Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

О Т Ч Е Т

по домашнему заданию № 1

Название: Программирование на Object Pascal с использованием классов

Дисциплина: Объектно-ориентированное программирование

Студент	ИУ6-22Б		С.В. Астахов
	(Группа)	(Подпись, дата)	(И.О. Фамилия)
Преподаватель			
		(Подпись, дата)	(И.О. Фамилия)

Москва, 2020

Задание 1

Разработать иерархию классов. Поместить определение классов в отдельном модуле.
Класс, позволяющий рисовать окружность некоторого размера с центром в точке, определенной нажатием левой клавиши мыши.
Класс, позволяющий рисовать квадрат того же размера с центром в точке, определенной нажатием правой клавиши мыши.
Размер и цвет фигур задавать с использованием интерфейсных элементов.
В отчете показать иерархии используемых классов VCL и разработанных классов, граф состояния пользовательского интерфейса и объектную декомпозицию.

Исходный код

(Unit1.pas)

```
unit Unit1;

{$mode objfpc}{$H+}

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, ExtCtrls, StdCtrls,
  ComCtrls, ColorBox, Unit2;

type

  { TForm1 }

  TForm1 = class(TForm)
    ColorBox1: TColorBox;
    ComboBox1: TComboBox;
    Edit1: TEdit;
    Label1: TLabel;
    PaintBox1: TPaintBox;
    UpDown1: TUpDown;
    procedure Button1Click(Sender: TObject);
    procedure PaintBox1Click(Sender: TObject);
    procedure PaintBox1MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: integer);
  private

  public

  end;
```

```
var
  Form1: TForm1;

implementation

var
  sq: TSquare;
  circle: TCircle;

{$R *.lfm}

{ TForm1 }

procedure TForm1.Button1Click(Sender: TObject);
begin

end;

procedure TForm1.PaintBox1Click(Sender: TObject);
begin

end;

procedure TForm1.PaintBox1MouseDown(Sender: TObject; Button:
TMouseButton;
  Shift: TShiftState; X, Y: integer);
var
  ss, code: integer;
begin
  val(Edit1.Text, ss, code);
  PaintBox1.Canvas.Brush.Color := ColorBox1.Color;
  if ComboBox1.ItemIndex = 1 then
  begin
    sq.Init(X, Y, ss, PaintBox1);
    sq.Draw(ColorBox1.Selected);
  end;
  if ComboBox1.ItemIndex = 0 then
  begin
    circle.Init(X, Y, ss, PaintBox1);
    circle.Draw(ColorBox1.Selected);
  end;
end;
```

```
end.
```

(Unit2.pas)

```
unit Unit2;

{$mode objfpc}{$H+}

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics,
  Dialogs, ExtCtrls, StdCtrls,
  ComCtrls, ColorBox;

type
  TFigure = object
  private
    x, y: integer;
    connect: TPaintBox;
  public
    procedure Init(a, b: integer; assoc: TPaintBox);
  end;

  TSquare = object(TFigure)
  private
    size: integer;
  public
    procedure Init(a, b, s: integer; assoc: TPaintBox);
    procedure Draw(clr: TColor);
  end;

  TCircle = object(TFigure)
  private
    size: integer;
  public
    procedure Init(a, b, s: integer; assoc: TPaintBox);
    procedure Draw(clr: TColor);
  end;
```

```
implementation

procedure TFigure.Init(a, b: integer; assoc: TPaintBox);
begin
  Self.x := a;
  Self.y := b;
  Self.connect := assoc;
end;

procedure TSquare.Init(a, b, s: integer; assoc: TPaintBox);
begin
  inherited Init(a, b, assoc);
  Self.size := s;
end;

procedure TSquare.Draw(clr: TColor);
begin
  Self.connect.canvas.brush.color := clr;
  Self.connect.canvas.rectangle(x, y, x + size, y + size);
end;

procedure TCircle.Init(a, b, s: integer; assoc: TPaintBox);
begin
  inherited Init(a, b, assoc);
  Self.size := s;
end;

procedure TCircle.Draw(clr: TColor);
begin
  Self.connect.canvas.brush.color := clr;
  Self.connect.canvas.ellipse(x, y, x + size, y + size);
end;

end.
```

