Обучение передовым технологиям разработки ПО: проблемы и методы их решения

В. В. Кулямин, В. А. Омельченко, О. Л. Петренко

1. Введение

В современном мире все больше задач в экономике, социальной и частной жизни людей решаются с использованием программного обеспечения (ПО). В связи с этим функциональность индустриально разрабатываемого ПО все больше усложняется, и вместе с тем возрастают требования к его надежности и другим показателям качества. Проблемы построения надежного ПО высокой сложности не могут быть решены без привлечения передовых методов его разработки, использующих последние достижения научной мысли.

Однако внедрение подобных методов в промышленную практику сталкивается с целым рядом проблем, которые можно классифицировать следующим образом.

1. Проблемы исследовательского плана, связанные с принципиальной ограниченностью метода и возможностью использовать его только для разработки ПО, относящегося к определенным предметным областям, или построенного по архитектуре определенного вида. Решение таких проблем предполагает значительную модификацию метода с целью расширения его области применимости на новые классы ПО, и, обычно, требует большого объема исследовательских работ.
2. Проблемы технического характера, связанные с необходимостью привязки метода к технологиям и инструментам разработки, использующимся в организации, куда он внедряется. Проблемы такого рода обычно можно решить, разработав инструменты или дополнения к имеющимся инструментам, которые позволяют использовать рассматриваемый метод разработки из привычного персоналу организации окружения.
3. Проблемы организационного характера, связанные с необходимостью сочетать сложившиеся процессы разработки ПО и практику управления этими процессами с новым, нетрадиционным методом разработки. При использовании обычных форм управления и распределения труда, обычных метрик производительности такой метод часто снижает

эффективность производства или дает настолько парадоксальную его картину, что строить работу по ней большинство управленцев с традиционной подготовкой не способно.

4. Проблемы культурного плана возникают из-за повышенных, или просто отличающихся от традиционных, требований к персоналу, предъявляемых передовыми методами разработки ПО. При этом эффективное производство разлаживается как из-за неспособности имеющегося персонала справиться с новыми задачами, так и из-за неспособности менеджеров найти подходящий персонал в силу необычности требований к его знаниям и умениям.

Данная статья предлагает подходы к решению проблем, относящихся к последним двум категориям напрямую связанных с проблемами внедрения. Технические и исследовательские аспекты передовых методов разработки ПО в ней рассматриваются лишь в контексте проблем внедрения новых методов в производство.

С точки зрения преодоления проблем как организационного, так и культурного характера, существенным недостатком передовых методов разработки является большая трудоемкость их освоения, которая вызвана сложностью, внутренне присущей проблемам разработки сложного многоцелевого ПО, и недостаточной «обкатанностью» подобных методов на практике. Методы такого рода появляются на основе исследований, и их первоначальное представление обыч-но отличается большей абстрактностью изложения по сравнению с обычным уровнем, на котором привыкло работать большинство разработчиков ПО, и слабой привязкой к задачам, с которыми им постоянно приходится сталкиваться.

Сложность передовых методов и трудность их освоения отталкивает многие организации от внедрения подобных подходов. Поэтому, успешное использование их на практике предполагает предварительное *обучение* пользователей и руководителей эффективному использованию этих методов и поддерживающих их инструментов. Руководителей приходится обучать также эффективным подходам к организации труда в соответствии с новыми реалиями, складывающимися в организации, использующей такие методы, и способам измерения производительности труда и других показателей, учитывающим специфику этих подходов.

Обучение разного рода является практически единственным эффективным методом решения проблем организационного и культурного характера, с которыми приходится сталкиваться при внедрении передовых методов разработки, как в краткосрочном, так и в долгосрочном плане. Разные виды обучения включают следующие:

1. Обучение студентов ВУЗов по специальностям, связанным с разработкой ПО, как подготовка кадров для индустрии производства ПО.

2. Обучение непосредственных пользователей инструментов, поддерживающих передовые методы разработки.
3. Обучение руководящего персонала методам эффективного управления проектами в новом окружении.
4. Обучение сотрудников организаций, отвечающих за модернизацию процессов разработки методам успешного внедрения рассматриваемых подходов.
5. Широкая пропаганда передовых методов, проведение открытых семинаров по ним для руководителей и технических специалистов.

Примером передовых методов разработки ПО являются методы тестирования ПО на основе моделей, позволяющие эффективно контролировать качество промышленного ПО высокой сложности, чего практически невозможно добиться с помощью традиционных подходов. Несмотря на наличие такого важного преимущества внедрить такие методы не просто. При внедрении методов тестирования на основе моделей в индустриальные процессы разработки возникают те же проблемы, что и при внедрении большинства передовых методов разработки ПО.

Данная статья представляет подход к обучению таким методам на примере технологии тестирования на основе моделей UniTesK, разработанной в группе верификации, спецификации и тестирования ИСП РАН (RedVerst, [1]).

В статье излагается почти десятилетний опыт использования различных образовательных методов при обучении будущих тестировщиков, архитекторов, руководителей проектов, а также студентов ВУЗов передовым технологиям разработки тестов на основе моделей, полученный группой верификации, спецификации и тестирования ИСП РАН.

2. Цели обучения и его структура

Конечной целью обучения будущих пользователей новой технологии является внедрение этой технологии в практику промышленного производства ПО. Для достижения этой цели необходимо учитывать интересы и различия в восприятии и оценках различных групп пользователей.

