На правах рукописи

Павлова Елена Анатольевна

МЕТОДЫ И ПРОГРАММНЫЕ СРЕДСТВА РАЗРАБОТКИ ЛОГИЧЕСКИХ КОМПОНЕНТОВ СИСТЕМ С ПОШАГОВЫМИ СТРАТЕГИЯМИ

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

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата технических наук

Автор: _____

Москва –2011

Работа выполнена в Национальном исследовательском ядерном университете «МИФИ» (НИЯУ МИФИ)

Научный руководитель: кандидат технических наук, доцент Сергиевский Максим Владимирович

Официальные оппоненты: доктор технических наук, профессор Косяченко Станислав Анатольевич

кандидат технических наук, доцент Серов Владимир Александрович

Ведущая организация: Институт Системного Программирования Российской Академии Наук

Защита состоится 30 июня 2011 г. в 15 часов 00 минут на заседании диссертационного совета Д 212.130.03 при Национальном исследовательском ядерном университете «МИФИ» по адресу: 115409, г. Москва, Каширское ш., 31. Тел. для справок: +7 (495) 323-95-26.

С диссертацией можно ознакомиться в библиотеке Национального исследовательского ядерного университета «МИФИ» (НИЯУ МИФИ).

Отзывы в двух экземплярах, заверенные печатью, просьба направлять по адресу: 115409, г. Москва, Каширское ш., д.31, отдел диссертационных советов НИЯУ МИФИ, тел.: +7 (495) 323-95-26.

Автореферат разослан « 26 » мая 2011 г.

Ученый секретарь
диссертационного совета Леонова Н.М.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. Разработка систем с пошаговыми стратегиями является одним из самых динамично развивающихся направлений информационных технологий (ИТ). Развитие аппаратного обеспечения современных компьютеров напрямую связано с состоянием дел в области разработки систем с пошаговыми стратегиями. В современных исследованиях, посвящённых обучающим, в частности, развивающим информационным системам, существенное внимание уделяется технологиям их разработки на основе систем с пошаговыми стратегиями. Такие системы применяются в учебных заведениях, а также при подготовке специалистов на производстве и в военном деле. Таким образом, разработка систем с пошаговыми стратегиями представляется актуальным и перспективным направлением ИТ.

В настоящей работе под системами с пошаговыми стратегиями понимаются различные гражданские и военные программы-тренажёры, симуляторы, а также обучающие и развивающие компьютерные игры, в которых особенное внимание уделяется способности пользователя системы принимать решения и вырабатывать правильную стратегию. Программа-тренажер – это программа, предназначенная для ускорения выработки навыков в определенной профессиональной области. Программы-тренажеры часто бывают спроектированы и реализованы в форме компьютерных игр. Под компьютерной игрой понимается программа, служащая для организации игрового процесса. В дальнейшем из рассмотрения исключаются программы-тренажёры и игры, выполняемые на мобильных устройствах, консолях и игровых автоматах. Большинство исследователей, выделяют следующие компоненты, входящие в состав систем с пошаговыми стратегиями: графические, логические, физические, манипулирования данными, сетевые, звуковые, искусственного интеллекта.

Логический компонент (logic engine) – компонент, инкапсулирующий правила и предоставляющий интерфейс, содержащий методы определения изменений состояния системы в результате действий пользователей. В настоящей работе рассматриваются системы с пошаговыми стратегиями, в частности программы-тренажеры и игры. Термин “пошаговая” означает, что функционирование системы может быть описано с помощью упорядоченных во времени процессов, называемых ходами. Смысл работы системы заключается в эффективном управлении некоторыми ресурсами, используя которые, необходимо добиться определенного результата, например, преимущества над противником. Обязательным является наличие оперативного плана, разрабатываемого с учётом меняющейся обстановки. Обычными ресурсами в военных стратегиях являются войска (отдельные персонажи, подразделения или армии) и позиция, которые следует развивать и использовать для достижения преимущества. В экономических стратегиях акцент ставится на развитие экономической инфраструктуры, подконтрольной одной из сторон. Современные системы с пошаговыми стратегиями, как правило, соединяют в себе как военные, так и экономические факторы.

Системы класса пошаговых стратегий имеют относительно простую архитектуру и предъявляют высокие требования только к функциональности логического компонента. Для создания системы с пошаговыми стратегиями необходимо разработать логический и графический компоненты, компонент искусственного

интеллекта и компонент манипулирования данными. Основное внимание при разработке уделяется именно логическому компоненту, в качестве остальных компонентов при создании прототипа системы могут быть использованы готовые разработки сторонних специалистов. Понятная архитектура и возможность использования готовых компонентов позволяют проектировать системы с пошаговыми стратегиями в автоматизированном режиме.

Следует отметить, что индустрия систем с пошаговыми стратегиями, в частности, компьютерных игр характеризуется невысоким уровнем качества выпускаемой продукции по сравнению со средним уровнем качества в индустрии программного обеспечения в целом. Одной из причин невысокого качества продуктов можно назвать несовершенство существующих методов и средств разработки. На текущем этапе развития индустрии не существует общепринятых методов и шаблонов разработки систем с пошаговыми стратегиями. Методы обычно создаются отдельными компаниями и зачастую меняются от проекта к проекту. Информация о методах обычно является закрытой, что затрудняет их повторное использование и модернизацию широким кругом разработчиков. Основное внимание уделяется проектированию архитектуры физического уровня, реализации отдельных компонентов и некоторым аспектам создания правил работы системы. Существуют работы, затрагивающие вопрос логического проектирования, однако они в основном рассматривают применение шаблонов проектирования. В литературе подробно рассмотрены методы разработки графических компонентов, компонентов, моделирующих физические процессы, и компонентов искусственного интеллекта. Сравнительно мало исследований посвящено логическим компонентам, хотя именно они являются важнейшими составляющими систем с пошаговыми стратегиями.

