УДК 004.42

О некоторых вопросах разработки моделей компьютерных сетей

Суин Илья Алексеевич
Смоленский государственный университет
Магистрант

Аннотация
В статье затронуты проблемы моделирования компьютерных сетей. Автором рассмотрены понятия сети, моделирования и визуализации. Обсуждается возможность программной реализации приложения создающего модели сетей, отвечающих определенным требованиям. На примере подобного приложения проанализированы процедуры, отвечающие за взаимодействие с базами данных. В завершении статьи охарактеризованы дальнейшие перспективы данной разработки.
Ключевые слова: информатика, программирование, информационно-коммуникационные технологии, базы данных, пользовательское приложение, объектно-ориентированное программирование, алгоритм, модели, компьютерные сети, моделирование, визуализация

Some aspects of the creating computer network models

Suin Ilya Alekseevich
Smolensk State University
Postgraduate

Abstract The article touches upon the problems of modeling computer networks. The author examines the concepts of network, modeling and visualization. The possibility of software implementation of an application of a network model that creates a network that meets certain requirements is discussed. An example of such an application analyzed the procedures responsible for interacting with databases. At the end of the article, further prospects of this development are described.
Keywords: informatics, programming, information and communication technologies, databases, user application, object-oriented programming, algorithm, models, computer networks, modeling, visualization

На данный момент, одним из наиболее значимых видов сетей являются компьютерные сети. По крайней мере, они широко востребованы в области информационно-коммуникационных технологий, которая находит применение практически во всех сферах жизнедеятельности современного человека [1, 2, 3]. Она присутствует во многих аспектах нашей жизни, от места работы и учебы, до дома и мест отдыха. Степень сложности подобных сетей растет пропорционально их объему и набору функций, которые они призваны выполнять. На больших предприятиях сеть компьютеров,

включающая в себя персональные компьютеры, сервера, сетевое оборудование и так далее, имеет весьма сложную структуру. Эта сложность часто затрудняет процесс анализа сети, что в свою очередь может привести к понижению эффективности её работы. Для предотвращения подобного результата, сетевые администраторы создают модели подконтрольных им сетей.

Создание модели тесно связано с процессом визуализации, целью которого, в свою очередь, является подача основной информацию о системе, максимально простым для восприятия способом. Сложность таких визуальных моделей также напрямую зависит от функций, которые они должны выполнять. Иногда достаточно основной схемы, записанной на листе бумаги. В других случаях, при наличии дополнительных требований к модели, необходимо создание более сложных схем, где будут указаны не только основные элементы сети и связи между ними, но и их характеристики. Например, силы этих связей, свойства объектов и связей, точное их расположение и многое другое. Создание подобной модели на бумаге не только крайне затруднительно, но и исключает одну из наиболее важных возможностей работы с моделями – функцию качественного и эффективного редактирования.

Модель компьютерной сети может быть создана различными способами. Так с одной стороны, сеть можно представить в виде таблиц и поместить в базу данных, используя, например, приложение Microsoft Access. Это позволяет систематизировать имеющуюся информацию о объектах, связях и их свойствах. С другой стороны, визуальной моделью сети может быть граф. Графовая модель прекрасно реализует принцип наглядность, и позволяет быстро получить информацию об объектах и связях между ними, а также лучше понять структуру сети.

Современные среды программирования предоставляют возможность разрабатывать приложения, которые позволяют создать одновременно и графовую и табличную модель сети [4]. Созданная таким образом модель будет эффективна по двум причинам. Во-первых, с точки зрения принципа наглядности, будет выгодно использовать граф в качестве её визуального отображения [5, 6]. Это позволит быстро получать основную информацию о модели. Во-вторых, она будет весьма информативна, так как все свойства объектов, их связи и другие важные параметры будут храниться внутри программы в табличной форме [7].

В качестве примера, в объектно-ориентированной среде программирования Microsoft Visual Studio было разработано приложение, отвечающее заданным функциям. Среда Microsoft Visual Studio была выбрана по причине наличия в ней эффективного инструмента для работы с базами данных - SQL Server.

