СИСТЕМА ОБРАЗОВАНИЯ КАК ФАКТОР НАЦИОНАЛЬНОГО СУВЕРЕНИТЕТА В СФЕРЕ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Ф.В.Ткачев
Институт ядерных исследований РАН, Москва
Координатор общественного проекта Информатика-21

Доклад представлен на Совещание <...>, октябрь 2006 г.
В данном публичном варианте текста сделаны некоторые купюры.

Аннотация

В докладе суммируется пятилетний опыт проекта Информатика-21 и дается комплекс конкретных предложения по исправлению неудовлетворительной ситуации в сфере образования по тематике информационных технологий (ИТ). Суть предложений — незамедлительное создание единой системы вводных курсов основ информатики, охватывающей старшие классы средней школы и младшие курсы университетов, подобно базовым курсам весьма успешной системы преподавания математики.

Общая посылка анализа состоит в том, что **образование — один из ключевых рычагов контроля над всей сферой ИТ**, определяющий в конечном счете общий уровень качества, надежности и безопасности быстро растущего массива программного обеспечения (ПО), от которого все больше зависит нормальное функционирование нашего социума.

Утверждение о ключевой роли образования доказывается простой ссылкой на активность, которую проявляют в этой области *все* крупные корпорации сферы ИТ, действующие в России: Майкрософт, Интел, Борланд ... Эта активность не имеет аналога в других отраслях индустрии, и поэтому важно выяснить, в чем смысл и каковы долгосрочные последствия этой активности.

Без выстраивания эффективной *системы* ИТ-образования невозможно приблизиться к решению больной проблемы нехватки хорошо подготовленных программистов (как «чистых», так и специалистов других профессий, серьезно использующих компьютеры).

Наконец, ***без четкого контроля над образованием нельзя обеспечить национальный суверенитет в сфере ИТ***: для страны с уникальной математической традицией, где воспитываются победители международных чемпионатов по программированию, ***ограничить целеполагание оффшорным программированием — недостаточно*** в долгосрочной перспективе. Оффшорное программирование — это, по сути, эквивалент в сфере ИТ экспорту сырья в традиционной экономике, с аналогичными долгосрочными последствиями (зависимость от зарубежных центров управления и влияния, и т.д.).

Подробнее о проекте Информатика-21 см. в тексте данного доклада.
В проекте в качестве консультантов участвуют: ***Н.Вирт*** *— пионер информатики и тьюринговский лауреат, автор языка программирования Паскаль;* ***А.А.Колташев*** *— начальник отдела системного программирования для космических аппаратов НПО ПМ, г. Железногорск;* ***Н.В.Чистяков*** *— главный конструктор комплексов дистанционно пилотируемых летательных аппаратов (ДПЛА); и др.*
Основной документ проекта Информатика-21 — текст [10] в списке источников в конце данного доклада.

Консультанты проекта Информатика-21

Никлаус Вирт — пионер научной информатики и ИТ-образования, ведущий мировой специалист. За выдающийся вклад в становление научной информатики он получил высшую награду в мире ИТ — премию им. Тьюринга (1984 г.).
Н.Вирт — автор классических учебников, переведенных на многие языки мира, в том числе на русский язык («Алгоритмы и структуры данных», «Программирование на языке Модула-2»).
Его идеи оказывают глубокое влияние на мировую индустрию ПО (например, сразу два его ученика — Клеменс Шиперски и Ральф Зоммерер — являются сотрудниками в стратегическом исследовательском подразделении Майкрософт).
Созданные им языки программирования Паскаль, Модула-2, Оберон (см. Приложение 1) серьезно используются в российском образовании и индустрии.

А.А.Колташев — начальник отдела системного программирования для космических аппаратов Научно-производственное объединение прикладой механики им. Решетнева, г. Железногорск Красноярского края. (НПО ПМ — главное спутникостроительное предприятие России; акад. М.Ф.Решетнев — сподвижник С.П.Королева и основатель НПО ПМ.)
Разработал принципы и технологии построения бортового ПО для спутников связи (в т.ч. ГЛОНАСС), руководил созданием ОС для первого советского стационарного спутника с БЦВМ (1981). Руководил созданием действующей технологии разработки БПО для данного приложения, в т.ч. уникальной мобильной кросс-системы программирования на основе виртовской Модулы-2.
С 1990 г. читает курс «Технологии программирования» сначала в Сибирской Аэрокосмической Академии, затем в Красноярском гос. тех. университете. Доцент каферды АСОИУ КрГТУ. Профессиональная обязанность и научный интерес — создание мобильной технологии программирования с гарантированным уровнем качества для встроенных компьютеров.

Н.В.Чистяков — Главный конструктор комплексов дистанционно пилотируемых летательных аппаратов (ДПЛА): комплекса «Строй-П» с ДПЛА «Пчела», известного по антитеррористическим кампаниям, комплексов «ГрАНТ», «БРАТ» и других. Основатель Научно-производственного конструкторского центра «Новик-XXI век». Программное обеспечение системы управления, воплощающее в себе весь опыт и знания конструкторского коллектива, — это мозг беспилотного летательного аппарата, а его проектирование и разработка — обязанность и привилегия Главного конструктора.

Автор доклада **Ф.В.Ткачев** — доктор физ.-мат. наук, ведущий научный сотрудник Отдела теоретической физики Института ядерных исследований РАН, руководитель научной темы «Вычислительные методы и информационные технологии в физике высоких энергий». Специалист по математическим методам физики элементарных частиц, автор ряда основополагающих вычислительных алгоритмов. Входит в число наиболее цитируемых российских ученых.
С 2001 г. читает специальный курс «Введение в современное программирование» для студентов теоретических кафедр Физического факультета МГУ им. Ломоносова.
В 2001-2003 гг. прочел экспериментальный курс информатики для группы школьников 9-11 классов в лицее г. Троицка Московской области.
Представляет РФ в программном комитете международных конференций Joint Modular Languages Conference (JMLC'2003, JMLC'2006).

Никто из консультантов проекта не несет ответственности за конкретные формулировки настоящего доклада.

Автор доклада благодарен:
А.И.Попкову, региональному координатору проекта Информатика-21 по Сибири, за помощь по исправлению ошибок и редактированию данного текста;
И.А.Дехтяренко, участнику проекта Информатика-21, за некоторые полезные ссылки, использованные в данном тексте.

Структура доклада

Доклад состоит из основной части, приложений, а также списка источников.
В квадратных скобках даются ссылки на список источников в конце доклада.

Содержание

Дополнительные материалы

1) А.А.Колташев, Н.В.Чистяков, А.И.Попков, Н.П.Кучер, Ф.В.Ткачев. Система ИТ-образования с точки зрения российских национальных интересов и научно-образовательных традиций. - Доклад проекта Информатика-21 на 6-й Международной конференции «Проблемы систем информатики», Новосибирск, 27-30 июня 2006 г. (см. [10] в списке источников).

2) С.З.Свердлов. Об использовании языка Оберон для программирования встроенных систем высокой надежности.

3) Ф.В.Ткачев. Преподавание информатики: российская перспектива. - Доклад на международной конференции JMLC'2003 (см. [8] в списке источников)

О проекте Информатика-21

Общественный научно-образовательный проект Информатика-21 существует с сентября 2002 г. и имеет публичный сайт в интернете [1], а также интернет-форум для преподавателей [2].

Проект является на данный момент единственным «игроком» на поле ИТ-образования, в котором делается попытка **систематически учесть долгосрочные, стратегические национальные интересы РФ в сфере ИТ**, в том числе интересы стратегических отраслей индустрии и науки РФ.

Другие влиятельные игроки на поле ИТ-образования — это, во-первых, российская ИТ-индустрия, подавляющее большинство компаний которой является дистрибьюторами зарубежных марок [14], и которая преследует, как и любой нормальный бизнес, прежде всего краткосрочные цели. Неудивительно, что на первый план выходит оффшорное программирование, подчиненное в отношении стандартов, платформ и т.п. экстра-территориальным интересам. Однако следует четко понимать, что оффшорное программирование является в сфере ИТ аналогом сырьевому бизнесу в обычной индустрии, с аналогичными как положительными (в краткосрочном плане), так и нежелательными долгосрочными последствиями.

Во-вторых, это зарубежные корпорации, прямолинейно преследующие свои собственные интересы, довольно цинично эксплуатируя «российскую специфику» — готовность преподавателей и чиновников системы образования за ничтожные «гранты» превращать лучшие российские учебные заведения в профтехучилища для западных корпораций в ущерб основательному образованию, дающему специалисту возможность поддерживать собственную конкурентоспособность на протяжении длительного времени (см. об этом ниже). Для сравнения отметим, что *ни один* из большого списка курсов по профилю ИТ, объявленных в ведущем американском Массачусетском технологическом институте MIT [18], не имеет «промышленной» направленности. Кроме того, невозможно вообразить, что

та же Майкрософт будет так же прислушиваться к российскому правительству в вопросах национальной безопасности, как она это делает в отношении американского [16].

Вокруг проекта Информатика-21 сформировалось **растущее сообщество** программистов и преподавателей как в России (территориальный охват — от Брянска, Вологды и Орла до Магадана), так и в странах СНГ (Беларусь, Киргизия, Казахстан, Украина).
В качестве консультантов участвуют представители **одной из ведущих в мире швейцарской школы информатики** (Н.Вирт и Ю.Гуткнехт).
В идейно-технологическом плане проект связан с **международным, главным образом, европейским сообществом** программистов, сложившимся вокруг школы Вирта и Гуткнехта (см. презентации на рабочем совещании в европейской лаборатории ядерной физики CERN [6]).

Проект опирается на основательный научный анализ проблематики производства ПО: главным консультантом является признанный пионер научной информатики и ИТ-образования, ведущий мировой специалист Никлаус Вирт.

Предложения проекта имеют **системный характер** и суммированы в конце доклада в разделе «Выводы для системы образования» (конкретные списки мероприятий даны в Приложении 2). Наши предложения:

— отражают **многолетний опыт** разработки ПО с предельно высокими требованиями на качество и надежность (спутники связи [3], беспилотные летательные аппараты [4], фундаментальная ядерная физика [5]);

— являются **конкретными** (вплоть до уникального комплекта русифицированного ПО для школ и вузов, сборника задач по программированию и др.);

— являются **реалистичными** (например, в части образования они апробированы в коллективной педагогической практике участников проекта на школьном и университетском уровнях, начиная с 2001 г.).

Проект Информатика-21 уникален как систематическая попытка **правильно сбалансировать приоритеты** и учесть реальности:

— не только индустрии, но и преподавания;

— не только профессионального программирования, но и программирования как интегрального компонента работы инженеров, ученых и др.;

— не только обучения профессиональных программистов, но и широкого круга специалистов других профилей (физиков, химиков, строителей, экономистов, лингвистов ...);

— не только университетского преподавания, но и общего школьного;

— не только немедленные потребности индустрии, но и стратегические перспективы долгосрочного развития индустрии ПО в Российской Федерации.

За пять лет существования проекта были услышаны, вероятно, все возможные вопросы к проекту. Можно заключить, что вопросы со стороны ИТ-профессионалов возникают по причине непонимания специфики «чужих» сфер, без учета которой системное решение невозможно. Таких сфер, трудных для понимания ИТ-профессионалами, две.

Во-первых, ИТ-профессионалы имеют неадекватное представление о проблемах, связанных с программированием, у специалистов других профессий («**программистов-непрофессионалов**»: инженеров, физиков и т.д.), а таких специалистов большинство (по оценке Майкрософт, их в три раза больше, чем профессионалов [11]). Более того, катастрофы типа обрушения Трансвааль-парка в 2004 г. происходят как раз по причинам, связанным с использованием сложных программ «непрофессионалами», — ведь такое использование подразумевает приличный уровень понимания программирования, иначе может быть нелегко даже корректно проинтерпретировать документацию такой программы.

Подчеркнем, что именно после публичной постановки проектом Информатика-21 задачи систематического обучения «непрофильных специальностей» (при открытии сайта проекта в сентябре 2002 г.) эта тема привлекла внимание московского отделения Майкрософт (контакты с проектом представителя по связям с университетами В.С.Люцарева), вылившееся в целую программу Майкрософт (серия докладов на образовательных конференциях с весны 2003 г., см., например, [15]), а также запуск с середины 2005 г.

специальной линейки продуктов Express.
Наконец, совсем недавно намерение уделить самое серьезное внимание на этот сегмент рынка было заявлено как одна из целей проводимой реструктуризации известного производителя инструментальных средств Борланд [26].

Вторая трудная для понимания ИТ-профессионалами область — **школьное образование**. Здесь типично практически полное непонимание школьной среды (возможности и способности не только школьников, но и учителей). Эта среда настолько специфична, что нельзя принимать всерьез рекомендации человека, не проучившего типичную группу школьников хотя бы в течение года. (Заметим, что суммарный опыт проекта Информатика-21 в этом пункте гораздо больше.)