В исследовательских работах явно отмечаются следующие виды деятельности, необходимые для создания логических компонентов: анализ предметной области, проектирование правил, реализация кода компонента на основе правил и тестирование. Как правило, проектирование правил и проверку результатов выполняет проектировщик правил функционирования систем с пошаговыми стратегиями. Он создаёт общую спецификацию, правила, описывает предполагаемый интерфейс системы и все возможные события, которые могут произойти с пользователем. Правила, являющиеся основой логического компонента, обычно описываются неформально. Значительная часть правил представляется в виде обширных таблиц и слабо структурированных документов на естественном языке, которые могут содержать сотни страниц текста. Процесс описания, модификации и проверки правил требует значительных ресурсов, в том числе временных и трудовых. Неформальное описание правил затрудняет применение формальных методов для проверки их корректности, автоматизацию процесса описания и проверки, а также повторное использование правил.

Традиционными языками разработчиков систем с пошаговыми стратегиями являются C и C++, кроме того, часто используются ассемблерные вставки. В связи с этим, реализация часто выполняется в среде разработки Microsoft Visual Studio. При реализации логического компонента и определении структур данных разработчики анализируют описания правил. Поскольку чаще всего используется неформальное описание правил, процесс реализации логического компонента требует значительных

усилий и ресурсов, затруднена генерация кода на основе правил.

Проверка логических компонентов чаще всего выполняется путём тщательного просмотра кода и проектной документации и/или длительного тестирования кода без применения средств автоматизации. Часто отсутствует чёткое представление о проверяемых свойствах, так например, существует множество существенно отличающихся друг от друга представлений о свойстве сбалансированности. Ряд свойств, например совместность, цикличность, непротиворечивость часто вообще не проверяется. Так как проверка осуществляется неформальными методами, по окончанию проверки не гарантируется выполнение конкретных свойств или их отсутствие. Проверка, в том числе правил функционирования системы на предмет сбалансированности, проводится главным образом после комплексной интеграции на этапе тестирования. Реально же нарушение свойств сбалансированности обычно происходит на этапе проектирования. Таким образом, выявление ошибок происходит на поздних этапах разработки, что повышает стоимость исправления ошибок и увеличивает время разработки.

Для создания более структурированных и лаконичных описаний, позволяющих проектировщикам легче вносить изменения в правила, приходится осваивать языки моделирования и программирования общего назначения и соответствующие инструменты, что требует значительных затрат времени и других ресурсов. В качестве альтернативы используются специализированные инструменты проектирования, однако они не учитывают особенности систем с пошаговыми стратегиями, то есть не используется часть накопленного опыта, заведомо пригодная для тиражирования. Примерами могут служить типы сущностей, которые в рамках одного класса имеют устоявшийся набор основных свойств и методов, типы отношений между сущностями, условия выигрыша. В существующих инструментах проектирования описание правил совмещается с описанием графической составляющей системы, а в ряде случаев с описанием элементов искусственного интеллекта. Такое совмещение затрудняет независимый анализ, проверку и модификацию компонентов игры, усложняет повторное использование, а также быструю разработку прототипа системы на основе правил. Повышение эффективности разработки и качества выпускаемых продуктов может быть достигнуто при помощи научных и технических методов, используемых в индустрии программного обеспечения.

Все вышесказанное определяет актуальность создания нового метода разработки логических компонентов систем с пошаговыми стратегиями, который будет учитывать недостатки существующего подхода, позволять структурировать процесс моделирования правил, учитывать специфику пошаговых стратегий, применять формальные методы верификации на этапе проектирования, выполнять автоматическую генерацию кода прототипа.

Объектом исследования диссертационной работы являются системы с пошаговыми стратегиями.

Предмет исследования диссертационной работы – разработка логических компонентов систем с пошаговыми стратегиями.

Целью диссертационной работы является создание метода разработки логических компонентов систем с пошаговыми стратегиями и разработка

инструментального средства поддержки создания логических компонентов, основанного на этом методе.

В соответствии с поставленной целью в диссертационной работе **решаются следующие задачи**:

- анализ существующих методов и инструментальных средств для проектирования, реализации и верификации логических компонентов систем с пошаговыми стратегиями;
- создание метода разработки логических компонентов систем с пошаговыми стратегиями;
- разработка языка представления правил систем с пошаговыми стратегиями;
- формализация свойств, которым должны удовлетворять правила систем с пошаговыми стратегиями и разработка алгоритмов формальной верификации этих свойств;
- проектирование архитектуры системы поддержки разработки логических компонентов систем с пошаговыми стратегиями;
- реализация системы поддержки разработки логических компонентов систем с пошаговыми стратегиями.

Основными **методами исследований**, используемыми в работе, являются методы теории формальных языков, верификации программного обеспечения, теории сложности, объектно-ориентированного анализа и проектирования.

