ИССЛЕДОВАНИЕ ВОЗМОЖНОСТЕЙ ОБЪЕКТНОГО ПРЕДСТАВЛЕНИЯ ДАННЫХ В ПРИКЛАДНЫХ СИСТЕМАХ

В. И. Гришков

Санкт-Петербургский институт информатики и автоматизации РАН
199178, Санкт-Петербург, 14-я линия В.О., д.39
Grishkov@sparm.com

В. И. Гришков. **Исследование возможностей объектного представления данных в прикладных системах** // Труды СПИИРАН. Вып. 1, т. 3. — СПб: СПИИРАН, 2003.
Аннотация. *Цель данной статьи заключается в том, что бы детально исследовать возможности объектного представления данных. В статье проводится анализ объектного подхода. Обсуждается неоднозначность объектного подхода и его возможности. Объектный подход предоставляет мощные средства в работе с данными.* — Библ. 5 назв.

*V. I. Grishkov***. Investigation of object representation of data in applied systems** // SPIIRAS Proceedings. Issue 1, v.3. — SPb: SPIIRAS, 2003.
Abstract.*The purpose of this paper is to investigate in details means of objective data presentation. In the paper the analysis of the objective approach will be carried out. It is discussed ambiguity of the objective approach and its opportunity. The approach provide a mesure that is appropriate strength for data representation*. — Bibl. 5 Items.

1. Введение

Выбор наиболее оптимального способа представления данных всегда являлся одним из самых важных этапов создания прикладных систем. Задача любой прикладной системы заключается, прежде всего, в том, чтобы обрабатывать и хранить информацию. Эта информация может поступать из принципиально различных источников. В качестве источников информации могут быть человек, датчик, другая прикладная система и все, что способно передавать информацию в том или ином виде.

В данной статье исследуется объектная парадигма представления данных. Сложность и неоднозначность такого способа представления данных трудно переоценить. На данный момент этот способ является не только практически, но и теоретически слабо проработанным. Но это не в коем случае не умоляет принципиальную важность самого подхода, а лишь указывает на невообразимое множество проблем, с которыми вынуждены сталкиваться разработчики и теоретики. И дело тут не столько в самом подходе, сколько в самом феномене информации. Структура информации такова, что вынуждает искать максимально оптимальные способы ее хранения и обработки. Объектное представление данных является одним из таких способов.

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

2. Принципы реализации реляционного представления данных

Начнем с понятия реляционной модели. Реляционная модель, предложенная Е. Коддом в 1970 г, имеет строгую математическую модель. Эта модель всецело описывается математическим аппаратом теории множеств.

Множество является наиболее общим понятием в математике. Множество не обладает внутренней структурой и его можно определить как совокупность элементов, обладающих некоторым общим свойством. Между отдельными изолированными элементами множества отсутствуют какие-либо взаимосвязи. Для того чтобы некоторую совокупность элементов можно было назвать множеством, необходимо, чтобы существовало правило, позволяющее определить, принадлежит ли указанный элемент данной совокупности элементов, а также должно существовать правило, позволяющее однозначно идентифицировать элементы.

Как известно, над множествами может быть введен ряд операций: объединение, пересечение, разность и декартово произведение. Для нас наиболее важно понятие отношения, т.к. оно лежит в основе реляционного представления данных. Отношением (relation) называется некое подмножество R декартового произведения множеств. Из этого определения вытекает, что отношением является не всякое множество, а лишь то множество, которое содержит в себе только однотипные элементы (кортежи). При этом отношение не включает в себя все возможные кортежи из декартового произведения. Для каждого отношения должен иметься некоторый критерий, который позволил бы определить факт вхождения кортежа в отношение. Можно сказать, что этот критерий определяет семантику отношения.

Между реляционной моделью данных и предикатной всегда существует связь. Любому n-мерному отношению можно сопоставить логическую функцию от n-аргументов, которая служила бы критерием вхождения в отношение кортежа, составленного из этих аргументов.

