УДК 681.3

Е.М. Лаврищева

Технология электронного обучения базовым аспектам программной инженерии

Предложен оригинальный веб-сайт (*http://sestudy.edu-ua.net*) «Программная инженерия. Технологии и обучение», обеспечивающий электронное обучение спектру технологий разработки сложных программ из готовых компонентов повторного использования, онтологии представления знаний о предметных областях, предметно-ориентированному *DSL* для описания задач доменов, теории взаимодействия систем и сред между собой, нормативным курсам «Программная инженерия», *Java*, *C#*.

An original web-site (http://sestudy.edu-ua.net) «Software Engineering. Technologies and Education» is suggested, providing an e-learning to the spectrum of technologies of a complex program development from the prepared components repeated use, the ontology of representation of the knowledge about subject areas, in-oriented DSL for the description of the tasks of domains, the theory co-operations of the systems and environments between itself, to the normative courses «Software Engineering», Java, C#.

Запропоновано оригінальний веб-сайт (*http://sestudy.edu-ua.net*) «Програмна інженерія. Технології і навчання», призначений для електронного навчання спектру технологій розробки складних програм з готових компонентів повторного використання, онтології подання знань про предметні області, предметно-орієнтованому *DSL* для опису завдань доменів, теорії взаємодії систем і середовищ між собою, нормативним курсам «Програмна інженерія», *Java*, *C#*.

Введение. Программная инженерия (ПИ) с момента своего возникновения заняла центральное место среди компьютерных наук, в информатике, информационных системах и технологиях. ПИ ориентирована на разработку программного обеспечения прикладных и информационных систем разного назначения. *Это система методов*, *средств и дисциплин планирования*, *разработки*, *эксплуатации и сопровождения программного обеспечения*, *готового к внедрению*. Основные принципы ПИ – продуктивность, индустрия и качество. Основу ее составляет ядро знаний *SWEBOK* (*Software Engineering body of Knowledge*, *www.swebok.com*, 2001 г.), созданное международным комитетом *IEEE* и *ACM*, и состоит из 10 разделов знаний (*area knowledge*). Первые пять разделов – это инженерия требований, проектирование, конструирование, тестирование и сопровождение программного обеспечения (ПО). Следующие пять разделов – организационные. К ним относятся: управление проектом, конфигурацией, качеством, методы и средства инженерии (технологии) ПО. Этим разделам соответствуют процессы жизненного цикла (ЖЦ) стандарта *ISO*/*IEC* 12207, которые также ориентированы на реализацию ПО. Исследования ПИ показали, что разделы знаний *Swebok* не охватывают вновь появившихся целевых объектов ПИ (ПС, СПС, домены, семейства систем и др.), технологических и инструментальных средств обработки программ в современных операционных средах (*Vs.Net*, *IBM*, *Intel*, *Corba* и др.). Самое главное – они не отображают аспектов индустрии программных продуктов (ПП) и новых способов описания доменов, решения задач защиты данных, экономики затрат, стоимости ПП и др.

Создатели ПИ считают, что главное его назначение – это обеспечение индустрии ПП. Под *индустрией* подразумевается производство разных видов продуктов массового применения и средств их изготовления, т.е. основная задача любой индустрии – массовый выпуск продукции и получение прибыли [1–5]. Индустрия *программной продукции* мировых фирм-производителей базового ПО (*Microsoft*, *IBM*, *Corba*, *Java*, *Intel* и др.), а также индийской фирмы по разработке и обновлению наследуемых (*legase*) систем программ дает огромные прибыли. В нашей стране за последние два десятилетия развитие индустрии ПП фактически отсутствовало, не было государственных и научных программ ее развития. Вместе с тем в появилось много разных зарубежных предприятий коммерческого типа, изготавливающих необходимые им продукты силами студентов и специа-

листов Украины [5]. Обучение студентов отдельным аспектам индустрии в вузах практически находится на начальной стадии. Признано, что индустрия ПП в Украине отстает более чем на 20 лет. Для устранения такого отставания правительство Украины в 2011 г. провело международный научный конгресс, обсуждение в Верховной Раде и создало государственную программу развития индустрии ПП до 2017 г.

В русле решения этих задач в данной статье предложены дисциплины ПИ, регламентирующие технологию создания разных видов ПП и новый подход электронного обучения студентов вузов Украины разным аспектам ПИ, способствующим получению знаний по *производству* ПП. В основе подхода – технологические линии изготовления отдельных элементов ПП, реализованные на веб-сайте (*http://sestudy.edu-ua.net*). Они идеологически близки сборочному конвейеру академика Глушкова (1975 г.) для выпуска разных видов ПП из готовых компонентов повторного использования (КПИ) и артефактов. Сайт предназначен для обучения основам ПИ: линиям (их 15), технологиям, применению готовых продуктов в новых ПП, а также посредством изложенных методов и средств учебника «Программная инженерия» автора. Технологии обучения включают в себя стандарты ЖЦ, теории преобразования типов данных *GDT–FDT*, конкретно реализованным онтологиям доменов – ЖЦ, вычислительной геометрии как нормированного курса в КНУ, а также современным сервисным средствам, необходимым при выполнении разных видов деятельности по производству ПП [6–12].

Далее рассмотрены базовые элементы индустрии программной продукции и описаны технологии обучения аспектам ПИ на данном вебсайте.