Основной функцией такого приложения является систематизация имеющихся данных о сети. Предполагается, что у пользователя уже есть вся необходимая информация о сети, однако эта информация имеет разрозненный характер. Пользователь же хочет во-первых, хранить все

данные о сети в одном месте, а во-вторых максимально упростить доступ к ним.

Прежде всего, среда программирования позволяет создать интерфейс приложение. Большую часть экрана занимает рабочее поле, где будет отображаться создаваемая графовая модель [8]. В правой части окна программы расположена основная панель редактирования (рис. 1).

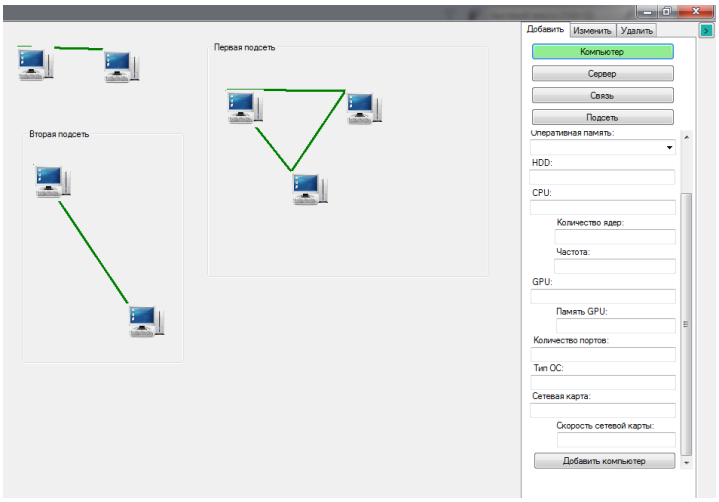

Рисунок 1. Интерфейс приложения

Пользователь может добавлять на форму объекты следующего типа - персональный компьютер, сервер, связь и подсеть. При нажатии на соответствующий элемент, на панели редактирования появляется панель свойств (рис. 2).

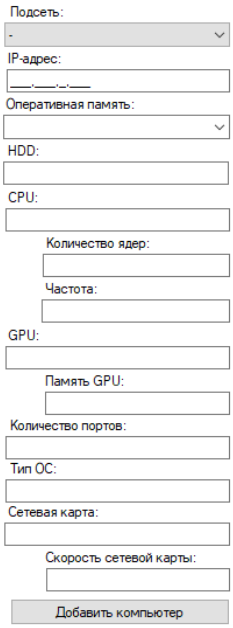

Рисунок. 2. Панель свойств персонального компьютера

Большую часть полей для ввода свойств пользователь должен будет заполнять сам, однако некоторые имеют функцию выбора из предложенных вариантов.

Перемещение объектов по форме реализовано стандартными средствами среды программирования. Аспект того, что объекты находящиеся

внутрь подсети перемещаются вместе с ней, также учтен. Если пользователь должен поместить объект в одну из имеющихся подсетей, то в соответствующей графе он должен выбрать эту подсеть. После этого, объект PictureBox будет закреплен в соответствующем объекте GroupBox - подсети. При перемещении подсети, находящиеся в нем объекты будут перемещаться вместе с ней. Это подчеркивает эффективность выбора элемента типа GroupBox в качестве реализации элемента типа «подсеть».

За размещение объектов каждого вида на форму отвечает свой отдельный метод (рис. 3)

```
ссылка: 1
private void AddComputer(string _subnet)
{

    var computer = new PictureBox();
    computer.Name = textBoxComputerIP.Text;
    computer.Image = NetworkResearch.Properties.Resources.computer;
    computer.Width = 50;
    computer.Height = 50;
    computer.SizeMode = PictureBoxSizeMode.StretchImage;
    ToolTip t = new ToolTip();
    t.SetToolTip(computer, computer.Name);
    MoveObject.LearnToMove(computer);
    SearchSubnet(_subnet).Controls.Add(computer);
    computers++;
}
```

