УДК 681.3

Ю.Г. Шаталова, доцент, канд. техн. наук
Севастопольский национальный технический университет
ул. Университетская 33, г. Севастополь, Украина, 99053
E-mail: volnajulia@mail.ru

МОДЕЛИРОВАНИЕ ПРОЦЕССА ОБРАБОТКИ ЗАПРОСОВ К РАСПРЕДЕЛЕННОЙ БАЗЕ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ СЕТИ ПЕТРИ-МАРКОВА

Анализируются модели организации взаимодействия клиентов и распределенной базы данных. Обосновывается выбор трехуровневой модели. Строится модель выбранной архитектуры на основе сети Петри-Маркова.

***Ключевые слова**: сеть Петри-Маркова, распределенная база данных, сервер приложений.*

Распределенные базы данных в условиях развития и повсеместного использования компьютерных сетей приобрели отличный от классического представления смысл. Если первоначально распределенные системы разрабатывались с семантическим распределением, то сейчас это распределение дополнительно обосновывается технологическими аспектами. Теперь распределенные базы данных действительно распределены территориально. Различные объекты одной базы данных могут располагаться на удаленных компьютерах и управляться различными системами управления базами данных (СУБД). Тем не менее, основные функции базы данных (БД) остались прежними. Распределенная система должна обеспечить выполнение следующих условий.

1. БД в каждый момент должна отражать текущее состояние предметной области, которое определяется не только собственно данными, но и связями между объектами данных. То есть данные, которые хранятся в БД, в каждый момент времени должны быть непротиворечивыми и актуальными.

2. БД должна отражать некоторые правила предметной области, законы, по которым она функционирует (*Business rules*).

3. Требуется постоянный контроль за состоянием БД, отслеживание всех изменений и адекватная реакция на них.

4. Необходимо, чтобы возникновение некоторой ситуации в БД четко и оперативно влияло на ход выполнения прикладной задачи.

5. Одной из важнейших проблем СУБД является контроль типов данных. В настоящий момент СУБД контролирует синтаксически только стандартно-допустимые типы данных, то есть такие, которые определены в *DDL* (*Data Definition Language*) — языке описания данных [1]. Однако, в реальных предметных областях могут возникнуть данные, которые содержат еще и семантическую составляющую, что может породить необходимость ввести дополнительные ограничения целостности на данные и тогда СУБД должна отслеживать и эти ограничения.

Кроме перечисленных задач, непосредственно связанных с БД на систему возлагаются дополнительные «организационные» задачи: надежность, быстродействие, гарантированное обслуживание пользовательских запросов, проверка запросов на право доступа и т.д. При повышении количества пользователей на базу данных возрастает нагрузка. Поэтому актуальной задачей является выбор архитектуры распределенной базы данных, обеспечивающей выполнение перечисленных требований.

Задача выбора архитектуры распределенной базы данных распадается на две подзадачи.

1. Проектирование распределенной базы данных.

2. Выбор модели организации взаимодействия клиентов базы данных и СУБД.

Целью данной работы является анализ возможных путей решения второй подзадачи, обоснование выбора наиболее подходящего из них и построение модели выбранной архитектуры на основе сети Петри-Маркова.

Как бы ни была организована база данных в основе ее работы лежит модель «клиент -сервер». В открытых системах клиенты и сервер располагаются на различных компьютерах. Причем, поскольку все компьютеры в сети обладают собственными ресурсами, то разумно так распределить нагрузку на них, чтобы максимальным образом использовать их потенциал.

Первая задача, которая должна быть решена для обеспечения распределения нагрузки — разбить весь необходимый набор функций, обеспечивающих корректное взаимодействие на части и реализовать их на различных компонентах системы.

Основной принцип технологии «клиент-сервер» применительно к технологии баз данных заключается в разделении функций на 5 групп [1]:

— функции ввода и отображения данных;

— прикладные функции, определяющие основные алгоритмы решения задач приложения (бизнес-логика);

— функции обработки данных внутри приложения;

— функции управления информационными ресурсами базы данных;

— служебные функции, играющие роль связок между функциями первых четырех групп.

Распределение перечисленных функций между клиентом и сервером позволяет изменять модели системы. Рассмотрим некоторые из них.