Базовые элементы ПИ для поддержки индустрии программных продуктов

Дисциплины ПИ. Для отражения проблем индустрии в части регламентированного управления выпуском ПП на фабриках программ автором предложена новая классификация разделов знаний – дисциплины ПИ (рис. 1) [6]:

научная дисциплина – классические науки (теория алгоритмов, множеств, доказательства, математическая логика и пр.), теория программирования и соответствующие языковые средства проектирования абстрактных моделей и архитектур целевых объектов, стандарты ЖЦ, теория интеграции (сборки) и др.;

инженерная дисциплина – совокупность технологических средств и методов проектирования ПП с помощью стандартных моделей ЖЦ, техника анализа ПП, инженерия требований, приложений, доменов по линиям (*Product Lines*) продукта, сопровождение, изменение и адаптация ПП к другим платформам и средам;

управленческая дисциплина – общая теория управления, адаптированная к коллективной разработке ПП, включающая в себя графики работ, наблюдения за их выполнением, управление рисками, версиями ПП и сопровождение;

экономическая дисциплина – совокупность методов экспертного, качественного и количественного оценивания промежуточных артефактов и конечного результата процессов ЖЦ, а также экономических методов расчета времени, объема, трудозатрат и стоимости изготовления ПП;

производственная дисциплина – линии изготовления отдельных компонентов и ПС из готовых КПИ (сервисов, аспектов, агентов и пр.), накопленных в глобальных библиотеках и репозиториях, а также конфигурирования, тестирования и оценивания качества продукта.

Рис. 1. Структура дисциплин ПИ

Каждая из приведенных дисциплин представляет собой методику, содержащую последовательность действий и форм документов линии, сориентированных на регламентное выполнение определенной деятельности при изготовлении программного элемента с учетом инженерных и оценочных методов измерения разных сторон ПП.

Индустрия ПП базируется на наборе технологических линий (ТЛ) программ, для которых автором статьи разработана технология их изготовления [7], а также на продуктовых линиях (*Product Line*) [8], используемых для изготовления коммерческих систем семейств.

Линии продуктов. Основу технологии создания ТЛ составляет процессный подход, получивший название *технологической подготовки* разработки (ТПР), включающий в себя работы по формированию структуры, схемы линий из процессов и действий для производства отдельного элемента будущей ПС (рис. 2).

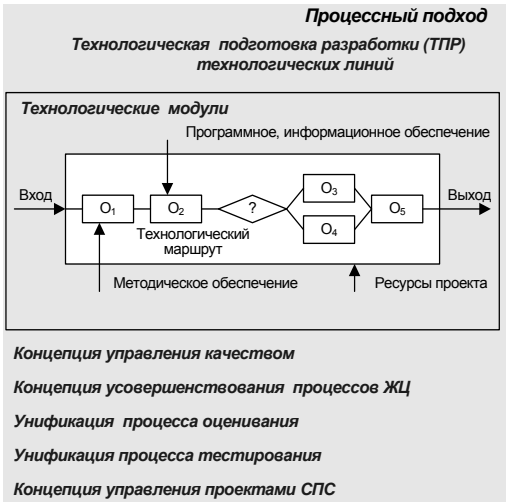

Рис. 2. Схема технологической подготовки разработки

Главное требование к построению ТЛ для производства программ или компонентов – комплектование линии из процессов ЖЦ, соответствующих разрабатываемому домену, стандартных инструментов и технологических модулей (ТМ) или новых разработанных для отображения специфики ПрО в операционной среде (ОС) и комплекс нормативно-методических документов. Для ТЛ подбираются готовые ресурсы (ГоР), КПИ, средства и инструменты порождения и реализации функций программ или отдельных их элементов, а также строится маршрут плана управления процессами ТЛ для заполнения определенных шаблонов (патернов), фиксирующих проектные решения, изменения состояний элементов и оценки показателей качества ПП [8–10].

Процессы ТЛ создаются с учетом требований международного стандарта *ISO*/*IEC* 12207–96, 2007 и ДСТУ 3628–99, подкрепляются отобранными методами, средствами и инструментами для проведения соответствующих изменений состояний элементов ТЛ. Они описываются технологическим маршрутом (см. рис. 2) операций процессов и картой подключения инструментов и методик к соответствующей линии. На вход маршрута подаются объекты и задание на тип обработки. В целом маршрут реализует концепцию управления разработкой, качеством, тестированием, т.е. он определяет целенаправленную деятельность выполнения процессов и операций построения и трансформации элементов ПП по сформулированной методике и форме представления.

Продуктовая линия (*Product Lines*) *SEI* включает в себя *product line* (линия продуктов) и *product family* (семейство продуктов, в дальнейшем ПС) отождествляют.

Линии определены в словаре *ISO*/*IEC FDIS* 24765:2009(*E*) – *Systems and Software Engineering Vocabulary* как «группа продуктов или услуг, имеющих общее управляемое множество свойств, удовлетворяющих потребностям определенного сегмента рынка».

Моделями представления деятельности разработки ПС институтом *SEI* есть – инженерная и процессная модели (рис. 3).

Деятельность по *разработке КПИ* предусматривает определение области СПС, планирования изготовления множества элементов ПС с учетом контекста их применения, ограничений и стратегии производства. Деятельность по *разработке* ПС предполагает построение плана реализации каждой ПС на основе множества разработанных КПИ, ГоР и общего плана производства СПС. Деятельность *по управлению* ГоР ориентирована на координацию работ и

есть двухуровневой, предусматривает решение задач организационного и технического управления СПС, а также каждым ее членом.

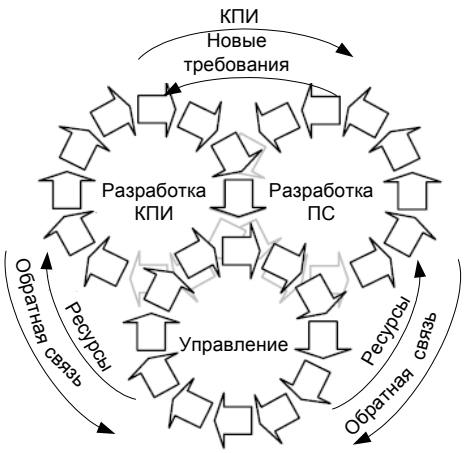

Рис. 3. Модель инженерии разработки ПС

Модель инженерии соответствует инженерии ПП с тремя видами деятельности – разработкой КПВ, разработкой из них ПС и управлением КПИ.

