Developing a Multimedia Distribution

Norsk Regnesentral
ANVENDT DATAFORSKNING

Norwegian Computing Center/Appled Research and Development

NOTAT/NOTE

DART/02/02

Wolfgang Leister
Eirik Maus
Bjørn Nordlund
Ole Aamot
Joachim Lous

Oslo
December 2002

NR Norsk Regnesentral
ANVENDT DATAFORSKNING

Notat / Note

Tittel/Title:
Developing a Multimedia Distribution

Dato/Date: December
År/Year: 2002
Notat nr/:
Note no: DART/02/02

Forfatter/Author:
Wolfgang Leister, Eirik Maus,
Bjørn Nordlund, Ole Aamot
Joachim Lous

Sammendrag/Abstract:

Open Source software is becoming increasingly interesting. While we recognise that multimedia applications are a vital part of systems today, many multimedia applications are not yet distributed as open source products for various reasons. Therefore, NR has created an open source add-on distribution for Linux **mediAkit** with an emphasis on multimedia software. This software can be used in a number of application areas. The goal of our development is to provide an add-on to a standard distribution, so that the user can easily install and use multimedia applications.

In this report we present the principles that **mediAkit** is built upon. Our approach can be extended to application areas other than multimedia, including that of distribution of customised software packages that are developed for customers.

The report also contains several overviews on software for building a distribution, and a selection of multimedia software available for the Linux operating system. The portal pages of **mediAkit** are available on `http://mediakit.nr.no`.

Emneord/Keywords: Multimedia, Distribution, Open Source, Linux

Målgruppe/Target group: NR, research institutions

Tilgjengelighet/Availability: Open

Prosjektdata/Project data: Channel S, Setup

Prosjektnr/Project no: 11012

Antall sider/No of pages: 58

Norsk Regnesentral/Norwegian Computing Center
Gaustadalléen 23, Postboks 114, N-0314 Oslo, Norway
Telefon (+47) 22 85 25 00, telefax (+47) 22 69 76 60

Contents

Part I
The Setup Project

1 The Setup project

1.1 The "Channel S" framework program

This document is the result of projects at Norsk Regnesentral having the joint goal to learn about open software, and to prepare a distribution of open source software with an emphasis of multimedia. Besides collecting suitable software for a distribution, we develop the software necessary to make an easy installation of the distribution possible.

This project is called "Setup"; it is part of the Channel S research program at Norsk Regnesentral in 2002. Other research projects within the Channel S program might also be of interest for the reader. These include work addressing digital TV, image encoding with wavelet technology, and multimedia multi-channel production[1]. Especially the project on digital TV [7] is of relevance, as parts of the software catalogues in both projects contain common software projects. The project will be continued in 2003.

1.2 Open Source software and Linux

The "Setup" project has as a goal to look at Open Source software with an emphasis on multimedia. Open Source software has been popular for some time. The ideas upon which Open Source are founded have existed for several decades. In former days it usually was called "Free Software". Often, idealistic ideas were behind these thoughts, and not strictly technical reasons.

The GNU project [12] (see `http://www.gnu.org`) is a direct successor of this generation, and major components of what many people now call 'GNU–Linux', have been developed by the Free Software Foundation (FSF) and its volunteers for the GNU[2] system (see `http://www.gnu.org/philosophy/categories.html#TheGNUsystem`) since 1984.

In 1991 Linus Torvalds announced to the world that he was working on a Minix-like operating system kernel. The development methodology of the Linux kernel was different from the GNU project [12]. This is described in the article "The Bazaar and the Cathedral" by Eric S. Raymond [11]. The type of software development is often summarised in the motto: "Release early, release often".

After the Linux kernel became very robust, more developers adapted the Linux development model and the way of distributing software, and the Linux project became a model for open source development. Other large projects like Mozilla, StarOffice / OpenOffice.org followed.

Open source as a business model was released about the year 1999, with Eric S. Raymond as a spokesman [11]. Several program packages have been released using the open source model, and open source products are more often used in business, public administration, and schools (see, e.g., the Skolelinux project [1]).

The main thought behind open source products is not necessarily, that these are at no costs. While most of the software comes with a license that permits use of the software for free, installation, maintenance, support, and the use of other services usually causes costs. It would be dangerous for enterprises to neglect these costs, even if these costs are to pay employees for their time.

The real issue for open source is that the source code is openly available, i.e., that programmers can inspect the code, possibly make changes, and give those changes back to the community. The characteristics of open source software gives several advantages:

- The user can inspect the code and consider the quality of the code.
- The user can see whether problems could occur, e.g., with regard to security issues.
- The user can make changes, adjust the product for his use, and fix bugs that might occur.

[1] For more information, see http://www.nr.no/channelS/projectPlans/projectPlan2002.html.

[2] GNU refers to a recursive acronym for "GNU's Not Unix!".

The development model of open source projects is often different from the usual development projects. The project owners are responsible for coordination, maintain a database, take decisions, perform some of the programming, etc. Besides the project owners there are the users of the system, some of whom also play a role as contributors, when they make changes. For most of the projects there are web pages available, including possibilities for download, information on the project, communication between developers, addresses for reporting bugs, etc. There are also communities that house many projects, e.g. Sourceforge (`http://www.sourceforge.net`) and Savannah (`http://savannah.gnu.org/`).

The distribution model is often done by using web servers (or ftp servers), where the software is available. Often the project's web server provides a download facility. Many projects also have mirrors in order to avoid overload of the download servers.

When software packages are bundled together, the term **distribution** is used. For Linux many distributions are available; a guess is that there are over 300 distributions with emphasis on different subjects available (see `http://www.distrowatch.com`). Distribution projects with an emphasis on multimedia are presented in Section I.

1.3 Open Source multimedia projects for Linux

For many application areas open source software is available, including multimedia software. However, the setup procedures for this software can be too complicated for the common user. Therefore, in reality this multimedia software is not yet available to the broad public. Many packages require special skills to set up. One of the main problem is that the packages are often not integrated and depend on special resources such as dynamic libraries.

We take a look at the wider area of multimedia, in order to develop a distribution. Our distribution is meant as an easy-to-install add-on to the ordinary distributions. We feel that there is not yet a distribution available with that emphasis. There are other interesting areas that could have been chosen: E-learning, E-government, subjects from sciences (biology, medicine, physics, mathematics, etc.), but these subjects will be kept for future work.

Of course, there are already other initiatives within different application areas. Most of these are repositories on a web site using some search engine, and mechanisms for contribution of software. Examples are `Mathtools.net` (`http://www.mathtools.net/`, Linux Online (`http://www.linux.org/`), The GNU Free Software Directory (`http://www.gnu.org/directory/`), freshrpms.net (`http://www.freshrpms.net/`), Sourceforge (`http://www.sourceforge.net`), and several others.

Soon after the CDROM was detected as distribution medium the "Prime Time Software for Unix" [8] and "PowerTools for Unix" [9] were released as distributions of software.[3] These can be seen as early attempts to create a distribution of useful software for the user. A book or booklet, and installation programs followed with the CDROMs, in order to make it easier for the user to install the programs.

Several other multimedia distribution initiatives exist. Examples include the AGNULA project, the DeMuDi, or Planet CCRMA at Home. More information on these distributions is given in Section I.

When we speak of distribution, our goal is not a stand-alone software (which includes operating system), but to be a supplement to the operating system. Therefore, our specifications are different to major Linux distributions, like RedHat, SuSE, or Debian.

Today the availability of multimedia software is vital for an operating system to succeed in the market. By several reasons, multimedia software is not always available for Linux. Not all decoding algorithms for multimedia are available as open source[4]. The unavailability of some kinds of multimedia software makes that users no longer use Linux as "their" operating system.

To show an example, we look at the possibility to play DVDs on Linux. While the technology is available, there are laws that forbid the distribution of the unscrambling system for DVDs (called CSS), especially in the United States. For Europe the laws are more unclear on whether it

[3] In former Red Hat releases there has also been a Powertools CD as an add-on.

[4] Many of the formats are patented or closed, in order to prohibit knowledge of the software. Therefore, decoding software cannot be developed legally as open source.

is possible to distribute the DeCSS software. The DVD players available play only unscrambled DVDs.

A commercial DVD player for Linux has been announced for a long time: LinDVD by InterVideo with a price tag of about 30 US$. However, a look at the web pages of the provider shows that LinDVD is only available for manufacturers for evaluation and integration. Therefore, in reality there is still no legal possibility to play all DVDs on Linux.

Other obstacles for the introduction of multimedia software in Linux includes patent issues and closed media formats, where reverse engineering is prohibited. Sometimes Win32-libraries are altered and used, but also this possibility can cause copyright problems.

There are also technical reasons why it is hard to distribute multimedia software. Often the processing power necessary to present multimedia content is rather high, and special optimisations are required with respect to hardware (e.g., CPU, cards, periphery). These requirements often need a more profound knowledge than the common user has. Auto-detection techniques with respect to software configuration is often not developed to the necessary extent.

1.4 What is a distribution

According to the dictionaries the term **distribution** refers to the commercial activity of transporting and selling goods from a producer to a consumer.[5] The Jargon File [14] (version 4.2.3, 23 Nov 2000) gives the following definition:

> 1. A software source tree packaged for distribution; but see *kit*. Since about 1996 unqualified use of this term often implies 'Linux distribution'. The short for **distro** is often used for this sense.

The term **kit** refers to the following:

> [Usenet; poss. fr. DEC slang for a full software distribution, as opposed to a patch or upgrade] A source software distribution that has been packaged in such a way that it can (theoretically) be unpacked and installed according to a series of steps using only standard Unix tools, and entirely documented by some reasonable chain of references from the top-level README file. The more general term *distribution* may imply that special tools or more stringent conditions on the host environment are required.

The definition of the word *kit* already contains some hints on how an early distribution of software was organised, and what it contained. The expectations of the user are higher today with respect to automatic setup routines, and therefore a distribution in that manner would no longer suffice.

[5] There are also some other meanings of the word which are not relevant for our purposes.

2 Developing a distribution

The purpose of the Setup project is to provide an easy-to-install, easy-to-use distribution of multimedia tools for Linux, covering a range of uses. This section will examine how various pieces of software are created, interact, and are distributed. The purpose is to cover the challenges that a multimedia distribution has to overcome.

A distribution contains several software packages that can be installed on a computer. A distribution has usually a unified installation and maintenance procedure for all software packages. Therefore, besides the software, the distribution includes an infrastructure part, which includes: installation procedures, a data base, query scripts, software configuration tools, package management, etc. Also a bug tracking data base, tools for cooperative work, and a Web server for storing the distribution are parts of the total project content.

The provision of a distribution is twofold: (1) collecting, configuring and packaging the software, and (2) installing the software at the end user site. Both steps have the need for tools, and both steps must play together like in a client-server model.

In the packaging phase the following steps are performed:

- configure software packages, find dependencies, retrieve dependent software, compile and test the software, etc. This step should be heavily supported by tools, e.g., *make*, configuring tools, integration test tools, etc.
- the software packages are packed together using a package management system (PMS).
- Software for installing the distributed software packages at the client side, and to perform queries must be added to the distribution.

On the end user side the following steps must be supported:

- Queries whether the software already is installed, and in which version;
- unpacking the software, configuring, compilation and installation.

In the following we explain the necessary steps and decisions necessary for developing the distribution.

2.1 Steps for distributing a software package

Each running software package consists of a set of files that contain the executable code, library functions, setup information, etc. Multimedia software has a more complex setup than ordinary programs, because of possible plugins, codecs, etc. All files are generated from a code base in some source code. In order to create the executable programs tools have to be used, e.g., software configuration tools, compilers, linkers, ...

The setup procedures of the distribution must support the generation of the necessary files from the distributed file of the software package. All these files need to be suitable for the appropriate version of the software to be installed. When installing a software package, the administration software of the distribution must unpack the file, detect dependencies between files, possibly require other packages, compile, configure and install the package.

In the following we shall not present how to develop multimedia software. However it is vital to know that complex software usually contains many files containing interfaces[6], and functions. In the compilation process these files are translated into object files and libraries (see Figure 1), and linked into the executable code (see Figure 2). Dynamic libraries (in Linux these use the suffix `.so`) are linked during run time of the program, and have to be placed at specific places in the file system.

More information on compiling and linking programs is beyond the scope of this report. We refer to the documentation and manuals for the specific operating system, compiler and programming language.

Dynamic libraries are used to prohibit duplication of code in the computer, to avoid huge program files, and to make it possible to update software without updating all programs. Additionally, plugins are often implemented using dynamic libraries. Despite of the many advantages of dynamic libraries, there are some some pitfalls connected to the use of dynamic libraries:

[6] Interface files are often called header-files.

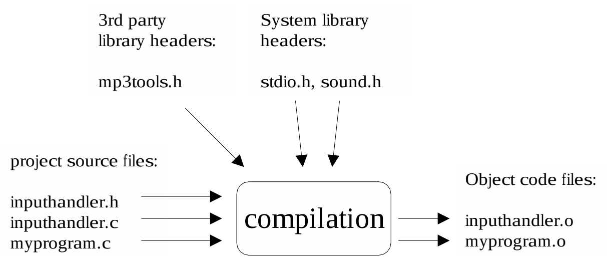

Figure 1: Compilation

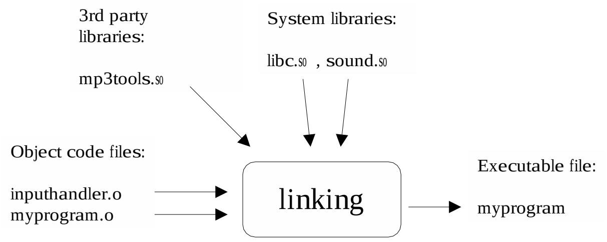

Figure 2: Linking

- Functionality is used in several software packages, which creates a hard to overlook net of dependencies.
- Software packages are dependent on having specific versions of a dynamic library installed, while other software packages expect another.[7] Therefore, upgrades should not remove obsolete versions of a dynamic library as long as some software is dependent on it.
- In order to ensure installations to be complete, and programs actually will work after the installation, the allowed versions of the specific libraries must be specified when packaging the software. During installation, it must be possible to query and update a database of installed modules on the system.

The need for being able to handle several versions of a library introduces additional complexity to the installation procedure. To make it easier to handle this complexity, different systems for managing packages are available, see sections B and 2.3.

2.2 Tools for handling compilation complexity

The dependencies between the different source files can become quite complex in big projects. Several libraries may use the same source file, they may reference to each other. Several tools are available in order to release the programmer from the task to type compile commands, and to remember which compilation steps are necessary. A utility called *make* is often used for this purpose. Other important tasks include the configuration of software packages according to available hardware, installed software, and the end users wishes.

