

# АРХИТЕКТУРА ПРОМЕЖУТОЧНОГО СЛОЯ ПРЕДМЕТНЫХ ПОСРЕДНИКОВ ДЛЯ РЕШЕНИЯ ЗАДАЧ НАД МНОЖЕСТВОМ ИНТЕГРИРУЕМЫХ НЕОДНОРОДНЫХ РАСПРЕДЕЛЕННЫХ ИНФОРМАЦИОННЫХ РЕСУРСОВ В ГИБРИДНОЙ ГРИД-ИНФРАСТРУКТУРЕ ВИРТУАЛЬНЫХ ОБСЕРВАТОРИЙ\*

Д. О. Брюхов<sup>1</sup>, А. Е. Вовченко<sup>2</sup>, В. Н. Захаров<sup>3</sup>, О. П. Желенкова<sup>4</sup>, Л. А. Калиниченко<sup>5</sup>, Д. О. Мартынов<sup>6</sup>, Н. А. Скворцов<sup>7</sup>, С. А. Ступников<sup>8</sup>

**Аннотация:** Рассматривается архитектура промежуточного слоя предметных посредников для решения научных задач над множеством интегрируемых неоднородных распределенных информационных ресурсов в гибридной грид-инфраструктуре виртуальной обсерватории (ВО). Архитектура реализована как объединение системы поддержки ВО АстроГрид, разработанной в Великобритании, и средств поддержки предметных посредников, созданных в ИПИ РАН. Реализован подход, при котором для класса приложений формируется спецификация предметной области посредников независимо от существующих информационных ресурсов. Создание прототипа гибридной архитектуры требует сопряжения исполнительных механизмов двух инфраструктур (АстроГрида и предметных посредников), поэтому основное внимание уделено проблемам переписывания запросов к посредникам в планы их реализации над конкретными информационными ресурсами. Приведено краткое описание объединенной архитектуры исполнительных механизмов АстроГрида и средств поддержки предметных посредников. Дан пример реализации предметного посредника для решения задач поиска далеких галактик в гибридной архитектуре. Показаны отличия реализованного подхода от известных прототипов интеграции баз данных в ВО. Разработанные средства планируется использовать при решении задач Российской ВО (РВО).

**Ключевые слова:** предметный посредник; каноническая информационная модель; виртуальная обсерватория; унификатор информационных моделей; уточнение; переписывание формул; семантическая интеграция неоднородных информационных ресурсов; регистрация ресурсов в посреднике; онтологическая модель; релевантные посреднику ресурсы; промежуточный слой; описание предметной области задачи, движимое приложениями

## 1 Введение

В различных областях науки наблюдается экспоненциальный рост объема получаемых экспериментальных (наблюдательных) данных. Например, в астрономии текущий и ожидаемый темп роста данных от наземных и космических телескопов удваивается в течение периода от шести месяцев до одного года. Сложность использования таких данных увеличивается еще и вследствие

их естественной разнородности. Число организаций, получающих данные наблюдений в отдельных областях науки в мире, велико. Разнообразие (информационная несогласованность) получаемой информации вызывается, в частности, не только большим числом организаций, производящих наблюдения, и их независимостью, но и разнообразием объектов наблюдения, непрерывным и быстрым совершенствованием техники наблюдений, вызывающим адекватные изменения структуры и со-

\* Работа выполнена при частичной финансовой поддержке РФФИ (гранты 06-07-08072-офи-а, 06-07-89188-а, 08-07-00157-а) и программы ОИТВС РАН «Фундаментальные основы информационных технологий и систем» (проект 1-10).

<sup>1</sup> Институт проблем информатики Российской академии наук, brd@ipi.ac.ru

<sup>2</sup> Институт проблем информатики Российской академии наук, itsnein@gmail.com

<sup>3</sup> Институт проблем информатики Российской академии наук, vzakharov@ipiran.ru

<sup>4</sup> Специальная астрофизическая обсерватория Российской академии наук, zhe@sao.ru

<sup>5</sup> Институт проблем информатики Российской академии наук, leonidk@synth.ipi.ac.ru

<sup>6</sup> Институт проблем информатики Российской академии наук, domartynov@gmail.com

<sup>7</sup> Институт проблем информатики Российской академии наук, nskv@ipi.ac.ru

<sup>8</sup> Институт проблем информатики Российской академии наук, ssa@ipi.ac.ru

держания накапливаемой информации. Это приводит к необходимости использования неоднородной, распределенной информации, накопленной в течение значительного периода наблюдений технологически различными инструментами.

Чрезвычайно быстро развивается также программный инструментарий, включающий многообразные сервисы для поддержки различных видов обработки информации при решении научных задач и проведении исследований. Такие сервисы производятся различными научными организациями, их описания неоднородны и неполны.

Увеличивающийся разрыв между исследователями и источниками данных и сервисов приводит к необходимости поиска новых путей создания информационных систем, в которых особое внимание было бы сосредоточено на специальных средствах организации решения задач над множеством распределенных информационных ресурсов (данных и программ), накапливаемых в разнообразных научных центрах. Разработан (разрабатывается) ряд инфраструктур, которые технически способствуют организации решения задач в такой среде. Среди них Веб-сервисы, Гриды данных, Семантический Веб, инфраструктуры онтологического моделирования, интеграции информационных ресурсов, интероперабельные инфраструктуры промежуточного слоя и др. Вместе с тем сложны и все еще далеки от решения проблемы семантики, возникающие при:

- (1) определении понятий предметной области;
- (2) отображении и интеграции контекстов предметных областей информационных ресурсов в контекст предметной области задачи;
- (3) идентификации релевантных задач информационных ресурсов и формировании их композиций;
- (4) доказательно правильном отображении информационных моделей ресурсов в общую информационную модель задачи;
- (5) интеграции схем ресурсов в схеме предметной области задачи и устранении разнообразных конфликтов;
- (6) выявлении семантически подобных компонентов ресурсов в процессе интеграции схем;
- (7) адекватном преобразовании формул (запросов) программы решения задачи, выраженных в терминах схемы предметной области задачи, в формулы, выраженные в схемах релевантных ресурсов, и пр.

Следует заметить, что аналогичные проблемы создания и развития информационных систем возникают не только в научных применениях, но и в

корпоративных информационных системах, создаваемых для решения задач в разнообразных областях государственного управления, производства, бизнеса. Достаточно назвать системы, конструируемые в рамках программ «Электронная Москва», «Электронная Россия», системы для решения задач правоохранительных органов, для инспекции безопасности дорожного движения, банковские системы, системы управления транспортом и др.

Настоящая статья в качестве применений ограничивается рассмотрением инфраструктур информационных систем для науки, которые в последнее время приобретают вид международных ВО. Виртуальные обсерватории, в свою очередь, могут быть отнесены к классу корпоративных информационных систем, поэтому результаты настоящей работы применимы и к последним. Анализ инфраструктуры РВО для астрономии дан в ее аванпроекте [1]. Значительное внимание в нем занимает анализ состояния и решений наиболее продвинутых в мире проектов ВО: проекта Национальной ВО (NVO) США, проекта АстроГрид Великобритании, проекта Европейской ВО Euro-VO. Координация проекта РВО с международными проектами — основная стратегическая линия развития РВО. Основной координирующей организацией создания проектов ВО в различных странах мира является Альянс Международной ВО (IVOA) [2], в который входит и Россия. Альянс Международной ВО разрабатывает совокупность стандартов, которые должны обеспечить совместимость ВО, создаваемых в различных странах мира. Технически инфраструктура, продвигаемая IVOA, является сервис-ориентированной. В стандартах различаются сервисы доступа к астрономическим данным, вычислительные сервисы обработки данных, сервисы регистрации разнообразных сервисов, позволяющие публиковать и обнаруживать их. Подробный анализ разрабатываемых стандартов IVOA дан в [2, 3]. Слабым местом стандартов IVOA является недостаточность средств интеграции неоднородных астрономических ресурсов для решения над ними разнообразных научных задач. Поэтому в инфраструктуре РВО дополнительно к инфраструктуре IVOA вводится концепция предметных посредников, позволяющих задавать определение прикладных областей для формулирования и решения классов научных задач в терминах понятий этих областей, структур информационных объектов, декларативно объявляемых сервисов и процессов. Посредники располагаются между исследователями, формулирующими задачи в терминах посредников, и разнообразными распределенными информационными ресурсами (данными, сер-

висами, процессами), необходимыми для решения задачи.

В итоге инфраструктура РВО определена многоуровневой [1] и включает (при движении снизу вверх) уровень информационных ресурсов, промежуточный слой, обеспечивающий интероперабельность ресурсов благодаря технической унификации их интерфейсов и введению дистанционных механизмов обращения к ресурсам (адаптеры, осуществляющие преобразование запросов, выраженных в канонической информационной модели посредников, в их представление в информационной модели ресурса, включаются в состав промежуточного слоя), уровень *предметных посредников*, каждый из которых создает спецификацию предметной области для решения некоторого класса задач, используя каноническую информационную модель («эсперанто») для представления семантики предметной области и унифицированного отображения разнообразных видов информационных моделей ресурсов (моделей данных, сервисных моделей, онтологических моделей, процессных моделей), уровень задач (приложений), формулируемых в терминах одного или нескольких посредников. Следует заметить, что посредники, в свою очередь, могут рассматриваться как информационные ресурсы для использования в посредниках более высокого уровня. В такой инфраструктуре предметные посредники играют ключевую роль для решения семантических проблем, перечисленных выше. В частности, при интеграции неоднородных ресурсов в посреднике нужно уметь семантически отождествлять объекты, представленные в различных информационных моделях, и семантически правильно отображать схемы интегрируемых ресурсов в схему посредника. Поскольку в общем случае ресурсы неоднородны (представлены в различных моделях), при интеграции неоднородных ресурсов для однородного представления их семантики требуется приведение различных информационных моделей к унифицированному виду в рамках некоторой унифицирующей информационной модели, которая называется *канонической*.

Для унификации разнородных спецификаций прежде всего требуется умение сопоставлять спецификации различных ресурсов друг с другом так, чтобы можно было отвечать на вопрос, можно ли при реализации посредника использовать спецификацию существующего ресурса вместо фрагмента спецификации посредника. Для этого достаточно доказать, что рассматриваемые спецификации находятся в отношении уточнения. Говорят, что спецификация  $A$  уточняет спецификацию  $D$ , если  $A$  можно использовать вместо  $D$  так, что пользователь  $D$  не будет замечать этой замены. Средств

ва доказательства факта уточнения спецификации некоторого компонента спецификацией другого компонента (реализуемые на основе теоретико-модельных нотаций и соответствующего инструментария [4–6]) составляют фундамент применяемых методов конструирования унифицирующих (канонических) моделей представления информации в посредниках. Каноническая информационная модель служит в качестве общего языка, эсперанто, для адекватного выражения семантики разнородных моделей представления информации, используемых в разнообразных информационных ресурсах, релевантных посреднику. Методы отображения информационных моделей и синтеза расширяемых канонических информационных моделей для слоя предметных посредников подробно рассмотрены в [7]. Там же предложена архитектура Унификатора информационных моделей.

Задачей Унификатора моделей является унификация множества информационных моделей (называемых *исходными*), использующихся в различных информационных ресурсах некоторой информационной системы. Унификация исходной модели  $R$  есть приведение ее к канонической информационной модели  $C$ , т.е. создание такого расширения  $E$  канонической модели (которое может быть и пустым) и такого отображения  $M$  исходной модели в расширенную каноническую, что исходная модель *уточняет* расширенную каноническую модель. Уточнение моделей означает, что для любой допустимой спецификации  $r$  модели  $R$  ее образ  $M(r)$  при отображении  $M$  уточняется спецификацией  $r$ . Синтез канонической модели заключается в построении указанных расширений ядра канонической модели для всех исходных моделей и объединении таких расширений. В качестве ядра канонической информационной модели применяется язык СИНТЕЗ [8], который ориентирован на спецификацию предметных посредников и синтез канонических моделей. В настоящей работе Унификатор моделей предполагается входящим в качестве неотъемлемой части слоя предметных посредников. Его архитектура и подробности конструирования определены в [7].

Другой конкретной совокупностью семантических проблем создания посредников для ВО являются проблемы (1), (2), (5) и (6) из приведенного выше списка. Эти проблемы должны решаться при определении предметного посредника и регистрации в нем релевантных посреднику информационных ресурсов.

Спецификация предметного посредника для класса задач включает определения понятий предметной области, выражаемых соответствующими онтологическими спецификациями, специфика-

ции классов объектов предметной области, спецификации типов экземпляров названных классов и их методов, определяющих их поведение, спецификации процессов решения задач данного класса как совмещенных во времени последовательностей действий, реализуемых методами классов, сервисами и другими процессами. Предполагается, что такие спецификации преобразуются в спецификации канонической модели, имеющей формальную семантику. Результат определения предметного посредника, выполняемого заинтересованным научным сообществом, составляет спецификацию посредника, образуемую в результате достижения консенсуса в таком сообществе, а сама деятельность по спецификации посредника называется периодом его *консолидации*.

Регистрация релевантных посреднику ресурсов рассматривается как задача композиционного проектирования систем [9, 10]. Регистрация ресурсов есть процесс целенаправленной трансформации спецификаций, включающий декомпозицию спецификаций посредника на непротиворечивые фрагменты, поиск среди спецификаций релевантных ресурсов подходящих типов данных — кандидатов для уточнения ими спецификаций типов посредника, построение выражений, определяющих классы ресурсов в виде композиции классов посредника. Для подобного манипулирования спецификациями разработано специальное *исчисление спецификаций* [11]. В нем предложен принцип декомпозиции спецификаций типов в набор редуктов спецификаций типов, служащих основными единицами повторного использования и композиции. Введена операция определения *наибольшего общего редукта* спецификаций типов ресурсов и посредника. На основе частично упорядоченного множества спецификаций типов определены также решетка и алгебра типов. Эти структуры послужили теоретической базой для разработки репозитория метаинформации, хранящего спецификации посредников, ресурсов, а также промежуточные спецификации, возникающие в процессе отображения таких определений.

Принципиальным моментом в этой схеме является реализация доказательства уточнения фрагментов спецификаций посредника спецификациями ресурсов в процессе построения отображений таких спецификаций. Идентификация релевантных ресурсов (предшествующая регистрации) основана на использовании трех моделей — *модели метаданных*, характеризующих свойства ресурсов, собранных в некотором реестре, *онтологической модели*, позволяющей формально определять понятия

предметной области посредника или ресурса, и *канонической модели*, позволяющей формально определять структуру и поведение объектов предметной области посредника и информационных ресурсов. Рассуждения в канонической и в онтологической модели основаны на семантике канонической модели и средствах доказательства уточнения. При этом в онтологической модели необходимо достичь согласования понятийной семантики спецификаций посредника и регистрируемых в нем ресурсов. Рассуждения в модели метаданных являются эвристическими на основе нефункциональных требований к нужным в классе задач ресурсам (к таковым относятся, в частности, показатели качества данных в ресурсах). Необходимые модели метаданных и алгоритмы поиска составляют часть метода. В целях проектирования спецификации посредника и ресурсов задаются в однородном их представлении в канонической модели, хотя для этого может потребоваться преобразование в такую модель из некоторого другого языка спецификаций, например из UML<sup>1</sup>.

Сложной проблемой регистрации ресурсов в посреднике является согласование прикладных контекстов посредника и конкретных ресурсов. В разработанном инструментарии регистрации [10] такое согласование осуществляется на основе онтологического подхода. Онтологические спецификации играют роль «клея» фрагментов спецификаций посредника и ресурсов для их семантического отображения и композиции. Онтологические определения аннотируют элементы спецификаций посредника и спецификаций ресурсов (заданных в форме типов, классов, процессов). В действующем подходе используется вербальная (задаваемая подобно определениям терминов в толковом словаре) форма определения понятий. Вербальное представление онтологии дополняется более формальными спецификациями на основе абстрактных типов данных канонической модели и техники доказательства уточнения.

Перечисленные подходы положены в основу разработанного прототипа средств идентификации и регистрации информационных ресурсов в посреднике [10]. При этом онтологические спецификации используются для идентификации классов посредника, семантически релевантных классам ресурса. Максимальное подмножество информации класса ресурса, релевантное классу посредника, устанавливается на основании максимального общего фрагмента спецификаций соответствующих типов экземпляров этих классов. Конкретизирующие типы, устраняющие возникающие конфликты (зна-

<sup>1</sup>UML — Unified Modeling Language — унифицированный язык моделирования.

чений, структур данных и поведения) в названных типах экземпляров, определяются так, чтобы тип экземпляра класса посредника уточнялся бы типом экземпляра класса ресурса. Основным результатом регистрации является выражение, определяющее, как класс ресурса выражается через посредство классов посредника. Архитектура и принципы построения инструмента регистрации ресурсов в посреднике, который является неотъемлемой частью слоя посредников, определены в [10] и рассматриваются в деталях в настоящей статье не будут.

Существенным для данной работы является то обстоятельство, что в Великобритании развивается система АстроГрид [12] как полная инфраструктура для создания систем решения научных задач в астрономических ВО. Инфраструктура АстроГрида позволяет конструировать гибкие распределенные структуры ВО, компонентами которых могут быть разнообразные средства хранения данных и доступа к ним, средства размещения и временного хранения файлов в процессе решения задач группами пользователей, реестры метаданных, в которых регистрируются ресурсы ВО, средства программирования приложений и организации их интероперабельного исполнения, средства создания и исполнения потоков работ, которые позволяют решать сложные задачи в распределенной системе. Разнообразные подмножества такой совокупности компонентов могут быть установлены на различных машинах. Взаимодействие между компонентами основано на использовании технологии Веб-сервисов, а сами компоненты являются Веб-приложениями. Компоненты удовлетворяют стандартам IVOA там, где такие стандарты уже существуют. В случае, если стандартов еще нет, АстроГрид предлагает проекты стандартов для рассмотрения IVOA.

