The monthly GNU Column

Brave GNU World

Welcome to another issue of the Brave GNU World. We may always have suspected it, but this issue presents some proof: Free Software is good for your health! BY GEORG C. F. GREVE

In this monthly column we bring you the news from within the GNU project. We aim to give you an insight into the programs and some of their philosophies. In this issue we will look at the ways Free Software is helping in the world of medicine and health. After that we take a look at another alternative to .NET and then finally onto some strategy games to take your mind off work.

Debian-Med

Like many areas, the area of medical applications has a large quantity of projects which are sometimes quite advanced, but it is beyond the skills of any "normal" user to assemble them or integrate them into a solution. To solve this, Andreas Tille began the Debian-Med [5] project in early 2002, which aims at customizing the Debian distribution for users in the medical and microbiological fields and seeks to integrate software in these areas.

Frequent readers of the Brave GNU World might be reminded of the Debian-Jr. [7] project introduced in issue #23, [6] and in fact the idea for Debian-Med was inspired by that project.

Software which needs to interact with the private and the very intimate spheres of human life – as is the case for doctors – has to fulfil certain basic criteria, which currently only Free Software can hope to fully satisfy.

It must, for instance, be sure that the confidentiality and security of patient data is upheld. This requires a certain transparency, which is best secured through a Free development process.

Also safety from data-loss can be very important, because some tests are a hazard to health and so sometimes cannot be repeated. If data gets lost, this is clearly reducing the quality of the medical service. At the same time it will normally cause a loss of trust by a patient concerning the physician.

Therefore this area requires a secure, trustworthy and stable fundament (the operating system) with similar applications. Using Free Software is more and more becoming a necessity for every conscientious physician.

Not surprisingly, privacy, protection of data, trustworthiness and security are at the top of the list of Debian-Med's aims.

Other core issues are ease of use, easy installation and administration.

The program is easy to use thus preventing both errors and frustration which would work against the patient's interests. Also easy installation and administration makes sure that conscientious physicians have fewer problems when moving to Free Software, which should be taken into consideration for practical reasons.

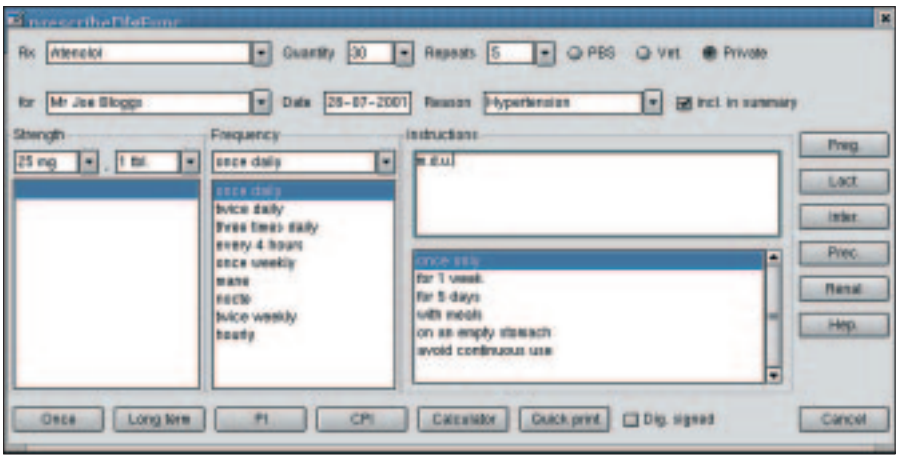

Figure 1: Gnumed prescription dialog

Currently the project, which consists of Andreas and about 70 more interested people, looks to find a solution for every common problem and to make it install-ready.

The mid-term perspective is to present Debian-Med as a real alternative to physicians and create a demonstration live CD.

Areas in most need of help are documentation and translation, but a logo is also still missing. Andreas has been thinking about a combination of Debian logo and snake for this.

The license status of the whole project is determined by the individual software licenses, of course. Free Software under a Debian Free Software Guidelines (DFSG) approved license is preferred.

Unfortunately the project also plans packaging proprietary software that is distributable at no cost, creating a weakness for the mid- and long-term perspectives. But if enough people express their wish to think in the long-term here, I'm sure that this decision is not carved in stone.

Bringing Free Software into the medical area is something that I always considered to be quite important and if you are looking for a useful project to engage yourself in, Debian-Med will most likely be a good choice.

Gnumed

Among the programs used within Debian-Med is Gnumed, [8] an official GNU project designed for a paperless medical practice.

