Документация, содержащая описание функциональных характеристик программного обеспечения «UFOModeler» и информацию, необходимую для установки и эксплуатации программного обеспечения

Оглавление

1. Назначение программного обеспечения «UFOModeler»

1.1. Функциональное назначение программного обеспечения

Функциональным назначением программного обеспечения «UFOModeler» является:

– построение моделей производственно-технологических и организационно-деловых процессов на основе системно-объектного подхода «Узел-Функция-Объект»;

– имитация функционирования построенной модели в реальном времени;

– аналитическое представление моделируемых данных, полученных в результате имитации, в виде графиков.

1.2. Краткое описание программного обеспечения

Программное обеспечение «UFOModeler» создано для описания и имитации производственно-технологических и организационно-деловых процессов и систем, основываясь на подходе «Узел-Функция-Объект».

«Узел-Функция-Объект» – системный подход, представляющий систему в виде целостной трехэлементной конструкции, состоящей из узла, функции и объекта (УФО-подход).

УФО-подход позволяет рассматривать любую систему или предметную область как совокупность:

– взаимодействующих «перекрестков» с набором входящих и исходящих связей, называемых узлами;

– методов обработки связей узлов, называемых функциями;

– обработчиков связей узлов, называемых объектами.

Графическое описание процессов и систем в рамках данного подхода представляет собой диаграмму (контекстного уровня или уровня декомпозиции), которая состоит из элементов «Узел-Функция-Объект» (УФО-элементов) и связей, и называется далее УФО-моделью.

Описание функционирования системы, то есть того, как объект в узле преобразует входящие связи в исходящие, проводится путем написания скрипта функционирования для каждого узла с использованием специального языка UFO-скрипт.

Имитация функционирования системы, т.е. преобразование УФО-модели в имитационную модель осуществляется следующим образом:

– для каждого функционального элемента выделяются динамические характеристики: время выполнения, блокирования процесса;

– задаются значения динамических параметров функциональных элементов модели;

– определяется порядок возникновения событий в модели, т.е. поступление сообщений из надсистемы моделируемой системы;

– производится отсчет времени и отслеживается изменение параметров функциональных элементов УФО-модели до какого-либо заданного момента времени или до какого-либо состояния модели.

Аналитическим инструментом для отслеживания изменений параметров функциональных элементов является построение графиков, отражающих изменение параметров узла или связей, с течением времени или отражение зависимости между двумя выбранными параметрами.

В программе имеется возможность экспорта модели в файлы с расширениями *jpg*, *bmp*. А также предусмотрена функция отправки на печать построенной модели.

1.3. Краткое описание языка UFO-скрипт

Описание функционирования системы, то есть того, как объект в узле преобразует входящие связи в исходящие, проводится путем написания скрипта функционирования для каждого узла с использованием специального языка UFO-скрипт.

Язык UFO-скрипт основан на языке Pascal. Далее приведены синтаксические правила языка.

– Текстовые константы выделяются апострофами и должны располагаться на одной строке: *‘Текстовая строка’*.

– Если в тексте необходимо вставить апостроф, то его нужно заменить двумя апострофами: *‘Строка ’’ строка’*.

– Комментарии: // комментарий до конца строки

либо:

*/*многострочный*

комментарий/*

либо:

{многострочный

комментарий}

– Программа может содержать блок описания переменных и исполняемый блок:

Блок описания переменных начинается служебным словом *var* и заканчивается разделителем «;» (точка с запятой).

В блоке описания переменных может содержаться любое количество блоков переменных, которые содержат список идентификаторов (имен переменных), разделенных запятой, после которого следует двоеточие и идентификатор типа, после которого, в свою очередь, ставится символ «;» (точка с запятой): *a, b:integer;*

– Исполняемый блок выделяется служебными словами *begin* и *end*, заканчивается символом «.» (точка).

– Исполняемый блок состоит из операторов, разделяемых символом «;» (точка с запятой).

Список операторов, процедур и функции, а также математические функции приведены в Практическом руководстве Приложении 1.

Пример создания UFO-скрипта приведен в Практическом руководстве «Создание решения 2».

2. Условия применения

2.1. Минимальные аппаратные требования

– Процессор 32 или 64 разрядный, с тактовой частотой 1 ГГц (или быстрее);

– оперативная память 2 Гб (32 разрядный процессор) или 3 Гб (64 разрядный процессор);

– 100 Мб свободного места на жестком диске;

– видеокарта с поддержкой DirectX 9 и поддержкой стандартного видеодрайвера ОС Windows «WDDM 1.0» или выше;

– монитор с разрешением 1024x768;

– подключение к сети Интернет (для активации UFOModeler).

2.2. Минимальные программные требования

Операционная система Windows XP,7,8,10.

3. Практическое руководство «UFOModeler»

3.1. Создание решения 1

Построим контекстную модель процесса с использованием UFOModeler. Запустите приложение «UFOModeler.exe». Контекстная диаграмма представляет собой наиболее общее описание процесса или системы. В отобразившемся диалоговом окне необходимо выбрать пункт «Создать новую модель». В новом окне вводим название модели, при необходимости задаем стиль и размер окна. Диалоговое окно представлено на рисунке 1.

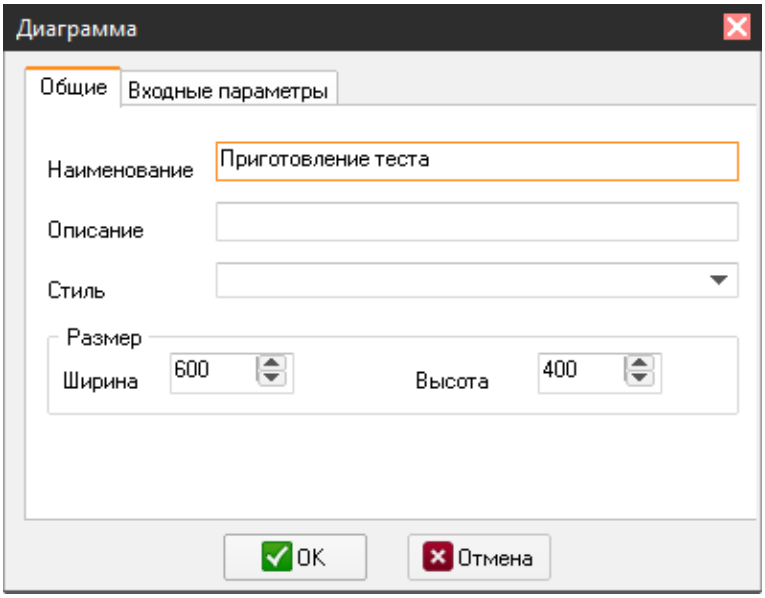

Рис. 1. Диалоговое окно «Диаграмма»

В поле «Наименование» вводится наименование модели, поле «Описание» предназначено для текстового описания создаваемой модели, в поле «Стиль» задается стиль отображения. В поле «Размер» задается ширина и высота рабочего поля, в котором создается модель.

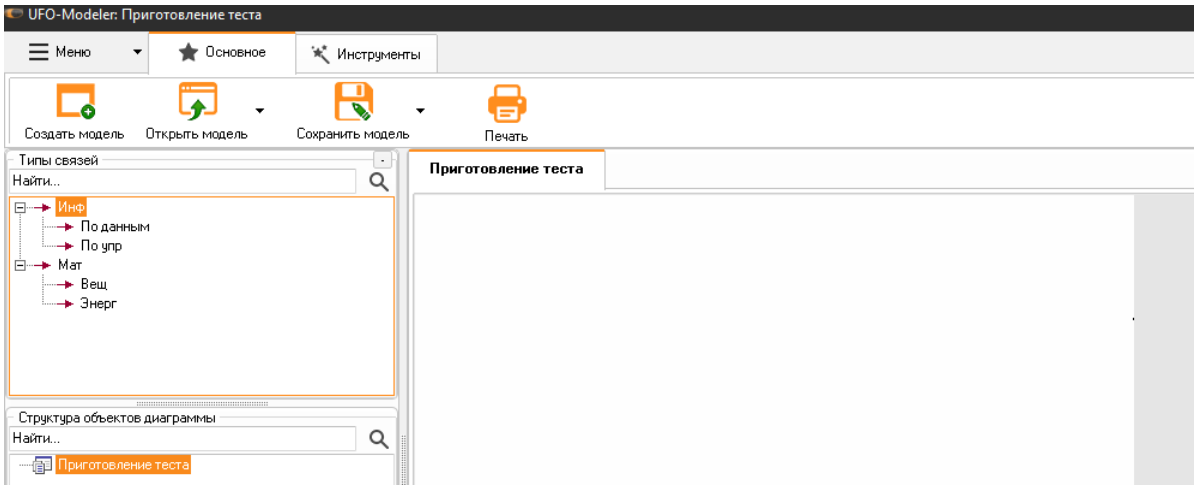

Рис. 2. Создание новой модели

После нажатия кнопки «ОК» открывается поле для создания модели. Скриншот окна представлен на рисунке 2.

В верхней части окна расположено меню с выпадающим списком, а так же вкладки «Основное» и «Инструменты». В блоке «Типы связей» отображаются все используемые при моделировании связи, распределенные по типам. В блоке «Структура объектов диаграммы» отображаются используемые в модели объекты и соответствующие им функции. И, наконец, большую часть окна занимает рабочая область, на которой будет расположена разрабатываемая модель.

На рисунках 3 - 5 представлены основные пункты меню программы.

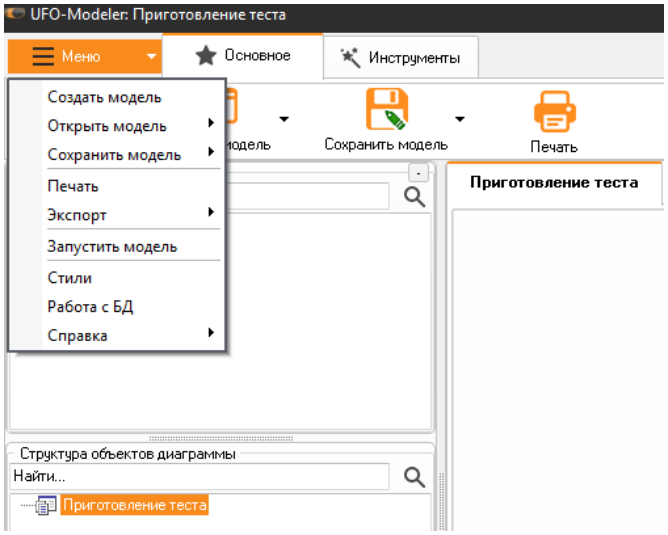

Рис. 3. Вкладка «Меню»

При нажатии на кнопку «Меню» доступны такие функции, как создание, открытие и сохранение модели, печать и экспорт, запуск модели, а также работа со стилями, БД и справка.

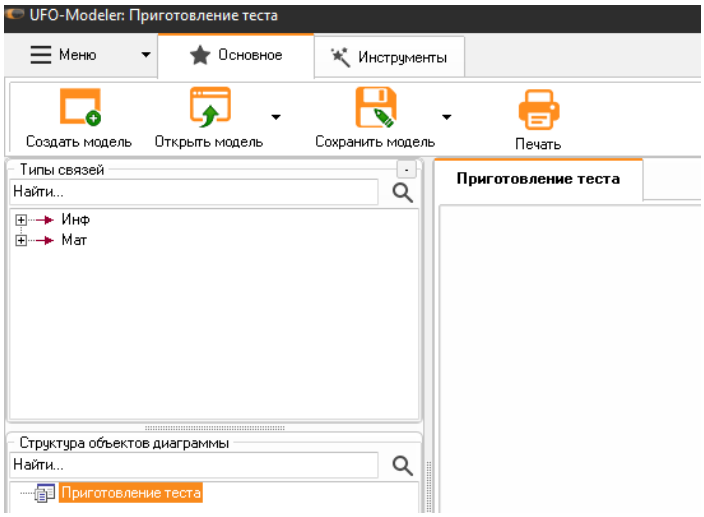

Рис. 4. Вкладка «Основное»

На вкладке «Основное» доступны функции создания, открытия, сохранения и печати модели. На вкладке «Инструменты» доступны функции создания узлов, связей, а также запуска и экспорта модели.

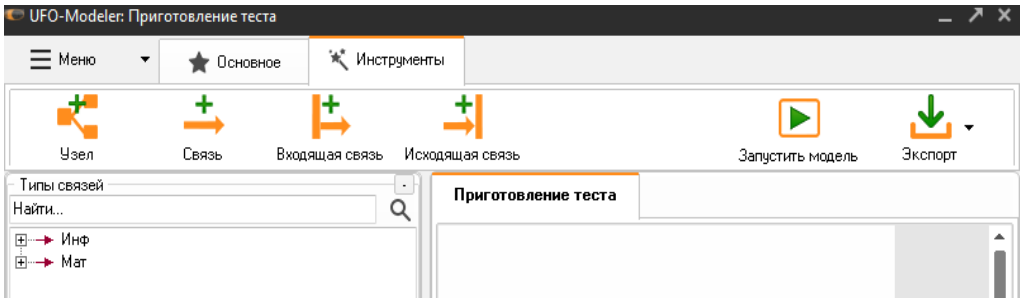

Рис. 5. Вкладка «Инструменты»

Рассмотрим построение модели простого процесса – приготовления теста. Начнем создание модели с добавления новых связей. В УФО-подходе различают следующие типы связей:

– материальная связь (вещественная или энергетическая);

– информационная связь (по данным – для передачи данных, по управлению – для передачи управляющей информации).