По поддерживаемым функциям сервисы АстроГрида, в основном, соответствуют требуемым для реализации инфраструктуры РВО. АстроГрид установлен в Москве для использования астрономами в качестве Центра коллективного пользования [13]. Вместе с тем, АстроГрид не предусматривает развитых средств решения задач над множествами неоднородных информационных ресурсов, подобных предметным посредникам, поэтому в 2006 г. при поддержке РФФИ был инициирован проект создания гибридной архитектуры, объединяющей возможности АстроГрида с архитектурой предметных посредников. Целью настоящей статьи является обзор достигнутых в этом проекте результатов развития такой гибридной архитектуры, акцентируя новые методы и средства определения предметных областей, информационных ресурсов и решения задач в гибридной инфраструктуре посредников и АстроГрида.

Поскольку создание гибридной инфраструктуры касается, главным образом, сопряжения исполнительных механизмов двух инфраструктур, в статье им и уделено основное внимание. Дальнейшее изложение материала в статье структурировано следующим образом. В разд. 2 рассматриваются основные виды предметных посредников и методология интеграции неоднородных ресурсов в них. В разд. 3 определена архитектура исполнительных механизмов предметных посредников. В разд. 4 особое внимание уделено переписыванию запросов к посредникам в планы их реализации над конкретными информационными ресурсами. В разд. 5 рассматриваются особенности инфраструктуры системы АстроГрид. В разд. 6 приведено краткое описание объединенной архитектуры АстроГрида и средств поддержки исполнительного слоя предметных посредников. Пример реализации простого предметного посредника для решения задач поиска далеких галактик в объединенной архитектуре дан в разд. 7. В разд. 8 дана краткая характеристика развития методов интеграции неоднородных источников информации в инфраструктурах ВО. Наконец, заключение статьи подводит итог обсуждению и намечает планы дальнейшего развития работы.

## 2 Виды предметных посредников и методология интеграции неоднородных ресурсов в посредниках

Одним из широко распространенных взглядов на ВО является рассмотрение их как инфраструктур, предназначенных для интеграции данных и сервисов в различных исследовательских центрах, с целью предоставления таким образом всем ученым доступа к информации, необходимой для решения задач. Виртуальная обсерватория может рассматриваться как специальная модель межорганизационного взаимодействия. Основным аспектом моделирования такой распределенной деятельности в e-science, предусматривающей совместную работу организаций, является многообразие типов ресурсов для использования в процессе исследований и многообразие контекстов, в которых такие ресурсы могут рассматриваться. Часто такие ресурсы реализуются как унаследованные приложения. Перспективные подходы моделирования в e-science основаны на фундаментальной стратегии интеграции, при которой тот или иной фрагмент информации извлекается из некоторого исследовательского центра и помещается в иной контекст — контекст конкрет-

ной задачи. Формирование и реализация нужного контекста задачи и моделирование интегрируемой информации в этом контексте являются основными проблемами определения посредников.

Следующие соображения будут затронуты в ходе этого обсуждения. Различные приложения должны поддерживаться ВО, каждое из них работает в собственном, конкретном контексте (возможно, пересекающемся с контекстами других приложений), который должен быть определен семантически. Неоднородные информационные ресурсы различных видов (ресурсы данных, сервисные ресурсы, процессные ресурсы, онтологические ресурсы), релевантные ВО, должны использоваться в контексте конкретного приложения. Многие такие ресурсы автономны и эволюционируют во времени. Множество ресурсов, релевантных конкретному классу задач, может изменяться весьма быстро. Технологии, применяемые к релевантным ресурсам, также быстро эволюционируют. Поэтому доказательно правильная идентификация ресурсов, релевантных задаче, достижение семантической интеграции различных видов ресурсов в контексте конкретного приложения, достижение стабильности спецификации проблемной области в быстро развивающемся мире разнообразных ресурсов являются трудно-разрешимыми задачами. Требуется создание новых методов и средств разработки приложений ВО над множествами распределенных неоднородных ресурсов.

Различаются два принципиально различных подхода к проблеме интегрированного представления описания предметной области задачи по отношению к множеству релевантных задаче информационных ресурсов: (1) двигаясь от ресурсов к задачам (схема посредника образуется как интегрированная схема множества ресурсов независимо от приложения) и (2) двигаясь от приложения к ресурсам (описание предметной области приложения образуется независимо от ресурсов (в терминах понятий, структур данных, функций, процессов), а затем релевантные приложению ресурсы отображаются в это описание). Первый подход, *движимый информационными ресурсами*, является немасштабируемым по отношению к числу ресурсов, не дает возможности достижения семантической интеграции ресурсов в контексте конкретного приложения, не ведет к доказательной идентификации релевантных приложению ресурсов, не способствует повышению стабильности спецификации посредника в процессе эволюции ресурсов, релевантных приложению. Эти недостатки являются характерными для подхода, при котором глобальная схема является взглядом (Global as View — GAV) [14, 15]. Схема GAV может служить в качестве базовой тех-

ники подхода, движимого информационными ресурсами.

Другой подход (*движимый приложениями*) предполагает создание предметного посредника, который поддерживает взаимодействие между приложением и ресурсом на основе определения прикладной области (определения посредника). Второй подход имеет очевидные преимущества по отношению к подходу, движимому информационными ресурсами. Процесс регистрации неоднородных информационных ресурсов в предметном посреднике в подходе, движимом приложениями, основан на технике GLAV, комбинирующей два подхода: LAV (Local as View) — когда локальная схема ресурса является взглядом над схемой посредника — и GAV. Согласно LAV [15], схемы регистрируемых ресурсов рассматриваются как материализованные взгляды над виртуальными классами посредника. В этом случае GAV взгляды служат для разрешения различных конфликтов между спецификациями ресурсов и посредника и обеспечивают правила трансформации результатов запроса от ресурса в представление в посреднике. Подобная техника регистрации обеспечивает стабильность спецификации приложения ВО при изменении конкретных информационных ресурсов и их фактического присутствия (удаление ресурса, добавление новых ресурсов и пр.), а также масштабируемость посредников по отношению к числу ресурсов, регистрируемых в них. В версии LAV, именуемой GLAV [16], голова определения правила взгляда LAV может содержать произвольный запрос над схемой ресурса и тем самым выразить случай, когда схема ресурса используется для определения составных частей глобальной схемы (GAV).

Основными принципами разработки посредников ВО, движимых приложениями, над множеством неоднородных информационных ресурсов, являются следующие:

- (1) независимость спецификации приложения (посредника) от существующих информационных ресурсов;
- (2) определение предметного посредника как результата консолидации усилий соответствующего сообщества;
- (3) применение семантических канонических определений в спецификации посредника;
- (4) независимость интерфейсов пользователя от ресурсов, регистрируемых в посреднике; пользователи должны только знать определение предметной области (определение посредника);
- (5) независимость публикации вновь разработанных информационных ресурсов от посредни-

ков (как результат такой публикации могут быть инициированы действия по регистрации ресурсов в соответствующем посреднике);

- (6) трехстадийная идентификация информационных ресурсов, релевантных посреднику, обеспечивающая (а) релевантность ресурсов нефункциональным требованиям, выражаемых метаданными; (б) онтологическую релевантность ресурсов; (в) структурную и поведенческую релевантность ресурсов;
- (7) семантическая интеграция релевантных неоднородных информационных ресурсов в канонической спецификации посредника;
- (8) интегрированный доступ к информационным ресурсам, зарегистрированным в посреднике, на основе канонической модели и системы переписывания запросов;
- (9) рекурсивная структура посредника: каждый посредник может быть зарегистрирован как новый информационный ресурс. Такая способность посредника обеспечивает возможность интеграции различных приложений, что особенно важно для разработки виртуальных организаций.

Принципы (1)–(8) определяют преимущества разработки ВО на основе идеи посредников, применения подход, движимый приложениями, по сравнению с подходом, движимым информационными ресурсами.

Настоящая статья основана, главным образом, на подходе, движимом приложениями.

### 3 Архитектура исполнительных механизмов слоя предметных посредников

Основные решения реализации исполнительных средств слоя посредников заключаются в следующем:

- предметный посредник представляет собой Веб-сервис;
- адаптеры неоднородных ресурсов, зарегистрированных в посреднике, представляют собой Веб-сервисы;
- посредники и адаптеры регистрируются в реестрах;

- один адаптер может работать с несколькими посредниками и с несколькими ресурсами;
- адаптеры могут содержать собственную систему управления базами данных (СУБД) для выполнения сложных планов запросов;
- связь между посредником и адаптерами обеспечивается посредством SOAP<sup>1</sup> протокола;
- база метаинформации посредников и вычисление остаточных запросов реализуются на СУБД Oracle;
- для задания запросов к посреднику пользователь может использовать либо Веб-портал, либо специальный клиент.

На рис. 1 показана общая архитектура исполнительных средств слоя посредников. Она содержит следующие компоненты.

**Application Client.** Клиентская программа, предоставляющая пользователю графический интерфейс для работы с посредником, включающий:

- графические дружественные пользователю средства, обеспечивающие поддержку применения канонического языка запросов на уровне посредников;
- поддержку запросов к посреднику на языке Syfs<sup>2</sup> [8];
- услуги по удобному представлению выводимой информации, включая сегментацию информационных объектов, их агрегирование и слияние.

**Portal.** Предоставляет доступ к посреднику через Интернет. По набору функций эквивалентен обычному клиенту.

**Run-time environment.** Исполнительная среда слоя посредников обеспечивает:

- обработку программ (запросов) пользователя;
- переписывание программ в запросы над зарегистрированными в посреднике ресурсами, отправку переписанных запросов на выполнение соответствующим адаптерам;
- получение результатов запросов от адаптеров;
- выполнение остаточного запроса и отправку результата запросов клиенту.

Исполнительная среда включает следующие компоненты:

<sup>1</sup>SOAP — Simple Object Access Protocol — протокол обмена структурированными сообщениями в распределенной вычислительной среде.

<sup>2</sup>Syfs — подмножество языка формул в языке СИНТЕЗ.

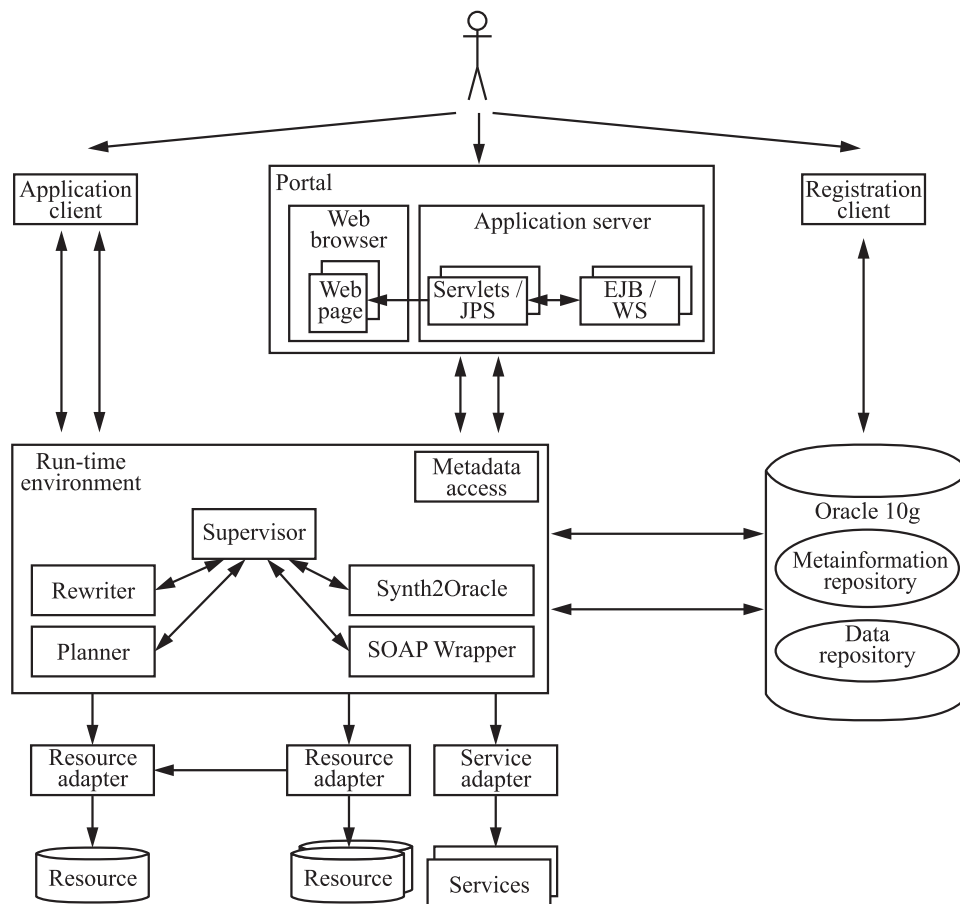


Рис. 1 Архитектура исполнительных средств слоя посредников

**Supervisor** является связующим компонентом посредника и определяет его интерфейс. Supervisor обрабатывает запросы пользователя и для их выполнения взаимодействует с другими компонентами посредника. Фактически код данного компонента реализует взаимодействие всех компонентов в посреднике.

**Rewriter** реализует функцию переписывания запроса пользователя на языке Syfs над схемой посредника в запрос над зарегистрированными ресурсами, который выражается на языке Asyfs<sup>1</sup> и передается компоненту Planner для дальнейшей обработки.

**Planner** реализует функцию планирования запроса над ресурсами. В результате планирования строится распределенный запрос, состоящий из удаленных запросов к отдельным ресурсам. Оптимизированная последовательность выполнения удаленных запросов определяет план, имеющий вид «дерева выполнения».

**SoapWrapper** инкапсулирует детали физической реализации сервисов, которые используют компонент Supervisor и адаптеры. Таким образом, адаптеры могут быть реализованы как Веб-сервисы либо как классы локальных программ.

**Synth2Oracle** реализует поддержку канонической модели данных посредника над базой Oracle в период исполнения. Этот компонент позволяет загружать и выгружать данные, которые в канонической модели представляются как набор классов, преобразованный в набор таблиц базы Oracle. Данные загружаются и выгружаются в формате VOClass<sup>2</sup>. Компонент Synth2Oracle реализует также поддержку выполнения остаточных запросов на языке Asyfs, транслируя их в язык SQL.

**Adapter.** Адаптеры связывают посредник с конкретными ресурсами. В адаптерах происходит преобразование запросов на языке посредника (Asyfs) в язык конкретного ресурса и, наоборот, преобразо-

<sup>1</sup>Asyfs — алгебраический вариант языка Syfs.

<sup>2</sup>VOClass — аналог представления таблиц в стандарте IVOA (в соответствии с XML Schema) для классов языка СИНТЕЗ.

вание результата запроса, полученного в формате, выдаваемом ресурсом, в формат посредника. Дополнительно, если это позволяет ресурс, адаптер реализует возможность загрузки в базу ресурса промежуточных классов периода выполнения запросов, которые используются в выполняемом адаптером запросе.

**Metainformation and Data repository.** Репозиторий состоит из двух частей: репозиторий метаданных и репозиторий данных. Репозиторий метаданных используется для хранения метаинформации посредника (такой как схема посредника, схемы зарегистрированных в посреднике ресурсов, правила, задающие соответствия между классами ресурсов и классами схемы посредника посредством GLAV), используемой при преобразовании запросов. Репозиторий данных используется для хранения промежуточных результатов запросов, возвращаемых ресурсами, и для выполнения остаточных запросов над этими данными.

## 4 Методы переписывания запросов и планирования в предметных посредниках

### 4.1 Основные характеристики ядра канонической модели

В настоящем проекте в качестве ядра канонической модели посредника выбран язык СИНТЕЗ [8], который является гибридной слабоструктурированной/объектной моделью [17]. В этом разделе приводятся основные особенности языка, необходимые для понимания примеров, демонстрирующих идеи переписывания запросов.

Рассматриваемое композиционное исчисление спецификаций [11] не зависит от какой-нибудь определенной информационной модели. Каноническая модель [8] предоставляет поддержку широкого диапазона данных — от слабоструктурированных до строго типизированных данных. Типизированные данные являются экземплярами *абстрактных типов данных* (АТД). Спецификация АТД включает имена и типы параметров операций, определяющих поведение типа. Операции определяются также предикативными спецификациями, описывающими смешанные пред- и постусловия. Абстрактные типы данных могут быть объектными и неobjектными. Объектный тип является подтипом неobjектного АТД с дополнительной операцией *self* на своем интерфейсе, предоставляющей уникальные идентификаторы объектов. В данной статье

используются только типизированные возможности языка СИНТЕЗ. Обсуждение переписывания запросов со слабоструктурированными (фреймовыми) данными планируется в будущих работах. *Множества* в языке (наряду с мультимножествами, последовательностями) рассматриваются в качестве общего механизма группирования значений АТД. Класс рассматривается как подтип типа множества. Поэтому эти, в общем, различные конструкции могут использоваться единым образом: класс может использоваться везде, где может использоваться множество. Например, для формул языка запросов исходные и результирующие данные представляются множествами значений АТД (т. е. коллекциями) или множествами объектов (т. е. классами).

#### 4.1.1 Операции над спецификациями типов

Семантика операций над классами в канонической модели объясняется в терминах композиционного исчисления спецификаций [11]. Действия в исчислении включают декомпозицию спецификаций типов в непротиворечивые фрагменты (редукты), выделение общих фрагментов, композицию выделенных фрагментов в более сложные спецификации соответствующих результирующих типов SPJ (select, project, join) операций. Исчисление использует следующие концепции и операции.

Сигнатура  $\Sigma_T$  спецификации типа  $T = \langle V_T, O_T, I_T \rangle$  включает множество операций  $O_T$ , задающих типы аргументов и результатов операций, и множество предикатов инвариантов типа  $I_T$ . Конъюнкция всех инвариантов из  $I_T$  формирует инвариант типа. Экстенционал  $V_T$  типа  $T$  (носитель типа) есть множество допустимых экземпляров типа.

**Определение 1.** *Редукт типа.* Сигнатура редукта  $R_T$  типа  $T$  определяется как подсигнатура  $\Sigma'_T$  сигнатуры типа  $\Sigma_T$ , которая включает носитель  $V_T$ , множество символов операций  $O'_T \subseteq O_T$ , множество символов инвариантов  $I'_T \subseteq I_T$ .