The project was born in Australia, where a heated discussion about the

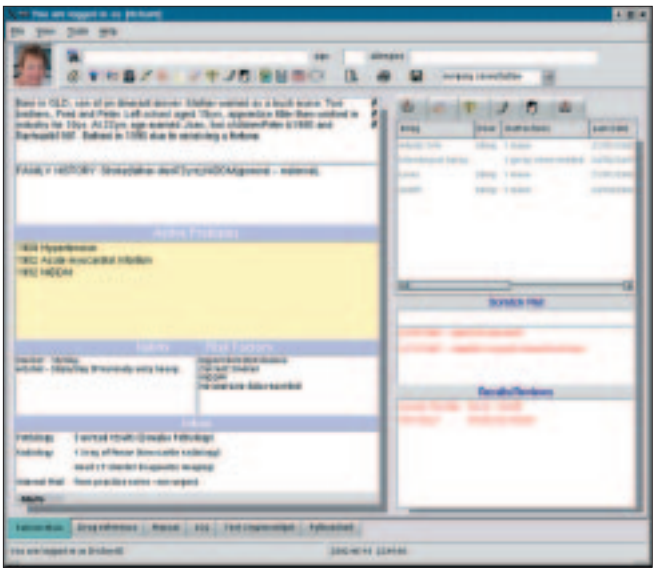

Figure 2: Gnumed summary screen

dangers of proprietary software in the health sector took place in March 2000. Physicians refused to base their decisions on non-transparent algorithms. Within this discussion Hors Herb was accused of unconstructive criticism, which he took as a trigger to start working on Gnumed.

After a first working alpha-release was presented at the MedInfo2001 in London, the international interest in Gnumed made an total redesign of the internal structure necessary. Implementing this new structure is currently the main task for the project co-ordinations Horst Herb and Karsten Hilbert, who work on this together with about 17 other developers and many volunteers.

After completing a minimal version, which they hope to already be useful, it is planned to make Gnumed a complete medical solution which should include decision support.

The problems that Gnumed faces on the way to this is a lack of free pharmaceutical databases, different health systems with different regulations, lack of data formats, transfer standards and standardised messaging protocols, as well as lack of a system to create a globally unique ID for a patient.

Programming languages used in this project are Python and C/C++ on the client side, PgSql, C and Python on the server side, with reliability and security being the most important paradigms; both of which are not adequately treated in proprietary solutions in the opinion of the Gnumed team. In the Gnumed team there are many physicians from many different fields, who know what they want, but often not how to implement it.

Therefore some more experienced developers would be a very welcome addition to the Gnumed team.

Gnumed, which seeks to have an easy, ergonomic and highly configurable GUI, support for different languages and health systems, as well as relative platform independence in the end, is published as Free Software under the GNU General Public License.

If you wish to get active in this sector, Gnumed is surely a project to help with.

OIO

The "Open Infrastructure for Outcomes" (OIO) [9] is called the "Search for the holy grail" of data portability by its author, Andrew Ho. Nandalal Gunaratne, Alesander Chelnokov and others accompany him on this quest.

OIO was used for production at the Harbor-UCLA Medical Center in March 2001 before being published as Free Software under the GNU General Public License in August. By September it was managing data of more than 1000 patients and since February 2002, it is being used as a hospital-wide information system. So it is safe to say that OIO has proven itself already in daily use.

The primary components of OIO include the server, which is accessed via any browser through HTML and the OIO library. The server is a flexible, web-based data management system, which manages users, patients and information about them; although it would of course also be possible to use it for invoices, deliveries or accounts.

The OIO-library is a metadata repository, which allows exchanging metadata like plug-and-play web forms or project descriptions between server and client.

An OIO user can create or modify forms through a web browser, which is then immediately available to be used for data collection over the web.

Later forms can be exported as XML-data to be transferred into a metadata repository like the OIO library or uploaded to another OIO server.

Of course it is also possible to assemble data from different forms into a single dataset that can then be searched/queried over the web with help of logical operations.

Although OIO has been used for some time, development is not complete. Among the planned features for future releases is support for wireless PDAs. Plug-and-play protocols will also be supported. Most helpful at the moment would be more users, more feedback and better packaging.

At least with the last point Debian-Med should be able to help.

Res Medicinae

Res Medicinae [10] by Christian Heller is also used by the Debian-Med project. Together with Karsten Hilbert he works on making Res Medicinae an extensive software solution in the medical area.

