А.Л. КОРОЛЕВ, Н.Б. ПАРШУКОВА

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ОБЪЕКТОВ, ПРОЦЕССОВ И СИСТЕМ

Учебно-практическое пособие

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

А.Л. КОРОЛЕВ, Н.Б. ПАРШУКОВА

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ОБЪЕКТОВ, ПРОЦЕССОВ И СИСТЕМ

УЧЕБНО-ПРАКТИЧЕСКОЕ ПОСОБИЕ

Челябинск
2022

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

Королев, А.Л. Компьютерное моделирование объектов, процессов и систем: учебно-практическое пособие / А.Л. Королев, Н.Б. Паршукова. – Челябинск: Изд-во Южно-Урал. гос. гуманитар.-пед. ун-та, 2022. – 308 с. – Текст: непосредственный.

ISBN 978-5-907611-11-5

Пособие содержит тексты лабораторных работ и методические указания к ним. Лабораторные работы предназначены для курса «Моделирование систем» для студентов бакалавриата, обучающихся по направлению: 09.03.02 «Информационные системы и технологии», профиль: «Информационные технологии в образовании».

Цель пособия – формирование у студентов умений разрабатывать компьютерные модели объектов, процессов и систем на основе современной методологии моделирования с использованием современных технологий и основных естественно-научных законов, а также проводить модельные исследования и компьютерные эксперименты в области профессиональной деятельности. Лабораторные работы построены на доступном для образовательных целей программном обеспечении.

Королев А.Л. – главы 1–4
Паршукова Н.Б. – глава 5

ISBN 978-5-907611-11-5

Рецензенты:

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

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Моделирование является общенаучным методом изучения законов окружающего мира, свойств объектов и систем самой различной природы или при создании новых объектов и систем. Моделирование как метод – мощный инструмент науки и техники. Любое исследование заканчивается построением определенной модели объекта или процесса, т.е. выделением главных факторов которые определяют поведение объекта в конкретных условиях. Таким образом, методология моделирования во многом совпадает с методологией научных исследований. Как показывает опыт, активное участие в моделировании вырабатывает более глубокое понимание сути протекающих процессов и наблюдаемых явлений и реализует роль компьютера как инструмента исследования.

Развитие компьютерных технологий предоставляет в профессиональной деятельности новые возможности 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 путем непосредственного

программирования с получением в итоге программного комплекса требует значительных затрат времени и средств, а также специфических знаний и умений в области программирования и численных методов.

В этом смысле компьютерное моделирование на основе специализированных инструментальных программных комплексов типа «КОМПАС 3D», MVS, RMD, AnyLogic 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 программы. Это позволяет использовать технологию компьютерного моделирования в учебном процессе по ряду дисциплин профессионального и естественнонаучного 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 представлений, умений и навыков 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 раз-

личных областях практической деятельности. Таким образом, пособие позволяет решить следующие задачи подготовки ИТ-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 области применения межпредметных знаний.

В результате изучения представленного в пособии материала, студенты должны быть способны использовать основные законы естественно-научных дисциплин в профессиональной деятельности, применять методы математического анализа и моделирования, теоретического и экспериментального исследования, проводить моделирование процессов и систем. Результатом формирования компетенции в области моделирования будет являться готовность к участию в постановке и проведению экспериментальных исследований; способность обосновывать правильность выбранной модели, сопоставляя результаты экспериментальных данных и полученных решений; способность использовать математические методы обработки, анализа результатов профессиональных исследований; способность применять естественно-научные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности, а также применение математических моделей, методов и средств проектирования при моделировании систем.

Пособие будет полезно студентам бакалавриата направления: 09.03.02 «Информационные системы и технологии» при изучении курса «Моделирование систем».

Настоящее пособие с учетом опыта преподавания курсов «Компьютерное моделирование» и «Моделирование систем», изменения объема аудиторной работы дополнено, переработано и является продолжением учебного пособия: Королев, А.Л. Компьютерное моделирование. Лабораторный практикум. Москва: ЛБЗ-БИНОМ, 2012. – 296 с., – ISBN 978-5-9963-6270-3.

За время, прошедшее с момента написания данного пособия, изменились требования по составу компетенций в курсе «Моделирование систем» и появились новые программные комплексы и технологии. Содержание лабораторных работ реализует межпредметный подход и начальное освоение исследовательских компетенций. Таким образом, переработка и обновление учебных пособий становятся актуальной задачей.

ГЛАВА 1. 3D-МОДЕЛИРОВАНИЕ В СРЕДЕ «КОМПАС 3D»

1.1. Инженерная цифровая среда проектирования

Исторически идеи создания какого-либо объекта предварительно воплощались в виде чертежей, а проверялись при изготовлении материальной модели, опытных образцов в лабораториях и при испытаниях. Следовательно, проектирование начиналось с создания **графических геометрических моделей**. Поэтому традиционное построение графических моделей было рассчитано на использование чертежных инструментов и различных видов проецирования (в основном ортогонального проецирования), выполнение эскизов и чертежей, нанесение размеров.

Геометрические модели отображают форму, размеры, взаимное расположение частей, ориентацию в пространстве, т.е. **геометрические свойства** объектов.

Графические модели отражают свойства объекта, естественно, средствами графики. Основная технология компьютерного графического моделирования – **векторная графика**.

Главная проблема компьютерного геометрического моделирования – создание графического отображения **трехмерных объектов** на **плоской поверхности.** В техническом черчении разработана методика построения изометрических проекций, которая позволяет на бумаге статично отобразить трехмерный объект.

Для кодирования изображения используется векторная и растровая графика. Растровый рисунок – **матрица цветных точек** (пикселей). Векторный рисунок – **совокупность графических объектов**.

Каждый графический объект имеет набор параметров и методов его построения. Перед выводом изображения объекта на экран программа производит вычисления координат экранных точек в изображении объекта с учетом масштаба. Векторную графику называют **параметрической графикой.**

В основе векторной графики лежит математическое описание геометрических свойств объектов.

Особенности векторной графики

1. Масштабирование без потери качества изображения с проявлением мелких деталей.

2. Возможность проведения преобразований путем изменения параметров графических объектов.

3. Возможность программной обработки параметров, следовательно, самих графических объектов.

Результат векторного геометрического моделирования – математическая модель геометрии объекта.

Математическая модель позволяет

- графически отобразить объект на экране и проводить с его изображением различные манипуляции (вращение, сдвиг и т.д.);
- рассчитать его геометрические характеристики;
- рассчитать массовые и кинематические характеристики объекта;
- исследовать физические свойства объекта;
- подготовить программный код для изготовления объекта на станке с числовым программным управлением (ЧПУ).

Технология компьютерного моделирования трехмерных объектов имеет следующие виды:

Трехмерное каркасное моделирование

- Отображение только ребер и вершин объекта (рис. 1.1.1).
- Перенос «бумажных» технологий на компьютер.
- Поверхности не определены.
- Трудность восприятия модели.

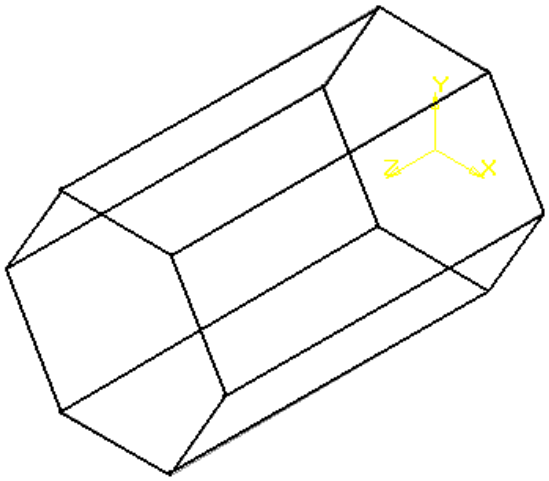


Рис. 1.1.1. Каркасная 3D-модель

Трехмерное поверхностное моделирование

Объект представляется совокупностью тонких поверхностей, под которым находится пустое пространство. Поверхность объекта описана математической зависимостью. Построение поверхности производится с помощью «проволочной» сетки серией пересекающихся линий, принадлежащих поверхности объекта (рис. 1.1.2).

- Поверхностное моделирование разработано для моделирования объектов из листового материала.
- Использует плоскости, поверхности вращения, поверхности параллельного переноса, математические поверхности.
- Недостаток: трудно моделировать внутренние пространства.

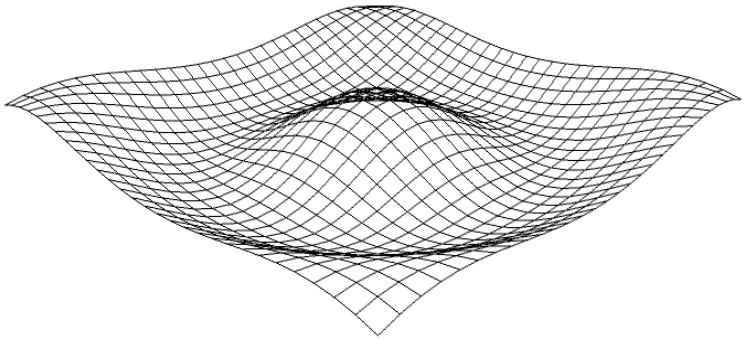


Рис. 1.1.2. Отображение поверхности

Трехмерное твердотельное моделирование

Твердотельная модель – это несколько оболочек, одна оболочка является наружной, а остальные оболочки являются внутренними. Для всех оболочек задано, с какой стороны поверхности находится материал тела. Твердотельная модель позволяет отображать и геометрические, и некоторые физические свойства объекта, например, распределение температуры по телу (рис. 1.1.3).

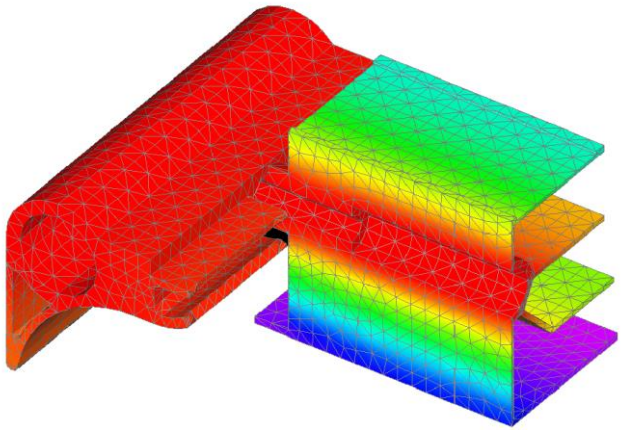


Рис. 1.1.3. Твердотельная 3D-модель с отображением полей температуры

Развитие аппаратных и программных графических средств, включая трехмерное твердотельное 3D-моделирование, изменили методологию, т.е. логическую организацию, методы и средства инженерной разработки проектов.

Эффективное использование всей мощи новых компьютерных инструментов, включая 3D-печать, требует широкого диапазона разнообразных навыков. Пространственное воображение становится все более важным, так как компьютерные трехмерные модели все чаще заменяют изготовление реальных вещественных моделей. Инженер должен представлять, какой вид будет иметь создаваемое им изделие, и должен уметь представить это изделие с помощью программ системы автоматизированного проектирования (САПР).

САПР определяется как проектирование, осуществляемое взаимодействием человека и ЭВМ. Под проектированием здесь понимается про-

цесс разработки описания, необходимого для создания в заданных условиях еще не существующего объекта.

Свойства САПР

1. В САПР создается объектно-ориентированный цифровой документ.

2. В САПР создается математическая модель геометрии объекта, которая может быть передана для обработки в другие приложения, например, для получения чертежей, для исследования свойств модели математическими методами, для подготовки управляющих программ для станков с ЧПУ, для подготовки файлов для 3D-принтера.

3. В САПР создается электронный цифровой документ (со всеми его преимуществами). Средства САПР предоставляют пользователю «виртуальные» электронные инструменты построения и моделирования, свойства которых значительно превосходят возможности традиционных инструментов черчения.

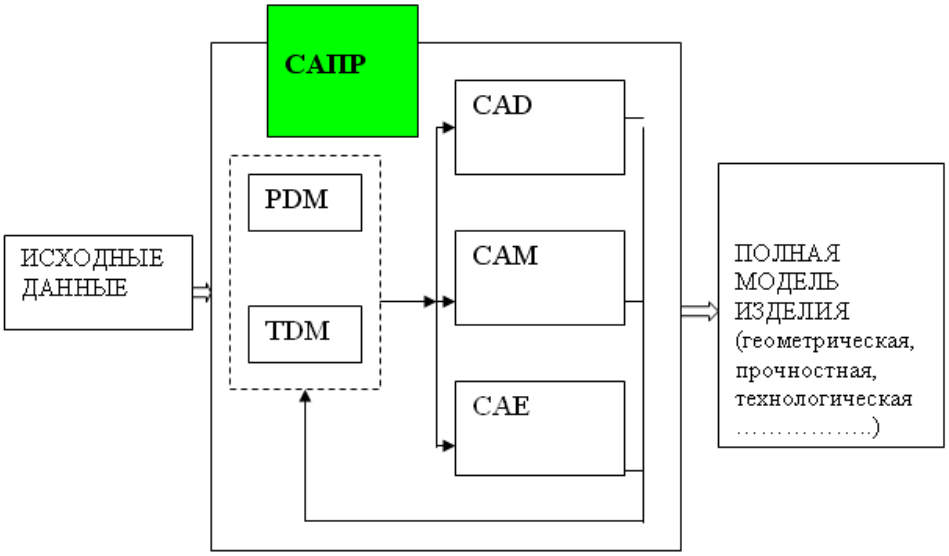


Рис. 1.1.4. Структура САПР

Отдельные модули САПР: **CAD, CAM, CAE, TDM** – решают весь спектр производственных задач (рис. 1.1.4).

CAD (Computer Aided Design) – модуль компьютерного геометрического (проектирования) моделирования.

CAM (Computer Aided Manufacturing) – модуль технологической подготовки производства.

CAE (Computer Aided Engineering) – модуль компьютерного инженерного анализа.

PDM (Product Data Management) – модуль, позволяющий управлять данными о продукции на протяжении всего жизненного цикла изделия при проектировании и подготовке производства.

TDM (Technical Data Management) – модуль управления базами данных, включая документооборот конструкторской и технологической документации.

Во всех современных системах трехмерного моделирования построение моделей производится с помощью операций твердотельного моделирования: это объединение, вычитание и пересечение, которые производятся с трехмерными объектами**.** Эти объекты создает пользователь по определенным правилам при построении модели. Многократное применение названных операций позволяет построить достаточно сложную трехмерную модель.

Для создания объемных элементов используются трехмерные операции, которые связаны с перемещением плоских фигур в пространстве. При этом ограничивается часть пространства, которая определяет форму элемента: перемещение прямоугольника приводит к формированию призмы; в результате поворота линии вокруг оси симметрии будет сформировано тело вращения; перемещение окружности вдоль направляющей формирует объемный элемент – круглый стержень и т.п.

Плоская фигура, в результате перемещения которой образуется объемное тело, называется **эскизом,** а само перемещение – **операцией**. Эскиз может располагаться в одной из стандартных плоскостей проекций (**фронтальная, горизонтальная, профильная**), на плоской грани существующего тела или на плоскости, положение которой определено пользователем.

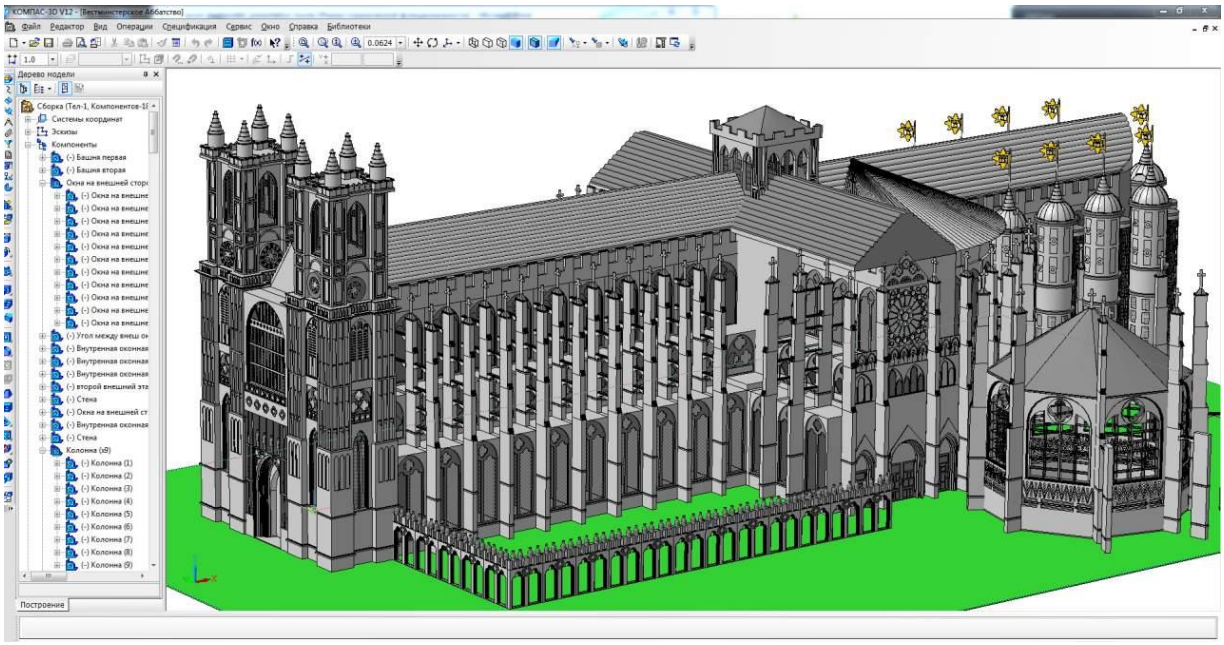


Рис. 1.1.5. 3D-модель собора Парижской Богоматери

Операции, которые можно выполнять при построении модели:

- **Выдавливание** эскиза в направлении, перпендикулярном плоскости эскиза.
- **Вращение** эскиза, которое осуществляется вокруг оси в плоскости эскиза.
- **Кинематическая операция** – перемещение эскиза вдоль направляющей.
- **Операции по сечениям** – построение объемного элемента по нескольким эскизам, которые располагаются в нескольких параллельных плоскостях.
- **Сечение тела поверхностью** и другие операции.
- **Операция вырезания** (выдавливанием, вращением, кинематически).

Таким образом, процесс создания трехмерной модели заключается в многократном добавлении или вычитании дополнительных объемов, каждый из которых представляет собой элемент, образованный при помощи операций над плоским эскизом. Объемные элементы, из которых состоит трехмерная модель, образуют в ней грани, ребра и вершины. В

итоге получается трехмерная модель объекта, построенная по технологии векторной графики (рис. 1.1.5).

1.2. Построение первой 3D-модели

Для перехода в подсистему трехмерного моделирования «КОМПАС 3D» выполните команду **«Файл» – «Создать»**. В окне **«Новый документ»** выберите **«Деталь»**.

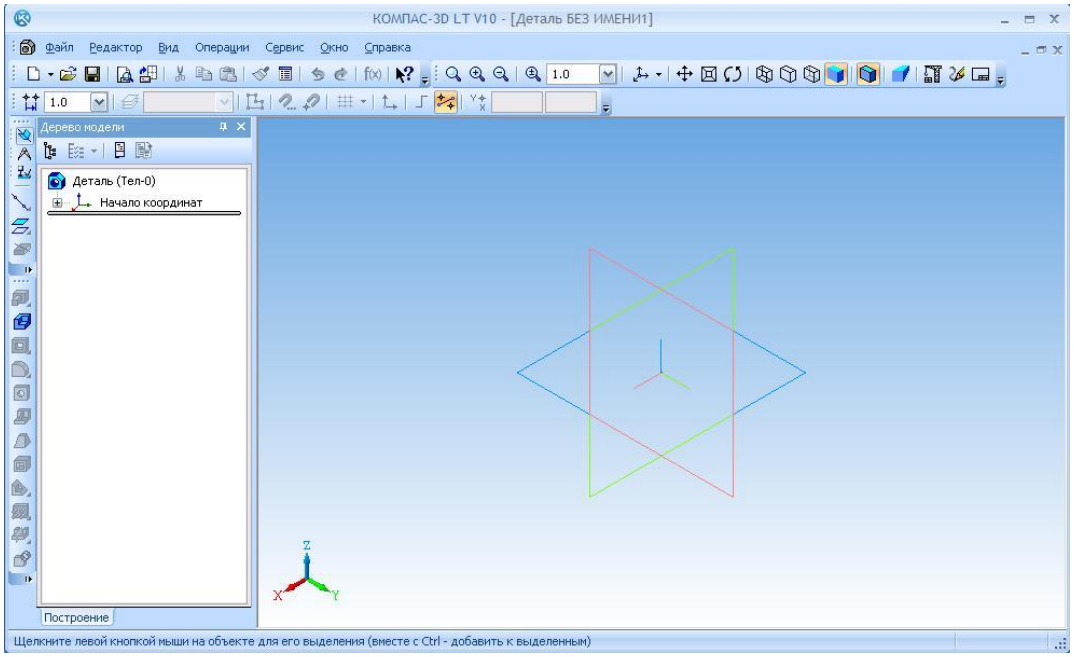


Рис. 1.2.1. Интерфейс системы «КОМПАС 3D» в режиме 3D-моделирования

«Инструментальная панель» трехмерных операций (рис. 1.2.2) по умолчанию расположена в левой части главного окна и состоит из пяти страниц. Для переключения между страницами используются кнопки переключения, расположенные над **«Инструментальной панелью»**.

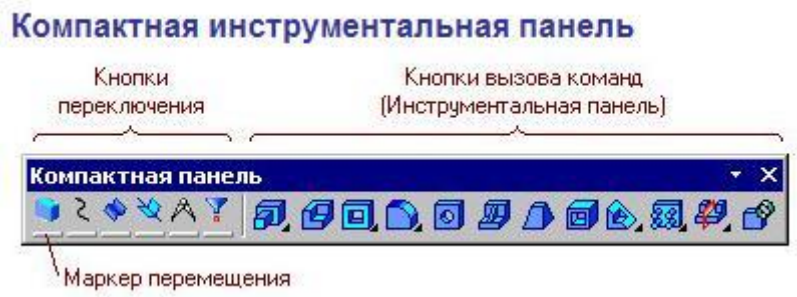


Рис. 1.2.2. Инструментальная панель 3D-операций и «Панель переключения»

На инструментальной панели некоторые кнопки сгруппированы по вариантам возможного выполнения. Такие кнопки обозначены небольшим **черным треугольником** в правом нижнем углу. Для получения доступа к другим командам надо щёлкнуть и удерживать кнопку мыши нажатой некоторое время. При появлении панели расширенных команд, надо установить курсор на нужную кнопку и отпустить клавишу мыши.

Инструментальная панель обеспечивает доступ к следующим операциям: выдавливание, вращение, кинематическая операция, операция по сечениям; деталь-заготовка; вырезать выдавливанием, вырезать вращением, вырезать кинематически, вырезать по сечениям; скругление, фаска; отверстие; ребро жесткости; уклон; оболочка; сечение поверхностью.

Весь ход построения трехмерной модели отражается в **«Дереве модели»** (рис. 1.2.3; 1.2.4).

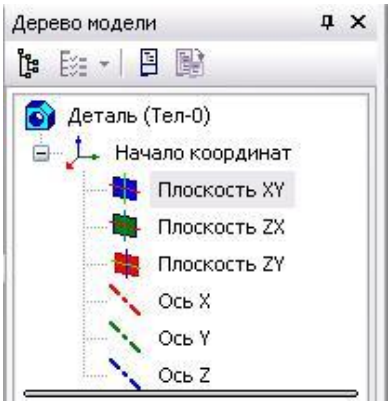


Рис. 1.2.3. Начальное состояние дерева модели

Окно **«Дерева модели»**, в котором в виде списка отображаются все выполненные трехмерные операции (рис. 1.2.4), первоначально содержит три стандартные плоскости проекций и символ начала координат.

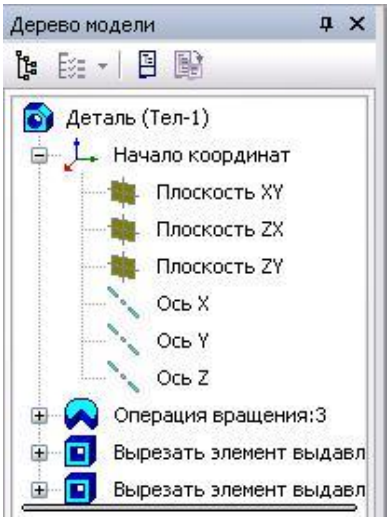
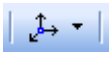


Рис. 1.2.4. Дерево построения после выполнения 3D-операций

В каждой трехмерной модели существует система координат и плоскости проекций. Эти объекты появляются в «**Дереве модели»** сразу после создания новой детали.

Раскройте список видов, выполнив команду **«Вид» – «Ориентация»** или выбрав пиктограмму  на панели инструментов, выберите вид **«Изометрия YZX»** (рис. 1.2.5).

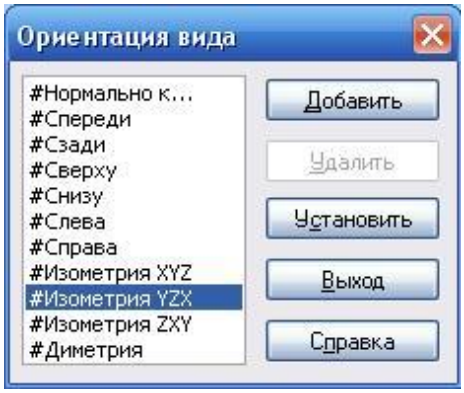


Рис. 1.2.5. Список видов

Установленный по умолчанию цвет плоскости в окне модели можно изменить. Для этого выполните команду **«Сервис»–«Параметры».** В окне диалога (рис. 1.2.6) выберите **«Свойства абсолютной СК».**

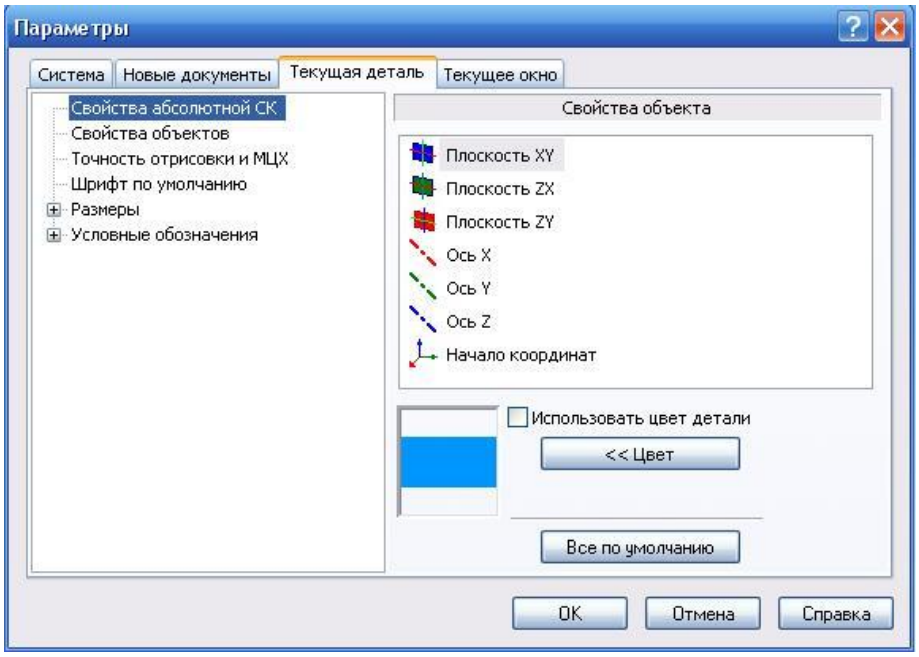


Рис. 1.2.6. Окно диалога «Настройка параметров текущей детали»

В окне **«Свойства объекта»** выберите **«Плоскость XY».** Теперь нажмите **«Цвет»** и выберите, например, оранжевый. Не забудьте нажать кнопку ОК.

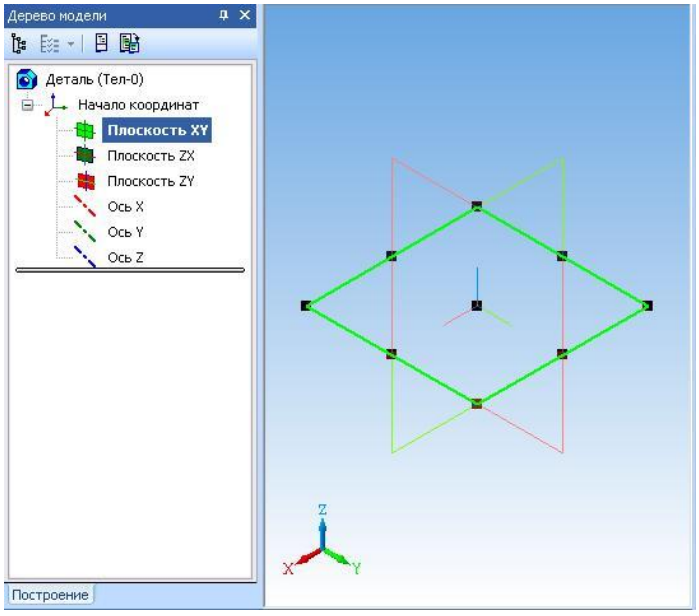
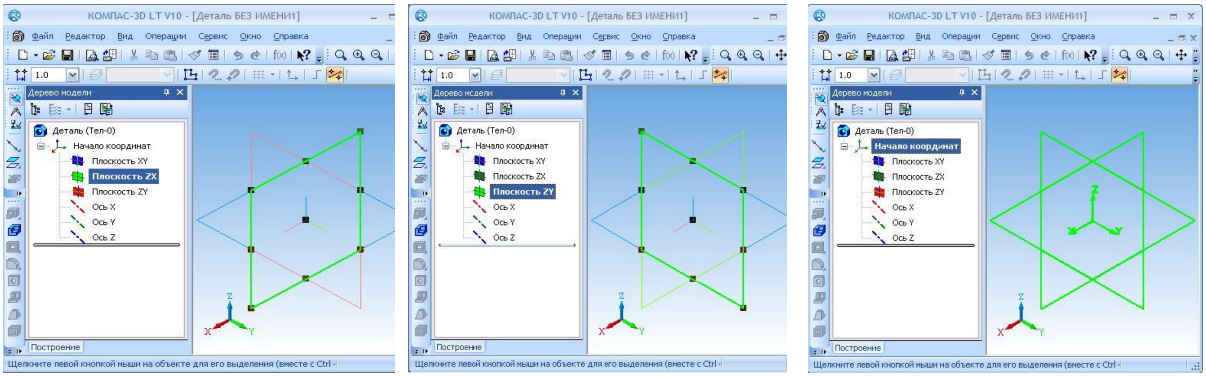


Рис. 1.2.7. Плоскость XY

Информация о масштабе текущей детали расположена в верхней части окна. 

. Данная область панели позволяет производить необходимые действия по масштабированию.

Выберите в **«Дереве модели»** элемент **«Плоскость XY».** Вы увидите (рис. 1.2.7) условное изображение плоскости XY в виде прямоугольника с характерными точками. Размер прямоугольника, конечно, можно поменять, но этого делать пока не нужно. Последовательно выберите в **«Дереве модели»**: **«Плоскость ZX»**, **«Плоскость ZY»** и **«Начало координат»**. При этом будут выделяться соответствующие элементы (рис. 1.2.8).



а Б в

Рис. 1.2.8. а – плоскость ZX, б – плоскость ZY, в – изометрия

Термином «**эскиз**» в системе «КОМПАС» обозначается плоская фигура, на основе которой образуется объемный элемент. Построение эскиза осуществляется с помощью инструментов подсистемы плоского черчения.

При построении трехмерной модели всегда необходимо задать **плоскость построения эскиза**. В **«Дереве модели»** выберите **«Плоскость XY».**

Для построения эскиза нажмите кнопку  . На рис. 1.2.9 вы видите инструментальную панель **«Геометрия»** и переключатель инструментальных панелей. Эскиз удобнее строить, если его плоскость совпадает с плоскостью экрана. Для этого необходимо раскрыть список стандартных ориентаций в строке текущего состояния и выбрать вид **«Спереди»** (рис. 1.2.10).

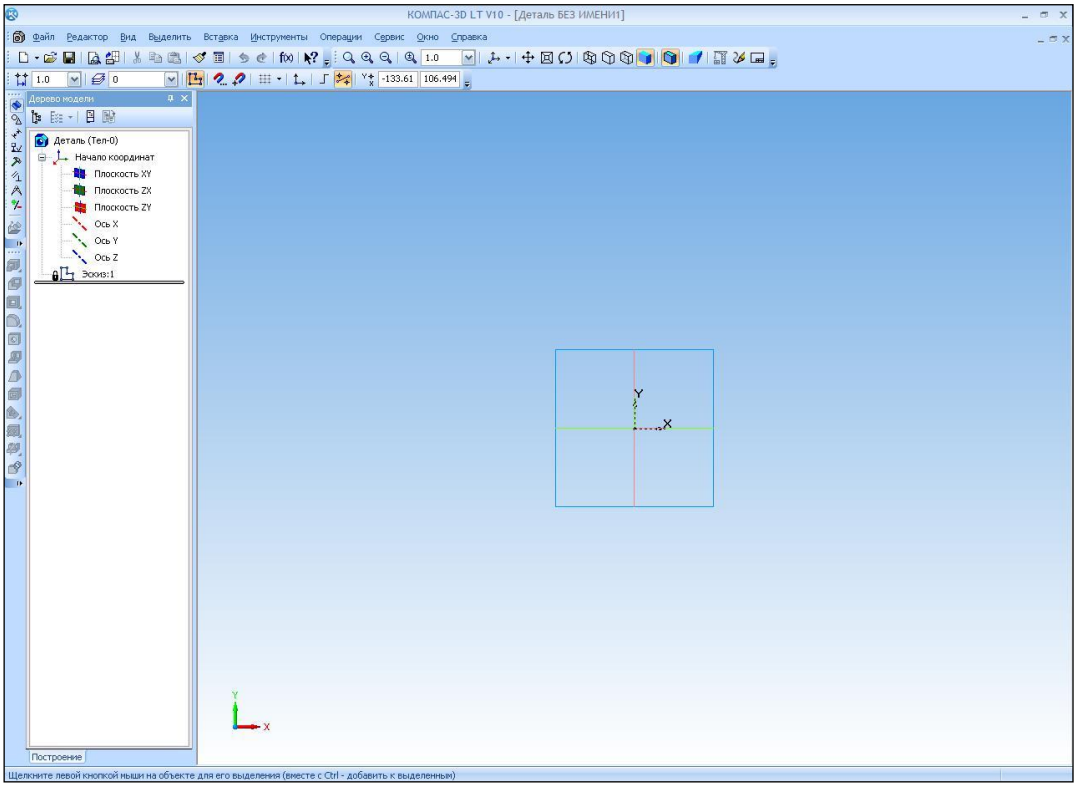


Рис. 1.2.9. Экран системы при создании эскиза

Режим редактирования эскиза представляет собой режим плоского черчения. Для обслуживания этого режима меняется набор кнопок на **«Панели управления»** и на **«Инструментальной панели»**, а также состав **«Строки меню».**

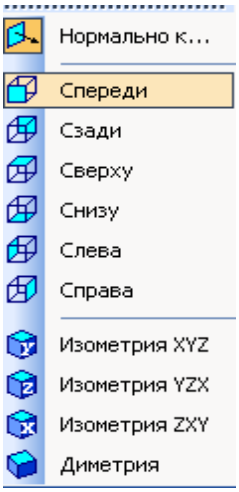


Рис. 1.2.10. Выбор вида

а б в

Рис. 1.2.11. Панели инструментов

а – панель геометрических построений;

б – панель размеров;

в – панель редактирования

Изображение в эскизе должно подчиняться определенным правилам. Построение эскиза выполняется линиями стиля «**Основная».** Такие линии имеют синий цвет. Для выбора стиля линии вызывается пункт контекстного меню **«Стиль линии»** (рис. 1.2.12).

Контуры в эскизе должны быть *замкнутыми, не должны пересекаться или иметь общих точек*. Система не сформирует трехмерный объект, если эскиз будет выполнен неудовлетворительно.

Для операции **«Выдавливание»** в эскизе может быть один или несколько контуров. Единственный контур может быть замкнутым или разомкнутым. Если контуров несколько, то все они должны быть замкнутыми, один из них должен быть наружным, а другие должны быть вложенными в него. Допускается один уровень вложенности контуров.

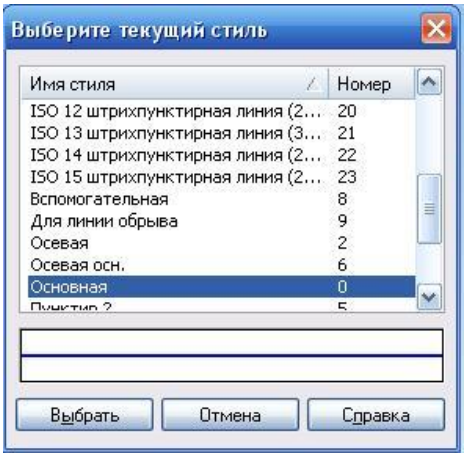


Рис. 1.2.12. Выбор стиля линии

В процессе построения эскиза необходимо точно устанавливать положение курсора в различные точки. Этому помогает технология **привязок**, которая поддерживается системой (рис. 1.2.13).

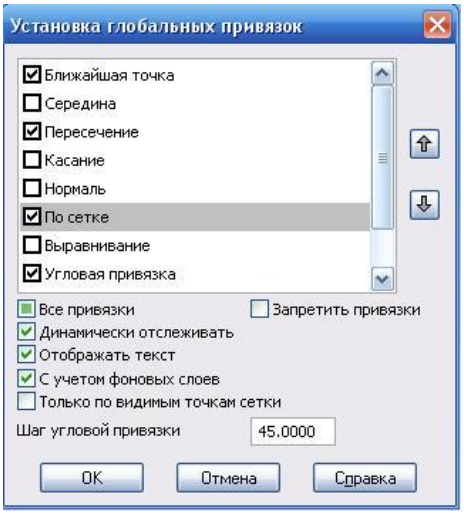


Рис. 1.2.13. Установка привязок

Начало построения трехмерной модели

Построить тонкую пластину, лежащую в **«Плоскости ZY»**.

1. Выберем **«Плоскость XY»** и построим отрезок в трехмерном пространстве с координатами (X = 0, Y = – 50, Z = 0) и (X = 0, Y = 50, Z = 0).

Так как мы выбрали **«Плоскость XY»**, то для всех точек этой плоскости координата Z = 0. Выберете ориентацию **«Нормально к…».** Далее необходимо выполнить команду **«эскиз»**  , установить режим **«сетка»**  и привязки **«По сетке»**  . После чего построить отрезок. Для этого потребуется команда **«Ввод Отрезка»**  на панели **«Геометрия».**

Выберете по сетке координаты начальной точки отрезка (X = 0, Y = – 50) и координаты конца отрезка – (X = 0, Y = 50). По умолчанию шаг сетки равен 5 мм. Третью координату Z вводить не нужно, т.к. она определена выбором плоскости построения. Выбрать пункт контекстного меню **«Создать отрезок»** или нажать комбинацию клавиш Ctrl + Enter. Вид экрана показан на рис. 1.2.14.

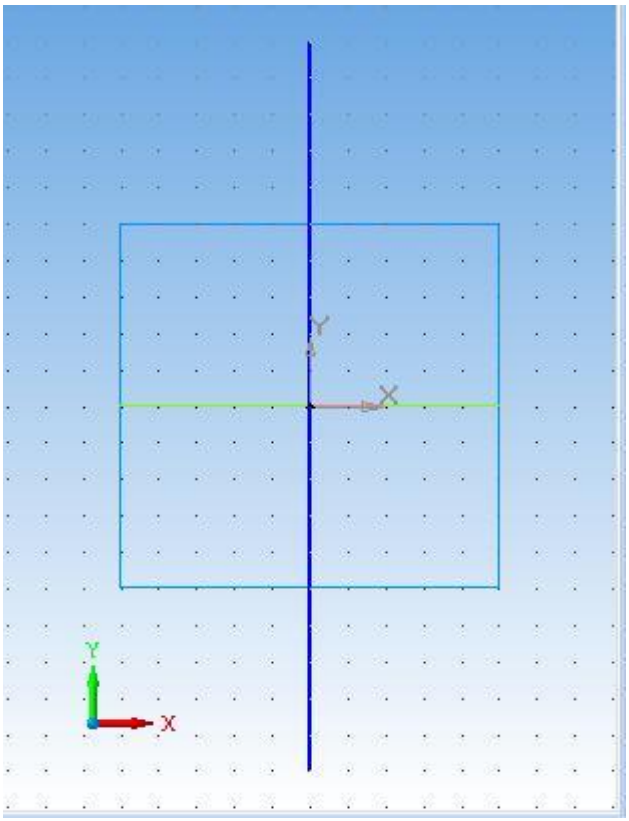


Рис. 1.2.14. Эскиз первой операции

Нажмите кнопку **«Эскиз»** для завершения создания эскиза. Отрезок в **плоскости XY** построен (рис. 1.2.15). В дереве модели появился новый элемент построения – **«Эскиз: 1**».

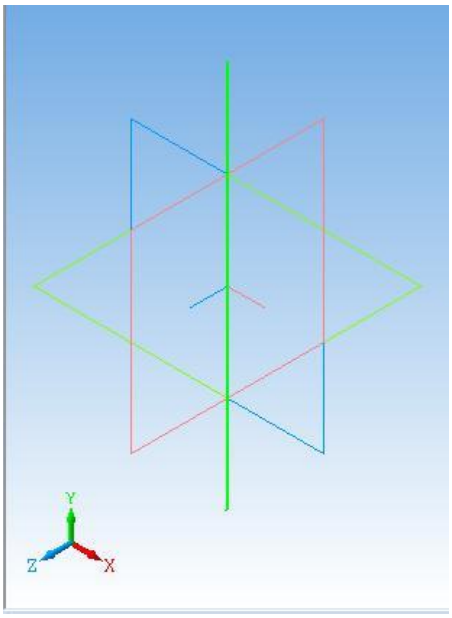


Рис. 1.2.15. Построенный отрезок. Ориентация – «Изометрия XYZ»

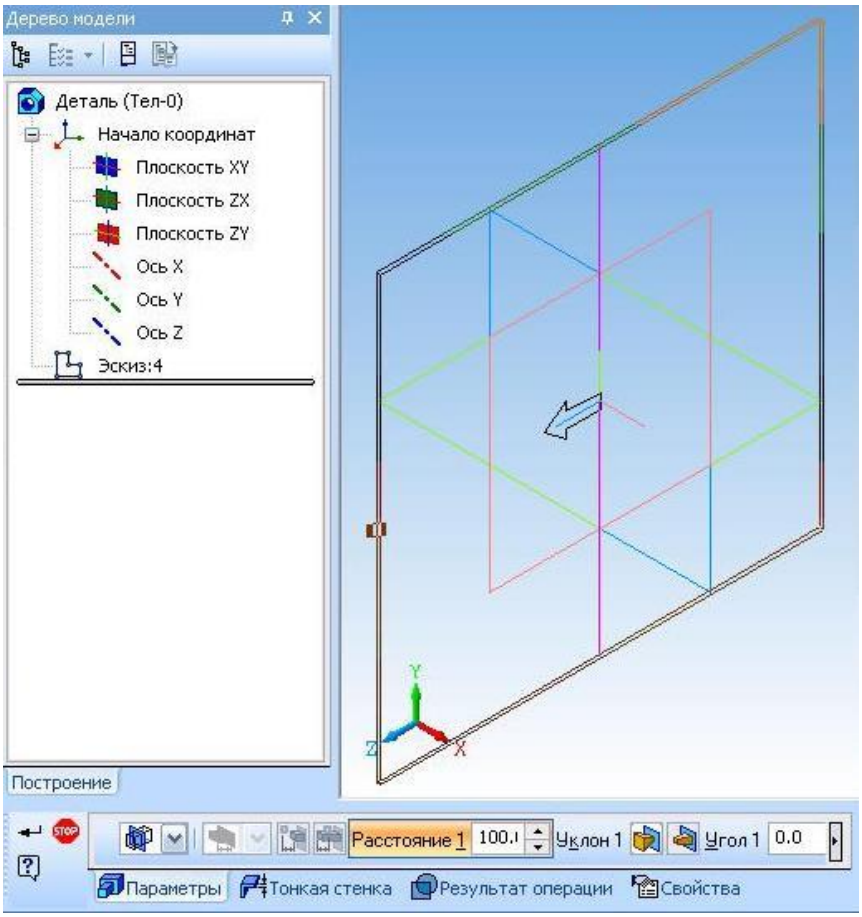


Рис. 1.2.16. Окно диалога операции «Выдавливания»

2. Выберите в дереве модели **«Эскиз:1»**.

Слева на «Инструментальной панели**»** редактирования детали  активна кнопка команды **«Операция выдавливания»**  . Она позволяет построить деталь-основание путем перемещения эскиза (в данном случае это отрезок) в направлении, перпендикулярном его плоскости. Отрезок лежит в плоскости **XY**, следовательно, выдавливание будет происходить в направлении плоскости **ZY** (рис. 1.2.16).

3. Для установки параметров операции выдавливания переместите указатель мыши на панели инструментов и выберите пункт **«Панель свойств»** контекстного меню или выполните команду: **«Вид»–«Панели инструментов»–«Панель свойств».** Выберите параметры, как показано на рисунках.

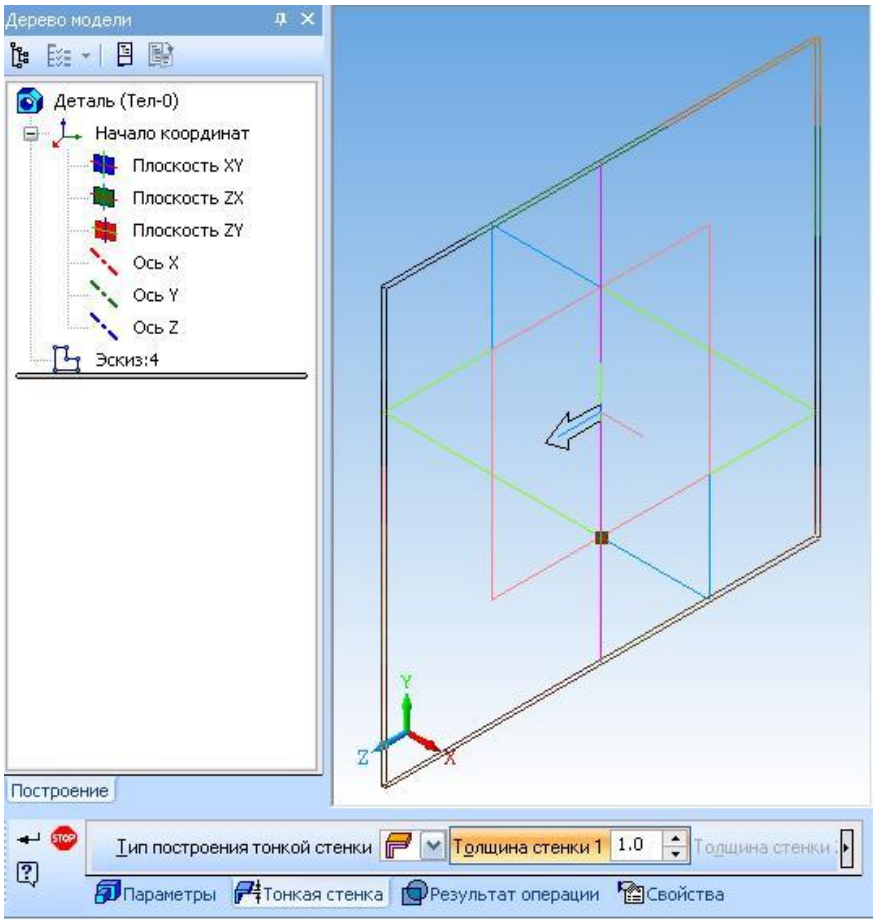


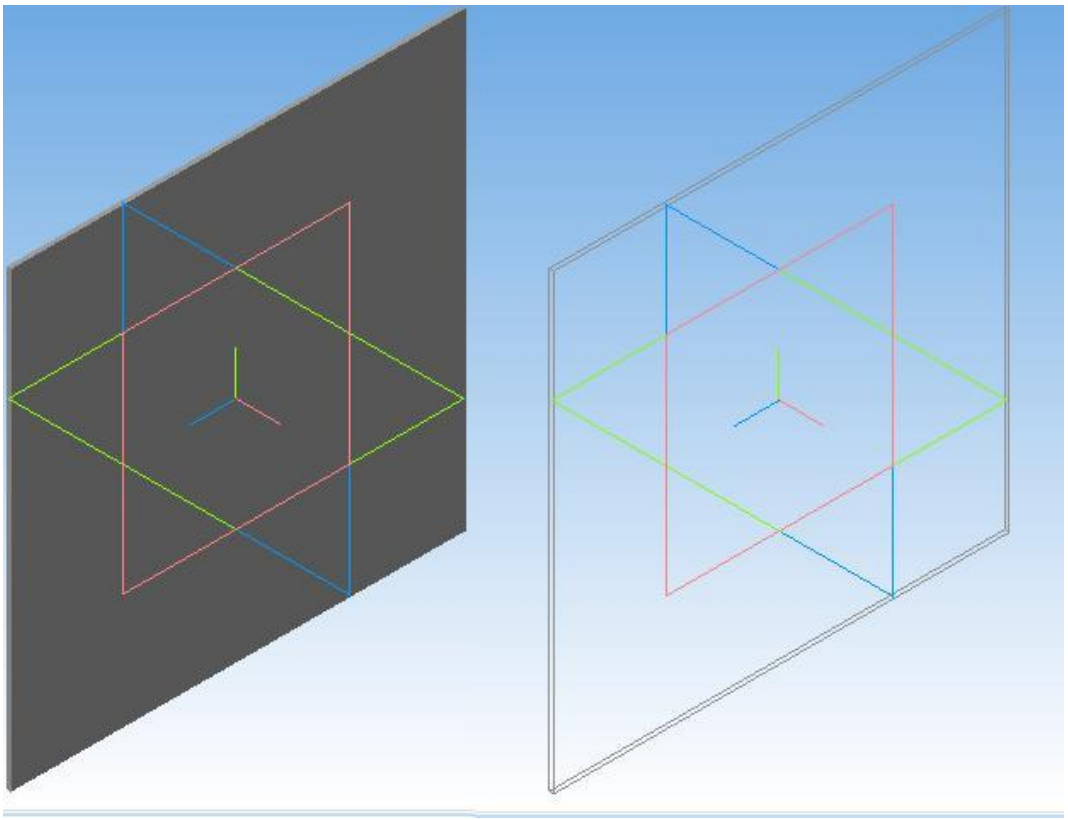
Рис. 1.2.17. Окно диалога операции «Выдавливания» и параметры тонкой стенки

По отрезку система «КОМПАС 3D» предлагает построить тонкую стенку с заданными параметрами.

4. Теперь до выбора команды контекстного меню **«Создать операцию»** или нажатия на кнопку можно «поэкспериментировать» с фантомом тонкой стенки. При этом необходимо выяснить, на что влияют параметры операции выдавливания: «Прямое направление», «Обратное направление», «Два направления», «Средняя плоскость».

5. Установите направление **«Средняя плоскость»** и **«Расстояние»** 100 мм. Уклон не устанавливать.

6. Перейдите на закладку **«Параметры тонкой стенки»** и выясните, на что влияют задания параметров: «Наружу», «Внутрь», «Два направления» и «Средняя плоскость». Для этих экспериментов удобно установить значение **«Толщина»** (тонкой стенки), равное 5 мм.



Полутоновое Каркас

Рис. 1.2.18. Построенная тонкая стенка

7. После окончания экспериментов установите флажок **«Средняя плоскость», «Толщина»** (1 мм) и нажмите кнопку **«Создать»**. При этом будет построена тонкая квадратная пластина толщиной 1 мм, которая лежит в профильной плоскости, т.е. в плоскости **ZY**. Вы можете выбрать команду **«Полутоновое»** –  и увидеть соответствующее изображение детали (рис. *1.2.*18).

Продолжение построения 3D-модели

Построить тонкую квадратную пластину, лежащую в **«Плоскости XY»** толщиной 1 мм, сторона квадрата 100 мм. (Продолжение предыдущего задания).

1. В **«Дереве модели»** выберите **«Плоскость ZY**», установите текущую ориентацию **«Нормально к…».**

2. Для построения эскиза нажмите кнопку команды **«Новый Эскиз»** – 

.

3. Построим отрезок в трехмерном пространстве с координатами (X = 0, Y = – 50, Z = 0) и (X = 0, Y = 50, Z = 0). Так как мы выбрали **«Плоскость ZY»**, то для всех точек этой плоскости координата X = 0.

4. Выберите команду **«Ввод отрезка»** и задайте указанные координаты начальной точки отрезка (Z = 0, Y = – 50) и конечной (Z = 0, Y = 50) точек отрезка. Нажмите кнопку команды **«Эскиз»** –  . Отрезок в плоскости **ZY** построен, ориентация **«Изометрия XYZ»** (рис. 1.2.19).

В дереве модели появился новый элемент построения – **«Эскиз:2»**.

5. Выберите в дереве построений **«Эскиз:2»**.

6. Выполните команду операция **«Выдавливания»** или  . Эта команда позволяет добавить к ранее построенной детали (пластина, лежащая в плоскости **ZY**) тело выдавливания (пластина, лежащая в плоскости **XY**).

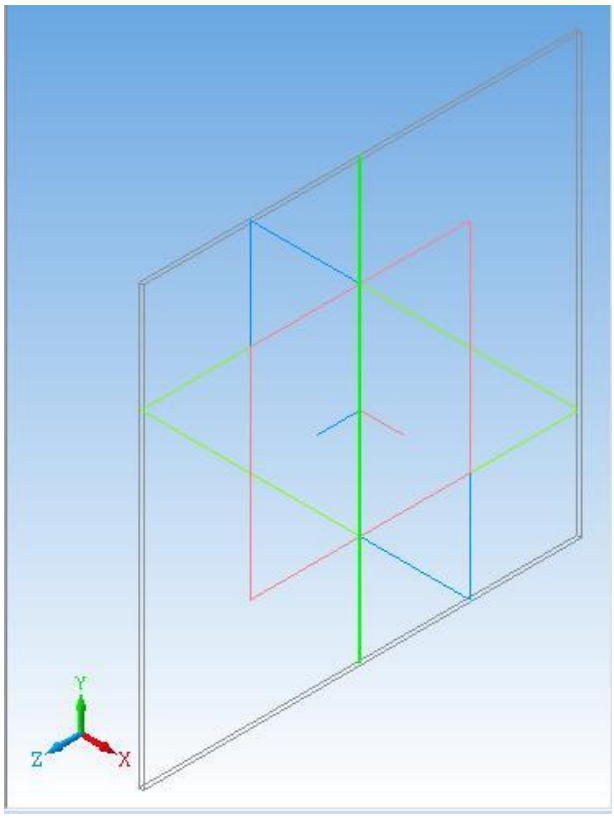


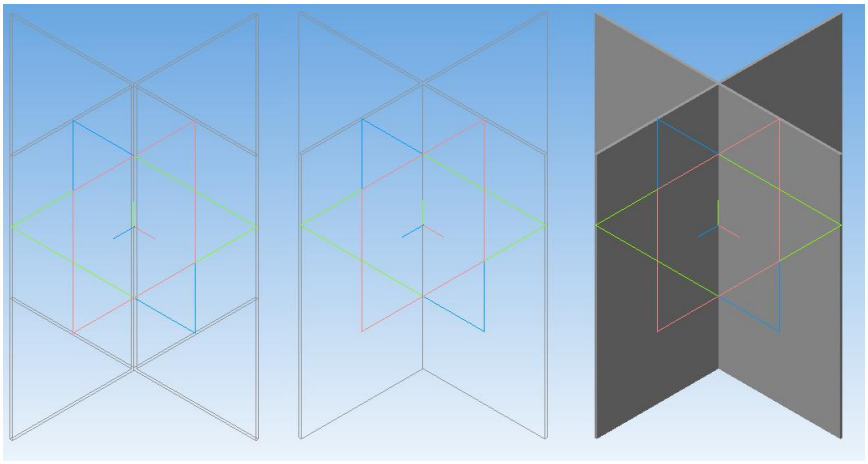
Рис. 1.2.19. Отрезок в плоскости ZY, «Изометрия XYZ»

Отрезок (рис. 1.2.19) лежит в плоскости **ZY**, следовательно, выдавливание будет происходить в направлении плоскости **XY**. Параметры опера-

ции **«Выдавливания»** выбираются на двух закладках окна диалога **«Панель свойств»:** параметры операции выдавливания и параметры тонкой стенки.

Система помнит установленные ранее параметры. Проверьте правильность установки: установлен флажок **«Средняя плоскость»** и задано **«Расстояние»** 100 мм. **«Уклон»** не установлен. **«Толщина»** тонкой стенки – 1 мм. Если все параметры установлены, то нажмите кнопку **«Создать объект»** .

Итак, построена тонкая квадратная пластина толщиной 1 мм, которая лежит в плоскости **XY** (рис. *1.2.*20). Вы можете выбрать различные варианты отображения модели и увидеть соответствующие изображения двух пластин.



Без невидимых Полутоновое

Рис. 1.2.20. Каркас

Окончание построения 3D-модели

Построить тонкую квадратную пластину, лежащую в плоскости **ZX** с толщиной 1 мм, сторона квадрата 100 мм.

1. В **«Дереве модели»** выберите **«Плоскость XY»**, установите текущую ориентацию «**Нормально к …»** (рис. 1.2.21).

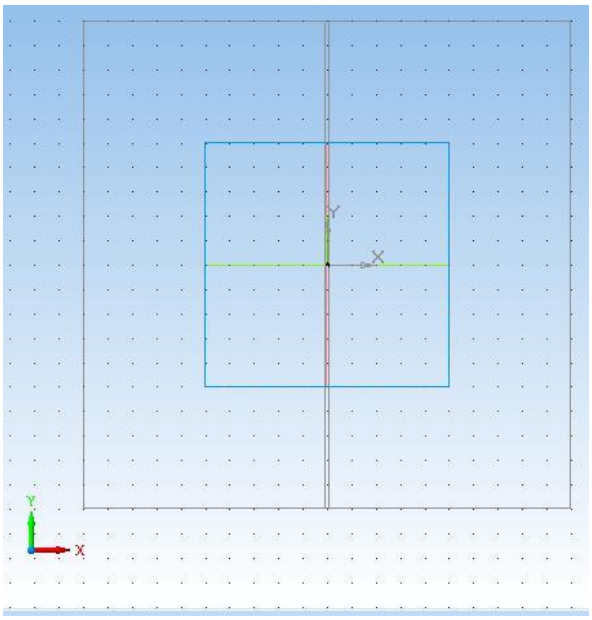


Рис. 1.2.21. Результат выполнения

2. Для построения эскиза нажмите кнопку команды **«Эскиз»** – .

3. Построим отрезок в с координатами (X = – 50, Y = 0, Z = 0) и (X = 50, Y = 0, Z = 0). Так как мы выбрали **«Плоскость XY»**, то для всех точек этой плоскости координата Z = 0.

4. Выберите команду **«Ввод отрезка»** и постройте отрезок с координатами начальной точки (X = – 50, Y = 0) и координатами конца отрезка – (X = 50, Y = 0). Нажмите кнопку **«Эскиз»** – . Отрезок в плоскости **XY** построен (рис. 1.2.22).

5. Выберите команду **«Операция Выдавливания».** Отрезок лежит в плоскости XY, следовательно, выдавливание будет происходить в направлении, перпендикулярном плоскости **XY**.

6. Проверьте правильность установки параметров выдавливания в окне диалога: установлен флажок **«Средняя плоскость»** и расстояние 100 мм. **«Толщина»** 1 мм. Нажмите кнопку **«Создать»**.

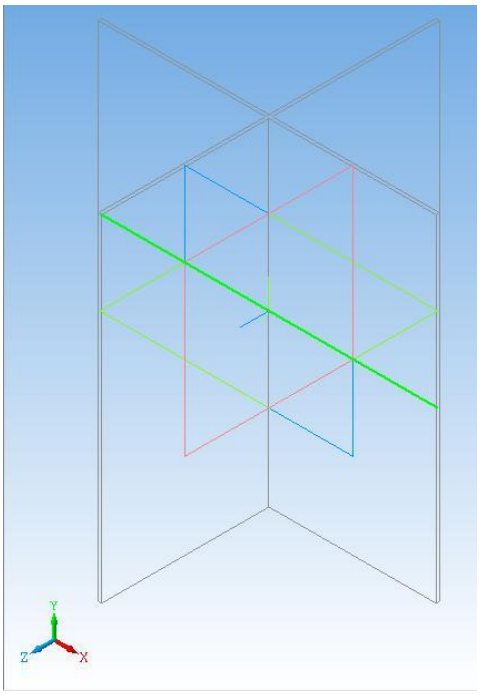


Рис. 1.2.22. Результат выполнения операции

Итак, мы построили тонкую квадратную пластину толщиной 1 мм, которая лежит в плоскости **XZ** (рис. *1.2.*23).

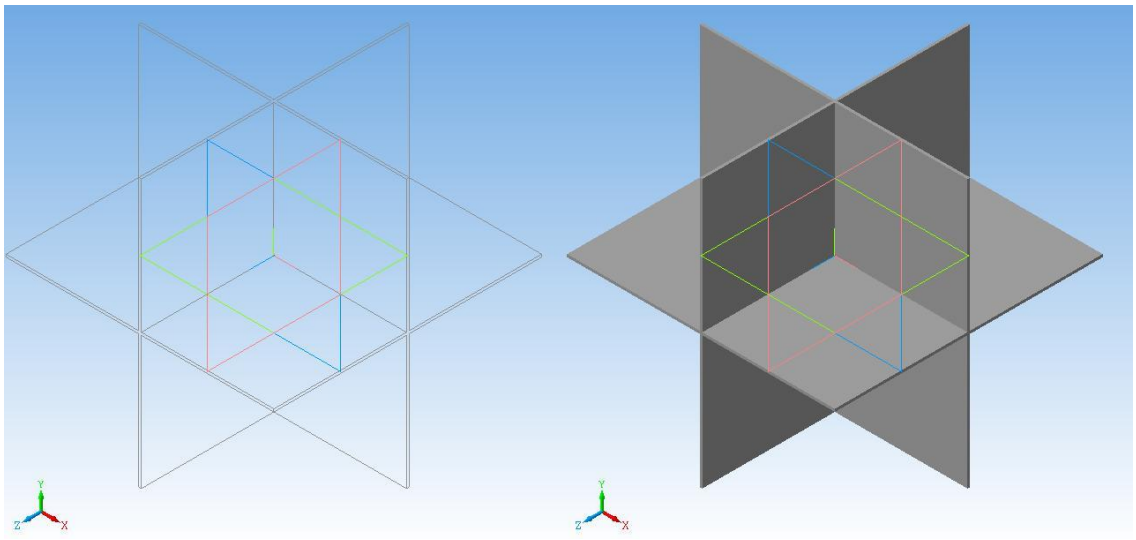


Рис. 1.2.23. Окончательный вид построенного 3D-объекта

1.3. Операция выдавливания

В данной лабораторной работе мы продолжаем знакомство с системой трехмерного твердотельного моделирования «КОМПАС 3D». В системах твердотельного трехмерного моделирования построение моделей производится с помощью последовательного создания трехмерных объектов и выполнения с ними типовых операций. Эти объекты создаются пользователем при построении модели. Создание трехмерных объектов связано с перемещением плоских фигур (эскизов) в пространстве. Эскиз может быть построен в одной из стандартных плоскостей (**XY, ZX, ZY)**, на плоской грани существующего тела или на дополнительной плоскости, которая построена пользователем. Эскизы создаются средствами плоского черчения.

Процесс создания трехмерной модели заключается в многократном добавлении или вычитании (вырезании) дополнительных трехмерных объектов, каждый из которых, образован при помощи трехмерных операций над плоским эскизом. Порядок применения и параметры операций отражаются в **«Дереве модели».**

Создание трехмерной модели аналогично созданию реальной детали. Действительно, сначала создается заготовка, затем от нее отсекается лишнее или добавляются необходимые части.

Операции, которые будем использовать в данной работе это **«Операция выдавливания»** и **«Вырезать выдавливанием»**.

Постановка задачи моделирования. Построить трехмерную модель цилиндра, диаметр основания 70 мм, высотой 80 мм. Цилиндр (рис. 1.3.1) имеет одно продольное и два поперечных сквозных отверстия диаметром 30 мм.

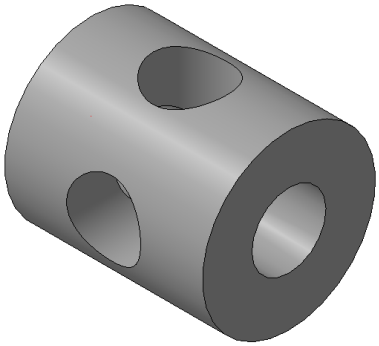


Рис. 1.3.2. Цилиндр

Порядок выполнения лабораторной работы. Первым шагом построения трехмерной модели является запуск подсистемы трехмерного моделирования, который производится при нажатии кнопки **«Создать…»** и кнопки **«Деталь»** на стартовой странице (рис. 1.3.2).

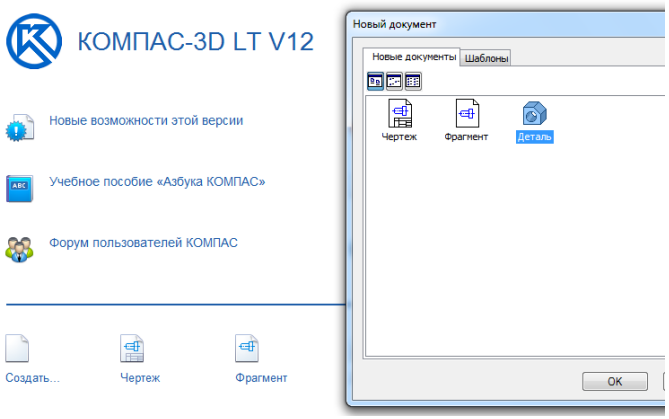


Рис. 1.3.2. Интерфейс системы «КОМПАС-3D» после запуска

В результате откроется окно построения трехмерной модели (рис. 1.3.3).

Первоначально познакомимся с функциональными кнопками подсистемы трехмерного моделирования. Обратите внимание на кнопку интерактивной справки , которая позволяет оперативно получить справку о любой команде.

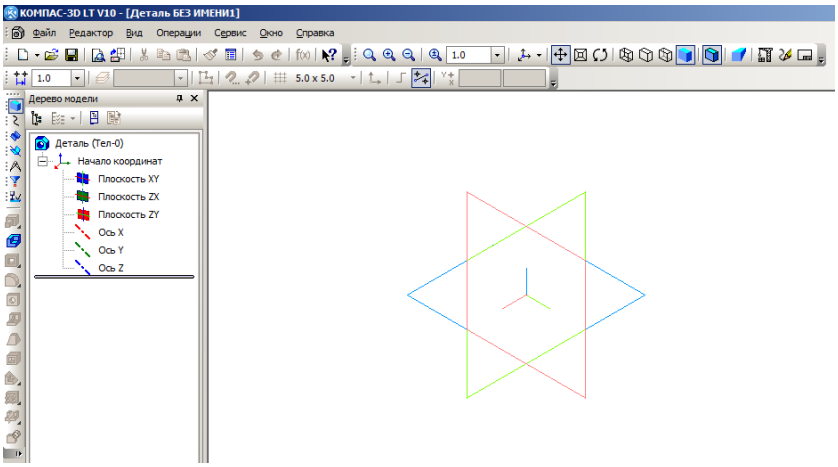


Рис. 1.3.3. Окно построения трехмерной модели (детали)

Первоначально, в **«Дереве модели»** (рис. 1.3.3) имеются система координат и три системные плоскости. Теперь следует выбрать одну из системных плоскостей (**XY, ZX, ZY)** для построения первого эскиза (рис. 1.3.4–1.3.6). Выберем плоскость **XY.**

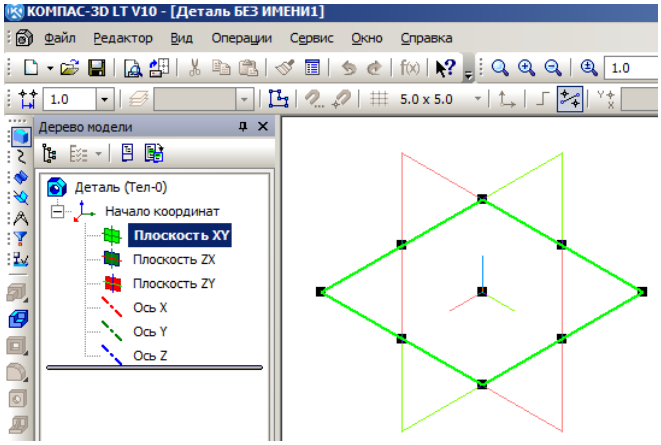


Рис. 1.3.4. Выделена плоскость XY

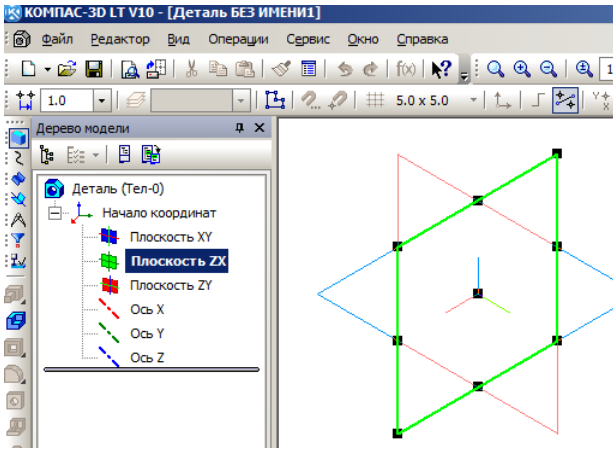


Рис. 1.3.5. Выделена плоскость ZX

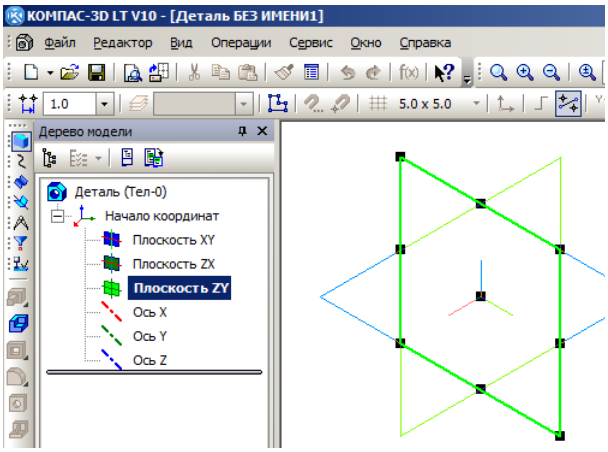


Рис. 1.3.6. Выделена плоскость ZY

Далее следует задать ориентацию выделенной плоскости (рис. 1.3.7). Выберем вариант **«Нормально к…»**. Новое положение выделенной плоскости представлено на рис. 1.3.8.

Теперь необходимо создать плоский эскиз. Для этого следует включить **режим редактирования эскиза** (рис. 1.3.9). Содержание интерфейса изменилось, стали доступны **операции построения плоского эскиза** (рис. 1.3.10). Необходимо включить режим отображения сетки (рис. 1.3.11). По умолчанию шаг сетки 5 мм.

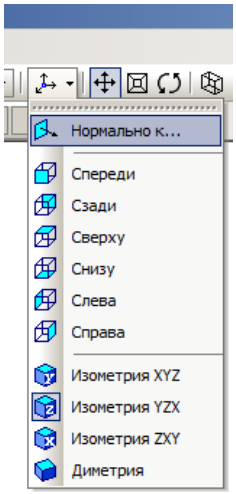


Рис. 1.3.7. Выбор ориентации плоскости эскиза

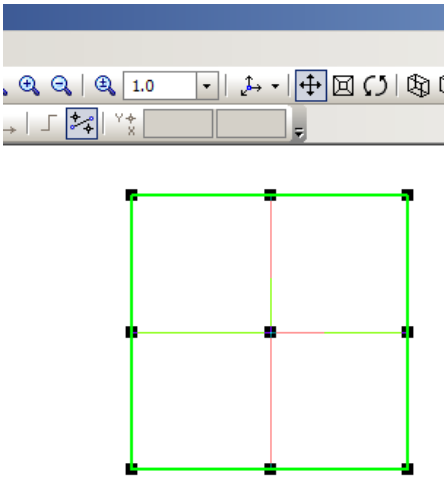


Рис. 1.3.8. Положение плоскости после выполнения команды «Нормально к…»

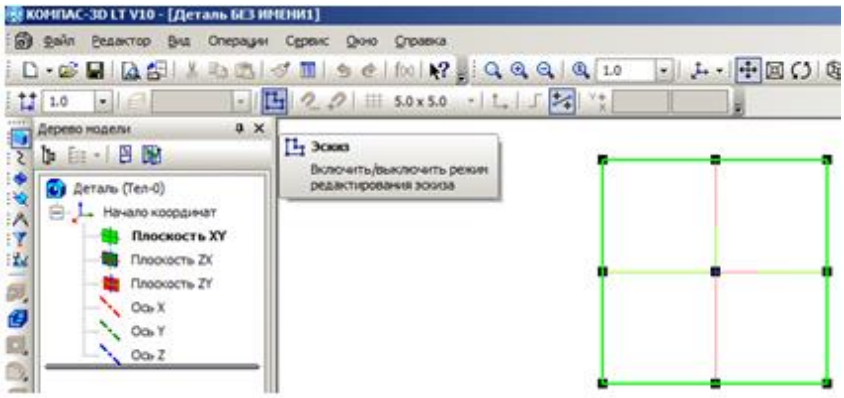


Рис. 1.3.9. Выполнение команды «Включить/выключить режим редактирования эскиза»

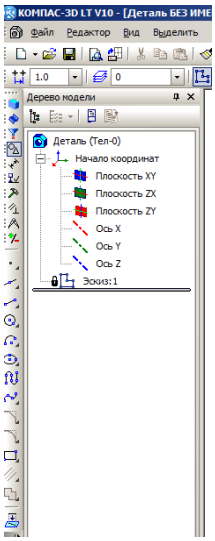


Рис. 1.3.10. Режим плоского черчения

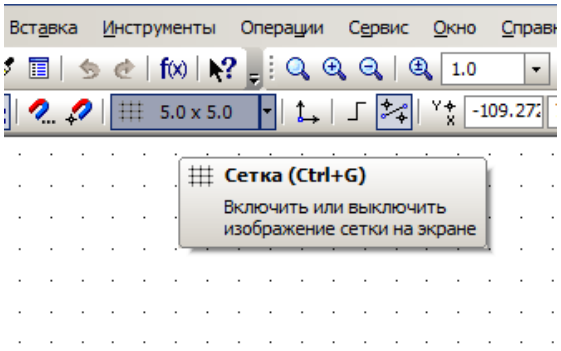


Рис. 1.3.11. Включение отображения сетки

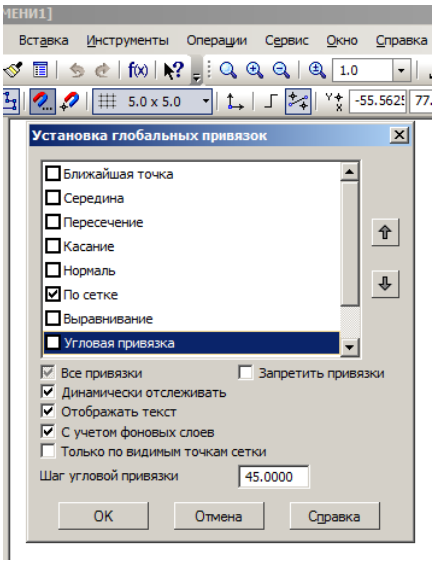


Рис. 1.3.12. Установка привязок

Далее для точного построения эскиза установим режим привязок (рис. 1.3.12) **«По сетке»**.

Построим первый эскиз. Это будет окружность – основание цилиндра диаметром **70 мм**. Выберем вариант построения: **«Центр окружности – точка на окружности»** (рис. 1.3.13). Центр окружности привяжем к началу координат. Эскиз представлен на рис. 1.3.14.

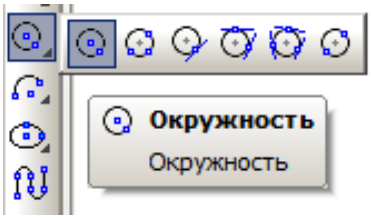


Рис. 1.3.13. Выбор способа построения окружности

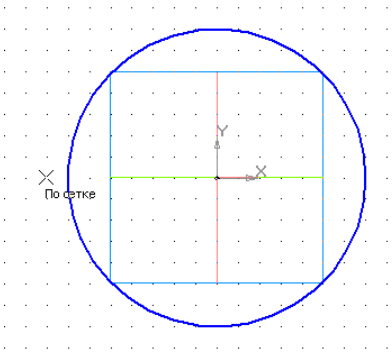


Рис. 1.3.14. Первый эскиз – окружность диаметром 70 мм

Закончим редактирование эскиза и зададим его ориентацию **«Изометрия XYZ»** (рис. 1.3.15). Выполним операцию **выдавливания** . Расстояние выдавливания 80 мм. На рис. 1.3.16 представлены параметры операции и ее фантом.

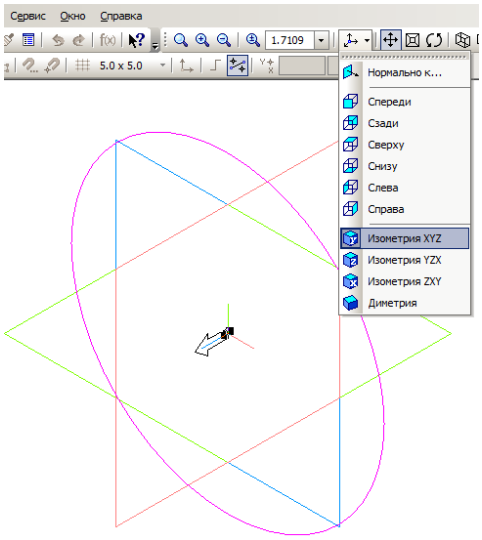


Рис. 1.3.15. Ориентация эскиза

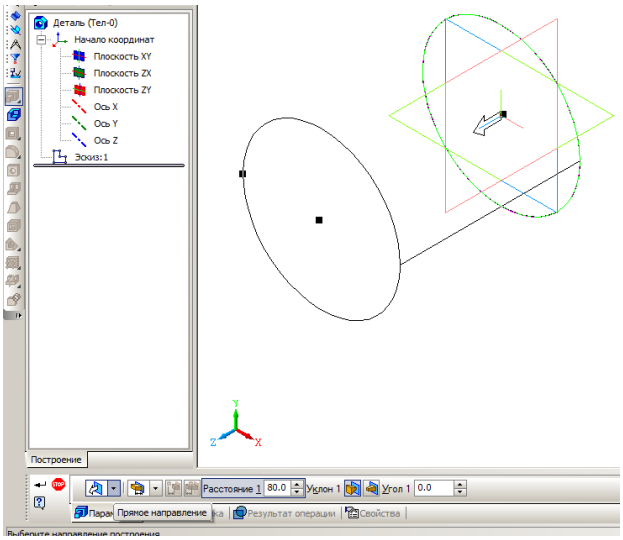


Рис. 1.3.16. Выполнение операции «Выдавливания»

Выполнение операции происходит при нажатии кнопки **«Создать»** . Результат показан на рис. 1.3.17.

Выполнение операции приведет к изменению дерева модели (рис. 1.3.18). Таким образом, заготовка для будущей детали построена. Продолжим создание детали. Теперь необходимо вырезать в детали продольное отверстие диаметром 30 мм.

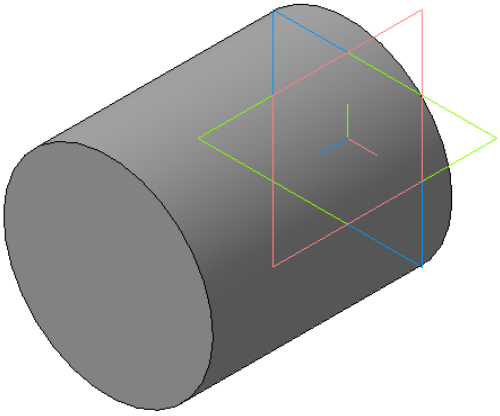


Рис. 1.3.17. Результат выполнения операции «Выдавливания»

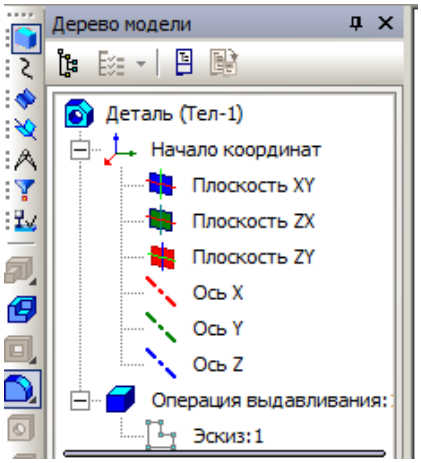


Рис. 1.3.18. Дерево модели

Выделим плоскость **XY** (рис 1.3.4)**.** Выберем ее ориентацию «**Нормально к…»** (рис. 1.3.7). Включим режим редактирования эскиза (рис. 1.3.9). Сетка и привязки по сетке уже установлены. Создадим новый эскиз (рис. 1.3.19); – окружность диаметром 30 мм.

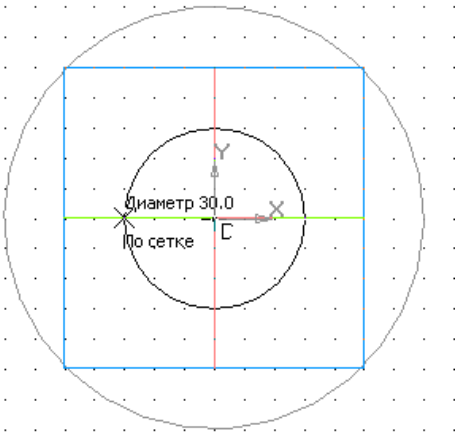


Рис. 1.3.19. Новый эскиз – окружность диаметром 30 мм

Завершим редактирование эскиза и зададим для него ориентацию **«Изометрия XYZ»** (рис. 1.3.15). Выполним операцию **«Вырезать выдавливанием»** (рис. 1.3.20 – 1.3.21). Обращаем внимание: необходимо правильно выбрать направление вырезания, иначе отверстие не будет вырезано. Расстояние вырезания следует задать несколько больше, чем длина детали. После выполнения операции деталь примет вид, представленный на рис. 1.3.22.

Вид делали на экране может быть задан в нескольких вариантах: **«Каркас»**, **«Каркас без невидимых линий»**, **«Невидимые линии тонкие»**, **«Полутоновое»** (рис. 1.3.22). Режимы переключаются кнопками  .

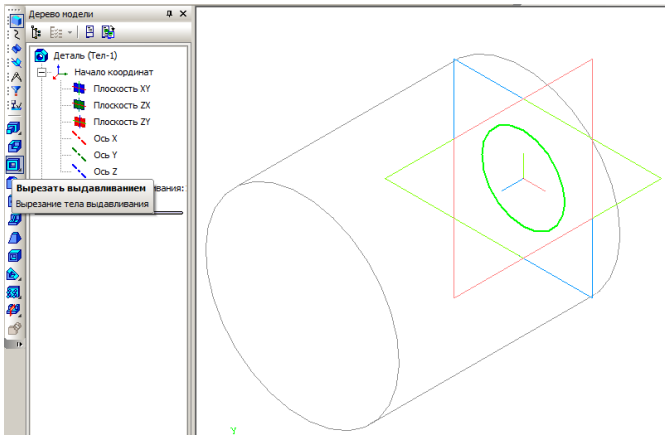


Рис. 1.3.20. Операция «Вырезать выдавливанием»

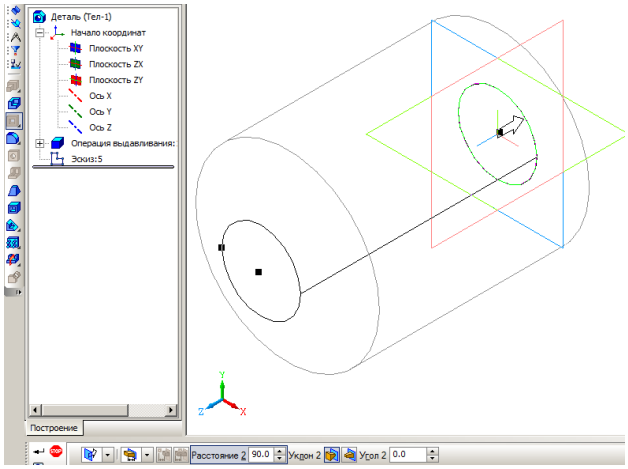


Рис. 1.3.21. Выполнение операции «Вырезать выдавливанием»

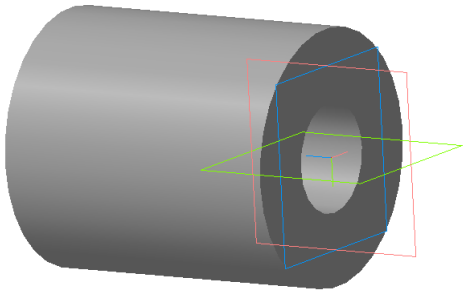


Рис. 1.3.22. Результат выполнения операции «Вырезать выдавливанием»

Теперь вырежем по центру детали поперечные отверстия диаметром 30 мм. Для построения эскиза следует выбрать плоскость **ZX**. При выполнении операции следует выбрать режим **«Два направления»** (рис. 1.3.23) и задать расстояния, достаточные для создания сквозного отверстия.

