Министерство просвещения Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Южно-Уральский государственный гуманитарно-
педагогический университет»

А.Л. КОРОЛЕВ

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ В ИНСТРУМЕНТАЛЬНОЙ СРЕДЕ MODEL VISION STUDIUM (MVS)

Практикум

Челябинск
2022

УДК 681.4(021)
ББК 32.973:2-018я73
К 68

К 68 **Королев, А.Л. Компьютерное моделирование в инструментальной среде Model Vision Studium (MVS): практикум** / А.Л. Королев; Федеральное государственное бюджетное образовательное учреждение высшего образования «Южно-Уральский государственный гуманитарно-педагогический университет». – Челябинск: Изд-во Южно-Урал. гос. гуманитар.-пед. ун-та, 2022. – 116 с. – ISBN 978-5-907611-61-0. – Текст: непосредственный.

Практикум содержит учебную информацию, а также подробные методические указания по работе в среде MVS при построении компьютерных моделей.

Цель издания – формирование у обучающихся умений разрабатывать компьютерные модели объектов на основе современной методологии моделирования с использованием новых технологий в будущей профессиональной деятельности.

Данное пособие может использоваться в курсе «Компьютерное моделирование», «Моделирование систем для студентов бакалавриата, обучающихся по направлению: 44.03.05 «Педагогическое образование» по профилям: «Информатика – английский язык», «Математика–информатика», а также при разработке выпускных квалификационных работ. Практикум построен на доступном для образовательных целей программном обеспечении и может также использоваться в учебном процессе в рамках школьного курса «Информатика».

УДК 681.4(021)
ББК 32.973:2-018я73

Рецензенты: Т.В. Карпета, канд. физ.-мат. наук, доцент
Г.Б. Поднебесова, канд. пед. наук, доцент

ISBN 978-5-907611-61-0

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Моделирование является общенаучным методом изучения законов окружающего мира, свойств объектов и систем самой различной природы. Моделирование как метод – мощный инструмент науки и техники. Любое исследование заканчивается построением определенной модели объекта или процесса. Таким образом, методология моделирования во многом совпадает с методологией исследований. Как показывает опыт, активное участие в моделировании вырабатывает более глубокое понимание сути протекающих процессов и наблюдаемых явлений.

Развитие компьютерных технологий предоставляет в профессиональной деятельности новые возможности в педагогической практике с максимальной степенью наглядности и оперативности получить и представить информацию о свойствах объектов и характере протекающих в них процессов. Применение компьютерных методов моделирования, проведение компьютерных экспериментов способствуют углублению и расширению знаний о процессах, протекающих в конкретных системах или объектах, существующих или проектируемых.

Долгое время достаточно сильным препятствием в этом направлении была необходимость создания моделей средствами какой-либо системы программирования. В этом случае собственно моделирование отодвигалось на второй план, так как разработка модели начиналась специалистом в конкретной предметной области, затем математик формулировал математическую модель, а программист создавал компьютерную модель путем непосредственного программирования с получением в итоге программного продукта. Такой процесс требует привлечения узких специалистов и значительных затрат времени и средств, а также специфических знаний и умений.

В этом смысле компьютерное моделирование на основе специализированных инструментальных программных комплексов типа: MVS, RMD, AnyDynamics8 предоставляет возможность построить процесс моделирования, который будет принципиально отличаться тем, что модель создается средствами быстрой разработки, визуальными методами, с автоматическим выбором численных методов и генерацией исполняемого файла (**моделирование без программирования**).

Это позволяет использовать технологию компьютерного моделирования в учебном процессе по ряду дисциплин физико-математического и естественно-научного циклов. Таким образом, инструментальные программные комплексы моделирования, дающие возможность конструирования моделей с наглядным представлением результатов при минимальной потребности в программировании, имеют особую ценность.

Визуализация – уникальная возможность компьютерной технологии моделирования, так как показать невидимое явление способны только компьютерные модели. Моделирование составляет неотъемлемую часть современной науки и техники, причем по важности моделирование приобретает первостепенное значение. Термин «моделирование» в большинстве случаев означает «компьютерное моделирование», так как применение компьютеров существенно расширило возможности и породило новые технологии и методики.

Цель настоящего пособия – формирование умений и навыков применения современных методов построения, реализации и исследования на основе компьютерных моделей объектов, процессов или систем разнообразной природы. Расширение представления студентов о моделировании как о методе научного познания; знакомство с методологией моделирования; обучение применению компьютера как средства познания в различных областях практической деятельности.

Таким образом, пособие позволяет решить следующие задачи подготовки специалистов:

• познакомить с современными методами и технологиями построения моделей и проведения модельных экспериментов средствами специализированных программных комплексов;

• обучить эффективному применению моделирования и модельного эксперимента;

• развить творческий потенциал будущего специалиста, необходимый для дальнейшего самообучения в условиях непрерывного развития и совершенствования информационных технологий;

• сформировать первоначальные навыки исследовательской деятельности.

В результате изучения представленного в пособии материала студенты должны научиться использовать основные законы естественнонаучных дисциплин в профессиональной деятельности, применять

методы математического моделирования, теоретического и экспериментального исследования, проводить моделирование процессов и систем. Результатом формирования компетенции в области моделирования будет являться готовность к участию в постановке задач проведения экспериментальных исследований; способность обосновывать правильность выбранной модели, сопоставляя результаты экспериментальных данных и полученных модедьшых решений в профессиональной деятельности.

Пособие предназначено для студентов направлений: 44.03.01 «Педагогическое образование», профиль «Информатика»; 44.03.05 «Педагогическое образование», профиль «Информатика (с дополнительной специальностью)».

Настоящее пособие, с учетом опыта преподавания курса «Компьютерное моделирование», изменения объема аудиторной работы дополнено, переработано и является продолжением следующего учебного пособия: А.Л. Королев «Компьютерное моделирование. Лабораторный практикум».

За время, прошедшее с момента написания данного пособия, изменились требования по составу компетенций в курсе «Компьютерное моделирование», появились новые программные комплексы и технологии. Таким образом, переработка и обновление учебных пособий становятся актуальной задачей. Лабораторный практикум построен на доступном программном обеспечении, многие лабораторные работы могут использоваться в школе в рамках курса «Информатика».

В заключение следует отметить, что компьютерное моделирование изначально базируется на математике, но никак не заменяет ее, а лишь дополняет новыми возможностями при исследовании моделей.

ГЛАВА 1. КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ В СРЕДЕ ПАКЕТА MVS

1.1. Математическое моделирование

Элементы математического моделирования используются с момента появления точных наук. Собственно и само рождение науки математики связано с решением практических задач на основе вычислений и моделирования еще в древней Греции. Многие методы математического моделирования носят имена великих ученых-математиков.

Актуальность математического моделирования связана со следующими обстоятельствами:

• использование научных знаний для решения конкретных практических задач. Например, в задаче проектирования – установить значения параметров объекта, чтобы его свойства соответствовали требуемым.

• общенаучное значение математического моделирования состоит в том, что оно дает возможность сравнения выводов теории с результатами наблюдений и экспериментов.

Теоретические выводы всегда получают точный и практически значимый характер, если они выражены в виде математических моделей.

С развитием науки и техники возникла потребность в обработке и объяснении результатов испытаний, результатов, полученных в виде каких-либо зависимостей (например, дифференциальных уравнений или алгебраических соотношений). Обработка полученных результатов приводила к необходимости сложных вычислений. Такое положение дел дало стимул развитию численных методов и вычислительных средств, но и математическому моделированию, когда модель объекта позволяла получать новое знание в виде качественных результатов.

Например, математическая модель солнечной системы, построенная на основе закона всемирного тяготения и законов механики Ньютона, привела к открытию новой планеты. Таким образом, математическое моделирование – это и всеобъемлющая научная дисциплина, и метод исследования. Особенностью математического моделирования является общая схема изучения, возможно, любых объектов и

методика конструирования и обработки математических моделей вне зависимости от их конкретного смысла.

Существует мнение, что современная техника и современные технологии, определившие современное состояние жизни, основаны на успехах фундаментальной науки и являются детищем математического и компьютерного моделирования.

Цели и задачи такого моделирования могут быть кратко сформулированы как качественное и количественное изучение объектов и явлений природы, техники и общества. При этом под качественным изучением подразумевается достижение понимания существа изучаемого объекта или процесса, его свойств, поведения, возможных явлений и определяющих их причин. Такая ситуация часто актуальна в инженерной деятельности. Качественные исследования породили понятие «мягких» математических моделей [1].

Общая схема компьютерного математического моделирования в значительной мере устоялась, а ее реализация опирается на фундаментальную науку, методы исследования математических моделей (аналитические, качественные и численные) и на современную вычислительную технику. Вместе с тем продумывание этой схемы обнаруживает в ней действия, неподдающиеся формализации. Это, прежде всего, относится к построению модели и отчасти к ее исследованию. То, что не формализуемо, можно отнести к искусству моделирования.

При исследовании объекта, естественно, следует стремиться к построению возможно более простых моделей с точки зрения их анализа и возможностей работы с ними, но обеспечивающих «удовлетворительную адекватность» изучаемым объектам. Впрочем, если модели лишь частично и односторонне оценивают рассматриваемый объект, для качественной, а иногда и количественной оценки они полезны, причем полученные с их помощью представления позволяют строить более точные модели. Но, как показывает практика, даже самых простых моделей достаточно для решения задач, которые возникают в практической деятельности.

На примере простых моделей, использованных в пособии, рассматривается сущность компьютерного математического моделирования, в среде MVS, общие принципы построения моделей, средства и источники получения моделей, возможные некорректные подходы и т.д.

В то же время появились задачи, которые не решаются методами классического математического моделирования. Эти задачи составляют основу имитационного моделирования. Подобного вида задачу представляет, например, игра «Жизнь», которая имеет весьма сложное математическое описание, но просто решается методами имитационного моделирования. Эту игру разработал в 1970 г. английский математик Джон Конвей. Название связано с тем, что она имитирует рост, распад и различные изменения в популяции живых организмов. В эту игру можно поиграть, ничего не зная о каких-либо уравнениях, а на компьютере все выглядит весьма красиво и наглядно.

Имитационные модели являются более универсальными и могут быть построены и при **отсутствии математической модели** объекта моделирования.

В свое время С. Уолфрем, американский математик и программист, высказал гипотезу, согласно которой для многих сложных систем не существует простого (математического) описания, их анализ возможен только путем вычислительного компьютерного эксперимента. Он выдвинул фундаментальное предложение о переходе от непрерывного к дискретному моделированию и от аналитических вычислений к прямой численной имитации. Появились новые научные направления, например, вычислительная физика.

Мы живем в сложном нелинейном мире. Огромную роль в его познании сыграли компьютеры, позволившие исследовать множество нелинейных математических моделей, описывающих реальность. Возникла и обратная связь. Результаты компьютерного моделирования приводят к рождению новых теорий, понятий, моделей. Изучение этих моделей с помощью компьютеров приводит к рождению теорий и моделей нового поколения. Многие важнейшие открытия в науке XX столетия связаны с выявлением эффектов согласованного поведения (синергизма) совокупностей отдельных элементов.

При протекании знаменитой химической реакции Белоусова – Жаботинского в пробирке периодически пробегает волна изменения цвета. Это означает, что хаотически движущиеся атомы и молекулы становятся периодически участниками каких-то согласованных процессов, которые, вероятно, очень быстро развиваются и охватывают огромное число элементов среды, обеспечивая единое коллективное поведение. Это представляет собой достаточно глубокую аналогию с поведением

стаи поведения множества людей под волнами моды, социальными течениями, войнами и революциями, втягивающими огромные массы, часто даже против их воли.

В появлении упорядоченности важную роль играют диссипативные процессы: диффузия, вязкость, теплопроводность и множество других. Однако представление о том, что процессы, уничтожающие порядок в простейших линейных системах, могут быть в нелинейном мире «архитекторами упорядоченности», до сих пор кажется парадоксальным. Чтобы подчеркнуть необычность этого взгляда. Один из основоположников теории самоорганизации Илья Пригожин назвал упорядоченность, возникающую в открытых нелинейных системах, далеких от равновесия, и существенно связанную с рассеянием энергии, вещества или информации, диссипативными структурами. Исследование подобных сложных нелинейных моделей возможно только на основе компьютерного моделирования. Дополнительный эффект дает совместное применение численных и аналитических методов исследования.

Так как в науке стало актуальным изучение множества нелинейных моделей, то второе рождение математического моделирования связано с появлением компьютеров, которые избавили ученых и инженеров от огромной рутинной работы по проведению расчетов и существенно расширили сферы приложения математического моделирования, без которого нельзя представить современную науку и технику.

Прямой натурный эксперимент для большинства объектов долог, дорог, опасен или невозможен. Многие из технических систем существуют в единственном экземпляре, а результаты экспериментов с реальными объектами могут привести к необратимым отрицательным последствиям.

При создании новых технических объектов их необходимо сначала спроектировать и установить, какие значения должны иметь параметры объекта, чтобы его свойства и функции соответствовали требуемым. Иными словами, параметры нового объекта, прежде чем он будет создан, нужно рассчитать и необходимо провести анализ свойств нового объекта.

Совершенно очевидно, что результаты теоретических исследований в любой области науки будут иметь наибольшее практическое значение, если они будут выражены в виде конкретных **количественных зависимостей,** или, попросту говоря, в виде математических формул

или вычислительных алгоритмов. Это залог эффективного применения теории в практических целях. Кроме того, теоретические результаты необходимо сопоставлять с результатами измерений, полученных в ходе экспериментов или испытаний. Для решения подобных проблем необходимо моделирование на основе количественных закономерностей протекающих процессов. Именно такую возможность представляет компьютерное математическое моделирование. Оно является незаменимой составляющей в развитии науки и техники.

Применение компьютерного моделирования дает множество эффектов.

К ним можно отнести значительное расширение возможностей математического моделирования по сложности решаемых задач, возможность проведения **вычислительного эксперимента**.

Появились **новые** виды моделирования:

- имитационное моделирование;
- моделирование знаний;
- **визуализация** результатов моделирования (3D-графика, анимация, виртуальная реальность);
- **автоматизация** построения самой модели, выбора **математических методов** и средств **отображения результатов.**

Параметры исследуемого объекта можно разделить по степени принадлежности собственно объекту или по характеру взаимодействия с окружающей средой:

1. Входные параметры. Характеризуют внешние воздействия на объект (например, внешние управляющие воздействия).

2. Выходные параметры. Характеризуют реакцию объекта, его воздействие на окружающую среду, то есть характеризуют внешнее проявление объекта.

3. Внутренние параметры. Характеризуют свойства процессов, протекающих в самом объекте. Внутренние параметры могут быть зависимыми или независимыми. Естественно, что независимые параметры можно изменять произвольно, а зависимые параметры изменяются только косвенно, в силу их зависимости от других факторов.

Таким образом, объект моделирования характеризуется совокупностью:

- внешних управляющих воздействий – $X(t)$;
- воздействий окружающей среды – $V(t)$;

• внутренних независимых параметров – *H(t);*
• внутренних зависимых параметров – *P(t)*;
• выходных параметров системы – *Y(t).*

Учитывая вышесказанное, можно утверждать, что математическая модель устанавливает количественные связи между входными и независимыми внутренними параметрами, с одной стороны, и выходными и внутренними зависимыми параметрами – с другой. Как правило, математическая модель – это система уравнений (алгебраических, дифференциальных, интегральных).

В общем случае векторные величины *X, V, H* могут содержать как детерминированные, так и стохастические (случайные) составляющие. Входные воздействия *X*, внутренние параметры *V* и частично воздействия окружающей среды являются независимыми переменными. Совокупность векторных величин *X, V, H, Y* полностью характеризует состояние объекта. Процесс изменения состояния объекта можно выразить следующей обобщенной математической моделью:

$$Y(t) = F_1(t, X, V, P, H), \quad P(t) = F_2(t, X, V, H), \quad V(t) = F_3(t, Y).$$

Представленные соотношения дополняются начальными и краевыми условиями, которые определяют состояние объекта моделирования в момент времени $t = 0$ и взаимодействие с окружающей средой. Данные соотношения называются законом функционирования объекта. Подобные модели принято называть динамическими. В этом случае параметры, которые изменяют свои значения во времени и требуют определения, называются переменными.

Совокупность переменных, определяющих состояние динамической системы, называют фазовым вектором, а область изменения этого вектора – **фазовым пространством**.

В большинстве случаев математическая модель представляет собой задачу некоторого раздела математики, для которой методы исследования уже разработаны.

Замечательным свойством является формальное сходство (**аналогия**) математических моделей разнородных по своей природе объектов и процессов. Таким образом, имеется возможность сгруппировать математические модели в однородные с точки зрения математики классы и исследовать их как самостоятельные абстрактные математические объекты безотносительно оригиналов.

Основой построения математической модели могут быть **фундаментальные законы природы**. Наиболее распространенный способ построения математических моделей как раз и состоит в применении фундаментальных законов к конкретной ситуации. Однако чисто теоретическим путем математическую модель какого-либо объекта построить проблематично, на определенном этапе всегда приходится использовать данные экспериментов и наблюдений, феноменологические законы или полуэмпирические зависимости.

В научной литературе в качестве необходимых условий содержательного математического моделирования предполагается наличие **априорной информации** о природе и характере исследуемых объектов и процессов, например, в форме научных теорий, законов и т.п. Кроме того, необходимо наличие некоторых опытных данных о процессах в исследуемом объекте. Из исходной априорной информации выводятся общие математические соотношения, описывающие законы функционирования объекта моделирования. А на основе статистической обработки опытных данных определяются численные значения параметров модели. Результаты такой обработки опытных данных могут использоваться в виде фундаментальных физических констант или полуэмпирических зависимостей, которые в большом количестве можно найти в специальной справочной литературе.

Таким образом, при построении математических моделей существует несколько возможностей решения задачи.

1. Построение модели на основе законов, описывающих протекающие в объекте процессы, на основе знания о механизмах процессов и явлений с привлечением фундаментальных законов природы.

Такой метод можно назвать **аналитическим или теоретическим**. При построении модели составляется описание закономерностей протекающих в объекте процессов в виде набора математических соотношений. Далее, на основе анализа модели, делаются определенные выводы, которые проверяются на практике. Достоинством этого метода является то, что он обеспечивает получение новой информации о свойствах объекта моделирования. Например, гелиоцентрическая модель солнечной системы построена на основе закона всемирного тяготения и законов механики Ньютона. Такая модель позволила установить наличие в солнечной системе неизвестной ранее планеты.

Первый путь реализуется при достаточной изученности общих закономерностей процессов, протекающих в моделируемом объекте. Параметры таких моделей определяются либо на основе полуэмпирических зависимостей, либо на основе теории подобия, либо путем обработки данных экспериментов. Например, для применения закона всемирного тяготения в моделировании движения космических тел требуется экспериментальное определение гравитационной константы.

Недостатком аналитических моделей является сложность и нелинейность получающихся при этом уравнений. Достоинством является общность результатов моделирования и большая информативность моделей, способных предсказать новые неизвестные свойства изучаемых процессов и явлений.

2. Построение модели объекта путем ее идентификации, то есть чисто формальным путем с помощью статистической обработки результатов измерений без опоры на какие-либо знания о закономерностях процессов.

Суть метода состоит в том, чтобы по данным наблюдений за входными и выходными параметрами объекта построить такую математическую модель (математическую зависимость), которая описывала бы связь между этими параметрами. Как правило, заранее выбирается определенный вид математической **зависимости** (например, в виде алгебраического многочлена). В этом случае при идентификации определению подлежат только параметры принятого математического описания. Такие модели используются при моделировании систем. Элементы систем моделируются предельно простыми формальными математическими зависимостями, которые называются «**черный ящик**» и описывают связь только между входными и выходными параметрами.

Второй путь, который называется **экспериментальным** методом, применяется при отсутствии информации о механизмах процессов, слабой изученности либо сложности объекта моделирования. Этот путь используется при исследовании объекта в достаточно узком, «рабочем», диапазоне параметров. Подобные методы чаще всего основаны на предположении о линейности зависимостей и сосредоточенности параметров объекта. При таком подходе требуется проведение опытов непосредственно на самом изучаемом объекте. Достоинством экспериментального метода является простота получаемых моделей

при достаточно точном описании свойств объекта в узком диапазоне изменения параметров. Однако экспериментальный метод не всегда позволяет распространить полученные результаты на другие однотипные объекты.

Сочетание обоих методов, аналитического описания и экспериментального определения неизвестных параметров модели, позволяет соединить сильные стороны каждого метода.

3. Построение модели системы на основе моделей элементов.

Обычно этот метод используется тогда, когда необходимо построить модель сложной системы на основе моделей ее элементов или когда из заданного набора элементов необходимо составить сложный объект и определить его свойства. Подобный подход используется в программном комплексе **Simulink**. Третий путь характерен для имитационного моделирования сложных систем, когда исследователя интересуют свойства системы в целом.

1.2. Основные характеристики пакета MVS

Model Vision Studium (MVS) – это интегрированная графическая среда для быстрого создания интерактивных компьютерных визуальных моделей сложных динамических систем и проведения с ними вычислительных экспериментов. Главными проблемами при разработке MVS являлись: поддержка технологии объектно-ориентированного моделирования**;** удобное и адекватное описание дискретно-непрерывных (**гибридных**) систем; обеспечение достоверного численного решения; обеспечение **визуализации** результатов моделирования ***без какого-либо программирования.***

Система MVS является мощным средством быстрого построения моделей для анализа систем различной природы и степени сложности, имеющих гибридное поведение. Она позволяет строить модели, пользуясь технологией объектно-ориентированного моделирования (ООМ) и блочного моделирования, с визуализацией результатов в виде 2D и 3D анимации.

Авторы – исследовательская группа «Моделирование сложных динамических систем» (MVSTUDIUM Group) на базе Санкт-Петербургского государственного политехнического университета:

• Инихов Дмитрий Борисович,
• Исаков Андрей Алексеевич,
• Колесов Юрий Борисович,
• Сениченков Юрий Борисович.

Первоначально MVS создавался для использования в образовании, т.е. при обучении компьютерному моделированию. Однако для применения MVS в других областях были созданы версии MVS 4, MVS 6 и RMD и AnyDynamics.

Версия MVS 3.2.24 Free является наиболее простой, наглядной, с достаточным для образования набором функций и рекомендуется для первоначального освоения. Впрочем, другие современные версии могут, не понадобится.

Основными областями применения MVS являются:
• проведение научных вычислительных экспериментов;
• проектирование технических систем;
• моделирование экономических систем;
• обучение;
• разработка математических моделей физических систем и процессов;
• создание компьютерных тренажеров.

Поддерживаются следующие виды моделирования:
• непрерывное;
• дискретно-событийное;
• непрерывно-дискретное (гибридное);
• объектно-ориентированное моделирование;
• стохастический вычислительный эксперимент.

Программный комплекс MVS позволяет быстро создавать модели непрерывных, дискретных и гибридных (непрерывно-дискретных) систем.

Гибридная система – это объект, обладающий одновременно непрерывными и дискретными свойствами. Для таких систем достаточно трудно получить корректное численное решение, так как математическая задача содержит разрывные функции. При создании MVS авторы исходили из того, что корректное численное решение должно получаться автоматически.

Система MVS включает специальную форму наглядного представления гибридного поведения – **карту поведения**.

Анализ свойств решаемой математической задачи и выбор и настройка метода решения выполняются MVS, а не пользователем. Пользователь имеет возможность активно вмешиваться в ход вычислительного эксперимента и при необходимости получать о решении как можно больше дополнительной информации.

MVS не предъявляет никаких требований к знаниям по программированию: используются интуитивно понятные общепринятые формы для описания математических зависимостей и визуальные диаграммы для описания структуры и качественных изменений поведения моделируемой системы. Численные методы решения задачи также выбираются автоматически.

Непрерывное поведение систем описывается с помощью дифференциальных и алгебраических уравнений первого и второго порядка произвольной формы (в том числе неразрешенных относительно производных). Уравнения задаются в естественном математическом представлении. Для описания дискретного и гибридного поведения используются визуальные карты поведения. Дискретные действия записываются с помощью несложного алгоритмического языка, включающего хорошо известные базовые конструкции традиционных алгоритмических языков.

Программный код выполняемой модели (файл с расширением exe) автоматически генерируется на основе математической модели и компилируется. В MVS имеется достаточно богатый набор численных методов, предназначенных для воспроизведения поведения гибридных систем. Это программные реализации методов решения нелинейных алгебраических уравнений, систем обыкновенных дифференциальных уравнений и систем алгебраических и дифференциальных уравнений.

Имеется возможность проведения вычислительного эксперимента, а также встраивания выполняемой модели во внешнем приложении.

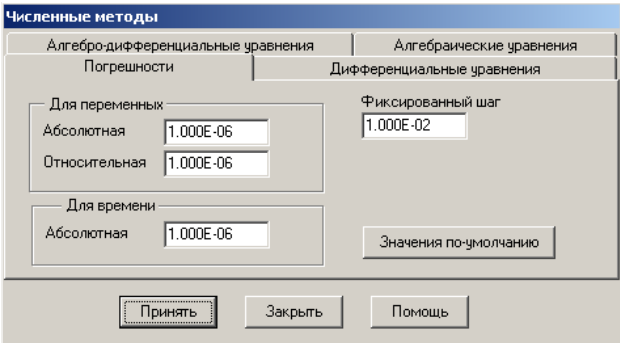

Рис. 1.2.1. Численные методы MVS

Для каждой группы задач имеется свой автоматический решатель, цель которого обеспечивать получение решения на заданном временном участке, любыми доступными, из числа имеющихся в пакете, методами. Любой автоматический решатель для новой задачи выбирает наименее трудоемкий метод, а если при этом возникают какие-либо сложности, пытается подобрать метод, способный их преодолеть. Каждая смена поведения рассматривается как новая численная задача.

Если автоматический решатель не справляется с поставленной задачей, тогда пользователь может попытаться управлять выбором метода самостоятельно. Пользователю доступны такие же программные реализации численных методов, что и автоматическому решателю (рис. 1.2.1).

Программно-аппаратные требования: Intel-совместимые компьютеры с операционной системой MS Windows (XP, Vista, 7, 8, 10).

1.3. Установка и запуск пакета MVS

Для установки пакета необходимо запустить программу **MvStudium_3.2.24.exe** и указать место на жестком диске для размещения пакета. Версия 3.2.24 занимает около 25 Мбайт на жестком диске. По умолчанию пакет устанавливается по адресу: **C:\Program Files\mv30** (рис. 1.3.1).

Для удаления пакета войдите в «**Панель управления/Установка и удаление программ**», найдите «**Model Vision Studium Free**» и удалите его.

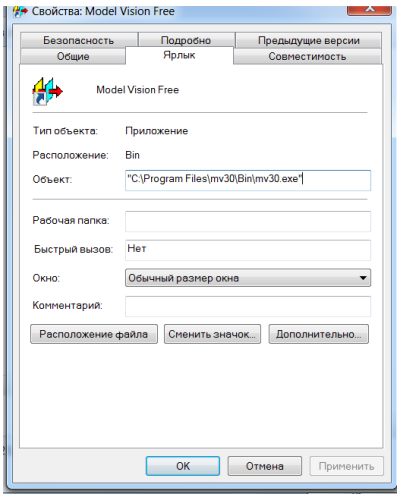

Рис. 1.3.1. Свойства пакета MVS

Пакет MVS можно скачать с образовательного сайта Exponenta (**URL http://old.exponenta.ru/SOFT/OTHERS/mvs/mvs.asp**).

Дальнейшим развитием системы **MVS** является пакеты **RMD** и **AnyDynamics** (свободно распространяемая версия **RMD**). Но для решения учебных задач пакета **MVS вполне достаточно,** он значительно проще и нагляднее.

Запуск на выполнение MVS осуществляется из папки **mv30\Bin\ mv30.exe** или из меню «**Пуск**» – «**Model Vision Free**». При запуске MVS откроется окно (рис. 1.3.2).

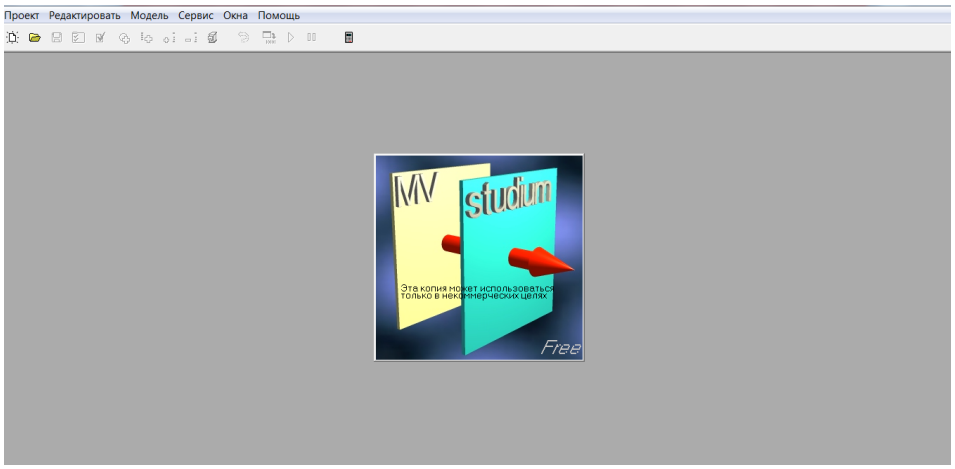

Рис 1.3.2. Окно MVS при запуске пакета

При создании новой модели (новый проект) в главном меню пакета MVS необходимо войти в пункт «**Проект**» и выбрать подпункт «**Новый...**» или нажать кнопку  (рис. 1.3.3–1.3.4).

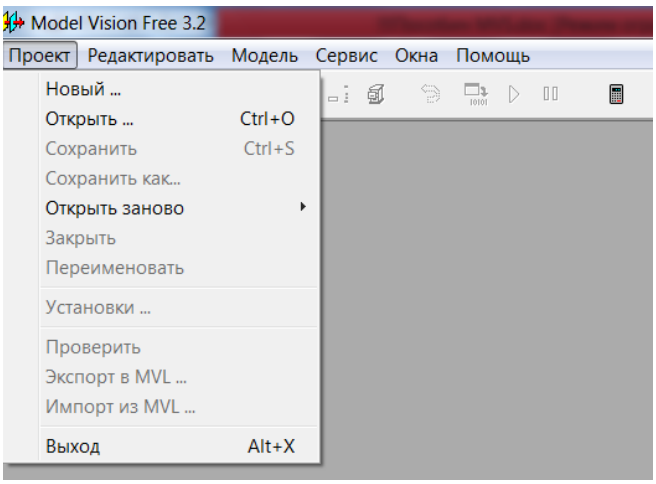

Рис. 1.3.3. Содержание пункта «Проект» главного меню MVS

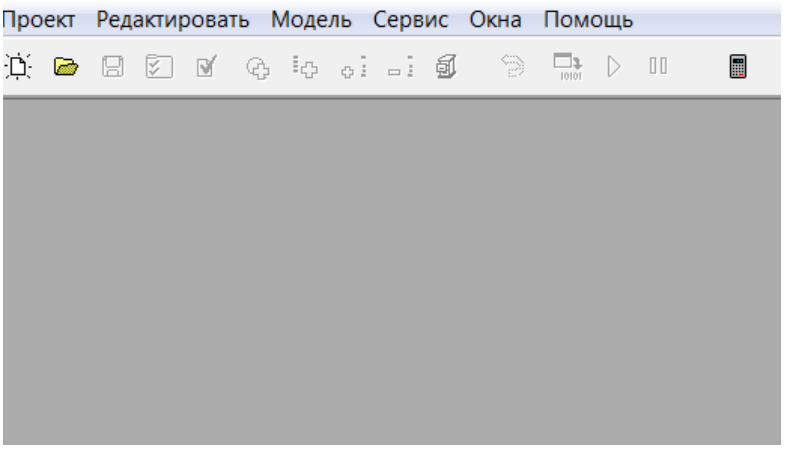

Рис. 1.3.4. Кнопка создания нового проекта в главном меню MVS

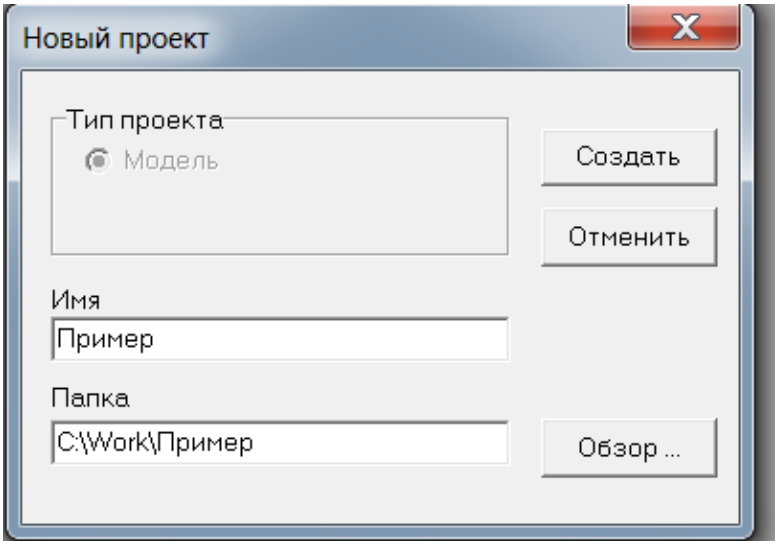

Рис. 1.3.5. Задание имени проекта и папки для его размещения

Для нового проекта необходимо задать имя и место на компьютере для его размещения. Папка проекта должна быть доступна для записи. На рис. 1.3.5 задано имя проекта «**Пример**» и вобрана папка **C:\Work**, причем в папке «**Work**» автоматически будет создана папка «**Пример**». Выбор папки для нового проекта осуществляется с помощью кнопки «**Обзор**». Далее следует нажать кнопку «**Создать**» (рис. 1.3.5), при этом откроется пустое окно нового проекта (рис. 1.3.6).

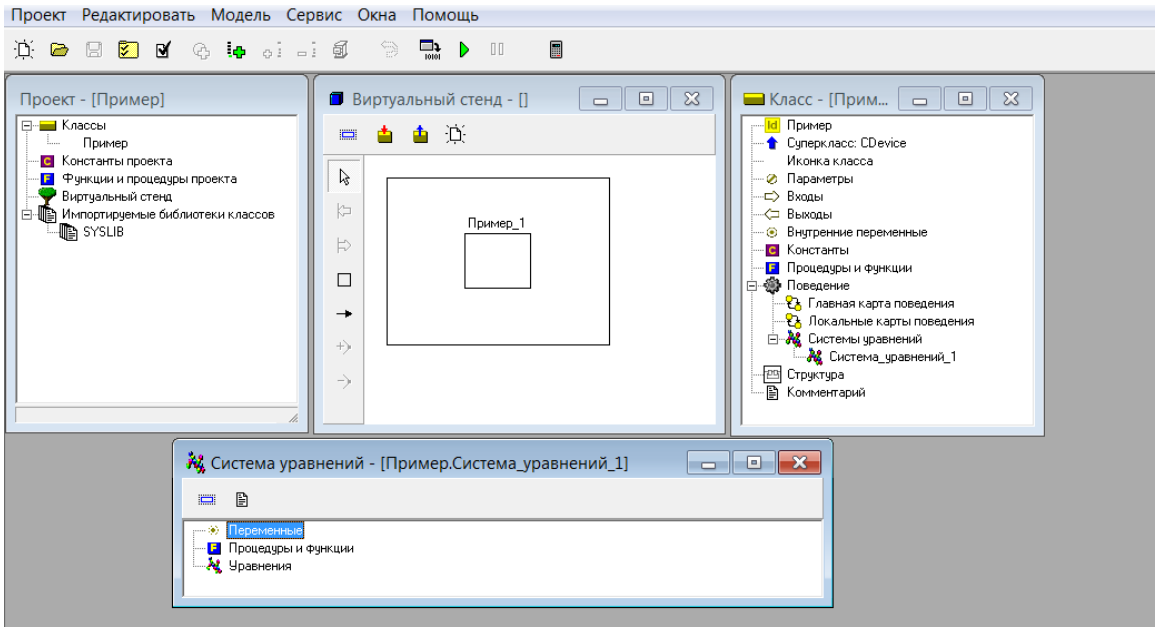

Рис. 1.3.6. Окно MVS при создании нового проекта

Окно нового проекта содержит дочерние окна: «**Проект (Имя проекта)**», «**Виртуальный стенд**», «**Класс (Имя проекта)**, «**Система уравнений**». Назначение этих окон будет рассматриваться далее при создании моделей. Данные проекта хранятся в нескольких файлах, расположенных в папке проекта. Основной файл проекта (база данных проекта) имеет расширение «**.mvb**». Значительная часть действий в пакете выполняется методом «drag and drop» («перенеси и брось»).

Глава 2. МОДЕЛИ НЕПРЕРЫВНЫХ ПРОЦЕССОВ

2.1. Создание простой непрерывной модели

Рассмотрим создание MVS-модели осциллятора. Это модель непрерывной, изолированной системы. Естественно, что предварительно должна быть сформулирована математическая модель объекта моделирования.

