Руслан Богатырёв

Язык как основа архитектуры. Средства кросс-разработки и технологии XDS

Богатырев Р. Язык как основа архитектуры // ComputerWeek-Moscow. 1998. № 21.

В предыдущих статьях из серии "Язык как основа архитектуры" рассказывалось о проектах Lilith и "Кронос", связанных с созданием в Швейцарии (ETH) и России (Kronos Research Group) новой специализированной платформы — языка высокого уровня (Modula-2), операционной системы (Medos, Excelsior) и персонального компьютера (Lilith, "Кронос"). Одним из побочных результатов тех работ стало создание технологии перенастраиваемых компиляторов (XDS) для самых разных платформ. О средствах кросс-разработки и технологиях XDS и пойдет речь в этой статье.

Кросс-разработка

Бум в области создания различных встроенных систем, бытовой и промышленной электроники способствовал тому, что интерес к средствам кросс-разработки ныне разгорелся с новой силой. Дополнительный импульс этому процессу придали компании Microsoft (Windows CE) и Sun Microsystems (PersonalJava, EmbeddedJava, JavaOS), никогда ранее серьезно не занимавшиеся данным сектором рынка. Активно вторгшись вместе с Lucent Technologies (ОС Inferno) во владения таких компаний, как Wind River Systems (VxWorks), Microware (OS-9), QNX Software Systems (QNX, Neutrino), Geoworks (Sokoto) и др., эти два гиганта в значительной степени нарушили статус-кво и образовали мощную технологическую воронку, втягивающую в себя все новые и новые компании.

Методы кросс-разработки, сегодня довольно основательно подзабытые, пользовались немалой популярностью в 70–80-х годах, когда отношение к ресурсам было куда более бережным. В основе их лежит вполне естественная идея разделения инструментальной (где разрабатывается ПО) и целевой платформ (где оно будет использоваться). Чаще всего эти платформы отличаются друг от друга (как правило, для разработки ПО ресурсов не жалеют, а вот эксплуатация должна быть экономичной). Говоря о платформе в случае кросс-разработки, обычно имеют в виду ее аппаратную составляющую, а не операционную и уж тем более не языковую. Хотя логично было бы распространить принцип кросс-разработки и на них (ретрансляция протоколов и API-интерфейсов операционных систем, конвертеры с одних языков в другие).

Ныне признанным лидером в области технологий кроссразработки является компания Metrowerks, хорошо известная своей инструментальной средой CodeWarrior многим программистам, работающим с Mac OS. Несмотря на то что проект CodeWarrior стартовал только в 1991 г. и к 1994 г. воплотился в первом продукте этой серии — CodeWarrior for Macintosh, — за какие-то четыре года был создан мощный комплекс средств кросс-разработки, по своей номенклатуре практически не имеющий аналогов в мире. Чтобы представить потенциал компании и сложность решаемых ею задач, достаточно перечислить аппаратные, операционные и языковые платформы, на которые ориентирован CodeWarrior. Из аппаратных это Intel x86, Intel MMX, K6, AMD-3NOW!, Motorola 68K, CPU32/32+, PowerPC, Motorola AltiVec, Motorola MPC 821/860, MIPS, ARM, StrongARM, THUMB, Hitachi SuperH, NEC V810/V830, NEC VR, Sony PlayStation, 3Com PalmPilot. Из операционных — Windows 95/NT, Mac OS, Mac OS X, Novell NetWare, QNX, QNX/Neutrino, BeOS, PalmOS, PS-X OS, Windows CE, Nucleus PLUS, OS-9, VxWorks. Что касается языков, то это Си, C++, Java и Паскаль.

Виртуальная Java-машина Metrowerks лицензирована компанией Apple Computer и является основной для платформы Macintosh. Из последних новаций можно отметить поддержку Motorola AltiVec (128-разрядного векторного сопроцессора для архитектуры PowerPC), а также интерфейсов Carbon API для новой Mac OS X, ставшей перевоплощением так и не вышедшей Rhapsody. В мае этого года компания Macromedia лицензировала Java-компилятор Metrowerks для поставки в рамках Macromedia Director. А в начале июня 1998 г. было подписано соглашение

между Metrowerks и QNX Software Systems, предусматривающее доминирующую роль инструментария CodeWarrior для перспективной платформы QNX/Neutrino.

