УДК 519.65, 004.75

# МУЛЬТИАГЕНТНАЯ СРЕДА ИССЛЕДОВАНИЯ АЛГОРИТМОВ БАЛАНСИРОВКИ НАГРУЗКИ В ОТКРЫТЫХ ИНФОРМАЦИОННЫХ СИСТЕМАХ

© 2014 г.

***Д.В. Жевнерчук, В.Л. Чепкасов***

Нижегородский государственный технический университет им. Р.Е. Алексеева

zhevnerchuk@yandex.ru

*Поступила в редакцию 26.11.2013*

Рассматривается методика исследования распределенных сервис-ориентированных программно-аппаратных комплексов, обладающих свойствами открытых систем. Методика построена на основе мультиагентного моделирования с подмешиванием данных мониторинга. Подробно описаны структура и поведение агентов, а также экспериментальная система, автоматизирующая процессы сбора статистики использования ресурсов, построения теоретических зависимостей, проведения имитации.

*Ключевые слова:* мультиагентная система, моделирование, гетерогенная информационная система, балансировка нагрузки, сервер, открытая система.

## Введение

Открытые информационные системы представляют собой сложные программно-аппаратные комплексы со встроенными механизмами поддержки расширяемости, масштабируемости и интероперабельности [1, 2]. В работе [3] отмечается, что несмотря на дорогостоящие проекты компаний IBM, Microsoft, Oracle, SAP и др. в настоящее время по-прежнему отсутствует эффективная платформа для интеграции разнородных программных комплексов. В частности, до конца не решены вопросы включения клиентских и серверных узлов в произвольной точке вычислительной сети и динамического масштабирования ресурса. Известные алгоритмы балансировки нагрузки, определяющие механизмы масштабирования вычислительных систем, в основном применяются к кластерным и GRID системам, ориентированным на ограниченный круг задач. В облачных системах, структура которых динамически меняется, они оказываются неэффективными.

Таким образом, вопросы создания технологии открытых информационных систем и эффективных средств исследования облачных систем с динамической распределенной структурой остаются актуальными.

## 1. Постановка задачи

Объектом исследования являются распределенные информационные системы, включающие множество вычислительных узлов, построенных на основе различных программно-аппаратных платформ. Каждый узел обрабатывает 1 и более сервисов и дополнительно оснащается службой мониторинга, посредством которой собираются данные об обработке процессов и ресурсах, которые они потребляют. Основная вычислительная нагрузка ложится на серверные узлы, входящие в состав облака. Структура облака и нагрузка могут динамически меняться в процессе функционирования. Таким образом, исследуются сложные открытые информационные системы, которые, в общем случае, включают различные типы распределенных вычислительных систем, находящихся во взаимодействии друг с другом.

В работах [4, 5] отечественных и зарубежных ученых представлен исчерпывающий, с большим количеством ссылок, аналитический обзор моделей балансировки нагрузки. Основной чертой полученных результатов является ориентация на частные случаи вычислительных систем, такие как кластеры и GRID, и для применения их в открытых информационных системах требуются дальнейшие исследования. В работах [6–9] представлены модели, средства имитации и мониторинга масштабируемых сетей, а также ссылки на дополнительные аналитические обзоры и отдельные исследования. Основными недостатками существующих систем мониторинга являются:

– ограничения по сбору статистики использования ресурсов процессов, которые функционируют под управлением различных виртуальных машин;

– ограничения по развертыванию в масштабах глобальной сети Интернет;

– отсутствие единой среды, позволяющей исследователю применять в комплексе методы имитации, мониторинга, интеллектуальной обработки данных к большим массивам статики мониторинга и симуляции.

В рамках текущей работы необходимо решить задачу автоматизации процесса исследования масштабируемости сложных облачных систем, для чего требуется разработать модель и единую среду исследования алгоритмов балансировки нагрузки с подмешиванием данных мониторинга гетерогенных [2] информационных систем.

Таким образом, требуется рассмотреть вопросы интеграции средств мониторинга, систем моделирования, OLAP и Data Mining технологий, что в комплексе позволяет исследовать модели балансировки нагрузки в сетях с различной конфигурацией и интенсивностью запросов, что выделяет текущую работу из числа существующих подходов и технологий исследования облачных систем.

## 2. Теоретическая часть

Рассмотрим вычислительную систему, включающую $n$ узлов, разбитых на 4 подгруппы, и описываемую множеством

$$N(C,R,B,S,U),$$

где $C$ – множество клиентских узлов, $R$ – множество серверов маршрутизации, $B$ – множество серверов балансировки нагрузки, $S$ – множество серверов ресурсов, $U$ – множество пользователей.

Множество пользователей представляет собой внешнюю среду, задающую нагрузку.

