

Joseph Ryan | COSE50647 | January 9, 2015

Airplane Terminal Simulator with Logger Concurrent Network Applications

Table of Contents

[Design 2](#_Toc408572898)

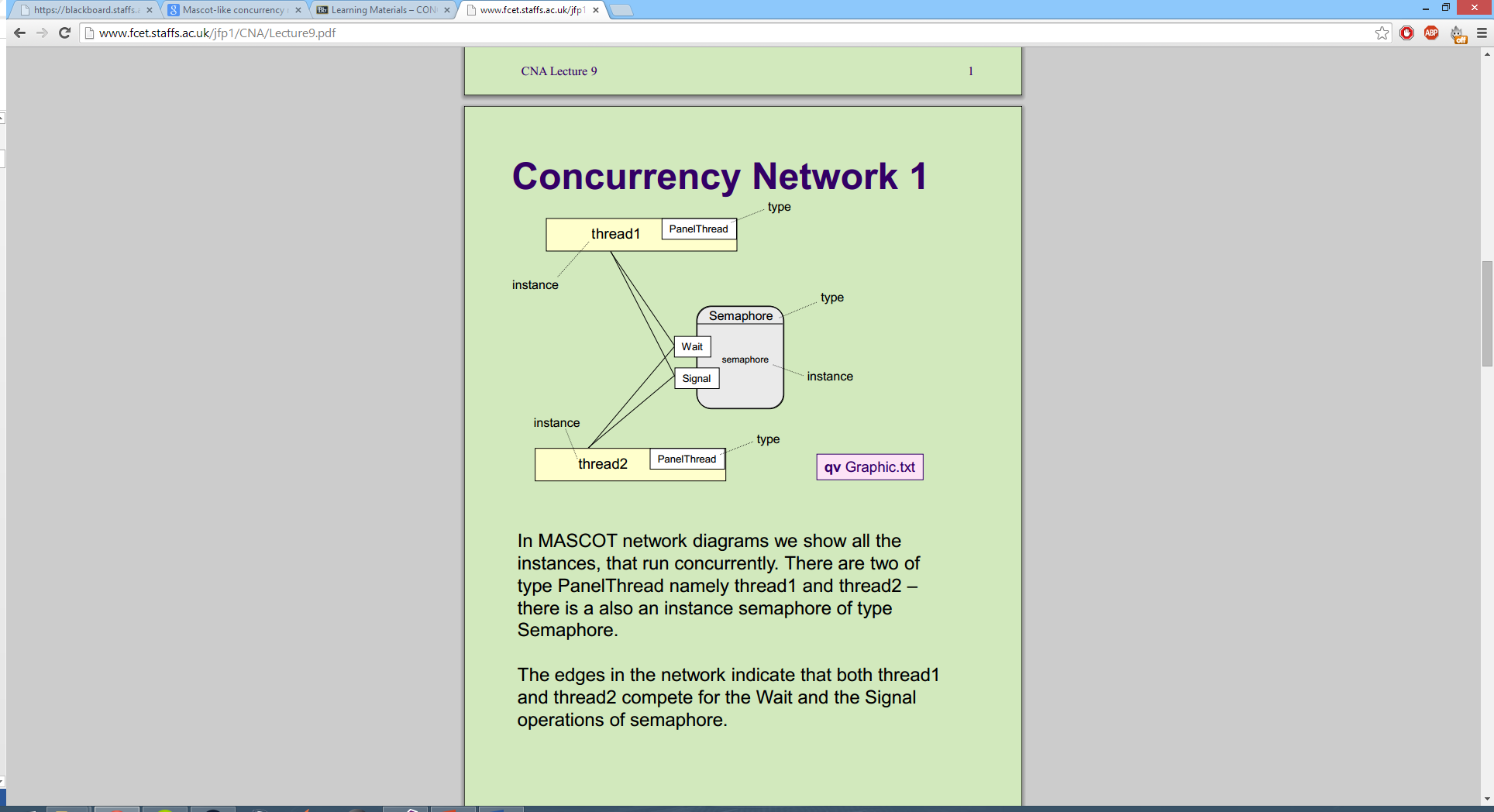
[How-To 6](#_Toc408572899)

[Server 6](#_Toc408572900)

