

Флорко

ФЛОРКО  
Андрей Борисович

**Повышение отказоустойчивости многоэтапных событийных  
информационных систем массового обслуживания**

Специальность 05.13.01

Системный анализ, управление и обработка информации  
(отрасль: информационные, телекоммуникационные и инновационные  
технологии)

Автореферат диссертации на соискание учёной степени  
кандидата технических наук

Дубна, 2010 г.

Работа выполнена в Международном университете природы, общества и человека «Дубна», в Институте системного анализа и управления на кафедре системного анализа и управления.

Научный руководитель:

доктор физико-математических наук

Э.Г. Никонов

Официальные оппоненты:

доктор технических наук, профессор

А. С. Минзов

кандидат физико-математических наук

М.А. Назаренко

Ведущая организация:

Московский государственный университет приборостроения и информатики  
(МГУПИ)

Защита диссертации состоится 2 июля 2010 года в 15-30 в аудитории 1-300 на заседании диссертационного совета Д800.017.02 при Международном университете природы, общества и человека «Дубна», по адресу: Московская обл., г. Дубна, ул. Университетская, д. 19.

С диссертацией можно ознакомиться в библиотеке Международного университета природы, общества и человека «Дубна».

Автореферат разослан 1.06 2010г.

Ученый секретарь диссертационного совета  
кандидат физико-математических наук



Н.А.Токарева

## Актуальность темы

Различные области человеческой деятельности требуют привлечения высокопроизводительных, отказоустойчивых и масштабируемых информационных систем (ИС) массового обслуживания. Спектр подобных систем самый широкий, от SCADA (Supervisory Control And Data Acquisition – диспетчерское управление и сбор данных) систем в жизненно-важных для человека областях (авиация, энергетика, транспорт) до web-сервисов и биржевых систем потоковой обработки информации.

Критически важным требованием для функционирования таких систем массового обслуживания как SCADA является максимально возможное время наработки на отказ (величина временного промежутка до первого отказа в обслуживании поступившего в систему запроса). Большое время работы системы до первого отказа позволяет оператору принимать более взвешенные управленческие решения при анализе возникшей в системе массового обслуживания ситуации (нередко аварийного характера).

ИС массового обслуживания также нередко внедряются в расчёте на возрастающую интенсивность потока заявок (например, рост числа посетителей с ростом популярности интернет ресурса) или с расчётом на многократное превышение штатной интенсивности заявок (биржевая паника, аварийная ситуация). Поэтому масштабируемость, способность системы линейно увеличивать производительность с увеличением аппаратных возможностей нередко выступает ещё одним важным требованием при разработке.

Одной из распространенных программных архитектур, используемых для построения масштабируемых ИС массового обслуживания, в перечисленных выше областях применения является архитектура, в которой ИС проектируется в виде событийной сети компонентов, взаимодействующих между собой при помощи обмена сообщениями. Как правило, сообщение возникает в системе благодаря внешнему событию (например, поступившей заявке), содержит необходимые данные для обработки, а сама обработка выполняется последовательно в несколько этапов компонентами системы.

Перераспределяя системные потоки между компонентами системы и назначая компонентам временные интервалы приостановки обработки сообщений, удаётся уменьшить среднее время обработки сообщений на некоторых компонентах за счёт

увеличения среднего времени обработки на других компонентах сети, когда компоненты разделяют общие вычислительные ресурсы.

Анализ существующих механизмов распределения вычислительных ресурсов, применяемых при построении подобных многоэтапных событийных систем (МСС) массового обслуживания, показал, что существующие механизмы (механизм с фиксированным числом системных потоков<sup>1</sup> у каждой из компонентов системы и адаптивный механизм [9]) не позволяют существенно увеличить время наработки системы на отказ.

Изложенные обстоятельства обуславливают актуальность разработки механизма распределения вычислительных ресурсов в МСС, ориентированного на увеличение времени первого отказа.

Целью диссертационной работы является повышение отказоустойчивости многоэтапных событийных информационных систем массового обслуживания за счёт нового механизма распределения вычислительных ресурсов, способного увеличить временной промежуток до первого отказа в обслуживании поступившего в систему запроса.

Достижение цели диссертационной работы осуществляется решением следующих задач.

- Исследование предметной области, связанной с проектированием, разработкой и использованием отказоустойчивых масштабируемых ИС массовой обработки информации.
- Создание математической модели МСС массового обслуживания на основе обмена сообщениями в условиях разделяемых компонентами системы вычислительных ресурсов.
- Разработка механизма распределения вычислительных ресурсов МСС, позволяющего повысить время наработки на отказ МСС в условиях пиковой нагрузки.
- Разработка ИС, основанной на предложенном в диссертации механизме, и сравнение её времени наработки на отказ с результатами, полученными для аналогичной ИС, основанной на существующих механизмах.

---

<sup>1</sup> Под системными потоками подразумеваются потоки управления операционной системы (например, Windows), при помощи которых реализуется многозадачность, способность системы к параллельному исполнению программного кода.

## **Предмет и методы исследования**

Предметом исследования являются механизмы распределения вычислительных ресурсов МСС и используемые в них алгоритмы. Для решения поставленных исследовательских задач использовались методы системного анализа, методы и модели теории управления, а также модели теории очередей.

## **Научная новизна**

- Предложена математическая модель многоэтапной событийной системы, основанной на обмене сообщениями, в которой компоненты системы разделяют общие вычислительные ресурсы.
- Предложен новый механизм распределения вычислительных ресурсов МСС, ориентированный на увеличение времени наработки системы на отказ в условиях пиковых нагрузок.
- Впервые разработано специализированное программное обеспечение в виде библиотеки классов на языке программирования Java, которое может быть использовано для создания отказоустойчивых МСС в условиях пиковых нагрузок.

## **Практическая значимость**

Применение разработанного в диссертации механизма позволяет создавать МСС с большим временем наработки на отказ по сравнению с использованием существующих механизмов распределения вычислительных ресурсов, что особенно актуально для SCADA систем.

На основе разработанного в диссертации механизма создана и зарегистрирована в Роспатенте библиотека классов, которая представляет самостоятельный практический интерес с точки зрения использования при конструировании МСС потоковой обработки информации, поскольку позволяет создавать устойчивые к пиковым нагрузкам МСС.

## **Апробация результатов работы**

Основные результаты диссертации были представлены в виде докладов на XII и XIII конференциях студентов, аспирантов и молодых специалистов (г. Дубна 2005, 2006), IX и X конференции молодых учёных и специалистов ОИЯИ (г. Дубна 2005, 2006)

По результатам выполненных исследований опубликовано 7 печатных работ, получено свидетельство о государственной регистрации программы для ЭВМ в Роспатенте.

Построенная с использованием разработанного автором в диссертационной работе механизма отказоустойчивая масштабируемая ИС массового обслуживания внедрена в Лаборатории Сетевых Технологий (г. Дубна).

#### **Публикации и личный вклад**

Диссертация основана на теоретических, методических и проектных исследованиях и разработках, выполненных автором. По теме диссертации опубликовано 7 печатных работ, из них 2 статьи в журналах из перечня ведущих рецензируемых научных журналов и изданий из перечня ВАК Министерства образования и науки РФ. Также получено свидетельство о регистрации программы для ЭВМ в Роспатенте.

В работах, выполненных в соавторстве, соискатель внёс определяющий вклад в разработку представленного в диссертации механизма распределения вычислительных ресурсов МСС потоковой обработки информации для повышения отказоустойчивости в условиях пиковых нагрузок. Автором также были разработаны все алгоритмы и программы, представленные в диссертации.

#### **Объём и структура диссертации**

Диссертация состоит из введения, 4 глав, заключения, списка литературы, приложения и содержит 52 рисунка, 1 таблицу. Общий объём работы – 136 страниц. Основное содержание работы изложено на 111 страницах текста. Список литературы содержит 110 наименований.