The concept of *make* goes back to S. Feldman in 1979 [6]. Make reads instructions from a `Makefile` describing the various pieces to build in the software package to build, its dependencies, and how to compile the software package. See also Section A.1.

For complex program packages the creation of makefiles can be a complex task. Therefore, the GNU project [12] has created a set of tools to ease the development and distribution of software across Unix platforms. They provide means to distribute program packages in source code and have them easily configured on a different destination system, built and installed.

The use of these features is a two phased process: In the first part a software package is prepared for distribution, using these tools. The purpose of the preparation phase is to create a shell script program that automatically locates the needed packages, libraries and programs, the

[7] In Linux this is partially solved by installing several versions of a dynamic library, using the version numbering in the file name.

location of system files etc., and then prepare a suitable Makefile so that the package can be built and installed (hopefully) without errors.

A variety of tools can be used to extract information from the target system, the source code files and various libraries installed, and many files get written and read by the different tools. However, quite often the developer has to edit the files created by the autotools manually.

The second phase is to configure a software package for the end user, build and install it. In general, it should be sufficient to execute a configuration program, tell the make system to build the software and tell it to execute necessary instructions to install the built pieces.

See Section A.2 for further details.

2.3 Package management systems

A package management systems (PMS) is used to pack and distribute software, mainly pre-built libraries and programs in binary form.

Reasons for distributing software packages in compiled form (vs. distributing the source code) include:

- Configuring, compiling and linking is time-consuming and prone to errors. Installation can be done much faster if the modules are already pre-built.
- Most users only want to use programs, not alter them. Thus, they don't want or need the source code. This is especially true for desktop users of Linux. Some newer Linux desktop installations don't even install development tools such as compilers per default (unless the user selects "install all"), since many users don't intend to do any programming anyway.

However, by technical and juridical reasons, software we decided that it is more useful to distribute in source code (vs. compiled version).[8]

PMS are tool suites that ease distribution of software packages, managing dependencies and version control. These are normally built around a database that keeps track of installed modules, provides version information, and resolves dependencies between the various modules. A PMS also provides means to query a database, install new modules (including an update of the install-database), and deinstalling modules (if possible due to dependencies). An overview of PMS is given in Section B.

An overview of binary package formats used for Linux platforms can be found at `http://www.kitenet.net/~joey/pkg-comp/`, where the formats `deb` (used by Debian), `rpm` (used by Red Hat), `tgz` (used by Slackware), `slp` (Stampede Linux), and `pkg` (SVr4 package format for Solaris) are compared. Debian and the Norwegian effort "Skolelinux" [1] (see `http://www.skolelinux.no`) is built on top of the **Dpkg** system developed by Debian (see section B.1), while other major Linux distributions, including Red Hat, Mandrake and SuSE, are based on the Red Hat Package Management System RPM (see section B.2).

Many Linux distributions are based on RPM [2]. However, RPM does not provide configuration, building and installation from source code packages using a *single command*. Since desk-top end users like simplicity, this policy favours distribution of software as pre-built binaries.

PMS are usually not able to inter-operate, nor do they share the same data base on installed modules. The distribution format of the PMS is based on different file formats. Therefore, since coordination of the installation is the most vital part of a PMS, each system should have only one PMS in use. Between some of the distribution package formats converters exist, e.g., Alien (see `www.kitenet.net`) which converts between deb and rpm formats.

2.4 Linux kernel tuning for multimedia

The Linux kernel is the core of a GNU-Linux system. The kernel manages the hardware and the programming interfaces to it for the user space programs. For the mediAkit distribution we do not want to ship a special kernel, like other multimedia distributions do (see Section I). However, it should be mentioned that several kernel modifications exist to enhance low-latency and real-time

[8] The juridical reasons include the fact that some software, e.g. software for encoding MP3 streams, cannot be distributed as binaries. The technical reasons include possible optimisations of code, which could be vital for multimedia software.

performance. Most of these modifications are available as patches against the kernel, maintained by Linus Torvalds.

The following possibilities for enhancing the multimedia performance are available:

Linux scheduling latency. Latency can be defined as the elapsed time (delay) between the generation of an event and its realization. If the delay is great enough to be perceptible, there is a latency problem [10]. More information is available at `http://www.zip.com.au/~akpm/linux/schedlat.html`

Preemptible kernel. TimeSys Corporation claims that their Linux GPL preemptible kernel patch reduces the average preemption latency to about 1,000 microseconds, and also states that (contrary to some belief), the preemptible kernel does not make Linux into a Real-Time Operating System [4]. See also `http://www.kernel.org/pub/linux/kernel/people/rml/preempt-kernel/`

2.5 Multimedia software projects out there

An important part of the creation of a distribution is to collect the relevant software. This includes to take decisions which packages to choose, and to classify and prepare the packages. In order to find relevant projects search engines at relevant sites (e.g., `http://www.linux.org/apps/`) or sites that host projects (e.g., sourceforge) can be consulted.

It is a general observation that there are many open source multimedia projects available, but many of these are not (yet) included in "standard" Linux distributions. In our current project, we have gathered an extensive list of what seems to be the most relevant and useful open source multimedia of these projects.

For many of these projects the status is unclear; some projects are active, while others are abandoned, are no longer supported, or have died. The quality of many projects can be lower than what can be expected from commercial products, especially with regard to documentation.

Especially within the area of multimedia legal and patent issues may be associated with some projects and software packages. When choosing software packages for a distribution this must be taken into account, and we have to decide carefully whether a package could be included to which terms.

An extensive list of possible software packages is listed in Part III of this report. We are aware that this list cannot be a complete list of all available software within multimedia. Our intention has been to collect relevant open source software so that the a user without special knowledge of how to set up this software can use multimedia on her computer.

Our list of software includes candidates for the distribution; not all of these are included in the mediAkit distribution. For many multimedia tasks several projects with similar functionality exist. In some cases we include several packages with the same functionality. Sometimes the products have their strengths in some areas, while the implementation could be unstable at other parts; another project could have the opposite problems[9].

In general it is no problem to include a package in a distribution. However, we must provide the user with some means to decide which package to choose for a certain task. Therefore, the distribution must contain a tool to ease the decision process for the user when installing packages from the distribution.

[9] An example for this is the observation that some Web browsers do not work for some Web pages. Usually the users use an alternative browser when a page does not work, in order to access content.

3 Solutions chosen for the Setup project

This section covers decisions and solutions made in the Setup project. The objective of is twofold: explain decisions on what software to distribute, and to present technical decisions on routines and maintenance of the distribution.

3.1 Software distributed in mediAkit

We emphasise software to produce, store, serve, stream and play multimedia (sound, images, video, etc.). However, the principles and routines developed for 'mediAkit' can also be employed for distributions with other content.

Multimedia is a quite general term. We have focused on software for capturing using microphone and camera, for editing such material, for playback, encoding and for distribution of it. We have gathered an extensive list of relevant software projects. See the contents of our software catalogue in Section III for a extensive list of subjects included in mediAkit.

3.2 Principles for mediAkit

During the development of the mediAkit distribution we emphasised on the following:

- mediAkit must provide routines for preparing software packages that are newly added to the distribution. Routines on how to include new versions of already included software must be deployed. This includes recipes how to handle software packages in general.
- mediAkit distributes all program packages in source code. Besides avoiding some juridical reasons, this allows to generate optimised versions for the target machine. However, we recognise that this procedure is more time consuming during installation. Additionally, due to dependencies the distribution of source code could generate several problems while creating the packages.
- A build-system for distribution and installation must be employed which handles interdependencies, hardware- and software dependencies.
- Bug handling system, forward bug reports, or suggestions for problem solutions to those responsible. The system must have room for discussion about problem reports in order to let users propose patches.
- Maintenance of distribution: Check whether new versions, bug fixes, etc. of original software are available.
- Maintenance phase: automatic build system regularly in order to detect problems immediately.
- GUI for software installation, ...

3.3 How to create a software package for mediAkit

This section describes how to create a software package for mediAkit. We decided to use the RPM format and package management system (see Section B.2).

RPM supports the distribution of source RPM packages. However, note that RPM (as of version 4.1) doesn't yet support a way to specify source package dependencies, only requirements for installed binary packages (`Build-Deps:`).

The following procedure is employed to create a software package: The software package is supposed to be available as source code, should be implemented for the Linux platform and tested by the implementation team. In order to prepare software packages for the distribution the following steps are necessary:

- Prepare the software packages to meet the requirements (e.g., use of autotools, make mechanism, etc.).
- Place the software at the correct place in the source tree.
- Create a spec file (see Section J.1.3).
- Create a Source RPM package with *rpmbuild* (see Section J.2)

- Build machine specific Binary RPM packages from the Source RPM package with *rpmbuild*

We provide a packaging example in Section J.

3.4 Maintenance of the distribution

New versions of software are developed continuously, including new features, corrected bugs, etc. Therefore, the distribution must follow this development, and should be held up to date. For maintenance, automated routines must be employed to follow up new developments, update existing software packages, etc. Additionally, an infrastructure for reporting and forwarding problem reports, and a exchange of messages, etc. is necessary.

For the project we set up a server using Postnuke (See Section G). This solution includes a portal with the possibility to post messages. The mediAkit web server has two purposes: offer the distribution and its files, and offer a forum for problem reports, support, etc. The portal is implemented at the address `http://mediakit.nr.no`.

A system for receiving and forwarding problem reports has not yet been installed. It remains yet unclear whether we can implement a general procedure for forwarding bugs for software that is implemented by third parties.

For the automatic update of software, some scripts were implemented, that check the availability of new versions of a software package. However, this makes use of conventions that are not used by all implementors. Normally, a general link to a software package points to the newest version, and when the md5 signature[10] changes, a new version has been published.

[10] We maintain a database with md5 signatures, and the script checks regularly whether this signature has changed.

4 Conclusions and Future Work

At the end of the first term of the project the server for mediAkit is about to be released. The infrastructure for the web-server is already operative, including the discussion groups.

The install system has been tested on several Red Hat installations. Some selected applications can be installed successfully. However, the installer is yet unfinished, and some more implementation and testing has to be done in order to roll out the distribution.

Before the distribution is released we need to do more research on licensing issues. While the distributed software comes with the licenses the original authors provide, the distribution as such also needs a license. Possibly we have to restrict the availability of the distribution due to legal issues. Alternatively some software packages must be excluded from the distribution, thus reducing the functionality the distribution can offer.

The work with the **mediAkit** distribution continues in year two. The main goal is to get the distribution operative, followed by extensions and a even more user-friendly interface.

Part II
Tools for building and distribution of software

This part of the report contains a short description of relevant software for the various tasks that have to be solved within the setup project. We intended to keep the descriptions short. However, in order to support the internal discussion regarding some problem domains and its solutions a more profound description was emerging for some areas.

A Build systems

A build system supports how to build executable code out of source code. For our purposes the discussion is limited to tools like make, and how to produce portable makefiles.

A.1 make (GNU make)

Package:	make	**Version:**	3.80	**Date:**	December 16, 2002
Exports:				**Type:**	Control file generation
Uses:				**License:**	GPL
Url:	`http://www.gnu.org/software/make/make.html`				
Author(s):	Richard Stallman and Roland McGrath (FSF)				

GNU make is a tool which controls the generation of executables and other non-source files of a program from the program's source files. A make utility is normally provided along with C compilers. The concept of *make* goes back to S. Feldman in 1979 [6].

make gets its knowledge of how to build your program from a so-called `makefile`, which is a text file containing a set of named *targets* that can be built. For each target a list of other targets or files that this target depends upon can be listed, followed by a set of command lines that must be executed in order to build this target. Often, a target is the name of a library or a file. Some targets are very often used by convention, e.g., "all" or "install".

When building a target, make checks recursively through the list of dependent targets, and performs build actions when necessary, e.g., a newer version of a target is available. In case a library target that a program depends upon has changed, the program has to be rebuilt in order to work with the new library version. If a source file that the library depends upon has changed too, the library has to be rebuilt first.

The make system provides functionality for using macros and general rules. If, for instance, the name of the source and output directories are defined using macros, these can easily be redefined without having to rewrite all the target-commands in the makefile.

For more information on make we refer to the book on GNU Make [13], the documentation in Texinfo [3] format and the manual page.

A.2 GNU Autotools

FSF has created a set of tools to support the development and distribution of software across Unix systems. The list of tools includes **autoconf**, **autoheader**, **automake**, **aclocal** and **libtool**.

The use of these tools is a two phase process. In the first phase a software package is prepared for distribution, while in the second phase this package is configured for use on the target system, built and installed. The complexity of the first phase is to ensure maximum flexibility and simplicity for performing the second. The purpose of the first phase is to create (at least) two files: `Makefile.in` and `configure`. The file *configure* is a shell script program that checks for existence, placement and features of tools, libraries and installed hardware and software that the build process depends on. Thereafter, it configures a set of appropriate `makefiles` from the templates in `Makefile.in`.

A.2.1 autoconf

Package:	autoconf	**Version:**	2.57	**Date:**	December 16, 2002
Exports:				**Type:**	Configure source packages
Uses:	m4, Perl			**License:**	LGPL
Url:	`http://www.gnu.org/software/autoconf/autoconf.html`				
Author(s):	Free Software Foundation (FSF)				

autoconf is an extensible package of m4 macros that produce shell scripts to automatically configure software source code packages. These scripts can adapt the packages to many kinds of UNIX-like systems without manual user intervention. *autoconf* creates a configuration script for a package from a template file that lists the operating system features that the package can use, in the form of m4 macro calls.

Producing configuration scripts using *autoconf* requires GNU *m4* (version 1.4 or later). The configuration scripts produced by *autoconf* are self-contained. Therefore, for the configure-process on the target machine neither *autoconf* nor GNU m4 need to be installed. Some optional utilities of *autoconf* use *perl*.

A.2.2 autoscan

The **autoscan** program helps to create and/or maintain a `configure.in` file for a software package. *autoscan* examines source files in the directory tree rooted at a directory given as a command line argument, or the current directory if none is given. It searches the source files for common portability problems and creates a file `configure.scan` which is a preliminary `configure.in` for that package, and checks a possibly existing `configure.in` for completeness.

A.2.3 aclocal

aclocal is a utility that can provide the file `aclocal.m4` by copying the definition of m4-macros used in `configure.in` from m4-files found on the system. This file is optional input to *autoheader* to help configuring the software for compilation on different platform.

A.2.4 autoheader

autoheader is a tool for automatically generating the `config.h.in` file, compliant with the GNU Coding Standards. This file can be used as input to the *configure* script in order to provide cross-platform compatibility.

