Федеральное государственное бюджетное учреждение науки
Институт вычислительных технологий Сибирского отделения Российской академии наук

На правах рукописи

Киселев Илья Николаевич

МОДУЛЬНОЕ МОДЕЛИРОВАНИЕ БИОЛОГИЧЕСКИХ СИСТЕМ НА ПРИМЕРЕ СЕРДЕЧНО-СОСУДИСТОЙ СИСТЕМЫ ЧЕЛОВЕКА

03.01.09 – Математическая биология, биоинформатика

Диссертации на соискание ученой степени
кандидата физико-математических наук

Научный руководитель:
к.б.н. Ф.А. Колпаков

Новосибирск – 2016

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность работы. Математическое моделирование сложных биологических систем, таких как сердечно-сосудистая система (ССС) человека, играет важную роль для понимания их внутреннего строения и функционирования. Адекватные математические модели позволяют проводить эксперименты *in silico*, тестировать новые лекарства на моделях, предсказывать поведение в различных условиях [Ventura et al., 2006, Karr et al., 2012, Milligan et al., 2013]. Одной из важнейших проблем является исследование механизма регуляции артериального давления, который включает в себя взаимодействие различных органов и тканей человека. На данный момент существует большое количество математических моделей, описывающих ССС человека целиком или ее части на различном уровне абстракции и использующих различный математический аппарат. Существуют глобальные модели, охватывающие практически всю систему, например [Guyton et al., 1972; Ikeda et al., 1979; Hester et al., 2011], однако им часто не хватает подробного описания отдельных подсистем. Кроме того, из-за сложной структуры и большого объема (до нескольких тысяч переменных), с такими моделями трудно работать и каким-либо образом расширять их.

Модульное моделирование – известный и активно используемый в системной биологии в последние несколько лет подход [Snoep et al., 2006; Fenner et al., 2008; Hernandez et al., 2009; Cooling et al., 2010; Hucka et al., 2013]. Он опирается на особенности структуры биологических систем, позволяющие разложить их на подсистемы в соответствии с их функциональной ролью и иерархической организацией. Каждая из таких подсистем может быть смоделирована отдельно. Модель всей системы может быть сконструирована как комбинация более простых математических моделей (модулей). Каждый модуль может быть создан, отлажен отдельно, используя свой собственный формализм, масштаб времени и детализацию. Такой подход делает более явной внутреннюю структуру системы, описывая ее как взаимодействие модулей, и является перспективным для

создания сложных моделей, описывающих биологические системы на разных уровнях организации и сложности. Кроме того, данный подход позволяет переиспользовать существующие модели биологических систем, и создавать на их основе более сложные модели, вплоть до наиболее полных моделей целых организмов, таких как полная модель физиологии человека [Hunter et al., 2002; Fenner et al., 2008].

Агентное моделирование используется для симуляции работы систем состоящих из автономных, взаимодействующих агентов [Woolridge, 2002]. В рамках агентного моделирования, формализуются правила, по которым функционируют агенты, из взаимодействия которых складывается поведение всей системы [Inchiosa and Parker, 2002]. Этот метод в основном используется для моделирования систем, поведение которых на глобальном уровне слишком сложно или невозможно формализовать: моделирование социальных процессов [Bonabeau, 2002], фондовых рынков [Chan et al., 1999], предсказание распространения инфекционных заболеваний [Perez and Dragicevic, 2006] и т.д. Метод также хорошо подходит для изучения сложных биологических систем. Автономность агентов делает возможной работу с подмоделями, такими как модели различных органов или клеток, гибкий механизм взаимодействия между агентами позволяет объединять эти подмодели в комплексные модели.

Для эффективного переиспользовании моделей, важно поддерживать общепринятые стандарты описания моделей. Для биологических моделей таким стандартом де-факто является язык разметки SBML – Systems Biology Markup Language [Hucka et al., 2003]. Для графического отображения моделей наиболее широко использующимся стандартом является SBGN – Systems Biology Graphic Notation [Le Novere et al., 2009].

Таким образом, актуальной задачей является разработка подходов и инструментария для упрощения работы со сложными моделями путем разделения их на связанные подмодели и создания сложных моделей путем объединения различных моделей, разработанных различными авторами с использованием различных математических формализмов.

Цель работы – разработка программного обеспечения для создания модульных моделей биологических систем, их графического представления, а также численных алгоритмов для расчетов как в случае модулей с одинаковым математическим формализмом, так и в общем случае – на основе принципов агентного моделирования, тестирование разработанного ПО и апробирование на примере создания модульной модели ССС человека.

Для достижения поставленной цели были поставлены и решены следующие задачи:

1. Разработать способ формального описания модульных моделей биологических систем. Разработать нотацию для графического представления таких моделей в виде блочных диаграмм.
2. Для случая, когда все модули представлены системами обыкновенных дифференциальных уравнений (ОДУ) с мгновенными событиями, разработать алгоритм трансформации модульной модели в “плоскую” модель, представляющую из себя одну систему ОДУ с мгновенными событиями. Реализовать поддержку формата SBML для модульных моделей подобного типа.
3. Разработать алгоритм численных расчетов на основе принципов агентного моделирования для случая модулей, использующих различные методы численного моделирования.
4. Реализовать разработанные алгоритмы в виде программного модуля для платформы формального и визуального описания биологических систем BioUML.
5. Протестировать разработанное ПО путем реализации в виде модульных диаграмм моделей различных частей ССС: артериального дерева, почки и сердца. Создать на их основе комплексную модульную модель ССС человека. Валидировать созданную модель.

Результаты, выносимые на защиту.

1. Подход к моделированию сложных систем, включающий формальное описание и визуальное представление моделей в виде модульных

диаграмм, алгоритм генерации ОДУ-модели с мгновенными событиями на основе модульной диаграммы в случае, когда все модули содержат ОДУ-модели с мгновенными событиями, и алгоритм численных расчетов на основе принципов агентного моделирования в случае различных математических формализмов модулей.

2. Разработанный на языке Java программный модуль для платформы BioUML для графического представления, создания, редактирования и проведения численных расчетов модульных моделей сложных систем.
3. Комплексная модульная модель ССС человека, способная демонстрировать как долговременные (изменение концентрации различных гормонов в организме), так и кратковременные (биение сердца, ток крои по отдельным сосудам) процессы в системе.

Научная новизна.

1. Разработаны новые подходы к формальному и визуальному описанию биологических систем.
2. Разработан новый алгоритм генерации диаграммы, описывающей ОДУ-модели с мгновенными событиями на основе модульной диаграммы.
3. Впервые создана расширяемая модульная модель ССС человека, включающая подмодели артериального дерева из 55 крупнейших сосудов, сердца и почки, использующие различные математические формализмы и масштабы.
4. Для модульной модели, включающая артериальное дерево проведена процедура персональной настройки параметров и валидация на физиологических данных для 946 человек (Мельников и др. 2012).

Практическая значимость. Созданное в процессе работы программное обеспечение может использоваться для моделирования биологических систем широкого спектра. В данный момент, программное обеспечение **внедрено** и используется в Институте общей генетики им. Н. И. Вавилова, РАН, Москва, Институте математики им. С. Л. Соболева СО РАН, Новосибирск, что подтверждено актами о внедрении и Genexplan GMBH, Wolfenbuttel, Germany.

Всё описанное программное обеспечение находится в свободном доступе, как часть платформы BioUML, по адресу: http://www.biouml.org/.

Созданное программное обеспечение было успешно использовано для создания комплексной модульной модели апоптоза [Kutumova et al., 2012; Кутумова, 2012], объединяющей 13 различных моделей, взятых из литературы и и переработанных в 9 модулей.

Созданная комплексная модель ССС может быть использована для проведения экспериментов *in silico* и для изучения взаимодействия описанных в ней подсистем. Благодаря своей модульной структуре, модель может быть расширена путем добавления новых модулей, описывающих различные органы человека, такие как легкие, печень, поджелудочная железа и т.д. Таким образом, созданная модель может служить основой для создания наиболее полной модели физиологии человека.

Методология исследования опирается на современные информационно-вычислительные технологии, предусматривающие использование адекватных математических моделей изучаемого явления и эффективных вычислительных алгоритмов. Для организации численного моделирования используется аппарат, основанный на теории агентного моделирования, для непосредственных численных расчетов используются алгоритмы на основе методов Эйлера, Адамса и Гира.

Достоверность результатов моделирования подтверждается успешным тестированием алгоритмического и программного инструментария на модельных задачах, а также удовлетворительными результатами сопоставления с результатами расчетов по различным математическим моделям и численным алгоритмам, в том числе других авторов, с аналитическими и экспериментальными данными.

Апробация работы. Основные положения и результаты диссертации были представлены на следующих научных мероприятиях: IX, XI и XII международных конференциях по системной биологии ICSB (Гетеборг, Швеция, 2008; Эдинбург, Шотландия, Великобритания, 2010; Хайдельберг, Германия, 2011); VIII

международной конференции по биоинформатике регуляции и структуры геномов и системной биологии BGRS (Новосибирск, 2012); международной конференции по современным проблемам математики, информатики и биоинформатики, посвященной 100-летию со дня рождения член-корреспондента АН СССР Алексея Андреевича Ляпунова (Новосибирск, 2011); школе молодых ученых «Биоинформатика и Системная биология» (Новосибирск, 2012); семинаре «Информационные и вычислительные технологии в медицине» (Новосибирск, 2012, 2013); международный семинаре «From virtual cell to virtual human and virtual patient» (Новосибирск, 2012); международной конференции по биомедицинской инженерии и компьютерных технологиях SIBIRCON (Новосибирск, 2015). В полном объеме работа докладывалась на объединенном семинаре Института вычислительных технологий СО РАН, Конструкторско-технологического института вычислительной техники СО РАН и Новосибирского государственного университета (Руководители семинара: академик РАН Ю. И. Шокин, чл. кор. РАН А. М. Федотов, д.ф.-м.н. С. К. Голушко; Новосибирск, 2012, 2013).

Основные результаты диссертации **опубликованы** в 19 печатных работах, в том числе 3 статьи в рецензируемых научных журналах, рекомендованных ВАК [Киселев и др, 2012; Kutumova et al., 2012; Киселев и др. 2015], одна публикация в международном рецензируемом журнале [Kiselev and Kolapkov, 2013], 10 публикаций в сборниках тезисов международных и всероссийских конференций, две главы в монографиях [Kiselev et al., 2012a; Kiselev et al., 2012b] и учебное пособие для НГУ [Бибердорф и др. 2015].

Личный вклад автора. Результаты, составляющие основное содержание диссертации, получены автором самостоятельно. В совместных работах автор принимал непосредственное участие в создании модульной модели ССС человека и самостоятельно разработал алгоритм численных расчетов на основе принципов агентного моделирования. В работе [Kutumova et al., 2012] автор самостоятельно разработал алгоритма автоматической генерации плоской модели на основе модульной модели. В работах [Kiselev and Kolpakov, 2013; Kolpakov et al., 2009;

Kiselev et al., 201; Kiselev et al., 2012] автор принимал непосредственное участие в разработке программного обеспечения для визуального создания модульных моделей, а также самостоятельно разработал алгоритмы генерации агентных моделей и вычислений. В работе [Киселев и др. 2015] автор принимал непосредственное участие в разработке процедуры персонализации параметров модели и самостоятельно реализовал программу на языке Java для проведения численных расчетов. В работах [Baranov et al., 2016; Kiselev et al., 2015; Kiselev et al., 2016] автор принимал непосредственное участие в разработке процедуры персональной настройки параметров модели и самостоятельно реализовал программу на языке Java для настройки параметров и валидации модели.

Структура и объем диссертации. Работа состоит из введения, четырех глав, заключения, списка сокращений, списка цитируемой литературы из 149 наименования и 5 приложений. Полный объем диссертации составляет 177 страниц, включая 138 страниц текста, 60 рисунков (из них 14 в приложении) и 22 таблицы (из них 3 в приложении).

Автор выражает глубокую благодарность и признательность своему научному руководителю к.б.н. Ф.А. Колпакову за постановку задачи и постоянное внимание в ходе выполнения работы, а также благодарит д.ф.-м.н. С. К. Голушко, к.ф.-м.н. Б. В. Семисалова, к.ф.-м.н. Э. А. Бибердорф и Р. Н. Шарипова за ценные критические замечания.

ГЛАВА 1. ОБЗОР

1.1. Модульный подход к моделированию биологических систем

Модульный подход широко применяется при разработке и моделировании современной техники и в последнее десятилетие его важность при моделировании биологических систем также существенно возросла [Bassingthwaighte et al., 2010; Cooling et al., 2010; Fenner et al., 2008; Thomas et al., 2007; Hucka et al., 2013]. Одна из причин этого заключается в том, что модульная структура модели хорошо соответствует структуре биологических систем на различных уровнях: от клеток [Hartwell et al., 1999] до отдельных органов и организмов в целом [Ginkel et al., 2003]. В [Snoep at al., 2006] описывается концепция создания всеобъемлющей модели, описывающей клеточную систему на уровне химических реакций (т.н. виртуальная клетка). Подобная модель может быть создана лишь на базе большого количества моделей, разработанных сообществом исследователей и хранящихся в едином хранилище.

Большая исследовательская работа посвящена созданию наиболее полной модели человеческой физиологии – Виртуального физиологического человека (Virtual Physiological Human). Она поддерживается Международным Объединением Физиологических Наук (International Union of Physiological Sciences Physiome Project) [Hunter et al., 2002] и инициативой EuroPhysiome [Fenner et al., 2008], координирующих усилия многих исследовательских групп по всему миру. Кроме того, существуют подобные проекты во Франции – SAPHIR (Systems Approach for PHysiological Integration of Renal) [Thomas et al., 2007], Америке [Bassinghtwaighte et al.,1999] и Японии [Taishin et al., 2010]. Одной из важнейших проблем при реализации этих проектов является интеграция различных моделей [Fenner et al., 2008].

Достижение этих актуальных целей (создание виртуальной клетки и модели физиологии человека) требует:

1) соблюдения общепринятых стандартов описания моделей для передачи моделей от одной группы исследователей другой и переиспользованию моделей;
2) развития подходов к объединению моделей, созданных различными группами исследователей, в том числе, с использованием различных математических формализмов;
3) развития специализированных программных продуктов для создания, анализа и проведения численных расчетов модульных моделей;
4) использования визуального подхода к работе с модульными моделями.

Существует два общих подхода к численным расчетам модульных моделей [Hernandez et al., 2009; Vangheluwe, 2000]:

1. **Трансформация формализмов (Formalism-transformation)**. Модульная модель трансформируется в обычную модель, численные расчеты для которой могут быть проведены одним из стандартных методов (ОДУ, стохастическая модель и т.д.) Этот процесс требует формального описания процесса трансформации различных формализмов друг в друга, что является не тривиальной задачей и может сильно ограничить типы возможных подмоделей. Более простым частным случаем является вариант, когда все модули используют один и тот же формализм или даже описаны одним и тем же формальным языком (например, SBML).
2. **Ко-симуляция (Co-simulation)**. Данный подход предполагает, что численные расчеты для каждой из подмоделей проводятся отдельно, используя свой собственный численный решатель. Этот процесс управляется мета-решателем, который обеспечивает синхронизацию подмоделей и обмен информацией между ними во время численных расчетов.

Оба подхода имеют преимущества и недостатки. В первом случае, имеются строгое математическое обоснование существования и единственности решения задачи, а также гарантированная точность используемых методов, но есть ограничение на типы используемых модулей. Во втором случае, в модульную

модель могут быть объединены произвольные подмодели, однако математическое обоснование корректности полученного решения может быть недостаточным. Возможно также сочетание обоих подходов.

Ключевым вопросом в обоих случаях является то, каким образом взаимодействуют между собой подмодели в рамках модульной модели. Обычно существуют дополнительные элементы (связи), указывающие на характер взаимодействия подмоделей. В случае трансформации формализмов эти связи указывают на то, как элементы подмоделей должны быть изменены (удалены, заменены другими элементами) во время трансформации. В случае ко-симуляции связи указывают на характер обмена информацией между подмоделями.

Другим важным вопросом является то, каким образом модульная модель может взаимодействовать с элементами подмоделей. Элементы подмоделей (уравнения, химические реакции, переменные и т.п.) могут быть явно доступны для других частей модели. Другой вариант – по умолчанию все элементы недоступны извне модуля. Для того чтобы элемент сдал доступен, модуль должен определить дополнительный элемент: порт. Совокупность таких портов определяет интерфейс модуля. Первый подход более гибкий, в то время как второй позволяет создавать более контролируемые и хорошо определенные модульные модели.

Если модели подсистем изначально разработаны с целью объединения в модульную модель, они могут определять свои интерфейсы для взаимодействия с другими подмоделями в рамках модульной модели. В этом случае, создание модульной модели сводится к выбору нужных подмоделей и установке нужных связей между их интерфейсами.

Различные подходы к созданию модульных моделей на основе SBML обсуждаются в [Randhawa, 2008; Randhawa et al., 2009; Randhawa et al., 2010]:

1. **Слияние (Fusion)** – это ручное создание модели, включающей в себя элементы используемых подмоделей.

2. **Композиция (Composition)** означает, что комплексная модель включает в себя подмодели в явной форме вместе со всем своим содержимым. Связи могут устанавливаться между их внутренними элементами.
3. **Аггрегация (Aggregation)** отличается от композиции тем, что подмодели заранее определяют интерфейсные элементы, которые могут быть использованы для связи с другими подмоделями. Все остальные элементы подмоделей скрыты от пользователя. Этот подход подразумевает, что подмодели изначально созданы с целью объединения в модульную модель.
4. **Генерация “плоской” модели (Model flattening)** это процесс автоматического создания “плоской” модели на основе модульной модели, созданной с помощью композиции или агрегации. Это частный случай трансформации формализмов. Под “плоской” моделью подразумевается математическая модель, не содержащая подмоделей.

Наиболее широко используемыми стандартами для описания математических моделей биологических систем являются языки разметки SBML [Hucka et al., 2003] и CellML [Cuellar et al., 2006].

Обсуждение включения синтаксиса для описания модульных моделей в язык SBML продолжается с момента создания языка в 2000 году, за это время было предложено несколько вариантов от различных авторов. Официальная спецификация для расширения языка SBML (SBML “comp”) была опубликована в 2013 году [Smith et al., 2013]. Одновременно был опубликован набор тестов для этого расширения. Расширение позволяет SBML моделям ссылаться внутри себя на другие SBML модели. Модель, на которую делается ссылка, может быть определена как в том же файле, что и основная модель, так и во внешнем источнике. В этом случае ссылка делается посредством URI. Интеграция подмоделей обеспечивается элементами замены, с помощью которых любой элемент основной модели может быть заменен на элемент такого же типа из подмоделей или наоборот, элемент основной модели может заменять собой элемент подмодели. Два элемента из различных подмоделей могут быть

отождествлены путем создания дополнительного элемента в основной модели, который заменяет указанные элементы подмоделей. Элементы подмоделей могут быть адресованы напрямую или через заданные заранее интерфейсные элементы - порты.

CellML был изначально разработан для описания модульных моделей, однако, только версия 1.1 [Cuellar et al., 2006] поддерживает импорт и переиспользование внешних моделей в качестве модулей. В отличие от SBML, в CellML модули могут быть связаны только через свои переменные. Модули должны определять интерфейсные порты. Каждый порт соответствует переменной математической модели, заключенной в модуле. Порты могут быть двух типов – входные и выходные. Связь между двумя портами (выходным портом одного модуля и входным портом другого) означает, что переменная, соответствующая входному порту, будет заменена на переменную, соответствующую выходному порту.

На данный момент существует большое количество программных продуктов для создания и работы с моделями биологических систем. Список продуктов, поддерживающих язык SBML, доступен по адресу www.sbml.org и включает в себя более чем 250 программных продуктов, однако большая их часть не поддерживает модульный подход. Причина этого в том, что SBML сравнительно недавно был расширен синтаксисом для описания модульных моделей. Ниже приводится краткий обзор программных продуктов для системной биологии, поддерживающих концепцию модульности. В конце параграфа приведены сводные таблицы для сравнения функциональности описанных продуктов.

Одним из первых инструментов для моделирования, поддерживающих модульный подход, является ProMoT [Mirschel et al., 2009]. Он представляет объектно-ориентированный язык, способный описывать модульные модели, использующие ОДУ-формализм. На данный момент он поддерживает версию SBML l2v1 за исключением дискретных событий.

JigCell [Vass et al., 2004] – это набор инструментов, поддерживающих SBML l2v1 и позволяющих объединять SBML модели в иерархические модели. Иерархическая модель описывается в рамках специально разработанного формализма [Randhawa et al., 2010]. Модули редактируются как таблицы элементов. JigCell также позволяет проводить численные расчеты и анализ моделей. Последняя версия датируется 2009 годом, в данный момент не находится в активной разработке.

iBioSim [Myers et al., 2009] позволяет визуально редактировать SBML модели, включая модульные.

Существует также некоторое количество инструментов для проведения численных расчетов, поддерживающих модульные SBML модели. Примерами могут служить RoadRunner и Systems Biology Simulation Core Library [Keller et al., 2013].

TinkerCell [Chandran et al., 2009], подобно iBioSim, поддерживает как модульность моделей, так и визуальное представление. Благодаря собственной модульной структуре, TinkerCell позволяет подключать дополнительные пакеты для численных расчетов и анализа. Поддерживает только экспорт в SBML формат. Последняя версия датируется 2011 годом и к сожалению на данный момент больше не находится в активной разработке.

M2SL [Hernandez et al., 2009] поддерживает модели с множественным формализмом используя ко-симуляционный подход. На данный момент он не поддерживает визуальное моделирование и, кроме того, не поддерживает стандарты SBML и CellML. Последняя версия датируется 2011 годом.

Отдельно нужно отметить наиболее популярный инструмент для работы с SBML-модели – COPASI [Hoops et al., 2009] и популярный инструмент визуализации SBML-моделей в виде SBGN-диаграмм VANTED [Rohn et al., 2012], которые, однако, не поддерживают модульность, поэтому не включены в наше сравнение.

Количество инструментов, поддерживающих CellML (обзор приводится, например в [Garny, 2008], список доступен по адресу www.cellml.org) значительно

меньше, чем для SBML. Более того, хотя язык по своей природе поддерживает модульность, многие инструменты поддерживают только импорт модульных CellML-моделей с автоматической трансформацией в плоскую модель. (JSim [Raymond et al., 2003], VirtualCell [Moraru et al., 2008]). Инструменты, которые поддерживают модульное описание моделей (OpenCell, COR [Garny et al., 2009]) не предлагают графического представления моделей. Среди инструментов, представленных на официальном сайте CellML, только два используют визуальное представление моделей:

GUICellML предоставляет визуальное представление модульной модели в виде связанных между собой подмоделей. Внутреннее содержание подмоделей представлено в виде CellML-текста. CellModelViewer позволяет только просматривать содержимое моделей, но не создавать или редактировать их. Также, в данный момент в разработке находится программный продукт CellML Viewer [Wimalaratne et al., 2009].

Языки SBML и CellML не являются “человекочитаемыми”, т. е. не предназначены для чтения людьми. Их назначение – хранение и передача моделей. Особенно это касается модульных моделей с большим количеством подмоделей. По этой причине, они нуждаются в программных продуктах, позволяющих редактировать записанные с помощью этих языков модели с помощью пользовательского интерфейса. Существуют также разработанные “человекочитаемые” языки для описания математических моделей биологических систем. Например, Antimony [Smith et al., 2009] (поддерживает модульные SBML- и CellML-модели), PySB [Lopez et al., 2013] использует язык программирования Python для описания моделей, little-B [Mallavarau et al., 2009] основан на языке программирования Lisp.

Наконец нужно отметить, что графическая нотация SBGN поддерживает описание модульных моделей [Le Novere et al., 2009].

Таблица 1.1 – Программные продукты, поддерживающие модульное моделирование биологических систем. Зачеркнуты более не поддерживаемые. Базовый функционал

Название	**Визуальное моделирование**	**Визуальное модульное моделирование**	**Численные расчеты**	**Анализ**	**Оптимизация параметров**
iBioSim	+	+	+	+	+
ProMoT	+	+	+	+	+
RoadRunner	-	-	+	+	+
M2SL	-	-	+	+	+
SB Simulation Core Library	-	-	+	-	-
QTAntimony	-	-	-	-	-
COR	-	-	+	-	-
Little-B	-	-	-	-	-
~~JigCell~~	-	+	+	+	+
~~TinkerCell~~	+	+	+	+	+
~~OpenCell~~	-	-	+	-	-

Таблица 1.2 – Программные продукты, поддерживающие модульное моделирование биологических систем. Зачеркнуты более не поддерживаемые. Поддерживаемые стандарты и формализмы

Название	**Версия SBML**	**SBML comp**	**CellML**	**Поддерживаемые формализмы**	**Множественный формализм**
iBioSim	l3v1	+	-	ОДУ, алгебраические уравнение, события, стохастические модели, агентные модели.	-
ProMoT	l2v1, кроме событий	-	-	ОДУ, алгебраические уравнения, логические модели.	-
RoadRunner	l3v1	+	-	ОДУ, алгебраические уравнения, события.	-
M2SL	-	-	-	ОДУ, алгебраические уравнения, события, клеточные автоматы.	+
SB Simulation Core Library	l3v1	+	-	ОДУ, алгебраические уравнения, события.	-
QTAntimony	l3v1	+	+	ОДУ, события.	-
COR	-	-	+	ОДУ, алгебраические уравнения.	-
Little-B	-	-	-	ОДУ.	-
~~JigCell~~	l2v4	-	-	ОДУ, алгебраические уравнения, события, стохастические модели.	-
~~TinkerCell~~	Экспорт	-	-	ОДУ, алгебраические уравнения, события, стохастические модели.	-
~~OpenCell~~	-	-	+	ОДУ, алгебраические уравнения.	-

1.2. Агентное моделирование

Агентное моделирование – относительно новый раздел имитационного моделирования, которое применяется когда формализовать поведение системы как единого целого трудно или невозможно. Оно подразумевает представление системы в виде набора взаимодействующих сущностей способных самостоятельно принимать решения - агентов. Общее поведение системы складывается из взаимодействия агентов. Агентное моделирование связано с модульным в том смысле, что оно тоже моделирует систему как набор связанных подсистем.

Не существует общепринятого определения, что такое агент. Из практических соображений, принимают, что агент должен обладать следующими свойствами [Macal and North, 2009]:

- агент должен быть **автономным**, действовать независимо от других агентов за исключением ряда определенных взаимодействий;
- агент должен быть определен как **отдельная** сущность со своим набором характеристик;
- агент должен **взаимодействовать** с другими агентами.
- часто агенты рассматривается как существующие в некоторой **внешней среде**.

Существуют различные классификации агентов, в частности агенты бывают [Woolridge, 2002; Odell, 2001]:

- **реактивные**, т.е. по определенным правилам реагирующие на внешние сигналы. Такой агент принимает сигнал от внешней среды или других агентов, и генерирует определенную реакцию на этот сигнал.
- **адаптивными (агенты с состоянием)**, т.е. изменяющие свое внутреннее состояние и, следовательно, изменяющие свою реакцию на сигналы. Такие агенты хранят внутреннее состояние и под воздействием внешнего сигнала меняют его. Действие агента зависит от того как изменилось внутреннее состояние.

- **проактивными**, т.е. имеющими некоторую глобальную цель и действующими так чтобы достичь этой цели. Это свойство может быть реализовано путем задания функции оценки текущего состояния и корректировки действий агента с целью улучшения текущего состояния.
- обучающиеся, эволюционирующие, рассуждающие и др.

Различают собственно **агентные модели** (agent-based models), которое обычно подразумевает большое (сотни и тысячи) количество относительно простых агентов, поведение которых может создавать поведение системы, и **мультиагентные системы** (multi-agent systems), которые предполагают небольшое количество взаимодействующих (конкурирующих) интеллектуальных агентов. У обоих подходов общим принципом является описание системы как набора взаимодействующих сущностей.

Исторически первым инструментом для создания агентных моделей является Swarm, созданный в 1994 г. в институте Санта Фе [Swarm wiki].

Repast [North et al., 2013] и NetLogo [Wilensky and Rand, 2015] предоставляют графические интерфейсы а также специализированные языки программирования (ReLogo и NetLogo соответственно) для создания агентных моделей. Оба программных продукта бесплатны с открытым исходным кодом. Repast также позволяет создавать модели с помощью кода на Java и предоставляет версию для высокоэффективных вычислений Repast for High Performance Computing

Библиотеки языка Java с открытым исходным кодом MASON [Luke et al., 2005] (специализируется на мультиагентных системах) и Ascape [Parker, 2001] (для агентных моделях). Обе библиотеки предоставляют как базовые инструменты для создания и численных расчетов агентных моделей, так и средства визуализации.

Jade – библиотека языка Java, разработанная в первую очередь для распределенных на несколько устройств агентных моделей [Bellifemine et al., 2003].

Anylogic – написанный на языке Java отечественный проприетарный программный комплекс, предоставляющий возможности для создания дискретно-непрерывных и имитационных моделей, в основном в областях управления бизнес-процессами, производством, финансами и социальной динамикой [Карпов, 2009]. Anylogic предоставляет графический интерфейс для создания моделей, при этом поведение агентов задается пользователем путем написания кода на языке Java.

В работе [Bajracharya and Duboz, 2013] одна и та же модель распространения заразных заболевания была реализована в трех разных программных продуктах, включая NetLogo и Repast. Показано, что результаты численных расчетов значительно отличаются. Авторы предполагают, что причина может быть в том, что пользователю не всегда предоставлен полный контроль над тем, как проходят численные расчеты – различные программные продукты используют различные стратегии управления агентами, не всегда эти стратегии в достаточной степени документированы. Таким образом, необходимо с осторожностью относиться к результатам расчетов выполненных с помощью агентного моделирования.

Все вышеперечисленные программные продукты не являются специализированными, в какой либо области. В частности, ни один из них не поддерживает формат SBML для биологических моделей. EPISIM [Sütterlin et al., 2012] заполняет этот пробел, используя COPASI для численных расчетов отдельных агентов и MASON для агентного моделирования.

В данный момент в разработке находится расширение “dyn” для языка SBML. Это расширение, в первую очередь, нацелено на создание моделей клеточной популяции и позволит создавать правила, по которым в процессе численных расчетов должны динамически удаляться и создаваться новые элементы модели. Кроме того оно добавит пространственные координаты и возможность перемещаться объектам [Stevens and Myers, 2013].

Общепринятого стандарта описания биологических систем, который позволил бы объединять модели с различным математическим формализмом, насколько известно автору, не существует.

1.3. Модели сердечно-сосудистой системы человека

Модель регуляции кровообращения, предложенная профессором Гайтоном и соавторами в 1972 году [Guyton et al., 1972], является наиболее известной глобальной моделью, основанной на клинических и экспериментальных данных. Модель демонстрирует долговременные эффекты регуляции ССС человека, где значительную роль играет почка. При этом модель оперирует усредненными по времени величинами. Краткосрочные эффекты, такие, как биение сердца, в модели не учитываются. Модель была изначально написана на языке программирования FORTRAN, позже была переписана на язык С и долгое время дорабатывалась и расширялась ее авторами. Исходная модель снабжена структурной схемой, представляющей ее в виде блочной структуры, состоящей из более чем 350 элементарных блоков. Блоки соответствуют числовым константам, переменным и операциям, таким как сложение, умножение, интегрирование.

Модель общей регуляции телесных жидкостей, опубликованная в 1979 году Икедой с соавторами [Ikeda et al., 1979], является расширением модели Гайтона, делая при этом акцент на регуляцию кислотно-щелочного баланса и большую роль почки.

Несмотря на новые исследования и открытия в области функционирования ССС человека, модель Гайтона до сих пор вызывает интерес исследователей [Nguyen et al., 2008; Osborn et al., 2009; Thomas et al., 2008]. В 2010 году модель была воссоздана в среде MATLAB SIMULINK [Kofranek and Rusz, 2010], одновременно был исправлен ряд ошибок в исходной схеме.

Дальнейшее развитие модели Гайтона можно условно разделить на три направления.

1. Создание на ее базе еще более глобальных моделей путем включения дополнительных переменных и уравнений. Так были построены глобальная

модель QCP (Quantitative Circulatory Physiology) [Abram et al., 2007], содержащая более 4000 переменных, и ее более новая версия – Hummod [Hester et al., 2011], содержащая более 5000 переменных и представляющая модели в XML-формате, что облегчает расширение модели. Использование и анализ таких моделей существенно осложнены их большим размером и отсутствием полного формального описания.

2. Более детальная проработка отдельных подсистем, когда части модели, не относящиеся непосредственно к рассматриваемым системам, упрощаются или отбрасываются. Так, модель Гайтона содержит упрощенное по современным представлениям описание почки [Guyton, 1990], которая является одним из главных регуляторов артериального давления [Cowley et al., 1997; Guyton, 1990]. Модель почки, предложенная Карааслaном [Karaaslan et al., 2005], во многом основывается на модели Гайтона, дополняя ее более детальными описаниями гормональных эффектов из модели [Uttamsingh et al., 1985] и почечных канальцев из модели [Coleman and Hall, 1992].

3. Объединение с моделями краткосрочных процессов. Модель Гайтона не содержит описания кратковременных процессов в системе, зависящих от колебаний между систолой и диастолой. Моделирование таких процессов необходимо для более точного моделирования различных патологий сердца и кровообращения, таких как сердечная недостаточность [Le Rolle et al., 2011] или аритмия также, включение таких процессов в модель позволяет рассчитывать значения клинически измеряемых параметров, таких как максимальное и минимальное артериальное давление. Первая попытка такого объединения описана в работе [Werner et al., 2002]. Кроме того, модель Гайтона, наряду с моделью Икеды, используется в проекте SAPHIR в качестве основы для создания глобальной модели физиологии человека [Thomas et al., 2008], описывающей как краткосрочные, так и долгосрочные процессы в организме. В [Le Rolle et al., 2010; Le Rolle et al., 2011] описывается объединение модели Гайтона с моделью пульсирующего сердца с помощью программного комплекса M2SL [Hernandez et al., 2009] в рамках проекта SAPHIR.

Среди моделей кровообращения также стоит упомянуть модели, разработанные под руководством профессора Н.М. Амосова [Амосов и др., 1969] и его учеников [Лищук, 1991]. Эти модели изначально разрабатывались с целью клинического применения.

1.3.1. Модели пульсирующего сердца.

На данный момент существует довольно большое количество моделей пульсирующего сердца, в том числе, описывающие сердце в виде трехмерного объекта [Campen et al., 1994]. Более простой подход описывает сердце как резервуар, переключающийся между двумя состояниями: систола, при которой сердце сокращается и выбрасывает кровь в артериальную систему, и диастола, при которой сердце забирает кровь из венозной системы. Данный подход подробно рассмотрен в [Солодянников, 1994], где описан класс моделей ССС основанный на работах академика РАН В.И. Шумакова [Шумаков и др, 1969; Шумаков и др, 1971; Шумаков и др, 1975]. Базовая модель, описанная, предложенная профессором Солодянниковым, содержит описание тока крови по трем резервуарам: артериальной системе, венозной системе и левому желудочку. Параметры резервуаров являются усредненными по объему (не рассматривается ток крови по отдельным сосудам). Малый круг кровообращения в базовой модели отсутствует. Дальнейшим развитием модели является модель, включающая малый круг кровообращения, созданная А. П. Прошиным и Ю. В. Солодянниковым [Прошин, Солодянников, 2006]. На основе этой модели её авторами была разработана компьютерная система, предназначена в первую очередь для исследования влияния физических нагрузок, но она также может быть использована для моделирования широкого круга патологий сердечно-сосудистой системы человека: сердечно-сосудистой недостаточности, инфаркта миокарда, гипоксии и, в том числе, артериальной гипертензии. Система доступна по адресу http://www.samara-dialog.ru/silverlight/HeartSL5.html.

1.3.2. Модели сосудистого русла.

В упомянутых выше моделях, артериальная система рассматривается в виде системы ОДУ с переменными, описывающими глобальные величины, такие, как общий объем крови, среднее артериальное давление и т.д. Такие модели часто называют **моделям с сосредоточенными параметрами** [Кошелев, 2010]. Детальное математическое моделирование тока крови по трехмерным сосудам [Педли, 1983; Ригирер, 1971; Remuzzi et al., 1992] требует слишком больших вычислительных затрат [Блохин, 2009], поэтому на практике, как правило, применяются различные приближения трехмерной модели. Одномерные модели гемодинамики получается усреднением сосудов по поперечному направлению, т. е. сосуды считаются одномерными. Дальнейшее приближение приводит к моделям с сосредоточенными параметрами. В качестве примеров моделей сосудистого русла с соредоточенными параметрами можно привести [Григорян и Лиссов, 2004] и модели, рассматриваемые в [Лищук, 1991]. Подробно вывод уравнений для одномерных моделей приводится, например, в [Quarteroni, Formaggia, 2003, Абакумов и др., 1997]. Вопросы комбинации моделей различной размерности рассматриваются, например, в [Астраханцева и др., 2005]. В работе [Lamponi, 2004] подробно описывается одномерная модель тока крови по 55 наиболее крупным сосудам человеческого организма (артериальному дереву).

Одномерные модели описываются системами УЧП, требующими задания граничных условий. Обычно граничные условия описывают приток крови из сердца, отток в капилляры и т.д. Для задания этих условий могут использоваться эмпирические функции (например, фиксированный профиль сердечного выброса на выходе и фиксированное давление на выходе [Колпаков и др., 2009]).

В МГУ им. Ломоносова группой профессора А.П. Фаворского разрабатывается модель, описывающая ток крови по сосудистой системе, представленной в виде графа [Кошелев и др., 2010], в частности, исследуется динамика глюкозы в крови [Борзов и др., 2015]. В этих работах используются и более сложные формулы, регулирующие переключение между систолой и диастолой в зависимости от потока крови из сосудистой системы в сердце. Эти

условия являются частью модели ССС и их изменение или задание более детального взаимодействия сосудистого русла с другими подсистемами (сердце, почка) затруднено.

Также актуальной задачей является использование более сложных моделей (системы ОДУ, трехмерные модели сердца и т.д.) для задания граничных условий. В [Семисалов, 2010] описана попытка объединения моделей вручную – проводятся расчеты для одной модели, затем полученные данные подставляются в другую и т.д. Такой подход весьма трудоемок и его возможное использование сильно ограничено. Полноценная же интеграция подобных моделей требует создания специализированного программного обеспечения [Le Rolle et al., 2005].

1.4. Выводы по главе 1.

Наиболее быстрым и рациональным путем создания моделей сложных биологических систем, таких как ССС, является переиспользование и объединение существующих моделей.

Для эффективного переиспользования моделей, их составных частей и воспроизведения результатов, полученных другими исследовательскими группами необходимо поддерживать существующие форматы описания и визуального представления моделей биологических систем.

Визуальное моделирование - создание и редактирование модели в виде диаграммы – позволяет существенно упростить процесс построения моделей.

Специализированных программных продуктов, которые бы поддерживали модульное моделирование биологических систем немного и еще меньше поддерживающих визуальное модульное моделирование и существующие стандарты SBML и SBGN. Отечественных программных продуктов, реализующих такой функционал, не существует.

Практически не существует специализированных программных продуктов для создания моделей биологических систем, поддерживающих объединение моделей с различными математическими формализмами.

ГЛАВА 2. ПЛАТФОРМА BIOUML

2.1. Основные особенности BioUML

BioUML (Biological Universal Modeling Language) [Колпаков, 2011] — это открытая (open-source), бесплатно распространяемая, написанная на языке программирования Java, платформа для моделирования биологических систем. Она применима для решения широкого круга задач, включающий доступ к различным базам данных, математическое описание и визуальное представление биологических систем, проведение численных расчетов, подбор параметров и различные типы анализа математических моделей и данных.

Основные возможности платформы BioUML:

- возможность работы как в локальной (standalone) версии программы, так и удаленно, через web-интерфейс;
- поддержка общепринятых стандартов описания биологических систем (SBML) и их графического представления (SBGN);
- визуальное моделирование биологических систем и процессов – пользователь графически создает и редактирует математическую модель;
- поддержка различных математических формализмов – ОДУ, алгебраические уравнения, мгновенные события, УЧП, стохастические модели;
- модульная (plugin-based) архитектура платформы, облегчающая добавление новых типов моделей, численных решателей, и т. д.

BioUML использует технологию динамических свойств (Dynamic Property) [Fowler, 1997], позволяющую добавлять и удалять свойства объектов во время исполнения. Для автоматического создания пользовательского интерфейса, позволяющего редактировать свойства объектов, BioUML использует технологию BeanExplorer [BeanExplorer].

В BioUML модель биологической системы описывается на трех взаимосвязанных уровнях [Колпаков, 2011]:

1) граф – структура модели описывается графом, узлы и ребра которого обозначают различные элементы системы и связи между ними;
2) базы данных – каждый элемент графа может содержать ссылку на какой-либо объект базы;
3) математическая модель – любой элемент графа может быть интерпретирован как элемент математической модели.

В этих терминах может быть полностью представлена информация из многих широко используемых баз данных по биологическим системам и моделям (BioModels, BioPAX, KEGG, TRANSPATH, Reactome и др.). Для представления и работы с этими данными BioUML использует подход визуального моделирования, который предполагает наличие двух основных компонент:

- графическое представление модели: создание и редактирование модели визуально, путем добавления новых элементов;
- автоматическая генерация исполняемого кода для численных расчетов на основе визуального представления.

Такой подход широко используется для моделирования различных систем. В качестве примера можно привести MATLAB SIMULINK, Anylogic [Карпов, 2006].

Схема визуального моделирования в BioUML представлена на рис. 2.1. Пользовательский интерфейс BioUML для визуального создания диаграмм представлен на рис. 2.2. В BioUML граф модели представляется визуально в виде диаграммы. Тип диаграммы определяет [Колпаков, 2011]:

- **типы вершин и ребер графа** – типы биологических и математических объектов и их взаимодействий, которые могут быть представлены на диаграмме;
- **графическую нотацию элементов** – условные обозначения для графического представления объектов на диаграмме в зависимости от их типа;
- **семантические правила и ограничения** - правила, соблюдение которых обеспечивает смысловую целостность и корректность модели.

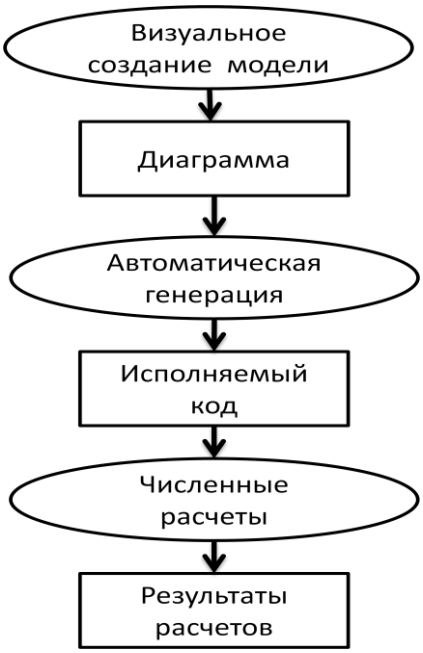


Рисунок 2.1 – Схема визуального моделирования в BioUML

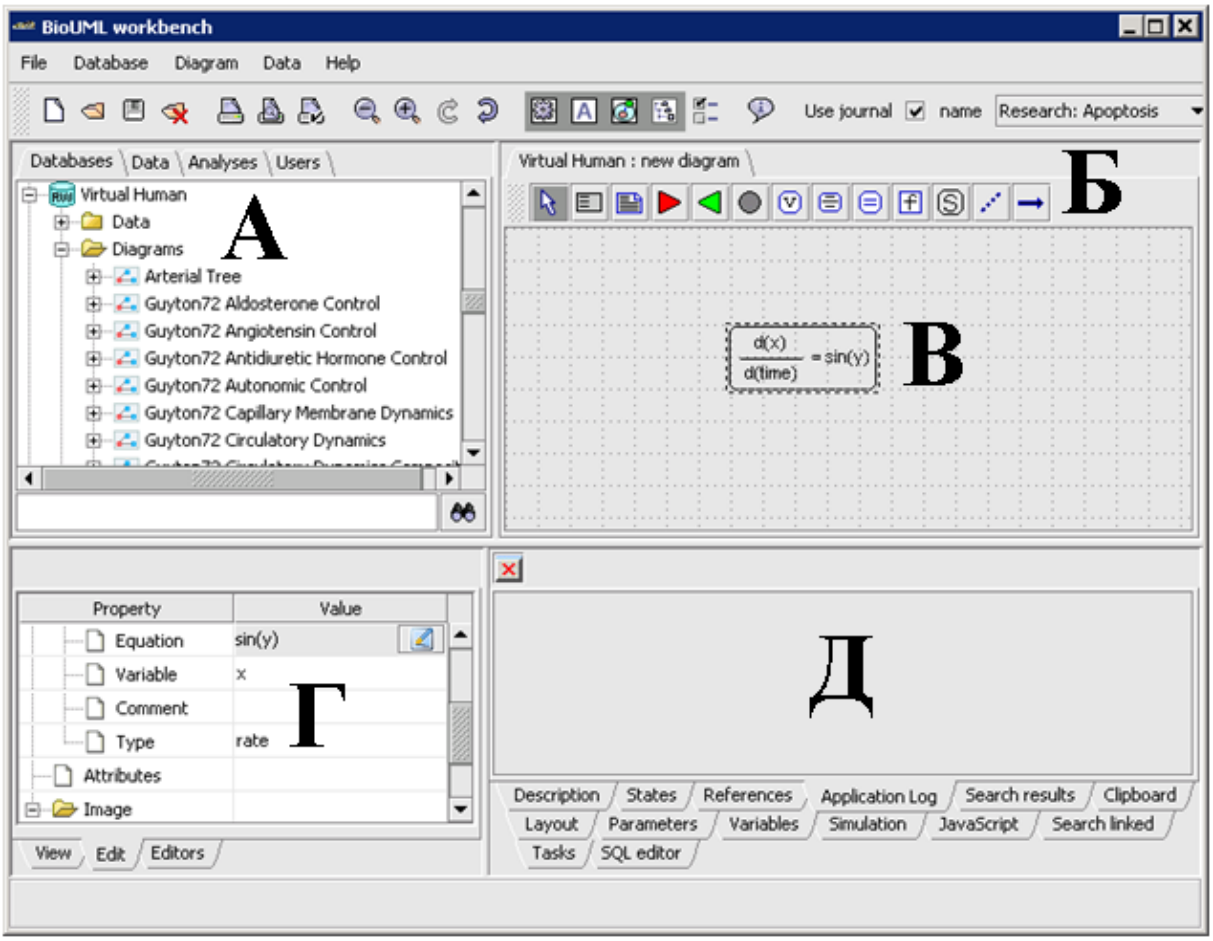


Рисунок 2.2 – Пользовательский интерфейс для визуального создания диаграмм в BioUML**.** А – список уже существующих моделей. Б – панель элементов, которые могут быть добавлены в выбранную модель. В – Графическое представление модели. Г – Параметры выбранного элемента модели (тип уравнения, правая часть, переменная). Д – Дополнительные панели для редактирования переменных, запуска численных расчетов, журнала событий и т.д.

BioUML позволяет создавать математические модели различного типа. В рамках данной работы, нас будут интересовать диаграммы, соответствующие дискретно-непрерывным математическим моделям. Основными типами таких диаграмм являются “Модель биологических путей”, “SBML модель” и “SBML-SBGN модель”. Эти диаграммы описывают количественные связи между биологическими объектами – биохимические реакции или процессы другого типа с заданными численно законами. Такие процессы могут быть проинтерпретированы как системы обыкновенных дифференциальных уравнений. Также диаграммы могут содержать математические объекты, такие как алгебраические, дифференциальные уравнения, операции присваивания, мгновенные события и т.д. Более подробно объекты данных диаграмм и их математический смысл описан в следующем параграфе.

Отличаются эти типы диаграмм способом хранения (“SBML модель” и “SBML-SBGN модель” хранятся в виде документов SBML), графической нотацией (“SBML-SBGN модель” использует SBGN, остальные – BioUML-нотацию) и некоторыми элементами. При этом эти типы математически эквивалентны и могут быть преобразованы друг в друга без существенной потери информации.

Графическая нотация биологических элементов данных диаграмм приведена в таблице 2.1. Графическая нотация математических и дополнительных элементов приведена в таблицах 2.2 и 2.3.

Единственный элемент SBML не поддержвиаемый в BioUML – **ограничения** (constraints), налагаемые на численные значени переменных. Они не влияют на расчеты, так как спецификация SBML не описывает, что делать в случае нарушения ограничения.

Таблица 2.1 – Графическая нотация основных биологических элементов диаграмм BioUML соответствующее обозначение SBGN-нотацией и элемент языка SBML.

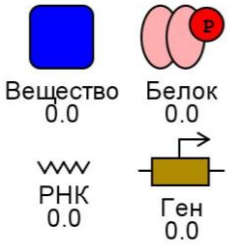
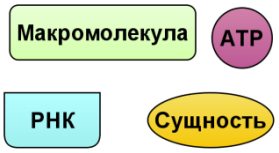
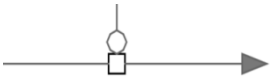
BioUML-нотация	SBGN	SBML	Описание
Процессы и их участники			
		Species	**Сущность** (ген, участок РНК, белок, химическое вещество) ассоциированная с математической переменной, выражающей ее количество (концентрацию).
		Compartment	**Компартмент** (отдел) клетки или организма, задающий локализацию отдельных компонентов.
		Reaction, Specie Reference (реактант, рпродукт, модификатор)	**Процесс** (напр. химическая реакция) описывает преобразование сущностей друг в друга. Ребра указывают на участников процесса: реактант, продукт, модификатор (напр. фермент).

Таблица 2.2 – Графическая нотация дополнительных элементов диаграмм BioUML. соответствующее обозначение SBGN-нотацией и элемент языка SBML.

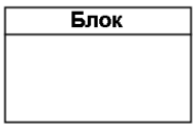
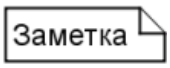
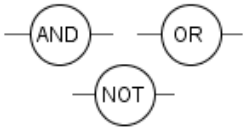
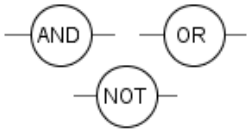
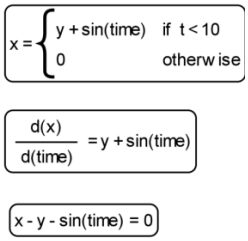
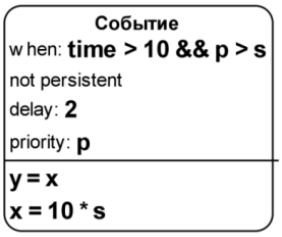
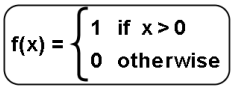
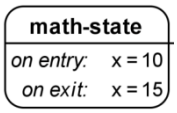
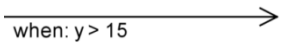
BioUML-нотация	SBGN	SBML	Описание
	-	-	**Блок** уравнений. Элемент для визуальной группировки уравнений.
	-	-	**Примечание.** Содержит произвольный текст, заданный пользователем.
	-	-	**Указатель примечания**, служит для связи примечания с элементом на диаграмме.
-	-	Units	**Единицы измерения.** В BioUML создаются и редактируются в отдельной вкладке.
		-	**Фенотип.** Процесс, не имеющий участников.
		-	**Логический элемент.** Служит для аннотации модификаторов процесса.

Таблица 2.3 – Графическая нотация математических элементов диаграмм BioUML, соответствующее обозначение в SBGN и элемент языка SBML.

BioUML-нотация	SBGN	SBML	Описание
	-	Rule: rate, algebraic, scalar.	**Уравнение** - дифференциальное, - алгебраическое, - присваивание.
	-	Event	**Мгновенное событие**, заключающееся в скачкообразном изменении значений переменных модели при выполнении определенного условия (например, в определенный момент времени или при заданном соотношении между переменными).
	-	Function	Заданная пользователем **функция**, которая может использоваться в уравнениях или реакциях модели.
	-	Event	**Состояние** системы, характеризуемое скачкообразным изменением переменных модели при входе в состояние и выходе из него. Вместе с переходами математически эквивалентно набору мгновенных событий.
	-	Event	**Переход** из одного состояния в другое при выполнении определенного условия.

