П.С.Ковальчук, *студент*
(Белорусский государственный экономический университет)

Феномен технологии JAVA

В данной работе рассказывается о вещах, имеющих непосредственное отношение к основе Java-технологии — языку Java. Вот некоторые из них: HotJava, JavaScript, Oak, Self, Joe, JDBC, PJava, Spring, Clarity. Цель работы: проанализировать (насколько это возможно) истоки, суть и перспективы этой технологии, применение ее в учебно-познавательном процессе.

Причины появления Java. В 1995—96 гг. резко обострился кризис в индустрии разработки ПО — кризис объектно-ориентированного программирования (ООП). Если вернуться на десяток лет назад, встает целый ряд вопросов, требующих ответа.

1. Почему на роль флагмана (уж если крупные фирмы были кровно заинтересованы в становлении ООП) был выбран только что появившийся на свет довольно сложный и весьма противоречивый язык C++? А язык Smalltalk, простота и ясность концепций которого не подвергалась сомнению, оказался незаслуженно забыт и, по сути, заброшен на многие годы?

2. Удалось ли решить главную задачу, ради которой велась коренная перестройка программной индустрии: сократить затраты на разработку и сопровождение новых программ и систем?

3. Возросли ли с внедрением ООП надежность и качество больших программных комплексов?

4. Решены ли проблемы зависимости ПО от используемой платформы (процессор + ОС)?

Независимость от архитектуры — изюминка Java-технологии. Существует пять основных принципов, декларированных разработчиками языка:

независимость от архитектуры (architecture neutrality);
простота (simplicity);
надежность (robustness);
безопасность (security);
расширяемость (extensibility).

Разработчики Java-технологии (здесь язык как бы отходит уже на второй план) пошли по пути интерпретации кода. Схема, используемая при реализации языка Java, во многом схожа с принципами P-кода.

В схеме, использованной в Oak и унаследованной Java, применен следующий подход: исходный текст на языке Java транслируется специальным конвертером (Java Compiler) в Java-байт-код. При использовании в Internet/Intranet этот код передается по каналам связи на удаленную машину, где попадает в загрузчик байт-кодов (Bytecode Loader). Затем он подвергается специальной динамической проверке, которую осуществляет верификатор байт-кодов (Byte-code Verifier). Предварительные проверки, возложенные на верификатор, гарантируют корректность типов параметров всех пришедших байт-

конструкций, отсутствие переполнения стека операндов и, наконец, корректность доступа к полям объекта с учетом атрибутов (private, public или protected). Это предпринято с целью обеспечить безопасность исполняемого кода, а также для проведения скрытой оптимизации. Затем проверенный байт-код передается интерпретатору, который посредством исполняющей системы виртуальной Java-машины осуществляет привязку к конкретной архитектуре.

Для достижения бинарной переносимости нет надобности отказываться от компиляции в пользу интерпретации. Суть подхода Франца состоит в том, что компилятор транслирует исходный текст в компактный промежуточный код особой структуры с сохранением всей информации, необходимой для последующей оптимизации и генерации кода. Этот объектный код затем исполняется на целевой машине, но несколько необычным образом: загрузчик на каждой архитектуре специальный и, прежде чем приступить к своим обычным функциям, он осуществляет оптимизирующую генерацию загружаемого кода.

Проблема унификации операционной среды. В рамках технологии Java проблема унификации графического интерфейса решается за счет инструментария Abstract Windowing Toolkit (AWT), встроенного на уровне классов в систему программирования Java.

Операционная система JavaOS. Построена JavaOS на основе ставшей весьма популярной в последние годы концепции микроядра.

Микроядро в случае JavaOS имеет необычный вид, так как оно формируется из двух слоев: виртуальной Java-машины и, если так можно выразиться, "наноядра" (nucleus), где собраны функции загрузки, обработки прерываний, поддержки потоков управления (threads).

Наноядро Java поддерживает три основные абстракции: домен (domain), поток (thread) и дверь (door).

Java-архитектура. В настоящее время существует четыре варианта привязки Java-программы к оборудованию: через Web-броузер, через ОС для настольного компьютера, через компактную ОС для встроенных систем и, наконец, через JavaOS.

Java-платформа. Она состоит из виртуальной Java-машины и разных наборов API-интерфейсов. Она включает в себя базовую Java-платформу (Java Base Platform) и расширенную Java-платформу (Extended Java Platform).

Интеграция языка и операционной системы. Наконец-то на уровне компьютерной индустрии происходит ломка представлений о том, что должно быть первично: аппаратура или программное обеспечение. Ведь как дело обстояло до сих пор: разработчики новой архитектуры при проектировании набора команд процессора, его регистров, стеков и прочих неотъемлемых атрибутов современных вычислительных устройств исходили из множества действительно важных факторов. Но вот особенности тех целевых языков, которые в дальнейшем будут работать на этой платформе, практически не учитывались.

Появление Java послужило катализатором перехода крупных коммерческих фирм (IBM, Microsoft, Hewlett-Packard, Borland и

др.) на новые перспективные технологии программирования, которые даже безотносительно использования в Internet могут послужить отправной точкой отсчета новой технологической революции.

Применение данной технологии в образовании. Очевидно, что область применения данной технологии довольно широка. Я приведу лишь несколько возможностей ее использования в познавательном процессе. Используя Java-технологию, можно реализовывать тестовые программы для проверки и оценки знаний студентов. Преимущество использования данной технологии в том, что она позволяет осуществлять централизованный контроль, программы-тесты имеют Windows подобный интерфейс, надежно защищены, совместимы с аппаратурой и довольно просты в реализации. Данная технология позволит создать локальную сеть в университете для обмена информацией между преподавателями при минимуме затрат на поддержание ее работоспособности, облегчить доступ к данным и автоматизировать процесс их сбора и организации, организовать централизованный доступ к службам Internet как преподавателей, так и студентов.

Л.К.Голенда, *доцент*
А.А.Лысюк, *студент*
(Белорусский государственный экономический университет)

Коммуникационная сеть SWIFT в Республике Беларусь (Учебный банк: лабораторная работа)

Вопрос о коммуникационных сетях в Республике Беларусь встал особенно остро в период перехода к рыночной экономике, так как именно в это время возникло много финансовых компаний, совместных предприятий, коммерческих банков, которые поддерживают связи с иностранными партнерами.

Анализ разработанных и эксплуатируемых автоматизированных систем в Республике Беларусь в настоящее время (система "Мебиус-банк", продукты фирмы "Надежные программы", "Операционный день фирмы "SoftClub" и др.) показывают, что ни в одной из них не уделено должного внимания возможностям подключения к международным коммуникационным сетям. Причины сложившегося положения кроются в том, что в нашем государстве только начинают использоваться возможности коммуникационных сетей, тогда как в развитых странах мира все имеющиеся системы непосредственно подключены к международным коммуникационным сетям. Будущие специалисты должны хорошо представлять рынок коммуникационных сетей, их возможности, перспективы развития, а также представлять процесс работы в каждой из них.

Поэтому высшие учебные заведения Беларуси, готовящие экспертов для работы в финансово-кредитных учреждениях, должны создать эффективную и гибкую систему обучения, которая является важным условием повышения профессионального уровня. Главную