В *процессной модели* выделяется множество процессов, выполняемых на двух уровнях – доменной инженерии (или инженерии ПрО), которое еще называют разработкой «для обеспечения повторного использования» (*for reuse*), и инженерии ПС, называемой также разработкой «с использованием КПИ» (*with reuse*) [8].

На линии сборки используются готовые КПИ, улучшающие время и уровень готовности ПС и способные обеспечить весь цикл производственных работ в соответствии со специфическими требованиями и потребностями сегмента рынка ПП.

Главный элемент на линии – инструментально-технологический модуль – ИТМ (редактор, компилятор, транслятор, верификатор, прувер, тестировщик, брокер интерфейсов, генератор данных и др.). Такие ИТМ поддерживают основные операции линий по производству отдельных элементов ПП [13]. Многие ИТМ как инструменты разработки любых программ стандартизованы и находятся в таких библиотеках как *Demral*, *IP*, *Draco*, библиотеки Буча и др.

Сборочный конвейер (*Assembly line*/*belt assembly line*) определяет автоматизированную одну ли более линий сборки на фабрике. Каждая линия повышает производительность работы исполнителей, улучшает условия их работы, сокращает число инженеров на процессах линий, повышает качество продукции и снижает себестоимость продукции.

Именно о таком конвейере мечтал Глушков, когда говорил (1975 г.): «Пройдет 20–30 лет и сложные программы будут выпускаться по принципу сборочного конвейера, как в автомобильной промышленности Форда». Эта идея сборочного конвейера Глушкова нами последовательно развивалась много лет и реализована теоретически и практически. Сегодня можно сказать, что мы уже приблизились вплотную к реализации сборочного конвейера фабрики программ, базисом которого служат линии изготовления отдельных элементов выходного продукта и линия сборки из них сложных ПП, реализованных на веб-сайте.

Соответственно словарю, конвейер и его линии могут быть автоматизированными (как станок) или частично автоматизированными на фабрике. В индустрии ПП рассматриваются только частично автоматизированные линии, так как «комплектующие детали» фабрики *не материальные*, они относятся к классу творческих «неосязаемых» объектов, невидимых внутри компьютера, а лишь внешне, как описание некоторого артефакта, КПИ, специфицированных языков программирования (ЯП).

Поскольку линия ПП состоит из процессов, выполняемых с участием одного или больше разработчиков, каждый из них вводит необходимые входные данные для процесса производства программ на линии (например, готовый КПИ, его интерфейс и данные, необходимые для сборки с другими).

В машиностроительной промышленности могут быть непрерывные линии, функционирующие без участия рабочих на промежуточных действиях конвейера, либо с остановками в установленные промежутки времени или в зависимости от промежуточного результата. На фабрики программ инженерные специалисты – менеджеры, разработчики, верификаторы, валидаторы, эксперты, тестировщики, оценщики каче-

ства и другие выполняют работы полуавтоматизированно.

Фабрики программ. Под этим термином понимается интегрированная инфраструктура для организации сборки готовых ресурсов в ПП, необходимых государственным, коммерческим и другим заказчикам. Центр фабрики – сборочный конвейер, оборудованный продуктовыми линиями, набором средств, инструментов и сервисов для автоматизированного выполнения процессов линий в операционной среде [13–15].

Фабрика функционирует посредством выполнения линий конвейера и соответствующего набора средств поддержки процессов разработки простых и сложных продуктов. Линиям разработки простых продуктов, как правило, соответствует ЖЦ, например, реализованный в среде *MS.Net* с использованием каркасов (*framework*), *DSL*-языка (*Domain Specific Language*) и др. Линия разработки сложных ПП по существу есть сборочной из готовых КПИ, которые подбираются из библиотек и репозитариев [16–19].

Исходя из практики автоматизированной сборки разнородных программ в ОС ЕС и опыта современных фабрик программ международных фирм (*IBM*, *OMG*, *Microsoft*, *Oberon* и др.) определен общий набор элементов:

- ***готовые программные ресурсы*** (артефакты, программы, системы, *reuses*, *assets*, КПИ и т.др.);
- ***интерфейс*** как спецификатор паспортных данных готовых разнородных ресурсов, независимо от языков программирования и языков спецификации интерфейса (*IDL*, *API*, *SIDL*, *WSDL*, *RAS* и пр.);
- ***операционная среда*** с системными программными средствами и инструментами для поддержки сборочного конвейера из разнородных программных ресурсов;
- ***технологические линии*** (ТЛ) или ***продуктовые линии*** (*Product Lines*) для производства ПП;
- ***метод разработки продукта***;
- ***сборочный конвейер***.

Эти элементы – необходимые атрибуты любых, в том числе и известных фабрик программ [20–23] и др.).

Метод сборочного программирования – базис перехода к индустрии ПП

Одна из задач программной инженерии – создание теоретических и прикладных основ построения сложных программ из более простых с помощью линий разработки и объединения разнородных готовых КПИ в *новые* программные структуры (ПС, СПС, семейство систем, продуктовые линии – *Product lines* и пр.). В 90-х годах прошлого столетия сформировалось сборочное программирование, которое поддерживал академик А.П. Ершов [15], объекты которого – разные программные объекты (модули, компоненты, программы, сервисы и др.) и их паспорта, описываемые ЯП и которые накапливались в библиотеках (теперь в электронных и репозиториях Интернета), а также в архивах разработчиков ПП [16–19].

Основа этого программирования – метод сборки для соединения (взаимосвязи) разноязычных объектов в ЯП и их обработки. Метод базируется на теории фундаментальных типов данных А. Хоара и К. Вирта, алгебраических системах преобразования нерелевантных типов данных (ТД) в требуемые [17, 18].