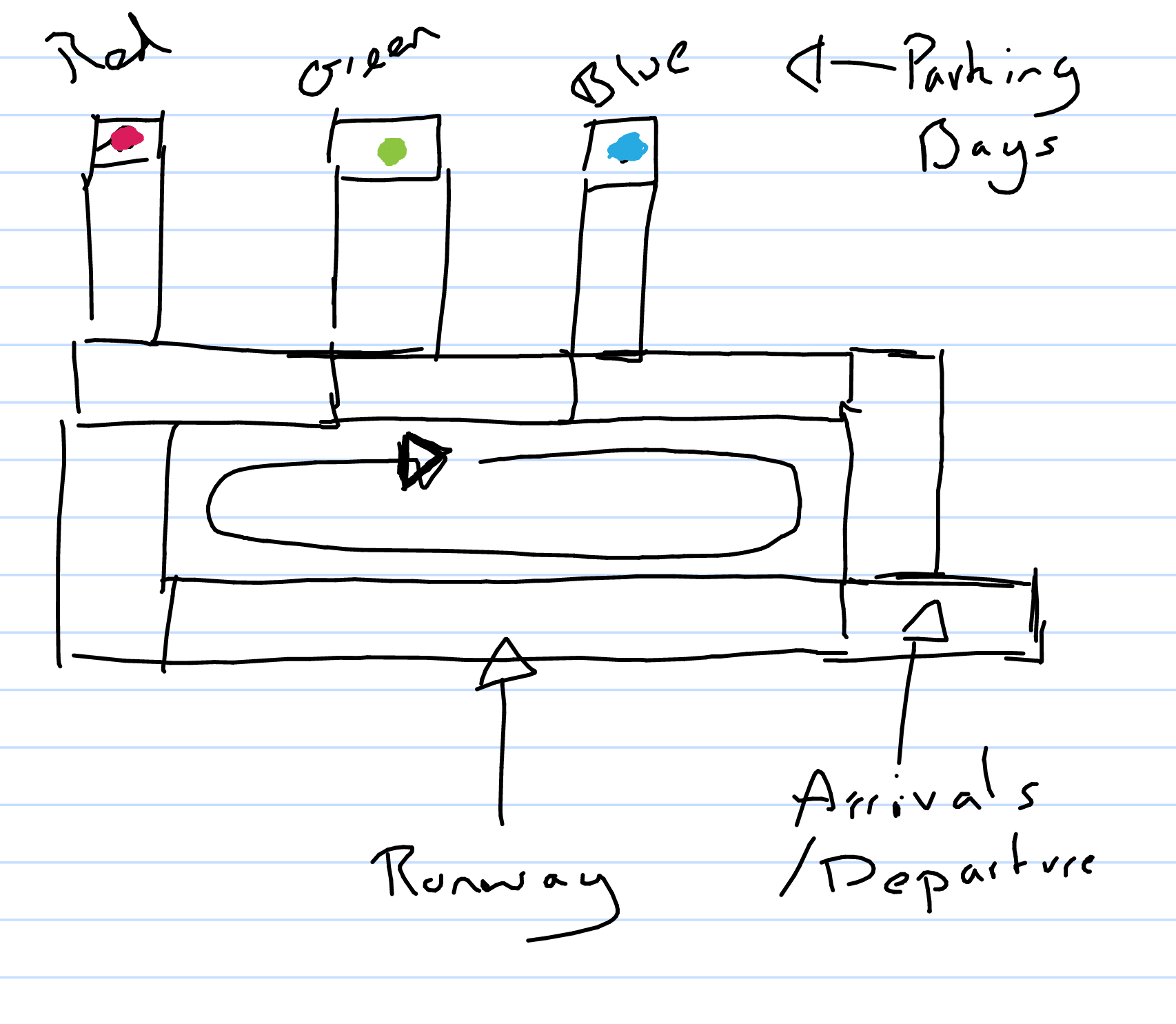
[Documentation 6](#_Toc408572901)