Моделируемая система представляет собой тело, прикрепленное к невесомой пружине, другой конец которой жестко закреплен (рис. 2.1.1). На тело действует сила упругости пружины и сила трения. Примером такой системы может служить амортизатор автомобиля. Первоначально система выведена из состояния равновесия.

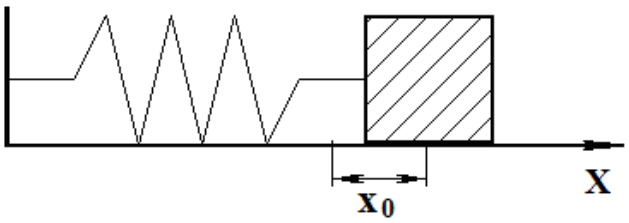

Рис. 2.1.1. Схема объекта моделирования.
Начальное состояние – пружина растянута

Исходная размерная модель динамики колебаний, записанная в виде системы дифференциальных уравнений, имеет следующий вид:

$$m\frac{dV}{dt} = -c \cdot x - k \cdot V, \quad \frac{dx}{dt} = V$$

Или

$$\frac{dV}{dt} = -\frac{c}{m} \cdot x - \frac{k}{m} \cdot V, \quad \frac{dx}{dt} = V$$

Начальные условия: $x\ (t = 0) = x_0$, $V\ (t = 0) = 0$.

Здесь m – масса тела, x – координата тела, V – скорость движения тела, c – жесткость пружины, k – коэффициент трения. Значение координаты $x = 0$ соответствует положению равновесия.

Первое уравнение – это уравнение второго закона Ньютона, второе уравнение – кинематическое определение скорости движения тела.

Требуется средствами MVS построить модель системы и отобразить изменение переменных в виде временной и фазовой диаграмм, а также в виде 3D-анимации.

После запуска MVS нажмите кнопку  или выполните команду главного меню **Проект\Новый**... (рис. 2.1.2).

В окне «**Новый проект**» выберите путь к папке проекта. Введите имя проекта и нажмите кнопку «**Создать**». В данной папке будет создан файл базы данных проекта «**Осциллятор.mvb**» (рис. 2.1.3). На рис. 2.1.3 для проекта выбрана папка «**WORK**» на диске **D**:, имя проекта – «**Осциллятор**». При создании модели необходимо выбирать доступную на запись папку.

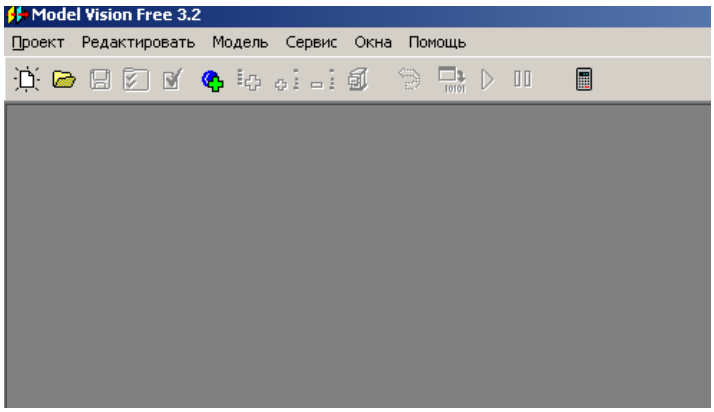

Рис. 2.1.2. Система MVS после запуска

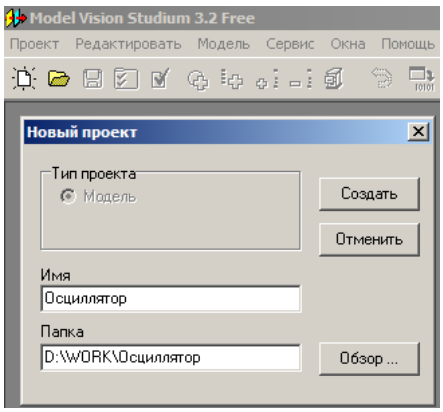

Рис. 2.1.3. Создание нового проекта

После этого в среде MVS появятся «заготовка» проекта со следующими окнами (рис. 2.1.4):

- **Окно проекта;**
- **Окно виртуального стенда**. По умолчанию в виртуальный стенд помещен экземпляр класса «**Осциллятор**» с именем «**Осциллятор_1**».
- **Окно класса** содержит **дерево составляющих класса**. Поскольку данный блок предполагается непрерывным, то по умолчанию в него добавлена пустая система уравнений с именем «**Система_уравнений_1**».
- **Окно системы уравнений** «**Система_уравнений_1**».

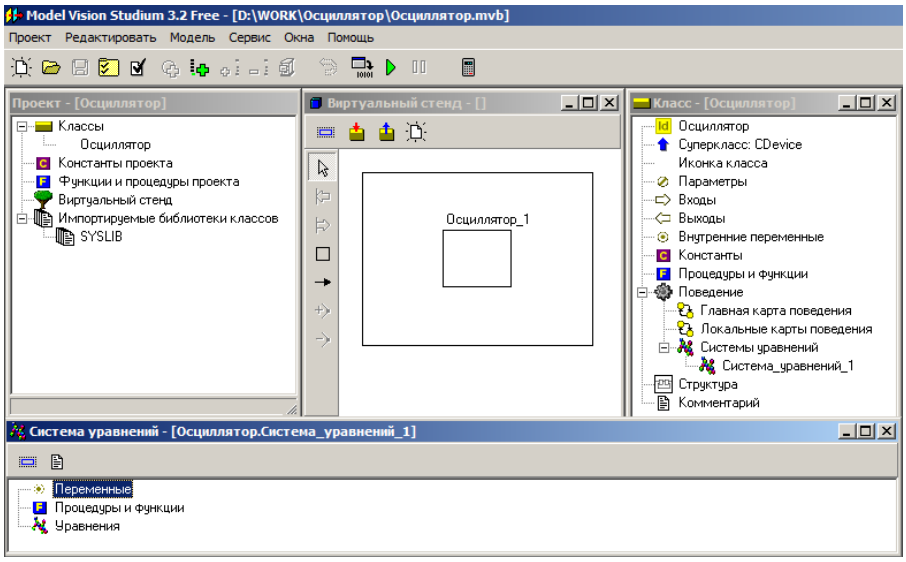

Рис. 2.1.4. «Заготовка» проекта

Осциллятор – непрерывная система. Для построения ее модели подходит класс, создаваемый по умолчанию при открытии нового проекта. В этот класс необходимо добавить соответствующие переменные, параметры, константы и уравнения. Естественно, что переменные изменяются во времени, значения параметров можно изменить для отдельного эксперимента с моделью, а значения констант остаются неизменными, их значения задаются при создании проекта.

В окне класса «**Осциллятор**», выделяем узел «**Внутренние переменные**», выполним команду «**Добавить**» (рис. 2.1.5) с помощью контекстное меню.

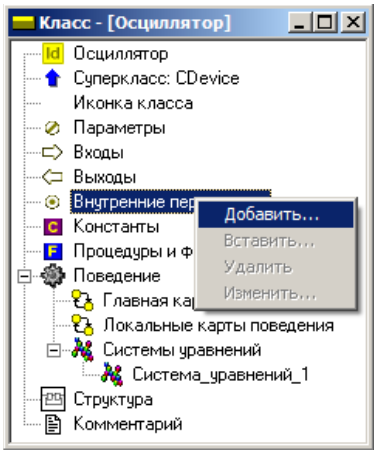

Рис. 2.1.5. Добавление переменной

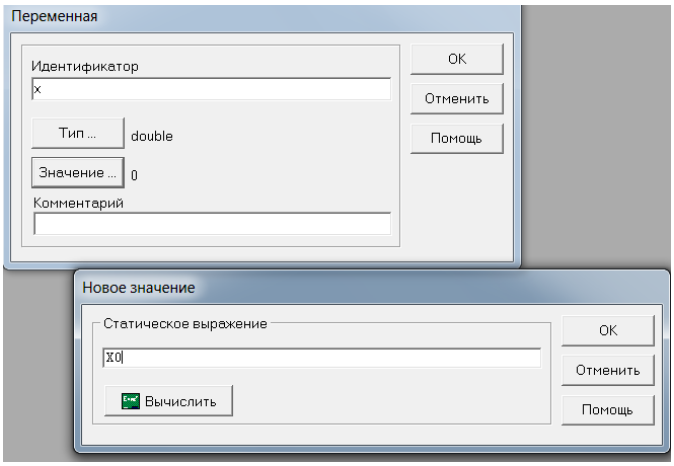

Рис. 2.1.6. Задание имени и значения переменной

В появившемся окне вводим имя переменной ***x***, оставляем заданный по умолчанию тип **double**, задаем начальное значение: параметр x_0 (рис. 2.1.6).

MVS при построении моделей оперирует латинским шрифтом (имена констант, параметров, переменных), различает строчные и прописные буквы и «не признает» пробелов. Любые имена должны начинаться с буквы. Если в окне класса для определения первоначального значения переменной используются другие переменные или параметры, то они уже должны быть заданы. То есть до создания переменной *x* необходимо в разделе «Параметры» окна **Класс-[Осциллятор]** (рис. 2.1.5) создать параметр $\mathbf{x}_0$ и задать его значение.

Аналогично добавляем другие переменные и параметры. Можно изменить или удалить введенные определения, дважды кликнув на них мышью, с помощью команд контекстного меню. Заполненное окно представлено на рис. 2.1.7.

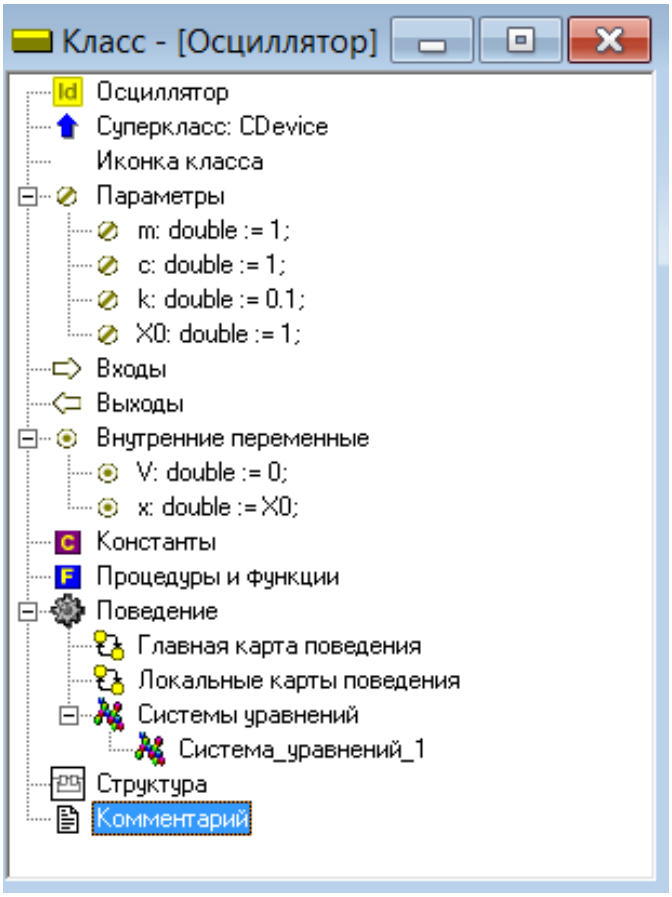

Рис. 2.1.7. Заполненное окно класса «Осциллятор»

В окне **Класс-[Осциллятор]** с помощью двойного щелчка мыши на узле «**Уравнения**» или команды «**Изменить**» контекстного меню вызываем **редактор формул**, который позволяет вводить математические выражения в виде, который близок к математической форме записи. С помощью редактора вводим необходимые уравнения (рис. 2.1.8). Специальный знак производной $\frac{d}{d}$ вводится через пункт меню «**Вставка**» редактора формул или с помощью кнопок на панели инструментов, которая обозначает первую производную, вторая производная вводится аналогично.

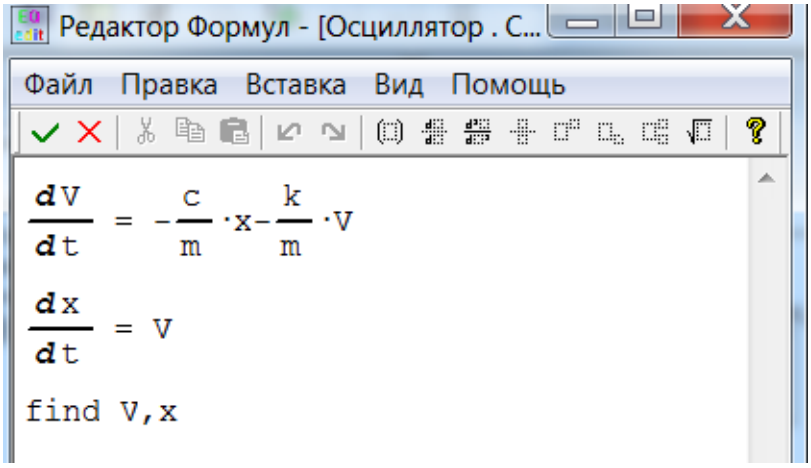

Рис. 2.1.8. Уравнений проекта

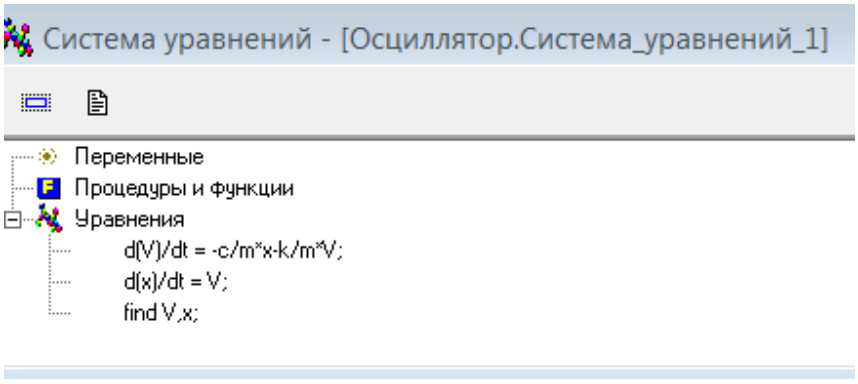

Рис. 2.1.9. Окончательный вид системы уравнений проекта

После редактирования система уравнений примет вид по рис. 2.1.8. Нажатие кнопки ✓ в окне **редактора формул** (рис. 2.1.8) переведет систему уравнений в вид по рис. 2.1.9.

После окончания создания проекта выполняем запуск модели с помощью команды «**Модель/Пуск**» главного меню или кнопки ▶.

На рис. 2.1.10 показано главное окно визуальной модели после создания. Визуальная модель является многооконным приложением. В левой части инструментальной панели отображается текущее значение модельного времени (начальное значение равно нулю).

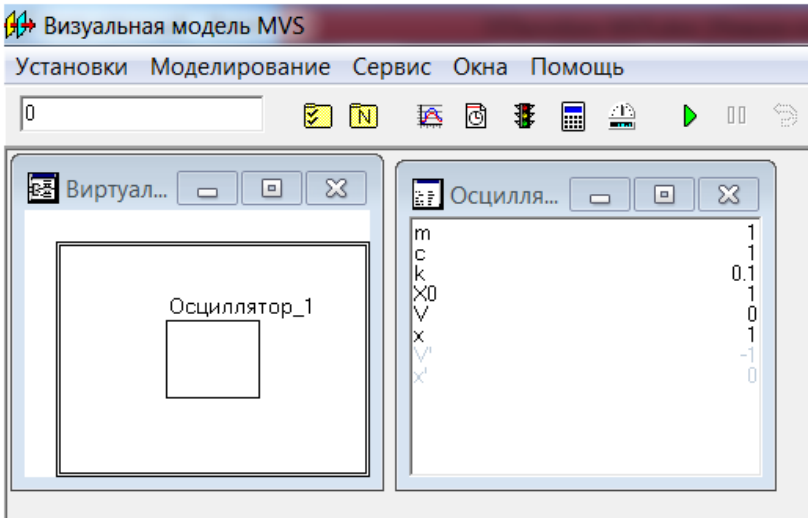

Рис. 2.1.10. Главное окно визуальной модели

Слева расположено окно виртуального стенда, которое отражает структуру модели. Для блока «**Осциллятор_1**», автоматически открывается окно переменных. Обратите внимание на то, что после создания экземпляра этого устройства его параметры приняли указанные значения и фазовые переменные инициализированы заданными выражениями.

Запустим выполнение модели с помощью кнопки ▶ или с помощью команды «**Моделирование/Пуск**» главного меню (рис. 2.1.10). При этом начнет изменяться модельное время и значения фазовых переменных. Останов выполнения модели производится с помощью кнопки ⏸ или команды «**Моделирование/Стоп**». Возврат модели в начальное состояние производится с помощью кнопки ↺ или команды «**Моделирование/Рестарт**». Модельное время снова равно нулю.

С помощью кнопки «**Новая диаграмма**» создадим окно диаграммы (по умолчанию это будет временная диаграмма, т.е. по оси абсцисс будут откладываться значения модельного времени). Методом drag-and-drop «перетащим» в окно «**Временная диаграмма**» из окна переменных «**Осциллятор_1**» переменные ***x*** и ***V***. Запустим модель и получим следующий график (рис. 2.1.11).

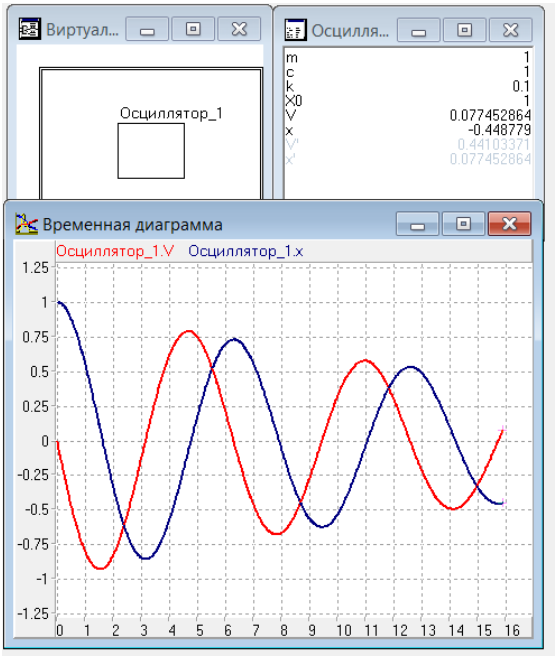

Рис. 2.1.11. Временная диаграмма процесса

Может оказаться, что эти несложные уравнения решаются так быстро, что вы просто не успеваете ничего заметить. С помощью кнопки  или команды «**Установки/Модель**» вызовите диалог редактирования установок. На станице «**Выполнение**» переключите параметр «**Соотношение модельного и реального времени**» из положения «**так быстро как можно**» в положение «**число**» (по умолчанию это 1, то есть моделирование в **реальном времени**). Изменяя это число, вы можете ускорять или замедлять прогон модели (рис. 2.1.12). Здесь же можно задать время останова прогона модели и другие установки.

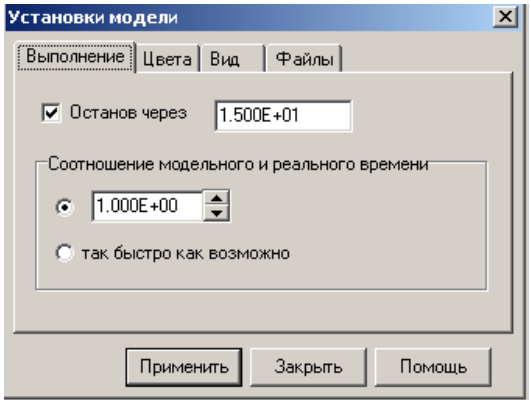

Рис. 2.1.12. Окно «Установки модели»

Построим **фазовую диаграмму,** т.е. график зависимости ***V(x)***. Для этого создадим **новую диаграмму**, перетащим в нее те же самые переменные, а затем правой клавишей мыши откроем на ней контекстное меню и выполним команду «**Настройка**». В появившемся диалоге настроек укажем с помощью двойного щелчка мышью в крайне правом поле ***X***, что по оси абсцисс откладываются значения переменной «**Осциллятор_1.x**» (рис. 2.1.13). В этом же окне можно задать и другие параметры диаграммы. Запустив модель, получим следующий график (рис.2.1.14) – **фазовую диаграмму**.

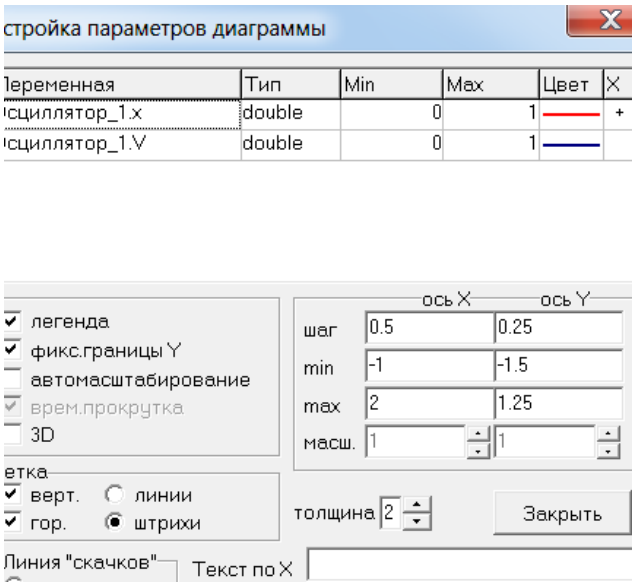

Рис. 2.1.13. Настройка параметров диаграммы

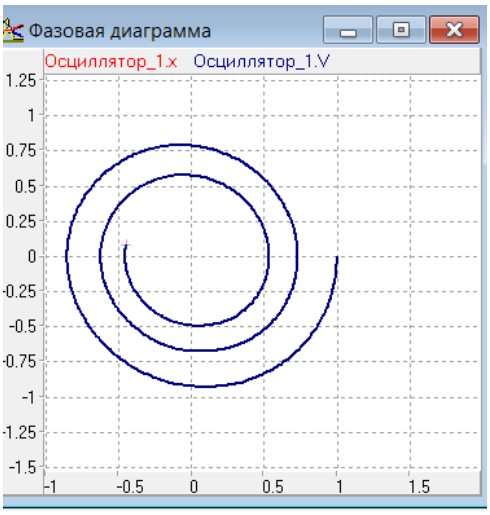

Рис.2.1.14. Фазовая диаграмма

Для моделей механических систем, можно получить больше информации из непосредственного наблюдения поведения трехмерного изображения. В визуальной модели для этого предназначено окно 3D-анимации. Создать его можно с помощью команды главного меню «Окна» – «Новая 3D-анимация».

Окно 3D-анимации позволяет строить динамические трехмерные модели, используя совокупность трехмерных примитивов (линия, шар, цилиндр, конус и т.д.), параметры, которых связываются со значениями соответствующих переменных модели.

С помощью команды «**Свойства**» контекстного меню, вызовем диалог редактирования свойств 3D-анимации. В данной модели нам понадобится только два стандартных объекта – пружина (**spring**) и сфера (**sphere**), рис. 2.1.15.

Один конец пружины должен всегда находиться в начале координат (параметр ***x1 = 0***), а координаты второго конца (параметры ***x2***) – изменяться в соответствии со значением переменной ***x***. Для задания этого соответствия «перетащим» необходимую переменную из **окна переменных** и бросим их в колонке «**Переменная**» соответствующих параметров отрезка (рис. 2.1.15). Аналогичным образом эту же переменную ***x*** мы сопоставляем с координатой центра сферы (параметр ***x1***, рис 2.1.15).

После чего достаточно запустить модель, и вы увидите действующий осциллятор (рис. 2.1.16). В любой момент вы можете изменить точку наблюдения, нажав левую клавишу мыши и перемещая ее с прижатой клавишей. Таким образом, вы можете рассматривать колебания осциллятора сверху, снизу и т.д.

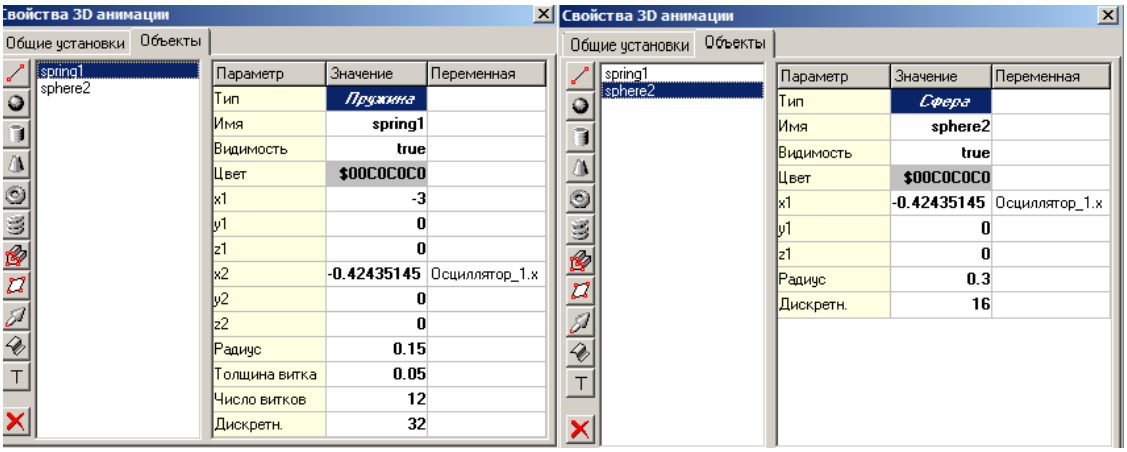

Рис. 2.1.15. Параметры 3D-объектов

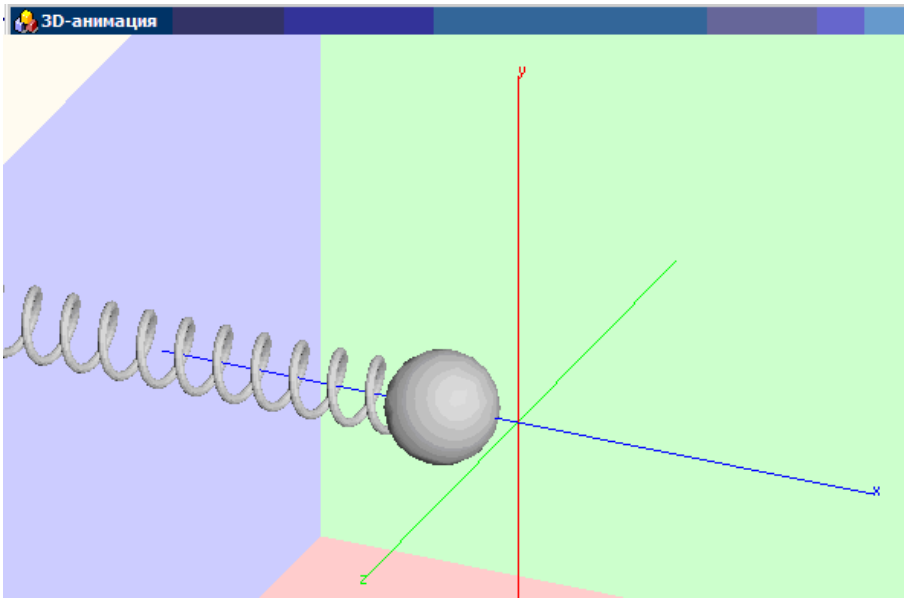

Рис. 2.1.16. Представление результатов моделирования средствами 3D-анимации

Первоначальное значение параметра ***k*** задано равным *0.1*. Можно изменять значение параметра ***k*** в среде визуальной модели (рис. 2.1.17) через контекстное меню. Новое значение параметра будет действовать в одном эксперименте с моделью. Таким же образом можно изменить начальные значения переменных, а значения констант не изменяются.

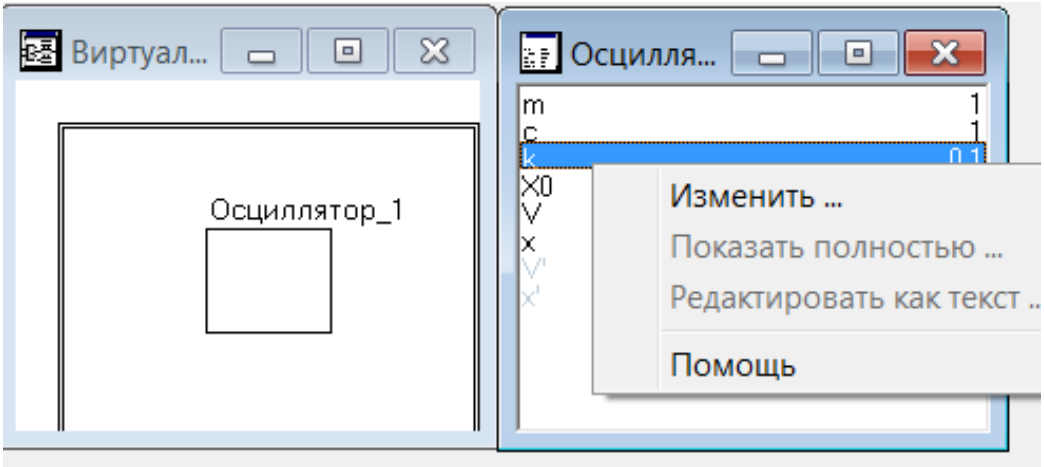

Рис. 2.1.17. Изменение значения параметра

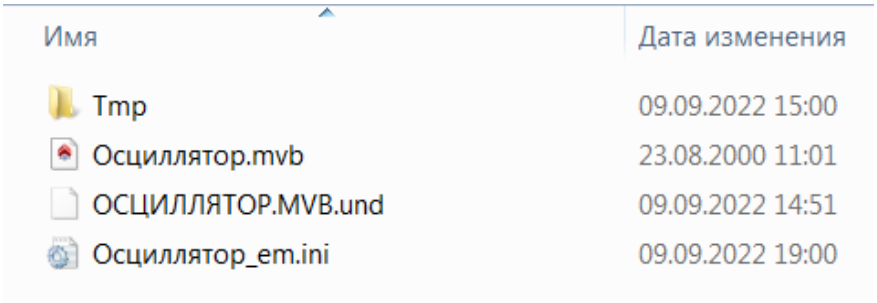

Рис. 2.1.18. Содержание папки проекта «Осциллятор»

Содержание папки «**Осциллятор»,** которая создается для проекта автоматически, представлено на рис. 2.1.18**.** Файл «**Осциллятор.mvb**» является файлом проекта «**Осциллятор**» и может быть открыт из среды пакета MVS.

2.2. Создание независимой от MVS выполняемой модели

Для создания выполняемой MVS-модели необходимо нажать кнопку «**Создать модель**»**.** Появится сообщение о создании выполняемой модели (рис. 2.2.1). Автоматически в папке **Tmp,** которая находится в папке проекта «**Осциллятор**» (рис. 2.1.18), будет создан файл «**model.exe**» (рис. 2.2.2).

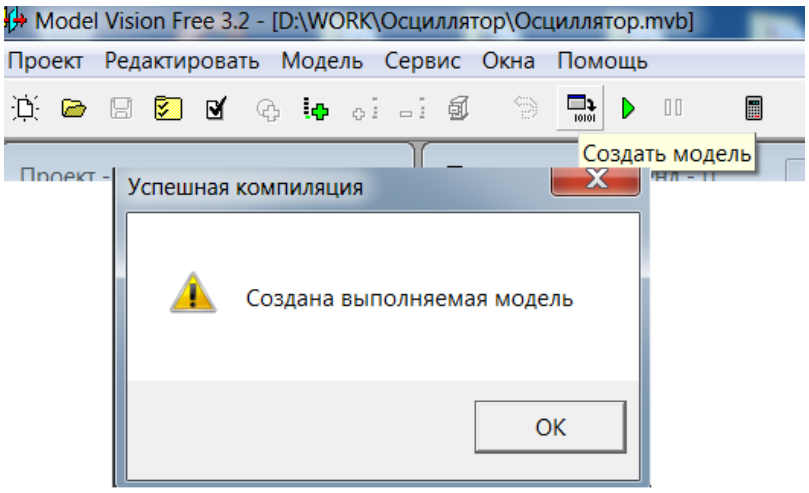

Рис. 2.2.1. Положение кнопки «Создать модель»

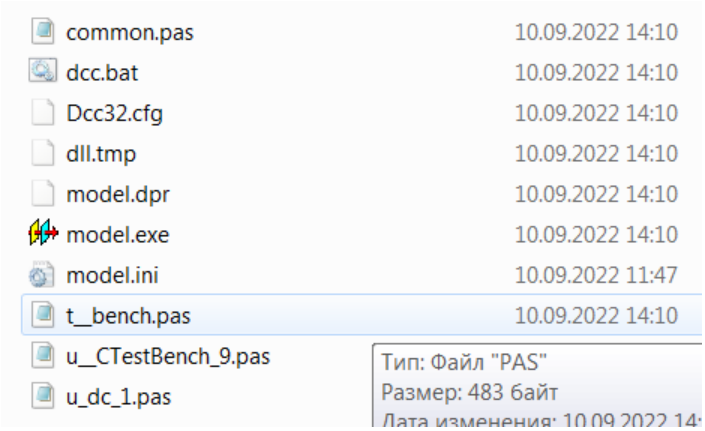

Рис. 2.2.2. Содержимое папки Tmp

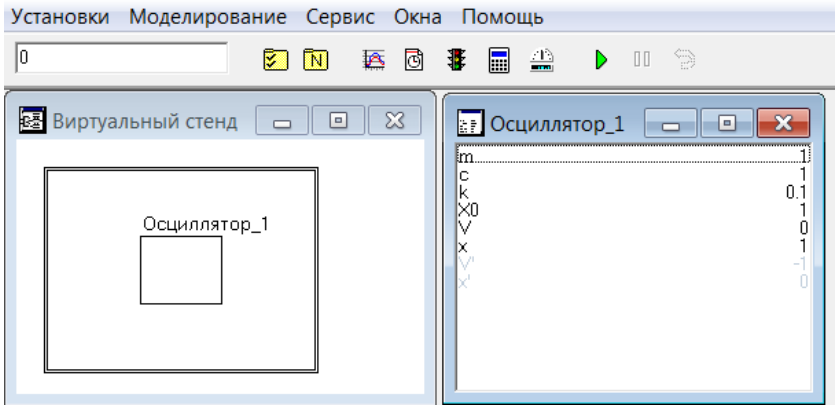

Рис. 2.2.3. Вид интерфейса MVS-модели поле первого запуска файла model.exe

Скопируем файл **model.exe** в отдельную папку, также скопируем в эту же папку файл **mathmvs.dll** из папки **C:\Program Files\mv30\Dll.**

Необходимо запустить файл **model.exe** и откроется модель «**Осциллятор**» (рис. 2.2.3). Ее нужно дополнить необходимыми диаграммами и сохранить.

В итоге содержимое папки, где находятся файлы **model.exe** и **mathmvs.dll** изменится, появится файл **model.ini** (рис. 2.2.4).

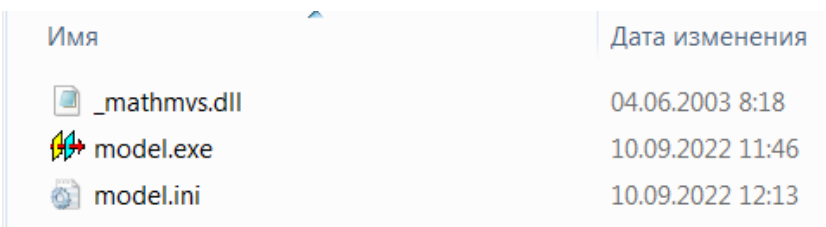

Рис. 2.2.4. Содержимое папки с выполняемой независимой от MVS моделью model.exe

Теперь можно запустить модель, как любую другую программу или использовать ее в других приложениях.

2.3. Моделирование типовых звеньев систем управления в среде MVS

Во многих системах управления имеются типовые элементы. Цель данной работы построить модели переходных процессов для интегрирующего, инерционного и колебательного звеньев (рис. 2.3.1).