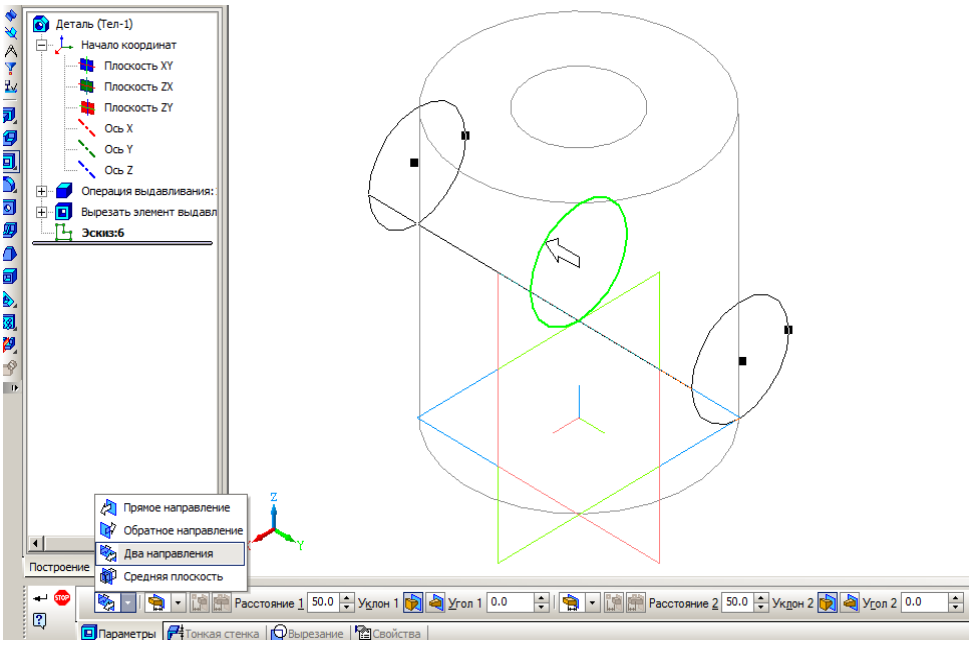


Рис. 1.3.23. Вырезание поперечного отверстия

Самостоятельно создайте второе поперечное отверстие с тем же диаметром (рис. 1.3.24). Отображение системных плоскостей и координатных осей можно скрыть, если в дереве модели в контекстном меню каждой плоскости задать соответствующий режим (рис. 1.3.25).

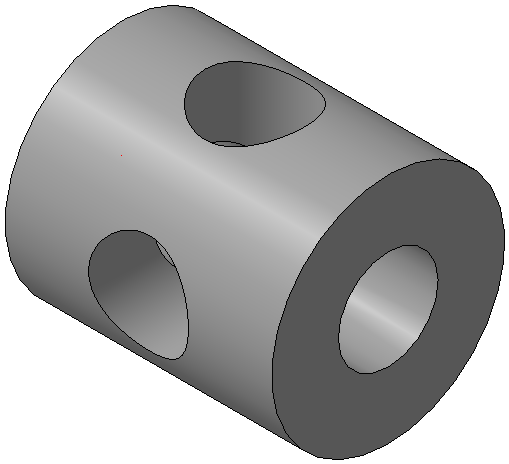


Рис. 1.3.24. Готовая деталь

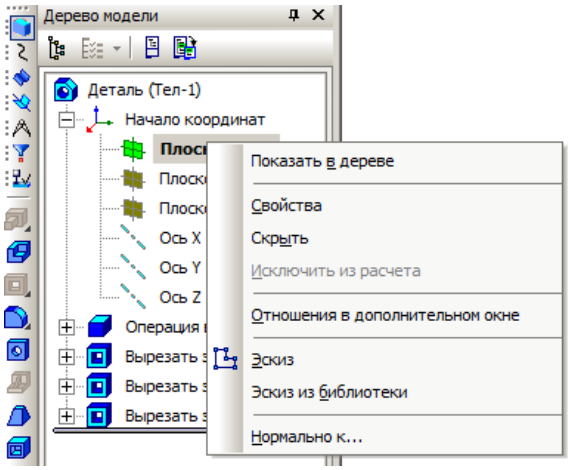


Рис. 1.3.25. Задание режима отображения системных плоскостей

Цвет и оптические свойства детали (рис. 1.3.24) установлен по умолчанию. Изменить эти параметры можно, выбрав в контекстном меню детали пункт **«Свойства»** (рис. 1.3.26).

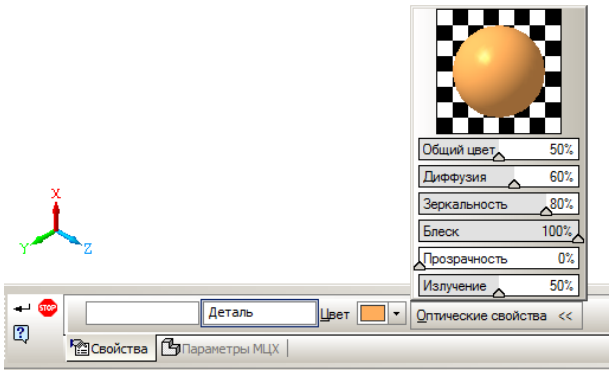


Рис. 26. Изменение цвета и оптических свойств детали

В заключение построим сечение тела плоскостью **ZX.** В дереве модели выделим плоскость **ZX**. Выполним операцию **«Сечение поверхностью»** (рис. 1.3.27). В результате применения операции деталь примет вид (рис. 1.3.28). На сечении ясно видно, что деталь является твердотельной моделью.

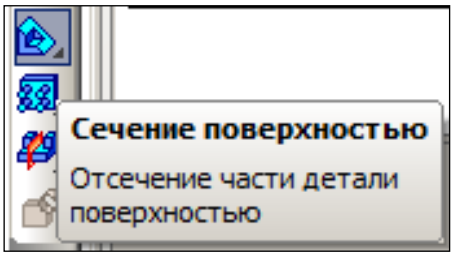


Рис. 1.3.27. Операция «Сечение поверхностью»

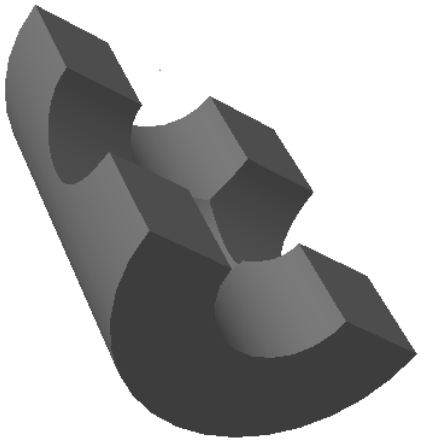


Рис. 1.3.28. Сечение детали плоскостью

1.4. Построение тел вращения

Рассмотрим технологию построения тел, которые можно получить с помощью операции **«Вращения»:** цилиндр, конус, шар, тор и т.п.

Задание 1. Построение цилиндра с помощью операции «Вращения»

Теперь построим цилиндр с помощью операции **«Вращения»**. Параметры цилиндра: высота 50 мм, диаметр основания 50 мм.

1. Создадим эскиз в плоскости **XY**, который содержит образующую цилиндра – отрезок прямой и ось вращения, проходящую через начало координат. Длина отрезка образующей 50 мм, тип линии **«Основная».** Ось вращения, тип линий – **«Осевая»**.

2. Отрезок образующей проведем через точки с координатами (X = 25, Y = 0) и (X = 25, Y = 50).

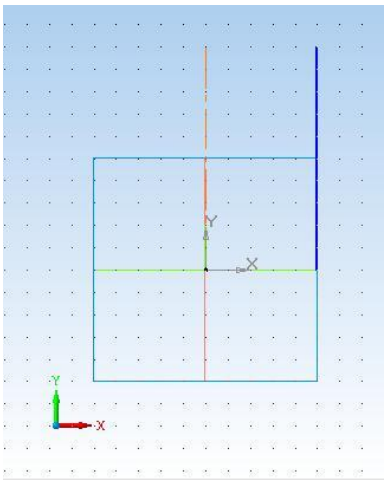


Рис. 1.4.1. Эскиз операции «вращение»

3. Ось вращения (не забудьте сменить тип линии) проведем через точки (X = 0, Y = 0) и (X = 0, Y = 50). Длина оси вращения может быть любой. Построенный эскиз будет иметь вид, рис. 1.4.1.

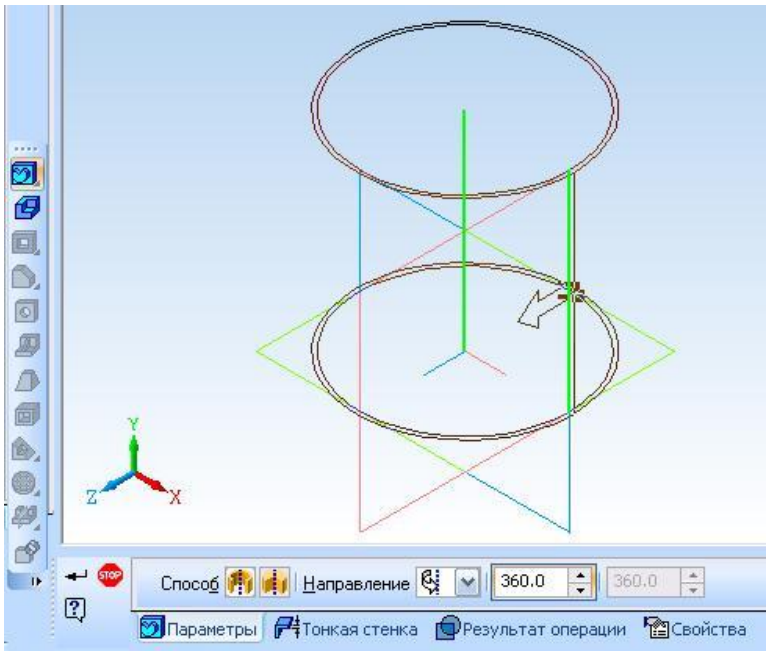


Рис. 1.4.2. Параметры операции «вращение»

4. Закончим редактирование эскиза. Выделим его и выполним команду операция **«Вращения»**  . После вызова **«Панели свойств»** на экране появляется окно параметров этой команды (рис. 1.4.2).

Операция **«Вращение»** может быть выполнена в режиме **«Тороид»,** при котором создается тонкостенная оболочка. В режиме **«Сфероид»** получается сплошной полнотелый элемент. На рис. 1.4.3 показаны результаты выполнения операции в двух режимах: **«Тороид» и «Сфероид».**

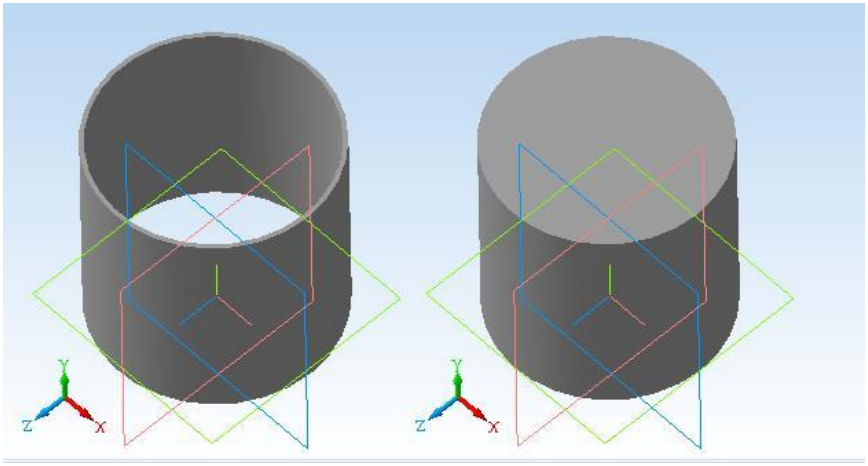


Рис. 1.4.3. «Тороид» «Сфероид»

Задание 2. Построить цилиндрический стакан

С помощью операции редактирования создать цилиндрический стакан с толщиной стенок 2 мм.

1. Используя **«Дерево модели»** из объектного меню выберем операцию **«Редактировать эскиз»** (рис. 1.4.4).

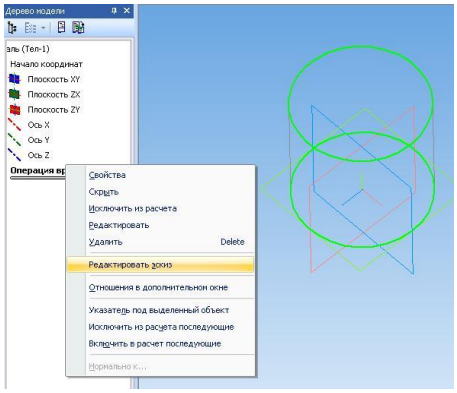


Рис. 1.4.4. Выбор операции редактирования эскиза

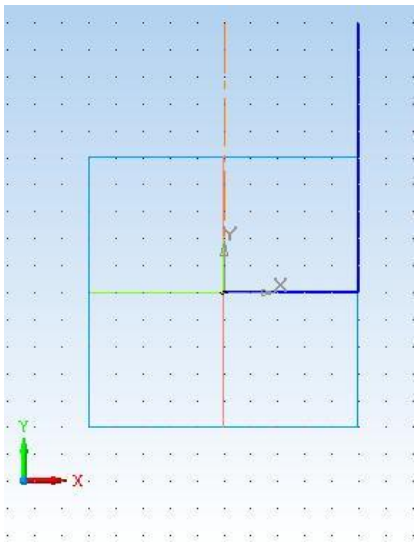


Рис. 1.4.5. Эскиз для построения стакана

2. Построим горизонтальный отрезок из начала координат до образующей цилиндра (рис. 1.4.5). Режим **«Привязка по сетке»** поможет избежать пересечения образующей с осью вращения, которое недопустимо.

3. Закончим редактирование эскиза. В **«Дереве модели»** выберем **«Редактировать»** (рис. 1.4.6).

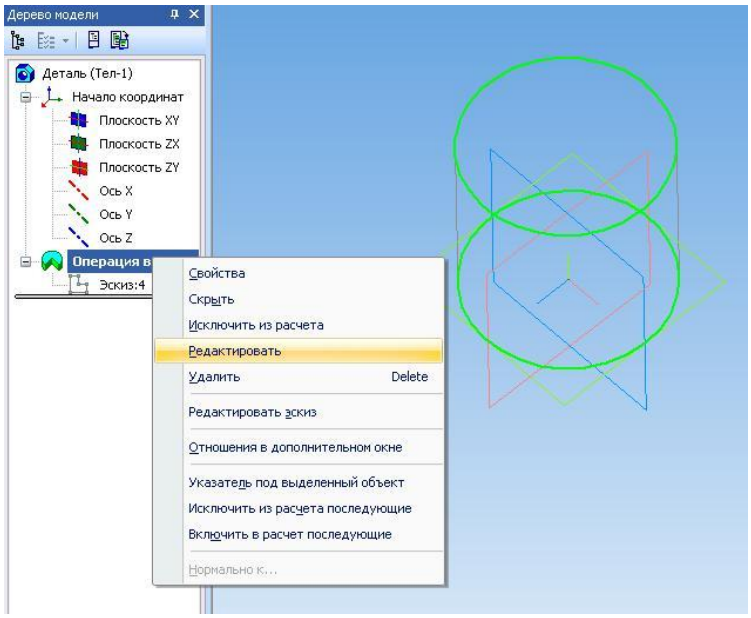


Рис. 1.4.6. Редактирование

4. В окне диалога операции вращения (рис. 1.4.2) установим режим **«Тороид»**, толщина стенок 2 мм. Цилиндрический стакан построен (рис. 1.4.7).

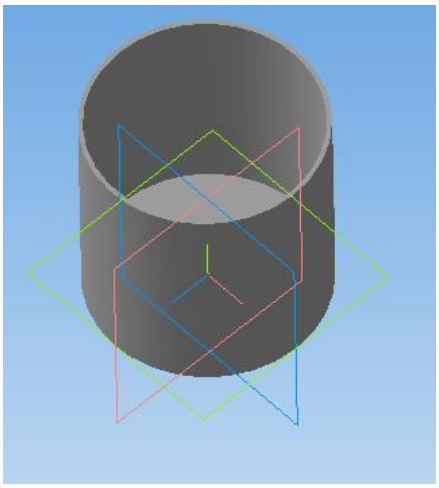


Рис. 1.4.7. Готовая деталь

Задание 3. Построение конуса

Построить конус. Радиус основания 35 мм, высота конуса 80 мм. Задание выполним с помощью операции вращения.

Пример эскиза на рис. 1.4.8. Первый отрезок (X = 0, Y = 0), (X = 35, Y = 0) – заготовка основания. Второй отрезок (X = 35, Y = 0), (X = 0, Y = 80) образующая конуса.

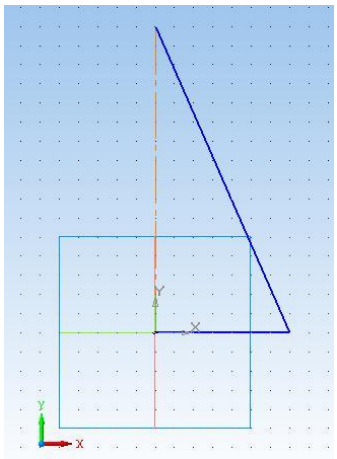


Рис. 1.4.8. Эскиз для создания конуса

После выполнения операции вращения в режиме **«Сфероид»** с отключенной тонкой стенкой получим конус (рис. 1.4.9).

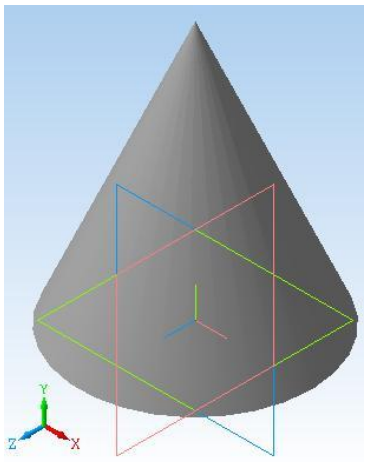


Рис. 1.4.9. Конус

Самостоятельно с помощью операции вращения создайте усеченный конус.

Задание 4. Построение тора

Создать тор, радиус образующей круговой траектории 80 мм, радиус образующей тора 15 мм. В результате выполнения задания вы можете получить пустотелый тор или тор из сплошного материала (рис. 1.4.10).

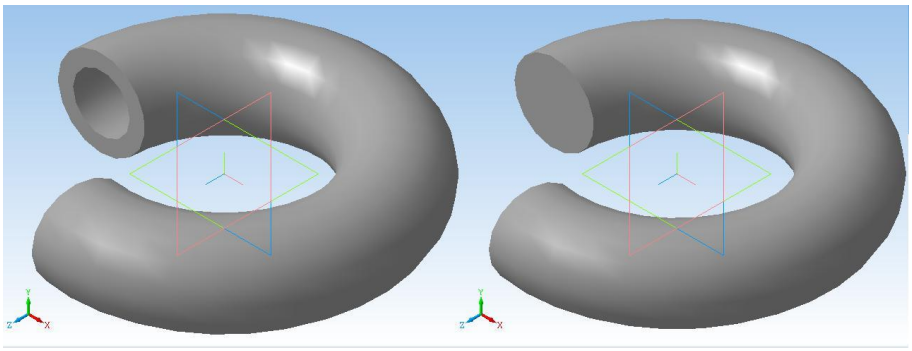


Рис. 1.4.10. Тор

Задание 5. Построение сферы

1. Постройте в плоскости **XY** эскиз, который должен включать полуокружность и ось вращения. Для оси вращения выбирается осевая линия. Центр полуокружности должен лежать на оси вращения. Радиус полуокружности 30 мм. Ось вращения и эскиз не должны пересекаться. Для построения полуокружности примените операцию **«Ввод дуги»** в режиме **«Привязка по сетке»**. Получите следующее изображение эскиза (рис. 1.4.11).

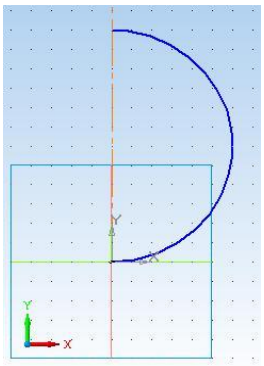


Рис. 1.4.11. Эскиз для создания сферы

2. Для того чтобы видеть результат выполнения операции выберите значение угла для прямого направление вращения 270^0. При выборе способа построения **«Сфероид»** вы получите шар, если выбрать **«Тонкая стенка»** 1 мм, то получим пустотелый шар (рис. 1.4.12).

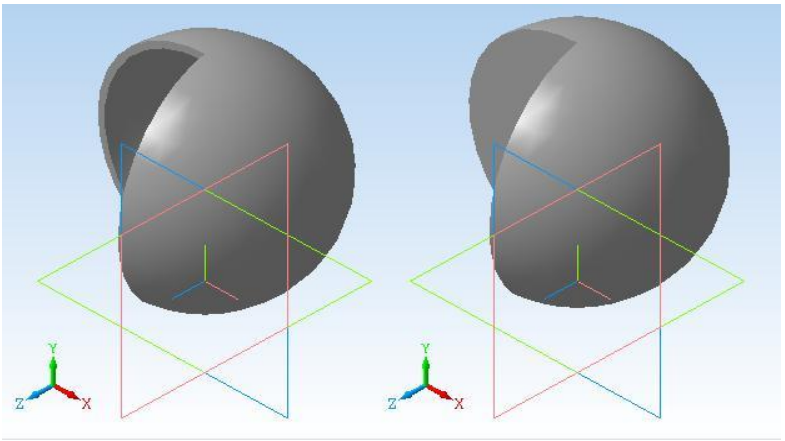


Рис. 1.4.12. Готовая деталь – сфера

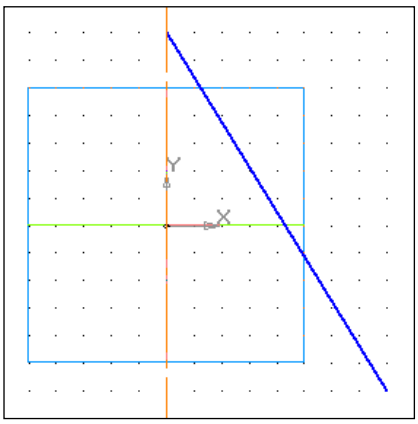


Рис. 1.4.13. Эскиз конуса

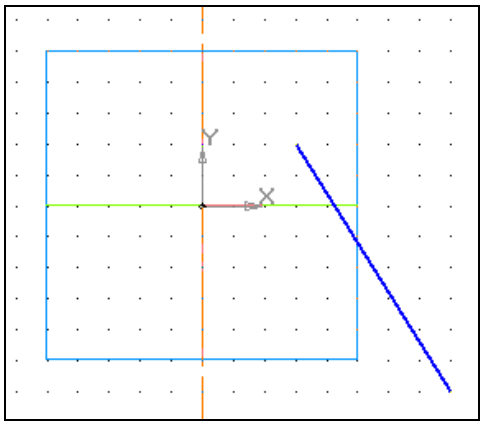


Рис. 1.4.14. Эскиз усеченного конуса

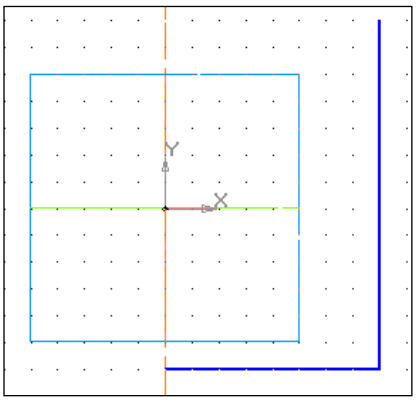


Рис. 1.4.15. Эскиз стакана

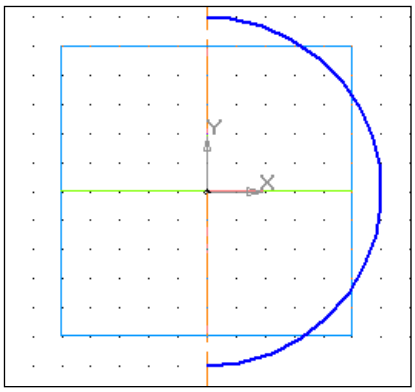


Рис. 1.4.16. Эскиз сферы

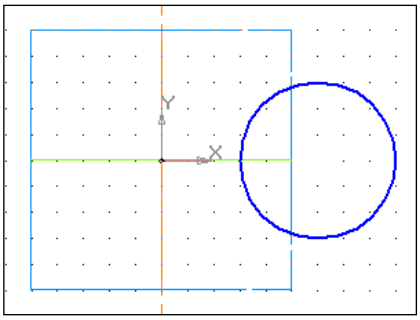


Рис. 1.4.17. Эскиз образующей тора

1.5. Построение 3D-модели по сечениям

Все представляют себе самую обычную лодку. Проектирование подобного рода объектов производится по сечениям (рис. 1.5.1).

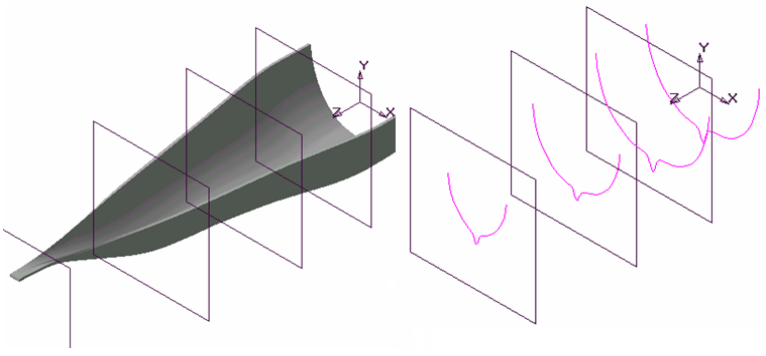


Рис. 1.5.1. Пример применения операции по сечениям

Сначала создают несколько поперечных сечений лодки, затем эти сечения располагают на расстоянии друг от друга. После этого сечения соединяют между собой в определенном порядке для того, чтобы получился корпус лодки. Конечно, эта схема упрощена, но все же позволяет понять принцип создания трехмерного объекта по сечениям.

В системе «КОМПАС 3D» имеется возможность создания плоскостей, смещенных относительно определенной плоскости трехмерного пространства, например, плоскости **XY**. Они носят название смещенная плоскость. Смещенные плоскости могут располагаться параллельно или под углом друг к другу. В каждой из таких плоскостей можно создать эскиз, по которым будет сформирована объемная модель. Формирование трехмерной модели происходит при объединении эскизов с помощью операции **«По сечениям»**  .

Порядок выполнения работы

Задание 1. Построение детали

1. Выполните команду **«Сервис»–«Параметры»–«Текущая деталь».**

2. Установим более контрастный цвет для смещенной плоскости. В списке параметров текущей детали в левой части окна выберите элемент **«Свойства объектов»** (рис. 1.5.2). В правой части окна откроется список, в котором перечислены все типы элементов трехмерной модели «КОМПАС 3D». Рядом с каждым названием элемента показана пиктограмма, соответствующая ему в **«Дереве модели».**

3. В правом списке укажите элемент **«Смещенная плоскость»** и нажмите кнопку **«Цвет»**.

4. В палитре основных цветов укажите любой яркий цвет, например, красный. Нажмите ОК и закройте данное окно.

5. Щелчком на кнопке **«Вспомогательная геометрия»**  раскройте одноименную страницу **«Инструментальной панели»** и нажмите кнопку **«Смещенная плоскость»**  .

6. В **«Дереве модели»** выберите элемент **«Плоскость XY».**

7. С помощью **«Панели свойств»** (**«Вид»–«Панели инструментов»–«Панель свойств»**, ) введите значение смещения 40 мм создаваемой плоскости относительно указанной.

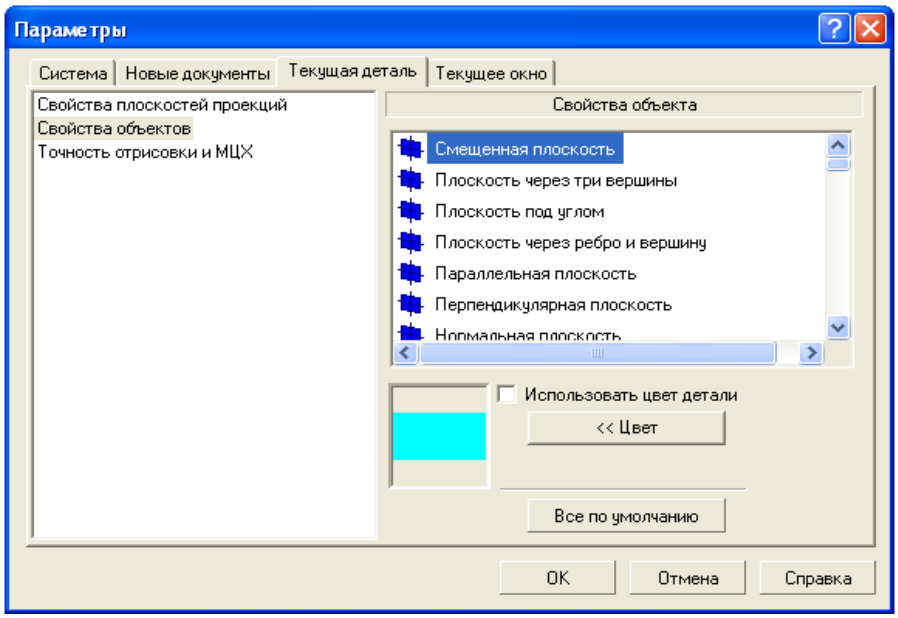


Рис. 1.5.2. Установка цвета смещенной плоскости

8. Для построения плоскости нажмите кнопку **«Создать объект»** на **«Панели свойств».** В **«Дереве модели»** появится новый элемент **«Смещенная плоскость: 1»,** а в окне модели изображение новой плоскости в виде прямоугольника (рис. 1.5.3).

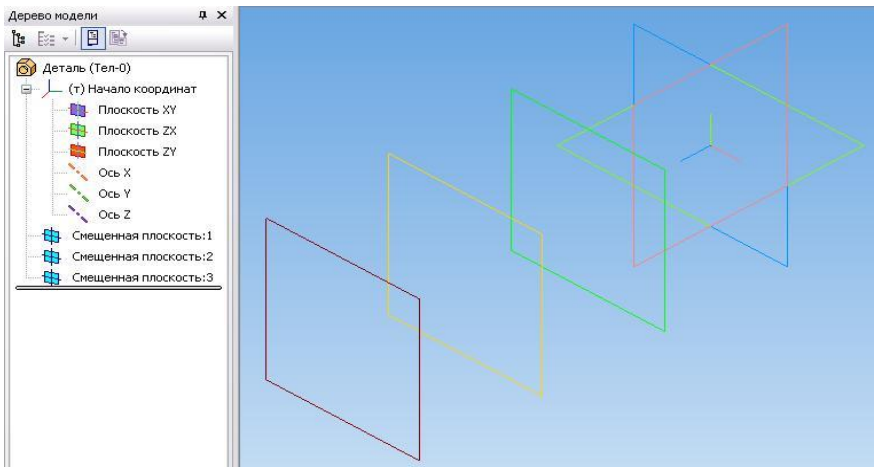


Рис. 1.5.3. Смещенная плоскость

9. Вновь укажите плоскость **XY** в **«Дереве модели».** В поле **«Смещение»** введите значение смещения для второй плоскости 40 мм и нажмите кнопку **«Создать объект»** . В результате в окне модели будут созданы две смещенные плоскости, положение которых задано относительно системной плоскости **XY**.

10. В качестве базовой можно указывать любую из имеющихся в модели плоскостей или граней. Например, укажите в **«Дереве модели»** элемент **«Смещенная плоскость: 2».**

В поле **«Смещение»** панели свойств введите значение смешения для третьей плоскости 40 мм и нажмите кнопку **«Создать объект»**  . Таким образом, положение третьей смещенной плоскости будет задано относительно второй смещенной плоскости (рис. 1.5.4).

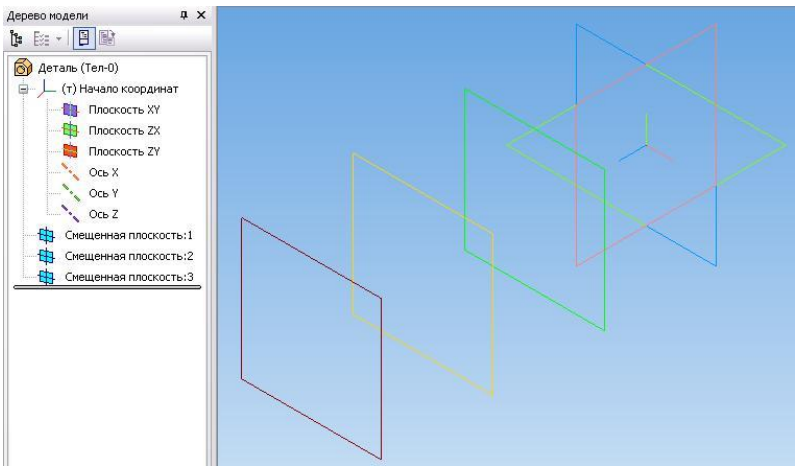


Рис. 1.5.4. Система смещенных плоскостей

11. Завершите  команду **«Смещенная плоскость»**.

12. Укажите элемент плоскость **XY** в **«Дереве модели»** и нажмите кнопку **«Эскиз»**. Ориентация **«Нормально к..»**. **«Сетка», «Привязки по сетке»**  **.**

13. Создайте в эскизе квадрат со стороной 60 мм и закройте его (рис. 1.5.5).

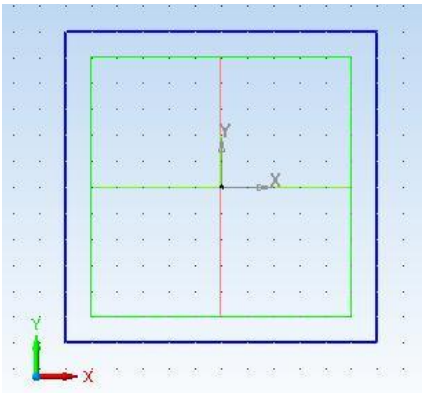


Рис. 1.5.5. Эскиз

14. Создайте новый эскиз на элементе **«Смещенная плоскость: 1»**. Нарисуйте в эскизе окружность диаметром 50 мм с центром в начале координат эскиза (рис. 1.5.6).

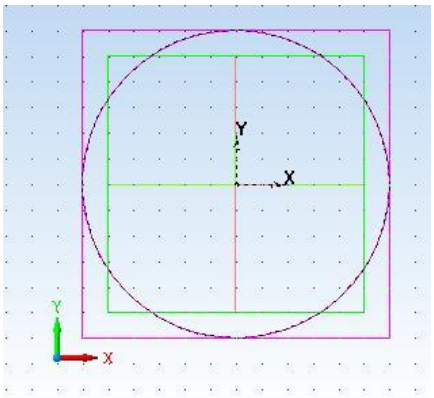


Рис. 1.5.6. Эскиз на смещенной плоскости: 1

15. Создайте новый эскиз на элементе **«Смещенная плоскость: 2»**. Нарисуйте в эскизе окружность диаметром 50 мм с центром в начале координат эскиза.

16. Создайте новый эскиз на **«Смещенная плоскость: 3»**. В текущем эскизе создайте окружность диаметром 70 мм.

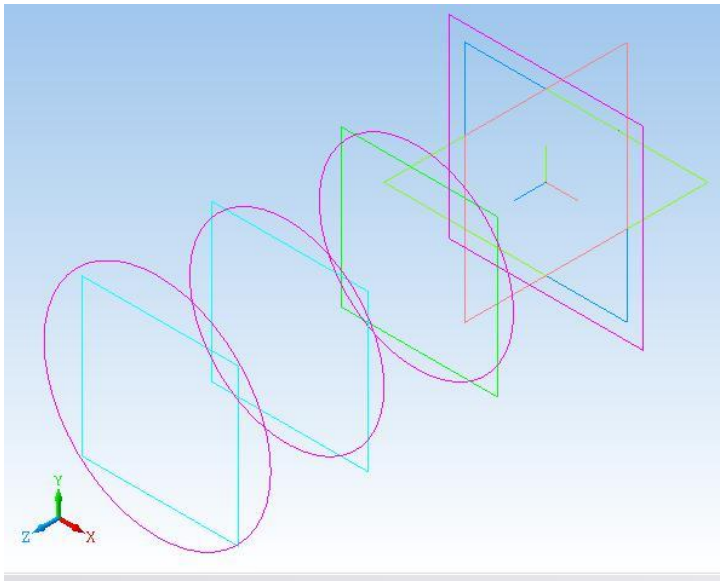


Рис. 1.5.7. Система эскизов

17. Закройте эскиз и установите ориентацию **«Изометрия XYZ»**. В окне модели вы увидите изображение четырех эскизов, каждый из которых размещен на своей плоскости (рис. 1.5.7).

18. Выберите в **«Дереве модели» «Деталь»** и выполните команду **«Операция по сечениям»**  **.** После вызова **«Панели свойств»** на экране появится диалог, в котором нужно задать параметры элемента по сечениям (рис. 1.5.7). По умолчанию включена кнопка **«Сечения»** в группе **«Объекты»**, команда находится в режиме указания сечений. Сечения требуется указывать в том порядке, в котором они следуют в элементе.

19. В **«Дереве модели»** последовательно укажите элементы **Эскиз: 1, Эскиз: 2, Эскиз: 3** и **Эскиз: 4.** По мере указания сечений в окне модели будет отображаться фантом будущего элемента (рис. 1.5.8). Сечения можно указывать прямо в окне модели, щелкая мышью по графическим объектам в эскизах.

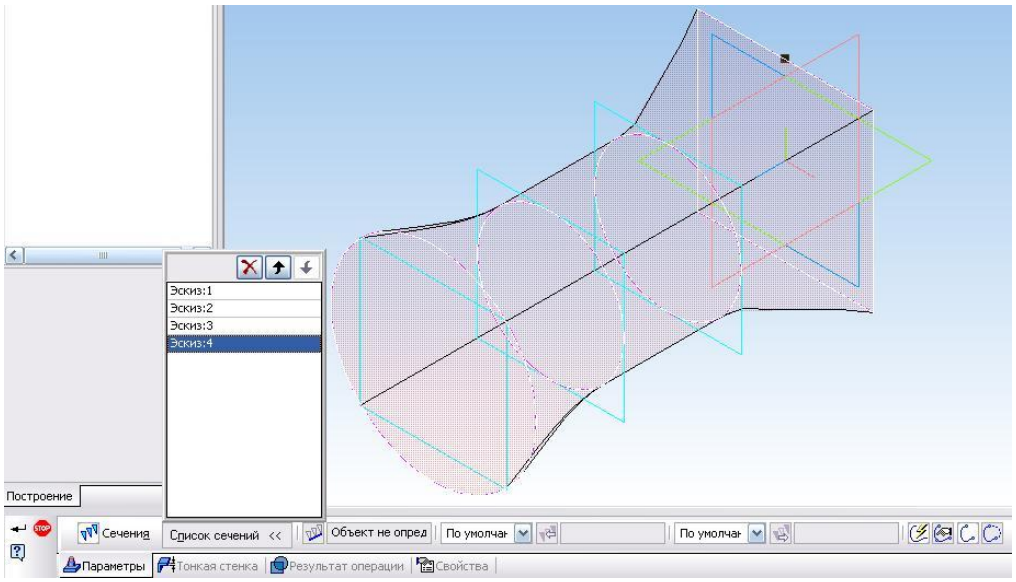


Рис. 1.5.8. Выполнение операции «По сечениям»

20. Нажмите кнопку **«Создать»** для выполнения операции построения детали (рис. 1.5.9).

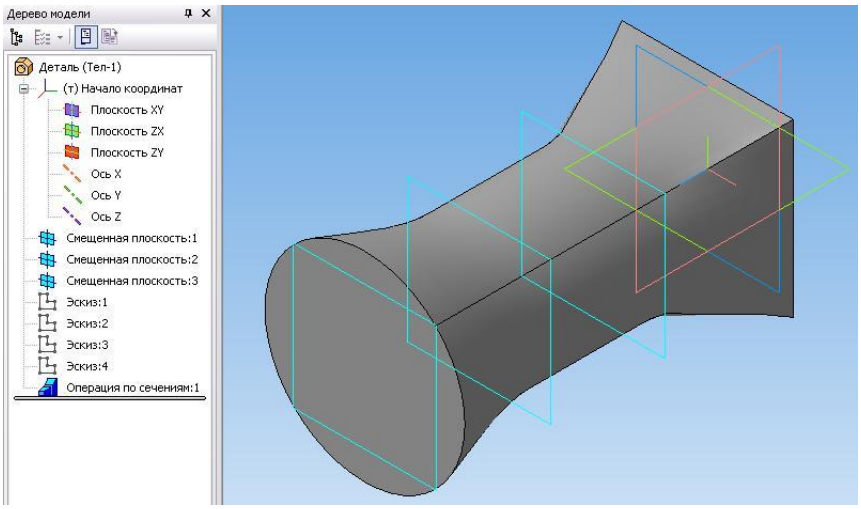


Рис. 1.5.9. Готовая деталь

1.6. Построение сечения тела плоскостью

Задание 1. Выполнить сечение тела системной плоскостью.

1. Предварительно постройте параллелепипед высотой 80 мм, основанием которого является правильный восьмиугольник. В основание параллелепипеда можно вписать окружность диаметром 60 мм (рис. 1.6.1). Тело строится с помощью операции **«Выдавливания»** в режиме **«Создать тонкую стенку»** толщиной 2 мм. Подробная схема построения тела показана в **«Дереве построения»** (рис. 1.6.1).

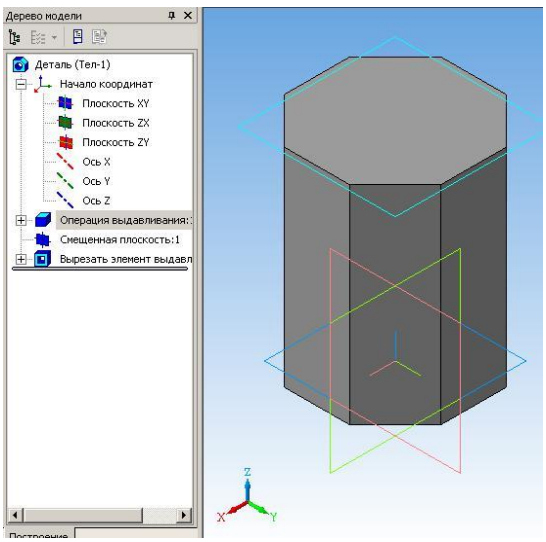


Рис. 1.6.1. Заготовка для выполнения операций

2. Выполним сечение тела плоскостью ZX. Для этого выделим в **«Дереве построения»** элемент **«Плоскость ZX»** (рис. 1.6.2.).

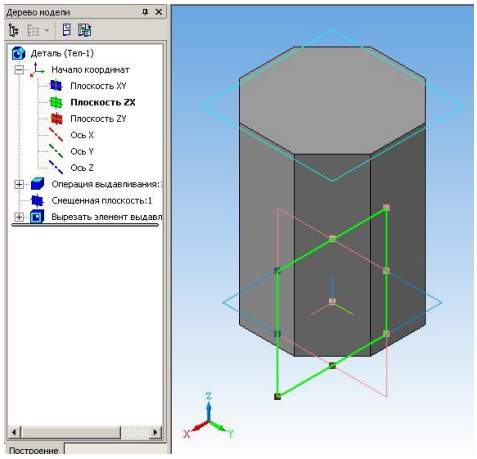


*Рис. 1.6.2. Выделена плоскость **ZX***

3. Выполните команду **«Операции Сечение Поверхностью»**  . Параметры операции задайте по рис. 1.6.3.

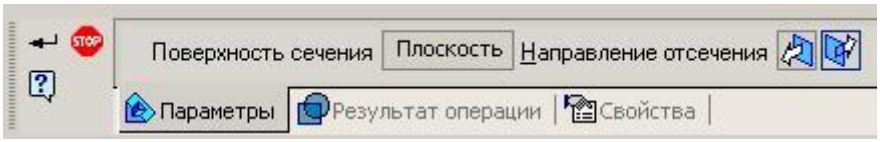


*Рис. 1.6.3. Параметры операции **Сечение поверхностью***

4. Выполнение команды **«Создать»** даст следующий результат (рис. 1.6.4).

5. Далее по аналогичному сценарию выполните операцию сечения **плоскостью ZY** (рис. 1.6.5).

6. Если в **«Дереве построения»** удалить элементы **«Сечение поверхностью: 1»** и **«Сечение поверхностью: 2»**, то тело вернется в исходное состояние (рис. 1.6.1).

Задание 2. Выполнить сечение тела плоскостью.

1. Выполните команду **«Операция Плоскость через три вершины»**.

2. Затем последовательно привяжитесь к трем вершинам тела на рис. 1.6.6.

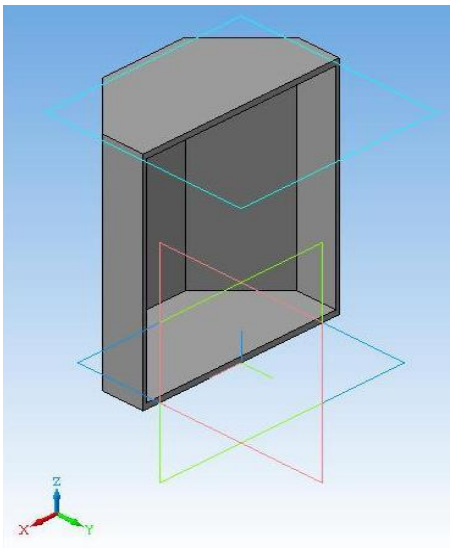


Рис. 1.6.4. Сечение плоскостью ZX

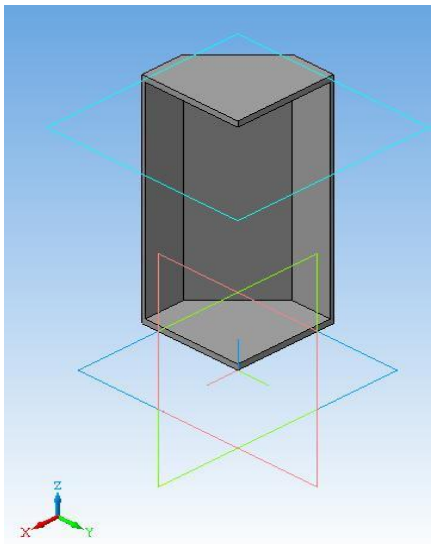


Рис. 1.6.5. Сечение плоскостью ZY

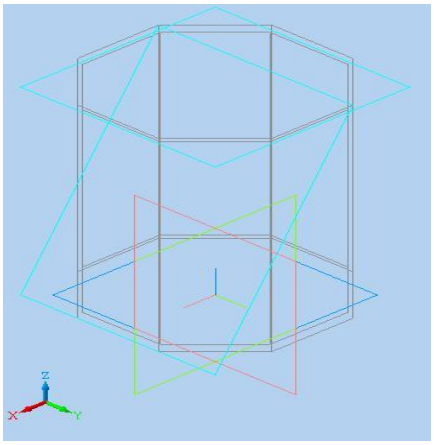


Рис. 1.6.6. Построение плоскости через три вершины

3. Выполните операцию **«Сечение поверхностью»** –  , при этом параметры операции задайте по рис. 1.6.7.

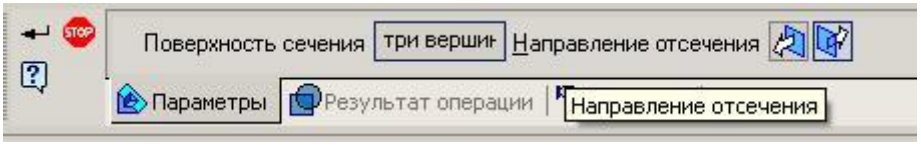


Рис. 1.6.7. Параметры операции «сечение тела плоскостью»

4. Выполнение команды **«Создать»** даст следующий результат (рис. 1.6.8).

5. Путем удаления из **«Дерева построения»** элемента **«Сечение поверхностью»** восстановите исходное состояние тела и выполните сечение плоскостью через три других вершины.

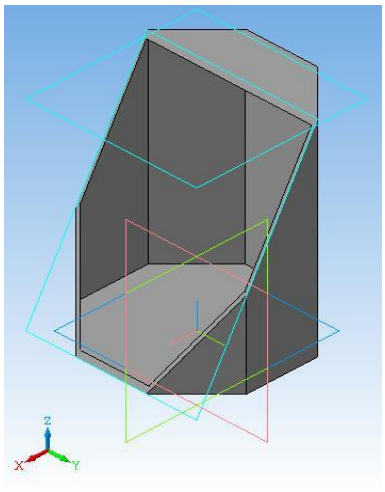


Рис. 1.6.8. Результат операции

6. Путем удаления элементов из **«Дерева построения»** восстановите исходное состояние тела (рис. 1.6.1).

7. В **«Дереве построения»** выделите **плоскость XY**, ориентация – **«Снизу»**.

8. Выделите мышкой грань тела, параллельную плоскости экрана (рис. 1.6.9).

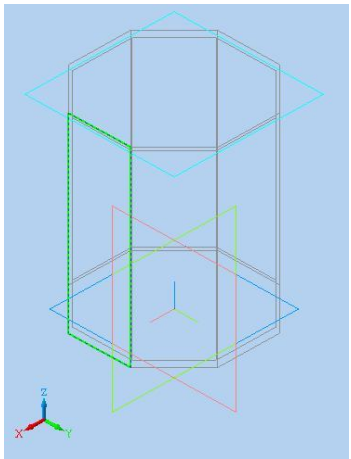


Рис. 1.6.9. Выделение грани

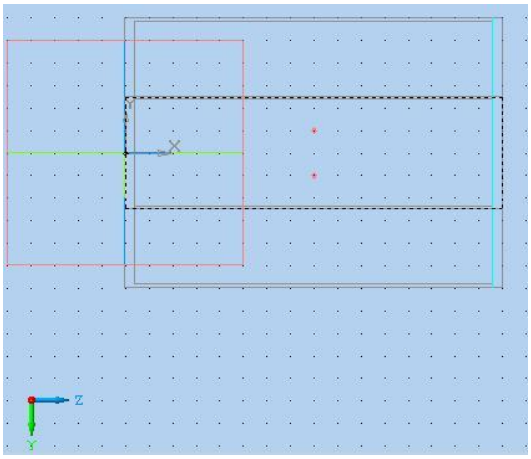


Рис. 1.6.10. Построение нового эскиза

9. Выполните команду **«Эскиз»** –  (рис. 1.6.10). В эскизе проставьте две точки (команда **«Ввод точки»** – ) с координатами (40; 5), (40; – 5). Выполнение эскиза следует проводить в режиме **«Привязка по сетке»**.

10. Постройте смещенную плоскость **«По трем вершинам»** (рис. 1.6.11; 1.6.12) и выполните ею сечение тела.

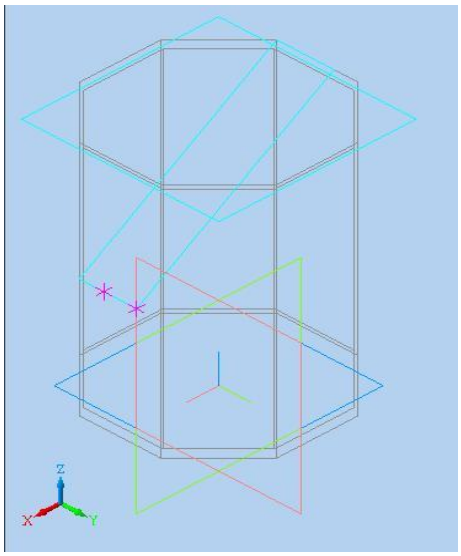


Рис. 1.6.11. Построение плоскости

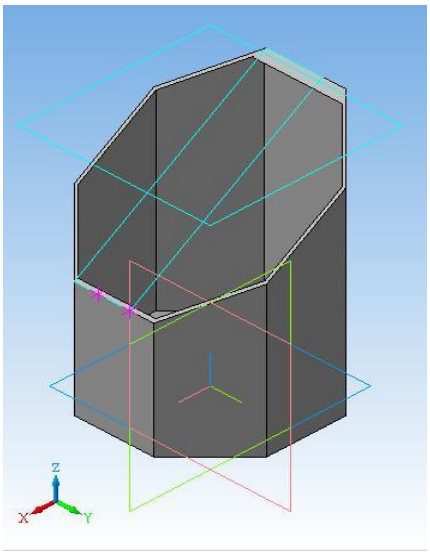


Рис. 1.6.12. Результат выполнения операции

САМОСТОЯТЕЛЬНАЯ РАБОТА

1.7. Создание модели вала

1. С помощью операции вращения создайте вал по рис. 1.7.1 и рис. 1.7.2.

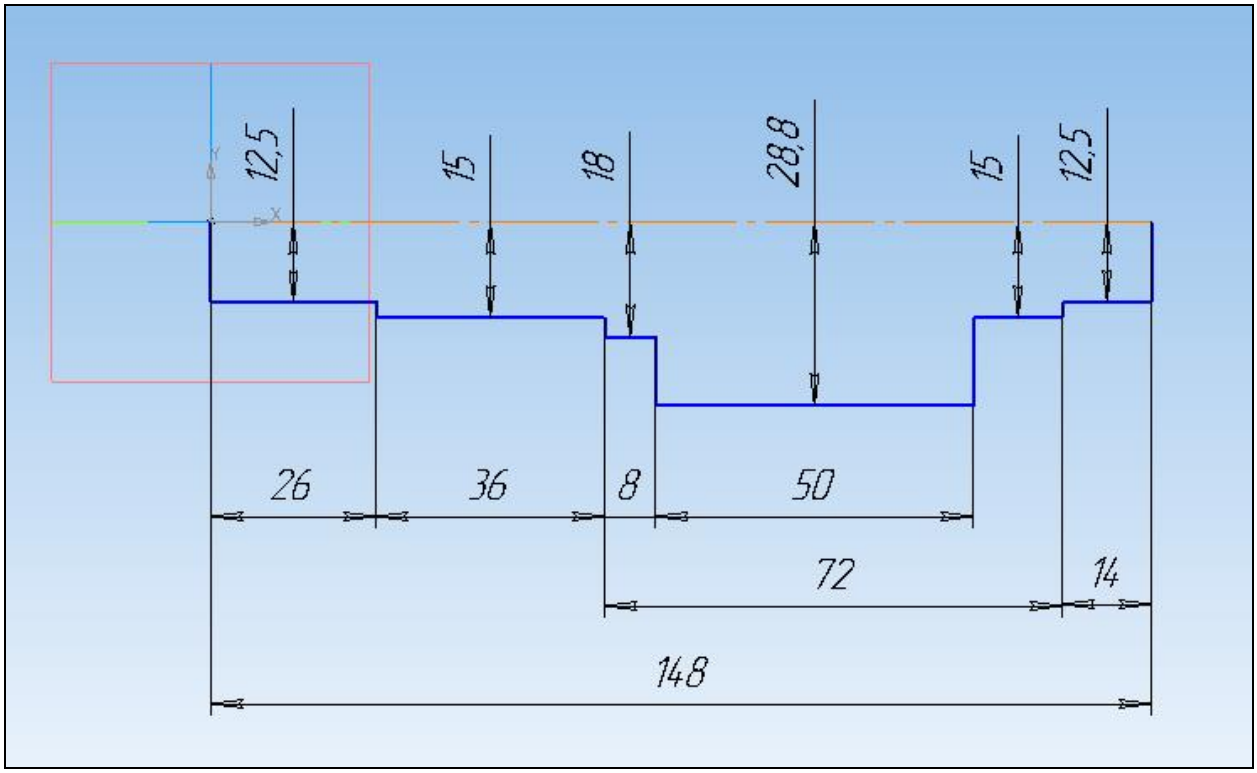


Рис. 1.7.1. Эскиз вала

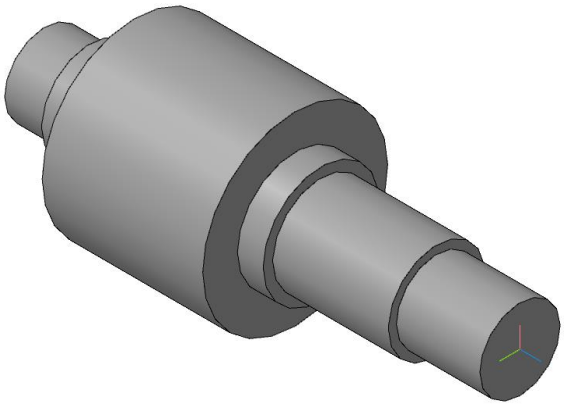


Рис. 1.7.2. 3D-модель вала в промежуточном состоянии

2. Создадим шпоночный паз. Для этого создадим вспомогательную плоскость.

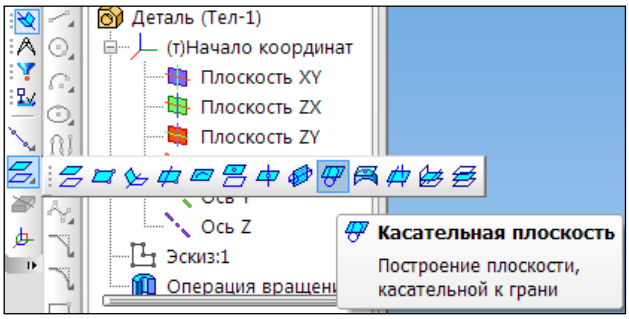


Рис. 1.7.3. Создание вспомогательноцй плоскости

Указываем поверхность, на которой будем производить построение. К цилиндрической поверхности можно построить бесконечное количество касательных плоскостей, поэтому нужно дополнительно указать плоскость, которая проходит через ось цилиндрической грани и показывает линию касания. В дереве модели укажите плоскость **ZX** (рис. 1.7.4).

3. Для построенной плоскости из контекстном меню выбираем **«Эскиз из библиотеки»** (рис. 1.7.5).

4. В Дереве библиотеки откройте папку **«Пазы и бобышки»**.

5. В списке элементов папки укажите **«Паз-1»**. В окне предварительного просмотра будет показан его контур.

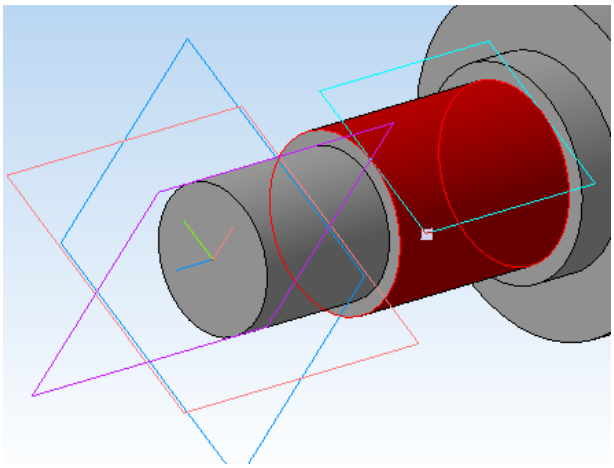


Рис. 1.7.4. Вспомогательная плоскость

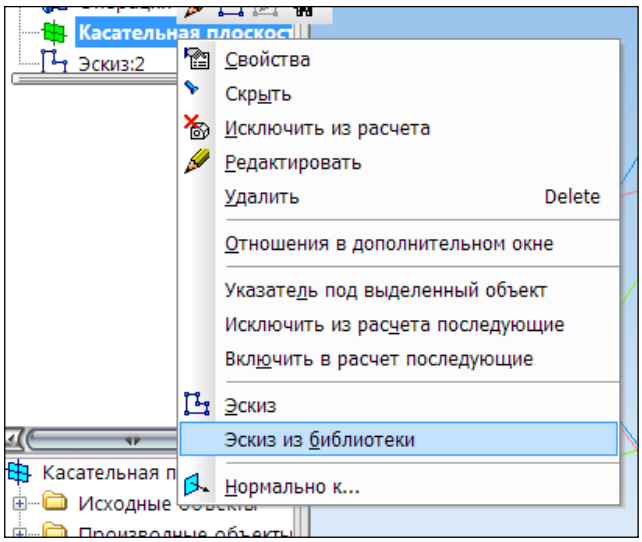


Рис. 1.7.5. Эскиз из библиотеки

6. Нажмите кнопку **«Создать»** объект

7. Отредактируйте **«Эскиз: 2»** (рис. 1.7.6)

8. Закройте эскиз и примените к нему операцию **«Вырезать выдавливанием»** в **прямом направлении** на **расстояние** 4 мм.

9. Скруглите дно паза радиусом 0,25 мм. Укажите саму грань – система автоматически определит все принадлежащие ей ребра.

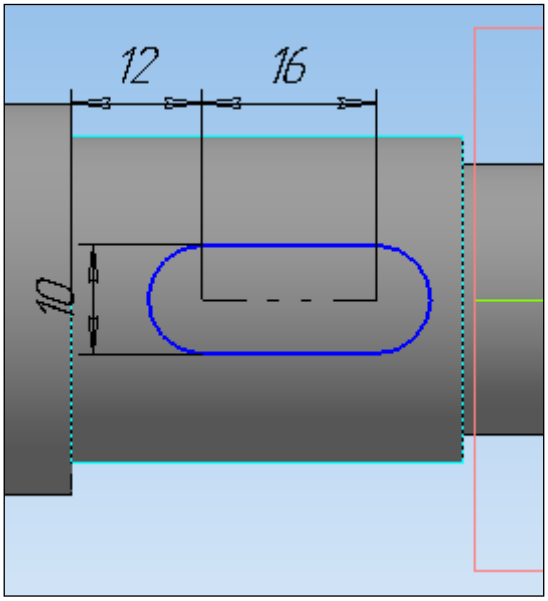


Рис. 1.7.6. Редактированный эскиз

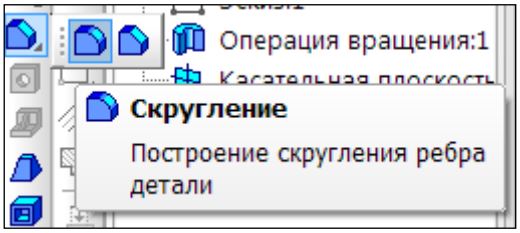


Рис. 1.7.7. Скругление

Создание центровочных отверстий

1. Укажите плоскую грань на торце детали.

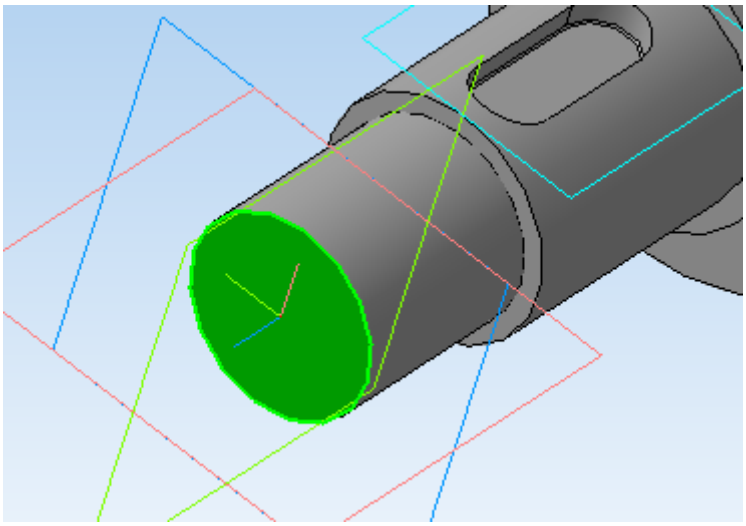


Рис. 1.7.8. Торец детали

2. Нажмите кнопку **«Отверстие»** на панели **«Редактирование детали»**.

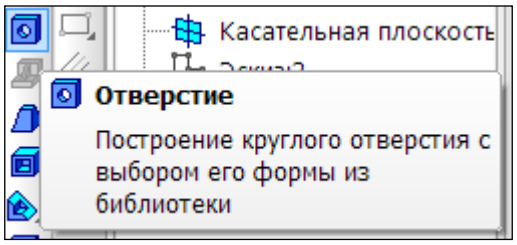


Рис. 1.7.9. Библиотека отверстий

3. В окне **«Библиотеки отверстий»** откройте папку **«Центровые отверстия»** и укажите отверстие **«Форма А»**.

4. В таблице параметров задайте диаметр отверстия **d 4 мм**, глубину конической части 13,9 мм и глубину цилиндрического участка 5 мм.

5. Нажмите кнопку **«Создать»** объект .

6. Повторите построение центрового отверстия на противоположном торце вала (рис. 10).

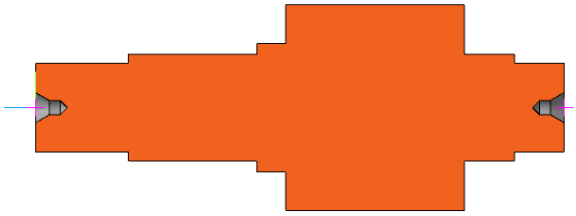


Рис. 1.7.10. Вал с центровочными отверстиями

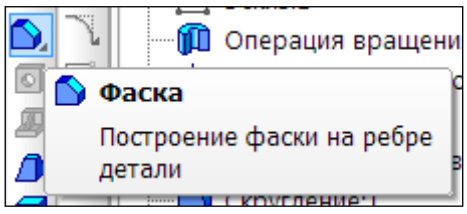


Рис. 1.7.11. Выбор фаски

1. На четырех круглых ребрах постройте фаски (рис. 1.7.11) длиной 2 мм под углом 45 градусов (желтые стрелки на рис. 1.7.12).

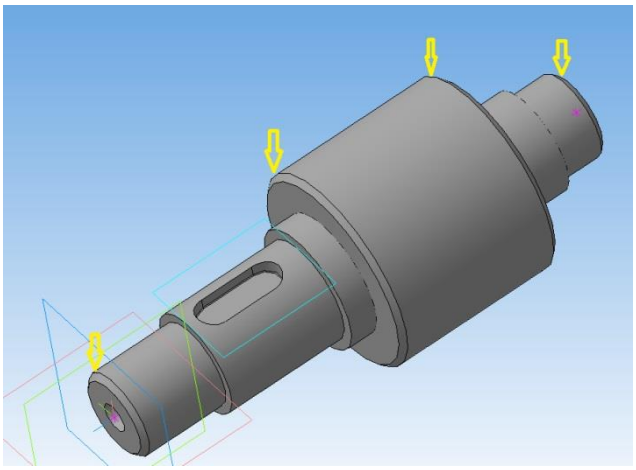


Рис. 1.7.12. Готовый вал с фасками

1.8. Создание сплайновых поверхностей

В данной работе мы познакомимся со способами моделирования поверхностей сложной формы, которые впервые потребовались для математического описания формы новых автомобилей, самолетов, космических кораблей, многочисленных бытовых приборов при их проектировании. В этой работе познакомимся с новым термином: **«сплайн»**.

Термин **«сплайн»** происходит от английского слова **«spline»**. Так называется гибкая полоса стали (рис. 1.8.1), при помощи которой чертежники проводили через заданные точки плавные кривые. Раньше подобный способ построения плавных обводов различных тел, таких как, например, корпус корабля, кузов автомобиля был довольно широко распространен в практике чертежно-графических работ в машиностроении. Появление компьютеров позволило перейти от этого механического метода к более эффективному математическому способу задания поверхности тела.

Рис. 1.8.1. Сплайновая кривая

Сплайн – это гладкая кривая, которая строится с использованием дуг (многочлены 3-й степени) и проходит через несколько контрольных точек, управляющих формой сплайна.

В основе этого подхода к описанию поверхностей лежит использование сравнительно несложных формул (многочлен 3-й степени), позволяющих восстанавливать облик изделия с необходимой точностью. Для большинства тел, встречающихся на практике, невозможно найти универсальную формулу, которая может описать соответствующую поверхность глобально, или, как принято говорить, в целом. Вместе с тем аналитическое описание (посредством формул) внешних обводов изделия должно быть достаточно экономным. Это особенно важно, когда речь идет об обработке изделий на станках с числовым программным управлением.

Построение кривой Безье

1. В системе «КОМПАС 3D» создадим фрагмент.

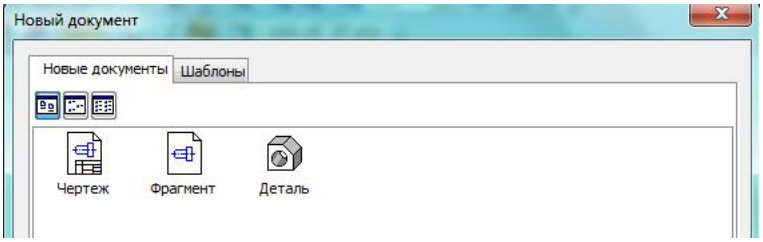


Рис. 1.8.2. Выбор фрагмента

На панели геометрических построений выберем **«Ввод кривой Безье»** –  . Построим кривую Безье по заданным точкам. Выполним последовательно привязку к точкам A, B, C и D с координатами: (0; 0), (20; 50), (50; 50) и (70; 0) (рис. 1.8.3). После задания точек создаем объект – **«кривая Безье»** (кнопка ).

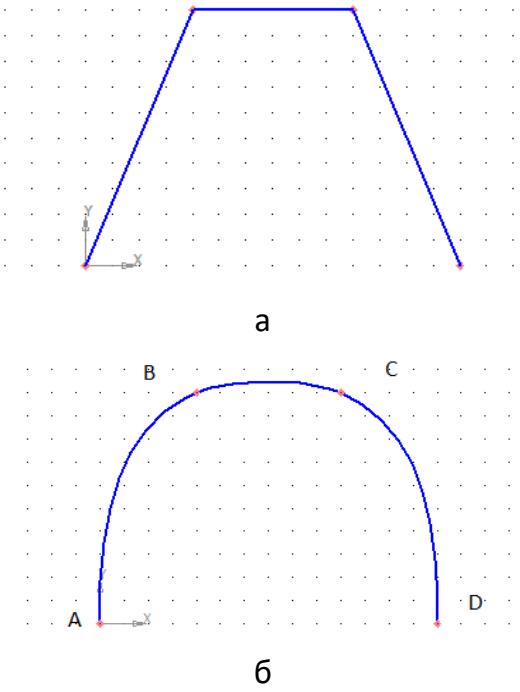


Рис. 1.8.2. Ломаная линия (а) и кривая Безье (б)

Кривая Безье действительно напоминает гибкую линейку, которая закреплена в вершинах. Кривая Безье может быть замкнутой. Для замыкания необходимо выбрать соответствующий режим на панели свойств (рис. 1.8.3).

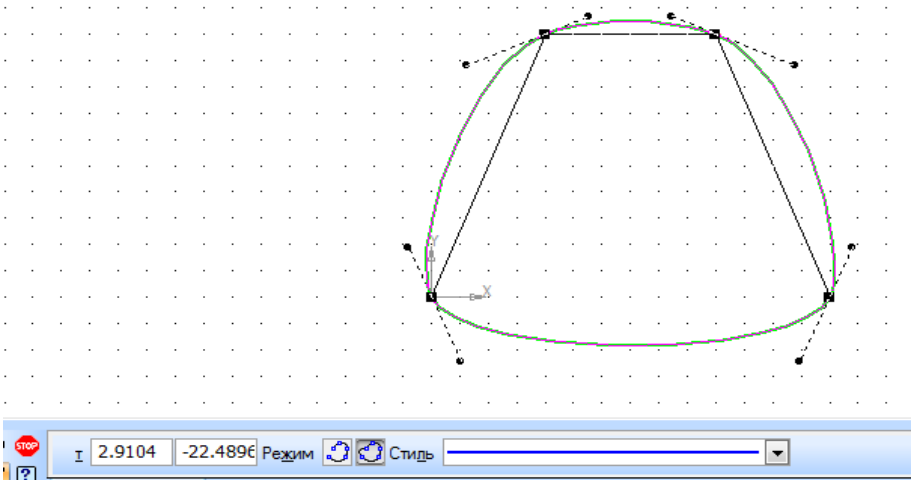


Рис. 1.8.3. Замкнутая кривая Безье

Редактирование кривой Безье

Щелкните дважды на кривой Безье. У каждой вершины кривой Безье появились касательные, на концах которых есть управляющие маркеры (рис. 1.8.4).

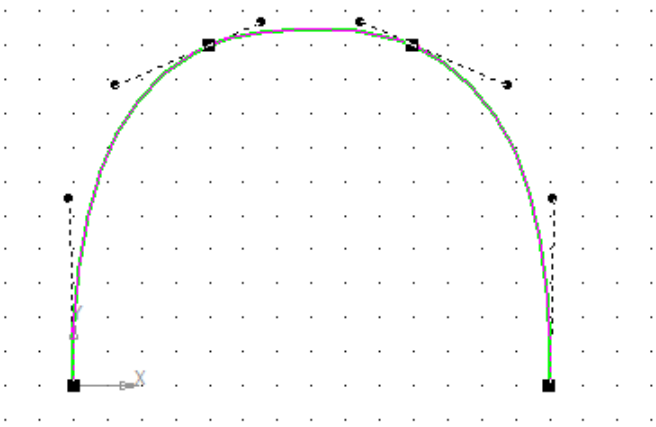


Рис. 1.8.4. Кривая Безье в режиме редактирования

Переместите вершины кривой Безье. Поверните управляющие точки (маркеры); добавьте новые вершины к кривой Безье. Для этого достаточно щелкнуть мышью в нужном месте кривой (рис. 1.8.5). Для удаления точки щелкните мышью на нужной точке и нажмите клавишу **«Delete»**.

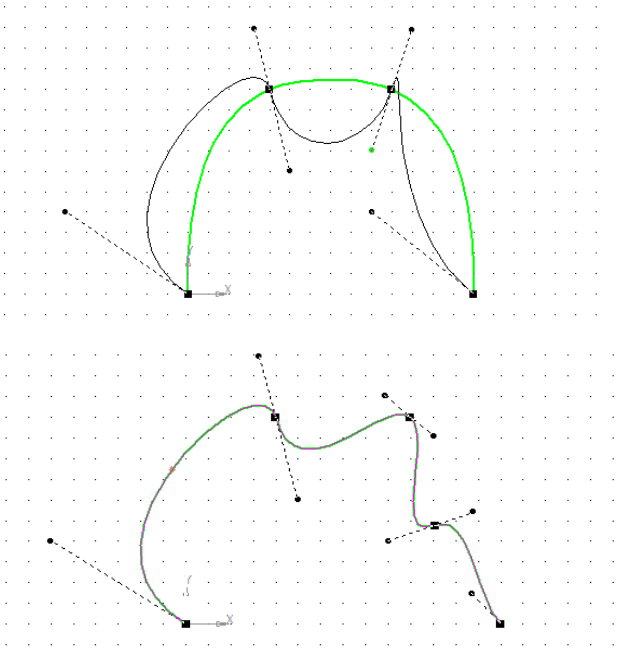


Рис. 1.8.5. Изменение параметров кривой Безье

Проделайте эти операции и обратите внимание на изменения, которые происходят с кривой Безье.

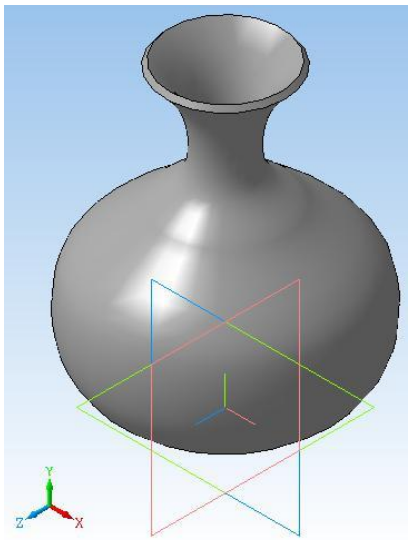


Рис. 1.8.6. Ваза

Задание: создать вазу (рис. 1.8.6). Будем выполнять с помощью операции **«Вращения».** В качестве образующей используем **«Кривую Безье»**.

1. Выберите плоскость **XY,** задайте ориентацию **«Нормально к…».** и выполните команду **«Эскиз»** .

2. Выберите команду **«Кривая Безье»** . Для создания образующей вазы выберем семь точек со следующими координатами: (0; 0), (40; 20), (40; 40), (20; 60), (10; 70), (10; 85) и (20; 100). До создания объекта вы видите фантом кривой Безье и управляющие точки.

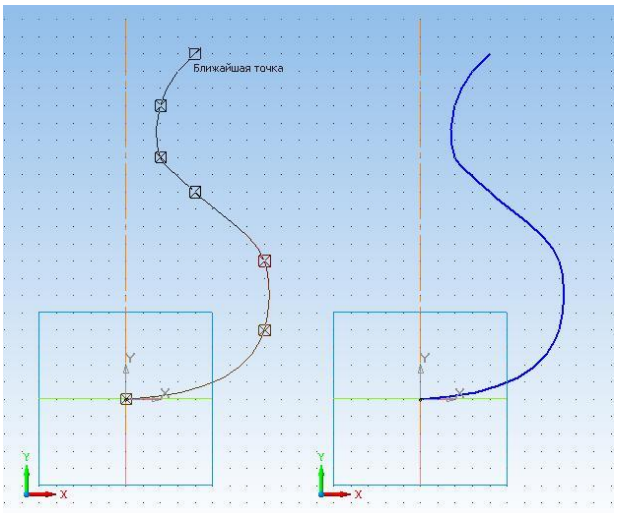


Рис. 1.8.7. Кривая Безье и ее фантом

3. Создайте объект. **«Кривая Безье»** будет иметь следующий вид (рис. 1.8.7). Для создания кривой можно щелкнуть правой кнопкой мыши и выбрать пункт **«Создать кривую Безье»**.

Так как мы используем для моделирования вазы операцию **«Вращение»**, то необходимо провести вертикальную линию – ось вращения.

4. Выберите команду **«Отрезок».** Смените тип линии на осевую.

5. Привяжитесь к точке (0; 0) и проведите вертикальный отрезок. Длина отрезка значения не имеет (рис. 1.8.7).

6. Закончите редактирование эскиза  .

7. Выберите текущую ориентацию **«Изометрия XYZ».**

8. В **«Дереве модели»** выберите **«Эскиз: 1»** и примените к нему операцию вращения, выполнив команду **«Операции»–«Операция»–«Вращения»**. Вызовите панель свойств (**«Вид»–«Панели инструментов»–«Панель свойств»**). В окне диалога укажите:  **«Тороид», «Два направления»** (по 180^0). Параметры тонкой стенки: **«Внутрь», «Толщина»** (2 мм)**.** Нажмите кнопку **«Создать»**.

В режиме отображения **«Полутоновое»** вы увидите вазу, (рис. 1.8.8).

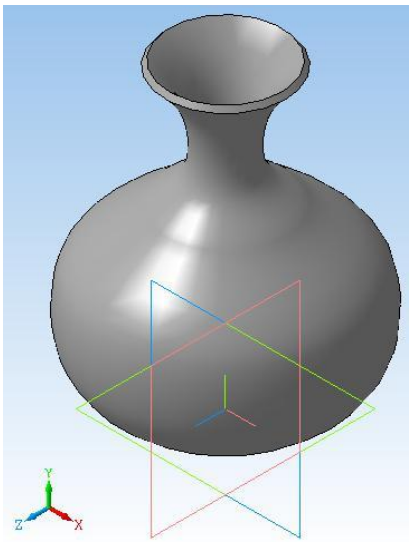


Рис. 1.8.8. Деталь после выполнения операции вращения

Если повернуть вазу и рассмотреть ее с разных сторон, то станет ясно, что такая ваза не сможет стоять на столе.

Задание: создать дно вазы.

Дном вазы будет служить цилиндр высотой 5 мм , радиусом 15 мм.

1. Выберите плоскость **ZX**, ориентация **«Снизу»** (**«Вид»–«Ориентация»–«Снизу»)**, режим отображения, **«Невидимые линии тонкие»** (**«Вид»–«Отображение»–«Невидимые линии тонкие»**).

2. Выберите команду **«Эскиз»**.

3. Выберите команду **«Ввод окружности»**. Смените тип линии на **«Основную»**! Постройте окружность, центр в точке (0; 0), диаметр 30 мм.

4. Завершите редактирование эскиза.

5. Выберите текущую ориентацию **«Изометрия XYZ»** (рис. 1.8.9).

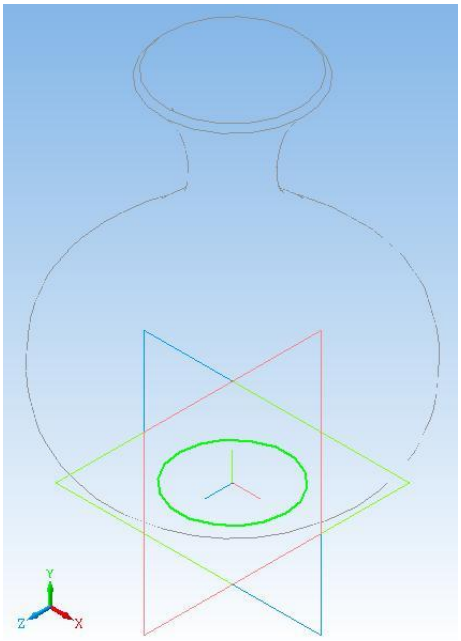


Рис. 1.8.9. Приклеивание дна вазы

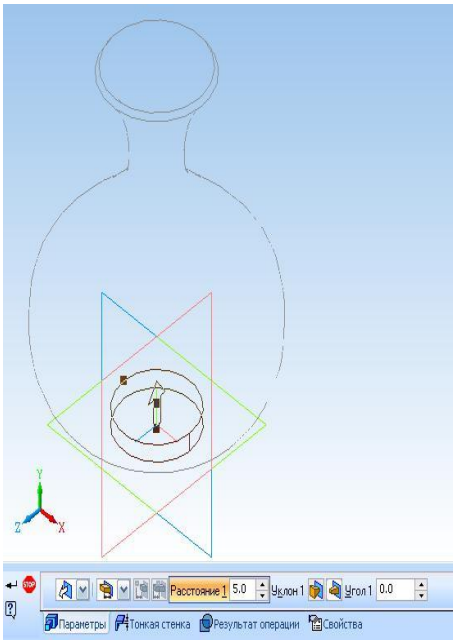


Рис. 1.8.10. Выполнение операции «Выдавливания»

6. Выделите в **«Дереве модели» «Эскиз: 2»** и выполните команду **«Операция выдавливания»**. В окне панели свойств установите направление выдавливания – **«Прямое»** на расстояние 5 мм (рис. 1.8.10). При этом дно должно «проникнуть» в тело вазы. После выполнения команды **«Создать»** ваза примет вид по рис. 1.8.11.

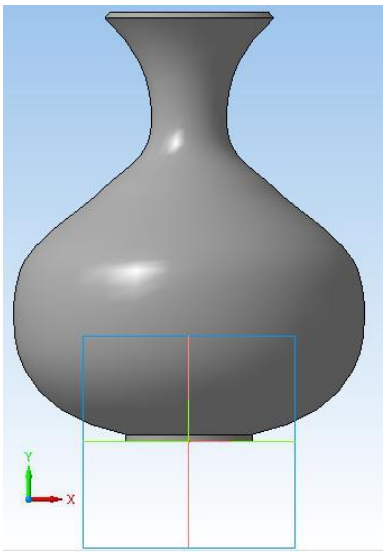


Рис. 1.8.11. Готовая ваза

1.9. Построение конуса, призмы, пирамиды

В данной работе постоим несколько моделей с использованием операций **«Выдавливания»**, **«Вырезать выдавливанием»**. Будут поострены простейшие геометрические тела: призма, пирамида, конус, которые используются для построения более сложных моделей.

Постановка задачи 1. Создать 3D-модель усеченного конуса. Диаметр основания конуса 60 мм. Высота конуса 40 мм.

Порядок выполнения работы. Запустим систему трехмерного моделирования. В дереве модели выделим плоскость **XY**. Зададим ориентацию выделенной плоскости **«Нормально к…»**. Включим режим редактирования эскиза и режим отображения сетки. Установим привязки **«По сетке»**. Это типовые операции, которые выполняются всегда при построении трехмерной модели.

Создадим первый эскиз – окружность диаметром 60 мм (рис. 1.9.1).

Закончим редактирование эскиза и зададим ориентацию эскизу **«Изометрия XYZ»**. Далее выполним операцию **«Выдавливания»**. Расстояние выдавливания 40 мм (рис. 1.9.2).

