Инженерный вестник Дона, №9 (2021)
ivdon.ru/ru/magazine/archive/n9y2021/7172

Имитационные модели функционирования распределенных систем обработки информации и их программная реализация

В. К. Михайлов, А. Н. Скоба, В.В. Бадашев, Д.В. Шахов, А.В. Реков, А.Л. Можаев.
Южно-Российский государственный политехнический университет (НПИ) им. М. И. Платова, Новочеркасск

Аннотация: В работе описаны основные агенты с их компонентами и диаграммами процесса, разработанные имитационные модели функционирования распределенных систем обработки информации. Описан метод конструирования моделей с самоподобным трафиком заявок. В статье также представлен способ получения реальных (эталонных) показателей среднего времени реакции системы на запросы пользователей. Представлены результаты численных экспериментов над созданными моделями функционирования распределённых систем обработки информации с самоподобным трафиком заявок. Сделаны основные выводы о проделанных экспериментах.
Ключевые слова: распределенная система обработки информации, самоподобный поток, среднее время реакции системы, имитационное моделирование, агентное моделирование, распределение Вейбулла.

Высокие темпы развития вычислительной техники, а также средств связи ускорили процессы автоматизации промышленных предприятий, учреждений народного хозяйства, социальной сферы, образования и обусловили тенденцию перехода от централизованных структур построения систем обработки информации (СОИ), в которых ограниченное количество пользовательских терминалов и абонентских пунктов индивидуально подключается к центральной ЭВМ, к распределённым СОИ. Распределённая СОИ представляет собой децентрализованную систему, которая объединяет как на физическом, так и на информационном уровне локальные независимые информационные подсистемы, имеющие одинаковый статус и активно взаимодействующие друг с другом. В соответствии с этим возникает задача нахождения интегральных характеристик их функционирования:

- среднее время реакции (СВР) системы, как на конкретную группу запросов пользователей, так и на все запросы к системе;
- средняя дисперсия времени реакции (СДВР) системы;
- функциональная надежность;

- среднее время выполнения операций в сети;

- общие эксплуатационные затраты (хранение и обработка информации).

При проектировании банковских и биллинговых систем; систем обработки и анализа данных; эргатических систем и т.п. [1], решение задачи нахождения интегральных характеристик функционирования таких систем является актуальной. Однако, как показал проведенный анализ современной литературы [2-4], в основе разработанных ранее математических моделей функционирования распределённых систем обработки информации лежало предположение о существовании марковского случайного процесса, описывающего вероятности формирования заявок пользователями и распределение интервалов времени поступления (обслуживания) таких заявок в узлах СОИ. Однако детальные исследования показали, что входные потоки современных распределённых систем обработки информации обладают свойствами самоподобия (самоподобные потоки), которые оказывают существенное влияние на их вероятностно-временные характеристики [5,6].

Как было отмечено в работе [6], в настоящее время, при конструировании концептуальных и математических моделей функционирования, распределённых СОИ с самоподобными потоками, широко используется аппарат имитационного моделирования. Это прежде всего обусловлено тем, что из-за отсутствия серьезной и проверенной теоретической базы, которая бы смогла дополнить традиционную теорию распределённых вычислительных систем, в данный момент не существует другой методики расчета характеристик и показателей качества функционирования распределённых систем обработки информации, кроме как методы имитационного моделирования [7]. Важно отметить, что

методика должна быть достоверна и учитывать условия эффекта самоподобия трафика сети.

Однако, для моделирования самого фрактального случайного процесса, имитирующего поток заявок на входе распределённой системы обработки информации может быть использовано: логнормальное распределение [5], распределение Парето, применение которого для моделирования случайного процесса, с признаками самоподобия, имитирующего поток заявок на входе распределённой системы обработки информации, было подробно рассмотрено в работе [8]; распределения Вейбулла [3].

Для программной реализации имитационных моделей была использована платформа разработки *AnyLogic* [9], которая охватывает основные направления современного имитационного моделирования: системную динамику, агентное моделирование.

При разработке моделей были созданы следующие агенты:

Агент *Main* (рисунок 1) отвечает за конструирование исходных данных для моделирования.

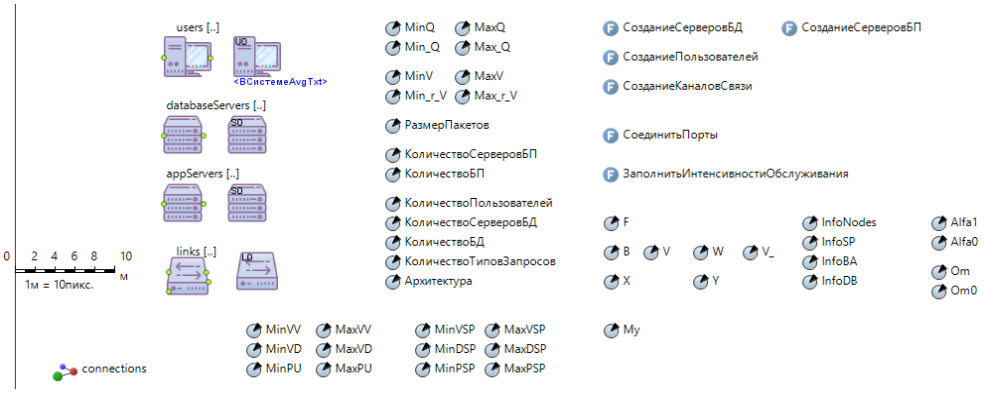

Рисунок 1 – Основные компоненты агента *Main*

На рисунке 1: *«users[]»* – коллекция экземпляров объектов агента *User*; *«databaseServers[]»* – коллекция экземпляров объектов агента *Server*, с параметром «ЭтоСБП» равным «Ложь»; *«appServers[]»* – коллекция экземпляров объектов агента *Server*, с параметром «ЭтоСБП» равным