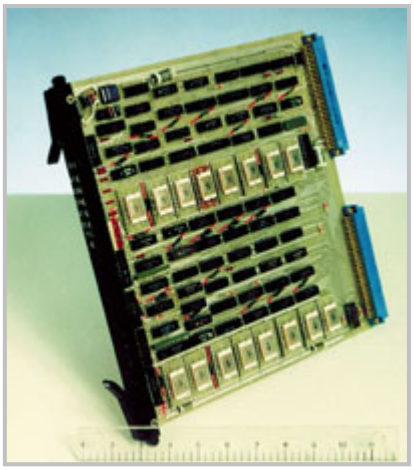
Очевидно, что для быстрой переориентации на выпуск новых инструментальных средств требуется наличие таких технологий (и, разумеется, их организационной поддержки), которые бы позволили в кратчайший срок осуществлять "переналадку конвейера". Опять-таки логичным решением здесь является разделение компиляторов на две части — "инструментальную" (front-end, стадия анализа) и "целевую" (back-end, стадия синтеза). Именно по этому пути одними из первых пошли компания Metrowerks, JPI (TopSpeed Corp.) и ряд других разработчиков промышленных компиляторов.

Своеобразным аналогом компании Metrowerks, но уже в области иных языковых платформ (малораспространенных, заказных или специализированных языков) является российская компания XDS, об истории создания которой шла речь в предыдущей статье.

Технологии XDS

Итак, продолжим прерванный разговор с Алексеем Недорей, одним из инициаторов проекта "Кронос" и нынешним руководителем компании XDS (это данные по состоянию на 1998 г., когда публиковалась данная статья – прим. ред., 2003). Фирма XDS, как и Metrowerks, разрабатывает трансляторы с различных, в том числе и специализированных, языков с ориентацией на самые разные операционные и аппаратные платформы. С чем же связан подобный универсализм, и не в ущерб ли он качеству?

"Я бы не использовал здесь слово "универсализм", — возразил Алексей Недоря. — Наша технология не универсальная, хотя она и позволяет многое делать очень быстро. Мы шли к ней достаточно долго, и об этом стоит рассказать подробнее" (фото 2).



Итак, где-то к 1988 г. специалистам из Kronos Research Group стало ясно, что они не смогут угнаться за технологией разработки аппаратуры. Правда, вначале "кроносы", благодаря своей простой архитектуре, были существенно лучше остальных машин, которыми тогда удавалось воспользоваться. Вставив плату "Кронос-2" вместо LSI-11 в корпус "Электроники-60", пользователь получал гораздо более быструю 32-разрядную машину с несегментированной памятью (flat memory). "Кронос-2.6" был если не быстрее 286-х, то гораздо удобнее для программирования. Только 386-е процессоры обогнали "кроносов" на тестах, но компилятор Modula-2 на "Кроносе" работал все равно существенно быстрее.

Примерно в это же время новосибирцы стали экспериментировать с другими процессорами и с переносимыми компиляторами. Сначала Евгений Тарасов написал генератор кода для i80286. Потом Недоря реализовал генерацию для NS32032. Одной из наиболее удачных конфигураций был "Кронос 2.5" в корпусе Labtam. В этой машине стоял процессор Zilog-80 на вводе/выводе, i8086 обслуживал дисплей, а основной процессор можно было менять, точнее, менять можно было плату. Более того, можно было поставить сразу несколько процессоров. Сама фирма Labtam продавала машины на i80286, Motorola 68000 и National Semiconductor 32032. "Сейчас уже мало кто помнит эти процессоры, — продолжает Недоря, — а ведь они были первыми серийно выпускаемыми 32-разрядными процессорами с довольно удачной системой команд".

