Денис Силаков

Silicon Graphics и Open Source

Имя SGI прочно ассоциируется со всевозможными графическими технологиями работы с графикой; в частности, большинство пользователей свободных ОС знакомы с разработанной в SGI открытой спецификацией OpenGL. Однако вряд ли многие задумывались о причинах, побудивших компанию пойти на публикацию своего API, а также о других инициативах SGI в этой области.

Полагаю также, что многие пользователи Linux слабо осведомлены о вкладе SGI в разработку этой ОС – попробуйте вспомнить навскидку что-то кроме XFS и вещей, связанных с OpenGL. А ведь деятельность компании на арене Linux была достаточно бурной, хоть и недолгой – начавшись во второй половине 90х, она практически сошла на нет к 2003 году, как результат кризиса дот-комов. Тем не менее, ряд проектов, начатых SGI, актуальны и сегодня.

В этой статье я постараюсь рассказать о том, что же SGI успела сделать для мира FOSS до того, как столкнулась с финансовыми затруднениями, приведшими, в конце концов, к банкротству и поглощению Rackable Systems.

OpenGL

Популярность решений SGI в мире компьютерной графики основывалась не только на мощном аппаратном обеспечении, но и на удобном (но проприетарном) API для доступа к нему – IRIS GL (Integrated Raster Imaging System Graphics Library). В 1992 году SGI сделала несколько неожиданный ход – API был опубликован в виде открытой спецификации под именем OpenGL, а для его дальнейшего развития был создана группа OpenGL Architectural Review Board (ARB).

Относительно причин этого шага, мнения расходятся. Популярна версия, что причиной послужила разросшаяся реализация IRIS GL, нуждавшаяся в переработке. Но на мой взгляд, несколько наивно полагать, что основным поводом открытия API стала надежда на помощь “со стороны” в переработке и дальнейшей поддержке библиотеки. К тому же, открыта была именно спецификация API; реализовывать функции каждый заинтересованный производитель должен был самостоятельно (“эталонная” реализация от SGI – OpenGL Sample Implementation – была открыта только в 2000 году).

Так что вполне имеет право на существование и другая версия – основной причиной выпуска OpenGL стали опасения относительно конкурентов (HP, IBM и Sun), начавших продвижение своих продуктов для работы с графикой, использующих другой стандарт – PHIGS. Открытие и активное продвижение технически более совершенного OpenGL позволили SGI превратить его в стандарт де-факто; при этом у SGI уже были готовые отработанные решения, поддерживающие OpenGL, а конкуренты оказались в роли догоняющих.

Впрочем, открытием API дело все-таки не ограничилось – помимо уже упомянутой Sample Implementation, SGI разработала расширение GLX (OpenGL Extension to the X Window System, привязка OpenGL к X11) и способствовала его интеграции в XFree86. Инженеры SGI сотрудничали с проектом Mesa (свободной реализацией OpenGL) и даже предоставили проекту доступ к официальным (закрытым) сертификационным тестам OpenGL (которые, по заверению

разработчиков, были успешно пройдены).

Одним из дальнейших направлений развития OpenGL стал Open Inventor (OI) - API более высокого уровня, позволявший оперировать объектами отображаемой сцены. OI основан на проприетарном IRIS Inventor; “открытие” состоялось в 2000 году, официально – в ответ на желание пользователей иметь реализацию под Linux, но более вероятно – из-за потери интереса SGI к проекту – после открытия, OI практически не развивался. К тому времени компания завязла в разработке “все более лучших” графических библиотек – в середине 1990х планировалось объединить IRIS Inventor с проприетарным OpenGL Performer в проекте Cosmo3D. До релиза дело не дошло, SGI переключилась на OpenGL++, а затем отказалась и от него в пользу Farenheit – совместного проекта с Microsoft по объединению OpenGL с DirectX, также тихо почившего.

В итоге, стандартного решения более высокого уровня, чем OpenGL, так и не появилось. Но коммерческая версия OI и поныне предоставляется компанией Visualization Sciences Group, а кроме того, существует полностью совместимая с OI на уровне API библиотека Coin (http://www.coin3d.org/), доступная под двойной лицензией (GPL и коммерческой).

Open Inventor лег в основу еще одного открытого стандарта – VRML (Vritual Reallity Modelling Language), языка описания трехмерных сцен. Конечно, лавров OpenGL этот стандарт не снискал, но свою аудиторию нашел; последователь VRML – стандарт X3D – в настоящее время развивается консорциумом Web3D.

