ПОСТРОЕНИЕ МОДЕЛИ РАСПРЕДЕЛЕННОГО ХРАНИЛИЩА ДАННЫХ С ИСПОЛЬЗОВАНИЕМ ОБЪЕКТНО-ОРИЕНИТРОВАННОГО ПОДХОДА

Лаздынь С. В, Петров А. В.
Донецкий национальный технический университет
кафедра автоматизированных систем управления
e-mail: slazd@ukr.net, petrovolexandr@gmail.com

Abstract
Lazdyn S. V, Petrov A. V. Building of a distributed data warehouse model using object-oriented approach*. In the article given the new object model of distributed data warehouse is considered. The typical components are selected and their object models are developed. The common object model of distributed data warehouse is developed by setting cooperation between object models of typical components.*

Характеристика проблемы

Распределенные хранилища данных (РХД) необходимы для хранения и обеспечения быстрого доступа к данным в условиях различных организаций, филиалы которых удалены друг от друга. В современном мире технологии распределенных хранилищ данных нашли широкое применение в информационных системах банков, правоохранительных органов, межрегиональных и международных организаций.

Важной задачей является создание модели распределенного хранилища данных достоверно отображающей распределенное хранилище данных. При практической разработке РХД возможность проведения предварительного анализа разрабатываемого РХД на модели позволяет выявить и устранить слабые места и исправить возможные недостатки РХД и тем самым повысить эффективность создаваемого хранилища данных.

Краткий анализ проведенных исследований по моделированию РХД.

Вопросом построения моделей хранилищ данных посвящены ряд научных работ и публикаций [1–6]. Проведенный анализ показал, что в настоящее время существуют следующие основные модели хранилищ данных: сетчатая модель [1], ER-модель [2, 3], UML-модель [4], AsmL-модель [5].

Сетчатая модель. Сетчатая модель [1] основана на том, что в хранилище данных значения многих записей в представлениях вычислимы по значениям записей в других представлениях в зависимости от уровня детализации. Сама модель представляет ориентированный граф, вершины которого представляют собой материализованные представления с указанием измерений, по которым проводится группировка и количества записей в представлении, а дуги – взаимосвязи между представлениями. Данная модель обладает рядом недостатков: в частности, данная модель строится на основе описания взаимосвязей материализованных представлений, а такие важные компоненты хранилища данных как таблицы фактов и измерений не моделируются, не учитывается возможность распределенности хранилища данных, не учитываются частоты возникновения запросов, а также возможность вертикальной фрагментации таблицы фактов.

ER-модель. Реляционная ER-модель, предложенная Питером П. Ченом [2]. Была расширена Е. Франкони и А. Камбле для моделирования хранилищ данных [3]. Помимо сущностей и связей, присущих классической ER-модели, были введены простые и многомерные агрегированные сущности и материализованные представления. Данная модель принципиально не отличается от стандартной ER модели и включает все её недостатки и преимущества. В частности в данной модели невозможно определить связи между атрибутами измерений, что влияет на количество возможных материализованных представлений.

Также данная ER-модель не позволяет создать модель распределенного хранилища данных, является статичной, то есть отсутствует возможность промоделировать работу хранилища данных. Также в данной модели нет возможности моделирования хранилища данных с учетом наиболее часто возникающих запросов.

UML-модель. При моделировании хранилища данных для представления хранилищ данных используется совокупность UML-диаграмм на трёх уровнях: концептуальном, физическом и логическом. Данная модель проста в реализации и наглядно представляет структуру хранилища данных [4]. Недостатками данной модели являются отсутствие учета распределенности хранилища данных и статичность, модель не отображает динамику работы хранилища данных.

AsmL-модель. Данная модель описывает хранилище данных на языке AsmL и позволяет моделировать как стационарные, нераспределенные хранилища данных на основе стандартной схемы звезда, так и распределенные хранилища данных. РХД в данной модели представляется в виде конечного автомата. При моделировании РХД с помощью AsmL такие его компоненты как таблицы фактов, измерений, материализованные представления представляются в виде состояний системы. Запросы с вертикальной и горизонтальной фрагментацией представляются переходами системы [5]. Данная модель достаточно полно описывает основные компоненты логической архитектуры РХД, такие как измерения, таблицы фактов, материализованные представления, запросы. Вместе с тем в модели не отражены компоненты физической архитектуры: сервера, каналы связи, сетевые устройства и т.п.

Ни одна из рассмотренных моделей не может описывать распределенное хранилище данных с учетом всех факторов влияющих на его эффективность. Так во всех моделях не учитывается влияние компонентов физической архитектуры, кроме того, все эти модели являются статичными и не позволяют производить динамическое моделирование хранилища данных. Также сетчатая модель, ER-модель и UML-модель не учитывают распределение данных по узлам компьютерной сети и не могут быть использованы для моделирования РХД.