Все объекты, которые входят в состав системы, могут быть представлены «реагирующими агентами» [9], действия которых определяются только текущим состоянием. Агенты представимы функцией

$$F : S \to A,$$

где $S$ – множество состояний агента, $A$ – множество действий агента.

Рассматриваемая открытая информационная система моделируется пятью группами агентов:

1. Агент «Пользователь» UA с действиями: запросить сервис *CallService* и подготовить данные для запроса к сервису $PrepData_i$. Индекс $i$ определяет номер шага решения задачи в диалоговом режиме. Если конечный ответ формируется после единственного запроса, индекс можно опустить.

2. Агент «Клиент» CA с действиями: запрос списка сервисов *CallServiceList*, запрос адреса сервера с требуемым сервисом *CallServiceAdr*, запрос сервиса *CallService*, запрос пользовательского интерфейса *CallUI*, переход результата пользователю *SendResForUser*, переход в ожидание запроса от пользователя *WaitUserReq*.

3. Агент «Сервер маршрутизации» RA с действиями: формирование списка сервисов *CreateServiceList*, выполнение поиска адреса для запрашиваемого сервиса *GetAdrFromTable*.

4. Агент «Сервер ресурсов» ResA с действиями: запуск сервиса *RunService*, обработка запроса пользователя *ProcessReq*, ожидание запроса на запуск сервиса или на обработку *WaitClientReq*.

5. Агент «Сервер балансировки нагрузки» BA с действиями: формирование кластера серверов ресурсов, поддерживающих выполнение определенного сервиса *CreateCluster*, классификация сервиса *ClassifService*, выполнение балансировки нагрузки *Balance*.

6. Агент «Процесс мониторинга» MA с действиями: сбор параметров использования процессора(ов) и памяти на сервере ресурсов *GetStat*.

Ресурсы вычислительного узла резервируются за отдельными программами, на базе которых может быть развернут 1 и более сервисов. В момент запуска создается процесс с некоторым определенным по умолчанию приоритетом. Значение приоритета может меняться алгоритмами балансировки нагрузки. В модели реализована шкала приоритетов от 0 до 1. Считая 1 самым высоким приоритетом, загрузку *CPU* можно рассчитать по формуле

$$CPU = \sum_{i=1}^{n} p_i \cdot 100\%,$$

где $p_i$ – приоритет $i$-го процесса.

В случае, когда загрузка узла составляет более 100%, доля процессорного времени для $k$-го процесса определяется следующим образом:

$$CPU_k = p_k \Big/ \sum_{i=1}^{n} p_i ,$$

где $p_i$ – приоритет $i$-го процесса, $p_k$ – приоритет $k$-го процесса.

Выделим два типа приоритетов: фиксированный и масштабируемый. Фиксированным приоритетом обладают процессы, для которых объем потребляемого ресурса остается постоянным даже в случае его перегрузки. Масштабируемый приоритет определяет долю ресурса, которая в случае перегрузки зависит от приоритетов других процессов. Для процессов с масштабируемым приоритетом загруженность процессора определяется по формуле

$$CPU_k = p_k \left(1 - \sum_{i=1}^{n} h_i\right) \Big/ \sum_{i=1}^{n} p_i ,$$

где $p_i$ – динамический приоритет $i$-го процесса, $p_k$ – динамический приоритет $k$-го процесса, $h_i$ – фиксированный приоритет $i$-го процесса.

Согласно построенным формулам загрузка процессора может быть меньше 100%, даже если имеется множество низкоприоритетных процессов. Эта особенность используется для моделирования:

– режимов реального времени;

– ситуаций, когда требуется при минимальной задержке запускать сервисные процессы операционных систем;

– алгоритмов балансировки нагрузки, в которых минимизируется время переброски процессов между узлами.

Эффективность алгоритмов балансировки оценивается по средневзвешенному времени обращения:

$$W = \frac{\sum_{i=0}^{N} \frac{t_i^e - t_i^b}{T}}{N},$$

где $t_i^e$ – момент времени завершения обработки процесса, $t_i^b$ – момент времени начала обработки процесса, $T$ – время обработки процесса в режиме монопольного использования ресурсов, $N$ – количество обслуживаемых процессов.

Сформулируем допущения, принятые при построении мультиагентной модели:

1. В системах предусмотрено несколько слоев активности: пользователи, клиентские системы, серверные вычислительные узлы (серверы ресурсов), адресные серверы.

2. Активность пользователей определяется законами распределения интервалов между передачей соседних заявок.

3. Введено 3 класса сервисов: длительные во времени; диалоговые, которые присутствуют в памяти длительное время, но обладают невысокой степенью загрузки процессора; разовые, характеризующиеся малым периодом обработки запроса и выгружаемые сразу после обработки.