Опишем также концепцию **состояний** модели, которая понадобится нам в дальнейшем. Состояние описывается списком изменений в модуле по сравнению с отсутствием состояний по умолчанию. Изменения могут быть следующих типов:

- удаление элемента (обозначается красным крестиком);
- добавление элемента (обозначается зеленой каймой);
- изменение свойств элемента (обозначается желтой каймой).

Назначение состояний – описывать различные модификации модели, например, это могут быть патологические состояния моделируемой биологической системы

или различные эксперименты с моделью. Пример приведен на рис. 2.3. Вверху – модель, состоящая из двух реакций между тремя сущностями. Внизу – состояние этой модели, в котором удалена сущность C и одна из реакций, свойства сущности A измены, так чтобы ее концентрация не менялась реакцией, и добавлено уравнений A = 2.

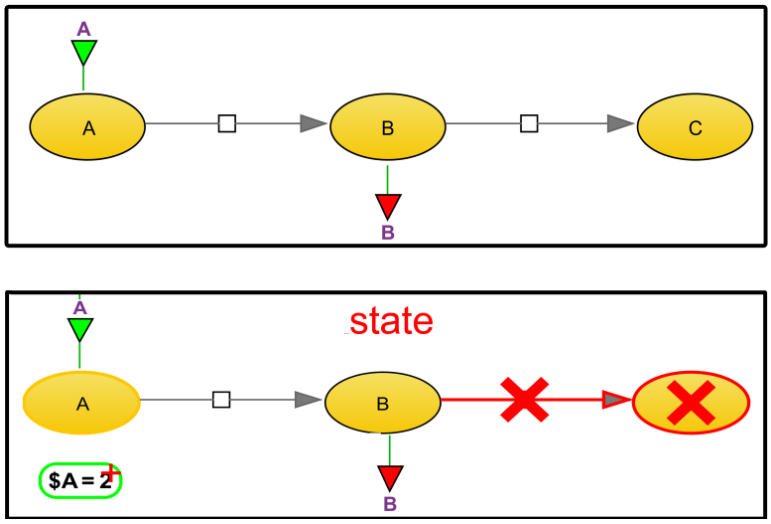


Рисунок 2.3 – Концепция состояний. Сверху – модель без состояния, снизу – с примененным состоянием “state”

2.2. Математическая модель

Данный параграф посвящен описанию математического смысла элементов модели типа “Модель биологических путей”.

Далее под **переменной** будем понимать некоторый объект, обладающий следующими атрибутами:

- имя;
- численное значение;
- единицы измерения;
- флаг, обозначающий, является ли значение переменной фиксированным или может меняться.

Также могут быть определены дополнительные свойства, например:

- тип единиц измерения (концентрация или количество вещества),
- флаг, обозначающий, может ли значение меняться в ходе химических реакций или других процессов и т.д. При работе в BioUML

дополнительные свойства могут быть определены с помощью технологии динамических свойств.

Сущность (ген, РНК, белок и т.д.) Соответствует переменной математической модели, может ссылаться на запись в базе данных. Имеет пространственную локализацию внутри компартмента. Может участвовать в реакциях в трех ролях: реактант, продукт, модификатор. Соответствующую сущности математическую переменную будем обозначать тем же символом что и саму сущность. Переменная описывает концентрацию или количество вещества соответствующей сущности. Важным свойством, которым мы будем пользоваться в дальнейшем, является **граничное условие** (boundary condition), в случае, если оно установлено – значение переменной не может меняться в ходе реакций (но может подчиняться закону, заданному дифференциальным или алгебраическим уравнением).

Реакция. Транслируется в дифференциальные уравнения, описывающие динамику переменных, соответствующих веществам-участникам реакции [Варфоломеев, Гуревич, 1999]. При определенных условиях, реакции также могут быть интерпретированы как стохастические процессы. Таким образом, одна и та же BioUML-диаграмма может быть интерпретирована либо как ОДУ-модель, либо как стохастическая модель [Gillespie, 2007; Gibson and Bruck, 2002].

Параметр. Математическая переменная модели, для которой нет выделенного элемента на диаграмме. Может быть добавлена в модель путем явного упоминания в каком-либо математическом выражении (уравнении, законе реакции и т.д.). Математический смысл у нее такой же, как и у переменной, соответствующей сущности, отличается только набор свойств. В дальнейшем, если специально не оговорено иное, мы не будем различать параметры и переменные соответствующие сущностям.

Обыкновенное дифференциальное уравнение. Задает динамику некоторой переменной x зависимостью вида:

$$\frac{dx}{dt} = f(X, t).$$

Вместе с начальными условиями эти уравнения формирует задачу Коши, которая может быть численно решена одним из методов, встроенных в BioUML.

Алгебраическое уравнение. Задает зависимости между переменными модели, которые должны быть выполнены все время функционирования модели. В частности, это могут быть различные законы сохранения. Общий вид зависимости:

$$f(X,t) = 0.$$

Формируют систему (в общем случае нелинейных) алгебраических уравнений.

Правило присваивания. Напрямую выражают значения одних переменных через другие уравнениями вида:

$$x(t) = f(X,t).$$

Присваивания делятся на два типа: присваивания, которые выполняются один раз в начальный момент времени (**начальное присваивание**) и присваивания, выражающие зависимости, которые должны быть выполнены в течение всего времени функционирования модели. Поскольку присваивания первого типа просто задают начальные значения переменных системы, в дальнейшем будем говорить только о присваиваниях второго типа.

Мгновенное событие. Задает скачкообразные изменения переменных модели при выполнения определенного **условия**:

$$trigger\colon \mathbb{R}^N \times \mathbb{R}_+ \longrightarrow \{\text{"истина"},\text{"ложь"}\}$$

Если в момент времени $t_{trigger}$ значение выражения $trigger(X,t)$ меняется с “ложь” на “истина”, то будем говорить, что событие **активируется** (triggers) в момент времени $t_{trigger}$. Событие задает набор присваиваний, которые должны быть осуществлены в случае активации события. Этот процесс называется **выполнением** (execution) события. Выполнение происходит через заданный интервал времени – **задержку** события после активации.

Таким образом, для события различаются **время активации** события $t_{trigger}$ и **время выполнения** события: $t_{exec} = t_{trigger} + \Delta$.

Состояния и переходы между ними. Описывают состояния, в которых может находиться система (например, это могут быть различные стадии клеточного цикла– G1, S, G2, M или состояния систолы и диастолы) и условия, при которых осуществляется переход между ними. Математически эквивалентны набору событий, так что в дальнейшем не будем на них останавливаться.

Пользователь создает модель, добавляя нужные уравнения на диаграмму, после чего BioUML автоматически из этих уравнений формирует систему ОДУ с мгновенными событиями. Для того чтобы созданная система была корректной, на содержание диаграммы накладываются следующие **семантические ограничения**:

1) одна и та же переменная не может находиться в левой части нескольких дифференциальных уравнений и/или правил присваивания. Исключение составляет начальное присваивание, которое может быть создано в дополнение к другому уравнению;
2) правила присваивания не должны иметь циклических зависимостей;
3) количество неизвестных в алгебраических уравнениях должно равняться количеству таких уравнений. BioUML автоматически определяет переменную как неизвестную в уравнении $\mathrm{g(X,t)} = 0$, если она явно входит в левую часть уравнения, и, кроме того, для нее нет правила присваивания или дифференциального уравнения и она не определена пользователем как константа.

Если все эти ограничения соблюдены, на основе содержания диаграммы можно сформировать систему ОДУ, которая в общем виде выглядит следующим образом:

$$\begin{cases} \dfrac{dx}{dt} = f(x, y, z, t), \\ g(x, y, z, t) = \overline{0}, \\ z(t) = h(x, y, z, t). \end{cases} \tag{4.1}$$

Кроме того, на систему влияет набор мгновенных событий:

$$\text{ЕСЛИ } trigger_i(X, t) \text{ ТО } X = A_i(X, t), \qquad i = 1, \dots, m, \tag{4.2}$$

здесь:

$X = (x_1, \dots, x_N)^T$ – вектор всех переменных модели;

$x = (x_1, \dots, x_d)^T$ – переменные модели, для которых есть дифференциальные уравнения;

$y = (x_1, \dots, x_a)^T$ – неизвестные алгебраической системы (4.2);

$z = (z_1, \dots, z_s)^T$ – переменные модели, для которых есть правила присваивания;

$f: \mathbb{R}^N \times \mathbb{R}_+ \longrightarrow \mathbb{R}^d$, $g: \mathbb{R}^N \times \mathbb{R}_+ \longrightarrow \mathbb{R}^a, h: \mathbb{R}^N \times \mathbb{R}_+ \longrightarrow \mathbb{R}^s$ – векторные функции;

$trigger_i(X, t)$ – условие i-го события.

Система (4.1-4.2), дополненная начальным значением $X^0 = (x_1^0, \dots, x_N^0)$ и интервалом времени $[t^0, t^M]$, формирует серию задач Коши.

2.3. Численные расчеты в BioUML

Проведением численных расчетов в BioUML управляет **инструмент численного решения** (SimulationEngine). На основе содержания диаграммы, инструмент численного решения автоматически генерирует численную модель (Model) соответствующего типа, пригодную для расчетов. Кроме того, он определяет список численных решателей (Simulator), пригодных для сгенерированной модели, запускает численные расчеты, обеспечивает вывод результатов на график и сохранение результатов в файл или базу данных. На рис. 2.4. приведена схема взаимодействия основных классов и компонентов архитектуры визуального моделирования в BioUML.

Пользователь осуществляет выбор одного из подходящих инструментов для численных расчетов. В настройках инструмента, пользователь выбирает начальное время расчетов, шаг, с которым будут сохраняться или выводиться на график результаты, а также конечное время. Также пользователь выбирает один из численных решателей, предоставляемых данным инструментом, и настраивает параметры выбранного решателя. Список численных решателей ОДУ-моделей с мгновенными событиями:

- Euler – классический метод Эйлера;

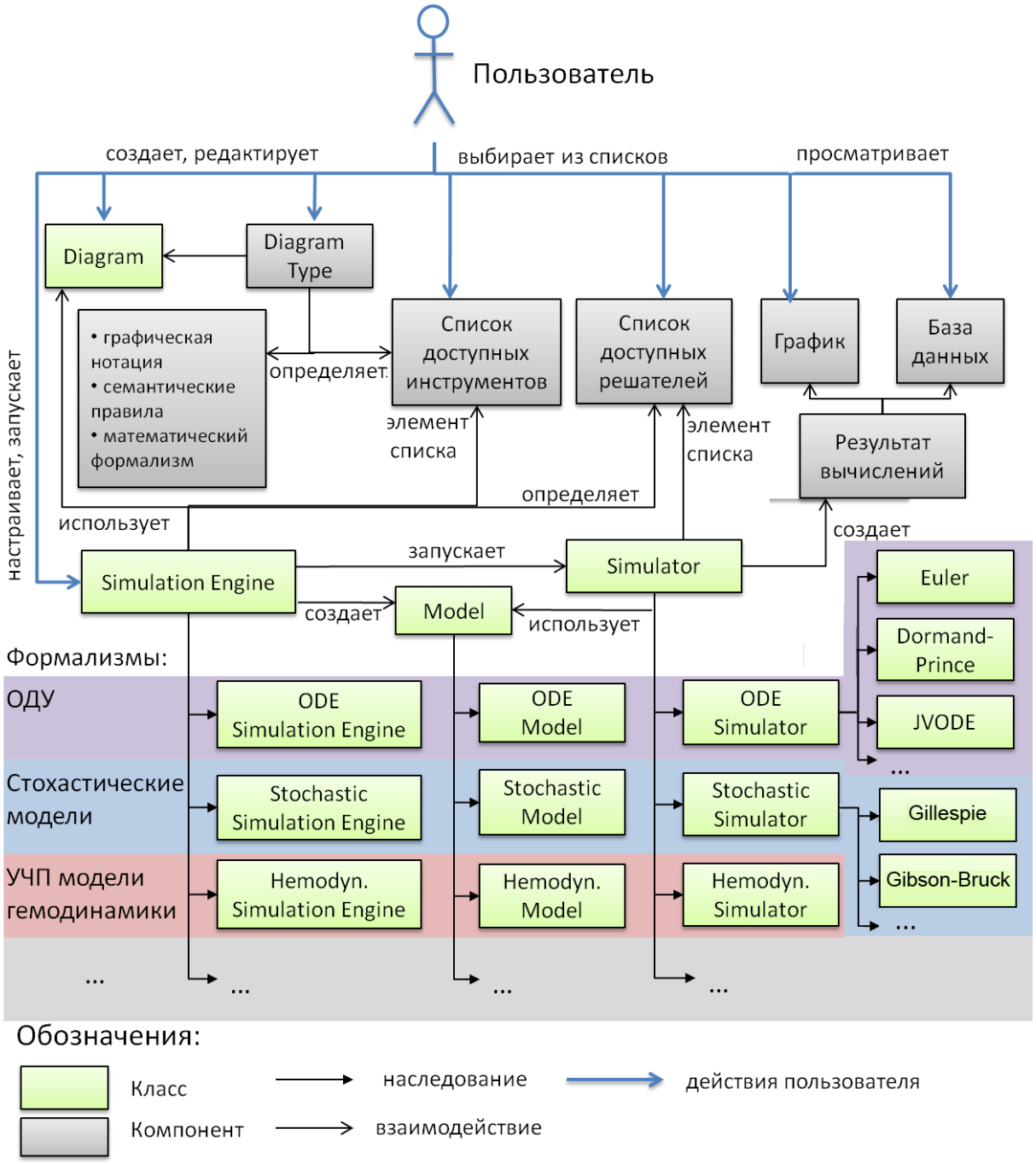


Рисунок 2.4 – Основные компоненты и классы архитектуры визуального моделирования в BioUML

- Dormand-Prince – классический численный решатель, использующий вложенные схемы Рунге-Кутты 4 и 5 порядка для контроля точности решения и автоматического управления длиной шага [Dormand, Prince, 1980]. Может использоваться для решения нежестких систем. Этот метод используется в качестве основного в среде MATLAB (ode45);

- Imex [Patterson, 2002] – метод, одновременно использующий явную и неявную схемы семейства Рунге-Кутты. Предполагается, что правая часть системы дифференциальных уравнений представима в виде суммы

$$\frac{dx}{dt} = f(X,t) = f_1(X,T) + f_2(X,T),$$

где задача $\frac{dx}{dt} = f_1(X,T)$ предполагается нежесткой, в то время как задача для $f_2(X,T)$ – жесткая. Кроме того, предполагается, что функция $f_2(X,T)$ – линейная. Решение системы ищется в виде суммы решений двух задач двумя различными методами;

Если в модели присутствуют мгновенные события, то используется специальный численный решатель EventLoopSimulator, содержащий внутри себя выбранный пользователем решатель. Шаг работы данного решателя состоит в следующем.

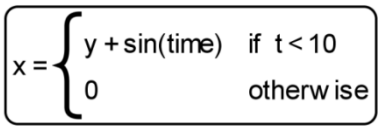
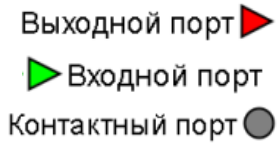
1. Запустить внутренний решатель (напр. Dormand-Prince), дождаться завершения его работы.
2. Если внутренний решатель обнаружил мгновенное событие, то:
   - выполнить произошедшие события,
   - установить новые начальные значения,
   - перейти к шагу 1.
3. Завершить расчеты.

Эта схема подразумевает, что внутренний численный решатель отслеживает условия мгновенных событий и находит моменты времени (с заданной погрешность) в которые эти условия становятся выполненными.

2.4. Диаграммы типа “Артериальное дерево”

Отдельный тип диаграмм в BioUML специально разработан для работы с моделями тока крови по сосудистому дереву. Модель включает в себя набор сосудов составляющих бинарное дерево. Конфигурацию сосудов в дереве и их свойства задает пользователь. Графическая нотация элементов дерева представлена в таблице 2.4.

Таблица 2.4 – Графическая нотация диаграммы типа “Артериальное дерево”

Графическая нотация	Описание
	**Точка бифуркации** (ветвления). Вспомогательный элемент, позволяющий соединять сосуды между собой.
	Сосуд. Характеризуется следующими свойствами: название, длина, площадь поперечного сечения, коэффициент жесткости.
	**Сердце**. Вспомогательный элемент, указывающий точку, через которую кровь втекает в дерево.
	**Уравнение.** Присваивает переменной в левой части значение выражения в правой части.
	**Контрольная точка.** Выводит значение параметров сосудов на график во время численных расчетов. Пользователь определяет: - сосуд, которому соответствует точка, - область сосуда, в которой замеряется значение: начало сосуда, середина или конец, - тип переменной: давление, полное давление, поток крови, скорость потока, сопротивляемость или площадь сечения сосуда.
	**Порт** (входной, выходной, контактный) – указывает на переменные, значения которых должны быть переданы в модель извне или переданы из модели наружу. Подробнее см. главу 3.

Подробное описание математической модели гемодинамики можно найти в [Блохин и др., 2009]. При создании модели использованы следующие предположения:

1. Осевая симметрия сосуда. Все величины не зависят от угловой координаты φ.
2. Радиус трубки R является функцией времени t и осевой координаты z. Стенка сосуда является упругой и перемещается только вдоль радиального направления.
3. Ось цилиндра зафиксирована и не перемещается со временем.
4. Давление постоянно на каждом осевом сечении.
5. Массовые силы отсутствуют.

6. Преобладает осевая компонента скорости u_z. По сравнению с ней компоненты скорости, ортогональные оси z, являются пренебрежимо малыми.

Система уравнений для модели из одного сосуда выглядит следующим образом:

$$\begin{cases} \partial_t A + \partial_z Q = 0, \\ \partial_t Q + \alpha \partial_z \left(\dfrac{Q^2}{A} \right) + \dfrac{A}{\rho} + K_r \dfrac{Q}{A} = 0, \end{cases}$$

где $A(t,z)$ – площадь осевого сечения; $Q(t,z)$ – средний поток крови через сечение; $P(t,z)$ – давление крови в сосуде; α – коэффициент Кориолиса; ρ – постоянная плотность крови; $K_r = 8\pi\nu$ – коэффициент трения о стенки сосуда; ν – постоянный коэффициент вязкости крови. Давление в сосуде определяется в соответствии с законом Гука:

$$p(A, A_0, \beta) = \beta \frac{\sqrt{A} - \sqrt{A_0}}{A_0},$$

где A_0 – площадь сечения сосуда в ненапряженном состоянии, $\beta = \sqrt{\pi} h_0 E$ – характеристика упругости сосуда, E – модуль Юнга, h_0 – постоянная толщина стенки сосуда. При выполнении естественного физического условия $A > 0$ система строго гиперболическая и имеет по одной характеристике, уходящей с каждой из границ $z = 0$ и $z = l$, где l - длина сосуда, поэтому на этих границах должно быть поставлено по одному условию [Блохин и др., 2009].

Модель, содержащая n сосудов, описывается системой уравнений:

$$\partial_t U_i + B(U) \partial_z U_i = S(U_i), \qquad U_i = (A_i \quad Q_i)^T \;\; i = \overline{1,n},$$

Для проведения численных расчетов необходимо задать граничные условия на свободных концах сосудов – аорты и всех **терминальных** сосудов, т. е. таких, которые не разветвляются на два других. Множество таких сосудов обозначим через $\aleph$. В [Biberdorf et al., 2012] предложен алгоритм “упаковки” бинарного дерева, позволяющий записать граничные условия в матричном виде $LU = \varphi$, $RU = \psi$, где матрицы L и R имеют размерности $2n \times n$. В модели могут быть установлены следующие граничные условия:

1. Входящий поток крови в аорту $Q_1(t, l_1) = Q_{in}(t)$.
2. Давление на входе в аорту $P_1(t, l_1) = P_{in}(t)$.

На выходе из артериального дерева могут быть установлены следующие условия:

1. Выходящий поток крови из терминальных сосудов:

$$Q_i(t, l_i) = Q_{i,out}(t),\ i \in \aleph.$$

2. Давление на выходе из терминальных сосудов:

$$P_i(t, l_i) = P_{i,out}(t),\ i \in \aleph.$$

3. Условие фильтрации, отражающее процесс фильтрации крови через капилляры. При этом задается соответствие между потоком крови и давлением на выходе из последних сосудов. Внешними параметрами являются давление в венах P_{veins} и сопротивляемость току крови в капиллярах $R_{capillary}$: $Q_i(t, l_i) = \frac{P_i(t,l_i) - P_{veins}}{R_{capillary}},\ i \in \aleph.$

С использованием диаграммы типа “Артериальное дерево” в BioUML была реализована модель тока крови по 55 крупнейшим артериям человеческого организма [Евшин и др., 2009]. Характеристики отдельных сосудов взяты из работ [Stergiopulos et al., 1992; Westerhof et al., 1969] с исправлениями из [Wang and Parker, 2004; Lamponi, 2004] и представлены в Приложении А. Модель в пользовательском интерфейсе BioUML представлена на рис. 2.5.

В рамках данной работы, тип диаграмм “Артериальное дерево” был доработан автором следующим образом:

1. Переработан механизм численного расчета для получения результатов расчетов динамически – результаты выводятся на график сразу при достижении численным решателем соответствующей временной точки (ранее результаты выводились после окончания всех расчетов).
2. Добавлен элемент “контрольная точка”, позволяющий выводить на график параметры отдельных сосудов (поток, давление, полное давление, площадь сечения, скорость потока, сопротивляемость) во время расчетов.
3. Добавлены элементы для интерфейсных портов (входного, выходного, контактного).

4. Численная модель теперь создается с помощью генерации кода аналогично ОДУ-моделям. Это, в частности, позволило вычислять значения заданных пользователем уравнений в модели
5. Добавлена возможность создавать уравнения.

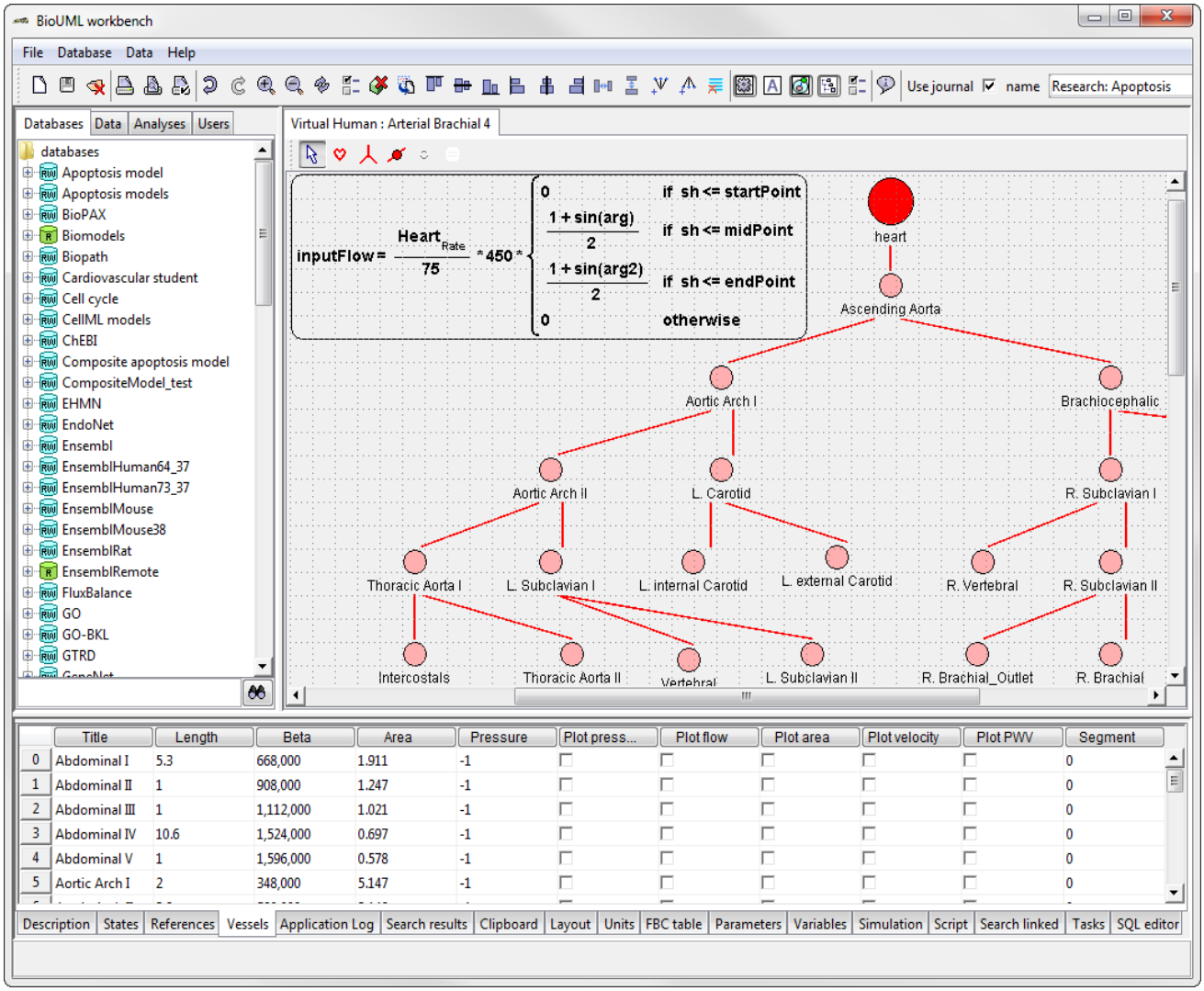


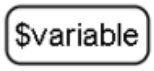
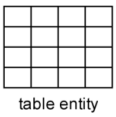
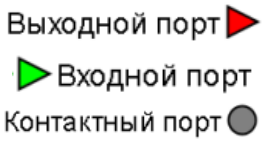
Рисунок 2.5 – Пользовательский интерфейс визуального создания диаграммы типа “Артериальное дерево”. Графическая нотация приведена в таблице 2.3

2.5. Диаграммы типа “Модель физиологических процессов”

Модели, описывающие различные физиологические процессы в организме, такие, как ток крови, секреция гормонов, зависимости между различными параметрами организма, как правило, не могут быть записаны в виде набора процессов. Визуально, модели таких систем представляют собой системы дифференциальных и алгебраических уравнений. Если модель достаточно

большая – становится трудно отследить зависимости между этими уравнениями. Для упрощения работы с такими моделями в BioUML, в рамках данной работы автором был реализован тип диаграммы “Модель физиологических процессов”. Диаграмма может содержать следующие элементы: уравнения, события, состояния, переходы между состояниями, примечания, указатели примечания, блоки и порты. Помимо этого добавлены дополнительные элементы, их графическая нотация представлена в таблице 2.5.

Таблица 2.5 – Графическая нотация дополнительных элементов модели физиологических процессов

Графическая нотация	Описание
	Элемент, ассоциированный с **переменной**.
	**Зависимость** между уравнениями: положительная (красным), отрицательная (синим).
	**Табличный элемент**. Задает зависимость между двумя переменными в табличной форме.
	**Порт** (входной, выходной, контактный) – указывает на переменные, значения которых должны быть переданы в модель извне или переданы из модели наружу. Подробнее смотри главу 3.

Переменная. Элемент на диаграмме, соответствующий переменной. В отличие от сущности, переменная соответствует не биологическому объекту, (белок, химическое вещество и т.п.), а свойству или характеристике объекта (длина или упругость сосуда, объем крови и т.п.).

Зависимости между уравнениями. Автоматически генерирующиеся элементы, указывающие на связь между уравнениями модели.

Табличный элемент. Задает (в виде таблицы) зависимость между двумя переменными или между переменной и временем. Табличному элементу на диаграмме соответствует таблица, хранящаяся в BioUML. По таблице автоматически генерируется кубический сплайн, который представляется в модели в виде правила присваивания.

$$x(y) = \begin{cases} a_1 + b_1 y + c_1 y^2 + d_1 y^3, & y \le y_1, \\ a_2 + b_2 y + c_2 y^2 + d_2 y^3, & y \in (y_1, y_2), \\ \dots & \\ a_n + b_n y + c_n y^2 + d_n y^3, & y \ge y_n. \end{cases}$$

Зависимости автоматически генерируются при изменении пользователем диаграммы. Зависимость между двумя уравнения e_1 и e_2 задается, если переменная, вычисляемая в e_2, явно входит в правую часть e_1. При этом, если приращение этой переменной увеличивает значение правой части e_1 при остальных аргументах считающихся постоянными, устанавливается **усиливающая связь** $(e_2 \to e_1)$, если уменьшает значение – устанавливается **ослабляющая связь** $(e_2 \dashv e_1)$, если не изменяет значение – устанавливается **неопределенная связь.** Формально:

$$e_1 : x = f(X, t), \qquad e_2 : y = g(X, t), \qquad \Delta > 0,$$
$$f(X, y, t) < f(X, y + \Delta, t) \Rightarrow e_2 \to e_1,$$
$$f(X, y, t) > f(X, y + \Delta, t) \Rightarrow e_2 \dashv e_1.$$

Если для переменной y создан входной или интерфейсный порт, связь устанавливается между портом и уравнением e_1 по аналогичным правилам. Если выходной порт ассоциирован с переменной, определяемой некоторым уравнением (т. е. переменная находится в его левой части уравнения), то между уравнением и портом устанавливается неопределенная связь. Связи не меняют математический смысл диаграммы, их назначение – наглядное представление путей распространения сигналов между уравнениями модели от входных портов – к уравнениям модели и далее к выходным портам.

Диаграмма имеет два различных варианта представления: полный вариант (рис. 2.6) – все уравнения модели явно представлены на диаграмме и краткий вариант (рис. 2.7) – уравнения представлены только именем переменной, рассчитываемой в данном уравнении.

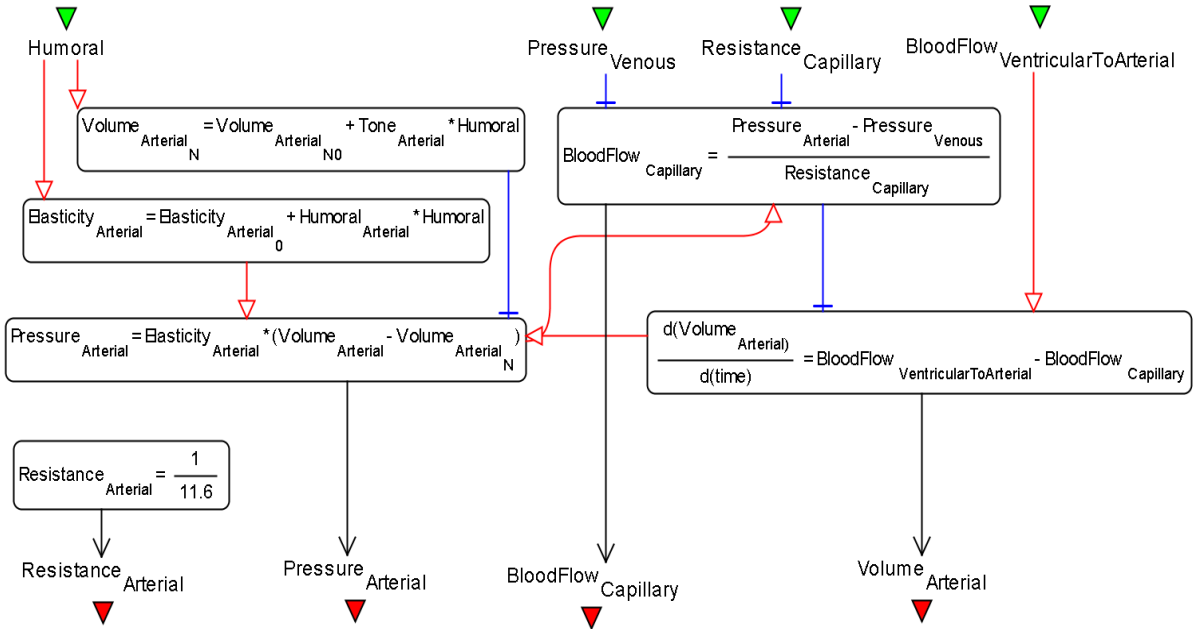


Рисунок 2.6 – Модель артериальной системы, реализованная в виде диаграммы типа “Модель физиологического процесса”. Полное представление

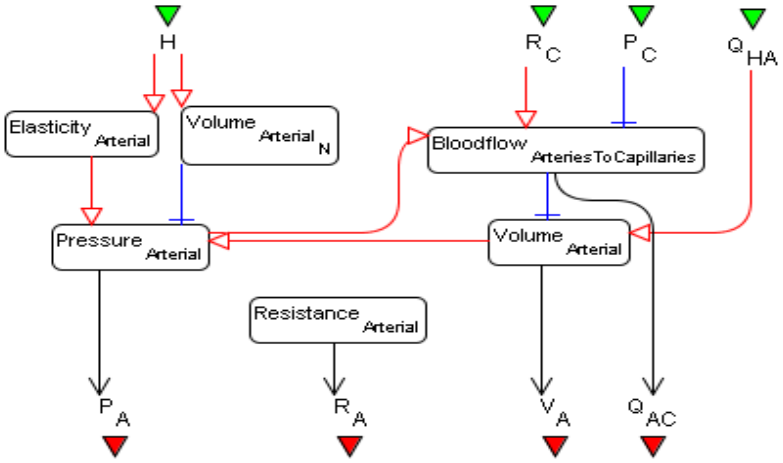


Рисунок 2.7 – Модель артериальной системы, реализованная в виде диаграммы типа “Модель физиологического процесса” – краткое представление

2.6. Расширенная обработка событий

2.6.1.Свойства событий

В рамках данной работы, обработка событий в платформе BioUML была существенно переработана автором на основе новой версии SBML l3v1 [Hucka et al., 2010], а именно добавлены следующие свойства событий, отсутствовавшие в предыдущей версии SBML l2v4.

Приоритет указывает на порядок, в котором должны быть выполнены события в том случае, если их времена выполнения совпадают. Приоритет

рассчитывается в момент времени выполнения события и может зависеть от переменных модели и времени:

$$p\colon \mathbb{R}^N \times \mathbb{R}_+ \longrightarrow \mathbb{R} \cup \{-\infty, +\infty\}.$$

При этом его необходимо рассчитывать непосредственно перед выполнением каждого события, так как события могут влиять на приоритет других событий, меняя его. Если несколько событий имеют одинаковый приоритет – выбор очередного события должен быть случайным с равной вероятностью для всех событий с одинаковым приоритетом. Если событие не имеет приоритета, то очередность его выполнения не задается спецификацией. В нашем случае будем полагать приоритет таких событий равным $-\infty$.

Неустойчивость означает, что событие выполняется только в том случае, если его условие остается истинным с момента активации и до момента выполнения. Неустойчивое событие может быть не выполнено и в том случае, если времена активации и выполнения совпадают, так как другие события, выполняющиеся в тот же момент времени, но с более высоким приоритетом, могут изменить условие его выполнения.

Начальное состояние – индикатор, указывающий на то, активировано ли событие до начального момента времени $t = t_0$. Если событие не активировано до начального момента времени и при этом условие события выполнено при (X_0, t_0), то событие считается активированным в начальный момент времени. Если же индикатор указывает на то, что событие активно до начального момента времени, то событие не может быть активировано в момент времени t_0 (так как оно было активировано в некоторый момент времени предшествующий t_0).

Способ выполнения события – индикатор, регулирующий выполнение события. Есть два варианта:

1. Использовать значения переменных в момент времени активации события: $X(t_{execution}) = A\big(X\big(t_{trigger}\big), t_{trigger}\big)$.
2. Использовать значения переменных в момент времени выполнения события: $X(t_{execution}) = A(X(t_{execution}), t_{execution})$.

Каскады событий. Если в процессе выполнения какого-либо события, произошла активация других событий – они должны быть выполнены в тот же момент времени, в соответствии со своими приоритетами..

2.6.2. Предварительная обработка событий

Для корректной генерации кода для событий была реализована программа предварительной обработки (препроцессор), трансформирующая диаграмму перед генерацией кода. Измененная схема моделирования представлена на рис. 2.8. Для каждого события в диаграмме выполняется следующий алгоритм.

Пусть имеем событие с условием активации $trigger(X,t)$, правилом присваивания заданным функцией $A(X,t)$ и задержкой Δ и использующее значения в момент активации. Событие удаляется из диаграммы. Вместо него создаются два новых события: **вычисляющее** событие и событие **присваивания**. Оба события имеют одинаковые условия выполнения, совпадающие с условием выполнения первоначального события. Первое событие во время выполнения вычисляет значение выражения $A(X,t)$ и присваивает его временной переменной X_{temp}. Второе – выполняет присваивание: $X = X_{temp}$. Вычисляющее событие имеет приоритет равный $+\infty$. Событие присваивания имеют тот же приоритет, что и первоначальное событие. Если приоритет не установлен, он считается равным $-\infty$. Таким образом, гарантируется, что вычисление выражения всегда происходит раньше его использования.

Если $\Delta \neq 0$, то формируется очередь из событий. В момент выполнения вычисляющего события, в очередь добавляется новое событие с условием выполнения $t > t_{rigger,i} + \Delta$, правилом присваивания $X = X_{temp,i}$ и задержкой $\Delta_i = 0$. При этом условие выполнения события присваивания определяется как условие выполнения первого события в очереди. Правила присваивания события определяются как правила присваивания первого события в очереди. В момент выполнения события присваивания первое событие из очереди удаляется.

Если событие неустойчиво, то генерируется дополнительное событие, условие выполнения которого противоположно условию выполнения основного

события $trigger_{off}(X,t) = \sim trigger(X,t)$. Присваивание такого события: $I_{off} =$ "истина". Во время выполнения, данное событие “отключает” основное событие. Условие выполнения основного события модифицируется следующим образом: $trigger_{new}(X,t) = trigger(X,t) \wedge \sim I_{off}$.

Рисунок 2.8 – Схема визуального моделирования в BioUML с использованием предварительной обработки

2.6.3. Обработка событий

Опишем, как выглядит расширенная обработка событий во время проведения численных расчетов, т. е. шаг работы программы EventLoopSimulator. Будем предполагать, что после предварительной обработки, описанной в предыдущем параграфе, в модели, для которой проводятся численные расчеты, присутствует m мгновенных событий. Пронумеруем эти события и будем обращаться к событию по его номеру.

Подготовка к первому шагу вычислений.

1. Вычислить условия всех событий в начальный момент времени при заданных начальных значениях: $trigger_i(X^0, t^0),\ i = \overline{1,m}$.

2. Сформировать список событий, произошедших в начальный момент времени (Здесь $Initial_i$ – начальное состояние *i*-го события):

$Event(X^0, t^0) = \{i \in \{1, \dots, m\} | \; trigger_i(X, t) \wedge \sim Initial_i\}$.

3. Запустить процедуру обработки событий из множества $\mathrm{Event}(\mathrm{X}^0, \mathrm{t}^0)$.

Шаг вычислений.

1. Запустить шаг внутреннего решателя системы ОДУ.
2. Если внутренний решатель завершил все численные расчеты, то завершить работу.
3. Если внутренний решатель не обнаружил мгновенных событий, то перейти к пункту 1.
4. Вычислить условия всех событий $trigger_i(X, t), i = \overline{1, m}$, сформировать список произошедших событий:

   $Event(X, t) = \{i \in \{1, \dots, m\} \, | trigger_i(X, t) = \text{"}истина\text{"}\}$.
5. Запустить процедуру обработки событий из множества $Event(X, t)$.
6. Перейти к 1.

Обработка событий из множества $\boldsymbol{Event(X, t)}$.

1. Вычислить приоритет для каждого из произошедших событий

$$p_i(X, t), i \in Event(X, t).$$

2. Сформировать список произошедших событий с максимальным приоритетом

$$\widehat{Event}(X, t) = \{i \in Event(X, t) | \; p_i(X, t) = \max_{Event(X,t)} p_i(X, t)\}.$$

3. Случайно выбрать одно событие из списка событий с максимальным приоритетом: $i_{cur}(X, t) \in \widehat{Event}(X, t)$..
4. Выполнить выбранное событие: $X_{i_{cur}} = A_{i_{cur}}(X, t)$. Убрать $i_{cur}(X, t)$ из множества $Event(X, t)$.
5. Вычислить условия всех событий $\widetilde{trigger}_i(X, t), i = \overline{1, m}$.
6. Для всех неустойчивых событий из $Event(X, t)$: если $\sim \widetilde{trigger}_i(X, t) \wedge trigger_i(X, t)$, то: убрать *i*-е событие из $Event(X, t)$.

7. Для всех событий. Если $\widetilde{trigger}_i(X,t) \wedge trigger_i(X,t)$, то добавить i-ое событие в множество $Event(X,t)$.

Если обработка происходит в начальный момент времени t^0, то добавление события происходит при выполнении условия:

$$\widetilde{trigger}_i(X,t) \wedge trigger_i(X,t) \wedge \sim Initial_i$$

8. Если множество $Event(X,t)$ пусто, то завершить обработку событий, иначе – перейти к пункту 1.

2.7. Численный решатель CVODE

Программный пакет CVODE [Cohen, Hindmarsh, 1996; Hindmarsh et al., 2005] создан в Ливерморской национальной лаборатории им. Э. Лоуренса. Он написан на языке программирования C и основан на программном пакете LSODE [Radhakrishnan et al., 1993], написанном на языке FORTRAN. LSODE, в свою очередь, использует идеи и методы, предложенные профессором Вильямом Гиром [Gear, 1967]. Пакет содержит численные алгоритмы, предназначенный для решения задачи Коши для системы ОДУ первого порядка, разрешенной относительно производной:

$$\begin{cases} \dfrac{dx}{dt} = f(t,x), \qquad t \in (t^0, t^m). \\ x(t^0) = x^0, \end{cases} \tag{2.1}$$

где $x \in \mathbb{R}^n$, $f: \mathbb{R}^n \to \mathbb{R}^n$.

Метод Адамса основывается на рассмотрении задачи (2.1) в интегральном виде:

$$x(t^n) = x(t^{n-1}) + \int_{t^{n-1}}^{t^n} f(t, x(t))\, dt. \tag{2.2}$$

В (2.2) функция $f(t,x(t))$ заменяется интерполяционным многочленом, проходящим через точки $(f^{n-q}, t^{n-q}), \dots, (f^n, t^n)$. После интегрирования получаем формулу:

$$x^n = x^{n-1} + h \sum_{i=0}^{q} \beta_i f^{n-i}. \tag{2.3}$$

Метод дифференцирования назад (BDF – Backward Differential Formula) основывается на замене искомой функции $x(t^n)$ интерполяционным многочленом, проходящим через точки $(x^{n-q}, t^{n-q}), \dots, (x^n, t^n)$. После дифференцирования получаем формулу:

$$\sum_{i=0}^{q} \alpha_i x^{n-i} = h f^n. \tag{2.4}$$

Обе формулы (2.3) и (2.4) могут быть описаны следующим уравнением:

$$\sum_{i=0}^{K_1} \alpha_i \, x^{n-i} + h \sum_{i=0}^{K_2} \beta_i f^{n-i} = \bar{0}.$$

Для метода Адамса-Мултона: $K_1 = 1$, $K_2 = q = \overline{1,12}$. Для метода обратной производной: $K_1 = q = \overline{1,5}$, $K_2 = 0$. Применение обеих численных схем сводится к решению системы нелинейных уравнений:

$$G(x^n) \equiv x^n - h\beta_0 f(t^n, x^n) - a_n = \bar{0}, \tag{2.5}$$

$$a_n = \sum_{i=1}^{K_1} \alpha_i x^{n-i} + h \sum_{i=1}^{K_2} \beta_i x^{n-i}.$$

В рамках CVODE используются два различных метода:

1. Метод простой итерации. $x^{n(m+1)} = h\beta_0 f\left(t^n, x^{n(m)}\right) + a_n$.
2. Метод Ньютона. $\partial G / \partial x \left[x^{n(m+1)} - x^{n(m)}\right] = -G\left(x^{n(m)}\right)$, где

$$\partial G / \partial x = I - h\beta_0 \, \partial f / \partial x. \tag{2.6}$$

При этом могут использоваться три варианта приближения матрицы Якоби $\partial f / \partial x$:

1. Плотная матрица, т. е. вычисляются все элементы матрицы.
2. Ленточная матрица – вычисляются только элементы, для которых верно:

$$j > i - \mu_l, \qquad j < i + \mu_d, \tag{2.7}$$

где μ_l, μ_d – параметры метода, задаваемые пользователем. Остальные элементы полагаются равными нулю.

3. Диагональная матрица – вычисляются только диагональные элементы, остальные элементы матрицы Якоби полагаются равными нулю.

На каждом шаге вычислений CVODE контролирует погрешность полученного решения и автоматически варьирует порядок метода и величину шага в процессе вычислений. При этом для реализации схемы с переменным шагом используется “вектора Нордсика” $z^n = \left(x^n, hx^{n(1)}, \dots, hx^{n(k)}\right)$, где $x^{n(k)}$ – приближение к-й производной функции $x(t)$ в точке t^n.

В рамках данной работы пакет CVODE переписан автором с языка программирования C на язык Java и адаптирован к API BioUML. Создан JvodeSolver расширяющий базовый класс Simulator и реализующий схему численных вычислений CVODE. Перед запуском расчетов пользователь задает параметры метода в меню, созданном с помощью технологии BeanExplorer.

Параметрами данного численного решателя являются:

1. Абсолютная и относительная погрешности.
2. Метод решения: метод Адамса-Мултон или метод обратной производной.
3. Метод решения нелинейной системы (2.5): метод простой итерации или метод Ньютона.
4. Тип матрицы Якоби: плотная, ленточная или диагональная (если выбран метод Ньютона).
5. Значения μ_l и μ_d, используемые в (2.7) (если выбрана ленточная матрица).
6. Лимит количества шагов метода.
7. Минимальный шаг метода по времени.
8. Максимальный шаг метода по времени.

На каждом шаге вычислений алгоритм проверяет условия всех мгновенных событий в модели. При обнаружении выполненного условия – ищется момент времени (с заданной точностью) в который данное событие произошло. После этого алгоритм прекращает работу. Происходит обработка событий (см. параграф

2.4) и алгоритм опять запускается с той же самой задачей (2.1), но новым начальным значением вектора x.

На данный момент, решатель JvodeSolver является стандартным в платформе BioUML. Все численные расчеты, связанные с решением задачи Коши в рамках данной работы, проведены автором с его помощью.

2.8. Тестирование

Алгоритм численных расчетов в BioUML тестируется набором тестов, разработанных авторами SBML (http://sbml.org/Facilities/Database). Каждый тест включает в себя SBML-модель, параметры численного решения и ожидаемые результаты. Расширенная обработка событий и портирование численного решателя CVODE на язык Java позволили увеличить количество проходимых тестов с 927 (версия тестов от 15 ноября 2011 года) до 1126 теста (версия тестов от 22 мая 2013 года). Кроме того пройдены все из еще не опубликованного на данный момент дополнительного набора из 21 теста. Результаты доступны по адресу http://www.biouml.org/sbml_tests/overview%203.0.0.html и на рис. 2.9.

SBML	Details	Tests	Successful	Success rate	Time (s)
Level 1 version 2	by order by categories	250	250	100.0 %	26
Level 2 version 1	by order by categories	885	885	100.0 %	162
Level 2 version 2	by order by categories	1041	1041	100.0 %	334
Level 2 version 3	by order by categories	1041	1041	100.0 %	500
Level 2 version 4	by order by categories	1043	1043	100.0 %	655
Level 3 version 1 Core	by order by categories	1077	1077	100.0 %	181

Рисунок 2.9 – результаты тестирования инструмента численного решения BioUML. Распределение тестов по уровням SBML

Сводная таблица тестирования различных программных продуктов (включая BioUML, COPASI, iBioSim и другие) доступна на сайте SBML по адресу http://sbml.org/Facilities/Database/Simulator.

При этом надо отметить, что на данный момент (сентябрь 2016 г.) кроме BioUML еще только два программный продукт прошел все тесты – iBioSim [Myers et al., 2009] и Systems Biology Simulation Core Library [Keller et al., 2013].

2.9. Выводы по главе 2

В результаты данной работы возможности BioUML по численному моделированию биологических систем были существенно улучшены и расширены:

1. Реализован типа диаграмм “Модель физиологического процесса” для графического представления математических моделей, состоящих только из алгебраических и дифференциальных уравнений.
2. Доработан и улучшен тип диаграмм “Модель артериального дерева”:
   - генерация исполняемого кода для проведения численных расчетов
   - динамический вывод результатов расчетов на график,
   - возможность включать пользовательские уравнения в модель,
   - различные варианты граничных условий.
3. Улучшена и приведена в соответствие со спецификацией [Hucka et al., 2010] обработка мгновенных событий.
4. Добавлен шаг предварительной обработки модели (рис. 2.8).
5. Портирован с языка C и адаптирован к BioUML численный решатель CVODE.

ГЛАВА 3. МОДУЛЬНОЕ МОДЕЛИРОВАНИЕ

3.1. Основные определения

Под модулем мы будем понимать черный ящик, содержащий набор переменных и обладающий способностью принимать и отправлять сообщения. Сообщения содержат значения переменных модуля, при этом модуль определяет подмножества переменных, значения которых могут быть переданы из модуля и в модуль. Модули могут быть объединены связями, которые определяют, какие именно сообщения будут передаваться между ними. Таким образом, модульная модель описывает систему в терминах передачи сообщений между отдельными модулями.

Формально определим **модуль (module)** следующим образом:

$$M = (\mathcal{X}, \mathcal{I}, \mathcal{O}, \mathcal{C}),$$

где:

$\mathcal{X} = \{x_1, \dots, x_n\}$ – множество переменных модуля;

$\mathcal{I} \subseteq \mathcal{X}$ – множество **входных (input)** переменных модуля, т. е. переменные, значения которых определяются извне модуля;

$\mathcal{O} \subseteq \mathcal{X}$ – множество **выходных (output)** переменных модуля, т. е. переменные, чьи значения рассчитываются внутри модуля и могут быть переданы наружу;

$\mathcal{C} \subseteq \mathcal{X}$ – множество **контактных (contact)** переменных модуля, т. е. переменных, значения которых могут изменяться как снаружи модуля, так и внутри.

Будем также использовать термин **порт (port)**. Так, если модуль имеет интерфейсную переменную, будем говорить, что он определяет порт соответствующего типа (входной, выходной или контактный). Эта терминология может быть более удобной в том случае, когда модуль имеет переменные, не являющиеся интерфейсными (т.е. такие для которых порт не определен).

Множества $\mathcal{I}, \mathcal{O}, \mathcal{C}$ в совокупности определяют **интерфейс** модуля, соответствующие переменные будем называть **интерфейсными.** Заметим, что в общем случае:

$$\mathcal{I} \cap \mathcal{O} \cap \mathcal{C} \neq \emptyset,\ \mathcal{I} \cup \mathcal{O} \cup \mathcal{C} \neq \mathcal{X},$$

т. е. одна и та же переменная может являться одновременно входом и выходом модуля, либо не являться ни тем, ни другим и, таким образом, быть недоступной извне модуля. Как правило, между входными и выходными переменными модуля установлено какое-либо соответствие, которое позволяет вычислить значения выходных переменных в зависимости от заданных значений входных. В этом смысле модуль можно представить как функцию, которая по значению входных переменных определяет значения выходных переменных. При этом контактные переменные являются одновременно входными и выходными.

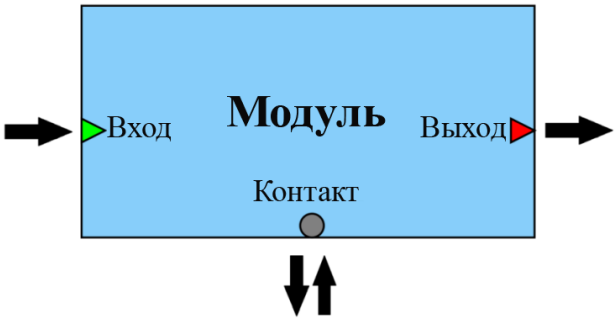


Рисунок 3.1 – Концепция модуля

Модулями могут быть математические модели различных биологических подсистем. Тогда входные переменные представляют собой сигналы, приходящие от других подсистем, а выходные – результат функционирования подсистемы. Модули, представляющие собой математические модели, будем также называть **подмоделями**, обычно модули создаются следующими способами:

1) модуль создается как отдельный функциональный блок, предназначенный для объединения с другими модулями;
2) модуль создается путем разложения большей модели на части.

