УДК 681.03.06

Е.М. ЛАВРИЩЕВА

РАЗВИТИЕ ОТЕЧЕСТВЕННОЙ ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

Аннотация. Рассмотрена эволюция развития технологий компьютерных систем по пути автоматизации и создания сборочных линий. Приведены основные положения, объекты технологии программирования и сборки сложных систем из готовых ресурсов (модулей, объектов, компонентов и сервисов), которые конфигурируются в вариантные структуры ПС по соответствующим линиям на фабриках программ.

Ключевые слова: программирование, технология, модуль, объект, компонент, готовые ресурсы, инженерия систем, сборочный конвейер, сборочное программирование, методология линий, фабрика программ.

ВВЕДЕНИЕ

Академик В.М. Глушков основал и сформулировал общие положения технологии ЭВМ, систем и программ в 60-х годах прошлого столетия, когда началось построения семейства отечественных ЭВМ и систем. Ученый предвосхитил разработку гибких компьютерных технологий, обеспечивающих производство компьютерных систем по типу конвейерной сборки, проверенной на автомобильных заводах (factory) Форда. По мнению В.М. Глушкова, технология является двигателем прогрессивного развития любой науки, в том числе и теории создания ЭВМ, информационных систем, АСУ, АСУТП и др. [1].

Идею технологии конвейерной сборки В.М. Глушков изложил на научном семинаре Института кибернетики АН УССР (1974 г.), представив ее как набор технологических линий производства компьютерных продуктов и прикладных систем. На основе этой концепции строились новые средства вычислительной техники — большие и малые высокопроизводительные вычислительные системы с макроконвейерной и многопроцессорной организацией ЭВМ («Украина», «Днепр», «Маяк» и др.), а также информационные системы типа АИС «Юпитер» в Институте кибернетики АН УССР [2, 3].

Идея конвейерной сборки оказалась настолько продуктивной, что и сегодня компьютерные сборочные технологии обеспечивают выпуск различных моделей компьютеров, которые широко используются 70 % населения Земного шара. На этих компьютерах решается множество бизнес-задач, реализуются многочисленные информационные услуги с помощью Интернета и др. Однако для решения сложных научно-технических задач современные технологии разработки соответствующих информационных и программных систем не достаточно совершенны. Их еще предстоит улучшать и развивать, чтобы поднять уровень производства.

Технология программирования (ТП) сложных систем создавалась в течение многих лет научными коллективами академических институтов многих республик СССР.

В данной статье рассмотрены пути развития, основные положения и объекты ТП, а также технология сборки сложных систем из готовых разнородных программных элементов, называемых компонентами повторного использования (КПИ) или reuses.

Для первых ЭВМ программы писались на языках операторных граф-схем, адресном, АЛГОЛ-60, ФОРТРАН, КОБОЛ и др. Их преобразование в коды ЭВМ осуществлялось с помощью программирующих программ (ПрП) или систем программирования (трансляторов, интерпретаторов). Они разрабатывались для ЭВМ, которые создавались в СССР, в том числе и в Украине. Далее представлены этапы становления ТП.

Программирование. Всевозможные аспекты программирования при написании программ для решения различных математических задач на ЭВМ описаны в монографиях «Быстродействующие электронные машины АН СССР» (С.А. Лебедев, 1952 г.) ,«Элементы программирования» (А.И Китов, И.А.Криницкий, 1956 г.), «Начальные сведения о решении задач для ЭВМ» (А.А. Ляпунов, Г.А. Шестопалов, 1957 г.), «Алгоритмы и машинное решение задач» (В.А. Трахтенгерц, 1957 г.), «Общее описание БЭСМ и методика выполнения операций» (С.А. Лебедев и В.А. Мельников, 1958 г.), «Адресное программирование» (Е.Л. Ющенко, 1962 г.), а также граф-схемное программирование А.А. Ляпунова, А.П. Ершова и др. Авторы этих научных трудов открыли эру алгоритмизации вычислительных задач и их представления машинными языками.

Программирующая программа. На языке операторных схем алгоритм программы представляется схемой, соответствующей управляющему графу программы, и совокупностью спецификаций каждого оператора. Первая программирующая программа (ПрП-1) разработана в 1954 г. Э.З. Любимским, А.П. Ершовым в МГУ под руководством М.Р. Шура-Бура. На ее основе проектировались другие ПрП на машинах БЭСМ, «Стрела», М-20 и др. В результате создавались фундаментальные алгоритмы трансляции и теория программирования.

В.М. Глушков в статье «Об одном методе автоматизации программирования» и А.А. Стогний в статье «О принципах построения специализированной программирующей программы» (Программирование, № 1, 1958 г.) обосновали библиотечный метод ПрП на примере алгоритма решения систем дифференциальных уравнений. В Украине ПрП реализовались для отечественных машин на основе адресного языка Е.Л. Ющенко и специализированных языков — автокодов. Адресный язык подобен языку математики: алгоритмы на нем записывались набором операторов из одной или нескольких формул. Отличительной особенностью этого языка являлся указатель второго ранга для многократного перехода. Средствами адресного языка описывались задачи вычислительного характера и ПрП для машин «Киев», «Урал», «Днепр» и др.

Системы программирования. В 1960 г. появились новые универсальные языки программирования (ЯП): АЛГОЛ-60, ФОРТРАН, КОБОЛ, ПРОЛОГ, АДА и др.). Одним из первых был АЛГОЛ-60. Во многих институтах АН СССР начали разрабатывать трансляторы с языка АЛГОЛ: ТА-1 — С.С. Лавров (ЛГУ, 1962 г.), ТА-2 — М.Р. Шура-Бура и Э.З. Любимский (ИПМ, 1963 г.), ТА-3 — (АЛЬФА-система) в русской версии языка АЛГОЛ-60 – А.П. Ершов и др. (СО АН СССР, 1964 г.), ТА-4 — для УВК «Днепр-2» Е.Л. Ющенко, Е.М. Лаврищева (ИК АН СССР, 1965 г.).

В ТА-1 разработана быстрая и простая схема трансляции, стековый подход к программированию выражений, реализации процедур без их типизации и рекурсивности. В ТА-2 разработан оригинальный алгоритм программирования процедур, механизмы управления памятью (оперативной и внешней) и метод таблично-управляемой генерации кода. В ТА-3 реализована оптимизация (выражений, циклов, процедур, памяти и др.) для повышения эффективности выходного кода, подобного коду программ, созданного вручную, а также операции над

многомерными значениями и комплексными типами данных. В ТА-4 реализован СМ-метод табличного представления базовых символов языка и соответствующих семантических программ их реализации. Новым подходом в реализации системы программирования «Днепр-2» явилось создание общего арифметического блока для двух языков: АЛГОЛ и КОБОЛ.

