Р10-2009-85

Э. Г. Никонов, А. Б. Флорко*

ПОВЫШЕНИЕ ОТКАЗОУСТОЙЧИВОСТИ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ В УСЛОВИЯХ СТРЕССОВОЙ НАГРУЗКИ

*Университет «Дубна»

Никонов Э. Г., Флорко А. Б. Р10-2009-85
Повышение отказоустойчивости систем массового
обслуживания в условиях стрессовой нагрузки

Различные информационные системы потоковой обработки информации (системы биржевых торгов, web-серверы, SCADA-системы) сталкиваются с непредвиденными ситуациями в работе, когда информационный поток, требующий анализа и ответной реакции, многократно возрастает, превосходя возможности системы в обработке. Подобные ситуации стрессовых нагрузок нередко требуют вмешательства диспетчера или администратора, поэтому критически важным оказывается время наступления первого отказа системы в обслуживании.

В статье рассматриваются распространенная архитектура систем массового обслуживания на основе обмена сообщениями и существующие подходы к распределению процессорного времени для ее функционирования.

Предлагается новый подход, ориентированный на отсрочивание первого отказа.

Работа выполнена в Лаборатории информационных технологий ОИЯИ.

Сообщение Объединенного института ядерных исследований. Дубна, 2009

Nikonov E. G., Florko A. B. P10-2009-85
Fault Tolerance Improvement for Queuing Systems
under Stress Load

Various kinds of queuing information systems (exchange auctions systems, web servers, SCADA) are faced to unpredictable situations during operation, when information flow that requires being analyzed and processed rises extremely. Such stress load situations often require human (dispatcher's or administrator's) intervention that is the reason why the time of the first denial of service is extremely important.

Common queuing systems architecture is described. Existing approaches to computing resource management are considered. A new late-first-denial-of-service resource management approach is proposed.

The investigation has been performed at the Laboratory of Information Technologies, JINR.

Communication of the Joint Institute for Nuclear Research. Dubna, 2009

ВВЕДЕНИЕ

Самый широкий спектр современных информационных систем массового обслуживания предполагает поэтапную обработку поступающих в систему запросов. В ответ на поступивший запрос (будь то принятие клиентского пакета из сети, ввод данных пользователем или появление в системе периодического события) такие системы выполняют определенную последовательность действий по обработке данных сообщения, включающую, в общем случае, несколько этапов. К примеру, web-сервер в ответ на запрос статической HTML-страницы последовательно разбирает адрес страницы, выполняет ее поиск в локальном кеше, затем (если необходимо) обращается к жесткому диску, читает данные, подготавливает ответный пакет и отправляет его клиенту.

Один из существующих архитектурных подходов, применяемых для создания масштабируемых многоэтапных событийно-управляемых информационных систем массового обслуживания, рассматривает систему как сеть компонент, обменивающихся сообщениями (рис. 1).

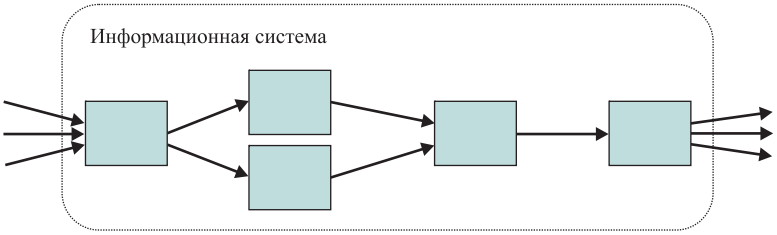

Рис. 1. Системы многоэтапной обработки информации

С каждой компонентой системы ассоциирована очередь входящих сообщений фиксированного размера и пул системных потоков, выделенных для ее обработки. В общем случае сообщение проходит маршрут из нескольких компонент (каждая из которых выполняет определенную работу над данными сообщения), прежде чем сообщение удаляется из системы.

Среди информационных систем, использующих рассмотренный архитектурный подход, можно перечислить следующие:

- системы потоковой обработки данных*;
- web-серверы**;
- системы управления базами данных***;
- системы**** реального времени*****.

