Перспективы разработки аналогов пакета SIMULINK

Куракин Павел Вячеславович

Научный сотрудник, Институт проблем управления им. В.А. Трапезникова РАН

117997, Россия, г. Москва, ул. Профсоюзная, 65

pvkurakin@yandex.ru

Малинецкий Георгий Геннадьевич

доктор физико-математических наук

Заведующий отделом, Институт прикладной математики им. М. В. Келдыша РАН

125047, Россия, г. Москва, Миусская площадь, 4

gmalin@keldysh.ru

Митин Николай Алексеевич

кандидат физико-математических наук

Ведущий научный сотрудник, Институт прикладной математики им. М.В. Келдыша РАН

125047, Россия, г. Москва, Миусская площадь, 4

Mitin@keldysh.ru

Статья из рубрики "Методы, языки и модели человеко-машинного взаимодействия"

Аннотация.

Объектом исследования статьи являются необходимость и возможность разработки ресурсами академической среды, то есть на некоммерческой основе, специализированных графических редакторов, обеспечивающих среду визуального проектирования экземпляров прикладных математических задач тех или иных типов. Рассмотрены имеющиеся на рынке программные продукты, а также их расширения, примерно соответствующие указанной задаче. Показано, по каким причинам эти средства оказались неподходящими для тех прикладных задач или тех заказчиков, с которыми пришлось ранее иметь дело авторам. Метод исследования основан на сопоставлении функциональных возможностей имеющихся на рынке программных средств и их расширений, а также имеющихся у автором заделов, с теми прикладными задачами, на которых авторы делают акцент. Новизна проведенного исследования и сделанных выводов состоит в предложении разделить задачу визуального проектирования и автоматической генерации вычислительного кода, как это принято делать в существующих продуктах. Имеет смысл передать создание вычисляющего кода на прикладного математика, работающего с государственным заказчиком, но при этом организовать эффективное разделение труда между заказчиком и математиком. При таком угле зрения задачи графического редактора упрощаются. С другой стороны, такой редактор можно сделать более гибким.

Ключевые слова: графический редактор, математические расчеты, веб браузер, визуальное проектирование, пользовательский интерфейс, математический граф, МАТЛАБ, Simulink, JavaScript, JSON

DOI:

10.7256/2454-0714.2018.4.27078

Дата направления в редакцию:

15-08-2018

Дата рецензирования:

17-08-2018

Работа была поддержана грантами РФФИ 16-01-00342 и 18-511-00008

Введение

Ранее авторы сообщали [1, 2, 3] об оригинальной архитектуре специализированных вычислительных комплексов, разработанных ими в ходе ряда государственных контрактных работ в космической отрасли.

Проблему, которая решалась этой разработкой, можно кратко, на качественном уровне, описать следующим образом. Требуются программные средства, аналогичные известной связке программных пакетов MATLAB + Simulink [4, 5, 6], при этом в системе должно быть некоторое хранилище данных и описаний задач. Вычислительный комплекс должен быть основан только на бесплатном (свободно распространяемом) программном обеспечении. В целом, конструкции такого рода оказались необходимы в системах поддержки принятия решений [7].

В статье излагаются перспективные предложения по развитию интерфейсного слоя указанной программной архитектуры.

1. Постановка задачи

Напомним: система Simulink является расширением пакета MATLAB и позволяет создавать описание задачи визуально, при помощи встроенного графического редактора. Предполагается, что графически задача выглядит как система *блоков* разных типов, которые связаны *линиями* , также разных типов. Согласно [6], существуют следующие типы блоков:

· - «Источники»: используются для генерации всевозможных сигналов;

· - «Стоки»: используются для вывода или отображения сигналов;

· - «Дискретные (элементы)»: линейные элементы дискретного времени (передаточные функции, модели пространства состояний, и т.п.);

· - «Линейные (элементы)»: линейные элементы систем непрерывного времени (суммирующие соединения, линейные усилители, и т.п.);

· - «Нелинейные (элементы)»: нелинейные операторы (произвольные функции, линии насыщения и задержки)

· - «Соединения»: мультиплексоры [8], демультиплексоры [9], и т.п.