Можно утверждать следующее:

> ***Без учета потребностей программистов-непрофессионалов,***
> ***а также специфики школьного образования,***
> ***нельзя решить проблему построения* системы *ИТ-образования.***

В настоящее время просто нет альтернативы основным предложениям проекта Информатика-21 по исправлению ситуации с ИТ-образованием, которая столь же полно учитывала бы весь спектр проблем и интересов (в деталях, конечно, возможны варианты).

Апробация. Первое публичное обсуждение идей проекта состоялось на международной конференции Joint Modular Languages Conference, Австрия, 2003 [8].

Программный доклад [10] был представлен на конференциях по проблемам ИТ-образования в МГУ (июнь 2005) и в Институте систем информатики им.А.П.Ершова СО РАН (июнь 2006), где он вызвал живейший интерес преподавателей (дискуссия на заключительном круглом столе была переведена участниками на обсуждение проекта Информатика-21).

Наконец, процитируем еще один отзыв на доклад [10] одного из руководителей петербургской школы призеров мировых чемпионатов по программированию:

> «*Прочитал Вашу статью. Согласен с каждым словом.*»
> — А.А.Шалыто, профессор СПбГУ ИТМО, зав. кафедрой технологий программирования, электронное сообщение от 2005-09-19.

Следует особо подчеркнуть, что проект Информатика-21 возник и развивается как ответ на больную проблему нехватки квалифицированных специалистов с «правильно поставленными мозгами» в отношении качественного построения программных систем. В проекте нет ни одного предложения, выдуманного из отвлеченных соображений или из соображений педагогического экспериментирования. Весь комплекс предложений представляет собой **конкретный систематический ответ на конкретные реальные проблемы** и опирается на широкий реальный опыт.

Что сейчас «не так» с преподаванием информатики

Сейчас работающая схема непрерывной подготовки «школа-вуз-предприятие» существует, видимо, только в СПбГУ ИТМО [19], но она ориентирована сугубо на подготовку высококлассных ИТ-профессионалов и там почти «штучная» работа со школьниками. Есть еще несколько университетов, где есть программы подготовки программистов под конкретного заказчика — крупного мирового производителя в ИТ-индустрии. Но эти очаги ИТ-образования функционируют разобщенно, и их недостаточно.

Однако на самом деле задача гораздо шире [10] — хотелось бы ожидать, чтобы *любой* хороший студент физико-математических, технических, естественнонаучных специальностей, пройдя через курсы информатики в школе, а затем в вузе, имел некий прочный минимум знаний о программировании на обычных императивных языках. Не просто что-то знал о синтаксисе одного конкретного языка, а четко понимал ключевой набор базовых понятий и методов программирования (включая некий минимум архитектурных аспектов организации больших программ) так, чтобы его можно было научить работать в любой системе программирования, просто пройдясь по особенностям системы и синтаксиса соответствующего языка. Но даже на физическом факультете МГУ, куда приходят достаточно сильные школьники из хороших школ, знания по ИТ-проблематике, которые у них остаются к 4-му году обучения, — это, в лучшем случае, набор специфических умений, в основном вторичной важности, на которые почти невозможно опереться.

Вкратце перечислим основные конкретные пункты критики имеющегося положения дел, очевидные всем серьезным преподавателям:

— хаос с языком программирования для вводных курсов всех уровней, усугубляющийся тем, что заслуженно популярный Турбо Паскаль давно устарел, и в образовавшуюся брешь «прорываются» совершенно неприемлемые C/C++ и Бейсик (см. об этом ниже);

— дикая мешанина в программе из знаний фундаментальных (опорных, структурообразующих, вечных) и знаний сугубо прикладных и конъюнктурных, которые устареют лет через пять (не говоря уже о полной чуши вроде определений курсора как «светового пятна, которое...»);

— чрезмерная доля «мусорных» знаний и навыков, никак не тянущих (в отличие от математики) на роль «гимнастики ума», и которые не служат опорой для дальнейшего обучения;

— устаревшее содержание (чрезмерное внимание в алгоритмике темам, когда-то актуальным в научном программировании, в ущерб таким фундаментальным темам как «структурирование информации» [32], «динамические структуры данных», и т.п.).

— разрыв между школой и вузом: в вузовских вводных курсах (да и не только вводных) сейчас невозможно предполагать никаких «правильных» знаний у вчерашних школьников, так что теряется как минимум целый семестр; по этой же причине трудно организовать осмысленный вступительный экзамен.

Более того, даже если школьники прошли через какие-то школьные курсы программирования, «постановка мозгов» им в школе (за редкими исключениями) нередко оказывается выполненой настолько неправильно (укажем на трудноискоренимую привычку программировать «методом тыка», «сидя» в пошаговом отладчике), что последствия ощущаются *годы* даже при активном переучивании (именно эта проблема явилась конкретной причиной нашего интереса к проблематике школьных курсов информатики и возникновению проекта Информатика-21; см. доклад [8]).

О «постановке мозгов» будущим программистам

Вспомним, о том, какую определяющую роль играет постановка базовой техники в спорте, музыке, балете и т.п.: правильную технику нужно ставить с самого начала обучения, т.к. переучивать потом бывает слишком трудно, зачастую практически невозможно. В музыке система музыкальных школ и училищ обеспечивает надлежащую начальную подготовку практически всем будущим профессионалам. И миновать эту систему практически невозможно. Аналогичная система существует и в спорте. Тем более подобная система должна существовать для такой сложной интеллектуальной деятельности, как программирование.

> ***Говорить без этого о безопасности ПО или о стратегическом развитии ПО — значит, строить дом на песке.***

В конечном счете смысл постановки правильной техники при первоначальном обучении во всех профессиях — *эффективность и качество профессиональной деятельности*.
Для программирования правильное начальное обучение по крайней мере не менее критично:

> *Изначально **правильная** «**постановка мозгов**» в программировании обеспечивает, в конечном итоге:*
> *— ощутимый (кратный) рост **производительности**;*
> *— ощутимый (кратный) рост **качества** (надежности, безопасности, эффективности) создаваемых программ;*
> *— четкий ориентир и **легкость освоения новых инструментов** (языков программирования), что является необходимой частью программистской деятельности, но, к сожалению, **не артикулируется как цель основного образования**.*

Первые два пункта совершенно аналогичны тому, что достигается при своевременной правильной постановке базовой техники в любых других профессиях. Третий пункт в списке отражает быстрое развитие ИТ и специфичен для проблематики ПО (автору, отнюдь не занимающемуся программированием 100% рабочего времени, за свою учебно-научную карьеру пришлось поработать более чем с десятком разных языков программирования, не считая нескольких вариантов фортрана и бейсика).

Сущность правильной «постановки мозгов» программиста — навык систематического конструирования программы с опорой на четкую логику (следуя Дейкстре, Вирту и др.), а не «методом тыка» с использованием пошагового отладчика. Но именно стиль «методом тыка» развивается при спонтанном (само)обучении: без специальных целенаправленных усилий, без ясных ориентиров, и при обучении на непригодном для целей образования «промышленном» инструменте.

К сожалению, именно так происходит (само)обучение подавляющего большинства современных программистов. В результате,

> *«... те молодые люди, которых набирают сейчас в наши фирмы — это обычное народное ополчение в области программирования, а то и **просто партизаны,** которые вообще не обучены.»*
> — В.Г.Парфенов, декан факультета информационных технологий и программирования СПбГУ ИТМО, см. [19].

Если выполнить начальную «постановку мозгов» неправильно, переучивать потом будет чрезвычайно сложно. Это известно давно, и постоянно подтверждается практикой преподавания. Вот, например, выдержка из дискуссии на Конференции НАТО по разработке ПО в 1969 г. после доклада создателя структурного программирования Э.Дейкстры, лауреата премии им. Тьюринга за 1972 г., см. [17]:

> ***Перлис:** Как вы думаете, можно ли обучить вашим методам программистов, которые всегда сначала кодировали, и только потом думали?*
> ***Дейкстра:** ... Мой опыт говорит, что **интеллектуальная деградация** вследствие некоторых методов обучения — серьезное препятствие для ясного мышления. От*

студентов, желающих работать со мной, я требую, чтобы они не были знакомы с фортраном. Я не шучу.

Выводы Дейкстры подтверждаются уже в течение более чем тридцати лет. Можно сделать следующий вывод:

***Смысл** вводных курсов программирования — **правильная** «**постановка мозгов**» учащимся, т.к. при спонтанном обучении приобретаются «дурные привычки», сильно влияющие и на производительность, и на качество результата.*
*Опыт показывает, что **искоренять спонтанно возникающие дурные привычки чрезвычайно трудно**.*

Важен вопрос, с какого возраста нужно начинать обучение. Известно, что обучать «настоящему» программированию (в отличие от специальных методических «черепашек» для младших школьников) можно, как правило, детей, начиная с возраста примерно 12 лет. В этот период или чуть позже начинает формироваться способность к абстрактному мышлению. Примерно так и устроены существующие курсы информатики в школе.

Педагогам хорошо известно, что соответствующие «критические периоды» — оптимальный возраст для освоения разных умений, например, возраст 6-10 лет является критическим для умения читать. Прилагательное «критический» точнее передает смысл, чем «оптимальный», т.к. например, человека старше 14 лет, никогда не читавшего, можно научить читать только по слогам.

Кроме того, нужно учитывать и притягательность компьютеров для детей. Большинство из тех, кто рано и серьезно заинтересовался компьютерами, будут спонтанно втягиваться в программирование по мере формирования у них соответствующих интеллектуальных способностей, т.е. с самого начала вхождения в соответствующий «критический возраст». Именно поэтому дети должны подвергаться правильной «постановке мозгов» в отношении программирования *с самого начала* соответствующего критического периода.

Сказанное выше — вместе с аргументами о необходимости обучать основам программирования всех будущих профессионалов физико-математического, инженерно-технического и естественнонаучного профиля [10], [8] — обосновывает необходимость уделить самое серьезное внимание построению базовых курсов информатики, начиная со старших классов средней школы.

Курсы информатики** в старших классах средней школы — нередко сейчас трактуемые как современные уроки труда — **должны быть серьезно повышены в статусе** и перефокусированы на «постановку мозгов» в отношении фундаментальных основ программирования **как продолжения математики другими средствами.

Вообще по смыслу информатика (имеющая дело, в сущности, с жестко выстроенными текстами, предназначенными для точного описания фрагментов реальности) является мостиком между математикой с одной стороны, и естественными языками с другой, роль которых для интеллектуального развития была понята и положена в основу образовательной реформы еще в 1871 г. [25].

*Весь спектр дисциплин: **математика-информатика-естественные языки** должен, в идеале, рассматриваться целостно.*

О важности нотации для базовых курсов

Следующий ключевой пункт — выбор нотации для курса, т.е. языка программирования. Поскольку относительно языков программирования в мире программирования ведутся настоящие «религиозные войны» (что само по себе показательно), рассмотрим этот пункт подробнее.

Нотация как набор выразительных конструктивных средств — это «нити», из которых «плетется» программный текст. Поэтому она должна уже сама по себе быть наивысшего качества. Нотация должна максимально способствовать «техничному» программированию, помогать развивать хороший стиль и т.п. — определенные вещи запрещать (например, оператор безусловного перехода), а другие гарантировать (сокращенное вычисление логических выражений для грамотной организации циклов).
При неудачной нотации слишком много внимания отвлекается на «подводные камни» нотации, т.е. ее иррегулярности и дефекты дизайна, так что способы обходить их выходят на первый план и затемняют содержательный аспект программирования.

Хорошая нотация в обучении настолько важна, что такой выдающийся авторитет в мире информатики и преподавания, как Никлаус Вирт, специально строил свои собственные языки, чтобы обеспечить эффективность и основательность курса программирования. Такие языки оказали сильнейшее влияние на мир ИТ благодаря своему качеству. В частности, в 1970 г. он опубликовал свой первый язык, спроектированный именно с учетом весьма жестких требований, возникающих в условиях начальных курсов программирования, широко известный язык программирования Паскаль, принесший его автору премию им. Алана Тьюринга в 1984 г., и широко распространившийся в университетах всего мира. (Любопытно отметить, что по рассказам Вирта, его стремление построить основательную систему образования с использованием регулярно спроектированного языка наталкивалась ровно на такие же возражения коллег, какие можно услышать и сейчас: мол, обучать надо на «промышленных» системах программирования. Если бы Вирт не устоял тогда, где была бы знаменитая на весь мир цюрихская школа информатики ...)

