УДК 004.0272.3:004.652.5
DOI: 10.25559/SITITO.16.202003.564-574

Оригинальная статья

Объектно-ориентированный подход к разработке моделей данных

Е. П. Емельченков, В. И. Мунерман*, Д. В. Мунерман, Т. А. Самойлова
ФГБОУ ВО «Смоленский государственный университет», г. Смоленск, Российская Федерация
214000, Российская Федерация, г. Смоленск, ул. Пржевальского, д. 4
* vimoon@gmail.com

Аннотация

Современное понимание объектно-ориентированного подхода концентрируется на одной из основных проблем программирования — его автоматизации. В этом направлении достигнуто много важных и практически полезных результатов. Разработаны методы создания объектных библиотек и технологии программирования, существенно облегчающие разработку программного обеспечения. Особенно это проявляется в разработке WEB-приложений. Технология MVC позволила повысить эффективность разработки программ, связывающих удаленные базы данных с клиентскими приложениями. Однако этой и аналогичным ей технологиям присущ один существенный недостаток, проявляющийся в том, что время получения данных их базы часто не удовлетворяет требованиям как разработчика, так и пользователя. На протяжении длительного времени в интернете постоянно ведутся дискуссии на эту тему. Предлагаются различные методы ускорения обращений к БД, которые, как правило, усложняют программирование и представляют собой искусственные приемы, выходящие за рамки основной технологии, применимые при решении частных задач и требующие существенной переработки в каждом конкретном случае. В статье предполагается, что причина возникновения этих проблем кроется в том, что в программистском сообществе укрепилось мнение, суть которого состоит в том, что реляционная модель данных, служащая основой большинства современных СУБД, не имеет ничего общего с объектно-ориентированной парадигмой. Поэтому разработчики технологий программирования используют промежуточную технологию, называемую объектно-реляционным отображением (ORM), которая, по их мнению, должна связать необъектную БД и объектную программу, обеспечивающую связь клиентского приложения с базой данных.

На самом деле, любой способ организации систем баз данных полностью соответствует объектно-ориентированной парадигме. Исходя из этих посылок в статье рассматривается объектный метод разработки моделей данных. Для этой цели формулируются свойства моделей данных, необходимые для того, чтобы они были объектными и обеспечивали высокую производительность программ, реализующих обработку данных на их основе. Вводится понятие абстрактной алгебраической машины, которая представляет собой двухосновную алгебраическую систему. Приведены примеры, иллюстрирующие применение абстрактной алгебраической машины для решения различных задач. Рассмотрена возможность и предложен метод для построения алгебраических систем для сложных структур данных, а именно, кортежей, которые могут рассматриваться как элементы таких агрегатов данных как многомерные матрицы, отношения, файлы. В заключении сделан вывод о том, что предложенный подход позволяет доказать соответствие различных моделей данных, что, в свою очередь, дает возможность эффективной реализации параллельной обработки различных БД на программно-аппаратных комплексах, архитектура которых основана на многомерно-матричной модели вычислений.

Ключевые слова: модели данных, параллельное программирование, алгебра сложных структур данных.

Авторы заявляют об отсутствии конфликта интересов.

Для цитирования: Емельченков, Е. П. Объектно-ориентированный подход к разработке моделей данных / Е. П. Емельченков, В. И. Мунерман, Д. В. Мунерман, Т. А. Самойлова. — DOI 10.25559/SITITO.16.202003.564-574 // Современные информационные технологии и ИТ-образование. — 2020. — Т. 16, № 3. — С. 564-574.

Original article

The Object Oriented Approach to Designing Data Models

Ye. P. Emelchenkov, V. I. Munerman*, D. V. Munerman, T. A. Samoilova
Smolensk State University, Smolensk, Russian Federation
4 Przhevalsky St., Smolensk 214000, Russian Federation
* vimoon@gmail.com

Abstract

The modern understanding of the object-oriented approach focuses on one of the main problems of programming — its automation. Many important and practically useful results have been achieved in this direction. Methods for creating object libraries and programming technologies have been developed, which greatly facilitate software development. This is especially evident in the development of WEB-applications. MVC technology has improved the efficiency of developing programs that link remote databases to client applications. However, this and similar technologies have one significant drawback, which is manifested in the fact that the time of obtaining data from their database often does not meet the requirements of both the developer and the user. For a long time, there have been ongoing discussions on this topic on the Internet. Various methods are proposed to speed up database calls, which, as a rule, complicate programming and are artificial methods that go beyond the basic technology, which are applicable to solving particular problems and require significant processing in each specific case. The article assumes that the reason for these problems lies in the fact that in the programming community there is a solidified opinion, the essence of which is that the relational data model, which serves as the basis for most modern DBMS, has nothing to do with the object-oriented paradigm. Therefore, developers of programming technologies use an intermediate technology called object-relational mapping (ORM), which, in their opinion, should connect the non-object database and the object program that provides the client application with the database.

In fact, any way you organize your database systems is in the object-oriented paradigm.

Based on these premises, the article discusses the object method of developing data models. For this purpose, the properties of data models are formulated, which are necessary for them to be object and to ensure high performance of programs that implement data processing on their basis. The concept of an abstract algebraic machine is introduced, which is a two-base algebraic system. Examples are given that illustrate the use of an abstract algebraic machine for solving various problems. The possibility is considered and a method is proposed for constructing algebraic systems for complex data structures, namely, tuples, which can be considered as elements of such data aggregates as multidimensional matrices, relations, files. In conclusion, it is concluded that the proposed approach allows one to prove the correspondence of various data models, which, in turn, makes it possible to effectively implement parallel processing of various databases on software and hardware complexes, the architecture of which is based on a multidimensional matrix computation model.

Keywords: data models, parallel programming, algebra of complex data structures.

The authors declare no conflict of interest.

For citation: Emelchenkov Ye.P., Munerman V.I., Munerman D.V., Samoilova T.A. The Object Oriented Approach to Designing Data Models. *Sovremennye informacionnye tehnologii i IT-obrazovanie* = Modern Information Technologies and IT-Education. 2020; 16(3):564-574. DOI: https://doi.org/10.25559/SITITO.16.202003.564-574

Введение

Современное понимание объектно-ориентированного подхода концентрируется на одной из основных проблем программирования — его автоматизации. В этом направлении достигнуто много важных и практически полезных результатов. Разработаны методы создания объектных библиотек и технологии программирования, существенно облегчающие разработку программного обеспечения. Особенно это проявляется в разработке WEB-приложений. Технология MVC позволила повысить эффективность разработки программ, связывающих удаленные базы данных с клиентскими приложениями. Однако этой и аналогичным ей технологиям присущ один существенный недостаток, проявляющийся в том, что время получения данных их базы часто не удовлетворяет требованиям как разработчика, так и пользователя. На протяжении длительного времени в интернете постоянно ведутся дискуссии на эту тему. Предлагаются различные методы ускорения обращений к БД, которые, как правило, усложняют программирование и представляют собой искусственные приемы, выходящие за рамки основной технологии, применимые при решении частных задач и требующие существенной переработки в каждом конкретном случае.

