Luau

Official White-paper

Author: David Eklund
Date: 14 Sept 2003
Last Updated: 20 Jun 2004
Revision: 1.03

Table of Contents

I.Introduction

1.1.Preface

One of the mantras of the Open Source community is “Release Early, Release Often.” This is often considered one of the central convictions of the “bazaar style” of software development. To quote Eric S. Raymond:

> In the bazaar view, on the other hand, you assume that bugs are generally shallow phenomena—or, at least, that they turn shallow pretty quickly when exposed to a thousand eager co-developers pounding on every single new release. Accordingly you release often in order to get more corrections, and as a beneficial side effect you have less to lose if an occasional botch gets out the door.

There is, however, a downside to this development model.

When you install your brand-new Linux distribution, you're on the cutting edge. You've got thousands of packages, fresh from the presses, ready to serve your every need. But as times wears on and you explore the inner recesses of each of these thousands of packages, a problem arises – in your exploration, you discover the ever-dreaded presence of *bugs*. After all, no software release is perfect – even the Big Guys have to release patches every once in a while for their employees' minor bungles.

So what do you do? There's a couple solutions available to you:

1) Live with it
2) Whine about it
3) See if a new version has been released that fixes it

What we're most interested in is option three. Certainly if konqueror stalls and spits out strange characters in a foreign character set every time you try to visit http://luau.sourceforge.net (which is, of course, your homepage, right?), you're going to go see if the problem is recognized and has been fixed – and let's say it has. But now there's another problem.

Konqueror relies on a total of **35 external libraries**. While for this issue it's likely the bug is in konqueror itself or libkonq, any small bug in any of the 34 other libraries could conceivably bring Konqueror to a grinding halt. And if you're not the type to go source code spelunking, then you have but one choice: *find and install the newest version of **all 35 libraries***. Plus Konqueror itself.

Of course, it's just good system administration in general to keep on top of all the latest software updates – or at least the ones that can affect your system's performance, stability, and security. The problem with trying to keep up with all the updates for all those thousands of packages your distribution installed has certainly been tackled before. Debian has apt. FreeBSD has the ports tree. Gentoo has a ports look-alike. And RedHat has the Red Hat Network. And these all work rather well for their purposes – but what

I'm here to provide is an alternate solution.

1.2.Overview

The problem with distributor's providing access to all those new software updates is that, truthfully, even they can't keep up with all the latest. If you're not big into the cutting-edge, then apt's system works pretty well, for example – but if you want the latest and greatest, you're simply going to have to go to the source and download it yourself. Luau isn't designed so much for production status servers (in fact, I would recommend against it) where everything needs to work together perfectly – rather, it's designed for the enthusiast who wants instantaneous updates, straight from the source. It's for those of us who want to keep up with the latest developments for all our favorite software packages – and maybe some of the more mundane ones as well. In the end, it's for the home user, not the corporate user.

So finally, you ask, what is Luau? Luau is a system for retrieving software updates and status messages from package maintainers **individually** – that is, it gets the information straight from the source as opposed to a third party, such as Red Hat. Of course, there's a caveat to this – in order to get information directly from the source, the source has to provide some information (all you developers out there – it's really easy! Go skip down to the *Developer's Guide* and read the overview). So if the maintainer for your favorite software package doesn't support Luau and you want to use it, go bug her – she most likely simply hasn't heard about it.

Luau works through the use of a simple "software respoitory" file that's publicly available (published by the software maintainer) and describes all the currently available updates to all current software releases. In general, this will simply consist of a list of all the currently available releases, though more complicated scenarios (along with the possibility to send messages and other configuration updates to the user-base) are also available. For more information on this, skip on down to the *Developer's Guide*.

1.3.Why Luau?

At this point, you may be asking why all this trouble to implement a solution to a problem that's already been solved by the likes of apt, RHN, and so on. As mentioned before, Luau is not meant to replace these systems unilaterally, but rather to supplement them. Both ways of working have their pros and cons, some of which I've tried to enumerate below:

	Pros	***Cons***
Apt, RHN, ports, etc.	• Software updates more likely to work with one another • More packages available (for the moment...) • Usually more stable software	• Updates not as current • In some cases (RHN) is not free • Sometimes hard to convince to support lesser known packages • No personal connection with actual software developers

	Pros	***Cons***
Luau	• Provides not only software updates but also status messages • Instant updates as soon as the maintainer posts them • Decentralized, so you don't have to rely on only one source • Only the packages you care about are monitored • If you can only convince the maintainer to support Luau, any package you like will be supported	• Software may be unstable, or may not work well together with other packages • Does not provide automatic dependency tracking and downloading (possible in future?) • Only available for packages for which respository files are provided for by the maintainer

1.4.Revision History

Version 1.03
20 Jun. 2004
This release again updates the Developer's Guide to describe the new 1.1 Luau repository XML file. Also the "Using luau-register" section was modified to describe the new --from-file and --from-url options. A couple of other minor spelling/grammatical/stylistic modifications were also made.

