На правах рукописи

Ступников Сергей Александрович

МЕТОДЫ И СРЕДСТВА ВЕРИФИЦИРУЕМОЙ ИНТЕГРАЦИИ НЕОДНОРОДНЫХ ДАННЫХ

2.3.5 – Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей

АВТОРЕФЕРАТ

диссертации на соискание ученой степени

доктора технических наук

Москва – 2026

Работа выполнена в Федеральном государственном учреждении «Федеральный исследовательский центр «Информатика и управление» Российской академии наук» (ФИЦ ИУ РАН)

Научный консультант: **Виктор Николаевич Захаров**
доктор технических наук,
ученый секретарь ФИЦ ИУ РАН

Официальные оппоненты: **Позин Борис Аронович**
доктор технических наук,
главный научный сотрудник ИСП РАН

Махортов Сергей Дмитриевич
доктор физико-математических наук,
заведующий кафедрой программирования и информационных технологий ФГБОУ ВО «ВГУ»

Мунерман Виктор Иосифович
доктор технических наук,
доцент кафедры прикладной математики и информатики СмолГУ

Ведущая организация: Национальный исследовательский университет «Высшая школа экономики»

Защита диссертации состоится 16 декабря 2026 г. в 15 часов 00 минут на заседании диссертационного совета 24.1.224.04 при ФИЦ ИУ РАН по адресу: 119333, г. Москва, ул. Вавилова, 44, корп. 2.

С диссертацией можно ознакомиться в библиотеке ФИЦ ИУ РАН по адресу: 119333, г. Москва, ул. Вавилова, 44, корп. 2 и на сайте официальном сайте ФИЦ ИУ РАН http://www.frccsc.ru.

Отзывы на автореферат в двух экземплярах, заверенные печатью учреждения, высылать по адресу: 119333, г. Москва, ул. Вавилова, 44, корп. 2, ученому секретарю диссертационного совета 24.1.224.04.

Автореферат разослан 2026 г.

Ученый секретарь
диссертационного совета 24.1.224.04 Р.В. Разумчик

Общая характеристика работы

Актуальность темы исследования

Роль данных в различных областях деятельности человека – научных исследованиях, здравоохранении, образовании, промышленности и т.д. – непрерывно растет в последние годы. Укрепляется новая парадигма в науке и информационных технологиях, связанная с интенсивным использованием данных – так называемая *четвертая парадигма*[1], объединяющая теорию, эксперимент и симуляцию. Данные поступают от инструментов или симуляторов, обрабатываются (очищаются) программным обеспечением, а затем ученые производят анализ извлеченной информации. Собственно научное знание образуется в процессе интенсивного анализа накапливаемых данных, приводящего в конечном счете к извлечению знаний из данных. Данные используются для решения задач в различных предметных областях (называемых *областями с интенсивным использованием данных*) – астрономии, материаловедении, управлении землепользованием, биологии, медицине, физике и других[2].

Основная проблема новой парадигмы состоит в том, что объемы и разнообразие поступающих данных значительно превышают возможности современных методов и средств хранения, обработки и анализа данных. Наблюдается экспоненциальный рост объема накапливаемых экспериментальных (наблюдательных) данных, оформляемых в виде источников данных, которые концентрируются в *исследовательских инфраструктурах данных*, предоставляющих доступ к данным, вычислительным сервисам и процессам. Такие инфраструктуры предназначены для поддержки полного цикла управления и обработки данных, от сбора до хранения и анализа. Примеры наиболее крупных исследовательских инфраструктур – это EUDAT[3] (совместная инфраструктура для синхронизации, обмена, хранения, поиска, репликации, предоставления доступа к данным и вычислений над ними) и PRACE[4] (высокопроизводительные вычисления). Важность развития подобных инфраструктур в России подчеркивается, в частности, дорожной картой по развитию высокопроизводительных вычислений, алгоритмов искусственного интеллекта, грид-технологий и суперкомпьютерной инфраструктуры[5].

Однако, существующие инфраструктуры данных достаточно редко предоставляют комплекс средств, обеспечивающий извлечение максимальной выгоды из инвестиций в сбор данных и исследования. Для того, чтобы подвигнуть провайдеров данных и инфраструктур преодолеть этот разрыв, академическим сообществом Force11 были разработаны и опубликованы принципы управления данными FAIR Data Principles[6]. В соответствии с принципами FAIR, данные должны быть *обнаруживаемыми* (данные аннотированы достаточным количеством метаданных с уникальными и постоянными идентификаторами), доступными (метаданные и данные должны понятны и читаемы как людьми, так и компьютерами, и размещены в хранилище с разделяемым доступом), обеспечивать *совместное использование* (данные или инструменты из разных источников могут совместно работать и интегрироваться друг с другом; для описания метаданных должен использоваться общий формальный язык представления знаний) и *повторное использование* (данные и коллекции прозрачно лицензированы и содержат

---

[1] Hey T., Tansley S., Tolle K. The Fourth Paradigm: Data-Intensive Scientific Discovery. — Microsoft Research, 2009.

[2] Калиниченко Л. А., Вольнова А. А., Гордов Е. П. и др. Проблемы доступа к данным в исследованиях с интенсивным использованием данных в России // Информатика и её применения, 2016. Т. 10. С. 2–22.

[3] EUDAT Collaborative Data Infrastructure. — EUDAT, 2026. https://eudat.eu/

[4] PRACE – Partnership for Advanced Computing in Europe. — PRACE, 2026. http://www.prace-ri.eu/

[5] Распоряжение Правительства Российской Федерации от 12 марта 2026 г. № 482-р. — Москва, 2026.

[6] Wilkinson M. D., Dumontier M., Aalbersberg I. J. et. al. The fair guiding principles for scientific data management and stewardship // Scientific Data, 2016. Vol. 3. No. 1. P. 160018. doi: 10.1038/sdata.2016.18.

метаданные о своем происхождении). Повышение уровня реализации принципов FAIR в инфраструктурах данных облегчает и упрощает процесс обнаружения, оценки и повторного использования данных. Для этого развиваются новые информационные технологии, в которых данные становятся доминирующим фактором, новые подходы к концептуализации, организации и реализации информационных систем. При этом требуется не только создание методов и средств оперирования данными, объемы которых выходят за рамки возможностей современных технологий баз данных, но и разработка новых подходов, позволяющих справляться с разнообразием массово и хаотично развивающихся *моделей данных*[7], поскольку одна из важнейших проблем инфраструктур данных – это очень большая неоднородность данных.

Разнообразие моделей данных включает традиционную реляционную модель (диалекты языка SQL) и ее объектно-реляционные расширения, объектные модели (ObjectStore), тензорные и графовые модели, семантические модели (как RDF и OWL), модели слабоструктурированных данных (XML, JSON), модели потоков работ и бизнес-процессов (XPDL, BPEL, BPMN), различные NoSQL модели: документные модели (системы MongoDB, Couchbase, CouchDB), модели с семействами столбцов (системы Cassandra, HBase), модели «ключ-значение» (системы Redis, Memcached). Эти модели предоставляют существенно различные языки манипулирования и запросов для доступа к данным и их изменения.

Наблюдаемый в настоящий период развития информационных технологий рост числа новых моделей данных сопровождается другой тенденцией - накоплением использующих подобные модели источников данных, число которых растет экспоненциально.

Такой рост вызывает все увеличивающуюся потребность совместного использования (*интеграции*) модельно неоднородных данных и сервисов в различных применениях, а также их повторного использования и композиции при решении новых задач в науке или промышленности. Различные по семантике и структурированности данные могут быть необходимы при решении одной задачи. Вопросы интеграции данных в инфраструктурах данных с течением времени становятся все более и более важными и сложными. Интеграция данных делает их более обнаруживаемыми, доступными и пригодными для совместного и повторного использования, чем разнородные данные, хранимые в никак не взаимодействующих источниках. Фактически, интеграция данных – это необходимое предусловие для решения задач анализа данных, в частности, в рамках фундаментальных научных исследований с использованием высокопроизводительных вычислительных инфраструктур.

Может быть выделено несколько *уровней интеграции данных*. Эти уровни соответствуют уровням моделей в архитектуре метамоделирования MOF[8]. Модель определяется в соответствии с семантикой некоторой *метамодели*, при этом говорят, что модель *соответствует* или *конформна* (conforms to) метамодели. Стандартом OMG MOF определена четырехуровневая архитектура моделей: модели уровня M0 описывают объекты реального мира, модели уровня M1 называются обычно *схемами*, модели уровня M2 представляют собой собственно модели данных, модели уровня M3 – мета-метамодели, предназначенные для описания моделей уровня M2.

В соответствии с уровнями моделей, могут быть выделены следующие *уровни интеграции данных*, от более высокого к более низкому: интеграция моделей данных (унификация), сопоставление и интеграция схем данных (интеграция метаданных) и собственно интеграция данных. На уровне моделей данных сопоставляются элементы

---

[7] В данной работе под термином «модель данных» понимается комбинация языка определения данных (ЯОД) и языка манипулирования данными (ЯМД). Таким образом модель данных – это язык, предназначенный для описания схем баз данных и сервисов и запросов к ним.

[8] OMG Meta Object Facility (MOF) Core Specification. version 2.5.1. — OMG, 2019. https://www.omg.org/spec/MOF/2.5.1/PDF

языков определения и манипулирования данными. На уровне схем сопоставляются структуры (типы) данных, из атрибуты и методы. На уровне собственно данных определяются правила трансформации коллекций данных, их экземпляров и значений атрибутов. Обычно завершение интеграции на более высоком уровне является необходимым предусловием для интеграции на более низких уровнях.

Наиболее зрелый уровень интеграции обеспечивается в системах интеграции данных, таких как *предметные посредники* и *хранилища данных*.

Предметные посредники[9] – специальный вид программного обеспечения, образующий промежуточный слой между пользователем (приложением) и ресурсами. Создание предметного посредника состоит из двух фаз. На фазе *консолидации* происходит концептуальное моделирование предметной области посредника силами некоторого экспертного сообщества. Во время *операционной* фазы провайдеры источников данных регистрируют их в посреднике, т.е. представляют спецификации ресурсов в терминах концептуальной схемы посредника. Предметные посредники реализуют *виртуальную интеграцию данных.* Запросы (программы) пользователя определяются в некоторой унифицирующей модели данных. Эти запросы декомпозируются во множества подзапросов, которые передаются на разнородные источники и исполняются. Источники соединяются с посредником при помощи *адаптеров*, которые преобразуют запросы в языки исходных моделей данных и преобразуют результаты исполнения запросов из исходной модели данных в целевую. Результаты исполнения запросов передаются через адаптеры в посредник, соединяются и передаются пользователю. Одна из современных тенденций применения технологии предметных посредников – их конструирование над озерами данных.

Хранилища данных, реализующие *материализованную интеграцию данных*, в настоящее время являются одной из основных составляющих инфраструктур поддержки систем бизнес-аналитики. Данные извлекаются из различных коллекций, преобразуются к единой схеме хранилища, интегрируются и загружаются в хранилище. Над хранилищем выстраивается слой приложений, осуществляющих анализ данных (например, при помощи методов математической статистики, машинного обучения и др.) и выдающих результат пользователю в необходимой форме.

Ввиду роста неоднородности источников данных, их количества и объемов данных, актуальными остаются вопросы разработки методов спецификации и реализации интеграции данных в масштабируемых инфраструктурах распределенного хранения и обработки данных, подобных Apache Hadoop.

Для интеграции неоднородных источников данных и сервисов FAIR-инфраструктуры данных могут поддерживать как виртуальную, так и материализованную интеграцию данных, а также их комбинацию, когда хранилища данных рассматриваются как источники для предметных посредников.

В общем случае источники данных неоднородны, представлены в различных моделях данных. А потому, при их интеграции с использованием систем виртуальной или материализованной интеграции для однородного представления семантики требуется приведение различных моделей данных к унифицированному виду в рамках некоторой унифицирующей модели данных, которая называется *канонической*. Такое семантическое преобразование разномодельных спецификаций к унифицированному виду в канонической модели нуждается в специальных методах и инструментальных средствах.

После решения вопросов интеграции на уровне моделей в системах виртуальной интеграции необходима интеграция на уровне схем и построение представлений (взглядов), связывающих схемы источников данных и схему посредника. При этом

---

[9] Kalinichenko L. A., Briukhov D. O., Martynov D. O. et al. Mediation framework for enterprise information system infrastructures: application-driven approach // Proceedings of the Ninth International Conference on Enterprise Information Systems, 2007. Vol. DISI. P. 246–251.

необходимо семантическое сопоставление элементов схем источников данных и элементов схемы посредника.

Идеальным описанием комбинированного процесса интеграции схем и собственно данных в хранилищах можно было бы считать совокупность декларативных правил (подобную программе на языке Datalog), такой подход называется обмен данными (data exchange). Однако, на практике, процесс интеграции данных представляет собой набор операций трансформации данных, представленных в SQL или подобном ему языке, причем важным является порядок исполнения операций. Обычно интеграция данных организуется в виде потоков работ.

Обычно разработчикам систем интеграции данных на всех уровнях приходится полагаться на интуитивные приемы сопоставления элементов моделей, без точного учета семантики используемых моделей и подтверждения правильности выполняемых преобразований. Такой учет возможен только при помощи определения формальной семантики моделей данных и *верификации процесса интеграции данных на всех уровнях*, т.е. формальной доказательной проверки на соответствие заданным требованиям.

Существует два основных метода верификации сложных систем: *проверка моделей* (model checking) и *доказательство теорем* (theorem proving). Верификация моделей основана на построении конечной модели системы и проверке выполнения заданных свойств в этой модели. В силу конечности модели, такая верификация всегда может быть выполнена автоматически. Основная проблема верификации моделей состоит в том, что для проверки свойств сложных систем далеко не всегда возможно построить конечную модель приемлемого размера, и даже просто конечную модель.

Доказательство теорем представляет собой метод, в котором система описывается на языке некоторой формальной логики, а требуемые свойства системы описываются как формулы логики. Доказательство теорем представляет собой процесс вывода заданных формул из аксиом логики с использованием правил вывода. Некоторые выводы могут быть получены автоматически, для получения более сложных выводов используются средства интерактивного доказательства.

В контексте методов верификации систем формализуется одно из наиболее важных понятий в области формальных методов – понятие *уточнения*[10]. В настоящей работе уточнение понимается следующим образом: система *B* уточняет систему *A*, если пользователь может использовать систему *B* вместо системы *A*, не замечая факта замены *A* на *B*. Уточнение формализовано в различных языках спецификаций, и изначально предназначалось для разработки информационных систем «сверху-вниз» методом пошагового уточнения (stepwise refinement). При этом система описывается на высоком уровне абстракции, затем уточняется серией конкретизаций вплоть до кода на языке программирования. Каждая последующая конкретизация системы является более детальной, чем предыдущая. При этом каждый шаг уточнения формально доказывается, и полученная в результате система обладает всеми необходимыми свойствами.

Формальная верификация с успехом применялась в большом количестве проектов по разработке систем в различных областях: медицине, ядерной энергетике, телефонии, транспорте, космической технике, разработке микропроцессорной техники, верификации протоколов и пр.[11] Применение формальных методов доказательства сопряжено с известными сложностями, однако при разработке современных сложных информационных систем стоимость исправления ошибки на этапе эксплуатации превышает стоимость исправления ошибки на этапе проектирования в 10-100 раз[12]. Поэтому издержки на привлечение специалиста для интерактивного доказательства

---

[10] Wirth N. Program development by stepwise refinement // Commun. ACM, 1971. Vol. 14. No 4. P. 221–227.

[11] Butler M., Korner P., Krings S. et al.. The first twenty-five years of industrial use of the B-method // Lecture Notes in Computer Science, 2020, Vol. 12327. P. 189–209.

[12] White N., Matthews S., Chapman R. Formal verification: Will the seedling ever flower? // Philosophical Transactions A, 2017. Vol. 375. P. 20150402.

корректности представляются оправданными, особенно при разработке систем, ошибки которых критичны для безопасности функционирования человека (safety-critical systems).

Все вышесказанное позволяет сделать вывод о необходимости и актуальности разработки методов и средств верифицируемой интеграции неоднородных данных.

В 1980-90х годах Л.А. Калиниченко было заложено направление сохраняющих семантику отображений моделей данных[13]. Был предложен метод аксиоматического расширения реляционной модели данных для выражения различных видов зависимостей и построения эквивалентных отображений с сетевой и иерархической моделями данных. Предложен принцип коммутативного отображения моделей данных, сводящийся к коммутативности диаграмм отображения схем и последовательностей операций языка манипулирования данными, и его усовершенствование с использованием формального языка для определения семантики моделей и уточнения моделей вместо эквивалентного отображения.

