Программные средства визуализации имитационного и математического моделирования

С. Н. Мищенко
Финансовый университет при Правительстве Российской Федерации (Финуниверситет), Financial University
sundae.05@yandex.ru

***Аннотация*. Имитационное моделирование охватывает широкий спектр применения в различных областях науки: логистике, математике, теории вероятностей, географии, медицине и, конечно же, в экономических науках. Часто можем наблюдать, что аналитическая модель не учитывает все факты, следовательно, модель может быть нечеткой. Имитационные модели же используются в сложных объектах моделирования и учитывают множество различных факторов, которые приближают модель к совершенству. Так же отличием имитационного моделирования (далее ИМ) от аналитического является то, что при первом варианте нет нужды перехода к единой шкале для всех факторов, в то время как аналитическая модель это требует.**

Ключевые слова: имитационное моделирование; алгоритм; визуализация моделей; программирование

I. Введение

Имитационное моделирование (ИМ) имеет возможность строить модели со многими критериями с разными шкалами измерения. Стоит отметить, что главной особенностью ИМ является способность исследовать процесс, протекающий во времени, с учетом случайных нерегулируемых факторов. Совершенствование и развитие компьютерной техники, обладающей большой вычислительной сложностью, повлекло разработку имитационных моделей. ИМ оказывает достаточно важную роль в экономических сферах деятельности.

Для того, чтобы наглядно увидеть моделируемую область и происходящие в ней процессы и изменения используется один из новейших аспектов имитационного моделирования – визуализация.

В систему ИМ входят: AnyLogic, Repast Symphony, MadKit, MASON, Breve, Framsticks, NetLogo. Некоторые из них содержат слабые и незначительные средства визуализации (рисунки со слабой изобразительностью, на уровне точек и пикселей) другие содержат сильнейшие подсистемы, требующие значительные временные затраты (WEB-серверы и библиотеки графических моделей).

Если встроенный набор графических моделей не охватывает исследуемую область, то возникает необходимость использования стороннего программного обеспечения, такого как графические пакеты (3DSMax, Blender, Maya). Выбор, в данном случае, обусловлен опытом работы исследователя с выбранным программным обеспечением. Стоит отметить, что графические пакеты достаточно сложны для освоения, и если нет опыта по работе с ними, то время, потраченное на освоение системы, может замедлить процесс исследовательской работы, следовательно, обуславливает неэффективное использование ПО.

II. Этапы и структура программного комплекса имитационного моделирования. Основные принципы разработки системы визуализации

Применение визуализации происходит на этапах конструирования, разработки и отладки модели, а также при анализе результатов. Часто используют иерархический подход для описания структуры моделируемой системы и ее логики. Также используются технологии реляционных баз данных при хранении и обработке данных об имитационном проекте и т.д. Но даже в таких сильных симуляторах некоторые этапы ИМ пока не охвачены. Достаточно сложно и, наверное, невозможно сделать симулятор универсальным для всех дискретных систем. Потому что всегда находятся нюансы и особенности, и для каждого конкретного приложения требуются уникальные доработки.

Сейчас имеется практическая реализация подобных комплексов для различных приложений. В каждой такой реализации были свои особенности, как, например, интерфейс и данные. Но в целом можно выделить множество общих особенностей:

Управляющая программа (УП) – это основная программа комплекса. Основная ее задача – объединять все программные компоненты в единый программный комплекс, обеспечивать универсальный подход к языку диалога системы, реализовать диалог первого уровня – справочно-информационного и интеграционного, дающего возможность перейти на любые этапы исследований.

УП также делится и на все остальные основные подсистемы:

- **Задания структуры системы и ее логики.** Реализуют этапы постановки задачи, ее

формализации и задание системе структуру и логику системы.

- **Ввод, хранение и управление данными.** Это специальная база данных, в которой все данные объединяются в проекты.
- **Моделирование.** Здесь, в качестве моделирующего ядра выбирается язык GPSS World. Этот язык является наиболее сильным, проверенным и популярным во всем мире.
- **Вывод и анализ информации.** Возможен вывод результата имитации тремя способами – анимированный, графический, табличный.
- **Планирование экспериментов и оптимизации.** В нынешнее время используются как процедуры GPSS World по планированию эксперимента и оптимизации, так и собственные разработанные программы.

Данное программное обеспечение должно учитывать недостатки, созданных на сегодняшний день систем визуализации и отвечать некоторым требованиям:

- Независимость от имитационной модели.
- Возможность адаптироваться к предметной области.
- Возможность создать сложные графические объекты на основе геометрических примитивов, соответствующих области исследования.
- Отображать динамику поведения модели в 3D (анимацию, динамическое изменение текстур).
- Кроссплатформенность (Windows/Unix/Mac OS).
- Отсутствие необходимости устанавливать дополнительные программные обеспечения. Например, библиотеку и т.д.

