На правах рукописи

Никифоров Игорь Валерьевич

Методы автоматизации построения поведенческой модели программного продукта на основе UCM-спецификаций

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 - 2013

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

Научный руководитель: КОТЛЯРОВ Всеволод Павлович
кандидат технических наук, доцент

Официальные оппоненты: ТЕРЕХОВ Андрей Николаевич,
доктор физико-математических наук,
профессор, заведующий кафедрой системного программирования Математико-механического факультета Санкт-Петербургского государственного университета

ИВАНОВСКИЙ Сергей Алексеевич,
кандидат технических наук, доцент,
заведующий кафедрой математического обеспечения и применения ЭВМ Санкт-Петербургского государственного электротехнического университета

Ведущая организация: ОАО «Научно-производственное объединение «Импульс», г. Санкт-Петербург

Защита состоится «15» мая 2014 г. в 16^{00} часов на заседании диссертационного совета Д. 212.229.18 при ФГБОУ ВПО «Санкт-Петербургский государственный политехнический университет» по адресу: 195251, г. Санкт-Петербург, Политехническая ул., д. 29, уч. корп. 9, ауд. 325.

С диссертацией можно ознакомиться в Фундаментальной библиотеке ФГБОУ ВПО «Санкт-Петербургский государственный политехнический университет». Автореферат диссертации доступен на официальном сайте СПбГПУ (http://www.spbstu.ru/).

Автореферат разослан « » 2014 г.

Ученый секретарь
диссертационного совета
Д 212.229.18 к.т.н., доцент  Васильев Алексей Евгеньевич

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы исследования и степень ее разработанности. На современном этапе развития информационных технологий при создании программного продукта особое значение приобретает разработка требований к проектируемой системе. Требования — это официальный документ, в котором изложены функциональные свойства, характеристики и условия использования объекта. Эти атрибуты требований описываются спецификациями и должны быть реализованы в системе или компоненте системы для удовлетворения контракта, стандарта, спецификации или иных документов, оговоренных заказчиком. Отсутствие требований или неправильная их формулировка к началу создания системы приводят к появлению ошибок и дефектов в разработанном программном продукте. Ошибки в требованиях приводят к значительным временным и финансовым затратам ресурсов для их исправления, корректировки кода, проектной документации и планов.

Одним из способов доказательства корректности исходных спецификаций требований, является использование формальных моделей, составленных на языке описания требований. Фундаментальные работы отечественных (Карпов Ю. Г., Ершов А. П. Лавров С.С. и др.) и зарубежных (Летичевский А. А., Бухр К., Маерс Г., Кларк Э.М., Грамберг О., Пелед Д. и др.) ученых составляют базис теорий тестирования и формальных методов, который необходимо применять и адаптировать к требованиям современной индустрии программного обеспечения. На основе формальных моделей спецификаций с помощью верификации автоматически осуществляется проверка их полноты и непротиворечивости. По корректным формальным моделям автоматически создается множество тестов, удовлетворяющих заданным критериям покрытия, которые в дальнейшем используются в системах тестирования.

Существующие системы автоматической верификации и тестирования зачастую используют собственные форматы, связанные с внутренним представлением, что существенно осложняет их интеграцию с трансляторами языков описания требований. Это обстоятельство, в свою очередь, затрудняет пользовательский контроль результатов верификации и тестирования относительно исходной модели требований. Как следствие, резко возрастают трудозатраты и временные издержки на разработку качественного программного обеспечения.

Актуальной проблемой автоматической проверки требований и генерации тестовых сценариев на основе формальных моделей является сокращение трудоемкости проектирования программного обеспечения за счет автоматизации следующих этапов разработки:

— автоматизации преобразования требований, понятных заказчику и разработчику, в модели на языке, воспринимаемом системой верификации;

— автоматизации поиска и отладки ошибок в формальных моделях;

— автоматизации анализа результатов генерации тестовых сценариев.

Проведенный в работе анализ области автоматизации доказательства корректности исходных спецификаций показал, что для осуществления проверки требований на полноту, непротиворечивость, недетерминизм и для генерации набора тестовых сценариев, целесообразно использовать технологию VRS/TAT. На сегодняшний день она является реальным инструментарием статического анализа промышленных программных систем, интегрирующим символьную верификацию и тестирование.

Для описания требований на ранних этапах разработки системы в технологии VRS/TAT использован язык спецификаций Use Case Map (UCM). Его отличительными особенностями являются простота графической нотации и ориентированность на описание поведенческих сценариев систем. Для доказательства корректности UCM-модели проектируемой системы и формирования тестовых сценариев инструментальная система VRS/TAT использует промежуточный формальный язык базовых протоколов, разработанный А. А. Летичевским.

Настоящая работа посвящена созданию методов автоматизации построения поведенческих моделей программного продукта в виде множества символьных сценариев на основе UCM-спецификаций. В работе решаются задачи разработки UCM-спецификаций по исходным неформализованным требованиям. Для целей автоматизации осуществляется преобразование формальной UCM-модели системы в эквивалентную модель на языке, воспринимаемом инструментами верификации. Кроме того обеспечивается автоматизация анализа и отладки результатов верификации.

Цель и задачи диссертационной работы. Цель диссертационной работы - сокращение трудоемкости процесса разработки поведенческой модели программного продукта путем автоматизации преобразования его высокоуровневых UCM-спецификаций в модель на языке базовых протоколов, использующуюся в технологии тестирования и верификации VRS/TAT, и сокращение трудоемкости разработки тестового набора за счет автоматизации отладки поведенческих сценариев и отслеживания покрытия требований сгенерированными сценариями.

Для достижения цели в работе решены следующие задачи:

— представлен анализ языков описания спецификаций требований и обоснован выбор языка UCM;

— проведен сравнительный анализ технологий работы с UCM-спецификациями и обоснован выбор технологии VRS/TAT;

— разработан метод создания визуальной поведенческой UCM-модели системы на основе последовательности событий, полученных из неформальных требований и описанных на языке UCM;
— предложен метод автоматической проверки ограничений на элементы и конструкции языка UCM, описывающие поток данных в системе и параллельное исполнение;
— создан метод автоматического преобразования UCM-модели, содержащей многопоточные конструкции, временные задержки и прерывания, в формальную модель на языке базовых протоколов, с которым работают инструменты верификации и автоматизации тестирования VRS/TAT;
— реализован инструмент автоматического преобразования визуальной поведенческой UCM-модели в формальную модель на языке базовых протоколов;
— разработан метод и инструментарий автоматизированной отладки сгенерированных по UCM-модели символьных тестовых сценариев и анализа покрытия требований тестовыми сценариями на уровне исходной UCM-модели;
— созданные в работе методы интегрированы в технологическую цепочку автоматизации верификации и тестирования;
— применена и проверена работоспособность предложенных методов и инструментальных средств на промышленных телекоммуникационных проектах.

