Краткое руководство по предварительной подготовке программного пакета к включению в дистрибутив и репозиторий СПО СО РАН

Голосов И.С., Ефремов И.Е.,Фурсов М.Ю. (УНИПРО)

Назначение документа

Документ предназначен для авторов программного обеспечения, передаваемого для включения в репозиторий и дистрибутив СПО СО РАН, а также центральные репозитории ПО операционной системы Ubuntu Linux. Руководство содержит рекомендации по выбору свободной лицензии (обязательно для включения в репозитории Ubuntu), упорядочиванию файловой структуры программного пакета, написанию Make-файлов.

Содержание

Раздел 1. Выбор свободной лицензии

1.1 Определение свободной лицензии. Согласно Wikipedia[1], свободные лицензии — особый вид лицензий, предназначенный для обеспечения юридической защиты прав пользователей на неограниченные воспроизведение, изучение, распространение и изменение различных продуктов интеллектуальной деятельности.

Существуют множество свободных лицензий, в которых поддерживаются основные принципы, но отличаются методы и характер защиты прав пользователя, а также различны виды продуктов интеллектуальной деятельности. Почти все свободные лицензии на ПО принадлежат к нескольким категориям, наибольшую популярность из которых получило определение свободного ПО от Free Software Foundation[2].

1.2. Категории лицензий. Можно выделить следующие основные категории свободных лицензий (по увеличению «открытости»):

- **GNU General Public License (GPL).** Исходный код, единожды открытый под GPL, уже не может быть закрыт (поскольку любой, кто его получил, может продолжить его распространение), лицензия не может быть изменена, и его использование является абсолютно свободным. Линковаться с GPL-кодом может только ПО с совместимой лицензией.
- **GNU Lesser General Public License (LGPL).** Основное отличие от GPL – разрешение на линковку с LGPL-кодом не ограничивается.
- **GPL-совместимые лицензии (Apache License v2, Boost License и другие).** Основное отличие от GPL/LGPL – разрешение на использование исходного кода в проприетарных приложениях, т. к. не требуется распространять модифицированные версии кода под той же лицензией.
- **Лицензии совместимые с Open Source Initiative[3], но несовместимые с GPL (Apache License v1, Mozilla Public License).** Лицензии, по разным причинам несовместимые с GPL, например, жесткие требования к именованию, требования к выбору правовой нормы, указание конкретной территории действия лицензии и т. д.
- **Прочие, наименее ограничивающие использование исходного кода, лицензии (BSD License, MIT License), могут быть GPL/OSI совместимы**. Практически не ограничивает использование лицензированного исходного кода, единственные ограничения касаются использования имен авторов и владельцев copyright.
- **Public Domain.** Отказ авторов от всех прав на исходный код, т. е. предоставление работе статуса «общественного достояния».

1.3. Рекомендации по выбору лицензии

- Перед выбором свободной лицензии для программного продукта, нужно определиться с целями его лицензирования: популяризация ПО, поиск новых идей и партнеров, повышение доступности по сравнению с аналогами и прочее.
- В случае если использование исходного кода ПО в проприетарных приложениях нежелательно, следует выбрать GPL/LGPL.
- Для библиотек LGPL может оказаться предпочтительнее чем GPL: исходный код защищен теми же условиями, но библиотека может линковаться с проприетарными приложениями, или приложениями с несовместимой лицензией.
- В противном случае, следует использовать GPL-совместимую лицензию, такую как Apache License v2. Несовместимость с GPL может не позволить ПО входить в некоторые репозитории свободных программ. Использование более свободной чем GPL лицензии позволит проприетарному ПО включать лицензированный код (с указанием авторства/copyright и пр.), что послужит значительно более широкому распространению лицензированного кода.

Важно заметить, что выпуск исходных кодов под свободной лицензией не отнимает у автора его прав на ПО. Автор в любой момент может выпустить тот же самый открытый код под любой другой лицензией, либо выпускать закрытые версии ПО. Возможно двойное лицензирование одного и того же исходного кода, в зависимости от целей его применения, например – коммерческое использование может быть разрешено только по специальной, несвободной лицензии. Подобные модели двойного или даже тройного лицензирования широко используется коммерческими организациями.

Другое важное замечание касается патентного права – использование некоторых лицензий, например GPL, несовместимо с патентованием исходных текстов.

Ссылки к разделу 1:

[1] http://ru.wikipedia.org/wiki/Свободная_лицензия
[2] http://www.gnu.org/philosophy/free-sw.ru.html
[3] http://www.opensource.org/

Раздел 2. Рекомендации по структуре пакета исходных кодов

2.1. Основные аспекты структурирования пакета исходного кода:

- Необходимо распределить отдельные составляющие пакета по директориям:
  - src – непосредственно исходный код библиотеки/приложения
  - tests – тесты
  - examples – примеры, например, использования библиотеки
  - doc – документация
  - и так далее.
- Пакет обязательно должен распаковываться из архива в отдельную директорию.
- Рекомендуется использовать принятую в open-source сообществе схему именования: projectname-1.2.3, где projectname – имя пакета, набранное строчными буквами, 1.2.3 – версия, состоящая из трех компонент (мажорная/минорная/патч). Допускается использование версии из двух компонент: мажорная/минорная. При этом непосредственно пакет (архив) может именоваться как projectname-1.2.3.tar.gz, либо projectname-1.2.3.src.tar.gz. Архив должен, как указано выше, распаковываться в директорию именующуюся projectname-1.2.3.
- К пакету обязательно должен быть приложен файл README, помещенный в корень пакета. Его рекомендуемое содержимое:
  - Краткое описание пакета;
  - Веб-сайт проекта;
  - Общее описание структуры пакета, его директорий;
  - Прочая важная информация, которую авторы считают нужным изложить.
- К пакету обязательно должен быть приложен файл COPYING, содержащие текст (либо тексты) лицензий, под которой распространяется содержимое пакета.
- Рекомендуется включать в пакет (помещать в корневую директорию) прочие полезные документы:
  - INSTALL – руководство по сборке и инсталляции, описывающее необходимое окружение и набор инструментов, потенциальные проблемы портабильности между платформами;
  - AUTHORS – список авторов и участников проекта и их контакты;

Следование этим простым рекомендациям значительно упрощает и организует работу с программным пакетом.