Добавим новые связи. Для модели «Приготовление теста» необходимы только материальные связи. В меню «Типы связей» нажать правой кнопкой на типе связи «Вещественный» и выбираем пункт «Добавить вложенный элемент» (рисунок 6).

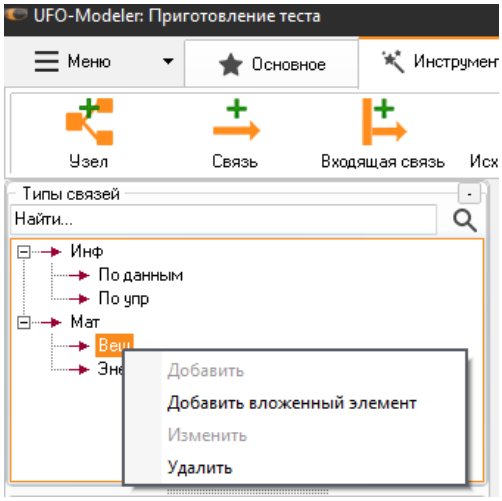

Рис. 6. Добавление связи

На рисунках 7 - 8 представлены примеры заполнения параметров при создании связи. Во вкладке «Общие» задается наименование и описание связи.

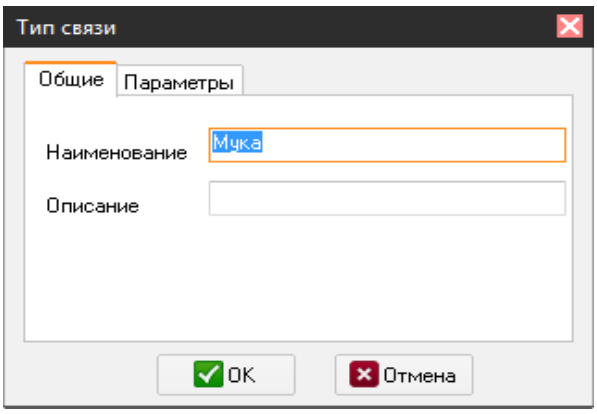

Рис. 7. Вкладка «Общие» параметры вещественной связи «Мука»

Во вкладке «Параметры» добавляем тип и наименование параметров связи (к примеру, параметр «Вес» в граммах).

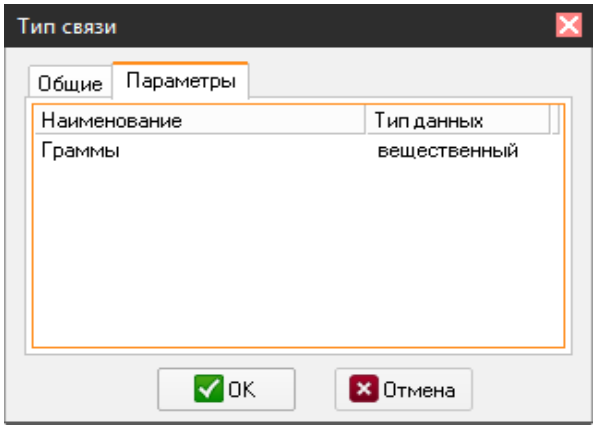

Рис. 8. Вкладка «Параметры» вещественной связи «Мука»

Таким образом, создаем все типы связей, представленные на рисунке 9.

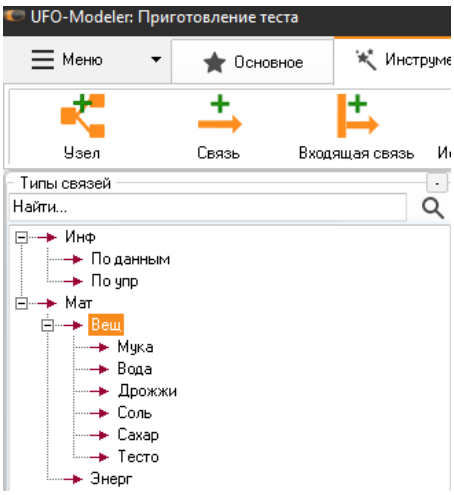

Рис. 9. Типы связей

Следующий этап – создание узла. Узел представляет собой перекресток для связей, в этот перекресток поступают материальные и информационные связи, которые преобразуются в выходящие связи.

Создадим узел, преобразовывающий входные связи в выходные, и назовем его «Смешивание ингредиентов». Для этого на вкладке «Инструменты» нажимаем кнопку «Узел», и в диалоговом окне задаем название и размеры узла (рисунок 10).

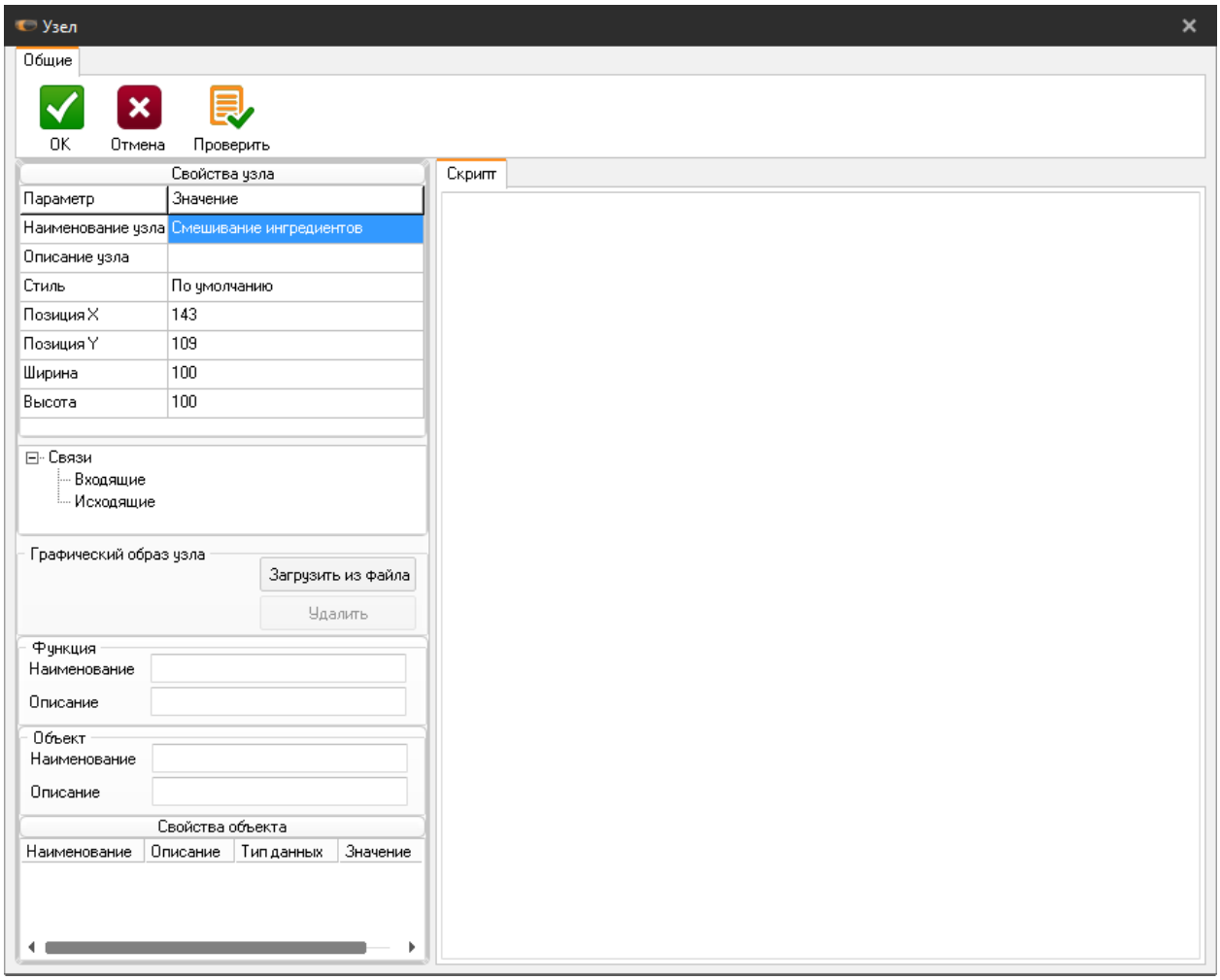

Рис. 10. Создание узла

В левой колонке можно определить свойства узла: наименование, описание, стиль, позицию на рабочем поле, размеры узла, загрузить графический образ узла. Правая колонка предназначена для задания УФО-скрипта.

Узел создан (рисунок 11). При необходимости можно изменить положение узла на модели и его свойства.

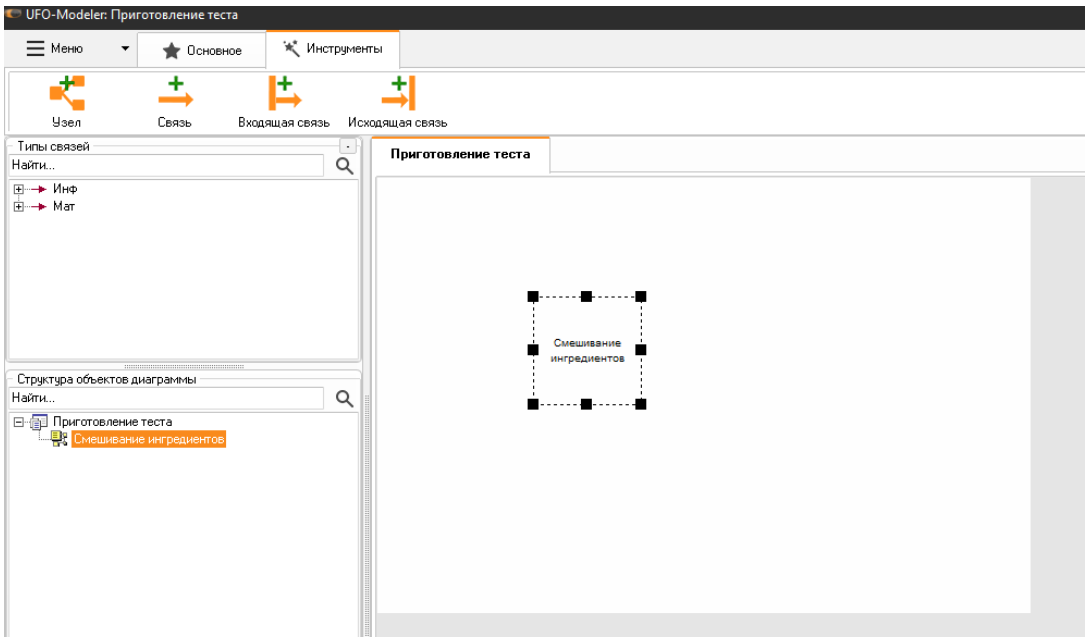

Рис. 11. Узел «Смешивание ингредиентов»

Теперь необходимо добавить связи на модель. Начнем с входящих связей. Для добавления выбираем нужную связь в окне «Типы связей», нажимаем на кнопку «Входящая связь» пункта меню «Инструменты» и выбираем узел, для которого эта связь станет входящей. Модель приобретает вид, представленный на рисунке 12. При необходимости положение стрелок и названий можно скорректировать после нажатия на требующую корректировки связь.

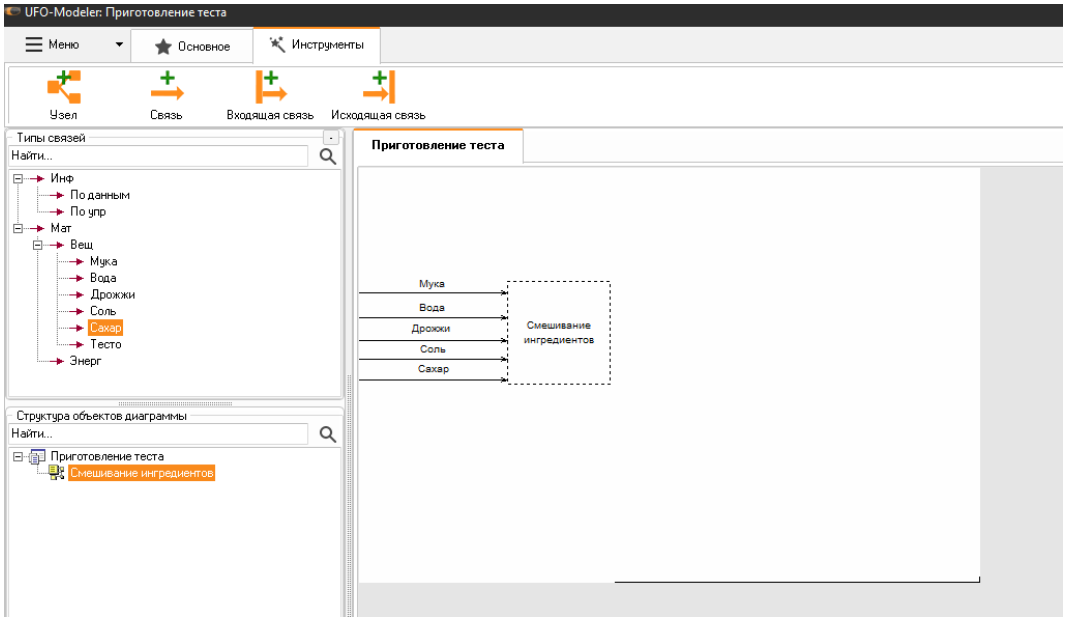

Рис. 12. Узел и входящие связи

Теперь создадим исходящую связь «Тесто». Необходимо выбрать связь в окне «Типы связей», нажать кнопку «Исходящая связь» в окне инструментов и выбрать узел,

для которого связь будет исходящей. Таким образом модель «Приготовление теста» построена и представлена на рисунке 13.