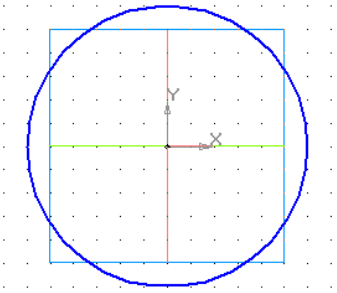


Рис. 1.9.1. Первый эскиз – окружность

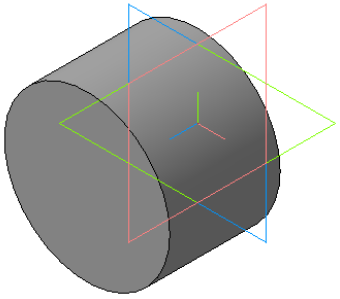


Рис. 1.9.2. Заготовка детали

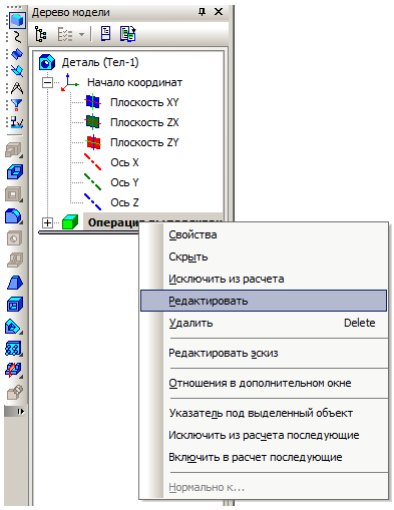


Рис. 1.9.3. Включение режима редактирования операции

В дереве модели выделим операцию выдавливания и выберем пункт контекстного меню **«Редактировать»** (рис. 1.9.3). В этом режиме становятся доступны параметры операции (рис. 1.9.4), которые можно изменить.

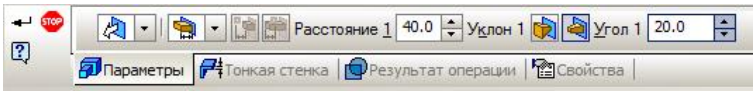


Рис. 1.9.4. Панель свойств операции

На панели свойств зададим режим **«Уклон внутрь»** и величину угла уклона в 20^{o} (рис. 1.9.4). Фантом операции представлен на рис. 1.9.5.

В результате выполнения операции получим готовую деталь (рис. 1.9.6). Подбором угла уклона можно получить не усеченный конус. Для усеченного конуса по рис. 1.9.6 такой угол составляет $36{,}7^{o}$.

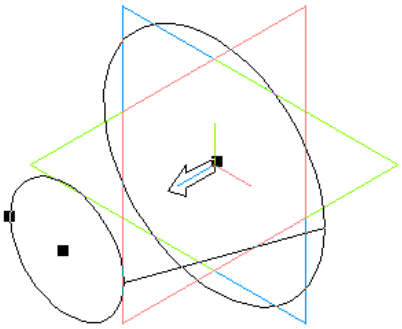


Рис. 1.9.5. Фантом цилиндра с уклоном

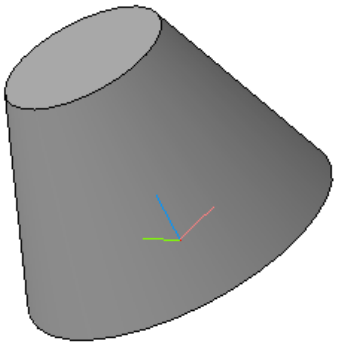


Рис. 1.9.6. Усеченный конус

Постановка задачи 2. Аналогичным способом построим пирамиду. Основание пирамиды – правильный шестиугольник с радиусом вписанной окружности 60 мм.

Порядок выполнения работы. Выполним все типовые операции, которые описаны выше. Для построения эскиза шестиугольника, выберем инструмент **«Многоугольник»** (рис. 1.9.7).

Рис. 1.9.7. Выбор инструмента «Многоугольник»

На панели свойств многоугольника (рис. 1.9.8) выбираем число вершин и способ построения **«По вписанной окружности».** При построении многоугольника его центр привяжем к началу координат. Диаметр окружности отображается при построении шестиугольника (рис. 1.9.9).

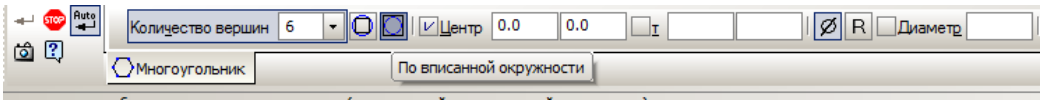


Рис. 1.9.8. Панель свойств многоугольника

После создания эскиза выполним операцию выдавливания на расстояние 100 мм (рис. 1.9.10). Зададим уклон внутрь и угол $16{,}5^{o}$. После выполнения операции получим пирамиду (рис. 1.9.11).

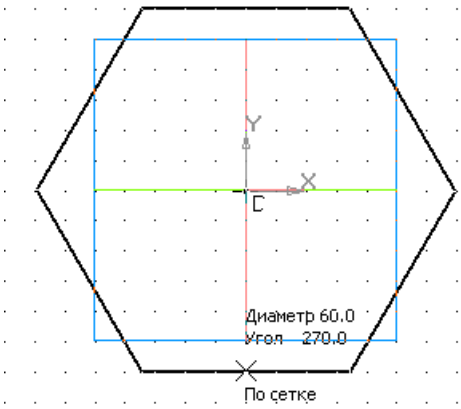


Рис. 1.9.9. Построение шестиугольника

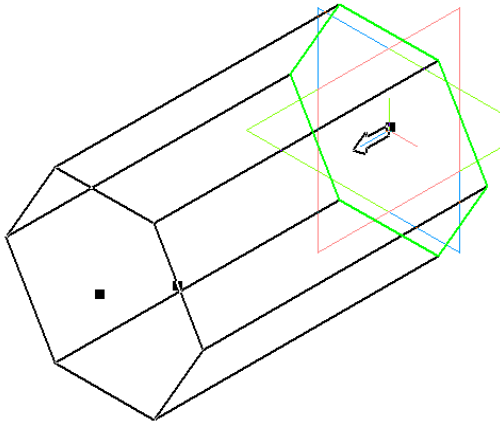


Рис. 1.9.10. Выполнение операции выдавливания

Эскиз детали можно изменить. В дереве модели выделим эскиз операции выдавливания и выберем пункт контекстного меню: **«Редактировать»**. Затем удалим шестиугольник и построим восьмиугольник с теми же параметрами. После завершения редактирования деталь автоматически будет перестроена (рис. 1.9.12).

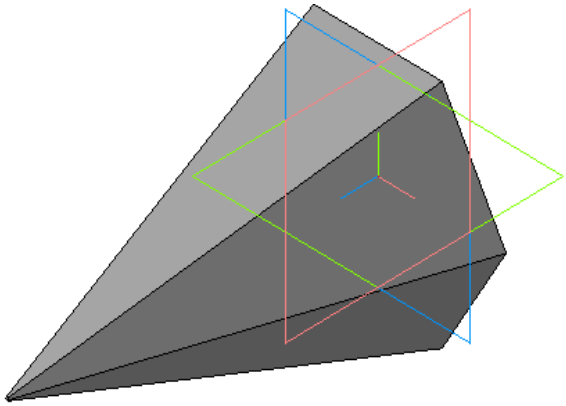


Рис. 1.9.11. Готовая деталь – пирамида

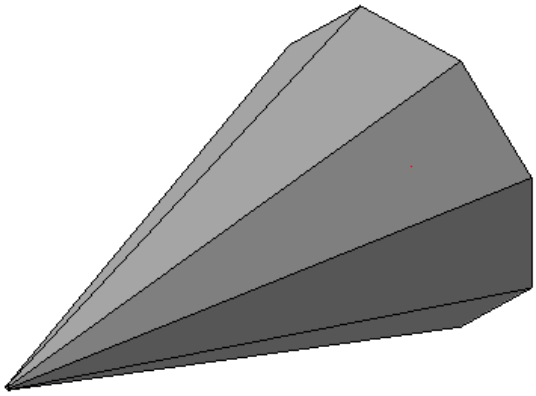


Рис. 1.9.12. Преобразованная пирамида

В **«Дереве модели»** в контекстном меню для операции выдавливания выделим пункт **«Редактировать»**. На панели параметров операции (рис. 1.9.13) установим режим **«Два направления»**. Для второго направления зададим значения параметров одинаковые с первым направлением. Результат операции представлен на рис. 1.9.14.

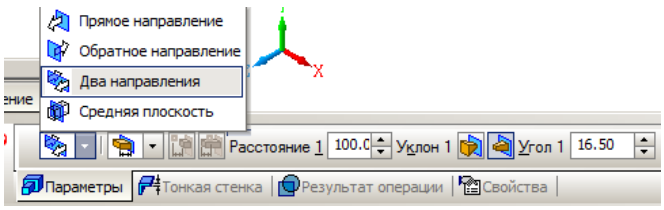


Рис. 1.9.13. Режим выдавливания: «Два направления»

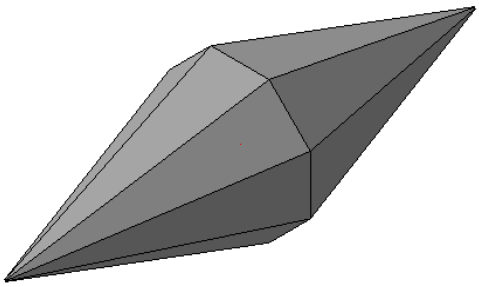


Рис. 1.9.14. Новая деталь

1.10. Контрольные вопросы к главе 1

1. Что такое САПР?
2. Что такое каркасное моделирование?
3. Что такое поверхностное моделирование?
4. Что такое твердотельное моделирование?
5. Перечислите возможности твердотельного моделирования?
6. Перечислите основные твердотельные операции.
7. Сколько отрезков требуется для операции вращения?
8. С помощью какой 3D-операции можно произвести вырезание?
9. Можно ли восстановить вид модели после выполнения какой-либо 3D-операции?
10. Можно ли изменить эскиз 3D-операции, что при этом произойдет?

ГЛАВА 2. КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ В СРЕДЕ MVS

2.1. Компьютерное математические 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.

Математическое моделирование существовало столько, сколько существует наука, как средство познания мира и, насколько возможно, поставить себе на службу его явления. С развитием науки и техники возникла потребность в обработке и объяснении результатов испытаний, результатов, полученных в виде каких-то зависимостей (например, дифференциальных уравнений или алгебраических соотношений). Обработка полученных результатов приводила к необходимости сложных вычислений. Такое положение дел дало стимул не только 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 компьютерного и математического моделирования могут быть кратко сформулированы как качественное и количественное изучение всевозможных 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 (смотри, например, Арнольд, В.И. «Жесткие» и «мягкие» математические модели / В.И. Арнольд. – Москва: МЦНМО, 2000).

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. Требования от моделирования точных количественных результатов и полного качественного исследования противоположны: первое требует достаточно полного учета многих факторов, усложняющих модель; второе, напротив, упрощения модели.

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, 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. К таким задачам относится, 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.

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 анализа приводят к рождению новых теорий, понятий, моделей. Изучение этих моделей с помощью ком-

пьютеров приводит к рождению теорий и моделей нового поколения. Многие важнейшие открытия в науке 20-го столетия связаны с выявлением эффектов согласованного поведения (синергизма) на макроуровне совокупностей отдельных элементов, хаотически ведущих себя на микроуровне.

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, с поведением людей, волнами моды, социальными течениями, войнами и революциями, втягивающими огромные массы людей, часто даже против их воли.

В появлении упорядоченности важную роль играют диссипативные процессы: диффузия, вязкость, теплопроводность и множество других. Однако представление о том, что эти 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 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 рас-

ширили сферы приложения математического моделирования, без которого нельзя представить современную науку и технику.

Актуальность математического моделирования связана еще 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, которое дает возможность сравнения выводов теории с результатами наблюдений и экспериментов.

Теоретические выводы всегда получают точный и практически значимый характер, если они выражены в виде количественных соотношений. Прямой натурный эксперимент для многих 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 математиче-

ское моделирование. Оно является незаменимой составляющей в развитии науки и техники.

Применение компьютерного моделирования дает множество эффектов.

Значительное расширение возможностей математического моделирования по сложности решаемых задач, возможность проведения **вычислительного эксперимента**.

Появление **новых** видов моделирования:

- Имитационное моделирование
- Моделирование знаний
- **Визуализация** результатов моделирования (3D-графика, анимация, виртуальная реальность)
- **Автоматизация** построения самой модели, выбора **математических методов** и средств **отображения результатов**

Любой объект имеет множество свойств. Математическое моделирование какого-либо объекта или процесса связано с отражением **количественных характеристик** его свойств, как правило, в числовом виде. Как уже отмечалось, в последнее время получило развитие направление применения и изучения так называемых «жестких» и «мягких» математических моделей. «Мягкие» модели ориентированы на изучение качественных свойств объектов или изучение тенденций в развитии процессов.

Количественное выражение свойства объекта выполняется с помощью системы **параметров**, но только в том случае, если эти свойства можно **измерить**. Лишь в результате измерения свойств параметры могут получить свои значения. Измерения любого свойства всегда связаны с какой-либо **шкалой**, в рамках которой они проводятся. Именно шкала позволяет выполнить сравнение нескольких объектов по одному свойству.

В результате измерения наблюдаемому состоянию объекта ставится в соответствие определенное обозначение: число, номер, знак, символ. Такое соответствие обеспечивает информативность результатов измерений.

Будем рассматривать такие состояния объекта, про которые определенно можно сказать, различимы они или нет. Допустим, что число различимых состояний конечно. Каждому состоянию поставим в соответствие свое обозначение. Суть измерений состоит в определении принадлежности результата к конкретному классу. Результат измерения описывают с помощью определенного обозначения. Такие измерения называются измерениями в **шкале наименований** (классификационной шкале).

Раз обозначения классов – символы (даже, если это цифры), то при обработке данных в шкале наименований можно выполнить только операцию проверки различия или совпадения этих результатов.

Когда наблюдения позволяют **сравнивать** разные классы, для измерения можно выбрать более сильную **порядковую шкалу.** В этом случае между классами можно установить соотношения типа A > B или B = A. К таким шкалам можно отнести систему воинских званий, обозначение мест в конкурсе, оценки успеваемости в процессе обучения (школьные отметки не являются числами). В этом случае для обозначения классов могут использоваться слова (отлично, хорошо и т.д.) или цифры (5, 4, 3, 2). Отношения порядка не определяют **«расстояния»** между классами. Результаты измерений в таких шкалах нельзя обрабатывать как числа (например, вычислять среднеарифметическое значение оценок успеваемости).

Если упорядочение можно выполнить настолько точно, что становятся известны **расстояния** между любыми двумя классами, то измерения можно проводить в **шкале интервалов.** Расстояние между классами должно выражаться в единицах, одинаковых по всей шкале. Подобная шкала может иметь произвольное начало отсчета. Такие шкалы используются при измерении времени, высоты, температуры. В подобных шкалах только длины интервалов имеют смысл настоящих чисел, а с числами можно выполнять арифметические операции. Частным случаем интервальных шкал является **циклическая шкала** (время суток, время года, фаза колебаний и т.п.).

Наиболее «сильной» шкалой является **абсолютная шкала,** она имеет абсолютный нуль и абсолютную единицу. Абсолютная шкала безразмерна (например, абсолютная шкала температур). С результатами измерений в такой шкале можно выполнять любые операции.

Рассмотренные выше шкалы имеют общее свойство: они основаны на том, что два состояния или результаты двух измерений либо тождественны, либо различимы. В действительности очень часто встречаются случаи, когда о результатах измерения нельзя говорить с полной уверенностью. Наиболее ярко это видно на примере шкал, где классы обозначаются словами естественного языка, например: высокий, низкий; тяжелый, легкий и т.д. В подобных случаях мы имеем дело с нечеткой оценкой свойств объекта. Размытость встречается не только в естественном языке. В математике успешно применяются понятия **«значительно больше»** (>>) или **«приблизительно равно»** (≅), которые являются расплывчатыми.

Опираясь на общее определение модели, можно сказать, что математическая модель – это математический объект, который по своим свойствам подобен объекту-оригиналу. Математическая модель – это абстракция, в которой отношения и связи между реальными элементами объекта моделирования **заменены подходящими математическими соотношениями между параметрами объекта**. Математическая модель отражает количественные характеристики процессов, которые протекают в объекте, т.е. она отражает связи между параметрами объекта. Параметры 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, т.е. характеризуют внешнее проявление объекта.

3. **Внутренние параметры.** Характеризуют свойства процессов, протекающих в самом объекте. Внутренние параметры могут быть зависимы-

ми 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);
- совокупностью воздействий окружающей среды: V(t);
- совокупностью внутренних независимых параметров*:* H(t);
- совокупностью внутренних зависимых параметров: P(t);
- совокупностью выходных параметров системы: Y(t).

С учетом вышесказанного, 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 стороны. 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.

Статические модели отражают состояние объекта, которое не меняется во времени. 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 фундаменталь-

ных 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, аналитическое описание и экспериментальное определение неизвестных параметров модели позволяют 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.

В данном разделе в качестве примеров построения математических моделей используются модели достаточно простых процессов. Материал будет рассмотрен в рамках подхода, основанного на построении обобщенных безразмерных моделей и применении инструментальной системы моделирования для их реализации.

Математические модели многих объектов или процессов, как правило, имеют достаточно большое количество параметров. Это существенно затрудняет выявление каких-либо закономерностей. Построение безразмерной модели еще на этапе, предшествующем проведению вычислительных экспериментов с математической моделью, позволяет существенно повысить ее информативность и информативность результатов моделирования. Это следствие сокращения количества параметров и выявления безразмерных комплексов, которые и определяют свойства моделируемого объекта.

Построение безразмерной модели позволяет установить законы **подобия**. При построении моделей технических объектов были впервые разработаны основные положения теории подобия, которые впоследствии нашли применение при решении многих задач.

Рассмотрим общую методику построения безразмерных моделей, суть которой состоит в следующем:

1. **Построение модели.** Математическая модель строится в виде системы уравнений в размерной форме (параметры имеют размерность). Только в этом виде уравнения отражают «физическую» суть моделируемых процессов.

2. **Определение безразмерных переменных.** Все переменные (пространственные координаты, время, искомые переменные) представляются в безразмерной форме путем введения неопределенных масштабов.

3. **Преобразование модели к безразмерному виду**. Уравнения, краевые и начальные условия простыми чисто алгебраическими методами преобразуются к безразмерному виду. В этом случае в уравнениях образуются безразмерные комплексы размерных параметров. В эти комплексы включены неопределенные масштабы.

4. **Определение масштабов.** Безразмерные комплексы, содержащие неопределенные масштабы, приравниваются к единице. Тем самым образуется система алгебраических уравнений, которая позволит получить выражения для неопределенных масштабов. Естественно, что число алгебраических уравнений, которые используются для определения масштабов, должно быть равно числу определяемых масштабов.

В оставшихся безразмерных комплексах масштабы теперь получают конкретные значения. Эти комплексы и образуют безразмерные параметры задачи. Таким, чисто формальным путем, могут быть получены естественные безразмерные параметры модели, которые в каждом конкретном случае имеют свой «физический» смысл. Результаты исследования безразмерной модели распространяются на множество реальных объектов, для которых безразмерные параметры имеют одинаковое значение.

Построим модель движения тела под действием силы тяжести в среде с сопротивлением. Рассмотрим весь путь построения модели и принимаемые допущения. Обычно подобные задачи решаются без анализа всех необходимых допущений, которые определяют адекватность модели.

Объект представляет собой тело, которое совершает прямолинейное движение под действием силы тяжести и при этом испытывает сопротивление движению со стороны окружающей среды. Расчетная схема процесса представлена на рис. 2.1.1.

$X(t=0) = 0, \quad V(t=0) = V_0.$
$m = const; g = const;$
$P = mg;$
$F = - kV ; k = const.$

Рис. 2.1.1. Расчетная схема движения тела

Допущения, принятые при построении модели:

- Тело имеет правильную геометрическую форму, например шар, движение происходит прямолинейно под действием силы тяжести и силы сопротивления. Законы движения шарообразного тела и тела, имеющего форму пластины, имеют существенное различие.
- Масса тела – постоянная величина. Таким образом, модель не описывает, например, движение тел, вещество которых испаряется, сгорает или растворяется.
- В процессе движения форма тела не изменяется. Капля жидкости изменяет свою форму, для описания ее движения требуются более сложные модели.
- Плотность тела существенно выше плотности окружающей среды. Таким образом, силой Архимеда можно пренебречь.
- Вращение тела отсутствует. Таким образом, полет футбольного мяча данная модель в полной мере описать не в состоянии.
- Сила сопротивления линейно зависит от скорости движения тела: *F = kV.* Данное допущение справедливо в определенном диапазоне скоростей движения тела. На практике часто реализуется квадратичный закон сопротивления: *F = k V abs(V)*.
- Гравитационная сила – постоянная величина. Таким образом, модель не распространяется на описание процессов космического масштаба,

например: движение космического летательного аппарата по околоземной орбите или полет метеорита.

Если перечисленные допущения выполняются, то тело можно считать **материальной точкой,** движение которой описывается законами классической механики Ньютона. Все допущения направлены на то, чтобы определить область корректного применения законов движения материальной точки. Однако определение значения коэффициента *k* возможно только путем идентификации этого параметра или с помощью полуэмпирических зависимостей.

На примере данной модели мы хотим подчеркнуть, что любая модель имеет свою область применения. Действительно, перечень допущений достаточно велик, и невыполнение любого из них приведет к неадекватным результатам моделирования. Например, вращение тела при движении в среде с сопротивлением создает дополнительную силу (эффект Магнуса). Проявление данного эффекта можно наблюдать в ходе футбольного матча, когда мяч после углового удара залетает в ворота.

Цель моделирования - построить модель, на основе которой можно определить закон изменения скорости движения тела *V(t)* и закон изменения координаты *x(t)* во времени. В соответствии с принятыми допущениями модель движения тела строится на основе второго закона механики. Система дифференциальных уравнений, описывающая движение тела, имеет вид:

$$m\frac{dV}{dt} = mg - kV, \quad \frac{dx}{dt} = V.$$

Начальные условия: $V(t = 0) = V_0$; $X(t = 0) = 0$.

Здесь *m* – масса, *g* – ускорение свободного падения, *k* – коэффициент сопротивления, *V* – скорость, *t* – время, *x* – координата.

Данная модель имеет четыре параметра: *m*, *k, g,* V_0, что существенно затрудняет анализ. Для упрощения анализа результатов моделирования необходимо свести количество параметров к минимуму.

Преобразуем модель к безразмерному виду следующим образом (обе части уравнения разделим на *mg*):

$$\frac{1}{g}\frac{dV}{dt} = 1 - \frac{k}{mg}V .$$

Определим безразмерную скорость как отношение текущего значения скорости к ее начальному значению V_0: $\bar{V} = V / V_0$. С учетом этого соотношения преобразуем уравнение движения:

$$\frac{V_0}{g}\frac{d\bar{V}}{dt} = 1 - \frac{kV_0}{mg}\bar{V}.$$

Обозначим: $t^* = V_0/g$ – характерное время процесса. Введем безразмерное время $\bar{t} = t/t^*$. Тогда уравнение движения примет вид:

$$\frac{d\bar{V}}{d\bar{t}} = 1 - (\frac{kV_0}{mg})\bar{V} .$$

Обозначим $\bar{k} = kV_0/mg$ – безразмерный коэффициент сопротивления. С учетом принятых обозначений уравнение движения запишем в следующей форме:

$$\frac{d\bar{V}}{d\bar{t}} = 1 - \bar{k}\bar{V}.$$

Кинематическое уравнение преобразуется аналогичным образом:

$$\frac{1}{V_0 t^*}\frac{dx}{d\bar{t}} = \bar{V},$$

$$\bar{x} = x/(V_0 t^*).$$

В итоге получим систему безразмерных дифференциальных уравнений:

$$\frac{d\bar{V}}{d\bar{t}} = 1 - \bar{k}\bar{V}, \quad \frac{d\bar{x}}{d\bar{t}} = \bar{V}.$$

$$\bar{x}(\bar{t} = 0) = 0, \quad \bar{V}(\bar{t} = 0) = 1.$$

После преобразований задача приведена к безразмерному виду и имеет всего один безразмерный параметр $\bar{k}$. Естественно, анализ свойств подобной модели проводить значительно проще, чем исходной.

При анализе модели в исходном размерном виде, задав конкретные значения параметров, мы установим свойства лишь единственной конкретной системы. Анализ модели в безразмерной форме для заданного значения $\bar{k}$ дает информацию о свойствах бесконечного числа реальных систем, для которых:

$$\bar{k} = \frac{kV_0}{mg} = const.$$

Различные реальные системы, имеющие одинаковые значения параметра $\bar{k}$, называются **подобными**, а параметр $\bar{k}$ для данной задачи называется критерием подобия.

В частном случае, когда $V_0 = 0$, можно получить безразмерную модель с нулевым количеством параметров. Такая система называется **автомодельной**. В этом случае все реальные системы подобны друг другу.

Таким образом, проделанные предварительные преобразования существенно повысили информативность модели и упростили дальнейший вычислительный эксперимент.

На примере рассматриваемых в данном разделе задач показан весь цикл построения математической модели с момента принятия упрощающих допущений до получения уравнений, описывающих моделируемые процессы, и компьютерной реализации модели в среде инструментальной системы моделирования. Инструментальная система моделирования MVS существенно упрощает построение компьютерной модели, т.к. выбор численного метода, генерация программного и исполняемого кодов производится автоматически средствами системы.

Понятие подобия физических процессов родственно понятию геометрического подобия. В этом случае под подобием понимается такое взаимно однозначное соответствие между сопоставляемыми объектами или процессами, при котором правила перехода от параметров одного объекта к параметрам другого объекта известны или заданы.

Замечательным свойством окружающего нас мира является то, что модели процессов и явлений самой разнообразной природы сводятся к тождественным по форме безразмерным математическим моделям.

Наиболее известными примерами являются электротепловая **аналогия** и электромеханическая аналогия. Оказывается, что закономерности электрических и механических процессов, электрических и тепловых процессов описываются одинаковыми уравнениями. Различие состоит лишь в разной физической интерпретации переменных и параметров, входящих в эти уравнения.

По **аналогии** можно применить результаты исследования математических моделей на различные процессы другой физической природы, которые описываются аналогичными математическими моделями.

Таким образом, выявление законов подобия при моделировании систем различной природы показывает, как наиболее информативно представить результаты модельных экспериментов и выявить закономерности процессов.

Построение математической модели и преобразование ее к безразмерному виду позволяет уже на этой стадии получить определенную информацию о свойствах решения и, следовательно, объекта моделирования.

Существует целый класс качественных математических методов, предназначенных для выявления свойств объекта или процесса без решения уравнений, составляющих математическую модель этого объекта. Они позволяют установить свойства решения только на основе анализа параметров модели. Однако во многих случаях получение решения уравнений математической модели необходимо.

Используемые в настоящее время методы исследования различного рода математических моделей в прикладных областях можно разделить на следующие виды: точные методы; асимптотические методы; приближенные методы; численные методы.

Под **точными методами** понимаются методы, которые позволяют получить решение исходной задачи в **аналитическом виде**. Такие методы

применимы для решения достаточно простых линейных задач с постоянными коэффициентами. Достоинством точных аналитических решений является их наглядность, компактность и большая информативность.

Для решения современных задач моделирования такие методы практически не применяются. Однако точные решения могут использоваться в качестве тестовых задач при разработке других методов (приближенных и численных). Технические, экономические, экологические и другие системы, изучаемые современной наукой, не поддаются исследованию в необходимой для практики полноте аналитическими методами. Это обстоятельство обусловлено сложностью и нелинейностью их математических моделей. Для нелинейных задач большим достижением считается получение даже частных решений.

Изучение любого явления начинается с определения **основных закономерностей** его функционирования на качественном уровне. При этом уравнения модели часто настолько сложны, что необходимо построение упрощенных моделей, без которых невозможно выявить механизмы процессов и составить их ясное понимание.

В ряде случаев упрощение достигается за счет того, что рассматривается такой вариант задачи, в котором в уравнениях удается выделить **«малый»** параметр. В этом случае решение представляется в виде разложения в ряд по «малому» параметру. Подобные методы носят название **асимптотических**, или методов возмущений, и широко применяются в гидродинамике, квантовой механике и т.д.

До сих пор сохраняют свое значение **приближенные методы**, которые опираются на неформальное понимание сути процессов. Приближенные методы удобны для получения грубых оценок на предварительном этапе исследования. Они играют большую роль в получении качественных представлений и часто используются, например, в инженерной практике, когда исходная задача имеет приближенную постановку и значения параметров определены с точностью до порядка. В качестве примера приближенных методов можно привести решение нелинейной задачи по лине-

аризованной модели, решение задачи с переменными параметрами по алгоритмам для задач с постоянными коэффициентами и т.д.

Как бы ни были разнообразны методы приближенного или качественного анализа математических моделей, область их применения ограничена. Это либо достаточно простые модели, либо упрощенные фрагменты сложных нелинейных моделей.

Как уже отмечалось, математические модели реальных объектов являются нелинейными. Для проведения экспериментов с ними, т.е. для получения информации об их свойствах, требуется компьютерная реализация математических моделей на основе **численных методов**. Это единственный достаточно универсальный способ исследования математических моделей с помощью средств компьютерной техники. Поэтому современное математическое моделирование всегда предполагает применение численных методов анализа и проведение компьютерных вычислительных экспериментов.

Численные решения являются всегда приближенными и имеют дискретный характер. Доступный для компьютера вычислительный алгоритм должен удовлетворять достаточно жестким требованиям. К ним относятся, прежде всего, необходимость получить решение с заданной точностью за разумное время. Объемы обрабатываемой информации при этом не должны превышать возможностей компьютера.

Проблемы численного моделирования не решаются сами собой при развитии вычислительной техники. Так как постоянно происходит усложнение задач, выдвигаемых теорией и практикой, существует необходимость проведения большого количества серий вычислительных экспериментов для более полного изучения объекта. Поэтому разработка и применение эффективных вычислительных методов остается одной из ключевых задач математического моделирования.

Для математических моделей, представленных дифференциальными уравнениями, процесс создания вычислительных алгоритмов состоит из построения **дискретного алгебраического аналога** исходных уравнений и

его численного решения. Дискретный аналог позволяет приближенно определить решение уравнений в конечном числе фиксированных точек по времени или по пространственным координатам.

Численные методы позволяют получить решения в областях, где другие методы не применимы. Однако огромный потенциал численных методов не делает их всесильным средством. К недостаткам численных методов можно отнести слабую наглядность и информативность результатов по сравнению с аналитическими и приближенными решениями, трудоемкость реализации, трудности применения в разрывных точках, отсутствие строгого обоснования корректности вычислительных процедур для сложных нелинейных задач, проявление эффектов конечной разрядной сетки кодирования чисел. Любой численный метод привносит в решение задачи количественную погрешность, а в ряде случаев и **качественные эффекты**, распознать природу которых весьма затруднительно.

Рассмотрим основные характеристики численных методов, актуальные для моделирования.

Устойчивость метода. Неустойчивый численный метод приводит к накоплению и неограниченному росту вычислительных ошибок. Характерный признак неустойчивости вычислений заключается в том, что решение имеет пилообразный характер с быстро растущей амплитудой (рис. 2.1.2).

Естественно, что неустойчивые методы применять нельзя. Существуют **условно и безусловно** устойчивые методы. Условно устойчивый метод это такой метод, при котором устойчивое решение получается при выполнении определённого условия для параметров вычислений. Для безусловно устойчивого метода устойчивость вычислений обеспечивается при любых сочетаниях вычислительных параметров.

Степень точности метода (порядок метода)**.** Любой численный метод решения, например, дифференциальных уравнений имеет основной характерный параметр. Обычно степень точности метода указывается в виде зависимости погрешности метода от величины этого параметра, например:

$\delta \sim h^2$, где δ – погрешность метода, h – шаг вычисления. Такой метод называется методом второго порядка точности, если шаг вычислений уменьшить в 2 раза, то погрешность метода уменьшится в 4 раза.

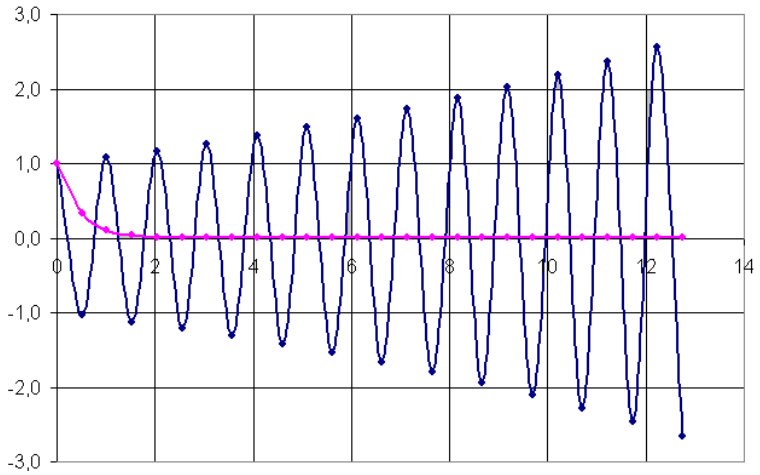


Рис. 2.1.2. Результаты решения задачи устойчивым и неустойчивым методами

Выбор численного метода при реализации математической модели имеет большое значение. Так как любой численный метод является приближенным, то результаты моделирования будут содержать определенную заранее неизвестную ошибку. Существует множество научных публикаций, в которых проявление свойств численных методов ошибочно принималось в качестве свойств объекта моделирования.

Следует иметь в виду, что выбор численного метода определяется задачей моделирования, исходя из принципа **«оптимальной неточности»**. Это означает, что в задачах с неточными исходными данными или при качественном анализе процессов применение методов высокой степени точности ничем не оправдано. В большинстве случаев результат может быть получен на основе простейших методов.

Сходимость метода. Сходимость – теоретическая характеристика метода, которая констатирует тот факт, что в пределе метод может дать точное решение при стремлении его основного параметра к нулю (или бесконечности). В практике численного моделирования играет большую роль **скорость сходимости** метода, т.е. скорость приближения к «точному» значению. Например, при решении нелинейных алгебраических уравнений наибольшую скорость сходимости имеет метод Ньютона.

2.2. Инструментальная среда моделирования Model Visio Studium

Model Vision Studium (**MVS**) – это интегрированная графическая среда для быстрого создания интерактивных 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 **гибридных** (дискретно-непрерывных) систем; обеспечение достоверного численного решения; обеспечение **визуализации** результатов моделирования *без какого-либо **программирования**.*

Пакет предназначен для численного моделирования гибридных 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** – анализ свойств решаемой математической задачи и выбор и настройка метода решения должны выполняться MVS, а не пользователем. Пользователь должен иметь возможность активно вмешиваться в ход вычислительного эксперимента, и, при необходимости, получать о решении как можно больше дополнительной информации.

Система MVS основана на использовании схемы **гибридного автомата**, который включает активные динамические объекты и специальную форму наглядного представления гибридного поведения – **карту поведения** (рис. 2.2.3).

Использование карты поведения при описании переключений состояний, а также непосредственное описание непрерывных поведений системы предоставляет большие возможности в описании гибридного поведения со сложной логикой переключений. Каждая модель в MVS представляет собой **проект** (рис. 2.2.1), который включает **виртуальный стенд**

(рис. 2.2.2), **карту поведения** (рис. 2.2.3), **систему уравнений** (рис. 2.2.4) и **описание компонент класса** (рис. 2.2.5).

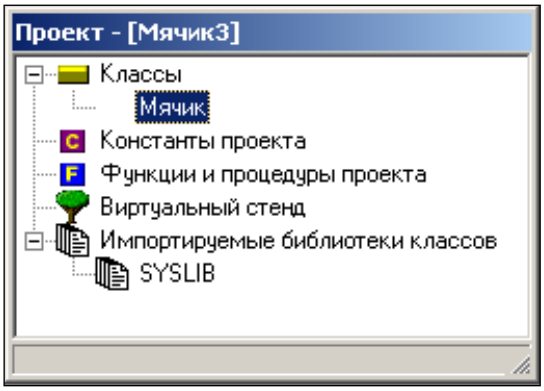


Рис. 2.2.1. Проект «Мячик»

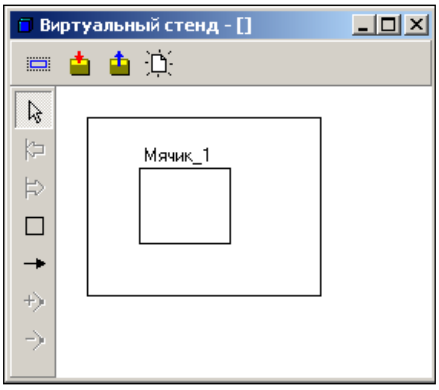


Рис. 2.2.2. Виртуальный стенд проекта

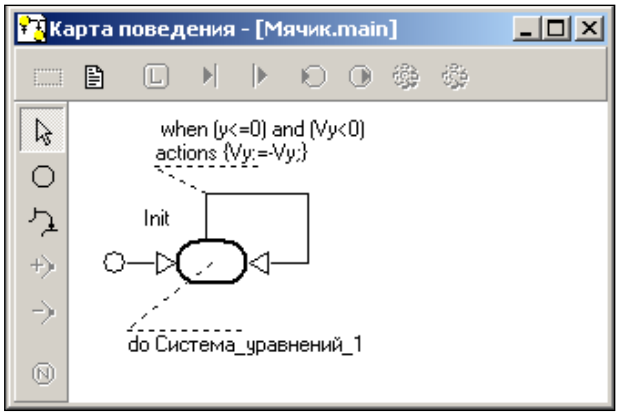


Рис. 2.2.3. Карта поведения

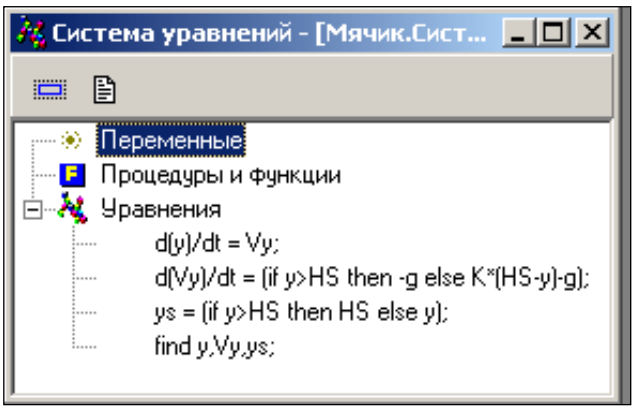


Рис. 2.2.4. Система уравнений

Основным «строительным» элементом в процессе построения модели в MVS является **устройство** (CDevice)**.** Устройство – это некоторый активный объект, функционирующий параллельно и независимо от других объектов в непрерывном времени. Устройство представляет собой систему типа **«вход – выход – состояние»** и может быть как простым, так и составным. В описании устройства (класса) содержатся следующие элементы: **вход, выход, переменные состояния, константы, параметры, поведение** и т.д**.** (рис. 2.2.5).

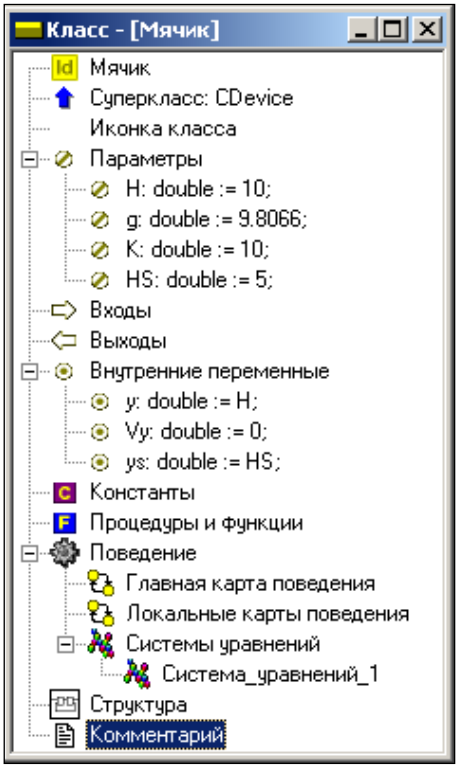


*Рис. 2.2.5. Компоненты класса «**Мячик**»*

Устройства могут объединяться в блоки, что позволяет строить модели, организованные по блочному принципу. Блок является объектом, все взаимодействия которого с окружающей средой осуществляются только через его **входы и выходы**, составляющие **интерфейс** блока (рис. 2.2.6). Блок преобразует информацию, которую получает на **входе**, преобразует ее и передает к **выходу**. Все остальные свойства блока **инкапсулированы** внутри него. **Входы, выходы** и **переменные состояния** являются фазовыми переменными.

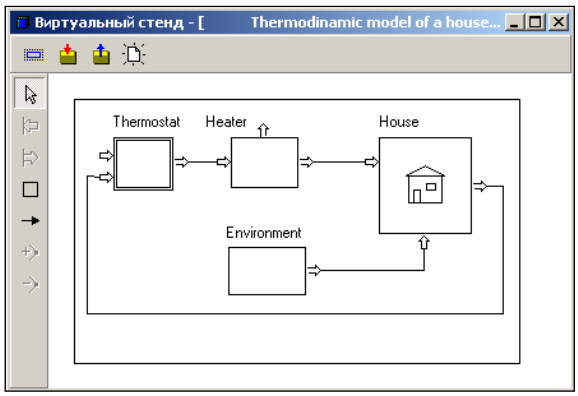


Рис. 2.2.6. Модель, построенная в виде блок-схемы

Блоки могут соединяться между собой однонаправленными функциональными связями и входить в состав других блоков, образуя иерархическую структуру.

Отображение результатов моделирования возможно в виде временной, фазовой диаграмм и 3D-анимации (рис. 2.2.7).

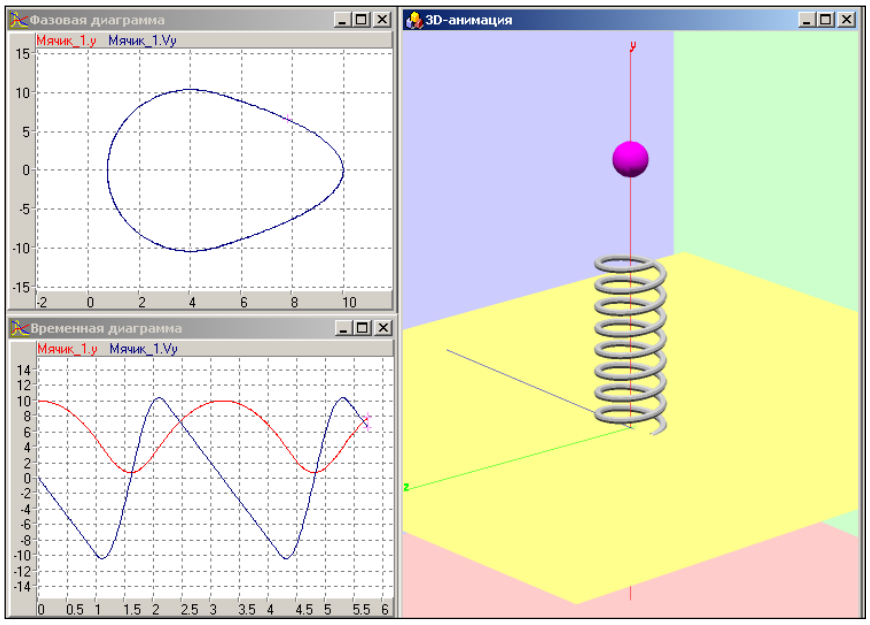


Рис. 2.2.7. Отображение результатов моделирования в виде временной и фазовой диаграмм, а также средствами 3D-анимации

Карта поведения – это ориентированный граф, в котором узлам приписывается некоторое локальное непрерывное поведение, а дуги интер-

претируются как переходы от одного поведения к другому (рис. 2.2.8). В каждый момент времени один из узлов графа является текущим.

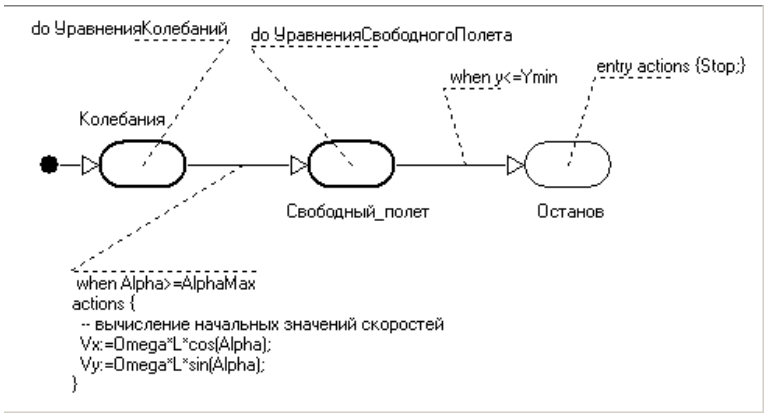


Рис. 2.2.8. Карта поведения с условиями срабатывания переходов

Когда узел становится текущим, создается экземпляр приписанного ему локального поведения, который уничтожается, когда узел перестает быть текущим. Смена текущего узла происходит в результате срабатывания переходов. Один из узлов должен быть помечен как начальный, он становится текущим, когда становится активным экземпляр устройства, к которому принадлежит данная карта поведения. Карта поведения представляет собой простую и наглядную форму визуального представления смены поведений. С картой поведения связан гибридный интерпретатор поведений, объединяющий интерпретатор карт состояний, интерпретатор интерактивных воздействий и менеджер численных методов.

Локальное поведение в узле может быть задано как непрерывное поведение картой поведения или пустым поведением.

Непрерывное поведение в общем случае задается совокупностью обыкновенных дифференциальных уравнений вида $\frac{ds}{dt} = f(s,t)$, алгебраических уравнений вида *F(s,t) = 0* и формул вида *s = <выражение, не зависящее от s>* (рис. 2.2.4).

Карта поведения в MVS должна быть детерминированной, поэтому в каждый момент времени может сработать только один переход. Для перехода должно быть задано запускающее событие: **логическое условие**,

определяющее возможность срабатывания перехода; поступление **внешнего сигнала**; **истечение** наперед заданного **времени** пребывания в текущем узле, действия (actions) в переходе (рис. 2.2.8).

Срабатывание перехода представляет собой следующую последовательность действий: данный узел перестает быть текущим; выполняется последовательность мгновенных действий в том порядке, как они записаны для перехода; текущим становится новый узел.

Для каждого узла могут быть заданы **входные действия** (entry actions) и **выходные действия** (exit actions).

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 алгебро-дифференциальных уравнений.

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 начинает решать каждую новую задачу наименее трудоемким методом, 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 те же программные реализации численных методов, что и автоматическому решателю. Для отладки или первоначального решения задач в форме обыкновенных дифференциальных уравнений, когда у пользователя нет отчетливого представления о свойствах решаемой задачи или в описании поведения допущена ошибка, которую и надо найти, предусмотрена группа отладочных численных методов.

На этапах редактирования исходного описания и генерации кода выполняемой модели проверяется корректность непрерывных поведений.

В пакете MVS используются специальное внутреннее представление информации о проекте в форме деревьев объектов с перекрестными связями, хранящееся в объектно-ориентированной базе данных **MVBase** (на диске вся информация о проекте находится в одном файле с расширением *.mvb).

Редакторы интегрированной среды немедленно проверяют синтаксическую и семантическую правильность вводимых конструкций. При генерации «кода» выполняемой модели на промежуточном языке производится проверка правильности всего проекта в целом, и при обнаружении ошибки неверная конструкция выводится в окно соответствующего редактора. В MVS в качестве промежуточного языка используется Object Pascal. Для всех файлов проекта создается временная рабочая папка с именем **Tmp.**

Система MVS является мощным средством быстрого построения моделей для анализа систем различной степени сложности, имеющих гибридное поведение. Она позволяет строить модели, пользуясь технологией объектно-ориентированного и блочного моделирования, с визуализацией результатов в виде 2D-анимации и 3D-анимации. Дальнейшим развитием системы **MVS** является пакет **RMD**, но для решения учебных задач пакета MVS вполне достаточно.

Для установки пакета запустите программу **setup.exe** и укажите место на жестком диске для размещения пакета. Версия 3.2.x занимает около 25 Мбайт на жестком диске.

Интегрированная оболочка пакета является приложением с многооконным интерфейсом. Многооконный интерфейс предполагает наличие главного окна приложения и произвольного числа дочерних окон (рис. 2.2.9). Заголовок главного окна содержит наименование пакета и путь к открытому в данный момент проекту. Проект – это совокупность данных, относящихся к одной модели. Данные проекта хранятся в нескольких файлах, расположенных в папке данного проекта. Основной файл проекта (база данных проекта) имеет расширение **«.mvb**».

Запуск на выполнение MVS осуществляется из папки **mv30\Bin\mv30.exe** или из меню **«Пуск»** – **«Model Vision Free»**. После запуска MVS откроется окно нового проекта (рис. 2.2.9).

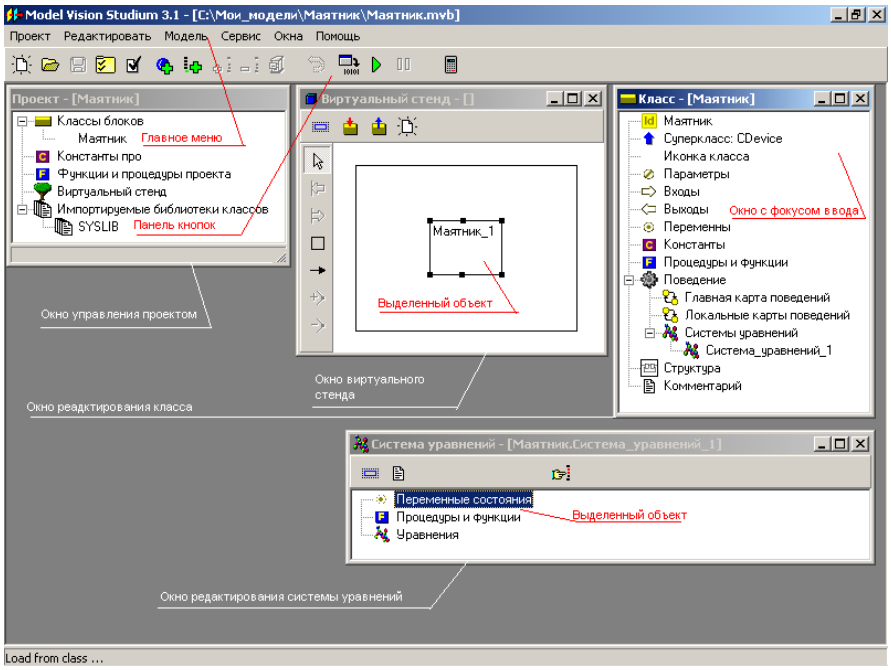


Рис. 2.2.9. Окно MVS при открытии нового проекта

2.3. Моделирование колебаний маятника

Моделируемый объект (маятник) представляет собой материальную точку (шарик достаточно малого размера), прикрепленную к нерастяжимому и невесомому стержню длиной ***L***, другой конец которого шарнирно закреплен в начале системы координат (рис. 2.3.1). Потери на трение в системе отсутствуют.

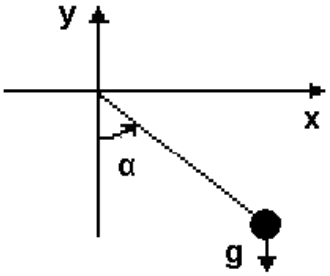


Рис. 2.3.1. Схема объекта моделирования

Динамика колебаний маятника в соответствии с основными законами механики определяется дифференциальными уравнениями:

$$\begin{cases} \dfrac{d\alpha}{dt} = \omega; \\ \dfrac{d\omega}{dt} = \dfrac{-g \cdot \sin\alpha}{L}; \end{cases}$$

с начальными условиями $\alpha(t=0) = \alpha_0, \omega(t=0) = \omega_0$, $\alpha_0 = -\dfrac{\pi}{2}$, $\omega_0 = 0$. Модель содержит **параметр *L*** – длина стержня, **константу *g*** – ускорение свободного падения, **переменные *α* – угол отклонения маятника** и ***ω* – угловая скорость колебаний.** Для представления движения маятника средствами 3D-анимации потребуются дополнительные **переменные** – координаты ***x*** и ***y*** материальной точки*: x = Lsin(α); y = Lcos(α).*

Постановка задачи моделирования. Средствами MVS построить модель маятника и установить влияние параметров системы на ее свойства.

Порядок выполнения лабораторной работы.

1. Запуск MVS. После запуска MVS нажмите кнопку  или выполните команду главного меню **Проект/Новый**… (рис. 2.3.2).

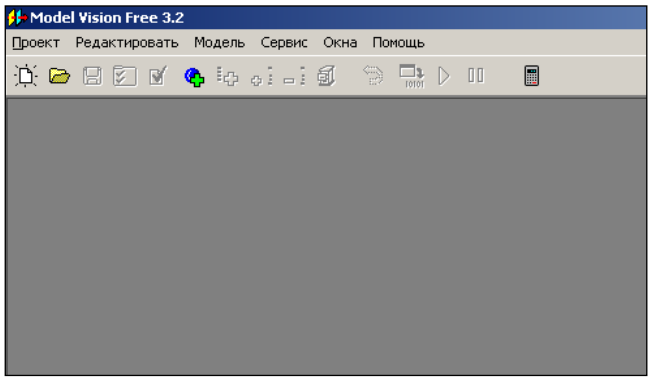


Рис. 2.3.2. Система MVS при первоначальном запуске

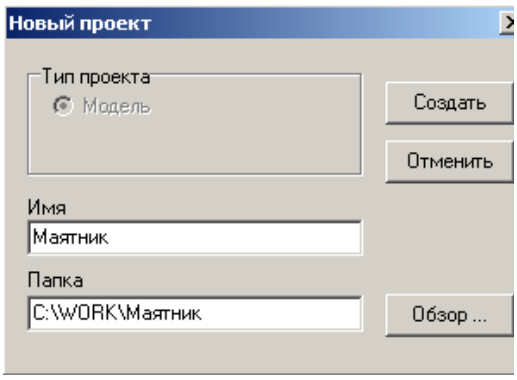


Рис. 2.3.3. Пример создания нового проекта

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 «Маятник.mvb». На рис. 2.3.3 для создания проекта выбрана папка WORK на диске C:, имя проекта – **«Маятник»**. При создании модели выберите доступный вам на запись диск.

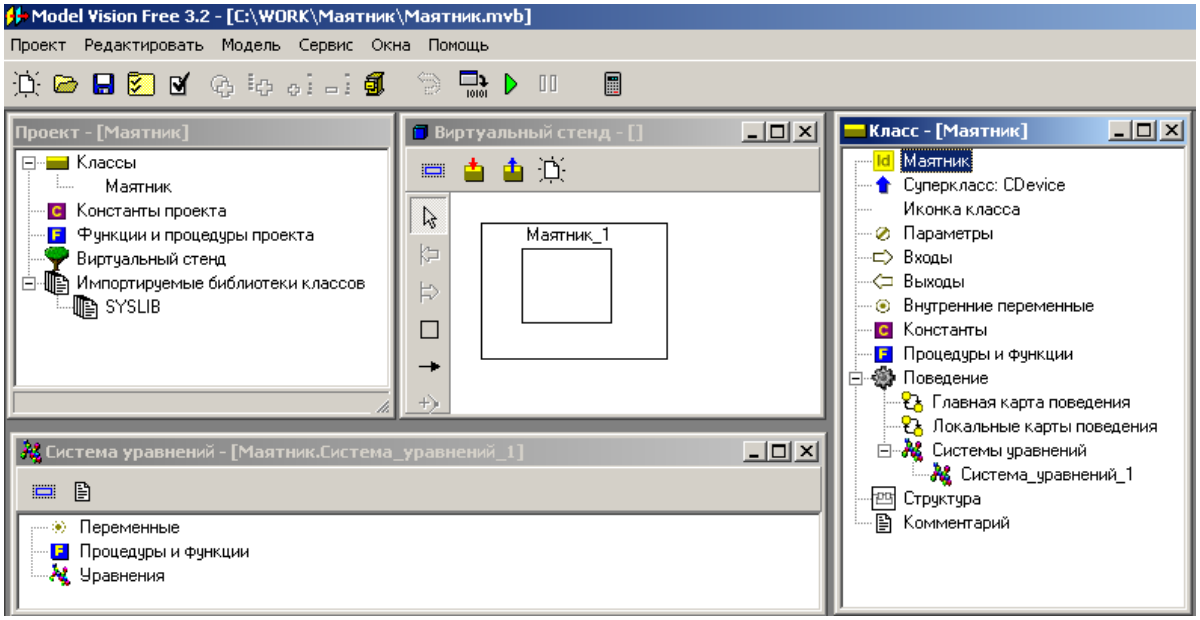


Рис. 2.3.4. «Заготовка» проекта

После этого в среде MVS появятся следующие окна (рис. 2.3.4):

Окно проекта, которое содержит дерево основных составляющих проекта.

Окно виртуального стенда, которое содержит структурную схему моделируемой системы, т.е. экземпляры блоков и связи между ними. По умолчанию в виртуальный стенд помещен экземпляр класса **«Маятник»** с именем **«Маятник_1»**.

Окно класса (**«Маятник»**) 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».

Модель **«Маятник»** – это модель непрерывной системы. В класс, который создан по умолчанию, необходимо добавить соответствующие переменные, параметры, константы и уравнения.

2. Ввод переменных, параметров и констант. В окне класса **«Маятник»** выделяем в дереве объектов узел **«Параметры»**, вызываем контекстное меню и выполняем команду **«Добавить»** (рис. 2.3.5).

В появившемся окне вводим имя ***Alpha0***, оставляем заданный по умолчанию тип ***double***, задаем начальное значение ***– pi/2*** (рис. 2.3.6).

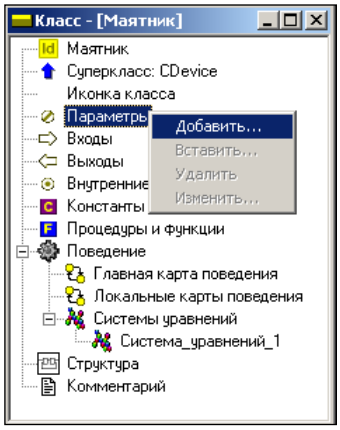


Рис. 2.3.5. Добавление параметра

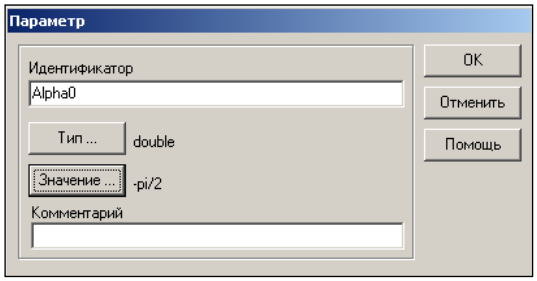


Рис. 2.3.6. Задание имени и значения параметра

Аналогичным образом добавляем параметр ***L***. Далее выделяем узел **«Внутренние переменные»** и добавляем переменные ***Alpha, Omega, X, Y***, а затем выделяем узел **«Константы»** и добавляем константу ***g*** (рис. 2.3.7).

Можно изменить или удалить введенные определения, дважды кликнув на них мышью, с помощью команд контекстного меню.

3. Ввод уравнений модели. В окне **«Класс»–[Маятник]** 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. 2.3.8). Специальный знак производной $\frac{d}{dt}$ вводится с помощью кнопок на панели инструментов (рис. 2.3.8).

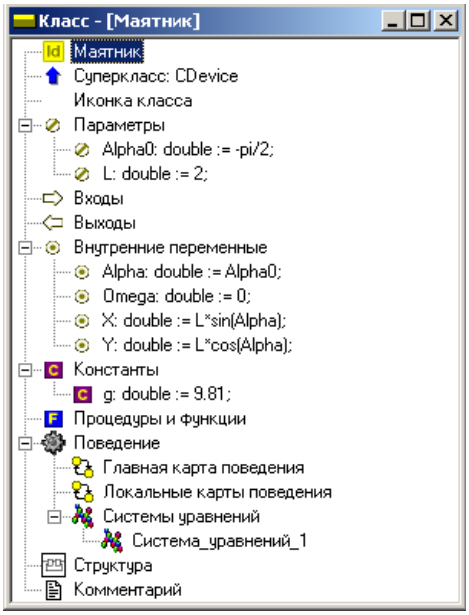


Рис. 2.3.7. Заполненное окно класса «Маятник»

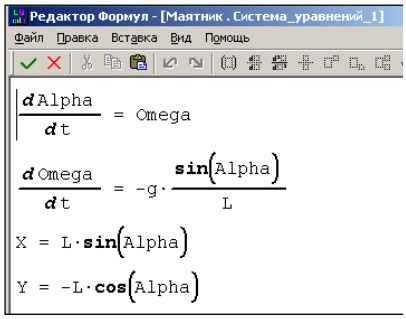


Рис. 2.3.8. Редактирование уравнений

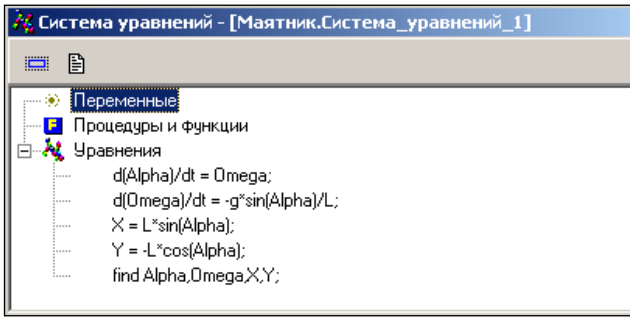


Рис. 2.3.9. Окончательный вид системы уравнений

Система уравнений может включать в себя обыкновенные дифференциальные уравнения первого и второго порядков, а также алгебраические уравнения. После редактирования она примет вид, как показано на рис. 2.3.9.

4. Создание и запуск выполняемой модели. Запуск и создание модели производится с помощью команды **«Модель/Пуск»** главного меню или кнопки **.**

5. Эксперименты с визуальной моделью. На рис. 2.3.10 показано главное окно визуальной модели после первого запуска. Визуальная мо-

дель является многооконным приложением. В левой части инструментальной панели отображается текущее значение модельного времени (начальное значение равно нулю). В левом верхнем углу расположено окно виртуального стенда, которое отражает структуру модели. Для блока **«Маятник_1»** автоматически открывается окно переменных. После создания экземпляра этого устройства его параметры приняли указанные значения, а фазовые переменные инициализированы заданными выражениями.

6. Запуск и рестарт модели. Запустим выполнение модели с помощью кнопки  или с помощью команды **«Моделирование/Пуск»** главного меню. При этом начнет изменяться модельное время и значения фазовых переменных. Останов выполнения модели производится с помощью кнопки  или команды **«Моделирование/Стоп»**.

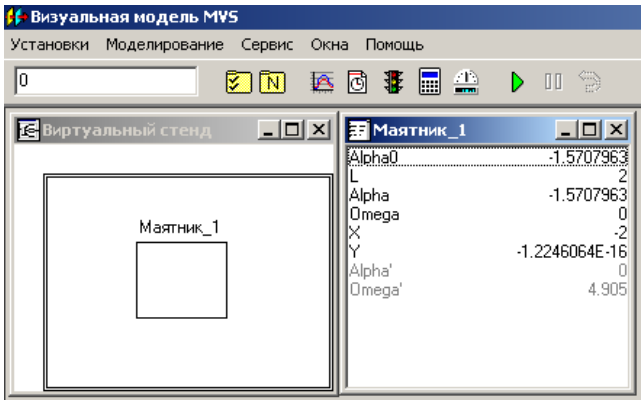


Рис. 2.3.10. Главное окно визуальной модели

Возврат модели в начальное состояние производится с помощью кнопки  или команды **«Моделирование/Рестарт»**.

7. Построение временной и фазовой диаграмм. С помощью кнопки **«Новая диаграмма»** –  или команды **«Окна/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 **«Маятник_1»** переменные ***Alpha*** и ***Omega***. Запустим модель и получим следующий график (рис. 2.3.11).

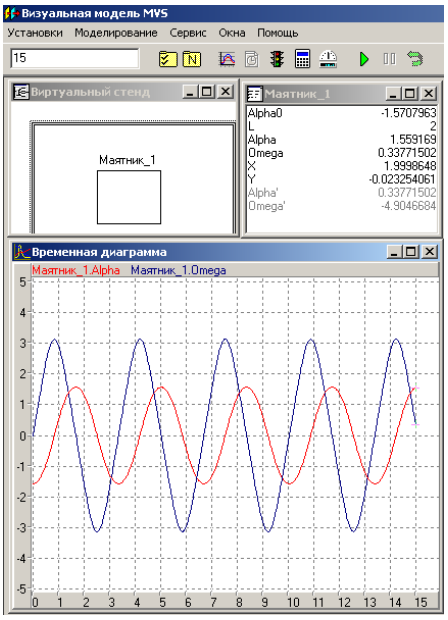


Рис. 2.3.11. Модель и временная диаграмма процесса

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 странице **«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. 2.3.12). Здесь же можно задать время останова прогона модели и другие установки.

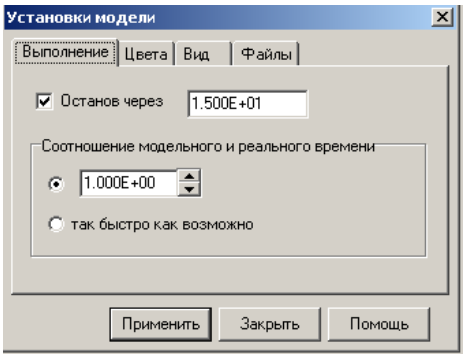


Рис. 2.3.12. Окно «Установки модели»

Временную диаграмму можно преобразовать в фазовую (рис. 2.3.13–2.3.14)

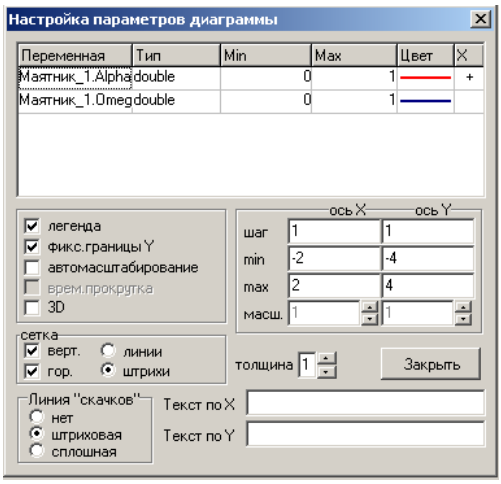


Рис. 2.3.*13. Настройка параметров фазовой диаграммы*

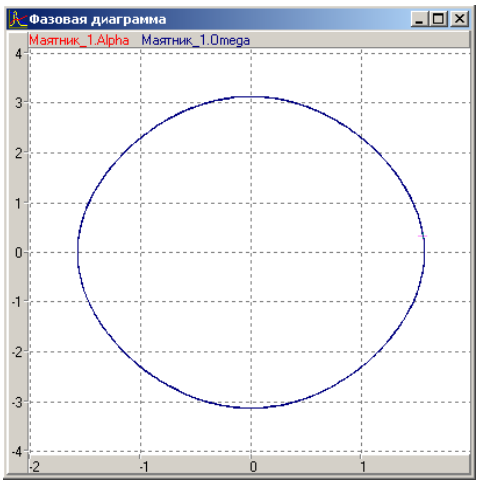


Рис. 2.3.*14. Фазовая диаграмма*

8. Применение 3D-анимации. Для моделей механических систем можно получить больше информации при непосредственном наблюдении поведения трехмерного изображения моделируемой системы. 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: **отрезок** (**line**) и **сфера** (**sphere**, рис. 2.3.15).

Один конец линии должен всегда находиться в начале координат (параметры ***x1 = 0, y1 = 0***), а координаты второго конца (параметры ***x2, y2***) должны изменяться в соответствии со значением переменных ***X*** и ***Y*** модели маятника. Для задания этого соответствия «перетащим» необходимые переменные из **окна переменных** и бросим их в колонке **«Переменная»** соответствующих параметров отрезка (рис. 2.3.15). Аналогичным образом

этим же переменным ***X, Y*** мы сопоставляем координаты центра сферы (параметры ***x1, y1***).

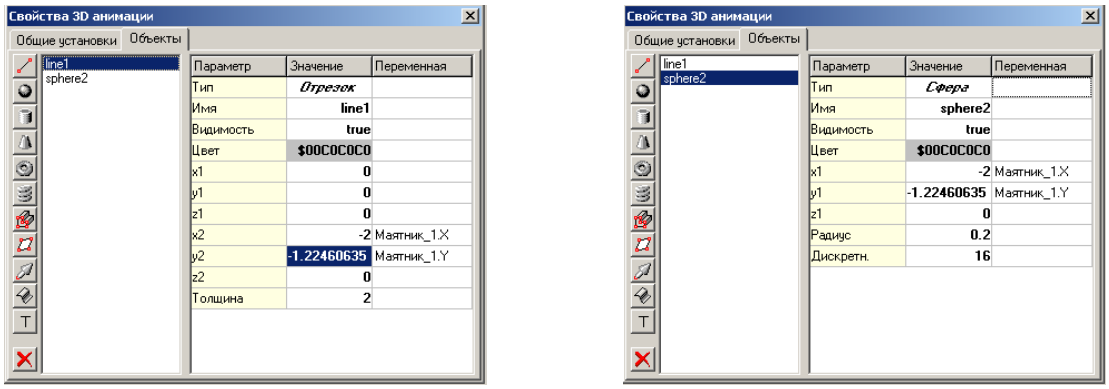


Рис. 2.3.15. Параметры 3D-объектов

После чего достаточно запустить модель и вы увидите качающийся маятник (рис. 2.3.16). В любой момент вы можете изменить точку наблюдения, нажав левую клавишу мыши и перемещая ее с прижатой клавишей. Таким образом, вы можете рассматривать колебания маятника сверху, снизу и т.д.

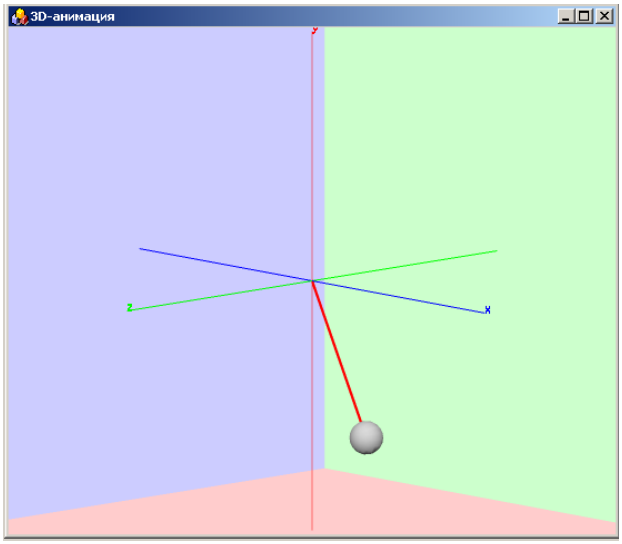


Рис. 2.3.16. Представление результатов моделирования средствами 3D-анимации

Анализ результатов моделирования. Установить влияние параметров объекта на период колебаний маятника.

Дополнительные задания. В среде MVS построить модель вынужденных колебаний математического маятника под действием внешней периодической силы. Результаты моделирования отобразить в виде временной и фазовой диаграммы для ***X*** и ***Y***.

$$\frac{d\alpha}{dt} = \omega, \quad \frac{d\omega}{dt} = -g \cdot \frac{\sin\alpha}{L} - \sin(\theta t).$$

$$X = L \cdot \sin(\alpha), \quad Y = -L \cdot \cos(\alpha).$$

$$\omega(t=0) = 0; \quad \alpha(t=0) = \pi/4; \quad \theta = 1.5; \quad L = 3.$$

В среде MVS построить модель маятника Фуко.

$$\frac{dVx}{dt} = 2 \cdot Vy \cdot \omega + \omega^2 \cdot x - g\frac{x}{L}; \quad \frac{dVy}{dt} = -2 \cdot Vx \cdot \omega + \omega^2 \cdot y - g\frac{y}{L};$$

$$\frac{dx}{dt} = Vx; \quad \frac{dy}{dt} = Vy.$$

Значения параметров (два варианта):

$$L = 100; \quad \omega = 0.04; \quad g = 9.81; \quad x(t=0) = 5; \quad y(t=0) = 5; \quad Vx(t=0) = 0; \quad Vy(t=0) = 0.$$

$$L = 50; \quad \omega = 0.04; \quad g = 9.81; \quad x(t=0) = 2; \quad y(t=0) = 2; \quad V_x(t=0) = 0; \quad V_y(t=0) = 0.$$

Какое явление подтверждает маятник Фуко?

2.4. Гибридная модель «Отрывающийся маятник»

Данная модель будет построена по аналогии с моделью из работы «Моделирование колебаний маятника».

Постановка задачи моделирования. Построить модель маятника, который отрывается от нити в определенной фазе колебаний. Моделируемая система является гибридной, т.к. она имеет два различных непрерывных поведения: колебания и свободный полет.

Порядок выполнения работы. Новый проект создадим на основе разработанного ранее проекта **«Маятник.mvb»**. Для этого с помощью команды **«Проект/Сохранить как»** сохраним этот проект как **«ОтрывающийсяМаятник.mvb»** и далее будем редактировать уже существующий класс **«Маятник»** (рис. 1.4.1).

Новая модель – модель изолированной системы с несколькими качественными состояниями: **«Колебания», «Свободный полет», «Останов»**. Каждое из состояний подчиняется своим законам. Для описания переходов из одного состояния в другое потребуется создание **«Карты поведения»**. Процесс ее создания будет подробно описан ниже.

Предварительно внесем изменения в **«Класс–[Маятник]»** (рис. 1.4.1). Добавим новые параметры и переменные, которые будут необходимы в дальнейшем.

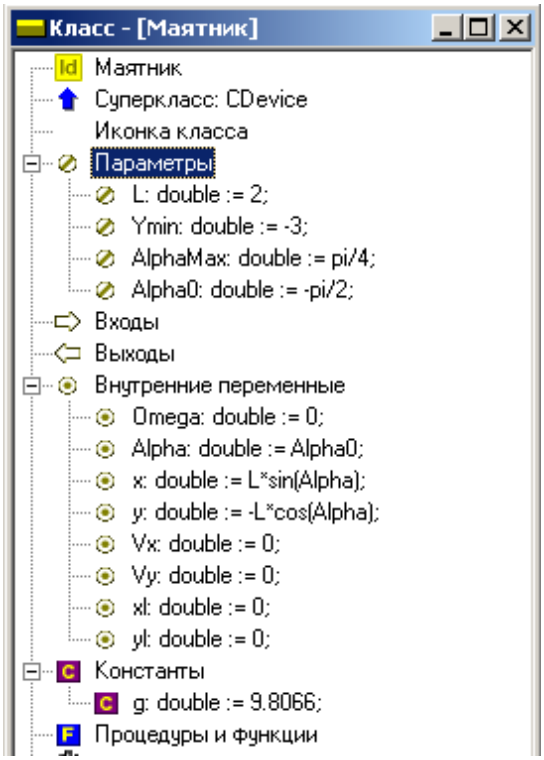


Рис. 2.4.1. Описание нового класса – «ОтрывающийсяМаятник»

Новые параметры: ***Ymin, AlphaMax***. Новые переменные: ***Vx, Vy, xL, yL.***

В окне редактирования класса выделим главную **карту поведения** и с помощью контекстного меню выполним команду **«Изменить»** (рис. 2.4.2).

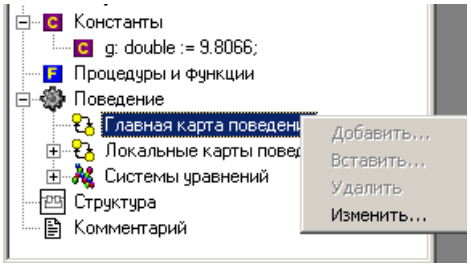


Рис. 2.4.2. Открытие главной карты поведения

Выделим узел **Init** в главной карте поведения (рис. 2.4.3) и с помощью кнопки  или команды **«Установить пустое локальное поведение»** контекстного меню сделаем узел **Init** пустым.

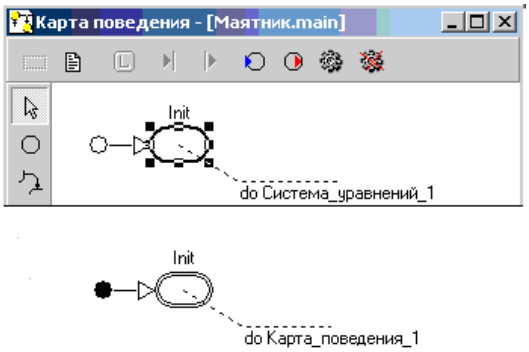


Рис. 2.4.3. Главная «Карта поведения» и ее преобразование

Учитывая, что узел **Init** унаследован от родительского класса **«CDevice»** и ни удалить, ни переименовать его нельзя, можно изменить главную карту поведения гибридной модели следующим образом (рис. 2.4.3–2.4.4):

Необходимо построить новую карту поведения (локальную) – **«Карта_поведения_1»** (рис. 2.4.4), которая приписывается узлу **Init** (рис. 2.4.3). Теперь при запуске модели моментально будет выполнен переход из узла **Init** на **Карту_поведения_1**.

Создание карты поведения

Новую карту поведения можно создать двумя способами:

1. Переместить мышь на секцию **«Локальные карты поведения»** в окне класса, вызвать по правой кнопке мыши контекстное меню и выполнить команду **«Добавить»**. В дальнейшем вам придется приписать новую

карту поведения какому-то узлу другой карты поведения более высокого уровня иерархии методом drag-and-drop.

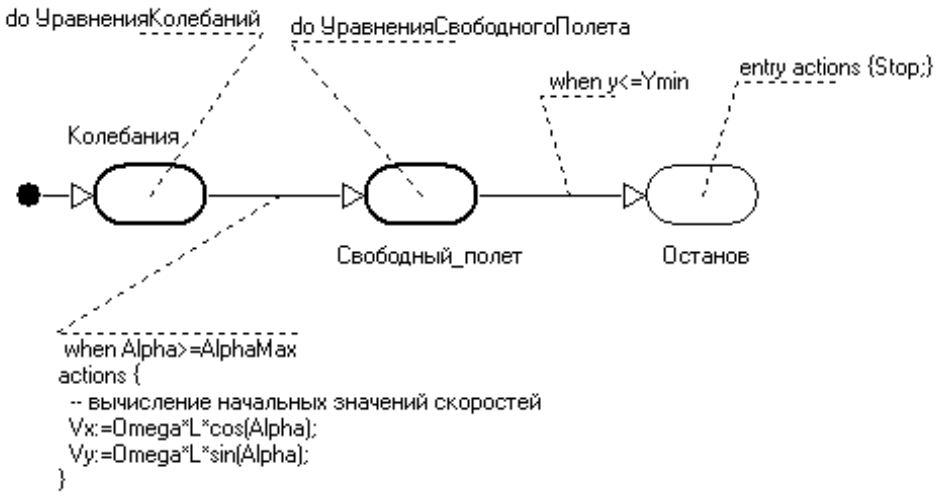


Рис. 2.4.4. Карта_поведения_1

2. Выделить узел, которому хотим приписать новую карту поведения, вызвать правой кнопкой мыши контекстное меню и выполнить команду **«Создать новую карту поведения»**. В результате новая карта поведения будет не только создана (она появится в секции **«Локальные карты поведения»** окна класса), но и будет приписана данному узлу.

Редактирование карты поведения

Карта поведения представляет собой граф, узлы которого соответствуют состояниям моделируемой системы с непрерывным поведением, а дуги – переходам из одного состояния в другое (рис. 2.4.4). Узлам-состояниям соответствуют определенные виды поведения:

- **пустое поведение**, при котором значения переменных не изменяются, соответствующий узел изображается тонкой линией (рис. 2.4.4. узел **«Останов»**):
- **непрерывное поведение**, заданное системой уравнений; соответствующий узел изображается жирной линией;

- **дискретное,** или гибридное, поведение, заданное другой картой поведения; соответствующий узел изображается двойной тонкой линией (рис. 2.4.3).

Переходы изображаются ломаной линией со стрелкой, указывающей направление перехода. Переход, у которого отсутствует начальный или конечный узлы, является незавершенным. Незавершенные переходы изображаются пунктирными линиями и отсутствуют в выполняемой модели. Один из переходов является начальным (вместо исходного узла изображается жирная точка). Он срабатывает сразу при инициализации экземпляра карты поведения и таким образом указывает на начальный узел карты поведения.

Кроме того, в окне карт поведения изображаются:

- **имена узлов**. Для задания имени данного узла необходимо выделить этот узел и нажать кнопку  на левой инструментальной панели;
- сноски, указывающие **входные** и **выходные** действия в узле, **условия и действия перехода.**

Один из элементов карты поведения может быть выделен нажатием левой кнопки мыши на изображении этого элемента. Выделение отображается опорными точками. Выделенный элемент можно перемещать с помощью мыши.

Нажатием левой кнопки мыши при прижатой клавише Shift можно выделить группу элементов. Группу элементов можно переместить с помощью мыши, либо использовать для выравнивания с помощью команды **«Редактировать/Выровнять»** главного меню.

Любое изменение в карте поведения может быть отменено с помощью кнопки  на панели кнопок главного окна или команды **«Редактировать/Отменить изменения»** главного меню.

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

(рис. 2.4.5). Стандартным является режим выделения, переходом в этот режим, как правило, заканчиваются все операции.

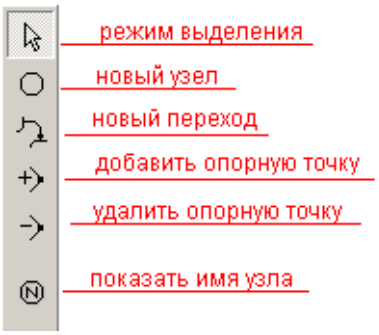


Рис. 2.4.5. Левая панель кнопок редактирования карты поведения

Верхняя панель содержит кнопки, связанные с редактированием свойств выделенного элемента или карты поведения в целом (Таблица 2.4.1).

Таблица 2.4.1

Назначение кнопок редактирования «Карты поведения»

Кнопка	Назначение
	Переход в диалог редактирования имени объекта
	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

Редактирование узлов карты поведения

Чтобы создать новый узел, нужно:

- С помощью кнопки  или команды **«Создать новый узел»** всплывающего меню 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. В дальнейшем вы можете сделать начальным другой узел, переместив начальный переход так, чтобы его стрелка оказалась на изображении этого узла.

Чтобы удалить узел, выделите его и выполните либо команду **«Удалить узел»** всплывающего меню, либо команду **«Редактировать/Удалить»** главного меню.

Чтобы переместить узел, нажмите на его изображении левую кнопку мыши и, перемещая мышь с прижатой кнопкой, переместите в нужное место, после чего отпустите кнопку. Для более точного перемещения выделенного узла в определенном направлении можно использовать клавиши перемещения курсора на клавиатуре или специальные кнопки в правой части панели кнопок главного окна. Кроме того, вы можете точно задать положение узла с помощью команды всплывающего меню **«Положение»**. Вместе с узлом автоматически перемещаются последние сегменты связанных с этим узлом переходов.

Чтобы изменить размер узла, выделите его, схватите мышью одну из опорных точек, показанных квадратами, и тащите ее до нужного положения.

Чтобы переместить имя узла, перетащите его методом drag-and-drop.

Чтобы изменить имя узла, щелкните по нему дважды мышью или выделите его и выполните команду **«Редактировать/Редактировать как текст»** главного меню. На месте имени появится однострочный текстовый редактор, в котором вы можете отредактировать имя узла. Чтобы отменить редактирование, нажмите Esc, чтобы завершить редактирование, нажмите Enter или щелкните мышью за пределами редактора строки.

Чтобы поместить в узел уже существующее локальное поведение (систему уравнений или карту поведения), выделите соответствующее поведение в окне класса и методом drag-and-drop переместите на изображение узла.

Чтобы поместить в узел новое локальное поведение, выполните команду **«Создать новую систему уравнений»** или команду **«Создать новую карту поведения»** всплывающего меню.

Чтобы удалить из узла приписанное ему локальное поведение, выделите узел и нажмите кнопку  или выполните команду **«Установить пустое локальное поведение»** всплывающего меню.

Руководствуясь этими правилами, в классе **«Маятник»** с помощью команды **«Добавить»** создадим новую карту поведения **«Карта_поведения_1»** (раздел **«Локальные карты поведения»** по рис. 2.4.2), кликнув дважды мышью на ее названии, выполним команду **«Изменить»** и перейдем в окно редактора карт поведения, введем в ней три новых узла. Изменим имена узлов: **«Колебания»**, **«Свободный** п**олет»** и **«Останов»** (рис. 2.4.6).

Рис. 2.4.6. Создание узлов карты поведения

С целью улучшения внешнего вида карты поведения выделим все три узла как группу (это делается левой кнопкой мыши при прижатой клавише Shift) и выполним команду **«Редактировать/Выровнять»** главного меню. В появившемся диалоге (рис. 2.4.7) выберем выравнивание по верхним сторо-

нам. Нажмем кнопку «ОК» и выполним эту же команду снова, выбрав на этот раз выравнивание по ширине и высоте. В результате мы получим три совершенно одинаковых по размерам узла, расположенных на одной линии.

