

Мечта программиста

Андрей Терешко

Один программист, пишущий на С и С++, написал статью, где были такие слова: "Контроль указателей, защищенная область памяти, автоматическая сборка мусора и переносимость скомпилированного кода до сих пор являются мечтой огромного числа программистов."

И вот она пришла эта мечта, пришла не во сне, а наяву, но "огромное число программистов" ее так и не заметили. Вернее не то, чтобы не заметили, не заметить ее очень даже проблематично, ибо речь о ней, как и об Интернете повсюду, но не приняли душой и сердцем и отвергли разумом.

О чем мечтает программист? О разном. Чего он хочет? Хочет так приспособить свой инструмент для работы, чтобы был удобен, был "по руке", был разумен и ясен и чтобы не раздражал. Чтобы даже по истечении длительного срока можно было быстро въехать, посмотрев свой исходный текст. Чтобы чужой текст и чужая документация не казались текстом инопланетян. И чтобы можно было зарабатывать себе на жизнь.

Те программисты, которые программируют на С или С++, но душой к нему не прикипели, которых раздражают все проблемы связанные с программированием на этом языке, но которых почти устраивает почти ясный синтаксис этого языка, не раз задумывались о том, что могло бы быть в этом языке, но чего нет, и, возможно, уже никогда и не будет.

Язык мечты программиста был взят не с потолка, а сознательно создан тогда, когда раздражение от программирования на С++ у одного из программистов достигло предела, и он решил бросить свою компанию и работу. Однако руководство компании попросило его составить письмо, с изложением того, чего же он хочет. И, что было самое удивительно, по его письму было принято решение образовать группу из 6 человек для создания мечты программиста. И они ее "эту мечту, создали.

Идеи, которые воплотила эта группа, не являются революционными. Эти идеи уже

существовали в различных исследованиях и разработках. Да и сами, разработчики мечты, очевидно, хорошо изучили Eiffel, Ada, Smalltalk, Modula, Objective C и т.д. И, в отличие от академических ученых, разработчикам приходилось сразу же, с колес, на практике применять свою мечту в программировании ... чайника. Ну, не совсем чайника, это я пошутил, но что-то типа PDA, названное "ручным пультом дистанционного управления".

Удивительные превращения претерпевает эта мечта программиста в своем развитии. Программное обеспечение, созданное фактически для чайника, пылесоса, холодильника или видеомэгнитофона вдруг стало использоваться для создания динамического интерфейса при путешествиях с помощью броузера по Интернет. Не прошло и нескольких лет, как эту мечту программиста объявляют одним из основных инструментов для бизнеса, чуть ли не Коболом начала 21 века (даже появились программы для автоматического преобразования программ на Коболе). Но корни свои мечта не забывает"начинает использоваться и в пластиковых карточках, и в игровых приставках, и приставках для путешествия по Интернет и даже для программирования роботов, созданных из кубиков Lego. А почему?

А потому"что мечта осуществилась. Язык был создан если и не в гараже, то уж не по заказу военных (как Кобол или Ада) или ученых (как Fortran). Язык был создан практическими программистами для себя, совсем как C, который был разработан в начале семидесятых годов как инструментальное средство для реализации операционной системы Unix на PDP-11"ничего не скажешь, развитие идет по спирали. И как в свое время C, вобравший все лучшее в то время, был создан для упрощения программирования по сравнению с ассемблером, так и мечта программиста вобрала все лучшее, что было известно к концу 20 века в языках программирования, упростив программирование по сравнению с языком C и C++.

Те программисты, кто устали от бесконечных проблем языка C и C++, от неявных преобразований, от "проблемы хрупкости базового класса" (проблемы "постоянной перекомпиляции") которые жаждали ясности и простоты, без снижения мощности языка, которые хотели "жить" в новом объектом мире, мире, где не надо постоянно думать о переполнении буфера (или выхода индекса за пределы массива) и постоянно латать программы, когда какой-то хакер не найдет это переполнение в программе и мгновенно не известит об этом весь Интернет, те программисты восприняли эту мечту.

Мечта программиста

Те, кто жаждал стандартных библиотек ввода/вывода, нитей, графического интерфейса, классов-контейнеров и программирование в сети, те получили ожидаемое. Все это было уже, но "жемчуг" был рассыпан по различным закромам. Собрать все воедино, соединить и стандартизировать"работа трудная, в других языках проходившая годами"была сделана и, что удивительно, этот процесс стандартизации не прекращается"стандарт на API шифрования, работа с e-mail, 2D и 3D графикой, базами данных, доступом к архивам, звуком и т.д. (Под стандартом здесь понимается поставка в "едином флаконе" очередного номера версии, а не продолжительное, годами измеряемое, ожидание стандартизации в академической или какой-другой комиссии по стандартизации. Время не ждет, пока "старики-академики" проснутся.)

