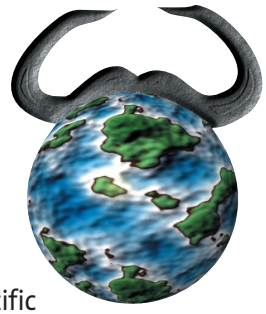
The monthly GNU column

Brave GNU World



Welcome to another issue of the Brave GNU World. This month we concentrate on some important scientific projects and free audio environment for interactive multimedia applications

BY GEORG C.F. GREVE

The first feature for this month is the Skidbladnir [1] project, the name of which might be as mysterious as its description to some people.

Skidbladnir

The author, Lars Brand, describes it as a "toolbox with information and programs for Computer Aided Innovation."

Computer Aided Innovation may not be very well known, so we are lucky that that Lars went to the extra effort to give some background information.

The scientific background of Skidbladnir is known as the "Theory of Inventive Problem Solving", in English abbreviated as TIPS, in German and Russian known as TRIZ. The theory has its origins in 1946 with Prof. Altschuller, who at the time was a patent engineer for the army.

When Prof. Altschuller sent a letter to Stalin, informing him about the beginnings of his theory, he immediately found himself imprisoned and sent to a camp for suspicious subjects. During his imprisonment he met numerous professors from different scientific areas, who helped him combine knowledge of many different disciplines.

Released from the camp and again having become one of the undesirables around the end of the 60s until the mid 70s, he began publishing his results as Sci-Fi novels in order to raise funds for his scientific work as Genrikh Altov. The TIPS theory is based upon the principle that independent of scientific discipline or industrial area, abstracted problems and their solutions usually repeat themselves. When analyzing about 40.000 extraordinarily successful patents, it was found that all of these were based on just 40 different solutions.

Another principle is that the evolution of technical systems follows certain tendencies and that essential innovation often requires an influx of scientific results from another area.

What this describes is an abstraction of problem-solving strategies; an often used example is "A massive steel cube of 1m edge length is to be moved into a deep cavity without using cranes, ropes or similar tools. It also must not be thrown. Develop 3 appropriate solutions within 10 minutes."

The most well-known project to solve this and other problems with software is probably the TechOptimizer, which, like all the other applications in this field, is proprietary and very expensive.

Skidbladnir now seeks to provide this functionality as Free Software. Besides the installation components, the project consists of andax.php, which contains the basic principle to resolve technical contradictions, project sporadikus for web-brainstorming and perplexus.php, in which over 250 effects are available.

The project isn't complete and is not very comfortable to use yet. The effects database is not as large as Lars would like it to be. In particular, the networking of effects, which allows for efficient combinations, should be expanded.

The following provides a very simple example for this:

- (a) luminescent material converts UV light into visible light;
- (b) fine ground/spread metal inhibits luminescence;
- (c) before a moving part of an engine fails, small bits of metal are released into its oil. Combining these three facts easily leads to the idea that adding luminescent material to oil will allow determination of when certain parts of machinery need to be replaced before they fail, because the luminescence in the oil will stop.