«Истина»; *«links[]»* – коллекция экземпляров объектов агента *Link*; *«F»* – матрица вероятностей формирования запросов пользователями; *«B»* – матрица обращений типов запросов или бизнес-приложений к таблицам БД или файлам; *«V»* – матрица объемов информации, элемент которой представляет собой объем информации по *SQL*-запросу, сформированному бизнес-приложением к таблице баз данных или файлу; *«V_»* – матрица объемов информации ($\bar{V}$), получаемой после процессорной обработки в генераторе или в бизнес приложении; *«W»* – матрица обращений типов запросов к бизнес-приложениям; *«X»* – матрица распределения таблиц баз данных или файлов по серверам; *«Y»* – матрица распределения бизнес-приложений по серверам приложений; *«My»* – матрица интенсивностей обслуживания запросов пользователей в узлах сети (μ); *«InfoNodes»* – матрица характеристик серверов БД или файл-серверов; *«InfoDB»* – матрица характеристик таблиц баз данных или файлов; *«InfoSP»* – матрица характеристик серверов приложений; *«InfoBA»* – матрица характеристик бизнес-приложений; *«Om»* – матрица характеристик каналов связи; «РазмерПакетов» – параметр, определяющий размер пакетов; «КоличествоСерверовБП» – параметр, описывающий количество серверов приложений в модели; «КоличествоБП» – параметр, описывающий количество бизнес-приложений в модели; «КоличествоСерверовБД» – параметр, описывающий количество серверов баз данных или файл-серверов в модели; «КоличествоБД» – параметр, описывающий количество таблиц баз данных или файлов в модели; «КоличествоПользователей» – параметр, описывающий количество генераторов трафика в модели; «КоличествоТиповЗапросов» – параметр, описывающий количество возможных типов запросов в модели; «Архитектура» – параметр, описывающий архитектуру РСОИ; *«MinQ», «MaxQ»* – параметры,

описывающие минимальный и максимальный объем информации, запрашиваемый запросом; *«Min_Q», «Max_Q»* – параметры, описывающие минимальный и максимальный объем информации, полученный по запросу; *«MinV», «MaxV»* – параметры, описывающие минимальный и максимальный объем таблиц баз данных или файлов; *«Min_r_V», «Max_r_V»* – параметры, описывающие минимальный и максимальный объем бизнес-приложений; *«MinVV», «MaxVV»* – параметры, описывающие минимальную и максимальную скорость чтения жестких дисков в серверах БД или файл-серверах; *«MinVD», «MaxVD»* – параметры, описывающие минимальную и максимальную скорость записи жестких дисков в серверах БД или файл-серверах; *«MinPU», «MaxPU»* – параметры, описывающие минимальную и максимальную производительность процессора в серверах БД или файл-серверах; *«MinVSP», «MaxVSP»* – параметры, описывающие минимальную и максимальную скорость чтения жестких дисков в серверах приложений; *«MinDSP», «MaxDSP»* – параметры, описывающие минимальную и максимальную скорость записи жестких дисков в серверах приложений; *«MinPSP», «MaxPSP»* – параметры, описывающие минимальную и максимальную производительность процессора в серверах приложений; *«Alfa1»* – параметр (α_2), описывающий постоянную задержку при обработке в серверах приложений; *«Alfa0»* – параметр (α_1), описывающий постоянную задержку при обработке в серверах БД и файл-серверах; *«Om0»* – параметр (θ_0), описывающий постоянную задержку при передаче данных по каналу связи; «СозданиеСерверовБД» – функция, которая создает заданное количество экземпляров серверов БД или файл-серверов в модели; «СозданиеСерверовБП» – функция, которая создает заданное количество экземпляров серверов приложений в модели; «СозданиеПользователей» – функция, которая создает заданное количество экземпляров генераторов

трафика в модели; «СозданиеКаналовСвязи» – функция, которая создает заданное количество экземпляров каналов и коммутаторов в модели; «СоединитьПорты» – функция, которая связывает все объекты модели между собой, согласно правилам соединения; «ЗаполнитьИнтенсивностиОбслуживания» – функция расчета элементов матрицы интенсивностей обслуживания запросов пользователя в узлах РСОИ.

