На правах рукописи

Воинов Никита Владимирович

Методы генерации тестовых сценариев на основе структурированных UCM-моделей проектируемой системы

Специальность 05.13.11 –
Математическое и программное обеспечение
вычислительных машин, комплексов и компьютерных сетей

Автореферат

диссертации на соискание ученой степени кандидата технических наук

Санкт – Петербург - 2011

Работа выполнена в Государственном образовательном учреждении высшего профессионального образования «Санкт-Петербургский государственный политехнический университет».

Научный руководитель – кандидат технических наук, профессор
Котляров Всеволод Павлович

Официальные оппоненты – доктор технических наук, профессор
Никифоров Виктор Викентьевич
– доктор технических наук, профессор
Сафонов Владимир Олегович

Ведущая организация – Федеральное государственное унитарное предприятие «Научно-производственное объединение «Импульс»

Защита состоится «12» мая 2011 г. в 16[00] часов на заседании диссертационного совета Д 212.229.18 при ГОУ ВПО «Санкт-Петербургский государственный политехнический университет» по адресу: 195251, Санкт-Петербург, Политехническая ул., д.29, 9 уч. корп., ауд. 325.

С диссертацией можно ознакомиться в Фундаментальной библиотеке ГОУ ВПО «Санкт-Петербургский государственный политехнический университет».

Автореферат разослан « » 2011 г.

Ученый секретарь
диссертационного совета Васильев А.Е.

Разрешаю печать автореферата. Ученый секретарь Д 212.229.18 Васильев А.Е.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность работы. Из практики промышленного производства программного обеспечения (ПО) известно, что стоимость исправления ошибок в пределах жизненного цикла продукта экспоненциально растет по мере отдаления момента обнаружения дефекта, вызванного какой-либо ошибкой, от момента ее внесения. Отсюда, самой эффективной стратегией является поиск и исправление как можно большего числа ошибок на самой ранней фазе разработки требований и спецификаций. Отвечая этой тенденции, в последнее время активно развиваются методы и инструменты тестирования на основе моделей.

Анализ предметной области показал, что на данный момент большинство промышленных инструментов тестирования на основе моделей в качестве формальной нотации используют такие языки, как UML, SDL и конечные автоматы. Безусловно, перечисленные нотации, например, UML, предлагают достаточно наглядную и удобную форму представления поведения разрабатываемой системы, однако требуют, во-первых, элементарных знаний языка UML, его конструкций, видов диаграмм и т.д., во-вторых, детализацию модели уже на самых ранних этапах разработки, когда детали еще неизвестны. Как показывает практика, ранняя детализация моделей в нотациях UML, SDL не очевидна и является слишком низкоуровневой, в частности для контроля заказчиком этапа дизайна проектируемой системы. Поэтому не всегда согласование заказчиком и исполнителем использованного подхода в дизайне системы проходит гладко и эффективно, что приводит к последующим изменениям системы и как следствие к дополнительным расходам на разработку ПО.

В связи с этим, актуальной задачей является усовершенствование общепринятой технологии тестирования на основе моделей для контролируемого заказчиком процесса получения тестовых сценариев на проектируемую систему. Процесс станет более эффективным, если у заказчика и исполнителей проекта появится возможность в понятных обеим сторонам терминах согласовывать действия и получаемые результаты на каждом этапе, включая преобразование исходных требований в формальное представление, выработку критериев покрытия требований, оценку и анализ результатов покрытия и т.д. Для этого автором предлагается использовать две формальные модели: одну – для доступности и прозрачности описания поведения на уровне исходных требований, другую – для его последующего анализа и трансляции, в частности, в набор тестов. Создание двух формальных моделей вместо одной увеличивает общее время дизайна, но, с другой стороны, сокращает время автоматического преобразования высокоуровневой модели в формальную модель, по которой осуществляется генерация тестовых сценариев. Логично предположить,

что время ручного создания высокоуровневой формальной модели значительно меньше времени создания формальной модели для верификации и генерации тестов, поскольку соответствующая нотация выбирается таким образом, чтобы максимально уменьшить трудоемкость ручного создания модели. Более того, поскольку эта модель создается разработчиками в тесном сотрудничестве с заказчиком, владеющим знанием предметной области, то ошибки разработчика, возникающие из неверной интерпретации поведения модели, находятся и исправляются достаточно быстро.

Настоящая работа посвящена разработке методов генерации тестовых сценариев на основе высокоуровневых UCM-моделей проектируемой системы. В качестве базового инструментария в работе используется технология VRS/TAT (Verification of Requirements Specifications (VRS) – инструментарий верификации требований; Test Automation Toolset (TAT) – инструментарий автоматизации тестирования), которая является одним из наиболее перспективных современных инструментов верификации и автоматизации тестирования на основе моделей. Реализованные в работе методы позволили сократить трудоемкость процесса разработки тестовых сценариев путем автоматизации ручных этапов, устранив следующие недостатки и ограничения в инструментарии VRS/TAT: трудоемкая ручная формализация модели базовых протоколов; отсутствие методов структурирования модели для возможности ее анализа по частям и на разных уровнях абстракции; генерация трасс осуществляется по всему дереву поведения модели, что приводит к большому количеству ненужных трасс и длительному времени генерации; отсутствует возможность отслеживать соответствие между отдельными требованиями и элементами создаваемой модели и как следствие сложность определения покрытия требований в полученных тестовых сценариях.

Цели и задачи диссертационной работы. Целью диссертационной работы является сокращение трудоемкости процесса разработки тестовых сценариев путем применения высокоуровневых UCM-моделей, контролируемых заказчиком. Сокращение трудоемкости осуществляется автоматизацией следующих этапов: преобразование UCM-моделей в модели базовых протоколов; структурирование модели базовых протоколов; поиск определенных сценариев поведения модели; отслеживание соответствия между требованиями и элементами модели; поиск покрытия требований полученными тестовыми сценариями. Для достижения цели в работе решены следующие задачи:

- обоснование выбора высокоуровневой формальной нотации для контролируемого заказчиком описания программной системы на стадии ее дизайна;
- разработка методов и инструментов автоматического преобразования высокоуровневой формальной модели в формальную модель в нотации, с которой

работают инструменты верификации и автоматизации тестирования (например, VRS/TAT);

- разработка методов структурирования формальной модели для возможности пошагового анализа по компонентам и на различных уровнях абстракции;
- формулировка критериев покрытия тестовыми сценариями набора функциональных требований на систему с возможностью отслеживания соответствия между требованиями и элементами модели;
- разработка методов и инструментальных средств построения тестовых сценариев, удовлетворяющих заданному критерию, на основе анализа формальной модели;
- разработка методов и инструментальных средств сокращения полученного набора тестовых сценариев;
- интеграция инструментальных средств, реализующих разработанные методы, в технологическую цепочку VRS/TAT;
- применение предложенных инструментальных средств в составе технологической цепочки VRS/TAT в крупных промышленных телекоммуникационных проектах.

Предметом исследования являются методы и инструментальные средства тестирования на основе моделей.

Методы исследования. Для решения поставленных в работе задач используются теория инсерционного программирования, аппарат формальных спецификаций. Применяются стандарты языков Use Case Maps (UCM) и Message Sequence Charts (MSC).