Для новых ЭВМ ТА-1–ТА-3 и ОС реализовались в машинном коде этих машин, а ТА-4 — на автокоде машины «Днепр-2», который входил в общесистемное программное обеспечение.

Для усовершенствования разработки трансляторов Э.З. Любимский (1963 г.) предложил промежуточный язык АЛМО для перевода в него любого ЯП, а затем в код ЭВМ. Язык АЛМО — это некоторая абстрактная машина, отражающая особенности класса ЭВМ в СССР. Этот и другие языки (ЭПСИЛОН, СИГМА) стали языками-посредниками при трансляции программ с различных ЯП. С их помощью осуществлялся перевод с m входных языков в n машинные языки, т.е. «из m в один» и «из одного в n». Язык АЛМО был реализован для лучших отечественных машин того времени (М-20, БЭСМ-6, «Минск-2», «Урал-1») на примере трансляторов с АЛГОЛ-60 и ФОРТРАН.

Теория программирования. Конструирование алгоритмов программ вначале проводилось с помощью операторных схем Ляпунова, Янова, структурных схем и блок-схем. Эти схемы программ переводились в один из ЯП для реализации в ПрП. Одновременно создавались новые методы и средства программирования.

Алгебраические языки. При формальном программировании использовались теории автоматов, алгоритмов, математики (алгебры и теории множеств), а также теория операторных схем, систем алгоритмических алгебр (САА) и методы реализации семантики ЯП в трансляторах для различных ЭВМ. В результате появились теоретические аспекты программирования с позиций алгебры, теории множеств, теории языков и метаязыков, а также теория САА, контекстно-свободных ЯП, семантического описания ЯП и др., что отображено в первой монографии В.М. Глушкова [4], которая была переведена на английский язык. Под руководством Е.Л. Ющенко разработан СМ-метод представления семантики ЯП в трансляторах с языков АЛГОЛ-60 и КОБОЛ для машины «Днепр-2», а также схемный алгоритм синтаксического анализа ЯП в РТК-комплексе.

Язык УТОПИСТ ПРИЗ. Этот язык разработал Э.Х. Тыугу для синтеза программ в ЯП на основе описания семантической модели предметной области решения математических задач в пакетах прикладных программ (ППП). Синтез семантических программ в PL/1, ФОРТРАН, АССЕМБЛЕР реализован методом подстановки прикладным задачам семантики их реализации в ЯП и операциями системы ПРИЗ в ОС ЕС.

Графический Р-язык. Данный язык предназначался для конструирования структуры произвольных программ и их схемной реализации Р-графами. Был создан отечественный и международный стандарты ISO/IEC для описания Р-графов и устройство синтаксического контроля программ в нескольких патентах [5]. Структурная интерпретация синтаксиса и семантики ЯП выполнялась с использованием этого языка. Графический способ программирования применялся во многих организациях военно-промышленного комплекса СССР.

Объекты ТП. Ключевым достижением программирования явились библиотеки стандартных подпрограмм вычислительной математики. Методы программирования в разных ЯП основывались на аппарате повторного использования подпрограмм из библиотек программ. Их вызов, а также функции и процедуры (например, subroutine в ФОРТРАНЕ) осуществлялись операторами типа CALL

с указанием их имен и списка фактических параметров. При разработке сложных систем первоначально использовались модули и отдельные программы.

Модуль. Логически законченным фрагментом программы на любом ЯП, выполняющим определенную функцию и имеющим свойства завершенности, раздельной трансляции и повторного использования, является модуль. Для него разрабатывался паспорт — информационный блок, содержащий описание входных и выходных данных, а также операторов вызова других модулей. Они активно использовались при сборке разноязыковых модулей.

Интерфейс. Ключевым понятием технологии разработки сложных прикладных систем из разнородных и разноязыковых модулей стал интерфейс — модуль-посредник между двумя объединяемыми модулями [6, 7]. В нем описывались данные и операторы их передачи от одного модуля другому, а также полученный результат. Интерфейс реализован в системе АПРОП с помощью библиотеки интерфейсных функций преобразования отличающихся типов данных, которыми обмениваются модули между собой. Позднее для описания интерфейса был создан язык IDL (Interface Definition Language) и брокер объектных запросов в системе CORBA в рамках объектного подхода. Описание интерфейса посредниками stub и skeleton упростило задачу сборки разноязыковых модулей.

Функции и композиции. Другим формальным направлением объединения функций цепочками «данные–функция–имя», «функции–композиция–дескрипция» является композиционное программирование [8]. Программная поддержка этих цепочек — система ДЕФИПС, обеспечивающая построение программ из функций, заданных на некотором множестве именованных данных, дескрипций и денотатов (значений). Семантика задается ординарными функциями обработки операций, интерфейсными функциями, арными функциями, а также именованными данными. Операции композиции — это подкласс стандартных композиций и декомпозиционных функций.

Компоненты повторного использования. Наряду с разработкой подпрограмм и библиотечных программ осуществлялся сбор готовых к использованию программ и алгоритмов в Государственный и республиканский фонды алгоритмов и программ. В то время за рубежом проводились аналогичные мероприятия — reusebility.

Прикладное программирование. Многие институты АН СССР и предприятия страны занимались реализацией научных и прикладных задач.

Создание АСУ, АСУТП. Для автоматизации предприятий различного уровня В.М. Глушков предложил теорию АСУ и АСУТП [2], которую он реализовал, принимая участие в создании первых АСУ в Украине (для Лисичанского химкомбината, Донецкого горно-обогатительного комбината, Львовского телевизионного завода), Болгарии и ГДР (для металлургического комбината). Автор настоящей статьи в составе группы ученых принимал участие в разработке АСУТП проката металла на металлургических комбинатах Берлина и Лейпцига (ГДР) на базе УВК «Днепр-2» в рамках межгосударственного проекта Украина–ГДР (1971–1975 гг.). Принципы и структуру этого проекта разработал В.М. Глушков.

Пакеты прикладных программ. Пакет программ — система управляющих, прикладных и обслуживающих модулей с языком пользователя для постановки задания для решения задачи из заданного класса. Сформировалась общая структура, сущность и принципы построения ППП. По данной технологии было разработано более 20 ППП для решений задач обработки данных на транспорте, в экономике, статистике, математических вычислениях, системной оптимизации и др. В результате создания различных ППП сформирована индустриальная технология построения ППП под руководством академика И.В. Сергиенко, описанная

в монографии «Прикладные программные системы» (В.Н. Редько, И.В. Сергиенко, А.С. Стукало, 1992 г.).

В развитие отечественной технологии программирования весомый вклад внесли ведущие специалисты академических институтов АН СССР, а именно В.М. Глушков, И.В. Вельбицкий, А.П. Ершов, Э.З. Любимский, В.С. Лавров, М.Р. Шура-Бура, Е.Л. Ющенко, Э.Х. Тыугу, И.В. Сергиенко, которые непосредственно создавали системы программирования, прикладные системы, АСУ и АСУТП. На конференциях по ТП обсуждались полученные новые теоретические и прикладные аспекты проектирования, разработки, тестирования различных видов систем (прикладных, информационных) для больших ЭВМ, а также задачи эксплуатации и сопровождения ПО.