#### **Положения, выносимые на защиту**

1. Математическая модель МСС позволяет описывать поведение информационной системы, исполняемой на одной рабочей станции с общими для всех компонентов вычислительными ресурсами и основанной на обмене сообщениями.
2. Новый механизм распределения вычислительных ресурсов МСС, основанный на перераспределении числа системных потоков и назначении времени приостановки обработки входящих сообщений на компонентах, повышает отказоустойчивость МСС в сценариях пиковых нагрузок.

3. Специализированное программное обеспечение в виде библиотеки классов для создания отказоустойчивых МСС обработки информации на основе авторской модели и авторского механизма распределения вычислительных ресурсов позволяет разрабатывать МСС с большим, по сравнению с существующими механизмами, временем наработки на отказ.

### **Содержание работы**

**Во введении** приведены результаты анализа предметной области, определяется проблемное поле будущего исследования и обосновывается актуальность выбранного автором направления исследования, сформулирована его основная цель. Сформулированы задачи, решение которых приводит к достижению цели. Кратко излагается содержание разделов диссертации.

**В первой главе** приводится обзор различных существующих программных архитектур ИС массового обслуживания, таких как классическая архитектура, основанная на потоковом параллелизме с различными вариациями, например, отложенное создание потоков [11], а также архитектура, основанная на использовании фиксированного пула системных потоков. Как обобщение рассматривается распространённая современная программная архитектура, используемая для построения ИС массового обслуживания SEDA [10], предполагающая организацию ИС в виде сети компонентов на основе обмена сообщениями [5, 6]. Сообщения, как правило, появляются в системе в ответ на внешние события (поступающие заявки) и проходят обработку в несколько этапов.

Для подобных МСС рассматриваются механизмы распределения вычислительных ресурсов: механизм с фиксированным числом системных потоков и адаптивный механизм [9].

В ходе проведённого анализа было выявлено, что существующие механизмы не ориентированы на увеличение времени наработки системы на отказ. Как следствие, существующие механизмы не могут эффективно применяться в ИС системах массового обслуживания, где критическим требованием выступает большое время первого отказа, например, по причине переполнения очереди входящих сообщений на одной из компонентов сети в сценариях пиковых нагрузок. Такие сценарии характерны для SCADA систем при возникновении аварийных ситуаций.

Во второй главе приведено описание предложенного нового механизма распределения вычислительных ресурсов МСС массового обслуживания, в условиях, когда все компоненты разделяют общие вычислительные ресурсы [1 – 4]. Данный механизм применим к МСС различной топологии (рис. 1). Главным назначением предложенного автором механизма является увеличение времени наработки на отказ по сравнению с традиционными механизмами в условиях пиковой нагрузки.

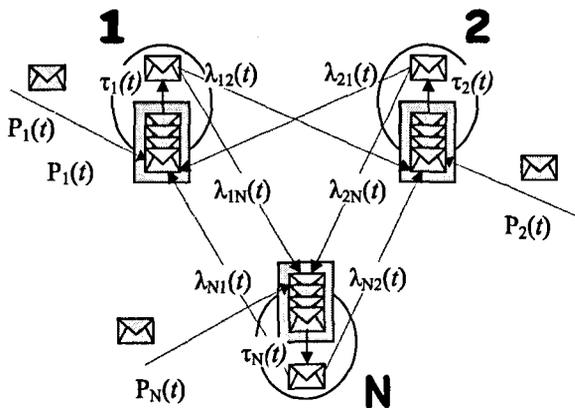


Рис. 1. Архитектура многоэтапной событийной системы (МСС) на основе обмена сообщениями

МСС на основе обмена сообщениями состоит из  $N$  компонентов, обменивающихся между собой сообщениями с частотой  $\lambda_{ij}(t)$  зависящей от времени, где  $i$  индекс компоненты-отправителя,  $j$  индекс компоненты-получателя  $i, j = 1, N$ . Возникновение сообщения в  $i$ -ой компоненте обязано какому-то внешнему событию, например, поступлению данных от измерительного оборудования. Вероятность возникновения сообщения в  $i$ -ой компоненте описывается функцией распределения  $P_i(t)$ . Сообщение содержит все важные для обработки события данные. Сообщение проходит маршрут из нескольких компонентов, каждая из которых выполняет над данными, содержащимися в сообщении, некоторую работу за время обслуживания  $\tau_i(t)$  зависящее как от характера работы, выполняемой на компоненте, так и от занятости всех компонентов, разделяющих общие вычислительные ресурсы (процессор). Когда  $i$ -ая компонента не успевает обрабатывать поступающие сообщения, то сообщения выстраиваются в очередь с предельным размером  $q_i^{\max}$ .

Отказ системы в данном случае определяется, как попытка поставить сообщение на одном из этапов его обработки в очередь, достигшую предельного размера. Заявки в системе не обладают приоритетом и предельно-допустимым временем обработки.

Число системных потоков  $l_i$ , ассоциированных с очередью сообщений  $i$ -ой компоненты, 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 позволяет обрабатывать сообщения параллельно. Общее число системных потоков не должно превышать фиксированной администратором величины  $T$ , связанной с ограничениями операционной системы.

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

Если определить число системных потоков  $l_i$ , выделенных на обработку сообщений каждой компоненты, то можно распределить процессорное время рабочей станции для уменьшения среднего времени обработки  $\tau_i(t)$  на одних компонентах, за счёт увеличения среднего времени обработки на других компонентах.

В предложенном автором механизме распределения вычислительных ресурсов определяется число системных потоков, выделенных на обработку очереди входящих сообщений  $l_i$  через равные интервалы времени из следующих соображений:

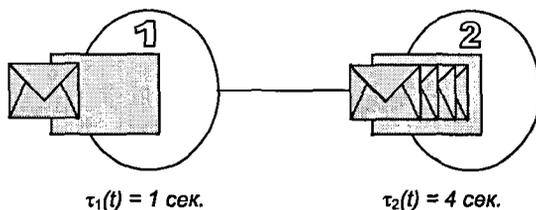


Рис. 2. Время обслуживания двух компонентов отличается в 4 раза

Предположим, система состоит из двух компонентов, где время обслуживания первой  $\tau_1(t) = 1 \text{ сек.}$  в 4 раза меньше времени обслуживания второй  $\tau_2(t) = 4 \text{ сек.}$ , а размер пула потоков  $T = 5$  (рис. 2).

Рассмотрим сценарий, в котором первая компонента принимает сообщения с равным интервалом, передавая сообщения на обработку второй компоненте.

В случае, когда компонентам будет выделено по одному системному потоку ( $l_1 = l_2 = 1$ ), то первая компонента будет успевать обрабатывать 4 сообщения и передавать их второй компоненте, за время, пока вторая компонента будет успевать обрабатывать лишь одно сообщение. Очередь сообщений второй компоненты начнёт увеличиваться, что приведёт к появлению отказа в системе.