4. Пользователи сгруппированы по категориям, которые определяют тип и режим использования сервисов.

5. Каждый сервер ресурсов моделируется вместе с собственной системой мониторинга, посредством которой становится доступна статистика использования процессора и памяти.

В состав моделируемой системы входят серверные вычислительные узлы с установленным гипервизором, операционной системой и сервисами на базе прикладного программного обеспечения. Пользователи передают запросы посредством клиентских систем. Предварительно список доступных сервисов запрашивается у адресных серверов. Далее клиентская система передает данные о выбранном сервисе серверу балансировки нагрузки, на котором формируется кластер узлов, поддерживающих выполнение сервиса. Клиенту передается адресная информация сервера ресурсов с наименьшим показателем фактической и прогнозируемой загрузки. Выполняется запуск приложения, поддерживающего выбранный сервис. От клиентских систем поступают заявки на использование сервисов. В работе [10] предлагается методика построения моделей запросов пользователей к сервисам информационных систем.

На серверах ресурсов в фоновом режиме выполняется система мониторинга, которая периодически собирает статистику об использованных ресурсах и передает ее серверу балансировки нагрузки. Если прогнозируется перегрузка, то выбираются узлы из ранее сформированного кластера и сервисы, выполнение которых прерывается. Для сервисов класса 1 будут переданы промежуточные результаты. Клиентам, взаимодействующим с сервисами класса 2, передается адресная информация о новом ресурсном сервере.

На рис. 1 представлены диаграммы состояний для агентов «Клиент» и «Сервер маршрутизации».

Агент «Клиент» изначально находится в режиме ожидания пользователя. Переходы 1, 5 в состояние ожидания списка сервисов могут быть выполнены либо по тайм-ауту, либо по таблице расписания пользовательской активности. Переходы 2, 3, 4, 7, 8 также выполняются по тайм-ауту и определяются законами распределения периодов времени передачи запросов по каналам связи и обработки на серверах, а также интервалами времени, в течение которых пользователи формируют запросы на клиентах. Переход 6 выполняется при получении результатов обработки пользовательского запроса, а переход 9 выполняется при получении результатов обработки всех пользовательских запросов. Во всех состояниях, кроме «Ожидание пользователя» и «Ожидание UI», формируется и передается сообщение агентам «Сервер маршрутизации», «Сервер ресурсов», «Балансировщик нагрузки», «Пользователь».

Агент «Сервер маршрутизации» изначально ожидает сообщения от клиента или сервера ресурсов, при получении сообщения 1 от клиента переходит в состояние формирования списка, в рамках которого передает сообщение 2 агенту «Клиент». Если приходит сообщение 2 от сервера ресурсов, то возникает состояние, в рамках которого вносятся изменения в адресную таб-