Источником объединения пар разноязычных объектов служат операторы удаленного вызова (*RPC*, *RMI* и др.), задающие список передаваемых друг другу параметров и значений формальных параметров. Аксиомы и утверждения аппарата алгебраических систем обеспечивает проверку ТД на соответствие их описанию и генерацию их к необходимому типу связываемого объекта в специальном модуле-посреднике (мостику двух объектов), т.е. он содержит генерированные операторы эквивалентного преобразования ТД по каждому удаленному вызову. Аналогичный подход генерации интерфейсных модулей-посредников (*stub*, *skeleton*) реализован в системе *Corba*.

Программирование КПИ соответствует сборочному конвейеру. В нем роль комплектующих «деталей» выполняют готовые КПИ разной степени сложности, а роль стыковки – интерфейсные посредники.

Итак, сборочное программирование это:

• один из методов программирования подчиняется общим закономерностям и функциям общих методов типа «объект–операция»;

• одна из форм поддержки повторного использования готовых программных ресурсов и объектов есть КПИ;

• экономически выгодный метод сборки стандартизованных готовых КПИ и их интерфейсов.

Объект сборки как базовое понятие ООП имеет свойства (наследование, полиморфизм и инкапсуляция), включает в себя данные и операции (методы) для взаимодействия между собой через удаленные вызовы и сообщения. Объекты сборки могут обладать общими свойствами и методами, образовывать классы и быть в нем экземплярами класса (например, язык *C++* имеет несколько библиотек классов общего применения). Для технологичности сборки все объекты должны иметь паспорта, содержащие данные, необходимые для информационного соединения и организации общего функционирования в рамках более сложной объединенной структуры.

Операции сборки объектов выполняют функции их обработки, основанные на их паспортных данных и типах операций соединения (замены, удаления, добавления и др.) между собой разноязычных объектов и новых операций взаимодействия программ из готовых КПИ, созданных в одной среде с переносом в другую среду. Информация в паспортах должна быть систематизирована и выделена в группы: передачи данных, совместного использования файлов и др.

В рамках работ по компонентному программированию как развитию сборочного программирования автором определены операции компонентной алгебры, ориентированные на сборку компонентов [18]:

redevelop PS (*IntA*, *IntB*) – преобразование типов данных *A*, *B*;

linkconfig SPS (A^{l1}, B^{l1}, C^{l1} ($Int^{idl}A$, $Int^{idl}B$, $Int^{idl}C$) – сборка методом конфигурации программ *A*, *B*, *C* в одном ЯП *L*1 и списком параметров в интерфейсах в языках *IDL* и *API*;

makeaway PS (*A*) – удаление из системы *PS* объекта *A*;

add PS (*A*, *C*) – добавление компонентов *A*, *C* к системе *PS*;

insert $F \Rightarrow PS$ – вставка объекта *F* в систему *PS*;

rename $A \Rightarrow B$ – замена имен объектов;

regoing $x, y \Rightarrow BD$ – передача данных *x*, *y* в соответствующем формате в базу данных БД;

rego TD (*X*,*Y*) – передача данных после преобразования нерелевантных значений;

redact A (*PS*) – изменение программы *A* в системе *PS*.

Следующим шагом в направлении развития данного программирования будут новые принципы обеспечения взаимодействия (интероперабельности) систем и сред между собой [25–28]. Созданный стандарт *OSI* (1992) не решал вопросы совместной работы разных сред. Каждая развитая операционная среда (*VS.Net*, *IBM*, *Corba*, *Intel*, *Apple* и др.) имеет свои особенности автоматизации сложных программ из готовых КПИ выполнения, которые различаются между собой и требуют формальных средств для перехода в другую среду. В настоящее время это крайне важно, поскольку в этих средах создано огромное количество готовых к употреблению программ, сборка из них сложных систем экономически выгодна и эффективна. Поэтому для обеспечения взаимодействия систем и сред автором разработан оператор: *interconnect PS* (*A*, *B*, *C*, *IntA*, *IntB*, *IntC*).

После исследования принципов взаимодействия в упомянутых средах два студента (Островский А. – МФТИ и Радецький И. – КНУ) выполнили оригинальную программную поддержку оператора взаимодействия в среде системы *Eclipse* генерирующего программирования [24–28].

Процесс сборки может осуществляться ручным, автоматизированным и автоматическим способами с помощью приведенных операций. Ручной способ менее целесообразен, поскольку описание процесса сборки готовых компонентов требует большого объема рутинных действий. Наиболее приемлемый способ – это автоматизированная сборка, которая по спецификациям программ осуществляет генерацию интерфейсного посредника для связи компонен-

тов между собой с помощью операций алгебры и стандартных правил соединения (прямого или виртуального) разнородных объектов.

Средства, автоматизирующие сборку программ, называют инструментальными средствами сборочного программирования. К ним относятся средства комплексирования, интеграции; интерфейсные средства описания и использования моделей из совокупности моделей компонентного программирования (модель компонента и интерфейса, модель компонентной системы, модель среды) [19], а также новые средства обеспечения взаимодействия программ и систем, изготовленных в разных средах [22].

Приведем условия применения сборочного метода программирования:

- наличие большого количества разнообразных КПИ и ГоР как объектов сборки;
- паспортизация объектов сборки;
- наличие полного набора стандартных правил сопряжения объектов, алгоритмов их реализации и средств автоматизации процесса сборки;
- технологические линии с набором операций подготовки КПИ, установления связей между ними для образования взаимосвязи систем, подсистем и семейств ПП.

Операции метода сборки реализованы в *инструментально-технологическом комплексе* (ИТК), описание которого приведено далее.

Реализация ИТК как веб-сайта для обучения приемам индустрии ПП

В результате многолетних исследований задач фундаментального проекта НАНУ «Теоретический фундамент генерирующего программирования (ГП) и средства его поддержки» (2007–2011) в отделе «Программная инженерия» ИПС НАН Украины были разработаны: объектно-компонентный метод проектирования доменов, ПС из КПИ; теория взаимодействия, вариабельности и живучести ПС; теория моделирования и адаптации спроектированных ПС в других средах, методология производства семейства ПС (СПС) из совокупности простых линий для изготовления членов семейства ПС (рис. 4) [29–36].

