Руслан Богатырев

http://rbogatyrev.livejournal.com/2007/06/29/

1. Как зарождалась свобода открытых исходных текстов

В 2003 г. в издательстве *Addison-Wesley* вышла книга Эрика Реймонда “The Art of *UNIX* Programming”, которая в переводе на русский язык появилась в 2005 г. под названием “Искусство программирования для *Unix*”.

Реймонд, будучи отцом-основателем и главным идеологом движения открытых исходных текстов (Open Source Initiative), решил восполнить пробелы по летописи UNIX и нарисовать грандиозное полотно, привлекая комментарии всех более-менее значимых фигур UNIX-культуры во главе с автором системы — Кеном Томпсоном. На книгу у него ушло пять лет. Реймонд постарался представить и отдельные нотки критики *UNIX* (в разумных, с его точки зрения, пределах). Конечно, не без лукавства и не без умалчивания, но всему своё время — мы ещё обсудим и технические моменты.

Пока же в центре нашего внимания будет свобода, свобода программного обеспечения, которая взлетела на крыльях *UNIX* и привела нас к нынешнему апофеозу в лице одного из клонов *UNIX* с именем *Linux* (сразу хочу остудить горячие головы, возмутившиеся этой фразой — ядро *Linux*, лежащее в основе пяти с лишним сотен современных дистрибутивов, и формирует совместно с системной “обвязкой” то, что сам Эрик Реймонд в этой книге признает клоном *UNIX)*.

Намеренно сосредоточусь пока на точке зрения одного Эрика Реймонда. В последующих разделах мы затронем взгляды и других "фигурантов", включая Столлмена и Торвальдса.

В эпоху мэйнфреймов и мини-ЭВМ, пока программы не отчуждались от компьютеров, в отношении обмена исходными текстами особых заморочек не было. Программы воспринимались как сопутствующая часть железа, как документация. С появлением минимашин, микрокомпьютеров и персоналок резко расширился круг потребителей, появились независимые производители программ. Одновременно с этим возникла потребность прятать исходные тексты, затрудняя процесс модификации и сопряжения по необнародованным интерфейсам.

Программы стали продаваться как самостоятельные продукты. О деятельности Билла Гейтса на поприще микрокомпьютеров и о первых трансляторах *Microsoft*, думаю, всем хорошо известно. Но и в области больших и средних машин произошли серьёзные подвижки. Первая *UNIX*-компания (*Santa Cruz Operation, SCO*) начала свою деятельность в 1978 г., когда стала продавать первый коммерческий компилятор *Cu* (Whitesmiths). Цены на важные продукты были заоблачными.

При этом, по иронии судьбы, сама *AT&T*, в исследовательских лабораториях которой была создана *UNIX*, не могла продавать свой софт. По соглашению об урегулировании антитрастового дела 1958 г. корпорации *AT&T* запрещалось входить в компьютерный бизнес. Систему *UNIX* от *Bell Labs* нельзя было превращать в продукт. Реймонд пишет: "Действительно, в соответствии с положениями соглашения, *Bell Labs* должна была лицензировать свою нетелефонную деятельность всем желающим. Кен Томпсон без огласки начал отвечать на запросы, отправляя ленты и дисковые пакеты, каждый из которых, согласно легенде, подписывался "с любовью, Кен" (love, ken)".

Заметная "утечка" и ветвление *UNIX* (на коммерческую *UNIX System V* и некоммерческую *BSD UNIX*) начались с университета Калифорнии в Беркли (Berkeley). В 1975–1976 гг. Кен Томпсон преподавал в университете в рамках своего академического отпуска. В 1977 г. Билл Джой, будущий основатель и вице-президент *Sun Microsystems*, создал первую *BSD*-версию (*Berkeley Software Distribution*). С этого момента распространение *UNIX* по линии университетов США заметно активизировалось, а Беркли стал центром разработки и, если так можно выразиться, центром компетенции *UNIX*.

Основной прорыв *UNIX* в мир программного обеспечения произошёл благодаря Министерству обороны США, когда в 1980 г. *DARPA* (Агентство передовых исследований) выбрало *UNIX* для реализации протоколов *TCP/IP* на компьютерах *DEC VAX*. Выбрало в основном из-за проблем с корпорацией *DEC* (проблемы с модификацией её ОС *VAX/VMS*) и того, что *BSD* была свободна и доступна. Как пишет Реймонд, "это была, несомненно, наиболее важная поворотная точка в истории *UNIX* с момента её появления". В 1983 г. поддержка *TCP/IP* под *BSD* заработала.

Реймонд пишет: "Официально лицензия на исходный текст *UNIX System III* стоила 40 тыс. долл., однако *Bell Labs* закрывала глаза на рост количества распространяемых нелегальных лент с *Bell Labs UNIX*, а университеты продолжали обмениваться текстами с *Bell Labs*".

В 1982 г. была основана *Sun Microsystems*, которая взяла за основу "железо" разработки *Стэнфордского университета* и *BSD* в качестве базовой ОС. Система *BSD* стала подвергаться перековке и превращаться в коммерческий продукт под брендом *Sun*.

UNIX шагнул и в мир персоналок. К 1983 г. было не менее шести *UNIX*-подобных ОС для *IBM PC*: *uNETix, Venix, Coherent, QNX, Idris* и вариант, поддерживаемый на плате *Sritek PC*. Но не было вариантов ни *AT&T System V*, ни *BSD*. Свой клон *UNIX*, заплатив за лицензирование *AT&T*, выпустила и *Microsoft* (он назывался *XENIX*, 1980-1989), причём вплоть до 1992 г. использовала его для внутренних разработок (на компьютерах *Sun* и на *DEC VAX*). В 1987 г. в обмен на 25% акций *SCO* корпорация *Microsoft* переуступила права на *XENIX* именно компании *SCO* (тем, кто следил за тяжбой между *IBM* и *SCO* в отношении *Linux*, стоит иметь это в виду).

В 1983 г. произошло важное событие, причём не без участия *Sun*: Министерство юстиции США выиграло второй антитрастовый процесс против *AT&T* и раздробило корпорацию, что освободило ту от соглашения 1958 г. и позволило приступить к коммерциализации *UNIX System V*. Как пишет Реймонд, "этот шаг почти уничтожил *UNIX*". Он продолжает: "Никто из нас в то время не осознавал, что превращение *UNIX* в коммерческий продукт разрушит свободный обмен исходным текстом, который дал столько жизненной силы развивающейся системе. Не зная другой модели, кроме секретности для накопления прибыли от программного обеспечения и кроме централизованного управления разработкой коммерческого продукта, *AT&T* прекратила распространение исходных текстов. Нелегальные ленты с *UNIX* стали намного менее интересными, так как их использование могло повлечь угрозу судебного преследования. Интеллектуальный вклад от университетов начал иссякать".

Начались затяжные междоусобные *UNIX*-войны. А тем временем *Microsoft* спокойно набирала силу, ожидая удобного случая, чтобы расправиться с пригревшей её у себя на груди корпорацией *IBM*. Голубой гигант, как величественно именуют *IBM*, намеренно сдерживал рост производительности ПК, чтобы сохранить на всей своей линейке единое соотношение "цена-производительность". Но *Compaq* рванула вперёд с выпуском PC-клона на базе нового 32-разрядного процессора *Intel 80386*. Создалась ситуация, когда наличие на такой машине бесплатного *UNIX* тут же выводило компьютер в разряд рабочих станций и давила по ценам снизу на продукцию *Sun*.

Тем временем, в середине 1980-х, когда эра больших и средних машин стала клониться к закату своей славы, потребовалась среда, которая вобрала бы в себя старую гвардию программистов. Для неё юнцы персоналок воспринимались как выскочки, стремящиеся побольше урвать в мутной водичке всеобщего PC-ажиотажа.

Реймонд пишет: "Смерть хакерской культуры, развивавшейся вокруг компьютеров *DEC PDP-10* разработки *MIT AI Lab*, побудила программиста Ричарда Столлмена к началу создания проекта GNU, полностью свободного клона *UNIX*". В 1985 г. Столлмен опубликовал Манифест *GNU*, где изложил взгляды на свободное программное обеспечение (*Free Software*). Из-под его пера появилась первая лицензия *GPL*. На бранном поле борцов за свободу столкнулись два лагеря: *GNU* и *BSD*. Вот как это описывает Реймонд: "До 1995 г. наиболее серьёзное сопротивление плану GNU оказывали разработчики *BSD*. Приверженцы *BSD*, которые помнили, что они написали свободно распространяемое и модифицируемое программное обеспечение ещё за годы до манифеста Столлмена, отвергали претензии проекта *GNU* на историческое и идеологическое главенство. Особенно они возражали против передающейся, или "вирусной", собственности *GPL*, предлагая *BSD*-лицензию как "более свободную", поскольку она содержала меньше ограничений на повторное использование кода. Это не помогало проекту Столлмена, и его попытки создания центральной части системы провалились, хотя его Фонд свободного ПО (*Free Software Foundation*) выпустил

большую часть полного программного инструментария. Спустя 10 лет после учреждения проекта *GNU*, ядро *GNU* всё еще не появилось… После 1995 г. спор вокруг идеологии Столлмена принял несколько иной оборот. Её оппозиция стала ассоциироваться с именем Линуса Торвальдса, а также автора этой книги".