1. *Студенты*, получающие базовое теоретическое образование, которое они надеются с пользой применить на своем будущем месте работы. Для студентов крайне важно, чтобы полученные ими теоретические знания подкреплялись практическим опытом работы в реальных проектах, который им может обеспечить ИСП РАН, вовлеченный в академические и промышленные разработки. Наличие навыков использования знаний на практике позволит студенту в дальнейшем получить более интересную работу.
2. *Тестировщики ПО*, которые непосредственно после обучения должны будут использовать полученные навыки в конкретных проектах.

3. *Проектировщики и архитекторы ПО*, которые должны понять, каким образом данная технология влияет на их работу, что она потребует делать в дополнение к обычной практике, и как это отразится на свойствах результирующего продукта.
4. *Руководители проектов*, которым необходимо ознакомиться с экономическими показателями технологии в различных типах проектов, понять, как должны быть изменены процессы разработки при использовании в них данной технологии, какие формы деятельности и документов она добавляет в процесс, а какие ранее использовавшиеся типы документов и виды деятельности становятся излишними, какие метрики надо применять для оценки состояния процесса и результирующего продукта.

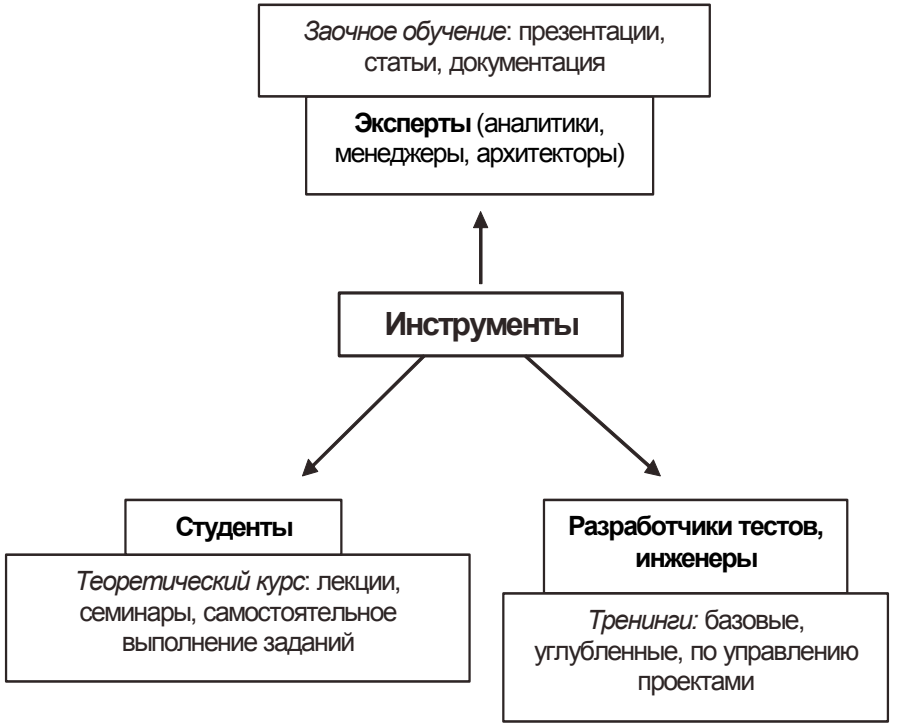

Рис. 1. Общая структура обучения.

Поскольку удовлетворить потребности всех лиц, затрагиваемых технологией, в рамках одного курса обучения практически невозможно, были разработаны следующие виды курсов, нацеленные на более узкие группы лиц:

1. Теоретический курс по формальным методам разработки программ и их использованию в тестировании для студентов ВУЗов, дополненный серией практических занятий.
2. Базовые тренинги по использованию инструментов, поддерживающих технологию разработки тестов на основе моделей, для тестировщиков ПО. Для каждого инструмента требуется отдельный такой тренинг.
3. Углубленные тренинги по использованию инструментов, поддерживающих технологию разработки тестов на основе моделей, для проектировщиков и архитекторов ПО.
4. Тренинг по управлению проектами с использованием технологии разработки тестов на основе моделей для руководителей проектов.

Можно использовать также заочный способ обучения, когда специалист изучает статьи, документацию и другие материалы, но при этом важно поддерживать активную связь с обучаемым, контролировать получение им необходимых знаний и навыков. Общая структура обучения показана на Рис. 1.

3. Организация обучения

3.1. Теоретический курс

В учебном плане факультета ВМК МГУ им. Ломоносова есть курс «Формальные спецификации программ», который преподается студентам 4-го курса. Одной из важнейших целей данного курса является демонстрация современных достижений в области использования формальных методов при создании реального программного обеспечения. Для решения этой задачи, с одной стороны, необходимо было выбрать язык спецификаций, хорошо зарекомендовавший себя как инструмент при использовании в реальных проектах, и, с другой стороны, показать, как формальные методы могут использоваться в ходе всех фаз жизненного цикла программного обеспечения: от проектирования до внедрения ПО. Первая часть курса содержит теоретические лекции, которые позволяют студентам получить знания о формальных методах на основе изучения языка RSL. Затрагиваются следующие темы:

- Место формальных спецификаций в промышленной разработке ПО.
- Виды формальных спецификаций.
- Метод разработки RAISE (RAISE Development Method) и язык спецификаций RAISE (RAISE Specification Language, RSL) [2].

Вторая часть курса посвящена тестированию на основе моделей. В ней рассматриваются как общие подходы к такому тестированию, так и конкретный пример — технология разработки тестов UniTesK, разработанная в Институте системного программирования РАН в 1994-2000 годах. Теоретический материал этой части излагается в трех лекциях:

«Математические модели ПО, используемые в тестировании», «Спецификации и оракулы», «Проблемы построения тестовой последовательности».