Авторы считают, что причина возникновения этих проблем кроется в том, что в программистском сообществе укрепилось мнение, суть которого состоит в том, что реляционная модель данных, служащая основой большинства современных СУБД, не имеет ничего общего с объектно-ориентированной парадигмой. Поэтому разработчики технологий программирования используют промежуточную технологию, называемую объектно-реляционным отображением (ORM), которая, по их мнению, должна связать необъектную БД и объектную программу, обеспечивающую связь клиентского приложения с базой данных.

На самом деле, любой способ организации систем баз данных полностью соответствует объектно-ориентированной парадигме. На примере реляционного подхода это выглядит следующим образом:

1. Реляционная модель данных представляет собой базовый объект **(**суперкласс**)**.
2. СУБД — конкретная реализация реляционной модели.
3. Схема БД — это абстрактный объект (абстрактный класс), наследник базового объекта в рамках СУБД, содержащий таблицы (свойства) и запросы (методы). Запросы могут быть реализованы хранимы процедурами и функциями.
4. БД — экземпляр объекта (объект), наследник схемы БД, в котором таблицы заполнены реальными значениями, различными у разных пользователей, и запросы в каждом сеансе принимают конкретные значения параметров.

Исходя из этих посылок в статье рассматривается объектный метод разработки моделей данных. Для этой цели формулируются свойства моделей данных, необходимые для того, чтобы они были объектными и обеспечивали высокую производительность программ, реализующих обработку данных на их основе. Вводится понятие абстрактной алгебраической машины, которая представляет собой двухосновную алгебраическую систему. Приведены примеры, иллюстрирующие применение абстрактной алгебраической машины для решения различных задач. Рассмотрена возможность и предложен метод для построения алгебраических систем для сложных структур данных, а именно, кортежей, которые могут рассматриваться как элементы таких агрегатов данных как многомерные матрицы, отношения, файлы.

В заключении сделан вывод о том, что предложенный подход позволяет доказать соответствие различных моделей данных, что, в свою очередь, дает возможность эффективной реализации параллельной обработки различных БД на программно-аппаратных комплексах, архитектура которых основана на многомерно-матричной модели вычислений.

Свойства моделей данных

При разработке моделей данных необходимо чтобы они удовлетворяли следующим требованиям:

1. ***Соответствие моделей данных и вычислений.*** Нефор-мально, требование соответствия двух моделей означает наличие у них свойств, позволяющих использовать одну модель данных вместо другой, а также в возможности распространения полезных свойств одной модели на другую. В статье рассматриваются два типа соответствия моделей. Первый тип состоит в том, что:
   - каждому набору данных (прообразу), представленному в одной модели (например, отношению в реляционной, файлу в теоретико-множественной), ставится в соответствие один и только один набор данных (образ), представленный в другой модели (например, многомерная матрица);
   - каждой операции в одной модели ставится в соответствие одна и только одна операция или композиция операций в другой модели, и результату операции над прообразами соответствует результат операции над образами.
   - В математике, в частности, в теории абстрактных алгебраических систем, такое соответствие называется изоморфизмом. Второй тип соответствия состоит в том, что:
   - каждому набору данных (прообразу), представленному в одной модели, ставится в соответствие единственный набор данных (образ), представленный в другой модели (например, логическая многомерная матрица), то есть, нескольким прообразам может соответствовать один и тот же образ;
   - каждой операции в одной модели ставится в соответствие одна и только одна операция или композиция операций в другой модели, и результату операции над прообразами соответствует результат операции над образами.

   Такое соответствие называется гомоморфизмом.

2. ***Процедурность.*** Модель должна обеспечивать алгебраическую процедурную формулировку запроса, которая задает правила его реализации. То есть запрос, представленный на языке модели, может быть вычислен на основе выполнения элементарных алгебраических операций, определенных в модели, с учетом приоритетности, воз-

можного наличия скобок и некоторых дополнительных правил, определяющих порядок их выполнения. К требованию процедурности рассматриваемых в работе моделей добавляются два дополнительных требования.

Первое требование состоит в том, что и сами элементарные алгебраические операции, которые реализуют выполнение запроса, должны иметь формализованные описания, позволяющие проектировать процедуры, реализующие их алгоритмы, таким образом, чтобы они наилучшим образом выполнялись в используемой модели вычислений. В распространенных в настоящее время моделях данных операции не имеют формализованных описаний. Поэтому качество их реализации определяется мастерством программистов, которые разрабатывают СУБД.

Второе требование заключается в предоставлении прикладному программисту возможности применения способов организации и распределения данных, не реализованных в конкретной СУБД, и разработки на основе имеющегося языка манипулирования данными процедур запросов, эффективно использующих выбранные организацию данных и модель вычислений. В современных СУБД эта возможность реализована средствами хранимых процедур и функций (UDF), которые разрабатывает прикладной программист. После трансляции и оптимизации они сохраняются в БД и могут быть вызваны в ходе решения конкретной задачи. Кроме того, современные СУБД позволяют встраивать в БД новые типы данных в виде объектов, содержащих переменные (на принятом сегодня языке — свойства), а также процедуры и функции, реализующие операции над этими переменными (методы). Пример такой реализации — среда CLR в технологии .NET компании Microsoft.

Применение моделей данных, соответствующих этим требованиям позволит:

- программистам, разрабатывающим СУБД, — расширить круг архитектур вычислительных комплексов, на которых станет возможным применение этих СУБД;
- прикладным программистам — возможность привязывать различные СУБД к выбранной для решения прикладной задачи архитектуре вычислительного комплекса.

3. ***Параллелизм алгебраических операций.*** Формализация операций должна обеспечивать возможность распараллеливания операций совместной обработки двух или более агрегатов данных. Так формальное определение операций в теоретико-множественной и реляционной моделях [1] позволяет использовать методы распараллеливания, основанные на оптимальном распределении агрегатов данных (файлов, таблиц) между запоминающими устройствами процессоров, реализующих одновременную обработку их фрагментов, а в многомерно матричной модели — системам на основе векторных (матричных) процессоров. Кроме того, модель должна обеспечиваться возможность распараллеливания обмена между оперативной памятью и внешней запоминающей средой (традиционными внешними носителями информации или хранилищами данных).

