Руслан Богатырев

Язык как основа архитектуры. Проект Lilith

Богатырев Р. Язык как основа архитектуры // ComputerWeek-Moscow. 1998. № 19.

Данная статья открывает цикл публикаций, посвященных проблемам органичного взаимодействия программ и оборудования, в том числе взаимного проникновения программной и аппаратной составляющих компьютерных систем.

Вопрос о том, что первично — материя или сознание, должен, казалось бы, интересовать только философов. К компьютерной индустрии он вроде бы не имеет никакого отношения. Но в то же время нетрудно заметить, что многие проблемы в этой области проистекают от взаимного несоответствия, постоянно существующего разрыва между аппаратными средствами и программным обеспечением. Совершенствование элементной базы и наращивание мощи процессорной части за счет одной лишь тактовой частоты, изощренного конвейерного механизма и средств кластеризации неизбежно ведут к тому, что для решения вполне конкретных практических задач приходится впустую расходовать массу "энергии", не добиваясь при этом более или менее приемлемого результата. Что уж тут говорить о "кпд" процессорной части.

Движение компьютерной индустрии по пути экстенсивного развития до недавнего времени было оправдано прежде всего высокой технологической дешевизной подобных решений. Однако по достижении чисто физического предела аппаратных компонентов волей-неволей приходится задумываться о том, что инженерная мысль, воплощенная в программном обеспечении, должна иметь эффективную основу, фундамент, способный объединить "материю" и "сознание".

Даже если примитивно рассматривать ПО в качестве топлива для процессорного блока (двигателя), отводя ему заведомо подчиненную роль, все равно придется признать, что "накачивать" магистраль чем попало вряд ли разумно. Нужно точно знать вид требуемого топлива и правильную его дозировку. Если от машин и агрегатов перекинуть мост к строительной индустрии, с которой нередко сравнивают компьютерную отрасль, то опять же легко заметить, что в строительстве сосуществуют две составляющие — сознание и материя, искусство (архитектора) и инженерия (проектировщика и инженера-строителя). От их органичного сочетания в конечном итоге зависит судьба сооружения. Казалось бы, красивый замысел, оторванный от реального воплощения, превращает архитектора в рядового чертежника. Однако жесткая привязка лишь к имеющимся в наличии стройматериалам и конкретной технологии строительства ведет к другой крайности — ремесленничеству в худшем понимании этого слова.

Одним из стимуляторов пересмотра сложившегося подхода к производству аппаратных и программных средств стала видимая даже невооруженным глазом революция в области устройств бытовой и промышленной электроники. Миниатюризация исполнения, требующая одновременно дешевых и гибких решений, заставляет по-иному взглянуть на противоборство универсализации и специализации, на создание информационных, вычислительных и управляющих систем нового поколения, проанализировать тот эволюционный путь, по которому пошло развитие аппаратуры и ПО.

Цикл публикаций под общим заголовком "Язык как основа архитектуры" мы откроем небольшим экскурсом в историю, рассказом о малоизвестном проекте создания компьютера, полностью ориентированного на вполне конкретный язык программирования.

Вершины маленькой Швейцарии

Швейцария гордится не только изысканными сортами сыра и безупречными по качеству часами, не только надежной банковской системой и сказочными по своей красоте швейцарскими

Альпами. Воплощением изысканности, качества, надежности и красоты стал знаменитый швейцарский армейский нож, даже вне зависимости от того, какой из двух непримиримых конкурентов его выпускает — Victorinox или Wenger.

