Liquid War 6

A unique multiplayer wargame
Edition 20060101, for Liquid War 6 Version 20060101
1 January 2006

by Christian Mauduit <ufoot@ufoot.org>

Liquid War 6, a unique multiplayer wargame.

Table of Contents

1 Introduction 1

2 Getting the game 3
- 2.1 Download 3
- 2.2 GNU Arch repository 3
  - 2.2.1 About GNU Arch 3
  - 2.2.2 Accessing the repository 3

3 Installation 5
- 3.1 Dependencies 5
- 3.2 ./configure 5
- 3.3 3-D hardware and drivers 5
  - 3.3.1 Why is 3-D acceleration required? 5
  - 3.3.2 Choose your hardware carefully! 6
  - 3.3.3 List of working hardware 6
    - 3.3.3.1 Matrox G450 6
    - 3.3.3.2 ATI Radeon Mobility 9000 6

4 Mailing lists 9
- 4.1 General discussion 9
- 4.2 Announcements 9
- 4.3 Reporting bugs 9

5 Savannah 11
- 5.1 A central point for development 11
- 5.2 Bugs 11
- 5.3 News 11

6 Project status 13
- 6.1 As of today, what does Liquid War 6 do? 13
- 6.2 .plan 13
  - 6.2.1 So what's up? 13
  - 6.2.2 Complete rewrite 13
  - 6.2.3 Technologies 13
    - 6.2.3.1 Script + standard C + assembly 13
    - 6.2.3.2 OpenGL 14
    - 6.2.3.3 CSound 14
  - 6.2.4 Functionnalities 14
    - 6.2.4.1 Visual enhancements 15
    - 6.2.4.2 Rules enhancements 15
    - 6.2.4.3 Hey, you forgot my idea!!! 16
  - 6.2.5 Road map 16

7 How you can help 19
7.1 Help GNU 19
7.2 What is needed? 19

8 Designing new levels 21
8.1 Introduction 21
8.2 Format description 21
8.2.1 Basics 21
8.2.2 Elements reference 21
8.2.2.1 map.png 21
8.2.2.2 elevation.png 21
8.2.2.3 texture.png 22
8.2.2.4 background.png 22
8.2.2.5 foreground.png 22
8.2.2.6 settings.xml 22
8.2.2.7 README 22
8.3 Migrating levels from Liquid War 5 22
8.4 Legal issues 22
8.4.1 Why data must be free 22
8.4.2 Common pitfalls 22

9 Hacking the game 23
9.1 Architecture 23
9.1.1 Cross-platform 23
9.1.2 OpenGL 23
9.1.3 Guile 23
9.1.4 Internal libraries 23
9.1.4.1 Program splitted in libraries 24
9.1.4.2 Private and public interfaces 24
9.1.4.3 Linking 24
9.2 Coding guidelines 24
9.2.1 Project goals 25
9.2.2 Common sense 25
9.2.3 Application specific issues 25
9.2.3.1 Unitary tests 25
9.2.3.2 Memory allocation 25
9.3 Modules 26
9.3.1 libliquidwar6sys 26
9.3.1.1 Log API 26
9.3.1.2 Memory allocation 26
9.3.1.3 File utilities 27
9.3.1.4 String utilities 27
9.3.1.5 Chained lists 27
9.3.1.6 Associative arrays 27
9.3.1.7 i18n support 28
9.3.1.8 XML tools 28
9.3.2 libliquidwar6cfg 28

9.3.3 libliquidwar6gfx 28
9.3.4 libliquidwar6map 29
9.3.5 libliquidwar6ker 29
9.3.6 libliquidwar6snd 29
9.3.7 libliquidwar6net 29
9.3.8 liquidwar6script 29

Appendix A GNU GENERAL PUBLIC LICENSE 31

Preamble 31
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION 32
Appendix: How to Apply These Terms to Your New Programs 36

Appendix B Copying this manual 37

B.1 GNU Free Documentation License 37
B.1.1 ADDENDUM: How to use this License for your documents 43

1 Introduction

Liquid War 6 is a unique multiplayer wargame. Your army is a blob of liquid and you have to try and eat your opponents. Rules are very simple yet original, they have been invented by Thomas Colcombet. It is possible to play alone against the computer but the game is really designed to be played with friends, on a single computer, on a LAN, or on Internet.

An older version: Liquid War 5, is available, but is not part of the GNU Project. Only Liquid War 6 is part of the GNU Project, it is a complete rewrite.

2 Getting the game

2.1 Download

Liquid War 6 can be found on http://download.savannah.gnu.org/releases/liquidwar6/.

However, it is still in alpha stage, so it is not playable yet.

These files might also be mirrored, the list of all the availables downloads is accessible on http://www.ufoot.org/liquidwar/v6/download.

Check out the MD5 checksums and GnuPG signatures to verify the integrity and authenticity of the files you download.

2.2 GNU Arch repository

2.2.1 About GNU Arch

There is no CVS repository for Liquid War 6. Instead, a GNU Arch repository is used to follow the different versions. Read the GNU Arch tutorial to learn how Arch works. Note that there are many other source control managers available, some of which provide functionnalities similar to GNU Arch / tla. GNU Arch has been chosen for Liquid War 6 because:

- it is Free Software,
- it is not limited to per-file commits like CVS, and supports atomic commits involving several files,
- it is distributed,
- it enables developpers to sign each of their contributions.