Обоснованность и достоверность полученных результатов обеспечивается корректным использованием теории инсерционного программирования; использованием аппарата формальных спецификаций; положительными итогами использования разработанных методов и программных средств в четырех индустриальных проектах с помощью технологии VRS/TAT.

Научные результаты и их новизна.

- Разработан подход к процессу генерации тестовых сценариев на основе двух формальных моделей, новизна которого заключается в одновременном применении следующих нотаций: высокоуровневой нотации UCM, позволяющей контролировать заказчиком описание поведения системы и согласовывать с ним поведенческие сценарии, и нотации базовых протоколов для последующего создания тестовых сценариев по модели базовых протоколов.
- Разработан метод автоматизированной генерации базовых протоколов, новизна которого заключается в трансформации UCM-модели в структурированную модель

базовых протоколов. Данный метод позволяет заменить ручное создание базовых протоколов их генерацией по высокоуровневой UCM-модели и анализировать поведение модели по компонентам и на различных уровнях абстракции.

- Разработаны методы создания эвристик, новизна которых заключается в автоматизированном построении цепочек базовых протоколов, направляющих поиск определенных сценариев в дереве поведения модели базовых протоколов, что помогло сократить время получения тестовых сценариев при работе с крупными проектами.
- Впервые применен критерий покрытия требований, новизна которого заключается в выявлении последовательностей наблюдаемых событий одновременно как в терминах исходных требований, так и в терминах соответствующих им элементов формальной модели базовых протоколов.
- Для применения критерия цепочек наблюдаемых событий были разработаны новые методы отслеживания соответствия между требованиями и элементами модели базовых протоколов и оценки покрытия требований набором тестовых сценариев по данному критерию, позволяющие определить качество покрытия до этапа создания исполняемых тестов и их исполнения на целевой платформе.

Практическая значимость работы. На базе полученных научных результатов разработан комплекс программных средств, интегрированный в технологию VRS/TAT. Усовершенствованная технология VRS/TAT после интеграции разработанных методов была применена в ряде промышленных телекоммуникационных проектов и доказала свою высокую эффективность для обеспечения проверки качества разрабатываемого ПО, позволив сократить более чем на 60% трудоемкость разработки тестовых сценариев по сравнению с существующим подходом.

Апробация работы. Основные положения и результаты диссертационной работы доложены и обсуждены на международных научных конференциях “The Third Spring Young Researchers’ Colloquium on Software Engineering” (Moscow, 2009), Motorola Technology Day (SPb, 2007, 2010), “Технологии Microsoft в теории и практике программирования” (СПб, 2008, 2009, 2010), XXXVI неделя науки СПбГПУ (СПб, 2007), XXXIX неделя науки СПбГПУ (СПб, 2010).

Публикации. Основные положения диссертации изложены в 9 печатных работах, в том числе в двух работах в журналах из перечня ВАК.

Внедрение. Разработанные методы внедрены в компаниях ЗАО ”Моторола ЗАО”, ООО “ИЦ “Северо-Западная лаборатория” и использованы при разработке учебно-

методического комплекса СПбГПУ по курсам “Технология разработки программного обеспечения” и “Индустриальные технологии разработки ПО” на кафедре “Информационные и управляющие системы”. Практическое использование представляемых на защиту результатов подтверждено соответствующими актами о внедрении.

Структура и объем работы. Диссертация состоит из введения, четырех глав, заключения, списка литературы и трех приложений. Общий объём диссертации с приложениями – 169 страниц машинописного текста, содержит 62 рисунка, 11 таблиц, список литературы содержит 125 наименований.

СОДЕРЖАНИЕ РАБОТЫ

Введение. Во введении показана актуальность темы диссертации, определены цели и задачи проведенных исследований, отражена научная новизна и практическая значимость полученных результатов, приведены сведения о реализации работы, об апробации, о публикациях и структуре диссертации.

В первой главе рассмотрены известные подходы к улучшению качества ПО, приведена классификация видов тестирования, проведен обзор 11 инструментов тестирования, используемых в практике создания промышленного ПО, проанализированы их достоинства и недостатки. Сделан вывод, что несмотря на ряд преимуществ тестирования на основе моделей по сравнению с традиционным тестированием, широкому применению данного подхода препятствует сложность и недостаток навыков исполнителей проектов в использовании формальных методов.

Рассмотрены особенности тестирования на основе моделей, проведен обзор методов создания тестовых сценариев на основе моделей, проанализированы достоинства и ограничения применимости данного подхода.

Проведен сравнительный анализ следующих инструментов тестирования на основе моделей, являющихся наиболее распространенными на рынке и информация по которым доступна в открытых источниках: GOTCHA-TCBeans, mbt, MOTES, TestOptimal, AGEDIS, ParTeG, Qtronic, Test Designer, Spec Explorer, UniTESK, инструментарий верификации и автоматизации тестирования VRS/TAT.

На основе проведенного анализа были сделаны следующие выводы по состоянию дел в области тестирования на основе моделей:

- Рассмотренные инструменты не используют высокоуровневые формальные нотации.
- Большинство изученных инструментов не поддерживает полный процесс проверки качества ПО на основе моделей, состоящий из создания модели, ее верификации,

построения и исполнения тестов, анализа результатов тестирования, ограничиваясь реализацией только отдельных этапов этого процесса.

- По результатам сравнительного обзора современных инструментов тестирования на основе моделей инструментарий VRS/TAT был оценен как наиболее перспективный. Он объединяет целый набор методов и инструментов в единую технологическую цепочку, в которой каждый из инструментов реализует отдельный этап в рамках процесса создания модели, ее верификации, тестирования и анализа результатов тестирования.
- Технология VRS/TAT имеет следующие ограничения и недостатки:
    - трудоемкая ручная формализация модели базовых протоколов, которая к тому же сложна для согласования с заказчиком;
    - отсутствие методов структурирования модели для возможности ее анализа по частям и на разных уровнях абстракции, что затрудняет применение технологии при работе с крупными проектами;
    - генерация трасс осуществляется по всему дереву поведения модели, что приводит к большому количеству ненужных трасс и длительному времени генерации;
    - отсутствует возможность отслеживать соответствие между отдельными требованиями и элементами создаваемой модели;
    - как следствие предыдущего пункта – сложность определения покрытия требований в полученных тестовых сценариях, созданных по модели;
    - отсутствует возможность сокращения полученного набора трасс, поэтому исполняемый тестовый набор может содержать большое число избыточных тестов, покрывающих требования, уже покрытые другими тестами.

Во второй и третьей главах представлены концепция, реализация и интеграция в технологию VRS/TAT методов генерации тестовых сценариев на основе высокоуровневых UCM-моделей, позволивших сократить трудоемкость процесса получения тестовых сценариев путем автоматизации отдельных этапов технологической цепочки и тем самым устранить отмеченные выше недостатки данной технологии.