Аналогом этого простого и продуманного инструмента в области компьютерных технологий стоит, пожалуй, признать язык Modula-2, разработанный швейцарским профессором Никлаусом Виртом (Niklaus K. Wirth). Этот ученый-затворник известен миру прежде всего своим языком Паскаль (который не следует путать с многочисленными диалектами в виде Turbo Pascal, Object Pascal и др., имеющими весьма отдаленное отношение к первоначальному замыслу автора канонического варианта). В то же время лишь немногие знают, что Вирт создал нечто большее, чем просто несколько языков программирования (PL360, Algol-W, Паскаль, Modula/Modula-2, Oberon/Oberon-2): за три десятилетия он построил свою Школу, своеобразную фабрику технологий, которая дала миру немало идей, талантливых ученых и инженеров. Она расположилась в тихом и респектабельном Цюрихе, в стенах Швейцарского федерального технологического института ETH (Eidgenoessische Technische Hochschule), а точнее, в небольшом Институте компьютерных систем (Institute for Computer Systems). Трудно сказать, строил ли Вирт ее по образу и подобию американского Стенфорда (Stanford University), где в должности профессора в середине 1960-х несколько лет преподавал компьютерные дисциплины. С уверенностью можно говорить лишь о том, что университетский городок ETH по структуре и духу был близок знаменитой американской кузнице кадров.

Почти два года назад, летом 1996 г., во время очередного приезда Вирта в Россию (в Москву и Новосибирск) мне посчастливилось с ним побеседовать. Думаю, собранная информация будет интересна читателю (тем более в свете нашего разговора о синтезе программных и аппаратных решений).

Поворотным моментом в судьбе самого Вирта и его Школы стал проект Lilith. Никлаус Вирт до сих пор вспоминает о нем с большой теплотой. К середине 1970-х, по словам Вирта, ему уже порядком надоели и языки программирования, и необходимость писать хорошие компиляторы для имеющихся компьютеров. Ему неосознанно хотелось создать своими руками нечто новое, неизведанное. Используя свой законный год отпуска (sabbatical year), предоставляемый профессорам университетов раз в семь лет, он отправился за океан, в исследовательский центр Xerox PARC. Здесь в 1976—1977 гг. в лаборатории Алана Кея (Alan Kay) он увидел Xerox Alto — настоящую персональную станцию с вертикальным монитором, мышью и собственным (!) диском. Больше не надо было сражаться за ресурсы мэйнфрейма с другими пользователями: рядом, под рабочим столом, находился личный компьютер, канал доступа к которому в 5 тыс. раз превышал привычный 3-КГц канал мэйнфрейма.

Итак, Вирт воочию убедился, что почти на коленках (а немало ключевых решений компьютерной индустрии создавалось, как правило, в гаражах и именно на коленках!) можно создать собственный компьютер, да еще реализовать на нем компилятор языка (Mesa), куда более сложного, чем его Паскаль. Профессионалу часто нет нужды детально разбираться с новой идеей, ему достаточно показать прецедент решения. Готовая реализация снимает многие сомнения, и все остальное может быть выполнено почти "на автомате". Но Вирт не хотел повторять путь, проложенный командой Кея. Да и цели он ставил перед собой совсем иные, нежели концепция Dynabook. Прежде всего ему хотелось создать целостную компьютерную систему не только для пользователя, но и для разработчика, причем в основе ее должна лежать персональная рабочая станция, "заточенная" под конкретный язык. Вирт решил "тряхнуть стариной" и вспомнить свое университетское образование (закончив в 1958 г. ETH и став инженером в области электроники, он продолжил учебу в Квебеке и в Калифорнийском университете в Беркли).

Тогда, по словам Вирта, он понял, что хорошо разбираться в компьютерной аппаратуре без знания программного обеспечения практически невозможно. После возвращения осенью 1977 г. из Xerox PARC он сделал для себя еще одно важное открытие: программное обеспечение должно подчинять себе аппаратуру, а не наоборот. Легко сказать, но как этого добиться?