Агент *Request* (рисунок 2) имитирует передачу заявок в моделируемой распределённой СОИ. Включает следующие компоненты:

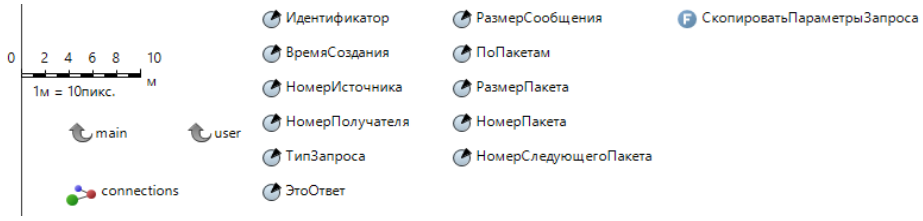

Рисунок 2 – Основные компоненты агента *Request*

На рисунке 2: *«main»* – ссылка на агента верхнего уровня; *«user»* – ссылка на агента верхнего уровня; *«connections»* – коллекция связей агента с другими агентами; «Идентификатор» – параметр, определяющий идентификатор агента в коллекции *dataBases[]* агента *User*; «ВремяСоздания» – параметр, который хранит время создания данного агента; «НомерИсточника» – параметр, который хранит номер источника, создавшего данный объект; «НомерПолучателя» – параметр, определяющий адрес сервера, которому предназначено данное сообщение; «ТипЗапроса» – параметр, определяет тип запроса (создание, чтение, обновление и удаление данных); «ЭтоОтвет» – параметр, определяющий, является ли данное сообщение ответом от сервера или нет; «РазмерСообщения» – параметр, определяющий размер сообщения; «ПоПакетам» – параметр, определяющий необходимость деления сообщения на пакеты; «РазмерПакета» – параметр,

определяющий размер пакета при делении; «НомерПакета» – параметр, определяющий номер пакета; «НомерСледующегоПакета» – параметр, определяющий номер следующего пакета при передаче; «СкопироватьПараметрыЗапроса» – функция, которая принимает на вход два параметра «Приемник» и «Источник» типа *Request* и инициализирует свойства объекта «Приемник» соответствующими свойствами объекта «Источник».

Агент *User* отвечает за обработку потока заявок в распределенной СОИ. Данный агент имеет следующие компоненты, необходимые для адекватного функционирования распределённой СОИ (рисунок 3).

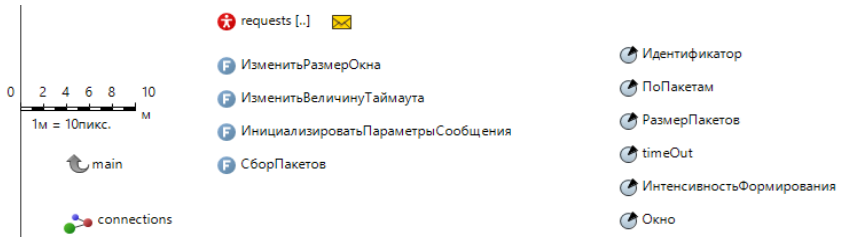

Рисунок 3 – Основные компоненты агента *User*

На рисунке 3: *«main»* – ссылка на агента верхнего уровня; *«requests[]»* – коллекция сообщений, сгенерированных агентом; «Идентификатор» – параметр, определяющий идентификатор агента в коллекции агента верхнего уровня *users[]*; «ПоПакетам» – параметр, определяющий деление сообщения на пакеты; «РазмерПакетов» – параметр, определяющий размер пакетов; *«timeOut»* – параметр, который определяет время задержки пакета в буфере до повторной передачи (вычисляется динамически на основании замеров времени прохождения пакетов); «ИнтенсивностьФормирования» – параметр, являющийся случайной величиной, имеющей распределение *Pareto*; «Окно» – параметр, определяющий максимально допустимое количество одновременно отправляемых пакетов без ожидания подтверждения от сервера (динамически изменяется); «ИзменитьРазмерОкна» – функция,

позволяющая динамически влиять на размер «окна» агента; «ИзменитьВеличинуТаймаута» – функция, позволяющая динамически влиять на время задержки пакета в буфере до повторной передачи; «ИнициализироватьПараметрыСообщения» – функция, инициализирующая параметры создаваемого сообщения; «СборПакетов» – функция, позволяющая проверить возможность и собрать пакеты в сообщение, проходит по всем пакетам, находящимся во входном буфере агента.

На рисунке 4 представлена диаграмма процесса созданного агента *User*.

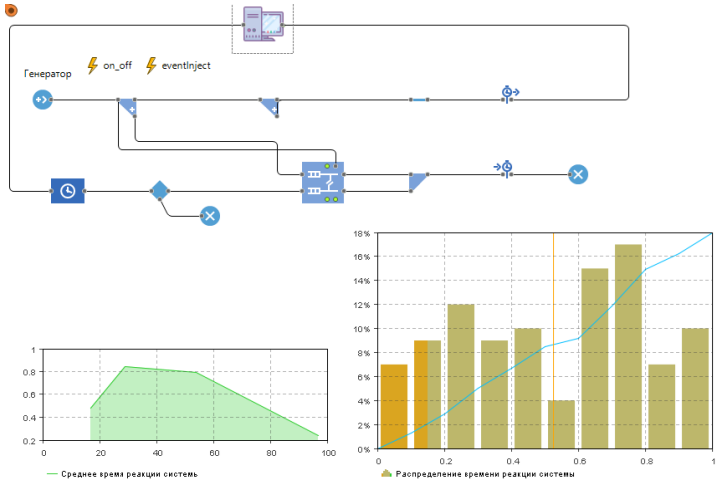

Рисунок 4 – Диаграмма процесса агента *User*

Из рисунка 4 видно, что агент имеет временную диаграмму, которая отображает среднее время реакции системы на запросы данного агента и гистограмму распределения времени реакции системы, которая отображает: плотность вероятности распределения заявок; интегральную функцию распределения заявок; среднее значение времени реакции системы.

