На правах рукописи

Марков Александр Владимирович

АВТОМАТИЗАЦИЯ ПРОЕКТИРОВАНИЯ И АНАЛИЗА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С ИСПОЛЬЗОВАНИЕМ ЯЗЫКА UML И СЕТЕЙ ПЕТРИ

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

Автореферат диссертации на соискание ученой степени кандидата технических наук

Новосибирск – 2015

Работа выполнена в Федеральном государственном бюджетном образовательном учреждении высшего образования «Новосибирский государственный технический университет»

Научный руководитель: доктор технических наук, профессор Воевода Александр Александрович

Официальные оппоненты: *Хусаинов Ахмет Аксанович, доктор физико-математических наук, профессор, Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Комсомольский-на-Амуре государственный технический университет», профессор кафедры «Математическое обеспечение и применение ЭВМ»*

Непомнящий Валерий Александрович, кандидат физико-математических наук, доцент, Федеральное государственное бюджетное учреждение науки Институт систем информатики им. А.П. Ершова Сибирского отделения Российской академии наук, заведующий лабораторией теоретического программирования

Ведущая организация: Федеральное государственное бюджетное учреждение науки «Санкт-Петербургский институт информатики и автоматизации Российской академии наук», Санкт-Петербург

Защита состоится «18» июня 2015 г. в 14 часов 00 минут на заседании диссертационного совета Д 212.173.06 при Федеральном государственном бюджетном образовательном учреждении высшего образования «Новосибирский государственный технический университет» по адресу: 630073 РФ, г. Новосибирск, пр. К. Маркса 20.

С диссертацией можно ознакомиться в библиотеке ФГБОУ ВО Новосибирского государственного технического университета и на сайте http://www.nstu.ru.

Автореферат разослан «__» _____ 2015 г.

Ученый секретарь диссертационного совета  Фаддеенков Андрей Владимирович

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. В настоящее время быстрая, экономичная, успешная работа различных предприятий зависит от качественных программных приложений, которые используют в экономической, управленческой и технической деятельности. Для создания качественного продукта пользуются различными техниками разработки *программного обеспечения* (ПО) от водопадной и итерационной моделей до других различных вариантов и их модификаций. В настоящее время применяются различные способы разработки ПО, но сложность применения затрудняет их использование в реальной жизни. Зачастую при проектировании ПО прибегают к CASE-технологиям, которые описаны F.P. Broooks, А.Л. Фуксманом, А.Н. Тереховым, А.М. Вендеровым, В.В. Липаевым и др. Одним из самых популярных средств визуального моделирования является *UML* (Unified Modeling Language), предложенный группой разработчиков в *OMG* (Object Management Group) под руководством G. Booch, J. Rumbaugh, I.H. Jacobson и описанный в работах M. Fowler, K. Scott, И. Грэхэма, Д. Харела. Но, как известно, данная структура не предусматривает формальной проверки созданных систем и отдельных диаграмм. Использование сетей Петри – математического аппарата, подробно описанного в трудах W. Reisig, M.H.T. Hack, James L. Peterson, S. Haddad, G.W. Brams, В.Е. Котова и А.А. Лескина, И.А. Ломазовой, О.Л. Бандман, И.Б. Вербицкайте, В.А. Непомнящего, помогает решить проблему анализа многопоточных систем, параллельных вычислений (А.П. Ершов, Ч. Хоар, В.В. Корнеев, В.И. Воробьев, В.Э. Малышкин) и проектируемых диаграмм (M, Westergaard, L. Baresi, L.Z. Zhu).

При анализе сетей Петри, разработчик может столкнуться с трудностью, заключающейся в существенном количестве состояний проектируемой системы. Несмотря на простоту структуры некоторых сетей Петри, их пространство состояний может достигать значительных размеров, что приводит к одной из главных проблем при анализе автоматов и графов – “взрыву” пространства состояний (экспоненциальный рост количества исследуемых состояний, останавливающий процесс анализа по причине отсутствия требуемого количества памяти). Работы G.J. Holzmann, S. Cristensen и L.M. Kristensen посвящены решению данной проблемы. Даже несмотря на увеличивающиеся мощности машин, используемых при анализе, рост сложности структур и логики систем тоже возрастает, поэтому решение данной проблемы не может сводиться только к увеличению мощности анализаторов. Стоит отметить, что существуют трудности при совместном использование UML диаграмм и сетей Петри, заключающие в

отсутствии формализации правил преобразования и, следовательно, автоматической трансляции из одного вида сущности к другой.

На кафедре “Автоматика” на протяжении порядка десяти лет ведётся работа по исследованию совместного использования UML диаграмм и сетей Петри, результатом которой стала методика совместного использования UML диаграмм, описывающих статические и динамические свойства, и сетей Петри, используемых для анализа полученных диаграмм. Разработанная методика применима к задачам проектирования программных приложений для персональных компьютеров, для автоматизированных систем, для системы управляемого светофора, взаимодействия пользователя с банкоматом и др. Разработаны алгоритмы и программные продукты, нацеленные на сокращение времени проектирования ПО, а также его анализа: выявление логических ошибок (С.В. Коротиков, Д.О. Романников). Например, алгоритм автоматической трансляции поведенческих UML диаграмм в сети Петри, приложение по преобразованию графического вида к матричной форме с последующим анализом.

Цель диссертационного исследования. Разработать методику проектирования программного обеспечения с использованием UML диаграмм и сетей Петри, позволяющую находить и устранять логические ошибки (зацикливание, тупиковые маркировки, мертвые переходы), в которой в отличие от более ранних версий не используется детализированная диаграмма прецедентов и диаграмма состояний, а для анализа отдельных сценариев используется диаграмма последовательности. Разрабатываемая методика должна быть применима для разработки систем широкого круга задач: производственные процессы, приложения для персональных компьютеров, систем транспортного регулирования, систем обработки информации и распределенных систем.

Для достижения поставленной цели решаются следующие **задачи**:

– составление пошаговой процедуры проектирования ПО с использованием UML диаграмм и сетей Петри для анализа поведенческих UML диаграмм;

– разработка способа проверки достижимости заданного состояния сети, представление правил и алгоритма для его реализации;

– разработка алгоритма преобразования UML диаграмм в сети Петри и способа выполнения автоматической трансляции UML диаграмм в сети Петри с использованием форматов .xmi и .cpn;

– разработка программного обеспечения для преобразования комбинации мест и переходов маркированных сетей Петри к матричной форме: составление матриц входной и выходной функций, получение составной матрицы и вектора начальной маркировки;

– разработка способа проектирования сетей Петри для задачи создания ПО в сфере робототехники, а именно робота-манипулятора; реализация рекурсивных функций в сетях Петри;
– разработка способов анализа сетей Петри, заключающихся в проверке частей пространства состояний.

Объект исследования. Средства проектирования и анализа программного обеспечения с целью их автоматизации. **Предметом исследования** является совместное использование унифицированного языка моделирования UML и сетей Петри.

Методы исследования. При выполнении задач диссертационного исследования применялись следующие методы: объектно-ориентированный анализ; математический аппарат сетей Петри; метод "плавающей" линии (*sweep-line method*); хеширование (*bit state hashing*). При реализации примеров применения предлагаемой модифицированной методики использовались отраслевые и международные стандарты, CASE-технологии, современные инструментальные среды и пакеты моделирования: Rational Rose (среда проектирования UML диаграмм, способная преобразовывать проектируемые диаграммы в программный код основных языков программирования), Magic Draw (среда проектирования UML диаграмм с возможностью сохранять полученные диаграммы с расширением *.xmi*), CPN Tools (среда проектирования цветных сетей Петри – Colour Petri Nets Tools).

Научная новизна. В предлагаемом диссертационном исследовании выделим следующие результаты: предложена формализация правил реализации автоматической трансляции UML диаграмм в сети Петри; новый способ проверки достижимости сетей Петри через реализацию инверсии самой сети и/или её графа состояний; разработка нового способа задания исходных данных о структуре проектируемой системы с использованием информации, закладываемой в метке (интерпретация информации, передаваемой в системе и/или системой) сети Петри; проектирование регулирующих элементов поддержания давления и температуры для автоматизированных систем с использованием сетей Петри.