Многие проектные и технологические решения по данному проекту реализованы в ИТК. Основные научные результаты многолетних исследований и разработок отдела представлены в электронной монографии [35] и в других публикациях. Сущность основных результатов такова:

- интерфейс, метод сборки, объектно-компонентный метод проектирования СПС из готовых КПИ [16–19];
- методология изготовления программ и систем на основе предметно-ориентированного языка *DSL* (*Domain Specific Language*) и КПИ (*Reuses*, *Assets*, *Services*) [35–39];
- теория взаимодействия (интероперабельности программ и систем), вариабельности (изменяемости и адаптации СПС к новым условиям современных сред), живучести (отказоустойчивости и восстанавливаемости систем) ПС [25–28, 34, 35];
- концепция технологических линий производства и фабрик программ [7– 10, 30–33];
- качество ПС и процессов ЖЦ [11, 12, 34–38];
- методика представления знаний КПИ в репозитории [35, 40];
- технология работы с инструментами ИТК – *Protege*, *Eclipse*, *Corba*, *Eclipse –DSL*, *Ant*, *C#*, *Java*, *Basic* [11, 29, 30, 35] и линиям продуктов в ИТК [39].

Рис. 4. Функциональная структура задач ИТК

Реализованные в ИТК новые средства ориентированы на производство ПС и СПС из готовых КПИ по простым линиям обработки КПИ на разных ЯП [35]. ИТК фактически отображает реализацию идеи сборочного конвейера академика Глушкова. Технология работы с линиями разработки, сборки, обеспечения взаимодейст-

вия программ и систем в современных средах осуществляется на веб-сайте (*http://sestudy.edu-ua.net*) и на экспериментальной фабрике программ сайта факультета кибернетики КНУ имени Тараса Шевченко (*http://programsfactory.univ.kiev.ua*). Фабрика программ реализована студентами (Ароновым А. и Дзюбенко А.) под руководством преподавателя курсов «программная инженерия» и «технологии программирования ИС» и автора данной работы. С 2006 г. также работает сайт *www.intuit.ru*, в котором размещен учебник по методам и средствам программной инженерии. Данный курс автор читала с 2001 г. в филиале МФТИ при ИК имени Глушкова НАН Украины.

Функции и структура веб-сайта. Данный сайт разрабатывался как инструментарий ТП и одновременно демонстрировались отдельные моменты ПИ по новому курсу *SE – Software Engineering* (*www.swebok.org*) «Программная инженерия» на лекциях в КНУ. В результате возникла идея внедрить разработанный ИТК в систему обучения студентов и аспирантов аспектам ПИ (рис. 4).

Перед веб-сайтами ставилась задача способствовать организации процесса обучения на многочисленных технологических инструментах и средствах, а именно, на линиях ИТК: разработка программ и КПИ, обслуживание КПИ в репозитории, генерация компонентов и доменов в современных предметно-ориентированных языках типа *DSL*, обеспечение взаимодействия систем и программ между собой и средами, моделирования домена из курса КНУ – вычислительная геометрия, сборка ПС и СПС из готовых КПИ в более сложные структуры, расчет стоимости, затрат и качества созданных ПС и др.

В курс обучения были включены электронные технологии программирования в современных ЯП *Java*, *C#*, *C++*, *Basic*, а также обучения компьютерным предметам – программная инженерия, вычислительная геометрия и пр.

Планируется реализовать на фабрике программ КНУ технологию *Product Lines*.

Так, была выбрана стратегия обучения многим аспектам технологии промышленного изготовления программ и систем. Для постепенной и последовательной реализации этой стратегии в ИТК взят современный аппарат дизайна веб-сайта, современные действующие системы, поддерживающие те или иные аспекты технологии разработки программ:

- система *Protégé* для моделирования онтологий доменов и предметных областей;
- *Eclipse* как инструмент подключения разного рода программных и системных компонентов к среде ИТК с помощью механизмов плагинов;
- *VS.Net* как многофункциональная организация коллективной разработки новых систем, в том числе через Интернет на разных ЯП в ООП, *UML*, а также поддержки вычислений программ в *Cloud Computing* – системах (*Azure*, *SkyDriven*, *Amazon*, *VShpere* и др.);
- система *CORBA* с универсальным брокером по обеспечению взаимодействия разноязычных программ с помощью проверенного многолетней практикой аппарата интерфейса *Stub* для клиентской стороны и *Skeleton* – для серверных приложений;
- новый предметно-ориентированный язык *DSL* (*Domain Specific Language*) графического типа для проектирования систем (*Domains*, *Applications*, *Families*) и инструментальные средства поддержки описаний *DSL* в *Eclipse–DSL*, *Microsoft DSL Tools* и др.

Каждый раздел содержит подразделы с ключевыми словами, которые обозначают название линии разработки. Все разделы и подразделы построены по общему образцу. Разделы включают в себя общее теоретическое описание, пример, иллюстрирующий смысл описания (в большинстве случаев разработан средствами одной из поддерживаемых сред ИТК), реализацию и выполнение примера на соответствующем инструменте этой среды.

В реализации данного сайта и отдельных линий технологии изготовления отдельных аспектов программ из готовых КПИ принимали участие сотрудники отдела (Зинькович В.М., Куцаченко Л.И.), магистранты МФТИ (Островский А., Скотников И.), дипломанты КНУ имени Т. Шевченко (Радецкий И. и Аронов А., Дзюбенко А.).

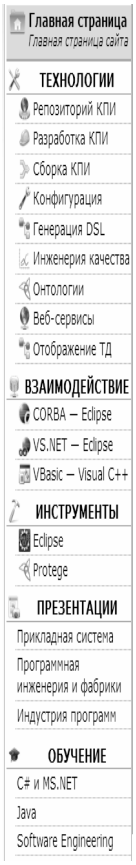

Рис. 5. Разделы главной страницы сайта

Данный сайт содержит разделы, перечень которых приведен на рис. 5.