Двухуровневая модель является результатом распределения пяти указанных функций между двумя процессами, которые выполняются на двух платформах: на клиенте и на сервере. К двухуровневым моделям относятся:

— модель удаленного управления данными;

— модель файлового сервера;

— модель сервера базы данных.

Модель удаленного управления данными и модель файлового сервера работают с пассивным сервером, большинство операций возлагается на клиента. Эти модели обладают следующими недостатками:

— высоким сетевым трафиком, который связан с передачей по сети множества блоков и файлов, необходимых приложению;

— узким спектром операций манипулирования с данными, который определяется только файловыми командами;

— отсутствием адекватных средств безопасности доступа к данным (защита только на уровне файловой системы);

— дублированием кода приложений на всех клиентах.

Модель сервера баз данных работает с активным сервером. В этой модели бизнес-логика разделена между клиентом и сервером. На сервере бизнес-логика реализована в виде процедур — специальных программных модулей, которые хранятся в БД и управляются непосредственно СУБД. Клиентское приложение обращается к серверу с командой запуска хранимой процедуры, а сервер выполняет эту процедуру и регистрирует все изменения в БД, которые в ней предусмотрены. Сервер возвращает клиенту данные, релевантные его запросу, которые требуются клиенту либо для вывода на экран, либо для выполнения части бизнес-логики, которая расположена на клиенте.

Достоинствами данной архитектуры являются следующие:

— трафик обмена информацией между клиентом и сервером резко уменьшается;

— осуществляется централизованный контроль с использованием механизма триггеров;

— сервер является активным, т. к. может инициировать обработку данных, используя механизм триггеров;

— процедуры и триггеры хранятся в словаре БД, они могут быть использованы несколькими клиентами, что существенно уменьшает дублирование алгоритмов обработки данных в разных клиентских приложениях.

Сервер базы данных выполняет следующие функции:

— осуществляет мониторинг событий, связанных с триггерами;

— обеспечивает автоматическое срабатывание триггеров при возникновении связанных с ними событий;

— обеспечивает исполнение внутренней программы каждого триггера;

— запускает хранимые процедуры по запросам пользователей;

— запускает хранимые процедуры из триггеров;

— возвращает требуемые данные клиенту;

— обеспечивает все функции СУБД: доступ к данным, контроль и поддержку целостности данных в БД, контроль доступа, обеспечение корректной параллельной работы всех пользователей с единой БД.

Требования к клиентам в этой модели резко уменьшаются, но сервер перегружен, что является основным недостатком данной модели, поскольку выход сервера из строя приведет к выходу из строя системы в целом. Кроме того перегрузка сервера приводит к снижению скорости обработки запросов.

Избавиться от перечисленных недостатков позволит использование трехуровневой модели. Эта модель является расширением двухуровневой модели сервера базы данных и в ней вводится дополнительный промежуточный уровень между клиентом и сервером. Архитектура трехуровневой модели изображена на рисунке 1. Этот промежуточный уровень содержит один или несколько серверов приложений.

В этой модели компоненты приложения делятся между тремя исполнителями.

Клиент обеспечивает логику представления, включая графический пользовательский интерфейс, выполнение локальных приложений. Дополнительно, реализация взаимодействия между клиентом и сервером может включать в себя управление распределенными транзакциями.

Серверы приложений составляют новый промежуточный уровень архитектуры. Они могут исполнять общие функции для клиентов. Серверы приложений поддерживают функции клиентов, как