Распространение персональных компьютеров с начала 1980-х гг. поставило в повестку дня реализацию концепции всеобщей «компьютерной грамотности». Благодаря подвижнической деятельности российского пионера научной информатики А.П.Ершова и его коллег, к тому времени был накоплен педагогический опыт, указывавший на целесообразность начинать обучение основам «настоящего» программирования школьников с возраста примерно 12-13 лет. В школах стали возникать курсы информатики, охватывавшие старшие классы (8-11). Из-за естественного в начале пути отсутствия понимания возникающих здесь проблем, а также из-за отсутствия соответствующего программного обеспечения, наиболее естественным было использовать реализацию языка Бейсик, прикладывавшуюся к стандартной операционной системе персональных компьютеров.

Однако дефективность Бейсика с точки зрения вводных курсов была хорошо известна уже и тогда. В частности, широко известно совершенно справедливое высказывание упоминавшегося пионера информатики Дейкстры:

> ***Использование Бейсика в начальном курсе непоправимо травмирует ум программиста.***

Все серьезные специалисты согласны как с этим утверждением, так и с центральной ролью первого языка программирования с точки зрения «постановки мозгов» будущих программистов. Попыткой решить эту проблему в СССР был школьный алгоритмический язык, известный в школьной среде как «ершол» по фамилии его автора А.П.Ершова, и система его поддержки «Кумир». Но постепенно использование «Кумира» сошло на нет по причине его «игрушечности», и он был вытеснен Турбо Паскалем.
Вот продолжение цитировавшейся выше дискуссии на международной конференции [17]:

> ***Бауэр:*** *Полностью согласен с Дейкстрой; некоторые современные языки наносят серьезный вред в процессе обучения; нельзя не упомянуть здесь PL/1.*
>
> ***Ранделл:*** *И еще Алгол 68.*
>
> ***Бемер:*** *Я согласен с Дейкстрой о значении элегантности и простоты в мышлении и в программировании. И я согласен, что трудно отучать людей от старых привычек, поэтому у меня такой вопрос: какой процент всех программистов нужно переучивать, по-вашему?*

> ***Дейкстра:*** *Сто десять процентов!*

Поразительно, как мало изменилась ситуация за тридцать с лишним лет — вот цитата из доклада Н.Вирта на Международной конференции по ИТ-образованию в Аархусе, Дания, 2002 [9]:

> *«Достаточно заменить PL/1 на C++ или Java, а JCL — на Windows или Linux, и вы чудесным образом перенесетесь в настоящее.»*

Подчеркнем еще раз:

> ***Выбор языка программирования — самое ответственное первое решение*** *в построении базовых курсов. При недостаточной компетентности массы преподавателей его* ***нельзя пускать на самотек.***

Неудивительно, что когда с конца 1980-х стала доступной удобная версия Паскаля для персональных компьютеров — Турбо Паскаль, наиболее продвинутые преподаватели программирования в СССР и затем в России стали работать в направлении полного перевода всех школьных и вводных университетских курсов программирования на Паскаль.

Заметим, что здесь сыграла роль как традиционно высокая российская математическая культура, благодаря которой был по достоинству оценен Паскаль и его потенциал для образования, так и существующий образец весьма эффективной, уникальной системы математического образования, в которой ключевую роль, роль стержня, играют единые для всей системы вводные курсы: элементарной математики для школ и основ высшей математики для вузов.

Однако, несмотря на более чем десятилетние усилия высококвалифицированных специалистов, «демократический» способ решения проблемы не дал возможности довести благое дело до конца: в системе образования остался известный разнобой в этом отношении, а руководство российского образования в условиях начавшихся реформ не смогло ни понять важности наметившейся тенденции, ни помочь ей реализоваться.

> *Из-за начавшихся реформ не удалось довести до логического завершения стихийное движение, возникшее среди квалифицированных педагогов, в направлении создания* ***единой системы вводных курсов на основе Паскаля.***
>
> ***Важность такой системы*** *для эффективного обучения программированию* ***очевидна*** *из примера уникальной и эффективной российской системы математического образования.*

Одним из результатов сохранившегося разнобоя стала незащищенность еще не завершенной системы от разрушающих влияний.

После изобретения Паскаля (1970) понимание технологий программирования углубилось (модульные и объектные методы). Сам Н.Вирт учел это последовательно в двух языках, прямых потомках Паскаля, — это Модула-2 (1980) и Оберон (1988). Модула-2, появившаяся одновременно с другим паскалеобразным языком Ада, разработанным по заказу министерства обороны США, но более простая и эффективная, естественным образом пришла на смену Паскалю в российской космической [3] и оборонной индустрии (см. прилагаемый к докладу материал С.З.Свердлова).

Оберон — наиболее совершенный наследник старого Паскаля, спроектированный с тщательнейшим учетом требований как преподавания, так и эффективной профессиональной работы — будет предметом нашего разговора в дальнейшем как естественная основа единой системы вводных курсов в современных условиях.

Однако научно-педагогические разработки Вирта оказались в тени маркетинга фирмы Борланд, производителя Турбо Паскаля. Поэтому часть преподавателей, особенно университетских, ушла с Турбо Паскаля на более «современный», «настоящий», «промышленный» продукт фирмы Борланд — Дельфи (Delphi). Система Дельфи, однако, основана на версии Паскаля, в которую было бездумно добавлено множество модных «наворотов», резко усложнивших язык.

Для сравнения: язык Оберон, в который Вирт также ввел, по сравнению с Паскалем и Модулой-2, ключевые элементы объектно-ориентированного программирования, ***оказался проще***, а не сложнее старого Паскаля. Но Оберон ***качественно мощнее*** «дельфийского» Паскаля, несмотря на всю сложность последнего (благодаря механизму автоматического управления памятью, до Оберона остававшемуся прерогативой интерпретируемых языков).

На будущее заметим, что Оберон существует в нескольких мало отличающихся вариантах, одному из которых ученики Вирта дали название Компонентный Паскаль, чтобы подчеркнуть близость и прямую преемственность со старым Паскалем. В дальнейшем мы будем употреблять название **Оберон/Компонентный Паскаль**.

Подчеркнем, что в самой индустрии ПО наибольшую популярность стали приобретать проекты Java и C#, сделанные, хотя и на другой синтаксической основе, но под сильнейшим влиянием именно Оберона! Можно сказать, что индустрия ПО под давлением объективной реальности сумела лучше разобраться в глубинной природе процесса создания ПО, чем преподаватели, соблазнившиеся поверхностной синтаксической преемственностью «дельфийского» Паскаля с первоначальным вариантом языка.

> *Тот факт, что подавляющее большинство преподавателей — в контрасте с развитием индустрии ПО — «прозевали» Оберон (в том числе его версии, с 1993 г. пригодные для преподавания на всех уровнях), соблазнившись обманчивой видимостью «индустриальной мощи» избыточно сложного «дельфийского» Паскаля, только подтверждает тезис о том, что решение важнейшего вопроса о правильном выборе языка вводных курсов программирования не может быть пущено на самотек.*

Однако уход в «дельфийский» Паскаль вместо «правильного» Оберона/Компонентного Паскаля — далеко не самое страшное. В хаосе последних 10-15 лет появились еще две, гораздо более тревожных и опасных тенденции — настолько вредоносных и опасных, что мы не поколеблемся обозначить их как «холера» и «чума». «Холера» — это распространение Си-образных языков. «Чума» — это циничный маркетинг корпорацией Майкрософт своего варианта языка Бейсик в системе среднего образования.

Прежде чем разбирать «чуму» и «холеру», коснемся более общего вопроса, а именно, почему не могут быть хорошим аргументом в пользу той или иной системы программирования для общих курсов соображения типа, что та или иная система является «промышленной», «широко используемой в промышленности» и т.п. Ограничимся цитатой мнения Джоеля Сполски, профессионального программиста и руководителя успешной компании по производству ПО, известного также своими проницательными аналитическими статьями о проблемах индустрии ПО. Вот что он написал о задачах факультетов «компьютерных наук» в университетах [34]:

> *«.. Это не профессиональные училища! Натаскивать людей для работы в промышленности — не их задача. Это задача для местных колледжей и для правительственных программ профпереподготовки беженцев ...*
> *Университеты должны давать студентам фундаментальные средства для того,* ***чтобы прожить жизнь****, а не готовить их к первой паре недель на рабочем месте. ..»*

Подчеркнем, что это мнение руководителя успешной компании-производителя ПО. Сказанное, очевидно, относится и к базовым курсам обшего (фундаментального) среднего образования.

«**Холера Си/C++**». В середине 1970-х была создана и в дальнейшим широко распространилась в разных вариациях (в т.ч. знаменитый линукс) весьма мощная операционная система Юникс. «На спине» ее в мир программирования «въехал» и получил распространение язык Си, использованный для ее написания. В дальнейшем на его основе был построен (как и «дельфийский» Паскаль, бездумным прямым добавлением множества новомодных, зачастую непроверенных средств) язык C++.

Язык Си был сочинен (именно сочинен, т.к. проектированием это назвать нельзя) из бесхитростно проинтерпретированных практических соображений как замена ассемблеру в написании операционной системы Юникс (которая сама по себе явилась, безусловно, прорывной разработкой). При создании Си была проигнорирована **вся** накопленная к тому времени мудрость создания высоконадежных языков программирования. В результате на Си легко написать совершенно нечитаемые конструкции. Приведем цитату с форума [4]:

> *Известны 10 преимуществ Паскаля перед Си:)*
> *Я приведу только одно, но самое важное:*

10. На Си Вы можете написать:

*for(;P("\n").R-;P("|"))for(e=3DC;e-;P("_"+(*u++/8)%2))P("| "+(*u/4)%2);*

На Паскале Вы НЕ МОЖЕТЕ <такого> написать.
*Кстати, может кто-нибудь перевести **эту абракадабру** на Паскаль?*

При таком аморфном синтаксисе программист совершенно не защищен от случайных ошибок (причем в Си рассыпано и множество других «граблей» — см. разбор в [23]).

Было обнаружено, что плотность ошибок в больших программных текстах на языке Си при прочих равных (квалификация разработчиков, объем и сложность ПО, время разработки и т.п.) **в 16 раз** превышает плотность ошибок в программах на наиболее совершенном потомке Паскаля Обероне [24]. Грамотным специалистам очевидно, что с точки зрения обучения программированию Си еще опаснее, чем Бейсик:

> «***Обучение программированию с помощью Си эквивалентно развращению малолетних.***»
> — А.А.Берс, ведущий научный сотрудник Института систем информатики им. Ершова СО РАН (высказывание сделано — и не встретило возражений — на круглом столе ведущих преподавателей информатики новосибирских университетов и школ во время визита Н.Вирта в ИСИ СО РАН 3 октября 2005 г.).

Тем не менее именно те свойства языка Си, которые делают его использование источником «дыр» в больших программах, привели к его популярности среди незрелых «партизан» от программирования, рассматривающих его как игру, соревнование, арену демонстрации собственного «интеллекта». Важно понимать, что подобные явления не являются специфичными для программирования: в филологии и детской психологии хорошо известно аналогичное явление «детского фольклора» (страшилки и т.п.), демонстрирующее устойчивость на протяжении многих десятилетий [20]. Таким образом, можно говорить о стихийном распространении своеобразной «мифологии» вокруг языка Си и его производных, коренящейся в естественном недостатке знаний и опыта, а также в особенностях психики юных «программеров» — желания самоутвердиться среди сверстников («настоящие программеры пишут на Си») и т.д.

Возникновение монструозно сложного языка C++, сочиненного путем «тупого» добавления к Си без исправления его многочисленных дефектов модных конструкций объектно-ориентированного программирования (о недостатках C++ написаны целые трактаты [22]), только усугубило ситуацию: ведь молодежь по наивности путает сложность и мощь языка, об ошибочности чего постоянно говорит Н.Вирт.

> *Стихийное распространение в среде юных «партизан» от программирования порочной мифологии, возникшей вокруг языков C/C++, имеет резоны в примитивных архетипах подростковой психологии.*

К этому стихийному явлению нельзя относиться пренебрежительно по следующим причинам.

Во-первых, на примере программистов, учившихся на фортране еще в 1960-х гг., мы видим, как долго и упорно сохраняются в зрелом возрасте эмоциональные предпочтения и привычки программистского мышления, зафиксировавшиеся в юности (это явление отражено в старом анекдоте: «Пожилому программисту требуется домработница, говорящая на фортране»). Обучать современным эффективным методам программистов с уже сложившимися привычками почти невозможно, и такие программисты постоянно «тащат» в серьезные проекты эти негодные инструменты.

Во-вторых, масса фанатов Си/C++ создает «стадный эффект»: с одной стороны, на него ориентируются коммерческие издательства (которые, как показывает опыт, функционируют как мощный «усилитель» любого мракобесия, если только находится достаточно покупателей; напомним, что изначально потребность знать Си была создана распространением операционной системы юникс, а теперь и линукс). С другой стороны, новички-самоучки (коих множество) не имеют другого ориентира, кроме правила «иди за толпой». Наконец, менеджеры под давлением сиюминутных обстоятельств вынуждены ориентироваться на «предложение» рынка рабочей силы. Возникает порочный замкнутый

круг, который разрывается только в относительно специфических классах приложений (встроенные системы управления и т.п.), где в игру вступает неумолимая объективная реальность, разрушающая мифологию.