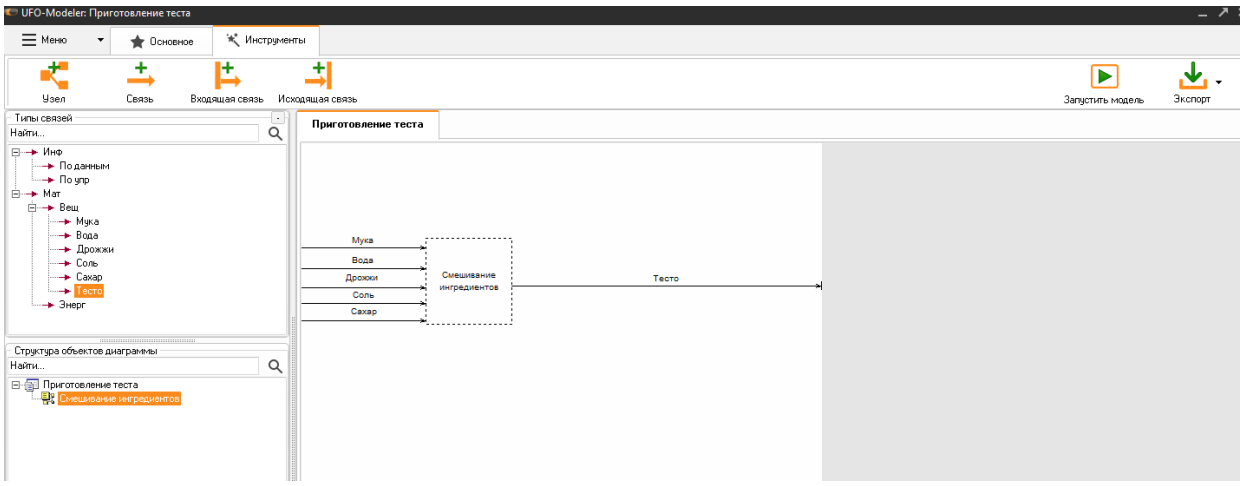

Рис. 13. Модель «Приготовление теста»

Диаграмму можно декомпозировать до требуемого уровня, то есть исследуемый процесс разбивается на подпроцессы. Для того, чтобы декомпозировать какой-либо узел, необходимо щелкнуть на этом узле правой кнопкой мыши и в появившемся меню выбрать пункт «Создать вложенный узел». После этого появляется окно создания узла, задаем необходимые параметры, и узел появляется на диаграмме декомпозиции. Таким образом, добавляем необходимые узлы, и соединяем связи с соответствующими узлами.

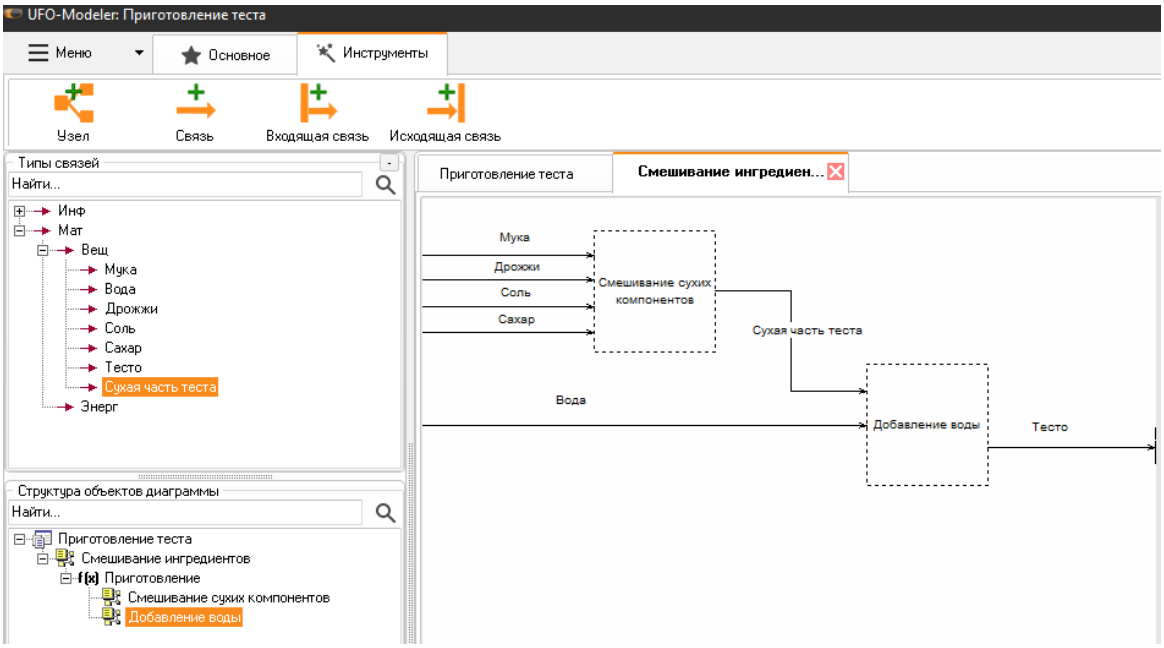

Рис. 14. Декомпозиция модели «Приготовление теста»

На рисунке 14 представлена декомпозиция модели. Процесс «Приготовление теста» был разбит на 2 этапа, то есть созданы 2 новых узла – «Смешивание сухих

компонентов» и «Добавление воды». При этом была добавлена новая связь между узлами «Сухая часть теста». Эту связь между элементами устанавливается путем использования инструмента «Связь» панели инструментов, после выбора связи и инструмента «Связь» необходимо выбрать узел, откуда связь выходит, и узел, для которого связь является входящей. После построения сохраним модель (Меню –> Сохранить –> Сохранить в файл).

3.2. Создание решения 2

С использованием UFO-скрипта попробуем описать процесс сложения целых чисел. Создадим информационные связи X, Y, Z и сумма, для каждого определим параметр «Значение» и тип данных – целый. Добавим узел «Сложение», для которого X, Y, Z будут входящими, а «Сумма» - исходящей связью. Результат представлен на рисунке 15.

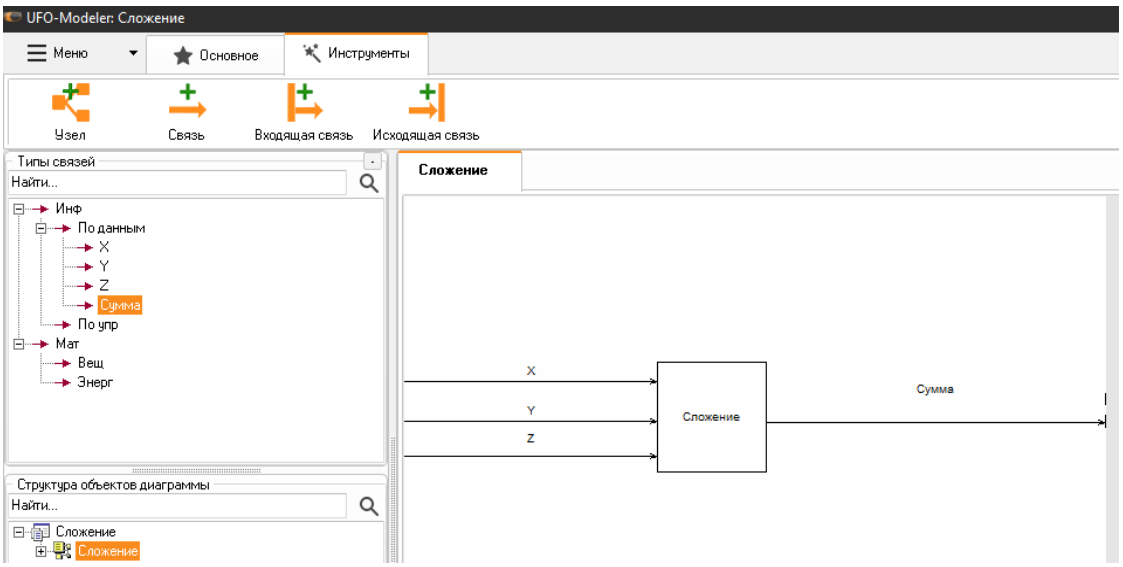

Рис. 15. Модель «Сложение»

Теперь перейдем к свойствам узла (Правая кнопка мыши на узле -> Свойства узла). В открывшемся окне мы можем задать свойства узла, видим перечень входящих и исходящих связей, в поле «Скрипт» можно описать функцию узла с использованием специального скрипта.

Введем скрипт, представленный на рисунке 16, в предназначенное для этого поле. При этом ссылки на связь можно добавить путем двойного щелчка на параметр связи в окне «Связи» в левой части окна.

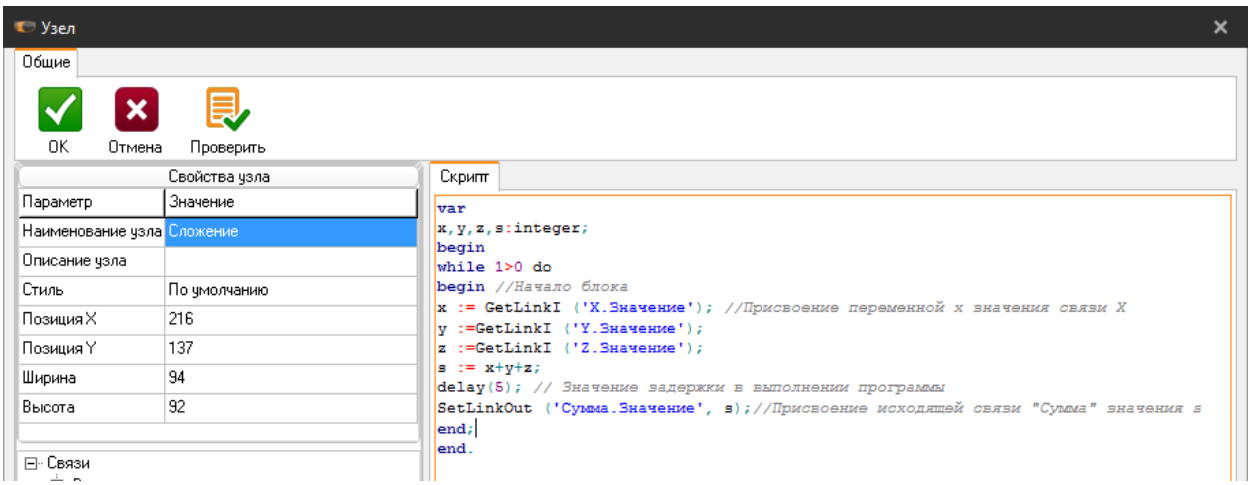

Рис. 16. Пример скрипта для сложения

После этого проверим введенный скрипт на наличие ошибок (Кнопка «Проверить» в верхней части окна). Поле наименования функции должно быть заполнено в поле «Функция» (рисунок 17). В случае если компиляция выполнена успешно, и наименование функции введено, нажимаем кнопку ОК.

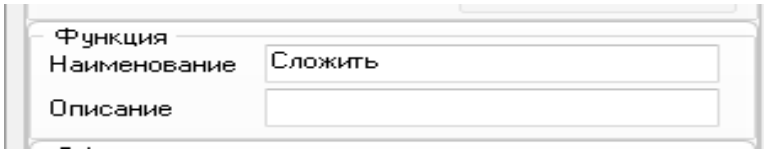

Рис. 17. Наименование функции

Зададим значения параметров для входящих связей. Для этого на модели двойным щелчком по связи вызываем меню свойств связи, переходим во вкладку «Параметры» и задаем значение параметров (рисунок 18).

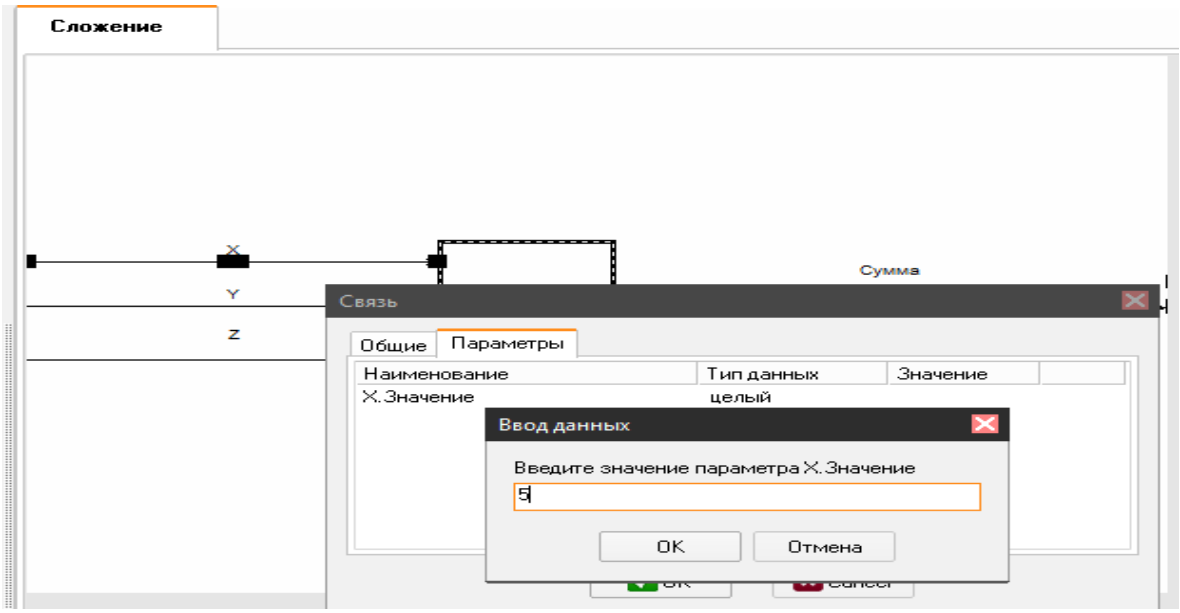

Рис. 18. Параметры связи

Попробуем запустить модель (Вкладка «Инструменты» -> Запустить модель). В появившемся окне для всех связей (X, Y, Z, Сумма) необходимо поставить галочку возле параметра «Значение» (рисунок 19). Таким образом, при запуске модели все количественные характеристики будут представлены на экране.