Предлагаемый подход к построению модели РХД

Ранее авторами был предложен объектно-ориентированный подход к построению модели РХД [6]. Данный подход состоит в следующем: проводится структурный анализ распределенного хранилища данных и выделяются его типовые компоненты, для типовых компонентов разрабатываются объектные модели в виде классов. Общая объектная модель распределенного хранилища данных, строится как система взаимодействующих объектных моделей типовых компонентов РХД.

В распределенном хранилище данных можно выделить две группы компонентов: компоненты логической архитектуры и компоненты физической архитектуры.

К компонентам логической архитектуры относятся все компоненты хранилища данных связанные с данными, и не связанные с технической стороной функционирования хранилища. В качестве компонентов логической архитектуры были выделены следующие типовые компоненты:

1. Измерение – представляет информацию о фактах;
2. Фрагмент таблицы фактов и фрагменты таблицы измерений – части соответственно таблицы фактов и таблицы измерений, хранящиеся независимо от остальных частей и основной таблицы;
3. Материализованное представление – заранее вычисленный результат запроса, используемый для ускорения выполнения часто вызываемых запросов;
4. Запрос на вставку, запрос на выборку, запрос на обновление – запросы, выполняющиеся в хранилище данных;

5. Узел хранилища данных – структурный компонент хранилища данных объединяющий в себе серверы, фрагменты таблиц фактов и измерений, материализованные представления, входящие и исходящие каналы связи, территориально находящиеся в одном месте;

6. Приложение – выполняющийся на серверах процесс, служащий источником запросов;

7. Пользователь – человек, обращающийся через приложения к серверам и являющийся инициатором запуска приложений и получателем результатов запросов.

К компонентам физической архитектуры относятся те компоненты хранилища данных, которые отвечают за техническую сторону функционирования РХД. В данной модели к типовым компонентам физической архитектуры были отнесены следующие типовые компоненты:

8. Сервер – компонент предоставляющий ресурсы для хранения данных и выполнения запросов;

9. Канал связи – компонент предоставляющий ресурсы для передачи данных между узлами.

Объектные модели типовых компонентов РХД

Для всех классов, описывающих компоненты введен общий родительский класс, содержащий свойства общие для всех объектов: уникальный идентификатор, наименование. Класс, представляющий данную объектную модель, носит имя TBaseComp и содержит следующие поля: ID – уникальный идентификатор объекта и Name – наименование объекта.

Для типовых компонентов представляющих наборы данных: «Фрагмент таблицы фактов», «Фрагмент таблицы измерения», «Материализованное представление» введен общий родительский класс, содержащие следующие, общие для этих компонентов свойства: уникальный идентификатор, наименование, список местоположений, размер записи, количество записей. Под списком местоположений понимается список всех серверов, на которых встречается данный набор данных. Данный класс носит имя TDataFragment и является потомком класса TBaseComp, от которого наследует поля ID – уникальный идентификатор объекта и Name – наименование объекта. Кроме того, данный класс содержит следующие поля: Locations – список местоположений, RecordSize – размер записи, RecordCount – количество записей.

Для типовых компонентов представляющих запросы: «Запрос на выборку» и «Запрос на вставку» введен общий родительский класс, содержащий следующие, общие для этих компонентов свойства: уникальный идентификатор, наименование, таблица фактов, к которой происходит обращение в запросе, количество записей из таблицы фактов, с которым оперирует запрос, список подзапросов порождаемых данных запросом, а также количество транзакций необходимых для выполнения запроса. Данный класс носит имя TDWQuery и является дочерним классом для TBaseComp, от которого наследует следующие поля: ID – уникальный идентификатор, Name – наименование. Кроме того, данный класс содержит собственные поля: FactTable – фрагмент таблицы фактов, к которой обращается запрос, SizeFact – количество записей, отбираемых запросом, SubQueries – список подзапросов, формируемых данным запросом, TransCount – количество транзакций необходимых для выполнения запроса. Также данный класс содержит виртуальный метод Execute – моделирующий выполнение запроса.

Объектные модели указанных выше типовых компонентов РХД представляют собой классы, производные от базового класса TBaseComp и классов TDataFragment, TDWQuery. UML-диаграмма иерархии классов-объектных моделей представлена на рис. 1.

Рассмотрим подробное описание классов для отдельных компонентов РХД.