В-третьих (и это самое главное), продолжающий расти массив программ, написанных на C/C++, есть постоянный источник проблем — от потерь времени пользователей (современные версии MS Word продолжают регулярно «падать», как и 15 лет назад, хотя и научились лучше сохранять работу), до «дыр» безопасности, связанных с дефектами программ (переполнение буфера, утечки памяти и др.). Дефекты, порожденные использованием таких опасных языков программирования, как Си/C++, очень трудно устранять. Процитируем сайт проекта Информатика-21 [1]:

> *«... В августе 2001 г. вице-президент Майкрософт Джим Олчин (Jim Allchin) объявил во время доклада на открытии конференции Intel Developers Forum в Сан Хосе, что в новой операционной системе Windows XP все возможные проблемы из разряда переполнение буфера были устранены посредством специального анализа исходных текстов на предмет безопасности (security audit). Но в декабре того же года была найдена «дыра» в одной из программ в составе Windows XP (в программах поддержки стандарта подключения внешних устройств Universal Plug and Play), причем, дыра оказалась именно из категории «переполнение буфера». ...»*

Вообще вся ситуация здесь выглядит совершеннейшим абсурдом: то, чего можно без труда добиться просто выбором качественного и простого языка программирования, решается ... невообразимо сложным и дорогостоящим способом, а главной причиной, в конечном счете, оказывается умственная инерция программистов, в начале карьеры подвергшихся «заражению холерой Си». Отсюда понятно, почему серьезные специалисты употребляют такие сильные выражения как «развращение малолетних», «интеллектуальная зараза» и т.п., говоря о языках Си/C++.

Разумеется, проблемы Си/C++ ощущаются и в индустрии (не все же там, в конце концов, «партизаны» от программирования или менеджеры), даже несмотря на то, что конкуренция здесь ослаблена из-за быстрого расширения сферы ИТ. Именно поэтому возник язык Java, представляющий из себя попытку сделать то, что не сумели сделать авторы C++, а именно, не просто бездумно добавить на предательски ненадежную основу Си все возможные новомодные средства, но и сделать какой-то осмысленный отбор, а также перепроектировать «основание» языка, исключив коварные ловушки, которыми знаменит Си. При этом сильнейшее влияние оказал виртовский Оберон (компилятор которого, по сообщению Вирта, команда создателей Java изучила в исходниках, дав ему высокую оценку, за несколько лет до выпуска Java) — строгая типизация, автоматическое управление памятью, отказ от множественного наследования — фактически, это отказ от главных культовых свойств Си, делающих его столь опасным («полная свобода программисту, включая свободу делать любые ошибки»). Разумеется, чтобы убедить фанатов Си согласиться с таким отказом, нужен был сильный способ отвлечь их внимание, и средство для этого было найдено в анти-майкрософтовских сантиментах, широко распространенных в мире ИТ. Дорогостоящая маркетинговая кампания, в значительной степени построенная на эксплуатации этой «обманки», имела успех.

(Интересно, до какой степени во всем этом подтверждаются наблюдения основателя социальной психологии Густава Лебона: «*Толпами нельзя руководить посредством правил, .. надо отыскивать то, что может произвести на нее впечатление и увлечь ее,*» и еще: «*Только вникая глубже в психологию масс, можно понять, до какой степени сильна над ними власть внушенных идей.*» [29]).

Успех маркетинга Java привлек достаточно разработчиков, чтобы преимущества Java в плане производительности из-за отказа от культовой «свободы» Си могли себя проявить. Вслед за Java возник и ее «близнец» C#. Однако, ***оба этих языка — это, прежде всего, инструменты конкурентной борьбы соответствующих корпораций*** за удержание разработчиков: именно поэтому эти языки постоянно усложняются — это делает почти невозможным перенос больших массивов программных текстов на другие языки, прочно привязывая разработчиков к соответствующей платформе. Разработчики же, исходя из мифических соображений «эффективности» и т.п. и не понимая долгосрочную цену своих решений, стремятся использовать все средства, предоставляемые языком, «увязая» в ловушке.

Собственно, здесь мы наглядно видим пользу фундаментального образования — образования, дающего обучающемуся не только «прикладные» знания (в нашем контексте — лежащий на поверхности смысл отдельных конструкций популярного языка), а такие знания глубинных, неочевидных механизмов, скрытых под поверхностью явлений, добыть которые самостоятельно может не хватить всей жизни (в нашем контексте — корректные схемы организации эффективных программ, начиная со схемы линейного поиска).

«**Чума Майкрософт**»**.** Теперь обратимся ко второму явлению, разрушительно действующему на систему начального обучения программированию, — к деятельности корпорации Майкрософт. Прежде всего, Майкрософт является ***неоднократно осужденным*** (Евросоюз, американский штат Калифорния, не говоря о недоведенных до конца исках) ***монополистом,*** и сама мысль, что в системе образования может себя вольготно чувствовать субъект со столь неоднозначной репутацией, — крайне сомнительная.
Тем не менее, Майкрософт активно внедряет (выделив на эти цели финансовые ресурсы в миллионах долларов США [15]) в российское среднее образование свою коммер-ческую систему программирования на основе варианта языка Бейсик.

Однако вспомним, что, как уже упоминалось выше, язык Бейсик зарекомендовал себя однозначно отрицательно с точки зрения начального обучения программированию. (Можно говорить о том, что современный Бейсик далеко ушел от старого, но усложнение языка без изменения его сущности не делает его более пригодным для образования, а ровно наоборот.)

О высказываниях Дейкстры на эту тему прекрасно знают все грамотные специалисты по ИТ-образованию, в том числе не может не знать и Майкрософт, продающая Бейсик больше тридцати лет. Получается, что хотя внедрение Бейсика в начальное обучение противоречит самому смыслу образования, Майкрософт и российские преподаватели идут здесь навстречу друг другу. Почему?

Что касается российских преподавателей, то объяснение тут банальное: во-первых, есть материальная причина (нет нужды распространяться на эту тему; об этом сказано достаточно [33]). Во-вторых, работает «совковый» менталитет: среднему россиянину крайне лестно внимание знаменитой *западной* корпорации, ведь это сразу поднимает его самоуважение (а также уважение со стороны соседей и родственников). Эти два конкретных и сильных фактора — финансовый и психологический, конечно же, перевешивают любые абстрактные соображения о вреде Бейсика, ведь вред этот проявится в неопределенном будущем, и расплачиваться будет не преподаватель, а кто-то другой.

Указанные два фактора работают и на уровне чиновников от образования.

В сущности такое явление — что называется, «**сдача по дешевке**» системы основного образования коммерческим интересам (вдобавок еще и экстра-территориальным, в конечном счете) — вполне аналогично по смыслу известным случаям, когда, например, пенсионерка или мать-одиночка занимается продажей «паленой» водки или наркотиков. Существование объективных резонов для такой деятельности не отменяет сугубой вредоносности явления.

Теперь разберем мотивы Майкрософт. Во-первых, глава монополиста Билл Гейтс в юные годы начал свой бизнес с написания компилятора Бейсика, испытывает к этому языку сентиментальную привязанность и, несмотря на нехорошую репутацию Бейсика для начального обучения, навязал этому языку видное место в качестве макроязыка в продуктах фирмы. (Между прочим, крайнее высокомерие вообще типично для «капитанов» американской ИТ-индустрии — Ларри Эллисона (Oracle), Скотта Макнили (Sun) и др. Для этих людей, что называется, *закон не писан*.)

Во-вторых, сверхзадача, на решение которой с 2000 г. брошены все силы Майкрософт, — это продвижение платформы .NET. Руководители корпорации неоднократно заявляли, насколько критически важной они считают эту стратегию для будущего ***выживания их бизнеса***. Теперь они просто выбрали из набора языков, доступных в системе .NET, тот, который легче всего «проталкивать» как подходящий для начального образования (действительно, когда-то в доисторические, по меркам мира ИТ, времена Бейсик был придуман именно для обучения, да и в российских школах еще сохраняются курсы и

учебники, основанные на старых версиях Бейсика).

Таким способом монополист с примитивной прямолинейностью стремится решить простую маркетинговую задачу — максимально увеличить число программистов, с юных лет приученных к его продуктам. Только такая интерпретация фразы «постановка мозгов» имеет смысл с точки зрения коммерческих интересов Майкрософт — ***никаких других интересов эта компания никогда не преследовала***, а если и делала вид, то только под давлением судебных решений. Наивно ожидать чего-то другого от суперуспешной компании в условиях «зрелого капитализма» (можно почитать о механизме благотворительности Билла Гейтса [30]).

(Любопытно заметить, что сам монополист не пострадает от разрушения системы образования: для себя он с помощью высоких зарплат всегда сможет привлечь редкие таланты; получается, что уменьшая будущий резерв хорошо подготовленных программистов, т.е. разрушая систему образования, монополист фактически лишь укрепляет свое положение. Вряд ли это это является осознанной целью; просто они уверены, что лучших программистов они всегда сумеют нанять, а остальное им безразлично.)

> *Используя свой огромный финансовый ресурс для продвижения Бейсика — негодного инструмента для основательных вводных курсов информатики — в российское общее среднее образование, Майкрософт преследует* ***исключительно свои коммерческие цели.***
> *Эти цели* ***противоречат*** *целям качественного образования,*
> *а потому и* ***долгосрочным национальными интересами РФ.***

Заметим, что подобный открытый маркетинг продукции фирмы в системе общего образования ***невозможен ни в одной из развитых стран***, хотя бы потому, что как руководители образования, так и родители там имеют представление ***о реальной стоимости такой рекламы***.

Отсюда следует вывод, что должно быть прекращено распространенное в России явление, когда продукты коммерческих фирм становятся предметом изучения в рамках основной сетки курсов и в учебниках, санкционированных Министерством образования РФ. Ничтожный в сущности «грант» автору учебника или учителю — ни вообще отдельному индивидууму — не может рассматриваться как адекватная цена за подобный маркетинг.

> *Трудно квалифицировать деятельность Майкрософт в российском образовании иначе как маркетинговую* ***агрессию****, эксплуатирующую коммерческое невежество российского народонаселения и/или нечистоплотность отдельных деятелей системы образования.*

> *В стране, претендующей на настоящий суверенитет в постиндустриальном мире, качественная система общего образования — в первую очередь, его содержание — должна быть* ***неприступной крепостью для экстра-территориальных коммерческих интересов****.*

Выводы для системы образования

Ключевая задача *в данный момент —* ***возродить, защитить*** *и, наконец,* ***завершить на современном уровне*** *начатое на рубеже 1980/90-х гг. движение в направлении* ***единой системы вводных курсов*** *информатики/программирования, охватывающей старшие классы средней школы и младшие курсы университетов.*

Польза такой системы очевидна из примера эффективной и уникальной российской **системы** ***математического образования.*** *Она решит или создаст условия для решения целого комплекса задач,*
— исключив многократное дублирование курсового материала;
— сфокусировав внимание на действительно важных, «бессмертных» вещах;
— спрямив дорогу к популярным современным промышленным технологиям.
Наконец, единая система поставит необходимый заслон разрушительному стихийному распространению среди юных программистов как порочной мифологии Си/C++, так и маркетинговой агрессии экстра-территориальных монополистов, противоречащей долгосрочным национальным интересам РФ.

Единая система вводных курсов — как единая система основных курсов математики — послужит ***стержнем, ясным ориентиром и настоящей опорой*** *для всей системы специальных курсов по ИТ-предметам, в том числе для курсов профессиональной подготовки индустриальной и коммерческой направленности.*

В долгосрочной перспективе *такая единая система, выстроенная на действительно научных принципах, выверенных в многолетней практике создания сложного ПО с предельно высокими требованиями к качеству, безусловно приведет к серьезному повышению качества* ***— корректности, надежности, безопасности, эффективности — всего окружающего нас массива ПО.***

Наконец, ***невозможно представить себе все долгосрочные последствия*** *создания такой единой системы курсов, дополняющей и развивающей уникальную российскую систему математического образования.*
Внедрение больших объемов математики в курсы гимназий и реальных училищ в рамках реформы, ***с твердостью и решительностью*** *проведенной Российским Императорским правительством в 1871 г. [25], имело следствием расцвет российской инженерной школы в начале XX в. (А.И.Попов, С.А.Чаплыгин, В.К.Зворыкин, И.И.Сикорский, А.Н.Крылов, ...), и затем математики, теоретической физики, созданию аэрокосмической и атомной индустрии.*

Наш исторический анализ однозначно указывает и на некоторые конкретные соображения, которые необходимо учесть для успешного решения задачи:

— **Основой вводных курсов** должен стать безупречный с научно-педагогической точки зрения, настоящий наследник старого Паскаля — **Оберон/Компонентный Паскаль**. Ведь именно за Обероном в существенном пошла индустрия (Java, C#).
Ценность Оберона/Компонентный Паскаля для преподавания — в успешном выделении и предельно ясной подаче небольшого набора самых главных, универсальных конструктивных средств, причем без потери выразительности языка для самых сложных приложений. (О недостатках иногда предлагаемых альтернатив см. в конце Приложения 1.)