2.2.2 Accessing the repository

The repository for Liquid War 6 is accessible on http://arch.sv.gnu.org/archives/liquidwar6. This is a read-only access, but with the distributed nature of GNU Arch, it still allows you to keep track of your own changes, and submit patches. Accessing it in read/write mode with sftp requires a Savannah account and special rights on the Liquid War 6 project.

Here are typicall commands one can use to get Liquid War 6 source from the GNU Arch repository:

```
tla register-archive http://arch.sv.gnu.org/archives/liquidwar6
tla get -A liquidwar6@sv.gnu.org liquidwar6--alpha
```

All the patches in the archive are signed with GnuPG, so you can check their authenticity with my public key.

You might need to edit your `$HOME/.arch-params/signing/=default.check` file and put the following text in it:

```
tla-gpg-check gpg_command="gpg --verify-files -"
```

3 Installation

3.1 Dependencies

Before installing Liquid War 6, you'll need to install the following:

1. Mesa. This library provides an API similar to OpenGL and enables 2-D and 3-D drawing.
2. zlib. Required by other libraries, but can also be used directly by Liquid War 6 to compress network messages for instance.
3. libpng. Used by SDL_image. Liquid War 6 also uses libpng to read levels (maps), without using the SDL_image layer.
4. SDL. SDL is used to set up a working OpenGL environnement, and handle input (mouse and keyboard).
5. SDL_image. This SDL extension is used to read textures and other graphics from disk.
6. SDL_ttf. This SDL extension is used to draw fonts. It is UTF-8 enabled.
7. Guile. Possibly the most required library, since Liquid War 6 is a scheme program which uses a set of functions coded in standard C.
8. expat. Used to read and write XML files, which contain constants and configuration data.

3.2 ./configure

Once all these libraries are installed, use the standard sequence:

```
./configure
make
make install
```

Liquid War 6 makes a heavy use of Automake and Autoconf.

3.3 3-D hardware and drivers

3.3.1 Why is 3-D acceleration required?

Liquid War 6 requires Mesa, a 3-D library which offers a high level API similar to OpenGL. While Liquid War 6 will compile and run with theorically any Mesa configuration, it might not run fast enough with software rendering only. The frame rate will probably be well below 10 fps, which is not acceptable.

The reason for it being so slow is that the OpenGL API was not really designed for software rendering. It is possible to achieve better performance in 2-D and 3-D graphics in software mode, but not using a high-level API like OpenGL. You need to get your hands in the dirt. On the other side, using Mesa is very comfortable, and when it is configured to use the 3-D features of the hardware, it runs faster than any pure software solution.

Therefore Liquid War 6 requires some sort of hardware acceleration. And it will probably do for some time. The reason for using a 3-D library to provide acceleration is that fundamentaly, most recent hardware provide acceleration functions through their 3-D interfaces, and barely anything is done on the pure 2-D side.

The good news is that while Liquid War 6 requires acceleration, it does not require any fast, recent or expensive video card. It can reasonnably be run on video cards which date from the last century. If not from the last millenium.

3.3.2 Choose your hardware carefully!

It is important, if you want to run Liquid War 6 in the best conditions, to have a hardware which provides acceleration, and for which a free (as in speech) driver is available. As GNU/Linux is the platform Liquid War 6 is being developped on, it is also the platform it is being designed for.

In a general manner, knowing which hardware devices support GNU/Linux is important not only for practical reasons—you want your hardware to work with the software that you want to use—but also for ethical and political reasons. You can help the free software movement by purchasing hardware from manufacturers who support our goals and not purchasing from those who don't.

See the list of hardware devices that support GNU/Linux for more informations.

Keep in mind that owning a hardware which has no driver for it will make Liquid War 6 and possibly any 3-D game simply unplayable.

3.3.3 List of working hardware

The following list describes some hardware configurations which are capable of running Liquid War 6 in correct conditions. This list obviously does not pretend to be perfect or exhaustive. If you know any other hardware capable of sufficient 3-D acceleration (fast enough to run Liquid War 6) with a free (as in speech) driver, please inform the maintainer.

3.3.3.1 Matrox G450

This card is fully supported under GNU/Linux.

You need to load the '`mga`' driver into the Linux kernel, and the XFree86 configuration file should contain a section like:

```
Section "Device"
        Identifier      "G450"
        Driver          "mga"
        Option          "AGPMode" "2"
        VideoRam        32768
EndSection
```

3.3.3.2 ATI Radeon Mobility 9000

This card is supported under GNU/Linux. It is typically shipped with notebooks. Make sure to use the free (as in speech) driver. It exists and works.

Note that the ATI Radeon 9200 is the latest ATI card to be fully supported by a free driver. There is an effort to develop a free driver for recent ATI cards, but this is done through reverse engineering, for ATI do not give the specs of these recent cards. It is therefore extremely hard to write drivers for this hardware.

You need to load the '`radeon`' driver into the Linux kernel, and the XFree86 configuration file should contain a section like:

```
Section "Device"
        Identifier      "ATI9000"
        Driver          "ati"
EndSection
```

4 Mailing lists

4.1 General discussion

The main discussion list is <help-liquidwar6@gnu.org>, and is used to discuss all aspects of Liquid War 6, including installation, development, game strategies, and whatever subject players and hackers might want to talk about, provided it is Liquid War 6 related.