Предметом исследования являются методы и инструментальные средства автоматизации построения поведенческих моделей программных продуктов, анализа их корректности и тестирования систем.

Научные результаты и их новизна

1. Разработан метод построения поведенческих UCM-спецификаций по исходным требованиям, новизна которого заключается в использовании последовательности событий на языке UCM для создания структурной и поведенческой модели разрабатываемой системы. Наличие в процедурах проверки требований событий на языке UCM позволяет создать обратную связь UCM-модели с требованиями и обеспечить проверку выполнимости требований на сгенерированных поведенческих сценариях.

2. Впервые предложены ограничения на ряд UCM-конструкций, моделирующих многопоточные системы, которые препятствуют созданию некорректного описания системы. А именно: ограничения на конструкции, использующие неограниченное рекурсивное порождение потоков, на конструкции, нарушающие правила порождения и синхронизации потоков, и на конструкции, использующие разделяемые ресурсы потоками без синхронизации.

3. Разработан и реализован метод автоматической проверки ограничений на элементы и конструкции языка UCM, новизна которого заключается в предотвращении класса системных ошибок, препятствующих созданию UCM-моделей, пригодных для автоматической генерации тестовых сценариев.

4. Предложен и реализован метод преобразования элементов и конструкций языка UCM в формальную модель на языке базовых протоколов, отражающую поведение исходных спецификаций. Новизна метода заключается в расширении автоматического преобразователя UCM-модели в модель системы на языке базовых протоколов конструкциями, моделирующими параллельные потоки, временные задержки, исключения и прерывания, а также в создании эффективной технологии генерации тестов на основе высокоуровневых UCM-спецификаций.

5. Разработан и реализован метод автоматизированной отладки процесса генерации тестовых сценариев и анализа покрытия требований тестовыми сценариями, новизна которого заключается в автоматизации анализа тестовых сценариев и явном отображении покрытия непосредственно на уровне исходной UCM-модели.

Теоретическая значимость работы

1. Предложена математическая модель ограничений на конструкции языка UCM для многопоточных систем, которая гарантирует корректность UCM-спецификаций и позволяет автоматический проводить ее анализ на наличие синтаксических и семантических ошибок.

2. Сформулированы правила преобразования многопоточных UCM-моделей, поддерживающих временные задержки, исключения и прерывания в эквивалентные модели на языке базовых протоколов, позволяющие автоматизировать трансляцию моделей.

3. Впервые предложена модель преобразования и отображения тестовых сценариев на UCM-спецификации, что позволяет визуально определять степень удовлетворения тестовыми сценариями требований.

Практическая значимость работы

На базе полученных научных результатов разработан комплекс программных средств, интегрированный в технологию VRS/TAT. Усовершенствованная технология VRS/TAT применена в ряде промышленных телекоммуникационных проектов и показала высокую эффективность при проверке качества создаваемого программного обеспечения, позволив сократить трудоемкость разработки тестовых сценариев на 25 % по сравнению с принятым в настоящее время подходом.

Методология и методы исследования. Для решения поставленных в работе задач используются теория инсерционного моделирования, теория автоматов, аппарат формальных

спецификаций. Применяются стандарты языков Use Case Map (UCM) и Message Sequence Charts (MSC).

Положения, выносимые на защиту

1. Методы сформулированные в работе:

— метод преобразования неформальных требований в формальную модель системы на основе критериальных цепочек на языке UCM;

— метод автоматической проверки ограничений на элементы и конструкции языка UCM;

— метод автоматического преобразования UCM-проекта в формальную модель на языке базовых протоколов.

— метод автоматизированной отладки процесса генерации тестовых сценариев на основе гидов.

2. Алгоритмическая и программная реализация разработанных методов.

3. Результаты применения разработанных методов и инструментальных средств в четырех экспериментальных проектах, подтвердившие их преимущества.

Степень достоверности и апробация работы. Обоснованность и достоверность полученных результатов обеспечивается корректным использованием теории инсерционного программирования, применением аппарата формальных спецификаций, положительными итогами использования разработанных методов и программных средств в экспериментальных проектах.

Основные положения и результаты диссертационной работы доложены и обсуждены на региональных и международных научных конференциях: «Summer Young Researchers' Colloquium on Software Engineering» (Perm, 2012), «Third Spring Young Researchers' Colloquium on Software Engineering» (Kazan, 2013), «Program Semantics, Specification and Verification» (SPb, 2012, 2013), «Технологии Microsoft в теории и практике программирования» (СПб 2009, 2010, 2011, 2012, 2013), XXXVIII неделя науки СПбГПУ (СПб, 2009), XXXIX неделя науки СПбГПУ (СПб, 2010), XL неделя науки СПбГПУ (СПб, 2011), XLI неделя науки СПбГПУ (СПб, 2012), XLII неделя науки СПбГПУ (СПб, 2013).

Публикации. Основные положения диссертации изложены в 27 печатных работах, в том числе 9 работ в журналах из перечня ВАК и 4 на английском языке.

Внедрение. Разработанные методы внедрены в компании ЗАО «Моторола Солюшнз», ООО «ЛЭПКОС» и использованы при разработке учебно-методического комплекса Санкт-Петербургского государственного политехнического университета (СПбГПУ) по учебным дисциплинам «Технология разработки программного обеспечения» и «Технология

разработки промышленных приложений на базе IDE Eclipse» на кафедре «Информационные и управляющие системы» СПбГПУ. Практическое использование представляемых на защиту результатов подтверждено соответствующими актами о внедрении.

Структура и объем работы. Диссертация состоит из введения, пяти глав, заключения, списка литературы и 11 приложений. Объём диссертации – 149 страниц, объем приложений 56 страниц; диссертация содержит 108 рисунков, 19 таблиц, список литературы включает 123 названия.

СОДЕРЖАНИЕ РАБОТЫ

Введение. В разделе «Введение» показана актуальность темы диссертации, определены цель и задачи проведенных исследований, отражена научная новизна и практическая значимость полученных результатов, приведены сведения о реализации работы, апробации, публикациях и о структуре диссертации.

