Математические структуры и моделирование
2011, вып. 24, с. 55–68

УДК 004.4'2

ПРЕДМЕТНО-ОРИЕНТИРОВАННЫЕ ПРОЕКТНЫЕ РЕШЕНИЯ ДЛЯ ТЕМАТИЧЕСКОЙ ОБЛАСТИ ОБУЧАЮЩИХ ПРОГРАММНЫХ СРЕДСТВ НА ОСНОВЕ ЛИНГВИСТИЧЕСКИХ ИГР

С.В. Гусс

В работе представлено обобщённое (без деталей, приводимых в оригинальной документации) описание предметно-ориентированных проектных решений для тематической области обучающих программных средств на основе лингвистических игр. Проектные решения для этой области предлагаются впервые; в их состав входят описание предметной области, а также модели программных систем и элементов, сопровождаемые критериями оценок качества и примерной оценкой затрат на разработку программных продуктов.

Введение

В Государственном стандарте (далее — ГОСТ) 24.703-85 «Типовые проектные решения в АСУ — Общие положения» на разработку автоматизированных систем управления представлено определение термина «типовое проектное решение». Если перенести его на область разработки программного обеспечения (далее — ПО), то получится следующее («АСУ» заменено на «комплекс программ» согласно терминологии программной инженерии). ***Типовое проектное решение*** — «техническая документация, содержащая проектные решения по части объекта проектирования, включая программные средства (*далее — ПС*), и предназначенная для многократного применения в процессе разработки и функционирования *комплексов программ* с целью уменьшения трудоёмкости разработки, сроков и затрат на создание *комплекса программ* и его частей».

Под инженерией, согласно стандарту Международной организации по стандартизации (далее — ISO: International Organization for Standardization), Международной электротехнической комиссии (далее — IEC: International Electrotechnical Commission), Института инженеров по электротехнике и радиоэлектронике (далее — IEEE: The Institute of Electrical and Electronics Engineers, Inc.) ISO/IEC/IEEE 24765:2010 «Systems and Software Engineering — Vocabulary» (Системная и программная инженерия — Термины и определения), понимается применение систематизированного, упорядоченного и количественно

Омский государственный университет им. Ф.М. Достоевского
E-mail: InfoGuss@gmail.com

измеримого подхода к структурам, продукции, системам и процессам. Применительно к программной инженерии, на основании IEEE 1517-2010 «System and Software Life Cycle Processes — Reuse Processes» (Процессы жизненного цикла систем и ПО — Процессы повторного использования), инженерный процесс создания и модернизации ПО основывается на многократном использовании ресурсов.

Согласно ГОСТ 34.003-90 «Автоматизированные системы — Термины и определения», ***типовое проектное решение*** — это проектное решение, предназначенное для повторного использования при проектировании.

Таким образом, согласно обозначенной теме, для предметной области игровых обучающих программных средств (далее — ОПС) предлагаются такие технические, инженерные решения (библиотеки компонентов, модели, шаблоны и т.д.), которые можно повторно использовать в различных программных проектах, руководствуясь соответствующими документами и рекомендациями, для достижения эффекта, в виде уменьшения трудоёмкости, сроков и затрат на разработку конечных программных продуктов.

Смысл предметно-ориентированных проектных решений — задать основу для готовых производственных решений, на экономические характеристики которых, согласно [1], могут влиять: «результаты системного анализа, применение функциональных и информационных моделей предметной области, формализация спецификаций требований, функциональная декомпозиция программных комплексов и последовательная детализация проектов». Библиотеки компонентов и шаблоны могут стать программным заделом, а модели, примеры и описание предметной области — основой для составления руководящих материалов организации разработчиков ОПС на основе лингвистических игр.

1. Распространение программной инженерии

На сегодняшний день часть методов и средств программной инженерии направлена на обеспечение мобильности и повторного использования программных элементов и систем [1]. Современные методы, предлагающие способы решения проблемы эффективности повторного использования и качества конечных систем, развиваются в направлении обеспечения поддержки автоматизации и систематизации этого процесса [2–4]. Всё это происходит в рамках конкретной предметной области, в контексте разработки качественных программных продуктов и требует серьёзных работ по проектированию структуры и взаимодействия составляющих ПС, пригодных для повторного использования.