To subscribe to it, please send an empty mail with a Subject: header line of just "subscribe" to the `-request` list, that is <help-liquidwar6-request@gnu.org>.

You can also subscribe to the list and view its archives using the Mailman web interface for liquidwar6-help.

4.2 Announcements

Announcements about LiquidWar 6 are made on <info-liquidwar6@gnu.org>. Subscribe to it to be informed of major releases, and other significant news.

To subscribe to it, please send an empty mail with a Subject: header line of just "subscribe" to the `-request` list, that is <info-liquidwar6-request@gnu.org>.

You can also subscribe to the list and view its archives using the Mailman web interface for liquidwar6-info.

Please also consider reading the latest news on Savannah.

4.3 Reporting bugs

There is also a special list used for reporting bugs, <bug-liquidwar6@gnu.org>. Please try and describe the bug as precisely as possible. The more accurate the description, the more chances it will get to be fixed.

It is also possible, and also more convenient, to directly report bugs on Savannah.

5 Savannah

5.1 A central point for development

Liquid War 6 uses the services provided by Savannah.

Starting from http://savannah.gnu.org/projects/liquidwar6/ you'll get links to various informations related to Liquid War 6 development.

5.2 Bugs

The list of Liquid War 6 bugs is accessible on: http://savannah.gnu.org/bugs/?group=liquidwar6. It should be updated on a regular basis and reflect the current status of outstanding bugs.

While this Savannah interface allows you to view the list of existing bugs, it also allows you to report new bugs. Of course bugs can still be reported using the bug mailing list, but any such reported bug will end up as an entry on Savannah's bugtracker. Therefore it saves time to submit the bug on Savannah directly.

5.3 News

Latest Liquid War 6 news should be available on: http://savannah.gnu.org/news/?group=liquidwar6.

The news mailing list is another way to get the latest news, however Savannah has the advantage to offer a dedicated interface.

6 Project status

6.1 As of today, what does Liquid War 6 do?

Currently, there's nothing playable. However all the technical framework has been set up. The program already uses both Guile and OpenGL, has i18n support through gettext, and so on. It is capable of loading maps, store configuration options in a config file, and the menus are interactive.

For now it seems a very complicated piece of software for what it does, but the point is that the fundations are here, and now it's possible to build something solid.

If you want to play Liquid War right now, use Liquid War 5.

6.2 .plan

6.2.1 So what's up?

Here's my .plan file, which describes what I (Christian Mauduit) have planned for Liquid War 6. There's no garantee that what's written here is a precise description of the real future, however it should give a good idea of what I have in mind.

Note that the information here was written in summer 2005, it might or not be accurate now, as the main reason for plans to exist is that people never follow them. I'm no exception.

6.2.2 Complete rewrite

Liquid War 6 will be an almost complete rewrite. I mean that common code between branches 5 and 6 might end up in representing 0% of the total code. I think this is a wise decision, for the current code is really hard to maintain, and would not survive any serious cleanup. LW5 was first written in 1998, for DOS, when I had much less experience in programming. In 7 years I - and other people as well - hacked major enhancements in it such as cross-platform support, network games, and if you compare release 5.0 with the latest 5.x.x release, you'll see that a bunch of things have changed. I had never expected I would patch and fix this game for so long, and it's no surprise that it's bloated today.

FYI, here's a list of what makes LW5 unsuitable for major improvements without a complete rewrite:

- global variable hell. Lots of things are stored in globals.
- hard-coded C GUI. Read src/level.c to get an idea of how horrible it is.
- hard-coded 256 colors paletted mode. A clever bet in 1998 (performance...). Not anymore.
- generally bloated code. Makes bug-finding very tricky.

6.2.3 Technologies

Liquid War 6 will use a different technical framework than Liquid War 5.

6.2.3.1 Script + standard C + assembly

It happens that coding a large project in pure C is a waist of time, if possible at all.

If one applies the standard 80/20 rule to a computer game, one might state that 80% of the code eat up 20% of the CPU and the other 20% of the code eat up 80% of the CPU, the former being high-level glue code and the latter being low-level algorithmic code.

With Liquid War, one could speak of the 99/01 rule. I mean that 99% of the CPU time concerns only 1% of the code, and vice-versa. Basically, Liquid War has a very CPU-greedy core algorithm, still spends a fair amount of CPU displaying stuff (but this is delegated to the low-level game programming library) and the rest is totally unsignificant, in terms of CPU. Point is this "rest" represents the vast majority of the code, and also represents the very same buggy code I spend nights to patch on Liquid War 5. I'm talking about network code, GUI, and other high-level glue-code which are currently being written in C.

This idea is to write all this in a convenient scripting language. There won't be any impact on performances. I can't garantee Liquid War 6 will be blazingly fast, but for sure it won't be the scripting language fault. And of course if, as in Liquid War 3 and 5, I feel the need to implement some stuff in assembly for performances issues, I will do it.

We end up with a multi-language architecture: script + C + assembly.

My guess is that I'll use Scheme as an extension language. Python would be a good choice too. Let's say I'll give Scheme a chance, and if it's really not adapted, I'll switch back to Python. The point is that today I know Python and don't really know Scheme, but, well, it's always a pleasure for me to learn new things. It's fun.