4. ***Оптимизация запросов.*** Модель должна обеспечивать возможность оптимизации процессов совместной обработки нескольких агрегатов данных, реализующих запросы. Это означает, что в терминах модели процесс, реализующий запрос, должен иметь формальное представление в виде алгебраического выражения, которое можно либо автоматически синтезировать с заданными характеристиками, либо преобразовывать для улучшения его характеристик. Спецификацией для построения выражения может служить неформальное описание запроса, например, это может быть набор, содержащий описания входных, данных, правил преобразований атрибутов (формул для вычисления значений) и результата. Формальное описание запроса на языке любой модели данных, например, SQL-модели, также может служить спецификацией для синтеза выражения, или его оптимизации в процессе трансляции на язык связующей модели. Следовательно, связующая модель должна содержать средства, с помощью которых возможно реализовать синтез нового оптимального процесса и оптимизацию имеющегося процесса посредством эквивалентных преобразований, оптимизировать имеющийся процесс. Поскольку методы оптимизации процесса, как правило, имеют высокую вычислительную сложность, в большинстве случаев оперативная оптимизация может быть затруднительной. Поэтому целесообразно оптимизировать многократно выполняющиеся процессы. Вместе с тем, параллельная реализация некоторых методов оптимизации позволяет надеяться на возможность оперативной оптимизации процессов. Проблема оптимизации запросов имеет давнюю историю и множество различных подходов к ее решению [2-4]. Авторы считают, что достижение хороших результатов оптимизации возможно в том случае, когда используются методы, учитывающие особенности не только модели данных, но и модели вычислений [5].

5. ***Объектно-ориентированная парадигма.*** Это требование означает, что рассматриваемые в статье формальные модели должны обеспечивать возможность применения современных объектно-ориентированных методов проектирования и программирования. Вместе с тем, принятый сегодня подход к описанию объектных моделей данных имеет сугубо технологический характер [2-4], ведущий свою родословную от первых работ в этой области [6]. Технологический характер выражается в том, что отсутствует строгое определение объекта, которое заменяется неформальным описанием его свойств и свойств систем баз данных, построенных на основе объектно-ориентированного подхода. Это приводит к тому, что даже активные сторонники и основоположники объектного подхода к построению систем баз данных указывают на его недостаток, состоящий в отсутствии строго определения объекта, то есть «объект — это все, что угодно» [7]. Это требование не случайно завершает перечисление всего набора требований к моделям данных, поскольку объектный подход к проектированию программно-аппаратных комплексов обеспечивает наилучшую реализацию всех

остальных требований [8]. Рассматриваемые в статье модели построены именно как объектные, на основе предложенного в [9] строгого определения абстрактного типа данных (объекта, класса) как универсальной многоосновной алгебраической системы.

Построение моделей данных

Для построения моделей данных предлагается выбрать базовые абстрактный тип данных (АТД), свойства которого будут достаточными для того, чтобы, используя механизм наследования, можно было бы строить необходимые универсальные алгебры или алгебраические системы для использования их в качестве моделей данных.

Среди множества произвольных АТД предлагается рассмотреть специфический АТД, называемый в дальнейшем *универсальной алгебраической машиной* [10, 11].

Определение 1. Универсальная алгебраическая машина — это двухосновная алгебраическая система вида $\boldsymbol{E}$=<S, T; Ω; П>. Основа S называется структурой, а основа T — типом.

Структура представляет собой некоторую конструкцию, составленную из экземпляров данного *типа*. Примеры такого рода структур — векторы, матрицы, графы. Выбор структуры и типа определяется особенностями решаемой задачи. Причем для некоторых классов задач одной структуре могут соответствовать несколько типов. Следующий пример иллюстрирует эту ситуацию.

Пример 1. Для решения задач: поиск кратчайших путей, определение доступности вершин графа, разузлование, — может быть применен метод, основанный на алгоритме вычисления транзитивного замыкания квадратной матрицы M. Эта матрица задает отношение объектов некоторой предметной области: населенных пунктов и дорог, которые их соединяют, изделий и узлов, и деталей, из которых они состоят, и тому подобных. Транзитивное замыкание матрицы M вычисляется по следующей формуле

$$M^* = \sum_{i=1}^{K} M^i,\ M^i \neq Z, \text{для всех } i \leq K, 8\ M^{K+1} = Z,$$

где Z – нуль-матрица

В этом случае абстрактная алгебраическая машина имеет следующий вид $\boldsymbol{E_M}$=<M, X; Ω; П>, где X–тип, а M — множество квадратных матриц, составленных из элементов типа. Минимальное требование к типу X состоит в том, чтобы на X были определены две алгебраические операции, одна из которых трактуется как аддитивная, а вторая — как мультипликативная. То есть тип X должен быть по каждой из этих операций, по крайней мере, алгебраической структурой, называемой группоидом. В реальных задачах типами могут быть достаточно сложные алгебраические структуры, такие как кольца и поля. В таблице 1 приведены формальные определения и описания операций сигнатуры Ω абстрактной алгебраической машины $\boldsymbol{E_M}$.

Т а б л и ц а 1. **Сигнатура операций** E_M

T a b l e 1. **Operation signature** E_M

Операция	**Описание операции**
$\oplus : X \times X \to X$	аддитивная операция над элементами матриц;
$\otimes : X \times X \to X$	мультипликативнаяоперация над элементами матриц;
$^{T} : M \to M$	транспонирование матрицы;
$+ : M \times M \to M$	сумма матриц;
$\times : M \times M \to M$	произведение матриц;
$\mathrm{D} : M \to X$	определитель матрицы.

В реальных задачах в роли типа X могут быть, такие множества как:

- в «задаче разузлования» — множество неотрицательных действительных чисел R^0, с аддитивной операцией сложения и мультипликативной операцией умножения чисел, Ω={**+**, **×**};
- в задаче вычисления кратчайших путей в графе — множество положительных действительных чисел R^+, с аддитивной операцией вычисления минимума из двух чисел, и мультипликативной операцией сложения, Ω={*min*, **+**};
- в задаче определения доступности вершин в графе множество {0, 1} с аддитивной операцией дизъюнкции и мультипликативной операцией конъюнкции, Ω={∨, ∧}.

Операции над матрицами, входящие в сигнатуру операций Ω, реализуются хорошо известными последовательными и параллельными стандартными алгоритмами, о которых речь пойдет далее.

Важность рассмотренного примера состоит, прежде всего, в том, что он наглядно показывает наличие универсальных двухосновных алгебраических систем, обладающих качеством, которое можно сформулировать как возможность замены одного основного множества, а именно, типа и операций над его элементами, при сохранении другого множества и алгоритмов, реализующих операции над его элементами. Этот факт можно сформулировать в виде очевидного утверждения.