Научная новизна работы заключается в предложенном методе разработки логических компонентов систем с пошаговыми стратегиями, который включает:

- модели процессов разработки логических компонентов систем с пошаговыми стратегиями. Модели являются развитием существующих моделей процессов разработки. Процесс охватывает следующие фазы жизненного цикла: анализ, проектирование, верификацию и реализацию. Проведение верификации после этапа проектирования позволяет выявить ошибки до реализации программного кода;
- новый формальный язык для описания правил систем с пошаговыми стратегиями;
- формализованные свойства, которым должны удовлетворять правила систем с пошаговыми стратегиями. Свойства совместности позволят проверять корректность взаимодействия объектов правил и наличие декларированной функциональности. Свойства сбалансированности формализуют понятия справедливости правил, используемые в литературе, в виде формул временной логики, позволяя, таким образом, формально проверять эти свойства;
- новый алгоритм статической проверки свойств совместности;
- новый алгоритм сокращения числа состояний для верификации свойств сбалансированности при помощи проверки на модели (model checking).

Практическая значимость результатов определяется следующим:

- реализован язык для описания правил систем с пошаговыми стратегиями. Язык учитывает особенности таких систем и позволяет проектировщику правил создавать графические описания правил в терминах предметной области;
- разработана система поддержки разработки логических компонентов систем с пошаговыми стратегиями. Она позволяет создавать системы с пошаговыми

стратегиями при помощи предложенного метода разработки логических компонентов, используя предметно-ориентированный язык для описания правил, и проводить верификацию при помощи предложенных алгоритмов.

Результаты работы могут быть использованы при решении задач разработки тренажёров, компьютерных игр, обучающих программ и других интерактивных графических сред.

Внедрение результатов исследований. Результаты диссертационной работы внедрены в ОАО «Туристский информационный центр города Москвы» для решения задачи верификации корректности компонентной модели системы бронирования туристических услуг и в ООО «Некки» при решении задачи автоматизации разработки элементов компьютерных игр.

Публикации и апробация работы. Результаты диссертации изложены в 13 публикациях и докладывались на конференциях и семинарах различного уровня:

- на Научной сессии МИФИ в 2008 г.;
- на V Всероссийской конференции студентов, аспирантов и молодых ученых «Технологии Microsoft в теории и практике программирования» в 2008 г.;
- на Software Engineering Conference (Russia), SEC(R) в 2008 г.;
- на Научной сессии МИФИ в 2009 г.;
- на VI Всероссийской конференции студентов, аспирантов и молодых ученых «Технологии Microsoft в теории и практике программирования» в 2009 г.;
- на Third Spring Young Researchers' Colloquium on Software Engineering (SYRCoSE) в 2009 г.;
- на научном семинаре ИСП РАН в 2009 г.;
- на Научной сессии МИФИ в 2010 г.

Основные положения, выносимые на защиту:

- результаты анализа методов проектирования, верификации и реализации логических компонентов систем с пошаговыми стратегиями;
- метод разработки логических компонентов систем с пошаговыми стратегиями, включающий:
  - модели процессов разработки логических компонентов систем с пошаговыми стратегиями;
  - формальный язык для описания правил систем с пошаговыми стратегиями;
  - формализованные свойства совместности и сбалансированности, которым должны удовлетворять правила систем с пошаговыми стратегиями;
  - алгоритм верификации совместности правил систем с пошаговыми стратегиями;
  - алгоритм сокращения числа состояний для верификации свойств сбалансированности при помощи проверки на модели (model checking);
- архитектура системы поддержки разработки логических компонентов систем с пошаговыми стратегиями;
- программная система, обеспечивающая поддержку разработки логических компонентов систем с пошаговыми стратегиями.

Структура работы. Работа состоит из введения, четырёх глав, заключения, списка литературы, включающего 144 наименования, и семи приложений. Текст диссертации изложен на 165 страницах, включая 39 рисунков и 18 таблиц.

СОДЕРЖАНИЕ РАБОТЫ

Во **введении** обосновывается актуальность темы диссертации, определяются цель и задачи исследования.

В **первой главе** рассматриваются особенности индустрии систем с пошаговыми стратегиями по сравнению с индустрией разработки программного обеспечения, определяются особенности систем с пошаговыми стратегиями как программных систем, рассматривается понятие метода разработки программного обеспечения и структуры метода, проводится обзор методов и средств разработки логических компонентов систем с пошаговыми стратегиями.

Особенности разработки систем с пошаговыми стратегиями как программных систем проявляются уже на этапе сбора требований. Как правило, предъявляются жесткие требования к срокам разработки. Кроме того, для систем с пошаговыми стратегиями типичны высокие требования к производительности отдельных программных компонентов.

Особенности проектирования и реализации систем с пошаговыми стратегиями заключаются в привлечении большого числа нетехнических специалистов (сценаристов, дизайнеров, музыкантов), модель работы которых близка к водопадной, что отражается на процессе разработки систем в целом. Следует отметить, что существует общепринятое представление о высокоуровневой компонентной архитектуре систем с пошаговыми стратегиями и задачи разработки некоторых компонентов систем с пошаговыми стратегиями хорошо изучены.