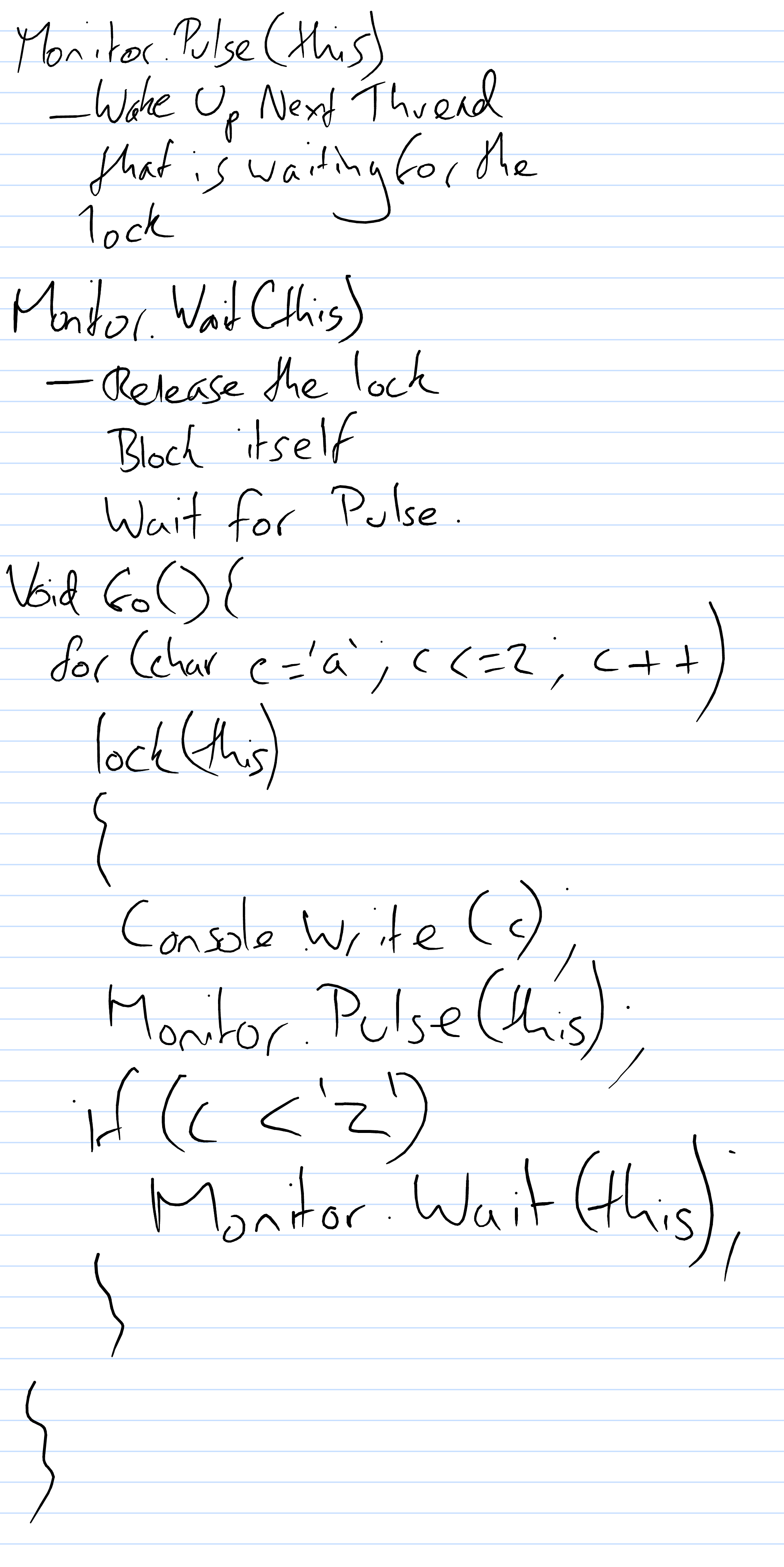
[Code 11](#_Toc408572902)