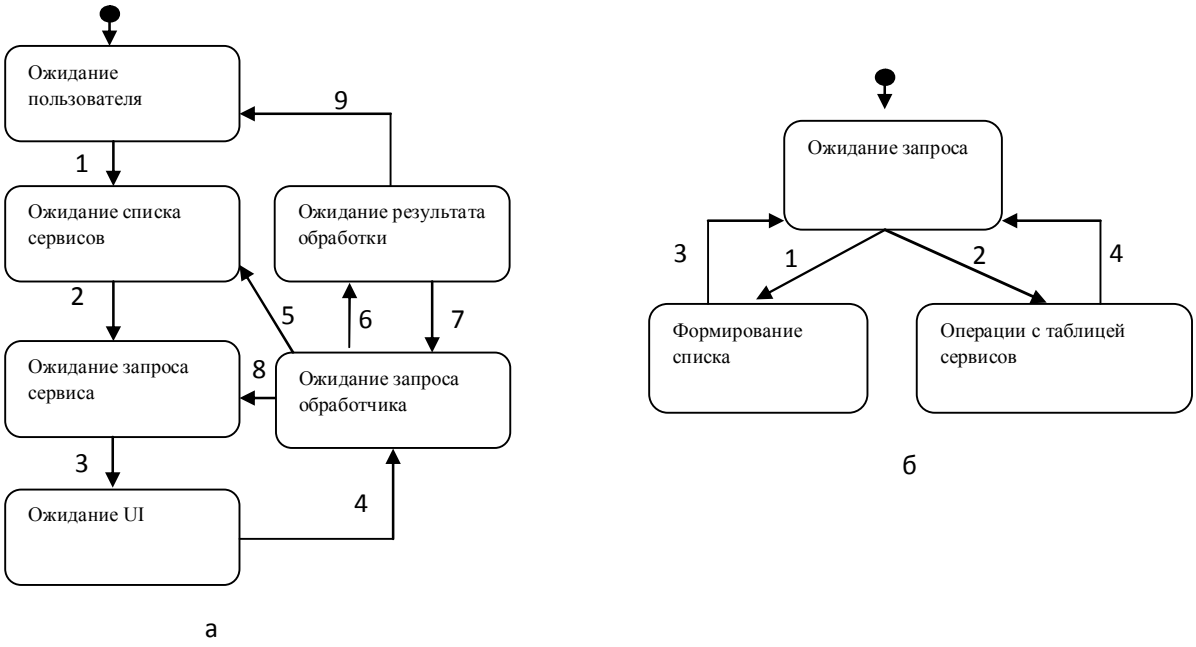


Рис. 1. Диаграмма состояний агентов: а) «Клиент», б) «Сервер маршрутизации»

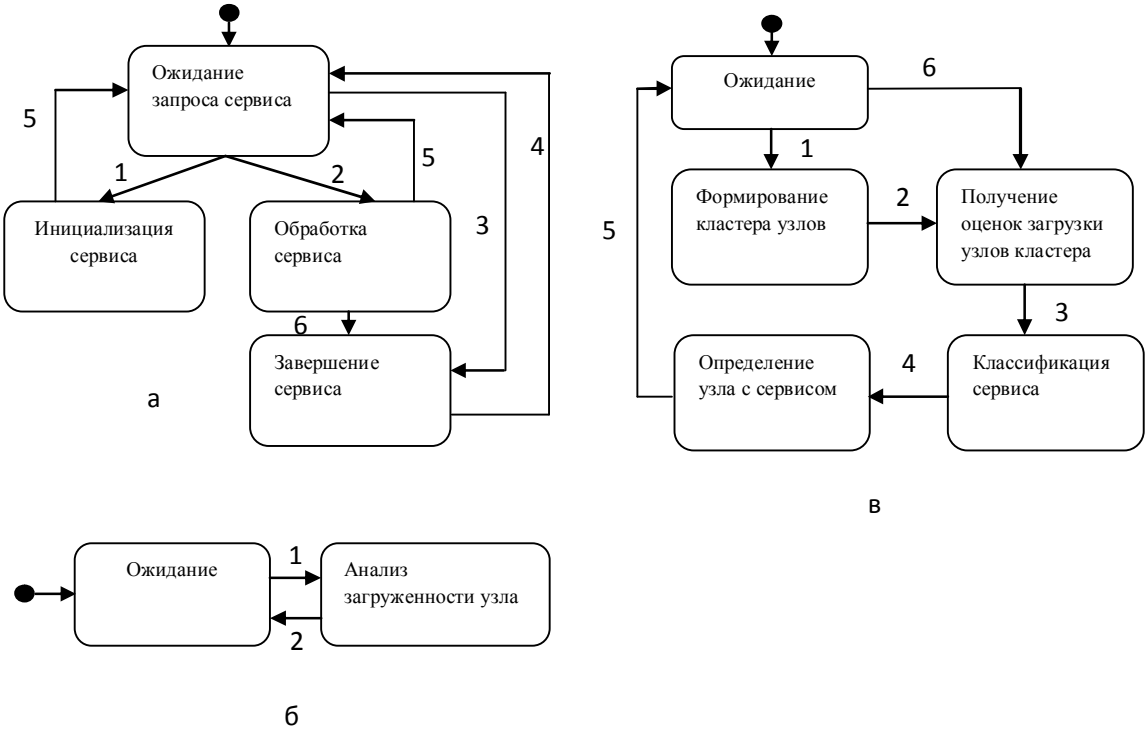


Рис. 2. Диаграмма состояний агентов: а) «Сервер ресурсов», б) «Процесс мониторинга», в) «Балансировщик нагрузки»

лицу ресурсов. Сообщения 3 и 4 адресного сервера вызываются по тайм-ауту.

На рис. 2 представлены диаграммы состояний для агентов «Сервер ресурсов», «Процесс мониторинга», «Балансировщик нагрузки».

Агент «Сервер ресурсов» переходит в состояние инициализации сервиса при получении сообщения 1 от клиента или от агента «Балансировщик нагрузки». Состояние «Завершение сервиса» вызывается по окончании обработки пользовательского запроса (сообщение 6) либо при получении сообщения 3 от агента «Балансировщик нагрузки». Переходы 2, 4, 5 выполняются по тайм-ауту. Агент «Процесс мониторинга» по тайм-ауту переходит в состояние анализа загруженности узла и в случае фактической или прогнозируемой перегрузки посылает сообщение 6 агенту «Балансировщик нагрузки». Переходы 2, 3, 4 агента «Балансировщик нагрузки» выполняются по состояниям тайм-аута,