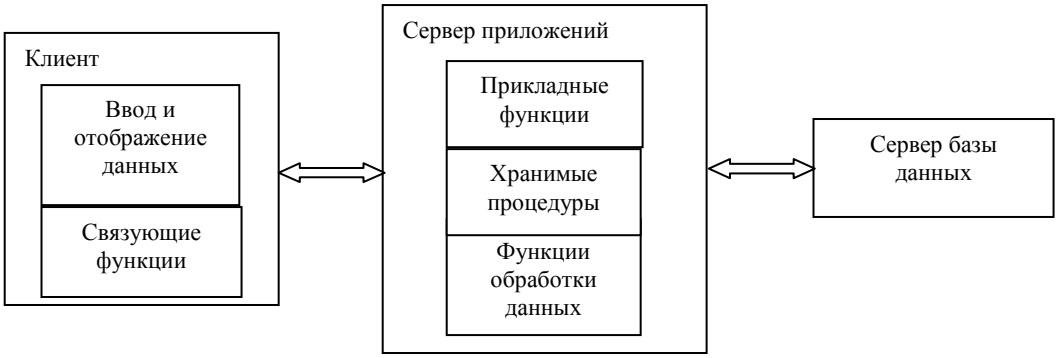

Рисунок 1 — Модель сервера приложений

частей взаимодействующих рабочих групп, поддерживают сетевую доменную операционную среду, хранят и исполняют наиболее общие правила прикладных функций, поддерживают каталоги с данными, обеспечивают обмен сообщениями и поддержку запросов, особенно в распределенных транзакциях. Еще одна функция, выносимая на этот уровень — проверка возможности обработки запросов и выполнения транзакций.

Серверы баз данных в этой модели занимаются исключительно функциями СУБД: обеспечивают функции создания и ведения БД; поддерживают целостность реляционной БД; обеспечивают функции хранилищ данных. Кроме того, на них возлагаются функции создания резервных копий БД и восстановления БД после сбоев, управления выполнением транзакций и поддержки приложений.

Достоинствами модели являются:

— снижение нагрузки на трафик;

— уменьшение нагрузки на сервер базы данных;

— отсутствие необоснованного дублирования кода приложений, за счет переноса общего кода на сервер приложений;

— отсеивание «неуполномоченных» запросов до их поступления на сервер базы данных, что позволяет не рассматривать такие запросы;

— клиентские приложения не зависят от конкретной СУБД, что увеличивает переносимость системы и ее масштабируемость.

Таким образом, проведенный анализ возможной организации взаимодействия клиентов и распределенной базы данных позволяет сделать вывод о целесообразность применения трехуровневой архитектуры. Однако для ее применения в конкретной системе необходимо исследовать следующие характеристики:

— оптимальное количество серверов приложений;

— гарантированную обработку запросов;

— дополнительную задержку на сервере приложений.

Для расчета оптимального количества серверов воспользуемся статистическими данными и формулами, используемыми при расчете в системах массового обслуживания. Известно, что для системы с потоком заявок и несколькими обрабатывающими устройствами существование стационарного режима выражается соотношением: $\Theta\lambda < NB$, где Θ — среднее значение трудоемкости выполнения задания; λ — интенсивность потока заявок; N — количество обрабатывающих устройств; B — быстродействие обрабатывающего устройства. Для нашего примера возьмем оценки интенсивности, трудоемкости и быстродействия равными соответственно: $\lambda = 0{,}5$ с$^{-1}$; $\Theta = 150$ млн.оп.; $B = 50000000$ оп/с. Тогда количество обрабатывающих устройств равно двум. Таким образом, можно установить количество серверов в трехуровневой модели. Для нашего примера достаточно двух серверов, но необходимо еще проверить, будут ли запросы пользователей обработаны в случае возможных отказов серверов. Необходимо также оценить задержку обработки запроса на промежуточном уровне.

Для исследования возможной задержки на сервере приложений и для проверки гарантированной обработки запросов построим модель системы.

Поток запросов от клиентов можно рассматривать как случайный. Естественной моделью для описания случайной последовательности смены состояний некоторого объекта во времени является марковский или полумарковский процесс [2]. Однако наличие взаимосвязей в достаточно сложной системе приводит к необходимости учета взаимодействия элементарных полумарковских процессов и формирования единого случайного процесса, учитывающего не только смену состояний отдельных элементов моделируемой системы, но и весь комплекс взаимодействий между элементами.

Идеальным инструментарием для анализа взаимодействия процессов являются сети Петри [2], однако, являясь асинхронными по определению, модели указанного типа позволяют лишь ответить на вопросы о принципиальной достижимости состояний системы, соответствующих заданным требованиям. Спрогнозировать моменты переключения в указанные состояния с помощью сетей Петри в их классической интерпретации затруднительно. Объединение двух подходов к моделированию процессов в сложных системах порождает сеть Петри-Маркова (СПМ), которая представляет собой структурно-параметрическую модель, заданную множеством $Y = \{П, М\}$, где П — описание структуры двудольного графа, представляющего собой сеть Петри; М — описание параметров, накладываемые на структуру П, и определяющих временные, вероятностные и логические характеристики СПМ.