3. Недостатки реляционного подхода

Каждая прикладная система, построенная на базе реляционной системы управления базами данных (СУБД), обязана иметь некоторое множество программ специально написанных для конкретной прикладной области. Программы, написанные для другой прикладной системы, нельзя использовать повторно. Такая ситуация с программами сильно усложняет процесс создания прикладных систем, т.к. каждую прикладную систему приходится строить с нуля. Это не только затягивает процесс создания прикладной системы, но и усложняет процесс тестирования и внедрения. Попробуем ответить на вопрос: почему так происходит? Рассматривая процесс создания прикладных систем на базе реляционных СУБД, можно четко выделить два этапа:

- создается схема базы данных (БД) адекватная прикладной области;
- пишется множество программ призванных визуализировать данные или представлять их в каком-нибудь другом удобном для пользователя виде.

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

По мере того как реализуется второй этап создания прикладной системы, часто выясняется, что схема БД не оптимальна с точки зрения производительности и ее приходиться менять.

Существует еще множество причин, по которым меняются схемы БД, а производительность – это только одна из них, быть может, не самая главная. Любая прикладная область имеет собственную динамику, постоянно развивается, ее требования к задачам автоматизации все возрастают.

При изменении схемы БД меняется структура реляционных таблиц (отношений), их количество и взаимосвязь. Уже написанные программы приходится переделывать под новую схему БД. Этот трудоемкий процесс отнимает время у разработчиков и переводит проектирование в другую плоскость. Проблематика прикладной области отходит на второй план. Разработчик вынужден учитывать уже две модели: модель прикладной области и модель прикладной системы, которая начинает все более отдаляться от прикладной области. Это и есть основная причина краха реляционного подхода к представлению данных в прикладных системах.

4. Принципы реализации сетевого представления данных

Рассмотрим сетевую модель организации данных т.к. она является наиболее общей. Известно, что сетевой подход к организации данных является расширением иерархического подхода. В иерархических СУБД любая запись-потомок должна иметь в точности одного предка; в сетевых СУБД запись-потомок может иметь любое число предков. Сетевая БД состоит из набора записей и набора связей между этими записями. Набор связей между записями представляет собой типы связей. Тип связи определяется для предка и потомка. При этом предок и потомок могут быть связаны более чем одним типом связи.

В общем, сетевую модель данных не составляет труда перевести в предикатную, а предикатную в реляционную. На формальном уровне все эти модели обладают примерно равной функциональностью. Но есть и несущественные отличия. Так, например, в общем случае, с помощью предикатной функции можно реализовать отношение с вычисляемыми (виртуальными) кортежами, которые реально в БД не хранятся. Отношения, предложенные Е. Коддом, такой функциональностью не обладают. Однако этот недостаток легко покрывается с помощью встроенных процедур.

5. Реляционно-объектные СУБД

Начиная с 90-х годов рынок объектных СУБД начал существенно набирать обороты. Из-за этого доходы компаний от продаж реляционных СУБД начали падать. Поэтому ими была предпринята попытка включить некоторые особенности объектной модели в реляционные СУБД. Так появились гибридные реляционно-объектные СУБД.

Некоторые исследователи доказывали, что реляционно-объектное представление данных является следующим шагом в развитии объектной модели. Но детальное ознакомление с такими продуктами позволило обнаружить неполноцен-

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

ность такого подхода. На практике оказалось, что на базе реляционно-объектных СУБД почти невозможно построить эффективные прикладные системы. При сравнении реляционного и объектного подходов становится ясно, в чем причина краха такого рода систем.

6. Основные понятия объектного подхода

На формально-логическом уровне представления данных в прикладных системах удобно оперировать в терминах объектно-ориентированных языков программирования (ООЯП).