Наш опыт показал, что использовать или адаптировать данный набор типов блоков можно для описания далеко не всяких типов задач, особенно для задач, характерных для систем поддержки принятия решений, с которыми мы столкнулись. Попросту, эти блоки характерны только для определенного класса научно – технических задач и математических методов их решения – обширного, но в известной мере стандартного. Мы бы сказали, что Simulink, в целом, ориентирован на готовые парадигмы и методы математического моделирования, а пользователь изначально работает, в математическом смысле, на языке этих парадигм и методов.

Пакет MATLAB – коммерческий. Сама по себе эта проблема легко преодолима для специалистов, самостоятельно занимающихся программированием «вручную», поскольку существует бесплатный аналог этой программы – система Octave [10, 11]. Однако, Simulink работает именно с MATLAB. Насколько известно авторам, производители Octave не поставляют к нему системы аналог Simulink. Однако существуют визуальные разработки других производителей, способные подключаться и использовать Octave (см. далее).

Существует альтернативные MATLAB системы математических расчетов, снабженные графическим редактором, например, FlowDesigner [12]. Разработчики в документации на сайте [12] заявляют об этой системе как аналоге Simulink: «FlowDesigner – свободно распространяемая (лицензия GPL/LGPL) среда разработки, ориентированная на потоки данных. Ее можно использовать для создания сложных приложений путем комбинирования небольших типовых (reusable) строительных блоков. В некотором отношении, эта среда аналогичная системам Simulink и LabView (см. далее), но вряд ли ее можно назвать клоном любой из них». Далее мы снова видим вроде бы широкий, но достаточно жестко очерченный круг задач, где ее можно применять:

· - Обработка сигналов;

· - Обработка звуковых сигналов (и произвольных цифровых сигналов);

· - Векторное квантование;

· - Нейронные сети;

· - Нечеткая логика;

· - Обработка звуковых сигналов в режиме реального времени (бета-версия);

· - Линейная алгебра;

· - Плагин Octave (бета-версия);

· - Робототехника.

Понятно, что эти задачи очень важны, но мы снова видим в той или иной мере «стандартные», широко распространенные (в прикладной науке, на производстве) задачи. Для этих задач имеется массивный наработанный математический инструментарий их решения, и графический редактор ориентирован на понятийный аппарат этого инструментария.

Еще одна альтернатива MATLAB - бесплатная система математических расчетов Scilab [13]

и расширяющая его система графического проектирования и моделирования Scicos [14]. Согласно [14], «Scicos– независимое программное приложение для моделирования, но доступ к Scilab и его функциональным возможностям обеспечивает большую гибкость и расширяет диапазон возможностей моделирования». О системе Scicos сообщается как о системе моделирования динамических систем, причем основной акцент в системе сделан на моделировании больших потоков данных. Это снова говорит об определенной направленности всего инструментария в целом. Этот небольшой обзор имеющихся средств визуального проектирования позволяет понять, чем была вызвана к жизни описываемая разработка. Авторами двигала идея создать доступный программный инструмент специализированных математических расчетов для некоторых нишевых задач, снабженный удобным средством визуального проектирования. Из обзора становится понятно, что речь шла именно о нишевых задачах, так как для их визуального конструирования типовые средства, как оказалось, подходят плохо. Можно указать два типа таких задач:

1. Моделирование производства в космической отрасли в смысле потоков стоимостей и поставок комплектующих между предприятиями.

2 . Моделирование логистики дальних космических полетов в условиях нескольких параллельных стартов.

Согласно [15] система Labview (также [16]), упомянутая выше - "это среда разработки и платформа для выполнения программ, созданных на графическом языке программирования «G» фирмы National Instruments (США)...LabVIEW используется в системах сбора и обработки данных, а также для управления техническими объектами и технологическими процессами". Можно сказать, что в лице Labview мы снова имеем дело скорее с некоторым стандартным типом (статистических и технологических) задач.

На данный момент можно констатировать, что по итогам проведенных работ создан и опробован в космической отрасли действующий прототип программного комплекса задуманного типа – на основе исключительно свободно распространяемого ПО. Комплекс позволяет визуально редактировать экземпляры задач соответствующих типов, отправлять задание на расчет в вычислительный блок, загружать из хранилища готовые решения и вспомогательные данные.

Архитектура комплекса организована по слоям:

1 . - слой пользовательского интерфейса - графический конструктор, основанный на возможностях браузерного программирования (HTML+ JavaScript);

2. - слой вычислительного блока и хранилища данных;

3. - промежуточный слой на основе веб-сервера осуществляет взаимодействие первого и второго слоев (несущих основную функциональную нагрузку).