Если смотреть со стороны визуализации, степень адаптации системы ИМ к предметной области определяется ее способностью отобразить все нужные ей объекты, но это невозможно при использовании ограниченных библиотек графических объектов. Появляется проблема обеспечить функциональность, позволяющую неподготовленному пользователю эффективно создавать графические объекты для исследуемой области. Наиболее успешным направлением решения этой проблемы является описание элементов визуализации на основе онтологии, где поведение и свойства созданных объектов будут инкапсулированы в объектах иерархии онтологий (рис. 1).

Онтология приложений для описания конкретных объектов, используемых для моделирования исследуемой области, которые находятся на сцене и обладают рядом параметров, таких как: используемая визуальная модель, тип поведения, возможные визуальные эффекты.

Каждому объекту предметной области будет соответствовать графический объект, который еще сильнее отображает все его основные свойства. Соответственно, каждый графический объект может поддерживать несколько разных анимаций, которые могут изменять определенные свойства объекта. Каждый экземпляр анимации несет ссылку на ее тип из общей таблицы анимаций и на характеристики объекта, которые будут изменены в процессе анимации. Свойства объекта также могут изменяться вследствие определенных событий или действий, предусмотренных для совершения данным объектом.

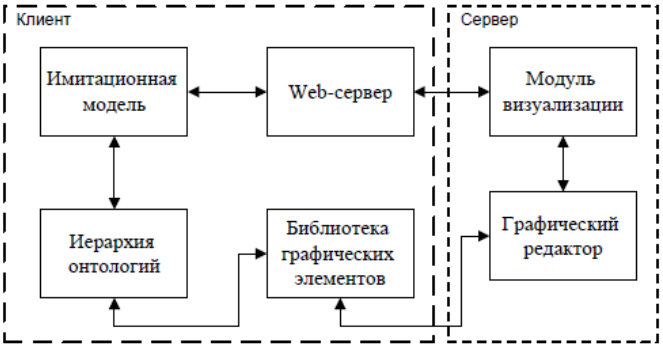

Рис. 1. Общая архитектура программного средства

III. Архитектура программного средства

Общая система архитектуры программного средства составлена в соответствии с представленными выше основными принципами разработки (рис. 1).

Программное средство можно разделить на две составляющие:

- Серверные компоненты.
- Клиентские компоненты.

Эти части архитектуры так же состоят из нескольких компонентов. Рассмотрим их подробнее.

Серверные компоненты:

- Web-сервер.
  Он позволяет производить обмен данными между модулем визуализации и имитационной моделью.
- Библиотека графических элементов.
  «Хранилище» графических моделей и примитивов, которые были созданы пользователями. Впоследствии их можно использовать для создания новых объектов, меняя структуру.

Клиентские компоненты:

- Модуль визуализации.
  Модуль содержит в себе созданные пользователем графические объекты и с помощью них создается визуализация – создание трехмерной сцены. Трехмерную сцену можно отобразить с помощью любого браузера.

- Графических редактор.
  Пользователь создает любые графические объекты (от легкой степени сложности до очень высокой) на основе геометрических примитивов (полукруг, квадрат, конус, прямоугольная пирамида и т.д.)

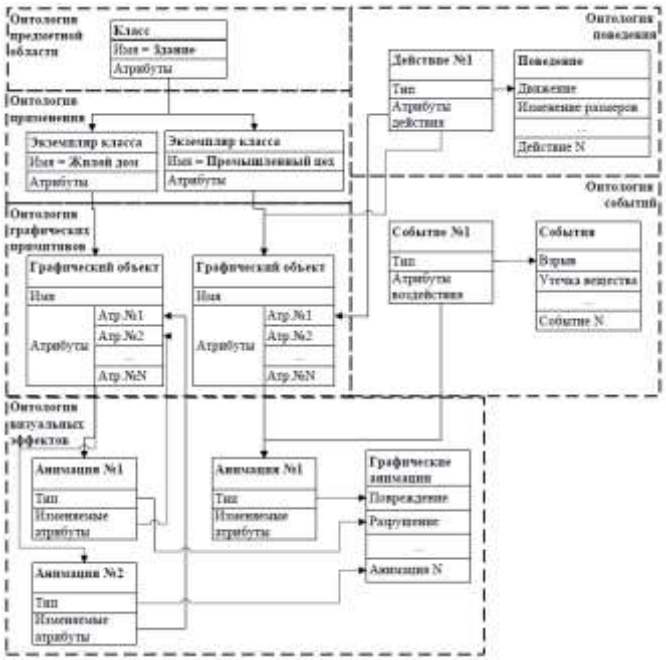

Рис. 2. Схема взаимодействия онтологий

IV. Предлагаемая реализация компонентов программного средства

Программное средство будет реализоваться с помощью браузера в виде web-приложения. Не будет затрат, не будет сложностей с установкой, обновлением и поддержкой программного средства, что немаловажно.