Новые подходы, учитывающие специфику предметной области, позволяют получить прирост производительности и уменьшить трудоёмкость до 5–10 раз. Предметно-ориентированные подходы применяет в своей практике ряд известных в мире компаний. Среди них «Nokia» (приложения для мобильных устройств, полученный результат — до 10 раз уменьшения трудоёмкости); «United States Air Force» (военное управление, до 3 раз уменьшения трудоёмкости); «Lucent» (ПО на заказ, результаты — от 3 до 10 раз уменьшения трудоёмкости) и многие другие [3].

Эти новые подходы и методики способствуют улучшению показателей процессов и результатов разработки ПО. Они применяются в контексте так называемого предметно-ориентированного проектирования (далее — DDD: Domain-Driven Design), в рамках которого активно используются достижения предметно-ориентированного моделирования (далее — DSM: Domain-Specific Modeling).

Предметно-ориентированные решения, получаемые в рамках DDD, могут быть оформлены как фабрики разработки ПО [5] (далее — SF: Software Factory). DDD пользуется достижениями, полученными в результате развития дисциплины разработки ПО в различных предметных областях. Здесь учитывается как техническая, так и экономическая сторона вопроса. Центральная идея — накопление опыта создания ПС в конкретной предметной области, в виде повторно применимых элементов ПО (модулей, библиотек компонентов, каркасов, инструментов, специальных языков, профилей, схем, сценариев и документации для различных этапов процесса разработки).

Предметные области, в которых уже применяется DDD, — приложения для микроконтроллеров, бизнес-процессы, обслуживание вызовов, медицина, телекоммуникации и т.д. На сегодняшний день единого подхода к созданию и формализации предметно-ориентированных решений не существует, поскольку каждая предметная область имеет свои характерные особенности, которые и определяют свойства проекта. Тем не менее имеются определённые рекомендации и систематизированные решения для разработки ПО общего назначения. Часто это решения в проблемной, а не предметной области. Первая область шире, чем вторая и имеет дело с повторным использованием элементов технического характера, нежели прикладного, тематического.

Например, существуют следующие решения в различных проблемных областях, оформленные в виде SF — «Smart Client Software Factory» (Фабрика разработки композитных приложений), «Mobile Client Software Factory» (Фабрика разработки клиентских модулей приложений для мобильных устройств), «Web Service Software Factory» (Фабрика разработки Web-сервисов) и другие, в рамках инициативы «Microsoft Patterns and Practices» [6].

Для узких предметных областей также существуют подобные решения, однако, учитывая их коммерческую ценность и ограниченную применимость, они представлены научному и инженерному сообществу чаще в виде краткого описания результатов их применения и не раскрывают деталей функционирования.

В качестве примера доступного предметно-ориентированного решения для узкой предметной области (близкой к той, которая обозначена в теме статьи, однако не учитывающей обучающий аспект), можно привести проект исследователя Андре Фуртадо (Andre Furtado) — «SharpLudus» [7] для предметной области компьютерных игр в жанре «аркада», частично оформленный в соответствии с методологией SF. Проект представлен не только как описание результата — опубликованы и детали его реализации (в работах [8,11]). Стоит, однако, подчеркнуть, что решения, оформленные в виде SF, — это уже готовые производственные решения. Цель же исследования в рамках программной инженерии — предоставить проектные решения, которые могут быть использо-

ваны в организации для составления производственных решений, например, в виде тех же SF.

В целом, идеи DDD и DSM находятся в состоянии развития, поскольку потребность в информатизации различных областей применения с каждым годом возрастает и ожидает накопления результатов, а также дополнительной информации и статистики, полученной на практике. При малой освещённости практические результаты такого рода представляют определённый научный интерес для развития методологии программной инженерии [1,12], в особенности в отечественной программной инженерии.