Скриншоты графического интерфейса

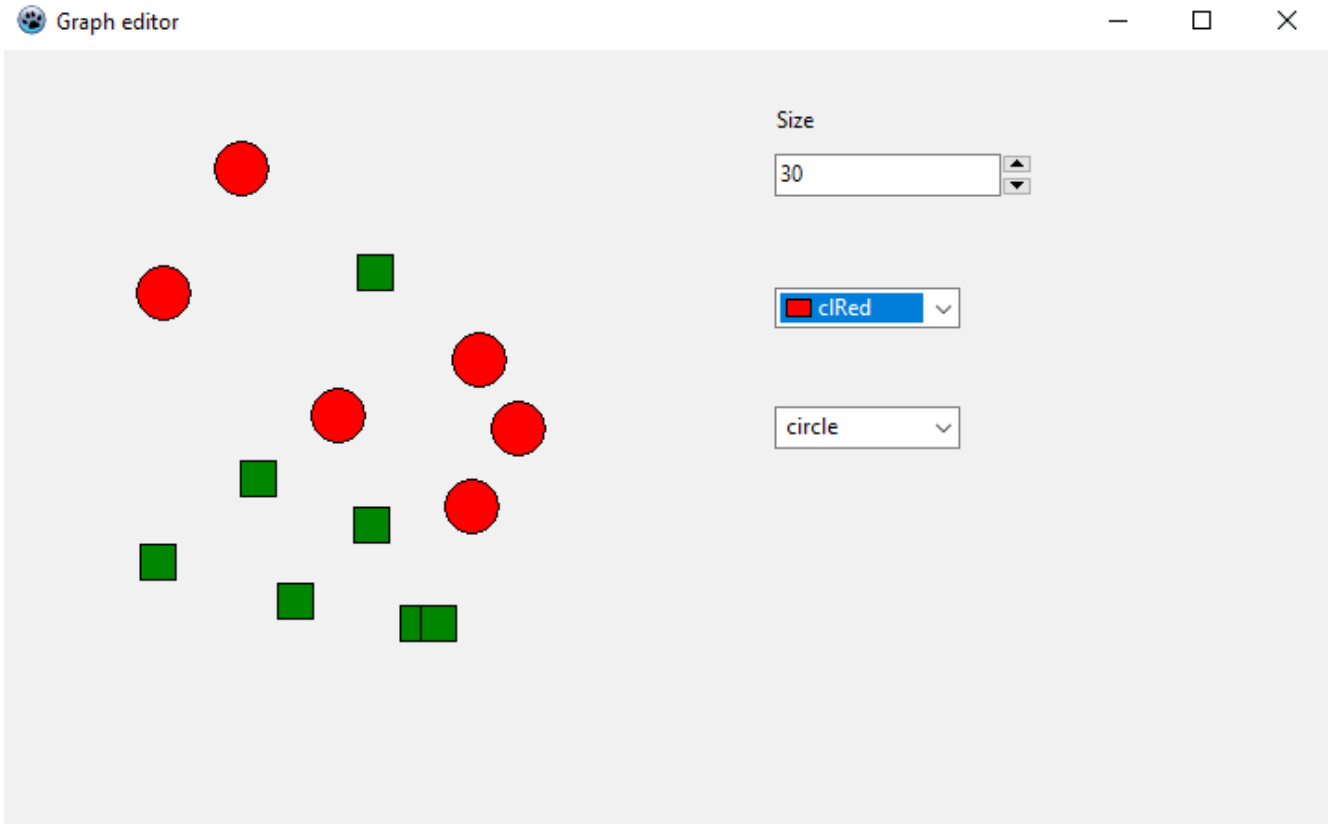


Диаграмма классов визуальных компонентов

(Unit1)