Linux

Считается, что одним из факторов, приведших к краху SGI, стал отказ от собственной линейки MIPS-процессоров и переход на Itanium. В то же время мир FOSS от этого перехода, скорее, выиграл – вместе с новыми процессорами, SGI перешла на Linux (вместо IRIX), и внесла немалый вклад в развитие этой системы.

Естественно, существенная часть работ была нацелена на улучшение работы Linux на Itanium, но многие наработки не ограничены только этим процессором. Были открыты и перенесены на Linux некоторые продукты и технологии IRIX, не зависящие от аппаратной платформы; в частности, была открыта файловая система XFS, в настоящее время развиваемая и поддерживаемая сообществом. О ряде менее известных проектов речь пойдет ниже.

Большое внимание уделялось возможности использования Linux в многопроцессорных системах. SGI принимала активное участие в проектах Atlas (http://atlas-64.sourceforge.net/) и Linux Scalability Effort (http://lse.sourceforge.net/), нацеленных на повышение масштабируемости ядра и добавление поддержки технологий, применяемых в многопроцессорных системах (таких, как NUMA - Non-Uniform Memory Access).

SGI поощряла выпуск дистрибутивов для Itanium и даже предлагала собственный SGI Advanced Linux Environment (ALE), основанный на RHEL, но большой популярности он не снискал.

Помимо Itanium, SGI некоторое время участвовала в разработке порта Linux на MIPS. Линейку MIPS в SGI свернули в 2006 году, а поддержку порта – и того раньше. (Конечно, это не означало конец поддержки MIPS в Linux – силами других участников, проект http://www.linux-mips.org по-прежнему развивается)

В целом, почти все работы SGI в Linux так или иначе были связаны с ядром. Рассмотрим ряд наиболее значимых инициатив компании в этой области.

Ядро - управление процессами

Одним из наиболее заметных направлений стало расширение возможностей ОС по управлению процессами. Так, патчи для ядра и утилиты PAGG (Process AGGregates) позволяли объединять процессы в задания и работать с заданиями, как с процессами (похоже на группы процессов, но с более удобными возможностями контроля со стороны администратора). Со временем появилась возможность оповещать модули ядра о вызовах fork, exit и exec в процессах, а проект был переименован в pnotify ("p" – от "process"; не путать с кросплатформенным аналогом inotify – http://mark.heily.com/pnotify, – который к SGI отношения не имеет).

Возможности pnotify использовались в CSA (Comprehensive System Accounting) – инструментарии, позволявшем отслеживать использование системных ресурсов процессами и заданиями. Помимо поддержки заданиями, "фишкой" CSA была возможность отслеживания работы с диском.

Судя по всему, CSA и pnotify оказались слабо востребованы. Последние патчи pnotify выпущены в 2006 году и рассчитаны на ядро 2.6.16, а CSA не обновлялся с 2007 года. Впрочем, часть их функциональности сегодня предоставляется другими инструментами; в частности, отслеживание работы с диском реализовано непосредственно в ядре, начиная с версии 2.6.20, чем пользуются утилиты типа iotop.

Более востребованным оказался CpuSets – механизм привязки процессов к конкретным процессорам и банкам памяти в многопроцессорных системах. CpuSets изначально был реализован для ядер 2.4.x (и назывался CpuMemSets), а начиная с версии 2.6.12 включен в основную ветку ядра.

Инструменты разработки ядра

Помимо расширения функциональности ядра Linux, инженеры SGI уделяли много внимания инструментам его разработки.

В SGI был создан отладчик ядра kdb – актуальный проект, ведь Линус Торвальдс долгое время не хотел добавлять какие-либо средства отладки в ядро. Поддержка "родного" отладчика kgdb появилась лишь в ядре 2.6.26, и при этом для его использования необходима дополнительная машина. А вот kdb можно использовать на том же компьютере, что и отлаживаемое ядро, так что он все еще актуален и поддерживается (страничка http://oss.sgi.com/projects/kdb явно устарела, но в разделе 'Downloads' свежие версии имеются).

Для ветки ядра 2.4 предоставлялся профилировщик kernprof, существенно превосходивший штатный. Но в 2003 году проект практически остановился, и в настоящее время стандартом де-факто для профилировки ядра является совсем другой инструмент – oprofile.

