УДК 004.738.5:519.682:519.872

Тестирование клиента и сервера для выбора объекта проведения экспериментов в инструментальном программном средстве имитационного моделирования на основе технологии «облачных» вычислений

© А.Ю. Быков, Н.В. Медведев, Ф.А. Панфилов

МГТУ им. Н.Э. Баумана, Москва, 105005, Россия

Рассмотрено средство имитационного моделирования, выполненное в виде Web-приложения на основе технологии «облачных» вычислений. Web-приложение разработано с использованием технологии Java-сервлетов и специальной библиотеки классов языка Java для имитационного моделирования. Приложение позволяет выбирать объект для проведения имитационных экспериментов – клиент или сервер. Рассмотрены тесты для выбора объекта, на котором необходимо проводить эксперименты, с целью уменьшения длительности цикла моделирования. Представлены результаты тестов в различных условиях.

Ключевые слова: *имитационное моделирование, «облачные» вычисления, распределение заданий, клиент, сервер, балансировка нагрузки.*

Введение. В технологии распределенных вычислений одной из решаемых задач является задача распределения заданий между элементами распределенной вычислительной системы. Эту задачу также называют задачей балансировки нагрузки. При использовании технологии «облачных» вычислений также может решаться задача распределения заданий между элементами системы, например, между клиентом и сервером, если вычислительные и программные возможности клиента позволяют решать некоторые задания.

В «облачных» вычислениях считается, что вычислительное средство клиента работает, используя из прикладного программного обеспечения только Web-браузер (программа для просмотра Web-страниц) со стандартными функциями. Большинство Web-браузеров, за исключением браузеров для мобильных устройств, позволяет выполнять на стороне клиента некоторые специальные Web-приложения: Java-апплеты, элементы управления ActiveX.

Ниже будет рассмотрено Web-приложение, основанное на технологии «облачных» вычислений, в котором можно осуществлять выбор различных технологий моделирования, отличающихся местами (клиент или сервер) выполнения заданий одного цикла моделирования. В качестве основного показателя выбора используется оценка времени выполнения всего комплекса заданий. В Web-приложении используется библиотека классов языка Java [1], разработанная для имитационного моделирования систем массового обслуживания

(СМО), библиотека классов является развитием основных подходов представленных в [2, 3].

Рассмотрим задачи (при выполнении задачи на ЭВМ будем использовать термин «задание»), решаемые при имитационном моделировании (ИМ), если использовать библиотеку классов языка Java [1]. При этом могут решаться последовательно следующие основные задачи, составляющие один цикл моделирования и решаются последовательно:

1. ввод исходного кода реализации имитационной модели на языке Java;
2. компиляция исходного кода с исправлением ошибок;
3. проведение имитационных экспериментов;
4. вывод и представление результатов имитационного моделирования.

Проведем анализ того, где могут решаться (на стороне клиента или сервера) перечисленные задачи.

Исходя из особенностей функционирования «облачных» средств на стороне клиента через Web-браузер обязательно должны решаться задачи с номерами: 1, 4, т. е. это задачи обеспечения интерфейса пользователя: ввод пользователем данных и просмотр результатов.

На стороне Web-сервера обязательно должна решаться задача 2 (компиляция), так как на стороне клиента из программного обеспечения установлен Web-браузер со стандартными функциями.

Задача 3 может выполняться как на стороне сервера, так и на стороне клиента (если возможности клиента позволяют). На стороне сервера, как правило, существуют достаточно мощные вычислительные ресурсы, а на стороне клиента возможно выполнение через Web-браузер специальных приложений (ActiveX, Java-апплеты), позволяющих решать подобные вычислительные задачи с некоторыми ограничениями. Таким образом, появляется возможность гибко распределять вычислительные ресурсы между решаемыми заданиями в зависимости от текущих условий: загрузки сервера, быстродействия клиента, скорости передачи данных по сети и т. д.

На рис. 1 перечисленная последовательность задач представлена в виде графа, на котором обозначено символами: K – задача должна выполняться только на клиентском компьютере; S – задача должна выполняться только на сервере; X – задача может выполняться и на клиентском компьютере, и на сервере. Задачи последнего класса могут распределяться между клиентом и сервером, т. е. возникает задача выбора объекта для проведения имитационных экспериментов.

При этом будем учитывать, что задача 1 – ввод исходного кода модели выполняется на стороне клиента, и время ее выполнения определяется человеком, т. е. его квалификацией, скоростью работы

и т. п.; при вычислении длительности выполнения всего цикла задач моделирования это время учитывать не будем.