Классическая клиент-серверная архитектура обладает существенным недостатком – синхронным обменом данными, т.е. необходимостью для клиента ожидать ответа от сервера. В случае визуализации результатов имитационного моделирования, когда может потребоваться интенсивный обмен больших массивов данных, может возникнуть существенное снижение производительности модуля визуализации. Для решения этой проблемы предлагается использовать протокол WebSocket, который осуществляет асинхронный обмен сообщениями между браузером и веб-сервером в режиме реального времени. При такой реализации, имитационная модель и модуль визуализации будут выступать WebSocket-клиентами. Результаты имитационного моделирования будут поступать на WebSocket-сервер, который затем адресно перенаправит их на пользовательские терминалы, которые будут осуществлять визуализацию результатов.

WebSocket-клиенты исполнены на языке Java и представляют собой программный интерфейс обмена данными с WebSocket-сервером.

Для того, чтобы установить соединение с сервером, потребитель должен сформировать GET-запрос и с помощью Handshake-запроса клиенту отвечает сервер, это означает, что канал обмена данными готов к использованию. Каждому клиенту после подключения для идентификации присваивается уникальный код ID. Также, каждый клиент указывает свое собственное имя и тип, характеризующий его роль в процессе обмена данными. Предлагается использовать 3 типа реализации клиентской части WebSocket (рис. 3):

- DataSource-клиент предназначен только для передачи данных на сервер или другому клиенту, а также получению сообщений от сервера в качестве обратной связи. Наиболее яркий пример использования такой реализации – клиентская часть, встраиваемая в средство имитационного моделирования.
- Terminal-клиент предназначен только для приема сообщений, как от самого WebSocket-сервера, так и данных от других клиентов. Примером использования реализации является клиентская часть, встраиваемая в модуль визуализации результатов моделирования.
- Crossbreed-клиент – гибридный тип клиента, позволяющий как принимать сообщения от других клиентов, так и передавать их.

На данный период времени есть множество графических движков, таких как: CryEngine, Unity, UDK - они позволяют формировать трехмерные стены. Для визуализации, в качестве основы, предлагается использовать WEBGL (WEBGL — это программная библиотека, которая позволяет реализовать на JavaScript интерактивную графику). Реализовать клиентскую часть в виде web-страницы позволяет то, что WEBGL поддерживается всеми основными браузерами.

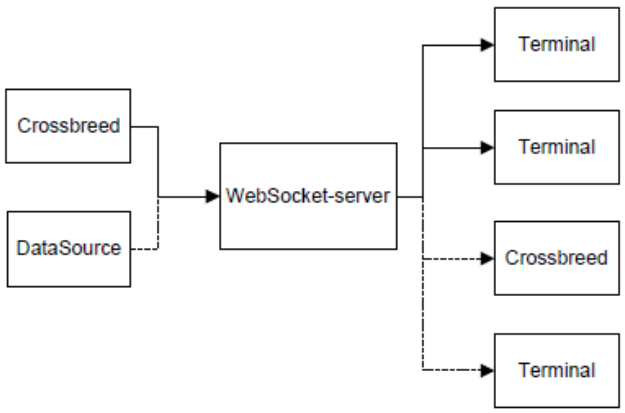

Рис. 3. Схема обмена данными между разными типами клиентов

V. Анализ программных средств, направленных на обработку и визуализацию статистических данных

Актуальность исследования заключается в необходимости анализа, структурирования сложных

статистических данных и преобразования их в легко воспринимающиеся доступные формы (графики, диаграммы) для принятия решений, связанных с управлением и организацией процессов.

Основной задачей исследования является анализ и выбор программного продукта для обработки данных и представления их в графической форме. Формальная постановка задачи заключается в необходимости выбора лучшей альтернативы (программного продукта). Выбор осуществляется на основе расчета математического метода поддержки принятия решений – метода анализа иерархий. Суть метода анализа иерархий (МАИ) заключается в попарном сравнении альтернатив по отношению к некоторому критерию. Проанализировав научную литературу возможно сформулировать качества подходящего программного средства. Оно должно содержать большое количество модулей анализа данных, эффектных диаграмм и графиков, расширение системы, доступ к другим форматам и обладать возможностью экспорта результатов графики. Далее необходимо выбрать программное средство методом анализа иерархий.

Метод аналитической иерархии использует множество различных критериев, которые в свою очередь включают в себя другие критерии (более общие критерии, разделяющиеся на критерии частного характера). Каждые группы критериев имеют свои коэффициенты важности. С целью определения критериальной ценности, альтернативы сравниваются между собой по отдельным критериям.