Следует заметить, что благодаря обобщенному определению модуля, внутреннее содержание может быть любым. Обычно это некоторое описание соотношений между интерфейсными переменными модуля, которое может быть:

- количественным описанием,
- математической моделью (ОДУ, УЧП),

- модульной моделью,
- произвольным кодом (Java, java-script, MATLAB, R), рассчитывающим значения выходных переменных на основе входных и т. д.

Пусть есть два модуля: $M_1 = (\mathcal{X}_1, \mathcal{I}_1, \mathcal{O}_1, \mathcal{C}_1)$ и $M_2 = (\mathcal{X}_2, \mathcal{I}_2, \mathcal{O}_2, \mathcal{C}_2)$. **Связь (connection)** между ними определяется, как соответствие между их интерфейсными переменными. А именно определим два типа связи:

Направленная связь (directed connection) – это связь между выходной переменной одного модуля и входной переменной другого. Такая связь означает прямое влияние одной подсистемы на другую. Формально определим ее следующим образом:

$$(x, y, p), \qquad x \in \mathcal{I}_1, \qquad y \in \mathcal{O}_2, \qquad p\colon \mathbb{R} \longrightarrow \mathbb{R},$$

здесь p – функция, задающая зависимость между значениями x и y. В том случае, если p не задана, будем полагать, что значение x должно быть положено равным значению y. Введем обозначение $y \xrightarrow{p(y)} x$ или $y \to \mathrm{x}$, если нам не важен вид функции p. Кроме того, если необходимо, будем также использовать более подробные обозначения, указывающие на то, между какими модулями устанавливается связь: $(M_1, \mathrm{y}) \xrightarrow{\mathrm{p}} (\mathrm{M}_2, \mathrm{x})$ и $(M_1, \mathrm{y}) \to (\mathrm{M}_2, \mathrm{x})$.

Ненаправленная связь (undirected connection) устанавливается между двумя контактными переменными. Такая связь означает, что модули влияют друг на друга, одновременно изменяя значение общей переменной. Формально определим ее следующим образом:

$$(x, y, z), \qquad x \in \mathcal{C}_1, \qquad y \in \mathcal{C}_2,$$

z – переменная, свойства которой будут использованы вместо обеих переменных x и y. В качестве z может быть задана новая переменная либо использована одна из связанных: x или y. Если z не задана, то выбор между x и y осуществляется произвольно. Введем обозначение $\mathrm{x} \overset{z}{\leftrightarrow} \mathrm{y}$ или сокращенно $\mathrm{x} \leftrightarrow \mathrm{y}$. Также как и в предыдущем случае, иногда будем пользоваться более подробными обозначениями: $(M_1, \mathrm{y}) \overset{\mathrm{z}}{\leftrightarrow} (\mathrm{M}_2, \mathrm{x})$ и $(M_1, \mathrm{y}) \leftrightarrow (\mathrm{M}_2, \mathrm{x})$.

Цепочки связей. Далее если нам попадется ситуация вида: $(x \to \text{y}) \wedge (\text{y} \to \text{z}) \wedge \ldots \wedge (z \to \text{u})$, будем сокращенно писать $x \to \text{y} \to \text{z} \to \cdots \to \text{u}$. Аналогично для ненаправленной связи: $x \leftrightarrow \text{y} \leftrightarrow \text{z} \leftrightarrow \cdots \leftrightarrow \text{u}$.

Семантические ограничения. Исходя из смысла связей, естественно наложить определенные ограничения на их установление между моделями:

1. если переменная является входом для направленной связи, то других связей с ней установлено быть не может, т. е. не может быть цепочек вида $x \to y \leftarrow z, \;\; x \to y \leftrightarrow z$.
2. не может быть циклических направленных связей, т. е. цепочек вида $x \leftarrow y \leftarrow \cdots \leftarrow z \leftarrow x$. Иначе становится невозможно определить значение переменной;
3. связи не могут быть установлены между переменными одного и того же модуля. Все зависимости между входами и выходами модуля должны быть определены внутри него;

Теперь мы можем определить **модульную модель (modular model)**, как множество связанных между собой модулей. Формально это:

$$MM = (M, DC, UC, E),$$

где

$M = \{M_1, \ldots, M_N\}$ – множество модулей;

DC – множество направленных связей между модулями;

UC – множество ненаправленных связей между модулями;

E – **внешняя среда**, содержащая математическую модель, которая описывает условия, в которых находятся модули. Среда может влиять на модули таким же образом, как модули друг на друга – через установленные связи. Внешняя среда определяет два интерфейса – приватный и публичный, каждый из которых содержит по три множества входных, выходных и контактных переменных.

Приватный (private) интерфейс используется для изменения поведения модулей. С этой точки зрения, внешняя среда может рассматриваться как еще один модуль, взаимодействующий с включенными в нее модулями через данный интерфейс. Концепция приватных интерфейсов пояснена на рис. 3.2.

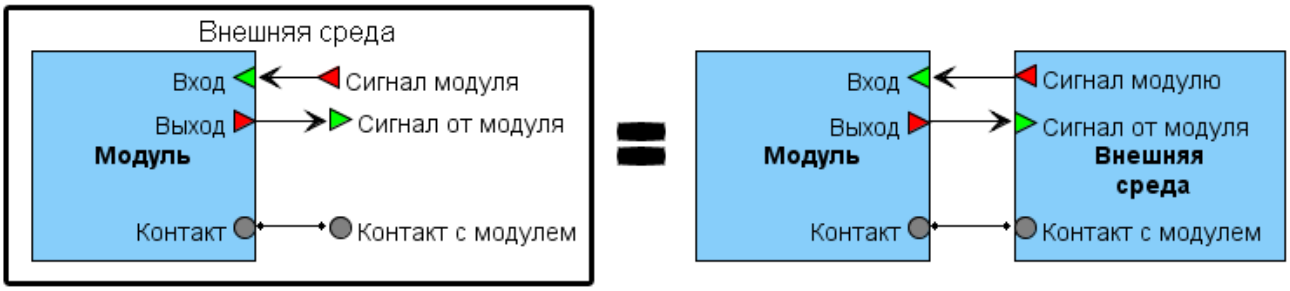


Рисунок 3.2 – Концепция приватных портов

Публичный (public) интерфейс используется для представления модульной модели в виде модуля и включения ее в другую модульную модель. В этом случае публичный интерфейс модульной модели служит интерфейсом ее представления в виде модуля.

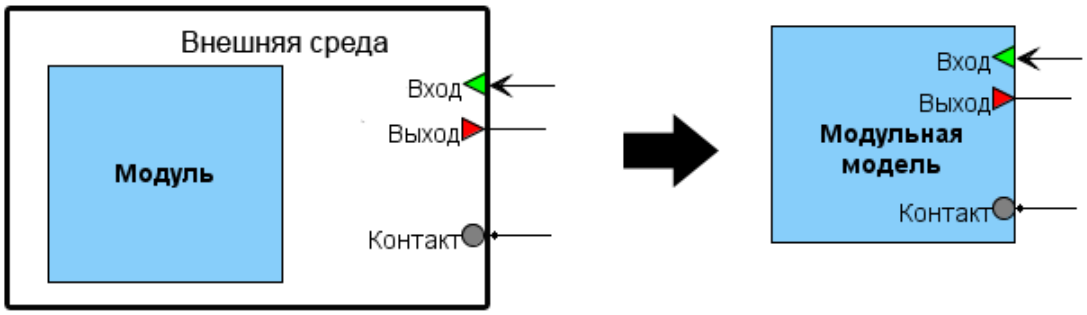


Рисунок 3.3 – Концепция публичных портов

Есть два способа создать публичный интерфейс в модульной модели:

1. По аналогии с обычным модулем – создать интерфейсную переменную во внешней среде.
2. “Расширить” интерфейс одного из модулей, включенных в модель до публичного интерфейса самой модели. Таким образом, значение переменной, соответствующей выходному порту модуля передается наружу модульной модели, либо наоборот, значение переменной, приходящее в модульную модель, передается напрямую одному или нескольким ее модулям (см. рис 3.4.).

Нужно отметить, что концепция публичных и приватных интерфейсов сходна с концепцией, используемой в языке CellML. С ее помощью модульная модель может быть представлена в виде модуля:

$$MM = (\{M_1, \dots, M_N\}, DC, UC, E) \longrightarrow M = \left(\bigcup_{i=1}^{N} \mathcal{X}_i, \mathcal{I}_{public}, \mathcal{O}_{public}, \mathcal{C}_{public}\right),$$

где $M_i = (\mathcal{X}_i, \mathcal{I}_i, \mathcal{O}_i, \mathcal{C}_i)$.

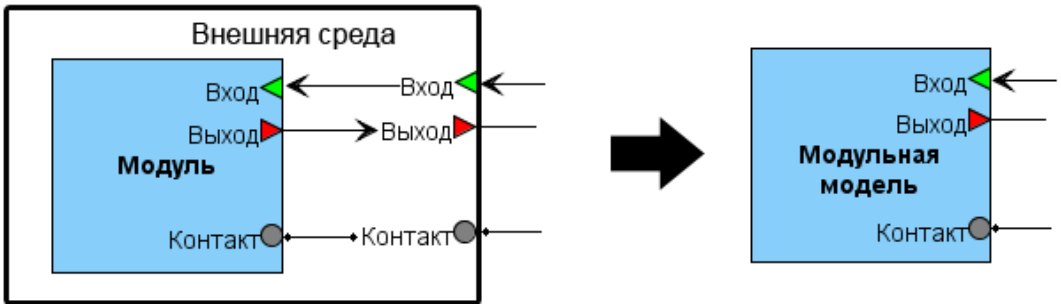


Рисунок 3.4 – Концепция вынесенных (propagated) портов

3.2. Представление в BioUML

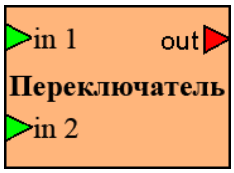
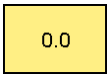
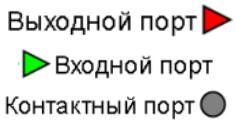
Для визуального представления модульных моделей в BioUML были созданы два новых типа диаграмм “Модульная модель” и “Агентная модель”. Второй тип отличается от первого дополнительными элементами и описан в параграфе 3.5. Ниже опишем тип диаграмм “Модульная модель” Каждая подмодель представлена на диаграмме специальным элементом содержащим ссылку на некоторую другую диаграмму в репозитории BioUML. Помимо модулей и связей между ними, диаграмма такого типа может содержать и другие элементы – уравнения, события, функции и другие, описанные в таблице 2.2. Все эти элементы описывают внешнюю среду, в которой функционируют модули. Графическая нотация новых элементов, добавленных в тип диаграмм “Модульная модель” приведена в таблице 3.1.

На данный момент подмодели могут быть следующих типов.

- “Модульная модель” (формируя иерархическую модульную модель),
- SBML-модель (в нотации SBGN или BioUML),
- Модель, содержащая список ОДУ, алгебраических уравнений и мгновенных событий,
- модель, содержащая набор химических реакций (могут быть интерпретированы как ОДУ или как случайные события).

Каждая подмодель может определять специальное состояние, в котором она находится в модульной модели. Состояние определяет, какие изменения (удаления, добавления и изменения свойств элементов) должны быть сделаны над моделью в рамках модульной модели, подробнее о концепции состояний см. параграф 2.1. Одна и та же модель может входить в модульную модель несколько раз в различных состояниях. Состояние модульной модели помимо других изменений может определять изменение состояний модулей.

Таблица 3.1 – Основные типы модулей и связей между ними

Графическая нотация	**Описание**	**Множества** $\mathcal{X}, \mathcal{I}, \mathcal{O}, \mathcal{C}$
Типы модулей		
	**Подмодель.** Соответствует математической модели какой-либо биологической подсистемы.	Определяются свойствами модели.
	**Переключатель.** выдает на выход одно из двух значений, поступающих на вход: $out = \begin{cases} in1, & h(t), \\ in2, & \text{иначе}, \end{cases}$ где $h(t)$ – предикат, заданный пользователем.	$\mathcal{X}$ = {in1, in2, out}, $\mathcal{I}$ = {in1, in2}, $\mathcal{O}$ = {out}, $\mathcal{C} = \emptyset$.
	**Константа.** Подает на выход одно константное значение.	$\mathcal{X}$ = {x}, $\mathcal{I} = \emptyset$, $\mathcal{O} = \mathcal{X}$, $\mathcal{C} = \emptyset$.
Порты и связи		
	**Шина**. Элемент, ассоциированный с переменной внешней среды. Может использоваться для установления связей с портами модулей или внешней среды. Несколько шин могут соответствовать одной и той же переменной.	-
	**Порт** (входной, выходной, контактный). Обозначают входные, выходные и контактные переменные модуля. Каждый порт соответствует одной переменной.	Порты соотвествуют элементам множеств $\mathcal{I}$,$\mathcal{O}$,$\mathcal{C}$.
	**Связи.** Направленная связь соединяет выходной порт одного модуля с входным портом другого. Ненаправленная связь – два контактных порта.	-

Шина это специальный элемент в модульной модели, ассоциированный с некоторой переменной модульной модели. Он служит в качестве приватного

интерфейсного порта к внешней среде модели. Тип порта определяется установленными с шиной связями. С одной шиной могут быть установлены связи только одного типа (направленные или ненаправленные), также не может быть установлено несколько входящих направленных связей. Кроме того, несколько шин в модели могут быть ассоциированы с одной и той же переменно. Таким образом, с помощью шин можно связать переменные различных модулей не устанавливая между ними связи напрямую и уменьшая количество пересечений ребер на диаграмме. Пример использования шин приведен на рис. 3.5.

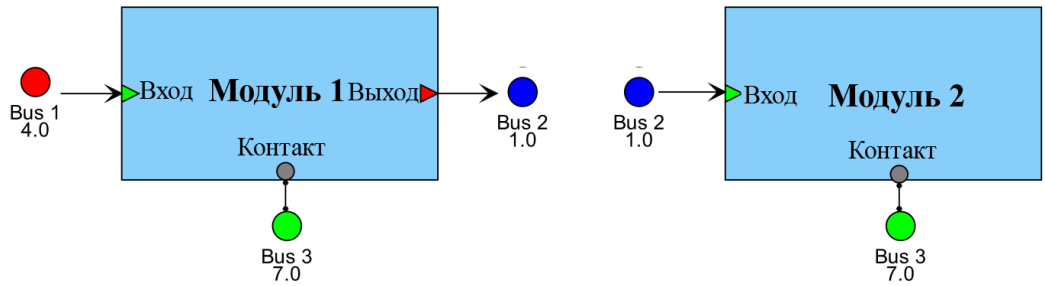


Рисунок 3.5 – Пример использования шин. Несколько шин могут быть ассоциированы с одной и той же переменной, при этом графически они отмечены одним цветом. Порты модулей могут быть соединены дистанционно, сокращая количество пересекающихся ребер

Модульная модель должна быть проинтерпретирована в численную модель для проведения расчетов. Это может быть сделано несколькими различными способами. В данной работе мы рассмотрим два метода.

1. Преобразование по определенным правилам модульной модели к не-модульной, для которой процедура создания численной модели уже реализована (т.н. генерация “плоской” модели [Randhawa et al., 2010]).
2. Алгоритм численных расчетов на основе принципов агентного моделирования (т.н. ко-симуляция [Hernandez et al., 2009]).

Первый подход накладывает ограничения на возможные типы модулей, допустимы только такие, которые могут быть преобразованы к единому математическому формализму. Диаграммы типа “Модульная модель” могут быть преобразованы к формализму “ОДУ с мгновенными событиями”, подробнее алгоритм описан в параграфе 3.3. Диаграммы “Агентная модель” содержат

дополнительные модули и для них применяется второй подход (см. параграф 3.5.).

Эти два подхода также могут использоваться в комбинации – часть модулей объединяются в один модуль, далее с преобразованной агентной моделью проводятся численные расчеты в соответствии со вторым подходом.

3.3. Алгоритм генерации плоской модели

В данном параграфе опишем алгоритм трансформации модульной модели в “плоскую” модель – ОДУ модель с мгновенными событиями, пригодную для численных расчетов с помощью стандартных методов. Входом алгоритма является модульная модель одного из следующих типов:

1) SBML модель (в SBGN или BioUML нотации);
2) модель, содержащая ОДУ с мгновенными событиями, созданная в BioUML;
3) модель, описывающая биологические пути, созданная в BioUML;
4) модульная модель, чьи модули удовлетворяют условиям 1-4.

Также допустимы технические модули – переключатель, константа, график. Исключение составляют “усреднитель” и “скрипт” – так как они не могут быть трансформированы таким образом, чтобы быть включенным в ОДУ-формализм.

Результатом работы алгоритма является математическая модель системы, выполненная в том же формализме и пригодная для численных расчетов в системе BioUML. Алгоритм может применяться в двух случаях:

1) автоматически, когда пользователь запускает численные расчеты модульной модели. Результат алгоритма передается выбранному численному решателю, который генерирует результат. Плоская модель при этом не сохраняется в базу данных и не демонстрируется пользователю;
2) в случае, когда пользователь хочет трансформировать модульную модель в “плоскую” и сохранить результат для дальнейшей работы. В этом случае результирующая модель сохраняется в базу данных.

Алгоритм применяет предварительный обработчик модульных моделей. Схема визуального моделирования в случае модульных диаграмм представлена на рис. 3.6.

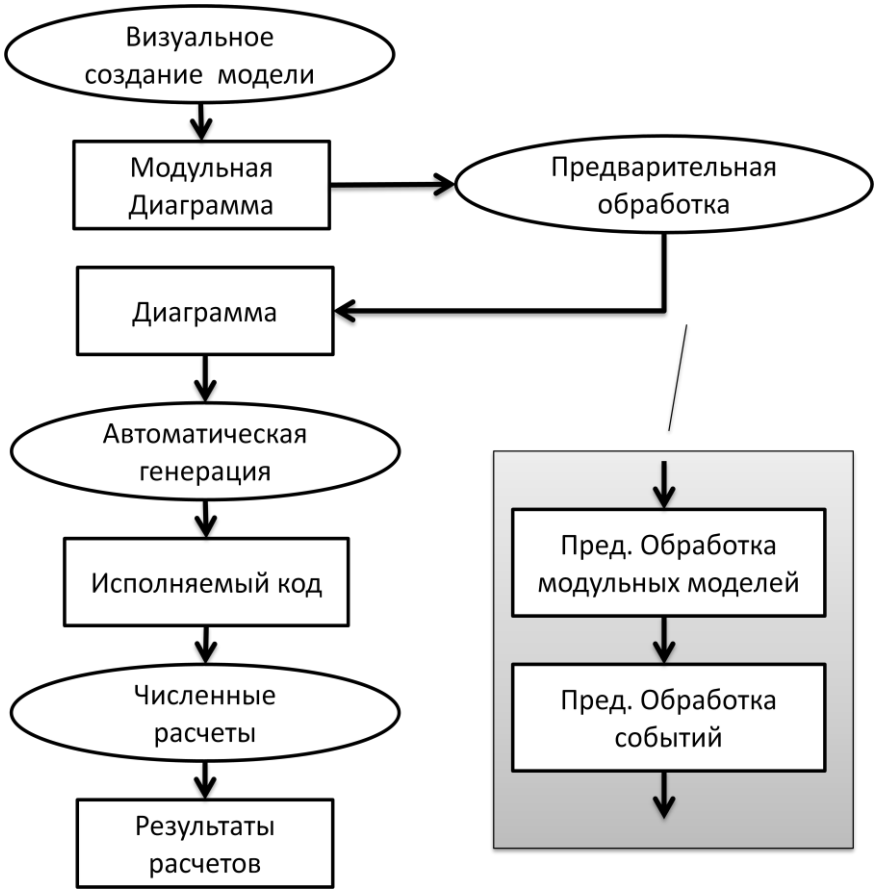


Рисунок 3.6 – Схема визуального создания модульных диаграмм в BioUML

3.3.1. Описание алгоритма

Пусть есть модульная модель

$$MM = (M, DC, UC, E), \qquad M = \{M_1, \dots, M_N\}.$$

$$M_i = (\mathcal{X}_i, \mathcal{I}_i, \mathcal{O}_i, \mathcal{C}_i), \qquad i = 1, \dots, N.$$

Обозначим множество всех переменных всех модулей через $\mathcal{X} = \bigcup_{i=1}^{N} \mathcal{X}_i$. Алгоритм выглядит следующим образом:

Шаг 1. Если какой-либо из модулей M_i ссылается на модульную модель, то данный алгоритм применяется рекурсивно к данной модели.

Шаг 2. Каждой переменной каждого модуля устанавливается в соответствие новое имя, уникальное в рамках модульной модели. Таким образом,

мы получаем множество неповторяющихся идентификаторов переменных N_X. При необходимости, в дальнейшем мы можем расширить это множество, если нам понадобятся новые переменные. Обозначим установленное соответствие следующим образом: $N: \mathcal{X} \to N_X$.

Шаг 3. Используя содержащиеся в модульной модели связи, генерируются **правила замены** переменных. Мы построим отображение, которое каждой переменной каждого модуля будет сопоставлять строку, описывающую некоторое математическое выражение. Опишем процедуру построения отображения S. Пусть $x \in \mathcal{X}_i$.

Если для x нет входящей направленной или ненаправленной связи ($\nexists y \in x: x \leftarrow y \vee x \leftrightarrow y$), то положим $S(x) = N(x)$. Более того, если имена переменных в различных модулях не повторяются, то $S(x) = x$.

Пусть теперь $\exists y \in \mathcal{X}: x \leftrightarrow y$. Построим цепочку из ненаправленных связей (возможно замкнутую):

$$x \leftrightarrow y \leftrightarrow \cdots \leftrightarrow z.$$

В такой цепочке выберем (произвольно) одну переменную, называемую **главной.** Обозначим ее $Main(x)$. Очевидно, что $Main(x) = Main(y) = \cdots = Main(z)$. Для всех переменных, участвующих в цепочке положим:

$$S(x) = \cdots = S(z) = S\big(Main(x)\big) = N(Main(x)).$$

Для главной переменной: $Main(x) = x \Rightarrow S(x) = N(x)$.

Последний вариант, когда $\exists y \in \mathcal{X}: x \overset{p}{\leftarrow} y$. Положим $S(x) = p(S(y))$. Покажем, что такое определение корректно. В случае если для y нет других связей или есть ненаправленная связь, то значение $S(y) = N(Main(y))$ уже определено. Если для y есть входящая направленная связь $y \overset{q}{\leftarrow} z$, то будем применять это правило к z:

$$S(x) = p(S(y)) = p\left(q\big(S(z)\big)\right).$$

Этот процесс гарантированно закончится, т.к. количество переменных в модели конечно и семантические правила запрещают существование циклических и

множественных входящих направленных связей. В результате, мы придем к выражению:

$$S(x) = p\left(q\left(\dots r\left(N\left(Main(v)\right)\right)\right)\right) = p\left(q\left(\dots r\left(N(u)\right)\right)\right).$$

Таким образом, мы рассмотрели все возможности и определили отображение, сопоставляющее каждой переменной из множества $\mathcal{X}$ заменяющее его выражение:

$$S(x) = \begin{cases} S\left(Main(x)\right), & \exists y \in \mathcal{X}: x \leftrightarrow y \\ p(S(y)), & \exists\, y \in \mathcal{X}: x \overset{p}{\leftarrow} y \\ N(x), & \textit{иначе} \end{cases}$$

Шаг 4. К модульной модели применяется рекурсивная процедура, осуществляющая обход модулей и копирование их элементов в новую “плоскую” модель, применяя определенные выше правила замены. При этом для каждой переменной x все вхождения ее имени (кроме вхождений в левую часть дифференциальных уравнений и правил присваивания) заменяются на $S(x)$. В результате мы получаем ОДУ модель, уравнения которой описывают взаимодействие между новыми переменными с именами (не повторяющимися) из множества $N(\mathcal{X})$. При этом связанные переменные “склеиваются” в одну. Дополнительно применяется набор специальных правил.

Уравнения. Присваивания (включая начальные присваивания и присваивания при событии) и дифференциальные уравнения, которые определяют динамику переменной x т. е. $S(x) \neq N(x)$, игнорируются. Такие переменные имеют либо входящую направленную связь и, следовательно, их динамика контролируется другой переменной, либо входят в цепочку ненаправленных связей и при этом не выбраны “главными” в своих цепочках.

Дифференциальные уравнения. Если несколько переменных объединены цепочкой ненаправленных связей и для каждой из них есть дифференциальное уравнение:

$$x_1 \leftrightarrow x_2 \leftrightarrow \dots \leftrightarrow x_m, \qquad S(x_i) = x, \qquad \frac{dx_i}{dt} = f_i, \qquad i = \overline{1,m},$$

то в плоскую модель добавляется новое дифференциальное уравнение: $dx/dt = \sum_{i=1}^{m} f_i$.

Таким образом, цепочке связанных переменных соответствует одна новая переменная. Если для некоторых из связанных переменных нет дифференциального уравнения, соответствующее слагаемое в сумме отсутствует. Согласно предыдущему пункту все остальные уравнения для таких переменных игнорируются.

Компартменты. Если два компартмента соединены направленной илиненаправленной связью, то они объединяются в новый компартмент, содержащий все их элементы (рис. 3.7).

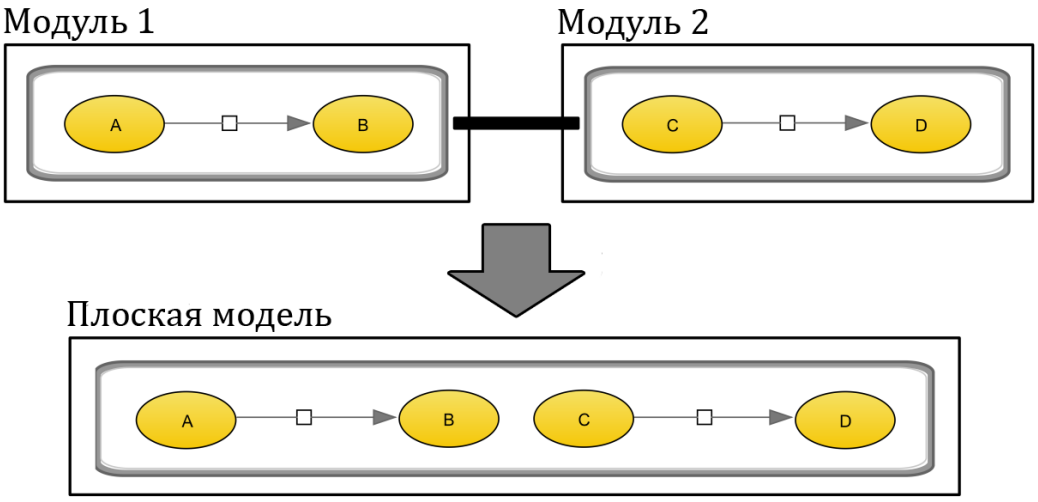


Рисунок 3.7 – Объединение компартментов. Два компартмента объединяются в один. Новый компартмент содержит все элементы исходных компартментов

Сущности и параметры. В математической модели, сущности представлены переменными, описывающими их концентрацию или количество вещества. Их динамика может определяться присваиваниями, событиями, дифференциальными и алгебраическими уравнениями. Также для них могут быть созданы интерфейсные порты и установлены связи в рамках модульной модели. Единственное отличие – сущности могут участвовать в реакциях и имеют специальный элемент на диаграмме и обладают некоторыми дополнительными атрибутами. Таким образом, необходимо специально обрабатывать связи между параметрами и сущностями. Допустим, у нас есть сущность S в одном из

модулей и параметр p в другом. Возможны три варианта установления связи между ними:

1. $p \leftrightarrow S$. В этом случае, переменная, ассоциированная с S, будет выбрана в качестве главной в цепочке и заменит p во всех уравнениях и присваиваниях, где этот параметр встречается.
2. $p \xrightarrow{f} S$. переменная S не будет удалена из диаграммы, вместо этого на сущность S будет наложено граничное условие (boundary condition), указывающее на то, что реакции не могут изменять ее количество, и будет создано новое уравнение $S = f(p)$.
3. $p \xleftarrow{f} S$. Как и в случае с двумя связанными параметрами: параметр p будет замещен переменной ассоциированной, с S, а все уравнения, которые влияют на его динамику, будут удалены (таким образом, его динамика будет целиком определяться динамикой переменной S).

Связанные сущности. Если две сущности связаны ненаправленной связью, они должны рассматриваться как единая сущность в рамках модульной модели. Один из вариантов – создать новую сущность, которая заменит исходные везде, где они встречаются в модели. Визуально, все реакции, в которых участвуют исходные сущности, будут перенаправлены на вновь созданную сущность. В случаях, когда таких реакций достаточно много и графические элементы разнесены на достаточно большое расстояние – результат может представлять собой граф с большим количеством пересечений ребер, с которым трудно работать. Другой подход заключается в использовании маркера клона [Le Novere et al., 2009], который указывает на то, что два объекта на диаграмме соответствуют одной и той же сущности (и ассоциированы с одной и той же математической переменной). Пример приведен на рис. 3.8.

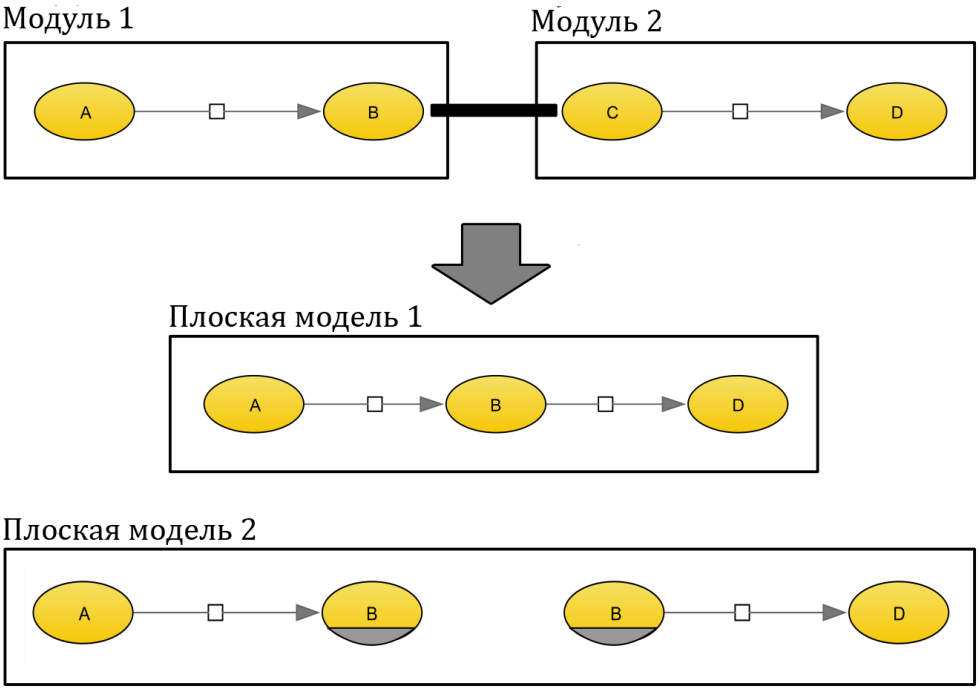


Рисунок 3.8 – Два способа создания плоской модели для веществ, соединенных ненаправленной связью. В первом случае связанные вещества объединены в одно, все реакции в которых они участвовали перенаправлены. Во втором случае используется указатель на то, что два элемента на диаграмме соответствуют одному и тому же веществу [Le Novere et al., 2009]. Математически обе модели эквивалентны

Теперь рассмотрим ситуацию, когда две сущности соединены направленной связью $A \rightarrow B$. Направленная связь подразумевает, что B и A должны быть объединены, причем все что влияет на динамику переменной B в ее модуле – должно быть удалено. Однако если B участвует в некоторых реакциях, влияя на другие вещества, эти реакции должны быть сохранены. В данном случае, сущность B не удаляется из модели. Вместо этого ставится граничное условие на B и добавляется уравнение: $B = A$.

Порты. Все публичные и вынесенные порты модульной модели копируются в плоскую модель. Если порту соответствовала переменная x, новому порту ставится в соответствие переменная $S(x)$. Если $S(x)$ – некоторое математическое выражение, то создается новая переменная и уравнение для нее: $\tilde{x} = S(x)$.

3.3.2. Пример применения алгоритма

Смысл описанной интерпретации направленных и ненаправленных связей можно наглядно представить простым примером. Речь пойдет о химических реакциях, обозначаемых следующим образом:

$$A \overset{k}{\rightarrow} B$$

Эта запись означает, что вещество A превращается в B, k - **константа скорости реакции**. Чтобы не путать реакции с направленными связями, будем обозначать последние (в рамках данного параграфа) следующим образом:

$$(M_1, x_1) \overset{p}{\rightarrow} (M_2, x_2).$$

Эта запись означает, что переменная x_1 модуля M_1 связана направленной связью с переменной x_2 модуля M_2.

Химические реакции описываются системой дифференциальных уравнений [Варфоломеев и Гуревич, 1999]:

$$\begin{cases} \frac{dA}{dt} = -kA, \\ \frac{dB}{dt} = kA, \end{cases}$$

здесь символы A, B обозначают концентрации данных веществ. Пусть теперь у нас есть три модели, каждая из которых описывает одну химическую реакцию:

Модель 1: $A \xrightarrow{k_1} B$. Модель 2: $B \xrightarrow{k_2} C$. Модель 3: $E \xrightarrow{k_3 B} F$. Вещество B присутствует в каждой из трех моделей, играя различные роли: в первой модели это – продукт реакции, во второй – реактант, в третьей – фермент, катализирующий реакцию. Можно предположить, что эти модели описывают различные части одной и той же биологической системы. В соответствии с принципами модульного моделирования построим модель этой системы в виде композиции частных моделей (которые будут выступать модулями). Естественным представляется задать следующие связи:

$$(M_1, B) \leftrightarrow (M_2, B), \qquad (M_1, B) \overset{p}{\rightarrow} (M_3, B),$$

где p – функция преобразования, необходимая, например, если в третьей подмодели используются другие единицы измерения. Преобразование S, соответствующее установленным связям, задается таблицей 3.2.

Таблица 3.2 – Преобразование S, устанавливающее какими математическими выражениями должны быть заменены переменные модулей

(M, x)	(M_1, A)	(M_1, B)	(M_2, B)	(M_2, C)	(M_3, E)	(M_3, F)	(M_3, B)
$N(x)$	A	M_1_B	M_2_B	C	E	F	M_3_B
$S(x)$	A	M_1_B	M_1_B	C	E	F	$p(M_1\_B)$

Применяя преобразование S к модульной модели, получим следующий набор реакций: $A \xrightarrow{k_1} M_1\_B \xrightarrow{k_2} C,\ E \xrightarrow{k_3 p(M_1\_B)} F$. Соответствующая система ОДУ:

$$\frac{d}{dt}\begin{pmatrix} A \\ M_1\_B \\ C \\ E \\ F \end{pmatrix} = \begin{pmatrix} -k_1 A \\ k_1 A - k_2 M_1\_B \\ k_2 M_1\_B \\ -k_3 p(M_1\_B) E \\ k_3 p(M_1\_B) E \end{pmatrix}$$

3.4. Иерархические SBML модели

В 2013 году была опубликована спецификация расширения для языка описания математических моделей SBML [Smith et al., 2013]. Расширение позволяет создавать иерархические SBML-модели, т.е. такие которые содержат ссылки на другие SBML-модели. Спецификация также включает в себя правила, по которым должна генерироваться “плоская” (т.е. не иерархическая) SBML модель. А также авторами спецификации разработан набор тестов, для проверки корректности работы программных продуктов, поддерживающих импорт и численные расчеты иерархических SBML-моделей.

В данном разделе опишем алгоритмы преобразования между модульными моделями, разработанными в данной работе и иерархическими SBML-моделями, описанными в [Smith et al., 2013]. Данная задача является актуальной, поскольку ее решение позволит конвертировать созданные в BioUML модульные модели в общепринятый стандарт, воспроизводить полученные с их помощью результаты и работать с ними в других системах моделирования, поддерживающих формат

SBML (например, iBioSim [Myers et al., 2009]). И наоборот – импортировать в BioUML существующие модульные модели.

Модели SBML базового типа могут быть представлены в BioUML в виде диаграммы с использованием SBGN или BioUML-нотации. Для представления иерархических SBML-моделей был создан дополнительный тип диаграмм “Модульная SBML-модель” совмещающий нотацию SBGN (см. таблицу 2.1) и функционал диаграмм типа “Модульная модель“ (см. таблицу 3.1). Отношения между SBML-моделями и диаграммами BioUML представлены на рис. 3.9.

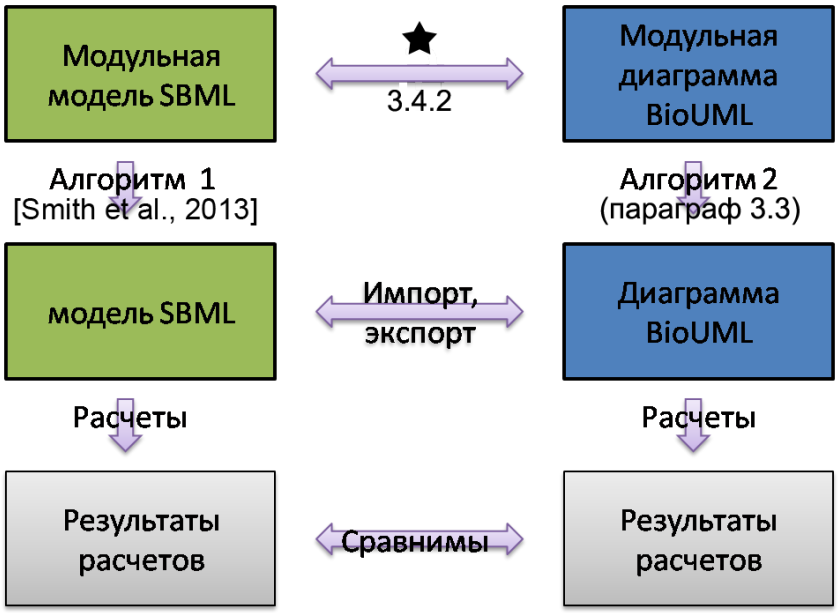


Рисунок 3.9 – Отношения между SBML-моделями и BioUML-диаграммами. Звездочкой отмечены алгоритмы трансформации, разработанные в данной работе и описанные в пункте 3.4.2

Задача заключается в разработке алгоритмов трансформации модульных моделей в иерархические SBML-моделей и обратно таким образом, чтобы следующие модели были математически эквивалентны в смысле совпадения результатов численных расчетов:

- SBML-модель, созданные по правилам, описанным в [Smith et al., 2013] на основе иерархической SBML-модели.
- модель, созданная с помощью алгоритма, описанного в параграфе 3.3, на вход которому подается импортированная версия той же самой иерархической SBML-модели.

3.4.1. Описание расширения SBML

SBML описывает математические модели в терминах **классов** и **объектов**. Класс представляет собой тип данных с набором атрибутов. Например, компартмент – это тип данных, имеющий такие атрибуты, как имя, начальное значение, единицы измерения, размерность и т.д. Конкретный компартмент, имеющий свои собственные значения всех атрибутов, является объектом класса компартмент. Отдельные классы могут расширять другие классы, являясь **подклассами**. Подкласс имеет все атрибуты базового класса и расширяет его новыми. Конкретная модель, созданная с помощью SBML, представляет собой набор объектов. Расширение языка SBML для иерархических моделей позволяет SBML-моделям ссылаться внутри себя на другие SBML-модели. Для этого в SBML были добавлены новые классы, а также модифицированы старые.

Иерархическая SBML-модель может содержать все те же элементы что и обычная SBML-модель, а также элементы, ссылающиеся на другие SBML-модели (подмодели). При этом говорят, что элементы иерархической модели, не принадлежащие подмоделям, находятся на **верхнем уровне**. Верхний уровень соответствует внешней среде, определенной в параграфе 3.1. Полный список измененных и добавленных в этом расширении элементов приведен в таблице 3.3. Объединение подмоделей происходит путем замены элементом верхнего уровня элемента подмодели такого же класса или наоборот. Согласно спецификации [Smith et al., 20013] возможный алгоритм генерации плоской модели выглядит следующим образом:

1. Если подмодель содержит модули, применить к ней данный алогритм.
2. Пусть “M” – идентификатор выбранной подмодели. Положить P = “M__”. До тех пор пока в этой подмодели есть объекты с идентификаторами, начинающимися с последовательности символов P, полагать P = P + “_” (здесь + это конкатенация строк). Этот процесс закончится, так как количество элементов в подмодели конечно.
3. Удалить все элементы подмодели указанные в ReplacedElement или Deletion.

4. У всех оставшихся элементов заменить идентификаторы, добавляя к ним префикс “P”.
5. Преобразовать все ссылки на элементы: если элемент указан в ReplacedElement или Deletion, заменить ссылкой на новый элемент, иначе добавить префикс “P”.
6. Добавить все элементы всех подмоделей в новую модель.

Таблица 3.3 – Расширенные и добавленные новые классы в пакете SBML для иерархических моделей [Smith et al., 2013]

Название	Описание
Базовые классы, существовавшие ранее	
SBase	Базовый класс, от которого наследуются практически все остальные классы. Дополнен списком объектов ReplacedElement, указывающим на объекты которые должны быть заменены данным SBase, и объектом ReplacedBy, указывающим на объект который должен заменить данный SBase.
SBML	Класс, содержащий модель, а также метаинформацию, расширен списком объектов ModelDefinition, позволяющим определять в одном документе несколько моделей, а также списком объектов ExternalModelDefinition, позволяющим устанавливать ссылки на модели, определенные в других SBML-документах
Model	Основной класс, описывающий SBML-модель, расширен путем добавления списка интерфейсных портов (Port) и подмоделей **(**SubModel). Для добавления другой модели в качестве подмодели, для нее должен быть создан объект ModelDefinition или ExternalModelDefinition.
Классы для создания подмоделей	
ModelDefinition	Полностью аналогичен классу Model, за исключением того что в одной SBML-модели можно объявлять несколько объектов типа ModelDefinition.
ExternalModelDefinition	Указывает (с помощью URI) на SBML-модель, определенную во внешнем документе.
SubModel	Содержит ссылку на ModelDefinition или ExternalModelDefinition, определенные в том же документе. Указанная модель используется как подмодель в рамках данной иерархической модели. SubModel может определять список удалений (Deletion) элементов указанной модели. Эти объекты должны быть удалены из модели перед ее включением в иерархическую модель.

Окончание таблицы 3.3

Классы для ссылок на объекты модели	
SBaseRef	Класс, использующийся для ссылок на объекты SBase. Как правило объекты этого класса содержатся внутри SBase и ссылается на другой объект SBase. Класс имеет четыре атрибута для ссылок, одновременно может использоваться только один из них: - idRef может указывать на идентификатор произвольного объекта типа SBase; - portRef может указывать на идентификатор порта модели; - unitRef может указывать на объекты типа UnitDefinition; - metaIdRef может указывать на атрибут metaId объекта типа SBase. SBaseRef также может использоваться рекурсивно, так как может содержать вложенный объект SBaseRef. В этом случае внешний объект может ссылаться на подмодель, а внутренний – на конкретный объект подмодели. Все классы ниже наследуются от SBaseRef.
ReplacedElement	Содержащий его объект SBase должен заменить указываемый в ReplacedElement объект. Т. е. все ссылки на заменяемый объект должны быть заменены ссылками на заменяющий. ReplacedElement содержит следующие дополнительные атрибуты submodelRef – ссылка на подмодель, элемент которой является объектом замены и conversionFactor – множитель, служащий для согласования единиц измерения заменяемого и заменяющего объектов.
Port	Используется в качестве интерфейсного элемента для установления связей между моделью и ее подмоделями. Ссылка, на порт ссылающаяся на некоторый объект имеет тот же смысл что и ссылка на данный объект. Порт имеет следующие ограничения: - порт не может ссылаться на другие порты, т. е. использовать атрибут portRef; - порт не может ссылаться на объект, на которой уже ссылается другой порт, который удален с помощью Deletion или заменен с помощью ReplacedElement.
ReplacedBy	Содержащий его объект SBase должен быть заменен указываемым объектом. Все ссылки на него должны быть заменены ссылками на заменяющий объект. Дополнительно определяет атрибут submodelRef, указывающий на подмодель, содержащую замену.
Deletion	Указываемый объект должен быть удален из содержащей его модели перед включением данной модели в другую модель.

3.4.2. Алгоритмы импорта и экспорта иерархических SBML-моделей

Подход, примененный для описания иерархических SBML-моделей принципиально отличается от описываемого в данной работе способом интеграции модулей, однако между ними все же можно установить соответствие.

Заметим, что как следует из алгоритма, предварительно все подмодели, которые сами содержат подмодели, должны быть трансформированы в плоские модели. Следовательно, не может существовать замен элементов добавленных в иерархическом пакете: не может быть замен других замен, замен портов, замен подмоделей и т.д.

Согласно алгоритму в спецификации, замена объекта равносильна удалению старого объекта и изменению всех мест, где он упоминается. В Таблице 3.4. представлены зависимости между элементами SBML, для каждого элемента указано какие элементы ссылаются на него и, следовательно, должны быть обновлены при его замене. Как видно из таблицы зависимостей, замена реакций, уравнения и событий эквивалентна удалению старых элементов и добавлению новых, таким образом, они могут быть описаны в рамках модульной BioUML-модели как удаления старых элементов в специальном состоянии модуля. Замены объектов соответствующих математическим переменным (species, parameter и compartment) могут быть сопоставлены связям между соответствующими переменными. Элемент Function отличается от остальных – он не ассоциирован с переменной и его замена должна сопровождаться изменением всех мест, где функция вызывается.

Теперь мы можем описать алгоритмы трансформации модульной диаграммы в формат SBML импорта (таблица 3.5) и наоборот построение модульной диаграммы по SBML-модели (таблица 3.6).

Таблица 3.4 – Зависимости между элементами SBML

Заменяемый элемент	**Элементы, которые должны быть обновлены при замене**
Species	Reaction, Rule, Event
Compartment	Species, Reaction
Parameter	Species, Reaction, Rule, Event, ReplacedElement
Reaction	-
Rule (rate, algebraic, scalar)	-
Event	-
Function	Reaction, Rule, Event, Function

Таблица 3.5 – Создание иерархической SBML-модели по модульной диаграммы

Элемент диаграммы BioUML	**Действия при экспорте диаграммы в формат SBML**
Модуль	Транслируется в элемент Submodel, ассоциированная с ним диаграмма транслируется в объект ModelDefinition. Если модуль определяет состояние ассоциированной с ним модели, то все удаления, сделанные в данном состоянии, транслируются в элементы deletion. Если в данном состоянии элемент модели изменен или добавлены новые, то такую ситуацию невозможно описать в рамках композитной SBML-модели. В этом случае создается новая копия модели с произведенными изменениями, согласно установленному состоянию.
Порт	Публичный порт преобразуется в SBM-порт, остальные типы портов игнорируются. Вынесенный порт соответствует рекурсивной структуре SBaseRef (см. рис. 3.10).
Ненаправленная связь	Связь, соединяющая переменную внешней среды с переменной модуля, трансформируется в ReplacementElement или ReplacedBy в зависимости от установленной переменной. Если ненаправленная связь соединяет переменные двух модулей, то во внешней среде создается новая переменная с двумя элементами ReplacementElement. Таким образом, эта переменная заменяет обе соединенные ненаправленной связью переменные модулей.
Направленная связь	Транслируется в один или три элемента замены аналогично ненаправленной связи, однако при этом в соответствующей SBML-подмодели создаются элементы deletion, удаляющие все уравнения, влияющие на входную переменную. Если входом направленной связи является внешняя среда, то уравнения, влияющие на входную переменную, удаляются из SBML-модели (для верхнего уровня в SBML невозможно определить deletion). Дополнительно, если направленная связь определяет трансформирующую функцию, то возможны два варианта. 1. Функция имеет вид $f(x) = cx$, где c – некоторое число. В этом случае в элемент замены устанавливается свойство conversionFactor = “c”. 2. Функция имеет произвольный вид. В этом случае в подмодели, содержащей x добавляется новая переменная $\tilde{x}$ и уравнение $\tilde{x} = f(x)$. Направленная связь $x \xrightarrow{f} y$ заменяется связью $\tilde{x} \to y$.
Дополнительные модули – константа, переключатель.	Преобразуются в SBML-модели, содержащие эквивалентные математические формулы.
Модуль график	Игнорируется, так как не влияет на математический смысл модели и служит только для демонстрации результатов расчетов.
Шина	Транслируется в переменную модели верхнего уровня с соответствующими заменами.

Таблица 3.6 – Создание модульной диаграммы по иерархической SBML-модели

Название класса SBML	**Действие при импорте в BioUML**
ExternalModelDefinition	Модель по внешней ссылке импортируется как отдельная диаграмма в BioUML.
ModelDefinition	Импортируется как отдельная диаграмма в BioUML
Port	Если порт соответствует parameter, species или compartment, он транслируется в порт на диаграмме. В противном случае он игнорируется.
Submodel	Транслируется в подмодель на модульной диаграмме.
ReplacedElement, ReplacedBy	Если элемент замены ссылается на parameter, species или compartment, то он транслируется в связь между соответствующими переменными. При этом на верхнем уровне создается приватный порт. В случае, когда элемент замены ссылается на порт, аннотированный как вход или выход, создается направленная связь, если порт не аннотирован – ненаправленная. Если элемент замены не соответствует переменной, то заменяемый элемент помечается как удаленный в текущем состоянии модуля. Рекурсивной структуре элементов замены соответствует цепочка вынесенных портов (см. рис. 3.10).
Deletion	Элемент, на который ссылается deletion, помечается как удаленный в текущем состоянии модуля.

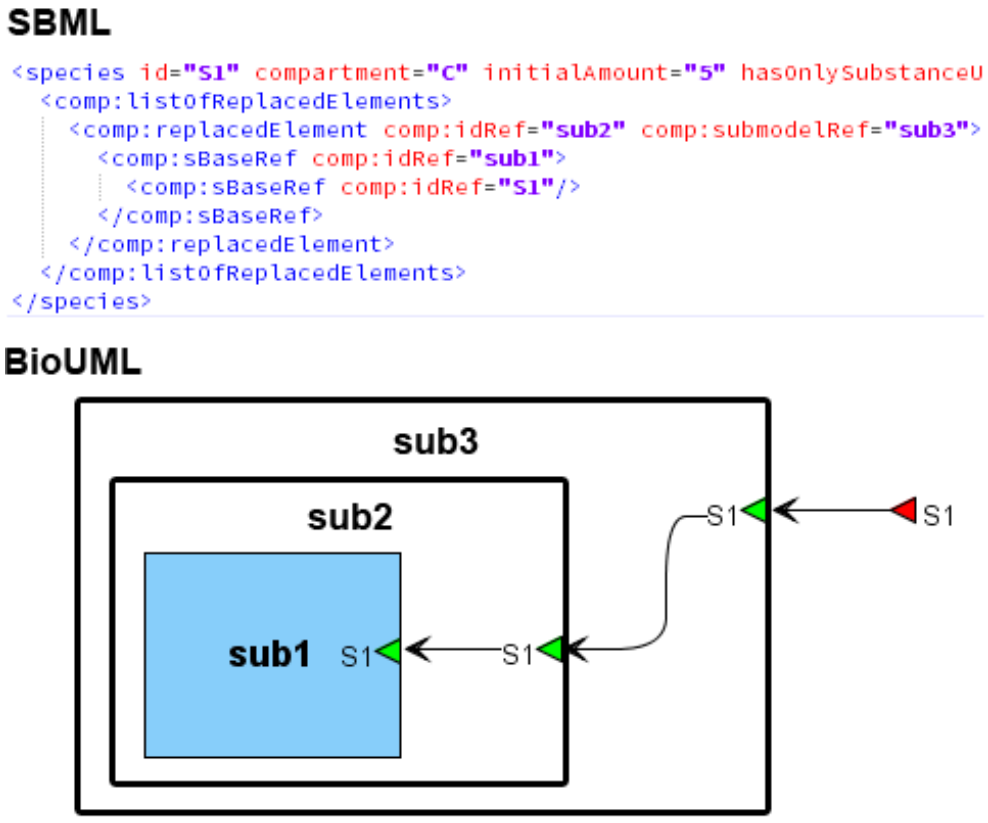


Рис. 3.10 – Рекурсивная замена. S1 из модели верхнего уровня (внешней среды) заменяет собой S1 из подмодели sub1, находящейся в подмодели sub2 которая находится в подмодели sub3.