Наконец, утилиты проекта Linux Kernel Crash Dumps (LKCD) позволяли получать дамп памяти ядра при падениях (kernel panic, oops) и проводить последующий анализ этих дампов. В настоящее время LKCD вытеснен новым kdump.

Завершая тему ядра, напомню, что одним из немногих примеров переиспользования кода UNIX в Linux, продемонстрированных SCO в приснопамятном разбирательстве, были примерно 200 строк, внесенные SGI для поддержки Itanium. К моменту демонстрации их уже убрали из ядра ввиду наличия другого кода с аналогичной функциональностью. Более того, Рич Альтмайер (Rich Altmaier, в то время – вице-президент SGI) заявил в письме к сообществу (http://oss.sgi.com/letter_100103.txt), что проблемный код хоть и происходит из UNIX System V, но попадает в категорию общественного достояния (public domain). Тем не менее, с сайта SGI код убрали, заодно “подчистив” еще несколько потенциально спорных фрагментов.

Работа с распределенными системами

Логичным дополнением работ SGI по поддержке многопроцессорности стал ряд продуктов для многомашинных комплексов.

Проект Linux Failsafe (порт IRIX Failsafe, разработанный совместно с SUSE) предоставлял ПО для кластеров, повышавшее отказоустойчивость в случае выхода из строя физических машин. Увы, после кризиса в начале века SGI забросила и этот проект, а одной SUSE, по словам Ларса Маровски-Бри (Lars Marowsky-Bree, инженер SUSE), он был не по силам (http://article.gmane.org/gmane.linux.failsafe/277/). Но Ларс заверил, что все наработки переиспользуются в схожем проекте Linux-HA (High-Availability Linux, http://www.linux-ha.org/).

Другой порт с IRIX – Open/SpeedShop, профилировщик приложений, поддерживающий кластерные решения (в частности, обеспечивающий анализ вызовов MPI). Проект развивается и сейчас, но уже без SGI, а силами института Крелла (Krell Institute) и Лос-Аламосской и Ливерморской национальных лабораторий.

Еще один изначально проприетарный продут – Performance Co-Pilot – предоставляет инфраструктуру для мониторинга и управления ресурсами многомашинных комплексов. Проект был открыт в 2000 году и в настоящее время поддерживается сообществом, располагаясь по-прежнему на http://oss.sgi.com/projects/pcp/ (и являя собой чуть ли не единственный “живой” раздел портала oss.sgi.com).

Наконец, проект, знакомый и пользователям десктопов – это FAM (File Alteration Monitor) - монитор, оповещающий заинтересованные приложения об изменениях файлов – как локальных, так и находящихся на удаленных машинах (средствами локальных демонов на этих машинах, не перегружая сеть). FAM не развивается с 2003 года, но существует независимый проект Gamin, стремящийся к совместимости с FAM на уровне API и ABI (о нем, правда, тоже мало слышно последние пару лет). Частичной альтернативой FAM является механизм ядра inotify, но он присутствует только в Linux, да и работа с удаленными машинами там реализована по-другому и требует поддержки со стороны модуля файловой системы (а для NFS, например, такой поддержки нет). Так что FAM/Gamin все еще популярны и присутствуют в современных дистрибутивах Linux.

Заключение

Таким образом, SGI сделала немало для мира открытого ПО. Многие разработки компании достаточно низкоуровневые и поэтому неизвестны большинству пользователей, но они сыграли существенную роль в становлении Linux как надежной платформы для корпоративного сектора – в частности, за счет улучшения поддержки многопроцессорных и многомашинных

комплексов.

В настоящее время часть разработок SGI интегрирована в другие продукты (включая ядро Linux), часть развивается сообществом, а что-то оказалось невостребованным и представляет разве что историческую ценность. К сожалению, об участии самой компании в открытых проектах приходится говорить в прошедшем времени – Rackable Systems, купившая SGI в 2009 году, пока никак не проявила себя в этой области. Но частью доставшегося ей наследства Rackable уже достаточно умело воспользовалась, сменив имя на Silicon Graphics International – то есть, все тот же SGI. Интересы Rackable лежали в области аппаратного обеспечения, но, возможно, “новый” SGI еще вернет свои позиции в разработке ПО и не обделит вниманием сообщество FOSS. Поживем – увидим.