кие средства. Х.: Вища шк. Изд-во при Харьк. ун-те, 1984. 144 с. **4.** *Солонская С.В.* Возможности использования алгебры предикатов для классификации воздушных объектов по радиолокационному спектральному изображению// Радиотехника. 2004. Вып. 139. С.73-76.

Поступила в редколлегию 28.06.05

Рецензент: д-р техн. наук, проф. Шабанов-Кушнаренко С.Ю.

Жирнов Владимир Витальевич, канд. техн. наук, вед. научн. сотрудник НИЦ КВ КП ХНУРЭ. Научные интересы: обработка радиолокационной информации, распознавание амплитудных и спектральных радиолокационных изображений. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел.: (057)702-14-72, e-mail: vzh@kture.kharkov.ua.

Солонская Светлана Владимировна, инженер НИЦ КВ КП ХНУРЭ. Научные интересы: системы обработки и распознавания изображений, теория информации. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел.: (057)702-14-72, e-mail: svsol@kture.kharkov.ua.

УДК 330.45:330.47

АНАЛИЗ ТЕХНОЛОГИЙ ДОСТУПА К БАЗАМ ДАННЫХ В МНОГОПОЛЬЗОВАТЕЛЬСКИХ ИНФОРМАЦИОННЫХ СИСТЕМАХ

КОБЫЛИН А.М., МАРКОВА Е.Ю.

Рассматриваются универсальный механизм доступа к данным Microsoft ADO и полезные для приложений расширения библиотеки ADO, а именно: ADO Extension for DDL and Security (ADOX) Jet and Replication Objects (JRO) и ADO Multidimensional (ADO MD)), малоописанные в литературе.

1. Постановка проблемы

Универсальный механизм доступа к данным (Universal Data Access) предполагает высокопроизводительный доступ к различным источникам информации, включая реляционные и нереляционные типы данных, в том числе к данным, хранящимся на мэйн-фреймах, данным электронной почты и файловой системы, текстовым, графическим и другим типам данных. На современном этапе деятельности предприятий различных форм собственности возникает потребность в использовании достаточно большого числа источников разнообразных форматов данных. Учитывая перспективы развития предприятий, могут появляться новые форматы данных и способы их хранения. Поэтому разумным требованием к механизму доступа к данным является возможность поддержки не только существующих форматов источников данных, но и форматов данных, которые будут созданы в будущем. Удовлетворить указанным требованиям возможно на основе разработки механизма доступа, основывающегося на единой модели доступа к данным. В настоящее время фирмой Microsoft разработан универсальный механизм доступа к данным, который поддерживает все наиболее популярные настольные и серверные СУБД. Составной частью архитектуры предлагаемого механизма является технология Microsoft ADO (ActiveX Data Objects). ADO широко применяется не только в средствах разработки фирм Microsoft и Borland, но и в таких программных продуктах как Microsoft Office, Microsoft Inter Explorer, ASP - приложениях и др. ADO также является и частью ядра широко применяемых операционных систем семейства Windows 2000.

Несмотря на это технология ADO и расширения библиотеки ADO недостаточно освещены в литературе.

Цель исследования – рассмотреть универсальный механизм доступа к данным Microsoft ADO и полезные для приложений расширения библиотеки ADO, которые позволяют существенно сократить время и ресурсы компьютера для доступа к данным, и решение следующей задачи: ADO Extension for DDL and Security (ADOX) Jet and Replication Objects (JRO) и ADO Multidimensional (ADO MD)).

2. Основной материал

Архитектура универсального механизма доступа к данным может быть представлена в виде, изображенном на рис. 1.

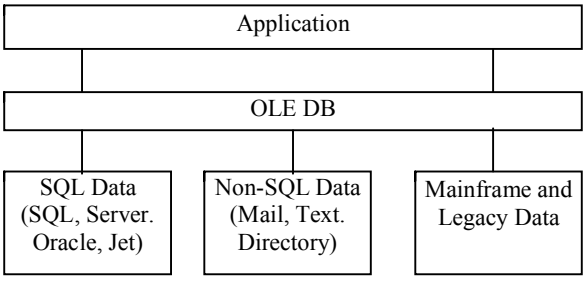

Рис. 1. Архитектура UDA ADO

Основными компонентами такой архитектуры являются: ActiveX Data Objects (ADO), OLE DB и Open Database Connectivity (ODBC).