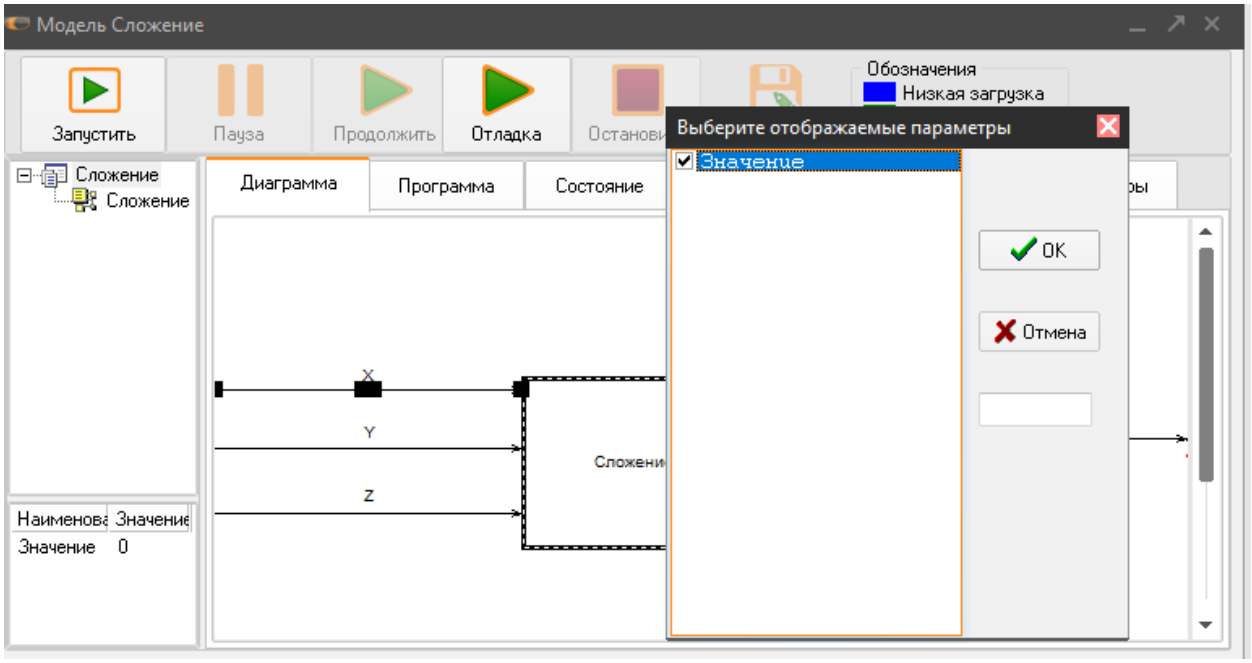

Рис. 19. Выбор отображаемых параметров

После настроек можем запустить модель (Кнопка «Запустить» в левом верхнем углу окна). После запуска изначально значение параметра «Сумма» равно нулю, так как в скрипте задана системная задержка (Рисунок 20).

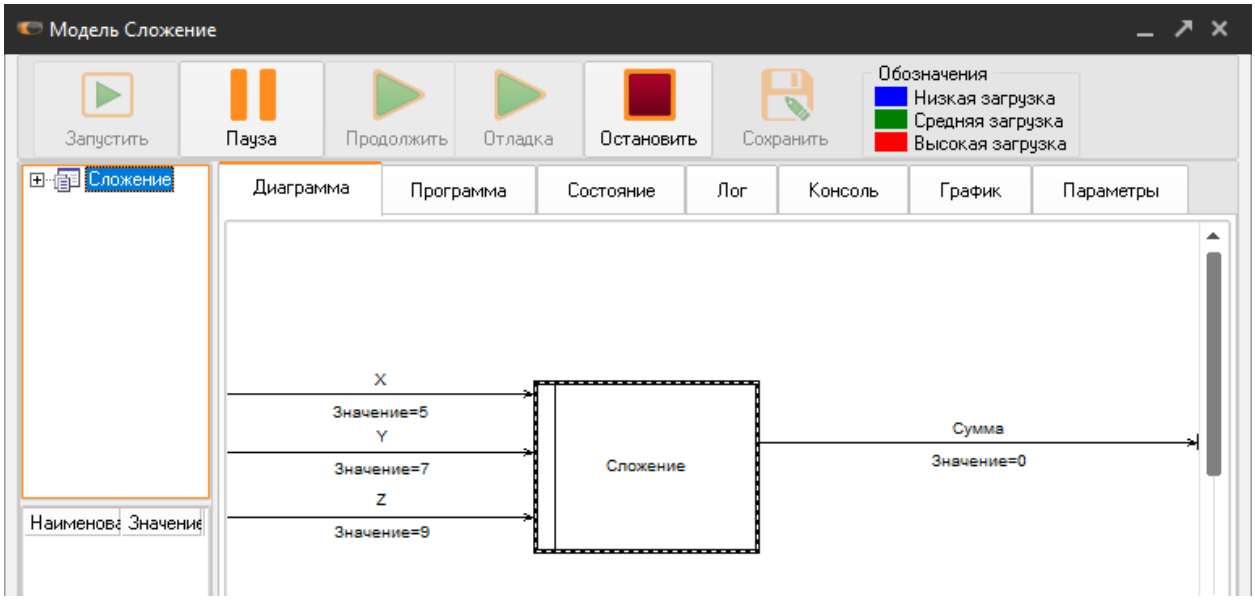

Рис. 20. Запуск модели

Изменение параметра «Сумма» происходит сразу после истечения системной задержки. Результаты запуска модели представлены на рисунке 21.

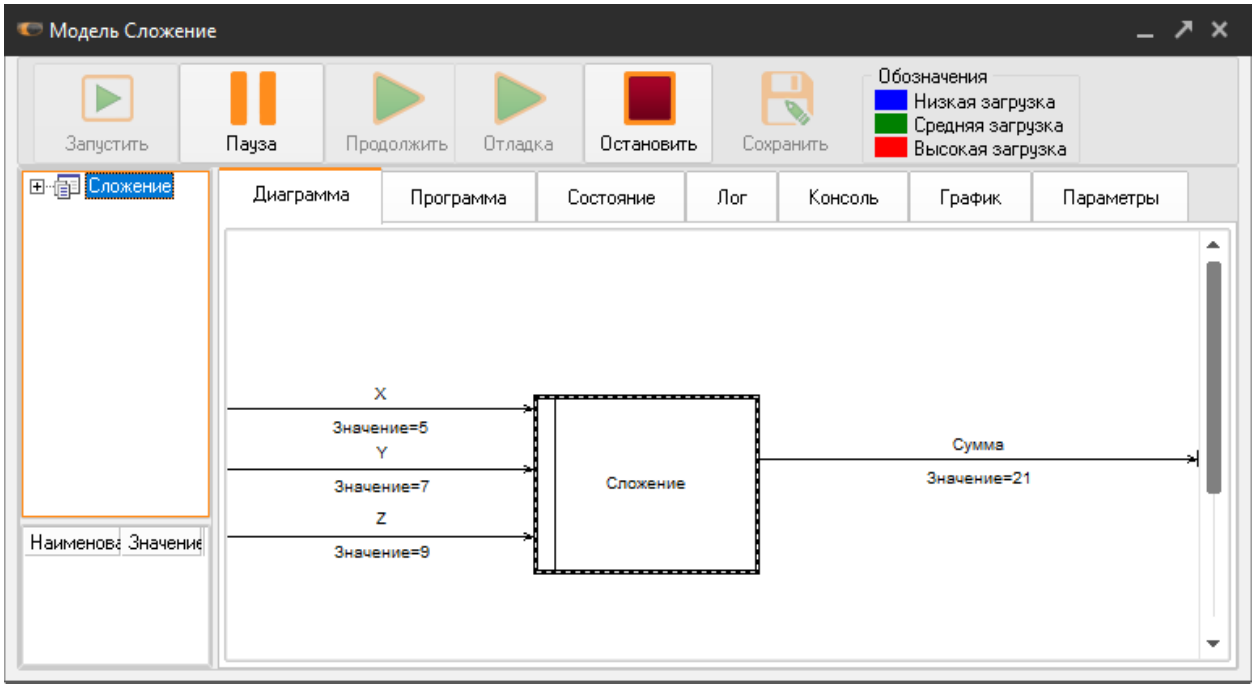

Рис. 21. Результат работы модели

Теперь рассмотрим окно запуска модели более подробно. В верхней части окна расположена панель, которая содержит следующие элементы:

– кнопка «Пауза» - приостановить выполнение программ модели;

– кнопка «Продолжить» - продолжить выполнение программ модели и расчет модельного времени;

– кнопка «Отладка» - особый режим, продолжает выполнение программ модели на одну итерацию алгоритма, а также позволяет проанализировать выполненные на текущем шаге действия программ;

– кнопка «Остановить» - выполнение программ модели прекращается. В дальнейшем продолжить выполнение программ невозможно, будет доступна только кнопка «Запустить», которая начинает выполнение программ модели заново (с инициализацией значений переменных, параметров связей и свойств объектов).

На вкладке «Диаграмма» отражается диаграмма узлов модели. Текст программы при выборе узла отображается во вкладке «Программа». Вкладка «Состояние» предназначена для отображения значений всех свойств узлов, параметров связей и переменных из скриптов каждого узла. Вкладка «Лог» предназначена для вывода сообщений об ошибке в работе скриптов, вкладка «Консоль» выводит сообщения

процедур. Во вкладке «Параметры» задаются параметры исполнения скриптов модели – масштаб времени, время интегрирования, загрузка объекта.

Вкладка «График» делится на 2 вкладки «Настройка» и «График». Наборы данных для вывода на графике необходимо выбрать во вкладке «Состояние».

Создадим модель для вычисления системы уравнений (1) (рисунок 22).

$$y = \begin{cases} \frac{x^2}{2}, x - четное \\ -\frac{x^2}{2}, x - нечетное \end{cases} \quad (1)$$

Алгоритм вычисления значения функции Yсостоит в следующем: если при решении системы (1) целая часть числа X является четной, то вычислим значение функции Y по формуле из первой строки, если целая часть числа X является нечетной, то вычисление произведем по формуле второй строки. Для связей «аргумент» и «результат» зададим параметр «X» и соответственно «Y» с вещественным типом данных и отображением 2 знаков после запятой.

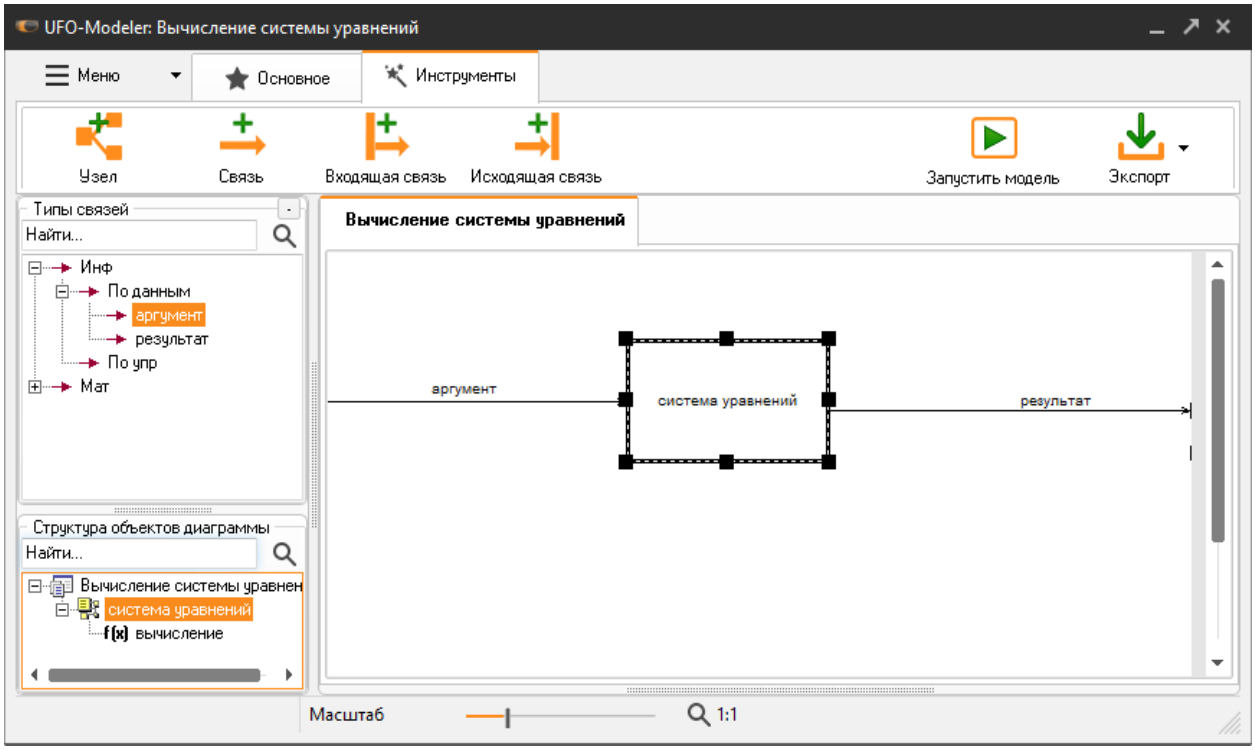

Рис. 22. Модель вычисления системы

УФО-скрипт представлен на рисунке 23. Решение системы (1) будет состоять в бесконечном цикле с шагом 0,1. В процессе работы скрипта будет происходить чередование сначала формулы первой строки, так как целая часть аргумента на отрезке

[0..1) равна 0 и является четной, затем формулы второй строки, так как целая часть X на отрезке [1..2) равна 1 и является нечетной.

