УДК 001.57

ОБЗОР СПОСОБОВ ОРГАНИЗАЦИИ ОЧЕРЕДЕЙ ЗАЯВОК ДЛЯ РЕШЕНИЯ ЗАДАЧ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ ФУНКЦИОНИРОВАНИЯ РАСПРЕДЕЛЕННЫХ СИСТЕМ СБОРА И ОБРАБОТКИ БОЛЬШОГО ОБЪЕМА ДАННЫХ

Р.А. Яковенко, Л.И. Сучкова
Алтайский государственный технический университет им. И.И. Ползунова
г. Барнаул

Статья посвящена изучению подходящих моделей для организации работы с очередями заявок в вопросах имитационного моделирования функционирования распределенных систем сбора и обработки больших объемов данных.

Ключевые слова: система массового обслуживания, имитационное моделирование, big data, очередь сообщений, заявка.

За последние десятки лет использование технологий обработки и анализа больших данных значительно возросло. Такая тенденция связана с тем, что в процессе деятельности различных организаций генерируется большой объем информационных ресурсов. Например, ежедневно социальные сети обрабатывают и хранят терабайты информации (фото- и видеоматериалы, документы, сообщения и т.п.). По прогнозам специалистов в области информационных технологий такая тенденция, как минимум, сохранится и будет стремительно развиваться [1].

В подавляющем большинстве высоконагруженные распределенные системы обработки и хранения данных являются системами массового обслуживания (далее, СМО), в которых в произвольные моменты времени появляются заявки на обслуживание от клиентов, а также присутствует устройство или комплекс устройств для обработки таких заявок [2].

Распределенная система сбора и обработки данных представляет собой клиент-серверную архитектуру, в которой клиенты и сервера это хосты, каждый из которых имеет свой IP-адрес, MAC-адрес, операционную систему, размер оперативной памяти и множество других параметров.

Целью данного исследования является создание имитационной модели взаимодействия информационных процессов в высоконагруженных системах распределенного хранения большего объема данных. Имитационная модель и реализующее ее программное обеспечение, как правило, должны собирать и выводить данные о загруженности каждого сервера распределенной системы. Поскольку процессы, протекающие в такой системе, носят случайный характер, интерес представляет использование принципов теории массового обслуживания при построении имитационной модели взаимодействия информационных процессов [3].

В данной работе в качестве технического решения для работы с очередями, взята за основу система RabbitMQ.

RabbitMQ – это программный комплекс, который предназначен для обеспечения надежного обмена сообщениями между приложениями. Данный комплекс работает на всех основных операционных систем и имеет поддержку огромного количества платформ [4]. RabbitMQ имеет открытый исходный код.

Основная идея данной системы заключается в следующем: сервер RabbitMQ принимает и передает сообщения в виде BLOB-данных.

В рамках использования RabbitMQ используются следующие термины и определения [5]:

1) Производитель (Producing). Программа, которая отправляет сообщения. Обозначается символом "P" (рисунок 1).

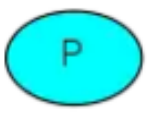

Рисунок 1 – Обозначение производителя в RabbitMQ

2) Очередь (Queue). Стек сообщений, который не связан никакими пределами и может хранить много сообщений - это по существу бесконечный буфер. Многие производители могут отправлять сообщения, которые поступают в одну очередь, многие потребители могут получать данные из одной очереди. Очередь обозначается символом Q и на схемах выглядит, как показано на рисунке 2.

Рисунок 2 – Обозначение очереди Q

3) Потребитель (Consuming). Потребитель - это программа, которая в основном ожидает приема сообщений, обозначается символом "C" (рисунок 3).

Рисунок 3 – Обозначение потребителя

Стоит обратить внимание, что производитель P, потребитель C и сервер RabbitMQ с очередями Q_i могут не функционировать на одном компьютере, в большинстве случаев они распределены по разным хостам.

На рисунке 4 представлен простой пример работы с очередями. В примере, имеется производитель, который посылает одно сообщение и потребитель, который получает сообщение.

Рисунок 4 – Простая схема работы с очередью

Рассмотрим более сложные способы организации очередей в СМО.

Модель «Очередь задач». Основная идея такой модели заключается в выполнении ресурсоемких задач в ситуациях, когда они требуют длительного времени своего выполнения. Понятие задачи будет приравнено к понятию «сообщение». При запуске многих рабочих задач происходит их распределение между обработчиками (потребителями C_i). Одним из преимуществ использования очереди задач является возможность легко распараллеливать выполнение работы. При увеличении сложности задач всегда существует возможность добавить больше обработчиков и таким образом систему легко масштабировать.

Эта модель демонстрирует, как при помощи системы работы с очередями RabbitMQ все заявки могут быть распределены циклически (или равномерно) между более чем одним обрабатывающим устройством, при этом число их неограниченное.