Личный вклад. Все основные результаты получены автором. А именно, разработана методика совместного использования UML диаграмм и сетей Петри, предложены правила инверсии сетей Петри, разработано приложение по преобразованию и анализу сетей в матричной форме. В совместной работе с Зимаевым И.В. получена взаимосвязь между статическими и динамическими диаграммами. При участии Романникова Д.О. разработаны алгоритмы для моделирования систе-

мы поддержания давления в трубопроводе. Также в совместных работах соавторам принадлежат постановка задачи и обсуждение результатов исследований.

Практическая ценность и внедрение. Проведенные исследования подтвердили эффективность применения предлагаемой методики на этапе проектирования программного обеспечения.

Результаты диссертационной работы были использованы в проектировании системы подготовки железорудных окатышей на горно-обогатительном комбинате Акционерного Общества Соколовско-Сарбайского Горно-обогатительного Объединения (АО ССГПО, Республика Казахстан, г. Рудный), в проектировании ПО локальных подсистем АСУ ТП водоснабжения (г. Тюмень), в разработке интернет сайтов в ООО «Дабаз» (г. Новосибирск), а также в конкурсе фундаментальных и прикладных исследований внутренних грантов НГТУ «Использование UML-диаграмм и аппарата сетей Петри как формальных методик анализа архитектуры программного обеспечения». Результаты диссертационного исследования используются в учебном процессе НГТУ на кафедре “Автоматика”.

Полученные результаты рекомендуется использовать при проектировании программных приложений для компьютеров в производственных сферах, для систем обработки информации и для распределенных систем: для регулирование температуры при обжиге окатышей, для регулирования давления в трубопроводе, для управления светофором, для работы банкомата, для передвижения манипулятора в ограниченном пространстве с препятствиями, а также для программных приложений.

Кроме того, исследования были поддержаны грантом на выполнение проекта, отобранного для финансирования в 2012 году в рамках реализуемой программы стратегического развития НГТУ по итогам конкурса НИОКР, определяющих формирование научно-технического задела по приоритетным направлениям развития науки. Направление 2.3.: "Информационные и цифровые технологии и системы". Работа выполнена при финансовой поддержке Минобрнауки России по государственному заданию №2014/138 тема проекта "Новые структуры, модели и алгоритмы для прорывных методов управления техническими системами на основе наукоемких результатов интеллектуальной деятельности".

На защиту выносятся следующие основные результаты и положения:

– методика проектирования ПО, включающая построение структурных UML диаграмм: классов и объектов, диаграмм поведения: прецедентов, активности и после-

довательности, две последние транслируются в сети Петри для анализа и устранения логических ошибок (зацикливания, тупиковые маркировки, мертвые переходы);

– формализованные правила для автоматической трансляции UML диаграмм в цветные сети Петри;

– рекомендации по анализу систем с большим количеством состояний;

– способ инверсии сетей Петри для проверки их основных свойств, а также анализ отдельных состояний и частей пространства состояний;

– программа для представления сетей Петри в матричной форме: составление матриц входной и выходной функций, получение составной матрицы и вектора начальной маркировки;

– способ моделирования систем со сложной структурой при использовании сетей Петри с нагруженными метками, реализация рекурсии в сетях Петри.

Апробация работы. Основные результаты диссертационной работы были представлены на следующих конференциях и семинарах: международная русско-индийская конференция “The 2nd Russian-Indian Joint Workshop on Computational Intelligence and Modern Heurisics in Automation and Robotics” (Новосибирск, 10-13 сентября 2011 г.); одиннадцатая международная научно-техническая конференция “Актуальные проблемы электронного приборостроения” (Новосибирск, 2-4 октября 2012 г.); двенадцатая международная научно-техническая конференция “Актуальные проблемы электронного приборостроения” (Новосибирск, 2-4 октября 2014 г.); III международная конференция “Современные информационные технологии и ИТ-образование” (факультет вычислительной математики и кибернетики МГУ им. М.В. Ломоносова, 8-10 ноября 2013 г.); международная научная конференция “Математическое и компьютерное моделирование” (Омский государственный университет, 18-19 октября 2013 г.); 61-я международная молодёжная научно-техническая конференция “Молодёжь. Наука. Инновации” (Владивосток, МГУ им. адм. Г.И. Невельского, 20-21 ноября 2013 г.); школа молодых учёных САИТ-2011. Секция №2 “Информационные технологии в системах автоматического и автоматизированного управления” (Новосибирск, 12-16 сентября 2011г.); XIII Международная конференция “Информатика: проблемы, методология, технологии” (Воронеж, 7-8 февраля 2013 г.); международная научно-практическая конференция “Тенденции формирования науки нового времени” (Уфа, 18 октября 2014 г.); XIII международная научно-практическая конференция “Наука и современность” (Новосибирск, 15 ноября 2011

г.), международная заочная научно-практическая конференция “Наука и техника XXI века” (Новосибирск, 14 ноября 2011 г.); результаты диссертационной работы регулярно представлялись на конференциях и семинарах НГТУ (2011-2014 гг.).

Соответствие работы научной специальности. Диссертация соответствует п. 1 «Модели, методы и алгоритмы проектирования и анализа программ и программных систем, их эквивалентных преобразований, верификации и тестирования», п. 8 «Модели и методы создания программ и программных систем для параллельной и распределенной обработки данных, языки и инструментальные средства параллельного программирования», п. 7 «Модели, методы, алгоритмы, языки и программные инструменты для организации взаимодействия программ и программных систем» паспорта специальности 05.13.11 – «Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей».

Публикации. Основные положения и результаты диссертационной работы опубликованы в 36 работах (список основных 24 работ приведен ниже), в том числе: 3 программы, зарегистрированные в Роспатент, стати в изданиях, рекомендованных ВАК РФ – 6, в сборниках научных трудов – 17, в материалах международных симпозиумов и конференций – 10.

Структура и объём диссертации. Диссертация состоит из введения, четырёх глав, заключения, библиографического списка использованной литературы и шести приложений. Работа изложена на 176 стр. машинописного текста: основное содержание на 138 стр. и включает 85 рисунков, 4 таблицы и список литературы из 150 наименований.

СОДЕРЖАНИЕ РАБОТЫ

Во **введении** обоснована актуальность темы диссертационного исследования. Определены цели и задачи, сформулированы положения диссертационного исследования, которые вынесены на защиту, их научная новизна и практическая ценность.

В **первой главе** диссертационного исследования на основе анализа подходов к разработке ПО (RUP, Agile, RAD, CMMI, MSF и др.), различных парадигм написания приложений, а также методов моделирования (семейство стандартов IDEF, UML, switch-технология, SDL**,** MSC) и анализа проектируемого программного продукта (метод *среза пучка (slice cluster)*, метод *гибридного спектрального среза (hybrid spectrum slice)*, подход *minimal-MUMCUT*, метод локализации ошибок). Остановимся на итеративной разработки ПО, объектно-ориентированном подходе и проектировании систем при использовании UML диаграмм (программные сред-

ства: *Rational Rose, MagicDraw,* Sybase PowerDesigner, Oracle Developer Suite и др.), с последующим анализом на основе математического аппарата сетей Петри (программные средства: CPN-AMI, CPN Tools, GreatSPN, PEP, WoPeD и др.).

UML – это язык графического описания систем. Для моделирования UML диаграмм выберем *Rational Rose* от компании IBM, *Magic Draw* от компании NoMagic. Данные пакеты имеют большой арсенал для моделирования диаграмм, связи их в единые проекты и преобразования полученных наборов в коды. UML включает диаграммы, описывающие структуры проектируемого ПО: диаграмма классов, диаграмма объектов, а также динамические свойства: диаграмма активности, диаграмма последовательности, диаграмма состояний.

Сеть Петри представляют четверкой $N = (P, T, F, m_I)$, где $P = \{ p_1, p_2, \dots, p_n \}$ – множество мест, $T = \{ t_1, t_2, \dots, t_m \}$ – множество переходов, таких что $P \cap T = \emptyset$, $F \subseteq P \times T \cup T \times P$ – отношение, а $m_I : P \to N$ – начальная маркировка, которая отображает размещение меток по местам сети Петри, моделирование которых выполним в CPN Tools. Для лучшего понимания элементов сети Петри и её графического представления на рисунке 1 представлена часть сеть Петри протокола передачи, которая содержит три места *P*, три перехода *T*, с начальной маркировкой m_I*: P = {F, E, S1}; T = {Recive Packet2, transmit packet2, send packet2};* m_I *= (0, 0, 1).* Основными свойствами сетей Петри является ограниченность, достижимость, покрываемость и живость.

