

Российская Академия Наук
Институт Проблем Информатики

на правах рукописи

БРЮХОВ Дмитрий Олегович

**Конструирование информационных систем на основе
интероперабельных сред информационных ресурсов**

05.13.11. - математическое и программное обеспечение
вычислительных машин, комплексов, систем и сетей

Автореферат

диссертации на соискание ученой степени
кандидата технических наук

МОСКВА
2003

Работа выполнена в лаборатории Композиционных методов проектирования информационных систем Института проблем информатики РАН.

Научный руководитель: доктор физико-математических наук
профессор
Л.А. Калиниченко

Официальные оппоненты: доктор физико-математических наук
В.П. Шириков
кандидат физико-математических наук
Н.А. Маркова

Ведущая организация: Вычислительный Центр РАН

Защита состоится ”23” октября 2003 года в 14 часов на заседании диссертационного совета Д002.073.01 при Институте проблем информатики РАН по адресу:

117333 г. Москва, ул. Вавилова 44 корп.2.

С диссертацией можно ознакомиться в библиотеке Института проблем информатики РАН.

Автореферат разослан ”15” сентября 2003 г.

Ученый секретарь
Диссертационного совета
д.т.н.

С.Н. Гринченко

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность проблемы

Идея конструирования информационных систем (ИС) из компонентов развивается на протяжении многих лет. По существу, речь идет о мегапрограммировании (Дж.Видерхольд) – крупноблочном конструировании систем. Согласно этой идее, целесообразнее осуществлять капиталовложения в создание компонентов, которые можно было бы многократно использовать, чем всякий раз осуществлять разработку ИС сверху – вниз: от спецификаций требований к работающей системе. Несмотря на очевидную привлекательность идеи и многочисленные исследования и разработки в этой области, достигнутые результаты не соответствуют существенно возросшим в последнее время технологическим возможностям.

Сеть Интернет наводнена компонентами. Развивающиеся технологии промежуточного слоя (такие, как, например, CORBA, Java RMI, DCOM, .NET, или в последнее время Web Services) обеспечивают техническую возможность конструирования распределенных, интероперабельных ИС, как из *программных*, так и из *информационных* компонентов. Эти технологии, также, позволяют накапливать репозитории компонентов для их дальнейшего использования при создании новых ИС. Развитие Интернет уже в ближайшем будущем позволит рассматривать его как базу данных с моделью данных, соответствующей стандарту XML. При этом, сайты образуют информационные компоненты, пригодные для использования в составе ИС. Технологии промежуточного слоя и Интернет интегрированы и могут использоваться совместно при создании ИС.

Что же достигнуто в области конструирования ИС из компонентов. Главным образом развиваются методы и средства компонентно-базированного проектирования ИС. При этом традиционно рассматриваются программные компоненты. Примерами таких технологий являются Microsoft's .NET и SunSoft's JavaBeans. Существующие средства создания программ, такие как Microsoft's Visual Studio, Inprise's Delphi и Protosoft's Power Builder позволяют конструировать программы из готовых компонентов, включая компоненты .NET, ActiveX и JavaBeans. Отличительной чертой этих средств является их ориентированность на неполные спецификации компонентов и предварительное, детальное знание программистом возможностей этих компонентов. Эти средства хороши для работы с “локальными” библиотеками, когда предполагается их хорошее знание разработчиком. Такой подход является также небезопасным, поскольку из-за неполноты спецификаций невозможно убедиться в адекватности компонентов спецификации требований.

Наряду с такими технологиями, методы Объектного Анализа и Проектирования (ОАП) предлагают графические нотации и методологии для проектирования ИС. Основной используемой нотацией методов ОАП в

последнее время является унифицированный язык UML. По сути, эти методы реализуют традиционное проектирование сверху – вниз. Средства, которые предоставляются методами ОАП (такими, например, как Rational Rose от Rational Software и Paradigm Plus от Platinum) для повторного использования, требуют глубокого знания разработчиком используемых компонентов, что находит применение для использования библиотек программ (например, использование графических библиотек в средствах визуального проектирования). Часто названные средства используются комбинированно - анализ и проектирование ИС осуществляется на основе методов ОАП, а реализация системы осуществляется с использованием средств визуального программирования (например, Delphi, Power Builder).

Таким образом, известные методы и средства не позволяют использовать потенциальные возможности компонентов, накапливаемых в Интернет, не являются масштабируемыми по числу накопленных компонентов, не позволяют конструировать системы из компонентов различных видов – программных, информационных, являются небезопасными.

Данная работа является исследованием, выполненным в соответствии с *композиционным* подходом к конструированию ИС, развиваемым в рамках проекта СИНТЕЗ в Лаборатории композиционных методов проектирования информационных систем Института проблем информатики РАН. Этот подход ориентирован на преодоление указанных основных ограничений существующих технологий. Он предназначен для *корректной композиции* существующих компонентов, *семантически интероперабельных* в контексте конкретного применения. В отличие от системной, технической интероперабельности (обеспечиваемой инфраструктурами промежуточного слоя), подход рассматривает интероперабельность в более широком, семантическом аспекте. Семантическая интероперабельность означает комбинацию нескольких способностей: способности решения вопроса о релевантности имеющихся компонентов разрабатываемому применению, о соответствии их прикладных контекстов контексту применения, а также о том, что интероперабельная композиция ресурсов будет непротиворечивой в контексте разрабатываемого применения.

Анализ текущего состояния технологических приемов проектирования систем из компонентов и исследований в этой области показывает, что в настоящее время не известны результаты, которые позволяли бы конструировать ИС как композицию компонентов, семантически интероперабельных в контексте спецификации требований, и уточняющую спецификацию требований.