Объект — это некоторая сущность, поведение которой и структура описывается с помощью класса, к которому он принадлежит. Иногда объект обозначают термином «экземпляр класса»

Класс — это структура, которая может порождать однотипные объекты с заданными свойствами и поведением. В применении к объектно-ориентированным языкам под классом обычно понимают формальное описание совокупности объектов. Классы описываются либо в текстах прикладных программ, либо в репозитории — словаре классов. Классы имеют собственную подструктуру. Подструктура делится на элементы: события, методы, свойства. По желанию, часть элементов можно сделать внутренними, т.е. запретить к ним доступ со стороны других объектов.

Объекты могут обладать состоянием, поведением, а также получать или генерировать события. Под состоянием объекта понимается совокупность его статических и динамических свойств. Под поведением объекта понимаются его возможность изменять свое состояние из-за внешних воздействий и способность самому воздействовать на другие объекты прикладной системы. Поведение объекта однозначно описывается в методах класса и может зависеть от его динамических свойств. В современных ООЯП при описании свойств объекта допускается возможность их связи с методами. Так, например, для свойств назначаются события записи и считывания, а к этим событиям прикрепляются методы, которые предназначены для отработки этих событий.

7. Принципы объектного представления данных

Совокупность объектно-ориентированных анализа, проектирования и программирования формирует объектно-ориентированный подход к разработке прикладных систем.

Объектное представление данных всегда подразумевает реализацию инкапсуляции и иерархии.

Инкапсуляция позволяет заключать объекты в четко очерченные интерфейсы взаимодействия объекта с другими объектами предметной области и с внешним миром. Благодаря инкапсуляции становится возможным скрыть все детали внутренней реализации объекта и определить допустимый набор свойств и методов, посредством которых другие объекты могут взаимодействовать с этим объектом.

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

Иерархия позволяет распределять классы объектов по уровням. Примером иерархии может служить наследование (в том числе и множественное) и агрегация.

Дополнительно существует еще две особенности объектного подхода — типизация и сохраняемость.

Типизация защищает разработчика от некорректного использования в прикладных программах объектов одного класса вместо другого. Сохраняемость (или хранимость) позволяет объекту существовать в системе после завершения выполнения породившего его процесса. Принцип сохраняемости является чрезвычайно важным для концепции объектных СУБД.

8. Отношения между классами

Для описания взаимодействия между классами вводится понятие отношения или связи. Можно выделить следующие основные отношения: ассоциация, наследование, агрегация, использование.

Ассоциация — наиболее общий тип связи. Ассоциация — это связь “по смыслу”, т.е. она позволяет определить по имеющемуся объекту (объектам) другой объект (объекты). По мощности ассоциативные отношения бывают: «один-к-одному», «один-ко-многим», «многие-ко-многим».

Наследование выражает «общее-частное». Для двух классов, один из которых унаследован от другого, справедливо, что класс-потомок наследует некоторые свойства и поведение от своего класса-родителя. Наследование может быть одиночным (простым) и множественным. Как следует из названий, в случае одиночного наследования, класс наследует свойства и поведение от одного класса-родителя, в случае множественного — от нескольких.

Агрегация представляет другое очень важное отношение между классами — «часть» и «целое». Агрегация имеет место тогда, когда в методах одного класса используется функциональность другого класса. Возможны и другие способы реализации агрегации.

9. Сравнение подходов к представлению данных в реляционных и объектных СУБД

Рассмотрим современные реляционные СУБД. Реляционные СУБД хранят данные в таблицах (отношениях). В переделах одной таблицы каждый ряд (кортеж) это экземпляр той сущности, которую таблица призвана представлять. Колонки таблицы – это свойства сущности. Таким образом, самой таблице можно сопоставить класс, а записям в таблице можно сопоставить экземпляры класса, т.е. объекты. Кортежи в таблице однозначно идентифицируются ключами. Значение ключа в пределах таблицы всегда уникально, иначе было бы нарушено правило уникальности кортежей. Следовательно, всегда можно обратиться к объекту класса по его ключу [2]. Все просто и понятно. Такой подход как раз и предлагается реляционно-объектными СУБД. Это и послужило причиной их неудачного внедрения. Нельзя спорить с тем, что реляционная таблица является классом, это то как раз и

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

правильно. Совершенно без внимания было оставлен тот факт, что все отношения схемы БД — это классы, которые наследуются от одного класса-родителя. Этот класс-родитель и реализует всю функциональность аппарата теории множеств. Следовательно, все отношения по существу однотипны и не могут представлять объекты реального мира.