A.2.5 automake

Package:	automake	**Version:**	1.7.2	**Date:**	December 16, 2002
Exports:				**Type:**	Generate 'Makefile.in'
Uses:	autoconf			**License:**	GPL
Url:	`http://www.gnu.org/software/automake/automake.html`				
Author(s):	Free Software Foundation (FSF)				

automake is a tool for automatically generating `Makefile.in` files compliant with the GNU Coding Standards. This is the basis for the `Makefile` that is output by the *configure* shell script program.

A.2.6 libtool

Package:	libtool	**Version:**	1.4.3	**Date:**	December 16, 2002
Exports:				**Type:**	Library support script
Uses:				**License:**	GPL
Url:	`http://www.gnu.org/software/libtool/libtool.html`				
Author(s):	Gordon Matzigkeit, FSF				

libtool is a generic library support script. Libtool hides the complexity of using shared libraries behind a consistent, and portable interface.

```
source files --> [autoscan*] --> [configure.scan] --> configure.in

configure.in  ------.
[acinclude.m4] -----.
                    +--> [aclocal*]  -->  [aclocal.m4]
.m4-files found ----'

configure.in --.   .------> autoconf* -----> configure
               +---+
[aclocal.m4] --+   `---.
[acsite.m4] ---'       |
                       +--> [autoheader*] -> [config.h.in]
[acconfig.h] ----.     |
                 +-----'
[config.h.top] --+
[config.h.bot] --'

[Makefile.am's]--->[automake*]-+
                               |
                               v
Makefile.in -------------------------------> Makefile.in
```

Figure 3: Using autotools.

A.2.7 Using autotools

Preparing a software package with autotools. The use of autotools is quite complex, and involves a number of files and programs. Figure 3 presents a diagram from the autotools manual showing the information flow and program calls when using the autotools. For a more profound description, we refer to the documentation that follows the autotools software package.

Many files may be involved in preparing a software package for distribution. Most of these are optional. Only the files `configure.in` and `Makefile.in` and the use of *autoconf* are mandatory. Using the other tools and files may however ease the maintainability of the software package.

The first step is to run *autoscan* on the source files in order to create the file `configure.in`. The program *aclocal* searches the system for macros in .m4-files. Definitions for macros used in `configure.in` are copied to the file `aclocal.m4`.

The two files `configure.in` and `aclocal.m4` are used as input to the command *autoconf*, which creates a shell-script program called *configure*, which is used later to configure the software distribution at the remote site.

The program *autoheader* is used to create the file `config.h`, which contains declarations and compiler switches used in the program. These definitions are included in the program at compile time, which makes it easier to maintain flexibility and compilation across platforms.[11]

A template for the `makefile` must be provided in the file `Makefile.in`. In the installation phase the program *configure* will create the appropriate makefile. The program *automake* (see Section A.2.5) can be used to maintain a `Makefile.in`.

Configuration and installation of software with autotools. Files and programs used during configuration of a software package are shown in Figure 4. As mentioned above, the shell-script *configure* is generated by the *autoconf* program. The process of configuring, building and installing the program starts by executing this shell-script. *configure* will find the location of directories and system libraries, check for compiler features etc., and save the results in several files. Most of the work is performed by an accompanying script called config.status. As a result of this process the appropriate files (e.g., `Makefile`, `config.h`, etc.) are created based on the template files with the extension `.in`.

[11] The concept of header files comes from the programming language C, but can be used for other purposes. For that the preprocessor *cpp* can be used.

```
                       .------------> config.cache
configure* -----------+-------------> config.log
                       |
[config.h.in] -.       v           .-> [config.h] -.
               +--> config.status* -+               +--> make*
Makefile.in ---'                   `-> Makefile ---'
```

Figure 4: Configuring software.

If the configuring step was successful *make* is run in order to build the software. Finally, *make install* is called to perform the steps necessary to install the newly built parts of the software package.

A.3 GAR

Package:	GAR **Version:** CVS	**Date:**	
Exports:		**Type:**	Build/install source code
Uses:	GNU make	**License:**	BSD
Url:	`http://gar.lnx-bbc.org/`		
Author(s):	Nick Moffitt		

GAR is a library of *make* targets that can be configured by setting certain special variables or overridden file-by-file. Each package in the GAR system is contained in a directory inside the directory representing its category. Inside the package directory, there are 2 files: `Makefile` and `checksums`

A.4 Ant

Package:	Ant **Version:** 1.5.1	**Date:**	Oct 3, 2002
Exports:		**Type:**	Build/install source code
Uses:	Java Runtime or SDK	**License:**	Apache
Url:	`http://jakarta.apache.org/ant/index.html`		
Author(s):	?		

Ant is a platform-independent build tool created by one of the Apache web server developers. The build-specifications are in a platform-independent XML text file. *Ant* is based on the abstraction of "tasks" rather than shell commands, since shell commands are inherently bound to the platform (and sometimes even installation) that they were created on. Dependencies and most other features from *make* are supported. The concept of "task" is more general than the concept of "target" in *make*. Generally, *Ant* is recognised as more versatile and flexible than *make*. *Ant* is a Java program, and needs a Java runtime environment and some Java jar-libraries (for xml processing etc.) in order to run.

B Package Management systems

B.1 Dpkg

Package:	Dpkg	**Version:**	1.9.21 (stable)	**Date:**	
Exports:				**Type:**	Package Management System
Uses:	Perl			**License:**	GPL
Url:	`http://packages.debian.org/stable/base/dpkg.html`				
Author(s):	Debian developers				

Dpkg is the package maintenance system for Debian GNU/Linux. The package format is defined in the Debian Policy manual.

B.2 RPM

Package:	RPM	**Version:**	4.1 (stable)	**Date:**	
Exports:				**Type:**	Package Management System
Uses:	libpopt			**License:**	GPL
Url:	`http://www.rpm.org/`				
Author(s):	Red Hat, Inc.				

RPM is a command-line package management system. It is capable of installing, uninstalling, verifying, querying, and updating software packages. Each software package consists of an archive of files along with information about the package, e.g., version, textual description, etc. The RPM-4.1 format support cryptographic verification of source package builders, using OpenPGP V3 packets. Currently, RPM is being used in RedHat, Mandrake and other distributions. There are also various package managers that support RPM.

Internally, RPM uses the `cpio` format, which is used to archive files in Unix systems (see `man cpio` on a Unix system).

The RPM Package Manager is capable of archiving pre-compiled binaries, as well as compressed source code with binary and text patch files, in a single, common file format archive (cpio).

RPM can be used to build, install, query, verify, update and erase individual software packages. RPM works with a database of installed software and files, which maintains dependencies and requirements between installed and upgraded software packages. The RPM database supports queries for files and package details.

RPM supports cryptographic Al validation of the source of packages, by the usage of OpenPGP compatible digital signatures in packages supported by tools, such as the GNU Privacy Guard (GPG).

See Section 16 in the Linux Standard Base Common Specification [15] for the definition of the RPM File format. The command line tool *rpm* is used in to manage existing packages, while *rpmbuild* is used to create them. See the manual pages for *rpm(8)* and *rpmbuild(8)* for more details.

Working with RPM

There are three different forms of RPM-packages, binary RPMs, source RPMs (SRPMs) and development RPMs. The naming convention of RPM files follows the following rule:

```
<pkg-name>-<version>-<release>.<type>.rpm
```

where `<pkg-name>` is the unique name of the software package, `<version>` is a 3-digit major-minor-micro version number, `<release>` is the release number of this version and `<type>` is an identifier for the package type. Valid RPM package types include

- `src`, `nosrc`
- `<architecture>` (`noarch`, `i386`, . . . , `i686`, . . .)

The entire process of creating and building RPM packages is bound to a strict directory structure:

- `BUILD`
- `RPMS`
- `SOURCES`
- `SPECS`
- `SRPMS`

The locations of these directories can be overridden by passing `--define` options to *rpmbuild* tool. See Section J for an example. The content, building and installation of the package is controlled by a text file called the `spec`-file, usually placed in the SPECS directory. The file describes

- Library name, version and vendor
- Dependencies on other packages
- Files (executables and libraries) contributed to the system by installation of the package.
- Dependencies for building (normally development packages with header files etc.)
- Archive(s) that contain the source files, patches etc. needed to build the package.
- The set of commands that must be executed in order to perform preparation, to perform building, installation, cleanup etc.

The program *rpmbuild* can be used to build a Source RPM (SRPM) from source file archives in the SOURCES directory and a spec-file. It can also build binary RPMs from an SRPMs. The SRPM contains all the necessary source files and the spec file. A RPM contains compiled and linked versions of all the libraries and programs in the package, as well as the spec file.

It is widespread, but not mandatory, to use *autoconf / autotools* and *make* for controlling the build process. RPM is built on top of a database. Queries are sent to the database in order to find out whether dependent packages are installed.

B.3 ESP Package Manager (EPM)

Package:	EPM	**Version:**	3.5.1	**Date:**	December 18, 2002
Exports:				**Type:**	Package Management System
Uses:				**License:**	GPL
Url:	`http://www.easysw.com/epm/`				
Author(s):	Easy Software Products				

EPM is a free UNIX software/file packaging program that generates distribution archives from a list of files. EPM generates portable script-based distribution packages complete with installation and removal scripts, and vendor distributions in AIX, BSD, Compaq Tru64, Debian, HP-UX, IRIX, MacOS X, Red Hat, and Solaris formats.

C Install and Update systems

C.1 Loki Setup Graphic Installer

Package:	Loki Setup Graphic Installer	**Version:**	1.5.8	**Date:**	
Exports:				**Type:**	Graphical installer
Uses:	libxml 1.4.0			**License:**	GPL
Url:	`http://www.lokigames.com/development/setup.php3`				
Author(s):	Sam Lantinga				

The **Loki Setup Graphic Installer** allows the developer to create a simple, easy-to-use, standardised install routine. It requires Loki SetupDB which can be installed from the same site. The installer requires libxml 1.4.0 to parse XML configured description of the files to be installed, and libglade 0.7 to dynamically load the GTK+ user interface definitions.

C.1.1 Loki Update Tool

Package:	Loki Update Tool	**Version:**	1.0.13	**Date:**	December 20, 2002
Exports:				**Type:**	Software Update Tool
Uses:				**License:**	GPL
Url:	`http://www.lokigames.com/development/loki_update.php3`				
Author(s):	Loki Software, Inc.				

The **Loki Update Tool** is written by Loki Software, Inc., designed to be used in conjunction with their setup and patch tools to easily update products.

This end-user tool allows to quickly and easily update Loki software that is installed already. It features easy-to-use mirror support, the ability to resume broken connections, support for HTTP and FTP proxies, and automatic update verification to make sure that the correct download is performed.

C.1.2 Loki Patch Tool

Package:	Loki Patch Tools	**Version:**	1.0.1	**Date:**	December 20, 2002
Exports:				**Type:**	Binary Patch Tool
Uses:	Loki SetupDB, xdelta			**License:**	GPL
Url:	`http://www.lokigames.com/development/loki_patch.php3`				
Author(s):	Loki Software, Inc.				

The **Loki Patch Tool** is a set of tools written by Loki Software, Inc., designed to create a set of binary patches that can be easily used in conjunction with the Loki Update Tool.

C.2 Red Hat Network Up2date

Package:	Red Hat Network Update Agent	**Version:**	3.0.7	**Date:**	
Exports:				**Type:**	Software maintenance
Uses:				**License:**	GPL
Url:	`http://rhn.redhat.com`				
Author(s):	Red Hat, Inc.				

The Red Hat Update Agent features a GNOME interface for installation and updates of RPM packages from Red Hat and specified software channels.

C.2.1 up2date

Package:	up2date	**Version:**	2.7.61	**Date:**	
Exports:				**Type:**	Software maintenance
Uses:				**License:**	GPL
Url:	`http://www.redhat.com/docs/manuals/RHNetwork/ref-guide/up2date.html`				
Author(s):	Red Hat, Inc.				

up2date uses the Red Hat Network, and is free for the first computer, but if we want to distribute our own packages we need the server. Red Hat has not released this server, but freely available servers such as Current and nrh-up2date exists.

C.2.2 Current

Package:	Current	**Version:**	1.4.1	**Date:**	
Exports:				**Type:**	Software maintenance
Uses:				**License:**	GPL
Url:	`http://www.biology.duke.edu/computer/unix/current/`				
Author(s):	Hunter Matthews				

Current is a server for Red Hat's up2date, which is a suite of tools (sometimes called the Red Hat Network) for keeping a Red Hat Linux system "up to date" with new security/bugfix RPM's. It also allows you to add completely new rpm's to clients over the network. The tools do a thorough job of handling bootloader configurations, new kernels, packages dependencies and conflicts.

Unfortunately, Red Hat have not released their server, which does not fully address the needs of some users/departments. Thus, Current was born. Current runs under apache. The locations of rpm and srpm files, including so-called channels are defined in `/etc/current/current.conf`. The up2date client gets the address of the used server in `/etc/sysconfig/rhn/up2date`.

C.2.3 NRH-up2date

Package:	NRH-up2date	**Version:**	1.1	**Date:**	December 18, 2002
Exports:				**Type:**	Software maintenance
Uses:	Perl, Frontier::RPC2, Digest::MD5			**License:**	GPL
Url:	`http://www.nrh-up2date.org/`				
Author(s):	Alex Kramarov				

NRH-up2date is a collection of utilities and instructions to use with Red Hat's up2date client independent of the Red Hat Network.

C.3 APT

Package:	APT	**Version:**	0.5.4	**Date:**	December 18, 2002
Exports:				**Type:**	tool
Uses:				**License:**	GPL
Url:	`http://packages.debian.org/stable/base/apt.html`				
Author(s):	APT Development team				

APT (Advanced Packaging Tool) is a set of tools to support downloading, installation, and handling of package dependencies. Originally, APT was developed for the Debian distribution, but later ported to other distributions. **APT** is the Debian package management facility. In addition to standard dependency checking and installation functions, the APT system is also able to resolve all dependencies recursively and download necessary packages. The command

```
prompt> apt-get install some-package
```

will

- search server locations listed in the file `/etc/apt/sources.list` for the package called *some-package.*
- download the package.

- check the dependencies of this package.
- recursively find, download and install packages that *some-package* depend on that are not yet installed in the proper version.
- install *some-package*.

APT can load packages from various source URLs (including ftp: and http:) as long as they follow a specific directory and content structure. APT searches for packages in locations as specified by the file `/etc/apt/sources.list`.