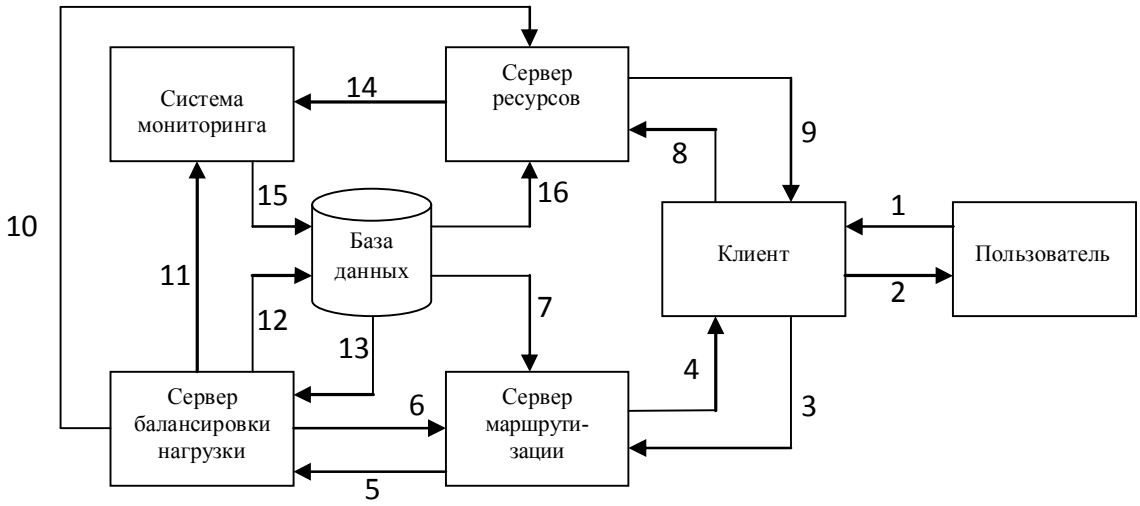


Рис. 3. Функциональная схема модели

которые зависят от применяемых алгоритмов data mining и объема базы знаний о сервисах, выполняемых на узлах. Из состояния «Определение узла с сервисом» выполняется передача сообщения 4 агенту «Сервер ресурсов».

На функциональной схеме (рис. 3) видно, что работа агентов происходит распределенно. Пользователь формирует заявки и отправляет сообщение 1 «Инициализация сервиса», далее ожидает сообщение 2 «Завершение выполнения».

Клиент при получении сообщения 1 отправляет сообщение 3 инициализации сервиса серверу маршрутизации и ожидает сообщение-подтверждение доступности сервиса 4. После получения сообщения 4 клиент отправляет сообщение 8 на выполнение сервиса. Также возможна отправка сообщения «Остановка сервиса», если есть необходимость завершить обработку задачи преждевременно. При получении сообщения 9 клиент переходит в режим ожидания завершения обработки задачи или ожидает заявки от пользователя (сообщение 1).

Сервер маршрутизации при получении сообщения 3 собирает информацию о серверах ресурсов (сообщение 7), отправляет запрос балансировки (сообщение 5) и ожидает наиболее подходящий ресурс (сообщение 6). Далее происходит отправка ресурса клиенту (сообщение 4).

Сервер ресурсов ожидает сообщения 8 или 10, после получения которых он инициализирует или завершает процесс. Сообщение 10 от сервера балансировки также может означать, что нужно перенести данные на другой сервер. Сообщение 16 позволяет получить необходимую информацию о выполняемых процессах.

Сервер балансировки нагрузки отправляет сообщение 11 с целью сбора информации о состояниях серверов ресурсов. Сообщения 13, 10 и 6 необходимы для балансировки нагрузки.

## 3. Описание эксперимента

Экспериментальное исследование построенных моделей проводилось на базе программно-аппаратного ресурса ИжГТУ им. М.Т. Калашникова, включающего вычислительные узлы двух типов: на основе платформ Windows и *nix. Дополнительно выделен компьютер для удаленного управления процессами (далее – координатор) на узлах через telnet, в задачи которого входит:

– запуск процессов на удаленных узлах под управлением Windows и *nix платформ;

– остановка и удаление процессов на удаленных узлах под управлением Windows и *nix платформ.

Также на координаторе установлена СУБД MySQL, система WEKA Data Mining, хранится xml файл с расписанием запуска процессов и клиент telnet, с помощью которого процессы могут быть запущены вручную. На отдельном узле развернута система моделирования Anylogic.

Экспериментальная проверка построенных моделей включает два основных этапа:

1. Подготовка программно-аппаратного комплекса, определение множества вычислительных процессов, формирующих основную вычислительную нагрузку. Получение эмпирических данных о работе процессов в монопольном режиме, а также в режимах разделения времени. Расчет средневзвешенного времени отклика процессов. Запись полученной статистики в хранилище аналитической информации для последующей передачи и обработки системами интеллектуального анализа данных.