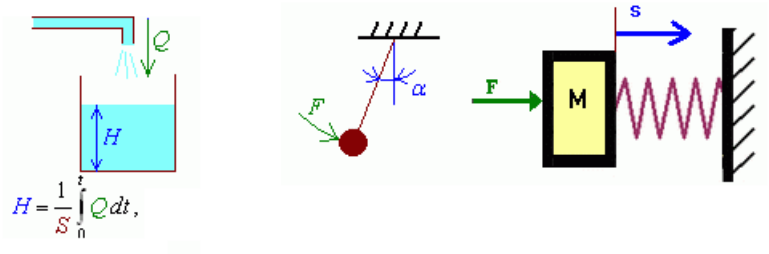

Рис. 2.3.1. Примеры различных звеньев

MVS-модель интегрирующего звена (рис. 2.3.2). имеет вид:

$$\frac{dy}{dt} = k \cdot x \quad x = 1, \quad k = 1, \quad y(t=0) = 0.$$

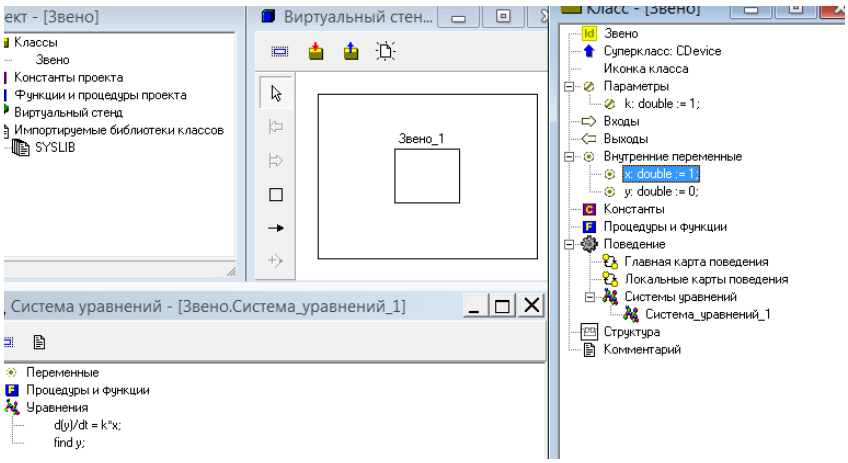

Рис. 2.3.2. MVS-модель интегрирующего звена

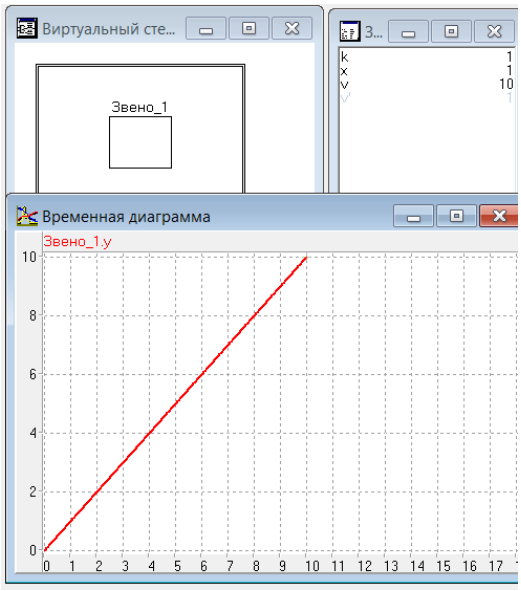

Рис. 2.3.3. Результат моделирования

Математическая модель инерционного звена имеет вид:

$$m \cdot \frac{dy}{dt} + c \cdot y = x \quad x = 1, \;\; m = 1, \;\; c = 2, \;\; y(t = 0) = 0.$$

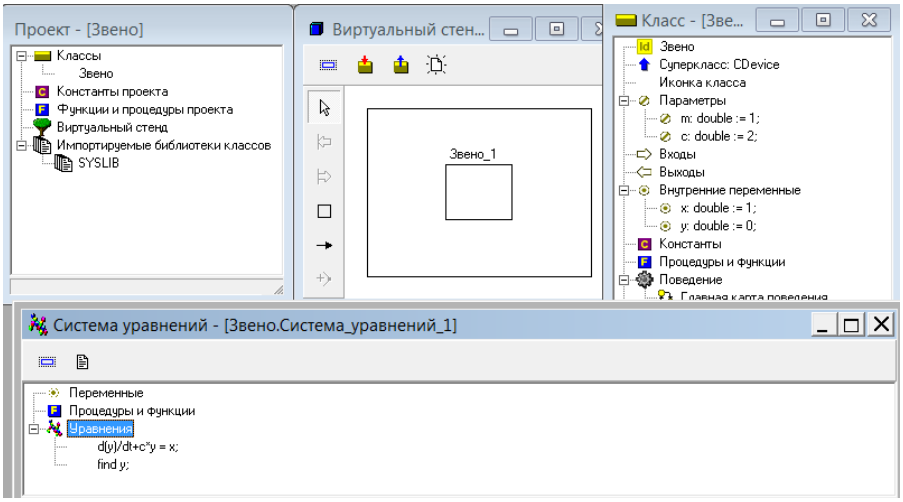

Рис. 2.3.4. MVS-модель инерционного звена

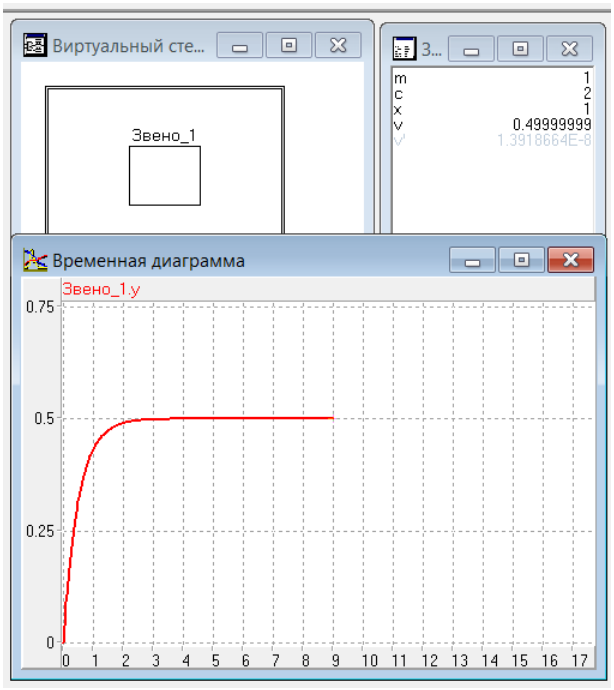

Рис. 2.3.5. Результат моделирования инерционного звена

Колебательное звено имеет следующую математическую модель:

$$m \cdot \frac{d^2 y}{dt^2} + k \cdot \frac{dy}{dt} + c \cdot y = x, \quad m = 1, \quad k = 1, \quad c = 2, \quad x = 1, \quad y(t = 0) = 0.$$

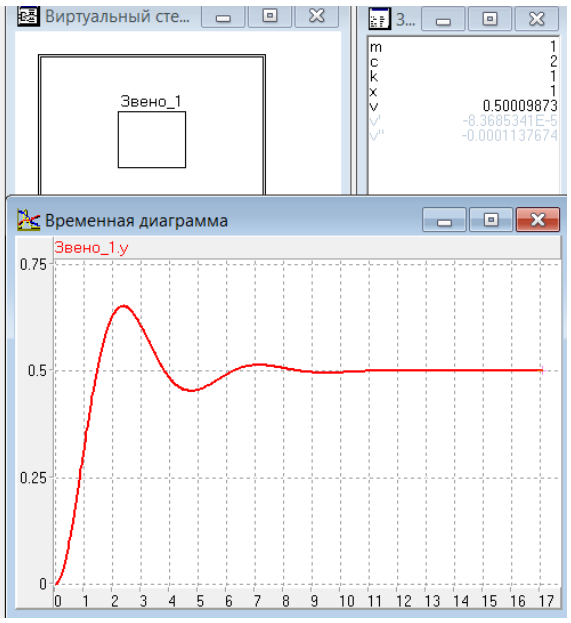

Рис. 2.3.6. Результат моделирования колебательного звена

Для каждой модели построить временную диаграмму и провести анализ влияния параметров на свойства переходных процессов. Исследовать поведение колебательного звена при гармоническом воздействии с частотой *w: x = sin(w*time) (x = sin(w*t)* – аналог вибрационного воздействия), определить резонансную частоту.

2.4. Моделирование развития популяции

Одной из первых моделей роста популяции была модель **Мальтуса**, модель развития популяции в условиях неограниченных ресурсов среды обитания. При этом считалось, что скорость роста популяции пропорциональна ее численности:

$$\frac{dx}{dt} = Rx .$$

Здесь R – врожденный коэффициент скорости роста популяции x – численность популяции. Данное уравнение имеет экспоненциальное решение с неограниченным ростом во времени: $\boldsymbol{x = x_0\, exp(Rt)}$.

Однако растущая популяция со временем исчерпает ресурсы окружающей среды и ее численность стабилизируется. В этом случае развитие популяции описывается логистическим уравнением **Ферхюльста**:

$$\frac{dx}{dt} = Rx\left(\frac{k-x}{k}\right).$$

Здесь k – экологическая емкость среды, т.е. максимальная численность популяции, которую способна «прокормить» экологическая система. Уравнение имеет простую интерпретацию – скорость роста популяции пропорциональна ее численности и свободной части среды обитания.

Необходимо провести исследование свойств модели развития популяции. Для этого требуется построить MVS-модель и провести модельный эксперимент с целью выявления качественных закономерностей развития популяций. Первый этап выполнения работы – получение безразмерной модели методом неопределенных масштабов с целью со-

кращения числа параметров [3]. После преобразования модель примет вид:

$$\frac{dx}{dt} = x(1-x) \quad , \qquad x(t=0) < 1.$$

Требуется создать модель в среде MVS соответственно уравнению. Изменение ***x*** во времени необходимо отразить с помощью временной диаграммы.

В ходе численного эксперимента необходимо проверить теоретические предположения о характере развития и вымирания популяции.

Как будет развиваться популяция, если $x_0 < 1$ или $x_0 > 1$.

С этой целью введем в модель уравнение:

$$\frac{dy}{dt} = y(1-y) \qquad y(t=0) > 1.$$

Второе уравнение не зависит от первого, переменные ***x(t = 0)*** и ***y(t = 0)*** имеют разные значения. Результат моделирования представлен на рис. 2.4.1. Введение в модель второго уравнения позволяет анализировать процесс развития популяции при различных начальных условиях в рамках одно MVS-модели.

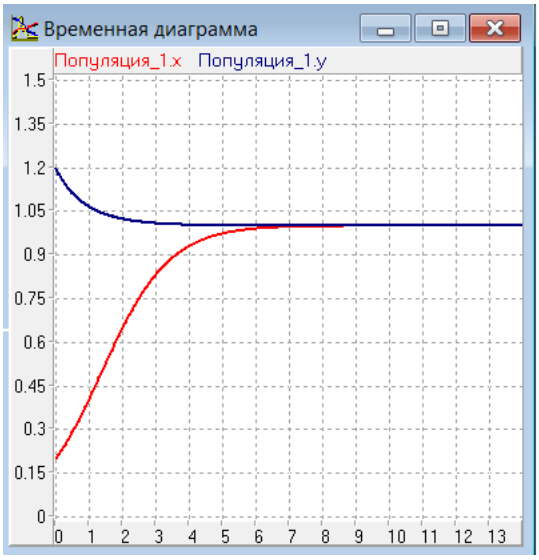

Рис. 2.4.1. Результат моделирования

Исследуйте точки *x = 0* и *x = 1* на устойчивость.

2.5. Моделирование межвидовой конкуренции

Пусть в экологической системе существует два вида, которые эксплуатируют общий жизненный ресурс (общая пища или территория существования) и находятся в конкурентной борьбе за его использование. Развитие каждого вида описывается модифицированным логистическим уравнением, которое учитывает взаимодействие видов за счет изменения экологической емкости окружающей среды. Модель построена для непрерывного процесса взаимодействия и имеет следующий вид:

$$\begin{cases} \dfrac{dx}{dt} = R_1 x\left(\dfrac{k_1 - x - \alpha y}{k_1}\right), \\ \dfrac{dy}{dt} = R_2 y\left(\dfrac{k_2 - y - \beta x}{k_2}\right). \end{cases}$$

Здесь R_1, R_2 - коэффициенты размножения, k_1, k_2 - параметры, характеризующие экологическую емкость среды для каждого вида соответственно. Коэффициент α – отражает влияние вида y на x. В свою очередь, коэффициент β отражает влияние вида x на вид y. Естественно, что при нулевых значениях α и β взаимодействие между видами отсутствует.

После приведения модели к безразмерному виду по методу неопределенных масштабов уравнения будут представлены соотношениями:

$$\begin{cases} \dfrac{dx}{dt} = x\left(1 - x - \varepsilon_1 y\right) \\ \dfrac{dy}{dt} = \gamma y\left(1 - y - \varepsilon_2 x\right) \end{cases}, \quad \text{где} \quad \varepsilon_1 = \frac{\alpha k_2}{k_1}; \quad \varepsilon_2 = \frac{\beta k_1}{k_2}; \quad \gamma = \frac{R_2 k_1}{R_1 k_2}.$$

Дополнив систему уравнений начальными условиями:

$$x(t = 0) = x_0, \ \ y(t = 0) = y_0,$$

получим безразмерную модель взаимодействия двух популяций.

Провести исследование развития системы из двух популяций в зависимости от соотношения параметров. Моделирование выполняется

в среде MVS по аналогии с предыдущими работами. Динамику развития популяций необходимо отразить в виде временной диаграммы.

На основе данной модели необходимо проследить следующие варианты развития популяций:

$\varepsilon_1 < 1, \varepsilon_2 < 1$ с экологической точки зрения это означает, что межвидовая конкуренция слабее, чем внутривидовая. Вывод: возможно совместное существование двух видов. Докажите путем выполнения компьютерных экспериментов, что независимо от начальных условий при неизменных значениях параметров с течением времени система приходит к одному и тому же состоянию.

Так при $\varepsilon_1 > 1, \varepsilon_2 > 1$ межвидовая конкуренция является главным фактором, в зависимости от начальных условий выживает один из видов. В ходе экспериментов с моделью докажите это утверждение.

Если $\varepsilon_1 > 1, \varepsilon_2 < 1$; $\varepsilon_1 < 1, \varepsilon_2 > 1$ – один из видов вытесняет другой.

Все представленные варианты развития системы двух конкурирующих популяций необходимо подтвердить или опровергнуть в ходе выполнения численных экспериментов с моделью. Кроме того, необходимо установить возможные стационарные состояния системы и проверить их устойчивость. С этой целью необходимо выявить такие начальные значения $x(t = 0) = x_0$ и $y(t = 0) = y_0$, которые дают решение, неизменное во времени. Один из таких режимов очевиден: $x(t = 0) = 0$, $y(t = 0) = 0$. Исследуйте поведение системы в том случае, когда начальные значения переменных мало отличаются от стационарных.

2.6. Моделирование системы управления

Системы управления предназначены для поддержания определенного режима функционирования объекта или его заданного состояния. Системы управления с обратной связью учитывают состояние объекта при регулировании (рис. 2.6.1).

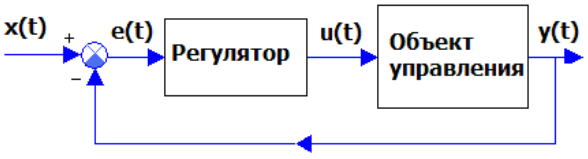

Рис. 2.6.1. Схема системы управления с обратной связью

Здесь: $x(t)$ – управляющее воздействие, $e(t)=x(t) - y(t)$, $u(t)$ – воздействие регулятора на объект управления, $y(t)$ – регулируемый параметр. Примерно так происходит стабилизация спутника на орбите с помощью импульсных воздействий.

Математическая модель объекта управления имеет вид:

$$\frac{d^2y(t)}{dt^2} + a \cdot \frac{dy(t)}{dt} + b \cdot y(t) = k \cdot u(t) \ .$$

Модель регулятора суть следующее:

$$u(t) = u_0 \cdot sign(x(t) - y(t)) \ .$$

Средствами MVS построить модель системы управления (рис. 2.6.2).

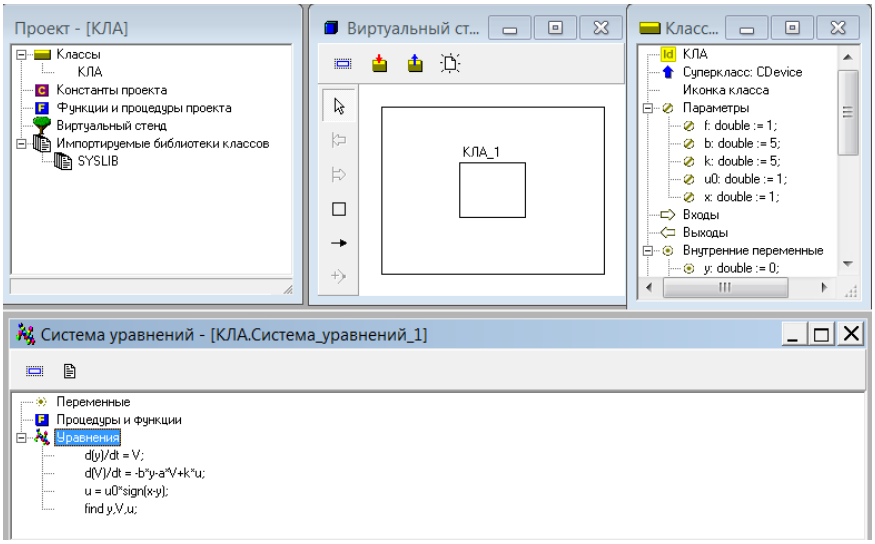

Рис. 2.6.2. MVS-модель системы управления

Модель всей системы управления описывается системой дифференциальных уравнений:

$$\frac{dy}{dt} = V; \quad \frac{dV}{dt} = -b \cdot y - a \cdot V + k \cdot u; \quad u = u_0 \cdot sign(x - y)$$

Начальные условия: $y(t = 0) = 0$, $V(t = 0) = 0$.

Построить временную диаграмму переходного процесса для переменной $y(t)$ и воздействия регулятора $u(t)$. Построить MVS-модель объекта управления, на который не действует управление $u(t)$.

Значение параметров: $a = 1;\ b = 5;\ k = 5;\ u_0 = 1;\ x = 1.$ Исследовать влияние значения параметра k на поведение системы. Выявить роль параметра x.

Построить временную диаграмму переходного процесса для переменной $y(t)$ и воздействия регулятора $u(t)$. Построить MVS-модель объекта управления, на который не действует управление $u(t)$.

Значение параметров: $a = 1;\ b = 5;\ k = 5;\ u_0 = 1;\ x = 1.$

Исследовать влияние значения параметра k на поведение системы. Выявить роль параметра x.

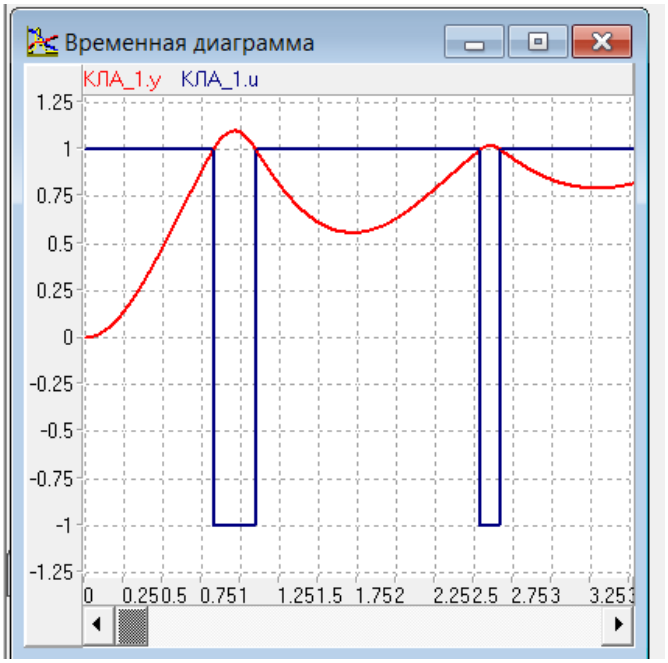

Рис. 2.6.3. Результат моделирования

2.7. Модель шарика, падающего на пружину

Шарик падает на свободный конец пружины, которая имеет длину L. Пружина установлена вертикально на плоскости (рис 2.7.1). В начальный момент времени скорость шарика $V_y(t = 0) = 0$, а падение начинается с высоты H относительно плоскости $(H>L)$.

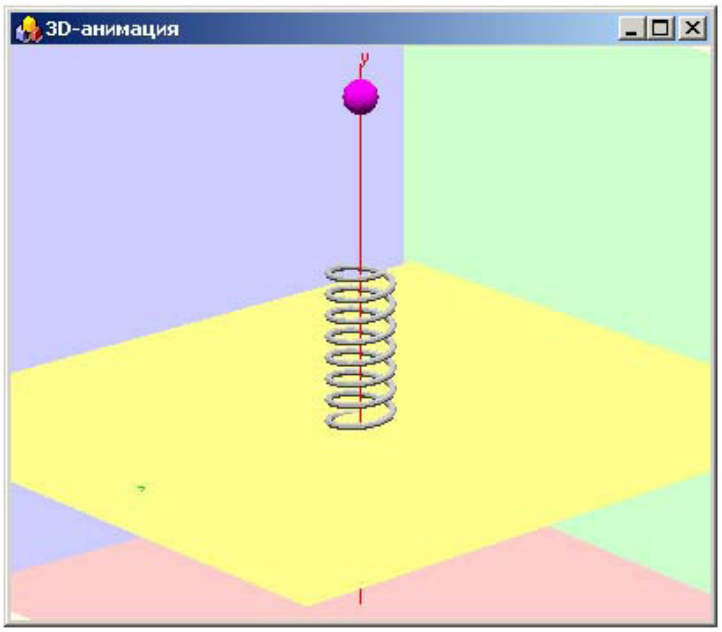

Рис. 2.7.1. 3D модель системы

Система уравнений движения шарика имеет вид:

$$\begin{cases} \dfrac{dy}{dt} = V_y \\ \dfrac{dV_y}{dt} = if\ \ y > L\ then - g\ else\ K \cdot (L - y_s) / L - g, \\ y_s = if\ \ y > L\ then\ L\ else\ y, \end{cases}$$

где y_s – координата свободного конца пружины, K – коэффициент жесткости пружины. Инерция самой пружины в задаче не учитывается.

Коэффициент жесткости пружины и параметры шарика таковы, что удар шарика о плоскость не происходить, всегда $y > 0$ (рис. 2.7.2).

Построить модель системы средствами MVS. Движение шарика и пружины отобразить на временной и фазовой диаграммах (рис. 2.7.2) и средствами 3D-анимации (рис. 2.7.1).

Начальные условия назначить самостоятельно. Определить значение H, при котором шарик не будет отскакивать от пружины.

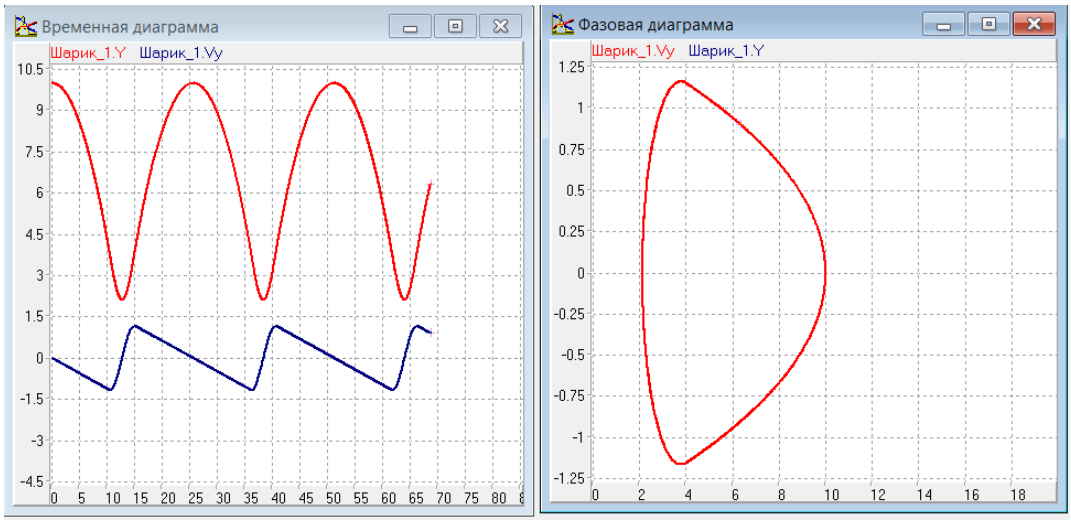

Рис. 2.7.2. Временная и фазовая диаграммы

2.8. Моделирование обучения

Пусть ***x(t)*** объем знаний, накопленных учащимся к моменту времени ***t***, объем накопленных умений рассуждать, решать задачи, разбираться в излагаемом преподавателем материале – ***y(t)***; доля времени, отведенного на накопление знаний – ***u***. Любое знание состоит частично из «информации» и частично из «умения». Можно считать, что увеличение объема знаний учащегося пропорционально потраченному на это времени ***udt*** и накопленным умениям ***y(t):***

$$\frac{dx(t)}{dt} = k_1 uy(t)$$

где коэффициенты $k_1 > 0$ и $k_2 > 0$ зависят от индивидуальных особенностей учащегося.

Увеличение знаний за то же время пропорционально потраченному на это времени ***(1-u)dt***, имеющимся умениям ***y(t)*** и знаниям ***x(t)***. Следовательно,

$$\frac{dy(t)}{dt} = k_2(1-u)x(t)y(t)$$

Замена переменных $z(t)=k_2x(t)$, $w(t)= k_1 k_2y(t)$ дает более простую систему уравнений:

$$\frac{dz}{dt} = uw; \qquad \frac{dw}{dt} = (1-u)zw.$$

Построить MVS-модель учебного процесса, исследовать влияние начальных значений ***w, z*** и параметра ***u***. Доказать, что максимальная скорость обучения наблюдается, когда на одну лекцию приходиться два практических занятия или семинара: ***u=1/3.*** MVS-модель представлена на рис. 2.8.1, результат моделирования представлен на рис. 2.8.2.

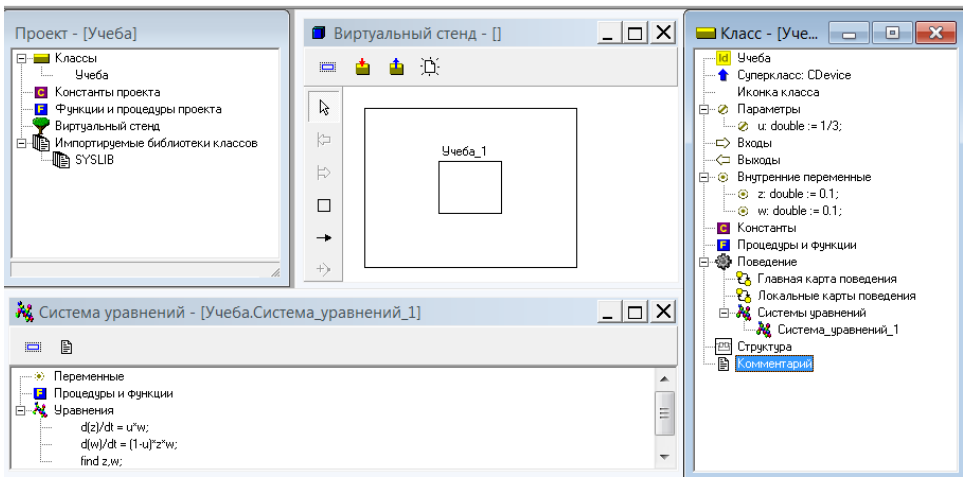

Рис. 2.8.1. MVS-модель процесса обучения

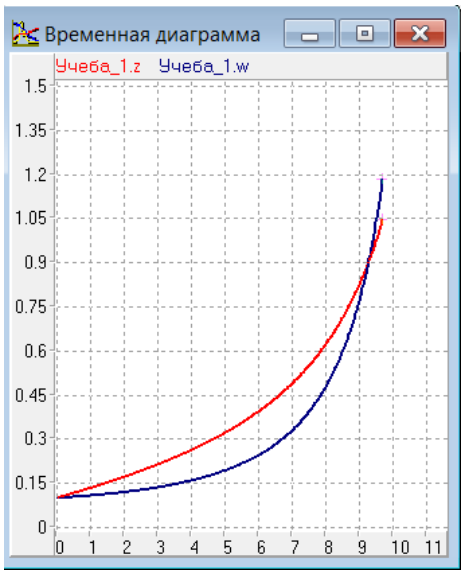

Рис. 2.8.2. Результат моделирования обучения при *u=1/3*

2.9. Распространение инноваций

Одним из главных факторов, определяющих скорость процессов распространения нововведений (инноваций), является межличностное общение между сторонниками данной новинки и теми, кто еще колеблется или ничего не слышал о нововведении. Если обозначить численность людей, принявших инновацию к моменту времени ***t***, через ***y***, то число лиц, которых, в принципе, можно сагитировать, равно $M - y$, где ***M*** – емкость рынка, максимально возможное число лиц, принимающих нововведение.

Можно считать, что прирост числа сторонников новинки пропорционален числу встреч между сторонниками новинки и сомневающимися. Число таких встреч пропорционально произведению $(M-y)y$. Если отразить данные рассуждения количественно, то получим следующее уравнение для прироста числа сторонников инновации:

$$\frac{dy}{dt} = a(M - y)y \quad ,$$

где a – коэффициент пропорциональности. MVS-проект для данной задачи имеет вид:

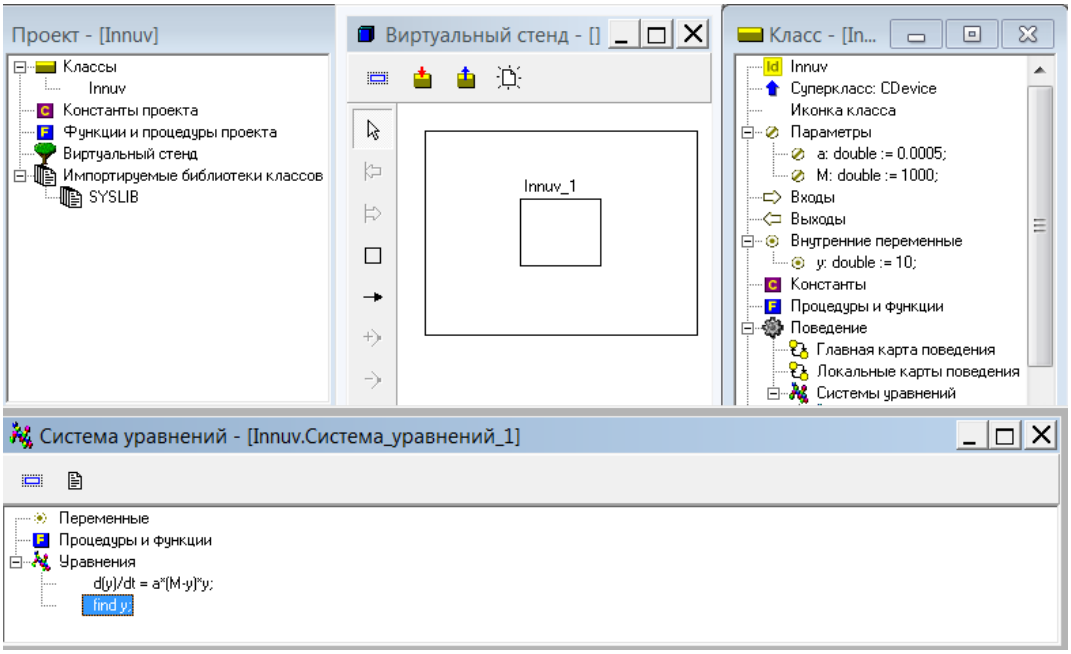

Рис. 2.9.1. MVS-модель распространения инноваций

Подобная модель хорошо описывает процессы обновления техники, смену технологий, эволюционные процессы в социокультурной сфере, распространение движений протеста в обществе и даже распространение слухов.

Построить MVS-модель распространения инноваций (рис. 2.9.1) и исследовать ее свойства.

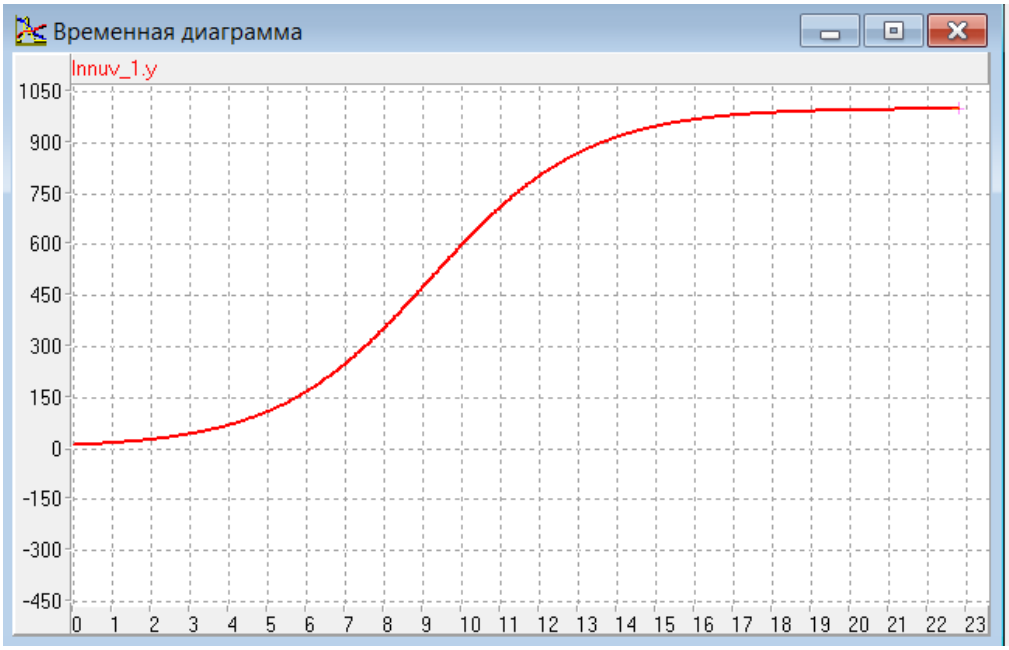

Рис. 2.9.2. Результат моделирования

Глава 3. МОДЕЛИРОВАНИЕ ДИСКРЕТНО-НЕПРЕРЫВНЫХ СИСТЕМ

3.1. Создание гибридной модели движения тела по баллистической траектории

В начальный момент времени тело находится в точке с координатами: x(t = 0) = 0, y(t = 0) = 0. Тело начинает движение под углом α со скоростью V_0. (рис. 3.1.1)

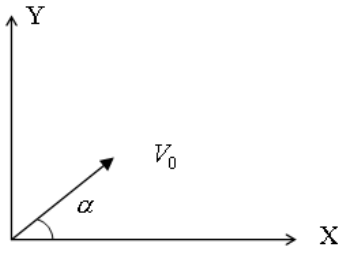

Рис. 3.1.1. Схема начала движения

Модель построена на основе второго закона механики и кинематических уравнений. В безразмерном виде уравнения модели имеют следующий вид [3]:

$$\frac{dVy}{dt} = -1 - k \cdot Vy; \cdot \qquad Vx(t=0) = \cos(\alpha);$$
$$\frac{dVx}{dt} = -k \cdot Vx; \qquad Vy(t=0) = \sin(\alpha);$$

$$\frac{dx}{dt} = Vx\ , \quad \frac{dy}{dt} = Vy \quad \cdot \quad x(t=0) = 0, \quad y(t=0) = 0\ .$$

Здесь ***t*** – время, α – угол к горизонту при начале движения, ***x, y*** – координаты движущегося тела, ***Vx*** и ***Vy*** – скорости движения тела по оси **x** и по оси ***y***, ***k*** – коэффициент сопротивления среды, в которой происходит движение.

Целью моделирования является построение траектории движения в прямоугольной системе координат. Модель является гибридной моделью, так как должна остановить движение при падении на Землю.

Построение компьютерной модели выполняется в среде MVS. В случае $k = 0$ траекторией будет парабола. Этот результат можно использовать для проверки компьютерной модели. MVS-модель представлена на рис. 3.1.2. На рис. 3.1.3 представлена система уравнений движения в проекте «**Баллистика**». В модели угол α задается в градусах. Функция MVS ***rad***(α) переводит значение угла α из градусов в радианы.