В технологии VRS/TAT в качестве высокоуровневой формальной нотации была выбрана нотация UCM для описания последовательности действий системы в ответ на внешние воздействия пользователей или программных систем. Варианты использования отражают функциональность системы с точки зрения описания ее архитектуры. Дизайн системы в нотации UCM представляет собой набор взаимодействующих между собой

диаграмм. В свою очередь, каждая из диаграмм сосредоточена на описании взаимодействия компонентов (агентов, процессов системы), объектов, наблюдателей и подсистем. Каждый компонент или подсистема содержит элементы ответственности (Responsibilities), соответствующие событиям в системе, а также упорядоченную последовательность их возникновения. Таким образом, совокупность компонентов и диаграмм дает пользователю наглядное представление поведения системы и взаимодействий между ее компонентами. На рис. 1 изображена одна из UCM-диаграмм проекта автомобильного радио, на которой описано взаимодействие двух компонентов: пользователя (User) и радио (Radio). Сценарии взаимодействия изображены линиями поведения, нагруженными элементами UCM (например, Responsibility, Timer, Stub), порядок следования которых определен направленностью линий. При нажатии пользователем кнопки включения радио (событие Button_On) радио начинает работу (событие Radio_On), происходит инициализация настроек (поведение описывается на диаграмме Stub_Initialize). После этого пользователю становятся доступны следующие функции работы с радио: изменение громкости, частоты, переключение диапазонов, автопоиск станций, сохранение станций в памяти радио. Сценарии поведения при выборе каждой из функций описываются на отдельных UCM-диаграммах соответствующих элементов Stub. При нажатии пользователем кнопки выключения радио (событие Button_Off) радио завершает работу (событие Radio_Off).

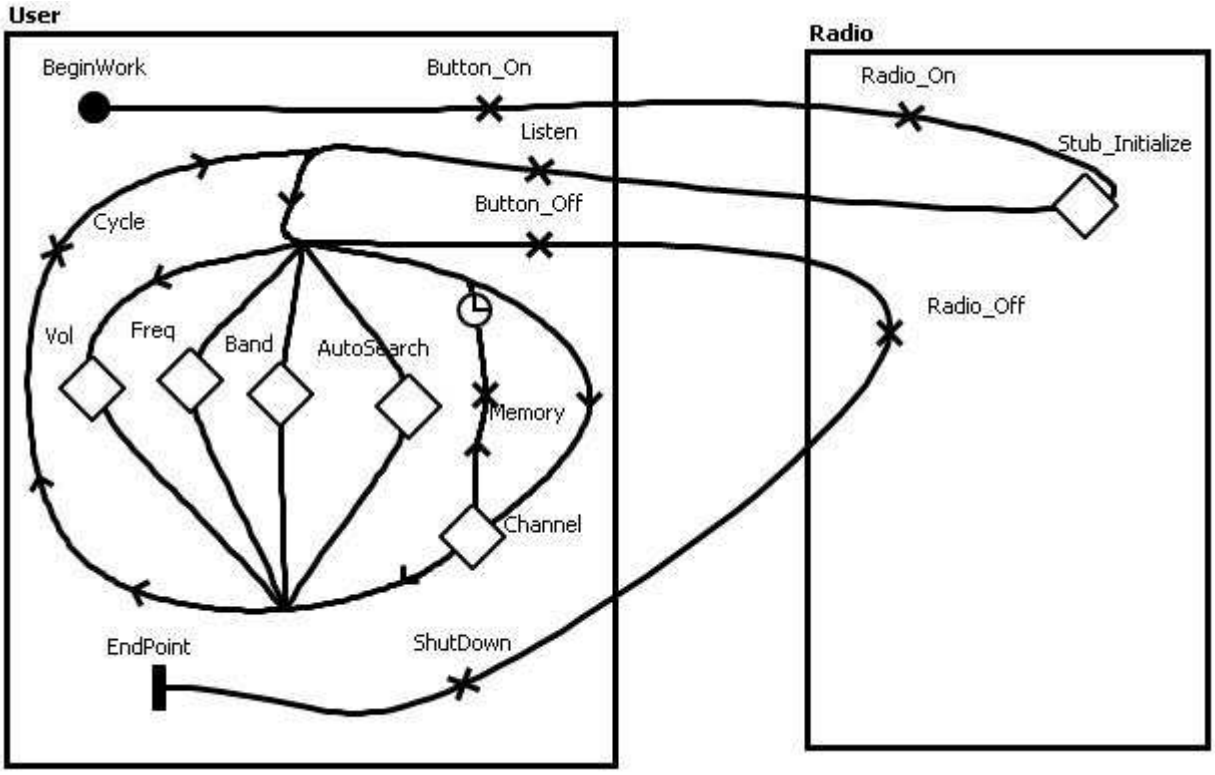

Рис. 1. UCM-диаграмма проекта автомобильного радио

Инструменты верификации и автоматизации тестирования в технологии VRS/TAT работают с формальными моделями, представленными в нотации базовых протоколов на основании принципов инсерционного программирования, разработанного А.А. Летичевским. Базовый протокол описывает переход в модели и определяется тройкой $\alpha \rightarrow < P > \beta$, где α и β – пред- и постусловие соответственно, а P – процесс базового протокола. Условия α и β представляются с помощью логических выражений некоторого базового языка и определяют условия на множестве состояний модели.

Последовательность базовых протоколов (трасса) определяет историю функционирования модели, характеризующую ее поведение: $s_0 \xrightarrow{bp_1} s_1 \xrightarrow{bp_2} s_2 \ldots$, где s_0, s_1 … – состояния модели; bp_1, bp_2 – базовые протоколы. Каждая трасса может быть использована в качестве тестового сценария.

Основные преимущества UCM по сравнению с нотацией базовых протоколов – близость к формулировке исходных требований, понимание и возможность контроля заказчиком, согласование с ним в терминах событий различных режимов поведения разрабатываемой системы. Недостатки – отсутствие инструментальной поддержки проверки корректности создаваемой модели поведения и автоматической генерации из нее тестовых сценариев. Преимущества нотации базовых протоколов – возможность доказательства свойств модели, проверка ее корректности с помощью мощных инструментов верификации VRS и автоматическое создание символических тестовых сценариев по проверенной корректной модели для последующей генерации исполняемых тестов. Недостатки – сложность для создания, визуального восприятия и согласования с заказчиком и специалистами в предметной области. Интеграция двух нотаций их взаимно дополняет, т.е. недостатки одной модели устраняются преимуществами другой. К тому же возможность автоматического преобразования UCM в нотацию базовых протоколов сокращает трудозатраты в случае корректировки исходных требований.

В рамках подхода с двумя формальными моделями в единой технологии тестирования были разработаны методы генерации тестовых сценариев на основе структурированных UCM-моделей проектируемой системы, интегрированные в технологическую цепочку VRS/TAT. Всего можно выделить 6 методов, каждый из которых устраняет одно из сформулированных выше ограничений данной технологии и позволяет сократить трудоемкость процесса получения тестовых сценариев путем автоматизации соответствующего этапа. На рис. 2 приведена схема обновленной технологической цепочки VRS/TAT после интеграции с разработанными методами, выделенными жирными линиями.