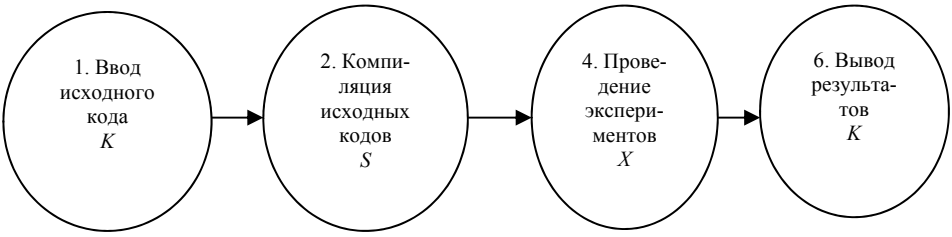


Рис. 1. Граф задач, решаемых при имитационном моделировании

Описание Web-приложения. Для поддержки ИМ через Интернет реализовано Web-приложение, разработанное в среде Eclipse, серверная часть которого основана на Java-сервлетах. В Web-приложении используется библиотека классов языка Java для поддержки ИМ [1]. Для разворачивания приложения на Web-сервере необходимо вначале установить специальную программу – сервер приложений Tomcat (контейнер сервлетов), разработанную корпорацией Apache Software Foundation. Web-приложение реализует следующие основные функции:

1. Проведение имитационных экспериментов на стороне сервера:

– ввод исходного кода модели на языке, похожем на GPSS [4], в текстовую область формы HTML-страницы, передача кода модели на сервер, запуск на сервере интерпретатора, представленного специальным классом Java, интерпретатор проводит эксперименты, результаты экспериментов передаются клиенту в виде HTML-страницы;

– ввод исходного кода модели на языке Java в текстовую область формы HTML-страницы, передача кода модели на сервер, компиляция кода модели на сервере компилятором Java, при наличии ошибок выдача сообщений о них клиенту в виде HTML-страницы, при отсутствии ошибок запуск на сервере интерпретатора Java, который проводит эксперименты, результаты экспериментов передаются клиенту в виде HTML-страницы.

2. Проведение имитационных экспериментов на стороне клиента:

– загрузка Web-страницы с апплетом и интерпретатором языка, похожего на GPSS, представленным специальным классом Java, ввод исходного кода модели на языке, похожем на GPSS, в текстовую область апплета, запуск интерпретатора, который проводит эксперименты, результаты экспериментов отображаются клиенту в отдельном окне;

– ввод исходных кодов модели и апплета как контейнера для вывода результатов на языке Java в текстовую область формы HTML-страницы, передача кодов на сервер, компиляция кодов на сервере

компилятором Java, при наличии ошибок выдача сообщений о них клиенту в виде HTML-страницы, при отсутствии ошибок загрузка клиенту апплета с исполняемым кодом модели, проведение экспериментов на стороне клиента, результаты экспериментов отображаются в отдельном окне клиента.

3. Тестирование сервера и клиента с целью выбора объекта, на котором будет проводиться моделирование.

Рассмотрим процесс тестирования клиента и сервера для выбора объекта моделирования более подробно, при этом модели будем разрабатывать на языке Java с использованием библиотеки классов. При тестировании будем оценивать суммарное время выполнения комплекса заданий цикла моделирования с учетом времени выполнения заданий и времени передачи данных и исполняемых модулей по сети. Оцениваться будет среднее время по нескольким тестам.

При выполнении экспериментов на сервере общее время решения заданий цикла моделирования определяется суммой следующих элементов:

– время загрузки исходного кода модели на сервер (определяется объемом кода и скоростью передачи данных по сети);

– время компиляции исходного кода (определяется быстродействием сервера и объемом исходного кода);

– время выполнения экспериментов с моделью (определяется быстродействием сервера и объемом экспериментов, в модели объем экспериментов определяется числом обрабатываемых заявок);

– время передачи результатов клиенту (определяется объемом передаваемых данных и скоростью передачи).

При выполнении экспериментов на стороне клиента общее время решения заданий цикла моделирования определяется суммой следующих элементов:

– время загрузки исходного кода модели и апплета на сервер;

– время компиляции исходного кода на сервере (определяется быстродействием компьютера сервера, объемом исходного кода);

– время загрузки HTML-страницы с апплетом и библиотекой классов для ИМ на клиентский компьютер (определяется объемом кода страницы, апплета, библиотеки классов для ИМ и скоростью передачи данных по сети);

– время запуска апплета в браузере и время проведения экспериментов с моделью (определяется быстродействием компьютера клиента и объемом экспериментов, в модели объем экспериментов определяется числом обрабатываемых заявок).

При тестировании сервера реализуется следующий алгоритм:

1. Пользователь открывает HTML-страницу с формой, в текстовой области которой находится исходный код модели на языке Java (имитация, как будто пользователь его ввел сам), пользователь должен нажать кнопку на форме «Начать тестирование».

2. Код модели передается на сервер, объем переданных данных запоминается.

3. На сервере проводится компиляция исходного кода (в коде тестовой модели ошибки отсутствуют), время компиляции запоминается.

4. После компиляции на сервере запускается интерпретатор Java, и выполняются эксперименты, время проведения экспериментов запоминается.

5. Сервер формирует HTML-страницу с результатами экспериментов, она загружается клиенту, объем переданных данных клиенту запоминается, на эту страницу также выводятся результаты полученных тестов.

При тестировании клиента реализуется следующий алгоритм:

1. Пользователь открывает HTML-страницу с формой, в текстовой области которой находится исходный код тестовой модели на языке Java, оформленный в виде апплета, пользователь должен нажать кнопку на форме «Начать тестирование».