Если же число системных потоков ( $T = 5$ ) распределить между компонентами пропорционально отношению времен обслуживания  $\tau_1(t)$  к  $\tau_2(t)$

$$l_1 = T * \tau_1(t) / (\tau_2(t) + \tau_1(t)) = 1$$

$$l_2 = T * \tau_2(t) / (\tau_2(t) + \tau_1(t)) = 4,$$

то производительность обеих компонентов окажется равной – вторая компонента будет успевать обрабатывать весь поток входящих сообщений от первой компоненты. Теперь, когда число принимаемых первой компонентой сообщений превысит возможности системы к обработке, то очереди начнут увеличиваться на обеих компонентах одновременно. Сценарий переполнения с одновременным ростом двух очередей сообщений (при  $l_1 = 1, l_2 = 4$ ) более предпочтителен, чем сценарий роста лишь одной из очередей (при  $l_1 = l_2 = 1$ ), поскольку первый отказ системы откладывается за счёт удержания в оперативной памяти большего числа сообщений, ожидающих обработки, чем в первом сценарии ( $q^{max}_1 + q^{max}_2 > q^{max}_2$ );

Обобщая рассуждения на  $N$  компонентов, мы получаем правило, согласно которому число потоков  $l_i$ , выделенных  $i$ -ой компоненте определяется из отношения суммарного времени обработки сообщений  $i$ -ой компоненты ( $\tau_i * n_i$ ) к суммарному времени обработки всех сообщений сети, как

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

В дополнение к распределению системных потоков в системе, используется приостановка обработки сообщений на компонентах-отправителях, что позволяет компонентам-получателям обработать свои очереди сообщений, приближающиеся к переполнению. Длительность приостановки  $i$ -ой компоненты-отправителя определяется пропорционально её вкладу в наполнение очереди  $j$ -ого получателя

$(\lambda_{ij})$  согласно штрафной функции  $f_{ij}(\lambda_{ij}, q_j, q_j^{th})$ , где  $q_j$  – текущий размер очереди  $j$ -ой компоненты-получателя,  $q_j^{th}$  – пороговое значение размера очереди  $j$ -ого получателя в диапазоне  $[0..q_j^{max}]$ , где  $q_j^{max}$  – предельный размер очереди, устанавливаемый администратором системы (алгоритм *обратного давления*).

Вид штрафной функции  $f$  выбирается эвристически, в общем случае величина штрафа оказывается тем большей, чем ближе очередь компоненты получателя к переполнению. В качестве примера штрафной функции  $i$ -ого отправителя может быть использована пороговая функция вида:

$$f_{ij}(\lambda_{ij}, q_j, q_j^{th}) = \begin{cases} k_{ij} \frac{\lambda_{ij}}{\max} (q_j - q_j^{th})^2, & q_j > q_j^{th} \\ \sum_{j=1} \lambda_{ij} & \\ 0, & q_j \leq q_j^{th} \end{cases}$$

Величина *силы штрафа*  $k_{ij}$  выбирается эвристически. Итоговая величина штрафа, назначаемая  $i$ -ой компоненте отправителю, определяется как сумма штрафов от компонентов получателей  $f_j$ .

*Основным результатом второй главы является построение математической модели МСС, компоненты которой разделяют общие вычислительные ресурсы. Предложен оригинальный механизм, основанный на перераспределении вычислительных ресурсов между компонентами, направленный на повышение отказоустойчивости в условиях пиковых нагрузок.*

**В третьей главе** приводится сравнительный анализ существующих механизмов распределения вычислительных ресурсов и предложенного механизма на примере типовых систем потоковой обработки информации в реальном времени<sup>2</sup>. Исследовалось время появления первого отказа в зависимости от используемого механизма, архитектуры системы, размеров очередей компонент, длительности обработки сообщений отдельными компонентами, параметров потока заявок на обслуживание.

В качестве системы потоковой обработки информации был выбран типовой сервер сбора и обработки информации (рис. 3). В обязанности сервера входит

<sup>2</sup> Под системой реального времени подразумевается система, реагирующая в предсказуемые времена на предсказуемый поток событий [12].

принятие сообщений от измерительного оборудования, обработка поступивших данных и сохранение результатов обработки в базе данных.

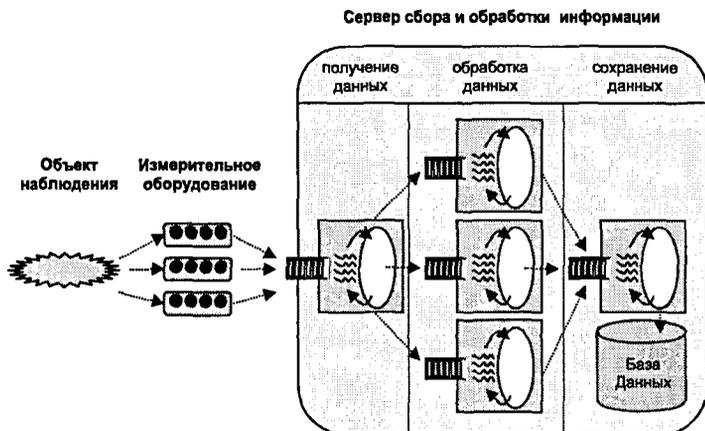


Рис. 3. Архитектура гипотетического сервера сбора и обработки информации

Для минимизации погрешности в серии тестов использовалась встроенная (in-memory) база данных. Поведение приложений с использованием внешних, исполняемых на других рабочих станциях компонентах, рассматривается в четвёртой главе.

Во всех проведённых тестах исследовались сценарии перегрузки системы. Наиболее отчётливо различие представленных механизмов распределения вычислительных ресурсов проявилось в сценариях *кратковременной* перегрузки. Так, до тех пор, пока система в состоянии справляться с числом поступающих в неё заявок, независимо от используемого механизма, в системе не наблюдается отказов. Когда система находится в состоянии длительной перегрузки – число отказов оказывается примерно одинаковым независимо от механизма распределения вычислительных ресурсов, потому, что число сообщений, которое система может обработать в секунду, не зависит от выбранного механизма. Различие механизмов проявляется в те временные интервалы, когда в систему поступает число сообщений, превышающее способности системы к обработке, но переполнение очередей не носит перманентный характер.

С точки зрения многоэтапной событийной архитектуры в выбранной для теста системе (рис. 4) присутствуют следующие 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-методу<sup>3</sup>) СУБД.

Операция получения пакетов от измерительного оборудования занимает ничтожно малое время порядка нескольких микросекунд. Данные пакета упаковываются в сообщение, которое поступает на обработку в компоненты слоя обработки данных. В этом слое каждая компонента отвечает за обработку данных одного-единственного типа пакетов.

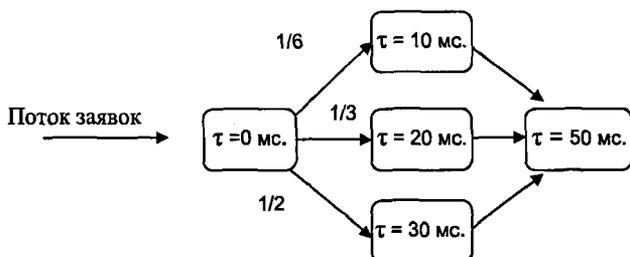


Рис. 4. Сценарий функционирования сервера сбора и обработки информации

Времена обработки сообщений  $\tau_i$  на компонентах и вероятности получения пакетов различных типов были выбраны так, как показано на рис. 4. Размеры очередей  $q_i^{\max}$  для всех компонент были установлены в 200 сообщений, размер пула потоков  $T$  был выбран 50. Исследования проводились на двухядерном компьютере и система, в рассматриваемом сценарии функционирования, была способна обработать порядка  $2 * (1000 / (50 + 10 * 1/6 + 20 * 1/3 + 30 * 1/2)) \approx 27$  пакетов в секунду. Интенсивность потока заявок возрастала с нуля и останавливалась на пиковом уровне, превышающем способности системы к обработке, например, так, как это показано на рис. 5.

<sup>3</sup> Встраиваемые (in-методу) базы данных, такие как, например, Sql Server CE (Compact Edition) широко применяются на устройствах с ограниченной памятью и слабым процессором: смартфонах, “наладонниках” (PDA), устройствах биометрии [13] и других интеллектуальных сенсорах.