Рассмотрим традиционный подход к разработке логических компонентов, представление о котором можно составить по описанию отдельных особенностей подхода в исследовательской литературе. Подход включает следующие действия: анализ предметной области, проектирование правил, их реализацию и тестирование. Проектирование правил систем с пошаговыми стратегиями осуществляет проектировщик правил. Результатом проектирования является текстовый документ на естественном языке и сопровождающие его таблицы. Документ содержит подробное описание правил, а таблицы представляют свод численных характеристик объектов. Текстовый документ не имеет чётко регламентированной структуры. Он создаётся при помощи текстового процессора, а таблицы – при помощи процессора электронных таблиц.

Реализацию выполняет команда разработчиков. Традиционными языками разработки являются C и C++. В связи с вышесказанным, реализация часто выполняется при помощи среды разработки Microsoft Visual Studio. Тестирование выполняется командой тестировщиков, которая может быть набрана из числа пользователей игры или программы-тренажёра. Тестирование отдельных свойств (например, сбалансированности правил) может выполняться проектировщиком правил путём тщательного просмотра проектной документации. В отдельных работах предлагается отложить тестирование на этап, следующий за выпуском продукта. Для автоматизации разработки логических компонентов применяются инструментальные

средства поддержки разработки (например, Torque Engine Advanced, FPS Creator, NeoAxis Engine, Offset Engine, Unreal Engine и C4 Engine). Отметим основные недостатки этих средств: многие инструменты не поддерживают графическое описание правил, описание совмещается с описанием графической составляющей и элементов искусственного интеллекта, не учитываются особенности конкретных систем. В частности, не существует средства, учитывающего специфику пошаговых стратегий. Многие инструменты не поддерживают автоматизированную проверку правил и не позволяют автоматизировать процесс создания прототипа программы на основе описания правил.

Методы проверки свойств совместности правил и баланса между пользователями наименее исследованы. Они могут быть проверены формальными методами, гарантирующими выполнение свойств в случае успешного завершения проверки. Наиболее перспективным методом проверки представляется сочетание статического анализа свойств совместности и верификации при помощи проверки на модели (model checking) для свойств баланса. Такой подход позволит проводить проверку в автоматизированном режиме и гарантировать выполнение или отсутствие определённых свойств. В качестве инструментального средства для проверки на модели предлагается использовать верификатор SPIN, позволяющий проводить параллельную верификацию модели на нескольких компьютерах, реализующий основные алгоритмы проверки на модели и подходы, связанные с сокращением числа состояний проверяемой модели. Для этого инструмента существует подробная документация и множество описаний примеров использования в исследовательских работах.

Методы и инструментальные средства разработки систем с пошаговыми стратегиями могут быть существенно улучшены за счёт адаптации подходов к разработке сложных информационных систем, используемых в области программной инженерии. В частности, элементы подхода к разработке на основе моделей и предметно-ориентированных языков могут быть применены при разработке систем с пошаговыми стратегиями для создания моделей правил, конфигураций и быстрой генерации прототипов на основе этих моделей.

Во **второй главе** предлагается новый метод разработки логических компонентов систем с пошаговыми стратегиями, являющийся развитием существующего подхода к разработке. Описание метода выполняется согласно стандарту и включает описание стадий разработки (стадии анализа, проектирования, верификации и реализации), процессов, соответствующих каждой стадии, задач, соответствующих каждому из процессов. В описании определяются действующие лица (роли), принимающие участие в процессах, модели и программные системы, получаемые в результате разработки, определяется соответствие результатов разработки задачам. Определяются языки, используемые для описания моделей, и диаграммы, документирующие модели.

Далее подробно рассматривается язык описания правил пошаговых стратегий. Язык позволяет описывать объекты правил (включая их данные и поведение), отношения между объектами, реакцию одних объектов на поведение других. Синтаксис языка задаётся формальной грамматикой. Язык принадлежит к классу LL(1). Семантика языка задаётся в виде денотационной семантики, где конструкциям языка ставятся в соответствие выражения в $\lambda\varsigma$-исчислении (исчисление объектов,

подобное функциональному λ-исчислению), расширенном элементами логики Хоара для описания ограничений предусловий и постусловий поведения объектов. В описании семантики языка определяются семантические домены, функции и уравнения, задающие отображение синтаксических элементов в семантические. Для языка разработаны графическая и текстовая нотации. На основе ядра языка, позволяющего описывать объекты, отношения и реакции, построен репозиторий, содержащий типы объектов для пошаговых стратегий и отношения между ними.

Правила, описанные при помощи предложенного языка, могут быть проверены на совместность. Понятия синтаксической и семантической совместности определяются при помощи расширенного λς-исчисления, позволяющего описывать интерфейсы, отношения между ними и операции над интерфейсами. Модель синтаксически совместна тогда и только тогда, когда сумма всех требуемых интерфейсов объектов модели входит в сумму всех предоставляемых интерфейсов объектов модели:

$$\sum_{i,j} I_{ij}^{req} \subset \sum_{i,j} I_{ij}^{pr}, \quad (1)$$

где I_{ij}^{req} – требуемый интерфейс i-го объекта для взаимодействия с j-м, I_{ij}^{pr} – предоставляемый интерфейс i-го объекта для взаимодействия с j-м. Модель семантически совместна тогда и только тогда, когда семантически совместны все пары взаимодействующих объектов модели:

$$\forall object_i, object_j : (D_I(I_i^{req}, I_j^{pr}) = True) \,\&\, (D_I(I_j^{req}, I_i^{pr}) = True). \quad (2)$$