На рис. 1 приведено схематическое изображение этой архитектуры (в том или ином виде мы уже приводили эту схему в предыдущих работах). Надо сказать, что такая архитектура, хоть мы и пришли к ней независимо, в настоящее время уже не является оригинальной. Насколько известно авторам, на таких же принципах и клиент-серверной архитектуре основана система интерактивных вычислений IPython [17, 18], но эта система не предполагает визуального проектирования, на интерфейсной стороне там имеется стандартный текстовый ввод команд.

На данный момент уместно говорить именно об имеющемся прототипе или макете желаемого комплекса. В силу этого каждый слой также имеет статус прототипа или макета. Описанные работы были выполнены, как отмечалось выше, в рамках государственных договорных работ. Поэтому возникает вопрос о возможных путях самостоятельного развития имеющихся заделов.

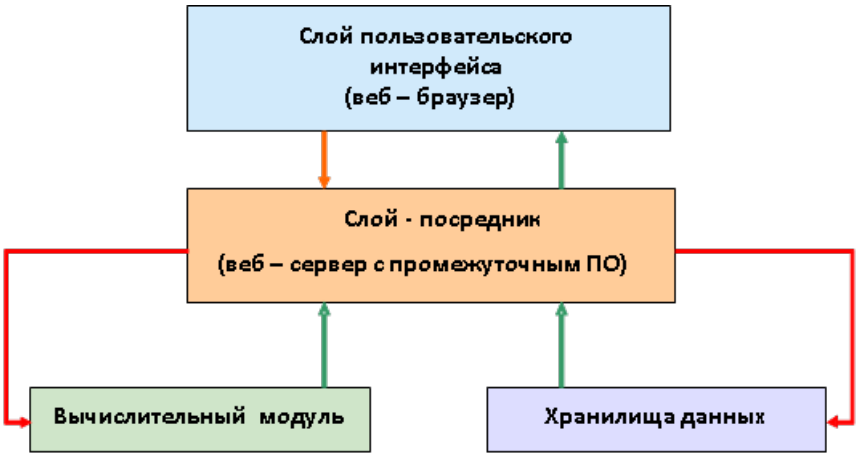

Рис. 1.

Программная архитектура вычислительных комплексов на основе браузерного графического редактора

2. Пути развития программного комплекса и его компонентов

В настоящее время авторы поставили для себя задачу продолжить описанные работы, но при этом сосредоточиться на автономном развитии только одного модуля данного комплекса, потому что считают, что такая задача имеет самостоятельную ценность и хорошую перспективу.

А именно – в настоящее время имеет смысл развивать в первую очередь слой пользовательского интерфейса, то есть – графический редактор. Развернуто, такой редактор можно определить, как среду визуального конструирования экземпляров вычислительных задач того или иного класса.

Выше были указаны типы задач, для которых был разработан имеющийся прототип. На данный момент реализовано две версии графического редактора, адаптированных, соответственно, для двух родственных типов задач [1, 2, 3]. В обоих случаях базовым математическим объектом, который используется для формирования описания экземпляра задачи, является математический граф. Полное описание экземпляра задачи (далее – *конфигурация* задачи) представляет собой совокупность связанных друг с другом модулей. Каждый модуль также представляет собой конфигурацию некоего относительно простого объекта. Эти объекты представляют собой узлы графа, они могут быть разных типов. Конфигурация объекта каждого типа – та или иная структура данных (в простейшем случае – просто перечень численных параметров). Объекты – узлы графа связаны между собой объектами связей (ребра графа). Связи между узлами фактически представляют собой самостоятельный тип объекта, у которого также есть своя конфигурация. Все конфигурации можно редактировать – задавать значения параметров в соответствующей структуре данных объекта.