# Design

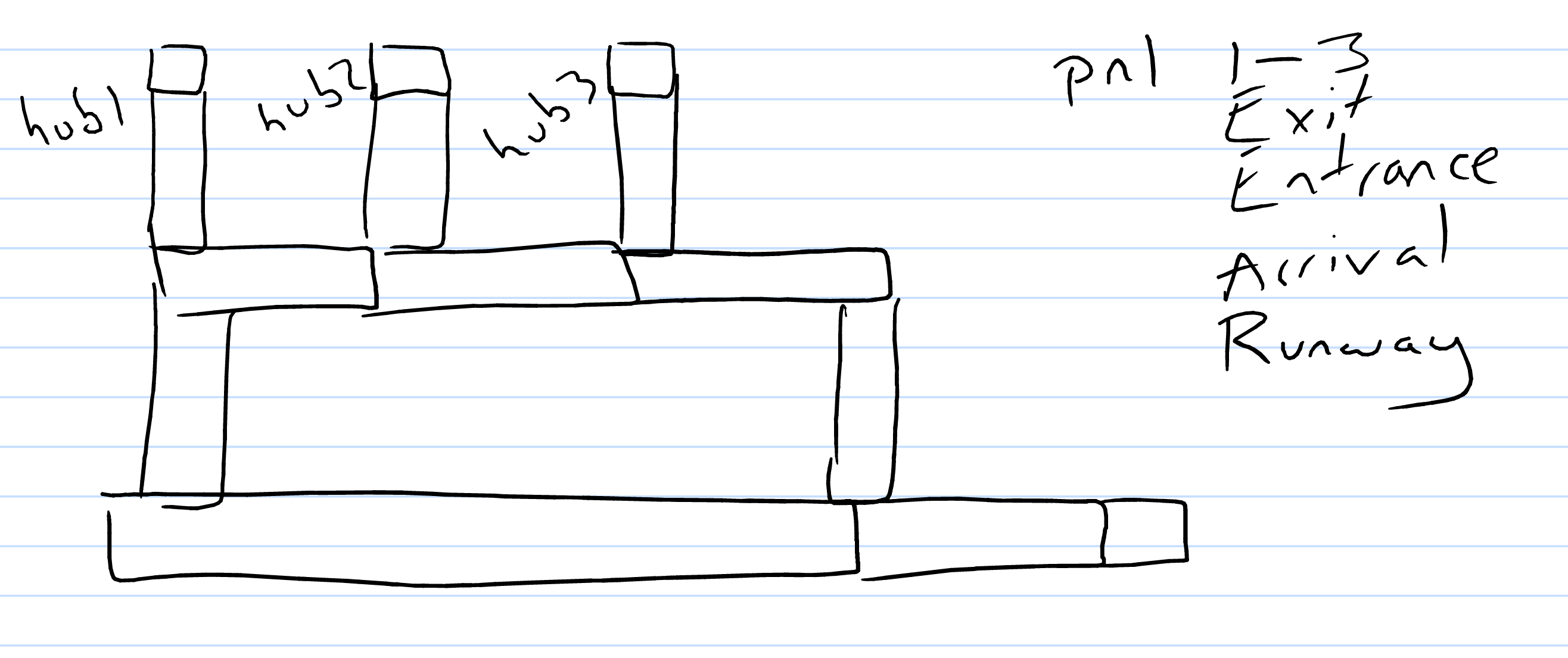
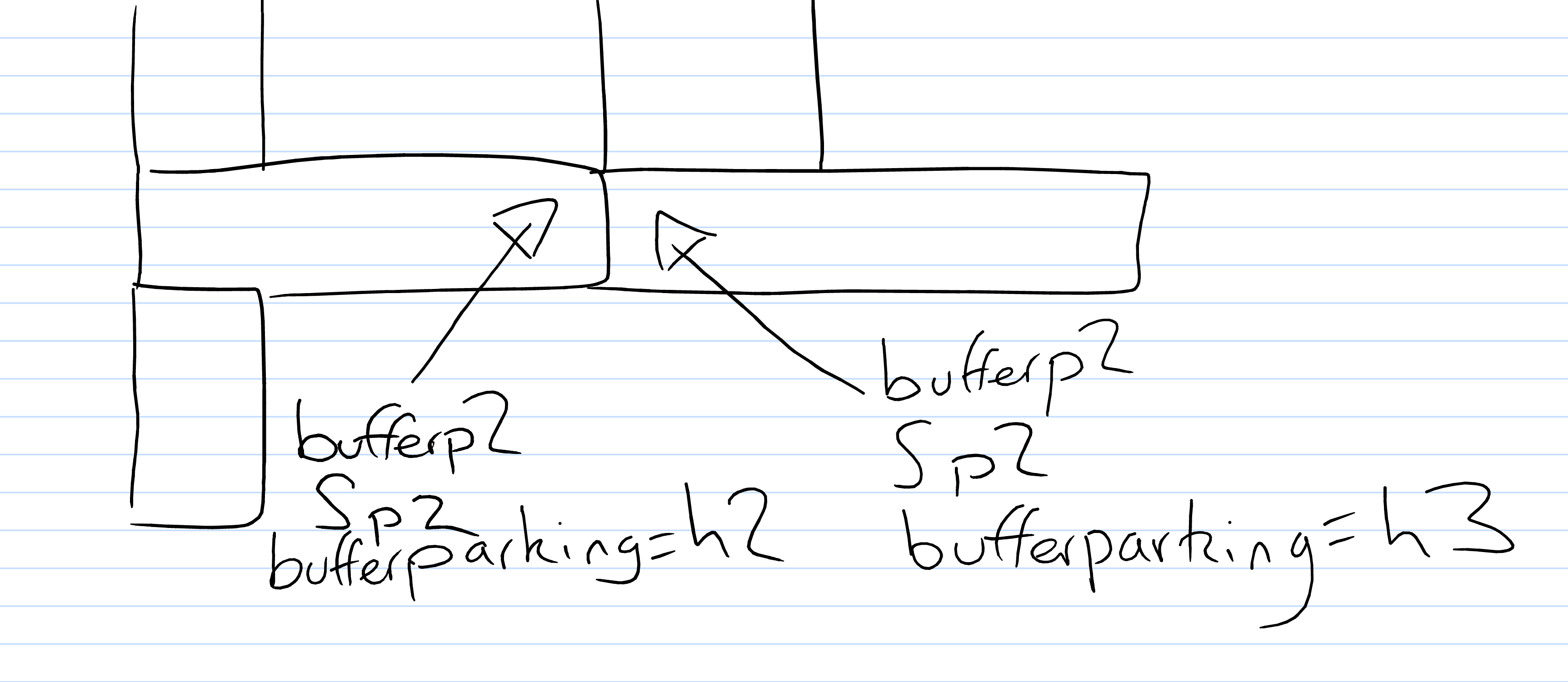
The program uses the idea of any panel with an output can also have an input, complete with its own Buffer and Semaphore. In the Diagram below, thread1 would be a plane departing a panel, and thread2 is the panel it arrives at. They both use the same semaphore and buffer for this link between them, but the thread contains additional semaphores for alternative routes, i.e. parking.