В мире в области определения формальной семантики моделей данных существенные результаты были получены А.В. Замулиным, М.Ш. Цаленко, А.В. Манциводой, А.А. Савиновым, K. Lano, M. Butler, R. France, S. Kim, D. Carrington, B. Beckert, P. Schmitt, P. Facon, E. Meyer, J. Souquieres, H. Ledang, W. Conen, R. Fikes, T. Mossakowski, J. de Bruijn, M. Schneider, E. Franconi, P. Guagliardo, V. Benzaken. Конструирование трансформаций моделей исследовались С.В. Зыкиным, M.D. del Fabro, P. Valdurez, R. Falleri, M. Wimmer, Z. Diskin, A. Bohannon, V.A. Bollati. Методы и средства преобразования схем данных развивались группой P. Atzeni. Методы и средства верификации трансформаций моделей исследовались Б. Улитиным, Э. Бабкиным, K. Lano, D. di Ruscio, K. Stenzel, F. Buttner, H. Ledang, J. Cabot, Z. Cheng, P.G. Larsen, P. Vassiliadis.

Данная работа нацелена на целостное решение *проблемы верификации интеграции неоднородных данных на всех трех уровнях: моделей данных, схем данных и собственно данных* с обеспечением формальной методической и программной базы в отличие от известных работ, уделяющих внимание отдельным аспектам верификации интеграции данных на отдельных уровнях. В частности, работа развивает и расширяет основные составляющие заложенного Л. А. Калиниченко направления: формализацию методов, формирование доказательной базы, программную реализацию методов, применение методов на широком спектре моделей данных. Работа объединяет результаты, полученные автором в области верификации интеграции данных. Методы, развиваемые в данной работе, призваны служить формальными основаниями для совместного использования (мета)данных, интеграции структурированных данных (данных, конформных некоторой схеме, определяющей типы (структуры) данных) и повторного использования в FAIR-инфраструктурах данных.

Цель диссертационной работы состоит в разработке методов и средств верифицируемой интеграции неоднородных данных, обеспечивающих формальное доказательство корректности интеграции.

Для реализации этой цели необходимо решить следующие основные **задачи**:

1. Разработать метод конструирования сохраняющих семантику отображений моделей данных, обеспечивающий формальное доказательство корректности интеграции моделей данных.

2. Разработать метод верификации интеграции схем данных в системах виртуальной интеграции данных, обеспечивающий формальное доказательство корректности интеграции схем данных.

---

[13] Калиниченко Л. А. Методы и средства интеграции неоднородных баз данных. — М.: Наука, 1983.

3. Разработать метод верификации интеграции данных в системах материализованной интеграции данных, обеспечивающий формальное доказательство корректности интеграции данных.

4. Разработать схемы функциональной структуры сред решения задач над неоднородными источниками данных с поддержкой верифицируемой интеграции данных.

5. На основе полученных методов разработать программные средства их автоматизации, включая: средства поддержки верифицируемой интеграции моделей данных, верификации интеграции схем, верификации интеграции данных.

Объектом исследования является процесс интеграции неоднородных данных.

Предметом исследования выступают методы и программные средства верифицируемой интеграции неоднородных данных.

Методы исследования

При решении поставленных задач использовались методы математической логики первого порядка и теории множеств, методы формальной спецификации предметных областей, методы теории уточнения спецификаций, методы денотационной и аксиоматической семантики, методы отображения и трансформации моделей данных и схем данных.

Основные положения, выносимые на защиту

1. Метод конструирования сохраняющих семантику отображений моделей данных, сочетающий формальное определение синтаксиса и семантики моделей данных, интеграцию синтаксисов исходной и целевой моделей, создание расширений целевой модели, автоматизированное конструирование трансформаций моделей, и основанный на уточнении семантических спецификаций, обеспечивает формальное доказательство корректности интеграции данных на уровне моделей данных и применим для интеграции широкого спектра моделей данных.

2. Метод верификации интеграции схем данных, основанный на уточнении спецификаций типов и запросов, обеспечивает формальное доказательство корректности интеграции данных в системах виртуальной интеграции данных.

3. Метод верификации интеграции данных, основанный на доказательстве истинности интегрирующих соотношений, связывающих исходную и целевую базы данных, после исполнения программы интеграции данных, обеспечивает формальное доказательство корректности интеграции данных в системах материализованной интеграции данных.

4. С использованием схем функциональной структуры среды комбинированной виртуально-материализованной интеграции данных и основанной на правилах мультидиалектной среды как методологической основы, осуществима разработка программных комплексов решения аналитических задач над множествами неоднородных источников данных и сервисов.

Научная новизна

1. Разработан новый метод конструирования сохраняющих семантику отображений моделей данных, и в отличие от известных, сочетающий формальное определение синтаксиса и семантики моделей данных, интеграцию синтаксисов исходной и целевой моделей, создание расширений целевой модели, автоматизированное конструирование трансформаций моделей, верификацию корректности отображения моделей. Метод обеспечивает доказательство корректности интеграции моделей данных с использованием формально определенного уточнения спецификаций.

2. Разработан новый метод верификации интеграции схем данных в системах виртуальной интеграции данных, основанный, в отличие от известных, на уточнении

абстрактных типов данных и запросов пользователя к системе интеграции. Метод обеспечивает формальное доказательство корректности интеграции схем данных.

3. Разработан новый метод верификации интеграции данных в системах материализованной интеграции данных. В отличие от известных методов, формальная семантическая спецификация сопоставляется высокоуровневой программе интеграции данных, и необходимые свойства программы интеграции проверяются с использованием инструментальных средств доказательства. Метод обеспечивает формальное доказательство корректности интеграции на уровне данных.

4. Разработаны новые программные средства автоматизации методов интеграции данных, включающие средства поддержки верифицируемой интеграции моделей данных, верификации интеграции схем, верификации интеграции данных.

5. Разработаны новые схемы функциональной структуры сред решения задач над неоднородными источниками данных с поддержкой верифицируемой интеграции данных как методологическая основа для разработки систем решения аналитических задач над множествами неоднородных источников данных и сервисов.

Диссертационное исследование **соответствует следующим пунктам паспорта специальности** 2.3.5 «Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей»:

– модели, методы и алгоритмы проектирования, анализа, трансформации, верификации и тестирования программ и программных систем»;

– языки программирования и системы программирования, семантика программ»;

– модели, методы, архитектуры, алгоритмы, языки и программные инструменты организации взаимодействия программ и программных систем;

– модели и методы создания программ и программных систем для параллельной и распределенной обработки данных, языки и инструментальные средства параллельного программирования.

Теоретическая и практическая значимость

Разработанные методы позволяют определять семантику, отображения и трансформации моделей данных различных классов: объектных, реляционных, онтологических, графовых, тензорных, процессных; составляют собой формальные основания для автоматизированной верификации процессов интеграции неоднородных данных. Разработанные методы и средства нацелены на применение при разработке систем интеграции неоднородных данных, необходимых как при решении задач в исследовательских инфраструктурах в различных предметных областях, так и в промышленных информационных системах. Разработан комплекс программ, нацеленный на поддержку методов автоматизированной верификации интеграции данных. Полученные результаты диссертационного исследования были реализованы или использованы в научно-исследовательской и практической деятельности ФГБНУ ФИЦ «Почвенный институт им. В.В Докучаева», Институт астрономии РАН, Институт металлургии и материаловедения им. А.А. Байкова Российской академии наук (ИМЕТ РАН), ООО «Паллада». Материалы диссертационной работы используются в курсе «Виртуальная интеграция неоднородных данных и унификация моделей данных» совместной магистерской программы факультета Вычислительной математики и кибернетики МГУ им. М. В. Ломоносова и ФИЦ ИУ РАН «Большие данные: инфраструктуры и методы решения задач». Результаты диссертационного исследования были реализованы или использованы в рамках научно-исследовательских работ Российского научного фонда, Российского фонда фундаментальных исследований, Минобрнауки РФ, Отделения информационных технологий и вычислительных систем (Отделения нанотехнологий и информационных технологий) РАН, Президиума РАН, выполненных в ФИЦ ИУ РАН, МГУ им. М. В. Ломоносова, Институте астрономии РАН,

Национальном исследовательском ядерном университете «МИФИ», ФГБНУ ФИЦ «Почвенный институт им. В.В Докучаева».

Достоверность и обоснованность полученных в диссертации результатов подтверждена применением формальных методов определения семантики и отображений моделей данных, а также средств автоматизированного доказательства, опирающихся на логику предикатов первого порядка и теорию множеств; использованием теоретических результатов и опытом практической реализации результатов исследования в ряде НИР, сопоставлением результатов исследования с родственными зарубежными и отечественными работами; обсуждением результатов диссертационного исследования на международных и всероссийских научных мероприятиях.

Апробация результатов работы осуществлена на международных и всероссийских научных конференциях, в том числе:

– Международная конференция «Аналитика и управление данными в областях с интенсивным использованием данных» DAMDID/RCDL (Обнинск, 2015; Ершово, 2016; Москва, 2017, 2018, 2021; Воронеж, 2020; Санкт-Петербург, 2022, 2025; Нижний Новгород, 2024).

– Второй симпозиум по моделирования к программам M2P (Финляндия, 2020).

– Конференция «Наука о данных в Европе» (Сербия, 2020).

– Международная конференция «Иванниковские чтения» (Великий Новгород, 2019).

– Международная конференция по базам данных и информационным системам ADBIS (Словакия, 2002; Венгрия, 2004; Греция, 2006; Финляндия; 2008; Латвия, 2009; Австрия, 2011; Польша, 2012; Италия, 2013; Сербия, 2014; Словения, 2019).

– Научная сессия «Исследовательские и образовательные компетенции для цифровой экономики» ОНИТ РАН (Москва, 2018).

– Конференция «Технологии баз данных» (Москва, 2016).

– Конференция «Центры коллективного пользования и уникальные научные установки в организациях, подведомственных ФАНО России» (Москва, 2015).

– Международная конференция по Открытым и большим данным OBD (Италия, 2015).

– Российская конференция по электронным библиотекам RCDL (Суздаль, 2006; Казань, 2010; Воронеж, 2011; Переславль, 2012; Ярославль, 2013; Дубна, 2014).

– Семинар Московской секции ACM SIGMOD (Москва, 2009).

– V Конференция Итальянской секции ассоциации по информационным системам itAIS (Франция, 2008).

– 21-я конференция «Научная информация для общества – из настоящего в будущее» CODATA (Украина, 2008).

– II Научная сессия ИПИ РАН «Проблемы и методы информатики» (Москва, 2005).

– Симпозиум «Развитие исследований баз данных в Восточной Европе» (Германия, 2003).

– 5-я Международная Балтийская конференция по азам данных и информационным системам BalticDB&IS (Эстония, 2002).

– Международная научно-практическая конференция по программированию УкрПРОГ (Украина, 2002).

– XXIV Конференция молодых ученых МГУ им. М.В. Ломоносова (Москва, 2002).

– Научные семинары ФИЦ ИУ РАН, ВМК МГУ им. М.В. Ломоносова.

Материалы диссертационной работы использовались при формировании курса «Виртуальная интеграция неоднородных данных и унификация моделей данных»

в рамках на магистерской программы «Большие данные: инфраструктуры и методы решения задач» факультета вычислительной математики и кибернетики МГУ им. М. В. Ломоносова.

Публикации

По теме диссертации автором опубликовано 66 статей и 2 монографии. Из них 23 статьи опубликованы в журналах из списка ВАК (К1 и К2), 18 статей - в изданиях, индексируемых в международных базах научных изданий Web of Science и Scopus (Q1-Q4). Получено 12 свидетельств о государственной регистрации программ для ЭВМ.

Личный вклад

Все выносимые на защиту результаты получены автором лично. В работах [19][40][54][36][64][65] разработан метод унификации моделей данных, на примерах проиллюстрированы его этапы, проведено обсуждение применения метода для интеграции данных в исследовательских инфраструктурах. В работе [34] изложены основные идеи реализации метода унификации моделей данных с использованием движимой моделями инженерии, выделены преимущества этой реализации по сравнению с реализацией на основе метакомпиляции и переписывания термов. В работе [55] изложен краткий обзор методов унификации и определения формальной семантики для сервисных и процессных моделей. В работах [63][17] разработан метод автоматизации построения трансформаций моделей. В работах [1][22][58][18] определено и проиллюстрировано понятие редукта АТД канонической модели данных, формализованы определения соединения и пересечения АТД, определены общие правила интерпретации формул объектного исчисления, определена формальная семантика конъюнктивных запросов. В работе [68] разработаны программные средства синтаксического и семантического анализа спецификаций канонической модели. В работе [23] разработан метод представления конструкций языков UML и OCL в канонической модели данных. В работах [53][14] были созданы предикативные спецификации инвариантов в канонической модели данных. В работе [24] определена формальная денотационно-аксиоматическая семантика ядра канонической объектной модели данных. В работах [21][62] разработан метод отображения спецификаций, выраженных средствами ядра канонической модели, в язык AMN. В работах [67][2][57][26] определена формальная семантика ядра канонической модели процессов, определены общие принципы построения расширений канонической модели процессов, определено отображение расширенных сетей потоков работ в AMN, проиллюстрирован на примере метод доказательства корректности уточняющего расширения канонической модели процессов. В работах [15][59] проведен анализ семантик логических программ. В работе [60] рассмотрены принципы создания каркаса логических диалектов RIF, синтаксические конструкции и семантические структуры каркаса, правила интерпретации синтаксических конструкций в семантических структурах, пример синтаксиса и семантики диалекта как специализации каркаса. В работе [37] были созданы логические правила на языке RIF-FLD, используемые для аннотирования элементов онтологии. В работе [26] определены общие принципы конструирования разрешимых расширений канонической модели на основе классов логических зависимостей в данных с сохранением семантики языков запросов. В работах [51][61] разработано взаимное отображение канонической модели данных и языка OWL 2 QL, разработан и применен метод верификации отображения моделей на основе эквивалентного представления в единой семантической структуре. В работах [52][48][44][9][11] проведена унификация тензорной и графовой моделей данных в канонической объектной модели. В работах [66][20] разработан метод автоматизации верификации уточнения при композиционном проектировании информационных систем и предметных посредников. В работе [56] разработан метод верификации корректности интеграции веб-сервисов с использованием представления их семантики в языке AMN.

В работе [30] разработан метод основанной на запросах верификации корректности интеграции данных. В работе [33] разработаны основные положения метода повторного использования и композиционного проектирования потоков работ. В работах [41][43][60] разработан и применен метод верификации интеграции собственно данных путем определения семантики высокоуровневых программ интеграции данных в языке AMN. В работах [39][42] разработан метод спецификации мультимодельных правил интеграции данных с использованием логического языка на правилах и их реализации с возможностью последующей верификации. В работе [35] определены основные принципы реализации правил интеграции схем данных в модели RDF при помощи высокоуровневого языка анализа данных для исполнения в распределенной вычислительной среде. В работе [38][32] определены общие принципы расширяемого подхода к материализованной интеграции больших данных. В работах [47][12] разработаны основные принципы построения и реализации комбинированной виртуально-материализованной среды интеграции больших неоднородных коллекций данных. В работе [45] определены этапы процесса извлечения информации из разноструктурированных данных и ее приведения к целевой схеме, определены отдельные компоненты программной архитектуры средств извлечения информации, ее интеграции и анализа, и связи между ними, и правила трансформации данных. В работе [8] разработано преобразование коллекций данных графовых моделей и модели RDF в интегрированное представление. В работе [5] разработана архитектура системы извлечения и интеграции информации из данных по Арктической зоне и правила трансформации данных. В работе [6] проведено описание задачи анализа системного риска, определены этапы процесса решения задач в виртуально-материализованной среде интеграции, определена концептуальная схема предметной области задачи и схемы источников данных, определено описание задачи в виде декларативной программы, определены семантические отображения схем источников в концептуальную схему. В работе [3] разработаны основные логические структуры данных для решения задач в области управления землепользованием. В работах [50][13] разработана программная архитектура мультидиалектной среды для решения задач над неоднородными источниками данных, основные принципы ее реализации, концептуальные спецификации примера применения. В работах [46][7] определены основные принципы представления потоков работ в мультидиалектной среде и программная архитектура расширения мультидиалектной среды, ориентированного на потоки работ. В работе [16] разработан пример спецификации задачи в мультидиалектной архитектуре предметных посредников. В работе [56] описан метод мультидиалектной спецификации потоков работ, онтология структуры потоков работ. В работе [49] определен состав деятельностей по семантическому контролю научных потоков работ.