3.4.3. Реализация алгоритмов импорта и экспорта, тестирование

Алгоритмы преобразования реализованы в платформе BioUML. Пользователь может:

- создать и редактировать диаграмму типа “Иерархическая SBML-модель”;
- импортировать иерархическую SBML-модель, при этом она будет автоматически преобразована в диаграмму данного типа;
- экспортировать диаграмму типа “Иерархическая SBML-модель”, при этом она будет трансформирована в иерархическую SBML-модель.

Для тестирования описанного алгоритма использовался набор тестов (http://sbml.org/Facilities/Database), разработанный авторами SBML для тестирования поддержки расширения для создания иерархических моделей. Набор выпущен 17 мая 2013 года, состоит из 60 тестов, каждый из которых содержит иерархическую SBML-модель, параметры для численных расчетов и ожидаемые результаты расчетов. Все тесты были успешно пройдены, результаты доступны по адресам http://www.biouml.org/sbml_tests/ и http://sbml.org/Facilities/Database/Simulator. Модульные модели, представляющие собой импортированные версии тестовых моделей, доступны в web-версии BioUML.

Отметим одно из различий между модульными BioUML и SBML моделями, касающееся связанных ненаправленной связью переменных, определяемых дифференциальными уравнениями:

$$\frac{dx}{dt} = f(x,t), \qquad x \leftrightarrow y, \qquad \frac{dy}{dt} = g(y,t)$$

Плоская модель с помощью алогоритма (3.3):

$$\frac{dx}{dt} = f(x,t) + g(y,t)$$

Плоская модель по правилам [Smith et al., 2013] (не корректна):

$$\frac{dx}{dt} = f(x,t), \qquad \frac{dx}{dt} = g(x,t)$$

С другой стороны есть случаи, когда заменяемый объект не может быть помечен как удаленный в BioUML, в частности формальные правила не запрещают заменять списки элементов целиком (listOfSpecies, listOfReactions и т.д.). Эта ситуация может моделироваться набором удалений по одному для каждого элемента списка. Также на данный момент не поддерживается замена функций.

Таким разработанный формализм модульных моделей позволяет описывать иерархические SBML-модели практически без потери информации, но в то же время позволяет описывать модели более широкого класса. В следующем параграфе опишем тип диаграмм, расширяющий тип “Модульная модель” и добавляющий новые объекты, которые не могут быть представлены в SBML.

3.5. Агентное моделирование

В данной работе принципы агентного моделирования используются для создания алгоритма численных расчетов для модульной модели в случае, когда алгоритм генерации плоской модели не может быть применен. Модульная модель может быть проинтерпретирована как агентная, при этом каждому модулю сопоставляется отдельный агент, связи в модульной модели определяют взаимодействия между агентами.

В описанных выше терминах наши агенты являются адаптивными, так как изменяют свое внутреннее состояние под воздействием внешней среды и других агентов. В рамках данного подхода мы будем рассматривать наиболее простую схему, не учитывая возможность агентов перемещаться в пространстве, динамически рождаться или умирать при выполнении некоторого условия. Эти возможности могут быть добавлены в будущем, в частности в соответствии со спецификацией расширения SBML “dyn”.

3.5.1. Основные определения

Мы будем оперировать агентами, построенными на основе модулей, определенных в параграфе 3.1. Агент определим следующим образом:

$$A = (\mathcal{X}, \mathcal{I}, \mathcal{O}, \mathcal{C}, X^0, T, F),$$

где:

$\mathcal{X} = \{x_1, \dots, x_n\}$ – множество переменных агента;

$\mathcal{I} \subseteq \mathcal{X}$ – множество **входных (input)** переменных агента, т. е. переменные, значения которых определяются извне агента;

$\mathcal{O} \subseteq \mathcal{X}$ – множество **выходных (output)** переменных агента, т. е. переменные, чьи значения рассчитываются внутри агента и могут быть переданы наружу;

$\mathcal{C} \subseteq \mathcal{X}$ – множество **контактных (contact)** переменных агента, т. е. переменных, значения которых могут изменяться как снаружи агента, так и внутри;

$X^0 = (x_1(t^0) \quad \cdots \quad x_n(t^0) \in \mathbb{R}^n)$ – **начальное состояние** агента, определенное численными значениями всех его переменных;

$T = (t^0, \dots, t^k)$ – **временная сетка** агента, $t^j \in \mathbb{R}_+, \quad j = \overline{0,k}$;

F – **функция шага** агента, определяющая состояние агента в следующий момент времени через состояния в предыдущие моменты времени:

$$X^{j+1} = X\left(t^{j+1}\right) = \begin{pmatrix} x_1(t^{j+1}) \\ \vdots \\ x_n(t^{j+1}) \end{pmatrix} = F\left(X^0, \dots, X^j, t^{j+1}\right) = \begin{pmatrix} f_1\left(X^0, \dots, X^j, t^{j+1}\right) \\ \vdots \\ f_n\left(X^0, \dots, X^j, t^{j+1}\right) \end{pmatrix},$$

здесь $X(t) \in \mathbb{R}^N$ – **состояние** агента в момент времени t.

Шаг агента заключается в переходе от j-й временной точки к j+1-й. При этом новое состояние определяется через предыдущие с помощью функции шага.

Симуляция агента состоит из последовательного порождения состояний, начиная с начального: $X^0, X^1, \dots, X^k$.

Будем предполагать, что состояние любого агента, как минимум, включает одну переменную t – **внутреннее время**, которое может принимать значения только из временной сетки: $t\left(t^j\right) = t^j, \ j = \overline{0,k}$.

Наиболее простой агент выглядит следующим образом.

$$A = (\{\mathrm{t}\}, \emptyset, \emptyset, \emptyset, (t^0), (t^0, t^1), F)$$

Состояние этого агента задается только его внутренним временем. В этом случае шаг заключается в присвоении нового значения внутреннему времени: $X^1 = F((t^0), t^1) = (t^1)$. В дальнейшем мы, в основном, будем иметь дело с агентами, у которых следующее состояние зависит только от предыдущего:

$$X^{j+1} = F\left(X^j, t^{j+1}\right), \qquad F: \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}^n.$$

Связь (направленная, ненаправленная) между агентами – это связь между их переменными, определяется аналогично связи между модулями.

Таким образом, агент отличается от модуля способностью функционировать во времени, в то время как модуль – статический объект. Аналогично модульной модели можно определить и **агентную модель**:

$$AM = (A, DC, UC, E, T),$$

где:

$A = \{A_1, \dots, A_N\}$ – множество агентов модели,

DC – множество направленных связей между агентами,

UC – множество ненаправленных связей между агентами,

E – внешняя среда, определяется аналогично внешней среде для модульной модели.

$T = (t^0, \dots, t^M)$ – временная сетка модели.

Состояние агентной модели определяется как вектор, составленный из состояний всех входящих в нее агентов:

$$X = (x_{11}, \dots, x_{1n_1}, \dots, x_{N1}, \dots, x_{Nn_N})^T,$$

где $X_i = \left(x_{i,1}, \dots, x_{i,n_i}\right)^T$ - состояние i-го агента, $i = 1, N$. Состояния агентов в следующий момент времени определяется через состояния в текущий момент времени, поэтому это верно и для состояния самой агентной модели. $X^{k+1} = F(X^k, t^{k+1})$. Однако в этом случае на состояния агентов влияют также связи, установленные в модели. В следующем параграфе опишем конкретный вид функции шага F в зависимости от установленных связей. Таким образом, агентную модель можно представить в виде агента

$$AM = \left(\bigcup_{\mathrm{i}=1}^{\mathrm{n}} \mathcal{X}_\mathrm{i}, \mathcal{I}_{public}, \mathcal{O}_{public}, \mathcal{C}_{public}, X^0, F_{AM}, T_{AM}\right),$$

где $\mathcal{I}_{public}, \mathcal{O}_{public}, \mathcal{C}_{public}$ - публичный интерфейс внешней среды. Таким образом, можно создавать **иерархические агентные модели**, включающие другие агентные модели.

Модульная модель может быть трансформирована в агентную, для этого каждый модуль должен быть преобразован в агент, что может быть сделано добавлением функции шага и сетки времени. Начальное состояние автоматически получается из значений всех переменных модуля. В частности, если модуль представляет собой математическую модель, функция шага представляет собой шаг численного решателя:

$$\text{МОДУЛЬ}(\mathcal{X},\mathcal{I},\mathcal{O},\mathcal{C}) + \text{РЕШАТЕЛЬ}(\text{F}) + \text{СЕТКА}(\text{T}) = \text{АГЕНТ}(\mathcal{X},\mathcal{I},\mathcal{O},\mathcal{C},X^0,\text{F},\text{T}).$$

3.5.2. Диаграммы типа “Агентная модель”

Для численных расчетов агентных моделей был реализован новый инструмент численного решения – AgentModelSimulationEngine, который создает численную агентную модель на основе визуального представления, генерирует исполняемый код для агентов и запускает численные расчеты. Таким образом, в диаграмму взаимодействия классов и компонент BioUML, добавляется новый уровень, соответствующий формализму агентной модели (см. рис. 3.11).

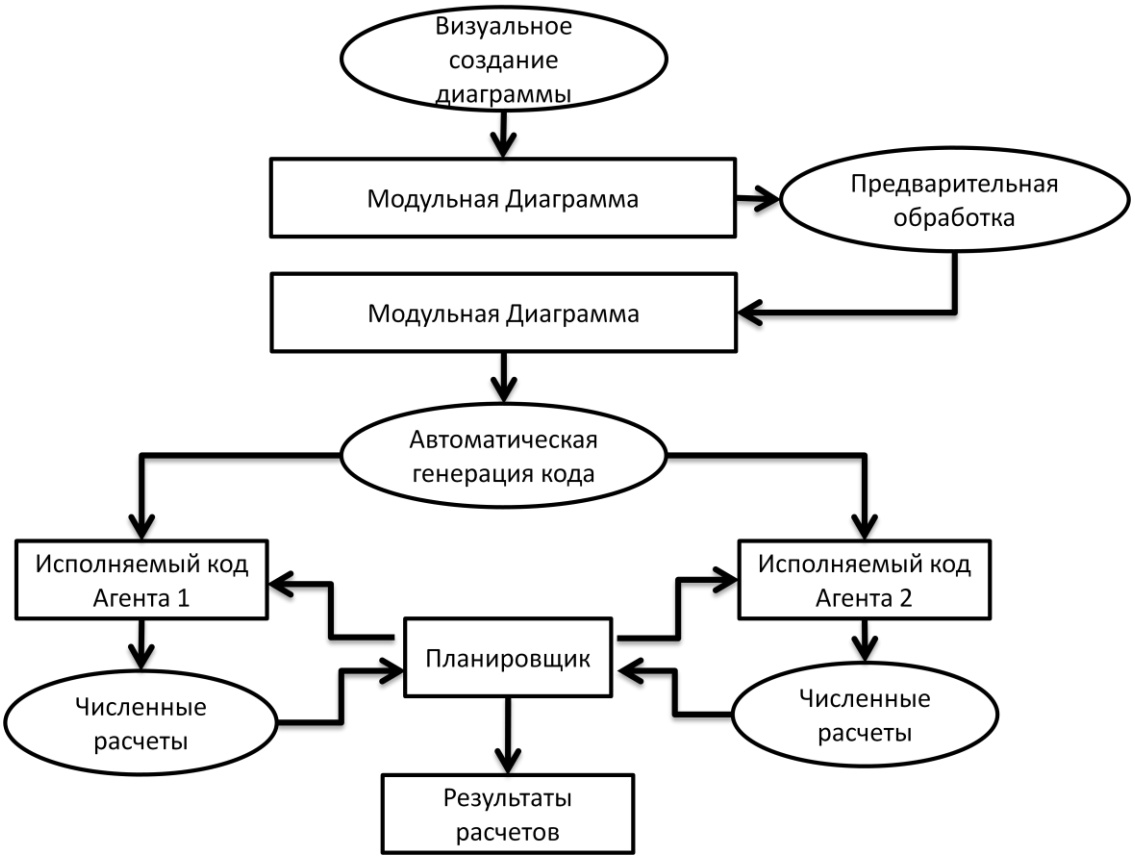


Рисунок 3.11 – Схема агентного визуального моделирования в BioUML

С помощью данного инструмента любая диаграмма, формально описываемая как модульная модель, может быть интерпретирована как агентная модель. В частности это диаграммы типов “Модульная модель” и “Модульная SBML-модель”.

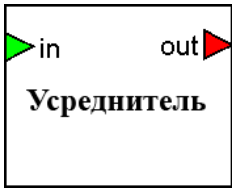
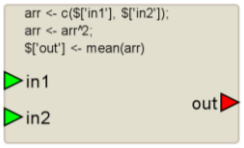
Для этого в интерфейсе BioUML пользователь для каждой подмодели определяет соответствующий численный решатель и сетку времени. Например, если подмодель содержит модель физиологического процесса, то пользователь может сопоставить ему инструмент численного решателя ОДУ и один из методов, которые выбранный инструмент предлагает (метод Эйлера, Дорманда-Принса, Адамса и т.д.). Шаг работы агента является шагом численных расчетов с помощью установленного решателя.

Для реализации более широких возможностей моделирования, в BioUML был создан новый тип диаграмм “Агентная модель”, который может содержать все элементы диаграмм “Модульная модель” (перечисленные в таблице 3.1) и определяет новые типы модулей (таблица 3.7) и новые допустимый тип подмодели: “Модель артериального дерева”. Эти диаграммы не содержат принципиального ограничения на тип подмоделей, потенциально любой тип диаграмм для которого реализован алгоритм численных расчетов может быть включен в состав диаграммы “Агентная модель”.

Дополнительно каждый модуль определяет атрибут timescale (безразмерный), определяющий соответствие между единицами измерения времени агента и глобальным временем модели. По умолчанию timescale для всех агентов равен 1, т.е. единицы измерения времени во всех агентах совпадают между собой. Если, например, для одного модуля timescale = 1, а для другого – 60, это значит, что 1 шаг второго модуля по модельному времени равен 60 шагам первого.

Агентное моделирование может применяться совместно с генерацией плоской модели, а именно: все модули, использующие формализм ОДУ объединяются в один модуль с помощью алгоритма генерации плоской модели, после чего полученная модульная модель интерпретируется как агентная.

Таблица 3.7 – Элементы диаграммы “Агентная модель”, помимо описанных в таблице 3.1

Графическая нотация	Описание	Множества $\mathcal{X}, \mathcal{I}, \mathcal{O}, \mathcal{C}$
Типы модулей		
	**Подмодель**. Соответствует математической модели какой-либо биологической подсистемы. В отличие от диаграмм “Модульная модель”, могут быть также сто	Определяются свойствами модели.
	**Усреднитель**. Вычисляет скользящее среднее значений, поступающих на вход и передающий его на выход. вычисляет скользящее среднее значений поступающих на вход и подает на выход. Формула расчета $$out(t_k) = \frac{1}{n}\sum_{i=k-n}^{k} in(t_i),$$ где n – количество запоминаемых значений.	$\mathcal{X}$ = {in, out}, $\mathcal{I}$ = {in}, $\mathcal{O}$ = {out}, $\mathcal{C} = \emptyset$.
	**Скрипт.** позволяет выполнять во время расчетов произвольный пользовательский код, написанный на одном из следующих языков: MATLAB, java-script или R. Входные и выходные переменные такого модуля определяются пользователем.	Определяются пользователем
	**График**. Динамически выводит на график результаты численных расчетов.	$\mathcal{X}$ определяется пользователем, $\mathcal{I} = \mathcal{X}$, $\mathcal{O} = \mathcal{C} = \emptyset$.

3.5.3. Алгоритм численных расчетов

Так же, как и любой другой инструмент численного решения, AgentModelSimulationEngine предоставляет численный решатель, пригодный для проведения численных расчетов модели. В случае агентной модели, решателем выступает **планировщик** (Scheduler). Планировщик синхронизирует работу агентов, обеспечивает корректное взаимодействие между ними в соответствии с установленными связями и обеспечивает выдачу рассчитанных значений.

Шаги агентов происходят по очереди, т. е. на каждом шаге вычислений выбирается один агент, который совершает шаг, после чего возвращает управление планировщику. Для каждой установленной в модели связи, планировщик предоставляет **канал связи** (Link), который хранит информацию о том, какие два агенты связаны, между какими конкретно переменными

установлена связь, а также тип установленной связи. По каналу связи передаются **сообщения (**Message**)**, содержащее информацию о том, как было изменено значение переменной во время некоторого шага агента:

$$Message^k(x) = (t^k, x^k, t^{k+1}, x^{k+1}), \qquad k = 0,1,2, \ldots,$$

где:

x – переменная агента-отправителя, которой соответствует данное сообщение,

t^k – момент времени, в который агент начал делать шаг,

x^k – значение переменной x в момент времени t^k,

t^{k+1}– время, на котором агент закончил свой шаг,

x^{k+1} – значение переменной x в момент времени t^{k+1}.

Получив такое сообщение, планировщик хранит его до тех пор, пока оно не будет затребовано агентом-получателем. Сообщения передаются агенту-получателю перед началом каждого его шага. Пусть это моменты времени $s^k, k = 0,1,2, \ldots$. Если связь направленная (с преобразующей функцией p), то планировщик должен вычислить значение переменной x в запрошенный момент времени (с помощью интерполяции), вычислить значение преобразующей функции и установить его соответствующей переменной агента-получателя (пусть это будет y):

$$y(s^k) = p\left(x(s^k)\right).$$

Если же связь ненаправленная, то агент запрашивает приращение переменной за период времени $s^k - s^{k-1}$, значение этого приращения прибавляется к текущему значению переменой: $y(s^k) = y(s^k) + x(s^k) - x(s^{k-1})$. В обоих случаях значения $\mathrm{x}(\mathrm{s}^{\mathrm{k}})$, $x(s^{k-1})$ вычисляются с помощью интерполяции по двум ближайшим точкам: $\mathrm{t}^{\mathrm{i}} \le \mathrm{s}^{\mathrm{k}} \le \mathrm{t}^{\mathrm{i}+1}$, $\mathrm{t}^{\mathrm{j}} \le \mathrm{s}^{\mathrm{k}-1} \le \mathrm{t}^{\mathrm{j}+1}$.

Для того чтобы алгоритм работал корренто, необходимо следить, чтобы сообщения посылались планировщику раньше, чем запрашивались. Так как агенты запрашивают сообщения перед началом своего шага, это равносильно тому, что агент не имеет права делать шаг, пока есть связанные с ним агенты, чье внутренне время меньше чем время данного агента. Это выполняется, если в качестве текущего агента всегда выбирать агента с минимальным текущим

временем. Шаг работы численного решателя (планировщика) выглядит следующим образом.

1. Планировщик выбирает агента с минимальным текущим временем.
2. Агент получает входящие сообщения от планировщика.
3. Выполняется один шаг выбранного агента.
4. Создаются и отправляются планировщику исходящие сообщения агента.
5. Проверяется условие завершения работы агента. Если оно выполнено (агент достиг финального состояния или произошла ошибка), то планировщик больше не может выбрать его для совершения шага.
6. Если все агенты завершили работу, расчеты заканчиваются.

Пример схемы работы модели представлен на рис. 3.12.

Рассмотрим теперь, как меняется состояние агентной модели в процессе симуляции. Пусть модель состоит из одного агента:

$$AM = (\{A_1, \dots, A_N\}, DC, UC, T_{AM}), \qquad T = (t^0, \dots, t^M),$$

$$A_i = (X_i, F_i, T_i), \qquad T_i = \left(t_i^0, \dots, t_i^{m_i}\right), \qquad X_i^k = \begin{pmatrix} x_{i1}^k \\ \vdots \\ x_{in_i}^k \end{pmatrix}, \qquad i = 1, \dots, N.$$

Состояния всех агентов изменяются в соответствии с их функцией шага:

$$X_i^{k+1} = F_i(X_i^k, t^{k+1}), \qquad i = 1, \dots, N.$$

В матричной форме:

$$\begin{pmatrix} x_{i1}^{k+1} \\ \vdots \\ x_{in_i}^{k+1} \end{pmatrix} = \begin{pmatrix} f_{i1}(X_i^k, t^{k+1}) \\ \vdots \\ f_{in_i}(X_i^k, t^{k+1}) \end{pmatrix} = \begin{pmatrix} f_{i1}(x_{i1}^{k+1}, \dots, x_{in_i}^{k+1}, t^{k+1}) \\ \vdots \\ f_{in_i}(x_{i1}^{k+1}, \dots, x_{in_i}^{k+1}, t^{k+1}) \end{pmatrix}, \qquad i = 1, \dots, N.$$

Состояние агентной модели строится как совокупность состояний всех агентов:

$$X^k = \begin{pmatrix} x_{11}^k & \cdots & x_{1l_1}^k & \cdots & x_{N1}^k & \cdots & x_{Nl_N}^k \end{pmatrix}^T = \begin{pmatrix} X_1^k \\ \vdots \\ X_N^k \end{pmatrix}$$

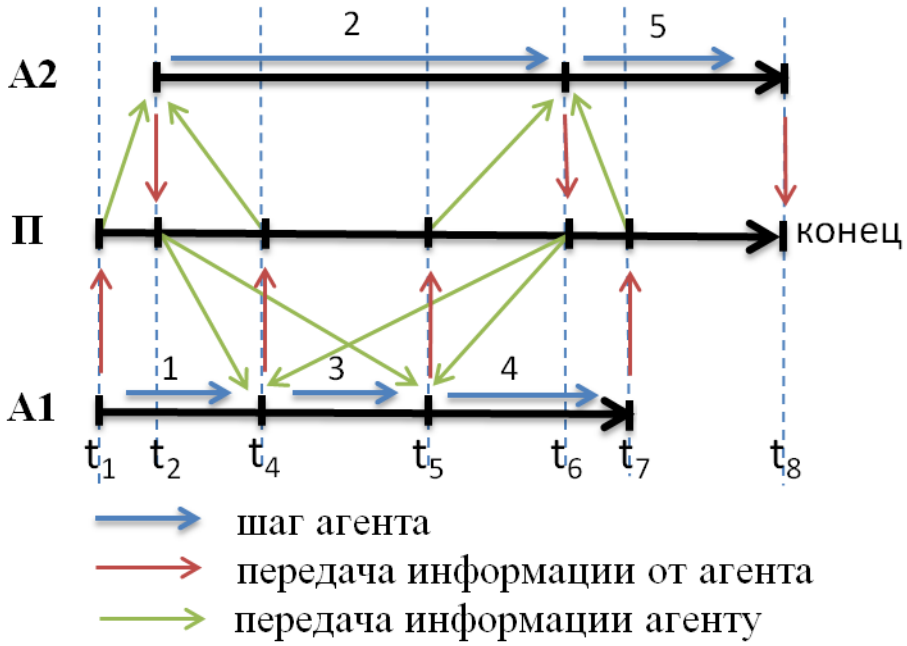


Рисунок 3.12 – Схема последовательности шагов агентов и обмена сообщений между ними. А1, А2 – агенты модели, П – планировщик. Цифрами над стрелками шагов обозначен порядок, в котором делают шаги агенты. Сообщения от агентов создаются перед шагом и обновляются после него (вместе эта информация составляет одно сообщение). В моменты передачи сообщений агентам значения переменных интерполируются по двум соседним точкам

Следовательно, состояние агентной модели в следующий момент времени также будет зависеть от текущего состояния.

$$X^{k+1} = H(X^k, t^{k+1}), \qquad X^{k+1} = \begin{pmatrix} X_1^{k+1} \\ \vdots \\ X_N^{k+1} \end{pmatrix} = \begin{pmatrix} H_1(X_1^k, \dots, X_N^k, t^{k+1}) \\ \vdots \\ H_N(X_1^k, \dots, X_N^k, t^{k+1}) \end{pmatrix}.$$

Функция шага F будет зависеть от того какие связи установлены в модели. Опишем, как она будет выглядеть в разных случаях. При этом гарантируется, что временные сетки T_1 и T_2 содержат все временные точки сетки $T_{\text{АМ}}$ (см. предыдущий параграф). Таким образом, для любой точки временной сетки агентной модели можно получить точное состояние каждого из агентов. Сначала рассмотрим случай одинаковых временных сеток $T_i = T = (t^0, \dots, t^M),\ i = 1, .., N$.

Если связей в модели нет, то расчеты для агентов происходят независимо. Начальное значение определяется начальным состоянием всех агентов по отдельности: $X^0 = (X_1^0, \dots, X_N^0)^T$. На каждом шаге расчетов, агенты делают свои шаги не влияя друг на друга:

$$X_i^{k+1} = F_i(X_i^k, t^{k+1}), \qquad i = 1, \dots, N.$$

Состояние модели складывается из состояний всех агентов:

$$X^{k+1} = \begin{pmatrix} X_1^{k+1} \\ \vdots \\ X_N^{k+1} \end{pmatrix} = H(X^k, t^{k+1}) \equiv \begin{pmatrix} F_i(X_i^k, t^{k+1}) \\ \vdots \\ F_i(X_i^k, t^{k+1}) \end{pmatrix}$$

$$x_{ij}^{k+1} = f_{ij}(X_i^k, t^{k+1})), \qquad i = 1, \dots, N, j = 1, \dots, n_i.$$

Рассмотрим случай, когда есть направленная связь между переменными двух агентов. Пусть эти агенты A_r и A_s, связанные переменные: x_{r,j_r} и x_{s,j_s}. Для простоты обозначим через $x = x_{r,j_r}$ и $y = x_{s,j_s}$, соответствующие компоненты функций шага $f_{r,j_r} \equiv f, f_{s,j_s} \equiv g$. Установленная связь $(A_r, x) \xrightarrow{p} (A_s, y)$ означает, что перед началом каждого шага агента A_s его переменной y будет устанавливаться значений $p(x)$, где x – переменная агента A_r. При этом сам агент A_s не изменяет значение этой переменной: $g(y, t) \equiv y$.

$$y^{k+1} = p(x^{k+1}) = p\left(f(X_r^k, t^{k+1})\right), \qquad y^0 = p(x^0).$$

На остальные переменные эта связь не оказывает влияния:

$$z_{ij}^{k+1} = f_{ij}\left(X_i^k, t^{k+1}\right), \qquad i \neq s, j \neq j_s.$$

$$z_{ij}^{k+1} = \begin{cases} f_{ij}(X_i^k, t^{k+1}) \\ p\left(f(x_{rj_r}^k, t^{k+1})\right), \ i = s, j = j_s \\ f(x_{rj_r}^k, t^{k+1}), \qquad i = r, j = j_r \end{cases} \tag{3.1}$$

Пусть теперь есть ненаправленная связь $(A_r, x) \overset{z}{\leftrightarrow} (A_s, y)$. Она означает, что перед шагом каждого из агентов A_r и A_s, они получают сообщение об изменении соответствующей переменной сделанной другим агентом с момента предыдущего шага.

$$y^{k+1} = g(X_r^k, t^{k+1}) + \Delta(t^k, t^{k+1}, x).$$

$$x^{k+1} = f(X_s^k, t^{k+1}) + \Delta(t^k, t^{k+1}, y).$$

$$\Delta(t^k, t^{k+1}, x) = f(X_r^k, t^{k+1}) - x^k$$

$$\Delta(t^k, t^{k+1}, y) = g(X_s^k, t^{k+1}) - y^k$$

Начальное значение устанавливается самой связью:

$$y^0 = z = x^0$$

Легко видеть, что если при этом $y^{k+1} = x^{k+1}, k = 0, \dots, M$. Это и есть вычисленное значение разделяемой переменной, на которое влияют оба агента (модуля). Как и в предыдущем случае, на остальные переменные эта связь не оказывает влияния.

Если временные сетки агентов различны, то чтобы получить значение $y^{k+1} = y(t^{k+1})$ необходимо применять интерполяцию.

$$y^{k+1} = p(\tilde{x}^{k+1})$$

Значение $\tilde{x}^{k+1}$ должно быть вычислено с помощью интерполяции по значениям $x^{l-m}, \dots, x^l$, где $t_r^l \geq t_s^{k+1}$ и $t_r^{l-1} < t_s^{k+1}$, m – количество точек по которым производится интерполяция.

Аналогично в случае ненаправленной связи:

$$\Delta x = \tilde{x}^{k+1} - \tilde{x}^k$$

Но так как здесь значение x также подвергается изменению извне, интерполяция должна учитывать только изменения сделанные агентом A_r, т.е. использовать точки $\left(t_r^{l-m+1}, f(x^{l-m}, t_r^{l-m+1})\right), \dots, \left(t_r^{l-m+1}, f(x^{l-m}, t_r^{l-m+1})\right)$.

3.5.4. Оценка погрешности агентного моделирования

Важным вопросом является оценка точности алгоритма численных расчетов. В общем случае для агентной модели это сделать нельзя, так как нельзя строго формализовать понятие точного решения. Рассмотрим частный случай, когда агенты содержат системы ОДУ. В этом случае точным решением будет аналитическое решение системы, полученной после генерации плоской модели. Покажем, что в этом случае расчеты агентной модели соответствуют решению соответствующей плоской модели. Итак, пусть у нас есть модель, содержащая систему дифференциальных уравнений

$$\frac{dX}{dt} = f(X, t),$$

где f – дифференцируемая функция. Данной модели сопоставлен определенный метод численного решения задачи Коши для системы ОДУ и временная сетка T. Тогда можно построить соответствующий агент $A = (\mathcal{X}, \mathcal{I}, \mathcal{O}, \mathcal{C}, X^0, T, F)$, функция

шага которого $X_i(t) = F_i(X_i^0, t^0, t)$ представляет собой процесс решения задачи Коши:

$$\begin{cases} \dfrac{dX}{dt} = G(X,t), \qquad t \in [t^k, t^{k+1}], \\ X(t^0) = X^0. \end{cases}$$

Таким образом, результат симуляции одного агента с ОДУ моделью представляет собой последовательность вычисленных значений решения задачи Коши в точках временной сетки $t^0, t^1, \dots, t^m$.

Пусть теперь имеем агентную модель, каждый агент которой содержит задачу Коши. Численные расчеты для агентов происходят по отдельности. Численные расчеты для всей модели соответствуют решению задачи Коши:

$$\frac{dX}{dt} = G(X,t), \qquad X(t^0) = X^0,$$

или в векторной форме:

$$\frac{d}{dt}\begin{pmatrix} X_1 \\ \vdots \\ X_N \end{pmatrix} = \begin{pmatrix} G_1(X_1,t) \\ \vdots \\ G_N(X_N,t) \end{pmatrix}, \qquad \begin{pmatrix} X_1 \\ \vdots \\ X_N \end{pmatrix}(t^0) = \begin{pmatrix} X_1^0 \\ \vdots \\ X_N^0 \end{pmatrix},$$

где $G_i(X_i,t)$ – правая часть задачи для i-го агента. Пусть $Y_i(t)$ – решение i-той задачи Коши. Численные расчеты для каждого из агентов генерируют последовательность

$$Y_i\left(t_i^0\right), Y_i(t_i^1), \dots, Y_i\left(t_i^{m_i}\right), \qquad i = 1, \dots, N.$$

$$z_{ij}^{(k)}(t^k) = \begin{cases} z_{ij}^{(k-1)}(t^k) + x^{(k-1)}(t^k) - x^{(k-1)}(t^{k-1}), & \exists\, x : x \leftrightarrow z_{ij} \\ p(x), \ \exists\, p, x : x \xrightarrow{p} z_{ij} \\ z_{ij}^{(k-1)}(t^k), \qquad \text{иначе.} \end{cases}$$

Покажем теперь, как соотносится решение, полученное с помощью агентной симуляции с решением системы ОДУ созданной с помощью алгоритма генерации плоской модели. Пусть $u(t^k)$ – точное решение дифференциального уравнения, созданного по правилам генерации плоской модели, изложенным выше:

$$\begin{cases} \frac{du}{dt} = f(u,t) + g(u,t), t \in [t^0, t^m] \\ u(t^0) = z^0 \end{cases}$$

Применим разложение по формуле Тэйлора:

$$u(t^0 + h) = u(t^0) + \frac{h}{1!}\frac{du}{dt}(t^0) + O(h^2),$$

$$u(t^0 + h) = z^0 + h[f(z^0, t^0) + g(z^0, t^0)] + O(h^2). \quad (3.1)$$

Пусть теперь $z(t^k), k = 0, \ldots$ – решение, полученное с помощью агентной симуляции модели из двух агентов с одной ненаправленной связью.

$$A_1: \begin{cases} \frac{dx}{dt} = f(x,t) \\ x(t^0) = x^0 \end{cases}, \quad A_2: \begin{cases} \frac{dy}{dt} = g(y,t) \\ y(t^0) = y^0 \end{cases}, \qquad x \overset{z_0}{\leftrightarrow} y.$$

Имеем:

$$z(t^1) = z(t^0 + h) = x(t^0 + h) + O(h^{q+1}) + y(t^0 + h) + O(h^{p+1}) - z^0, \quad (3.2)$$

Где x, y – точные решения задач Коши для агентов A_1 и A_2, p и q – порядки численных методов, использованных для их решения.

$$x(t^0 + h) = x(t^0) + \frac{h}{1!}\frac{dx}{dt}(t^0) + o(h^2) = z^0 + hf(z^0, t^0) + O(h^2), \quad (3.3)$$

$$y(t^0 + h) = y(t^0) + \frac{h}{1!}\frac{dy}{dt}(t^0) + o(h^2) = z^0 + hg(z^0, t^0) + O(h^2). \quad (3.4)$$

Суммируя (3.3) и (3.4) и учитывая (3.1), имеем:

$$u(t^0 + h) = x(t^0 + h) + y(t^0 + h) - z^0 + O(h^2) = z(t^0 + h) + O(h^2).$$

Т.о. имеем локальную погрешность алгоритма – $O(h^2)$. Оценим теперь глобальную погрешность метода. Разлагая u, x и y в ряд Тэйлора с остаточным членом в форме Лагранжа, имеем:

$$u^{i+1} - z^{i+1} = u^i + h[f + g]\left(u^i, t^i\right) + \frac{h^2}{2}u''(t^i + \theta^i h) - x^{i+1} - y^{i+1} + z^i. \quad (3.5)$$

$$x^{i+1} = z^i + hf\left(z^i, t^i\right) + \frac{h^2}{2}x''\left(t^i + \varphi^i h\right), \quad (3.6)$$

$$y^{i+1} = z^i + hg\left(z^i, t^i\right) + \frac{h^2}{2}y''\left(t^i + \psi^i h\right), \quad (3.7)$$

где $u^i = u(t^i),\ x^i = x(t^i),\ y^i = y(t^i),\ z^i = z(t^i),\ 0 < \theta, \varphi, \psi < 1$. Подставляя (3.6) и (3.7) в (3.5), имеем:

$$u^{i+1} - z^{i+1} = u^i - z^i + h[f+g](u^i, t^i) - h[f+g](z^i, t^i) + \frac{h^2}{2} d,$$

где

$$d = u''(t^i + \psi^i h) - x''(t^i + \theta^i h) - y''(t^i + \varphi^i h).$$

Используя теорему Лагранжа о среднем, получаем:

$$h[f+g](u^i, t^i) - h[f+g](z^i, t^i) = h(u^i - z^i)\frac{\partial}{\partial u}[f+g](\xi, t^i),$$

где $\xi = wu^i + (1-w)z^i,\ 0 < w < 1$. Обозначив ошибку $\varepsilon^i = u^i - z^i$, получаем:

$$\varepsilon^{i+1} = \varepsilon^i + h\varepsilon^i \frac{\partial}{\partial u}[f+g](\xi, t^i) + \frac{h^2}{2} u''(t^i + \psi^i h).$$

Пусть $\exists S, C_1, C_2, C_3$ т.ч. при $t \in [t^0, t^m]$ и $|\xi| < \infty$ выполнены следующие неравенства:

$$\left|\frac{\partial}{\partial u}[f+g](\xi, t^i)\right| \le S, \qquad |x''(t)| \le C_1, \qquad |y''(t)| \le C_2, \qquad |u''(t)| \le C_3$$

имеем:

$$|\varepsilon^{i+1}| \le (1 + hS)|\varepsilon^i| + (C_1 + C_2 + C_3)h^2$$

Из этой оценки следует, что алгоритм численных расчетов имеет первый порядок точности [Хакимзянов и Черный, 2003]. Таким образом, если решения задач Коши для каждого из связанных агентов имеют ограниченную вторую производную и их правые части имеют ограниченные частный производные по связанным переменным, то алгоритм численных расчетов для агентной модели имеют первый порядок точности.

Кроме того легко видеть, что в случае, когда $f(u,t) \equiv f(t),\ g(u,t) \equiv g(t)$, агентная симуляция не добавляет дополнительной погрешности. В этом случае:

$$x(t^0 + h) = x(t^0) + h\sum_{k=0}^{\infty} \frac{h^k f^{(k)}(t^0)}{k!}, \qquad y(t^0 + h) = y(t^0) + h\sum_{k=0}^{\infty} \frac{h^k g^{(k)}(t^0)}{k!}.$$

Откуда, учитывая (3.2), имеем:

$$z(t^0 + h) = z(t^0) + h\sum_{k=0}^{\infty}\frac{h^k[f^{(k)}(t^0) + g^{(k)}(t^0)}{k!} + O(h^{q+1}) + O(h^{p+1}) =$$

$$= z(t^0) + h\sum_{k=0}^{\infty}\frac{h^k[f+g]^{(k)}(t^0) + g^{(k)}(t^0)}{k!} + O(h^{q+1}) + O(h^{p+1}) =$$

$$= u(t^0 + h) + O(h^{q+1}) + O(h^{p+1}).$$

Направленные связи в модели означают, что для соответствующих переменных на каждом шаге будут использоваться значения с предыдущей временной точки, что также соответствует первому порядку метода. Если сетки времени агентов отличаются, используется интерполяция, которая вносит дополнительную погрешность.

3.6. Модульное моделирование процесса апоптоза

Описанный в данной главе подход был апробирован при создании комплексной модели программируемой гибели клеток (апоптоза) [Kutumova et al., 2012]. Модель была построена на основе девяти ранее опубликованных, воспроизводящих различные аспекты этого процесса. В отличие от моделей ССС человека, рассматриваемых в данной работе, эти модели в основном описывают белок-белковые взаимодействия, характеризуемые цепочками биохимических реакций. Сигнальные пути апоптоза, выделенные на основе этих моделей, были разделены на 13 функциональных модулей и объединены в модульную модель, которая содержит 280 белков, 372 реакции, а также 459 параметров. Полученная модульная модель представлена на рис. 3.13. Пример модуля, соответствующего сигнальному пути, индуцируемому рецептором TRAIL-R, приведен на рис. 3.14.

Основным видом связи в данной модульной модели является ненаправленная связь, указывающая на то, что два объекта представляют одну и ту же сущность (белковый комплекс). Эта сущность участвует во всех реакциях, участниками которых были связанные объекты. Для удобства визуального представления модели были использованы шины. Нахождение численного решения модели осуществляется при помощи процедуры генерации плоской модели, описанной в параграфе 3.3.

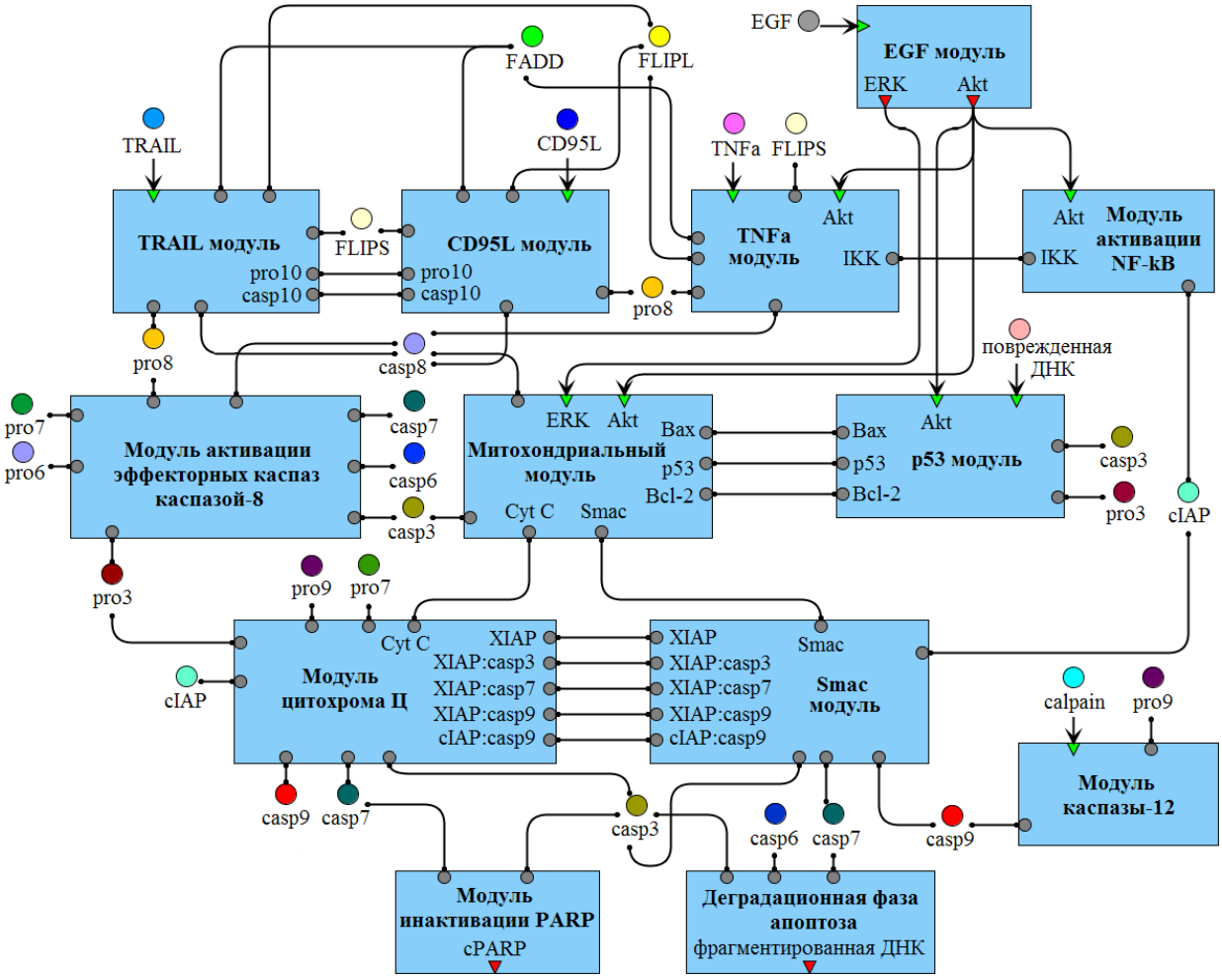


Рис. 3.13 – Модульная модель апоптоза [Kutumova et al., 2012]

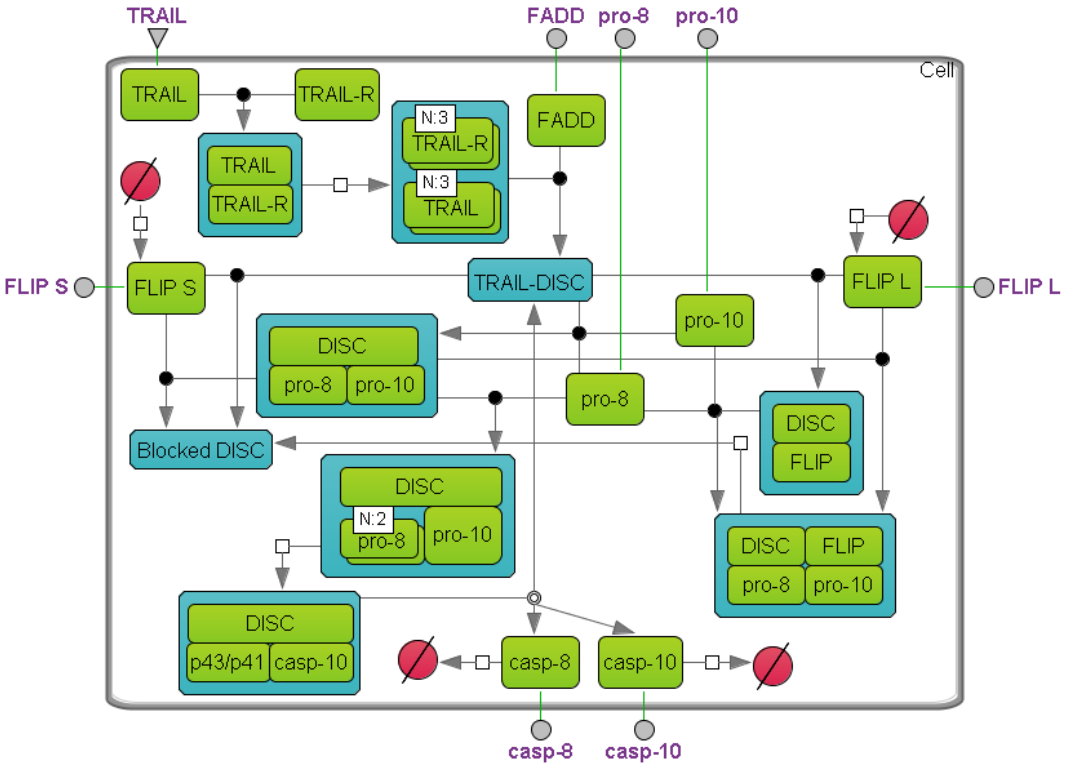


Рис. 3.15 – Модуль “TRAIL” модульной модели апоптоза (нотация SBGN)

3.7. Выводы по главе 3

Разработана концепция модульной модели – модели, включающей другие модели как составные части. Разработан алгоритм генерации “плоской” модели на основе модульной в случае, когда все модули могут быть преобразованы в единый математический формализм – ОДУ с мгновенными событиями.

Разработан алгоритм численных расчетов модульной модели на основе принципов агентного моделирования. В рамках этого подхода расчеты для модулей происходят по отдельности, координация и обмен сообщениями между ними осуществляется специальной программой планировщиком.

Показано, что в случае, когда все агенты содержат системы ОДУ с мгновенными событиями, численные расчеты на основе принципов агентного моделирования эквивалентны решению серии задач Коши, результат расчетов соответствует численным расчетам для соответствующей “плоской” модели и имеет первый порядок точности.

Разработаны алгоритмы трансформации модульной диаграммы в иерархическую SBML-модель и обратно. Алгоритмы проверены набором тестов от авторов SBML.

Все описанные алгоритмы реализованы как часть программного комплекса BioUML. Созданы новые типы диаграмм – “Модульная модель”, “Модульная SBML-модель”, “Агентная модель”.

Реализованный инструментарий опробован при создании модульной модели апоптоза [Kutumova et al., 2012].

ГЛАВА 4. МОДУЛЬНОЕ МОДЕЛИРОВАНИЕ СЕРДЕЧНО-СОСУДИСТОЙ СИТЕМЫ ЧЕЛОВЕКА

В данной главе опишем применение модульного и агентного подходов при создании модели сердечно-сосудистой системы человека. Модель создана на основе существующих моделей, путем разложения их на части, доработки и объединения этих частей в новую модель. Ниже рассмотрим использовавшиеся модели по отдельности.

4.1. Модель всеобщей циркуляции

Модель, предложенная профессором Гайтоном [Guyton et al., 1972], представляет собой первое глобальное описание процессов в человеческом организме, связанных с кровообращением, и демонстрирует долговременные эффекты регуляции ССС человека со значительной ролью почечной регуляции. В 2010 году модель была реконструирована в среде MATLAB SIMULINK [Kofranek and Rusz, 2010]. На основе данной реконструкции, модель реализована автором в системе BioUML в виде модульной диаграммы (рис. 4.1.). При этом набор блоков алгебраических операций (суммирование, умножение, интегрирование и т.д.) из которых состоит модель в SIMULINK, был преобразован в систему ОДУ.

Реализованная модель состоит из 18 модулей, суммарно включающих 229 параметров, 133 операции присваивания, 39 обыкновенных дифференциальных уравнений. Модуль “Гемодинамика”, описывающий кровоток по резервуарам большого и малого кругов кровообращения, реализован в виде модульной модели (рис 4.2), включающей 6 модулей (в порядке следования крови): “Левый желудочек”, “Артерии большого круга”, “Вены большого круга”, “Правый желудочек”, “Артерии и вены малого круга”. В качестве примера, на рисунке 4.3 приведен модуль “Артерии большого круга”. Обозначения переменных модели сохранено. Графическая нотация приведена в таблице 3.1. Расшифровка параметров приведена в Приложении Г.