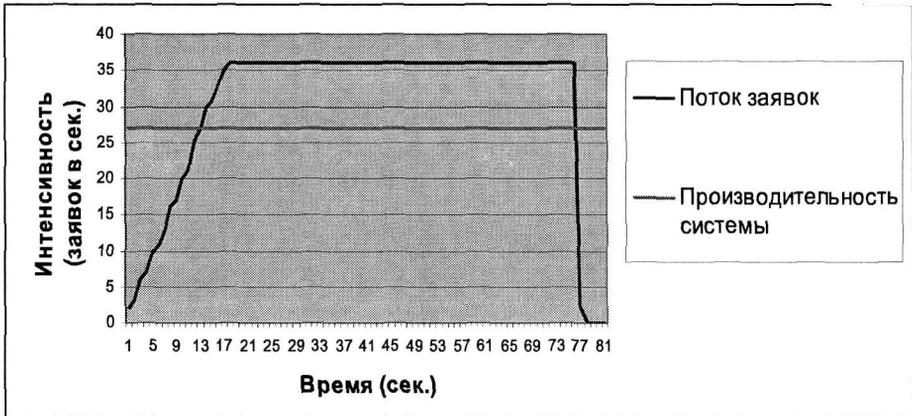


Рис. 5. Пример изменения интенсивности потока заявок, превосходящего возможности системы к обработке

Чем значительней интенсивность потока превышала уровень производительности системы, тем быстрее в системе наступал отказ при попытке поставить сообщение в заполненную очередь. Зависимость времени первого отказа от интенсивности потока с использованием различных механизмов распределения процессорного времени показана на рис. 6.

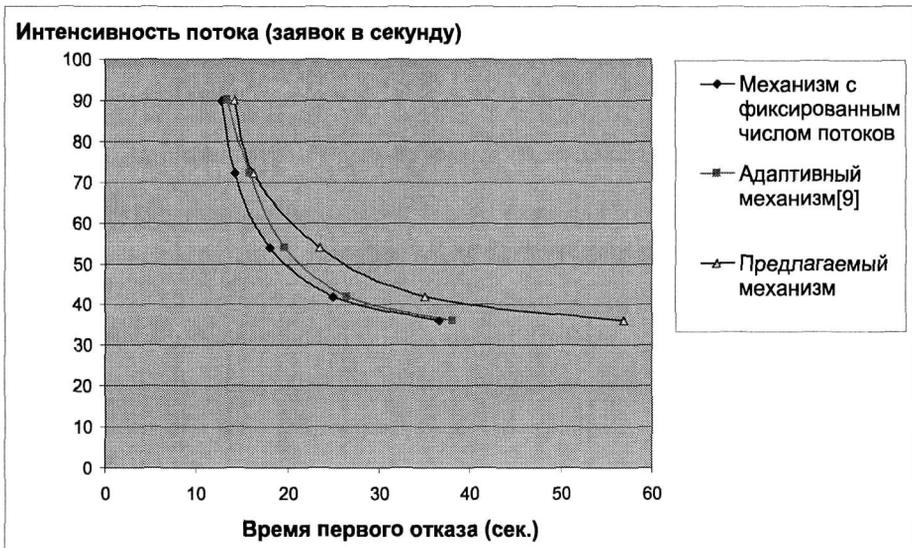


Рис. 6. Зависимость времени первого отказа от интенсивности потока заявок

Система с использованием предлагаемого механизма продемонстрировала наибольшее время наработки на отказ по сравнению с существующими

механизмами. Разница во времени первого отказа была тем существеннее, чем менее пессимистичный сценарий исследовался.

Повторное исследование зависимости времени первого отказа от интенсивности потока заявок были проведены после изменения вероятностей появлений пакетов различных типов так, как это показано на рис. 7.

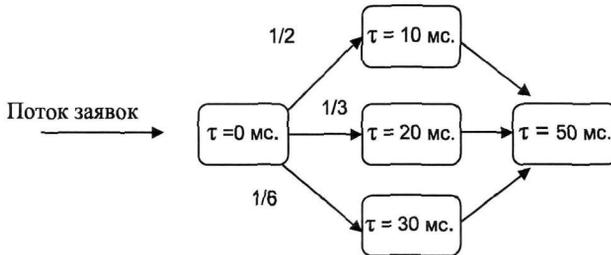


Рис. 7. Сценарий функционирования сервера сбора и обработки информации после изменения вероятностей появления пакетов различных типов

Результаты повторных испытаний представлены на рис. 8.

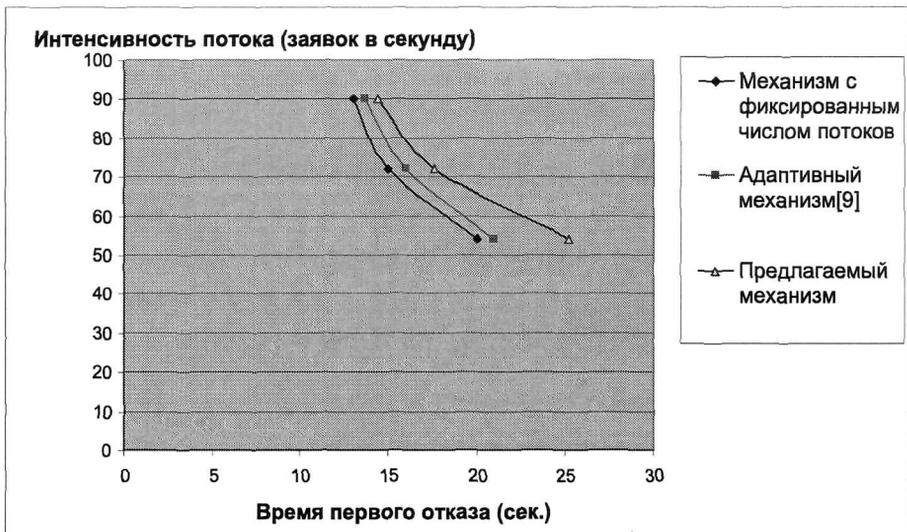


Рис. 8. Зависимость времени первого отказа от интенсивности потока заявок после изменения вероятностей появления пакетов различных типов

Система с использованием предлагаемого механизма распределения вычислительных ресурсов показала большее время наработки на отказ, как и в предыдущем сценарии.

Результаты исследования зависимости времени первого отказа от длины очереди компонент показаны на рис. 9.

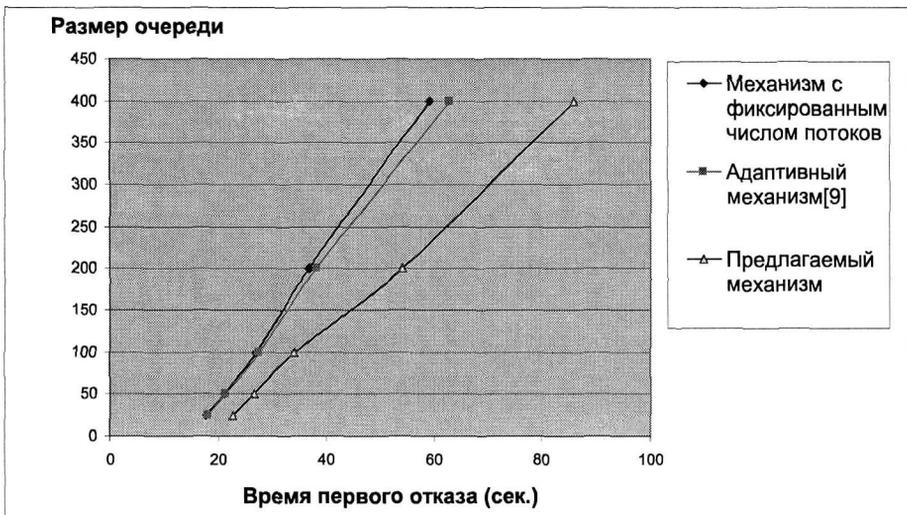


Рис. 9. Зависимость времени первого отказа от длины очередей компонент для интенсивности потока сообщений 36 сообщений в секунду

При уменьшении размера очередей отказ в системе наступал быстрее, однако наибольшее время первого отказа система показывала с применением предлагаемого механизма.