— Создание единой системы вводных курсов естественно проводится в **два этапа**.

Этап 1. Перевод основных курсов программирования на Оберон/Компонентный Паскаль без изменения программ курсов, используемых преподавателями. Как показывает опыт, это не представляет серьезных проблем благодаря близости старого и нового Паскалей. Этот шаг можно и нужно сделать в кратчайшие сроки, т.к. необходимые комплекты ПО разработаны участниками проекта Информатика-21 и доступны совершенно бесплатно и свободно с сайта проекта [1].
Внедрение единой технологической платформы автоматически интенсифицирует процессы накопления и обмена методическим опытом и материалами, а также создаст

открытое поле для эффективной конкуренции учебников и методик обучения. (В настоящее время основная конкуренция происходит отнюдь не на поле методик преподавания.)

Этап 2. Разработка программ единой системы курсов, скоординированных по линии школа/университет и дающих **оптимальный минимум основных базовых знаний по программированию, структурированию информации и алгоритмике.** (Список конкретных административных мер по обоим этапам дан в Приложении 2.)

— Опыт показал, что невозможно полагаться на компетентность основной массы преподавателей в плане способности сопротивляться влиянию маркетинга и поверхностных аргументов и разобраться в глубинных тенденциях развития сферы ИТ. Кроме того, здесь наблюдается квази-коррупционная подверженность «коммерческим» соблазнам.
Вопрос выбора базового языка и инструментария для единой системы основных вводных курсов **должен быть выведен из компетенции отдельных преподавателей.**

— **Коммерческие интересы должны быть полностью выведены за пределы основных курсов.** Такие интересы с неизбежностью имеют сугубо краткосрочный характер.
Нет никаких препятствий, чтобы такие интересы реализовывались в факультативных курсах — для чего возможность остается в рамках осуществляемых реформ системы образования.
Такая организация позволит по-настоящему гибко реагировать на изменения текущей конъюнктуры без влияния на основные курсы, сфокусированные на «вечных», ключевых, опорных знаниях.

— Невозможно полагаться на активность массы преподавателей в том, что касается сроков перехода на «единую систему». ***Задержка чревата безвозвратной потерей уникальной исторической возможности.*** Для быстрейшей реализации этой возможности стоит включить политический ресурс самого высокого уровня.

— Чем быстрее будет реализована такая единая система, ***тем быстрее она даст шанс всем талантливым ребятам, в том числе из глубинки***
(заработает, по выражению В.Г.Парфенова [19], как «пылесос»).

— В дальнейшем естественней всего, чтобы **полномочия менять технологическую платформу для единой системы курсов информатики/программирования были выведены из недр Мининстерства науки и образования на более высокий правительственный уровень** при соответствующей экспертной поддержке. Это повысит качество подобных ключевых решений и уменьшит влияние клановых интересов и коррупционные риски, а также придаст системе необходимую устойчивость.
Вряд ли подобное решение придется снова принимать в обозримом будущем — Паскаль служил с начала 1970-х гг., а Оберон/Компонентный Паскаль является его уточнением. Основополагающие концепции, выделенные здесь, не подвержены быстрым изменениям — как основы евклидовой геометрии, тригонометрии или ньютоновской механики.

Невозможно предвидеть все долгосрочные положительные последствия от создания единой системы вводных курсов информатики на основе единой технологической платформы. Но люди, в 60-70х гг. XIX века внедрявшие большие объемы математики в российские гимназии и реальные училища, тоже не могли ничего знать о будущих А.И.Попове, С.А.Чаплыгине и др., тем более не могли они предвидеть ни аэрокосмос, ни атомный проект. Они просто считали, что ***система образования должна приучать молодежь основательно мыслить***.
Эту же основную цель преследуют, в сущности, и сформулированные выше предложения.

Приложение 1. Языки программирования Паскаль, Модула-2, Оберон/Компонентный Паскаль

Язык программирования (ЯП) — главный инструмент программиста при написании программ. С одной стороны, ЯП оказывает сильное влияние на мышление программиста, с другой — должен обеспечивать эффективное отображение на машинный код.

ЯП играет ключевую роль во всем процессе разработки ПО, в его проектировании нужно учесть множество противоречивых требований, особенно потому, что ЯП играет роль моста через громадную пропасть, разделяющую человеческое мышление и то, как работает компьютер.

«Виртовские» языки программирования Паскаль, Модула-2, Оберон созданы пионером информатики Никлаусом Виртом (Швейцария). За выдающийся вклад, в частности, создание Паскаля и Модулы-2, Вирт был удостоен в 1984 г. весьма престижной премии им. Алана Тюринга, являющейся в мире ИТ аналогом Нобелевской премии. Н.Вирт является не только признанным еще с середины 1960-х гг. специалистом по проектированию языков программирования, но и ведущим специалистом по проектированию компиляторов (N.Wirth. Theory and Techniques of Compiler Construction. Addison-Wesley, 1996).

Столь глубокое понимание проблематики создания ЯП ставит разработки Вирта на совершенно особое место.

Все виртовские языки спроектированы с учетом жестких требований, налагаемых потребностями процесса обучения программистов. Они отличаются

— замечательной ясностью (структурированность синтаксиса);

— защищают программиста от случайных ошибок (строгая статическая типизация);

— эффективны (допускают быструю компиляцию в эффективный машинный код).

Это уникальная комбинация качеств, делающая **виртовские языки предпочтительными для создания сложных и высоконадежных программных комплексов** (наиболее жесткие требования по надежности возникают при создании встроенных систем управления — космические аппараты, беспилотные летательные аппараты, и т.п.).

Специальное сравнительное изучение [24], выполненное на примере реальных промышленных программ, показывает, что **плотность ошибок** в программных текстах, написанных на последнем виртовском языке Обероне, **более чем на порядок меньше,** чем в текстах на популярном языке Си, при прочих равных условиях.

Паскаль (1970) — первый язык в виртовском семействе. К концу 60-х гг. Н.Вирт уже получил признание как один из ведущих специалистов по языкам программирования (язык высокого уровня Euler, достаточно широко применявшийся язык низкого уровня PL/360, эффективный компилятор для подмножества Алгола-60 — Algol-W). В этом качестве он был включен в состав Международного комитета IFIP по разработке универсального языка программирования на основе Алгола-60. Однако из-за категорического несогласия с проектной философией, возобладавшей в комитете, Н.Вирт из него вышел в 1968 г. и в 1970 г. представил свой Паскаль. Блестящий успех Паскаля на фоне провала Алгола-68, произведенного комитетом, доказали проницательность Н.Вирта.

В рамках проекта Паскаль-P (1972-74) был эффективно применен промежуточный псевдо-машинный язык P-code для облегчения переносимости компиляторов Паскаля; сейчас эта идея легко узнается в байт-коде для языка Java корпорации Sun и в еще большей степени — в аналогичном промежуточном языке проекта .NET корпорации Microsoft.

Модула-2 (1979) наиболее четко воплотила концепции модульного программирования, зарекомендовав себя как лучший в свое время язык создания особо надежных программных комплексов (сейчас эта честь перешла Оберону и его производным, см. ниже; впрочем, Модула-2, благодаря лучшей стандартизации, имеет известные преимущества для низкоуровневой работы типа написания драйверов; заметим, что программное обеспечение для бортовых компьютеров российских спутников связи создается с помощью уникальной системы кросс-программирования на основе Модулы-2 [3]. (См. также отдельный материал С.З.Свердлова.)

Построенный Н.Виртом в 1983-85 гг. однопроходный компилятор для Модулы-2, по

быстроте и компактности на порядок превзошедший альтернативные разработки, продемонстрировал:
(1) мощь систематических математизированных методов в создании сложного программного обеспечения;
(2) громадные преимущества языков с четким компактным формальным определением синтаксиса при создании компиляторов и сопутствующего инструментального обеспечения.

Важнейший проект **Оберон** (1988-1992 и по наст. время) синтезировал более четверти века исследований Н.Вирта по методологии и языкам программирования. В Обероне ему с учениками удалось добиться точного синтеза «старых» достижений структурного и модульного программирования (представленных еще в Модуле-2) с «новыми» объектными методами (языки Simula-67, Smalltalk ...). При этом была решена задача выделить (а) рациональное, (б) обозримое, (в) универсальное ядро конструктов программирования, в том числе объектных, которое могло бы составить основу для разработки любого программного обеспечения. Такое компактное ядро представлено в исключительно тщательно спроектированном, мощном, но простом и прозрачном, языке программирования Оберон. Оберон воплотил приверженность Н.Вирта принципу систематической простоты.

С технической точки зрения, важнейшим достижением Оберона по сравнению с Модулой-2 и Паскалем является «замыкание» системы типов, ставшей теперь полностью герметичной, что сделало возможным интегрировать в язык механизм автоматического управления памятью. Это свойство является уникальным среди всех других эффективно компилируемых императивных языков.

С методической точки зрения, Оберон указал базовый минимум языковых средств, который обеспечивает оптимальную производительность и качество собственно программирования без потери языковой выразительности для больших программных комплексов.

В Обероне в полном блеске продемонстрирована концепция компонентно-ориентированного программирования (КОП). КОП интенсивно обсуждается в мировой индустрии с 90-х гг. и представляет собой очередной шаг в развитии технологий программирования после широкого принятия на вооружение объектных методов в 80-х гг.

Воздействие идей школы Н.Вирта на развитие технологий программирования прослеживается в стратегических мега-проектах лидеров информационной индустрии: Java (Sun) и Microsoft.NET.
Можно говорить о формировании под влиянием Оберона стандартной парадигмы процедурного программирования (впрочем, Оберон выходит за рамки классического процедурного программирования Тьюринга-фон Неймана, заимствуя самые ценные характеристики фукнциональных ЯП).

В орбите этой парадигмы оказываются также языки Java и C#, лежащие в основе проектов Java и .NET. При этом сам Оберон остается непревзойденным по чистоте дизайна и, как прямое следствие, по эффективности как инструмент создания сложных программных комплексов.

Есть несколько близких вариантов Оберона: «классический» ETH Оберон, Оберон-0, Оберон-2, Компонентный Паскаль. Все они чрезвычайно близки друг другу и наследуют все лучшие характеристики Паскаля и Модулы-2: ясность, эффективность, надежность.

Компонентный Паскаль — вариант Оберона, специально уточненный учениками Н.Вирта с учетом промышленного опыта для облегчения проектирования больших систем и с целью наилучшего «мирного сосуществования» с популярными промышленными языками Java и C#. Особое название дано, чтобы подчеркнуть как прямую преемственность со всемирно известным Паскалем, так и ориентированность на поддержку актуальной технологии компонентно-ориентированного программирования; система программирования получила названия **Блэкбокс** — BlackBox Component Builder.

Оберон как операционная система

Следует иметь в виду, что Оберон был спроектирован не только как язык, а как система программирования, в наиболее чистом виде демонстрирующая эффективный подход к разработке программных комплексов. В этом отличие и огромное достоинство Оберона по

сравнению с распространенными средами, несущими в себе наследие средств пакетной разработки еще с 1970-х гг. В Обероне исключен шаг статической линковки программы, и это вместе с модульностью дает возможность строить весьма компактные системы.

Система Оберон является модульной и, благодаря динамической загрузке модулей, допускает эффективную расширяемость непосредственно в процессе работы. Подобный подход к построению системы делает ее чрезвычайно гибкой и привлекательной как полноценная операционная среда.

Действительно, уже первоначальный Оберон был самостоятельной операционной системой. В настоящее время известно несколько автономных операционных систем, основанных на Обероне: Native (ETH) Oberon, XO/2, BlueBottle, а также ряд встраиваемых систем управления, отличающихся высокой надежностью при замечательном минимализме (компании XO/2, SoftSolutions, Coraid, ...).

Хорошо известная система LabVIEW для автоматизации измерительных комплексов компании National Instruments также спроектирована по модели Оберона.

Блэкбокс (BlackBox) — весьма удачный и наиболее популярный вариант Оберона, реализующий язык программирования Компонентный Паскаль,

Авторы Блэкбокса — небольшая группа учеников Н.Вирта, объединившаяся в компанию Oberon microsystems Inc. (Цюрих, Швейцария). Один из них — Клеменс Шиперский (Clemens Szyperski) — является ведущим авторитетом актуальной методологии компонентно-ориентированного программирования (книга «Component Software. Beyond Object-Oriented Programming», Addison, 1998), в настоящее время сотрудник стратегического исследовательского подразделения Майкрософт.