UniTesK — это технология тестирования на основе формальных спецификаций. Все инструменты семейства UniTesK используют общие методы описания (спецификации) функциональности тестируемого ПО и унифицированную архитектуру тестовой системы. Инструменты семейства отличаются друг от друга языком разработки ПО, для тестирования которого они предназначены. Это может быть C, C++, Java или C#. Основные шаги разработки тестов по технологии UniTesK представлены на следующем рисунке, подробнее об этой технологии см. [3,4].

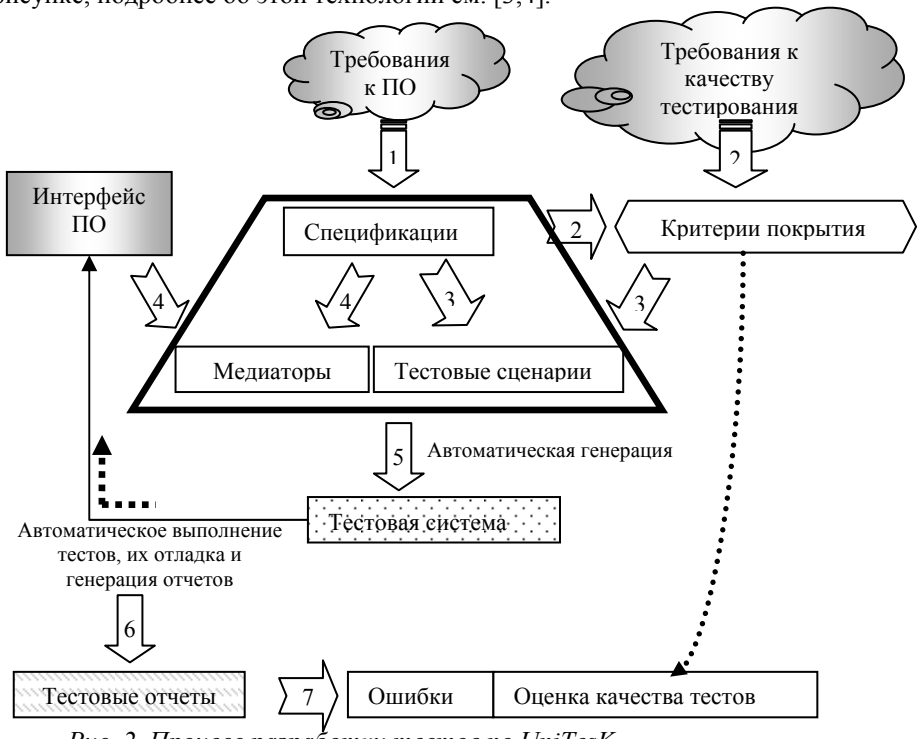

Рис. 2. Процесс разработки тестов по UniTesK.

Особенностью курса можно считать большую практическую часть, в которой студенты сначала знакомятся со способами разработки формальных спецификаций на языке RSL, а затем с инструментами тестирования, используемыми в реальной практике (например, J@T).

Студенты проходят практику в ИСП, где их включают в рабочие группы по проектам с выполнением конкретного задания. У каждого студента есть «наставник» в проекте, который помогает организовать реальную работу и получение конкретных результатов, поскольку, даже имея хорошую

теоретическую подготовку, с применением своих знаний на практике студенты обычно испытывают трудности. В дальнейшем полученные ими результаты находят отражение в курсовых и дипломных работах.

После окончания курса студенты знают:

- Как используются математические модели при тестировании ПО;
- Как проектируются тестовые системы для сложного ПО;
- Как используется тестирование в промышленном производстве ПО;
- Как можно организовать автоматизированное тестирования разных видов ПО

и умеют:

- Разрабатывать формальные спецификации функциональности ПО;
- Разрабатывать медиаторы, связывающие спецификации с реализацией целевой системы;
- Разрабатывать тестовые сценарии;
- Запускать разработанные тесты и проводить анализ результатов тестирования.

Для углубленного изучения (по выбору студентов) проводится спецсеминар «Верификация и валидация программ».

На пятом курсе Московского физико-технического института читается обзорный курс «Языки формальных спецификаций», который содержит следующие лекции:

- Место формальных спецификаций в промышленной разработке ПО;
- Виды формальных спецификаций;
- Абстрактные типы данных и их применение в спецификациях;
- Описание инвариантов, аксиом, пред- и постусловий — основа спецификаций;
- Моделирование с использованием языков формальных спецификаций, проверка согласованности моделей;
- Использование формальных спецификаций для тестирования программ.

После окончания этого курса студенты знают, чем языки формальных спецификаций отличаются от языков программирования, умеют различать языки структурных и функциональных спецификаций, понимают принципы моделирования с использованием формальных спецификаций и их использование в тестировании. Поскольку курс запланирован как обзорный, то практической части в нем нет.

Все перечисленные курсы разработаны сотрудниками ИСП РАН, имеющими уникальный многолетний опыт использования формальных спецификаций как для описания функциональности сложного промышленного ПО и анализа его свойств, так и для последующей разработки тестов на основе спецификаций. Некоторые из них являются ведущими мировыми специалистами по методам построения тестов на основе моделей.

Вторая часть курса «Формальные спецификации программ» распространяется в университетах Европы, Азии и Австралии под академической лицензией.

Рассматриваемый подход к обучению студентов позволяет:

- Поддерживать хороший теоретический и практический уровень преподаваемых курсов;
- Быстро знакомить студентов с новейшими достижениями данной отрасли;
- Давать твердые практические навыки, применимые в реальной работе;
- Отбирать на ранних этапах обучения перспективных студентов для научной работы.

