В фреймворке Kilim акторы представлены типом Task, являются легковесными потоками и взаимодействуют между собой при помощи типа Mailbox. Обработка байт-кода классов выполняется специальным процессом, называемым «weaver». На этапе исполнения планировщик управляет пулом потоков ядра ограниченного размера, позволяющим выполнять множество легковесных потоков с минимальными затратами на запуск и переключение контекста [2].

Основное отличие ActorFoundry от Kilim – предоставляемый интерфейс программирования приложений (API), а также дополнение кода приложения сгенерированным кодом на этапе сборки, после чего процесс «weaver» обрабатывает байт-код классов.

Actors Guild работает на этапе исполнения программы, динамически изменяя байт-код при помощи библиотеки ASM. Данное решение сравнительно легко интегрировать в проект, по сравнению с Kilim или ActorFoundry. Основные задачи, которые позволяет решить библиотека ASM, - это анализ, изменение существующих class-файлов и генерация новых class-файлов, представленных в формате байт-кодов JVM.

В фреймворке Akka акторы являются динамическими активными объектами [4], поэтому создаются при помощи специального метода actorOf(). Когда актор начинает функционировать, Akka помещает его в реестр, так что он становится доступным, пока не будет остановлен. В Akka реализовано три типа отправки сообщений – fire-and-forget («отправил и забыл», асинхронная отправка сообщения без ожидания результата), request-reply («вопрос-ответ», отправка сообщения и ожидание ответа, синхронный режим), request-reply-with-future («отправить и получить ответ в будущем», отправка сообщения и получение ответа дальше по коду с помощью специального объекта).

Функциональность акторной модели можно реализовать с использованием стандартной модели потоков Java, основанной на классе Thread. Дальнейшее исследование предполагает проверку трудоёмкости и сравнительной эффективности акторного кода при исполнении нагрузочных тестов по сравнению с описанными акторными библиотеками. В качестве технологии преобразования API потоков в акторную модель будет рассмотрена технология, применяемая в системе Templet [5] для программирования в модели акторов на языке C++.

Литература

1. Ага Г., Мейсон И., Смит С., Талкотт К. Основания для вычислений акторов / Journal of Functional Programming, 1993.

2. Вторая волна разработки Java-приложений: Представляем Kilim [Электронный ресурс]. – Режим доступа: https://www.ibm.com/developerworks/ru/library/j-javadev2-7/

3. Srinivasan S., Mycroft A. Kilim: Isolation-Typed Actors for Java (A Million Actors, Safe Zero-Copy Communication) , University of Cambridge Computer Laboratory [Электронный ресурс]. – Режим доступа: https://www.cl.cam.ac.uk/research/srg/opera/publications/papers/kilim_ecoop08.pdf

4. Subramanian V. Programming concurrency on the JVM. Mastering synchronization, STM, Actors – Pragmatic Programmers, LLC, USA 2011.

5. Востокин, С.В. Препроцессор языка Templet: инструмент программирования в терминах модели «процесс-сообщение» / Вестн. Сам. гос. техн. ун-та. Сер. Физ.-мат. Науки, 3(36) (2014), 169–182.

А.В. Трошин, В.А. Лапшин

КОЛЬЦЕВОЕ И ПАРАЛЛЕЛЬНОЕ РЕЗЕРВИРОВАНИЕ В СЕТЯХ ETHERNET

(Поволжский государственный университет телекоммуникаций
и информатики)

Одним из основных способов повышения надежности сетей Ethernet является включение дополнительных линий между коммутаторами для создания избыточности на втором уровне модели OSI. В случае отказа одной из линий трафик передается по резервному маршруту. Однако такое решение может приводить к появлению петель, когда кадры проходят один и тот же маршрут не доходя до узла назначения. Для устранения петель в сетях Ethernet применяется протокол STP (Spanning Tree Protocol) и его различные модификации. При работе данного протокола каждый коммутатор рассылает специальные BPDU (Bridge Protocol Data Unit) кадры через все свои активные интерфейсы второго уровня для определения корневого коммутатора. Корневой коммутатор служит для построения "дерева" маршрутов до всех узлов второго уровня, все избыточные линии не входящие в это "дерево" блокируются. В случае отказа какой-либо линии, входящей в дерево маршрутов, происходит переконфигурация "дерева" с подключением альтернативного маршрута и разблокирования резервных линий. Основными недостатками STP являются: достаточно большое время сходимости - до нескольких десятков секунд, резко возрастающее при большом числе коммутаторов; низкая эффективность использования ресурсов сети из-за того, что пропускная способность резервных линий не используется. Для повышения скорости сходимости и эффективности STP в настоящее время в сетях Ethernet широко используют различные модификации протокола STP, такие как RSTP, PVST+, MSTP. Данные модификации хотя и позволяют повысить время сходимости STP до нескольких секунд, но даже такое время восстановления связи может не удовлетворять требованиям современных гигабитных сетей Ethernet, задействованных для обработки трафика реального времени в центрах обработки данных.