2. Код модели передается на сервер.

3. На сервере проводится компиляция исходного кода апплета (в коде тестовой модели ошибки отсутствуют), время компиляции запоминается. Формируется HTML-страница с апплетом, которая отправляется клиенту.

4. После загрузки HTML-страницы на клиентский компьютер апплет сразу стартует в браузере и выполняются эксперименты с моделью.

5. После завершения экспериментов на сервер отправляется время выполнения экспериментов.

6. Получив сигнал от апплета о завершении экспериментов, сервер отправляет апплету тестовый массив данных для тестирования скорости приема данных от сервера, объем переданных данных запоминается.

7. Получив тестовый массив данных, апплет определяет время приема данных (по времени, прошедшему после отправки предыдущих данных серверу), отправляет серверу это время вместе с тестовым массивом данных для расчета скорости отдачи данных для клиента.

8. Сервер, получив данные, рассчитывает скорость приема данных для клиента и посылает сигнал клиенту (в качестве сигнала отправляется 1 байт данных).

9. Получив сигнал от сервера, апплет определяет время отдачи данных (по времени, прошедшему после отправки предыдущих данных серверу), отправляет серверу это время, после этого апплет завершает работу.

10. Сервер рассчитывает скорость отдачи данных для клиента.

11. Сервер на HTML-страницу с апплетом дополнительно выводит результаты тестирования.

Тестировать клиента и сервер можно в любом порядке, но следует отметить, что скорость каналов передачи данных по сети тестируется при тестировании клиента.

Описание тестовой задачи. Будем проводить эксперименты с типовой СМО – одноканальной СМО с неограниченной очередью, моделируется прохождение через систему 10 000 000 заявок (число выбрано экспериментально с учетом быстродействия вычислительных средств, чтобы продолжительность экспериментов с моделью измерялась в секундах).

Параметры СМО: пуассоновский входящий поток заявок, среднее время между заявками 10 секунд ($\lambda = 0{,}1$ заявки с); время обслуживания заявки распределено по показательному закону, среднее время обслуживания 8 секунд ($\mu = 1/8$ заявки с); если заявка приходит в момент времени, когда канал занят, то она становится в очередь, длина очереди неограниченна. Провести имитационное моделирование СМО, определить параметры СМО: параметры очереди задач (среднюю длину очереди и среднее время ожидания заявки в очереди), загрузку канала обслуживания.

Для контроля полученных результатов задача имеет аналитическое решение [5]. Основные соотношения имеют следующий вид:

$p_0 = 1 - \rho$ – вероятность того, что в системе нет заявок,

$\bar{Q}_{очер}$ $\frac{2}{1}$ – средняя длина очереди,

$\bar{T}_{очер}$ $\frac{2}{1}$ – среднее время ожидания в очереди,

$K_{загр} = 1 - p_0 = \rho$ – коэффициент загрузки канала обслуживания.

Подставив заданные значения, получаем результаты:

$\bar{Q}_{очер} = 3{,}2$, $\bar{T}_{очер} = 32$ с, $K_{загр} = 0{,}8$.

Исходный код реализации этой системы на языке Java для выполнения на сервере в виде приложения представлен в листинге 1.

Листинг 1

Исходный код реализации модели

```
import SimJava.*;
import java.io.*;
import java.awt.*;
public class TestModel  extends Syst implements Runnable {
    Facility lF; // Ссылка на канал обслуживания
    Queue lQ; // Ссылка на очередь
    public TestModel(int tg1) // Конструктор класса
    {
```

```
        super(tg1, // Значение счетчика завершения
                null, // Ссылка на файл с результатами
                null); // Ссылка на контейнер для вывода данных
        th=new Thread(this); // Создание потока запуска модели
        // Создаем поток вывода для HTML страницы
        out=new PrintWriter(System.out, true);
        // Создание модельной среды
        lQ=new Queue(this, "Очередь"); // Создаем очередь
        lF=new Facility(this, "Администратор"); // Создаем прибор

    }
    public void run()
    {
        initGenerate(1, 0); // Транзакт заявка направляется к перво-
му событию

        while(tg1!=0) // Конец моделирования, когда TG1
                            // (счетчик завершения равен 0)
        {
            plan();  // Протяжка модельного времени
            switch(sysEvent)
            {
            case 1:
                generate(v1.randExp(0.1, false)); break;
                                                // Генерация заявок
            case 2:
                lQ.queue(1); break; // Занимаем очередь
            case 3:
                lF.seize(); break; // Занимаем канал обслуживания
            case 4:
                lQ.depart(1); break; // Освобождаем очередь
            case 5:
                advance(v12.randExp(1./8, false)); break;
                                // Задержка - время обслуживания
            case 6:
                lF.release(); break; // Освобождаем канал обслуживания
            case 7:
                terminate(1);
                      // Транзакт уничтожается, счетчик завершения
уменьшается на 1
            }
        }
        printAllHTML(); // Вывод результатов моделирования в HTML
страницу
    }
    public static void main(String args [])
    {
        TestModel model=new TestModel(10000000);
        model.th.start();
        try { model.th.join(); } catch(Exception e) {}
    }
}
```

Тестирование проводилось в локальной сети на основе технологии Wi-Fi, сервер и клиент являлись компьютерами с разными техническими характеристиками, в разных тестах клиент и сервер менялись местами.