Поскольку перечисленные системы применяются во многих важных для человеческой жизнедеятельности областях (военной, аэрокосмической, химической, транспортной), особое внимание при разработке следует уделять отказоустойчивости. Под *отказоустойчивостью* понимается способность вычислительной системы продолжать действия, заданные программой, после возникновения неисправностей [8]. Требование отказоустойчивости в последние годы стало особенно актуальным как в связи с экспоненциальным ростом числа террористических атак по всему миру, так и с ежегодным увеличением числа регистрируемых природных и техногенных катастроф. В то время как неспособность системы справиться с нагрузкой в случае интернет-сервера может закончиться судебными исками покупателей******, в случае со SCADA — техногенными авариями и человеческим жертвами.

Среди всевозможных причин, ведущих к отказу системы (неисправность оборудования, исчерпание доступных вычислительных ресурсов), в статье рассматриваются сценарии стрессовой нагрузки, в которых число принимаемых системой сообщений превосходит способность системы к обработке. Входящие сообщения начинают выстраиваться в очередь, и отказ системы возникает при попытке поставить сообщение в очередь, достигшую предельного размера у одной из компонент.

Перегрузка может носить временный, непредсказуемый характер. Так, например, во время событий 11 сентября 2001 г. новостной сайт CNN испытывал 20-кратную перегрузку по сравнению с запланированным пиком на протяжении двух с половиной часов. SCADA-системы после ввода в эксплуатацию могут столкнуться с неисправностью оборудования, посылающего

*Например, процессоры обработки потоков (stream processing engines) Aurora [5], STREAM [6], TelegraphCQ [7].

**Например, Flash [1].

***Например, SQL Server 2000 [4].

****Например, брокер запросов для приложений реального времени Real Time CORBA ORB [3]

*****Системы обычно считаются системами реального времени, если время их реакции имеет порядок миллисекунд; диалоговыми считаются системы с временем реакции порядка нескольких секунд, а в системах пакетной обработки время реакции измеряется часами или днями [10].

******Что имело место против компании E*Trade [9].

на обработку непредусмотренное число команд, а web-серверы должны быть готовы к атакам, направленным на отказ в обслуживании (denial of service).

В рассмотренных сценариях стрессовой нагрузки под отказоустойчивостью будем понимать в соответствии с определением, приведенным выше, способность системы продолжать работу после возникновения неисправностей максимально продолжительное время, достаточное для того, чтобы выполнить критически важное управляющее воздействие, необходимое для нейтрализации возникшей нештатной ситуации.

АРХИТЕКТУРА МНОГОЭТАПНЫХ СОБЫТИЙНО-УПРАВЛЯЕМЫХ СИСТЕМ

В одном из существующих архитектурных подходов, применяемых для создания масштабируемых информационных систем массового обслуживания, таких как системы SCADA (Supervisory Control and Data Acquisition), web-серверы* (и другие системы потоковой обработки информации), приложение рассматривается как совокупность подсистем, обменивающихся между собой сообщениями (рис. 2).

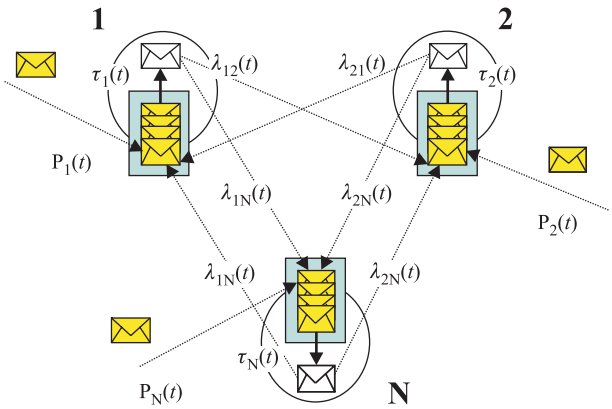

Рис. 2. Многоэтапная событийно-управляемая система (МСУС) на основе обмена сообщениями

МСУС на основе обмена сообщениями состоит из N подсистем, обменивающихся между собой сообщениями с частотой $\lambda_{ij}(t)$, зависящей от времени где i — индекс подсистемы-отправителя, j — индекс подсистемы-получателя

*В качестве примера можно привести web-сервер Flash [1].