Блэкбокс доступен бесплатно и с полными исходными текстами, защищен одной из стандартных лицензий «открытого кода», аналогично системе Линукс (по настоянию пользователей Блэкбокс использует одну из стандартных лицензий SleepyCat, в которых смягчен экстремистский характер лицензии GPL).

Блэкбокс представляет собой полноценную операционную среду, «погруженную» в другие операционные системы и допускающую взаимодействие с ними. В настоящее время полная версия Блэкбокса работает под ОС MS Windows, а полноценная серверная версия работает под ОС Linux (платформа i386).

Блэкбокс изолирует программиста и пользователя от операционной системы, предоставляя уникальные возможности разработки приложений с интерактивной графикой — как стандартного типа, так и, что самое интересное, нестандартные. По отзывам профессиональных программистов — более мощную и современную, чем графическая система .NET.

Благодаря сочетанию эффективности, автоматического управления памятью и легкости создания специальной интерактивной графики Блэкбокс оказался весьма привлекательным для сложных научных расчетов синтетического характера («умные» численные алгоритмы, символическая алгебра большого объема, системы обработки данных и моделирования). Например, на Блэкбоксе создано ПО для расчета бортовых фазированных решеток истребителя Eurofighter; полный объем ПО — около миллиона строк. На Блэкбоксе сделана система управления крупнейшего каскада ГЭС на Амазонке (корпорация Alstom Power), а также целый ряд других приложений, в т.ч. для экспериментальной и теоретической ядерной физики.

Все Обероны отличаются замечательной переносимостью, но Блэкбокс еще обеспечивает средства хорошей интеграции с операционной системой, в которую он погружен. Блэкбокс может быть поставлен и на «голое железо», а также использоваться как основа системы кросс-разработки (например, компонент Denia).

Подчеркнем, что Блэкбокс, хотя и аналогичен виртуальным машинам Java и .NET, но работает с эффективным компилируемым кодом. Благодаря этому программы, написанные на Компонентном Паскале в Блэкбоксе, качественно более эффективны, чем программы на Java и C#. В то же время сохраняется возможность компилировать Компонентный Паскаль как в Java, так и в MSIL (известный компилятор GPCP — Gardens Point Component Pascal).

В 1998 г. Блэкбокс был удостоен премии журнала BYTE за высокое качество технических решений («technical excellence award»).

Два слова об альтернативах для образования

Как отмечено в основном тексте статьи, преподаватели, искавшие современную и более мощную замену Турбо Паскалю, пошли, так сказать, не за «Паскалем» (т.е. теми принципами, которые попытался реализовать Вирт), а за «Турбо», т.е. за разработчиками корпорации Борланд, которые, ориентируясь на краткосрочные интересы (совместимость), чрезмерно усложнили язык, просто добавляя к нему новомодные средства без надлежащего отбора и интеграции в языковое ядро. Чрезмерная сложность — в полном согласии с «принципом Калашникова» («Избыточная сложность есть уязвимость») — в конце концов вышла Борланду боком: корпорация недавно объявила, что прекращает работу по инструментальным средствам и избавляется от этого бизнеса [26].

Дельфи не годится для обсуждаемой единой системы вводных курсов еще по той причине, что является слишком сложной для общих курсов в средних школах, а дробить единое образовательное пространство — значит, воспроизводить те же проблемы, на решение которых направлены предложения проекта Информатика-21.

По той же причине (непригодность для общих школьных курсов) не годятся Java и C#: во-первых, из-за своего Си-образного синтаксиса; во-вторых, они сделаны их авторами чрезмерно сложными по тактическим маркетинговым соображениям.

Другая группа языков — интерпретируемые языки сценариев perl, python и т.п. — набирает популярность, как указывают проницательные аналитики (см., например, [31]), едва ли не в первую очередь потому, что в них, как и в Обероне, решены проблемы автоматического управления памятью. Однако это тоже довольно сложные языки, неэффективные, и область их применения заведомо уже, чем у Оберона.

Ни один из известных участникам проекта Информатика-21 потенциальных кандидатов на роль платформы для единой системы общих вводных курсов программирования не может рассматриваться как серьезная альтернатива Компонентному Паскалю и Блэкбоксу.

Приложение 2. Конкретные мероприятия по созданию единой системы вводных курсов информатики

Сводка подготовлена по материалам проекта Информатика-21 http://www.inr.ac.ru/~info21/ . Цифры в квадратных скобках ([1] и т.п.) отсылают к списку источников.

Данные ниже предложения обобщают 5-летний опыт общественного научно-образовательного проекта Информатика-21 (см. сайт [1] и доклады [10], [8]) и конкретизируют программу, очерченную в разделе «Выводы для системы образования» в основной части доклада.
Предложения опираются на российские образовательные традиции:

— более чем вековой опыт общей системы математического образования,
— опыт широкого использования Паскаля во вводных курсах информатики.

Предложения направлены прежде всего на средние школы, а также базовые курсы основ программирования на младших курсах университетов.
Поскольку информатика/компьютинг есть продолжение математики другими средствами, то оказывается исключительно полезным постоянно иметь в качестве образца существующую систему математического образования. Это позволяет получать ясные ответы на большинство возможных здесь вопросов общего характера.

Повторим обоснованную в основном тексте главную задачу, на решение которой нацелены данные ниже предложения:

> ***Ключевая задача*** *в данный момент —* ***возродить, защитить*** *и, наконец,* ***завершить*** *на современном уровне начатое на рубеже 1980/90-х гг. движение в направлении* ***единой системы вводных курсов*** *информатики/программирования, охватывающей старшие классы средней школы и младшие курсы университетов.*

По поводу дальнейших соображений в обоснование наших предложений, см. основной текст доклада, доклады участников проекта Информатика-32 [10], [8], а также материалы сайта проекта [1].

Чтобы представлять себе, куда двигаться, нужно хотя бы примерно представить себе идеальную систему вводных курсов.

В идеале составные части единой системы вводных курсов информатики суть следующие:

а) Общий школьный курс основ информатики (охватывающий темы алгоритмики и структурирования информации) для старших (8-11) классов средней школы для всех профилей (кроме самых гуманитарных). Подобный курс можно делать в формате обычного школьного курса информатики. Однако необходимо четко отделить фундаментальную часть (данный курс) от всякого рода конъюнктурных и быстро устаревающих «информационных технологий», к которым следует относиться как, скажем, к курсам вождения автомобиля (т.е. вынести их в систему дополнительного/профессионального обучения).

Курс, который мы имеем в виду, идейно стоит в ряду между математикой и языковыми курсами, и хотя по форме это введение в программирование, имеет фундаментальный характер. Его цель — дать минимум основополагающих «вечных» знаний и концепций, не зависящих в том числе и от языка программирования. Точный объем знаний и умений должен быть уточнен, исходя из опыта и в координации с соответствующим вводным университетским курсом.

Из данного курса необходимо исключить глупости вроде определения курсора как «светового пятна ...» — достаточно пальцем показать курсор на экране. Необходимый минимум вспомогательных сведений о том, как включать компьютер, сведения по файловой системе и т.п. можно вполне усвоить по ходу дела. (Достаточно скоро ученики будут поголовно все это знать еще до прихода на первый урок в 8-м классе.) Весь подобный

псевдо-научный мусор следует вывести за рамки данного курса, который — повторим — мыслится как стоящий в ряду с математикой с одной стороны, и родным и иностранным языками с другой.

На уровне школы должны быть охвачены, грубо говоря, все, кто учит тригонометрию (физико-математический профиль, инженерно-технический, естественно-научный, экономический, лингвистический [лингвисты на филологическом факультете МГУ изучают Турбо Паскаль].). Другими словами, курс должен носить **общий характер**. Возможно, слегка варьируясь по глубине, как и в случае с математикой.

б) Общий университетский курс основ систематического программирования (имеется в виду программирование как «в малом», так и «в большом», т.е. не только алгоритмика, но и аспекты построения больших программ) для студентов 1-2 курса. Здесь возможны вариации по содержанию в зависимости от специализации, как и в случае курсов матанализа.

Общий характер вводных курсов важен еще и потому, что программные системы, созданные прикладными специалистами, обычно ценятся выше и используются чаще, чем программы, созданные профессиональными программистами, потому что точнее отражают содержательный аспект соответствующих задач. Прямой вывод, который отсюда следует, — нужно максимально облегчить для «непрофессионалов» создание качественного ПО, в том числе и за счет правильного начального обучения.

в) Единая инструментальная платформа и язык программирования **Компонентный Паскаль** для использования в общих вводных курсах как на уровне школ (а), так и университетов (б). Такая система должна удовлетворять множеству требований, включая бесплатность, простоту, современность при максимальной преемственности со старым Паскалем. Такая система найдена и адаптирована к использованию в российских школах в рамках проекта Информатика-21. Система называется Блэкбокс (BlackBox Component Builder), а язык программирования Компонентный Паскаль является прямым современным потомком Паскаля (подробнее см. Приложение 1 и материалы проекта Информатика-21 [1]).

С научно-педагогической точки зрения, система Блэкбокс имеет безупречную «родословную»: она разработана в школе легендарного пионера информатики Никлауса Вирта, автора языков Паскаль, Модула-2 и Оберон, а также ряда учебников программирования, переведенных на множество языков мира. Ограничимся единственной цитатой из одного из интернет-форумов:

> ***«Компонентный Паскаль обречен на успех в российских школах»***
> — С.А.Жилин, преподаватель информатики, школа 887, Москва.

Пакет ПО для школ, свободно доступный с сайта [1], содержит модуль эмуляции графики Турбо Паскаля, обеспечивающий безболезненный перевод программных примеров с Турбо Паскаля на Компонентный.

Подчеркнем, что Блэкбокс — безупречно современная система, удобная, простая и эргономичная (легко настраиваются размер и гарнитура экранного шрифта).
В школьном пакете произведена полная русификация системы: на русский язык переведены меню и даже сообщения компилятора, причем учитель может их легко модифицировать под нужды конкретного класса или даже ученика. *Ни одна другая система программирования не позволяет этого сделать.*

За пять лет существования проекта «Информатика-21» не найдено никаких реалистичных альтернатив Компонентному Паскалю и системе Блэкбокс на роль технологической основы системы вводных курсов информатики. Все промышленные системы оказываются слишком сложными, а Турбо Паскаль безнадежно устарел.

Блэкбокс в этом смысле предоставляет идеальный баланс: идейная преемственность со старым Паскалем, весьма мощная система промышленного качества, но одновременно простая и образцово спроектированная, что бесценно для школьного преподавания.

Абсолютно важно, что это должна быть **единая платформа** (см. об этом далее).

На уровне университетов могут быть более значительные вариации по глубине и охвату материала в зависимости от специальности (как и в случае с высшей математикой), но на

школьном уровне вариации в основном курсе должны быть меньше, т.к. здесь речь идет о более универсальном материале.

Нет речи о том, чтобы не изучать или запретить другие языки и системы программирования:

речь только о том, что изучение «промышленных» языков и систем, отягощенных архаическим наследием, воплощающих внетехнологические интересы (например, привязка разработчиков к конкретной платформе через избыточную сложность языка) и служащих для продвижения коммерческих интересов, должны быть вынесены из базовых курсов в специальные и факультативные курсы. Этот тезис основан на конкретном опыте, давшем толчок проекту Информатика-21 [1]: когда для вводных курсов используются применяемые, скажем, физиками фортран или C++, то все усилия уходят на изучение «особенностей» (читай, дефектов) соответствующих языков, и цель прочного усвоения основ предмета остается недостигнутой. С другой стороны, в силу минимализма Оберона/Компонентного Паскаля перейти после соответствующего обучения на любой широко распространенный промышленный язык не представляет проблемы.
Кроме того, при таком подходе расчищается поле для конкуренции промышленных технологий в собственно техническом аспекте.

На прочном фундаменте системы вводных курсов должны строиться курсы, ориентированные на узкоспециальную, профессионально-ориентированную подготовку. Такие курсы тем легче строить, чем увереннее преподаватель может предполагать знание определенного базового минимума понятий у учащихся так же, как преподаватель матанализа не заморачивается на объяснение тригонометрических тождеств.

Итак, **полная система** курсов ИТ-ориентации тогда приобретет примерно следующий вид:

Общий школьный курс основ информатики
+Углубленный школьный курс, посвященный алгоритмизации и основам программирования;
+Введение в «промышленные» системы программирования (среднее специальное образование)

Общий университетсккий курс (основы систематического программирования + основы архитектуры)
+Курс алгоритмов
+Курс программной архитектуры
+Курс численных методов
+Курсы по особенностям «промышленных» систем и языков
...