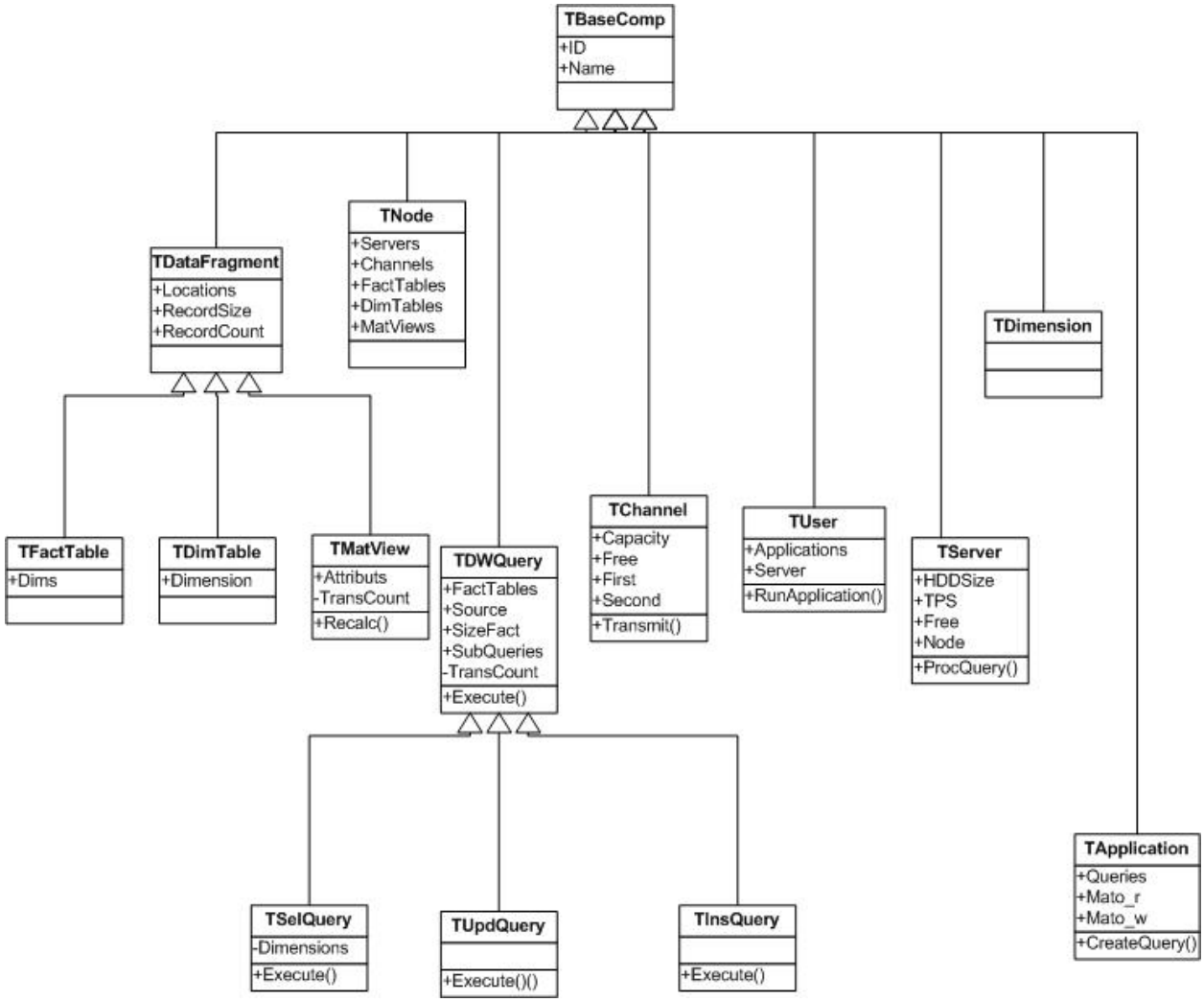

Рисунок 1 – UML-диаграмма иерархии классов объектной модели РХД

Объектная модель типового компонента «Измерение». Данный типовой компонент описывает измерение хранилища данных. В состав типового компонента «Измерение» входят следующие свойства: уникальный идентификатор, наименование. Класс, представляющий данную объектную модель, носит имя TDimension и является потомком класса TBaseComp, от которого наследует следующие поля: ID – уникальный идентификатор объекта и Name – наименование объекта.

Объектная модель типового компонента «Фрагмент таблицы фактов». В состав объектной модели типового компонента «Фрагмент таблицы фактов» входят следующие свойства: уникальный идентификатор, наименование, список местоположений, размер записи, количество записей, список измерений. Класс, представляющий данную объектную модель, носит имя TFactTable, является потомком класса TDataFragment и наследует от него следующие свойства: ID – уникальный идентификатор, Name – наименование, Locations – список местоположений, RecordSize – размер записи, RecordCount – количество записей, а также содержит собственное свойство Dims – список измерений связанных с данным фрагментом таблицы фактов.