Это определение легко можно расширить с уровня сигнатур до уровня спецификаций таким образом, что тип редукта  $R_T$  может рассматриваться как подспецификация (с сигнатурой  $\Sigma_T$ ) спецификации типа  $T$ . Спецификация  $R_T$  должна формироваться так, что  $R_T$  становится супертипом типа  $T$ . Предполагается, что только состояния, допустимые для типа, остаются допустимыми для редукта этого типа (никакие другие состояния не являются допустимыми). Таким образом, носитель редукта предполагается равным носителю типа.

**Определение 2.** *Наиболее общий редукт  $R(T_1, T_2)$*  для типов  $T_1$  и  $T_2$  — это редукт  $R_{T_1}$  типа  $T_1$  такой,

что существует редукт  $R_{T_2}$  типа  $T_2$  такой, что  $R_{T_2}$  уточняет  $R_{T_1}$ , и не существует другого редукта  $R'_{T_1}$  такого, что  $R(T_1, T_2)$  является редуктом  $R'_{T_1}$ ,  $R'_{T_1}$  не равен  $R(T_1, T_2)$ , и существует редукт  $R'_{T_2}$  типа  $T_2$ , который уточняет  $R'_{T_1}$ .

Тип  $C$  является *уточнением* типа  $R$  тогда и только тогда, когда:

- существует биективное отображение  
 $Ops: O_R \rightarrow O_C$ ;
- существует всюду определенная функция абстракции  $Abs: V_C \rightarrow V_R$ ;
- для всех  $\nu \in V_C$ :  $Inv_C(\nu)$  влечет  $Inv_R(Abs(\nu))$ ;
- для всех  $o \in O_R$ ,  $o$  уточняется  $Ops(o)$ .

Операция  $o_1$  уточняет  $o_2$  тогда и только тогда, когда предусловие операции  $pre(o_2)$  влечет предусловие  $pre(o_1)$  и постусловие  $post(o_1)$  влечет постусловие  $post(o_2)$ . Для сигнатур операций уточнение означает, что сигнатуры совпадают с точностью до следующего отношения типов параметров: типы входных параметров операции  $o_1$  являются супертипами типов соответствующих входных параметров операции  $o_2$  и типы выходных параметров операции  $o_1$  являются подтипами типов соответствующих выходных параметров операции  $o_2$ .

Основываясь на понятиях редукта и уточнения типов, можно установить меру общей информации между типами в решетке типов  $T$ . Отношение *подтипа* определяется подобно уточнению, только  $Ops$  становится инъективным отображением.

**Определение 3.** *Операция пересечения типов.* В общем случае тип  $T$  — результат операции  $T_1 \& T_2$  — включает атрибуты, общие для типов  $T_1$  и  $T_2$ . Общие элементы типов определяются с помощью наиболее общих редуктов  $R(T_1, T_2)$  и  $R(T_2, T_1)$ . Формально  $O_{T_1 \& T_2}$  определяется следующим образом:

$$O_{T_1 \& T_2} = O_{R(T_1, T_2)} \cup O_{R(T_2, T_1)}.$$

Инвариант типа  $T$  определяется как дизъюнкция инвариантов операндов  $Inv_{T_1} \vee Inv_{T_2}$ .

Можно заметить, что если  $T_2(T_1)$  является подтипом типа  $T_1(T_2)$ , то  $T_1(T_2)$  является результатом пересечения типов. Тип  $T$  помещается в решетку типов как супертип типов операндов этой операции.

**Определение 4.** *Операция соединения типов.* В общем случае тип  $T$  — результат операции  $T_1 | T_2$  — включает объединение спецификаций типов  $T_1$  и  $T_2$ .

Общие элементы спецификаций типов  $T_1$  и  $T_2$  включаются в объединение (результатирующий тип)

только один раз. Общие элементы типов определяются с помощью наиболее общих редуктов  $R(T_1, T_2)$  и  $R(T_2, T_1)$ . Формально  $O_{T_1 | T_2}$  определяется как

$$O_{T_1 | T_2} = (O_{T_1} \setminus O_{R(T_1, T_2)}) \cup (O_{T_2} \setminus O_{R(T_2, T_1)}).$$

Инвариант типа  $T$  определяется как конъюнкция инвариантов операндов  $Inv_{T_1} \wedge Inv_{T_2}$ .

Можно заметить, что если  $T_2(T_1)$  является подтипом типа  $T_1(T_2)$ , то  $T_2(T_1)$  является результатом операции соединения типов. Тип  $T$  помещается в решетку типов как подтип типов операндов операции.

Операции композиционного исчисления формируют решетку типов [11] на основе отношения подтипа (как отношения частичного порядка). В языке СИНТЕЗ операции композиции типов используются для формирования типовых выражений, которые в языке используются для определения типа. Типовые выражения включают следующие конструкции:

- указатель типа ( $T$ );
- конструкция редукта ( $R[e_1, \dots, e_n]$ , где  $R$  — указатель типа,  $e_i$  — элементы редукта);
- выражения ( $T$ ),  $T|S$  и  $T \& S$ , где  $T$  и  $S$  — указатели типов.

Конструкция редукта имеет следующий вид:  $R[e_1, \dots, e_n]$ , где  $R$  — исходный АТД,  $e_i$  — элементы редукта, каждый из которых определяет атрибут образуемого АТД. Редукт определяет также функцию преобразования экземпляров исходного АТД в экземпляры редукта. В общем случае элемент редукта имеет следующий вид:

$$a/T : t,$$

где  $a$  — идентификатор атрибута,  $T$  — тип,  $t$  — атрибутный путь (вида  $a_1.a_2. \dots .a_k$ ). Доступны следующие сокращенные формы элементов редукта:

- $a : t$  — сокращение для  $a/T : t$ , где  $T$  — тип термина  $t$  (т.е. для пути  $a_1. \dots .a_k$  его типом является тип атрибута  $a_k$ );
- $a/T$  и  $a$  — сокращения для  $a/T : a$  и  $a : a$  соответственно.

Редукт  $R[a_1/T_1 : t_1, \dots, a_n/T_n : t_n]$  образует АТД  $T_R$ , который содержит атрибуты  $a_1, \dots, a_n$ , типы которых соответственно:  $T_1, \dots, T_n$ . Функция преобразования экземпляров АТД  $R$  в экземпляры редукта типа определяется следующим образом:

$$P(x) = \{y \in T_R | y.a_1 = x.t_1, \dots, y.a_n = x.t_n\}.$$

#### 4.1.2 Подмножество языка запросов СИНТЕЗ

В статье рассматривается ограниченное подмножество языка запросов СИНТЕЗ, ориентированное на представление объединения конъюнктивных запросов с SPJ семантикой.

Для определения формулы запроса используется разновидность типизированного (многоуровневого) языка логики предикатов первого порядка. Предикаты в формуле соответствуют коллекциям (таким как множества и мультимножества необъектных экземпляров), классам, которые рассматриваются как множества объектных значений, и функциям. Предикат класса (или предикат коллекции) всегда унарный предикат. Предикат функции  $F$  синтаксически имеет вид  $n$ -арного предиката  $F(X, Y)$ , где  $X$  — последовательность термов, соответствующих входным параметрам функции,  $Y$  — последовательность типизированных результирующих атрибутов (конструкция вида  $y/Y$ , где  $Y$  определяет тип атрибута  $y$ ), соответствующих выходным параметрам функции (результатирующие атрибуты принимают значения соответствующих выходных параметров). В качестве термов могут выступать: переменные, константы, атрибуты, функции (включая арифметические функции), конструкция взятия атрибута  $t.a$ , где  $t$  — терм и  $a$  — атрибут из АД, типизирующего  $t$ . Частным случаем функции является операция (метод), где первым входным параметром передается экземпляр АД, в котором определен метод.

**Определение 5.** Конъюнктивным запросом СИНТЕЗа (*Synthesis Conjunctive Query, SCQ*) является правило следующего вида

$$q(\nu/T_\nu) : \neg C_1(\nu_1/T_{\nu_1}) \& \dots \& C_n(\nu_n/T_{\nu_n}) \&$$

$$F_1(X_1, Y_1) \& \dots \& F_m(X_m, Y_m) \& B,$$

где  $q(\nu/T_\nu)$ ,  $C_1(\nu_1/T_{\nu_1}), \dots, C_n(\nu_n/T_{\nu_n})$  — предикаты коллекций (классов);  $F_1(X_1, Y_1), \dots, F_m(X_m, Y_m)$  — предикаты функций,  $B$  — ограничение, представляющее собой конъюнкцию арифметических предикатов. Каждый предикат  $C_i(\nu_i/T_{\nu_i})$  или  $F_j(X_j, Y_j)$  называется *подцелью*. Подцели и  $B$  называются *телом правила*,  $q(\nu/T_\nu)$  называется *головой правила*. Запрос SCQ также называется правилом. Семантика SPJ тела SCQ будет приведена далее.

В языке Syfs допускаются правила, представляющие собой объединение правил с SPJ семантикой. Несколько правил на языке Syfs составляют программу.

Общая схема вычисления результирующей коллекции тела SCQ имеет следующий смысл:

$$C_1(\nu_1/T_{\nu_1}) \& \dots \& C_n(\nu_n/T_{\nu_n}) \&$$

$$F_1(X_1, Y_1) \& \dots \& F_m(X_m, Y_m) \& B.$$

Сначала выполняется соединение коллекций, которые определяются предикатами классов  $C_i(\nu_i/T_{\nu_i})$ . Предикат класса  $C(\nu/T)$  формирует результирующую коллекцию, которая строится на основе проекции класса  $C$ , где проекция определяется типом  $T$ . В общем случае  $T$  является редуктом  $R[a_1/T_1 : t_1, \dots, a_n/T_n : t_n]$ , и к экземплярам класса  $C$  применяется определяемая редуктом функция преобразования. Типом результирующей коллекции является АД  $R_{C(\nu/T)}$  с атрибутами  $\nu, a_1, \dots, a_n$ , типы которых есть, соответственно:  $T, T_1, \dots, T_n$ .

Множество экземпляров результирующей коллекции является следующим:

$$\{y \in R_{C(\nu/T)} \mid \exists x \in C :$$

$$y.\nu = x, y.a_1 = x.t_1, \dots, y.a_n = x.t_n\}.$$

Результатом соединения двух коллекций  $C_1$  и  $C_2$  с типами экземпляров  $T_1$  и  $T_2$ , соответственно, является коллекция  $C_1 \& C_2$ , тип экземпляров которой равен  $T_1|T_2$ , и множество экземпляров определяется следующей формулой:

$$C_1 \& C_2 = \{x \in T_1|T_2 \mid \exists y_1 \in C_1, y_2 \in C_2 :$$

$$x = y_1, x = y_2\}.$$

Затем по очереди вычисляются функции  $F_j$ , и к результирующей коллекции присоединяются (are appended) результирующие атрибуты.

Вычисление функции выполняется для каждого экземпляра исходной коллекции. Значения атрибутов текущего экземпляра могут использоваться при вычислении значений термов, аргументов для входных параметров функции, и текущий экземпляр расширяется результирующими атрибутами предиката функции.

После этого выбираются только те экземпляры результирующей коллекции, для которых выполняется ограничение  $B$ . Результирующий класс  $q$  формируется как проекция результирующей коллекции тела, которую описывает редукт  $T_\nu$  в голове правила.

Два запроса SCQs с головами  $q_1(\nu_1/T_1)$  и  $q_2(\nu_2/T_2)$  называются *сравнимыми*, если тип  $T_1$  является подтипом типа  $T_2$ . Для двух сравнимых запросов  $q_1$  и  $q_2$  запрос  $q_1$  *содержится* в запросе  $q_2$  (обозначается  $q_1 \subseteq q_2$ ), если для любого экземпляра базы данных все экземпляры результирующего класса  $q_1$  являются экземплярами результирующего класса  $q_2$ , т. е. результирующий класс  $q_1$  является подклассом результирующего класса  $q_2$ .

## 4.2 Регистрация ресурсов в посреднике и инверсные правила

В предметном посреднике процесс регистрации неоднородных информационных ресурсов основан на подходе LAV/GLAV. Подход GLAV позволяет голове взгляда LAV содержать элемент, выраженный запросом над схемой ресурса, и таким образом выражать случай, когда схемы ресурса используются для определения конструкций глобальной схемы (GAV). Подход GAV позволяет задавать функции разрешения различных видов конфликтов между спецификациями ресурса и посредника. Он также позволяет задавать правила преобразования результатов запросов от ресурса в посредник.

Таким образом, процесс регистрации заключается в конструировании множества взглядов GLAV, которые имеют следующий вид:

- правило LAV

$$V(\nu/T_\nu) :- Cm_1(\nu_1/T_{\nu 1}) \& \dots \& Cm_n(\nu_n/T_{\nu n}) \& Fm_1(X_1, Y_1) \& \dots \& Fm_m(X_m, Y_m) \& B_m ;$$

- правило GAV

$$V(\nu/T_\nu) :- Cs_1(\nu_1/T_{\nu 1}) \& \dots \& Cs_n(\nu_n/T_{\nu n}) \& Fs_1(X_1, Y_1) \& \dots \& Fs_m(X_m, Y_m) \& B_s ,$$

где  $Cm_i$  — предикат класса посредника;  $Fm_i$  — предикат функции посредника;  $Cs_i$  — предикат класса ресурса;  $Fs_i$  — предикат функции ресурса или функция разрешения конфликтов;  $B_m$  и  $B_s$  — ограничения.

Формально переписывание запросов основано на понятии инверсного правила (ИП). Рассматриваются два вида инверсных правил: инверсное правило класс в класс и инверсное правило функция в функцию. Инверсное правило класс в класс получается из взгляда LAV как пара головы взгляда и подцели взгляда. Инверсное правило класс в класс имеет следующий вид:

$$C(x/T) \leftarrow V(-/R) ,$$

где предикат  $C(x/T)$  — голова инверсного правила и  $V(-/R)$  — тело инверсного правила (символ подчеркивания обозначает анонимную переменную). Для взгляда

$$V(-/R) :- C_1(x_1/T_1) \& \dots \& C_n(x_n/T_n) \& F_1(t_1, y_1) \& \dots \& F_m(t_m, y_m) \& B$$

конструируются следующие инверсные правила класс в класс:

$$C_1(x_1/T_1) \leftarrow V(-/R), \dots, C_n(x_n/T_n) \leftarrow V(-/R) .$$

Инверсные правила определяют, как отдельные подцели класса, выраженные в терминах схемы посредника, могут быть переписаны в подцели, выраженные в терминах взглядов GAV и схем ресурсов. Например, для того чтобы получить переписывание правила, которое содержит вхождения  $C(x/T)$ , можно заменить вхождения  $C(x/T)$  вхождениями  $V(-/R)$ , если при этом существует инверсное правило  $C(x/T) \leftarrow V(-/R)$  и редукты  $T$  и  $R$  задают типы, связанные так, что  $R$  уточняет  $T$ .

В общем случае редукт  $R$  не включает все атрибуты, которые определяются в теле взгляда, и поэтому редукт  $R$  должен быть расширен в процессе сколемизации. Так как в нашей информационной модели атрибуты определяются как соответствующие *get*-функции, мы рассматриваем *get*-функции атрибутов, которые формируют расширение, как сколемовские функции (помечая их символом #). Например, во взгляде

$$\nu(-/R[a, b]) :- p(x/T_p[a, b]) \& q(y/T_q[b, c]) \& f(x, z)$$

редукт в голове не включает атрибуты  $x$ ,  $y$ ,  $c$  и  $z$ , которые определяются в теле взгляда, поэтому редукт расширяется сколемовскими атрибутами  $\#x$ ,  $\#y$ ,  $\#c$  и  $\#z$ :

$$R[a, b, x : \#x, y : \#y, c : \#c, z : \#z] .$$

Затем на основе сколемизированного взгляда строятся два инверсных правила:

$$p(x/T_p[a, b]) \leftarrow \nu(-/R[a, b, x : \#x])$$

и

$$q(y/T_q[b, c]) \leftarrow \nu(-/R[b, y : \#y, c : \#c]) .$$

Инверсные правила функция в функцию задаются непосредственно в процессе регистрации. Инверсное правило функция в функцию является формулой следующего вида:

$$\text{all } x_1/T_1, \dots, x_n/T_n (F(t, y) \leftarrow G(s, y)) ,$$

где  $x_1, \dots, x_n$  — переменные типов  $T_1, \dots, T_n$ ,  $t$  и  $s$  — термы с вхождениями этих переменных, а  $y$  — результирующий атрибут. Инверсное правило функция в функцию выражает отношение уточнения между двумя функциями таким образом, что тело инверсного правила  $G(s, y)$  уточняет голову инверсного правила  $F(t, y)$  как функций над переменными  $x_1, \dots, x_n, y$ . Инверсные правила функция в функцию используются для переписывания подцелей функции аналогично тому, как инверсные правила класс в класс используются для переписывания подцелей класса.

### 4.3 Переписывание запросов

Для поддержки подхода GLAV был разработан алгоритм переписывания запросов [18], состоящий из трех фаз: формальное переписывание и семантический анализ, реализующие подход LAV и третья фаза — разворачивание взглядов GAV. Для нерекурсивных программ запроса, состоящих из нескольких правил, переписывание применяется к каждому правилу в отдельности. Реализация подхода LAV основана на разновидности алгоритма инверсных правил [19], расширенного для типизированного языка запросов [20]. К результатам фазы формального переписывания применяется семантический анализ, который выполняет проверку на выполнимость правил и устраняет вхождения сколемовских функций.

Формальное переписывание применяется к телу правила. Для каждой подцели, которая не является встроенным предикатом, выбирается такое инверсное правило, голова которого унифицируется с этой подцелью. Унификация головы инверсного правила с подцелью также влечет следующее преобразование тела инверсного правила.