В диссертационной работе осуществляется исследование и разработка методов и средств композиционного подхода, ориентированных на создание ИС из компонентов, технически интероперабельных в рамках некоторого промежуточного слоя.

Цель и задачи работы

Целью диссертационной работы является исследование и разработка алгоритмов конструирования ИС из компонентов, технически интероперабельных в рамках некоторого промежуточного слоя.

Достижение цели предполагает решение следующих задач:

- разработка алгоритмов поиска компонентов, онтологически релевантных спецификации требований;
- разработка алгоритмов разрешения конфликтов между спецификациями требований и компонентов;
- разработка алгоритмов выявления фрагментов спецификации компонентов, которые могли бы служить уточнением соответствующих фрагментов спецификации требований;
- разработка алгоритмов построения композиции таких фрагментов в спецификацию, уточняющую спецификацию требований;
- создание инструментария эксперта-конструктора ИС на основе перечисленных алгоритмов.

Методы исследования

При решении поставленных в работе задач использовались методы объектного проектирования, методы теории множеств, теории графов, теории уточнения спецификаций.

Научная новизна

В диссертационной работе получены следующие новые научные результаты:

- предложен оригинальный подход к конструированию ИС из программных и информационных компонентов, технически интероперабельных в рамках некоторого промежуточного слоя;
- предложена онтологическая модель и на ее основе разработаны алгоритмы поиска релевантных спецификаций по описаниям связанных с ними понятий на естественном языке;
- разработан метод для разрешения различных рассогласований между спецификациями требований и компонентов, объединяющий в себе два известных подхода в области интеграции схем баз данных – применение набора предопределенных правил структурных преобразований, и применение языка высокого уровня для описания функций разрешения конфликтов;
- разработан оригинальный метод конструирования композиции фрагментов спецификации компонентов с использованием операций над типами. Этот метод основан на понятии уточнения - уточняющие спецификации, образуемые при конструировании таких композиций,

могут использоваться всюду вместо уточняемых спецификаций требований, так что пользователи не замечают этой замены.

Практическая ценность

Предлагаемый в диссертационной работе композиционный подход к конструированию ИС может быть использован для создания информационных систем как в рамках Интранет, так и в рамках Интернет. Использование данного подхода позволяет ускорить процесс создания новых ИС. Это также повышает надежность систем за счет использования уже проверенных, протестированных компонентов и уменьшает время на отладку и тестирование систем. Все вместе это ведет к снижению стоимости разработки ИС. Благодаря используемым методам спецификации и поиска компонентов, композиционный подход является масштабируемым по числу накопленных компонентов.

Разработанные алгоритмы могут использоваться и в других задачах. В частности, они использовались при создании предметного посредника для электронных библиотек (в методе регистрации неоднородных коллекций и в методе проектирования персонализированных виртуальных электронных библиотек, создаваемых для конкретных пользователей).

Разработанный на базе этих алгоритмов инструментарий может быть применен также для создания ИС в различных прикладных областях, например, в ИС для научных исследований, электронных библиотеках для образования, интеллектуальных системах управления и принятия решений, и других.

Результаты диссертационной работы использованы в проектах, выполняемых по планам ИПИ РАН, в проектах РФФИ 97-07-90369 и 00-07-90086, в проекте ИНТАС INTAS-94-1817, а также в совместном проекте с Siemens Corporate Research and Development.

Апробация работы

Основные результаты диссертации докладывались на Международных конференциях ADBIS (Москва 1995, Москва 1996, Познань 1998, Марибор 1999, Вильнюс 2001, Дрезден 2003), на Международном симпозиуме по персонализации и рекомендационным системам в электронных библиотеках (Дублин 2001), на Российских конференциях по электронным библиотекам RCDL (Протвино 2000, Петрозаводск 2001, Дубна 2002), на семинаре Московской секции ACM SIGMOD (Москва 1998), на научных семинарах по проекту СИНТЕЗ лаборатории Композиционных методов проектирования информационных систем Института проблем информатики РАН.

На защиту выносятся следующие, полученные автором результаты:

- композиционный подход к конструированию ИС из программных и информационных компонентов, технически интероперабельных в рамках некоторого промежуточного слоя;

- метод и реализующие его алгоритмы поиска элементов спецификаций компонентов, релевантных элементам спецификации требований, на основе онтологической модели;
- метод и реализующие его алгоритмы разрешения структурных конфликтов между спецификациями требований и компонентов;
- метод и реализующие его алгоритмы выявления фрагментов спецификаций компонентов, которые могли бы служить уточнением соответствующих фрагментов спецификации требований;
- метод и реализующие его алгоритмы построения композиции таких фрагментов в спецификацию, уточняющую спецификацию требований;
- разработанный на основе перечисленных алгоритмов инструментарий эксперта-конструктора ИС.

Публикации по теме диссертации

По теме диссертации автором опубликовано 8 работ.

Структура работы

Текст диссертации включает введение, пять глав, заключение, список литературы и два приложения. Основное содержание работы изложено на 145 страницах текста, включая 16 рисунков и 3 таблицы.

СОДЕРЖАНИЕ РАБОТЫ

Введение

Во введении обосновывается актуальность темы диссертационной работы, формулируется цель, научная новизна и практическая значимость полученных результатов, дается краткое содержание глав работы.

Глава 1

В первой главе представлен подход к проектированию ИС на основе композиции существующих компонентов. Этот подход ориентирован на преодоление основных ограничений существующих методов ОАП и методов компонентного проектирования. Он предназначен для проектирования ИС на основе композиции существующих компонентов, технически интероперабельных в рамках некоторого промежуточного слоя.