Агент *Link* имеет следующие компоненты, необходимые для адекватного функционирования распределённой СОИ (рисунок 5).

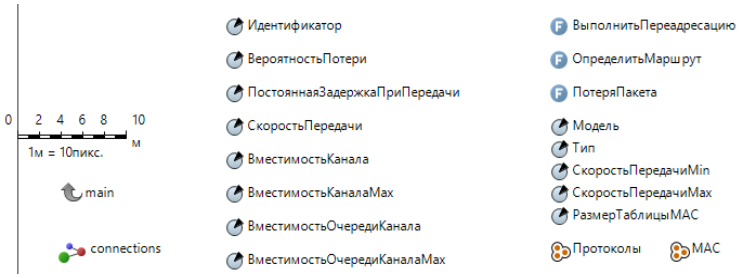

Рисунок 5 – Основные компоненты агента *Link*

На рисунке 5: *«main»* – ссылка на родительский класс агента; *«connections»* – коллекция связей агента с другими агентами; «Идентификатор» – параметр, определяющий идентификатор агента в коллекции агента верхнего уровня *links[]*; «Модель» – параметр, определяющий наименование модели коммутатора; «Тип» – параметр, определяющий тип коммутатора; «ВероятностьПотери» – параметр, определяющий вероятность потери при передаче пакета по сети; «ПостояннаяЗадержкаПриПередачи» – параметр, определяющий постоянную задержку при передаче по каналу связи; «СкоростьПередачи» – параметр, определяющий текущую скорость передачи данных в сети; «СкоростьПередачиMin» – параметр, определяющий минимальную скорость передачи данных в сети; «СкоростьПередачиMax» – параметр, определяющий максимальную скорость передачи данных в сети; «ВместимостьКанала» – параметр, определяющий текущую пропускную способность канала связи; «ВместимостьКаналаMax» – параметр, который определяет, является ли пропускная способность канала связи неограниченной или нет; «ВместимостьОчередиКанала» – параметр, определяющий текущую максимальную вместимость очереди в канал связи; «ВместимостьОчередиКаналаMax» – параметр, который определяет, является ли вместимость очереди в канал связи неограниченной или нет; «РазмерТаблицыMAC» – параметр, определяющий максимальное количество строк в таблице маршрутизации; «Протоколы» – коллекция, определяющая

список поддерживаемых протоколов передачи данных; *«MAC»* – коллекция, определяющая таблицу маршрутизации; «ВыполнитьПереадресацию» – функция, которая передает управление следующему коммутатору, при отсутствии адреса-получателя в таблице маршрутизации; «ОпределитьМаршрут» – функция, которая в соответствии с таблицей маршрутизации определяет адрес-получателя; «ПотеряПакета» – функция, которая имитирует процесс потери данных в сети.

На рисунке 6 представлена диаграмма процесса созданного агента *Link.*

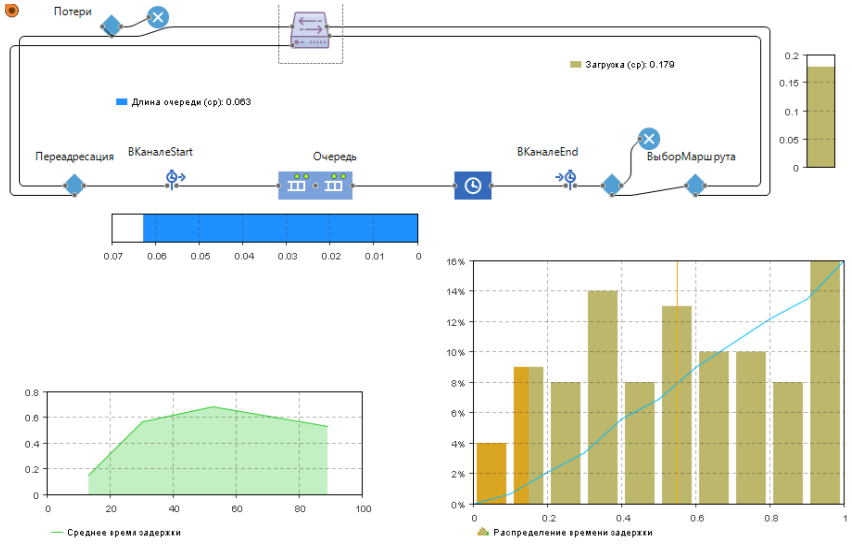

Рисунок 6 – Диаграмма процесса агента *Link*

Из рисунка 6 видно, что кроме основного графика распределения времени задержки в агенте, есть еще две столбиковые диаграммы, отражающие среднюю длину очереди и среднюю загрузку коммутатора в данный момент и временная диаграмма, отражающая среднее время задержки.