Ситуацию усугубляло то, что Столлмен и его команда в проекте *GNU* заковырялись с реализацией ядра (*Hurd*), которое планировали выпустить аж в 1985 г. Годы шли, а его всё не было.

Акулы бизнеса, конечно же, не дремали. С выходом *UNIX* из недр *AT&T Bell Labs* появились предпосылки унификации ОС на разных аппаратных архитектурах. Но довольно быстро бесплатность и открытость была обращена лидерами рынка в свою пользу: они формировали собственные коммерческие клоны UNIX. Начался новый виток противостояния. В 1988 г. *IBM, DEC, Hewlett-Packard* и др. учредили Фонд открытого программного обеспечения (*OSF, Open Software Foundation*) и развязали войну против *AT&T* и *Sun*. Это было не открытое программное обеспечение (как можно подумать из названия), а унификация стандартов в области *UNIX* с целью недопустить поражения альянса на этом рынке. Организатором был Армандо Стеттнер (Armando Stettner) из *DEC*. Основателями стали крупнейшие производители компьютеров, которых назвали “банда семи” (Gang of Seven): это американские *IBM, Digital Equipment Corporation (DEC)*, *Hewlett-Packard, Apollo Computers,* немецкие *Nixdorf Computer, Siemens AG* и французская *Groupe Bull*. Чуть позже к ним примкнули голландская *Philips* и японская *Hitachi*. Альянс *AT&T* с *Sun* стал продвигать UNIX *System V Release 4*, а *OSF* — разумеется, *BSD*. В 1996 г. *OSF* слилась с консорциумом *X/Open* и образовала известную ныне The *Open Group*, в ведении которой находятся стандартизация *UNIX* (включая *POSIX, CORBA, COE* и т.д.)

Весной 1990 г. с выпуском *Windows 3.0* началась новая эра. *Microsoft* ударила в спину ничего не подозревавшую *IBM*, утопив *OS/2* в своей саботажной политике. В том же 1990 г. Уильям Джойлитц опубликовал серию статей о переносе *BSD* на процессор *Intel 80386*. Однако проект *386BSD* сильно тормознулся, когда Джойлитц, возмущённый тем, что спонсоры проекта не разрешили выпускать исходные тексты в свободном виде, вышел из проекта.

Как бы то ни было, создались весьма благоприятные условия, когда появление на ПК бесплатного *UNIX* могло изменить расклад сил. Так и произошло. Линус Торвальдс занялся разработкой собственного ядра под *Intel 80386* для *GNU*-обвязки. Ему хотелось работать с *UNIX*. Реймонд пишет: "Торвальдс также отмечал, что если бы он знал о проекте *BSD UNIX*, то скорее присоединился бы к нему, чем создавал бы собственный проект. Однако проект 386BSD не распространялся до начала 1992 г. – несколько месяцев спустя после появления первой версии *Linux*".

Яростное противостояние гигантов привело к быстрому росту империи *Microsoft* и захвату ею ключевых высот. Реймонд отмечает: "Годы с 1989 по 1993 были самыми мрачными в истории *UNIX*. Выяснилось, что мечты всего *UNIX*-

сообщества провалились. Междоусобная вражда чрезвычайно ослабила *UNIX*-индустрию, почти лишила её возможности противостоять *Microsoft*. Элегантные процессоры компании *Motorola*, которые предпочитали *UNIX*-программисты, проиграли неказистым, но недорогим процессорам *Intel*. Проект *GNU* оказался неспособен выпустить свободное ядро *UNIX*, что было обещано ещё в 1985 г., и после нескольких лет отговорок доверие к нему стало падать. PC-технология была безжалостно превращена в коммерческий продукт… Приверженцы *UNIX* с опозданием поняли, что прежняя монополия *IBM* сменилась новой монополией *Microsoft*, а плохо сконструированное программное обеспечение *Microsoft* всё больше заполняло рынок".

В 1993-1994 гг. начался лавинообразный рост *Интернет*, где *UNIX* с открытыми исходными текстами и поддержкой *TCP/IP* был крайне востребован. Именно это привело к взлёту *BSD* и *Linux*. Тут возник ещё один немаловажный момент, который помог *Linux*. Компания *BSDI*, развивавшая проект Джойлитца, попала под судебное преследование со стороны *AT&T*. В результате большая часть ключевых разработчиков *BSD* переметнулась в лагерь *Linux*. В 1995 г. *Linux* заполучила мощный катализатор своего продвижения в массы через *Интернет*: это веб-сервер *Apache*.

“Спаразитировав” на клиентской базе *Minix* и добившись неплохого технического оснащения, самоделка Торвальдса стала выходить из тени. Реймонд пишет: “Единственное, чего не предложил Торвальдс, это новая идеология — новый рациональный или генеративный миф хакерского движения, позитивное суждение, заменяющее враждебность Столлмена к интеллектуальной собственности программой, более привлекательной для людей как внутри, так и вне хакерской культуры.

Автор данной книги невольно восполнил этот недостаток в 1997 г., пытаясь понять, почему Linux-разработка несколько лет назад не была дезорганизована”. Здесь Реймонд явно лукавит. Очевидно он увидел блестящую возможность убрать со своей дороги сильного конкурента – Столлмена во главе с *GNU*, воспользовавшись успехом *Linux*. За счёт более благообразного вида свободного ПО переключить на новое движение внимание бизнес-сообщества и похоронить, тем самым, бренды *Free Software* и *GNU*. Формальный повод был простой – перегруженное смыслом слово “free” (в пер. с англ. — свободный, вольный, независимый, доступный, добровольный, бесплатный). Оно вводило двусмыслицу. Люди воспринимали “свободный” как “бесплатный”, что было далеко от истины. Свобода по Столлмену – это свобода смотреть как сделано, разрешать переделывать и использовать в своих работах, но сохраняя такую “свободу” для получившегося ПО. *Free Software* через *Free Software Definition* подразумевает четыре свободы: свободу использования, изучения/адаптации, распространения оригинала, распространения производных версий. Предусматривает обязательное наличие исходных текстов.

Переворот произошёл 27 мая 1997 г. Именно в этот день на *Linux Kongress* в Германии 30-летний Реймонд провозгласил свой манифест “Собор и базар” ("The

Cathedral and the Bazaar"). Ход был достаточно простой: он взял в качестве примера для разбора свою программу fetchmail и подвёл под её разработку идеологический фундамент: противопоставление закрытой централизованной разработки (“собор”) и свободной децентрализованной разработки (“базар”), опираясь на сравнение, приведённое в известном бестселлере Фредерика Брукса “Мифический человеко-месяц, или как создаются программные системы” (там, кто не помнит, фигурировал величественный Реймский собор). Что требовалось для завлечения масс? Правильно: показать, что классик Брукс с его аристократическими подходами пасует перед распределённой демократической разработкой в свободном *Интернет*-сообществе. И обосновать, что *Linux* — тот самый феномен, который вызван временем и требует новых подходов.

Для пущей убедительности Реймонд сослался на анархические воззрения князя Петра Алексеевича Кропоткина (к ним мы ещё не раз вернёмся), которые довольно популярны на Западе. В контексте того, что Нильс Торвальдс (отец Линуса) был активным членом компартии Финляндии, — это было весьма уместно. Образ скромного европейского студента-очкарика, из тех, кого зовут “ботаниками”, получал выигрышный духовный облик пламенного революционера, который во имя свободы человечества от порабощения компьютерными империями готов безвозмездно жертвовать своим трудом. Итак, Реймонд предложил миру взглянуть на альтернативу: “Закон Линуса” против “Закона Брукса”.