Структура и объем диссертации

Диссертационная работа состоит из введения, пяти глав, заключения, списка литературы и четырех приложений, в том числе. Общий объём диссертации, включая 39 рисунков и 55 таблиц и приложения, составляет 433 страницы. Список литературы состоит из 382 наименований.

Основное содержание работы

Во **введении** обоснована актуальность темы диссертации, сформулирована цель исследования, указаны основные задачи и научные положения, выносимые на защиту, охарактеризованы научная новизна, теоретическая и практическая значимость, указаны используемые методы исследования.

В **первой главе** рассматриваются основные понятия интеграции неоднородных данных и методы интеграции, подлежащие верификации. Кратко охарактеризованы *основные виды моделей данных*, появившиеся с конца 1960-х годов до наших дней, начиная с сетевой и иерархической до векторной. Более подробно описаны те виды моделей данных, которым в работе уделено наибольшее внимание.

Онтологические модели данных предназначены для представления онтологий[14], моделирующих знания об индивидах, их атрибутах и связях между индивидами в различных предметных областях. Знания формализуются в виде сущностей (классов), обладающих атрибутами (свойствами), и связей между сущностями. Онтологии обычно рассматриваются как более высокий уровень абстракции, чем схемы баз данных. Для определения семантики онтологий обычно используются *дескриптивные логики*[15], в которых задачи определения непротиворечивости онтологии или поглощения понятий разрешимы. Основанные на онтологиях технологии (OBDA, OBDI) предполагают использование баз данных на основе концептуализации прикладных областей и доступа к реляционным базам данных, опосредованного концептуальным представлением данных. В *тензорных* моделях данных основной структурой является многомерный массив – коллекция элементов данных одного типа, где каждый элемент имеет координаты, располагающиеся в прямоугольной решетке с параллельными осями – подмножестве конечномерного Евклидова пространства[16]. *Графовые* модели данных оперируют графами как основными структурами данных[17]. В *атрибутированных* графах как вершины, так и ребра могут быть аннотированы метками ключ-значение. Аналитические запросы к графам могут включать сложные операции, подобные нахождению кратчайшего пути или клики. *Процессные* модели или модели *потоков работ*[18] предназначены для описания деятельности различных организаций при решении соответствующих им сложных задач. При помощи потоков работ обычно формализуются задачи интеграции данных и сервисов. Использование и развитие *языков на правилах* для логического программирования, дедуктивных баз данных, представления знаний, создания интеллектуальных информационных систем продолжается уже более сорока лет. Для эффективного использования потенциала подходов, основанных на правилах, разработан стандарт RIF (Rule Interchange Format)[19] для унифицированного, интероперабельного использования спецификаций, представленных на различных языках на правилах. Идея RIF состоит в создании расширяемого унифицированного языка на правилах, обеспечивающего возможность построения сохраняющих семантику отображений в него различных языков на правилах. Такой унифицированный язык представляется как семейство диалектов, которые имеют

---

[14] Gruber T. Ontology // Encyclopedia of Database Systems. Springer, 2018. P. 2574–2576. doi.org/10.1007/978-1-4614-8265-9_1318

[15] Borgida A. Description Logics // Encyclopedia of Database Systems. – Springer, 2018. P. 1058–1063. doi.org/10.1007/978-1-4614-8265-9_1310

[16] Baumann P. Array Databases // Encyclopedia of Database Systems. – Springer, 2018. P. 165–177. doi.org/10.1007/978-1-4614-8265-9_2061

[17] Wood P.T. Graph Database // Encyclopedia of Database Systems. – Springer, 2018. P. 1639–1643. doi.org/10.1007/978-1-4614-8265-9_183

[18] Palmer N. Workflow Model // Encyclopedia of Database Systems. – Springer, 2018. P. 4716. doi.org/10.1007/978-1-4614-8265-9_818

[19] Kifer M., Boley H. RIF Overview (second edition). W3C Recommendation. — W3C, 2013. https://www.w3.org/TR/rif-overview/

общее ядро (корневой диалект) и совокупность расширяющих его диалектов, образующих ориентированный граф без циклов.

Описана *роль канонической модели данных в процессе интеграции данных*: в инфраструктуре данных она служит языком представления знаний, упоминаемым в принципе FAIR I1 (*(мета) данные используют формальный, доступный, разделяемый и широко применимый язык для представления знаний*). Каждая *исходная* модель данных, используемая для представления некоторого набора источников данных в инфраструктуре, должна быть отображена в каноническую модель. При необходимости отображение может сопровождаться *расширением* ядра канонической модели, для того, чтобы покрыть все особенности исходных моделей данных. Примерами расширений могут служить структуры (типы) данных, ограничения (зависимости), сложные операции над данными. Отображение должно быть формализовано и верифицировано: должно быть предоставлено формальное доказательство того, что отображение сохраняет семантику структур данных и операций манипулирования данными исходной модели данных. Каноническая модель для инфраструктуры данных конструируется как объединение ядра и всех расширений. В рамках работы в качестве *ядра канонической модели данных* в основном рассматривается язык СИНТЕЗ[20], разработанный основателем направления сохраняющих семантику отображений моделей данных – Л.А. Калиниченко. Язык СИНТЕЗ представляет собой комбинированную слабоструктурированную и объектную модель данных. Слабоструктурированные (самоописываемые) данные представлены *фреймами*. Фреймы используются для описания любого объекта, включая объекты самого языка, такие как спецификации типов, классов, функций, утверждений. Типизированные данные должны соответствовать *абстрактным типам данных* (АТД), описывающих состояние и поведение своих экземпляров в терминах *атрибутов* и методов типа. В определения типов могут быть включены *инварианты* типов (ограничения, выраженные в виде замкнутой логической формулы). Помимо АТД, язык содержит также обширную коллекцию встроенных типов данных. Методы и инварианты АТД описываются при помощи встроенного типа *функции.* Спецификация типа функции включает описание параметров функции и предикативную спецификацию функции. Для задания предикативных спецификаций функций в канонической модели используется язык логических формул многосортного объектного исчисления – типизированный вариант логики предикатов первого порядка. Предикативная спецификация позволяет задавать смешанные пред и постусловия, определяющие действия, реализуемые функцией. Однородные совокупности объектов предметной области представляются в канонической модели *классами*. Для описания длительных составных деятельностей используются классы *скриптов,* основанных на сетях Петри. Для расширения языка СИНТЕЗ в качестве канонической модели используются метаклассы, метафреймы, параметризованные конструкции, включая утверждения и родовые типы данных.

Кратко описаны *методы интеграции схем данных*, *подлежащие верификации – регистрация источников* в системе виртуальной интеграции (предметном посреднике), включающая поиск для элементов схемы посредника релевантных им элементов схем источников данных, конструирование представлений (views), связывающих элементы схем источников и схемы посредника. Задачей верификации интеграции схем при этом является формальное доказательство того, что виртуальная схема посредника корректным образом реализуется посредством схем источников.

Кратко описаны *методы интеграции данных*, *подлежащие верификации*: трансформация данных (их структур и форматов); разрешение сущностей

[20] Kalinichenko L. A., Stupnikov S. A., Martynov D. O. SYNTHESIS: a Language for Canonical Information Modeling and Mediator Definition for Problem Solving in Heterogeneous Information Resource Environments. — M.: IPI RAS, 2007.

(группирование, сопоставление, кластеризация сущностей из разных источников)[21]; слияние сущностей (образование интегрированного представления о сущности реального мира из разных описаний, полученных из разных источников, с разрешением конфликтов для каждого из атрибутов)[22].

Приведено краткое описание *языков и инструментов конструирования трансформаций моделей данных*, применяемых при программной реализации метода унификации моделей данных (раздел 2.9). Декларативные языки метакомпиляции SDF (Syntax Definition Formalism) и ASF (Algebraic Specification Formalism), поддержаны инструментарием Meta-Environment[23], основанным на переписывании термов. Декларативно-императивный язык ATL (ATLAS Transformation Language)[24] развивается в рамках движимой моделями инженерии MDE (Model-driven Engineering)[25], поддерживаемой Object Management Group. Для языка ATL на базе платформы Eclipse реализована интегрированная среда разработки, называемая ATL Development Tools (ADT).

Приведен краткий обзор методов и средств уточнения спецификаций. Существующие методы формализации уточнения различаются языками спецификации систем, технологиями доказательства уточнения и инструментальными средствами доказательства уточнения. Рассмотрены практическая теория программирования Hehner, программирование из спецификаций Morgan, исчисление уточнения Back, формальные языки спецификаций Z, VDM, RSL. Основное внимание уделено языку AMN (Abstract Machine Notation)[26], который используется в работе в качестве языка формализации уточнения. Язык AMN основан на логике первого порядка и теории множеств Цермело-Френкеля. AMN выбран как язык, поддержанный индустриальной технологией и средствами доказательства Atelier B[27]. Накоплен более чем двадцатилетний мировой опыт применения языка AMN и средств его инструментальной поддержки при разработке промышленных программных систем. Показана связь используемой в работе терминологии с государственными стандартами РФ.

Во **второй главе** рассмотрен метод унификации моделей данных.

Инфраструктура данных объединяет источники данных, которые представлены с использованием различных моделей данных в модельно неоднородную среду. Для обеспечения возможности совместного использования данных из различных источников и их интеграции эти модели данных и соответствующие им языки манипулирования данными должны быть *унифицированы* в рамках некоторой канонической модели данных *C*. Основным принципом конструирования (синтеза) канонической модели для инфраструктуры данных является *расширяемость ядра* канонической модели в неоднородной среде, чтобы охватить особенности моделей исходных данных[28]. Ядро $Kernel_C$ канонической модели фиксируется. Конкретная исходная модель данных *R* среды *унифицирована*, если создано такое расширение *E* ядра канонической модели (это расширение может быть пустым) и такое отображение *M* исходной модели в расширенную каноническую модель, что исходная модель *уточняет* расширенную

---

[21] Christophides V., Efthymiou V., Palpanas T., Papadakis G., Stefanidis K. An Overview of End-to-End Entity Resolution for Big Data // ACM Comput. Surv., 2021. Vol. 53. No. 6. Article 127. doi.org/10.1145/3418896

[22] Canalle G.K., Salgado A.C., Loscio B.F. A survey on data fusion: what for? In what form? What is next? // J Intell. Inf. Syst., 2021. Vol. 57. P. 25–50. doi.org/10.1007/s10844-020-00627-4

[23] The ASF+SDF Meta-Environment. — CWI, 2012. https://github.com/cwi-swat/meta-environment

[24] ATL – a model transformation technology. — Eclipse, 2025. https://eclipse.org/atl/

[25] da Silva R.A. Model-driven engineering: A survey supported by the unified conceptual model // Computer Languages, Systems & Structures, 2015. Vol. 43. P. 139–155.

[26] Abrial J. R. The B-book: assigning programs to meanings. — New York: Cambridge University Press, 1996.

[27] Atelier B: The Industrial Tool to Efficiently Deploy the B Method. — ClearSy, 2026. https://www.atelierb.eu/

[28] Kalinichenko L. A. Method for data models integration in the common paradigm // Proceedings of the First East-European Symposium on Advances in Databases and Information Systems. — Nevsky Dialect, 1997. Vol. 1. P. 275–284.

каноническую модель. Уточнение модели *C* моделью *R* означает, что для любой допустимой спецификации (схемы) *S*, определенной с использованием модели *R*, ее образ *M(S)* в модели *C уточняется* спецификацией *S* (рис. 1).

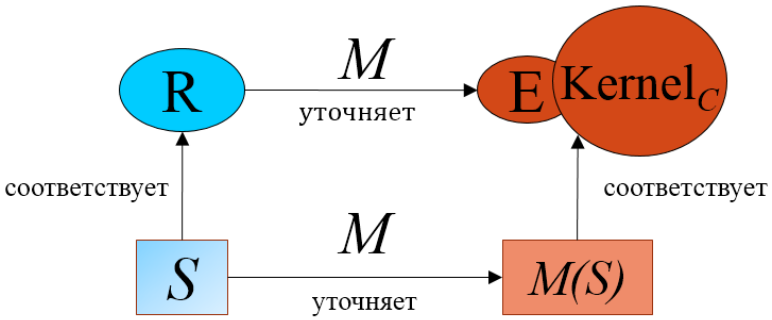

Рис. 1. Унификация исходной модели данных

Такое уточняющее отображение моделей означает сохранение информации и операций исходной модели (т.е. ее семантику) после преобразования ее схем в каноническую модель. В результате унификации также должна быть получена возможность формально доказывать уточнение произвольной конкретной схемой *S* модели *R* ее образа *M(S)*. Каноническая модель для инфраструктуры данных синтезируется как объединение расширений, построенных для всех моделей данных используемых источников.

Для обеспечения деятельности по унификации моделей данных необходимы следующие основные языки и формализмы: ядро канонической информационной модели; формализм, позволяющий описывать синтаксис моделей данных; формализм, позволяющий описывать трансформации из одной модели данных в другую; формализм, поддерживающий доказательство уточнения. В качестве ядра канонической информационной модели в настоящей работе в большинстве случаев рассматривается язык СИНТЕЗ, ориентированный на семантическую интероперабельность и композиционное проектирование информационных систем в широком диапазоне существующих неоднородных информационных компонентов. Однако, метод унификации моделей может быть применен также в случаях когда в качестве ядра выбрана какая-либо другая модель данных, например, SQL или RDF. В качестве примеров формализмов описания синтаксиса в данной работе рассматриваются языки SDF и CS (Concrete Syntax Specification Language). В качестве примеров формализмов описания трансформаций моделей – языки ASF и ATL. Для формализации семантики моделей данных и верификации уточнения используется язык AMN.

Деятельность по унификации исходной модели *R* в канонической модели *C*, осуществляемая экспертом при поддержке инструментальных средств унификации моделей (Унификатора моделей), разбивается на следующие этапы:

- формализация синтаксиса моделей *R* и *C*;
- интеграция абстрактных синтаксисов моделей *R* и *C*;
- создание необходимого расширения *E* канонической модели *C;*
- построение трансформации модели *R* в расширенную каноническую модель;
- формализация семантики моделей *R* и *C*;
- верификация уточнения моделью *R* расширенной канонической модели.

Формализация синтаксиса и семантики канонической модели осуществляется один раз и используется затем при унификации всех исходных моделей данных неоднородной среды.

Основные идеи этапов унификации состоят в следующем:

1. Формализация синтаксиса моделей данных. Входными данными этого этапа является синтаксис моделей, изложенный в некоторых нормативных документах

(описаниях соответствующих языков). Синтаксис моделей чаще всего представляется в каком-либо варианте формы Бэкуса–Наура. По нормативному описанию синтаксиса создается *абстрактный синтаксис* модели, представляющий собой модель уровня MOF M2, соответствующую некоторой выбранной метамодели уровня MOF M3, например, Ecore, используемая в проекте Eclipse Modeling Project[29]. Также с использованием выбранного языка определения конкретного синтаксиса (например, SDF или CS) создается *грамматика конкретного синтаксиса* модели (соответствующая абстрактному синтаксису), и это новое определение считается формальным описанием конкретного синтаксиса. На основании грамматики конкретного синтаксиса инструментарий *текстового определения синтаксиса* автоматически генерирует *синтаксический анализатор* (парсер) схем модели данных. Синтаксический анализатор может принимать на вход схемы модели данных, проверять их соответствие синтаксису и автоматически генерировать по схемам модели уровня MOF M1, соответствующие модели абстрактного синтаксиса.

2. Интеграция абстрактных синтаксисов моделей. Абстрактный синтаксис модели данных – это описание, содержащее значимые связи между понятиями, соответствующими конструкциям (элементам) модели. Понятия и связи, составляющие абстрактный синтаксис, могут быть аннотированы вербальными определениями. Целью интеграции абстрактных синтаксисов является выделение релевантных конструкций исходной и канонической моделей. Входными данными этапа являются синтаксисы моделей и их вербальная семантика, описанная в нормативных документах. Соответствия между релевантными элементами исходной и канонической моделей могут быть установлены экспертом вручную с использованием, например, инструментов, подобных простому редактору *Ecore2Ecore*, входящему в проект Eclipse Modeling Framework. Классический подход к автоматизированному установлению соответствий – автоматическое порождение векторов дескрипторов элементов синтаксиса на основе вербальных определений и поиск элементов синтаксиса канонической модели, близких к элементам синтаксиса исходной модели, с использованием вычисления меры близости векторов. Также для установления соответствий могут быть применены различные современные инструменты сопоставления схем, вплоть до использующих большие языковые модели[30]. После применения автоматизированных инструментов эксперт подтверждает либо отвергает найденные соответствия, а также добавляет соответствия, не найденные автоматически. Результатом интеграции является список соответствий элементов исходной и канонической моделей. Одному элементу канонической модели может соответствовать несколько конструкций исходной модели (и наоборот).