Пара взаимодействующих объектов $object_i$ и $object_j$ совместна тогда и только тогда, когда допустим контракт предоставляемого для j-го объекта интерфейса i-го, и требуемого от i-го интерфейса j-го, и допустим контракт предоставляемого для i-го интерфейса j-го, и требуемого от j-го интерфейса i-го. Контракт предоставляемого и требуемого интерфейсов допустим $(D_I(I^{req}, I^{pr}) = True)$ тогда и только тогда, когда для любого метода m_1 из требуемого интерфейса I^{req} найдётся метод m_2 предоставляемого интерфейса I^{pr} такой, что контракт этих методов является допустимым:

$$(D_I(I^{req}, I^{pr}) = True) \Leftrightarrow \forall m_1 \in I^{req} \; \exists m_2 \in I^{pr} : D_m(m_1, m_2) = True. \quad (3)$$

Контракт двух методов m_1 и m_2 полагается допустимым тогда и только тогда, когда совпадают сигнатуры методов, из предусловия m_2 следует предусловие m_1, а из постусловия m_1 следует постусловие m_2 :

$$(D_m(m_1, m_2) = True) \Leftrightarrow ((Q_{pred2} => Q_{pred1}) \,\&\, (R_{post1} => R_{post2}) = True) \& (m_1 \equiv m_2), \quad (4)$$

где Q_{pred1} – предусловие метода m_1, Q_{pred2} – предусловие метода m_2, R_{post1} – постусловие метода m_1, R_{post2} – постусловие метода m_2.

Ниже приводится предложенный в работе алгоритм проверки свойств совместности правил игр и программ-тренажёров класса пошаговых стратегий.

Шаг 1. Определить для каждого объекта модели правил предоставляемый интерфейс.

Шаг 2. Определить для каждого объекта требуемые интерфейсы.

Шаг 3. Проверить синтаксическую совместность модели при помощи шагов 3.1–3.3.

Шаг 3.1. Построить сумму $\sum_{i,j} I_{ij}^{pr}$ всех предоставляемых интерфейсов модели.

Шаг 3.2. Построить сумму $\sum_{i,j} I_{ij}^{req}$ всех требуемых интерфейсов модели.

Шаг 3.3. Если выполняется $\sum_{i,j} I_{ij}^{req} \subset \sum_{i,j} I_{ij}^{pr}$, модель синтаксически совместна, перейти к шагу 4, иначе модель синтаксически несовместна, перейти к шагу 6.

Шаг 4. Проверить семантическую совместность модели правил игры. Для этого для всех пар взаимодействующих объектов A и B модели проверить допустимость контрактов требуемых и предоставляемых интерфейсов $(D_I(I_A{}^{req}, I_B{}^{pr}) = True)\, u\, (D_I(I_B{}^{req}, I_A{}^{pr}) = True)$. Если модель семантически совместна, перейти к шагу 6, иначе перейти к шагу 5.

Шаг 5. Модель правил несовместна.

Шаг 6. Конец.

Рассмотрим ещё один вид свойств, которым должны обладать правила - свойство баланса между игроками. Формализуем наиболее часто используемые в литературе критерии баланса. Правила, не допускающие выигрыша одного из игроков, считаются несбалансированными. Свойство «игру может выиграть i-й игрок» формулируется в логике CTL следующим образом: *EF (win=i)*, где i- переменная, в которой хранится номер игрока, win – переменная, содержащая номер победителя или 0, если игра не окончена. В LTL-логике возможно выразить отрицание этого свойства:

$$\overline{EF(win = i)} = A!(F(win = i)). \quad (5)$$

Несбалансированными считаются правила, при которых i-й игрок всегда проигрывает за первые b ходов. Сформулируем отрицание критерия «Существуют варианты игры, в которых i-й игрок не проигрывает за первые b ходов», count — счётчик ходов, loose – номер проигравшего игрока:

$$\overline{EF((loose! = i)\,\&\,(count \le b))} = A!(F((loose! = i)\,\&\,(count \le b))). \quad (6)$$

Считать несбалансированными правила, при которых состояние i-го игрока всегда хуже или равно состоянию j-го на протяжении первых b ходов:

$$\overline{EF((v_i > v_j)\,\&\,(count \le b))} = A!(F((v_i > v_j)\,\&\,(count \le b))), \quad (7)$$

где v_i, v_j – оценки состояния игроков на момент конкретного хода. Несбалансированны правила, при которых состояние i-го игрока ухудшается на протяжении первых b ходов, $deltav_i$ – приращение состояния i-го игрока за один ход:

$$\overline{EF((deltav_i > 0)\,\&\,(count \le b))} = A!(F((deltav_i > 0)\,\&\,(count \le b))). \quad (8)$$

Несправедливыми будем считать правила, когда отношение ущерба $damage_i$, наносимого отрядом i, к его оценочной стоимости $price_i$ отличается более чем в b раз от отношения $damage_j$, наносимого отрядом j, к его оценочной стоимости $price_j$:

$$AG\left(\frac{damage_i}{price_i} = \frac{damage_j}{price_j} \cdot b\right). \quad (9)$$

В исследовательских работах приводится определение сбалансированной игры как игры, в которой «игрок не может быть поставлен в условия, где невозможен выигрыш, иначе как по своей вине».