So what is planned today is that Liquid War 6 will be a Scheme program, which will call callbacks functions written in C and/or assembly. These functions will do all the low-level time consuming algorithmic and graphical stuff. The rest of the code being entirely scripted.

6.2.3.2 OpenGL

Liquid War is not a 3D game, so why use OpenGL?

- it's a very convenient way to access video hardware acceleration with XFree86.
- low-end computers and/or computers without 3D acceleration can still run Liquid War 5.
- I'm interested in learning/using this API 8-)

This choice implies that I won't use Allegro anymore. Allegro stays a very convenient library and I would recommend it for it's excellent, easy to learn, powerfull, and stable. But for the needs of Liquid War 6 I'll use something else (because of OpenGL). I first thought of using GLUT but I might end up simply using SDL. The idea is just fo have an OpenGL wrapper which sets up OpenGL in a similar manner on all platforms, and handles basic things such as mouse or keyboard.

6.2.3.3 CSound

I've got two excellent books on Csound, and the will to learn how to use this tool.

I'll probably use Csound for a number of things, ranging from "bubbling sounds" to full blown music. Stay tuned 8-)

6.2.4 Functionnalities

6.2.4.1 Visual enhancements

Of course Liquid War 6 will look nicer than Liquid War 5, blah blah blah. What do you think?

Maybe I'll try to use some OpenGL features to make it possible to play on a ball, on a Moebius ring, or other fancy things. I have zillion of ideas, future will decide which ones will be implemented first.

To make it clear, visual enhancements aren't my top-level priority. However I'll try and make room for these enhancements, and prepare the terrain correctly. So it's possible that the first releases of Liquid War 6 won't be that much better than Liquid War 5, but at least Liquid War 6 will have the possibility to evolve. Something Liquid War 5 doesn't have.

6.2.4.2 Rules enhancements

There are many things that could be done easily:

- several cursors for one team
- alliances between teams
- deep places on a map, where more liquid can reside
- circular maps which "connect" the left border to the right one
- ...

As for graphical improvements, this is not my top-level priority. Simply, I'll make the game ready-to-improve. Again, all these enhancements are very hard to code in Liquid War 5, else I would already have coded them. Network enhancements

That's my top-level prioriry.

Why is that? Well, think of Liquid War in terms of "what makes it a good game?" and "what makes it a poor game?".

It's a good game because:

- the idea is original
- the gameplay is addictive
- you can play on a LAN
- all the family can play
- it's cross-platform
- it's Free Software

It's a poor game because:

- it's somewhat ugly and has a retro "back in the eighties" look
- network games are slow on Internet
- there are not enough active Internet servers

For the ugliness, well, OpenGL and some artwork should make it. But for the network, what's the real problem?

The real problem is that in the current situation, the server needs to have all "keystrokes" before doing anything, and all players must be connected before a game starts. Here's what I plan to do to fix this:

- players will be able to connect on a game "on the fly". This is done by most online games, and it's IMHO a required features for a network mode to work on Internet (not speaking of local networks, but real wide online gaming). How this will fit with Liquid War's rules is not totally decided, but I already know of several way to achieve this.
- I'll implement an "anticipation" system "a la" U61. This means that no matter if a remote player has a poor network connection, things will behave as if everything was fine. Internally, the system keeps 2 images of the game. One which is "anticipated" and displayed to the player, and one which is validated but outdated, kept internally. It's a little hard to explain, consumes twice as much CPU and memory, but it works. It happens that today the lacking ressource for playing Liquid War online is more on the network side than on the local CPU and memory aspects.
- I'll take it to the next level and implement a "peer-to-peer-like" network model, in which any client can become a server. The idea behind is that if a server quits the game, then a client takes its role, letting the game continue for hours. This way one could virtually have a never ending Liquid War game which would last weeks. I believe this could be really cool. I also believe no proprietary game will ever implement that, for in this model there's no way to force people to access a centralized server, this server usually being the major key in the business model of a company which sells proprietary software.

This third point will be the real enhancement of Liquid War with version 6. It's one of the very points which drives me to rewrite it completely. First because it's impossible to implement it without some heavy work. Then because I find it very motivating.

6.2.4.3 Hey, you forgot my idea!!!

Many gamers submitted suggestions, either by mail or by posting messages on the mailing list.

Don't worry, I keep them. Not reading them here does not mean I won't implement them. It simply means I won't implement them first. I first need the game basically function before enhancing it with fancy stuff.

6.2.5 Road map

As I stated on the mailing list, when thinking about Liquid War 6, think of years rather than months (unless I get fired, jobless, or spend several months in a hospital with a laptop).

Note that this road map takes it for granted that I'll be the lone coder on the project. It's unlikely that someone is going to help me for the first stages, until there's at least something real, something playable. Something that proves that the concept is valid. Besides, (real) team work implies a significant overhead, especially at project start. It's hard to figure out how to distribute tasks when the tasks themselves are not clearly identified. But for the rest (starting in 2007 or 2008), it's possible that external help might greatly... ...help!