Для рассматриваемого примера сеть Петри-Маркова является наиболее подходящим инструментом моделирования.

Структура СПМ характеризуется множеством:

$П = \{S,\ Z,\ I_S(Z),\ O_S(Z)\}$, где $S = \{s_{1(s)}, ..., s_{j(s)}, ..., s_{J(s)}\}$ – конечное множество позиций; $Z = \{z_{1(z)}, ..., z_{j(z)}, ..., z_{J(z)}\}$ – конечное множество переходов; $I_S(Z) = \{I_S(z_{1(z)}), ..., I_S(z_{j(z)}), ..., I_S(z_{J(z)})\}$ и $O_S(Z) = \{I_S(z_{1(z)}), ..., I_S(z_{j(z)}), ..., I_S(z_{J(z)})\}$ — соответственно входная и выходная функции переходов; $J(s)$ — общее количество позиций; $J(z)$ — общее количество переходов.

Параметрические аспекты M сети Петри-Маркова описываются следующим множеством $M = \{q, p, f(t), L\}$, где $q = [q_{1(z)}, ..., q_{j(z)}, ..., q_{J(z)}]$ — вектор, определяющий вероятность начала процесса в одном из переходов множества Z; $p = [p_{j(s)j(z)}]$ — матрица с количеством элементов $J(s) \times J(z)$, определяющая априорные вероятности участия $j(s)$-го элемента в одном из возможных «соревнований» $z_{j(z)}$; $f(t) = [f_{j(s)j(z)}(t)]$ — матрица с количеством элементов $J(s) \times J(z)$, определяющая плотности распределения времени t пребывания процесса в состоянии $S_{j(s)}$ при участии в «соревновании» $z_{j(z)}$; $L = [l_{i(z)i(s)}]$ — матрица с количеством элементов $J(z) \times J(s)$, определяющих логические условия окончания взаимодействий, с последующим переключением в состояния множества $O_S(z_{j(z)})$.

Сеть Петри-Маркова, отражающая трехуровневую модель с двумя серверами приложений, изображена на рисунке 2.

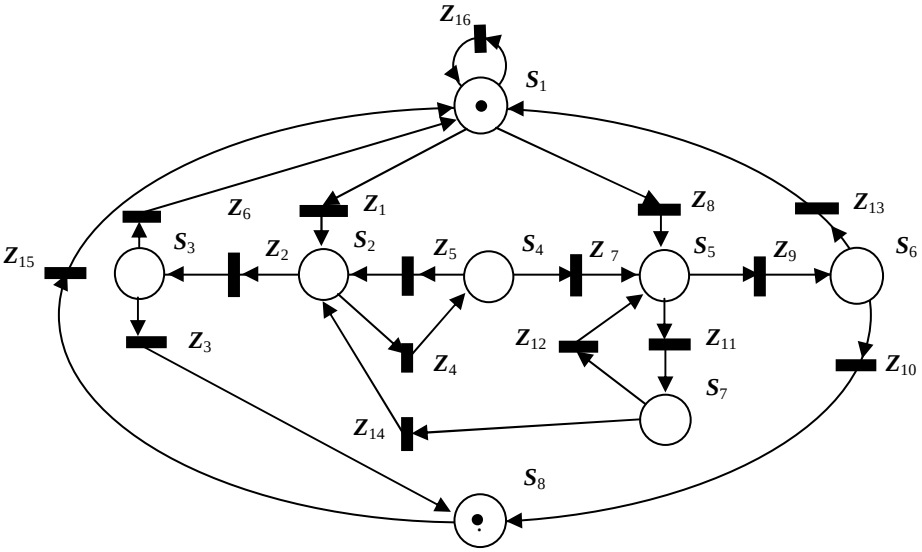

Рисунок 2 — Сеть Петри-Маркова, соответствующая трехуровневой модели сервера приложений с двумя серверами