To achieve maximum portability, Res Medicinae is based on Java (API/Swing, Servlets/JSP, JDBC) with some

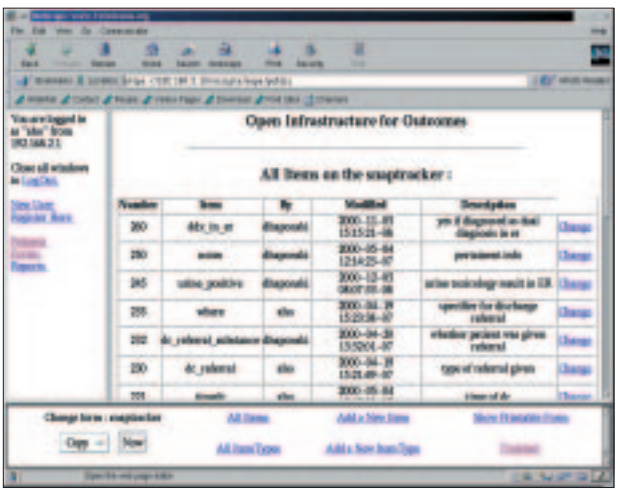

Figure 3: Viewing a complete form

CORBA/IDL and SOAP/XML. This already shows the largest problem of this Free Software project under the GNU General Public License and GNU Free Documentation License, because lacking a full-featured Free Software Java implementation, the freedom of the project is in danger.

But freedom was a major motivational factor for Christian to begin working on Res Medicinae. He wants to overcome the very expensive and proprietary scene of medical information systems and give users in less privileged countries access to a free, stable, secure, platform independent and extensive system.

The project is still rather young. According to the plans, at the end of 2002, the ResMedLib framework should be consolidated and prototypes for two complete modules should be available. In 2003, the administrative module, printing forms and generating reports should work.

Afterwards, an image processing and a management tool as well as a billing and statistical module should be added. A training module as well as a decision support module will then finish the whole project.

So you should probably not try to use the project in your daily life, but those who are interested to bring medical competence, language translations, Java programming or webpage design into the project, will receive a warm welcome in Res Medicinae.

As far as the authors know, Res Medicinae is currently the only Java based GPL project in the medical area and they plan to work together with OpenEMed, a similar Java project under a BSD license and the already mentioned Gnumed project to achieve full project interoperability.

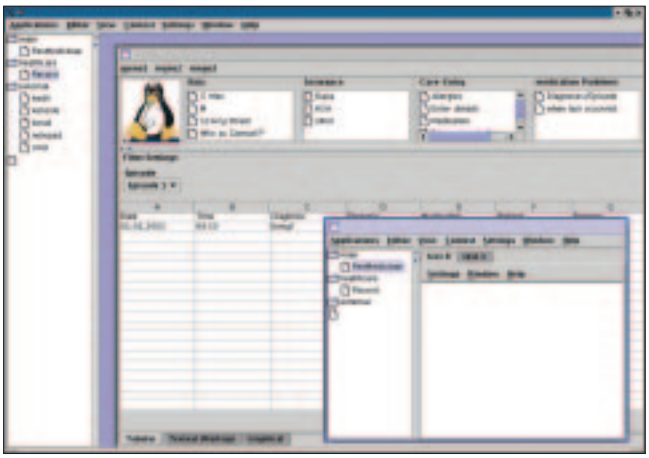

Figure 4: Res Medicinae forms aid open source doctors

Figure 5: JavaRisk preparing to battle

That should be enough health for today, if there are other projects in this area, I would like to present them in a later issue. An email [1] would be the appropriate way to get this going.

Romance

Romance [11] is the attempt by Bertrand Lamy and Jean-Baptiste Lamy, to give Free Software a real, Free alternative to Microsoft's .NET.

According to Bertrand, their motivation is that Ximian will not be able to deliver a Free implementation of .NET. That Microsoft has already promised to fight all Free alternatives using software patents does indeed make this a plausible scenario.

Also standards controlled by those companies without any Free reference implementation always have the advantage that the company is several steps ahead, while the Free projects have many problems following. The situation around Java suffers from this effect.

The answer is clear: We need a Free standard with a Free implementation. This is what Romance seeks to provide.

The first part – and beginning of development – is Rose, the "Romance Object System rosE." Rose provides a protocol, which allows for the sharing of objects between the different programming languages.

The next step of development will be WiSe, the "Romance Widget Server." It will be available as a GUI/toolkit library to all Romance applications through the Romance server.

The paradigm employed in WiSe is that all widgets remain the property of the WiSe process, and not of the different applications. That should allow Romance to make sharing of widgets very fast and simple.

Since Bertrand and Jean-Baptiste believe that 75% of all desktop applications should be written in script languages, they have concentrated on supporting Python, Guile and C first. According to their plans, Rose will also support Perl, Ruby, Lisp, Scheme and other dynamic languages in the future.

There are many examples how Romance can be used.