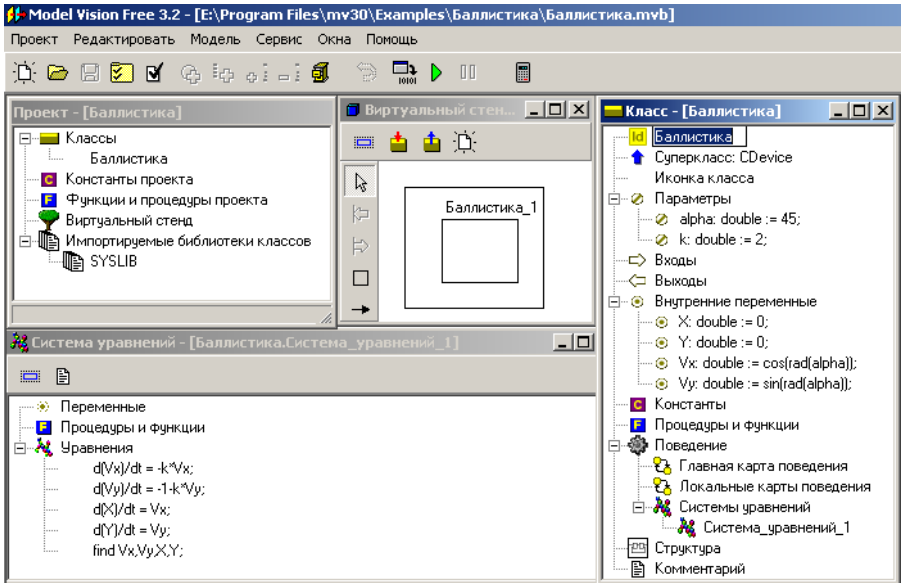

Рис. 3.1.2. MVS-модель движения тела по баллистической траектории

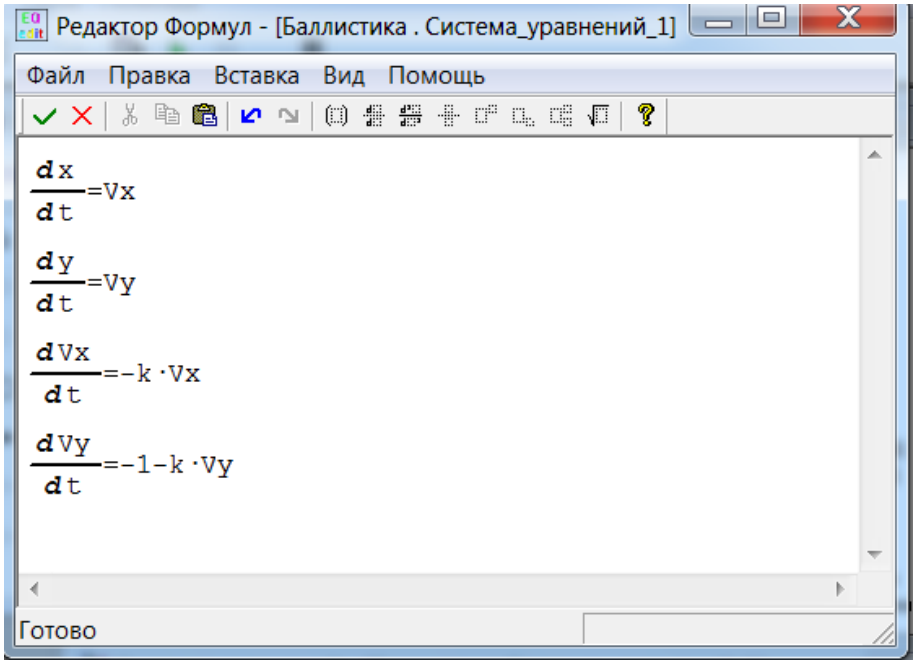

Рис. 3.1.3. Система уравнений движения в проекте «Баллистика»

По результатам моделирования можно установить влияние сопротивления на вид траектории движения тела. При отсутствии сопротивления ($k = 0$) траектория является параболой. Траектория движения тела строится в виде фазовой диаграммы ***Y(X)*** (рис. 3.1.4).

Для остановки движения тела при падении на Землю необходима карта поведения.

Задать другие значения, например, для параметров ***k*** и ***alpfa*** можно в окне «**Баллистика_1**» (рис. 3.1.4) средствами контекстного меню при выделении соответствующего параметра, выполнения команды «**Изменить..**», и вводом нового значения параметра.

Карта поведения в проекте «**Баллистика**» должна остановить движение тела при его падении на Землю. В окне «**Класс-[Баллистика]**» (рис. 3.1.2) выделим «**Главную карту поведения**» и с помощью контекстного меню выполним команду «**Изменить**» (рис. 3.1.5).

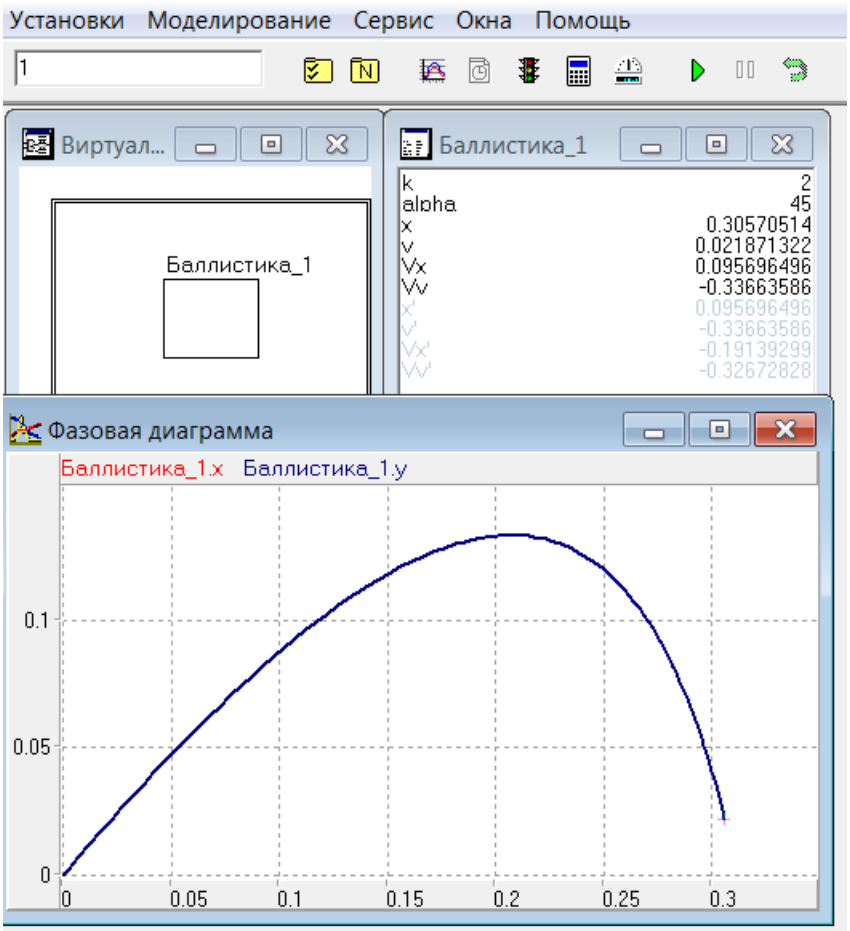

Рис. 3.1.4. Визуальная модель и фазовая диаграмма с траекторией движения тела

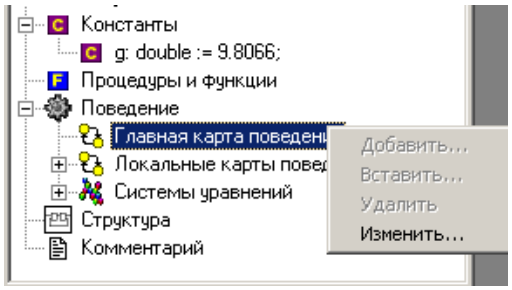

Рис. 3.1.5. Открытие главной карты поведения

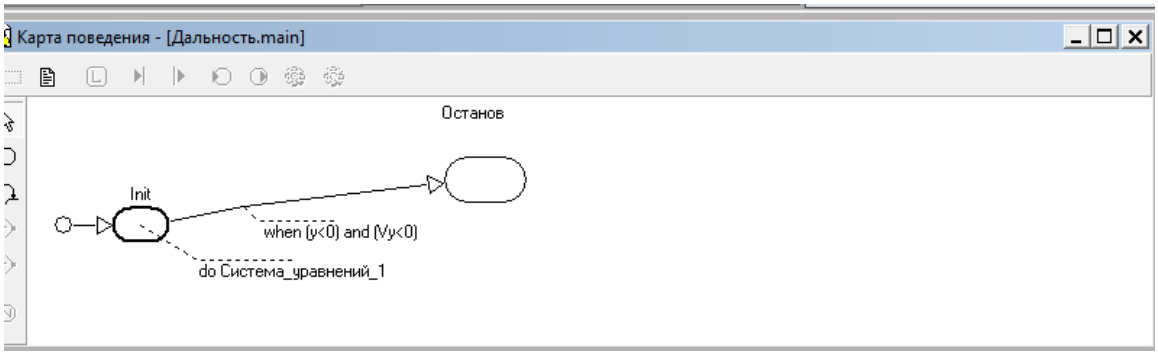

Рис. 3.1.6. Главная карта поведения

Узел «**Init**» ни удалить, ни переименовать нельзя. Свяжем методом drag and dropc с узлом «**Init**» «**Систему_уравнений_1**» (рис. 3.1.6). Создадим новый узел «**Останов**» (рис. 3.1.6). Кнопки редактирования карты поведения показаны на рис. 3.1.7.

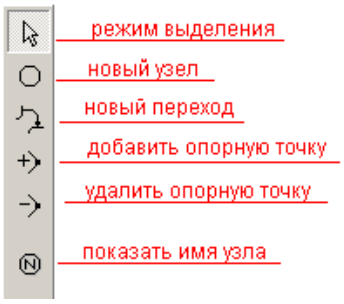

Рис. 3.1.7. Левая панель кнопок редактирования карты поведения

В дальнейшем придется связать **переходом** узел «**Init**» главной карты поведения с узлом «**Останов**». Теперь при запуске модели будет выполняться «**Система_уравнений_1**».

Чтобы создать новый узел на карте поведения нужно:

С помощью кнопки ○ перевести редактор в режим создания нового узла (признаком этого режима служит появление крестообразного курсора мыши);

Поставить крестообразный курсор в место, где должен располагаться левый верхний угол нового узла, нажать левую кнопку мыши и, не отпуская ее, перемещать мышь вправо и вниз. Вместе с мышью изменяется изображение узла. Положение курсора соответствует правому нижнему углу узла. В нужном положении следует отпустить кнопку мыши.

Новый узел будет по умолчанию иметь имя «Node_1, 2, ...». Если новый узел был первым в карте поведения, то ему будет автоматически сопоставлен начальный переход, то есть первый узел по умолчанию считается начальным.

Чтобы изменить имя узла щелкните на нем дважды мышью. В этом проекте необходимо создать узел «**Останов**».

Чтобы создать переход, необходимо:

• с помощью кнопки перейти в режим создания перехода (признаком этого режима служит появление крестообразного курсора мыши);

• подвести курсор мыши на изображение исходного узла (например, «**Полет**»). Курсор при этом сменится на изображение креста в круге. Затем нажать левую кнопку мыши и, не отпуская ее, переместить мышь на изображение конечного узла (например, «**Останов**»). Когда курсор при этом сменится на изображение креста в круге, отпустить кнопку. Исходный и конечный узел может быть один и тот же. Получим переход из узла «**Init**» в узел «**Останов**». Условие остановки движения **y <= 0 и *Vy* < *0***. Это условие является условием срабатывания перехода из узла «**Init**» в узел «**Останов**». Задание условия срабатывания перехода представлено на рис. 3.1.8, задается после нажатия кнопки (см. рис. 3.1.6).

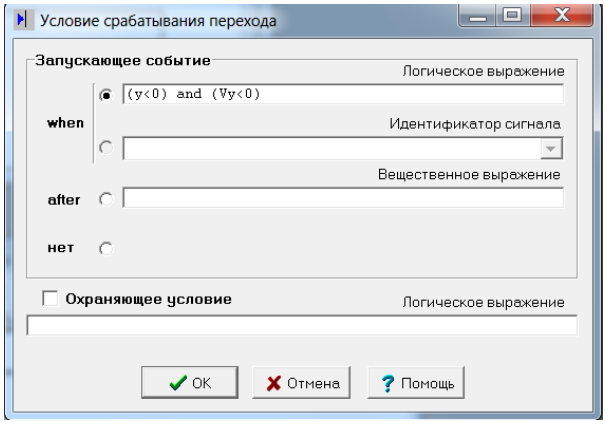

Рис. 3.1.8. Задание условия срабатывания перехода из узла «Init» в узел «Останов»

Структура карты поведения «рисуется» с помощью кнопок по рис. 3.1.7. Карта поведения представляет собой граф, узлы которого соответствуют состояниям моделируемой системы с непрерывным поведением, а дуги соответствуют переходам из одного состояния в другое.

Узлу «**Init**» соответствует непрерывное поведение, которое описывается «**Систеиой_уравнений_1**». Свяжем методом **drag and drop** «**Систему_уравнений_1**» из окна класса «**Баллистика**»с узлом «**Init**» (рис. 3.3.6).

Переходы изображаются ломаной линией со стрелкой, указывающей направление перехода. Действия перехода здесь нет. Окно редактора карты поведения имеет две панели кнопок: левую и верхнюю. Левая панель содержит кнопки с фиксацией, связанные с различными режимами работы графического редактора карты поведения. Стандартным является режим выделения, переходом в этот режим, как правило, заканчиваются все операции.

Входное действие в узле «**Останов**» задается при нажатии кнопки  при выделенном узле «**Останов**» (рис. 3.1.9).

Результат моделирования представлен на рис. 3.1.10.

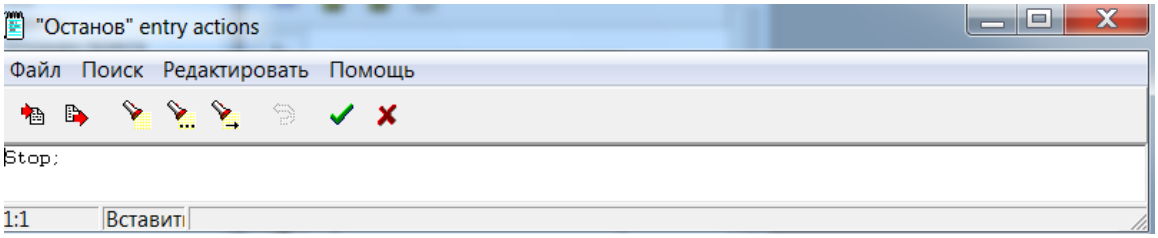

Рис. 3.1.9. Задание входных действий в узле «Останов»

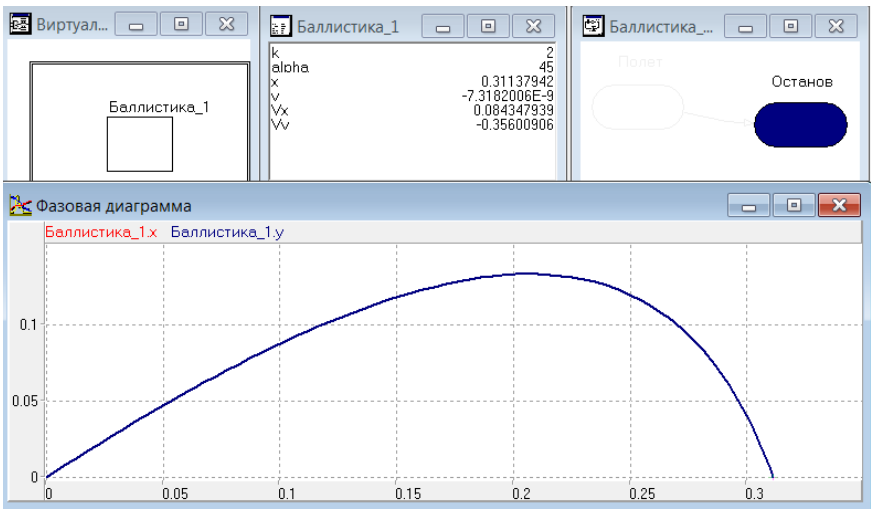

Рис. 3.1.10. Результат моделирования

По значениям переменных видно, что останов полета произошел при выполнении условия приземления. Значения угла α и параметра ***k*** для проведения следующего эксперимента можно изменить, как описывалось ранее в разделе «**Создание простой непрерывной модели**».

3.2. Определение наибольшей дальности полета

В данной задаче используется математическая модель, аналогичная той, которую мы применяли при построении баллистической траектории. Эта модель в безразмерном виде представлена ниже:

$$\frac{dVy}{dt} = -1 - k \cdot Vy; \cdot \quad \frac{dVx}{dt} = -k \cdot Vx; \quad \frac{dx}{dt} = Vx\ , \quad \frac{dy}{dt} = Vy.$$

$$x(t=0) = 0, \quad y(t=0) = 0\ ,\ Vx(t=0) = \cos(\alpha);\ Vy(t=0) = \sin(\alpha).$$

Целью моделирования является определение значения угла α (рис. 3.1.1), при котором будет достигнута максимальная дальность полета. Примерно такая задача может стоять перед расчетом артиллерийского орудия.

При отсутствии сопротивления движению (***k = 0)*** задача имеет известное решение: α = ***45***°. Задача теперь состоит в том, чтобы определить оптимальное значение угла α, которое обеспечивает максимальную дальность полета при наличии сопротивления. Модель для решения этой задачи строится в среде пакета MVS. Первоначально средствами MVS необходимо построить модель движения тела «**Дальность**» (рис. 3.2.1).

В отличие от предыдущей модели переменная ***x*** объявляется как **выходная переменная**. В систему уравнений должна быть добавлена переменная $z = (1 - x)^2$ – величина, которая является разностью между возможным максимальным расстоянием полета при ***k = 0*** и реальным значением расстояния полета, которое пролетело тело.

Расстояние при ***k = 0*** является максимальным (рис. 3.2.2). Таким образом, для определения максимального расстояния, требуется найти минимальное значение ***z***.

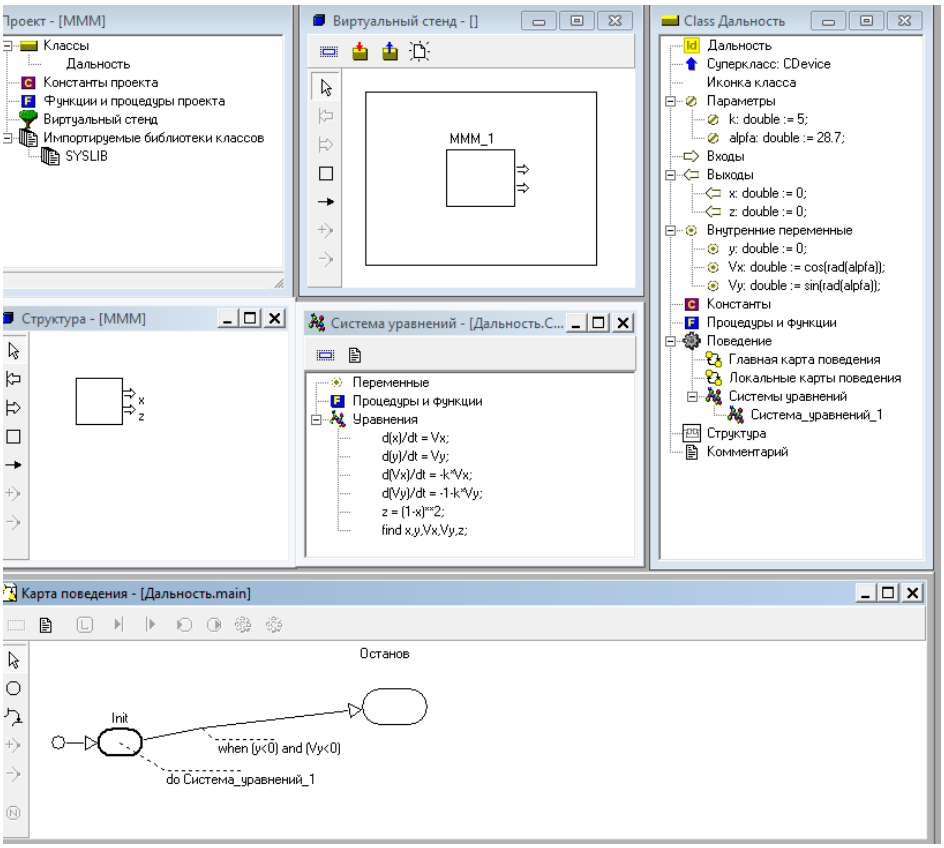

Рис. 3.2.1. Модель движения тела «Дальность»

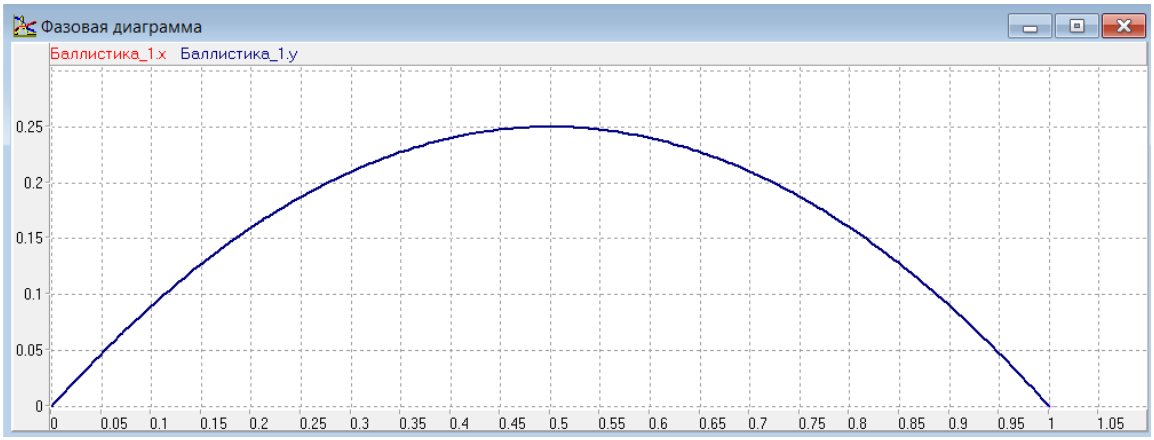

Рис. 3.2.2. Траектория полета при $k = 0$

Z так же объявляется **выходной переменной.** С переменными ***x*** и ***z*** будет взаимодействовать пакет **оптимизации** MVS. Дополнительное окно «**Структура**» появится автоматически после объявления переменных выхода ***x*** и ***z*** (рис. 3.2.3).

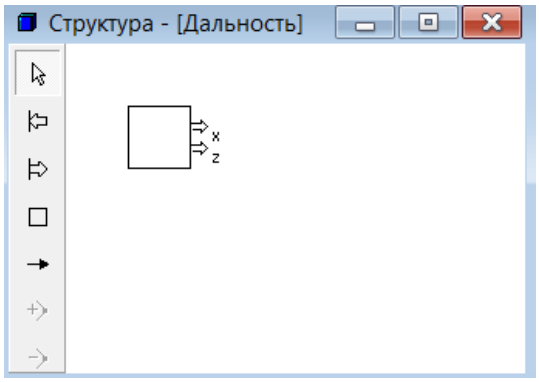

Рис. 3.2.3. Дополнительное окно «Структура»

Для решения данной задачи необходимо построить карту поведения и предусмотреть останов полета, если выполняется условие приземления: **y<= 0 and Vy < 0** (рис. 3.2.1). Именно для этого момента времени в подсистеме оптимизации MVS будет определяться максимальное расстояние, которое пролетело тело, и вычисляться значение **z**.

После выполнения данных действий можно приступить к решению задачи определения значения угла α, которое обеспечит максимальную дальность полета. При решении задачи будем использовать пакет оптимизации MVS. Для применения пакета оптимизации в окне проекта (рис. 3.2.1) выполним действия: пункт меню «**Сервис**» – «**Оптимизация**» (рис. 3.2.4). В результате откроется окно подсистемы оптимизации (рис. 3.2.5), поля которого: «**Параметры**», «**Функционал**», «**Ограничения**», «**Вычислять функционал в момент...**»; необходимо заполнить с использованием метода перетаскивания в соответствии с рис. 3.2.5.

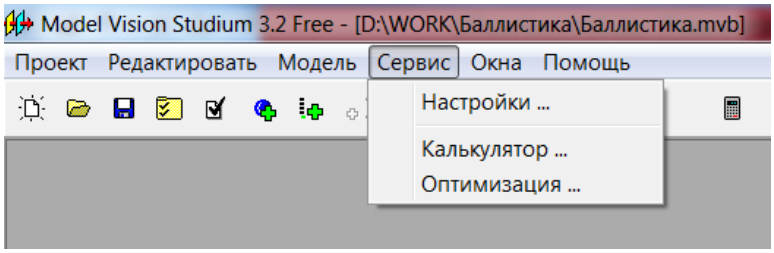

Рис. 3.2.4. Запуск подсистемы оптимизации

Решение задачи поиска **минимального** значения ***z*** будет выполнено после нажатия кнопки . Если задать ***k = 0,*** то находится оптимальное значение α **= 45°**.

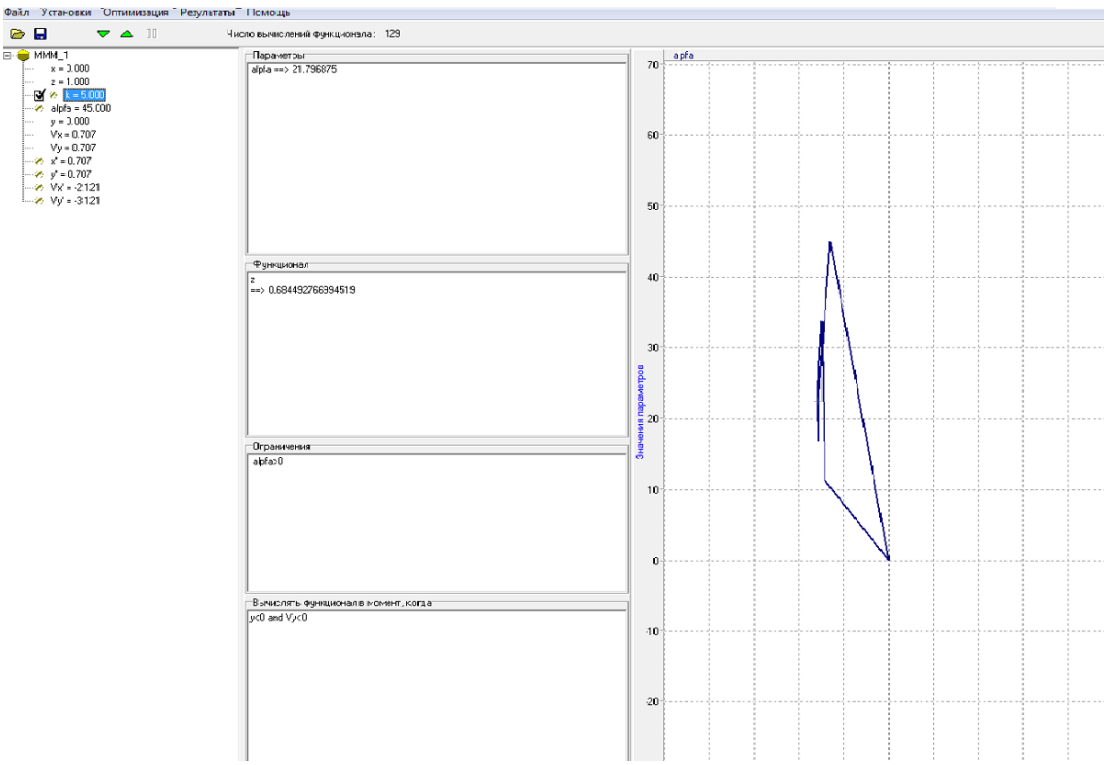

Рис. 3.2.5. Работа подсистемы оптимизации

Вторая задача, которую можно решить подобным способом, попадание в цель с заданными координатами. Модель для этой задачи представлена на рис. 3.2.6.

После выполнения действий, которые представлены в предыдущей задаче, можно приступить к решению задачи определения значения угла α, которое обеспечит попадание в цель. При ее решении также будем использовать пакет оптимизации MVS. Для применения пакета оптимизации в окне проекта (рис. 3.2.6) выполним действия: пункт меню «**Сервис**» – «**Оптимизация**». В результате откроется окно подсистемы оптимизации (рис. 3.2.7), поля которого: «**Параметры**», «**Функционал**», «**Ограничения**», «**Вычислять функционал в момент...**», необходимо заполнить с использованием метода перетаскивания в соответствии с рис. 3.2.7.

Решение задачи поиска **минимального** значения z будет выполнено после нажатия кнопки  . Предварительно в пункте меню «Установки» необходимо выбрать метод оптимизации **“Random search”** (рис. 3.2.8). Процесс поиска минимального значения z и угла α, при котором это значение достигается, представлен на рис. 3.2.5.

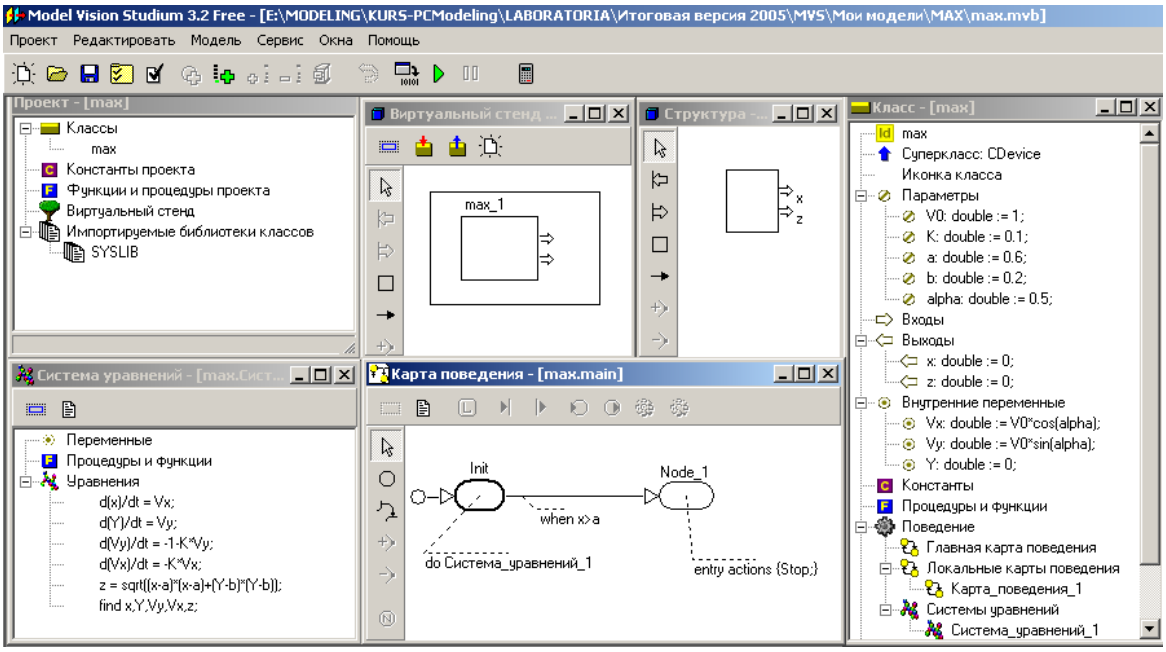

Рис. 3.2.6. Модель движения тела

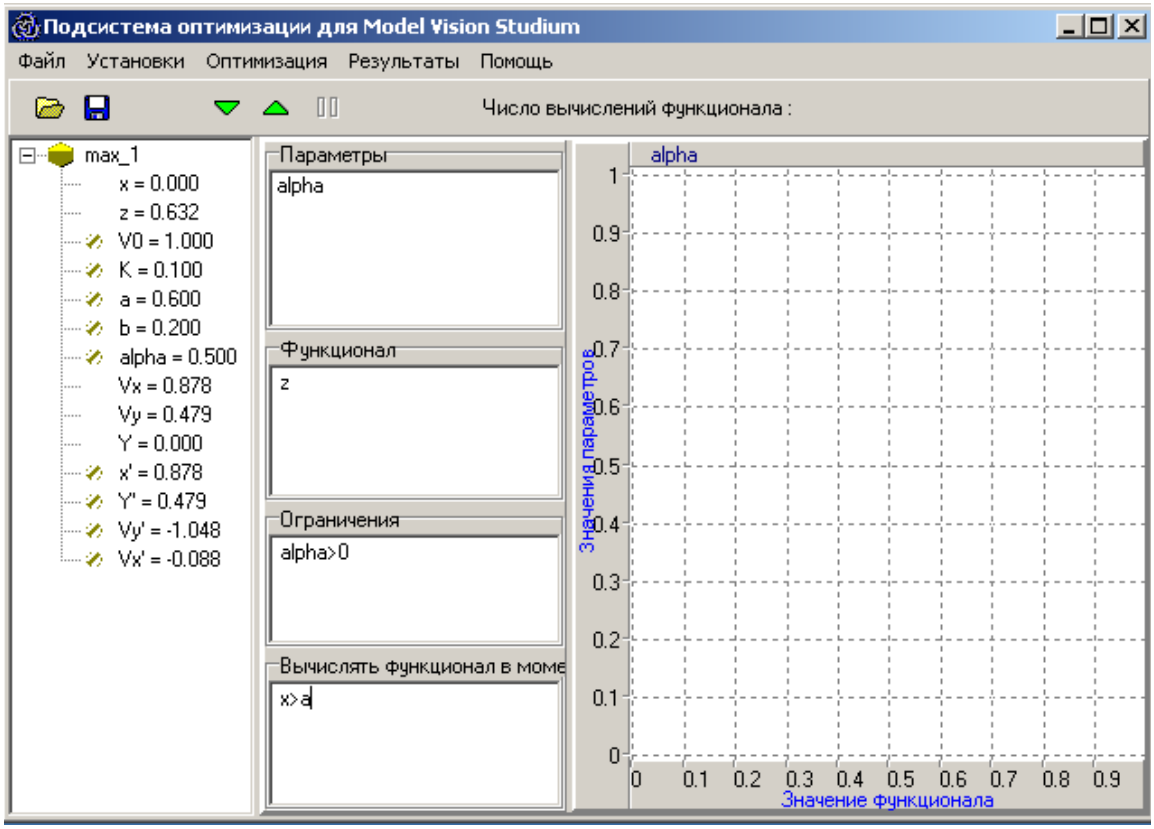

Рис. 3.2.7. Начальное состояние подсистемы оптимизации

Далее выполним запуск модели , дополним модель фазовой диаграммой и проведем эксперимент с начальными значениями параметров по рис. 3.2.6. Его результат представлен на рис. 3.2.7. Очевидно, что при первоначально выбранном значении параметра α попадания в цель не происходит.

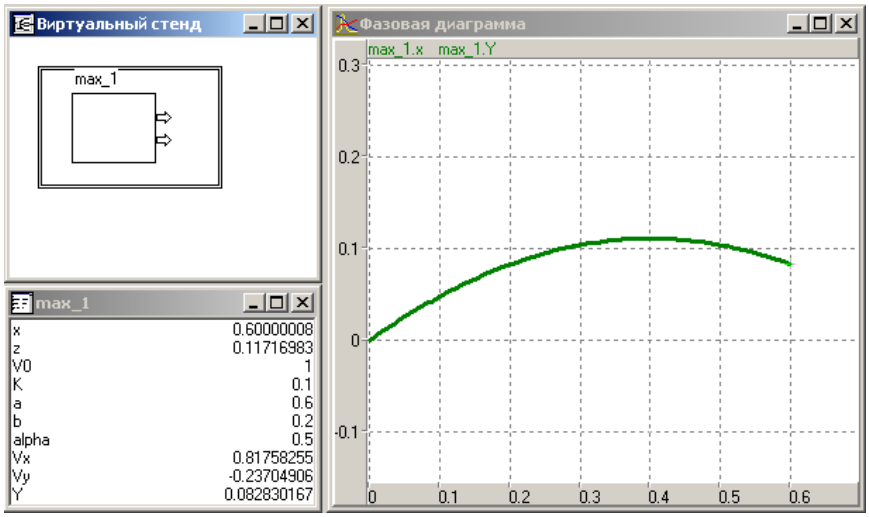

Рис. 3.2.7. Результат моделирования для начальных значений параметров

После проведения данного эксперимента необходимо закрыть окно визуальной модели по схеме: «**Рестарт**» – «**Установки**» – «**Выход**». На вопрос о сохранении изменений модели необходимо ответить: «**Да**».

После выполнения указанных действий можно приступить к решению задачи определения значения угла α, которое обеспечит попадание в цель. При решении задачи также будем использовать пакет оптимизации MVS. Для применения пакета оптимизации в окне проекта (рис. 3.2.1) выполним действия: пункт меню «**Сервис**» – «**Оптимизация**». Выбор метода оптимизации необходимо провести в соответствии с рис. 3.2.8. В результате откроется окно подсистемы оптимизации (рис. 3.2.9), поля которого: «**Параметры**», «**Функционал**», «**Ограничения**», «**Вычислять функционал в момент...**»; необходимо заполнить с использованием метода drag and drop.

Подсистема оптимизации пакета MVS определит такое значение параметра, α при котором значение переменной z будет минимальным. Результат поиска оптимального значения представлен на рис. 3.2.9.