- 2005 : Project framework should be done. This implies that the scripting engine is up and running, graphical mode works, config and data loading work, basic menus are available. Nothing playable.
- 2006 : Import the core algorithm from Liquid War 5, make the game playable in "demo mode" ("Ã la" Liquid War 2), implement the network "peer-to-peer-like" mode. At this stage, it will be possible to know wether Liquid War 6 is true vaporware or not.

- 2007 : glue all this together to make something usable by anyone, heavy work on the GUI, on the options, on error checking, many bug fixes. The goal is to have a game which is equivalent to Liquid War 5, with the network aspects pushed to the next level.
- 2008 : tadaaaaaaaaaaa! Release the game "publicly" - inform Freshmeat 8-) - and enhance it with all the feedback from gamers (bug reports and suggestions received since 1998). Work on artwork (both graphics and musics). Write documentation.
- 2009 : stabilize the game, patch it for all those things which had been forgotten back then in 2005, optimize for speed, bug-fix bug-fix bug-fix.
- 2010 : stop maintaining Liquid War 5, invite Liquid War fans and coders to a hudge party in my garden, sing all night, drink beers and wine, teach Liquid War strategies to my 5 and 6 year old daughters, remember the old times when Liquid War wasn't so cool 8-)

7 How you can help

7.1 Help GNU

Please remember that development of Liquid War 6 is a volunteer effort, and you can also contribute to its development. For information about contributing to the GNU Project, please read How to help GNU.

7.2 What is needed?

According to the current road map, I (Christian Mauduit) will need to work for some time on setting up the basis of the game. Basically the game must be at least playable in some demo mode for people to start hacking efficiently on it.

Meanwhile, there are always tasks which can save me a lot of time:

- testing, and making bug-reports, is always welcome;
- translating can also be done right now, since Liquid War 6 already has i18n support;
- artwork, including icons, musics, textures, good-looking color sets are something I'm pretty bad at doing myself, any help on this side is appreciated;
- level design, this has been one of the major contribution field with Liquid War 5, and should it be the case again with Liquid War 6, it would just be fine;
- OpenGL and Guile hacking, I currently have a rather superficial knowledge of OpenGL and Guile, so experienced programmers in those domains could possibly help the project a lot, maybe not by developping directly on it, but at least by sharing their knowledge.

8 Designing new levels

8.1 Introduction

As of Liquid War 5, most levels have been contributed by players. While the maintainer of Liquid War 6 has technical knowledge to develop the game, artistic talent and taste might not be his domain of excellence 8-)

Therefore contribution are truely welcomed when they take the form of a new, original, fun and good looking level.

Note that this manual might refer to levels and maps: they are just two different names to describe the very same thing. It's an alias.

8.2 Format description

8.2.1 Basics

Liquid War 6 stores level information in a plain directory.

There is no such thing as an opaque `.dat` binary file. The name of the level is the name of the directory itself, and its elements are the files contained in it.

Files must follow a precise naming scheme. For instance Liquid War 6 expects a `map.png` file to be present in each map directory.

All image files in a level use the Portable Network Graphics format.

It is possible that in the long term, Liquid War 6 will be able to handle levels as `.tar.gz` or `.zip` files. In that case these files will only be a compressed image of the actual level directory.

See the `./data/map/` directory of the source Liquid War 6 distribution to see example of maps.

8.2.2 Elements reference

8.2.2.1 map.png

This is the only required file in a level.

In fact, the existence of `map.png` makes a directory a level. When checking wether a directory is a correct level, Liquid War 6 simply tests the existence and validity of `map.png`.

This image is a simple black & white area, where white zones are the background, the sea, the places where fighters can move, and black zones are the foreground, the walls, the places where fighters can't go.

This informations can be stored in a 2-color indexed file, or in a grayscaled or even truecolor RGB file, but color information won't be used. Internally, Liquid War 6 will read the color of every point. If it is over 127 on a 0 to 255 scale, it will be considered as background, if it is below 127, it will be considered as foreground.

8.2.2.2 elevation.png

Todo...

8.2.2.3 texture.png

Todo...

8.2.2.4 background.png

Todo...

8.2.2.5 foreground.png

Todo...

8.2.2.6 settings.xml

Todo...

8.2.2.7 README

A README which describes the map. Should contain a short description, and copyright information.

It is a deliberate choice not to use specific fields to store these informations and use a global README instead. It makes both program code and map design simpler.

8.3 Migrating levels from Liquid War 5

Todo...

8.4 Legal issues

8.4.1 Why data must be free

Many Free Software games come with a free game engine, but without free data. A very good example of this is Doom. While the engine of this game is free, released under the GNU GPL, the data required by it is still proprietary. The Freedoom project addresses this issue, and aims at creating a complete Doom-based game which is Free Software. This requires time and energy, and it is very usefull since a Free Software game without free data to run with is not really usable.

All the data in Liquid War 6 are released under the GNU GPL, along with the source code. Data is considered as being part of the game, since running Liquid War 6 without any level makes no sense.

While the act of running Liquid War 6 is not restricted (see the complete terms of the GNU GPL), no non-free levels or graphics will be distributed with the game.

8.4.2 Common pitfalls

Here are some points you should think about before designing new maps:

- any reference to a trademark, a copyrighted logo, and in a general manner any non-free element makes your work non-free. If in doubt, create your levels from scratch, starting with a blank page;
- the fact that an image can be viewed with no fee on Internet does not mean it is free to use. Again, it is safer not to rely on someone else's work.