3.2. Обучающий тренинг

Вести обучение тестировщиков, работающих в промышленных компаниях, занимающихся разработкой ПО, по аналогии с обучением студентов нельзя, так как основными требованиями к процессу обучения являются: короткий срок обучения (несколько дней) и гарантированный результат. Как способ обучения в этом случае, был выбран *обучающий тренинг*. Обучающий тренинг рас-сматривается как базовый курс, после прохождения которого человек уже может заниматься тестированием программ. С другой стороны, он является той основой, которая позволяет совершенствовать дальнейшее мастерство тестирования и грамотно использовать ко-верификационные процессы разработки. Ко-верификационный процесс [5] — это процесс разработки ПО, при котором одновременно с разработкой самого ПО разрабатываются средства проверки его корректности. Использование процессов такого вида является одним из перспективных подходов к созданию качественного программного обеспечения.

Цель тренинга — выработка необходимых умений и навыков для разработки тестов на основе спецификаций по технологии UniTesK.

Для быстрого и качественного усвоения навыков разработки тестов на основе моделей в основу обучающего тренинга были положены следующие принципы:

1. Тщательный отбор содержания обучения в соответствии с поставленными целями.
2. Выбор адекватных форм обучения.
3. Обучение слушателей в активном, деятельностном режиме.
4. Создание благоприятной образовательной среды [6].

Каждый из этих принципов определенным образом реализован в предлагаемом тренинге, что позволило сократить сроки обучения до 4-5 дней и давать гарантированные результаты обучения.

3.2.1. Отбор содержания

Для отбора содержания используется метод генерализации учебного материала, при котором выделяется один или несколько основополагающих

принципов, которые многократно и всесторонне раскрываются в курсе с самых разных сторон. В данном случае такими принципами являются основные положения UniTesK:

- тестирование рассматривается как сопоставление свойств поведения тестируемого ПО свойствам модели, заданной формальными спецификациями;
- тестируемое ПО рассматривается как «черный ящик», т.е. оно тестируется как набор интерфейсов (операций, процедур, методов, структур данных и пр.), внутренняя организация которых, их реализация, алгоритмы работы не рассматриваются;
- спецификация задается в форме пред- и постусловий операций и инвариантов типов, задающих ограничения целостности данных;
- интерфейсы самого целевого ПО и его модели, заданной спецификациями, могут различаться как по структуре так и уровнем абстракции, связь между соответствующими интерфейсами задается специальными программными компонентами, называемыми медиаторами;
- UniTesK предлагает унифицированную архитектуру тестовой системы, эта архитектура предусматривает отдельные компоненты для генерации тестовых воздействий и для анализа поведения тестируемой системы;
- подсистема генерации тестовых воздействий разбита на два уровня: генерация входных тестовых данных для отдельных операций (итераторы значений параметров) и генерация последовательности вызовов операций (тестовой последовательности);
- в основе метода генерации тестовой последовательности лежит идея обхода некоторого конечного автомата, который строится на основе спецификаций операций и тестового сценария.

Отобранное по этим принципам содержание обучения представляется в виде программы тренинга. На Рис. 3 приводится программа одного из разработанных нами тренингов по технологии CTesK, входящей в семейство UniTesK и предназначенной для тестирования программ, написанных на языке C.

Получение знаний, перечисленных в этой программе, дает необходимую базу для усвоения следующих умений и навыков:

1. Тестирование функций, поведение которых не зависит от истории взаимодействия системы с окружением:
   - выделение предусловия функции;
   - выделение ветвей функциональности;
   - выделение постусловия функции;
   - выделение инвариантов типов;
   - описание итерации параметров.

Программа курса по технологии тестирования CTesK

Введение

Назначение технологии тестирования CTesK. *Понятия целевой программной системы, тестовой системы, результаты выполнения тестов.*

Архитектура теста в CTesK. *Постоянные компоненты. Генерируемые компоненты. Компоненты, разрабатываемые вручную.*

Шаги разработки теста в CTesK. *Разработка спецификаций, разработка медиаторов, разработка сценария, отладка, выполнение тестов.*

Тестирование на соответствие. *Формальные требования. Критерии покрытия требований.*

Спецификации функций целевой программной системы

Спецификации функций, поведение которых не зависит от истории взаимодействия системы с окружением. *Спецификации данных: определение модельных типов данных, инварианты типов. Спецификации функций: структура спецификации функции, пред- и постусловия, критерии покрытия.*

Спецификации функций, поведение которых зависит от истории взаимодействия системы с окружением. *Спецификация данных: состояние системы, инварианты переменных состояния. Спецификации функций: пре- и пост- состояние; пре-состояние в предусловии и критериях покрытия; пре- и пост- состояние в постусловии.*

Медиаторы

Связь спецификаций и реализации целевой системы. *Структура медиаторов на SE C.* ***Преобразования модельных типов в реализационные и обратно.*** *Вычисление параметров для вызова целевых функций. Вычисление возвращаемого значения спецификации.*

Синхронизация состояния спецификации с состоянием реализации. *Вычисление модельного состояния из реализационного (открытое реализационное состояние). Вычисление модельного состояния без доступа к реализационному (закрытое реализационное состояние).*

Сценарии

Сценарии для тестирования функций, поведение которых не зависит от истории взаимодействия системы с окружением. *Структура сценария. Определение сценария с единственным состоянием. Итераторы параметров спецификаций функций. Процедуры инициализации и завершения теста.*

Сценарии для тестирования функций, поведение которых не зависит от истории взаимодействия системы с окружением. *Определение сценария с множеством состояний, определенным в спецификациях. Определение сценария с обобщенным состоянием.*

Отладка

Шаги отладки тестовой ***системы****. Добавляемые трассировки. Связь компонентов тестовой системы и спецификаций. Виды ошибок.*

Инструменты разработки тестов

Способы вызова инструмента. ***Входная****/выходная информация. Опции в командной строке. Сообщения об ошибках.*

Рис. 3. Программа тренинга по технологии CTesK.

