УДК 681.51.015.26:330.43

РАЗРАБОТКА СИСТЕМЫ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ ЭКОНОМИЧЕСКИХ ОБЪЕКТОВ НА ОСНОВЕ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПОДХОДА

А.А. Мицель, Е.Б. Грибанова

Томский государственный университет систем управления и радиоэлектроники
E-mail: maa@asu.tusur.ru

Рассматривается объектно-ориентированная архитектура системы имитационного моделирования экономических объектов. Приведен пример решения задачи моделирования с помощью реализованной программы.

Введение

Согласно определению Г. Буча [1] под объектно-ориентированным подходом понимается технология создания программного обеспечения, основанная на представлении программы в виде совокупности программных объектов, каждый из которых является экземпляром определенного типа (класса), а классы образуют иерархию с наследованием свойств. Как правило, полученные решения являются простыми для модификации и обслуживания и более пригодны для повторного использования, поэтому данный подход используется для реализации сложных систем.

В настоящее время объектно-ориентированная технология все чаще применяется и для разработки систем имитационного моделирования. При этом предпринимаются попытки создания как универсального средства, так и предметно-ориентированных систем (для имитации системы массового обслуживания, цепей поставок и т. д.). Наибольшее количество работ посвящено рассмотрению дискретных имитационных моделей, состояния которых меняются в определенные моменты времени. Их построение может быть выполнено с использованием трех подходов: событийного (в основе лежит список упорядоченных по времени событий, наступление которых определенным образом планируется), сканирования активностей (планирования наступления событий не происходит, а действия совершаются по мере выполнения определенных условий), процессно-ориентированного (с точки зрения «жизненного цикла» объекта) [2].

Обзор объектно-ориентированных систем имитационного моделирования

К сожалению, большинство разработчиков программного обеспечения имитационного моделирования экономических систем не приводят описание своих архитектурных решений. В данном разделе будут рассмотрены некоторые известные шаблоны, а также программы, разработанные на основе объектно-ориентированной технологии, предназначенные для решения задач в различных областях экономики.

Система Pilgrim (МЭСИ), реализованная на C++, является результатом работы, направленной

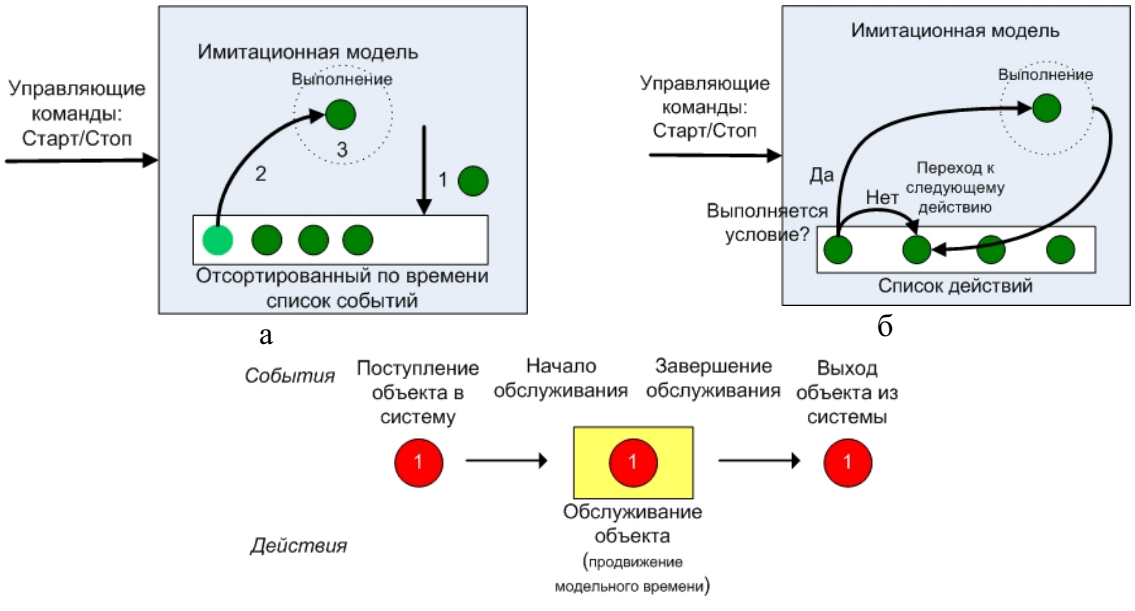

Рис. 1. *Подходы: а) событийный – 1 – планирование события; 2 – получение события из списка; 3 – выполнение события; б) сканирования активностей; в) процессный*