9 Hacking the game

9.1 Architecture

9.1.1 Cross-platform

Liquid War 6 is developped on GNU/Linux and other platforms are not supported yet. However there is no reason that would prevent it from running on other platforms, such as Microsoft Windows, Mac OS/X, FreeBSD and in a general manner any *NIX system.

Liquid War 6 uses libraries which are cross-platform (OpenGL, Guile...) so porting it is merely taking the time to fix makefiles and build tools for the given platform. This does not mean it is immediate, but it's a feasible task.

9.1.2 OpenGL

Liquid War 6 uses OpenGL to handle all the video rendering. It is a deliberate option to use this high-level API. The advantages are that it is cross-platform and uses the acceleration features of modern video hardware.

The major drawback is that without accelerated hardware, the game will still run using a software renderer such as Mesa, but it will definitely be too slow to be playable. The option for players who do not own such hardware, or who do not have a correct driver - many vendors keep their specifications secret, which forbids Free Software developpers to create free (as in speech) drivers for them - is to use Liquid War 5 instead, which does not require fancy hardware and accelerated drivers.

Liquid War 6 uses SDL to set up the OpenGL environment. Indeed, OpenGL itself does not handle keyboard or mouse input, and requires some platform dependant glue code to be initialized. SDL provides all this.

9.1.3 Guile

One of the lessons learned from Liquid War 5 is that standard C is very convenient for some tasks, but is plainly inadapted in many cases. Handling menus with a high-level API is one example of something really painfull to do with C, whereas it is child's play with a scripting language.

For this reason, Liquid War 6 is a C program which embeds a script interpreter. The C routines provide low-level routines which requires speed and/or are interfaced with C libraries, while the script interpreter handles all the game logic and high-level aspects.

The script interpreter used is Guile, which is recommended by the GNU project. Note that many other scripting languages could have been used, including Python, Perl or Lua. Still, the needs of Liquid War 6 are sufficiently simple that any scripting language can do the job. This is not to say that Scheme is inferior to any language - it might on the contrary be more powerfull - but for Liquid War 6, it was powerfull enough, and no other language had that very special feature which could have justified their used instead of Guile/scheme.

9.1.4 Internal libraries

9.1.4.1 Program splitted in libraries

The liquidwar6 executable itself is fairly small. In fact most of the C code - not even speaking of Scheme code - is located in a set of libraries/modules which are installed with the main program. This is different from the external libraries (libGL, libguile...). Here we are speaking of libraries which are very Liquid War 6 specific.

However, they might prove usefull in other games or applications, and after all, they are ready to use. Moreover, splitting the program into libraries makes modularity a reality. Experience shows than when a program is a monolithic C program, the temptation of putting everything in the same place is very strong, and this leads to spaghetti code.

9.1.4.2 Private and public interfaces

Each library exports a public interface and hides its internal. Since Liquid War 6 uses standard C and no C++, there's no real standard way to handle public/private features. The convention used in Liquid War 6 is to show internal structures as opaque pointers (`void *`) whenever some function needs to operate on a structure which has possibly private fields. This way the caller function has no way to access the internals, and we are sure that no reference to any internal implementation specific feature will appear.

Here's a code excerpt from `src/gfx/setup.c`:

```
void _lw6gfx_quit(_LW6GFX_CONTEXT *context) {
  /*
   * Implementation here.
   */
[...]
}

void lw6gfx_quit(void *context) {
  _lw6gfx_quit((_LW6GFX_CONTEXT *) context);
}
```

The function `_lw6gfx_quit` (note the "_") is internal, declared in `internal.h` whereas the function `lw6gfx_quit` is public, and is therefore exported in `gfx.h`.

This way, functions in the program using `lw6gfx_quit` do not know what is in the `_LW6GFX_CONTEXT` structure, and they need not know it.

This does not mean it is not possible to have public structures, only these structures must reflect some truely public, accessible and safe to access structures.

9.1.4.3 Linking

For now, linking on any of the internal library will pull down all the Liquid War 6 dependencies, including Guile, OpenGL, and the rest, even when it is not needed. This is due to automake/autoconf which automatically set these dependencies when a corresponding `AC_CHECK_LIB` call is present in `configure.ac`.

This might be fixed in the future, spending time on this would require that there is actually a real project planning to use one of Liquid War 6 internal libraries.

9.2 Coding guidelines

9.2.1 Project goals

One of the purposes of Liquid War 6 is to make a cleaner implementation of Liquid War than the previous one, namely Liquid War 5. While the latter has achieved the practical goal of providing a playable implementation of the game, it failed at providing an evolutive platform. Network capabilities where finally added to Liquid War 5, but anyone who played on Internet with someone a few hundreds of milliseconds away would agree that it's far from being perfect. The main reason for this is that it is really had to hack on Liquid War 5, especially when you are not the core developper. The core developper himself, even knowing all the various hacks in the game, is very quickly lost when trying to implement major changes.

To put it short, Liquid War 5 is a global variable hell, a pile of hacks on top of a quick and dirty implementation. Still, it works.

With Liquid War 6, the idea is to take the time to make something stable, something nice which will enable developpers to implement the cool features, and have fun along the way.