$i,\ j = 1,\ N$. Возникновение сообщения в i-й подсистеме обязано какому-то внешнему событию, например, поступлению данных от измерительного оборудования. Вероятность возникновения сообщения в i-й подсистеме подчиняется закону распределения $P_i(t)$. Сообщение содержит все важные для обработки события данные. Сообщение проходит маршрут из нескольких подсистем, каждая из которых выполняет над данными сообщения работу за время обслуживания $\tau_i(t)$, зависящее как от характера работы, выполняемой на подсистеме, так и от занятости всех подсистем, разделяющих общие вычислительные ресурсы (процессор, оперативную память). Когда i-я подсистема не успевает обрабатывать поступающие сообщения, сообщения выстраиваются в очередь с предельным размером q_i.

Под отказом системы в данной работе подразумевается попытка поставить сообщение на одном из этапов его обработки в очередь, достигшую предельного размера.

Число системных потоков l_i, ассоциированных с очередью сообщений i-й подсистемы, может быть разным. Так в случае одного системного потока, выделенного на обработку очереди, сообщения будут обрабатываться последовательно, в то время как большее число рабочих потоков позволит обрабатывать сообщения параллельно. Общее число системных потоков не должно превышать фиксированной администратором величины T, связанной с ограничениями операционной системы:

$$T = \sum_{i=1}^{N} l_i. \tag{1}$$

Если определить число системных потоков l_i, выделенных на обработку сообщений каждой подсистемы, можно оптимизировать распределение вычислительных ресурсов рабочей станции (в первую очередь процессора) для повышения производительности отдельных маршрутов за счет других.

ОБЗОР СУЩЕСТВУЮЩИХ ПОДХОДОВ К УПРАВЛЕНИЮ

Существуют различные подходы к управлению числом потоков в МСУС.

Классическим решением является *фиксированный пул потоков*, где число потоков, отданных каждой из подсистем, определяется администратором эвристически и остается неизменным все время работы системы: $l_i = \mathrm{const}$. Подход хорошо зарекомендовал себя при прогнозируемых маршрутах работы системы и известном времени обслуживания τ_i каждой из подсистем. Однако, если характер поведения системы изменяется (например, изменяется характер появления внешних событий $\mathrm{P}_i(t)$, и, как следствие вероятность возникновения маршрутов в системе), выбранное однажды число системных потоков может оказаться неоптимальным.

Для преодоления указанного недостатка классического решения в работе [2] был предложен адаптивный подход, основанный на наблюдениях системы через равные интервалы времени. По прошествии каждого интервала принимается решение о числе системных потоков, выделенных подсистеме для минимизации времени обслуживания каждого сообщения подсистемой. Для поиска локального минимума времени обслуживания подсистемы от числа ассоциированных с очередью сообщений системных потоков применяются градиентные методы. В дополнение используется эвристическое правило, уменьшающее число системных потоков подсистемы в случае их бездействия дольше заданного администратором периода времени.

Предложенный подход [2] позволял системам адаптироваться к изменяющимся условиям функционирования, однако принимаемые эвристическим правилом и градиентными методами решения о числе потоков нередко вступали в противоречия. Так, применявшиеся градиентные методы не обладали точной оценкой прироста/падения производительности подсистемы в зависимости от принятых ранее решений о числе потоков, потому что свой вклад в изменение производительности подсистемы вносили абсолютно все решения для всех подсистем (подсистемы разделяют общие вычислительные ресурсы — один процессор и одну оперативную память).

ОТКАЗОУСТОЙЧИВОСТЬ МНОГОЭТАПНЫХ СОБЫТИЙНО-УПРАВЛЯЕМЫХ СИСТЕМ

Рассмотренные существующие решения: фиксированный пул потоков и адаптивный подход [2] — были призваны достичь максимальной производительности системы, однако для создания систем в жизненно важных для человека областях не менее актуален подход, ориентированный на повышение отказоустойчивости.

Под отказоустойчивостью подразумевается способность системы отложить время первого отказа по причине переполнения очереди в сценариях пиковых нагрузок, характерных для SCADA-систем, когда число принимаемых системой сообщений превышает способности системы к обработке. Пиковые нагрузки могут возникать по причине неисправности оборудования или непредвиденного поведения объекта наблюдения, требующего незамедлительного вмешательства диспетчера.