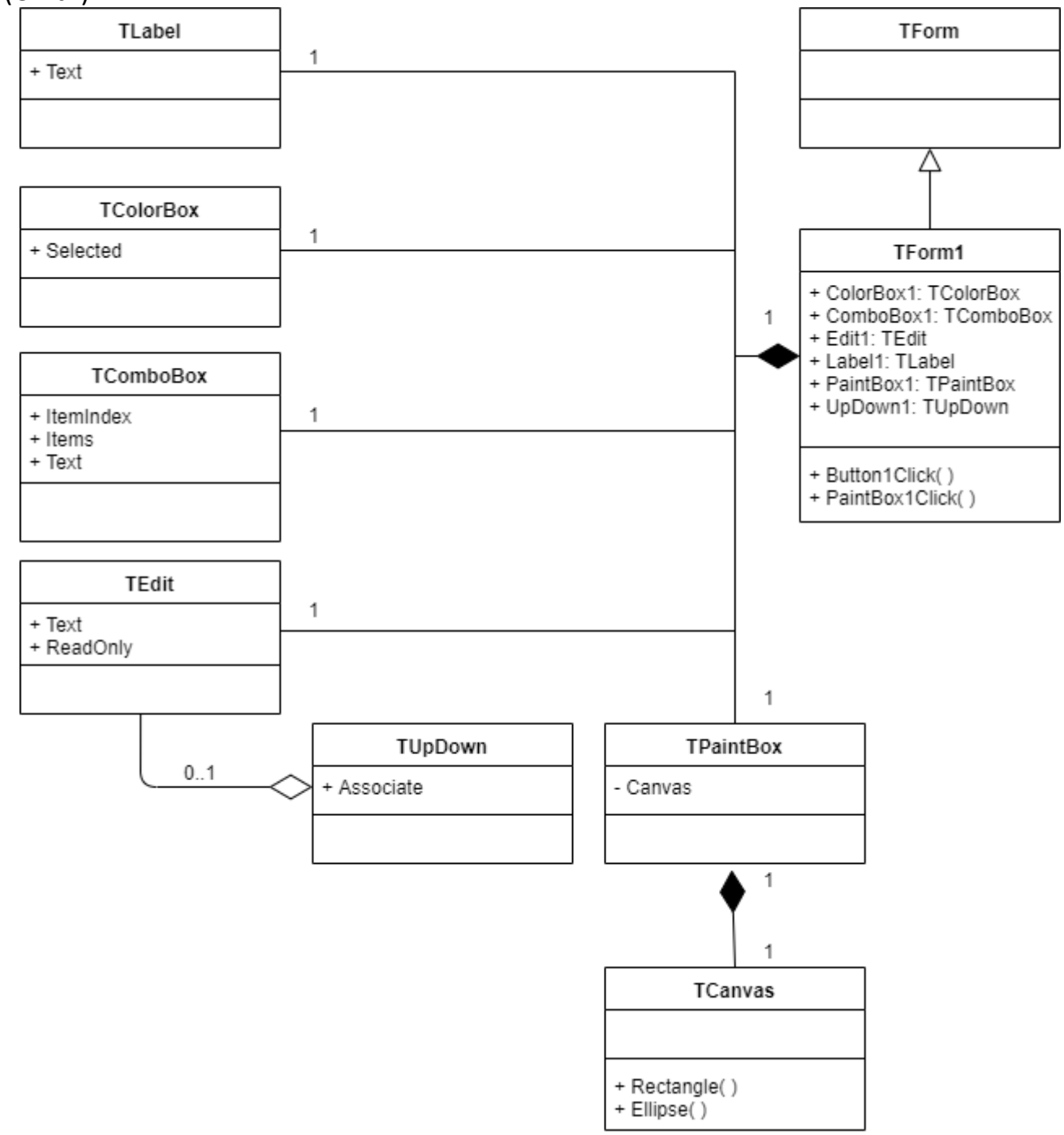


Диаграмма классов невизуальных компонентов

(Unit2.pas)