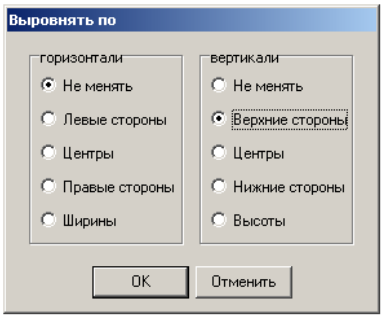


Рис. 2.4.7. Окно параметров операции выравнивания

Напомним, **что любое изменение в карте поведения** может быть отменено **с** помощью кнопки на панели кнопок главного окна или команды **«Редактировать/Отменить изменения»** главного меню.

Все созданные узлы карты поведения по умолчанию содержат пустые локальные поведения, то есть при нахождении модели в этом состоянии в ней ничего не будет меняться.

Переименуем **«Система_уравнений_1»** – **«Уравнения_колебаний».** Эта система, описывающая колебания маятника, уже существует в классе **«Маятник»**, и мы просто методом drag-and-drop перетащим ее из окна **класса** на узел **«Колебания»**.

Для обеспечения корректной работы 3D-анимации модернизируем эту систему уравнений (рис. 2.4.8):

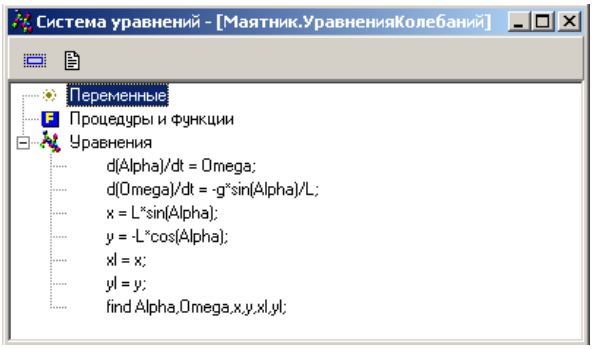


Рис. 2.4.8. Модернизированная система уравнений колебаний маятника

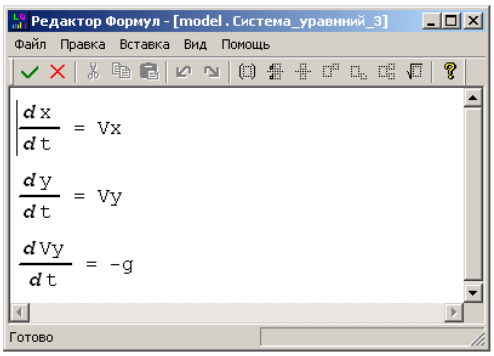


Рис. 2.4.9. Система уравнений свободного полета

Систему уравнений, описывающую свободный полет с именем **«Уравнения_свободного_полета»**, необходимо создать (рис. 2.4.9), а затем присвоить узлу **«Свободный_полет»** путем перетаскивания.

Узел **«Останов»** должен содержать в своих входных действиях вызов предопределенной процедуры **«Stop».** Выделим этот узел и затем с помощью кнопки  или команды **«Входные действия в узле»** всплывающего меню перейдем в окно редактирования последовательности входных действий (рис. 2.4.10).

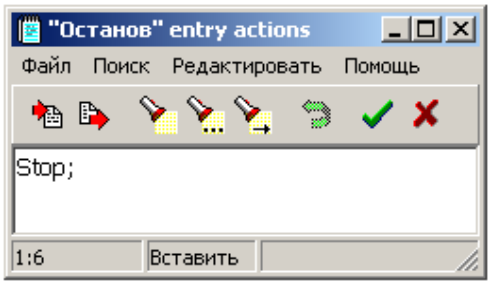


Рис. 2.4.10. Окно редактора входных действий в узле

Последовательность мгновенных действий (входных действий в узле, выходных действий в узле или действий в переходе) – это последовательность операторов, заданная в текстовой форме. Набор допустимых алгоритмических операторов (подмножество языка программирования ADA) описан в разделе справочной системы **«Входной язык/Алгоритмические операторы»**. В данном случае последовательность включает один единственный оператор – обращение к процедуре **Stop**. Теперь создадим переходы между узлами.

Редактирование переходов

Чтобы создать завершенный переход, нужно:

- с помощью кнопки  или команды **«Создать новый переход»** всплывающего меню перевести редактор в режим создания нового перехода (признаком этого режима служит появление крестообразного курсора мыши);
- подвести курсор мыши на изображение исходного узла (например, **«Колебания»**). 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, **«Полет»**). Когда курсор при этом сменится на изображение креста в круге, отпустить кнопку. Исходный и конечный узлы могут быть одним и тем же.

Так получаем переход из узла **«Колебания»** в узел **«Свободный полет».** Точно таким же образом создаем переход из узла **«Свободный полет**» в узел **«Останов»** (рис. 2.4.11).

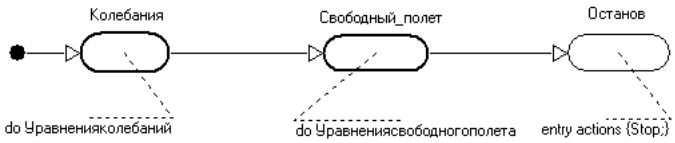


Рис. 2.4.11. Создание переходов в карте поведения

Чтобы изменить линию перехода, нужно выделить переход, нажать левую кнопку мыши на квадрате, изображающем одну из опорных точек (не крайних), затем, не отпуская кнопки, перемещать мышь и в требуемом положении отпустить кнопку.

Чтобы изменить исходный или конечный узел перехода, нужно перетащить мышью соответствующую крайнюю опорную точку на изображение другого узла. Если перетащить ее на свободное поле, можно сделать переход незавершенным.

Чтобы добавить новую опорную точку на линию перехода, нужно выделить переход и с помощью кнопки  или команды всплывающего меню **«Добавить опорную точку»** перевести редактор в режим добавления опорной точки (признаком этого режима служит появление крестообразного курсора мыши). Затем следует подвести курсор к нужной точке линии перехода (курсор изменится на крест с квадратом в центре) и щелкнуть левой кнопкой мыши.

Чтобы удалить опорную точку с линии перехода, нужно выделить переход и с помощью кнопки  или команды всплывающего меню **«Уда-**

лить опорную точку» перевести редактор в режим удаления опорной точки (признаком этого режима служит появление крестообразного курсора мыши). Затем следует подвести курсор к опорной точке, которую следует удалить (курсор изменится на изображение руки с вытянутым указательным пальцем) и щелкнуть левой кнопкой мыши.

Чтобы создать незавершенный переход, нужно в режиме создания нового перехода нажимать или отпускать кнопку вне изображения какого-либо узла, когда курсор имеет форму креста. Завершенный переход можно сделать незавершенным, если «оттащить» крайнюю точку линии перехода от изображения узла и бросить ее на свободном поле.

Все введенные нами переходы по умолчанию являются **безусловными**, то есть они сработают немедленно, как только исходный узел станет текущим.

Чтобы этого не случилось, для переходов должны быть определены **условия срабатывания.** Выделим переход **«Колебания» → «Полет»** и с помощью кнопки  или команды **«Условие срабатывания перехода»** всплывающего меню перейдем в диалог (рис. 2.4.12), в котором введем необходимое условие.

Кроме условия в этом переходе имеются еще мгновенные действия, выполняемые при срабатывании перехода, – расчет начальных значений составляющих скорости при отрыве маятника. Выделим переход и с помощью кнопки  или команды всплывающего меню **«Действия перехода»** вызовем **«Редактор последовательности действий»** (рис. 2.4.13).

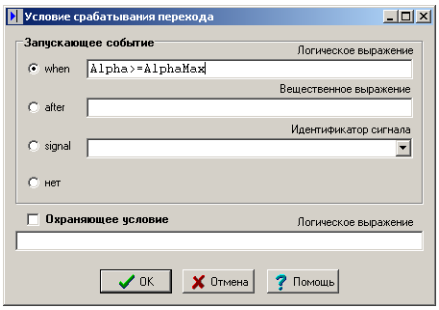


Рис. 2.4.12. Задание условия срабатывания перехода

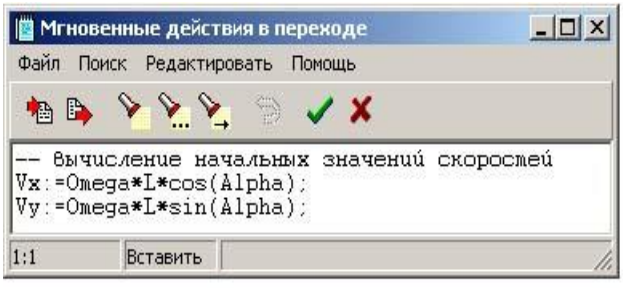


Рис. 2.4.13. Редактор действий в переходе

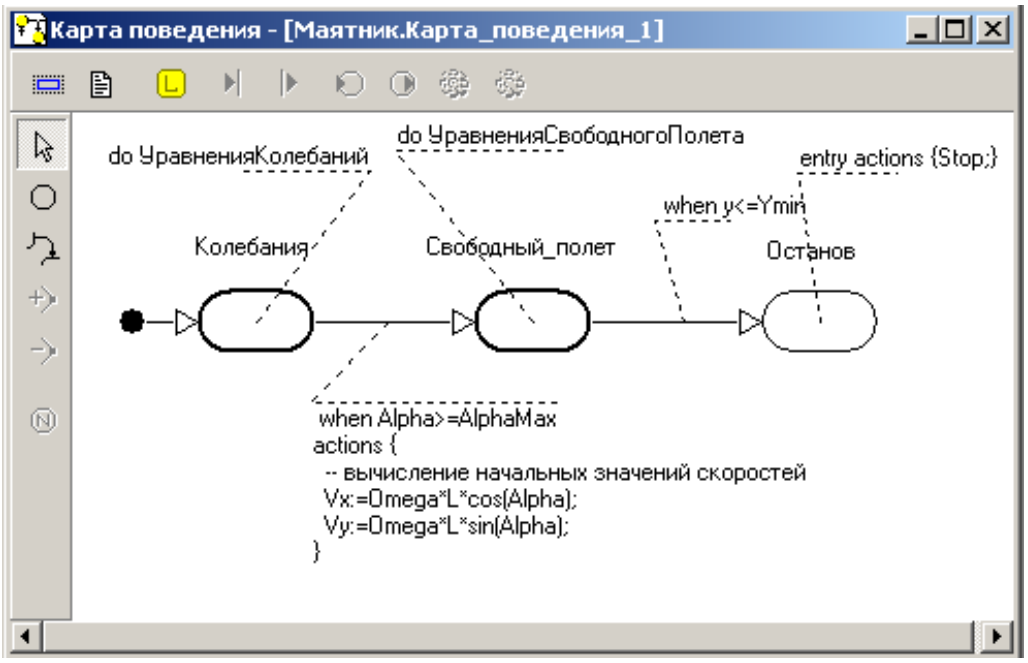


Рис. 2.4.14. Карта_поведения_1

Аналогичным образом введем условие срабатывания **«when y<=Ymin»** для перехода **«Полет» → «Останов».** Окончательно локальная карта поведения будет иметь вид (рис. 2.4.14).

Путем перетаскивания поместим **Карту_поведения_1** (рис. 2.4.14) в узел **Init** главной карты поведения (рис. 2.4.15).

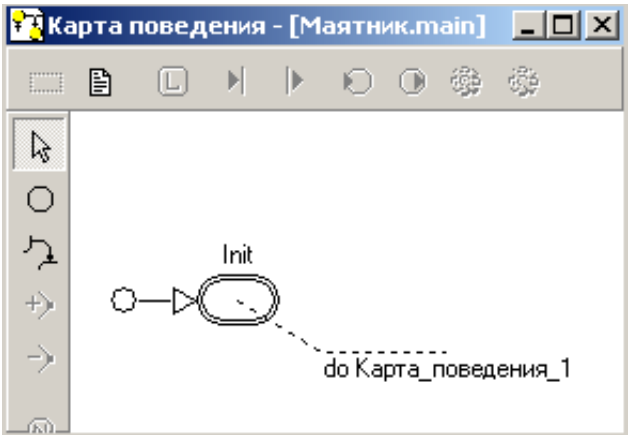


Рис. 2.4.15. Главная карта поведения

Визуальная модель отрывающегося маятника

Запустив модель, мы видим по фазовой диаграмме, что маятник действительно отрывается (рис. 2.4.16).

Рис. 2.4.16. Траектория отрывающегося маятника

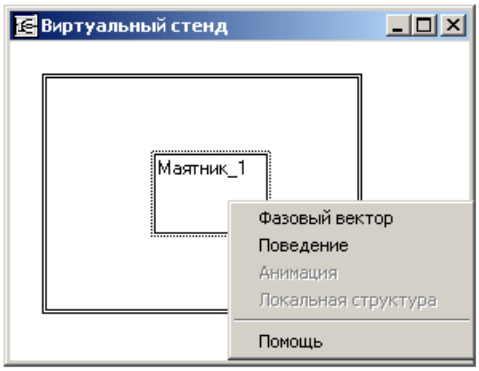


Рис. 2.4.17. Виртуальный стенд

Система MVS позволяет наблюдать в динамике смену качественных состояний модели на карте поведения. Для осуществления наблюдения в окне виртуального стенда щелкнем правой кнопкой мыши на изображении экземпляра маятника **«Маятник_1»** и выполним команду всплывающего меню **«Поведение»** (рис. 2.4.17). В результате выполнения этой команды откроется окно динамической визуализации карты поведения данного блока (рис. 2.4.18).

Рис. 2.4.18. Динамическая визуализация изменения карты поведения

При динамической визуализации карты поведения текущий узел показан сплошной закраской, активные переходы показаны черной линией, неактивные узлы и переходы показаны серыми линиями. Линия срабатывающего перехода подсвечивается на интервал времени, достаточный для фиксации глазом.

3D-анимация колебаний маятника нуждается в уточнении. В окне анимации видно, что после момента отрыва стержень начинает растягиваться, не отрываясь от шарика. Это происходит потому, что в качестве ко-

ординат свободного конца стержня в установках анимации заданы координаты материальной точки.

Снова запустим новую визуальную модель, в установках анимации координаты свободного конца стержня сопоставим переменным ***xL***, ***yL*** и получим, наконец, правильную картинку (рис. 2.4.19).

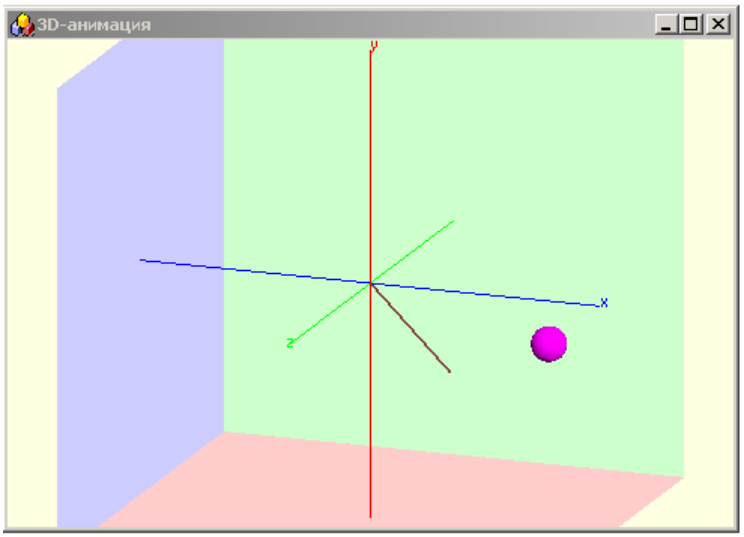


Рис. 2.4.19. 3D-анимация поведения маятника

Анализ результатов моделирования. Выполнить серию компьютерных экспериментов с построенной моделью, в которых следует изменить условия срабатывания перехода при обрыве нити.

2.5. Моделирование движения тела в среде с сопротивлением

В данной работе предстоит построить модель движения тела в среде с сопротивлением. В силу принятых допущений тело рассматривается как материальная точка. Детальное описание модели рассмотрено в п. 2.1.

Модель движения тела построена на основе второго закона механики. Система безразмерных дифференциальных уравнений, описывающая движение тела и начальные условия, имеют вид:

$$\frac{d\bar{V}}{d\bar{t}} = 1 - \bar{k}\bar{V}, \quad \frac{d\bar{x}}{d\bar{t}} = \bar{V}$$

Начальные условия:

$$\bar{x}(\bar{t} = 0) = 0, \quad \bar{V}(\bar{t} = 0) = 1.$$

Модель движения тела представляет собой систему линейных дифференциальных уравнений с постоянными коэффициентами. Подобная задача может быть решена аналитически, но в данной работе применяются численные методы.

Постановка задачи моделирования. Построить модель, на основании которой определить характер изменения скорости движения тела *V(t)* и координаты *X(t)* как функций времени. Объяснить влияние сопротивления на закономерности движения.

Порядок выполнения работы. Следуя рис. 2.5.1–2.5.2, построить модель движения тела средствами MVS. Результаты моделирования отобразить в виде временной диаграммы и 3D-анимации (рис. 2.5.2).

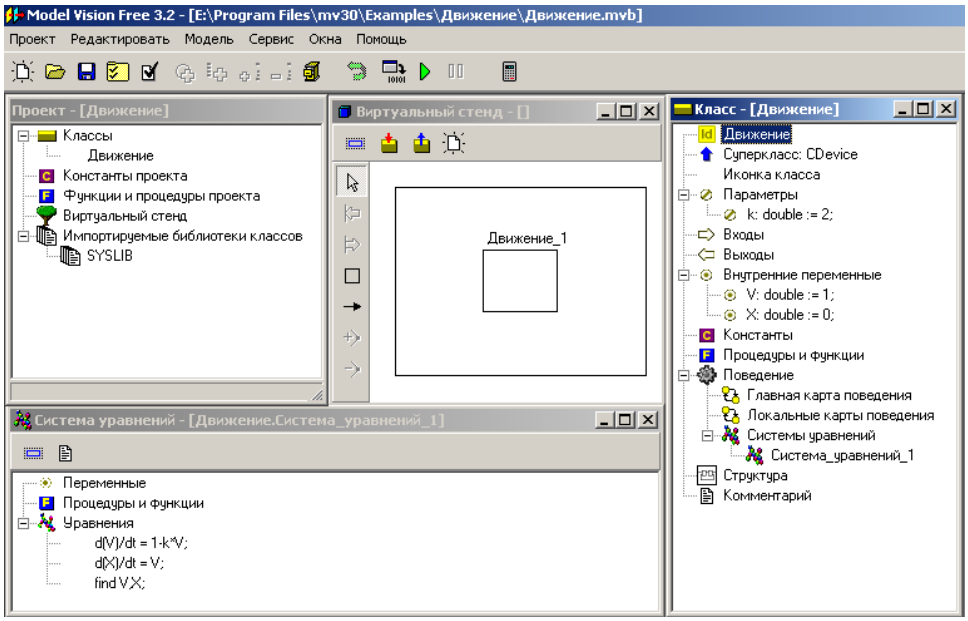


Рис. 2.5.1. MVS-модель движения тела в среде с сопротивлением

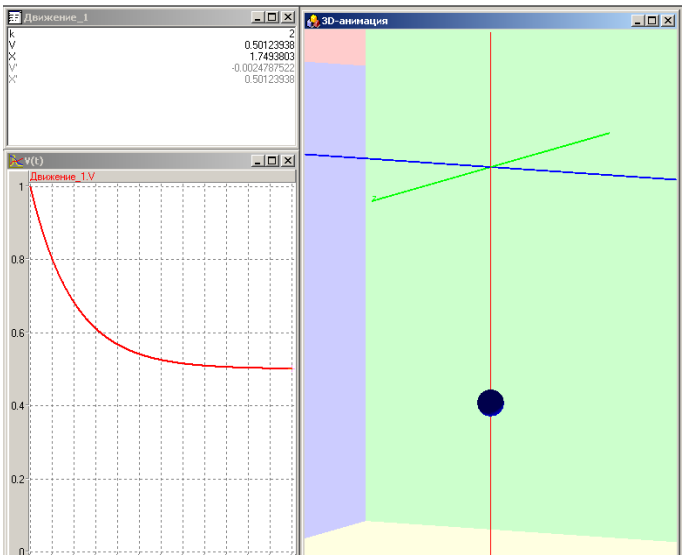


Рис. 2.5.2. Представление результатов моделирования в виде временной диаграммы и средствами 3D-анимации

Дополнительное задание. Как будет протекать процесс движения, если начальное значение скорости *V (t = 0) = – 1*?

Переработайте MVS-модель для квадратичного закона сопротивления. В этом случае система уравнений движения тела будет иметь вид:

$$\frac{d\bar{V}}{d\bar{t}} = 1 - \bar{k}\,|\bar{V}|\cdot\bar{V}, \qquad \frac{d\bar{x}}{d\bar{t}} = \bar{V}, \qquad \bar{k} = \frac{kV_0^2}{mg}$$

Начальные условия:

$$x(t=0)=0,\ V(t=0)=1.$$

Сравнить результат для линейного и квадратичного законов.

2.6. Моделирование движения тела по баллистической траектории

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

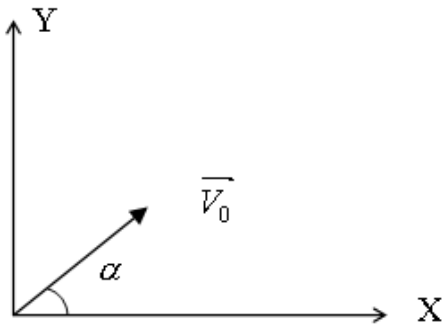


Рис. 2.6.1. Схема начала движения

Модель построена на основе второго закона механики и кинематических уравнений. В безразмерном виде уравнения модели суть следующее:

$$\frac{d\overline{V_y}}{d\bar{t}} = -1 - \bar{k}\overline{V_y}; \qquad \overline{V_x}(\bar{t}=0) = \cos(\alpha);$$

$$\frac{d\overline{V_x}}{d\bar{t}} = -\bar{k}\cdot\overline{V_x}; \qquad \overline{V_y}(\bar{t}=0) = \sin(\alpha);$$

$$\frac{d\bar{x}}{d\bar{t}} = \overline{V_x}, \quad \frac{d\bar{y}}{d\bar{t}} = \overline{V_y}. \quad \bar{x}(\bar{t}=0) = 0, \quad \bar{y}(\bar{t}=0) = 0.$$

Постановка задачи моделирования. Целью моделирования является построение траектории движения в прямоугольной системе координат и выявление влияния сопротивления на траекторию движения. Полученным результатам необходимо дать качественное объяснение.

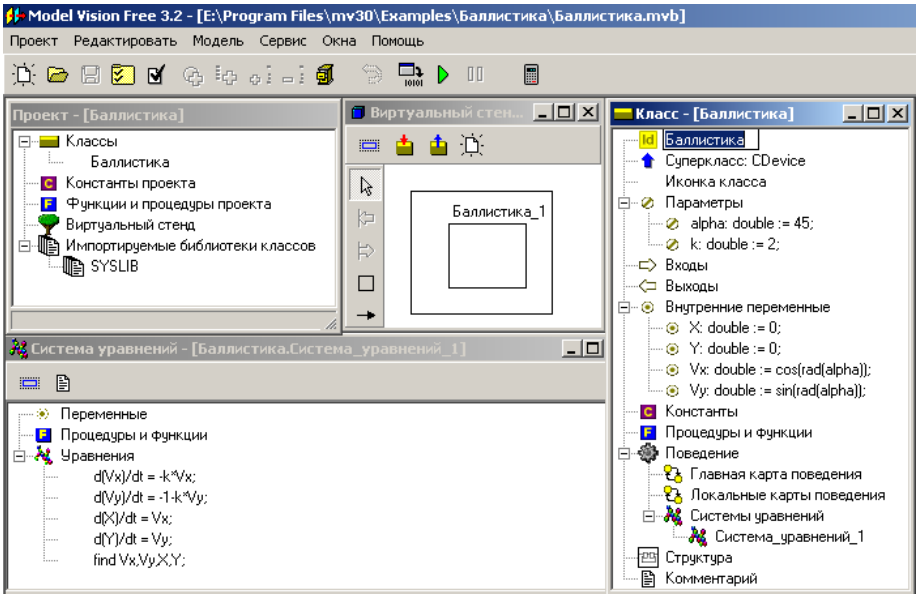


Рис. 2.6.2. MVS-модель движения тела по баллистической траектории

Порядок выполнения работы. Работа выполняется в среде MVS. Провести вычислительный эксперимент для ряда значений коэффициента сопротивления, в первую очередь для *k=0*. В этом случае (*k = 0*) траекторией будет парабола. Этот результат можно использовать для проверки численной модели. Построить MVS-модель (рис. 2.6.2).

Анализ результатов моделирования. По результатам моделирования построить траекторию движения (рис. 2.6.3), установить и объяснить на качественном уровне влияние сопротивления на вид траектории движения тела. При отсутствии сопротивления (*k = 0*) траектория является параболой.

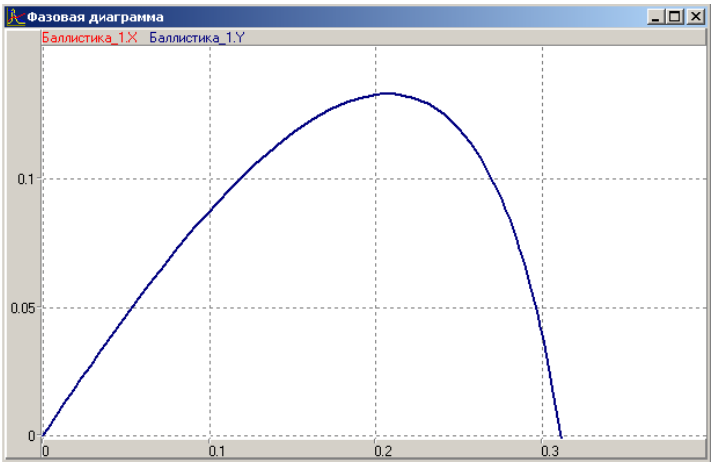


Рис. 2.6.3. Траектория движения тела, k>0

Почему при большом сопротивлении траектория в конце полета становится вертикальной? Объясните либо с точки зрения математики, либо физики. Проведите компьютерный эксперимент с различными значениями параметра *k.*

2.7. Моделирование осциллятора

Моделируемая система представляет собой тело, прикрепленное к невесомой пружине, другой конец которой жестко закреплен (рис. 2.7.1). На тело действует сила упругости пружины и сила трения. Примером по-

добной системы может служить амортизатор автомобиля. Первоначально система выведена из состояния равновесия. Далее система будет совершать затухающие колебания.

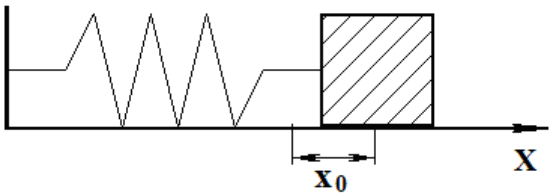


Рис. 2.7.1. Схема объекта моделирования (осциллятора)

Система уравнений движения осциллятора в безразмерной форме будет иметь вид:

$$\frac{d\bar{V}}{d\bar{t}} = -\bar{x} - \bar{k}\cdot\bar{V}, \quad \frac{d\bar{x}}{d\bar{t}} = \bar{V}.$$

Здесь $\bar{k} = \frac{k}{\sqrt{mc}}$ – безразмерный параметр (безразмерный коэффициент трения). Начальные условия:

$$\bar{x}(\bar{t}=0) = 1, \quad \bar{V}(\bar{t}=0) = 0.$$

Безразмерная форма математической модели получена методом неопределенных масштабов. Здесь *m* – масса тела, *x* – координата тела, *V* – скорость движения тела, *c* – жесткость пружины, *k* – коэффициент трения. Значение координаты $x = 0$ соответствует положению равновесия.

Постановка задачи моделирования. Средствами MVS построить модель системы и исследовать влияние значения безразмерного коэффициента трения на ее свойства. Создать 3D-анимацию.

Порядок выполнения лабораторной работы

1. Запуск MVS. После запуска MVS нажмите кнопку  или выполните команду главного меню **Проект/Новый**… (рис. 2.7.2).

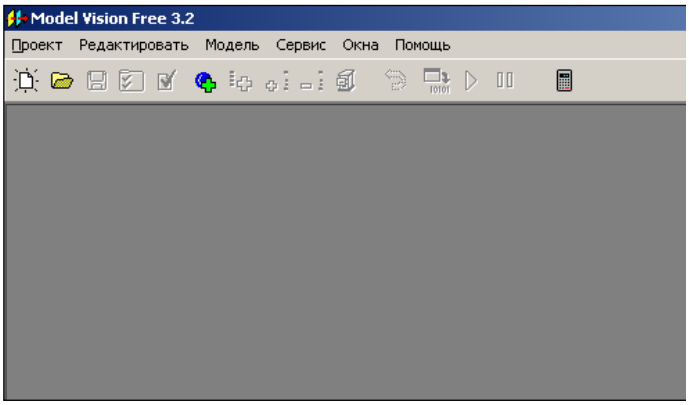


Рис. 2.7.2. Система MVS при первоначальном запуске

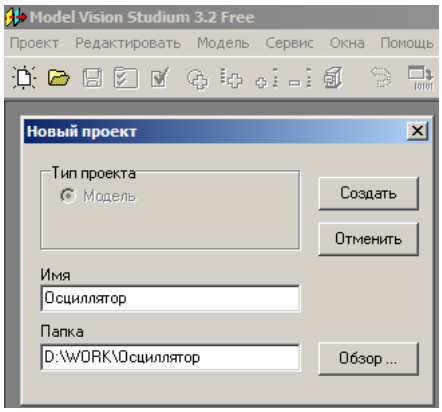


Рис. 2.7.3. Пример создания нового проекта

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. 2.7.3). На рисунке для создания проекта выбрана папка WORK на диске D. Имя проекта – **«Осциллятор»**. При создании модели выберите доступный вам на запись диск.

После этого в среде MVS появятся следующие окна (рис. 2.7.4): **окно проекта, окно виртуального стенда, окно класса, окно системы уравнений «Система_уравнений_1»**. По умолчанию в виртуальный стенд помещен экземпляр класса **«Осциллятор»** с именем **«Осциллятор_1»**.

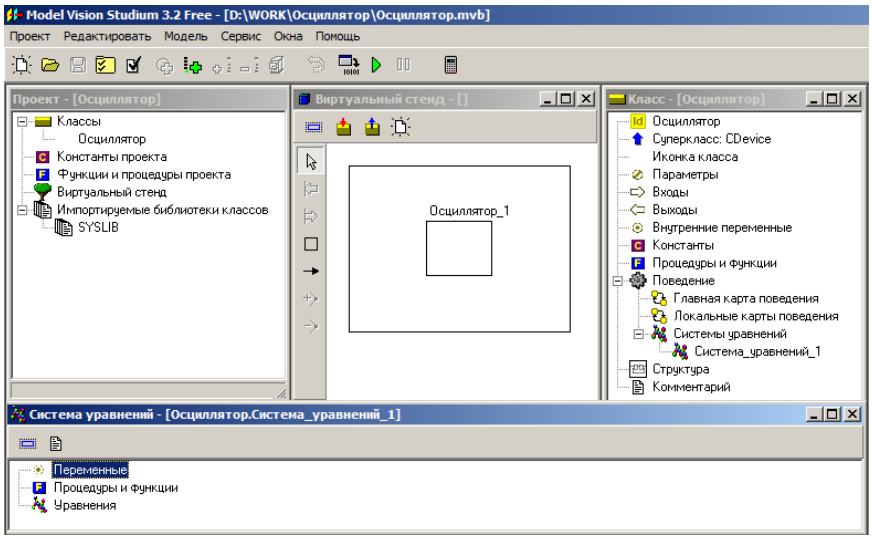


Рис. 2.7.4. «Заготовка» проекта

Модель **«Осциллятор»** – это модель непрерывной системы. Для ее построения подходит класс, создаваемый по умолчанию при открытии нового проекта. В этот класс необходимо добавить соответствующие переменные, параметры, константы и уравнения.

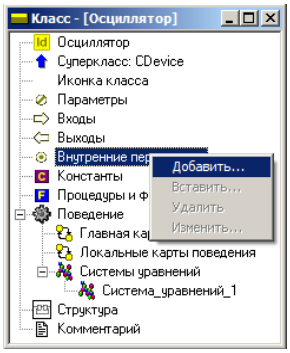


Рис. 2.7.5. Добавление параметра

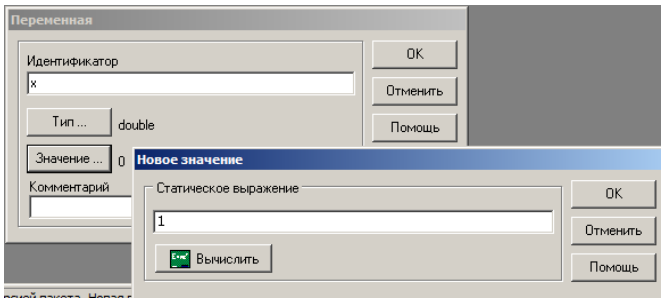


Рис. 2.7.6. Задание имени и значения переменной

В окне класса **«Осциллятор»** выделим в дереве объектов узел **«Внутренние переменные»,** вызовем контекстное меню и выполним команду **«Добавить»** (рис. 2.7.5; 2.7.6)**.** Аналогичным образом добавляем переменные и параметры. На рис. 2.7.7 показано заполненное окно класса **«Осциллятор»**.

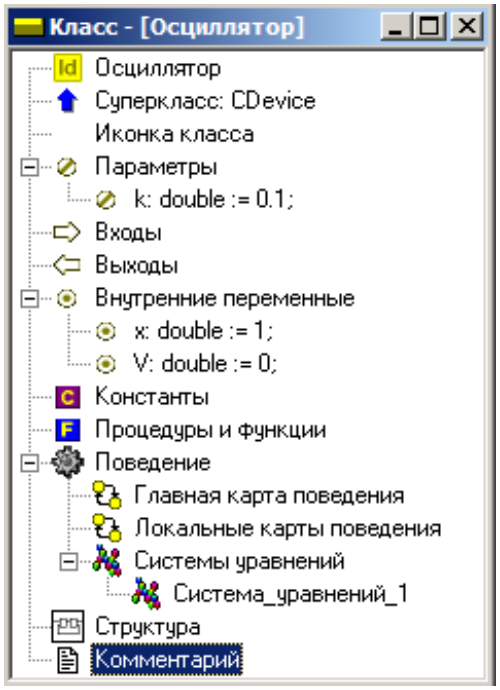


Рис. 2.7.7. Заполненное окно класса «Осциллятор»

3. Ввод уравнений модели. 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. 2.7.8). Специальный знак производной $\frac{d}{dt}$ вводится через пункт меню **«Вставка»** редактора формул или с помощью кнопок на панели инструментов (рис. 2.7.8). После редактирования система уравнений примет следующий вид (рис. 2.7.8; 2.7.9).

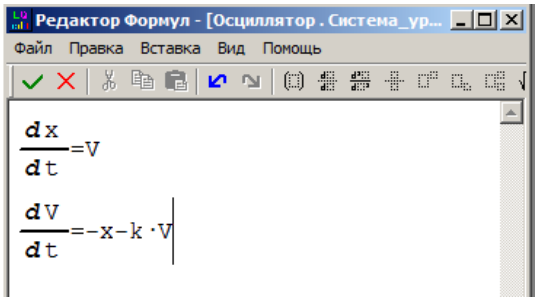


Рис. 2.7.8. Редактирование уравнений

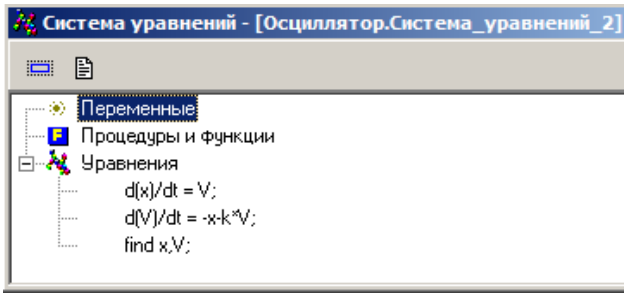


Рис. 2.7.9. Окончательный вид системы уравнений

4. Создание и запуск выполняемой модели. Создание исполняемой модели производится с помощью команды **«Модель/Пуск»** главного меню или кнопки  .

5. Эксперименты с визуальной моделью. На рис. 2.7.10 показано главное окно визуальной модели после первого запуска. Визуальная модель является многооконным приложением. В левой части инструментальной панели отображается текущее значение модельного времени (начальное значение = 0). В левом верхнем углу расположено окно виртуального стенда, которое отражает структуру модели. Для блока **«Осциллятор_1»** автоматически открывается окно переменных. После создания экземпляра этого устройства его параметры приняли указанные значения и фазовые переменные инициализированы заданными выражениями.

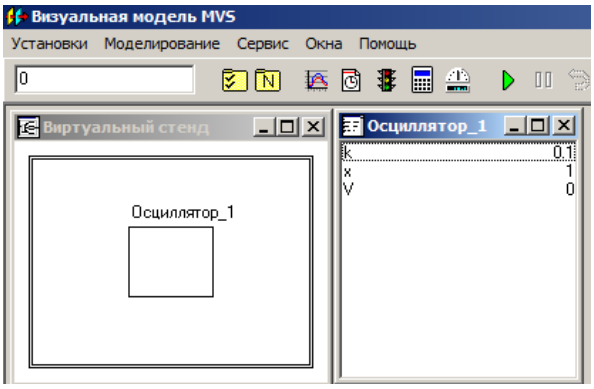


Рис. 2.7.10. Главное окно визуальной модели

6. Запуск и рестарт модели. Запустим выполнение модели с помощью кнопки или с помощью команды **«Моделирование/Пуск»** главного меню. 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»**. В результате этих действий данный экземпляр испытуемой системы будет уничтожен и создан новый, снова с начальными значениями переменных. Модельное время снова равно нулю.

7. **Построение временной и фазовой диаграмм.** С помощью кнопки **«Новая диаграмма»** – или команды **«Окна/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 график (рис. 2.7.11).

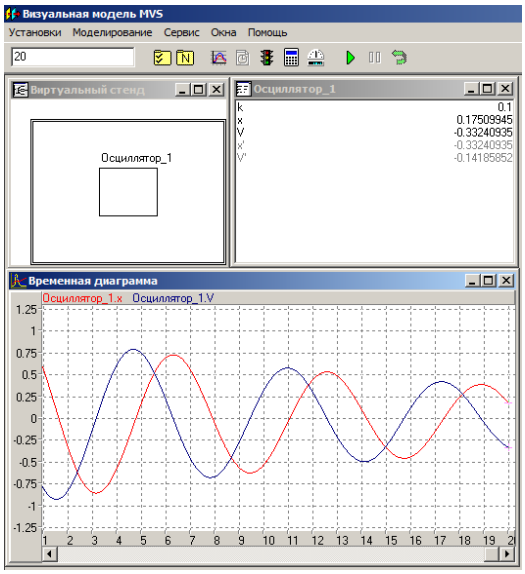


Рис. 2.7.11. Модель и временная диаграмма процесса

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. 2.7.12). Здесь же можно задать время останова прогона модели и другие установки.

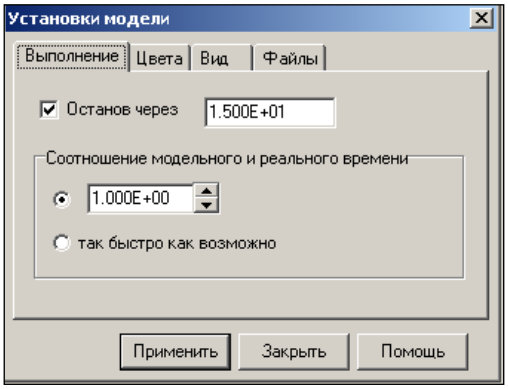


Рис. 2.7.12. Окно «Установки модели»

Теперь 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 поле ***X***, что по оси абсцисс откладываются значения переменной ***x*** (рис. 2.7.13). В этом же окне можно задать и другие параметры диаграммы. Запустив модель, получим следующий график (рис. 2.7.14) – **фазовую диаграмму**.

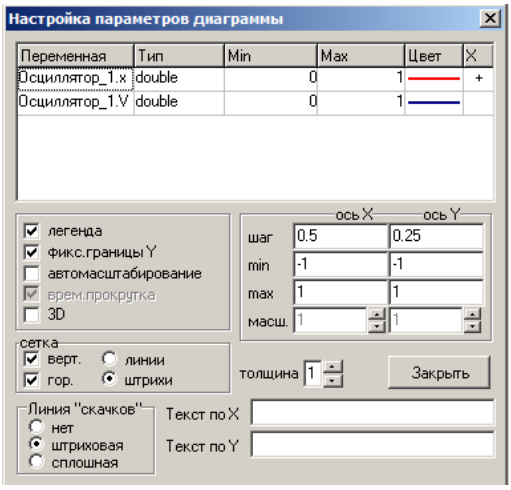


Рис. 2.7.13. Настройка параметров диаграммы

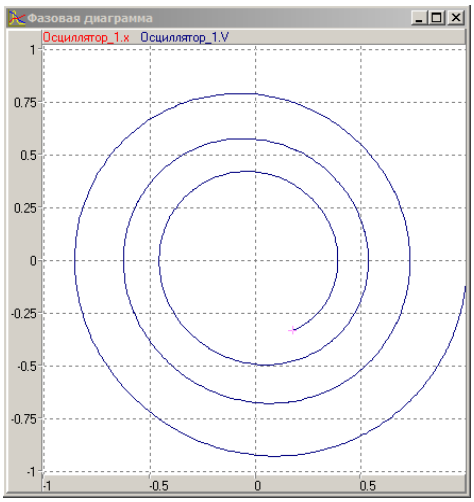


Рис. 2.7.14. Фазовая диаграмма

8. Применение 3D-анимации. Можно получить больше информации из непосредственного наблюдения поведения трехмерного изображения моделируемой системы. 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. 2.7.15).

Один конец пружины должен всегда находиться в начале координат (параметры *x1 = 0, y1 = 0*), а координаты второго конца пружины (пара-

метры *x2, y2*) должны изменяться в соответствии со значением переменных ***X*** и ***Y***. Для задания этого соответствия «перетащим» необходимые переменные из окна переменных и бросим их в колонке **«Переменная»** соответствующих параметров (рис. 2.7.15). Аналогичным образом этим же переменным ***X, Y*** мы сопоставляем координаты центра сферы (параметры ***x1, y1***).

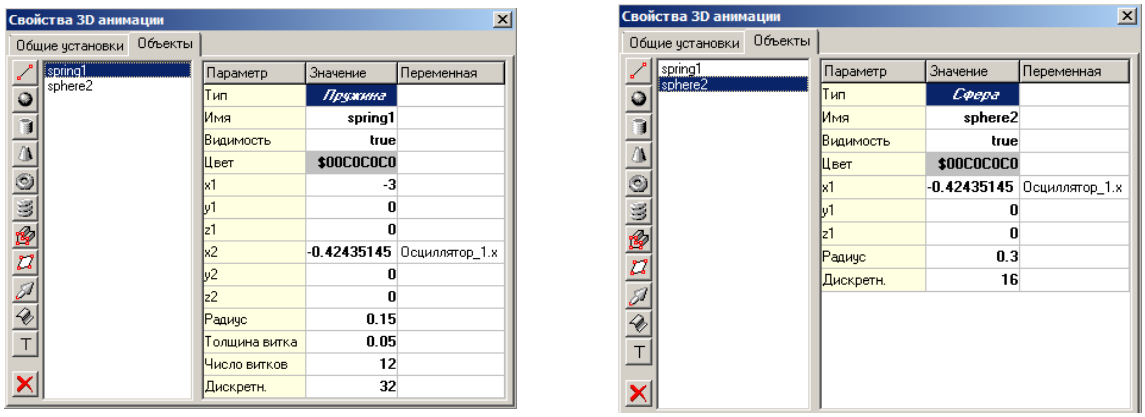


Рис. 2.7.15. Параметры 3D объектов

После чего достаточно запустить модель и вы увидите колебания осциллятора (рис. 2.7.16). 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.

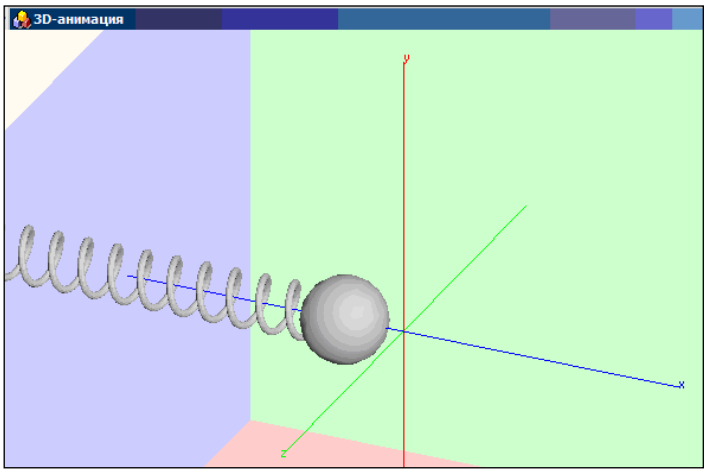


Рис. 2.7.16. Представление результатов моделирования средствами 3D-анимации

Анализ результатов моделирования. Установить влияние параметра ***k***. Первоначальное значение параметра ***k*** задать равным нулю. Установите время затухания колебаний до уровня 5% от начальной амплитуды.

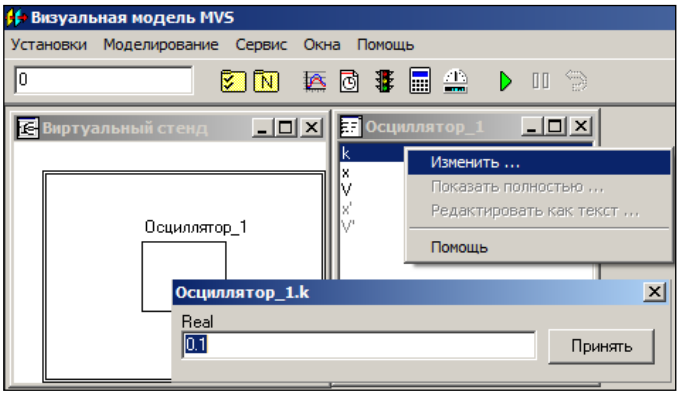


Рис. 2.7.17. Изменение значения параметра

Дополнительное задание. Проведите эксперимент с исходными данными: $x_0 = -1$, $V_0 = 0$, $k = 0{,}1$ (в начальный момент пружина сжата).

2.8. Моделирование системы осцилляторов

Моделируемая система представляет собой систему связанных осцилляторов (рис. 2.8.1). Первоначально система выведена из состояния равновесия (рис. 2.8.1). Далее система будет совершать затухающие колебания.

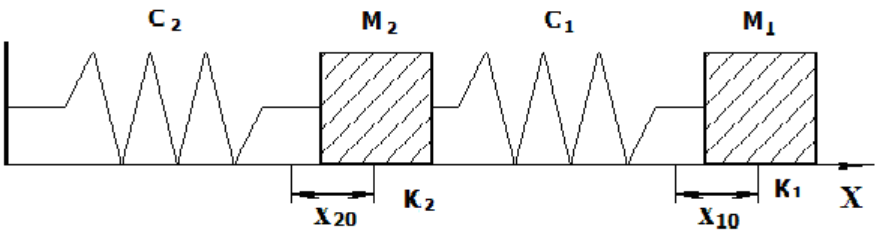


Рис. 2.8.1. Схема системы двух осцилляторов

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

$$\begin{cases} m_1 \dfrac{dV_1}{dt} = -c_1 \cdot (x_1 - x_2) - k_1 \cdot V_1; \quad \dfrac{dx_1}{dt} = V_1; \\ m_2 \dfrac{dV_2}{dt} = c_1 \cdot (x_1 - x_2) - c_2 \cdot x_2 - k_2 \cdot V_2; \quad \dfrac{dx_2}{dt} = V_2; \end{cases}$$

$$x_1(t=0) = x_{10}, \quad x_2(t=0) = x_{20}, \quad V_1(t=0) = 0, \quad V_2(t=0) = 0.$$

Где *m* – масса тела, *x* – координата тела (отклонение от положения равновесия), *V* – скорость движения тела, *c* – жесткость пружины, *k* – коэффициент трения. Значение координаты $x = 0$ соответствует положению равновесия. Модель строится в размерной форме.

Постановка задачи моделирования. Средствами MVS построить модель системы из двух осцилляторов. Построить временную и фазовую диаграммы для координат. Построить 3D-анимацию модели.

Таблица 2.8.1

Значения параметров осцилляторов

N	***X0***	***C***	***k***	***M***
1	5,25	2,75	0,02	1,5
2	1	1,25	0,05	2,5

Анализ результатов моделирования. Установить влияние параметров на свойства системы путем варьирования их значений. Построить модели для каждого осциллятора отдельно и оцените эффект взаимного влияния.

2.9. Моделирование системы регулирования

Системы регулирования предназначены для поддержания определенного режима функционирования объекта или его заданного состояния.

Системы управления с обратной связью учитывают состояние объекта при регулировании (рис. 2.9.1).

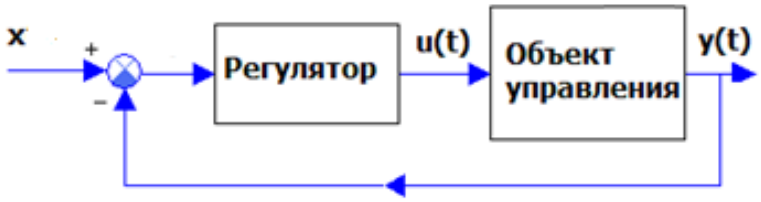


Рис. 2.9.1. Схема системы управления с обратной связью

Здесь: *x(t)* – управляющее воздействие, *e(t) = x(t) – y(t)*, *u(t)* – воздействие регулятора на объект управления, *y(t)* – регулируемый параметр.

Пусть математическая модель объекта управления имеет вид:

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

Модель регулятора суть следующее:

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

Модель всей системы управления описывается следующей системой дифференциальных уравнений:

$$\frac{dy(t)}{dt} = V(t); \qquad \frac{dV(t)}{dt} = -b \cdot y(t) - a \cdot V(t) + k \cdot u(t);$$

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

Начальные условия: *y (t = 0) = 0; V (t = 0) = 0.*

Постановка задачи моделирования. Средствами MVS построить модель системы управления (рис. 2.9.1). Построить временную диаграмму переходного процесса для переменной ***y(t)*** и воздействия регулятора ***u(t)***. Построить MVS-модель такого же объекта, на который не действует управление ***(u(t) = 0)***.

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

Анализ результатов моделирования. Выявить результат применения управления. Исследовать влияние значения параметра *k* на поведение системы. Выявить роль параметра *x.*

2.10. Моделирование звеньев систем регулирования

Системы регулирования включают в свой состав типовые звенья. Цель данной работы построить модели переходных процессов для интегрирующего, инерционного и колебательного звеньев (рис. 2.10.1).

Математические модели звеньев суть следующее:

1. Интегрирующее звено:

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

2. Инерционное звено:

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

Параметры: *m = 1, c = 1*.

3. Колебательное звено:

$$m \cdot \frac{d^2 y}{dt^2} + k \cdot \frac{dy}{dt} + c \cdot y = x \quad .$$

Последнюю модель удобнее представить в виде системы уравнений:

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

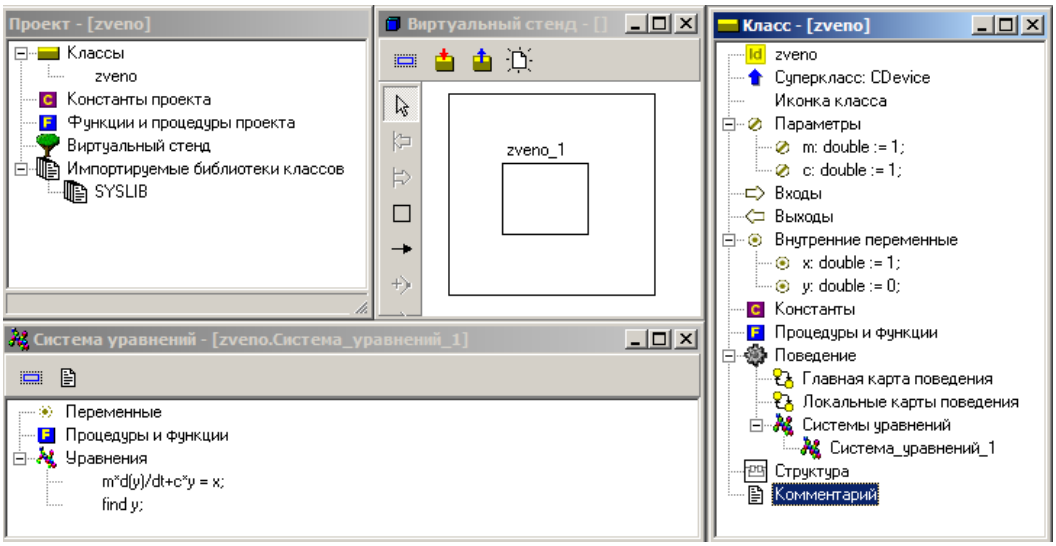


Рис. 2.10.1. MVS-модель инерционного звена

Задание для моделирования

Для каждой модели построить временную диаграмму (рис. 2.10.2) и провести анализ влияния параметров на свойства переходных процессов.

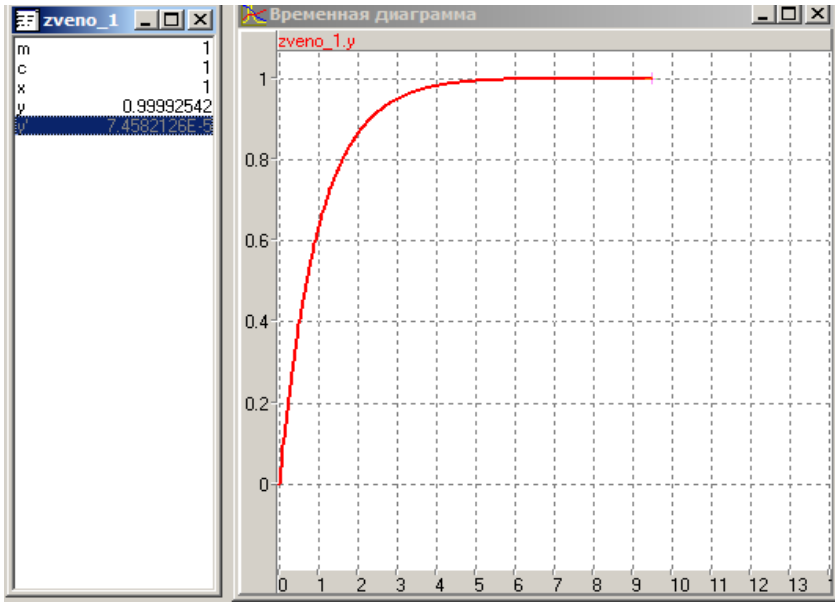


Рис. 2.10.2. Временная диаграмма переходного процесса

Исследовать поведение при *x = sin (w*time),* определить резонансную частоту для колебательного звена.

$$\frac{dy}{dt} = V; \quad m \cdot \frac{dV}{dt} + k \cdot V + c \cdot y = x;$$

$$y(t=0) = 0; \quad V(t=0) = 0; \quad m = 1, \quad k = 1, \quad c = 1.$$

2.11. Моделирование полета искусственного спутника Земли

Модель построена на основе закона всемирного тяготения и второго закона механики. Расчетная схема представлена на рис. 2.11.1. Будем считать, что в начальный момент времени космический летательный аппарат (КЛА) выводится ракетой носителем высоту *H* от поверхности Земли.

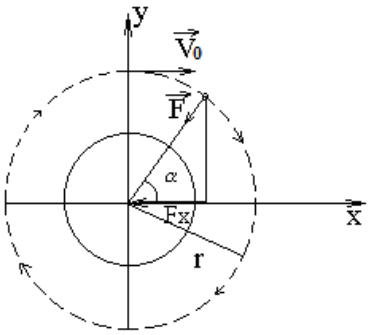


Рис. 2.11.1. Расчетная схема движения КЛА

Именно так и выводятся спутники на орбиту. Сначала ракета-носитель доставляет спутник на заданную высоту, затем он разгоняется по орбите до требуемой скорости разгонным блоком. Примем следующие начальные условия: $t = 0$, $V_{0y} = 0$, $V_{0x} = V_0$, $X = 0$, $Y = R+H$, где R – радиус Земли.

Модель движения в прямоугольной системе координат имеет вид:

$$\frac{dV_x}{dt} = -(g \cdot R^2) \cdot \frac{x}{(x^2+y^2)\sqrt{x^2+y^2}}, \qquad \frac{dV_y}{dt} = -(g \cdot R^2) \cdot \frac{y}{(x^2+y^2)\sqrt{x^2+y^2}}.$$

$$\frac{dx}{dt} = V_x; \qquad \frac{dy}{dt} = V_{y.}$$

Начальные условия при $t = 0$:

$V_y = 0$, $V_x = V_0$, $X = 0$, $Y = R+H$,

$R = 6378000$ м

Постановка задачи моделирования. Путем интегрирования уравнений движения построить траекторию движения КЛА по околоземной орбите, исследовать влияние начальных условий на вид и форму траектории.

Порядок выполнения лабораторной работы. Работа выполняется в среде MVS. Создадим новый проект с именем **«КЛА»**. В окне редактирования класса создадим необходимые переменные, параметры и константы (рис. 2.11.2): *x, y, V_x, V_y; k; g, R.*

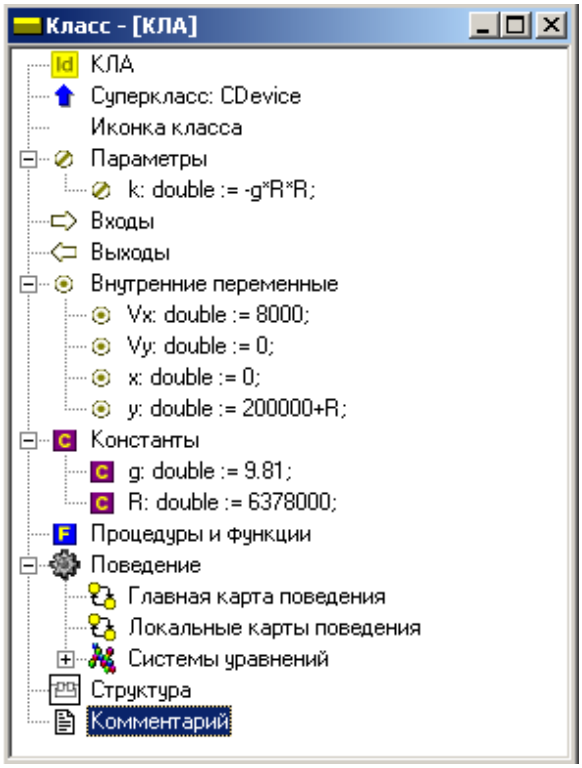


Рис. 2.11.2. Класс «КЛА»

Начальные значения задайте в соответствии с рис. 2.11.2. Все физические величины представлены в системе СИ. После создания всех необходимых параметров и переменных в окне класса создадим систему уравнений движения КЛА (рис. 2.11.3).

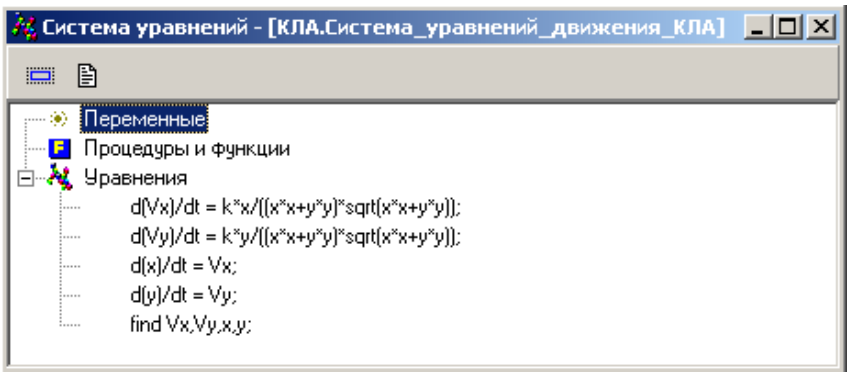


Рис. 2.11.3. Система уравнений, описывающая полет КЛА по орбите

Запустим модель . В окне визуальной модели создайте фазовую диаграмму для отображения траектории полета в координатах ***x–y*** (рис. 2.11.4).

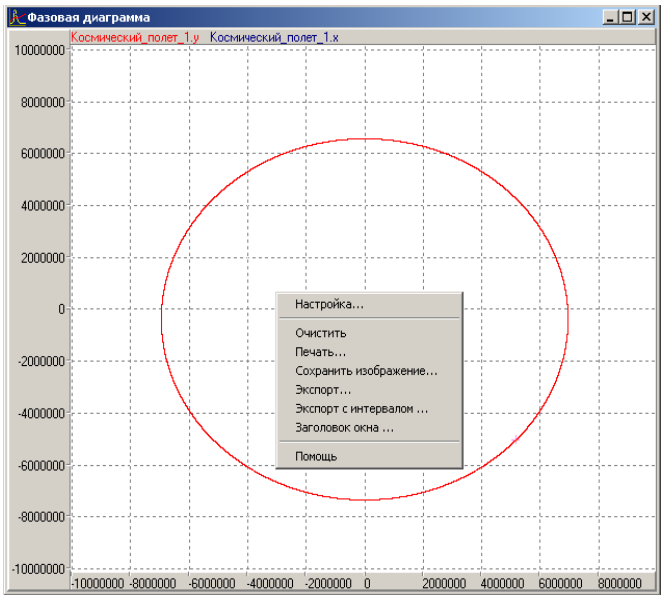


Рис. 2.11.4. Траектория полета

Для правильного отображения траектории настройте свойства диаграммы в соответствии с рис. 2.11.4. Включите режим **«Автомасштабирование»**. Проведите эксперимент с различными начальными значениями параметров.

Анализ результатов моделирования. С помощью построенной модели выполните старт с высоты 100 км с начальной скоростью 7 000 м/сек, с высоты 1 000 км с начальными скоростями 8 000 – 10 000 м/сек. Установите время полного витка по орбите. Установите характер изменения формы траектории при изменении начальных условий старта. Чему равны 1-я и 2-я космические скорости?

Установите, с какой минимальной скоростью следует запустить спутник на околоземную орбиту на высоте 200 км.

2.12. MVS-МОДЕЛЬ С ЭЛЕМЕНТАМИ УПРАВЛЕНИЯ

Построить модель фонарика, состоящую из «лампочки» и «кнопки». При нажатии кнопки «лампочка» должна загораться. Данная модель – ста-

тическая, т.е. ее параметры не изменяются во времени. Срабатывание кнопки и зажигание света происходит в модельном времени мгновенно.

Порядок выполнения работы в среде MVS. 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. 1.12.1). Дать проекту имя **Фонарик_1** и, нажав кнопку «обзор», поместить его, например, в папку C:\Мои модели.

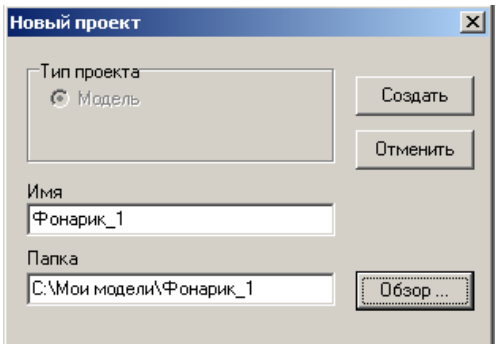


Рис. 2.12.1. Создание нового проекта

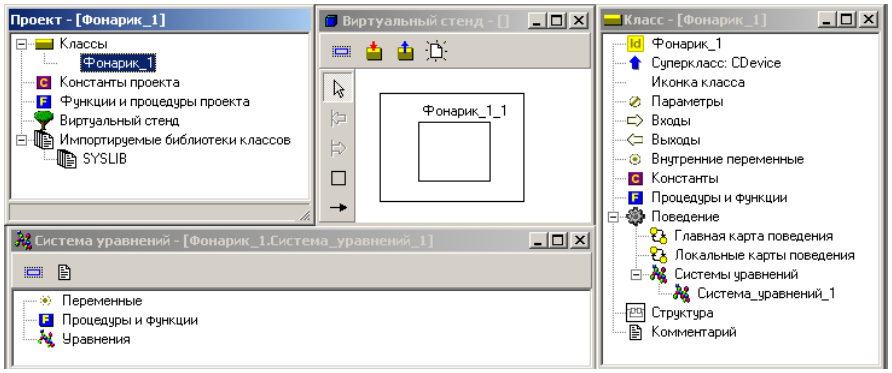


Рис. 2.12.2. Окна проекта

Нажать кнопку **«Создать»**. После чего в главном окне проекта появятся окна **«Проекта Фонарик_1»** (рис. 2.12.2).

В окне **класса «Фонарик_1»** следует добавить **внутреннюю переменную** булевского типа с именем **«Вкл».** Для добавления переменной, в окне **класса «Фонарик_1»**, 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. 2.12.3):

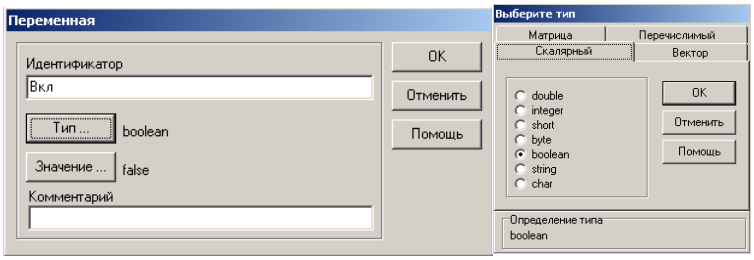


*Рис. 2.12.3. Задание имени перемененной **Вкл** и ее типа **boolean** с исходным значением **false***

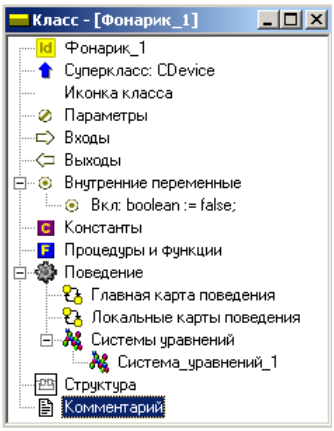


Рис. 2.12.4. Состояние класса Фонарик

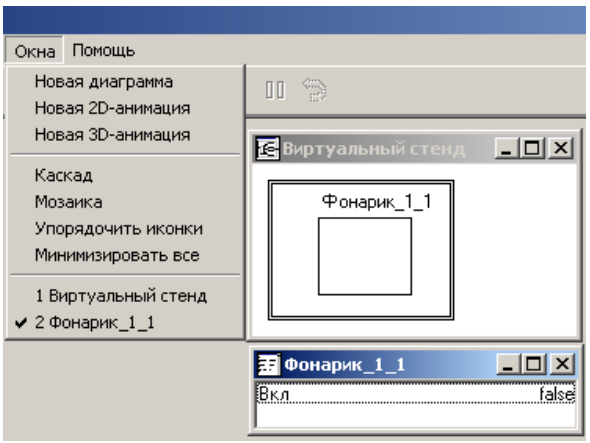


Рис. 2.12.5. Окна виртуального стенда и переменных испытательного стенда

После нажатия **ОК** в окне выбора типа, затем **ОК** в окне переменной в окне **«Класса Фонарик_1»** появится переменная **«Вкл»** типа **boolean**, со значением **false** рис. 2.12.4.

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. 2.12.5). Создадим окно для 2D-анимации: **Окно – Новая 2D-анимация** (рис. 2.12.5):

В меню **«Окна»** выберем пункт **«Новая 2D-анимация»**. В меню **«Сервис»** выберем **«Стандартные 2D-компоненты»** (рис. 2.12.6).

*Рис. 2.12.6. 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. 2.12.7).

Щелчок правой кнопкой над любым 2D-компонентом и выбор пункта меню **«Надпись»** позволяет подписать элементы в окне 2D-анимации (рис. 2.12.8).

Свяжем переменную **«Вкл»** с индикатором и кнопкой. Для этого поставим курсор на строку с именем переменной **«Вкл»** в окне переменных **«Фонарик_1_1»** (рис. 2.12.5), и перетащим переменную на кнопку в окне **«2D-анимации»** (рис. 2.12.7). И еще раз перетащим переменную **«Вкл»**, на индикатор.

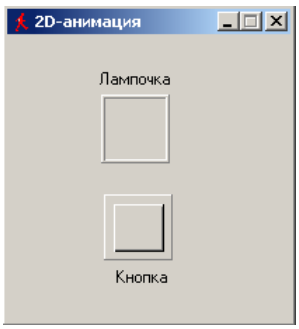


Рис. 2.12.7. Размещение сплошного линейного индикатора и кнопки на поле 2D-анимации

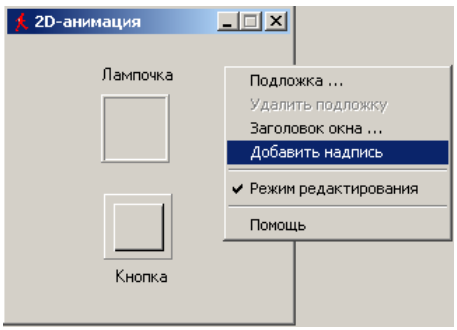


Рис. 2.12.8. Создание надписи

Щелкнем по кнопке – индикатор изменит цвет. Сделаем свет фонарика ярче: нужно правой кнопкой мыши щелкнуть по индикатору, 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_1_1** при щелчке по кнопке. Переменная **«Вкл»** меняет свое значение с **false** на **true.** Поскольку эта переменная связана с индикатором, то и он изменяет состояние.

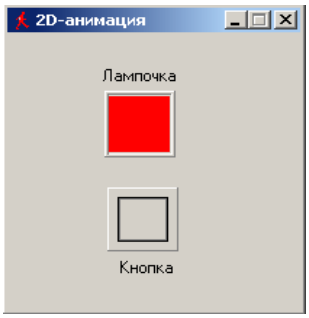


Рис. 2.12.8. Зажженный «фонарик»

В созданной модели фонарик светит только при нажатии кнопки. 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 (рис. 2.12.8).

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

Постановка задачи моделирования. Новая динамически управляемая модель будет, как и ранее, состоять из кнопки и «лампочки». Таким образом, можно использовать уже созданный проект. При нажатии кнопки на лампочку должно подаваться напряжение от батарейки. 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.

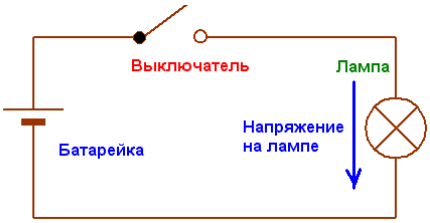


Рис. 2.12.9. Электрическая схема моделируемого объекта

Изменение напряжения батарейки со временем описывается уравнением:

$$U(t) = U_0 \cdot e^{-t/T_{\max}}, \quad (1)$$

где: *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 (1).

Порядок выполнения работы. Используем проект **«Фонарик_1»** и продолжим создание модели, потому что там уже выполнена значительная часть работы.

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.

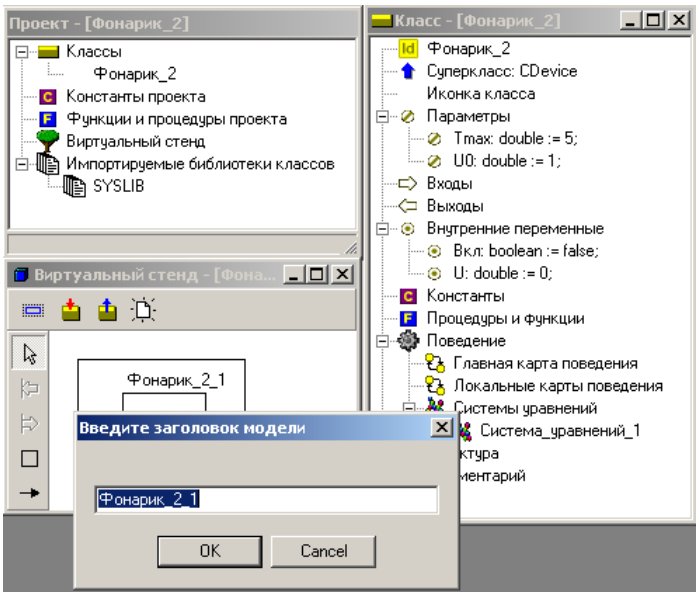


Рис. 2.12.10. Переименование модели

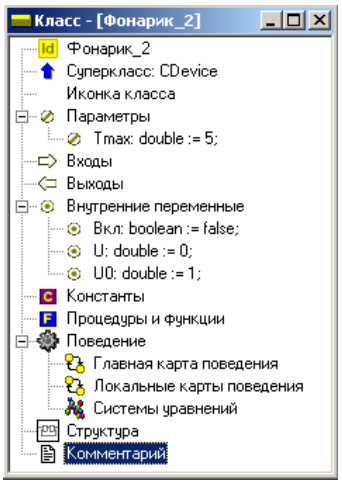


*Рис. 2.12.11. Добавлены переменные **U, U0** и параметр **Tmax***

Параметр – это величина, которую, в отличие от переменной, нельзя менять в процессе работы модели, но можно изменить до запуска модели. Аналогично добавим переменную ***U0*** – начальное напряжение батарейки (***U0 = 1***). После добавлений окно класса примет вид как на рис. 2.12.11. Напоминаем: MVS **различает** строчные и прописные буквы в именах переменных.

Введем уравнение, описывающее изменение напряжения батарейки с течением времени. Для этого нужно открыть окно **«Система уравнений»**, в окне **«Система уравнений»** щелкнуть правой кнопкой по пункту **«Уравнения»** и в контекстном меню выбрать **«Изменить»** (рис. 2.12.12).

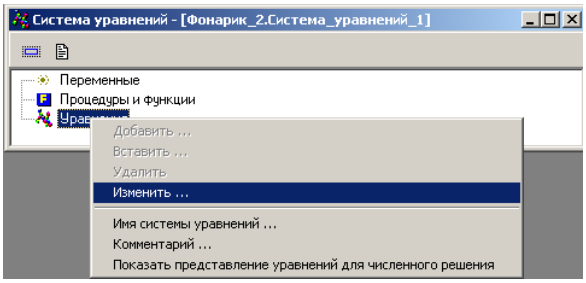


Рис. 2.12.12. Открытие редактора формул для изменения системы уравнений

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

$U = U_0 \exp(\mathbf{Time}/Tmax)$.

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

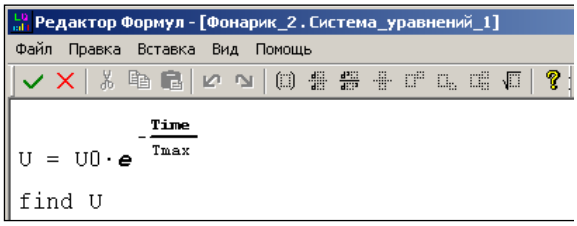


Рис. 2.12.13. Запись уравнения в редакторе формул

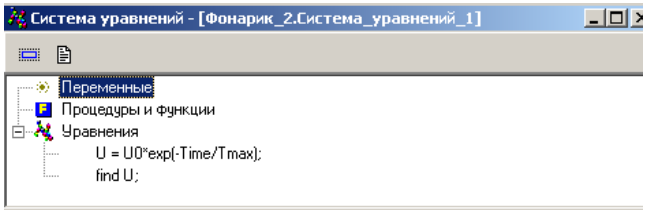


Рис. 2.12.14. Уравнение разряда батарейки

После чего щелкнуть по кнопке – **«Сохранить и выйти»**. Окно системы уравнений примет вид см. рис. 2.12.14.

Создадим модель. Для этого нужно, как и ранее, нажать на кнопку – **«Запустить модель»** в главном окне проекта. В результате появится окно **«Испытательного стенда»** (рис. 2.12.15).

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.

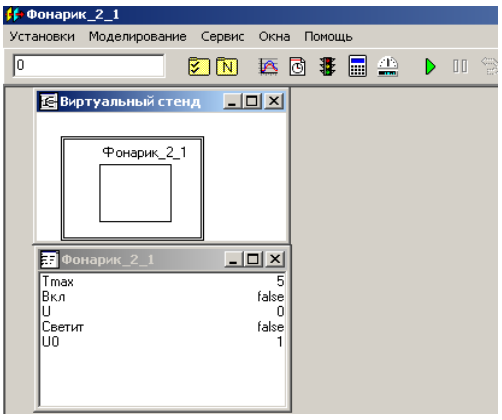


Рис. 2.12.16. Окно испытательного стенда

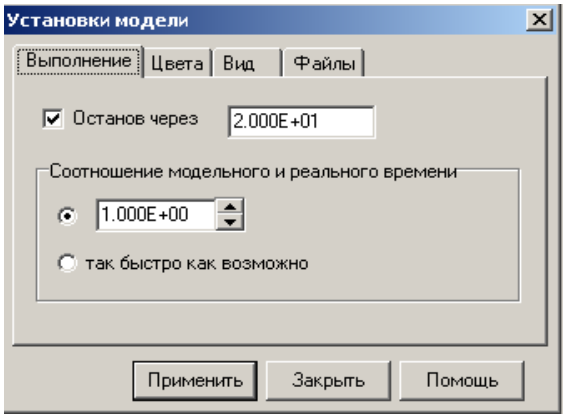


Рис. 2.12.17. Установка времени моделирования

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»** переменную ***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 (рис. 2.12.17). Результат моделирования представлен на рис. 2.12.18.

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»**.

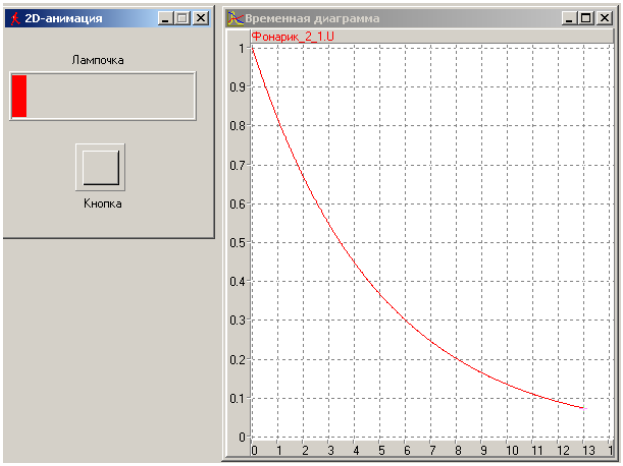


Рис. 2.12.18. Результат первого запуска модели

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 модель 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 (1). 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]»**. 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. Редактирование **«Карты поведения»** представлено в таблице 2.12.1.

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.

Таблица 2.12.1

Кнопки редактирования «Карты поведения»

Кнопка	Назначение
	Переход в диалог редактирования имени объекта
	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

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.

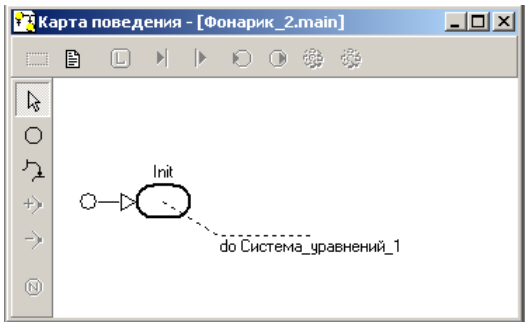


Рис. 2.12.19. Исходный вид главной карты поведения

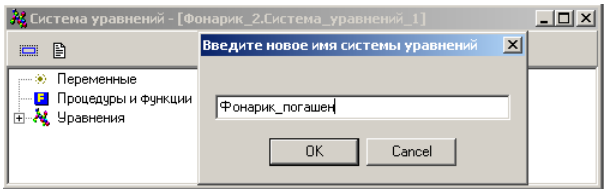


*Рис. 2.12.20. Окно исходной системы уравнений узла **Init***

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»**. Появится новое окно (рис. 2.12.19) **«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. 2.12.19, приписана узлу **Init.** Появится и станет активным **окно «Система уравнений».**

Щелкнем по кнопке **«Имя системы уравнений» –** , в появившемся окне (рис. 2.12.20) введем **«Фонарик_погашен»** и щелкнем по кнопке ОК. Изменим положение надписи **«Фонарик_погашен»** в окне карты поведения. Она примет такой вид см. рис. 2.12.21.

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. 2.12.23).

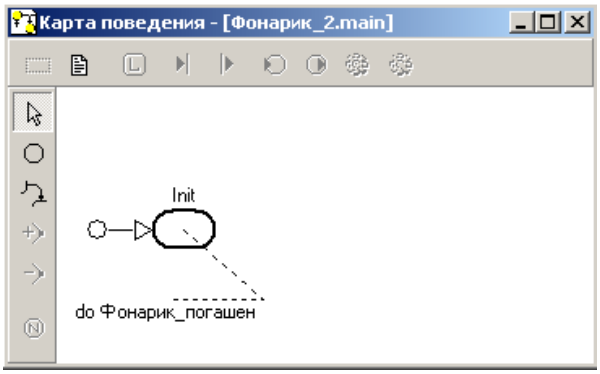


*Рис. 2.12.21. Карта поведения с новым названием системы уравнений узла **Init***

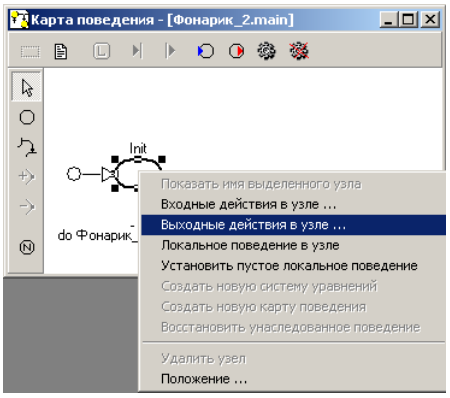


*Рис. 2.12.22. Открытие редактора **входных действий**, которые должны выполниться при входе в узел **Init***

В появившемся окне редактора формул введем оператор **Светит: = false;** и щелкнем по кнопке **«Сохранить и выйти»** (рис. 2.12.24).

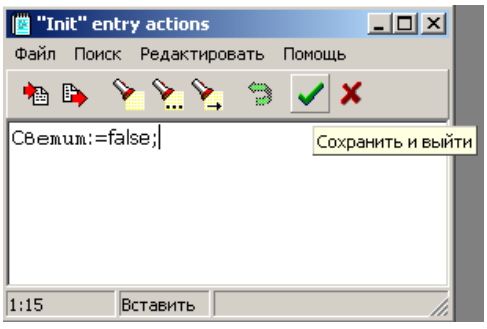


Рис. 2.12.25. Окна редактора формул для входных действий узла Init

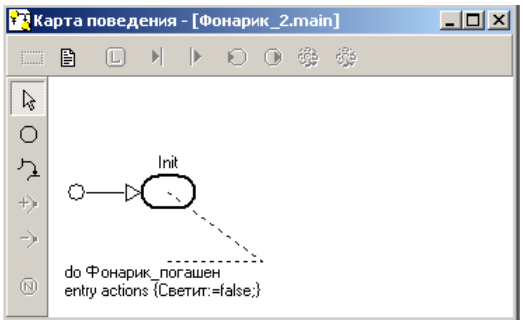


*Рис. 2.12.26. Действия при входе (**entry actions**) в начальный узел **Init***

Карта поведения примет вид по рис. 2.12.26. Добавим узел, соответствующий включенному состоянию фонарика. Нужно щелкнуть по кнопке **«Создать новый узел»** в окне карты поведения (рис. 2.12.26), переме-

стить курсор на поле карты (он примет вид крестика), в правой части карты нажать левую клавишу и удерживая ее растянуть вправо вниз овал нового узла.

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. 2.12.24–2.12.25). В редакторе формул необходимо записать: **Светит: true;** (рис. 2.12.27). Щелкнуть по кнопке  **«Сохранить и выйти»**. Карта поведения примет вид – рис. 2.12.28.

Примечание. 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 (рис. 2.12.28).

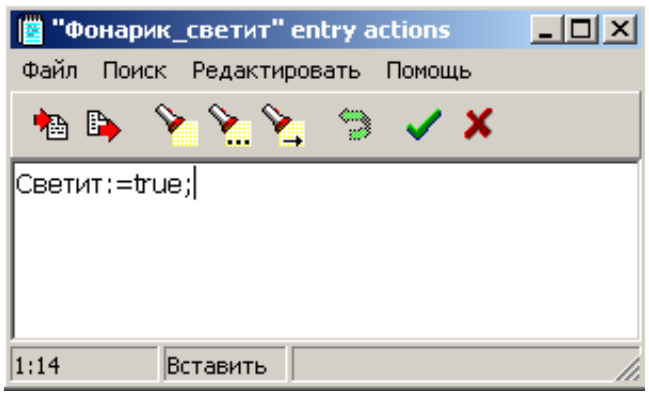


*Рис. 2.12.27. Входное действие (**entry actions**) узла **Фонарик_светит***

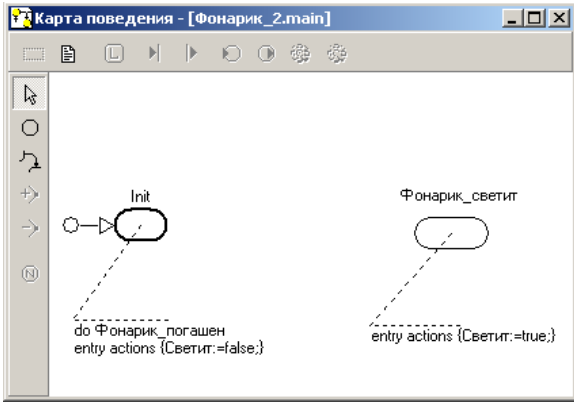


Рис. 2.12.28. Создание узлов на карте поведения. Указаны входные действия

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. Теперь линию можно изменять, ухватившись как за одну, так и за другую опорную точку.