На Рис.1 представлена общая схема предлагаемого в данной работе композиционного подхода к проектированию ИС. Этот подход включает в себя прямую и обратную фазы. При прямой фазе (левая часть рисунка) осуществляется разработка спецификаций требований к проектируемой ИС. ИС разрабатывается в рамках определенной прикладной области, которая задает прикладную семантику для спецификации требований. Этапы планирования требований и анализа реализуются с помощью применения произвольного метода ОАП, использующего нотацию UML. Выбранный метод ОАП расширяется онтологическими спецификациями, спецификациями функций и

инвариантов, и другими понятиями канонической модели. Полученные спецификации в виде UML диаграмм отображаются в спецификации в канонической модели.

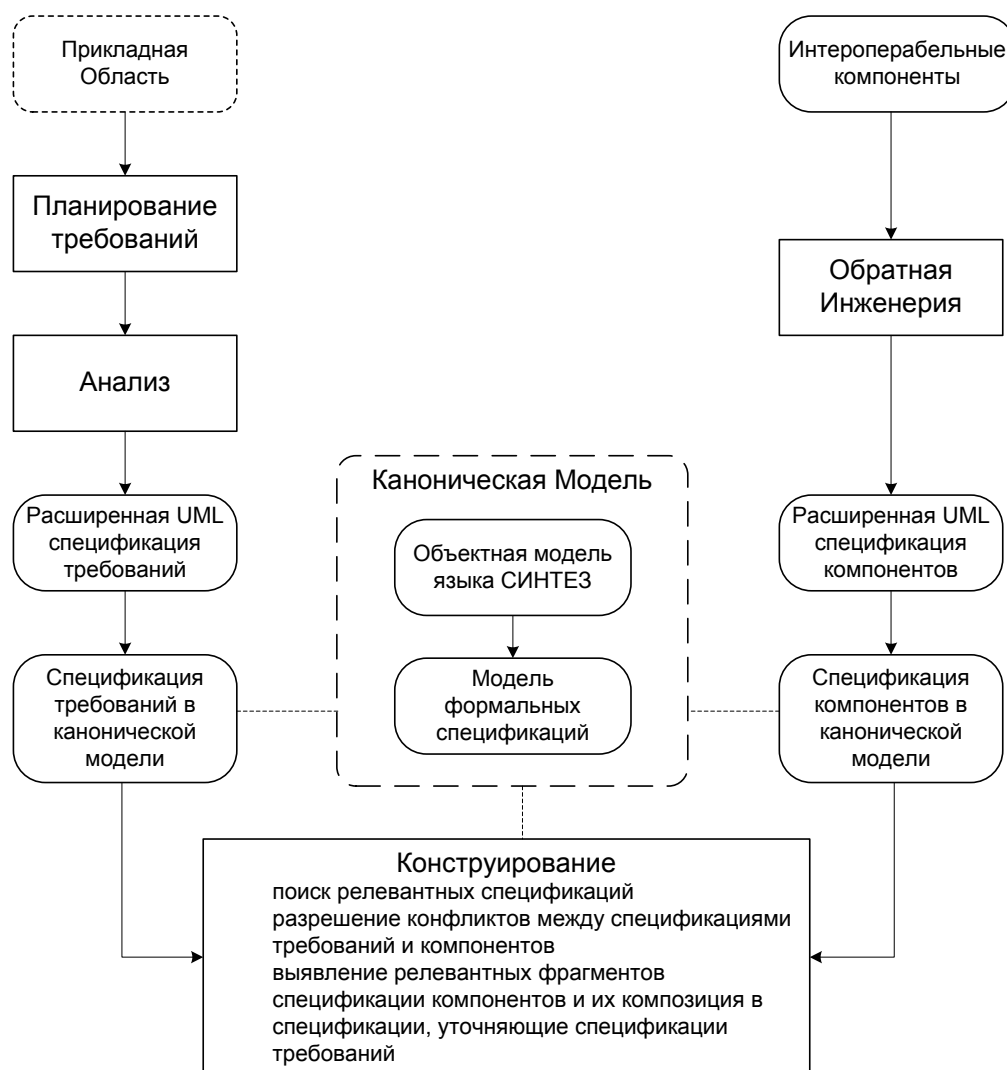


Рис. 1 Схема композиционного подхода к проектированию ИС

При обратной фазе (правая часть рисунка) происходит описание спецификаций компонентов. Для получения спецификаций компонентов (в случае их отсутствия), этап обратной инженерии может быть реализован с помощью применения метода ОАП, использующего нотацию UML. Выбранный метод ОАП расширяется онтологическими спецификациями. Полученные спецификации в виде UML диаграмм отображаются в спецификации в канонической модели, имеющей формальную интерпретацию.

В качестве канонической модели используется язык СИНТЕЗ, ориентированный на семантическую интероперабельность и композиционное проектирование ИС в широком диапазоне существующих неоднородных

информационных компонентов. Для придания канонической модели точной интерпретации существует ее отображение в формальную модель, в качестве которой выбрана Нотация Абстрактных Машин В-Технологии. Представление основных понятий языка СИНТЕЗ в нотации UML описано в Приложении Б.

Полученные спецификации требований и компонентов используются на этапе конструирования. Этап проектирования абстрактной модели методов ОАП является более широким, по сравнению с тем, что рассматривается в диссертационной работе. Та часть этапа проектирования, в рамках которой реализуется использование существующих компонентов в разрабатываемой ИС, называется в работе этапом конструирования. Этот этап определяет процесс уточнения спецификаций требований ИС композицией существующих программных и информационных компонентов.

Этап конструирования является основой композиционного подхода к проектированию ИС из интероперабельных информационных компонентов. Диссертационная работа сфокусирована на данном этапе. Целью этапа является конструирование *композиции фрагментов спецификаций компонентов* для получения спецификаций, *уточняющих* спецификации требований. Этап конструирования состоит из следующих шагов:

1. поиск релевантных спецификаций;
2. разрешение конфликтов между спецификациями требований и компонентов;
3. выявление релевантных фрагментов компонентов и их композиция в спецификации, уточняющие спецификации требований.