На рисунке 2 S_1 — позиция, моделирующая процесс поступление запросов от клиентов; S_2, S_5 — позиции, моделирующие работоспособное состояние первого и второго сервера соответственно; позиции S_3, S_6 — моделируют проверку возможности обработки запроса; S_4, S_7 — моделируют неработоспособное состояние сервера; S_8 — моделирует связь с сервером базы данных; z_1, z_8 — переходы, означающий поступление запросов клиентов на один из серверов; z_2, z_9 — переходы, моделирующий идентификацию пользователя; z_3, z_{10} — переходы, моделирующие обращение к серверу базы данных; z_4, z_{11} —переходы, моделирующие сбой серверов, первого и второго, соответственно; z_5, z_{12} —переходы соответствующих северов в работоспособное состояние; z_7, z_{14} — передача запросов на другие сервера; z_6, z_{13} — сообщение клиентам о невозможности обработки запросов; z_{15} — передача обработанных данных клиентам; z_{16} — подготовка к приему новых запросов. Метки в позициях S_1, S_8 обозначают возможные начальные состояния.

Структура СПМ описывается множеством

$\Pi = \{\{S_1, S_2, S_3, S_4, S_5, S_6, S_7, S_8\}, \{z_1, z_2, z_3, z_4, z_5, z_6, z_7, z_8, z_9, z_{10}, z_{11}, z_{12}, z_{13}, z_{14}, z_{15}, z_{16}\}, \{\{S_1\}, \{S_2\}, \{S_3\}, \{S_2\}, \{S_4\}, \{S_3\}, \{S_4\}, \{S_1\}, \{S_5\}, \{S_6\}, \{S_5\}, \{S_7\}, \{S_6\}, \{S_7\}, \{S_8\}, \{S_1\}\}, \{\{S_2\}, \{S_3\}, \{S_8\}, \{S_4\}, \{S_2\}, \{S_1\}, \{S_5\}, \{S_5\}, \{S_6\}, \{S_8\}, \{S_7\}, \{S_5\}, \{S_1\}, \{S_2\}, \{S_1\}, \{S_1\}\}$.

Вектор начала процесса имеет вид (процесс с вероятностью единица начинается в переходе z_1): $q = (1, 0, 0, 0, 0, 0, 0, 0)$.

Матрицы вероятностей и плотностей распределения имеют вид:

$$p = \begin{pmatrix} p_{11} & 0 & 0 & 0 & 0 & 0 & 0 & p_{18} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p_{116} \\ 0 & p_{22} & 0 & p_{24} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & p_{33} & 0 & 0 & p_{36} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & p_{45} & 0 & p_{47} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p_{59} & 0 & p_{511} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p_{610} & 0 & 0 & p_{613} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p_{712} & 0 & p_{714} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p_{815} & 0 \end{pmatrix},$$

$$f(t) = \begin{pmatrix} f_{11}(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & f_{18}(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & f_{116}(t) \\ \xi(t) & f_{22}(t) & \xi(t) & f_{24}(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) \\ \xi(t) & \xi(t) & f_{33}(t) & \xi(t) & \xi(t) & f_{36}(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) \\ \xi(t) & \xi(t) & \xi(t) & \xi(t) & f_{45}(t) & \xi(t) & f_{47}(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) \\ \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & f_{59}(t) & \xi(t) & f_{511}(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) \\ \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & f_{610}(t) & \xi(t) & \xi(t) & f_{613}(t) & \xi(t) & \xi(t) & \xi(t) \\ \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & f_{712}(t) & \xi(t) & f_{714}(t) & \xi(t) & \xi(t) \\ \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & \xi(t) & f_{815}(t) & \xi(t) \end{pmatrix},$$

где $\xi(t) = \lim\limits_{\tau \to \infty} \delta(t - \tau)$, $\delta(t)$ – функция Дирака.

Матрица логических условий имеет вид:

$$L = \begin{pmatrix} 0 & [s_1, z_1] & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & [s_2, z_2] & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & [s_3, z_3] \\ 0 & 0 & 0 & [s_2, z_4] & 0 & 0 & 0 & 0 \\ 0 & [s_4, z_5] & 0 & 0 & 0 & 0 & 0 & 0 \\ [s_5, z_6] & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & [s_4, z_7] & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & [s_1, z_8] & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & [s_5, z_9] & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & [s_6, z_{10}] \\ 0 & 0 & 0 & 0 & 0 & 0 & [s_5, z_{11}] & 0 \\ 0 & 0 & 0 & 0 & [s_7, z_{12}] & 0 & 0 & 0 \\ [s_6, z_{13}] & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & [s_7, z_{14}] & 0 & 0 & 0 & 0 & 0 & 0 \\ [s_8, z_{15}] & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ [s_1, z_{16}] & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Для анализа гарантированной обработки запросов построим граф достижимости сети Петри.