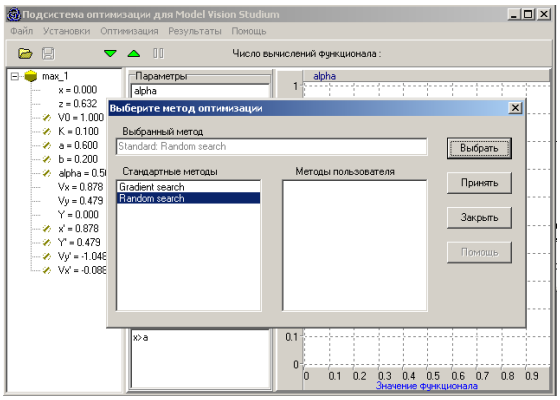

Рис. 3.2.8. Выбор метода оптимизации

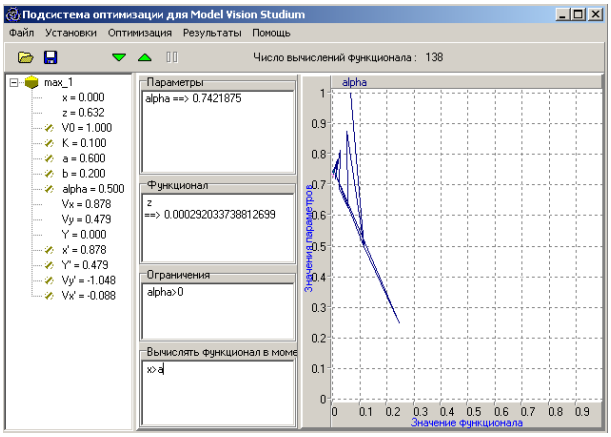

Рис. 3.2.9. Результат поиска оптимального значения угла α

Задать найденное значение параметру α и провести моделирование движения тела повторно. Результат представлен на рис. 3.2.10.

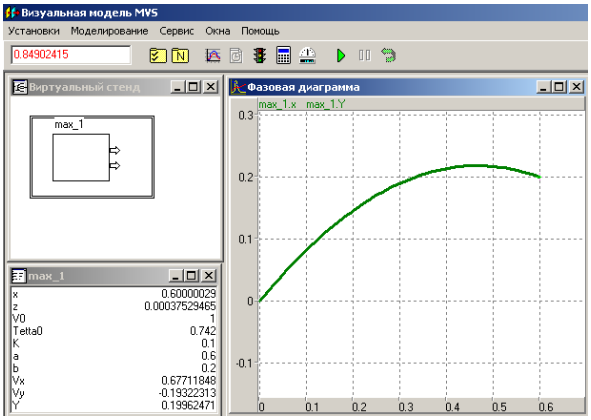

Рис. 3.2.10. Траектория движения тела при оптимальном значении угла α

В экспериментах с моделью можно задать другие значения параметра ***k*** и решить задачу оптимизации.

3.3. Модель обучения по Р.В. Майеру

Приращение знаний происходит в первом приближении в соответствие с зависимостью [5]:

$$\frac{dZ}{dt} = k \cdot \alpha \cdot Z \cdot (U - Z) - \gamma \cdot Z.$$

Z – объем знаний, γ – коэффициент забывания, α – коэффициент обучения, ***U*** – уровень требований в обучении, ***k*** – индивидуальный коэффициент обучаемости.

Построить MVS-модель обучения. При построении модели использовать карту поведения. Параметр ***T*** является границей действия требований в обучении, если ***Time*** > ***T***, следует переход из узла **Init** на новый узел **Node_1** карты поведения – это условие срабатывания перехода (рис. 3.3.1).

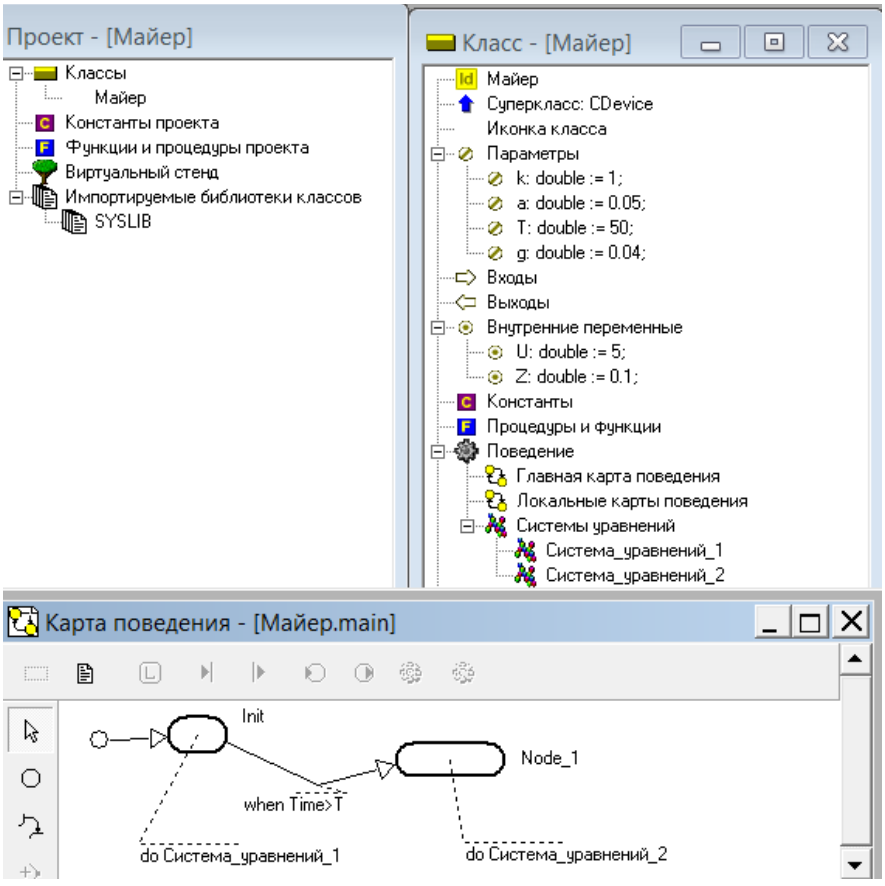

Рис. 3.3.1. MVS-модель обучения по Р.В Майеру.

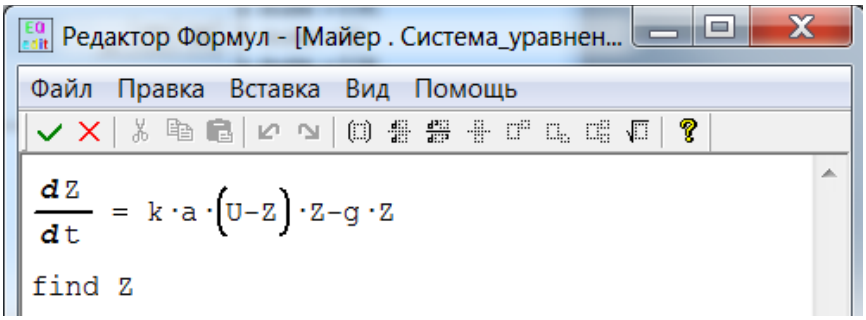

Рис. 3.3.2. Система_уравнений_1. Time<T

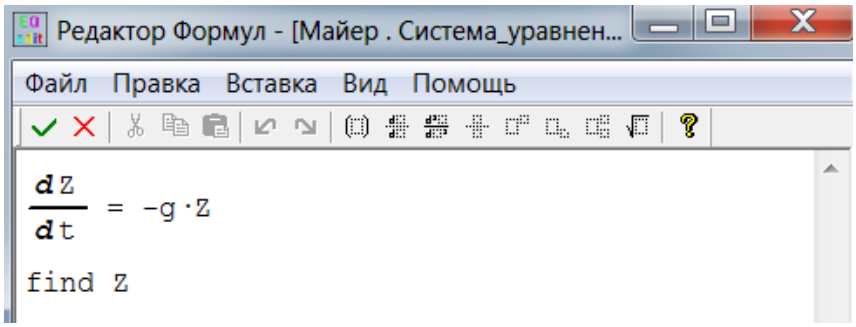

Рис. 3.3.3. Система_уравнений_2. Time>T

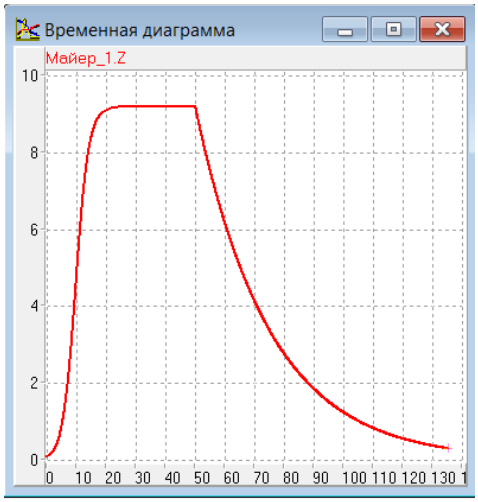

Рис. 3.3.4. Развитие обучения

Объясните полученные результаты. Модель не учитывает известное положение: «Образование – это то, что остается, когда все выученное забыто».

3.4. Прыгающий мячик в среде с сопротивлением движению

Разработать MVS-модель прыгающего мячика. Это гибридная система. Абсолютно упругий мячик (материальная точка) падает с высоты ***H*** на абсолютно твердую горизонтальную плоскость.

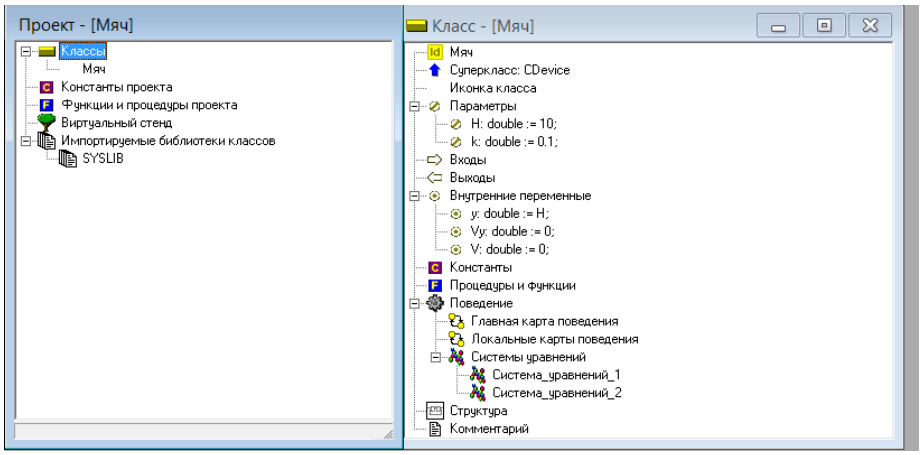

Рис. 3.4.1. MVS-модель прыгающего мяча

Непрерывный процесс падения мячика под действием сил тяжести и сопротивления описывается системой безразмерных уравнений:

$$\frac{dy}{dt} = V_y, \quad \frac{dV_y}{dt} = -1 - k * V_y. \quad y(t=0) = H, \; V_y(t=0) = 0,$$

где y – вертикальная координата мячика относительно плоскости, V_y – вертикальная скорость мячика.

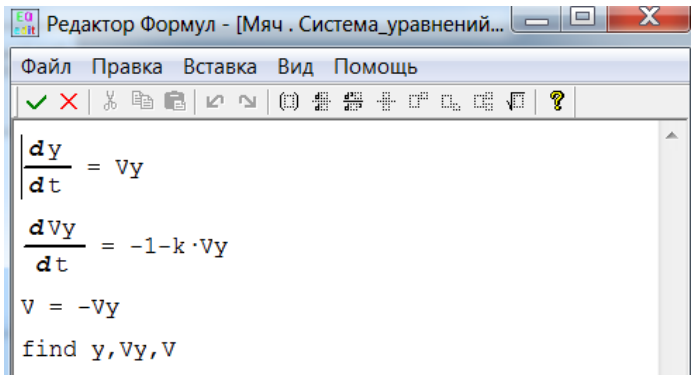

Рис. 3.4.2. Система уравнений, описывающая падение мяча

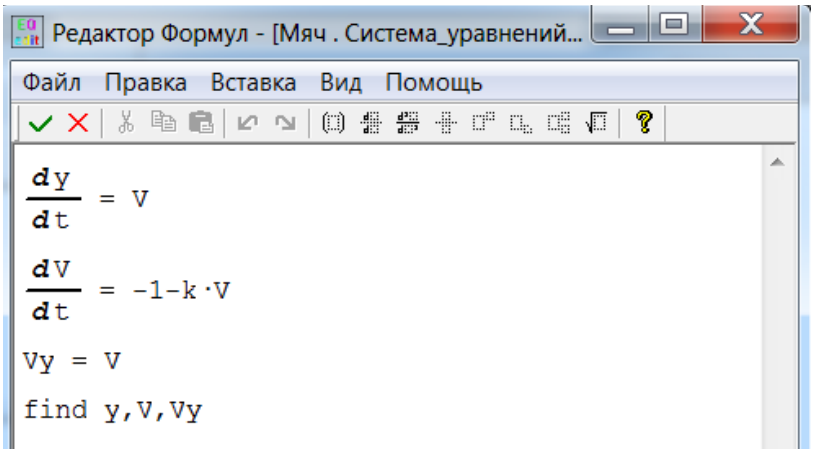

Рис. 3.4.3. Система уравнений, описывающая движение мяча при отскоке от твердой поверхности

При мгновенном абсолютно упругом ударе скорость V_y скачком меняет знак на противоположный. Соответствующая карта поведения показана на рис. 4.

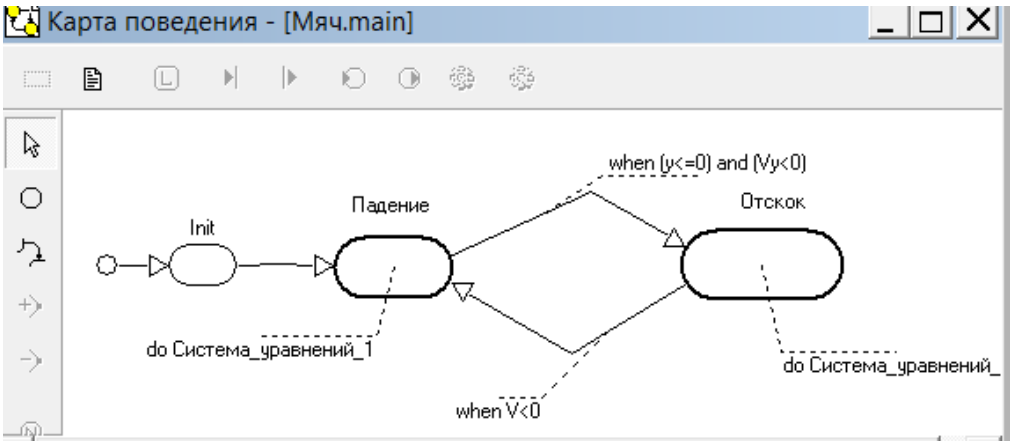

Рис. 3.4.4. Главная карта поведения

Условие отскока мяча ***y <= 0 and Vy < 0***, при этом переменная ***V*** получает значение ***−Vy,*** так как удар о поверхность без потери скорости, т.е. абсолютно упругий. В узле «**Отскок**» вычисления производятся с переменной ***V.*** Если начинается падение мяча ***(V < 0),*** то происходит переход в узел «**Падение**», при этом начальное значение ***Vy = V.*** Решается та же система уравнений с новыми начальными условиями. Карта поведения системы представлена на рис. 3.4.4.

Временная диаграмма процесса показана на рис. 3.4.5.

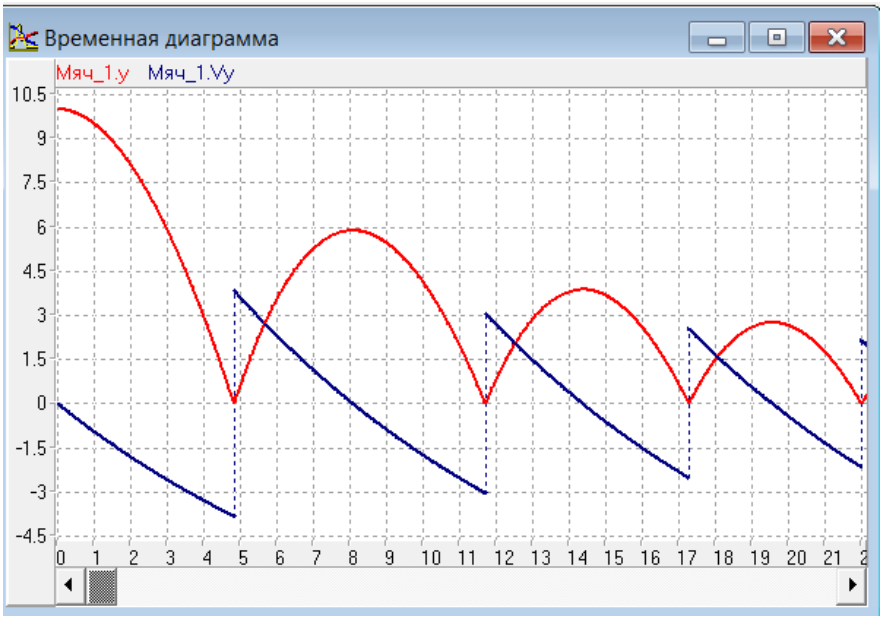

Рис. 3.4.5. Временная диаграмма процесса

Построить средствами MVS-модель прыгающего мячика. Движение мячика отобразить с помощью **временной и фазовой** диаграмм и **3D-анимации**.

3.5. Моделирование одноразрядного двоичного сумматора

В данной работе строится модель логической схемы одноразрядного двоичного сумматора.

Таблица 3.1

Сложение одноразрядных двоичный чисел

A	B	M	S	P
0	0	0	0	0
0	1	0	1	0
1	1	0	0	1
1	0	0	1	0
0	0	1	1	0
0	1	1	0	1
1	1	1	1	1
1	0	1	0	1

Результат сложения двух одноразрядных двоичных чисел представлен в таблице 3.1. Здесь: ***A, B*** – одноразрядные двоичные числа (слагаемые), ***M*** – перенос из младшего разряда, ***S*** – сумма, ***P*** – перенос в старший разряд. Логическая схема сумматора представлена на рис. 3.5.1.

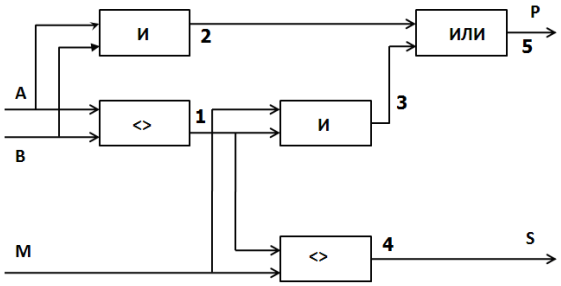

Рис. 3.5.1. Логическая схема одноразрядного двоичного сумматора

Создать модель одноразрядного сумматора средствами MVS. Модель должна допускать ввод исходных значений слагаемых, индикацию результатов и исходных данных, также многократный ввод исходных значений и выполнение вычислений (рис. 3.5.2–3.5.5).

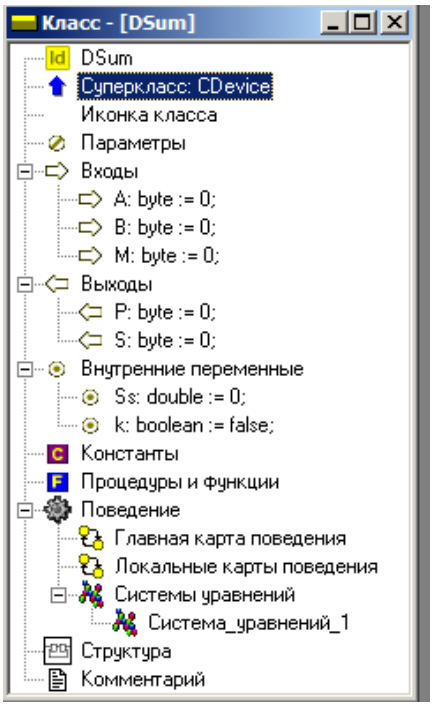

Рис. 3.5.2. Окно класса DSum

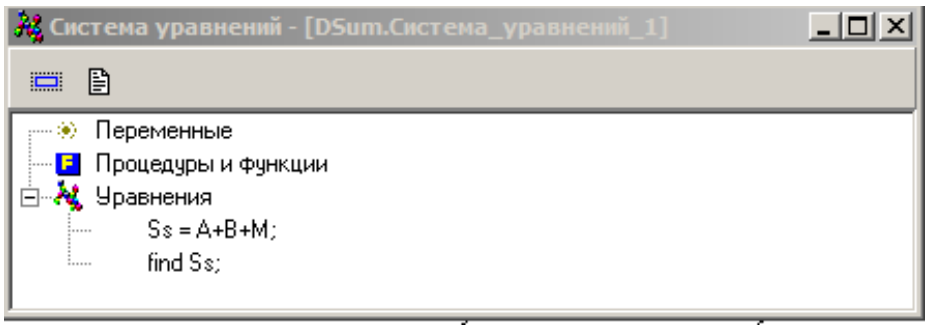

Рис. 3.5.3. Система уравнений проекта

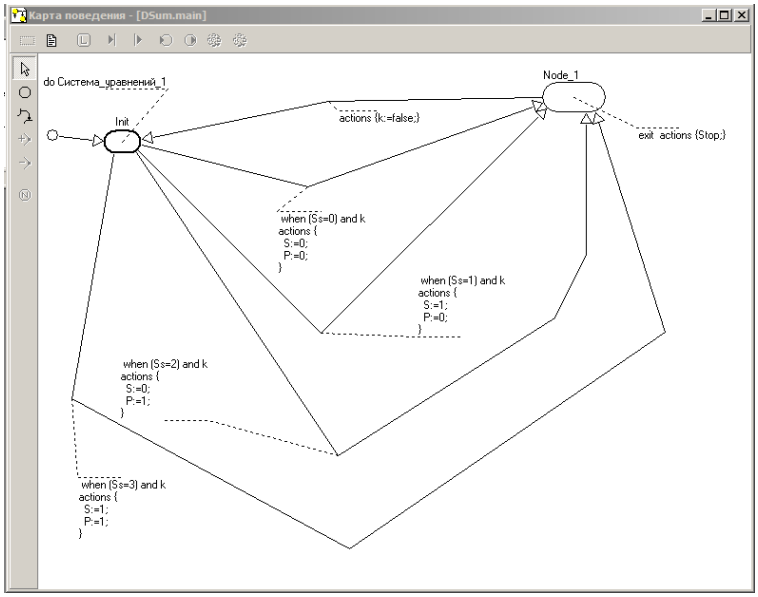

Рис.3.5.4. Карта поведения проекта

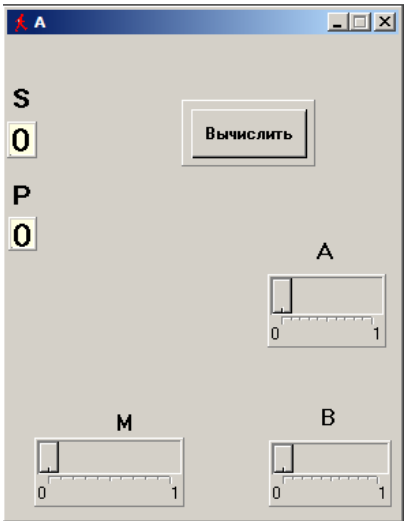

Рис. 3.5.5. Стенд управления проектом

Рис. 3.5.6. Окно стандартных 2D-компонентов

Стенд управления проектом (рис. 3.5.5) собирается в окне «**Новая 2D анимация**» с использованием стандартных 2D-компонент MVS: «**Кнопка**», «**Цифровой индикатор**», «**Ползунок**» (рис. 3.5.6). Переменные проекта путем перетаскивания необходимо связать с соответствующими 2D-компонентами.

Глава 4. MVS-МОДЕЛИ С ЭЛЕМЕНТАМИ УПРАВЛЕНИЯ

4.1. Моделирование системы стабилизации

Системы регулирования предназначены для поддержания определенного состояния объекта. Рассмотрим систему регулирования частоты вращения двигателя постоянного тока (рис. 4.1.1). Задача системы – стабилизация частоты вращения двигателя постоянного тока при его включении или при действии внешней механической нагрузки.

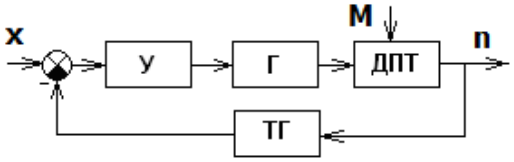

Рис. 4.1.1. Схема системы управления с обратной связью

Здесь **У** – усилитель; **Г** – генератор постоянного тока; **ДПТ** – двигатель постоянного тока; **ТГ** – тахогенератор (устройство, которое генерирует электрическое напряжение, пропорциональное частоте вращения); **M** – момент внешней нагрузки на валу двигателя; n – частота вращения вала двигателя, **x** – внешнее электрическое воздействие (при включении двигателя).

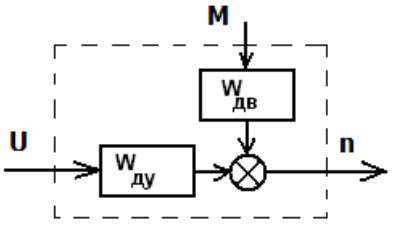

Рис. 4.1.2. Модель двигателя

U – электрическое напряжение, подаваемое на двигатель, **M** – момент внешней нагрузки на валу двигателя.

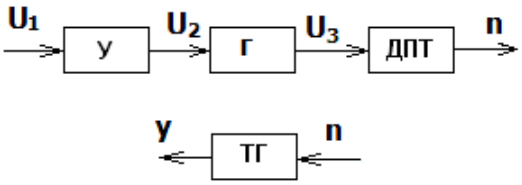

Рис. 4.1.3. Схема взаимодействия элементов системы

Математическая модель усилителя (рис. 4.1.3) представляется следующей зависимостью:

$$U_2 = k_u \cdot U_1$$

.

В свою очередь, математическая модель генератора имеет вид:

$$a_g \frac{dU_3}{dt} + b_g \cdot U_3 = k_g \cdot U_2$$

.

Математическая модель двигателя постоянного тока, описывающая поведение двигателя при изменении питающего напряжения, суть следующее:

$$a_d \cdot \frac{d^2 n}{dt^2} + b_d \cdot \frac{dn}{dt} + c_d \cdot n = k_d \cdot U_3 \quad .$$

В свою очередь математическая модель двигателя, описывающая поведение двигателя при изменении момента механической нагрузки на валу имеет вид:

$$a_d \cdot \frac{d^2 n}{dt^2} + b_d \cdot \frac{dn}{dt} + c_d \cdot n = k_m \cdot (a_m \frac{dM}{dt} + b_m \cdot M) .$$

В итоге, с учетом принципа суперпозиции (рис. 4.1.2), для двигателя будем иметь:

$$a_d \cdot \frac{d^2 n}{dt^2} + b_d \cdot \frac{dn}{dt} + c_d \cdot n = k_d \cdot U_3 + k_m \cdot (a_m \frac{dM}{dt} + b_m \cdot M) .$$

Модель тахогенератора имеет вид: $y = k_t \cdot n$. Тогда $U_1 = x - y$.

В данной работе строится единая для всей системы математическая модель в виде общей системы уравнений, которая включает уравнения для всех элементов системы и соотношения для связей между элементами (рис. 4.1.4). Параметры системы регулирования представлены на рис. 4.1.4.

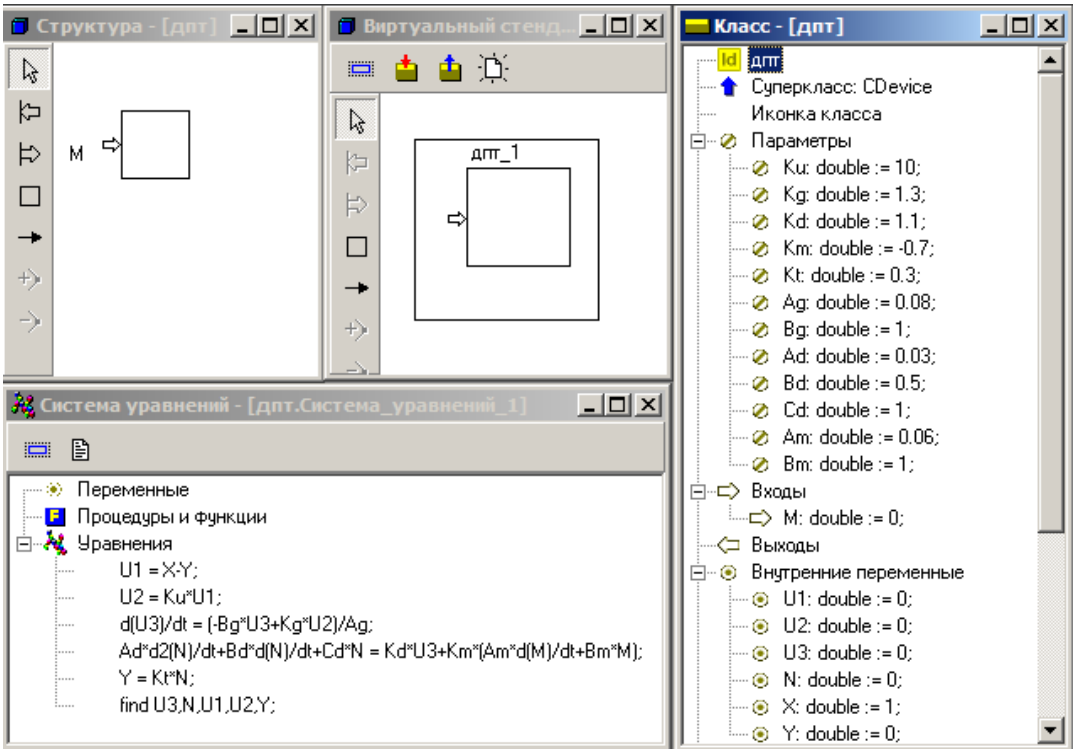

Рис. 4.1.4. MVS-модель системы стабилизации ДПТ

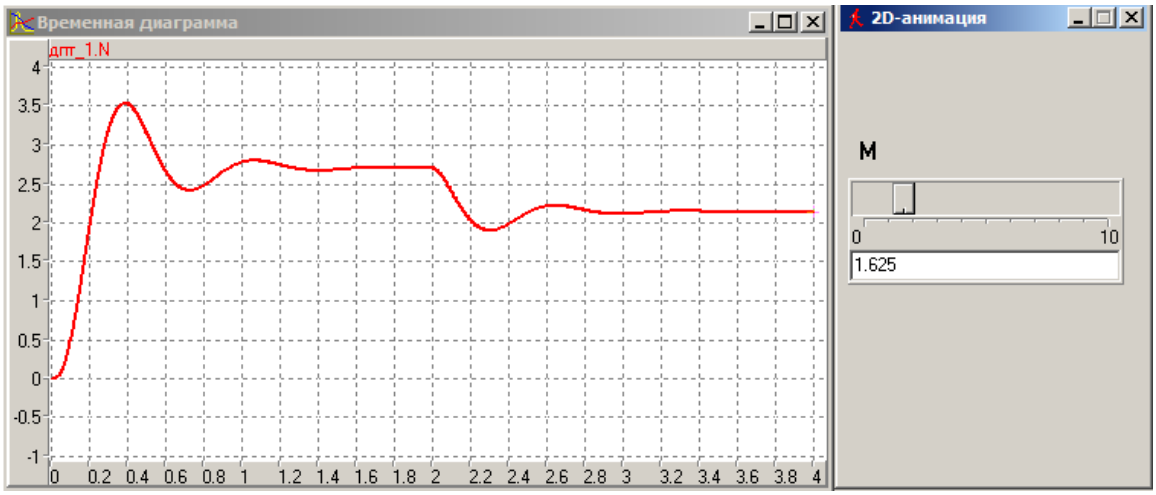

Рис. 4.1.5. Временная диаграмма и элементы управления экспериментом

Построить MVS-модель системы регулирования (рис. 4.1.4-4.1.5). Построить временные диаграммы переходных процессов при включении двигателя. Средствами 2D-анимации создать управление в виде ползунка для придания внешней нагрузки на включенный двигатель.

В меню «**Окна**» виртуальной модели выбрать пункт «**Новая 2D-анимация**». В меню «**Сервис**» выбрать «**Стандартные 2D-компоненты**» (заготовки). Появится окно «**Стандартные 2D-компоненты**» (рис. 4.1.6):

Рис. 4.1.6. Окно 2D-компонентов

Нужно «перетащить» «**Ползунок**» на поле 2D-анимации и связать переменную ***M*** с ползунком методом drag and drop. Средствами контекстного меню для ползунка установить **max** и **min** значения ***M*** и свойство «**Значение**» для показа величины ***M.***

4.2. Модель с анимацией и средствами управления моделью

Построить модель фонарика, состоящую из «лампочки» и «кнопки». При нажатии кнопки «лампочка» должна загораться. Данная модель – статическая, т.е. ее параметры не изменяются во времени. Срабатывание кнопки и зажигание света происходит в модельном времени мгновенно.

Решение задачи состоит в том, что для класса «**Фонарик**» необходимо создать одну переменную булевского типа с именем «**Вкл**», связать ее с кнопкой и с окошком. В исходном состоянии, при отпущенной кнопке, значение переменной «**Вкл**» будет **false** (ложь). При нажатии кнопки значение переменной «**Вкл**» поменяется на **true**. Поскольку переменная «**Вкл**» будет связана и с окошком, то, получив значение **true**, она «зажжет» свет. При отпускании кнопки, значение переменной «**Вкл**» станет **false**, и фонарик «погаснет».

Запустить программу MVS. Создать новый проект (рис. 4.2.1). Дать проекту имя «**Фонарик**» и, нажав кнопку обзор, поместить его, например, в папку C:\Мои модели или в любую другую папку, которая доступна для записи.

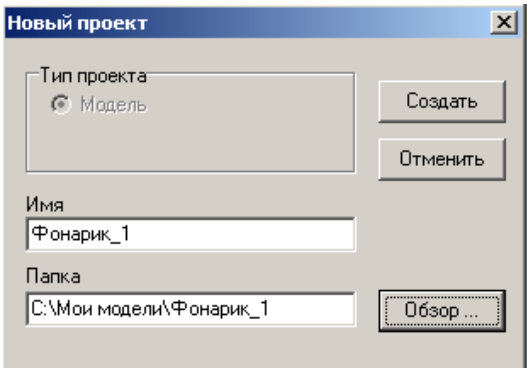

Рис. 4.2.1. Создание нового проекта

Нажать кнопку «**Создать**». После чего в главном окне проекта появятся окна проекта «**Фонарик**» (рис. 4.2.2):

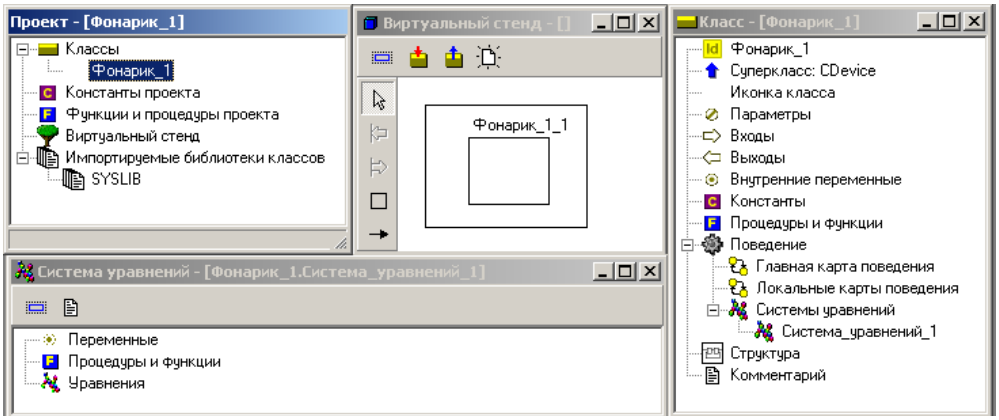

Рис. 4.2.2. Окна проекта

В окне **класса** «**Фонарик**» следует добавить **внутреннюю переменную** булевского типа с именем «**Вкл**». Для добавления переменной в окне **класса** «**Фонарик**» следует щелкнуть правой кнопкой мыши по пункту «**Внутренние переменные**» и в контекстном меню выбрать пункт: «**Добавить**». В появившемся окне задать имя переменной «**Вкл**», нажать кнопку «**Тип**» и в окне «**Выберите тип**» выбрать **boolean** (рис. 4.2.3):

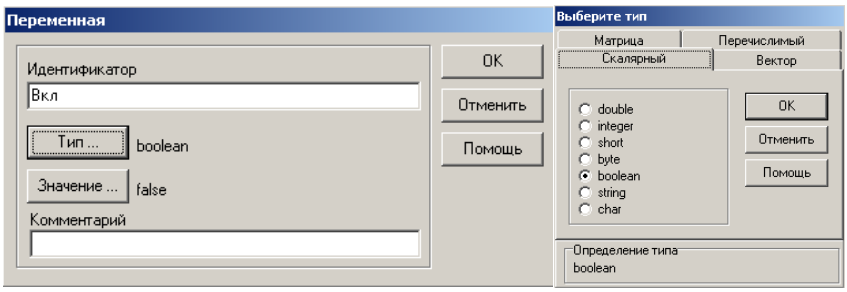

Рис. 4.2.3. Задание имени переменной «Вкл и ее типа boolean с исходным значением false

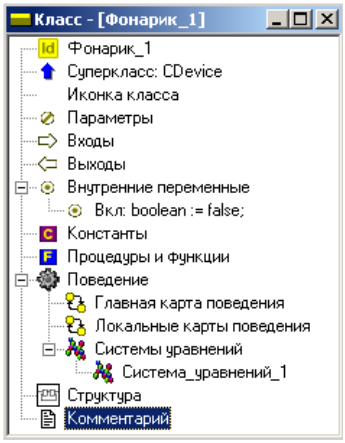

Рис. 4.2.4. Состояние класса «Фонарик»

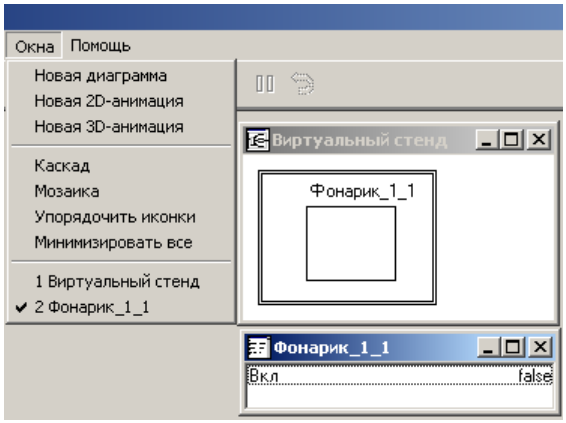

Рис. 4.2.5. Окна «Виртуального стенда» и переменных «Испытательного стенда»

После нажатия **ОК** в окне выбора типа, затем **ОК** в окне переменной в окне класса «**Фонарик**» появится переменная «**Вкл**» типа **boolean**, со значением **false** (рис. 4.2.4).