Автором дан обзор существующих подходов и методов в перечисленных областях.

Анализ текущего состояния технологических приемов проектирования систем из компонентов и исследований в этой области показывает, что в настоящее время не известны результаты, которые позволяли бы конструировать ИС как уточняющую спецификацию требований композицию компонентов, семантически интероперабельных в контексте этой спецификации.

Глава 2

Во второй главе описывается подход к поиску элементов (классов, типов, атрибутов, функций) спецификаций компонентов, подходящих для их использования в разрабатываемой ИС, и алгоритмы, реализующие этот подход. Основой для поиска служит онтологическая модель.

Онтологическое понятие отражает существенные свойства, связи и отношения класса объектов реального мира, воспринимаемые агентами в данной предметной области.

Для определения онтологии конкретной предметной области используются онтологические спецификации, задающие определения понятий предметной

области и связей между ними. Один из практических подходов к представлению онтологий заключается в их определении в виде словаря понятий предметной области, отношений между ними и накладываемых на них ограничений.

В диссертационной работе спецификация понятия ограничивается его словарным описанием на естественном языке. При этом рассматриваются следующие ассоциации между понятиями: обобщения/специализации и позитивные ассоциации.

Онтологический контекст определяет онтологическое связывание для компонентов. Элементы спецификации компонентов связываются с соответствующими им онтологическими понятиями. Использование близких по смыслу онтологических спецификаций в разных компонентах является необходимой предпосылкой корректной взаимной интерпретации элементов спецификаций. Тем самым, онтологии предоставляют основу для семантического взаимодействия компонентов.

В работе онтологические спецификации используются для поиска релевантных элементов спецификаций требований и компонентов. Элемент спецификации компонента *онтологически релевантен* элементу спецификации требований того же вида (класс, тип, атрибут, функция, параметр), если между соответствующими им онтологическими понятиями установлена позитивная ассоциация, или ассоциация обобщения/специализации.

Поскольку спецификация требований и каждый компонент могли разрабатываться с использованием разных онтологических спецификаций, возникает задача интеграции онтологий компонентов и онтологии спецификации требований. Для этого необходимо установить ассоциации между онтологическими понятиями спецификации требований и компонентов.

Для решения этой задачи вводится общая онтология. Общая онтология содержит онтологические понятия, характерные для конкретной предметной области. Эксплуатация такой онтологии предполагает согласованное понимание ее понятий и отношений. В этом случае отображение онтологических понятий спецификации требований и компонентов в понятия общей онтологии позволит находить соответствия между ними.

Предполагается, что существует множество общих онтологических модулей для различных предметных областей. При создании онтологии спецификации требований разрабатываемой ИС эксперт может использовать 2 варианта.

При первом варианте эксперт использует существующую общую онтологию предметной области, близкую к той, в рамках которой разрабатывается ИС. При этом эксперт может добавлять новые понятия, характерные для разрабатываемой ИС.

При втором варианте эксперт разрабатывает онтологию спецификации требований независимо от общей онтологии. Общая онтология выбирается по возможности близкой к предметной области ИС.

Диссертационная работа ориентирована на второй вариант, как более общий. Использование первого варианта упрощает процесс установления связей между понятиями онтологии спецификации требований и общей онтологии.

Онтологии компонентов могли разрабатываться с использованием одного из перечисленных вариантов. При первом варианте онтология компонента могла разрабатываться как в терминах той же общей онтологии, что и спецификация требований, так и в терминах совершенно другой онтологии.

В диссертационной работе рассматривается второй вариант, как более общий.

При использовании общей онтологии алгоритмы поиска онтологически релевантных элементов спецификации требований и компонентов включают:

- алгоритмы отображения понятий спецификации требований (и компонентов) в понятия общей онтологии.

При этом устанавливаются связи между понятиями на основе функций корреляции;

- алгоритмы установления ассоциаций между понятиями спецификации требований и компонентов на основе композиции ассоциаций между понятиями.

Понятия связываются, если существует путь из понятия спецификации требований в понятие компонента;

- алгоритмы установления ассоциаций между элементами схем спецификации требований и компонентов.

Элементы спецификаций компонентов онтологически релевантны элементам спецификации требований, если между соответствующими им онтологическими понятиями установлена позитивная ассоциация, или ассоциация обобщения/специализации.

В Главе 2 разработан подход, ориентированный на достижение семантической интероперабельности компонентов при конструировании ИС. Разработанный подход основан на интеграции онтологических контекстов компонентов и спецификации требований, используя понятие общей онтологии предметной области. Определены алгоритмы, обеспечивающие указанную интеграцию, и поиск в таком интегрированном контексте спецификаций компонентов и их фрагментов, онтологически релевантных спецификации требований.

Представленные алгоритмы обеспечивают поиск семантически релевантных компонентов для использования на последующих шагах конструирования разрабатываемой ИС.

Глава 3

В третьей главе описывается подход к разрешению конфликтов между фрагментами спецификаций требования и компонентов, и реализующие его алгоритмы.

При конструировании ИС из готовых компонентов неизбежно возникают различные конфликты между спецификациями требований и компонентов. Конфликты могут возникать как из-за разных областей применения, так и из-за разного видения разработчиками представления спецификаций одной и той же ИС. Метод разрешения конфликтов между спецификациями, предлагаемый в диссертационной работе, основан на комбинации двух известных подходов в области интеграции схем баз данных – применение набора предопределенных правил структурных преобразований, и применение языка высокого уровня для описания функций разрешения конфликтов. Функции разрешения конфликтов задаются с помощью формул языка СИНТЕЗ. Язык формул является вариантом типизированного языка логики первого порядка.