Элемент архитектуры ADO представляет собой программный интерфейс для доступа к данным из различных приложений. С точки зрения программирования ADO и его расширения являются упрощенным высокоуровневым объектно-ориентированным интерфейсом для OLE DB. ADO позволяет манипулировать данными с помощью любых OLE DB-провайдеров, как входящих в состав Microsoft Data Access Components и некоторых других продуктов Microsoft, так и произведенных сторонними производителями. ADO содержит набор объектов, используемых для соединения с источником данных, для чтения, добавления, удаления и модифи-

кации данных. На рис. 2 изображена его объектная модель.

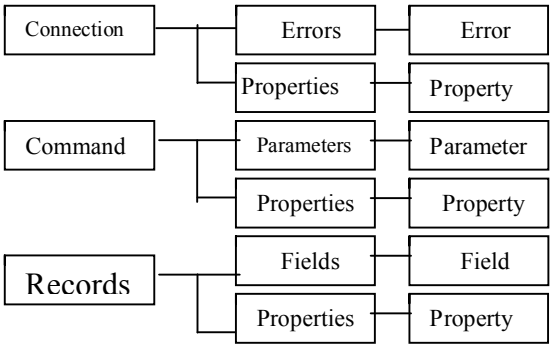

Рис. 2. Объектная модель элемента ADO

Объект ADO Connection применяется для установки связи с источником данных. Он представляет единственную сессию. Этот объект позволяет изменить параметры соединения с базой данных, а также начать или завершить транзакцию. Используя объект Connection, можно выполнять команды (например, SQL-запросы) с помощью метода Execute. Если команда возвращает набор данных, автоматически создается объект Recordset, который возвращается в результате выполнения этого метода. Объект Error используется для получения сведений об ошибках, возникающих в процессе выполнения.

Объект Command представляет собой команду, которую можно выполнить в источнике данных. Команда может содержать SQL-предложение или вызов хранимой процедуры. В последнем случае для определения параметров процедуры может быть использована коллекция Parameters объекта Command. Объект Recordset представляет собой набор записей, полученных из источника данных, и может быть использован для добавления, удаления, изменения и просмотра записей. Данный объект может быть открыт непосредственно или создан с помощью объектов Connection или Command. Объект Field представляет собой колонку в наборе данных, представленных объектом Recordset. Он может быть использован для получения значений конкретного поля, его модификации, извлечения метаданных, таких как имя колонки и тип данных.

Библиотека ADO 2.5, являющаяся составной частью операционной системы Windows 2000, содержит два новых объекта Record и Stream. Объект Record представляет одну запись внутри объекта Recordset и может быть использован для работы с гетерогенными и иерархическими данными. Объект Stream представляет двоичные данные, связанные с объектом Record. Например, если объект Record представляет собой файл, то его объект Stream должен содержать данные внутри этого файла.

Объекты ADO осуществляют взаимодействие с объектами OLE DB по трем возможным направлениям. Объект ADO Connection использует объекты OLE DB DataSource и Session. Транзакции поддерживаются с помощью интерфейсов ITransaction и ITransactionLocal. Метод Execute использует метод Execute интерфейса ICommand или метод OpenRowset интерфейса IOpenRowset. Большинство свойств доступно с помощью IDBProperties. Объект Error поддерживается с помощью интерфейса IErrorRecords. Объект ADO Command использует объект OLE DB Command и интерфейс ICommand. Например, его метод Execute вызывает непосредственно одноименный метод объекта OLE DB Command, его свойство CommandText доступно с помощью методов GetCommandText и SetCommandText интерфейса ICommandText. Коллекция Parameters объекта ADO Command доступна с помощью интерфейса ICommandWithParameters. Объект AI Recordset использует объект OLE DB Rowset. Он применяет интерфейс IRowset, IRowsetLocate и IRowsetInfo для реализации большинства методов свойств и коллекций. Объект Field использует интерфейс IColumnsInfo.

Для решения задач, недоступных с помощью обычных объектов ADO, применяется расширение ADO Extension for DDL and Security (ADOX). Например, используя объекты ADOX, можно извлекать метаданные из базы данных и переносить структуру данных из одной базы данных в другую (в том числе и иного типа). Другая возможность состоит в манипулировании сведениями о безопасности. Например, с помощью ADOX можно получать информацию о пользователях базы данных и группах пользователей, а также создавать новых пользователей и группы. ADOX расширяет объектную модель ADO десятью новыми объектами, которые можно использовать как отдельно, так и совместно с другими объектами ADO, в частности можно применять объект ADO Connection для соединения с источником данных и извлекать метаданные из него.