Нажмите кнопку «**Запустить модель**» на панели инструментов главного окна проекта. Вы увидите окно «**Виртуального стенда**» и **окно переменных**. В **окне переменных** указана только что введенная переменная «**Вкл**» и ее исходное значение **false** (рис. 4.2.5). Создадим окно для 2D-анимации: «**Окна**» – «**Новая 2D-анимация**» (рис. 4.2.6):

В меню «**Окна**» выберем пункт «**Новая 2D-анимация**». В меню «**Сервис**» выберем «**Стандартные 2D-компоненты**». Появится окно «**Стандартные 2D-компоненты**»:

Рис. 4.2.6. Окно стандартных 2D-компонентов

Для создания модели лампочки фонарика используем «**Линейный индикатор сплошной**», а для выключателя – «**Кнопку**». Нужно «перетащить» «**Линейный индикатор сплошной**» и «**Кнопку**» на поле 2D-анимации (рис. 4.2.7).

Щелчок правой кнопкой над 2D-компонентом и выбор пункта меню «**Надпись**», позволяет подписать элементы в окне 2D-анимации (рис. 4.2.8).

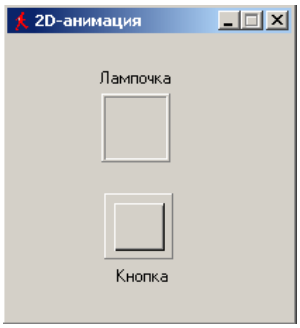

Рис. 4.2.7. Размещение сплошного линейного индикатора и кнопки на поле 2D-анимации

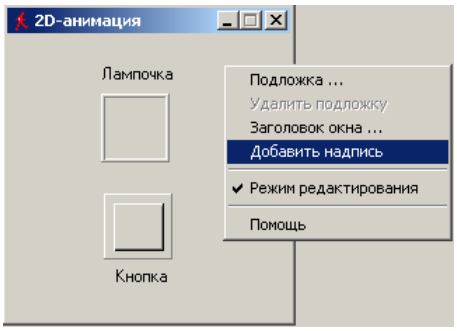

Рис. 4.2.8. Создание надписей

Свяжем переменную «**Вкл**» с индикатором и кнопкой. Для этого поставим курсор на строку с именем переменной «**Вкл**» в окне переменных «**Фонарик**» (рис. 4.2.5) и перетащим переменную на кнопку в окне 2D-анимации (рис. 4.2.7). И еще раз перетащим переменную «**Вкл**» на индикатор.

Щелкнем по кнопке, индикатор изменит цвет. Сделаем свет фонарика ярче: нужно правой кнопкой мыши щелкнуть на индикаторе, выбрать в меню «**Цвет**», в появившемся окне выбрать **красный цвет** и нажать «**ОК**». Теперь при нажатии кнопки «**фонарик**» светит красным светом.

Обратите внимание на то, что происходит со значением переменной «**Вкл**» в окне переменных «**Фонарик**» при щелчке по кнопке. Переменная «**Вкл**» меняет свое значение с **false** на **true.** Поскольку эта переменная связана с индикатором, то и он изменяет состояние.

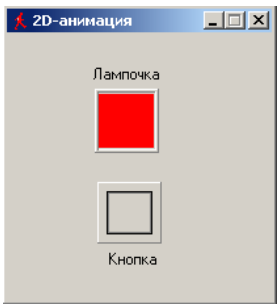

Рис. 4.2.9. Зажженный «фонарик»

В созданной модели фонарик светит только при нажатии кнопки (рис. 4.2.9). Исправим это. Щелкнем по кнопке правой клавишей мыши и в меню выберем пункт «**Фиксация**». Теперь, при щелчке по кнопке фонарика левой клавишей мыши, она остается утопленной, и фонарик светит постоянно. Чтобы его выключить, нужно щелкнуть по кнопке еще раз.

Остается выполнить сохранение проекта. Закрыть главное окно модели. На вопрос: «**Сохранить текущие установки?**» – ответить «**Да**» и закрыть главное окно проекта.

Новая модель (продолжение созданного проекта) будет, как и ранее, состоять из кнопки и «лампочки». При нажатии кнопки на лампочку должно подаваться напряжение от батарейки (рис. 4.2.10). Модель должна учитывать то обстоятельство, что напряжение, которое обеспечивает батарейка, со временем снижается, и лампочка светит слабее. В этом и состоит динамика процесса: напряжение и связанная с ним яркость свечения лампочки зависят от времени.

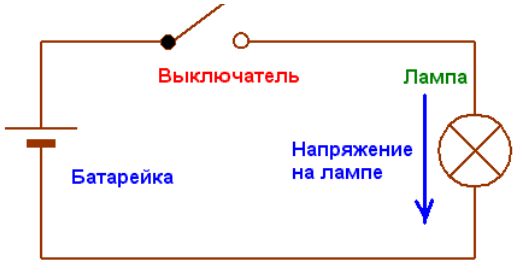

Рис. 4.2.10. Электрическая схема моделируемого объекта

Изменение напряжения батарейки со временем описывается уравнением: $U(t) = U_0 \exp(-t/Tmax) - (1)$.

$U(t)$ – напряжения батарейки – функция времени; U_0 – начальное напряжение батарейки; t – время работы батарейки в фонарике во включенном состоянии; $Tmax$ – срок службы батарейки. Это характерное время, в течение которого батарейка разрядится почти на две трети. У многих батареек для фонариков время $Tmax$ составляет несколько часов. Таким образом, модель фонарика должна учитывать время работы батарейки и изменение ее напряжения в соответствии с формулой изменения напряжения.

Используем проект «**Фонарик**» и продолжим создание модели, потому что там уже выполнена значительная часть работы (рис. 4.2.11).

Переменная «**Вкл**» уже определена в проекте, добавим переменную ***U*** в окне «**Класс**». Необходимо щелкнуть по пункту «Внутренние переменные», выбрать в выпавшем меню «**Добавить**» и в появившемся окне ввести имя переменной – ***U***. Оставить тип **double** и значение **0**. Нажать «**ОК**».

Таким же образом добавим параметр ***Tmax*** – срок службы батарейки, тип **double**, начальное значение 5 часов. Для этого нужно щелкнуть по пункту «**Параметры**», выбрать «**Добавить**» и в появившемся окне ввести имя параметра – ***Tmax*** и его значение. Нажать «**ОК**».

Параметр – это величина, которую, в отличие от переменной, нельзя менять в процессе работы модели, но можно изменить до запуска модели. Аналогично добавим переменную U_0 – начальное напряжение батарейки. $U_0 = 1$. После добавлений окно класса примет вид: (рис. 4.2.12). Напоминаем: MVS различает строчные и прописные буквы в именах переменных.

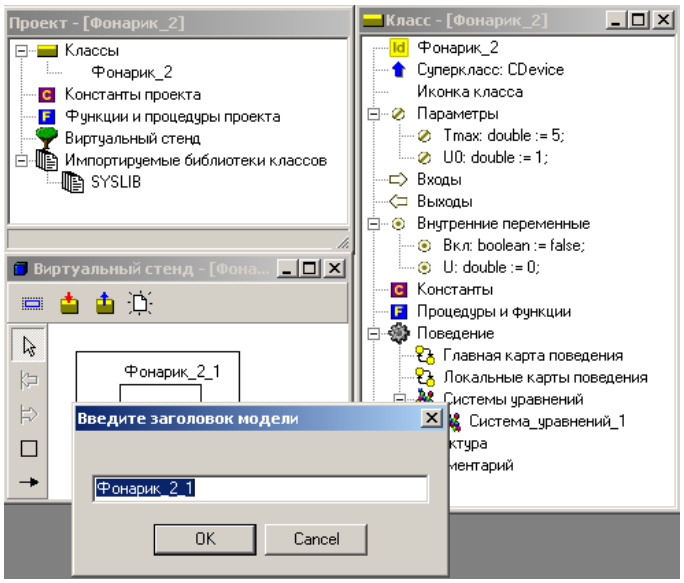

Рис. 4.2.11. Переименование модели

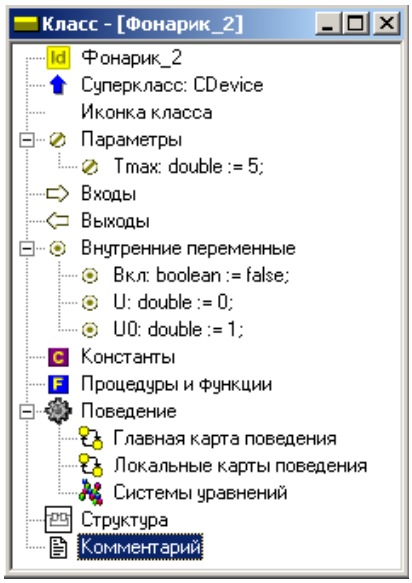

Рис. 4.2.12. Добавлены переменные *U*, U_0 и параметр *Tmax*

Введем уравнение, описывающее изменение напряжения батарейки с течением времени. Для этого нужно открыть окно «**Система уравнений**» и щелкнуть правой кнопкой по пункту «**Уравнения**» и в контекстном меню выбрать «**Изменить**» (рис. 4.2.13).

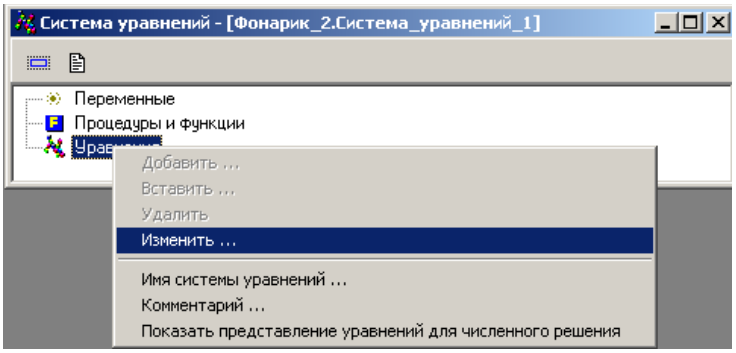

Рис. 4.2.13. Открытие редактора формул для изменения системы уравнений

В появившемся окне «**Редактор формул**» ввести формулу:

$$U = U_0 exp(-Time/Tmax).$$

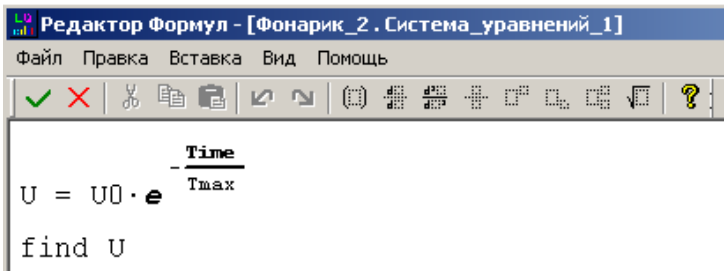

Рис. 4.2.14. Запись уравнения в редакторе формул

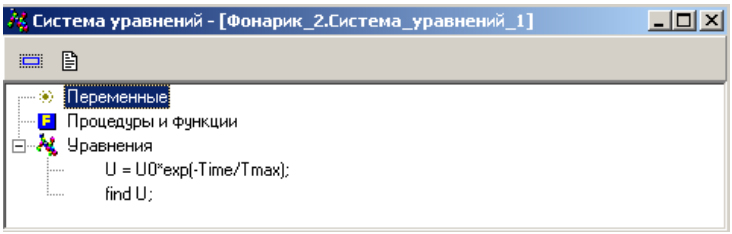

Рис. 4.2.15. Уравнение разряда батарейки

В этой формуле переменная: ***Time*** – время работы модели с момента ее запуска, внутренняя переменная MVS. Редактор формул представит уравнение в следующем виде (рис. 4.2.14–4.2.15)

После чего щелкнуть по кнопке «**Сохранить и выйти**». Окно системы уравнений примет вид см. рис. 4.2.15.

Для создания модели, как и ранее, необходимо нажать на кнопку – «**Запустить модель**» в главном окне проекта. В результате появится окно «**Испытательного стенда**» (рис. 4.2.16).

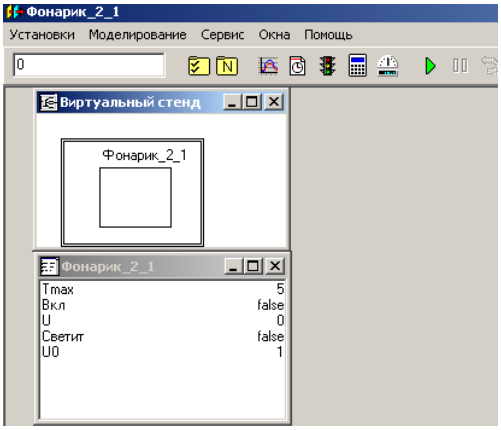

Рис. 4.2.16. Окно испытательного стенда

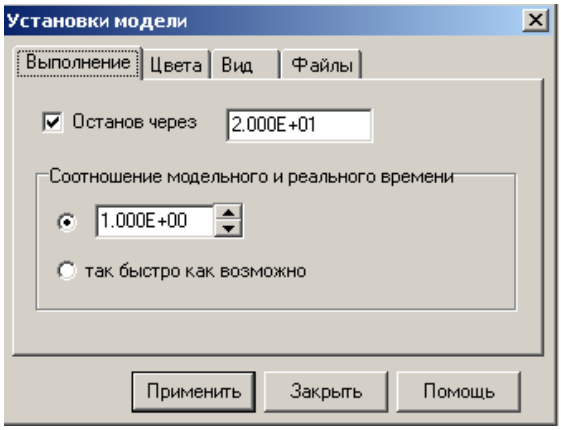

Рис. 4.2.17. Установка времени моделирования

Поскольку окно 2D-анимации уже было построено в предыдущем проекте, то остается лишь связать переменные с его элементами.

В появившемся окне «**Испытательного стенда**» следует перетащить переменную ***U*** из окна переменных «**Фонарик**» на непрерывный индикатор в окне 2D анимации. Если задержать курсор на индикаторе, то всплывет надпись, сообщающая о том, что индикатору присвоена переменная ***U***, а также о том, что индикатор отображает величины в пределах от 0 до 1.

Проделайте для контроля то же самое с кнопкой фонарика. Должно появиться сообщение, что кнопка связана с переменной «**Вкл**». Если такое сообщение не появится, то свяжите кнопку с переменной «**Вкл**», перетащив ее из окна переменных на кнопку. Внешний вид индикатора изменим – растянем его по горизонтали.

С помощью кнопки «**Новая диаграмма**» добавим в модель временную диаграмму и перетащим на нее из окна переменных ***U***.

Теперь запустим модель в работу (кнопка) и проверим ее в динамике. Но предварительно в меню «**Установки**» «**Испытательного стенда**» выберем пункт «**Модель**», в появившемся окне установим: «**Останов через 20 сек**. Результат моделирования представлен на рис. 4.2.18.

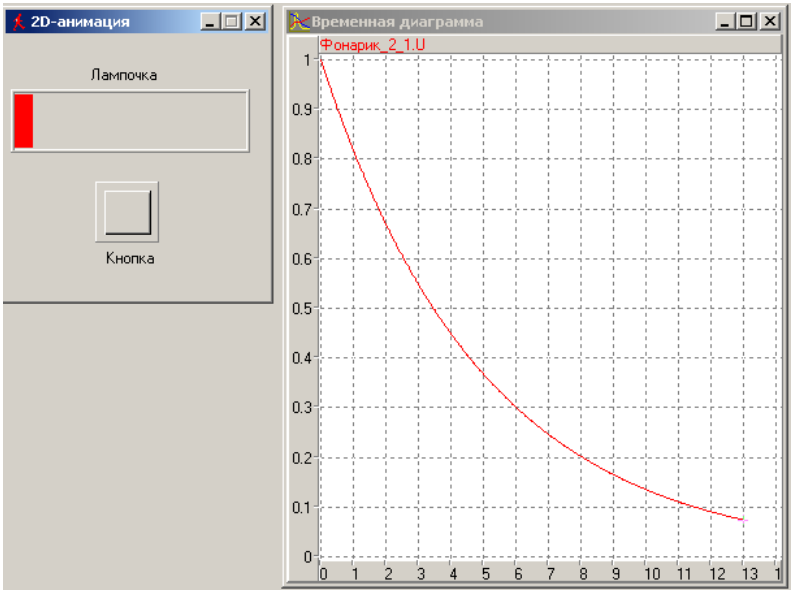

Рис. 4.2.18. Результат первого запуска модели

Света в окошке лампочки с течением времени будет становиться все меньше и значение переменной ***U***, как видно на диаграмме, будет уменьшаться. Однако модель **не реагирует на нажатие кнопки**, напряжение изменяется непрерывно с момента запуска модели до окончания времени моделирования.

Однако объект моделирования должен проявлять гибридное поведение. В процессе функционирования должны быть два качественно различных состояния с непрерывным поведением. Это состояния: «**Лампочка горит**» и «**Фонарик выключен**». Переход из одного состояния в другое происходит по **событию**: включение кнопки фонарика. **Событие** в программировании означает, что в процессе выполнения программы что-то произошло, что будет замечено и воспринято компьютером, и он должен на это как-то отреагировать. Событием может быть щелчок мышкой, или то, что одна величина станет равной другой и т.п.

Если событие влияет на состояние объекта, то до того, как событие случилось, модель MVS функционировала в соответствии с **одними уравнениями**, то **после** события она должна подчиняться **другим уравнениям**. Например, **до** включения кнопки напряжение на лампочке равно нулю, то **после** «события» – **включение кнопки**, напряжение на лампочке изменяется в соответствие с формулой. Созданный проект событий не учитывает, поэтому лампочка горит непрерывно.

Для модернизации MVS-модели введем дополнительную переменную с именем «**Светит**», тип – **boolean**, начальное значение – **false**. Для этого необходимо окно «**Класс-[Фонарик]**» (рис. 4.2.11). Процедура добавления переменной была описана выше.

Модель, управляемая событиями, в MVS может быть наглядно представлена графически. В системе моделирования MVS для этого служит «**Карта поведения**». Она создается в специальном окне, где можно графически изобразить и то, как модель мгновенно реагирует на события, и то, как она себя ведет в промежутках между событиями. Программа MVS позволяет пользователю строить модель и программировать ее поведение, используя **две формы** описания модели **в карте поведения**.

Первая форма применяется в **действиях переходов**, **во входных и выходных действиях узлов**. Здесь пользователь может записывать в виде операторов условия срабатывания переходов, мгновенные действия, которые должна будет выполнить модель.

Фактическое время, затрачиваемое программой на выполнение операторов, не учитывается во времени функционирования модели. Для модели эти действия осуществляются мгновенно.

Вторая форма описания поведения модели используется исключительно для **описания непрерывного поведения** в узле, здесь описывается непрерывное поведение объекта во времени. Удобство этой формы заключается в том, что пользователь записывает уравнения, характеризующие поведение модели, в виде, близком к форме записи обычных математических уравнений. Заботиться о составлении алгоритма решения уравнений и записи программы на алгоритмическом языке нет необходимости, MVS это сделает самостоятельно. В совокупности, обе формы дают гибкий и удобный в использовании аппарат описания поведения моделей различной степени сложности.

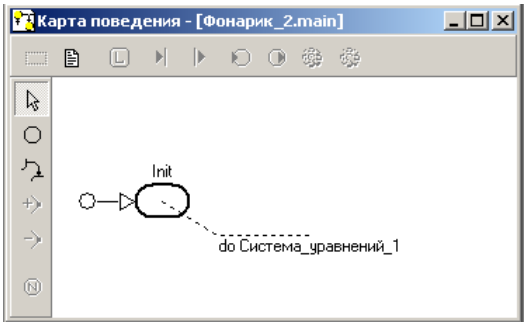

Рис. 4.2.19. Исходный вид главной карты поведения

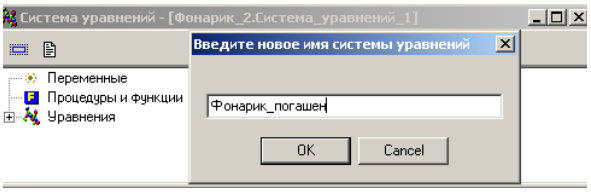

Рис. 4.2.20. Окно исходной системы уравнений узла Init

Теперь следует приступить к построению «**Главной карты поведения**». Других карт поведения в этом проекте не будет. В окне «**Класс**» следует дважды щелкнуть по пункту «**Главная карта поведения**» (рис. 4.2.19). Появится новое окно «**Карта поведения**».

Узел «**Init**» в проекте имеется всегда, и ему может быть приписано непрерывное во времени поведение модели в соответствии с уравнениями «**Система_уравнений_1**». Содержимое системы уравнений вводится в специальном редакторе формул. По смыслу действий для узла «**Init**» в системе уравнений может быть только формула ***U = 0***. Название системы уравнений может быть изменено соответственно поведению модели в узле.

В рассматриваемом проекте в исходном состоянии фонарик должен быть выключен, т.е. его кнопка должна быть отпущена, а лампочка не должна гореть. Другими словами, узел «**Init**» карты поведения должен соответствовать выключенному состоянию фонарика. Изменения состояния фонарика в узле «**Init**» с течением времени не будет. Для повышения наглядности карты поведения изменим название системы уравнений узла «**Init**». Щелкнем дважды в окне «**Класс**» по пункту «**Система_уравнений_1**», которая, как видно на рис. 4.2.21, приписана узлу «**Init**». Появится и станет активным **окно «Система уравнений»:**

Щелкнем по кнопке «**Имя системы уравнений**» –  , в появившемся окне введем «**Фонарик_погашен**» и щелкнем по кнопке «**ОК**». Изменим положение надписи «**Фонарик_погашен**» в окне карты поведения. Она примет такой вид (рис. 4.2.21).

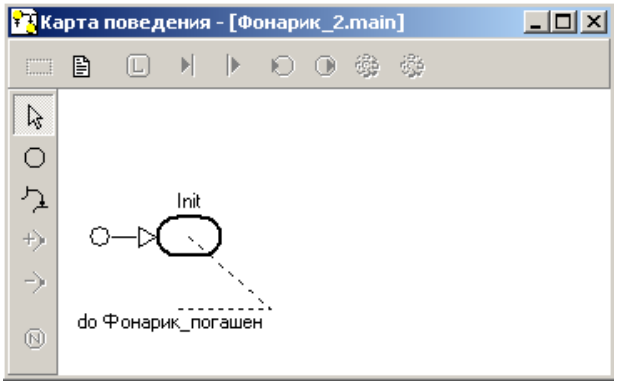

Рис. 4.2.21. Карта поведения с новым названием системы уравнений узла Init

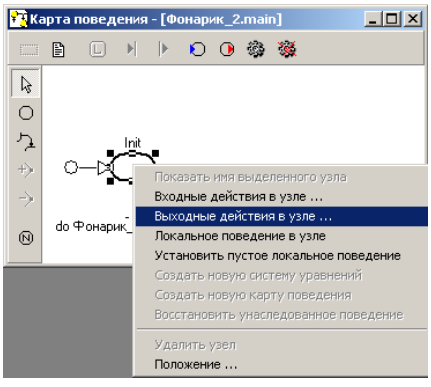

Рис. 4.2.22. Открытие редактора входных действий, которые должны выполнится при входе в узел Init

В момент перехода в узел «**Init**», фонарик переходит в состояние **выключен**. Щелкнем правой клавишей по узлу «**Init**» и в контекстном меню выберем пункт «**Входные действия в узле**» (рис. 4.2.22)**.**

В появившемся окне редактора формул введем оператор **Светит := false;** и щелкнем по кнопке «**Сохранить и выйти**» (рис. 4.2.23).

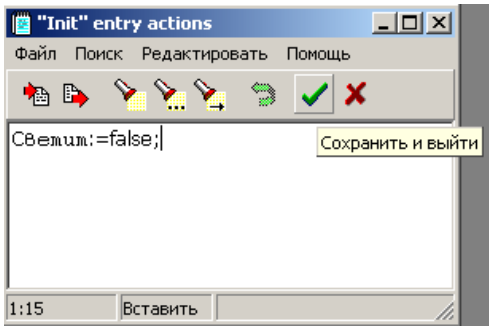

Рис. 4.2.23. Окна редактора формул для входных действий узла Init

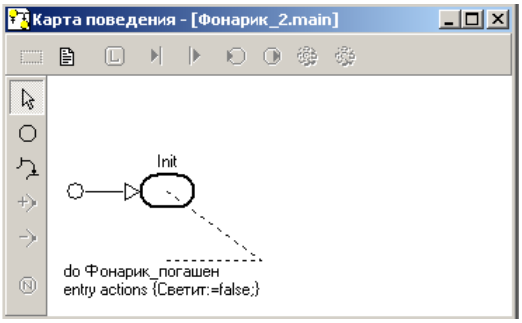

Рис. 4.2.24. Действия при входе (entry actions) в начальный узел Init

Карта поведения примет вид по рис. 4.2.24. Добавим узел, соответствующий включенному состоянию фонарика. Нужно щелкнуть по кнопке «**Создать новый узел**» –  в окне карты поведения (рис. 4.2.26), переместить курсор на поле карты (он примет вид крестика), в правой части карты нажать левую клавишу и, удерживая ее, растянуть вправо вниз овал нового узла.

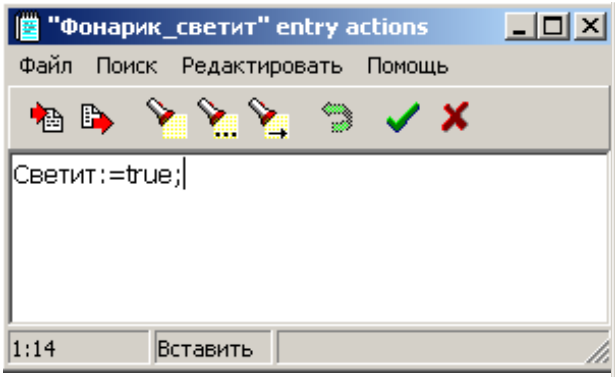

Рис. 4.2.25. Входное действие (entry actions) узла Фонарик_светит

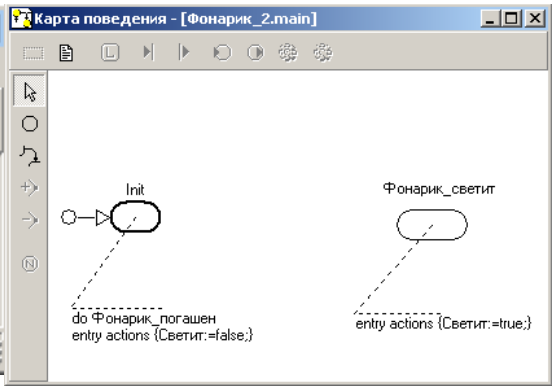

Рис. 4.2.26. Создание узлов на карте поведения. Указаны входные действия

Изменим название узла на содержательное. Для этого необходимо щелкнуть дважды на тексте «**Node_1**» и ввести новое название

«**Фонарик_светит**». Теперь необходимо задать действия, которые будут выполнены при входе в узел «**Фонарик_светит**». Делается это точно так же, как и для узла «**Init**» (рис. 4.2.25). В редакторе формул необходимо записать: **Светит := true;** (рис. 4.2.25). Щелкнуть по кнопке  «**Сохранить и выйти**». Карта поведения примет вид по рис. 4.2.26.

Редактор формул замечает синтаксические ошибки и при попытке сохранить и выйти сообщает об этом. Например, если забыть поставить в конце оператора точку с запятой (**;**), то редактор откажется сохранять такую запись и сообщит об этом. После перемещения элементов на свободные места, карта поведения примет вид – см. рис. 4.2.25.

Переходы от одного вида поведения модели к другому происходят вследствие и сразу после свершения событий, которые могут произойти при работе модели. Переходы на карте поведения представляются линиями, связывающими узлы. В нашем проекте два события. Первое – это нажатие на кнопку фонарика (фонарик светит). Второе – это отпускание этой кнопки (фонарик выключен). И то и другое события будут происходить при щелчке по кнопке.

Слева на карте поведения щелкнем по кнопке «**Создать новый переход**»  . Курсор примет вид крестика в кружочке. Следует навести его на узел **Init**, нажать левую клавишу мыши и, удерживая ее, переместить курсор на узел «**Фонарик_светит**». Получится горизонтальная линия перехода с точкой посредине. Ухватите за точку и поднимите ее вверх. Можно добавить и еще одну точку, щелкнув по линии перехода правой клавишей и выбрав из выпавшего меню «**Добавить опорную точку**». Курсор станет крестиком, подвести его в нужном месте к линии и, когда возникнет точка на курсоре, щелкнуть левой клавишей.

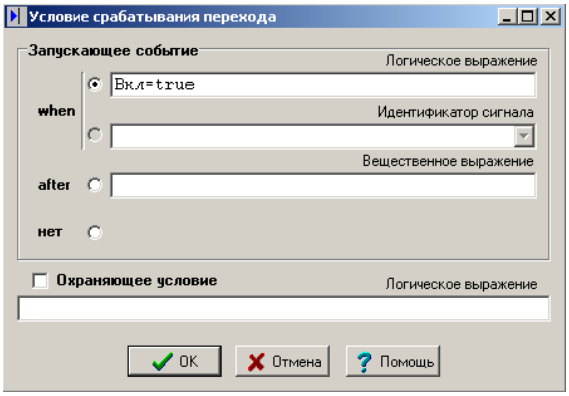

Рис. 4.2.27. Задание условия срабатывания перехода

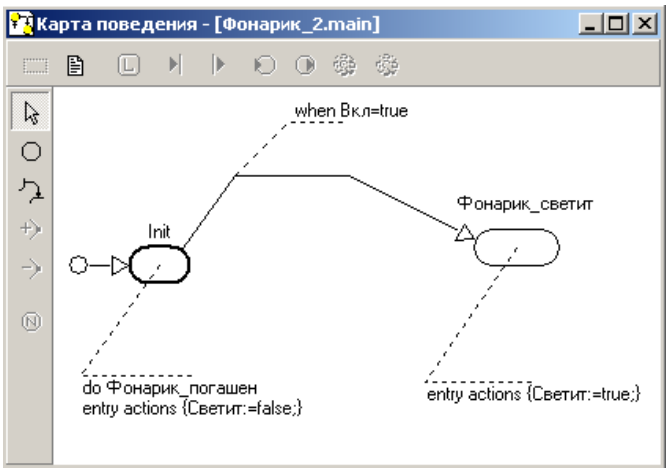

Рис. 4.2.28. Переход из узла Init в узел Фонарик_светит

Теперь следует поставить **условие срабатывания перехода**. Переход определяется **событием** – нажатием кнопки фонарика. Поэтому переход должен происходить в момент, когда переменная «**Вкл**» примет значение **true** (рис. 4.2.27). Щелкнем правой клавишей по переходу и выберем в меню пункт «**Условие срабатывания перехода**». В появившемся диалоговом окне выберем **when** (когда) и введем условие **Вкл = true** (рис. 4.2.27).

Переход от одного вида поведения фонарика, соответствующего узлу «**Init**», к другому произойдет в тот момент, когда переменная «**Вкл**» примет значение **true**.

Нажать «**ОК**». В окне карты поведения рядом с переходом появится сноска **when Вкл = true.** Эту надпись можно переместить, схватив левой кнопкой мыши. В результате карта поведения примет вид – см. рис. 4.2.28.

В момент срабатывания перехода, при входе в узел «**Фонарик_светит**» переменная «**Светит**» примет значение **true,** и лампочка, которую мы свяжем позднее с этой переменной, загорится.

Щелкнем по кнопке «**Создать новый переход**» в карте поведения (рис. 4.2.28). Соединим узел «**Фонарик_светит**» с узлом «**Init**» (именно в таком направлении). Щелкнем правой клавишей по линии вновь созданного перехода и в выпавшем меню выберем «**Условие срабатывания перехода**». В появившемся окне выберем запускающее событие **when** и введем логическое выражение **Вкл = false** (рис. 4.2.29).

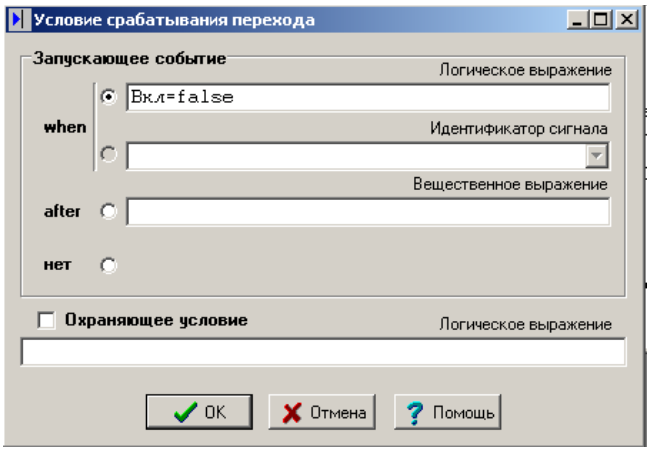

Рис. 4.2.29. Ввод условия срабатывания при переходе от Фонарик_погашен к Init

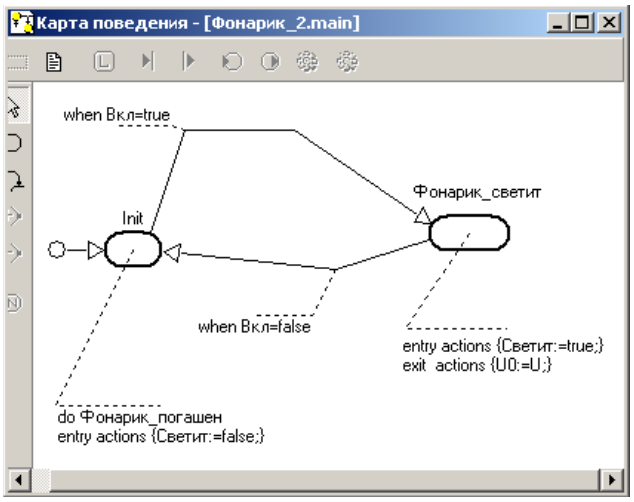

Рис. 4.2.30. Карта поведения

Напряжение батарейки уменьшается только во время, когда фонарик светит. Первая модель, которую мы построили, не предусматривала отключение фонарика. Для того чтобы при новом включении изменение напряжения начиналось с величины, которое оно имело при предыдущем отключении, добавим действие на выходе из узла «**Фонарик_светит**». Выполняется это так же, как и добавление входных действий: выделим узел «**Фонарик_светит**» и щелкнем правой кнопкой, затем выберем пункт «**Выходные действия в узле**». В редакторе формул необходимо записать:

U0 := U; результат представлен на рис. 4.2.30. Щелкнем по кнопке «**ОК**».

Теперь необходимо создать систему уравнений, которая описывает непрерывное поведение объекта в узле «**Фонарик_светит**». Выделим узел «**Фонарик_светит**» и после щечка правой кнопкой мыши выберем пункт «**Создать новую систему уравнений**». Новой системе уравнений зададим имя: **Фонарик_светит**. Запишем в систему формулу изменения напряжения во времени. При записи формулы использована переменная MVS **LocalTime**, которая имеет начало отсчета в момент времени, когда узел «**Фонарик_светит**» становится активным (рис. 4.2.31). Аналогичным образом наполним формулами систему уравнений «**Фонарик_погашен**», в которой переменной *U* задается нулевое значение (рис. 4.2.32).