3. Создание необходимого расширения канонической модели. К необходимости создания расширения эксперт приходит в процессе интеграции эталонных схем исходной и канонической моделей: обнаруживается, что средств ядра (или ядра с уже построенными расширениями) канонической модели недостаточно для выражения некоторых конструкций исходной модели. К этому выводу эксперт приходит, если: (1) между конструкциями исходной модели и конструкциями канонической модели алгоритмы интеграции не установили связей; (2) на основе анализа документов, описывающих синтаксис и семантику исходной и канонической модели сделано экспертное заключение о невыразимости конструкций исходной модели в канонической.

При создании расширения для канонической модели предполагаются определенными документы, описывающие ее синтаксис и семантику; формальный синтаксис; AMN-семантика (т.е. вербальные правила и инструментальные средства

[29] Steinberg D., Budinsky F., Paternostro M., Merks E. EMF: Eclipse Modeling Framework, 2nd Edition. — Addison-Wesley Professional, 2008.

[30] Chugh T., Zambre D. ASTRA: Automatic Schema Matching using Machine Translation // Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track. — ACL, 2024. P. 1237-1244.

отображения в язык AMN). Для исходной модели необходимо обеспечить нормативные документы, описывающие синтаксис и семантику; формальный синтаксис; AMN-семантику; установленные автоматически или с помощью эксперта и подтвержденные экспертом соответствия элементов исходной и канонической моделей; неформальное определение отображения тех конструкций исходной модели в конструкции канонической, для которых установлено соответствие.

Создание расширения включает следующие шаги: (3.1) эксперт определяет название расширения и расширяемую версию канонической модели; (3.2) эксперт итеративно редактирует синтаксис расширения и вербальной семантики (включая правила отображения AMN). При этом эксперт обращается к описаниям исходной и канонической моделей, к интерфейсу интеграции абстрактных синтаксисов моделей. Одновременно эксперт осуществляет дополнение вербального описания отображения исходной модели в каноническую для недостающих конструкций; (3.3) уточняется картина интеграции абстрактных синтаксисов исходной модели и расширенной канонической модели: устанавливаются и фиксируются соответствия между конструкциями исходной модели и конструкциями расширения; (3.4) относящаяся к расширению метаинформация фиксируется в *реестре моделей* (специализированной базе данных).

4. Построение трансформации исходной модели в расширенную каноническую модель. Входными данными этапа являются список соответствий конструкций исходной и канонической моделей, синтаксис и вербальная семантика исходной и канонической моделей. Эксперт разрабатывает неформальные правила отображения исходной модели в расширенную каноническую. Автоматически генерируется заготовка трансформации исходной модели в расширенную каноническую. Заготовка строится на основе информации, содержащейся в списке соответствий элементов исходной и канонической моделей и синтаксисах моделей. Эксперт формализует вербальные правила отображения, превращая автоматически порожденную заготовку трансформации в полноценную трансформацию исходной модели в каноническую.

5. Формализация семантики моделей. Входными данными этого этапа являются описания семантики моделей, изложенные в описаниях соответствующих языков. Семантика модели может быть как вербальной, представленной в виде комментария к синтаксису, так и частично формализованной (например, с использованием логики и теории множеств). Семантика модели формализуется экспертом в виде определяемой на выбранном языке трансформаций (например, ASF или ATL) трансформации схем модели в язык AMN. При построении семантических трансформаций моделей используется их формальный синтаксис. Такая формализация семантики необходима для последующей верификации уточнения моделей.

6. Верификация уточнения исходной моделью расширенной канонической модели. Верификация уточнения осуществляется на наборе образцов спецификаций исходной модели. Желательно, чтобы образец являлся достаточно характерной спецификацией исходной модели (типовой конструкцией). Примерами могут служить образцы потоков работ (workflow patterns), использованные для синтеза канонической процессной модели. В случае, когда для модели затруднительно выделить образцы, верификация может быть проведена для используемых при интеграции данных конкретных схем источников, представленных в исходной модели. Идея верификации уточнения образцом *S* исходной модели его образа *CS* при отображении в каноническую модель состоит в следующем: образец *S* и его образ *CS* отображаются в язык AMN, и уточнение доказывается средствами B-технологии для соответствующих AMN-спецификаций. Уточнение AMN-спецификаций будет свидетельствовать об уточнении образцом *S* его образа *CS*. Этап включает следующие шаги: (6.1) экспертом выделяются образцы исходной модели, строятся их спецификации S_i и проверяются на соответствие формальному синтаксису; (6.2) с использованием трансформации исходной модели в

AMN (построенной на этапе 5) автоматически порождаются AMN-спецификации S_i^{AMN} для образцов; (6.3) с использованием трансформации исходной модели в каноническую (построенной на этапе 4) автоматически порождаются канонические спецификации CS_i образцов; (6.4) с использованием трансформации канонической модели в AMN (построенной на этапе 5), автоматически порождаются канонические AMN-спецификации CS_i^{AMN} для образцов; (6.5) Экспертом с использованием средств B-технологии автоматически и/или интерактивно доказывается уточнение спецификациями S_i^{AMN} спецификаций CS_i^{AMN}. Текст доказательств фиксируется.

Заметим, что метод верификации уточнения моделей на наборе образцов спецификаций исходной модели не является единственно возможным. Семантика модели данных R может быть представлена в виде единой AMN-спецификации R_{AMN}. При этом структуры данных модели – например, классы, массивы и т.д. – представляются переменными спецификации, различные свойства структур данных представляются инвариантами, характерные операции моделей данных представляются операциями AMN. Уточнение моделей данных при этом сводится к уточнению их семантических AMN-спецификаций.

Также в случаях, когда одна из моделей данных имеет строго определенную нормативную семантику в некоторой семантической структуре, а другая модель (ее подмножество) допускает определение своей семантики в той же семантической структуре, возможно доказательство корректности *взаимного отображения* моделей данных. Взаимным называется такое отображение моделей данных, которое определено как множество взаимооднозначных соответствий между элементами моделей. При этом доказательство эквивалентности семантики элементов моделей, связанных отображением, проводится индукцией по синтаксису моделей.

Каждый из методов верификации имеет свои преимущества и недостатки. Так, верификация на наборе образцов позволяет разделить доказательства уточнения моделей на непротиворечивые фрагменты, соответствующие образцам. Однако, покрытие всего множества возможных схем исходной модели образцами подобно покрытию программного кода тестами – невозможно строго доказать его полноту заранее. При этом для каждой конкретной схемы источника данных доказательство уточнения возможно.

Верификация на основе единых спецификаций моделей более сложна, поскольку предполагает, что все структуры модели и характерные операции выражены в одной спецификации, а не разделены на множество образцов. Однако, при этом доказательство обладает естественным единством и комплексностью.

Доказательство корректности взаимного отображения моделей данных с математической точки зрения даже сильнее, чем уточнение моделей (фактически, это взаимное уточнение моделей). Однако, такое отображение не всегда возможно. Кроме того, для такого рода верификации нужна семантика обеих моделей в единой семантической структуре, что тоже не всегда возможно и удобно. Также, для доказательства корректности уже неприменима B-технология, доказательства проводятся вручную. Вопросы применения для доказательства других автоматизированных систем, подобных Isabelle/HOL или Lean, выходят за рамки данной работы.

Приведены детальные описания всех этапов метода унификации, сопровожденные примерами. Формализация синтаксиса моделей данных, интеграция абстрактных синтаксисов проиллюстрирована на примерах подмножеств канонической модели данных (языка СИНТЕЗ) и онтологической модели данных OWL. Создание двух видов расширений проиллюстрировано в виде специализированных метаклассов ассоциаций для унификации языка OWL и в виде специализированных классов для унификации графовой модели. Конструирование трансформации исходной модели в расширенную каноническую модель проиллюстрировано на примере трансформации OWL в СИНТЕЗ.

Рассмотрен *метод автоматизированного построения трансформаций моделей данных* на основе установленных соответствий между элементами моделей с использованием средств движимой моделями инженерии на языке ATL. Результатом применения метода являются декларативно-императивные трансформации, реализующие отображения произвольных исходных моделей уровня M1, соответствующих исходной метамодели уровня M2, в целевые модели уровня M1, конформные целевой метамодели уровня M2. Основным содержанием метода является набор *порождающих правил*, обеспечивающих автоматическое построение ATL-модуля трансформации заданной исходной модели *S* в заданную целевую модель *T,* включающих*:*

– порождающее правило для построения заголовка трансформации;

– порождающие правила для построения сигнатуры сопоставляющих правил ATL, составляющих трансформацию (сигнатура включает имя, исходный и целевые элементы правила);

– порождающие правила для построения инициализации целевых элементов в правилах трансформации.

Рассмотрен метод *формализации семантики канонической объектной модели данных в языке AMN*. Производится определение отображения абстрактного синтаксиса подмножества ядра канонической модели (языка СИНТЕЗ) в спецификации AMN. Отображение задается при помощи семантических функций, определенных на элементах синтаксиса канонической модели, а также при помощи алгоритмов. Определение семантики структуры спецификаций типов осуществляется при помощи средств композиции абстрактных машин AMN. Изложена процедура преобразования ориентированного графа структуры модуля канонической модели, представляющего композиционную структуру спецификаций типов модуля, в ориентированный граф, представляющий композиционную структуру набора абстрактных машин. Определены семантические функции для атрибутов, методов, инвариантов АТД, классов, формул – спецификаций инвариантов и методов. Описана реализация построенного семантического отображения канонической модели данных в язык AMN в виде трансформации на языке ATL.

Рассмотрен *метод верификации корректности отображения моделей на основе уточнения образцов моделей* данных. Метод проиллюстрирован на примере отображения модели OWL в язык СИНТЕЗ. Изложены и проиллюстрированы основные идеи формализации семантики языка OWL как исходной модели в языке AMN. На примере образца онтологии проиллюстрировано его отображение в каноническую модель и уточнение семантической спецификацией образца канонической семантической спецификации образца.

Рассмотрен *метод верификации корректности отображения моделей на основе уточнения AMN-спецификаций моделей.* Идея метода заключается в следующем. Рассмотрим исходную модель *S* и целевую модель *T*. Построим отображение θ модели *S* в модель *T.* Выразим семантику моделей в виде спецификаций AMN, построив при этом машины M_S и M_T соответственно. При этом структуры данных моделей – например, классы, массивы и т.д. – представляются переменными машин, различные свойства структур данных представляются инвариантами машин, характерные операции моделей данных представляются операциями машин. Рассматриваемые операции исходной и целевой модели должны быть связаны отображением ЯМД. Отображение ЯОД представляется в виде специального *склеивающего инварианта* – замкнутой формулы, связывающей состояния машин M_S и M_T. Будем считать отображение θ *сохраняющим информацию и семантику операций*, если машина M_S, соответствующая исходной модели, уточняет машину M_T, соответствующую целевой модели. Метод проиллюстрирован на примере доказательства сохранения информации и семантики операций при отображении графовой модели данных в каноническую.

Рассмотрен *метод верификации корректности взаимного отображения моделей данных* на основе эквивалентного представления в единой семантической структуре. Метод применим в том случае, когда одна из моделей имеет строго определенную нормативную семантику в некоторой семантической структуре, а другая модель допускает определение своей семантики в той же семантической структуре. *Взаимным* называется такое отображение моделей данных, которое определено как множество взаимооднозначных соответствий между элементами моделей. Метод состоит из следующих этапов:

– определение отображения моделей данных (подмножеств моделей данных) как множества взаимооднозначных соответствий между элементами моделей;

– выбор нормативного документа с описанием семантики одной из моделей в некоторой семантической структуре;

– определение семантики второй модели в той же семантической структуре;

– доказательство эквивалентности семантики элементов моделей, связанных отображением.

В случае, если доказательство определено для всех элементов моделей данных (подмножеств моделей данных), связанных отображением, взаимное отображение моделей считается корректным. Применение метода рассматривается ниже на примере языка определения данных - подмножества модели OWL 2 QL, а также на примере языка манипулирования данными – подмножества языка на правилах RIF-BLD.

Рассмотрены два варианта *схемы функциональной структуры и реализации инструментальных средств унификации моделей данных*:

– реализация с использованием декларативного языка трансформации ASF и средств переписывания термов;

– реализация с использованием декларативно-императивного языка трансформации моделей ATL (на основе технологии движимой моделями инженерии MDE).

Перечислены программные компоненты функциональной структуры, описаны их назначение (связь с этапами метода унификации моделей), взаимосвязи и особенности реализации. Проведено сравнение реализаций для всех этапов метода унификации моделей данных, выделены преимущества подхода MDE.

Проведен обзор родственных работ в направлениях методов и инструментов для преобразования схем, конструирования трансформаций моделей, верификации трансформаций моделей. Выделены главные отличительные черты разработанного метода унификации моделей данных:

– метод сочетает разработку унифицирующих трансформаций моделей и верификацию уточнения моделей данных, чтобы устранить модельную неоднородность современных СУБД, обеспечить формальную основу для отображения схем и виртуальной или материализованной интеграции в инфраструктурах неоднородных данных;

– для трансформаций моделей данных обеспечивается формальное доказательство сохранения информации и семантики операций ЯМД;

– для обеспечения унификации разнообразных моделей данных в неоднородных инфраструктурах метод предлагает механизм расширения и синтеза унифицирующей канонической модели данных;

– в качестве ядра канонической модели данных применена объектная модель с механизмами расширения в виде специализированных типов данных, ограничений (зависимостей, аксиом), сложных операций над структурами данных;

– метод применен для унификации широкого спектра моделей данных: сетевой модели данных, процессной модели, онтологической модели, графовой модели, тензорной модели, модели на логических правилах.

В **третьей главе** рассмотрены два варианта метода верификации интеграции схем данных.

Первый вариант метода, основанный на уточнении типов данных, имеет своей целью формальное доказательство того, что некоторый предметный посредник, использующий в качестве ядра унифицирующей канонической модели данных язык СИНТЕЗ, осуществляет корректную интеграцию данных из некоторого набора источников данных. Корректность доказывается путем отображения схем посредника и схем источников данных в формальный язык AMN, и дальнейшим применением автоматизированных средств доказательства уточнения для языка AMN.

В *предметном посреднике* задана *схема предметной области* (*глобальная схема*), представленная в канонической модели данных. В посреднике интегрированы *N источников данных.* Каждый из источников, в общем случае, использует собственную модель данных: в этой модели представляются *схема* источника и *запросы* к нему. Предполагается, что унификация моделей данных источников в канонической модели данных уже проведена в соответствии с методом, рассмотренным во второй главе. При этом построены *трансформации*, преобразующие схемы, представленные в моделях данных источников $M_1, \ldots M_N$ в схемы канонической модели. Трансформации применены для преобразования схем источников в каноническую модель данных, получены *канонические схемы*. Источник данных связывается с посредником при помощи *адаптера*. Для каждой модели данных источника конструируется собственный адаптер. Для каждого конкретного источника настраивается экземпляр адаптера, соответствующего модели данных источника: в экземпляре адаптера сохраняется IP-адрес источника и его схема. Установлены соответствия между АТД глобальной схемы и релевантными им АТД схем источников. Для каждого источника в посреднике определен набор *представлений* (взглядов), связывающий сущности схемы источника и сущности глобальной схемы.

Приведен *основной алгоритм метода верификации интеграции схем*, принимающий на вход глобальную схему S_{MED}, схемы источников $Sources = \{S_1, \ldots, S_N\}$ в канонической модели, множество соответствий типов глобальной схемы и релевантных им АТД из схем источников *Sims*, множество взглядов $V = \{V_1, \ldots, V_L\}$. Каждый элемент V имеет вид Global and Local asView (GLAV)[31]:

$$G_1(v_1/T_{v1}), \ldots, G_p(v_p/T_{vp}), B_G \supseteq H_1(w_1/T_{w1}), \ldots, H_q(w_q/T_{wq}), B_H,$$

где $G_1, \ldots, G_p$ – классы глобальной схемы, $T_{v1}, \ldots, T_{vp}$ – типы их экземпляров; $H_1, \ldots, H_q$ – классы схем источников, $T_{w1}, \ldots, T_{wq}$ – типы их экземпляров; B_G, B_H – конъюнкции предикатов-условий на значения атрибутов экземпляров классов, обозначенных переменными $v_1, \ldots, v_p$ и $w_1, \ldots, w_q$ соответственно; либо вид функционального взгляда:

$$T_M.g(u_{1, \ldots,} u_r) \supseteq T_S.h(u_{1, \ldots,} u_r),$$

где T_M – АТД глобальной схемы, g – один из его методов, T_S – АТД схемы некоторого источника, h – один из его методов.