ПРЕДЛАГАЕМЫЙ ПОДХОД К УПРАВЛЕНИЮ, НАПРАВЛЕННЫЙ НА ПОВЫШЕНИЕ ОТКАЗОУСТОЙЧИВОСТИ

Предлагается адаптивный подход с двумя оригинальными контроллерами. Данный подход основан на наблюдениях системы в равные интервалы времени аналогично подходу, предложенному в работе [2].

Первый контроллер принимает решение о числе системных подходов из следующих соображений.

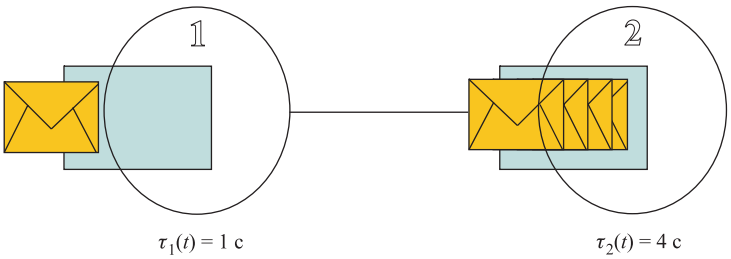

Рис. 3. Время обслуживания двух подсистем отличается в 4 раза

Предположим, система состоит из двух подсистем, где время обслуживания первой $\tau_1(t) = 1$ с в 4 раза меньше времени обслуживания второй $\tau_2(t) = 4$ с, а размер пула потоков $T = 5$, как показано на рис. 3.

Рассмотрим сценарий, в котором первая подсистема принимает сообщения с равными интервалами, передавая их на обработку второй подсистеме.

В случае, когда подсистемам будет выделено по одному системному потоку ($l_1 = l_2 = 1$), первая подсистема будет успевать обрабатывать четыре сообщения и передавать их второй за время, необходимое второй, чтобы успеть обработать лишь одно. Очередь сообщений второй подсистемы начнет увеличиваться, что приведет к появлению отказа в системе.

Если второй подсистеме выделить $l_2 = 4$ системных потока, оставив первой $l_1 = 1$ пропорционально отношению времен обслуживания $\tau_1(t)$ к $\tau_2(t)$ исходя из общего числа потоков $T = 5$:

$$l_1 = T \cdot \tau_1(t)/(\tau_2(t) + \tau_1(t)),$$
$$l_2 = T \cdot \tau_2(t)/(\tau_2(t) + \tau_1(t)),$$

то производительность обеих подсистем окажется равной — вторая подсистема будет успевать обрабатывать весь поток входящих сообщений от первой подсистемы. Теперь, когда число принимаемых первой подсистемой сообщений превысит возможности системы к обработке, очереди начнут увеличиваться на обеих подсистемах одновременно. Сценарий переполнения с одновременным ростом двух очередей сообщений более предпочтительным, чем сценарий роста лишь одной из очередей, поскольку первый отказ системы откладывается за счет удержания в оперативной памяти большего числа сообщений, ожидающих обработки, чем в первом сценарии ($l_1 + l_2 > l_2$).

Обобщая рассуждения на N подсистем, мы получаем правило, согласно которому число потоков l_i, выделенных i-й подсистеме, определяется из отношения суммарного времени

$$l_i = T \frac{\tau_i n_i}{\sum\limits_{j=1}^{N} \tau_j n_j}$$

обработки сообщений i-й подсистемы ($\tau_i \cdot n_i$) к суммарному времени обработки всех сообщений сети как

$$l_i = T \frac{\tau_i n_i}{\sum\limits_{j=1}^{N} \tau_j n_j}.$$

Второй предложенный механизм управления (контроллер обратного давления) состоит в приостановке обработки сообщений на подсистеме-отправителе, что позволяет подсистеме-получателю обработать очередь сообщений, приближающуюся к переполнению. В случае нескольких подсистем отправителей длительность приостановки каждой из них определяется пропорционально вкладу (λ_{ij}) подсистемы-отправителя в переполнение очереди получателя q_j согласно штрафной функции $f(\lambda_{ij}, q_j)$. Вид штрафной функции f выбирается эвристически, в общем случае величина штрафа оказывается тем больше, чем ближе очередь подсистемы получателя к переполнению.