В заключение своего манифеста Реймонд говорит о реальных шагах бизнес-сообщества, которое уловило-таки сигнал, посланный ему Эриком: “Очень странно осознавать, что ты помогал вершиться Истории... 22 января 1998 г., приблизительно через семь месяцев после того как я впервые опубликовал эту статью, *Netscape Communications* объявила о своих планах сделать открытыми исходные тексты *Netscape Communicator*. Однако, я и представить себе не мог, что предшествовало этому объявлению. Эрик Ханн, исполнительный вице-президент и глава технологического отдела *Netscape*'а написал мне: "От имени всех членов корпорации, я хочу поблагодарить Вас за то, что вы помогли нам понять эту проблему. Ваши убеждения и Ваши статьи оказались наиболее вескими доводами в пользу принятия этого решения". На следующей неделе я вылетел в *Кремниевую долину* для однодневной стратегической конференции с исполнительными и техническими сотрудниками *Netscape'*а. Мы вместе разработали лицензию и стратегию выпусков релизов исходников *Netscape'*а, а также составили планы по внесению положительных вкладов в *OpenSource*-сообщество”.

Месяц спустя, в феврале 1998 г. Эрик Реймонд и Брюс Перенс (бывший лидер проекта *Debian GNU/Linux* и инициатор проекта стандартизации *Linux Standard Base*) основали некоммерческую организацию *Open Source Initiative*, которая дала крышу новому движению. А сам Перенс сформулировал требования к OpenSource-программам, которые вылились в 10 пунктов (*Open Source Definition*), наследовавших и уточнявших принципы Столлмена.

“А что же Ричард Столлмен и движение свободного программного обеспечения?

— пишет Реймонд. — Понятие “открытый исходный текст” было явно предусмотрено для замены понятия “свободного программного обеспечения”, которое предпочитал Столлмен. Он полусерьёзно воспринимал данное понятие, а впоследствии отверг его на том основании, что оно не способно представить моральную позицию, которая была центральной с его точки зрения. С тех пор движение свободного программного обеспечения настаивает на своей обособленности от “открытого исходного текста”, создавая, вероятно, наиболее значительный политический раскол в хакерской культуре последних лет”.

Довольно быстро лидеры рынка, противостоящие *Microsoft*, смекнули, что открытость *Linux* — это замечательное средство демпинга и передела рынка. *Novell, IBM, Sun, Hewlett-Packard (HP)* стали вкладывать серьёзные финансовые средства и предоставлять свои мощные возможности, в том числе и маркетинговые, для раскрутки *Linux*. Развитие ядра получило сильную подпитку, включая и передачу новых технологий. А сама операционная система *GNU/Linux*, из которой по дороге изъяли бренд *GNU*, была превращена в конструктор по принципу клонирования PC. За счёт этого удалось получить новые каналы сбыта для "убийцы *Microsoft*" через компании, которые взялись формировать из этого конструктора свои варианты (дистрибутивы). Эти варианты во многих случаях стали коммерческими. Новая бизнес-модель заработала.

2. **Разные взгляды на дух свободы программного обеспечения**

Общественные движения, внешне близкие по анархическому духу свободы, — *Free Software* и *Open Source* — возникли по нескольким причинам, из которых можно выделить:

- социальные;
- технологические;
- маркетинговые.

Если первые положены во главу угла движения *Free Software* во главе с Ричардом Столлменом, то вторые подаются автором ядра *Linux* Линусом Торвальдсом как свободный контроль над технологиями. В действительности, ключевой аспект — маркетинговый. Это прекрасно понял третий участник — Эрик Реймонд — и вовремя подсуетился.

2.1. **Социальные аспекты**

Социальный момент был особенно ярко выражен в предшественнике и конкуренте *Open Source* — движении *Free Software*. Здесь на первый план выходит социальный протест против диктата ведущих компаний, борьба за свободу изучать, модифицировать ПО и распространять его дальше.

В одном из своих интервью Столлмен подробно раскрыл мотивы борьбы: “*BSD* возникла в 1984 г. как модифицированная версия *AT&T UNIX*, но она не являлась

свободным ПО. Вам требовалось предъявлять лицензию на исходники *AT&T* (соглашение о неразглашении), чтобы получить копию BSD. В 1985 и 1986 гг. я посетил разработчиков *BSD* и предложил им отделить свой код от кода AT&T, чтобы они могли выпускать свои программы как свободное ПО. После этого они начали работать в данном направлении, а ещё через несколько лет решили переписать и оставшуюся часть системы, чтобы выпускать полностью свободную версию. Первые свободные системы BSD были доступны с 1994 г. Они не входят в *GNU*, но своим существованием во многом ему обязаны".

Вот как Столлмен комментирует непростые взаимоотношения с *Open Source*: "*Free Software Movement*, которое мы организовали в 1983–1984 гг., имело идеалистические цели: пользователь должен иметь право изучать, контролировать и модифицировать своё ПО. Мы разработали операционную систему GNU, с помощью которой можно жить свободно. Чтобы достичь цели, нам пришлось создать полностью свободное ПО. В 1990 г. свободное ПО приобрело репутацию мощного и надёжного. Миллионы пользователей тогда на практике ощутили реальную пользу свободного ПО, но они не задумывались о самой свободе, не обсуждали её друг с другом. В 1998 г. некоторые из них организовали *Open Source Movement*. Это движение преследует те же цели, что и мы, но оно не настаивает на них. *Open Source Movement* даёт возможность разрабатывать технически совершенное ПО, но не как предмет свободы и социального стиля жизни. Поэтому сегодня существуют два политических движения в обществе свободного ПО: *Free Software Movement* и *Open Source Movement*. Мы не считаем, что более молодое движение — наш враг. Враг — частное ПО. Однако мы видим в *Open Source Movement* своего конкурента, потому что оно поощряет пользователей не придавать значения этическим аспектам *Free Software Movement*".

Что же их сближает? Столлмен отмечает: "Свобода так же важна в бизнесе, как и в других областях человеческой деятельности. Если ПО является свободным, то люди могут контролировать его использование. Если же ПО является частным (коммерческим), то его контролирует владелец-разработчик, он также контролирует то, что вы делаете на компьютере... Поддержка свободного ПО — свободный рынок. Поддержка несвободного (коммерческого) ПО обычно монополизирована. Свободный рынок способен удовлетворять потребности пользователей намного лучше, нежели монополия. Поэтому за свои деньги вы сможете получать лучшую поддержку именно для свободного ПО".

И всё же Столлмен не скрывает социально-политической подоплёки своей позиции. В другом интервью он говорит: "Закрытое ПО неэтично, потому что оно лишает пользователя базовых свобод — контролировать свой собственный компьютер и сотрудничать с другими пользователями. Оно может быть также низкокачественным или небезопасным, но это вторичные вопросы. Я откажусь от него, даже если оно лучшее в мире, просто потому, что слишком высоко ценю свою свободу, чтобы отказаться от неё ради такого ПО".

2.2. Технологические аспекты

Технологический мотив в большей степени превалирует в *Open Source*. Его идеологи и сторонники считают, что открытые исходные тексты дают возможность свободно обмениваться технологиями и создают более высокое качество.

В своей книге “Just for Fun. Рассказ нечаянного революционера” (2002) Торвальдс пишет: "Модель открытых исходников утвердилась не за счёт идеологии. Она начала привлекать внимание, когда стало очевидно, что это лучший метод разработки и усовершенствования технологии высочайшего качества. Теперь эта модель завоёвывает рынок, что ещё больше укрепляет её авторитет. Оказалось, что можно создавать компании для оказания разнообразных дополнительных услуг или использовать открытые исходники для популяризации технологий. Денежный поток — очень убедительный аргумент”.

Торвальдс недвусмысленно отмечает, в чём расходятся его взгляды со взглядами Столлмена: “Ричард Столлмен хочет, чтобы всё было общедоступно. Для него это вопрос политический, и он готов биться, чтобы с помощью *GPL* перевести всё в открытый доступ. Он не допускает других возможностей. Я же, честно говоря, сделал исходники *Linux* открытыми вовсе не из-за таких высоких соображений. Мне нужна была обратная связь. И потом, именно так действовали на заре компьютерной эры, когда основные разработки выполнялись в университетах и оборонных учреждениях. В итоге всё было совершенно открыто. Код предоставлялся любому университету по его просьбе. А вот Ричард — когда его отлучили от его любимых проектов — стал первым принципиальным сторонником открытых исходников. Да, можно получить огромные преимущества, раскрыв миру свою технологию и сделав её доступной на тех же условиях, что *Linux* и множество других открытий. Чтобы получить представление об этих преимуществах, достаточно просто бросить беглый взгляд на сравнительно низкое качество всех закрытых программных продуктов. *GPL* и модель открытых исходников позволяет создавать лучшие технологии. Вот и всё. Кроме того, они не позволяют утаить технологию и гарантируют, что каждый заинтересованный может принять участие в её разработке. Это важный момент. Столлмена, которому нужно поставить памятник за создание *GPL*, к борьбе за открытые исходники побудило в первую очередь то, что его лишили возможности работать над рядом интересных проектов, когда они перешли из открытого мира *Массачусетского технологического института (MIT)* в частную корпоративную среду. Самым примечательным таким проектом была *LISP*-машина. *LISP* возник в рамках исследований по искусственному интеллекту. Потом, как часто бывает, разработка показалась перспективной и кто-то решил создать специальную компанию, чтобы зарабатывать на ней деньги. В университетах это обычное дело. Однако Ричард не занимался коммерцией, поэтому, когда в 1981 г. *LISP* стал коммерческим проектом в рамках компании под названием *Symbolics*, он оказался за бортом. Усугубило положение то, что *Symbolics* переманила на работу многих его коллег по лаборатории искусственного интеллекта. И в такую ситуацию он попадал неоднократно. Насколько я понимаю, его тяга к открытым исходникам

объясняется в первую очередь не борьбой против коммерциализации, а борьбой против исключения. Для него открытые исходники — это возможность не остаться в стороне. Возможность продолжать работу над проектом независимо от того, стал ли он коммерческим".