9.2.2 Common sense

Here are a few guidelines which I think are common sense advice, but they are still worth mentionning:

- try and respect the GNU coding standards,
- absolutely no `strcpy` or `sprintf` anywhere in the code, use their equivalent `strncpy` and `snprintf` systematically, as they are part of the glibc and are an order of magnitude safer,
- keep global variables for when there is something truely global, and even in that case try to fit them in clearly identified structures.

9.2.3 Application specific issues

9.2.3.1 Unitary tests

Each of the internal libraries in Liquid War has a "test" program associated with it. For instance `liquidwar6sys-test` is associated to `libliquidwar6sys`, and its purpose is to test the features of this library.

While it is fairly easy to test out unitary functions which require no peculiar context, testing high-level functions which requires files, graphical and possibly network contexts to exist is obviously harder to achieve. There's no easy way to draw the line, but the idea is to put in these test executables as much features as possible, to be sure that what is tested in them is rock solid, bullet proof, and that one can safety rely on it and trust that code when running it in a more complex environnement.

These test executables are also very good places to see a library API in action, find code fragments, and make experiments.

9.2.3.2 Memory allocation

The `libliquidwar6sys` provides macros to allocate and free memory. One should use them systematically, except when trying to free something allocated by another library.

See the documentation for module `libliquidwar6sys` for more information on how to use the macros.

9.3 Modules

9.3.1 libliquidwar6sys

System functions. Provides access to various utilities which can be used by any other module.

9.3.1.1 Log API

A basic log API is provided. The idea is not to try and make better than `syslog` or any existing standard log API, but simply to wrap log calls so that they are handled in a uniform manner in the application, and that it is trivial to change logs behaviors.

Using this API is pretty straightforward:

```
lw6sys_log_info("abc",_("this is %s"),"ABC");
```

Using `lw6sys_log_info` means the message is purely informative. Other options are `lw6sys_log_debug`, `lw6sys_log_warning` and `lw6sys_log_error`.

Second argument uses a call to function `_` which means the text is i18n enabled with gettext.

Third argument is just to show that the functions can handle string formatting the way `printf` does.

9.3.1.2 Memory allocation

Dynamic memory allocation is a common pitfall in C programming. One advantage of higher level languages such as Scheme, Perl or Python is that they handle memory management for you and therefore avoid many bugs, and consequently many hours of debugging.

The module provides two macros, `LW6SYS_MALLOC` and `LW6SYS_FREE`. Both work the way you think they should, that is like `malloc` and `free`.

Still, there's some magic happening under the hood.

Indeed, these functions:

- keep a track of the number of allocations and freeing of memory blocks. This is very convenient for it makes it possible at the end of the program to check that every single block of allocated memory has been freed. A very valuable tool to track memory leaks;
- automatically log a message with the source file name and the line number whenever a memory allocation fails. While this does not mean the caller does not need to check if the returned pointer is NULL, it can save time debugging and does save the hassle of manually printing a warning message after each failed memory allocation.

To implement this, some global variables (to hold the global memory allocation/freeing counters) need to be declared, the macro `LW6SYS_MALLOC_WIZARDRY` does this for you. See the unitary test programs (for instance `src/sys/test.c`) to see how this work in practice (also check the use of `LW6SYS_CHECK_MALLOC_FREE_COUNT`).

Concerning performance, calling these macros will obviously be slower than calling directly their glibc equivalents. The choice in Liquid War 6 is to renounce to this form of optimization and prefer the comfort of handy debugging tools to the risk of memory leaks.

9.3.1.3 File utilities

The module contains utility functions which ease up file handling, for instance it allows you to read a whole file in one call, or test the existence of a file.

9.3.1.4 String utilities

The module also contains utilities to handle strings, that is 0 terminated `char *` pointers.

Most of the time these are only simple wrappers which call three and sometime only one standard glibc function, but it's convenient to use them to:

- handle the same problem in the same manner all the time,
- use our home-made memory functions, which help tracking memory leaks down.

A good example is string copy. There is a builtin glibc function which is `strdup`. But we prefer using `lw6sys_str_copy` for it will allocate memory using `LW6SYS_MALLOC` and therefore keep track of the call, and expect and check for the matching call to `LW6SYS_FREE`.

9.3.1.5 Chained lists

The module provides tools to handle chained lists.

Again, the idea is to be consistent with the use of `LW6SYS_MALLOC`.

While one might argue that chained list handling is typically Scheme's domain of excellence, and that it's akward to do this manually in a Guile enabled program, the answer is that:

- some list processing must still be done in C. The most obvious example is manipulating C structure before or after calling Guile functions which deal with list themselves. There's a chicken and egg problem here, someone has to materialize the list first, and it cannot always be Guile.
- some code needs to be run even before the Guile interpreter has been set up, or it can simply be interesting to have some chuncks of code not depending on Guile.

The implementation is very basic, no fancy list handling, only `push`, `pop`, `is_empty` and that's about it.

The data is stored in a `void *` pointer which should point to your data. You'll need to cast it manually when you want to read your data.

A `free_func` attribute can be defined, which is called when list nodes are deleted. This enable the chained list tools to handle objects which have been allocated dynamically and free them properly through a callback system. A side effect of having `LW6SYS_MALLOC` defined as a macro is that it can't be used directly as a callback. Use the function `lw6sys_free_callback` for this. But do not use it systematically instead of the `LW6SYS_FREE` macro, as the macro gives more debugging informations when it fails, including source file and line number for instance.