Version 1.02
14 Mar. 2004
This release is updated to cover the changes in the XML format. Also the “Release Date Compatibility Scheme” section has been dropped (it was obsolete) and replaced with a description of Luau's version parsing capabilities. Section 3.4 (“Using luau-register”) has been amended to describe Autopackage's “root name” scheme. Finally, some minor spelling/grammatical updates have been made.

Version 1.01
25 Dec. 2003
This release is updated to reflect the change in the updates file style and the new options added to the Luau console clients (`luau` and `luau-register`).

Version 1.00
14 Sept. 2003
This is the first public release of this document. Nothing to say here!

1.5.Contact Information

David Eklund
Project Maintainer and Head Developer
<deklund@fastmail.fm>

Please contact me with any ideas, criticism, or bugs you have to share with me. Furthermore, if you believe you have anything you can contribute to Luau in the form of coding, translations, web page design, or whatever else you can think of, talk to me – most likely I can use your services.

Luau Web Site: http://luau.sourceforge.net

1.6.Distribution Policy

This document is distributed under the GNU Free Documentation License. For a full copy of the license, see Appendix A.

II.End-User's Guide

2.1.Luau Basics

Luau is a framework for software distribution and update transmission. The first thing to make clear is that *Luau is not in any way a package manager, nor does it have anything to do with package management*. Luau does support various types of packages – specifically RPMs, DEBs, Autopackage files, self-extracting executables, and generic tar balls – and, for those that it can, it supports automatic installation (meaning everything except tar balls, which obviously cannot be installed automatically), but it is not *designed* for package installation, uninstallation, or management. You will have to find a different tool for this – there are many available.

The second thing to make clear is that *only software packages that explicitly support Luau can be updated through Luau*. At the moment, this is a very small number (only one at the time of this writing – that would be Luau itself), but hopefully this number will increase. If you would like to use Luau with some of the software packages you use often, contact the developers! If enough people are interested, they will most likely start supporting it, since it takes very little upkeep on their part. If you're particularly motivated, you can even support it yourself – you don't have to edit any source code. You only have to provide an file describing all the current software revisions available (see the *Developer's Guide* for details), and then you can contact the package developers again and – unless they're extremely opposed to the idea – they will simply link up to your page. It's all very simple – again, see the *Developer's Guide* a few pages down for details.

There are **three** basic kinds of updates supported by Luau: software updates, message updates, and Luau configuration updates. The first kind are the most prevalent and the main focus – they provide updates to installed software, such as when a software project makes available a new release. The second, message updates, are for when software maintainers have an important message to distribute to their user-base. This may be to inform users when there is a major security flaw in certain versions of the software package, or if the development team is in need of developers or artists, for example. The final kind of update is an update to the internal configuration of Luau itself. Luau fetches updates about software projects by downloading a configuration file specified upon

installation by the software maintainer. If this location should have to change, then this kind of update will inform you of the change in server status and – if you accept it, which you certainly should – will automatically update Luau's internal database to reflect the change. These should be rare, but be certain to pay close attention should one become available – if you ignore or don't accept it, Luau will most likely stop working with respect to the specified program, and you'll have to update the program configuration by hand to get it working again.

2.2.Using `luau` – The Console App

The `luau` console application is the most basic and fundamental user-oriented segment of the Luau suite. That being said, you'll most likely want to use Luau-X for most of your needs, since it is much more intuitive and user-friendly. However, when you don't have access to an X server, or you need something to use in a shell-based environment (e.g., for a shell script), `luau` is just as powerful and as flexible as Luau-X.

There are two basic modes `luau` supports – command-line based and interactive. These should be familiar to anyone with a *nix background: the command-line based mode simply takes options from the command-line, interprets them, and executes the desired activity (download an update, list all registered programs, etc.), while interactive mode provides a prompt to the user into which (s)he can execute a string of commands.