HTML-страница с результатами тестирования сервера в случае, когда технические возможности компьютера-сервера превосходят возможности компьютера-клиента (сервер-процессор Intel ® Core ™ i5-2410, тактовая частота 2,3 ГГц; клиент-процессор Inter ® Celeron ® CPU 1.80 ГГц), каналы сети находились в нагруженном состоянии (на счет передачи файлов большого размера по сети в это же время), представлены на рис. 2. На этом рисунке в представленной таблице с результатами тестов отсутствуют данные по времени передачи по сети, так как сеть тестируется при выполнении экспериментов на стороне клиента.

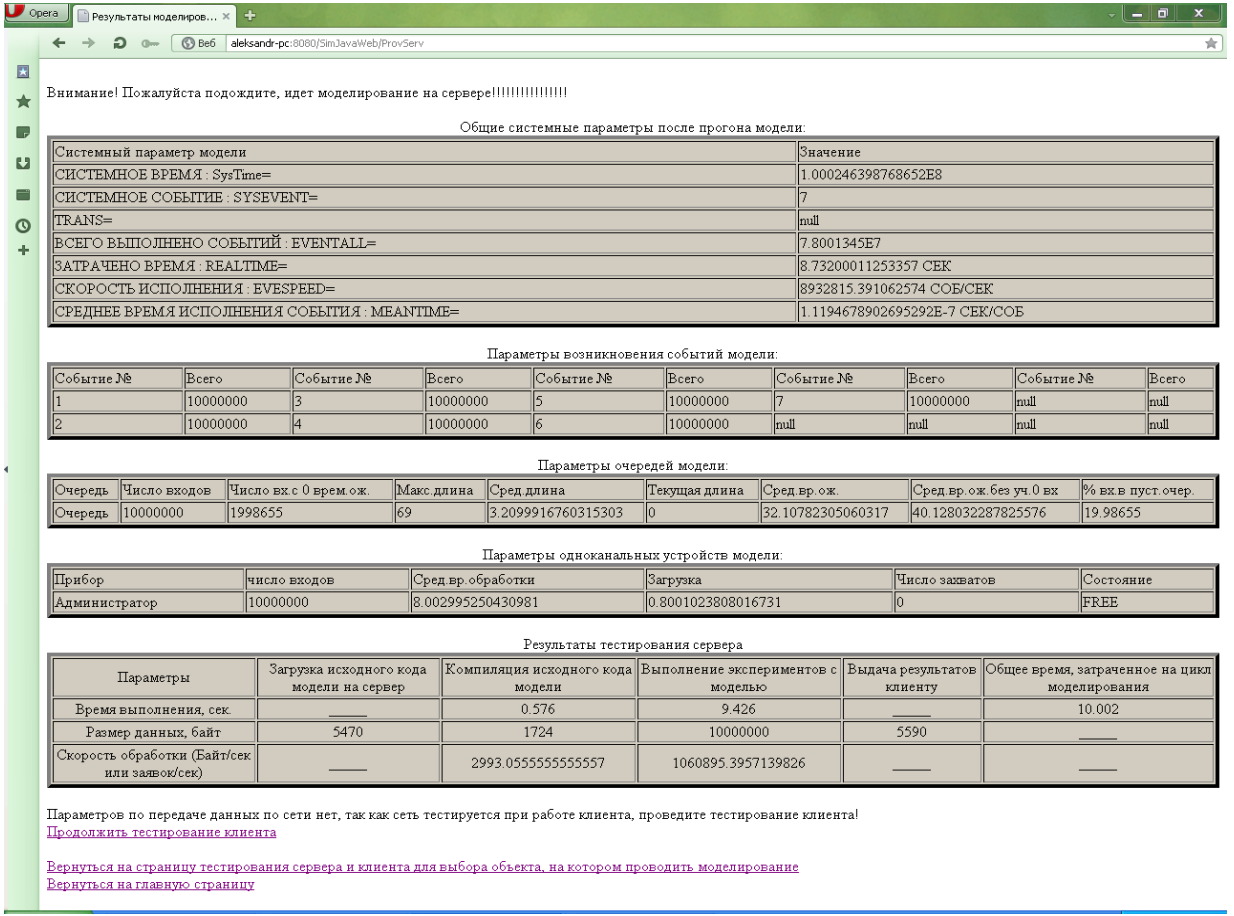
HTML-страница с результатами тестирования клиента и итоговыми результатами представлена на рис. 3, на этой же странице представлены окончательные данные тестов сервера, включая время передачи данных по сети. При этом общее время выполнения заданий цикла моделирования составило: при выполнении экспериментов на сервере ≈10 с, при выполнении экспериментов на клиентском компьютере ≈ 71 с.