В качестве примера рассмотрим конфликт различного набора атрибутов. Имеем 2 релевантных типа *Proposal* и *Submission* с разным набором атрибутов:

```
{Proposal;
  in: type;
  starting_date: time;
  termination_date: time;
  ...
}
{Submission;
  in: type;
  starting_date: time;
  duration: time;
  ...
}
```

Между атрибутами *starting_date* типа *Proposal* и типа *Submission* конфликтов нет. Атрибуту *termination_date* типа *Proposal* нет эквивалентного атрибута в типе *Submission*, но он может быть вычислен из атрибутов *starting_date* и *duration*. Функция разрешения конфликта выглядит следующим образом:

```
get_termination_date: {in: function;
  _params: {+s/Submission, -returns/time};
  {{ returns = s.starting_date + s.duration }}
}
```

Для разрешения конфликтов структурного вида используется подход, основанный на применении набора правил структурных преобразований. Они устанавливают релевантность путей спецификаций требований и компонентов, и задают правила построения функций разрешения конфликтов. Предлагаемый

подход позволяет автоматизировать процесс поиска и разрешения структурных конфликтов между спецификациями требований и конкретных компонентов.

Под *связью* понимается любое прямое соединение между элементами спецификаций. Различаются *атрибутная связь* между типом и его атрибутом, *ссылочная связь* между двумя типами посредством атрибута-ссылки, *связь тип/подтип* между подтипом и супертипом. Два элемента могут быть также соединены посредством композиции связей, называемых *путем*. Понятия связи и пути определяются следующим образом:

Определение 1 $X \rightarrow Y$ есть связь (простой путь), если имеет место один из ниже перечисленных случаев:

- Y является простым атрибутом (типом которого является встроенный тип данных) типа X .

Обозначение: $X \rightarrow Y$ (и $X \rightarrow Y$ для случая множественного атрибута).

Пример: $Proposal \rightarrow name$;

- тип X содержит атрибут-ссылку a , указывающий на тип Y .

Обозначение: $X-a \rightarrow Y$ (и $X-a \rightarrow Y$ для случая множественного атрибута).

Пример: $Proposal-leader \rightarrow Researcher$;

- тип X является подтипом (супертипом) типа Y .

Обозначение: $X \Rightarrow Y$ ($X \Leftarrow Y$).

Пример: $Researcher \Rightarrow Person$.

В дальнейшем будем использовать обозначение $X \rightarrow Y$, если не рассматриваем конкретный случай.

Определение 2 $X_1 \rightarrow X_2 \rightarrow \dots \rightarrow X_n$ есть путь, если каждая $X_i \rightarrow X_{i+1}$ является связью.

Обозначение: $X \dots Y$ - любой путь.

Пример: $Company-address \rightarrow Address \rightarrow city$.

Определение 3 Путь $X_1 \dots X_2$ является релевантным пути $Y_1 \dots Y_2$, если они удовлетворяют одному из правил структурных преобразований, введенных в диссертации.

При установлении релевантности путей рассматриваются только пути, удовлетворяющие следующим условиям:

Определение 4 Условия потенциальной релевантности путей

- полнота
Путь $X_1 \dots X_2$ может быть релевантным пути $Y_1 \dots Y_2$, только если X_1 является релевантным Y_1 , а X_2 – релевантным Y_2 .
- ацикличность

Путь $X_1...X_2$ может быть релевантным пути $Y_1...Y_2$, только если и путь $X_1...X_2$, и путь $Y_1...Y_2$ не содержат циклов.

- минимальность

Путь $X_1...X_2$ может быть релевантным пути $Y_1...Y_2$, если не существует подпути $X_3...X_4$ ($Y_3...Y_4$) пути $X_1...X_2$ ($Y_1...Y_2$) такой, что $X_3...X_4$ ($Y_3...Y_4$) релевантен произвольному пути $Y_5...Y_6$ ($X_5...X_6$).

Пример правила структурных преобразований:

Правило 1 Релевантность путей при использовании агрегированных типов

Путь $X_0...X_1$ релевантен пути $Y_0...Y_1$, если они удовлетворяют условию потенциальной релевантности, описанному выше, и они имеют следующий вид:

$$a) X_0 \rightarrow a_1 \rightarrow X_1 \sim Y_0(-b \rightarrow | \Rightarrow) * Y_3 \rightarrow b_1 \rightarrow Y_1$$

$$б) X_0 \rightarrow X_1 \sim Y_0(-b \rightarrow | \Rightarrow) * Y_3 \rightarrow Y_1$$

где $(-b \rightarrow | \Rightarrow) *$ - произвольный путь, содержащий только связи ассоциаций посредством атрибутов-ссылок и связи подтип/супертип.

При этом функции разрешения конфликтов имеют следующий вид:

a)

```
get_a1: {in: function;
  params: {+y/Y0, -returns/X1};
  {{ returns = y(.b)*.b1 }}
}
```

б)

```
get_X1: {in: function;
  params: {+y/Y0, -returns/X0.X1};
  {{ returns = y(.b)*.Y1 }}
}
```

где $(.b) *$ - список атрибутов-ссылок.

Пример:



```
get_in_street: {in: function;
  params: {+y/University,
    -returns/Organization.in_street};
  {{ returns = y.address.street }}
}
```

Вкратце, алгоритм поиска релевантных путей спецификаций требования и компонентов выглядит следующим образом. Для произвольной пары типов –

типа спецификации требований Ts_1 и типа компонента Tr_1 из списка онтологически релевантных типов:

- находится тип спецификации требований Ts_2 и тип компонента Tr_2 из списка онтологически релевантных типов, такие, что пути $Ts_1..Ts_2$ и $Tr_1..Tr_2$ удовлетворяют одному из правил структурных преобразований;
- если эксперт подтверждает релевантность найденных путей, они заносятся в список релевантных путей.