The command line options are as follows:

```
Modes: (only one may be specified)
  -d, --download=ID     download update with given ID for specified program
  -e, --install=ID      install update with given ID for specified program
  -g, --getupdates      retrieve a list of updates for the specified program
  -i, --interactive     run in interactive mode [default]
  -l, --list            list all registered programs
  -h, --help            display this message

Extra Information:
  -m, --email=ADDRESS   email the results to ADDRESS, if any results at all
  -o, --output=PATH     when downloading an update, specify where to download
  -p, --program=NAME    specify a program
  -t, --type=TYPE       specify the type of update to download/install
  -q, --quiet           suppress all unnecessary output
  -v, --verbose         display more informational output

To download or install updates, both the program name and the type to download
MUST be specified.
```

This can, of course, be seen any time by executing `luau -h` (or `luau -help`). Little extra explanation is needed here – the basic idea is that one (and *only* one) mode must be specified, in addition to any number of "extra information" options. Note that (as should be obvious) you *need* to specify both a program name and an update type (-p, -t) in order to download or install an update. Similarly, it doesn't make any sense to specify an update type when you're only trying to list registered programs; `luau` probably won't complain, though. Interactive mode offers almost the same functionality:

```
List of commands:

get <PROGRAM-ID> <UPDATE-ID> [<PACKAGE-TYPE>] [<FILENAME>]
 - Retrieve a package of type PACKAGE-TYPE and download it to FILENAME
   (if specified), or the current directory (if not).
install <PROGRAM-ID> <UPDATE-ID> [<PACKAGE-TYPE>]
```

```
 - Retrieve and install an update of type PACKAGE-TYPE.
list [<PROGRAM-ID>]
 - If PROGRAM-ID is specified, show whether it is registered.  Otherwise,
   list all registered programs.
updates [<PROGRAM-ID>]
 - Retrieve and list all updates available for PROGRAM-ID (or all
   registered programs, if not specified).
quit
 - Quit luau and return to command prompt

Note that for PACKAGE-TYPE, you must enter one of the following:
  RPM DEB SRC AUTOPKG EXEC
```

Again, this can be viewed at any time by entering “help” or “?” at the interactive prompt.

One final note: although it's not specified in the small help message, if you select list mode (-l) and specify a program name (-p), it will output *only* the name and version of the specified package, and it'll only do that if a program with the name specified is registered with Luau. This may be helpful in shell scripts (or `configure` scripts) to see if a program needed is installed. Other details like this can always be found in the one definitive source: the man page (`man luau`).

2.3.Using `luau-x` – The X App

This will most likely be where you do most of your work with Luau supported programs as an end-user. Luau-X s the X version of `luau` – it provides all the same functionality, but in a neat little graphical package. Luau-X actually provides more functionality than `luau` in the fact that it shows and can modify the data associated with each registered program (see `luau-register` in the *Other Apps* section below for information on how to do this from the console). However, the main focus of Luau-X is to provide an easy way to find available updates for all registered programs simultaneously, and to view and install those you find appropriate.

The main factor in getting used to Luau-X is simply to use it – it's a rather simple interface and finding out how to do what you want to do should be a rather simple process. There are a few caveats to point out, however. First of all, as you may know if you've browsed the *Programmer's Guide*, Luau (or rather the library interface, libuau) provides a standard error output facility. In Luau-X this is redirected into an error log. To view the error log, simply select View->Error Log from the main window. This may seem like a trivial fact to point out here, but it is of utmost importance in ironing out the problems that will almost certainly arise as developer's grow accustomed to the Luau interface and as the Luau system itself matures. If you have any problems concerning downloading updates, installing updates, or anything else, this is the first place to look – it may help you solve your problem, and if not, it will be the first thing any of the Luau developers will ask for if you ask them for help.

Another thing to note is that, like the rest of Luau, Luau-X is a work-in-progress, and bugs are unfortunately to be expected. If you have any problems, don't just live with it – email me! First of all you should make sure you have the latest revision installed (which shouldn't be hard to find out, considering you have Luau installed and that's exactly what it's for), and if possible see if you can fix the problem on your end, but if nothing works, please feel free to drop me a line. My e-mail address is <deklund@fastmail.fm> and I'm

always more than willing to answer questions and receive bug reports.