HTML-страница с итоговыми результатами тестирования клиента и сервера в случае, когда технические возможности компьютера-клиента превосходят возможности компьютера-сервера (сервер-Inter ® Celeron ® CPU 1.80 ГГц; клиент-процессор Intel ® Core ™ i5-2410, тактовая частота 2,3 ГГц), каналы сети находятся в относительно свободном состоянии, представлены на рис. 4. При этом общее время выполнения заданий цикла моделирования составило: при выполнении экспериментов на сервере ≈52 с, при выполнении экспериментов на клиентском компьютере ≈17 с.

Как видно из представленных материалов, результаты моделирования СМО соответствуют результатам, полученным из аналитических расчетов, с небольшой погрешностью.

Результаты тестов приведены в таблицах, в которых представлены времена выполнения операций, объемы выполняемых операций (объемы данных, число обрабатываемых заявок в модели) и скорости проведения операций.

Представленные результаты тестов показали, что основным параметром, от которого зависит общее время проведения цикла моделирования, является время проведения экспериментов, если размер передаваемых данных по сети не является большим и скорость передачи достаточно высока. В этом случае основным фактором, влияющим на выбор объекта моделирования, является реальное, с учетом



Внимание! Пожалуйста подождите, идет моделирование на сервере!!!!!!!!!!!!!!!

Общие системные параметры после прогона модели:

Системный параметр модели	Значение
СИСТЕМНОЕ ВРЕМЯ : SysTime=	1.000246398768652E8
СИСТЕМНОЕ СОБЫТИЕ : SYSEVENT=	7
TRANS=	null
ВСЕГО ВЫПОЛНЕНО СОБЫТИЙ : EVENTALL=	7.8001345E7
ЗАТРАЧЕНО ВРЕМЯ : REALTIME=	8.73200011253357 СЕК
СКОРОСТЬ ИСПОЛНЕНИЯ : EVESPEED=	8932815.391062574 СОБ/СЕК
СРЕДНЕЕ ВРЕМЯ ИСПОЛНЕНИЯ СОБЫТИЯ : MEANTIME=	1.1194678902695292E-7 СЕК/СОБ

Параметры возникновения событий модели:

Событие №	Всего	Событие №	Всего	Событие №	Всего	Событие №	Всего	Событие №	Всего
1	10000000	3	10000000	5	10000000	7	10000000	null	null
2	10000000	4	10000000	6	10000000	null	null	null	null

Параметры очередей модели:

Очередь	Число входов	Число вх.с 0 врем.ож.	Макс.длина	Сред.длина	Текущая длина	Сред.вр.ож.	Сред.вр.ож.без уч.0 вх	% вх.в пуст.очер.
Очередь	10000000	1998655	69	3.2099916760315303	0	32.10782305060317	40.128032287825576	19.98655

Параметры одноканальных устройств модели:

Прибор	число входов	Сред.вр.обработки	Загрузка	Число захватов	Состояние
Администратор	10000000	8.002995250430981	0.8001023808016731	0	FREE