Технологии – фабрика программ КНУ, репозиторий КПИ, их разработка, сборка, конфигурирование, генерация описания программ в языке *DSL*, онтология вычислительной геометрии, оценка затрат, вебсервисы, генерация ТД из стандарта *ISO*/ *IEC* 11494.

Взаимодействие программ, систем и сред по моделям интероперабельности этих целевых объектов: *Corba–Eclipse–Java*, *VS.Net C#–Eclipse*, *Basic–C++*.

Инструменты – *Eclipse*, *Protégé* и технология их использования;

Презентации – прикладная система ведения зарубежных командировок в НАНУ [39];

Обучение: технологии разработки программ в *C# VS.Net*, языка *Java* и электронного учебника по предмету «Программная инженерия», выполняемых на сайте КНУ *http*://*programsfactory*.*univ*.*kiev*.*ua*.

Подразделы главных разделов (позиций) сайта отображают реализацию простых линий разработки элементов программ и систем.

Все разделы построены по одной схеме рамок для нажатия в них текста:

- описание смысла технологии на конкретной линии;
- метод реализации на примере некоторой программы;
- загрузка *exe*-кода для выполнения на рабочем столе;
- возврат в сайт для дальнейшей работы.

Техника реализация сайта. Для отображения структуры и содержания линий была выбрана архитектура, промежуточная между статическими веб-страницами и характерной архитектурой *Model–View–Controller* (*MVC*).

Все страницы, отображающие статьи по тематикам сайта, строятся по единому шаблону, включающему в себя следующие основные детали:

- заголовок, единый для всех страниц, содержащий баннер сайта и его название;
- главное меню, включающее в себя панель выбора языка и ссылки для навигации по разделам сайта;
- панель навигации, содержащую ссылки для перехода к различным подразделам текущей статьи;
- строку местонахождения;
- содержание статьи;
- «подвал», содержащий сведения об авторах сайта.

Формирование динамических компонентов страницы, к которым относятся все перечисленные детали, кроме заголовка и «подвала», осуществляется на языке программирования *PHP*. При этом используется древовидная структура представления разделов и соответствующих им статей, позволяющая без особых усилий формировать все перечисленные детали страницы. Для простоты представления и долговременного хранения информации о заголовках статей и их содержимого используется база данных *SQLite*. В связи с частыми повторами в различных статьях однообразных элементов (например, нумерованных рисунков и таблиц с информацией о скачиваемых файлах), содержание статей, помимо стандартных *HTML*-тегов, может содержать также *XML*-теги, которые перед отображением страницы преобразуются определенным образом в *HTML* при помощи препроцессора.

Основные разделы сайта «Программная инженерия. *Технологии и обучение*»

Главная страница содержит изложение сущности предмета «Программная инженерия» (ядро знаний *SWEBOK* (*www*.*swebok*.*org*), фабрики программ, технология программирования и др.).

Раздел «Технологии»

Разработанная интегрированная технология сайта представлена набором ТЛ в ИТК:

- фабрика программ – это спецификация КПИ и их паспортов, а также обучение программированию *VS.Net* и дисциплинам ПИ на студенческой фабрике программ;
- репозиторий КПИ – составная часть описанной фабрики;
- сборка разноязычных программ и компонентов в ПС посредством конвертирования несовместимых типов данных;
- конфигурация КПИ в сложную структуру ПС с точками (*вариабельности*) возможного изменения программ по требованию заказчика *MS.NET Workflow*;
- описание доменов прикладных областей в языке *DSL* на примере домена ЖЦ стандарта *ISO*/*IEC* 12207–2007 с графическим и текстовым представлением, реализованным в среде *Eclipse-DSL*;
- инженерия качества и затрат продукта с помощью программного модуля *Softest*, предназначенного для прогнозирования оценок трудозатрат и стоимости разработки ПС;
- создание онтологии предметной области на примере предметного домена «Вычислительная геометрия» в среде *Protege*;
- конструирование СПС путем объединения компонентов, использующих разные программные платформы, с помощью веб-сервисов;
- отображение общих и фундаментальных (*GDT* и *FDT*) типов данных стандарта *ISO*/*IEC* 11404 в виде примитивов библиотеки и приближенной к системе *GRID*.
- генерация ГоР и объединение их конфигуратором по модели вариабельности в структуру – программа, член семейства, СПС;
- тестирование программы в среде ОС для получения правильного продукта и сбора сведений об отказах и ошибках, необходимых при оценке надежности;

В целом, реализация линий постепенного изготовления ПС путем объединения элементов простых линий характеризует приближение к фабрике программ.

Раздел «Взаимодействие»

Современный Интернет обеспечивает пользователя разными видами взаимодействия между распределенными системами, средами и их инструментами выполнения.

Программная поддержка взаимодействия программ, систем и сред разработана согласно теории взаимодействия [20, 23], сущность которой состоит в усовершенствовании общих методов и средств адаптации систем при переносе программ из одной программной среды в другую, с использованием механизмов репозитория *Eclipse* в ИТК.

Операционные среды (*VS.Net*, *IBM*, *CORBA*, *Java*, *Eclipse*), включенные в веб-сайт, реализуют процессы ЖЦ разработки разнородных программ и методы их объединения в разные структуры ПП с помощью специальных механизмов связи в каждой среде. Новые средства взаимодействия, реализованные в веб-сайте, с теоретической точки зрения решают проблему миграции систем из одной среды в другую через систему *Eclipse*, создавая «водораздел» между средами указанных распределенных систем. Данный подход не имеет прототипа. Новые механизмы взаимодействия систем между собой апробированы в ИТК на следующих примерах:

- Взаимодействие программ *Visual Basic* ↔ *C++* выполняется через интерфейсный посредник, передающий данные от одной программы к другой и при необходимости преобразовывающий несовместимые типы данных ЯП средствами динамической библиотеки (по методике И. Бея);
- Взаимодействие систем и сред *CORBA* ↔ *Eclipse* платформ *Java* и *MS.NET* осуществляется брокером объектных заявок системы *CORBA* с использованием языка *IDL* для описания интерфейсов взаимосвязей этих систем;
- Взаимодействие сред *VS.Net* ↔ *Eclipse* осуществляется на примере передачи данных программ, разработанных на платформе *Visual Studio*, в репозиторий *Eclipse*, с использованием механизма плагинов.