2.4.Other Apps

Currently, the only other application provided with the Luau suite is `luau-register`. In general, end-users shouldn't have to deal with this program – in fact, it's generally a bad idea to at all, unless you know exactly what you're doing. It's used to register new programs with the Luau database or to update existing programs to reflect configuration changes. As such, it will most often be called from installation scripts run by Luau-supporting software packages as they register themselves with the Luau database in order for you to be able to check and download updates for them. If you'd like more information about `luau-register` or how this process works as a whole, skip on down to the *Developer's Guide* – there's plenty of useful information on everything you need or want to know there.

2.5.Final Notes

I've already mentioned it, but I'd just like to re-emphasize – please, please contact me with all bugs, inconsistencies, or just general comments that you have. One of the driving factors of Open Source software is the close link and communication between users and developers (and I believe Luau will only help encourage this, through the use of message updates) – and as such, it is your responsibility as a user to keep the development team updated with any problems there may be. And believe me: if you're trying out one of these first Luau releases, issues will inevitably arise.

So once again, in case you missed it, my email address is <deklund@fastmail.fm>, and I would love to hear from you. All correspondence is greatly appreciated, and I will try to get back to you as promptly as possible.

III.Developer's Guide

3.1.How Luau Works

The heart of Luau's updating system lies within the software repository file. This is a file hosted somewhere public by the project maintainers which in essence describes all currently available updates and messages. As a developer, the *only* thing you have to do to support Luau is to host this file and keep it updated. Then you simply add a command to the installation script for your project (more on this later) in order to register the location of the updates file with the Luau database stored on your client's computer, and you're done. If you want more flexibility, you can also interface with the libuau library and take direct control over the Luau facilities (see the *Programmer's Guide* for details), but for most purposes, simply registering your program and then letting the Luau application suite deal with the rest should be enough.

3.2.Understanding the Luau XML File

The style of the updates file is a simple XML format:

```
<keyword [att1=<value1> [att2=<value2>, ...]]>
```

Since there are three separate types of updates – software updates, messages, and Luau configuration updates (see section **2.1**) – it should make sense that there is a different set of keywords available to each type. However, there's also a set of keywords common to all types: we'll cover those first.

We'll begin with an example updates file (this is in fact a copy of the `example.respository.xml` file distributed under `docs/` with Luau):

```
<?xml version="1.0"?>
<!DOCTYPE luau-repository SYSTEM
                  "http://luau.sourceforge.net/luau-repository-1.0.dtd">

<luau-repository interface="1.0">

  <program-info id="@example.com/myproject">
    <shortname>myproject</shortname>
    <fullname>MyProject Projector</fullname>
    <desc>Projectilize projecting projectiles</desc>
    <url>
      ftp://ftp.example.com/myproject/myproject.repository.xml
    </url>
  </program-info>

  <software version="1.1.1">
    <date>2003-04-12</date>
    <keyword>UNSTABLE</keyword>
    <short>Upgrade to myproject version 1.1.1</short>

    <long>
Version 1.1.1 supports CoolStuff extensions and drops legacy support
for pre-1.0 versions.
Many bug fixes have also been made, including:
  o Preventing myproject from crashing when user inputs curse words
  o No more gamma radiation problems (hopefully)
!!! NOTE !!!
INSTALLING version 1.1.1 WILL DROP COMPATIBILITY WITH BINARIES COMPILED
WITH PRE-1.0 versions!
If this is a problem, please stick with the 1.0 branch
    </long>

    <package type="RPM" size="1200144"
                        md5="29168bed9cc7e04652e82bc09b3c7b98">
      ftp://ftp.example.com/binaries/myproject-1.1.1.rpm
    </package>

    <package type="DEB" size="1432534"
                        md5="98336d9f8cb04bc909a9a2a13a9e893c">
      ftp://ftp.example.com/binaries/myproject-1.1.1.deb
    </package>

    <package type="SRC" size="6323511"
                        md5="717ab9d5dfcb8900ca14182dc0a0dbb2">
      <mirror weight="80%">
        ftp://ftp.example.com/src/myproject-1.1.1.tar.gz
      </mirror>

      <mirror weight="20%">
```

```
        http://www.example.com/myproject/myproject-1.1.1.tar.gz
      </mirror>
    </package>

    <valid type="date" from="2003-05-02" to="2003-05-15"/>
  </software>

  <update type="message">
    <id>201</id>
    <date>2003-04-14</date>
    <short>Artists needed!</short>

    <long>
myproject is looking for some talented artists to help with the newly
planned action simulation adventure game, "MyProject: The GAME."  If
you're interested, please contact jimbo@example.com
    </long>
  </update>

  <update type="luau-config">
    <id>301</id>
    <date>2003-04-14</date>
    <keyword>IMPORTANT</keyword>
    <short>New server</short>

    <long>
We've got our own ftp server now, so we're trying to get everyone to
switch over to using the example.com database.  Please accept this
update, since the old ftp server will be unavailable in a month or so,
and trying to autoupdate from it after that will fail.  Thanks. :)
    </long>

    <set url="ftp://ftp.example.com/pub/myproject/updates"/>
  </update>
</luau-repository>
```