C.3.1 APT-RPM

Package:	APT-RPM	**Version:**	0.5.4cnc7	**Date:**	
Exports:				**Type:**	Software maintenance
Uses:	rpm			**License:**	GPL
Url:	`http://freshmeat.net/articles/view/192/`				
Author(s):	Conectiva Inc., Alfredo Kojima				

APT-RPM is a port of Debian's apt tools to a RPM based distribution (Red Hat, Conectiva, SuSE etc.). apt-get is an advanced package management utility front-end which allows the user to easily perform package installation, upgrading and removal. Dependencies are automatically handled, and dependent packages are automatically installed if needed.

C.4 Ximian Red Carpet

Package:	Ximian Red Carpet Client	**Version:**	2.1.9.1	**Date:**	
Exports:				**Type:**	Software maintenance
Uses:				**License:**	GPL
Url:	`http://www.ximian.com/products/redcarpet/`				
Author(s):	Ximian, Inc.				

Ximian Red Carpet is a software management solution for Linux desktops from Ximian. It provide a software channel with automatic dependency and conflict resolution that makes it easy to install, update and maintain software over the Internet from Ximian, leading Linux distribution providers and a variety of independent software vendors. Specifically designed for organisations running mixed computing environments, Red Carpet handles a broad range of GNU/Linux distributions, including Red Hat, SuSE, Mandrake, Debian and more.

Ximian released version 1.0 of the Red Carpet Daemon (rcd) and Red Carpet command-line client (rc). See `http://developer.ximian.com/projects/rcd/` for details.

C.5 InstallBase

Package:	InstallBase	**Version:**	1.0a2	**Date:**	December 20, 2002
Exports:				**Type:**	Graphical installer
Uses:				**License:**	BSD license
Url:	`http://installbase.sourceforge.net`				
Author(s):	Damon Courtney				

InstallBase InstallBase MPI is a multi-platform GUI installer designed to be completely cross-platform and function on Windows, Macintosh and most all versions of UNIX.

D Hardware Detection systems

D.1 harddrake

Package:	harddrake **Version:** 0.9	**Date:**	2001
Exports:		**Type:**	Hardware Detection
Uses:	GTK+ 1.2, detect	**License:**	GPL
Url:	`http://www.mandrakelinux.com/harddrake/`		
Author(s):	Mandrake, Inc		

HardDrake is a hardware configuration tool written in GTK+ for the Mandrake distribution.

D.2 detect

Package:	detect **Version:** 0.9.72	**Date:**	2001
Exports:		**Type:**	Hardware Detection
Uses:		**License:**	GPL
Url:	`http://www.mandrakelinux.com/harddrake/`		
Author(s):	Mandrake, Inc		

Detect is a library for automatic hardware detection. The following hardware can be detected: CPU, memory, disks, web cams, joysticks, partitions, Ethernet cards, floppy drives, modems, mice, SCSI, sound cards, and video cards.

D.3 kudzu

Package:	kudzu **Version:** 0.99.69 (CVS)	**Date:**	October 18, 2002
Exports:		**Type:**	Hardware Detection
Uses:	pciutils python newt	**License:**	GPL
Url:	`http://rhlinux.redhat.com/kudzu/`		
Author(s):	Red Hat, Inc.		

kudzu is the Red Hat Linux hardware probing library, and the associated configuration program. The probing library is used by various system utilities, such as anaconda, Xconfigurator, and hwbrowser. The tool runs at system boot time to determine what hardware has been added or removed from the system.

D.4 discover

Package:	discover **Version:** 2.0.1	**Date:**	October 24, 2002
Exports:		**Type:**	Hardware Detection
Uses:	libdiscover	**License:**	GPL
Url:	`http://archive.progeny.com/progeny/discover/`		
Author(s):	Progeny, Inc.		

Discover is a cross-platform hardware detection system that uses system-dependent modules (selected at build time) for detecting the hardware on a system. It also provides system-independent interfaces for querying XML data sources about this hardware.

D.5 Kernel Autoconfigure

Package:	Kernel Autoconfigure **Version:** 1.9	**Date:**	December 16, 2002
Exports:		**Type:**	Hardware Detection
Uses:	cml2	**License:**	GPL
Url:	`http://sourceforge.net/projects/kautoconfigure/`		
Author(s):	Giacomo Catenazzi		

Kernel Autoconfigure is a tool for configuring the Linux kernel, detecting hardware and used software protocols.

D.6 libhardware

Package:	libhardware	**Version:**	0.7.1	**Date:**	November 19, 2002
Exports:				**Type:**	Hardware Detection
Uses:				**License:**	GPL
Url:	`http://www.badpenguin.org/`				
Author(s):	Antonio Gallo				

libhardware is a static library used in many utilities that are part of the AGX's GNU/Linux distribution "Bad Penguin". It provides functions to access information about the hardware of the system.

E Bug Tracking systems

E.1 Bugzilla

Package:	Bugzilla	**Version:**	2.17.1	**Date:**	December 20, 2002
Exports:				**Type:**	Bug Tracking system
Uses:	Apache			**License:**	MPL
Url:	`http://www.mozilla.org/projects/bugzilla/`				
Author(s):	Dave Miller				

Bugzilla is a web-oriented database for reporting problems (bugs) in software. It lets people report bugs and assigns these bugs to the appropriate developers. Developers can use Bugzilla to keep a to-do list as well as to prioritise, schedule and track dependencies.

E.2 PHP bugtracker

Package:	PHP bugtracker	**Version:**	0.9.0	**Date:**	December 20, 2002
Exports:				**Type:**	Bug Tracking system
Uses:	Apache, PHP			**License:**	GPL
Url:	`http://phpbt.sourceforge.net/`				
Author(s):	Benjamin Curts				

PHP bugtracker is designed as a replacement for Bugzilla. This project grew out of the frustrations experienced in installing and using Bugzilla. Design goals for PHP bugtracker are simplicity in use and installation, use of templates to achieve presentation independence, and the use of a database abstraction layer for independence from a data base.

E.3 Bug-buddy

Package:	Bug-buddy	**Version:**	2.2.0	**Date:**	December 20, 2002
Exports:				**Type:**	Bug Tracking System
Uses:	GNOME, Bugzilla			**License:**	GPL
Url:	`ftp://ftp.gnome.org/pub/GNOME/stable/sources/bug-buddy/`				
Author(s):	Jacob Berkman				

Bug-buddy is an interface tool for Bugzilla. Features include the ability to obtain a stack trace from a core file or crashed application and to determine versions of packages installed on a system.

E.4 GNATS

Package:	GNATS	**Version:**	3.999.2 (stable)	**Date:**	December 20, 2002
Exports:				**Type:**	Bug Tracking System
Uses:				**License:**	GPL
Url:	`http://sources.redhat.com/gnats/`				
Author(s):	GNATS development Team				

GNATS contains a database and tools for administering problem reports. The data base is file based. Most of the administration tools are based on formatted messages. Gnats includes administration tools, a server program and various tools to access the information by the end user, e.g., interfaces towards email, emacs, and web.

F Software Image systems

A software image system is capable of building (bootable) install images from a tree of software packages. For our purposes the discussion is limited to building and recording a image of the software distribution packages onto CDROM and DVD media.

F.1 BICK

Package:	BICK	**Version:**	0.8.0	**Date:**	December 16, 2002
Exports:				**Type:**	ISO image generation
Uses:				**License:**	GPL
Url:	`http://phil.ipal.org/freeware/bick/`				
Author(s):	Philip Howard				

BICK builds a bootable Linux ISO image from a file tree that represents the run time file tree loaded into tmpfs with options to leave /opt and /usr mounted on the CDROM to save space. For systems with enough RAM, up to as much file data as will fit on the CDROM can be loaded; initial ramdisk is not used for the runtime files.

F.2 Progeny Graphical Installer

Package:	Progeny Graphical Installer	**Version:**	1.0.1	**Date:**	December 16, 2002
Exports:				**Type:**	ISO image generation
Uses:				**License:**	GPL
Url:	`http://archive.progeny.com/progeny/pgi/`				
Author(s):	Progeny				

Progeny Graphical Installer (PGI) is a multi-architecture system for creating graphical installers for installing a bootable Debian GNU/Linux operating system onto a new computer system.

The PGI package enables the user to create ISO images containing a bootable installer that guides the user through the steps of installation. The installer supports text and graphical installation modes. ISO images can be generated with complete or partial package archives, or with the installer only (useful for network-only installs, which PGI supports).

F.3 Anaconda

Package:	Anaconda	**Version:**	CVS	**Date:**	December 16, 2002
Exports:				**Type:**	ISO image generation
Uses:	python			**License:**	GPL
Url:	`http://rhlinux.redhat.com/anaconda/`				
Author(s):	Red Hat, Inc.				

Anaconda is the Red Hat Linux Installation Program. It supports installation media tests, and several installation methods.

G Web portal and content management systems

G.1 Apache

Package:	Apache	**Version:**	1.3.27	**Date:**	December 18, 2002
Exports:				**Type:**	HTTP/1.1 Server
Uses:				**License:**	Apache Software License
Url:	`http://httpd.apache.org/`				
Author(s):	Apache Software Foundation				

The **Apache** HTTP Server Project is an effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows NT.

G.2 Plone

Package:	Plone	**Version:**	1.0 RC1	**Date:**	December 20, 2002
Exports:				**Type:**	Content Management
Uses:	Apache, Zope			**License:**	BSD
Url:	`http://www.plone.org/`				
Author(s):	Alexander Limi, Alan Runyan, Vidar Andersen				

Plone is built on top of the open source application server Zope (see `http://www.zope.org/`) and the accompanying Content Management Framework. Plone can be used as an intranet server, as a document publishing system and as a groupware tool for collaboration between separately located entities. Plone currently supports approximately 20 languages.

G.3 Postnuke

Package:	Postnuke	**Version:**	0.7.2.3	**Date:**	December 20, 2002
Exports:				**Type:**	Content Management
Uses:	Apache, PHP			**License:**	GPL
Url:	`http://www.postnuke.com/`				
Author(s):	PostNuke development team				

Postnuke was originally a fork of PHPNuke by Francisco Burzi. Many of the concepts of PHP-Nuke survive even though the code is rewritten. Postnuke is also influenced by earlier products such as *slash* and *thatware.*

Many open source Content Management solutions exist, and choosing the right one is often a subjective decision. We chose postnuke because it fulfils our needs, is easily to configure, easy to install, easy to use and because we had previous positive experiences with it.

Part III

The software catalogue

This part of the report contains an overview over various multimedia program packages for Linux. The list is not exhaustive. We have tried to list the projects that are most useful for end users and most likely to be included in our distribution.

H Open Source Multimedia Software

Quite a lot of interesting software exist for Linux systems. Until recently, Linux distributions had only modest amount of multimedia software included. This may be influenced by the facts that much of this software is officially pre-release, the software development projects are moving quickly, that the set of dependencies between the various projects is quite complex and that some of the software is hardware-dependent.

This part of the report contains an overview of relevant multimedia software. We are aware that there are many other software packages out there, that would fit into the categories mentioned in our software catalogue. Open source software on digital TV is part of another project within the Channel S program, and therefore not listed here. Readers interested in open source projects on digital TV are referred to the project report of the digital TV project [7].

H.1 Frameworks and multitools

H.1.1 GStreamer

Package:	GStreamer **Version:** 0.5.0	**Date:**	December 16, 2002
Exports:	most available formats	**Type:**	framework
Uses:	Gnome widgets, external tools, e.g., Icecast	**License:**	GPL
Url:	`http://gstreamer.net/`		
Author(s):	Erik Walthinsen, (many others)		

GStreamer is a framework similar to Microsoft DirectShow (aka. ActiveMovie). Can handle (input and output) many file types. Many plugins exist, including for Icecast. However, there is no support for Win32 plugins. A command line utility, graphical graph editor and multimedia SDK are included in the same package.

H.2 Capturing

H.2.1 Coriander

Package:	Coriander **Version:** 0.27.1	**Date:**	November 26, 2002
Exports:		**Type:**	library
Uses:	GTK+ >= 1.2, libraw1394 , libdc1394, 1394-compatible kernel	**License:**	GPL
Url:	`http://www.tele.ucl.ac.be/PEOPLE/DOUXCHAMPS/ieee1394/coriander/`		
Author(s):	Damien Douxchamps		

Coriander is a GUI for Linux to control a digital camera through the IEEE1394 bus (aka. FireWire, or iLink). The digital camera must comply with the IIDC v1.04 (or later) Digital Camera Specifications, published by the 1394 Trade Association.[12] We didn't test coriander, due to compilation problems with *libdc1394*.

[12] Note that all cameras in which can be inserted a video tape (camcorders, . . .) are not compliant with the IIDC specs. Those cameras record compressed DV video on a tape, while the IIDC protocol is specified for uncompressed, on-the-fly video flow.

H.2.2 libgphoto2

Package:	libgphoto2 **Version:** 2.1.1	**Date:**	November 2002
Exports:	JFIF, TIFF, EXIF	**Type:**	Digital camera library
Uses:	libusb	**License:**	LGPL
Url:	`http://gphoto.sourceforge.net/`		
Author(s):	Scott Fritzinger, Lutz Müller, et.al.		

The **libgphoto2** library is used by applications to access and control various digital still camera models. A command-line frontend follows with the library. Several graphical front ends for the library are available.

H.2.3 gtkam

Package:	gtkam **Version:** 0.1.10	**Date:**	November 2002
Exports:	JFIF, TIFF, EXIF	**Type:**	Digital photo application
Uses:	libgphoto2 >= 2.1.1, GTK+ >= 2.0	**License:**	GPL
Url:	`http://gphoto.sourceforge.net/`		
Author(s):	Scott Fritzinger, Lutz Müller		

gtkam provides a GTK+ front-end to libgphoto2, a common library to access and control digital still cameras.

H.2.4 digiKam

Package:	digiKam **Version:** 0.5.1	**Date:**	December 20, 2002
Exports:		**Type:**	Digital photo application
Uses:	libgphoto2, KDE	**License:**	GPL
Url:	`http://digikam.sourceforge.net/`		
Author(s):	Renchi Raju		

digiKam is a KDE application which acts as a frontend to gphoto2 for communicating with a digital camera and download pictures from it.

H.2.5 SLab

Package:	SLab **Version:** 4.09b	**Date:**	December 20, 2002
Exports:		**Type:**	Multitrack HD recording
Uses:		**License:**	Partly GPL
Url:	`http://slabexchange.org/`		
Author(s):	Nick Copeland, Toby Shepard		

SLab is a digital audio recording software suite for UNIX platforms, which is currently supported for Linux, and FreeBSD.

SLab can record from up to eight stereo sound cards to files. The audio data can then simultaneously be read back from disk as separate tracks, and mixed down back to the audio devices. SLab supports up to 64 tracks.[13] Note, SLab is not a real time mixer, so its use is not indended for live work.