ТАБЛИЦА I АЛЬТЕРНАТИВЫ И КРИТЕРИИ ВЫБОРА

Критерии / Альтернативы	Разработчик	Цена, $	Платформа	Возможности расширения системы	Графика	Пользователь	Импорт/Экспорт данных
MC Excel	Microsoft	150	W	Virtual Basic	2D, 3D	H-M	Отсут.
SPSS	SPSS, Inc.	980 / 1102	W, M, L	Скрипты на языке SPSS	2D, 3D	H	Присут.
Statistika	StatSoft, Inc.	999	W	STATISTICA Visual Basic C++, Java, VB	2D, 3D, 4D	H-M	Присут.
Statis	ППО	500	W	Макросы	2D, 3D	H-M	Присут.
Attestat	ИЛ. Гайдамакин	0	W	Virtual Basic	2D, 3D	H-M	Отсут.
Systat	SPSS, Inc.	995	W	Макросы	2D, 3D	H	Присут.
Matlab	MINITAB Inc.	895	W, M	Макросы	2D, 3D	M-L	Присут.

Пользователь – квалификация типичного пользователя: **H (High)** – статистик-профессионал; **M (Middle)** – «есть базовые статистические знания»; **L (Low)** – «отсутствие базовых знаний»; **H-M** – промежуточный вариант.

Платформа – операционная система: **W (Windows)**; **M (Mac)**; **L (Linux)**.

Метод аналитической иерархии складывается из выполнения следующих этапов: 1) структурирование в виде иерархической структуры задач принятия решений (цели-критерии-альтернативы); 2) осуществление попарного сравнения элементов каждого уровня (результаты в виде чисел) человеку, который принимает решение (ЛПР); 3) вычисление весовых коэффициентов для элементов каждого уровня 4) вычисление количественной оценки качества каждой из альтернатив по следующей формуле:

$$U\left(X_k\right)=\sum_{i=1}^{m} a_i y_i\left(X_{k)}\right.$$

и определение наилучшей альтернативы по формуле:

$$X^{*}=argmax_{x_k \in X}\, U(X_k)$$

Применив данный метод для решения поставленной задачи получен результат означающий, что для принятия решений о целесообразности приобретения программного средства между всеми рассматриваемыми альтернативами

$X_i \in X, i=1(1)7$ выявлены следующие отношения строгого предпочтения: $x_3 > x_2 > x_4 > x_1 > x_5 > x_7 > x_6$.

Проведя краткий обзор и проанализировав ситуацию, выбранным программным продуктом становится удовлетворяющий всем требованиям – пакет Statistika. Также есть некоторые предпочтения и пожелания: МАИ можно сделать более гибким и учитывающим предпочтения ЛПР. К примеру, предложить множество способов оценки весовых коэффициентов, и применяя эти способы решить задачу. Далее предложить ЛПР рассмотреть множество решений с возможностью выбора, подходящего для него.

По итогам исследования новыми являются следующие положения и результаты. Предлагаемый подход к генерации графических объектов на основе онтологий, позволит создавать графические объекты, соответствующие исследуемой области, не прибегая к помощи сторонних графических редакторов, требующих значительное время на их освоение. Предлагаемые модели онтологий, позволяют дополнить существующее описание предметной области правилами графического отображения объектов. Использование WebSocket-технологии для обеспечения обмена данными при визуализации результатов имитационного моделирования, позволяет решить проблему существующих систем визуализации, связанную с передачей больших объемов данных.

Необходимо отметить, что предлагаемая концепция обладает рядом недостатков, в частности, необходимостью использования геометрических примитивов в качестве основы для новых объектов, что накладывает определенные ограничения на процесс их создания.

СПИСОК ЛИТЕРАТУРЫ

[1] Звягин Л. С. Современные проблемы динамики социально-экономических систем в фокусе производственных отношений и управляющего развития // Теоретическая экономика. 2018. № 4 (46). С. 76-81.

[2] Звягин Л. С. Комплексная оценка безопасности функционирования моделей экономических систем // Экономика и управление: проблемы, решения. 2017. Т. 4. № 1. С. 18-25.

[3] Звягин Л. С. Системы распределенного моделирования и методы управления модельным временем // Экономика и управление: проблемы, решения. 2018. Т. 1. № 10. С. 4-11.

[4] Комарцова Л. Г., Максимов А. В. Нейрокомпьютеры: Учеб. пособие для вузов. 2-е изд., перераб. и доп. М.: Изд-во МГТУ им. Н. Э. Баумана, 2004. 400 с.

[5] Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечеткие системы: Пер. с польск. И. Д. Рудинского. М.: Горячая линия -Телеком, 2006. 452 с.

[6] Фролов А. А., Муравьев И. П. Информационные характеристики нейронных сетей. М.: Наука, 2005, 160 с.