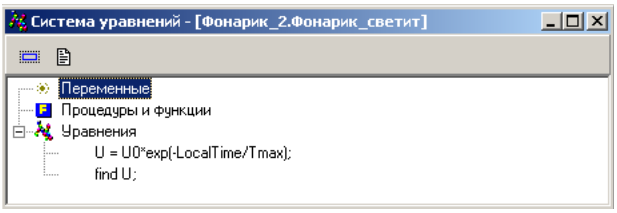

Рис. 4.2.31. Измененная система уравнений

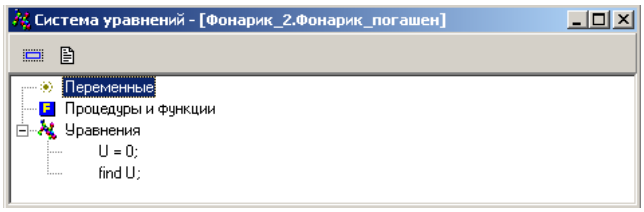

Рис. 4.2.32. Система уравнений «Фонарик_погашен»

Теперь карта поведения примет окончательный вид (рис. 4.2.33).

Добавим еще один линейный индикатор, который будет отражать свечение лампочки. Теперь свяжем переменную ***U*** с линейным индикатором «**Напряжение**», переменную «**Вкл**» с кнопкой, переменную «**Светит**» с линейным индикатором «**Лампочка**», и проверим работу модели (рис. 4.2.34). Действительно, теперь лампочка реагирует на нажатие кнопки, а напряжение уменьшается только во время свечения лампочки.

В качестве переменных модели выберем переменную «**Вкл**» типа boolean, которая будет соответствовать состоянию кнопки выключателя, и переменную ***U*** типа **double**, которая будет определять напряже-

ние, подаваемое на лампочку в момент, когда «**Вкл**» становится **true**. В момент времени, когда кнопка отжимается, «**Вкл**» должна становиться **false** и напряжение от лампочки отключается, т.е. переменная ***U*** должна принимать значение нуль. Переменная ***U*** при запуске модели должна изменяться во времени в соответствии с формулой (1), если переменная «**Вкл**» имеет значение **true**.

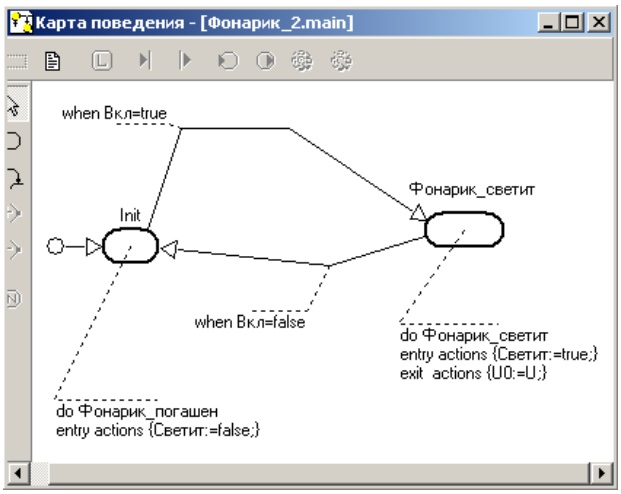

Рис. 4.2.33. Готовая карта поведения

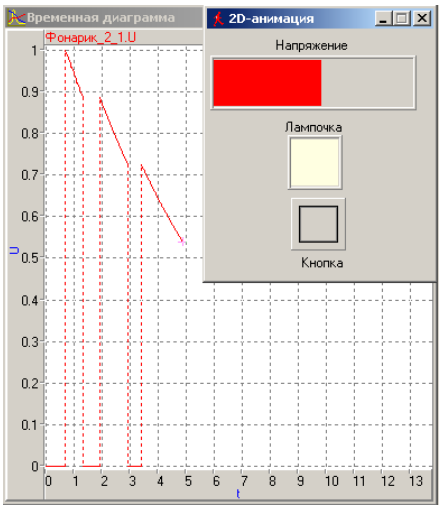

Рис. 4.2.34. Результат моделирования

Запустить модель кнопкой  . В меню появившегося главного окна модели – «**Испытательного стенда**», выбрать пункт «**Новая 3D-анимация**» (рис. 4.2.35):

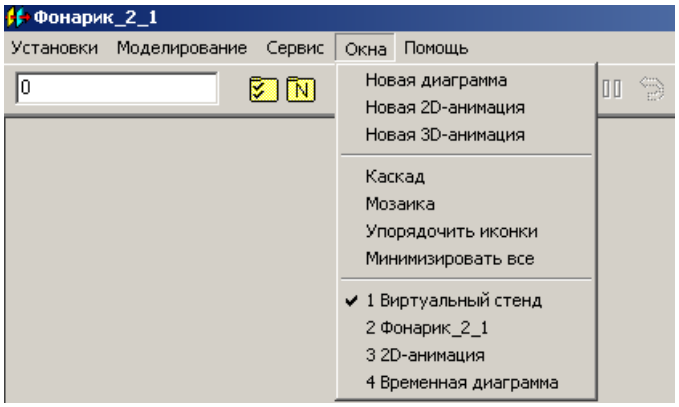

Рис. 4.2.35. Окно новой 3D-анимации

3D-анимация позволяет видеть на экране монитора движущиеся предметы, размеры и форма которых может также меняться в соответствии с изменением переменных модели. В этой работе мы создадим такую модель.

Особенность 3D-анимации, созданной в MVS, состоит в том, что движущиеся предметы здесь воспроизводятся не просто мультипликацией. 3D-анимация, созданная в MVS, изображает движущиеся и изменяющиеся предметы строго в соответствии с точными уравнениями и управляющими событиями. Это позволяет проводить физические эксперименты с моделью, в которые может вмешиваться экспериментатор.

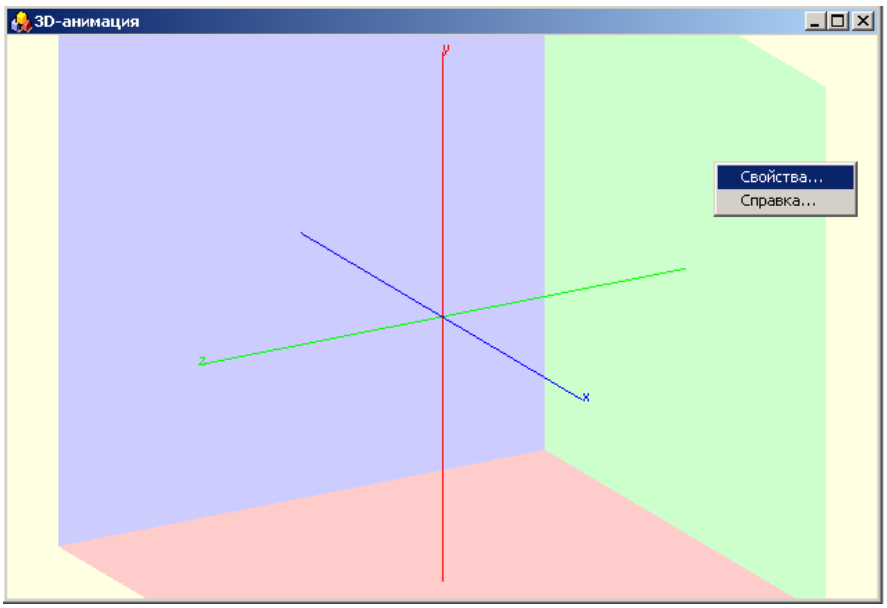

Рис. 4.2.36. Начальное состояние окна 3D-анимации

Щелкнуть на окне «**3D-анимация**» правой кнопкой и в появившемся меню выбрать пункт «**Свойства**» (рис. 4.2.35). Появится окно «**Свойства 3D-анимации**». На вкладке «**Общие установки**» ввести «**Заголовок Фонарик светит**» (рис. 4.2.37).

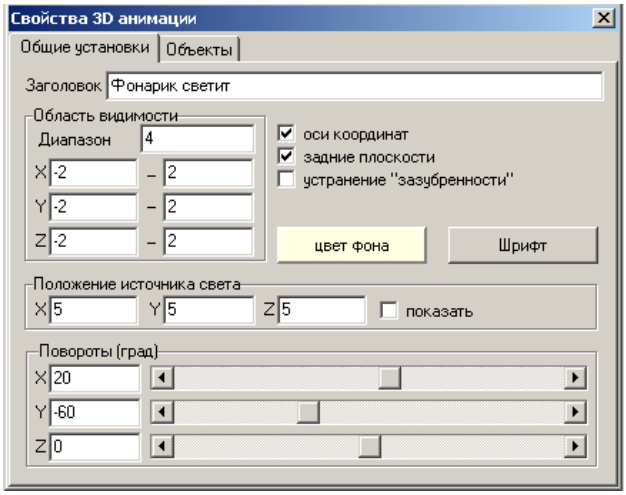

Рис. 4.2.37. Задание параметров 3D-анимации

Анимация будет составлена из простейших фигур: цилиндра и конуса, которые будут изображать корпус фонарика, и конуса желтого цвета, для изображения света фонарика. В окне «**Свойства 3D-анимации**» (рис. 4.2.37) перейти на вкладку «**Объекты**». Щелкнуть по кнопке «**Добавить цилиндр**» и ввести его свойства в таблице, появившейся в правой части окна свойств (рис. 4.2.38):

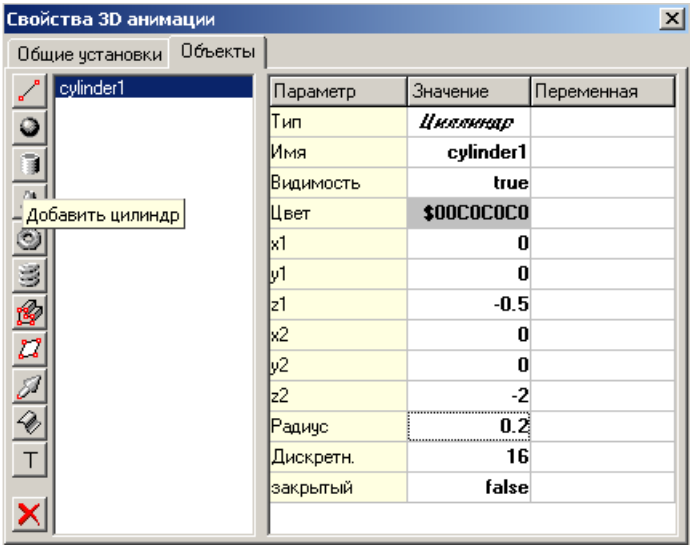

Рис. 4.2.38. Добавление цилиндра – корпуса фонарика

Щелкнем по кнопке «**Добавить конус**» на вкладке «**Объекты**» в окне «**Свойства 3D-анимации**» задать его свойства (рис. 4.2.39).

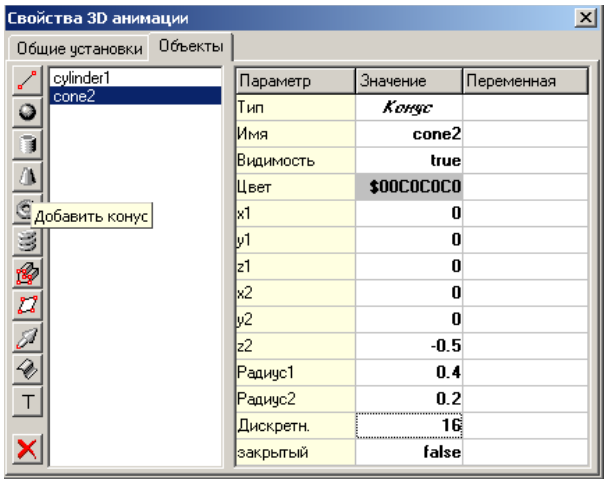

Рис. 4.2.39. Свойства отражателя фонарика

Добавим конус, который будет изображать свет. Щелкнем по кнопке «**Добавить конус**» и зададим его свойства (рис. 4.2.40).

Проведем связывание переменных со свойствами примитивов в окне 3D- анимации. В 3D-анимации изменяться со временем будет высота и раствор конуса света. Поэтому переменные проекта «**Фонарик**» будут связаны только со свойствами второго конуса «**cone3**». Перетащить и бросить переменную «**Вкл**» в строку «**Видимость**» раздела «**Переменная**» свойств объекта **cone3**, а переменную ***U*** в строки ***z2*** и **Радиус2**.

Закрыть окно «**Свойства 3D-анимации**». Создание 3D-анимации завершено. Можно приступать к тестированию модели.

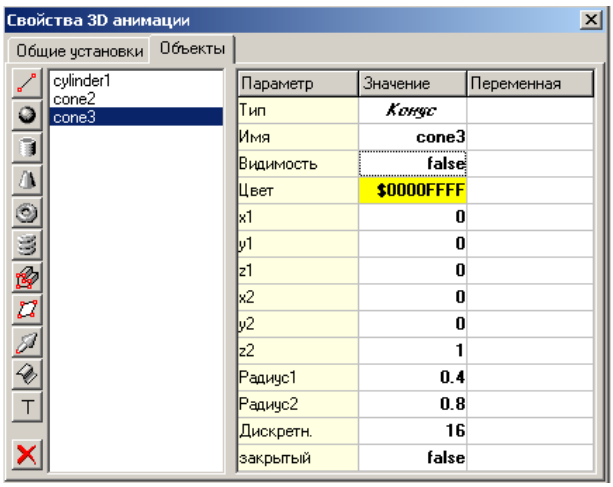

Рис. 4.2.40. Задание свойств конуса света фонарика

Если все сделано правильно, то при запуске модели и включении фонарика, в окне «**3D-анимации**» появляется сноп желтого света, а при выключении фонарика свет исчезает. Фонарик можно поворачивать и рассматривать с разных сторон, схватив левой клавишей мыши и сдвигая ее. Для продолжения работы модели следует нажать кнопку «**Пуск**», а для возобновления работы заново следует нажать кнопку «**Рестарт**» на панели инструментов главного окна модели.

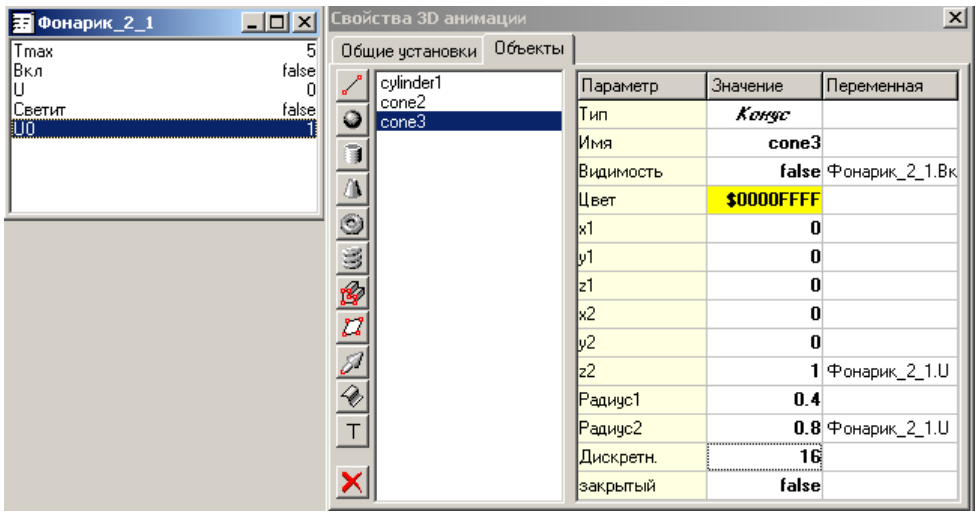

Рис. 4.2.41. Связывание переменных модели со свойствами объекта cone3

Результат моделирования представлен на рис. 4.2.42.

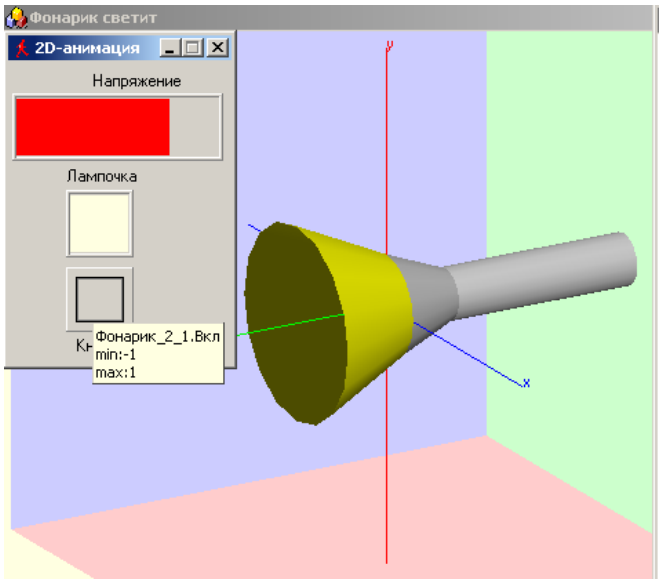

Рис. 4.2.42. Результаты моделирования с 3D-анимацией

Глава 5. МОДЕЛИРОВАНИЕ ЭЛЕКТРИЧЕСКИХ ЦЕПЕЙ

5.1. Моделирование действия переменного напряжения

Методику построения моделей электрических цепей рассмотрим на примере моделирования переходных процессов в электрическом контуре. Задача заключается в нахождении тока протекающего в электрической цепи, и напряжения на конденсаторе C, индуктивности L и сопротивлении R под действием переменного напряжения. Схемы цепей представлены на рис. 5.1.1–5.1.3. Начальные условия полагаем нулевыми (ток в цепи отсутствует, и конденсатор не заряжен).

Модели элементов данной системы (цепи) являются моделями типа «черный ящик». Эти модели связывают входное воздействие (напряжение) и реакцию элемента (электрический ток). Процессы в элементах не рассматриваются.

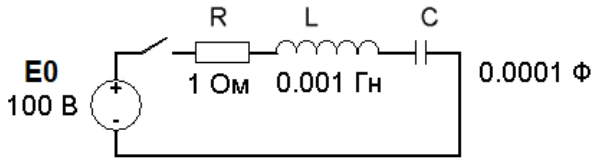

Рис. 5.1.1. Расчетная электрическая схема

Математические модели элементов цепи (закон Ома):

$$u_R = R \cdot i, \;\; u_L = L \cdot \frac{di}{dt}, \;\; C \cdot \frac{du_C}{dt} = i \;\; .$$

Связь между элементами системы описывается соотношением (закон Кирхгоффа):

$$E = u_R + u_L + u_C \; .$$

Математическая модель электрической цепи имеет следующий вид:

$$\frac{di}{dt} = (E - u_R - u_C)/L; \quad \frac{du_C}{dt} = i/C; \quad u_R = R \cdot i.$$
$$i(t=0) = 0, \quad u_C(t=0) = 0.$$

Здесь ***i*** – ток в цепи, ***uc*** – напряжение на конденсаторе, u_L – напряжение на индуктивности, u_R – напряжение на активном сопротивлении, ***R*** – величина активного сопротивления, ***L*** – индуктивность катушки, ***C*** – емкость конденсатора.

Средствами MVS построить модель RLC-цепи (рис. 5.1.1). Средствами MVS построить модель электрических цепей (рис. 5.2.2–5.1.3). Построить временную диаграмму переходного процесса.

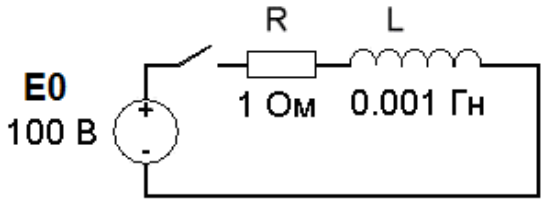

Рис. 5.1.2. RL-цепь

Для данной цепи математическая модель имеет вид:

$$u_R = R \cdot i, \quad u_L = L \cdot \frac{di}{dt}, \quad E = u_R + u_L.$$

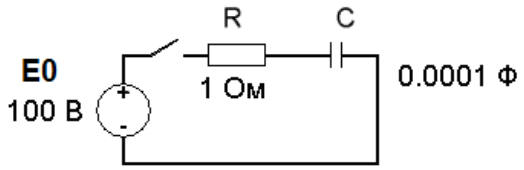

Рис. 5.1.3. RC-цепь

Соответственно, для RC-цепи математическая модель имеет вид:

$$u_R = R \cdot i, \quad C \cdot \frac{du_C}{dt} = i, \quad E = u_R + u_C.$$

Напряжение является переменной величиной $E=E_0 \cos(wt)$ для всех цепей по рис. 5.1.1–5.1.3. Построить фазовые диаграммы для тока и напряжения на элементах цепей ***R, C, L***.

5.2. Моделирование включения и выключения электрической цепи

Имеется схема электрической цепи (рис. 5.2.1), состоящая из конденсатора (емкости), индуктивности, активных сопротивлений и лампочки. При замыкании выключателя образуется замкнутая электрическая цепь, которая, определенным образом реагирует на напряжение батареи. По цепи течет электрический ток.

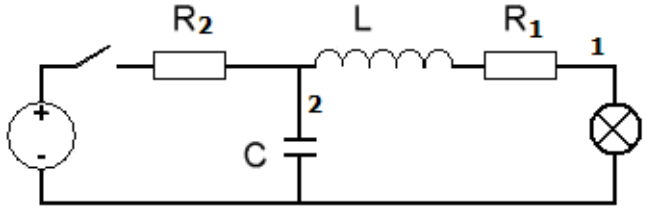

Рис. 5.2.1. Электрическая цепь

Математические модели элементов цепи (законы Ома) имеют вид:

$$u_R = R \cdot i, \;\; u_L = L \cdot \frac{di}{dt}, \;\; C \cdot \frac{du_C}{dt} = i \;\; .$$

Связь между элементами в замкнутой цепи описывается соотношениями (законы Кирхгоффа):

$$E = u_{R1} + u_{R2} + u_L; \;\; E = u_{R2} + u_C; \;\; i = i_1 + i_2.$$

В разомкнутой цепи связь между элементами суть следующее:

$$u_C = u_{R1} + u_L; \;\; i_1 + i_2 = 0.$$

Математическая модель электрической цепи при первоначальном замыкании выключателя имеет вид:

$$u_{R1} = R \cdot i_1, \;\; u_{R2} = R \cdot i \;\; , \;\; \frac{di_1}{dt} = \frac{u_L}{L}, \;\; i_2 = C \cdot \frac{du_C}{dt},$$
$$i = i_1 + i_2, \;\; u_L = E - u_{R1} - u_{R2}, \;\; u_C = E - u_{R2}.$$

$$i_1(t=0) = 0; \;\; i_2(t=0) = 0; \;\; u_C(t=0) = 0; \;\; u_L(t=0) = 0; \;\; u_{R1}(t=0) = 0 \;\; u_{R2}(t=0) = 0.$$

Здесь $i_{1,2}$ – ток в соответствующей ветви цепи (рис. 5.2.1), u_C – напряжение на конденсаторе, u_L – напряжение на индуктивности, u_{R1}, u_{R2} – напряжения на активных сопротивлениях, R_1, R_2 – величина сопротивлений, L – индуктивность катушки, C – емкость конденсатора.

При размыкании цепи ее математическую модель составляют зависимости:

$$u_{R1} = R \cdot i_1, \quad \frac{di_1}{dt} = \frac{u_L}{L}, \quad \frac{du_C}{dt} = \frac{i_2}{C}, \quad i_2 = -i_1, \quad u_L = u_C - u_{R1}.$$

Начальное состояние, в этом случае, соответствует состоянию цепи в момент размыкания выключателя. При дальнейшем замыкании или размыкании выключателя начальное состояние цепи определяется ее предыдущим состоянием.

Построить MVS-модель электрической цепи с выключателем и с индикацией свечения лампочки. Изменение напряжений в цепи отобразить временной диаграммой.

Работа выполняется в среде MVS. Создать проект и поместить его в доступную для записи папку, например, D:\WORK (рис. 5.2.2).

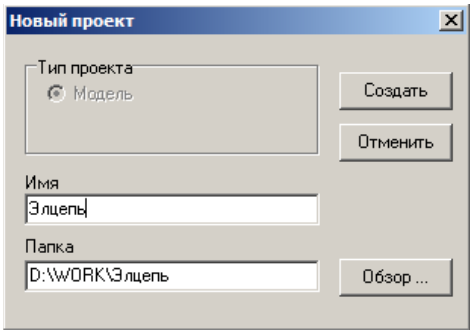

Рис. 5.2.2. Создание нового проекта

В главном окне появятся окна проекта «**Элцепь**» (рис. 5.2.3):

В окне **класса** «**Элцепь**» добавить **внутренние переменные** и **параметры** (рис. 5.2.4). Для добавления переменной в окне **класса** следует щелкнуть правой кнопкой мыши по пункту «**Внутренние переменные**» и в контекстном меню выбрать пункт: «**Добавить**». Аналогичным образом добавляются параметры.

Параметр, в отличие от переменной, не меняется в процессе работы модели, но его можно изменить до запуска модели. Напоминаем: MVS **различает** строчные и прописные буквы в именах переменных

и не «признает» пробелов. Переменная **ON** играет роль выключателя (ON = true – цепь замкнута; ON = false – цепь разомкнута). Переменная **ON** – булевского типа. При задании переменной **ON**, нажать кнопку «**Тип**» и в окне «**Выберите тип**» выбрать тип «**boolean**» (рис. 5.2.5):

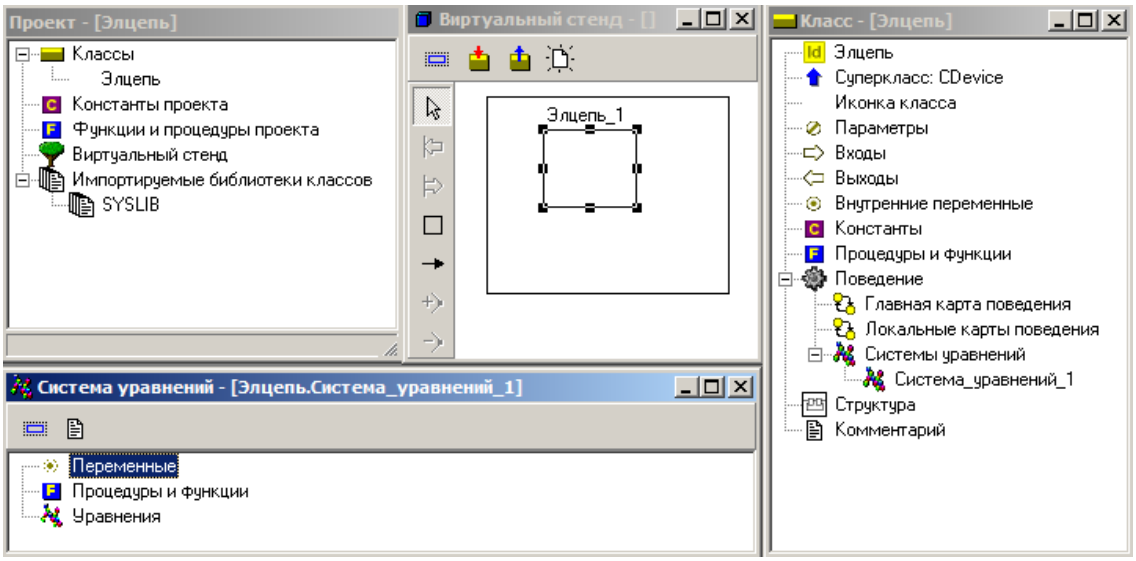

Рис. 5.2.3. Окна проекта «Элцепь»

Создадим математическую модель электрической цепи. Новая модель будет отражать изменение электрических параметров цепи во времени. Модель должна учитывать то обстоятельство, что при выключении цепи поведение системы меняется. Изменение поведения отображается с помощью «**Карты поведения**».

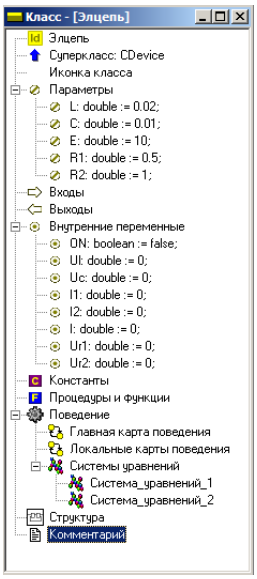

Рис. 5.2.4. Переменные и параметры проекта

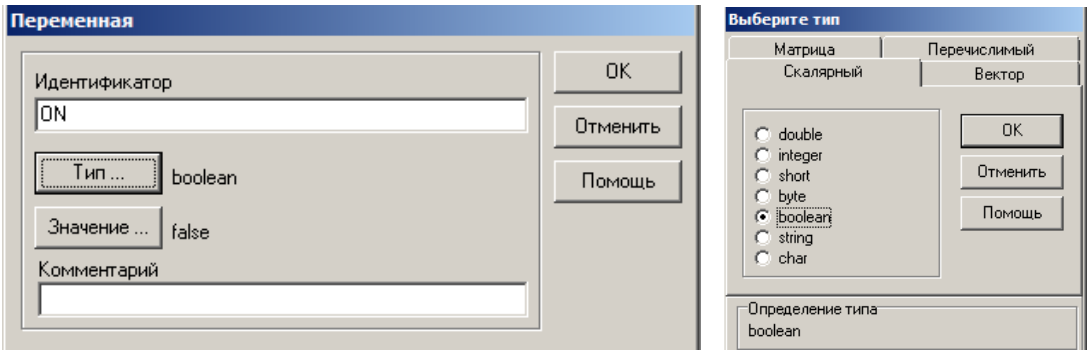

Рис. 5.2.5. Задание переменной ON и ее типа boolean с исходным значением false

Введем уравнения, описывающее изменение переменных проекта с течением времени при замыкании цепи. Для этого нужно открыть окно «**Система уравнений_1**», в окне «**Система уравнений_1**» щелкнуть правой кнопкой по пункту «**Уравнения**» и в контекстном меню выбрать пункт «**Изменить**» (рис. 5.2.6).

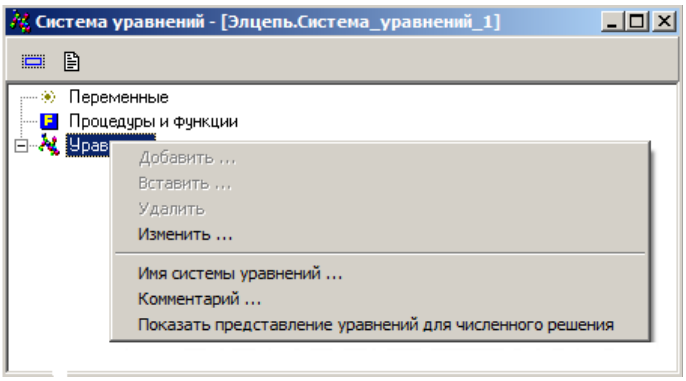

Рис. 5.2.6. Открытие редактора формул для изменения системы уравнений

В окне «**Редактор формул**» ввести уравнения для замкнутой цепи (рис. 5.2.7).

После чего щелкнуть по кнопке – **Сохранить и выйти**. Далее необходимо щелкнуть по пункту «**Системы уравнений**» класса «**Элцепь**» и добавить «**Систему уравнений_2**». Эта система уравнений (рис. 5.2.8) описывает процессы в разомкнутой электрической цепи.

Решение задачи управления электрической цепью состоит в том, что для класса «**Элцепь**» (рис. 5.2.3) необходимо создать карту поведения, которая обеспечит смену поведения модели цепи при ее замыкании и размыкании. Требуется создать «**Главную карту поведения**» (рис. 5.2.3), которая в законченном виде представлена на рис. 5.2.9.

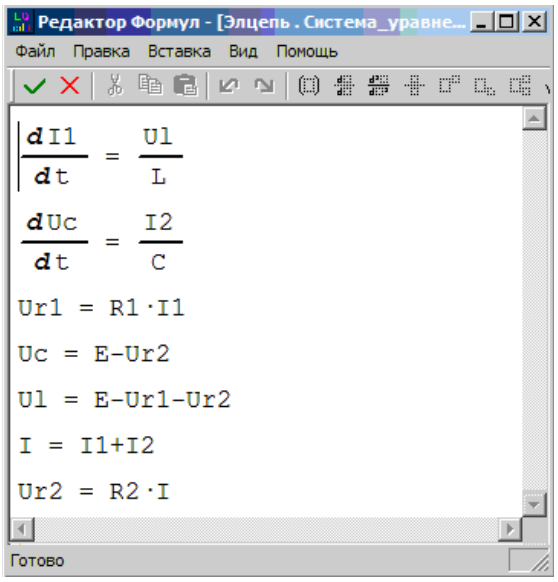

Рис. 5.2.7. Уравнения замкнутой цепи

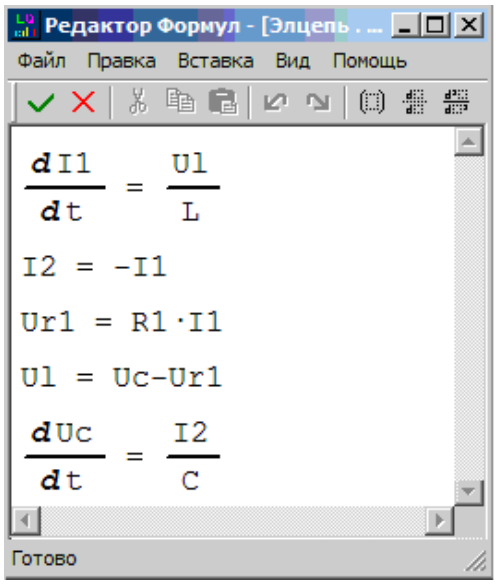

Рис. 5.2.8. Уравнения разомкнутой цепи

Карта поведения состоит из узлов: начальный узел **Init** (создается автоматически), узел **Цепь_разомкнута** и узел **Цепь_замкнута**. Узлы соединены переходами, которые имеют соответствующие условия срабатывания. С узлами необходимо связать (перетаскиванием) **Систему_уравнений_1** и **Систему_уравнений_2**. Начальному узлу **Init** следует задать «**Пустое поведение**». Структура карты поведения создается с помощью кнопок окна редактирования карты поведения (рис. 5.2.9).

Структура карты поведения создается с помощью кнопок окна редактирования карты поведения (табл. 1) или контекстного меню. Задание условий срабатывания переходов показано на рис. 5.2.10.

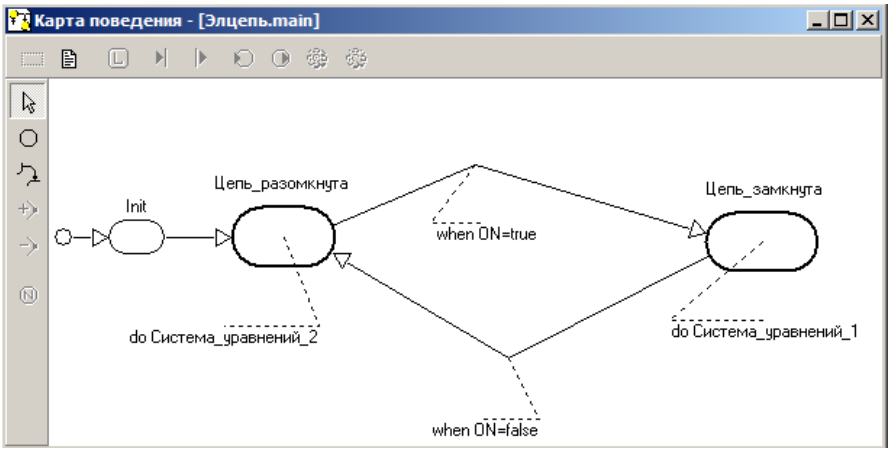

Рис. 5.2.9. Готовая карта поведения

Таблица 5.1

Кнопки редактирования карты поведения

Кнопка	Назначение
	Переход в диалог редактирования пояснительного текста для карты поведения
	Редактирование локальных переменных, функций и процедур карты поведения
	Переход в диалог редактирования условия срабатывания выделенного перехода
	Переход в диалог редактирования последовательности мгновенных действий в выделенном переходе
	Переход в диалог редактирования последовательности входных действий выделенного узла
	Переход в диалог редактирования последовательности выходных действий выделенного узла
	Перехода в окно редактирования поведения, приписанного выделенному узлу
	Приписывание выделенному узлу пустого поведения
	Режим выделения
	Создание нового узла
	Создание нового перехода
	Добавить опорную точку
	Удалить опарную точку
	Показать имя узла

Создадим модель. Для этого нужно, как и ранее, нажать на кнопку  – «**Запустить модель**» в главном окне проекта. В результате появится окно «**Виртуальный стенд**» и окно переменных и параметров (рис. 5.2.11).