2. Развитие предметной области обучающих программных средств

В таких областях, как сетевое и межсетевое взаимодействие, а также медицина, в виду высокой критичности проектов уже имеется ряд полезных решений, повышающих эффективность и качество разработок. Например, для сетевого и межсетевого взаимодействия существует программный каркас «Adaptive Communication Environment» [13], позволяющий эффективно решать задачи сетевого программирования, предоставляя доступ к уже реализованным высококачественным элементам. Для медицины — каркас для работы с рентгенографическими и томографическими изображениями [14], а также каркас для разработки новых алгоритмов и прототипирования ПС медицинских симуляторов [15]. В этих областях достигнуты определённые успехи, а полученные на практике знания упорядочены и оформлены для соответствующей целевой аудитории разработчиков ПО. Однако этого нельзя сказать (с полной уверенностью) о предметной области ОПС и поддержке электронного игрового обучения (далее — ЭИО).

Есть и ещё одна проблема. В ПС для сферы образования имеется явный перевес — в сторону предоставления только учебного содержимого, чем удовлетворяется потребность ряда заказчиков. Это подход, когда материальные информационные элементы переводятся в электронную форму с добавлением средств мультимедиа (так называемые «электронные учебники»). Такие тиражируемые продукты (или материалы, если речь идёт о системах поддержки дистанционного обучения) пользуются спросом, однако не удовлетворяют потребность некоторых преподавателей и учителей в выражении своих творческих идей.

В данном случае встаёт вопрос об эффективном создании надёжных ПС для предоставления уникального учебного содержимого, которое способен передать конкретный преподаватель. А ведь именно в уникальности, в специфической видимости и понимании предмета, обуславливающейся приобретённым опытом, и заключается его (преподавателя) ценность. Здесь речь идёт, в равной степени, как о необходимости передачи статического содержимого, так и о желании, в некоторых случаях, придать динамический характер работе с определённым содержимым (наиболее подходящий кандидат на средства предоставления такой возможности — лингвистическая игра [16]).

Имеющиеся достижения в решении этого вопроса (о разработке надёжных ОПС, предоставляющих возможность творческого самовыражения преподавателей) часто ограничиваются фантазией и некоторыми техническими способностями самих учителей и преподавателей и не имеют ничего общего с разработкой эффективных, качественных продуктов. Более того, это не способствует развитию процесса разработки и эффективности ПО. В такой ситуации проектные решения не формализуются, а накопленные знания и опыт являются достоянием группы индивидов и не распространяются за её пределы.

Для решения этой проблемы необходимо тщательно подобрать определённые методы и средства, представить ряд проектных решений и разработать программные активы, которые могли бы быть полезными в предметной области. Для практики интерес представляют средства поддержки процесса проектирования (описания, модели, каркасы, шаблоны, критерии оценки характеристик качества и т.д.), учитывающие специфику предметной области. Для теории определённый интерес имеется в обосновании, развитии и уточнении методов, основанных на следующих подходах.

1. **Написание небольших алгоритмических частей**, 5–15 строк [17]. Для предметной области ОПС автор предлагает модель и программную базу в виде каркаса для уточнения игровых компонентов. Уточнение одного определённого аспекта (для которого нужно составить алгоритм) в рамках такого каркаса в среднем составляет 5–15 строк программного кода.

2. **Написание легко интегрируемого кода предметной области** [18]. Предлагается разбиение системы на ряд подсистем с выделением проблемно и предметно-ориентированных компонентов, которые можно использовать в последующих программных проектах. Простота интеграции проблемно-ориентированных компонентов достигается за счёт сокрытия специфической информации за внешним программным интерфейсом. Для упрощения интеграции предметно-ориентированных компонентов предлагается специальный шаблон настройки каркаса игровых компонентов, составляющих предметно-ориентированную часть конечного приложения ОПС.

3. Основания для составления проектных решений

Далее в подразделах приводятся результаты исследований рынка ОПС и направлений развития дисциплины повторного использования ПО.

3.1. Рынок обучающих программных средств

1. В академическом секторе рынок ОПС ещё не сложился и находится на стадии формирования. Поэтому неизвестно, сколько средств тратится на электронное обучение (далее — ЭО), известно лишь, что как таковые выделяются средства на закупку ПО.

2. В корпоративном секторе затраты на ЭО, в том числе и ОПС, составляют в среднем 12% от бюджета компании. Здесь рынок, в определённой степени, сформирован, имеются свои производители.