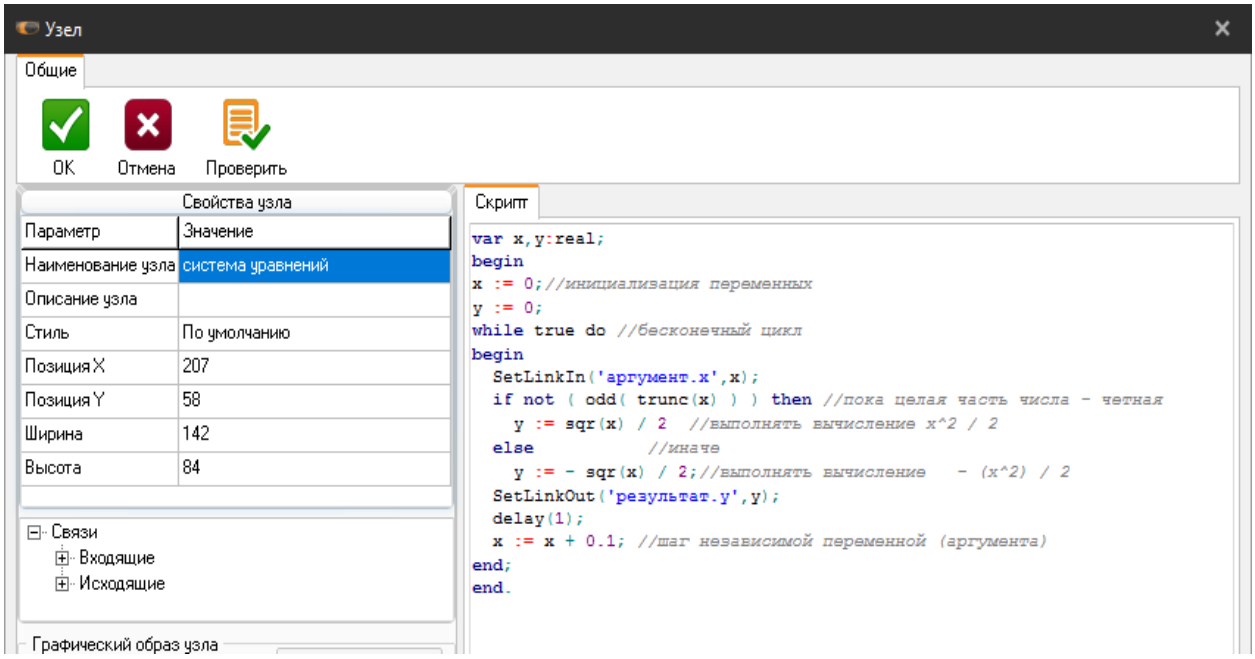

Рис. 23. УФО-скрипт вычисления системы

После проверки скрипта запускаем модель, при этом параметры значения X и Y должны быть отображаемы. Результат работы программы представлен на рисунке 24.

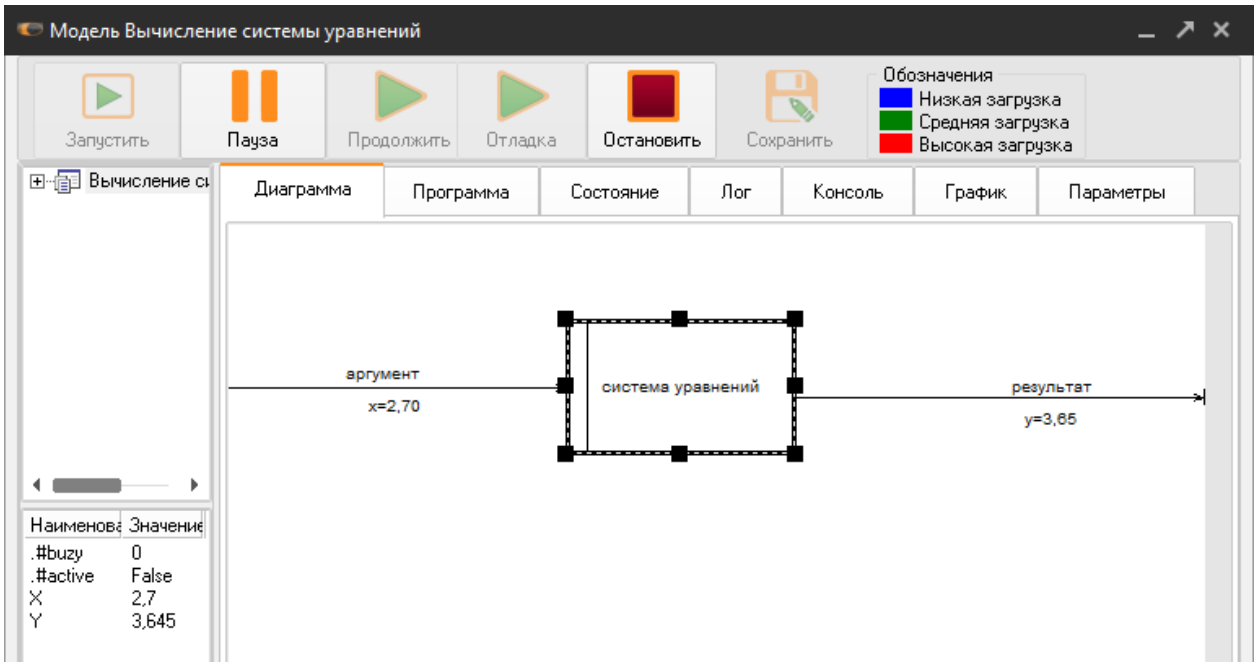

Рис. 24. Вычисление системы

В окне запуска модели перейдите на вкладку «Диаграмма», выберите узел «Система уравнений» левой кнопкой мыши, нажмите кнопку «Отладка» при этом произойдет выполнение одного очередного шага программ модели.

Перейдите на вкладку «Программа» при этом отобразится программный код функции «Вычисление». На данной вкладке отображено место останова функции текущего узла. Программный код, выделенный синим цветом, отражает выполненный участок функции узла. Соответственно, участок программного кода без выделения показывает, какие команды будут выполнены после продолжения работы модели. Исходя из рисунка 25, можно отметить приостановку алгоритма функции на операторе delay(1) – задержка выполнения алгоритма (пауза).

Произведите повторное нажатие кнопки «Отладка» (узел «Система уравнений» должен быть выделен), пронаблюдайте за изменением результата.

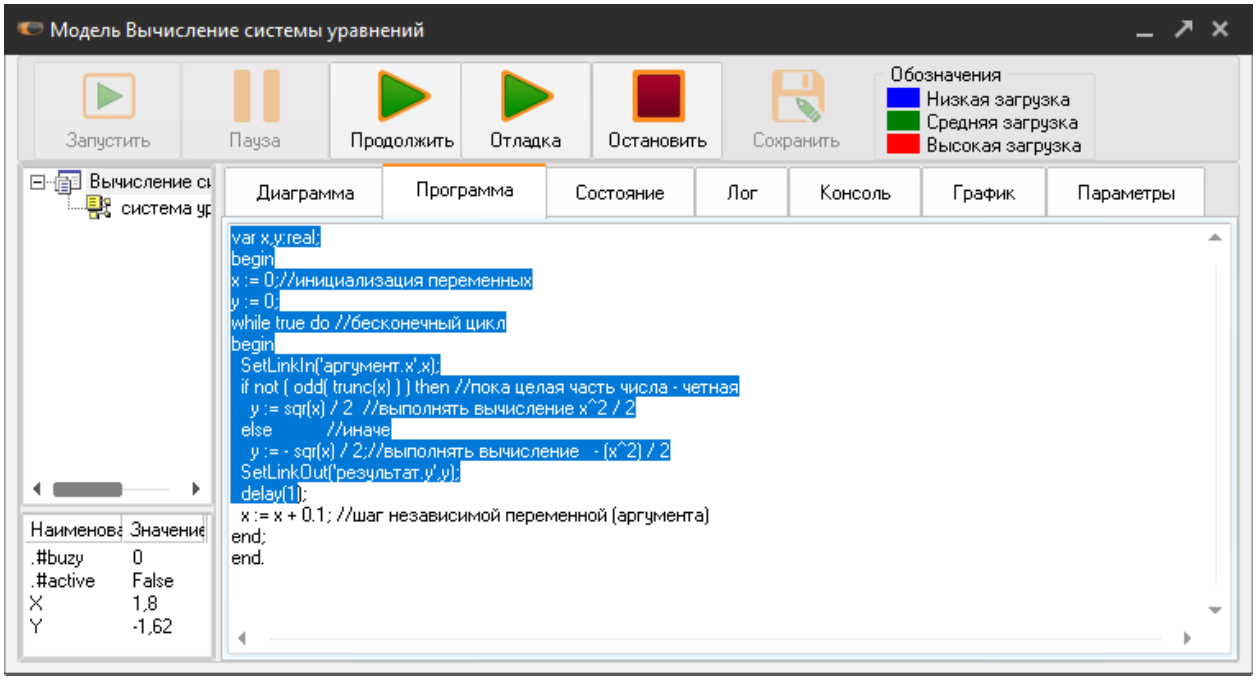

Рис. 25. Программный код функции «Вычисление» в режиме «Отладка»

Перейдите на вкладку «Консоль» при этом отобразится список действий, произошедших на текущем шаге работы всех программ модели. Данная функция необходима для отслеживания порядка выполнения команд, которые выполняются в функциях узлов. На рисунке 26 отображены действия над переменными, вычисляющими значения переменной Y в модели, произошедшие задержки оператора delay(), а также очередность выполнения команд модели «Вычисление системы уравнений». Проанализируйте другие действия, произошедшие в процессе выполнения модели (рисунок 26).

Произведите повторное нажатие кнопки «Отладка», пронаблюдайте за действиями, выполненными на текущем шаге работы модели.

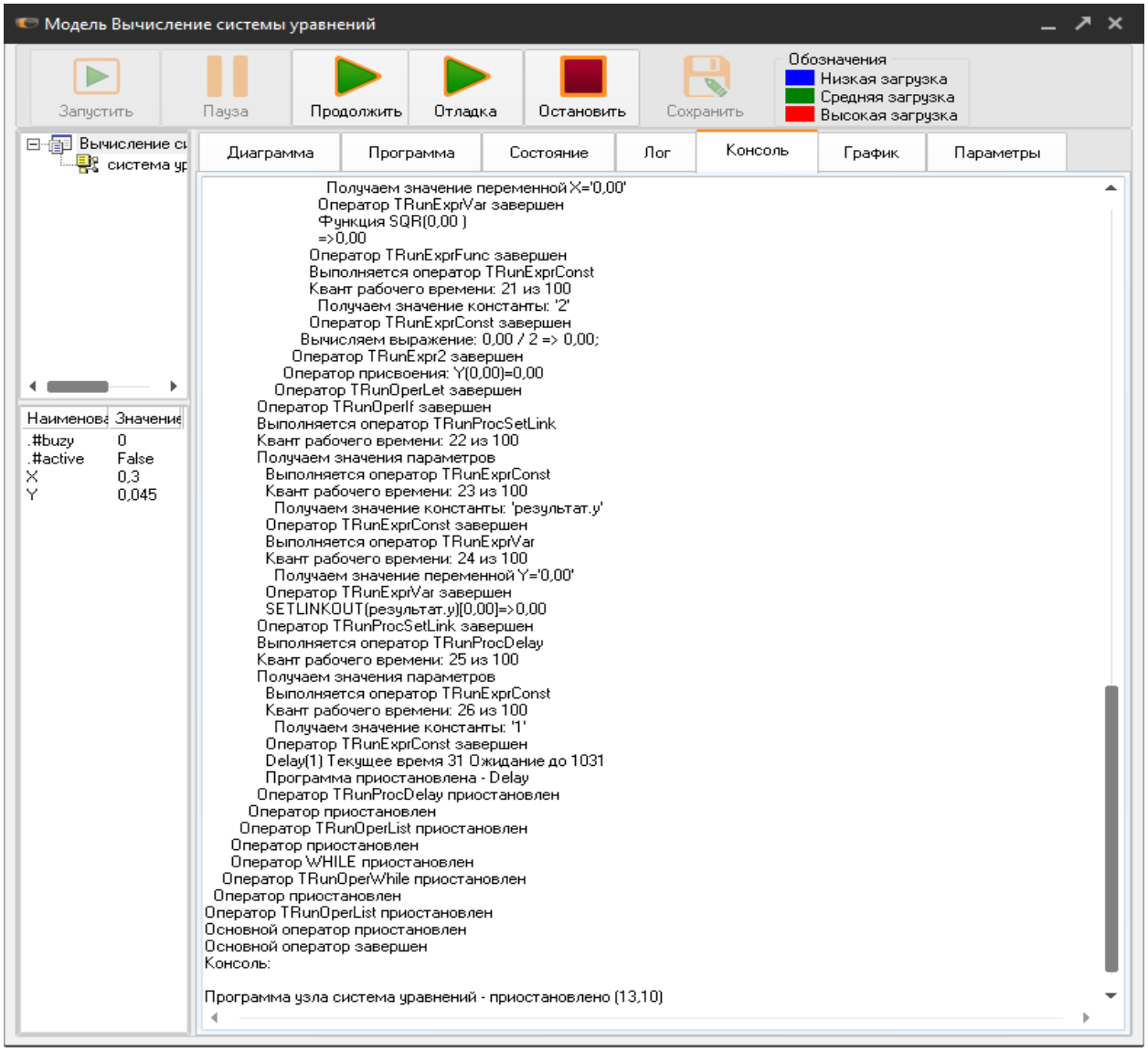

Рис. 26. Список выполненных действий программ модели

Перейдем во вкладку «Состояние» и отметим параметры значения X и Y (рисунок 27).