В первой главе рассмотрены методы и подходы повышения качества программного продукта, приведен обзор способов представления требований к системе и сравнительный анализ формальных языков описания спецификаций, проведен обзор и сравнение технологий автоматизации тестирования и верификации, использующих формальный язык UCM. Представлен сравнительный анализ преобразователей формальных UCM-спецификаций в формальные языки, пригодные для средств автоматизации тестирования и верификации. Проанализированы достоинства и недостатки технологий, поставлены цель и задачи диссертационной работы.

Сравнительный анализ формальных языков описания требований проведен с целью выявления наиболее ориентированного на пользователя языка спецификаций. По результатам сравнения выбран язык UCM, т. к. он наиболее интуитивен и понятен как разработчику, так и заказчику.

Для сравнения технологий повышения качества программного продукта были выделены только технологии, использующие формальные модели систем на языке UCM. К ним относятся: SPEC-VALUe, RT-TROOP, UCM/SDL HLTD, UCM CPN и инструментарий автоматизации верификации и тестирования VRS/TAT. На основе сравнительного анализа выделена технология VRS/TAT как технология, занимающая преимущественное положение по сравнению с остальными за счет того, что позволяет проводить не только верификацию модели, но и генерацию и исполнение тестовых сценариев. Как показано в работе, несмотря на указанные преимущества, и эта технология не свободна от недостатков.

Все рассмотренные технологии основаны на преобразовании UCM-спецификаций в целевой язык, воспринимаемый средствами автоматизации верификации и тестирования.

Поэтому в работе проведен анализ существующих преобразователей языка UCM. На основе анализа были сформулированы критерии, которым должны удовлетворять преобразователи, использующиеся в промышленных проектах. К таким критериям отнесены: поддержка многопоточных и временных конструкций, поддержка исключений и прерываний, а также возможность обратного отображения результатов верификации на UCM-модель. Сделан вывод о том, что существующий прототип преобразователя UCM-модели в модель на языке базовых протоколов требует усовершенствований.

Проведенные в главе обзор предметной области и сравнительные анализы позволили сформулировать цель и задачи диссертационной работы: технология VRS/TAT требует усовершенствований для автоматизации и сокращения трудоемкости разработки тестовых сценариев. К основным недостаткам технологии отнесены следующие:

— невозможность отслеживания связи UCM-модели с исходными неформальными требованиями, что не позволяет оценить степень покрытия требований генерируемыми тестовыми сценариями;

— отсутствие автоматического преобразования UCM-конструкций, моделирующих многопоточное исполнение, временные задержки и прерывания, в модель на языке базовых протоколов. Это не позволяет полностью автоматизировать переход между двумя формальными моделями;

— несовершенство механизма отладки, а именно отсутствие автоматизации поиска и исправления ошибок в тестовых сценариях, что значительно увеличивает трудоемкость создания набора тестов;

— отсутствие средств анализа результатов верификации и тестовой генерации на уровне моделей исходного языка, не позволяет объективно оценить покрытие формальной модели и исходных требований тестовыми сценариями.

Перечисленные ограничения являются существенными для автоматизации технологической цепочки верификации и тестирования, и они были устранены автором в диссертационной работе.

Во второй главе представлен анализ методов и подходов к доказательству корректности исходных требований, которые используются в автоматизированной технологии верификации и тестирования VRS/TAT.

Отдельным разделом в главе представлено описание инсерционного программирования, которое лежит в основе реализации теории базовых протоколов. Рассмотрены способы генерации и отбора тестовых сценариев, удовлетворяющих выбранному интегральному критерию покрытия. Дается определение методу направленного поиска и понятию гида.

Проанализированные в главе работы позволили сделать следующие выводы, необходимые для реализации поставленных задач:

1. Методы теории инсерционного программирования могут быть использованы для трактовки конструкций языка UCM и преобразования их в модель на языке базовых протоколов.

2. Использование представленных методов в системе позволяет осуществить проверку программных проектов больших размеров.

3. Использование гидов в процессе тестовой генерации позволяет значительно сузить число рассматриваемых поведений в системе.

В третьей главе рассмотрены методы, направленные на устранение перечисленных в главе 1 недостатков существующей технологии автоматизации верификации и тестирования VRS/TAT. **В четвертой главе** представлены реализующие предложенные методы комплексы и программные средства. Схема модернизированной технологической цепочки, с учетом интеграции четырех разработанных методов, приведена на рисунке 1.

Модернизированная технологическая цепочка позволяет автоматизировать разработку и отладку процесса генерации тестовых сценариев, тем самым делает технологию применимой в промышленных телекоммуникационных проектах, и позволяет расширить спектр решаемых в ней задач. Жирными линиями и пунктиром на рисунке 1 выделены изменения, связанные с интеграцией разработанных методов.

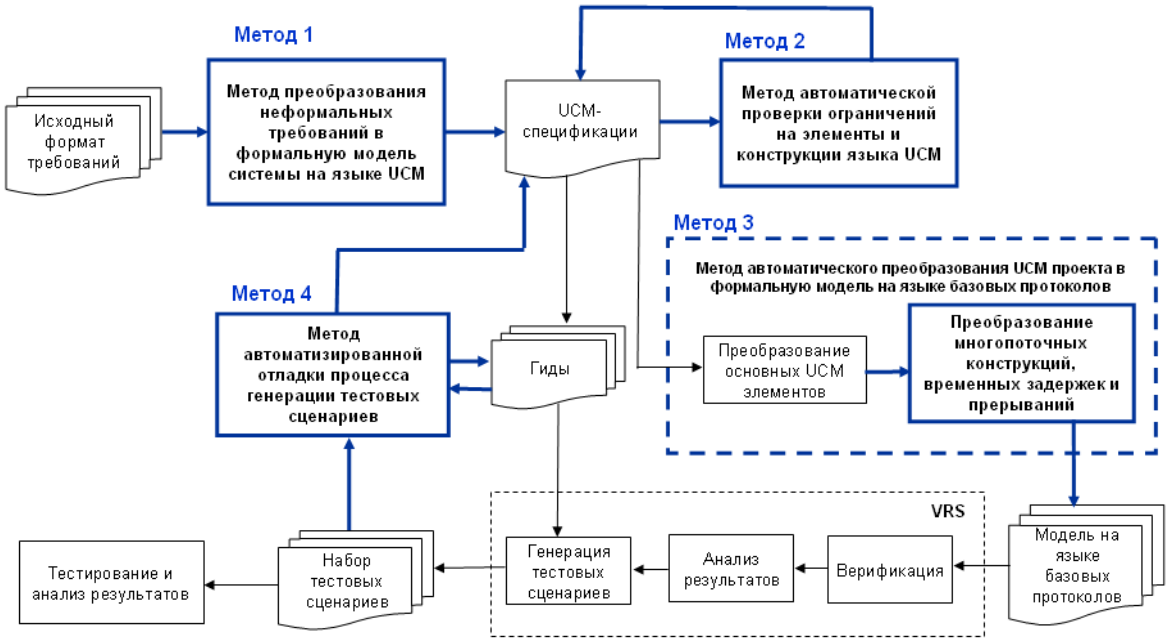


Рисунок 1 - Схема модернизированной технологической цепочки VRS/TAT с интегрированными методами построения формальной модели и анализа тестовых сценариев

Ниже приведен список разработанных методов, а также особенностей их реализации.