Пусть S — структура, а T_1, ..., T_n — допустимые для этой структуры типы. Тогда T_1, ..., T_n — однотипные универсальные алгебраические системы. То есть, можно установить такое взаимно-однозначное соответствие между их сигнатурами операций Ω и Ω', при котором любая операция $F \in \Omega$ и соответствующая ей операция $F' \in \Omega'$ будут n-арными с одним и тем же n.

Практическая ценность универсальных алгебраических машин состоит в том, что суть операций над элементами структуры S не изменяется при изменении сути операций над элементами типа T. Это свойство универсальных алгебраических машин может быть полезным в практическом программировании. Такой подход особенно полезен при разработке программно-аппаратных комплексов, ориентированных на параллельную обработку больших объемов данных [12-17]. В этом случае операции над структурами реализуются алго-

ритмами, сложность программирования и отладки которых во много раз превышает сложность разработки соответствующих им последовательных алгоритмов. Подтверждением этого тезиса служит разработка алгоритмов последовательного и параллельного умножения матриц. Программирование и отладка трех вложенных циклов не идет ни в какое сравнение по сложности с разработкой алгоритмов Фокса или Кэннона.

Если типы T_1, ..., T_n — гомоморфные или изоморфные универсальные алгебраические системы, то становится возможной отладка операций над структурой на наиболее простом типе данных. Отлаженная таким образом структура становится базовым АТД, от которого можно порождать конкретные АТД (реализации), предназначенные для решения задач на сложных типах данных.

При таком подходе реальные алгебраические машины, работающие с конкретными типами данных, разрабатываются как АТД — наследники абстрактных алгебраических машин, у которых они наследуют операции над структурами и включают в себя операции над реальными элементами этих структур.

Алгебраические системы сложных типов данных

Один из основных сложных типов данных, используемых в современных моделях данных, — это структурный тип, называемый кортежем. В языках программирования для задания кортежей используются конструкции ***struct*** (***C***-подобные языки) и ***record*** (***Pascal*** и ему подобные языки). Поэтому проблема построения АТД, соответствующих реальным алгебраическим машинам, на основе которых строятся такие распространенные модели данных как реляционная, многомерная, NoSQL, неотделима от проблемы создания алгебры кортежей произвольной структуры. Далее рассматривается способ построения бинарных операций над кортежами, без которых невозможно построение бинарных операций над агрегатами данных, которые используются в качестве элементов структуры (отношениями, файлами, многомерными матрицами). В различных моделях данных эти операции над кортежами, заданные явно или неявно, интерпретируются либо как аддитивные, либо как мультипликативные. Предложенный метод позволяет делать явные формальные описания различных, в том числе бинарных операций над кортежами.

Обычно используется общепринятое в математике определение кортежа: кортеж — это конечный набор (t_1, ..., t_n) длины n, (где n — неотрицательное целое число), каждый элемент которого t_i *принадлежит* некоторому типу T_i ($1 \le i \le n$).

Однако, для решения поставленной задачи необходимо более существенная формализация этого понятия, описанию которой посвящено дальнейшее изложение.

Пусть *Atr* — универсальное множество имен атрибутов *AS* — конечное множество алгебраических систем. В качестве алгебраических систем, включенных в *AS*, могут быть, «простые» типы данных, например, ***real*** = **< *R***;+, –, ×, :; <, >, ≤, ≥, ≠, = **>** — система действительных чисел, ***string*** = **< *S***; +; <, = **>** — система строковых величин, ***boolean*** = **<{0, 1}**; ¬, ∨, ∧; ~ > – система булевых значений, Z_5 = <{K_0, K_1, K_2, K_3, K_4};+, –, ×, = > – система классов вычетов по модулю 5. Предполагается, что в любой алгебраической системе определены отношения «=» (равно) и «≠» (не равно) для элементов основного множества (носителя), и поэтому они часто не упоминаются при описании системы.

Для дальнейшей формализации необходимо определить понятие объекта типа *Sh* над множествами *Atr* и *AS*.

Определение 2. Понятие *объекта типа Sh или объекта со схемой Sh* над множеством имен атрибутов *Atr* и базовым множеством алгебраических систем *AS* определяется следующим образом:

1. Если a — элемент носителя dom(*AlgSyst*) алгебраической системы *AlgSyst* из *AS*, то a называется атомным объектом типа *AlgSyst* или атомом типа *AlgSyst*.
2. Для каждого типа T существуют два специальных объекта ∇ — произвольный объект типа T и Δ — неопределенный объект для типа T.
3. Если O_1, ..., O_n — объекты типов T_1, ..., T_n соответственно и a_1, ..., a_n — различные имена атрибутов из множества *Atr*, то объект (a_1:O_1, ..., a_n:O_n) называется кортежным объектом типа (a_1:T_1, ..., a_n: T_n) или кортежным объектом со схемой (a_1:T_1, ..., a_n: T_n). В дальнейшем, для простоты под словом кортеж будет пониматься именно кортежный объект.

Важную роль в процессе построения моделей данных играют специфические кортежи вида (a_1:Δ_1, ..., a_n:Δ_n). Они необходимы для формализации операций над агрегатами данных. Поскольку в практическом программировании вместо неопределенных элементов Δ обычно используют нейтральные элементы типов, то для этих кортежей будет использоваться термин «нуль-кортеж». Поэтому в дальнейшем нуль-кортеж рассматривается как кортеж, состоящий только из нейтральных элементов типов T_1, ..., T_n.

На следующем этапе построения формальной алгебры-модели кортежей рассматриваются только кортежи, которые не могут содержать сложные элементы, например, другие кортежи.

Пусть T_1,..., T_n — совокупность основных множеств универсальных алгебр или алгебраических систем, которые принято в языках программирования называть ***простыми типами***. Это, как было показано, могут быть числа с фиксированной или плавающей точкой, строки, а также типы, полученные из простых типов добавлением новых операций. Пусть x_1, ..., x_p, y_1,..., y_q, ($0 \le p, q \le n, p+q=n$) — набор переменных (атомов), каждая из которых принимает значения в одном и только одном из множеств T_1,..., T_m. На этих множествах определяется система функций:

$$f^{j}_{\alpha_1 \dots \alpha_k, \beta_1 \dots \beta_l}(x_{\alpha_1}, \dots, x_{\alpha_k}, y_{\beta_1}, \dots, y_{\beta_l}) : (T_{\alpha_1} \times \dots \times T_{\alpha_k} \times T_{\beta_1} \times \dots \times T_{\beta_l}) \to T_i.$$

Здесь выполняются неравенства $1 \le k \le p$, $1 \le l \le q$, $j > 0$ и $1 \le i \le n$. Далее будет использоваться сокращенная запись этих функций — $f^{j}_{\alpha_1 \dots \alpha_k, \beta_1 \dots \beta_l}$.