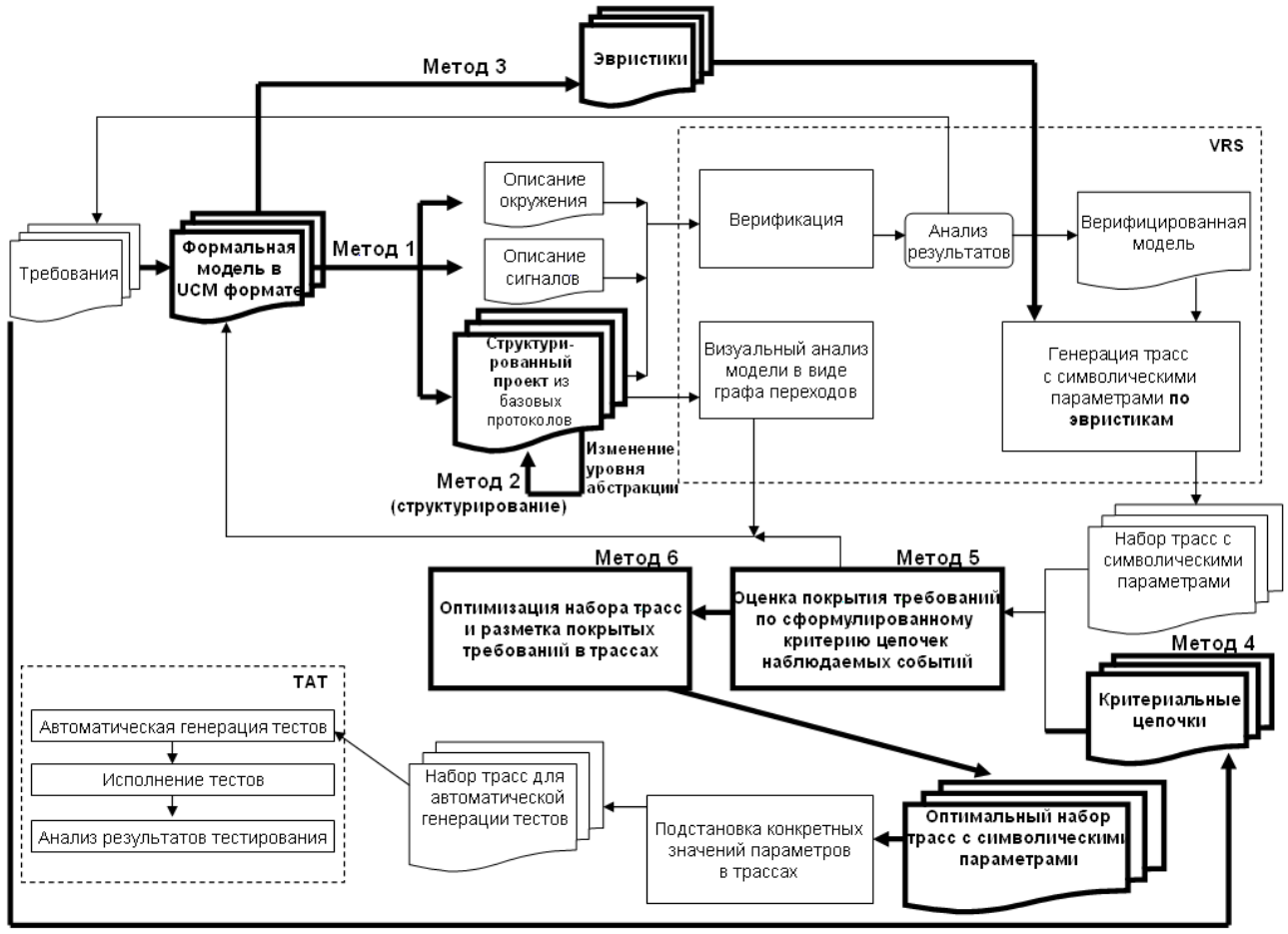

Рис. 2. Внедрение разработанных автором методов в технологическую цепочку VRS/TAT

Метод 1 – автоматическое создание формальной модели базовых протоколов по нотации UCM заключается в преобразовании компонентов и элементов UCM в базовые протоколы. При этом используются метаданные, которые могут быть добавлены практически ко всем элементам UCM-диаграммы, что позволяет перенести всю необходимую информацию из требований в UCM-дизайн. Метаданные представляют собой неограниченный набор расклассифицированных маркерами данных формата: “Маркер” – “Текстовая информация заданного типа”, где каждый маркер соответствует определенному типу информации и управляет созданием определенного элемента базового протокола.

Разработанный автором метод позволяет исключить ручное создание формальной модели в виде набора базовых протоколов, заменив его автоматизированной генерацией базовых протоколов.

Метод 2 – структурирование формальной модели заключается в преобразовании UCM-модели, факторизованной элементами Stub, в модель базовых протоколов, факторизованную расширенными протоколами, интерпретирующими эти элементы Stub.

Когда UCM-диаграмма, изображающая поведение всей системы или ее фрагмента, становится слишком сложной для визуального восприятия и анализа, она структурируется с

помощью элементов Stub. Часть описания поведения системы выносится на отдельную диаграмму, его место занимает элемент Stub и исходная UCM-диаграмма верхнего уровня рассматривается как главная диаграмма, а остальные диаграммы – как поддиаграммы. Поддиаграммы могут также содержать элементы Stub. Чем сложней поведение системы, тем больше будет создаваться уровней вложенности, реализуемых элементами Stub.

В свою очередь, расширенные протоколы укрупняют детализацию описания поведения модели базовых протоколов с помощью замены последовательности базовых протоколов (фрагмента поведенческого сценария) на один расширенный протокол, который описывает результат моделирования поведения заменяемого фрагмента трассы.

При создании базовых протоколов, исходя из соображений детализации, на каждом уровне элемент Stub интерпретируется как расширенный протокол.

Структурирование состоит из трех основных этапов.

1) Для каждой UCM-диаграммы из исходного проекта создается директория в проекте базовых протоколов с соответствующим именем:

$$\forall DIAG_i \in S\_PROJ, DIR_{DIAG_i} = CREATE\_DIR(DIAG_i), DIR_{DIAG_i} \in T\_PROJ$$

где $DIAG_i$ – UCM диаграмма исходного проекта, S_PROJ – исходный проект, DIR_{DIAG_i} – директория целевого проекта, T_PROJ – целевой проект.

В директорию DIR_{DIAG_i} помещаются базовые протоколы, соответствующие элементам Responsibility на исходной диаграмме $DIAG_i$.

2) Если UCM-диаграмма содержит один или несколько элементов Stub, то в соответствующей директории создается директория EP:

$$\forall DIR_{DIAG_i} \in T\_PROJ, DIR_{DIAG\_EP_i} = CREATE\_DIR(DIAG\_EP_i), DIR_{DIAG\_EP_i} \in T\_PROJ$$

где DIR_{DIAG_i} – директория целевого проекта, $DIR_{DIAG\_EP_i}$ – директория целевого проекта внутри директории DIR_{DIAG_i}, T_PROJ – целевой проект.

В директорию $DIR_{DIAG\_EP_i}$ помещаются расширенные протоколы, которые представляют детальное поведение, описываемое внутри элементов Stub, на диаграмме более высокого уровня, т.е. на той диаграмме, где элемент Stub расположен.

3) Если UCM-диаграмма содержит несколько компонентов, то в соответствующей директории DIR_{DIAG_i} создается директория !Connectors:

$$\forall DIR_{DIAG_i} \in T\_PROJ, DIR_{DIAG\_CONN_i} = CREATE\_DIR(DIAG\_CONN_i), DIR_{DIAG\_CONN_i} \in T\_PROJ$$

где DIR_{DIAG_i} – директория целевого проекта, $DIR_{DIAG\_CONN_i}$ – директория целевого проекта внутри директории DIR_{DIAG_i}, T_PROJ – целевой проект.