на создание универсального инструмента имитационного моделирования [3]. Данный пакет позволяет создавать дискретно-непрерывные модели и включает CASE-оболочку. Концепция моделирующей системы базируется на шести основных понятиях: граф модели, транзакт (формальный запрос на какое-либо обслуживание), узел графа сети (т. е. центры обслуживания транзактов), событие, ресурс, пространство (узлы, транзакты и ресурсы могут быть привязаны к точкам пространства и мигрировать в нем). Всего в системе Pilgrim 17 типов узлов, и некоторые из них предоставляют следующие средства: возможность работы с непрерывными процессами; моделирование пространственной динамики; работа с ресурсами, представляющими собой деньги и материальные ценности, счета бухгалтерского учета, банковские счета.

Рассмотрим также объектно-ориентированную архитектуру SIMFONE [4]. Данная разработка может быть использована в качестве базиса для создания библиотек и полезна для понимания концепций имитационного моделирования. Так, реализация архитектуры выполнена авторами в виде библиотеки классов Java Simulation Library (JSL) для поддержки имитации на Java.

SIMFONE разделен на 6 главных пакетов:

- управление имитацией;
- выполнение имитации;
- процессы;
- ресурсы и очереди;
- генерирование случайных чисел;
- статистика.

Каждый из перечисленных пакетов предназначен для создания определенной организации набора классов, которая выполняет важные функции.

Приведем теперь некоторые предметно-ориентированные разработки.

В работе [5] описана архитектура системы, предназначенной для имитации цепей поставок. Ее реализация была выполнена на языке Java с применением распределенной системы D-SOL, использующей событийный подход [6]. Архитектура содержит три слоя: первый формируется набором «актеров», которые могут обмениваться сообщениями. Второй слой содержит наборы «актеров», характерных для цепей поставок (таких как покупатель, поставщик и т. д.) и обменивающихся специфическими сообщениями (заказ, оплата). Третий слой расширяет библиотеку для реализации определенного режима работы с системой: демонстрационного, интерактивного либо игрового, когда часть «актеров» контролируется компьютерами, а часть – студентами.

SISCO [7] – инструмент имитационного моделирования цепей поставок, который позволяет пользователю графически строить структуру цепи, и определять соответствующую информацию. Основная структура системы включает три модуля:

- The Visual Supply Chain Editor (используется для определения структуры цепи, которая затем сохраняется в файл на специальном языке описания, основанном на XML);
- The Experiment Designer (служит для определения специфической информации имитационного моделирования: число случайных реализаций, выходная статистика и т. д.)
- The Model Builder (осуществляет автоматическую генерацию модели на основе графической структуры, и инициализирует ее выполнение с помощью Silk – набора Java классов, предназначенного для имитационного моделирования).

Используемый в SISCO подход (процессный) предполагает осуществление моделирования с точки зрения жизненного цикла заказа, который проходит следующие стадии: создание, размещение транспортировка, обработка, получение.

В статье [8] приводится описание объектно-ориентированной архитектуры имитационной системы управления запасами, для реализации которой была использована библиотека JSL. Свою разработку автор назвал «мостом», который (благодаря открытой архитектуре) связывает коммерческие системы общего назначения и специфические имитационные модели.

Программа имитационного моделирования систем массового обслуживания ObjectSim [9] (язык реализации – Delphi, событийный подход) позволяет без изучения синтаксиса конкретного, специализированного языка имитационного моделирования строить имитационные модели, комбинируя визуальные компоненты с написанием собственных обработчиков (используя скриптовые языки) для различных событий.

В сети Интернет бесплатно распространяется программа JASA [10] (реализована на Java). Ее функциональность позволяет осуществлять моделирование двойного аукциона, в котором участвует множество покупателей и продавцов одной единицы предмета аукциона. Также данная программа дает возможность использовать при имитации поведения участников различные стратегии. Целью моделирования с помощью JASA является исследование рынков, где одновременно взаимодействуют поставщики и потребители, например, фондовые рынки.

Рассмотрев некоторые из существующих работ, можно прийти к выводу, что объектно-ориентированный подход успешно применяется для разработки систем имитационного моделирования, используемых в различных областях экономики. При этом реализованные имитационные модели могут иметь различное назначение: оценка вариантов предполагаемых изменений, игровое обучение сотрудников, визуальное представление функционирования рассматриваемого объекта во времени и т. д.