Модель «Обмен сообщениями». Одна рабочая очередь может использоваться для распределения трудоемких задач между несколькими получателями. Полная модель обмена сообщениями в RabbitMQ выглядит, как показано на рисунке 5.

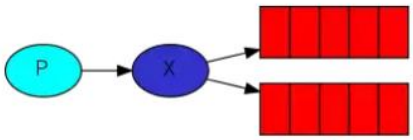

Рисунок 5 – Полная модель обмена сообщениями

Основная идея в модели обмена сообщениями заключается в том, что производитель никогда не посылает сообщения непосредственно в очередь. Вместо этого производитель может отправлять сообщения только на обмен (обозначение «X»). С одной стороны, компонент обмена получает сообщения от производителей, а с другой стороны он направляет их в очередь. Компонент обмена решает следующие вопросы:

1) Должно ли сообщение быть добавлено к определенной очереди?

2) Должно ли сообщение быть добавлено ко многим очередям?

Модель «Публикация/подписка». Обмен по типу разветвления достаточно прост и заключается в том, чтобы передать все полученные сообщения для всех очередей канала.

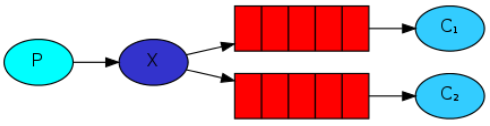

Рисунок 6 – Организация работы с очередями публикация/подписка

Модель можно расширить, добавив возможность подписываться только на подмножества сообщений (рисунок 6). Таким способом можно реализовать фильтрацию сообщений по категориям. Стоит отметить, что обмен типа «ветвление» не дает такой гибкости, потому что он способен только на широ-

ковещательную передачу сообщений (broadcast). В данном случае требуется прямой обмен (direct).

Модель «Маршрутизация сообщений». Алгоритм маршрутизации через прямой обмен сообщений прост: сообщение отправляется в очередь, ключ связывания которой соответствует ключу маршрутизации сообщения (рисунок 7).

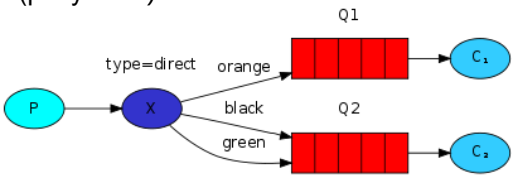

Рисунок 7 – Схема прямого обмена

В таком случае организатор прямого обмена X взаимодействует с двумя связанными с ним очередями. Первая очередь связана с ключом связывания «orange», а второй имеет две привязки: одна с ключом «black», а другая – «green». Сообщение, опубликованное с ключом маршрутизации «orange», будет направлено в очередь Q1. Сообщения с ключом маршрутизации «black» или «green» отправится в очередь Q2. Все остальные сообщения будут отброшены.

Механизм работы с очередями при маршрутизации сообщений всё же имеет недостаток: нет возможности маршрутизации сообщений на основе нескольких критериев. Набор критериев придает большую гибкость поступления сообщений в очереди. Поэтому, чтобы обрабатывать сообщения через анализ нескольких условий, необходимо пользоваться моделью «Тематика» (Topics). Особенность данной модели в том, что сообщения, отправляемые в очередь, не могут иметь произвольный ключ маршрутизации. Как правило, он состоит из списка слов, разделенных точками. Например, «stock.usd.nyse», или «quick.orange.rabbit». Что касается ключа связывания очереди, то он должен быть в том же формате, что и ключ маршрутизации. Логика topics-обмена похожа на логику direct-обмена. Однако есть два важных частных случая:

1) * (Звездочка) может заменить ровно одно слово;

2) # (Хэш) может заменить ноль или более слов.

Схема модели продемонстрирована на рисунке 8.

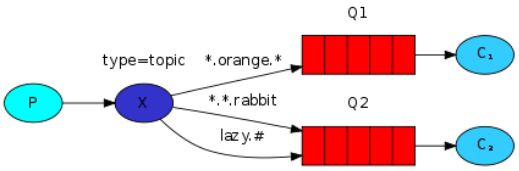

Рисунок 8 – Обмен типа Topics

В этом примере происходит отправка сообщений, которые описывают животных. Сообщения будут отправляться с ключом маршрутизации, который состоит из трех слов (две точки).

Первое слово в ключе маршрутизации будет описывать быстроту, второе цвет и третий разновидность, то есть формат имеет вид "<быстрота>.<цвет>.<вид>". Ключ связывания для очереди Q1 «*.orange.*», для Q2 – «*.*.rabbit» и «lazy.#». Иначе говоря, очередь Q1 принимает всех «оранжевых» животных, а очередь Q2 – всех кроликов и всех ленивых животных. Сообщение с ключом маршрутизации из набора ключей «quick.orange.rabbit» будет доставлено в обе очереди. В то время как сообщение «quick.orange.male.rabbit» (три точки) не будет соответствовать ни одному ключу связывания, поэтому будет потеряно. Хотя с другой стороны, сообщение «lazy.orange.male.rabbit» несмотря на то, что имеет четыре слова, будет соответствовать последнему ключу связывания и будет доставлено во вторую очередь.