Разработку средств автоматизации ТП финансировал ГКНТ СССР. Выходили специальные постановления Совета Министров СССР, направленные на развитие средств вычислительной техники и технологии изготовления программных продуктов (ПП) и сдачи их в фонды алгоритмов и программ для использования другими организациями.

Автоматизация программирования. Индустриальный выпуск сложных программ В.М. Глушков предлагал реализовать с помощью сборочного конвейера по типу линий на заводах Форда. По его мнению, фабрика производства программ должна оборудоваться линиями программирования различных программ, объединяющими их в новую агрегатную структуру. Автоматизацией разных аспектов разработки прикладных систем с использованием программ, которые сдавались в Фонды алгоритмов и программ, занимались многие специалисты институтов Советского Союза, в том числе ИК АН УССР. Данной проблематикой автор настоящей статьи начал заниматься с 1965 г., когда по инициативе В.М. Глушкова группу разработчиков трансляторов с языков АВТОКОД, АЛГОЛ-60, КОБОЛ из отдела Е.Л. Ющенко ИК АН УССР перевели в СКБ ММС ИК АН УССР для построения трансляторов для новых ЭВМ, а именно «Мир», УВК «Днепр-1» и «Днепр-2» и др. [9].

Проекты автоматизации прикладных систем и ППП в ИК, финансируемые ГКНТ СССР, включали следующие разработки: формальные технические задания (Ю.В. Капитонова, А.А. Летичевский); система автоматизации программ АПРОП (Е.М. Лаврищева); ППП методов численного анализа (И.Н. Молчанов), ППП ДИСПРО (И.В. Сергиенко, А.С. Стукало) и статистики (И.Н. Парасюк); технологический комплекс программиста ТКП (И.В. Вельбицкий); САА «Мультипроцесист» (Г.Е. Цейтлин); композиционное программирование ДЕФИПС (В.Н. Редько); диалоговые системы обучения ЯП (Е.Л. Ющенко, А.М. Довгялло); система «Макробол» (Л.П. Бабенко), система мультипрограммирования «Терем» (Н.М. Мищенко) и др.

Средства автоматизации разрабатывали также и другие научные центры СССР: система «Приз» (Э.Х. Тыугу, ИК АН ЭССР), системы «Альфа» и «Бета» (А.П. Ершов, НГУ, Новосибирск), система АДА (С.С. Лавров, ЛГУ), система автоматизации математических задач (М.Р. Шура-Бура, ИПМ АН СССР), система модульного программирования (Е.А. Жоголев, МГУ), система автоматизации трансляторов (М.А. Гонца, Институт математики, МССР), система послойного проектирования программных проектов (Фуксман В.И., РГУ), системы телекоммуникации (В. Бардзинь, Лат.ССР) и т.д. Так, в 1982 г. в Калинине был создан программно-строительный завод для производства АСУ.

При выполнении научных проектов в СССР сформировались различные аспекты технологии программирования и индустриальные методы производства прикладных программных систем, АСУ и ППП.

Все новые теории в физике, математике и других естественных науках подтверждались на практике соответствующими технологиями.

В компьютерных науках (Computer science, CS) сформировались аналогичные ТП (компьютеров, систем и программ), а именно Computer Engineering — инженерия (конструирование) компьютеров, Systems Engineering — инженерия (технология) систем и Software Engineering (SE) — инженерия (технология) программного обеспечения (ПО).

Информация о SE впервые появилась на конференции НАТО в 1968 г. Официальное определение SE дал международный комитет специалистов ACM и IEEE в SWEBOK (Software Engineering Body Knowledge) в 2001 г. (www.swebok.com). Ядро знаний SWEBOK постоянно совершенствуется в направлении развития его методов и средств для эффективного их применения в производстве, реинженерии и реверсной инженерии ПП.

По сравнению с SWEBOK принятый в СССР термин ТП в отличие от инженерии SE имеет более широкий смысл, поскольку включает теорию, методы, средства и инструменты формального проектирования и реализации прикладных систем и ППП. Приведем краткое определение описанных технологий по материалам энциклопедии Computer science.

Технология компьютеров — это дисциплина реализации теории и принципов построения компьютеров (frameworks, микропроцессов, кластеров, суперкомпьютеров и т.п.), а также системного обеспечения (ОС, трансляторов, компиляторов и т.д.). Основными в данной теории технологии являются теория Тьюринга, Фон Неймана, автоматов, алгоритмов и кибернетики В.М. Глушкова [1–3], а также математика, логика, анализ систем и т.п. Технология компьютеров в настоящее время развита теоретически и практически как в направлении элементной базы (устройства, блоки, микросхемы, карты и т.п.), так и их системной организации (фреймворки, кластеры и др.). Уровень компьютерной технологии в сотни раз выше системной и программной.

Технология систем — это теория, методы и принципы построения информационных и автоматизированных систем, а также систем управления и компьютерных (Computer Systems). Средствами компьютерных технологий моделируются различные типы компьютерных приложений (Computer Applications, Domain Engineering, Family Engineering) и новых средств управления информационными системами (ОС, БД, СУБД и др.). К средствам этой технологии относятся принципы и теория АСУ В.М. Глушкова [3], а также новые интеллектуальные дисциплины (комбинаторика, графика и т.п.) и методы управления экономической, финансовой, банковской деятельностью и т.д.

Программная инженерия (Software Engineering) — это система методов, способов и дисциплин планирования, разработки, эксплуатации и сопровождения ПО, обеспечивающих промышленное производство ПП. Она включает все аспекты создания ПО от начала формулировки требований, разработки продукта до его использования и сопровождения [6, 10–15]. Базисом данной технологии являются теории алгоритмов, программирования, вычислений и распределенной, телекоммуникационной обработки данных в сетях. При массовом производстве ПП используется теория планирования, регулирования процессов и ресурсов, тестирования, измерения результатов, оценки риска и достижения качества [15].

Информационные системы — это компьютерные системы обработки разнообразной информации на предприятиях и бизнес деятельности, включая бухгалтерский учет, расчет заработной платы, документооборот на всех уровнях управления государством и т.п., предназначены для управления и обработки ин-

формации, обеспечения продуктивности и эффективности работы систем информатизации. Например, информационные системы и поисковые системы Интернета являются инструментами накопления, поиска и выбора различных информационных ресурсов, необходимых для их применения пользователями [3].