Приведём правила срабатывания переходов: выполнение $t \in T$ в маркировке $m: P \to N$ возможно, если $m \geq \bullet t$. Если *t* разрешён в *m*, то происходит срабатывание перехода *t*, что приводит к маркировке m'. Записать это можно следующим образом: $m \left[t \right\rangle m'$, где m' определяется как $m' = \left(m - \bullet t \right) + t \bullet$ (маркировка m' получена из m, учитывая взаимосвязи до перехода t и после него). Взаимосвязи между вершинами сети (рисунок 2, а) на некоторых рисунках будем представлять, как показано на рисунке 2, б.

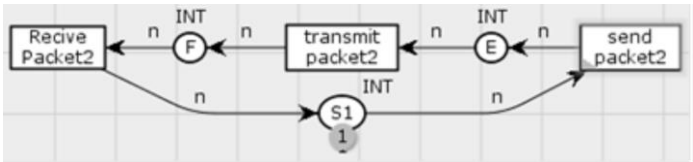
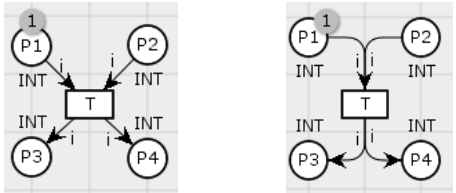


Рисунок 1 – Часть сети Петри протокола передачи данных



а) б)

Рисунок 2 – Представление взаимосвязей сети Петри: а) стандартное, б) компактное

Создание программных продуктов на основе *совместного использования UML диаграмм и сетей Петри* разделяют на несколько этапов: выбор и разработка необходимых UML диаграмм, трансляция полученных диаграмм в сети Петри, анализ сетей Петри и внесение необходимых изменений в UML диаграммы в соответствии с результатами анализа.

Поскольку динамические модели UML не имеют достаточно формальной семантики, работы C. Eichner, H. Fleischback, S. Emadi, F. Shams, M. Kessentini, A. Bouchoucha посвящены анализу диаграмм последовательности с использованием сетей Петри по приведенным ими правилам преобразования. В работах J. Merseguer, J. Campos, T. Miyamoto, H. Kurahata, I. Rabova рассматривается преобразование диаграммы состояний и диаграммы активности в сети Петри для проверки проектируемых диаграмм по предложенным правилам. В работах M. Noguerra, M.V. Hurtado описывается моделирование системы при помощи языка OWL-DL с последующим преобразованием в UML диаграммы и сеть Петри, приводится описание элементов UML, OML и CPN в виде сводной таблицы. Методика проектирования ПО с использованием UML диаграмм и сетей Петри, предложенная С.В. Коротиковым, состоит из двух этапов: проектирование статических и динамических свойств системы. В данном диссертационном исследовании предлагаются правила преобразования UML диаграмм в сети Петри и показана корректность данного преобразования, с использованием предложенной методики анализируются сети Петри со значением пространства состояний порядка 10^4, а при дальнейшем развитии удалось достигнуть анализа сетей Петри со значением пространства состояний порядка 10^5. Развитие методики, предложенной Д.О. Романниковым, заключалось в объединении двух этапов и детализации некоторых диаграмм. Тем не менее, использовались диаграммы, наличие которых необязательно. По этой причине модифицируем методику: используем диаграмму прецедентов (*USD*), диаграмму классов (*ClD*) и диаграмму объектов (*ObD*), которые при необходимости могут быть модифицированы после моделирования диаграммы активности (*AcD*), диаграммы последовательности (*SeqD*) и анализа их в сетях Петри. В одной из первых работ на основе предлагаемой методики выполняется проектирование ПО на примере современной станции управления лифтом, для которой разрабатывается расширение для пакета MATHCAD при решении задачи полиномиального синтеза. Ключевым моментом методики является отказ от моделирования диаграммы состояний (*StD*), которую заменяет исследование пространства состояний сети Петри, полученной из диаграммы активности. Поскольку в перечисленных работах упоминалось о необходимости автоматической трансляции UML диаграмм в сети Петри, предлагается способ выполнения данной

процедуры, для чего основные правила преобразования представляются в алгоритмическом виде. Разработаем алгоритмы для основных правил преобразования диаграмм активности в сети Петри и способ автоматической трансляции

Сети Петри разрабатывались с целью проектирования многопоточных систем и их последующего *анализа*, для чего используют, например, программный пакет CPN Tools. Основной способ анализа сетей Петри – построение и исследование графа состояний, который является четверкой (V, E, src, trg), где V – множество вершин, E – множество рёбер между вершинами, а $src, trg : E \to V$ отображает в каждом ребре вершину, из которого оно получено, и вершину, к которому приводит выполнение срабатывания перехода, соответственно. Пусть $m, m' \in V$ два состояния, а $t \in T$ – переход. Если $(m, t, m') \in E$, тогда срабатывание t в m приведёт к m' или кратко $m \xrightarrow{t} m'$. Последовательность состояний m_i и переходов t_i может быть записана, как последовательность вида: $m_1 \xrightarrow{t_1} m_2 \xrightarrow{t_2} m_3 \xrightarrow{t_3} \ldots \xrightarrow{t_{n-1}} m_n \xrightarrow{t_n} m_{n+1}$. Стандартный способ построения графа состояний представим в виде алгоритма 1.

Алгоритм 1 – Построение графа состояний, уточненного и дополненного комментариями

$Require: (P,\ T,\ F,\ m_I)$ // дана сеть Петри,

$Ensure: (V, E)$ // необходимо исследовать соответствующий граф состояний,

1: $V := \{ m_I \}$ // переменной V (множество посещённых состояний) присваивается множество с единственной переменной – начальное состояние m_I,

2: $W := \{ m_I \}$ // переменной W (множество не посещённых состояний) присваивается множество с единственной переменной – начальное состояние m_I,

3: $E := \emptyset$ // переменной E (множество рёбер) присваивается значение $\emptyset$,

4: *while* $W \neq \emptyset$ *do* // пока множество W не станет пустым, выполняем следующие действия,

5: *Select an* $m \in W$ // выбираем состояния m из множества W,

6: $W := W \setminus \{ m \}$ // убираем из множества W состояние m,

7: *for all* t, m' *such that* $m \xrightarrow{t} m'$ *do* // для всех t, m', удовлетворяющих $m \xrightarrow{t} m'$, выполняем следующие действия,

8: $E := E \cup \{ (m, t, m') \}$ // добавляем новое ребро во множество,

9: *if* $m' \notin V$ *then* // если найдено новое состояние, не входящее в множество V, тогда,

10: $V := V \cup \{ m' \}$ // добавляем его к этому множеству,

11: $W := W \cup \{ m' \}$ // и к множеству W,

12: *return* (V, E) // возвращаем обновленные значения переменных V и E.

Основной проблемой при анализе систем является возможность “взрыва” пространства состояний, которая заключается в экспоненциальном росте количества состояний $\exp\left(|P|\right) = |V|$, что приводит к досрочному завершению анализа из-за недостатка нужных объёмов оперативной памяти, а использование постоянной памяти приведёт к значительному росту времени. Таким образом, без оценки остаётся значительная часть пространства состояний, в которой могут присутствовать зацикливания $m' \to {}^*m'$ и тупиковые маркировки $m' \in V,\ m' = m_n,\ \left(m_n, t_n, m_{n+1}\right), t_n \notin T$, что свидетельствует о некорректном построении системы. Существуют нестандартные способы анализа пространства состояний: *sweep-line method* (метод “плавающей” линии или метод линии-диапазона) и *bit-state hashing* (побитовое хеширование состояний) и их возможные модификации. Анализ проектируемых сетей Петри осуществляется в программной среде CPN Tools при генерации и исследовании пространства состояний с последующим построением соответствующего отчета, в котором отображаются тупиковые маркировки, мертвые переходы, зацикливания.