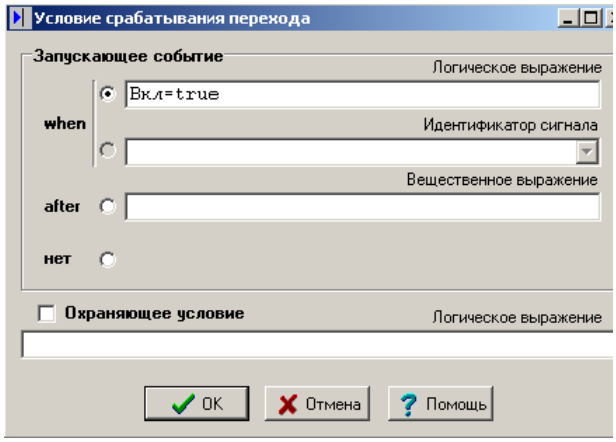


Рис. 2.12.18. Задание условия срабатывания перехода

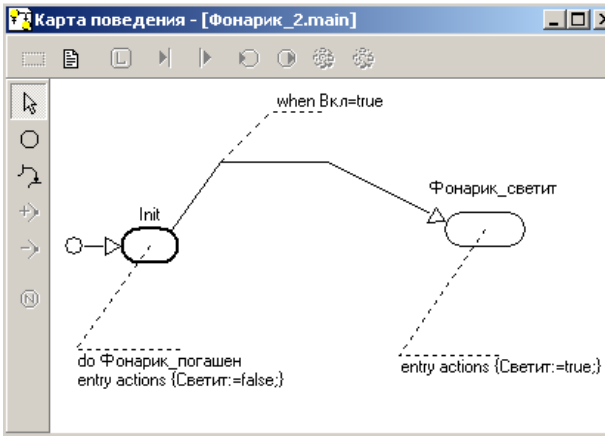


Рис. 2.12.19. Переход из узла Init в узел Фонарик_светит

Теперь следует поставить **условие срабатывания перехода**. Переход определяется **событием** – **нажатие кнопки фонарика**. Поэтому переход должен происходить в момент, когда переменная **«Вкл»** примет значение **true**. 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. 2.12.29).

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

Нажать кнопку **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 рис. 2.12.30.

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. 2.12.30). 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**.

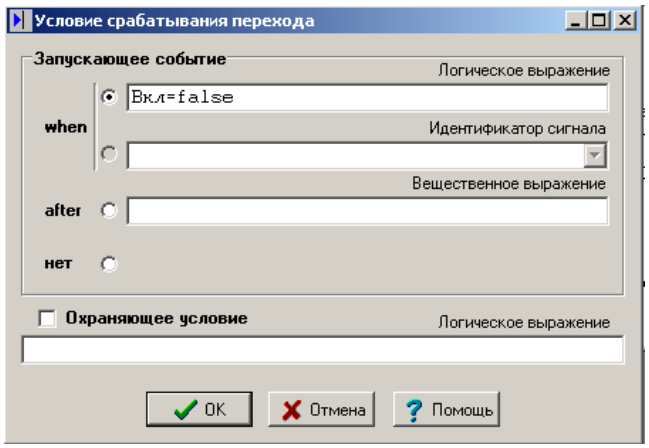


Рис. 2.12.31. Ввод условия срабатывания при переходе от Фонарик_погашен к Init

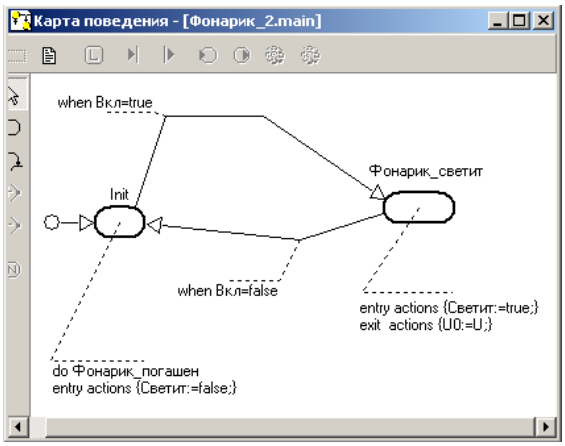


Рис. 2.12.32. Карта поведения

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 предыду-

щем 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. 2.12.33. Щелкнем по кнопке ОК. Если подравняем элементы карты поведения, то она примет примерно такой вид.

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 (рис. 2.12.32). При записи формулы использована переменная MVS ***LocalTime***, которая имеет начало отсчета в момент времени, когда узел **«Фонарик_светит»** становится активным. Аналогичным образом наполним формулами систему уравнений **«Фонарик_погашен»,** в которой переменной **U** задается нулевое значение (рис. 2.12.33).

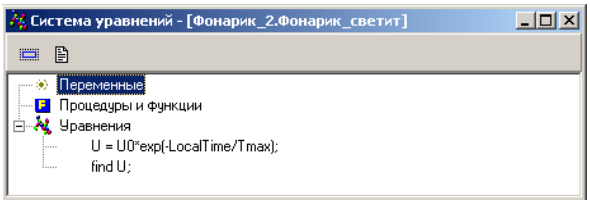


Рис. 2.12.32. Измененная система уравнений

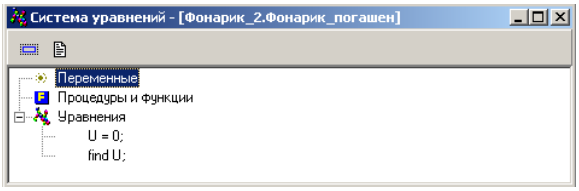


Рис. 2.12.33. Система уравнений «Фонарик_погашен»

Теперь карта поведения примет окончательный вид (рис. 2.12.34).

Связывание переменных с объектами 2D-анимации. 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. 2.12.35). Действительно, теперь лампочка реагирует на нажатие кнопки, а напряжение уменьшается только во время свечения лампочки.

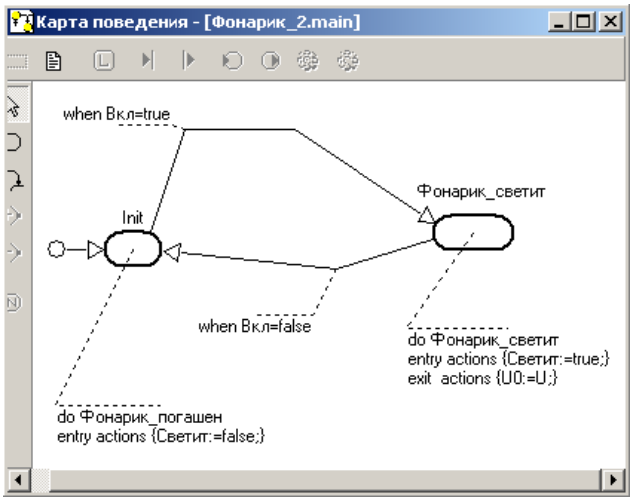


Рис. 2.12.34. Готовая карта поведения

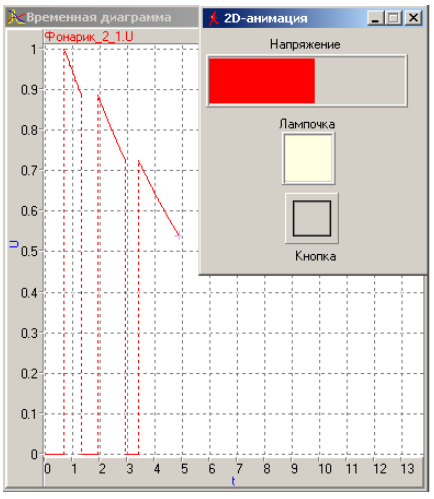


Рис. 2.12.35. Результат моделирования

В качестве переменных модели выберем булевскую переменную **«Вкл»** типа **boolean**, которая будет соответствовать состоянию кнопки выключателя, и переменную ***U*** типа **double**, которая будет определять напряжение, подаваемое на лампочку в момент, когда **«Вкл»** становится **true**. В момент времени, когда кнопка отжимается, **«Вкл»** должна становиться **false3**, 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**.

2.13. Поражение цели с заданными координатами

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

$$\frac{d\overline{V_y}}{d\overline{t}} = -1 - \overline{k}\overline{V_y}; \qquad \frac{d\overline{V_x}}{d\overline{t}} = -\overline{k} \cdot \overline{V_x}.$$

$$\frac{d\bar{x}}{d\bar{t}} = \bar{V}_x, \frac{d\bar{y}}{d\bar{t}} = \bar{V}_y.,$$

Начальные условия:

$$\bar{V}_x(\bar{t}=0) = \cos(\alpha); \quad \bar{V}_y(\bar{t}=0) = \sin(\alpha);$$

$$\bar{x}(\bar{t}=0) = 0, \quad \bar{y}(\bar{t}=0) = 0.$$

Постановка задачи моделирования. Целью моделирования является определение значения угла α, которое обеспечит попадание тела в точку с координатами *x = a, y = b* или максимально возможное приближение к данной точке. Примерно такая задача стоит перед расчетом артиллерийского орудия.

Порядок выполнения работы. Работа выполняется в среде пакета MVS. Первоначально средствами MVS необходимо построить модель движения тела (рис.2.13.1). В отличие от предыдущей модели переменная ***x*** объявляется как **выходная переменная**. В систему уравнений должна быть добавлена переменная $z = \sqrt{(x-a)^2 + (y-b)^2}$ – расстояние до заданной точки, которая также объявляется **выходной** переменной. С данными переменными будет взаимодействовать пакет оптимизации MVS.

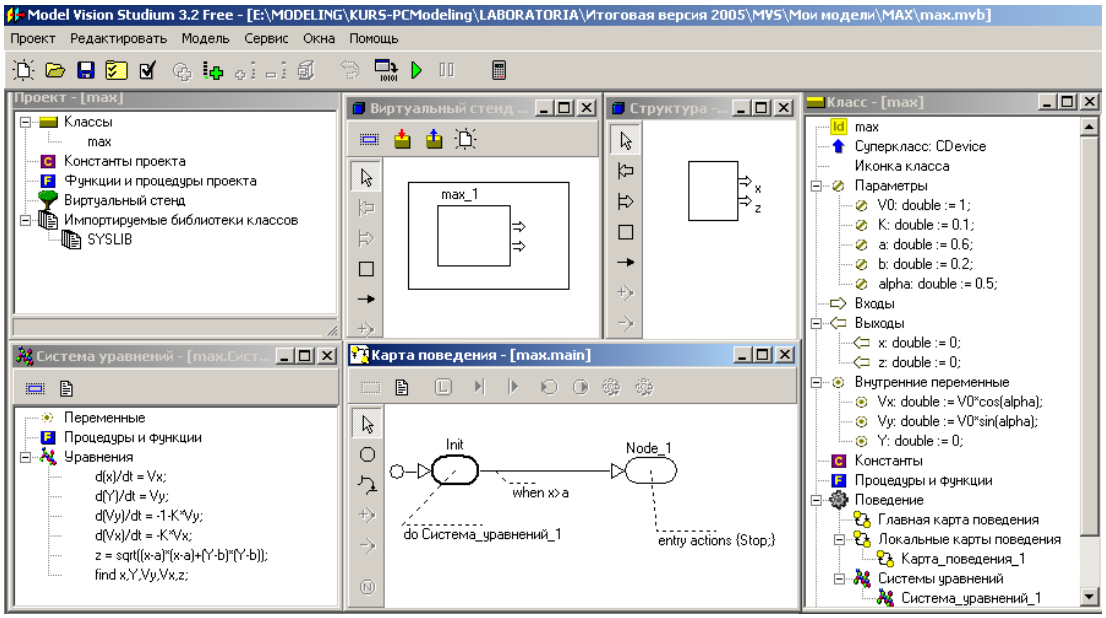


Рис. 2.13.1. Модель движения тела

Дополнительное окно **«Структура»** появится автоматически после объявления переменных ***x*** и ***z***. Для решения данной задачи необходимо построить карту поведения и предусмотреть останов полета, если выполняется условие*: x > α*. Именно для этого момента времени в подсистеме оптимизации MVS будет определяться минимальное расстояние до заданной точки.

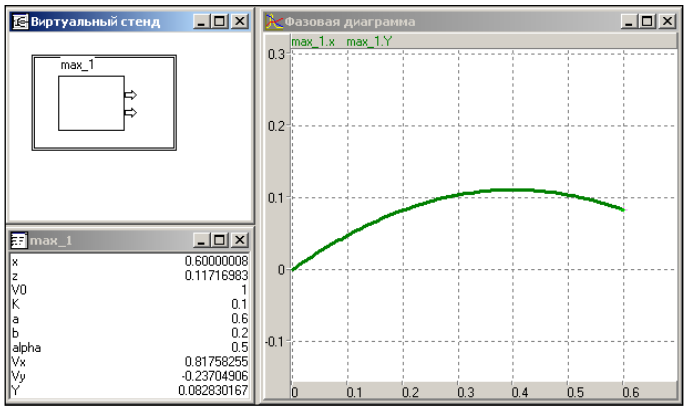


Рис. 2.13.2. Результат моделирования для начальных значений параметров

Далее выполним запуск модели  , дополним модель фазовой диаграммой и проведем эксперимент с начальными значениями параметров по рис. 2.13.2. Его результат представлен на рис. 2.13.2. Очевидно, что при первоначально выбранном значении параметра α *= 0,5* попадания в цель не происходит. 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 будем использовать пакет оптимизации MVS. Для применения пакета оптимизации в окне проекта (рис. 2.13.1) выполним действия: пункт меню **«Сервис» – «Оптимизация»**. В результате откроется окно подсистемы оптимизации (рис. 2.13.3), поля которого: **«Па-**

раметры», «Функционал», «Ограничения», «Вычислять функционал в момент…»; необходимо заполнить с использованием метода перетаскивания в соответствии с рис. 2.13.3.

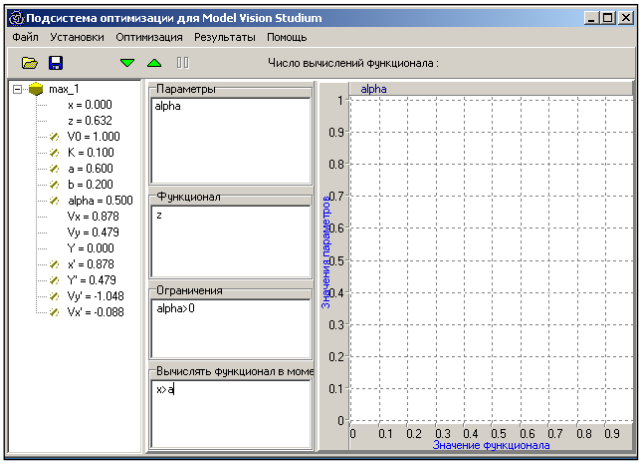


Рис. 2.13.3. Начальное состояние подсистемы оптимизации

Решение задачи поиска **минимального** значения *z* (расстояние до заданной точки) будет выполнено после нажатия кнопки  . Предварительно в пункте меню **«Установки»** необходимо выбрать метод оптимизации **«Random search»** (рис. 2.13.4).

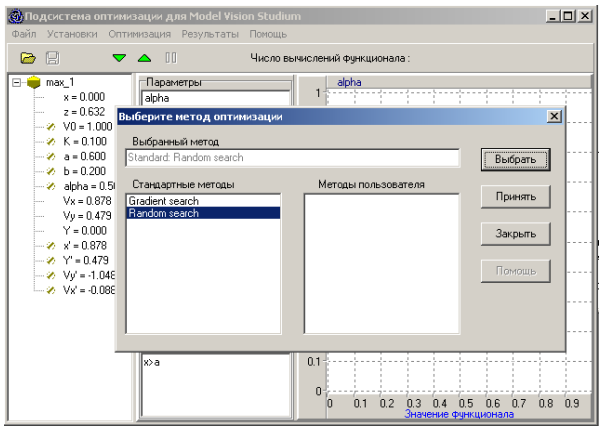


Рис. 2.13.4. Выбор метода оптимизации

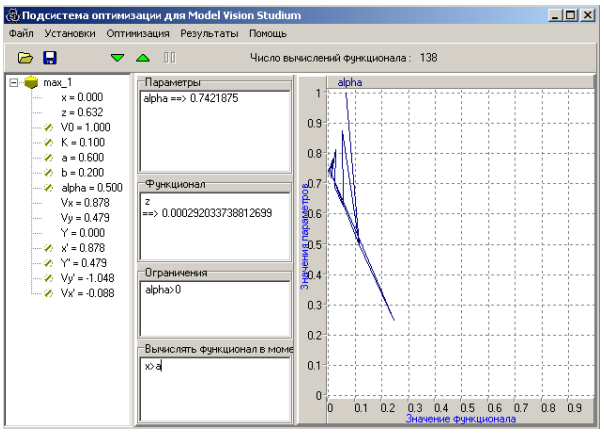


*Рис. 2.13.5. Результат поиска оптимального значения угла **α***

Подсистема оптимизации пакета MVS определит такое значение параметра ***α,*** при котором значение переменной ***z*** будет минимально. Результат поиска оптимального решения представлен на рис. 2.13.5.

Анализ результатов моделирования. Необходимо задать найденное значение параметру ***α*** и провести моделирование движения тела повторно. Результат, представленный на рис. 2.13.6, свидетельствует о том, что попадание в цель действительно происходит.

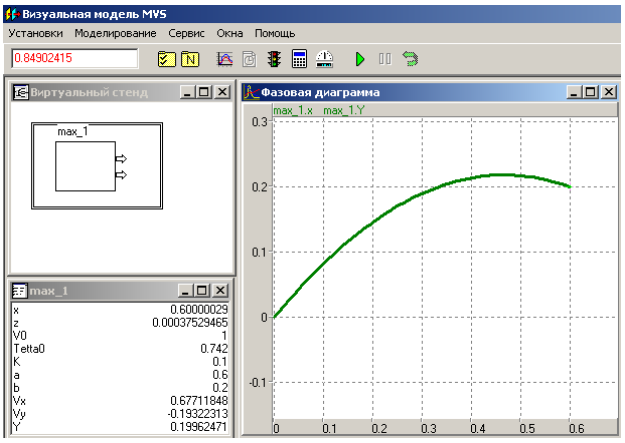


Рис. 2.13.6. Траектория движения тела при оптимальном значении угла ***α***

Если попадание в заданную точку невозможно, то пакет оптимизации определит значение угла ***α***, при котором траектория полета пройдет максимально близко от заданной точки. Попробуйте в экспериментах с моделью задать координаты: точки ***a = 0,4, b = 0,4*** и решить задачу оптимизации.

САМОСТОЯТЕЛЬНАЯ РАБОТА

2.14. 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].

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 <= p.***

Вероятность противоположного события равна ***1–p***. Следовательно, если соотношение ***x = p*** выполняется, то исходом испытания является событие *A*, в противном случае исходом испытания является событие *НЕ А.*

Постановка задачи моделирования. Построить компьютерную MVS-модель реализации случайного события с заданной вероятностью и исследовать ее свойства.

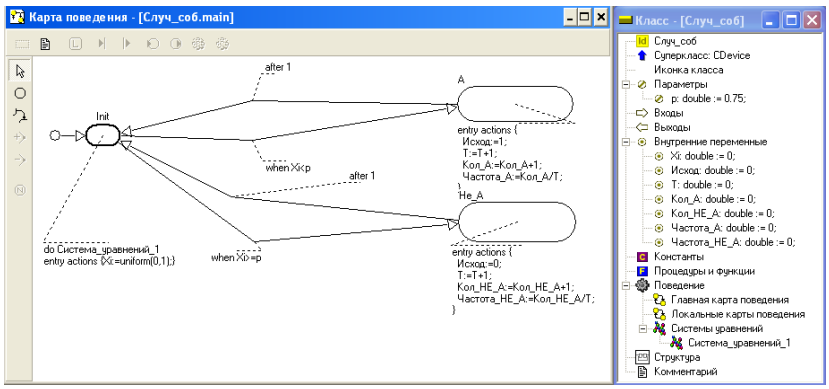


Рис. 2.14.1. Модель случайного события

Для решения задачи необходимо построить модель по рис. 2.14.1–2.14.2. Соответственно, они являются вспомогательными для расчета частоты событий, т.е. значений переменными ***Частота_А*** и ***Частота_НЕ_А.***

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 < p***, или значение **0** в противном случае. Переменные ***Кол_А*** и ***Кол_НЕ_А*** подсчитывают количество исходов событий, которые произошли или не произошли. Постройте временную диаграмму рис. 2.14.2.

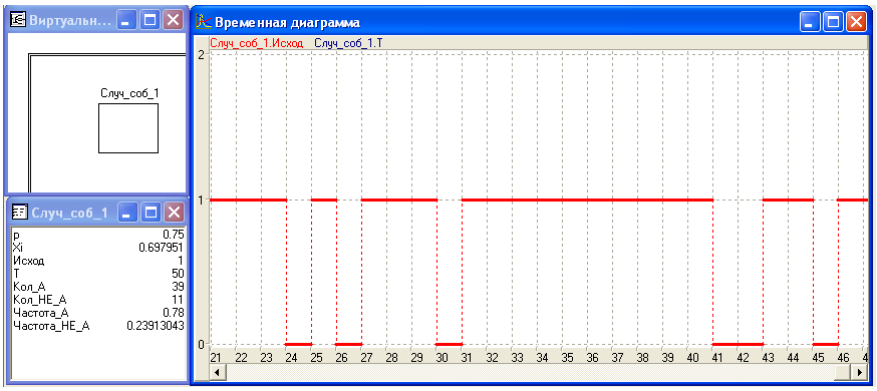


Рис. 2.14.2. Временная диаграмма событий

Анализ результатов эксперимента. Провести ряд экспериментов при разном количестве реализаций события ***A***: 50; 100; 2000 и сопоставить значение заданной вероятности события ***p*** и частоты его реализации в эксперименте.

2.15. Модель 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 гаража начинается с прихода в гараж нового автомобиля в момент t_0. Он 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.

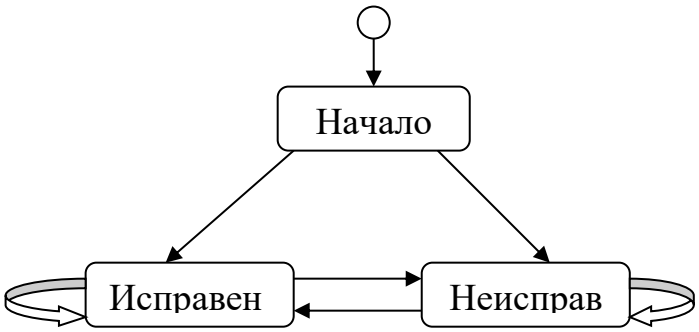


Рис. 2.15.1. Карта состояний при моделировании работы гаража

Работу автомобиля можно представить графом (рис. 2.15.1), 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. 2.15.2).

Все вероятности переходов в этой модели зависят только от конкретного состояния, а не от того, как система пришла в данное состояние. Нас интересую 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.

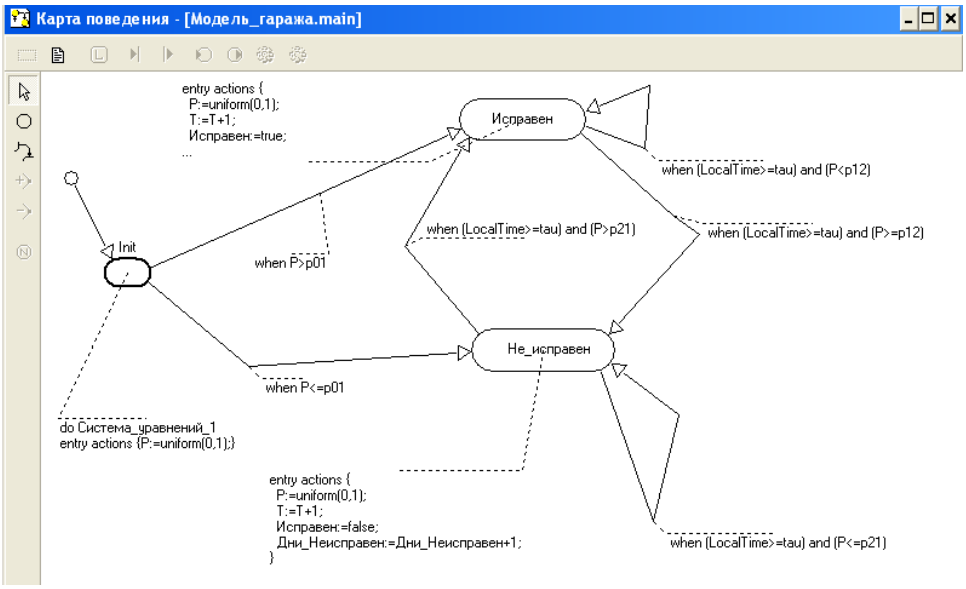


Рис. 2.15.2. Карта поведения при моделировании работы гаража

Вернемся к примеру с гаражом (рис. 2.15.2). На этом рисунке представлена имитационная модель автомобиля, и теперь уже изображенный граф является картой поведения. Мы моделируем случайное событие как возможность в момент прихода 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.

Постановка задачи моделирования. Построить компьютерную модель работы гаража и исследовать ее свойства.

Порядок выполнения работы

1. Создадим новую модель, назовем ее **«Модель гаража»**. Введем переменные на рис. 2.15.3.

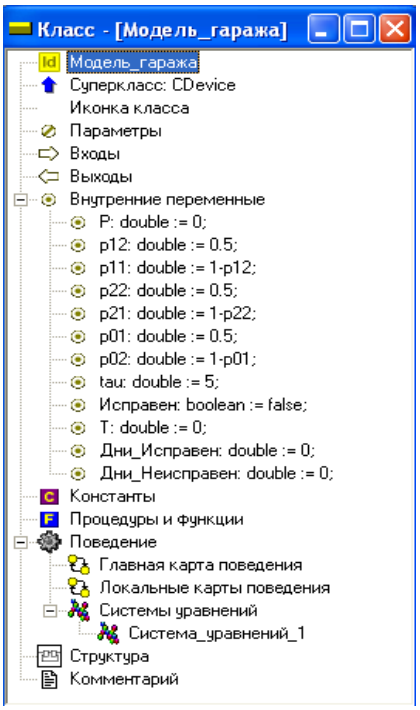


Рис. 2.15.3. Исходные данные для моделирования гаража

- 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), значение 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.

2. Составим карту поведения как показано на рис. 2.15.2.
3. Запустите модель.

4. Создайте фазовую диаграмму (рис. 2.15.4).

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

Настройка фазовой диаграммы:

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

Рис. 2.15.4. Фазовая диаграмма моделирования работы автомобиля

5. 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**.

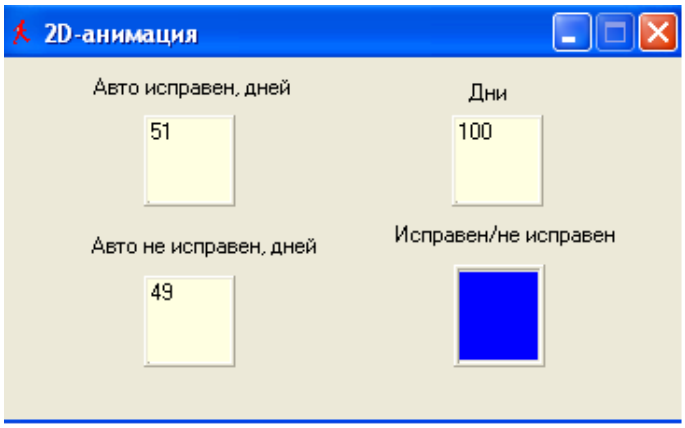


Рис. 2.15.5. 2D-анимация отображения результатов моделирования

2.16. 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, которое происходит 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:

$$(X,Y);\quad (X,Y+1);\quad (X,Y-1);\quad (X+1,Y);\quad (X+1,Y+1);$$

$$(X+1,Y-1);\quad (X-1,Y);\quad (X-1,Y+1;\quad X-1,Y-1).$$

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

Вероятность изменения каждого параметра задана: $PX_a,\ PX_b,\ PX_c;\quad PY_a,\ PY_b,\ PY_c$. Причем ***$PX_a + PX_b + PX_c = 1$, $PY_a + PY_b + PY_c = 1$.*** 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.

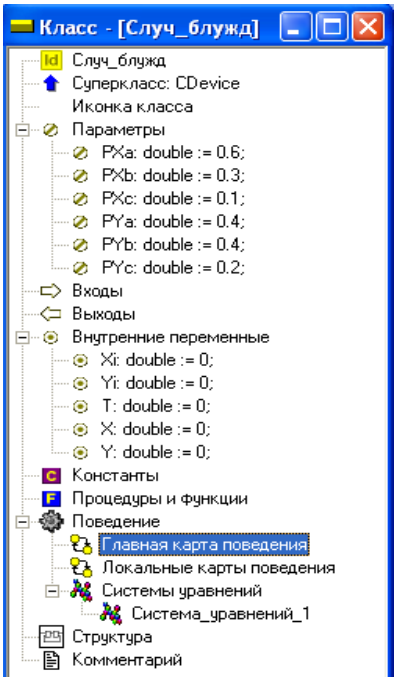


Рис. 2.16.1. MVS-моделирование случайного блуждания

Постановка задачи моделирования. Построить компьютерную модель реализации случайного блуждания. Исследовать поведение объекта. Моделирование выполняется в среде MVS. Для решения задачи необходимо построить модель по рис. 2.16.1–2.16.2.

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

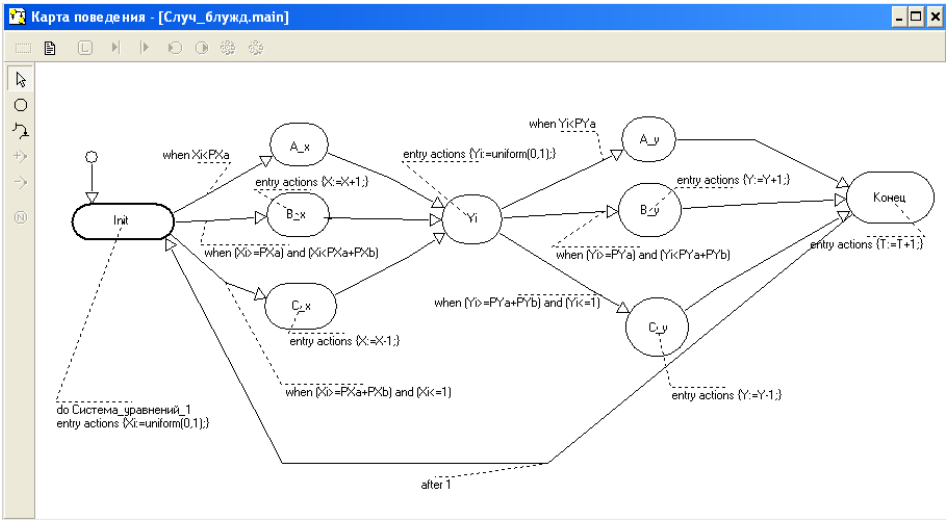


Рис. 2.16.2. Карта поведения для MVS-модели случайного блуждания

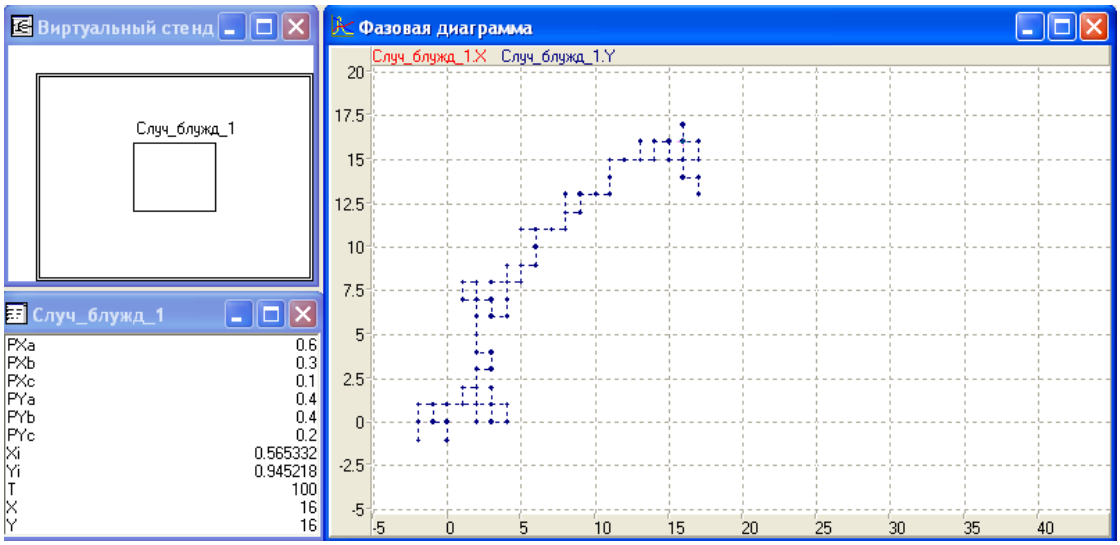


Рис. 2.16.3. Результат моделирования: траектория блуждания

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

2.17. Контрольные вопросы к главе 2

1. Требуются ли численные методы при создании модели в MVS?
2. Требуются ли программирование при создании модели в MVS?
3. Какие окна создает MVS при создании модели?
4. Какие диаграммы можно создать для представления результатов моделирования в MVS?
5. Можно ли найти оптимальное значение переменной в MVS?
6. Для чего создается «Карта поведения»?
7. Что такое событие в MVS?
8. Для каких моделей создается «Карта поведения»?
9. Что такое гибридная модель?
10. Чем отличается в MVS константа от параметра и от переменной?

ГЛАВА 3. КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ В СРЕДЕ MS EXCEL

3.1. Применение электронных таблиц в моделировании

Математическое моделирование в электронных таблицах проводится по общей схеме, которая содержит: постановку задачи, разработку модели, компьютерный эксперимент и анализ результатов. Среда электронных таблиц – это инструмент, который позволяет быстро выполнить расчет и пересчет количественных характеристик исследуемого объекта или процесса и позволяет наглядно представить результаты.

Электронные таблицы позволяют решать задачи, связанные с числовыми расчетами, анализировать большие объемы данных, создавать графики и диаграммы. Один из наиболее распространенных табличных процессоров – MS Excel. Электронные таблицы позволяют просто решить задачу поиска оптимального решения, исследовать стохастические модели. Однако алгоритмы решения уравнений необходимо реализовывать в виде последовательности вычислительных формул. MS Excel содержит богатую библиотеку стандартных функций и ряд надстроек, которые необходимо подключить при необходимости. Возможности MS Excel могут быть существенно расширены путем создания макросов (программ в объектно-ориентированной среде программирования *Visual Basic for Applications-VBA*), которые встраиваются в книгу MS Excel. Причем некоторые задачи моделирования в MS Excel решать удобнее и проще.

3.2. Технология решения задач оптимизации

В электронных таблицах Excel с помощью надстройки **«Поиск решения»** можно решать задачи оптимизации. Первоначально необходимо

включить надстройку. Последовательность подключения надстройки показана на рис. 3.2.1–3.2.4. Запустить Exccl и открыть вкладку «Файл». Далее открыть пункт **«Параметры»** (рис. 3.2.1).

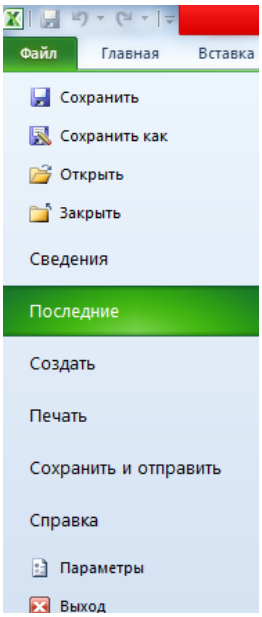


*Рис. 3.2.1. Вкладка «Файл», пункт **«Параметры»***

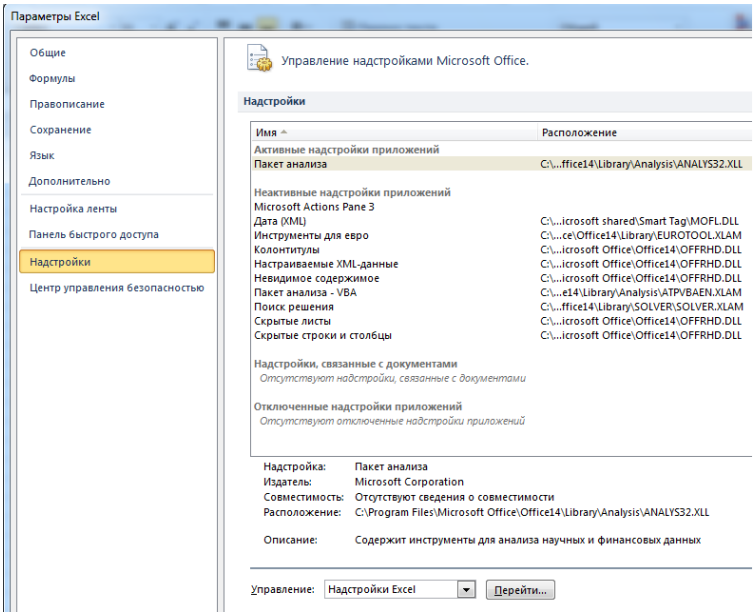


Рис. 3.2.2. Пункт «Надстройки» – кнопка «Перейти»

Далее перейти по пункту **«Надстройки»** и по кнопке **«Перейти»** вызвать **«Надстройки»** и отметить пункт **«Поиск решения»** (рис. 3.2.1–3.2.4).

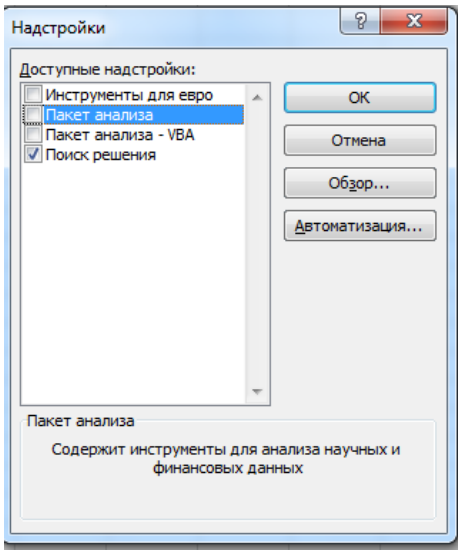


Рис. 3.2.3. Выбор надстройки ***«Поиск решения»***

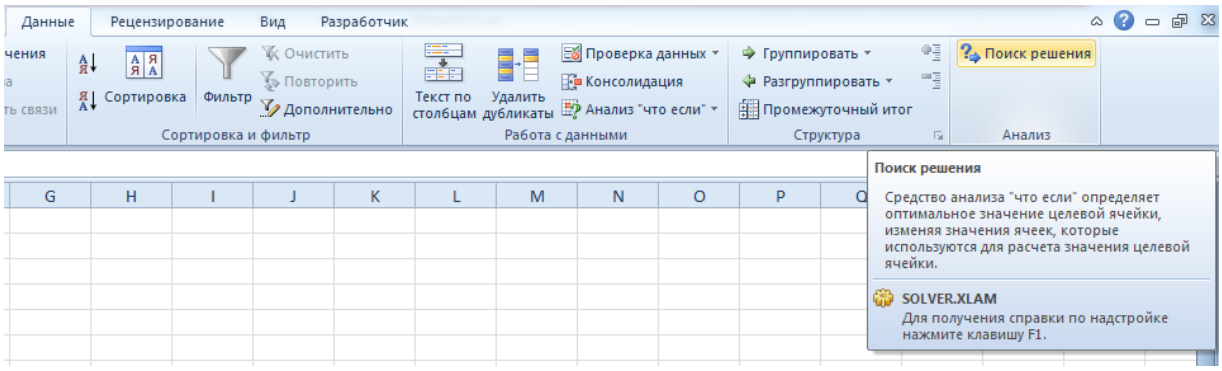


Рис. 3.2.4. Доступ к надстройке, ***«Поиск решения».***
Открыта вкладка ***«Данные»***

Отправной точкой при поиске оптимального решения является модель вычисления, созданная в рабочем листе в виде электронной таблицы. Надстройке **«Поиск решения»** необходимы следующие данные:

1. **Изменяемые ячейки** – это ячейки, которые в итоге будут содержать искомые значения параметров. При этих значениях целевая функция получит экстремальное (минимальное или максимальное) значение.

2. **Целевая ячейка** – это ячейка таблицы, значение в которой должно быть максимизировано или минимизировано. Она должна содержать формулу, которая прямо или косвенно ссылается на изменяемые ячейки. Эта формула представляет собой целевую функцию.

3. Значения в изменяемых ячейках будут последовательно изменяться до тех пор, пока не будет получено нужное значение в целевой ячейке. Эти ячейки, следовательно, прямо или косвенно должны влиять на значение целевой ячейки.

4. Вы можете задать **ограничения и граничные условия.** Можно задать также ограничения для других ячеек, прямо или косвенно присутствующих в модели.

После задания всех необходимых параметров можно запустить выполнение поиска решения.

Порядок решения задачи с помощью надстройки «Поиск решения»

1. Создайте оптимизационную модель, которую следует представить в виде электронной таблицы.

2. Выполните переход **«Данные/Поиск решения»**. В открывшемся окне диалога в поле **«Целевая ячейка»** будет представлена ссылка на активную ячейку.

3. Выделите в таблице целевую ячейку. Соответствующая ссылка затем будет автоматически отображена в окне диалога **«Поиск решения».**

4. В группе **«Максимум», «Минимум», «Значения:»** установите переключатель в нужное положение, в зависимости от того, должно ли в результате оптимизации значение в целевой ячейке быть максимальным, минимальным или же равняться определенному заданному значению.

5. В поле **«Изменяя ячейки переменных»** укажите ссылки на изменяемые ячейки. Поместите курсор в это поле и выделите соответствующие ячейки.

6. При необходимости задайте нужные ограничения (рис. 3.2.5), используя описанную ниже процедуру определения ограничений.

7. Нажмите кнопку **«Найти решение»**. Надстройка определит решение, выполняя последовательные вычисления и изменяя значения в соответствующих ячейках. Если программа нашла решение, то это будет отображено в окне диалога **«Результаты поиска решения»** (рис. 3.2.6) и в модели.

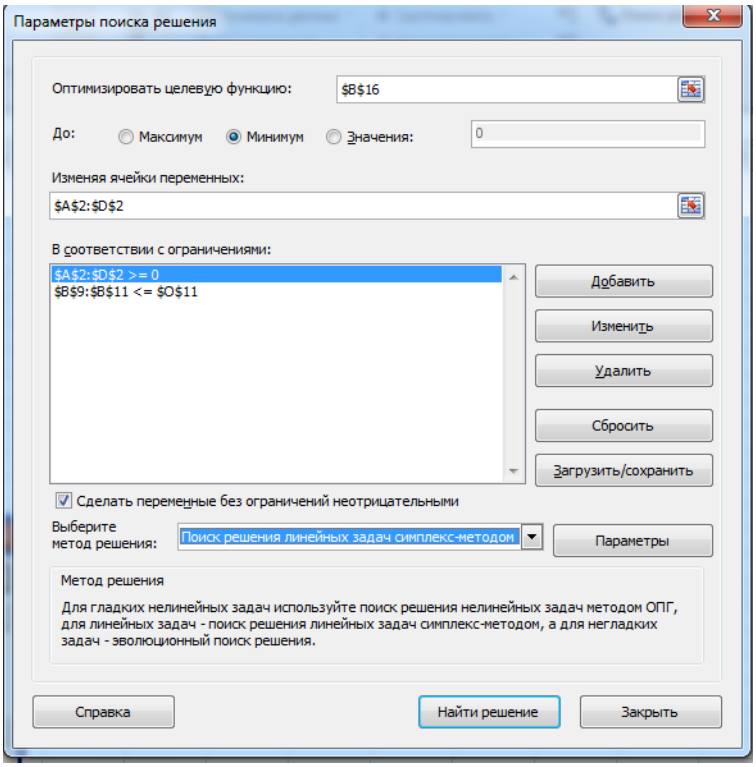


*Рис. 3.2.5. Окно **«Поиск решения»***

8. Определите, следует ли сохранить в модели найденное решение.

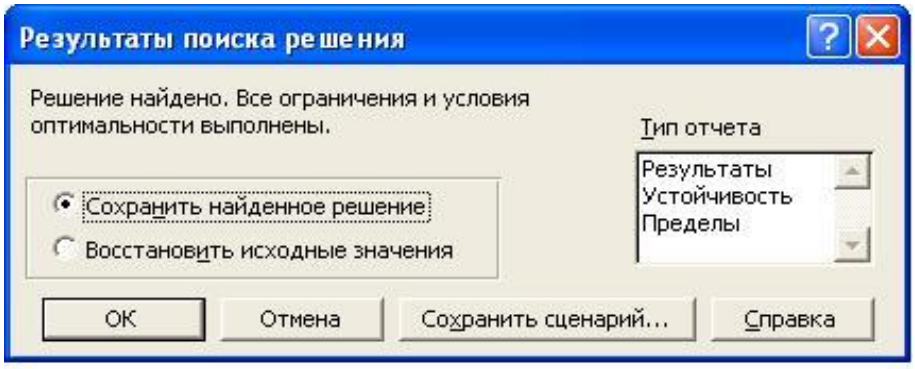


*Рис. 3.2.6. Окно **«Результаты поиска решения»***

Если поиск решения не привел к нахождению оптимального результата, то в окне диалога будет отображено соответствующее сообщение.

Ограничения – это условия, которые должны быть выполнены при поиске решения при решении задачи оптимизации. Ограничения опреде-

ляют допустимые значения переменных, которые варьируются в процессе поиска решения.

1. В окне диалога **«Поиск решения»** в поле **«Ограничения»** нажмите кнопку **«Добавить»**. На экране будет отображено окно диалога для определения ограничения (рис. 3.2.7).

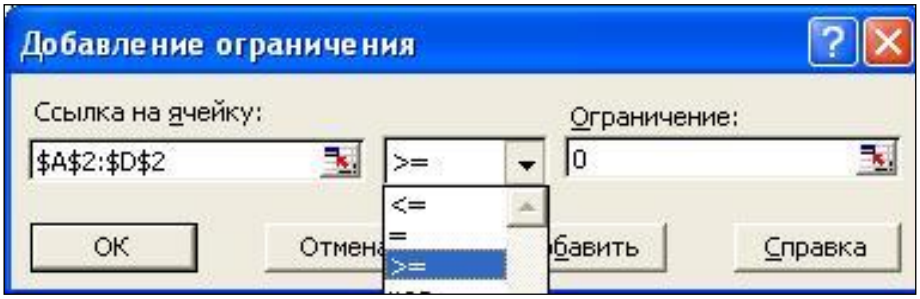


*Рис. 3.2.7. Окно **«Добавление ограничений»***

2. В поле **«Ссылка на ячейку»** укажите ссылку на ячейку или диапазон ячеек, для которой(ых) должно действовать ограничение.

3. Выберите из списка нужный режим сравнения: **=, <=, >=.** Дополнительно в распоряжении имеется режим **«Цел»** (для переменной допустимы только целые значения) и **«Двоич.»** – допустимы только значения **0** или **1**.

4. В поле **«Ограничение»** укажите верхнюю или нижнюю границу. Вы можете указать как число, так и ссылку на ячейку, диапазон ячеек, а также формулу.

5. Нажмите кнопку **«Добавить»**. При этом окно не будет закрыто.

6. Аналогичным способом задайте и другие ограничения.

7. Завершите определение последнего ограничения кнопкой **ОК**.

Установите **«Поиск решения линейных задач симплекс-методом»**, если в вашей модели присутствуют только линейные зависимости. Процесс поиска решения будет ускорен.

Решить задачу линейной оптимизации:

Найти значения переменных x_1 x_2 x_3 x_4, при которых достигается максимум функции $f = 3x_1 + 2{,}5\,x_2 + 5x_3 + 4x_4$.

При ограничениях:

$$\begin{cases} 3x_1 + 5x_2 + 2x_3 + 4x_4 \le 60 \\ 22x_1 + 14x_2 + 18x_3 + 30x_4 \le 400 \\ 10x_1 + 14x_2 + 8x_3 + 16x_4 \le 128 \\ x_1, x_2, x_3, x_4 \ge 0 \end{cases}$$

Решение задачи состоит из следующих этапов.

Создание электронной таблицы, которая содержит все формулы и исходные данные (рис. 3.2.8).

	A	B	C	D	E
3			Переменные		
4		X1	X2	X3	X4
5		0	0	16	0
6			Нижняя граница переменных		
7		0	0	0	0
8			Верхняя граница переменных		
9					
10			Коэффициенты левой части ограничений		
11		3	5	2	4
12		22	14	18	30
13		10	14	8	16
14			Левая часть ограничения	Знак ограничения	Правая часть ограничения
15			32	<=	60
16			288	<=	400
17			128	<=	128
18					
19			Коэффициенты целевой функции		
20		3	2,5	5	4
21					
22			Целевая функция		
23			80		

Рис. 3.2.8. Электронная таблица решения задачи оптимизации

Ячейки **B5:E5** – изменяемые ячейки. В линейных задачах их начальные значения не влияют на результаты и алгоритм поиска.

Диапазон ячеек **B11:E11** следует заполнить коэффициентами левых частей зависимостей для ограничений.

Диапазон ячеек **E15:E17** надо заполнить значениями, составляющими правые части неравенств.

В диапазон ячеек **B20:E20** следует записать коэффициенты целевой функции.

Для целевой функции в ячейку **C23** записываем формулу: **= СУММПРОИЗВ (B5:E5;B20:E20)**

Для левых частей ограничений в ячейки **C15:C17** вводим формулы, начиная со знака равно, «=»:

= СУММПРОИЗВ (B5:E5;B11:E11);

= СУММПРОИЗВ (B5:E5;B12:E12);

= СУММПРОИЗВ (B5:E5;B13:E13).

Процесс создания собственно оптимизационной модели с помощью надстройки **«Поиск решения»** описан выше.

3.3. Поиск критического пути на графе

Теоретическое введение. Структурные модели широко используются менеджерами в виде моделей сетевого планирования. Подобные модели представляют собой ориентированный граф взаимосвязи выполнения различных работ при реализации какого-либо проекта.

Основными элементами сетевой модели планирования являются: событие, работа, путь. Работа характеризует действия, требующие использования ресурсов. Графически работа отображается стрелкой (дуга графа), которая соединяет два события. Каждая работа имеет определенную продолжительность.

Событием называется результат выполнения одной или нескольких работ. События в сетевой модели – это вершины графа. Всегда имеется начальное событие (начало работы по реализации проекта) и конечное событие (полное завершение проекта). Работы, выходящие из некоторого события, не могут начинаться, пока не будут завершены все работы, входящие в это событие. Путь – это цепочка работ, соединяющих начальную и конечную вершины сетевой модели.

Сетевая модель позволяет установить последовательность работ, которая имеет наибольшую длительность (критический путь), и определяет сроки выполнения всего проекта. Несвоевременное выполнение работ, составляющих критический путь, ведет к срыву выполнения всего проекта.

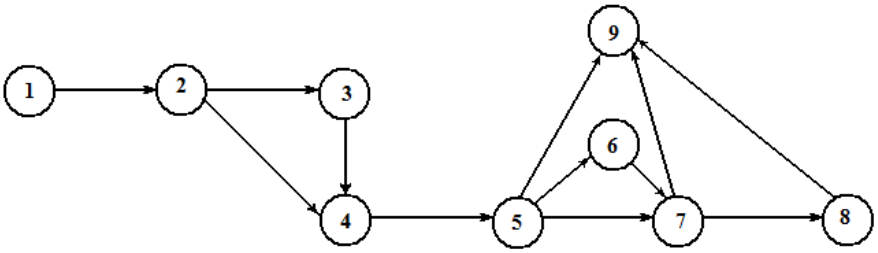


Рис.3.3.1. Сетевая модель

Постановка задачи моделирования. Определить критический путь реализации проекта по схеме, заданной на рис. 3.3.1.

Порядок выполнения работы. Для решения задачи оптимизации средствами Excel предварительно требуется создать таблицу по рис. 3.3.2. В таблице представлены работы и события. Для каждой работы задана ее продолжительность, а также номера узлов графа сетевой модели (рис. 3.3.1–3.3.2), с которыми связана данная работа.

Решением задачи является значения ячеек из диапазона **B4:B17** (Дуга), которые могут принимать значения: 1 или 0. Значение Дуга = 1 – работа включена в критический путь. Диапазон ячеек **B4:B17** является диапазоном **изменяемых ячеек**. Длина пути (ячейка **M13**) рассчитывается по формуле: =СУММПРОИЗВ (Дуга; Длительность). **Требуется найти путь, длина которого → max.** Таким образом, ячейка **M13** является целевой ячейкой.

Для каждого узла, включенного в путь, должно выполняться условие баланса потоков: для начального узла: **Выход – Вход = 1** (узел имеет только выход); для промежуточных узлов: **Выход – Вход = 0** (узел имеет один вход и один выход); для конечного узла: **Выход – Вход = –1** (узел имеет только вход). Необходимо задать значения ограничений для баланса потоков. Эти значения обеспечивают построение цепочки событий (построение

пути от начального к конечному узлу), с выполнением для каждого узла баланса потоков.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	**Поиск критического пути**												
2	**Операции (работы)**							**События**					
3	Работа	Дуга	Начало	Конец	Длительность			Событие	Вершина	Вход	Выход	Баланс	Ограничение баланса потоков
4	Разработка документации		1	2	3			Начало строительства	1	0	0	0	1
5	Подготовка площадки		2	3	8			Разрешение на строительство	2	0	0	0	0
6	Заказ материалов		2	4	5			Заказ материалов	3	0	0	0	0
7	Подготовка кадров		3	4	5			Начало строит. работ	4	0	0	0	0
8	Доставка материалов		3	6	6			Закладка фундамента	5	0	0	0	0
9	Закладка фундамента		4	5	9			Подготовка к строит стен	6	0	0	0	0
10	Ограждение территории		5	6	3			Окончание строит. каркаса здания	7	0	0	0	0
11	Строительство основных элементов		5	7	2			Окончание внешних работ	8	0	0	0	0
12	Благоустройство		5	9	5			Сдача объекта	9	0	0	0	-1
13	Возведение стен		6	7	4				Критический путь				0
14	Обустройство подвала		6	8	7								
15	Обустройство окон		7	8	6								
16	Возведение крыши		7	9	5								
17	Внутрение работы		8	9	10								

Рис. 3.3.2. Электронная таблица для решения задачи

В таблице для каждого события необходимо определить входящий (**Вход**) и выходящий (**Выход**) поток, их баланс (**Выход-Вход**). Для вычисления потока в узлах необходимо использовать функцию вычисления суммы величин, которые удовлетворяют определенному условию.

В Excel такую процедуру выполняет функция **СУММЕСЛИ** (познакомьтесь со справкой по функции). Например, входящий поток для узла определяется формулой **=СУММЕСЛИ (Диапазон концов дуг; Номер узла; Диапазон коэффициентов «Дуга»)**, то есть суммируются потоки по тем дугам, включенным в путь, концы которых соединены с заданной вершиной. Выходящий поток определяется аналогично.

При создании оптимизационной модели в окне **«Параметры поиска решения»** (рис. 3.3.3) следует задать целевую ячейку, диапазон изменяемых ячеек и ограничения. В окне **«Выберите метод решения»** установить **«Поиск решения линейной задачи…»**.

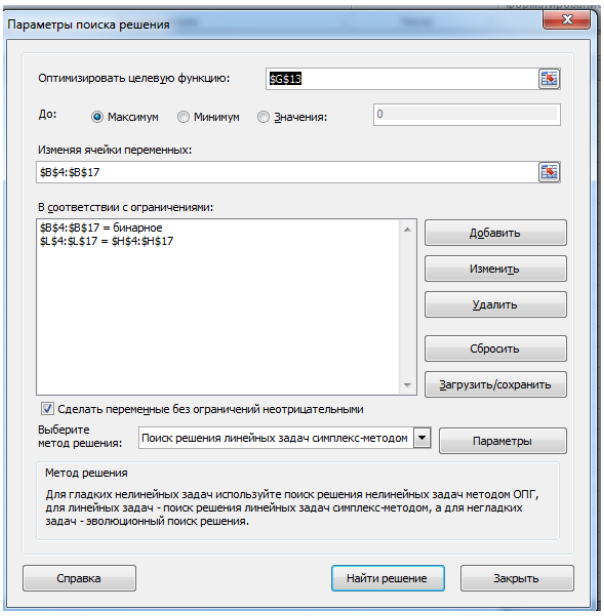


Рис. 3.3.3. Оптимизационная модель

3.4. Идентификация параметров математической модели

При построении математических моделей обычно выбирается класс математических объектов, которые в принципе могут отражать количественные характеристики свойств моделируемого объекта. Один из способов построения математических моделей основан на использовании законов, в математической форме (в виде уравнений), описывающих процессы в объекте.

При этом значения лишь некоторых параметров математической модели могут быть определены путем прямых измерений свойств объекта моделирования. Таким способом можно, например, определить массу или размеры объекта. Однако далеко не все параметры объекта допускают прямые измерения, некоторые из них могут быть определены только косвенно по данным экспериментов. Если не заданы значения параметров, то математическая модель остается неопределенной и не пригодной для исследования свойств объекта моделирования. Вопрос об определении па-

раметров модели является одним из важнейших в математическом моделировании.

В настоящей работе определяются параметры модели движения тела в среде с сопротивлением на основе данных измерений. Модель движения тела под действием силы тяжести и силы сопротивления имеет вид:

$$m\frac{dV}{dt} = mg - kV, \qquad \frac{dx}{dt} = V,$$

начальные условия:

$$V(t=0) = 0; \qquad x(t=0) = 0.$$

Здесь: *m* – масса тела, *V* – скорость движения тела, x – координата тела, g – ускорение свободного падения, *k* – коэффициент сопротивления движению, t – время. Тело считается материальной точкой. При построении модели принят линейный закон зависимости силы сопротивления от скорости.

Постановка задачи моделирования. Для данной модели необходимо определить значение параметра *k* по данным измерений параметров движения тела. Пусть результаты измерений представлены в виде массива значений координаты *x(t)*, полученных при измерения в определенные моменты времени t_n. Причем $t_n = t_{n-1+}\tau..$, где τ – шаг по времени (таблица 3.4.1).

	A	B	C	D	E	F	G
1	**τ**	**0,5**	**t**	**V(t)**	**X(t)**	**Хэксп**	**(X-Хэксп)^2**
2			0,00	0,00	0,00	0,0	0,0000
3	**k**	**2,50**	0,50	2,18	0,55	0,3	0,0601
4			1,00	3,15	1,88	2,0	0,0150
5	**g**	**9,81**	1,50	3,58	3,56	3,9	0,1155
6			2,00	3,77	5,40	5,5	0,0103
7	**m**	**1**	2,50	3,86	7,31	7,1	0,0423
8			3,00	3,90	9,24	9,5	0,0656
9	**Σ(X-Хэксп)^2**	**0,9061**	3,50	3,91	11,20	11,0	0,0382
10			4,00	3,92	13,15	12,5	0,4269
11			4,50	3,92	15,11	15,4	0,0819
12			5,00	3,92	17,08	17,3	0,0503

Рис. 3.4.1. Электронная таблица решения задачи идентификации

Задача идентификации решается в среде электронных таблиц (рис. 3.4.1). Численная реализация математической модели проводится неявным методом Эйлера по следующим расчетным формулам:

$$\frac{V_{n+1}-V_n}{\tau} = g(1-\frac{k}{mg}V_{n+1}); \qquad \frac{x_{n+1}-x_n}{\tau} = \frac{1}{2}(V_{n+1}+V_n);$$

$$V_{n+1} = \frac{\tau g + V_n}{(1+\frac{k}{m}\tau)}; \qquad x_{n+1} = x_n + \frac{1}{2}\tau(V_{n+1}+V_n).$$

Здесь $V_n = V(t_n)$, $x_n = x(t_n)$, $V_{n+1} = V(t_{n+1})$, $x_{n+1} = x(t_{n+1})$, $t_{n+1} = t_n+\tau$. При расчетах по данным формулам в момент времени t_n значения переменных считаются известными.

Значение параметра *k* определяется методом наименьших квадратов с помощью надстройки электронных таблиц **«Поиск решения»** (рис. 3.4.1). При этом целевой функцией, минимум которой необходимо определить, (**B9** – **целевая ячейка**) является сумма квадратов разностей расчетных значений координаты x_n и данных измерений $\tilde{x}_n$: $\sum(x_n-\tilde{x}_n)^2$.

Требуется найти такое значение параметра k (**B3 – изменяемая ячейка**), при котором сумма квадратов разностей экспериментальных и расчетных значений координаты ***x*** имеет минимальное значение. В задаче имеется одно естественное ограничение: *k> = 0*. Так как задача нелинейная, необходимо установить метод решения **«Поиск решения нелинейных задач методом ОПГ»** (рис. 3.4.2) и необходимо в таблице по рис. 1 задать **начальное приближение для значения *k.***

Ограничение задается в специальном окне (кнопка **«Добавить»**, рис. 3.4.3). Задача поиска оптимального значения k решается методом итераций, поэтому следует задать некоторое начальное значение этого параметра. Решение задачи будет найдено автоматически (кнопка **«Выполнить»**, рис. 3.4.2). Результат решения задачи представлен на рис. 3.4.4.

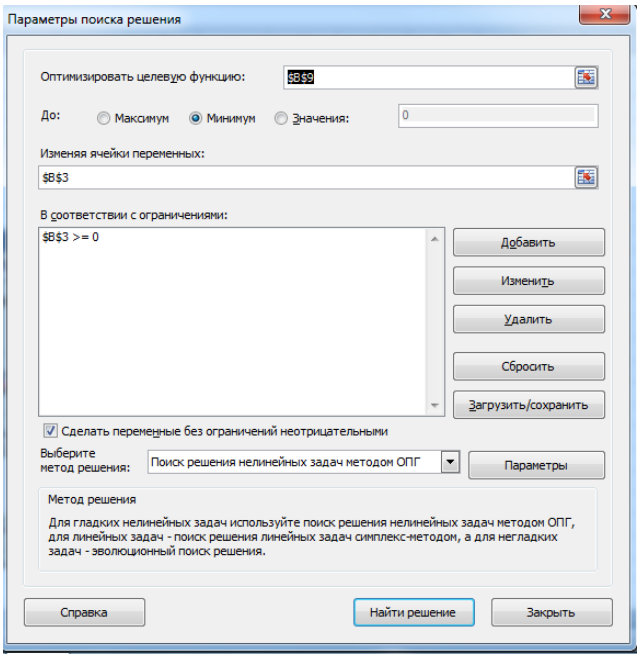


Рис. 3.4.2. Окно надстройки ***«Поиск решения»***

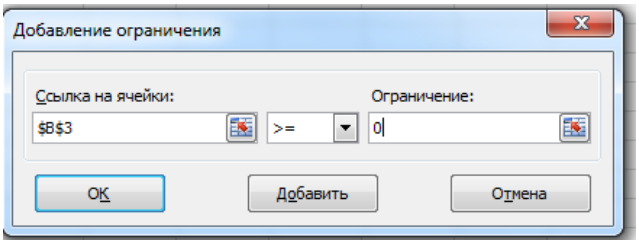


Рис. 3.4.3. Окно ***«Добавление ограничения»***

Таблица 3.4.1

Результаты измерений

t	Хэксп
0,00	0,0
0,50	0,3
1,00	2,0
1,50	3,9
2,00	5,5
2,50	7,1
3,00	9,5
3,50	11,0
4,00	12,5
4,50	15,4
5,00	17,3

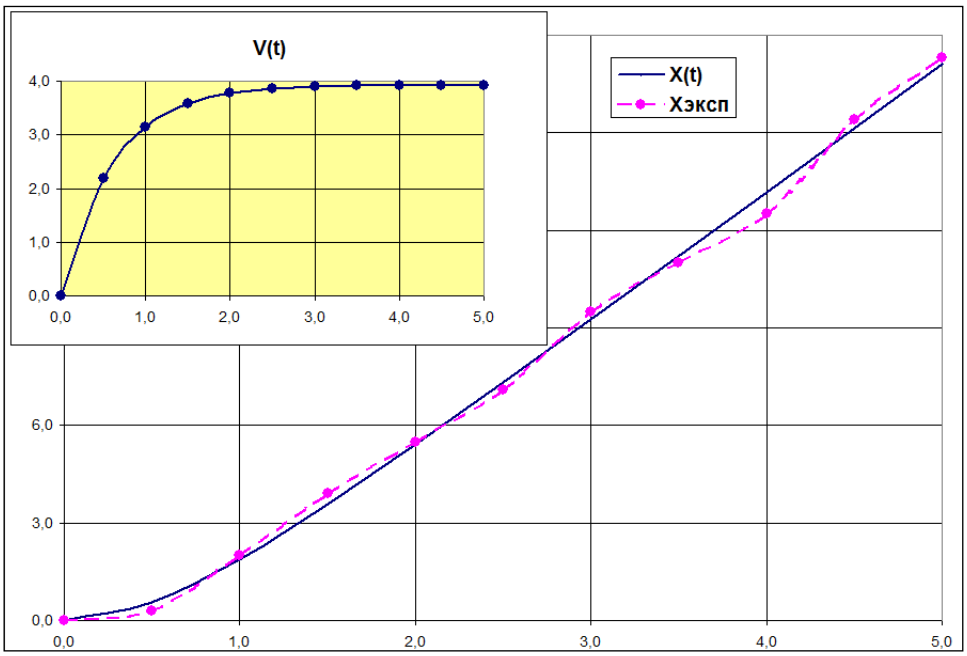


Рис. 3.4.4. Результат решения задачи идентификации

3.5. Построение регрессионной модели

Одно из характерных явлений современной науки – это переход к изучению **сложных, больших и плохо организованных систем.** В начале XX века на основе методов математической статистики были сделаны первые шаги по изучению таких систем, в которых нельзя выделить отдельные физически однородные явления и разграничить их действие.

Математическое моделирование с использованием законов, описывающих протекающие в подобных системах процессы, приводит к весьма сложным нелинейным системам дифференциальных уравнений, в которых параметры либо заданы приближенно, либо неизвестны. В то же время подобные системы достаточно просто поддаются экспериментальному изучению.

В этом случае при построении модели реализуется чисто **эмпирический подход – устанавливается формальная количественная связь** между свойствами объекта и определяющими их факторами в виде какой-либо формальной математической зависимости (т.е. построение уравнения регрессии или регрессионной модели).

Пусть эксперимент поставлен и проведен, результаты наблюдений получены и зафиксированы. Теперь их необходимо представить в какой-то компактной форме, удобной для практического применения и анализа.

Пусть явление или процесс характеризуется только двумя величинами ***X*** и ***Y,*** значения которых фиксируются путем измерений в процессе эксперимента и всегда имеют случайный характер. Задачей исследования является установление так называемой статистической зависимости между случайными величинами ***X*** и ***Y (Y(X))***.

Выбор той или иной формы зависимости определяет точность, с которой модель описывает реальную действительность. Регрессионный анализ сводится к тому, что на основе экспериментальных данных определяются коэффициенты модели и делается оценка ее адекватности.

Рассмотрим простой пример. Предположим, что выполнено *N* измерений случайных величин ***X*** и ***Y*** – (x_i, y_i). Допустим, что зависимость Y от X предполагается линейной: $f(x) = a\,x + b$.

Результаты измерений (x_i, y_i) всегда представляют собой случайные величины. Требуется построить такую линейную зависимость, которая наилучшим образом приближает результаты наблюдений. Таким образом, требуется определить параметры линейной функции ***a*** и ***b***.

Для решения данной задачи используется **метод наименьших квадратов (МНК)**. Параметры линейной функции должны иметь такие значения, чтобы в целом точки $(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)$ лежали как можно ближе к прямой.

Назовем отклонением разность: $f(x_i) - y_i$. Подберем параметры ***a*** и ***b*** таким образом, чтобы сумма квадратов отклонений была минимальна:

$$\sum_{i=1}^{N} (f(x_i) - y_i)^2 \rightarrow \min .$$

В этом и состоит требование метода наименьших квадратов. Данная сумма есть функция искомых параметров *a* и *b:*

$$\Phi(a,b) = \sum_{i=1}^{N} ((a \cdot x_i + b) - y_i)^2 .$$

Постановка задачи моделирования. По заданным значениям (x_i, y_i), полученным в результате измерений (таблица 3.5.1), методом наимень-

ших квадратов определить параметры линейной регрессионной модели и оценить ее адекватность.

Подобным образом метод наименьших квадратов применяется для нахождения параметров линейной и нелинейной регрессии.

Для оценки адекватности регрессионной модели используется **коэффициент детерминации**, который характеризует соответствие полученной зависимости имеющимся эмпирическим данным:

$$D = 1 - \frac{\sigma^2_{ост}}{\sigma^2_{общ}} \ .$$

Величина $\sigma^2_{общ}$ – общая дисперсия, характеризует разброс наблюдаемых величин y_i относительно среднего значения $\bar{y}$; $\sigma^2_{ост}$ – остаточная дисперсия, характеризует отклонения наблюдаемых величин $\boldsymbol{y}_i$ от значений, полученных по регрессионной зависимости:

$$\sigma^2_{общ} = \frac{\sum_{i=1}^{N}(y_i - \bar{y})^2}{N-1}, \qquad \sigma^2_{ост} = \frac{\sum_{i=1}^{N}(y_i - f(x_i))^2}{N-1}.$$

Если *D = 1,* значит $\sigma^2_{ост} = 0$, т.е. все наблюдаемые точки соответствуют построенной регрессионной зависимости. В противном случае при *D = 0,* регрессионная связь между величинами отсутствует (В Excel коэффициент детерминации *D* обозначается R^2**)**.

	A	B	C	D	E	F
1	Σ(F(Xi)-Yi)^2			Xi	Yi	(F(Xi)-Yi)^2
2	64,4			0	8,375	6,17
3				-1	3,377	2,29
4	a	1,001916		-2	2,980	0,82
5	b	5,891839		-3	2,457	0,18
6				-4	1,726	0,03
7				-5	1,308	0,18

Рис. 3.5.1. Решение задачи путем поиска оптимального значения параметров a, b с помощью надстройки «Поиск решения»

Порядок выполнения работы. Первоначально построим линейную зависимость. Для исходных данных (вариант 1, таблица 3.5.1) выполнить поиск значений параметров линейной функции с помощью надстройки **«Поиск решения»**. Таблица для решения задачи представлена на

рис. 3.5.1. Целевой функцией является зависимость:

$$\Phi(a,b) = \sum_{i=1}^{N} ((a \cdot x_i + b) - y_i)^2.$$

Таблица 3.5.1

Исходные данные для построения линейной регрессионной модели

Вариант 1		Вариант 2		Вариант 3		Вариант 4		Вариант 5		Вариант 6	
Xi	**Yi**	**Xi**	**Yi**	**Xi**	**Yi**	**Xi**	**Yi**	**Xi**	**Yi**	**Xi**	**Yi**
0	8,38	0	0,04	–10	109,41	–10	71,24	0	51,3	0	3,48
–1	3,38	–1	0,61	–9	113,98	–9	181,78	1	16,7	1	9,61
–2	2,98	–2	2,47	–8	112,02	–8	135,27	2	26,6	2	0,52
–3	2,46	–3	7,95	–7	42,95	–7	108,83	3	55,9	3	5,88
–4	1,73	–4	8,46	–6	34,33	–6	40,02	4	40,9	4	0,97
–5	1,31	–5	8,97	–5	39,30	–5	150,11	5	67,9	5	1,98
–6	–2,00	–6	9,03	–4	27,61	–4	109,82	6	42,1	6	6,46
–7	–1,01	–7	15,72	–3	39,07	–3	122,88	7	76,2	7	5,25
–8	–0,46	–8	12,09	–2	49,67	–2	38,20	8	93,9	8	2,87
–9	–4,28	–9	18,10	–1	20,25	–1	83,14	9	56,7	9	4,54
–10	–5,57	–10	21,73	0	77,36	0	29,89	10	64,3	10	9,70
–11	–2,55	–11	18,03	1	65,91	1	50,73	11	93,0	11	3,68
–12	–5,08	–12	22,03	2	22,11	2	95,02	12	81,1	12	8,78
–13	–8,95	–13	23,88	3	5,39	3	23,46	13	94,4	13	2,94
–14	–9,29	–14	26,73	4	36,70	4	41,85	14	103,4	14	2,71
–15	–10,15	–15	32,70	5	8,40	5	97,00	15	109,9	15	1,71
–16	–7,76	–16	31,98	6	9,98	6	92,41	16	124,8	16	3,17
–17	–10,67	–17	34,62	7	12,65	7	94,75	17	121,9	17	9,59
–18	–11,65	–18	35,83	8	–29,04	8	82,81	18	105,9	18	9,48
–19	–14,42	–19	38,16	9	–22,04	9	84,73	19	103,7	19	0,63
–20	–14,63	–20	42,69	10	–41,30	10	–32,83	20	115,6	20	0,10
–21	–12,62	–21	40,72	11	–36,58	11	–34,06	21	147,6	21	7,52
–22	–16,12	–22	44,55	12	17,82	12	–21,88	22	146,5	22	0,28
–23	–18,95	–23	46,04	13	–43,87	13	42,10	23	163,2	23	5,30
–24	–15,03	–24	44,62	14	–4,14	14	58,45	24	169,9	24	5,54
–25	–18,65	–25	50,11	15	–40,19	15	25,91	25	176,1	25	8,15
–26	–19,78	–26	50,71	16	–43,22	16	–15,82	26	162,8	26	4,60
–27	–21,30	–27	52,83	17	–19,87	17	–70,12	27	143,6	27	2,35
–28	–23,83	–28	56,79	18	–60,30	18	–57,70	28	151,6	28	2,29
–29	–23,79	–29	60,49	19	–87,49	19	41,36	29	149,2	29	7,40
–30	–24,94	–30	57,20	20	–91,66	20	–69,63	30	185,2	30	0,62

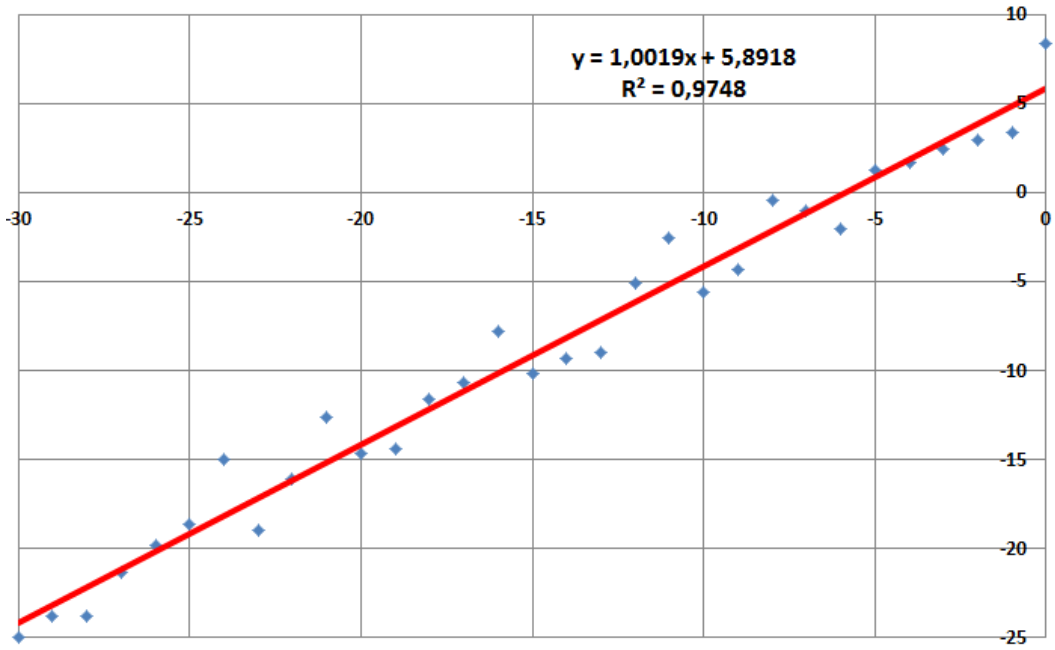


Рис. 3.5.2. Линия тренда и величина достоверности (коэффициент детерминации R^2*).*

Задача определения параметров регрессионной зависимости может быть решена средствами электронных таблиц путем построения **линии тренда** (рис. 3.5.2). При этом ЭТ реализуют метод наименьших квадратов автоматически.

При построении уравнения регрессии кроме линейной функции могут быть использованы зависимости, которые представлены в таблице 3.5.2.

Таблица 3.5.2

Виды регрессионных зависимостей, генерируемых MS Excel

Тип зависимости	Уравнение	Параметры
Линейная	$Y = ax + b$	a, b
Полиномиальная	$Y = a_1x + a_2x^2 + \ldots + a_nx^n + b$	$a_1, \ldots a_n$, b
Логарифмическая	$Y = a + b \ln(\mathbf{x})$	a, b
Экспоненциальная	$Y = a \exp(bx)$	a, b
Степенная	$Y = a\, x^b$	a, b

Для построения уравнения регрессии необходимо создать таблицу числовых данных. Затем по таблице построить диаграмму **«точечная»**. При этом выбрать вариант диаграммы, который отображает табличные данные только в виде **точек**. Далее необходимо левой кнопкой мыши выделить

любую точку и правой кнопкой вызвать контекстное меню. В нем выбрать команду **«Добавить линию тренда».** В открывшемся диалоговом окне **«Линия тренда»** (рис. 3.5.3) выбрать **тип зависимости**.

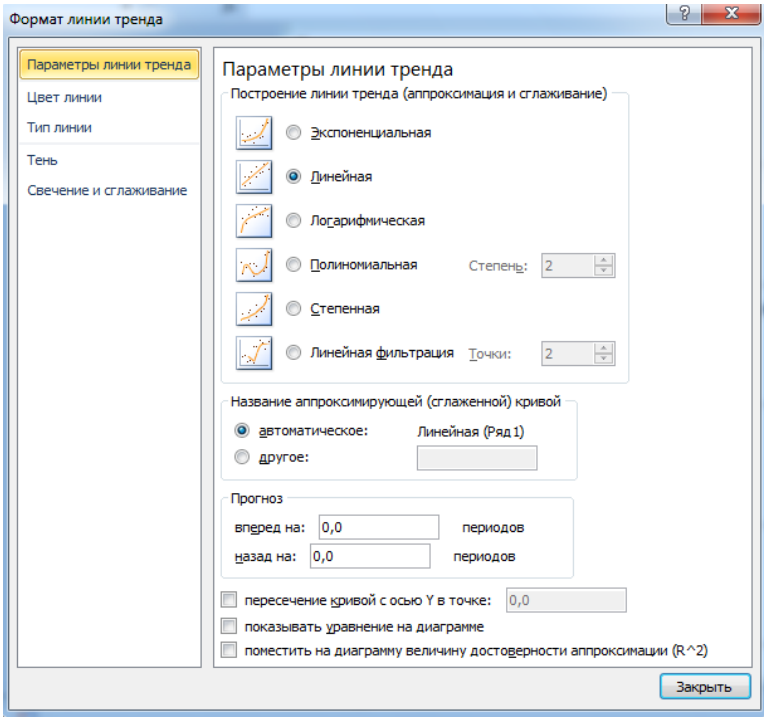


Рис. 3.5.3. Диалоговое окно «Линия тренда»

Выбрать опции **«Показывать уравнение»** и **«Поместить величину достоверности».** Для исходных данных по таблице 1 постройте **линию тренда, тип зависимости – линейная** (рис. 3.5.3). Параметры модели и величина достоверности должны совпадать с данными по рис. 3.5.1–3.5.2.

Анализ результатов моделирования. После построения таблицы и линии тренда (рис. 3.5.2), выполните расчеты для других вариантов исходных данных (таблица 3.5.1). Сделайте выводы по результатам выполнения расчетов. Сравните значение коэффициента достоверности и характер расположения точек относительно линии тренда. Сделайте вывод о достоверности.

Дополнительное задание. Выполнить построение нелинейных регрессионных моделей средствами ЭТ. На основе значения **коэффициента детерминации** выбрать наиболее адекватную модель. Исходные данные представлены в таблице 3.5.3.

Таблица 3.5.3

Исходные данные для построения нелинейной регрессионной модели

Вариант 1		Вариант 2		Вариант 3		Вариант 4		Вариант 5	
Xi	**Yi**	**Xi**	**Yi**	**Xi**	**Yi**	**Xi**	**Yi**	**Xi**	**Yi**
1	0,30	5	–53,50	–2	0,74	5	–1,84	1	0,84
2	0,42	4,9	–50,02	–1,9	0,77	6	–1,99	1,1	0,89
3	0,52	4,4	–34,79	–1,8	0,81	7	–2,11	1,2	0,93
4	0,60	3,9	–22,86	–1,7	0,85	8	–2,22	1,3	0,96
5	0,67	3,4	–13,85	–1,6	0,90	9	–2,31	1,4	0,99
6	0,74	2,9	–7,40	–1,5	0,94	10	–2,40	1,5	1,00
7	0,79	2,4	–3,11	–1,4	0,99	11	–2,47	1,6	1,00
8	0,85	1,9	–0,63	–1,3	1,04	12	–2,54	1,7	0,99
9	0,90	1,4	0,43	–1,2	1,10	13	–2,61	1,8	0,97
10	0,95	0,9	0,44	–1,1	1,15	14	–2,67	1,9	0,95
11	1,00	0,4	–0,23	–1	1,21	15	–2,72	2	0,91
12	1,04	–0,1	–1,20	–0,9	1,28	16	–2,77	2,1	0,86
13	1,08	–0,6	–2,09	–0,8	1,34	17	–2,82	2,2	0,81
14	1,12	–1,1	–2,54	–0,7	1,41	18	–2,87	2,3	0,75
15	1,16	–1,6	–2,15	–0,6	1,48	19	–2,91	2,4	0,68
16	1,20	–2,1	–0,57	–0,5	1,56	20	–2,95	2,5	0,60
17	1,24	–2,6	2,59	–0,4	1,64	21	–2,99	2,6	0,52
18	1,27	–3,1	7,70	–0,3	1,72	22	–3,03	2,7	0,43
19	1,31	–3,6	15,13	–0,2	1,81	23	–3,06	2,8	0,34
20	1,34	–4,1	25,26	–0,1	1,90	24	–3,10	2,9	0,24
21	1,38	–4,6	38,47	0	2,00	25	–3,13	3	0,14
22	1,41	–5,1	55,13	0,1	2,10	26	–3,16	3,1	0,04
23	1,44	–5,6	75,61	0,2	2,21	27	–3,19	3,2	–0,06
24	1,47	–6,1	100,29	0,3	2,32	28	–3,22	3,3	–0,16
25	1,50	–6,6	129,55	0,4	2,44	29	–3,25	3,4	–0,26
26	1,53	–7,1	163,76	0,5	2,57	30	–3,28	3,5	–0,35
27	1,56	–7,6	203,29	0,6	2,70	31	–3,30	3,6	–0,44
28	1,59	–8,1	248,52	0,7	2,84	32	–3,33	3,7	–0,53
29	1,62	–8,6	299,83	0,8	2,98	33	–3,35	3,8	–0,61
30	1,64	–9,1	357,59	0,9	3,14	34	–3,38	3,9	–0,69
31	1,67	–9,6	422,17	1	3,30	35	–3,40	4	–0,76

3.6. Построение модели на основе корреляционного анализа

При построении регрессионной модели формируется перечень факторов, которые могут влиять на свойства объекта. Исходя из содержательного описания объекта, выявляются переменные – параметры модели и производится их разделение на зависимые, малозначимые и взаимозависимые.

В модель включаются все факторы, скорее всего с избытком, которые, по мнению исследователя, могут оказать какое-либо влияние на свойства объекта. Таких факторов может оказаться достаточно много. Какие факторы являются главными, а какие второстепенными необходимо выяснить.

Пусть исходная информация о свойствах объекта, полученная в ходе экспериментов, представляется в таблице 3.6.1. Значения для выполнения лабораторной работы содержатся ниже в таблице 3.6.1.

Таблица 3.6.1

X_1	X_2	X_3	….	X_n	Y
……	……	……	……	……	……

Для уменьшения количества факторов необходимо выявить и исключить **взаимозависимые** и **малозначимые** факторы, которые оказывают несущественное влияние на величину зависимой переменной ***Y***. С этой целью строится корреляционная матрица следующего вида (таблица 3.6.2).