В большинстве современных СУБД метаданные, представляющие собой описания объектов базы данных (таблиц, полей, индексов, ключей, представлений, хранимых процедур и прочих объектов), определяются с помощью языка SQL. До появления ADOX единственным программным способом извлечения метаданных из источников данных с помощью ADO был метод OpenSchema объекта ADO Connection. Для создания новых объектов в базе данных применялся язык Data Definition Language (DDL), который является подмножеством языка SQL, или объект ADO Command. ADOX предоставляет более универсальный способ манипуляции метаданными, не требующий знания SQL для того, чтобы получить структуру базы данных или даже создать новые объекты. Обратите внимание на то, что ADOX работает далеко не со всеми базами данных. Его функциональность ограничена Microsoft Access и Microsoft SQL Server, а также несколькими типами других СУБД. ADOX обладает собственной объектной моделью, состоящей из 10 объектов, представленных на рис. 3.

Иерархия объектов ADOX начинается с объекта Catalog. Этот объект содержит коллекции таблиц, представлений, процедур, пользователей и групп и

может быть использован для открытия существующей базы данных (с помощью объекта ADO Connection) или создания новой. В версии ADO 2.1 можно создавать только базы данных Jet (т.е. Microsoft Access - файлы с расширением .mdb). Открывать же можно базы данных других типов. Имея объект Catalog, мы можем работать с таблицами, процедурами и представлениями. Например, просматривая коллекцию Tables, можно узнать, какие таблицы имеются в базе данных, а также получить более детальные сведения о таблицах, изучив коллекции Columns, Indexes и Keys объекта Table. Изучая свойства объектов базы данных, можно получить сведения о метаданных и, в частности, сохранить их в отдельном файле или перенести в другое место. Используя коллекции Users и Groups, мы можем манипулировать правилами доступа к данным, создавая отдельных пользователей или группы пользователей базы данных. Отметим, что база данных в этом случае должна быть защищенной (secured): в случае Microsoft Access нам следует включить ссылку на базу данных System.mdw, ответственную за хранение сведений о правилах доступа к данным, в содержимое строки, указывающей на источник данных (connection string). Еще одна интересная особенность ADOX заключается в том, что с помощью этого расширения можно создавать базы данных и объекты внутри них «из ничего». Например, можно создавать базы данных Access, добавлять таблицы, поля, записи, индексы, ключи, а затем добавлять в созданную таким образом базу данных и сами данные (вручную или с помощью кода). Это бывает полезно в тех ситуациях, когда нужно некоторым образом организовать «сырые» данные. В общем случае для создания новой базы данных следует создать объект Catalog и применить метод Add коллекций Tables, Columns, Keys и Indexes для добавления в него объектов базы данных. Обсудив объекты ADOX, мы не коснулись объектов User и Group. В настоящее время в текущей версии ADO эти объекты доступны только для Microsoft Access (Microsoft Jet OLE DB Provider), и пока нет никаких сведений о том, что в последующих версиях ADO будет реализована поддержка этих объектов для других типов баз данных.