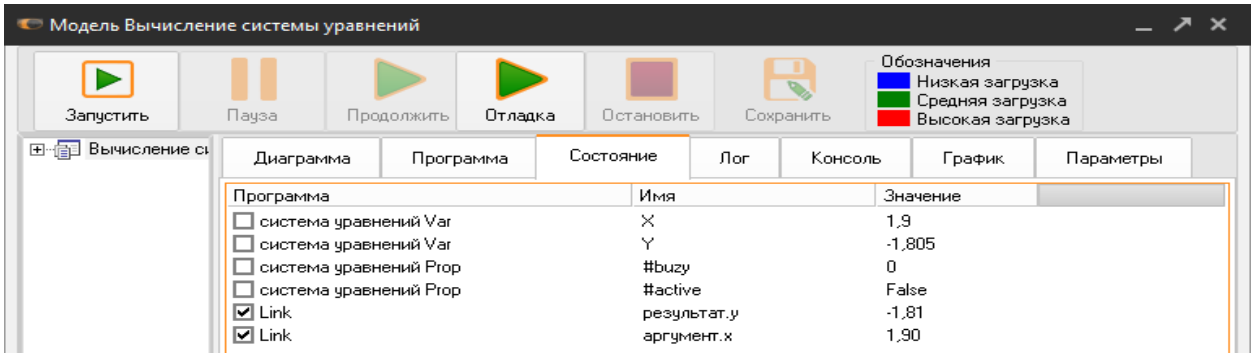

Рис. 27. Вкладка «Состояние»

Затем перейдем во вкладку «График». В настройках способа построения графика выберем пункт «Зависимость первого набора данных от второго» (рисунок 28).

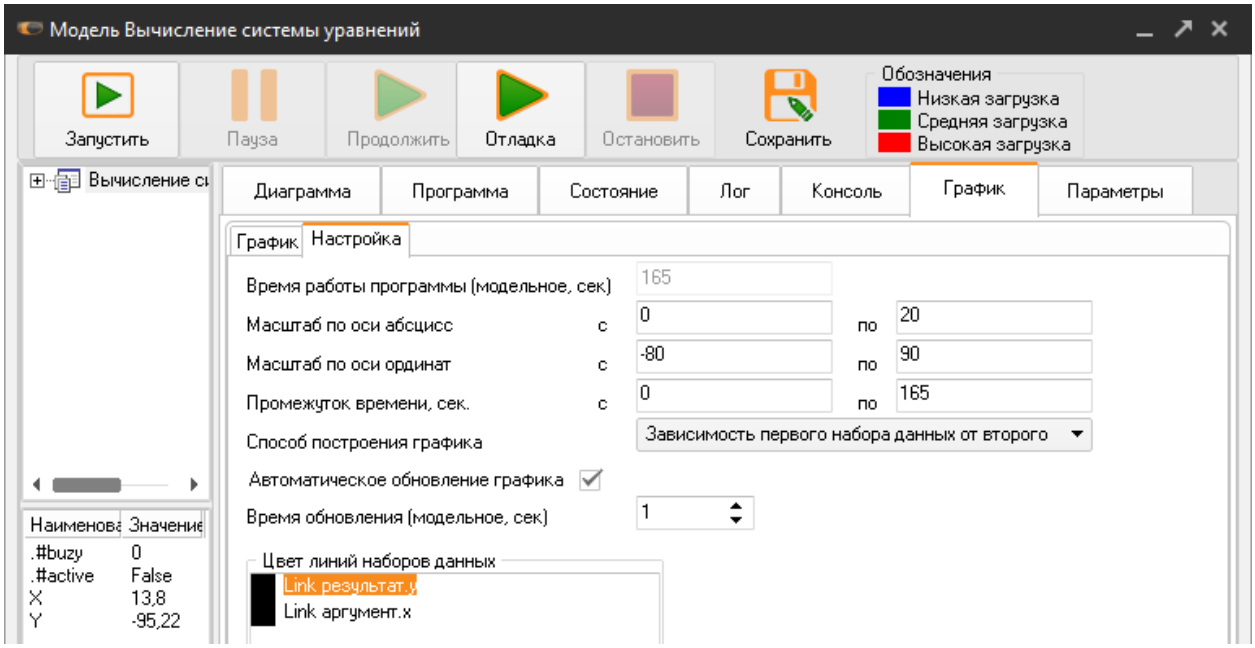

Рис. 28. Настройка отображения графика

Во вкладке график появляется построенный график зависимости значения Y от значения X (рисунок 29).

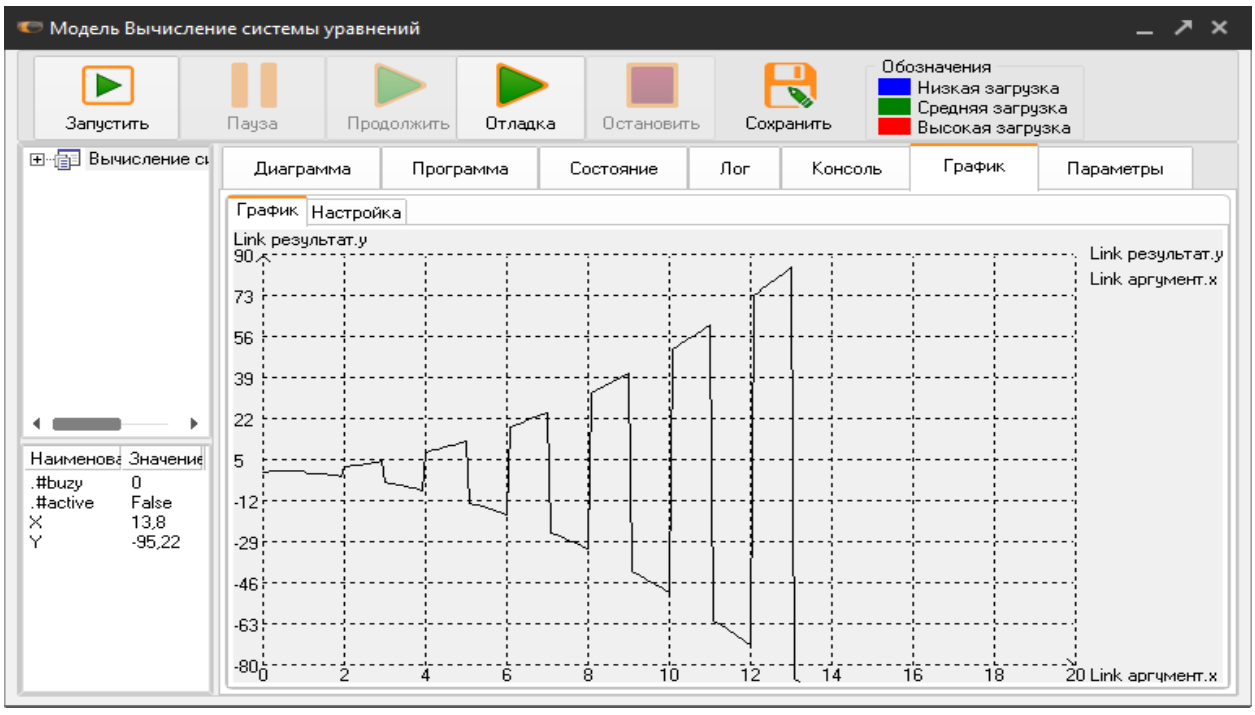

Рис. 29. График зависимости Y от X

Попробуйте изменить настройки отображения графика и проанализировать получившийся результат.

3.3. Создание решения 3

В банковском отделении находится банкомат, который предназначен для быстрого обслуживания посетителей банка.

Построим с помощью программного обеспечения UFOModeler модель простой системы массового обслуживания – модель банковского отделения, в котором через фиксированные промежутки времени в очередь поступают клиенты (2 условные единицы времени).

Нажмите на кнопку «Создать новую модель» при этом отобразится всплывающее окно, в котором необходимо указать параметры модели, такие как название, описание и размеры диаграммы. При необходимости параметры модели можно изменить нажатием правой кнопки мыши в поле диаграммы и выбора пункта «Свойства модели» в контекстном меню.

Введите название модели «Отделение банка» нажмите кнопку «ОК».

Главное окно программы представлено на рисунке 30.

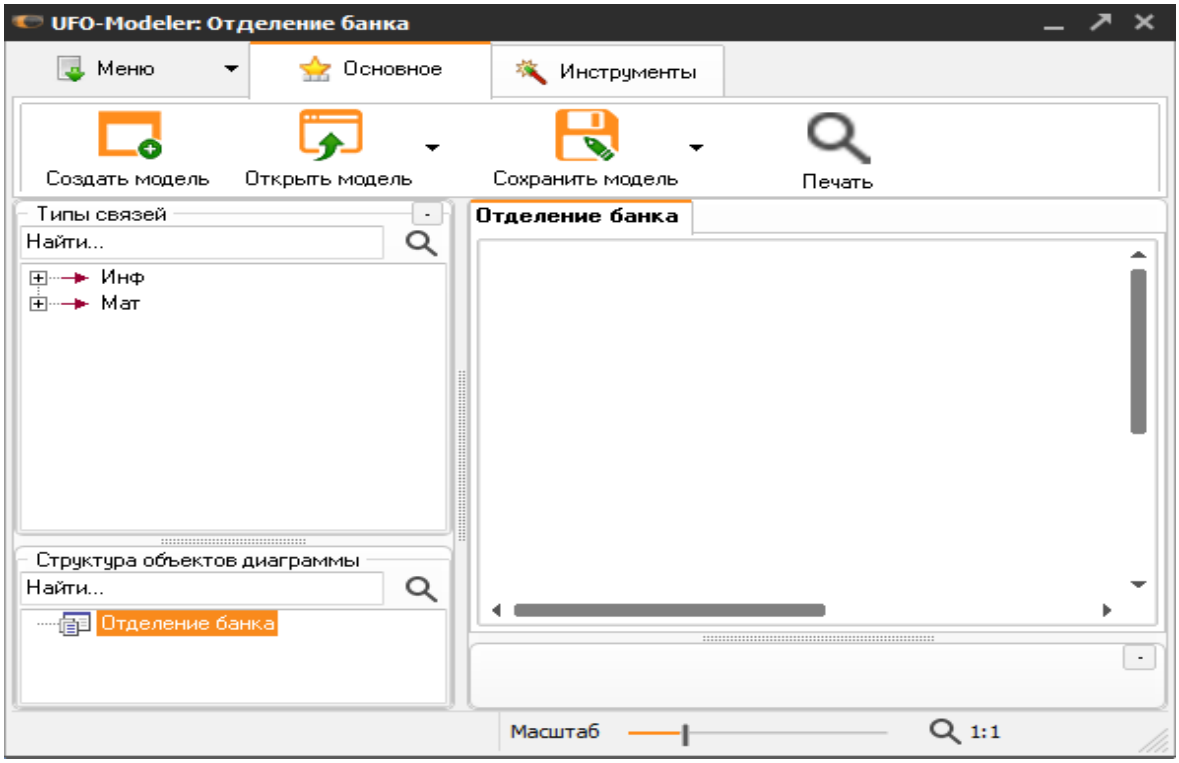

Рис. 30. Главное окно программы

В Главном меню на вкладке «Инструменты» выберите пункт «Узел». Создайте новый узел на схеме. В меню «Свойства узла» введите наименование узла «Очередь». Добавьте описание узла. В случае необходимости скорректируйте положение узла и его размеры на схеме.

Установите «Графический образ узла» с помощью кнопки «Загрузить из файла» установите графическое изображение, представляющее узел на схеме.

Аналогичным образом добавьте узел «Транзакция клиента».

На рисунке 31 представлен результат создания объектов модели.