Real situations are often much more complex and require a large database of

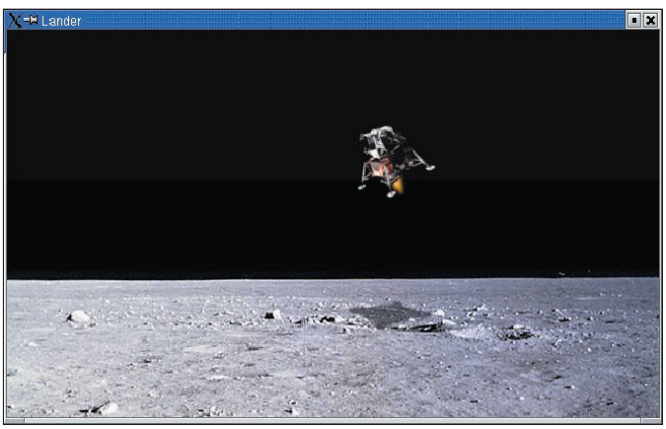


Figure 1:The Moon-lander game written in Lush

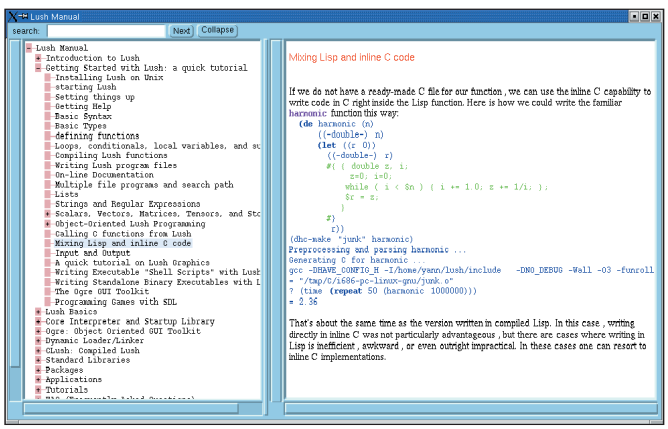


Figure 2:Lush showing problem free Lisp and C mixed within a program

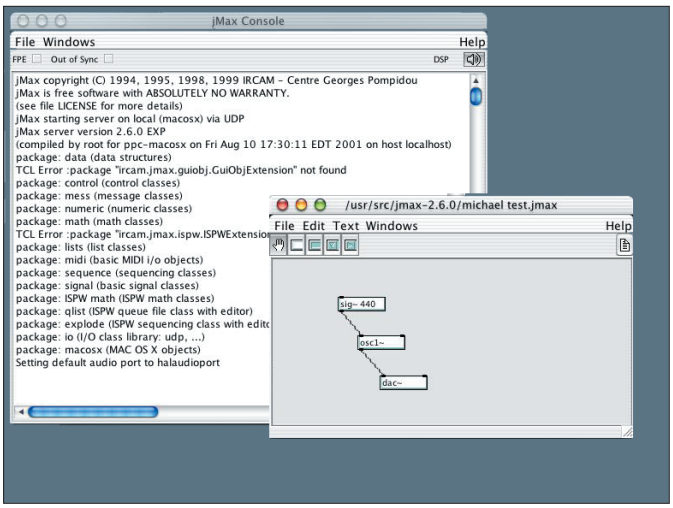


Figure 3:The music program jMax is platform independent

effects that has to be assembled from technical literature. This is a tedious and work-intensive task.

Skidbladnir was written in Perl, PHP and MySQL and it is published under the GNU General Public License (GPL). Compared to proprietary projects, Skidbladnir may have comparatively few effects, but it already contains software effects; possibly the first of its kind.

Help is very welcome in any form, developers especially, as well as users who are willing and able to give much needed feedback. Also more data about effects and access to real-life problems would be appreciated.

In the mid to long-term, Lars is convinced that Free Software will be extraordinarily successful in this area, as both TIPS and Free Software are based upon the idea of preserving knowledge and making it accessible.

Making these insights and methods available to all people seems like a very important project not much unlike an encyclopedia, worthy of support.

As a side-effect it also reduces the market-entry barrier in this area, since many potential users cannot afford to use the proprietary solutions, so it could help revitalizing the technical area.

Anyone interested and people from this area are encouraged to take a look. More information about the background are also available on the web. [2]

Lush

The second project of this issue is also scientifically oriented. Lush [3] is an object-oriented programming language for scientists, experimenters and engineers in need of larger numerical and graphical applications.

The design of Lush seeks to combine the strengths of 3 different programming approaches into one. The first is an interpreted, dynamic LISP-like language with automatic garbage collection and weak typing. The second language is a compiled, lexical language using the same syntax, but strong typing. The third language is C, which can be mixed with the Lush syntax within a program or even a single function.