Те, кто хотел работать с простой и надежной моделью памяти, где исключена возможность "грохнутья" не так высчитав указатель, получили такую возможность, а не только возможность самому сделать такую модель.

Те, кто устал таскать за собой интерфейсы на все "случаи жизни", то есть на все платформы (от чайника до сервера), те получили возможность исполнения кода на множестве платформ.

Те, кто решил глубже погрузится в предметную область, модель которой они и программируют и за которую с них и спрашивает, в конечном счете, заказчик, оставили повышение производительности узким специалистам, которые постоянно совершенствуют исполнения кода, и программы начинают "летать" без какого-либо переписывания исходного кода. Повышенная мощь компьютеров плюс развитие технологии компиляции позволяет компилятору самому решать, сколько делать переходов по исходному тексту и когда метод должен быть вставлен inline.

Те, кто устал, сначала изучать фактически "чужой" (или каждый свой, каждый под себя) язык препроцессора, без которого, просто читая текст программы, понять что-либо было затруднительно, те с изумлением и радостью узнали, что препроцессора просто нет. (Замечу, что программисты, которые без препроцессора просто жить не могут, оказывают такой прессинг на владельцев мечты, что те, не сегодня так завтра, могут дрогнуть и включить препроцессор в стандарт. Вне стандарта препроцессоры уже реализованы сторонними компаниями, но особым восторгом среди программистов не пользуются.)

Те, кто уже привык, хоть теоретически, мыслить объектами (ох, как это не просто перейти от процедурного мышления к объектному), те получили объектный язык "с самого начала". Однако, какой же нажим идет от "процедурников" на то, чтобы владельцы мечты включили в стандарт функции типа `printf`, `scanf` и `sscanf`? так сильно желание привычки и не желание (или не понимание) мыслить объектами, используя методы классов `Format` (и его наследников) и `StreamTokenizer`.

Кто понял, что множественное наследование обычно заводит в тупик, выход из которого только в отказе от множественного наследования, те получили взамен понятие интерфейса. Для множественного наследования есть две проблемы: конфликт имен и повторное наследование. С ними борются, довольно оригинально, либо считают конфликт имен ошибкой (`Smalltalk`), либо, в основном, добавкой к именам префиксов, указывающих на имена классов (`C++`) или "обобществляют" повторно наследуемые классы. Идея интерфейса пришла в мечту, очевидно, из языка `Flavors`, где используется понятие примесей (`mixin`). Надо отметить, что необходимость программировать весь набор методов интерфейса вызывает некоторое недовольство в среде программистов, требующих сохранить интерфейсы, но дать возможность не программировать весь (без изъятия) набор методов интерфейса. Взамен они получили понятие адаптера (уловка программирования).

Те, кто ожидает шаблонов, возможно, вскорости их и получит. Вариантов реализации шаблонов много, но выбрать такой, чтобы вписывался в идеологию мечты, чтобы очень сильно не усложнял и не отпугивал программистов, пока не получается. Но сторонние компании уже предлагают свои библиотеки реализации шаблонов. Радует все же что шаблоны (регулярные выражения) не будут обязательны для применения в программах, без них вполне можно обойтись, оставив исходный код, если уж и не компактным, то простым и ясным, в чем сила мечты.

Как же называется эта мечта?"спросит читатель. А как называется тот сосуд, та бочка, куда залито молодое вино, выжатое из спелейших ягод, впитавших в себя все соки земли и свет солнца, снятых с прилежно и трудолюбиво возделываемых на протяжении многих лет виноградных лоз? Как называется та новая бочка, в которой залили новое вино, которое с течением времени набирает необходимую крепость и неповторимый вкус? Спросите об этом у виноделов, и они Вам, уважаемый читатель, ответят, что называется эта новая бочка — `newoak`. Или просто `oak`. Желаете отведать

Мечта программиста

этого напитка (уже пятилетней выдержки!) и приобщиться к мечте программиста"попробовать можно с сайтов: <http://www.javable.com/> и <http://java.iba.com.by/javaweb/ibajavat.nsf>. Замечу, что с целью привлечения к "напитку" "трезвенников" коммерческое название его несколько изменили, не изменив, однако, сути.

Возможно стиль этой статьи и носит несколько эмоциональный характер, но как же рассказывать без эмоций о мечте программиста!