Аналогичное свойство можно сформулировать в логике CTL*:

$$AG(AF(win!=i)\rightarrow f), \qquad (10)$$

где f- счётчик ошибок. То есть, пока игрок не ошибся хотя бы один раз, он не проиграл.

Можно формулировать близкое по смыслу свойство в логике LTL:

$$AG(!f\ R\ !(loose)). \qquad (11)$$

В рассматриваемых критериях баланса используется понятие «состояние игрока». В работе формализуется это понятие и предлагается способ его расчёта. Верификация свойств сбалансированности проводится методом проверки на модели. Одной из основных сложностей проведения такой проверки является необходимость рассмотрения большого числа состояний дерева игры. В работе предлагаются способы уменьшения числа состояний, основанные на особенностях правил пошаговых стратегий. Рассматриваются следующие способы: исключение из рассмотрения состояний, не являющихся состояниями конца хода; исключение из рассмотрения клеток карты, не содержащих объектов, атрибуты которых используются в проверяемом свойстве; ограничение глубины анализа при проверке конкретных свойств; рассмотрение только тех путей в дереве состояний игры, которые соответствуют определённым стратегиям поведения игроков.

В **третьей главе** рассматривается проектирование инструментального средства, основанного на предложенном методе разработке. На рисунке 1 приведена общая архитектура инструментального средства поддержки разработки логических компонентов систем с пошаговыми стратегиями. На концептуальном уровне проектирования формулируются требования, разрабатываются варианты использования инструмента и создаётся его концептуальная архитектура. При проектировании архитектуры логического уровня происходит предварительное определение высокоуровневых компонентов инструмента, построение диаграмм последовательностей для компонентов, определение основных классов инструмента и отношений между ними. На физическом уровне проектирования уточняются диаграммы классов, определяются атрибуты классов и их типы, а также методы классов. Классы распределяются по компонентам, уточняется компонентная архитектура, проектируется модель развёртывания. Диаграммы вариантов использования, последовательностей, диаграммы классов, компонентов и диаграммы развёртывания описаны на основе стандарта UML 2.0 в инструментальном средстве IBM Rational Suite.