Объектная модель типового компонента «Фрагмент таблицы измерения». В состав объектной модели типового компонента «Фрагмент таблицы измерения» входят следующие свойства: уникальный идентификатор, наименование, список местоположений, размер записи, количество записей, указатель на измерение. Класс, представляющий данную

объектную модель, носит имя TDimTable, является потомком класса TDataFragment и наследует от него следующие свойства: ID – уникальный идентификатор, Name – наименование, Locations – список местоположений, RecordSize – размер записи, RecordCount – количество записей, а также содержит собственное свойство Dimension – указатель на измерение, представляемое данным фрагментом таблицы измерения.

Объектная модель типового компонента «Материализованное представление». В состав типового компонента «Материализованное представление» входят следующие свойства: уникальный идентификатор, наименование, список местоположений, размер записи, количество записей, список представляемых фрагментов таблицы фактов, а также список измерений, по которым группируются данные в данном представлении. Класс, представляющий данную объектную модель, носит имя TMatView, является потомком класса TDataFragment и наследует от него следующие свойства: ID – уникальный идентификатор, Name – наименование, Locations – список местоположений, RecordSize – размер записи, RecordCount – количество записей, а также содержит собственные свойства: FactTable – список представляемых фрагментов таблицы фактов, Dimensions – список измерений, по которым группируются данные в данном материализованном представлении, TransCount – количество транзакций требуемых для обновления, MatViews – список представлений, использующих данное представление.

Данный класс содержит метод Recalc() моделирующий пересчет материализованного представления. Этот метод возвращает время, затраченное на пересчет материализованного представления. При расчете времени вызывается метод ProcQuery() класса TServer, которому в качестве параметра передается количество транзакций требуемых для обновления материализованного представления. Также инициируются обновления всех представлений использующих данное представление.

Объектная модель типового компонента «Запрос на выборку». В состав объектной модели типового компонента «Запрос на выборку» входят следующие свойства: уникальный идентификатор, наименование, таблица фактов, к которой обращается запрос, а также список измерений, по которым проводится группировка, количество записей отбираемых из таблицы фактов, список подзапросов порождаемых данных запросом, а также количество транзакций необходимых для выполнения запроса. Класс, описывающий данный типовой компонент носит имя TSelQuery и является наследником класса TDWQuery. Класс TSelQuery наследует от класса TDWQuery следующие поля: ID – уникальный идентификатор объекта, Name – наименование, FactTable – указатель на фрагмент таблицы фактов, к которому происходит обращение, Dimensions – список измерений, по которым производится группировка, SizeFact – количество отбираемых записей из таблицы фактов, SubQueries – список формируемых подзапросов. Также в состав данного класса входит метод Execute() выполняющий моделирование выполнения запроса.

Время выполнения запроса определяется соотношением:

$$T_{eq} = t_s + t_t + \sum_{i=1}^{N} \left(t_{si} + t_{ti} \right), \qquad (1)$$

где t_s – время выборки данных из основного фрагмента, t_t –время передачи данных по каналам связи, t_{si} – время выборки данных при выполнении i-го подзапроса, t_{ti} – время передачи данных по каналам связи при выполнении i-го подзапроса N – количество подзапросов.

Объектная модель типового компонента «Запрос на вставку». В состав объектной модели типового компонента «Запрос на вставку» входят следующие свойства: уникальный идентификатор, наименование, таблица фактов, к которой обращается запрос, с указанием количества вставляемых записей, список подзапросов порождаемых данных запросом, а также количество транзакций необходимых для выполнения запроса. Класс, описывающий данный типовой компонент носит имя TInsQuery и является наследником класса TDWQuery. Класс TInsQuery наследует от класса TDWQuery следующие поля: ID – уникальный идентифи-

фикатор объекта, Name – наименование, FactTable – указатель на фрагмент таблицы фактов, к которому происходит обращение, Dimensions – список измерений, по которым производится группировка, SizeFact – количество отбираемых записей из таблицы фактов, SubQueries – список формируемых подзапросов. Также в состав данного класса входит метод Execute() выполняющий моделирование выполнения запроса. Время выполнения запроса определяется соотношением:

$$T_{eq} = t_i + t_t + \sum_{i=1}^{M} t_{ri} + \sum_{j=1}^{N} \left(t_{ij} + t_{tj} \right), \qquad (2)$$

где t_i – время вставки данных в основной фрагмент, t_t –время передачи данных по каналам связи, t_{ri} – время пересчета i-го материализованного представления, использующего данные из этого фрагмента данных, t_{ij} – время вставки данных при выполнении j-го подзапроса, t_{tj} – время передачи данных по каналам связи при выполнении j-го подзапроса N – количество подзапросов, M – количество материализованных представлений использующих данные из данного фрагмента таблицы фактов.