Инверсное правило класс в класс

$$C(x/T[x_1/T_1 : t_1, \dots, x_n/T_n : t_n]) \leftarrow \\ \leftarrow V(-/R[x/T : y, x_1/T_1 : y_1, \dots, x_n/T_n : y_n]),$$

где  $t_1, \dots, t_n$  — пути атрибутов, можно унифицировать с подцелью

$$C'(z/T'[z_1/S_1 : t_1.s_1, \dots, z_k/S_k : t_k.s_k]),$$

где  $k \leq n$ ,  $C'$  — подкласс класса  $C'$ ,  $S_i$  — подтип типа  $T_i$ , а  $s_1, \dots, s_2$  — пути атрибутов.

Преобразованное тело инверсного правила строится как следующий предикат класса:

$$V(-/R[z/T : y, z_1/S_1 : y_1.s_1, \dots, z_k/S_k : y_k.s_k]).$$

Инверсное правило функция в функцию

$$\text{all } x_1/T_1, \dots, x_n/T_n (F(t, y) \leftarrow G(s, y))$$

можно унифицировать с подцелью  $F(t', z)$ , если унификация термов  $t$  и  $t'$  может быть установлена со следующим наиболее общим унификатором как подстановкой

$$u = \{t'_1|x_1, \dots, t'_n|x_n, \dots, t_1|z_1, \dots, t_m|z_m\},$$

где переменные  $z_i$  имеют вхождения в терме  $t'$ ,  $t'_i$  — подтермы терма  $t'$ , а  $t_i$  — подтермы терма  $t$ . Преобразованное тело инверсного правила строится как следующая конъюнкция:

$$G(s', z) \& E,$$

где  $E = z_1 = t_1 \& \dots \& z_m = t_m, s'$  получается из  $s$  посредством применения  $u$ , т.е. вхождения переменных  $x_1, \dots, x_n$  заменяются на термы  $t'_1, \dots, t'_n$  соответственно, и вхождения переменных  $z_1, \dots, z_m$  заменяются на термы  $t_1, \dots, t_m$  соответственно. Например, инверсное правило функция в функцию  $\text{all } x/\text{real} (f(45, x, y) \leftarrow g(x, y))$  можно унифицировать с предикатом функции  $f(a, b, c)$  и с преобразованным телом инверсного правила:  $g(b, c) \& a = 45$ .

Для заданного правила

$$Q(-/R) : -C_1(x_1/T_1) \& \dots \& C_n(x_n/T_n) \& \\ F_1(t_1, y_1) \& \dots \& F_m(t_m, y_m) \& B$$

и набора инверсных правил такого, что унификация инверсного правила класс в класс с подцелью  $C_i(x_i/T_i)$  влечет преобразование тела правила  $V_i(-/R_i)$ , а унификация инверсного правила функция в функцию с подцелью  $F_j(t_j, y_j)$  влечет преобразование тела инверсного правила  $G_j(s_j, y_j) \& E_j$ , формируется следующее переписывание:

$$Q(-/R) : -V_1(-/R_1) \& \dots \& V_n(-/R_n) \& G_1(s_1, y_1) \& \dots \\ \dots \& G_m(s_m, y_m) \& E_1 \& \dots \& E_m \& B.$$

Построенное переписывание правила включается в исходное правило и таким образом формирует подзапрос.

### 4.4 Семантический анализ

Полученные в результате формального переписывания правила могут содержать сколемовские функции и невыполнимые ограничения, поэтому необходимо применить дополнительный анализ для того, чтобы образовать правила, не содержащие сколемовские функции, и отбросить невыполнимые правила. В ходе данной фазы отслеживаются атомарные значения всех термов из правила, которые привязываются к отдельным переменным.

Для терма  $t$ , который имеет не скалярный тип, такой как АД с атрибутами  $a_1, \dots, a_n$ , выполняется рекурсивно привязка к переменным термов  $t.a_1, \dots, t.a_n$ . Проекция и соединения транслируются во множество равенств над этими переменными. Предикаты функций транслируются в равенства вида  $f(\nu_1, \dots, \nu_n) = y$ .

Например, для правила

$$r(-/T[a, b]) : -p(-/T[a, b, c]) \& f(c, y)$$

атрибуты предиката головы привязываются к переменным  $\nu_1$  и  $\nu_2$ , атрибуты в предикате класса —

к переменным  $\nu_3, \nu_4, \nu_5$ , атрибут  $y$  — к переменной  $\nu_6$ . Правило транслируется в множество равенств:  $\nu_1 = \nu_3, \nu_2 = \nu_4, f(\nu_5) = \nu_6$ . Для построения конгруэнтных классов термов к множеству равенств применяется разновидность алгоритма конгруэнтного замыкания [21]. Множество конгруэнтных классов предоставляет удобную форму для представления множества равенств с учетом свойства транзитивности равенств.

Например, из множества равенств  $\nu_1 = \nu_3, \nu_2 = \nu_4, f(\nu_5) = \nu_6$  конструируется два конгруэнтных класса  $\nu_1, \nu_3$  и  $\nu_2, \nu_4$ . Анализ правила и взглядов LAV выполняется в терминах этого представления. Конгруэнтные классы, получаемые из взглядов LAV, на которые имеются ссылки в правиле (посредством предикатов классов), представляют равенства, имплицитные взглядами, и поэтому они безопасно могут быть исключены из множества равенств правила. Затем по множеству равенств восстанавливаются правила, которые уже не содержат сколемовские функции. Таким образом, реализуется устранение сколемовских функций в фазе семантического анализа.

Наряду с устранением сколемовских функций выполняется проверка выполнимости правил. Дополнительно собираются все арифметические предикаты, обнаруженные в ограничении правила и в ограничениях, использующихся в правиле LAV взглядов. Арифметические предикаты обрабатываются в терминах привязки к переменным, и следовательно, рассматриваются только переменные, константы и арифметические предикаты. Рассматриваются арифметические предикаты  $<, \leq, \geq, >$ , и для проверки выполнимости конструируется граф неравенств [22]. Также выделяется граф неравенств, который строится из арифметических предикатов, порождаемых взглядами, и используется для устранения арифметических предикатов, которые связаны со сколемовскими функциями.

## 4.5 Разворачивание взглядов GAV

Первые две фазы производят переписанные правила, которые могут ссылаться на взгляды GAV. Для того чтобы получить переписывания правил, выраженные в терминах локальных схем, выполняется разворачивание взглядов GAV. В нашем подходе проблема разворачивания взглядов GAV рассматривается в контексте ограниченного случая, так как в результате процесса регистрации образуются взгляды GAV, которые не имеют общих локальных функций и классов. Поэтому можно выполнять разворачивание каждого взгляда GAV

независимо. Разворачивание взгляда GAV в подцели правила выполняется как унификация головы взгляда GAV с этой подцелью. Унификация влечет преобразование тела взгляда GAV. Дополнительно, в качестве оптимизации, из преобразованного тела взгляда GAV удаляются все предикаты функции, результирующие атрибуты которых не используются. Полученным телом взгляда GAV заменяется вхождение данной подцели. Для того чтобы при подстановке тела взгляда замкнутые переменные и атрибуты (не встречающиеся в голове взгляда GAV) не вступали в конфликт с другими замкнутыми переменными правила, к идентификаторам замкнутых атрибутов и переменных приписывается уникальный суффикс.

Например, рассмотрим следующие два взгляда GAV:

$$r_1(-/[a, b, c]) : -p_1(x/[a, b, c])$$

$$r_2(-/[a, b]) : -p_2(x/[a, c]) \& p_3(y/[b, c])$$

и правило, для которого выполняется разворачивание этих взглядов:

$$r(-/[a]) : -r_1(/[a, c]) \& r_2(/[a, b]).$$

Результат разворачивания взглядов GAV будет следующим:

$$r(-/[a]) : -p_1(x_1/[a, b]) \& p_2(x_2/[a, c_1]) \& p_3(y_1/[b, c_1]).$$

## 4.6 Планирование распределенного запроса

В посреднике запрос переписывается в последовательность запросов, выраженных в терминах ресурсов, зарегистрированных в посреднике. Так как в реальности посредник имеет дело с удаленными неоднородными ресурсами, доступ к которым осуществляется посредством адаптеров к этим ресурсам, возникает необходимость в решении задачи планирования распределенного запроса. В процессе планирования выделяются подзапросы к отдельным ресурсам, а также остаточный запрос, который будет выполняться в посреднике над результатами, полученными от этих локальных ресурсов. В архитектуре посредника поддерживаются ресурсы, которые могут принимать внешние коллекции (т.е. предоставляют интерфейс для загрузки данных во временное хранилище ресурса), и позволяют использовать загруженные коллекции в запросах к этим ресурсам (в примере, который рассматривается в данной статье, в качестве ресурса с такими возможностями выступает ресурс SDSS<sup>1</sup>). Поэтому

<sup>1</sup>SDSS — Sloan Digital Sky Survey — Цифровой обзор неба Слоан.

в процессе планирования необходимо решить, куда передаются данные: от одного ресурса к другому или в посредник. Подробнее этап планирования запроса будет рассмотрен для примера задачи поиска далеких галактик.

## 5 АстроГрид как основа инфраструктуры Российской виртуальной обсерватории

Цель системы АстроГрид — поддержка инфраструктуры для решения научных задач в астрономических ВО. Для решения научных задач АстроГрид предоставляет средства доступа к астрономическим каталогам, цифровым обзорам и архивам изображений, а также к реестрам метаданных, в которых регистрируются ресурсы ВО. Система обеспечивает размещение файлов и хранение промежуточных результатов в процессе решения задач группами пользователей, средства доступа к реестрам метаданных, в которых регистрируются ресурсы ВО, средства создания и исполнения потоков работ, средства взаимодействия между различными внешними приложениями в распределенной среде ВО.

Система АстроГрид построена на основе архитектуры Веб-сервисов. Каждый компонент системы представляет собой Веб-сервис, который реализует определенную функцию. Веб-сервисы могут располагаться как на одном сервере, так и на разных серверах, что позволяет добиться максимальной гибкости и производительности. Далее рассматриваются основные компоненты АстроГрида.

**Registry** (реестр) представляет собой коллекцию метаданных — XML-документов, описывающих ресурсы, которые могут использоваться при решении задач с помощью ВО. Registry реализован на основе стандарта OAI PMH<sup>1</sup> [23], специализированного IVOA для нужд ВО. Поиск в Registry осуществляется по ключевым словам на этапе планирования и выполнения приложений. В качестве ресурсов могут выступать приложения, сервисы, информационные ресурсы, компоненты АстроГрида, а соответствующие им метаданные в Registry содержат сведения о том, где располагаются ресурсы, как они могут быть использованы, кем разработаны, для чего предназначены и т. д. Схемы для метаданных, хранящихся в Registry, а также интерфейсы сервисов определены спецификациями IVOA. Registry поддерживает функцию сбора информации о ресурсах, зарегистрированных в других реестрах ВО (harvesting).

**Community** обеспечивает регистрацию и персональную аутентификацию пользователей. Для управления учетными записями пользователей предоставляется Веб-интерфейс. Учетная запись пользователя позволяет использовать компонент АстроГрида MySpace, который может потребоваться для решения научных задач.

**MySpace** представляет собой виртуальное хранилище данных, к которым могут иметь доступ все сервисы системы АстроГрид. Компонент MySpace используется для хранения и передачи файлов между сервисами в рамках одной задачи и между задачами, решаемыми при помощи ВО.

**Common Execution Architecture** (CEA — Общая исполнительная архитектура) определяет способ оформления приложения в сервис АстроГрида. Приложения могут быть трех видов: Веб-сервисы, приложения командной строки unix, приложения Java. Приложение, оформленное как CEA, обладает рядом преимуществ:

- вызывается как из потоков работ, так и как настольное приложение, при этом не требуется изменения кода клиентских программ;
- регистрируется и обнаруживается в реестрах АстроГрида;
- приложение может считывать и записывать данные в MySpace;
- может выполняться в асинхронном режиме, что важно в случае продолжительных задач;
- приложение может быть защищено средствами безопасности системы АстроГрид, если требуется ограничить доступ к нему;
- приложение может использоваться без написания дополнительного кода: CEA предоставляет Веб-сервис, в который встраивается приложение.

**DataSet Access** (DSA) реализует подключение базы данных к системе АстроГрид. Компонент DSA предоставляет два интерфейса доступа: CEA-приложение, которое выполняет запросы к базе данных; Cone Search, который позволяет производить поиск в некоторой заданной области неба. В настоящее время для использования баз данных в DSA поддерживаются следующие СУБД: PostgreSQL, MySQL, MS SQL Server, HSQLDB.

**Astro Runtime** обеспечивает доступ к сервисам и ресурсам ВО посредством простого программного интерфейса, функции которого могут быть вызваны из любых приложений. Библиотеки

<sup>1</sup>OAI-PMH — Open Archives Initiative Protocol for Metadata Harvesting — архивный открытый протокол сбора метаданных.

для работы с Astro Runtime разработаны для ряда языков. Таким образом, обеспечивается возможность создания новых приложений, взаимодействующих с ВО.

**PLASTIC** (*PLatform for Astronomical Tool Inter Co-operation*) — это протокол, обеспечивающий взаимодействие между средствами АстроГрида и внешними программными инструментами в среде ВО. Посредством PLASTIC приложения могут обмениваться данными или управлять друг другом, например одно приложение может попросить другое загрузить изображение некоторой области неба и работать с полученным результатом запроса. Хотя подобные операции достаточно просты, но они открывают дополнительные возможности для взаимодействия между приложениями в среде ВО.

**Workbench** — это клиент системы АстроГрид, который позволяет работать с астрономическими ресурсами. Workbench включает:

- Task Launcher — позволяет запускать удаленные приложения (СЕА), реализующие запросы к астрономическим каталогам данных, и/или их обработку, например CrossMatch;
- Workflow Builder — позволяет конструировать потоки работ, состоящие из шагов, которые могут выполняться последовательно, параллельно или в цикле;
- MySpace Browser — позволяет манипулировать пользовательскими данными, находящимися в MySpace;
- LookOut — позволяет следить за выполняющимися задачами (или потоками работ), предоставляет возможности отмены, паузы или досрочного завершения задачи;
- AstroScore — позволяет находить все ресурсы, связанные с некоторым астрономическим объектом;
- HelioScore — обеспечивает доступ к данным по Солнцу.

Согласно аванпроекту информационной инфраструктуры РВО (РВОИИ) [1], основными принципами построения инфраструктуры являются следующие:

- архитектура реализуется как иерархия взаимодействующих между собой Веб-сервисов (воплощаемых в грид-архитектуре по мере того, как такие архитектуры будут созревать и будут окончательно стандартизированы мировым сообществом);
- приближение обработки к данным, мотивацией для чего является большой объем накопленных

данных и ресурсно-интенсивный характер прикладных задач ВО. Во многих случаях данные не существуют как удаленные, их еще нет во время формулирования задач: они могут быть извлечены из баз данных по запросам, формулируемым при реализации задач;

- создание программ для большинства астрономических проектов является основной проблемой при разработке и эволюции ВО, поэтому модульность архитектуры, способствующей повторному использованию и композиции программ является основополагающим принципом построения инфраструктуры РВО;
- учитывая существование тысяч астрономических архивов и каталогов в мире, формирование глобальной схемы базы данных для ВО является не масштабируемым подходом. Поэтому основным принципом инфраструктуры ВО является следование идее предметных посредников, обеспечивающих взаимодействие между исследователями и релевантными ресурсами информации на основе спецификации предметной области для соответствующего класса задач.

Анализ показывает, что использование АстроГрида в качестве основы РВОИИ позволяет реализовать перечисленные принципы, а именно: поддержку грид-интероперабельных сервисов, модульность архитектуры, возможность повторного использования и композиции сервисов, создание многослойной архитектуры. Компоненты АстроГрида являются непосредственно применимыми в качестве ядра архитектуры РВОИИ:

- Registry — для поиска ресурсов в реестрах на основе метаданных;
- MySpace — для управления виртуальными областями хранения данных совместного пользования;
- Workbench — для интерфейса пользователя ВО и решения задач на основе потоков работ;
- Community — для администрирования и управления доступом к данным ВО;
- СЕА — в качестве средства поддержки сети интероперабельных сервисов;
- DSA — в качестве средств хранения данных, размещаемых на требуемом уровне реализации системы (задачи).

## 6 Гибридная архитектура АстроГрида и исполнительных средств поддержки слоя предметных посредников

При разработке гибридной архитектуры во главу угла ставилась задача обеспечения интероперабельности компонентов слоя предметных посредников и АстроГрида. В качестве необходимых были выделены следующие возможности гибридной архитектуры:

- регистрация и поиск посредников в реестрах АстроГрида;
- просмотр пользователем схем посредников и другой информации о посредниках;
- задание запросов к посредникам как на языке Syfs, так и на языке ADQL<sup>1</sup> [24];
- использование программ посредников как шагов потоков работ в системе АстроГрид;
- просмотр результатов программ посредников в среде АстроГрида.

В результате анализа архитектур средств поддержки слоя посредников и АстроГрида выработаны следующие основные решения гибридной архитектуры:

- посредник реализуется как СЕА-приложение и регистрируется в реестре АстроГрида при помощи сервера-адаптера приложений. Метаданные, которыми характеризуется регистрируемый посредник, расширяются схемой посредника;
- на интерфейсе посредника имеются методы задания запросов на Syfs и на ADQL;
- результаты запросов к посреднику представляются в формате VOClass, для которого формат VOTable [2] является подмножеством, и сохраняются в MySpace;
- для подключения к посредникам DSA ресурсов, уже зарегистрированных в системе АстроГрид, используется адаптер к DSA ресурсам;
- для осуществления поиска по метаданным ресурсов или приложений в реестрах системы АстроГрид используется адаптер к реестрам АстроГрида, реестр регистрируется в посреднике как обычный ресурс;
- в качестве клиентского интерфейса и приложения используется Workbench;
- для поддержки астрономических онтологий, принятых в системе АстроГрид, разрабатываются средства онтологической интеграции

на основе UCD (Unified Content Description) [2] — унифицированных дескрипторов контента, широко используемых в астрономии.