Раздел «Инструменты»

В состав раздела входит описание средств разработки *Eclipse* для использования при аг-

регации инструментов ИТК больших возможностей по расширению функциональности – механизм плагинов и методическое руководство проектирования моделей предметных областей и последующего их представления средствами нового предметно-ориентированного языка *DSL* средствами *Protégé*. В разделе приведено описание технологии моделирования предметных областей или доменов средствами *Protégé* с примером онтологической модели информационно-технологических ресурсов сети Интернет и системы *Eclipse* для формирования репозитория КПИ и разработки из них новых программ.

Раздел «Презентации»

Раздел содержит по теме программной инженерии три презентации:

- системы автоматизации производственной деятельности отдела международных связей НАН Украины;
- основные принципы создания фабрик программ, структуры, запаса КПИ и ГоР, технических программных и людских ресурсов, методов и средств поддержки процессов разработки на технологических линиях;
- концепции и аспекты индустрии программ и систем, предложенных в упомянутом фундаментальном проекте.

Раздел «Обучение»

Раздел состоит из трех линий:

- электронное обучение современному языку – *C# VS.Net*;
- обучение курсу ЯП *Java* по учебнику Хабибуллина (Санкт-Петербурга) со свободным доступом и включает в себя последовательные действия по проведению процесса программирования, трансляции, выполнения на контрольных примерах автора;
- обучение предмету «Программная инженерия» проводится преподавателем по электронному учебнику Лаврищевой Е.М. (укр.) на данном сайте и на русском языке в Интернете (*www.intuit.ru*).

Заключение. Предложенный сайт предназначен для поддержки технологии производства программ, ПС, СПС и для обучения через набор упрощенных линий общего назначения. В нем реализованы такие главные концепции и методы:

организация взаимодействия разнородных программ и систем с возможностью их переноса в другую операционную среду для работы с данными, которые передаются через интерфейсы или выбираются из баз данных, он-лайн хранилищ данных типа *Azure*;

технология разработки КПИ и их описание в стандарте *WSDL* интерфейсных данных, сохраняемых в репозитории интерфейсов и КПИ для отбора готовых КПИ с целью применения в новых прикладных системах;

сборка разноязычных программ из готовых КПИ репозитория, имеющих паспортные данные, необходимые для объединения и трансформации несовместимых типов данных КПИ, которые передаются между ними и могут располагаться на разных платформах сред выполнения ПС;

описание предметных областей и доменов (ЖЦ стандарта *ISO*/*IEC* 12207 вычислительная геометрия, тестирование) на языке *DSL* и их реализация инструментальными средствами *DSL Tools VS.Net*/*Eclipse–DSL*, включенными в среду ИТК;

генерация примитивных функций преобразования некоторых типов данных (таблиц, массивов, последовательностей и др.) *GDT<=>FDT* стандарта *ISO*/*IEC* 11404.

конфигурация КПИ по экспериментальной модели вариабельности разных программ или членов семейства СПС;

технология обучения с помощью линий разработки программ в ЯП *C#*, *Java*, *Basic*, *C++* в среде *VS.Net*, *Corba* и *Eclipse*, а также обучения электронному курсу предмета «Программная инженерия» по учебнику автора.

Перспективные направления развития:

- доработка метода сборки ПП сервисами и сертификации компонентов на соответствие общепринятым стандартам и адекватно заданным требованиям к СПС;
- усовершенствование модели качества СПС для обеспечения вариабельности и оценки в прогнозных точках вариантов количественных показателей качества ПС;
- развитие веб-сайта путем добавления новых предметов ПИ и компьютерной науки для дистанционного обучения студентов, аспирантов и магистрантов.

1. *Лавріщева К.М.* Концепція індустрії наукового софтвера і підхід до обчислення наукових задач // Проблеми програмування. – 2011. – № 1. – С. 3–17.
2. *Андон П.І.*, *Лавріщева К.М.* Розвиток фабрик програм в інформаційному світі // Вісн. НАН України. – 2010. – № 10. – С. 15–41.
3. *Анісімов А.В.*, *Лавріщева К.М.*, *Шевченко В.П.* Про індустрію наукового софтвера // Conf. Theoretical and Applied Aspects of Cybernetics, Kiev, Feb., 2011, Ukraine.
4. *Андон П.І.*, *Лавріщева К.М.* Теоретичні і прикладні підходи до індустрії програмної продукції / Матеріали пр. «Міжнародного наукового конгресу з розвитку інформаційно-комунікаційних технологій та розбудови інформаційного суспільства в Україні при Кабміні Україні (17–18 лист. 2011 р.). – Київ. – С. 6–7.
5. *Анісімов А.В.*, *Лавріщева К.М.* Початок електронного навчання програмної інженерії в КНУ для виробництва програм і артефактів. Теза. – Там же. – С. 8–9.
6. *Лавріщева К.М.* Програмна інженерія – напрями розвитку // Пр. міжнар. конф. «50 років Інституту кібернетики імені В.М. Глушкова НАН України» – Київ, 2008. – С. 336–345.

