Вдохновившим меня
Кену Томпсону
и Деннису Ритчи

Предисловие

Unix — не столько операционная система, сколько история.

—Нил Стефенсон (Neal Stephenson)

Существует огромная разница между знаниями и опытом. Знания позволяют определить необходимые действия аналитическим путем, тогда как опыт делает верные действия рефлекторными, едва ли требующими осознанных размышлений вообще.

Целью книги является попытка преподнести читателям аспекты разработки программ в Unix, которые интуитивно известны экспертам данной операционной системы. Поэтому в данной книге, в отличие от большинства других книг о Unix, рассматривается меньше технических подробностей и больше вопросов *коллективной культуры*, как в явной, так и в скрытой ее формах, а также ее осознанные и неосознанные традиции. Данная книга не дает ответов на вопросы о том, "как сделать что-либо", она не является сборником документов how-to, скорее в ней собраны ответы на вопросы наподобие "почему это следует сделать" (why-to).

Подход why-to обладает большой практической важностью, поскольку слишком многие программы проектируются неудачно. Для большинства из них характерен большой размер, чрезвычайная сложность сопровождения и чрезмерные трудности при переносе на новые платформы или расширении, которое не было предусмотрено создавшими их программистами. Данные проблемы являются симптомами ошибочного проектирования. Авторы надеются, что эта книга позволит популяризировать некоторые относящиеся к Unix знания в области хорошего проектирования.

Книга разделена на четыре части: "Контекст", "Проектирование", "Реализация" и "Сообщество". В первой части ("Контекст") освещены философские и исторические аспекты, способствующие созданию основы и заинтересованности для восприятия последующего материала. Во второй части ("Проектирование") принципы философии Unix разворачиваются в более специфические рекомендации, касающиеся проектирования и реализации. В третьей части ("Реализация") основное внимание уделено программному обеспечению Unix. Четвертая часть ("Сообщество") касается межличностного взаимодействия и соглашений, которые делают Unix-культуру столь эффективной в своей области.

Поскольку данная книга посвящена коллективной культуре, автор с самого начала не планировал создавать ее в одиночестве. Читатели заметят, что текст включает в себя высказывания выдающихся Unix-разработчиков, основателей традиций Unix. Эти светила были приглашены прокомментировать и обсудить текст в ходе длительного процесса публичного рассмотрения книги.

Используя слово “мы”, автор не пытается показаться всеведущим, а старается отразить тот факт, что в книге предпринята попытка ясного выражения опыта всего сообщества.

Поскольку данная книга нацелена на передачу культуры, она включает в себя гораздо больше исторических фактов, фольклора и замечаний, чем обычная техническая книга. Наслаждайтесь: все это также является частью обучения Unix-программиста. Ни одна из исторических подробностей не является жизненно важной, но в совокупности все они существенны. Авторы полагают, что это делает книгу более интересной. Гораздо важнее понимание того, откуда произошла операционная система Unix и как она встала на путь своего развития. Это поможет развить интуитивное чувство Unix-стиля.

По той же причине авторы отказались от написания книги таким образом, как если бы история была завершенной. Читатели найдут большое количество ссылок относительно времени написания книги. Эти ссылки призваны обратить внимание читателя на то, что соответствующие факты могли устареть и должны быть перепроверены.

Кроме того, настоящая книга не является ни сборником уроков по языку C, ни руководством по командам и API-функциям операционной системы Unix. Она также не является справочником по программам *sed* или *yacc*, языкам Perl или Python, букварем для сетевого программиста или исчерпывающим описанием секретов системы X. Это не экскурс во внутреннее устройство или структуру операционной системы Unix. Указанные вопросы лучше рассмотрены в других книгах, и в тексте данной книги в соответствующих случаях имеются ссылки на них.

За пределами всех специфических технических вопросов Unix-культура обладает неписаной традицией, которая развивалась на протяжении миллионов человеко-лет[1] инженерной практики. Данная книга написана с верой в то, что понимание этой традиции и включение в свой инструментарий ее моделей поможет читателям стать лучшими программистами и проектировщиками.

Культура создается людьми, и традиционный путь изучения Unix-культуры состоит в постепенном ее восприятии через фольклор от членов сообщества. Эта книга не заменит передачи культуры в личном общении, но она способна ускорить этот процесс, позволяя изучить опыт других.