СРАВНИТЕЛЬНЫЙ АНАЛИЗ

Сравнительный анализ существующих подходов и предложенного подхода был выполнен на примере типового приложения потоковой обработки реального времени (времен обработки сообщений, времен отклика?). В качестве такого приложения был выбран гипотетический сервер сбора и обработки информации, как на рис. 4. В обязанности сервера сбора и обработки информации входит принятие сообщений от измерительного оборудования, обработка поступивших данных и сохранение результатов обработки в базе данных.

Для минимизации погрешности в серии тестов реально осуществляемая работа была заменена на эквивалентные по времени циклы, использующие ресурсы процессора, исключающие обращение к внешним системам. Подобный подход к проведению тестирования позволил минимизировать фактор случайности, что позволило более точно исследовать особенности подходов к управлению ресурсами. Проведенные тесты отражают особенности функционирования систем потоковой обработки информации, использующих встроенную (in-memory) базу данных.

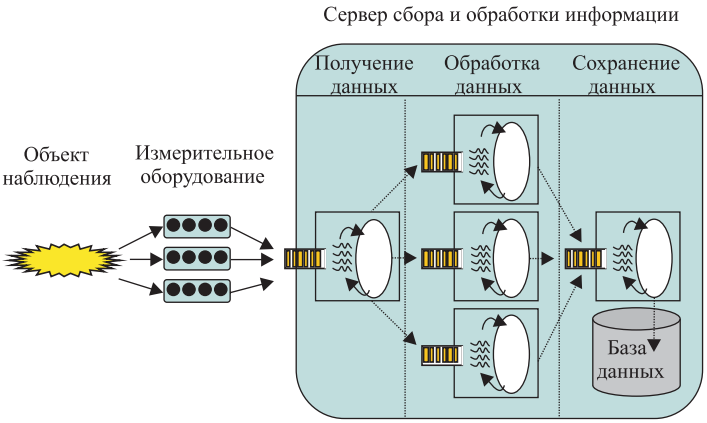

Рис. 4. Архитектура гипотетического сервера сбора и обработки информации

Во всех предлагаемых далее тестах исследовались сценарии перегрузки системы. Наиболее отчетливо различие представленных подходов проявилось в сценариях *кратковременной* перегрузки. Так, до тех пор пока система в состоянии справляться с числом поступающих в нее сообщений, все подходы ведут себя одинаково хорошо. Когда система находится в состоянии длительной перегрузки, все подходы ведут себя одинаково плохо потому, что число сообщений, которое система может обработать в секунду, не зависит от выбранного метода управления ресурсами. Ни один подход не может увеличить производительность процессора или оптимизировать по производительности однажды созданный разработчиком код. Различие подходов проявляется в те временные интервалы, когда в систему поступает число пакетов, превышающее способности системы к обработке, но переполнение очередей не носит перманентного характера.

С точки зрения многоэтапной событийно-управляемой архитектуры в выбранной для теста системе рис. 4 присутствует три слоя:

- слой, отвечающий за принятие клиентских пакетов. Здесь создаются сообщения, содержащие полученные от измерительного оборудования данные;
- слой, отвечающий за преобразование и обработку данных. Здесь производится преобразование и обработка сообщений;
- слой, отвечающий за взаимодействие с базой данных. Используется встраиваемая (in-memory) СУБД.

Операция получения пакетов от измерительного оборудования занимает ничтожно малое время. Данные пакета упаковываются в сообщение, которое поступает на обработку в подсистемы слоя обработки данных. В этом слое

каждая подсистема отвечает за обработку данных одного-единственного типа пакета. В зависимости от размера данных пакета и его структуры обработка на этом уровне занимает 1–3 мс, обращение к базе данных и выполнение хранимой процедуры на следующем слое выполняется за 50 мс в отсутствие других задач.