Объектная модель типового компонента «Запрос на обновление». В состав объектной модели типового компонента «Запрос на обновление» входят следующие свойства: уникальный идентификатор, наименование, фрагмент таблицы фактов, к которому обращается запрос, с указанием количества обновляемых записей, список подзапросов порождаемых данных запросом, а также количество транзакций необходимых для выполнения запроса. Класс, описывающий данный типовой компонент носит имя TUpdQuery и является наследником класса TDWQuery. Класс TUpdQuery наследует от класса TDWQuery следующие поля: ID – уникальный идентификатор объекта, Name – наименование, FactTable – указатель на фрагмент таблицы фактов, к которому происходит обращение, Dimensions – список измерений, по которым производится группировка, SizeFact – количество отбираемых записей из таблицы фактов, SubQueries – список формируемых подзапросов.

В состав данного класса входит метод Execute() выполняющий моделирование выполнения запроса. Время выполнения запроса определяется соотношением.

$$T_{eq} = t_u + t_t + \sum_{i=1}^{M} t_{ri} + \sum_{j=1}^{N} \left(t_{uj} + t_{tj} \right), \qquad (3)$$

где t_u – время обновления данных в основном фрагменте, t_t –время передачи данных по каналам связи, t_{ri} – время пересчета i-го материализованного представления, использующего данные из этого фрагмента данных, t_{uj} – время выборки данных при выполнении j-го подзапроса, t_{tj} – время передачи данных по каналам связи при выполнении j-го подзапроса N – количество подзапросов, M – количество материализованных представлений использующих данные из данного фрагмента таблицы фактов.

Объектная модель типового компонента «Узел хранилища данных». В состав объектной модели типового компонента «Узел хранилища данных» входят следующие свойства: уникальный идентификатор, наименование, списки серверов, каналов связи, таблиц фактов, измерений, материализованных представлений. Класс, описывающий данный типовой компонент носит имя TNode и является потомком класса TBaseComp. Класс TNode наследует от класса TBaseComp следующие поля: ID – уникальный идентификатор, Name наименование, а также содержит собственные поля: Servers – список серверов, Channels – список присоединённых каналов, FactTables – список фрагментов таблицы фактов, DimTables – список фрагментов таблицы измерений, MatViews – список материализованных представлений.

Объектная модель типового компонента «Приложение». В состав объектной модели типового компонента «Приложение» входят следующие свойства: уникальный идентификатор, наименование, списки порождаемых запросов, математическое ожидание событий запуска приложения, а также математическое ожидание времени работы

приложения. Класс, описывающий данный типовой компонент носит имя TApplication и является потомком класса TBaseComp. Класс TApplication наследует от класса TBaseComp следующие поля: ID – уникальный идентификатор, Name наименование, а также содержит собственные поля: Queries – список запросов, инициируемых данным приложением, Mato_R – математическое ожидание события запуска приложения, Mato_W – математическое ожидание времени работы приложения.

В состав данного класса входит метод CreateQuery() – выполняющий создание запроса. Этот метод создает запрос и добавляет его в глобальную очередь запросов.

Объектная модель типового компонента «Пользователь». В состав объектной модели типового компонента «Пользователь» входят следующие свойства: уникальный идентификатор, наименование, списки серверов, каналов связи, таблиц фактов, измерений, материализованных представлений. Класс, описывающий данный типовой компонент носит имя TUser и является потомком класса TBaseComp. Класс TUser наследует от класса TBaseComp следующие поля: ID – уникальный идентификатор, Name наименование, а также содержит собственные поля: Applications – список приложений, используемых данным пользователем, Server – сервер, к которому подключается данный пользователь. Данный класс содержит метод RunApplication(), моделирующий запуск приложения. При этом рассчитываются момент запуска и продолжительность работы приложения. Время завершения работы приложения определяется соотношением:

$$T_b = T_{\tilde{n}} + A_b, \tag{4}$$

где T_b – время запуска приложения, T_c – текущий момент времени, A_b – равномерно распределенная случайная величина с математическим ожиданием равным математическому ожиданию времени запуска приложения. Время завершения работы приложения определяется соотношением:

$$T_w = T_b + A_w, \tag{5}$$

где T_w – продолжительность приложения, T_b – момент запуска приложения, A_w – равномерно распределенная случайная величина с математическим ожиданием равным математическому ожиданию продолжительности работы приложения.

Объектная модель типового компонента «Сервер». В состав типового компонента «Сервер» входят следующие свойства: уникальный идентификатор, наименование, объем жесткого диска, производительность сервера, время освобождения, очередь выполнения запросов. Класс, описывающий данный типовой компонент носит имя TServer и является потомком класса TBaseComp. Класс TServer наследует от класса TBaseComp следующие поля: ID – уникальный идентификатор, Name наименование, а также содержит собственные поля: HDDSize – размер жесткого диска, TPS – производительность сервера, Free – время освобождения сервера, Queue – очередь выполнения запросов.