Для кого предназначена эта книга

Книга рекомендуется для опытных Unix-программистов, которые обучают начинающих разработчиков либо спорят с приверженцами других операционных систем и находят трудным ясное изложение преимуществ Unix-подхода.

Книгу стоит прочесть программистам на C, C++ или Java, имеющим опыт работы в других операционных системах и планирующим начать Unix-проект.

Книгу также следует прочесть пользователям Unix, как новичкам, так и пользователям со средним уровнем квалификации, но имеющим небольшой опыт разработки и желающим изучить методики проектирования эффективного программного обеспечения в этой операционной системе.

---

1 Три с половиной десятилетия между 1969 и 2003 гг. — это время исторической эволюции ОС Unix, воплотившей достижения более 50 млн. человеко-лет.

Материал книги также рекомендуется изучить тем, кто, не являясь Unix-программистом, осознает ценность Unix-традиций. Авторы верят в правоту этой точки зрения; Unix-философию можно использовать в других операционных системах. Поэтому в данной книге этим другим системам (в особенности системам, разработанным корпорацией Microsoft) уделено больше внимания, чем обычно в книгах по Unix. Если инструментарий и учебные примеры применимы к таким системам, в тексте имеются соответствующие замечания.

Данная книга рекомендуется для изучения разработчикам прикладных программ, рассматривающим платформы или методы реализации для крупного общецелевого или вертикального приложения. Таким специалистам книга поможет понять преимущества Unix в качестве платформы для разработки, а также Unix-традиций применительно к открытым исходным кодам как методу разработки.

В то же время читателям *не стоит* искать здесь подробности программирования на языке C или использования API-интерфейса ядра Unix. По этим темам имеется множество хороших книг, в частности: *"Advanced Programming in the Unix Environment"* **[81]** — классический труд по изучению Unix API, а также книга *"The Practice of Programming"* **[40]**, которая входит в перечень рекомендованной литературы для всех программистов на языке C (а в действительности рекомендуется всем программистам на всех языках).

Как использовать эту книгу

Данная книга одновременно является практической и философской. Некоторые ее части являются афористичными и общими, в других изучаются специфические примеры Unix-разработок. Рассмотрение общих принципов и афоризмов предваряется или завершается иллюстрирующими их примерами. Примеры взяты не из учебных программ, а из реально действующего кода, который используется ежедневно.

Авторы преднамеренно избегали наполнения книги длинными листингами или примерами файлов спецификаций, даже если во многих случаях это облегчало написание (а в некоторых, возможно, и чтение). В большинстве книг по программированию дано чрезмерное количество низкоуровневых деталей и примеров. Вместе с тем в них наблюдается недостаток высокоуровневого объяснения того, что в действительности происходит. Авторы данной книги предпочитают "заблуждаться в противоположном направлении".

Таким образом, несмотря на то, что читателям часто предлагается рассмотреть код или файлы спецификации, фактически в книгу включено сравнительно небольшое число таких примеров. Вместо них указаны ссылки на примеры, представленные на Web-ресурсах.

Усвоение данных примеров поможет превратить изученные принципы в полуинстинктивные практические знания. В идеальном случае данную книгу следует читать рядом с консолью работающей Unix-системы, имеющей удобный Web-браузер. Подойдет любая Unix-система, но уже установленные и немедленно доступные для изучения учебных примеров программы, вероятнее всего, будут находиться на какой-либо Linux-системе. Указатели в данной книге побуждают просматривать соответствующие страницы и экспериментировать.

Следует заметить: несмотря на все усилия, направленные на включение URL-адресов, которые должны оставаться стабильными и доступными, авторы не могут гарантировать их постоянную доступность. Если выяснится, что какая-либо процитированная ссылка устарела, рекомендуется, учитывая общий смысл, воспользоваться поиском словосочетаний в привычной поисковой машине в Web. Там, где это возможно, рядом с URL-адресами предложены способы поиска информации.

Большинство аббревиатур, использованных в данной книге, расшифровываются при первом использовании. Для удобства в приложении также приводится глоссарий.

Ссылки на дополнительную литературу обычно выполняются по номеру книги в списке (см. приложение B). Также даны нумерованные сноски на URL-адреса, которые могут затруднять чтение или предположительно являются ненадежными. Это же относится к примечаниям, историческим фактам и шуткам.