Представленные алгоритмы позволяют автоматизировать процесс поиска и разрешения структурных конфликтов между спецификациями требований и компонентов, что обеспечивает применимость подхода к большему числу компонентов, а также ускорение разработки ИС.

Глава 4

В четвертой главе представлен процесс конструирования и уточнения спецификаций типов, который является основой конструирования ИС из компонентов. Процесс конструирования основан на выявлении фрагментов спецификаций существующих компонентов и их дальнейшей композиции, уточняющей спецификацию требований. Для этого используются операции над типами, ведущими к трансформации их спецификаций – операции декомпозиции и композиции.

Процесс конструирования основан на понятии уточнения. *Уточняющие спецификации*, образуемые при конструировании, согласно теории уточнения, могут использоваться всюду *вместо уточняемых* спецификаций требований, так что пользователи не замечают этой замены. Методы уточнения позволяют формально устанавливать факт уточнения, гарантируя адекватность полученных спецификаций требуемым.

Основной операцией декомпозиции типов является операция взятия *редукта* типа.

Определение 5 Редукт R_T типа T определяется как подсигнатура сигнатуры типа, при этом множество атрибутов редукта является подмножеством множества атрибутов типа, множество функций редукта является подмножеством множества функций типа, множество предикатов редукта является подмножеством множества предикатов типа. Редукт R_T является супертипом типа T .

Редукты являются базисом для дальнейшей композиции с целью получения спецификации, уточняющей спецификацию требований.

Определение 6 Общий редукт для типов T_1, T_2 есть редукт R_{T_1} типа T_1 такой, что существует редукт R_{T_2} типа T_2 , при этом редукт R_{T_2} является уточнением редукта R_{T_1} . Редукт R_{T_2} называется сопряженным редуктом к общему редукту.

Определение 7 Наиболее общий редукт $R_{MC}(T_1, T_2)$ для типов T_1, T_2 есть редукт R_{T_1} типа T_1 такой, что существует редукт R_{T_2} типа T_2 который является уточнением редукта R_{T_1} и не существует другого редукта R'_{T_1} такого, что

$R_{MC}(T_1, T_2)$ является редуктом R'_{T_1} , R'_{T_1} не равен $R_{MC}(T_1, T_2)$ и существует R'_{T_2} являющийся уточнением редукта R'_{T_1} .

Понятие *наиболее общего редукта* является фундаментальным для этапа конструирования: оно составляет базис для определения повторно используемых фрагментов.

Вкратце, алгоритм конструирования наиболее общего редукта состоит в следующем. Для определения наиболее общих редуктов для каждой пары онтологически релевантных типов T_s и T_r требуется найти максимальный набор A пар атрибутов (a_{T_s}, a_{T_r}) , которые являются онтологически релевантными и имеют типы такие, что атрибут a_{T_r} можно использовать вместо a_{T_s} .

Другими словами, атрибут спецификации требований a_{T_s} типа T_s и релевантный ему атрибут компонента a_{T_r} типа T_r включаются в A , если тип атрибута a_{T_s} является супертипом типа атрибута a_{T_r} . Функция f_{T_s} типа T_s и релевантная ему функция f_{T_r} типа T_r включаются в A , если они имеют эквивалентные сигнатуры с точностью до отношения на типах параметров (контравариантность для входных параметров и ковариантность для выходных параметров). Атрибут a_{T_s} и функция f_{T_r} включаются в A , если f_{T_r} является функцией разрешения конфликтов для атрибута a_{T_s} .

Атрибуты (и функции) a_{T_s} формируют наиболее общий редукт $R(T_s, T_r)$, а атрибуты (и функции) a_{T_r} – конкретизирующий редукт $CR(T_s, T_r)$. Конкретизирующий редукт $CR(T_s, T_r)$ является сопряженным редуктом, и содержит, также, список соответствия между атрибутами редукта и атрибутами конкретизирующего редукта.

Пример спецификации редукта и конкретизирующего редукта выглядит следующим образом:

```
{R_Prop_Subm;
  in: reduct;
  metaslot
    of: Proposal;
    taking: {name, area, consortium, budget,
             starting_date, termination_date};
  c_reduct: CR_Prop_Subm;
  end
}
{CR_Prop_Subm;
  in: c_reduct;
  metaslot
    of: Submission;
    taking: {name, research_area, participates,
             req_money, starting_date, duration};
  reduct: R_Prop_Subm;
  end;
  simulating: {
```

```

    p.name ~ s.name,
    p.area ~ s.research_area,
    p.consortium ~ s.participates,
    p.budget ~ s.req_money,
    p.starting_date ~ s.starting_date,
    p.termination_date ~ f_termination_date
};
f_termination_date: {in: function;
  params: {+ext/CR_Prop_Subm, -returns/time};
  predicative: {ex s/Submission (s = ext/CR_Prop_Subm) &
    returns = s.starting_date+s.duration }}}
}

```

Слот *taking* содержит список атрибутов редуцируемого типа. Слот *simulating* содержит список соответствия между атрибутами/функциями редукта *R_Prop_Subm* и конкретизирующего редукта *CR_Prop_Subm*. Атрибут/функция конкретизирующего редукта уточняет соответствующий атрибут/функцию редукта. Функция *f_termination_date* – функция разрешения конфликтов.

Операции композиции типов позволяют определять новые типы из существующих. Основными операциями являются операции *meet* и *join*.