Рисунок 3. Метод создания объекта Computer на форме

Связи на форме представлены линиями, цвет которой зависит от типа выбранной связи. При добавлении в программу связи, пользователь должен выбрать – какой из трех типов связи он планирует создать: компьютер-компьютер, компьютер-подсеть и подсеть-подсеть. Данное разделение присутствует в программе по множеству причин. Например, связь компьютер-подсеть является связью типа «от одного ко многим», в то время как связь типа компьютер-компьютер представляет собой связь «от одного к одному» (рис. 4).

```
ссылок: 3
private void SetLink(int type, string _obj1, string _obj2)
{
    Point p1 = new Point();
    Point p2 = new Point();

    if (type == 0) {
        List<Control> all_pb = new List<Control>();
        GetAllTypedControls(
        this, all_pb, typeof(PictureBox));
        foreach (PictureBox pb in all_pb)
            if (pb.Name == _obj2)
                p2 = new Point(Controls.Find(pb.Name, true)[0].Location.X, Controls.Find(pb.Name, true)[0].Location.Y);
        foreach (PictureBox pb in all_pb)
            if (pb.Name == _obj1)
            {
                p1 = new Point(Controls.Find(pb.Name, true)[0].Location.X, Controls.Find(pb.Name, true)[0].Location.Y);
                Graphics gr = (pb).Parent.CreateGraphics();
                SolidBrush br1 = new SolidBrush(Color.Green);
                Pen pn = new Pen(br1, 3);
                gr.DrawLine(pn, p2, p1);
            }
    }
```

Рисунок 4. Фрагмент кода, отвечающий за создание связи компьютер-компьютер

Помимо создания объектов, на форме присутствуют вкладки для работы с уже существующими объектами - «Изменить», и удаления объектов - «удалить» (рис. 1).

Прежде всего. Пользователь должен выбрать тип объекта, который он собирается изменить – компьютер, подсеть или связь. Далее необходимо выбрать искомый элемент из ниспадающего списка добавленных объектов данного типа. После этого пользователь может изменить любое из имеющихся свойств объекта (рис.5).

```
{
    NetworkResearchDataClassesDataContext context = new NetworkResearchDataClassesDataContext();
    var computers = from c in context.Computers
                    where c.Id == id
                    select c;
    foreach (Computers computer in computers)
    {
        textBoxComputerEditIP.Text = computer.ip.ToString();
        comboBoxComputerEditRAM.Text = computer.ram.ToString();
        textBoxComputerEditHDD.Text = computer.hdd.ToString();
        textBoxComputerEditCPU.Text = computer.cpu.ToString();
        textBoxComputerEditCPUCore.Text = computer.cpu_core_count.ToString();
        textBoxComputerEditCPUFrequency.Text = computer.cpu_frequency.ToString();
        textBoxComputerEditGPU.Text = computer.gpu.ToString();
        textBoxComputerEditGPUMemory.Text = computer.gpu_memory.ToString();
        textBoxComputerEditPorts.Text = computer.ports.ToString();
        textBoxComputerEditOS.Text = computer.os.ToString();
        textBoxComputerEditNetworkAdapter.Text = computer.network_adapter.ToString();
        textBoxComputerEditNetworkAdapterSpeed.Text = computer.network_adapter_speed.ToString();
    }
}
```

Рисунок 5. Фрагмент кода, отвечающий за вывод свойств компьютера для редактирования

Помимо создания на форме объекта PictureBox, программа также создает объект соответствующего класса, после чего заполняет его информацией, полученной от пользователя. Так, при размещении на форме «компьютера» также создастся объект Computer типа Computers (рис. 6), который в последующем будет записан в таблицу Computers (рис. 7) базы данных NetworkResearchDB.

```
AddComputer(comboBoxSubnet.Text);

// Создание объекта Computer типа Computers
Computers Computer = new Computers();

// Инициализация полей значениями введеными в пользовательском интерфейсе
Computer.ip = textBoxComputerIP.Text;
Computer.ram = comboBoxComputerRAM.Text;
Computer.hdd = textBoxComputerHDD.Text;
Computer.cpu = textBoxComputerCPU.Text;
Computer.cpu_core_count = textBoxComputerCPUCore.Text;
Computer.cpu_frequency = textBoxComputerCPUFrequency.Text;
Computer.gpu = textBoxComputerGPU.Text;
Computer.gpu_memory = textBoxComputerGPUMemory.Text;
Computer.ports = textBoxComputerPorts.Text;
Computer.os = textBoxComputerOS.Text;
Computer.network_adapter = textBoxComputerNetworkAdapter.Text;
Computer.network_adapter_speed = textBoxComputerNetworkAdapterSpeed.Text;

// Запись экземпляра в таблицу Computers базы данных NetworkResearchLocalDB с помощью классов LinqToSQL
NetworkResearchDataClassesDataContext context = new NetworkResearchDataClassesDataContext();
context.Computers.InsertOnSubmit(Computer);
context.SubmitChanges();

panelComputer.Visible = false;
```