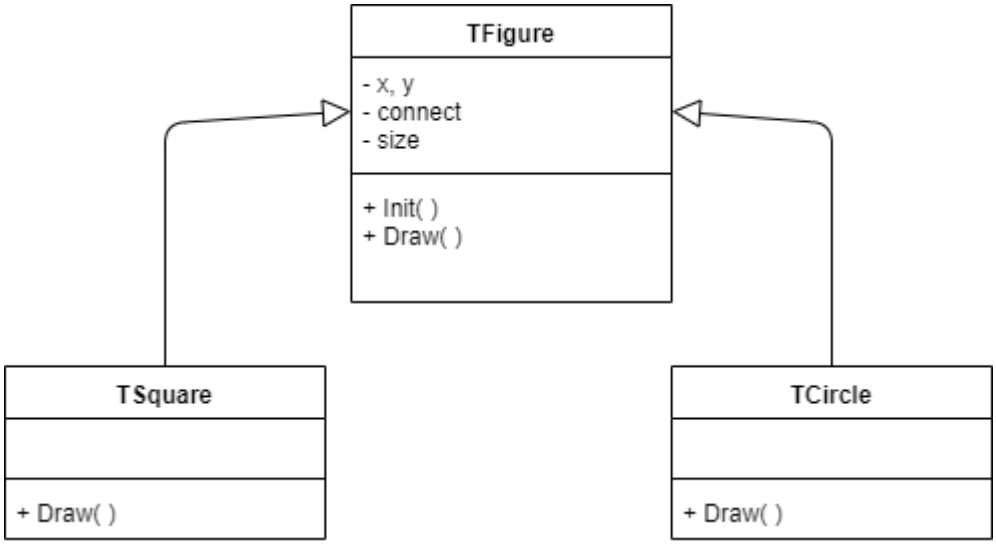
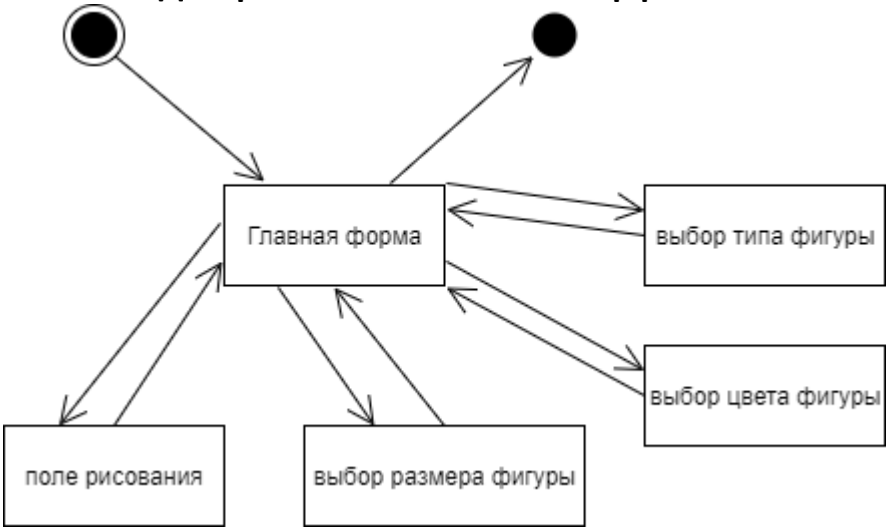
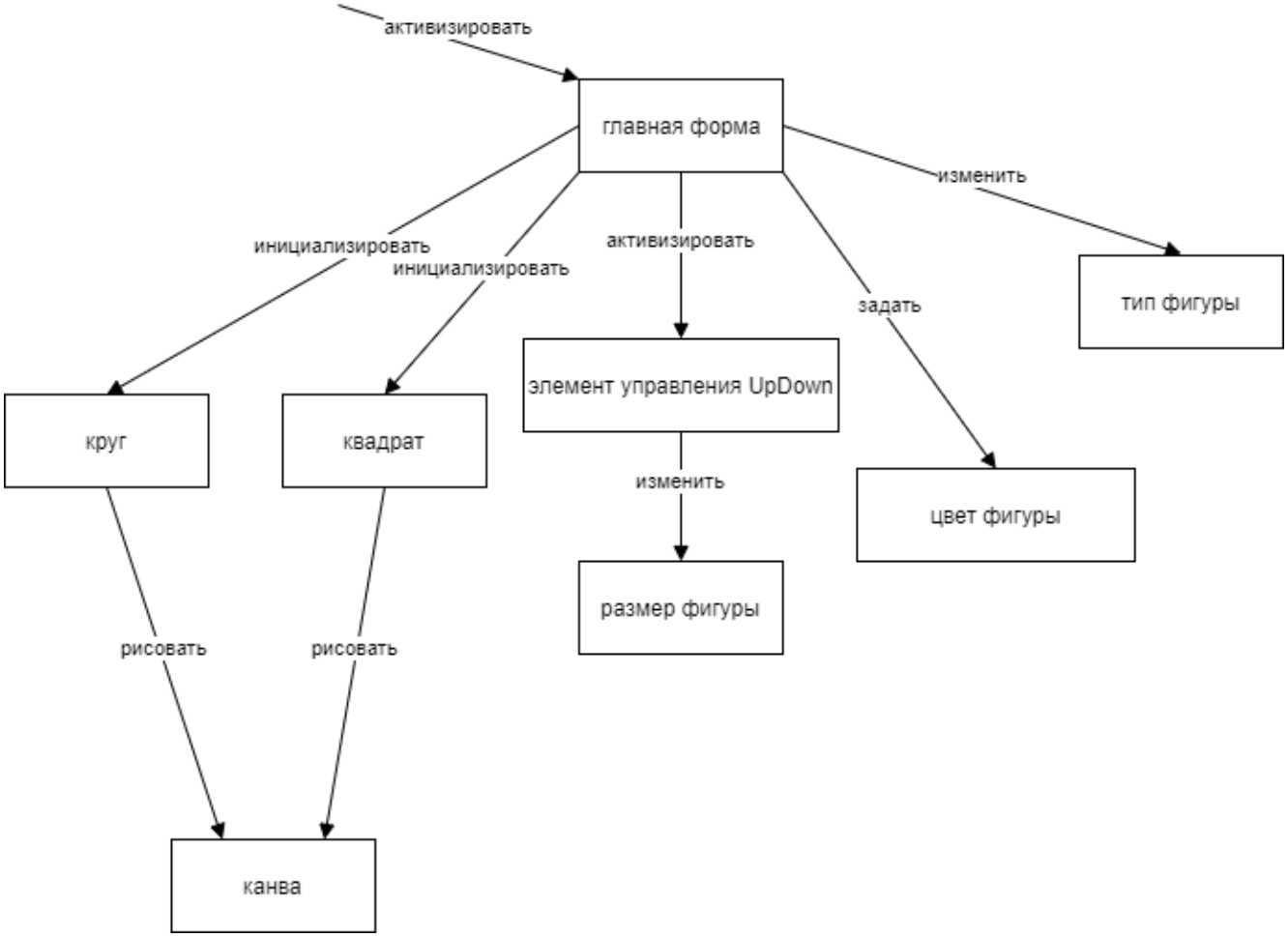


Диаграмма состояний интерфейса



Объектная декомпозиция



Задание 2

Разработать программу, содержащую описание трех графических объектов: круг с вырезанной четвертью, эллипс, квадрат.
Реализуя механизм полиморфизма, привести объекты в одновременное вращение вокруг их геометрических центров с различными угловыми скоростями.
В отчете привести диаграмму используемых классов VCL и разработанных классов, граф состояний пользовательского интерфейса и объектную декомпозицию.