Что характерно: сочувствуя Столлмену и вскрывая мотивацию его недовольства, Линус в книге про *Linux* очень красочно, с прибаутками рассказывает о своей роли, но почему-то забывает о тех, кто работал с ним рука об руку и сделал никак не меньше. О тех, кто незаслуженно попал в тень "великого Торвальдса". Я имею в виду программиста #2 в мире *Linux*, англичанина Алана Кокса (Alan Cox). Видимо факт отсутствия дружбы с "возвеличенным императором свободного мира" уже автоматически ведёт к забвению.

Организационно-технологический аспект в устах Эрика Реймонда в "Соборе и базаре" (1997) выливается в несколько пунктов. Главным из них он называет децентрализованный характер разработки при наличии у всех участников свободного доступа к исходным текстам. Пресловутый "базар". При этом отмечает зону эффективности "базарной" модели разработки. Это наличие "печки", от которой и вокруг которой все танцуют: "Абсолютно ясно, что никто в одиночку не может с нуля создать программу, опираясь на "базарную" модель. Один человек вполне в состоянии тестировать, отлаживать и совершенствовать программу, придерживаясь такого подхода, но будет крайне сложно начать реализацию проекта в "базарном" режиме. Так не делал Линус. Так не делал и я. Для зарождения сообщества разработчиков требуется работоспособный и готовый к тестированию продукт".

Итак, по Торвальдсу и Реймонду открытые исходники дают лучшее качество (всё открыто, много глаз, сплошная свобода). Типично рекламный лозунг. Любой здравомыслящий специалист поймёт, что это очень натянутое утверждение.

А что же Столлмен? Тот говорит прямо, без обиняков: "Но в отличие от некоторых, я никогда не говорил, что основной причиной перехода к свободному ПО является его большая надёжность и более широкие возможности. Так говорят люди из Open Source Movement. Мы не разделяем этого мнения". Это одна из ключевых точек расхождения взглядов борцов за "свободу" и "независимость".

2.3. **Маркетинговые аспекты**

Эта тема, по понятным причинам, не педалируется противборствующими сторонами борцов за свободу. Обе они не отрицают взимания платы за исходные тексты и коммерциализацию "свободы". Очевидно и то, что при прочих равных свободное распространение в бесплатном или крайне дешёвом виде — особая форма демпинга, позволяющая простыми усилиями захватить те или иные сектора рынка. О том, за счёт чего сумел стремительно взлететь *Linux*, мы ещё успеем поговорить. Здесь же посмотрим только на конкретную маркетинговую войну разных группировок, отстаивающих "свободу".

Вот как комментирует историю перехвата в 1998 г. инициативы Эриком Реймондом у Ричарда Столлмена сам Линус Торвальдс: “Когда в 1998 г. открытые исходники привлекли всеобщее внимание, бурные дебаты возникли уже по поводу самого названия. До этого мы говорили о совместном использовании программного обеспечения на условиях лицензии типа *GPL* как о “свободном ПО”, использовали термин “движение свободного ПО”. Последний связан с Фондом свободного ПО, основанным Ричардом Столлменом в 1985 г. для продвижения таких свободных программных продуктов, как *GNU*, — созданная им свободная *UNIX*-система. Неожиданно просветители типа Эрика Реймонда обнаружили, что журналисты путаются: “свободный” означает “ничего не стоит”? Или “без ограничений”? Оказалось, что Брайан Белендорф, говоривший с журналистами от имени *Apache*, испытывает те же затруднения. После нескольких недель обмена мейлами, в котором я участвовал пассивно, получая копии (меня не интересовали политические аспекты), был достигнут консенсус: мы будем говорить “открытые” вместо “свободные”. Поэтому движение свободного ПО стало движением открытого ПО — для тех, кто рассматривал его (пожалуй, справедливо) как движение. Однако Фонд свободного ПО продолжает называться *Фондом свободного ПО*, и Ричард Столлмен по-прежнему является его идейным вдохновителем”.

Возвращаясь к связи открытых исходных текстов с доступом к технологиям, что роднит их с результатами научных исследований (в том числе и в отношении извлечения прибыли), Торвальдс пишет: “Понять феномен открытых исходников помогает аналогия с тем, как наука воспринималась религией столетия назад (а иными и сейчас воспринимается так же). Изначально наука представлялась чем-то вредным, опасным и антиобщественным — именно так многие софтверные компании рассматривают открытые исходники. И точно так же как наука не родилась для подрыва религиозных устоев, так и движение открытых исходников не направлено на разрушение софтверной отрасли. Его задача — производить хорошие технологии и смотреть, что из этого получится. Сама по себе наука не приносит денег. Богатство возникает как побочный эффект развития науки. То же самое верно и в отношении открытых исходников. Они дают возможность создавать вспомогательные отрасли, которые бросают вызов существующим предприятиям точно так же, как побочные продукты развития науки бросали вызов церкви… Как и в науке, побочные эффекты открытых исходников бесконечны. Возникают возможности, которые до недавнего времени казались немыслимыми. Открываются неожиданные новые рынки. Используя Linux или другие проекты с открытыми исходниками, компании могут создавать собственные версии и вносить собственные изменения, что невозможно ни при каких других условиях. Меня греет мысль, что всего случившегося с Linux нельзя было даже предвидеть, когда мы начинали”.

Теперь оставим в стороне трёх главных действующих лиц нашего повествования и обратимся к взгляду стороннего наблюдателя. Через два года после появления “Собора и базара” вышла обстоятельная критическая работа сторонника

открытых исходных текстов, который смело заявил восторженному профессиональному сообществу, что король-то голый.

В 1999 г. Николай Николаевич Безруков, признанный корифей антивирусных исследований в бытность Советского Союза, ведущий аналитик по интернет-безопасности в корпорации *BASF* (Германия), проф. университета *Fairleigh Dickinson* (США) опубликовал критическую статью "Повторный взгляд на Собор и Базар" ("A Second Look at the Cathedral and Bazaar" — First Monday, December 1999).

Очень рекомендую внимательно и обстоятельно изучить его аргументацию. Несмотря на всю деликатность, многие вещи он называл своими именами. Хотя ещё более детально это изложено в последующих работах Безрукова, которые можно найти на сайте его образовательного проекта *SoftPanorama*.

Профессор Н.Н. Безруков ставит под сомнение следующие важные утверждения Реймонда, которые вошли в работу "Собор и базар":

- закон Брукса (Brooks' Law) неприменим к распределённым разработкам, основанным на *Интернете*;
- "при достаточном количестве глаз все ошибки лежат на поверхности";
- разработка *Linux* является классическим примером "базарной" модели разработки программ;
- разработка программ методом открытых исходных текстов автоматически приводит к лучшим результатам;
- "базарная" модель разработки является новой революционной моделью разработки программного обеспечения.

Что важно отметить: если в коммерческом ПО приходится в последующих версиях нередко тащить за собой груз старых ошибок (как проектных, так и реализации), поскольку к ним уже попритёрлись разработчики и пользователи, то ещё большая инерционность может возникать в сообществе открытых исходных текстов — ведь здесь всегда есть соблазн попользовать готовенькое вместо переработки или написания с нуля своего.