Эксперименты, проведенные в Kronos Research Group, показали, что все не так просто: нельзя взять однопроходный компилятор mx и заменить в нем кодогенерацию. Она была слишком сильно связана со стадией анализа. Пришлось проектировать компилятор заново. В 1990 г. был создан переносимый компилятор. Тарасов сделал генератор машинного кода для i80286, а Недоря для "Кроноса".

В январе 1991 г. Алексею Недоре довелось побывать в Цюрихе и познакомиться со всеми участниками проекта Oberon (который вслед за Lilith инициировала группа Никлауса Вирта). Удалось немного поработать на новом языке Вирта (Oberon) и узнать, как устроен OP2 (переносимый Oberon-компилятор). Отношение к языку Oberon в новосибирской команде в

первое время было скептическим, все казалось, что чего-то в нем не хватает. Тогда же стало понятно, что схему переносимого компилятора можно улучшить, и началась его переделка, при этом особое внимание обращалось на то, чтобы Oberon мог быть легко реализован в этой схеме. Кроме того, неожиданно пришла идея использовать в качестве переносимого кода текст на языке Си, и первой реализацией в новой схеме стал транслятор Modula-2 в Си.

Невольно встает вопрос, а что такое переносимый компилятор? Ведь понятие переносимости подчас столь вольно трактуется разными людьми, что не всегда ясно, о чем конкретно идет речь.

Любой компилятор, как отметил Недоря, состоит из двух фаз: анализа и синтеза. На фазе анализа компилятор "понимает" программу, написанную на входном языке, и если есть в ней ошибки, то на этом его работа завершается. На фазе синтеза порождается программа на другом языке, например на языке ассемблера или в машинных кодах (т. е. на языке процессора). Фаза - понятие логическое, а не структурное. Есть много компиляторов, которые анализируют небольшую часть программы и сразу синтезируют выходную часть. Под переносимым компилятором обычно понимается компилятор, который легко переделать под другой выходной язык, т. е. сменить фазу синтеза. Но мы будем рассматривать сразу более общую задачу.

Допустим, у нас есть N языков и M машин и нам нужны компиляторы со всех языков на все машины. Используя лобовой способ, мы должны будем написать N x M компиляторов. Более того, при появлении новой машины придется дописывать еще ь компиляторов. Работа слишком большая и ее необходимо сократить. Идея очень проста: надо разделить компилятор на две части — часть анализа и часть синтеза, написать N анализаторов и M генераторов (слово "синтезатор" имеет совсем другое значение!) и если нам требуется компилятор, то брать нужный анализатор и соединять с соответствующим генератором. Таким образом, необходимо написать всего N + M частей, а если предположить, что трудоемкость написания анализатора и генератора примерно одинакова, то трудоемкость разработки будет (N + M)/2 вместо N x M.

Объяснение Алексея Недори принципов создания переносимых компиляторов, на мой взгляд, чем-то напоминает коммутацию, только в данном случае уже надо говорить о коммутации программных компонентов. Невольно в памяти всплывает идея технологически независимого машинного интерфейса TIMI, который представляет собой решение IBM для компьютеров AS/400, позволяющее подменять реальный процессор. По всей видимости, язык промежуточного представления в технологии XDS в определенном смысле близок той идее. Но ведь неизбежно должны возникнуть проблемы интеграции таких компонентов.

"Действительно, — развивает свою мысль Алексей Недоря, — казалось бы, все хорошо, но мы забыли про то, что части компилятора не совсем изолированные, от анализатора к генератору надо передать информацию о программе". Вот тут-то и проявляется главная проблема. Каким бы ни был способ передачи, нужно уметь решать, как минимум, три задачи:

1) передавать информацию о программе, написанной на любом входном языке;

2) обеспечивать такую структуру информации о программе, которая была бы достаточной при генерировании кода для любой машины;

3) представлять информацию о программе в удобной форме для генератора кода.