Язык Modula-2

Одной из главных проблем неэффективной работы аппаратуры является разрыв между языком программирования и системой команд процессора. Именно это несоответствие и усиливало неприязнь Вирта к существовавшим архитектурам компьютеров, которые отравляли жизнь

разработчикам компиляторов. Впечатления от увиденного в Xerox PARC не давали ему покоя. Вирта целиком захватила идея разработать и построить с нуля собственную компьютерную систему, которая состояла бы из аппаратной части, микрокода, компилятора, операционной системы и различных сервисных программ. Фундаментом ее должен был стать новый язык, который мог бы в максимальной степени задействовать возможности будущей станции. Этот язык должен быть компактным и в то же время универсальным, способным одинаково хорошо решать задачи и системного, и прикладного программирования.

Оставалось всего ничего: придумать язык, собрать команду единомышленников и засучив рукава взяться за работу. Предварительный документ, описывающий цели и концепции языка Modula-2, Вирт подготовил к концу 1977 г.

Здесь надо заметить, что Modula-2 и сам проект Lilith возникли не на пустом месте. Речь идет не о Xerox PARC. В период с 1968 по 1970 г. был создан язык Паскаль, и первый его компилятор был реализован (после раскрутки) на самом же Паскале для мэйнфрейма CDC-6000 компании Control Data. Вслед за этим, в самом начале 1970-х, Вирта попросили помочь в переносе компилятора Паскаль на другие машины. Поскольку язык был предназначен прежде всего для обучения, быстродействие не играло большой роли, и группа Вирта решила создать унифицированную архитектуру виртуальной Паскаль-машины. Специально для нее был спроектирован байт-код (P-код), идеи которого оказались настолько популярны, что на их основе впоследствии появилось немало интересных решений. В 1991 — 1992 гг. концепция Паскаль-машины была использована Джеймсом Гослингом (James Gosling) при построении языка и архитектуры Java (кстати, Гослинг в начале 1970-х годов реализовал Паскаль-машину для компьютеров семейства Digital VAX).

К тому времени (1972 г.) Пер Бринч Хансен (Per Brinch Hansen) уже реализовал параллельный диалект языка Паскаль, получивший название Concurrent Pascal. Вслед за этим, в 1974 г., английский профессор Чарльз Энтони Хоар (Charles Anthony Hoare) в журнале Communications of the ACM опубликовал статью, в которой излагалась идея монитора (monitor). Ей автор отводил ключевую роль в построении структурированных ОС. Идея мультипрограммирования целиком захватила Вирта.

Первые эксперименты проводились на ставшем уже легендой компьютере PDP-11 фирмы Digital, который обладал 64 Кбайт памяти. За период 1974—1976 гг. (т. е. до отъезда в Xerox PARC) группе Вирта удалось провести исследования основных средств описания и взаимодействия параллельных процессов: прерываний, сигналов, семафоров, мониторов и критических секций. В результате был реализован язык, получивший название Modula (Modular Language). Основу его составляла концепция мониторов в понимании Хоара, обобщавшихся до программных модулей. При этом идея инкапсуляции оказалась отделена от принципа взаимного исключения (mutual exclusion). Обработка прерываний была встроена в диспетчеризацию процессов, и переключение контекстов могло производиться либо с помощью операций работы с сигналами, либо путем прерываний.

Итак, важной идеей языка Modula стал механизм поддержки квазипараллельных процессов (не следует их путать с процессами в UNIX). В современном понимании это были программные потоки (threads), работавшие в общем адресном пространстве и переключавшие свои контексты в четком соответствии с принципом сопрограмм (coroutines). Вскоре выяснилось, что введение средств поддержки мультипрограммирования непосредственно в язык неразумно, поскольку это очень сильно привязывало программиста к аппаратной и операционной платформе. Их можно было вынести на уровень библиотек, что и было сделано в языке Modula-2.

К 1977 г. язык Modula трансформировался в Modula-2. Здесь уже Вирт использовал средства разграничения переносимой (прикладной) и непереносимой (системной) частей языка. Для этого он ввел псевдомодуль SYSTEM, в котором разместились все средства низкоуровневого программирования (включая и функции преобразования типов). Факт импортирования SYSTEM говорил уже о потенциальной системной зависимости данного программного модуля.