Для того чтобы сделать данную книгу более доступной для технически менее подготовленных читателей, она была предложена непрограммистам для прочтения и определения терминов, которые кажутся непонятными и одновременно необходимы для последовательного изложения. Также использовались сноски для определений элементарных понятий, которые вряд ли понадобятся опытным программистам.

Дополнительные источники информации

Конечно, тематика данной книги уже рассматривалась в некоторых периодических изданиях и нескольких книгах, написанных первыми разработчиками операционной системы Unix. Среди них выделяется и по праву считается классической книга *"The Unix Programming Environment"* **[39]** Кернигана (Kernighan) и Пайка (Pike). Однако в ней не рассматривается Internet и World Wide Web или новая волна интерпретируемых языков программирования, таких как Perl, Tcl и Python.

Работая над этой книгой, авторы внимательно изучали работу *"The Unix Philosophy"* **[26]** Майка Ганкарза (Mike Gancarz). Данная книга является выдающейся в своем роде, однако Ганкарз не пытается раскрыть полный спектр тем, которые следовало бы рассмотреть. Тем не менее, авторы выражают благодарность ее создателю за напоминание о том, что простейшие модели проектирования в Unix являются наиболее устойчивыми и удачными.

Книга *"The Pragmatic Programmer"* **[37]** отражает остроумную дискуссию о хорошей практике проектирования, которая относится к несколько другому, по сравнению с данной книгой, уровню искусства проектирования программ (в ней более подробно рассматриваются вопросы кодирования и меньше проблемы более высокого уровня). Философия ее автора базируется на опыте работы с Unix, а книга представляется отличным дополнением к данному изданию.

В книге *"The Practice of Programming"* **[40]** рассматривается несколько тех же положений, что и в книге *"The Pragmatic Programmer"*, но в аспекте Unix-традиции.

Наконец, авторы рекомендуют *"Zen Flesh, Zen Bones"* **[68]**, важную коллекцию основных источников Дзэн-буддиста. Ссылки на Дзэн включены в данную книгу, поскольку Дзэн предоставляет словарный запас для обозначения некоторых идей, которые оказываются весьма важными при проектировании программного обеспечения, но очень трудны для запоминания.

Соглашения, используемые в данной книге

Термин “UNIX” технически и юридически является торговой маркой организации “The Open Group”, и формально он должен использоваться только относительно сертифицированных операционных систем, прошедших сложные тесты на соответствие стандартам “The Open Group”. В данной книге термин “Unix” используется в более свободном, широко распространенном среди программистов смысле. Здесь термин “Unix” применяется для обозначения любой операционной системы (имеющей формальное название Unix или нет), которая либо происходит из кода, унаследованного от системы Unix компании Bell Labs, либо “написана в строгом приближении к потомкам этой системы”. В частности, Linux (из которой взято большинство примеров книги), согласно данному определению, является Unix-системой.

В настоящей книге используются соглашения справочной системы Unix (manual page) для выделения средств Unix с последующим указанием в скобках номера раздела справочной системы. Обычно это делается при первом упоминании команды, если требуется показать, что она принадлежит Unix. Например, запись “munger(1)” следует читать как “программа munger, описанная в разделе 1 (пользовательские средства) справочной системы Unix, в случае если она присутствует в данной системе”. Раздел 2 содержит справочные сведения о системных вызовах C, раздел 3 — о вызовах библиотеки C, раздел 5 посвящен форматам файлов и протоколам, в разделе 8 описаны средства системного администрирования. Другие разделы отличаются в зависимости от принадлежности к различным Unix-системам, но они не цитируются в данной книге. Для получения более подробных сведений следует в приглашении командной строки Unix ввести команду **`man 1 man`** (в более давних системах ветви System V Unix может потребоваться команда **`man -s 1 man`**).

Иногда в тексте упоминается какое-либо приложение Unix (такое как *Emacs*), название которого приводится без указания номера раздела справочной системы и начинается с прописной буквы. Таким образом указывается название, которое фактически представляет широко распространенное семейство Unix-программ с одинаковыми функциями, и в тексте описываются общие свойства всех программ данного семейства. Например, семейство *Emacs* включает в себя программу *xemacs*.

