УДК 004.75

ОРГАНИЗАЦИЯ ОТКАЗОУСТОЙЧИВОЙ РЕПЛИКАЦИИ ПОТОКОВЫХ ДАННЫХ ПОДСИСТЕМЫ ТРАНСПОРТА СООБЩЕНИЙ В АРХИТЕКТУРЕ CORBA ПРИ ПОМОЩИ ТЕХНОЛОГИИ ZEROMQ

Ф.А. Козлов

Обсуждаются особенности построения отказоустойчивой распределенной системы, главным функционалом которой является репликация потоковых данных. Представлены концепции реализации данной системы с использованием технологии ZeroMQ. Рассматриваются основные преимущества технологии ZeroMQ перед другими аналогами в создании данного вида распределенных систем.

Ключевые слова: распределенная система, репликация, кластер, брокер, ZeroMQ.

Введение

CORBA (Common Object Request Broker Architecture) – технологический стандарт написания распределенных приложений. Технология CORBA создана для поддержки разработки и развертывания сложных объектно-ориентированных прикладных систем. В настоящее время используется множество систем CORBA, и их разработка и стандартизация развиваются по мере того, как растет число установленных систем. Примером таких систем является системы электронного документооборота (СЭД).

Технологический стандарт CORBA позволяет отдельным сервисам, написанным на разных языках программирования, взаимодействовать друг с другом. Взаимодействие реализуется с помощью распределенных объектов [1]. В СЭД, основанных на CORBA, используются подсистемы транспорта сообщений (ПТС), являющиеся надстройкой над реализацией CORBA и предоставляющие интерфейс для обмена данными между серверными и клиентскими модулями системы. В компаниях, состоящих из нескольких географически удаленных офисов, возникает необходимость создания единой СЭД. Единая СЭД – это распределенная система, позволяющая производить выборочный обмен информацией между локальными СЭД.

Распределенная система – это информационная система, состоящая из независимых элементов, находящихся на удалении друг от друга. Одно из главных требований к распределенной системе – ее единство в представлении пользователя. Применение технологии CORBA в распределенной СЭД затрудняет ее дальнейшую поддержку, так как возникает необходимость синхронизации версий каждого сервера СЭД при изменении интерфейса одного из распределенных объектов [2]. Решением данной задачи является создание распределенной системы ПТС СЭД. Обмен данных в ПТС основан на передаче сообщений в событиях. Такое решение позволяет производить передачу сообщений с локального сервера СЭД в распределенную систему, создавая необходимые фильтры внешних сообщений.

Главной функцией распределенной системы ПТС СЭД является передача сообщения, полученного одной ПТС от локального модуля, всем ПТС, входящим в распределенную систему. Таким образом, производится репликация данных, поступающих на ПТС, в распределенной системе. Основой репликации является передача данных от подчиненного узла системы к главному узлу, который передает полученное сообщение на все подчиненные узлы. Недостатком такой схемы является ее низкая надежность. При выходе из строя главного узла распределенная система полностью теряет свою работоспособность. Повышение надежности распределенной системы возможно путем создания на ее основе отказоустойчивого кластера.

Целью настоящей работы является разработка метода построения отказоустойчивого кластера, производящего выборочную репликацию данных ПТС СЭД.

Анализ технологий передачи данных

Для обеспечения связи локальной ПТС СЭД с распределенной системой необходимо создать специальный модуль – брокер, предоставляющий функционал для взаимодействия с узлами кластера. Существуют готовые решения, позволяющие производить обмен данных и репликацию в распределенных системах. Наиболее распространенные технологии в данной области – RabbitMQ и ActiveMQ. Обе технологии используют протокол AMPQ [3] и предоставляют готовый брокер для обмена сообщениями между клиентами и сервером. К недостаткам готовых брокеров при организации распределенных СЭД относятся избыточный функционал, большой объем занимаемого дискового пространства и увеличение пакета передаваемых данных путем наложения дополнительных системных протоколов. Эти недостатки влияют на скорость передачи данных в распределенной системе. Высокая скорость передачи потоковых данных является одним из определяющих требований к распределенной СЭД.