В директорию $DIR_{DIAG\_CONN_i}$ помещаются протоколы-коннекторы, которые сохраняют поток управления при переходах между компонентами. В трассах эти протоколы-коннекторы будут связывать базовые протоколы, полученные по элементам Responsibility, принадлежащим разным компонентам на UCM-диаграмме.

Разработанный автором метод позволяет анализировать поведение модели по частям на различных уровнях абстракции, а не работать со всей моделью сразу.

Метод 3 – автоматическое создание эвристик для генерации по ним тестовых сценариев. Поведение модели S(P), представленной базовыми протоколами, в состоянии α описывается уравнением, выведенным А.А.Летичевским:

$$\mathbf{S}_\alpha = \sum_{p \in P^1(\alpha)} \mathbf{proc}(p) * (\mathbf{T}(\alpha, p) : \Delta) * \mathbf{S}_{\mathbf{T}(\alpha,p)} + \sum_{p \in P^0(\alpha)} \mathbf{proc}(p) * (\mathbf{T}(\alpha, p) : \Delta) * (\mathbf{S}_{\mathbf{T}(\alpha,p)} + \Delta)$$

где $\mathbf{proc}(p)$ - процесс базового протокола; $\mathbf{T}(\alpha, p) = \mathbf{Tr}(\alpha, \mathbf{post}(p))$ - предикатный трансформер, позволяющий учитывать изменение состояния модели после применения базового протокола; $\mathbf{post}(p)$ - постусловие базового протокола.

Графически данное уравнение представляет собой пучок трасс, в котором каждое следующее состояние системы раскладывается в новый пучок трасс. Поведение системы описывается пучками трасс, в которых переходы являются базовыми протоколами. Если представить дерево поведения высоты L, вершинами которого являются состояния системы S, а стрелки, выходящие из вершины, помечены базовыми протоколами, которые могут быть применимы к соответствующему состоянию, то количество трасс будет равно количеству листьев этого дерева. Например, пусть система описана N базовыми протоколами и в каждом состоянии системы (на каждом шаге поведения) может примениться любой из этих N базовых протоколов, тогда общее количество трасс оценивается экспонентой $\mathbf{N^L}$. Даже в случае относительно небольшой системы, формализуемой N=100 базовыми протоколами, общее количество трасс длины L=10 базовых протоколов будет равно $\mathbf{100^{10}}$. Задача полного обхода такого дерева поведения нереальна, также как и получение и исполнение набора тестов для всех поведений системы. Для выбора среди них тех, которые считаются (по согласованию с заказчиком) основными, критическими, покрывающими наиболее важные сценарии функционирования системы, А.В.Колчиным был разработан язык эвристик, позволяющий пользователю задавать направление поиска трасс для тестовых сценариев при обходе пространства поведения модели. Таким образом, существенно ограничивается пространство поиска от одной промежуточной цели до другой, т.е. от одного события в тестовом сценарии до другого. При использовании эвристик количество получаемых трасс оценивается с помощью формулы:

$$N_0^{k_1} + N_1^{k_2-k_1} + N_2^{k_3-k_2} + ... + N_i^{L-k_i},\ k_i \le L, N_i \le N$$

где k_i – уровень глубины дерева поведения, на котором задан элемент, через который должен пройти трассовый генератор, N_i - количество базовых протоколов, применимых между двумя промежуточными точками, i – количество заданных промежуточных точек. Таким образом, чем подробней эвристика, тем быстрее будет сгенерирован соответствующий тестовый сценарий.

Однако в случае крупных моделей ручное создание эвристик является трудоемким процессом. Поэтому автор разработал методы автоматизированного построения эвристик, на основании которых генерируются трассы, имеющие смысловую нагрузку и удовлетворяющие определенному критерию.

Метод создания эвристик по набору MSC-диаграмм.

1) По UCM-диаграммам автоматически создаются наборы MSC, каждая из которых описывает одно из возможных поведений на UCM-диаграмме.

2) Каждая эвристика формируется в соответствии с последовательностью базовых протоколов в одной MSC-диаграмме.

3) По эвристикам создается файл с эвристиками:

$\forall MSC_i \in MSC\_DIR_j, GUIDE_{MSC_i} = CREATE\_GUIDE(MSC_i), GUIDE_{MSC_i} \in GUIDES\_FILE_{MSC\_DIR_j}$

где MSC_i – MSC-диаграмма из директории MSC_DIR_j, $GUIDE_{MSC_i}$ – эвристика, полученная по диаграмме MSC_i, $GUIDES\_FILE_{MSC\_DIR}$ – файл с эвристиками по всем MSC-диаграммам директории MSC_DIR_j.

Таким образом, количество эвристик будет равняться количеству MSC-диаграмм, т.е. числу возможных поведений на каждой UCM-диаграмме.

Метод создания эвристик по набору MSC-диаграмм и критериальным цепочкам (Критериальная цепочка – последовательность наблюдаемых событий, исполнение которой утверждает о выполнении соответствующего требования, описанного данной цепочкой)**.**

1) Для каждой критериальной цепочки из файла с цепочками по всей структуре директорий с MSC-диаграммами, полученными по целому UCM-проекту, осуществляется поиск последовательности базовых протоколов, указанной в цепочке.

2) Если последовательность будет найдена, то сгенерируется эвристика, соответствующая данной критериальной цепочке, иначе соответствующая эвристика сгенерирована не будет:

$$\forall CHAIN_i \in FILE\_CHAINS:$$
$$IF(CHAIN_i \in \{MSC_i,...,MSC_j\}), THEN(GUIDE_{CHAIN_i} = CREATE\_GUIDE(CHAIN_i)),$$
$$GUIDE_{CHAIN_i} \in GUIDES\_FILE_{FILE\_CHAINS}$$

где $CHAIN_i$ – критериальная цепочка из файла с цепочками $FILE\_CHAINS$, $GUIDE_{CHAIN_i}$ – эвристика, соответствующая цепочке $CHAIN_i$, $GUIDES\_FILE_{FILE\_CHAINS}$ – файл с эвристиками по всем критериальным цепочкам, которые были найдены во всех MSC-диаграммах $\{MSC_i,...,MSC_j\}$, полученных по UCM-проекту.

3) При этом в методе поддерживается структурирование модели: если базовые протоколы цепочки находятся на UCM-диаграммах разных уровней абстракции и, соответственно, в MSC-диаграммах, расположенных в различных директориях, то данный факт будет учтен, и поиск выполнится на всех уровнях абстракции.

Инкрементальный метод создания эвристик.

1) Для каждой MSC-диаграммы, созданной как для UCM-диаграммы верхнего уровня, так и для UCM-диаграмм элементов Stub, в терминах исходных требований формулируется поведение, покрываемое в данной MSC-диаграмме.

2) Пользователь выбирает детальность, которая ему необходима в тестовом сценарии, и отмечает отдельным идентификатором те уровни детализации и поведения модели на этих уровнях, которые он хочет увидеть в эвристике и, соответственно, в трассе, которая в дальнейшем будет получена по данной эвристике.