Информационные технологии с 1990-х годов стали базисом компьютерной инфраструктуры современных корпораций, предприятий и государственных органов управления, на которых решаются различные глобальные задачи обработки информации. На их разработку и подготовку ИТ-специалистов выделяются огромные финансовые ресурсы для поддержки всевозможных информационных систем Интернета и доступа к ним всех желающих. Цели и задачи построения информационных систем и технологий их применения сформулировал академик В.М. Глушков в [3].

Таким образом, зарубежные технологии способствуют ускоренному выпуску компьютеров и систем для потребностей мирового информационного общения.

ТЕХНОЛОГИЯ СБОРОЧНОГО ПРОГРАММИРОВАНИЯ

В 70-х годах прошлого века ИК АН УССР внес значительный вклад в индустрию ПП на больших ЭВМ (ЕС ЭВМ, БЭСМ-6 и др.) Разрабатывались методы, средства и инструментальные системы для создания прикладных систем, АСУ и АСУТП.

Автоматизация различных отраслей промышленности, в том числе систем в оборонной промышленности, проводилась на основе теории АСУ или АСУТП [2, 3] и с использованием готовых программ из Государственных фондов алгоритмов и программ.

В.М. Глушков в работе [9] выделил три основные направления развития ТП:

— модульная система автоматизации сложных программ АПРОП из стандартизованных программных заготовок «снизу-вверх» [6, 7];

— метод формализованных технических заданий для последовательной детализации проекта с использованием различных алгоритмических языков для описания отдельных блоков [16, 17];

— технология программирования для автоматизации изготовления систем управления в оборонной промышленности [18].

В.М. Глушков считал, что развитие технологии комплексного проектирования вычислительных систем, проектирование технических средств объединено в единый процесс с проектированием его базисного математического обеспечения. Эта идея реализована в системе ПРОЕКТ [17], в системном обеспечении ЭВМ МИР-2 для решения инженерных задач, а также в конкретных АСУ и АСУТП.

Для реализации повторного использования готовых программ, поддержки индустрии ПО на государственном уровне были созданы Фонды алгоритмов и программ и вышло постановление ГКНТ СССР о том, что программы имеют статус продуктов производственно-технического назначения.

Повторное использование готовых программ и модулей было затруднено ввиду отсутствия описания интерфейсов в программах Фондов для их применения в АСУ.

Сборочное программирование. В 1975 г. начало развиваться сборочное программирование для обеспечения индустрии сборки информационных и программных систем из готовых модулей средствами системы АПРОП. В рамках этой системы реализован метод сборки больших программ из стандартных модулей и интерфейсов, идентичных стыковочным деталям, используемым в автомобильной промышленности. Интерфейс стал основным элементом сборочного

конвейера [18]. По аналогии с автоматизированными линиями конвейера, на которых собирались отдельные крупные изделия из готовых стандартных деталей с помощью стыковочных «болтов и гаек», в АПРОП «деталями» являлись программные модули, объединяющиеся виртуально с помощью специальных стыковочных элементов, названных интерфейсами. Главными элементами сборочного программирования стали КПИ и интерфейсы [19].

Интерфейсный переходник между двумя модулями выполнял передачу данных от одного модуля другому, а также возвращал полученные результаты. Если тип переданных данных и результат был неэквивалентен исходному, то переходник трансформировал их в соответствующую форму представления данных принимающим модулем. Этот термин укоренился в программировании. Он обсуждался на международной конференции «Интерфейс СЭВ» (1987 г.), на которой представлялись межмодульный, межъязыковый и технологический интерфейсы [6, 7].

Межмодульный интерфейс — это интерфейс между передающими и принимающими модулями, выполняющий функции передачи, приема и преобразования нерелевантных данных. Он важен при обеспечении взаимодействия между разноязыковыми модулями в структуре сложной системы.

Межъязыковый интерфейс. Данное устройство представляет совокупность средств и методов представления и взаимно-однозначного преобразования структур и типов данных ЯП (например, преобразование матрицы по строкам в ФОРТРАНЕ в матрицу по столбцам в PL/1 и обратно, символьных данных в целое и др.).

Технологический интерфейс. Это совокупность методов и средств для взаимосвязи процессов и их операций в жизненном цикле (ЖЦ), реализации сложных программ, включая нормативные, методические документы и формы (каркас технологических линий (ТЛ), формат документов, язык связи процессов и др.). Методика создания ТЛ предложена в 1987 г. [19], апробирована и усовершенствована на шести линиях АИС «Юпитер-470». Дальнейшим развитием ТЛ стали зарубежные продуктовые линии (Product Lines) SEI http://sei.cmu.edu/productlines/frame_report/) 2004 г.

Организатор конференции «Интерфейс СЭВ» ГКНТ СССР наградил коллектив разработчиков системы АПРОП (включая автора статьи) почетной грамотой.

Сборочное программирование обсуждалось на конференциях по ТП. Так, на конференции ТП (1984 г.) академик А.П. Ершов отметил [20], что сборочное программирование решает задачи многократного и быстрого применения в процессе создания программы из заранее изготовленных «деталей». Оно эффективно и позволяет быстро решать любую задачу из некоторого класса проблем. Кроме того, в докладе «Отношение методологии и технологии программирования» на Всесоюзной конференции по ТП (1986 г.) А.П. Ершов сформулировал перспективы промышленной сборочной технологии программирования до 2005 г., включая нормативы производительности и надежности продукта, этапы разработки ПП и межмодульных интерфейсов поддержки этого программирования.

Сборочный конвейер. Основным элементом сборочного конвейера являются ТЛ. Их сущность и методика построения определена автором данной статьи и апробирована в системе АИС «Юпитер-470» для четырех объектов военно-морского флота СССР. Практически были представлены шесть ТЛ для изготовления различных видов программ, необходимых при решении прикладных задач на объектах АИС. Именно эти ТЛ стали первым вариантом представления линий сборочного конвейера фабрики программ [21]. С их помощью создано приблизительно 500 программ обработки данных для объектов АИС.

После распада Советского Союза (1992 г.) методика построения ТЛ развивается. Экспериментальная фабрика программ практически реализована студентами в КНУ имени Тараса Шевченко (2011 г.), как заключительный этап построения сборочного конвейера В.М. Глушкова, приуроченного к его 90-летию (2013 г.).

РАЗВИТИЕ ОБЪЕКТНОЙ ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

Поскольку большие программные системы создавались традиционными методами программирования на основе структурно-функционального подхода и процедурных ЯП, это привело к повышению сложности таких систем. Их тяжело было развивать и изменять ввиду отсутствия в них заранее предусмотренных реперных точек (названных точками вариантности) для корректировки отдельных фрагментов. Требовалось изменять не только процесс разработки, но и использовать появившийся объектно-ориентированный подход (ООП), как шаг к уменьшению сложности программной системы (ПС).