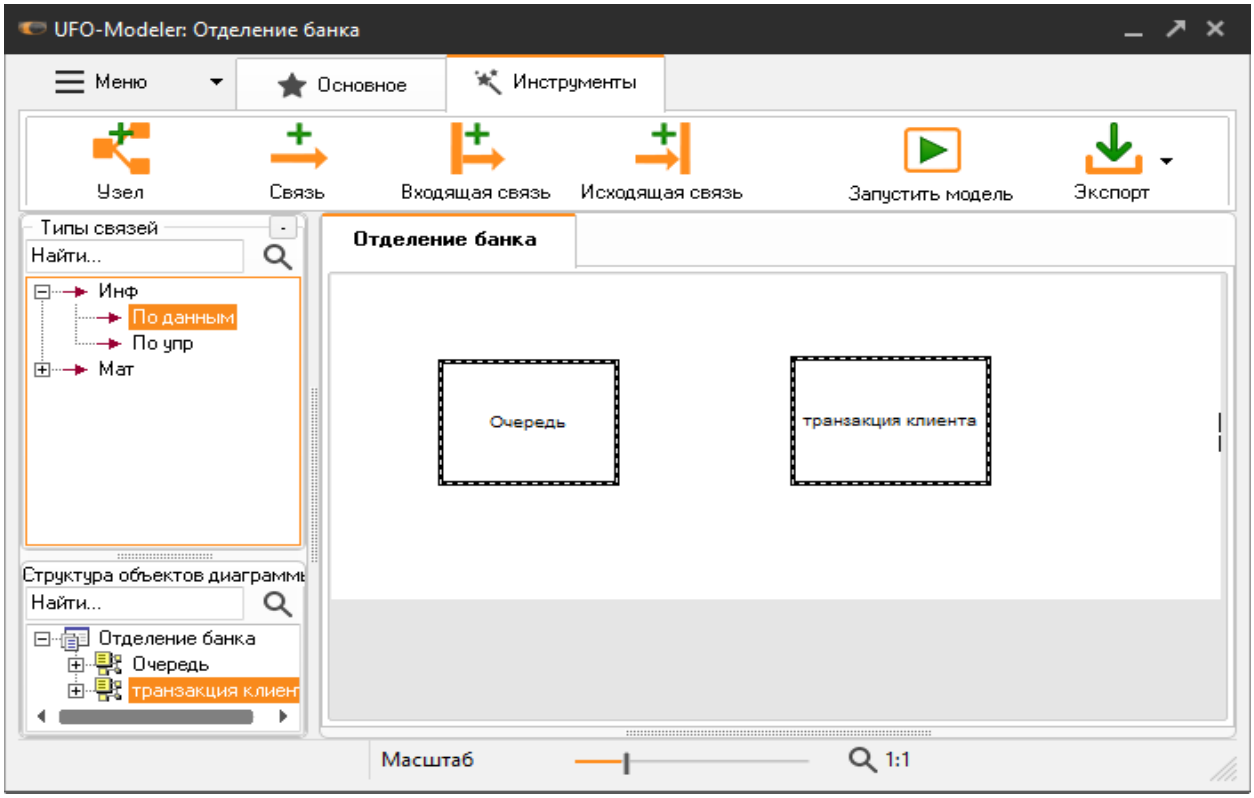

Рис. 31. Добавление объектов на схему

В меню «Типы связей» добавьте новый тип связи в разделе «Инф» – «По данным» для этого в контекстном меню выберите пункт «Добавить вложенный элемент» при этом отобразится всплывающее окно.

Во всплывающем окне на вкладке «Общие» введите название связи «клиенты». На вкладке «Параметры» в поле списка нажмите правой кнопкой мыши и в контекстном меню выберите пункт «Создать». В новом окне введите наименование «ожидающие» и тип данных «целый». Нажмите кнопку «ОК» для того, чтобы добавить новую связь.

На рисунке 32 представлено меню с результатом добавления связи.

Аналогичным образом добавьте следующие параметры для типа связи «клиенты»:

– «обслуженные» – тип данных «целый»;

– «срскорость» – тип данных «вещественный».

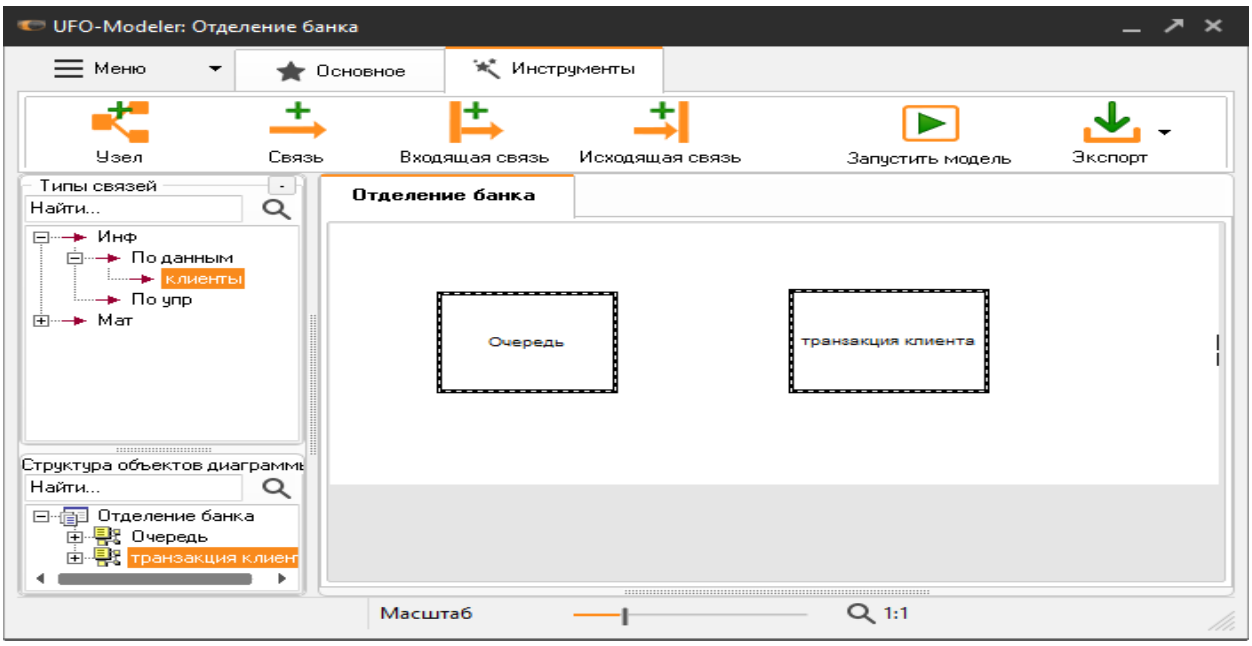

Рис. 32. Окно «Типы связей»

Для того чтобы настроить связи между узлом «Очередь» и узлом «Транзакция клиента» выберите в меню пункт «клиенты» и последовательно нажмите левой кнопкой мыши на левый и правый узел при этом на экране появится стрелка, обозначающая связь узлов.

На рисунке 33 представлен результат добавления связи на диаграмму.

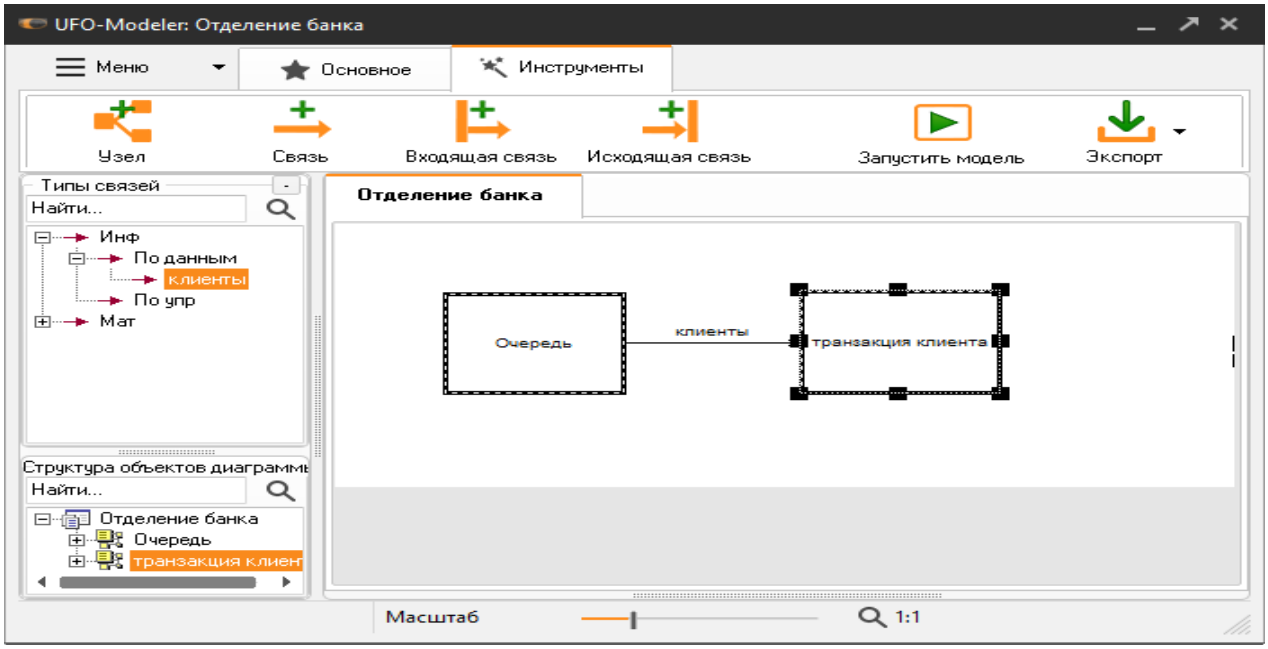

Рис. 33. Представление связей узлов на схеме

При двойном клике левой кнопкой мыши по объекту или при выборе в контекстном меню объекта «Свойства узла» отобразится окно редактирования узла. Добавьте следующий код функции для узла «Очередь» как представлено на рисунке 34.

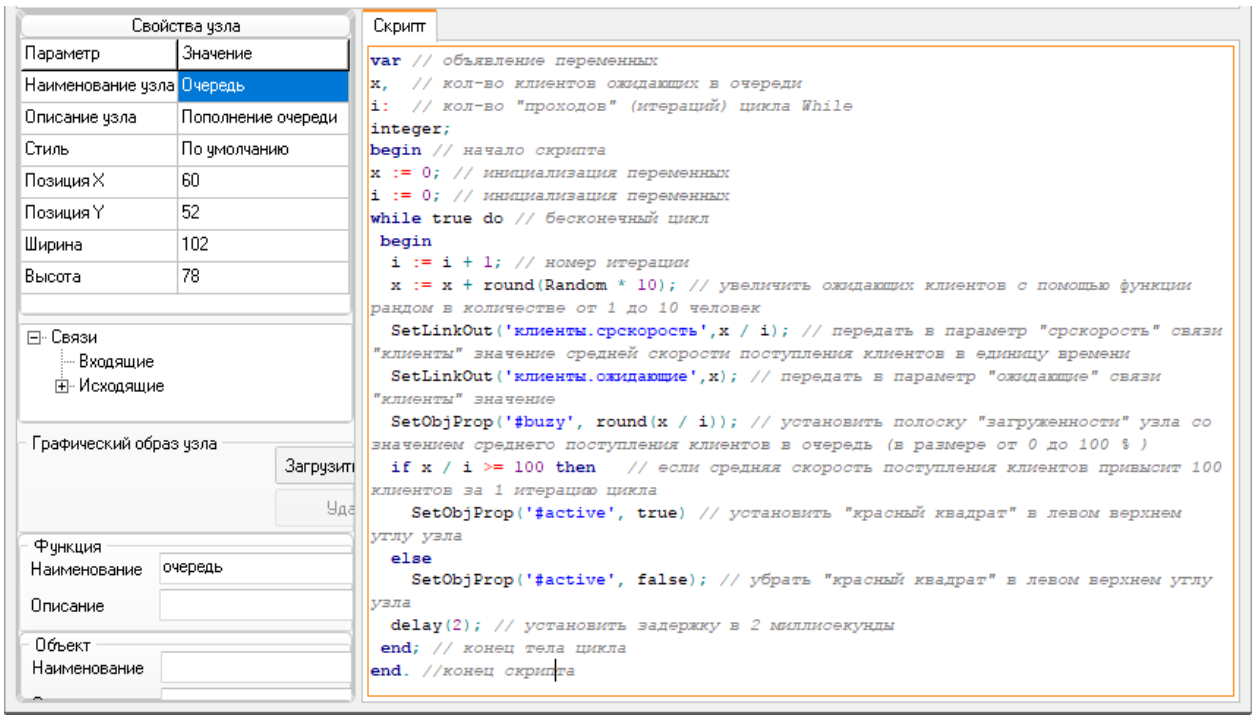

Рис. 34. Код функции узла «Очередь»

Добавьте следующий код функции для узла «Транзакция клиента» как представлено на рисунке 35.