3) По поведениям, отмеченным одинаковым идентификатором на различных уровнях детализации модели, будет получена одна эвристика:

$$\forall ID, ID \in \text{int}:$$
$$IF(DEFINED(\{MSC_i,...,MSC_j\}_{ID} \in \{UCM_k,...,UCM_l\}), \{UCM_k,...UCM_l\} \in S\_PROJ),$$
$$THEN(\exists GUIDE_{ID} : CHAIN_{\{MSC_i,...,MSC_j\}_{ID}} \in GUIDE_{ID})$$

где ID – идентификатор эвристики, $\{MSC_i,...,MSC_j\}_{ID}$ – набор отмеченных поведений, по которым требуется создать эвристику, $\{UCM_k,...UCM_l\}$ – UCM-диаграммы исходного UCM-проекта S_PROJ, $GUIDE_{ID}$ – получаемая эвристика, содержащая поведения, которые описаны в $\{MSC_i,...,MSC_j\}_{ID}$.

Разработанные автором методы автоматического создания эвристик позволили достаточно гибко настраивать пути для направленного поиска определенных сценариев в дереве поведения модели, что помогло сократить время получения тестовых сценариев при работе с крупными проектами.

Метод 4 – отслеживание соответствия между требованиями и элементами модели.

1) Для каждого требования формулируются последовательности наблюдаемых событий, наблюдение которых в указанном порядке свидетельствует о выполнении соответствующего требования.

2) Формулируются цепочки базовых протоколов, описывающих события в формализованной модели, нахождение которых в трассе будет для заказчика свидетельством покрытия требования в трассе.

3) Отслеживается соответствие между событиями, характеризующими требования в терминах исходных спецификаций, и элементами модели (базовыми протоколами):

$$\forall REQ_i \in REQ\_BOOK:$$
$$\forall EVENT_j \in REQ_i, \exists BP_j \in BP\_PROJ$$

где REQ_i – требование из исходного документа требований REQ_BOOK, $EVENT_j$ – наблюдаемое событие, характеризующее требование, BP_j – базовый протокол из проекта базовых протоколов BP_PROJ.

Разработанный автором метод позволяет однозначно определять тестовые сценарии, которые содержат элементы модели, связанные с выбранными требованиями; идентифицировать, каким сценарием покрывается то или иное требование; выявлять требования, которые еще не проверены ни одним сценарием.

Метод 5 – поиск покрытия требований в соответствии с критерием цепочек наблюдаемых событий. Метод заключается в поиске в сгенерированных трассах последовательностей базовых протоколов, описывающих требования. Нахождение последовательности базовых протоколов, соответствующей какому-либо требованию, в трассе означает покрытие данного требования трассой:

$$\forall CHAIN_i \in FILE\_CHAINS, \forall TRACE_j \in TRACES:$$
$$IF(CHAIN_i \in TRACE_j), THEN(REQ_{CHAIN_i} = COVERED_{TRACE_j}),$$
$$IF(CHAIN_i \notin TRACES), THEN(REQ_{CHAIN_i} = \mathrm{UNC}OVERED_{TRACES})$$

где $CHAIN_i$ – критериальная цепочка (последовательность базовых протоколов) из файла с цепочками $FILE\_CHAINS$, $TRACE_j$ – трасса из набора сгенерированных трасс $TRACES$, REQ_{CHAIN_i} – требование, проверяемое критериальной цепочкой $CHAIN_i$, состояние $COVERED_{TRACE_j}$ означает покрытие требования в трассе $TRACE_j$, состояние

$UNCOVERED_{TRACES}$ означает, что требование не было покрыто ни одной из трасс набора $TRACES$.

Разработанный автором метод позволяет в доступной и убедительной для заказчика форме определять степень покрытия требований набором тестовых сценариев по критерию цепочек наблюдаемых событий.

Метод 6 – сокращение тестового набора. Если трасса покрывает требования, уже учтенные в другой трассе, покрывающей большее количество требований, то первая трасса исключается из рассмотрения, таким образом, сокращая итоговый набор тестов:

$$\forall TRACE_i, TRACE_j \in TRACES:$$
$$IF((REQ_{TRACE_i}^{COVERED} \subset REQ_{TRACE_j}^{COVERED}) \wedge (TRACE_j > TRACE_i))$$
$$THEN(TRACE_i \notin TRACES_{OPT})$$

где $TRACE_i, TRACE_j$ – трассы из набора сгенерированных трасс $TRACES$, $REQ_{TRACE_i}^{COVERED}, REQ_{TRACE_j}^{COVERED}$ – требования, покрытые в трассах $TRACE_i, TRACE_j$ соответственно, $TRACES_{OPT}$ – оптимизированный набор трасс.

Разработанный автором метод сокращает итоговое количество тестовых сценариев, покрывающих требования, тем самым уменьшая время на этапе генерации тестов и их исполнения.

В четвертой главе представлены результаты внедрения разработанных методов в технологию верификации и автоматизации тестирования VRS/TAT. Проанализированы показатели эффективности применения в четырех различных проектах.

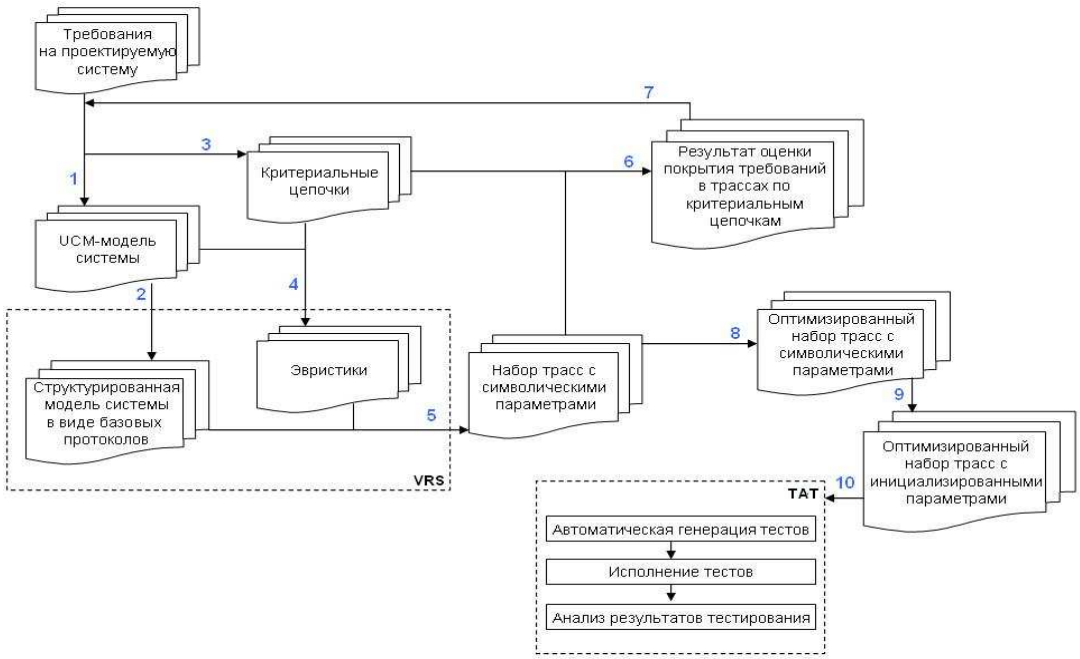

Рис. 3. Обобщенная схема применения разработанных методов

На рис. 3 приведена обобщенная схема применения разработанных методов, которая обеспечивает создание набора тестовых сценариев, покрывающих исходные требования на систему.