Агент *Server* имитирует работу одного из узлов распределённой СОИ. Данный агент имеет следующие компоненты, необходимые для адекватного функционирования распределённой СОИ (рисунок 7).

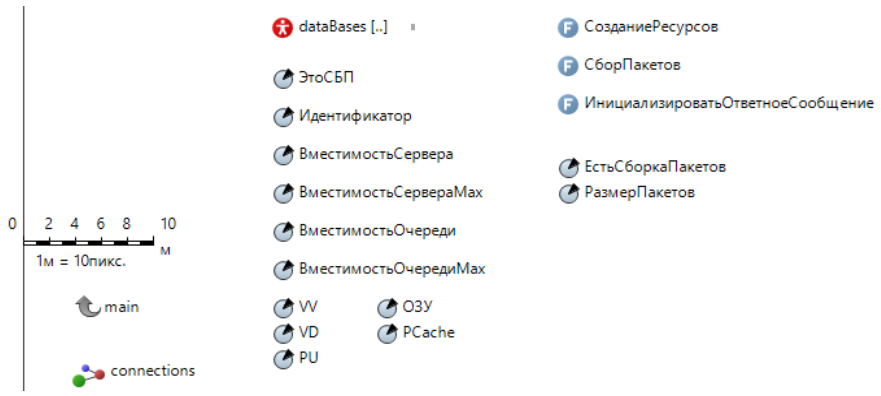

Рисунок 7 – Основные компоненты агента *Server*

На рисунке 7: *«main»* – ссылка на родительский класс агента; *«connections»* – коллекция связей агента с другими агентами; *«dataBases[]»* – коллекция ресурсов, размещенных в данном агенте; «ЭтоСБП» – параметр, который определяет, является ли данный агент сервером бизнес-приложений или нет; «Идентификатор» – параметр, определяющий идентификатор агента в коллекции агента верхнего уровня *databaseServers[]* или *appServers[]*; «ВместимостьСервера» – параметр, определяющий текущее максимальное количество одновременно выполняемых задач; «ВместимостьСервераMax» – параметр, который определяет, является ли количество одновременно выполняемых задач неограниченным или нет; «ВместимостьОчереди» – параметр, определяющий текущую максимальную вместимость очереди на обработку в сервере; «ВместимостьОчередиMax» – параметр, который определяет, является ли вместимость очереди на обработку в сервере неограниченной или нет; *«VV»* – параметр, определяющий скорость чтения жесткого диска сервера; *«VD»* – параметр, определяющий скорость записи жесткого диска сервера; *«PU»* – параметр, определяющий производительность процессора; «ОЗУ» – параметр, определяющий общий объем оперативной памяти в сервере; *«PCache»* – параметр, определяющий объем кэша процессора; «ЕстьСборкаПакетов» – параметр, который определяет, есть ли процесс сбора пакетов или нет; «РазмерПакетов» –

параметр, определяющий размер пакетов; «СозданиеРесурсов» – функция, которая создает ресурсы агента согласно исходным данным размещения объектов; «СборПакетов» – функция, позволяющая проверить возможность и собрать пакеты в сообщение. Проходит по всем пакетам, находящимся во входном буфере агента. «ИнициализироватьОтветноеСообщение» – функция инициализирует параметры ответного сообщения.

На рисунке 8 представлена диаграмма процесса созданного агента *Server*.

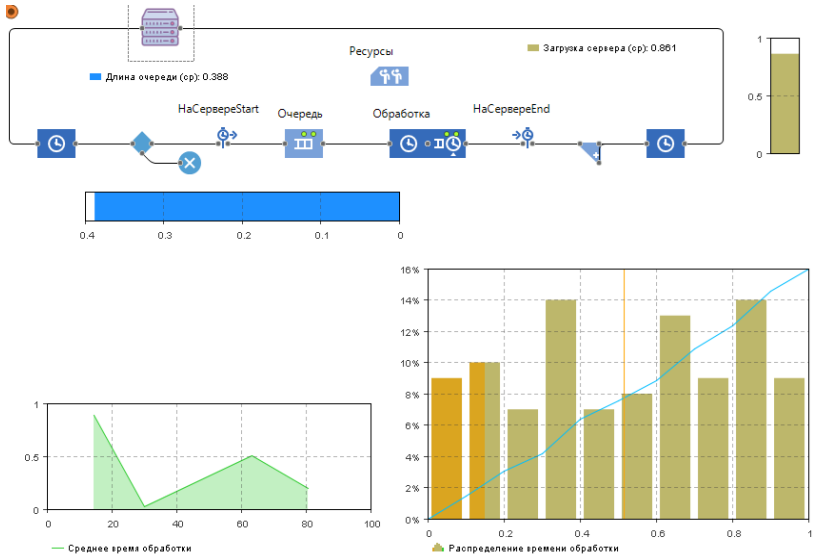

Рисунок 8 – Диаграмма процесса агента *Server*

Из рисунка 8 видно, что кроме основного графика распределения времени обработки в узле, есть еще две столбиковые диаграммы, отражающие среднюю длину очереди и среднюю загрузку узла в данный момент и временная диаграмма, отражающая среднее время обработки заявок.