2. Применение системы WEKA Data Mining для формирования эмпирических зависимостей, используемых в модели на основе статистики, полученной на первом этапе. Проведение ими-

*Таблица 1*

**Результаты мониторинга системы**

| № эксп. | Время выполнения процессов группы 1 в монопольном режиме, с | Время выполнения процессов группы 2 в монопольном режиме, с | Кол-во набл. | | Средневзвешенное время обращения по отдельному наблюдению с учетом переходных средних, с | | | | Средняя величина расхождения результатов одной пары процессов в разных наблюдениях, % | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | RR | WRR | RR | | WRR | | RR | WRR |
| | | | | | Min | Max | Min | Max | | |
| 1 | 23 | 201 | 10 | 10 | 5.6/ 1.6 | 6.4/ 1.73 | 2.6/ 1.05 | 2.9/ 1.16 | 9.1 / 8.0 | 10.0 / 9.8 |
| 2 | 43 | 405 | 10 | 10 | 4.81/ 1.42 | 5.3/ 1.51 | 2.07/ 1.09 | 2.29/ 1.15 | 7.7 / 6.1 | 10.6 / 8.1 |
| 3 | 60 | 201 | 10 | 10 | 2.35/ 1.66 | 2.71/ 1.8 | 1.28/ 1.94 | 1.4/ 2.11 | 11.6 / 8.4 | 8.9 / 9.4 |
| 4 | 78 | 405 | 10 | 10 | 3.18/ 1.49 | 3.58/ 1.6 | 1.3/ 1.2 | 1.43/ 1.26 | 11.9 / 7.8 | 10.2 / 5.5 |
| 5 | 23 | 249 | 10 | 10 | 5.8/ 1.34 | 6.3/ 1.51 | 2.8/ 1.03 | 3.2/ 1.08 | 9.6 / 10.8 | 10,7 / 6.2 |

тации для установления валидности модели: формирование списка входных параметров эксперимента с вычислительной сетью и имитационного эксперимента с моделью. Расчет средневзвешенного времени отклика процессов на основе результатов моделирования.

**Для проведения первого этапа** были разработаны программы, осуществляющие расчет параметров, заданных итерационными схемами с фиксированным количеством шагов. Реализованы схемы двух типов. Время выполнения первой схемы линейно зависит от выбранного диапазона модельного времени, а время выполнения второй – экспоненциально. В программах реализованы функции, которые по таймеру записывают в log-файл значение параметра, полученное на текущем шаге схемы. На каждом вычислительном узле функционируют службы мониторинга, которые выполняют обращения к операционным системам и считывают параметры ресурсов процессора и памяти, используемые прикладными процессами.

Сведения о работе процессов, записанные в файлы, передаются координатору через ftp. Далее информация записывается в базу данных MySQL.

Координатор, используя WEKA, строит функции регрессии для расчета времени выполнения процесса при заданных начальных условиях, а также при наличии других одновременно обрабатываемых процессов:

$$T_{p1} = \alpha(b - a), \quad (1)$$

$$T_{p2} = \lambda(e^{\beta(b-a)} - 1), \quad (2)$$

где $a$, $b$ – параметры, определяющие интервал модельного времени; $\alpha, \beta, \lambda$ – коэффициенты.

На первом этапе собираются сведения о функционировании прикладных процессов, реализующих итерационные расчетные схемы, и о параметрах используемых ресурсов. Замеры проводятся для процессов, запущенных как в одиночном режиме, так и в режиме разделения времени с другими процессами.

**На втором этапе** были разработаны списки входных параметров для сбора сведений об эффективности алгоритмов балансировки в реальной сети и с помощью имитационной модели. В режиме проверки достоверности построенной модели применялись простые алгоритмы балансировки нагрузки 4 уровня модели OSI: RR, WRR [10]. Были проведены эксперименты с агентной моделью и реальной сетью.

В таблице 1 приведены результаты мониторинга выполнения двух групп процессов на 5 однотипных узлах, а также результаты имитации, полученные с помощью разработанной агентной модели. Среднее время выполнения процессов первой группы в монопольном режиме от $\min(t_1) = 23$ с до $\max(t_1) = 78$ с. Среднее время выполнения процессов второй группы в монопольном режиме от $\min(t_2) = 201$ с до $\max(t_2) = 255$ с. В столбцах 6–11 таблицы до черты указываются значения средневзвешенного времени обращения для процессов первой группы, а после черты – для второй группы.

Из процессов первой и второй групп были сформированы 5 пар, с которыми проводились эксперименты. Интервалы времени между запуском процессов первой и второй групп равномерно распределены на [10; 30] с. Для каждой пары было выполнено 10 экспериментов. Длительность отдельного эксперимента 120 минут.