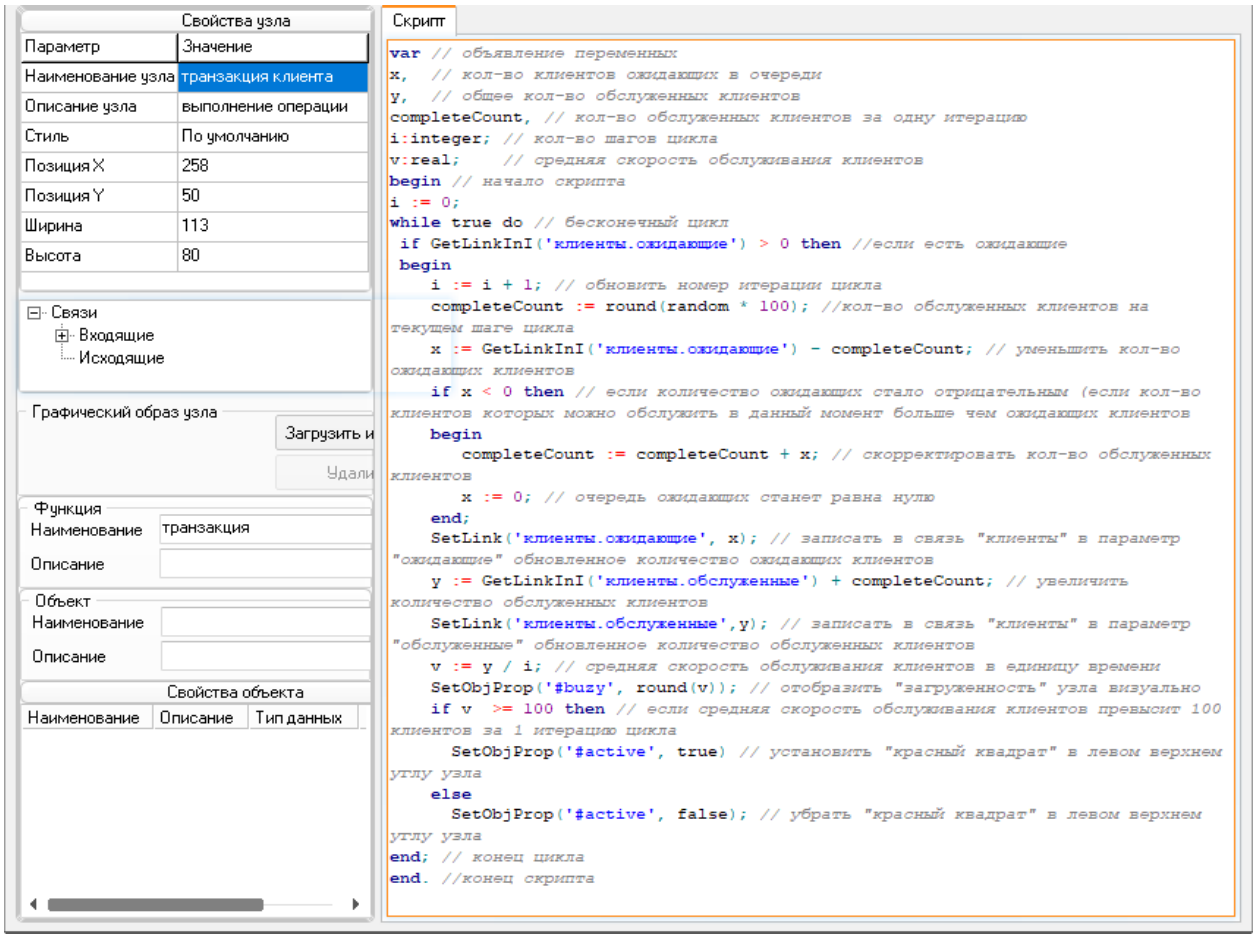

Рис. 35. Код функции узла «Транзакция клиента»

Нажмите кнопку «Запустить модель» на вкладке «Инструменты». В окне функционирования модели на вкладке «Параметры» выберите из выпадающего списка «Масштаб времени» значение «2:1». Пронаблюдайте работу модели.

Измените «Масштаб времени» и время задержки в функциональных скриптах узлов в операторе delay(). Пронаблюдайте за изменением работы модели.

Постройте график, на котором отображено количество ожидающих и обслуженных клиентов. Для этого произведите настройку графика согласно рисунку 36.

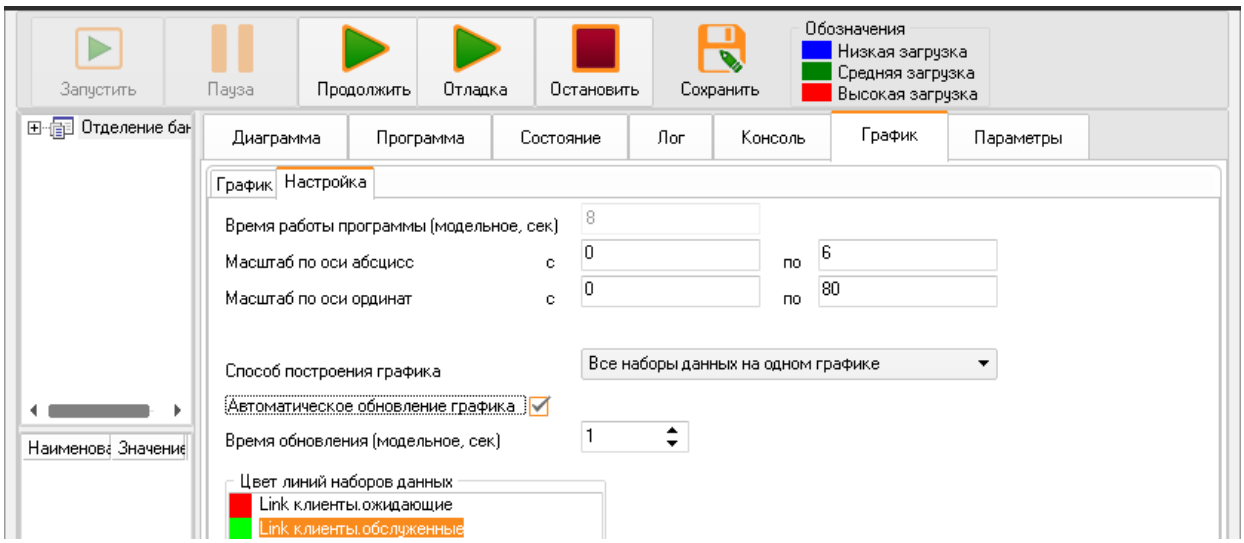

Рис. 36. Код функции узла «Транзакция клиента»

Пронаблюдайте построение графика на вкладке «График».

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ 1

Таблица 1 Операторы

Наименование		Описание
begin Оператор 1; Оператор 2; … Оператор N **end**		Составной оператор
if Условие **then** Оператор	Условие:boolean – логическое выражение	Условный оператор (сокращенная форма)
if Условие **then** Оператор1 **else** Оператор2	Условие:boolean – логическое выражение	Условный оператор (полная форма)
while Условие **do** Оператор	Условие:boolean – логическое выражение	Цикл с предусловием: выполняется пока Условие дает результат True.
repeat Оператор 1; Оператор 2; … Оператор N **until** Условие	Условие:boolean – логическое выражение	Цикл с пост условием: выполняется до тех пор, пока Условие не даст результат True
for Переменная := Значение1 **to** / **downto** Значение2 **do** Оператор	Переменная – счетчик цикла; Значение1 – начальное значение; Значение2 – конечное значение.	Цикл с параметрами
case Выражение **of** Метка1: Оператор1; Метка2: Оператор2;	Выражение – выражение, по значению которого выполняется	Оператор выбора. Выполняется оператор в зависимости от значения

… МеткаN: ОператорN; **else** ОператорElse 1; ОператорElse 2; … ОператорElse N **end**	переход к оператору с соответствующей меткой	выражения. Если не совпало ни одно значение, то выполняется список операторов в разделе между служебными словами else и end.

Таблица 2 Процедуры и функции – работа со свойствами текущего узла

Наименование	Параметры	Описание
GetObjPropI (propertyName:string):integer;	propertyName – наименование свойства	Получить значение свойства. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы.
GetObjPropF (propertyName:string):real;	propertyName – наименование свойства	Получить значение свойства. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы.
GetObjPropS (propertyName:string):string;	propertyName – наименование свойства	Получить значение свойства. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы.
GetObjPropB (propertyName:string):boolean;	propertyName – наименование свойства	Получить значение свойства. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы.
SetObjProp	propertyName –	Установить новое

(propertyName:string; value: integer);	наименование свойства; value – устанавливаемое значение	значение свойству. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы.
SetObjProp (propertyName:string; value: real);	propertyName – наименование свойства; value – устанавливаемое значение	Установить новое значение свойству. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы.
SetObjProp (propertyName:string; value: integer; time:integer);	propertyName – наименование свойства; value – устанавливаемое значение; time – время, в течение которого должно установиться значение	Установить новое значение свойству. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы.
SetObjProp (propertyName:string; value: real; time:integer);	propertyName – наименование свойства; value – устанавливаемое значение; time – время, в течение которого должно установиться значение	Установить новое значение свойству. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы.
SetObjProp (propertyName:string; value: string);	propertyName – наименование свойства; value – устанавливаемое значение	Установить новое значение свойству. Наличие свойства с таким наименованием и типом проверяется во время выполнения программы.
SetObjProp (propertyName:string; value:	propertyName – наименование свойства;	Установить новое значение свойству. Наличие

boolean);	value – устанавливаемое значение	свойства с таким наименованием и типом проверяется во время выполнения программы.

Таблица 3 Процедуры и функции – работа с исходящими и входящими в текущий узел связями

Наименование	Параметры	Описание
GetLinkI (linkName:string):integer;	linkName – наименование связи	Получить значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
GetLinkF (linkName:string):real;	linkName – наименование связи	Получить значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
GetLinkS (linkName:string):string;	linkName – наименование связи	Получить значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
GetLinkB (linkName:string):boolean;	linkName – наименование связи	Получить значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
GetLinkInI	linkName –	Получить значение

(linkName:string):integer;	наименование связи	входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
GetLinkInF (linkName:string):real;	linkName – наименование связи	Получить значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
GetLinkInS (linkName:string):string;	linkName – наименование связи	Получить значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
GetLinkInB (linkName:string):boolean;	linkName – наименование связи	Получить значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
GetLinkOutI (linkName:string):integer;	linkName – наименование связи	Получить значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
GetLinkOutF (linkName:string):real;	linkName – наименование связи	Получить значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
GetLinkOutS (linkName:string):string;	linkName – наименование связи	Получить значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.

GetLinkOutB (linkName:string):boolean;	linkName – наименование связи	Получить значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
SetLink (linkName:string; value: integer);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
SetLink (linkName:string; value: real);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
SetLink (linkName:string; value: string);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
SetLink (linkName:string; value: boolean);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение связи (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
SetLinkIn (linkName:string; value:	linkName – наименование связи;	Установить новое значение входящей связи. Наличие связи с

integer);	value – устанавливаемое значение	таким наименованием и типом проверяется во время выполнения программы.
SetLinkIn (linkName:string; value: real);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
SetLinkIn (linkName:string; value: string);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
SetLinkIn (linkName:string; value: boolean);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение входящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
SetLinkOut (linkName:string; value: integer);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
SetLinkOut (linkName:string; value: real);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
SetLinkOut (linkName:string; value: string);	linkName – наименование связи; value – устанавливаемое значение	Установить новое значение исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
SetLinkOut	linkName –	Установить новое значение

(linkName:string; value: boolean);	наименование связи; value – устанавливаемое значение	исходящей связи. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
LinkAdd (linkName:string; value: integer);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению связи указанное значение (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
LinkAdd (linkName:string; value: real);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению связи указанное значение (вне зависимости от направления: входящая или исходящая). Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
LinkInAdd (linkName:string; value: integer);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению входящей связи указанное значение. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
LinkInAdd (linkName:string; value: real);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению входящей связи указанное значение. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.
LinkOutAdd (linkName:string; value: integer);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению исходящей связи указанное значение. Наличие связи с таким наименованием и типом проверяется во время выполнения

		программы.
LinkOutAdd (linkName:string; value: real);	linkName – наименование связи; value – прибавляемое значение	Прибавить к текущему значению исходящей связи указанное значение. Наличие связи с таким наименованием и типом проверяется во время выполнения программы.

Таблица 4 Арифметические функции

Наименование	Параметры	Описание
Abs(x: integer): integer;	x – аргумент функции	модуль числа
Abs(x:real):real;	x – аргумент функции	модуль числа
ArcTan(x:real):real;	x – аргумент функции	арктангенс (тригонометрическая функция)
Cos(x:real):real;	x – аргумент функции	косинус (тригонометрическая функция)
Exp(x:real):real;	x – аргумент функции	экспонента
Frac(x:real):real;	x – аргумент функции	дробная часть числа
Odd(x:integer):boolean;	x – аргумент функции	является ли число x нечетным
Ord(x:string):intege	x –	ASCII-код первого символа строки

r;	аргумент функции	
Random():real;		Генератор случайных вещественных чисел в диапазоне от 0 до 1.
Random(x:integer):integer;	x – аргумент функции	Генератор случайных целых чисел в диапазоне от 0 до x-1
Round(x:real):integer;	x – аргумент функции	округление числа (до 0,5 в меньшую сторону, иначе – в большую)
Sin(x:real):real;	x – аргумент функции	синус (тригонометрическая функция
Sqrt(x:real):real;	x – аргумент функции	квадратный корень (x-неотрицательное число)
Sqr(x:real):real;	x – аргумент функции	квадрат вещественного числа
Sqr(x:integer):integer;	x – аргумент функции	квадрат целого числа
Trunc(x:real):integer;	x – аргумент функции	урезание дробной части числа

Таблица 5 Прочие процедуры и функции

Наименование	Параметры	Описание
Delay (value:integer);	value – время задержки (секунды)	Задержка в выполнении программы текущего узла в секундах.
DelayDay (value:integer);	value – время задержки (дни)	Задержка в выполнении программы

		текущего узла в днях.
Write(value1, value2, …, valueN)	value1, value2, …, valueN – список выводимых значений (любого типа: integer, real, string, boolean)	Вывод информации на консоль без перевода строки
WriteLn(value1, value2, …, valueN)	value1, value2, …, valueN – список выводимых значений (любого типа: integer, real, string, boolean)	Вывод информации на консоль с последующим переводом строки
WriteLn()	-	Перевод строки консоли