General Keywords:

<**luau-repository** interface="*version*">
  [...]
</**luau-repository**>
Values for `version`: "`1.1`" (**required)**
Enclosing tag for all software and updates. The "`interface`" attribute defines the luau-repository file format version – the most current (and only available, for the moment) is "1.1," and should be specified as such in all Luau XML files.

<**software** version="*revision*">[...]</**software**>
Values for `version`: parse-able program version identifier (**required)**
Example:

```
<software version="1.1.1">
  [...]
</software>
```

Starting declaration for a software package definition. The "`version`" attribute *must* be in a form Luau can parse – if your versioning scheme does not fit such a model (almost all do), then you'll have to specify *that* definition in a <**display-version**>

tag and use a different easily-readable version identifier here. See section **3.3** below.

<**update** type="*update-type*">[...]</**update**>
Values for type: message, luau-config (**required**)
Example:

```
<update type="message">
  [...]
</update>
```

Starting declaration for a new (non-software) update definition – both messages and configuration updates.

<**id**>*identifier*</**id**>
Example: <**id**>201</**id**>
Assigns an ID to an update. **Required** for all message and luau-config updates. *identifier* may contain any alphanumeric characters. Also, it must be **unique** (for the program in question)**.** There are no other specific requirements for *identifier*, but it suggested that you make it descriptive and concise (less than 12 characters or so if possible). It is also possible to specify an ID for a software package, but it is unnecessary: Luau by default simply uses the version of any given package (as specified by the version= attribute or the <display-version> tag) as its ID.

<**date**>*MM/DD/YYYY*</**date**>
Example: <**date**>03/17/2003</**date**>
Assigns a date to an update. Is valid for all three kinds of updates (and also recommended for all three), but not necessary for any. This is not used directly by Luau in any way, but it is useful for end-users to see when an update was released.

<**short**>*short-description*</**short**>
Example: <**short**>Upgrade to the myproject version 1.1.1</**short**>
Provides a short description for the update in question. Is valid and highly recommended for all three types of updates, though not required by any. Must be no more than one line long.

<**long**>*long-description*</**long**>
Example:

```
<long>
Version 1.1.1 supports CoolStuff extensions and drops
legacy support for...
</long>
```

Provides a long description for the update in question. Is valid and highly recommended for all three types of updates, though only required for message updates. Should be a paragraph or two long, depending. If you have anything particularly lengthy to describe, you should redirect the user to a web page instead, since the Luau message view dialog isn't really appropriate for reading pages and pages of material. For a software release, for example, you may want to include some brief release notes here, but a full Changelog would be inappropriate.

<**keyword**>*keyword-name*</**keyword**>
Values for `keyword:` `STABLE, UNSTABLE, IMPORTANT, [user_defined]`
Example: <**keyword**>STABLE</**keyword**>
Assigns a keyword to an update. This is used internally to demarcate and/or emphasize certain aspects of specific updates. Also, if you plan on using libuau, you may define your own keywords and use them in your own program – luau and luau-x will simply ignore them. Note that keywords are case-sensitive and all-uppercase keywords are reserved for Luau's use – if you want to use your own, make it lower case (or mixed case).