Вот что пишет профессор Н.Н.Безруков: "Когда исходный текст написан плохо, а как считает Кен Томпсон (автор *UNIX* — Р.Б.), таковы некоторые части ядра *Linux*, в процессе исправления ошибок легко могут быть внесены новые. Переписать заново, а не отлаживать, было бы в этом случае куда более разумным подходом. Однако как раз в этом плане модель разработки программ на базе открытых исходников, с её преувеличенной верой в отладку, может оказаться помехой. В коммерческих программных проектах опытный менеджер может частично избежать этой проблемы, оценивая качество отдельных модулей и применяя в нужных случаях данную ему власть. В мире открытых исходников модули, включённые в проект на ранних стадиях, могут намного пережить свою полезность. Благодаря инерции и перегрузке ведущих разработчиков, попытки переписать эти модули могут не предприниматься до тех пор, пока не возникнут

серьёзные проблемы, которые оправдают усилия”.

Далее он отмечает: “Это концептуально бесконечное множество существующих в программе ошибок (тесно связанных с архитектурными просчётами) исключает позитивное влияние бессистемного исправления ошибок на конечное качество продукта. В случае ядра *Linux* я не вижу какого-то прорыва в качестве кода по сравнению с *Solaris* или *FreeBSD*”.

В завершение этой части моего повествования — важный вывод, сделанный профессором Н.Н. Безруковым, который особенно актуален в наши дни, дни нового всплеска интереса к Linux: “Как профессиональный преподаватель программирования, я вижу определённую опасность в романтизации разработки методом открытых исходников, особенно в среде студентов”.

3. Open Research как альтернатива Open Source

Как мы смогли заметить, движения *Free Software* и *Open Source* связывают понятия свободы и открытости:

- свобода использования;
- свобода изучения;
- свобода модификации;
- свобода распространения;
- открытость изучения;
- открытость распространения.

При этом в *Open Source Definition* подразумевают, что “там, где некоторые формы продукта не распространяются с исходными текстами, должны быть обнародованы средства получения за цену, не превышающую обоснованную стоимость воспроизведения — предпочтительно, бесплатное скачивание из *Интернета*”. Причём “промежуточные формы, такие как вывод препроцессора или транслятора, не допускаются”.

Занятные требования, не правда ли? То есть, если кто-то использует некоторые собственные средства высокоуровневого программирования (например, специальный математический аппарат — конечные автоматы, сети Петри, нейронные сети и т.п.), с которого **автоматически** происходит отображение на исходный текст, представленный на том или ином языке реализации, ему сразу указывают на дверь — вон из нашего *Open Source*. Да если и не автоматически, а вручную – кто и как может определить, работал специальный транслятор при отображении или сам человек? Если кто-то посчитал возможным воспользоваться какими-то внеязыковыми средствами, требующими препроцессорной обработки (например, своими шаблонами) — он вне закона в *Open Source*.

Не менее занимательно и следующее требование: “Преднамеренное изменение

исходного текста таким образом, чтобы он вводил в заблуждение, не допускается”. Простите, а кто и как определяет: (1) введение в заблуждение, (2) преднамеренность введения?

В требованиях к *Open Source* есть и такое требование: “Производные работы должны поставляться на тех же условиях, как и исходные” – они наследуют всю ту “свободу” и “открытость”. Не будем сейчас касаться совершенно понятных вещей. Например, того, что свобода одних есть всегда ущемление свободы других. Поэтому мы, как правило, имеем дело с “несвободной свободой”. Здесь же налицо выламывание рук в чистом виде. Если автор что-то создал и отдал в *Open Source*, если это получено (бесплатно или за деньги) кем-то, кто захотел использовать данный продукт, то у него должно быть право применять правомерно полученный продукт по своему усмотрению. Иначе это весьма своеобразная свобода. В том смысле, что “я тебе формулу дал – будь любезен, всё, что с ней наработал, – верни”.

С другой стороны, кто мешает послать далеко всю эту пышно декларируемую “свободу” и “открытость” и делать действительно открытые и доступные вещи на уровне общественного достояния? Т.е. переводить свои работы в известный правовой режим авторского права, когда подобные продукты могут свободно использоваться любым лицом без выплаты авторского вознаграждения. Другими словами, задаром и без ограничений.

Уходят в прошлое те времена, когда под словами “исходные тексты” (source) понимался исключительно текст на языке *Cu*. Исходники делаются на разных языках. Если будет несколько языковых слоёв, да ещё новых языков, не знакомых потребителям продуктов, это не сильно упростит задачу посторонним понять, что там происходит, как они получены, из каких (абстрактных) моделей. Примитивизм *Linux* и подобных ему *OpenSource*-вещей в этом смысле уйдёт в прошлое.

С точки зрения науки, создавать гигантские верификаторы того, что наплодили программисты исходя из общих соображений и общеизвестных (разбалансированных) языков программирования — куда менее разумно, нежели позволять там, где это возможно, использовать научный, математический аппарат, анализировать абстракции научными средствами, после чего генерировать исходные тексты на том или ином языке (что зависит от требований проекта, включая целевую платформу). Однако *Open Source* это отвергает. Не потому ли, что программирование с подачи американцев давно превратилось в ремесло, где слово “наука” вызывает, по меньшей мере, недоумение.

Почему возникло требование обязательного наличия исходных текстов? Идеологи движений за свободу исходили из того, что это — гарантия контроля технологий. Да неужели? Нет такой гарантии. Нет и полной уверенности (при наличии всех исходных текстов), что продукт не содержит умышленных “закладок”, потайных лазеек.

Open Source — это просто исходники. Обладание ими не есть обладание технологиями. Чем больше объём подобных исходников, тем большей фикцией становится владение контролем над системой. Если они, разумеется, делались не вами и не вашей группой. Это наглядно показал автор *UNIX* Кен Томпсон в своей лекции при вручении премии Тьюринга в 1983 г. "Закладки" ("трояны") можно встраивать в компиляторы, которые могут генерировать с одного и того же исходника разный код. Лекция так и называется "Размышления о том, можно ли полагаться на доверие". Какова мораль? Томпсон выразил её просто: "Нельзя доверять программе, которую вы не написали полностью сами... Сколько бы вы не исследовали и не верифицировали исходный текст — это не защитит вас от троянской программы". Кто не читал эту лекцию (Лекции лауреатов премии Тьюринга за первые двадцать лет (1966–1985). http://www.europrog.ru/book/trng1993r.pdf)

Дело не только в закладках. Надо обладать контролем над архитектурно-технологическими аспектами, которые в случае необходимости (патентные претензии, особенности целевого компьютера) могут быть переделаны. Заниматься восстановлением "рабочих чертежей" с исходников (где всегда есть ошибки и где идёт борьба конкретного программиста с ограничениями конкретного языка реализации) – неблагодарное занятие.

Открытость исходного текста даёт возможность иметь представление о том, что внутри происходит (хотя бы примерно). Это лишняя подсказка при отсутствии документации или её ущербности. Это одна из возможностей создания производного программного обеспечения. Хотя производное можно создавать и без наличия исходных текстов, путём *расширяющего программирования (extensible programming)* или *объектно-ориентированного программирования*. Наличие же всех исходных текстов облегчает задачу взлома и несанкционированного доступа, а также лишает возможности разработчиков хотя бы на время закрыть от любопытных глаз недоделанные вещи (которые — предмет перспективных исследований или коммерческих контрактов).

Если вы что-то разрабатываете, то должны отдавать себе отчёт в том, что ваше творчество на языках программирования никак не защищено. Авторское право — охрана **формы**. Оно не распространяется по определению на **идеи**. Переписать чью-то часть — не самая большая проблема. И это ахиллесова пята Open Source.

Что касается распространения принципа открытости исходных текстов абсолютно на всё, возникает вопрос (отсылаю к Кену Томпсону) — если нет возможности обеспечить гарантию отсутствия "тайных мест" даже при наличии абсолютно всех исходников, то в чём разница между закрытым исходным текстом и открытым исходным текстом? Всё решает **контекст** (компилятор, его конкретные версии, настройки, внешние файлы конфигурации и т.д.). В условиях использования формальных моделей это тем более справедливо.

Но если не *Free Software* и не *Open Source*, то какой может быть альтернатива? Давайте попробуем порассуждать. Разработка программ может быть как

результатом исследований, так и результатом производства (кустарного или промышленного). В первом случае, как правило, технологическая новизна выходит на первый план. Во втором требуется решить задачу по накатанной колее (опытное и серийное производство). Как отмечает Эрик Реймонд, распределённое сообщество в *Open Source* востребовано, когда что-то уже создано и надо его просто развивать и совершенствовать. Следовательно, даже сами идеологи этого направления признают, что технологические новинки создавать всем "скопом" в рамках *Open Source* – неразумно. С другой стороны, исследования в области программного обеспечения по своему характеру не сильно отличаются от научных исследований, имеющих славную историю и богатые традиции. В научном мире открытый обмен достижениями — обязательное условие развития. В последние годы, правда, коммерциализация науки сделала свое пагубное дело и привела к излишнему перекосу в сторону прикладной науки, причём работающей на потребнбости бизнеса. Что же, тем важнее предлагать модели, которые позволят преодолевать косность и стимулировать дальнейшее развитие как науки, так и индустрии.