Not all parts of SLab are in open source. Most of the APIs are available as source, to access the database, build effects, extend some of the mixing algorithms; additionally all of the audio driver interface code is in the GPL.

[13] However, the CPU power might restrict that; a P133 will max out at about 8 tracks, a P450 has been tested to 16 tracks (even with the more intensive mix algorithms) with capacity to spare.

H.2.6 Ardour

Package:	Ardour	**Version:**	0.522.4	**Date:**	December 2002
Exports:	WAV			**Type:**	Multichannel HD recorder
Uses:	ALSA, JACK			**License:**	GPL
Url:	`http://ardour.sourceforge.net/`				
Author(s):	Paul Davis				

Ardour is a multichannel hard disk recorder (HDR) and digital audio workstation (DAW). It is capable of simultaneous recording 24 or more channels of 32 bit audio at 48kHz. Ardour is intended to function as a "professional" HDR system, replacing dedicated hardware solutions such as the Mackie HDR, the Tascam 2424 and more traditional tape systems like the Alesis ADAT series. It is also intended to provide the same or better functionality as proprietary software DAWs such as ProTools, Samplitude, Logic Audio, Nuendo and Cubase VST. It supports MIDI Machine Control, and so can be controlled from any MMC controller, such as the Mackie Digital 8 Bus mixer and many other modern digital mixers.

H.2.7 XMMS Recorder

Package:	XMMS Recorder	**Version:**	0.3.0	**Date:**	December 20, 2002
Exports:	WAV			**Type:**	plugin
Uses:	XMMS			**License:**	GPL
Url:	`http://xmms-recorder.sourceforge.net/`				
Author(s):	Tamás Skopkó, Mike Tyson				

XMMS recoder is an XMMS input plugin that records from your soundcard's input and sends audio to the selected output. It can be used for recording audio to disk, or running a visualisation on an input stream. XMMS Recorder works with OSS, eSounD and ALSA.

H.2.8 LiveIce, LiveIce XMMS

Package:	LiveIce / LiveIce XMMS	**Version:**	1.0	**Date:**	2001
Exports:				**Type:**	library
Uses:	GTK, GDK, libraw1394, libdc1394, 1394-compatible kernel			**License:**	GPL
Url:	`http://star.arm.ac.uk/~spm/software/liveice.html`				
Author(s):	Damien Douxchamps				

LiveIce was developed after the appearance of IceCast to provide the Real Time streaming functionality which makes it a viable solution for broadcasting. **LiveIce XMMS** is an effects plugin for xmms which sends streams to IceCast and provides a limited set of functions similar to those provided by liveice but without half the hassle. LiveIce XMMS lacks many of the features of the full version of LiveIce but provides the main functions needed to stream from xmms.

H.2.9 Jack, Jackit, jackd, libjack

Package:	Jack audio connection kit	**Version:**	0.34	**Date:**	2001
Exports:	sound, via low latency callbacks			**Type:**	library, daemon
Uses:	ALSA, RealTime kernel patch recommended			**License:**	GPL
Url:	`http://jackit.sourceforge.net`				
Author(s):	Paul Davis, Richard Günter et al.				

JACK is a low-latency audio server, written primarily for the GNU/Linux operating system. It can connect a number of different applications to an audio device, as well as allowing them to share audio between themselves. Its clients can run in their own processes (i.e. as normal applications), or can they can run within the JACK server (i.e. as a "plugin"). The system focuses on two key areas: synchronous execution of all clients, and low latency operation.

H.2.10 gPhoto

Package:	gPhoto	**Version:**	2.1.0	**Date:**	November 2002
Exports:				**Type:**	JFIF, EXIF, TIFF
Uses:				**License:**	GPL, LGPL
Url:	`http://gphoto.sourceforge.net`				
Author(s):	Christophe Barbe, Hubert Figuiere, Scott Fritzinger, Lutz Müller, Hans Ulrich Niedermann				

gPhoto is a program and library framework that lets users download pictures from their digital cameras. There are currently 253+ supported digital cameras across several platforms.

H.2.11 SANE

Package:	SANE	**Version:**	1.0.9	**Date:**	December 20, 2002
Exports:	TIFF, JFIF			**Type:**	Scanner Library
Uses:				**License:**	GPL
Url:	`http://panda.mostang.com/sane/`				
Author(s):	David Mosberger, others				

SANE stands for "Scanner Access Now Easy". It is an application programming interface (API) that provides standardised access to any raster image scanner hardware (flatbed scanner, hand-held scanner, video- and still-cameras, frame-grabbers, etc.). The SANE API is public domain and its discussion and development is open to everybody. The current source code is written for UNIX (including GNU/Linux). Several front ends to SANE are available.

H.2.12 vsound

Package:	vsound	**Version:**	0.5.5	**Date:**	December 20, 2002
Exports:	WAV			**Type:**	Recording program
Uses:				**License:**	GPL
Url:	`http://www.xenoclast.org/vsound/`				
Author(s):	Erik de Castro Lopo				

VSound can be compared with a virtual audio loopback cable. It allows to record the output audio stream of a program. This is similar to the process of connecting a loopback cable to the line-in and line-out jacks on the sound card, and recording the sound from the line-in jack without the DA/AD conversion losses. VSound can be used as part of a RealAudio-to-WAV file converter.

H.3 Production and Encoding

H.3.1 Heroine Virtual: Cinelerra

Package:	Heroine Virtual: Cinelerra	**Version:**	1.1	**Date:**	December 18, 2002
Exports:				**Type:**	library
Uses:	xmovie, libmpeg3 mpeg-2-movie, quicktime4linux (all included)			**License:**	
Url:	`http://heroinewarrior.com/cinelerra.php3`				
Author(s):	Heroine Virtual Ltd.				

Website: "Heroine Virtual Ltd. presents an advanced composing and editing system for native Linux at no cost to users. Cinelerra is not for consumer use. [...] If ease of use, simplicity, and convenience are your thing, you should use Virtualdub, Kino, MJPEG tools or MainActor instead." Cinelerra provides Renderfarm support.

H.3.2 Kino

Package:	Kino **Version:** 0.6	**Date:**	October 19, 2002
Exports:	Uncompressed video	**Type:**	Non-Linear Editor
Uses:		**License:**	GPL
Url:	`http://kino.schirmacher.de/`		
Author(s):	Arne Schirmacher		

Kino is a non-linear DV editor for GNU/Linux. It features integration with IEEE-1394 for capture, VTR control, and recording back to the camera. It captures video to disk in RawDV and AVI format, in both type-1 DV and type 2-DV (separate audio stream) encoding.

H.3.3 EffecTV

Package:	EffecTV **Version:** 0.3.8	**Date:**	December 18, 2002
Exports:	Uncompressed video	**Type:**	application
Uses:	video4linux/bttv-driver, SDL, GCC w/NASM, vloopback	**License:**	GPL
Url:	`http://effectv.sourceforge.net`		
Author(s):	Kentarou Fukuchi, Sam Mertens, Matthias Kleinmann, Buddy Smith, Christian W. Zuckschwerdt, Christian Berger, Ed Tannenbaum, Ico Doornekamp, Jun Ilo		

EffecTV offers a compilation of visual effects and a program that applies these in real-time to captured video. Built on existing libraries for capturing content (bttv-driver) and accessing graphics frame buffers (SDL - Simple DirectMedia Layer). Available for PlayStation 2 running Linux. The optional vloopback package lets other capture programs use the processed frames as if they were from the original data source.

H.3.4 Film Gimp

Package:	Film Gimp **Version:** 0.11	**Date:**	December 15, 2002
Exports:		**Type:**	Motion Picture Editor
Uses:	GTK+ >= 1.2.10, libtiff, libpng, libjpeg	**License:**	GPL
Url:	`http://filmgimp.sourceforge.net/`		
Author(s):	Robin Rowe, Sourceforge		

Film Gimp is a motion picture editing tool primarily used for painting and retouching of movies. Film Gimp is the most successful open source tool in feature motion picture work today. Film Gimp runs on Linux and SGI Irix. A Windows version is planned for release in December 2002, and Macintosh native in 2003.

H.4 Audio production and sound editing

H.4.1 Audacity

Package:	Audacity **Version:** 1.1.1	**Date:**	December 3, 2002
Exports:		**Type:**	Audio editor
Uses:	wxWindows	**License:**	GPL
Url:	`http://audacity.sourceforge.net/`		
Author(s):	Dominic Mazzoni		

Audacity is an audio editor which can record sounds, play sounds, import and export WAV, AIFF, and MP3 files, and more. It can be used to edit sounds using Cut, Copy and Paste (with unlimited Undo), mix tracks together, or apply effects to recordings. It also has a built-in amplitude envelope editor, a customisable spectrogram mode and a frequency analysis window for audio analysis applications. Built-in effects include Bass Boost, Wahwah, and Noise Removal, and it also supports VST plug-in effects. Audacity uses wxWindows, a cross platform GUI toolkit, available from `http://www.wxwindows.org`.

H.4.2 Common Lisp Music

Package:	CLM	**Version:**	2	**Date:**	December 2001
Exports:				**Type:**	Music synthesis
Uses:	Common Lisp environment			**License:**	Free
Url:	`http://www-ccrma.stanford.edu/software/clm/`				
Author(s):	Bill Schottstaedt				

Common Lisp Music (CLM) is a music synthesis and signal processing package in the Music V family.

H.4.3 Common Music Notation

Package:	CMN	**Version:**		**Date:**	December 2002
Exports:				**Type:**	Music notation
Uses:	Common Lisp environment			**License:**	Free
Url:	`http://www-ccrma.stanford.edu/software/cmn/`				
Author(s):	Bill Schottstaedt				

Common Music Notation (CMN) is a Lisp package that can create and display traditional western music scores.

H.4.4 DAP

Package:	DAP	**Version:**	2.1.4	**Date:**	
Exports:				**Type:**	Audio processor
Uses:	XForms			**License:**	GPL
Url:	`http://www.cee.hw.ac.uk/~richardk/`				
Author(s):	Richard Bruce Kent				

DAP is a digital audio processor for Unix, originally developed for SGI. It now supports Solaris, Linux and ships with several distributions of the Linux operating system.

H.4.5 GNUsound

Package:	GNUsound	**Version:**	0.3.7	**Date:**	
Exports:				**Type:**	Audio editor
Uses:				**License:**	GPL
Url:	`http://awacs.dhs.org/software/gnusound/`				
Author(s):	Pascal Haakmat				

GNUsound is a sound editor for Linux/x86. It supports multiple tracks, multiple outputs, and 8, 16, or 24/32 bit samples. It can read a number of audio formats through libaudiofile, and saves them as WAV.

H.4.6 MUSE

Package:	MUSE	**Version:**	0.5.2	**Date:**	
Exports:				**Type:**	Audio editor
Uses:				**License:**	GPL
Url:	`http://muse.seh.de/`				
Author(s):					

MusE is a MIDI/Audio sequencer with recording and editing capabilities. Not tested due to problems with installation.

H.4.7 Rezound

Package:	Rezound	**Version:**	0.4.0beta	**Date:**	
Exports:				**Type:**	Audio editor
Uses:				**License:**	GPL
Url:	`http://rezound.sourceforge.net/`				
Author(s):	Davy Durham				

Rezound ReZound aims to be a stable, open source, and graphical audio file editor primarily for but not limited to the Linux operating system.

H.4.8 TerminatorX

Package:	TerminatorX	**Version:**	3.73	**Date:**	December 16, 2002
Exports:				**Type:**	Audio synthesizer
Uses:	GTK+ >= 2.0, libaudiofile, libvorbis, mad			**License:**	GPL
Url:	`http://terminatorx.cx/`				
Author(s):	Alexander Koenig				

TerminatorX is a realtime audio synthesizer that allows the user to "scratch" on digitally sampled audio data (*.wav, *.au, *.ogg, *.mp3, etc.) the way hiphop-DJs scratch on vinyl records. It features multiple turntables, realtime effects (buit-in as well as LADSPA plugin effects), a sequencer and an easy-to-use GUI based on GTK+.

H.5 Encoders and Transcoders

H.5.1 sampeg-2

Package:	sampeg-2	**Version:**	0.6.4	**Date:**	
Exports:				**Type:**	encoder
Uses:				**License:**	GPL
Url:	`http://rachmaninoff.ti.uni-mannheim.de/sampeg/`				
Author(s):	Dirk Farin				

SAMPEG-2 is a software MPEG-1 and MPEG-2 encoder, designed to generate good image quality and provide a flexible framework for the design of new coding algorithms.[14]

The features include:

- supports parallel encoding on SMP systems with very high speed-ups,
- scene-change detection integrated to provide good reference frames for motion-estimation at scene-changes, and to exploit the temporal masking effect of the human visual system,
- adaptive quantization used to expoit the activity masking effect,
- several standard motion-estimation algorithms are implemented,
- MJPEG-AVI input with audio extraction,
- direct BTTV-device input for real-time encoding,
- MMX (x86) and VIS (UltraSparc) optimized code available.

H.5.2 ffmpeg

Package:	FFmpeg	**Version:**	CVS-2002-12-9	**Date:**	December 10, 2002
Exports:				**Type:**	Encoder
Uses:				**License:**	(L)GPL
Url:	`http://ffmpeg.sourceforge.net/`				
Author(s):	Gerard Lantau, Sourceforge				

The libavcodec library, which is a significant part of ffmpeg, provides a lot of codecs, such as RealVideo 1.0, MJPEG, H263, H263+, as well as MPEG-1 (VCD) and MPEG-2 (DVD) video and

[14] The development of SAMPEG-2 has been set on hold in favour of the SAMPEG-4 project which will also provide an integrated MPEG-4 encoder.

audio codecs. (mpeg1-video, mpeg-1/2 audiolayer 2, DivX4/5 (mpeg4), MS-mpeg4-v3 (DivX3), H.263+, mjpeg, AC3).

H.5.3 libmpeg2

Package:	libmpeg2	**Version:**	0.3.0	**Date:**	November 27, 2002
Exports:				**Type:**	library
Uses:				**License:**	GPL
Url:	`http://libmpeg2.sourceforge.net/`				
Author(s):	Sourceforge				

libmpeg2 is a free library for decoding MPEG-2 and MPEG-1 video streams.

H.5.4 MEncoder

Package:	MEncoder	**Version:**	0.90rc1	**Date:**	December 7, 2002
Exports:				**Type:**	
Uses:				**License:**	GPL
Url:	`http://www.mplayerhq.hu/`				
Author(s):	A'rpi/ESP-team, MPlayer community				

MEncoder can encode/transcode from MPlayer-playable movies (codecs include AVI, VCD, VOB, MPG, MOV, VIV, FLI, RM, NUV, NET) to other MPlayer-playable codecs like DivX4 (1 or 2 passes), libavcodec, PCM/MP3/VBRMP3 audio. MEncoder also supports stream copying and video resizing.