Изменение времени обработки на всех компонентах в  $k$  раз сохраняло большее время наработки системы на отказ с предлагаемым механизмом (рис. 10).

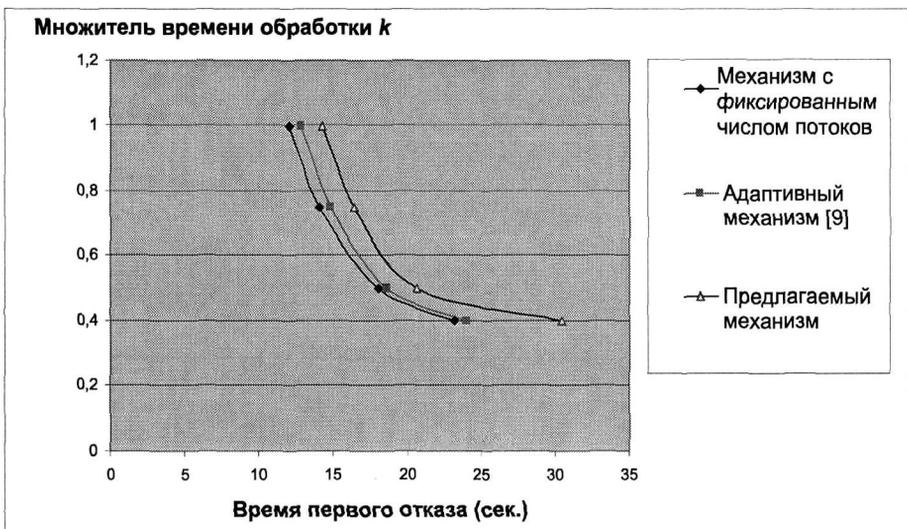


Рис. 10. Зависимость времени первого отказа при уменьшении длительности обработки сообщений компонентами системы

Исследования зависимости времени наработки на отказ от интенсивности проводились также для систем, где присутствовал компонент, формировавший набор команд измерительному оборудованию после обработки пакетов определённого типа (пунктир), так как это показано на рис. 11.

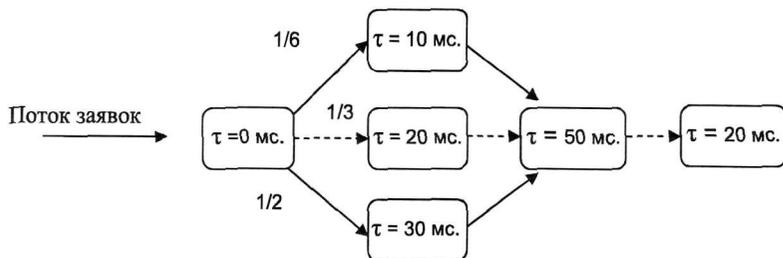


Рис. 11. Система, формирующая команды измерительному оборудованию после обработки пакетов определённого типа

Компонент, формировавший команд измерительному оборудованию, тративал 20 мс.. По результатам тестирования (рис. 12) лидерство по максимальному времени первого отказа осталось за предлагаемым механизмом.

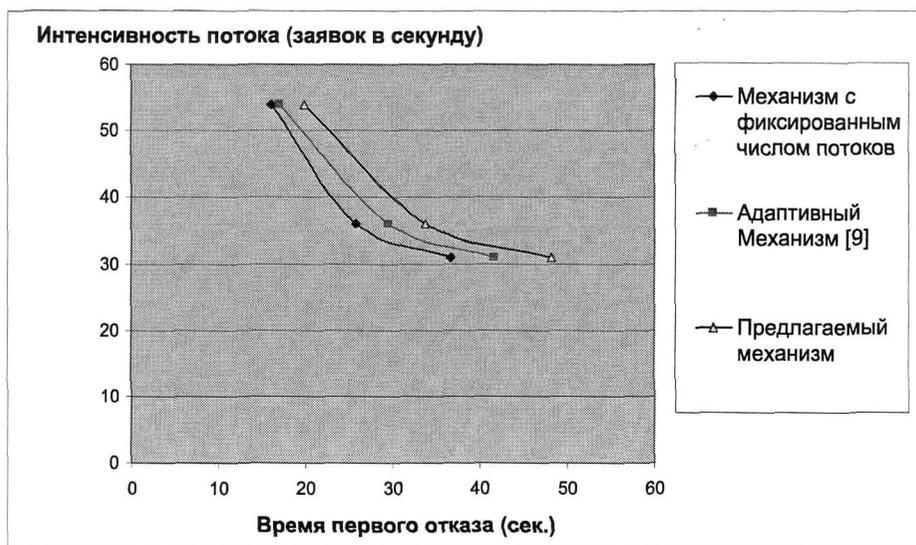


Рис. 12. Зависимость времени первого отказа от интенсивности потока для системы, отвечающей оборудованию на определённый тип пакетов после занесения результатов обработки в базу данных

Также исследовалась система, формировавшая набор команд измерительному оборудованию после получения пакетов определённого типа параллельно с занесением результатов обработки пакетов в базу (рис. 13).

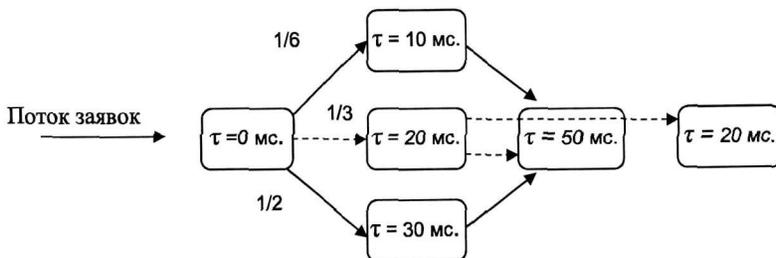


Рис. 13. Система, формирующая команды измерительному оборудованию на пакеты определённого типа параллельно с занесением результатов в базу

Время первого отказа оказалось максимальным с применением предлагаемого механизма (рис. 14).

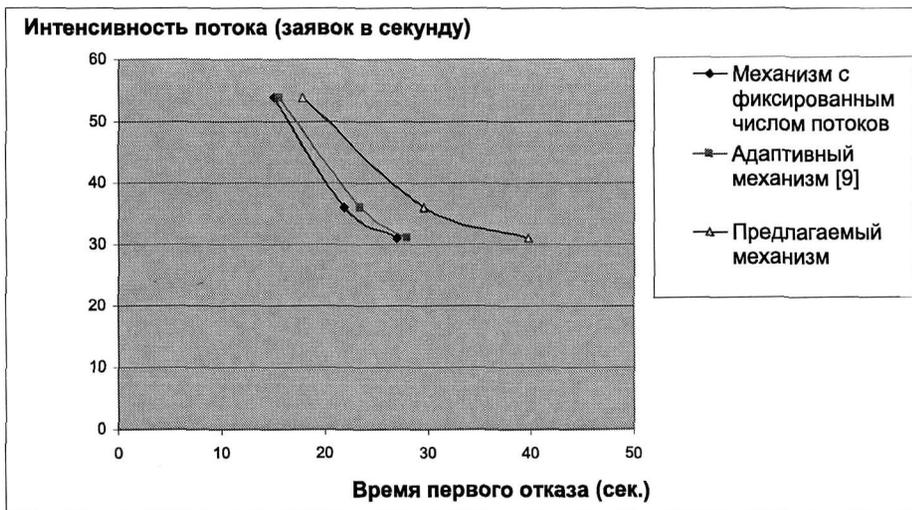


Рис. 14. Зависимость времени первого отказа от интенсивности потока для системы, отвечающей оборудованию на определённый тип пакетов параллельно с занесением результатов обработки в базу данных

Далее проводится детальный анализ поведения системы с применением различных механизмов. Исследуются отличительные черты поведения систем на однопроцессорном компьютере на примере типового сервера сбора и обработки информации. Как и в третьей главе в конфигурации системы присутствуют три слоя компонент. В зависимости от размера данных пакета и его структуры, обработка

занимает 1-3 мс., обращение к базе данных и выполнение хранимой процедуры выполняется за 50 мс. в отсутствии других задач.