Определение 8 Операция *meet*. Операция $T_1 \sqcap T_2$ образует тип T как пересечение спецификаций типов-операндов. Тип T образуется слиянием двух наиболее общих редуктов типов T_1 и T_2 : $R_{MC}(T_1, T_2)$ и $R_{MC}(T_2, T_1)$.

Определение 9 Операция *join*. Операция $T_1 \sqcup T_2$ образует тип T как объединение спецификаций типов-операндов. Тип T образуется слиянием спецификаций типов T_1 и T_2 . Общие элементы спецификаций типов T_1 и T_2 включаются в результирующий тип только один раз. Общие элементы определяются посредством слияния сопряженных редуктов двух наиболее общих редуктов типов T_1 и T_2 : $R_{MC}(T_1, T_2)$ и $R_{MC}(T_2, T_1)$.

Пример задания типа *CT_Prop_Subm_Res* посредством операции *meet* над типами *R_Prop_Subm* и *R_Prop_Res*:

```

CT_Prop_Subm_Res =
  R_Prop_Subm[name, area, consortium, budget, leader,
    staff, technical_staff, starting_date,
    termination_date]
   $\sqcap$  R_Prop_Res[name, area, consortium, budget, leader,
    staff, technical_staff, starting_date,
    termination_date]

```

Основной задачей этапа конструирования является спецификация взглядов (композиционных типов), уточняющих классы (типы) спецификации требований. Выше было показано, как найти и специфицировать фрагменты компонентов, которые могут быть использованы в разрабатываемой системе. Используя полученные фрагменты (в виде наиболее общих редуктов) на самом

нижнем уровне, конструируется иерархия взглядов (композиционных типов) посредством композиции классов/взглядов (типов/редуктов) более низких уровней соответственно. При этом конечный взгляд (композиционный тип) должен уточнять соответствующий класс (тип) спецификации требований. На Рис. 2 показана иерархия спецификаций для случая двух классов (справа) и двух типов (слева) компонентов.

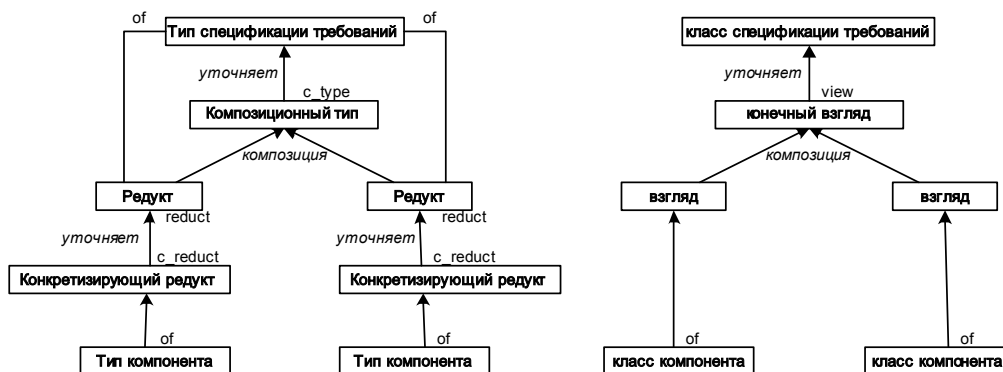


Рис. 2 Иерархия спецификаций композиционных типов и классов

Пример спецификации взгляда *v_prop_subm_res* на основе операции объединения классов *v_prop_subm* и *v_prop_res*:

```
{<v_prop_subm_res>;
  in: class;
  metaslot
    view_gen:{ in: function;
      params: {-returns/v_prop_subm_res as_class};
      enforcement: on_access;
      {{ v_prop_subm(s) | v_prop_res(r)
      }}
    }
  end;
  instance_section: CT_Prop_Subm_Res;
  class_section: {
    key: invariant, {unique; {name}};
  }
}
```

Вкратце, алгоритм конструирования уточняющих спецификаций выглядит следующим образом. В зависимости от вида разрабатываемой системы применяются различные сценарии конструирования уточняющих спецификаций.

При реализации программных систем задаются только спецификации типов. Композиция редуктов строится таким образом, чтобы покрыть как можно больше атрибутов типа спецификации требований. Если при этом некоторые атрибуты остаются не поддержанными, создается новый тип,

содержащий такие атрибуты. В дальнейшем, разработчик должен реализовать этот тип. Композиция редуктов осуществляется посредством операции *join*.

При разработке информационных систем главными являются классы. В зависимости от требований при композиции используются такие операции как, например, операции объединения, пересечения или соединения классов. При *объединении* классов тип экземпляра конструируемого взгляда образуется посредством операции *meet*, применяемой к типам экземпляров классов-операндов. При *пересечении (соединении)* классов тип экземпляра конструируемого взгляда получается посредством операции *join*, применяемой к типам экземпляров классов-операндов.

Описанные алгоритмы обеспечивают корректность композиции повторно используемых фрагментов компонентов в спецификацию, уточняющую спецификацию требований.

Глава 5

В пятой главе представлен инструментарий эксперта-конструктора ИС, включающий программные средства, реализующие разработанные в диссертационной работе алгоритмы конструирования ИС из компонентов, описана архитектура инструментария программных средств и его составных частей, определены пользовательские интерфейсы. На Рис.3 показана архитектура инструментария.