Lush has been developed since 1987. Originally it went by the name "SN" as a script language for simulation of neural networks. Over time if has developed into a complete programming language with compiler. The main actors in the development of Lush were Bell Labs (later "AT&T Labs"), in Holmdel, NJ, USA, Neuristique S.A. in Paris, France and NEC Labs in Princeton, NJ, USA.

After Lush/SN had been used for years within AT&T for their internal research and development projects, the parties involved eventually agreed upon the GNU General Public License (GPL) and relicensed Lush as Free Software.

Today the project is maintained by Yann LeCun , who answered Brave GNU World's questions, and Leon Bottou of the "NEC Labs America" in Princeton, who receive support by a large number of volunteers from all over the world, such as Fu Jie Hang, Patrice Simard, Patrick Haffner, Yoshua Bengio, Pascal Vincent, Jean Bourrelly, Xavier Drancourt or Secil Ugurel, to name a few. More volunteers are always welcome.

Primarily developed as a Free Software Matlab-replacement, Lush offers a complete all-purpose language. With easy integration of C, Lush is a very good choice for scripting or integration in order to assemble distributed functionality in a comfortable GUI application.

This integration also makes integrating existing libraries quite easy, which is why Lush has bindings to numerous scientific, graphical and audio-visual libraries, like the GNU Scientific Library (GSL; see issue #35 [4]), OpenGL / GLU / GLUT, ALSA, Video4Linux or the Intel Vision Library.

When comparing the levels of support for the GSL between Python and Lush, Lush looks pretty good with about 4000 supported functions as opposed to the few hundred supported by Python. The syntax is cleaner than Perl and should be easier to learn than Scheme. When comparing speed with Octave or Matlab, it is between 15 and 300 times faster, depending on the situation.

These advantages seem to make Lush an interesting choice and definitely worth a glance. Lush has even been used for games, like a simple moon-lander, seen in Figure 1.

Lush was written in C and it traditionally runs on GNU/Linux, Solaris, Irix and OpenBSD, although since February 2003 a Cygwin Windows port is available.

One of the problems of the project is that the compiler design is more than 10 years old and therefore has some strange and rigid limitations. Rewriting the compiler is therefore on the ToDo list.

Also on the list are adding support for more libraries, as well as improving the documentation system. Adding a template mechanism to Lush is also planned in for the mid-term.

Additionally, there is a lot of interest in a Mac OS-X port, for which volunteers are sought. An automated parser for C/C++ header files for automagical inclusion in Lush would also be a quite useful project.

Should you live in the United States, you run a good chance of already having been in indirect contact with Lush. Some ATMs by NCR use Lush-generated code on embedded DSP processors to automatically read the amounts on deposited checks. And a high-speed check reading engine written in Lush reads about 10% of all checks deposited in the USA.

jMax

As regular readers of the Brave GNU World should know, the Free Software Foundation Europe is a partner in the AGNULA project, [5] which aims at

putting together an entirely Free Software GNU/Linux distribution for professional audio users.

Another partner of the project is the "Institut de Recherche et Coordination Acoustique/Musique" (IRCAM), the center of music of the Centre Pompidou in Paris, France. One of the applications written by IRCAM is jMax [6], a graphical development environment for interactive multimedia applications.

Audio applications traditionally had the problem that they were often written for specific hardware and therefore very platform dependent. Because of the rapid development in hardware, programs had to be rewritten every three years for a new platform, otherwise the music written for these programs was in danger of being lost.

This motivated the development of a pure software solution that would not depend on a specific platform.

The paradigm employed in jMax allows creating and/or combining certain basic elements like frequency generators, signal filters, effects, input- and output-modules, sliders, DSPs and amplifiers with each other and assembling them to so-called "patches."

These patches integrate their components and can be combined into almost infinitely complex constructs, making it generally possible to implement any kind and type of digital signal processing, effect or synthesizer.