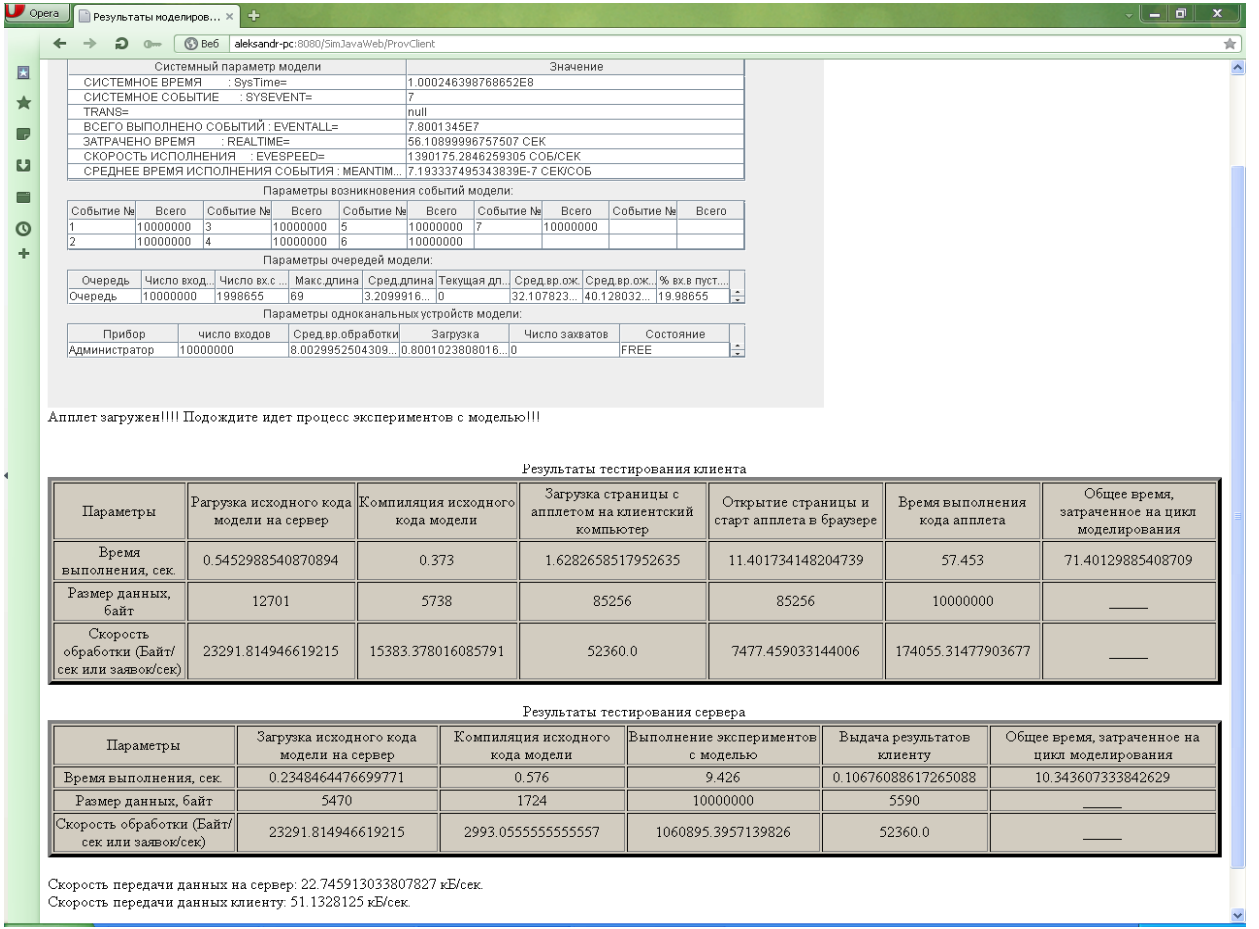
Результаты тестирования сервера

Параметры	Загрузка исходного кода модели на сервер	Компиляция исходного кода модели	Выполнение экспериментов с моделью	Выдача результатов клиенту	Общее время, затраченное на цикл моделирования
Время выполнения, сек.	_____	0.576	9.426	_____	10.002
Размер данных, байт	5470	1724	10000000	5590	_____
Скорость обработки (Байт/сек или заявок/сек)	_____	2993.0555555555557	1060895.3957139826	_____	_____

Параметров по передаче данных по сети нет, так как сеть тестируется при работе клиента, проведите тестирование клиента!
Продолжить тестирование клиента

Вернуться на страницу тестирования сервера и клиента для выбора объекта, на котором проводить моделирование
Вернуться на главную страницу

Рис. 2. Результаты тестирования сервера при производительности сервера, превышающей производительность клиента



Системный параметр модели	Значение
СИСТЕМНОЕ ВРЕМЯ : SysTime=	1.000246398768652E8
СИСТЕМНОЕ СОБЫТИЕ : SYSEVENT=	7
TRANS=	null
ВСЕГО ВЫПОЛНЕНО СОБЫТИЙ : EVENTALL=	7.8001345E7
ЗАТРАЧЕНО ВРЕМЯ : REALTIME=	56.10899996757507 СЕК
СКОРОСТЬ ИСПОЛНЕНИЯ : EVESPEED=	1390175.2846259305 СОБ/СЕК
СРЕДНЕЕ ВРЕМЯ ИСПОЛНЕНИЯ СОБЫТИЯ : MEANTIM...	7.193337495343839E-7 СЕК/СОБ

Параметры возникновения событий модели:

Событие №	Всего	Событие №	Всего	Событие №	Всего	Событие №	Всего	Событие №	Всего
1	10000000	3	10000000	5	10000000	7	10000000		
2	10000000	4	10000000	6	10000000				

Параметры очередей модели:

Очередь	Число вход...	Число вх.с ...	Макс.длина	Сред.длина	Текущая дл...	Сред.вр.ож.	Сред.вр.ож...	% вх.в пуст....
Очередь	10000000	1998655	69	3.2099916...	0	32.107823...	40.128032...	19.98655

Параметры одноканальных устройств модели:

Прибор	число входов	Сред.вр.обработки	Загрузка	Число захватов	Состояние
Администратор	10000000	8.0029952504309...	0.8001023808016...	0	FREE

Апплет загружен!!!! Подождите идет процесс экспериментов с моделью!!!