Далее в настоящей книге нередко встречается ссылка на методы “старой школы” и “новой школы”. Подобно рэп-музыке, новая школа возникла в 90-х годах прошлого века. В данном контексте новая школа связана с появлением языков написания сценариев (scripting language), графических пользовательских интерфейсов (Graphical User Interfaces — GUI), Unix-систем с открытым исходным кодом и Web-среды. Упоминание старой школы относится к периоду до 1990 года (и особенно до 1985 года), когда повсеместно применялись дорогостоящие (совместно используемые) компьютеры, частные Unix-системы, сценарии командного интерпретатора и программы на языке C. Данные различия стоит подчеркнуть, поскольку более дешевые машины с меньшими ограничениями памяти внесли значительные изменения в стиль Unix-программирования.

Учебные примеры

Многие книги по программированию основаны на “игрушечных” примерах, сконструированных специально для подтверждения точки зрения автора. В данном случае это не так. Учебные примеры, приведенные в данной книге, являются реальными, существующими блоками программного обеспечения, которое используется ежедневно. Приведем некоторые из главных примеров.

`cdrtools/xcdroast`

Данные отдельные проекты обычно используются вместе. Пакет `cdrtools` представляет собой набор CLI-инструментов для записи дисков CD-ROM. Информацию по данному пакету можно найти в Web с помощью поискового слова “cdrtools”. Приложение `xcdroast` — GUI-интерфейс для пакета cdrtools. Сайт проекта xcdroast доступен по адресу `<http://www.xcdroast.org>`.

`fetchmail`

Программа *fetchmail* получает почту с удаленных почтовых серверов с помощью почтовых протоколов POP3 или IMAP. См. домашнюю страницу fetchmail `<http://www.catb.org/~esr/fetchmail>` (поиск в Web с помощью ключевого слова “fetchmail”).

`GIMP`

GIMP (GNU Image Manipulation Program — GNU-программа для работы с графическими изображениями) полнофункциональная программа для создания и обработки изображений, которая способна осуществлять сложное редактирование множества различных графических форматов. Исходные коды программы доступны на домашней странице GIMP `<http://www.gimp.org/>` (или поиск в Web по слову “GIMP”).

mutt

Пользовательский почтовый агент *mutt* является лучшим среди современных текстовых Unix-агентов электронной почты, который известен благодаря хорошей поддержке MIME-форматов (Multipurpose Internet Mail Extensions — многоцелевые расширения почтового стандарта в Internet) и использованию таких средств безопасности, как PGP (Pretty Good Privacy) и GPG (GNU Privacy Guard). Исходный код продукта и исполняемые двоичные файлы доступны на сайте проекта Mutt `<http://www.mutt.org/>`.

`xmlto`

Команда *xmlto* преобразует DocBook-документы и другие XML-документы в различные форматы, включая HTML, текстовый формат и PostScript. Исходные коды и документация представлены на сайте проекта xmlto `<http://www.cyberelk.net/tim/xmlto/>`.

В целях сокращения кода, который необходимо прочесть пользователю для понимания примеров, авторы старались подбирать те из них, которые могут использоваться несколько раз, в идеальном случае иллюстрируя несколько различных принципов и практических рекомендаций по проектированию. По той же причине многие примеры взяты из проектов автора. Их не следует рассматривать как наилучшие из возможных примеров, просто автор находит их достаточно известными для использования во многих демонстрационных целях.

Авторские благодарности

Приглашенные помощники Кен Арнольд (Ken Arnold), Стивен М. Белловин (Steven M. Bellovin), Стюарт Фельдман (Stuart Feldman), Джим Геттис (Jim Gettys), Стив Джонсон (Steve Johnson), Брайан Керниган (Brian Kernighan), Дэвид Корн (David Korn), Майк Леск (Mike Lesk), Дуг Макилрой (Doug McIlroy), Маршал Кирк Маккьюзик (Marshall Kirk McKusick), Кит Паккард (Keith Packard), Генри Спенсер (Henry Spencer) и Кен Томпсон (Ken Thompson) внесли крупный вклад в создание данной книги. В частности, Дуг Макилрой в очередной раз продемонстрировал свою преданность идеям превосходного качества, которые он привнес в управление еще первой исследовательской группой Unix тридцать лет назад.

Особую благодарность автор выражает Робу Лэндли и своей жене Кэтрин Реймонд, которые строку за строкой редактировали черновик рукописи. Внимательные и тонкие комментарии Роба вдохновили меня на создание нескольких полных глав, кроме того, его замечания во многом определили нынешнюю организацию книги и круг рассмотренных в ней тем. Если бы он написал весь тот текст, который он заставил меня улучшить, то я должен был бы называть его соавтором. Кэти играла роль моей тестовой аудитории, представляя читателей, которые не знакомы с техникой программирования.