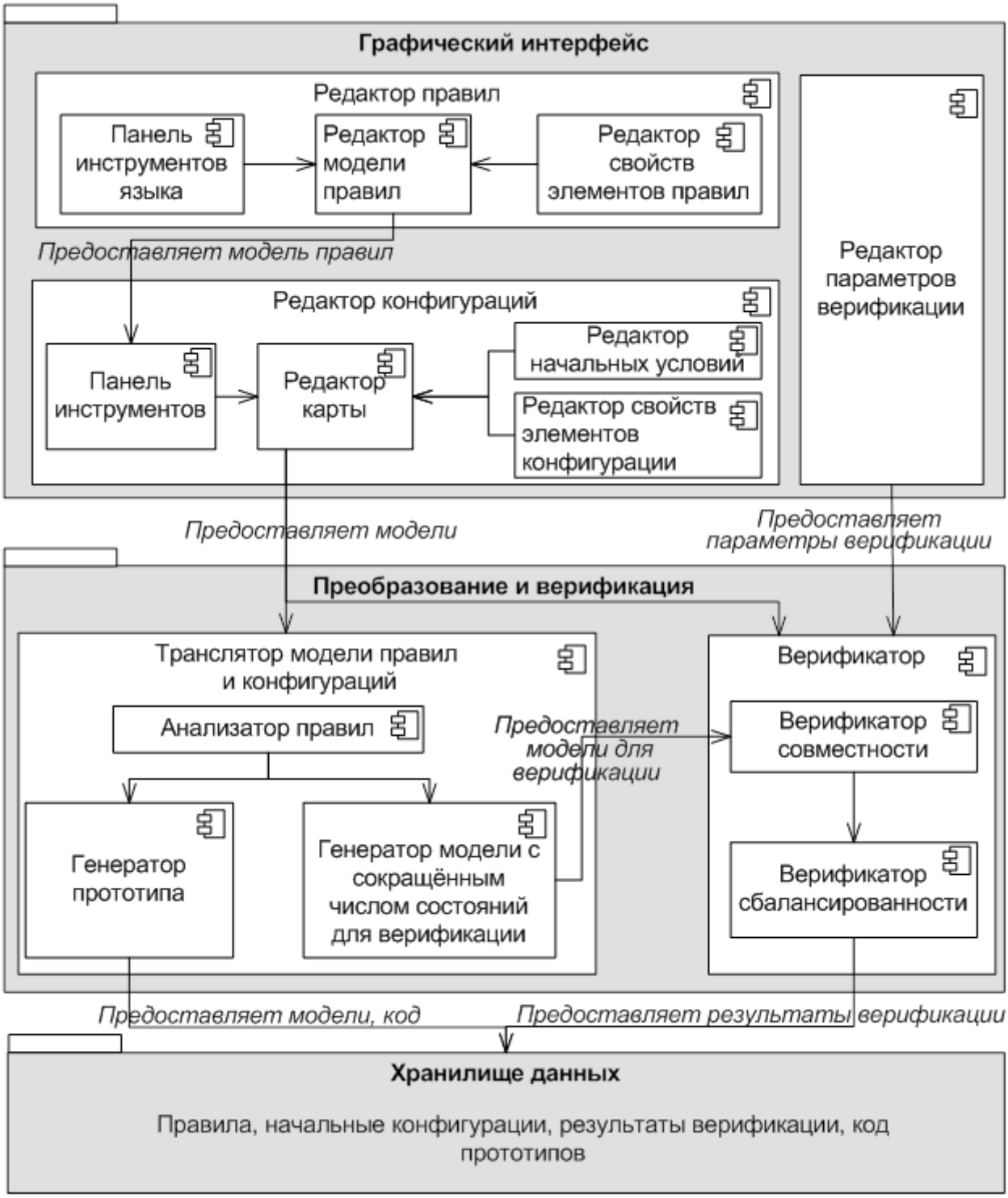

Рисунок 1 – Компонентная архитектура разработанного инструментального средства (в нотации UML 2.2)

В **четвёртой главе** рассматриваются особенности реализации и тестирования инструментального средства поддержки разработки логических компонентов. Реализация языка описания правил, спроектированного во второй главе, выполнена при помощи инструментального средства Visual Studio Domain-Specific Language Tools (далее DSL Tools). Для этого была построена графическая модель языка, содержащая элементы ядра языка и отношения между ними. На основе модели языка при помощи DSL Tools было сгенерировано приложение с графическим интерфейсом пользователя, способное интегрироваться в рабочую среду Visual Studio. Приложение содержит

редактор правил и конфигураций. Для генерации кода логического компонента были определены правила трансляции модели на разработанном языке в код на языке C#. При помощи DSL Tools на основе правил трансляции был реализован генератор прототипа, объединяющий сгенерированный на основе правил логический компонент и другие компоненты, входящие в состав систем с пошаговыми стратегиями.

Был реализован верификатор свойств совместности и сбалансированности. Для верификации сбалансированности был реализован алгоритм, преобразующий модель правил на предметно-ориентированном языке в код на языке Promela (внутренний язык верификатора SPIN) в сочетании с фрагментами кода на Си. Такой код описывает дерево всех возможных игр по разработанным правилам, где путь от начальной вершины дерева к концевой соответствует определённой игре. При преобразовании модели правил в код на Promela учитываются способы сокращения числа состояний, которые были предложены во второй главе. Дерево игр используется SPIN для проведения верификации.

Было проведено тестирование разработанного инструментального средства. В процессе тестирования были выделены следующие основные этапы: модульное, интеграционное и системное тестирование. На этапе модульного тестирования было проведено тестирование методов каждого класса инструментального средства. На этапе интеграционного тестирования проводился поиск ошибок, связанных с взаимодействием между компонентами. Цель системного тестирования состояла в выявлении дефектов, связанных с работой системы в целом, таких как неверное использование ресурсов системы, несовместимость с окружением, непредусмотренные сценарии использования, отсутствующая или неверная функциональность. На основании данных о тестовых покрытиях на каждом этапе тестирования были сделаны выводы о достижении критериев тестированности и успешном завершении тестирования.

Было проведено сравнение разработанного инструментального средства с системами, решающими аналогичные задачи, на основе критериев определённых во второй главе. Основными преимуществами инструментального средства являются: отделение описания правил от графического представления, возможность автоматической верификации игр, в том числе сбалансированности правил, и учёт специфики игр в классе пошаговых стратегий. Было проведено испытание разработанного инструментального средства на примере компьютерной программы-тренажёра TacOps 4, используемого более десяти лет вооруженными силами Канады, Австралии, Новой Зеландии, США и Германии. В результате верификации построенной для правил тренажёра модели при помощи инструментального средства было обнаружено нарушение свойства сбалансированности (4) для исходных правил, а также свойств (1) и (2) для модифицированных правил.

В **заключении** приведены основные результаты диссертационной работы.

ОСНОВНЫЕ ВЫВОДЫ И РЕЗУЛЬТАТЫ РАБОТЫ

1. проведен анализ существующих методов разработки систем с пошаговыми стратегиями, представленных в публикациях. В частности, исследованы методы проектирования правил и методы обеспечения качества логических компонентов. На основе результатов анализа сделан вывод о недостаточности обеспечиваемого этими методами уровня качества логических компонентов и необходимости синтеза нового

метода разработки;

2. предложен метод разработки логических компонентов систем с пошаговыми стратегиями, являющийся развитием существующих подходов к разработке. Метод охватывает основные стадии жизненного цикла: анализ предметной области, проектирование, реализацию и верификацию. Метод позволяет формально описывать и верифицировать правила и генерировать код компонента на основе правил, таким образом, позволяя автоматизировать разработку логических компонентов. Метод позволяет проводить верификацию на более ранних этапах разработки, снижая затраты на исправление ошибок. Метод определяет процесс разработки, его участников, применяемые инструменты и основные результаты;

3. разработан новый язык описания правил систем с пошаговмих стратегиями. Язык учитывает особенности систем этого класса и позволяет проектировщику правил создавать графические описания правил в терминах предметной области;

4. для языка разработана формальная спецификация, определяющая синтаксис и денотационную семантику языка. Спецификация позволяет применить формальные методы анализа к описаниям правил на этом языке;

5. формализованы свойства, которым должны удовлетворять правила. Разработан новый алгоритм верификации свойств совместности при помощи статического анализа и новый алгоритм сокращения числа состояний для верификации свойств сбалансированности при помощи проверки на модели (model checking). Предложенные алгоритмы позволяют автоматизировать проверку сформулированных свойств;