Модель удаленного вызова процедур (RPC). Как уже было продемонстрировано, при помощи механизма очереди задач (Work Queues) можно распределять трудоемкие задачи между большим количеством обработчиков. Технология RPC (Remote procedure call) не только инкапсулирует такие же возможности, но и позволяет клиенту ожидать результата, и реализовать соответствующий отклик на него.

Алгоритм RPC заключается в следующем:

1) Запуск клиента создает очередь обратного вызова.

2) Для RPC-запроса клиент отправляет сообщение и специальный идентификатор, который устанавливает уникальное значение для каждого запроса для идентификации ответа.

3) Запрос отправляется в очередь.

4) RPC-обработчик (сервер) ожидает запросы по этой очереди. Когда появится запрос, выполняется требуемая обработка сообщения и отправляется сообщение с результатом обратно клиенту, используя оче-

редь обратного вызова.

5) Программа-клиент ожидает данные по очереди обратного вызова. Когда появится сообщение, происходит проверка специального идентификатора. Если оно соответствует значению из запроса, то программа выполняет запланированный отклик.

На рисунке 9 показана схема алгоритма вышеописанных действий.

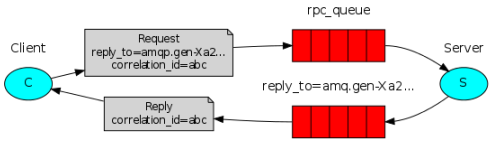

Рисунок 9 – Схема алгоритма

В данной работе выполнен обзор наиболее подходящих способов организации очередей заявок, которые применимы для решения задач имитационного моделирования обработки, хранения и передачи большого объема данных.

СПИСОК ЛИТЕРАТУРЫ

1. Тиндал Сьюзен. Большие данные: все, что вам необходимо знать. [Электронный ресурс] // 2012. – Режим доступа: http://www.pcweek.ru/idea/article/detail.php?ID=141962. Дата обращения 23.01.2016.
2. Гильмутдинов Р.Ф., Кирпичников А.П. Математическая модель замкнутой одноканальной системы массового обслуживания // Вестник Казанского государственного технологического университета – Казань: Изд-во Казан. гос. технол. ун-та, 2011 - № 6 - С. 189.
3. Имитационное моделирование [Электронный ресурс] – 2013. – Режим доступа: http://ru.wikipedia.org/wiki/Имитационное_моделирование.
4. Part 1: RabbitMQ for beginners – What is RabbitMQ? [Электронный ресурс] – 2015. – Режим доступа: https://www.cloudamqp.com/blog/2015-05-18-part1-rabbitmq-for-beginners-what-is-rabbitmq.html
5. RabbitMQ. Introduction [Электронный ресурс] – 2015. – Режим доступа: http://www.rabbitmq.com/tutorials/tutorial-one-python.html.
6. Харламов, А.И. Исследование схем хранения информации в распределенных системах с учетом основных закономерностей доступа к данным [Текст] / А.И.Харламов, Л.И. Сучкова, Е.В. Бочкарёва // Ползуновский вестник: измерение, информатизация, моделирование: проблемы и перспективы технологической разработки и применения. – Барнаул: Изд-во АлтГТУ, 2012. – №3/ 2. – С. 81 – 86.
7. Demchenko Y. Defining the Big Data Architecture Framework (BDAF). Outcome of the Brainstorming Session at the University of Amsterdam. SNE Group. University of Amsterdam, Amsterdam // 2013. – Режим доступа: http://bigdatawg.nist.gov/_uploadfiles/M0055_v1_7606723276.pdf. Дата обращения 03.02.2016.
8. Queueing Systems [Текст] / Ivo Adan and Jacques Resing // Department of Mathematics and Computing Science Eindhoven University of Technology/ The Netherlands March 26, 2015.
9. John D.C. Little and Stephen C. Graves Little's Law [Электронный ресурс] // Massachusetts Institute of Technology / 2002. – Режим доступа: http://web.mit.edu/sgraves/www/papers/Little's%20Law-Published.pdf . Дата обращения: 05.02.2016
10. J. Virtamo 38.3143 Queueing Theory [Электронный ресурс]/ Queueing systems / 2015. – Режим доступа: https://www.netlab.tkk.fi/opetus/s383143/kalvot/E_jonojarj.pdf. Дата обращения: 03.02.2016.
11. Allen, A.O. Probability, statistics, and queueing theory with computer science [Текст] / applications, 2nd ed. Academic Press, Inc., Boston, MA, 1990.

Яковенко Роман Алексеевич – магистрант, тел.: +7-923-714-85-11, e-mail: feurjarx@gmail.com;
Сучкова Лариса Иннокентьевна – доктор технических наук, профессор, e-mail: lara8370@yandex.ru; кафедра информатики, вычислительной техники и информационной безопасности.