Объектно-ориентированный подход. Данный подход в отличие от традиционного предлагает проектировать ПС с помощью объектов, которые способны самостоятельно работать, собираться и объединяться в более сложные структуры. На основе ООП сложился новый стиль программирования сложных систем из объектов в соответствующих ЯП, библиотек классов, routines и типов данных, а также систем автоматизации (COM, CORBA, DCE RPC и др.). Главное назначение системы CORBA — формальный аппарат определения объектной модели (ОМ) и брокер объектных запросов в ПС, созданных из объектов и элементов библиотек, которые можно пополнять и удалять. Интерфейс разнородных объектов задается на языке IDL в виде stub и skeleton. Брокер реализует взаимодействие разноязыковых объектов, которые описываются в языках C+, JAVA, SMAILTALK, COBOL, ADA и др.

Объектно-компонентный метод. Отдел «Программная инженерия» ИПС НАНУ, начиная с 1992 по 2012 гг., выполнял фундаментальные проекты, финансируемые ГКНТ и НАНУ, по развитию объектно-ориентированного, компонентного, генерирующего программирования [22, 23]. На рис. 1 показан объектно-компонентный метод (ОКМ) с четырьмя уровнями объектного проектирования доменов:

— на обобщенном уровне определяются базовые понятия, объекты функции (O_i) домена без учета их сущности и свойств, а также объекты данных (I), необходимых для работы домена;

— на структурном уровне строятся графы функций и дан-

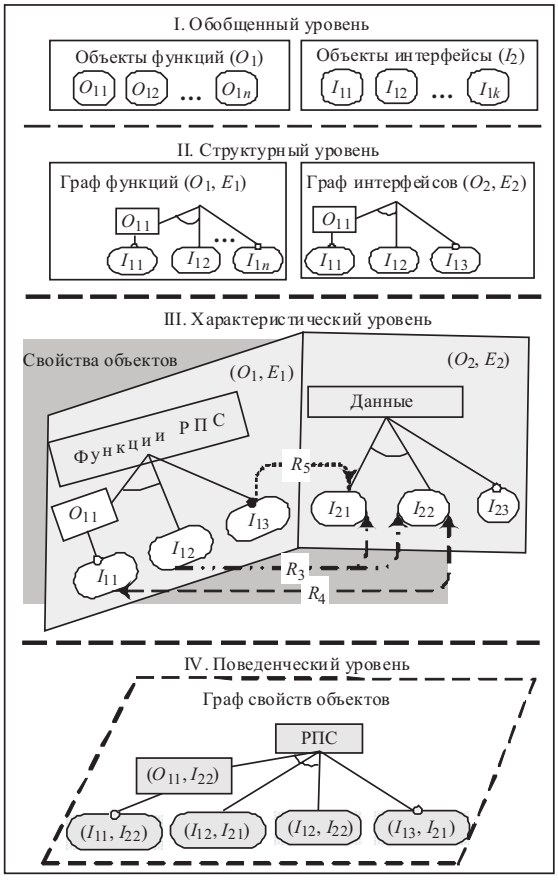

Рис. 1. Уровни проектирования домена

ных объектов, а также устанавливаются их связи и отношения, графы отображаются в ОМ, в которой дается описание объектов и взаимоотношения между ними;

— характеристический уровень предназначен для задания свойств объектов (внешних и внутренних), которые указывают на принадлежность к некоторому классу;

— поведенческий уровень служит для описания поведения объектов, которое зависит от выполнения действий над объектами и вызываемыми ими событиями.

В ОКМ объекты-функции (методы) объединяются в классы и подклассы в соответствии с их внешними и внутренними характеристиками. Объекты наследуют свойства, функции и интерфейс других объектов, полиморфных между собой через интерфейсы. Разработан формальный механизм перехода от объектов к компонентам и программным интерфейсам [21, 23, 24].

Для обеспечения изменяемости и адаптивности структуры сложных ПС, созданных из готовых программных ресурсов (модулей, объектов, компонентов и сервисов), данный ОКМ пополнен моделями взаимодействия, вариабельности ПС и их семейств [25].

Инженерия качества ПС. В рамках фундаментальных проектов ИПС НАН Украины (1998–2012 гг.) создана оригинальная теория и методология моделирования предметных областей из объектов [23]. Она включает вопросы инженерии приложений, основанной на процессах ЖЦ, структурной, поведенческой и функциональной моделях и подходах к обеспечению качества систем, начиная с ранних процессов проектирования элементов ПС и завершая их тестированием.

Инженерия качества основана на результатах верификации и тестирования. В [23] разработана концептуальная модель процесса тестирования семейства ПС из готовых ресурсов $SFT = \langle TM; TD, TA, Env \rangle$, где TM — подпроцесс управления тестированием; TD и TA — подпроцессы тестирования предметной области и приложений соответственно; Env — концептуальная и информационная среда процесса тестирования СПС. При этом все подпроцессы имеют унифицированное формальное представление

$$TM = \langle Task(TM, TD, TA), En(TM), CM(TM) \rangle,$$

$$En(TM) \cup En(TD) \cup En(TA) = Env,$$

где $Task$ — задачи, разрешимые при выполнении соответствующего подпроцесса; En — концептуальная и информационная среда и CM — подмодель координации операций.

Состав концептуальной среды Env определяется выражением

$$Env = TG \cup SG \cup T \cup P \cup RG \cup RP,$$

где TG и SG — тесты активов и программных КПИ соответственно; T и P — множество тестов и тестированных приложений соответственно; RG и RP — отчеты о выполнении тестовых КПИ и тестов соответственно.

Согласно этой модели формируются данные об интенсивности ошибок для организации оценки надежности в модели качества ПС.

Важным вкладом в индустрию ПП явилась разработка основ качества ПС, которая проводилась в рамках СЭВ под руководством ГКНТ. Первый совместный проект по качеству обсуждался в Дрездене (1987 г.). После распада СССР проблематика качества решалась в проектах ГКНТ Украины (1992–1998 гг.) под руководством А.Ф. Кулакова, а также в технологии доказательного программирования и организации проведения оценки качества и сертификации продукта. Созданные модели и методы оценки качества внедрены в рамках проекта инфор-

матизации МО Украины, они включали методики достижения и управления качеством ПС, оценивания стоимости и трудоемкости работ, проведения экспертно-аналитического оценивания процессов и продуктов на всех процессах ЖЦ, тестирования и измерения показателей качества для готового продукта [12].

По проблематике инженерии качества, тестирования, экспертно-аналитического оценивания качественных и эксплуатационных показателей ПС защищены три кандидатские диссертации (2004–2008 гг.).

Новые научные результаты описаны в учебниках по программной инженерии [10–15, 27] и управлению программными проектами [28], а также в монографиях [15, 29]. Учебник [13] на конкурсе Microsoft (2006 г., МГУ) получил призовое место и включен в состав сайта www.intuit.ru. Новый вариант учебника [14] представлен в электронном виде на студенческой фабрике программ http://programsfactoty.univ.kiev.ua.