Посмотрим на все сказанное в другом ракурсе. На первый взгляд объектные БД не содержат в себе никаких преимуществ по сравнению с реляционными БД. При использовании встроенных процедур в реляционных БД можно добиться не только процедурных связей между таблицами, но и между кортежами. Тогда таблица, взятая вместе с процедурами работы с ней, казалось бы ничем не отличается от объекта в формальном его понимании. Но тем не менее в реляционных БД отсутствует явный механизм инкапсуляции, что не позволяет однозначно определить границы объектов моделирующих прикладную область.

На деле в реляционных БД существуют только два принципиально разных класса объектов:

- реляционная таблица с конечным набором операций, которые допустимы для отношений (имеются в виду операции над множествами);
- встроенные процедуры, работающие с отношениями.

Но из этих двух классов объектов нельзя создавать совершенно новые типы т.к. в реляционных БД отсутствуют полноценные механизмы характерные для объектного подхода.

10. Свойство дискретности

Ввиду того, что прикладные информационные системы разрабатываются обычно на дискретной элементной базе или попросту говоря на компьютерах, то конечно же этот факт нельзя упускать из вида. Приходиться учитывать дискретность представления и те последствия, к которым она может привести. Вообще, проблеме дискретности нужно уделять особое внимание, т.к. она лежит в основе всего объектного представления данных и является его неотъемлемым свойством. В терминах объектного подхода дискретность выражается в виде экземпляров объектов некоторого класса или в виде дискретного множества классов. Дискретность также выражена в самой структуре класса, который выражается в виде дискретных элементов класса (свойства, методы и т.д.).

Свойство дискретности объектного представления данных в прикладных системах порождает трудности связанные с идентификацией и неразличимостью объектов реального мира [1].

11. Структура обобщенной объектной прикладной системы

При исследовании возможностей объектного представления и объектной обработки данных, в прикладных системах была выявлена некоторая протоструктура (общая основа), вследствие чего стало возможным создание обобщенной прикладной системы, на базе которой было бы удобно создавать достаточно широкий класс прикладных систем.

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

Любая прикладная система не является замкнутой. Задача прикладной системы заключается в том, чтобы адекватно реагировать на информационное сообщение. Об информационном сообщении можно говорить тогда, когда происходит информационное взаимодействие двух и более систем (например: человек — компьютер). Информационное взаимодействие осуществляется по схеме: источник сообщения — канал передачи — приемник сообщения [3]. Вполне возможна ситуация когда источник сообщения и приемник сообщения меняются местами.

Чтобы осуществлялось информационное взаимодействие, протоструктура должна иметь четыре функциональных части:

- регистрация информации;
- хранение информации;
- обработка информации;
- вывод информации;

Легко можно заметить, что функциональные части протоструктуры являются следствием особенностей информации как феномена.

Попытаемся протоструктуру сделать объектной. Эти четыре функциональных части не являются абсолютно независимыми друг от друга. Например, если мы имеем дело с интерактивным выводом визуальной информации, то объекты, классы которых использовались для вывода информации (в простейшем случае окно и кнопка на ней), должны уметь регистрировать события, соответствующие их функциональному назначению (в нашем случае — щелчок мыши на кнопке).

Сразу приходит мысль, что в зависимости от назначения, классы надо реализовывать в соответствующей программной среде т.к. нужную функциональность можно просто наследовать от уже существующих классов. Например, классы, которые отвечают за хранение информации, обычно пишутся в объектных СУБД (там есть соответствующий набор классов реализующих это хранение).