Исходный код

(Unit1.pas)

```
unit Unit1;

{$mode objfpc}{$H+}

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, ExtCtrls, StdCtrls,
Unit2;

type

  { TForm1 }

  TForm1 = class(TForm)
    Button1: TButton;
    PaintBox1: TPaintBox;
    PaintBox2: TPaintBox;
    PaintBox3: TPaintBox;
    Timer1: TTimer;
    procedure Button1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
  private

  public

  end;

var
  Form1: TForm1;

implementation

var
  circle: TCircle;
  square: TSquare;
```

```
  ellipse: TEllipse;
  arg1, arg2, arg3: real;


{$R *.lfm}

{ TForm1 }

procedure TForm1.Button1Click(Sender: TObject);
begin
  arg1 := 0;
  arg2 := 0;
  arg3 := 0;

  Circle.Init(PaintBox1);
  Square.Init(PaintBox2);
  ellipse.Init(PaintBox3);
  Timer1.Enabled := not (Timer1.Enabled);
  //test
 { arg := 0;
  while True do
    begin
    sleep(200);
    if CheckBox1.Checked then
    begin
      if arg > (PI * 2) then
        arg := 0;
      arg := arg + 0.01;
      Circle.Init(PaintBox1);
      Circle.Draw(arg);
    end;

    end;                  }
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
  if arg1 > (PI * 2) then
    arg1 := 0;
  arg1 := arg1 + 0.1;
  if arg2 > (PI * 2) then
    arg2 := 0;
  arg2 := arg2 + 0.2;
  if arg3 > (PI * 2) then
    arg3 := 0;
  arg3 := arg3 + 0.3;
```

```
  Circle.Draw(arg1);
  Square.Draw(arg2);
  Ellipse.Draw(arg3);
end;

end.
```

(Unit2.pas)

```
unit Unit2;

{$mode objfpc}{$H+}

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, ExtCtrls, StdCtrls;

type
  TFigure = object
  private
    canv: TPaintBox;
  public
    procedure Init(inp: TPaintBox);
    procedure Draw(t: real);
  end;

  TCircle = object(TFigure)
    procedure Draw(t: real);
  end;

  TSquare = object(TFigure)
    procedure Draw(t: real);
  end;

  TEllipse = object(TFigure)
    procedure Draw(t: real);
  end;

implementation

procedure TFigure.Init(inp: TPaintBox);
begin
  Self.canv := inp;
end;

procedure TFigure.Draw(t: real);
begin
```

```
  // abstract
end;

procedure TCircle.Draw(t: real);
var
  x1, y1, x2, y2: integer;
begin
  x1 := 125 - trunc(125 * cos(t));
  y1 := 125 - trunc(125 * sin(t));
  x2 := 125 - trunc(125 * cos(t + PI / 2));
  y2 := 125 - trunc(125 * sin(t + PI / 2));
  Self.canv.canvas.Clear;
  Self.canv.canvas.pie(25, 25, 225, 225, x1, y1, x2, y2);
end;

procedure TSquare.Draw(t: real);
var
  x1, y1, x2, y2, x3, y3, x4, y4: integer;
begin
  x1 := 125 - trunc(90 * cos(t));
  y1 := 125 - trunc(90 * sin(t));
  x2 := 125 - trunc(90 * cos(t + PI / 2));
  y2 := 125 - trunc(90 * sin(t + PI / 2));
  x3 := 125 - trunc(90 * cos(t + PI));
  y3 := 125 - trunc(90 * sin(t + PI));
  x4 := 125 - trunc(90 * cos(t + 3 * PI / 2));
  y4 := 125 - trunc(90 * sin(t + 3 * PI / 2));
  Self.canv.canvas.Clear;
  Self.canv.canvas.line(x1, y1, x2, y2);
  Self.canv.canvas.line(x2, y2, x3, y3);
  Self.canv.canvas.line(x3, y3, x4, y4);
  Self.canv.canvas.line(x4, y4, x1, y1);

  {Self.canv.canvas.line(125 - x1, 125 - y1, x2, y2);
  Self.canv.canvas.line(x1, y1, 125 - x2, 125 - y2);
  Self.canv.canvas.line(125 - x1, 125 - y1, 125 - x2, 125 - y2);       }
end;

procedure TEllipse.Draw(t: real);
var
  x1, y1, x2, y2, x3, y3, x4, y4, xc1, yc1, xc2, yc2: integer;
begin
  x1 := 125 - trunc(60 * cos(t));
  y1 := 125 - trunc(60 * sin(t));
  x2 := 125 - trunc(60 * cos(t + PI / 2));
```

```
  y2 := 125 - trunc(60 * sin(t + PI / 2));
  x3 := 125 - trunc(60 * cos(t + PI));
  y3 := 125 - trunc(60 * sin(t + PI));
  x4 := 125 - trunc(60 * cos(t + 3 * PI / 2));
  y4 := 125 - trunc(60 * sin(t + 3 * PI / 2));
  xc1 := trunc((x2 + x3) / 2);
  yc1 := trunc((y2 + y3) / 2);
  xc2 := trunc((x4 + x1) / 2);
  yc2 := trunc((y4 + y1) / 2);
  Self.canv.canvas.Clear;

  //Self.canv.canvas.line(xc1, yc1, xc2, yc2);
  Self.canv.canvas.line(x1, y1, x2, y2);
  //Self.canv.canvas.line(x2, y2, x3, y3);
  Self.canv.canvas.line(x3, y3, x4, y4);
  //Self.canv.canvas.line(x4, y4, x1, y1);

  Self.canv.canvas.arc(xc1 - 40, yc1 - 40, xc1 + 40, yc1 + 40, x3, y3, x2, y2);
  Self.canv.canvas.arc(xc2 - 40, yc2 - 40, xc2 + 40, yc2 + 40, x1, y1, x4, y4);

  {Self.canv.canvas.line(125 - x1, 125 - y1, x2, y2);
  Self.canv.canvas.line(x1, y1, 125 - x2, 125 - y2);
  Self.canv.canvas.line(125 - x1, 125 - y1, 125 - x2, 125 - y2);        }
end;

end.
```