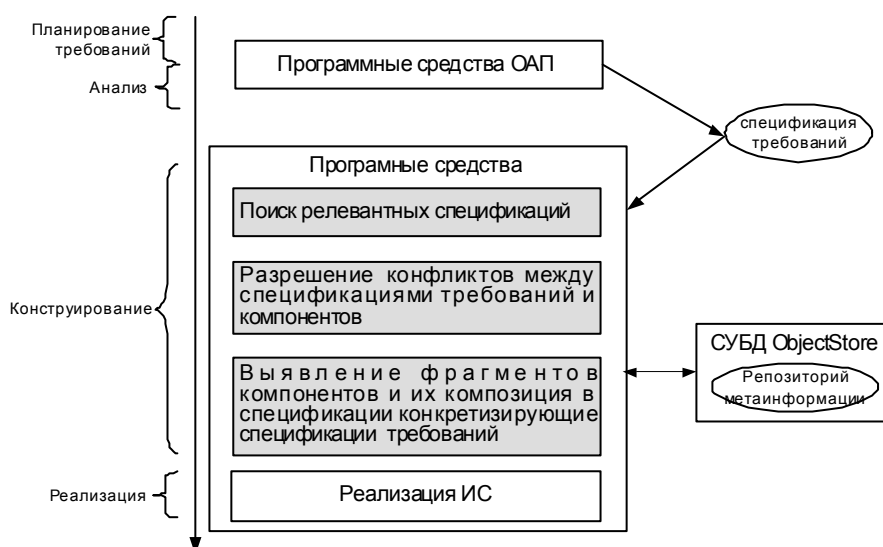


Рис. 3 Архитектура инструментария эксперта-конструктора ИС

Этапы планирования требований и анализа реализуются с помощью применения произвольного метода ОАП, использующего нотацию UML. Выбранный метод ОАП расширяется онтологическими спецификациями, спецификациями функций и инвариантов, и другими понятиями языка

СИНТЕЗ. Спецификации требований разрабатываемой ИС, полученные на этапах планирования требований и анализа при помощи произвольного метода ОАП, загружаются в репозиторий метаянформации.

Репозиторий метаянформации служит для хранения спецификаций в канонической модели, формируемых в процессе конструирования. Репозиторий метаянформации может содержать спецификации, отражающие онтологический, структурный и поведенческий аспекты сущностей, используемых на этапе конструирования.

Репозиторий метаянформации реализован в объектной СУБД ObjectStore. Доступ к нему осуществляется при помощи Java-интерфейсов.

Серым цветом выделены те программные средства, входящие в состав инструментария, которые реализуют алгоритмы конструирования ИС из компонентов, представленные в диссертационной работе. Эти программные средства реализованы автором на языке Java 2 в среде Windows.

Инструментарий реализован в виде интерактивной системы, позволяющей вовлечь эксперта в процесс конструирования ИС. Эксперт взаимодействует с инструментарием с помощью графических интерфейсов.

Разработанный инструментарий позволяет автоматизировать процесс конструирования ИС из существующих компонентов и может быть применен для создания ИС в различных прикладных областях, например, в ИС для научных исследований, электронных библиотеках для образования, интеллектуальных системах управления и принятия решений, и других.

ЗАКЛЮЧЕНИЕ

Основные результаты работы сводятся к следующему:

- предложен подход к конструированию ИС из компонентов, технически интероперабельных в рамках некоторого промежуточного слоя;
- разработаны алгоритмы поиска элементов спецификаций компонентов релевантных элементам спецификации требований, на основе онтологической модели;
- разработаны алгоритмы разрешения конфликтов между спецификациями требований и компонентов;
- разработаны алгоритмы выявления фрагментов спецификаций компонентов, которые могут служить уточнением соответствующих фрагментов спецификации требований;
- разработаны алгоритмы построения композиции таких фрагментов в спецификацию, уточняющую спецификацию требований;
- на основе перечисленных алгоритмов разработан инструментарий эксперта-конструктора ИС.

ПУБЛИКАЦИИ

Основные результаты диссертации опубликованы в следующих работах:

- [1] Briukhov D.O., Shumilov S.S. Ontology Specification and Integration Facilities in a Semantic Interoperation Framework // In *Proc. of the International Workshop ADBIS'95*, Springer, 1995, pp. 195-200
- [2] Брюхов Д.О., Задорожный В.И., Калиниченко Л.А., Курошев М.Ю., Шумилов С.С. Интероперабельные информационные системы: архитектуры и технологии // *Системы Управления Базами Данных* # 4/95, 1995, с. 96-113
- [3] Briukhov D.O. Interfacing of Object Analysis and Design Methods with the Method for Interoperable Information Systems Design // In *Proc. of the Third International Workshop ADBIS'96*, Moscow, MEPHI, 1996, pp. 165-170
- [4] Briukhov D.O., Kalinichenko L.A. Component-Based Information Systems Development Tool Supporting the SYNTHESIS Design Method // In *Proc. of the East European Symposium on Advances in Databases and Information Systems*, Poland, Springer, LNCS No.1475, 1998, pp. 305-327
- [5] Брюхов Д.О., Калиниченко Л.А., Скворцов Н.А. Композиционное проектирование информационных систем: методы и средства // *Системы и средства информатики. Вып.10*, М.: Наука, 2000, с. 128-147
- [6] Briukhov D.O., Kalinichenko L.A., Skvortsov N.A. Personalization through Specification Refinement and Composition // In *Proc. of the Joint DELOS-NSF Workshop on Personalisation and Recommender Systems in Digital Libraries*, Ireland, DCU, 2001, pp. 43-48
- [7] Briukhov, D.O., Kalinichenko, L.A., Skvortsov, N.A. Information sources registration at a subject mediator as compositional development // In *Proc. of the Fifth East European Symposium on Advances in Databases and Information Systems (ADBIS'01)*, Springer-Verlag, LNCS No.2151, 2001, pp. 70-83
- [8] Briukhov, D.O., Kalinichenko, L.A., Tyurin, I.N. Extension of Compositional Information Systems Development for the Web Services Platform // In *Proc. of the Seventh East European Symposium on Advances in Databases and Information Systems (ADBIS'03)*, Springer-Verlag, LNCS No.2798, 2003, pp. 16-29