Во **второй главе** приводится способ проектирования сетей Петри для задачи перемещения манипулятора в ограниченном пространстве с препятствиями. Для решения данной задачи предлагается использовать *нагруженные метки* (метка с дополнительной информацией об истории её передвижения или о структуре системы). Рассмотрим использование нагруженных меток на примере однозвенного манипулятора, перемещаемого в ограниченном пространстве. Для моделирования данной задачи используются метки с составным множеством цветов, т.е. составным множеством типов данных, для отображения ограниченного пространства (лабиринта) – *1`(1, 1, “0”) ++ 1`(2, 1, “0”) ++ ... ++ 1`(7, 7, “0”)* и однозвенного манипулятора (рисунок 3) – место *Robot 1`(5, 4, 5, 3, “0”, “L1: ur: dr: R: dr: D:”, “L2: R: ur: dr: R: dr:”)*.

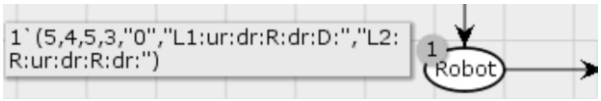


Рисунок 3 – Нагруженная метка: 5,4,5,3 – координаты положения манипулятора, *L1*, *L2* – история перемещения манипулятора

Первые две цифры у метки в месте *Robot* соответствуют координате первого элемента конструкции робота *(i, j)* – переменные типа *integer*, третья и четвёртая цифра иллюстрируют положение второго звена в лабиринте. С помощью последних двух типов данных *L1* и *L2* можно проследить историю передвижений первого и второго элемента, соответственно – переменные типа *string*.

Анализ пространства состояний полученной сети осуществлялся через автоматическую генерацию отчета о пространстве состояний, которое вычислено частично и содержит 21 687 состояний и 56 643 дуги, в сети отсутствуют мёртвые переходы. При усложнении конструкции ограниченного пространства количество состояний значительно возрастает и превышает количество порядка 10^6.

Отрабатывается возможность использования *рекурсивных функций* с использованием сетей Петри на примере переноса упорядоченного массива с ограничением выборки по одному элементу, нахождения выбранного числа из ряда Фибоначчи, а также вычисления факториала числа. На рисунке 4 представлена сеть Петри для рекурсивной задачи нахождения факториала числа. Место *Number* содержит метку *1`(12, 0)*, в которой показано искомое число факториала и произведение предыдущих двух чисел, место *List* предназначено для хранения получаемых значений произведения, получаемых в процессе нахождения факториала числа. Места *Complete*, *Number* и *List*, выделенные свечением являются местами сети с найденным решением задачи.

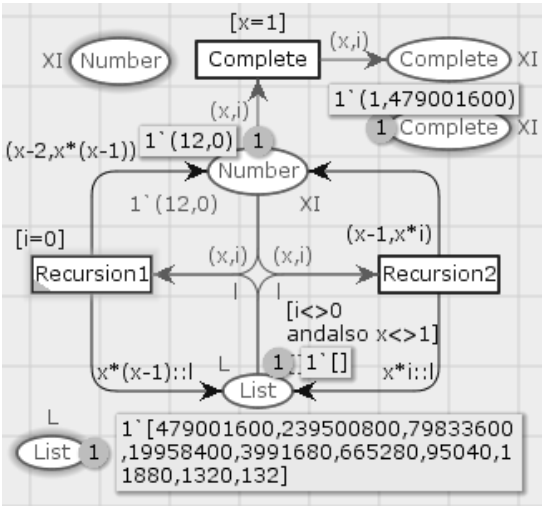


Рисунок 4 – Пример рекурсивной функции: вычисление факториала

Предлагается *способ автоматического преобразования UML* диаграммы активности *в сеть Петри*, который основан на формализации правил, ранее представленных в графическом виде и выполнении требований при проектировании диаграммы активности. Инструментом для автоматического преобразования диаграммы активности в сети выберем преобразование двух форматов с подобной структурой *.xmi* (содержится полная информация, как о создаваемых проектах, так и об отдельных диаграммах, использует синтаксис языка *xml*) и *.cpn* (использует синтаксис языка *xml*), что позволит сократить время на ручное преобразование диаграмм. Рассмотрим одно из наиболее часто встречающихся правил преобразования UML диаграмм активности в сети Петри и предложим алгоритмическую форму представления данного правила: состояние ожидания трансформируется в место $\left(SS \coloneqq SP \right) \vee \left(ES \coloneqq EP \right) \vee \left(AS_i \coloneqq PP_i \right)$, а состояние действия преобразуется в переход $\left(UST \coloneqq PST \right) \vee \left(UET \coloneqq PET \right) \vee \left(UT_i \coloneqq PT_i \right)$, который начинает действие (алгоритм 2).

Алгоритм 2 – Преобразование последовательности состояний из UML диаграммы в сеть Петри

1: $SS := SP$ // SS – начальное состояние на диаграмме активности, SP – начальное место в сети Петри,

2: $ES := EP$ // ES – конечное состояние на диаграмме активности, EP - конечное место в сети Петри,

3: $UST := PST$ // UST – стартовый переход на диаграмме активности, PST – начальный переход в сети Петри,

4: $UET := PET$ // UET – конечный переход на диаграмме активности, PET – конечный переход в сети Петри,

5: $A := \{ AS_i \}$ // переменная A – множество всех состояний действия на диаграмме активности,

6: $T := \{ UT_i \}$ // переменная T – множество всех переходов системы на диаграмме активности,

7: *while* $A \neq \emptyset$ *do* // пока A не равна $\emptyset$, выполняем …

8: *select an* $AS_i \in A$ // выборку всех состояний действия,

9: $AS_i := PP_i$ // трансформируем каждое состояние действия диаграммы активности в эквивалентное место сети Петри,

10: *return* PP_i // возвращаем значение место сети Петри

11: *while* $T \neq \emptyset$ *do* // пока T не равна 0, выполняем …

12: *select an* $UT_i \in T$ // выборку всех переходов диаграммы активности,

13: $UT_i := PT_i$ // и приравниваем их к соответствующим переходам сети Петри,

14: *return* PT_i // возвращаем значения перехода сети Петри.

Опираясь на анализ способов проектирования при совместном использовании UML диаграмм и сетей Петри, приведём пошаговую ***методику*** проектирования программного обеспечения, основанную на совместном использовании UML диаграмм и сетей Петри, ***в виде алгоритма*** на формализованном языке:

analysis { // в функции анализа выполняются следующие процедуры
translation diagram to Petri net; // транслируют диаграмму активности в сеть Петри
analysis Petri net // анализируют полученную сеть Петри
if results = ¬*satisfactory, then*; // если результаты неудовлетворительны, то
correct Activity Diagram; // выполняется корректировка диаграммы активности
else // иначе
end if } // условие выполнено

1: *do Use Case Diagram* // проектируем диаграмму прецедентов,

2: *do Class Diagram* // проектируем диаграмму классов,

3: *for needed class* ∈ *Class Diagram do Activity Diagram* // при необходимости для некоторых классов проектируем диаграмму активности,

4: *analysis* () *of Activity Diagram* // для полученной диаграммы активности выполняем функцию *analysis*,

5: *do Object Diagram* // проектируем диаграмму объектов,

6: *for needed object* ∈ *Object Diagram do Activity Diagram* // при необходимости для некоторых объектов проектируем диаграмму активности,

7: *analysis* () *of Object Diagram* // для полученной диаграммы активности выполняем функцию *analysis*,

8: *do Activity Diagram for system* // проектируем диаграмму активности,

9: *for needed scenario* ∈ *Activity Diagram do Sequence diagram* // при необходимости для некоторых сценариев, представленных на диаграмме активности, проектируем диаграмму последовательности,

10: *analysis*() *of Sequence diagram* // для полученной диаграммы последовательности выполняем функцию *analysis*,

11: *analysis*() *of Activity Diagram for system* // для полученной диаграммы активности системы выполняем функцию *analysis*,

12: *generation code* // происходит генерация кода.