Постановка задачи

Авторами данной работы были реализованы экономические имитационные модели, часть из

которых является классическими (и их описание приводится в работах [11, 12]), а остальные разработаны путем использования существующих алгоритмов в других областях экономики либо модификации самих моделей. В настоящий момент всего имеется 17 моделей в том числе: экскурсионной фирмы, грузоперевозок, кредитного отдела, вычислительного центра, производственной фирмы, мониторинга рынка, управления запасами с периодической стратегией подачи заявок и др. (описание некоторых из перечисленных имитационных моделей содержится в публикациях [13, 14]). Данные модели описывают различные системы: массового обслуживания, управления запасами и т. д.

Задача заключается в разработке архитектуры, предоставляющей возможности гибкой модификации и расширения системы, например, добавления новых моделей. При этом для каждой модели необходимо иметь возможность выполнять дополнительные подзадачи:

- расчет статистических характеристик, который включает получение следующих оценок выходной величины: тип распределения, среднее значение, дисперсия, среднее квадратическое отклонение, медиана, коэффициент вариации, асимметрия;
- оценка рисков: расчет вероятности того, что выходная величина будет не меньше некоторого заданного значения, а также вероятность и средний ожидаемый размер убытков;
- построение графиков, описывающих изменение различных величин по реализациям и на протяжении рассматриваемого периода, а также гистограммы и функции распределения;
- проведение экспериментов: имитация системы с различными значениями некоторой исходной величины для определения поведения выходного параметра.

Полученные таким образом данные должны пересчитываться при изменении входных параметров модели, а сами подзадачи создаваться и удаляться по желанию пользователя в связи со значительными затратами времени на их реализацию (особенно, при большом числе прогонов).

Описание архитектуры программы

При разработке архитектуры системы за основу был взят шаблон, описание которого приводится в источнике [15]. Согласно выбранной концепции алгоритм решения задачи разбивается на этапы, на каждом из которых на основании данных введенных пользователем, а также рассчитанных на других этапах происходит решение какой-либо подзадачи выбранным методом.

Представление описанной задачи для отдельной модели в виде дерева показано на рис. 2.

Для реализации этапов было использовано представление в виде многосвязного списка, каждый элемент которого является объектом класса «Узел». Можно выделить два типа таких объектов. К первому типу относятся объекты, создаваемые при загрузке системы, т.е. все используемые модели (список доступных пользователю моделей указывается в файле). Объекты класса «Узел» второго типа (показаны на рис. 2 на нижнем уровне) создаются и удаляются пользователем динамически.

Структура основных классов показана на рис. 3. Рассмотрим отличие архитектуры нашей системы от предложенной в работе [15]:

- наличие классов «ГенераторСлучайныхВеличинСобытий» (предназначен для генерирования случайных величин и событий, поскольку данные действия необходимо выполнять для каждой имитационной модели) и «Статистика» (содержит методы для обработки результатов моделирования);
- для реализации экспериментов применяются «циклические» узлы, вызывающие своих предков заданное число раз.

Так, на рис. 4 показана диаграмма последовательности для случая, когда пользователь изменил исходные данные модели управления запасами и действия, выполняемые при этом: имитация системы с новыми данными; проведение экспериментов. Т. е. сначала происходит вызов первого узла (с помощью метода «вызовПотомков»), который не является «циклическим», и вызывается метод «решение» соответствующего ему класса «Расчет». Далее