Основные шаги алгоритма состоят в следующем. К глобальной схеме и схемам источников применяется семантическая трансформация схем канонической модели в AMN. В результате получается контекстная машина и набор машин, соответствующих АТД схем. На основании соответствий АТД устанавливается соотношение между их экстенсионалами, выражается в виде конъюнкции предикатов равенства и добавляется в раздел PROPERTIES контекстной машины.

---

[31] Katsis Y., Papakonstantinou Y. View-Based Data Integration // Encyclopedia of Database Systems. – Springer, 2018. P. 4452–4461. doi.org/10.1007/978-1-4614-8265-9_1072

Далее для каждого взгляда, связывающего классы глобальной схемы и классы схем источников, производятся следующие действия. Все конструкции AMN, соответствующие используемым во взгляде АТД глобальной схемы, объединяются в одну уточняемую конструкцию *M.* Все конструкции AMN, соответствующие используемым во взгляде АТД источников, объединяются в одну уточняющую конструкцию M_{REF}. Фиксируется направление уточнения. Если для какого-то метода *g* глобальной схемы отсутствует взгляд, связывающий его с методом некоторого АТД источника (т.е. интегрируемые источники не реализуют этот метод), то операция *g* удаляется из уточняемой машины *M*, т.к. ее нечем будет уточнять. Если для какого-то метода *h* схемы источника отсутствует взгляд, связывающий его с методом некоторого АТД глобальной схемы, т.е. *h* не реализует напрямую какой-то метод глобальной схемы (но, возможно, участвует в реализации опосредованно), то в машину *M* добавляется одноименная операция с пустым телом для формальной корректности уточнения. Для каждого метода *g* глобальной схемы, связанного взглядом с методом *h* некоторого источника, операция *h* в машине M_{REF} переименовывается в *g* (для формальной корректности уточнения). Аналогично в соответствии с параметрами *g* переименовываются параметры *h*.

Определяется склеивающий инвариант уточнения и добавляется в раздел INVARIANT машины $M_{REF.}$ Машины *M*, M_{REF} загружаются в проект Atelier B, производится автоматизированное доказательство уточнения. В случае, если для всех взглядов доказательство уточнения завершено успешно, то корректность интеграции схем считается доказанной. В противном случае считается, что корректность интеграции схем доказать не удалось.

Детали метода иллюстрируются на примере применения интеграции схем данных в предметной области *проектирования землепользования*. Рассматривается интеграция одного источника сервиса данных о подборе сортов сельскохозяйственных культур. Иллюстрируются следующие детали метода: глобальная схема; схема источника; соответствие АТД глобальной схемы и релевантных им АТД из схемы источника; взгляды, связывающих классы и типы глобальной схемы с классами и типами источника; результаты применения семантической трансформации схем канонической модели в AMN к глобальной схеме и схеме источника; отношение экстенсионалов АДТ глобальной схемы и АТД схемы источника; необходимость объединения конструкций AMN, соответствующих нескольким АТД, в одну конструкцию; добавление операций с пустым телом в уточняемую конструкцию; переименование операций и параметров уточняющей конструкции; определение инварианта уточнения; статистика автоматизированного доказательства уточнения.

Второй вариант метода, основанный на уточнении запросов, имеет своей целью доказательство того, что некоторая *система интеграции данных,* использующая в качестве унифицирующей модели данных язык RDF и его расширения RDFS и OWL, осуществляет корректную интеграцию данных из некоторого набора источников данных. Корректность проверяется на наборе запросов пользователя к системе интеграции, выражаемых на языке SPARQL (именно поэтому верификация в данном разделе называется *основанной на запросах*). Корректность доказывается путем отображения схем предметной области, схем источников данных и запросов в формальный язык AMN, и дальнейшим применением автоматизированных средств доказательства уточнения для языка AMN.

Общая схема метода состоит в следующем. В *системе интеграции данных* задана *схема предметной области* на языках RDF, RDFS, OWL. В системе интегрированы *N источников данных.* Каждый из источников, в общем случае, использует собственную модель данных: в этой модели представляются *схема* источника и *запросы* к нему. Источник связывается с системой при помощи *интерфейса прикладных программ* (ИПП). Для каждого источника определен набор *представлений* (взглядов),

связывающий сущности схемы источника и сущности схемы предметной области. На вход от пользователя система принимает некоторый набор запросов $q_1, \ldots, q_M$, выраженных на языке SPARQL. Запросы выражены в терминах схемы предметной области. Набор запросов $q_1, \ldots, q_M$ должен включать основные типичные запросы пользователей в данной предметной области: именно на этом наборе запросов будет осуществляться верификация интеграции источников данных в системе. Для верификации определяется ряд семантических отображений:

– семантическое отображение конструкций языков RDF, RDFS, OWL в AMN, сопоставляющее произвольной схеме и набору запросов на языке SPARQL, спецификацию вида REFINEMENT. Каждому запросу q_i соответствует отдельная операция спецификации $op(q_i)$. Кроме того, общие конструкции языков RDF, RDFS, OWL, не зависящие от конкретной схемы, представляются отдельной спецификацией *ContextRDF* вида MACHINE;

– семантические отображения моделей данных источников в AMN, сопоставляющее произвольной схеме и набору запросов в модели данных источника спецификацию вида REFINEMENT. Каждому запросу q'_j соответствует отдельная операция спецификации $op(q'_j)$. Указывается, что спецификация источника уточняет (REFINES) спецификацию предметной области;

– семантические отображения представлений, связывающих сущности схемы источника и сущности схемы предметной области, в формулы AMN.

Единожды определенное семантическое отображение RDF, RDFS, OWL в AMN затем используется для верификации интеграции произвольных поддерживаемых источников данных в произвольных системах интеграции, основанных на RDF и SPARQL. То же самое верно и для семантических отображений конкретных моделей данных источников (например, для реляционной модели).

Итак, с использованием семантических отображений схемы данных предметной области и источников, представления и наборы запросов отображаются в спецификации AMN. Далее, спецификации помещаются в проект Atelier B типа *software development* (разработка программного обеспечения). Для каждой спецификации производится проверка типов (*Type Check*), автоматическая генерация теорем корректности и уточнения спецификаций (*Generate POs*). Для доказательства теорем применяются автоматические средства *Automatic*. Для доказательства теорем, оставшихся недоказанными автоматически, применяются средства интерактивного доказательства *Interactive Proof*. В случае, если все теоремы доказаны, интеграция данных считается *корректной на основе запросов* $q_1, \ldots, q_M$.

Детали метода иллюстрируются на примере применения интеграции данных в системе *Ontop* (относящейся к классу основанных на онтологиях систем интеграции – Ontology Based Data Integration, OBDI)[32]. В качестве предметной области рассматривается *онкологическое медицинское обслуживание*. Схемы предметной области и источников включают структуры данных пациентов и их заболеваний. Рассматривается интеграция одного источника данных, представленного в реляционной модели. Семантические отображения в AMN иллюстрируются на примерах схем и запросов.

Отображение RDF, RDFS, OWL строится в соответствии с нормативными документами – рекомендациями OMG, описывающими их семантику. Приведено отображение общих конструкций, не зависящих от конкретных схем, отображение схемы предметной области (включающей классы, их иерархию и свойства), запросов SPARQL. Для семантического отображения конструкций реляционной модели данных в AMN приведены отображения таблиц и их атрибутов, запросов на языке SQL. Приведено отображение представлений (где исходная часть представления – это SQL-запрос, а

[32] Ontop – A Virtual Knowledge Graph System. – Free University of Bozen-Bolzano, 2026. https://ontop-vkg.org/

целевая – набор RDF-триплетов) в формулы AMN. Представление отображается в логическую импликацию, в посылке которой стоит конъюнкция предикатов, соответствующих исходной части представления, в заключении - конъюнкция предикатов под квантором существования, соответствующих целевой части.

Приведена статистика доказательства теорем верификации для примера предметной области.

Приведен обзор родственных работ, посвященных определению семантики моделей данных и языков запросов, обладающих сходными с канонической моделью чертами, а также определению семантики стека языков RDF. Выделены отличительные черты разработанных семантических отображений по сравнению с родственными работами.

В **четвертой главе** рассмотрен *метод верификации интеграции данных,* нацеленный на определение формальной семантики программ интеграции и верификации на уровне интеграции собственно данных. При этом предполагается, что необходимое определение семантики и верификация на уровне моделей данных и схем уже проведено. Предполагается также, что процесс интеграции данных представляется в виде потока работ, деятельности которого представляют собой отдельные операции трансформации структурированных (типизированных) данных, разрешения или слияния сущностей.

Идея метода, предлагаемого в данной главе, состоит в том, чтобы сообщить высокоуровневым программам интеграции данных семантику в языке спецификаций, поддержанном средствами формального автоматического и/или интерактивного доказательства: для языка интеграции данных строится его отображение в язык спецификаций. Свойства программ, подлежащие проверке (корректность программ интеграции данных), связывающие состояния совокупности источников данных и состояние целевой интегрирующей базы данных, представляются экспертом в виде выражений выбранного языка спецификаций. Например, корректность может состоять в том, что для всех объектов из источников в целевой базе данных в результате применения программы интеграции появляются их правильные образы (отсутствуют потерянные данные). Затем, с использованием формальных средств доказательства, спецификация, выражающая семантику конкретного потока работ интеграции данных, проверяется на соответствие необходимым свойствам.

Для иллюстрации метода в качестве языка интеграции данных выбран HIL[33] – язык, разработанный компанией IBM, поставляемый в составе решения IBM InfoSphere Master Data Management. HIL – это язык высокого уровня для описания сложных потоков обработки данных, включающих операции трансформации данных, разрешения сущностей и слияния данных. Данные при этом могут поступать из больших коллекций данных, представленных в различных схемах. В качестве языка формальных спецификаций выбран AMN. Вместо AMN возможно использование и других языков, предоставляющих аналогичные возможности, например, RAISE или Z.

Семантическое отображение определяется с использованием набора семантических функций. В соответствии с принципами денотационной семантики, семантические функции отображают элементы *синтаксических доменов* языка HIL в конструкции языка AMN. Синтаксические домены HIL структурированы при помощи метамодели *Ecore.*

Каждая программа на HIL представляется в AMN отдельной конструкцией вида REFINEMENT. Семантическая функция *sProgram* для синтаксического домена *Program* определяется следующим образом:

---

[33] Hernández M.A., Koutrika G., Krishnamurthy R. et al. HIL: a high-level scripting language for entity integration // EDBT Conference Proceedings. — ACM, 2013. P. 549–560.

```
sProgram[Statement*] ≙
REFINEMENT HILProgramSemantics
ABSTRACT_CONSTANTS sAbstractConstants[Statement*]
PROPERTIES sProperties[Statement*]
VARIABLES sVariables[Statement*]
INVARIANT sInvariant[Statement*]
INITIALISATION sInitialisation[Statement*]
OPERATIONS sOperations[Statement*]
END
```

Эта функция формирует семантическую спецификацию для всей программы HIL. Отдельные секции спецификации (например, секции переменных или операций) формируются соответствующими семантическими функциями (например, *sVariables, sOperations*).

Типы сущностей и типы индексов представляются в AMN переменными абстрактных машин в разделе VARIABLES, и типизируются в разделе INVARIANT. Для типа в разделе переменных объявляется одноименная переменная, которая типизируется в инварианте как подмножество (POW) множества всех структур (struct) – кортежей. Кортежи состоят из атрибутов, соответствующих атрибутам типа. Между встроенными типами HIL и AMN установлено взаимооднозначное соответствие, используемое при типизации атрибутов структур в AMN. Например, тип *string* языка HIL соответствует типу STRING_TYPE в AMN, тип Boolean – типу BOOL и т.д. *Тип индекса fmap* в HIL представляет собой отображение из множества экземпляров типа ключа в экземпляры типа значения. *Функции* HIL представляются в AMN одноименными *абстрактными константами*. Константы типизируются в секции PROPERTIES спецификации AMN.

Каждому *правилу* языка HIL ставится в соответствие операция абстрактной машины AMN. Например, для операции разрешения сущностей *create link*, включающей секцию *select* определения сущностей формируемой коллекции, секцию *from* извлечения данных из исходных коллекций, секцию *match* предварительного отсева кортежей, полученных в результате соединения коллекций, секцию *check* дополнительной проверки отобранных в *match* кортежей, семантическая функция выглядит следующим образом:

```
sOperations[ create link name as
  select select* from fromClause*
  match using match1, match2
  check check1 check2 ] ≙
sCreateLinkOperationName[name] =
SELECT sStateCheckPredicate[name]
THEN
  name := { rr | #(sExistentialVariables[fromClause*]).(
    sExistVarTyping[fromClause*] & rr = rec(sRecord[select*]) &
    (sMatch[match1] or sMatch[match2]) & sCheck[check1] & sCheck[check2]
  )};
  sStateSetSubstitution[name]
END
```

Основные идеи отображения операции разрешения сущностей состоят в следующем. Результат операции формируется как множество, образованное при помощи конструкции выделения множества *{rr | F(rr)}* (здесь *rr* – имя переменной, отвечающей элементу множества, *F(rr)* – формула, которая должны обращаться в истину на элементах множества). Переменная *rr* типизируется типом записи *rec*, обладающим необходимыми для типа коллекции атрибутами. Для каждой из соединяемых коллекций заводится по переменной, переменные связываются квантором всеобщности # и типизируются как принадлежащие соответствующим коллекциям. Условия из секций

match и *check* представляются соответствующими условиями на переменные. Аналогично определяется семантика операций включения данных в коллекции (*insert into*).

Для отслеживания фактов исполнения операций в конструкции *HILProgramSemantics* заводится переменная *state*, отвечающая состоянию потока работ. Переменная типизируется в инварианте типом структуры *struct*, атрибуты которого отражают завершение каждой из операций потока работ. Предусловием исполнения каждой из операций является завершение всех других необходимых операций. Также, завершающим действием каждой операции является установка флага завершения операции переменной *state* в значение TRUE. Начальная инициализация переменной state производится в разделе INITIALISATION, флаги завершения всех операций устанавливаются в значение FALSE. Определены семантические функции, формирующие предикат типизации, инициализацию переменной состояния и подстановку присваивания переменной состояния.

Разработанные семантические функции, реализованы в виде трансформации на языке ATL. Разработана декларативно-императивная семантическая трансформация, преобразующая программы языка HIL в спецификации языка AMN. Большая часть семантического отображения реализована с использованием декларативных правил. Некоторые семантические функции реализованы при помощи императивных конструкций.

Применение формальной семантики для языка разрешения сущностей и слияния данных для верификации потоков работ интеграции данных осуществляется следующим образом. Программа интеграции данных на языке HIL, подлежащая верификации, автоматически преобразуется в семантическую спецификацию на языке AMN с использованием разработанной ATL-трансформации. Затем формулы, отражающие свойства потока работ интеграции данных, подлежащие верификации, добавляются вручную в инвариант конструкции AMN, отражающей семантику потока работ интеграции данных. Затем семантические спецификации помещаются в программное средство Atelier B, где осуществляется проверка синтаксиса спецификаций и корректности типизации переменных и термов. Далее осуществляется автоматическая генерация теорем, отражающих сохранение инварианта при выполнении операций спецификации. Теоремы доказываются с использованием автоматических и интерактивных средств доказательства Atelier B.

Метод определения формальной семантики конструкций языка HIL, позволяет использовать язык HIL для реализации верифицируемых трансформаций интеграции данных. Однако, для спецификации интеграции данных могут быть удобны языки более высокого уровня, например, основанные на логических правилах. Разработан *метод спецификации правил интеграции данных с использованием рекомендации W3C - логического диалекта RIF-BLD*[34] *и их реализации на языке HIL*. Тем самым обеспечивается возможность верификации свойств спецификаций интеграции данных, выраженных на высокоуровневых логических правилах. Возможности RIF-BLD позволяют использовать в одном правиле сущности из разных коллекций, представленные в *разных моделях данных*. Рассматриваются правила, в голове которых могут присутствовать лишь предикаты, соответствующие сущностям целевой схемы, а в теле – предикаты, соответствующие сущностям исходных схем.