One implementation of this paradigm that is rather well-known among musicians is the proprietary "Max." In 1995 jMax started out with the intention of creating a platform independent version of Max. In mid-1999 it was then released as Free Software under the GNU General Public License (GPL).

The IRCAM jMax team working on the project consists mainly of François Déchelle and Patrice Tisserand. François, who also filled out the Brave GNU World standard questions, sees the main advantages of jMax in its platform independence – it runs on GNU/Linux, Mac OS X and Windows, and the higher flexibility when compared with other implementations of the paradigm, such as Max or PD.

One of the key advantages is also that jMax consists of two components. The central component is a server, a real-time engine written in C, which does all the work. This allows running the engine with GUI, writing alternative GUIs or integrating the engine into a plugin environment (LADSPA).

Normally, this server is controlled via a client written in Java. Java was chosen so it would run on as many platforms as possible with the minimum amount of problems. Unfortunately the situation of Java is not without problems with reference to Free Software.

Java dependencies

The problem of Java is neither its technical specification not its implementation. Although some people have different opinions about them, they are not the cause of the problem for Free Software.

The cause of the problem is how Java itself is developed and distributed, since there are essentially only two widespread implementations, both of which are proprietary: one is maintained by Sun, the other by IBM. Although these may be distributed without licensing cost, they do not offer the freedoms necessary to make them Free Software.

In consequence every application running on these platforms – even software that is under a Free Software license – is putting the freedom of the user at risk. A situation similar to running Free Software on a Windows system.

There are some approaches and initiatives to implement Java entirely in Free Software (see "GNU and the Java language" [7]). However, since the dominant reference implementations are proprietary, the free projects always need to re-implement the features the proprietary versions have come out with.

Not every developer likes participating in such a biased race that cannot be won. Free Software is put at a disadvantage and therefore offers a smaller degree of functionality.

When developers of Java applications make use of the more advanced features of proprietary Java implementations, these can usually not be run on Free Software Java implementations anymore and in consequence are dependent on the proprietary platforms.

This is precisely the problem of the jMax client. Since adding proprietary software into AGNULA is out of question for all partners, AGNULA may not be capable of including jMax with a fully functional GUI.

pyMax

After none of the alternatives seemed very likely to resolve the problem in time – more information is available on the FSF Europe home page [8] – it was now decided to do without Java entirely and re-implement the client in Python.

The choice for Python was influenced by its platform independence and that it allows rapid development while (naturally) being entirely Free Software.

It is not clear whether IRCAM will be capable of finishing that client in time, though. They are looking for volunteers that can help them write the client.

According to François, IRCAM cannot make large promises, but they offer to provide priority support to people working on the Python client and guarantee a response time of 24hrs during working days. So if you are interested in this, you could take a look at the jMax developers mailing list. [9]

Enough for now

Should you come across an interesting project, please let me know. Often it is the readers of the column who find the gems – the Lush feature was triggered by a tip by Stefan Kamphausen, the author of the Brave GNU World logo. ■

INFO

[1] Skidbladnir home page: *http://mitglied.lycos.de/altow/*

[2] TIPS/TRIZ: *http://www.triz-journal.com/*

[3] Lush home page: *http://lush.sf.net*

[4] Linux Magazine, Issue 18, March 2002, Brave GNU World

[5] AGNULA home page: *http://www.agnula.org*

[6] jMax home page: *http://www.ircam.fr/jmax/*

[7] GNU and Java home page: *http://www.gnu.org/software/java/*

[8] AGNULA Java issues: *http://fsfeurope.org/projects/agnula/java.html*

[9] jMax developer mailing list: *http://listes.ircam.fr/wws/info/jmax*

[10] Home page of Georg's Brave GNU World: *http://brave-gnu-world.org. Send ideas, comments and questions to Brave GNU World: column@brave-gnu-world.org*