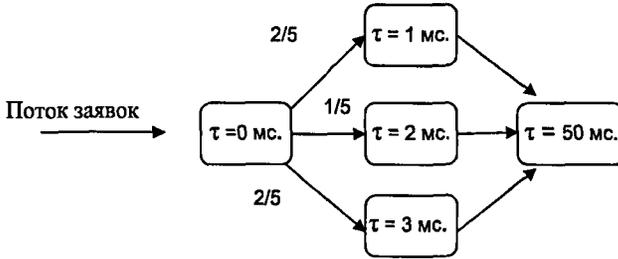


Рис. 15. Сценарий функционирования сервера сбора и обработки информации

В тестах гипотетический сервер сбора информации принимал сообщения от измерительного оборудования трёх типов, на обработку каждого из которых затрачивалось 1, 2 и 3 мс. Пакеты различных типов приходили с вероятностями 2/5, 1/5 и 2/5, как показано на рис. 15. В тестах размеры очередей компонентов  $q_i^{\max}$  были выбраны равными 200 сообщений.

В такой конфигурации однопроцессорная система способна обработать около 19 сообщений в секунду  $1000 / (50 + (1 * 0.4 + 2 * 0.2 + 3 * 0.4))$

В первом тесте создавалась кратковременная пиковая нагрузка, затем выдерживалась небольшая пауза, и сервер входил в режим длительной перегрузки, где каждую секунду сервер принимал на 5-6 пакетов больше, чем был в состоянии обработать (рис. 16).

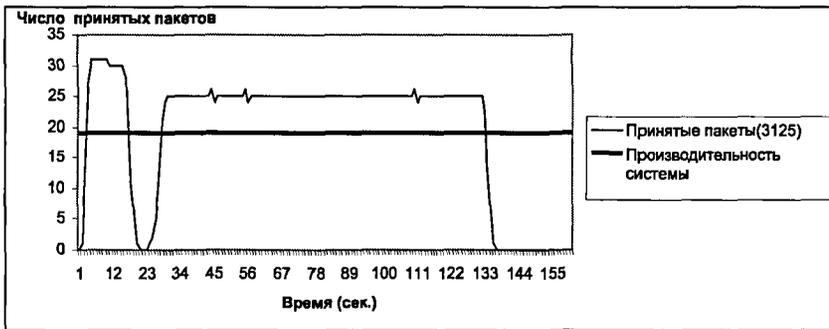


Рис. 16. Сценарий теста

В течение двух минут сервер принял к обработке 3125 пакетов. Сообщения начинают выстраиваться в очередях, ожидая обработки, что приводит в дальнейшем к регулярным отказам в обслуживании.

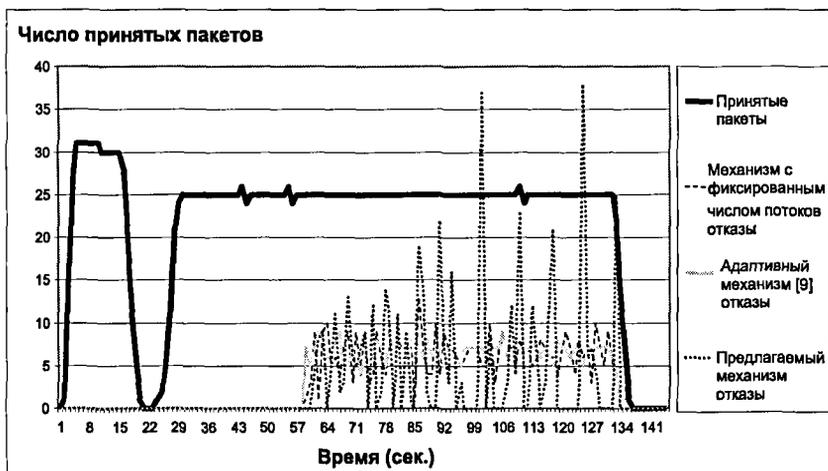


Рис. 17. Число потерь в системе при использовании различных механизмов распределения вычислительных ресурсов в тесте 1

По результатам теста (рис. 17) наименьшее число необработанных пакетов (площадь под графиками) было получено с использованием предлагаемого механизма (451, что на 5% лучше классического механизма с фиксированным числом системных потоков – 474).

Адаптивный механизм [9] показал практически эквивалентные результаты с классическим механизмом, использующим фиксированное число системных потоков (число отказов 479 и 474).

Отличительной особенностью разработанного в диссертации механизма является негладкий характер линии графика отказов. Это объясняется временными приостановками обработки сообщений с использованием алгоритма обратного давления. Обработка сообщений временно приостанавливалась на компонентах второго слоя (преобразование и обработка данных) при наполнении очереди компоненты третьего слоя (обращение к СУБД). В результате, отказ на компоненте третьего слоя откладывался, но создавались предпосылки для преждевременного отказа на приостановленных компонентах, 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 компоненты первого и второго слоя) позволял высвободить процессорное время для обработки поступающих

сообщений, что уменьшало общее число отказов. Первый отказ системы в рассмотренном тесте наступил на 2 секунды позже, чем первый отказ при использовании других механизмов распределения вычислительных ресурсов.

Однако, рассмотренная конфигурация не позволяла изучить особенности перераспределения системных потоков, выделяемых для обработки очередей входящих сообщений компонентов системы.

Во втором тесте исследовалось поведение сервера сбора и обработки информации, в котором время обработки сообщений на компонентах преобразования и обработки было увеличено в 10 раз, по сравнению с первым тестом. Условия второго теста позволяли исследовать поведение сервера при использовании различных алгоритмов распределения системных потоков.

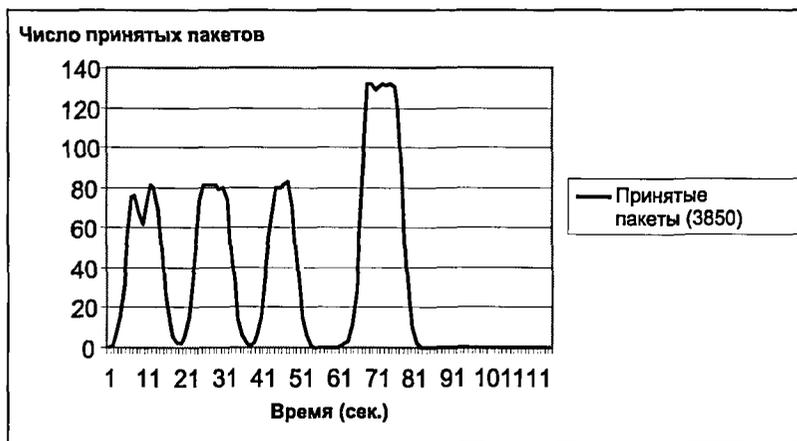


Рис. 18. Тест для исследования особенностей распределения системных потоков

Во втором тесте система приняла к обработке за полторы минуты 3850 пакетов от измерительного оборудования (рис. 18). В сценарии теста было создано четыре 15-секундных временных интервала, в течение которых система принимала к обработке в 6 раз больше пакетов, чем была в состоянии обработать.