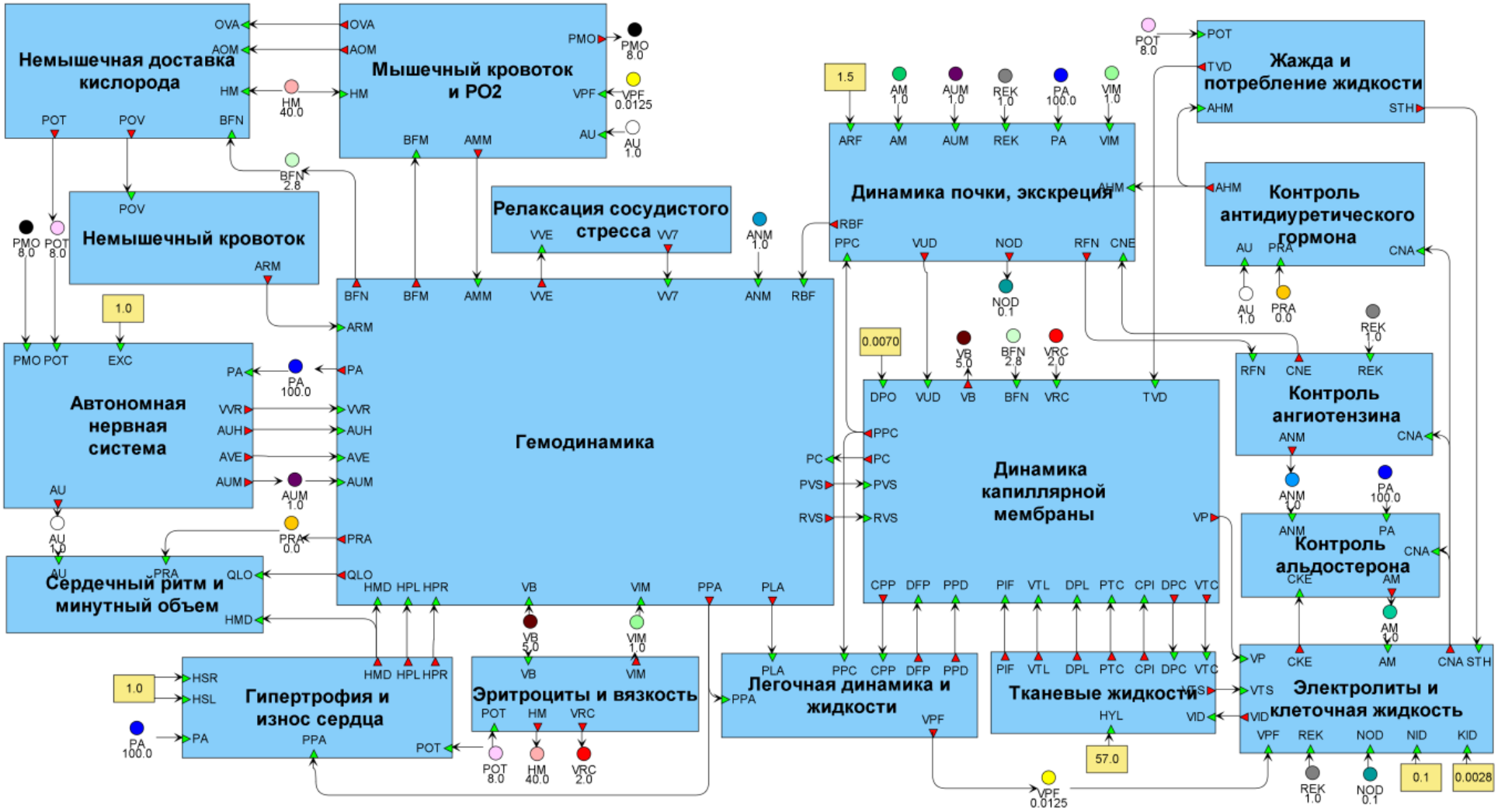


Рисунок 4.1 – Модель общей циркуляции (Guyton et al., 1972), реализованная в BioUML в виде модульной диаграммы

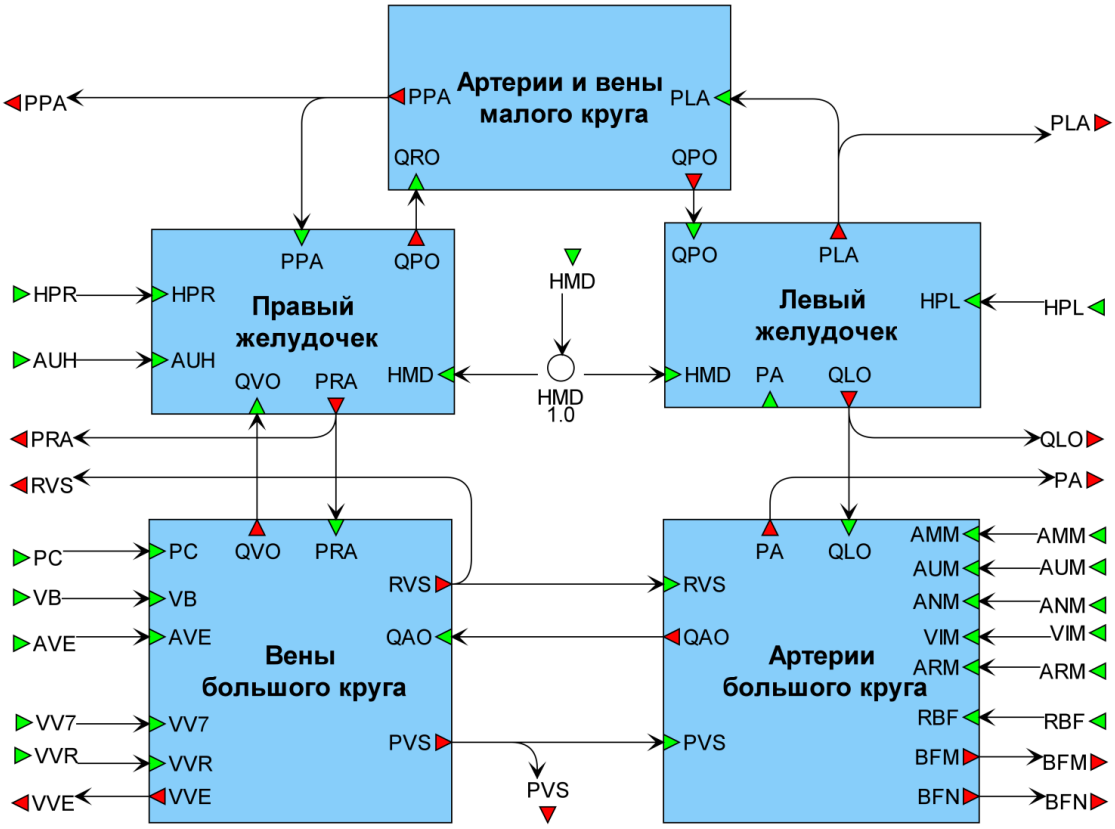


Рисунок 4.2 – Модуль “Гемодинамика” модульной модели общей регуляции, реализованный в BioUML

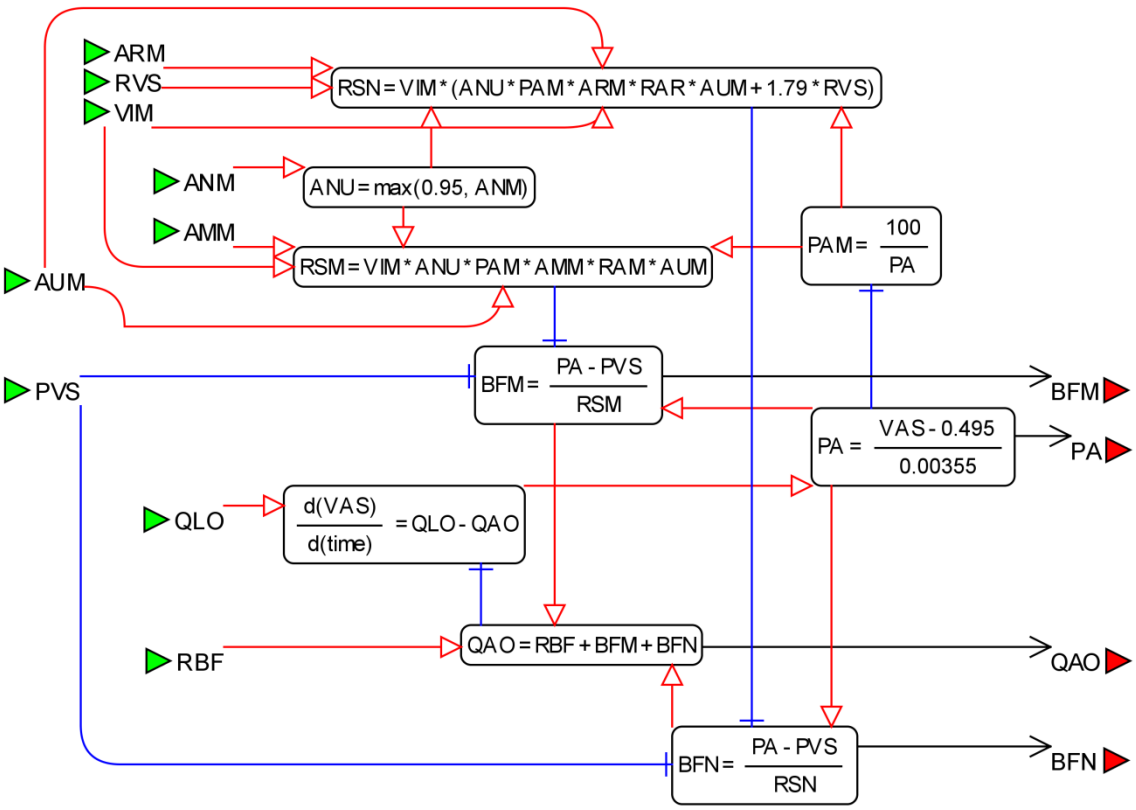


Рисунок 4.3 – Модуль “Артерии большого круга” в виде диаграммы “Модель физиологических процессов”

4.2. Модель сердечных сокращений

Модель ССС человека, делающая акцент на кратковременную регуляцию кровообращения [Солодянников, 1994], представлена в виде замкнутой системы резервуаров, включающих в себя (в порядке следования артериальной крови по системе): сердце, артериальную систему, капилляры и венозный резервуар. Кровообращение между резервуарами регулируется следующим набором законов:

$$\frac{dV_i}{dt} = Q_{ki} - Q_{ij},\ \ Q_{ij} = \frac{P_i - P_j}{R_{ij}},\ \ P_i = G_i(V_i - W_i), \tag{4.1}$$

где $i \in \{A, V, H\}$ – индекс резервуара (A – артериальная система, V – венозная, H – левый желудочек); V_i – объем крови в i-м резервуаре; P_i – давление крови в -м резервуаре; G_i – объемная эластичность i-го резервуара; W_i – ненапряженный объем i-го резервуара; Q_{ij} – поток крови из i-го в j-й резервуар; R_{ij} – сопротивление току крови из i-го в j-й резервуар.

Объем крови венозного резервуара рассчитывается исходя из предположения постоянства общего объема крови в системе:

$$V_V = A_{30} - V_H - V_A. \tag{4.2}$$

Модель учитывает влияние кислородного обмена в организме, используя концепцию кислородного долга. Сердце моделируется как насос, переключающийся между двумя состояниями – систолой и диастолой. Во время систолы активируется кровоток из желудочка в артериальный резервуар, во время диастолы – из венозного резервуара в желудочек. Переход от систолы к диастоле определяется в соответствии с законом Франка-Старлинга, согласно которому ударный объем (объем крови, выброшенный сердцем за одно сокращение) ϑ зависит от конечно-диастолического объема V_{ED} и инотропных коэффициентов k_1 и k_2: $\vartheta = k_1 V_{ED} - k_2$. Этот объем можно контролировать через текущий объем крови в желудочке. Таким образом, условие перехода от систолы к диастоле выглядит следующим образом:

$$(V_H \geq V_{ED} - \vartheta) \wedge (S = 1).$$

При этом происходит переключение индикатора систолы $S = 0$, запоминается длительность прошедшей фазы $T_S = T_{Current}$, таймер текущей фазы обнуляется: $T_{Current} = 0$ и рассчитывается объем сердечного выброса за прошедшую систолу:

$$V_{out} = V_{ED} - V_V.$$

Обратный переход от диастолы к систоле происходит в момент истечения времени, отведенного на один сердечный цикл. В модели это время управляется нейрогуморальным фактором и равно $1/\gamma$. Это условие может быть записано в виде:

$$\left(T_{Current} \geq \frac{1}{\gamma} - T_{Systole}\right) \wedge \ (S = 0).$$

При этом переключается индикатор систолы $\mathrm{S} = 1$, обнуляется таймер текущей фазы $\mathrm{T_{Currrent}} = 0$, запоминаются конечно-диастолические значения влияния нейрогуморального фактора $\gamma_{\mathrm{ED}} = \gamma$ и объем крови в желудочке $\mathrm{V_{ED}} = \mathrm{V_H}$.

При реализации модели в BioUML были сделаны модификации дающие возможность варьирования общего объема крови в системе. Изменение объема крови в резервуаре определяется уравнением:

$$\frac{dV_i}{dt} = Q_{ki} - Q_{ij} + \mathrm{c}_i \frac{dV}{dt},$$

где c_i – коэффициенты, задающие распределение изменения объема крови между резервуарами. Данные коэффициентов были взяты из [Thomas et al., 2008]. В разрешенной относительно производных форме этот закон задается в следующем виде.

$$\begin{cases} \dfrac{d\tilde{V}_i}{dt} = Q_{ki} - Q_{ij}, \\ V_i = \tilde{V}_i + \mathrm{c}_i V. \end{cases}$$

Уравнение для объема крови венозного резервуара (4.2) заменено дифференциальным уравнением согласно (4.1). Также была изменена формулы расчета периферической проводимости C_{AV}, а именно взята формула из более поздней модели того же автора [Моделирование кровообращения]:

$$C_{AV} = C_{AV_0} + DO_2 - C_H H,$$

вместо

$$C_{AV} = C_{AV_0} + DO_2 + C_H H.$$

Новая формула отражает вазоконстрикторный эффект при увеличении нейрогуморальных воздействий. Данное изменение также позволило более точно смоделировать физическую нагрузку (см. параграф 4.6.3).

Окончательно модель записывается в виде системы ОДУ с двумя мгновенными событиями:

$$\frac{d}{dt}\begin{pmatrix} \gamma \\ BO_2 \\ DO_2 \\ V_H \\ V_A \\ V_V \\ T_{Current} \end{pmatrix} = \begin{pmatrix} A_{12}A_{13}Q_{AV}/V_A - A_{12}A_{14}(P_A - A_{28}) - A_{12}\gamma \\ A_1(Q_{AV}(A_{22} - BO_2) - A_{15}) \\ -A_2(Q_{AV}(A_{22} - BO_2) - A_{15}) \\ Q_{VH} - Q_{HA} \\ Q_{HA} - Q_{AV} \\ Q_{AV} - Q_{VH} \\ 1 \end{pmatrix},$$

где

$$Q_{AV} = (A_{16} + A_4 DO_2 - A_3\gamma)(P_A - P_V), \qquad Q_{HA} = SA_{17}(P_S - P_A),$$

$$Q_{VH} = (1 - S)(A_{18} + A_7A_{15} + A_{10}A_{28} + A_6P_V)(P_V - P_D),$$

$$P_A = (A_{20} + A_9\gamma)(V_A - A_{19} - A_8\gamma), \qquad P_V = (A_{11}\gamma + A_{21})(A_{23} - V_V),$$

$$P_D = A_{29}\big(A_{24}((V_H - A_{26})^2 + A_{25}(V_H - A_{26})\big),$$

$$V_i = \tilde{V}_i + \mathrm{c}_i V, \qquad i \in \{A, V, H\},$$

$A_1, \ldots, A_{30}$ – константы, характеризующие конкретный организм, P_S – систолическое давление в желудочке, P_D – диастолическое давление в желудочке, S – индикатор фазы систолы.

В рамках данной работы, модель реализована в системе BioUML в виде модульной диаграммы (рис. 4.4), состоящей из 6 модулей: “Артериальная система”, “Желудочек”, “Венозная система”, “Капилляры”, “Нейрогуморальный фактор”, “Тканевый метаболизм” и 22 связи между ними. Графическая нотация приведена в таблице 3.1. Описание отдельных модулей приведено в Приложении Б. Расшифровка обозначений – в Приложении В. Модель включает в себя 58 переменных, 7 дифференциальных уравнений, 14 операций присваивания и 2 исполняемых события для мгновенного перехода между состояниями. Алгоритм

для численного решения, встроенный в BioUML, на каждом шаге численных расчетов проверяет условие перехода и рассчитывает точное время перехода модели из одного состояния в другое.

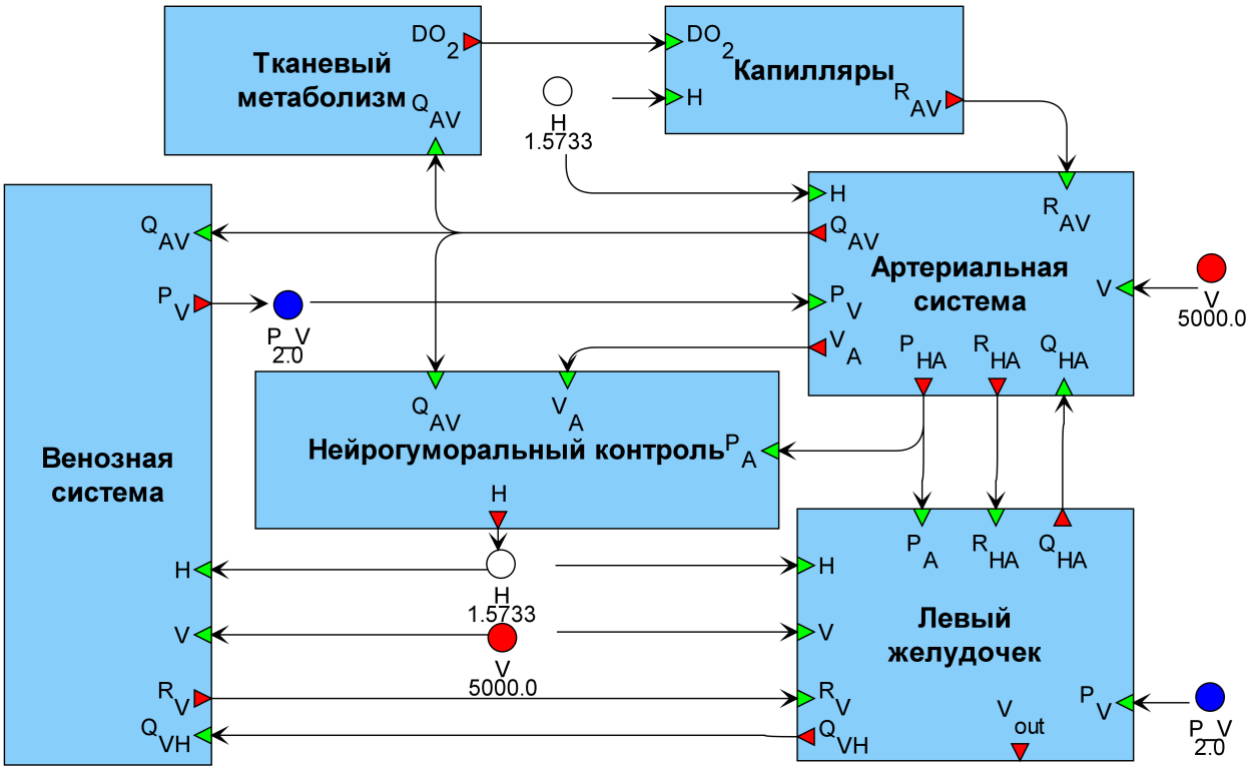


Рисунок 4.4 – Модель краткосрочной регуляции ССС человека с пульсирующим сердцем [Солодянников Ю., 1994], реализованная в виде блочной диаграммы в BioUML

Модель позволяет имитировать состояние физической нагрузки, различные патологии, связанные с нарушением сердечного цикла, такие как аритмия, кардиогенный шок и т.д. Далее в тексте модель упоминается как “модель сердечных сокращений”.

4.3. Модель почечной регуляции

Почка является важным компонентом ССС, регулирующим объем и состав жидкости в организме. Одной из наиболее проработанных моделей человеческой ССС, подробно описывающих работу почки, является модель долговременной регуляции артериального давления [Karaaslan et al., 2005]. Данная модель была создана на основе предшествующих моделей [Guyton et al., 1972; Uttamsingh et al., 1985; Coleman, Hall, 1992]. Схема модели из [Karaaslan et al., 2005] представлена на рис. 4.5. Всего в модели 86 переменных, участвующих в 9 интегральных и 63

алгебраических уравнениях. В процессе переписки с авторами модели, был получен ряд исправлений в формулах модели.

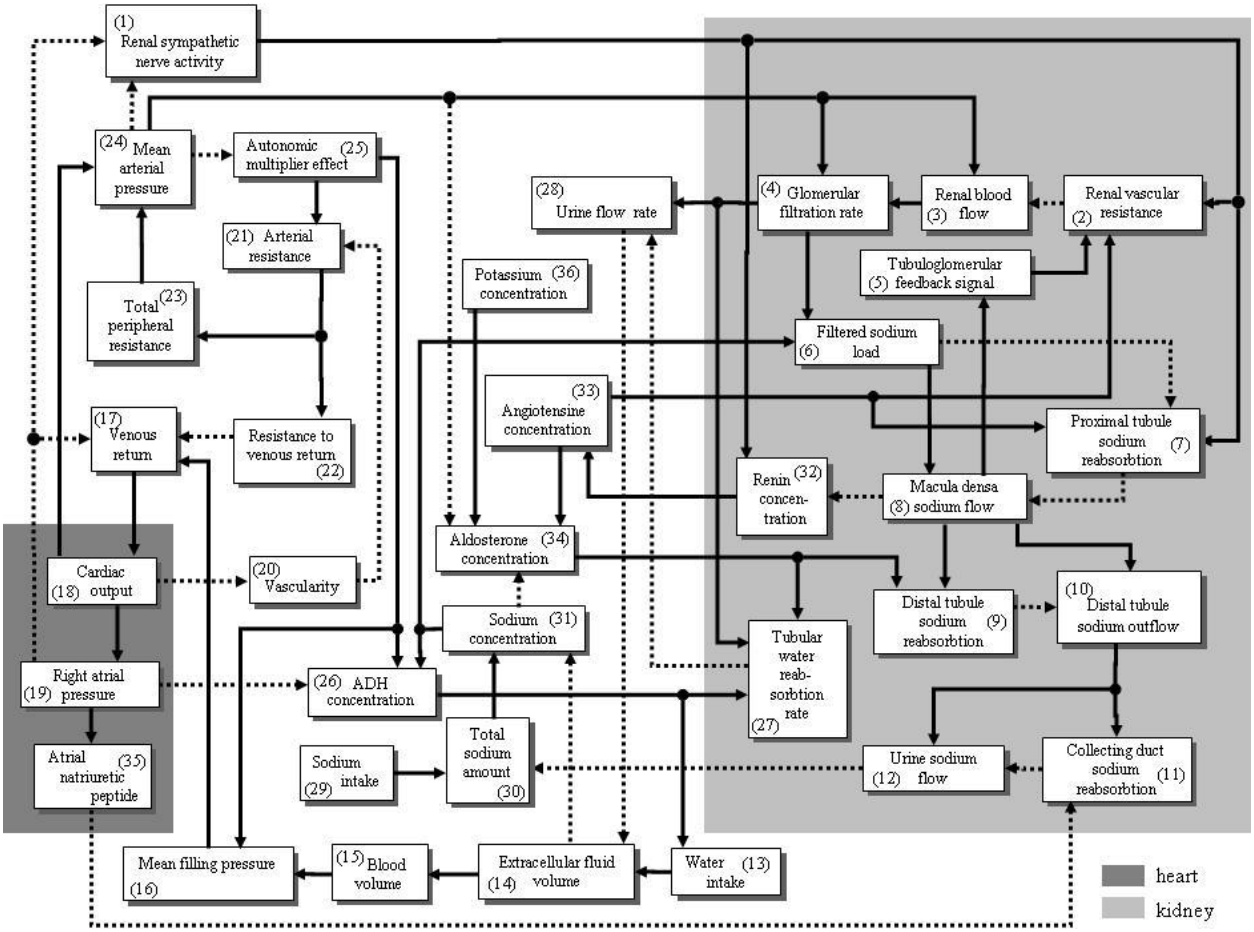


Рисунок 4.5 – Модель почечной регуляции [Karaaslan et al., 2005]. Каждый блок представляет собой набор уравнений для расчета определенных переменных модели. Стрелки указывают на зависимости между этими переменными. Непрерывная стрелка указывает на стимулирующий эффект, пунктирная – на подавляющий эффект. Исправленная версия получена в результате частной переписки, публикуется с разрешения автора

В рамках диссертационной работы, модель реализована в платформе BioUML. При этом интегральные уравнения преобразованы в дифференциальные с добавлением дополнительных переменных там, где это необходимо:

$$\frac{dV_{ecf}}{dt} = \Phi_{win} - \Phi_u, \qquad \frac{dvas}{dt} = vas_f - vas_d,$$

$$\frac{dC_r}{dt} = \frac{N_{rs} - C_r}{T_r}, \qquad \frac{dN_{al}}{dt} = \frac{N_{als} - N_{al}}{T_{al}},$$

$$\frac{dN_{adh}}{dt} = \frac{N_{adhs} - N_{adh}}{T_{adh}}, \qquad \frac{dM_{sod}}{dt} = \Phi_{sodin} - \Phi_{u-sod},$$

$$\frac{dtemp_1}{dt} = 5 \times 10^{-4}(a_{baro} - 0.75)\,, \qquad a_{baro} = 0.75a_{auto} - temp_1,$$

$$\frac{dtemp_2}{dt} = \delta_{ra}, \qquad \delta_{ra} = 0.2P_{ra} - 7 \times 10^{-4}temp_2.$$

Алгебраические уравнения в [Karaaslan et al., 2005] представлены в форме, разрешенной относительно искомой переменной, формируя циклические зависимости. Эту проблему удалось решить превращением трех алгебраических уравнений в дифференциальные по схеме:

$$x = f(X) \Rightarrow \frac{dx}{dt} = \frac{f(X) - x}{T}.$$

Новые дифференциальные уравнения, добавленные в модель:

$$\frac{d\varepsilon_{aum}}{dt} = \alpha_{chemo} - \alpha_{baro} - \varepsilon_{aum},$$

$$\frac{d\Phi_{md-sod}}{dt} = \Phi_{filsod} - \Phi_{pt-sodreab} - \Phi_{md-sod},$$

$$\frac{d\Phi_{co}}{dt} = \frac{P_{mf} - P_{ra}}{R_{vr}} - \Phi_{co}.$$

Здесь нужно отметить, что хотя такая замена потенциально может внести новую динамику в систему, она также соответствует биологическому смыслу модели – биологические системы подстраиваются под изменения не мгновенно (что описывалось бы алгебраическими уравнениями), а с некоторой задержкой. Этот подход применяется, например в [Kofranek and Rusz, 2010].

Реализованная в BioUML версия модели состоит из 6 модулей: “Сердце”, “Артерии”, “Почка”, “Почечная нервная система”, “Гормональная система”, “Контроль объема жидкости” (рис. 4.6.), содержит 85 параметров, 61 операцию присваивания и 11 обыкновенных дифференциальных уравнений. Графическая нотация приведена в таблице 3.1. Описание отдельных модулей приведено в Приложении Б. Расшифровка обозначений – в Приложении В. Далее в тексте модель упоминается как “модель почечной регуляции”.

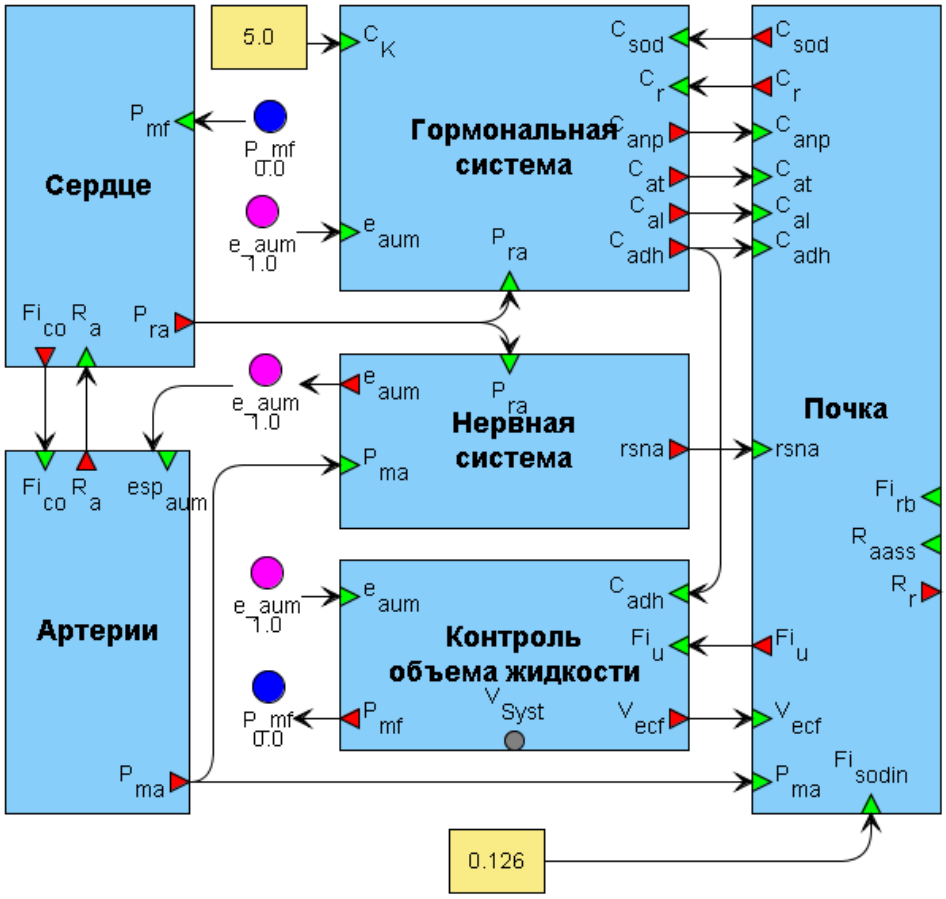


Рисунок 4.6 – Модель долгосрочной почечной регуляции ССС человека [Karaaslan et al., 2005], реализованная в виде модульной диаграммы в BioUML. Обозначения параметров модели сохранены

4.4. Объединенная модель ССС человека (вариант 1)

Первый вариант модели создан путем объединения моделей сердечных сокращений [Солодянников, 1994] и почечной регуляции [Karaaslan et al., 2005]. Модели используют одинаковый формализм – ОДУ с мгновенными событиями, однако имеют различные шкалы времени – секунды и минуты, соответственно, при этом интервал моделирования для модели почечной регуляции составляет часы и дни. Это существенно затрудняет их объединение обычным образом. Как видно из рис. 4.4 и 4.6, эти модели имеют блоки, описывающие одни и те же биологические объекты: сердце и артериальную систему. При этом модель сердечных сокращений содержит более детализированное представление работы сердца и артерий, в то время как модель долговременной почечной регуляции содержит усредненные по времени переменные. Исходя из этих соображений, эти блоки из модели почечной регуляции нужно заменить соответствующими

блоками модели сердечных сокращений – см. рис. 4.7. Для корректного объединения необходимо передать в модель почки выходные переменные этих модулей: среднее артериальное давление P_{ma} и давление в правом предсердии P_{ra}.

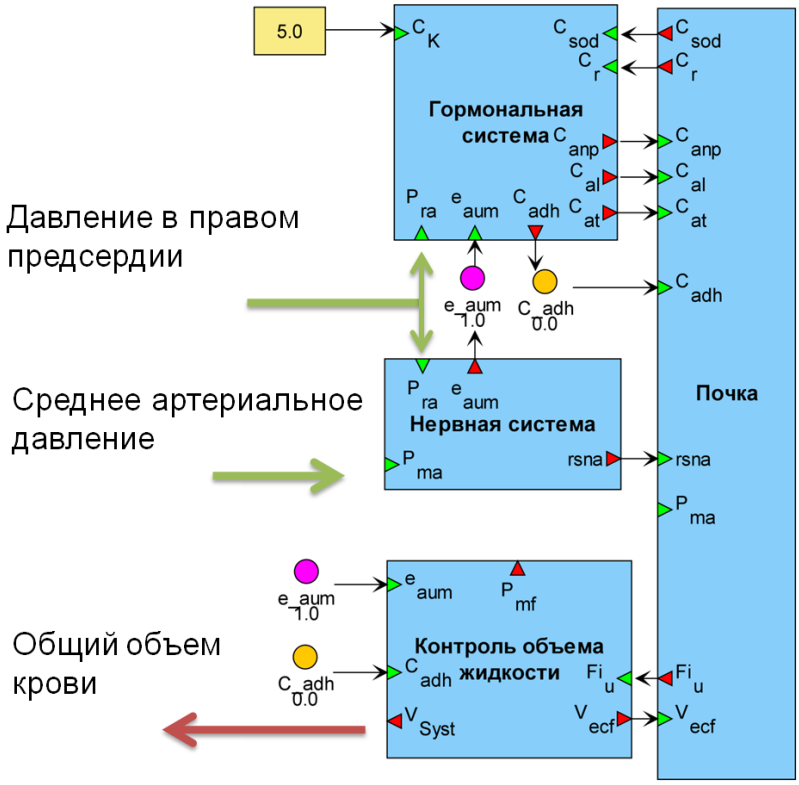


Рисунок 4.7 – Удаление блоков “Сердце” и “Артериальная система” из модели почечной регуляции

Среднее артериальное давление в модели почечной регуляции рассчитывается на основе величины сердечного выброса и периферической сосудистой сопротивляемости. В рамках модульной модели будем его рассчитывать, усредняя по времени колеблющееся артериальное давление P_A, передавая его сначала в усредняющий модуль и далее – в модель почечной регуляции. Таким образом, в модели создается модуль “Усреднитель 1” и устанавливаются следующие направленные связи:

$$(\text{"Усреднитель 1"}, out) \rightarrow (\text{"Нервная система"}, P_{ma}),$$

$$(\text{"Усреднитель 1"}, out) \rightarrow (\text{"Почка"}, P_{ma}).$$

Важным фактором в модели с почечной регуляцией является уровень давления в правом предсердии P_{ra}, влияющий на почечную симпатическую нервную

активность, секрецию антидиуретического гормона и натрийуретического пептида. Давление в правом предсердии является функцией минутного объема и рассчитывается на основе закона Франка-Старлинга:

$$P_{ra} = 0.2787e^{0.2281\Phi_{co}}.$$

При удалении блока “Сердце” необходимо рассчитывать этот параметр в другом месте. Рассчитать его в рамках модели сокращающегося сердца невозможно, т.к. в ней отсутствует малый круг кровообращения, как и правое предсердие. Таким образом, выделим уравнение для расчета P_{ra} из модуля “Сердце” в отдельный модуль “Правое предсердие”. Этот модуль принимает значение среднего по времени сердечного выброса (минутного объема) и выдает значение давления в правом предсердии. Перед передачей значения минутного объема в “Правое предсердие” его необходимо усреднить по времени, чтобы избавиться от осцилляций. Таким образом, в модульную модель добавляется модуль “Усреднитель 2” и устанавливаются связь:

$$(\text{"Усреднитель 2"}, out) \rightarrow (\text{"Правое предсердие"}, \Phi_{co}).$$

Полученная модель представлена на рис. 4.8.

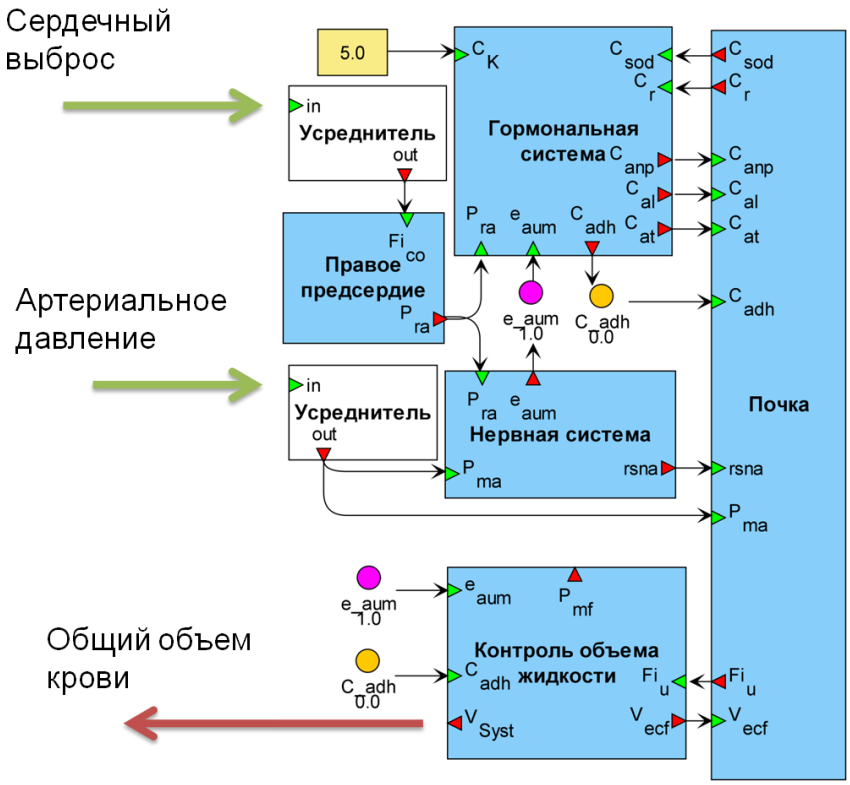


Рисунок 4.8 – Добавление усреднителей и модуля “Правое предсердие”

Теперь мы можем установить связи непосредственно с блоками модели сердечных сокращений:

(“Левый жуледочек”, P_A) → ("Усреднитель 1", in),

(“Левый жуледочек”, V_{out}) → ("Усреднитель 2", in),

Важным объектом регуляции в модели почечной регуляции является общий объем крови в системе, в то время как в модели сердечных сокращений он постоянен. Для учета этой регуляции в модульной модели, устанавливаются связи между общим объемом крови модели почечной регуляции V_b и общим объемом крови модели сердечных сокращений V:

(“Контроль объема жидкости“, V_b) $\xrightarrow{V_b}$ ("Венозная система", V),

(“Контроль объема жидкости“, V_b) $\xrightarrow{V_b}$ ("Артериальная система", V),

(“Контроль объема жидкости", V_b) $\xrightarrow{V_b}$ ("Левый желудочек", V),

Итоговая модель представлена на рис. 4.9. и состоит из 13 модулей, включая два модуля-усреднителя. Графическая нотация приведена в таблице 3.1. Описание отдельных модулей приведено в Приложении Б. Расшифровка обозначений – в Приложении В.

4.5. Объединенная модель ССС человека (вариант 2)

Созданная комплексная модель ССС человека (рис. 4.7) содержит модуль, описывающий артериальную систему человека как единый резервуар, в то время как модель артериального дерева (см. раздел 2.4) содержит описание тока крови по 55 наиболее крупным сосудам. Задача заключается в замене модуля “Артериальная система” на модуль, содержащий модель артериального дерева. Для этого необходимо добавить в модель артериального дерева расчет выходных переменных модуля “Артериальная система”, которые используются в остальных модулях модели: общий объем крови в артериальном резервуаре $\boldsymbol{V_V}$, среднее артериальное давление $\boldsymbol{P_A}$ и поток крови из артериальной системы $\boldsymbol{Q_{AV}}$.

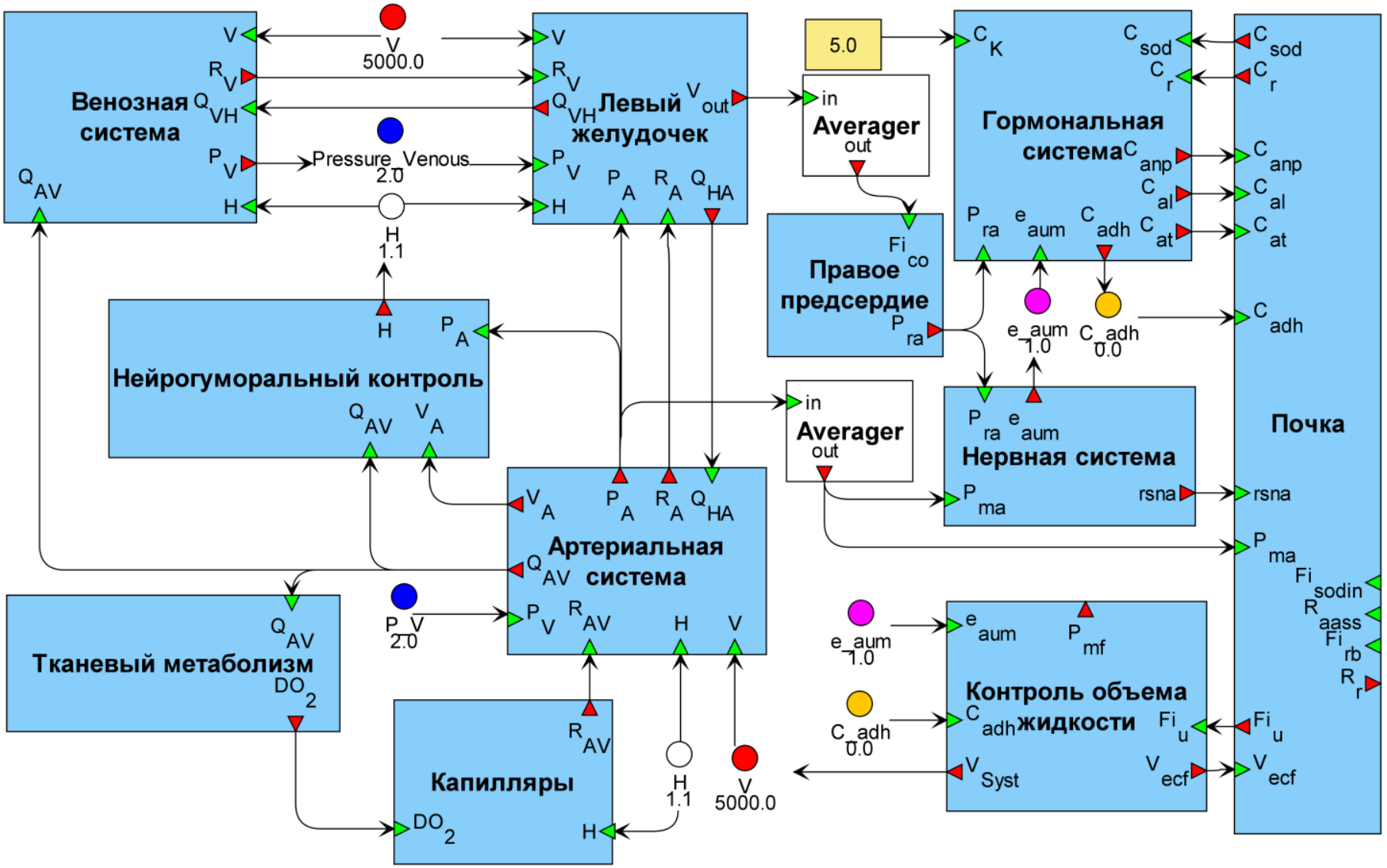


Рисунок 4.9 – Комплексная модель ССС человека, построенная в BioUML из блоков моделей сердечных сокращений [Солодянников, 1994] и модели почечной регуляции [Karaaslan et al., 2005].

Индекс A' обозначает, что соответствующая переменная рассчитывается в модели артериального дерева. Между моделями устанавливается следующая связь:
(“Артериальное дерево”, $V_{A'}$) → ("Нейрогуморальный контроль", V_A).
Вместо среднего давления в модели сердечных сокращений будет использоваться давление в аорте. При этом устанавливаются связи:

(“Артериальное дерево”, p_1) → ("Нейрогуморальный контроль", P_A),

(“Артериальное дерево”, p_1) → ("Левый желудочек", P_A).

Выходной поток из артериального дерева вычисляется по формуле:

$$Q_{A'V} = \sum_{i=1}^{55} Q_i(t, l_i).$$

Для его передачи в модель серденчых сокращений устанавливаются связи:
(“Артериальное дерево”, $Q_{A'V}$) → ("Капилляры", Q_{AV}),
(“Артериальное дерево”, $Q_{A'V}$) → ("Тканевый метаболизм", Q_{AV}).

С другой стороны, для численного решения уравнения гидродинамической модели артериального дерева необходимо задать краевые условия на свободных концах сосудов, что может быть сделано несколькими способами. В соответствии с входными переменными модуля артериальной системы модели сердечных сокращений, будем использовать поток крови на входе в восходящую аорту Q_{HA} и условие фильтрации на концах терминальных сосудов:

$$Q_i(t, l_i) = \frac{A_i(t, l_i)}{\sum_{j \in \aleph} A_j\left(t, l_j\right) R_{AV}} \left(p_i(t, l_i) - P_{veins}(t)\right), i \in \aleph,$$

$$Q_1(t, 0) = Q_{in}(t).$$

Связи:

(“Левый желудочек”, Q_{HA}) → ("Артериальное дерево", Q_{in}),

(“Капилляры”, P_V) → ("Артериальное дерево", P_{veins}),

(“Капилляры”, R_{AV}) → ("Артериальное дерево", R_{AV}).

Учет присутствующих в модели почки эффектов влияния вегетативной нервной системы ε_{aum} и васкуляризации vas на артериальное дерево достигается

передачей соответствующих значений переменных в модель артериального дерева и коррекцией общего артериального сопротивления:

$$R_{HA'}^{new} = \frac{\varepsilon_{aum}}{vas} R_{HA'}.$$

Как и в случае модели 1, будем учитывать изменение общего объема крови почкой в модели артериального дерева.

Для более детальной связи моделей на концах почечных артерий артериального дерева зададим граничное условие:

$$Q_{AR}(t) = Q_{r_1}(t, l_{r_1}) + Q_{r_2}(t, l_{r_2}) = \frac{p_r^{Average}}{R_r(t)} =$$

$$= \frac{A_{r_1}(t, l_{r_1}) p_{r_1}(t, l_{r_1}) + A_{r_2}(t, l_{r_2}) p_{r_2}(t, l_{r_2})}{R_r(A_{r_1}(t, l_{r_1}) + A_{r_2}(t, l_{r_2}))},$$

где r_1 и r_2 – индексы почечных артерий, R_r – почечная сосудистая сопротивляемость, рассчитываемая в модели почки с учетом влияния автономной нервной системы и потока натрия через *macula densa*, $p_r^{Average}$ – среднее давление почечных артерий, l_i – длина i-го сосуда. Базовая сопротивляемость афферентных артериол R_{aa-ss} является константной, поэтому будем ее рассчитывать в модели артериального дерева, основываясь на сопротивляемости почечных артерий:

$$R_{aa-ss}^{Mean} = c_3 Average(r_{r_1} + r_{r_2}),$$

где c_3 – поправочный коэффициент, введенный для моделирования блока артериол в модели артериального дерева из 55 основных сосудов. *Average* – скользящее среднее.

Созданная таким образом комплексная модульная модель ССС человека представлена на рис. 4.10. Графическая нотация приведена в таблице 3.1. Описание отдельных модулей приведено в Приложении Б. Расшифровка обозначений – в Приложении В.

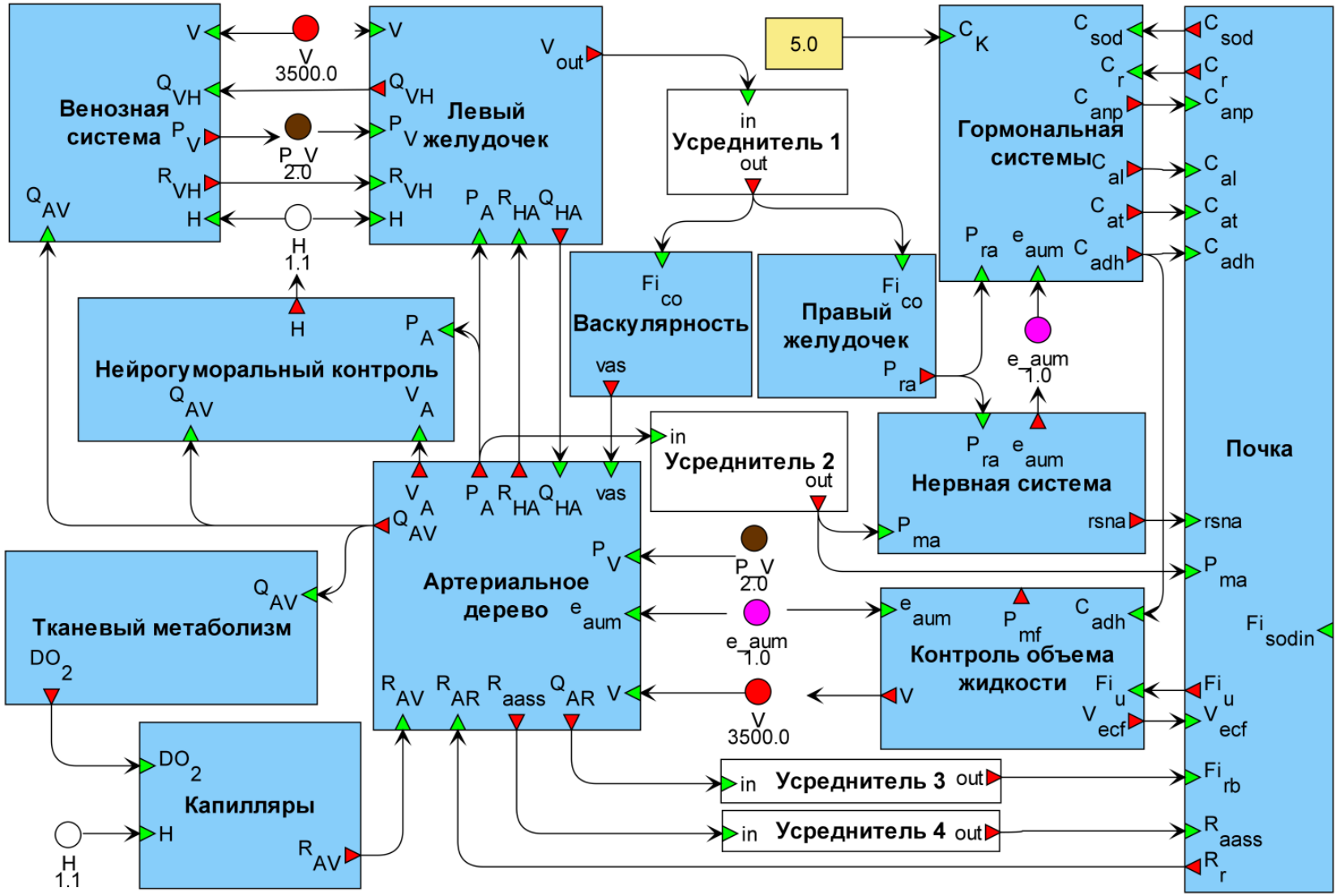


Рисунок 4.10 – Комплексная модель 2 ССС человека, собранная в системе BioUML из блоков моделей сердечных сокращений, модели почечной регуляции и модели артериального дерева

4.6. Тестирование модели

Прежде чем использовать созданные комплексные модели необходимо провести их валидацию – протестировать их в нормальных и специальных состояниях и сравнить значения физиологических параметров демонстрируемых моделями со значениями, наблюдающимися в реальности. Отметим, что при тестировании модели такого рода возникают следующие проблемы:

- Данные в литературе обычно охватывают небольшой спектр переменных модели, таким образом, приходится комбинировать данные различных экспериментов, относящихся к разным людям.
- Биологически допустимыми являются достаточно широкие диапазоны значений переменных, при этом неочевидными являются допустимые комбинации значений различных переменных.
- Чаще всего данные относятся к людям, страдающим тем или иным заболеванием сердечно-сосудистой системы, что затрудняет валидацию модели в норме.

Наиболее полной из созданных моделей является комплексная модель 2 (рис. 4.8.). Однако в некоторых случаях (эксперименты с варьированием потребления соли) мы будем использовать для проведения численных экспериментов более простую модель 1, так как модель артериального дерева имеет гораздо большие вычислительные затраты, чем другие используемые блоки.

4.6.1. Тестирование модели в норме

В таблице 4.1. приведен набор параметров комплексной модели 2, значения, полученные в модели, их границы нормы и т.н. “типичные” значения. В некоторых случаях эти значения взяты из литературы, в других были взяты средние значения из диапазона нормы. Почечный кровоток в модели рассчитывается путем усреднением суммы кровотоков в левой и правой почечной артериях. Скорость клубочковой фильтрации в модели также как и в [Karaaslan et al., 2005] нормирована на площадь поверхности тела, которую полагаем в среднем для мужчин равной 1.92 м^2 [Sacco et al., 2010]. Диапазон нормальных значений (в

мл/мин/1.73 м^2) приведен согласно данным для мужчин младше 40 лет из обзора в [Delanaye et al., 2012], включающем 25 исследований. Из таблицы 4.1. видно, что значения в модели укладываются в допустимые диапазоны и согласуются с типичными значениями. Необходимо также отметить, что значения приведены по комплексной модели 2, включающей в себя модель артериального дерева, значения в комплексной модели 1, в частности систолическое и диастолическое давления могут отличаться (см. рис. 4.10).

Таблица 4.1 –Параметров модели комплексной модели 2 ССС человека (мужчины)

Название	**Возможные значения**	**Типичное значение**	**Модель**	**Ед. измерения**
Общий объем крови	0.07 вес тела (кг)	5,25 (при весе 75кг)	5.307	л
Ударный объем	60-100	70	72	мл/удар
Пульс	60-100	75	75.7	удар/мин
Артериальное давление диастолическое	60-90	80	78.3	мм рт. ст.
Артериальное давление систолическое	100-140	120	121.9	мм рт. ст.
Объем крови в желудочке конечно-диастолический	65-219 [Cain et al., 2009]	142 [Cain et al., 2009]	154	мл
Объем крови в желудочке конечно-систолический	12-99 [Cain et al., 2009]	55.5 [Cain et al., 2009]	82	мл
Альдостерон в крови	70 – 300 [Aldosterone in Blood]	194.49 [Feng et al., 2001]	103.29	нг/л
Натрий в крови	136 – 145 [Sodium (Na) in Blood]	140.5 [Feng et al., 2001]	143.6	ммол/л
Ангиотензин II в крови	12-36 [Catt et al., 1969]	24	20.426	нг/л
Скорость клубочковой фильтрации (GFR)	$(81-144)\frac{1.92}{1.73}=$ $=89.8-159.8$ [Delanaye et al., 2012]	93.25 [Barba et al., 1996]	125.7	мл/мин
Почечный кровоток	1125.1-1328.9 [Функциональные методы исследования почки]	1227	1200	л/мин

4.6.2. Эксперимент с солевой нагрузкой

Проведем эксперимент с варьированием уровня потребления соли Φ_{sodin} с комплексной моделью 1 (без модели артериального дерева). Эксперимент описан

в [Feng et al., 2001] и заключается в варьировании потребления соли – увеличении примерно в два раза и последующем уменьшении до 6% от нормального уровня спустя пять дней. Общее модельное время расчетов – 12 дней. Наблюдаемые параметры – концентрация альдостерона, ренина и натрия в крови. Параметры замерялись в норме, до начала эксперимента, в конце периода высокой солевой нагрузки (через пять дней после начала эксперимента) и через 5 дней после понижения солевой нагрузки.