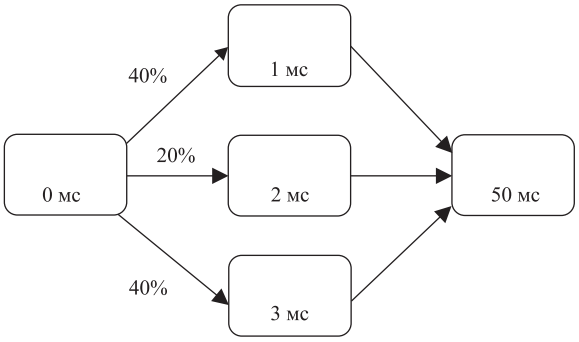

Рис. 5. Сценарий функционирования сервера сбора и обработки информации

В последующих тестах для рассмотренной системы будем считать, что измерительное оборудование присылает пакеты трех типов, на обработку каждого из которых затрачивается 1, 2 и 3 мс. Пакеты различных типов приходят с вероятностями $0{,}4$, $0{,}2$ и $0{,}4$, как показано на рис. 5. В тестах размеры очередей подсистем q_i выставлены в 200 сообщений. В такой конфигурации однопроцессорная система способна обработать около 19 сообщений в секунду: $1000/(50 + (1 \cdot 0{,}4 + 2 \cdot 0{,}2 + 3 \cdot 0{,}4))$.

В первом тесте создавалась кратковременная пиковая нагрузка, затем выдерживалась небольшая пауза, и сервер входил в режим длительной перегрузки, где в каждую секунду сервер принимал на 5–6 пакетов больше, чем был в состоянии обработать (рис. 6).

В течение двух минут сервер принял 3125 пакетов. Сообщения начинали выстраиваться в очереди, ожидая обработки, что приводило в дальнейшем к регулярным отказам в обслуживании.

По результатам теста на рис. 7 наименьшее число потерянных сообщений (площадь под графиками) было достигнуто с использованием предлагаемого подхода (451, что на 5 % лучше классического решения с фиксированным пулом потоков).

Адаптивный подход [2] показал практически эквивалентные результаты с классическим фиксированным пулом потоков.

Отличительной чертой предлагаемого подхода является «шероховатость» графика отказов. Это объясняется работой контроллера обратного давления,

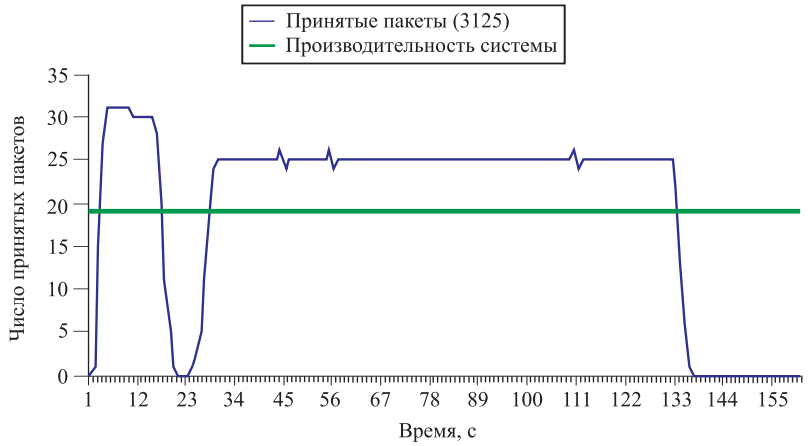

Рис. 6. Сценарий теста

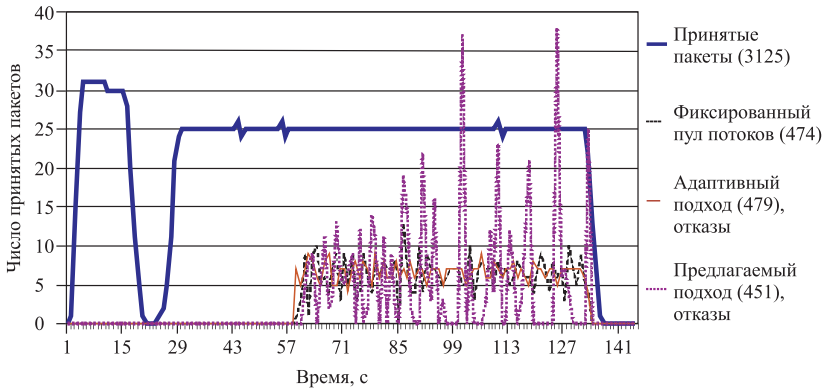

Рис. 7. Необработанные сообщения с применением различных стратегий управления ресурсами в тесте 1