Параллели с наукой были замечены и ранее. Линус Торвальдс писал: "Понять феномен открытых исходников помогает аналогия с тем, как наука воспринималась религией столетия назад (а иными и сейчас воспринимается так же). Изначально наука представлялась чем-то вредным, опасным и антиобщественным — именно так многие софтверные компании рассматривают открытые исходники". Гораздо более обстоятельно эти аналогии раскрыты в более ранней работе профессора Н.Н. Безрукова "Open Source Software Development as a Special Type of Academic Research (Critique of Vulgar Raymondism)" (First Monday, October 1999).

Приведём некоторые важные цитаты. "Убеждён,— пишет Безруков, — что существуют серьёзные проблемы, которые присущи модели разработки на основе открытых исходных текстов. Их нужно обсуждать и осознавать, а лучший способ понять их — использовать аналогию между сообществом открытых исходных текстов и научным сообществом (по сути они пересекаются)". Далее он продолжает: "Модель открытых исходных текстов очень близка модели научных работ, где исходные тексты трактуются как результаты исследований и публикуются для рецензирования и во благо человечества". В чём же аналогии? Безруков отмечает следующие моменты: взаимодействие в рамках распределённого сообщества, схожая модель финансирования, единая база (многие проекты создаются в научных подразделениях), одинаковые условия для взаимодействия через Интернет.

"Программное обеспечение, — пишет Безруков, — будет создаваться не отдельной личностью или группой людей, а глубоко взаимосвязанной сетью исполнителей. Хотелось бы сделать акцент на то, что виртуальные команды вероятно напоминают некогда действовавшие неформальные сообщества учёных, которые использовали традиционную почту с рукописными заметками для обмена достижениями, идеями и критическими взглядами". О финансировании и единой базе: "Финансирование проектов на основе исходных текстов очень схоже с

финансированием прикладной науки. Иногда научные исследования финансируются косвенно, поскольку люди, работающие в том или ином институте, становятся заинтересованы в конкретном явлении и ведут исследования за счёт имеющегося финансирования. Значительное число разработчиков в области ПО с открытыми исходными текстами находятся в академических институтах или больших корпорациях. Последние обычно предоставляют очень благоприятные условия для ведения проектов с открытыми исходными текстами, поскольку частенько дают пристанище талантливым разработчикам, которые прямым служебным вопросам уделяют лишь часть своего рабочего времени. Разработка *UNIX* в *Bell Labs* — хороший пример таких возможностей".

Что касается роли глобальных коммуникаций, Безруков отмечает: "Благодаря Интернету теперь можно создавать виртуальную команду для разработки ПО, подобную научной среде, где несколько исследователей, заинтересованных в конкретной проблематике, создают виртуальное научное сообщество для решения тех или иных задач. Малые проекты не нуждаются в явных координирующих структурах. Координация обычно автоматически осуществляется лидером проекта или группой лидеров. Этот подход плохо масштабируется. Для больших проектов авторитарная модель — единственный выбор, если ключевым фактором является скорость разработки. Если скорость рассматривается как важная цель проекта, то сообщество разработчиков осознанно или неосознанно будет идти по пути авторитарных тенденций в отношении своего лидера. Со временем это может приводить к проблемам".

Рассуждения о демократии и равенстве в рамках *Open Source* — это сознательная наивность: "Открытые исходные тексты воспринимаются как нечто демократическое, но это не так. Лидеры хорошо известных проектов в области открытых исходных текстов часто явно утверждают, что они диктаторы. Если Линусу Торвальдсу не нравится ваша правка, неважно сколь технически она совершенна".

В то же время, критика со стороны профессора Н.Н. Безрукова не идёт дальше — к формулировке альтернативной модели. Такой шаг предпринял профессор Анатолий Абрамович Шалыто (СПбГУ ИТМО). В 2003 г. он вышел с инициативой "Движения за открытую проектную документацию" (*Open Project Documentation)*, которую изложил в работе "Новая инициатива в программировании. Движение за открытую проектную документацию".

Отправной точкой послужила уже приведённая ранее работа профессора Безрукова "Повторный взгляд на Собор и Базар". Профессор А.А. Шалыто приводит цитату из этой работы: "центральный вопрос в практике программирования — вопрос о понимании программных текстов. Всегда хорошо иметь исходники, но проблема состоит в том, что зачастую их недостаточно. Чтобы понять некоторую нетривиальную программу, обычно требуется дополнительная документация. Эта потребность растёт экспоненциально с ростом объема кода. Анализ текстов программ, направленный на восстановление

первоначальных проектных решений, принятых разработчиками, и понимание программ являются двумя важными ветвями технологии программирования, существование которых неразрывно связано с недостаточностью исходных текстов для понимания программ. В качестве примера попробуйте понять структуру нетривиального компилятора при условии, что вы не располагаете определением того языка, который им компилируется".

Профессор А.А. Шалыто предлагает такой выход: "Итак, без исходных текстов плохо, но и с ними также нехорошо. Чего же не хватает для полного счастья? Ответ прост: проектной документации, выполненной весьма подробно и аккуратно, в которую программная документация входит как одна из составляющих".

Казалось бы, это снимает недостатки *Open Source*. Но так ли это? Каковы стимулы для создания проектной документации при условии, что лицензирование ведётся по-прежнему через схему *Open Source* со всеми вытекающими проблемами? Если речь идёт о проектной документации, то автоматически подразумевается производство. Но кто сказал, что программирование есть всегда производство? Технологические новинки ближе к научным вещам, к математическим абстракциям. Ученые-математики обмениваются проектной документацией? Вроде бы нет. Проектная документация подразумевает определённую унификацию (стандартизацию). Как это предполагается регулировать в сфере открытых исходных текстов? Получается, что особой мотивации нет и работа в рамках открытой проектной документации – это бремя, которое по доброй воле возлагают на себя её последователи.

Так не разумнее ли разграничить производство и исследования? И для исследований (точнее говоря, научно-исследовательских и опытно-конструкторских работ, НИОКР) предложить особую схему, учитывающую достоинства и недостатки ранее обсуждавшихся? Условно назовём её "открытые исследования" (*Open Research*).

Что туда должно войти:

1. Результаты труда должны быть общедоступны. Самая оптимальная форма — общественное достояние.
2. Проект должен быть открыт (публичен) ровно в той мере, как это принято в научном мире – раскрывать сырые, промежуточные результаты нет никакого смысла.
3. Информация о проекте должна быть представлена в разных жанрах, по большей части, неформальных заметках, где раскрываются: постановка проблемы, подходы к её решению, выбор оптимального варианта, конкретные абстракции и алгоритмы, инвариантные относительно языка реализации, описания ключевых спецификаций.
4. Исходные тексты должны рассматриваться как иллюстрация изложенных идей и подходов, а не как основная самодовлеющая часть.

5. Целесообразность представления той или иной части в открытом (исходном) тексте определяется разработчиками-исследователями. Другими словами, не должно быть ограничений на использование других форм представления программ, включая картриджи данных, в которых содержатся представления математических моделей.
6. В силу статуса общественного достояния нельзя налагать никаких ограничений на использование того, что в эту область авторами осознанно переведено.

Скептики возразят: “Понятно, хотите всё засекретить, прикрывшись мнимой открытостью”. Ну зачем же всё секретить? Открытость (публичность) проекта в рамках *Open Research* не подразумевает обсуждение перед публикой всех без исключения вопросов и установку веб-камер на рабочие места всех участников. Это не шоу.

А как же защита интеллектуальной собственности? Неужели всё на голом энтузиазме и чистом альтруизме? Зачем же? Общественное достояние сохраняет авторские права. Если есть необходимость “столбить” идеи и нет возможности (желания) воспользоваться патентной защитой — можно делать это традиционным образом: публикацией в различных изданиях (включая печатные и электронные). Какие-то вещи закрываются от прямого реинжиниринга специальными абстракциями. Это позволяет в определённой мере сохранять ноу-хау без потери доступности и открытости.

В остальном идеи свободы и открытости (если к ним прибегают) обретают более разумные черты. Здесь нет выкручивания рук. Нет наживы. Дело сугубо добровольное. *Open Research* создаёт все необходимые предпосылки для формирования виртуальных команд, получения внешнего финансирования (грантов и др.), создаёт основу для перехода к производству и внедрению. Да и для этих этапов данный подход вполне применим.