Таблица 3.6.2

Корреляционная матрица

	X_1	X_2	X_3	X_4	X_5	Y
X_1	**1**	Rx_1x_2	Rx_1x_3	Rx_1x_4	Rx_1x_5	Rx_1y
X_2		**1**	Rx_2x_3	Rx_2x_4	Rx_2x_5	Rx_2y
X_3			**1**	Rx_3x_4	Rx_3x_5	Rx_3y
X_4				**1**	Rx_3x_5	Rx_4y
X_5					**1**	Rx_5y
Y						**1**

Здесь X_i – факторы, ***Y*** – отклик системы. По определению коэффициента корреляции ясно, что данная матрица является симметричной относительно главной диагонали. Коэффициент корреляции вычисляется с помощью функции Excel ***КОРРЕЛ.***

Значение коэффициента корреляции заключены в пределах от – 1 до + 1. В случае $r_{xy} = 0$ связь между *x* и *y* отсутствует. При $r_{xy} = 1$ существует строгая положительная связь. Если $r_{xy} = -1$, то существует строгая отрицательная связь.

Взаимозависимость между факторами x_i, x_j имеет место, если коэффициент парной корреляции $\left|Rx_i x_j\right| \geq 0,7$. Опыт показывает, что один из факторов можно исключить, так как он существенно зависит от другого.

Кроме того, факторы, для которых $|Rx_iy| \leq 0,3$, фактически не связанны с *y* и подлежат исключению как **малозначимые.** Естественно, что далее регрессионная модель строится без учета малозначимых и взаимосвязанных факторов.

Постановка задачи моделирования. Выявить значимые и независимые факторы и на основе данных экспериментов построить регрессионную модель.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1		x1	x2	x3	x4	x5	y	X1	X2	X3	X4	X5	Y
2	x1	1,0000	0,1831	1,0000	0,1831	-0,0977	0,5694	0,16	2,62	8,32	0,12	4,53	8,54
3	x2		1,0000	0,1831	1,0000	-0,0756	0,0894	0,31	3,42	9,71	0,92	3,02	5,62
4	x3			1,0000	0,1831	-0,0977	0,5694	0,46	2,73	11,09	0,23	0,34	0,59
5	x4				1,0000	-0,0756	0,0894	0,60	2,58	12,48	0,08	0,32	0,70
6	x5					1,0000	0,6866	0,75	2,54	13,86	0,04	0,15	0,52
7	y						1,0000	0,89	3,27	15,25	0,77	0,06	0,50
8								1,04	2,57	16,64	0,07	3,09	5,92

Рис. 3.6.1. Корреляционная матрица и фрагмент таблицы исходных данных

Порядок выполнения лабораторной работы. Работа выполняется в среде электронных таблиц. Исходными данными является таблица результатов экспериментов с объектом моделирования (таблица 3.6.3), которая содержит значения факторов $\boldsymbol{X_1} - \boldsymbol{X_5}$ и отклика ***Y***. Применив корреляцион-

ный анализ, необходимо выявить и исключить малозначимые и взаимозависимые факторы.

С этой целью построить корреляционную матрицу (таблица 3.6.2). При построении корреляционной матрицы использовать функцию электронных таблиц ***КОРРЕЛ***.

На основе корреляционной матрицы выявить и исключить малозначимые и взаимозависимые факторы. После чего необходимо построить новую таблицу данных (таблица 3.6.3), которая не содержит исключенных факторов.

Таблица 3.6.3

Таблица без значений исключенных факторов

X_1	X_5	Y
…	…	…
…	…	…

Далее необходимо методом наименьших квадратов создать регрессионную модель, т.е. определить значения параметров $b_{i,j}$, при этом необходимо использовать надстройку **«Поиск решения»**. При построении регрессионной модели используем многочлен 2-й степени: $f(X_1, X_5) = b_0 + b_1 X_1 + b_2 X_5 + b_{12} X_1 X_5$. При этом целевая функция имеет вид:

$$\sum_{i=1}^{N} (f(X_1, X_5) - Y_i)^2 \to \min .$$

Дальнейшие вычисления проводятся аналогично работе **«Построение регрессионной модели»** (таблица 3.6.4).

Таблица 3.6.4

Таблица для расчёта коэффициентов многочлена

$$f(X_1, X_5) = b_0 + b_1 X_1 + b_2 X_5 + b_{12} X_1 X_5$$

b_0	b_1	b_2	b_{12}	X1	X5	Y	$(Y - f(X_1,X_5))^2$	$\sum(Y - f(X_1,X_5))^2$
1	1	1	1	….	….	….	…….	**Вычисление целевой функции**

Затем необходимо графически отобразить **поверхность отклика** по полученной регрессионной зависимости средствами электронных таблиц (диаграмма **«Поверхность»**). Для построения поверхности отклика необходимо на отдельном листе ЭТ построить таблицу значения функции отклика (рис. 3.6.2). Диапазон изменения факторов определить по таблице. При этом необходимо использовать тип диаграммы – **«Поверхность»** (рис. 3.6.3).

	A	B	C	D	E	F	G	H	I	J	K	L
1	**b0**	**b1**	**b2**	**b12**								
2	**-0,50421**	**1,002068**	**2,000605**	**-0,25014**								
3		0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
4	0	-0,50421	-0,30415	-0,10409	0,095974	0,296035	0,496095	0,696156	0,896216	1,096277	1,296337	1,496398
5	0,1	-0,404	-0,20644	-0,00888	0,188677	0,386236	0,583795	0,781354	0,978913	1,176472	1,374032	1,571591
6	0,2	-0,30379	-0,10874	0,086322	0,28138	0,476437	0,671495	0,866553	1,06161	1,256668	1,451726	1,646784
7	0,3	-0,20359	-0,01103	0,181526	0,374082	0,566639	0,759195	0,951751	1,144308	1,336864	1,52942	1,721976
8	0,4	-0,10338	0,086675	0,27673	0,466785	0,65684	0,846895	1,03695	1,227005	1,41706	1,607114	1,797169
9	0,5	-0,00317	0,18438	0,371934	0,559488	0,747041	0,934595	1,122148	1,309702	1,497255	1,684809	1,872362
10	0,6	0,097034	0,282086	0,467138	0,65219	0,837242	1,022294	1,207347	1,392399	1,577451	1,762503	1,947555
11	0,7	0,19724	0,379791	0,562342	0,744893	0,927444	1,109994	1,292545	1,475096	1,657647	1,840197	2,022748
12	0,8	0,297447	0,477497	0,657546	0,837595	1,017645	1,197694	1,377743	1,557793	1,737842	1,917892	2,097941
13	0,9	0,397654	0,575202	0,75275	0,930298	1,107846	1,285394	1,462942	1,64049	1,818038	1,995586	2,173134
14	1	0,497861	0,672907	0,847954	1,023001	1,198047	1,373094	1,54814	1,723187	1,898234	2,07328	2,248327

Рис. 3.6.4. Фрагмент таблицы значений функции отклика

Коэффициенты многочлена определить по методу наименьших квадратов, используя надстройку **«Поиск решения»**. Для решения этой задачи необходимо составить таблицу следующего вида (таблица 3.6.4):

Так как решение нелинейной задачи оптимизации требует задания начальных значений искомых параметров, то предварительные значения коэффициентов задайте по таблице 3.6.4. Изменяемые ячейки – это ячейки, содержащие значения коэффициентов $b_{i,j}$. Целевая ячейка содержит сумму квадратов разностей. Ограничений нет.

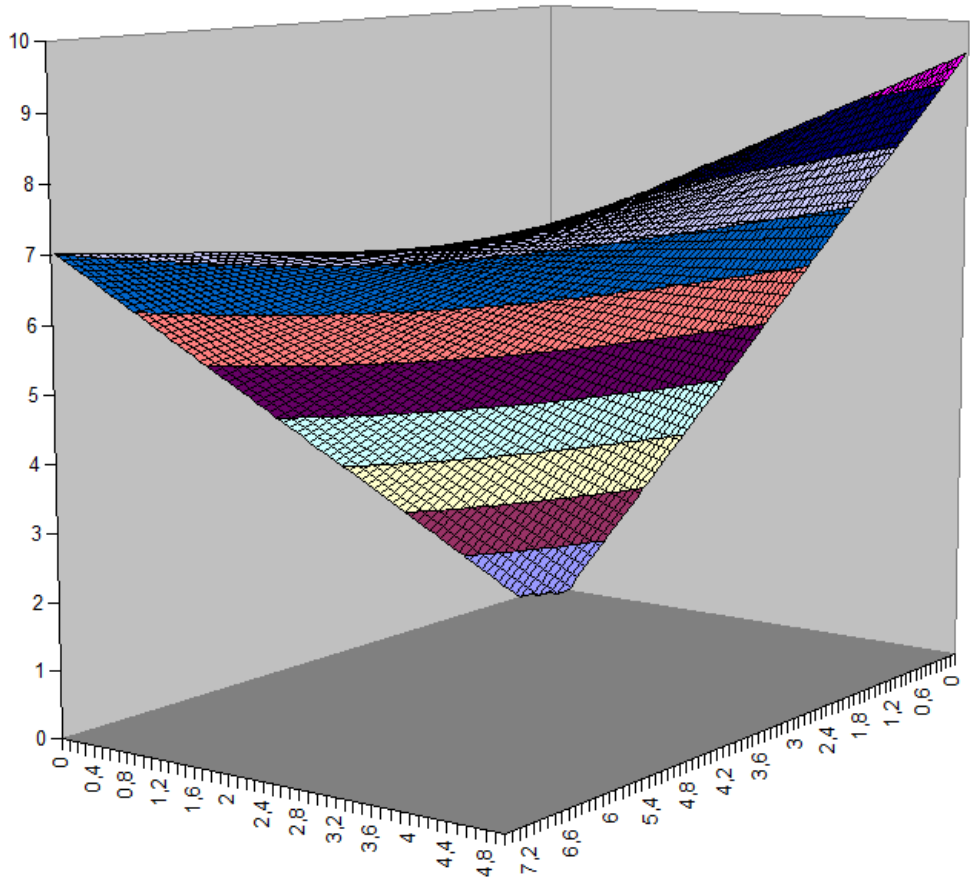


Рис. 3.6.3. Поверхность отклика

Задание по работе. По таблице 3.6.5 необходимо построить корреляционную матрицу, выявить значимые факторы и поверхность отклика.

Таблица 3.6.5

Результаты измерений свойств объекта

X1	***X2***	***X3***	***X4***	***X5***	***Y***
0,16	2,62	8,32	0,12	4,53	8,54
0,31	3,42	9,71	0,92	3,02	5,62
0,46	2,73	11,09	0,23	0,34	0,59
0,60	2,58	12,48	0,08	0,32	0,70
0,75	2,54	13,86	0,04	0,15	0,52
0,89	3,27	15,25	0,77	0,06	0,50
1,04	2,57	16,64	0,07	3,09	5,92
1,18	2,98	18,02	0,48	2,24	4,50
1,33	3,19	19,41	0,69	0,01	0,85
1,48	2,79	20,79	0,29	0,31	1,47
1,62	3,43	22,18	0,93	0,41	1,78
1,77	3,22	23,56	0,72	1,80	4,07

Окончание табл. 3.6.5

X1	*X2*	*X3*	*X4*	*X5*	*Y*
1,91	2,69	24,95	0,19	4,15	7,72
2,06	2,61	26,34	0,11	0,02	1,58
2,20	2,51	27,72	0,01	0,23	2,03
2,35	2,96	29,11	0,46	0,06	1,93
2,49	2,91	30,49	0,41	0,10	2,13
2,64	3,32	31,88	0,82	4,37	7,99
2,79	3,05	33,27	0,55	2,27	5,24
2,93	2,70	34,65	0,20	4,42	8,03
3,08	2,83	36,04	0,33	0,37	3,03
3,22	2,91	37,42	0,41	0,53	3,36
3,37	2,64	38,81	0,14	1,56	4,67
3,51	3,20	40,19	0,70	0,95	4,08
3,66	3,22	41,58	0,72	0,02	3,18
3,81	2,54	42,97	0,04	0,42	3,74
3,95	2,80	44,35	0,30	0,13	3,58
4,10	3,27	45,74	0,77	0,26	3,85
4,24	3,25	47,12	0,75	0,28	4,00
4,39	2,52	48,51	0,02	2,24	5,91
4,53	2,98	49,89	0,48	2,72	6,39
4,68	3,38	51,28	0,88	1,09	5,08
4,82	2,70	52,67	0,20	0,76	4,93
4,97	3,10	54,05	0,60	0,25	4,66
5,12	2,85	55,44	0,35	0,35	4,86
5,26	2,89	56,82	0,39	1,02	5,46
5,41	2,68	58,21	0,18	0,06	4,94
5,55	3,49	59,59	0,99	0,61	5,43
5,70	3,02	60,98	0,52	1,21	5,89
5,84	2,99	62,37	0,49	2,60	6,75
5,99	2,80	63,75	0,30	1,93	6,46
6,14	2,82	65,14	0,32	0,32	5,79
6,28	3,49	66,52	0,99	1,82	6,56
6,43	2,68	67,91	0,18	0,29	6,04
6,57	3,32	69,29	0,82	0,77	6,35
6,72	3,50	70,68	1,00	0,13	6,26
6,86	2,52	72,07	0,02	1,05	6,66
7,01	3,47	73,45	0,97	2,37	7,10
7,15	2,60	74,84	0,10	4,27	7,56
7,30	3,25	76,22	0,75	0,68	6,92
7,45	2,94	77,61	0,44	0,01	6,95

3.7. Построение модели по плану полного факторного эксперимента

Целью любого эксперимента является получение информации о зависимости некоторых свойств объекта от его параметров, которые предположительно влияют на эти свойства. Эти зависимости могут быть выражены в виде чисто формальных математических соотношений (регрессионных моделей типа **«черный ящик»**).

Наиболее подходящей моделью, которую можно построить на основе обработки результатов эксперимента является модель **«черный ящик»**. При таком подходе различают только входные и выходные переменные, которые разделяются на **факторы** и **отклики**. **Факторы** – это управляемые переменные, значения которых в ходе эксперимента можно менять. Каждый фактор в эксперименте может принимать определенные значения, которые называются **уровнями**. Набор уровней значений факторов определяет одно из возможных состояний системы в эксперименте.

В ходе эксперимента регистрируются определенные величины, характеризующие сущность изучаемого процесса или явления. Такие величины называются **откликами**. С аналитической точки зрения эксперимент представляет собой выявление связи отклика y с рядом факторов x_1, x_2,…x_n. Эта связь в итоге выражается с помощью уравнения регрессии:

$$Y = f(x_1\, x_2,\ldots x_n)$$

Теория планирования эксперимента предполагает получение формальной количественной модели в унифицированном виде, которая может быть пригодна для любых откликов и факторов. Подобной моделью может служить алгебраический многочлен:

$$y = b_0 + \sum_{i=1}^{k} b_i x_i + \sum_{i \neq j}^{k} b_{ij} x_i x_j + \sum_{i=1}^{k} b_{ii} x_i^2 + \ldots,$$

где b – коэффициенты регрессии, k – количество факторов.

Целью эксперимента является получение такого количества точек в пространстве факторов, которое будет с достаточной степенью точности характеризовать отклик (т.е. значение ***y***). Планирование эксперимента при

этом заключается в выборе оптимального количества точек (опытов) и размещении их в пространстве факторов таким образом, чтобы уравнение регрессии было определено с наибольшей точностью.

Основной принцип одного из видов активного эксперимента, который называется **полным факторным экспериментом** (ПФЭ), заключается в том, что каждый фактор **варьируется** в эксперименте **вместе со всеми** остальными факторами. При этом, чтобы исследовать ***k*** факторов на ***m*** уровнях, требуется выполнить m^k опытов.

Если выбранная модель включает в себя только линейные члены и произведения факторов, то для оценки коэффициентов модели используется план экспериментов с варьированием всех ***k*** факторов на двух уровнях. Такие планы ПФЭ называются планами типа 2^k, где $N = 2^k$ число всех возможных испытаний.

Планирование такого эксперимента можно предельно формализовать и упростить. Первоначально требуется преобразовать все факторы к безразмерному виду и нормировать их таким образом, чтобы на одном уровне фактор принимал значение + 1, а на другом уровне значение – 1.

При построении плана эксперимента необходимо определить границы варьирования переменных и их нулевой (средний) уровень. Для каждой переменной задается нулевой уровень x_{i0}, соответствующий наилучшим условиям построения модели, задается интервал варьирования Δx_i, который должен быть достаточно большим для статистической различимости уровней фактора на фоне ошибок его измерения. В качестве уровней варьирования выбираются значения факторов, симметрично расположенных относительно нулевой точки. Нижний и верхний уровни выбираются по соотношениям:

$$x_{iв} = x_{i0} + \Delta x_i \quad .... \quad x_{iв} = x_{i0} + \Delta x_i \, .$$

Для стандартизации и упрощения записи плана проводится нормирование факторов, при этом нижнему уровню фактора будет соответствовать значение – 1, а верхнему уровню значение + 1. Преобразование проводится по формулам:

$$\bar{x}_i = (x_i - x_{i0}) / \Delta x_i \, .$$

Полный факторный эксперимент дает возможность определить коэффициенты регрессии, соответствующие линейным членам, и коэффициенты, соответствующие взаимодействиям факторов. Эффект взаимодействия нескольких факторов проявляется при одновременном варьировании их значений, когда действие одного фактора зависит от уровня.

Формулы вычислений для представленной выше матрицы планирования эксперимента имеют вид:

$$b_0 = \sum_{t=1}^{N} x_0 y_t / N, \quad b_i = \sum_{t=1}^{N} x_{it} y_t / \sum_{t=1}^{N} x_{it}^2, \quad b_{ij} = \sum_{t=1}^{N} (x_i x_j)_t y_t / N.$$

Здесь индекс ***t*** означает номер испытания, ***N*** – общее количество испытаний, остальные обозначения соответствуют представленной выше таблице.

Постановка задачи моделирования. Используя результаты ПФЭ построить регрессионную модель объекта. Вычислить коэффициенты регрессии по результатам испытаний, представленных в таблице 3.7.1.

Таблица 3.7.1

Результаты эксперимента

Номер испытания	**План ПФЭ**				y_t
	$\bar{x}_0$	$\bar{x}_1$	$\bar{x}_2$	$\bar{x}_1\bar{x}_2$	
1	+1	–1	–1	+1	3
2	+1	+1	+1	+1	9
3	+1	–1	+1	–1	6
4	+1	+1	–1	–1	12

Порядок выполнения работы. Работа выполняется в среде электронных таблиц. На основе результатов ПФЭ, представленных в табл. 3.7.1. Средствами электронных таблиц (**тип диаграммы – «Поверхность»**), используя данные таблицы по рис. 3.7.1, построить **поверхность отклика**. Построение поверхности отклика потребует создания значений функции отклика $y = b_0 + b_1 x_1 + b_2 x_2 + b_{12} x_1 x_2$. Диапазон изменения факто-

ров x_1, x_2 **– [– 1; 1].** Шаг таблицы по любому фактору равен *0,1*. Примерная таблица представлена на рис. 3.7.1. (Замечание: значения параметров *b* необходимо вычислить, а не взять с рис. 3.7.1, это весьма просто проверяется).

	A	B	C	D	E	F	G	H	I	J	K
1	Номер испытания	План ПФЭ				y_t		b0	b1	b2	b12
2		$\bar{x}_0$	$\bar{x}_1$	$\bar{x}_2$	$\bar{x}_1\bar{x}_2$			7,5	3	0	-1,5
3	1	1	-1	-1	1	3					
4	2	1	1	1	1	9					
5	3	1	-1	1	-1	6					
6	4	1	1	-1	-1	12					
7											
8		x1/x2	-1	-0,9	-0,8	-0,7	-0,6	-0,5	-0,4	-0,3	-0,2
9		-1	3	3,45	3,9	4,35	4,8	5,25	5,7	6,15	6,6
10		-0,9	3,15	3,585	4,02	4,455	4,89	5,325	5,76	6,195	6,63
11		-0,8	3,3	3,72	4,14	4,56	4,98	5,4	5,82	6,24	6,66
12		-0,7	3,45	3,855	4,26	4,665	5,07	5,475	5,88	6,285	6,69
13		-0,6	3,6	3,99	4,38	4,77	5,16	5,55	5,94	6,33	6,72
14		-0,5	3,75	4,125	4,5	4,875	5,25	5,625	6	6,375	6,75
15		-0,4	3,9	4,26	4,62	4,98	5,34	5,7	6,06	6,42	6,78
16		-0,3	4,05	4,395	4,74	5,085	5,43	5,775	6,12	6,465	6,81
17		-0,2	4,2	4,53	4,86	5,19	5,52	5,85	6,18	6,51	6,84

Рис. 3.7.1. Построение модели по таблице ПФЭ

3.8. Моделирование случайного события

Моделирование случайных процессов и событий – это одно из важнейших направлений современного компьютерного моделирования – имитационного моделирования. Само понятие **«случайный»** является фундаментальным. Случайные события наблюдаются в технических, экологических, экономических, социальных и других системах. Таким образом, знакомство с подобными процессами и их характеристиками представляется актуальной задачей.

Решение задач моделирования случайных событий позволяет на наглядных примерах познакомиться с понятием вероятности и построить имитационную модель.

В качестве классической задачи стохастического моделирования обычно рассматривается задача вычисления площади круга методом Монте-Карло. Однако метод Монте-Карло является вычислительным приемом,

когда приближенное решение детерминированной задачи получается с помощью генерации последовательности случайных чисел с равномерным законом распределения.

Рассмотрим ряд задач моделирования случайных событий, которые легко могут быть реализованы с использованием электронных таблиц Excel. Моделирование случайных событий связано с генерацией случайных чисел с равномерным законом распределения на интервале (0; 1), которое реализуется функцией электронных таблиц **СЛЧИС ()**.

Рассмотрим задачу. Пусть некоторое случайное событие ***A*** наступает с вероятностью ***P***. Пусть имеется возможность генерировать последовательность значений случайной величины с равномерным распределением на интервале (0; 1): $x_1, x_2, \ldots x_n$. Определим, что событие ***A*** наступает **(*A = 1*)** в том случае, если значение сгенерированной случайной величины удовлетворяет неравенству $x_i \le P$, в противном случае исходом испытания является противоположное событие ***НеА*** **(*A = 0*).**

Рассмотрим построение компьютерной модели реализации случайного события с заданной вероятностью. Для решения задачи необходимо построить электронную таблицу по рисунку 3.8.1. Исходными данными для модели является значение вероятности реализации события ***P*** (**ячейка A2**). В колонку В (диапазон **B2:B101**) электронной таблицы вводится функция **СЛЧИС(),** которая генерирует последовательность случайных чисел x_i в диапазоне (0,1) с равномерным законом распределения. В ячейку **C2** вводится функция **ЕСЛИ()**, которая возвращает значение, равное 1, если выполняется неравенство $x_i \le p$, или значение 0 в противном случае: **=ЕСЛИ(B2<=$A$2;1;0)**. Далее формула копируется в ячейки столбца С (диапазон ячеек **C2:C101**).

В ячейках **D2** и **E2** с помощью функции **СЧЕТЕСЛИ ()** подсчитывается количество реализаций событий ***A (A = 1)*** и ***НЕА (A = 0);*** (**= СЧЁТЕСЛИ(C1:C101;1)**, **= СЧЁТЕСЛИ(C1:C101;0)**). В ячейках **D4** и **E4** необходимо рассчитать частоту реализации событий ***A*** и ***НЕА*** с помощью фор-

мул: =**D2/(E2+D2); =E2/(E2+D2).** Сумму частот рассчитать и поместить в ячейку **F4**. Естественно, сумма должна получиться равной единице.

	A	B	C	D	E
1	**P**	**Xi**	**A**	**Кол-во событий A**	**Кол-во событий НЕ A**
2	**0,75**	0,1738	1	**77**	**23**
3		0,1429	1	**Частота A**	**Частота НЕ A**
4		0,1960	1	**0,77**	**0,23**
5		0,2181	1		

Рис. 3.8.1. Таблица моделирования случайного события

Можно провести ряд экспериментов (на отдельных листах книги Excel) при разном количестве реализаций события: 20, 100; 500, 1 000, 2 500 и сопоставить значение заданной вероятности ***P*** события и частоты его реализации в эксперименте. Чем больше реализаций событий, тем больше частота приближается к вероятности.

Это – закон больших чисел (так называемая устойчивость частот). Например, Карл Пирсон, английский математик, статистик, биолог и философ, выполнил 24 000 бросаний монеты, герб выпал 12 012 раз, т.е. частота события равна 0,5005. Жорж Луи Бюффон, французский естествоиспытатель и популяризатор науки, провел 4 040 подбрасываний монеты. При этом получил частоту выпадения орла 0,508.

3.9. Моделирование полной группы случайных событий

Следующая задача моделирования случайных событий связана с построением модели полной группы случайных событий ***A, B, C***. События происходят с вероятностями P_a, P_b, P_c. Причем всегда реализуется только одно событие: $P_a + P_b + P_c = 1$ (рис. 3.9.1). Например, полная группа из шести несовместных случайных событий реализуется при бросании игрального кубика.

В столбец **B** вводится функция ***СЛЧИС()*** – генератор случайных чисел с равномерным распределением в диапазоне (0,1).

	A	B	C	D	E	F	G	H	I	J
1	**Pa**	**Xi**	**A**	**B**	**C**		**Кол-во событий A**	**Кол-во событий B**	**Кол-во событий C**	
2	0,6	0,3438	1	0	0		**60**	**28**	**12**	
3	**Pb**	0,4234	1	0	0		**Частота A**	**Частота B**	**Частота C**	**Сумма частот**
4	0,3	0,6923	0	1	0		**0,6**	**0,28**	**0,12**	**1**
5	**Pc**	0,0596	1	0	0					
6	0,1	0,2776	1	0	0					
7		0,9607	0	0	1					

Рис. 3.9.1. Моделирование полной группы случайных событий

Событие ***A*** реализуется, если выполняется условие $x_i \le P_a$, событие ***B*** реализуется, если справедливо соотношение $P_a < x_i \le P_a + P_b$.. В свою очередь, событие ***C*** реализуется, если справедливо соотношение $P_a + P_b < x_i$ (x_i < 1 по определению). Столбцы таблицы **C, D, E** (начиная с ячеек **C2, D2, E2**) заполняются по аналогии с предыдущей задачей с использованием функции «**ЕСЛИ»:**

=ЕСЛИ(B2<=$A$2;1;0) – произошло событие ***A (A = 1)***,

=ЕСЛИ(И(B2>$A$2;B2<=$A$2+$A$4);1;0) – произошло событие ***B (B = 1)***,

=ЕСЛИ(B2>$A$2+$A$4;1;0) – произошло событие ***C (C = 1)***.

Количество событий ***A, B, C*** подсчитывается с помощью функции **СУММ().** Частоту событий ***A, B, C*** можно посчитать с помощью функции **СРЗНАЧ().** Сумма частот должна быть равна единице (рис. 3.9.1).

3.10. Моделирование независимых случайных событий

Используя предыдущие задачи, можно построить модель реализации двух независимых событий ***A*** и ***B***, которые происходят с вероятностями P_a и P_b соответственно, причем события могут происходить совместно. Эта задача может быть решена, например, аналогично задаче моделирования полной группы случайных событий, так как в этом случае образуется полная группа из четырех несовместных событий:

Событие 1. (AuB), Событие 2. (НЕА и В), Событие 3. (А и НЕВ), Событие 4. (НЕА и НЕВ),

теоретические вероятности, которых соответственно равны:

$$P_1 = (P_a \cdot P_b,) \quad P_2 = (1 - P_a) \cdot P_b, \quad P_3 = (1 - P_b) \cdot P_a, \quad P_4 = (1 - P_a) \cdot (1 - P_b).$$

Задача решается с помощью двух датчиков случайных чисел отдельно для события ***A*** и для события ***B***. Тогда в результате моделирования можно проверить справедливость последних формул для вероятностей (рис. 3.10.1).

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Pa	Xi	A	Yi	B		A и B	НЕ A и B	A и НЕ B	НЕ A и НЕ B		Вероятности событий				
2	0,6	0,921	0	0,968	0		0	0	0	1		P1=Pa·Pb	P2=(1-Pa)·Pb	P3=Pa·(1-Pb)	P4=(1-Pa)·(1-Pb)	Сумма
3	Pb	0,926	0	0,786	0		0	0	0	1		0,180	0,120	0,420	0,280	1,000
4	0,3	0,468	1	0,329	0		0	0	1	0						
5		0,592	1	0,486	0		0	0	1	0		Частоты событий				
6		0,418	1	0,608	0		0	0	1	0		A и B	НЕ A и B	A и НЕ B	НЕ A и НЕ B	Сумма
7		0,309	1	0,109	1		1	0	0	0		0,187	0,093	0,413	0,307	1,000
8		0,973	0	0,811	0		0	0	0	1						

Рис. 3.10.1. Модель для двух независимых событий

Если событие ***(A и B)*** произошло, то в колонке **G** значение **1**, иначе **0**. Это определено с помощью функции **«ЕСЛИ»**: **=ЕСЛИ(И(C2=1;E2=1); 1;0).** Другие события определяются аналогично. Частоты событий вычисляются с помощью функции **СРЗНАЧ()** по соответствующему столбцу.

Сумма частот и вероятностей (рис. 3.10.1), естественно, равна единице, так как имеет полную группу из четырех несовместных событий: $P_1 + P_2 + P_3 + P_4 = 1$.

Провести ряд экспериментов при разном количестве реализаций событий: 20, 100; 500, 1 500.

САМОСТОЯТЕЛЬНАЯ РАБОТА

3.11. Имитационная модель автомобиля

Рассмотрим содержательную задачу моделирования случайных событий. Пусть необходимо выполнить прогноз работы автомобиля. В течение суток автомобиль может быть исправным или стать неисправным (рис. 3.11.1). Состояние автомобиля изменяется дискретно с шагом изменения «сутки».

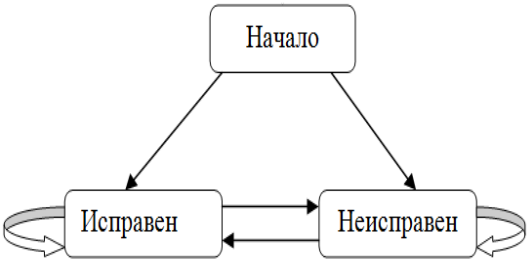


Рис. 1. Граф изменения состояния автомобиля

Смена состояния автомобиля является случайным событием, так как зависит от многих факторов, которые заранее учесть невозможно. Вероятности смены состояний автомобиля устанавливаются в реальной практике на основе статистических данных по эксплуатации и ремонту, а в данной задаче задаются как исходные величины.

В начальный момент времени автомобиль может быть исправным с вероятностью P_0. Естественно, что новый автомобиль в начальный момент с достаточно большой вероятностью исправен. Дальнейшее развитие событий может быть следующим: автомобиль в течение суток с вероятностью P_1 может сохранить свое исправное состояние. Если он станет неисправным, то на следующие сутки приступят к его ремонту. Неисправный автомобиль после суток ремонта с вероятностью P_2 вернется к работе, т.е. станет исправным, или его ремонт продолжится.

Изменение состояния автомобиля представлено графом (рис. 3.11.1). Будем считать, что вероятности переходов зависят только от конкретного состояния автомобиля (исправен – в ремонте), а не от того, каким путем данное состояние достигнуто. Далее автомобиль находится в каждом возможном состоянии ровно сутки и переходит в новое состояние или сохраняет текущее состояние с учетом вероятности перехода. Модель изменения состояния автомобиля представлена электронной таблицей на рис. 3.11.2.

Здесь X_i, Y_i – случайные числа, полученные с применением функции **СЛЧИС()**. Случайное число X_i используется при моделировании события «изменение состояния исправного автомобиля» (событие $A = 1$ – автомо-

биль исправен; событие *A = 0* – автомобиль неисправен). Случайное число Y_i используется при моделировании результата ремонта автомобиля (событие *C = 1* – автомобиль отремонтирован; событие *C = 0* – автомобиль остается в ремонте или переходит в исправное состояние).

	A	B	C	D	E	F
1	P0	P1	P2			
2	0,9	0,8	0,7			
3						
4	День	Xi	A-машина работает	Yi	B-машина в ремонте	C-машина отремонтирована
5	1	0,035	*1*		*0*	*0*
6	2	0,854	*0*		1	*0*
7	3	0,776	1	0,119	0	1
8	4	0,532	1	0,575	0	0
9	5	0,597	1	0,398	0	0
10	6	0,590	1	0,825	0	0
11	7	0,968	0	0,014	1	0
12	8	0,460	1	0,631	0	1
13	9	0,343	1	0,711	0	0
14	10	0,880	0	0,188	1	0

Рис. 3.11.2. Модель изменения состояния автомобиля

Начальное состояние автомобиля (в первый день, ячейка **C5**, рис.2) определяется формулой: **=ЕСЛИ(B5<=$A$2;1;0) (**определение состояния автомобиля в первый день как случайное событие). В ячейке **E5** вводится число 0, так как по условиям задачи он поступит в ремонт на следующий день, если в первый день окажется неисправным. Соответственно, в ячейку **F5** также вводится число 0, так как ремонт по условиям задачи длится минимум один день. В ячейку **F6** также вводится 0, так как в первый день в ремонте не может быть автомобиля. Случайные числа в ячейках **D5, D6** не потребуются.

Состояние исправного в первый день автомобиля (**C5 = 1**) на следующий день может измениться вследствие поломки – случайное событие. Поэтому в ячейку **C6** вводится формула: **=ЕСЛИ(И(B6<=$B$2;C5=1);1;0)**.

Если в текущий день автомобиль окажется неисправным, то на следующий день он поступит в ремонт: **=ЕСЛИ(C6+F6=0;1;0)** (ячейка **E6**, рис. 3.11.2). Действительно, в один и тот же день автомобиль может быть либо исправен, либо находиться в ремонте. Если автомобиль в этот день отремонтирован, то он на следующий день считается исправным. Например, ячейка: **E6 = 1** (день = 2, автомобиль поступил в ремонт) → **F7 = 1**

(день = 3, автомобиль отремонтирован) → **C7 = 1** (и в этот же день автомобиль приступил к работе, рис. 3 11.2).

Результат ремонта определяется формулой **=ЕСЛИ(И(E6=1;D7<=$C$2);1;0)** (ячейка **F7**, рис. 3.11.2), в ремонте находится неисправный автомобиль (**E6 = 1**). Результат ремонта – случайное событие. Состояние автомобиля на третий день (ячейка **C7**, рис. 2) и последующие дни определяется тем, что автомобиль либо сохраняет свое исправное состояние, либо поступает из ремонта: **=ЕСЛИ(ИЛИ(И(C6=1;B7<=$B$2);F7=1);1;0).** В остальные ячейки столбцов **C, D, E, F** формулы копируются. Результат моделирования изменения состояния одного автомобиля представлен на рис. 3.11.3.

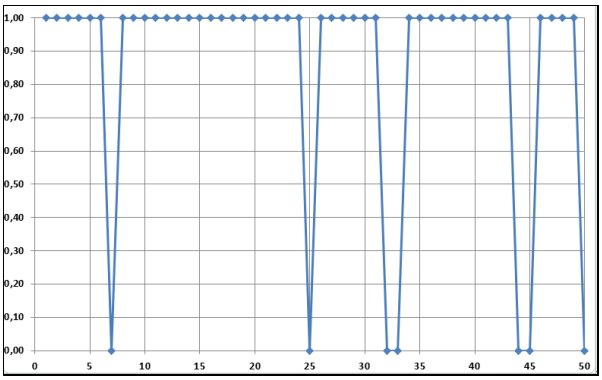


Рис. 3.11.3. Результат моделирования состояния одного автомобиля

Если автопарк состоит из нескольких автомобилей, то аналогичным образом строятся модели изменения состояния каждого автомобиля. Выполнить это можно путем копирования таблицы по рис. 3.11.2 и подсчета общего числа исправных автомобилей на каждый день (рис. 3.11.4).

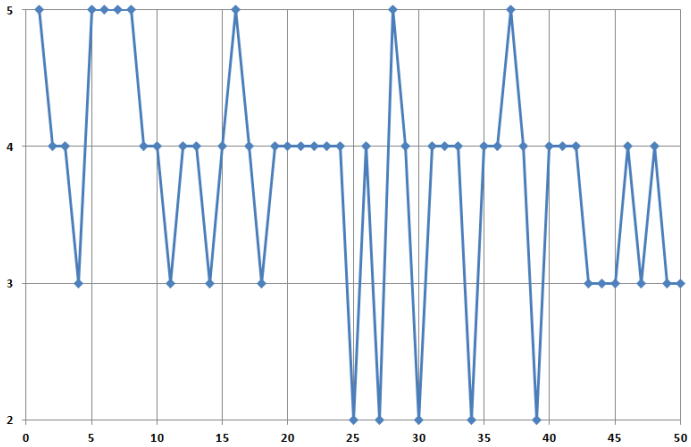


Рис. 3.11.4. Диаграмма изменения состояния автопарка из пяти автомобилей

Теперь можно решать типичную задачу **имитационного моделирования**. Например, какое количество автомобилей необходимо иметь, с учетом их возможной поломки, чтобы выполнить за заданное время определенный объем работ, который определен в машино-днях, какова вероятность выполнения этой работы имеющимся парком автомобилей за требуемое время.

В этом случае требуется многократное повторение имитационных экспериментов для получения статистических данных о состоянии парка автомобилей. Это является особенностью стохастического моделирования. Естественно, что в рассмотренных моделях приняты достаточно грубые предположения о вероятностях поломки автомобиля и окончания его ремонта. Однако уточнение модели требует привлечения теории надежности, что в школьном курсе информатики неприемлемо.

Задач имитационного моделирования в учебной литературе для школьников чрезвычайно мало. В настоящей работе предлагается построение задачи имитационного моделирования в среде электронных таблиц.

Допустим, что фирме, имеющей пять автомобилей, требуется выполнить за 100 дней определенную работу. Объем этой работы в машино-днях известен. Ясно, что 100 дней все автомобили работать не смогут, так как возможны их поломки. Для выяснения возможности выполнения работы был проведен имитационный эксперимент (модель для пяти автомобилей), который состоял из 20 испытаний. Его результаты представлены на рис. 3.11.5.

Таким образом, если работа требует 370 машино-дней или менее, то она будет выполнена. Объем работ в 380 машино-дней будет выполнен с вероятностью 0,9. В этом случае, скорее всего, дополнительной техники не потребуется. Если для выполнения работы требуется 390 машино-дней, то вероятность выполнения работы в срок оценивается как 0,5. Можно сделать вывод, что для выполнения такого и большего объема работ необходимы дополнительные автомобили. Ясно, что имитационный эксперимент позволит оценить вероятность выполнения работы имеющимся количеством ав-

томобилей. Кроме того, результаты экспериментов позволят исключить привлечение излишнего количества автомобилей для выполнения работы.

P0	P1	P2
0,9	0,8	0,7
Количество автомобилей	Срок выполнения работы (дней)	
5	100	
Объем работы (машино-дни)	Число положительных результатов	Частота реализации события: "Работа выполнена"
370	20	1,00
380	18	0,90
390	10	0,50
400	4	0,25
410	0	0,00

Рис. 3.11.5. Результат имитационного эксперимента

Можно, например, рассчитать, на сколько дней необходимо арендовать дополнительные автомобили, чтобы выполнить требуемую работу. Собственно в этом и состоит цель имитационного моделирования – определить оптимальные значения параметров системы. Проведите компьютерный имитационный эксперимент и постройте таблицу наподобие таблицы по рис. 3.11.5, если компания имеет в распоряжении только 4 машины.

3.12. Моделирование процесса переноса

Работа посвящена моделированию переходного процесса в системе с распределенными параметрами. Пусть имеется теплообменник, представляющий собой канал, обогрев которого производится, например, пропусканием электрического тока через его стенки.

Примем следующие предположения: в теплообменник поступает холодная жидкость с известной температурой T_0; скорость движения жидкости (теплоносителя) внутри канала считается неизменной; свойства теплоносителя (плотность, теплоемкость и т.п.) считаются неизменными. Температура теплоносителя однородна по сечению канала.

В начальный момент времени $t = 0$ включается обогрев канала, т.е. *q = 0*, при *t < 0* и *q > 0*, при *t > 0* . Здесь *q* – количество тепла, которое передается через стенки канала к теплоносителю в единицу времени через единицу его поверхности.

Расчетная схема объекта моделирования представлена на рис. 3.12.1.

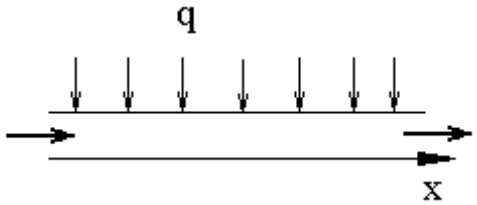


Рис. 3.12.1. Схема объекта моделирования

До начала обогрева теплообменник находился в стационарном режиме. После включения обогрева теплообменник через некоторое время выйдет на новый стационарный режим.

С учетом принятых допущений, динамика изменения температуры теплоносителя в теплообменнике может быть описана следующим уравнением, которое является следствием закона сохранения энергии:

$$S(\rho c\frac{\partial T}{\partial t} + c\rho u\frac{\partial T}{\partial x}) = q\Pi; \quad S = \pi R^2, \ \Pi = 2\pi R.$$

Здесь *S* – площадь поперечного сечения канала теплообменника, П – его периметр, ρ – плотность теплоносителя, *c* – его теплоемкость, *u* – скорость движения теплоносителя по каналу теплообменника. После преобразований безразмерное уравнение процесса переноса тепла примет вид:

$$\frac{\partial y}{\partial \bar{t}} + \frac{\partial y}{\partial \bar{x}} = 1.$$

Краевые и начальные условия, с учетом преобразований, задаются следующими соотношениями:

$$y(\bar{t}, \bar{x} = 0) = 0 \qquad y(\bar{t} = 0, \bar{x}) = 0.$$

Для численного решения данной начально-краевой задачи применим метод конечных разностей. Дифференциальные операторы $\frac{\partial}{\partial t}$ и $\frac{\partial}{\partial x}$ заменим приближенными конечно-разностными аналогами. В итоге получим:

$$\frac{y_j^{n+1}-y_j^n}{\tau}+\frac{y_j^{n+1}-y_{j-1}^{n+1}}{h}=1.$$

Здесь $y_j^n = y(t_n, x_j)$, $y_j^{n+1} = y(t_{n+1}, x_j)$, $y_{j-1}^{n+1} = y(t_{n+1}, x_{j-1})$. Пусть в момент времени $t = t_n$ нам известны значения y_j^n в любой точке канала по оси *x*. Преобразуем конечно-разностный аналог уравнения следующим образом:

$$y_j^{n+1}\left(\frac{1}{\tau}+\frac{1}{h}\right)=1+\frac{y_j^n}{\tau}+\frac{y_{j-1}^{n+1}}{h}, \qquad y_j^{n+1}=\frac{1}{\left(\frac{1}{\tau}+\frac{1}{h}\right)}+\frac{y_j^n\left(\frac{1}{\tau}\right)}{\left(\frac{1}{\tau}+\frac{1}{h}\right)}+y_{j-1}^{n+1}\frac{\frac{1}{h}}{\left(\frac{1}{\tau}+\frac{1}{h}\right)}.$$

Последнее соотношение есть расчетная формула, применяя которую последовательно для *j = 2 – M*, (*M* – номер последней точки по координате *x*) на основе известных значений y_j^n получим искомые величины y_j^{n+1}. Для *j = 1* , т.е. при *x = 0,* значение y_j^{n+1} всегда известно, так как задано краевыми условиями. Рассмотренный метод решения уравнения в частных производных называется методом «бегущего» счета, метод безусловно устойчив и имеет первый порядок точности.

Постановка задачи моделирования. Установить закономерность перехода теплообменника в новое состояние после включения обогрева, установить закон изменения температуры теплоносителя в канале теплообменника как функции времени и координаты – ***T (t,x)***.

Порядок выполнения работы. Для решения задачи определения нового температурного режима создадим таблицу (рис. 3.12.4). Все необходимые расчетные формулы представлены ниже.

Значения **A, B, C** рассчитываются по формулам (см. ниже). Значение y при *x = 0* задается явно исходя из краевых условий. Значения координат узловых точек по *x* рассчитываются в ЭТ по формуле (можно использовать автозаполнение). Начальное распределение y_0 задается явно как исходные данные**.** Другие значения $y_1 - y_{50}$ вычисляются по методу бегущего счета (формулы см. ниже). В результате должен быть получен график изменения *y(x)* по времени (см. рис. 3.12.3) для различных моментов времени (каждое пятое значение y).

	A	B	C	D	E	F	G	H	I
1	T	Xi	Y0	Y1	Y2	Y3	Y4	Y5	Y6
2	0,04	0	0	0	0	0	0	0	0
3	h	0,02	0	0,0133	0,0178	0,0193	0,0198	0,0199	0,0200
4	0,02	0,04	0	0,0222	0,0326	0,0370	0,0388	0,0396	0,0398
5	A	0,06	0	0,0281	0,0444	0,0528	0,0568	0,0587	0,0594
6	0,01333	0,08	0	0,0321	0,0537	0,0664	0,0734	0,0769	0,0786
7	B	0,10	0	0,0347	0,0607	0,0779	0,0882	0,0940	0,0971
8	0,33333	0,12	0	0,0365	0,0660	0,0872	0,1012	0,1097	0,1146
9	C	0,14	0	0,0377	0,0699	0,0948	0,1124	0,1240	0,1311
10	0,66667	0,16	0	0,0384	0,0727	0,1008	0,1218	0,1366	0,1462
11		0,18	0	0,0390	0,0748	0,1054	0,1297	0,1476	0,1600
12		0,20	0	0,0393	0,0763	0,1091	0,1362	0,1571	0,1724

Рис. 3.12.2. Расчетная таблица

$$y_j^{n+1} = \frac{1}{\left(\frac{1}{\tau}+\frac{1}{h}\right)} + \frac{y_j^n\left(\frac{1}{\tau}\right)}{\left(\frac{1}{\tau}+\frac{1}{h}\right)} + y_{j-1}^{n+1}\frac{\frac{1}{h}}{\left(\frac{1}{\tau}+\frac{1}{h}\right)}.$$

$$A = \frac{1}{\left(\frac{1}{\tau}+\frac{1}{h}\right)}, \quad B = \frac{\left(\frac{1}{\tau}\right)}{\left(\frac{1}{\tau}+\frac{1}{h}\right)}, \quad C = \frac{\frac{1}{h}}{\left(\frac{1}{\tau}+\frac{1}{h}\right)}.$$

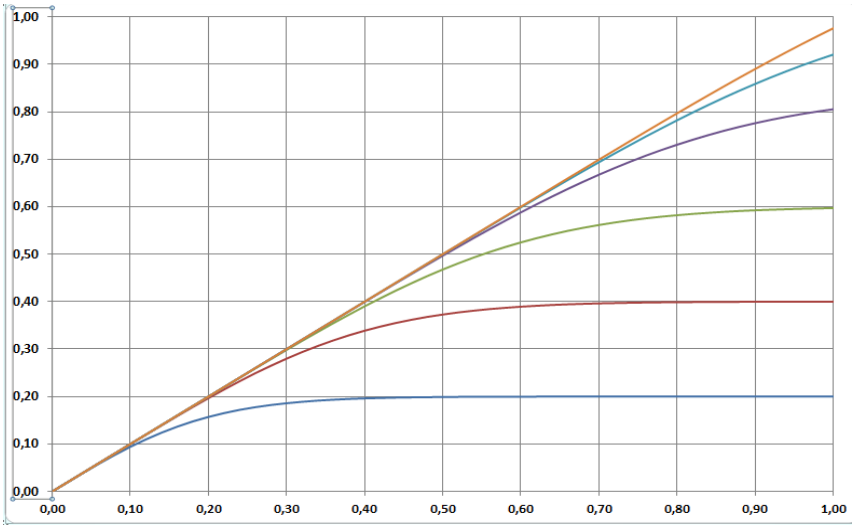


Рис. 3.12.3. Результат моделирования

Дополнительное задание. Самостоятельно построить компьютерную модель и провести расчеты для следующей задачи, которая в безразмерном виде представляется уравнением:

$$\frac{\partial y}{\partial \bar{t}} + \frac{\partial y}{\partial \bar{x}} = 1 - ky,$$

с начальными и краевыми условиями:

$$y(\bar{t}, \bar{x} = 0) = 0 \qquad y(\bar{t} = 0, \bar{x}) = 0.$$

Получить численное решение задачи, используя метод конечных разностей по аналогии с предыдущей задачей. Установить влияние на результат моделирования значения параметра ***k.***

3.13. Контрольные вопросы по главе 3

1. Какие надстройки Excel используются при поиске оптимальных решений?
2. Какая функция Excel используется при моделировании случайных событий?
3. Какую информацию может дать имитационная модель гаража?
4. Что такое «Целевая ячейка».
5. Какие преимущества при создании моделей имеет Excel?
6. Можно ли в Excel построить модель распределенной системы?
7. Что такое случайная величина и случайное событие?
8. Что такое полная группа случайных событий?
9. Какие случайные числа генерирует функция СЛЧИС?
10. Чему равна вероятность события НЕА, если вероятность А равна Р?

ГЛАВА 4. МОДЕЛИРОВАНИЕ СИСТЕМ УПРАВЛЕНИЯ

4.1. Системы управления

Потребность в автоматическом регулировании и управлении возникла и стала весьма актуальной с ростом сложности техники и расширением применения технических систем в производстве. Возникла необходимость управления процессами, которые подвержены действию помех и внешних воздействий, не только нарушающих точность функционирования, но и работоспособность системы в целом. Стало очевидно, что человек (оператор) не в состоянии обеспечить требуемую точность и оперативность регулирования режима работы технической системы. Например, необходимость регулирования актуальна для таких точных механических систем, как часы. Для них требуется обеспечить точность хода в условиях постоянного действия небольших помех.

Развитие устройств регулирования и управления началось в эпоху промышленной революции. Одним из первых промышленных регуляторов стал поплавковый регулятор питания котла паровой машины И.И. Ползунова и центробежный регулятор скорости вращения вала паровой машины Дж. Уатта. Паровая машина стала первым объектом применения регулирования, так как она не обладает свойством устойчивой работы. Известно множество случаев, когда оператор (человек) паровой машины был не способен выполнять функции регулирования, а наоборот, «раскачивал» колебания ее параметров. Именно потребности регулирования паровой машины стимулировали развитие теоретических исследований в данной области.

В начале XX в. теория автоматического регулирования сформировалась как самостоятельная научная дисциплина с рядом прикладных разделов, которые связаны с конкретными техническими системами. Разрабатываются новые математические методы анализа линейных и нелинейных систем регулирования, методы исследования устойчивости и чувствительности, новые принципы управления: регулирование по возмущению, теория компенсации возмущения и инвариантности, принципы экстремального управления, теория оптимального управления и т.д.

Существенный вклад в развитие теории автоматического управления и регулирования внесли российские ученые Н.М. Крылов, Н.Н. Боголюбов, А.В. Михайлов, А.А. Ляпунов, А.А. Андронов, Л.С. Понтрягин, А.А. Красовский и другие.

Значение теории автоматического регулирования и управления актуально не только для технических систем. Управляемые процессы имеют место в самых различных системах, например: в экономических, технологических, социальных, человеко-машинных системах и т.д. В настоящее время теория автоматического управления является научно-технической дисциплиной, которая изучает общие принципы управления разнообразными техническими системами на основе построения их математических и компьютерных моделей, разрабатывает собственные методы анализа и синтеза систем управления, методы анализа устойчивости, методы оценки свойств проектируемых систем и качества регулирования.

Итак, **управление** – это специальное воздействие на объект управления с целью обеспечить его оптимальное состояние или поведение. **Регулирование** – это частный случай управления: обеспечение определенного состояния объекта регулирования. Автоматическое управление и регулирование осуществляются без участия оператора (человека). При возникновении отклонений от требуемого режима функционирования объекта система регулирования (управления) должна воздействовать на объект так, чтобы устранить отклонения.

Различают внешнее и внутреннее управление в системе. Внутренняя функция системы – самоуправление. Внешнее управление обеспечивает необходимое функционирование системы.

Следующим шагом было развитие кибернетики, основы которой Норберт Винер (1894–1964 гг.) изложил в книге «Кибернетика, или управление и связь в животном и машине», 1948 г. Во время Второй мировой войны, он работал над математическим аппаратом для систем наведения зенитного огня и управления американскими силами противовоздушной обороны.

Существуют следующие виды систем управления:

- **Системы стабилизации**. На начальном этапе развития техники системы стабилизации представляли единственный вид систем управления. Задачей данной системы является поддержание постоянства управляемого параметра.

- **Системы программного управления**. В этом случае алгоритм функционирования системы задан. В подобных системах существует датчик, который вырабатывает задающие воздействия по определенному алгоритму. Программное управление может быть организовано по любой схеме с обратной связью или без нее. Например, системы подобного рода применяются при программном управлении металлорежущими станками, где заложен алгоритм функционирования системы, который изменяет управляемую величину в соответствии с заранее заданной функцией времени.

- **Следящие системы**. Алгоритм функционирования в таких системах заранее неизвестен. Регулируемый параметр в таких системах должен воспроизводить изменение некоторого внешнего фактора, например, движения цели. Следящая система управления может быть организована также по любой схеме с обратной связью или без нее, если в ней будет присутствовать датчик слежения за изменением внешнего фактора.

- **Системы оптимального управления.** Определяют и поддерживают управляющие воздействия, соответствующие экстремальному значению управляемой величины.

- **Адаптивные системы управления.** Способны автоматически приспосабливаться к изменению внешних условий и свойств объекта управления.

Управляющие сигналы к объекту управления могут поступать непрерывно. Такие системы управления называются системами непрерывного действия. Примером является система управления паровой машиной на основе регулятора Уатта. Примером систем управления дискретного действия является система термостатирования, в которой обогрев или охлаждение включаются и выключаются в момент достижения определенных температур в термостате.

Виды управления: управление без обратной связи, управление с обратной связью, управление по возмущению.

Математическое моделирование систем управления – это классический подход к анализу подобных систем, который составляет основу теории автоматического управления и регулирования. Сегодня к чисто математическому моделированию добавилось компьютерное моделирование. В рамках теории автоматического регулирования были разработаны специфические методы анализа, которые затем использовались в других науках. Примером может служить теория устойчивости.

Методы математического моделирования, несомненно, актуальны в теории управления. Экспериментальный метод в данной области уместен лишь на этапе испытаний опытных образцов систем управления.

Первоначально в теории автоматического управления развивались аналитические и приближенные методы. В настоящее время разработаны специализированные программные комплексы компьютерного моделирования, предназначенные для моделирования систем управления и решения специальных задач этой области. Например, инструментальные системы компьютерного моделирования Simulink, VisSim и российский рус-

скоязычный аналог программный комплекс МВТУ (Моделирование В Технических Устройствах), разработанный в техническом университете (МВТУ имени Баумана, Москва).

Принцип моделирования программного комплекса МВТУ состоит в создании и исследовании виртуальной модели реальной системы, которая представляет собой блок-схему. При моделировании не обязательно записывать уравнения модели в явном виде, модель конструируется из готовых блоков путем их копирования и соединения в блок-схему. Принцип построения моделей в среде МВТУ аналогичен технологиям построения моделей в программных комплексах VisSim и Simulink.

Программный комплекс VisSim предназначен для построения и исследования виртуальных моделей технических объектов, в том числе и систем управления. Этот комплекс разработан и развивается компанией «Visual Solutions». VisSim – это аббревиатура выражения Visual Simulator – визуальная среда моделирования. Он предоставляет пользователю графический интерфейс, с помощью которого исследователь создает модель из виртуальных элементов, что позволяет исследовать модели достаточно сложных систем. По технологии моделирования VisSim подобен другим аналогичным программным комплексам (рис. 4.4.1).

Особенностью этого пакета моделирования является ориентация на задачи анализа систем автоматического управления. При описании и последующем построении модели в среде VisSim численные методы реализации модели выбираются автоматически. Результаты решения могут быть представлены в виде временных и фазовых диаграмм. При построении моделей в среде VisSim программирование не требуется, однако имеется возможность создания дополнительных моделей элементов и помещение этих моделей в библиотеку стандартных блоков VisSim.

Математическое и компьютерное моделирование систем управления основано на их декомпозиции, т.е. разделении на элементы (звенья).

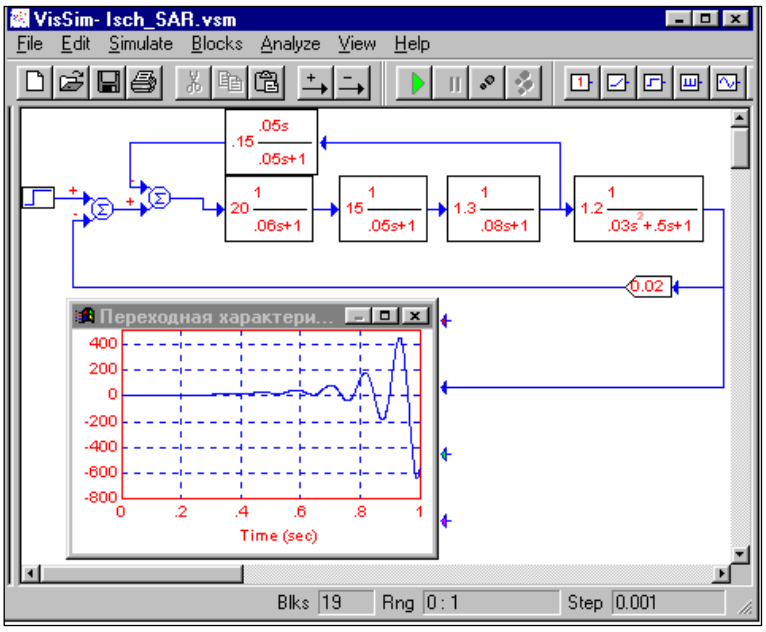


Рис. 4.4.1. Виртуальный стенд моделирования пакета VisSim

Математические модели элементов строятся на основе закономерностей процессов, которые в них протекают. Накопленный опыт исследования и разработки систем управления позволили выделить типовые математические модели элементов систем управления, из которых и создаются системы. Более сложные элементы могут быть получены путем соединения простейших типовых звеньев. Анализ свойств систем строится на результатах исследования свойств типовых элементов и свойств соединений элементов в более сложные агрегаты.

4.2. Моделирование в среде пакета VisSim

VisSim – визуальная система моделирования, предназначенная для моделирования динамических систем, систем автоматического управления и регулирования. VisSim сочетает в себе интуитивный интерфейс для создания блочных моделей и мощное моделирующее ядро. Пакет разработан американской компанией Visual Solutions, которая находится в городе Уэстфорд (США).

Пакет VisSim широко используется в разработке систем управления и цифровой обработки сигналов для моделирования и дизайна. Она включает в себя блоки для арифметики, булевых и трансцендентных функций, а также цифровые фильтры, передаточные функции, численного интегрирования и интерактивного вывода. Основными областями моделирования являются аэрокосмическая, биологическая/медицинская, электродвигатели, электрические, гидравлические, механические, тепловые процессы, эконометрика.

Построение модели VisSim – это способ визуального представления некоторой ситуации. При построении модели, вместо вывода и решения системы уравнений для решения проблемы, используются графические блоки. Сила этого метода особенно хорошо проявляется на задачах, обычно содержащих математические уравнения, аналитическое решение которых представляется сложным или проблематичным. Однако, если может быть построена модель, которая иллюстрирует данную ситуацию, то становится понятно, где искать решение, и иногда оно становится просто очевидным.

Исходными данными для построения модели в VisSim являются структурно-функциональная схема моделируемой системы, процесса или объекта и описывающие их дифференциально-алгебраические уравнения. Вместо таких уравнений могут быть заданы операторы или функции, характеризующие отдельные элементы моделируемой системы, например, передаточные функции для линейных элементов и статические характеристики для нелинейных элементов.

Реальные системы и объекты состоят из отдельных, связанных и взаимодействующих друг с другом элементов. И для всей системы в целом, и для отдельных ее, должным образом выбранных элементов, можно указать место приложения воздействия, которое можно назвать входом, и место их реакции (отклика) на входное воздействие, называемое выходом. И воздействие, и реакция – это некоторые физические величины, являющиеся функциями времени.

Модели систем и объектов в программе VisSim строятся из отдельных элементов – блоков. Блок – это виртуальный аналог физического элемента реальной системы. Термин «аналог» предполагает, что блок функционирует и подчиняется тем же самым уравнениям, что и реальный, моделируемый элемент системы.

Виртуальные блоки VisSim могут иметь или вход, на который может быть подан выходной сигнал другого блока, или выход, виртуальный сигнал с которого может быть подан на вход другого блока. Наконец, блоки могут иметь и вход, и выход одновременно. Взаимодействие между блоками отображается линиями связи, указывающими направление передачи воздействий (сигналов) от одного блока к другому.

Взаимодействие между блоками моделируется сигналами – функциями времени, передаваемыми между блоками по линиям связи. Сигналы в модели могут быть измерены с помощью виртуальных измерительных устройств или рассмотрены и изучены с помощью виртуального осциллографа.

Внешне виртуальные блоки VisSim с некоторой степенью условности воспринимаются исследователем так же, как реальные устройства. Например, генераторы вырабатывают сигналы, блоки-преобразователи реагируют на входные сигналы в определенном смысле точно так же, как реальные устройства на реальные воздействия, индикаторы показывают величины сигналов.

Таким образом, принцип построения модели в VisSim состоит в вынесении на рабочее пространство моделей реальных элементов (блоков) и соединении их в соответствии с заранее составленной структурно-алгоритмической схемой моделируемой системы. Такое построение модели из виртуальных блоков очень похоже, с известной степенью условности, на построение реальной системы из настоящих блоков (генераторов, осциллографов и других устройств) в производственных условиях или на лабораторном стенде.

Блоки VisSim можно условно разделить на три основных категории и одну дополнительную:

– Блоки, имеющие только выход: генераторы.

– Блоки, имеющие вход и выход: преобразователи.

– Блоки, имеющие только вход: индикаторы.

– Блоки без входов и выходов: надписи, комментарии и др.

Важным компонентом модели является **соединительная линия** – виртуальный аналог физического соединения элементов, передающего сигналы от одного элемента к другому. Соединительные линии в VisSim однонаправленные, передают сигналы с выхода одного блока к входу другого.

Система динамического моделирования VisSim предназначена для исследования и анализа переходных и установившихся процессов в любых динамических системах, в том числе и в автоматических системах с использованием визуальных средств структурного моделирования. VisSim представляет собой симулятор систем, инструментальную среду визуального проектирования.

VisSim имеет решатель интерпретирующего типа, функционирующий в динамическом режиме с возможностью online-взаимодействия с оборудованием реального времени.

Пользователь может создать собственную библиотеку моделей. Программа VisSim обладает развитым графическим интерфейсом, используя который можно создать модель из виртуальных элементов. Это позволяет исследователю разрабатывать, исследовать и оптимизировать модели систем различной сложности и назначения. VisSim автоматически создаёт и решает дифференциальные уравнения по предложенной структуре системы и параметрам ее элементов. Результаты решения выводятся в наглядной графической форме. Поэтому программой могут пользоваться и те, кто не имеет глубоких познаний в математике и программировании. Кроме моделирования систем управления в VisSim можно решать дифференци-

альные уравнения причем, эффективнее и быстрее, чем известный математический пакет MathCAD.

При соизмеримой и более высокой производительности, чем у программы Simulink, входящей в этот пакет, VisSim требует существенно меньше места на жестком диске и в оперативной памяти. VisSim позволяет также решать задачи по физике от школьного уровня и до серьезных физических экспериментов на виртуальных лабораторных стендах.

4.3. Построение VisSim-моделей

Целью данной работы является знакомство с возможностями инструментальной системы моделирования VisSim, приемами построения моделей и исследование переходных процессов.

Если экспериментально получить график переходного процесса (разгонную характеристику) некоторого объекта, то по этому графику можно определить тип и параметры звена, приближенно соответствующего объекту, т.е. построить его модель. Этот процесс называется идентификацией объекта.

В теории автоматического управления при моделировании линейных систем применяют так называемые типовые звенья, которые приближенно соответствуют элементам реальных систем и просто описываются математически.

Типовое звено – это математическая модель динамического элемента системы автоматического регулирования в целом, обладающая определенным набором физических свойств, например, способностью к накоплению воздействия или к усилению воздействия и инерционностью.

Типовые звенья: пропорциональное; интегрирующее и дифференцирующее; инерционное; колебательное; звено запаздывания.

Типовые звенья линейных систем можно определять различными эквивалентными способами, в частности с помощью так называемой пере-

даточной функции, имеющей, как правило, дробно-рациональный вид, т.е. представляющей собой отношение двух многочленов (рис. 4.3.1):

$$\Phi(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \ldots + b_1 s^1 + b_0}{a_n s^n + a_{n-1} s^{n-1} + \ldots + a_1 s^1 + a_0},$$

где b_i и a_j – параметры передаточной функции или звена.

Передаточная функция связывает изображение *Y(s)* выходного сигнала *y(t)* звена с изображением *X(s)* его входного сигнала *x(t):* $Y(s) = \Phi(s) \cdot X(s),$

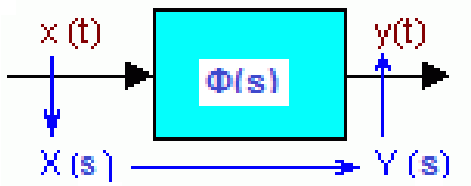


Рис. 4.3.1. Передаточная функция

Передаточная функция позволяет по любому известному входному сигналу *x(t)* найти выходной *y(t*). С точки зрения теории автоматического управления передаточная функция полностью характеризует звено.

Пропорциональное звено – это звено, выходной сигнал которого пропорционален входному: *y(t) = k x(t).*

Передаточная функция пропорционального звена равна: *Φ(s) = k,* здесь *k* – коэффициент усиления.

Интегратор – это звено, сигнал которого на выходе пропорционален интегралу по времени от входного сигнала:

$$\frac{dy(t)}{dt} = k \cdot x(t); \qquad y(t) = k \int_0^t x(t)dt.$$

Передаточная функция интегратора равна:

$$\Phi(s) = \frac{k}{s}.$$

Инерционное звено. Звено, выходной сигнал *y(t)* которого связан с входным *x(t)* дифференциальным уравнением:

$$T \cdot \frac{dy}{dt} + y = k \cdot x \cdot$$

Передаточная функция апериодического звена равна:

$$\Phi(s) = \frac{k}{sT + 1}.$$

Здесь параметры: *k* – коэффициент усиления и *T* – постоянная времени.

Колебательное звено имеет выходной сигнал *y(t)*, который связан со входным сигналом *x(t)* дифференциальным уравнением:

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

Его передаточная функция имеет вид:

$$\Phi(s) = \frac{1}{ms^2 + ks + c}$$

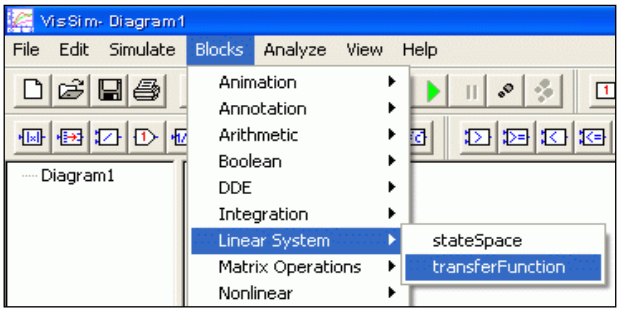


Рис. 4.3.2. Пункт меню VisSim для помещения на рабочее поле блока transferFunction

В программе VisSim апериодическое, колебательное и более сложные типовые звенья выносятся на рабочее поле с помощью блока **transferFunction (Blocks – Linear System – transferFunction**, рис. 4.3.2; 4.3.3**)**.

Блок **transferFunction** на рабочем поле VisSim формально обозначается передаточной функцией. Однако работает он, непосредственно решая соответствующее дифференциальное уравнение. Формула передаточной

функции в блоке на рабочем поле отображается вместо дифференциального уравнения просто из соображений экономии места на рабочем поле.

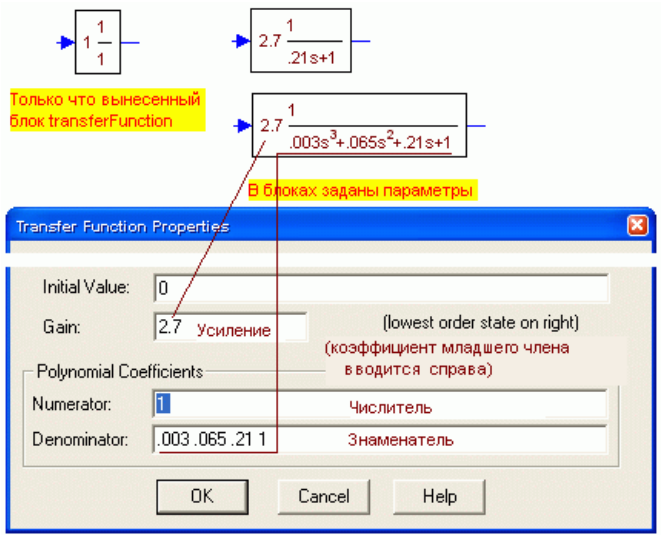


*Рис. 4.3.3. Параметры блока **transferFunction** на рабочем поле **VisSim***

Переходная функция ***h(t)*** это реакция, отклик линейной системы или звена на ступенчатое единичное воздействие ***1₀(t)***. Ступенчатое единичное воздействие ***1₀(t)*** это функция времени ***t,*** равная нулю, пока ***t*** меньше нуля и равная единице при ***t*** больше нуля. Пример (рис. 4.3.4):

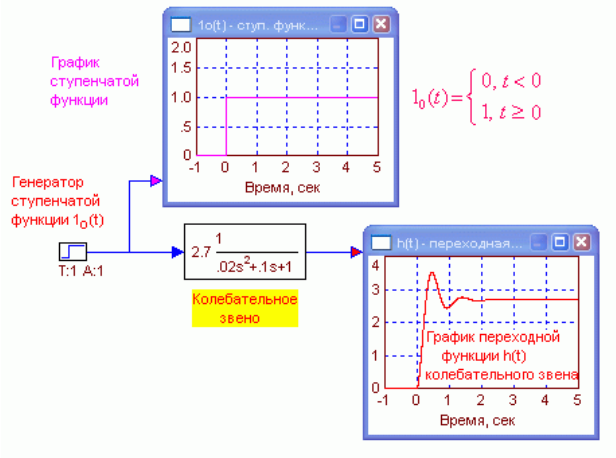


Рис. 4.3.4. Пример виртуального стенда, в VisSim

Переходная функция ***h(t)*** модели системы автоматического регулирования позволяет характеризовать ее качество (быстродействие и точность) в переходном режиме работы. Кроме того, зная переходную функ-

цию линейной системы можно определить реакцию системы на произвольное воздействие.

Примеры переходных функций типовых звеньев

Пропорциональное звено имеет переходную функцию $h(t) = k \cdot 1_0(t)$. Здесь k – коэффициент усиления звена. Пропорциональное звено не обладает инерцией, усиливает сигнал в k раз в любой момент времени, как бы быстро он не изменялся (рис. 4.3.5).

Интегратор способен накапливать поступающий на него сигнал с течением времени. В частности, рис. 4.3.6 показывает, что при подаче ступенчатого воздействия на интегратор, его выходной сигнал изменяется линейно с течением времени, т.е. накопление действительно происходит.

Рис. 4.3.5. Переходная функция пропорционального звена

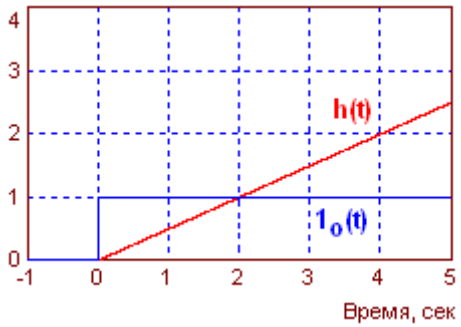


Рис. 4.3.6. Переходная функция интегратора

Апериодическое (инерционное) звено является простейшим из тех звеньев, которые обладают инерцией. Действительно, это звено не сразу, вначале быстро, а затем все более медленно, реагирует на ступенчатое воздействие (рис. 4.3.7).

Длительность переходного процесса в звене является мерой его инерционности.

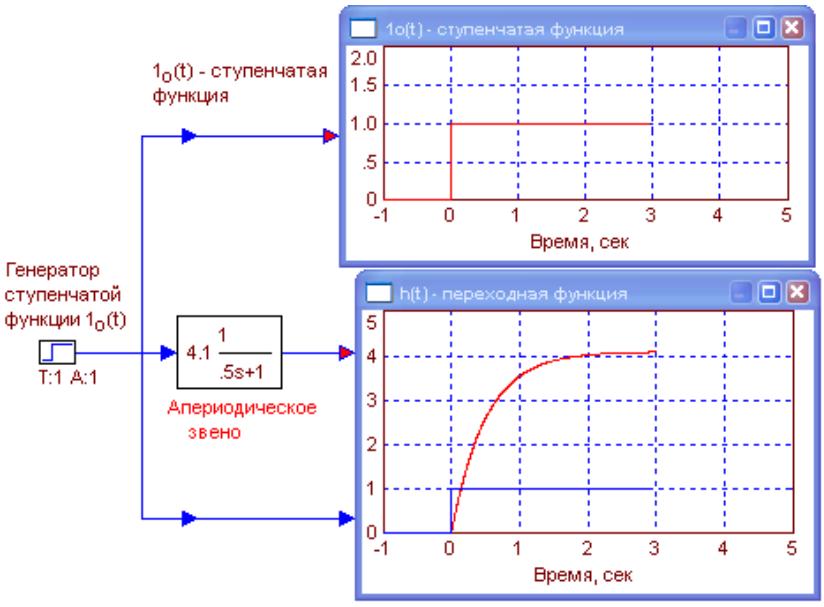


Рис. 4.3.7. Переходная функция апериодического звена

Колебательное звено наряду со свойствами, присущими уже перечисленным звеньям (способности к усилению, накоплению и инерционностью), обладает и еще одним свойством, которого нет у более простых звеньев, способно порождать колебания (рис. 4.3.8).

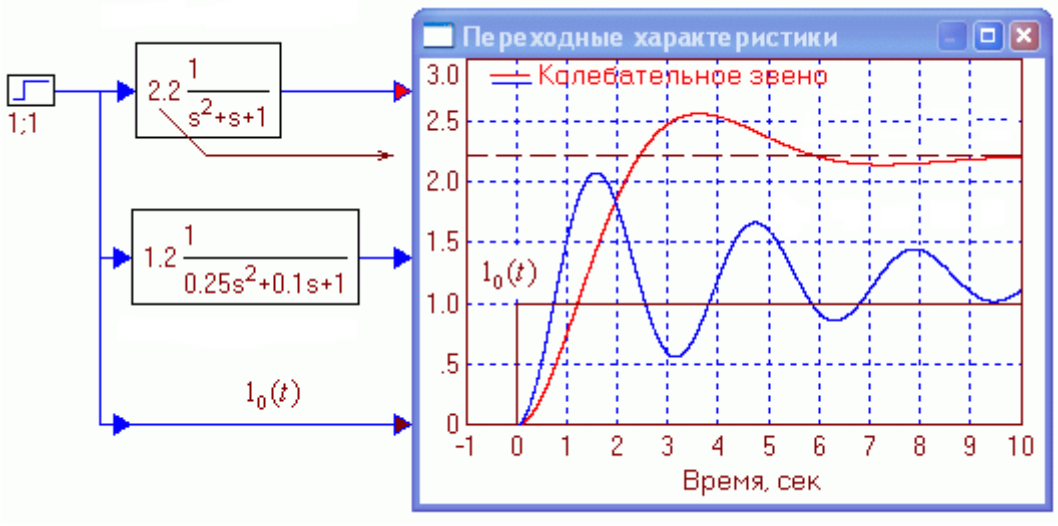


Рис. 4.3.8. Переходные функции колебательных звеньев

Звено запаздывания имеет переходную функцию по рис. 4.3.9.

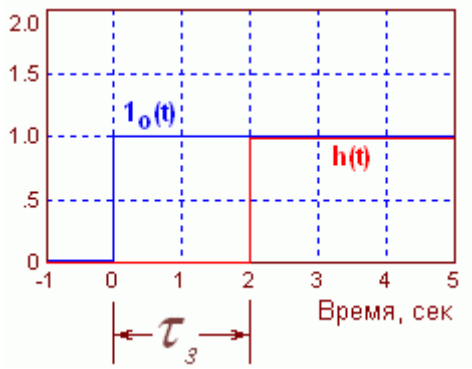


Рис. 4.3.9. Переходная функция звена запаздывания

Этим звеном моделируются системы и устройства, сигналы в которых задерживаются на ощутимую величину по сравнению с временными параметрами, характеризующими инерционность этих систем. Это, как правило, протяженные в пространстве устройства: линии связи, трубопроводы, транспортеры и т.п.

Задание 1. Построить в программе VisSim виртуальный стенд для исследования модели апериодического звена (рис. 4.3.10). Апериодическое звено создается вынесением на рабочее поле блока **transferFunction (Blocks – Linear System – transferFunction)** и заданием его параметров.

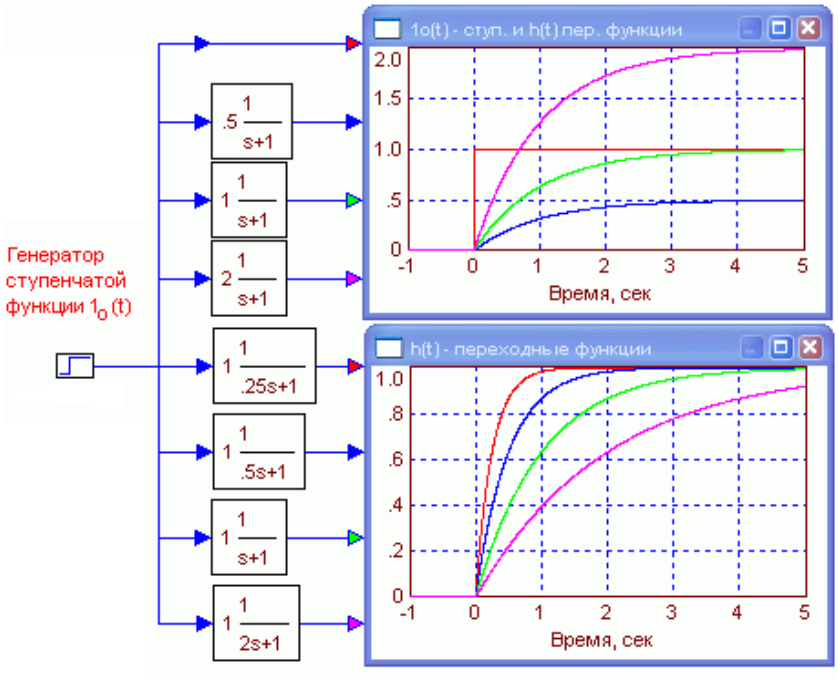


Рис. 4.3.10. Исследование апериодического звена

Параметры апериодического звена задаются в окне диалога, появляющегося при двойным щелчке по блоку **transferFunction** (рис. 4.3.11):

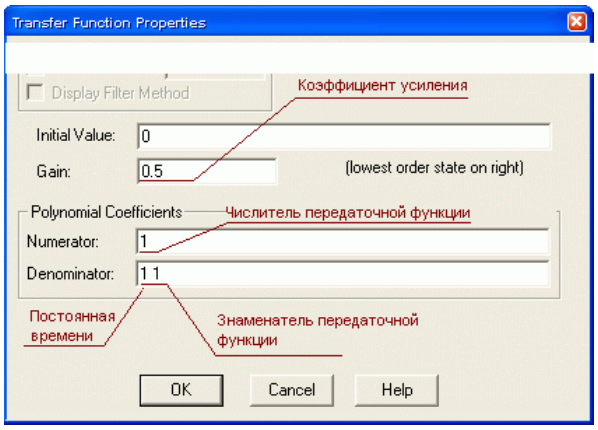


Рис. 4.3.11. Поля окна диалога для задания параметров апериодического звена

Задание 2. Построить в программе VisSim виртуальный стенд для исследования модели колебательного звена. Колебательное звено создается вынесением на рабочее поле блока **transferFunction** (**Blocks – Linear System – transferFunction**) и заданием его параметров (рис. 4.3.13).

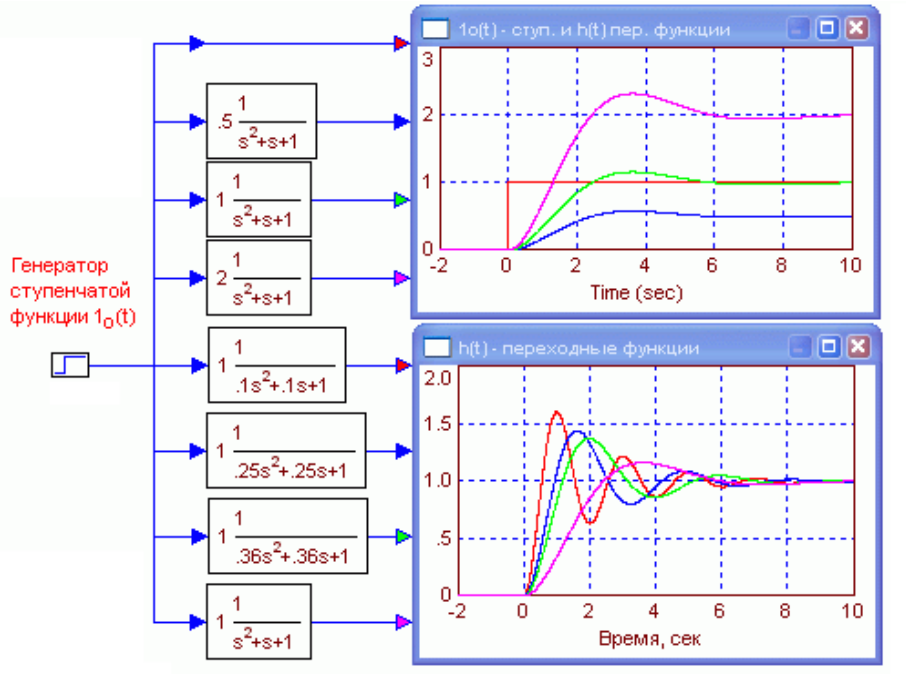


Рис. 4.3.12. Стенд для исследования колебательного звена

Числа (значения коэффициентов многочленов) должны быть отделены пробелами, аргумент ***s*** не вводится, программа выводит его в формуле сама.

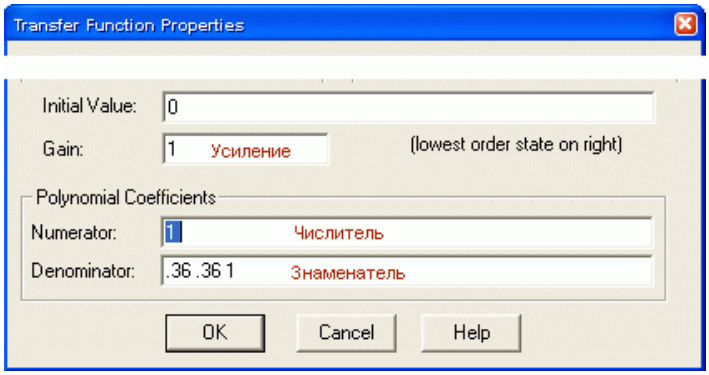


Рис. 4.3.13. Поля окна для задания параметров колебательного звена

Задание 3. Построить в программе VisSim виртуальный стенд для исследования модели сумматора (рис. 4.3.14). Сумматор выносится на рабочее поле из меню **Blocks – Arithmetic – SummingJunction**. Генераторы линейно растущего сигнала (**ramp**) и синусоидального (**sinusoid**) выносится из меню **Blocks – Signal Producer**.

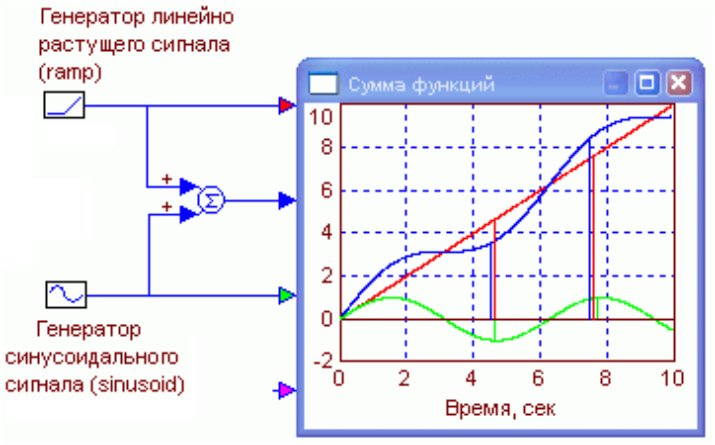
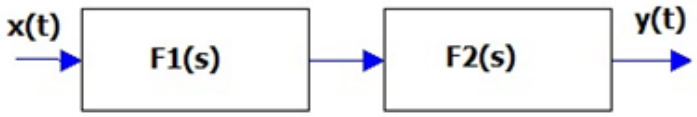


Рис. 4.3.14. Стенд для проверки работоспособности сумматора

Необходимо провести эксперимент: увеличить частоту генератора синусоиды и скорость роста линейного напряжения. Проверить, что сумматор успевает суммировать и более быстрые сигналы.