Результаты теоретических и программных разработок опубликованы в статьях и докладывались на конференциях ICTERI [26, 30, 41]. Они преподаются на нормативных курсах «Программная инженерия» в МФТИ и «Технология программирования информационных систем» в КНУ имени Тараса Шевченко (2001–2013 гг.).

ФАБРИКИ ИНДУСТРИИ ПРОГРАММНЫХ ПРОДУКТОВ

Последнее десятилетие характеризуется дальнейшим усовершенствованием объектов и линий сборки на фабриках программ. Анализ показал, что технология сборки [31] и принципы взаимодействия компонентов [32] представлены в зарубежных работах мультитехнология К. Чернецкого и К. Айзенекера [33] с лейтмотивом «от ручного труда к конвейерной сборке», технология И. Бея с автоматизированными приемами и способами взаимодействия разноязыковых программ [33], поточная сборка с use case UML фабрики Дж. Гринфильда [34] и Г. Ленца в. NET [35], сборочный конвейер М. Фаулера, ЕПАМ [36] и др.

Общим в проанализированных фабриках программ [37] являются автоматизированные линии сборки различных видов программ для массового использования, линии AppFab в индустриальных фирмах производства продуктов (IBM, MS.Net, Sun Microsystems, Oberon, Unix, Intel и др.) и Software Product Lines (http://www.sei.cmu.edu/productlines/).

Фабрики предоставляют набор ресурсов, средств и инструментов для автоматизации производства ПП, увеличения производительности их разработки на каждом процессе ЖЦ и сокращения объемов за счет использования готовых ресурсов (модулей, объектов, компонентов, сервисов). Некоторые фабрики программ используют новые подходы (UML, WorkFlow, DSL и др.), а также сервисно-компонентную IBM-технологию по разработке простых и сложных ПП.

Исходя из опыта автоматизированной сборки разнородных программ в ЯП и анализа современных зарубежных фабрик программ индустриального типа (IBM, OMG, Microsoft, Oberon и т.п.), сформировался общий набор элементов, характеризующих любую фабрику программы из готовых программных ресурсов:

— спецификация паспортных данных готовых программных ресурсов в одном из языков интерфейса (IDL, API, SIDL, WSDL, RAS и т.п.);

— операционная среда, содержащая системные средства и инструменты для системной сборки разнородных ресурсов (например, http://nant.sourceforge.net/);

— ТЛ Product Lines изготовления семейств ПС из готовых продуктов;

— метод разработки КПИ и их сборки с помощью интерфейсов на зыках UML, DSL и др.;

— набор конкретных линий производства программ.

Таким образом, технология сборочного программирования расширена новы-

ми видами ресурсов, накоплено большое количество программ в Интернете, в том числе в индустрии научного софтвера Европейского проекта Grid, Cloud Computing и др. Сформированы сотни фабрик программ различного назначения. Среди них описанные ранее фабрики и созданная студенческая фабрика программ в КНУ имени Тараса Шевченко на веб-сайте http://programsfactory.univ.kiev.ua [30, 38].

Важный результат этого периода развития ТП — формирование теоретических и практических аспектов компонентного программирования, теория анализа предметной области из объектов, создание из них объектной модели с использованием формальных математических операций определения отдельных элементов этой модели. Разработан метод трансформации объектной модели в изоморфную компонентную модель с компонентами и интерфейсами [39, 40]. Это дало возможность использовать механизмы сбора компонентов по ТЛ и тем самым обобщить сборочное программирование и на процессы технологии. Определена компонентная алгебра и операции внесения изменений в сложную структуру ПС и СПС [41].

Методология построения ТЛ. В настоящее временя появились новые варианты метода построения ТЛ на этапе технологической подготовки работ (ТПР) [20, 40] и продуктовых линий (Product Lines) (www.sei. com.edu). Согласно ТПР линию можно представить в виде схемы (маршрута) из процессов и операций (рис. 2).

Каждой операции схемы ставятся в соответствие элементы, которые обеспечивают реализацию КПИ или будущей ПС средствами ЯП, инструментами и стандартными технологическими модулями (ТМ) поддержки реализации специфики предметной области (ПрО) с помощью комплекса нормативно-методического обеспечения.

Все ресурсы связываются технологическим маршрутом, упорядочивающим процессы и операции ТЛ по заполнению каркасов проектных решений, их реализации и изменению элементов ПП. Конечной операцией маршрута является операция оценки качества продукта. Набор процессов ТЛ создается с учетом требований международного стандарта ISO /IEC ЖЦ 12207–2007. Выбираются методы, средства и инструменты преобразования состояний элементов на операциях маршрута для получения промежуточного элемента. Маршрут ТЛ описывается специальным языком BDHN с использованием специальных форм описания последовательной трансформации состояний элементов и данных, передаваемых между ТМ.

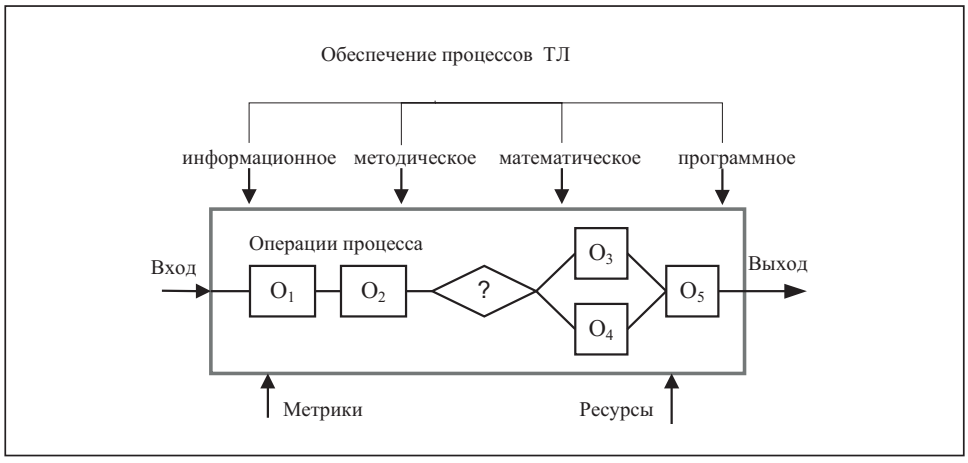

Рис. 2. Схема ТЛ

Методология построения продуктовой линии. Данная линия включает product line (линию продуктов) и product family (семейство ПП). Эти термины определены в словаре ISO/IEC FDIS 24765:2009 (E) “Systems and Software Engineering Vocabulary” как «группа продуктов или услуг, имеющих общее управляемое множество свойств ПП–СПП, которые удовлетворяют потребностям определенного сегмента рынка или виду деятельности».