2. Тестирование функций, поведение которых зависит от истории взаимодействия системы с окружением, не нуждающихся в факторизации:
   - выделение предусловия функции;
   - выделение ветвей функциональности;
   - выделение постусловия функции;
   - выделение инвариантов типов;
   - выделение инвариантов переменных состояния;
   - описание тестового состояния;
   - описание итерации параметров.
3. Тестирование функций, поведение которых зависит от истории взаимодействия системы с окружением, требующих факторизации:
   - выделение предусловия функции;
   - выделение ветвей функциональности;
   - выделение постусловия функции;
   - выделение инвариантов типов;
   - выделение инвариантов переменных состояния;
   - описание обобщенного состояния;
   - описание итерации параметров;
   - преобразование обобщенных параметров в параметры функций.

3.2.2. Выбор адекватных форм обучения

Для тренинга основной формой обучения выбран метод «погружения». На протяжении всего курса обучения (4-5 дней) слушатели полный день заняты только в тренинге и не отвлекаются на на другие задачи.

В тренинге предусмотрены следующие формы проведения занятий:

1. **Традиционная лекция.** Она предполагает работу лектора сразу со всей аудиторией в едином темпе и с общими задачами. Используется для сообщения нового теоретического материала. Этот теоретический материал используется в инструктивных лекциях, которые непосредственно предшествуют практическим работам.
2. **Инструктивная лекция** знакомит слушателей с технологией их предстоящей деятельности, с особенностями выполнения отдельных действий, способами работы, алгоритмами решения задач. Учитывая сложность изучаемого курса в инструктивных лекциях даются определения основным понятиям, структура понятий выделенной части предметной области и связи между ними.
3. **Практикумы** проводятся после изучения крупных разделов тренинга в форме самостоятельного выполнения практических работ, с возможностью обратиться за консультацией к тренеру.

   В данном курсе содержанием практических работ являются: написание спецификаций, разработка тестового сценария, запуск и отладка тестов.

Управление деятельностью слушателей во время практикумов сводится к самостоятельному выбору последовательности шагов при решении задачи и самооценки ее выполнения. Последовательность шагов может быть составлена из следующего набора:

- Знакомство с заданием.
- Самостоятельное выполнение задания.
- Сравнение выполненного задания с контрольным вариантом.
- Изучение предлагаемого алгоритма выполнения задания.
- Знакомство с выполнением одного шага предлагаемого алгоритма.

Разные последовательности шагов, которые могут быть составлены для выполнения конкретного задания представлены на Рис. 4.

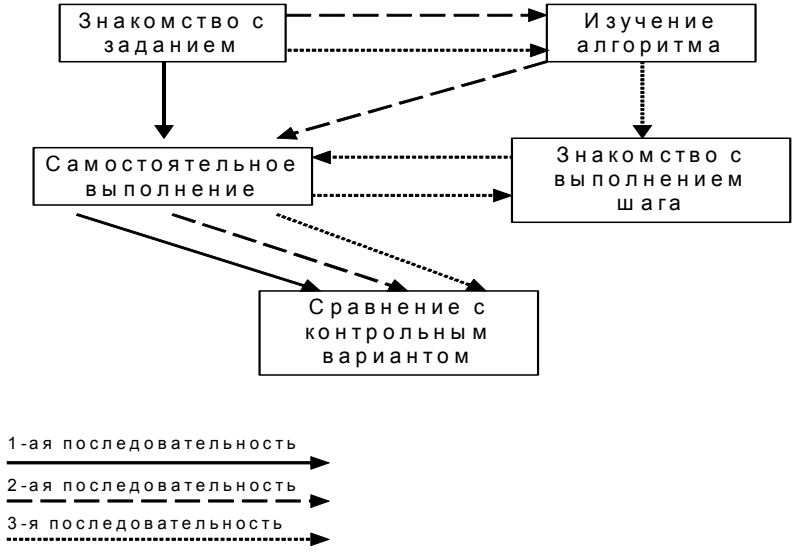

Рис. 4. Возможные последовательности выполнения практических заданий.

4. **Консультации** специалистов (индивидуальная форма обучения) используются как сопутствующая форма обучения, направленная на разъяснение отдельных вопросов, а также углубленное изучение предмета.

3.2.3. Обучение в деятельностном режиме

При выборе способа обучения учитывалось то, что участники тренинга по роду своей основной работы участвуют в различных проектах, которые требуют творческого мышления, коммуникационной и кооперативной активности.

Поэтому при прохождении тренинга решено было создать знакомую и привычную среду для работы.

При *активном способе обучения* слушатели на всех этапах занятия включаются в активную познавательную деятельность, самостоятельно открывают для себя те знания, которые при традиционном подходе излагаются преподавателем без активного участия самого слушателя [7]. Известно, что при деятельностном режиме обучения усваивается от 75% до 90% материала. (Для сравнения: при прочтении традиционной лекции усваивается около 20% изложенного [8]).