Данный класс содержит метод ProcQuery(), моделирующий выполнение запроса на сервере. Этот метод получает в качестве параметра указатель на объект класса TDWQuery и возвращает время выполнения запроса. Время выполнения запроса определяется из соотношения:

$$T_Q = \frac{Tr \cdot 60}{TPS}, \tag{6}$$

где T_Q – время выполнения запроса, Tr – количество транзакций при выполнении запроса, TPS – производительность сервера.

Объектная модель типового компонента «Канал связи». В состав типового компонента «Канал связи» входят следующие свойства: уникальный идентификатор, наименование, пропускная способность, время освобождения, указатели на серверы, находящиеся на концах канала связи, очередь передачи. Класс, описывающий данный типовой компонент носит имя TChannel и является потомком класса TBaseComp. Класс TChannel наследует от

класса TBaseComp следующие поля: ID – уникальный идентификатор, Name наименование, а также содержит собственные поля: Capacity – пропускная способность, Free – время освобождения канала, First – указатель на первый узел, Second – указатель на второй узел, Queue – очередь передачи. Данный класс содержит метод Transmit(), моделирующий передачу данных по каналу связи. Этот метод получает в качестве параметра сведения об объеме передаваемых данных и возвращает время, затраченное на передачу.

$$T_t = \frac{V}{C}, \tag{5}$$

где T_t – время передачи данных, V – объем передаваемых данных, C – пропускная способность канала связи.

Общая модель РХД

Для объединения объектных моделей в единое целое и управления процессом моделирования был введен еще один объект, который содержит методы, которые работают со всем хранилищем данных в целом. Класс, описывающий данный объект носит имя TDataWarehouse, и содержит свойства State – состояние хранилища данных (0 – моделирование, 1 – пауза моделирования, 2 – моделирование завершено), списки всех типовых компонентов РХД, очередь выполняемых запросов, список событий системы, а также содержит следующие методы: Initialize() – инициализация модели РХД, StartModeling() – старт моделирования, StopModeling() – останов моделирования, PauseModeling() – пауза моделирования, AddEvent() – добавление события в список событий, AddQuery() – добавление запроса в очередь запросов. Объект TDataWarehouse формирует списки объектов типовых компонентов РХД на основании исходных данных с помощью метода Initialize(), управляет процессом моделирования получая данные о запросах от приложений и добавляя их в глобальную очередь запросов с помощью метода AddQuery(), а также протоколирует работу РХД занося информацию обо всех событиях в модели в список событий с помощью метода AddEvent(). Схема объединения объектных моделей типовых компонентов с помощью объекта класса TDataWarhouse представлена на рисунке 2.

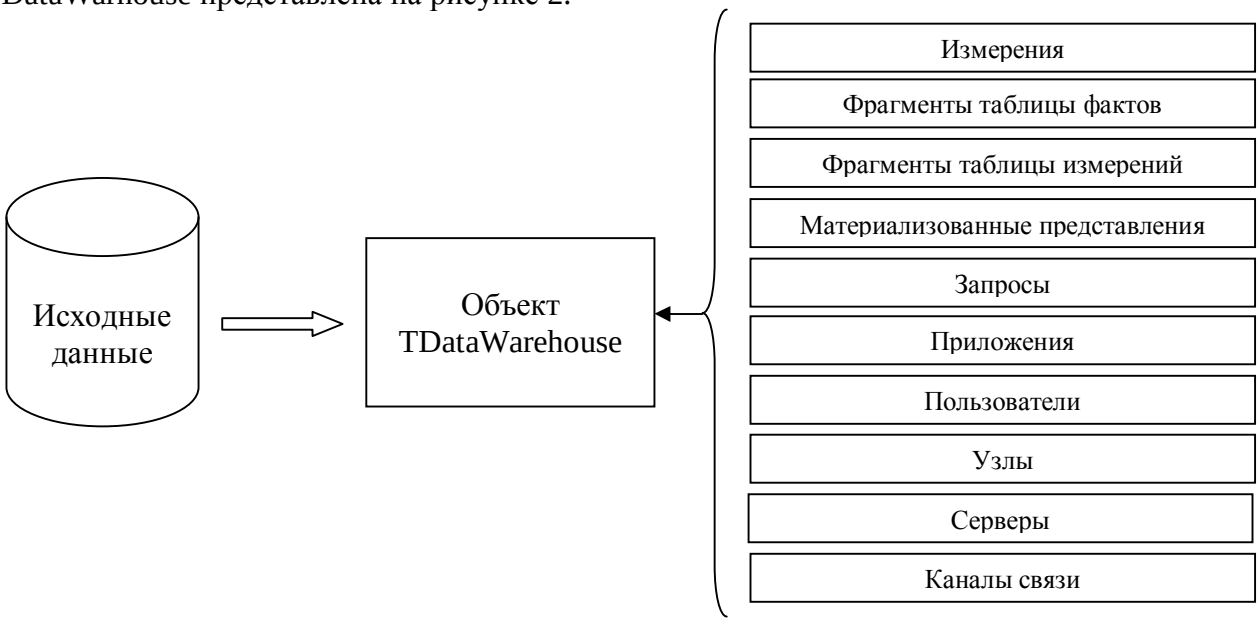

Рисунок 2 – Объединение моделей типовых компонентов с помощью объекта TDataWarehouse.