3. Для академического сектора, в плане получения полезного эффекта от внедрения ЭО, больше подходят внутрифирменные разработки (с возможной передачей части работы субподрядчикам или путём прямого сотрудничества преподавателя с разработчиками), нежели готовые тиражируемые продукты.

4. Для корпоративного сектора сформировался рынок розничной торговли и услуг в области образовательных ИТ.

5. Препятствия для распространения ЭО в академическом секторе: непонимание реальной цены продукта и его разработки; проблема финансирования; выбор организации, которая будет заниматься разработкой; затраты на эксплуатацию и сопровождение готового продукта.

6. В перспективе — политические заказы на игровые решения. Предполагаемые исполнители заказов — отечественные разработчики.

7. Серьёзные игры (применяющиеся для получения полезного эффекта, в том числе и для обучения) имеют государственную поддержку в развитых странах (Германии, Англии, Соединённых Штатах Америки и т.д.). Их опыт говорит о том, что на разработку таких ПС оказывают влияние следующие факторы: бюджет, время разработки, инструменты разработки, риски, предсказуемость успеха игры у целевой аудитории.

Таким образом, для развития рынка ЭО в академическом секторе (для корпоративного обучения уже есть свои производители) необходимы качественные продукты. Для этого нужна соответствующая поддержка процесса их проектирования и разработки.

3.2. Проектирование и разработка программного обеспечения

1. Особый интерес в рамках проектирования и разработки ПО представляют проблемно (реализующие общий функционал и техническую поддержку) и предметно-ориентированные (связанные со спецификой предметной области) компоненты, поскольку они могут занимать до 65% от всей системы.

2. Программная инженерия предъявляет ряд требований к проектам ПО, среди которых самые существенные — архитектурные: архитектура ПС должна соответствовать текущим целям и задачам, стратегическим и функциональным, быть готова к изменению, развитию и расширению.

3. Одна из причин лимитированного потенциала повторно используемых компонентов — изобилие внутренней информации и необходимость со стороны разработчиков, использующих эти компоненты, изучать большой объём этой информации.

4. Особые препятствия к внедрению дисциплины повторного использования создают требования к организации процесса проектирования и разработки ПО, а также вложение инвестиций на начальных его этапах.

5. Требования, предъявляемые заинтересованными в повторном использовании лицами к программным компонентам: доступность, понятность, возможность сопровождения, документированность, степень сложности компонентной интеграции.

Попытки, инициативы и результаты внедрения повторного использования на предприятиях явились основанием для развития таких приёмов и методик, как семейства ПС, линейки программных продуктов, DDD и DSM. Именно в этом направлении на сегодняшний день движется развитие дисциплины повторного использования.

3.3. Предметно-ориентированная разработка программного обеспечения

1. Сегодняшний день — время быстрых изменений требований к приложениям и смены рабочих кадров. Поэтому весьма оправдан подход, когда опытные разработчики накапливают (формализуют) и автоматизируют лучшие практики и рекомендации для других, менее опытных разработчиков. Таким образом, знания накапливаются, а проекты, в которых они используются, меньше подвержены рискам (даже если высококвалифицированный специалист уйдёт из проекта, формализованные знания останутся).

2. Наиболее перспективны предметно-ориентированные решения, основанные на трёхуровневой архитектуре: язык, генератор кода, каркас. Язык и генераторы работают на базе каркаса. Каркас реализуется над одним из языков программирования общего назначения. Оптимизация программного кода происходит на уровне предметной области, что, как показывает практика, эффективнее оптимизации на низком уровне в рамках обычного прикладного приложения.

3. Каркас — сосредоточение экспертных знаний. Язык — средство управления экспертными знаниями. Каркас и язык инкапсулируют знания об архитектурных правилах построения приложений и модели программирования.

4. Предметно-ориентированное решение может быть оформлено в виде пакета для программной среды разработки, созданного по принципам методологии SF. Однако этот способ пока что не предоставляет достаточно

удобных средств менее опытным разработчикам, поскольку предполагает понимание особенностей и нетривиальных знаний функционирования проектных решений, а также организации информации и инструментов в рамках пакета.