H.5.5 lame

Package:	lame	**Version:**	3.93	**Date:**	December 20, 2002
Exports:	lame			**Type:**	tools, mp3-encoder
Uses:	lavtools			**License:**	LGPL
Url:	`http://www.mp3dev.org`				
Author(s):	Mike Cheng				

LAME originally stood for LAME Ain't an Mp3 Encoder. The reason for this is that it was only a patch for the ISO demonstration mp3-encoder source. In May 2000 the LAME project became a stand-alone mp3-encoder with no dependency of the ISO source. Lame can encode MPEG1/2 Layer 3 audio with constant and variable bitrate.

H.5.6 liba52

Package:	liba52	**Version:**	0.7.4	**Date:**	December 10, 2002
Exports:				**Type:**	library
Uses:				**License:**	GPL
Url:	`http://liba52.sourceforge.net/`				
Author(s):	liba52 Team, Sourceforge				

liba52 is a library for decoding ATSC A/52 streams. The A/52 standard (also known as AC-3) is used in a variety of applications, including digital television and DVD. The liba52 distribution contains a test program, a52dec. It decodes ATSC A/52 streams, and also includes a demultiplexer for MPEG-1 and MPEG-2 program streams.

H.5.7 libdv

Package:	libdv	**Version:**	0.98	**Date:**	July 29, 2002
Exports:				**Type:**	library
Uses:				**License:**	GPL
Url:	`http://libdv.sourceforge.net`				
Author(s):	Charles Krasic and Erik Walthinsen, Oregon Graduate Institute of Science and Technology				

Libdv was developed according to the official standards for DV video: IEC 61834 and SMPTE 314M. The Quasar DV codec (libdv) is a software codec for DV video, the encoding format used by most digital camcorders, typically those that support the IEEE 1394 (a.k.a. FireWire or i.Link) interface.

H.5.8 mpegtools

Package:	dvb-mpegtools	**Version:**	0.9.4	**Date:**	2002-03-21
Exports:				**Type:**	applications
Uses:	code from MPEG			**License:**	GPL
Url:	`http://people.debian.org/~blade/dvb.html`				
Author(s):	Eduard Bloch, Markus Metzler (2001), Christoph Moar (1995)				

mpegtools are a variety of tools that transform between various MPEG stream formats. The package contains applications for conversion between several MPEG stream types: TS, PES, and ES. A special format for the Siemens DVB card is also included. Additionally there are tools to retrieve characteristics and type of a stream.[15]
The functions include:

- streamtype (information about stream)
- ts2pes (transforms a transport stream into an MPEG2 program stream)
- ts2av_pes transforms a TS into the AV_PES format used by the Siemens dvb card.
- pes2aud, pes2vid, pes2aud_es, pes2vid_es: extract audio or video streams from PES or PS streams depending on their respective names. The _es part of the name indicates that elementary streams are extracted in contrast to PES streams.
- av_pes2aud, av_pes2vid_es: same as above but for AV_PES streams.
- pes2av_pes, ps2av_pes, av_pes2ps, av_pes2pes, av_pes2ts: transform AV_PES in PS, PES and TS, respectively.
- check_av_pes: Gives some information about the AV_PES packets while outputting the AV_PES stream.
- tspids: return PIDs available in a stream.
- es2av_pes: turns a video ES into AV_PES.
- pes2ts2: transforms a PES stream into a TS (muxed).
- analyze: gives information about a PS/PES stream and all the packets contained therein.
- pes_demux, es_demux: demuxes a PS/PES into separate audio and video streams.
- ts_demux: demuxes a TS into PES.
- ts_es_demux: demuxes a TS into PES.
- ts2es: demuxes a TS into ES, but only for the given PID.
- pesplot: returns the audio and video PTS of the PES packets in a gnuplot plattable format.
- remux: is the first try of a program to remux a PS/PES stream.
- pes_repack: repacks your PES packets into smaller sizes.
- mplex: the main part of mplex is a rewrite of the multiplexing routines of bbmpeg, (mplex by Christoph Moar).

H.5.9 transcode

Package:	transcode	**Version:**	0.6.2	**Date:**	December 10, 2002
Exports:				**Type:**	tools
Uses:				**License:**	GPL
Url:	`http://www.theorie.physik.uni-goettingen.de/~ostreich/transcode/`				
Author(s):	Thomas Östreich				

transcode is a text console video-stream processing tool that supports elementary video and audio frame transformations. Some example modules, e.g. ffmpeg (see H.5.2), are included to

[15] mpegtools is included in the HIOF DVB package. It is also part of the dvb Debian package.

enable import of MPEG-1/2, Digital Video, and other formats. It also includes export modules for writing to AVI files with DivX, OpenDivX, XviD, Digital Video or other codecs. Direct DVD transcoding is also supported. A set of tools is available to extract and decode the sources into raw video/audio streams for import and to enable post-processing of AVI files.

H.5.10 mjpegtools

Package:	mjpegtools	**Version:**	1.6.1	**Date:**	December 10, 2002
Exports:	lavtools, xlav, mplex, aenc, mjpeg			**Type:**	tools
Uses:	lavtools			**License:**	GPL
Url:	`http://mjpeg.sourceforge.net/`				
Author(s):	Gernot Ziegler, Ronald Bultje, Bernhard Praschinger, Andrew Stevens				

The **mjpegtools** package provides Linux Audio and Video tools Motion-JPEG and MPEG. It implements MJPEG recording and playback, and simple cut-and-paste editing as well as MPEG compression of audio and video. The code has partially been written by the mjpegtools authors, but also code from various other open source projects is included. The package consists of the following parts / directories:

- **lavtools, xlav and utils:** contains a version of lavtools by Rainer Johanni, to handle AVI and Quicktime MJPEG files. The package contains applications to record, encode, replay and decode these files, user interface components, and some conversion tools for adding and extracting components (`lav2yuv`, `lav2wav`). The package also includes image processing filter utilities for raw video: medianfilter, scaling, and transitions. It also contains an encoder to MPEG-1/2 video streams from the YUV-format.
- **aenc:** Contains the source files for "mp2enc" which is MPEG-1 layer 2 audio compressor.[16]
- **mplex:** Mplex is a simple two-stream audio/video multiplexer for MPEG-1/MPEG-2. It accepts an MPEG-1/2 video stream and/or an MPEG layer II/III audio stream and multiplexes them into a combined program/system stream according to the constraints specified. Many different types of output structure are supported along with presets for standard VCD and SVCD streams[17]. Mplex is capable of automatically splitting the output stream into chunks of a specified size either independently or a sequence end/start points in the input video stream.
- **mjpeg, utils:** contains libraries to handle the new MJPEG movtar video format and a library to simplify MJPEG software and hardware playback.

H.5.11 MSYS toolkit

Package:	MSYS toolkit	**Version:**	1.0	**Date:**	1995-01-06
Exports:	encode, decode			**Type:**	appl/library
Uses:				**License:**	educational
Url:	`ftp://ftp.tek.com/tv/vnd/`				
Author(s):	Guy Cherry, Tektronix				

The MSYS Toolkit is a set of C++ classes for building and manipulating MPEG system streams. These classes can be used to build MPEG system multiplexers and demultiplexers, ond other MPEG related applications. The package is written to be instructional, and is therefor not optimized for efficiency.[18]

[16] It is not particularly good as encoders go, but is included for simplicity and completeness. It also has the virtue that for transcoding applications (e.g. AC3 to mp2) it can do sampling rate conversions. You need this to compress audio.

[17] VCD and SVCD streams can be used in the `vcdimager` software to produce (S)VCD media.

[18] The software is no longer available, as also noted on the web site `http://mpeg.org`.

H.6 Player software and presentation

H.6.1 Monkey Media

Package:	Monkey Media	**Version:**	0.5	**Date:**	December 20, 2002
Exports:				**Type:**	Media Library
Uses:	gnome-vfs, gstreamer			**License:**	LGPL
Url:	`http://people.nl.linux.org/~jorn/monkey-media.html`				
Author(s):	Jorn Baayen, Marco Pesenti Gritti, Bastien Nocera, Seth Nickell				

Monkey Media is a library for multimedia playback (of which only audio is supported at this point), offering support for seeking, managing volume, obtaining stream information like ID3 tags and bitrate, equalizer, and GStreamer visualization plugins.

H.6.2 Rhythmbox

Package:	Rhythmbox	**Version:**	0.4.1	**Date:**	December 20, 2002
Exports:				**Type:**	Player
Uses:	gstreamer			**License:**	GPL
Url:	`http://www.rhythmbox.org/`				
Author(s):	Jorn Baayen, Olivier Martin				

Rhythmbox takes its inspiration from Apple's iTunes application. The functionality includes importing audio CDs into mp3 or Ogg Vorbis format, play these music files and other music files, and burn new audio CDs from music files. Several visualisation plugins are available.

H.6.3 MPlayer

Package:	MPlayer	**Version:**	0.60	**Date:**	
Exports:				**Type:**	application
Uses:	libmpeg2			**License:**	GPL
Url:	`http://www.mplayerhq.hu/`				
Author(s):					

MPlayer is a movie player for Linux/Unix systems, e.g., based on X11. MPEG-1 and MPEG-2 formats are decoded with the native library libmpeg2. MPlayer supports a wide range of output drivers: X11, Xv, DGA, OpenGL, SVGAlib, fbdev, AAlib, SDL, VESA, and low-level drivers specific to hardware cards (Matrox, 3DFX and Radeon). Most of these drivers support software or hardware scaling for full-screen playback. MPlayer also support some MPEG decoder boards, such as DVB and DXR3/3 Hollywood+. An impressive list of supported software codecs is available from `http://www.mplayerhq.hu/DOCS/codecs-status.html`. Some codecs are decoded using modified win32 libraries.

H.6.4 XINE

Package:	Xine	**Version:**	0.9.13	**Date:**	October 3, 2002
Exports:				**Type:**	
Uses:				**License:**	GPL
Url:	`http://xine.sourceforge.net/`				
Author(s):	Guenter Bartsch				

XINE plays MPEG-2 and MPEG-1 video, DVDs (unlocked/unencrypted only), video CDs, SVCDs, and AVI files (using proprietary Windows 32 codecs) with synchronised audio and video, and optionally full-screen using the Xv extensions in Xfree86 4.x. XINE has a modular, multi-threaded architecture for input, demuxer, decoder, and output plugins.

H.6.5 gxine

Package:	gxine **Version:** 0.2	**Date:**	December 10, 2002
Exports:		**Type:**	application
Uses:		**License:**	GPL
Url:	`http://xine.sourceforge.net/`		
Author(s):	Guenter Bartsch		

gxine is both a GNOME frontend and a Mozilla plugin based on the core XINE library, xine-lib.

H.6.6 XMMS

Package:	XMMS **Version:** 1.2.7	**Date:**	December 20, 2002
Exports:		**Type:**	Multimedia player
Uses:		**License:**	GPL
Url:	`http://www.xmms.org`		
Author(s):	Peter Alm		

XMMS is a multimedia player for Unix systems. XMMS stands for X Multimedia System and can play media files such as MP3, MOD's, WAV and others with the use of Input plugins.

H.6.7 XMovie

Package:	XMovie **Version:** 1.9.7	**Date:**	December 10, 2002
Exports:		**Type:**	application
Uses:		**License:**	GPL
Url:	`http://heroinewarrior.com/download.php3`		
Author(s):	Heroine Virtual		

XMovie supports playback of MPEG-1, MPEG-2, MP3 audio, MP2 audio, AC3 audio and a range of other formats. According to the authors, "XMovie is primarily used for uncompressed, high resolution playback of output from an editing program". No statements about synchronised playback.

H.6.8 mpeg_play

Package:	mpeg_play **Version:** 2.4	**Date:**	April 6, 1998
Exports:		**Type:**	application
Uses:		**License:**	GPL
Url:	`http://bmrc.berkeley.edu/frame/research/mpeg/mpeg_play.html`		
Author(s):	Lawrence A. Rowe, Ketan Patel, Brian Smith		

The Berkeley MPEG Player, mpeg_play, is an MPEG player written in C. It uses X11 to display the decoded movies by default. It can optionally produce PPM files, SVGA graphics (Linux), Windows graphics calls, or work in a Mac window. It does not handle real-time synchronisation or audio streams.

H.6.9 Alsaplayer

Package:	Alsaplayer **Version:** 0.99.73	**Date:**	December 2002
Exports:		**Type:**	Audio Player
Uses:		**License:**	GPL
Url:	`http://www.alsaplayer.org/`		
Author(s):	Andy Lo-A-Foe		

AlsaPlayer is a PCM player. It is heavily multi-threaded and tries to exercise the ALSA library and driver. It has some very interesting features unique to Linux/Unix players. The goal is to create a fully pluggable framework for playback of all sorts of media with the focus on PCM audio data.

H.6.10 VideoLAN Client

Package:	VideoLAN Client	**Version:**	0.4.6	**Date:**	December 2002
Exports:				**Type:**	application
Uses:				**License:**	GPL
Url:	`http://www.videolan.org/vlc/`				
Author(s):	Students from the École Centrale Paris				

The VideoLAN Client (VLC) is a MPEG player, that support MPEG-2 TS unicasted or multicasted and HTTP protocol over IPv4 and IPv6 networks. VLC also support slow motion, fast forward, pause, arbitrary jumps within the stream, DVD chapter selection, on-the-fly language and subtitles selection. It uses X11 to display decoded movies.

H.6.11 gqview

Package:	gqview	**Version:**	1.2.0	**Date:**	December 20, 2002
Exports:				**Type:**	Image browser
Uses:				**License:**	GPL
Url:	`http://gqview.sourceforge.net`				
Author(s):	GQview team				

Image browser that features single click access to view images and move around the directory tree.

H.6.12 DirectPresenter

Package:	DirectPresenter	**Version:**		**Date:**	December 16, 2002
Exports:				**Type:**	ISO image generation
Uses:	DirectFB			**License:**	GPL
Url:	`http://team.gcu-squad.org/~fab/DP/`				
Author(s):	Fab				

H.6.13 GImageView

Package:	GImageView	**Version:**	0.2.10	**Date:**	December 20, 2002
Exports:				**Type:**	Image browser
Uses:				**License:**	GPL
Url:	`http://gtkmmviewer.sourceforge.net/`				
Author(s):	Takuro Ashie, Nyan2				

GImageView is a GTK+ based image viewer. It support tabbed browsing, thumbnail table views, directory tree views, drag and drop, reading thumbnail cache of other famous image viewers, and flexible user interface.