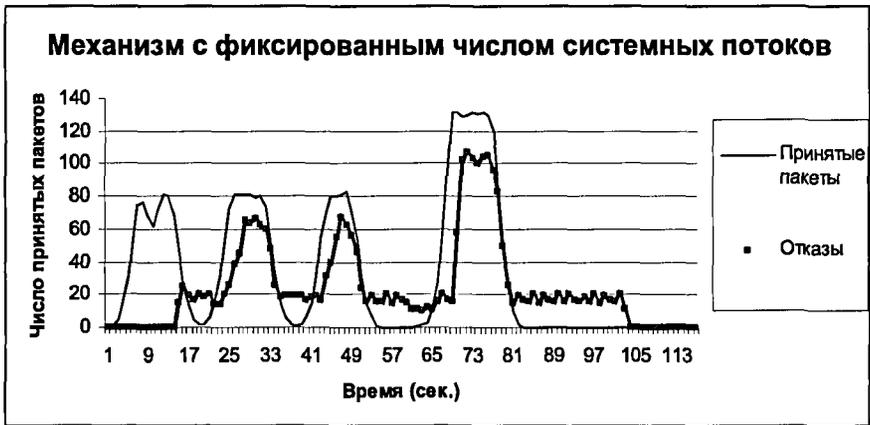


Рис. 19. Результаты тестирования системы с применением фиксированного числа системных потоков на компонентах системы

Классический механизм с фиксированным числом системных потоков (рис. 19) при одинаковом числе потоков, выделенных на обработку очереди сообщений каждой компоненты ( $l_i = const$ ), показал наибольшее (на 20%) число отказов (2840) из всех рассмотренных механизмов распределения вычислительных ресурсов. Отличительной чертой работы системы с применением механизма с фиксированным числом системных потоков являются отказы даже в те периоды, когда в систему не поступало никаких сообщений (период с 15 по 22 секунду от начала теста, периоды 50 – 65 сек. и 80 – 100 сек.). Подобное поведение объясняется тем, что компоненты первого и второго слоя, обладая равным числом системных потоков с компонентой третьего слоя (компонента обращения к базе данных), но меньшим временем обработки сообщения, продолжали отправлять сообщения в переполненную очередь компоненте третьего слоя, быстрее, чем та могла обработать. Вместо того, чтобы замедлить (за счёт искусственной приостановки или уменьшения числа рабочих потоков  $l_i$ ) частоту отправки сообщений  $\lambda_i$ , компоненты второго слоя обрабатывали и передавали сообщения в переполненную очередь третьего слоя, что вело к отказам даже тогда, когда в систему не поступало никаких сообщений.

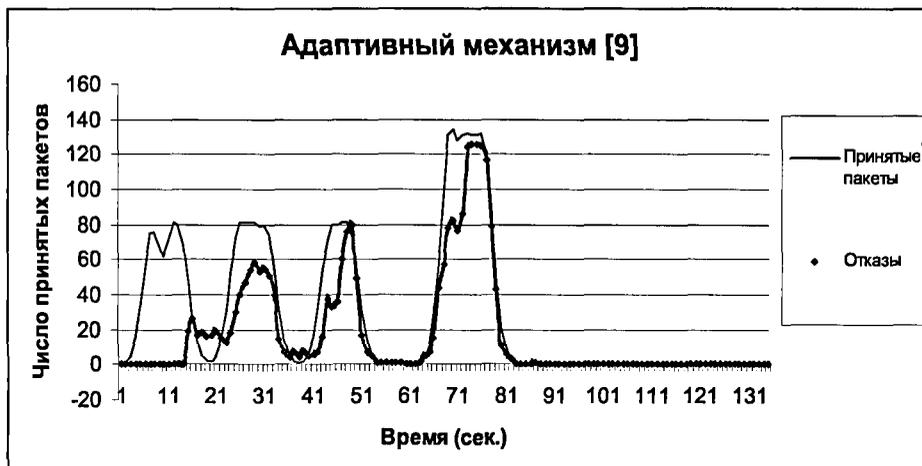


Рис. 20. Результаты тестирования адаптивного механизма [9]

При использовании адаптивного механизма [9] (рис. 20) сервер сбора и обработки информации продемонстрировал значительно лучшие результаты, чем при использовании механизма с фиксированным числом системных потоков. При этом число отказов уменьшилось на 20% и составило 2300. При распределении системных потоков  $l$  удавалось уменьшить потери за счёт перераспределения процессорного времени в пользу компоненты третьего слоя.

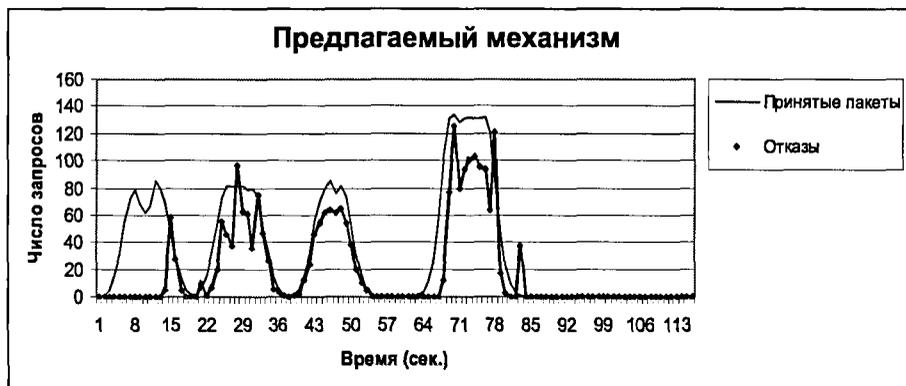


Рис. 21. Результаты тестирования предлагаемого механизма

Наименьшее количество отказов (2200) рассматриваемый в тестах сервер сбора и обработки информации продемонстрировал при использовании разработанного в диссертации механизма (рис. 21). Приостановка обработки сообщений с использованием алгоритма обратного давления, позволила

дополнительно уменьшить общее число отказов на 5% по сравнению с использованием адаптивного механизма [9] при погрешности, наблюдаемой в серии тестов, равной 1%.

Результаты тестирования показали, что с применением разработанного механизма без использования алгоритма обратного давления сервер сбора и обработки информации совершил такое же количество отказов, которое получается в результате применения адаптивного механизма [9] (рис. 22).

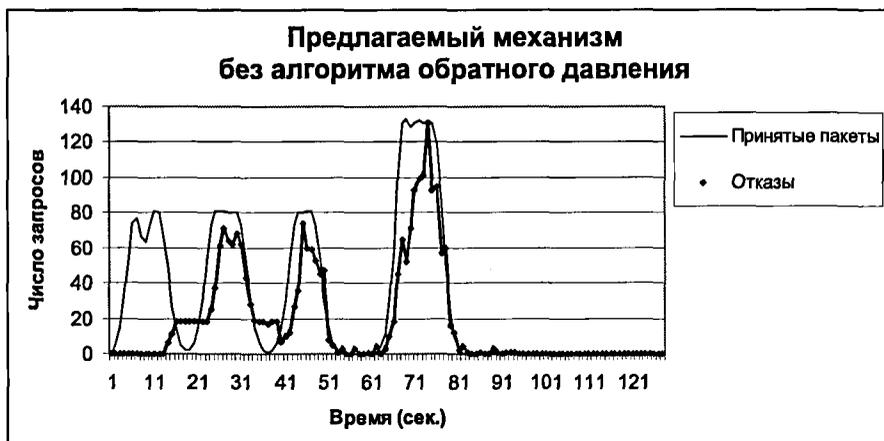


Рис. 22. Результаты тестирования предлагаемого механизма без алгоритма обратного давления

По результатам третьей главы делается вывод о том, что применение разработанного механизма при построении МСС в конфигурациях, типичных для SCADA систем, в полтора раза увеличивает время наработки на отказ по сравнению с применением существующих механизмов распределения вычислительных ресурсов в слабо пессимистичных сценариях нагрузки, когда число поступающих в систему заявок превосходит возможности системы к обработке на 50 – 100%. В сильно пессимистичных сценариях, когда интенсивность потока на порядок превосходит возможности системы к обработке, использование разработанного в диссертации механизма, тем не менее, даёт возможность увеличить время наработки на отказ на величину порядка ~ 5%.