Методика преподавания общих курсов информатики (педагогические университеты)

Подчеркнем также, что возможности среды Блэкбокс далеко выходят за рамки собственно программирования, например, в ней имеется полноценный, легко настраиваемый и достраиваемый текстовый редактор, сравнимый по возможностям (а по некоторым ключевым характеристикам, в том числе по надежности, превосходящий) MS Word, а также мощная среда создания интерактивных графических приложений. Это означает, что

> *Блэкбокс является прекрасной технологической **основой для единого информационного школьного пространства**, связывающего всю школу.*

Такое пространство естественно разовьется из ПО, которое будут создавать школьники в рамках индивидуальной и групповой работы, конкурсов и т.п. Ведь в отличие от Турбо Паскаля здесь есть возможность без особого труда создавать полноценные, «настоящие» приложения (составление расписания, моделирование экспериментов, тестирование, вспомогательные программы для учета оценок, и т.п.).

Подчеркнем, что альтернативные «промышленные» системы исключительно тяжеловесны и сложны для освоения учителями. С Блэкбоксом же можно ставить задачу, чтобы в перспективе *все* учителя могли хотя бы использовать его как текстовый редактор, а для учителей-математиков, физиков, химиков — использовать его возможности в своих курсах.

Какие проблемы решит единая система вводных курсов?

В содержательном плане единая система вводных курсов позволит решить следующие остроактуальные задачи:

1) Поднять уровень преподавания вводных курсов информатики **на уровень современных требований,** пролагая дорожку к популярным системам Java, C# и проч.

2) Дать учащимся ***знания-ключи*** («фундаментальные знания») для ориентации в море прикладных знаний, и чтобы можно было уверенно отсеивать «мусорную информацию», коей в мире ИТ очень много в силу рукотворного характера ИТ-артефактов. Кроме того, например, программистам приходится постоянно переучиваться — осваивать новые языки и т.п., и опыт показывает, что эта задача сильно облегчается, если специалист обучен правильному универсальному минимальному набору «инвариантов», которые достаточно научиться выражать в новом языке. Оберон/Компонентный Паскаль дает именно такой набор «инвариантов».

3) **Дать подросткам авторитетные ориентиры** в плане «что такое хорошо, и что такое плохо» в ИТ-технологиях. Обеспечить интеллектуальную профилактику против распространения в среде подростков «мусорных технологий» и порочной мифологии (явление, превратившееся в серьезную проблему: переучивать студентов чрезвычайно сложно).

4) **Максимально облегчить ИТ-подготовку «непрофильных» специалистов** (физиков, инженеров, строителей, экономистов, лингвистов, ...): как минимум, чтобы они имели минимальный общий язык для продуктивного общения с программистами-аналитиками при разработке требований для программных систем; как максимум, чтобы они могли при решении своих профессиональных задач использовать компьютеры максимально эффективно и так же свободно, как математические методы.

В методическом плане:

5) **Создать максимальную конкуренцию по качеству материалов и по методическим инновациям** среди авторов учебников и пособий. Это возможно только, если все учебники и пособия будут ориентированы на единый язык и систему программирования.

6) **Ликвидировать разнобой** в используемых системах программирования и, тем самым, обеспечить возможность эффективного тестирования (в том числе в системе ЕГЭ) по предмету выпускников средних школ, а также обеспечить возможность приемных экзаменов в университеты по единой программе.

7) Университетские преподаватели хотели бы четко знать, какой **минимум знаний ожидать от выпускников школ,** так же, как преподаватели высшей математики ожидают знания, например, тригонометрии и умения решать квадратные уравнения.

8) **Исключить дублирование усилий,** т.е. ситуацию, когда базовый вводный материал по программированию (понятия переменной, цикла, процедуры, параметра и т.п.) преподаватели вынуждены дублировать в курсах по узкоспециальным темам. Такое дублирование сильно снижает эффективность всей системы преподавания ИТ-дисциплин.

Нужно особо подчеркнуть, что при подготовке специалистов высшей квалификации **ни по одной** серьезной специальности во вводных курсах не ставится цель узкопрофессиональной подготовки или изучения промышленных инструментов.
Для ИТ-специальностей здесь невозможно договориться хотя бы потому, что спектр промышленного инструментария обширен и постоянно меняется.

А делать государственную систему основного общего образования заложницей краткосрочных интересов коммерческих компаний — преступно.

Поэтому **главной целью** вводных курсов должна быть **«постановка мозгов»**, постановка базовой техники, «импринтинг» в юных умах **правильных, научно обоснованных** первых принципов программирования как доказательной деятельности, как серьезной и ответственной инженерной дисциплины.

Апробация

1) Необходимо подчеркнуть, что весь проект изначально возник не как выдумка педагогов-методистов, а на каждом шаге **вырос из чисто конкретных нужд** (острая нехватка специалистов-физиков и т.п. с правильно поставленной базовой техникой программирования).

2) Свидетельством того, что проект сильнейшим образом попадает в резонанс с текущими проблемами ИТ-образования, особенно на уровне средних школ — энтузиазм участников, проявившийся в коллективном осуществлении на общественных началах полного перевода документации системы Блэкбокс.

3) Турне Н.Вирта по городам России (С.-Петербург, Москва, Нижний Новгород, Екатеринбург, Новосибирск и Томск) осенью 2005 г.: публичные лекции прошли с аншлагом, а круглые столы собрали в общей сложности порядка 350 ведущих преподавателей всех уровней. Основные положения проекта подробно обсуждались на этих круглых столах, и никаких возражений по существу не вызвали, а демонстрации подготовленной проектом Информатика-21 системы программирования вызывали живейший интерес преподавателей. Было распространено несколько сотен компакт-дисков с ПО, предлагаемым проектом Информатика-21.

4) На последней так называемой Ершовской конференции «Проблемы систем информатики-2006» в новосибирском Академгородке представление проекта в секции, посвященной ИТ-образованию, вызвало самый активный интерес, ряд преподавателей самых разных уровней вызвались участвовать в проекте (см. ниже раздел Разработка единой системы курсов).

5) В деятельности по программной поддержке системы на добровольных началах участвуют профессиональные программисты целого ряда городов России, вот неполный список: Брянск, Троицк (Моск. обл.), Орел, Стрежевой (Томская обл.), Казань, Томск, Магадан.

6) Большой интерес к проекту и системе Блэкбокс проявляют специалисты не только России, но и других стран на территории бывшего СССР:

— специалисты литовского Института математики и информатики (Вильнюс) отозвались, что Компонентный Паскаль — «это действительно практически идеальный язык для обучения программированию» (д-р Т.Евсикова, кафедра методики преподавания информатики);

— А.Б.Кондратович, руководитель ИТ-образования Витебской области (Беларусь), осуществляет **4-летнюю программу перевода школ области на Блэкбокс и Компонентный Паскаль**;

— родственные проекты были объявлены в университете г. Ош, Киргизия.

Подготовка внедрения единой системы вводных курсов

Самая трудная начальная часть подготовки уже выполнена усилиями большой группы энтузиастов:

— Сделан полный перевод документации системы Блэкбокс на русский язык.

— Выполнена русификация системы (включая меню и сообщения компилятора; подчеркнем, что полноценная система программирования, допускающая полную русификацию сообщений компилятора для целей школьного преподавания, — вещь **уникальная** и не имеющая аналогов в коммерческих системах).

— Подготовлены модули, облегчающие использование системы в школьных олимпиадах программирования.

— Разработан модуль эмуляции графики Турбо Паскаля для облегчения перевода на Компонентный Паскаль уже имеющихся курсов, основанных на Турбо Паскале.

— А.И.Попковым подготовлен сборник задач по программированию на Компонентном Паскале (более 300 задач), охватывающий потребности серьезного школьного курса программирования и демонстрирующий разнообразные возможности среды Блэкбокс.

— Имеется интернет-форум для преподавателей и активный сайт поддержки системы Блэкбокс [2].

Все перечисленные материалы свободно доступны с указанных сайтов.

Схема перехода на Компонентный Паскаль

Переход в преподавании на Компонентный Паскаль может быть выполнен в два этапа:

Первый этап. Поскольку старый и новый (Компонентный) Паскаль отличаются минимально как по синтаксису, так и по семантике, то первый этап может состоять в переводе курсов на Компонентный Паскаль **без изменения структуры курсов.** Опыт участников проекта Информатика-21 показывает, что для мало-мальски компетентных преподавателей, использующих в своих курсах старый Паскаль, такой переход не представляет никакой сложности. На этот этап можно отвести один учебный год.

Фактически большинство мероприятий первого этапа имеют информационный и рекомендательный характер:

— Необходимо как можно быстрее обеспечить возможность использования Компонентного Паскаля всеми желающими школьниками в олимпиадах по программированию для школьников на всех уровнях — от районного до всероссийского. Для этого необходимо издать приказ Минобрнауки с соответствующей рекомендацией; подчеркнем, что это чисто «разрешающее» действие, никаких рисков этот шаг не несет, поэтому задержек здесь быть не должно.

— Необходимо подключить профильные институты Минобрнауки (два таких есть — в Москве и Новосибирске) к работе над совершенствованием и распространением школьной версии системы Блэкбокс. Рассылки регулярно дополняемых версий системы вместе с новыми методическими материалами могли бы распространяться по школам России на компакт-дисках и выкладываться на сайтах областных и краевых институтов повышения квалификации работников образования для свободного скачивания и обновления.

— Приказом Минобрнауки рекомендовать Компонентный Паскаль как предпочтительный язык для школьных курсов информатики, с перспективой его принятия в качестве обязательного. В любом случае необходимо обозначить Компонентный Паскаль и систему Блэкбокс как полностью «легальные» во всех смыслах инструменты для школьного преподавания.

— Найти форму поддержки работы по редактированию русскоязычной документации системы Блэкбокс **в целях ее скорейшего завершения**. <…>

— Организовать обсуждение хода реализации проекта и обмена опытом в специализированных интернет-форумах. Для начала можно использовать уже существующий интернет-форум проекта Информатика-21, [2], и тогда достаточно просто проинформировать школьных учителей и методистов соответствующим циркуляром Минобрнауки.

— Информационным письмом Минобрнауки рекомендовать преподавателям базовых университетских курсов программирования, алгоритмики и т.п. перейти со всех версий устаревшего Паскаля (Турбо, Дельфи) на Компонентный Паскаль (вариант бесплатной, доступной с открытыми исходниками системы Блэкбокс, ориентированный на использование в университетах и используемый в настоящее время на физическом факультете МГУ, также разработан в рамках проекта Информатика-21 и свободно доступен с сайта проекта).

В отношении особенно Дельфи заметим, что Борланд, компания-производитель Дельфи, объявила о продаже подразделения разработки средств программирования и перепрофилировании своей деятельности [26]. Так что ориентироваться на эту систему, учитывая чрезмерно сложный язык, не имеющий перспектив, бессмысленно.

— Необходимо как можно быстрее **отменить** рекомендации Минобрнауки в отношении языков семейства Си/C++ в базовых университетских курсах программирования. Такого рода рекомендации суть проявления некомпетентности в лучшем случае. **Нет никаких препятствий** к тому, чтобы изучать особенности синтаксиса **Си в спецкурсах** после прочного освоения базового материала.

— Рекомендовать университетам, проводящим вступительные экзамены по программированию, переходить на Компонентный Паскаль. Наиболее эффективным было бы **просто приказать** все вступительные экзамены по программированию с лета 2008 г. проводить на Компонентном Паскале.

— Проинформировать издательства учебной ИТ-литературы о внедрении Компонентного Паскаля в преподавание с целью скорейшей подготовки учебников (для чего достаточно на первом этапе перевести на новый Паскаль уже зарекомендовавшие себя учебники). Очевидно, **издательства** (как и авторы учебников) **будут только рады** продать новые книги взамен старых.

— **Специальная поддержка** тех авторов, которые готовы уже в ближайшее время перевести свои курсы и учебники со старого на Компонентный Паскаль. Первые кандидаты здесь есть <...>

— **Коммерческая (маркетинговая) деятельность в рамках системы основного образования должна быть запрещена явным приказом.** Изучение коммерческих систем должно быть выведено в факультативные курсы, коммерческие курсы специализации и т.п.

— Ввиду подчеркивавшейся в основном тексте важности предмета, **полномочия определять технологическую платформу для единой системы курсов информатики/программирования естественней всего передать на правительственный уровень над Министерством образования и науки** — это уменьшит влияние клановых интересов и коррупционные риски, а также придаст системе необходимую устойчивость.

Сроки выполнения мероприятий первого этапа

При наличии достаточной политической воли, выполнение первого этапа — т.е. полный переход на Компонентный Паскаль во всех базовых курсах информатики/программирования — можно в существенном завершить **к 1 сентября 2008 г.** с помощью приказов Минобрнауки.