Основные принципы спецификации правил интеграции данных с использованием RIF-BLD состоят в следующем:

– правила интеграции имеют вид импликации, посылка которой называется *телом*, а заключение – *головой*;

[34] Boley H., Kifer M. RIF Basic Logic Dialect. W3C Recommendation. — W3C, 2013. https://www.w3.org/TR/rif-bld/.

– голова и тело правил представляют собой формулы, связывающие предикаты операциями конъюнкции или дизъюнкции;

– в теле правила допускаются предикаты, соответствующие сущностям только исходных схем, в голове – только целевой схеме;

– для описания свойств кортежей, принадлежащих отношениям реляционной модели, в правилах используются предикаты с именованными аргументами;

– для описания свойств структур данных произвольной степени вложенности (при помощи которых представляются данные различных нереляционных моделей – XML, документной, графовой и т.д.) используются фреймовые предикаты и предикаты членства;

– связь значений атрибутов сущностей исходных и целевой схем осуществляется при помощи переменных;

– свободные переменные, использующиеся в теле правила, связываются внешним квантором всеобщности;

– свободные переменные в голове правила, не встречающиеся в теле (фактически, они соответствуют данным, не определенным явно в исходных коллекциях), связываются квантором существования;

– нетривиальные предикаты-условия (отличные от равенства) и функции разрешения конфликтов определяются как внешние термы.

Основные принципы реализации правил интеграции данных на RIF-BLD в языке HIL состоят в следующем:

– сущности исходных и целевой схем, функции разрешения конфликтов и нетривиальные предикаты-условия объявляются при помощи директивы *declare*; для функций определяется их сигнатура;

– функции реализуются на языке Jaql в отдельных секциях программы, либо на языке Java во внешних файлах;

– для каждой сущности (отношения) в голове правила создается отдельный оператор *insert,* порождающий кортежи этих отношений:

1) в секции *from* объявляются исходные сущности, каждому предикату с именованными переменными или предикату членства в теле правила соответствует объявление с точностью до имени переменной;

2) в секции *select* указываются атрибуты целевых отношений (в соответствии с их названиями в предикатах головы правила) и выражения порождения их значений; выражения формируются в соответствии с термами в предикатах головы правила либо с термами во фреймовых предикатах тела правила;

3) в секции *where* указываются предикаты, отражающие способы соединения исходных сущностей (формируемые на основании совпадения имен переменных во фреймовых предикатах тела) и их отбора (соответствующие предикатам, налагающим условия на данные отдельных исходных коллекций в теле правила);

– если соединение сущностей исходных коллекций в теле правила осуществляется с использованием нетривиальных предикатов и функций (отличных от равенства атрибутов), предварительное соединение сущностей производится с использованием оператора разрешения сущностей *create link*:

1) в секции *from* указываются соединяемые коллекции;

2) в секции *select* указываются составные ключи, однозначно идентифицирующие исходные сущности;

3) в секции *match* указываются правила сопоставления сущностей;

4) коллекция пар сопоставленных сущностей затем используется в качестве входной в операторах *insert,* порождающих кортежи целевых отношений.

Метод иллюстрируется на данных из предметной области организации поисковых и спасательных операций в Арктической зоне, представленных в различных моделях: XML и реляционная модель, документная модель, графовая модель.

Приведен обзор родственных работ по определению семантики языков концептуального моделирования, моделей данных, императивных языков программирования, а также по верификации трансформаций моделей.

Особенность метода верификации интеграции данных, изложенного в данной главе, состоит в том, что формальная денотационная семантика в языке AMN сообщается высокоуровневому языку разрешения сущностей и интеграции данных. Эта семантика позволяет применять автоматизированные средства доказательства промышленного уровня для верификации корректности потоков работ интеграции данных.

В **пятой главе** описаны схемы функциональной структуры сред решения задач над неоднородными источниками данных, в которых применяются методы верифицируемой интеграции данных, описанные в предыдущих главах.

Функциональная структура комбинированной виртуально-материализованной среды интеграции неоднородных коллекций данных различного вида (структурированных, слабоструктурированных и неструктурированных), нацелена на поддержку как виртуальной, так и материализованной интеграции коллекций данных, представленных как в традиционных, так и нетрадиционных моделях данных. Необходимость поддержки двух различных видов интеграции объясняется тем, что как виртуальный, так и материализованный подходы интеграции имеют свои достоинства и недостатки.

Виртуальная интеграция осуществляется с использованием технологии предметных посредников, образующих промежуточный слой между пользователем (приложением) и неоднородными источниками данных и сервисов. При этом данные из источников не материализуются в посреднике. К достоинствам виртуальной интеграции относится то, что развертывание и поддержка системы виртуальной интеграции значительно дешевле развертывания и поддержки системы материализованной интеграции (хранилища данных) исходя как из временных, так и из материальных затрат. Обычно системы виртуальной интеграции используются для интеграции источников, данные из которых трудно преобразовывать и (или) владельцами которых являются разные лица. Однако, данные в удаленных интегрируемых источниках не должны быть слишком большими, либо запросы к источникам должны обладать достаточной степенью селективности для того, чтобы объем данных, передаваемых из ресурса, не был слишком велик. Ограничения на объем данных снимаются, если источники данных сосредоточены в одном *озере данных*.

При материализованной интеграции предполагается создание хранилища данных (warehouse), в которое загружаются коллекции данных, подлежащие интеграции. В процессе загрузки происходит преобразование данных из схемы коллекции в общую схему хранилища. При необходимости, хранилища могут масштабироваться на большие объемы данных (хотя это требует соответствующих материальных затрат). Хранилища предоставляют удобную и эффективную платформу преобразования и интеграции данных, а также решения сложных аналитических задач над данными.

Функциональная структура комбинированной среды интеграции неоднородных коллекций данных (рис. 2) основана на существующей архитектуре предметных посредников.

Основные черты функциональной структуры выглядят следующим образом:

– концептуальная схема данных, предлагаемая пользователю системой интеграции (посредником), формируется независимо от интегрируемых источников;

– источники, релевантные концептуальной схеме, регистрируются в посреднике: их схемы связываются с концептуальной схемой при помощи представлений (взглядов). Определение взглядов осуществляется полуавтоматически и требует привлечения экспертов;

– пользователь задает программы (запросы) над концептуальной схемой. Эти запросы переписываются в посреднике с использованием взглядов в частичные подзапросы в терминах конкретных источников. Переписывание осуществляется в языке правил канонической модели посредников;

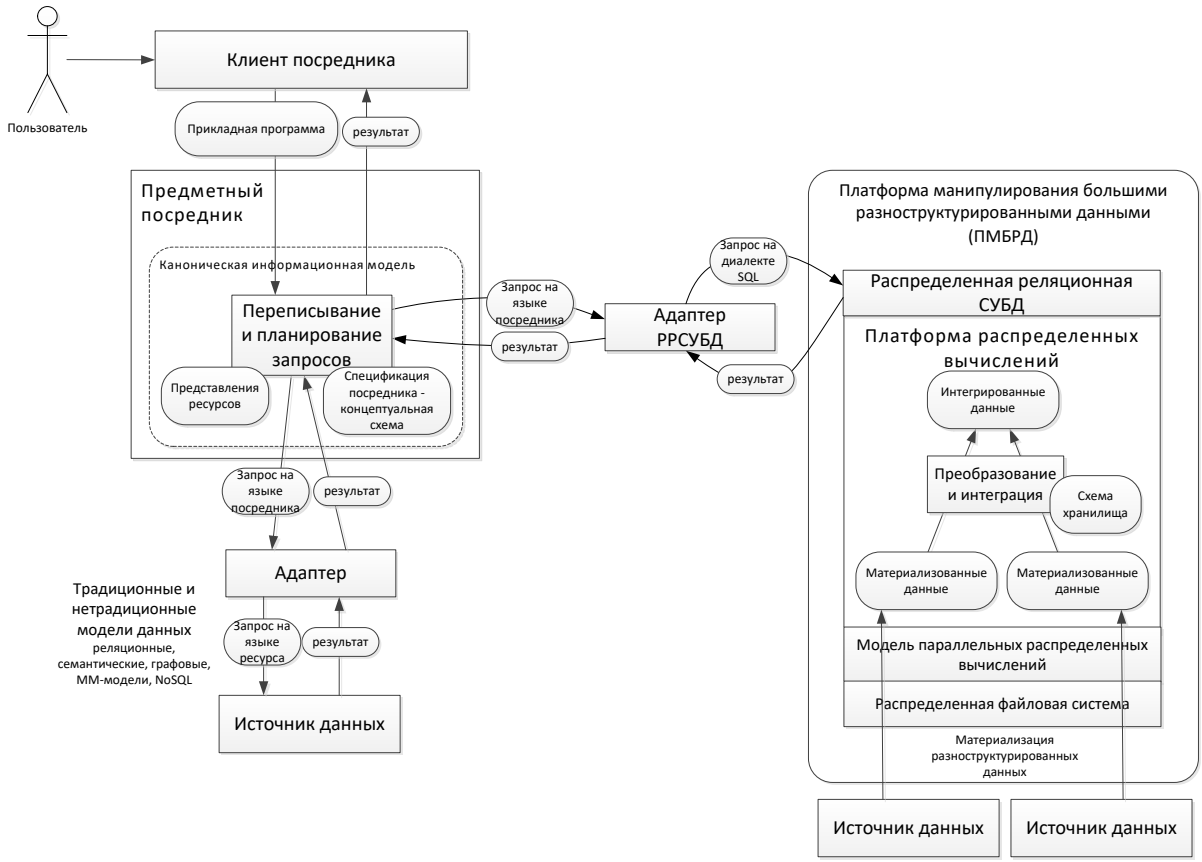

Рис. 2. Схема функциональной структуры комбинированной виртуально-материализованной среды интеграции больших неоднородных коллекций данных

– взаимодействие источников и посредника реализуется при помощи адаптеров, располагающихся между посредником и источниками. Адаптеры преобразуют запросы из языка правил канонической модели в язык источника, а также преобразуют результат исполнения запроса из формата источника в формат посредника. Формальной базой для построения адаптеров служат отображения моделей данных источников в каноническую модель, разработанные в результате унификации моделей источников.

Для *материализованной* интеграции источников в комбинированной функциональной структуре необходима масштабируемая платформа манипулирования большими разноструктурированными данными (ПМБРД), основанная на комбинации платформы распределенных вычислений и распределенной реляционной СУБД (рис. 3). ПМБРД предназначена для реализации процесса извлечения и интеграции информации из данных, состоящего из следующих этапов: идентификация источников данных, извлечение данных из источников и их преобразование к единому формату, сопоставление схем источников данных и единой схемы хранилища, трансформация данных, разрешение сущностей, слияние данных.

Процесс начинается с идентификации источников данных, которые могут быть использованы при решении задач в выбранной предметной области. Эти источники могут предоставлять как структурированные данные об объектах или субъектах предметной области (например, журналы информационных систем, данные с сенсоров), так и неструктурированные данные, например, анкеты или тексты интервью экспертов, сообщения в СМИ или социальных сетях. Извлечение структурированной информации

из разноструктурированных данных и ее интеграция становится одной из самых важных задач, стоящих перед разработчиками информационных систем. Это обусловлено тем, что существующие средства анализа данных, например, анализ многомерных кубов (OLAP) или интеллектуальный анализ данных (data mining), оперируют только структурированной информацией.

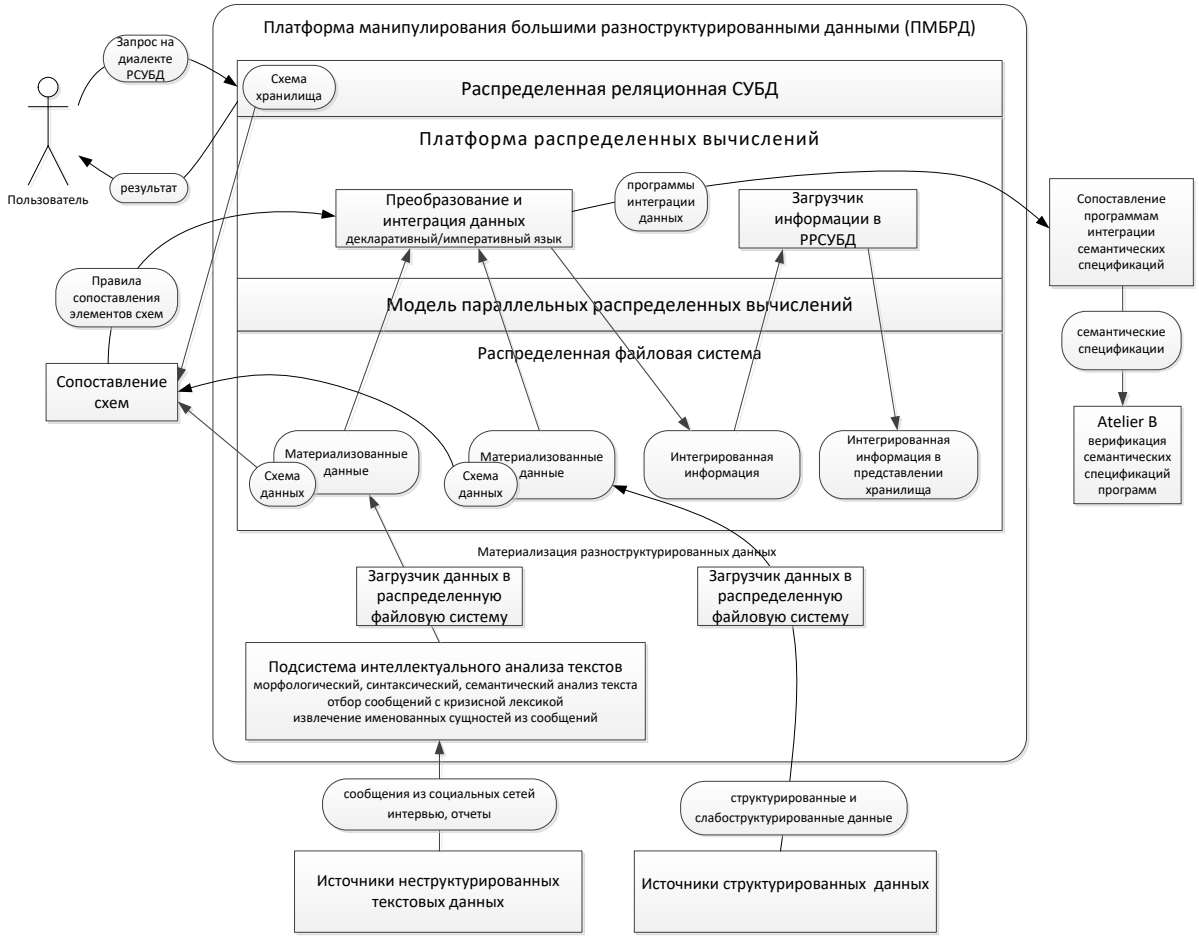

Рис. 3. Схема функциональной структуры платформы манипулирования большими разноструктурированными данными

Получаемые от источников данные сохраняются в файловой системе в виде файлов, формат и схема которых определяется каждым конкретным источником данных. Например, файлы могут быть представлены в форматах XML, CSV, или в текстовом формате. Неструктурированная текстовая информация предварительно обрабатывается средствами анализа текстов. Затем все загруженные данные преобразуются к единому формату. В качестве такового может быть выбран, например, формат JSON, позволяющий хранить как структурированные, так и слабоструктурированные данные. Собранные данные интегрируются и загружаются в единое хранилище. Интеграция данных, в свою очередь, состоит из этапов *сопоставления схем* (schema matching), *трансформации данных*, *разрешения сущностей* и *слияния данных*.

Один из самых естественных вариантов для применения в качестве платформы распределенных вычислений – это система Apache Hadoop включающая, в частности, распределенную файловую систему HDFS (Hadoop Distributed File System) и реализации программных моделей параллельных распределенных вычислений (MapReduce, Spark, Tez).

В качестве реализации реляционного хранилища данных над Hadoop может быть использован целый ряд систем: Apache Hive, Big SQL, Presto, Doris, Spark SQL.

Таким образом, в функциональной структуре обеспечивается возможность распределенного хранения, преобразования и интеграции больших разноструктурированных данных, а также унифицированный взгляд на материализованные данные через реляционную модель.