Но главной идеей Modula-2 стала концепция модуля, схожая с той, что была сформулирована в языке Mesa (из популярных языков ни Си, ни C++ эту концепцию не реализовали, ограничившись простейшей формой include-директив; лишь Ada, а спустя много лет в определенной степени и Java начали ее использовать). Модульность, опирающаяся на принцип экспорта/импорта программных элементов и на поддержку раздельной компиляции (separate compilation), стала доминантой нового языка: в свете представления Вирта о четкой взаимосвязи программ и оборудования программное обеспечение, как и аппаратуру, разумнее всего было собирать из готовых блоков с четко специфицированными интерфейсами.

Первый компилятор Modula-2, написанный Ван Ли (K. van Le) в 1977 г., имел семь проходов. Второй вариант для PDP-11/40 (в ОС RT-11) появился в 1979 г. и имел уже лишь пять проходов. Его автором стал Урс Амманн (Urs Ammann). Таким образом, на PDP-11 за три года была подготовлена инструментальная платформа для кроссразработки с прицелом на новый компьютер Lilith.

Компьютер Lilith

А что же сам Lilith? Как вспоминал впоследствии Вирт, к счастью для него, вовремя подоспела подмога. В 1977 г. в ETH неожиданно раздался телефонный звонок. Один специалист в области электроники хотел пройти расширенный курс ПО и защитить диссертацию. Звали его Ричард Оран (Richard Ohran); впоследствии он стал директором Modula Research Institute. Загоревшись идеей Lilith, Оран с головой погрузился в работу, напрочь забыв о своей диссертации. Первое, что его поразило, — это концепция P-кода и виртуальной P-машины. При разработке аппаратной части он решил модернизировать ее инструкции с учетом особенностей Modula-2 и реализовать их на микропрограммном уровне. Желание добиться компактности кода (он получил название M-код) привело к идее использования стековой архитектуры, чем-то напоминающей решения, предложенные еще в English Electric KDF-9 и Burroughs B5000.

Стек выражений реализовывался с помощью небольшого набора быстрых биполярных регистров (16-словная SRAM-память). Одно из интересных решений Орана состояло в том, что, несмотря на сравнительно большой набор инструкций (256), часто использовались коды операций длиной 4 разряда. Остальные 4 разряда байта отводились данным (смещению). Инструкции могли быть длиной 1, 2 и 4 байт. За счет стека и гибкой схемы адресации удалось добиться того, что средняя длина инструкций, включая код операции и данные, составляла 10 разрядов. Код для M-машины был в три-четыре раза меньше кода для PDP-11 и i8086, в два-три раза меньше кода для Motorola 68000 и в полтора-два раза меньше кода для National Semiconductor 32000. При проектировании были учтены и ошибки Xerox Alto. Так, в частности, управление растровым экраном отнимало в Alto 50% времени работы процессора. Доступ к памяти в Lilith был обеспечен через 64-разрядную шину, что позволило довести затраты процессора на управление экраном до 3%. Для более эффективной поддержки векторной графики, шрифтов, функций копирования блоков памяти, увеличения точности операций с плавающей запятой и даже управления лазерным принтером процессор Lilith использовал дополнительные микропрограммы.

Lilith состоял из четырех процессорных наборов Am2901, выполненных на плате, а не в кристалле (bit-slice). Частота его была 7 МГц, длина слова и ширина шины 16 разрядов. Оперативная память многопортовая (128 Кбайт). Дисплей, как и в случае Xerox Alto, был вертикальным, размером 15" (592 x 768 точек), в качестве диска использовался Honeywell-Bull D-120 емкостью 10 Мбайт. Ну и, конечно, имелась трехкнопочная мышь.