For large applications, it is often a good idea to define an expansion language. Instead of choosing one

language, a group of objects could be made available through Romance, which allows scripting the application in any language supported by Romance.

Networked applications often require a communication protocol, which can be supplied by Romance. Since it is lighter than CORBA, supports more languages than Java RMI and works with dynamic, non-static objects, Romance offers several advantages in this regard.

Romance can also act as "glue" between different parts of a program written in different programming languages, providing an alternative to SWIG. Being dynamic, Romance automatically makes sure that the interface is available in all of the "Romantic" languages.

Through WiSE, Romance also provides widgets for a graphical user interface, which can be shared in a lean and efficient manner between different processes. This makes linking against graphical toolkits unnecessary and allows the user to choose the look & feel of all applications through Romance.

While offering all these possibilities, Rose is very small, it has less than 500 lines of code in Python/Scheme.

Although this project, which is released under the GNU General Public License, is most relevant to developers, it should also give non-developers some interesting perspectives for the future.

JavaRisk

As an add-on to issue #39, [12] in which some Risk clones were presented, I would like to introduce JavaRisk [13] by Christian Domsch, Sebastian Kirsch and Andreas Habel.

JavaRisk is an implementation of the well-known board game Risk in Java under the GNU General Public with the rules based entirely on the German version of the game. Despite JavaRisk being a computer game, the authors did not implement any network support or artificial intelligence.

It has outstanding graphics, though, which are so good that the game J-TEG introduced in issue #39 implemented them as a theme.

JavaRisk is a typical student project, which means that the three authors did not stop playing while their professor was around.

Figure 6: Using XPM chess pieces from Xboard. The tarball includes 30 different sets

When he noticed the game, he immediately loved it. He suggested that they should write an artificial intelligence for the game as their 5th semester student project. Also, since he is a fan of everything Asian, he asked them to implement small animated Samurai-fighters to be displayed whenever China or Japan are being attacked.

Currently Christian, Sebastian and Andreas are working on a new version of JavaRisk, which will have more resemblance with a strategic war game like Empire. JavaRisk v2 will support networked play right from the start.

EmacsChess

Also in reaction to issue #39, Mario Lang wrote to me and recommended writing about the Emacs Chess [14] project by John Wiegley.

Emacs Chess consists of three major parts. The first part contains the display/front-end capabilities in order to display different types of boards in Emacs. The second part allows communication with different chess engines like GNU Chess and Crafty. The third part is a library for positions and games including a validity checker for moves and game-database management.

Among the very neat features of Emacs Chess is that the Emacs IRC Client (ERC) [15] supports Emacs Chess, so it is possible to start a game with somebody in IRC if that person also uses Emacs Chess and ERC.

Since the IRC chess protocol is based on CTCP, it is possible to implement a compatible functionality in other clients.

As Emacs can also run on a console, Emacs Chess also provides a nice Chess front-end for vision-impaired users, who can have moves announced in a "knight takes a4" form. Of course non vision-impaired people may also use this very neat feature.

People not using Emacs will probably not start doing so because of Emacs Chess, but it should truly be a worth a try for all of Emacs friends.

Emacs Chess is written in Emacs-Lisp and published under the GNU General Public License as betatest software, so you may expect some minor bugs.

End

Enough Brave GNU World for this month. Ideas, suggestions and comments for interesting projects are as always welcome to the usual address. ■

INFO

[1] Send ideas, comments and questions to Brave GNU World: *column@brave-gnu-world.org*

[2] Home page of the GNU Project: *http://www.gnu.org/*

[3] Home page of Georg's Brave GNU World: *http://brave-gnu-world.org*

[4] "We run GNU" initiative: *http://www.gnu.org/brave-gnu-world/rungnu/rungnu.en.html*

[5] Debian-Med home page: *http://www.debian.org/devel/debian-med/index.en.html*

[6] Brave GNU World issue #23: *http://brave-gnu-world.org/issue-23.en.html*

[7] Debian-Jr. home page: *http://www.debian.org/devel/debian-jr/index.de.html*

[8] Gnumed home page: *http://www.gnumed.org/*

[9] "Open Infrastructure for Outcomes" home page: *http://www.txoutcome.org/*

[10] Res Medicinae home page: *http://resmedicinae.sourceforge.net*

[11] Romance home page: *http://savannah.gnu.org/projects/romance/*

[12] Brave GNU World issue #39: *http://brave-gnu-world.org/issue-39.en.html*

[13] JavaRisk home page: *http://sourceforge.net/projects/javarisk*

[14] Emacs Chess home page: *http://emacs-chess.sourceforge.net*

[15] Emacs IRC Client home page: *http://sourceforge.net/projects/erc/*