Определение 3. Пусть кортеж c_1 длины p и кортеж c_2 длины q составлены из переменных x_1, ..., x_p и y_1,..., y_q соответственно. Тогда кортеж c_3 длины r, построенный по правилу $(f^{1}_{\alpha_1 \dots \alpha_k, \beta_1 \dots \beta_l}, \dots, f^{r}_{\alpha_1 \dots \alpha_k, \beta_1 \dots \beta_l})$, *можно рассматривать как результат бинарной операции над кортежами* c_1 и c_2 ($c_3 = c_1 * c_2$). Если функция определена на всех элементах кортежей c_1 и c_2, то используется обозначение $f^{j}_{c_1, c_2}$

Семантика операции над кортежами (аддитивность или мультипликативность) определяется семантикой операции, определенной над структурой, типами элементов которой могут

быть кортежи c_1, c_2 и c_3.

В следующем примере показано построение бинарных операций над кортежами.

Пример 2. Пусть S — множество строк, а R^+ — множество положительных действительных чисел. Переменные A, B, C, D принимают значения в множестве S, а переменные X, Y — в множестве R^+. Кортежи c_1 и c_2 имеют схемы $c_1(A, B, C, X)$ и $c_2(B, C, D, Y)$. Функции:

$$f^1_{c_1.A_1}(a) = a, (a \in A_1),$$
$$f^2_{c_2.D}(d) = d, (d \in D),$$
$$f^3_{c_1.X, c_2.Y}(x, y) = x \times y, (x \in X, y \in Y)$$

позволяют построить кортеж $c_3=c_1\times c_2$ со схемой $c_3(A, D, Z)$, где $z= x\times y$, и принимает значения в множестве R^+. Таким образом определяется мультипликативная операция над кортежами.

Пример 3. Если c_{31}, c_{32}, c_{33} — кортежи со схемой $c_3(A, D, Z)$, то функции

$$f^1_{c_3.A}(c_{31}.a) = c_{31}.a, (a \in A),$$
$$f^2_{c_3.D}(c_{31}.d) = c_{31}.d, (d \in D)$$
$$f^3_{c_3.Z, c_3.Z}(c_{31}.z, c_{32}.z) = Min(c_{31}.z, c_{32}.z), (z \in Z)$$

определяют аддитивную операцию над кортежами $c_{33}=c_{31} + c_{32}$.

Если предположить, что в реляционной модели кортежи c_1 и c_2 есть элементы (строки) отношений R_1 и R_2, то рассмотренные функции позволяют сформировать список полей в выражении, реализующем запрос:

```
SELECT R1.A, R2.D, MIN(R1.X*R2.Y) AS Z
FROM R1 INNER JOIN R2 ON (R1.A = R2.B) AND (R1.C = R2.D)
GROUP BY R1.A, R2.D;
```

Подобным образом рассмотренные функции могут быть использованы для построения алгебр элементов структур (файлов и многомерных матриц) в теоретико-множественной и многомерно матричной моделях.

В некоторых моделях при построении бинарных операций над элементами структуры может возникнуть ситуация, когда кортеж-результат формируется только из одного кортежа-операнда [18-20]. В многомерно матричной модели такое невозможно, так как в матрице присутствуют все возможные элементы, в том числе и нуль-кортежи. Но в реляционной и теоретико-множественной моделях отношения и файлы, как правило, не содержат строки и записи, состоящие только из нейтральных элементов. Поэтому целесообразно определить еще два вида функций $f^j_{\alpha_1\ldots\alpha_k,0}(x_{\alpha_1},\ldots,x_{\alpha_k}) : (T_{\alpha_1}\times\ldots\times T_{\alpha_k}) \to T_j$, и $f^j_{0,\beta_1\ldots\beta_l}(y_{\beta_1},\ldots,y_{\beta_l}) : (T_{\beta_1}\times\ldots\times T_{\beta_l}) \to T_j$, которые позволят конструировать операции, формирующие кортеж-результат только из одного кортежа-операнда. Использование этих операций обеспечивает единство формальной записи алгоритмов бинарных операций над структурами. Сокращенная запись этих функций имеет вид: $f^j_{0,\beta_1\ldots\beta_l}$, $f^j_{\alpha_1\ldots\alpha_k,0}$.

Предложенный способ позволяет определять различные аддитивные и мультипликативные операции над кортежами и получать при этом различные универсальные алгебры или алгебраические системы. Свойства построенных операций составляют список аксиом этих алгебраических систем. Благодаря этому возникает возможность формального построения произвольных универсальных алгебраических систем кортежей в соответствии с требованиями реальных задач [21-25].

От построенной таким образом универсальной алгебраической системы легко перейти к представлению АТД *Кортеж*. Такой АТД является базовым объектом, который может стать родоначальником множества объектов, соответствующих реальным задачам. Эти объекты будут, с одной стороны, наследовать свойства составляющих кортежи простых типов, с другой, базовые операции объекта, представляющего АТД *Кортеж*. Таким образом, в объектах-наследниках могут использоваться либо основные операции над кортежами из базового объекта, либо заменяющие их полиморфные операции, которые будут выполнять те действия над элементами простых типов данных, которые требуются в условиях конкретной решаемой задачи.

Выводы

Предложенный метод позволяет после определения операций кортежами, которые могут входить как составные части в другие кортежи, можно строить сложные иерархические кортежи. Такого рода структуры впервые появились в языке программирования COBOL и возможны в современных языках программирования. Поэтому построение алгебраических систем для реализации их обработки по-прежнему актуальная задача. В реляционной модели данных этот метод позволяет по-новому взглянуть на понятие атомарности. Иерархические кортежи дают возможность рассматривать построенные таким образом отношения как отношения в первой и последующей нормальных формах. Возможности современных языков манипулирования данными обеспечивают обработку этих отношений. Из чего можно сделать вывод о том, что реляционная модель есть объектная модель данных.

Кроме того, задание аддитивной и мультипликативной операций над кортежами позволяет рассматривать кортежные типы данных как типы элементов многомерных матриц, что обеспечивает возможность реализации операций сложения и (λ, μ)-свернутого произведения этих матриц. Такой подход позволяет доказать изоморфизм алгебры многомерных матриц и реляционной алгебры. Это в свою очередь дает возможность эффективной реализации параллельной обработки реляционных БД на программно-аппаратных комплексах, архитектура которых основана на многомерно-матричной модели вычислений.