В этом и следующем эксперименте, поступим также как в [Karaaslan et al., 2005], а именно масштабируем данные так, чтобы в норме значения совпали со значениями, получаемыми в ходе численных расчетов модели в норме.

$$x_{scaled} = x\frac{Normal_{model}}{Normal_{lab}},$$

где $Normal_{model}$ - значение переменной в норе в модели, $Normal_{lab}$ - значение переменной в норме в лабораторных данных. Таким образом, мы будем оценивать относительное изменение параметров во время эксперимента. Динамика потребления соли после масштабирования на нормальное значение в модели определяется следующим уравнением:

$$\Phi_{\text{sodin}} = \begin{cases} 0.126 \text{ ммоль/мин}, & \text{t} < 24 \text{ ч}, \\ 0.2432 \text{ ммоль/мин}, & 24 \text{ ч} < t < 24 \times 6 \text{ ч}, \\ 0.0069 \text{ ммоль/мин}, & \text{t} > 24 \times 6 \text{ ч}. \end{cases}$$

Масштабированные значения параметров приведены в таблице 4.2. Значения, полученные в ходе численных расчетов, приведены в таблице 4.3. Динамика значений в сравнении с экспериментальными данными приведена на рис. 4.11. На рис. 4.12. приведены графики колеблющегося артериального давления на первой секунде, на 5000 минуте (около 3.5 дней), на 10000 (около 7 дней) и 15000 минутах (около 10.5 дней). Графикам в комплексной модели также соответствуют агенты, которые начинают и заканчивают работать в определенные моменты времени, что позволило получить графики колеблющегося артериального давления с шагом по времени 0.01 секунды при общем модельном времени расчетов около 17000 минут.

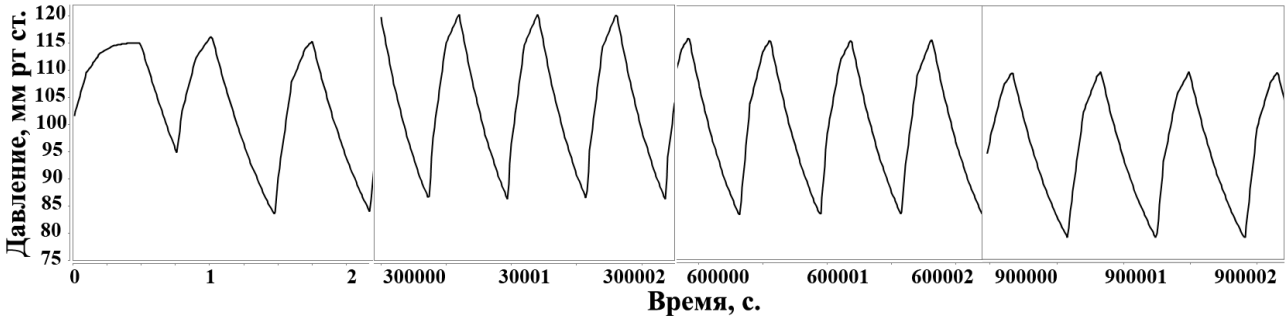


Рисунок 4.12 – Эксперимент с солевой нагрузкой в комплексной модели 1. Динамика артериального давления. Смотри пояснения в тексте

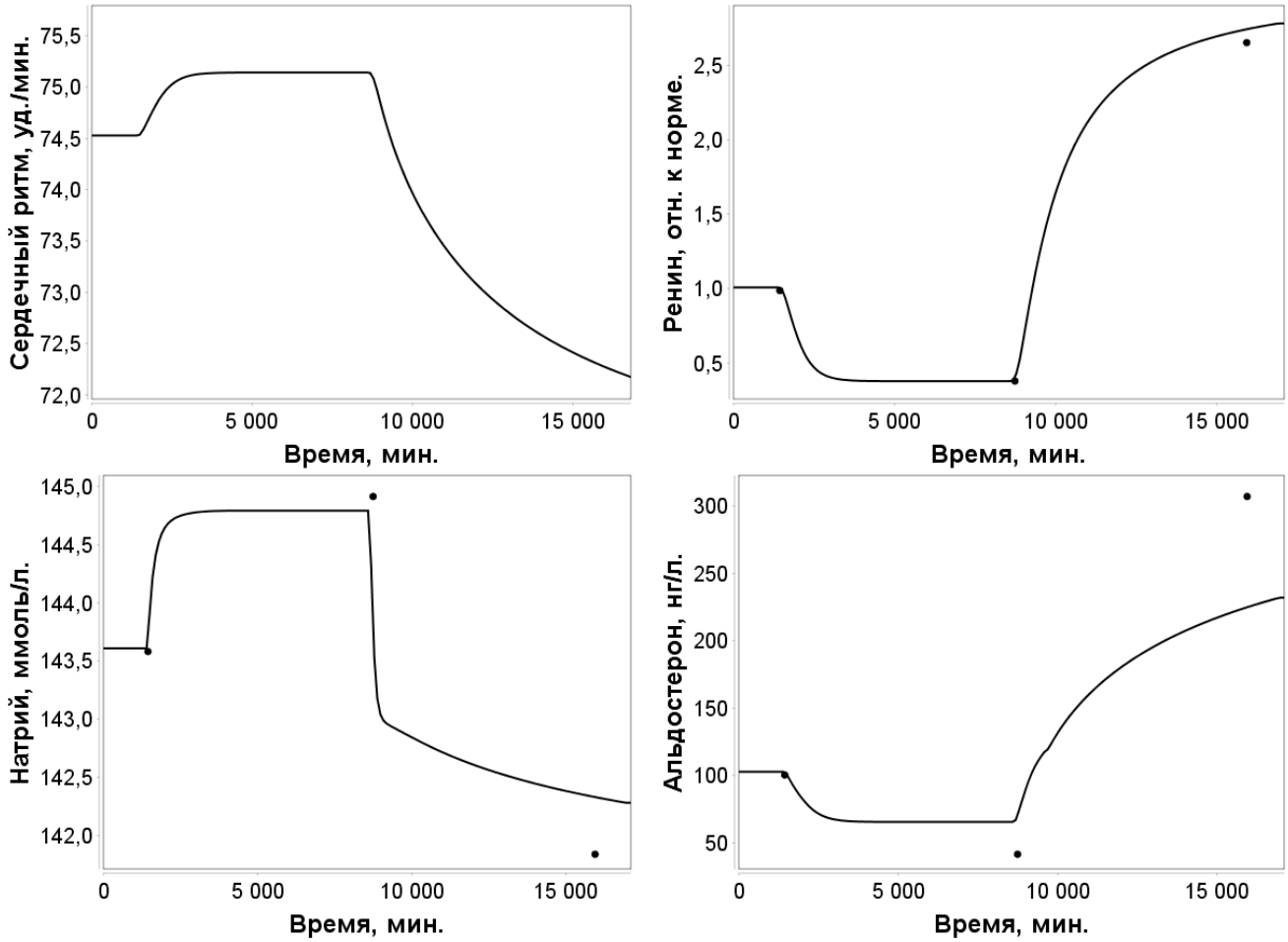


Рисунок 4.11 – Эксперимент с солевой нагрузкой в комплексной модели 1. Долгосрочная динамика переменных модели. Точками обозначены экспериментальные данные из [Feng et al., 2001]

Можно заключить, что качественно и количественно поведение комплексной модели хорошо соответствует лабораторным данным, а также соответствует поведению модели исходной модели [Karaaslan et al., 2005].

Таблица 4.2 – Масштабированные значения параметров во время эксперимента с солевой нагрузкой из [Feng et al., 2001]. Данные в формате “среднее ± стандартная ошибка среднего”

Параметр	**Альдостерон**	**Ренин**	**Натрий**
Ед. измерения	нг/л	отн. к норме	ммоль/л
Экспериментальные данные			
Норма	102.768 ±9.14	1.0085±0.08	143.61±0.4
Высокое потребление натрия	43.929±3.99	0.4016±0.058	144.9398±0.61
Низкое потребление натрия	309.408±23.27	2.67299±0.214	141.867±0.51
Результаты расчетов			
Высокое потребление натрия	65.793	0,38	144,708
Низкое потребление натрия	224.579	2.37981	142.3327

4.6.3. Эксперимент с солевой диетой

Второй эксперимент с комплексной моделью 1 - повторение эксперимента с солевой диетой из [Karaaslan et al., 2005]. Экспериментальные данные взяты из [Barba et al., 1996]. Эксперимент заключается в уменьшении потребления соли примерно на 62.5%. Наблюдаемые переменные – доля реабсорбции в дистальном и проксимальном канальце ($\eta_{dt-sodreb}$ и $\eta_{pt-sodreb}$), абсолютная скорость реабсорбции натрия в дистальном и проксимальном канальце ($\Phi_{dt-sodreb}$ и $\Phi_{pt-sodreb}$), концентрация натрия в крови (C_{sod}) и скорость клубочковой фильтрации (Φ_{gfilt}). Динамика потребления соли определяется следующим уравнением:

$$\Phi_{\text{sodin}} = \begin{cases} 0.126 \text{ ммоль/мин}, & \text{t} < 24, \\ 0.04725 \text{ ммоль/мин}, & \text{иначе.} \end{cases}$$

Наблюдаемые значения (масштабированные, как и в предыдущем эксперименте) и значения, полученные численно, приведены в таблице 4.3 и на рис. 4.13. Можно заключить, что поведение модели качественно соответствует реальным данным, а также соответствует поведению модели исходной модели [Karaaslan et al., 2005].

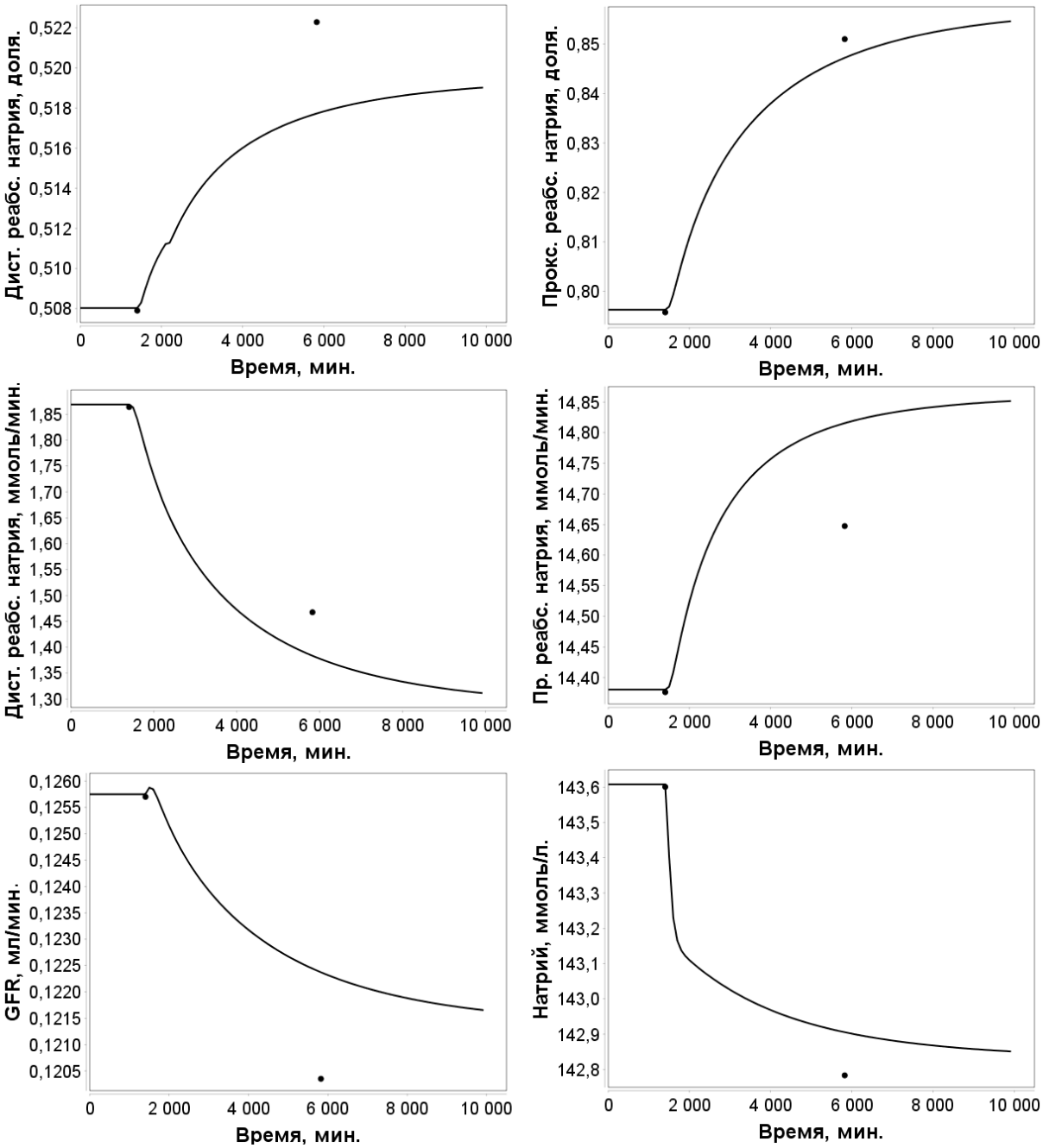


Рисунок 4.13 – Эксперимент с низким потреблением соли. Точками обозначены экспериментальные данные из [Barba et al., 1996]. Пояснения см. в тексте

Таблица 4.3 – Значения параметров, полученных в лаборатории во время эксперимента с низким потреблением соли по [Barba et al., 1996] и рассчитанные численно. Данные в формате “среднее ± стандартная ошибка среднего”

Параметр	$\Phi_{pt-sodreab}$	$\eta_{pt-sodreab}$	$\Phi_{dt-sodreab}$	$\eta_{dt-sodreab}$	C_{sod}	Φ_{gfilt}
Ед. измерения	ммоль/мин	доля от приходящего кол-ва	ммоль/мин	доля от приходящего кол-ва	ммоль/л	мл/мин
Норма	14.38±0.54	0.796±0,008	1.868±0,05	0.508±0,0016	143.6±0.4	0.126±0.003
Диета	14.65±1.1	0.8515±0.008	1.472±0.05	0.522±0.0005	142.8±0.3	0.120±0.003
Диета, численно	14.814	0.846	1.385	0.5176	142.907	0.1224

4.6.4. Эксперимент с физической нагрузкой

Следующий эксперимент проведен с комплексной моделью 2, включающей в себя в модель артериального дерева. Эксперимент заключается в моделировании физической нагрузки путем увеличения потребности организма в кислороде в 4 раза (с 4 мл/с до 16 мл/с) на 50 секунде эксперимента. Общее модельное время расчетов – 2 минуты.

$$\mathrm{Oxygen}_{\mathrm{Need}} = \begin{cases} 4\ \text{мл/с}, & t < 50\ \text{с}, \\ 16\ \text{мл/с}, & \text{иначе}. \end{cases}$$

Результаты расчетов представлены на рис. 4.14, 4.15 и 4.16. Вследствие увеличения мышечной нагрузки падает концентрация кислорода в венозной крови, возрастает сердечный ритм до 96 ударов в минуту, что приводит к увеличению минутного объема почти в полтора раза. Возрастает поток крови через капилляры, что обеспечивает увеличение доставке кислорода, удовлетворяющее повышенный запрос (рис. 4.15). Также существенно возрастает давление – от 120/80 до 150/100, объем желудочка возрастает не столь значительно – примерно на 10 мл. На рис. 4.16. показано изменение давления в отдельных артериях (аорте, подключичной, лучевой и передней большеберцовой артерии) на 40-й, 50-й и 112-й секундах эксперимента. Динамика модели качественно соответствует данным в [Солодянников, 1994, Шумаков др., 1971].

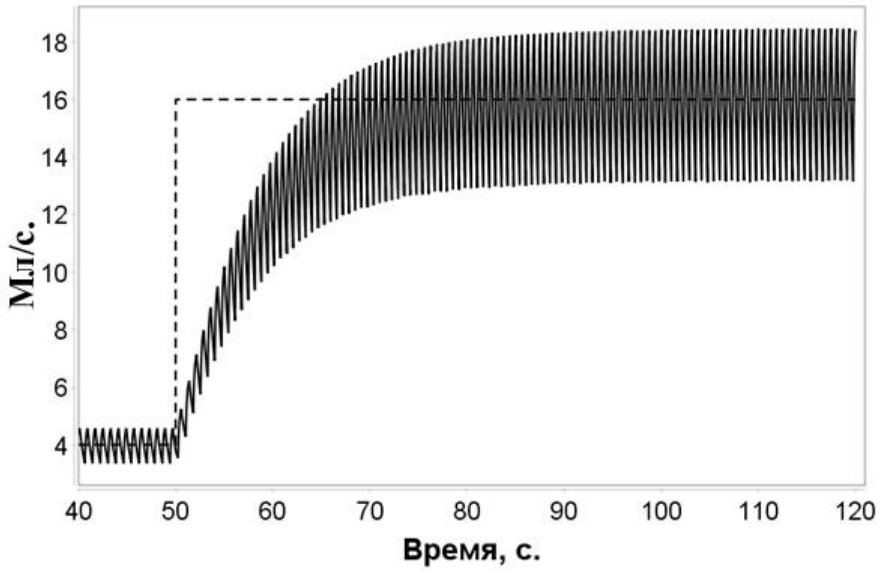


Рисунок 4.14 – Результаты моделирования физической нагрузки. Доставка кислорода (сплошная линия) и потребность в кислороде (пунктир)

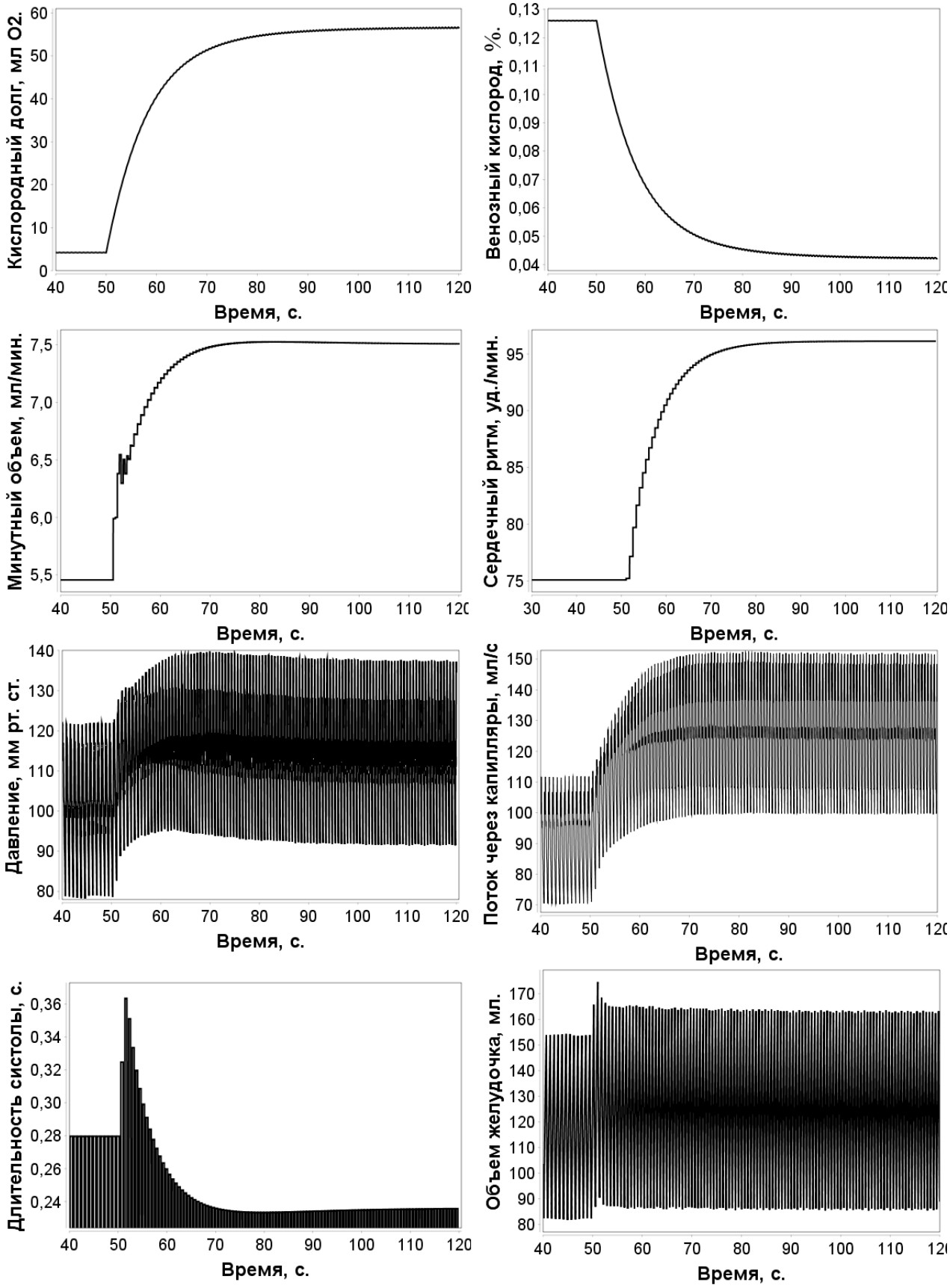


Рисунок 4.15 – Результаты моделирования физической нагрузки

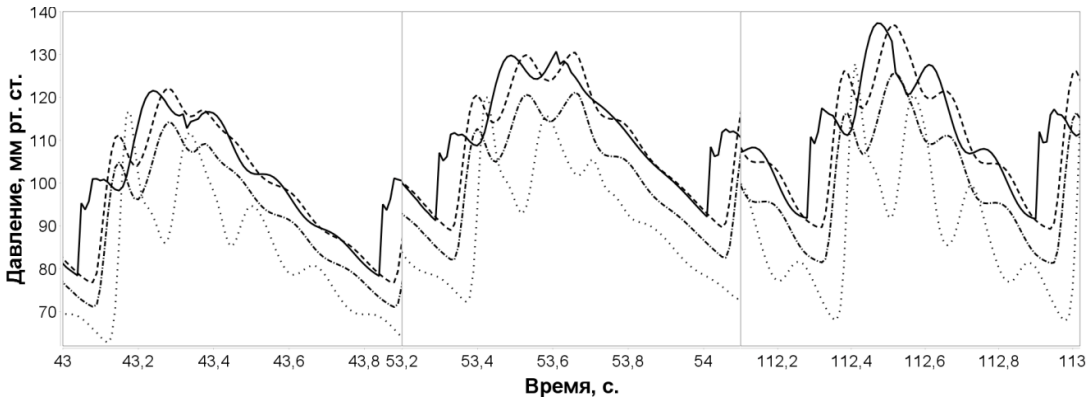


Рисунок 4.16 – Результаты моделирования физической нагрузки. Давление в аорте (сплошная линия), подключичная артерия (пунктир), лучевая артерия (пунктир-точка), передняя большеберцовая артерия (точка)

4.6.5. Эксперимент с пережатием почечных артерий

С комплексной моделью 2 проведем численный эксперимент, демонстрирующий взаимодействие между модулем артериального дерева и другими моделями. Эксперимент заключается в постепенном сужении почечных артерий, площадь сечения каждой из артерий линейно сужалась со 100% до 50% с 5 по 125 минуту эксперимента. Общее модельное время эксперимента – 240 минут. На рис. 4.15 представлены результаты расчетов комплексной модели в сравнении с расчетами модели почечной регуляции. На рис. 4.16 представлены графики изменения потока крови и артериального давления для отдельных сосудов в различных контрольных точках.

Давление в почечной артерии при условии нормального уровня потребления и выведения натрия с мочой – около 100 мм рт. ст. [Bendel et al., 1989]. Начальные значения указанных переменных находятся в допустимых рамках с точки зрения физиологии человека (см. таблицу 4.1).

В ходе эксперимента суммарный поток крови в почечных артериях, пораженных стенозом, уменьшается (рис. 4.17) и это хорошо видно, в частности, на примере правой почечной артерии (рис. 4.18). Давление крови в ССС возрастает, а в указанных артериях через 4 часа достигает значения 130-135 мм рт. ст. в систоле и 90 мм рт. ст. в диастоле, что соответствует симптомам реноваскулярной гипертензии [Bakris, 2014]. Следует отметить, что в модели артериального древа рост давления и потока крови наблюдается также в

остальных артериях ССС человека. Для сравнения на рис. 4.16 также приведены рассчитанные значения для правой общей сонной артерии.

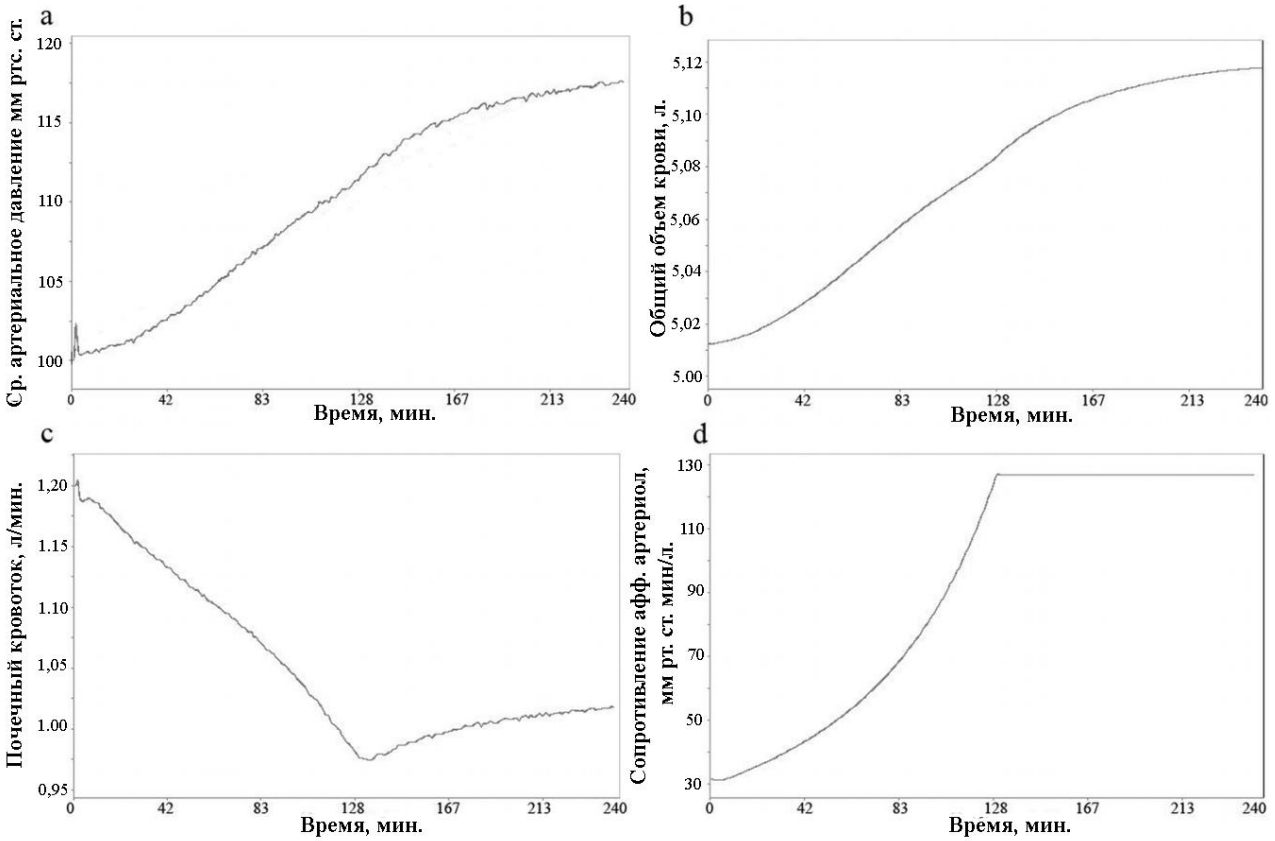


Рисунок 4.17 – Результаты эксперимента с сужением почечных артерий, a – среднее артериальное давление, b – общий объем крови в системе, c – общий почечный кровоток, d – сопротивление афферентных почечных артериол

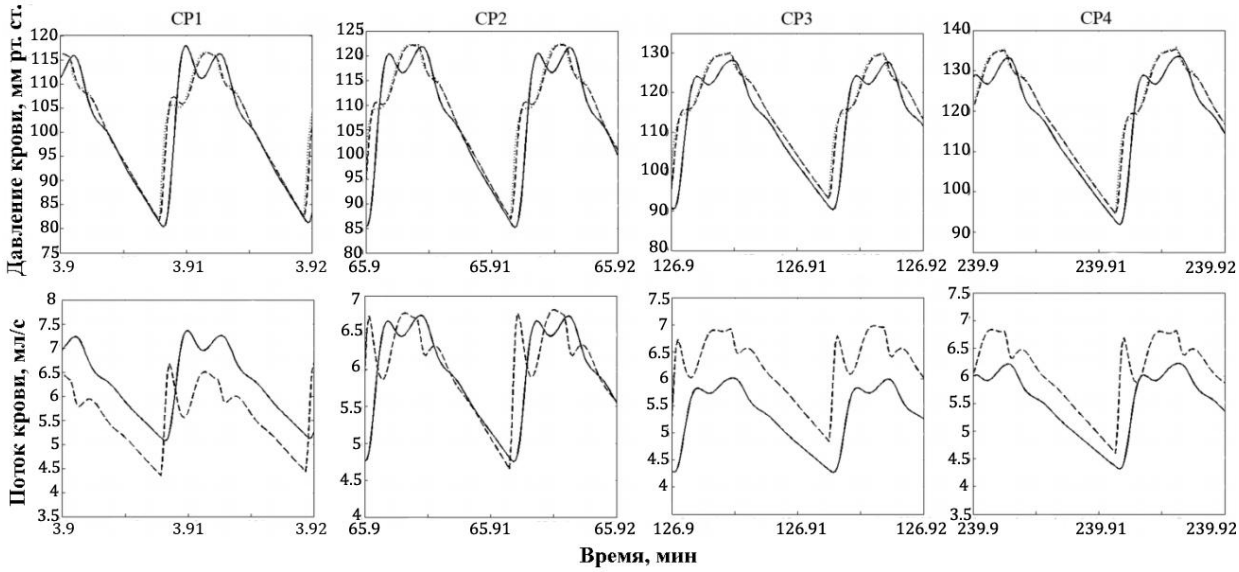


Рисунок 4.18 – Результаты эксперимента с сужением почечных артерий. Давление и поток крови на конце правой почечной артерии (сплошная линия) и в начале правой общей сонной артерии (пунктир)

4.6.6. Персонализация параметров и валидация модели

Результаты данного раздела изложены в работе [Киселев и др., 2015]. Построим процедуру персональной настройки параметров на основе индивидуальных параметров человека, наблюдаемых в условиях клинического обследования. Далее для каждого индивидуального набора параметров проведем валидацию модели путем сравнения вычисленных систолического и диастолического давления с наблюдаемыми. В качестве источника данных используем базу данных [Мельников и др., 2012], содержащую физиологические и клинические данные для 956 человек (после цензурирования). Описание используемых параметров сердечно-сосудистой системы человека из базы данных приведено в Таблице 4.4.

Таблица 4.4. Описание используемых параметров из базы данных [Мельников и др., 2012]

Обозначение	**Описание**	**Ед. измерения**
Индивидуализирующие показатели		
Date	Дата снятия показателей	-
HR	Сердечный ритм	удары/мин
H	Рост	см
W	Вес	кг
Физиологические показатели (каждый для 4 конечностей)		
P_D	Диастолическое артериальное давление	мм рт. ст.
P_S	Систолическое артериальное давление	мм рт. ст.
P_P	Пульсовое давление (разница между P_S и P_D)	мм рт. ст.
r_D	Эффективный диастолический радиус сосуда(радиус просвета)	см
K_{in}	Упругое сопротивление интактной стенки сосуда	дин/ $см^5$
Z_W	Модуль характеристического импеданса	дин с / $см^5$

Описанные параметры организма человека относятся в основном к сосудистому руслу, так что в дальнейшем будем использовать упрощенную модель (рис. 4.19), состоящую из артериального дерева, сердца, генерирующего входной поток в аорту и блока “Капилляры и вены”, устанавливающего граничные условия на концах терминальных сосудов.

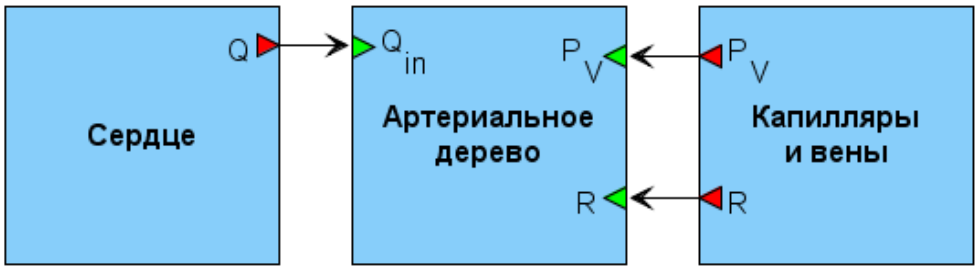


Рис. 4.19 – Комплексная модель 3. Упрощенная комплексная модель

Физиологические параметры (таблица 4.4.) приведены для левой и правой плечевых артерий и обобщенных сосудов в нижних конечностях. Для того чтобы эти данные можно было применить к модели, необходимо добавить в модель плечевую артерию. В оригинальном описании модели (Блохин, 2009) плечевая артерия объединена с подключичной. Радиус объединенного сосуда значительно превосходит типичное значение для плечевой артерии и не может меняться по его длине. Выделим из второго сегмента подключичной артерии (a. subclavia II) последние 15 сантиметров в отдельную артерию a. brachialis. Добавим также ответвление a. profunda brachii, для которого площадь сечения положим равным 30% от площади сечений a. brachialis, основываясь на данных [Avolio, 1980].

В нижних конечностях данные приводятся для обобщенного сосуда, чей объем равен сумме объемов этих трех сосудов (a. tibialis posterior, a. tibialis anterior, a. peronea). В используемой модели артериального дерева отсутствует a. peronea, будем считать модельный сосуд a. tibialis posterior “обобщенным”, объединяющим реальные a. tibialis posterior и a. peronea. Параметр упругости полагался одинаковым для обоих сосудов, в то время как площадь сечения делилась в пропорции 1 к 2, что также соответствует [Avolio, 1980].

Модуль “Сердце”

Для индивидуального задания условия на входе в аорту будем исходить из предположения, что длительность систолы пропорциональна длительности сердечного цикла. Тогда входной поток будем моделировать функцией

$$Q_{in}(t) = \frac{SV}{T_S}(1 + \sin(x(t)), \qquad x(t) = \pi \begin{cases} \dfrac{s(t)}{a} - 0.5, & s(t) < a, \\ \dfrac{s(t) - a}{T_S^* - a} + 0.5, & s(t) < T_S^*, \\ 1.5, & \text{иначе}, \end{cases}$$

где SV – величина сердечного выброса, T_S – длительность систолы, T_C – длительность сердечного цикла, T_C^* – "среднее" значение длительности цикла, T_S^* – "среднее" значение длительности систолы, t – время, $s(t) = T_C^* t \% T_C$ отображает отрезки $[iT_C, (i+1)T_C]$ в $[0, T_C^*]$, где i = 0,1,… , % – операция деления по модулю, функция. Предполагая, что длина систолы пропорциональна длине сердечного цикла, получаем, что объем крови, выброшенный сердцем за один сердечный цикл:

$$\int_{iT_C}^{(i+1)T_C} Q_{in}(t)dt = \frac{T_C}{T_C^*}\int_0^{T_C} Q_{in}(s)ds = \frac{T_C}{T_C^*}\int_0^{T_S} Q_{in}(s)ds = \frac{T_C}{T_C^*}T_S^*\frac{SV}{T_S} = SV, i = 0,1, \ldots.$$

Длительность цикла задается по формуле T_C=60/HR. Параметры a и T_C^* подобраны для соответствия описанной в литературе форме профиля потока (например [Best et al., 1990]). Поток, рассчитанный по этой формуле, приведен на рис. 4.20.

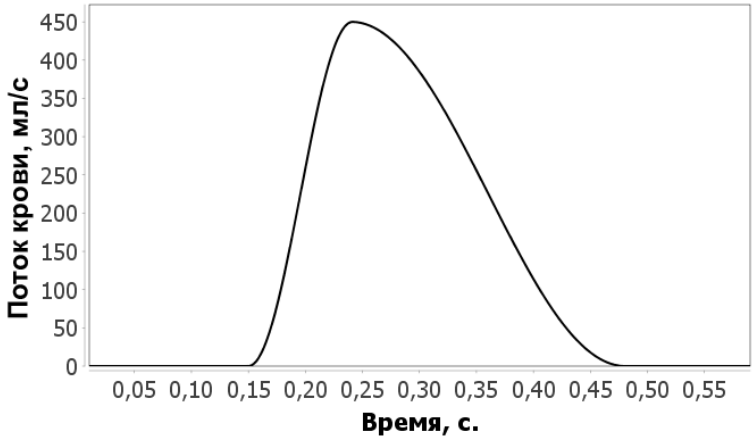


Рис. 4.20. Профиль входного потока в модели артериального дерева

Персонализация входного профиля осуществляется изменением параметров SV и HR. Значения сердечного ритма присутствуют в используемой нами базе данных, однако, значения сердечного выброса необходимо вычислять. Воспользуемся

аллометрическим соотношением для величины минутного объема CO [Simone et al, 1997]: $CO\left(\frac{мл}{мин}\right) = 2499H(м)^{1.16}$. После чего сердечный выброс считается естественным образом: $SV(мл/удар) = \frac{CO(мл/мин)}{HR(удар/мин)}$.

Модуль “Капилляры и вены”

В простейшей форме этот модуль будет просто отдавать значения двух параметров – давления в венах и сопротивляемость капилляров. При этом учтем что артериальное дерево состоит только из крупных артерий, вносящих несущественный вклад в общую периферическую сопротивляемость, основной вклад вносят артериолы и капилляры, следовательно, в качестве сопротивления на концах артерий необходимо установить значение близкое к общей периферической сопротивляемости, ее мы можем вычислить на основе наблюдаемых данных по формуле $R = MAP/CO$, где MAP – среднее давление.

Модуль “Артериальное дерево”

Перейдем к персональной настройке параметров основного блока. Каждый сосуд характеризуется двумя параметрами – площадью сечения в расслабленном состоянии A_0 и характеристикой упругости β. Для того чтобы связать данные из базы с параметрами модели запишем отдельно уравнение состояния для систолического и диастолического давления и добавим уравнения для упругой сопротивляемости K_{in} и модуля характеристического импеданса Z_W [Мажбич Б. И. 1990]. Получим систему из 4 уравнений и трех неизвестных.

$$\begin{cases} P_S = \beta\dfrac{\sqrt{A_S}-\sqrt{A_0}}{A_0}, \\ P_D = \beta\dfrac{\sqrt{A_D}-\sqrt{A_0}}{A_0}, \\ K_{in} = \dfrac{P_S-P_D}{V_S-V_D} \approx \dfrac{P_S-P_D}{l(A_S-A_D)}, \\ Z_W = \sqrt{\dfrac{\rho\beta}{2\sqrt{A_S}}}\dfrac{1}{A_S}, \end{cases}$$

где V_S - объем сосуда в систолу (максимальный); V_D – объем сосуда в диастолу (минимальный); l - длина артерии, ρ - плотность крови. Известные из базы данных параметры: $A_D, P_S, P_D, K_{in}, Z_W$. Неизвестные: A_S, A_0, β.

Поступим следующим образом – будем использовать три первых уравнения для вычисления значений неизвестных. С помощью последнего уравнения рассчитаем значение характеристического импеданса в модели, после чего сравним его со значением из базы. Площадь сечения в диастолу вычисляется естественным образом $A_D = \pi {r_D}^2$. Из третьего уравнения можем выразить площадь сечения сосуда в систолу. Из уравнений состояния для диастолического давления, получаем:

$$\frac{\beta}{A_0} = \frac{P_D}{\sqrt{A_D} - \sqrt{A_0}}.$$

Подставив это в уравнения для систолического давления, имеем:

$$P_S\left(\sqrt{A_D} - \sqrt{A_0}\right) = P_D\left(\sqrt{A_S} - \sqrt{A_0}\right).$$

Таким образом, получаем явные формулы для площади сечения сосуда в расслабленном состоянии и параметра упругости:

$$A_0 = \left(\frac{P_D\sqrt{A_S} - P_S\sqrt{A_D}}{P_S - P_D}\right)^2, \qquad \beta = A_0\frac{P_S - P_D}{\sqrt{A_S} - \sqrt{A_D}}.$$

Модуль характеристического импеданса будем вычислять исходя из полученных значений β и A_S по формуле (4.1). Полученное вычисленное значение Z_W^{Sim} сравним со значением из базы данных. На рис. 4.21. Представлены величины отношения $Z_W^{Ratio} = Z_W^{Sim}/Z_W$ и квадрат ошибки $Z_W^{Diff} = \left(Z_W^{Sim} - Z_W\right)^2$. Видно что величина Z_W^{Ratio} делит всю группу пациентов на две части в зависимости от даты снятия показателей: до или после 1 мая 1999 года. Также видно, что для второй группы вычисленное значение Средняя относительная ошибка расчета импеданса до мая 1999 года – 0,098, после – 0,02. Будем использовать вторую группу как основную для валидации модели, а первую в качестве контрольной.

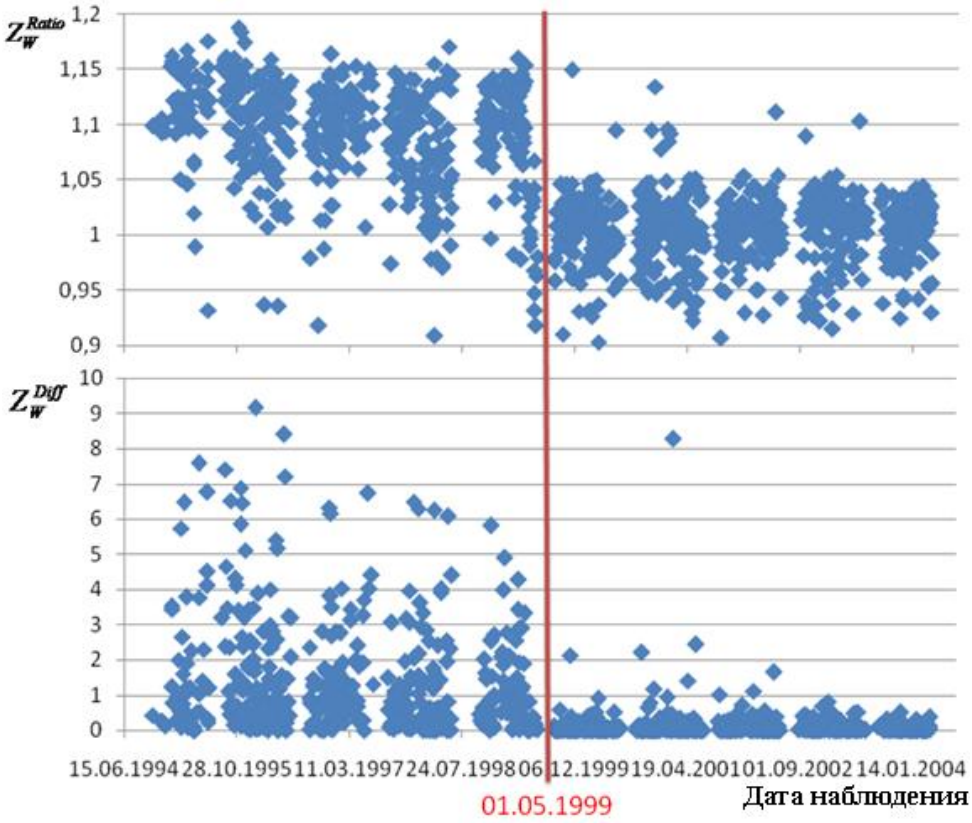


Рис. 4.21 – Сравнение рассчитанного моделя характеристического импеданса по формулам модели со значениями из базы данных [Мельников и др., 2012]

В результате на основе значений из базы данных мы можем вычислить значения параметров модели A_0 и β для четырех сосудов модели. Для того чтобы персонализировать параметры остальных сосудов, будем исходить из предположения, что отношение площадей сечения (и параметров упругости) соответствующих сосудов у двух разных людей – величина близкая к постоянной. В частности – разобьем артериальное дерево на две половины: верхнюю и нижнюю (по точке ветвления между IV и V сегментами артерии a. abdominalis). Будем предполагать, что если плечевая артерия у одного человека в два раза толще (или в два раза более упругая) чем у другого, это верно и для всех остальных сосудов верхней половины дерева. То же самое верно для нижней с заменой плечевой артерии на a. posterior tibialis. Таким образом, если мы знаем распределение параметров по сосудам для одного человека и параметры определенных сосудов другого человека – мы можем восстановить параметры всех остальных его сосудов.

Введем в модель параметры $A_{F,up}$, $A_{F,down}$, $\beta_{F,up}$, $\beta_{F,down}$. Перед началом расчетов параметры A_0 и β всех сосудов модели будут умножаться на эти множители. Для произвольного человека

$$A_{F,up} = \frac{A_{0,h}}{A^*_{0,h}}, \qquad A_{F,down} = \frac{A_{0,l}}{A^*_{0,l}}, \qquad \beta_{F,up} = \frac{\beta_h}{\beta^*_h}, \qquad \beta_{F,down} = \frac{\beta_l}{\beta^*_l},$$

где h - индекс левой плечевой артерии, l - индекс левой артерии a. tibialis, * обозначены значении соответствующих параметров по умолчанию.

Окончательно для персональной настройки параметров модели мы используем 7 параметров: $HR, SV, R, A_{F,up}, A_{F,down}, \beta_{F,up}$ и $\beta_{F,down}$ и их различные комбинации. Общая схема приведена на рис. 4.22. Для оценки адекватности работы модели будем использовать качество предсказания систолического и диастолического давлений для левой руки. Результаты расчетов для различных комбинаций параметров приведены в таблице 4.5. Сравнение с результатами, полученными для контрольной группы приведено в таблице 4.6.

Результаты расчетов, приведенные в таблице 4.6, показывают, что один единственный параметр, а именно, сопротивление капилляров, в состоянии объяснить до 70% дисперсии систолического и диастолического давления в рассматриваемой группе (коэффициенты корреляции 0,838 и 0,810, соответственно). Лучший результат предсказания давлений получается при персонализация периферического сопротивления, сердечного ритма и сердечного выброса (корреляции для систолического и диастолического давления 0,897 и 0,925, соответственно). Учет этих параметров наряду с сопротивлением значительно уменьшает среднюю относительную ошибку предсказания систолического давления с 0,164 до 0,041 и диастолического давления с 0,425 до 0,164. Параметры крупных сосудов A_0 и β, включенных в модель, не играют существенной роли в предсказании давлений. Пульсовое давление при данной схеме персональной настройки параметров адекватно предсказать не удалось.

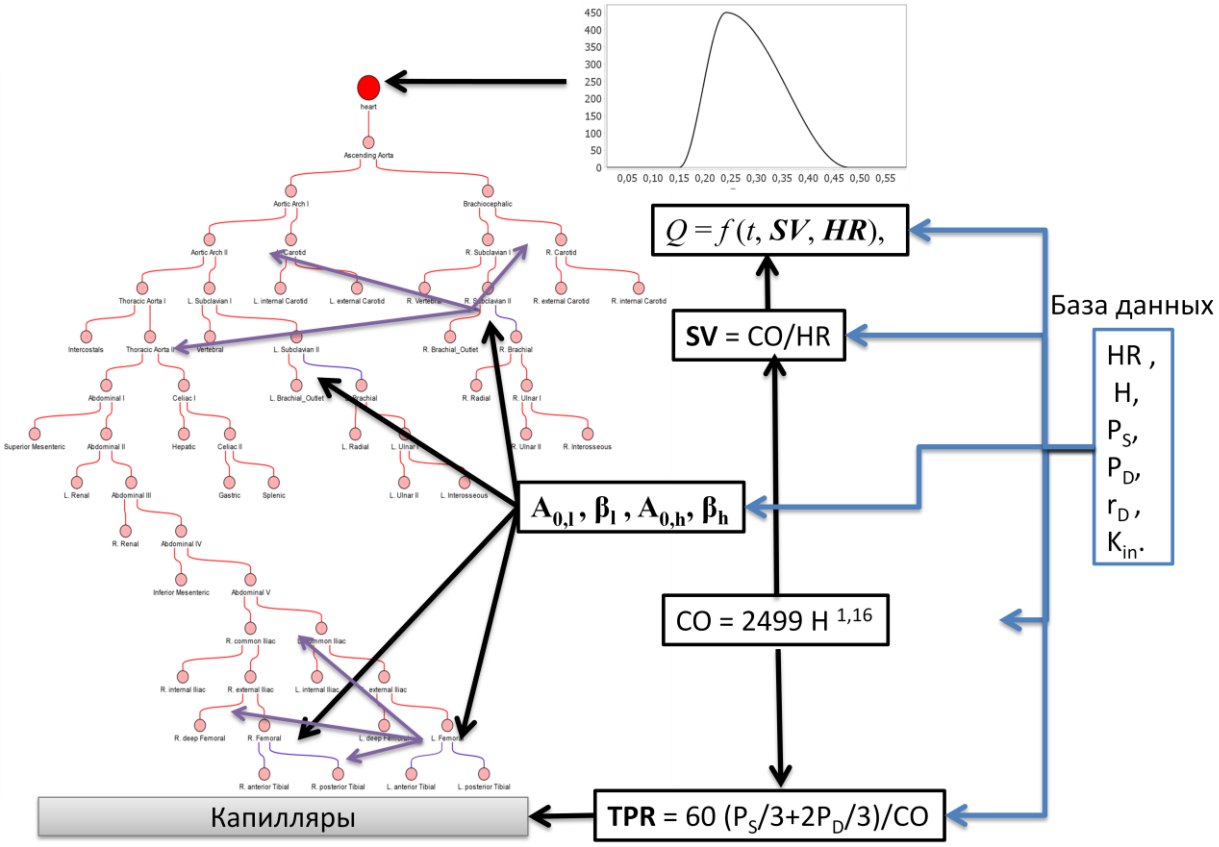


Рис. 4.22 – Схема персонализации параметров модели

Таблица 4.5. Относительные ошибки и корреляции между рассчитанными и реальными значениями давлений (систолического, диастолического и пульсового) в левой руке (a. brachialis dextra). Индексом *** обозначены корреляции со значением P-value< 0.001. Индексом * обозначены корреляции со значением P-value< 0.05. Общее количество – 556 человек

Параметры	Коэффициент корреляции			Средняя относительная ошибка		
	Ps	Pd	Pp	Ps	Pd	Pp
С использованием параметра периферического сопротивления R						
R A_0 β SV HR	0,784***	0,864***	0,011	0,055	0,208	0,372
R SV HR	0,897***	0,925***	-0,070	0,041	0,164	0,283
R HR	0,484***	0,556***	-0,318*	0,169	0,391	0,214
R SV	0,422***	0,530***	-0,075	0,119	0,249	0,283
R A_0β	0,693***	0,781***	0,006	0,158	0,472	0,328
R A_0	0,569***	0,838***	-0,125*	0,209	0,385	0,285
R β	0,806***	0,759***	0,085*	0,123	0,486	0,400
R	0,810***	0,838***	-0,401***	0,164	0,425	0,210
Без использованием параметра периферического сопротивления R						
A_0 β SV HR	0,007	-0,162***	0,046	0,171	0,126	0,342
SV HR	-0,060	-0,012	0,014	0,165	0,129	0,253
HR	0,034	0,077	-0,020	0,122	0,189	0,177
SV	-0,035	-0,073	0,019	0,167	0,161	0,252
A_0 β	0,080	-0,226	0,051	0,092	0,181	0,301
A_0	-0,145***	0,150***	-0,092*	0,113	0,120	0,283
β	0,294***	-0,297***	0,137*	0,086	0,195	0,365

Таблица 4.6. Результаты расчетов с разделением на две группы по времени снятия показателей: до мая 1999 года и после. В расчетах использовались все параметры: R, A_0, β, SV, HR. Индексом *** обозначены корреляции со значением P-value< 0.001. Индексом * обозначены корреляции со значением P-value< 0.05

Время снятия показателей	**#**	**Коэффициент корреляции**			**Средняя относительная ошибка**		
		Ps	**Pd**	**Pp**	**Ps**	**Pd**	**Pp**
До мая 1999 г.	420	0,733***	0,801***	0,111*	0,073	0,199	0,335
После мая 1999 г.	556	0,784***	0,864***	0,011	0,055	0,208	0,372
Вся выборка	976	0,754***	0,836***	0,098	0,062	0,205	0,356

4.7. Выводы по главе 4

В рамках данной работы были реализованы модели сердечно-сосудистой системы человека [Guyton et al., 1972], [Солодянников, 1994] и [Karaaslan et al., 1995] в виде модульных диаграмм в платформе BioUML.