5. Наибольшие затраты времени и ресурсов в процессе проектирования и разработки ПО в определённой предметной области приходятся на поиск хороших абстракций, которые будут использоваться в разработке, а не на создание инструмента, который их поддерживает. Более того, не всегда оправдано разрабатывать генератор программного кода и инструментальные средства поддержи языка. Самое главное — создать язык, на котором могли бы общаться разработчики и специалисты в контексте принимаемых решений и их реализации.

Таким образом, для решения ряда проблем в предметной области ОПС нужна систематизированная, детальная информация о предметной области. Также нужны ПС и документация на эксплуатацию и сопровождение, описанные в терминах, понятных специалистам предметной области и достаточных для формулирования требований к программному продукту.

4. Проектные решения

Далее представлено обобщённое описание проектных решений, предлагаемых автором статьи для предметной области ОПС. Приводится ряд характеристик и отличия от имеющихся решений в других предметных областях.

Предлагаемые предметно-ориентированные проектные решения состоят из следующих элементов.

1. **Описание предметной области**: функциональные, дополнительные и технические требования; решаемые педагогические задачи; системы и подсистемы; пользователи.

2. **Модели ОПС и подсистем**: функции, элементы, взаимодействие.

3. **Каркас игровых компонентов**: архитектура, конструкция, взаимодействие элементов, примеры детализации, средства интеграции компонентов на базе каркаса (шаблоны проекта и предоставления точек расширения).

4. **Критерии оценки характеристик качества** ОПС и их компонентов.

5. **Оценка затрат на разработку** ОПС на базе повторно используемых компонентов.

Отличия предложенных моделей ОПС от представленных в [19].

1. Более детальное представление за счёт большей конкретизации применения, а также поддержка игровой составляющей.

2. Использование нотации языка «UML», позволяющей быстрее перейти к разработке ОПС.

3. Предоставление примеров программной разработки на основе моделей и их концепций с планированием повторного использования. В [19] представлены только возможные сценарии работы систем, построенных на основе моделей.

Упоминаемый проект «SharpLudus», как и представленные в работе [3] проекты («IP Telephony and Call Processing», «Insurance Products», «Mobile Phone Applications»), не учитывает создание независимых повторно-используемых элементов. Эти проекты предлагают готовые производственные решения, которые ограничивают круг возможных приложений и предполагают заданную организацию процесса разработки. Предлагаемые же решения — материал, который можно использовать для создания готовых производственных решений.

С концептуальной точки зрения может быть предложен в сравнение проект «Электронные образовательные ресурсы нового поколения» [20], или просто ЭОР. Отличия предлагаемого подхода от ЭОР состоят в следующем. В ЭОР учебный материал представлен в форме электронной статьи, которая помимо учебного материала содержит мультимедийные вставки, визуализирующие трудные для понимания места. В сравнении с таким подходом, автор статьи предлагает не визуализацию, а предоставление «словесного поля» в форме лингвистической задачи, в рамках которой можно описать изучаемую действительность и с которой можно работать. Вместо наблюдения за анимированной картиной (в случае ЭОР) предлагается обыгрывание ситуации через игру со словами. В данной ситуации от преподавателя требуется проявить творческие способности в выборе типа лингвистической задачи и подбора материала. Предметно-ориентированные компоненты представлены средствами поддержки генерации кроссвордов и библиотекой каркаса программных компонентов. Библиотека кроссвордов представлена как отдельный компонент, который может использоваться в различных проектах. Библиотека каркаса программных компонентов лингвистических игр создавалась на базе знаний предметной области и основана на элементарной схеме взаимодействия субъектов и объектов в рамках игры с добавлением некоторых уточнений и точек расширения функциональности. Средства библиотек поддержки генерации кроссвордов (реализована поддержка классического, венгерского и китайского кроссворда) могут использоваться в рамках детализации каркаса игровых компонентов в случае, если тип выбранной лингвистической задачи — «кроссворд».

Разработанный каркас игровых компонентов предлагается в качестве базы для поддержки составления небольших (5–15 строк) алгоритмических частей, а предлагаемые шаблоны проекта и предоставления точек расширения - средством интеграции компонентов на базе каркаса.