H.7 Streaming and other servers

H.7.1 Icecast

Package:	Icecast	**Version:**	1.3.12	**Date:**	2001
Exports:				**Type:**	Audio streamer
Uses:				**License:**	GPL
Url:	`http://www.icecast.org/`				
Author(s):	Jack Moffitt, Barath Raghavan, Alexander Havang, Chad Armstrong and Jeremy Katz				

Icecast is an Internet audio broadcasting system based on MPEG audio technology. It implements to broadcast an audio stream to as many people as their bandwidth can support. The Icecast server supports clients as e.g., **shout**, WinAmp, or xmms.

H.7.2 VideoLAN Server

Package:	VideoLAN Server	**Version:**	0.4.4	**Date:**	11 Aug 2002
Exports:	MPEG transport stream			**Type:**	MPEG streamer
Uses:	libdvdread to play DVDs, libdvdcss for encrypted DVDs, libdvbpsi for DVB input			**License:**	GPL
Url:	`http://www.videolan.org/vls`				
Author(s):					

VideoLAN is a project of students from the École Centrale Paris and other developers. The VideoLAN Server (vls) is designed for handling many MPEG sources and broadcasting data over an IP network. VideoLan runs on GNU/Linux, Solaris, and Windows (port). The following MPEG sources are supported: MPEG1 PS, MPEG2 PS, MPEG2 TS, DVD, the Kfir video encoder and the WinTV-Nova satellite receiver. The output is MPEG2-TS over an IP network or to a file.

H.8 Metadata, search and retrieval

H.8.1 Advanced Authoring Format

Package:	Advanced Authoring Format	**Version:**	1.0.1	**Date:**	July 16, 2002
Exports:				**Type:**	SDK
Uses:				**License:**	
Url:	`http://sourceforge.net/projects/aaf/`				
Author(s):					

The file format **Advanced Authoring Format (AAF)** for use in professional multimedia creation and authoring comes with a cross-platform SDK. The AAF specification is administered by the AAF Association, which also coordinates development of the SDK.

H.9 Mastering

H.9.1 vcdimager

Package:	vcdimager	**Version:**	0.7 (unstable)	**Date:**	
Exports:				**Type:**	application
Uses:				**License:**	GPL
Url:	`http://www.vcdimager.org`				
Author(s):	Herbert Valerio Riedel				

GNU **VCDimager** is a mastering suite for authoring, disassembling and analyzing Video CD's and Super Video CD's. The core functionality consists of directly making Video CD BIN/CUE-style CD images from mpeg files, which (after being written to CDR(W) media) can be played on standalone VCD players or DVD players and on computers running GNU/Linux, MacOS, Win32 or any other OS capable of accessing VCD's. BIN/CUE images can be burned with the program **cdrdao** under GNU/Linux.

H.9.2 cdparanoia

Package:	cdparanoia	**Version:**	III-alpha9.8	**Date:**	March 2001
Exports:				**Type:**	application
Uses:				**License:**	GPL
Url:	`http://xiph.org/paranoia/`				
Author(s):	Monty				

Cdparanoia is a Compact Disc Digital Audio (CDDA) extraction tool. The application is built on top of the Paranoia library, which is doing the real work (the Paranoia source is included in the cdparanoia source distribution). Like the original cdda2wav, cdparanoia package reads audio from the CDROM directly as data, with no analog step between, and writes the data to a file or pipe in WAV, AIFC or raw 16 bit linear PCM.

H.9.3 mpg32ogg

Package:	mpg32ogg **Version:** 0.11	**Date:**	December 2002
Exports:		**Type:**	application
Uses:		**License:**	Artistic
Url:	`http://faceprint.com/code/`		
Author(s):	Nathan Walp		

mpg32ogg is a *Perl* script to convert MP3 files to Ogg Vorbis files, retaining ID3 information, bitrate, and optionally renaming the output files, as well as deleting the originals.

H.10 Digital TV software

We only list the packages that are of a more general nature. We do not list packages that need the LinuxTV kernel (see `linuxtv.org`), or need special DVB cards. More information in digital TV software can be found in a separate report [7].

H.10.1 DVBstream

Package:	dvbstream **Version:** 0.4pre2	**Date:**	2002-01-31
Exports:		**Type:**	library,application
Uses:	ts-rtp package from LinuxTV	**License:**	GPL
Url:	`http://linuxstb.org/dvbstream/`		
Author(s):	David Podeur, Dave Chapman		

DVBstream is based on the ts-rtp package available at `http://www.linuxtv.org`. It broadcasts a (subset of a) DVB transport stream over a LAN using the rtp protocol. There were a couple of small bugs in the original ts-rtp application, which have been fixed within this project. This software requires Linux, a supported DVB card, the Linux DVB drivers from `www.linuxtv.org` and a kernel with Multicast networking enabled.

H.10.2 librtp

Package:	librtp **Version:** 0.1-1	**Date:**	2001
Exports:		**Type:**	library
Uses:		**License:**	GPL
Url:	`http://librtp.sourceforge.net`		
Author(s):	Roland Dreier		

Librtp is a library for writing RTP/RTCP applications. The code was originally part of gnome-o-phone (see `http://gphone.sourceforge.net`).

H.10.3 HIØ DVB package

Package:	HIØ DVB package **Version:**	**Date:**	2001-10-31
Exports:		**Type:**	collection
Uses:	Mplayer 0.50, bfr 1.3, dvbstream 0.3, mpegtools	**License:**	GPL
Url:	`http://`		
Author(s):	Håvard Rast Blok, Høgskolen i Østfold		

This package contains the programs neccessary for receiving the RTP stream of high quality, MPEG2 video from Høgskolen i Østfold.[19] The package includes the following programs and libraries:

- LinuxTV (mpegtools and DVB driver) `http://www.linuxtv.org/`
- mpegtools, see Section H.5.8.

[19] The HIØ DVB package can be seen as a small add-on distribution for digital TV software to Linux. The purpose of this distribution is to turn a Linux computer into a set-top box.

- DVBStream (dumprtp) version 0.3 from `http://www.linuxstb.org/`. See Section H.10.1. dvbstream is used to broadcast a DVB transport stream over a LAN using the RTP protocol.
- The buffering software `bfr 1.3` by Mark Glines from `http://www.glines.org/software/buffer.html`. The software buffers data in order to avoid artefacts from jitter and delays on the network.
- OST (some header files only, probably necessary for a stand-alone version.)
- MPlayer 0.50.

H.11 IP phones, multimedia phones and phone conference equipment

H.11.1 Gnome-o-Phone

Package:	Gnome-o-Phone	**Version:**	0.5.2	**Date:**	
Exports:				**Type:**	Internet Telephone
Uses:	librtp, libgsm, GNOME			**License:**	GPL
Url:	`http://gphone.sourceforge.net/`				
Author(s):	Roland Dreier, Rob Johnson				

Gnome-o-Phone (gphone for short) is an Internet telephone. Calls are made over the Internet without using the PSTN infrastructure. gphone uses RTP/RTCP, and gphone complies to the standard to be able to talk to *speakfreely*. Gphone requires the GSM compression library.

H.11.2 Linphone

Package:	Linphone	**Version:**	0.9.0	**Date:**	
Exports:				**Type:**	
Uses:				**License:**	GPL
Url:	`http://www.linphone.org/`				
Author(s):	Simon Morlat				

Linphone is a web phone for the Gnome Desktip und Linux (can also be used under KDE). It works with a simple user interface. Since version 0.9.0, linphone can be compiled and used without gnome, in console mode, by using the program called *linphonec* . Linphone includes a variety of codecs. The Speex codec allows high quality talks even with slow internet connections, like 28k modems. Linphone understands the SIP protocol.

H.12 Video conference

H.12.1 GnomeMeeting

Package:	GnomeMeeting	**Version:**	0.93	**Date:**	
Exports:				**Type:**	Video conference
Uses:	OpenH323, PWlib			**License:**	GPL
Url:	`http://www.gnomemeeting.org/`				
Author(s):	Damien Sandras				

GnomeMeeting is a GNU/Linux H.323 client, which is compatible with Netmeeting and other H.323 products. It has been designed for the Gnome desktop, but it also works with other environments.[20]

GnomeMeeting is H.323 compliant. It supports H.245 Tunneling, Fast Start, auto answering of incoming calls, and do not disturb mode. It GnomeMeeting supports the following audio codecs: LPC10, GSM-06.10, MS-GSM, G.711-Alaw, G.711-uLaw, G.726; and video codecs: H.261-QCIF and H.261-CIF. Other features include Quicknet hardware devices, the G.732.1 audio codec, `callto://` URLs, gatekeeper support, video bandwidth control and text chat.

[20] GnomeMeeting was not tested by us due to installation problems.

H.13 Text To Speech and Speech to text

H.13.1 Festival

Package:	Festival	**Version:**	1.4.2	**Date:**	
Exports:				**Type:**	Speech syntezis
Uses:				**License:**	GPL?
Url:	`http://www.cstr.ed.ac.uk/projects/festival/`				
Author(s):	Alan W Black (CMU), Rob Clark (CSTR), Richard Caley (CSTR), and Paul Taylor				

Festival is a general multi-lingual speech synthesis system developed at CSTR. It offers a full text to speech system with various APIs, as well an environment for development and research of speech synthesis techniques. It is written in C++ with a Scheme-based command interpreter for general control.

H.13.2 Xvoice

Package:	Xvoice	**Version:**	0.9.5	**Date:**	
Exports:				**Type:**	Speech to text
Uses:	ViaVoice			**License:**	GPL
Url:	`http://www.compapp.dcu.ie/~tdoris/Xvoice/`				
Author(s):	Brian Craft				

Xvoice Xvoice enables continuous speech dictation and speech control of X applications. It uses the IBM ViaVoice speech recognition engine to convert users' speech into text. When in dictation mode Xvoice passes this text directly to the currently focussed X application. In command mode, Xvoice matches the speech with predefined, user-modifieable, key sequences or commands.

ViaVoice for Linux must be downloaded separately from IBM (`www.ibm.com`).

I Other Multimedia-oriented distributions

We found other multimedia related distributions and distribution projects, and present some of these initiatives. Besides these distributions, there are Linux-oriented software packages available for special purposes. This includes the software distributed for specific set-top boxes, e.g., the Nokia Mediaterminal (See e.g., [7]).

I.1 AGNULA

AGNULA (acronym for A GNU/Linux Audio distribution, pronounced with a strong g) is the name of a project funded by the European Commission (number of contract: IST-2001-34879; key action IV.3.3, Free Software: towards the critical mass). The project is coordinated by the Centro Tempo Reale in Firenze (`http://www.centrotemporeale.it`) and involves several research centres and institutions:

- IRCAM (Institut de Recherche et Coordination Acoustique/Musique), Paris (`http://www.ircam.fr/`);
- Universitat Pompeu Fabra, Music Technology Group, Barcelona (`http://www.iua.upf.es/mtg`);
- Kungl. Tekniska Högskolan (KTH), Music Acoustics Group, Stockholm (`http://www.speech.kth.se/music`);
- Free Software Foundation Europe (`http://fsfeurope.org`);
- and Red Hat France (`http://www.redhat.fr/`).

AGNULA supports the development of derived distributions of Debian GNU/Linux and Red Hat GNU/Linux. Dedicated to audio and multimedia these distributions will be called DeMuDi (after Debian Multimedia Distribution) and ReHMuDi (after Red Hat Multimedia Distribution) with the following features: easy to use (auto-installing, with auto-configuration, plug-and-play), full-featured GNU/Linux system (completely configurable and customisable, reliable and secure, strongly linked to networking the Internet, and easy to upgrade); as in all Debian and in most Red Hat distributions, most of the software is released with GNU/GPL licence, and all included software will be open source software.

The deliverables and reports of the AGNULA project are available at `http://www.agnula.org/project/deliverables`

The packages included in AGNULA, according to `http://www.agnula.org/user/packages`, are grouped into the following categories:

- Audiodevel
- CD-tools
- Compression
- Conversion
- Digital DJ
- DSP
- MIDI
- ModTrack
- Multitrack
- Network
- Notation
- Play/Record
- Plugin
- Sounddriver
- Soundedit
- Synthesis
- Video
- Visual

I.2 DeMuDi: Debian Multimedia Distribution

DeMuDi is part of the AGNULA project. The focus is on building packages for the Debian distribution. The packages included in Demudi are grouped into the following categories [5]:

- Sound Editors
- Multitrackers
- Software Synthesis
- MIDI and Trackers
- Compression and Internet
- CD Production
- Miscellaneous Tools
- Notation
- Visualisation
- Speech
- Video and Graphics
- Development Tools

I.3 Planet CCRMA at Home

Planet CCRMA at Home is a collection of RPMS that can be added to a computer running Red Hat 7.2, 7.3 or 8.0 to transform it into an audio workstation. The features include a low-latency kernel, current ALSA audio drivers and a set of applications for music, MIDI and audio. The project was started by Juan Reyes and Fernando Lopez-Lezcano at the Stanford University Center for Computer Research in Music and Acoustics (CCRMA).

I.4 NIST DASE

ATSC DASE, the American Digital Interactive Television standard, have released a reference implementation of their STB which runs on Linux. The ATSC DASE spec is very similar to the MHP spec. It includes JavaTV, HAVi and DAVIC. The reference implementation includes full source code, a JavaTV implementation, and a HAVi implementation. See `http://www.cmr.nist.gov/dase/` for more information.

Part IV
Examples

In the following we show some examples on how to prepare software packages for the mediAkit distributions.

J Software Packaging

Media Folder is a simple, yet unfinished library written by one of the authors. It will be used as an example to demonstrate some steps in the software packaging process. The package is available from `http://www.nr.no/~oka/software/mfolder/`. For another example, see the GNU *hello* package from the FSF (`http://www.gnu.org`).

J.1 Package Layout

- `AUTHORS` - text file that describes how to reach the package authors
- `autogen.sh` - script to regenerate autoconf and automake setup.
- `ChangeLog` - text file for recording changes and modifications to the package
- `configure.ac` - autoconf-specific rules from which autoconf generates a platform-neutral configure script
- `COPYING` - text file with licensing terms of the software package
- `HACKING` - special development policies and conventions
- `INSTALL` - text file that describes how to install the software
- `Makefile.am` - *automake*-specific rules from which a standards-compliant `Makefile.in` is generated
- `mfolder.pc.in` - pkgconfig[21] configuration file to be processed by *autoconf*-specific variables with versioning and link details for libraries.
- `mfolder.spec.in` - RPM specification file to be processed by *autoconf* for RPM package building
- `NEWS` - a list of new features and modified functionality
- `README` - textual description of the software package.
- `src/` - the directory that contains the source code
- `src/Makefile.am` - *automake*-specific rules from which a standards-compliant `Makefile.in` is generated
- `src/mfolder.h` - C header file with function prototype and variable definitions
- `src/mfolder-types.h` - C header file with function prototype and variable definitions
- `src/mfolder.c` - C source file that the library is built from.
- `src/mfolder-tests.c` - C source file for test executables using the library
- `THANKS` - text file with a list of persons who contributed/helped.
- `TODO` - text file with description of planned improvements.