Допустим, в задачу прикладной системы входит регистрация и запись показаний некоего датчика, который через внешний порт подключен к компьютеру. Класс, регистрирующий события, удобно написать скажем в C++, а фактическую запись показаний поручить классу который реализован в объектной СУБД. Связь регистрации события с его фактической записью заключается в том, что класс, регистрирующий события, функционально связан с классом, записывающим данные.

Как должна происходить обработка информации в объектной протоструктуре? Из только что приведенного примера видно, что обработка информации — это запуск необходимых цепочек отношений между объектами (или в пределах одного объекта), которые предписаны классами, которые регистрируют события.

Таким образом, протоструктуру вполне удобно реализовывать в виде классов.

Обобщенная прикладная система — это инструментарий, позволяющий максимально быстро с минимальными затратами труда реализовывать все четыре функциональные части любой прикладной системы.

Принимая во внимание выше изложенное, естественно реализовать обобщенную прикладную систему на базе объектной протоструктуры с дополнительными библиотеками классов.

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

12. Неоднозначность и сложность объектного подхода

Попытаемся посмотреть на объектный подход к представлению данных не с точки зрения особенностей его программной реализации, а с точки зрения парадигмы представления объектных данных. Интуитивно мы понимаем, что объект это нечто несоизмеримо более сложное, нежели простая реляционная таблица. Рассмотрим, например саму объектную БД. Если взять язык более высокого уровня, чем формальный (контекстно-независимый) то, надо ли понимать под объектной БД некоторое пространство, в котором потенциально могут существовать объекты? Или быть может объектная БД сама является одним неделимым объектом, задача которого — реагировать на внешние воздействия?

В данных вопросах нет никакого противоречия. Просто эти вопросы предполагают различные уровни абстрагирования. При изменении уровня абстрагирования меняются сами объекты и их количество. Так, например, множество объектов сливаются в один объект, если мы изучаем свойства их совокупности.

Законы, благодаря которым выделяются (распознаются) объекты внешнего мира, не подчинены одному лишь методу абстрагирования, как приему мышления. Эти законы подчиняются правилам существования контекстных потоков. Хотелось бы особо подчеркнуть, что объект — это устойчивый контекстный поток. Контекстные потоки обязаны быть устойчивыми, в противном случае объект находился бы на стадии формирования и может вообще не возникнуть в нашем сознании. Устойчивость контекстных потоков — одна из важных характеристик объектов реального мира, отраженных в нашем сознании.

В общем случае объекты реального мира нельзя алгоритмизовать [4]. Они не подчиняются законам строгих математизированных языков программирования. Поэтому на основании априорных понятий мы пытаемся выделять «чистые» объекты из объектов реального мира [5]. Моделируя «чистые» объекты во многих случаях не обязательно переходить в понятийную систему отличную от объектной. Тут срабатывает принцип экономии мышления. В этом еще одна из привлекательных сторон объектного подхода.

Литература

[1] *Лачинов В. М., Поляков А. О.* Информодинамика. — СПб СПбГТУ, 1999. — с. 432.

[2] *Поляков А. О, Гришков В. И.* Постобъектный подход к представлению знания // SCM'99. Сборник докладов. — СПб, 1999. — с. 244-247.

[3] *Полонников Р. И.* Информационные взаимодействия биообъектов // Телемедицина. Новые информационные технологии на пороге 21 века. — СПб, 1998. — с. 99-120.

[4] *Рахилина Е. В.* Когнитивная семантика; история, персоналии, идеи, результаты. — М, 1998.

[5] *Башляр Г.* Новый рационализм. — Биробиджан: Тривиум, 2000. — с. 376.

Труды СПИИРАН. 2003. Вып. 1. Т. 3. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2003. Issue 1. V. 3. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru