking for cost effective solutions, the future of open source systems is indeed quite bright. We've presented but one case study here of developing and deploying an environment for the analysis of very large collections of data as are common in many government departments and industry. The case study demonstrates that GNU/Linux is ready for both servers and the desktop. On the desktop it provides powerful tools to analyse data and to report on those analyses, while accessing and managing data on servers. We have presented our experiences and report on a tool we have developed to simplify the task of managing a Debian GNU/Linux system for both users and system administrators.

7 Acknowledgements and Short Biography

I am grateful to Dr Christopher Kelman, Medical Adviser to the Commonwealth Department of Health and Ageing, for his enthusiasm for exploring new approaches to delivering research within a major Government Department. Richard Solon of the Commonwealth Department of Health and Ageing was instrumental in building the Debian GNU/Linux environment for the research group within the department and has journeyed with me along the road of Debian GNU/Linux.

Dr Graham Williams is principal computer scientist, CSIRO Data Mining. Graham has lead many data mining projects for clients including the Health Insurance Commission, the Australian Taxation Office, the Commonwealth Bank, NRMA Insurance Limited, the Commonwealth Department of Health and Ageing, Queensland Health, and the Australian Customs Service. He has developed software and hardware environments for data mining, and implemented web services for the delivery of data mining. He has worked with Linux, and particularly Debian GNU/Linux, for 10 years, and with GNU software for almost 20 years. He is also the author of the wajig administrative tool for Debian GNU/Linux and author of Debian GNU/Linux Desktop Survival (`http://togaware.com/linux/survivor`).