Одним из альтернативных способов реализации резервирования в сетях Ethernet является применение кольцевой топологии для построения сети с использованием специально разработанного для данной топологии протокола MRP (Media Redundancy Protocol). Данный протокол описан в стандарте IEC 62439-2. В кольцевой топологии, состоящей из не более чем 50 коммутаторов, назначается ведущий - MRM (Media Redundancy Manager), остальные коммута-

торы считаются ведомыми - MRC (Media Redundancy Client). MRM коммутатор проверяет целостность кольца, посылая специальные кадры в одном направлении и принимая их с другого. В нормальном режиме - "замкнутое кольцо" (Ring-Closed) все кадры с противоположного направления отбрасываются MRM. При выходе из строя линии или коммутатора в кольце, происходит переключение в так режим "разомкнутого кольца" (Ring-Open), когда MRM начинает отправлять кадры в оба направления. Преимуществом кольцевого резервирования является стабильно низкое время переключения при отказе - до 200 мс. Однако для многих приложений может быть неприемлемым как кольцевая топология сети, так и даже настолько малое время восстановления.

Для параллельного резервирования между любыми двумя узлами сети создается два маршрута, идентичные кадры посылаются одновременно по каждому из маршрутов. В случае отказа одного маршрута кадры продолжают поступать через параллельный, вследствие этого отказ практически никак не сказывается на работе конечных приложений. Таким образом конечный узел в нормальном режиме всегда должен получать два одинаковых кадра, получение одного кадра сигнализирует об отказе одного из маршрутов. Для реализации параллельного резервирования разработан протокол PRP (Parallel Redundancy Protocol), стандартизированный как IEC 62439-3. Протокол PRP для работы фактически требует наличия двух параллельных Ethernet-сетей, поэтому для реализации общей сети, как правило, используется топология двойной звезды. В каждой отдельной сети может применяться дополнительное резервирование на основе протокола STP. Необходимо тщательное проектировании сети с резервированием PRP, так как любая связь между параллельными сетями может привести к мгновенному отказу обоих сетей вследствие образования петлевых маршрутов. К преимуществам PRP необходимо отнести отсутствие какой-либо специально поддержки протокола на промежуточных коммутаторах, что облегчает реализацию сетей с PRP. Поддержка PRP необходима только на конечных узлах, которые называют DANP (Double Attached Node for PRP), они должны иметь два сетевых подключения к каждой отдельной сети и проводить проверку дублирования поступающих кадров. Преимуществами протокола PRP являются6 нулевое время восстановления работоспособности при отказах, отсутствие каких-либо специальных требований к топологии. Главный недостаток состоит в необходимости удвоения ресурсов сети, что значительно увеличивает стоимость реализации.

Для кольцевой топологии существует адаптация протокола PRP - HSR (High -availability Seamless Redundancy). Как и в PRP, в HSR наличествует два параллельных маршрута, но реализованных в кольцевой топологии. По каждому кольцевому маршруту параллельно передаются идентичные кадры. Каждый HSR-коммутатор отбирает из кадры предназначенные для подключенных к нему узлов. Для отбрасывания дублированных кадров используется специальный заголовок, что требует специальной поддержки функций HSR на коммутаторах. К преимуществам HSR можно отнести более высокую скорость обработки кадров в сравнении с PRP. Недостатки HSR аналогичны недостаткам PRP, к которым добавляется необходимость поддержки функций HSR на коммутаторах.

На основе проведенного анализа можно сделать выводы о том, что пока существующие технологии традиционного, кольцевого и параллельного резервирования в сетях Ethernet далеки от идеала и не обладают достаточной гибкостью в сочетании с низкими затратами на реализацию. Необходимы дальнейшие исследования по их совершенствованию и поиску альтернативных решений, возможно совмещающих резервирование на втором и третьем уровнях модели OSI.

Литература

1. Стандарт IEEE 802.1D [Электронный ресурс]. Режим доступа: http://standards.ieee.org/getieee802/download/802.1D-2004.pdf

2. Стандарт IEC 62439-2:2010 / Industrial communication networks - High availability automation networks - Part 2: Media Redundancy Protocol (MRP) [Электронный ресурс]. Режим доступа: https://webstore.iec.ch/publication/7017

3. Стандарт IEC 62439-3:2010 / Industrial communication networks - High availability automation networks - Part 3: Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR) [Электронный ресурс]. Режим доступа: https://webstore.iec.ch/publication/20490

М.В. Терёхин, Е.И. Чигарина

КЛАССИФИКАЦИЯ И АНАЛИЗ ИСПОЛЬЗОВАНИЯ ОКОННЫХ ФУНКЦИЙ В СИСТЕМАХ РЕЛЯЦИОННЫХ БАЗ ДАННЫХ

(Самарский национальный исследовательский университет
имени академика С.П. Королёва)

В системах баз данных одной из основных задач является задача сокращения времени выполнения запросов при работе с данными с использованием минимального объема памяти, то есть задача эффективного выполнения запросов.

В настоящее время наибольшее распространение получили реляционные базы данных, для работы с которыми используется язык SQL (Structured Query Language – язык структурированных запросов) [1]. В языке SQL постоянно совершенствуются команды запросов для более эффективного их выполнения. В частности, в стандарте языка SQL для реализации аналитических задач стали использоваться многочисленные функции, оперирующие окнами.

В настоящее время не существует четкой классификации оконных функций в зависимости от их назначения и использования, а также не проводился анализ эффективности выполнения запросов с использованием оконных функций по сравнению с традиционными запросами без них. В реляционных базах данных традиционно используются скалярные и агрегатные функции. Агрегат-