Для реализации такого подхода используются следующие приемы работы:

1. Работа с рабочими материалами (*портфолио*) для самостоятельного отслеживания освоения учебного предмета и формулировки возникающих вопросов. Применялась на лекциях как форма организации индивидуальной работы слушателей.
2. Организация групповой работы на лекциях. Использовалась для обсуждения поставленных проблем, принятия решений и для определения в конце учебного дня общих невыясненных вопросов [9].
3. Индивидуальная система практических заданий, которая позволяет каждому слушателю построить собственную траекторию освоения практической части курса.
4. Применение на лекциях *опережающих заданий*, в рамках которых слушателям предлагается найти решение проблемы в процессе индивидуальной работы, обсуждения в парах или группах еще до получения основного материала курса по заданной теме, после чего организуется дискуссия по предложенной проблеме, а после получения нового материала проводится анализ её хода и результатов.
5. Использование одной из базовых моделей активного обучения: вызов-осмысление-рефлексия.

Вызов — процесс актуализации знаний слушателей по данному вопросу и формирование интереса к его изучению.

Осмысление — получение новой информации или идей.

Рефлексия — стадия включения нового знания в собственную систему знаний.

В качестве вызова, например, используется первая традиционная лекция, в которой актуализируются имеющиеся знания и формируются «зоны непонимания», которые будут ликвидироваться в процессе последующего обучения.

Вызов подготавливает, настраивает на ту информацию и на тот процесс, которые будут излагаться на следующих этапах работы.

Этап *осмысления* предполагает предоставление новой информации. Осознание нового осуществляется только в активной деятельности познающего, поэтому создаются специальные условия для активного включения слушателя в процесс усвоения новой информации. Хорошо зарекомендовали себя два

приема работы: INSERT (разметка текста и занесение результатов чтения в специальную таблицу) и «Зигзаг1» [14].

Рассмотрим пример работы с приемом «Зигзаг1». Текст инструктивной лекции «Тестирование функций, поведение которых зависит от истории взаимодействия системы с окружением, с обобщением состояния» разбивается на число частей, равное числу малых групп в аудитории[*]. Каждая группа получает одну часть текста и задание представить его в форме рисунка или схемы. Затем каждая группа представляет свой рисунок аудитории и отвечает с его помощью на возникающие вопросы. После представления работы каждой группы все слушатели получают весь текст лекции. Полнота восприятия информации в данном случае обеспечивается за счет неоднократного прочтения предложенного текста и представления его в другой форме (резко отличающейся от первоначальной), что позволяет задействовать разные каналы восприятия информации у человека [10].

На Рис. 5 и Рис. 6 показаны две формы представления одной и той же информации при перекрестном чтении текста.

Кроме того, для корректной работы обходчика требуется, чтобы при всех вызовах любой целевой функции группы с одними и теми же параметрами в одном и том же состоянии система переходила в одно и то же состояние (требование детерминированности). Но группа функций с состоянием, определенным в спецификациях, не всегда обладает этим свойством.

Эти проблемы решаются при помощи введения отношения эквивалентности на множестве состояний системы. Каждый класс эквивалентных состояний, порождаемый данным отношением эквивалентности, называется обобщенным состоянием.

Использование обобщенных состояний вместо обычных состояний системы позволяет обходчику сократить количество необходимых шагов тестирования до разумных размеров, а также избавиться от недетерминированности. С другой стороны, за это приходится расплачиваться некоторым снижением подробности тестирования: вместо тестирования каждой функции в каждом достижимом состоянии системы, обходчик будет тестировать каждую функцию в каждом достижимом обобщенном состоянии.

Под тестированием функции в обобщенном состоянии системы понимается последовательность вызовов этой функции в тот момент, когда система находится в некотором состоянии, принадлежащем данному обобщенному состоянию. При этом, как и в случае с обычными состояниями, обходчик вызывает каждую функцию с такими наборами параметров, чтобы в каждом состоянии попасть во все допустимые в нем ветви функциональности данной функции.

Рис. 5. 1-ая форма. Часть текста, предложенная группе.

[*] Заметим, что с материалом этой лекции слушатели еще не знакомились.

Заключительным этапом в каждом блоке является рефлексия. Рефлексия состоит в осознании способов деятельности, обнаружении ее смысловых особенностей и выявлении образовательных приращений обучающегося.

Использовались следующие формы рефлексии:

- Устное обсуждение с экспертами и обучающимися.
- Письменное анкетирование.
- Заполнение рабочих материалов (портфолио).

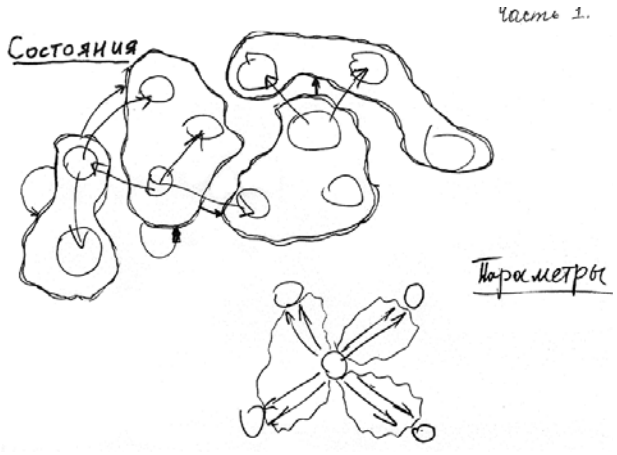

Рис. 6. 2-ая форма. Графическое изображение этого же текста, сделанное группой.

3.2.4. Создание благоприятной образовательной среды

Создание благоприятной образовательной среды — это одна из основных задач тренера. Этой задачей он занимается постоянно и для ее решения используются:

- вводное занятие;
- обратная связь в конце каждого дня;
- адаптация тренинга к конкретной аудитории;
- учет индивидуальных стилей обучения [11].

Цели вводного занятия:

- познакомиться с обучаемыми и дать им возможность познакомиться между собой;
- составить портрет аудитории;
- сделать презентацию курса;
- ознакомить обучаемых с программой курса;
- объяснить работу с портфолио;
- объяснить принципы обратной связи;
- создать благоприятную обстановку для работы.