Среди технологий обмена сообщениями выделяется низкоуровневая библиотека ZeroMQ, которая предоставляет интерфейс для обмена сообщениями через системные сокеты. Проведенные измерения времени передачи пакетов данных при использовании технологий ZeroMQ, RabbitMQ и ActiveMQ (табл. 1) показали, что технология ZeroMQ по скоростным характеристикам опережает свои аналоги. Измерения производились на ОС Ubuntu 12.04 с 6 потоками Pentium i7 и 2 ГБ оперативной памяти. Тестируемые технологии производили обмен пакетами размером 10 Б по связи «запрос–ответ». Относитель-

ная погрешность полученных результатов не превышает 1%. Средние скорости передачи пакетов составили: ZeroMQ – 27270 Б/с, RabbitMQ – 7000 Б/с, AtciveMQ – 230 Б/с. Высокая скорость передачи сообщений в сокетах ZeroMQ позволяет использовать данную технологию при создании брокера высокопроизводительной распределенной системы.

Количество пакетов	Время передачи пакетов, мс			
	RabbitMQ (Python)	ActiveMQ (Python)	ZeroMQ (Python)	ZeroMQ (C++)
100	–	4376	–	–
1000	1414	42137,66	352,33	379
10000	14694,66	430036,66	3332	3592,33
100000	140387,33	–	32465,33	36238,33

Таблица 1. Время передачи пакетов данных между клиентом и сервером

Анализ подсистемы транспорта сообщений

ПТС управляет каналами, по которым передаются сообщения. Модуль, получивший доступ к каналу, имеет возможность обмена сообщениями с подписанными на этот канал модулями СЭД. Сообщения в канале могут располагаться как в памяти самого ПТС, так и в базе данных. При сбое ПТС сохраненные в базе сообщения восстанавливаются при перезапуске системы.

Каналы ПТС разделяются на стабильные и нестабильные – по способу хранения сообщений в памяти, в также на глобальные и локальные – по области видимости канала. Для создания распределенной системы необходимо организовать потоковую передачу всех сообщений, поступающих в глобальные каналы ПТС, на локальный брокер, являющийся узлом кластера.

Архитектура распределенной системы

Так как в распределенной системе СЭД отсутствует условие обеспечения строгой идентичности всех серверов, в кластере используется асинхронная репликация данных. Организация отношений между узлами распределенной системы производится по схеме «главный–подчиненный» [4]. Предусматривается наличие в системе одного главного сервера (ГС), с которым соединены все остальные подчиненные сервера (ПС). Такой подход позволяет избежать лишних связей между узлами системы.

При использовании схемы «главный–подчиненный» необходимо обеспечить возможность автоматического возобновления репликации при сбое ГС в кратчайшие сроки. Для решения данной задачи создается кластер по схеме «Ротация резерва» (Rotating Standby) [5]. Эта схема предполагает наличие резервного сервера (РС), который при сбое ГС занимает его место. Место РС в такой ситуации занимает один из активных ПС. Главным преимуществом данной схемы является автоматическое возобновление работы кластера после сбоя и низкий промежуток простоя системы [6].

Брокер потоковых данных

Для реализации описанной выше архитектуры необходимо создать модуль брокера, предоставляющий определенный функционал в зависимости от ролей узла в кластере (рис. 1).

Рис. 1. Диаграмма вариантов использования системы

ПС обеспечивает асинхронный обмен сообщениями между локальным ПТС и ГС кластера. При этом ПС проходит регистрацию на ГС и обрабатывает системные сообщения кластера. Задачей ГС является прием и рассылка сообщений с удаленных ПТС. ГС производит выбор и наблюдение за РС, а при его сбое производит поиск нового РС среди ПС, а также производит регистрацию всех серверов кластера для дальнейшей рассылки системных сообщений. РС наблюдает за состоянием ГС. При сбое ГС РС переходит в состояние ГС и делает рассылку всем ПС системных сообщений о переподключении. Наблюдение за состояниями серверов производится с помощью технологии Heartbeat [7].