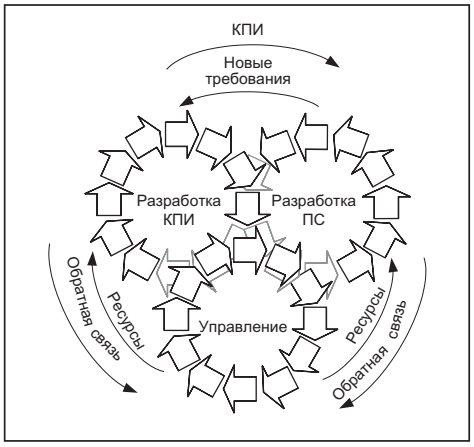

Рис. 3. Модель инженерии продукта

Основа построения данной продуктовой линии — инженерная и процессная модели. Инженерная модель предназначена для планирования производства ПС с учетом предметной области их применения, ограничений и стратегии производства продукта их готовых КПИ (рис. 3).

Разработка метода сборки продукта и семейства ПП обеспечивает построение плана реализации каждого отдельного программного компонента из множества разработанных программных ресурсов.

Управление разработкой СПП из готовых КПИ состоит в координации деятельности по выпуску необходимых продуктов и решения организационного и технического управления каждым членом семейства СПП.

Процессная модель включает множество процессов, выполняемых на двух уровнях: инженерии ПрО, обеспечивающая повторное использование (for reuse), и инженерии приложений Engineering Applications (или инженерии ПС) как разработки с использованием КПИ (with reuse).

В целом описанные линии разработки и сборки повышают производительность исполнителей, улучшают условия их работы, сокращают число сборщиков, повышают качество и снижают себестоимость выпуска продукта. Эти линии реализованы на фабрике программ, как образец сборочного конвейера академика В.М. Глушков по разработке и сборке КПИ, а также для обучения дисциплинам программной инженерии и программированию в разных ЯП.

Фабрики программ. Впервые в Украине фабрика программ была построена согласно концепции сборочного конвейера В.М. Глушкова в КНУ на факультете кибернетики [30, 38]. Фабрика создавалась для обучения всем аспектам программной инженерии, информационных систем и построения студентами программных артефактов в рамках лабораторных и дипломных работ. Артефакты и программы специфицируются в стандартном языке WSDL и сохранятся в репозитории программ и интерфейсов. Фабрика оборудована следующими ТЛ: программирование в языках C# VS.Net, JAVA, DSL; сборка программ в сложные структуры; е-обучение основам программной инженерии в среде в КНУ MS.Net и открытых систем Eclipse-DSL, Protege и т.п. [38]. Готовые КПИ и артефакты в стандартизованном виде запоминаются в репозитории, который может пополняться и другими продуктами.

Фабрику программ разработали студенты А. Аронов, А. Дзюбенко и А. Островский факультета кибернетики КНУ под руководством автора статьи. Фабрика работает в Интернете с 2012 г. (http://programsfactoty.univ.kiev.ua). Она входит в состав ИТК [38], как инструмент электронного обучения дисциплине «Программная инженерия» общего курса электронного обучения на факультете кибернетики КНУ.

На сайте фабрики использована лицензионная версия VS.Net в качестве среды его функционирования с использованием многоязычной разработки ПС и средств коллективного изготовления ПП. На фабрике имеются следующие ТЛ:

— программирование в C# VS.Net с получением готового продукта и сохранением его стандартного вида в репозитории;

— занесение и подбор готовых артефактов и программ из репозитория для дальнейшего использования;

— сборка готовых программ в сложные программные структуры систем;

— обучение методологии проектирования ПС и семейств систем по е-учебнику «Программная инженерия» и учебнику (www.intuit.ru) [13].

В ИТК реализованы средства:

— генерация ПС, описанных в языке DSL с использованием инструментальных систем Eclipse — DSL, Tool DSL Microsofts;

— трансформация GDT $\Leftrightarrow$ FDT общих ТД к фундаментальным ТД на основе средств генерации данных стандарта ISO/IEC 11404–2007;

— онтологическая модель стандарта ЖЦ ISO/IEC 12207 [26, 42];

— взаимодействие распределенных систем VS.NET-JAVA, Eclipse-VS.Net и др.

На ИТК (http://sestudy.edu-ua.net) размещены научные статьи разработчиков по фундаментальным основам индустрии ПП, е-монография [29] и доклады на международных конференциях [26]. Сайт ориентирован на обучение студентов фундаментальным основам курсов ТП и SE в других университетах Украины, стран СНГ, а также в англоязычных странах по специальностям: информатика, программная инженерия и Computer Sciences. Оба сайта многоязычные (укр., рус., англ.), уникальны и оригинальны, ими пользуются более 15000 респондентов.

ЗАКЛЮЧЕНИЕ

Приведены основные пути постепенного формирования отечественной технологии сборочного программирования сложных систем из готовых КПИ и реализации сборочного конвейера. Главными составляющими ТП являются:

— основные понятия и задачи программирования (языки, трансляторы, системы автоматизации ППП, АСУ и др.), а также технологии программирования, изготовления АСУТП для металлургического комбината ГДР с помощью созданной операционной системы программирования «Днепр-2»;

— система АПРОП для сборки готовых разноязыковых модулей, интерфейсов и библиотеки функций преобразования нерелевантных типов данных;

— принципы построения линий сборочного конвейера как главного способа индустриального изготовления различных типов ПС и СПС, которые опередили на 15 лет появление продуктовых линий;

— новый стиль сборочного программирования сложных программных и информационных систем из более простых и стандартизованных КПИ (модулей, объектов, компонентов, сервисов);

— объектно-компонентный метод моделирования объектных структур, трансформации объектов в программные компоненты, конфигурирования их в вариантные структуры ПС семейства СПС;

— фабрика программ КНУ как веб-сайт Интернета для предоставления и обмена научными артефактами, создаваемыми студентами по изучаемым предметам и компьтерным дисциплинам;

— электронное обучение студентов на языках C#, VS.Net, JAVA и предмету «Программная инженерия» на фабрике программ.