J.1.1 `Makefile.am`

`Makefile.am` in the top package directory contains *automake* specific build rules. It is possible to specify sub-directories, special directories on the installed system with a certain prefix where files (`mfolder.pc`) will be installed, and extra files that should go with the package (such as `mfolder.spec.in`, and the autoconf- generated `mfolder.spec` with the `Version:` field set to the to-be-released version of the package.

`$(libdir)/pkgconfig` is equal to `/usr/lib/pkgconfig`, when the source package is configured with the autoconf `prefix /usr`.

[21] *pkgconfig* is a program and *autoconf* macro to get compile and link flags for libraries the programs are linked with. It requires the library package to install a description file with the needed information.

We also specify the RPMFLAGS variable and a traditional *make* target for building RPM (B.2) packages of the package.

```
SUBDIRS = src
pkgconfigdir = $(libdir)/pkgconfig
pkgconfig_DATA = mfolder.pc
EXTRA_DIST = HACKING mfolder.spec.in mfolder.pc.in mfolder.spec

RPMFLAGS=       --define ``_sourcedir $$PWD'' \
                --define ``_builddir $$PWD/BUILD'' \
                --define ``_rpmdir $$PWD'' \
                --define ``_srcrpmdir $$PWD''
rpm: dist
        -$(RM) -r BUILD
        mkdir BUILD
        rpmbuild $(RPMFLAGS) -ba --clean mfolder.spec
```

Notice the rpm target which builds a SRPM and the binary RPM packages from the *autoconf*-generated `mfolder.spec` specification.

J.1.2 mfolder.pc.in

Below is a *pkgconfig*-configuration file to be replaced by a *autoconf*-generated version. The file contains versioning and link details for libraries. The content of the variables MFOLDER_PKG_DEPS and VERSION are expanded by *configure* and eventually stored in a file named `mfolder.pc`.

```
prefix=@prefix@
exec_prefix=@exec_prefix@
libdir=@libdir@
includedir=@includedir@

Name: mfolder
Description: Media Folder library
Requires: @MFOLDER_PKG_DEPS@
Version: @VERSION@
Libs: -L${libdir} -lmfolder
Cflags: -I${includedir}/mfolder-@VERSION@
```

J.1.3 mfolder.spec.in

Below is a RPM specification file to be processed by *autoconf* in order to make it easy to build new RPM packages for future releases of the package. The content of the VERSION variable is expanded by *configure* and stored in a new file, `mfolder.spec`.

```
%define ver @VERSION@
%define rel 1
%define prefix /usr

Summary: The Media Folder library
Name: mfolder
Version: %ver
Release: %rel
URL: http://www.nr.no/~oka/software/mfolder/
Source0: %{name}-%{version}.tar.gz
License: LGPL
Group: Development/Libraries
BuildRoot: %{_tmppath}/%{name}-root
Prereq: /sbin/ldconfig
Requires: gnome-vfs2 >= 2.0, glib2 >= 2.0
BuildRequires: gnome-vfs2-devel >= 2.0, glib2-devel >= 2.0
```

```
Prefix: %prefix

%description
Media Folder is a library for accessing and storing media files
in default locations.

%package devel
Summary: The static libraries and header files for Media Folder library
Group: Development/Libraries
Requires: %{name} = %{version}

%description devel
The mfolder-devel package contains the static libraries and header
files for the Media Folder library.  For developing
programs that use Media Folder, mfolder-devel must be installed.

%prep
%setup -q

%build
./configure --prefix=%prefix
if [ ‘‘$SMP’’ != ‘‘’’ ]; then
  (make ‘‘MAKE=make -k -j $SMP’’; exit 0)
  make
else
  make
fi

%install
rm -rf $RPM_BUILD_ROOT

%makeinstall

%post
/sbin/ldconfig

%postun
/sbin/ldconfig

%clean
rm -rf $RPM_BUILD_ROOT

%files
%defattr(-,root,root)
%doc README AUTHORS COPYING NEWS THANKS TODO ChangeLog
%{prefix}/lib/*.so.*

%files devel
%defattr(-,root,root)
%{_libdir}/*.a
%{_libdir}/*.la
%{_libdir}/*.so
%{_libdir}/pkgconfig/mfolder.pc
%{_includedir}/mfolder-%{ver}/*.h

%changelog
* Tue Nov 12 2002 Ole Aamot <oka@nr.no>
- Initial build.
```

J.1.4 configure.ac

A platform-independent configure script is expanded by *autoconf* from this template:

```
AC_INIT(src/mfolder.c)
AC_CANONICAL_TARGET([])
AM_INIT_AUTOMAKE(mfolder, 0.1.0)
AM_MAINTAINER_MODE
AM_CONFIG_HEADER(config.h)
AC_CONFIG_SRCDIR([src/mfolder.c])

# Checks for essential programs.
AC_PROG_CC
AC_PROG_CC_STDC
AC_ISC_POSIX
AC_HEADER_STDC
AM_PROG_LIBTOOL

# Check for essential libraries.
PKG_CHECK_MODULES(MFOLDER,
                  gnome-vfs-2.0 >= 2.0 \
                  glib-2.0 >= 2.0)
MFOLDER_PKG_DEPS=''glib-2.0, gnome-vfs-2.0''

# Checks for header files.
AC_HEADER_STDC
AC_CHECK_HEADERS([stdlib.h string.h errno.h sys/stat.h])

# Checks for typedefs, structures, and compiler characteristics.
AC_C_CONST

# Checks for library functions.
AC_CHECK_LIB(socket, connect)

# Modify the pkgconfig path.
AC_ARG_WITH(pkg-config-path,
   AC_HELP_STRING([--with-pkg-config-path],
                  [colon-separated list of pkg-config(1) dirs]),
  [export PKG_CONFIG_PATH=${withval}])

AC_SUBST(MFOLDER_LIBS)
AC_SUBST(MFOLDER_CFLAGS)
AC_SUBST(MFOLDER_PKG_DEPS)

AC_OUTPUT([
Makefile
mfolder.spec
mfolder.pc
src/Makefile
])
```

J.1.5 src/Makefile.am

```
CFLAGS= -g -Wall $(MFOLDER_CFLAGS) $(MFOLDER_CFLAGS) -DDEBUG_MFOLDER
LDFLAGS= $(MFOLDER_LIBS) $(MFOLDER_LIBS)

lib_LTLIBRARIES = libmfolder.la

libmfolder_la_SOURCES = mfolder.c
libmfolder_la_LDFLAGS = -version-info 0:1:0
libmfolder_la_LIBADD = # @INTLLIBS@
```

```
libmfolderincdir = $(includedir)/mfolder-@VERSION@
libmfolderinc_HEADERS = mfolder.h mfolder-types.h

DEPS = libmfolder.la
LDADDS = libmfolder.la

noinst_PROGRAMS = mfolder-tests mfolder-tests-dbg

# Build from mfolder-tests.c and libmfolder.la
mfolder_tests_SOURCES = mfolder-tests.c
mfolder_tests_LDADD = libmfolder.la # @INTLLIBS@

# Create an easier-to-debug version.
mfolder_tests_dbg_SOURCES = mfolder-tests.c
mfolder_tests_dbg_LDADD = libmfolder.la
mfolder_tests_dbg_LDFLAGS = -static $(LDFLAGS)
```

J.1.6 autogen.sh

The autogen.sh script contains

```
#!/bin/sh
autoheader
aclocal
automake
autoconf
```

If configure.ac or Makefile.am needs to be modified, then the system needs at least *autoconf* 1.6.3, *automake* 2.53, and *libtool* 1.4.2 in order to rebuild configure and Makefile.in with the autogen.sh script.

```
$ wget http://www.nr.no/~oka/software/mfolder/mfolder-0.1.0.tar.gz
$ tar zxvf mfolder-0.1.0.tar.gz
$ cd mfolder-0.1.0/
$ ./autogen.sh
```

J.2 Build Instructions

The following instructions are necessary to build the package. The machine in this example is running *rpm* 4.1, *rpm-build* 4.1, *wget* 1.8.2, and *tar* 1.13.25.

```
$ wget http://www.nr.no/~oka/software/mfolder/mfolder-0.1.0.tar.gz
$ tar zxvf mfolder-0.1.0.tar.gz
$ cd mfolder-0.1.0/
$ ./configure
$ make rpm
```

The rpm target generates a Source RPM in the current directory, and two Binary RPM packages in the i386/ subdirectory:

```
mfolder-0.1.0-1.src.rpm
i386/mfolder-0.1.0-1.i386.rpm
i386/mfolder-devel-0.1.0-1.i386.rpm
```

J.3 Install Instructions

To install the RPM packages on a compatible system, root permissions is required.

```
$ su
[enter root password]
# rpm -Uvh i386/mfolder-0.1.0-1.i386.rpm
# rpm -Uvh i386/mfolder-devel-0.1.0-1.i386.rpm
```

K Software Patches

When developers write enhancements or fixes to the original source package and would like to see the modifications included in the next version, it is common practice to use the *diff* program. Given the correct parameters *diff* creates a file that contains all of the modifications between two source files, the old without the modifications and the new file with the modifications according to special patch formatting rules.

K.1 Making a patch using *diff*

Although it is possible to use the *diff* program for only two files, it is useful to keep two separate directories:

- the original source tree
- the modified source tree

We made some modifications to `filmgimp.spec.in`, from which the RPM (see Section B.2) build specification is generated by *autoconf* for the filmgimp (H.3.4) package.

```
$ emacs filmgimp-0.11/filmgimp.spec.in
```

After the file (`filmgimp-0.11/filmgimp.spec.in`) was modified, we moved the original directory into a separate directory (`filmgimp-0.11.mod/`):

```
$ mv filmgimp-0.11/ filmgimp-0.11.mod/
```

Then the original source archive was uncompressed

```
$ tar zxvf filmgimp-0.11.tar.gz
```

At last *diff* was run and its output was compressed

```
$ diff --unified --recursive --new-file \
  filmgimp-0.11 filmgimp-0.11.mod/ >patch-spec-2002-12-18
$ gzip -9 patch-spec-2002-12-18
```

The steps produces a compressed patch file called `patch-spec-2002-12-18.gz`.

Example: `patch-spec-2002-12-18`

The content of `patch-spec-2002-12-18` is

```
--- filmgimp-0.11/filmgimp.spec.in      2002-12-04 08:10:08.000000000 +0100
+++ filmgimp-0.11.build/filmgimp.spec.in        2002-12-19 11:38:29.000000000 +0100
@@ -7,7 +7,12 @@
 Source0: http://prdownloads.sourceforge.net/filmgimp/filmgimp-@GIMP_VERSION@.tar.gz
 URL: http://filmgimp.sourceforge.net
 BuildRoot: %{_tmppath}/%{name}-root
-#Requires: /sbin/ldconfig FIXME: Find dependencies
+BuildRequires: gtk+-devel >= 1.2.10, glib-devel >= 1.2.10
+BuildRequires: libtiff-devel >= 3.5.7, libjpeg-devel >= 6b
+BuildRequires: libpng-devel >= 1.2.2
+Requires: gtk+ >= 1.2.10, glib >= 1.2.10,
+Requires: libtiff >= 3.5.7, libjpeg >= 6b, libpng >= 1.2.2
+PreReq: /sbin/ldconfig, grep, fileutils
 Prefix:    %{_prefix}

 %description
```

It is common and usually also expected that a `ChangeLog` entry such as

```
2002-12-19  Ole Aamot  <oka@naos.nr.no>

        * filmgimp.spec.in: Add BuildRequires: and Requires:
```

is submitted by mail with the patch file to the package maintainer team. It is not considered good practise to send a patch that contains direct modifications against the `ChangeLog` file.

K.2 Applying a patch using *patch*

When the developer team receives a new patch file created by *diff*, as above, the developers responsible for the package maintenance, will consider if the modifications are correct, and eventually reject or accept the patch into the official source tree by using the *patch* tool. To apply the patch file with *patch* in the top level directory:

```
$ cd filmgimp-0.11/
$ gzip -d < ../patch-spec-2002-12-18.gz | patch -p1
patching file filmgimp.spec.in
```

References

[1] Bahner, Furu, Heen, Ingebrigtsen, Kvalnes, Reinholdtsen, Trondsen, Trosterud, Wisløff, Yrvin. Skolelinux-prosjektet. Web: `http://cvs.skolelinux.no/info/rapporter/forprosjekt.pdf`, 2002.

[2] Ed Bailey. *Maximum RPM.* RedHat Press, 1997.

[3] Robert J. Chassell and Richard M. Stallman. *Texinfo: The GNU Documentation Format, for Version 4.* GNU Press, 1999.

[4] TimeSys Corporation. A timesys perspective on the linux preemptible kernel. Technical report, TimeSys Corporation, 2002.

[5] D. Phillips F. Déchelle, G. Geiger. Demudi: The debian multimedia distribution. In *ICMC2001*. available on http://www.demudi.org/doc/paper.pdf, 2001.

[6] S. Feldman. Make – a program for maintaining computer programs. *Software–Practice and Experience*, 9:225–265, 1979.

[7] W. Leister, S. Boudko, O. Aamot, and P. Holmes. Digital TV – A survey. Report 988, Norsk Regnesentral, Oslo, 2002.

[8] Rich Morin, editor. *Prime Time Freeware for UNIX, Issue 2-2.* Prime Time Freeware, Sunnyvale, CA, July 1993.

[9] J. Peek, T. O'Reilly, and M. Loukides, editors. *Unix Power Tools.* O'Reilly, Bantam Books, March 1993.

[10] Dave Phillips. Achieving low-latency response times under linux. O'Reilly Network, November 2000.

[11] Eric S. Raymond. *The Cathedral & the Bazaar.* O'Reilly, 1999.

[12] Richard M. Stallman. *Free Software, Free Society: Selected Essays.* GNU Press, 2002.

[13] Richard M. Stallman and Roland McGrath. *GNU Make: A Program for Directing Recompilation.* GNU Press, 2002.

[14] Guy Steele and Eric S. Raymond. The Jargon File. Web pages, `http://www.tuxedo.org/~esr/jargon/`, 2000.

[15] Free Standards Group Workgroup. Linux standard base specification 1.3. *Free Standards Group*, `http://www.linuxbase.org/spec/refspecs/LSB_1.3.0/gLSB/gLSB.pdf`, 2002.

Index