В данном разделе рассматриваются, в основном, те компоненты гибридной архитектуры, которыми дополняется система АстроГрид, а также исполнительные средства слоя посредников (рис. 2). Компоненты самой системы АстроГрид и компоненты исполнительных средств слоя посредников описаны ранее.

Посредник может регистрироваться в реестрах АстроГрида в двух вариантах: как СЕА-приложение или как DSA-компонент. Стоит отметить, что один и тот же посредник может быть зарегистрирован одновременно как СЕА и как DSA.

Посредник, зарегистрированный как СЕА-приложение, выполняет запрос на языке Syfs над объектной схемой посредника и возвращает результат в формате VOClass. Для реализации этой возможности разработан компонент *M-CEA*, который инкапсулирует конкретный посредник и может быть зарегистрирован в реестре АстроГрида как СЕА приложение.

Посредник, зарегистрированный как DSA, выполняет запрос на языке ADQL над уплощенной схемой посредника и возвращает результат в формате VOClass. Поскольку схема посредника объектная, а ADQL — язык запросов к реляционной схеме, необходим специальный компонент посредника — *преобразователь схем*, который по объектной схеме посредника строит ее уплощение (реляционное представление). Запрос на ADQL формулируется в терминах плоской схемы, а затем компонентом *ADQL2Syfs* транслируется в запрос на языке Syfs к исходной объектной схеме посредника. *Преобразователь схем* и *ADQL2Syfs* используются разработанным компонентом *M-DSA*, который инкапсулирует конкретный посредник и может быть зарегистрирован в реестре АстроГрида как DSA-приложение. Оформление посредника как DSA — важная особенность, так как предоставляет пользователям АстроГрида возможность видеть посредник в привычном виде как DSA-источник.

В любом случае посредник оформляется в качестве СЕА-приложения, которое может быть использовано как шаг потока работ или как самостоятельное приложение. Таким образом, обеспечивается возможность использования посредника в системе АстроГрид, а также возможность удобного просмотра схемы посредника прямо из клиента Workbench системы АстроГрид. Метаданные для регистрации посредника генерируются на основании его схемы, определенной на языке СИНТЕЗ.

<sup>1</sup>ADQL — Astronomical Data Query Language — астрономический язык запросов, поддерживаемый IVOA. Основан на SQL 92.

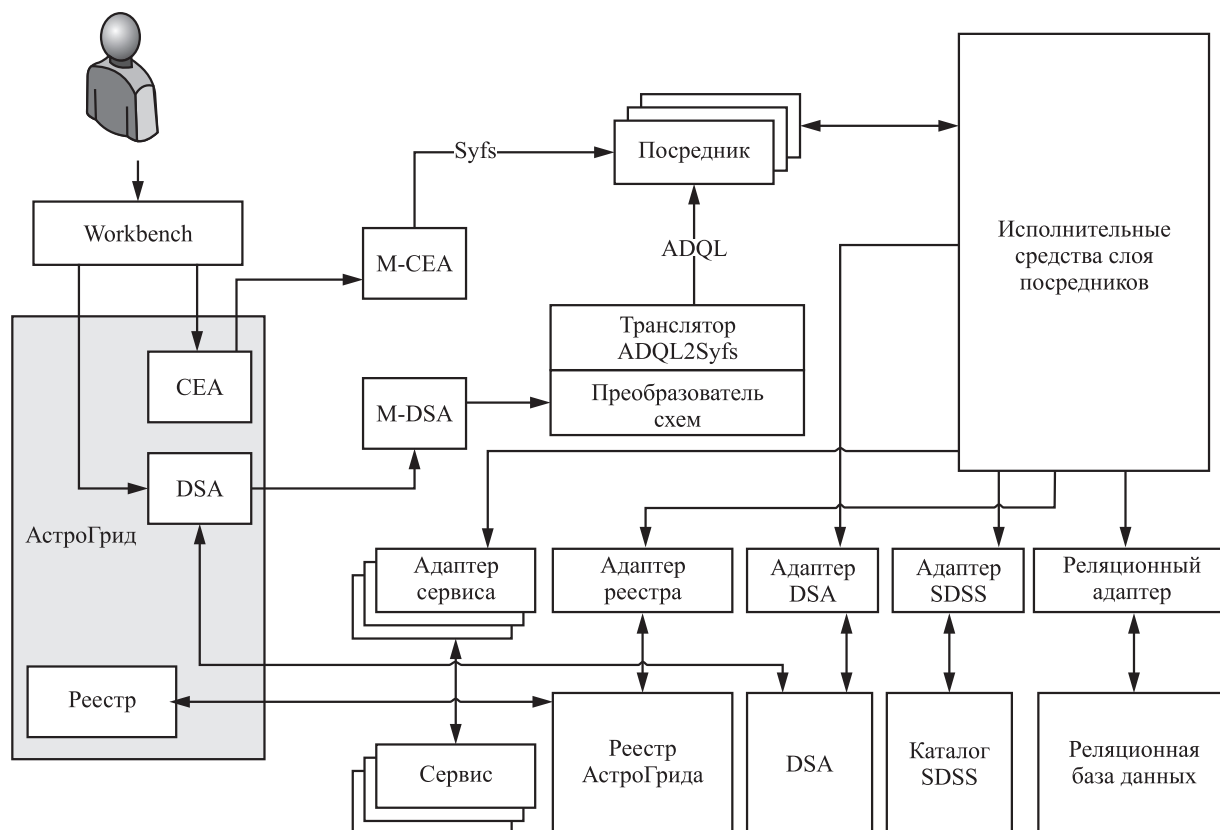


Рис. 2 Схема гибридной архитектуры АстроГрида и исполнительных средств слоя посредников

Адаптеры обеспечивают преобразование запросов на языке планов посредника (Asyfs) в запросы на языке ресурса, получение результата запроса от другого ресурса, а также преобразование объектов результирующего множества в объекты схемы посредника.

В действующем прототипе разработаны адаптер реляционных баз данных (*Relational Wrapper*), адаптер к астрономическому каталогу данных SDSS (*SDSS Wrapper*) с поддержкой возможности выполнения XMatch на сервере SDSS, адаптер DSA ресурсов (*DSA Wrapper*) и адаптер реестров АстроГрида (*Registry Wrapper*).

Для обеспечения возможности использования уже существующих ресурсов, зарегистрированных в реестрах АстроГрида как DSA ресурсы, был разработан адаптер к DSA ресурсам. Этот адаптер играет одну из ключевых ролей по реализации гибридной архитектуры, поскольку дополнительно обеспечивается возможность конструирования посредников над многочисленными ресурсами системы АстроГрид.

Реестр регистрируется в посреднике как обычный ресурс, к которому могут быть сформулированы запросы, что позволит идентифицировать ре-

сурсы, уже зарегистрированные в системе АстроГрид, и использовать их для конструирования новых посредников.

## 7 Пример: реализация средств поддержки задачи поиска далеких галактик в гибридной архитектуре

### 7.1 Задача поиска далеких радиогалактик

В качестве примера класса задач для решения в гибридной архитектуре слоя посредников и АстроГрида рассматривается поиск далеких галактик.

Радиогалактики являются самыми массивными звездными системами, наблюдаемыми на космологических расстояниях [25], но это достаточно редкие объекты по сравнению с обычными (нормальными) галактиками. Далекая галактика, порождающая мощное радиоизлучение, часто оказывается слабым объектом в оптическом диапазоне, что существенно затрудняет ее спектрофотометрические исследования и, следовательно, определе-

ние красного смещения. Усилия, которые были предприняты для поиска радиогалактик на больших красных смещениях, привели к обнаружению  $\sim 300$  радиогалактик с  $Z > 2$  (по данным из базы данных NED (<http://nedwww.ipac.caltech.edu>) для красных смещений, определенных по спектрам), но такого количества объектов недостаточно для статистических исследований свойств далеких галактик. Известны разные техники селекции радиисточников для поиска далеких галактик. К ним относятся глубокая спектроскопия пустых полей, прямые снимки в узкополосных фильтрах, спектроскопия объектов вокруг радиогалактики с большим красным смещением [26, 27], использование показателей цвета для выбора кандидатов (color-dropouts) [28].

Один из часто применяемых методов селекции кандидатов в далекие радиогалактики основан на корреляции наблюдаемой крутизны радиоспектра с красным смещением [29]. Именно этот подход использовался в программе «Большое Трио» [30], которая в течение ряда лет проводилась в Специальной астрофизической обсерватории (САО) РАН под руководством академика Ю. Н. Парийского. В начале 1990-х гг. были начаты работы по исследованию SS (Steep Spectrum) выборки каталога RC. Наблюдения по этой программе выполнялись на 3 крупных телескопах: ПАТАН-600, БТА и VLA (NRAO, США). Каталог RC, полученный на основе материалов глубокого обзора «Холод» (ПАТАН-600) и радиокаталог UTRAO [31] использовались для отбора объектов, определения спектров; радиотелескоп VLA — для получения точных координат источников, радиоизображений, морфологии; 6-метровый оптический телескоп БТА — для отождествления и глубокой спектроскопии радиисточников.

При составлении SS выборки каталога RC так же, как и в других группах, занимающихся поиском далеких радиогалактик, в качестве основного критерия отбора использовалась крутизна радиоспектра и еще дополнительные условия [32]:

- плотность потока (спектральная плотность потока излучения или плотность потока — количество электромагнитной энергии в единичном интервале частот, протекающей через единичную площадку за единицу времени. Внесистемная единица измерения, применяемая в радиоастрономии, — Янский ( $1 \text{ Ян} = 10^{-26} \text{ Вт}/(\text{м}^2 \text{ Гц})$ );
- морфология объектов. Были отобраны двойные радиисточники;
- угловые размеры. На больших красных смещениях не зарегистрировано объектов очень

больших угловых размеров, поэтому учитывалась феноменологическая зависимость размеров двойных радиогалактик от красного смещения.

С появлением в конце XX — начале XXI в. новых больших по площади покрытия глубоких обзоров неба в радиодиапазоне методика селекции по крутизне радиоспектра не теряет своей значимости. Чтобы повысить эффективность изучения выборки радиисточников с крутыми и ультракрутыми спектрами, часто используют дополнительные ограничения по показателю цвета [28] для отождествленных объектов. Трудоемкость работы с разнородными астрономическими каталогами и обзорами требует нового подхода при решении разных астрономических задач, в частности это относится и к задаче составления поисковых списков для дальнейших более детальных исследований.

Задача поиска далеких галактик была сформулирована в контексте подхода, развиваемого на основе программных средств ВО, а именно: как последовательность типичных действий (выборка данных, сравнение, визуализация) с астрономическими данными, доступными в Интернете по протоколам IVOA. В качестве ресурсов, используемых в задаче, были выбраны: каталог RC [30], полученный на частоте 3,9 ГГц из наблюдательного материала глубокого обзора  $20'$  полосы неба, centered на склонение источника SS433, проведенного на радиотелескопе ПАТАН-600; каталог и обзор неба на частоте 1,4 ГГц FIRST [33]; обзор в оптическом диапазоне SDSS [34]. Отметим, что оптическое отождествление радиисточников простыми алгоритмами кросс-мэтчинга, такими как ConeSearch или XMatch, возможно, если каталог имеет высокую точность координат и угловое разрешение (порядка  $1''$ ), для точечных источников. Каталог RC имеет координатную точность  $\sim 15''$  по прямому восхождению RA и  $\sim 45''$  по склонению DEC и не имеет данных по морфологии источников, поэтому необходимы дополнительные ограничения для оптических объектов, попадающих в область поиска, а именно, ограничения по показателю цвета. Высокое угловое разрешение и точность координат лучше уловой секунды при предельной чувствительности по плотности потока  $\sim 1 \text{ мЯн}$  обзора FIRST позволяют сделать выбор из кандидатов на оптическое отождествление.

Поток работ для такой задачи может выглядеть следующим образом:

- найти радиисточники, у которых значение спектрального индекса (спектральный индекс — показатель степени  $\alpha$  в выражении  $S(\nu) \sim \nu^{-\alpha}$ , характеризующем изменение спек-

тральной плотности потока излучения  $S(\nu)$  у ряда космических объектов с изменением частоты  $\nu$  находится в заданном диапазоне  $[\alpha_1, \alpha_2]$ . Если этот параметр неизвестен, то для его определения необходимо

- выбрать радиоисточники, для которых есть данные по плотности потока, по крайней мере, на двух заданных частотах  $\nu_1$  и  $\nu_2$ ;
  - вычислить спектральный индекс  $\alpha$  для выбранных радиоисточников;
  - выдать список радиоисточников, у которых спектральный индекс  $\alpha$  находится в диапазоне  $[\alpha_1, \alpha_2]$ ;
- найти радиоисточники, у которых линейные размеры меньше заданного значения  $d$  и величина потока  $s$  для заданной частоты лежит в диапазоне  $[s_1, s_2]$ ;
  - найти оптические объекты, совпадающие по координатам (с учетом ошибок координат и размеров радиоисточников) с выбранными радиоисточниками;
  - выдать список радиоисточников, для которых найдены соответствующие оптические объекты;
  - по запросу пользователя для выбранной пары радио- и оптического источников показать изображение источника в оптическом диапазоне с нанесенным контуром радиоизображения;
  - выдать запрос пользователю на подтверждение отождествления радио- и оптического изображений;
  - найти значения потоков (звездных величин) для указанных фильтров для оптических источников из сформированного списка;
  - вычислить показатели цвета для оптических источников;
  - выбрать источники, у которых показатели цвета  $c_1$  и  $c_2$  лежат в интервалах  $[a_1, b_1]$  и  $[a_2, b_2]$  соответственно;
  - выдать список источников — кандидатов в далекие галактики.

Выполнение описанной выше последовательности действий для составления поискового списка радиоисточников осложняется тем, что не всегда требуемые данные имеются в выбранных каталогах, а также точностью определения параметров в отдельно взятом ресурсе.

Каталог RC имеет недостаточную для оптического отождествления координатную точность, и

размер области кросс-мэтчинга с оптическим каталогом будет  $45'' \times 45''$  (если мы используем утреннюю ошибку координат по склонению). По нашей оценке при средней плотности объектов SDSS ( $\sim 7-8$  на кв. угл. мин в полосе обзора «Холод») в область поиска для источников каталога RC (размеры ее варьируются от  $45''$  до  $10'$ ) попадает до тысячи объектов. Поскольку точность определения координат радиоисточников не позволяет выполнить оптическое отождествление по позиционному совпадению, то нужна дополнительная селекция по свойствам объектов в оптическом диапазоне. Красное смещение приводит к сдвигу спектрального распределения энергии (Spectral Energy Distribution или SED) далеких галактик в более длинноволновую область спектра по сравнению с близкими галактиками. В оптическом диапазоне скачок в SED для  $\lambda = 912 \text{ \AA}$  для галактик с большим  $Z$  смещается из голубой в красную область спектра, т. е. далекие объекты — более красные и более яркие в этой области спектра по сравнению с близкими галактиками. Аналогично работает избыток излучения в линии водорода (свойство, присущее радиогалактикам)  $Ly$  ( $\lambda = 1216 \text{ \AA}$ ). Используя звездные величины родительских галактик радиоисточников, можно определить показатель цвета или вторые разности показателей цвета и использовать их для выбора наиболее вероятного кандидата из числа объектов, попавших в область поиска. В качестве дополнительного селекционного правила использовались следующие ограничения:

$$\frac{u+r}{2} - g > 1; \quad \frac{g+i}{2} - r > 1; \quad \frac{r+z}{2} - i > 1,$$

является наиболее вероятным кандидатом для отождествления.

Для подтверждения отождествления отобранных источников из обзора FIRST извлекаются изображения, по которым строятся контурные карты, накладываемые на оптические изображения, и отмечаются каталожные положения. По такому рисунку исследователь принимает решение.

Таким образом, поток работ для задачи окончательно определяется выбранными ресурсами и для набора: каталог RC, радиообзор FIRST и оптический обзор SDSS — выглядит следующим образом:

- извлечение из каталога RC источников, попадающих в область пересечения с обзорами FIRST и SDSS;
- извлечение из обзора SDSS объектов, попадающих в эту же область;
- проведение кросс-идентификации двух выборок;

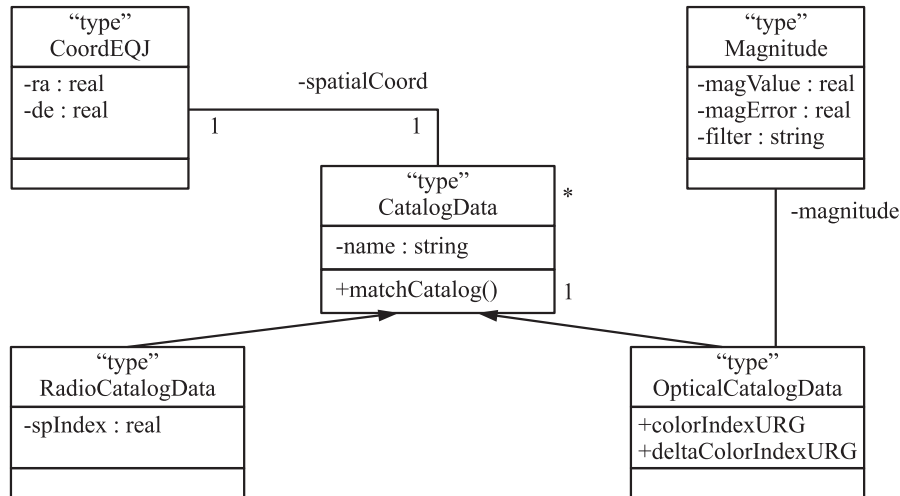


Рис. 3 Спецификация типов фрагмента схемы посредника

- вычисление для оптических объектов, попавших в область поиска, центрированную на положение источника каталога RC, показателей цвета по звездным величинам в фильтрах  $u$ ,  $g$ ,  $r$ ,  $i$  и  $z$  с учетом ошибок определения звездных величин;
- отбор из оптических объектов, удовлетворяющих одному из условий:  $(u + r)/2 - g > 1$ ,  $(g + i)/2 - r > 1$ ,  $(r + z)/2 - i > 1$ ;
- показ по запросу пользователя для выбранной пары радио- и оптического источников изображения источника в оптическом диапазоне с нанесенным контуром радиоизображения из обзора FIRST;
- занесение координаты оптического объекта в список при подтверждении отождествления.