2.2. Лицензирование исходного кода

Необходимые шаги для лицензирования исходного кода:

1. Каждый файл с исходным кодом программы должен содержать Copyright-строку следующего вида:

Copyright <год(ы)> <автор(ы)>. All rights reserved.

Годы выпуска релизов программы можно обозначать интервалом, либо перечислить через запятую. Список авторов может быть перечислен явно, либо авторы могут быть указаны косвенно, например, "<ProjectName> Authors". При этом, авторы и их контактная информация должна быть указана в файле AUTHORS в корневой директории пакета.

2. Каждый файл с исходным кодом программы должен включать текст, содержащий условия использования и указание на лицензию. Каждая известная лицензия предоставляет свой шаблон такого текста для включения в исходный код, примеры приведены в конце главы, а также по ссылкам [1] и [2].
3. Пакет должен содержать файл COPYING (альтернативное название LICENSE) с полным текстом используемой лицензии. В случае если лицензий несколько, допускается их помещение в один, либо в разные файлы, именованные как COPYING.LICENSE_1, COPYING.LICENSE_2 и т. д., где LICENSE_1 и LICENSE_2 – названия соответствующих лицензий.
4. Важно помнить, что каждый значимый файл содержащийся в пакете должен быть лицензирован. Это касается непосредственно исходных кодов, скриптов используемых для сборки и установки, документации, и прочего содержимого – например, научных данных.

Пример текста для включения в исходный код (для GPL):

Copyright (C) <year> <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 3 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, see http://www.gnu.org/licenses/

Пример текста для включения в исходный код (для Apache License):

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

Ссылки к разделу 2:

[1] http://www.gnu.org/licenses/gpl-howto.html

[2] http://www.apache.org/licenses/LICENSE-2.0.html

Полезные рекомендации по организации программного пакета (на английском языке):

[3] http://en.tldp.org/HOWTO/Software-Release-Practice-HOWTO/

Пример правильно оформленного программного пакета (программа «Hello World»):

[4] http://ftp.gnu.org/gnu/hello/

Раздел 3. Написание Makefile.

Этот раздел посвящен основным аспектам корректного создания make-файлов, и не является обучающим руководством.

make — утилита, автоматизирующая процесс преобразования файлов из одной формы в другую. Обычно, это компиляция исходного кода в объектные файлы и последующая сборка в исполняемые файлы или библиотеки. Утилита использует специальные make-файлы, в которых указаны зависимости файлов друг от друга и правила для их удовлетворения. На основе информации о времени последнего изменения каждого файла make определяет и запускает необходимые программы. [1] Wikipedia.

Зачастую, разработчики небольших проектов пренебрегают автоматизацией сборки и тестирования. Это является серьезным препятствием для создания любых установочных пакетов для репозиториев операционных систем. В большинстве случаев, для создания установочного пакета подходит любая распространенная система автоматизации сборки: **make, qmake, cmake, scons** и другие. Ниже приводятся рекомендации по основным характеристикам автоматизации сборки и установки на примере системы **make** и ее основе – **Make-файлах**:

- Сборка всего проекта должна осуществляться минимальным количеством команд: в самом простом случае, исполнением одной команды make без аргументов.
    - Если проект основан на **autotools,** то сборка должна выглядеть как **./configure && make**
    - Если проект основан на **qmake/cmake** или **аналоге**, то сборка должна осуществляться аналогично, двумя командами: например, **qmake && make**
- Установка (копирование файлов-результатов сборки в системные директории) должна также осуществляться единственной командой, исполненной с правами суперпользователя: **make install**.
- **Makefile** обязан поддерживать следующие операции:
    - **all** – сборка всего пакета (операция по умолчанию).
    - **clean** – удаление всех файлов. После выполнения команды **make clean** содержимое распакованного пакета должно совпадать с содержимым архива. Если процесс сборки конфигурируется с помощью **./configure**, должна поддерживаться операция **distclean**.
    - **install** – упомянутая выше установка результатов сборки
    - **uninstall** – деинсталляция результатов сборки.

- Путь для установки пакета обязан предваряться префиксом, который устанавливается переменной среды **$DESTDIR**, например:

prefix = $(DESTDIR)/usr

libdir = $(prefix)/lib

includedir = $(prefix)/include

docdir = $(prefix)/share/projectname

- Для установки требуется использовать **команду install** с обязательным параметром **“-p”** – перенос времени модификации исходных кодов на результирующие файлы.
- Внутренняя организация **makefile**: все цели, которые не создают файл, должны быть помечены как **.PHONY.**
- Если проект разбит на несколько модулей, например, основное дерево исходных кодов, тесты и примеры использования, то удобно организовывать **Make-файлы** в виде иерархической структуры: по одному на каждый блок проекта и один корневой. Если проект более сложен, то ручная поддержка **Make-файлов** может оказаться сложной и оптимальным будет использование более мощной системы: **CMake, autotools.**

Ссылки к разделу 3:

[1] http://ru.wikipedia.org/wiki/Make

[2] http://www.linux.org.ru/books/make.html