Результаты тестирования клиента

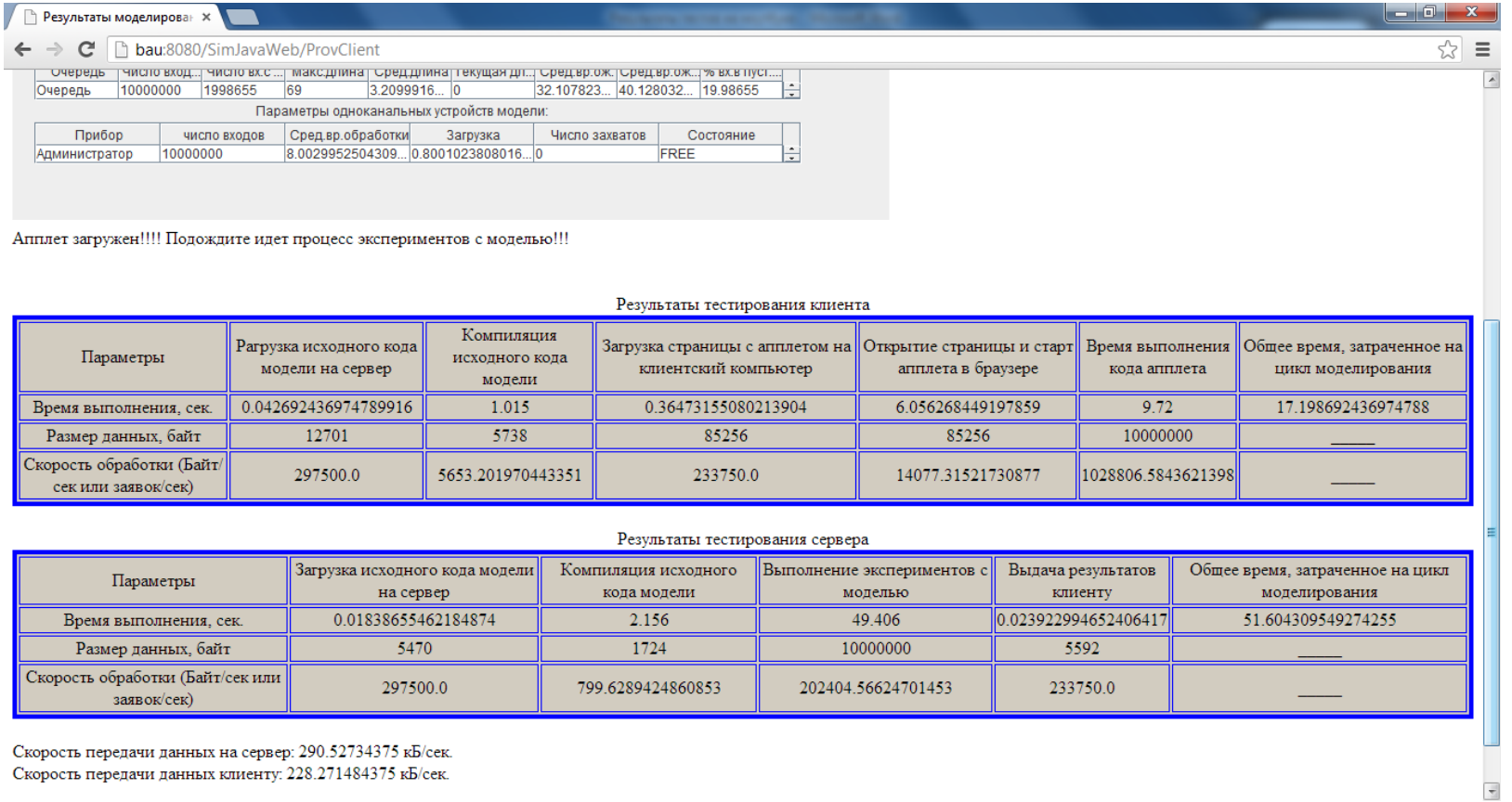
Параметры	Рагрузка исходного кода модели на сервер	Компиляция исходного кода модели	Загрузка страницы с апплетом на клиентский компьютер	Открытие страницы и старт апплета в браузере	Время выполнения кода апплета	Общее время, затраченное на цикл моделирования
Время выполнения, сек.	0.5452988540870894	0.373	1.6282658517952635	11.401734148204739	57.453	71.40129885408709
Размер данных, байт	12701	5738	85256	85256	10000000	_____
Скорость обработки (Байт/сек или заявок/сек)	23291.814946619215	15383.378016085791	52360.0	7477.459033144006	174055.31477903677	_____

Результаты тестирования сервера

Параметры	Загрузка исходного кода модели на сервер	Компиляция исходного кода модели	Выполнение экспериментов с моделью	Выдача результатов клиенту	Общее время, затраченное на цикл моделирования
Время выполнения, сек.	0.2348464476699771	0.576	9.426	0.10676088617265088	10.343607333842629
Размер данных, байт	5470	1724	10000000	5590	_____
Скорость обработки (Байт/сек или заявок/сек)	23291.814946619215	2993.0555555555557	1060895.3957139826	52360.0	_____

Скорость передачи данных на сервер: 22.745913033807827 кБ/сек.
Скорость передачи данных клиенту: 51.1328125 кБ/сек.

Рис. 3. Результаты тестирования клиента при производительности сервера, превышающей производительность клиента



Результаты моделирова

bau:8080/SimJavaWeb/ProvClient

Очередь	Число вход...	Число вх.с ...	Макс.длина	Сред.длина	Текущая дл...	Сред.вр.ож.	Сред.вр.ож...	% вх.в пуст....
Очередь	10000000	1998655	69	3.2099916...	0	32.107823...	40.128032...	19.98655

Параметры одноканальных устройств модели:

Прибор	число входов	Сред.вр.обработки	Загрузка	Число захватов	Состояние
Администратор	10000000	8.0029952504309...	0.8001023808016...	0	FREE

Апплет загружен!!!! Подождите идет процесс экспериментов с моделью!!!