На одиннадцатом шаге предлагаемой методики *analysis () of Activity Diagram for system* происходит анализ сетей Петри, одним из интересных способов видится анализ *свободного языка сетей Петри* (множество всех последовательностей срабатываний переходов сети от начальной маркировки до всех достижимых разметок сети). *Свободный язык сетей Петри* предлагается *в компактной форме* с целью анализа проектируемых сетей, который рассматривается на примере управляемого светофора и двухсимочного телефона (рисунок 5), сеть последнего состоит из двух мест *P*, шести переходов *T* и начальной маркировки m_I :

$P = \{ Mobile\ phone\_1, Session\ of\ phone\_2, SMS\ archive\_3, Action\_4 \};$
$T = \{ Ringing\ or\ onternet\_1, Ringing\ to\ first\ SIM\_2, Ringing\ to\ second\ SIM\_3, Internet\ to\ first\ SIM\_4, Internet\ to\ second\ SIM\_5, SMS\ to\ first\ SIM\_6, SMS\ to\ second\ SIM\_7 \};$
$m_I = (6, 0, 0\ 3).$

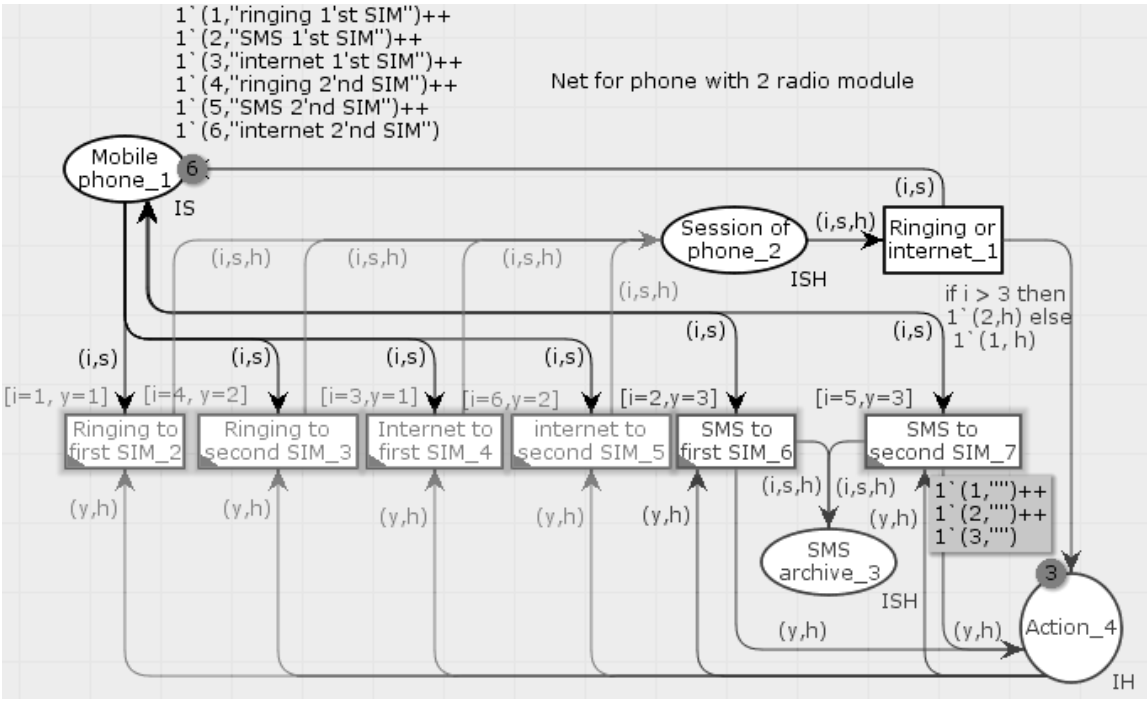


Рисунок 5 – Сеть Петри двухсимочного телефона

Для компактного отображения свободного языка сетей Петри переименуем переходы сети в соответствии с их порядковым номером: $\ldots SIM\_n = tn$. Свободный язык сети, который имеет вид: *t2, t3, t4, t5, t6, t7, t2t1, t2t6, t2t7, t3t1, t3t6,*

t3t7, t4t1, t4t6, t4t7, t5t1, t5t6, t5t7, t6t2, t6t3, t6t4, t6t5, t6t6, t6t7, t7t2, t7t3, t7t4, t7t5, t7t6, t7t7 ..., представим более компактно (рисунок 6).

Таким образом, представлена методика совместного использования UML диаграмм и сетей Петри, модификация которой заключается в отказе от использования диаграммы состояний, которую заменяют построением и исследованием пространства состояний сети Петри, и от детализированной диаграммы прецедентов. Предложен способ автоматического преобразования диаграмм в сети, используя схожую структуру двух форматов *".xmi"* (UML) и *".cpn"* (сети Петри), а также способ проектирование систем со сложной структурой для задачи перемещения манипулятора в ограниченном пространстве с препятствиями на основе сетей Петри с нагруженными метками. Показано выполнение рекурсии в сетях Петри при нахождении факториала числа и описан способ анализа при использовании свободного языка сетей Петри, при котором используется сжатую форму для более компактного представления.

В **третьей главе** предлагается способ, основанный на *построении отдельных сценариев* работы системы, который позволяет исследовать сценарии работы, наличие ошибок в которых является критическим для системы. Рассмотрим один из сценариев работы на примере взаимодействия пользователя с банкоматом, а именно проверки баланса счета (рисунок 7), при котором пользователь безошибочно с первого раза вводит PIN-код (*Inject PIN, PIN correct*), проверяет баланс (*Checking balance*), печатает чек (*Print chq I*) и забирает пластиковую карту (*Taking card*). Проектируемую диаграмму последовательности транслируем в соответствующую сеть Петри (рисунок 8), а пространство состояний данного сценария содержит шесть узлов и шесть дуг и выглядит следующим образом:

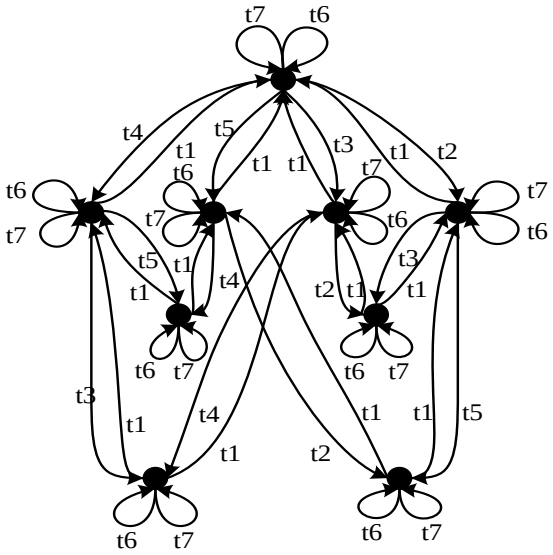


Рисунок 6 – Компактное представление свободного языка сети Петри двухсимочного телефона

$$V = \{\, m_1(1,0,0,0,0,0),\ m_2(0,1,0,0,0,0),\ m_3(0,0,1,0,0,0),$$
$$m_4(0,0,0,1,0,0),\ m_5(0,0,0,0,0,1),\ m_6(0,0,0,0,1,0)\};$$
$$E = \{\left(m_1\left[Insert\ card\right\rangle m_2\right),\ \left(m_2\left[Inject\ PIN\right\rangle m_3\right),\ \left(m_3\left[PINcorrect\right\rangle m_4\right),$$
$$\left(m_4\left[Taking\ card\right\rangle m_5\right),\left(m_4\left[Checking\ balance\right\rangle m_6\right),\ \left(m_6\left[Print\ chq\ I\right\rangle m_4\right)\}.$$

При необходимости *“склеивания”* (при совпадении маркировки у нескольких состояний необходимо оставить одно из них с учетом существующих взаимосвязей) графов состояний разных сценариев, необходимо учитывать количество разрядов у состояний. При различном количестве разрядов у маркировки состояний в каждом графе невозможно соединить их непосредственно. Для этого предлагается использовать одинаковое количество разрядов в маркировке, т.е. необходимо добиться одинакового количества мест для сети каждого из сценариев, которое будет равно количеству мест на их общей сети Петри. При отсутствии конкретного места в сети Петри для данного сценарии используется нулевая маркировка.