Keywords Specific to Software Updates:

<**package** type="*pkg-type*" size="*pkg-size*" md5="*md5sum*">
pkg-location **OR** *[<mirror>...]*
</**package**>
Values for `type:` `RPM, DEB, AUTOPKG, EXEC, SRC` (***required)***
Values for `size:` integer specifying package size in bytes (*optional)*
Values for `md5:` 32-character string specifying the md5sum of the package (*optional)*

Example:

```
<package type="RPM" size="1002453" md5sum="95dbc2411471aab385ad4adf18a8b5b7">
   <mirror weight="80%">
      ftp://ftp.example.com/pub/myproject-1.1.1/binaries/myproject-1.1.1.rpm
   </mirror>

   <mirror weight="20%">
      ftp://ftp.other.com/pub/myproject-1.1.1/binaries/myproject-1.1.1.rpm
   </mirror>
</package>
```

Specify a package where this update can be downloaded. *pkg-location* must be a URL that can be read and parsed by libcurl (any FTP or HTTP location should be fine). Required for any software update.

Supported <`mirror`> *Declarations (inside* <`package`> *tag):*
<**mirror** weight="*weight%*">*mirror-location*</**mirror**>
Values for `weight:` integer between 1 and 100 specifying how frequently that mirror should be chosen over the others (e.g., a mirror with weight 20% will be picked 20% of the time) (*optional*)
Example:

```
<mirror weight="80%">
  ftp://ftp.example.com/pub/myproject-1.1.1/binaries/myproject-1.1.1.rpm
</mirror>
```

Specifies a mirror where the file can be downloaded with a specific weight. Used when the package can be question can be obtained from several locations. If the weight is not specified, then every mirror is given equal weight.

<**valid** type="*valid-type*" from="*from*" to="*to*" *for="for" />*
Values for `type:` `date, version`
Example:

```
<valid type="date" from="02/05/2003" to="05/15/2003" />
<valid type="version" from="0.2.0" />
```
Specifies certain conditions for which this update is valid. You shouldn't have to use this under most cases, unless you specifically want to hide certain updates from certain users (one good example would be to put out a message only for users using a release witha major security vulnerability – in this case, you'd use something like `<valid type="version" for="0.2.4" />`. See section **3.3** below for more information on Luau's version parsing abilities.

Keywords Specific to MESSAGE Updates:

There aren't any. The LONG and SHORT keywords provide you with everything you should need.

Keywords Specific to LIBUPDATE Updates:

`<set url="valid_url" />`
Example: `<set url="ftp://ftp.example.com/pub/myproject/updates" />`
Defines the new place where an updates file can be accessed. *Valid_url* must be (surprise, surprise) a valid URL – specifically, one that can be parsed and downloaded by cURL. At the moment, this is the only update you can make through a "luau-config" update.

3.3.Version Parsing in Luau

One of the boons of the Luau style of software distribution is that any number of software revisions are available at any point in time – meaning that if you ever have the need to "down-grade" software due to unexpected limitations or bugs in a newer version, you always have that capability (for this reason it is suggested that distributors not remove old software revisions from the Luau XML file unless they truly are ancient). On the end-user's side, however, one doesn't want to be bogged down by a list of old and essentially irrelevant software revisions when one only wants to check if any new releases have come out. For this reason, both the Luau front-ends by default filter out and hide all available software versions older than the currently installed version.

The problem with this scheme is that not all versioning schemes fit the same template. Fortunately Luau's version parsing algorithm (based off the one used by Autopackage) is fairly flexible, but there will always be some exceptions.

In general, any versioning scheme that roughly follows any of the following formats shouldn't have a problem:

- **1.2.4, 3.5-rc10, 3:1:2 –** Any scheme that follows the pattern a.b.c.d.(...), where the first number (or letter) represents the most "significant" release number (e.g., 3.2 > 2.3) will work. Any non-alphanumeric character can be used to separate the major, minor, etc. revisions. Note that an alphabetic release *always* precedes a numeric release – for example, 1.2 > 1.2b, 2.0 > 2.0-rc10, 2.1.2 > 2.1b. This may cause some problems for some people – see below.

✔ **1999-12-03, 20030115 –** these date-like versions will work **only if** the year is listed first, followed by the month, followed by the day (i.e., with the most "significant" index first). It should be clear to see that this is parsed just like the major.minor.patch case, except with different separators.

These, on the other hand, will **not** work:

✗ **1.0 < 1.0a < 1.0b < 1.1** – As mentioned before, an alphabetic release is *always* "less than" a numeric release: in this case, version "1.0" will be read as being a **later** release than "1.0a." There isn't really anything I can do about this, because there is an intrinsic ambiguity in the notation – when you say "1.0a", do you mean "1.0 **alpha**" or "supplement to 1.0?" More schemes fall into the former case than the latter, so that's what gets supported.