1. Метод преобразования неформальных требований в формальную модель системы на основе критериальных цепочек, представленных на языке UCM.

В существовавшей технологической цепочке использовались последовательности событий (критериальные цепочки) на языке базовых протоколов, необходимые для проверки покрытия требований сгенерированными тестовыми сценариями на языке MSC. Такой подход не позволял связать исходные требования с UCM-моделью, а также не позволял осуществить анализ покрытия требований на уровне UCM-модели. Поэтому в работе предложено использовать критериальные цепочки на языке UCM, наличие которых в требованиях позволяет разработать UCM-модель, а также создать обратную связь между исходными требованиями и UCM-моделью с целью доказательства заказчику факта проверки требования.

Метод характеризуется выделением необходимой для построения модели информации из требований, а именно, сценариями проверки требований, их формализацией в виде критериальных цепочек на языке UCM. На рисунке 2 приведена схема реализации предложенного метода преобразования неформальных требований в формальную модель на языке UCM.

По неформальному описанию требований (рисунок 2, столбец «Requirements», переход. 1) выделяются согласуемые с заказчиком сценарии проверки (рисунок 2, столбец «Scenario of requirement», переход. 2). Сценарии проверки представляют собой последовательность действий пользователя и откликов системы на входные воздействия, которые необходимо осуществить и наблюдать для доказательства корректности требования. Затем сценарии проверки требований формализуются последовательностью событий (рисунок 2, столбец «Traceability», переход. 3) на языке UCM, которые затем наносятся на UCM-диаграмму.

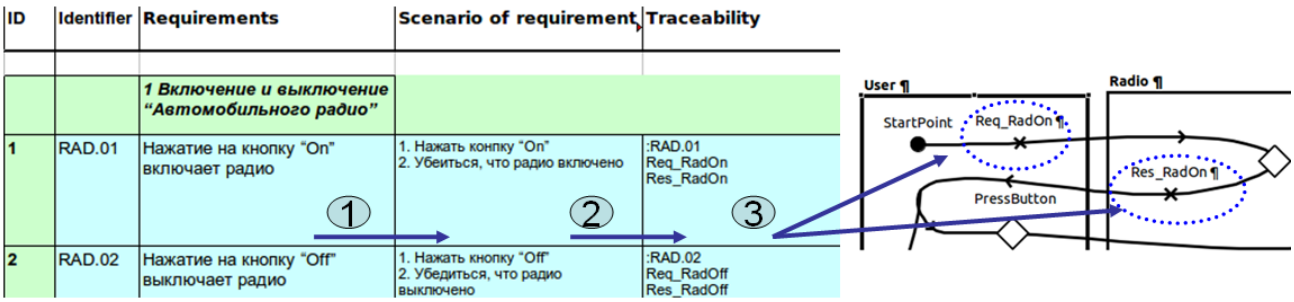


Рисунок 2 - Схема реализации метода перехода от требований к формальной модели на языке UCM

Проектируемая система «S» может быть представлена в виде совокупности требований «req»:

$$\mathbf{S} = \bigcup_{p \in \{1:N\}} \mathbf{req}(p),$$

где p — номер требования; N — общее число пунктов требования. Каждый из пунктов требований может быть представлен кортежем:

$$\mathrm{Req} = \langle \mathrm{Pre}, \mathrm{Events}, \mathrm{Post} \rangle,$$

где *Pre* — описание состояния системы, в котором под действием событий *Events*, система переходит в состояние *Post*.

Выделяя подпоследовательность событий из множества событий *Events*, пользователь формирует критериальную цепочку (*Chain*):

$$Chain = e_1 \to e_2 \to \ldots \to e_n,$$

где каждое событие $e \in Events$, n - размер множества событий *Chain*.

Метод, основанный на критериальных цепочках, позволяет разрабатывать UCM-модель на основе формализованных событий проверки требований на языке UCM и создавать обратную связь между UCM-моделью и исходными требованиями, т. е. продемонстрировать что цепочка *Chain* принадлежит успешно выполненному тестовому сценарию.

2. Метод автоматической проверки ограничений на элементы и конструкции языка UCM.

Несмотря на указанные в первой главе преимущества языка UCM и семантики его элементов, стандарт Z.151 содержит ряд неточностей, приводящих к разработке конструкций, которые препятствуют созданию и доказательству корректности системы при моделировании многопоточных систем. Так, в рамках реализации указанного стандарта не подвергаются анализу на пригодность к преобразованию в модель на языке базовых протоколов метаданные, необходимые для создания потока данных в UCM-модели.

В работе предложен метод, который характеризуется формулировкой ограничений на UCM-спецификацию многопоточных систем и автоматическим анализом UCM-модели на наличие синтаксических и семантических ошибок как в поведенческих диаграммах, так и в занесенных пользователем метаданных.

Предложенный метод автоматического анализа метаданных проекта включает три типа проверок:

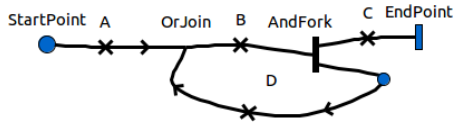
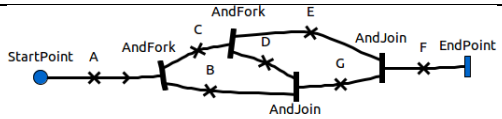
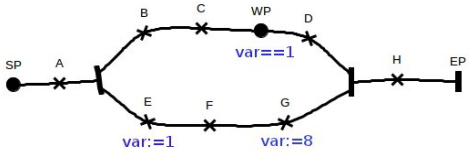
— проверку корректности описания окружения, разработка которого осуществляется согласно правилам описания грамматики окружения;

— проверку корректности описания метаданных элементов проекта, заданных согласно правилам описания грамматики метаданных;

— проверку пересечения имен атрибутов системы, имен UCM-элементов, состояний и имен агентов.