СПИСОК ЛИТЕРАТУРЫ

1. Капитонова Ю.В., Летичевский А.А. Парадигмы и идеи академика В.М. Глушкова. — Киев: Наук. думка, 2003. — 355 с.
2. Глушков В.М. Кибернетика, ВТ, информатика (АСУ). — Избр. труды в 3-х томах. — К.: Наук. думка, 1990. — Т. 1. — 262 с.; Т. 2. — 267 с.; Т. 3. — 281 с.
3. Глушков В.М. Основы безбумажной информатики. — М.: Наука, 1982. — 552 с.
4. Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л. Алгебра. Языки. Программирование. — К.: Наук. думка, 1974. — 318 с.
5. Лаврищева Е.М., Усенко Л.Г., Яффе В.А. Транслятор Д-АЛГАМС для машины «Днепр-2» // Труды научн.-техн. конф. СКБ ММС ИК АН УССР, Киев, 1970. — С. 31–42.
6. Система автоматизации производства программ (АПРОП) / В.М. Глушков, Е.М. Лаврищева, А.А. Стогний и др. — Киев: Ин-т кибернетики АН УССР, 1976. — 134 с.
7. Лаврищева Е.М., Грищенко В.Н. Связь разноязыковых модулей в ОС ЕС. — М.: Финансы и статистика, 1982. — 127 с.
8. Редько В.Н. Композиции программ и композиционное программирование // Программирование. — 1978. — № 5. — С. 17–26.
9. Глушков В.М. Фундаментальные основы и технология программирования // Там же. — 1980. — № 2. — С. 3–13.
10. Андон Ф.И., Лаврищева Е.М. Методы инженерии распределенных компьютерных приложений. — Киев: Наук. думка, 1997. — 328 с.
11. Бабенко Л.П., Лаврищева Е.М. Основы программной инженерии. — К.: Знання, 2001. — 269 с.
12. Lavrischeva E. Generative and composition programming: aspects of developing software system families // Cybernetics and Systems Analysis. — 2013. — **49**, N 1. — P. 110–123.
13. Лаврiщева К.М. Програмна інженерія. — К.: Академперіодика, 2008. — 319 с.
14. Основы инженерии качества программных систем / Ф.И. Андон, Г.И. Коваль, Т.М. Коротун и др. — К.: Академпериодика, 2007. — 672 с.
15. Задорожна Н.Т., Лаврiщева К.М. Менеджмент документообігу в інформаційних системах освіти. — К.: Педаг. думка, 2007. — 224 с.
16. Глушков В.М., Капитонова Ю.В., Летичевский А.А. О применении метода формализованных технических заданий к проектированию программ обработки структур данных // Программирование. — 1978. — № 6. — С. 31–40.
17. Вельбицкий И.В., Ходаковский В.Н., Шолмов Л.И. Технологический комплекс автоматизации программ на машинах ЕС ЭВМ и БЭСМ-6. — М.: Финансы и статистика, 1980. — 253 с.
18. Lavrishcheva E.M. Modular design of large programs // 1980. — **16**, N 2. — P. 244–249.
19. Lavrischeva K.M. Compositional programming: theory and practice // Cybernetics and Systems Analysis. — 2009. — **45**, N 6. — P. 845–853.
20. Лаврищева Е.М. Основы ТПР прикладных программ СОД. — К., 1987. — 30 с.
21. Lavrischeva K.M. Theory and practice of software factories // Cybernetics and Systems Analysis. — 2011. — **47**, N 6. — P. 961–972.
22. Czarnecki K., Eisernecer U. Generative programming. Methods, tools, and applications .— Addison Wesley, 2000. — 730 p.
23. Лаврiщева К.М. Підхід до формального подання онтології життєвого циклу програмних систем // Вісн. Київ. нац. у-ту імені Тараса Шевченка. Сер. фіз.-мат.наук.. — 2013. — № 4. — С. 140–149.
24. Лаврiщева К.М. Компонентне програмування. Теорія і застосування // Проблеми програмування. — 2012. — № 4. — С. 3–19.
25. Лаврищева Е.М., Грищенко В.Н. Сборочное программирование. — К.: Наук. думка, 1991. — 213 с.

26. Lavrischeva E., Ostrovski A., Radetskyi I. Approach to e–learning fundamentals aspects of software engineering // 8th Intern. Conf. ICTERI-2012 “ICT in Education, Research and Industrial Applications”. — 2012 —.

27. Лаврищева Е.М. Проблематика программной инженерии. — Киев: Знание, 1991. — С. 20.

28. Лаврищева Е.М. Методы программирования. Теория, инженерная практика. Киев: Наук. думка, 2006. — 454 с.

29. Андон П.І., Лавріщева К.М. Розвиток фабрик програм в інформаційному світі // Вісник НАН України. — 2010. — № 10. — С. 15–41.

30. Lavrischeva E., Aronov A., Dzubenko A. Programs factory — a conception of knowledge representation of scientifical standpoint of software engineering // J. Comput. Sci. — 2013. — P. 21–27.

31. Лавріщева К.М., Слабоспицька О.О., Коваль Г.І., Колесник А.Л. Теоретичні аспекти керування варіабельністю в сімействах програмних систем // Вісн. Київ. нац. у-ту імені Тараса Шевченка. Сер. фіз.-мат.наук. — 2011. — № 1. — С. 151–158.

32. Greenfield J., Short K. Software factory: assembling applications with patterns, models, frameworks, and tools. —- Wiley Publishing, 2004. — 591 p.

33. Bai Ying. Application interface programming using multiple languages. — Prentice Hall PTR (www.phpt.com), 2003. — 868 p.

34. Lenz G., Wienands C. Practical software factories in .NET. — From theory to practice — a primer reference and case study. — Apress, 2007. — 205 p.

35. Duvall P., Matyas S., Glover A. Continuous integration improving software quality and reducing risk. — Addison Wesley, 2009. —

36. Лавріщева К.М. Підхід до формального подання онтології життєвого циклу програмних систем // Вісн. Київ. нац. у-ту імені Тараса Шевченка. Сер. фіз.-мат.наук. — 2013. — № 4. — С. 140–149.

37. Лаврищева Е.М., Грищенко В.Н. Сборочное программирование. Основы индустрии программных продуктов. — К.: Наук. думка, 2009. — 371 с.

38. Лавріщева Е.М., Зинькович В.М., Колесник А.Л. та ін. Інструментально-технологічний комплекс розробки и навчання прийомам виробництва програмних систем. — Державна служба інтелектуальної власності України. — Свідоцтво про реєстрацію авторського права на твір. – № 45292 від 27.08.2012.

39. Нові теоретичні засади технології виробництва сімейств програмних систем у контексті генерувального програмування / К.М. Лавріщева , Г.І. Коваль , Л.П. Бабенко та ін. // ВИНИТИ РФ, 2012. — № 2–3, Анот. — с. 1–2, 277 с.

40. Андон П.І., Лавріщева К.М. Методологія побудови ліній виробництва програмних продуктів і їх застосування // Праці Міжнар. наук. конгресу «Інформаційне суспільство в Україні» 25–26 листопада 2012 р. — Київ: Держ. агентство з питань науки, іновацій та інформатизації України — .

41. Лавріщева К.М., Колесник А.Л., Стеняшин А.Ю. Об’єктне-компонентне проектування програмних систем. Теоретичні і прикладні питання // Вісн. Київ. нац. у-ту імені Тараса Шевченка. Сер. фіз.-мат.наук. — 2013. — № 4. — С. 150–162.

42. Lavrischeva E., Ostrovski A. New theoretical aspects of software engineering for development applications and e-learning // J. Software Engineering and Applications. — 2013. — **6**. — P. 34–40.

Поступила 15.02.2013
После доработки 24.12.2013