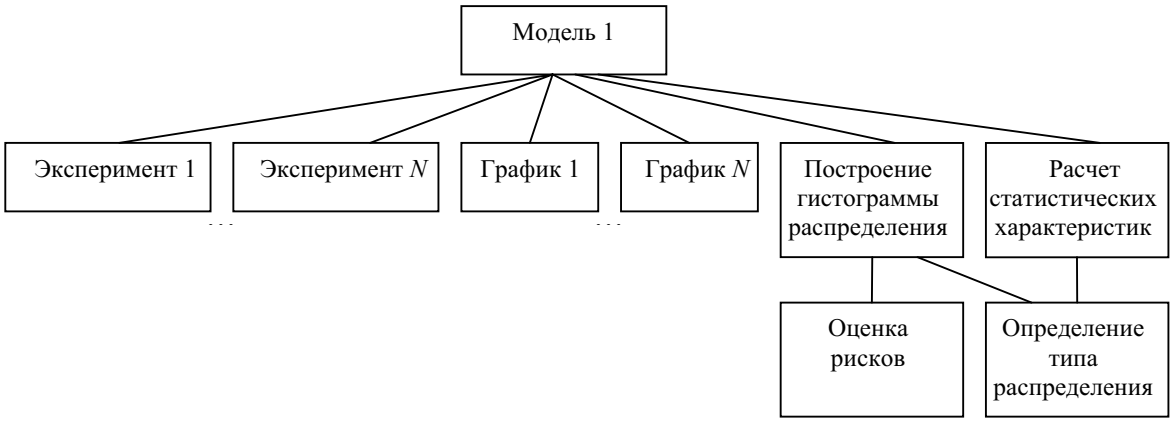

Рис. 2. *Дерево решения задачи*

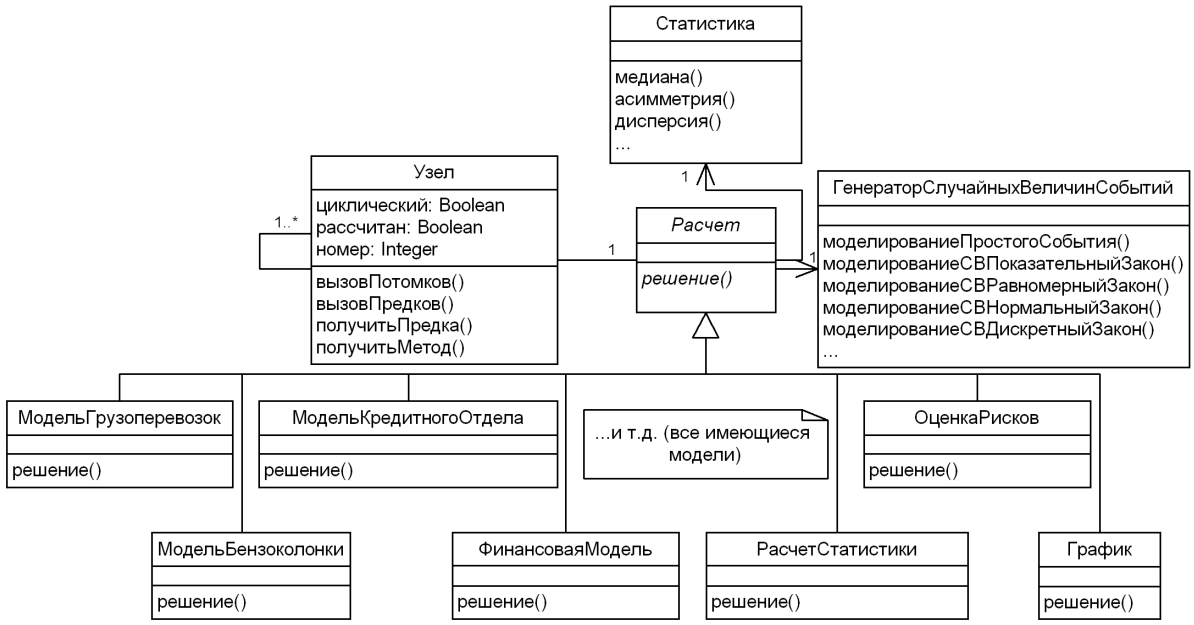

Рис. 3. *Структура классов*

вызывается «циклический» потомок текущего узла, и поэтому прежде, чем выполнить свою задачу, он осуществляет многократный вызов метода «решение» соответствующего его предку класса «Расчет».

С помощью метода «вызовПредков» рекурсивно просматриваются все предки вызванного узла (пока не будет достигнута начальная вершина) и автоматически рассчитываются, если данная операция еще не была выполнена.