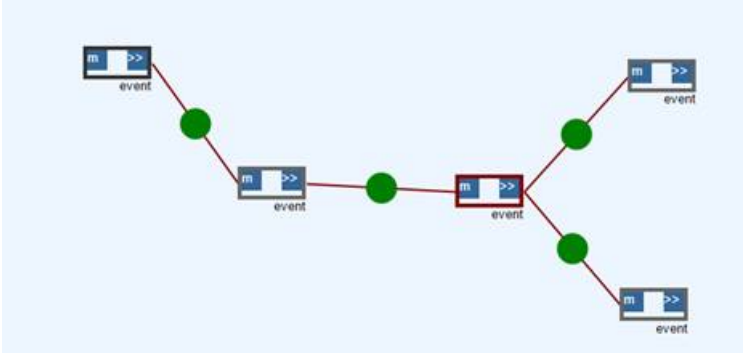

Рис. 2. Представление типовой конфигурации задачи в среде браузерного графического редактора

На Рис. 2 (который мы повторяем из [3]) приведен фрагмент конфигурации для задачи, в которой узлами графа являются «события» - речь шла о моделировании логистики пилотируемого полета на Луну. Ребра являются «переходами» между событиями. В другой задаче граф внешне выглядел так же, но узлы графа представляли собой предприятия космической отрасли, а ребра – поставки той иной продукции. В первой задаче проводилась оценка общей выполнимости того или иного маршрута доставки космических модулей на Луну. Во второй задаче проводились некоторые экономические расчеты. Кроме этого, для второй задачи было разработано еще одно «полотно» редактора, в котором пользователь системы мог задавать субъектную принадлежность предприятий (какое предприятие входит в какой холдинг, объединение, концерн и т.п.).

Как видим, содержательно решаемые задачи могут значительно отличаться, однако графическое представление задач, как оказывается, чаще всего имеет один и тот же общий вид, одинаковую структуру. Можно заметить, что все упомянутые графические редакторы – как наш, так и «фирменные» - опираются на одну и ту же базовую структуру данных – математический граф. Во всех случаях мы имеем, так или иначе, систему тех или иных «блоков», связанных теми или иными «линиями» (мы привыкли называть их «стрелками»). Вопрос в том, какое наполнение получают эти блоки и линии в каждом приложении.

В [1, 2, 3] перечислены специализированные библиотеки открытого доступа (написанные на языке браузерного программирования JavaScript), примененные при создании графического редактора.

Авторы берутся утверждать, что имеющийся задел в разработке браузерного графического редактора фактически представляет собой заготовку для нового программного продукта. Следует продолжить разработку имеющегося редактора, немного сместив первоначальные акценты. На данный момент редактор, в основном, умеет «рисовать» графы из «квадратиков» и «стрелок». Можно добавлять и удалять квадратики (узлы графа), соединять их «стрелками» (ребра графа). Каждый «квадратик» и каждая «стрелка» могут быть того или иного типа, и, в соответствии с этим типом, иметь свою конфигурацию – т.е. набор параметров, описывающих данный объект.

Полученная совокупная конфигурация изначально хранится в структурах данных языка JavaScript (встроенный язык всех браузеров), затем она может быть преобразована в структуры данных в формате JSON [19]. Этот формат оказался удобен тем, что уже давно существуют (бесплатные) библиотеки (на встроенном языке программирования пакета MATLAB / Octave) для конвертирования структур данных в формате во внутренний формат MATLAB / Octave.

В описанном выше программном комплексе конфигурация задачи, созданная пользователем в графическом редакторе браузера, через слой – посредник (веб - сервер) передается в вычислительный слой (Octave) для проведения вычислений. Предполагается, что на стороне вычислительного модуля уже имеется готовая программа (на языке программирования Octave) для проведения вычислений. В этом отличие нашей системы от упомянутых выше продуктов типа Simulink – там автоматизация значительно более глубокая: на основе визуальной конфигурации производится автоматическая генерация кода (на соответствующем языке) для вычислений.

Важно понять: стоит ли добиваться столь же глубокой автоматизации функций нашего программного средства при разработке с использованием тех ресурсов, которыми располагает научная академическая среда? На наш взгляд, ответ на этот вопрос должен быть гибким. С одной стороны, ясно, что в общем случае автоматическая генерация кода - слишком большая задача, посильная серьезным рыночным компаниям. Наша задача более скромная, но и она оставляет достаточно места для самостоятельной разработки. Наша работа, как отмечено выше, изначально была рассчитана на нишевые, а не универсальные задачи. Поэтому в общем случае не стоит требовать от такой системы автоматической генерации программного кода для математических вычислений.