В таблице 1 приведены предлагаемые ограничения на разработку многопоточных систем.

Таблица 1 - Ограничения на разработку многопоточных систем

№ п/п	Название	UCM-диаграмма	Ограничение
1	Неограниченное рекурсивное порождение потоков		Запрещается использовать неограниченное рекурсивное порождение потоков
2	Нарушение баланса скобок		Запрещается использовать параллельные конструкции с нарушением структуры потоков
3	Некорректная синхронизация		Запрещается использовать разделяемые ресурсы без синхронизации на параллельных участках исполнения

Для автоматической проверки UCM-модели на наличие ограничений при разработке параллельных процессов предложено использовать формальное определение потока и признаки некорректных ситуаций, которые реализованы в алгоритме автоматического анализа UCM-модели.

Под потоком T будем понимать четверку $T=(S,P,C,E)$, где S — элемент диаграммы, который порождает поток; P — множество всех потоков, которые являются «родителями» данного; C — множество всех потоков, которые являются «дочерними» к данному; E — множество элементов диаграммы, которые входят в этот поток.

Введем ряд сопутствующих определений, необходимых для формирования признаков некорректных конструкций модели:

$T_i \in T_a$ — i-й поток, содержащийся во всех потоках T_a UCM-модели, где $i \in \{1:N\}$;

$T_i^{S_j}$ — i-й поток, порожденный на элементе S_j, где $S_j \in S$;

$S_i \xrightarrow{<n>} S_j^{next}$ — UCM-элемент S_j^{next}, достижимый из элемента S_i за $<n>$ шагов;

$T_i \rightarrow C_j$ — поток T_i, приводящий к порождению дочернего потока C_j;

$T_i \perp T_j$ — поток T_i ,синхронизирующийся с потоком T_j;

$R \in T_i$ — поток T_i, использующий ресурс R.

Критерий, определяющий неограниченное рекурсивное порождение потоков, определяется формулой:

$$(\forall T_i^{S_j} \in T_a) \rightarrow (C_k^{S_m} \in T_a) \,\&\, (S_j \xrightarrow{<N\neq 0>} S_m) \,\&\, (S_i = S_m).$$

Критерий использования параллельных конструкций с нарушением структуры потоков определяется формулой:

$$(\forall T_i^{S_j} \in T_a) \perp (C_k^{S_m} \in T_a) \,\&\, (T_i \rightarrow C_k).$$

Критерий использования разделяемых ресурсов без синхронизации на параллельных участках исполнения определяется формулой:

$$(T_i \in T_a) \,||\, (T_j \in T_a) \,\&\, (R \in T_i) \,\&\, (R \in T_j).$$

В работе показано, что наложенные ограничения на разработку многопоточных систем и автоматический анализ UCM-проекта на синтаксическую и семантическую корректность позволяют обнаружить и исправить целый класс системных ошибок перед этапом автоматического преобразования UCM-модели в модель на языке базовых протоколов.

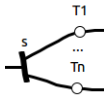
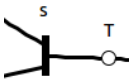
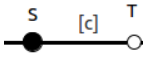
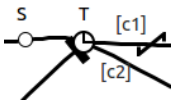
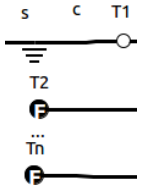
3. Метод автоматического преобразования UCM-проекта в формальную модель на языке базовых протоколов.

В существовавшей технологической цепочке VRS/TAT используется прототип преобразователя UCM-модели в модель на языке базовых протоколов. Этот прототип не поддерживает возможности автоматического преобразования конструкций, моделирующих параллельное исполнение, временные задержки и прерывания. Поэтому в работе поставлена задача автоматизации отражения многопоточных конструкций, временных задержек и прерываний языка UCM в эквивалентные конструкции формальной модели на языке базовых протоколов.

Разработанный в диссертации метод основан на функциях инсерционного программирования, которые были предложены А. А. Летичевским. На основе этих функций создано формальное описание семантических конструкций языка UCM, отвечающих за моделирование параллельных потоков, прерываний и временных задержек (таблица 2).

Трактовка семантики UCM-элементов и конструкций формулами инсерционного программирования позволила построить и реализовать алгоритмы преобразования многопоточных систем, конструкций моделирующих временные задержки, прерывания и исключения, в эквивалентные модели на языке базовых протоколов.

Таблица 2 - Трактовка UCM-конструкций формулами инсерционного программирования

Название конструкции	UCM-диаграмма	Семантическая трактовка	Комментарии
AndFork		$S = insert(T_1,...,T_n)$	$T_1,...,T_n$ – созданные в системе процессы.
AndJoin		$S = check(counter(T_1) > 0 \,\&$ $...\,\& \, counter(T_n) > 0).$ $apply(counter(T_n) + 1).S$	counter(T) – счетчик вошедших на элемент потоков по определенной ветке, S – состояние агента на элементе, n - номер потока
Waiting Place		$S = check(c).T$	«c» — условие ожидания наступления события; «T»— состояние перехода агента.
Timer		$S = check((c_1 \vee c_2) \,\&$ $trigger(T) > 0).$ $apply(trigger := 0).T$	«C_1», «C_2» — условия исходящих ветвей таймера, «trigger(T)>0» — условие наступления события отмены таймера
Failure Point		$S = check((!c).T1 +$ $check(c).apply(\text{expt}(T_2) := 1 \,\&$ $...\,\& \, \text{exp}\,t(T_n) := 1).insert(T_2,...,T_n)$	«c» — условие наступления прерывания, T1 — состояние нормального исполнения, следующее за прерыванием, expt(Tn) — установка флага прерываний и запуск обработчика прерывания на исполнение

4. Метод автоматизированной отладки процесса генерации тестовых сценариев на основе гидов.

Технологическая цепочка VRS/TAT не имела обратной связи между генерируемыми тестовыми сценариями на языке MSC и исходной UCM-моделью, т. е. не позволяла отобразить реализацию поведенческого сценария на уровне исходной UCM-модели. В связи с этим отладка процесса генерации тестовых сценариев на основе гидов требовала значительных временных затрат.