Вернемся к программному обеспечению. Важной особенностью проекта Lilith являлось то, что работы в нем были максимально распараллелены (ведь свободных рук не хватало!). После получения инструментальной среды для кросс-разработки на PDP-11 был создан новый компилятор Modula-2 уже для Lilith. Лео Гайсман (Leo Geissmann) и Кристиан Якоби (Christian Jacobi) смогли построить четырехпроходный компилятор. Началась разработка операционной системы Medos, которую возглавлял Свен Эрик Кнудсен (S. Knudsen). Эта ОС была достаточно простой (однопользовательской и однозадачной) и ориентированной преимущественно на пакетную обработку. Одновременно с ее отладкой Кристиан Якоби реализовал программную поддержку дисплея и графических окон. Появился текстовый редактор с раскрывающимися меню.

К декабрю 1980 г. силами студентов удалось собрать первую партию из 20 компьютеров (в общей сложности было построено свыше 60 компьютеров, большая их часть осталась в ETH). В 1982 г. появилась сеть на базе Ethernet (3 Мбит/с), сервер и лазерный принтер (Canon LBP-10). Lilith стал первым компьютером в Европе, который полностью использовал возможности лазерной печати, начиная от шрифтов, графики и отсканированных изображений и заканчивая электронными схемами. По мнению Вирта, компьютер Lilith был гораздо предпочтительнее VAX-750, стоимость и сложность которого были неизмеримо выше, чем у Lilith. К тому же он был создан, по сути, руками самих студентов!

После 1982 г. в промышленных кругах возник интерес и к Modula-2. Были созданы компиляторы для персональных компьютеров PC (Logitech, Borland, TopSpeed, Stony Brook), Apple II, Lisa,

Macintosh (MacMETH, Metrowerks), SIRIUS/VICTOR 9000, SAGE II, Amiga, Atari ST (Megamax), рабочих станций IBM RS/6000, Hewlett-Packard (MOCKA), Sun SPARC (Edinburgh), мэйнфреймов IBM (WATCOM), SGI и VAX(MOCKA), Digital Alpha (ModulaWare). По словам Вирта, IBM создала собственный компилятор и запрограммировала свою ОС для компьютеров AS/400 на Modula-2.

Интересно, что компьютеры Lilith были приобретены в свое время фирмой Microsoft. Говорят, они сыграли важную роль в судебных разбирательствах между Apple и Microsoft (в частности, по поводу Windows). Прецедент приоритета сторонней разработки (prior art) убедил тогда судью в необоснованности претензий Apple. После чего "вещественные доказательства" — компьютеры Lilith — оказались в музее Microsoft.

Заключение

За свои работы по проекту Lilith и языку Modula-2 Никлаус Вирт в 1984 г. был удостоен премии Алана Тьюринга (Alan Turing Award), которая вручается лишь раз в жизни и по своему значению в области компьютерных наук стоит в одном ряду с Нобелевской премией.

По иронии судьбы маэстро компьютерной лингвистики не рассматривает язык программирования именно как язык. Он отвергает общепринятое восприятие языка как среды общения между человеком и компьютером. Вирт видит в языке прежде всего абстрактный инструмент для конструирования компьютерных устройств. "В моем понимании, — говорит Вирт, — язык программирования — это неправильно выбранный термин, который многих вводит в заблуждение. Термин "программная символика" (program notation) был бы куда более подходящим".

С момента завершения проекта Lilith прошло уже почти два десятилетия. За это время в лабораториях Вирта были созданы новые технологии исполнения программного кода (OMI, Juice), новые языки (Oberon, Oberon-2) и компьютеры (семейство Ceres). Но речь сейчас не о них. Хотя появление языка Java, Java-процессоров и Java-технологии заставило взглянуть на работы швейцарского ETH совсем по-иному. Главное, на что хотелось бы обратить внимание читателя, — это последовательность Вирта в его стремлении обеспечить органичное единство аппаратной и программной составляющих через разработку соответствующего языка (программной символики), способного стать основой для целого семейства специализированных компьютерных платформ.

Об авторе. Руслан Богатырев — научный редактор журнала “Мир ПК”, гл. редактор приложения “Мир ПК — диск”, bogatyrev@pcworld.ru.