*Таблица 2*

**Результаты имитации**

| № | Средневзвешенное время обращения по отдельному прогону с учетом переходных средних, с | | | | Средняя величина расхождения результатов моделирования и мониторинга, % | | Вывод |
|---|---|---|---|---|---|---|---|
| | RR | | WRR | | RR | WRR | |
| | min | max | min | max | | | |
| 1 | 5.77/ 1.66 | 6.2/ 1.81 | 2.42/ 1.04 | 2.81/ 1.18 | 7.9 / 7.7 | 8.8 / 9.6 | Адекватно |
| 2 | 4.9/ 1.38 | 5.31/ 1.50 | 2.09/ 1.11 | 2.3/ 1.14 | 7.0 / 5.5 | 9.2 / 7.9 | Адекватно |
| 3 | 2.44/ 1.70 | 2.65/ 1.83 | 1.30/ 1.99 | 1.37/ 2.08 | 10.4 / 3.2 | 6.5 / 9.0 | Адекватно |
| 4 | 3.13/ 1.52 | 3.49/ 1.66 | 1.28/ 1.22 | 1.39/ 1.29 | 8.1 / 4.9 | 5.9 / 5.0 | Адекватно |
| 5 | 5.91/ 1.27 | 6.34/ 1.45 | 2.75/ 1.05 | 3.10/ 1.10 | 8.2 / 8.9 | 6.9 / 3.9 | Адекватно |

Для результатов, полученных для одной пары процессов в разных наблюдениях, были рассчитаны величины расхождения по формуле

$$\Delta = \left| \frac{w_i}{w_j} - 1 \right| \cdot 100\%, \qquad (3)$$

где $w_i, w_j; w_i \le w_j$ – средневзвешенное время обращения процессов одной пары разных наблюдений.

В таблице 2 представлены результаты имитационного эксперимента.

Допустимое отклонение определялось средней величиной расхождения средневзвешенного времени обращения одной пары процессов в разных наблюдениях. Таким образом, если расхождение между результатами моделирования и эксперимента с вычислительной сетью не выходит за рамки допустимой величины, то данный имитационный эксперимент характеризует модель как адекватную моделируемой системе.

Полученная погрешность определяется флуктуациями, возникающими вследствие работы служб операционных систем, а также из-за того, что задачи поступают в систему случайным образом. Построенную модель можно считать адекватной.

## Заключение

Была построена агентная модель открытой информационной системы и разработан программно-аппаратный комплекс для исследования алгоритмов балансировки нагрузки в гетерогенных информационных системах. Адекватность построенной модели была проверена на процессе расчета итеративных схем в вычислительной сети. Построенная модель и экспериментальная система позволяют исследовать загрузку гетерогенных информационных систем, используемых для поддержки вычислительных сервисов разных типов. Архитектура модели расширяема по алгоритмам балансировки нагрузки, а интерфейс с MySQL и системой WEKA позволяет автоматизировать аппроксимацию эмпирических данных мониторинга. Полученный результат может быть использован при исследовании масштабируемости открытых информационных систем любого уровня.

*Список литературы*

1. Kubicek H., Cimander R., Scholl H.J. Organizational Interoperability in E-Government. Hardcover, 2011. P. 185.

2. Aftab Iqbal, Ureche O., Hausenblas M., Tummarello G. LD2SD: Linked Data Driven Software Development // In: 21st International Conference on Software Engineering and Knowledge Engineering (SEKE 09), Boston. USA. 2009.

3. Батоврин В.К., Гуляев Ю.В., Олейников А.Я. Обеспечение интероперабельности – основная тенденция в развитии открытых систем // Информационные технологии и вычислительные системы. 2009. № 5. С. 7–15.

4. Хританков А.С. Модели и алгоритмы распределения нагрузки. Алгоритмы на основе сетей СМО // Информационные технологии и вычислительные системы. 2009. № 3. С. 33–48.

5. Хританков А.С. Модели и алгоритмы распределения нагрузки. Модель коллектива вычислителей. Модели с соперником // Информационные технологии и вычислительные системы. 2009. № 2. С. 65–80.

6. Ochin, Jugnu Gaur. Network monitoring system tools: an exploratory approach. Faculty of Engineering & Technology, Manav Rachna International University, http://www.seekdl.org/nm.php?id=1072.

7. Лоу А., Кельтон В. Имитационное моделирование. Питер, 2004. 848 с.

8. Ciraci S., Akyol B. An Evaluation of the Network Simulators in Large-Scale Distributed Simulation // HiPCNA-PG'11 2011. P. 59–65.