See the test program `./src/sys/test.c` to see the API in action.

9.3.1.6 Associative arrays

This is very similar to the chained list API, it provides a way to handle associative arrays, AKA dictionnaries.

Note that the code here is highly unoptimized, and that handling large associative arrays with it will be a performance killer. There is no hash-table, and when you query an object all the keys are read and compared with `strcmp` to figure which key is the right one.

Still, having this is convenient for it avoids limitating the program with hardcoded limits. You can fit any number of items in these associative arrays, if there are too much of them it will be slow, but it will still work.

Liquid War 6 does not make an intensive use of this, it is just here to handle things like representing options in memory after they have been loaded from disk or interpreted from the command line.

The `void *` pointer on the data, the value of the key/value pair which forms the dictionnary, is handled the same way than data in chained lists. That is it can be freed automatically with the `free_func` attribute, which is a callback, and can be set to `NULL` if you do not need that feature.

The `char *` pointer on the key, which is a string, is handle in a different manner. In fact, it is automatically duplicated when you create an entry, and automatically deleted when you delete an entry. Therefore using a freeing callback for this makes no sense.

See the test program `./src/sys/test.c` to see the API in action.

9.3.1.7 i18n support

Todo...

9.3.1.8 XML tools

The module provides a simple wrapper over expat functions. It is used to parse primitive XML files with a general key -> value scheme of the following form:

```
<element key="foo" value="bar" />
```

While the use of XML for storing such simple informations is questionnable, it makes no doubt it's safer to rely on the well tested routines of expat rather than code a home-made parser. Additionnally, using XML will make the transition easier the day we need to store more complex and structured information.

9.3.2 libliquidwar6cfg

Configuration routines. Provides a high-level API to read, update, and save configuration options.

Command-line options and file-based options are handled in a uniform manner.

9.3.3 libliquidwar6gfx

Graphics functions. Also provides input functions. This is logical since input is related to the video output system. For instance using OpenGL with SDL implies that SDL handles the input. The keyboard input is not handled the same way when running X11 and when running in console mode.

The current implementation of libiquidwar6gfx relies on SDL and OpenGL, but this is not mandatory. It is theorically possible to implement a new target (plain X11, ncurses, ...), without changing a single line of code in the other modules. However this is obviously not a priority.

Depends on libliquidwar6sys and libliquidwar6ker.

9.3.4 libliquidwar6map

Loads maps into memory. Basically this module is used to transform .png files located on the file system to a workable memory structure.

This has been separated from the rest since it's a little special, for it requires access to functions which are typically found in graphical libraries (read a .png file) so we need to link it to some graphics related .so files. But it does not do any actual video work. We use graphics formats as a well known easy-to-use storage backend.

Depends on libliquidwar6sys.

9.3.5 libliquidwar6ker

The core algorithm. This is where all the interesting and definitely Liquid Warish code is kept. It contains the shortest path algorithm imagined by Thomas Colcombet back in 1995. It tries to have as few dependencies as possible, to ease its reuse in other software.

Not implemented yet.

Ideally, depends on nothing, might depend on libliquidwar6sys.

9.3.6 libliquidwar6snd

Handles everything related to sound and music. What is planned is the use of CSound. It would allow the writing of cool music, and even contextualize the use of music - making it faster, slower, louder, scarier, whatever... - depending on what's happening within the game.

Not implemented yet.

Depends on libliquidwar6sys.

9.3.7 libliquidwar6net

Will handle all the network stuff. Using a simple POSIX socket API won't be enough, Liquid War 6 has an ambitious goal of getting rid of the server/client mode of connecting to games. Ideally, there would be no server, simply join a game and whenever the person who initialized the game quits, then another player's computer takes the responsability to handle the game. This way one could imagine a never-ending Liquid War game. Wether this will be implemented from scratch, or if a peer to peer enabling library such as GNUnet will be used, is not decided yet.

Will depend on libliquidwar6sys, maybe also libliquidwar6ker.

9.3.8 liquidwar6script

This is not, like the other modules, a shared library, but rather a collection of scheme scripts which contain all the logic of the game. These scripts call the other libraries API, and are the core of the game.

This is probably where hackers would like to start. The scripts are in the `./src/script` directory of the source tarball, and installed in `/usr/local/share/liquidwar6/script/` by default.

Appendix A GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
   a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

   This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.
8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.
10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software

which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the program's name and a brief idea of what it does.
Copyright (C) yyyy  name of author

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type ‘show w’.
This is free software, and you are welcome to redistribute it
under certain conditions; type ‘show c’ for details.
```

The hypothetical commands ‘`show w`’ and ‘`show c`’ should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than ‘`show w`’ and ‘`show c`’; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
‘Gnomovision’ (which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

Appendix B Copying this manual

B.1 GNU Free Documentation License

Version 1.2, November 2002

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

   If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

   If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

   If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

   It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

   You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

   A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any,

be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their

titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

   You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

   The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

   In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

   You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

   You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

   If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

   If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

    The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

    Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

B.1.1 ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  year  your name.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.  A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being list their titles, with
the Front-Cover Texts being list, and with the Back-Cover Texts
being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.