Скриншоты

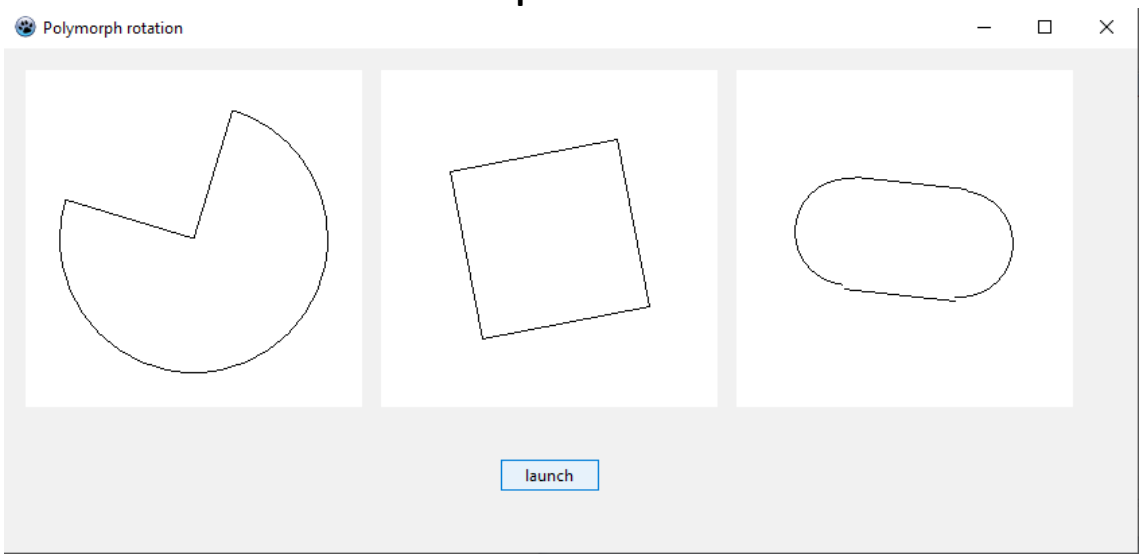


Диаграмма классов визуальных компонентов

(Unit1.pas)