Результаты тестирования клиента

Параметры	Рагрузка исходного кода модели на сервер	Компиляция исходного кода модели	Загрузка страницы с апплетом на клиентский компьютер	Открытие страницы и старт апплета в браузере	Время выполнения кода апплета	Общее время, затраченное на цикл моделирования
Время выполнения, сек.	0.042692436974789916	1.015	0.36473155080213904	6.056268449197859	9.72	17.198692436974788
Размер данных, байт	12701	5738	85256	85256	10000000	_____
Скорость обработки (Байт/сек или заявок/сек)	297500.0	5653.201970443351	233750.0	14077.31521730877	1028806.5843621398	_____

Результаты тестирования сервера

Параметры	Загрузка исходного кода модели на сервер	Компиляция исходного кода модели	Выполнение экспериментов с моделью	Выдача результатов клиенту	Общее время, затраченное на цикл моделирования
Время выполнения, сек.	0.01838655462184874	2.156	49.406	0.023922994652406417	51.604309549274255
Размер данных, байт	5470	1724	10000000	5592	_____
Скорость обработки (Байт/сек или заявок/сек)	297500.0	799.6289424860853	202404.56624701453	233750.0	_____

Скорость передачи данных на сервер: 290.52734375 кБ/сек.
Скорость передачи данных клиенту: 228.271484375 кБ/сек.

Рис. 4. Результаты тестов клиента и сервера при производительности клиента, превышающей производительность сервера

загрузки решением других фоновых задач, быстродействие компьютера объекта. Предпочтительнее проводить моделирование на объекте с большим быстродействием, но следует учитывать, что при выполнении экспериментов на клиентском компьютере некоторое время занимает открытие страницы с апплетом и старт апплета в браузере, за счет этого времени общее время, затраченное на решение задач цикла моделирования, будет немного больше. Также при низкой скорости передачи данных по сети, на общее время цикла моделирования будет влиять объем передаваемых данных, как правило, при выполнении экспериментов на стороне клиента, объем передаваемых данных, больше из-за необходимости загрузки исполняемых кодов библиотеки классов языка Java для ИМ.

Заключение. Разработанное Web-приложение позволяет проводить эксперименты с реализациями моделей систем как на стороне клиента, так и на стороне сервера. Существует возможность тестирования клиента и сервера с целью выбора объекта для проведения имитационных экспериментов с реализациями моделей с учетом реального быстродействия сервера, клиента, а также реальной скорости передачи данных по сети для снижения времени выполнения заданий цикла моделирования.

Исследование выполнено при поддержке Министерства образования и науки Российской Федерации (соглашение № 14.B37.21.0401).

ЛИТЕРАТУРА

[1] Быков А.Ю., Панфилов Ф.А., Сумарокова О.О. Имитационное моделирование с применением библиотеки классов языка Java, разработанной для «облачных» сервисов. *Инженерный журнал: наука и инновации*. Электронное научно-техническое издание (Вестник МГТУ им. Н.Э. Баумана: электронное издание), 2013, вып. 2. URL: http://www.engjournal.ru/catalog/it/hidden/535.html.

[2] Медведев Н.В., Быков А.Ю., Гришин Г.А. Имитационное моделирование систем массового обслуживания с использованием межплатформенной библиотеки функций языка Си++. *Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение*, 2005, № 4, с. 85–93.

[3] Журавлев А.М., Медведев Н.В., Быков А.Ю. Использование межплатформенной библиотеки функций языка СИ++ для реализации имитационных моделей типовых систем массового обслуживания. *Вестник МГТУ им. Н.Э. Баумана. сер. Приборостроение*, 2008, № 4, с. 3–15.

[4] Шрайбер Т.Дж. *Моделирование на GPSS*. Москва, Машиностроение, 1980, 592 с.

[5] Клейнрок Л. *Теория массового обслуживания*. Москва, Машиностроение, 1979, 432 с.

Статья поступила в редакцию 28.06.2013

Ссылку на эту статью просим оформлять следующим образом:

Быков А.Ю., Медведев Н.В., Панфилов Ф.А. Тестирование клиента и сервера для выбора объекта проведения экспериментов в инструментальном программном средстве имитационного моделирования на основе технологии «облачных» вычислений. *Инженерный журнал: наука и инновации*, 2013, вып. 11. URL: http://engjournal.ru/catalog/it/network/989.html

Быков Александр Юрьевич родился в 1969 г., окончил ВИКИ им. А.Ф. Можайского в 1991 г. Канд. техн. наук, доцент кафедры «Информационная безопасность» МГТУ им. Н.Э. Баумана. Автор около 25 работ в области информационной безопасности и исследования систем обработки информации и управления. e-mail: abykov@bmstu.ru

Медведев Николай Викторович родился в 1954 г., окончил МВТУ им. Н.Э. Баумана в 1977 г. Канд. техн. наук, доцент кафедры «Информационная безопасность» МГТУ им. Н.Э. Баумана. Автор около 50 научных работ в области информационной безопасности и исследования систем обработки информации и управления.

Панфилов Филипп Александрович родился в 1989 г. Ассистент кафедры «Информационная безопасность» МГТУ им. Н.Э. Баумана.