По требованиям на проектируемую систему вручную создается UCM-модель системы (1), по которой автоматически генерируется структурированный набор базовых протоколов (2). Для каждого требования вручную создаются критериальные цепочки, т.е. формулируются последовательности событий, выполнение которых в указанном порядке свидетельствует о покрытии соответствующего требования (3). Затем автоматически генерируется файл с эвристиками (4), как на основании критериальных цепочек, так и по MSC-диаграммам, полученным по UCM-модели, в зависимости от выбранного метода генерации. По эвристикам средствами VRS создается набор трасс (тестовых сценариев) с символическими параметрами (5). С помощью методов отслеживания соответствия между требованиями и элементами модели и поиска покрытия требований в соответствии с критерием цепочек выполняется оценка покрытия исходных требований (6). В случае неполного покрытия требований необходим анализ причин наличия непокрытых требований (7). Это может быть результатом некорректного создания UCM-модели или же пробелов в формулировке критериальной цепочки. После коррекции модели или цепочек процесс повторяется с пункта (2) или (4) соответственно. Если степень покрытия требований удовлетворяет пользователя, то применяется метод сокращения набора символических трасс для получения оптимизированного набора (8).

Дальнейшие этапы инициализации символических сценариев конкретными значениями (9) и генерации по инициализированным сценариям исполняемых тестов, исполнения тестов на целевой платформе и анализа результатов тестирования (10) осуществляются средствами инструмента автоматизации тестирования TAT.

Предложенные в работе методы и реализующие их программные средства проверены на работоспособность в четырех реальных промышленных проектах: базовая станция системы, реализующей технологию **CDMA** (характеристики модуля: 148 требований, 205 базовых протоколов, порядка 10^{13} возможных сценариев поведения; всего в проекте порядка 9000 требований); **C_ROUTER** – компонент телекоммуникационной сети, связывающий базовые станции с администратором базовых станций (характеристики компонента: 107 требований, 163 базовых протоколов, порядка 10^{8} возможных сценариев поведения; всего в проекте порядка 5000 требований); модуль **MPM**, решающий проблему совместимости между интерфейсами модемов и контроллера станции приемопередатчика, в рамках проекта WiMAX (характеристики модуля: 51, 283, 10^{10}; всего в проекте порядка 12000 требований);

S_HMI – проект электронного табло в кабине машиниста поезда метрополитена (57, 497, 10^{12}).

Полученные результаты приведены в таблице 1 (колонка 1 – Трудоемкость разработки тестовых сценариев на основе моделей при использовании технологии VRS/TAT после интеграции разработанных методов (человеко-недель); колонка 2 – Трудоемкость существующей разработки тестовых сценариев без использования моделей (человеко-недель); колонка 3 – Трудоемкость создания формальной UCM-модели и генерации модели базовых протоколов с помощью разработанного метода (человеко-дней); колонка 4 – Трудоемкость создания формальной модели базовых протоколов вручную (человеко-дней); колонка 5 – Кол-во трасс, покрывающих требования; колонка 6 – Оптимизированный набор трасс):

Таблица 1. Результаты применения разработанных методов

Проект	1	2	Оценка сокращения трудоемкости (%)	3	4	Оценка сокращения трудоемкости (%)	5	6	Оценка сокращения набора трасс (%)
C_ROUTER	1,8	5,9	69	4	6	33	113	49	56
CDMA	2,2	5,7	61	5	8	37	163	26	84
MPM	2,1	6,3	67	5	10	50	51	51	0
S_HMI	2,8	7	60	7	18	61	223	195	12

1. Преимущество применения технологической цепочки VRS/TAT (после интеграции разработанных методов) по сравнению с традиционным подходом к разработке тестовых сценариев без использования моделей: 60%-ное сокращение трудоемкости (рис. 4).

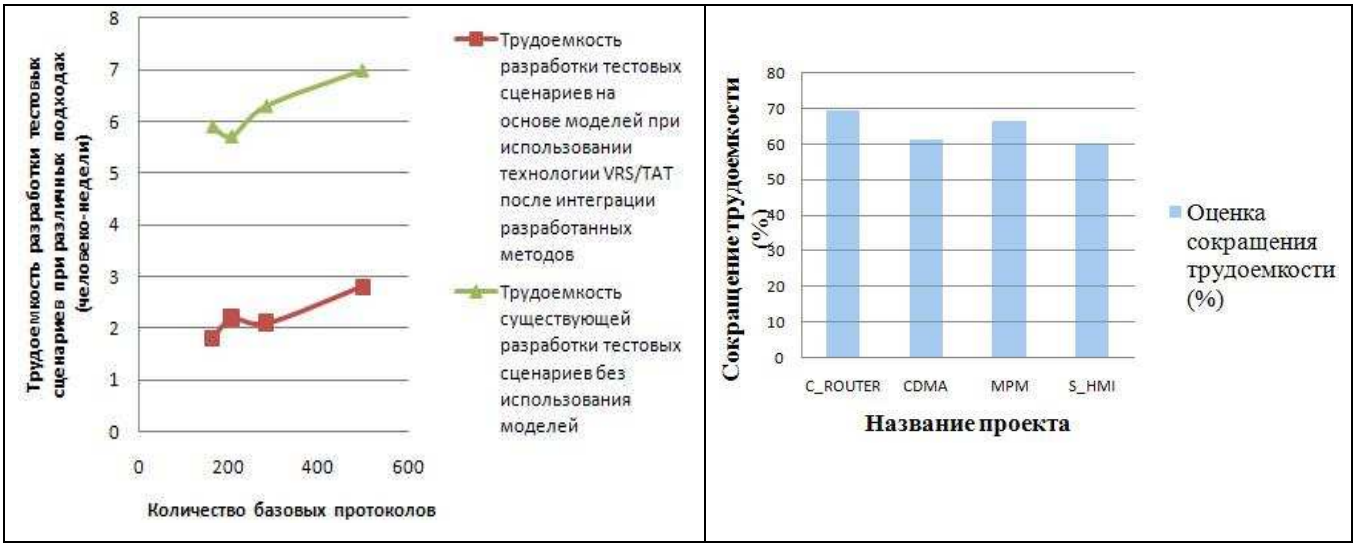

Рис. 4. Сравнение трудоемкости разработки тестовых сценариев при различных подходах

2. Сокращение в среднем на 45% трудоемкости создания формальной модели базовых протоколов инструментом UCM2BP по сравнению с ручной разработкой (рис. 5). При

этом с увеличением количества базовых протоколов показатель сокращения трудоемкости растет (рис. 5), так как при ручном создании формальной модели трудоемкость увеличивается быстрее, чем при автоматизированном построении модели.