С другой стороны, напомним: до сих пор речь шла о довольно большой, составной системе. В качестве вычисляющего модуля в ней используется профессиональный математический пакет (Octave). Разделение пользовательского интерфейса и вычислительного модуля оправдано при наличии сложных и объемных вычислительных алгоритмов. Однако, в случае относительно простых вычислений вовсе необязательно подключать такие массивные программные средства, поскольку довольно много вычислений можно проводить непосредственно в среде браузера, непосредственно на языке JavaScript: в последние годы интенсивно развивается разработка JavaScript - библиотек для математических вычислений. Несколько таких библиотек описано в [20]. Например, с библиотекой Numeric JavaScript можно решать дифференциальные уравнения. В случае относительно простых вычислений, вероятно, имеет смысл ставить задачу автоматической генерации кода. Более того, в этом случае уместно говорить о самостоятельном программном продукте, который содержит не только редактор экземпляров задач, но и непосредственно вычислительный блок. Разумеется, эта ниша на рынке не пустует, и такие программные решения тоже присутствуют - например, библиотека Tangle, также описанная в [20]. В приложениях, использующих эту библиотеку, можно «нарисовать» задачу и тут же получить необходимые расчеты. Важно понимать, что никакой продукт не является универсальным.

В нашем случае мы имеем в виду того пользователя, с которым знакомы из своей практики: это государственный заказчик, которому нужна некая система моделирования и математических расчетов для каких-то типовых для него задач. Такой заказчик может найти прикладного математика, чтобы тот разработал для него вычислительный алгоритм для задач этих типов. Однако наш заказчик, в силу характера его работы, хотел бы формировать экземпляры задач и наборы входных данных (для уже имеющегося вычислительного алгоритма или набора алгоритмов) многократно и самостоятельно, не прибегая постоянно к помощи математика.

Конечно, наш пользователь не может и не должен действовать совершенно независимо от математика. В частности (и в первую очередь), пользователю следует согласовать с математиком формат входных данных. На самом деле, в том и состоит задача предлагаемого графического редактора, чтобы эффективно *разделить труд* между

пользователем описанного типа и прикладным математиком, организовать их взаимодействие.

С другой стороны – до сих пор мы об этом не упоминали – наш пользователь хочет получить на выходе (уже после обработки входных данных вычислительным блоком) не просто числовые ряды, а результаты, также представленные в графическом виде. Современные библиотеки JavaScript это тоже позволяют, например, библиотека Raphael [21]. Мало просто вывести на экран тот или иной график, важно эффективно управлять структурой этого вывода – где, какой график, с какими осями, оси линейные или логарифмические, и т.п. Именно такие задачи призван решать в первую очередь графический редактор, который мы хотим получить.

Исходя из этого, сформулируем приоритетные задачи разработки и развития имеющегося графического редактора.

В первую очередь , следует добиваться максимально возможной гибкости и универсальности самого графического редактора. Необходимо постепенно расширять набор типов задач, экземпляры которых можно создавать в редакторе. Если речь идет – как сейчас – о задачах, в которых общее описание представляет собой математический граф, то должны настраиваться конфигурационные параметры объектов (узлов графа) и связей (ребер графа), а также способ объединения всех этих данных в общую конфигурацию. Инструментом этой настройки должен быть некий формальный, но простой и удобный язык описания структуры («разметки») той «картинки», которую создает редактор. Пользователь должен быть в состоянии изучить этот язык и самостоятельно создавать, фактически, разные редакторы для разных типов задач. О задачах новых типов будет сказано далее.

Во вторую очередь , следует заняться разработкой аналогичного языка форматирования для графического отображения результатов расчетов. Напомним, выше мы подчеркнули, что в общем случае мы не предполагаем немедленного выполнения расчетов на основе входных данных – наша первоочередная задача сформировать «эргономичным» способом набор входных данных для вычислительного блока в предположении, что нам известно, на каком средстве и языке программирования работает партнер пользователя – прикладной математик. Но, если нам уже известно, в каком виде наш математик представит нам результаты своих расчетов, мы вправе сформировать структуру графического вывода этих результатов.

И только *в последнюю очередь* , как довольно дальнюю перспективу, имеет смысл ставить задачу автоматической генерации кода. Мы считаем, что, скорей всего, это имеет смысл в тех случаях, когда вычислительный блок совмещен с графическим редактором (использование библиотек типа указанной выше Numeric JavaScript).