На самом деле, это практически та же схема (за вычетом статуса общественного достояния), по которой работала группа проф. Н.Вирта в *ETH Zurich*, начиная с 1970-х годов. Диссертации с идеями и подходами — в открытом доступе. Исходные тексты также доступны, в том числе для модификации. Более того, книги по таким проектам, ставшие мировыми бестселлерами, спустя годы переведены авторами в свободный (бесплатный) доступ.

Движение *Open Research* в состоянии стать вполне достойной альтернативой *Free Software* и *Open Source*. Проект создания новой перспективной отечественной операционной системы планируется проводить в рамках именно такой схемы.

Открытость программирования

Free Software и Open Source — это движения за открытость программирования. И в этом заключается их позитивная роль. Открытость — это инструмент борьбы с узурпированием знаний и технологий крупнейшими компаниями. Открытость нужна прежде всего потому, что при нынешних темпах деградации образования, программирование — одна из самых важных областей интеллектуального труда, являющегося локомотивом развития постиндустриального общества, — станет уделом избранных.

В конце января этого года на сайте старейшей в Европе компьютерной ассоциации — *Британского компьютерного общества* (*British Computer Society*), отмечающей в этом году своё 50-летие, была опубликована заметка под броским заголовком "Гибель компьютинга" (*Death of Computing*).

Её автор, преподаватель информационно-коммуникационных технологий (ИКТ) из английской "глубинки", выдвигает тезис о том, что настало время иначе взглянуть на роль фундаментальных знаний в сфере ИКТ. Если раньше на научный статус компьютерных наук никто не посягал, то теперь мы становимся свидетелями того, как восьмилетний карапуз своими руками создаёт виртуальных роботов без знания программирования и математики. Миновали те дни, когда всем заправляло программирование на ассемблере, когда программирование волновало воображение и было передовым рубежом науки, когда создавались сети компьютеров и требовалось заполнить белые пятна на карте мира прикладных программ. Всё это осталось в прошлом. Сегодня корабль получил пробоину ниже ватерлинии. В США число студентов, выбравших компьютерные науки, за период с 2000 по 2005 гг. упало на 39%. В Австралии на повестке дня вопрос о сокращении преподавательского состава по ИТ-дисциплинам. Прерывания, циклы, алгоритмы, формальные методы — всё это уже ничего не значащие слова. ИТ-индустрии нужны ресурсы для удовлетворения всё возрастающих информационных потребностей рынка. Наша жизнь в XXI веке кардинально изменилась по сравнению с тем, что было в 1960-х, 70-х, 80-х и даже в начале 90-х годов. Компьютеры стали неотъемлемой частью нашего существования — они унифицированы и вездесущи.

Что же, по мнению этого "революционера", надо делать? Разумеется, отбросить старые оковы, мешающие двигаться на пути прогресса. Нужны инновации, креативность, "дивергенция мышления". Новые факультеты компьютинга в высшей школе должны стать факультетами междисциплинарными, вбирающими в себя идеи из биологии, дизайна, истории, медицины. Это даст жизнь новой дисциплине — новому компьютингу XXI века!

Что тут сказать? Если бы дело происходило не в консервативной Британии, а по ту сторону Атлантического океана, была бы ещё понятна подобная риторика. Но здесь… Здоровый прагматизм против научного идеализма? На самом деле, подоплёка вопроса лежит гораздо глубже, чем это может показаться на первый

взгляд. Будем откровенны, разве нет подобных настроений у нас в стране? Это естественно, поскольку причины-то вполне объективны.

Владимир Игоревич Арнольд (академик РАН, президент Московского математического общества, член Исполкома Международного математического союза) пишет: "Расцвет математики в уходящем столетии сменяется тенденцией подавления науки и научного образования обществом и правительствами большинства стран мира. Ситуация сходна с историей эллинистической культуры, разрушенной римлянами, которых интересовал лишь конечный результат, полезный для военного дела, мореплавания и архитектуры. Американизация общества в большинстве стран, которую мы наблюдаем, может привести к такому же уничтожению науки и культуры современного человечества. Математика сейчас, как и два тысячелетия назад, — первый кандидат на уничтожение. Компьютерная революция позволяет заменить образованных рабов невежественными".

Компьютерная революция, идущая под флагом коммерциализации и под предводительством американцев, оказалась направлена не только на уничтожение культуры, науки и образования, но и на вымывание своего собственного фундамента. Действительно ли здесь чей-то злой умысел или это естественное следствие процесса коммерциализации? По всей видимости, второе. Как только бизнес и его ценности стали главенствующими — всё остальное подчиняется этому всемирному потопу, сметающему все преграды на своем пути.

В условиях современного коммерциализированного мира развитие идей вне рамок рынка существенно затруднено. Коммерциализация всё больше напоминает злокачественную опухоль. Её метастазы поражают практически все сферы цивилизации. Наука, образование, государственное управление — даже эти казалось бы внекоммерческие области становятся жертвами неизлечимой, смертельной болезни.

Такое ощущение, что в вузах (и далеко не только в наших) представлена прикладная математика, есть языки программирования и конкретные подходы (объектно-ориентированное программирование, функциональное программирование и т.д.), а дальше уже идут аморфные информационные технологии и всевозможные причиндалы. Следовательно, фундаментальные метазнания подменяются мозаичными знаниями "с миру по нитке". Для того, чтобы получить знания по ИТ, достаточно почитать книги. В свободное время. За чашкой кофе. Для того чтобы получить фундаментальные метазнания по программированию, надо работать под руководством Учителя и шевелить мозгами. В этом большая разница.

Математика имеет то принципиальное отличие от других дисциплин, что приучает думать, а не копить знания. Программирование же — это овеществлённая математика, не только позволяющая абстрактные идеи превращать в конкретные, наглядно-ощутимые вещи, но и дающая ключ к

познанию мира людей и машин. Их роднит и то, что в силу своей общности и фундаментальности они пронизывают другие науки, питая их и подпитывая себя.

Программирование в вузе должно играть ту же определяющую роль для формирования человека думающего, как и математика в школе. Обучение математике в школе не ставит целью дать стране максимум математиков. Равно и обучение программированию в вузе не должно быть связано с подготовкой программистов-кодеров. Их вполне по силам готовить и специальным ПТУ. Если раньше стране требовалась большая армия высококвалифицированных инженеров, и высшая математика играла ключевую роль, то сейчас на повестке дня подготовка армии высококвалифицированных инженеров интеллектуального мира. Важно знать суть и то, как её применять на практике.

Наш первый программист-академик Андрей Петрович Ершов выделял три ветви программирования: теоретическое (наука), системное (базис приложений) и прикладное (надстройка приложений). Ничего существенного в этой классификации не поменялось. Просто третье теперь поставлено во главу угла. Homo sapiens (человек разумный) постепенно трансформируется не в Homo intelligens (человек сведущий), а вырождается в Homo rationalis (человек рассудочный). С утратой понимания глубинной причинно-следственной связи вещей и явлений ("почему"), мы всё больше переходим на потребительско-рецептурный образ жизни ("как").

Академик В.И. Арнольд достаточно резко очертил социальную роль математической науки: "Роль доказательств в математике подобна роли орфографии или даже каллиграфии в поэзии. Тот, кто не научился искусству доказательства в школе, не способен отличить правильное рассуждение от неправильного. Такими людьми могут легко манипулировать безответственные политики. Результатом могут стать массовый гипноз и социальные потрясения. Л. Толстой писал, что сила правительства основана на невежестве народа, что правительство знает об этом, и потому будет всегда бороться против просвещения".

Казалось бы, какое отношение эта развернутая преамбула имеет к теме открытости программирования? Самое что ни на есть непосредственное. Открытость содержания (идеи, технологии) ныне подменяется открытостью формы (исходный текст). И это подается чуть ли не как идеал.

И вот уже известный специалист в области программирования, Диомидис Спинеллис, пишет обстоятельную книгу “Анализ программного кода на примере проектов *Open Source*” (2004), где раскрывает подходы к тому, как извлекать технические решения и используемые методы, ковыряясь в чужом исходном тексте. Потрясающе! Получается, что исходные тексты — это одолжение, которое делают его разработчики, не удосужившиеся подготовить детальную **проектную документацию** (**некогда, лень, жалко**) и при этом не чурающиеся получать деньги за передачу формы, утаив содержание.

Как вы думаете, когда реально разобраться в чужом исходном тексте при **полном отсутствии проектной документации**? Правильно, когда наукоемкость такого продукта близка к нулю и когда вы хорошо знаете язык, на котором написан продукт.