Никаких потерь, о которых стоило бы говорить не будет, так как все равно нет настоящей системы, а существующие курсы все равно в дальнейшем нужно будет «перелопачивать» как минимум для обеспечения связки «школа-университет».
Если преподаватель недостаточно квалифицирован, чтобы за оставшееся время полностью освоиться с Компонентным Паскалем и Блэкбоксом (для этого на самом деле достаточно одного-двух месяцев), то курс такого преподавателя вряд ли представляет такую ценность, чтобы его беречь.

> *Состояние школьного образования в области информатики настолько неудовлетворительное, а преимущества создаваемой единой системы настолько очевидны и велики, что* ***целесообразно внедрить Компонентный Паскаль во все вводные курсы*** «***одним махом***»*.*

Исключение можно сделать лишь для СПбГУ ИТМО, где есть работающая связка «школа-университет». Там есть хоть какой-то осмысленный аргумент для того, чтобы немножко растянуть переход («дайте доучить детей по-старому»). Да и там не очень понятно, какая польза будет достигнута от затяжки процесса, тем более с учетом их очень высокой квалификации.

Что касается возражений «коммерциализованных» университетских преподавателей, построивших вводные курсы на коммерческих платформах при «поддержке» коммерческих фирм, то им нужно просто предложить переформатировать такие курсы в специальные и воспользоваться шансом быть первыми в создании соответствующих пособий по изучению коммерческих систем на основе, которую дает единая система вводных курсов. Такие курсы/пособия заведомо позволят глубже рассмотреть особенности соответствующих систем, чем «полные» курсы программирования, и будут доступны более широкой аудитории.

Подчеркнем еще раз: **стихийное движение к единой системе уже имело место** 10-15 лет назад и частично реализовалось (на платформе Турбо Паскаля), и для такого движения были совершенно **объективные основания**, которые никуда не исчезли. Просто в суматохе последних лет предоставленные сами себе преподаватели не сумели разобраться, в каком направлении двигаться (см. основной текст), почему и произошло «размывание» наметившейся системы. Теперь мы знаем, что настоящим преемником

старого Паскаля является Оберон/Компонентный Паскаль и **необходимо просто возобновить процесс и завершить его** в кратчайшие сроки.

Заметим, что если бы единая система на платформе Турбо Паскаля уже существовала, то данный переход можно было бы подготовить и осуществить организованно и практически безболезненно. Разумеется, здесь есть риск ошибочных решений, а также определенные коррупционные риски («сотрудничество» крупных корпораций), именно поэтому прерогатива решать подобные вещи принадлежит уровням госуправления **над** Министерством образования и науки РФ.

Второй этап

После формирования единой основы программы школьных и вводных университетских курсов должны быть модифицированы с учетом новых возможностей Компонентного Паскаля (прежде всего, в отношении автоматического управления памятью и работы с динамическими структурами данных, что облегчает естественное введение элементов важной темы «структурирование информации», ср. [32]).

Кроме того, программы должны быть унифицированы с выделением базового минимума, оптимального для изучения в средней школе. Для этого необходимо в максимальной степени задействовать возможности Интернета (через дискуссии преподавателей на интернет-форумах).

— Программы вводных курсов должны быть скоординированы по линии «школа-университет», с тем, чтобы оптимально распределить учебный материал между средней школой и младшими курсами университетов.

— Должен быть завершен полный переход школьного преподавания информатики на Компонентный Паскаль и систему Блэкбокс, включая разработку и публикацию соответствующих методических материалов и т.д.

— Разработка и публикация программ базовых курсов программирования и информатики для **педагогических университетов — *для всех предметных специальностей*** вместе с соответствующими методическими материалами. Это необходимо **для создания в перспективе максимальной синергии** между различными предметами в рамках единого школьного ИТ-пространства.

— В рамках, например, 7-й конференции «Проблемы систем информатики-2009» (Академгородок, Новосибирск) подвести итоги работы по созданию «Единой системы вводных курсов» и наметить перспективы дальнейшей работы. По срокам конференция является удобной «контрольной точкой», к которой можно было бы приурочить подведение результатов конкурсов и т.п.

— Организовать программу по возможности поголовного **повышения квалификации учителей непрофильных специальностей** с целью хотя бы минимального ознакомления с возможностями системы Блэкбокс (работа с текстами, диалогами, интерактивной графикой и т.п.), имея в виду создание разного рода электронных пособий и проч. на материале соответствующих предметов. Организовать интернет-форумы для обмена опытом.

— Разработка и публикация программ базовых курсов программирования и информатики для различных непрофильных специальностей классических и технических университетов.

— Разработка и публикация курсов, опирающихся на новую систему вводных курсов. Например, курс по особенностям Си или фортрана, построенный по принципу интерпретации в синтаксисе этих языков фундаментальных схем организации кода в Компонентном Паскале.

Разработка единой системы курсов

Поскольку конкретная цель и смысл всех мероприятий — в создании единой, скоординированной системы вводных курсов, нужно, как можно скорее, предпринять шаги в этом направлении, хотя бы для разработки первоначальных методик, позволяющих широкой массе преподавателей включиться в процесс.

Необходимо найти форму организации и поддержки этих усилий ....

<...>
Заметим, что список охватывает практически весь основной диапазон: средние школы, вводные университетские курсы, методика преподавания.

Поддержка проекта с помощью конкурсов

Наличие единого языка программирования и единой технологической платформы дает **беспрецедентную возможность создать «конкуренцию инноваций»** в преподавании по самым разным направлениям. Очевидными являются конкурсы типа «Лучший базовый учебник/задачник информатики/алгоритмики». Поэтому сосредоточимся на новых, менее очевидных возможностях, и предложим для начала следующие конкурсы:

— «Лучшая система базовых учебников информатики». Здесь соревноваться должны скоординированные *пары* (или более широкие комплекты) учебников — для школ и для младших курсов университетов. Суть конкурса — в поиске оптимального разделения и координации программ обучения на двух уровнях. Уже сейчас есть несколько потенциальных участников такого конкурса <...>

— «Лучшее электронное пособие по ...» по разным школьным предметам. Суть конкурса — в поощрении синергии разных школьных предметов в рамках единой ИТ-платформы.

Данные выше предложения — лишь первая попытка увидеть новые возможности, которые здесь открываются.

Дальнейшая работа

По достижении ситуации, когда в школах будет существовать «единое ИТ-пространство», можно ставить задачу более полной интеграции понятий информатики в школьную программу. Напомним, что авторы известного фундаментального документа ЮНЕСКО [27] справедливо подчеркивают, что ряд понятий, которые приходится вводить в курсах информатики/программирования, на самом деле естественно разнести по соответствующим предметам (например, изучение формальной логики — в курс математики).

С другой стороны, языковые курсы могут в конечном счете выиграть в методическом плане, если изучение грамматики сможет опереться на соответствующие понятия формальных грамматик, в простейшем виде возникающих в курсе информатики. Соответствующая взаимная перестройка и оптимизация разных курсов должна составить цель дальнейшей работы.

Именно с точки зрения такого дальнего прицела важно заранее прилагать усилия к тому, чтобы *все* школьные преподаватели могли общаться на одном «языке» в рамках единого школьного ИТ-пространства (здесь имеется в виду, конечно, не только и не столько сам язык программирования Компонентный Паскаль, сколько общее понимание возможностей системы).

Внедрение современной, простой и доступной даже непрограммистам единой системы/ИТ-платформы в средней школе будет, очевидно, иметь глубокие последствия для всей системы школьного преподавания, которые сейчас трудно в деталях предсказать.

Список источников

[1] Интернет-сайт проекта Информатика-21: http://www.inr.ac.ru/~info21/
[2] Интернет-форум проекта Информатика-21: http://bbforum.metasystems.ru/
[3] A.A. Koltashev, "A practical approach to software portability based on strong typing and architectural stratification", сс. 98-101 в [7].
[4] Научно-производственный конструкторский центр «Новик-XXI век», Главный конструктор Н.В.Чистяков. Интернет-сайт: http://dpla.ru/Novik-XXI/
[5] F.V. Tkachov, см. презентацию в [6], а также [8].
[6] Международное рабочее совещание "Oberon Day @ CERN", ЦЕРН, 10 марта 2004 г. Интернет-сайт: http://cern.ch/oberon.day/; http//www.inr.ac.ru/~blackbox/oberon.day/
[7] Lecture Notes in Computer Science 2789. Springer-Verlag, 2003 (материалы Объединенной международной конференции по модульным языкам программирования, Joint Modular Languages Conference, Клагенфурт, Австрия, 25-27 августа 2003 г.)
[8] F.V. Tkachov, "Programming Education: a Russian Perspective", сс. 69-77 в [7]. Русский перевод: http://www.inr.ac.ru/~info21/texts/2003-08-JMLC/ru.htm
[9] Н.Вирт, «Преподавание информатики: потерянная дорога», приветствие на открытии Международной конференции по преподаванию информатики ITiCSE г. Аархус (Дания), 24 июня 2002 г., http://www.inr.ac.ru/~info21/texts/2002-06-Aarhus/ru.htm
[10] А.А.Колташев, Н.В.Чистяков, А.И.Попков, Н.П.Кучер, Ф.В.Ткачев: «Система ИТ-образования с точки зрения российских национальных интересов и научно-образовательных традиций». Доклад проекта Информатика-21 на 6й Международной конференции «Проблемы систем информатики», Новосибирск, 27-30 июня 2006 г.; http://www.inr.ac.ru/~info21/texts/2006-06-Ershov/welcome.html
[11] "Microsoft targets amateur programmers", CNET News.com, June 29, 2004.
[12] J.M.Fox, "Software and its development", Prentice-Hall, 1982; русский перевод: Дж.Фокс, «Программное обеспечение и его разработка», Мир, 1985.
[13] Н.Вирт. Интервью журналу Эксперт-Сибирь, №40, 24 октября 2005; http://www.inr.ac.ru/~info21/texts/2005-10-Expert/final.pdf
[14] «Путин обещает России Бангалор», 12 января 2005, http://www.inauka.ru/science/article51813/print.html
[15] А.А. Терехов (Microsoft), доклад на Международной конференции «Современные информационные технологии и ИТ-образование», МГУ им.Ломоносова, 19-21 сентября 2005 г.
[16] http://www.lenta.ru/news/2006/08/11/flaw/
[17] The Second NATO Software Engineering Conference, 1969. http://www.cs.ncl.ac.uk/people/brian.randell/home.formal/NATO/nato1969.PDF
[18] Программа курсов по ИТ-профилю Массачусетского технологического института (MIT, США): http://www.ci.ru/inform14_04/p_04.htm
[19] В.Г. Парфенов, интервью в Компьютер-Информ, №14, 26.07.2004; http://www.ci.ru/inform14_04/p_04.htm
[20] «Русский детский фольклор». Уч. пособие. Составитель Ф.С. Капица. Изд-во Флинта, 2002, 320 с.
[21] «Вирт предлагает Оберон», репортаж Новосибирской ГТРК, 3 октября 2005 г., http://novosibirsk.rfn.ru/rnews.html?id=8997&cid=7
[22] I. Joyner, "C++?? A C++ Critique", The ModulaTor, Nr. 10&11/Nov&Dec-1992. http://www.modulaware.com/mdlt28.htm
[23] P. J. Moylan, "The Case Against C", Tech Rept EE9240, University of Newcastle, July 1992. http://www.comp-inspirations.com/docs/casec.pdf
[24] R. Brega, доклад в [6].
[25] А.А. Корнилов: «Курс истории России в XIX веке», Высшая школа, Москва, 1993.
[26] D.Intersimone (vice-president, Borland), "Borland Gears Up", аудио-интервью журналу Dr. Dobb's Journal, Architecture & Design, August 04, 2006.
[27] UNESCO Computing Curricula 2001: Computer Science. Русский перевод: http://se.math.spbu.ru/cc2001/
[28] VI Международная (Ершовская) конференция «Проблемы систем информатики», 27-30 июня 2006 г., Институт систем информатики СО РАН, Академгородок, Новосибирск.
[29] Г. Лебон (Ле Бон) «Психология народов и масс», СПб, 1896.
[30] G. Palast, "Bill Gates: Killing Africans For Profit And P.R.", July 14, 2003, http://www.gregpalast.com/detail.cfm?artid=235&row=1
[31] D.Welton, "The Economics of Programming Languages", BYTE (www.byte.com), April 18, 2005.
[32] Н.В. Гредина (УрГПУ) «Структурирование информации в обучении информатике».
[33] А.А. Шалыто (СПбГУ ИТМО) «Поговорим о главном», http://is.ifmo.ru; «А ларчик просто открывался», PCWEEK № 35/2004, с.56, 59.
[34] J.Spolski, http://www.joelonsoftware.com/printerFriendly/articles/ThePerilsofJavaSchools.html