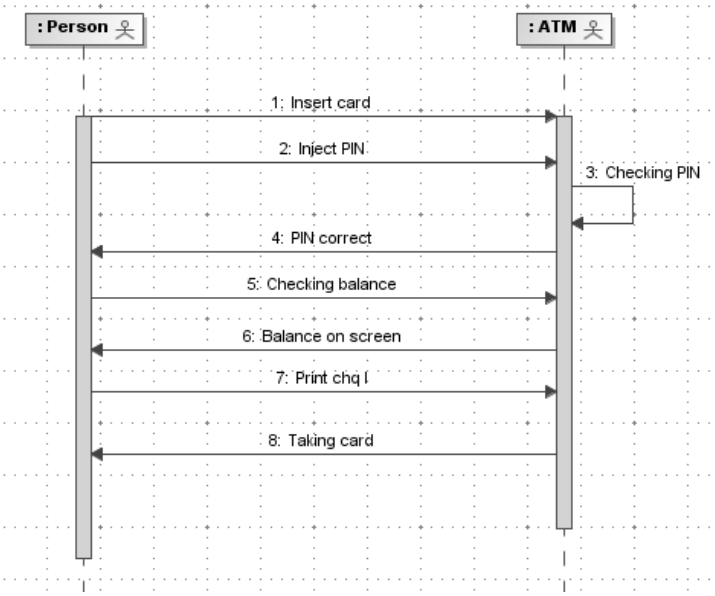


Рисунок 7 – Диаграмма последовательности сценария проверки баланса

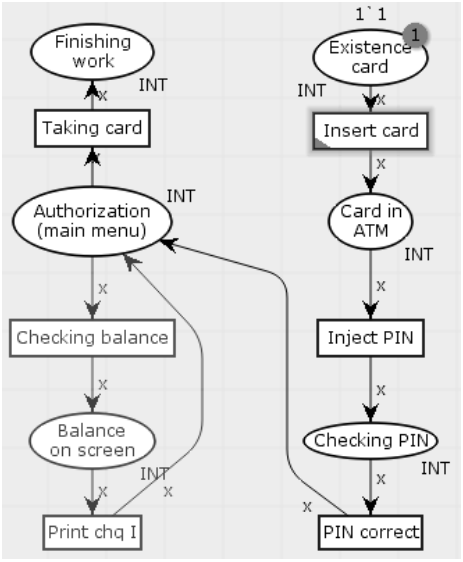


Рисунок 8 – Сеть Петри сценария проверки баланса

Альтернативным способом анализа пространства состояний является *представление* всей *структуры* системы, а именно сети Петри *в иерархическом виде*, с последующим разбиением некоторых частей на отдельные подсети. Если результаты, полученные при анализе свойств подсетей и основной сети, удовлетворяют по качеству желаемым, то можно утверждать, что при анализе свойств иерархической сети получают подобные результаты, но стоит отметить, что анализ подсети нужно начинать со всех возможных входных состояний в эту подсеть. Продемонстрируем данный способ на примере интернет-магазина в виде иерархической сети Петри (рисунок 9). Результаты анализа иерархической сети, подсетей и главной сети по отдельности, совпадают: в системе имеется две мертвые маркировки (выход из системы), отсутствуют зацикливания, все переходы могут сработать.

Вариант анализа сетей Петри, который основан на предложенных выше способах, заключается в *анализе некоторых частей пространства состояний* наиболее критичных к ошибкам, причем, данные части (их начальные состояния) могут

быть выбраны произвольно. Анализ любого участка пространства состояний возможен при соблюдении некоторых условий: проверка достижимости начального состояния и отсутствие возвратных взаимосвязей с более ранними состояниями, при наличии таких связей их необходимо перенести в анализируемую часть.

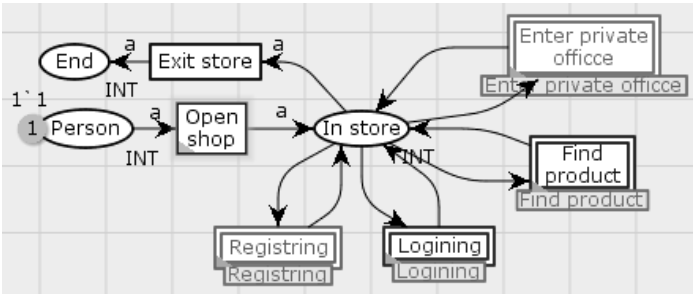


Рисунок 9 – Главная страница иерархической сети интернет-магазина

Предлагается *инверсия сетей Петри* для проверки достижимости выбранного состояния, которая заключается в изменении расположения, либо ориентации, элементов сети Петри в особом порядке, нарушающем обычный (прямой) порядок, тем самым, способствуя движению меток в обратном направлении, что приводит к начальному состоянию сети (алгоритм 3).

Алгоритм 3 – Инверсия простой ординарной сети Петри

1: *while* $F \subseteq \underset{t}{P \times T \cup T \times P'}$ // пока существует цепочка данного вида,

2: *select* $p \to p'$ // выбираем последовательно цепочку данного вида,

3: *for all* (t, p') *such that* $p \overset{t}{\to} p'$ *do invert* $p' \overset{t}{\to} p$ // реализуем инверсию,

4: *return* (P, T) // возвращаем преобразованные данные.

Стоит отметить, что прямая инверсия, при которой происходит смена ориентации дуг между вершинами сети, возможна только для простых ординарных сетей, а при наличии условий у переходов и дуг, стоит ввести определенные правила, выполнение которых позволит получить начальную маркировку.

Инверсия графа состояний – это один из способов проверки достижимости выбранного состояния, который заключается в смене ориентации взаимосвязей у вершин и является однозначным в преобразовании, но требующим восстановления сети после реализации. Инверсия графа состояний (рисунок 10) с разметкой, использующей четверичную систему исчисления, продемонстрируем на примере протокола передачи данных (сплошными линиями отобразим взаимосвязи между элементами в исходном графе состояний, а пунктирными – в инвертированном).

Предлагается *приложение, преобразующее сети Петри*, спроектированные в программной среде CPN Tools (version 3.4.0), из графической *в матричную форму* $\left(P, T, D^{-}, D^{+}\right)$, предложенную Дж. Питерсоном, и эквивалентную стандартной форме, что позволяет дать определения для сети в терминах векто-

ров и матриц. Отличие заключается лишь в появлении двух матриц D^- и D^+, представляющих входную и выходную функции, которые анализируются посредством задаваемых векторов запусков.

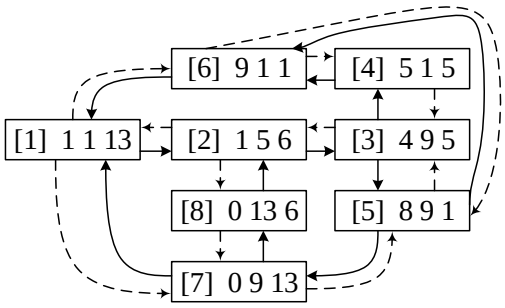


Рисунок 10 – Граф состояний протокола передачи данных

Предложены способы исследования сетей Петри и пространств состояний, при которых происходит анализ отдельных сценариев системы и различных частей пространства состояний. Для последнего способа необходима проверка достижимости выбранного состояния, с которого начнется анализ, которая осуществляется с использованием инверсии системы, т.е. сменой ориентации взаимосвязей между вершинами ординарной сети, для простых сетей предлагается воспользоваться правилами. Также описана инверсия графа состояний, как однозначной процедуры, но более затратной относительно временных ресурсов. Разработано приложение, при использовании которого анализируют сеть Петри на основе матриц входной и выходной функций, а также результирующей матриц и векторов начальной и полученной маркировки.

В четвёртой главе на основе модифицированной методики совместного использования UML диаграмм и сетей Петри проектируется программное обеспечение для автоматизированной системы регулирования температуры обжига железорудных окатышей. *В процессе обжига окатыши* проходят пять технологических зон: сушка, подогрев, обжиг, рекуперация, охлаждение. В каждой зоне поддерживается определенный температурный и газовый режим. Температуру в зоне обжига необходимо поддерживать в диапазоне $1250 \div 1270°C$, что является одним из самых сложных процессов при обжиге окатышей.

Диаграмма активности проектируется после получения диаграмм, описывающих статические свойства системы, и транслируется в соответствующую сеть Петри. После нескольких изменений в сети Петри, а именно корректировки регулятора, отрабатывается алгоритм поддержания температуры (рисунок 11). На основе системы переходов в сети Петри проектируется регулятор, изменяющий подачу газа в зависимости от изменений подачи железорудных окатышей. Каждый из переходов *Change corner_n* регулирует температуру при её отклонении на определённое значение. После, в соответствии с сетью Петри, изменяется диаграмма активности.