## 7.2 Схема посредника

На рис. 3 приведен пример фрагмента схемы посредника для задачи поиска далеких галактик. В языке СИНТЕЗ этому фрагменту соответствует следующая спецификация (спецификации элементов схемы посредника объединены в модуль *Mediator*).

```
{Mediator; in: module;
  type:
    {CoordEQJ; in: type;
      ra: real;
      de: real;
    },
    {CatalogData; in: type;
      name: string;
      spatialCoord: CoordEQJ;
      matchCatalog: {in: function;
```

```
  params:
    {+o/CatalogData,
      +rad1/real,
      +rad2/real,
      -returns/boolean};
    };
  },
  {Magnitude; in: type;
    magValue: real;
    magError: real;
    filter: string;
  },
  {RadioCatalogData; in: type;
    supertype: CatalogData;
    spIndex: real;
  },
  {OpticalCatalogData; in: type;
    supertype: CatalogData;
    colorIndexURG: real;
    deltaColorIndexURG: real;
    magnitude: {set;
      type_of_element: Magnitude;};
  };
class_specification:
  {scienceData; in: class;
    instance_section: ScienceData;
  },
  {catalogData; in: class;
    superclass: scienceData;
    instance_section: CatalogData;
  },
  {radioCatalogData; in: class;
    superclass: catalogData;
    instance_section:
      RadioCatalogData;
  },
```

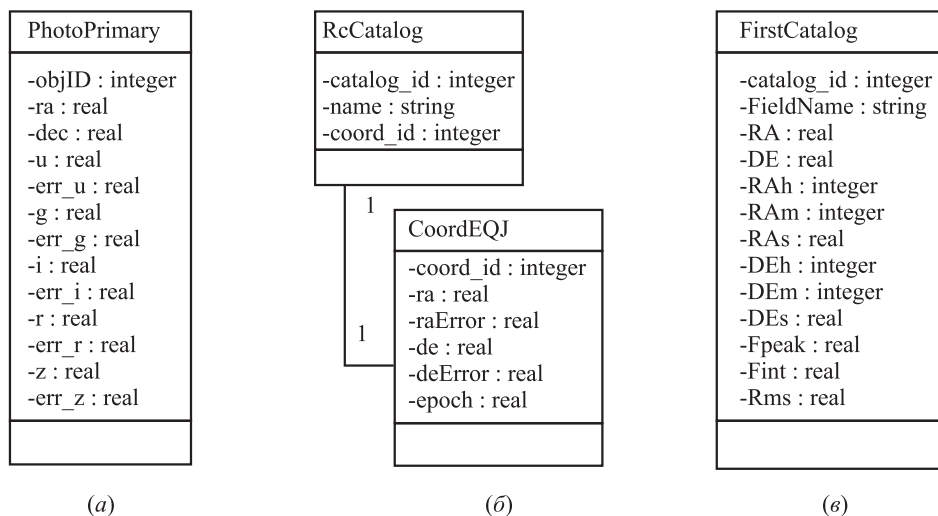


Рис. 4 Спецификация типов экземпляров классов фрагментов схемы SDSS (а), RC (б) и FIRST (в)

```
{opticalCatalogData; in: class;
  superclass: catalogData;
  instance_section:
    OpticalCatalogData;
};
}
```

В ней определены типы (снабжены стереотипом “type”) и классы, соответствующие астрономическим наблюдениям, накапливаемым в различных радио- и оптических каталогах. Суперкласс *CatalogData* содержит объекты из обоих каталогов. Этот класс содержит атрибут *spatialCoord*, который определяет координаты астрономического объекта, и метод (предикат) *matchCatalog()*, который возвращает истинное значение, считая два астрономических объекта тождественными, если они «близки» друг к другу в пространстве.

В текущем примере в посреднике были зарегистрированы следующие каталоги и обзоры неба:

- RC catalog — каталог радиоисточников, полученный из наблюдательного материала обзора неба «Холод» на частоте 3,9 ГГц на радиотелескопе ПАТАН 600 [30];
- FIRST — обзор неба на частоте 1,4 ГГц, полученный на радиотелескопе VLA; включает коллекцию цифровых изображений и каталог радиоисточников [33];
- SDSS — глубокий фотометрический (в фильтрах *u*, *g*, *r*, *i* и *z*) и спектральный обзор неба в оптическом диапазоне [34].

В процессе регистрации RC и SDSS рассматриваемый фрагмент схемы ресурса RC включает классы *rcCatalog* и *coordEQJ*; рассматриваемый

фрагмент схемы ресурса SDSS включает класс *photoPrimary* и функцию модуля *xmatch*. Функция *xmatch* представляет хранимую процедуру, доступную для выполнения в SDSS, которая выполняет координатную кросс-идентификацию объектов каталога SDSS и объектов внешней коллекции. Фрагменты схем ресурсов RC, SDSS и FIRST приведены на рис. 4.

В качестве примера дан следующий модуль спецификации фрагмента схемы ресурса SDSS:

```
{SDSS; in: module, local;
  type:
    {PhotoPrimary; in: type;
      objID: integer;
      ra: real;
      dec: real;
      u: real;
      err_u: real;
      g: real;
      err_g: real;
      i: real;
      err_i: real;
      r: real;
      err_r: real;
      z: real;
      err_z: real;
    };
  function:
    {xmatch; in: function;
      params: {
        +ra1/real, +del/real,
        +ra2/real, +de2/real,
        +rad/real, -returns/boolean
      };
    };
};
```

```
class_specification:
  {photoPrimary; in: class;
   instance_section: PhotoPrimary;
  };
}
```

В результате сопоставления элементов схемы посредника и схем ресурсов RC и SDSS в процессе регистрации были построены следующие взгляды. Взгляд для FIRST строится аналогично RC.

Взгляды для SDSS:

– LAV<sub>1</sub>:

```
v_SDSS_OData(x/[name,ra,de,
  colorIndexURG,deltaColorIndexURG])
:- opticalCatalogData(x/[name,
  ra: spatialCoord.ra,
  de: spatialCoord.de,
  colorIndexURG,deltaColorIndexURG])
```

– GAV<sub>1</sub>:

```
v_SDSS_OData(x/[name,ra,de,
  colorIndexURG,deltaColorIndexURG])
:- SDSS.photoPrimary(x/[ra,de:dec,
  objID,u,r,g,err_u,err_r,err_g])
& integerToString(objID, name)
& id(((u + r)/2.0-g),colorIndexURG)
& I_SDSS.deltaColorIndexURG(err_u,
  err_r,err_g, deltaColorIndexURG)
```

Во взгляде GAV<sub>1</sub> используется функция разрешения конфликтов, которая вычисляет значение атрибута *deltaColorIndexURG* на основе значений атрибутов *err\_u*, *err\_r*, *err\_g* из класса *photoPrimary*. Для функций разрешения конфликтов в посреднике определена их реализация посредством PL/SQL.

Взгляды для RC:

– LAV<sub>2</sub>:

```
v_RC_RData(_/[name, ra, de])
:- radioCatalogData(x/[name,
  ra: spatialCoord.ra,
  de: spatialCoord.de])
```

– GAV<sub>2</sub>:

```
v_RC_RData(x/[name, ra, de])
:- RC.rcCatalog(x/[name, catalog_id,
  coord_id])
& RC.coordEQJ(y/[ra, de, coord_id])
```

На основе взглядов LAV строятся следующие инверсные правила отображения класса в класс:

– IR<sub>1</sub>:

```
radioCatalogData(x/[name,
  ra: spatialCoord.ra,
  de: spatialCoord.de]) <-
v_RC_RData(_/[x:#x, name, ra, de])
```

– IR<sub>2</sub>:

```
opticalCatalogData(x/[name,
  ra: spatialCoord.ra,
  de: spatialCoord.de, colorIndexURG,
  deltaColorIndexURG]) <-
v_SDSS_OData(_/[x:#x, name, ra, de,
  colorIndexURG, deltaColorIndexURG])
```

Дополнительно определяется следующее инверсное правило отображения функции в функцию:

– IR<sub>3</sub>:

```
all x/CatalogData, y/CatalogData,
  rad/real
matchCatalog(x, y, rad, rad, b) <-
xmatch(x.spatialCoord.ra,
  x.spatialCoord.de,
  y.spatialCoord.ra,
  y.spatialCoord.de, rad, b)
```

Инверсное правило IR<sub>3</sub> связывает метод *matchCatalog* в схеме посредника с функцией кросс-мэтчинга *xmatch* в схеме ресурса SDSS.

### 7.3 Реализация примера поддержки решения задачи поиска далеких галактик в гибридной архитектуре

Выше был описан пример фрагмента посредника для поддержки решения задачи поиска далеких галактик. Этот посредник был оформлен как СЕА-приложение, к которому реализуются запросы на языке Syfs. В системе АстроГрид был написан поток работ с участием данного посредника (рис. 5) для решения задачи поиска далеких галактик. Этот поток работ состоит из 3-х шагов: нахождение галактик-кандидатов, преобразование результата, получение изображений.

#### 7.3.1 Первый шаг — поиск кандидатов в далекие галактики

На первом шаге происходит нахождение объектов — кандидатов в далекие галактики. Этот шаг реализуется посредником как СЕА-приложение. Для нахождения кандидатов в далекие галактики задается запрос к посреднику, который на основании заданных условий и кросс-мэтчинга объектов из двух радиокаталогов (RC и FIRST) с объектами оптического каталога (SDSS) отбирает объекты — потенциальных кандидатов в далекие галактики. На рис. 5 изображен первый шаг потока работ, на котором видно, что используется СЕА-приложение *ipi.ac.ru/execute/SyfsQuery*, реализующее описанный выше посредник. В качестве входного параметра данному приложению передается содержимое фай-

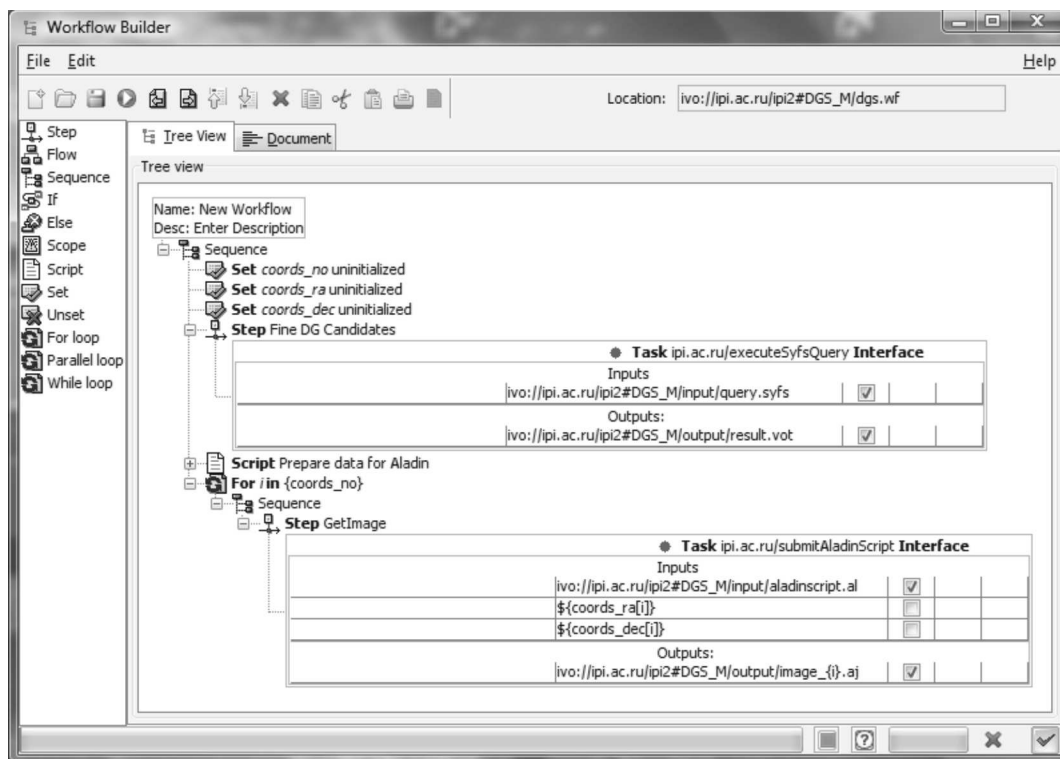


Рис. 5 Поток работ для задачи поиска далеких галактик

ла *query.syfs*, находящегося в виртуальном хранилище пользователя MySpace. В файле содержится следующий запрос на языке Syfs:

```
{ {r(x/[ra, de, name, name1, ra1, de1])
:-radioCatalogData(y/[name,
    ra: spatialCoord.ra,
    de: spatialCoord.de])
& opticalCatalogData(x/[name1: name,
    ra1: spatialCoord.ra,
    de1: spatialCoord.de,
    colorIndexURG,deltaColorIndexURG])
& matchCatalog(y, x, 45, 45, b)
& b = true
& ra >= 120.0 & ra <= 255.0
& de >= 4.39 & de <= 5.61
& ra1 >= 120.0 & ra1 <= 255.0
& de1 >= 4.39 & de1 <= 5.61
& colorIndexURG > deltaColorIndexURG}}
```

Запрос состоит из одного правила и является частью решения задачи поиска далеких галактик. По запросу должны возвращаться астрономические объекты, найденные в радио и оптических каталогах с применением функции кросс-мэтчинга, которые имеют координаты в заданном диапазоне и удовлетворяют специальному условию, ограничивающему значения атрибутов *colorIndexURG* и *deltaColorIndexURG*.

Таблица 1 описывает переписывание, которое применяется к телу правила. Подцели правила перечислены в первом столбце. Инверсные правила, подобранные для унификации с соответствующей подцелью, перечислены во втором столбце. Преобразованные в результате унификации тела инверсных правил перечислены в третьем столбце.

После того как унификация и преобразование тел выбранных инверсных правил выполнены, строится следующее переписывание правила:

```
r(_/[ra, de, name, name1, ra1, de1])
:- v_RC_RData(_/[y:#x, name, ra, de])
& v_SDSS_OData(_/[x:#x, name1: name,
    ra1: ra, de1: de, colorIndexURG,
    deltaColorIndexURG])
& xmatch(y.spatialCoord.ra,
    y.spatialCoord.de,x.spatialCoord.ra,
    x.spatialCoord.de, 45, b)
& ra >= 120.0 & ra <= 255.0
& de >= 4.39 & de <= 5.61
& ra1 >= 120.0 & ra1 <= 255.0
& de1 >= 4.39 & de1 <= 5.61
& colorIndexURG > deltaColorIndexURG
```

Полученное правило содержит сколемовские атрибуты *#x* во взгляде *v\_SDSS\_OData* и *#x* во взгляде *v\_RC\_RData*. Но в процессе семантического анализа устанавливается, что взгляд *v\_SDSS\_OData*

**Таблица 1** Переписывание подцелей правила

Подцель	ИП	Преобразованное тело ИП
radioCatalogData(y/[name, ra: spatialCoord.ra, de: spatialCoord.de])	IR <sub>1</sub>	v_RC_RData(_/[y:#x, name, ra, de])
opticalCatalogData(x/[name1: name, ra1: spatialCoord.ra, de1: spatialCoord.de, colorIndexURG, deltaColorIndexURG])	IR <sub>2</sub>	v_SDSS_OData(_/[x:#x, name1: name, ra1:ra, de1:de, colorIndexURG, deltaColorIndexURG])
matchCatalog(y, x, 45, 45, b)	IR <sub>3</sub>	xmatch(y.spatialCoord.ra, y.spatialCoord.de, x.spatialCoord.ra, x.spatialCoord.de, 45, b)

включает равенство переменной, привязанной к терму  $\#x.spatialCoord.ra$ , и переменной, привязанной к терму  $ra$ . Следовательно, с учетом того, что в правиле в проекции в подцели  $v\_SDSS\_OData$  приравниваются атрибуты  $x$  и  $\#x$ , а также атрибуты  $ra1$  и  $ra$ , из правила можно удалить равенства  $x.spatialCoord.ra = ra1$ . Это фактически означает, что в правиле вхождения термина  $x.spatialCoord.ra$  заменяются термом  $ra1$ . Аналогично семантический анализ выполняется для  $x.spatialCoord.de$  и  $de1$ , для  $y.spatialCoord.ra$  и  $ra$ , для  $y.spatialCoord.de$  и  $de$ . В результате образуется правило без вхождений сколемовских функций:

```
r(_/[ra, de, name, name1, ra1, de1])
:- v_RC_RData(_/[name, ra, de])
& v_SDSS_OData(_/[name1: name,
  ra1: ra, de1: de, colorIndexURG,
  deltaColorIndexURG])
& xmatch(ra, de, ra1, de1, 45, b)
& ra >= 120.0 & ra <= 255.0
& de >= 4.39 & de <= 5.61
& ra1 >= 120.0 & ra1 <= 255.0
& de1 >= 4.39 & de1 <= 5.61
& colorIndexURG > deltaColorIndexURG
```

Дальше выполняется разворачивание взглядов GAV (взгляда  $v\_RC\_RData$  и взгляда  $v\_SDSS\_OData$ ).