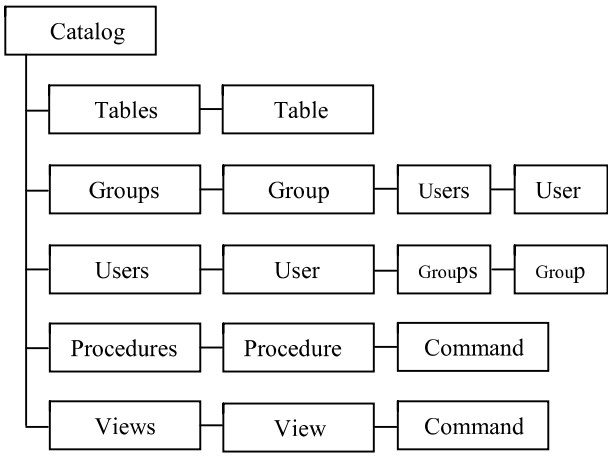

Рис. 3. Объектная модель ADOX

В настоящее время инструменты для анализа данных и поддержки принятия решений считаются одним из важнейших типов приложений. Реализация подобного анализа нередко базируется на построении многомерных хранилищ данных (data warehouses) и на аналитической обработке данных (On-Line Analytical Processing, OLAP) — популярной технологии многомерного бизнес-анализа. В настоящее время поддержка OLAP реализована в различных СУБД и инструментах. Подробности об OLAP можно найти на сайте www.olap.ru, сопровождаемом компанией «Интерфейс», поставщиком многих OLAP-продуктов на российском рынке. Рассмотрим, из чего обычно состоит многомерное хранилище данных. Представим себе торговую компанию, которая хранит все сведения о торговых операциях в какой-либо базе данных, содержащей среди прочих представление Invoices с подробными сведениями о заказах клиентов. Отметим, что хранение агрегатных данных в оперативной памяти — далеко не самый эффективный способ их получения и использования. Более разумным является однократное вычисление агрегатных данных и хранение их в какой-нибудь базе данных. Это и есть ключевая идея построения хранилищ данных (data warehousing) - процесса сбора, отсеивания и предварительной обработки данных в целях предоставления результирующей информации пользователям для статистического анализа и создания отчетов на основании OLAP.

Расширение ADO Multi-Dimensional Extensions (ADO MD) представляет собой набор объектов, позволяющих использовать многомерные данные в ADO-приложениях. Такие данные управляются OLAP-серверами, такими как Microsoft OLAP Server, входящий в комплект поставки Microsoft SQL Server 7.0 (или Analytical Services в Microsoft SQL Server 2000). OLAP-серверы широко применяются в системах принятия решений, где требуется статистический анализ больших объемов данных. Объектная модель ADO MD, представленная на рис. 4, состоит из двух ветвей объектов. Первая из них используется для доступа к метаданным многомерной базы данных, вторая применяется, когда необходимо извлечь данные с помощью запросов к OLAP-кубам. Объекты ADO MD кратко описаны в таблице.

Описание объектов ADO MD

Объект	Назначение
Catalog	Вся схема многомерной базы данных
CubeDef	Многомерный куб в базе данных
Dimension	Размерность внутри куба
Hierarchy	Иерархия внутри размерности
Level	Уровень иерархии
Cellset	Результат запроса к кубу
Cell	Ячейка в объекте CellSet
Axis	Одна из осей в объекте CellSet
Position	Позиция в объектах Cell и Axis
Member	Конкретное значение вдоль оси или уровня иерархии

В объектной модели ADO MD кросс-таблица, являющаяся результатом MDX-запроса, представлена объектом CellSet. Этот объект предоставляет доступ к объектам Cells, представляющим конкретные ячейки в кросс-таблице. Помимо этого объект CellSet содержит коллекцию объектов Axis (обычно в этой коллекции два объекта, соответствующие строкам и столбцам). Как объект Cell, так и объект Axis обладают коллекцией Positions объектов Position, представляющих позицию вдоль оси. Объект Position обладает коллекцией Members, представляющей конкретное значение данных на оси.