В данную книгу вошли полезные моменты, выявленные в ходе обсуждения с другими специалистами в течение пяти лет ее написания. Марк М. Миллер (Mark M. Miller) способствовал автору в освещении темы параллельных процессов. Джон Коуэн (John Cowan) дал несколько ценных рекомендаций, касающихся моделей проектирования интерфейсов и создал черновики учебных примеров программы *wily* и системы VM/CMS. Джеф Раскин (Jef Raskin) продемонстрировал происхождение правила наименьшей неожиданности. Группа системной архитектуры UIUC (UIUC System Architecture Group) также внесла свои полезные дополнения. Рецензия членов группы вдохновила автора на написание разделов *"Что в Unix делается неверно"* и *"Гибкость на всех уровнях"*. Рассел Дж. Нельсон (Russell J. Nelson) дополнил материал по образованию цепей Бернштайна в главе 7. Джей Мэйнард (Jay Maynard) значительно помог в разработке учебных примеров MVS в главе 3. Лес Хаттон (Les Hatton) предоставил множество полезных комментариев, касающихся главы *"Языки программирования: C или не C?"* и побудил автора к созданию раздела *"Инкапсуляция и оптимальный размер модуля"* в главе 4. Дэвид А. Вилер (David A. Wheeler) внес множество проницательных критических замечаний и некоторые материалы по учебным примерам, особенно в части *"Проектирование"*. Расс Кокс (Russ Cox) помог развить обзор системы Plan 9. Деннис Ритчи (Dennis Ritchie) корректировал некоторые исторические моменты, касающиеся языка C.

В период публичного рассмотрения книги с января по июнь 2003 года сотни Unix-программистов (слишком много, чтобы перечислить здесь их имена), вносили свои советы и комментарии. Как всегда, процесс открытия равноправного обсуждения посредством Web одновременно является крайне трудным и чрезвычайно полезным. Конечно, как всегда, всю ответственность за возможные ошибки автор принимает на себя.

На стиль изложения и некоторые вопросы, рассмотренные в данной книге, оказала определенное влияние известная концепция о моделях проектирования. Действительно, автор размышлял над названием *"Модели проектирования в Unix" (Unix Design Patterns)*. Однако оно было отклонено, поскольку автор был не согласен с некоторыми безоговорочными догмами данной школы и не испытывал необходимости использовать весь его формальный аппарат или принимать культурный багаж. Тем не менее, работы[2] Кристофера Александера (Christopher Alexander) (особенно *"The Timeless Way of Building"* и *"A Pattern Language"*) оказали свое влияние на авторский подход. Автор считает своим долгом выразить огромную благодарность Группе четырех (Gang of Four) и другим членам их школы за демонстрацию того, каким образом можно использовать работы Александра в отношении проектирования на высоком уровне, не оперируя полностью неясными и бесполезными общими фразами. Заинтересованным читателям в качестве введения в тему моделей проектирования рекомендуется изучить книгу *"Design Patterns: Elements of Reusable Object-Oriented Software"* **[24]**.

Название данной книги, несомненно, ассоциируется с *"Искусством программирования"* Дональда Кнута (Donald Knuth). Несмотря на то, что Кнут не связан с традициями Unix, он оказывает влияние на всех нас.

Редакторы с глубоким видением текста и богатым воображением встречаются не так часто, как хотелось бы. Один из них — Марк Тауб (Mark Taub), который смог оценить достоинства приостановленного проекта и деликатно подтолкнул автора к окончанию работы. Хорошим чувством прозаического стиля и достаточными способностями улучшить написанное отличается и Мэри Лау Нор (Mary Lou Nohr). Джерри Вотта (Jerry Votta) уловил авторскую идею обложки и сделал ее лучше, чем можно было представить. Весь коллектив издательства Addison-Wesley заслуживает высокой оценки за осуществление редактирования и производственного процесса, а также за терпимость к причудам автора, касавшимся не только текста, но и внешнего дизайна книги, оформления и маркетинга.

---

2 Оценка его работы со ссылками на Web-версии значимых частей находится на странице "Записки о Кристофере Александре" <http://http://www.math.utsa.edu/~salingar/Chris.text.html>.