На основе данных моделей, а также модели артериальной системы [Блохин, 2009] были созданы комплексные модели ССС человека в трех вариантах:

1. Комплексная модель 1, включающая блоки моделей [Солодянников, 1994] и [Karaaslan et al., 1995].
2. Комплексная модель 2, объединяющая комплексную модель 1 и модель артериального дерева.
3. Комплексная модель 3, содержащая модель артериального дерева и упрощенные блоки сердца, вен и капилляров.

Комплексная модель 1 была протестирована в состояниях солевой нагрузки и солевой диеты. Эти эксперименты рассчитаны на большое модельное время (дни и недели), что не позволяет задействовать модель артериального дерева.

Комплексная модель 2 была протестирована в норме, состояниях физической нагрузки и пережатии почечной артерии.

С комплексной моделью 3 была осуществлена процедура персональной настройки параметров и валидации на данных [Мельников и др., 2012], в основном относящихся к параметрам сосудов.

Можно заключить, что созданные модели показывают результаты, согласующиеся с результатами, полученными с помощью исходных моделей [Karaaslan et al., 2005] и [Солодянников, 1994], с рядом данных приведенных в

литературе, а также с физиологическими данными, полученными при обследовании реальных людей [Мельников и др., 2012].

Дальнейшее развитие модели, выходящее за рамки данной работы, будет идти по следующим путям:

1. Настройка параметров и валидация модели на основе данных по форме и скорости пульсовой волны.
2. Усовершенствование и усложнение отдельных модулей.
3. Установление новых связей между модулями, например, учет влияния гуморальной регуляции на состояние сосудов артериального дерева.
4. Добавление новых модулей, описывающих малый круг кровообращения, легкие, печень, поджелудочную железу и т.д. вплоть до наиболее полной модели человеческого организма. Этот процесс значительно облегчается модульной структурой модели.
5. Идентификация параметров модели по индивидуальным параметрам конкретного организма (организмов), создание виртуальной популяции для моделирования действия различных лекарственных препаратов.

Таким образом, предложенный подход построения моделей был успешно продемонстрирован при создании модульных моделей ССС. Разбиение исходных моделей на составные части – модули – позволяет их группировать в новые комплексные модели. При этом в зависимости от стоящей задачи могут быть использованы различные комбинации модулей.

ЗАКЛЮЧЕНИЕ

В качестве основы в работе используется платформа для формального описания биологических систем BioUML [Колпаков, 2011]. В рамках данной работы, возможности BioUML были расширены – добавлен тип диаграмм “Математическая модель”, существенно доработан тип диаграмм “Модель артериального дерева”. Расширена обработка мгновенных событий в соответствии с последними версиями спецификации языка SBML [Hucka et al., 2010] (добавлены такие свойства как устойчивость, приоритет, начальное значение и способ выполнения событий, добавлена поддержка каскадов событий) и портирован с языка C и адаптирован к API BioUML численный решатель CVODE [Hindmarsh et al., 2005].

В работе разработан подход к моделированию биологических систем, основанный на представлении системы в виде набора взаимосвязанных подсистем, каждая из которых моделируется отдельно. Модель всей системы создается как набор взаимосвязанных подмоделей (модулей).

В случае, когда все модули содержат системы ОДУ с мгновенными событиями, модульная модель может быть трансформирована в не-модульную модель с тем же математическим формализмом. Для таких моделей в BioUML были созданы новые типы диаграмм “Модульная модель” и “Модульная SBML-модель”.

Разработаны алгоритмы трансформации таких диаграмм в формат SBML с расширением для модульных моделей [Smith et al., 2013].

Создан алгоритм численных расчетов для модульной модели в случае различного математического формализма модулей. Алгоритм использует принципы агентного моделирования.

Описанные алгоритмы реализованы в виде программных модулей к платформе BioUML и предоставляют средства для визуального создания и численных расчетов моделей биологических систем широкого класса – моделей клеточных процессов, моделей подсистем живых организмов, агентных моделей.

Созданное программное обеспечение протестировано:

1. Набором тестов для SBML -моделей (все 1126 тестов пройдены).
2. Набором тестов для иерархических SBML-моделей (все 60 тестов пройдены).
3. Созданием модульной модели ССС человека, состоящей из трех различных ранее существовавших моделей:
   - модели с сокращающимся сердцем (ОДУ с мгновенными событиями) [Солодянников, 1994];
   - модели почечной регуляции (ОДУ) [Karaaslan et al., 2005];
   - модели артериального дерева (УЧП) [Блохин и др. 2009].

Комплексная модель создана в трех вариантах – содержащая первые две модели, содержащая все три модели и упрощенная, содержащая артериальное дерево и более простые блоки сердца и капилляров.

Созданные комплексные модели протестированы путем сравнения с данными в литературе, проведением экспериментов с солевой нагрузкой и диетой [Barba et al., 1996; Feng et al., 2001], пережатием почечной артерии, физической нагрузки [Солодянников, 1994].

Была разработана и реализована процедура персональной настройки параметров упрощенного варианта комплексной модели на основе базы данных [Мельников и др., 2012], была осуществлена валидация модели путем сравнения систолического и диастолического давления при разных вариантах настраиваемых параметров.

Благодаря своей структуре, модель является расширяемой – в неё могут быть добавлены новые модули, описывающие другие системы организма: малый круг кровообращения, почка, легкие, поджелудочная железа и т.д. Таким образом, модель может служить основой для создания глобальной модели физиологии человека.

По диссертационной работе можно сформулировать следующие **выводы**.

1. Формализовано понятие модульных моделей биологических систем, разработана графическая нотация для их визуального представления.

2. Разработаны алгоритмы для проведения численных расчетов в модульных моделях. В случае с одинаковым формализмом модулей (ОДУ с мгновенными событиями) – алгоритм генерации плоской модели, в общем случае – на основе принципов агентного моделирования. Показана корректность работы алгоритмов на наборе тестов для модульных SBML-моделей.
3. Разработаны алгоритмы трансформации модульных моделей в формат SBML и обратно.
4. Разработан программный модуль для платформы BioUML, позволяющий создавать модульные модели и проводить численные расчеты. Возможности численных расчетов BioUML расширены портированием численного решателя CVODE и расширением поддержки событий в соответствии со спецификацией SBML (уровень 3 версия 1). Показана корректность численных расчетов на наборе тестов для SBML-моделей.
5. Создана новая модульная модель сердечно-сосудистой системы человека, включающая сердце, артериальное дерево из 55 крупнейших сосудов и регуляцию водно-солевого баланса почкой. Показана адекватность модели на ряде компьютерных экспериментов: диета с повышенным и пониженным потреблением соли, физическая нагрузка, пережатие артерий.
6. Для персональной настройки параметров модели сердечно-сосудистой системы по экспериментальным данным (база данных "Показатели эластичности артерий и гемодинамики у здоровых и больных людей", 976 человек) разработана ее упрощенная версия. Показано, что такая модель адекватно предсказывает систолическое и диастолическое давление.
7. Разработанное программное обеспечение внедрено и используется для создания математических моделей биологических систем в Институте математики им. Соболева СО РАН и Институте общей генетики им. Вавилова РАН (см. Приложение Д).

СПИСОК СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

Сокращения

ОДУ – обыкновенные дифференциальные уравнения.

ССС – сердечно-сосудистая система.

УЧП – уравнения с частными производными.

API – Application Programming Interface (интерфейс программирования приложений).

BioUML – Biological Universal Modeling Language (универсальный язык моделирования в биологии).

CellML – Cell markup language (язык разметки биологической клетки).

SBGN – Systems Biology Graphical Notation (графическая нотация системной биологии).

SBML – Systems Biology Markup Language (язык разметки системной биологии).

SBO – Systems Biology Ontology (Онтология системной биологии).

UML – Universal Modeling Language (универсальный язык моделирования).

URI - Uniform Resource Identifier (унифицированный идентификатор ресурса).

XML – eXtensible Markup Language (расширяемый язык разметки).

Обозначения

$\mathbb{R}$	Множество вещественных чисел.
$\mathbb{R}^n$	Вещественное векторное пространство размерности n.
$\mathbb{R}_+$	Положительные вещественные числа, включая 0.
$\emptyset$	Пустое множество.
$\overline{1,n}$	Диапазон целых чисел от 1 до n (n>1).
$\wedge$	Бинарный оператор логического умножения.
$\vee$	Бинарный оператор логического сложения.
$\sim$	Унарный оператор логического отрицания.
$\times$	Декартово произведение множеств.
$\bar{0}$	Нулевой вектор.
%	Деление по модулю.

СПИСОК ЛИТЕРАТУРЫ

1. Абакумов М. В., Гаврилюк К. В, Есикова Н. Б., Кошелев В. Б., Лукашин А. В, Мухин С. И., и др. Математическая модель гемодинамики сердечно-сосудистой системы // Дифференциальные уравнения. 1997. Т. 33 № 7. С. 892-898.
2. Амосов Н. М., Лищук С. А., Пацкина С. А. Саморегуляция сердца. Киев: Наукова Думка. 1969.160 с.
3. Астраханцева Е. В. Гидаспов В. Ю., Ревизников Д. Л. Математическое моделирование гемодинамики крупных сосудов // Математическое моделирование, 2005. Т. 17 № 8. С. 61-80.
4. Блохин А. М., Трахинин Ю. Л., Бибердорф Э. А., Попова Н. И. Глобальное моделирование артериальной системы человека // Иванова Л. Н., Блохин А. М., Маркель А. Л. Система кровообращения и артериальная гипертония: биофизические и генетико-физиологические механизмы, математическое и компьютерное моделирование: Моногр. Новосибирск: Сиб. отд-ние РАН, 2009 (сер. Интеграционные проекты. Вып. 17). С. 106-134.
5. Борзов А. Г., Древаль А. В., Мухин С. И. Моделирование динамики глюкозы крови с учетом топологии большого круга кровообращения // Математическое моделирование, 2015, Т. 27 № 2. С. 3-24.
6. Варфоломеев С. Д., Гуревич К. Г. Биокинетика: Практический Курс. М.: ФАИР-ПРЕСС. 1999. 720 с.
7. Григорян Р. Д. Математическая модель сердечно-сосудистой системы человека // Биологическая медицинская кибернетика и бионика. Киев. 1984. С. 17-21.
8. Евшин И. С., Леонова Т. И., Семисалов Б. В., Колпаков Ф. А., Шарипов Р. Н. Компьютерное моделирование системы кровообращения // под ред. Иванова Л. Н., А.М. Блохин А. М., Маркель А. Л. Система кровообращения и артериальная гипертония: биофизические и генетико-физиологические механизмы, математическое и компьютерное моделирование: Моногр.

Новосибирск: Сиб. отд-ние РАН, 2009 (сер. Интеграционные проекты. Вып. 17). С. 106-134.

9. Карпов Ю. Г. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. СПб: БХВ-Петербург. 2006. 400 с.
10. Киселев И. Н. Реконструкция комплексной модели регуляции кровообращения у человека (по Guyton et al., 1972) с применением композиционного подхода в системе BioUML. Международная конференция “Современные проблемы математики, информатики и биоинформатики”, посвященная 100-летию со дня рождения члена-корреспондента АН СССР А.А. Ляпунова. С. 75.
11. Киселев И. Н., Бибердорф Э. А., Баранов В. И., Комлягина Т. Г., Мельников В. Н. Суворова И. Ю., Кривощеков С. Г., Колпаков Ф. А. Персонализация параметров и валидация модели сердечно-сосудистой системы человека // Математическая биология и биоинформатика. 2015. Т. 10 № 2. С. 526-547.
12. Киселев И. Н., Семисалов Б. В., Бибердорф Э. А., Леонова Т. И., Шарипов Р. Н., Блохин А. М., и др. Агентное моделирование сердечно-сосудистой системы человека. Международная конференция “Современные проблемы математики, информатики и биоинформатики”, посвященная 100-летию со дня рождения члена-корреспондента АН СССР А.А. Ляпунова. 2011. С.
13. Киселев И. Н, Семисалов Б. В, Бибердорф Э. А., Шарипов Р. Н., Блохин А. М., Колпаков Ф. А. Модульное моделирование сердечно-сосудистой системы человек // Математическая биология и биоинформатика. 2012. Т. **7** № 2. С. 703–736.
14. Колпаков Ф. А., Шарипов Р. Н., Евшин И. С., Леонова Т. И., Семисалов Б. В. Компьютерное моделирование системы кровообращения // под ред. Иванова Л. Н., Блохин А. М., Маркель А. Л. Система кровообращения и артериальная гипертония: биофизические и генетико-физиологические механизмы, математическое и компьютерное моделирование: Моногр. Новосибирск: Сиб. отд-ние РАН. 2009 (сер. Интеграционные проекты. Вып. 17). С. 135-204.

15. Колпаков Ф. А. Формальное описание и визуальное моделирование биологических систем. Дис. … канд. биол. наук. Москва. 2011.
16. Кошелев В. Б. Мухин С. И., Соснин Н. В., Фаворский А. П. Математические модели квази-одномерной гемодинамики: Методическое пособие. М. МАКС Пресс. 2010. 114 с.
17. Кутумова Е. О. Модульная модель процесса программируемой гибели клеток: построение, верификация и упрощение. Дис. ... канд. физ. мат. наук. Москва. 2012.
18. Литвинов А. В. Норма в медицинской практике: Справочное пособие. М.:МЕДпресс-информ. 2004. 144 с.
19. Лищук В. А. Математическая теория кровообращения. М.: Медицина. 1991. 256 с.
20. Мажбич Б. И. Осцилловазометрия артериальных сосудов конечностей. Новосибирск: Наука. Сиб. отд-ние. 1990. 152 с.
21. Мельников В. Н., Комлягина Т. Г., Речкина С. Ю., Лазарева И. Ф., Суворова И. Ю., Кривощеков С. Г. Показатели эластичности артерий и гемодинамики у здоровых и больных людей [База данных]. Зарегистрирована в Реестре баз данных Федеральной службы по интеллектуальной собственности 9 июня 2012 г. № 2012620540.
22. Моделирование кровообращения. URL: http://www.samara-dialog.ru/help/rus/help.htm (дата обращения 21.09.2016).
23. Педли Т. Гидродинамика крупных кровеносных сосудов. М.: Мир. 1983. 400 с.
24. Прошин А. П., Солодянников Ю. В. Математическое моделирование системы кровообращения и его практические применения // Автоматика и телемеханика. 2006. Т. 2. С. 174-188.
25. Регирер С. А., Скобелева И. М. Течение вязкой жидкости в трубке с деформирующейся стенкой // Механика жидкости и газа. 1971. Т. 3. С. 111-122.

26. Семисалов Б. В. Построение и анализ комплексной модели сердечно-сосудистой системы человека, включая биофизические и биохимические блоки. Вестн. Новосиб. гос. ун-та. Серия: Математика, механика, информатика. 2010. Т. 10 № 1. С. 95-107
27. Солодянников Ю. В. Элементы математического моделирования и идентификация системы кровообращения. Самара: Изд-во Самар. ун-та. 1994. 316 с.
28. Функциональные методы исследования почек. URL: http://www.medical-enc.ru/m/15/funktsionalnye-metody-issledovania-pochek-3.shtml (дата обращения 21.09.2016).
29. Хайрер Э., Ваннер Г. Решение обыкновенных дифференциальных уравнений. Жесткие и дифференциально-алгебраические задачи. М.: Мир. 1999. 685 с.
30. Хакимзянов Г. С., Черный С. Г. Методы вычислений: В 4 ч. Ч. 1: Численные методы решения задачи Коши для обыкновенных дифференциальных уравнений. Новосибирск: НГУ, 2003. 106 с.
31. Шумаков В. И., Сахаров М. П., Толепкин В. Е. Изучение некоторых параметров работы искусственных желудочков в организме с помощью электрической аналоговой модели // Биофизика. 1969. Т. 6. С. 15-21.
32. Шумаков В. И., Новосельцев В. Н., Сахаров М. П., Штейнгольд Е. Ш. Моделирование физиологических систем организма. М: Медицина. 1971. 352 с.
33. Шумаков В. И. Иткин Г. П. Егоров Г. Л. Штейгольд. Е. Ш. Исследование шунтирования левого желудочка сердца в условиях сердечной недостаточности // Scuipta medica. 1975. Т. 48. С. 175-480.
34. Abram S. R., Hodnett B. L., Summers R. L., Coleman T. G., Hester R. L., Quantitative Circulatory Physiology: an integrative mathematical model of human physiology for medical education // Advances in Physiology Education. 2007. V. 31 PP. 202-210.
35. Aldosterone in Blood. URL: www.webmd.com/a-to-z-guides/aldosterone (дата обращения 21.09.2016).

36. Avolio A. P. Multi-branched model of the human arterial system // Medical & Biological Engineering & Computing. 1980. V. 18. PP. 709-718.
37. Bakris G. L. Renovascular Hypertension. 2014. URL: http://www.merckmanuals.com/professional/cardiovascular-disorders/hypertension/renovascular-hypertension (дата обращения 21.09.2016).
38. Baranov V. I., Biberdorf E. A., Kiselev I. N., Kolpakov F. A., Komlyagina T. G., Krivoshchekov S. G., Melnikov V. N., Suvorova I. Yu. Patient-specific 1D model of the human cardiovascular system // The Siberian Scientific Medical Journal. 2016 V. 36. PP. 70-79.
39. Barba G., Cappuccio F. P., Russo L., Stinga F., Iacone R., and Strazzullo P. Renal function and blood pressure response to dietary salt restriction in normotensive men // Hypertension. 1996. V.27 PP. 1160-1164.
40. Bassingthwaighte J. B., Qian H., Zheng L. Cardiome project. An Integrated View of Cardiac Metabolism and Regional Mechanical Function // Advances in Experimental Medicine and Biology. 1999. V. 471. PP. 541-553.
41. Bassingthwaighte J. B., Raymon G. M., Butterworth E., Alessio A., Caldwell J. H. Multiscale modeling of metabolism, flows and exchanges in heterogeneous organs // Annals of the New York Academy of Sciences. 2010. V. 1188. PP. 111-120.
42. Функциональные методы исследования почек. URL: http://www.medical-enc.ru/m/15/funktsionalnye-metody-issledovania-pochek-3.shtml (дата обращения 02.13.2014).
43. Beanexplorer. URL: www.beanexplorer.org (дата обращения 21.09.2016).
44. Bellifemine F., Caire G., Poggi A., Rimassa G. JADE: A White Paper // EXP in search of innovation. 2003. V. 3. №. 3. PP. 6-19.
45. Bendel P., Buonocore E., Bockisch A., Besozzi M.C. Blood flow in the carotid arteries: quantification by using phase-sensitive MR imaging // American Journal of Roentgenology. 1989. V. 152. № 6. PP. 1307-1310.
46. Best C. H., Taylor N. B., West J. B., BEST & TAYLOR'S Physiological Basis of Medical Practice. Williams & Wilkins. 1990.

47. Biberdorf E. A., Blokhin A. M., Trakhinin Y. L. Global modeling of the human arterial system. Circulatory System and Arterial Hypertension: Experimental Investigation, Mathematical and Computer Simulation / Eds. Ivanova A. L., Markel A. M., Blokhin E. V., Mishchenko. Nova Science Publishers, Inc. 2012. PP. 115-142.
48. Bonabeau E. Agent-based modeling: Methods and techniques for simulating human systems // PNAS. 2002. V. 99. № 3. PP. 7280-7287.
49. Cain P.A., Ahl R., Hedstrom E., Ugander M., Allansdotter-Johnsson A., Friberg P., Arheden H. Age and gender specific normal values of left ventricular mass colume and function for gradient echo magnetic resonance imaging a cross sectional study // BMC Medical Imaging. 2009. V. 9. №. 2.
50. Campen D. H., Huyghe J. M. R. J., Bovendeerd P. H. M., Arts M. G. J. Biomechanics of the heart muscle // European Journal of Mechanics. A. Solids. 1994. V. 13. PP.19-41.
51. Catt K. J., Cain M. D., Zimmet P. Z., Cran, E. Blood Angiotensin II Levels of Normal and Hypertensive Subjects // British Medical Journal. 1969. V. 1. № 5647. PP. 819-821.
52. Chandran D., Bergmann F. T., Sauro H. M. TinkerCell: modular CAD tool for synthetic biology // Journal of Biological Engineering. 2009. V. 3. № 19.
53. Cohen S.D., Hindmarsh A.C. CVODE, A Stiff/Nonstiff ODE Solver in C // Computers in Physics. 1996. V. 10. № 2. PP. 138-143.
54. Coleman T. G., Hall J. E. A mathematical model of renal hemodynamics and excretory function Structuring Biological Systems: A Computer Modelling Approach ed. by S. S. Iyengar // CRC Press. 1992. PP. 89-124.
55. Cooling M. T., Rouilly V., Misirli G., Lawson Ju T., Halinan J., Wipat A. Standard virtual biological parts: a repository for modeling components for synthetic biology // Bioinformatics. 2010. V. 26. № 7. PP. 925-931.
56. Courtot M., Juty N., Knüpfer C., Waltemath D., Zhukova A., Dräger A., et al. Controlled vocabularies and semantics in systems biology // Molecular Systems Biology. 2011. V. 7. № 543.

57. Cowley A. W. Jr. Role of the renal medulla in volume and arterial pressure regulation // American Journal of Physiology. 1997. V. 273. PP. R1-15.
58. Chan N. T., LeBaron B., Lo A. W., Poggio T. Agent-Based Models of Financial Markets: A Comparison with Experimental Markets. MIT Sloan Working Paper. 1999.
59. Cuellar A., Nielsen P., Halstead M., Bullivant D., Nickerson D., Hedley W., et al. CellML 1.1 Specification. URL: http://www.cellml.org/specifications/cellml_1.1 (дата обращения 15.10.2013).
60. Delanaye P., Schaeffner E., Ebert N., Cavalier E., Mariat C., Krzesinski J., et al. Normal reference values for glomerular filtration rate: What do we really know? // Nephrology Dialysis Transplantation. 2012. V. 27. PP. 2664–2672.
61. Di Ventura B., Lemerle C., Michalodimitrakis K., Serrano L. From in vivo to in silico biology and back // Nature. 2006 V. 443 № 5. PP. 527-533.
62. Dormand J. R., Prince P. J. A family of embedded Runge-Kutta formulae // Journal of Computational and Applied Mathematics. 1980. V. 6. № 1. PP. 19–26.
63. Elert G. Volume of Blood in a Human. The Physics Factbook. URL: http://hypertextbook.com/facts/1998/LanNaLee.shtml (дата обращения 21.09.2016).
64. Feng J. H., Markondu N. D., MacGregor G. A. Importance of the renin system for determining blood pressure fall with acute salt restriction in hypertensive and normotensive whites // Hypertension. 2001. V. 38. PP. 321-325.
65. Fenner J. W., Brook B., Clapworthy G., Coveney P. V., Feipel V., Gregersen H., et al. The EuroPhysiome, STEP and a roadmap for the virtual physiological human regulation // Philosophical Transactions of the Royal Society. 2008. V. 366. PP. 2979-2999.
66. Fowler M. Dealing with properties. 1997. URL: http://martinfowler.com/apsupp/properties.pdf (дата обращения 21.09.2012).
67. Garny A., Nickerson P. D., Cooper J., Weber dos Santos R., Miller A.K., McKeever S., et al. CellML and associated tools and techniques // Philosophical Transactions of the Royal Society. 2008. A 366. PP. 3017-3043.

68. Garny A., Noble D., Hunter P. J., Kohl P. Cellular Open Resource (COR): current status and future directions regulation // Philosophical Transactions of the Royal Society. 2009. V. 367. № 1895. PP. 1885-1905.
69. Gear C. W. The Numerical Integration of Ordinary Differential Equations // Mathematics of Computation. 1967. V. 21. PP. 146-156.
70. Gibson M. A., Bruck J. Efficient Stochastic Simulation of Chemical Systems with Many Species and Many Channels // The Journal of Physical Chemistry. 2000. V. 104. PP. 1876-1889.
71. Gillespie D. T. Stochastic Simulation of Chemical Kinetics // Annual Review of Physical Chemistry. 2007. V. 58. PP. 35–55.
72. Ginkel M., Kremling A., Nutsch T., Rehner R., Gilles E.D. Modular modeling of cellular systems with PROMOT/DIVA // Bioinformatics. 2003. V. 10. № 9. PP. 1169-1176.
73. Guyton A. C., Coleman T. G., Granger H. J. Circulation: Overall regulation // Annual Review of Physiology. 1972. V. 34. PP. 13-46.
74. Guyton A. C. The surprising kidney-fluid mechanism for pressure control: its infinite gain! // Hypertension. 1990. V. 16. PP. 725-730.
75. Hartwell L. H., Hopfild J. J., Leibler S., Murray A.W. From molecular to modular cell biology // Nature. 1999. V. 402. PP. 47-52.
76. Hernandez A. I., Le Rolle V., Defontaine A., Carrault G. A multiformalism and multiresolution modelling environment: application to the cardiovascular system and its regulation // Philosophical Transactions of the Royal Society. 2009. V. 367. № 1908. PP. 4923–4940.
77. Hester R. L., Brown A. J., Husband L., Iliescu R., Pruett D., Summers R., et al. HumMod: a modeling environment for the simulation of integrative human physiology // Frontiers in Computational Physiology And Medicine. 2011. V. 2. № 12.
78. Hindmarsh A. C., Brown P. N., Grant K. E., Lee S. L., Serban R., Shumaker, D. E., et al. SUNDIALS: Suite of Nonlinear and Differential/Algebraic Equation

Solvers // ACM Transactions on Mathematical Software. 2005. V. 31. № 3. PP. 363-396.

79. Hoops S., Sahle S., Gauges R., Lee C., Pahle J., Singhal M., et al. COPASI – a complexpathway simulator // Bioinformatics. V. 22. PP. 3067-3074.
80. Hucka M., Finney A., Sauro H. M, Bolouri H., Doyle J. C., Kitano H., et al. The Systems Biology Markup Language (SBML): A medium for representation and exchange of biochemical network models // Bioinformatics, 2003. V. 19. № 4. PP. 524–531.
81. Hucka M., Bergmann F. T., Hoops S., Keating S. M., Sahle S., Schaff J. C., et al. The Systems Biology Markup Language (SBML): Language Specification for Level 3 Version 1 Core. 2010. URL: http://sbml.org/specifications/sbml-level-3/version-1/core/sbml-level-3-version-1-core.pdf (дата обращения 21.09.2016).
82. Hunter P. J., Robbins P., Noble D. The IUPS human physiology project // Pflugers Archiv European Journal of Physiology. 2004. V. 445. № 1. PP. 551–569.
83. Ikeda N., Marumo F., Shirataka M., Sato T. A model of overall regulation of body fluids // Annals of Biomedical Engineering. 1979. V. 7. PP. 135-166.
84. Inchiosa M. E., Parker M. T. Overcoming Design and Development Challenges in Agent-based Modeling Using Ascape // PNAS. 2002. 99, Suppl. 3: 7304-7308.
85. Karaaslan F., Denizhan Y., Kayserilioglu A., Ozcan Gulcur H. Long-term mathematical model involving renal sympathetic nerve activity, arterial pressure, and sodium excretion // Annals of Biomedical Engineering. 2005. V. 33. № 11. PP. 1607-1630.
86. Karr J. R., Sanghvi J. C, Macklin D. N., Gutschow M. V., Jacobs J. M., Bolival B Jr, Assad-Garcia N., Glass J. I., Covert M. W. A whole-cell computational model predicts phenotype from genotype // Cell. 2012. V. 150. № 2. PP. 389-401.
87. Keller R., Dörr A., Tabira A., Funahashi A., Ziller M. J., Adams R. et al. The systems biology simulation core algorithm // BMC Systems Biology. 2013. V. 7 PP. 55.
88. Kiselev I. N., Biberdorf E. A., Baranov V. I., Komlyagina T. G., Suvorova I. Y., Melnikov V. N., et al. Patient-specific 1d model of the human cardiovascular

system // International Conference on Biomedical Engineering and Computational Technologies (SIBIRCON). 2015. PP. 111-116.

89. Kiselev I. N., Biberdorf E. A., Baranov V. I., Komlyagina T. G., Suvorova I. Y., Melnikov V. N., Krivoshchekov S. G., Kolpakov F. A. Validation of the human arterial tree model // The 2nd International Conference Mathematical modeling and high performance computing in bioinformatics, biomedicine and biotechnology. 2016. P. 58.
90. Kiselev I., Kolpakov F. Agent Based Modeling – Plug-in for BioUML Platform // The 11th International Conference on Systems Biology. 2010. P. 195
91. Kiselev I. N., Kolpakov F. A. Modular Modeling of Biological Systems // Virtual Biology. 2013. V. 1. № 1.
92. Kiselev I. N., Kolpakov F. A., BioUML: Modular modeling of complex biological systems // The 8th International Conference on Bioinformatics of Genome Regulation and Structure. 2012. P. 146.
93. Kiselev I., Kutumova E., Sharipov R., Kolpakov F. BioUML Plugin for Modular Modeling // The 12th International Conference on Systems Biology. 2011. P. 229.
94. Kiselev I., Semisalov B., Biberdorf E., Leonova T., Sharipov R., Blokhin A., Kolpakov F. Agent-based modeling of the human cardiovascular system // The 12th International Conference on Systems Biology. 2011. P. 156.
95. Kiselev I. N., Semisalov B. V., Sharipov R. N., Kolpakov F. A. Modular approach to modeling complex biological systems. Circulatory System and Arterial Hypertension: Experimental Investigation, Mathematical and Computer Simulation / Eds. Ivanova L. N., Markel A.L., Blokhin A. M., Mishchenko E. V. Nova Science Publishers, Inc. 2012 a. PP. 143-154.
96. Kiselev I. N., Semisalov, B. V., Sharipov, R. N., Kolpakov, F. A. Modular modeling of the human cardiovascular system. Circulatory System and Arterial Hypertension: Experimental Investigation, Mathematical and Computer Simulation / Eds. L. N. Ivanova, A. L. Markel, A. M. Blokhin, E. V. Mishchenko. Nova Science Publishers, Inc. 2012 b. PP. 155-206.

97. Kofranek J. Rusz J. Restoration of Guyton's diagram for regulation of the circulation as a basis for quantitative physiological model development // Physiological Research. 2010. V. 59 № 6. PP. 897-908.
98. Kolpakov F., Tolstykh N., Lapukhov S., Kiselev I., Shadrin A. New possibilities of BioUML workbench // The 9th International Conference on Systems Biology. 2008. P. 194.
99. Kolpakov F., Tolstykh N., Kutumova E., Kiselev I., Shadrin A., Valeev T., et al. BioUML – Integrated Platform for Building Virtual Cell and Virtual Physiological Human // The 11th International Conference on Systems Biology. 2010. P. 196.
100. Kutumova E. O., Kiselev I. N., Sharipov R. N., Lavrik I. N., Kolpakov F. A. A modular model of the apoptosis machinery // Advances in Experimental Medicine and Biology. 2012. V. 736. PP. 235-245.
101. Lamponi D. One dimensional and multiscale model for blood flow circulation. Pour l'obtention du grade de docteur es sciences. Ecole Polytechnique Federale De Lausanne. 2004.
102. Le Rolle V., Hernandez A. I., Richard P. Y., Buisson J., Carrault G, A bond graph model of the cardiovascular system // Acta Biotheoreica. 2005. V. 53. № 4. PP. 295-312.
103. Le Rolle V., Ojeda D., Madelein R., Carrault G., Hernandez A.I. Coupling the Guyton model to pulsatile ventricles using a multiresolution modelling environment // Computers in Cardiology. 2010. V. 37. PP. 325–328.
104. Le Rolle V., Ojeda D., Hernandez A.I. Embedding a cardiac pulsatile model into an integrated model of the cardiovascular regulation for heart failure follow-up // IEEE Transactions on Biomedical Engineering. 2011. V. 58. № 10. PP. 2982-2986.
105. Le Novère N., Hucka M., Mi H., Moodie S., Schreiber F., Sorokin A., et al. The Systems Biology Graphical Notation // Nature Biotechnology. 2009. V. 27. PP. 735 – 741.
106. Lopez C. F., Muhlich J. L., Bachman J.A., Sorger P. K. Programming biological models in Python using PySB // Molecular Systems Biology. 2013. V. 9. № 646.

107. Luke S., Cioffi-Revilla C., Panait L., Sullivan K., Balan G. MASON: A Multi-Agent Simulation Environment // Simulation: Transactions of the society for Modeling and Simulation International. 2005. V. 82. № 7 PP. 517-527.
108. Mallavarapu A., Thomson M., Ullian B. Programming with models: modularity and abstraction provide powerful capabilities for systems biology // Journal of the Royal Society Interface. 2009. V. 6. № 32. PP. 257-270.
109. Milligan P. A, Brown M. J., Marchant B., Martin S. W., van der Graaf P. H., Benson N., et al. Model-Based Drug Development: A Rational Approach to Efficiently Accelerate Drug Development // Clinical Pharmacology and Therapeutics. 2013. V. 93. № 6. PP. 502-514.
110. Mirschel S., Steinmetz K., Rempel M., Ginkel M., Gilles E. D. PROMOT: modular modeling for systems biology // Bioinformatics. 2009. V. 25 № 5. PP. 687-689.
111. Moraru I. I., Schaff J. C., Slepchenko B. M., Blinov M. L., Morgan, F., Lakshminarayana A., et al. Virtual Cell modelling and simulation software environment // IET Systems Biology 2008. V. 2. № 5 PP. 352-62.
112. Myers C. J., Barker N., Jones K., Kuwahara H., Madsen C., Nguyen N. P. iBioSim: a tool for the analysis and design of genetic circuits // Bioinformatics. 2009. V. 25. № 921. PP. 2848-2849.
113. Nguyen C. N., Simanski O., Kahler R., Schubert A., Janda M., Bajorat J., et al. The benefits of using Guytons' model in a hypotensive control system // Computer Methods and Programs in Biomedicine. 2008. V. 89. PP. 153-161.
114. Noordergraaf A. Physical basis of ballistocardiography. *Ph.D. Thesis. Univ. of Utrecht.* 1956.
115. North M. J., Collier N. T., Ozik J., Tatara E., Altaweel M., Macal C.M., et al., Complex Adaptive Systems Modeling with Repast Simphony // Complex Adaptive Systems Modeling. 2013. V. 1 № 3.
116. Odell J. Agent Technology an Overview. 2010. URL: http://www.jamesodell.com/Agent_Technology-An_Overview.pdf (дата обращения 21.09.2016).

117. Osborn J. W., Averina V. A., Fink G. D. Current computational models do not reveal the importance of the nervous system in long-term control of arterial pressure // Experimental Physiology. 2009. V. 94 № 4. PP. 381-397.
118. Patterson M. D. Implementing Runge-Kutta solvers in Java, tech. rep., Acadia University, 2002. Honors Undergraduate Thesis, Jodrey School of Computer Science.
119. Perez L., Dragicevic S. An agent-based approach for modeling dynamics of contagious disease spread // International Journal of Health Geographics. 2009. V. 8. № 50.
120. Quarteroni A., Formaggia L. Mathematical modelling and numerical simulation of the cardiovascular system. In: Ciarlet P. G. (ed.), Handbook of numerical analyis, vol. 12, special volume: Ayache N.(ed.), Computational Models for the Human Body, 3–129, Elsevier Science & Technology, Amsterdam, 2003.
121. Radhakrishnan K., Hindmarsh A. C. Description and Use of LSODE, the Livermore Solver for Ordinary Differential Equations, LLNL report UCRL-ID-113855, December 1993.
122. Randhawa R. Model Composition and Aggregation in Macromolecular Regulatory Network. Virginia Polytechnic Institute. 2008.
123. Randhawa R., Shaffer C. A., Tyson J. J. Model aggregation: a building-block approach to creating large macromolecular regulatory networks // Bioinformatics. 2009. V. 25. № 24. PP. 3289-3295.
124. Randhawa R., Shaffer C. A., Tyson J. J. Model Composition for Macromolecular Regulatory Networks // Computational Biology and Bioinformatics, IEEE/ACM Transactions on. 2010. V. 7. № 2. PP. 278-287.
125. Raymond G. M., Butterworth E., Bassingthwaighte J. B. JSIM: Free software package for teaching physiological modeling and research // Experimental Biology. 2003. V. 280. № 5. P. 102.
126. Remuzzi A., Brenner B. M., Pata V. Three-dimensional reconstructed glomerular capillary network: blood flow distribution and local filtration // The American Journal of Physiology. 1992. V. 263. № 3. Pt. 2. PP. F562—F572.

127. Rohn H., Junker A., Hartmann A., Grafahrend-Belau E., Trutler H., Klapperstück M., et al. VANTED v2: a framework for systems biology applications // BMC Systems Biology. 2012. V. 6. PP. 139-139.
128. Sacco J. J., Botten J., Macbeth F., Bagust A., Clark P. The Average Body Surface Area of Adult Cancer Patients in the UK: A Multicentre Retrospective Study // PLoS One. 2010. V. 5 № 1.
129. Semisalov B. V., Kiselev I. N., Sharipov R. N., Kolpakov F. A. Computational and analytical aspects of a new complex model describing human cardiovascular system. The 8th International Conference on Bioinformatics of Genome Regulation and Structure, June 25-29, 2012, Novosibirsk, Russia. P. 282.
130. Simone G., Devereux R. B., Daniels S. R., Mureddu G., Roman M. J., Kimball T. R., et al. Stroke volume and cardiac output in normotensive children and adults: assessment of relations with body size and impact of overweight // Circulation. 1997. V. 95. PP. 1837–1843.
131. Smith L. P., Bergmann F. T., Chandran D. Sauro M. H. Antimony: a modular model definition language // Bioinformatics. 2009. V. 25 № 18. PP. 2452-2454.
132. Smith L. P., Hucka M., Hoops S., Finney A., Ginkel M., Myers C. J., et al. Hierarchical Model Composition, Version 1 Release 2. 2013, URL: http://identifiers.org/combine.specifications/sbml.level-3.version-1.comp.version-1.release-3 (дата обращения 21.09.2016).
133. Snoep J. L., Bruggeman F., Olivier B. G. Westerhoff H. V. Towards building the silicon cell: a modular approach // Biosystems. 2006. V. 83 PP. 207-216.
134. Sodium (Na) in Blood, URL: http://www.webmd.com/a-to-z-guides/sodium-na-in-blood (дата обращения 21.09.2016).
135. Stergiopulos N., Tardy Y., Meister J.-J. Nonlinear separation of forward and backward running waves in elastic conduits // Journal of Biomechanics. 1993. V. 26. PP. 201-209.
136. Stevens J. T., Myers C. J. Dynamic Modeling of Cellular Populations within iBioSim // ACS Synthetic Biology. 2013. V. 2. № 5. PP. 223-229.

137. Sütterlin T., Kolb C., Dickhaus H., Jager D., Grabe N. Bridging the scales: semantic integration of quantitative SBML in graphicsl multi-cellular models and simulations with EPISIM and COPASI // Bioinformatics. 2013. V. 29. № 2 PP. 223-229.
138. Swarm wiki. URL: www.swarm.org (дата обращения 21.09.2016).
139. Taishin N. Towards integration of biological and physiological fucntions at multiple levels // Frontiers in Physiology. 2010. V.1. № 1164.
140. Thomas S. R., Abdulhay E., Baconnier P., Fontecave J., Francoise J. P., Guillaud F., et al. SAPHIR – a multiscale, multi-resolution modeling environment targeting blood pressure regulation and fluid homeostasis. Proceedings of the IEEE EMBS. 2007.
141. Thomas S. R., Baconnier P., Fontecave J., Francoise J. P., Guillaud F., Hannaert P., et al. SAPHIR: a physiome core model of body fluid homeostasis and blood pressure regulation // Philosophical Transactions of the Royal Society. 2008. V. 366. PP. 3175-3197.
142. Uttamsingh R. J., Leaning M. S., Bushman J. A., Carson E. R., Finkelstein L., Mathematical model of the human renal system // Medical & Biological Engineering & Computing. 1985. V. 23. PP. 525-535.
143. Vangheluwe H. Multi-formalism modeling and simulation. D. Sc. Dissertation, Faculty of Sciences. Ghent University (UGent), December, 2000.
144. Vass M., Allen N., Shaffer C. A., Ramakrishnan N., Watson L. T., Tyson J. J. The JigCell Model Builder and Run Manager // Bioinformatics, 2004. V. 20 № 18. PP. 3680-3681.
145. Wang J., Parker J. H. Wave propagation in a model of the arterial circulation // Journal of Biomechanics. 2004. V. 37. PP. 457-470.
146. Werner J., Bohringer D., Hexamer M. Simulation and prediction of cardiotherapeutical phenomena from a pulsatile model coupled to the Guyton circulatory model // IEEE TBME. 2002. V. 49. PP. 430-439.
147. Westerhof N., Bosman F., Vries C., Noordergraaf A. Analog studies of the human systemic arterial tree // Journal of Biomechanics. 1969. V. 2. PP. 121-143.

148. Wilensky U., Rand W. An introduction to agent-based modeling: Modeling natural, social and engineered complex systems with NetLogo. Cambridge, MA: MIT Press. 2015.
149. Wimalaratne S. M., Halstead M. D. B., Lloyd C. M., Cooling M. T., Crampin E. J., Nielsen, P.F. A method for visualizing CellML models // Bioinformatics. 2009. V. 25. PP. 3012-3019.

ПРИЛОЖЕНИЕ А

В таблице А.1 приведены данные по 55 крупнейших сосудов артериальной системы человека, использованные в модели артериального дерева. Данные приведены по [Stergiopulos et al., 1992; Westerhof et al., 1969; Noordergraaf A., 1956] с правками из [Wang, Parker, 2004]. Жирным выделены терминальные сосуды, т. е. сосуды не разветвляющиеся на два дочерних. Под родительски сосудом понимается сосуд, который разветвляется на данный сосуд и какой-либо другой.

Таблица А.1 – Данные по 55 сосудам артериальной системы человека

Номер сосуда	Название	Длина l (см)	Площадь сечения (см^2)	Коэффициент жесткости β (кг/см^2)	Номер родительского сосуда
1	Ascending Aorta	4	5.983	388	-
2	Aortic Arch I	2	5.147	348	1
3	Brachiocephalic	3.4	1.219	932	1
4	R. Subclavian I	3.4	0.562	1692	3
5	R. Carotid	17.7	0.432	2064	3
6	**R. Vertebral**	**14.8**	**0.123**	**10360**	**4**
7	R. Subclavian II	42.2	0.510	1864	4
8	**R. radial**	**23.5**	**0.106**	**11464**	**7**
9	R. Ulnar I	6.7	0.145	8984	7
10	**R. Interosseous**	**7.9**	**0.031**	**51576**	**9**
11	**R. Ulnar II**	**17.1**	**0.133**	**9784**	**9**
12	**R. internal Carotid**	**17.6**	**0.121**	**10576**	**5**
13	**R. external Carotid**	**17.7**	**0.121**	**9868**	**5**
14	Aortic Arch II	3.9	3.142	520	2
15	L. Carotid	20.8	0.430	2076	2
16	**L. internal Carotid**	**17.6**	**0.121**	**10576**	**15**
17	**L. external Carotid**	**17.7**	**0.121**	**9868**	**15**
18	Thoracic Aorta I	5.2	3.142	496	14
19	L. Subclavian I	3.4	0.562	1664	14
20	**Vertebral**	**14.8**	**0.123**	**10360**	**19**
21	L. Subclavian II	42.2	0.510	1864	19
22	**L. Radial**	**23.5**	**0.106**	**11464**	**21**
23	L. Ulnar I	6.7	0.145	8984	21
24	**L. Interosseous**	**7.9**	**0.031**	**51576**	**23**
25	**L. Ulnar II**	**17.1**	**0.133**	**9784**	**23**
26	**Intercostals**	**8.0**	**0.196**	**3540**	**18**
27	Thoracic Aorta II	10.4	3.017	468	18
28	Abdominal I	5.3	1.911	668	27
29	Celiac I	2.0	0.478	1900	27

Окончание таблицы А.1

Номер сосуда	Название	Длина l (см)	Площадь сечения (см^2)	Коэффициент жесткости β (кг/см^2)	Номер родительского сосуда
30	Celiac II	1.0	0.126	7220	29
31	**Hepatic**	**6.6**	**0.152**	**4568**	**29**
32	**Gastric**	**7.1**	**0.102**	**6268**	**30**
33	**Splenic**	**6.3**	**0.238**	**3224**	**30**
34	**Superior Mesenteric**	**5.9**	**0.430**	**2276**	**28**
35	Abdominal II	1.0	1.247	908	28
36	**L. Renal**	**3.2**	**0.332**	**2264**	**35**
37	Abdominal III	1.0	1.021	1112	35
38	**R. Renal**	**3.2**	**0.159**	**4724**	**37**
39	Abdominal IV	10.6	0.697	1524	37
40	**Inferior Mesenteric**	**5.0**	**0.080**	**7580**	**39**
41	Abdominal V	1.0	0.578	1596	39
42	R. common Iliac	5.9	0.328	2596	41
43	L. common Iliac	5.8	0.328	2596	41
44	L. external Iliac	14.4	0.252	5972	43
45	**L. internal Iliac**	**5.0**	**0.181**	**12536**	**43**
46	L. Femoral	44.3	0.139	10236	45
47	**L. deep Femoral**	**12.6**	**0.126**	**10608**	**45**
48	**L. posterior Tibial**	**32.1**	**0.110**	**23232**	**46**
49	**L. anterior Tibial**	**34.3**	**0.060**	**36972**	**46**
50	R. external Iliac	14.5	0.252	5972	42
51	**R. internal Iliac**	**5.1**	**0.181**	**12536**	**42**
52	R. Femoral	44.4	0.139	10236	50
53	**R. deep Femoral**	**12.7**	**0.126**	**10608**	**50**
54	**R. posterior Tibial**	**32.3**	**0.110**	**23232**	**52**
55	**R. anterior Tibial**	**34.4**	**0.060**	**36972**	**52**

ПРИЛОЖЕНИЕ Б

Модули модели сердечных сокращений

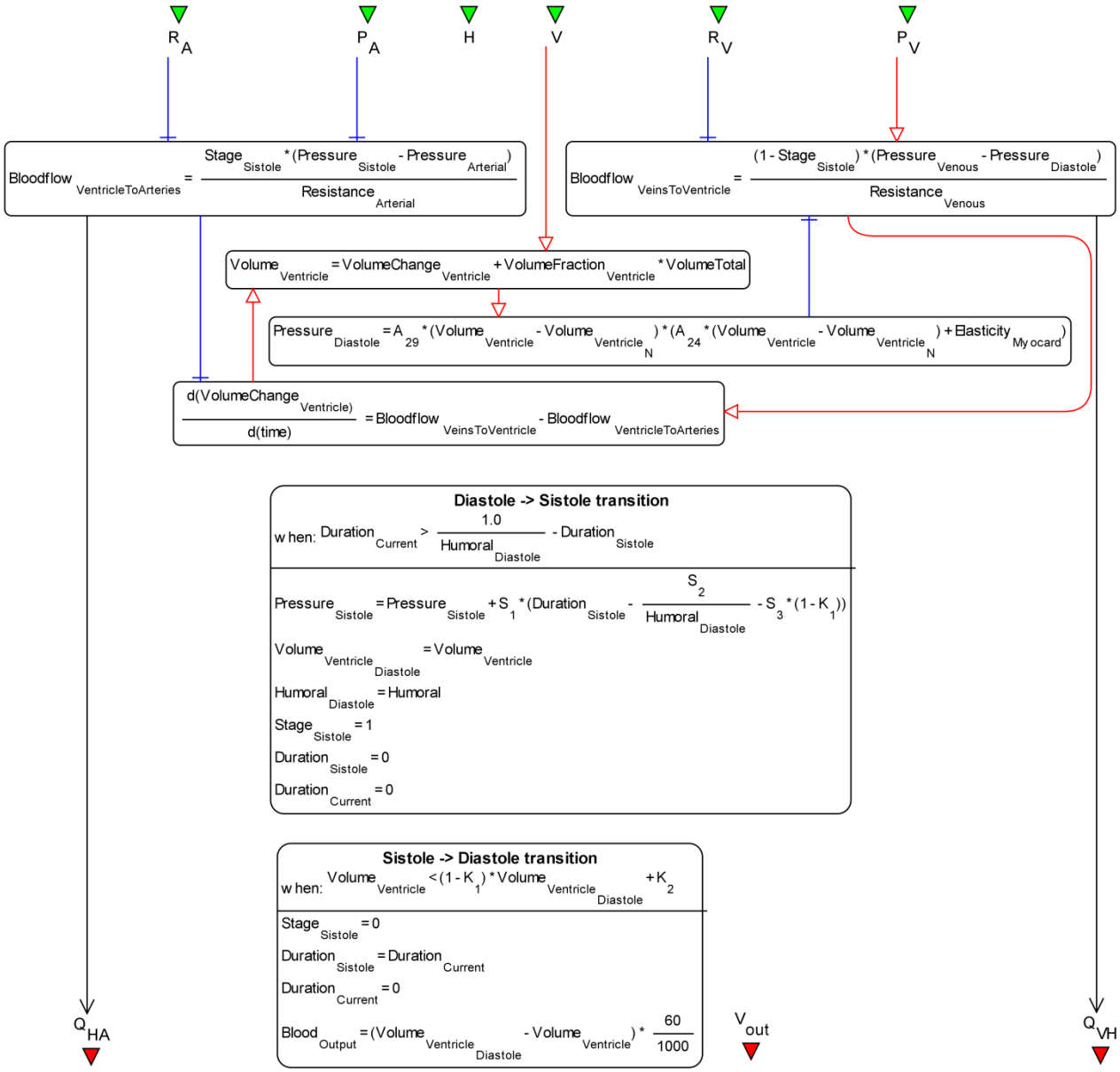


Рисунок Б.1 – Модуль “Левый желудочек”

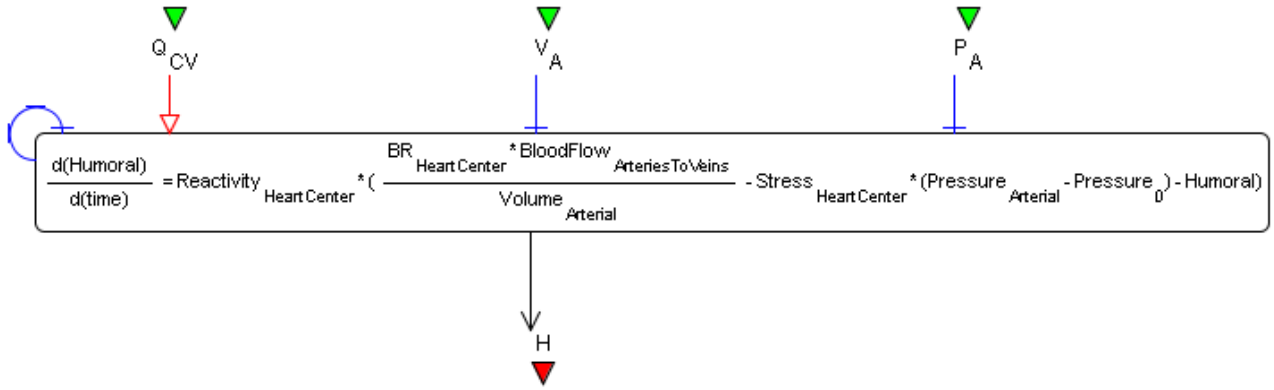


Рисунок Б.2 – Модуль “Нейрогуморальный контроль”