Для хранения очереди запросов и приложений введен новый объект, описывающий событие в системе. Данный объект обладает следующими свойствами: уникальный иденти-

фикатор, наименование события, указатель на объект породивший событие (запрос или приложение), время возникновения события. Данная объектная модель представляется классом TEvent, который содержит следующие поля: ID – уникальный идентификатор, Name – наименование, Parent – источник события, Time – время возникновения события.

Общая модель РХД строится как система взаимодействующих между собой объектных моделей типовых компонентов РХД. UML-диаграмма взаимодействия объектов показана на рисунке 3. На диаграмме представлены объекты модели и определены связи между объектами. В ходе функционирования распределенного хранилища данных главным процессом является процесс последовательного выполнения запросов. Рассмотрим взаимодействие объектов модели при выполнении запроса.

Главной целью создания хранилищ данных является обеспечение пользователям быстрого доступа к аналитической информации. Пользователи взаимодействуют с РХД посредством запуска приложений. Приложение, в свою очередь, выполняется на сервере и может формировать запросы к хранилищу данных. При формировании запроса, запрос передается на узел хранилища данных обрабатывающий данный запрос. Для передачи используется канал связи. Если канал связи свободен, то сразу выполняется передача запроса, в противном случае происходит постановка запроса в очередь передачи канала связи и ожидание освобождения канала. Как только канал освобождается, происходит передача запроса обрабатывающему узлу. В качестве обрабатывающего узла выбирается ближайший узел к узлу источнику запроса содержащий необходимые данные. В случае если требуется выборка данных из более чем одного узла РХД – формируются подзапросы и их результаты передаются на обрабатывающий узел. Обрабатывающий узел обращается к серверу, хранящему необходимый фрагмент таблицы или материализованное представление, сервер в зависимости от типа запроса осуществляет выборку/вставку/обновление в фрагменте таблицы и/или материализованном представлении, а также в случае необходимости осуществляет необходимые агрегационные вычисления. Узел РХД передает результат выполнения запроса, используя каналы связи, на узел-источник запроса.

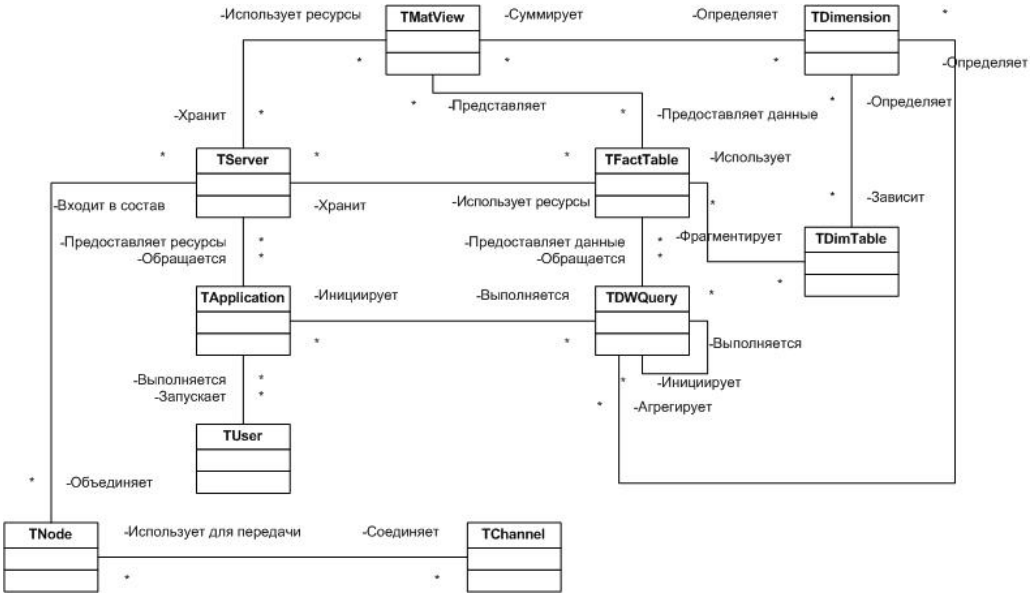

Рисунок 3 – Схема взаимодействия объектных моделей типовых компонентов РХД

В случае, если запрос выполняется полностью на узле источнике запроса, не происходит передачи данных по каналам связи. Диаграмма последовательности выполнения запроса представлена на рисунке 4.