Для реализации асинхронного взаимодействия как на уровне функционала, предоставляемого брокером (одновременная работа ГС и ПС на одном узле), так и на уровне функционала сервера (асинхронный прием и отправка сообщений на ПС) необходимо создать систему потоков. Первый уровень потоков запускает функционал, предоставляемый ГС, ПС или РС. Второй уровень потоков обеспечивает асинхронную работу внутри каждого сервера. Для реализации потоков брокера в исследовании была использована технология Boost Thread [8].

Каждый поток брокера оперирует определенным набором сокетов для обмена данными. Технология ZeroMQ предоставляет расширения системных сокетов для решения определенных задач. Основными типами связи сокетов в ZeroMQ являются:

– «Публикация–подписка» (PUB–SUB);
– «Запрос–ответ» (REQ–REP);
– «Потоковая передача» (PUSH–PULL);
– «Уникальная пара» (PAIR).

«Публикация–подписка» является связью, где на один публикующий сервер подписывается множество серверов. Это односторонняя связь. «Публикация–подписка» используется в брокере для рассылки сообщений для ПТС и системных сообщений кластера на ГС, а также для наблюдения за состояниями серверов.

Связь «Запрос–ответ» представляет собой классическую синхронную модель взаимодействия клиента и сервера. Клиент производит запрос необходимой информации на сервер и получает ответ. В брокере такая связь необходима для выбора РС и регистрации серверов в кластере.

Связи «Потоковая передача» и «Уникальная пара» являются односторонними и предназначены для потоковой передачи данных без блокировки сокета. «Потоковая передача» используется в брокере для сбора ГС сообщений с удаленных ПТС. «Уникальная пара» отличается от «Потоковой передачи» тем, что в первой присутствуют только один передающий и один принимающий сокет [9]. Связь используется в брокере для обмена сообщениями между ПТС и брокером.

Структура ПС, РС и ГС состоит из главного и прикладных потоков. Потоки производят управление и обработку результатов на определенных сокетах. Каждый сокет брокера имеет один тип и выполняет одну функцию.

Система устроена так, что брокер перенаправляет полученные с кластера сообщения на вход локального ПТС (рис. 2). Таким образом, в случае задержек при обработке полученных сообщений на ПТС новые сообщения с кластера будут вставать в очередь обработки на ПТС, а не на локальном брокере. Независимо от работоспособности локального ПТС, брокер будет способен в полной мере поддерживать работоспособность всего кластера.

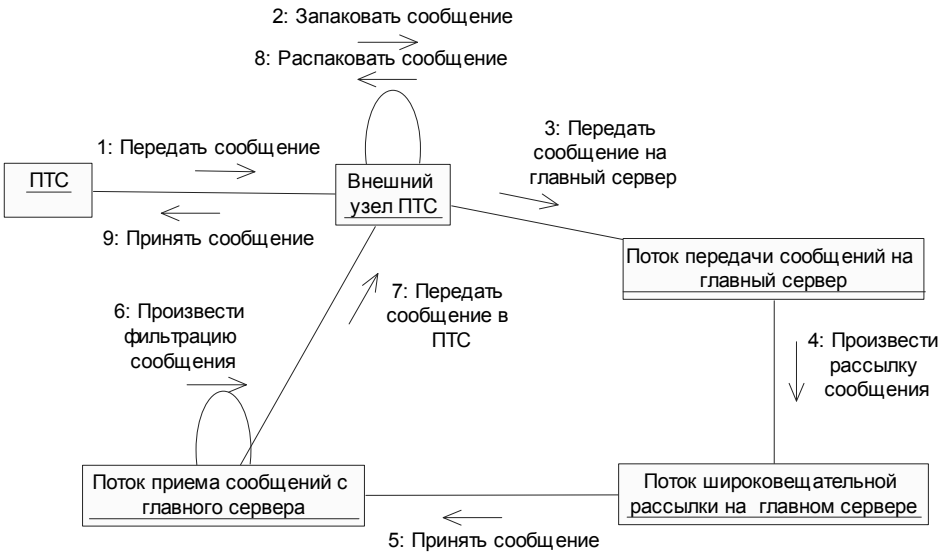

Рис. 2. Диаграмма коммуникации потоков системы при репликации данных в кластере

Способность технологии ZeroMQ производить обмен данных через сокеты на внутрипроцессном уровне позволяет управлять потоками в брокере с минимальной затратой ресурсов на время передачи сообщения.