Вспомним слова Линуса Торвальдса: "... будущее операционных систем будет определяться потребностями пользователей, а не любопытством программистов ... Технология более не управляет рынком. Им движут желания людей..." Насколько мне известно, Линус Торвальдс опубликовал и произнёс около сотни интервью и речей, но он **не опубликовал ни одной работы, освещающей структуру ядра *Linux***. Почему? Если вы посмотрите на деятельность профессора Вирта, который не кричал на каждом перекрёстке о свободе, но делал для неё неизмеримо больше многочисленных болтунов, то в его работах и книгах детально представлено внутреннее устройство систем, подходы к решениям, по винтику разбираются многие механизмы.

Чтобы владеть по-настоящему технологией, надо в её разработке участвовать. Желательно с самого начала. И цениться будет не технология сама по себе, а люди — живые ее носители.

Если помните, раньше, на заре туманной юности, идеологи *Open Source* проповедовали целую культуру. Она была на первом плане и в *BSD*-среде, и у Столлмена, и у Торвальдса. Речь шла о культуре созидания. Обратите внимания: созидания, а не распространения. В настоящее время разговоры вокруг *Open Source* нередко сводятся к обсуждению открытых лицензий. Но что такое лицензия? Это правовая обёртка, в которую надо заворачивать продукт, когда он уже дошёл до кондиции. Раньше ценился процесс. Общение людей, объединённых общей задачей, делающих общее дело. И не во имя будущих отчислений. В центре внимания был процесс творчества.

Если же сейчас некая компания (*Microsoft*, *Sun*, *IBM* и др.) всё сделает втихомолку, даже будет какое-то время продавать продукты без исходников и потом решит перейти в плоскость *Open Source*, то она с точки зрения лицензий ничем не будет отличаться от тех, кто изначально всё это строил на таких принципах. Понимаете? Таким образом, *Open Source* из культуры превращается в фантик, в удобную обёрточную бумагу, которая позволяет тем или иным компаниям за счёт демпинга решать конкретные рыночные задачи на данном этапе.

Исходники стали ценностью, они ногой открывают дверь в новые рыночные ниши, получают мгновенный доступ к массовой аудитории. Это демпинг по-новому. Демпинг с человеческим лицом. Гуманитарная помощь.

Не исходники должны быть ценностью, а идеи, определяющие реализацию; люди, их создающие; технологии, в которые воплощаются идеи; проектные решения, которые определяют будущий облик зданий программной индустрии. Реализация на конкретном языке не столь существенна. Язык может быть другой. Просто потому, что это требуется данным проектом.

Основополагающий принцип открытого исследовательского программирования (*Open Research Programming*) в другом: здесь важны **люди** (открытость), **идеи** (исследования), **результаты** (программирование). Именно эта тройка и раскрывает суть нового подхода. Для людей важна среда творческого общения. Идеи необходимо формулировать так, чтобы они были отчуждаемы от автора. Результаты выражаются не только и не столько в исходном тексте, сколько в совокупности материалов, важных для их применения, понимания и дальнейшего развития.

Для *Open Research Programming* как концепции важны три момента:

1.Собственность.
Общественное достояние. Продукт интеллектуального труда передаётся в общественное пользование (public domain).

2.Наука.
Исследования (основанные на науке) первичны. Продукт интеллектуального труда — не просто программный продукт с открытыми исходными текстами, но ещё и открытые технологии.

3.Применение.
Всегда (а не в отдельных случаях) неограниченное использование (никаких юридических и межлицензионных конфликтов, поскольку "общественное достояние").

Движение *Open Source* возникло как основа децентрализованного (кустарного) производства программных продуктов, где открытие исходных текстов является главной мотивацией участия добровольцев. В настоящее время *Open Source* используется также и в промышленном производстве (как правило, на стадии отчуждения готового результата, полученного традиционным закрытым образом в крупных компаниях). *Open Source* в случае участия большой группы людей крайне неэффективно при работе с нуля (мнение Эрика Реймонда). Сначала кто-то в очень узкой группе в закрытую создаёт основу, а потом её переводят в публичное развитие *Open Source*. По сути это своеобразная форма опытного производства (получения опытного образца). Серийным же производством занимаются группы и компании, специализирующиеся, например, на выпуске дистрибутивов *Linux*.

Free Software и *Open Source* с их акцентом на результат в виде исходных текстов стимулируют программное производство. Но ведь прежде чем переходить к производству (что кустарному, что промышленному), **надо провести исследования**. Научно-исследовательские и опытно-конструкторские работы (НИОКР, R&D). Это белое пятно в *Free Software* и *Open Source*. Только после прохождения этой стадии можно переходить к опытому и серийному производству. А если её нет — **наукоемкость нулевая, продуманность систем сомнительная**, работа больше строится на переделке известных вещей, так сказать, из общих соображений.

НИОКР в сфере программирования имеет свои нюансы. Это и теоретические изыскания, и конкурентная разведка, и использование макетирования для проверки идей, и критический анализ разных вариантов решения. Об особенностях исследовательского программирования написано немало статей. Так, в начале 1990-х годов швейцарские ученые из *ETH Zurich* А.Киральф, К.Чен и Й.Нивергельт выделили следующие важные моменты:

- разработчик ясно представляет направление поиска, но не знает заранее, как далеко он сможет продвинуться к цели;
- нет возможности предвидеть объём ресурсов для достижения того или иного результата;
- разработка не поддается детальному планированию, она ведётся методом проб и ошибок;
- такие работы связаны с конкретными исполнителями и отражают их личностные качества.

Мне могут возразить: но *Free Software* и *Open Source* не запрещают вести НИОКР, не запрещают создавать и предоставлять проектную документацию. Ещё бы они это запрещали! Но это не стимулируется. Никак. С таким же успехом там не запрещается и свободно продавать огнестрельное оружие (кстати, здесь проходит точка разногласий между Торвальдсом и Реймондом), или даже давать бесплатно, в нагрузку к софту. Всё это было бы смешно, когда бы не было так грустно…

Безусловно, если брать лицензионный уровень, то *Open Research Programming* приближено к лицензиям *MIT*, *BSD.* Оно покрывается требованиями *Open Source* (является частным случаем). Однако это принципиально иное: это другая КУЛЬТУРА, другое мировоззрение, другая система ценностей. Это наиболее крайняя, РАДИКАЛЬНАЯ форма воплощения идей сообщества открытых исходных текстов. На чистом сливочном масле. Не придраться. Потому что из-за отчуждения результатов в общественное достояние (public domain) деньги из рассмотрения изъяты. И собственность авторам остаётся только в головах и бирках (надписях в исходниках и лицензиях). Нечего делить. В этой схеме нет понятия отчислений (роялти). Нет таких отношений. Один оригинал и неограниченное количество бесплатных копий. Если за оригинал платят грантами, добровольными пожертвованиями (donation) — это нормально. Если не платят — тоже нормально. Деньги надо извлекать не отсюда, а из порождённых этим процессом вещей (новых знаний, новых продуктов, новых контактов).

Главная мысль, к которой пытаюсь подвести: уход в специализированные языки и математический аппарат не только повышает качество работы (и результата), но и затрудняет интеллектуальное воровство. Своровать-то можно, а что с ним потом делать, с наворованным-то? Цениться должны технологии, четко сформулированные идеи, алгоритмы, проектные решения, а не их следствие — исходный текст. И если полноценно защитить технологии в области программирования не в силах патенты и авторское право — остаётся защищаться самими же технологиями. Их уровнем. Порогом вхождения в них. Постоянным

развитием и совершенствованием. Тогда можно спокойно передавать их в открытом виде. Время на изучение и внедрение даёт необходимую фору во времени. При таком подходе появится стимул повышать наукоёмкость программного обеспечения, поскольку это станет сильным конкурентным преимуществом.

Однако повышая наукоёмкость, надо сохранить сообществу доверие к используемым программам, компонентам и системам: давать в этой схеме их полностью открытыми, со всеми необходимыми сопутствующими вещами (включая файлы данных), с описанием применяемых технологий (включая ссылки на диссертации и т.п.).

Если движение *Open Research Programming* начнёт принимать массовый характер, это существенно изменит ситуацию. Ситуацию с мозгами в первую очередь. Чем больше открываешь ты, тем больше открывают тебе. Возникает конкуренция открытости, а не её имитация ради получения прибыли. Именно поэтому *Open Research Programming* может стать хорошим катализатором развития наукоёмкого программного обеспечения. Оно в силах изменить нынешнюю безрадостную картину гегемонии бизнеса, подминающего под себя всё: и умы, и чаяния, и даже государственную власть.