Основная составляющая деятельностного освоения технологии — это поэтапное выполнение специальных учебных заданий. Успешность упражнений достигается при следующих основных психологических условиях:

- Каждое упражнение должно иметь конкретно определенную цель — чему учащийся должен научиться и как он может проверить это.
- Знания, которые должны осваиваться в упражнении, а также техника его выполнения и предлагаемая последовательность действий должны быть ясно и четко представлены в предоставляемом учащемуся материале.
- Правила выполнения действий, которым нужно научиться, должны осваиваться предварительно, до выполнения задания.
- Задания должны выполняться многократно в целях достижения все возрастающей точности и быстроты выполнения действий, являющихся предметом обучения.
- Хода и результаты действий учащегося должны подвергаться постоянному самоконтролю, а также должны управляться и контролироваться преподавателем.

Все эти психологические условия создаются при проведении упражнений, которые должны включаться в обучение постепенно, в несколько этапов. Для этого была разработана четырехуровневая система упражнений.

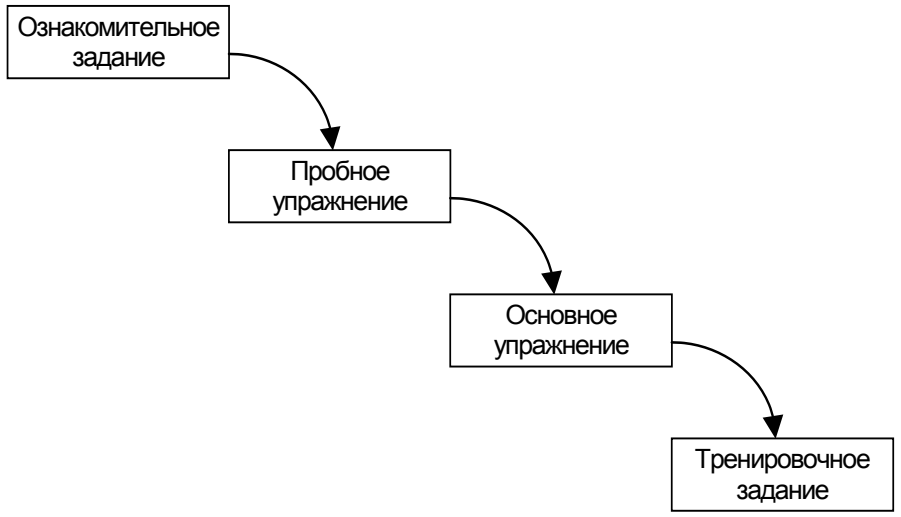

Рис. 7. Система практических заданий тренинга.

Ознакомительное упражнение проводится с целью создания ясных представлений о правильной последовательности действий путем ее показа слушателям на примере и мысленного воспроизведения ими предстоящих действий. Ознакомительное упражнение выполняет тренер во время инструктивной лекции и, тем самым, дает слушателям первые сведения о предстоящей работе.

Пробное упражнение проводится для усвоения правильной последовательности действий. После ознакомительного упражнения, когда у слушателя сложится ясное представление о всем процессе работы, в пробном упражнении слушатель пробует выполнить его самостоятельно. Пробные упражнения выполняются до их безошибочного выполнения.

Пробное упражнение является репродуктивным и проводится для отработки *одного* навыка, показанного и теоретически обоснованного в ознакомительном задании. Пробное задание может быть только учебным и должно отрабатывать навык в ситуации, которая на практике не встречается. Когда навык освоен, можно переходить к следующему уровню заданий.

Основное упражнение проводится с целью использования нескольких частных навыков для решения некоторой задачи. Основное упражнение строится по принципу добавления одного вновь полученного навыка ко всем предыдущим. Поэтому во время прохождения курса сложность и объем основных упражнений возрастает, так как вновь полученные навыки добавляются к уже освоенным.

Тренировочные упражнения наиболее приближены к реальной практике. Они основаны на использовании полученных навыков в ситуациях, не всегда повторяющие учебные упражнения. Тренировочные упражнения применяются с целью закрепления навыков, умений и развития способностей к выполнению задач в реальных условиях.

Заключительным этапом работы является выполнение ***учебного проекта***. Цель проекта — написать тестовую систему для реальной программы. Работа выполняется малой группой под руководством одного из членов группы.

Предложенный подход был использован при разработке следующих тренингов:

- инструмент тестирования CTesK [12];
- инструмент тестирования J@T [13];
- инструмент тестирования J@T-C++Link [13];
- инструмент тестирования Ch@se;
- инструмент для создания генераторов тестов для автоматического тестирования анализирующих и оптимизирующих модулей компиляторов ОТК.

Участники тренингов имели общую математическую подготовку в объеме ВУЗа, знание базового для технологии языка программирования, соответственно, C или Java, опыт разработки, отладки, тестирования программ.

Перед началом одного из тренингов его участники высказали следующие ожидания от курса:

«Овладение технологией тестирования в объеме ее дальнейшего доосмысления и использования»

Дмитрий М.

«Прослушать курс по технологии тестирования. Получить возможность посмотреть, что это такое на реальном проекте, оценить, насколько ее можно использовать»

Елена С.

Итоговые оценки курса:

«Тренинг достиг своей цели. После него я реально представляю как и где можно использовать инструмент тестирования»

Дмитрий М.

«Мне курс понравился. Результат — это получение некоторого представления о технологии тестирования. Технология интересная, ее стоит внедрять на предприятиях и использовать».

Елена С.