Список использованных источников

[1] Munerman, V. Realization of Distributed Data Processing on the Basis of Container Technology / V. Munerman, D. Munerman. — DOI 10.1109/EIConRus.2019.8656766 // 2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). — Saint Petersburg and Moscow, Russia, 2019. — Pp. 1740-1744. — URL: https://ieeexplore.ieee.org/document/8656766 (дата обращения: 16.08.2020).

[2] Кузнецов, С. Д. Основы баз данных. — М.: ИНТУИТ.ру, 2007. — URL: https://www.elibrary.ru/item.

asp?id=19589854 (дата обращения: 16.08.2020).

[3] Гарсиа-Молина, Г. Системы баз данных. Полный курс / Г. Гарсиа-Молина, Д. Ульман, Д. Уидом. — М.: Изд. дом «Вильямс», 2004.

[4] Багуи, С. Объектно-ориентированные базы данных: достижения и проблемы / С. Багуи // Открытые системы.СУБД. — 2004. — № 03. — URL: https://www.osp.ru/os/2004/03/184042 (дата обращения: 16.08.2020).

[5] Мунерман, В. И. Оптимизация процессов и операций массовой обработки данных / В. И. Мунерман, Д. В. Мунерман // Системы компьютерной математики и их приложения. — 2020. — № 21. — С. 172-178. — URL: https://www.elibrary.ru/item.asp?id=44237972 (дата обращения: 16.08.2020). — Рез. англ.

[6] Liskov, B. Programming with abstract data types / B. Liskov, S. Zilles. — DOI 10.1145/942572.807045 // ACM SIGPLAN Notices. — 1974. — Vol. 9, issue 4. — Pp. 50-59. — URL: https://dl.acm.org/doi/10.1145/800233.807045 (дата обращения: 16.08.2020).

[7] Дейт, К. Дж. Введение в системы баз данных / К. Дж. Дейт. — М.: Изд. дом «Вильямс», 2008.

[8] Мунерман, В. И. Алгебраический подход к построению программно-аппаратных комплексов для повышения эффективности массовой обработки данных / В. И. Мунерман, Д. В. Мунерман // Современные информационные технологии и ИТ-образование. — 2015. — Т. 11, № 2. — С. 391-396. — URL: https://www.elibrary.ru/item.asp?id=26167520 (дата обращения: 16.08.2020). — Рез. англ.

[9] Глушков, В. М. Алгебра. Языки. Программирование / В. М. Глушков, Г. Е. Цейтлин, Е. Л. Ющенко. — 1-е изд. — Киев: Наукова думка, 1974.

[10] Мунерман, В. И. Построение архитектур программно-аппаратных комплексов для повышения эффективности массовой обработки данных / В. И. Мунерман // Системы высокой доступности. — 2014. — Т. 10, № 4. — С. 3-16. — URL: https://www.elibrary.ru/item.asp?id=22831892 (дата обращения: 16.08.2020). — Рез. англ.

[11] Емельченков, Е. П. Алгебраический подход к оптимизации разработки и эксплуатации систем управления базами данных / Е. П. Емельченков, Н. А. Левин, В. И. Мунерман // Системы и средства информатики. — 2009. — Т. 19, №. 2. — С. 114-137. — URL: https://www.elibrary.ru/item.asp?id=13053911 (дата обращения: 16.08.2020). — Рез. англ.

[12] Емельченков, Е. П. О математическом аппарате информационно-логического обеспечения САПР ТП / Е. П. Емельченков, П. П. Крюков, Ю. С. Малеин // Автоматика и телемеханика. — 1990. — № 4. — С. 177-183.

[13] Yemelchenkov, Y. P. Functional dependencies in hierarchical structures of data / Y. P. Yemelchenkov, M. S. Tsalenko. — DOI 10.1007/3-540-54009-1_19 // MFDBS 91. MFDBS 1991. Lecture Notes in Computer Science; B. Thalheim, J. Demetrovics, H. D. Gerhardt (ed.). Springer, Berlin, Heidelberg. — 1991. — Vol. 495. — Pp. 258-275. — URL: https://link.springer.com/chapter/10.1007/3-540-54009-1_19 (дата обращения: 16.08.2020).

[14] Zakharov, V. Architecture of Software-Hardware Complex for Searching Images in Database / V. Zakharov, A. Kirikova, V. Munerman, T. Samoilova. — DOI 10.1109/EIConRus.2019.8657241 // 2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). — Saint Petersburg and Moscow, Russia, 2019. — Pp. 1735-1739. — URL: https://ieeexplore.ieee.org/document/8657241 (дата обращения: 16.08.2020).

[15] Baroody, A. J. An object-oriented approach to database system implementation / A. J. Baroody, D. J. DeWitt. — DOI 10.1145/319628.319645 // ACM Transactions on Database Systems. — 1981. — Vol. 6, no. 4. — Pp. 576-601. — URL: https://dl.acm.org/doi/10.1145/319628.319645 (дата обращения: 16.08.2020).

[16] Zhao, L. An Object-Oriented Data Model for Database Modelling, Implementation and Access / L. Zhao, S. A. Roberts. — DOI 10.1093/comjnl/31.2.116 // The Computer Journal. — 1988. — Vol. 31, issue 2. — Pp. 116-124. — URL: https://academic.oup.com/comjnl/article/31/2/116/352255 (дата обращения: 16.08.2020).

[17] Conrad, S. Database Design: Object-Oriented versus Relational / S. Conrad, G. Saake, I. Schmitt, C. Türker. — DOI 10.1007/978-3-322-84795-9_7 // Entwicklungsmethoden für Informationssysteme und deren Anwendung. Teubner-Reihe Wirtschaftsinformatik; R. Kaschek (ed.). Viewег+Teubner Verlag. — 1999. — Pp. 109-125. — URL: https://link.springer.com/chapter/10.1007/978-3-322-84795-9_7 (дата обращения: 16.08.2020).

[18] Zhang, Yu. Binary Structuring Elements Decomposition Based on an Improved Recursive Dilation-Union Model and RSAPSO Method / Yu. Zhang, Sh. Wang, Y. Sun, G. Ji, P. Phillips, Z. Dong. — DOI 10.1155/2014/272496 // Mathematical Problems in Engineering. — 2014. — Vol. 2014, Article 272496. — URL: https://www.hindawi.com/journals/mpe/2014/272496 (дата обращения: 16.08.2020).

[19] Shih, F. Y. Decomposition of binary morphological structuring elements based on genetic algorithms / F. Y. Shih, Y.-T. Wu. — DOI 10.1016/j.cviu.2005.01.001 // Computer Vision and Image Understanding. — 2005. — Vol. 99, issue 2. — Pp. 291-302. — URL: https://www.sciencedirect.com/science/article/pii/S1077314205000020 (дата обращения: 16.08.2020).