3. Примеры новых типов задач для графического редактора

В качестве примера задачи, заслуживающей автоматизации проектирования ее экземпляров, можно указать задачу моделирования макроэкономической динамики страны, изложенную в [22]. С формальной точки зрения эта математическая модель представляет собой систему большого (порядка 50) числа нелинейных алгебраических уравнений вида

$$x(i, t) = f\,(x(1,t), x(2, t), \ldots x(i-1, t), \ldots x(i+1, t), \ldots, x(N, t), a(1, t), \ldots a(M, t)) \quad (1).$$

Здесь x(i,t) - значение *i* -й динамической переменной модели в дискретный момент

времени t , a(j, t) - значение j -го параметра модели в этот же момент времени. В качестве как переменных, так и параметров модели выступают различные макроэкономические показатели, но временная (по годам) зависимость части из них предполагается заданной как прогнозы экспертов, временная зависимость остальных показателей предполагается подлежащей расчету. Иными словами, по заданным a(j, t) нам надо вычислить x(i,t).

Авторы [22] вели расчеты методом сжимающих отображений. В принципе, можно использовать другие алгоритмы, опираясь на библиотеки численных методов, которые имеются в наличии для того программного средства, на котором работает сотрудничающий с нашим пользователем математик. Либо, математик может разработать свой собственный численный метод. Лишний раз напомним, что наша задача – эффективно разделить труд между нашим пользователем и прикладным математиком.

Как должен работать, с визуальной точки зрения, графический интерфейс, при помощи которого пользователь задает экземпляр такой задачи? В данной задаче уже нельзя говорить, что структура модели представляет собой математический граф, во всяком случае – обычный граф. Уравнение (1) записано в общем виде, а конкретные зависимости из [22] как правило, довольно простые, включают не более 3-4 переменных с простыми алгебраическими операциями (сложение, умножение, деление). Каждая переменная зависит одним уравнением от небольшого числа других переменных и параметров, но эта зависимость в общем случае не является симметричной, т.е. мы, во всяком случае, не получим один ненаправленный граф.

С нашей точки зрения, в данной задаче не имеет большого смысла пытаться показать пользователю все описание задачи (уравнения) целиком. В каком-то смысле это и так сделано в текущей реализации задачи. В авторском исполнении вся модель реализована в среде Excel. Это удобно с той точки зрения, что уравнения, исходные данные и получаемые результаты расчетов в виде графиков содержатся в одном файле. Но это не делает описание задачи более обозримым, а работу с ним более технологичным.

Мы предлагаем следующее решение. Для каждой переменной графический редактор должен отображать две группы связей. Каждая переменная, во-первых, зависит от каких-то других переменных. Типичное уравнение из модели [22]: CN2(t) = n2(t)*(XO(t) + IM(t)) (2): чистые налоги на производство и импорт CN2 пропорциональны сумме выпуска XO в основных ценах импорта IM, n2 - ставка налогообложения. Переменная CN2 зависит от трех других переменных (n2, XO, IM).

С другой стороны, от данной переменной CN2 также зависят какие-то другие переменные: например, для номинального ВВП страны WWP в [22] имеется уравнение WWP(t) = WPR(t) + OT(t) + CN2(t) (3), где WPR - произведенный ВВП, OT - оплата труда. Поэтому, вторая группа связей, которую имеет смысл показывать в графическом редакторе для выбранной переменной – это все те переменные, которые в своих уравнения зависят от данной.

На Рис. 3 приведен примерный вид графа, соответствующего первой группе связей для переменной CN2.

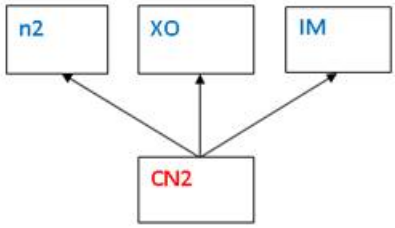

Рис. 3 Граф зависимости выбранной переменной CN 2 от других переменных.

При клике на квадратике CN2 должен открываться текстовый редактор для вывода и изменения формулы (2). Внешне, отображаемая в редакторе картинка значительно упрощается, но идея в том, что пользователю желательно иметь возможность навигации от одной переменной к другой. Кликнув по квадратику t2, мы должны получить аналогичный мини-граф для этой переменной, т.е. необходимо перемещаться от одной переменной к другой.