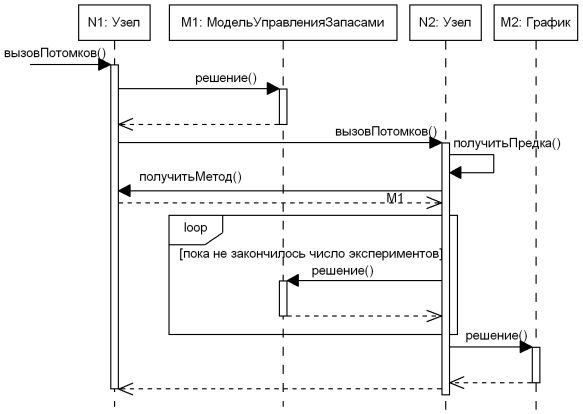

Рис. 4. *Диаграмма последовательности*

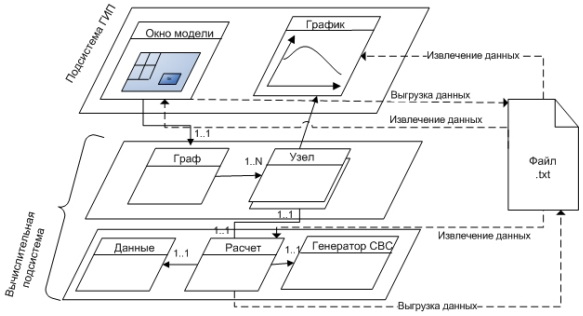

Рис. 5. *Взаимодействие вычислительной подсистемы с графическим интерфейсом пользователя*

Взаимодействие вычислительной подсистемы с графическим интерфейсом пользователя (ГИП) приведено на рис. 5. Обмен информацией здесь происходит с помощью текстового файла, содержащего выгруженные данные. Можно отметить, что интерфейс и вычислительная часть программы являются независимыми подсистемами и, следовательно, изменения в одной из них не могут непосредственно повлиять на другую.

Реализация описанной архитектуры выполнена с помощью языка программирования Java.

Пример

В качестве примера рассмотрим классическую модель управления запасами, взяв данные из источника [12]. На рис. 6 представлены результаты имитации, в том числе экспериментов, проводимых целью решения задачи моделирования – определение оптимального объема заказываемой партии товара (полученное значение составляет 300 шт.).

Заключение

Реализована программа, включающая имитационные модели различных экономических объектов, в том числе классических. Разработка выполнена с использованием объектно-ориентированного подхода, что упрощает модификацию и расширение системы. Программа содержит средства для статистической обработки результатов моделирования, оценки рисков, построения графиков, проведения экспериментов, которые могут быть динамически привязаны к конкретной модели. В качестве примера приведены результаты имитации системы управления запасами.

Программа предназначена для использования в учебном процессе, а также исследования экономических объектов.