Агент *Resource* имеет компоненты, представленные на рисунке 9.

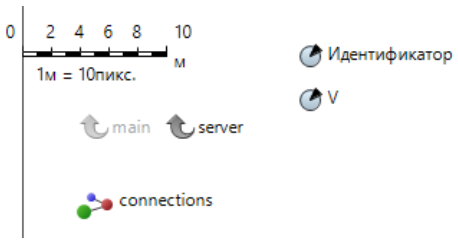

Рисунок 9 – Основные компоненты агента *Resource*

На рисунке 9: *«main»* – ссылка на агента верхнего уровня (удалена); *«server»* – ссылка на агента верхнего уровня; *«connections»* – коллекция связей агента с другими агентами; «Идентификатор» – параметр, определяющий идентификатор агента в коллекции агента верхнего уровня *dataBases []*; *«V»* – параметр, определяющий объем данного агента.

Главная форма программы с рабочей областью и консолью разработчика позволяет отслеживать процесс функционирования модели (рисунок 10).

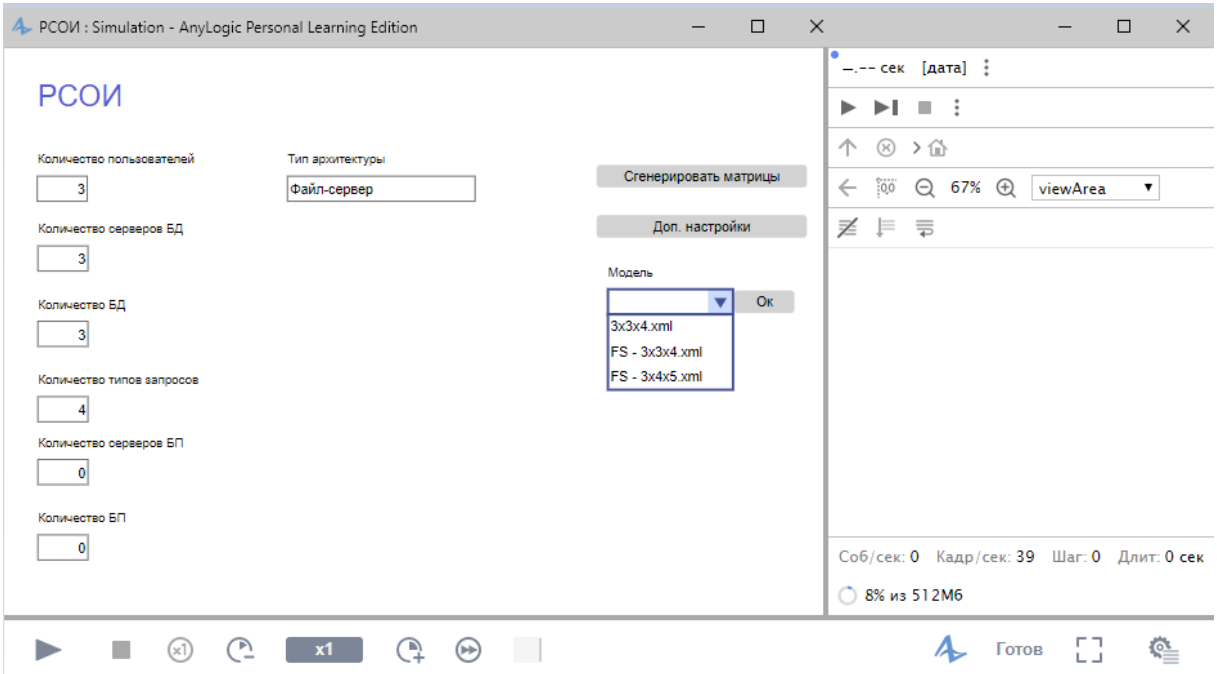

Рисунок 10 – Главная форма программы

Проведение экспериментов и подтверждение адекватности разработанных моделей. Эксперименты были проведены на ПК, имеющем следующие технические характеристики: процессор *Intel* 2,2 ГГц, ОЗУ 16 ГБ, при исходных данных, которые представлены в работе [8]. Для моделирования закона поступления заявок на вход системы было использовано распределение Вейбулла [3]. Реальные (эталонные) показатели среднего времени реакции распределенных СОИ были получены путем конфигурирования аналогичных архитектур сети между несколькими виртуальными средами, управляемыми свободным и открытым программным обеспечением *Vagrant*. Каждая среда при этом обладает поддержкой контейнеров, поэтому процесс доставки, развертывания и управления серверами полностью автоматизирован с помощью программного обеспечения *Docker* и программного обеспечения *Kubernetes* [10]. Контейнеры при этом могут быть следующих видов: клиент-приложение – сервер с консольным приложением, которое реализует логику клиентских обращений. В случае файл-серверной архитектуры – приложение содержит прикладные функции, которые определяют главные методы решения задач приложения, а также методы обработки данных. В случае клиент-серверной архитектуры – приложение выступает *http*-клиентом, реализуя *API*-методы сервера, выступает аналогом компонента «Генератор трафика» в имитационных моделях; *API*-приложение – сервер с развернутым *ASP. NET Core WebApi* приложением. Данный компонент используется в случае трехуровневой клиент-серверной архитектуры сети. Выступает аналогом компонента «Сервер бизнес-приложения» в имитационных моделях; СУБД-сервер – сервер с развернутой СУБД *PostgreSQL*. Данный компонент отсутствует в случае файл-серверной архитектуры. В случае двухуровневой архитектуры – дополнительно содержит *ASP. NET Core WebApi* приложением. Выступает аналогом компонента «Сервер баз-данных»