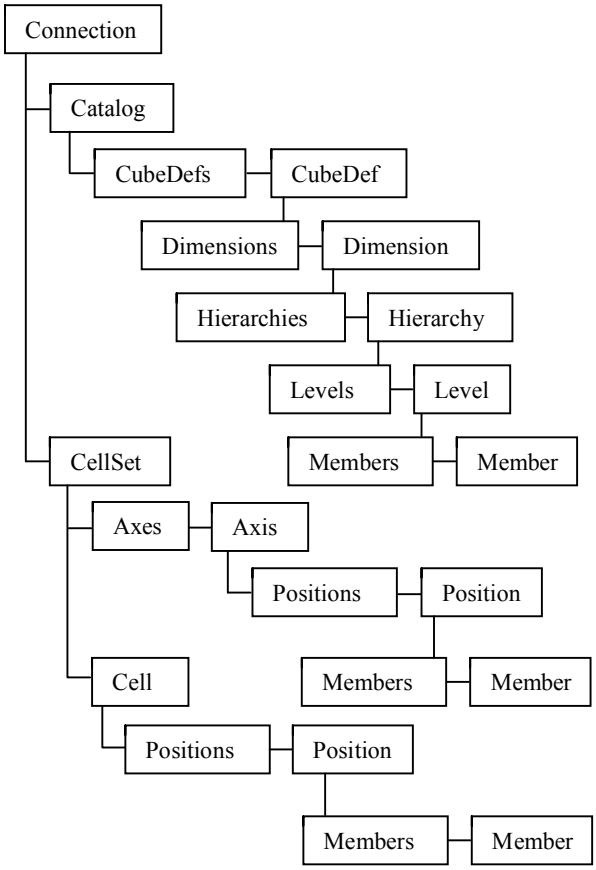

Рис. 4. Объектная модель ADO MD

Следующее расширение ADO, которое мы рассмотрим в данной статье, - Jet and Replication Objects (JRO). В отличие от других расширений ADO (ADOX и ADO MD), которые способны работать с различными источниками данных, объекты JRO были созданы специально для поддержки некоторых операций, характерных для репликации баз данных Jet. Это означает, что такие объекты могут быть использованы только с базами данных Microsoft Access. Объектная модель JRO включает объект JetEn-gine, экспонирующий некоторые особенности Microsoft Jet Engine. В частности, объект JetEngine может быть использован для сжатия базы данных, установки пароля, шифрования базы данных, обновления данных из кэша. JRO представляет собой набор объектов, специально предназначенных для использования совместно с Microsoft Jet OLE DB Provider. Его свойства позволяют создавать, модифицировать и синхронизировать реплики, представляющие копии баз данных, изменения в которых синхронизируются с главной базой данных (master database). Объект Replica используется для создания новых реплик, модификации свойств существующих реплик и синхронизации изменений с другими репликами.

Не углубляясь в подробности функционирования Microsoft Jet Engine, отметим, что при сжатии базы данных происходит следующее:

1. Реорганизуется расположение табличных данных на страницах - после сжатия таблица располагается на соседних страницах. Это повышает производительность, так как таблицы теперь дефрагментированы.

2. Объекты и записи, помеченные как удаленные, реально удаляются, что позволяет высвободить дополнительное пространство.

3. Текущие значения счетчиков, связанных с полями типа AutoNumber, переопределяются, поэтому следующее значение для такого поля будет на единицу превышать максимальное имеющееся значение.

4. Обновляются сведения о таблицах, используемые для оптимизации запросов.

5. Поскольку сведения о базе данных изменились, все запросы будут заново скомпилированы в момент первого обращения.

3. Выводы

Рассмотрены основные части набора технологий, реализующих универсальный механизм доступа к данным. Показано, что универсальный механизм доступа к данным представляет собой стратегию предоставления доступа к любому типу информации предприятия, включая реляционные и нереляционные данные. Рассмотрены основные компоненты архитектуры универсального механизма доступа ADO и расширения ADO. начиная с версии 2.1. Механизм, основанный на применении ADO и OLE DB, имеет все основания претендовать на роль ключевой технологии доступа к данным для Windows-приложений, ибо: 1) он входит в состав не только многих популярных продуктов Microsoft, но и семейства операционных систем Windows 2000, что позволяет безболезненно решать проблемы, связанные с поставкой таких приложений и их сопровождением; 2) OLE DB-провайдеры выпускаются не только автором технологии OLE DB, но и многими сторонними производителями (вспомним, например, что ситуация с BDE-драйверами как раз обратная); 3) доступ к нереляционным данным, предусмотренный данной технологией, является весьма привлекательной ее особенностью, и на данный момент другие универсальные механизмы доступа к данным не могут предложить ничего подобного; 4) данная технология базируется на использовании COM-интерфейсов, поэтому она поддерживается всеми средствами разработки, позволяющими создавать COM-клиенты с их помощью. Кроме того, многие средства разработки, даже

не принадлежащие Microsoft, поддерживают ADO на уровне собственных классов и компонентов (например, Delphi 5 и C++Builder; 5) , несмотря на наличие других встроенных механизмов доступа к данным.

Научная новизна заключается в проведении сравнительного анализа существующих технологий и в выработке рекомендаций по разработке приложений для работы с хранилищами данных в банковских компьютерных сетях.

Практическое значение в том, что объектная модель ADO позволяет придавать последним реализациям этой технологии новые возможности ,что обязательно приведет к повышению популярности этой технологии.

Будущее применение – это использование технологии ADO с нереляционными базами данных.

Литература: 1. *Федоров А., Елманова Н.* ADO в Delphi: Пер. с англ. СПб.: БХВ-Петербург, 2002. 816 с. **2.** Фаронов В.В. Delphi **3.** *Учебный* курс. М.: Нолидж, 1998. 400 с. **3.** *Фаронов В.В.* Delphi 5. Руководство программиста. М.: Нолидж, 2000. 780с. **4.** *Фаронов В.В., Шумаков П.В.* Delphi **5.** *Руководство* разработчика БД. М.: Нолидж, 2000. 640 с. **6.** *Кенту М.* Delphi 5 для профессионалов. СПб.: Питер, 2001. 944 с.