4. Заключение

Описанная методика была реализована при разработке тренингов по всем инструментам семейства UniTesK: CTesK [12], J@T, J@T-C++ link [13], Ch@se, ОТК. Тренинги проводились в следующих организациях:

- Lanit-Tercom (Санкт-Петербург, Россия);
- Systematic Software Engineering (Оденсе, Дания);
- ГосНИИАС (Москва, Россия);
- Saarland University (Саарбрюкен, Германия);
- ATS India (Ченнай, Индия);
- Luxoft (Москва, Россия);
- Intel Technologies Inc. (Нижний Новгород, Россия),

а также в исследовательских группах ИСП РАН и в Московском Государственном университете.

Опыт показал, что большая часть слушателей полностью справляется с поставленными задачами и овладевает инструментами в такой степени, что сразу после тренинга может его использовать самостоятельно, практически без консультаций с экспертами.

Из трудностей, которые еще предстоит преодолеть, можно назвать проблему стартовой фазы тренинга, в течение котор ой аудиторию приходится настраивать на активные методы обучения, что многим, привыкшим к академическому формату, дается с большим трудом. Проблемой является и длительность тренинга. Хотя сейчас удалось сократить длительность обучения с 2-3 месяцев до 4-5 дней — это все равно много для большинства коммерческих

компаний. В дальнейшем для решения этой проблемы предполагается разбить тренинг на части и выделить специальные блоки, адресованные разным целевым группам: менеджерам, архитекторам, разработчиками ПО, тестировщикам и т.д.

Разработанные способы обучения в тренинге возможно использовать и в университетских курсах. Из-за большого числа студентов на лекции применение активных методов работы, которые требуют обсуждений в группах, высказывания каждым студентом своего мнения, защиты проектов, весьма ограничены. Практикум можно полностью заимствовать из тренингов для промышленных предприятий, что обеспечивает хорошее и быстрое усвоение материала и получение практических навыков тестирования. Такая организация практических занятий также учит студентов самостоятельно принимать решения на основе анализа многих факторов.

Опыт общения группы RedVerst с потенциальными и действительными пользователями технологии UniTesK показывает, что успешное внедрение передовых методов разработки ПО, в особенности основанных на формальных методах и других техниках, доступных обычно для людей с университетским математическим образованием и серьезными навыками оперирования абстрактными сущностями, требует повышенного внимания именно к организационным и общекультурным аспектам их использования в индустрии.

Наш опыт взаимодействия с разработчиками ПО и тестов, работающими в разного рода организациях, производящих ПО, показал необходимость использования различных форм обучения для решения различных задач, возникающих при внедрении. Для подготовки кадров можно использовать традиционные академические курсы, подкрепленные практической работой по изучаемым технологиям на примерах реального ПО. Для эффективного обучения конечных пользователей больше подходят обучающие тренинги, основанные на активном подходе к обучению и полной погруженности учащихся в тренинг. Для обучения руководителей нужны специальные тренинги, дающие навыки управления проектами, работы в которых ведутся по новым технологиям, предлагающие наборы метрик, лучше традиционных подходящие для оценки состояния процесса разработки, готовности и качества результирующего продукта. Кроме того, необходимы особые формы кратко-срочных тренингов, презентаций, учебных пособий и вспомогательных материалов для широкой пропаганды передовых методов разработки ПО и создания условий для их более осмысленного усвоения промышленностью.

Хотя остается много открытых вопросов, связанных с выбором конкретных форм обучения, наиболее подходящих в том или ином случае, для специализированных тренингов руководителей разного уровня, в целом рассмотренный подход, использующий обучение для решения проблем организационного и культурного плана, возникающих при внедрении передовых методов разработки ПО в индустриальное производство, продемонстрировал свою эффек-

тивность. Нет сомнений и в том, что этот успех не является особенностью технологии UniTesK, которая была выбрана в качестве предмета обучения.

Литература

1. http://www.ispras.ru/groups/rv/rv.html
2. The RAISE Language Group. The RAISE Specification Language. Prentice Hall Europe, 1992.
3. В. В. Кулямин, А. К. Петренко, А. С. Косачев, И. Б. Бурдонов. Подход UniTesK к разработке тестов. Программирование, 29(6), 2003.
4. А. Баранцев, И. Бурдонов, А. Демаков, С. Зеленов, А. Косачев, В. Кулямин, В. Омельченко, Н. Пакулин, А. Петренко. Подход UniTesK к разработке тестов: достижения и перспективы. Труды ИСП РАН, №5, 2004.
5. http://www.ispras.ru/groups/rv/tutorial.html
6. Хуторской А. В. Современная дидактика. Питер, 2001.
7. Педагогика. Педагогические теории, системы, технологии. Под ред. С. А. Смирнова. Москва, 2000.
8. J. Hartley and I. K. Davies. Note-taking: A critical review. Programmed Learning and Educational Technology, 15, 207-224 (1978).
9. Л. Рай. Упражнения: схемы и стратегии. Пер. с анг. Питер, 2003.
10. Петренко О. Л., Прокофьева Л. Б. и др. Технологии открытого обучения. Москва, 2002.
11. К. Торн, Д. Маккей. Тренинг. Настольная книга тренера. Пер. с анг. Питер, 2001.
12. http://www.unitesk.com
13. http://www.atssoft.com
14. Ученик и учитель: возможность диалога и понимания. Сборник статей, составители Е.А. Генике, Е.А. Трифонова, т. 1, Москва, 2002 г.
15. D.W. Johnson, R.T. Johnson, and K.A. Smith, Active Learning: Cooperation in the College Classroom, 2nd Edn., Edina, MN, Interaction Book Company, 1998.