Граф достижимости изображен на рисунке 3.

На основе построенного графа достижимости, можно сделать вывод, что сеть является достижимой, т.е. все маркировки достижимы из начального состояния, следовательно, пользовательские заявки будут обработаны даже при выходе из строя одного из серверов.

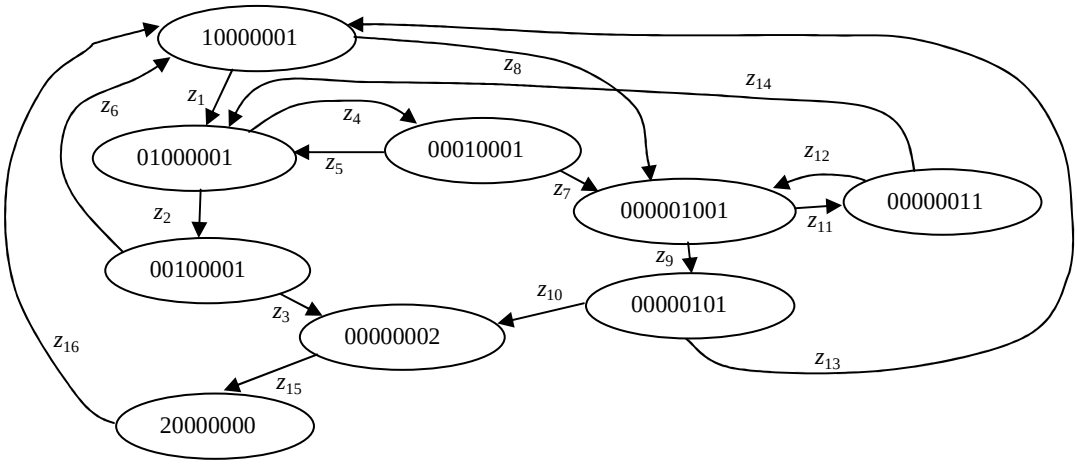

Рисунок 3 — Граф достижимости сети Петри

Таким образом, в работе была проанализирована возможная организация взаимодействия клиентов с распределенной базой данных, обоснован выбор трехуровневой архитектуры. Построена модель распределенной системы с двумя серверами приложений на основе сети Петри-Маркова. С помощью сети Петри была прослежена последовательность выполнения действий при обмене данными между клиентом и сервером и показано, что запросы будут обработаны даже при выходе из строя одного из серверов. Для того чтобы оценить временные характеристики, требуется использовать для анализа построенной модели соответствующие программные средства.

В дальнейшей работе предполагается исследовать поведение построенной модели в среде *AnyLogic*, а также исследовать возможность масштабирования трехуровневой архитектуры.

Библиографический список использованной литературы

1. Дейт К. Дж. Введение в системы баз данных / К. Дж. Дейт. — М.: Вильямс, 2001. — 1072 с.

2. Савин А.Н. Использование сетей Петри-Маркова в тестировании программных комплексов / А.Н. Савин // Известия ТулГУ. Сер. Вычислительная техника: Информационные технологии. — Тула, 2009. — Вып. 3. — С. 146–150.

Поступила в редакцию 10.05.2012 г.

Шаталова Ю.Г. Моделювання процесу обробки запитів до розподіленої бази даних із використанням мережі Петрі-Маркова

Аналізуються моделі організації взаємодії клієнтів і розподіленої бази даних. Обґрунтовується вибір трирівневої моделі. Будується модель обраної архітектури на основі мережі Петрі-Маркова.

Ключові слова: мережа Петрі-Маркова, розподілена база даних, сервер додатків.

Shatalova J.G. Modeling the process of requests treatment to the distributed database with Petri-Markov nets

The models of interaction of clients and distributed database are analyzed. The choice of a three-level model is proved. The model of the chosen architecture based on Petri-Markov nets is presented.

Keywords: Petri-Markov nets, distributed database, application server.