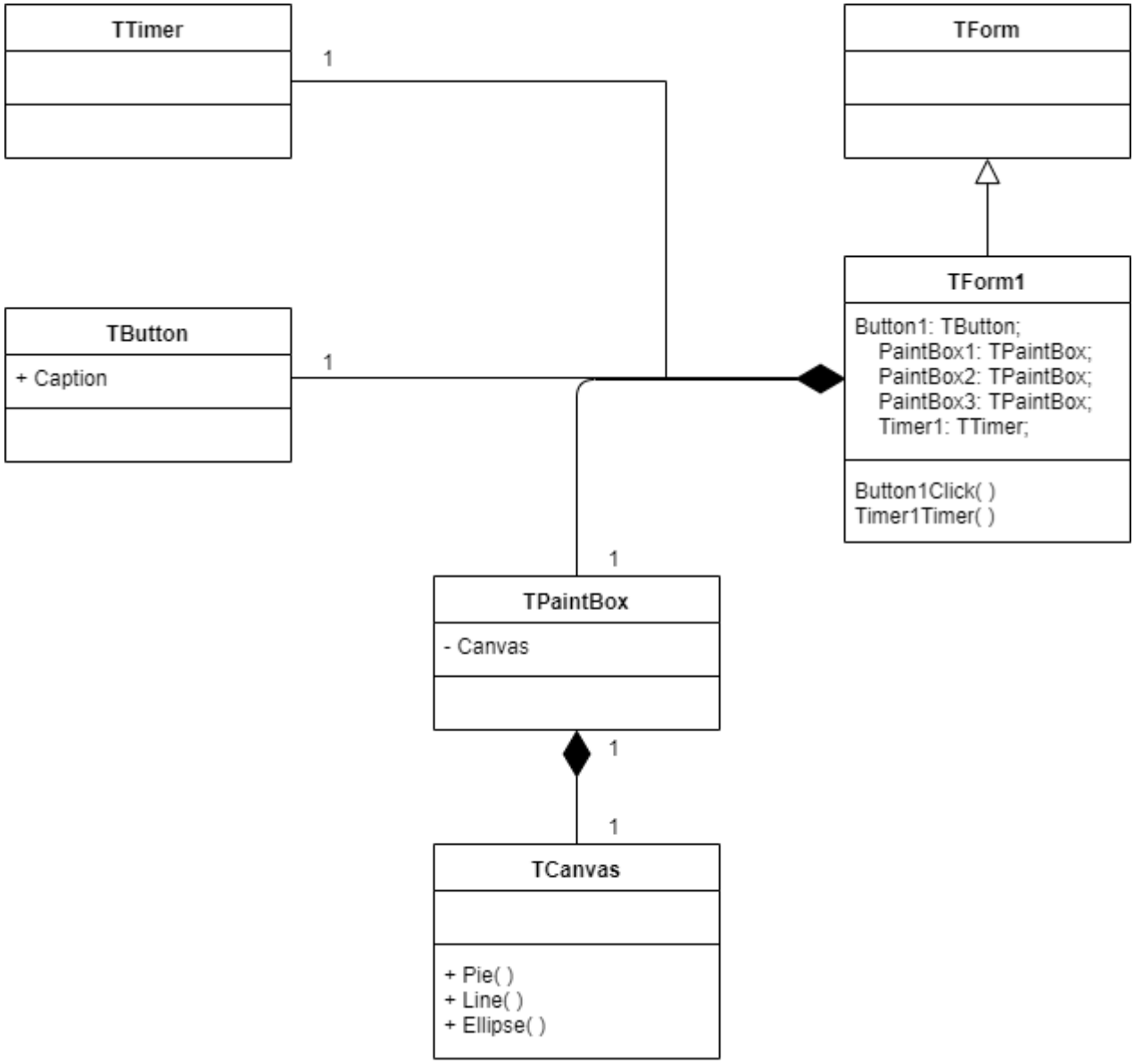


Диаграмма классов невизуальных компонентов

(Unit2.pas)