Для оценки и планирования качества разрабатываемых ОПС отобраны (на практике, в рамках разработки проектов ОПС, в большинстве проходивших под управлением канд. техн. наук, доцента каф. вычислительных систем Ефимова С.С.) следующие характеристики, определены соответствующие критерии

выбора и оценки компонентов и систем. Перечислим их.

1. **Первичные характеристики.**

   А. ***Функциональная пригодность***:

   - *назначение компонентов* (критерии — наличие описания предметной области, примеров и сценариев применения);
   - *функции компонентов* (критерии — требования к решаемым педагогическим задачам, обязанностям пользователей и функционированию в рамках конечной ОПС);
   - *полнота решения задачи компонентом* (критерии — пригодность к повторному использованию в последующих проектах, возможность расширения функциональности);
   - *адекватность, или соответствие требований конечного продукта установленным требованиям* (требования определяются на ранних стадиях разработки во время знакомства с документацией, примерами реализации на базе компонентов, соответствующими ограничениями и рисками, критерии в данном случае — точность и полнота документации и примеров);
   - *точность выполнения требований в продукте* (критерии — наличие подтверждения и отчёта о выполнении требований проведёнными экспериментами использования повторно используемых компонентов, моделей, сценариев или шаблонов, а также возможность проведения самостоятельных проверочных тестов);
   - *организация информационного обеспечения* (критерии — наличие поддержки шифрования файлов со словарными базами и заданиями, режимов ограничения доступа к системе во время проверки знаний обучающихся).

   Б. ***Корректность***:

   - *правила структурного построения* (критерии — использование проверенных временем методов и подходов к проектированию ПС, например, принципов «единственной обязанности», «открытости / закрытости» и «инверсии зависимости», а также шаблонов проектирования в структуре готовых программных компонентов);
   - *организация взаимодействия* (критерии — предоставление специальных средств, таких как шаблоны проекта и детализации компонентов на основе программных компонентных каркасов, сокрытия внутренней реализации компонента за внешним программным интерфейсом);
   - *организация интерфейсов* (критерии — соответствие общепринятым соглашениям и идиомам, а также непротиворечивость, концептуальная целостность, логичность, совместимость и согласованность).

В. ***Защищённость***:

- *угрозы* (критерии — наличие инструментов и средств восстановления нормальной работы, стратегия запуска и завершения работы приложений с конфиденциальной информацией).

2. **Определяющие характеристики**.

А. ***Надёжность***:

- *завершённость* (критерий — наличие и полнота статистики работы ПС на основе готовых компонентов в реальных условиях);
- *устойчивость и восстанавливаемость* (критерии — наличие специальных инструментов или средств восстановления работы, стратегия обработки исключительных ситуаций).

Б. ***Эффективность***:

- *временная эффективность* (не существенна: словарные базы, используемые обучающими системами, обычно малы, вычислительных операций немного, есть возможность маскировки обращений к базе знаний игровыми и визуальными эффектами, тем не менее важна информация о временных характеристиках работы компонента, наличие которой и является важным критерием временной эффективности);
- используемость ресурсов (критерий — размер информационного содержимого: текстовый, звуковой, графический материалы).

3. **Конструктивные характеристики**.

А. ***Практичность***:

- *сложность понимания готовых компонентов* (критерий — качество документации: внешние характеристики, субъективное впечатление);
- *сложность использования компонентов* (критерии — степень контролируемости и комфортности, определяющиеся расчётом соответствующих относительных трудозатрат и длительности этапов разработки);
- *сложность изучения принципов работы компонента* (критерии — расчёт и оценка трудоёмкости и длительности освоения).

Б. ***Пригодность к сопровождению***:

- *приспособленность к модификации* (критерии — расчёт сложности исправления, совершенствования и адаптации);
- *приспособленность к изменению* (критерии — предоставление механизмов расширения, соблюдение принципа «открытости / закрытости», когда изменение программного компонента не должно приводить к его перекомпиляции).