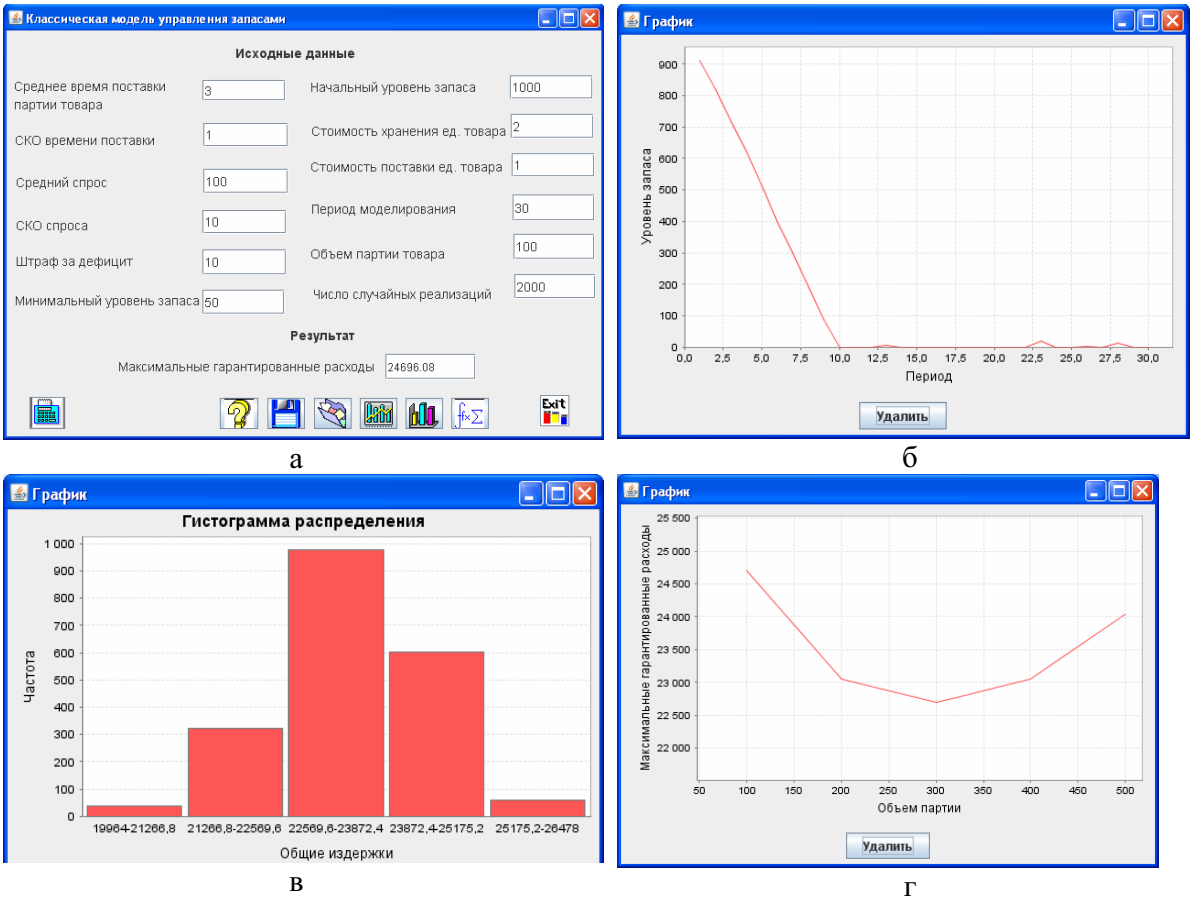

Рис. 6. *Результаты имитации: а) форма модели с исходными данными и результатом моделирования; б) изменение уровня запаса на складе в 1-й реализации; в) гистограмма распределения общих затрат; г) эксперименты с различными значениями (100, 200, 300,400, 500 шт.) заказываемого объема партии товара*

СПИСОК ЛИТЕРАТУРЫ

1. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++. – СПб.: Невский диалект, 1999. – 560 с.
2. Кельтон В., Лоу А. Имитационное моделирование. – СПб.: Питер; Киев: Издательская группа BHV, 2004. – 847 с.
3. Емельянов А.А., Власова Е.А., Дума Р.В. Имитационное моделирование экономических процессов. – М.: Финансы и статистика, 2002. – 368 с.
4. Rossetti M.D., Aylor B., Jacoby R., Prorock A., White A. SIMFONE: An Object-Oriented Simulation Framework // Proc. of the 2000 Winter Simulation Conference. – Orlando, 10–13 December 2000. – P. 1855–1864.
5. Verbraeck A., Houten S. From Simulation to Gaming: An Object-Oriented Supply Chain Training Library // Proc. of the 2005 Winter Simulation Conference. – Orlando, 4–7 December 2005. – P. 2346–2354.
6. Jacobs P., Lang N., Verbraeck A. D-SOL: A Distributed Java Based Discrete Event Simulation Architecture // Proc. of the 2002 Winter Simulation Conference. – San Diego, 8–11 December 2002. – P. 793–800.
7. Chatfield D., Harrison T., Hayya J. SISCO: A Supply Chain Simulation Tool Utilizing Silk and Xml // Proc. of the 2001 Winter Simulation Conference. – Arlington, 9–12 December 2001. – P. 614–622.
8. Rossetti M.D., Miman M., Varghese V., Xiang Y. An Object-Oriented Framework For Simulating Multi-Echelon Inventory Systems // Proc. of the 2006 Winter Simulation Conference. – Monterey, 3–6 December 1999. – P. 1452–1461.
9. Приступа А.В. Разработка программного комплекса имитационного моделирования СМО на основе объектно-ориентированной модели дискретно-событийного метода: Дис. ... канд. техн. наук. – Томск, 2006. – 160 с.
10. JASA – Java Auction Simulator API [Электронный ресурс]. – Режим доступа: http://www.csc.liv.ac.uk/~sphelps/jasa
11. Нейлор Т.Х. Имитационные эксперименты с моделями экономических систем. – М.: Мир, 1975. – 500 с.
12. Варфоломеев В.И. Алгоритмическое моделирование элементов экономических систем. – М.: Финансы и статистика, 2000. – 203 с.
13. Мицель А.А., Грибанова Е.Б. Компьютерное имитационное моделирование экономических объектов // Доклады ТУСУР. – 2005. – № 3. – С. 49–56.
14. Грибанова Е.Б. Имитационные модели экономических объектов // Актуальные проблемы экономики в творчестве студентов: Сб. статей под ред. Б.М. Генкина и др. – СПб.: СПбГИЭУ, 2006. – 2006. – С. 200–205.
15. Бойченко И.В. Программное обеспечение моделирования, обработки и анализа данных лидарного зондирования газового состава атмосферы: Дис. ... канд. техн. наук. – Томск, 2002. – 113 с.

Поступила 27.10.2006 г.