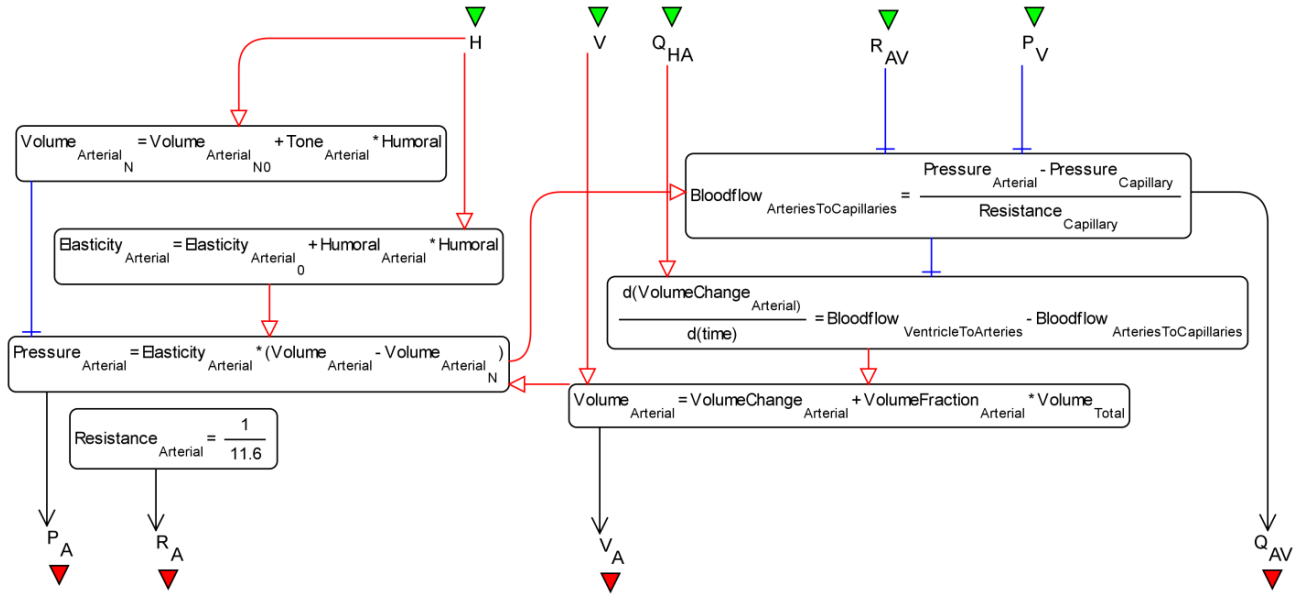


Рисунок Б.3 – Модуль “Артериальная система”

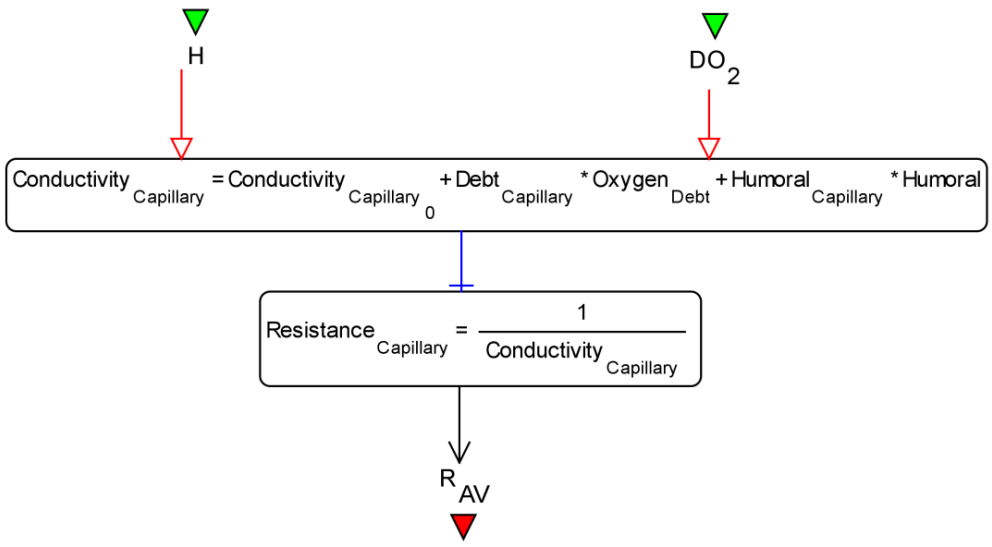


Рисунок Б.4 – Модуль “Капилляры”

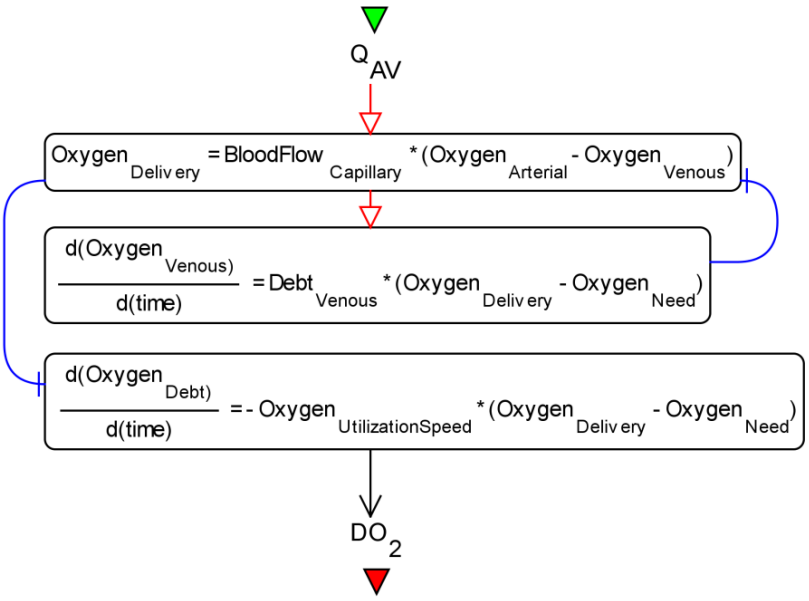


Рисунок Б.5 – Модуль “Тканевый метаболизм”

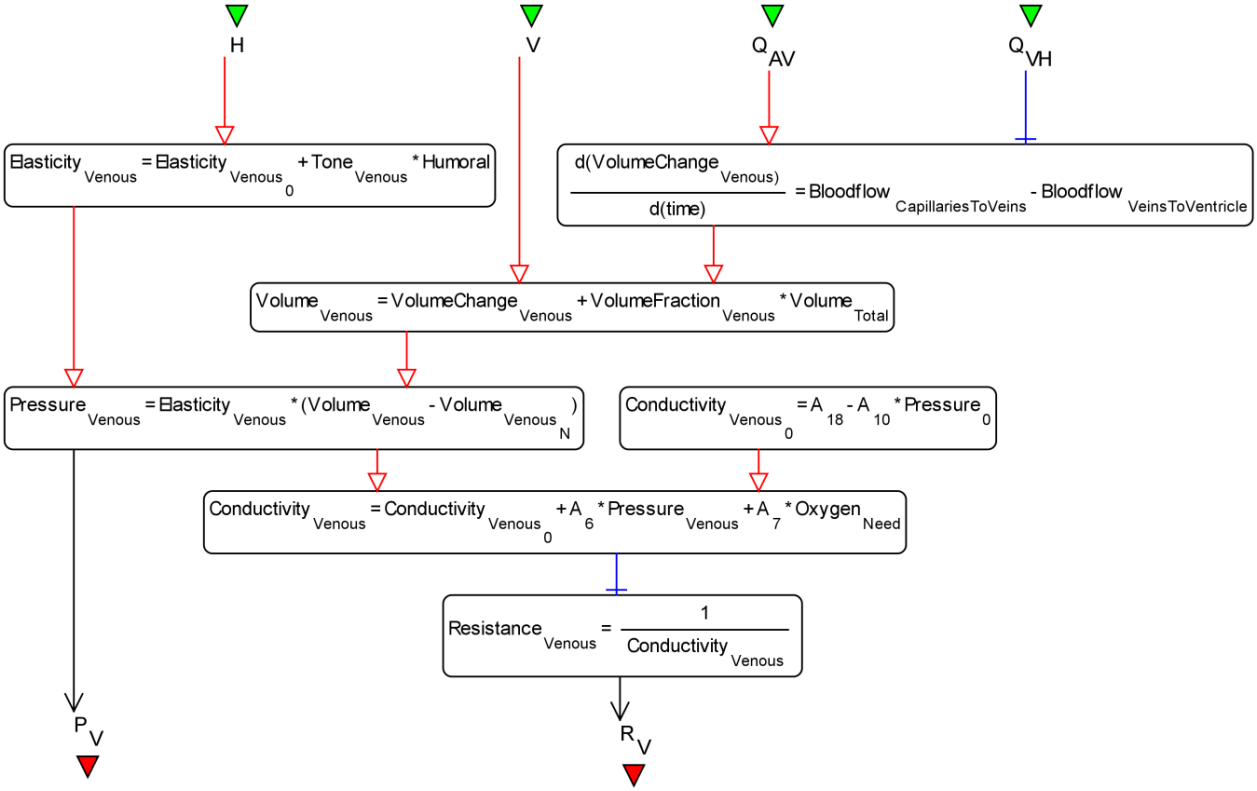


Рисунок Б.6 – Модуль “Венозная система”

Модули модели почечной регуляции

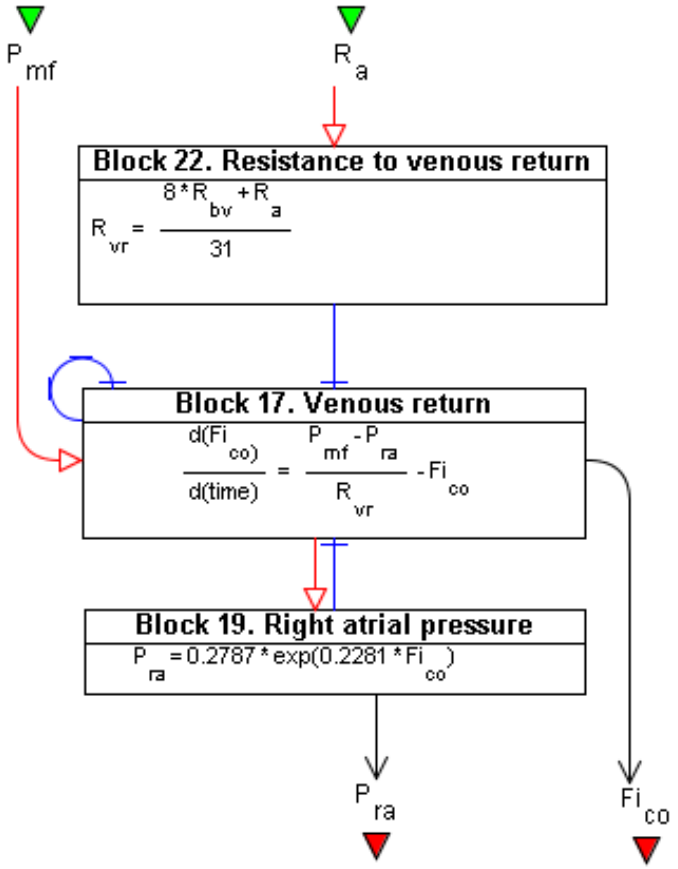


Рисунок Б.7 – Модуль “Сердце”

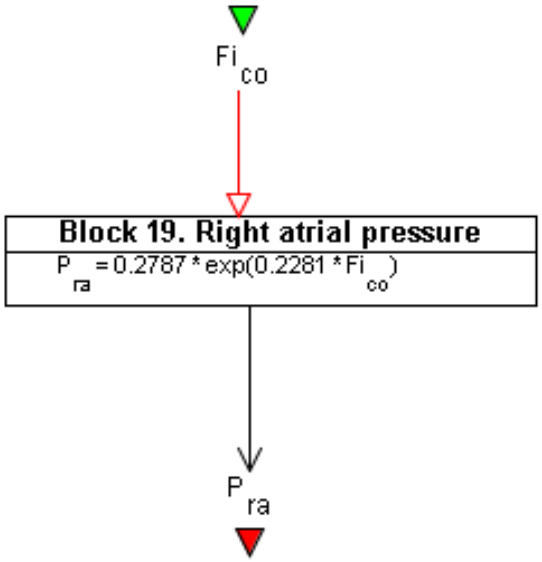


Рисунок Б.8 – Модуль “Правый желудочек”

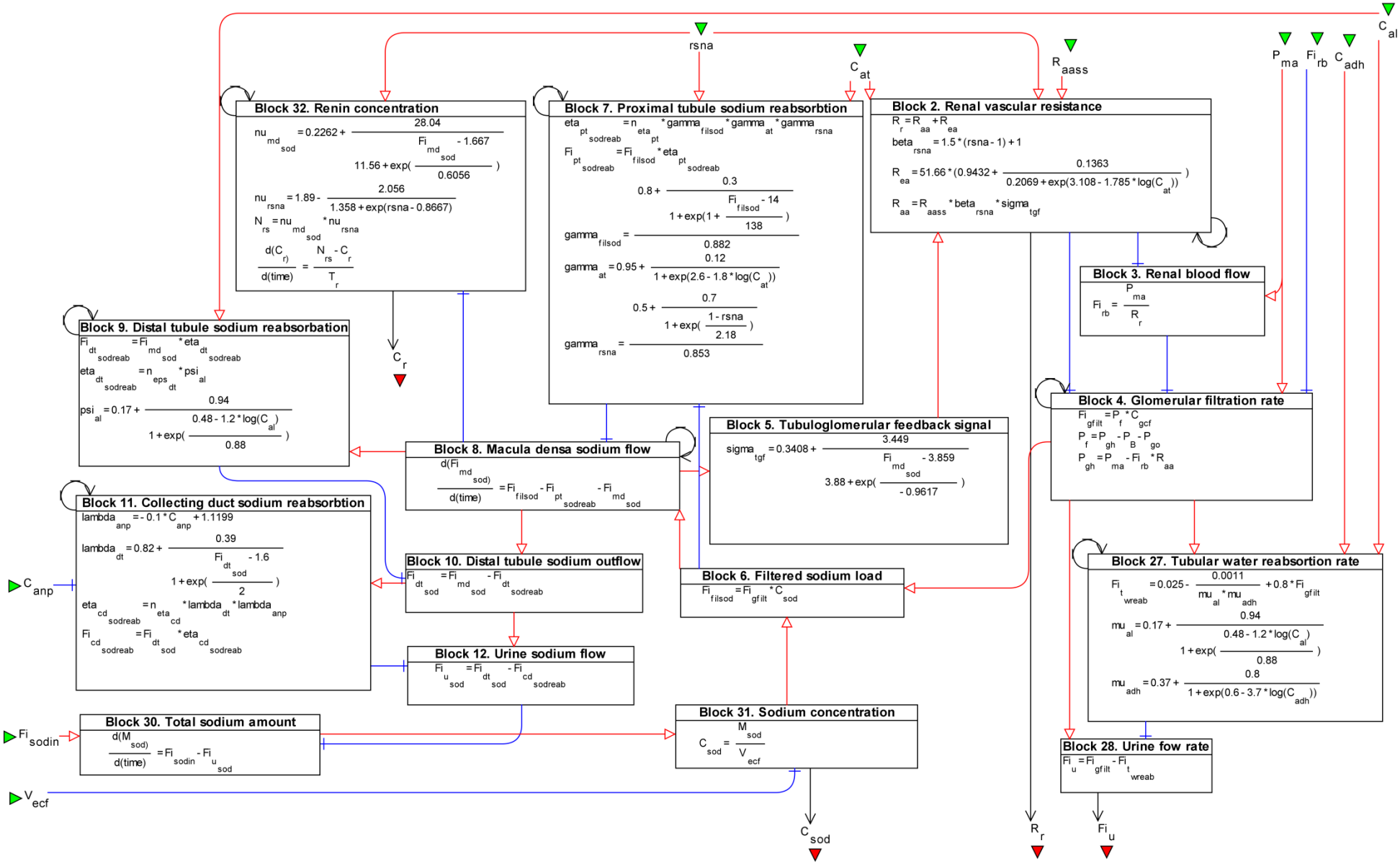


Рисунок Б.9 – Модуль “Почка”

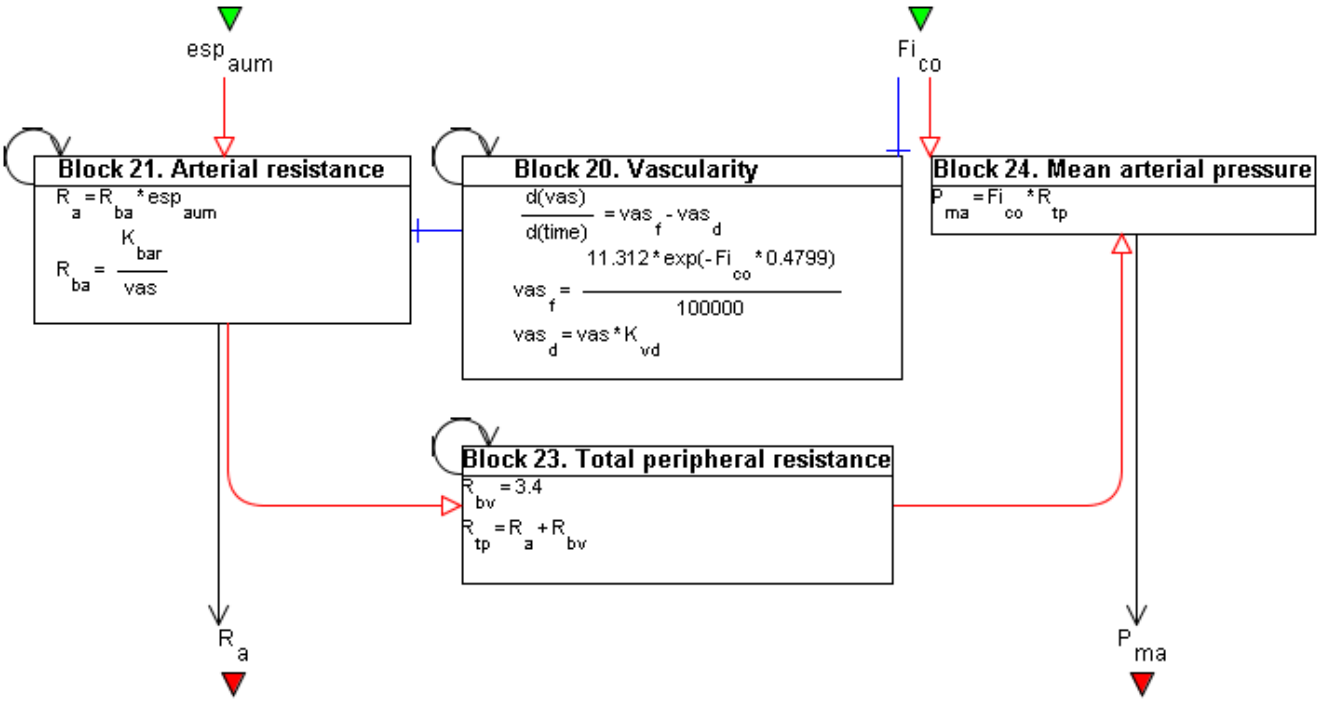


Рисунок Б.10 – Модуль “Артерии”

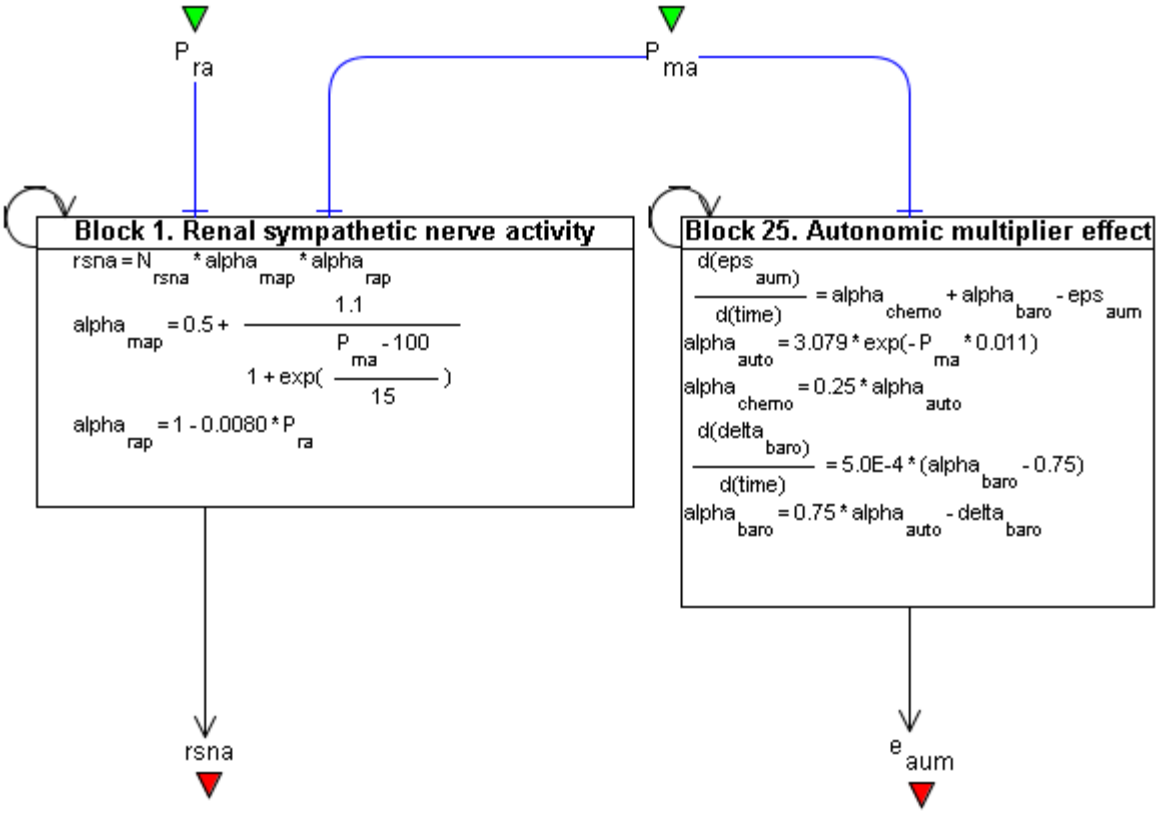


Рисунок Б.11 – Модуль “Нервная система”

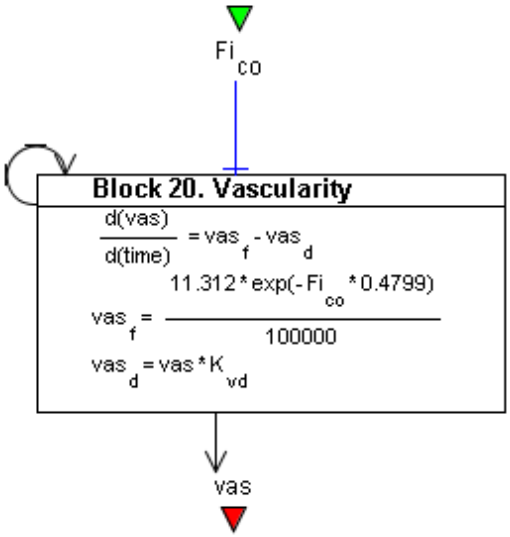


Рисунок Б.12 – Модуль “Васкулярность”

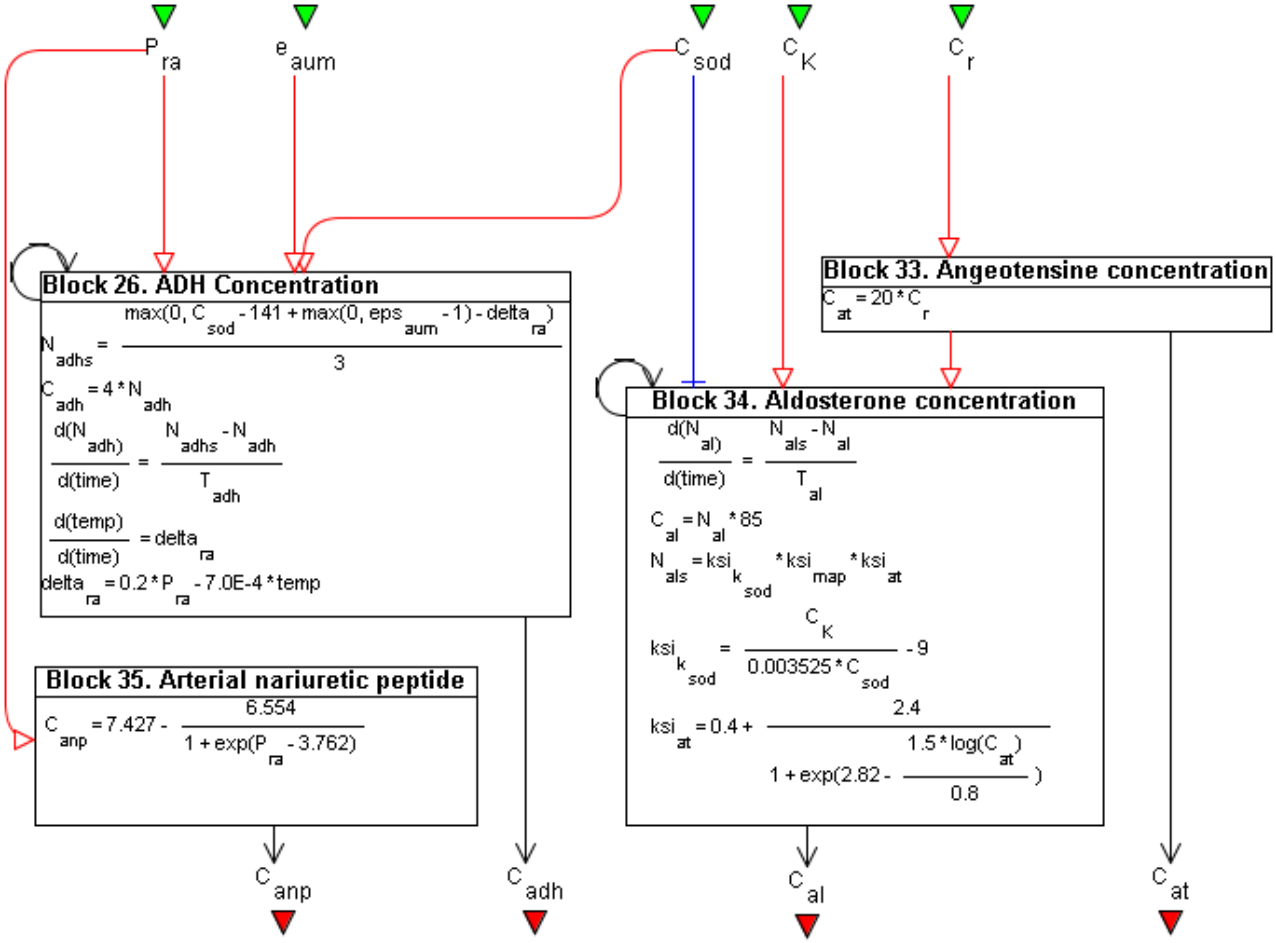


Рисунок Б.13 – Модуль “Гормональная система”

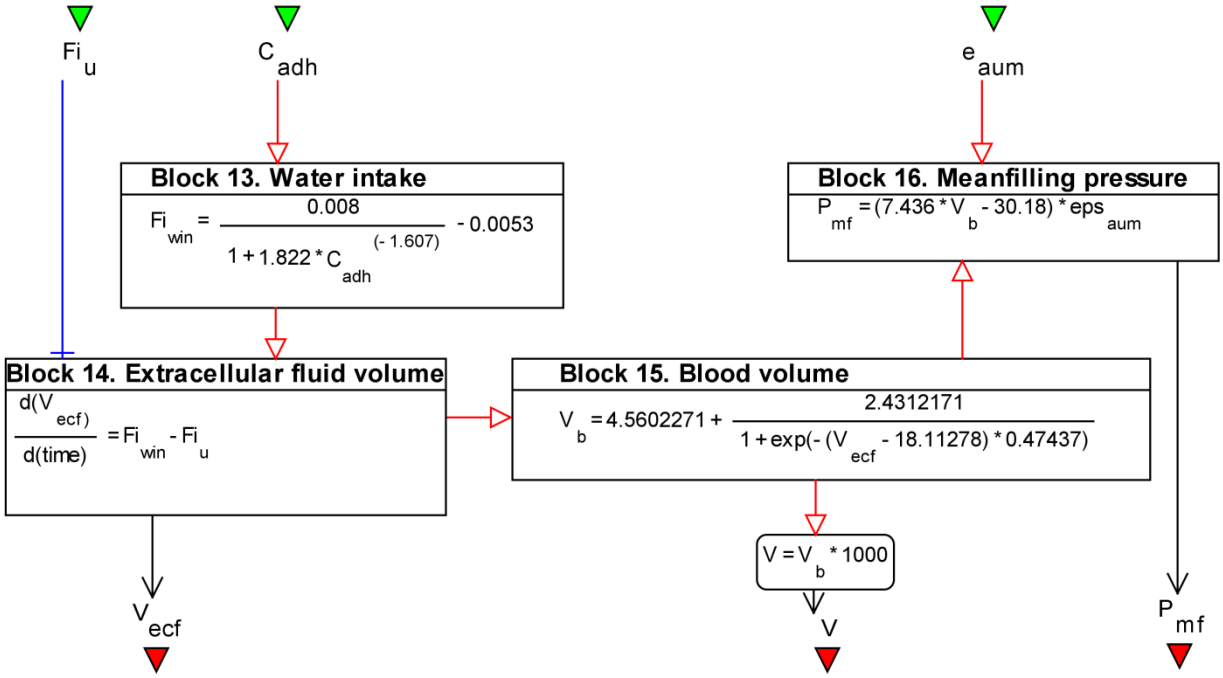


Рисунок Б.14 – Модуль “Контроль объема жидкости”

ПРИЛОЖЕНИЕ В

В таблице В.1. Приведен список всех переменных трех использованных моделей. В первом столбце приведено название переменной, использованное при создании модели в BioUML (они соответствуют названиям на рис. 1-14 приложения 2). В последнем столбце – название, использованное для обозначения переменной в модульных модулях в том случае, если эти переменные используются для установления связей между модулями.

Таблица В.1 – Переменные моделей, использованных в работе

Полное обозначение	Описание	Единицы измерения	Краткое обозначение (если есть)
Модель сердечных сокращений			
$BloodFlow_{ArtariesToVeins}$	Мгновенный кровоток из артерий в капилляры.	мл/с	Q_{AV}
$BloodFlow_{VeinsToVentricle}$	Мгновенный кровоток из вен в желудочек.	мл/с	Q_{VH}
$BloodFlow_{VentricleToArteries}$	Мгновенный кровоток из желудочка в артерии.	мл/с	Q_{HA}
$Blood_{Output}$	Ударный объем.	мл	V_{Out}
$BR_{HeartCenter}$	Барорецепторная чувствительность сердечного центра.	-	-
$Conductivity_{Arteries}$	Коэффициент проводимости артерий.	(мл/с)/мм рт. ст.	-
$Conductivity_{Capillaries}$	Коэффициент проводимости капилляров.	(мл/с)/мм рт. ст.	-
$Conductivity_{Venous}$	Коэффициент проводимости вен.	(мл/с)/мм рт. ст.	-
$Conductivity_{Venous\_0}$	Коэффициент проводимости вен при нулевом давлении.	(мл/с)/мм рт. ст.	-
$Debt_{Capillary}$	Чувствительность капилляров к кислородному долгу.	-	-
$Debt_{Venous}$	Чувствительность вен к кислородному долгу.	-	-
$Duration_{Systole}$	Длительность систолы.	с	-

Продолжение таблицы В.1

Полное обозначение	Описание	Единицы измерения	Краткое обозначение (если есть)
$Duration_{Current}$	Длительность текущей стадии (систолы или диастолы).	с	-
$Elasticity_{Arterial}$	Упругость артерий.	мм рт. ст./мл	-
$Elasticity_{Arterial}$	Коэффициент упругости артерий при нулевом давлении.	мм рт. ст./мл	-
$Elasticity_{Myocard}$	Упругость миокарда.	-	-
$Elasticty_{Venous}$	Коэффициент упругости артерий.	мм рт. ст./мл	-
$Elasticity_{Venous\_0}$	Коэффициент упругости артерий при нулевом давлении.	мм рт. ст./мл	-
$Humoral_{Arterial}$	Чувствительность артерий к нейрогуморальному фактору.	-	-
$Humoral$	Нейрогуморальный фактор.	1/с	H
$Humoral_{Diastole}$	Нейрогуморальный фактор при переходе от диастолы к систоле.	-	H_D
K_1	Инотропный коэффициент сердца.	-	-
K_2	Инотропный коэффициент сердца.	-	-
$Oxygen_{Arterial}$	Содержание кислорода в артериальной крови.	%	-
$Oxygen_{Debt}$	Кислородный долг.	мл	DO_2
$Oxygen_{Delivery}$	Мгновенная доставка кислорода тканям.	мл/с	-
$Oxygen_{Need}$	Мгновенная потребность в кислороде.	мл/с	NO_2
$Oxygen_{UtilizationSpeed}$	Скорость утилизации кислорода.	мл/с	-
$Oxygen_{Venous}$	Содержание кислорода в венозной крови.	%	-
$Pressure_{Arterial}$	Артериальное давление.	мм рт. ст.	P_A
$Pressure_{Diastole}$	Диастолическое давление в левом желудочке.	мм рт. ст.	P_D
$Pressure_{Systole}$	Систолическое давление в левом желудочке.	мм рт. ст.	P_S
$Presure_{Venous}$	Давление в венах.	мм рт. ст.	P_V
$Reactivity_{HeartCenter}$	Реактивность сердечного центра.	мм рт. ст./(мл/с)	-
$Resistance_{Arterial}$	Сопротивляемость артерий кровотоку из желудочка.	мм рт. ст./(мл/с)	R_{HA}
$Resistance_{Venous}$	Сопротивляемость вен кровотоку из капилляр.	мм рт. ст./(мл/с)	R_{VH}
$Resistance_{Capillary}$	Сопротивляемость капилляров кровотоку.	мм рт. ст./(мл/с)	R_{AV}

Продолжение таблицы В.1

Полное обозначение	Описание	Единицы измерения	Краткое обозначение (если есть)
S_1	-	-	-
S_2	-	-	-
S_3	-	-	-
$Stage_{Systole}$	Индикатор состояния систолы.	-	-
$Stress_{HeartCenter}$	Чувствительность сердечного центра к нагрузке.	-	-
$Tone_{Arterial}$	Артериальный тонус.	мл*с	-
$Tone_{Venous}$	Венозный тонус.	мл*с	-
$Volume_{Arterial}$	Объем артериального резервуара.	мл	V_A
$Volume_{Arterial\_N}$	Ненапряженный объем артериального резервуара.	мл	-
$Volume_{Venous}$	Объем венозного резервуара.	мл	V_V
$Volume_{Ventricular\_N}$	Ненапряженный объем венозного резервуара.	мл	-
Модель почечной регуляции			
a_{auto}	Активность автономной системы.	-	-
a_{baro}	Активность барорецепторов.	-	-
a_{chemo}	Активность хеморецепторов.	-	-
$alpha_{map}$	Эффект среднего артериального давления на почечную симпатическую нервную активность.	-	α_{map}
$alpha_{rap}$	Эффект давления в правом предсердии на почечную симпатическую нервную активность.	-	α_{rap}
$beta_{rsna}$	Эффект почечной симпатической нервной активности на сопротивление афферентной артериолы.	-	β_{rsna}
C_{adh}	Концентрация антидиуретического гормона.	мЭкв/л	-
C_{al}	Концентрация альдостерона.	нг/л	-
C_{anp}	Нормализованная концентрация натриуретического пептида.	-	-
C_{at}	Концентрация ангиотензина.	нг/л	-
C_{gcf}	Коэффициент фильтрации клубочковых капилляров.	-	-
C_K	Концентрация калия в крови.	мЭкв/л	-
C_r	Нормализованная концентрация ренина.	-	-
C_{sod}	Концентрация натрия в крови.	мЭкв/л	-

Продолжение таблицы В.1

Полное обозначение	Описание	Единицы измерения	Краткое обозначение (если есть)
$delta_{ra}$	Эффект давления в правом предсердии на нормализованную скорость секреции антидиуретического гормона.	-	δ_{ra}
eps_{aum}	Эффект автономного множителя.	-	$\varepsilon_{aum}, e_{aum}$
$eta_{cd\text{-}sodreab}$	Реабсорбция натрия собирательной трубкой (доля от проходящего количества).	-	$\eta_{cd-sodreab}$
$eta_{dt\text{-}sodreab}$	Реабсорбция натрия дистальным канальцем (доля от проходящего количества).	-	$\eta_{dt-sodreab}$
$eta_{pt\text{-}sodreab}$	Реабсорбция натрия проксимальным канальцем (доля от проходящего количества).	-	$\eta_{pt-sodreab}$
Fi_{co}	Сердечный выброс (минутный объем).	л/мин	Φ_{co}
$Fi_{dt\text{-}sod}$	Поток натрия, выходящий из дистального канальца.	мЭкв/мин	Φ_{dt-sod}
$Fi_{dt\text{-}sodreab}$	Абсолютная скорость реабсорбции натрия дистальным канальцем.	мЭкв/мин	$\Phi_{dt-sodreab}$
Fi_{filsod}	Нагрузка профильтровавшегося натрия.	мЭкв/мин	Φ_{filsod}
Fi_{gfilt}	Скорость клубочковой фильтрации.	л/мин	Φ_{gfilt}
$Fi_{cd\text{-}sodreab}$	Абсолютная скорость реабсорбции натрия собирательной трубкой.	мЭкв/мин	$\Phi_{cd-sodreab}$
$Fi_{md\text{-}sod}$	Скорость тока натрия через Macula densa.	мЭкв/мин	Φ_{md-sod}
$Fi_{pt\text{-}sodreab}$	Абсолютная скорость реабсорбции натрия проксимальным канальцем.	мЭкв/мин	$\Phi_{pt-sodreab}$
Fi_{rb}	Скорость тока крови в почке.	л/мин	Φ_{rb}
Fi_{sodin}	Потребление соли.	мЭкв/мин	Φ_{sodin}
$Fi_{t\text{-}wreab}$	Скорость реабсорбции воды в канальцах.	л/мин	$\Phi_{t-wreab}$
Fi_{u}	Скорость диуреза.	л/мин	Φ_{u}
$Fi_{u\text{-}sod}$	Поток натрия в составе мочи.	мЭкв/мин	Φ_{u-sod}

Продолжение таблицы В.1

Полное обозначение	Описание	Единицы измерения	Краткое обозначение (если есть)
Fi_{vr}	Венозный возврат.	мЭкв/мин	Φ_{vr}
Fi_{win}	Потребление воды.	мЭкв/мин	Φ_{win}
$gamma_{at}$	Эффект концентрации ангиотензина на реабсорбцию натрия проксимальным канальцем.	-	γ_{at}
$gamma_{filsod}$	Эффект нагрузки профильтровавшегося натрия на его реабсорбцию проксимальным канальцем .	-	γ_{filsod}
$gamma_{rsna}$	Эффект почечной симпатической нервной активности на реабсорбцию натрия проксимальным канальцем.	-	γ_{rsna}
K_{bar}	Коэффициент, связывающий базовое сопротивление артерий с васкуляризацией (обеспеченностью сосудами).	-	-
K_{vd}	Коэффициент снижения числа сосудов.	-	-
ksi_{at}	Эффект ангиотензина на скорость секреции альдостерона.	-	ξ_{at}
$ksi_{k/sod}$	Эффект отношения концентраций калия/натрия на скорость секреции альдостерона .	-	$\xi_{k/sod}$
ksi_{map}	Эффект среднего артериального давления на скорость секреции альдостерона.	-	ξ_{map}
$lambda_{anp}$	Эффект натриуретического пептида на скорость реабсорбции натрия собирательными трубками.	-	λ_{anp}
$lambda_{dt}$	Эффект тока натрия, прошедшего дистальный каналец, на скорость его реабсорбции в собирательных трубках.	-	λ_{dt}
M_{sod}	Общее количество натрия.	мЭкв	-

Продолжение таблицы В.1

Полное обозначение	Описание	Единицы измерения	Краткое обозначение (если есть)
mu_{adh}	Эффект концентрации антидиуретического гормона на скорость реабсорбции воды канальцами.	-	μ_{adh}
mu_{al}	Эффект концентрации альдостерона на скорость реабсорбции воды канальцами.	-	μ_{al}
N_{adh}	Нормализованная концентрация антидиуретического гормона.	-	-
N_{adhs}	Нормализованная скорость секреции антидиуретического гормона.	-	-
N_{al}	Нормализованная концентрация альдостерона.	-	-
N_{als}	Нормализованная скорость секреции альдостерона.	-	-
n_{eps_dt}	Нормальное значение реабсорбции натрия дистальным канальцем.	-	$n_{\varepsilon-pt}$
n_{eta_cd}	Нормальное значение реабсорбции натрия собирательной трубкой (доля от проходящего количества).	-	$n_{\eta-pt}$
n_{eta_pt}	Нормальное значение реабсорбции натрия проксимальным канальцем (доля от проходящего количества).	-	$n_{\eta-pt}$
N_{rs}	Нормализованная скорость секреции ренина.	-	-
N_{rsna}	Нормализованная почечная симпатическая активность.	-	-
nu_{md_sod}	Эффект тока натрия через Macula densa на скорость секреции ренина.	-	ν_{md-sod}
nu_{rsna}	Эффект почечной симпатической нервной активности на скорость секреции ренина.	-	ν_{rsna}
psi_{al}	Эффект альдостерона на реабсорбцию натрия дистальным канальцем.	-	ψ_{al}
P_B	Гидростатическое давление в капсуле Боумена-Шумлянского.	мм рт. ст.	-

Продолжение таблицы В.1

Полное обозначение	Описание	Единицы измерения	Краткое обозначение (если есть)
P_{gh}	Клубочковое гидростатическое давление.	мм рт. ст.	-
P_{go}	Клубочковое осмотическое давление.	мм рт. ст.	-
P_f	Фильтрационное давление в сети капилляров.	мм рт. ст.	-
P_{mf}	Среднее наполняющее давление.	мм рт. ст.	-
P_{ma}	Среднее артериальное давление.	мм рт. ст.	-
P_{ra}	Давление в правом предсердии.	мм рт. ст.	-
R_a	Артериальное сопротивление.	мм рт. ст.*л/мин	-
$R_{aa\text{-}ss}$	Сопротивление афферентной артериолы.	мм рт. ст./(л/мин)	-
R_{ba}	Базовое артериальное сопротивление.	мм рт. ст./(л/мин)	-
R_{bv}	Базовое венозное сопротивление.	мм рт. ст./(л/мин)	-
R_{ea}	Сопротивление эфферентной артериолы.	мм рт. ст./(л/мин)	-
rsna	Почечная симпатическая нервная активность.	-	*rsna*
R_{tp}	Общее периферическое сопротивление.	мм рт. ст./(л/мин)	R_r
R_r	Почечное сосудистое сопротивление.	мм рт. ст./(л/мин)	-
R_{vr}	Сопротивление венозному возврату.	мм рт. ст./(л/мин)	-
$sigma_{tgf}$	Сигнал тубулогломерулярной обратной связи.	-	Σ_{tgf}
T_{adh}	Константа времени для секреции антидиуретического гормона.	мин	-
T_{al}	Константа времени для секреции альдостерона.	мин	-
T_r	Константа времени для секреции ренина.	мин	-
vas	Васкуляризация.	-	-
vas_f	Скорость увеличения васкуляризации.	-	-

Окончание таблицы В.1

Полное обозначение	Описание	Единицы измерения	Краткое обозначение (если есть)
vas_d	Скорость снижения васкуляризации.	-	-
V_b	Объем крови.	л	-
V_{ecf}	Объем внеклеточной жидкости.	л	-
Модель артериального дерева			
α	Коэффициент кориолиса.	-	-
E	Модуль Юнга.	дин/см^2	-
h_i, i = 1,…,55	Толщина стенки сосуда.	см	-
ρ	Плотность крови.	г/см^2	-
ν	Коэффициент вязкости крови.	Па	-
$A_i(t, z)$, i = 1,…,55	Площадь поперечного сечения i-того сосуда.	см^2	-
K_r	Коэффициент трения крови о стенки сосудов.	Па	-
l_i, i = 1,…,55	Длина i-того сосуда.	см	-
$p_i(t, z)$, i = 1,…,55	Давление в i-том сосуде.	мм рт. ст.	P_A
$Q_i(t, z)$, i = 1,…,55	Кровоток в i-том сосуде.	мл/с	Q_{HA},Q_{AC},Q_{AR}
$R_i(t, z)$, i = 1,…,55	Радиус i-того сосуда.	см	-
$r_i(t, z)$, i = 1,…,55	Сопротивление кровотоку через i-тый сосуд артериального дерева.	мм рт. ст./(мл/с)	R_A,R_{AR}

ПРИЛОЖЕНИЕ Г

Таблица Г.1 – Расшифровка параметров модели Гайтона, упомянутых в работе.

Обозначение	Описание	Единицы измерения
AHM	Эффект антидиуретического гормона, отношение к норме.	-
AM	Эффект альдостерона, отношение к норме.	-
ANM	Эффект ангиотензина, отношение к норме.	-
AMM	Сужение сосудов мышечных тканей, отношение к норме.	-
ANU	Непочечный эффект ангиотензина, отношение к норме.	-
ARM	Сосудосуживающий эффект авторегуляции, отношение к норме.	-
AU	Активность автономной системы, отношение к норме.	-
AUH	Автономная стимуляция сердца, отношение к норме.	-
AUM	Суживающее влияние симпатической нервной системы на артерии.	-
AVE	Суживающиее влияние симпатической нервной системы на вены.	-
BFM	Мышечный кровоток	л/мин
BFN	Немышечный кровоток.	л/мин
CKE	Внеклеточная концентрация калия.	ммол/л
CNA	Внеклеточная концентрация натрия.	ммол/л
CP	Давление в капиллярах.	мм рт. ст.
CPP	Концентрация белков в плазме.	г/л
HM	Гематокрит, отношение к норме.	-
HPL	Степень гипертрофии левого желудочка.	-
HPR	Степень гипертрофии правого желудочка.	-
PA	Давление в аорте.	мм рт. ст.
PAM	Эффект артериального давления на растяжение сосудов.	-
PMO	Парциальное давление кислорода в мышечных клетках.	мм рт. ст.
PLA	Давление в левом предсердии.	мм рт. ст.
POT	Парциальное давление кислорода в немышечных клетках.	мм рт. ст.
PPA	Давление в легочных артериях.	мм рт. ст.
PRA	Давление в правом предсердии.	мм рт. ст.
PVS	Давление в венах.	мм рт. ст.
QAO	Поток крови из артерий большого круга в вены.	л/мин
QRO	Поток крови из правого желудочка в артерии малого круга.	л/мин
QPO	Поток крови из вен малого круга в левый желудочек.	л/мин
QLO	Поток крови из левого желудочка в артерии большого круга.	л/мин
QVO	Поток крови из вен большого круга в правый желудочек.	л/мин
RAR	Базовое сопротивление немышечных и непочечных артерий.	мм рт. ст./(л/мин)

Окончание таблицы Г.1

Обозначение	Описание	Единицы измерения
RAM	Базовое сопротивление мышечных сосудов.	мм рт. ст./(л/мин)
RBF	Почечный кровоток.	л/мин
REK	Масса почки, отношение к норме.	-
RSM	Сопротивляемость мышечных сосудов.	мм рт. ст./(л/мин)
RSN	Сопротивляемость немышечных и непочечных сосудов.	мм рт. ст./(л/мин)
RVS	Венозное сопротивление.	мм рт. ст./(л/мин)
TVD	Потребление воды.	л/мин
VB	Общий объем крови.	л
VRC	Объем эритроцитов.	л
VP	Объем плазмы.	л
VIM	Вязкость крови, отношение к норме.	-
VUD	Отток мочи.	л/мин
VAS	Объем крови артерий большого круга.	л
VVE	Уменьшенный объем сосудов из-за симпатической стимуляции.	л
VV7	Увеличение объема сосудов связанное с вазодилатацией.	л

ПРИЛОЖЕНИЕ Д

УТВЕРЖДАЮ
Зам. директора Федерального
государственного бюджетного
учреждения науки «Институт общей
генетики им. Н.И. Вавилова»
Российской академии наук

д.б.н. С.К. Абилев

"__" _____ 2016 г.

АКТ О ВНЕДРЕНИИ

Результатов диссертационной работы Киселева И. Н.
“Модульное моделирование биологических систем на примере сердечно-сосудистой системы человека”

Комиссия в составе:
Председатель: члена-корреспондента РАН Макеева В.Ю.
Члены комиссии: к.ф.-м.н. Урошлева Л.А., к.ф.-м.н. Касьянова А.С.

составили настоящий акт о том, что научные результаты диссертационного исследования Киселева Ильи Николаевича, реализованные в виде программных модулей для платформы BioUML, используются в Лаборатории системной биологии и вычислительной генетики ИОГен РАН для работы по теме 53.11 «Вычислительный анализ данных, полученных с помощью установок секвенирования нового поколения, для определения последовательности генов и структурных вариаций в геномах» для моделирования фенотипических последствий генетических вариантов.

Председатель комиссии: /Макеев В.Ю./

Члены комиссии: /Урошлев Л.А./

/Касьянов А.С./

Федеральное государственное бюджетное
учреждение науки
ИНСТИТУТ МАТЕМАТИКИ
им. С.Л. Соболева
Сибирского отделения
Российской академии наук
(ИМ СО РАН)

630090 Новосибирск, пр. Академика Коптюга, 4
Для телеграмм: Новосибирск, 90, Математика
Тел.: (8-383) 333-28-92. Факс: (8-383) 333-25-98
E-mail: im@math.nsc.ru

17.11.2016 № 15302-2-2434.3
На № _____ от _____

УТВЕРЖДАЮ
директор Федерального государственного бюджетного учреждения науки Института математики им. С. Л. Соболева Сибирского отделения Российской академии наук академик РАН
С. С. Гончаров

"17" 11 2016 г.

АКТ О ВНЕДРЕНИИ

Результатов диссертационной работы Киселева И. Н.
“Модульное моделирование биологических систем на примере сердечно-сосудистой системы человека”

Комиссия в составе:

председатель Блохин А. М., д.ф-м.н., заведующий лабораторией вычислительных проблем задач математической физики института математики им. С.Л. Соболева СО РАН,

члены комиссии Голубятников В.П., д.ф-м.н., главный научный сотрудник,
Бибердорф Э.А., к.ф.-м.н., старший научный сотрудник,

составила настоящий акт о том, что научные результаты диссертационного исследования Киселева Ильи Николаевича, реализованные в виде программных модулей для платформы BioUML, использованы в лаборатории вычислительных проблем задач математической физики в научных исследованиях по теме междисциплинарного интеграционного проекта фундаментальных исследований СО РАН №91. Разработанные программные модули включают оригинальные методы для объединения частных моделей биологических систем в комплексные (модульные) модели и проведения численных расчетов, в том числе для случая различных математических формализмов объединяемых моделей. Эти методы позволили реализовать в платформе BioUML модель артериального дерева из 55 крупнейших сосудов, разрабатываемую в институте математики им. С. Л. Соболева, численно исследовать ее поведение и использовать как часть более крупной модульной модели сердечно-сосудистой системы, включающей также блоки сердца и почки.

Председатель комиссии: _____ **д.ф.-м.н. А.М. Блохин**

Члены комиссии: _____ **д.ф.-м.н. В.П. Голубятников**

_____ **к.ф.-м.н. Э.А. Бибердорф**