Извлечение структурированных данных из неструктурированных текстовых осуществляется в рамках отдельной *подсистемы интеллектуального анализа текстов.* Для каждого источника структурированных (например, реляционных) или слабоструктурированных (например, в формате XML) данных создается отдельный *загрузчик файлов данных в распределенную файловую систему*. Компонент *сопоставления схем* осуществляет автоматизированное сопоставление элементов схем (сущностей, атрибутов) источников и хранилища. Компонент *преобразования и интеграции данных* представляет собой набор программ на высокоуровневых языках (SQL, Jaql, HIL). Для верификации (формального доказательства необходимых свойств) программ преобразования и интеграции данных, этим программам автоматически, с использованием компонента HIL2AMN сопоставляется формальная семантика в языке спецификаций AMN. Автоматизированное доказательство необходимых свойств осуществляется при помощи программного средства Atelier B. Компонент *Загрузчик информации в РРСУБД* помещает интегрированную информацию в хранилище данных. Свои запросы пользователь задает именно к РРСУБД на языке SQL.

Функциональная структура допускает масштабирование по количеству источников данных, из которых извлекается информация и по производительности извлечения, преобразования, интеграции информации и обработке запросов к ней.

Разработанная функциональная структура материализованной интеграции данных была реализована при создании прототипа системы извлечения и интеграции информации из данных по Арктической зоне [5]. Также предложенная схемы функциональная структура была реализована при создании прототипа системы решения задач социально-политического мониторинга [45].

В комбинированной функциональной структуре ПМБРД рассматривается как еще один вид источников, подлежащий виртуальной интеграции. Интеграция становится двухслойной: материализованная интеграция осуществляется внутри ПМБРД, виртуальная – на уровне предметных посредников. Виртуальной интеграции при этом могут подлежать источники:

– данные из которых сложно или невозможно материализовать по разным причинам и (или)

– модель данных включает специфические операции и алгоритмы, адаптация которых в реляционной модели сложна и (или) неэффективна. К таким моделям относятся, например, графовые, тензорные модели и т.д.

Для унификации моделей данных источников в канонической модели предметного посредника и ее верификации применяется метод, подробно изложенный и проиллюстрированный в главе 2. Для верификации интеграции схем источников данных применяется метод, изложенный и проиллюстрированный в главе 3.

Функциональная структура основанной на правилах мультидиалектной среды направлена на обеспечение средств поддержки разнообразия данных (разнородность типов данных, их представления и семантической интерпретации), семантическую интеграцию данных и поддержку декларативных спецификаций, необходимых для разработки сложных конвейеров с использованием языков высокого уровня для решения аналитических задач в областях с интенсивным использованием данных. Функциональная структура обеспечивает согласованное совместное использование многообразия средств логического программирования, представления знаний, Семантического веба и предметных посредников для интеграции неоднородных баз данных.

Мотивацией к разработке функциональной структуры послужили исторические тенденции развития средств концептуального моделирования и появление Формата обмена правилами (Rule Interchange Format – RIF) для Сематического веба как рекомендации консорциума W3C.

Функциональная структура мультидиалектной среды предусматривает совместное функционирование модулей предметных посредников, интегрирующих данные и сервисы, а также модулей, представляющих программы, основанные на знаниях и декларативных правилах, и основана на следующих принципах:

– *Мультидиалектные концептуальные спецификации задач создаются на основе сочетания подходов предметных посредников и RIF.* Спецификации представляются в виде функциональной композиции декларативных модулей, каждый из которых основан на своем собственном языке (диалекте) с соответствующей семантикой. Языку посредников должен соответствовать один из существующих диалектов RIF (или новый, специфический диалект).

– *Модули спецификации рассматриваются как узлы одноранговой сети (peers).* Взаимодействие узлов основывается на методах обеспечения интероперабельности RIF.

– *Интеграция источников данных и сервисов обеспечивается в рамках отдельных модулей-посредников.*

– *Модули, обеспечивающие логический вывод на правилах, могут использоваться для декларативного программирования над посредниками*.

– *Мультидиалектные концептуальные спецификации задач реализуются путем пирингового взаимодействия* предметных посредников и систем на правилах, входящих в среду, при помощи техники *делегирования* (передачи фактов и правил от одного узла к другому).

Концептуальная спецификация задачи (класса задач) определяется в контексте предметной области и состоит из набора RIF-документов (*документ* является единицей спецификации RIF). Каждый документ содержит *группы правил*. Концептуальная спецификация задачи – это абстрактная программа решения задачи. Концептуализация предметной области выполняется с использованием онтологий языка OWL, включающих понятия предметной области и связи между ними, и формирующих *концептуальную схему* (рис. 4, правая часть).

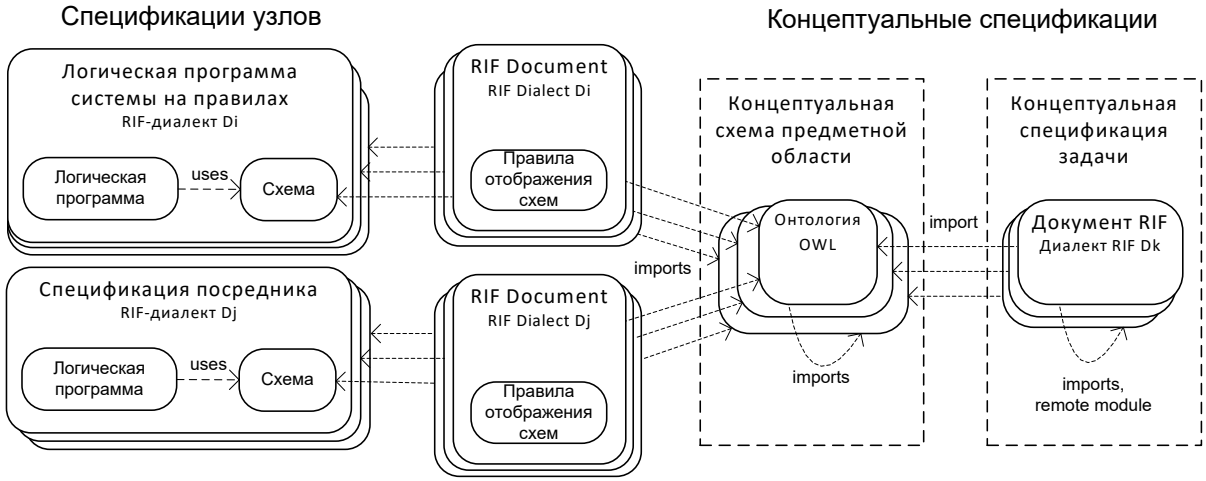

Рис. 4. Концептуальные спецификации и спецификации узлов

Извлечение результатов решения задачи осуществляется путем запроса к виртуальной базе знаний, сформированной объединенной средой посредников и систем на правилах. Запросы формулируются в терминах концептуальной схемы и предикатов

концептуальной спецификации задачи. Результатом является логическое значение (если формула замкнута) или набор кортежей значений свободных переменных формулы.

Схема S_R узла R мультидиалектной среды представляет собой набор сущностей – классов или отношений, предикатов логических программ и их атрибутов.

Система на правилах или посредник каждого узла R должны быть потребителями и поставщиками некоторого диалекта D_R (рис. 4, левая часть).

Узел *R релевантен* RIF-документу *d* концептуальной спецификации задачи (рис. 4, правая часть), если:

– D_R является поддиалектом диалекта документа *d* и

– сущности схемы S_R *онтологически релевантны* сущностям концептуальной схемы, имена которых используются в *d*.

Схема релевантного узла *отображается* в концептуальную схему. Отображение устанавливает соответствие концептуальных сущностей, на которые ссылаются документ *d*, их выражениям в терминах сущностей схемы S_R с использованием логических правил диалекта D_R. Эти правила отображения схем представляют собой отдельный RIF-документ (рис. 4, средняя часть).

Программы, определенные в концептуальной спецификации, реализуются в пиринговой среде, сформированной релевантными узлами, схемы которых связаны с концептуальной спецификацией правилами сопоставления (рис. 4, средняя часть).

Узлы взаимодействуют на основе подхода к распределенному выполнению логических программ. Основная идея этого подхода заключается в *делегировании* – передаче фактов и правил от одного узла к другому. Узел представляет собой комбинацию адаптера и системы на правилах или посредника. Адаптер преобразует программы и факты из диалекта RIF в язык системы на правилах или язык посредника и наоборот. Адаптер также реализует механизм делегирования. Передача фактов и правил между узлами выполняется на диалектах RIF (рис. 5).

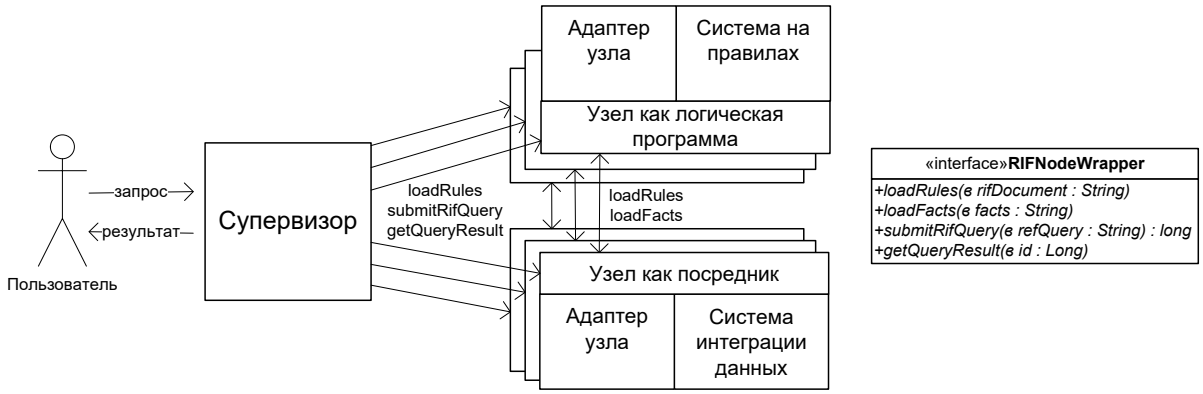

Рис. 5. Компоненты пиринговой функциональной структуры

Компонент функциональной структуры *Супервизор* хранит общую информацию о среде: концептуальные спецификации предметной области и задачи, список релевантных источников, RIF-документы отображения схем источников в концептуальную схему, переписывает документы концептуальной спецификации в RIF-программы для узлов, передает переписанные программ на узлы, релевантные документам концептуальной схемы. Адаптеры преобразуют RIF-программы в конкретные языки систем на правилах или посредников. После этого программы выполняются в пиринговой среде.

Для моделирования процессов решения задач в виде потоков работ в функциональной структуре предлагается использование диалекта продукционных правил RIF PRD. Документы, представленные при помощи данного диалекта, наряду с документами, представленными при помощи логических диалектов, становятся частью

концептуальной спецификации задачи. Поток работ состоит из набора деятельностей (task), организованных с помощью определенных конструкций – образцов потоков работ, определяющих порядок выполнения деятельностей. Характерные образцы – это, например, *последовательность* (sequence), *разделение* (split), *соединение* (join). Спецификация такой организации называется *каркасом рабочего процесса*. Каркас определяется с использованием продукционных правил RIF-PRD. Разработано расширение диалекта RIF-PRD встроенными предикатами завершения деятельности, определения и означивания переменной потока работ, определения и означивания параметра потока работ. Предикаты позволяют задавать переменные рабочего процесса и их значения и, таким образом, организовывать поток данных в рамках потока работ; задавать параметры потока работ и их значения и, таким образом, определять интерфейс рабочего процесса в терминах входных и выходных параметров; управлять порядком выполнения деятельностей потоков работ, используя условия и действия (action) продукционных правил. Разработаны шаблоны правил, предназначенные для представления основных образцов потоков работ.

Методы верификации интеграции моделей данных и схем данных, изложенные в главах 2 и 3 применяются в рассмотренной функциональной структуре при создании предметных посредников (для унификации моделей данных источников и интеграции схем источников) и адаптеров узлов систем на правилах (для разработки отображений диалектов RIF в языки систем).

На основе мультидиалектной функциональной структуры был реализован опытный образец инфраструктуры[35]. Подходы к концептуализации проблемной области на нескольких диалектах, делегированию правил, взаимодействию программ на основе правил и предметных посредников подробно объясняется и иллюстрируется на примере задачи информационной поддержки выбора диверсифицированного инвестиционного портфеля.

В **заключении** диссертационной работы перечисляются ее основные результаты.

В **приложении А** описаны применения метода унификации моделей для унификации сетевой модели данных CODASYL, процессной модели данных на основе образцов потоков работ, онтологической модели данных OWL, графовой модели данных, тензорной модели данных, языка логических правил RIF-FLD. Приведено описание этапов унификации от формального определения синтаксиса до конструирования трансформации и верификации.

В **приложении Б** описаны применения метода верификации интеграции данных в системе «Умное землепользование»[36], разрабатываемой в Почвенном институте имени В. В. Докучаева; в хранилище для поддержки поисково-спасательных операций в Арктической зоне [5][6]; в интегрированной базе данных[37] Института металлургии и материаловедения им. А.А. Байкова РАН; в базе данных двойных звезд[38], разрабатываемой в Институте астрономии РАН. Рассмотрены схемы источников данных и их семантика, схемы интегрированных баз данных и их семантика, программы интеграции данных и их семантика, свойства корректности интеграции данных, статистика доказательств корректности интеграции. Для определения схем данных источников и интегрированных баз данных в упомянутых системах использовались язык SQL, форматы XML, JSON и CSV, для определения программ интеграции данных – языки HIL, SQL (в частности, диалект PL/pgSQL), VBScript, Python.

---

[35] Kalinichenko L. A., Stupnikov S. A., Vovchenko A. E., Kovalev D. Y. Conceptual modeling of multi-dialect workflows // Информатика и ее применения, 2014. Т. 8. Вып. 4. С. 110–124.

[36] https://www.esoil.ru/activities/projects_programs/minobr/knp_2020/

[37] https://imet-db.ru/

[38] https://bdb.inasan.ru/

ПУБЛИКАЦИИ ПО ТЕМЕ ДИССЕРТАЦИИ

Монографии

1. Kalinichenko L. A., Stupnikov S. A., Martynov D. O. SYNTHESIS: a Language for Canonical Information Modeling and Mediator Definition for Problem Solving in Heterogeneous Information Resource Environments. — М.: ИПИ РАН, 2007. – 171 С.
2. Калиниченко, Ступников С. А., Земцов Н. А. Синтез канонических моделей для интеграции неоднородных источников информации. — М.: ИПИ РАН, 2005. – 85 С.