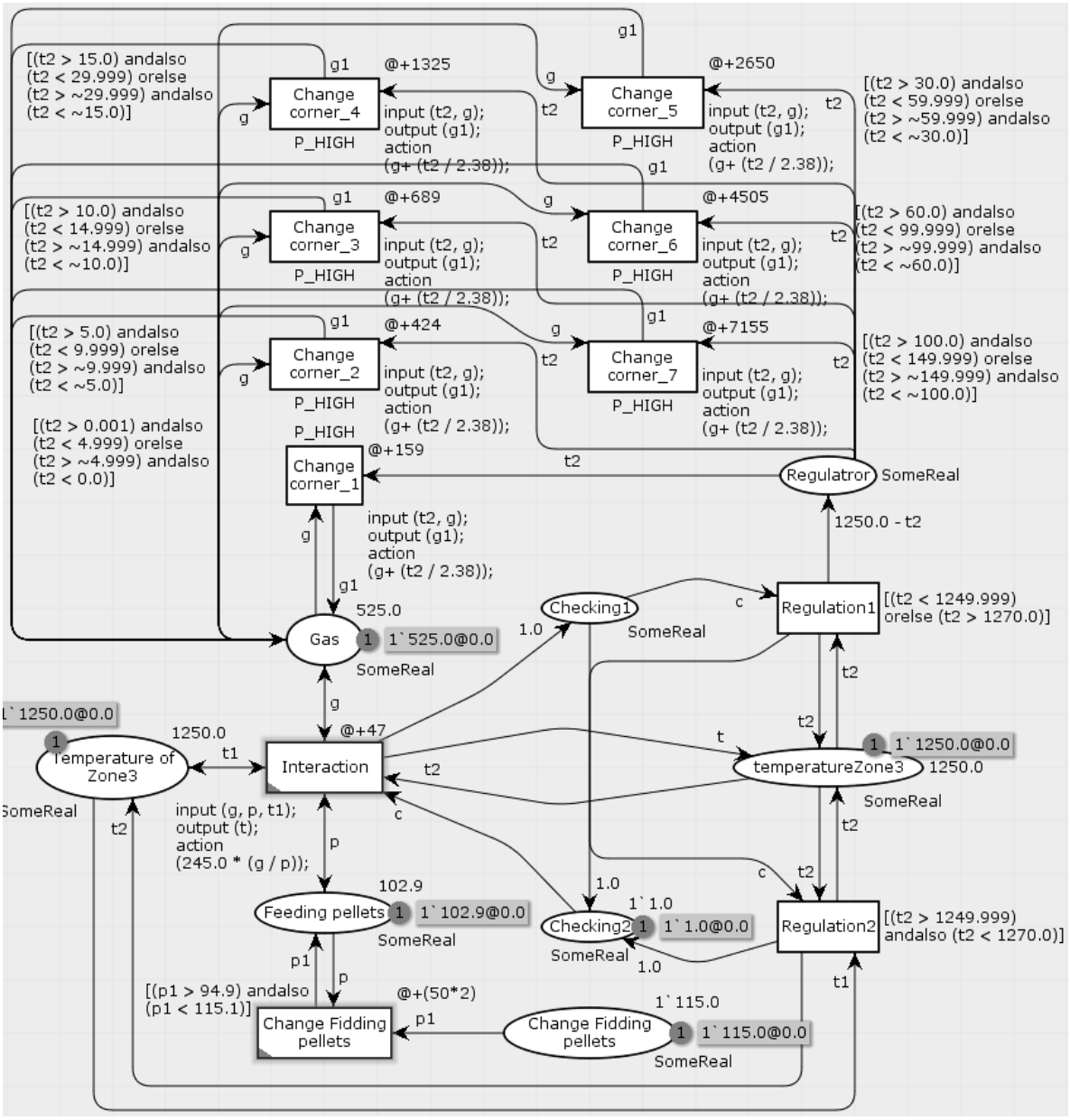


Рисунок 11 – Сеть Петри регулирования температуры в зоне обжига печи

При проектировании ПО для *АСУ ТП водоснабжения*, а именно регулирования давления в трубопроводе, используется предложенная методика совместного использования UML диаграмм и сетей Петри. Регулирование давление предлагается реализовать через *систему переходов*, добавленных к сети Петри, изменяющих мощность двигателя насоса и *имеющих несколько уровней дискретности*. Данная сеть Петри состоит из девяти основных мест P, 16 переходов T с начальной маркировкой m_I:

$$P = \{ FromPS, VNSPump1, VNSPump2, VNSValve1, VNSValve2, VNSValve3, RtoUser1, R1toUser1, R2toUser1, User1, Distrubance \};$$

$$T = \{ PStoVNSP1, PStoVNSP2, PStoVNSV3, VNSP1toVNSV1, VNSP2\ to, VNS2, VNS1toU, VNS2toU, VNS3toU, Controller\ P1, Controller\ P2,$$

$P1RtoUser1,\ RtoUser1,\ P2RtoUser1,\ Power\ below of\ VNSP1,\ Power\ above\ of\ VNSP1,\ Power\ below of\ VNSP2,\ Power\ above of\ VNSP2\}$;

$m_I = (1,1,1,1,1,1,1,1,1,1)$.

После проведения анализа получены результаты, которые помогли скорректировать сеть и тем самым избежать ложных состояний, которые могли навредить работе системы. Например, при открытии и закрытии задвижек оператором, данные элементы получали текущие состояния до выбранного действия. Анализ показал, что состояния задвижек должны меняться. После корректировки ошибок работа системы начала выполняться корректно. Анализ проектируемой сети Петри выполнен на вычислительной машине Intel Core i5-3330 CPU 3.00GHz, ОЗУ – DDR3 8Gb с операционной системой Windows 8.1 x64 и показал, что для исследования 100 000 состояний одной локальной водонапорной станции потребовалось 6 310 с, двух – 6 234с, трёх – 28 965 с. Поскольку количество состояний для одной водонапорной станции превышает значение 10^5, то можно утверждать, что для десяти водонапорных станции и всей системы водоснабжения значение будет превышать порядок 10^6.

Используется матричное представление сетей Петри, которое описано в главе 3. На примере двухсимочного телефона продемонстрируем данное представление, полученное на основе разработанного приложения, которое выглядит следующим образом:

$$D^- = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}; \quad D^+ = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} \end{array}; \quad D = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 1 & -1 & 0 & 1 \\ -1 & 1 & 0 & -1 \\ -1 & 1 & 0 & -1 \\ -1 & 1 & 0 & -1 \\ -1 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array}; \quad M = (6 \quad 0 \quad 0 \quad 3);$$

где D – составная матрица изменений, а M – вектор начальной маркировки.

На основе модифицированной методики совместного использования UML диаграмм и сетей Петри спроектировано ПО для автоматизированной системы обжига железорудных окатышей и АСУ ТП водонапорной станции. Для каждой из систем построены диаграмма прецедентов, диаграмма классов, диаграмма объектов и диаграмма активности, последняя транслируется в сеть Петри по правилам, описанным алгоритмически во второй главе. Полученные сети анализируются через автоматическую генерацию и исследования пространства состояний. Приведено матричное представления сетей Петри на примере управляемого светофора, которое получено из графического посредством разработанного приложения по преобразованию сетей Петри, проектируемых в программной среде CPN Tools.

В приложениях приводятся копии актов внедрения, копии свидетельств о регистрации программ, дается анализ и сравнение форматов *.xmi* и *.cpn*, правила реализации инверсии сетей Петри, представление и анализ сетей Петри в виде матриц, алгоритмы исследования пространства состояний.