в имитационных моделях; файловый сервер – сервер, содержащий тестовые наборы файловых данных. Данный компонент отсутствует в случаях клиент-серверной архитектуры сети. Выступает аналогом компонента «файл-сервер» в имитационных моделях; прокси-сервер – сервер с развернутым *Nginx* веб-сервером.

Результаты численных экспериментов приведены в таблицах 1-3.

Таблица 1. Распределённая СОИ на базе архитектуры «файл-сервер»

Размерность задачи *nxdxq*	Значение $\overline{T}$				Значение $\overline{T}$			
	exp CeMO	ДА	ИМ	Реал	exp CeMO(б)	ДА(б)	ИМ(б)	Реал(б)
5x10x10	0.612	0.701	0.834	0.726	0.711	0.841	1.003	0.914
7x30x20	0.855	1.012	1.896	1.673	1.411	1.527	1.842	1.701
9x50x30	1.145	1.242	1.438	1.521	2.275	2.432	2.867	2.715
11x70x40	1.359	1.326	1.437	1.497	3.018	3.237	3.775	3.482
13x90x50	1.432	1.481	1.818	1.711	3.436	3.634	4.112	3.958
15x100x60	1.616	1.828	2.022	1.944	4.218	4.416	5.024	4.795

Таблица 2. Распределённая СОИ на базе двухуровневой архитектуры «клиент-сервер»

Размерность задачи *nxdxq*	Значение $\overline{T}$				Значение $\overline{T}$			
	exp CeMO	ДА	ИМ	Реал	exp CeMO(б)	ДА(б)	ИМ(б)	Реал(б)
5x10x10	0.141	0.167	0.184	0.178	0.212	0.284	0.412	0.434
7x30x20	0.121	0.143	0.177	0.181	0.327	0.416	0.587	0.573
9x50x30	0.114	0.125	0.169	0.193	0.523	0.588	0.714	0.662
11x70x40	0.089	0.123	0.175	0.184	0.661	0.720	0.819	0.840
13x90x50	0.113	0.149	0.188	0.211	0.801	0.821	0.981	0.963
15x100x60	0.165	0.175	0.232	0.245	0.850	0.905	1.122	1.211

Таблица 3. Распределённая СОИ на базе трёхуровневой архитектуры «клиент-сервер»

Размерность задачи *nxdxq*	Значение $\bar{T}$				Значение $\bar{T}$			
	exp CeMO	ДА	ИМ	Реал	exp CeMO(б)	ДА(б)	ИМ(б)	Реал(б)
5x10x10	0.027	0.034	0.048	0.063	0.112	0.128	0.139	0.158
7x30x20	0.033	0.041	0.059	0.079	0.190	0.111	0.182	0.189
9x50x30	0.043	0.051	0.068	0.094	0.277	0.364	0.667	0.725
11x70x40	0.039	0.049	0.071	0.108	0.305	0.319	0.775	0.811
13x90x50	0.045	0.053	0.088	0.112	0.436	0.635	0.912	0.924
15x100x60	0.059	0.068	0.122	0.143	0.658	0.717	1.107	1.117

В таблицах 1-3: n – количество узлов; d – количество информационных отношений; q – количество запросов к информационным отношениям; expCeMO (expCeMO(б)) – использование точного метода моделирования, как с учетом блокировок, так и без; ДА (ДА(б)) – использование метода декомпозиционной аппроксимации [10], как с учетом блокировок, так и без; ИМ (ИМ(б)) - использование метода имитационного моделирования с распределением Вейбулла [3], как с учетом блокировок, так и без; Реал (Реал(б)) – реальные (эталонные) показатели среднего времени реакции, как с учетом блокировок, так и без.

Выводы

1. Реактивность моделей распределённых СОИ, полученная с использованием метода имитационного моделирования для самоподобного входного трафика заявок, на базе распределения Вейбулла, оказалась наибольшей в сравнении с реактивностями распределённых СОИ, полученных другими методами аналитико-численного моделирования, и при этом, значения данных величин приемлемо близки к значениям, полученным практическим способом (на реальной сети), что и подтверждает адекватность

разработанных имитационных моделей функционирования распределённых СОИ.

2. Наличие в моделях блокировок на уровне всего информационного отношения для рассмотренных распределённых СОИ не оказывает существенного влияния на величину их реактивности.