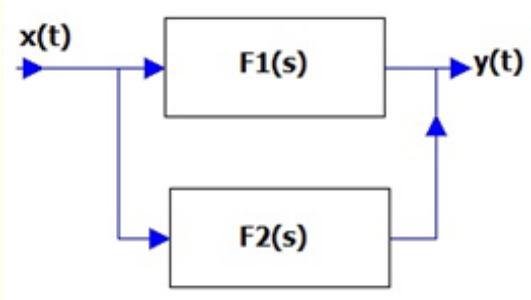
4.4. Построение передаточных функций эквивалентных схем

В системах управления звенья могут соединяться последовательно (рис. 4.4.1), параллельно (рис. 4.4.2) и по схеме отрицательной обратной связи (рис. 4.4.3).



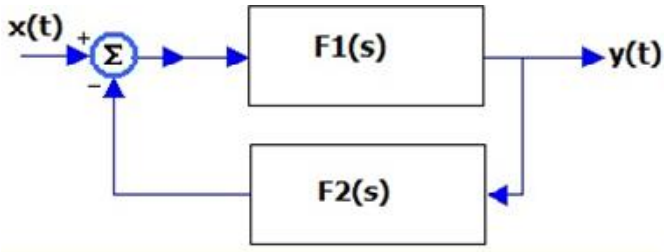
$$\Phi(s) = F_1(s)F_2(s)$$

Рис. 4.4.1. Последовательное соединение



$$\Phi(s) = F_1(s) + F_2(s)$$

Рис. 4.4.2. Паралельное соединение



$$\Phi(s) = \frac{F_1(s)}{1 + F_1(s)F_2(s)}$$

Рис. 4.4.3. Соединение – отрицательная обратная связь

Средствами VisSim построить эквивалентные схемы. Пример представлен на рис. 4.4.4. Соединение блоков заменяется блоком с эквивалентной передаточной функцией *Ф(s)*. Эквивалентность проверяется компьютерным экспериментом – переходные процессы совпадают (рис. 4.4.4).

Задание по работе. Построить эквивалентные схемы по рис. 4.4.1–4.4.3. В качестве F_1 и F_2 можно использовать передаточные функции инерционных звеньев (рис. 4.4.4), преподаватель может дать другой вариант F_1 и F_2.

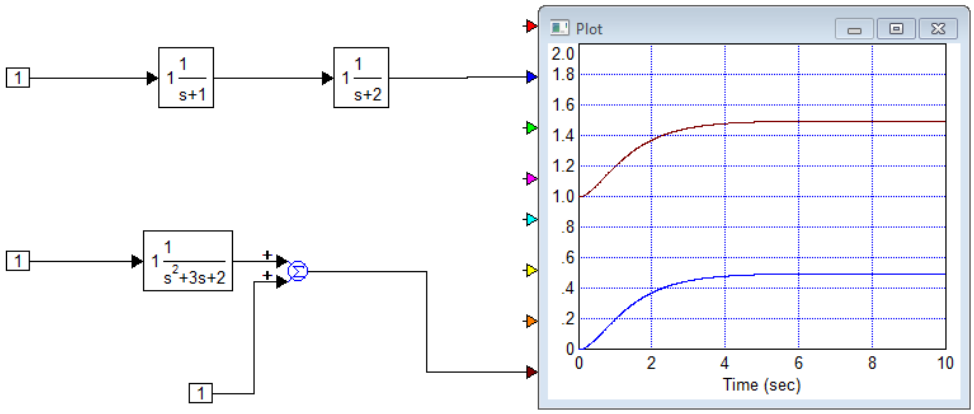


Рис. 4.4.4. Модель эквивалентной схемы

Эквивалентность проверить экспериментом (рис. 4.4.4). Формулы эквивалентных передаточных функций вывести самостоятельно.

4.5. Моделирование системы регулирования

Системы регулирования предназначены для поддержания определенного состояния объекта. Рассмотрим систему регулирования частоты вращения двигателя постоянного тока (рис. 4.5.1). Задача системы – стабилизация частоты вращения двигателя постоянного тока при его включении или при действии внешней механической нагрузки.

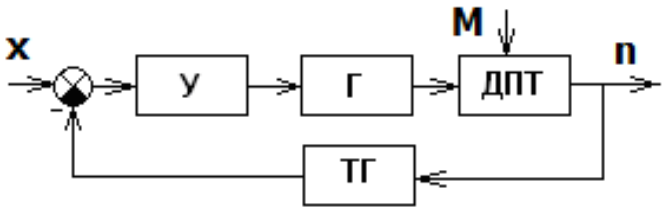


Рис. 4.5.1. Схема системы управления с обратной связью ДПТ

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

Постановка задачи моделирования. Построить VisSim-модель системы регулирования (рис. 4.5.2).

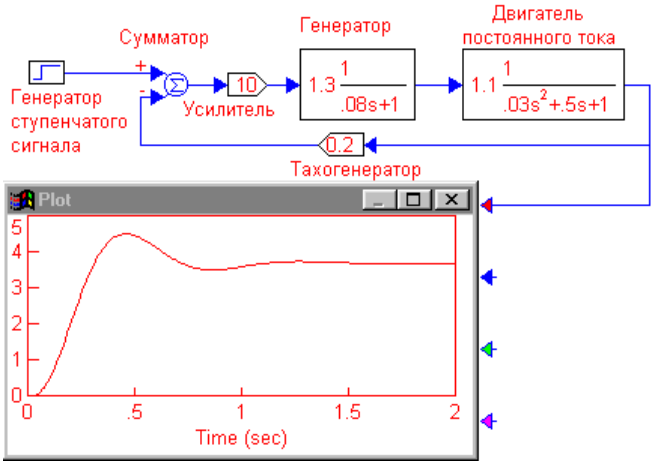


Рис. 4.5.2. Переходная функция замкнутой системы регулирования

Построить переходную функцию замкнутой системы регулирования (рис. 4.5.2) и разомкнутой системы (рис. 4.5.3). Исследовать устойчивость системы. Построить модель системы по рис. 4.5.4; 4.5.5.

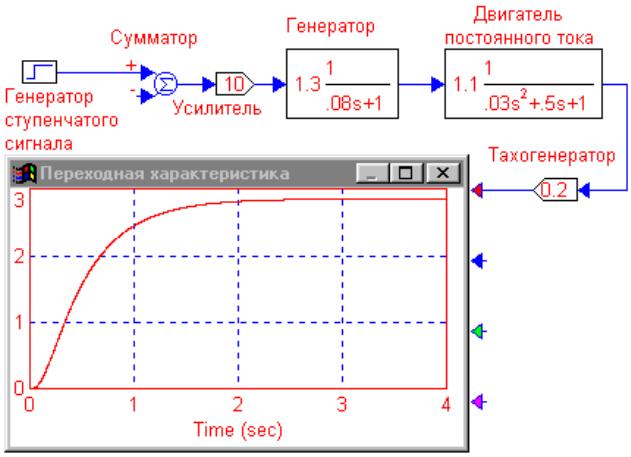


Рис. 4.5.3. Разомкнутая система регулирования

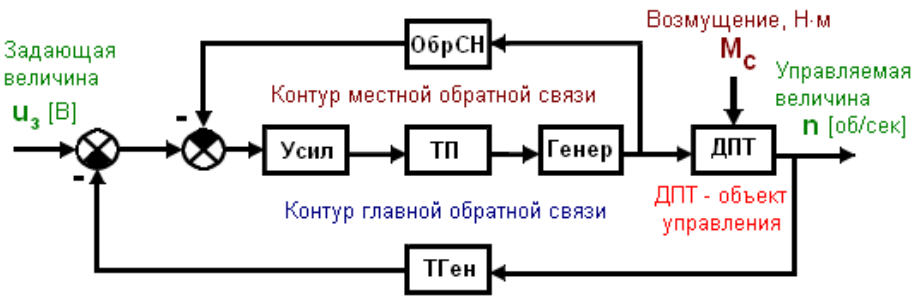


Рис. 4.5.4. Функциональная схема системы автоматического регулирования двигателя постоянного тока

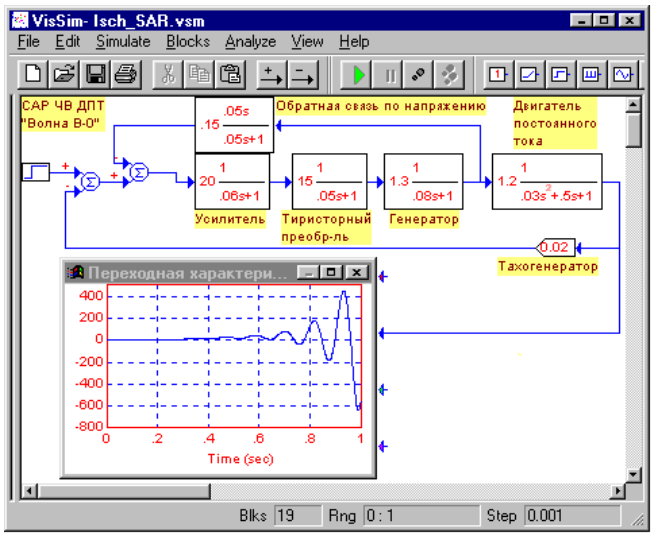


Рис. 4.5.5. Система неустойчива

Объект управления – двигатель постоянного тока (рис. 4.5.4), управляемая величина – частота вращения вала ДПТ. Система содержит контуры обратной связи, и может быть неустойчивой (рис. 4.5.5). В предлагаемой схеме фактически изменять можно только параметры усилителя и звена местной обратной связи. Эти элементы введены в схему как раз для того, чтобы обеспечить возможность ее коррекции. Не исключено, что параметры звена обратной связи по напряжению заданы неудачно и потребуют существенной коррекции.

Рис. 4.5.6. показывает, что и разомкнутая система неустойчива. Для обеспечения устойчивости, разомкнутую системы требуется стабилизировать.

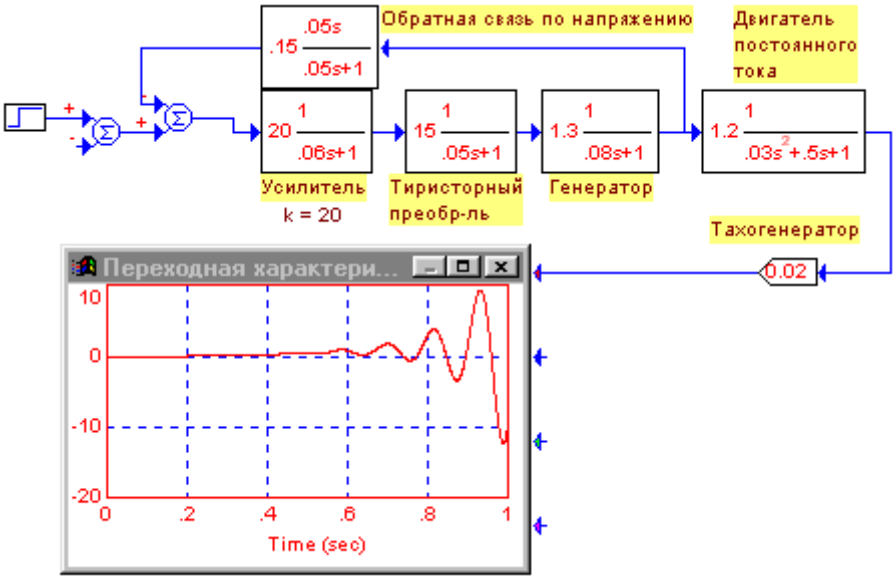


Рис. 4.5.6. Проверка устойчивости разомкнутого контура системы

Разомкнем контур главной обратной связи, подключим его к осциллографу и запустим моделирование (рис. 4.5.6):

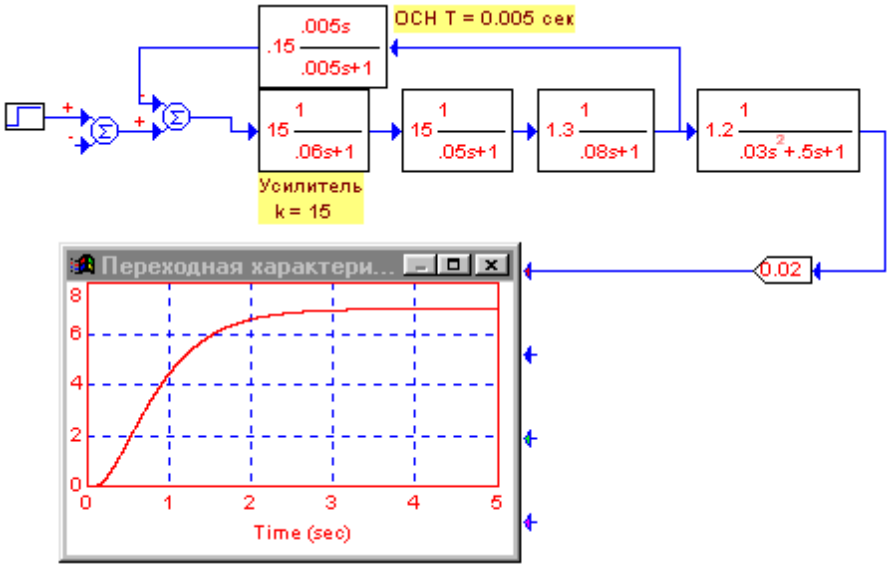


Рис. 4.5.7. Стабилизированный разомкнутый контур

Итак (рис. 4.5.7), разомкнутая система стабилизирована. Устойчива ли замкнутая система, разомкнутый контур которой только что стабилизирован. Для этого замкнем обратную связь и проверим, как поведет себя переходная характеристика системы (рис. 4.5.8):

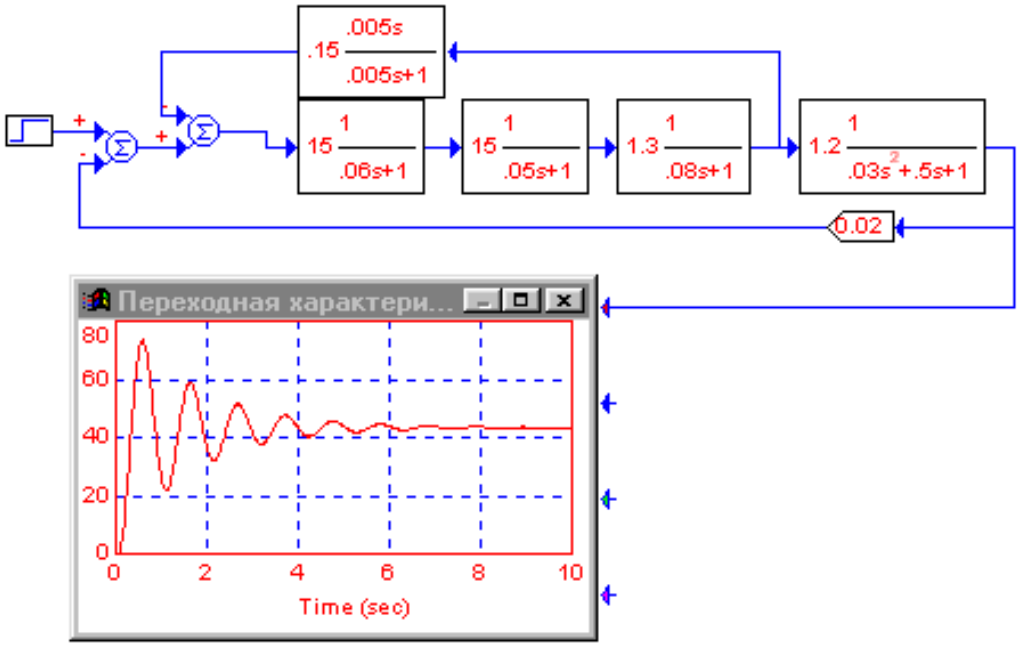


Рис. 4.5.8. Система после стабилизации разомкнутого контура

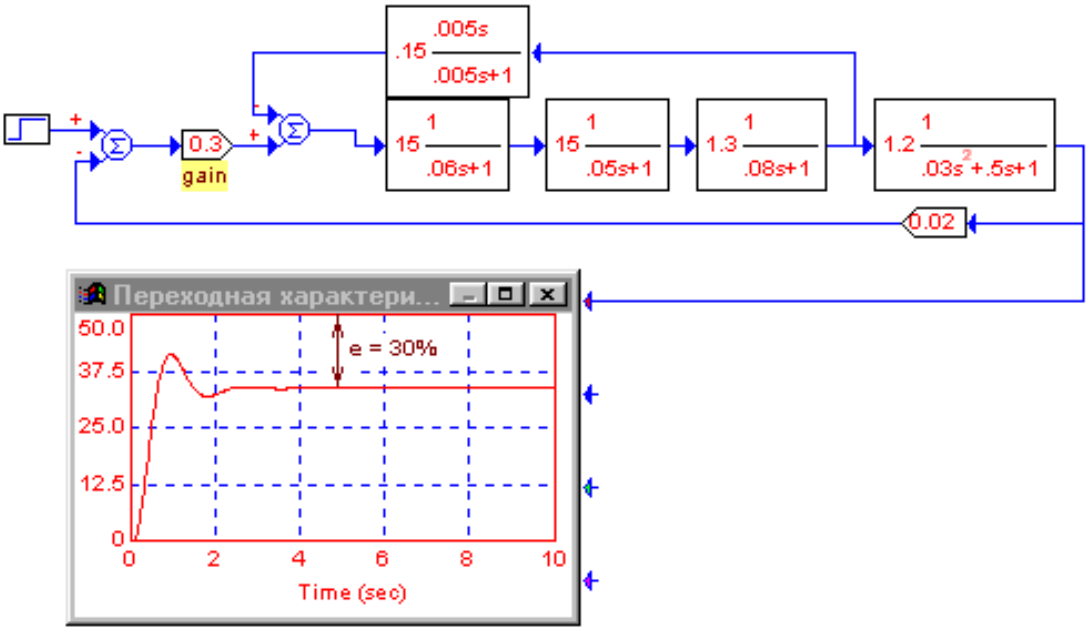


Рис. 4.5.9. Переходная характеристика скорректированной системы

Система устойчива, но ее способность к совершению колебаний чрезмерна (рис. 4.5.8). Замкнутая система требует коррекции (рис. 4.5.9).

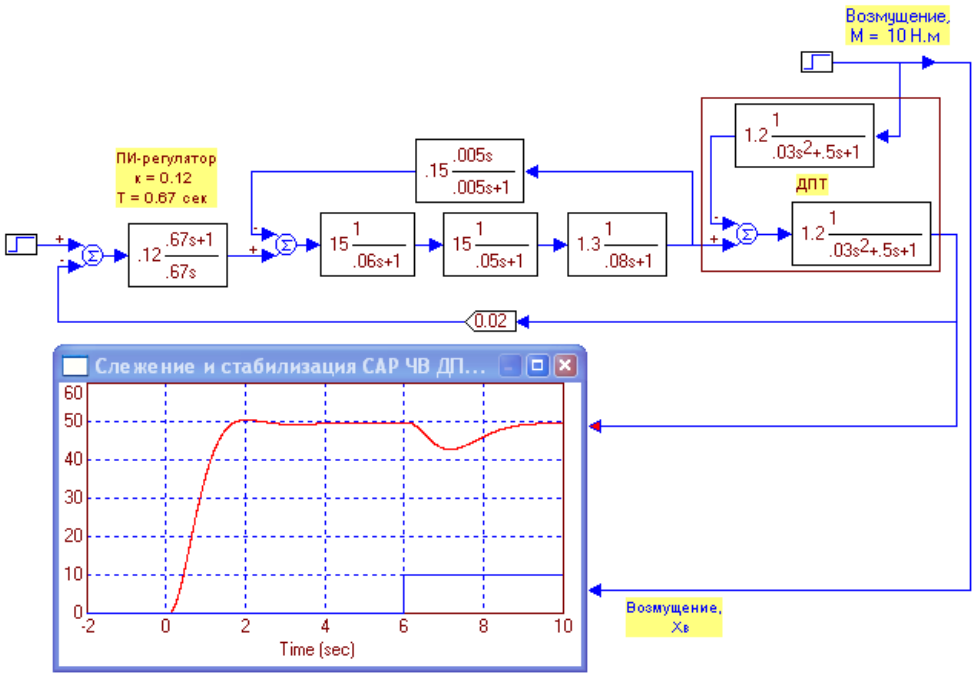


Рис. 4.5.10. Совокупное воздействие возмущений на систему регулирования

Теперь построим VisSim модель системы регулирования по рис. 4.5.4. Должна быть обеспечена стабилизация частоты вращения двигателя постоянного тока при его включении и при воздействии внешней механической нагрузки (рис. 4.5.10).

Первоначально ДПТ включается, затем через 6 сек прикладывается внешнее воздействие (механическая нагрузка). По переходной характеристике видно, что система регулирования компенсирует возмущение примерно за 3 сек. (рис. 4.5.10).

4.6. Контрольные вопросы к главе 4

1. Назначение программы VisSim.
2. Виды блоков VisSim.
3. Особенности переходной функции инерционного звена.
4. Особенности переходной функции колебательного звена.
5. Виды эквивалентных схем.

ГЛАВА 5. МОДЕЛИРОВАНИЕ В СРЕДЕ ANYLOGIC

5.1. Инструментальная система моделирования AnyLogic

Главной особенностью программного комплекса AnyLogic является простота и удобство использования. Благодаря графической среде моделирования создавать объекты исследования и управлять ими не составляет труда. Моделирование в AnyLogic построено на объектно-ориентированном принципе. Любая модель состоит из ряда объектов, каждый из которых имеет свои функции и взаимодействует с окружающей средой. (URL: https://redsoft.club/sistema/razrabotchiku/ anylogic содержит видео о системе AnyLogic).

Для того чтобы скачать и установить программу AnyLogic, следует выполнить ряд простых действий:

– Зайти на официальный сайт anylogic.ru и нажать на кнопку «Скачать» в правом верхнем углу.

– Выбрать интересующую версию программы и повторно нажать «Скачать».

– Загрузить установочный файл, предварительно указав операционную систему устройства. Программа корректно работает на операционных системах Windows 64, начиная с 7 версии, Mac OS X и Linux. Версия для самостоятельного обучения AnyLogic Personal Learning Edition (PLE) устанавливается без активации. Наиболее же функционально полная версия AnyLogic Professional требует пройти несложную операцию активации.

После проделанных шагов начнется скачивание файла. Объем файла до 1 Гб, поэтому процедура загрузки занимается несколько минут. Для запуска установки необходимо дважды нажать на загруженный файл, чтобы

запустился «Мастер установки». Следуйте всем инструкциям, чтобы успешно завершить установку.

Из основных шагов стоит выделить:

принятие лицензионного соглашения;

настройка директории установки и языка программы.

После этого начнется копирование файлов, а по завершению автоматически откроется «Мастер активации». Пользователям предлагается два варианта:

1. Использовать купленный ключ для активации программы.

2. Запросить ознакомительный ключ для временного бесплатного доступа.

В случае с платной версией программы достаточно ввести код активации и можно начинать работу. Но для ознакомительной версии необходимо запросить ключ активации, заполнить личные данные и ввести действующий адрес электронной почты, на которое придет сообщение с кодом.

Скачивать программу необходимо с официального сайта компании https://www.anylogic.ru/downloads/. Скачивать для учебных целей студентам подойдет версия 8.7.9 University Researcher для открытых исследований в университетах.

Система AnyLogic, разработанная компанией XJ Technologies (Россия 1999 г., сейчас это фирма AnyLogic), это среда компьютерного моделирования общего назначения. Это комплексный инструмент, охватывающий основные в настоящее время направления моделирования: дискретно-событийное моделирование, моделирование системной динамики и агентное моделирование. Использование AnyLogic дает возможность оценить эффект принимаемых решений в сложных системах реального мира.

В AnyLogic разработчик может гибко использовать различные уровни абстрагирования и различные стили, и концепции, и смешивать их при создании одной и той же имитационной модели (рис. 5.1.1).

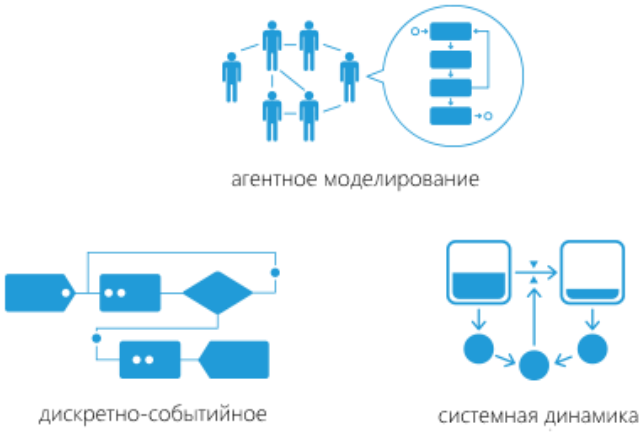


Рис. 5.1.1. Виды моделирования в AnyLogic

Можно выделить шесть основных преимуществ имитационного моделирования, которые присущи AnyLogic:

1. Имитационные модели позволяют анализировать системы и находить решения в тех случаях, когда такие методы, как аналитические вычисления и линейное программирование не справляются с задачей.

2. После того, как вы определитесь с уровнем абстракции, разрабатывать имитационную модель будет гораздо проще, чем аналитическую, поскольку процесс создания модели будет модульным.

3. Структура имитационной модели естественным образом отображает структуру моделируемой системы.

4. Имитационная модель позволяет отслеживать все объекты системы, учтенные в выбранном уровне абстракции, добавлять метрики и проводить статистический анализ.

5. Одним из главных преимуществ имитационного моделирования является возможность проигрывать модель во времени и анимировать ее поведение. Анимация будет неоспоримым преимуществом при демонстрации модели и может оказаться полезной для верификации модели и нахождения ошибок.

6. Имитационные модели намного убедительнее электронных таблиц. Если вы используете имитационное моделирование, то при презента-

ции проекта у вас будет явное преимущество перед теми, у кого на руках только цифры и решение, полученное из «черного ящика».

AnyLogic, так же как и MVS, использует объектно-ориентированный подход к представлению сложных систем. Этот подход позволяет простым и естественным образом организовать и представить структуру сложной системы с помощью иерархии абстракций. Например, на некотором уровне абстракции автомобиль можно считать неким единым объектом. Но более детально его можно представить как совокупность взаимодействующих подсистем: двигателя, рулевого управления, тормозной системы и т.п. Каждая из этих подсистем может быть представлена, если это необходимо, своей структурой взаимодействующих подсистем.

Именно такую иерархию абстракций позволяет создать AnyLogic при разработке моделей. Всю модель можно рассматривать как единый объект.

При построении модели в среде AnyLogic не предусмотрена возможность использования никаких других средств, кроме средств визуальной разработки (введения состояний и переходов, введения пиктограмм переменных и т.п.), задания численных значений параметров, аналитических записей соотношений переменных и аналитических записей условий наступления событий. Основной парадигмой, принятой в AnyLogic при разработке моделей, является **визуальное проектирование** – построение с помощью графических объектов и пиктограмм иерархий структуры и поведения активных объектов.

Решения, которые позволяет принять AnyLogic-моделирование (рис. 5.1.2):

1. Управление сложными современными производственными процессами и финансовыми операциями требует оперативности, планирования и оценки результата. AnyLogic помогает оценить последствия принятия нескольких альтернативных решений и выбрать лучшее; использовать прогностическую модель для оперативного, среднесрочного или стратегического планирования; оценить работу процесса «в виртуальном мире», проверить изменения до их реализации.

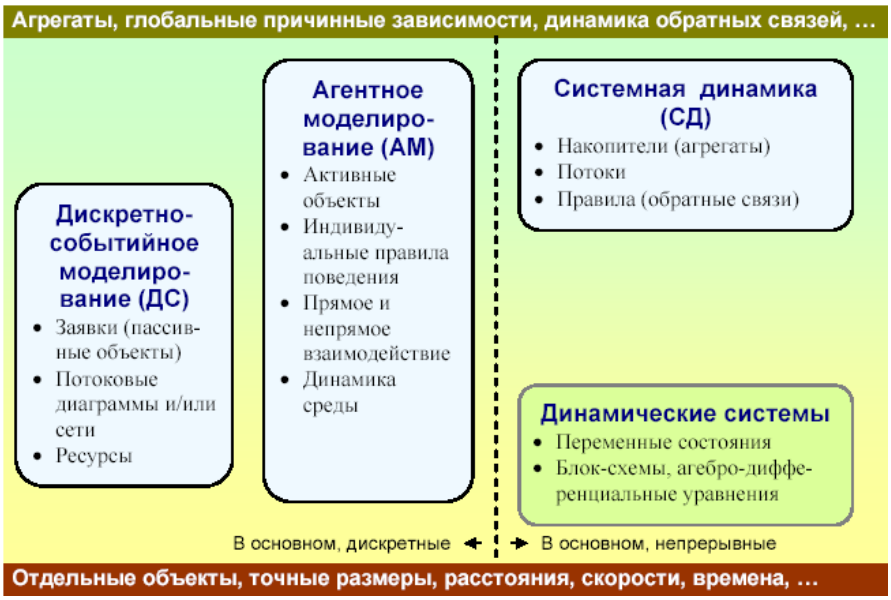


Рис. 5.1.2. Методы моделирования, реализуемые в AnyLogic

2. По построенной компьютерной модели, имитирующей работу реальной системы, возможно исследовать производительность системы, ее «узкие места», оценить влияние параметров (например, оценить, куда направить ограниченный объем капиталовложений).

3. Проводить эксперименты «что, если...», оценивать альтернативные варианты и выбирать технологию, дающую наилучшие результаты.

4. Проводить моделирование и компьютерную имитацию функционирования сложных систем совместно с их визуализацией, что становятся повседневным современным средством поддержки и обоснования принятия решений (рис. 5.1.3).

5. Для моделирования **систем с дискретными событиями** существует множество пакетов, облегчающих разработку дискретно-событийных моделей и проведение экспериментов с моделями в этой традиционной области моделирования.

Моделирование системной динамики – это методология изучения и моделирования систем, характеризующихся циклами обратных связей в сложных взаимных причинных зависимостях их параметров.

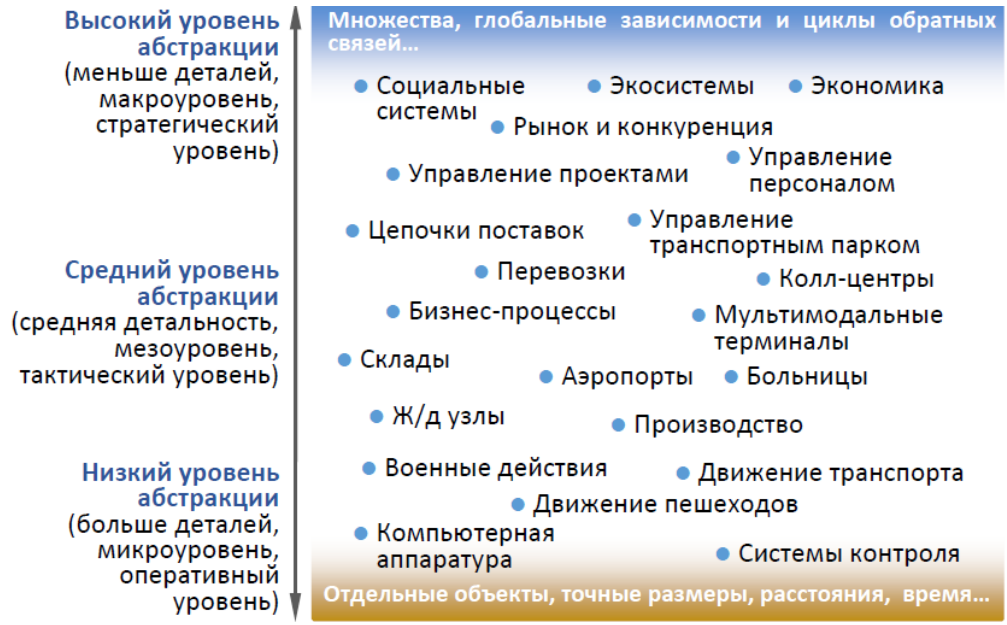


Рис. 5.1.3. Задачи, решаемые в AnyLogic

Математически эти системы описываются системами дифференциальных уравнений, приведенных к форме Коши. Такие модели рассматривали при изучении пакета MVS. Эти модели применяются для корпоративного планирования и анализа политики управления корпорацией, политики управления социальными и экономическими системами, в экологии и т.п. Джей Форрестер – американский инженер и системолог, разработчик теории системной динамики – в 1970-х гг. разработал модели «Мир-1» и «Мир-2», нацеленные на выработку сценариев развития всего человечества в его взаимоотношении с биосферой. Так родилось первое поколение компьютерных моделей, предназначенных для исследования долгосрочных тенденций мирового развития. Он опубликовал книгу «Мировая динамика», в которой обобщил свой вклад в создание первых компьютерных моделей, анализирующих глобальную систему.

Агентное моделирование. Под агентом понимается активный объект, обладающий поведением и имеющий возможность взаимодействия с другими агентами и со средой. Многоагентное моделирование позволяет вывести характеристики целого (множества агентов) из совокупности ло-

кальных поведений и характеристик отдельных активных элементов целого, распределенных в среде.

Моделирование многоагентных систем используется при анализе социальных процессов, процессов урбанизации и даже при исследовании рынка в анализе предпочтений различных социальных групп или корпораций, выступающих как агенты со своим поведением.

С AnyLogic разработчик не ограничен одним методом моделирования. Используя подходящий метод или их комбинацию, он может создать оптимальную для решения конкретной проблемы имитационную модель.

Таким образом, идеи и методы, направленные на управление сложностью, выработанные в последние десятилетия в области создания программных систем, позволяют разработчикам моделей в среде AnyLogic структурировать разработку и, в конечном счете, упростить и ускорить создание моделей.

5.2. AnyLogic-модель сервера

Постановка задачи. Рассмотрим задачу построения модели сервера, описанную в работе В. Д. Боева [3]. При клиент-серверном взаимодействии между браузерами и удаленным сервером происходит обработка множества запросов со средним значением 1 минута (используется экспоненциальный закон распределения). Пусть сервер обрабатывает один запрос со средним значением 10 секунд с таким же распределением. Максимальное количество запросов, которые сервер может хранить в своем буфере, установим в 50 запросов. Необходимо построить имитационную модель работы сервера для подсчета количества обработанных запросов и количества отказов, а также вероятности обработки запроса.

Модель будет представлять собой систему массового обслуживания (СМО). Для построения таких имитационных моделей в AnyLogic используется палитра компонентов.

Модель будет состоять из следующих блоков: **«Source», «Queue», «Delay», «Sink»** (рис. 5.2.1).

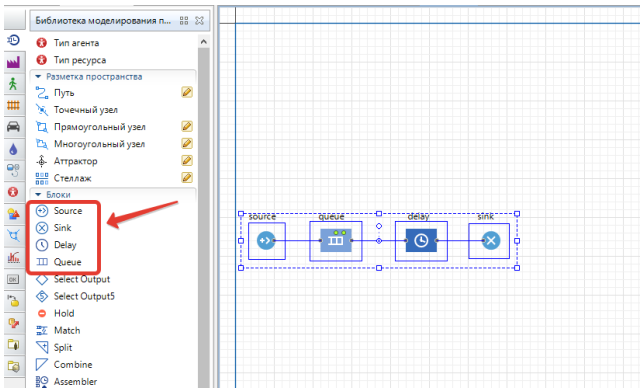


Рис. 5.2.1. Установка объектов для модели сервера

Объект **«Source»** генерирует заявки (**entities**) определенного типа через заданный временной интервал. Заявки представляют собой объекты, которые обрабатываются, обслуживаются или еще каким-нибудь образом подвергаются действию моделируемого процесса: это могут быть клиенты в системе обслуживания, детали в модели производства, документы в модели документооборота и т.д. В нашем примере заявками будут запросы к серверу от рабочих станций, а объект **«Source»** будет моделировать их поступление серверу.

Чтобы узнать детальное описание объектов библиотеки моделирования процессов, достаточно навести на них мышью в палитре и немного «зависнуть» на нем. Появится всплывающее окно, в котором представлено подробное описание объекта.

Объект **«Queue»** моделирует очередь из запросов, ожидающих обработки сервером.

Объект **«Delay»** моделирует задержку, которая вызвана обслуживанием. В нашем примере это будет время обработки сервером одного запроса.

Объект **«Sink»** обозначает конец блок-схемы и выполняет удаление обработанных запросов из системы.

Создание модели

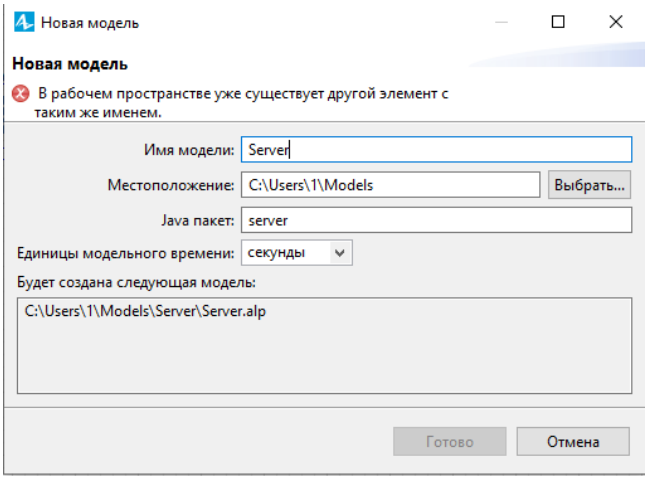


Рис. 5.2.2. Создание проекта Server

1. Для создания нового проекта в AnyLogic: выполним **«Файл» – «Создать» – «Модель»**. Появится диалоговое окно **«Новая модель»** (рис. 5.2.2). Задайте имя новой модели. В поле **«Имя модели»** введите «**Server»**.

2. Выберите каталог, в котором будут сохранены файлы модели. Если хотите сменить предложенный по умолчанию каталог на какой-то другой, то можете ввести путь к нему в поле **«Местоположение»** или выбрать этот каталог с помощью диалога навигации по файловой системе, открывающегося нажатием кнопки **«Выбрать»**. Щёлкните кнопку **«Готово»**.

3. Далее откроется пользовательский интерфейс (рис. 5.2.3). Рассмотрим элементы окна проекта.

4. В левой части рабочей области находятся панель **«Проект»** и панель **«Палитра»**. Панель **«Проект»** обеспечивает навигацию по элементам моделей, открытых в текущий момент времени. Модель организована иерархически. Она отображается в виде дерева. Сама модель образует верхний уровень дерева. Эксперименты, классы активных объектов и Java-классы образуют следующий уровень. Элементы, входящие в состав активных объектов, вложены в соответствующую подветвь дерева класса активного объекта и т.д.

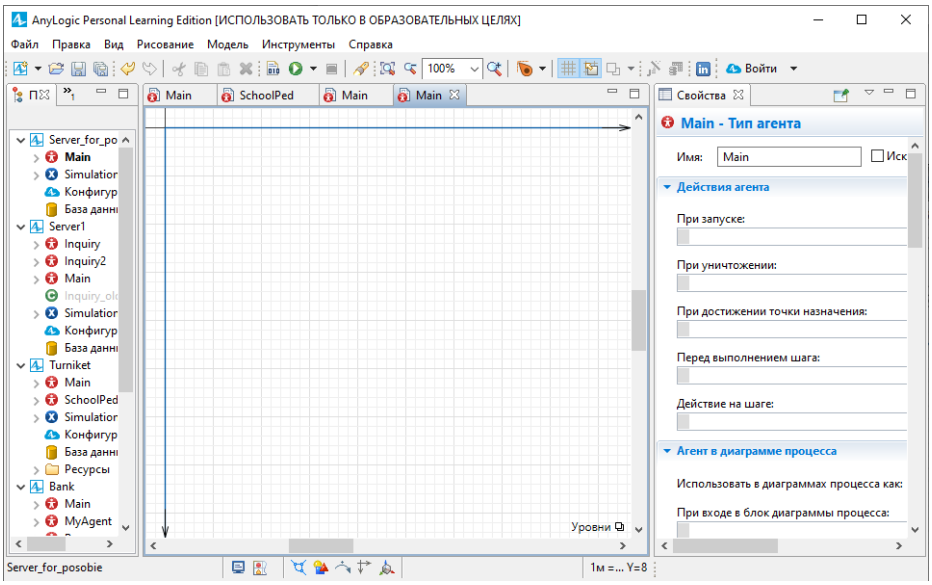


Рис. 5.2.3. Элементы окна проекта

Панель **«Палитра»** содержит разделённые по категориям элементы, которые могут быть добавлены на графическую диаграмму типа агентов или эксперимента (на рис. 5.2.1 раскрыта панель **«Палитра»**). Для того чтобы открыть нужную палитру, следует подвести курсор к иконке и щелкнуть мышью. Иконка будет иметь рамку выделения.

В правой части отображается панель **«Свойства»**. Панель **«Свойства»** используется для просмотра и изменения свойств выбранного в данный момент элемента (или элементов) модели.

В центре рабочей области AnyLogic размещён графический редактор диаграммы агента **«Main»**.

5. Создайте диаграмму процесса. Для этого в **«Палитре»** выделите **«Библиотеку моделирования процессов»** (рис. 5.2.1). Из неё перетащите объекты **«Source», «Queue», «Delay», «Sink»** на диаграмму и соедините, как на рис. 5.2.1.

Для добавления объекта на диаграмму, надо щёлкнуть его мышью и, не отпуская её, перетащить в графический редактор.

1. При перетаскивании объектов вы можете видеть, как появляются соединительные линии между объектами. Эти соединительные линии должны соединять только **порты**, находящиеся слева и справа от объекта. Если у вас не получится автоматическое соединение нужных объектов, то можно

дважды щёлкнуть на начальный порт. Он станет зелёным. Затем нужно протащить курсор к конечному порту. Он также станет зелёным. Для завершения процесса соединения дважды щёлкните конечный порт (рис. 5.2.4).

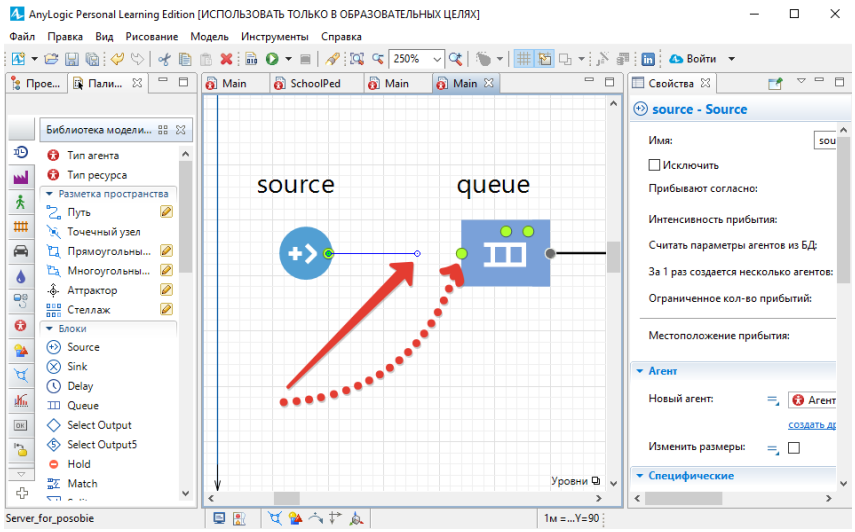


Рис. 5.2.4. Соединение портов двух объектов

2. Введем параметр **«time_mean»** – параметр со значением 10 (среднее время обработки запроса сервером). Для этого на **«Палитре»** перейдите на вкладку **«Агент»** и перетащите объект **«Параметр»** в графический редактор (рис. 5.2.5). В свойствах параметра **«time_mean»** установите **«Тип – int»**, значение поля **«Значение по умолчанию»** равным 10.

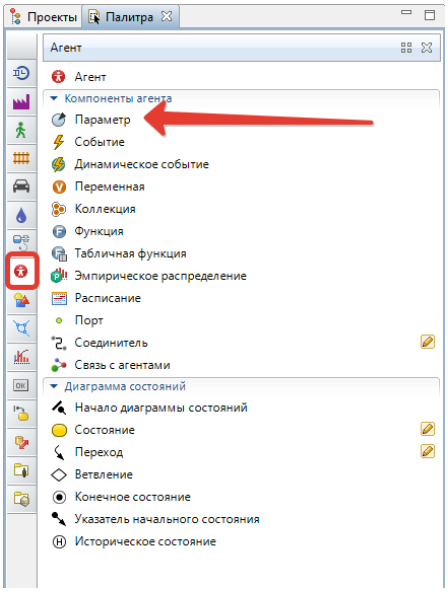


Рис. 5.2.5. Создание параметра time_mean

3. Аналогично предыдущему пункту создайте параметр **«emkBuf»** (емкость буфера сервера) и задайте ему **«Тип – int»**, **«Значение по умолчанию»** равно 50.

4. Установите для объектов следующие свойства (выделяя нужный объект, справа меняется окно **«Свойства»**):

- **«Source»** – время между прибытиями заявок на обслуживание, задается функцией **exponential(time_mean)**. Заявки прибывают согласно **«Времени между прибытиями».**
- **«Queue»** – вместимость очереди, в поле **«Вместимость»** внести «**emkBuf»**;
- **«Delay»** – время обработки сервером сигнала (поле **«Время задержки»**), значение вычисляется функцией **exponential (time_mean);**
- **Sink** – уничтожение заявки (запрос считается обработанным), нет параметров.

Запуск модели

1. Уже с данными параметрами модель можно запустить и изучить ее поведение. Вы можете сконфигурировать выполнение модели в соответствии с вашими требованиями. Модель выполняется с тем набором установок, которые задаются специальным элементом модели — **«Эксперимент»**. Можно создать несколько экспериментов с различными параметрами и изменять конфигурацию модели, просто запуская тот или иной эксперимент модели.

2. В панели **«Проект Simulation: Main»** эксперименты отображаются в дереве модели. Один эксперимент, названный **«Simulation»**, создается по умолчанию (см. справа). Это простой эксперимент, позволяющий запускать модель с заданными значениями параметров, поддерживающий режимы виртуального и реального времени, анимацию и отладку модели. Если нужно наблюдать поведение модели в течение длительного периода (до того момента, пока вы сами не остановите выполнение модели), то по

умолчанию времени остановки нет. Обработку запросов сервером планируется исследовать в течение одного часа, т.е. 3600 с.

3. В панели **«Проект»** выделите эксперимент **«Simulation:Main»**.
4. Щелчком раскройте вкладку **«Модельное время»**.
5. Установите **«Режим выполнения»** равным **«Виртуальное время» (максимальная скорость)** (рис. 5.2.6).

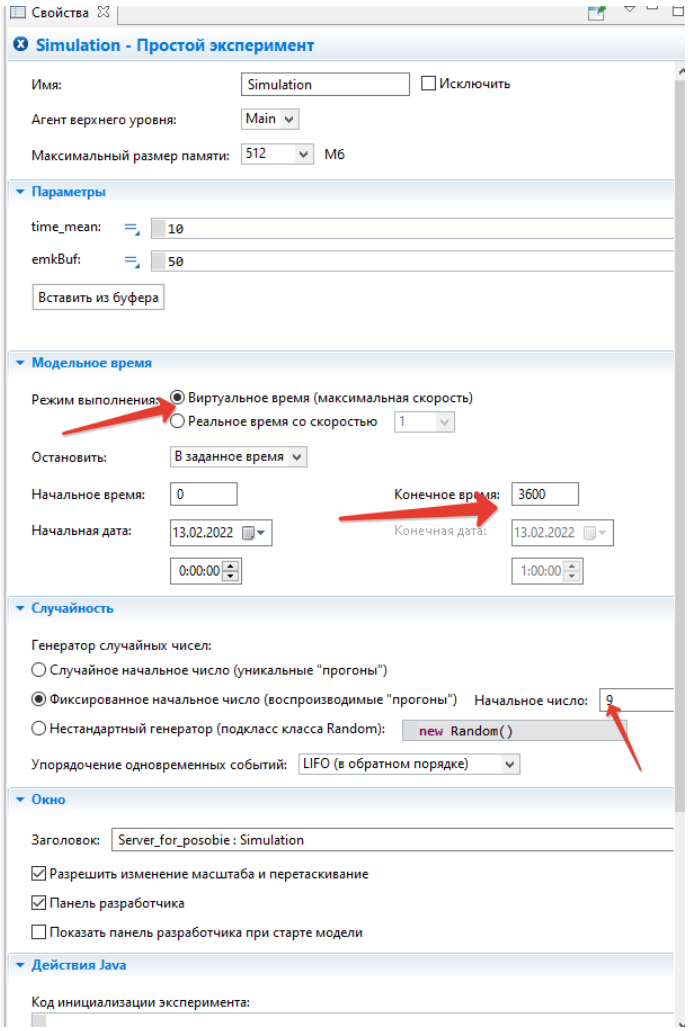


Рис. 5.2.6. Настройка параметров симуляции модели

6. В поле **«Остановить»** выберите из списка **«В заданное время»**.
7. В поле **«Конечное время»** установите 3600.
8. Раскройте вкладку **«Случайность»**. Выберите опцию **«Фиксированное начальное число» (воспроизводимые прогоны)**. В поле **«Начальное число»** установите **9**.

9. В панели **«Проект»**, выделите **«Server»**. Из выпадающего списка **«Единицы модельного времени»** выберите **секунды** (рис. 5.2.7).

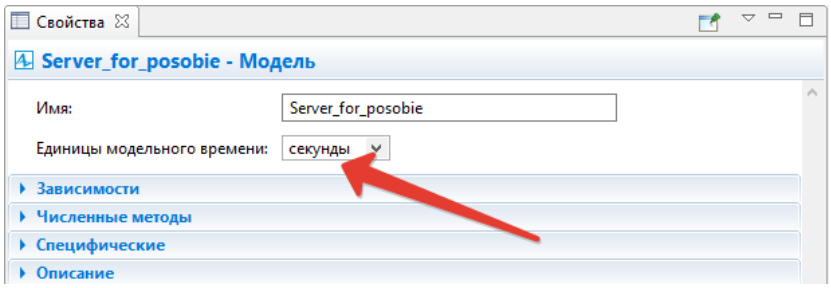


Рис. 5.2.7. Настройка единиц модельного времени

10. Запуск модели выполняется нажатием кнопки  на панели инструментов. После запуска появится окно симуляции модели и можно посмотреть, как она работает (рис. 5.2.8).

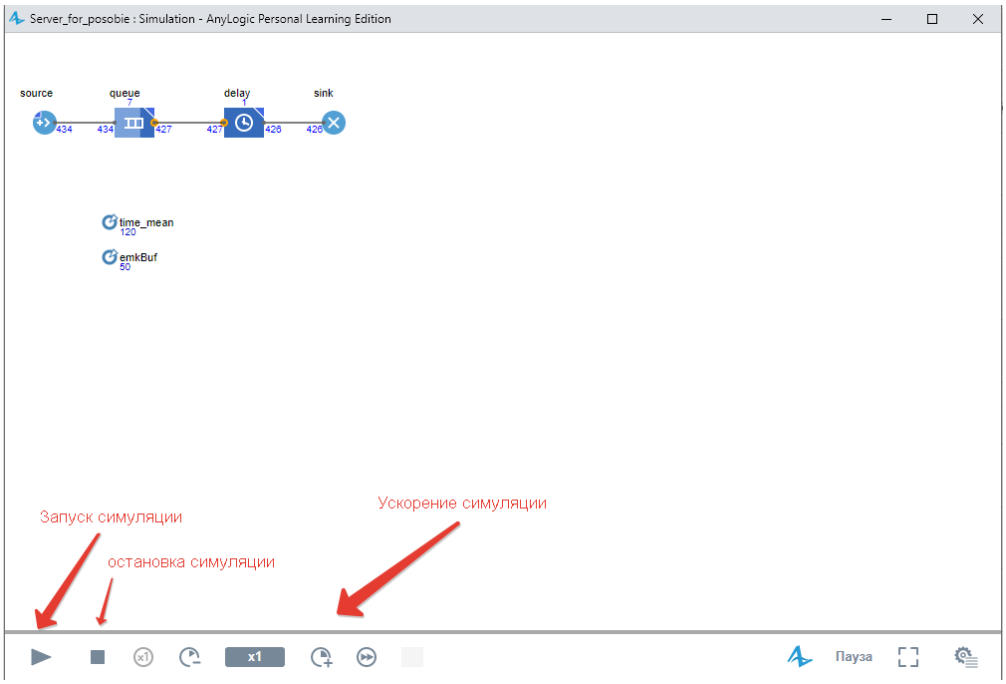


Рис. 5.2.8. Запуск модели на симуляцию

11. В дальнейшем нажатием кнопки **«Запустить»** будет запускаться тот эксперимент, который запускался в последний раз. Чтобы выбрать другой эксперимент, вам будет нужно щелкнуть мышью по стрелке, находящейся в правой части кнопки **«Запустить»**, и выбрать нужный вам эксперимент из открывшегося списка (или щелкнуть правой кнопкой мыши по

этому эксперименту в панели **«Проект»** и выбрать **«Запустить»** из контекстного меню).

12. После запуска модели вы увидите окно презентации этой модели. В нем будет отображена презентация запущенного эксперимента. AnyLogic автоматически помещает на презентацию каждого простого эксперимента заголовок и кнопку, позволяющую запустить модель и перейти на презентацию, нарисованную вами для главного класса активного объекта этого эксперимента («**Main**»). Нажмите на кнопку запуска модели (рис. 5.2.8).

Рис. 5.2.9. Панель управления запуском модели

13. После запуска модели вы увидите презентацию корневого класса активного объекта запущенного эксперимента. Для каждой модели, созданной в **«Библиотеке моделирования процессов»**, автоматически создается блок-схема с наглядной визуализацией процесса, с помощью которой можно изучать текущее состояние модели, например, длину очереди, количество обработанных запросов и так далее (рис. 5.2.8). На рисунке показано сколько на каждом узле входящих заявок (запросов к серверу), сколько обработано, сколько стоит в очереди, сколько запросов было полностью отработано.

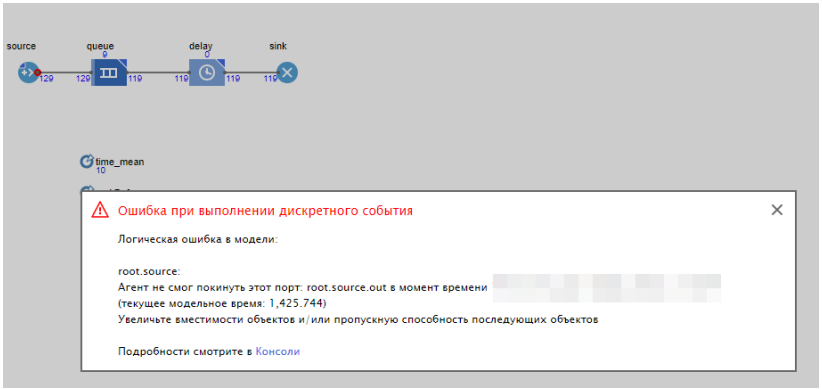


Рис. 5.2.10. Ошибка при выполнении дискретного события

14. Если емкость буфера будет недостаточной (для данных параметров установить значение emkBuf равным 10), то может возникнуть ошибка, показанная на рисунке 5.2.10.

Для устранения ошибки измените свойства параметра **«emkBuf»**, т.е. увеличьте максимальную вместимость очереди. Можно задать значение 50–60 и т.д. Ошибка, возможно, снова появится, так как зависит от длительности времени моделирования. Но это не критично.

Снова запустите модель.

15. Вы можете следить за состоянием любого объекта диаграммы процесса во время выполнения модели с помощью окна проверки этого объекта. Чтобы открыть окно проверки, щёлкните мышью по значку нужного блока. Окно проверки, подведя курсор, можно перемещать в нужное вам место. Также, подведя курсор к правому нижнему углу окна проверки, можно при необходимости изменять его размеры. В окне проверки будет отображена базовая информация по выделенному объекту: например, для объекта **queue** будут отображены вместимость очереди, количество заявок, прошедшее через каждый порт объекта и т.д. Такая же информация содержится в окне проверки и для объекта **delay** (рис. 5.2.11).

16. Если нужно остановить выполнение модели, то можно щёлкнуть мышью кнопку **«Прекратить выполнение эксперимента»**  панели управления окна презентации.

17. Для предотвращения остановок модели по ранее указанной ошибке – недостаточной ёмкости объекта **«Queue»** – мы увеличили ёмкость объекта **«Queue»**. Однако можно было бы изменять среднее время имитации поступления запросов объектом **«Source»** и среднее время обработки запросов сервером, т.е. среднее время задержки объекта **«Delay»**, оставляя неизменной длину очереди и добиваясь безошибочной работы модели. Конечно, при изменении свойств объектов модели нужно обязательно исходить из целей её построения.

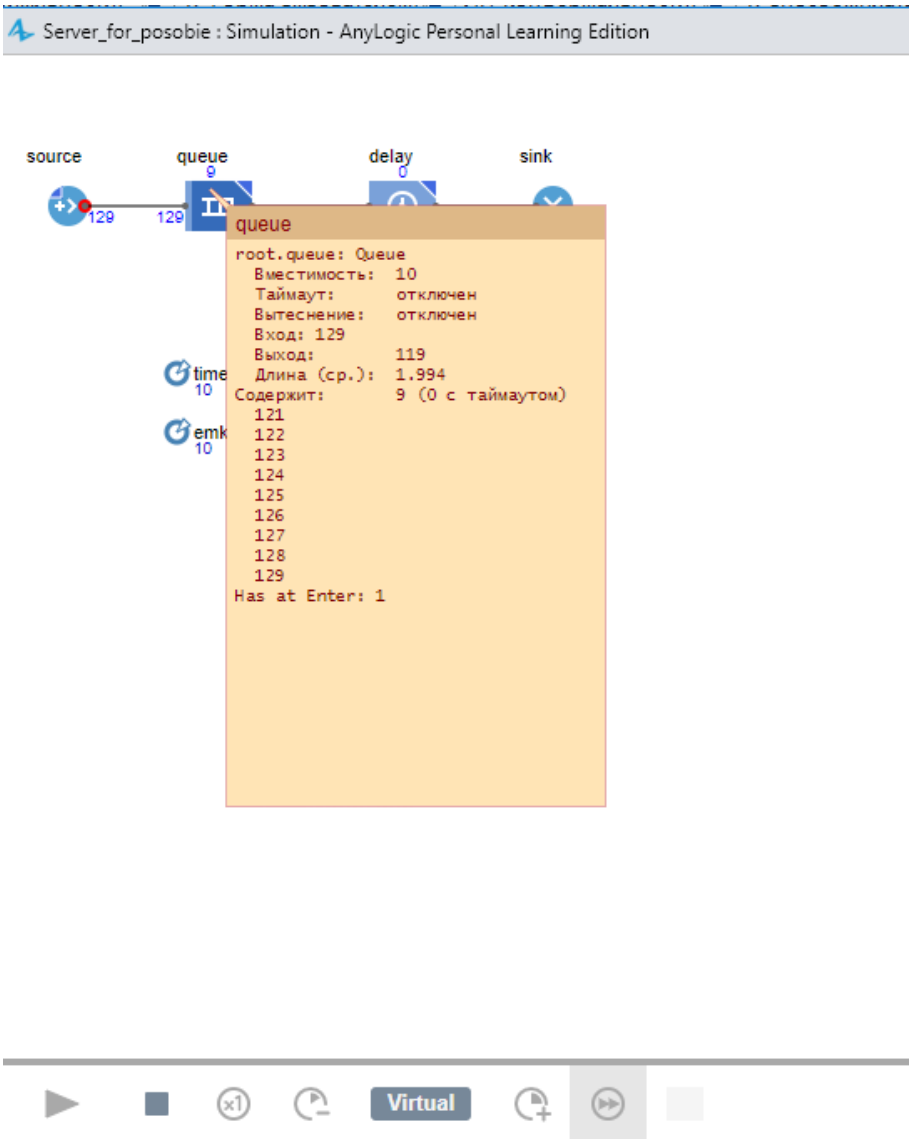


Рис. 5.2.11. Окно проверки для объекта queue

Анимация результатов моделирования

Можно ограничиться анализом и интерпретацией уже запущенной модели, но в ряде случаев этого недостаточно. AnyLogic позволяет построить более наглядную визуализацию с помощью анимации. Для данной задачи визуализируем процесс поступления запросов на сервер и обработку запросов сервером.

В данном случае нас не интересует конкретное расположение объектов в пространстве, поэтому мы можем просто добавить схематическую анимацию интересующих нас объектов – **сервер** и **очередь запросов к нему**. Анимация модели рисуется в той же диаграмме (в графическом ре-

дакторе), в которой задается и диаграмма моделируемого процесса. Для анимации будем использовать объекты палитры **«Презентация»**.

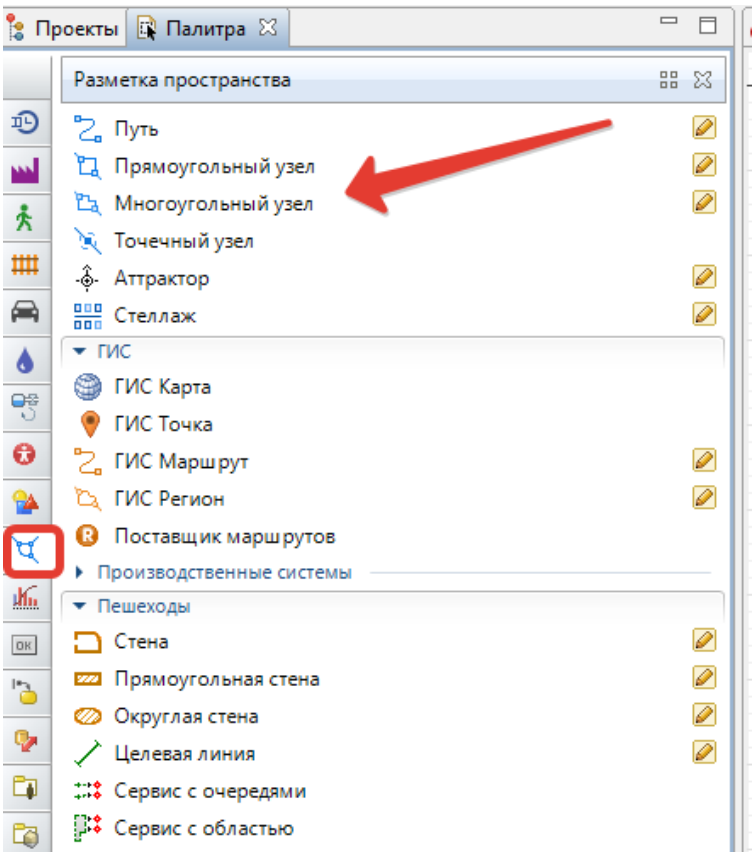


Рис. 5.2.12. *Палитра «Разметка пространства»*

1. Нарисуйте прямоугольный узел, который будет обозначать на анимации **сервер**. Для этого откройте палитру **«Разметка пространства»** (рис. 5.2.12).

2. Палитра **«Презентация»** содержит в качестве элементов различные фигуры, используемые для рисования презентаций моделей. Это **«Путь», «Прямоугольный узел», «Многоугольный узел», «Точечный узел», «Аттрактор»** и др.

3. Выделите элемент **«Прямоугольный узел»** и перетащите его на диаграмму класса активного объекта. Поместите элемент **«Прямоугольный узел»** так, как показано на рис. 5.2.13.

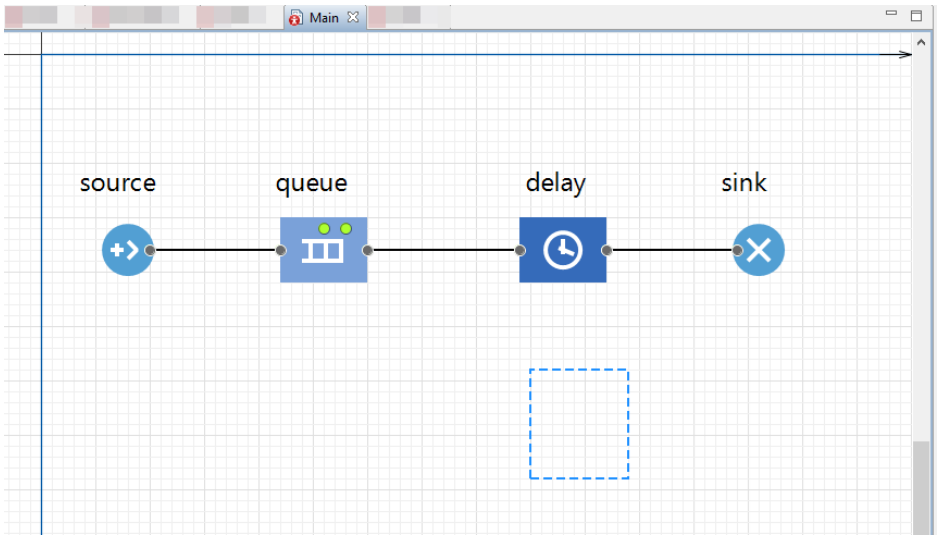


Рис. 5.2.13. Элемент «Прямоугольный узел» на диаграмме

4. Настроим работу **«Прямоугольного узла»** так, чтобы его цвет менялся в зависимости от того, обрабатывает ли сервер в данный момент времени запрос или нет.

Для этого выделите нарисованную **«Прямоугольный узел»** на диаграмме. Перейдите на страницу **«Внешний вид»** панели свойств (рис. 5.2.14).

Если нужно, чтобы по ходу моделирования то или иное свойство фигуры меняло своё значение в зависимости от каких-то условий, то можно ввести в поле соответствующего динамического свойства выражение, которое будет постоянно вычисляться заново при выполнении модели. Возвращаемый результат вычисления будет присваиваться текущему значению этого свойства.

Пусть во время моделирования цвет нашей фигуры будет меняться, поэтому щёлкните в поле **«Цвет заливки»**: по стрелке, выберите **«Динамическое значение»** и введите там следующую строку:

delay.size()>0?red:green

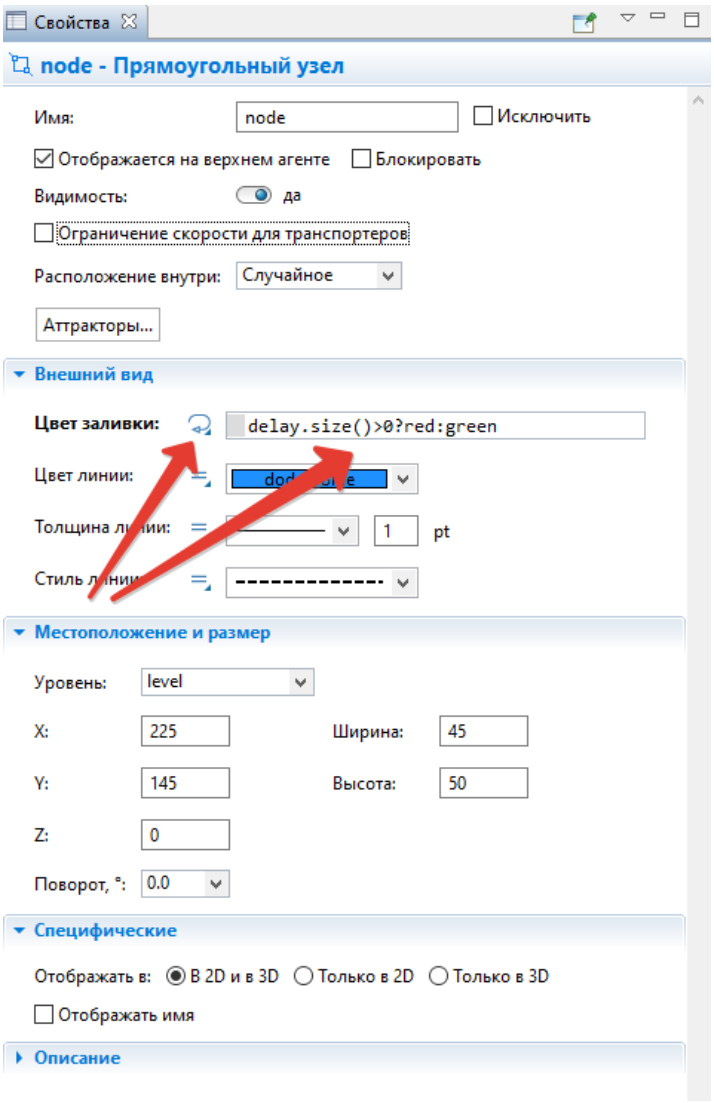


Рис. 5.2.14. Динамическое свойство для установки цвета заливки

Здесь **«Delay»** — это имя нашего объекта **«Delay»**. Функция **size()** возвращает число запросов, обслуживаемых в данный момент времени. Если сервер занят, то цвет кружка будет **красным**, в противном случае – **зелёным**.

5. Нарисуйте путь, который будет обозначать на анимации очередь к серверу (рис. 5.2.15). Чтобы нарисовать путь, сделайте двойной щелчок мышью по элементу **«Путь»** палитры **«Разметка пространства»**, чтобы перейти в режим рисования. Теперь вы можете рисовать путь точку за точкой, последовательно щелкая мышью в тех точках диаграммы, куда вы хотите поместить вершины пути. Чтобы завершить рисование, добавьте последнюю точку пути двойным щелчком мыши.

Очень важно, какую точку пути вы создаете первой. Заявки будут располагаться вдоль нарисованного вами пути в направлении от конечной точки к начальной точке. Поэтому обязательно начните рисование пути слева и поместите рядом с сервером конечную точку пути, которая будет соответствовать в этом случае началу очереди.

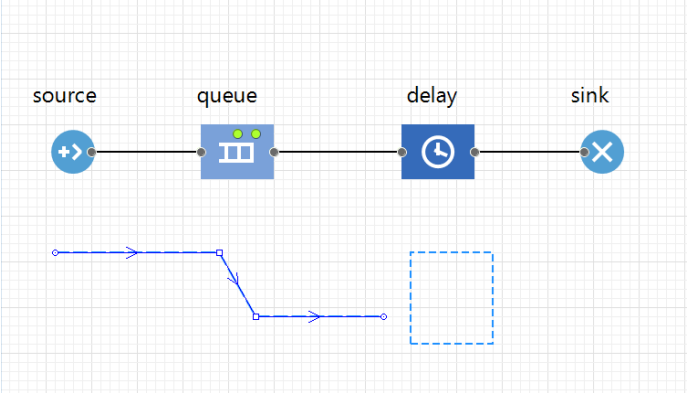


Рис. 5.2.15. Путь на диаграмме процесса

6. Зададим созданные анимационные объекты в качестве анимационных фигур объектов диаграммы процесса.

Задайте путь в качестве фигуры анимации очереди. Выделите объект **queue**. На странице свойств объекта **queue** в поле **«Место агентов»** выберите из выпадающего списка **«path»** (рисунок 5.2.16).

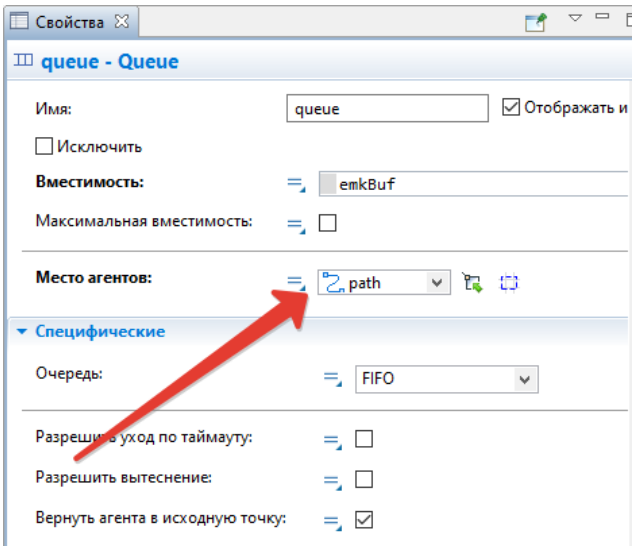


Рис. 5.2.16. Задание пути в качестве фигуры анимации очереди

Задайте прямоугольный узел в качестве фигуры анимации сервера. Выделите объект **«Delay»**. Введите в поле **«Место агентов»** из выпадающего списка имя прямоугольного узла **«node»** (рис. 5.2.17).

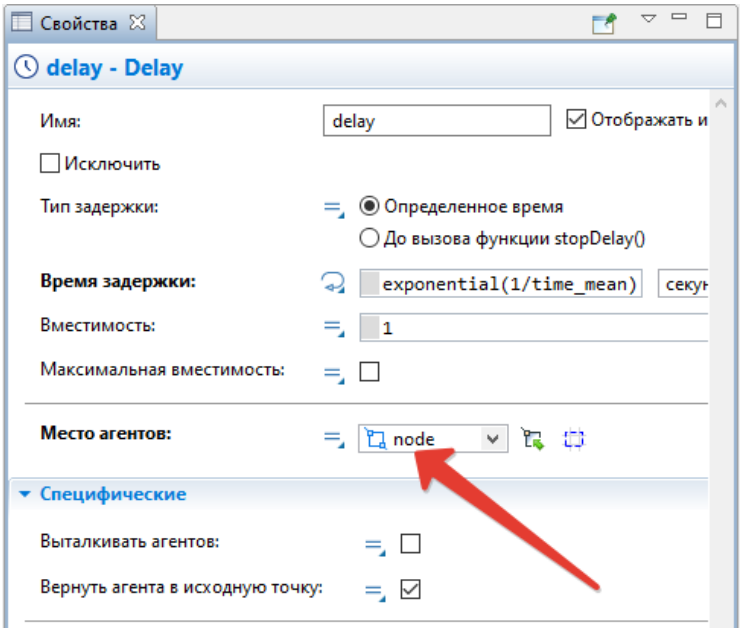


Рис. 5.2.17. Задание прямоугольного узла в качестве фигуры анимации сервера

7. Запустим модель. Можно увидеть, что у модели теперь есть простейшая анимация — сервер и очередь запросов к нему (рис. 5.2.18). Цвет фигуры сервера будет меняться в зависимости от того, обрабатывается ли запрос в данный момент времени или нет.

Если в процессе запуска модели у вас все же появляется логическая ошибка, то можно изменить настройки симуляции. Выберите в дереве проекта узел **«Simulation: Main»**. В окне **«Свойств»** на вкладке **«Модельное время»** установите **«Режим выполнения»** в значение **«Реальное время со скоростью»** 10. Запустите модель.

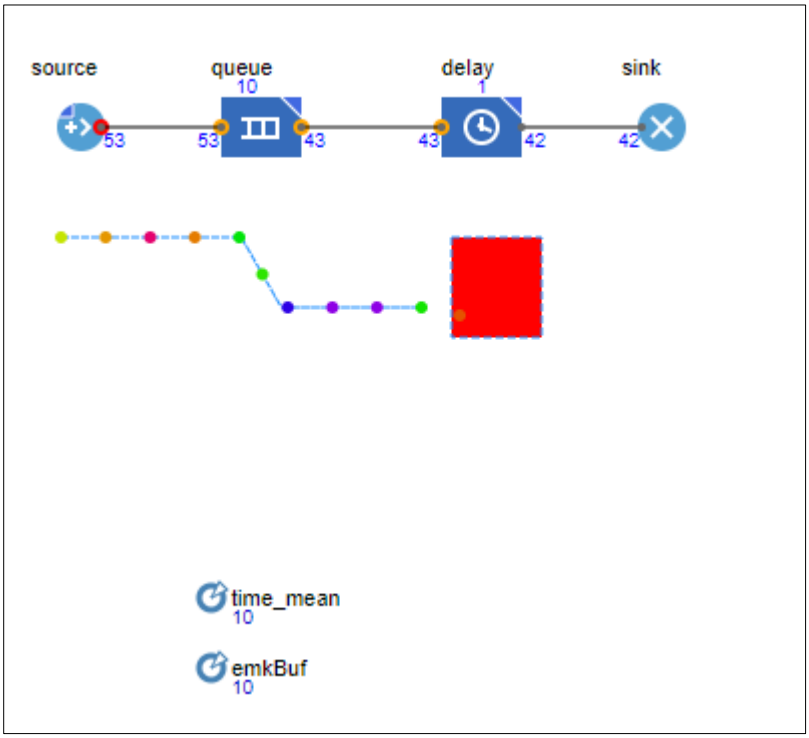


Рис. 5.2.18. Анимация модели

Сбор статистики использования ресурсов

AnyLogic предоставляет пользователю удобные средства для сбора статистики по работе блоков диаграммы процесса. Объекты **«Enterprise Library»** самостоятельно производят сбор основной статистики. Все, что нужно сделать — это включить сбор статистики для объекта. Мы уже выставили данный параметр для объектов **«Delay»** и **«Queue»**, поэтому теперь можно, например, просмотреть интересующую нас статистику (статистику занятости сервера и длины очереди) с помощью диаграмм.

Добавьте диаграмму для отображения среднего коэффициента использования сервера:

1. Откройте палитру **«Статистика»**. Эта палитра содержит элементы сбора данных и статистики, а также диаграммы для визуализации данных и результатов моделирования.

2. Перетащите элемент **«Столбиковая диаграмма»** из палитры **«Статистика»** на диаграмму класса и измените ее размер (рис. *5.2.*18).

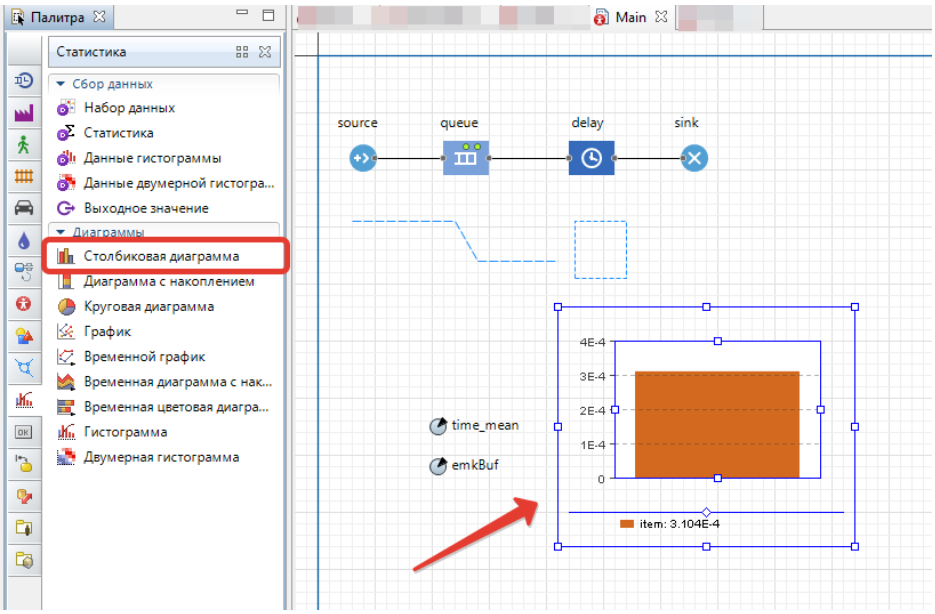


Рис. 5.2.19. Элемент «Столбиковая диаграмма» на диаграмме класса

3. Перейдите на панель *«Свойства»*. Щёлкните кнопку **«Добавить элемент данных»**. После щелчка появится секция свойств того элемента данных (chart – **«Столбиковая диаграмма»**), который будет отображаться на этой диаграмме (рис. 5.2.20).

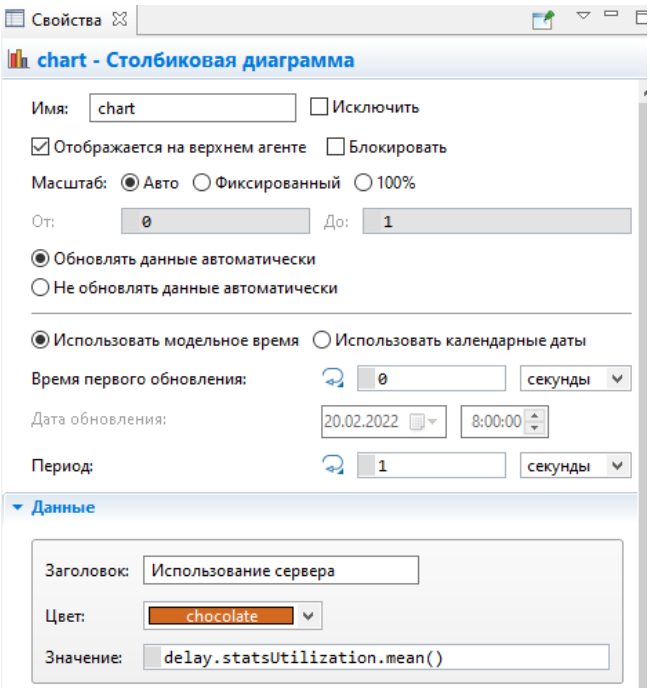


*Рис. 5.2.20. Страница **«Свойства»***

4. Измените Заголовок на **Использование сервера**.

5. Введите delay.statsUtilization.mean() в поле **Значение**. Здесь delay – это имя нашего объекта delay. У каждого объекта delay есть встроенный набор данных statsUtilization, занимающийся сбором статистики использования этого объекта. Функция mean() возвращает среднее из всех измеренных этим набором данных значений. Вы можете использовать и другие методы сбора статистики, такие как **min()** или **max()**.

6. Щёлкните **«Внешний вид»** (рис. 5.2.21). Установите свойства: **направление столбцов, цвета фона, границ, меток, сетки, положение подписей у столбцов**

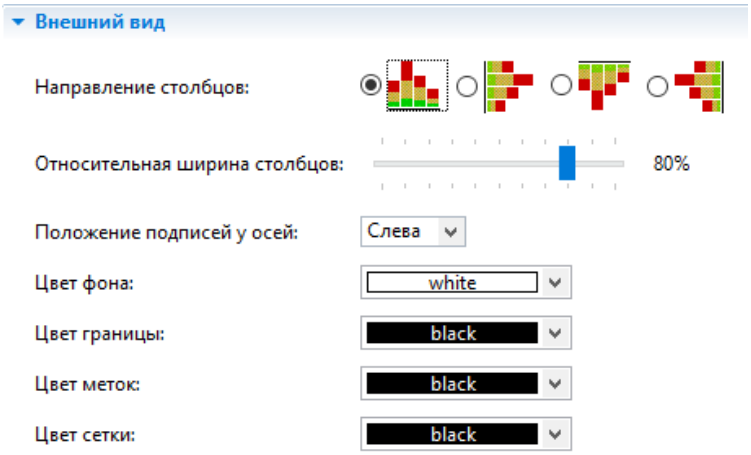


*Рис. 5.2.21. Вкладка «**Внешний вид**»*

7. Раскройте щелчками страницы (вкладки) Местоположение и размер, Легенда, Область диаграммы (рис. 21). Установите свойства, чтобы изменить расположение легенды относительно диаграммы (она будет внизу), размер диаграммы, высоту, ширину, координаты размещения на диаграмме, цвета текста, границы.

8. Аналогичным образом добавьте еще одну столбиковую диаграмму для отображения средней длины очереди. На панели **«Свойства»** щёлкните **«Добавить элемент данных»**. После щелчка появится страница **«Данные свойств элемента данных»** (chart1 – **«Столбиковая диаграмма»**), который также будет отображаться на этой диаграмме.

Заголовок сделайте **«Длина очереди»**, значение **queue.statsSize.mean()**.

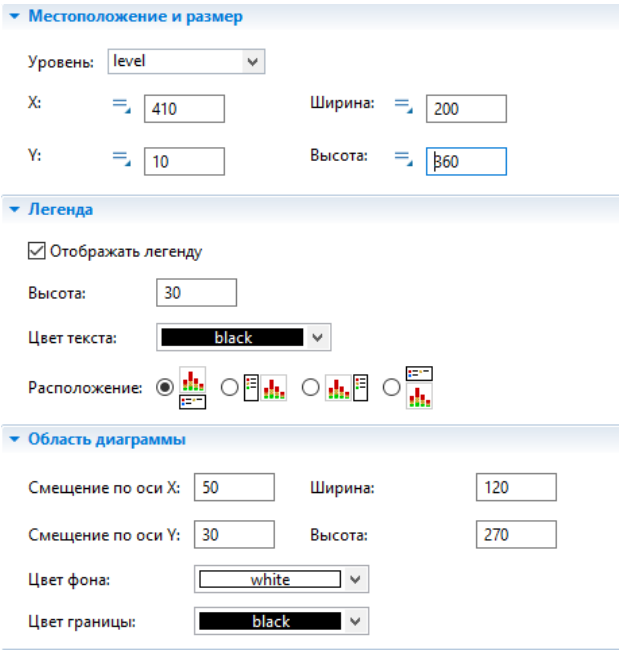


Рис. 5.2.22. Вкладки «Местоположение и размер», «Легенда», «Область диаграммы»

На страницах **«Внешний вид»**, **«Местоположение и размер»**, **«Легенда»**, **«Область диаграммы»** установите свойства самостоятельно. Столбцы диаграммы должны размещаться горизонтально (рис. 5.2.23).