✗ **"Decreasing" Version Numbers (e.g., "counting down from 100") –** I've seen a few people get really creative with their versioning scheme by doing things like counting down from a certain original release number (like 100 or 1000), decrementing the version number with each new release (100, 99, 98, and so on... apparently, by the time you get to zero, you will have obtained the perfect release candidate). This can't be supported for obvious reasons.

✗ **12/24/2003, 01-03-2004** – These schemes are not supported because they don't list the requisite parts of the version in a "logical" ordering. While it would be *possible* to try to get Luau to understand these schemes, it would almost inevitably break certain other non-date based valid schemes – again, there's an ambiguity in a version like "1.2.98": do you mean "January 2nd, 1998" or "Major: 1, Minor: 2, Patch: 98?" In the former case, "1.2.99" would certainly be a later release than "2.2.98," but the opposite is true in the latter. If you want to use a release date scheme, please use one as listed above or read on to find out how to get it to work.

Most people shouldn't be affected by this (from browsing freshmeat.net, almost all projects are labeled in a "major.minor.patch" form), but if your project's versioning scheme causes a problem, there *is* something you can do – though it isn't particularly ideal. What you have to do is use a **different** versioning scheme when you're amending your Luau XML file – I would suggest using a release date scheme as mentioned above to prevent any kind of confusion – and then put the "real" version number in a <display-version> tag in the relevant <software> section. For example, say you were using a decreasing scheme as mentioned above, and on February 13th, 2004 you released version "58." Then you may do something as follows:

```
<software version="2004-02-13">
     <display-version>58</display-version>
     ...
</software>
```

If on the other hand you used a "1.0 < 1.0a < 1.0b < 1.1" scheme, something as follows may work better (say you were releasing version "1.0b"):

```
<software version="1.0.2">
```

```
    <display-version>1.0b</display-version>
    ...
</software>
```

...where the alphabetic character “b” has been replaced with “.2”. This would work as you would expect.

Having said all that, however, I highly suggest that you stick to a “standard” versioning scheme. Doing otherwise is only bound to confuse your users.

3.4.Using `luau-register`

luau-register is an incredibly simple application, but it's also a very important one. Since Luau works on a decentralized basis, it's your responsibility as a developer to register your program with the internal database on each of your user's systems. This is a simpler task than it sounds – all you have to do is, after your software has been installed (in a post-install script, for example), automatically call `luau-register` with the appropriate values – program ID, version, release date, etc.

Luau's command arguments are as follows (as usual, you can also find this out by simply running `luau-register -h`):

```
Usage: luau-register [OPTION] ... [program_id]

Register a program with the specified details in the luau database.

Options:
  -r, --remove                        remove specified program from the database

  -u, --url=LOCATION                  location of Luau updates file
  -d, --date=\"MM/DD/YYYY\"           date of this program version's release
  -k, --keywords=”KEY1,KEY2,...”      keywords for this program revision
  -v, --version=VERSION               installed version of this program
  -i, --interface=VERSION             interface version for this program
  -n, --shortname=NAME                short “UNIX name” or program
  -f, --fullname=NAME                 full/display name of program
  -s, --desc=DESC                     one-line description of program
  -h, --help                          display this message

  -l, --from-url=URL                read program information from specified URL
  -e, --from-file=FILE              read program information from local file

  Note that both the server and the software repository URL must be set for
  Luau to work for any given program.  If no short name is specified, the
  program_id is used.  Please see the Luau whitepaper for more information.
```

An important thing to note is that your program ID *must* be unique from all other ID's used by other software projects that use Luau. The best way to deal with this is to use Autopackage's “root name” naming scheme, which looks something like this:

@example.org/program-name

...where “example.org” is the domain owned by the software maintainer, and program-id is just the name of the program in question. This way all program IDs are made unique and collisions should be rare since the root name system leverages off the uniqueness of top level domains. These names will never be exposed to the user, who will instead

always see either the “short name” (specified by luau-register) in the context of the command line, or the “full name” (again specified by luau-register) in the context of the graphical environment. As an example, Luau's root name in this scheme is “@luau.sf.net/luau”; its short name is “luau”; and its full name is “Luau Software Updater.”