приостанавливавшего работу подсистем преобразования при наполнении очереди подсистемы, отвечающей за обращение к СУБД. В результате отказ на подсистеме третьего слоя откладывался, но создавались предпосылки для преждевременного отказа на приостановленных подсистемах, так как теперь уже их очереди начинали ускоренно наполняться. С одной стороны, искусственная приостановка позволяла равномерно распределить сообщения между очередями, что откладывало по времени наступление первого отказа, с другой стороны — отказ на более раннем этапе обработки (в случае переполнения очереди подсистемы первого и второго слоя) позволял высвободить

вычислительные возможности для обработки поступающих сообщений, что уменьшало общее число отказов.

Рассмотренный тест дает представление о том, почему контроллер обратного давления позволяет системам, построенным с применением предлагаемого подхода, уменьшить число отказов. В рассматриваемом тесте большую часть времени система проводила в обработке сообщений, извлекаемых из одной-единственной очереди, и наиболее удачный подход оказывался тот, что мог «увеличить» ее размер за счет временного наполнения очередей подсистем — источников сообщений.

Однако рассмотренная конфигурация не позволяла изучить возможности управления числом потоков на очередях сообщений.

Для демонстрации преимуществ управления числом потоков на подсистеме против классического фиксированного пула было решено увеличить время обработки сообщений на подсистемах слоя обработки в 10 раз. Теперь времена обработки сообщений оказывались более равномерно распределенными между подсистемами, и эффект «единственной очереди» предыдущего теста нивелировался.

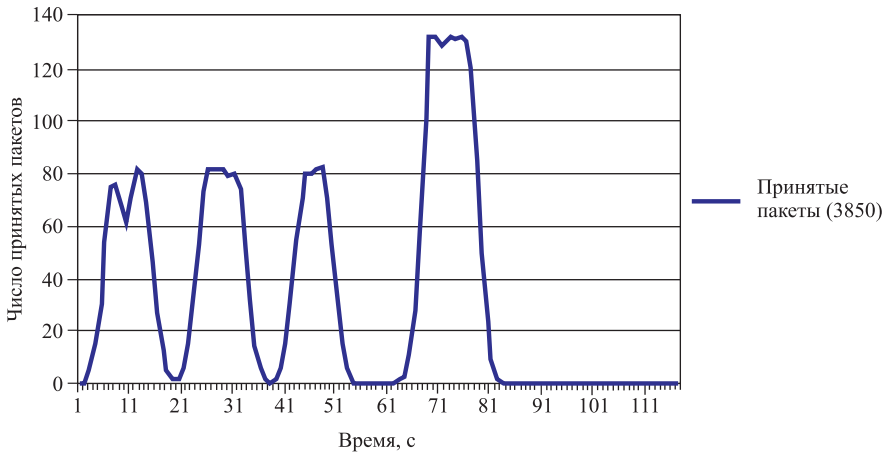

Рис. 8. Тест контроллера пула потоков

Во *втором тесте* на систему было оказано четыре 15-секундных стрессовых потока пакетов от измерительного оборудования (рис. 8), в каждый из которых система принимала как минимум в четыре раза больше пакетов, чем была способна обработать. За полторы минуты в систему пришло 3850 сообщений, что в два раза интенсивнее первого теста.

Адаптивный подход [2], отраженный на рис. 9 продемонстрировал значительно лучшие результаты в сравнении с фиксированным пулом потоков,