6. разработана архитектура системы поддержки разработки логических компонентов систем с пошаговыми стратегиями. Система поддерживает разработку логических компонентов при помощи предложенного метода и создание на базе этого компонента исполнимого прототипа программы;

7. разработано программное обеспечение на основе архитектуры системы поддержки разработки логических компонентов систем с пошаговыми стратегиями. Программное обеспечение позволяет создавать системы при помощи предложенного метода разработки логических компонентов, используя предметно-ориентированный язык для описания правил, и проводить верификацию при помощи предложенных алгоритмов. Разработанное программное обеспечение внедрено в практическую деятельность ОАО «Туристский информационный центр города Москвы» и деятельность компании ООО «Некки».

Основные публикации по теме диссертации:

Основные положения диссертационной работы опубликованы в 13 печатных трудах, в том числе в 4 статьях в журналах, включенных ВАК РФ в перечень ведущих рецензируемых научных журналов и изданий:

1. Гаврилов А.В., Павлова Е. А. Формализация проектирования сложных информационных систем на основе анализа функциональных интерфейсов // Информационные технологии №9, 2008. – М.: Новые технологии, 2008, с. 9-15. (Перечень ВАК)

2. Павлова Е.А. Проектирование формального предметно-ориентированного языка (DSL) для разработки правил компьютерных игр в классе пошаговых стратегий // Вестник компьютерных и информационных технологий №4, 2009 – М.: Машиностроение, 2009. – с. 45-52. (Перечень ВАК)

3. Павлова Е.А., Станкевичус А.А. Исследование сложности верификации моделей для правил логических компонентов с учётом свойств безопасности //

Безопасность информационных технологий. – М., 2009, №4. – с. 126-129. (Перечень ВАК)

4. Павлова Е.А. Автоматизация разработки правил компьютерных игр в классе пошаговых стратегий // Вестник компьютерных и информационных технологий №3, 2010 – М.: Машиностроение, 2010. – с. 45-52. (Перечень ВАК)

5. Гаврилов А.В., Павлова Е.А. Использование Domain Specific Languages для моделирования программных систем. В сб. Современные технологии в задачах управления, автоматики и обработки информации: труды XVI Международного научно-технического семинара. – Тула: Изд-во ТулГУ, 2007. – 344 с., с. 43-44.

6. Павлова Е.А. Доменно-специфичный язык для разработки компьютерных игр в жанре пошаговых стратегий. В сб. НАУЧНАЯ СЕССИЯ МИФИ-2008. Сборник научных трудов. В 15 томах. Т.11. Технологии разработки программных систем. Информационные технологии. – М.: МИФИ, 2008. 204 с., с. 58-59.

7. Павлова Е. А. Формальный предметно-ориентированный язык (DSL) для разработки компьютерных игр в классе пошаговых стратегий // Технологии Microsoft в теории и практике программирования: Труды V Всерос. конф. студентов, аспирантов и молодых ученых. Центральный регион. Москва, 1 - 2 апреля 2008 г. – М.: Вузовская книга, 2008. –с. 38-40.

8. Павлова Е. А. Формальный предметно-ориентированный язык (DSL) для описания правил компьютерных игр в классе пошаговых стратегий. Современные технологии в задачах управления, автоматики и обработки информации. Труды XVII международного научно-технического семинара. – СПб.: ГУАП, 2008, с. 260.

9. Pavlova E. The Formal Approach to Computer Game Rule Development Automation // Proceedings of the Third Spring Young Researchers' Colloquium on Software Engineering (SYRCoSE 2009). Moscow, May 28-29, 2009. – pp. 119-122.

10. Павлова Е. А. Технологии разработки современных информационных систем на платформе Microsoft.NET: Учебное пособие / Е. А. Павлова – М.:Интернет-Университет Информационных технологий; БИНОМ. Лаборатория знаний, 2009. – 112 с.: ил., табл. – (Серия «Основы информационных технологий»).

11. Павлова Е. А. Разработка формального предметно-ориентированного языка описания правил игр // Технологии Microsoft в теории и практике программирования: Труды VI Всерос. конф. студентов, аспирантов и молодых ученых. Центральный регион. Москва, 1 - 2 апреля 2009 г. – М.: Вузовская книга, 2009. –с. 175-177.

12. Павлова Е.А. Об одном подходе к верификации компьютерных игр // НАУЧНАЯ СЕССИЯ НИЯУ МИФИ-2010. XIII Международная телекоммуникационная конференция студентов и молодых учёных «МОЛОДЁЖЬ И НАУКА». Тезисы докладов. В 3-х частях. Ч.2. – Москва, НИЯУ МИФИ, 2010.– с.54-55.

13. Павлова Е. А. Метод разработки компьютерных игр класса пошаговых стратегий // Технологии Microsoft в теории и практике программирования: Труды VII Всерос. конф. студентов, аспирантов и молодых ученых. Центральный регион. Москва, 21 - 22 апреля 2010 г. – М.: Вузовская книга, 2010. –с. 123-124.

Подписано в печать __ _____ 2010 г. Формат 60х90 $^{1}/_{16}$. Объем 1 печ. л. Тираж 100 экз. Отпечатано в _____