7. *Лаврищева Е.М.* Классификация дисциплин программной инженерии // Кибернетика и системный анализ. – 2008. – № 6. – С. 3–9.
8. *Framework* for Software Product Line Practice, v. 5 – http://www.sei.cmu.edu/ productlines/index.html
9. *Лаврищева Е.М.* Основы технологической подготовки разработки прикладных программ СОД. – Киев: Знання, 1987. – 30 с.
10. *Лаврищева Е.М.* Становление и развитие модульно-компонентной инженерии программирования в Украине. – К.: ИК им. В.М. Глушкова, 2008. – 33 с.
11. *Основы* инженерии качества программных систем / Ф.И. Андон, Г.И. Коваль, Е.М. Лаврищева и др. – К.: Академпериодика. – 2007. – 672 с.
12. *Лавріщева К.М.* Програмна інженерія: Підручник. – К.: Академперіодика, 2008. – 319 с.
13. М*атериалы* и тезисы пленарных докладов // II Всесоюз. конф. «Технология программирования». – Киев, Ин-т кибернетики АН УССР, 1986.
14. *Глушков В.М.* Фундаментальные основы и технология программирования // Программирование. – 1980. – № 2. – С. 3–13.
15. *Капитонова Ю.В.*, *Летичевский А.А.* Парадигмы и идеи академика В.М. Глушкова. – К.: Наук. думка. – 2003. – 355 с.
16. *Лаврищева Е.М.*, *Грищенко В.Н.* Сборочное программирование. – К.: Наук. думка, 1991. – 213 с.
17. *Лаврищева Е.М.* Сборочное программирование. Теория и практика // Кибернетика и системный анализ. – 2009. – № 6. –С. 3–12.
18. *Лаврищева Е.М.*, *Грищенко В.Н.* Сборочное программирование. Основы индустрии программных продуктов. – К.: Наук. думка, 2009. – 371 с.
19. *Грищенко В.М.* Метод об'єктно-компонентного проектування програмних систем // Проблеми програмування. – 2007. – № 2. – С. 113–125.
20. *Бей И.* Взаимодействие разноязыковых программ. – М.–СПб–К.: Диалектика, Вильямс, 2005. – 868 с.
21. *Гринфильд Дж.* Фабрики разработки программ. – М. – СПб–К.: Диалектика, Вильямс, 2007. – 591 с.
22. *Lenz G.*, *Wienands C.* Practical Software Factories in .NET. – From theory to Practice – A Primer Reference and Case Study. – Munih: Apress, 2007. – 205 с.
23. *Чарнецки К.*, *Айзенекер У.* Порождающее программирование. Методы, инструменты, применение. – СПб: Питер, 2005. – 730 с.
24. *Лавріщева К.М.* Генерувальне програмування програмних систем і сімейств // Проблеми програмування. – 2009. – № 1. – С. 3–16.
25. *Лавріщева К.М.* Моделі взаємодії програм, систем і операційних середовищ // Там же. – 2011. – № 3. – С. 13–24.
26. *Островский А.В.* Подход к обеспечению взаимодействия программных сред *JAVA* и *Ms.Net* // Там же. – № 2. – С. 37–44.
27. *Радецький І.О.* Один з підходів до забезпечення взаємодії середовищ *MS.Net* і *Eclipse* // Проблеми програмування. – 2011. – № 2. – С. 45–52.
28. *Лаврищева Е.М.* Теория и практика фабрик программных продуктов // Кибернетика и системный анализ. – 2011. – № 6. – С. 145–158.
29. *Аронов А.О.*, *Дзюбенко А.І.* Підхід до створення студентської фабрики программ // Проблеми програмування. – 2011. – № 3. – С. 42–49.
30. *Коваль Г.И.*, *Коротун Т.М.*, *Лаврищева Е.М.* Об одном подходе к решению проблемы межмодульного и технологического интерфейсов // Диалоговые системы. – 1988. – С. 87–99.
31. *Лаврищева Е.М.* Интерфейс в программировании // Проблеми програмування. – 2007. – № 2. – С. 126–139.
32. *Лаврищева Е.М.* Проблема интероперабельности разнородных объектов, компонентов и систем. Подходы к ее решению // Матеріали 7 міжнар. конф. з програмування УКРПрог'2010. – С. 28–41.
33. *Лаврищева Е.М.* Формальные основы интероперабельности компонентов в программировании // Кибернетика и системный анализ. – 2010. – № 4. – С. 134–150.
34. *Теоретичні* аспекти керування варіабельністю в сімействах програмних систем / К.М. Лавріщева, О.О. Слабоспицька, Г.І. Коваль та ін. // Вісн. КНУ, Серія фіз.-мат. наук. – 2011. – № 1. – С. 151–158.
35. *Нові* теоретичні засади технології виробництва сімейств програмних систем у контексті ГП / К.М. Лавріщева, Г.І. Коваль, Л.П. Бабенко та ін. – Електронна монографія, ДРНТІ. – № 67 – УК – 2011 від 5.10.11. – 377 с.
36. *Лавріщева К.М.* Теоретичні та прикладні аспекти побудови програмних систем (*TAAPSD*'2010), 4–8 жовт., 2010. – С. 274–285.
37. *Коваль Г.І.* Підхід до моделювання якості сімейств програмних систем // Проблеми програмування. – 2009. – № 4. – С. 49–58.
38. *Лаврищева Е.М.*, *Слабоспицкая О.А.* Подход к экспертному оцениванию в программной инженерии // Кибернетика и системный анализ. – 2009. – № 4. – С. 151–168.
39. *Лавріщева К.М.* Інструментально-технологічний комплекс для розробки и навчання прийомам виробництва програмних систем // Вісн. НАН України. – 2012. – № 3. – С. 17–26.
40. *Бабенко Л.П.* Проблемы повторного использования в программной инженерии // Кибернетика и системный анализ. – 1999. – № 2. – С. 151–160.
41. *Грищенко В.М.*, *Куцаченко Л.І.* Автоматизована ін.-формаційна система підтримки міжнародної діяльності НАНУ. – Державний департамент інтелектуальної власності. – Свідоцтво № 32304 від 23.12.2009.

Поступила 18.03.2012
Тел. для справок: +380 44 526-3470 (Киев)