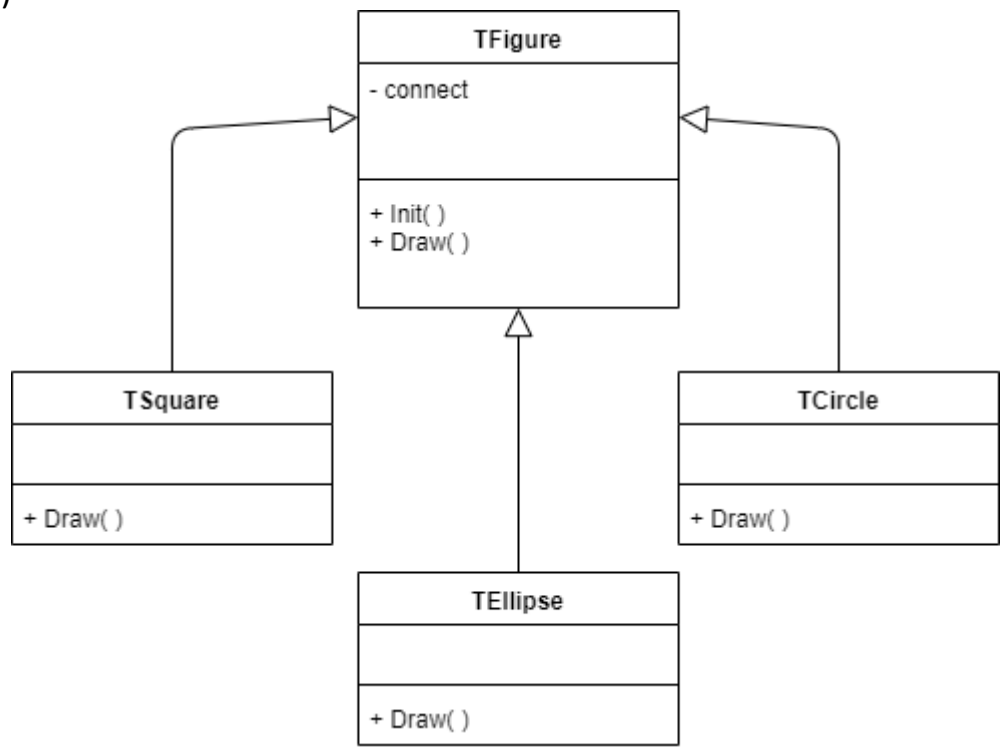
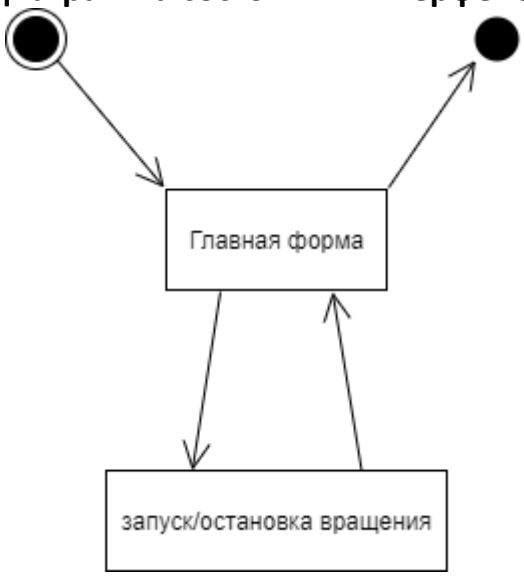
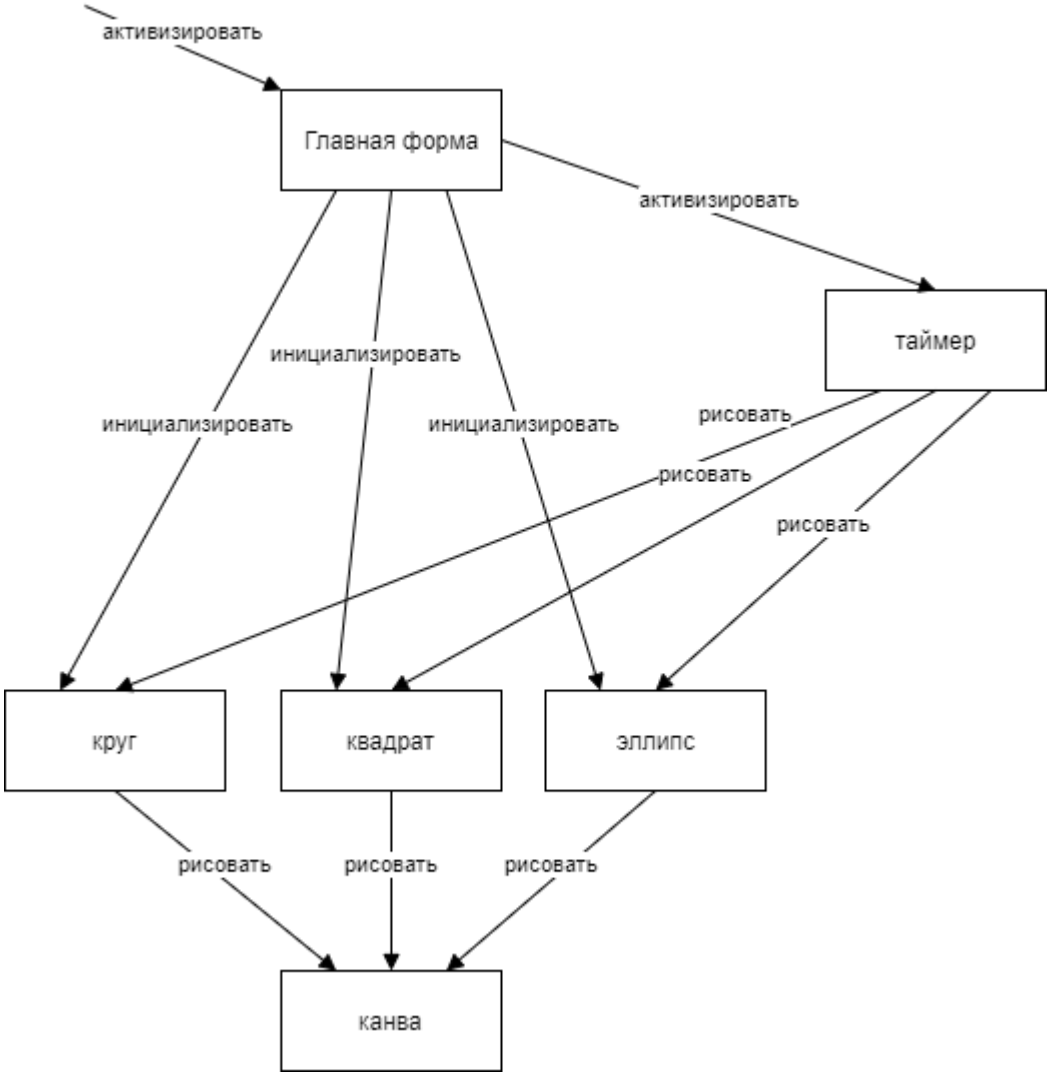


Диаграмма состояний интерфейса



Объектная декомпозиция



Вывод

- Delphi и VCL предоставляют большое количество классов для работы с визуальной частью программы, создания графических редакторов

- Чтобы реализовать несколько схожих по своей природе классов(здесь — различных фигур) при достаточном количестве различий между ними(здесь - форма) целесообразно использовать абстрактный класс(здесь - фигура), от которого затем будут полиморфно наследоваться другие классы(здесь — квадрат и круг, эллипс)