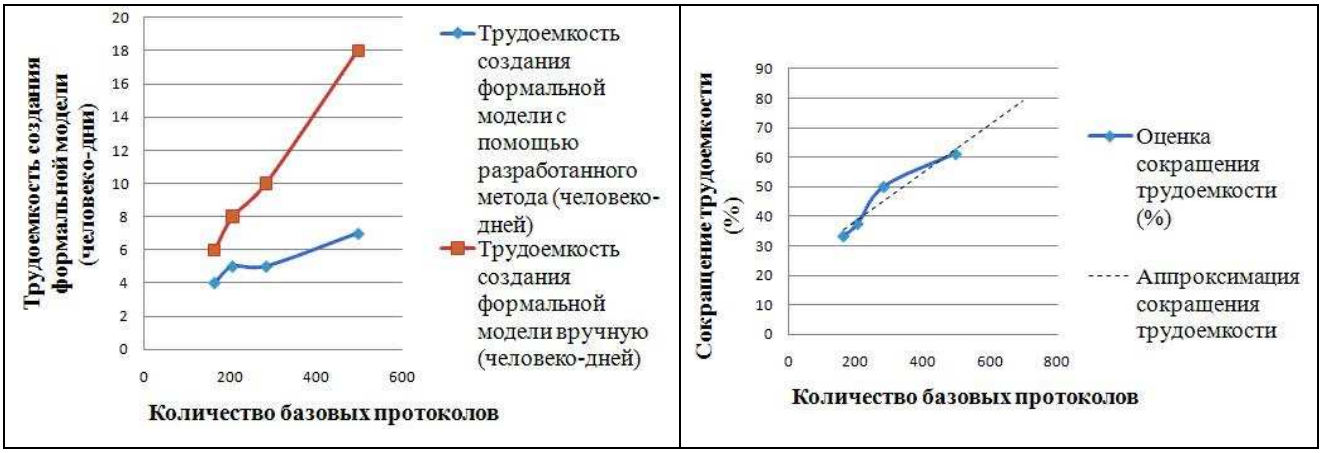

Рис. 5. Сравнение трудоемкости создания формальной модели

3. В итоговом наборе тестовых сценариев количество трасс, сгенерированных по эвристикам, было сокращено более чем на 80% в проекте CDMA. Данный показатель зависит от длины критериальных цепочек и детальности эвристик по отношению к критериальным цепочкам. Если цепочки короткие (один или несколько протоколов), а эвристика описывает длинный сценарий поведения, то возможно покрытие трассой, полученной по данной эвристике, сразу нескольких цепочек. В случае же длинных подробных цепочек эвристика будет совпадать с цепочкой по количеству базовых протоколов, и существенное сокращение количества тестовых сценариев станет невозможным.
4. Трудоемкость повторного цикла применения технологической цепочки для получения новых актуальных тестовых сценариев при изменениях в исходных требованиях значительно сокращается.
5. Применение UCM-моделей на начальном этапе формализации требований позволяет гораздо эффективней взаимодействовать с заказчиком для согласования дальнейших этапов проверки качества разрабатываемого ПО.

ЗАКЛЮЧЕНИЕ

Целью, достигнутой в результате диссертационной работы, являлось сокращение трудоемкости процесса разработки тестовых сценариев путем применения высокоуровневых UCM-моделей, контролируемых заказчиком.

Основные результаты, полученные в ходе выполнения работы:

1. Разработан и применен новый подход к процессу генерации тестовых сценариев на базе формализованных моделей, заключающийся в применении двух формальных моделей: одной – в высокоуровневой нотации Use Case диаграмм (UCM – Use Case Maps), используемой для контролируемого заказчиком описания поведения и согласования с ним поведенческих сценариев, другой – в нотации базовых протоколов для последующего создания по ней тестовых сценариев.
2. Разработаны и интегрированы в технологию VRS/TAT методы сокращения трудоемкости процесса получения тестовых сценариев:
   - автоматическое построение по UCM-модели структурированной формальной модели в виде базовых протоколов;
   - автоматическое создание эвристик;
   - отслеживание соответствия между требованиями и элементами модели;
   - поиск покрытия требований в соответствии с критерием цепочек наблюдаемых событий;
   - сокращение набора тестовых сценариев.
3. Усовершенствованная технология VRS/TAT после интеграции разработанных методов была применена в четырех крупных телекоммуникационных проектах в компаниях ЗАО ”Моторола ЗАО” и ООО “ИЦ “Северо-Западная лаборатория” и доказала свою высокую эффективность для обеспечения проверки качества разрабатываемого ПО, обеспечив более чем 60%-ное сокращение трудоемкости разработки тестовых сценариев по сравнению с традиционным подходом без использования моделей.

Практические результаты, полученные в ходе работы, позволяют сделать вывод, что поставленные задачи решены, и констатировать достижение цели всей работы. По своей значимости работа относится к новым технологическим средствам поддержки процесса тестирования на основе моделей, позволяющим повысить качество разрабатываемого ПО.

СПИСОК ПУБЛИКАЦИЙ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Воинов Н.В., Котляров В.П. Верификация и автоматизация тестирования UML-проектов // Научно-технические ведомости СПбГПУ. № 3 (80). СПб.: Изд-во Политехнического ун-та. – 2009. – С. 220-225. (**издание из перечня ВАК**)
2. Воинов Н.В., Котляров В.П. Применение метода эвристик для создания оптимального набора тестовых сценариев // Научно-технические ведомости СПбГПУ. № 4 (103). СПб.: Изд-во Политехнического ун-та. – 2010. – С. 169-174. (**издание из перечня ВАК**)

3. Воинов Н.В., Веселов А.О., Котляров В.П. Автоматизация тестирования UML проектов // Вычислительные, измерительные и управляющие системы. Сборник научных трудов. СПб.: Изд-во Политехнического ун-та. – 2007. – С. 57-65.
4. Воинов Н.В., Котляров В.П. Методика автоматизации тестирования проектов, специфицированных средствами UML // Технологии Microsoft в теории и практике программирования. Материалы межвузовского конкурса-конференции студентов, аспирантов и молодых ученых Северо-Запада. 17-18 марта 2009 г. СПб.: Изд-во Политехнического ун-та. – 2009. – С. 84-85.
5. Воинов Н.В., Котляров В.П. Применение метода оптимизации обхода пространства поведения формальной модели для генерации тестовых сценариев // Технологии Microsoft в теории и практике программирования. Материалы межвузовского конкурса-конференции студентов, аспирантов и молодых ученых Северо-Запада. 16-17 марта 2010 г. СПб.: Изд-во Политехнического ун-та. – 2010. – С. 139-140.
6. Воинов Н.В., Веселов А.О., Котляров В.П. Технология генерации тестовых наборов из UML спецификаций // Технологии Microsoft в теории и практике программирования. Материалы межвузовского конкурса-конференции студентов, аспирантов и молодых ученых Северо-Запада. 11-12 марта 2008 г. СПб.: Изд-во Политехнического ун-та. – 2008. – С. 22-24.
7. Здробилко С.Н., Воинов Н.В. Автоматизированная формализация функциональных требований XMPP сервера // XXXVIII Неделя науки СПбГПУ. Материалы межвузовской научно-технической конференции. 30 ноября – 5 декабря 2009 г. СПб.: Изд-во Политехнического ун-та. – 2009. – С. 100-102.
8. Синицкий Г.Ю., Воинов Н.В. Применение UCM нотации для автоматизации тестирования программного продукта // XXXIX Неделя науки СПбГПУ. Материалы межвузовской научно-технической конференции. 06-11 декабря 2010 г. СПб.: Изд-во Политехнического ун-та. – 2010. – С. 93-95.
9. Voinov N., Kotlyarov V. Verification and Testing Automation of UML Projects // Proceedings of the Third Spring Young Researchers' Colloquium on Software Engineering. Vol. 3. Moscow, 2009. P. 41-45.