Интерфейс между анализатором и генератором — половина успеха технологии построения компиляторов. Вот почему система XDS неоднократно переписывалась с целью добиться универсальности и удобства интерфейса. В 1994 г. технология XDS была реализована на языке Oberon-2 с использованием объектно-ориентированного подхода. Именно эта технология и используется теперь в нынешних разработках компании (семейство компиляторов XDS сейчас сопровождает отделившаяся от нее компания Excelsior: http://www.excelsior-usa.com — прим. ред., 2003).

Сейчас XDS-система поддерживает пять анализаторов (Modula-2, Oberon-2 и три специализированных языка) и шесть генераторов, включая генераторы кода для Intel x86, Motorola 68K, PowerPC, VAX и трансляторы в Си и C++. Причем операционных платформ для выходного кода тоже немало: Windows 95/NT, OS/2, Linux, FreeBSD (все для Intel x86), VxWorks (для Motorola 68K), AIX, Windows NT, VxWorks (для PowerPC). Трансляторы в Си и C++ (так называемая линия Via C) были разработаны для таких операционных платформ, как Windows 95/NT, OS/2, Linux, FreeBSD, SCO UnixWare, QNX, Windows NT (PPC Edition), AIX, Solaris, HP-UX, OSF-1 (Digital UNIX), OS-9, VxWorks.

"Технология XDS и накопленный опыт позволяют нам в короткие сроки выполнять контрактные работы в области разработки компиляторов. Об их качестве можно судить по самим заказчикам. Для известной телекоммуникационной компании Nortel мы стали чем-то вроде компиляторного отделения (compiler department), как компания "Эльбрус-2000" для Sun Microsystems", — подчеркнул Недоря.

Сотрудничество с Nortel

Поздней осенью 1997 г. пришло известие о том, что совместными усилиями канадской компании Nortel (Northern Telecom) и английской Norweb Communications была разработана и запатентована революционная технология 1-Meg Modem, позволяющая обеспечить высокоскоростную передачу данных (1Мбит/с) через сети электропитания. Первое подключение с помощью 1-Meg Modem к Internet было выполнено для английской школы Seymour Park Primary School (Траффорд, Манчестер).

Как оказалось, непосредственное отношение к созданию аппаратуры, поддерживающей новаторскую технологию, имеет и компания XDS (вновь налицо тесная интеграция программных и аппаратных решений). Дело в том, что скоро уже четыре года, как Nortel является крупнейшим партнером XDS. Именно для нее новосибирская команда разрабатывает специализированные инструментальные средства. Более того, московское представительство этой компании соединено прямым каналом с новосибирской фирмой XDS.

XDS сотрудничает с разработчиками из трех отделений Nortel. Калифорнийская группа создает офисную телефонную станцию, построенную на концепции PBX (Private Branch Exchange). При этом старшие модели могут обслуживать до 10 тыс. линий. XDS реализовала для этой группы компиляторы со специализированного языка SL1 в машинный код для процессоров Motorola 68K, Intel x86, PowerPC (для ОС VxWorks). Эти компиляторы порождают высокоэффективный код и позволяют существенно увеличить пропускную способность станций. Кроме того, ведется проект по реализации транслятора с SL1 в C++. Запланировано создание ряда отладчиков, а также реализация компилятора для платформы SPARC.

Группа из Северной Каролины работает над DMS-10 (большой цифровой АТС). Для этой группы продолжается разработка компилятора с SL1 в машинный код PowerPC (но уже для ОС Chorus). Кстати, Chorus недавно была приобретена фирмой Sun Microsystems и положена в основу ее будущей облегченной ОС для встроенных систем.

Группа из Оттавы занимается офисными радиотелефонами и мини-сотовой связью (NorStar, minicircular phones). Для этой группы был реализован транслятор с языка BNR Pascal в C++. В рамках завершившегося проекта было оттранслировано в общей сложности более 1,5 млн строк.

Среди других крупных заказчиков XDS можно отметить и НПО ПМ (Красноярск), для которого разрабатывался инструментарий, используемый для создания ПО бортовых компьютеров российских телекоммуникационных спутников нового поколения. Был реализован компилятор Modula-2 для VAX, а также компиляторы для Windows 95 и трансляторы с ISO Modula-2 в ANSI C (для платформ Digital Alpha и Windows 95).