Голова взгляда GAV<sub>1</sub>

```
v_SDSS_OData(x/[name,ra,de,
  colorIndexURG,deltaColorIndexURG])
:- SDSS.photoPrimary(x/[ra,de:dec,
  objID,u,r,g,err_u,err_r,err_g])
& integerToString(objID, name)
& id(((u + r)/2.0 - g),colorIndexURG)
& I_SDSS.deltaColorIndexURG(err_u,
  err_r,err_g, deltaColorIndexURG)
```

унифицируется с подцелью

```
v_SDSS_OData(_/[name1: name, ra1: ra,
  de1: de, colorIndexURG,
  deltaColorIndexURG]),
```

которая заменяется на преобразованное тело взгляда

```
SDSS.photoPrimary(x1/[ra1:ra,de1:dec,
  objID,u,r,g,err_u,err_r,err_g])
& integerToString(objID, name1)
& id(((u+r)/2.0-g), colorIndexURG)
& I_SDSS.deltaColorIndexURG(err_u,
  err_r,err_g, deltaColorIndexURG)
```

Аналогично для взгляда  $v\_RC\_RData$  соответствующая подцель заменяется преобразованным телом взгляда GAV<sub>2</sub>:

```
RC.rcCatalog(x2/[name, catalog_id,
  coord_id])
& RC.coorEQJ(y2/[ra, de, coord_id])
```

Результатом разворачивания взглядов GAV является следующее правило, которое переписывает исходное правило, выражая его в терминах схем ресурсов SDSS и RC:

```
r(_/[ra, de, name, name1, ra1, de1])
:-RC.rcCatalog(x2/[name, catalog_id,
  coord_id])
& RC.coorEQJ(y2/[ra, de, coord_id])
& SDSS.photoPrimary(x1/[ra1:ra,
  de1:dec,objID,u,r,g,
  err_u,err_r,err_g])
& integerToString(objID, name1)
& id(((u + r) / 2.0 - g),
  colorIndexURG)
& I_SDSS.deltaColorIndexURG(err_u,
  err_r,err_g, deltaColorIndexURG)
& xmatch(ra, de, ra1, de1, 45, b)
& ra >= 120.0 & ra <= 255.0
& de >= 4.39 & de <= 5.61
& ra1 >= 120.0 & ra1 <= 255.0
& de1 >= 4.39 & de1 <= 5.61
& colorIndexURG > deltaColorIndexURG
```

Аналогично переписывание с использованием взглядов для ресурса FIRST приводит ко второму переписыванию исходного правила запроса:

```
r(_/[ra, de, name, name1, ral, del])
:-FIRST.firstCatalog (x2/[name:
  FieldName, ra: RA, de: DE])
& SDSS.photoPrimary(x1/[ral:ra,
  del:dec,objID,u,r,g,
  err_u,err_r,err_g])
& integerToString(objID, name1)
& id(((u + r) / 2.0 - g),
  colorIndexURG)
& I_SDSS.deltaColorIndexURG(err_u,
  err_r,err_g, deltaColorIndexURG)
& xmatch(ra, de, ral, del, 45, b)
& ra >= 120.0 & ra <= 255.0
& de >= 4.39 & de <= 5.61
& ral >= 120.0 & ral <= 255.0
& del >= 4.39 & del <= 5.61
& colorIndexURG > deltaColorIndexURG
```

Полученный переписанный запрос состоит из двух правил, и результирующий класс *r* образуется как объединение результирующих множеств этих двух правил. Правила запросов обращаются к классам трех ресурсов (RC, FIRST и SDSS), поэтому дальше необходим этап планирования, в результате которого получается распределенный план запроса.

Алгоритм планирования выделяет подзапросы, которые адресованы каждому ресурсу, и остаточный запрос, который будет выполняться в СУБД посредника над результатами, полученными от ресурсов. Также алгоритм выбирает, как данные с результатами подзапросов передаются от одного ресурса к другому ресурсу или посреднику. В этом примере учитывается способность ресурса SDSS принимать внешнюю коллекцию, которая затем участвует в выполнении запроса к SDSS, и то, что число объектов, удовлетворяющих условиям запроса в RC во много раз меньше, чем число объектов в SDSS (число объектов в SDSS исчисляется сотнями миллионов, что делает невозможным выполнение прямого подзапроса к SDSS с последующей передачей результатов в посредник). Поэтому строится следующий распределенный план (для простоты составляющие плана даны в логическом языке Syfs, хотя в реальной реализации план запроса представляется на алгебраическом языке Asyfs).

Подзапрос к ресурсу RC:

```
rc0(_/[ra, de, name])
:-RC.rcCatalog(x2/[name, catalog_id,
  coord_id])
& RC.coordeQJ(y2/[ra, de, coord_id])
& ra >= 120.0 & ra <= 255.0
& de >= 4.39 & de <= 5.61
```

Результаты выполнения этого подзапроса передаются адаптером ресурса RC в адаптер ресурса SDSS, который загружает их во временную коллек-

цию на ресурсе SDSS. Далее выполняется подзапрос к ресурсу SDSS:

```
sdss1(_/[ra, de, name, ral, del,
  objID, colorIndexURG])
:-rc1(_/[ra, de, name])
& SDSS.photoPrimary(x1/[ral:ra,
  del:dec,objID,u,r,g,
  err_u,err_r,err_g])
& id(((u + r) / 2.0 - g),
  colorIndexURG)
& xmatch(ra, de, ral, del, 45, b)
& ral >= 120.0 & ral <= 255.0
& del >= 4.39 & del <= 5.61
```

Результат выполнения этого подзапроса передается адаптером SDSS в посредник. Подзапрос к ресурсу FIRST:

```
first0(_/[ra, de, name])
:-FIRST.firstCatalog (x2/[name:
  FieldName, ra: RA, de: DE])
& ra >= 120.0 & ra <= 255.0
& de >= 4.39 & de <= 5.61
```

Результаты выполнения этого подзапроса передаются адаптером ресурса FIRST в адаптер ресурса SDSS, который загружает их во временную коллекцию на ресурсе SDSS. Далее выполняется подзапрос к ресурсу SDSS:

```
sdss2(_/[ra, de, name, ral, del,
  objID, colorIndexURG])
:-first0(_/[ra, de, name])
& SDSS.photoPrimary(x1/[ral:ra,
  del:dec,objID,u,r,g,
  err_u,err_r,err_g])
& id(((u + r) / 2.0 - g),
  colorIndexURG)
& xmatch(ra, de, ral, del, 45, b)
& ral >= 120.0 & ral <= 255.0
& del >= 4.39 & del <= 5.61
```

Результат выполнения второго подзапроса передается адаптером SDSS в посредник. В посреднике над полученными данными выполняется остаточный запрос, вычисляющий конечный результат рассматриваемого примера:

```
r(_/[ra, de, name, name1, ral, del])
:- sdss1(_/[ra, de, name, name1,
  ral, del, objID, colorIndexURG])
& integerToString(objID, name1)
& I_SDSS.deltaColorIndexURG(err_u,
  err_r,err_g, deltaColorIndexURG)
& colorIndexURG > deltaColorIndexURG
```

```
r(_/[ra, de, name, name1, ral, del])
:- sdss2(_/[ra, de, name, name1,
```

```

    ral, del, objID, colorIndexURG])
& integerToString(objID, name1)
& I_SDSS.deltaColorIndexURG(err_u,
    err_r, err_g, deltaColorIndexURG)
& colorIndexURG > deltaColorIndexURG

```

В качестве выходного параметра указывается адрес файла *result.vot* в виртуальном хранилище пользователя MySpace, в который по завершению работы посредника помещается результат в формате VOTable. Результат работы первого шага (посредника) представляется таблицей, каждая строчка которой — кандидат в далекие галактики. Фрагмент таблицы в формате VOTable выглядит следующим образом:

```

<?xml version=-1.0- encoding=-UTF-8-?>
<votable xmlns="http://
www.ivoa.net/xml/VOTable/v1.1">
<resource>
<table name="r">
<field name="de" datatype="double" />
<field name="del" datatype="double" />
<field name="name" datatype="char"
    arraysize="*" />
<field name="name1" datatype="char"
    arraysize="*" />
<field name="ra" datatype="double" />
<field name="ral" datatype="double" />
<data><tabledata>
<tr>
    <td>5.0155554</td>
    <td>5.02104130192953</td>
    <td>RC 0819+0517</td>
    <td>587734948577674422</td>
    <td>120.183464</td>
    <td>120.179933980257</td>
</tr>
<tr>
    <td>5.0155554</td>
    <td>5.02523423172066</td>
    <td>RC 0819+0517</td>
    <td>587734948577673600</td>
    <td>120.183464</td>
    <td>120.179594574991</td>
</tr>
...
</tabledata></data>
</table>
</resource>
</votable>

```

Формат VOTable — это XML документ. Тэгами *field* обозначаются столбцы таблицы, определяя ее структуру. Данные для одной строки заключаются в тэг *tr*, а внутри строки каждое значение заключается в тэг *td*.

Для оценки производительности работы посредника в гибридной архитектуре были проведены 10 тестовых запусков первого шага. Среднее время выполнения первого шага в АстроГриде составило 26 с, минимальное время — 19 с, максимальное время — 42 с. Подобная разница времени выполнения связана с географией расположения ресурсов, зарегистрированных в посреднике. Радиокаталог RC представляет собой DSA источник системы АстроГрид, физически располагающийся на установке АстроГрида в ИПИ РАН. Радиокаталог FIRST представляет собой базу данных, физически располагающуюся на сервере в ИПИ РАН. Каталог SDSS располагается в США. В соответствии с планом, данные из радиоисточников перемещаются в SDSS, где выполняется кросс-мэтчинг, и далее результат возвращается в Москву, в MySpace АстроГрида в ИПИ РАН.

### 7.3.2 Второй шаг

На втором шаге происходит обработка результата, чтобы на третьем шаге данные можно было передать приложению, получающему изображения в цикле. Данный шаг состоит не из приложения SEA, а из скрипта на языке Groovy [35]. В этом скрипте происходит загрузка в память таблицы, полученной на первом шаге. Затем по этой таблице строятся три массива: массив координат га (*coords\_ra*), массив координат dec (*coords\_dec*) и массив индексов (*coords\_no*). При построении массивов координат происходит их преобразование из вещественного представления в формат H:M:S (час:мин:с).

### 7.3.3 Третий шаг

На третьем шаге используется астрономическое приложение Aladin. Aladin — это атлас неба, обеспечивающий визуализацию цифровых астрономических изображений; суперпозицию записей из астрономических каталогов и баз данных; доступ в интерактивном режиме к астрономическим базам данных Simbad, Vizier и др. Приложение обладает как графическим пользовательским интерфейсом, так и командной строкой. В командной строке приложения команды задаются на скриптовом языке программы Aladin.

На третьем шаге в цикле вызывается разработанное SEA-приложение, суть которого сводится к выполнению в Aladin'е программы, написанной на его скриптовом языке. Результатом работы приложения является стек Aladin'а, содержащий результаты запросов, выполняемых в программе. В стеке сохраняются данные из каталогов, изображения и контуры изображений и т. д. Данные стека визуализируются программой Aladin.

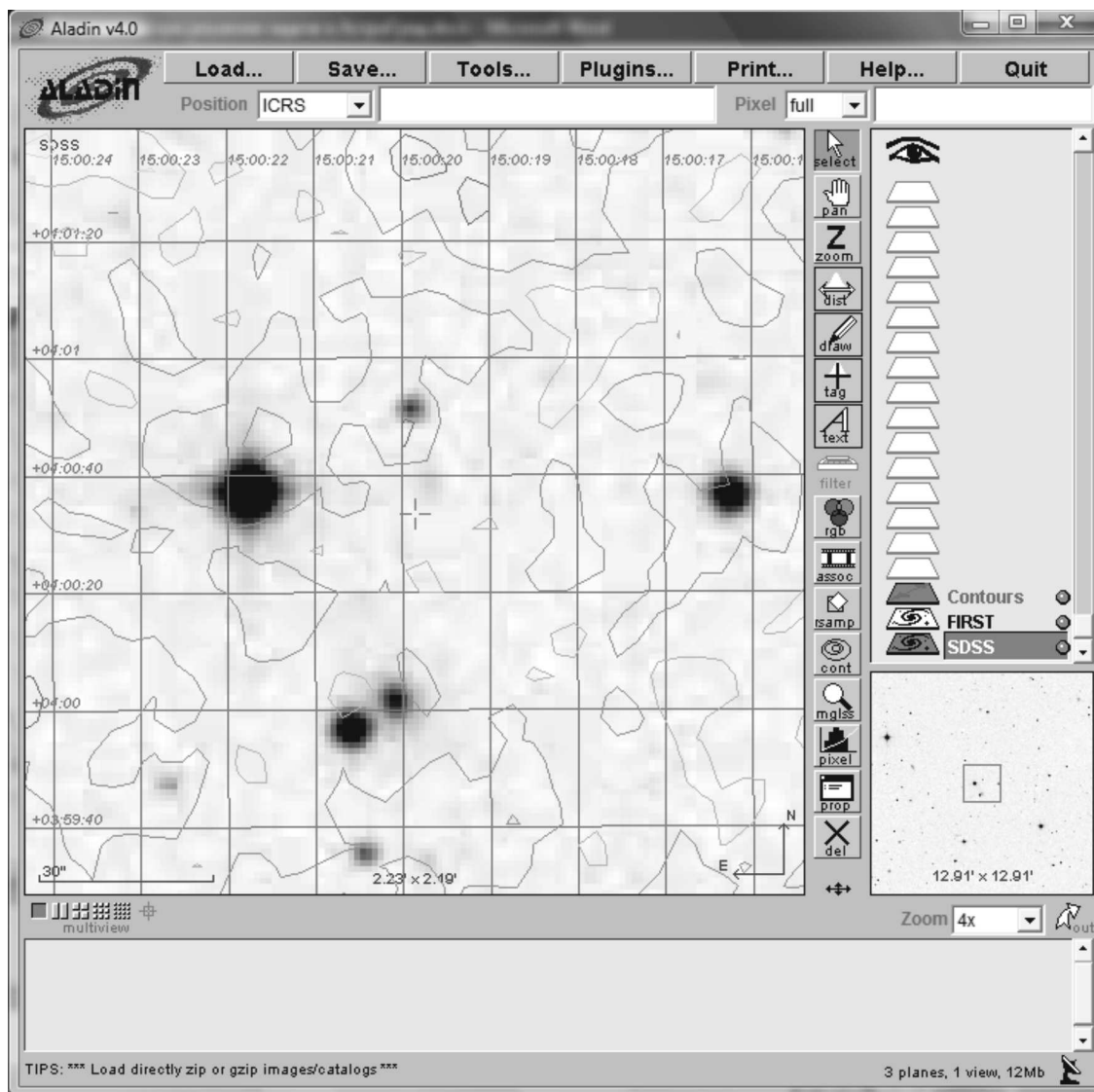


Рис. 6 Пример результата

На рис. 5 изображен вызов этого СЕА-приложения ([ipi.ac.ru/submitAladin.Script](http://ipi.ac.ru/submitAladin.Script)). В качестве второго и третьего входного параметра передаются координаты га и dec:  $i$ -е значение массивов  $coords\_ra[i]$  и  $coords\_dec[i]$  соответственно. В качестве первого входного параметра передается ссылка на файл *aladinscript.al*, находящийся в MySpace. Скрипт является параметризованным двумя переменными:  $\#Ra$  и  $\#Dec$ . В качестве значения этих переменных подставляются те значения, которые передаются СЕА-приложению вторым и третьим параметрами, соответственно. Таким образом, можно выполнять один и тот же скрипт для разных координат.

Скрипт загружает изображения из оптического обзора SDSS и радиообзора FIRST. На оптическое изображение накладываются контуры, построен-

ные по радиоизображению. Результирующее изображение радиооптика (рис. 6) предназначено для анализа специалистом.

## 8 Состояние проблемы интеграции неоднородных информационных ресурсов в посредниках

**Цели введения предметных посредников.** Отличительной особенностью введения посредников в настоящей работе по сравнению с подходами, рассматриваемыми в литературе [15, 36, 37], является их ориентация на решение задач над неоднородны-

ми распределенными информационными ресурсами [38–41] (интеграция информационных ресурсов является лишь частью этой проблемы). Одним из следствий такой цели является то, что схема посредника, наряду со спецификациями структур данных, включает спецификации функций (методов) и процессов. Уточняющая реализация функций посредника устанавливается в существующих ресурсах (сервисах) при их регистрации.

**Отображение схем и определение взглядов.** Отображение схем является одним из критических моментов интеграции информационных ресурсов. Проект Clío [42] интеграции данных и обмена данными поддерживает генерацию отображения схем. Для пары схем средства установления соответствия генерируют множество потенциальных соответствий схем (чисто синтаксических, основанных на отображении термов, имен полей и пр.). Отображение рассматривается как множество запросов из исходной к целевой схеме, осуществляющих преобразование данных ресурса в формат целевой схемы. Эти задачи являются интерактивными (полуавтоматическими). Clío поддерживает отображение реляционных и иерархических XML схем [42].

В другом методе, AutoMed [43], предпринята попытка поддержки трансформации схем для интеграции данных, применяя подходы LAV, GAV и GLAV. При этом используются низкоуровневая гиперграфовая модель данных и набор примитивных преобразований схем в этой модели. Модели данных более высокого уровня и преобразования в них выражаются в терминах общей низкоуровневой модели. Допускаются композиции примитивных трансформаций. Этот синтаксический метод является полуавтоматическим, его можно приспособить для разных структурированных моделей данных.

Настоящая статья представляет совершенно иной, *семантический* подход к трансформации информационных моделей. Во-первых, для информационной интеграции вводится расширяемая каноническая модель (в частности, ядро такой модели составляет гибридная объектно-ориентированная/фреймовая информационная модель [8]). Для любой модели ресурса (определяемой синтаксисом и семантикой двух языков — языка определения информации и языка манипулирования информацией) определяется отображение в расширение канонической модели. Такое отображение обычно конструируется реверсивно [44, 45] и поддерживается соответствующим транслятором (который играет также полезную роль в процессе генерации адаптеров). Подход, основанный на переписывании термов, обеспечиваемый средствами Meta Environment [46], применяется в настоящем

проекте для генерации компиляторов. Соответствие элементов схемы ресурса схеме посредника устанавливается онтологически [9, 10]. Релевантная посреднику схема ресурса, выраженная в некоторой исходной модели, компилируется в каноническую схему. Схема ресурса уточняет схему в канонической модели [45]. Уточнение, задаваемое отображением схем, может быть формально проверено [47]. Отображение схем порождается Унификатором моделей данных, подробно описанным в [7]. После этого, имея схемы, выраженные в одной и той же нотации (каноническая модель), применяется регистрация схемы ресурса в схеме посредника согласно подходу LAV/GLAV, полагаясь на онтологические связи между элементами схем (типами, атрибутами, функциями, параметрами, классами). В результате, применяя исчисление типов [11], формируются правила отображения для подхода LAV/GLAV, согласно описанному в разд. 4.