ЗАКЛЮЧЕНИЕ

В результате анализа основных подходов к разработке ПО, различных парадигм написания приложений, а также методов анализа проектируемого программного продукта как с точки зрения написания кода, так и управления процессом, получены следующие результаты:

1. Разработана методика проектирования ПО (архитектуры и исполняемого поведения), состоящая из семи этапов и включающая разработку UML диаграмм (прецедентов, классов, объектов, последовательности и активности) и сетей Петри для анализа диаграммы активности и последовательности.
2. Разработан алгоритм и правила реализации инверсии в сетях Петри для проверки достижимости выбранного состояния сети.
3. Предложены алгоритмы для преобразования UML диаграмм в сети Петри (действие, выполнение условия, разделение/слияние), а также способ выполнения автоматической трансляции UML диаграммы активности в сети Петри посредством подобной структуры форматов.
4. Разработано программное обеспечение для преобразования комбинации мест и переходов маркированных сетей Петри к матричной форме, предложенного Дж. Питерсоном.
5. Разработан способ проектирования сетей Петри, при котором задание исходных данных о структуре проектируемой системы производится без использования мест и переходов, а при помощи информации, хранящейся в метках, что демонстрируется на примере перемещения манипулятора в ограниченном пространстве. Представлена реализация рекурсивных функций в сетях Петри.
6. Разработаны способы анализа сетей Петри: анализ пространства состояний посредством моделирования сценариев работы проектируемой системы, представление сетей Петри в иерархическом виде с целью анализа отдельных подсетей, анализ отдельно взятых частей пространства состояний при условии выполнения достижимости их начальной маркировки и отсутствия возвратных рёбер.
7. Предложено моделирование работы регуляторов с использованием сетей Петри, которое показано на примерах поддержания давления в трубопроводе при использовании дискретных величин и температуры в зоне обжига печи на основе дискретизированных аналоговых величин.

Основные публикации по теме диссертации

Статьи в журналах, рекомендованных ВАК для публикации результатов диссертаций на соискание ученой степени доктора и кандидата наук:

1. Воевода, А.А. Методика автоматизированного проектирования программного обеспечения функционирования сложных систем на основе совместного использования UML диаграмм и сетей Петри [Текст] / А.А. Воевода, **А.В. Марков**. – Современные технологии. Системный анализ. Моделирование. – 2014. – №2(42). – С. 110–115.

2. Воевода, А.А. Разработка программного обеспечения: проектирование с использованием UML диаграмм и сетей Петри на примере АСУ ТП водонапорной станции [Текст] / А.А. Воевода, **А.В. Марков**, Д.О. Романников. – Труды СПИИРАН. – 2014. – №3(34). – С. 218–231.

3. **Марков, А.В.** Алгоритм автоматической трансляции диаграммы активности в сеть Петри [Текст] / **А.В. Марков**, Д.О. Романников. – Доклады АН ВШ РФ. – 2014. – №1(22). – С. 104–112.

4. **Марков, А.В.** Инверсия простой ординарной сети Петри [Текст] / **А.В. Марков**, А.А. Воевода. – Науч. вест. НГТУ. – 2013. – №4(53). – С. 215–218.

5. **Марков, А.В.** Проверка достижимости маркировки сетей Петри при помощи инвертирования деревьев состояний для протокола передачи данных [Текст] / **А.В. Марков**, А.А. Воевода. – Доклады ТУСУР. – 2014. – №1(31). – С. 143–148.

6. Романников, Д.О. Пример применения методика разработки ПО с использованием UML-диаграмм и сетей Петри [Текст] / Д.О. Романников, **А.В. Марков**. – Науч. вест. НГТУ. – 2012. – №1(46). – С. 175–181.

Свидетельства о государственной регистрации программ для ЭВМ:

7. **Марков А.В.** Приложение для преобразования графического представления сетей Петри в матричную форму / **А.В. Марков**, А.А. Воевода // Свидетельство о государственной регистрации программы для ЭВМ №201461133. – М.: Федеральная служба по интеллектуальной собственности (Роспатент). – 2014.

8. Романников Д.О. Алгоритм управления насосным агрегатом водонапорной станции / Д.О Романников, А.А. Воевода, **А.В. Марков** // Свидетельство о государственной регистрации программы для ЭВМ №2012618139. – М.: Федеральная служба по интеллектуальной собственности (Роспатент). – 2012.

9. Шоба Е.В. Расширение пакета MATHCAD для решения задачи полиноминального синтеза / Е.В. Шоба, А.А. Воевода, **А.В. Марков**, В.В. Вороной // Свидетельство о государственной регистрации программы для ЭВМ №2013614151. – М.: Федеральная служба по интеллектуальной собственности (Роспатент). – 2013.

Основные научные публикации в других изданиях:

10. **Марков, А.В.** Описание приложения, преобразующего графическое представление сетей Петри к матричной форме [Электронный ресурс] / **А.В. Марков**, А.А Воевода. – VIII Международная конференция «Современные информационные технологии и ИТ-образование», 8-10 ноября 2013 г. – Москва: МГУ, 2013. – С. 264–269. – Режим доступа: http://conf.it-edu.ru/sites/default/files/elektronnyy_sbornik_tom_1.pdf, свободный.

11. **Марков, А.В.** Описание структуры АСУ ТП водонапорной станции при помощи UML диаграмм [Текст] / **А.В. Марков**, А.А. Воевода, Д.О. Романников. – Двенадцатая международная научно-техническая конференция “Актуальные проблемы электронного приборостроения”, 2-4 октября 2014 г. – Новосибирск: Изд-во НГТУ, 2014. – С. 65–67.

12. **Марков, А.В.** Обзор работ журнала Journal of Systems and Software за 2012-2014 годы, посвященных анализу программного обеспечения [Текст] / **А.В. Марков,** Д.О. Романников. – Сб. науч. тр. НГТУ. – 2014. – №3 (77). – С. 125 – 136.

13. **Марков, А.В.** Применение UML-диаграмм и сетей Петри для проектирования ПО технологического процесса обжига окатышей [Текст] / **А.В. Марков.** – Сб. науч. тр. НГТУ. – 2014. – №3 (77). – С. 99–118.

14. **Марков, А.В.** Поиск манипулятором кратчайшего пути в лабиринте [Текст] / **А.В. Марков**. – Сб. науч. тр. НГТУ. – 2011. – №4(66). – С. 75–90.

15. Воевода, А.А. Рекурсия в сетях Петри [Текст] / А.А. Воевода, **А.В. Марков**. – Сб. науч. тр. НГТУ. – 2012. – №3(69). – С. 115–122.

16. **Марков, А.В.** Понятие рекурсии в сетях Петри: факториал числа, числа Фибоначчи [Текст] / **А.В. Марков**, А.А. Воевода. – Сб. науч. тр. НГТУ. – 2013. – №1(71). – С. 72–77.

17. **Марков, А.В.** Анализ сетей Петри при помощи деревьев достижимости [Текст] / **А.В. Марков**, А.А. Воевода. – Сб. науч. тр. НГТУ. – 2013. – №1(71). – С. 78–95.

18. **Марков, А.В.** Анализ отдельных частей дерева достижимости сетей Петри [Текст] / **А.В. Марков**. – Сб. науч. тр. НГТУ. – 2013. – № 3(73). – С. 58–74.

19. **Марков, А.В.** Инверсия сетей Петри [Текст] / **А.В. Марков**. – Сб. науч. тр. НГТУ. – 2013. – № 4(74). – С. 97–121.

20. Шоба, Е.В. Методология проектирования современного программного обеспечения применительно к станции управления лифтом [Текст] / Е.В. Шоба, **А.В. Марков** // Сб. науч. тр. НГТУ. – 2012. – №1(67). – С. 121–132.

21. **Марков, А.В.** Развитие системы “Перемещение манипулятора в пространстве с препятствиями” при помощи рекурсивных функций [Текст] / **А.В. Марков**, А.А. Воевода. – Автоматика и программная инженерия. – 2013. – №2(4). – С. 35–41.

22. **Марков, А.В.** Описание работы двухсимочных мобильных телефонов с помощью сетей Петри. Камбиев метод [Текст] / **А.В. Марков**, Д.В. Прытков. – Сб. науч. тр. НГТУ. – 2011. – №3(65). – С. 113–118.

23. **Марков, А.В.** Разработка программного обеспечения при совместном использовании UML-диаграмм и сетей Петри (обзор) [Текст] / **А.В. Марков**. – Сб. науч. тр. НГТУ. – 2013. – №1(71). – С. 96–131.

24. Воевода А.А. Совместное использование UML-диаграмм и сетей Петри на этапе проектирования программного обеспечения: обзор. [Текст] /А.А. Воевода, С.В. Коротиков, **А.В. Марков** Сб. науч. тр. НГТУ. – 2012. – № 2(66) С. 75–98.

Отпечатано в типографии Новосибирского
государственного технического университета
630073, г. Новосибирск, пр. К. Маркса, 20, тел./факс: (383) 346-08-57
формат 60х84 1/16, объем 1,5 п.л., тираж 100 экз.
заказ № 724 подписано в печать 17.04.2015 г.