Поступила в редколлегию 28.06.05

Рецензент: д-р физ.-мат. наук, проф. Машкаров Ю.Г.

Кобылин Анатолий Михайлович, канд. техн.наук, проф. каф. информатики ХНУРЭ. Научные интересы: информационные технолоии. Адрес: Украина, 61166, пр. Ленина, 14, (057)-7021-419.

Маркова Елизавета Юрьевна, студентка 5 курса ПММ ХНУРЭ. Научные интересы: математическое моделирование и информационные технологии. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. (057)-7021-419.

УДК 517.95

ИНТЕРЛОКАЦИОННЫЕ ФОРМУЛЫ ЛАГРАНЖА И ЭРМИТА ДЛЯ ЛОКУСОВ-ТОЧЕК

УВАРОВ Р.А., ШЕЙКО Т.И.

Исследуются интерлокационные формулы Лагранжа и Эрмита при восстановлении функции двух переменных, когда узлами являются локусы-точки. Модифицируется интерлокационная формула Эрмита для возможности использования сглаживающих функционалов при известных и неизвестных значениях частных производных функции в узлах. Для сглаживания начальных приближений применяются функционалы, моделирующие натяжение «мыльной плёнки» и изгиб пластины.

Постановка цели и задач исследования

Целью данной работы является модификация интерлокационных формул Лагранжа и Эрмита для локусов-точек.

Для выполнения поставленной цели нужно выполнить следующие *задачи*:

1) исследовать результаты применения интерлокационной формулы Лагранжа на системе локусов-точек, моделируя натяжение «мыльной плёнки», установить возникающие трудности и предложить пути их преодоления;

2) исследовать результаты применения интерлокационной формулы Эрмита на системе локусов-точек, моделируя изгиб пластины в случае, если известна информация о частных производных, путем некоторой её модификации;

3) модифицировать интерлокационную формулу Эрмита на системе локусов-точек в случае, если информация о частных производных неизвестна, оценить результаты её применения, моделируя натяжение мыльной пленки и изгиб пластины, применить её и оценить результаты при восстановлении кусочно-гладких функций.

Основное содержание исследования

Конструктивный аппарат теории R-функций [1], который позволяет учитывать геометрическую информацию в аналитическом виде, оказывается полезным математическим инструментарием для решения задач с неполной информацией об объекте, а также связанных с обработкой данных, имеющих разброс.

При восстановлении значения функции в области по заданным значениям функции на локусах используется интерлокационная формула Лагранжа, а когда на локусах заданы значения функций и производные по заданным направлениям, используется интерлокационная формула Эрмита.

Интерлокационная формула Лагранжа:

$$u = u_0 + u_1 = (\underset{i=0}{\overset{N}{\sim}} \omega_i) \cdot (\underset{i=0}{\overset{N}{\sim}} \frac{\omega_i}{\varphi_i})^{-1} + (\underset{i=0}{\overset{N}{\sim}} \omega_i)\Phi, \qquad (1)$$

где $\omega_i = \frac{|\hat{\omega}_i| - \hat{\omega}_i}{2}$ – функция, нулевой уровень которой определяет локус; $\hat{\omega}_i(x, y)$ – функция, нулевой уровень которой определяет границу локуса, а $\underset{i=1}{\overset{N}{\sim}} x_i = x_1 \sim x_2 \sim ... \sim x_N$.

Здесь используется ассоциативная операция R-равнозначности: $x \sim y = \frac{xy}{x + y}$, если $x, y > 0$ или $x \overset{m}{\sim} y = \frac{xy}{\sqrt[2m]{x^{2m} + y^{2m}}}$ – для произвольных x, y, где m – положительное целое число.

Интерлокационная формула Эрмита:

$$u = u_0 + u_1 =$$

$$= (\underset{i=0}{\overset{N}{\sim}} \omega_i^{k_i+1}) \cdot (\underset{i=0}{\overset{N}{\sim}} \frac{\omega_i^{k_i+1}}{\varphi_i - q_i \omega_i D_1^{(l_i)}(\varphi_i) + q_i \omega_i \psi_i})^{-1} +$$