Рисунок 6. Формирование объекта класса

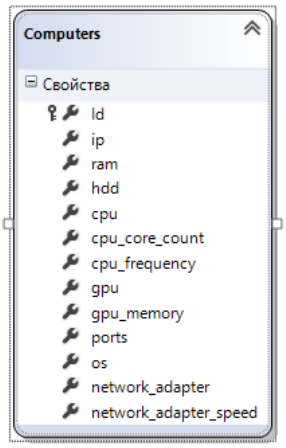

Рисунок 7.Таблица Computers

Таким образом, после завершения работы программы мы получаем качественную визуальную модель, представленную графом на форме. Данная модель подходит для быстрого получения ключевой информации по сети - обычного осмотра хватит чтобы получить информации о количестве компьютеров, серверов, подсетей в сети и вариаций связей между ними. Помимо визуальной модели, мы получаем непосредственно связанную с ней базу данных, в которой хранится вся интересующая нас информация о сети, включающей в себя полный набор свойств каждого компьютера и сервера, сведения о подсетях и связях.

Как было заявлено ранее - приложение служит для систематизации данных о существующей сети. Отличительной чертой данной программы являются перспективы её дальнейшей доработки. Средства объектно-ориентированной среде программирования Microsoft Visual Studio позволяют в дальнейшем внедрить в программу ряд дополнительных функций. Встроенные методы позволяют реализовать всевозможный набор функций для исследования содержания различного рода информационных систем. При этом методология алгебраических систем для изучения структуры и методология функционального анализа для изучения характера взаимосвязей как инвариантные методы исследования обеспечивают высокую точность вне зависимости от сигнатуры и семантики информационной системы [9].

В дальнейшем, представление данной информации в виде базы данных позволит проводить всесторонний анализ сети, средствами среды программирования Microsoft Visual Studio. В свою очередь это позволит не только создавать качественные визуальные и функциональные модели существующих или запланированных сетевых схем, но и проводить их улучшение.

Библиографический список

1. Суин И.А., Козлов С.В. Особенности разработки интерактивных приложений // Постулат. 2017. № 11 (25). С. 29.

2. Козлов С.В. Использование функциональных возможностей информационных систем в производственной сфере // Энергетика, информатика, инновации – 2017 (электроэнергетика, электротехника и теплоэнергетика, математическое моделирование и информационные технологии в производстве): Сборник трудов VII-ой Международной научно-технической конференции. В 3 томах. Смоленск, 2017. Т. 1. С. 298-301.
3. Козлов С.В. Особенности обучения школьников информатике в профильной школе // Научно-методический электронный журнал Концепт. 2014. № 1. С. 31-35.
4. Максимова Н.А. Разработка приложений на основе сервис-ориентированных систем // NovaInfo.Ru. 2016. Т. 3. № 46. С. 41-44.
5. Козлов С.В. Применение соответствия Галуа для анализа данных в информационных системах // Траектория науки. 2016. Т. 2. № 3 (8). С. 18.
6. Шатохина А. А., Киселева О. М. Построение индивидуальной траектории обучения информатике для детей с ограниченными возможностями здоровья // Молодёжь и наука: актуальные проблемы педагогики и психологии. 2017. № 2. С. 235-239
7. Гринченко Н. Н., Гусев Е. В., Макаров Н. П. Проектирование баз данных. СУБД Microsoft Access. Учебное пособие. – М., 2004. – 240 с.
8. Козлов С.В. Теория графов и соответствия Галуа как инструменты проектирования информационных систем // NovaInfo.Ru. 2016. Т. 3. № 48. С. 144-149.
9. Козлов С.В., Суин И.А. О некоторых подходах математического описания и анализа многомерной структуры информационных систем // Системы компьютерной математики и их приложения. Смоленск: СмолГУ, 2018. С. 177-182