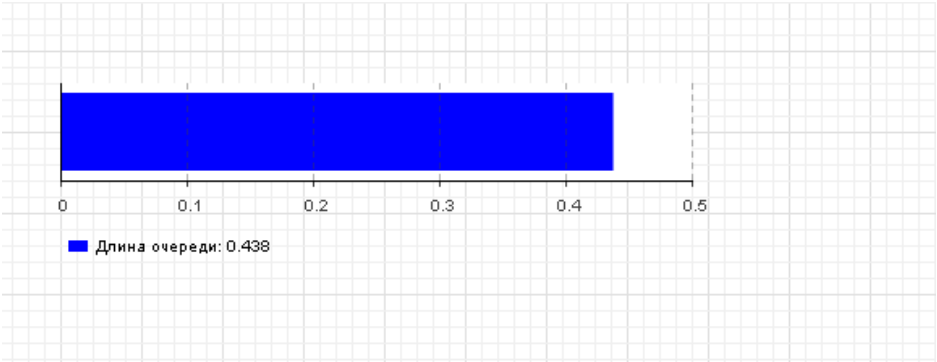


Рис. 5.2.23. Столбиковая диаграмма для отображения длины очереди

9. Запустите модель с двумя столбиковыми диаграммами, установив модельное время 3 600 единиц, и понаблюдайте за её работой (рис. 5.2.24).

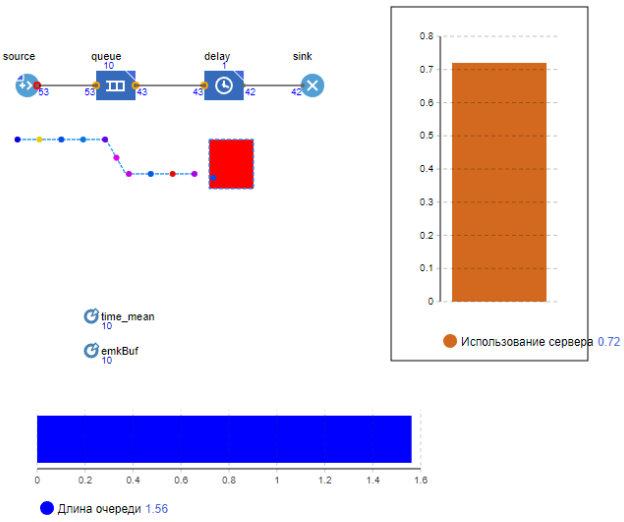


Рис. 5.2.24 Наблюдение за моделью с двумя столбиковыми диаграммами

Уточнение модели согласно ёмкости входного буфера

Объект **«Queue»** моделирует очередь заявок, ожидающих приёма объектами, следующими за ним в потоковой диаграмме, или же моделирует хранилище заявок общего назначения. При необходимости можно задать максимальное время ожидания заявки в очереди. Можно с помощью написанной вами программы извлекать заявки из любых позиций в очереди. Заявка может покинуть объект **«Queue»** различными способами: обычным способом через порт **out**, когда объект, следующий в блок-схеме за этим объектом, готов принять заявку; через порт **outTimeout**, если заявка проведет в очереди заданное количество времени (если включен режим **таймаута**); через порт **outPreempted**, будучи вытесненной другой поступившей заявкой при заполненной очереди (если включен режим вытеснения); «вручную», путем вызова функции **remove()** или **removeFirst()**. В первом случае объект **«Queue»** покидает заявка, находящаяся в самом начале очереди (в нулевой позиции).

Если заявка направлена в порт **outTimeout** или **outPreempted**, то она должна покинуть объект мгновенно. Если включена опция вытеснения, то объект **«Queue»** всегда готов принять новую заявку, в противном случае при заполненной очереди заявка принята не будет. Поступающие заявки помещаются в очередь в определенном порядке: либо согласно правилу

FIFO (в порядке поступления в очередь), либо согласно приоритетам заявок. Приоритет может быть либо явно храниться в заявке, либо вычисляться согласно свойствам заявки и каким-то внешним условиям. Очередь с приоритетами всегда примет новую входящую заявку, вычислит её приоритет и поместит в очередь в позицию, соответствующую её приоритету. Если очередь будет заполнена, то приход новой заявки вынудит последнюю хранящуюся в очереди заявку покинуть объект через порт **outPreempted**. Но если приоритет новой заявки не будет превышать приоритет последней заявки, то тогда вместо неё будет вытеснена именно эта новая заявка. Для выполнения условия постановки задачи воспользуемся последним способом вытеснения. Все запросы, вырабатываемые объектом **«Source»**, имеют один и тот же приоритет. Поэтому при полном заполнении накопителя (5 запросов) теряться будет последний запрос.

Изменим параметры очереди

1. Выделите объект **emkBuf**. Измените значение с 10 на 5 запросов.
2. У объекта **«Queue»** установите **«Разрешить вытеснение»**.

3. Для уничтожения потерянных запросов вследствие полного заполнения накопителя нужно добавить второй объект **«Sink»**. Откройте в Палитре **Библиотеку моделирования процессов** и перетащите блок **«Sink»** на диаграмму (рис. 5.2.25).

4. Запустите уточненную модель и понаблюдайте за ее работой. Сравните рис. 5.2.26 с рис. 5.2.23. На рис. 5.2.25 видно, что запросы при длине очереди в 10 запросов теряются, и ошибки при этом не возникает. Модель по ограничению ёмкости входного буфера и значениям других параметров соответствует постановке задачи. Однако согласно постановке задачи, требуется определить математическое ожидание времени обработки одного запроса и математическое ожидание вероятности обработки запросов.

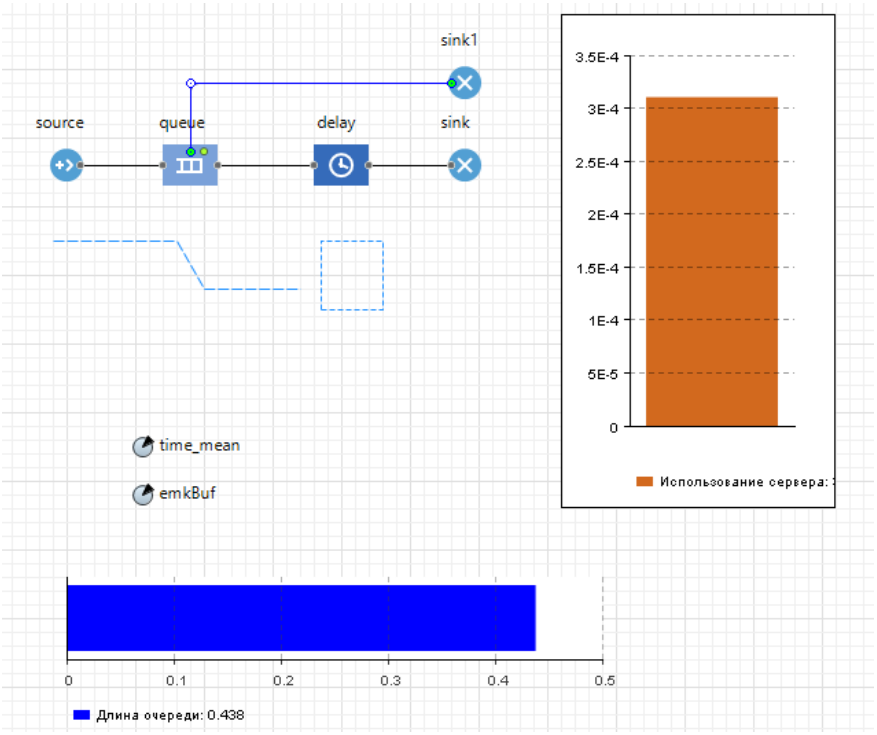


Рис. 5.2.25. Измененная модель

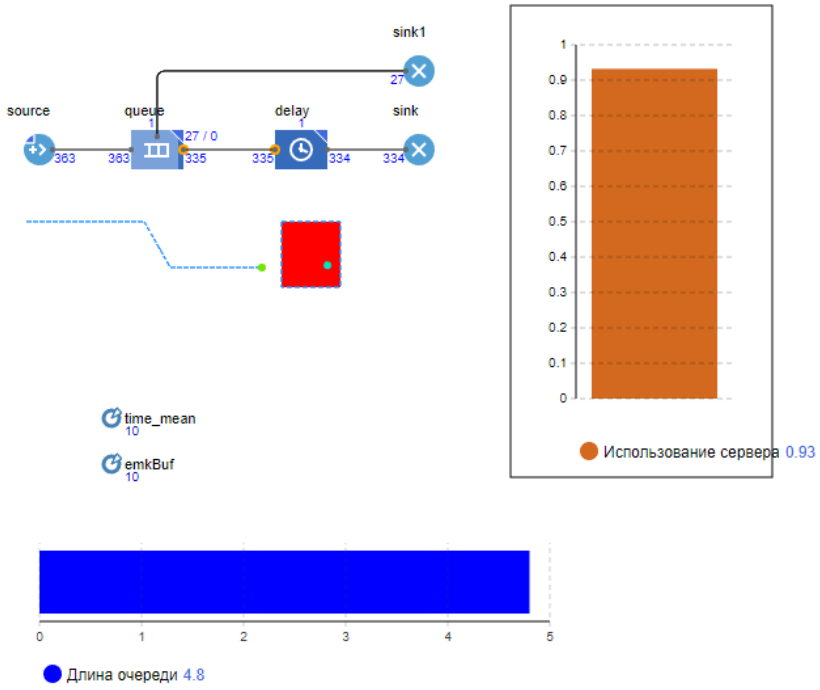


Рис. 5.2.26. Измененная модель в запущенном состоянии

Сбор статистики по показателям обработки запросов

«Entity» (заявка) является базовым классом для всех заявок, которые создаются и работают с ресурсами в процессе, описанном вами с помощью диаграммы из объектов **«Библиотеки моделирования процессов»**. **«Entity»** по существу является обычным Java-классом с теми функциональными возможностями, которые необходимы и достаточны для обработки и отображения анимации заявки объектами **«Библиотеки моделирования процессов»**. Эти функциональные возможности можно расширить добавлением дополнительных полей и методов и работой с ними из объектов диаграммы, описывающей моделируемый процесс. Согласно постановке задачи, нужно определять математическое ожидание времени и вероятности обработки запросов сервером.

Математическое ожидание или среднее время обработки одного запроса определяется как отношение суммарного времени обработки n запросов к их количеству, т.е. к n. Для определения суммарного времени нужно знать время обработки i-го запроса. Для этого введем дополнительные поля:

time_vhod – время входа запроса в буфер сервера;

time_vihod – время выхода запроса с сервера (входа в блок sink).

Тогда **time_obrabotki=time_vihod-time_vhod.** Вероятность обработки запросов сервером определяется как отношение количества обработанных запросов к количеству всех поступивших запросов. Значит, нужно вести счет запросов на выходе источника запросов и на выходе с сервера (входе в блок **sink**). Для этого также введем дополнительные поля:

col_vhod – количество поступивших всего запросов;

col_vihod – количество обработанных сервером запросов.

Тогда **ver_obrabotki=col_vihod/col_vhod.**

Замечание. Ничего необычного во введённых дополнительных полях нет. Это параметры реальных элементов потоков, в данном случае запросов. AnyLogic предоставляет возможность создавать запросы с теми параметрами, которые необходимы в модели.

Для включения в запросы дополнительных полей необходимо создать нестандартный тип заявки.

1. Откройте палитру **«Библиотека моделирования процессов»**. Перетащите элемент **«Тип агента»** в графический редактор. Появится тип заявки **«Entity»** (рис. 5.2.27).

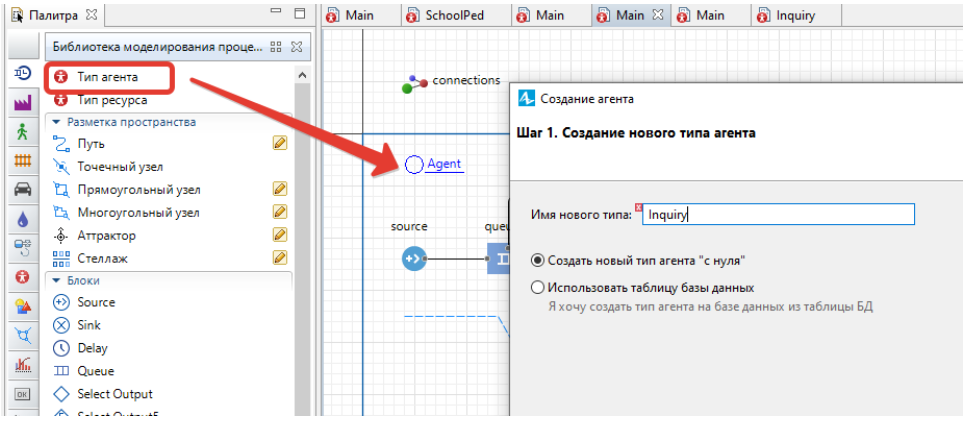


Рис. 5.2.27. Создание нового агента

Появится диалоговое окно **«Создание агента»** (рис. 5.2.27).

Шаг 1. Создание нового типа агента. В поле **«Имя нового типа»**: введите **Inquiry**.

Шаг 2. Выберите анимацию агента: установите 2D и выберите из выпадающего списка, например, **«Сообщение»**. Щёлкните **«Далее»** (рис. 5.2.28).

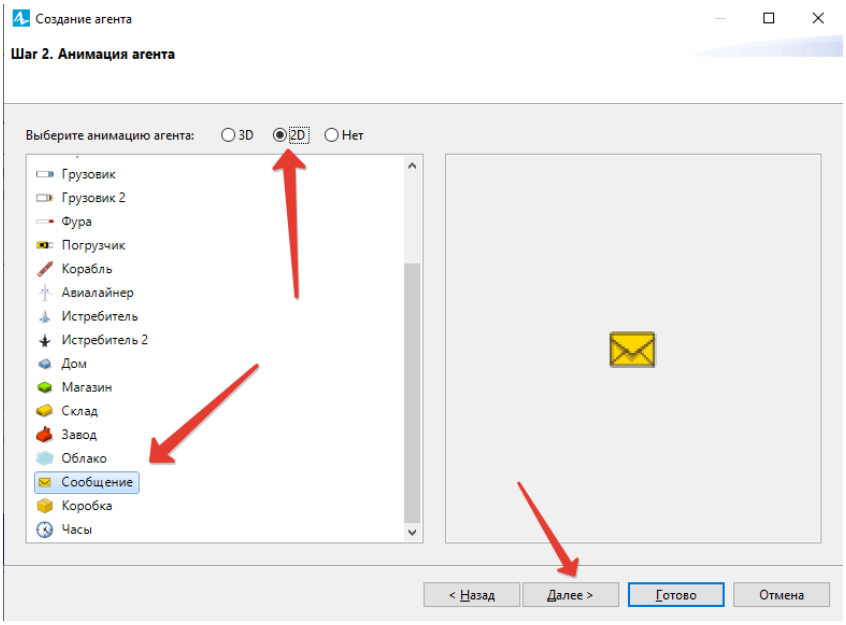


Рис. 5.2.28. Настройка анимации нового агента

Шаг 3. Параметры агента. Щёлкните **<добавить… >** (рис. 5.2.29). В поле Параметр введите:

- **time_vhod**, из выпадающего списка **«Тип»** выберите **double;**
- **time_vihod**, из выпадающего списка **«Тип»** выберите **double;**
- **col_vhod**, из выпадающего списка **«Тип»** выберите **double**;
- **col_vihod**, из выпадающего списка **«Тип»** выберите **double**;

Так как в поле **«Значение по умолчанию»** мы не устанавливали никаких значений, то всем параметрам будет установлен 0.

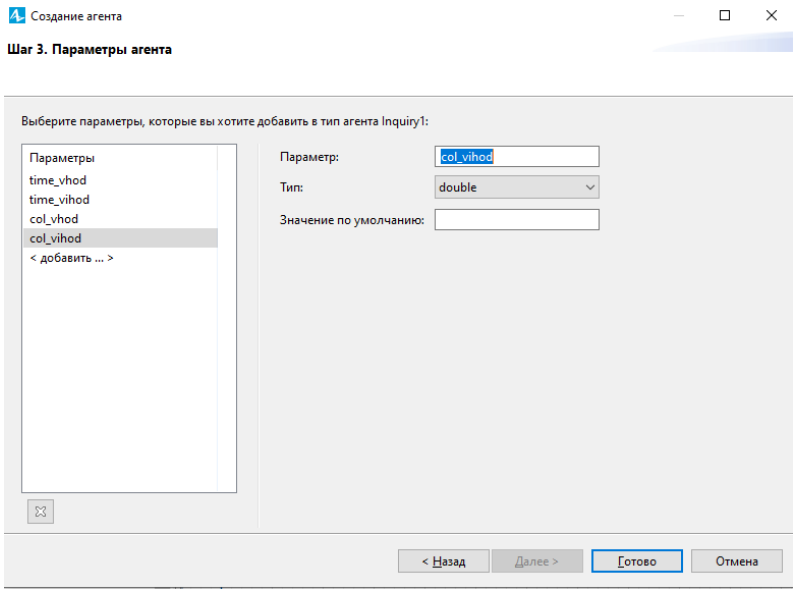


Рис. 5.2.29. Добавление параметров агента

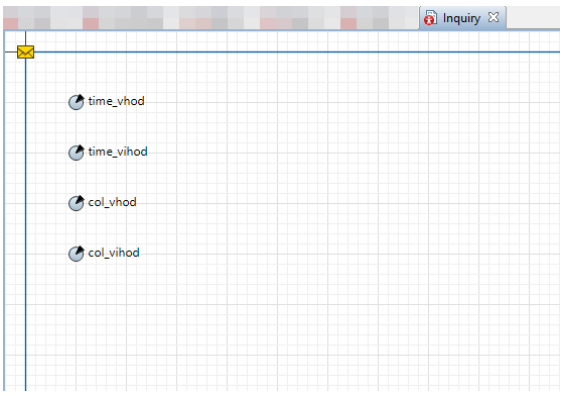


Рис. 5.2.30. Окно с параметрами нестандартного типа заявки Inquiry

2. Щёлкните кнопку **«Готово»**. Вы увидите окно, в котором будут показаны автоматически созданные параметры нестандартного типа заявок

Inquiry (рис. 5.2.30). Закройте окно, щелкнув крестик в закладке рядом с его названием.

3. Для сбора статистических данных о времени обработки запросов сервером необходимо добавить элемент статистики. Этот элемент будет запоминать значения времен для каждого запроса. На основе этого он предоставит пользователю стандартную статистическую информацию (среднее, минимальное, максимальное из измеренных значений, среднеквадратичное отклонение и т.д.).

Чтобы добавить элемент сбора данных гистограммы на диаграмму, перетащите элемент **«Данные гистограммы»** с палитры **«Статистика»** на диаграмму активного класса.

Задайте свойства элемента (рис. 5.2.31): измените **«Имя»** на **time_obrabotki**; сделайте **«Кол-во интервалов»** равным 50; задайте **«Нач. размер интервала»** 0,01.

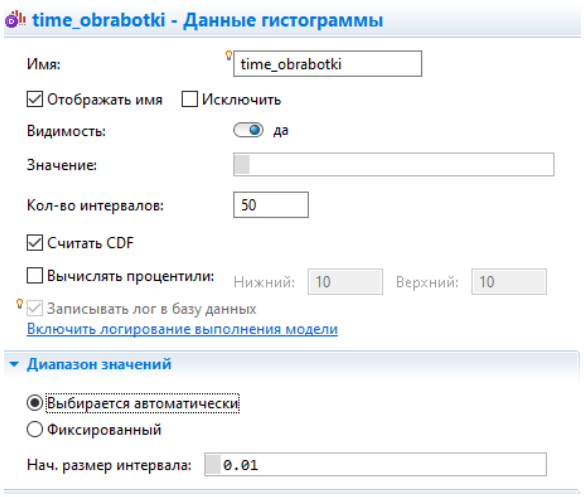


Рис. 5.2.31. Элемент сбора статистики о времени обработки запросов

4. Добавьте еще элемент сбора статистики для определения вероятности обработки запросов.

Перетащите элемент **«Данные гистограммы»** с палитры **«Статистика»** на диаграмму активного класса.

Задайте свойства элемента (рис. 5.2.32): измените **«Имя»** на **ver_obrabotki**; сделайте кол-во интервалов равным 50; задайте **«Нач. раз-**

мер интервала» 0,01. Диаграмма после добавления элементов сбора статистики представлена на рис. 5.2.32.

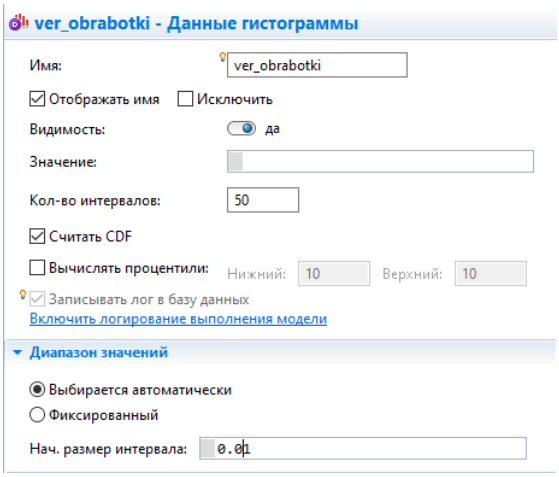


Рис. 5.2.32. Элемент сбора статистики о вероятности обработки запросов

5. Чтобы создавать заявки нестандартного типа, как в нашем случае **Inquiry**, нужно поместить вызов конструктора этого типа в поле **«Новая заявка»** объекта **«Source»**. Но, несмотря на то, что заявки в потоке теперь и будут типа **Inquiry**, остальные объекты диаграммы будут продолжать их считать заявками типа **«Агент»**. Поэтому они не позволят явно обращаться к дополнительным полям класса **Inquiry**. Чтобы разрешить доступ к полям вашего нестандартного типа заявки в коде динамических параметров объектов потоковой диаграммы, нужно указать имя нестандартного типа заявки в качестве Типа агента этого объекта. В нашей потоковой диаграмме с учётом блока **«Source»** всего пять объектов.

Для объектов **«source»**, **«queue»**, **«delay»**, **«sink»**, **«sink1»** установите в качестве поля **«Новый агент»** значение **Inquiry** (рис. 5.2.33).

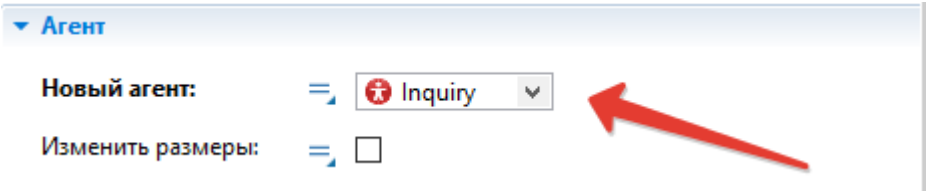


Рис. 5.2.33. Изменение поля Новый агент

6. Для объекта **«source»** в окне **«Свойства»** задайте в поле **«Действие при выходе»** равным **agent.time_vhod=time()**;

Код будет сохранять время создания заявки-запроса в параметре **time_vhod** нашего типа заявки **Inquiry**. Функция **time()** возвращает текущее значение модельного времени.

7. Для объекта **«Sink»** на вкладке **«Действие»** в поле **«При входе»** введите **time_obrabotki.add(time()-agent.time_vhod)**.

Этот код добавляет время обработки одного запроса в объект сбора данных гистограммы **time_obrabotki**. Данное время определяется как разность между текущим модельным временем **time()** и временем входа запроса в модель. **add** — встроенная функция добавления элемента в массив.

agent.col_vihod=sink.count();

agent.col_vhod=source.count().

Эти коды заносят количество запросов, вошедших в блок **«Sink»** и вышедших из блока **«Source»** соответственно. **count()** – встроенная функция этих блоков, возвращает количество вошедших в блок **«Sink»** и количество вышедших из блока **«Source»** заявок.

ver_obrabotki.add(entity.col_vihod/entity.col_vhod);

Этот код добавляет относительную долю обработанных запросов в объект сбора данных гистограммы **ver_obrabotki** при поступлении каждого обработанного запроса в блок **«Sink»**. На основе множества таких относительных долей определяется математическое ожидание вероятности обработки запросов сервером.

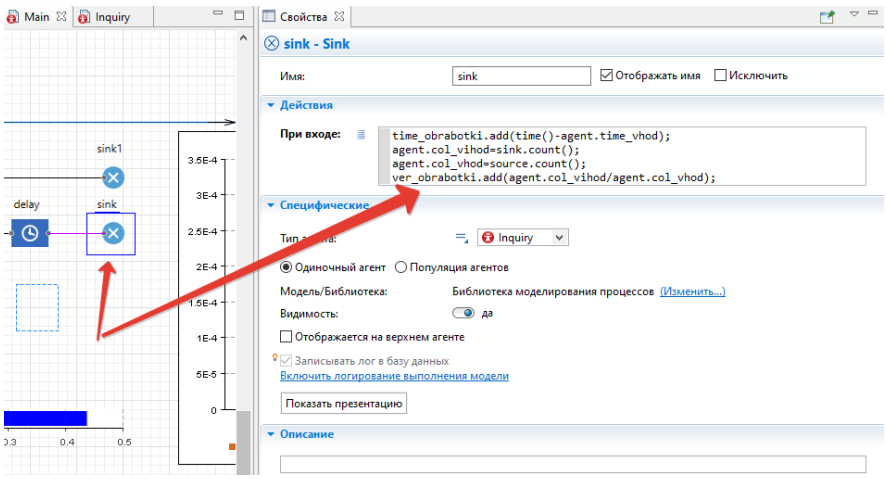


Рис. 5.2.34. Обработка действия «При входе» объекта sink

8. Запустите модель (рис. 5.2.35).

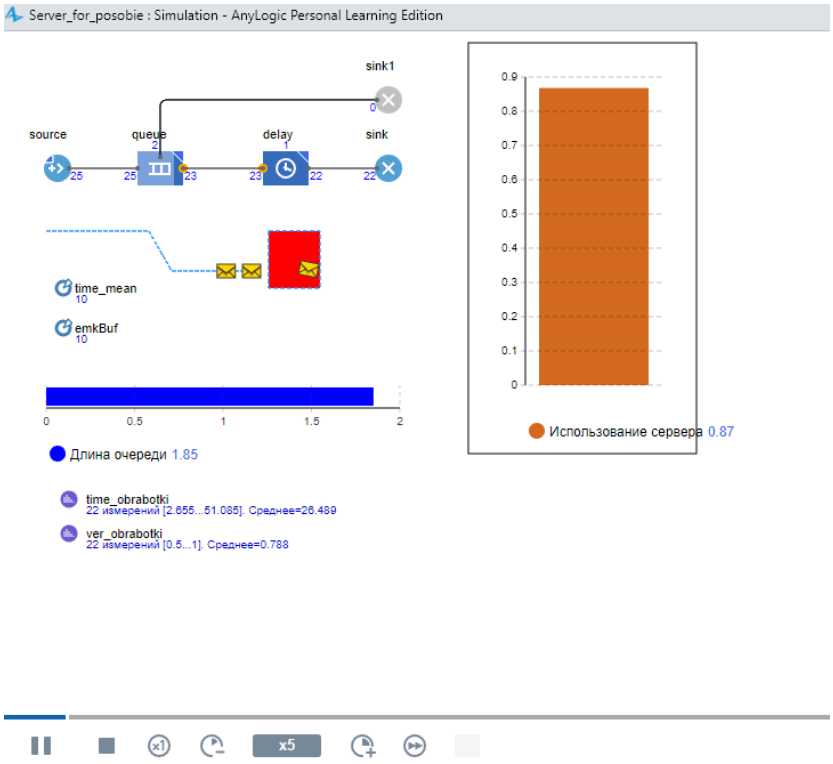


Рис. 5.2.35. Модель сервера в режиме запуска

Добавление параметров и элементов управления

Пусть вы хотите изменять среднее время обработки запросов **time_mean** в ходе моделирования. Используйте для этого элемент управления – бегунок.

1. Откройте палитру **«Элементы управления»** и перетащите элемент **«Бегунок»** из палитры на диаграмму класса **«Main»** (рис. 5.2.35).

2. Поместите бегунок под параметром **time_mean**, чтобы было понятно, что с помощью этого бегунка будет меняться среднее время обработки запросов объектом **«Delay»**.

3. Пусть вы хотите варьировать среднее время от 1 до 300. Поэтому введите 1 в поле **«Минимальное значение»**, а 300 – в поле **«Максимальное значение»** (рис. 5.2.36).

4. Установите флажок **«Связать с»** и в активизированное поле введите **time_mean**.

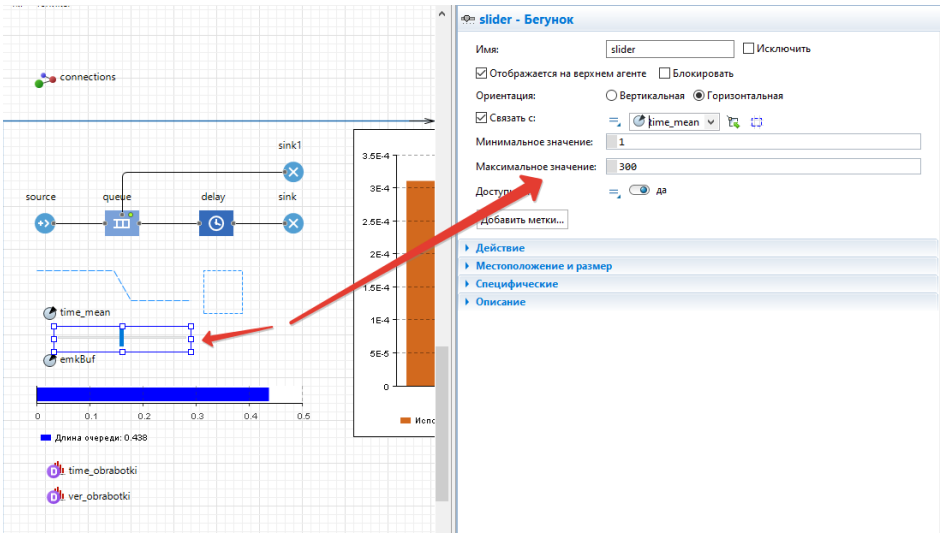


Рис. 5.2.36. Установка бегунка для параметра time_mean

5. Пусть теперь вы хотите также изменять ёмкость буфера в ходе моделирования. Используйте для этого также бегунок. Откройте палитру **«Элементы управления»** и перетащите элемент Бегунок из палитры на диаграмму класса **«Main»** (рис. 5.2.37).

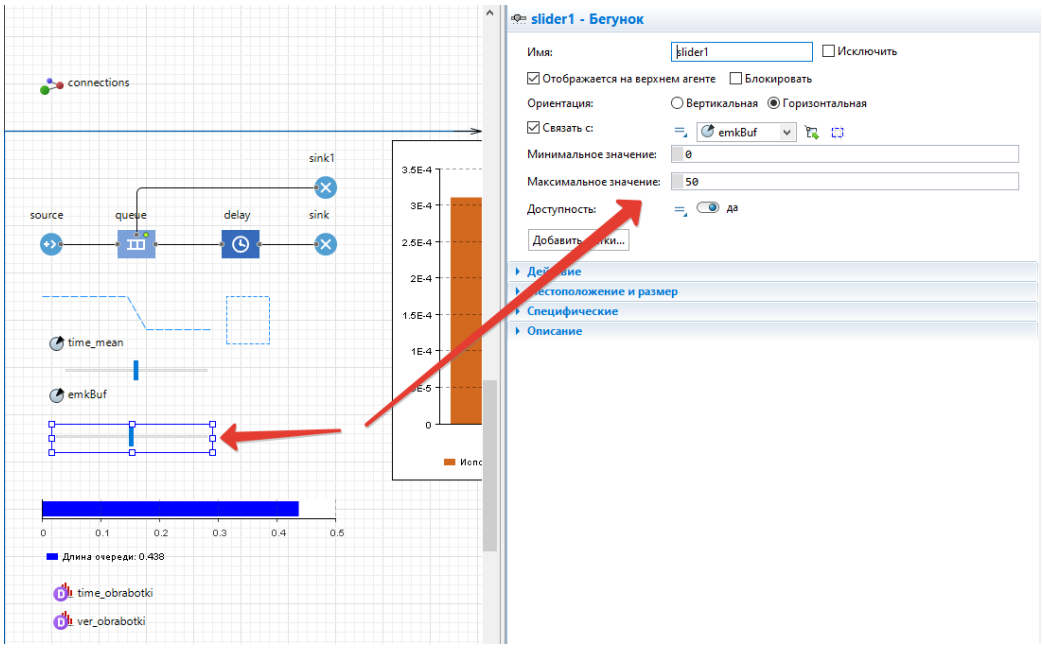


Рис. 5.2.37. Установка бегунка для параметра emkBuf

6. Запустите модель. Теперь вы можете изменять в процессе моделирования ёмкость входного буфера и среднее время обработки запросов с помощью бегунков. Можете также командой **Приостановить** приостано-

вить работу модели, изменить значения параметров, а затем продолжить моделирование.

Добавление гистограмм

1. Теперь добавим на диаграмму нашего потока гистограмму, которая будет отображать собранную временную статистику.

Перетащите элемент **«Гистограмма»** из палитры **«Статистика»** в то место графического редактора, куда хотите ее поместить.

Укажите, какой элемент сбора данных хранит данные, которые вы хотите отображать на гистограмме: щёлкните кнопку **«Добавить данные»** и введите в поле **Данные** имя соответствующего элемента: **time_obrabotki**.

Установите галочку в поле **«Отображать среднее».** В поле **«Заголовок»**: введите **Histogram Time obrabotki** (рис. 5.2.38).

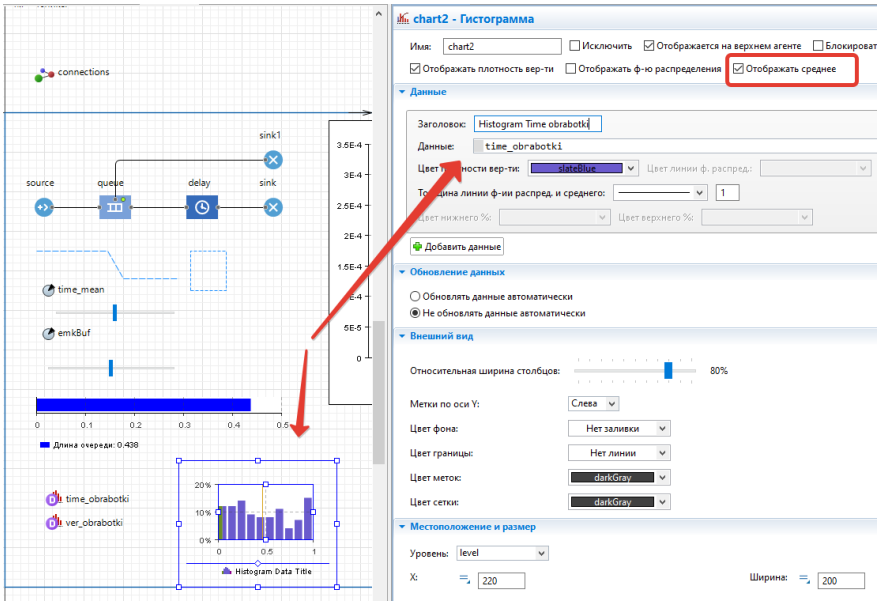


Рис. 5.2.38. Установка гистограммы для параметра time_obrabotki

2. Добавим на диаграмму нашего потока гистограмму, которая будет отображать собранную вероятностную статистику. Перетащите элемент **«Гистограмма»** из палитры **«Статистика»**.

Щёлкните кнопку **«Добавить данные»** и введите в поле **«Данные имя элемента»** значение **ver_obrabotki**. Установите **«Отображать среднее»**.

В поле **«Заголовок»** введите **Histogram Ver obrabotki**.

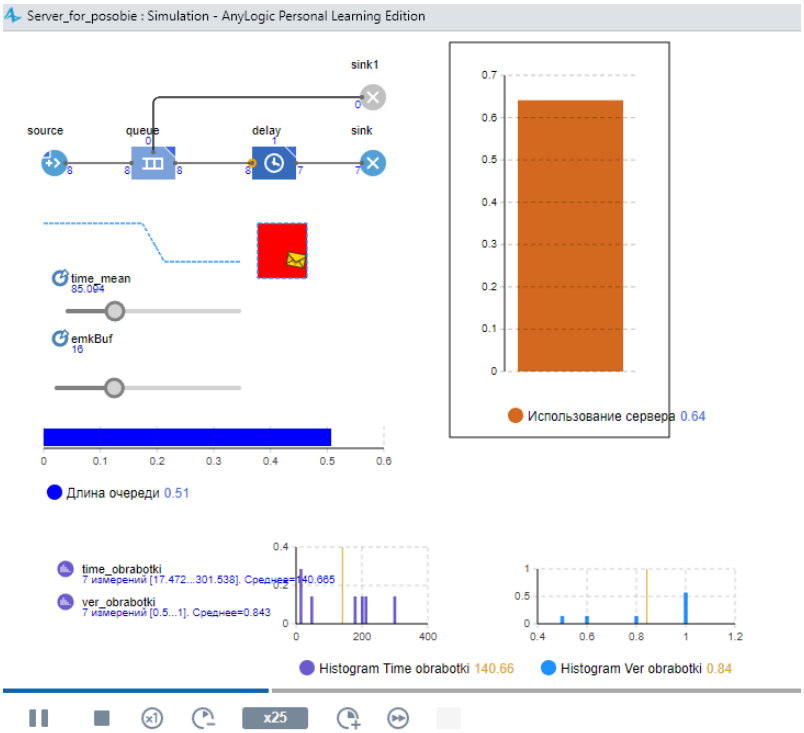


Рис. 5.2.39. Фрагмент работы модели с элементом управления и гистограммами

3. Запустите модель. Фрагмент работы показан на рис. 5.2.39.

Исследовательские задачи

1. При каких сочетаниях параметров модели получим значение вероятности обработки заявки (на уровне 0,90–0,95)?

2. При каких параметрах высоконагруженная система будет иметь отказы в обслуживании (емкость буфера, интенсивность поступления заявок)?

3. Как повлияет на среднюю длину очереди увеличение производительности сервера?

4. Как изменятся характеристики системы, если увеличить производительность сервера?

5. Если увеличить емкость входного буфера, то как изменится среднее время обработки запроса и средняя длина очереди?

6. Если увеличить интенсивность поступления запросов, то, как это повлияет на количество обработанных запросов?

7. Как изменятся характеристики системы, если увеличить производительность сервера?

5.3. Имитационная модель системы массового обслуживания

Постановка задачи. Рассмотрим задачу по созданию имитационной модели работы магазина. Современные супермаркеты породили большие проблемы, разрешить которые совсем не просто. Владелец магазина желает, чтобы вложенные в магазин деньги приносили максимум прибыли. А это означает, что оборудование и обслуживающий персонал должны работать эффективно, товар должен быстро продаваться и приносил прибыль. Как, например, узнать, сколько касс должно быть в магазине? С одной стороны кассы не должны простаивать, а с другой стороны чтобы из-за их нехватки не создавалось бы очереди. Лишние кассы – неэффективные затраты, очереди – потеря покупателей и объема продаж, а, следовательно, прибыли.

Готовые решения в данном случае найти трудно, но можно попытаться построить имитационную компьютерную модель и на ней «проиграть» различные сценарии. Например, предположим, что в час пик сломалась одна касса. Насколько увеличится время обслуживания одного покупателя? Или перед праздником возник наплыв посетителей – сколько дополнительных кассиров необходимо привлечь?

Процесс обслуживания чрезвычайно прост – покупатель стоит в очереди и подходит к кассе, затем выкладывает товар перед кассиром. С этого момента начинается подсчет итоговой суммы, пробиваемой на чеке. Время предъявления товара и время подсчета носит также случайный характер и распределено по заданному закону, как и количество товара в корзинке.

Нас может интересовать среднее время обслуживания одного покупателя в зависимости от числа касс и скорости работы кассиров, или средняя сумма приобретенных товаров одним покупателем в зависимости от вида законов, характеризующих дневные потребности и т.п.

С математической точки зрения, данная задача решается очень просто – надо случайным образом «создать» **поток покупателей**, т.е. множество корзинок с товарами. Каждая корзинка характеризуется **вектором количества товаров**, состоящих из четырех компонентов, которым необходимо присвоить определенные значения. Далее следует случайным образом приписать каждому покупателю время предъявления товара и длительность подсчета итоговой суммы.

Осталось усреднить сумму времени предъявления товара и времени подсчета, чтобы получить среднее время обслуживания, и просуммировать все чеки. Таким образом, нам необходимо построить имитационную модель системы массового обслуживания (модель универсама).

Модель будет представлять собой систему массового обслуживания, в которой будет использовано 5 параметров: bp (цена масла 120 рублей), brp (цена хлеба 38 рублей), cp (цена сыра 180 рублей), mp (цена молока 68 рублей), s (общая стоимость всех купленных товаров, начальное значение 0).

Создание модели. При создании модели создается новый агент **Prod** (на базе класса agent) с набором параметров, характеризующим количество каждого проданного товара. Каждый покупатель (заявка) покупает свой набор товаров (молоко, сыр, масло, хлеб) и определенное количество этих товаров, что и сохраняется в переменных **bread, butter, cheese, milk** агента **Prod**.

Каждый покупатель оплачивает свою корзину покупок, стоимость корзины зависит от количества товаров и цены. Вопрос – покупать или не покупать тот или иной товар – решается достаточно просто: случайным образом выбирается либо 0 (товар не положен в корзину), либо 1 (товар в корзине). Общую стоимость корзины покупателя можно рассчитать по

следующему алгоритму (данный код является Java-обработчиком действия **«При выходе»** объекта **«Delay»**).

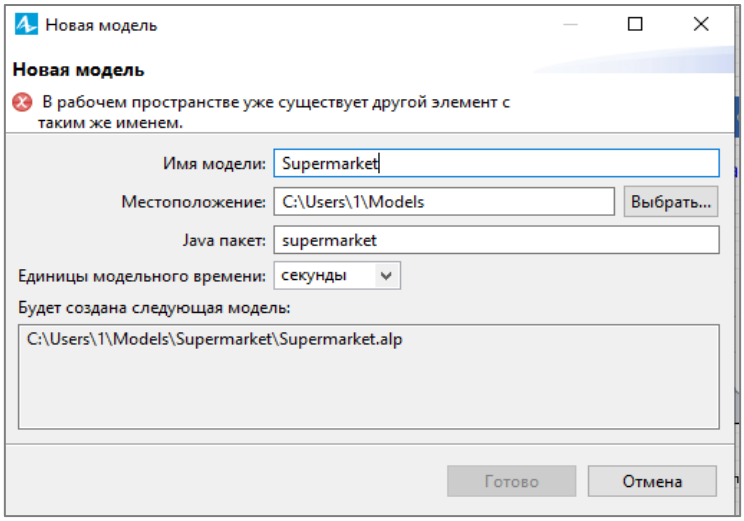


Рис. 5.3.1. Создание модели Supermarket

1. Создадим новый проект в AnyLogic: **«Файл»–«Создать»–«Модель»**. Назовите новую модель **Supermarket** (рис. 5.3.1).

2. Создайте диаграмму очереди из следующих объектов **«Source», «Queue», «Delay», «Sink»**. Нужные объекты находятся на вкладке **«Палитра»**, разделе **«Библиотека моделирования процессов»** (рис. 5.3.2).

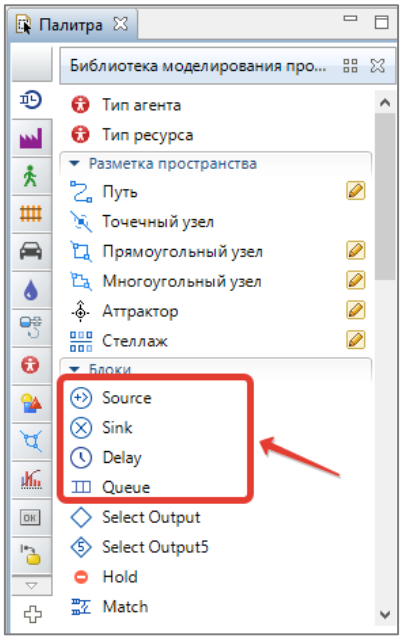


Рис. 5.3.2. Библиотека моделирования процессов

Объект **«Source»** моделирует источник заявок, в нашем случае это покупатели в магазине.

Объект **«Queue»** моделирует очередь на кассу.

Объект **«Delay»** моделирует задержку кассира, которая непременно возникнет, т.к. нужно просчитать все покупки, выставить счет, и покупатель должен его оплатить.

Объект **«Sink»** обозначает конец блок-схемы и выполняет удаление обработанных заявок (покупателей) из системы.

Соедините указанные блоки, как показано на рис. 5.3.3.

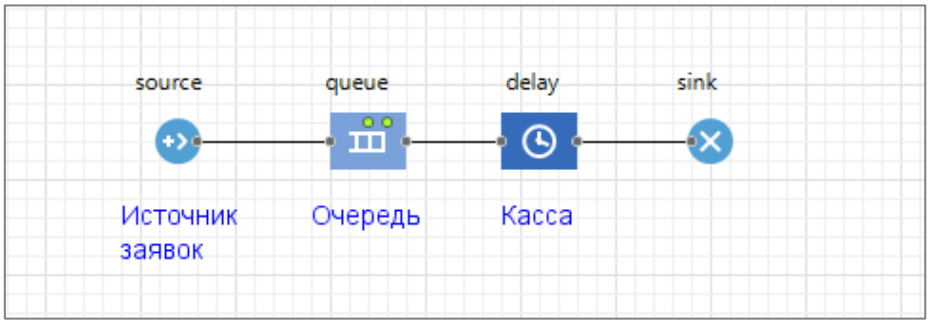


Рис.5.3.3. Модель универсама из блоков

3. На вкладке **«Свойства»** для объекта **«Source»** установите параметры:
   - **Прибывают согласно** – интенсивности.
   - **Интенсивность прибытия** равна 110 человек в час.
4. Для объекта **«Queue»** на вкладке **«Свойства»** установите параметр **«Вместимость очереди»** – 100 заявок**.**
5. Для объекта **«Delay»** на вкладке **«Свойства»** установите параметры:
   - **Тип задержки** – Определенное время.
   - **Вместимость** – 1 (1 человек обслуживается на кассе).
6. Создадим параметры для нашей модели, содержащие значения цены на молоко, сыр, масло, хлеб и их сумму.

Перейдите на вкладку **«Палитра»** в раздел **«Агент»**. Перенесите на форму объект **«Параметр»**. В появившемся справа окне **«Свойства»** установите следующие значения (рис. 5.3.4):

Имя **mp.**

Тип **double.**

Значение по умолчанию 68 (цена молока).

Рядом с параметром поместите из раздела **«Презентация»** вкладки **«Палитра»** объект **«Текст»** и подпишите – **Цена молока**.

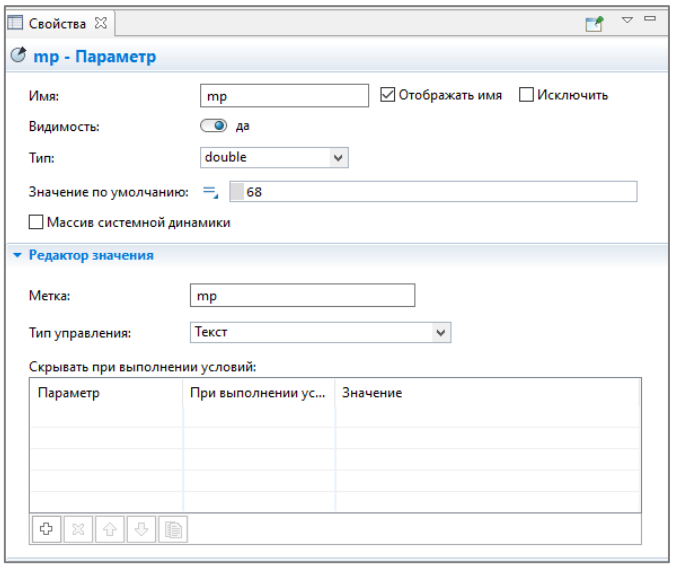


Рис. 5.3.4. Настройка свойств для параметра mp

Для второго параметра установите следующие значения в окне свойства:

Имя **cp**

Тип **double**

Значение по умолчанию 180 (цена сыра).

Рядом с параметром поместите из раздела **«Презентация»** вкладки **«Палитра»** объект **«Текст»** и подпишите – **Цена сыра**.

Для третьего параметра установите следующие значения в окне свойства:

Имя **bp**

Тип **double**

Значение по умолчанию **120** (цена масла).

Рядом с параметром поместите из раздела **«Презентация»** вкладки **«Палитра»** объект **«Текст»** и подпишите – **Цена масла**.

Для четвертого параметра установите следующие значения в окне свойства:

Имя **brp**

Тип **double**

Значение по умолчанию **36** (цена хлеба).

Рядом с параметром поместите из раздела **«Презентация»** вкладки **«Палитра»** объект **«Текст»** и подпишите – **Цена хлеба**.

Для пятого параметра установите следующие значения в окне свойства:

Имя **s**

Тип **double**

Значение по умолчанию **<пусто>.**

Рядом с параметром поместите из раздела **«Презентация»** вкладки **«Палитра»** объект **«Текст»** и подпишите – **Сумма**.

Вид диаграммы модели с параметрами показан на рис. 5.3.5.

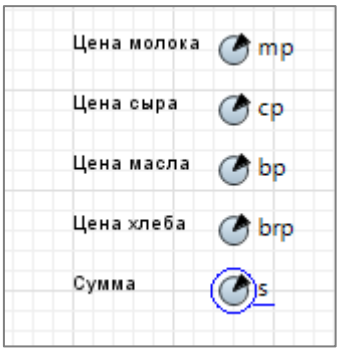


Рис. 5.3.5. Параметры модели

7. Для нашей задачи нужно ввести несколько переменных, следовательно, стандартный класс агента нам не очень подойдет. Создадим новый тип агента **«Prod»**. Для этого в дереве объектов выберите правой кнопкой мыши **«Создать»** – **«Тип агента»** (рис. 5.3.6).

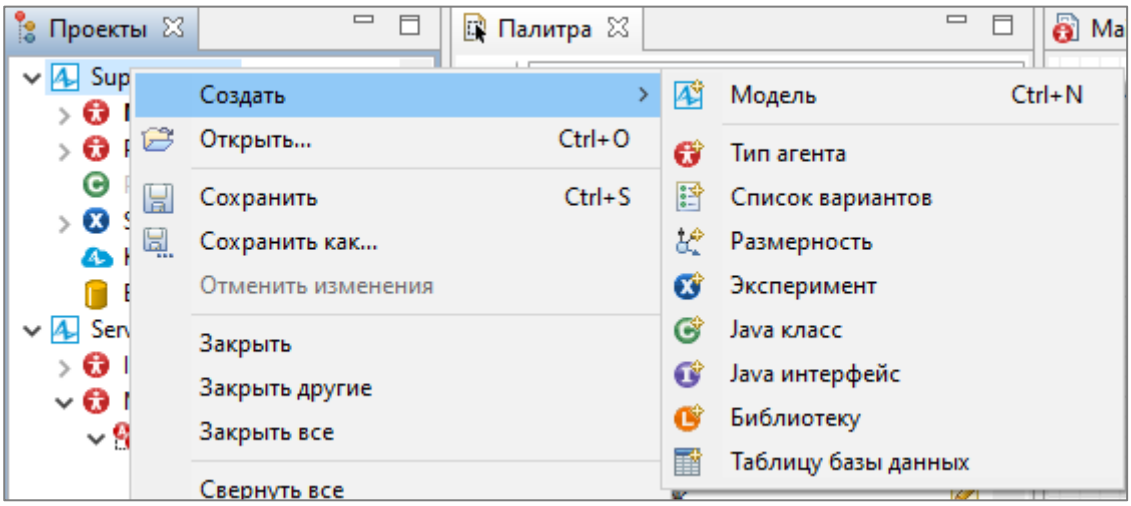


Рис. 5.3.6. Создание нового типа агента

8. В появившемся окне укажите
   - Имя нового типа: **Prod**
   - Установите галочку в поле **«Использовать новый тип агента»** **«с нуля»** (рис. 5.3.7).
   - Нажмите на кнопку **«Далее»**.

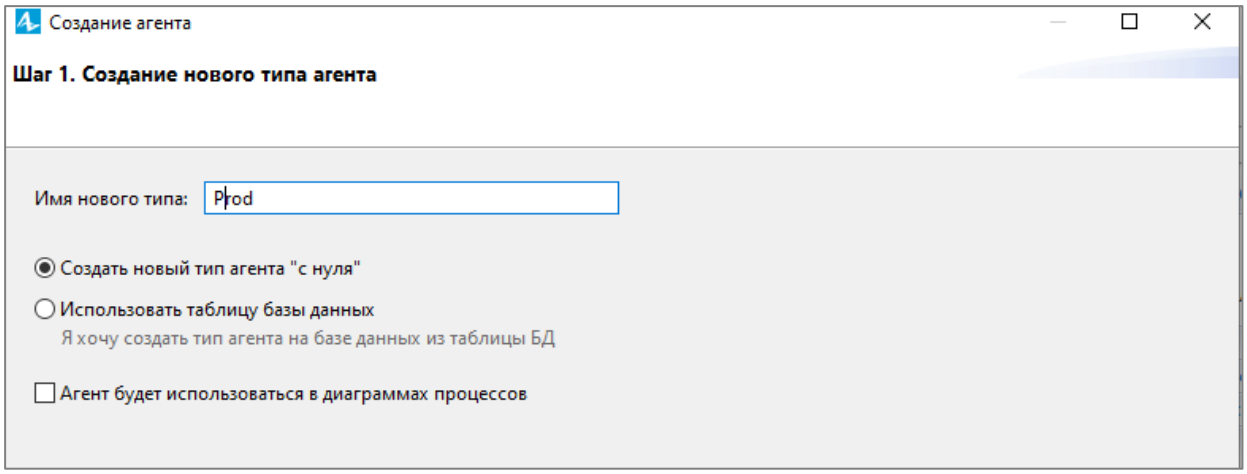


Рис. 5.3.7. Шаг 1 в создании нового типа агента «Prod»

9. На втором шаге **«Анимация агента»** выберите для анимации агента значение **«нет»**. Нажмите на кнопку **«Далее»**.

10. На третьем шаге просто нажмите на кнопку **«Готово».**

11. Откройте страницу нового класса **«Prod»** и добавьте 4 переменные (рис. 5.3.8):

- Имя **milk**, Тип выберите **double,** начальное значение 0;
- Имя **butter**, **Тип** выберите **double,** начальное значение 0;
- Имя **bread**, **Тип** выберите **double,** начальное значение 0;
- Имя **cheese**, **Тип** выберите **double,** начальное значение 0.

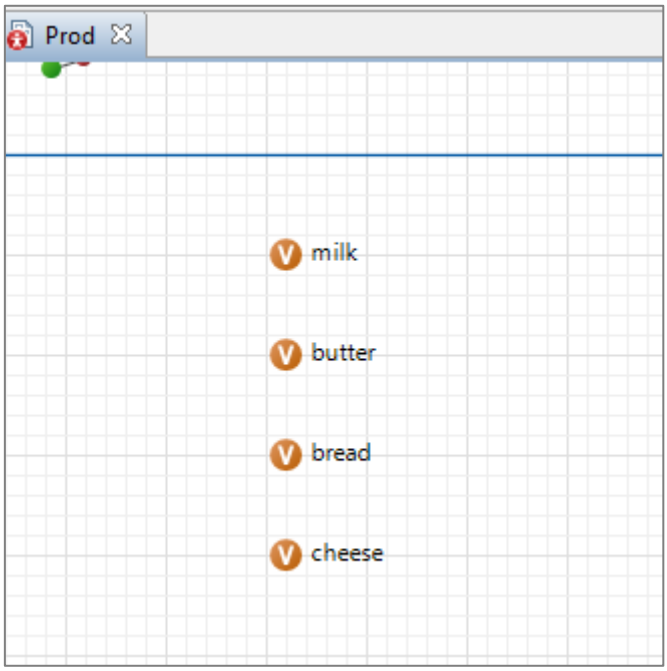


Рис. 5.3.8. Добавление переменных к классу Prod

12. Добавьте к классу **«Prod»** еще 4 переменных для подсчета количества каждого вида продуктов (рис. 5.3.9):

- Имя **milk_count**, Тип выберите **int,** начальное значение 0;
- Имя **butter_count**, **Тип** выберите **int,** начальное значение 0;
- Имя **bread_count**, **Тип** выберите **int,** начальное значение 0;
- Имя **cheese_count**, **Тип** выберите **int,** начальное значение 0.

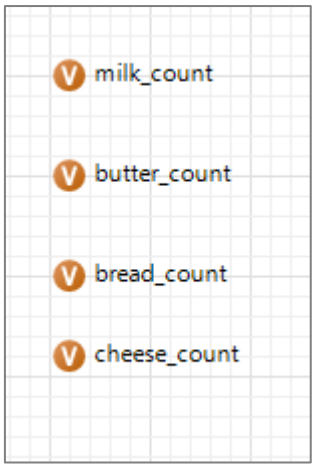


Рис. 5.3.9. Переменные для подсчета количества каждого вида продуктов, класс «Prod»

13. Перейдите на **«Main»** вашего проекта. Выберите объект **Source** (источник заявок) и на странице **«Свойства»** в разделе **«Агент»** из выпадающего списка **«Новый агент»** выберите **«Prod»,** в разделе **«Специфические»** выберите тип агента **«Prod»** (рис. 5.3.10).

14. Проверьте для объектов **«Queue»**, **«Delay»** и **«Sink»** – все поля **«Тип агента»** должны смениться на **«Prod»**. Если это по каким-то причинам не произошло, установите это значение из выпадающего списка самостоятельно.

15. Для объекта **«Queue»** (очередь) в окне **«Свойства»** в разделе **«Действия»** задайте в поле **«При входе»** следующее выражение:

```
((Prod)agent).milk_count = (int) round(uniform(0,1));
((Prod)agent).bread_count = (int) round(uniform(0,1));
((Prod)agent).cheese_count = (int) round(uniform(0,1));
((Prod)agent).butter_count = (int) round(uniform(0,1)).
```

Здесь устанавливается, сколько покупок совершает каждый покупатель. Для упрощения модели считаем, что из четырех продуктов отдельный покупатель либо берет какой-то товар, либо не берет его.

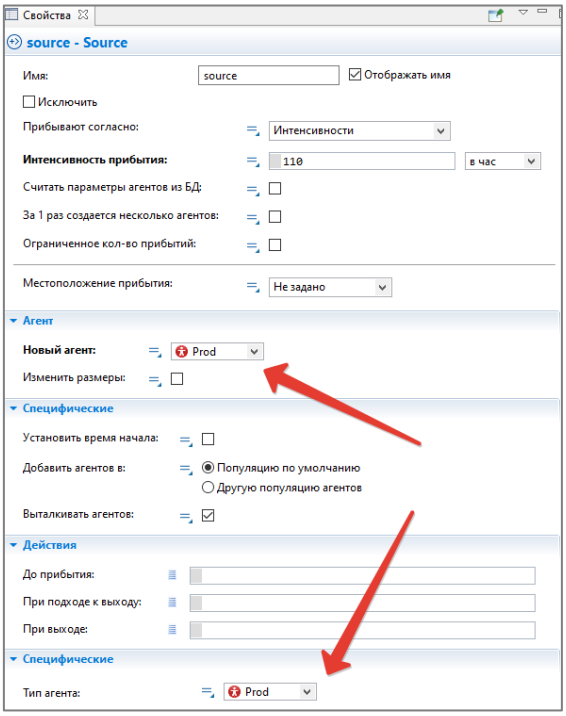


Рис. 5.3.10. Изменение агента модели

Запись

(int) round(uniform(0,1));

означает, что генерируется случайное число от 0 до 1 целого типа.

16. Вычислим время задержки на кассе каждого конкретного покупателя. Будем считать, что все покупатели расплачиваются с одной и той же скоростью, поэтому время обслуживания на кассе будет зависеть от количества и вида выбранных продуктов.

Выберите на диаграмме модели Main объект **Delay** (касса) и в окне Свойства в поле **«Время задержки»** установите значение:

((Prod)agent).milk_count * uniform(2,8) +

((Prod)agent).bread_count * uniform(2,8) +

((Prod)agent).cheese_count * uniform(2,8) +

((Prod)agent).butter_count * uniform(2,8).

Будем считать, что на кассовое обслуживание одного продукта уходит от 2 до 8 секунд.

17. Для объекта **«Delay»** в окне **«Свойства»** установите в поле **«Действия при входе»** выражение:

((Prod)agent).milk=((Prod)agent).milk_count*mp;

((Prod)agent).bread=((Prod)agent).bread_count*brp;

((Prod)agent).cheese=((Prod)agent).cheese_count*cp;

((Prod)agent).butter=((Prod)agent).butter_count*bp.

В этом выражении подсчитывается стоимость покупки каждого покупателя, т.к. происходит перемножение количества товара и стоимости одной единицы товара.

18. Для определения выручки магазина **s** в текущий момент времени используется специальный метод **sum(),** который добавляется к агенту **«Prod»** в качестве дополнительного кода класса.

Выберите в дереве объектов класс **«Prod»** и в окне **«Свойства»** в разделе Java для экспертов задайте в поле **«Дополнительный код класса»:**

double sum () { //метод возвращает значение типа double , и имеет имя sum

return this.milk + this.butter + this.bread +this.cheese; //вернуть сумму значений полей

}

Выражение **s+= ((Prod)agent).sum()**; будет подсчитывать суммарную выручку от всех покупок.

Настройка статистики модели

1. Для графической иллюстрации приобретения того или иного товара в модель вводятся 4 объекта **«Набор данных»** палитры **«Статистика»** – milk, cheese, butter, bread (рис. 5.3.11).

Набор данных в системе AnyLogic представляет собой двумерный массив измерений, который впоследствии может быть использован как источник значений для построения различных диаграмм.

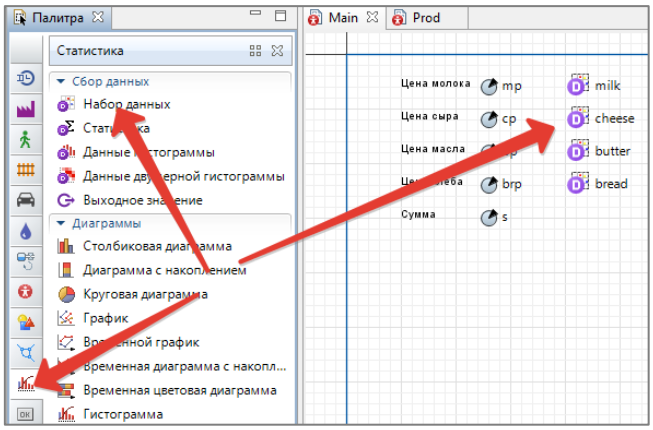


Рис. 5.3.11. Создание наборов данных для вывода статистики

Для изменения значений наборов данных служит метод **dat()**, который добавляет сумму, потраченную на приобретение каждого товара, в соответствующий набор данных (с целью графического отображения процесса покупки). Собственно для добавления значения в набор данных служит метод **add()**. Код метода **dat**() добавляется к агенту **«Prod»** как дополнительный код класса (рис. 5.3.12):

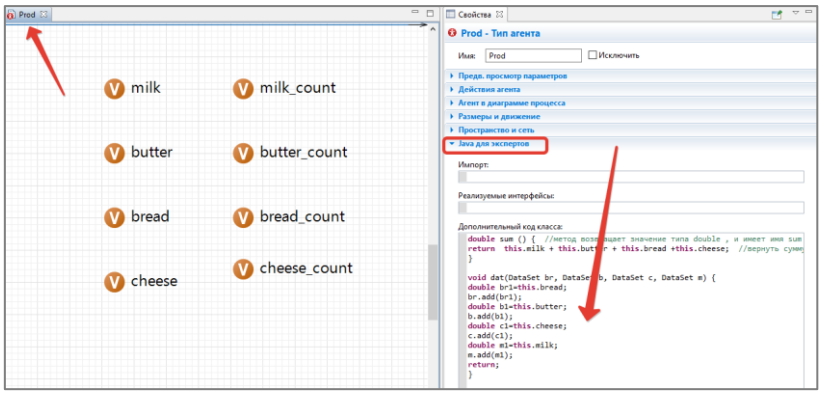


Рис. 5.3.12. Добавление метода dat() к коду агента Prod

```
void dat(DataSet br, DataSet b, DataSet c, DataSet m) {
double br1=this.bread;
br.add(br1);
double b1=this.butter;
b.add(b1);
double c1=this.cheese;
c.add(c1);
```

double m1=this.milk;

m.add(m1);

return;

}

2. Для подсчета количества купленных товаров используется метод **dat_count()**, который добавляет к каждому набору данных количество соответствующего вида продукции.

void dat_count(DataSet br, DataSet b, DataSet c, DataSet m) {

double br1=this.bread_count;

br.add(br1);

double b1=this.butter_count;

b.add(b1);

double c1=this.cheese_count;

c.add(c1);

double m1=this.milk_count;

m.add(m1);

return;

}

Вышеприведенный код также нужно добавить к агенту **«Prod»** как дополнительный код класса (рис. 5.3.13).

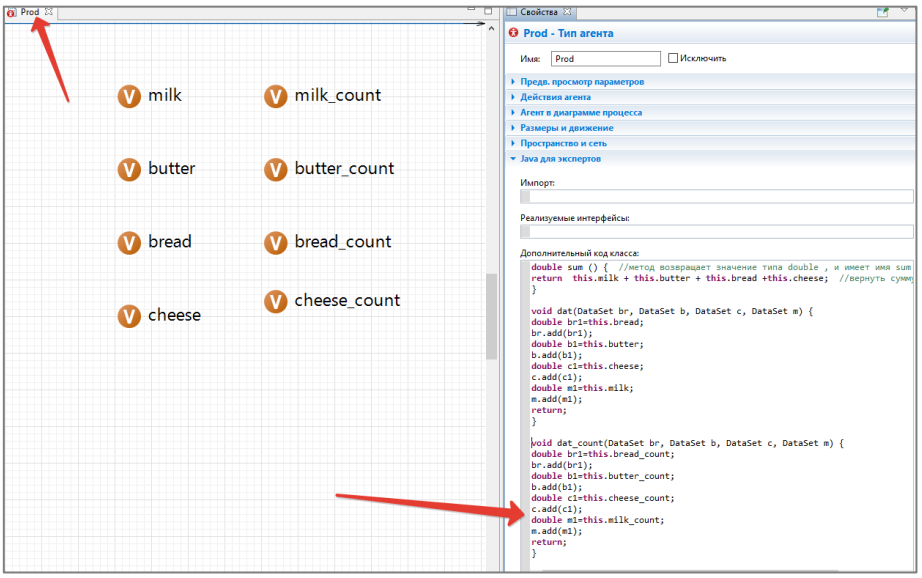


Рис. 5.3.13. Добавление метода dat_count() к коду агента Prod

3. В процессе моделирования происходит уничтожение заявок объектом **«Sink»**, в котором удобно регистрировать общую выручку, полученную магазином, а также значения наборов данных **milk, cheese, butter, bread**, отражающих получение суммы, потраченной на приобретение каждого товара. В объекте **«Sink»** в действии **«При выходе»** оформляется следующий код (рис. 5.3.14):

s+= ((Prod)agent).sum();

((Prod)agent).dat(bread,butter,cheese,milk);

((Prod)agent).dat_count(bread_count,butter_count,cheese_count,milk_count).

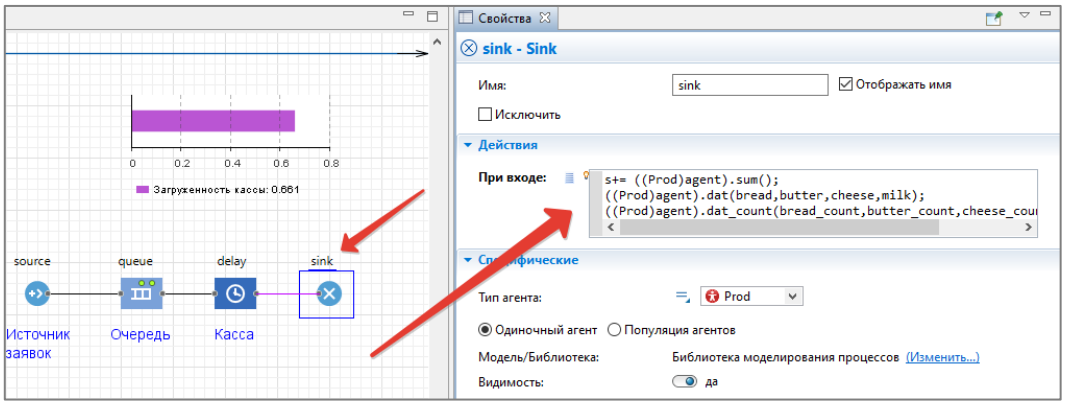


Рис. 5.3.14. Добавление действия при выходе для объекта Sink

4. Графическая иллюстрация модели состоит из двух столбиковых диаграмм и четырех временных графиков. Все они расположены на **«Палитре»** в разделе **«Статистика»**.

Столбиковые диаграммы:

- загруженность кассы отражает значение **delay.statsUtilization.mean()** (рис. 5.3.15);

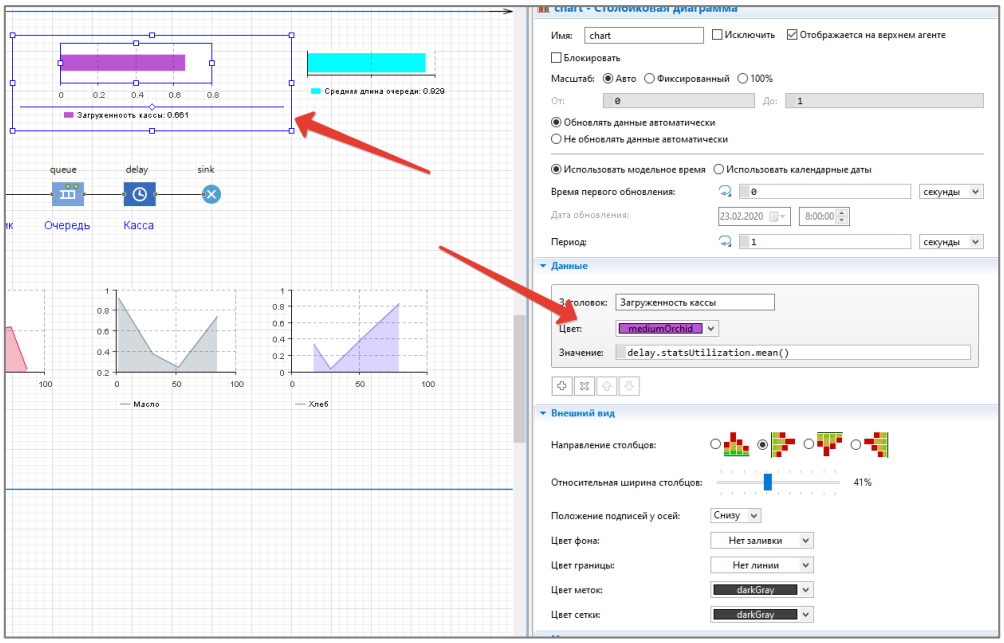


Рис. 5.3.15. Загруженность кассы

- средняя длина очереди отражает значение **queue.statsSize.mean()** (рис. 5.3.16).

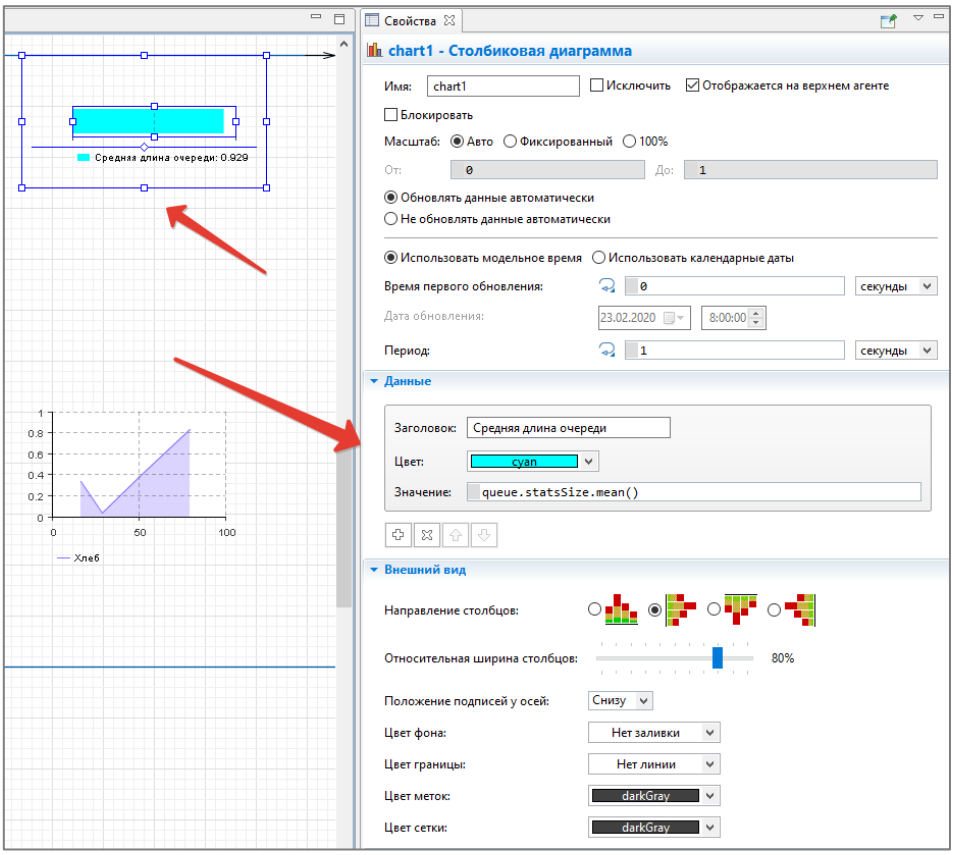


Рис. 5.3.16. Средняя длина очереди

5. Временные графики отражают наборы данных по каждому товару. На рис. 5.3.17 показан вид временного графика для товара **«Хлеб»**. Остальные графики сделайте аналогичным способом. Установите связь каждого графика со своим набором данных.

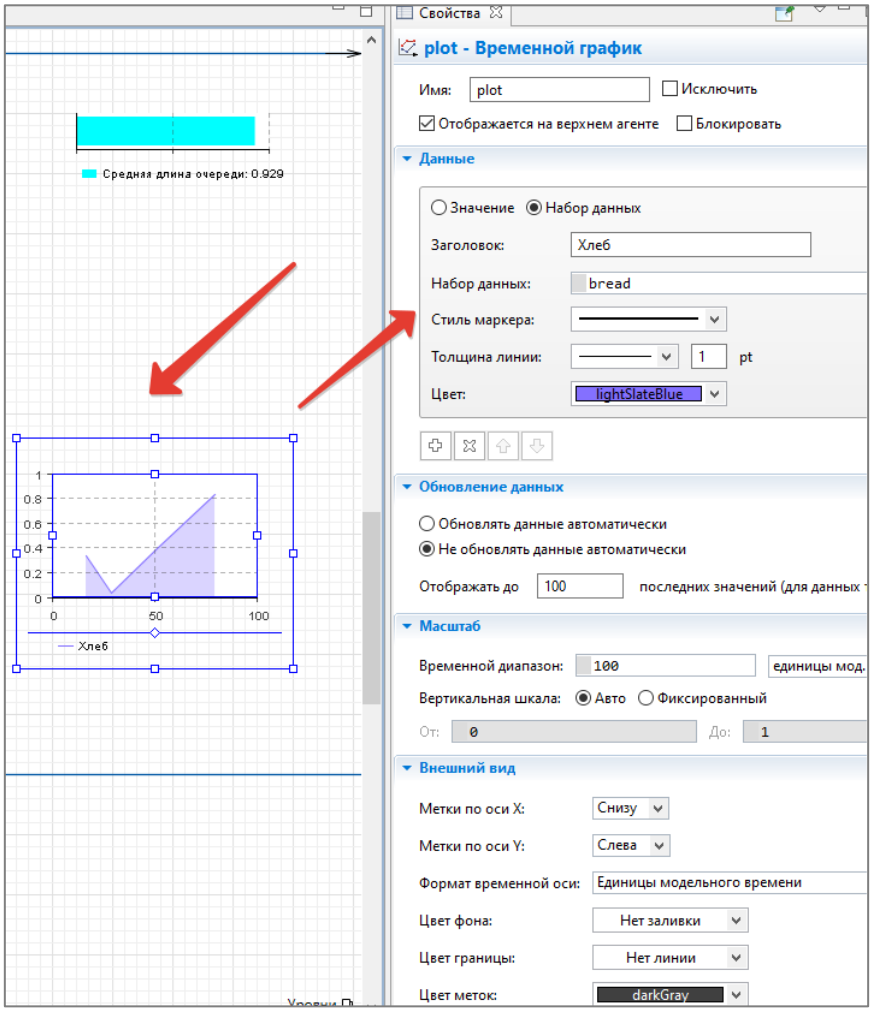


Рис. 5.3.17. Настройка свойств временного графика для отображения покупок хлеба

Запуск модели. Для запуска модели использовались значения, максимально приближенные к реальной ситуации в магазине. Так, единицей модельного времени являются минуты, заявки прибывают согласно интенсивности 40 покупателей в час (**source**), касса обслуживает отдельного покупателя с задержкой от 50 до 180 секунд (время задержки **delay** со значением **uniform (50,180)**). Симуляция модели происходит в течение двух часов (начальное время 1, конечное время 120).

Опишем процедуру настройки параметров для запуска модели подробно.

1. Перейдите в дерево объектов и выберите узел **«Simulation: Main»**. В окне **«Свойства»** установите следующие параметры (рис. 5.3.18).

- Режим выполнения: **реальное время со скоростью 1**
- Остановить: **в заданное время**
- Начальное время: 1
- Конечное время: 60.

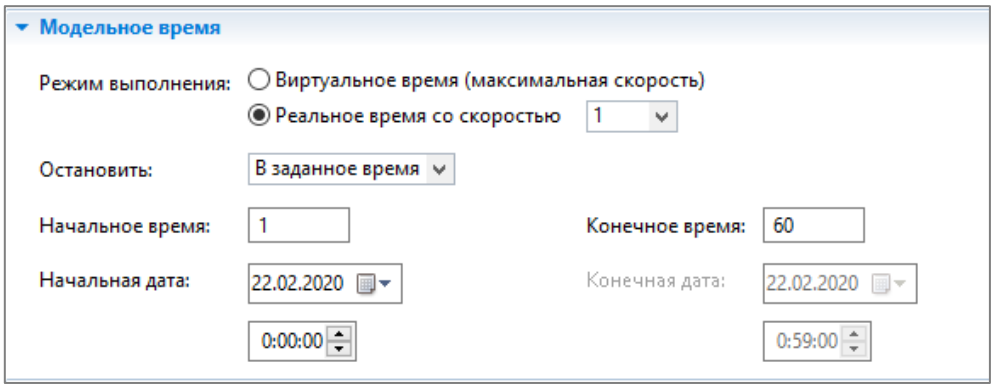


Рис. 5.3.18. Настройка параметров модельного эксперимента

2. Запустите модель. Примерный вид показан на рис. 5.3.19.

Рис. 5.3.19. Имитационная модель работы супермаркета в системе AnyLogic

5.4 Исследовательские задачи к главе 5

1. Какую выручку за время моделирования получит магазин?
2. На сколько загружена касса? При каких значениях параметров можно добиться коэффициента использования рабочего времени кассира 0,7?
3. Как повлияют на выручку магазина возможные отмены покупок товаров, например, из-за неправильно выставленных ценников (формально, если верхнее значение обслуживание отдельной заявки увеличится до 3,5 минут)?
4. Определить среднюю длину очереди при моделировании. Согласно статистике, очередь будет покидать каждый пятый покупатель. При каких параметрах это будет происходить?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. 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.

2. Бережная, Е.В. Математические методы моделирования экономических систем / Е.В. Бережная, В.И. Бережной. – Москва: Финансы и статистика, 2003. – 368 с. – ISBN 5-279-02291-8.

3. Боев В.Д. Компьютерное моделирование: пособие для практических занятий, курсового и дипломного проектирования в AnyLogic / В.Д. Боев. – СПб.: ВАС, 2014. – 432 с.

4. Григорьев, И.И. AnyLogic за три дня: практическое пособие по имитационному моделированию / И.И. Григорьев // Интернет-портал фирмы AnyLogic: [сайт]. – URL: https://www.anylogic.ru/resources/books (дата обращения: 12.12.2021). – 273 с.

5. Глухов, Д.О. Моделирование систем управления: практикум / Д.О. Глухов, И.В. Петухов; под ред. Д.О. Глухова. – Йошкар-Ола: Поволжский государственный технологический университет, 2015. – 84 с. – URL: http://www.iprbookshop.ru/75437.html. – ISBN 978-5-8158-1546-9.

6. Данелян, Т.Я. Теория систем и системный анализ: учеб. пособие / Т.Я. Данелян. – Москва: Евразийский открытый институт, 2011. – 303 с. – ISBN 978-5-374-00324-6. – URL: http://www.iprbookshop.ru/10867.html.

7. Карпов, Ю.Г. Имитационное моделирование систем. Введение в моделирование с AnyLogic / Ю.Г. Карпов. – Санкт-Петербург: БХВ-Петербург, 2005. – 400 с. – ISBN 5-94157-148-8.

8. Каталевский, Д.Ю. Основы имитационного моделирования и системного анализа в управлении / Д.Ю. Каталевский. – Москва: ДЕЛО РАНХ и ГС, 2015. – 485 с. – ISBN 976-5-7749-1072-4.

9. Колесов, Ю.Б. Объектно-ориентированное моделирование сложных динамических систем / Ю.Б. Колесов. – Санкт Петербург: Изд-во СПбГТУ, 2004. – 240 с.

10. Колесов, Ю.Б. Моделирование систем. Динамические и гибридные системы / Ю.Б. Колесов, Ю.Б. Сениченков. – Санкт-Петербург: БХВ-Петербург, 2006. – 224 с. – ISBN 5-94157-578-5.

11. Колесов, Ю.Б. Моделирование систем. Объектно-ориентированный подход / Ю.Б. Колесов, Ю.Б. Сениченков. – Санкт-Петербург: БХВ-Петербург, 2006. – 192 с. – ISBN 5-94157-579-3.

12. Колесов, Ю.Б. Моделирование систем: практикум по компьютерному моделированию / Ю.Б. Колесов, Ю.Б. Сениченков. – Санкт-Петербург: БХВ-Петербург, 2007. – 352 с. – ISBN 5-94157-580-7.

13. Королев, А.Л. Основы теории автоматического управления: учебное пособие и методические указания к лабораторным работам / А.Л. Королев. – Челябинск: Изд-во Южно-Урал. гос. гуманитар.-пед. ун-та, 102 с. – ISBN 978-5-907409-82-8.

14. Королев, А.Л. Компьютерное моделирование объектов процессов и систем / А.Л. Королев, Н.Б. Паршукова – Челябинск: Издательство ЮУрГГПУ, 2020. – 329 с. – ISBN 978-5-907409-15-6.

15.Неуймин, Я.Г. Модели в науке и технике / Я.Г. Неуймин. – Ленинград: Наука, 1984. – 283 с.

16. Потемкин, А.М. Трехмерное твердотельное моделирование в системе КОМПАС-3D / А.М. Потемкин. – Санкт-Петербург: БХВ-Петербург, 2004. – 512 с. – ISBN 5-94157-472-x.

17. Перегудов, Ф.И. Введение в системный анализ / Ф.И. Перегудов, Ф.П. Тарасенко. – Москва: Высшая школа, 1989. – 367 с. – ISBN 5-063-001569-6.

18. Сметанина Е.И. Системный анализ в вопросах и ответах: учебное пособие / Е.И. Сметанина. – Томск: Томский политехнический университет, 2016. – 108 с. – ISBN 978-5-4387-0678-6. – URL: http://www.iprbookshop.ru/83984.html.

19. Ссылки для загрузки архивов, помещенных на страницах раздела «Моделирование. ТАУ и смежные вопросы» сайта «Моделирование систем и явлений». – URL:https://klinachevnv.ru/fedosov/bt_download.html (дата обращения: 10.12.2020). – Режим доступа: свободный, яз. русский.

20. Тарасевич, Ю.Ю. Математическое и компьютерное моделирование. Вводный курс / Ю.Ю. Тарасевич. – Москва: Едиториал УРСС, 2019. – 149 с. – ISBN 5-354-00913-8.

21. Образовательный сайт компании «Ascon». – URL: http://www.edu-ascon.ru.

22. Образовательный сайт «VisSim в России». – URL: http://www.vissim.com.

23. Сайт компании «AnyLogic». – URL: http://www.anylogic.ru.

24. Сайт компании «MVStudium Group». – URL: http://www.mvstudium.com.

Учебное издание

Королев Александр Леонидович

Паршукова Наталья Борисовна

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ОБЪЕКТОВ, ПРОЦЕССОВ И СИСТЕМ

Учебно-практическое пособие

ISBN 978-5-907611-11-5

Рукопись рекомендована РИС ЮУрГГПУ

Протокол № 25, 2022 г.

Редактор Е.М. Сапегина

Технический редактор Н.А. Усова

Издательство ЮУрГГПУ

454080, г. Челябинск, пр. Ленина, 69

Подписано в печать 28.03.2022

Объем 11 уч.-изд. л. (36 усл. п. л.)

Формат 60*84/8 Тираж 100 экз.

Заказ

Отпечатано с готового оригинал-макета в типографии ЮУрГГПУ

454080, г. Челябинск, пр. Ленина, 69