3. Разработанные и программно реализованные имитационные модели функционирования распределённых СОИ с использованием самоподобного входного потока заявок на основе распределения Вейбулла могли бы использоваться при внедрении ИС в различных предметных областях, например, когда необходимо организовать более рациональный вычислительный процесс, или обеспечить требуемые показатели эффективности работы ИС.

Литература

1. Lozhkovskyi A., Levenberg Y. "Investigation of simulating methods for self-similar traffic flows: The QoS-characteristics depend on the type of distribution in self-similar traffic" / 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), 2017. – pp. 54-71.

2. Задорожный В. Н., Долгушин Д. Ю., Юдин Е. Б. Аналитико-имитационные методы решения актуальных задач системного анализа больших сетей / Под ред. В. Н. Задорожного. – Омск: Издательство ОмГТУ, 2013. – 324 с.

3. Будко П. А., Рисман О. В. Многоуровневый синтез информационно-телекоммуникационных систем. Математические модели и методы оптимизации. Монография. – СПб: ВАС, 2011. – 476 с.

4. Скоба А.Н., Состина Е.В. Математическая модель оптимального размещения распределённой базы данных по узлам ЛВС на базе

двухуровневой клиент-серверной архитектуры // Инженерный вестник Дона, 2015, №2. URL:ivdon.ru/ru/magazine/archive/n2y2015/2882.

5. Скоба А.Н., Айеш Ахмед Нафеа Айеш. Математическая модель функционирования распределённой информационной системы на базе трёхуровневой клиент-серверной архитектуры без учёта влияния блокировок // Инженерный вестник Дона, 2018, №1. URL:ivdon.ru/ru/magazine/archive/n1y2018/4658.

6. Скоба А.Н., Михайлов В.К., Айеш Ахмед Нафеа Айеш. Модель оптимального размещения информационных ресурсов по узлам распределённой системы обработки информации предприятия на базе трёхуровневой архитектуры «клиент-сервер» с учётом влияния блокировок. // Изв. вузов. Электромеханика, 2018. Т. 61. №3. С. 68-75.

7. Макаренко С. И. Анализ математических моделей информационных потоков общего вида и степени их соответствия трафику сетей интегрального обслуживания // Вестник Воронежского государственного технического университета. 2012. – Т. 8. – № 8. – С. 28-35.

8. Бахарева Н. Ф., Карташевский И. В., Тарасов В. Н. Анализ и расчет непуассоновских моделей трафика в сетях ЭВМ // Инфокоммуникационные технологии. 2009. – Т. 7. – № 4. – С. 61-66.

9. Zhang Huachuan, Tian Jie, Xu Jing. "The Implementation of a Distributed System Based on a Parallel Algorithm for Self-similar Network Traffic Simulation". International Symposium on Information Science and Engineering, 2008, pp. 53-57.

10. Тарасов В. Н., Бахарева Н. Ф., Горелов Г. А. Математическая модель трафика с тяжелохвостным распределением на основе системы массового обслуживания H2/M/1 // Инфокоммуникационные технологии. 2014. – Т. 12. – № 3. – С. 36-41.

References

1. Lozhkovskyi A., Levenberg Y. 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), 2017. pp. 54-71.

2. Zadorozhny`j V. N., Dolgushin D. Yu., Yudin E. B. Analitiko-imitacionnye metody resheniya aktual'nyh zadach sistemnogo analiza bol'shih setej [Analytical and simulation methods for solving actual problems of system analysis of large networks]. Omsk: Izdatel`stvo OmGTU, 2013. p. 324.

3. Budko P. A., Risman O. V. Mnogourovnevyj sintez informacionno-telekommunikacionnyh sistem. Matematicheskie modeli i metody optimizacii. Monografiya. [Multilevel synthesis of information and telecommunication systems. Mathematical models and optimization methods: Monograph]. SPb.: VAS, 2011. p. 476.

4. Skoba A.N., Sostina E.V. Inzhenernyj vestnik Dona, 2015. №2. URL:ivdon.ru/ru/magazine/archive/n2y2015/2882.

5. Skoba A.N., Ayesh Achmed Nafea Ayesh Inzhenernyj vestnik Dona, 2018. №1. URL:ivdon.ru/ru/magazine/archive/n1y2018/4658.

6. Skoba A.N., Mikhaylov V.K., Ayesh Achmed Nafea Ayesh. Izv. vuzov. E`lektromexanika, 2018. T. 61. №3. pp. 68-75.

7. Makarenko S. I. Vestnik Voronezhskogo gosudarstvennogo texnicheskogo universiteta, 2012. Vol. 8. №2. pp.28-35

8. Baxareva N. F., Kartashevskij I. V., Tarasov V. N. Infokommunikacionny`e texnologii, 2009. Vol.7. №4. pp. 61-66.

9. Zhang Huachuan, Tian Jie, Xu Jing. International Symposium on Information Science and Engineering, 2008, pp. 53-57.

10. Tarasov V. N., Baxareva N. F., Gorelov G. A. Infokommunikacionny`e texnologii, 2014. Vol. 12. № 3. pp. 36-41.