[20] Zhang, Y. Recursive Structure Element Decomposition Using Migration Fitness Scaling Genetic Algorithm / Y. Zhang, L. Wu L. — DOI 10.1007/978-3-642-21515-5_61 // Advances in Swarm Intelligence. ICSI 2011. Lecture Notes in Computer Science; Y. Tan, Y. Shi, Y. Chai, G. Wang (ed.). Springer, Berlin, Heidelberg. — 2011. — Vol. 6728. — Pp. 514-521. — URL: https://link.springer.com/chapter/10.1007/978-3-642-21515-5_61 (дата обращения: 16.08.2020).

[21] Яценко, Е. А. Обзор объектно-ориентированной парадигмы в приложении к разработке баз данных / Е. А. Яценко. — DOI 10.25205/1818-7900-2019-17-3-123-134 // Вестник Новосибирского государственного университета. Серия: Информационные технологии. — 2019. — Т. 17, № 3. — С. 123-134. — URL: https://www.elibrary.ru/item.asp?id=41570985 (дата обращения:

16.08.2020). — Рез. англ.

[22] Hudson, S. E. The Efficient Support of Functionally-Defined Data in Cactis / S. E. Hudson, R. King. — DOI 10.1007/978-3-642-84374-7_21 // On Object-Oriented Database Systems. Topics in Information Systems; K. R. Dittrich, U. Dayal, A. P. Buchmann (ed.). Springer, Berlin, Heidelberg. — 1991. — Pp. 341-355. — URL: https://link.springer.com/chapter/10.1007/978-3-642-84374-7_21 (дата обращения: 16.08.2020).

[23] Thearle, R. W. A Survey Of Object Oriented Database Systems / R. W. Thearle. — DOI 10.4324/9780429441110 // Object Management; R. Tagg, J. Mabon (ed.). Routledge. — 2019. — 8 pp.

[24] Dietrich, S. W. Fundamentals of Object Databases: Object-Oriented and Object-Relational Design. Synthesis Lectures on Data Management / S. W. Dietrich, S. D. Urban. — DOI 10.2200/S00315ED1V01Y201012DTM012. — Morgan & Claypool Publishers, 2010.

[25] Hong, S. A formal approach to the comparison of object-oriented analysis and design methodologies / S. Hong, G. van den Goor, S. Brinkkemper. — DOI 10.1109/HICSS.1993.284253 // [1993] Proceedings of the Twenty-sixth Hawaii International Conference on System Sciences. — Wailea, HI, USA, 1993. — Vol. 4. — Pp. 689-698. — URL: https://ieeexplore.ieee.org/document/284253 (дата обращения: 16.08.2020).

Поступила 16.08.2020; одобрена после рецензирования 25.10.2020; принята к публикации 03.11.2020.

Об авторах:

Емельченков Евгений Петрович, заведующий кафедрой информатики, физико-математический факультет, ФГБОУ ВО «Смоленский государственный университет» (214000, Российская Федерация, г. Смоленск, ул. Пржевальского, д. 4), кандидат физико-математических наук, доцент, ORCID: http://orcid.org/0000-0002-6589-694X, ypy1101@gmail.com

Мунерман Виктор Иосифович, доцент кафедры информатики, физико-математический факультет, ФГБОУ ВО «Смоленский государственный университет» (214000, Российская Федерация, г. Смоленск, ул. Пржевальского, д. 4), кандидат технических наук, доцент, ORCID: http://orcid.org/0000-0002-9628-4049, vimoon@gmail.com

Мунерман Даниил Викторович, лаборант-стажер кафедры информатики, физико-математический факультет, ФГБОУ ВО «Смоленский государственный университет» (214000, Российская Федерация, г. Смоленск, ул. Пржевальского, д. 4), ORCID: http://orcid.org/0000-0002-5139-6645, danvmoon@gmail.com

Самойлова Татьяна Аркадьевна, доцент кафедры информатики, физико-математический факультет, ФГБОУ ВО «Смоленский государственный университет» (214000, Российская Федерация, г. Смоленск, ул. Пржевальского, д. 4), кандидат технических наук, доцент, ORCID: http://orcid.org/0000-0002-3712-327X, tatsamilova24@gmail.com

Все авторы прочитали и одобрили окончательный вариант рукописи.

References

[1] Munerman V., Munerman D. Realization of Distributed Data Processing on the Basis of Container Technology. In: *2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*. Saint Petersburg and Moscow, Russia; 2019. p. 1740-1744. (In Eng.) DOI: https://doi.org/10.1109/EIConRus.2019.8656766

[2] Kuznetsov S.D. *Osnovy baz dannyh* [Database Basics]. INTUIT.ru, Moscow; 2005. Available at: https://www.elibrary.ru/item.asp?id=19589854 (accessed 16.08.2020). (In Russ.)

[3] Garcia-Molina H., Ullman J.D., Widom J. Database Systems: The Complete Book. 1st ed. Prentice Hall; 2001. (In Eng.)

[4] Bagui S. Object-Oriented Databases: Achievements and Challenges. *Open Systems.DBMS*. 2004; (03). Available at: https://www.osp.ru/os/2004/03/184042 (accessed 16.08.2020). (In Russ.)

[5] Munerman V.I., Munerman D.V. Optimization of Processes and Operations of Mass Data Processing. *Sistemy komp'yuternoy matematiki i ikh prilozheniya* = Computer Mathematics Systems and Their Applications. 2020; (21):172-178. Available at: https://www.elibrary.ru/item.asp?id=44237972 (accessed 16.08.2020). (In Russ., abstract in Eng.)

[6] Liskov B., Zilles S. Programming with abstract data types. *ACM SIGPLAN Notices*. 1974; 9(4):50-59. (In Eng.) DOI: https://doi.org/10.1145/942572.807045

[7] Date C.J. An Introduction to Database Systems. 8th ed. Pearson; 2003. (In Eng.)

[8] Munerman V.I., Munerman D.V. Algebraic approach to the construction of software and hardware systems to improve the efficiency of mass data processing. *Sovremennye informacionnye tehnologii i IT-obrazovanie* = Modern Information Technologies and IT-Education. 2015; 11(2):391-396. Available at: https://www.elibrary.ru/item.asp?id=26167520 (accessed 16.08.2020). (In Russ., abstract in Eng.)

[9] Gluschkow W.M., Zeitlin G.E., Justchenko J.L. Algebra. Sprachen. Programmierung. Akademie-Verlag, Berlin; 1980. (In Eng.)