Второй граф должен показывать связь текущей переменной с теми переменными, которые, наоборот, зависят от данной - она входит в уравнения для этих переменных, при этом мы имеем уже не одно уравнение, а много. При клике по квадратику переменной должен редактор уравнения для переменной, соответствующей квадратику (напомним, в это уравнение входит текущая переменная).

Также, пользователю надо предоставить линейный список (или таблицу, для лучшего обзора) всех переменных – чтобы сразу переходить к произвольно выбранной переменной.

Мы ожидаем, что в целом, при наличии большого числа переменных и параметров, структура описания экземпляра оказывается более простой, компактной и обозримой, чем этого добились сами авторы модели [21]. В оригинальном исполнении вся модель реализована в среде Excel. Это удобно с той точки зрения, что уравнения, исходные данные и получаемые результаты расчетов в виде графиков содержатся в одном файле. Однако ясно, что такое решение не технологично. Мы считаем, что видим возможность «уложить» задачу описания экземпляра задачи такого типа в некую общую парадигму. Это будет соответствовать программным целям, заявленным в [23].

Мы говорим о развитии задела в области графического редактора как самостоятельной задаче. Однако, уместно вспомнить, что в конечном счете мы всегда имеем в виду полный цикл решения задач математического моделирования, то есть – составной программный комплекс. Функциональный графический редактор в совокупности с модулем вычислений и базой данных может оказаться полезным для реализации большого проекта, который сейчас выполняется в России. Это создание сети ситуационных центров, работающих по единому регламенту в субъектах Федерации и органах власти федерального уровня. Цель этого проекта – обеспечить наблюдаемость социально – экономических процессов и на этой основе повысить качество государственного управления.

Вероятно, при создании таких центров будет широко использоваться концепция когнитивных центров, предложенная в Институте прикладной математики им. М. В. Келдыша РАН [24]. Последние включают в себя инструменты для использования экспертного знания и системы математических моделей объектов управления. В таких центрах поддержка принятия решений очень часто связана с системным

проектированием, при этом очень важно наглядно представить взаимосвязи между экспертами, информацией от органов власти и других источников, их данные и оценки – всю «анатомию» подготовки и принятия решений. И сами субъекты Федерации, и их возможности очень разнообразны. Кроме того, важно системное окружение и взаимодействие с субъектами и объектами управления. Авторы полагают, что предлагаемый в данной работе подход позволяет решать возникающие в этой области задачи значительно легче, чем многие другие.

Другая область приложения графических редакторов предложенного типа связана с расширением возможностей когнитивного моделирования. В данной области традиционной моделью является ориентированный граф, вершины которого соответствуют факторам, а ребра – взаимосвязям между ними. В типичном случае по системе запускают импульс, соответствующий изменению одного из факторов, и смотрят: как это повлияет на все остальные факторы. Очевидно, что как вершины – факторы должны характеризоваться разными параметрами, так и ребра – взаимосвязи должны быть, вообще говоря, нелинейными. (С чем-то подобным ученые сталкивались, разрабатывая модели мировой динамики.) Однако развитие качественно – количественного подхода в этой области сдерживается тем, что модели такого типа трудно представить и нелегко фиксировать уже сделанные попытки. Предлагаемая система позволяет снять или облегчить решения ряда возникающих в этой области проблем.

Наконец, есть еще направление, в котором подобные системы могли бы очень помочь. Это теоретическая или математическая история. В этом направлении историческая реконструкция опирается на математические модели и на анализ альтернативных сценариев исторических процессов [25]. «Историческое расследование» опирается на междисциплинарный анализ большого круга источников, ставших доступными благодаря развитию телекоммуникаций. В этой области использование систем, опирающихся на графические редакторы предложенного типа позволит ученым «не утонуть» в массиве информации и учесть различный уровень достоверности доступных сведений.

Работа была поддержана грантами РФФИ 16-01-00342 и 18-511-00008.