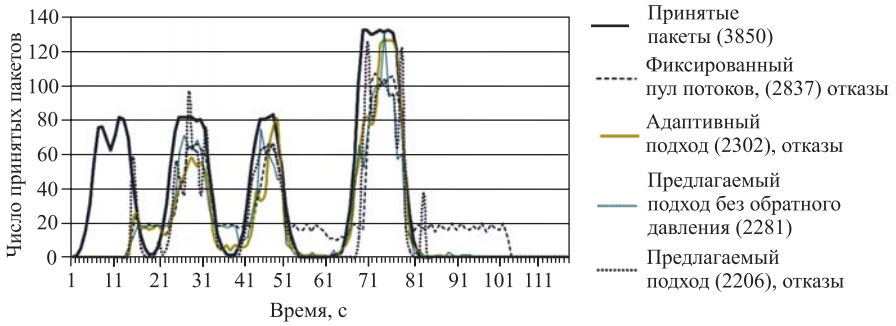

Рис. 9. Обобщенные результаты тестирования контроллера пула потоков

потеряв на 20 % меньше сообщений. Объяснение можно найти в графике потерь фиксированных пулов потоков в периоды спадов нагрузки (55–65 и 80–110 с). При фиксированном числе потоков «легковесные» подсистемы (подсистемы первого и второго слоя с малым временем обслуживания) обладали большими вычислительными ресурсами по сравнению с тем, что позволил им выделить адаптивный подход [2]. В результате легковесные подсистемы поставляли большее число сообщений, чем могла обработать следующая на маршруте «тяжеловесная» подсистема, что приводило к дальнейшему переполнению очереди тяжеловесной системы и отказам даже в отсутствие поступления новых сообщений в систему. При управлении числом потоков удавалось уменьшить потери за счет равномерного заполнения очередей всех подсистем. Предложенный в данной работе контроллер пула потоков в этом тесте оказался равноценен в качестве управления всей совокупности контроллеров адаптивного подхода [2] (2281 и 2302 потерянных сообщения (рис. 9)). Но именно контроллер обратного давления позволил вывести предлагаемый подход вперед по результатам всех тестов, обеспечив на 5 % меньше потерь при погрешности измерений в 1 %.

ЗАКЛЮЧЕНИЕ

Предложенный подход может эффективно применяться в МСУС на основе обмена сообщениями, вероятность сценариев работы которых заранее не известна. Данный подход приводит к значительному повышению отказоустойчивости в сценариях пиковых нагрузок. Предложенный подход может применяться в различных системах массового обслуживания, таких как web-серверы, SCADA-системы, биржевые системы, математические приложения с параллельными вычислениями.

ЛИТЕРАТУРА

1. *Pai V. S., Druschel P., Zwaenepoel W.* An Efficient and Portable WEB Server // Proceedings of the 1999 USENIX Annual Technical Conference, June 1999.
2. *Welsh M. D.* An Architecture for Highly Concurrent, Well-Conditioned Internet Services // Phd Thesis, Graduate Division of the University of California at Berkeley, 2002.
3. *Schmidt D., Vinoski S.* Object Interconnections: Real-Time CORBA, Part 3: Applications and Priorities // C / C++ Users Journal C++ Experts Forum, January 2002.
4. *Rankins R., Bertucci P., Jensen P.* Microsoft® SQL Server 2000 Unleashed, Second Edition, Sams, 2002.
5. *Carney D. et al.* Monitoring Streams: A New Class of Data Management Applications // Proceedings of the 28th International Conference on Very Large Data Bases (VLDB'02), Hong Kong, China, 2002.
6. *Arasu A.* STREAM: The Stanford Stream Data Manager // ACM SIGMOD Conference, June 2003.
7. *Chandrasekaran S. et al.* TelegraphCQ: Continuous Dataflow Processing for an Uncertain World // Proc. of the lst CIDR Conference, Asilomar, CA, 2003.
8. *Богданов А. В. и др.* Архитектуры и топологии многопроцессорных вычислительных систем. Интернет-университет информационных технологий — ИНТУИТ. ру.; 2004.
9. Bloomberg News. E*Trade Hit by Class-Action Suit, CNet News.com, February 9, 1999.
10. Толковый словарь по вычислительным системам / Под ред. В. Илленгуорта и др. Пер. с англ. М.: Машиностроение, 1989.

Получено 4 июня 2009 г.

Редактор *М. И. Зарубина*

Подписано в печать 21.07.2009.
Формат 60 × 90/16. Бумага офсетная. Печать офсетная.
Усл. печ. л. 1,18. Уч.-изд. л. 1,43. Тираж 290 экз. Заказ № 56667.

Издательский отдел Объединенного института ядерных исследований
141980, г. Дубна, Московская обл., ул. Жолио-Кюри, 6.
E-mail: publish@jinr.ru
www.jinr.ru/publish/