[10] Munerman V.I. Construction of Hardware-Software Complexes Architecture to Improve Massively Data Processing. *Highly available systems*. 2014; 10(4):3-16. Available at: https://www.elibrary.ru/item.asp?id=22831892 (accessed 16.08.2020). (In Russ., abstract in Eng.)

[11] Emelchenkov E., Levin N., Munerman V. The Algebraic Approach to Optimization of Development and Operation of Data Base Management Systems. *Sistemy i sredstva informatiki* = Systems and Means of Informatics. 2009; 19(2):114-137. Available at: https://www.elibrary.ru/item.asp?id=13053911 (accessed 16.08.2020). (In Russ., abstract in Eng.)

[12] Yemelchenkov E.P., Kryukov P.P., Malein Yu.S. On the mathematical tools of the informational and logic support of computer-aided process design systems. *Avtomatika i Telemekhanika* = Automation and Remote Control. 1990; (4):177-183. (In Russ., abstract in Eng.)

[13] Yemelchenkov Y.P., Tsalenko M.S. Functional dependencies in hierarchical structures of data. In: Thalheim B.,

Demetrovics J., Gerhardt H.D. (ed.) MFDBS 91. MFDBS 1991. *Lecture Notes in Computer Science*. 1991; 495:258-275. Springer, Berlin, Heidelberg. (In Eng.) DOI: https://doi.org/10.1007/3-540-54009-1_19

[14] Zakharov V., Kirikova A., Munerman V., Samoilova T. Architecture of Software-Hardware Complex for Searching Images in Database. In: *2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*. Saint Petersburg and Moscow, Russia; 2019. p. 1735-1739. (In Eng.) DOI: https://doi.org/10.1109/EIConRus.2019.8657241

[15] Baroody A.J., DeWitt D.J. An object-oriented approach to database system implementation. *ACM Transactions on Database Systems*. 1981; 6(4):576-601. (In Eng.) DOI: https://doi.org/10.1145/319628.319645

[16] Zhao L., Roberts S.A. An Object-Oriented Data Model for Database Modelling, Implementation and Access. *The Computer Journal*. 1988; 31(2):116-124. (In Eng.) DOI: https://doi.org/10.1093/comjnl/31.2.116

[17] Conrad S., Saake G., Schmitt I., Türker C. Database Design: Object-Oriented versus Relational. In: Kaschek R. (ed.) Entwicklungsmethoden für Informationssysteme und deren Anwendung. *Teubner-Reihe Wirtschaftsinformatik*. Vieweg+Teubner Verlag; 1999. p. 109-125. (In Eng.) DOI: https://doi.org/10.1007/978-3-322-84795-9_7

[18] Zhang Yu., Wang Sh., Sun Y., Ji G., Phillips P., Dong Z. Binary Structuring Elements Decomposition Based on an Improved Recursive Dilation-Union Model and RSAPSO Method. *Mathematical Problems in Engineering*. 2014; 2014:272496. (In Eng.) DOI: https://doi.org/10.1155/2014/272496

[19] Shih F.Y., Wu Y.-T. Decomposition of binary morphological structuring elements based on genetic algorithms. *Computer Vision and Image Understanding*. 2005; 99(2):291-302. (In Eng.) DOI: https://doi.org/10.1016/j.cviu.2005.01.001

[20] Zhang Y., Wu L. Recursive Structure Element Decomposition Using Migration Fitness Scaling Genetic Algorithm. In: Tan Y., Shi Y., Chai Y., Wang G. (ed.) Advances in Swarm Intelligence. ICSI 2011. *Lecture Notes in Computer Science*. 2011; 6728:514-521. Springer, Berlin, Heidelberg. (In Eng.) DOI: https://doi.org/10.1007/978-3-642-21515-5_61

[21] Yatsenko E.A. Overview of Object-Oriented Paradigm in an Appendix to the Development of Databases. *Vestnik NSU. Series: Information Technologies*. 2019; 17(3):123-134. (In Russ., abstract in Eng.) DOI: https://doi.org/10.25205/1818-7900-2019-17-3-123-134

[22] Hudson S.E., King R. The Efficient Support of Functionally-Defined Data in Cactis. In: Dittrich K.R., Dayal U., Buchmann A.P. (ed.) On Object-Oriented Database Systems. *Topics in Information Systems*. Springer, Berlin, Heidelberg; 1991. p. 341-355. (In Eng.) DOI: https://doi.org/10.1007/978-3-642-84374-7_21

[23] Thearle R.W. A Survey Of Object Oriented Database Systems. In: Tagg R., Mabon J. (ed.). Object Management. Routledge; 2019. 8 pp. (In Eng.) DOI: https://doi.org/10.4324/9780429441110

[24] Dietrich S.W., Urban S.D. Fundamentals of Object Databases: Object-Oriented and Object-Relational Design. Synthesis Lectures on Data Management. Morgan & Claypool Publishers; 2010. (In Eng.) DOI: https://doi.org/10.2200/S00315ED1V01Y201012DTM012

[25] Hong S., van den Goor G., Brinkkemper S. A formal approach to the comparison of object-oriented analysis and design methodologies. In: *[1993] Proceedings of the Twenty-sixth Hawaii International Conference on System Sciences*. Wailea, HI, USA. 1993; 4:689-698. (In Eng.) DOI: https://doi.org/10.1109/HICSS.1993.284253

Submitted 16.08.2020; approved after reviewing 25.10.2020; accepted for publication 03.11.2020.

About the authors:

Evgenii P. Emelchenkov, Head of the Department of Computer Science, Faculty of Physics and Mathematics, Smolensk State University (4 Przhevalsky St., Smolensk 214000, Russian Federation), Ph.D. (Phys.-Math.), Associate Professor, ORCID: http://orcid.org/0000-0002-6589-694X, ypy1101@gmail.com

Victor I. Munerman, Associate Professor of the Department of Computer Science, Faculty of Physics and Mathematics, Smolensk State University (4 Przhevalsky St., Smolensk 214000, Russian Federation), Ph.D. (Engineering), Associate Professor, ORCID: http://orcid.org/0000-0002-9628-4049, vimoon@gmail.com

Daniel V. Munerman, Laboratory Assistant of the Department of Computer Science, Faculty of Physics and Mathematics, Smolensk State University (4 Przhevalsky St., Smolensk 214000, Russian Federation), ORCID: http://orcid.org/0000-0002-5139-6645, danvmoon@gmail.com

Tatyana A. Samoilova, Associate Professor of the Department of Computer Science, Faculty of Physics and Mathematics, Smolensk State University (4 Przhevalsky St., Smolensk 214000, Russian Federation), Ph.D. (Engineering), Associate Professor, ORCID: http://orcid.org/0000-0002-3712-327X, tatsamilova24@gmail.com

All authors have read and approved the final manuscript.