В. ***Мобильность***:

- *адаптация* (критерий — расчёт и оценка трудоёмкости процесса приспособления к новым условиям и платформенным ограничениям);
- *инсталляция* (критерий — наличие предустановленных компонентов и средств поддержки работы продукта на различных программно-аппаратных платформах);
- *замена компонентов* (критерий — простота интеграции новых компонентов, выраженная в количестве выполняемых операций и их трудоёмкости).

Накопленная автором статьи статистика разработки ОПС на основе лингвистических игр для академического сектора позволяет сделать оценку затрат (идеальный случай) на разработку конечных продуктов, в случае когда доля (Д) повторно используемых компонентов равна 50%–65% (остальная доля — эксклюзивное информационное, графическое и звуковое содержимое конечного продукта). Полную трудоёмкость можно вычислить по формуле $C_i = \sum_{k=1}^{r} q_k * g_k(\text{Д})$, где i — номер версии ПС, k — конкретный этап из r этапов производства продукта, g_k — степень снижения трудоёмкости разработки продукта на базе повторно используемых компонентов, q_k — доля от полной трудоёмкости на конкретном этапе разработки ПО. Экспериментальные исследования показали, что можно получить сокращение трудоёмкости до 3 раз.

Если в расчёте трудоёмкости создания текущей версии программного продукта учитывать предыдущий опыт $f_{i,k}(\text{Д})$, то влияние доли повторно используемых компонентов на трудоёмкость k-го этапа можно представить как $f_{i,k}(\text{Д}) * g_k(\text{Д})$. Формула для вычисления трудоёмкости в таком случае примет вид: $C_i = \sum_{k=1}^{r} f_{i,k}(\text{Д}) * q_k * g_k(\text{Д})$. В данном случае можно получить дополнительное сокращение трудоёмкости. Пусть трудоёмкость создания первой версии ПС была $C_i = 0{,}44$. Тогда для разработки второй версии ПС она может быть 0,20 (например — разработка такой системы, как «Безопасность») и даже 0,05–0,1 (разработка дополнительного игрового компонента на базе каркаса).

Заключение

В рамках исследования предметной области ОПС произведено уточнение, конкретизация и распространение методов и подходов предметно-ориентированного проектирования и моделирования на класс обучающих игровых программных средств на основе лингвистических игр. В предлагаемых моделях предметной области проводится чёткое разграничение («дистилляция знаний» в терминах предметно-ориентированного проектирования) предметно-ориентированной и обслуживающей её проблемно-ориентированной составляющих. Такое разграничение позволяет яснее понять проблему предметно-ориентированной составляющей и влияние на неё проблемно-ориентированной компоненты.

Для компонентов предметной области предлагается объектно-ориентированный каркас как основной элемент программной архитектуры, построенной по принципам предметно-ориентированного моделирования. Для упорядочивания предметной части при проектировании структуры каркаса применялись классические паттерны проектирования «Стратегия» (Strategy) и «Шаблонный метод» (Template Method), а также такие принципы проектирования, как «принцип единственной обязанности» (Single Responsibility Principle), «открытости / закрытости» (Open / Closed Principle), «инверсии зависимости» (Dependency-Inversion Principle).

Предложенные проектные шаблоны интеграции и уточнения каркаса программных компонентов предметной области чётко выделяют область программного кода, в границах которой происходит уточнение каркаса, позволяют автоматизировать процесс настройки программных компонентов.

Выделенные критерии анализа и планирования характеристик качества обучающих программных средств и повторно используемых компонентов дают возможность сделать рациональный выбор программных продуктов и готовых компонентов.

В заключение хотелось бы ещё раз подчеркнуть, что в статье приводится обобщённое описание того, что было сделано в ходе исследований предметной области обучающих программных средств на основе лингвистических игр. Более подробную информацию можно найти в предыдущих статьях автора, опубликованных в журнале «Математические структуры и моделирование» (выпуски 20–23) [21]. Для вопросов и предложений можно обращаться к автору по электронному адресу `InfoGuss@gmail.com`. Все интересующиеся данной темой (разработчики обучающих игровых программных систем, а также учителя и преподаватели) могут запросить техническую документацию на проектные решения либо версии обучающих программных систем на основе лингвистических игр с руководствами пользователя.