Статьи в рецензируемых изданиях из списка ВАК

3. Брюхов Д. О., Ступников С. А. Логическая реляционная модель структур данных для решения задач в предметной области управления землепользованием // Информатика и ее применения, 2022. Т. 16. Вып. 4. С. 60-65. (К1)
4. Брюхов Д. О., Скворцов Н. А., Ступников С. А. Реализация методов интеграции данных в хранилище для поддержки поисково-спасательных операций в арктической зоне // Системы высокой доступности, 2018. Т. 14. Вып. 4. С. 36–54. (К2)
5. Брюхов Д. О., Скворцов Н. А., Ступников С. А. Методы интеграции разноструктурированных данных по арктической зоне для извлечения информации, нацеленной на поддержку поисково-спасательных операций // Системы высокой доступности, 2017. Т. 13. Вып. 2. С. 3–19. (К2)
6. Ступников С. А., Брюхов Д. О., Скворцов Н. А. Анализ системного риска совместного кредитования над неоднородными коллекциями данных // Информатика и ее применения, 2016. Т. 10. Вып. 1. С. 23–33. (К1)
7. Kalinichenko L. A., Stupnikov S. A., Vovchenko A. E., Kovalev D. Y. Conceptual modeling of multi-dialect workflows // Информатика и ее применения, 2014. Т. 8. Вып. 4. С. 110–124. (К1)
8. Ступников С. А., Вовченко А. Е. Методы построения отображений коллекций, представленных в нетрадиционных моделях данных, в интегрированное представление // Системы и средства информатики, 2014. Т. 24. Вып. 4. С. 29–44. (К1)
9. Ступников С. А., Скворцов Н. А., Будзко В. И. и др. Методы унификации нетрадиционных моделей данных // Системы высокой доступности, 2014. Т. 10. Вып. 1. С. 18–39. (К2)
10. Скворцов Н. А., Вовченко А. Е., Калиниченко Л. А. и др. Модель метаданных для семантического поиска реализаций потоков работ, выраженных в виде правил // Системы и средства информатики, 2014. Т. 24. Вып. 4. С. 4–28. (К1)
11. Ступников С. А. Отображение графовых моделей данных в каноническую модель в системах с интенсивным использованием данных // Системы высокой доступности, 2014. Вып. 2. С. 13–31. (К2)
12. Будзко В. И., Ступников С. А., Калиниченко Л. А. и др. Среда интеграции больших неоднородных коллекций данных // Системы высокой доступности, 2014. Вып. 3. С. 3–19. (К2)
13. Kalinichenko L., Stupnikov S., Vovchenko A., Kovalev D. Conceptual declarative problem specification and solving in data intensive domains // Информатика и ее применения, 2013. Т. 7. Вып. 4. С. 112–139. (К1)
14. Скворцов Н. А., Ступников С. А., Калиниченко Л. А. Использование таксономий отношений класс-экземпляр в спецификациях предметных областей // Вестник ВГУ. Серия системный анализ и информационные технологии, 2012. Т. 1. С. 157–167. (К1)
15. Калиниченко Л. А., Ступников С. А. Унификация языков систем на правилах для обеспечения интероперабельности декларативных программ // Информатика и ее применения, 2012. Т. 6. Вып. 2. С. 59–76. (К1)

16. Kalinichenko L. A., Stupnikov S. A., Zakharov V. N. Extending information integration technologies for problem solving over heterogeneous information resources // Информатика и ее применения, 2012. Т. 6. Вып. 1. С. 70–77. (К1)

17. Ступников С. А., Калиниченко Л. А. Методы автоматизированного построения трансформаций информационных моделей // Системы и средства информатики, 2009. Т. 19. С. 34–62. (К1)

18. Брюхов Д. О., Вовченко А. Е., Захаров В. Н. и др. Архитектура промежуточного слоя предметных посредников для решения задач над множеством интегрируемых неоднородных распределенных информационных ресурсов в гибридной грид-инфраструктуре виртуальных обсерваторий // Информатика и ее применения, 2008. Т. 2. Вып. 1. С. 2–34. (К1)

19. Захаров В.Н., Калиниченко Л.А., Соколов И.А., Ступников С.А. Конструирование канонических информационных моделей для интегрированных информационных систем // Информатика и ее применения, 2007. Т. 1. Вып. 2. С. 15–38. (К1)

20. Ступников С. А. Автоматизация верификации уточнения при композиционном проектировании информационных систем и посредников // Системы и средства информатики. Спец. вып. Формальные методы и модели в композиционных инфраструктурах распределенных информационных систем, 2005. С. 96–119. (К1)

21. Ступников С. А. Отображение спецификаций, выраженных средствами ядра канонической модели, в язык AMN // Системы и средства информатики. Спец. вып. Формальные методы и модели в композиционных инфраструктурах распределенных информационных систем, 2005. С. 69–95. (К1)

22. Калиниченко Л. А., Мартынов Д. О., Ступников С. А. Переписывание запросов на основе взглядов в типизированной среде посредников // Системы и средства информатики. Спец. вып. Формальные методы и модели в композиционных инфраструктурах распределенных информационных систем, 2005. С. 152–183. (К1)

23. Ступников С. А., Брюхов Д. О. Представление UML и OCL в канонической информационной модели // Системы и средства информатики. Спец. вып. Формальные методы и модели в композиционных инфраструктурах распределенных информационных систем, 2005. С. 120–129. (К1)

24. Ступников С. А. Формальная семантика ядра канонической объектной информационной модели // Системы и средства информатики. Спец. вып. Формальные методы и модели в композиционных инфраструктурах распределенных информационных систем, 2005. С. 40–68. (К1)

25. Земцов Н. А., Ступников С. А. Формальное моделирование спецификаций процессов для композиционного проектирования потоков работ // Системы и средства информатики, 2004. Т. 14. С. 186–198. (К1)

Публикации в журналах и сборниках трудов конференций, индексируемых в базах данных Web of Science и\или Scopus

26. Kalinichenko L., Stupnikov S. Synthesis of the canonical models for database integration preserving semantics of the value inventive data models // Lecture Notes in Computer Science, 2012. Vol. 7503. P. 223–239. (Scopus Q2)

27. Stupnikov S. A. Formal Semantics and Verification of Procedural SQL Programs Implementing Materialized Data Integration // Lobachevskii Journal of Mathematics, 2025. Vol. 46. No. 4. P. 1511–1525. (Scopus Q2)

28. Stupnikov S. A. Verification of Global and Local as View Data Integration in an Object Data Model // Lobachevskii Journal of Mathematics, 2025. Vol. 46. P. 4926–4940. (Scopus Q2)

29. Ступников С.А. Верификация интеграции данных в интегрированной системе баз данных по свойствам неорганических веществ и материалов // Учен. зап. Казан. ун-та. Сер. Физ.-матем. науки, 2025. Т. 167. Вып. 2. С. 1–17. (Scopus Q4)

30. Stupnikov S. A. Query-driven verification of data integration in the rdf data model // Lobachevskii Journal of Mathematics, 2023. Vol. 44. No. 1. P. 205–218. (Scopus Q2)

31. Palagashvili A.M., Stupnikov S.A. Reversible Mapping of Relational and Graph Databases // Pattern Recognit. Image Anal., 2023. Vol. 33. P. 113–121. (Scopus Q3)

32. Sazontev V.V., Stupnikov S.A. An Extensible Approach to Searching and Selecting Data Sources for Materialized Big Data Integration in Distributed Computing Environments // Pattern Recognit. Image Anal., 2023. Vol. 33. P. 147–156. (Scopus Q3)

33. Skvortsov N. A., Stupnikov S. A. A semantic approach to workflow management and reuse for research problem solving // Data Intelligence, 2022. Vol. 4. No. 2. P. 439 – 454. (Scopus Q1)

34. Stupnikov S. Applying model-driven approach for data model unification // Communicat. in Computer and Information Science, 2021. Vol. 1401. P. 212–232. (Scopus Q4)

35. Tang W., Stupnikov S. A transformation of the rdf mapping language into a high-level data analysis language for execution in a distributed computing environment // Communicat. in Computer and Information Science, 2021. Vol. 1427. P. 74–91. (Scopus Q4)

36. Stupnikov S., Kalinichenko L. Extensible unifying data model design for data integration in fair data infrastructures // Communications in Computer and Information Science, 2019. Vol. 1003. P. 17–36. (Scopus Q3)

37. Skvortsov N. A., Stupnikov S. A. Formalizing requirement specifications for problem solving in a research domain // Communications in Computer and Information Science, 2019. Vol. 1064. P. 266–279. (Scopus Q3)

38. Sazontev V., Stupnikov S. An extensible approach for materialized big data integration in distributed computation environments // 2019 Ivannikov Memorial Workshop (IVMEM). — IEEE, 2019. P. 33–38. (Scopus)

39. Stupnikov S. Rule-based specification and implementation of multimodel data integration // Communications in Computer and Information Science, 2018. Vol. 822. P. 198–212. (Scopus Q3)

40. Stupnikov S., Kalinichenko L. Fair data based on extensible unifying data model development // CEUR Workshop Proceedings, 2018. Vol. 2277. P. 9–13. (Scopus)

41. Stupnikov S. Semantics and verification of entity resolution and data fusion operations via transformation into a formal notation // Communications in Computer and Information Science, 2017. Vol. 706. P. 145–162. (Scopus Q3)

42. Stupnikov S. Specification and implementation of multimodel data integration rules // CEUR Workshop Proceedings, 2017. Vol. 2022. P. 197–205. (Scopus)

43. Stupnikov S. Formal semantics and verification of entity resolution and data fusion operations // CEUR Workshop Proceedings, 2016. Vol. 1752. P. 159–167. (Scopus)

44. Stupnikov S., Miloslavskaya N., Budzko V. Unification of graph data models for heterogeneous security information resources' integration // Proceedings of the International Conference on Open and Big Data OBD. — IEEE, 2015. P. 457–464. (Scopus)

45. Брюхов Д. О., Ступников С. А., Калиниченко Л. А., Вовченко А. Е. Извлечение информации из разноструктурированных данных и ее приведение к целевой схеме // CEUR Workshop Proceedings, 2015. Vol. 1536. P. 81–90. (Scopus)

46. Kalinichenko L., Stupnikov S., Vovchenko A., Kovalev D. Multi-dialect workflows // Lecture Notes in Computer Science, 2014. Vol. 8716. P. 352–365. (Scopus Q2)

47. Ступников С., Вовченко А. Комбинированная виртуально-материализованная среда интеграции больших неоднородных коллекций данных // CEUR Workshop Proceedings, 2014. Т. 1297. С. 201–210. (Scopus)

48. Ступников С. Отображение графовой модели данных в каноническую объектно-фреймовую информационную модель при создании систем интеграции неоднородных

информационных ресурсов // CEUR Workshop Proceedings, 2013. Т. 1108. С. 85–94. (Scopus)

49. Скворцов Н.А., Брюхов Д.О., Калиниченко Л.А., Ковалев Д.Ю., Ступников С.А. Метаданные о научных методах для обеспечения их повторного использования и воспроизводимости результатов // CEUR Workshop Proceedings, 2013. Т. 1108. С. 70–78. (Scopus)

50. Kalinichenko L., Stupnikov S., Vovchenko A., Kovalev D. Rule-based multi-dialect infrastructure for conceptual problem solving over heterogeneous distributed information resources // Advances in Intelligent Systems and Computing, 2013. Vol. 241. P. 61–68. (Scopus)

51. Kalinichenko L., Stupnikov S. Owl as yet another data model to be integrated // CEUR Workshop Proceedings, 2011. Vol. 789. P. 178–189. (Scopus)

52. Ступников С. Унификация модели данных, основанной на многомерных массивах, при интеграции неоднородных информационных ресурсов // CEUR Workshop Proceedings, 2012. Т. 934. С. 42–52. (Scopus)

53. Вовченко А.Е., Захаров В.Н., Калиниченко Л.А., Ковалёв Д.Ю., Рябухин О.В., Скворцов Н.А., Ступников С.А. Многоуровневые спецификации в концептуальном и онтологическом моделировании // CEUR Workshop Proceedings, 2011. Vol. 803. P. 17–25. (Scopus)

54. Kalinichenko L. A., Stupnikov S. A. Heterogeneous information model unification as a pre-requisite to resource schema mapping // Proceedings of the V Conference of the Italian Chapter of Association for Information Systems itAIS. — Heidelberg, 2010. P. 373–380. (Scopus)

55. Kalinichenko L.A., Briukhov D.O., Martynov D.O., Stupnikov S.A. Mediation framework for enterprise information system infrastructures: Application-driven approach // Proceedings of the Ninth International Conference on Enterprise Information Systems. — 2007. Vol. DISI. P. 246–251. (Scopus)

56. Stupnikov S.A., Kalinichenko L.A., Bressan S. Interactive discovery and composition of complex web services // Lecture Notes in Computer Science, 2006. Vol. 4152. P. 216–231. (Scopus Q2)

57. Kalinichenko L. A., Stupnikov S. A., Zemtsov N. Extensible canonical process model synthesis applying formal interpretation // Lecture Notes in Computer Science, 2005. Vol. 3631. P. 183–198. (Scopus Q2)

58. Kalinichenko L. A., Martynov D. O., Stupnikov S. A. Query rewriting using views in a typed mediator environment // Lecture Notes in Computer Science, 2004. Vol. 3255. P. 37–53. (Scopus Q2)

Публикации в других изданиях

59. Калиниченко Л. А., Ступников С. А. Анализ мотивации, целей и подходов проекта унификации языков на правилах // Труды Второго симпозиума «Онтологическое моделирование». — ИПИ РАН, 2011. С. 79–122.

60. Калиниченко Л. А., Ступников С. А. Каркас логических диалектов rif: анализ и демонстрация применения // Труды Второго симпозиума «Онтологическое моделирование». — ИПИ РАН, 2010. С. 123–150.

61. Ступников С. А., Скворцов Н. А. Взаимное отображение канонической информационной модели и языка owl 2 // Труды 12-й Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» RCDL. — Казань, 2010. С. 392–398.

62. Ступников С. А., Скворцов Н. А. Семантическая трансформация канонической информационной модели в формальный язык спецификаций для верификации уточнения // Труды 12-й Всероссийской научной конференции «Электронные

библиотеки: перспективные методы и технологии, электронные коллекции» RCDL. — Казань, 2010. С. 383–391.

63. Stupnikov S., Kalinichenko L. Methods for semi-automatic construction of information models transformations // Proc. of the Workshop “Model–Driven Architecture: Foundations, Practices and Implications”. — Riga, 2009. P. 432–440.

64. Kalinichenko L. A., Stupnikov S. A. Constructing of mappings of heterogeneous information models into the canonical models of integrated information systems // Proceedings of the 12th East European Conference “Advances in Databases and Information Systems” — Pori, 2008. P. 106–122.

65. Скворцов Н. А., Ступников С. А. Использование онтологии верхнего уровня для отображения информационных моделей // Труды Десятой Всероссийской научной конференции Электронные библиотеки: перспективные методы и технологии, электронные коллекции RCDL. — 2008. С. 122–127.

66. Briukhov D. O., Kalinichenko L. A., Stupnikov S. A. Compositional approach for heterogeneous sources registration at a subject mediator // Proceedings of the workshop of VLDB 2003 “Emerging Database Research In East Europe”. — Berlin, 2003. P. 5–11.

67. Stupnikov S. A., Kalinichenko L. A., Dong J. S. Applying csp-like workflow process specifications for their refinement in AMN by pre-existing workflows // Proceedings of the 6th East European Conference on Advances in Databases and Information Systems. — Bratislava, 2002. — Vol. 2: Research Communications. P. 206–216.

68. Ступников С. А., Калиниченко Л. А. Формирование базы метаинформации на основе спецификаций канонической модели // Проблемы программирования, 2002. Вып. 1-2. С. 301–308.

Список программ для ЭВМ

69. С.А. Ступников. Трансформация программ языка High-level Integration Language (HIL) в Нотацию абстрактных машин. Свидетельство о государственной регистрации программы для ЭВМ № 2018612285 от 14.02.2018.

70. С.А. Ступников. Генератор заготовок ATL-трансформаций информационных моделей. Свидетельство о государственной регистрации программы для ЭВМ № 2012610640 от 10.01.12.

71. С.А. Ступников. Генератор заготовок обратных ATL-трансформаций информационных моделей. Свидетельство о государственной регистрации программы для ЭВМ № 2012610642 от 10.01.12.

72. С.А. Ступников. Трансформация спецификаций языка СИНТЕЗ в Нотацию Абстрактных Машин. Свидетельство о государственной регистрации программы для ЭВМ № 2012611337 от ЭВМ 02.02.12.

73. С.А. Ступников. Абстрактный и конкретный синтаксис языка СИНТЕЗ. Свидетельство о государственной регистрации программы для ЭВМ № 2012610641 от 10.01.12.

74. С.А. Ступников. Графический интерфейс Конструктора унифицирующих информационных моделей. Свидетельство об официальной регистрации программы для ЭВМ N 2008614240 от 05.09.2008.

75. С.А. Ступников. Программа построения эталонных схем информационных моделей на основании формальных грамматик моделей. Свидетельство об официальной регистрации программы для ЭВМ N 2008614239 от ЭВМ 05.09.2008.

76. С.А. Ступников. Генератор заготовок трансляторов информационных моделей: Свидетельство об официальной регистрации программы для ЭВМ N 2008614241 от 05.09.2008.

77. С.А. Ступников. Программа загрузки спецификаций на языке UML XMI в репозиторий метаинформации с одновременным преобразованием в каноническую

информационную модель: Свидетельство об официальной регистрации программы для ЭВМ N 2007613885 от 12.09.2007.

78. С.А. Ступников, А.Е. Вовченко. Программа загрузки схем реляционных баз данных в репозиторий метаинформации с одновременным преобразованием в каноническую информационную модель. Свидетельство об официальной регистрации программы для ЭВМ N 2007613884 от 12.09.2007.

79. С. А. Ступников. Программа автоматического отображения спецификаций канонической информационной модели в Нотацию Абстрактных Машин. Свидетельство об официальной регистрации программы для ЭВМ № 2005611080 от РФ 05.05.2005.

80. С. А. Ступников. Программа загрузки спецификаций канонической информационной модели в репозиторий метаинформации. Свидетельство об официальной регистрации программы для ЭВМ №:2005611083 от 05.05.2005.