Предложенный в работе метод устранения данной проблемы направлен на сокращение трудоемкости отладки и анализа тестовых сценариев на уровне исходной UCM-модели. Метод характеризуется наличием обратной связи между получаемыми тестовыми сценариями и исходной формальной моделью системы, анализом и визуализацией покрытия тестовыми сценариями исходной формальной UCM-модели, а также алгоритмом и автоматизированным подходом к отладке процесса генерации тестовых сценариев на основе гидов.

На рисунке 3 представлена схема реализации обратной связи между тестовыми сценариями, гидами и UCM-моделью.

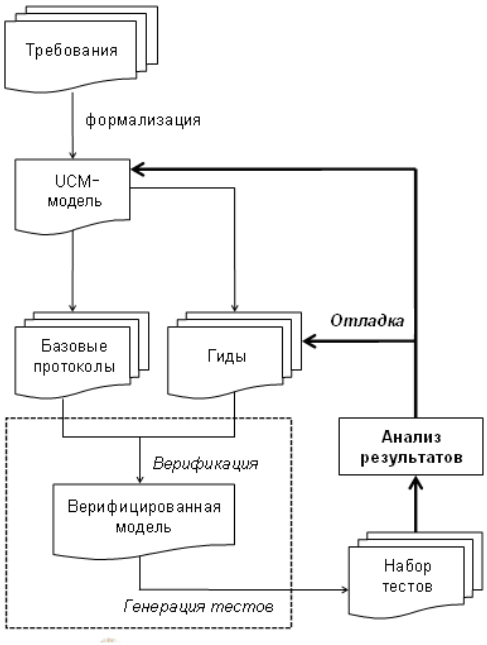


Рисунок 3 - Схема реализации обратной связи между тестовыми сценариями, гидами и UCM-моделью

В работе также предложен подход автоматического обратного отображения генерируемых тестовых сценариев на языке MSC, содержащих последовательность базовых протоколов, в последовательность событий на языке UCM. Каждому базовому протоколу при прямом преобразовании, используемом в методе автоматического преобразования UCM-проекта в формальную модель на языке базовых протоколов, ставится в соответствие UCM-элемент. Такая информация позволяет автоматически осуществлять обратное преобразование.

Тестовые сценарии не всегда успешно генерируются по гидам, т. е. некоторые события гида не могут быть достижимы в тестовом сценарии из-за ошибок дизайна. К наиболее частым причинам неуспешной генерации относятся недостаточная детализация гидов и ошибки в последовательности UCM-элементов в гиде. Причиной подобных ситуаций часто является некорректность использования переменных в UCM-модели, что в процессе верификации может идентифицировать в системе тупиковое состояние.

В работе предлагается способ автоматизации выяснения причин ошибок генерации тестовых сценариев по гидам за счет анализа переменных и принимаемых ими значений в тестовом сценарии.

Применение метода позволяет автоматизировать процесс отладки генерации тестового набора, с тем, чтобы обеспечить 100 % покрытие функциональности требований на основе заданных критериев тестирования.

В пятой главе представлены результаты внедрения разработанных методов в автоматизированную технологическую цепочку верификации и тестирования VRS/TAT. Здесь проанализированы показатели эффективности применения в четырех экспериментальных проектах: «**Автомобильное радио**» — проект, моделирующий поведение автомобильного радио; **SMTP** — модуль широко используемого сетевого протокола, предназначенного для передачи электронной почты в сетях TCP/IP ; **CDMA** — модуль базовой станции системы, реализующей технологию Control Division Multiple Access; «**Спутниковый терминал**» — модуль системы, отвечающий за взаимодействие базовой станции оператора со спутником и обеспечивающий ее подключение к необходимому контроллеру.

Результаты, полученные в рамках применения технологии VRS/TAT после интеграции метода автоматической проверки ограничений на элементы и конструкции языка UCM, свидетельствуют о более чем трехкратном сокращении трудоемкости поиска и исправления ошибок по сравнению с ручным подходом. Рисунок 4 иллюстрирует показатели сокращения трудоемкости в экспериментальных проектах.

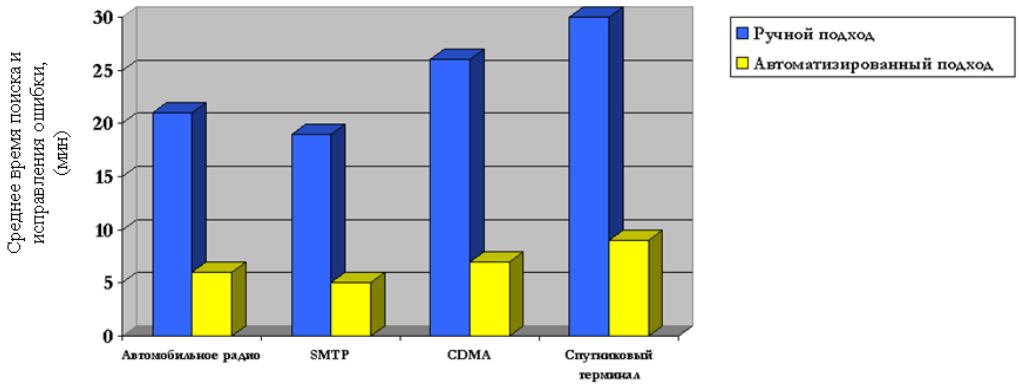


Рисунок 4 - Диаграмма сопоставления ручного и автоматизированного подхода поиска и исправления ошибки

Результаты, полученные в рамках применения технологии VRS/TAT после интеграции метода автоматического преобразования многопоточных модели на языке UCM в модель на языке базовых протоколов, свидетельствуют о сокращении трудоемкости в 2,5 раза по сравнению с ручным подходом.

Рисунок 5 иллюстрирует показатели сокращения трудоемкости, достигнутые при использовании модернизированной предложенными в диссертации методами и алгоритмами технологии автоматизированной генерации базовых протоколов, в 4-х экспериментальных проектах.