Заключение

Разработанный брокер позволяет объединять удаленные подсистемы транспорта сообщений в отказоустойчивый кластер и производить их репликацию. Были проведены измерения времени репликации данных одной подсистемы транспорта сообщений на все подсистемы транспорта сообщений распределенной системы (табл. 2). Из полученных результатов следует, что брокер обладает высокой скоростью передачи данных. При увеличении узлов системы происходит дополнительная нагрузка на ГС, что снижает производительность брокера.

Недостатками брокера является отсутствие гарантии доставки сообщения и низкий уровень безопасности канала. Разработанная архитектура предполагает использование брокера в закрытых и защищенных сетях, в которых скорость передачи сообщения преобладает над надежностью его передачи. Такими системами являются подсистемы транспорта сообщений систем электронного документооборота, системы мгновенных сообщений и системы записи логов. Данные системы обладают высокой частотой передачи данных и требуют от распределенной системы высокой пропускной способности.

Количество пакетов	Время передачи пакетов для определенного количества узлов в кластере, мс			
	2	3	4	5
10000	121	171	219	301
100000	1091	1570	2091	2668
1000000	10926	14804	20481	26903

Таблица 2. Время репликации данных в кластере

Разработанная архитектура позволяет создавать брокеры с расширенным функционалом. В брокере возможна организация передачи данных по защищенному соединению [10] и реализация восстановления и повторной передачи недоставленных сообщений на ГС. Для оптимизации передачи сообщений в брокере возможна фильтрация определенных каналов подсистемы транспорта сообщений. Упомянутые расширения предусматривают дополнительную нагрузку и вводятся в зависимости от требований к системе.

Литература

1. Таненбаум Э., ван Стеен М. Распределенные системы. Принципы и парадигмы. – СПб: Питер, 2003. – 877 с.
2. Zinky J., Bakken D., Schantz R. Architectural Support for Quality of Service for CORBA Objects // Theory and Practice of Object Systems. – 1997. – V. 3. – P. 55–73.
3. Vinoski S. Advanced Message Queuing Protocol // IEEE Internet Computing. – 2006. – V. 10. – №. 6 – P. 87–89.
4. Hercule K., Eugene M., Paulin B., Joel L. Study of the Master-Slave replication in a distributed database // IJCSI International Journal of Computer Science Issues. – 2001. – V. 8. – P. 319–326.
5. Маркарян К. Построение кластера высокой надежности под управлением GNU/Linux [Электронный ресурс]. – Режим доступа: http://www.ibm.com/developerworks/ru/library/l-Cluster_Linux_1/, свободный. Яз. рус. (дата обращения 20.01.2013).
6. Шубинский И.Б. Структурное резервирование в информационных системах. Предельные оценки // Надежность – 2012. – № 1. – С. 118–125.
7. Fei-fei Li, Xiang-zhan Yu, Gang Wu. Design and Implementation of High Availability Distributed System Based on Multi-level Heartbeat Protocol // CASE '09 Proceedings of the 2009 IITA International Conference on Control, Automation and Systems Engineering. – 2009. – P. 83–87.
8. Williams A., Botet Escriba V. The Boost C++ Libraries BoostBook Documentation Subset. Chapter 30. Thread 4.0.0 [Электронный ресурс]. – Режим доступа: http://www.boost.org/doc/libs/1_53_0/doc/html/thread.html, свободный. Яз. англ. (дата обращения 15.01.2013).
9. Hintjens P., ZeroMQ: The Guide [Электронный ресурс]. – Режим доступа: http://zguide.zeromq.org/, свободный. Яз. англ. (дата обращения 15.01.2013).
10. Tariq M., Koldehofe B., Altaweel A., Rothermel K. Providing basic security mechanisms in broker-less publish/subscribe systems // Conference: Distributed Event-Based Systems (DEBS). – NY, USA, 2010. – P. 38–49.

Козлов Федор Алексеевич – ОАО «Промышленные информационные системы», программист, аспирант, kozlovfedor@gmail.com