Библиография

1. Зухба Р.Д., Куракин П.В., Малинецкий Г. Г., Махов С. А., Митин Н. А., Торопыгина С.А.Система моделирования «КОСКОН» как инструмент поддержки принятия решений в космической отрасли. Препринты ИПМ им. М. В. Келдыша. — 2015. — № 113. — 36 с. — URL: http://library.keldysh.ru/preprint.asp?id=2015-113.
2. П. В. Куракин. «Новая программная архитектура для специализированных систем математических расчетов». Информационные технологии и вычислительные системы, 2016 г, № 2 стр. 66 – 74.
3. Куракин П. В. «Специализированные системы математических расчетов нового поколения». Программные системы и вычислительные методы – 2016 г, №1(14), стр. 80 – 94.DOI: 10.7256/2305-6061.2016.1.17997.
4. Дьяконов В. П. Справочник по применению системы PC MATLAB. — М.: «Физматлит», 1993. — 112 с. — ISBN 5-02-015101-7.
5. Simulink (статья в Wikipedia): https://en.wikipedia.org/wiki/Simulink.
6. Simulink basic tutorial (https://classes.soe.ucsc.edu/cmpe242/Fall10/simulink.pdf).
7. Ларичев О. И., Петровский А. Б. Системы поддержки принятия решений. Современное состояние и перспективы их развития. Итоги науки и техники. Сер.

Техническая кибернетика. — Т.21. М.: ВИНИТИ, 1987, с. 131—164.

8. Мультиплексор (статья в Wikipedia), URL: https://ru.wikipedia.org/wiki/Мультиплексор_(электроника).
9. Демультиплексор (статья в Wikipedia), URL: https://ru.wikipedia.org/wiki/Демультиплексор.
10. Octave (статья в Wikipedia), URL: https://ru.wikipedia.org/wiki/GNU_Octave.
11. Octave (официальная страница проекта) , URL: https://www.gnu.org/software/octave/
12. FlowDesigner (официальная страница проекта): https://sourceforge.net/projects/flowdesigner/.
13. Е. Р. Алексеев, Е. А. Чеснокова, Е. А. Рудченко. Scilab: Решение инженерных и математических задач. Москва ALT Linux; БИНОМ. Лаборатория знаний. 2008 г.
14. С. Данилов. «SCICOS – пакет SCILAB для моделирования динамических систем»: http://www.tstu.ru/book/elib2/pdf/2011/danilov.pdf. (Документ выложен на сайте Тамбовского государственного технического университета).
15. Labview (статья Wikipedia), URL: https://ru.wikipedia.org/wiki/LabVIEW.
16. Официальная страница проекта Labview на русском языке, URL: http://www.labview.ru.
17. IPython (статья в Wikipedia), URL: https://ru.wikipedia.org/wiki/IPython.
18. IPython (официальная страница проекта), URL: https://pypi.org/project/ipython/
19. Официальная страница формата JSON на русском языке, URL: http://json.org/json-ru.html.
20. John Mueller, Four Serious Math Libraries for JavaScript (https://blog.smartbear.com/testing/four-serious-math-libraries-for-javascript/).
21. Официальный веб-сайт проекта Raphael, URL: http://raphaeljs.com/.
22. Антипов В.И., Митин Н.А., Пащенко Ф.Ф.Макроэкономическая имитационная модель развития России. Препринты ИПМим. М.В.Келдыша. 2017. № 142. 48 с. doi:10.20948/prepr-2017-142. URL: http://library.keldysh.ru/preprint.asp?id=2017-142.
23. Куракин П. В., Гусев В. Б. «Инструментальная поддержка моделирования динамики саморазвивающихся систем». Седьмая Всероссийская научно-практическая конференция «Имитационное моделирование. Теория и практика» (ИММОД-2015). Труды конференции. 21 – 23 октября. Под общей редакцией академика РАН С. Н. Васильева, чл.-корр. РАН Р. М. Юсупова 2015 г. Москва. Т. 1., стр. 63-67.
24. И. В. Десятов, Г. Г. Малинецкий, С. К. Маненков, Н. А. Митин, П. Л. Отоцкий, В. Н. Ткачев, В. В. Шишов. Когнитивные центры как информационные системы для стратегического прогнозирования.Препринты ИПМ им. М.В.Келдыша. 2010. № 50. 28 с. URL:http://library.keldysh.ru/preprint.asp?id=2010-50
25. Капица С.П., Курдюмов С.П., Малинецкий Г.Г. Синергетика и прогнозы будущего. М.: Едиториал УРСС, 2003. — 288 с. — (Синергетика: от прошлого к будущему). — ISBN 5-354-00296-6.