XDS в основном работает с западными заказчиками. Возникает естественный вопрос, чем это вызвано? Почему фирма столь прохладно относится к российскому рынку?

"Так было. Но сейчас многое изменилось. Мы начинали работать по контрактам в 1994 г., — вспоминает Алексей Недоря. — В тот момент в России было очень трудно найти заказчика, который имел бы интересную для нас задачу и располагал приемлемыми финансовыми средствами. Мы работали с НПО ПМ почти бесплатно, потому что та работа была начата еще Институтом систем информатики СО РАН, а уже потом перешла к нам. К тому же кроме нас создать инструментальный комплекс для бортовых систем никто не мог. В какой-то степени мы работали на будущее. Сейчас ситуация в России заметно изменилась, так что мы заинтересованы и в российских заказчиках".

Участие отечественной компании в разработке заказного и тиражируемого ПО с ориентацией на западный рынок, очевидно, сопряжено с большими трудностями. А есть ли у XDS фирмы-партнеры?

"Да, у нас есть дистрибьюторы, — ответил Недоря, — которые продают наши компиляторы. Это несколько небольших фирм в разных частях света (США, Франция, Германия, Великобритания), и много продать они не могут. Рынок Modula-2/Oberon-2 компиляторов очень мал".

Почему же в качестве входных языков для серии компиляторов и трансляторов (в Си/C++) были выбраны именно Modula-2 и Oberon-2? Оказывается, это связано большей частью с историческими причинами. Команда XDS работает на Modula-2 с 1984 г., а на Oberon — с 1992 г. Эти языки, с точки зрения Недори, хороши практически всем, кроме одного: они очень мало распространены. К тому же пробиться на рынок компиляторов для известных языков (Си, C++, COBOL) да еще с нуля весьма сложно. И все же сейчас фирма XDS добавляет в свою инструментальную среду еще и язык Java (в настоящее время одним из основных продуктов компании Excelsior является оптимизатор JET для Java — прим. ред., 2003).

Заказное и тиражируемое ПО

Сложилось устойчивое мнение, что ниша российских разработчиков — это заказное ПО, что они обладают высокой квалификацией, но не могут самостоятельно довести свое детище до тиражируемого рыночного продукта. Так ли это?

"Я согласен с тем, что российские разработчики успешнее работают в области заказного ПО, — говорит Алексей Недоря. — Для раскрутки тиражируемого рыночного продукта, как правило, нужны довольно крупные деньги. А таких денег у наших команд нет. Можно выйти на рынок с помощью какого-то принципиально нового продукта, но найти такой продукт и нишу непросто.

Что же касается доводки продукта, то здесь я не вижу каких-то особых проблем, специфичных для отечественных разработчиков: в фирме должны быть грамотные специалисты по рынку, должна быть отлажена технология доработки продукта, сопровождения и т. д. Мне достаточно легко сравнивать разработку тиражируемого и заказного ПО, так как наша фирма занимается и тем и другим. У нас есть свои продукты: компиляторы Modula-2 и Oberon-2 практически для всех известных платформ, и, кроме этого, мы делаем заказное ПО по контрактам.

Я вижу только одну принципиальную разницу в разработке двух видов ПО: если при разработке заказного ПО заказчик знает о том, что ему надо, и может выдать спецификации, то при разработке тиражируемого ПО надо видеть тенденции развития рынка, а это не всегда просто".

Новосибирский Академгородок

Вот и подошел к концу наш разговор с Алексеем Недорей, одним из инициаторов проекта "Кронос" и руководителем компании XDS. Но все же осталась какая-то недоговоренность. Что-то очень важное, о чем речь так и не зашла. Хотелось из уст практика-профессионала услышать о его отношении к нынешней и будущей структуре таких инженерно-научных центров, каковым является новосибирский Академгородок, где и зародились проект "Кронос" и технологии XDS.