9. Weingartner E., vom Lehn H., Wehrle K. A performance comparison of recent network simulators // In: ICC'09, 2009. P. 1–5.

10. Коннов А.Л., Ушаков Ю.А. Методы расчета показателей производительности сетей ЭВМ с неоднородным трафиком: монография. Оренбургский гос. ун-т. Оренбург: ОГУ, 2013. 139 с.

11. Бугайченко Д.Ю., Соловьев И.П. Абстрактная архитектура интеллектуального агента и методы ее реализации. Системное программирование / Сб. ст. Вып. 1. Под ред. А.Н. Терехова, Д.Ю. Булычева. СПб.: Изд-во СПбГУ, 2005. С. 36–67.

12. Жевнерчук Д.В., Николаев А.В. Методика моделирования нагрузки на сервер в открытых системах облачных вычислений // Информатика и ее применения. 2012. Т. 6. Вып. 2. С. 99–106.

13. Zhang Lin, Li Xiao-ping. A content based dynamic load-balancing algorithm for heterogeneous Web server cluster // ComSIS. 2010. V. 7. № 1. Special Issue [электронный ресурс]: http://www.comsis.org/archive.php?show=pprncs-0012.

## MULTI-AGENT ENVIRONMENT TO STUDY LOAD BALANCING ALGORITHMS IN OPEN INFORMATION SYSTEMS

***D.V. Zhevnerchuk, V.L. Chepkasov***

The article considers the technique to study distributed service-oriented software-hardware systems having properties of open systems. The technique is based on multi-agent simulation with monitoring data admixture. The agent structure and behaviour are described in detail, along with the experimental system, which automates the processes of collecting resource utilization statistics, constructing theoretical dependences, and carrying out the simulation.

*Keywords*: multi-agent system, simulation, heterogeneous information system, load balancing, server, open system.

*References*

1. Kubicek H., Cimander R., Scholl H.J. Organizational Interoperability in E-Government. Hardcover, 2011. P. 185.

2. Aftab Iqbal, Ureche O., Hausenblas M., Tummarello G. LD2SD: Linked Data Driven Software Development // In: 21st International Conference on Software Engineering and Knowledge Engineering (SEKE 09), Boston. USA. 2009.

3. Batovrin V.K., Guljaev Ju.V., Olejnikov A.Ja. Obespechenie interoperabel'nosti – osnovnaja tendencija v razvitii otkrytyh sistem // Informacionnye tehnologii i vychislitel'nye sistemy. 2009. № 5. S. 7–15.

4. Hritankov A.S. Modeli i algoritmy raspredelenija nagruzki. Algoritmy na osnove setej SMO // Informacionnye tehnologii i vychislitel'nye sistemy. 2009. № 3. C. 33–48.

5. Hritankov A.S. Modeli i algoritmy raspredelenija nagruzki. Model' kollektiva vychislitelej. Modeli s sopernikom // Informacionnye tehnologii i vychislitel'nye sistemy. 2009. № 2. S. 65–80.

6. Ochin, Jugnu Gaur. Network monitoring system tools: an exploratory approach. Faculty of Engineering & Technology, Manav Rachna International University, http://www.seekdl.org/nm.php?id=1072.

7. Lou A., Kel'ton V. Imitacionnoe modelirovanie. Piter, 2004. 848 s.

8. Ciraci S., Akyol B. An Evaluation of the Network Simulators in Large-Scale Distributed Simulation // HiPC-NA-PG'11 2011. R. 59–65.

9. Weingartner E., vom Lehn H., Wehrle K. A performance comparison of recent network simulators // In: ICC'09, 2009. R. 1–5.

10. Konnov A.L., Ushakov Ju.A. Metody rascheta pokazatelej proizvoditel'nosti setej JeVM s neodnorodnym trafikom: monografija. Orenburgskij gos. un-t. Orenburg: OGU, 2013. 139 s.

11. Bugajchenko D.Ju., Solov'ev I.P. Abstraktnaja arhitektura intellektual'nogo agenta i metody ee realizacii. Sistemnoe programmirovanie / Sb. st. Vyp. 1. Pod red. A.N. Terehova, D.Ju. Bulycheva. SPb.: Izd-vo SPbGU, 2005. S. 36–67.

12. Zhevnerchuk D.V., Nikolaev A.V. Metodika modelirovanija nagruzki na server v otkrytyh sistemah oblachnyh vychislenij // Informatika i ee primenenija. 2012. T. 6. Vyp. 2. S. 99–106.

13. Zhang Lin, Li Xiao-ping. A content based dynamic load-balancing algorithm for heterogeneous Web server cluster // ComSIS. 2010. V. 7. № 1. Special Issue [электронный ресурс]: http://www.comsis.org/archive.php?show=pprncs-0012.