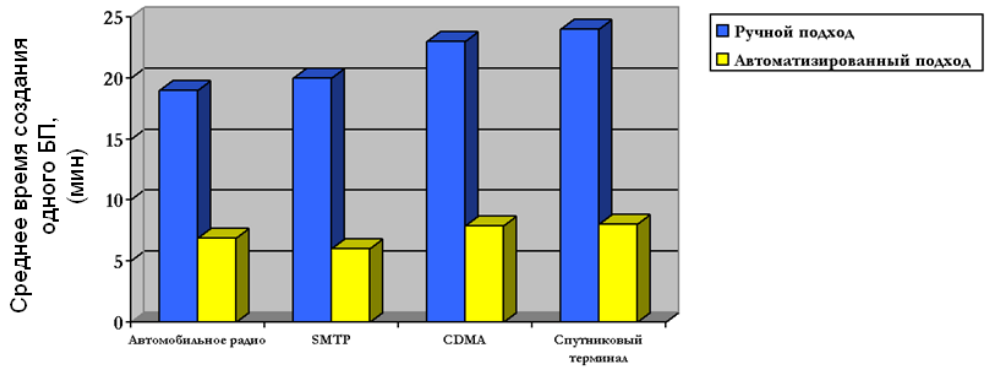


Рисунок 5 - Диаграмма сопоставления ручного и автоматизированного подходов создания одного базового протокола

Таблица 3 демонстрирует преимущества применения технологической цепочки VRS/TAT (с использованием разработанных методов) по сравнению с неавтоматизированным подходом.

Таблица 3 - Оценка преимущества технологии VRS/TAT при разработке тестовых сценариев

Проект	**Количество требований**	**Количество базовых протоколов**	**Трудоемкость создания тестовых сценариев с применением разработанных методов (человеко-недель)**	**Трудоемкость разработки тестовых сценариев без использования разработанных методов (человеко-недель)**	**Оценка сокращения трудоемкости (%)**
Автомобиль-ное радио	30	35	2,5	3,2	22
SMTP	24	30	2,3	2,9	21
CDMA	148	205	3,2	4,3	17
Спутниковый терминал	430	392	5	6,9	28

Результаты, полученные в рамках применения модернизированной технологической цепочки VRS/TAT на четырех экспериментальных проектах, свидетельствуют в среднем о 25 % - ном сокращении трудоемкости по сравнению с существовавшим подходом. Учитывая, что сокращение трудоемкости проекта эквивалентно сокращению себестоимости тестирования, полученный результат является существенным для промышленного производства программного продукта.

На рисунке 6 изображена диаграмма сравнения трудоемкости разработки тестовых сценариев вручную и автоматически при использовании технологии VRS/TAT с интегрированными методами.

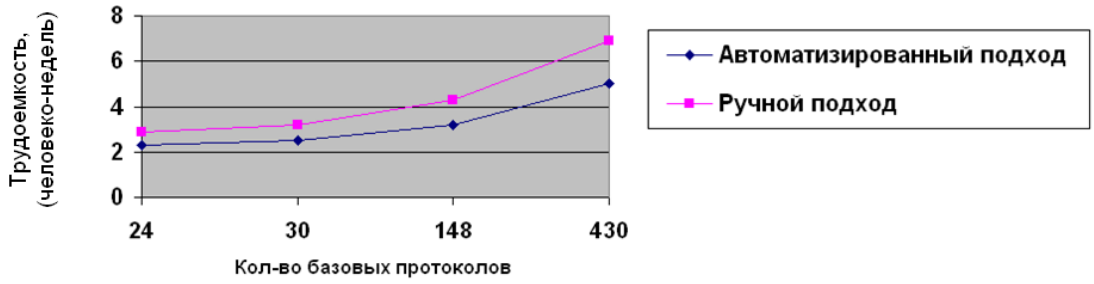


Рисунок 6 - Графики трудоемкости разработки тестовых сценариев

Проведенный в главе 5 анализ результатов применения разработанных методов технологии VRS/TAT для приведенных проектов позволяет сделать следующие выводы:

— предложенные в диссертации методы автоматизации могут быть эффективно использованы в различных промышленных проектах для целей автоматизированного создания тестовых сценариев, покрывающих требования по выбранному интегральному критерию;

— каждый из методов дает значительный выигрыш во времени по сравнению с ручным подходом при решении аналогичных задач. Так, например, применение метода анализа покрытия и автоматизированного поиска причины расхождения трассы и гида приводит к сокращению трудоемкости на один порядок. Метод проверки ограничений на элементы и конструкции языка UCM позволяет добиться как минимум 3-кратного сокращения трудоемкости поиска ошибок (рисунок 4). Метод автоматического построения формальной модели на языке базовых протоколов сокращает до 2,5 раза время создания формальной модели на языке базовых протоколов;

— в рамках всей технологии VRS/TAT после интеграции разработанных методов трудоемкость создания и отладки тестовых сценариев по сравнению с неавтоматизированными этапами сокращается более чем на 25 %;

— использование инновационных методов вне зависимости от особенностей проекта и детальности критериальных цепочек позволяет обеспечить покрытия требований на 100 %;

— трудоемкость повторного цикла применения технологической цепочки для получения новых актуальных тестовых сценариев при изменениях в исходных требованиях значительно сокращается.

ЗАКЛЮЧЕНИЕ

В результате выполненной диссертационной работы созданы новые технологические средства поддержки процесса генерации тестовых сценариев на основе UCM-моделей, позволяющих повысить качество программного обеспечения.

Поставленная цель работы по сокращению трудоемкости процесса разработки поведенческих моделей программных систем на основе автоматизации преобразования высокоуровневых UCM-спецификаций в модель на языке базовых протоколов и автоматизации отладки поведенческих сценариев, а также отслеживания покрытия требований сгенерированными сценариями, достигнута.

Основные результаты, полученные в ходе выполнения работы:

1. Разработан и применен новый подход к процессу создания формальных UCM-моделей, реализуемый на основе создания критериальных цепочек.

2. Разработаны и интегрированы в технологическую цепочку VRS/TAT программные средства и инструменты, реализующие методы сокращения трудоемкости процесса получения поведенческих моделей, в том числе:

— метод автоматического анализа UCM-модели на наличие синтаксических и семантических ошибок;

— метод автоматического построения по UCM-модели формальной модели на языке базовых протоколов, поддерживающий многопоточные системы, временные конструкции и прерывания;

— метод анализа покрытия требований на уровне UCM-модели;

— методика автоматизированного поиска и анализа ошибок в гидах, участвующих в процессе генерации тестовых сценариев.

3. Внедрена при разработке промышленных телекоммуникационных проектов в компаниях ЗАО «Моторола Солюшнз», ООО «ЛЭПКОС» усовершенствованная технология VRS/TAT, которая доказала свою эффективность при проверке качества разработанного программного обеспечения, обеспечив 25 % сокращение трудоемкости разработки тестовых сценариев по сравнению с традиционным подходом без использования методов.

4. Результаты работы, в виде методов, алгоритмов и программных средств, использованы при создании учебно-методического комплекса Санкт-Петербургского государственного политехнического университета (СПбГПУ) по учебным дисциплинам «Технология разработки программного обеспечения» и «Технология разработки промышленных приложений на базе IDE Eclipse» на кафедре «Информационные и управляющие системы» СПбГПУ.