**Переписывание запросов.** Обзор исследований в области ответа на запрос посредством взглядов, которые простираются от теоретических оснований до алгоритмов соответствующих систем и их реализации, дан в [15]. Кроме того, оценки алгоритмов переписывания запросов даны в работах [19, 36]. Алгоритмы инверсных правил выделяются благодаря их концептуальной простоте, модульности и способности продуцировать максимально включенные переписывания за время, полиномиальное относительно длины запроса и взглядов. Переписывание объединений конъюнктивных запросов при использовании взглядов [19] на основе инверсных правил имеет преимущества по сравнению с известными алгоритмами, в частности этот алгоритм обобщает алгоритмы MiniCon [48] и Ujoin [49] и является более эффективным, чем алгоритм Bucket. Нахождение включаемого переписывания для запросов с объединением конъюнктивных запросов (когда как запрос, так и ограничения взгляда могут содержать встроенные предикаты) является важным свойством этого алгоритма [19]. В то же время исследования семантики объектных запросов в типизированной среде не известны. Одним из вкладов настоящей работы является расширение подхода к переписыванию запросов на основе взглядов в типизированной среде предметных посредников. В противоположность [19], в [20] было показано, как расширить конъюнктивные запросы объектной SPJ семантикой, основанной на отношении уточнения типов и типовом исчислении. Применяемый при этом подход LAV [20] показывает, что уточнение типов экземпляров классов ресурса является основным отношением, необходимым для достижения включения запросов в объектной сре-

де. В настоящей статье, при сохранении типовой среды, показано, как подход LAV/GLAV применяется для реализации функций разрешения конфликтов, возникающих при отображении схем ресурс/посредник. Головы правил определения LAV взглядов могут содержать произвольный запрос в терминах схемы ресурса, разрешающий конфликт при помощи конкретизирующего взгляда, выражаемого как GAV правило над соответствующим ресурсом. Для завершения переписывания взгляд над ресурсом должен быть разрешен в соответствии с подходом GAV (см. разд. 4).

**Прототипы интеграции баз данных для ВО.** Система OpenSkyQuery [50] в NVO [51] обеспечивает интерфейс для выполнения распределенных запросов над федерацией зарегистрированных астрономических архивов. Доступ к каждой базе данных обеспечивается посредством Веб-сервиса SkyNode. Это интерфейс SOAP, воспринимающий запрос на ADQL и возвращающий данные. Узлы в системе могут быть вида Basic или Full. Только узлы Full могут участвовать в операции XMatch. Возможности узла SkyNode Full включают: интерфейс QueryCost(), который выдает простой ADQL запрос для оценки плотности объектов в квадратном градусе в зависимости от условий запроса; возможность использования сложных структур в запросах на ADQL, выполняющих кросс-мэтчинг между содержащимся в базе обзором неба и таблицей в формате VOTable; реализацию метода ExecutePlan, который принимает ExecutePlan и передает релевантную часть плана следующему узлу. Все узлы должны быть зарегистрированы в Реестре ВО. Данные возвращаются обратно от каждого узла и над ними выполняется кросс-мэтчинг. Наконец, результат возвращается клиенту. Система OpenSkyNode использует GAV для интеграции схем баз данных.

Другой прототип распределенной обработки запросов был разработан в Манчестерском университете для гридовой архитектуры OGSA (Open Grid Services Architecture). OGSA-DQP<sup>1</sup> [52] является распределенным процессором запросов, выполненным по отношению к пользователю как OGSA-совместимый Грид сервис. Этот сервис поддерживает компиляцию и вычисление запросов над множеством сервисов Грида. DQP применяет посредник на основе GAV подхода к интеграции данных (объединение схем баз данных составляет глобальную схему). OGSA-DQP поддерживает только такие Веб-сервисы, которые воспринимают примитивные типы в качестве параметров и возвращают либо примитивные типы, либо массивы примитивного типа.

<sup>1</sup>DQP — Distributed Query Processing.

Предлагаемая в настоящей статье гибридная архитектура предметных посредников и АстроГрида отличается от известных прототипов применением расширяемой канонической объектной модели данных, применением движимого приложения-метода формирования предметных посредников на основании LAV/GLAV подхода, использованием развитых методов переписывания запросов к посредникам в типизированной (объектной) среде.

## 9 Заключение

В настоящей статье рассматриваются первые результаты создания промежуточного слоя предметных посредников в гибридной грид-инфраструктуре виртуальной обсерватории для решения научных задач над множеством интегрируемых посредниками неоднородных распределенных информационных ресурсов (таких как базы данных, программные сервисы, онтологические спецификации). Введение такого промежуточного слоя призвано решить ряд семантических проблем взаимодействия ученого, решающего задачу в некоторой предметной области, и разнообразных релевантных задачи результатов наблюдений и средств их обработки. Гибридная архитектура ВО реализована как объединение инфраструктуры системы поддержки ВО АстроГрид, разработанной в Великобритании, и средств поддержки исполнительного слоя предметных посредников, созданных в ИПИ РАН. В исследованной архитектуре предметных посредников реализован подход, движимый приложениями, при котором для класса приложений формируется спецификация предметной области независимо от существующих информационных ресурсов. Далее происходит идентификация ресурсов, релевантных задаче, и их регистрация в посреднике на основе техники GLAV. В различных разделах статьи показано, как, благодаря посредникам, могут быть решены проблемы семантики, возникающие при доказательно правильном отображении информационных моделей ресурсов в каноническую информационную модель посредников, при отображении и интеграции контекстов предметных областей информационных ресурсов в контекст предметной области задачи, при идентификации релевантных задаче информационных ресурсов и формировании их композиций, при интеграции схем ресурсов в схеме посредника и устранении разнообразных конфликтов, при адекватном преобразовании формул (запросов) программы решения задачи, выраженных в терминах схемы предметной области

задачи, в формулы, выраженные в схемах релевантных ресурсов. Показана важная роль уточнения как математически точного соотношения спецификаций при разрешении названных семантических проблем.

Создание прототипа гибридной инфраструктуры касается главным образом сопряжения исполнительных механизмов двух инфраструктур (АстроГрида и средств поддержки исполнительного слоя предметных посредников). Поэтому в статье основное внимание уделено проблемам переписывания запросов к посредникам в планы их реализации над конкретными информационными ресурсами, приведено краткое описание объединенной архитектуры исполнительных механизмов АстроГрида и средств поддержки предметных посредников, дан пример реализации простого предметного посредника для решения задач поиска далеких галактик в гибридной архитектуре.

Полученные результаты свидетельствуют о перспективности исследованного подхода. Планируется их использование при решении разнообразных задач РВО. Вместе с тем планируется существенное развитие предложенного подхода в ряде направлений, включая увеличение гибкости идеи посредников введением комбинированного подхода к их определению, движимому как приложениями, так и ресурсами; расширение средств языка формул, эффективно поддерживаемых совершенствуемыми алгоритмами переписывания запросов; дальнейшее развитие применения онтологического моделирования в совокупности с развитием средств семантической идентификации релевантных ресурсов и их регистрации в посредниках; развитие эффективности архитектуры посредников введением средств оптимизации планов реализации запросов в распределенной среде, совершенствованием адаптеров для повышения эффективности реализации запросов, более тесным сопряжением с Грид архитектурами; расширение применения предметных посредников в других научных областях и развитие методологии решения задач на основе идеи посредников.

## Литература

1. Briukhov D. O., Kalinichenko L. A., Zakharov V. N., et al. Information infrastructure of the Russian Virtual Observatory (RVO). 2nd ed. — IPI RAN, 2005. 173 p.
2. Virtual observatory architecture overview. Version 1.0, IVOA Note 15 June 2004.
3. Briukhov D., Kalinichenko L., Zakharov V. Diversity of domain descriptions in natural science: Virtual observatory as a case study // 7th Russian Conference on Digital Libraries RCDL2005 Proceedings. Yaroslavl, October 2005. P. 23–30.
4. Abrial J.-R. B-technology: Technical overview. — B-Core (UK) Ltd., 1993.
5. Abrial J.-R. The B-book: Assigning programs to meanings. — Cambridge: Cambridge University Press, 1996.
6. The B-toolkit online documentation. <http://www.b-core.com/ONLINEDOC/BToolkit.html>.
7. Захаров В. Н., Калиниченко Л. А., Соколов И. А., Ступников С. А. Конструирование канонических информационных моделей для интегрированных информационных систем // Информатика и её применения, 2007. Т. 1. Вып. 2. С. 15–38.
8. 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.: IPI RAS, 2007. 171 p.
9. Briukhov D. O., Kalinichenko L. A. Component-based information systems development tool supporting the SYNTHESIS design method // 2nd East-European Conference “Advances in Databases and Information Systems” Proceedings. — Berlin-Heidelberg: Springer-Verlag, 1998. P. 305–327.
10. Briukhov D. O., Kalinichenko L. A., Skvortsov N. A. Information sources registration at a subject mediator as compositional development // East-European Conference “Advances in Databases and Information Systems” Proceedings. Lithuania, Vilnius, Springer, LNCS No. 2151, 2001.
11. Kalinichenko L. A. Compositional specification calculus for information systems development // Conference “Advances in Databases and Information Systems (ADBS’99)” Proceedings. Maribor, Slovenia, September 1999, Springer Verlag, LNCS.
12. AstroGrid. <http://www.astrogrid.org>.
13. RVO public utility center. <http://synthesis.ipi.ac.ru/synthesis/projects/RVOAG/>.
14. Ullman J. D. Information integration using logical views // 6th International Conference on Database Theory (ICDT’97) Proceedings, 1997.
15. Alon Y. Halevy. Answering queries using views: A survey // VLDB J. 2001. Vol. 10. No. 4.
16. Friedman M., Levy A., Millstein T. Navigational plans for data integration // National Conference on Artificial Intelligence (AAAI) Proceedings, 1999.
17. Kalinichenko L. A. Integration of heterogeneous semistructured data models in the canonical one // 1st Russian Conference on Digital Libraries RCDL’99 Proceedings. — St.-Petersburg: St.-Petersburg University, 1999.
18. Briukhov D. O., Kalinichenko L. A., Martynov D. O. Source registration and query rewriting applying LAV/GAV techniques in a typed subject mediator // Труды Девятой Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» RCDL, 2007. —

- Переславль-Залесский: Изд-во «Университет города Переславля», 2007. С. 253–262.
19. Wang J., Maher M., Topor R. Rewriting unions of general conjunctive queries using views // 8th International Conference on Extending Database Technology, EDBT'02 Proceedings. Prague, Czech Republic, March 2002.
  20. Kalinichenko L. A., Martynov D. O., Stupnikov S. A. Query rewriting using views in a typed mediator environment // East-European Conference “Advances in Databases and Information Systems” (ADBIS'04) Proceedings. Lecture notes in Computer Science. — Hungary, Budapest: Springer, 2004. Vol. 3255.
  21. Nieuwenhuis R., Oliveras A. Congruence closure with integer offsets // 10th International Conference Logic for Programming Proceedings. Artif. Intell. and Reasoning (LPAR). LNAI. 2003. Vol. 2850. P. 78–90.
  22. Afrati F., Li C., Mitra P. Rewriting queries using views in the presence of arithmetic comparisons // Theor. Comput. Sci. 2006. Vol. 368. No. 1–2. P. 88–123.
  23. The open archives initiative protocol for metadata harvesting. Protocol Version 2.0 of 2002-06-14, Document Version 2004/10/12T15:31:00Z. <http://www.openarchives.org/OAI/2.0/openarchivesprotocol.htm>.
  24. IVOA Astronomical Data Query Language Version 1.01. <http://www.ivoa.net/Documents/WD/ADQL/ADQL-20050624.pdf>.
  25. Spinrad H., Dey A., Stern D., Dunlop J., Peacock J., Jimenez R., Windhorst R. LBDS 53W091: An Old, Red Galaxy at  $z = 1.552$ . *ApJ*. 1997. Vol. 484. P. 581.
  26. Venemans B. P., Rottgering H. J. A., Overzier R. A., Miley G. K., De Breuck C., Kurk J. D., van Breugel W., Carilli C. L., Ford H., Heckman T., McCarthy P., Penttericci L. Discovery of six Ly $\alpha$  emitters near a radio galaxy at  $z \sim 5.2$  // *A&A*. 2004. Vol. 424. P. L17–L20.
  27. Overzier R. A., Miley G. K., Bouwens R. J. et al. Clustering of star-forming galaxies near a radio galaxy at  $z = 5.2$  // *AJ*. 2006. Vol. 637. P. 58–73.
  28. Madau P. Radiative transfer in a clumpy universe: The colors of high-redshift galaxies // *ApJ*. 1995. Vol. 441. P. 18–27.
  29. Rottgering H. J. A., Lacy M., Miley G. K., Chambers K. C., Saunders R. Samples of ultra-steep spectrum radio sources // *A&AS*. 1994. Vol. 108. P. 79–141.
  30. Parijskij Yu., Goss W., Kopylov A., et al. Investigation of RATAN-600 RC radio sources // *Bulletin of Special Astrophys. Obs.* 1996. Vol. 40. P. 5–124.
  31. Douglas J. N., Bash F. N., Torrence G. W., Wolfe C. The Texas survey — Preliminary + 18DEG Strip // *Publ. in Astronomy*. University of Texas. 1980. Vol. 17. No. 1.
  32. Копылов А. И., Госс В. М., Парийский Ю. Н., Соболева Н. С., Желенкова О. П., Темирова А. В., Витковский В. В., Наугольная М. Н., Верходанов О. В. Оптическое отождествление подвыборки радиисточников RC-каталога с крутыми спектрами с помощью 6-метрового телескопа САО РАН. Требования к координатной точности и глубине изображения, наблюдения на VLA и способы оценки фотометрического красного смещения // *АЖ*, 1995. Т. 72. № 4. С. 437–446.
  33. Becker R. H., Helfand D. J., White R. L., et al. The VLA FIRST survey: Faint images of the radio sky at twenty-centimeters. <http://sundog.stsci.edu/top.html>.
  34. Sloan Digital Sky Survey, Data Release 6. <http://www.sdss.org/dr6/>.
  35. Groovy: An agile dynamic language for the Java Platform. <http://groovy.codehaus.org/>.
  36. Grant J., Minker J. A logic-based approach to data integration // *Theory and Practice of Logic Programming*, 2002. Vol. 2(3). P. 293–321.
  37. Lenzerini M. Data integration: A theoretical perspective // *ACM Symposium on Principles of Database Systems (PODS) Proceedings*. 2002.
  38. Калиниченко Л. А., Ступников С. А., Земцов Н. А. Синтез канонических моделей для интеграции неоднородных источников информации. — М.: ИПИ РАН, 2005.
  39. Kalinichenko L. A., Stupnikov S. A., Vovchenko A. E., et al. Russian Virtual Observatory Community Centre for scientific problems solving over multiple distributed information sources // 18th Russian Conference on Digital Libraries RCDL2006 Proceedings. — Suzdal, Russia, 2006. P. 120–129.
  40. Kalinichenko L. Subject mediation approach for scientific problem solving in Virtual Observatories // XXVI General Assembly of the International Astronomical Union. Prague, August 14–25, 2006. P. 454–455.
  41. Kalinichenko L. A., Briukhov D. O., Martynov D. O., Skvortsov N. A., Stupnikov S. A. Mediation framework for enterprise information system infrastructures: Application-driven approach // 9th International Conference on Enterprise Information Systems ICEIS 2007 Proceedings. V. Databases and Information Systems Integration. 2007. P. 246–251.
  42. Haas L. M., Hernández M. A., Ho H., Popa M., Roth M. Clio grows up: From research prototype to industrial tool // *ACM SIGMOD Conference Proceedings*. — Baltimore, Maryland, USA. June 14–16, 2005. P. 805–810.
  43. Jasper E., Tong N., Mc.Brien P., Poulovassilis A. Generating and optimising views from both as view data integration rules // 6th Baltic Conference on Database and Information Systems (DBIS'04) Proceedings. Riga, June 2004.
  44. Kalinichenko L. A., Skvortsov N. A. Extensible ontological modeling framework for subject mediation // 4th All-Russian Conference on Digital Libraries, RCDL'2002 Proceedings. Dubna, October 15–17, 2002.
  45. Kalinichenko L. A. Canonical model development techniques aimed at semantic interoperability in the heterogeneous world of information modeling // Open INTEROP Workshop “Enterprise modeling and ontologies for interoperability” Proceedings at the 16th Conference on “Advanced Information Systems Engineering (CAiSE).” Riga, Latvia, 7–11 June, 2004. P. 101–116.
  46. Van Den Brand M. G., Heering J., Klint P., Oliver P. A. Compiling language definitions: The ASF + SDF compiler // *ACM TOPLAS*. 2002. Vol. 24. No. 4.

47. *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 / Под ред. И. А. Соколова. — М.: ИПИ РАН, 2005. С. 69–95.
48. *Pottinger R., Levy A.* A scalable algorithm for answering queries using views // International Conference on Very Large Data Bases (VLDB) Proceedings, Cairo, Egypt, 2000.
49. *Qian X.* Query folding // International Conference on Data Engineering (ICDE) Proceedings. New Orleans, LA, 1996. P. 48–55.
50. SkyQuery. <http://www.skyquery.net>.
51. *Hanisch R., Quinn P.* The international virtual observatory. <http://www.ivoa.net/pub/info/>.
52. *Alpdemir M. N., Mukherjee A., Gounaris A., Paton N. W., Watson P., Fernandes A. A. A., Fitzgerald D. J.* OGSA-DQP: A service for distributed querying on the grid // Proceedings of EDBT 2004, Springer, LNCS 2992, March 2004.