Литература

1. Липаев В.В. Экономика производства программных продуктов. Издание второе. М. : СИНТЕГ, 2011. 352 с.
2. Чарнецки К., Айзенекер У. Порождающее программирование: методы, инструменты, применение. СПб. : Питер, 2005. 731 с.
3. Kelly S., Tolvanen J-P. Domain-Specific Modeling. Enabling Full Code Generation. Canada : John Wiley & Sons, Inc., 2008. 448 p.
4. Sutcliffe A. The Domain Theory: Patterns for Knowledge and Software Reuse. United States of America : CRC Press, 2002. 424 p.
5. Фабрики разработки программ: потоковая сборка типовых приложений, моделирование, структуры и инструменты. / Гринфилд Д. [и др.]. М. : ООО «И.Д. Вильямс», 2007. 592 с.
6. Инициатива «Microsoft Patterns and Practices» // MSDN: сеть для разработчиков, использующих платформу компании «Microsoft» URL: `http://msdn.microsoft.com/en-us/practices/bb190332` (дата обращения: 02.11.2011).

7. Проект «SharpLudus» // официальный сайт сообщества разработчиков «CodePlex Open Source Community». URL: http://sharpludus.codeplex.com (дата обращения: 02.11.2011).
8. Furtado A., Santos A., Ramalho G. A Computer Games Software Factory and Edutainment Platform for Microsoft .NET // официальный сайт организации «Centro de Informatica UFPE». URL: http://www.cin.ufpe.br/~awbf/files/IET_SharpLudus.pdf (дата обращения: 02.11.2011).
9. Furtado A., Santos A. Defining and Using Ontologies as Input for Game Software Factories // официальный сайт организации «Centro de Informatica UFPE». URL: http://cin.ufpe.br/~sbgames/proceedings/files/DefiningandUsingOntologies.pdf (дата обращения: 02.11.2011).
10. Furtado A., Santos A. Extending Visual Studio .NET as a Software Factory for Computer Games Development in the .NET Platform // официальный сайт организации «Centro de Informatica UFPE». URL: http://www.cin.ufpe.br/~awbf/files/IVNET2006_SharpLudus.pdf (дата обращения: 02.11.2011).
11. Furtado A., Santos A. Using Domain-Specific Modeling towards Computer Games Development Industrialization // DSM Forum: форум сообщества разработчиков, продвигающих предметно-ориентированное моделирование. URL: http://www.dsmforum.org/events/DSM06/Papers/1-Furtado.pdf (дата обращения: 02.11.2011).
12. Боэм Б.У. Инженерное проектирование программного обеспечения. М. : Радио и связь, 1985. 512 с.
13. The ADAPTIVE Communication Environment // официальный сайт высшего учебного заведения «Washington University in St. Louis». URL: http://www1.cse.wustl.edu/~schmidt/ACE.html (дата обращения: 02.11.2011).
14. Medical Imaging Software Framework for Rapid Development and Clinical Interoperability // официальный сайт компании «TATRC». URL: http://www.tatrc.org/docs/PLR/08_ImagingSoftware.pdf (дата обращения: 02.11.2011).
15. SOFA — an Open Source Framework for Medical Simulation // Официальный сайт компании «Evasion». URL: http://www-evasion.imag.fr/Publications/2007/ACFBPDDG07/ (дата обращения: 02.11.2011).
16. Любич Д.В. Лингвистические игры. СПб. : Изд-во Буковского, 1998. 272 с.
17. Степанов А., Мак-Джонс П. Начала программирования, М. : ООО «И. Д. Вильямс», 2011. 272 с.
18. Эванс Э. Предметно-ориентированное проектирование (DDD): структуризация сложных программных систем. М. : ООО «И.Д. Вильямс», 2011. 448 с.
19. Башмаков А.И., Башмаков И.А. Разработка компьютерных учебников и обучающих систем. М. : Информационно-издательский дом «Филинъ», 2003. 616 с.
20. Официальный сайт Федерального центра информационно-образовательных ресурсов. URL: http://eor.edu.ru (дата обращения: 02.11.2011).
21. Журнал «Математические структуры и моделирование», Омск: Омский государственный университет, Вып. 20–23.