5. Реализован программный комплекс поддержки методов автоматизации в объединенной технологии верификации и тестирования. Общий объем разработанного программного обеспечения, реализующего разработанные методы, составил около

350 KLOC на языке программирования Java. Объем документации на программное обеспечение и методы автоматизации составил более 200 страниц.

Практические результаты, полученные в ходе работы, позволяют сделать вывод, что поставленные исследованием задачи решены, и констатировать достижение общей цели работы.

СПИСОК ОСНОВНЫХ ПУБЛИКАЦИЙ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Никифоров И. В., Юсупов Ю. В., Использование Use Case Map в современной технологической цепочке разработки программного обеспечения // XXXVIII Неделя науки СПбГПУ. Материалы межвузовской научно-технической конференции. 30 ноября - 5 декабря 2009 г. СПб.: Изд-во Политехн. ун-та, 2009. — С. 92-93.

2. Никифоров И. В., Юсупов Ю. В., Преобразование высокоуровневого дизайна программного продукта в нотации Use Case Map в модель на формальном языке // 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 Политехн. ун-та, 2010. — С. 161-162.

3. Никифоров И.В., Петров А.В., Юсупов Ю.В., Генерация формальной модели системы по требованиям, заданным в нотации USE CASE MAP // Научно-технические ведомости СПбГПУ. № 4(103). СПб.: Изд-во Политехн. ун-та, 2010. — С. 191-195. (**Издание из перечня ВАК**).

4. Никифоров И.В., Петров А.В., Юсупов Ю.В., Котляров В.П., Применение различных методик формализации для построения верификационных моделей систем по UCM-спецификациям // Научно-технические ведомости СПбГПУ. № 3(126). СПб.: Изд-во Политехн. ун-та, 2011. — С. 180-184. (**Издание из перечня ВАК**).

5. Никифоров И.В., Петров А.В., Котляров В.П., Особенности трансляции исключений и прерываний UCM в базовые протоколы // XL Неделя науки СПбГПУ. Материалы межвузовской научно-технической конференции. 5 - 10 декабря 2011 г. СПб.: Изд-во Политехн. ун-та, 2011. – С. 103.

6. Никифоров И.В., Васин А.А., Котляров В.П., Анализ трасс и эвристик, основанный на данных из UCM проекта // Технологии Microsoft в теории и практике программирования. Материалы межвузовского конкурса-конференции студентов, аспирантов и молодых ученых Северо-Запада. 27 марта 2012 г. СПб.: Изд-во Политехн. ун-та, 2012. — С. 51-52.

7. Никифоров И.В., Петров А.В., Котляров В.П., Статический метод отладки тестовых сценариев, сгенерированных с помощью эвристик // Научно-технические ведомости СПбГПУ. № 4(152). СПб.: Изд-во Политехн. ун-та, 2012. — С. 114-119. (**Издание из перечня ВАК**).

8. Тютин Б.В., Никифоров И.В., Котляров В.П., Построение системы автоматизации статической и динамической проверки требований к программному продукту// Научно-технические ведомости СПбГПУ. № 4(152). СПб.: Изд-во Политехн. ун-та, 2012. — С. 119-123. (**Издание из перечня ВАК**).

9. Никифоров И.В., Маслаков А.П., Котляров В.П., Отладка UCM-проекта при генерации тестовых сценариев // XLI Неделя науки СПбГПУ. Материалы межвузовской научно-технической конференции. 3-8 декабря 2012 г. СПб.: Изд-во Политехн. ун-та, 2012. — С. 86-87.

10. Ануреев И.С., Баранов С.Н., Белоглазов Д.М., Бодин Е.В., Дробинцев П.Д., Колчин А.В., Котляров В.П., Летичевский А.А., Летичевский А.А. мл., Непомнящий В.А., Никифоров И.В., Потиенко С.В., Прийма Л.В., Тютин Б.В. Средства поддержки интегрированной технологии для анализа и верификации спецификаций телекоммуникационных приложений // Труды СПИИРАН- 2013-№3(26). – С. 349-384. (**Издание из перечня ВАК**).

11. Маслаков А.П., Никифоров И.В., Котляров В.П., Методы поиска расхождений гайдов и трасс в анализаторе последовательностей событий // Технологии Microsoft в теории и практике программирования. Материалы межвузовского конкурса-конференции студентов, аспирантов и молодых ученых Северо-Запада. 26 марта 2013 г. СПб.: Изд-во Политехн. ун-та, 2013. – С. 72-73.

12. Никифоров И.В., Дробинцев П.Д., Котляров В.П., Интегральные критерии проверки требований к программному обеспечению // Научно-технические ведомости СПбГПУ. № 3(174). СПб.: Изд-во Политехн. ун-та, 2013. — С. 111-118. (**Издание из перечня ВАК**).

13. Дробинцев П.Д., Никифоров И.В., Котляров В.П. Методика проектирования тестов сложных программных комплексов на основе структурированных UCM моделей // Научно-технические ведомости СПбГПУ. № 3(174). СПб.: Изд-во Политехн. ун-та, -2013. — С. 99-105. (**Издание из перечня ВАК**).

14. Никифоров И.В., Котляров В.П., Дробинцев П.Д. Ограничения на многопоточные конструкции и временные задержки языка UCM // Научно-технические ведомости СПбГПУ. № 3(174). СПб.: Изд-во Политехн. ун-та, 2013. — С. 148-153. (**Издание из перечня ВАК**).

15. Drobintchev P., Kotlyarov V., Nikiforov I., “Technology Aspects of State Explosion Problem Resolving for Industrial Software Design”, Proceedings of the 7th Spring/Summer Young Researchers’ Colloquium on Software Engineering, Kazan, May 30-31, 2013, pp. 46-51.

16. Drobintsev P.D., Nikiforov I.V., Kotlyarov V.P., Semantics adjustment of UCM Real Time Constructions for Implementation in Translator of UCM to Basic Protocols // Humanities and Science University Journal. № 5 (2013): научный журнал / Санкт-Петербургский университетский консорциум. – СПб., 2013. — С.. 207-215. (**Издание из перечня ВАК**).

17. Tyutin B.V., Nikiforov I.V., Kotlyarov V.P., “Distributed Testing of Multicomponent Systems”, Proceedings of the 6th Spring/Summer Young Researchers’ Colloquium on Software Engineering, Perm, May 30-31, 2012, pp. 75-78.