The "--from-file" and "--from-url" are also quite useful, since they allow you to simply specify a Luau repository file and read in the program information from there. As a developer, you have the choice of simply specifying this instead of writing out all the program information by hand again; **however**, you **must** (in addition to using the "--from-file" option) specify the version, interface, date, and any keywords associated with the given package being installed, since that cannot be read from the repository file (since luau-register doesn't know which package you're installing). Also, please don't use the "--from-url" option in an installation script: while it sounds convenient, not all users have always-on Internet connections (or even access to one). In reality, the "--from-file" and "--from-url" options are more meant for use by the user than by the the software developer.

Any of the following commands would be valid:

```
luau-register --from-file="doc/example.repository.xml"
              --version="1.1.1"
              --date="2004-06-02"

luau-register --url="http://example.org/example.repository.xml"
              --version="1.1.1"
              --date="2004-06-02"
              --shortname="myproject"
              --fullname="MyProject Projectilizer"
              --desc="Projectilize Projecting Projects"
              "@example.org/myproject"
```

3.5.Good Policies (aka, *How Not To Annoy Your Users*)

In general, the really important thing to realize when using Luau is that your users are using it to retrieve updates for the project, and that's it. Luau provides an avenue for you to send messages to your user-base (a feature that you'll hopefully find useful), but it's important not to abuse it. Users want to hear about important security updates, or possibly *occasional* calls for recruitment – they don't, on the other hand, want to hear about your personal life, or other things unrelated to the software at hand: don't, for example, use it as your personal blog (unless a desire to read your blog was the specific reason people downloaded your software – unlikely, at best). Overall, message updates should be relatively rare things – if you're putting them out more than about once a month, maybe you should rethink what you consider to be truly important.

Particularly, *don't* send out a message update with every new software release. This is what software updates are for – no reason to be redundant. Only put out a message announcing a software release if you think it's really important – if, for example, it fixes a *major* security hole, or if it's a *major* enhancement over the previous versions (and even that's questionable).

IV.Programmer's Guide

*Sorry, this hasn't been written yet! However, there is a rather thorough description of the libuau interface (made using **doxygen**) available both in the Luau distribution (under docs/api/) and online: http://luau.sourceforge.net/api/*

Specifically, you'll want to look at the libuau.h *interface, since this is the only methods you, as a libuau programmer, will have access too. The interface is relatively simple, but beware – in its current state, it is **very** volatile. There will most likely be many updates and changes before the interface stabilizes – therefore, I recommend that, unless you specifically need to extra functionality, you simply stick to the means available (ie,* luau *and* luau-x*) instead of writing your own solution.*

V.Contributor's Guide

*Sorry, this hasn't been written yet either! Once again, however, there is a detailed description of the **internal** API as well as the libuau interface in the Luau distribution (under docs/api/) and online: http://luau.sourceforge.net/api/*

*Also, I have attempted to comment the source code and make it as clear as possible for anyone interested in contributing. As always, if you have any questions or comments (or, even better, patches!), please e-mail me at <*deklund@fastmail.fm*>. Thank you for your help!*

VI.Conclusion

As a developer, I have invested a number of months into the preparation, planning, execution, and documentation of the Luau software suite. As such, it is my sincere hope that it will be found useful and applicable to the Open Source community. However, I realize that my work alone cannot fulfill the needs of every software developer, and that certainly the needs I have addressed with the creation of Luau will not necessarily meet all the needs of others. Therefore, I ask once again for anyone who has taken the time to read through this document, or even anyone who only read the Introduction and then just skimmed the rest (you know who you are) to e-mail me any feedback you may have. I'm open to suggestions, criticism, and any offers of help you can provide.

Also let me reiterate that I am always open to providing support – if you're a developer who wants to implement Luau in your software project, but are having problems getting it to work or can't understand one of the specifics, don't give up: just ask me! I have tried to make everything as clear as possible both in coding and in documentation, but I'm no expert technical writer – I'm sure there have been spots in this document where things got confusing, redundant, or simply irrelevant. As always, any clarification you need I will happily provide. Once more, my e-mail address is <deklund@fastmail.fm> - take advantage of it.

Finally, thank you for taking the time to find out more about Luau. I believe that it has the capability of being a boon to the Open Source development model – but only if it is widely heard about and implemented by many software projects. If you know anyone who may be interested in using it, please feel free to tell them about it and redirect them to this document and the Luau web-page, http://luau.sourceforge.net.

- David Eklund, Project Maintainer and Head Developer

VII.Appendices

Appendix A. GNU Free Documentation License

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 noncommercially. 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.