Известно, что создавался он по образу и подобию американского Станфорда (Stanford University). У истоков Академгородка стояли Михаил Алексеевич Лаврентьев (первый президент Сибирского отделения АН СССР), Сергей Львович Соболев и Гурий Иванович Марчук. Именно в это уединенное место тогда съехались ведущие ученые из Москвы, Ленинграда, Киева, Иваново, Уфы, Тбилиси. За ними последовали и их лучшие ученики. В кратчайшие сроки был создан научный центр мирового класса. В Академгородке возникали все новые и новые центры и институты. Если я не ошибаюсь, то как раз после посещения в начале 60-х новосибирского Академгородка Шарль де Голль принял решение организовать подобные центры во Франции: там появились Орсей и Экс-Марсей.

Структура и развитие Академгородка интересны прежде всего тем, что сейчас как никогда остро стоит проблема существенного сокращения сроков выхода новейших технологий на рынок

компьютерной индустрии. Маркетинговая сторона дела изучена и неплохо проработана (тому есть масса примеров). А вот что касается организационной и технологической — здесь все еще немало белых пятен. Известная разнородная цепочка "университет — лаборатория — фирма", к сожалению, грешит крайне длительным сроком внедрения идей (от 7 до 20 лет) и довольно низким кпд. Ряд компаний уже не устраивает структура технопарков и технологических оазисов (вроде Кремниевой Долины в США и ее младшей сестры во Франции), оторванных от университетов (где чаще всего и зарождаются наиболее плодотворные идеи).

Часть своих исследовательских лабораторий фирмы переносят поближе к университетам (в том числе из США и Японии в Европу). Однако продуктивность работы все еще оставляет желать лучшего, ведь для создания инфраструктуры и необходимой творческой атмосферы нужны годы. В этом смысле модель новосибирского Академгородка с его богатыми традициями представляется одной из наиболее перспективных. Но это в идеале, а как обстоят дела в нашей суровой действительности?

"Я думаю, — подтвердил мои догадки Недоря, — что нужна очень большая переделка Академгородка для организации взаимовыгодного сотрудничества образования, науки и производственных фирм. В настоящее время мы имеем бедный универститет, большие и очень бедные институты и несколько производственных компаний, пытающихся использовать потенциал Академгородка.

Институты при этом почти не изменились, это по-прежнему довольно странные образования, в которых сотрудникам платят очень мало денег и не требуют ничего взамен. Чтобы получить отдачу, нужна коренная переделка всей системы.

Какая структура могла бы работать? Я вижу это так: институты становятся очень маленькими (10–15 научных сотрудников в каждом), причем главная задача такого института — преподавание. Преподавательская нагрузка на каждого научного сотрудника должна быть невелика, с тем чтобы он мог заниматься наукой, а зарплата должна быть существенной, чтобы место оказалось престижным.

Университет со временем становится центром Академгородка, все институты финансируются через университет. Рядом с ним организуется технопарк, который поддерживает молодые (spin-off) компании. Каждая вновь образованная фирма может получить офис и все необходимые услуги (доступ в Internet, оборудование и т. д). Основная исследовательская работа идет в этих компаниях".

Да, многие западные специалисты в области компьютерных наук (и не только) по-настоящему озабочены сейчас "вымыванием" из университетов фундаментальных исследований. Работа в угоду сиюминутным прикладным задачам, активная борьба за будущих специалистов с привлечением их к коммерческой деятельности отрицательно сказываются на общем уровне подготовки ученых и инженеров. Более того, даже крупные компьютерные компании не могут себе позволить такую роскошь, как финансирование фундаментальных исследований, а потому многие решения выполняются большей частью на основе того богатого задела, который был создан в предшествующие два десятилетия.

Длительный срок проведения фундаментальных работ и спокойную творческую обстановку проще всего обеспечить именно в университетах. Однако для этого требуется государственная поддержка. По крайней мере сегодня ею могут похвастаться несколько европейских стран (те же Швейцария и Германия). В любом случае настают времена, когда нужно изменить свое отношение к сложившейся практике передачи знаний и технологий. Быть может, наш собственный богатый опыт сослужит на сей раз добрую службу?

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