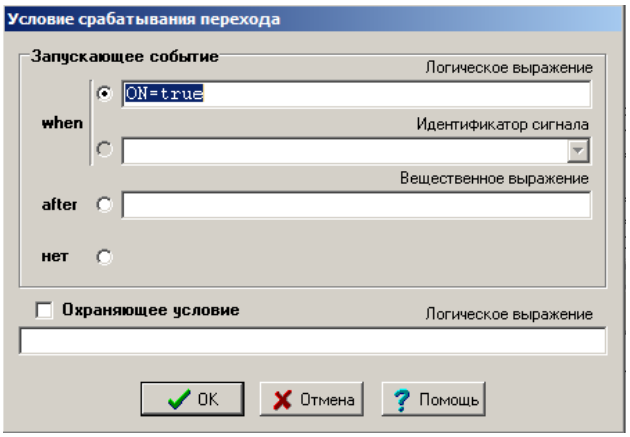

Рис. 5.2.10. Задание условия срабатывания перехода

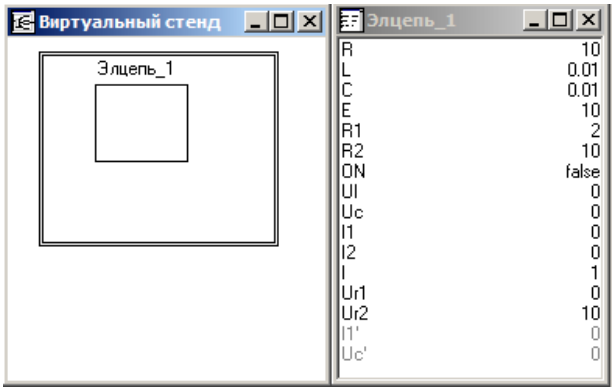

Рис. 5.2.11. Окно виртуального стенда

В окне **переменных** среди прочих указана переменная **ON** и ее исходное значение **false** (рис. 5.2.11). Создадим окно для 2D-анимации. В пункте меню **Окна** выберем подпункт «**Новая 2D-анимация**». В меню «**Сервис**» выберем «**Стандартные 2D-компоненты**». Появится окно (рис. 5.2.12):

Рис. 5.2.12. Окно стандартных 2D-компонентов

Следует поместить в окно **2D-анимация** компонент «**Кнопка**» (рис. 5.2.13). Переменную **ON** необходимо связать с кнопкой путем перетаскивания. Теперь при нажатии кнопки переменная **ON** будет

менять свое значение, и карта поведения будет менять состояние электрической цепи (замыкать и размыкать).

Запустите модель (кнопка ▶). При нажатии кнопки переменная **ON** изменит свое значение, и цепь перейдет в новое состояние (рис. 5.2.14). Аналогичным образом добавьте в окно «**2D-анимация**» линейный индикатор и свяжите его с переменной **ON.** Теперь при нажатии кнопки (замыкание цепи) индикатор будет менять цвет, что соответствует свечению лампочки.

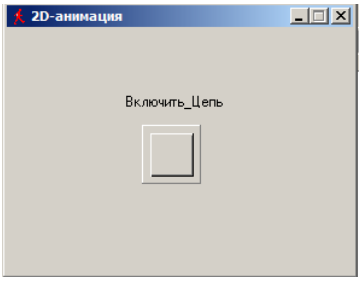

Рис. 5.2.13. Кнопка управления проектом

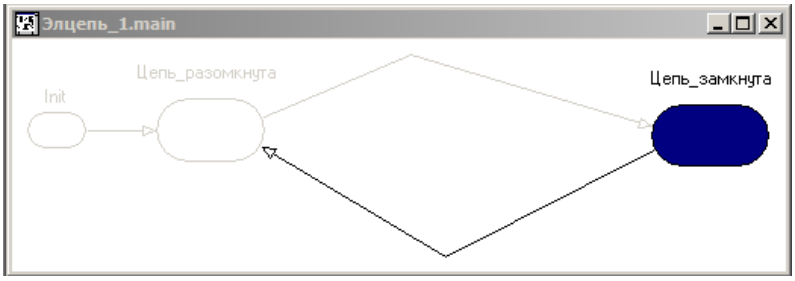

Рис. 5.2.14. Поведение цепи

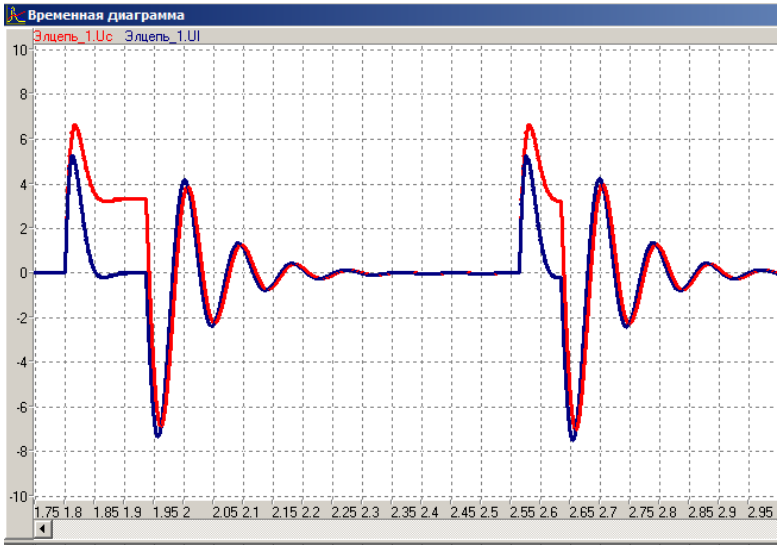

Рис. 5.2.15. Временная диаграмма изменения напряжения

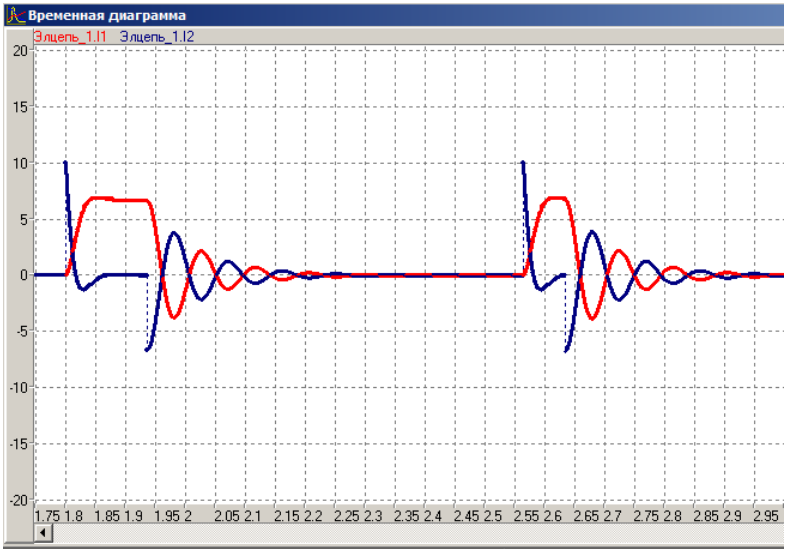

Рис. 5.2.16. Временная диаграмма изменения тока

В заключение создания модели необходимо добавить временные диаграммы, которые будут отражать изменение переменных ***I1, I2 и Ul, Uc*** во времени (рис. 5.2.15–5.2.16).

На основе созданной модели исследуйте свойства электрической цепи в зависимости от ее параметров (R_1, R_2, C, L).

Глава 6. СТОХАСТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

6.1. Моделирование случайных событий

Моделирование случайных процессов – это одно из важнейших направлений современного компьютерного моделирования. Само понятие «случайный» является фундаментальным: Событие называется случайным, если оно достоверно непредсказуемо.

Под событием будем понимать всякий факт, который может наблюдаться в данных условиях. Различают достоверное событие, которое наступает каждый раз при реализации определенного комплекса условий. Невозможное (недостоверное) событие не наступает никогда.

Формирование компьютерных реализаций случайных событий (т.е. моделирование случайных событий) сводится к генерации случайных чисел с равномерным законом распределения вероятности на интервале [0; 1], например, функцией электронных таблиц **СЛЧИС( ).** В MVS имеется аналогичная встроенная функция ***uniform(0,1)***.

Пусть событие *A* наступает с заданной вероятностью ***p***. Пусть имеется возможность генерировать последовательность значений случайной величины с равномерным распределением вероятности на интервале [0; 1]: $x_1, x_2, \ldots x_n$, определим, что событие *A* наступает в том случае, если значение сгенерированной случайной величины удовлетворяет неравенству $x_1 < P$.

Вероятность противоположного события равна $P(\overline{A}) = 1 - P$. Следовательно, если соотношение $x_1 < P$ выполняется, то исходом испытания является событие ***A***, в противном случае исходом испытания является событие $\overline{A}$.

Построить компьютерную модель реализации случайного события с заданной вероятностью и исследовать ее свойства.

Для решения задачи необходимо построить модель по рис. 6.1.3–6.1.5. Исходными данными для модели является вероятность реализации события ***p***. Для генерации случайного события используется функция uniform(0,1). Переменная **Исход** возвращает значение равное **1**, если выполняется неравенство $x_1 < P$ или значение **0** в противном случае. Переменные **Кол_А** и **Кол_НЕ_А** подсчитывают количество исходов событий, которые произошли и не произошли соответственно,

они являются вспомогательными для расчета частоты событий, выражаемых переменными **Частота_А** и **Частота_НЕ_А**.

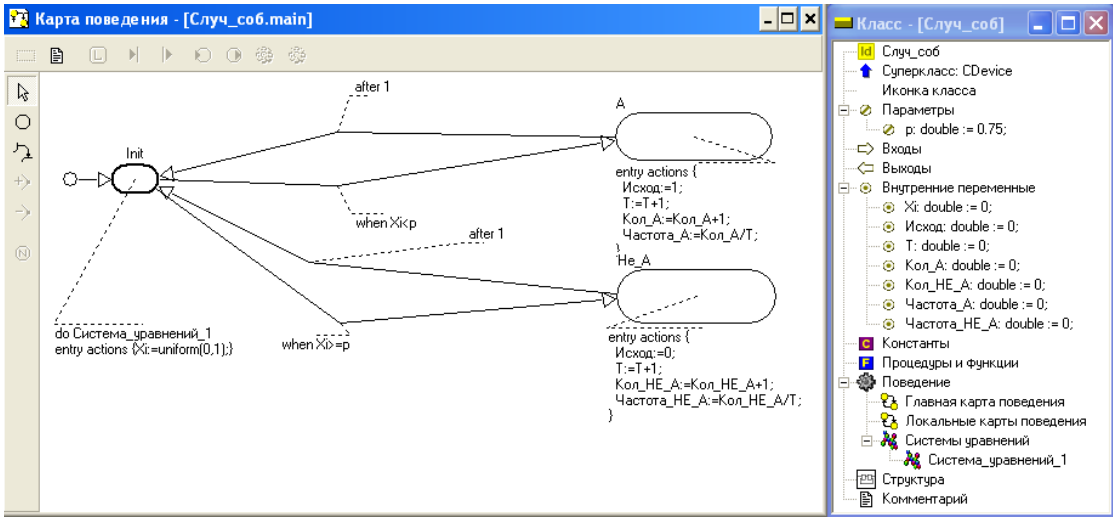

Рис. 6.1.3. Модель случайного события

Постройте временную диаграмму рис. 4

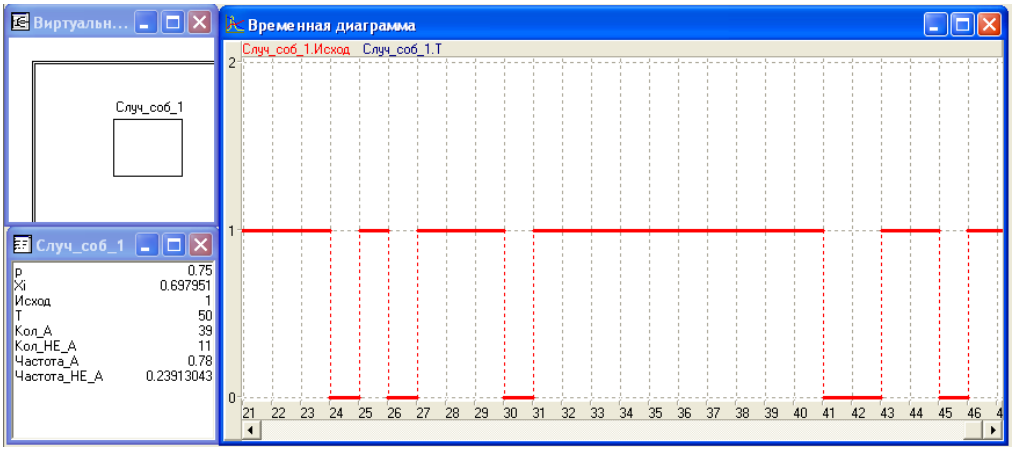

Рис. 6.1.4. Временная диаграмма события

Провести ряд экспериментов при разном количестве реализаций события **А**: 50; 100; 200 и т.д. и сопоставить значение заданной вероятности события ***p*** и частоты его реализации в эксперименте. Проанализируйте поведение переменных.

6.2. Модель полной группы случайных событий

Построить MVS-модель реализации полной группы событий: ***A, B, C*** с вероятностями P_a, P_b, P_c, причем $P_a + P_b + P_c = 1$ (рис. 6.2.1, 6.2.2).

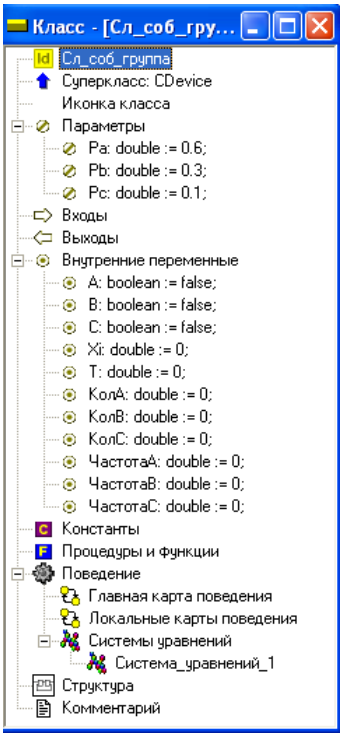

Рис. 6.2.1. Моделирование полной группы случайных событий

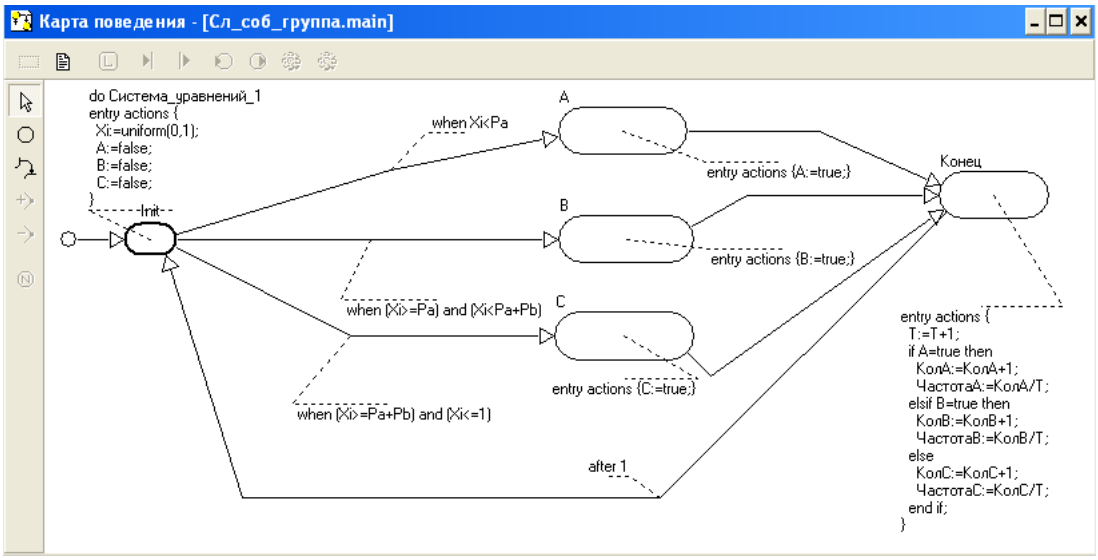

Рис. 6.2.2. Карта поведения при моделировании полной группы из 3-х событий

Для демонстрации результатов моделирования выполните команду **Окна – Новая 2D анимация**, разместите в появившемся окне три цветовых индикатора, свяжите их с переменными, соответствующими исходам событий. Используя контекстное меню, установите цвет для минимального и максимального значения. В установках модели подберите такое соотношение модельного и реального времени, чтобы результаты моделирования были наглядны.

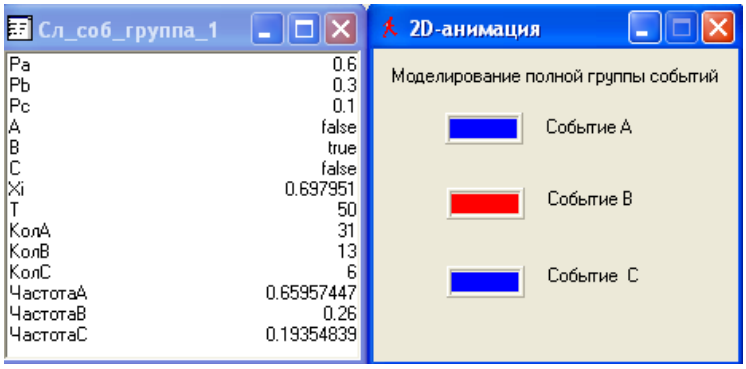

Рис. 6.2.3. Результаты моделирования полной группы событий

Провести так же ряд экспериментов с моделью при разном количестве реализаций и оценить результат моделирования. Подсчитать частоту реализации каждого события.

6.3. Моделирование случайного блуждания

В данной работе моделируется случайное изменение состояния объекта. Состояние объекта характеризуется двумя параметрами ***X*** и ***Y***. Изменение состояния объекта связано с изменением значений этих параметров, происходит оно случайно и дискретно. Каждый параметр может остаться неизменным, либо изменить свое значение на +1 или –1.

Таким образом, возможна реализация 9 взаимоисключающих событий, которые образуют полную группу:

$$X,Y;\ X,Y+1;\ X,Y-1;\ X+1,Y;\ X+1,Y+1;\ X+1,Y-1;\ X-1,Y;\ X-1,Y+1;\ X-1,Y-1.$$

Вероятность изменения каждого параметра задана:

$$PX_a,\ PX_b,\ PX_c;\quad PY_a,\ PY_b,\ PY_c.$$

Причем

$$PX_a + PX_b + PX_c = 1,\quad PY_a + PY_b + PY = 1.$$

Таким образом для каждого параметра возможны следующие события: событие ***A*** – параметр остался неизменным; событие ***B*** – параметр изменился на +1; событие ***C*** – параметр изменился на –1. Изменение значения одного параметра происходит независимо от другого.

Построить компьютерную модель реализации случайного блуждания. Исследовать поведение объекта. Моделирование выполняется в среде MVS. Для решения задачи необходимо построить модель по рис. 6.3.1–6.3.2.

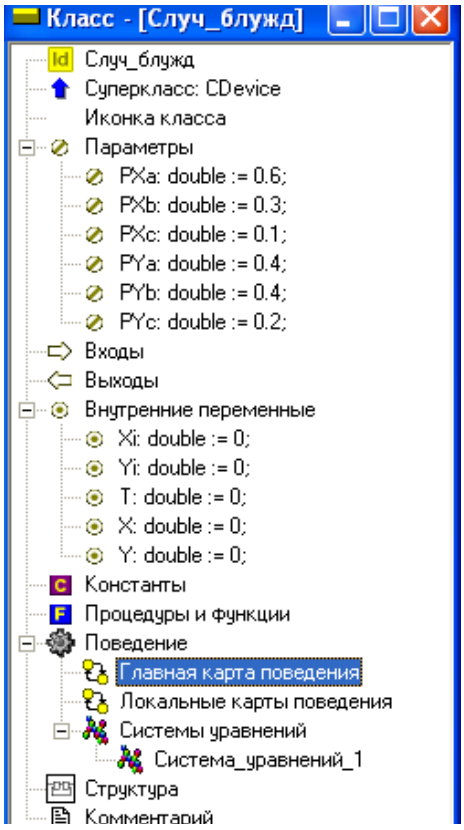

Рис. 6.3.1. MVS-моделирование случайного блуждания

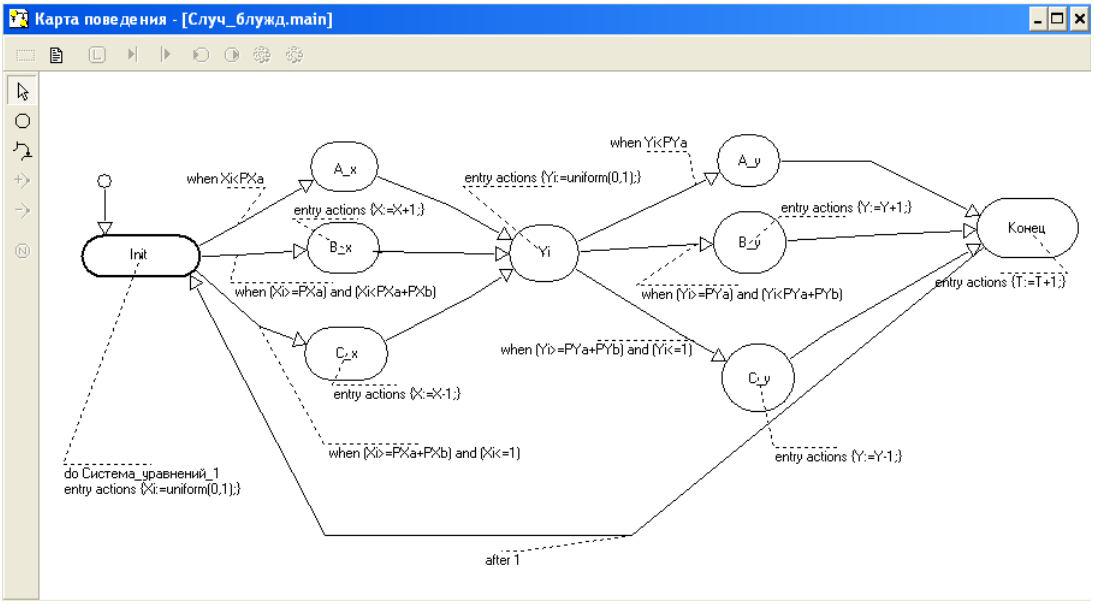

Рис. 6.3.2. Карта поведения для MVS-модели случайного блуждания

Исходными данными для модели является вероятности реализации событий ***A, B, C***: PX_a, PX_b, PX_c; PY_a, PY_b, PY_c. Начальные значения: $X = 0;\quad Y = 0$. При моделировании случайных событий изменения ***X*** и изменения ***Y*** использовать отдельный датчик случайных чисел – функцию **uniform (0,1)**.

Проведите серии экспериментов при разных значениях вероятностей событий и понаблюдайте, как изменяется диаграмма.

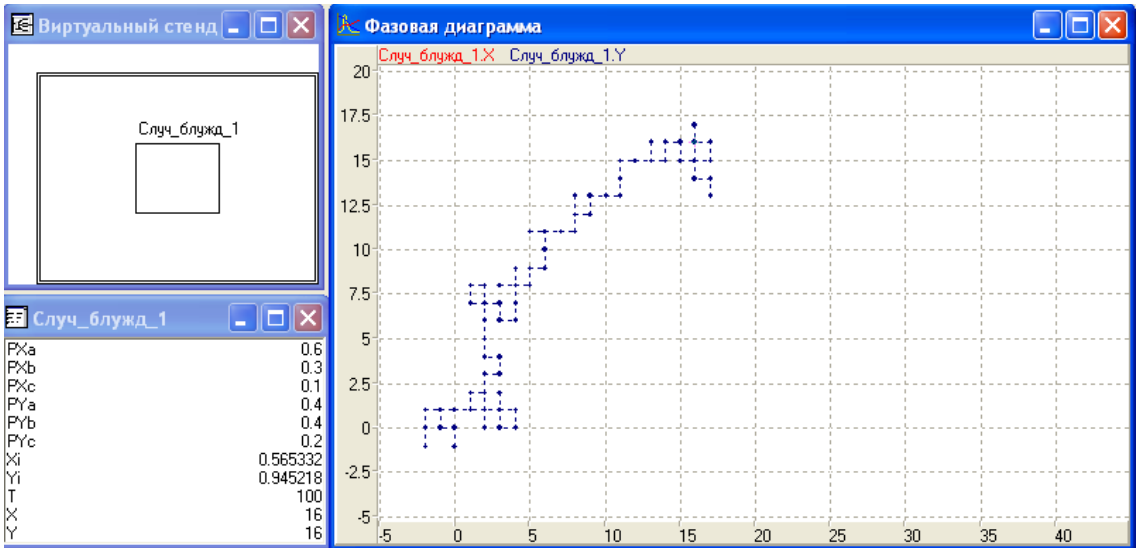

Рис. 6.3.3. Результат моделирования: траектория блуждания

6.4. Имитационная модель работы автомобиля

Рассмотрим пример прогнозирования работы автомобиля. В течение суток автомобиль может быть исправным или неисправным. Будем через ***T,*** обозначать количество суток, прошедших с начала наблюдения, то есть мы выбираем дискретное время с единицей измерения «сутки».

История автомобиля начинается с его прибытия в гараж в момент t_0. Пусть автомобиль может быть исправным или неисправным с одинаковой вероятностью $P_{01} = P_{02} = 0,5$.

Дальнейшее развитие событий можно описать следующими законами: исправный автомобиль может сломаться, но это обнаружат только на следующие сутки, после чего приступят к ремонту; исправный автомобиль будет продолжать работу на следующие сутки; неисправный автомобиль после суток ремонта продолжит работу; неисправный автомобиль не успеют отремонтировать за сутки.

Таким образом:

P_{11} – вероятность исправного автомобиля не сломаться за сутки;

P_{12} – вероятность исправного автомобиля сломаться за сутки;

P_{21} – вероятность неисправного автомобиля не быть починенным за сутки;

P_{22} – вероятность неисправного автомобиля быть починенным за сутки.

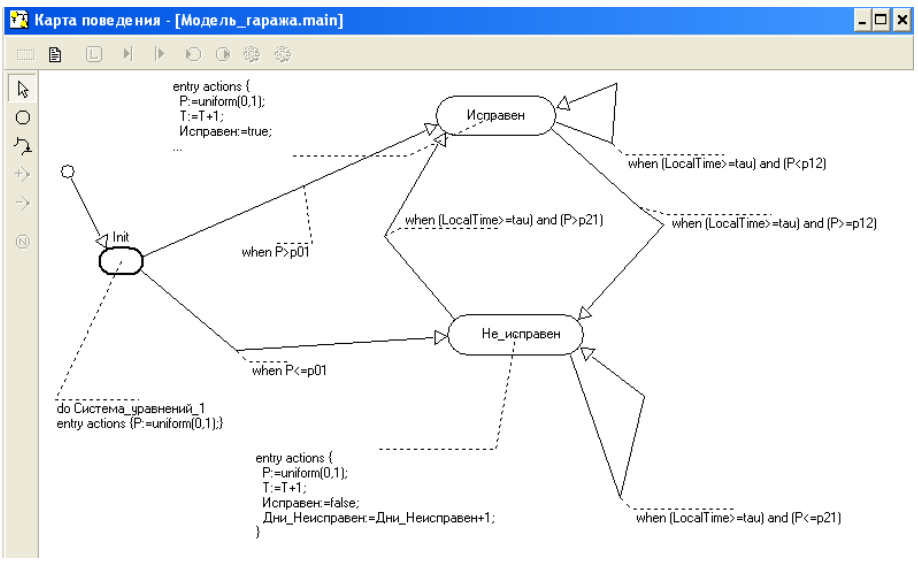

Рис. 6.4.1. Карта состояний при моделировании работы автомобиля

Работу автомобиля можно представить графом (рис. 1), который похож на карту поведения, но у этой карты есть существенное отличие. Переход из текущего в новое состояние осуществляется с учетом вероятностей. А именно, на каждом такте дискретного времени генерируется значение случайной величины, распределенной по заданному закону. На основании этого значения выбирается тот или иной переход, после чего система переходит в новое текущее состояние. Если ввести события: «автомобиль работает», «автомобиль сломался» и так далее и построить генератор событий, выбирающий одно из независимых событий на каждом такте, то можно этот граф представить соответствующей картой состояний, т. к. мы ее трактовали раньше (рис. 6.4.2).

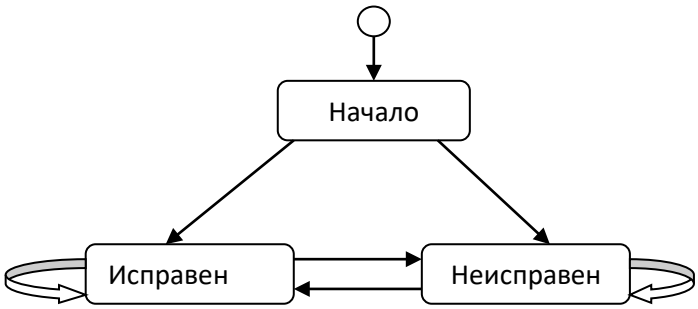

Рис. 6.4.2. Карта поведения при моделировании работы автомобиля

Все вероятности переходов в этой модели зависят только от конкретного состояния, а не от того, как система пришла в данное состояние. Нас интересует вопрос, с какой вероятностью на *N*-е сутки автомобиль будет исправным или будет неисправным.

Система относится к числу стохастических. Вероятность смены состояния постоянна и зависит от того, в каком текущем состоянии находится система.

Вернемся к примеру с автомобилем (см. рис. 6.4.2). На этом рисунке представлена имитационная модель, и теперь уже изображенный граф является картой поведения. Мы моделируем случайное событие возможность в момент прихода директора найти машину исправной или неисправной. Далее модель находится в каждом возможном состоянии ровно сутки и переходит в новое состояние опять же с учетом реализовавшегося случайного события.

Построить компьютерную MVS-модель работы автомобиля и исследовать ее свойства.

Создадим новую модель, назовем ее «Модель гаража». Введем переменные и параметры на рис. 6.4.3.

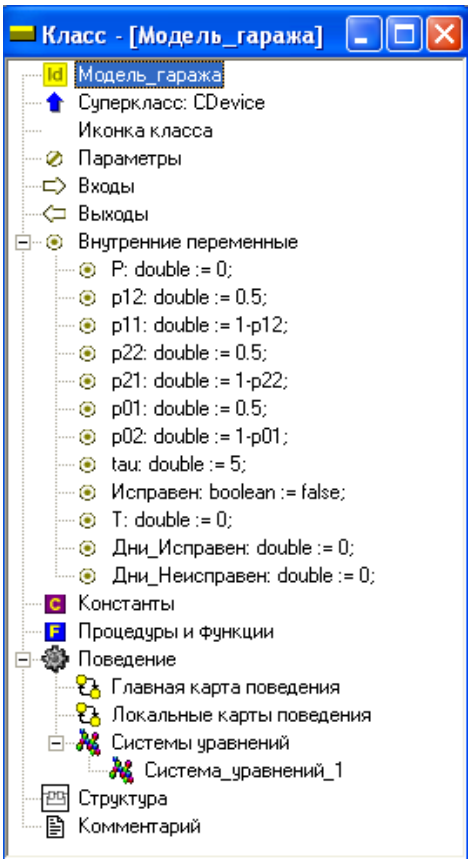

Рис. 6.4.3. Исходные данные для моделирования

• P_{11} — вероятность исправного автомобиля не сломаться за сутки; (double), значение *1*– P_{12};

• P_{12} — вероятность исправного автомобиля сломаться за сутки; (double), значении 0,5;

• P_{21} — вероятность неисправного автомобиля не быть починенным за сутки; (double), значение 1– $P_{22;}$

• P_{22} — вероятность неисправного автомобиля быть починенным за сутки. (double), значении 0,5;

• *P* – переменная (double);

• *T* – количество суток (double);

• P_{01} – вероятность застать машину в начальный момент времени исправной (double), значение 1- P_{02};

• P_{02} – вероятность застать машину в начальный момент времени неисправной (double), значение 0,5;

• *Tau* - типа double, = 5;

• Исправен – типа Boolean;

• Дни_Исправен – количество дней, когда автомобиль исправен;

• Дни_Неисправен – количество дней, когда автомобиль не исправен.

Составим карту поведения как показано на рис. 6.4.2.

Запустим модель

Создайте фазовую диаграмму (рис. 6.4.4).

Перетащите на неё ***T*** и «**Исправен**».

Настройка (Фазовая диаграмма):

Отложить ***T*** по оси X: шаг = 1 Min 0 Max 50.

Рис. 6.4.4. Фазовая диаграмма моделирования работы автомобиля

Добавим новую 2D – анимацию.

Добавим «Цифровой индикатор» и «Цветовой индикатор».

На цифровые индикаторы перетащим ***T***, «**Дни_Исправен**» «**Дни_Неисправен**», на цветовой индикатор «**Исправен**».

На цветовом индикаторе установите цвет **min** и цвет **max** (рис. 6.4.5).

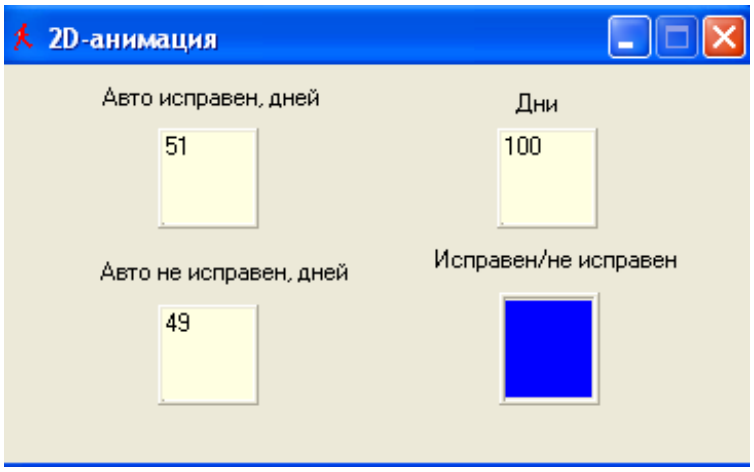

Рис. 6.4.5. Результаты моделирования работы автомобиля

ЗАКЛЮЧЕНИЕ

Лабораторный практикум позволяет студентам, будущим учителям информатики или математики, эффективно использовать программный комплекс MVS в будущей профессиональной деятельности на уроках информатики или математики как при проведении занятий по теме «Моделирование» в курсе «Информатика», так и при использовании моделей для демонстрации при изучении математики или при изучении наук физико-математического или естественнонаучного профиля в университете.

Лабораторный практикум посвящен компьютерному пакету быстрой разработки моделей, который не требует программирования и глубоких знаний в области численных методов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Арнольд, В.И. «Жесткие» и «мягкие» математические модели / В.И. Арнольд. – Москва: МЦ НМО, 2018. – ISBN 978-5-4439-2729-9. – Текст: непосредственный.
2. Бенькович Е. С. Практическое моделирование динамических систем / Е.С. Бенькович, Ю.Б. Колесов, Ю.Б. Сенюченков. – Санкт-Петербург: БХВ-Петербург, 2002. – 464 с. – ISBN 5-94157-099-6. – Текст: непосредственный.
3. Королев А. Л. Компьютерное моделирование. Лабораторный практикум / А.Л. Королев. – Челябинск: Изд-во ЮУрГГПУ, 2022. 258 с. – ISBN 978-5-907611-29-0.
4. Королев, А. Л. Компьютерное моделирование объектов процессов и систем: учебно-практическое пособие / А.Л. Королев, Н.Б. Паршукова. – Челябинск: Издательство ЮУрГГПУ, 2020. – 329 с. – ISBN 978-5-907409-15-6. – Текст: непосредственный.
5. Тарасевич, Ю.Ю. Математическое и компьютерное моделирование. Вводный курс / Ю.Ю. Тарасевич. – Москва: Едиториал УРСС, 2019. – 149 с. – ISBN 5-354-00913-8. – Текст: непосредственный.
6. Майер, Р.В. Психология обучения без огорчения / Р.В. Майер. – Глазов: ГГПИ, 2010. – 36 с. – Текст: непосредственный.

Учебное издание

Королев *Александр Леонидович*

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ В ИНСТРУМЕНТАЛЬНОЙ СРЕДЕ MODEL VISION STUDIUM (MVS)

Практикум

ISBN 978-5-907611-61-0

Работа рекомендована РИС ЮУрГГПУ
Протокол № 27, 2022

Редактор *О.Э. Карпенко*

Издательство ЮУрГГПУ
454080, г. Челябинск, пр. Ленина, 69

Подписано в печать 10.11.2022
Формат 60 × 84 $^{1}/_{8}$. Бумага офисная
Уч.-изд. л. 4,97. Усл. печ. л. 13,49
Тираж 100 экз. Заказ № 765

Отпечатано с готового оригинал-макета
в типографии Южно-Уральского
гуманитарно-педагогического университета
454080, г. Челябинск, пр. Ленина, 69