Из изложенного следуют первое и второе защищаемые положения:

Математическая модель МСС, позволяет описывать поведение информационной системы, исполняемой на одной рабочей станции с общими для

всех компонентов вычислительными ресурсами и основанной на обмене сообщениями.

Предложенный механизм распределения вычислительных ресурсов МСС, основанный на перераспределении числа системных потоков и назначении времени приостановки обработки входящих сообщений на компонентах, повышает отказоустойчивость МСС в сценариях пиковых нагрузок.

В четвёртой главе рассматривается практическое применение предложенного механизма на примере разработанной автором многоэтапной событийной SCADA системы [7] *Jacinth*. *Jacinth* реализована с применением авторской библиотеки разработчика JME (Java Messaging Engine [8]), подробно представленной в приложении 1. JME упрощает создание многоэтапных событийных систем и содержит реализацию разработанного в диссертации механизма перераспределения вычислительных ресурсов.

В *Jacinth* был собран SCADA проект, опрашивающий два контроллера Кристалл-2С2<sup>4</sup> и визуализирующий состояние объекта наблюдения. Для хранения обработанных данных использовалась база данных Oracle 10g, доступная по локальной сети (рис. 23).

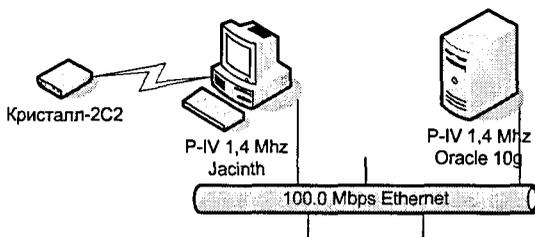


Рис. 23 Сетевая конфигурация SCADA проекта Jacinth

В SCADA проекте присутствовали две компоненты, опрашивающие Кристаллы-2С2; компонент ядра SCADA системы (используемый в качестве разделяемого всеми компонентами хранилища данных); компонент, отвечающий за интерпретацию и обработку полученной от измерительного оборудования информации (реализует модель объекта охраны); компонент визуализации состояния объекта охраны, а также компонент, выполняющий обращения к базе данных.

<sup>4</sup> Полное название: прибор приёмно-контрольный охранный магистральный ППКО-М "Кристалл-2С2.ХХ" еФ2.407.008-02. Производитель ОАО "Тензор", г. Дубна

Результаты тестирования SCADA проекта, использующего различные механизмы распределения вычислительных ресурсов в сценарии пиковой нагрузки, показали превосходство адаптивных механизмов над механизмом фиксированного числа потоков. В тоже время адаптивный механизм [9] и разработанный механизм показали примерно равное время наступления первого отказа.

*Результатом четвёртой главы является вывод о том, что предложенный и реализованный в библиотеке разработчика механизм распределения вычислительных ресурсов МСС может успешно применяться как в системах, функционирующих в пределах одной рабочей станции, так и в системах, компоненты которых размещены на нескольких рабочих станциях.*

Из изложенного следует третье защищаемое положение: специализированное программное обеспечение в виде библиотеки классов для создания отказоустойчивых МСС обработки информации на основе авторской модели и авторского механизма распределения вычислительных ресурсов позволяет разрабатывать МСС с большим, по сравнению с существующими механизмами, временем наработки на отказ.

В **Заключении** сформулированы основные результаты, полученные в диссертации:

1. Разработана математическая модель МСС для описания поведения информационных систем, исполняемых на одной рабочей станции с общими для всех компонентов вычислительными ресурсами и основанных на обмене сообщениями.
2. Разработан новый механизм распределения вычислительных ресурсов МСС, основанный на перераспределении числа системных потоков и назначении времени приостановки обработки входящих сообщений на компонентах, для повышения отказоустойчивости МСС в сценариях пиковых нагрузок.
3. Реализовано специализированное программное обеспечение в виде библиотеки классов, для создания отказоустойчивых МСС обработки информации на основе авторской модели и авторского механизма распределения вычислительных ресурсов, которое позволяет разрабатывать МСС с большим, по сравнению с существующими механизмами, временем наработки на отказ.

Результаты исследования могут быть эффективно использованы при построении МСС, для которых наиболее важным показателем работы является время наступления первого отказа.

Разработанный в диссертации новый механизм распределения вычислительных ресурсов в МСС, нацеленный на повышение времени наработки на отказ, может применяться при разработке широкого спектра программного обеспечения в различных областях человеческой деятельности (аэрокосмические исследования, энергетика, химическая промышленность, военно-промышленный комплекс и т.д.), где время наработки на отказ ИС является одним из важнейших требований.

## **Список работ, опубликованных по теме диссертации**

1. Никонов Э.Г., Флорко А.Б. Повышение отказоустойчивости многоэтапных событийно-управляемых систем анализа и обработки информации в сценариях стрессовой нагрузки // Информационные ресурсы России. – 2008. – №5 – С. 34 - 36.
2. Никонов Э.Г., Флорко А.Б. Повышение отказоустойчивости информационных систем, построенных на основе обмена сообщениями // Информационные технологии. – 2007. – №4. – С. 6 - 10.
3. Никонов Э.Г., Флорко А.Б. Повышение отказоустойчивости систем массового обслуживания в условиях стрессовой нагрузки: Сообщение ОИЯИ Р10-2009-85. – Дубна: ОИЯИ, 2009. –12 с.
4. Флорко А.Б. Разработка и реализация оптимизационного метода управления ресурсами в многоэтапных событийно-управляемых системах // Материалы 13-й научной конференции студентов, аспирантов и молодых специалистов. - Дубна, 2006. - Ч.2.-С.147-149.
5. Флорко А.Б. Управление ресурсами в информационных системах, построенных на основе многоэтапной событийно управляемой архитектуры SEDA. Материалы X конференции молодых учёных и специалистов ОИЯИ. – Дубна, 2006. – С. 214-218.
6. Флорко А.Б. Архитектура универсальной среды обмена сообщениями для создания масштабируемых решений // Материалы XII научной конференции студентов, аспирантов и молодых специалистов. – Дубна, 2005. – С.102-105.
7. Флорко А.Б. Масштабируемая архитектура универсальной SCADA системы. // Материалы IX конференции молодых учёных и специалистов ОИЯИ. – Дубна, 2005. – С. 59-63.
8. Свидетельство о государственной регистрации программы для ЭВМ в Роспатенте. № 2010611251. JME. Библиотека классов разработчика – Java Messaging Engine. Правообладатель: Флорко А.Б.

## Цитированная литература

9. M.D. Welsh. An architecture for Highly Concurrent, Well-Conditioned Internet Services. – Phd Thesis, Graduate Division of the University of California at Berkeley, 2002.
10. M. Welsh, D. Culler, E. Brewer. SEDA: An architecture for well-conditioned, scalable Internet services. In proceedings of the 18<sup>th</sup> ACM Symposium on Operating Systems Principles, Banff, Canada, October 2001.
11. S. Goldstein, K. Schauser, and D. Culler. Enabling primitives for compiling parallel languages. In Third Workshop on Languages, Compilers, and Run-Time Systems for Scalable Computers, Rochester, NY, May 1995.
12. Жданов А. А. Операционные системы реального времени. PC Week, 8/1999.
13. O. Krejcar, J. Cernohorsky. Recent Advances In Biology And Biomedicine archive. Proceedings of the 2nd WSEAS international conference on Computational chemistry table of contents. Puerto De La Cruz, Spain, 2008. pp: 80-84.