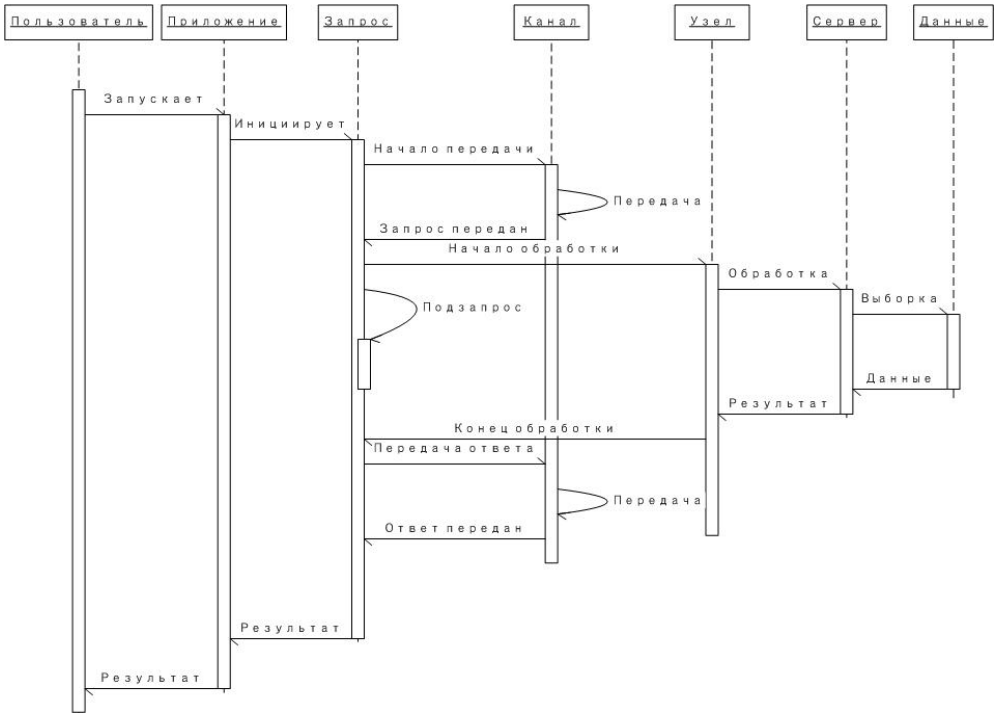

Рисунок 4 – Диаграмма последовательности процесса выполнения запроса

Заключение

Разработана новая объектная модель распределенного хранилища данных, учитывающая специфические свойства хранилища данных, распределенного по различным узлам компьютерной сети. Для этого были разработаны объектные модели в виде классов на языке программирования C++ для основных компонентов физической и логической архитектуры распределенного хранилища данных. Модель РХД построена как система взаимодействующих объектных моделей её компонентов. Новая модель распределенного хранилища данных отражает основные процессы, связанные с выполнением запросов, в распределенном хранилище данных. Данная модель позволяет проводить исследование эффективности распределенного хранилища данных на стадии разработки и анализировать время выполнения запросов на выборку, вставку и обновление, загруженность серверов и каналов.

Литература

1. Harinarayan V., Rajaraman A., Ullman J. D. Implementing Data Cube Efficiently.// ACM SIGMOD international conference on Management of data, 1996, Page(s) 118 – 127;

2. Peter Pin-Shan Chen. The Entity-Relationship Model — Toward a Unified View of Data, In: ACM Transactions on Database Systems, 1(1), March, 1976 Page(s) 9 – 36;

3. Enrico Franconi and Anand Kamble. A Data Warehouse Conceptual Data Model // Scientific and Statistical Database Management, 2004. Proceedings. 16th International Conference on Volume, Issue 21-23, June 2004, Page(s): 435 – 436;

4. LujanMora S., Trujillo J., Physical Modelling of Data Warehouses using UML// *DOLAP'04*, Washington, DC, USA, November, 2004, Page(s): 12–13.

5. Shewe K. D., Zhao J. Balancing Redundancy and Query Costs in Distributed Data Warehouses.// Proceedings of the 2nd Asia-Pacific conference on Conceptual modelling - Volume 43, APCCM '05, Wollongong, New South Wales, Australia, 2005, Page(s) 82–101;

6. Лаздынь С. В, Петров А. В. Разработка объектной модели распределенного хранилища данных.// Наукові праці Донецького національного технічного університету. Серія: "Обчислювальна техніка та автоматизація". Випуск 107 / Редкол.: Башков Є. О. (голова) та ін. – Донецьк: ДонНТУ, 2006. – с.122-128.