This means that a correct diagram for this program is actually a circle, with branches for each plane hub, that ultimately circle back around to rejoin the main loop. This adds room for growth and expansion, that is easy to connect. The code itself contains comments for easy substitution of the panels, and adding a new hub would be the inclusion of 2 new lines, changing a panel departure for the new hub, and the hub departure to the previous panels arrival buffer and semaphore.

These two images are exports straight from initial notes, showing the layout for airport, and the initial idea for the arrivals hub to have both an arrivals and departure buffer and semaphore.

This second image shows code for the Pulse and Wait methods used in the program. Using this code allows the concurrent threads to work side-by-side, without overlaps or ‘plane crashes’.

These two designs show the ideas implemented, with panels having multiple buffers and semaphores, allowing the program to use the correct destination buffers, but not allowing full global access to all panels, resulting in faster code and no risk of plane crashes.



# How-To

The program will open with all planes in their starting hubs and the green plane ready to ‘land’ at the runway.

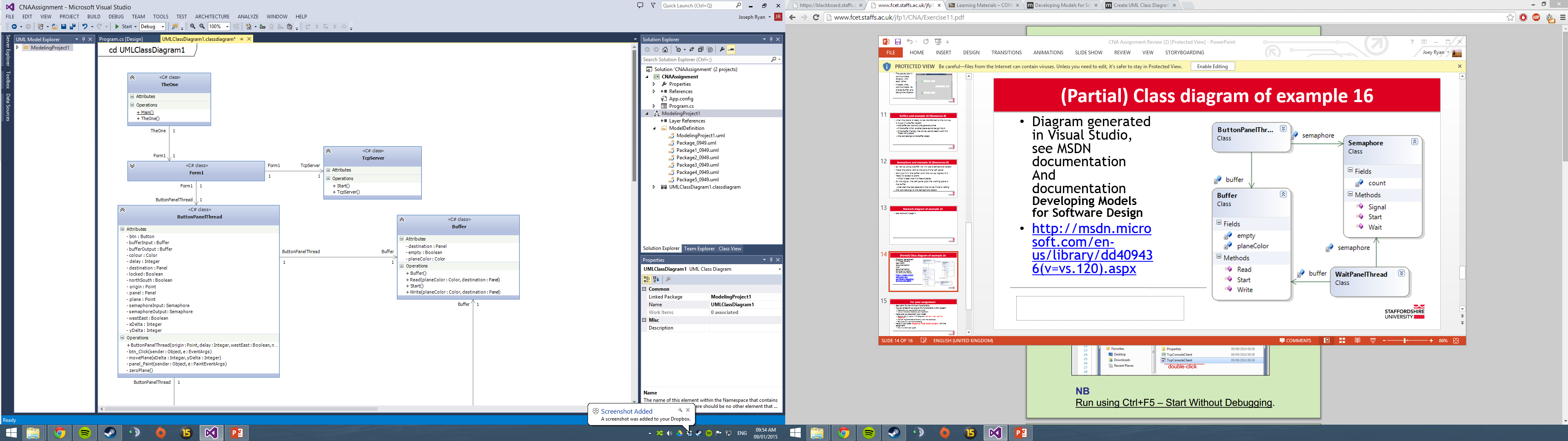
Start planes moving by clicking the buttons next to them and the planes will start their journey to the runway to take-off.

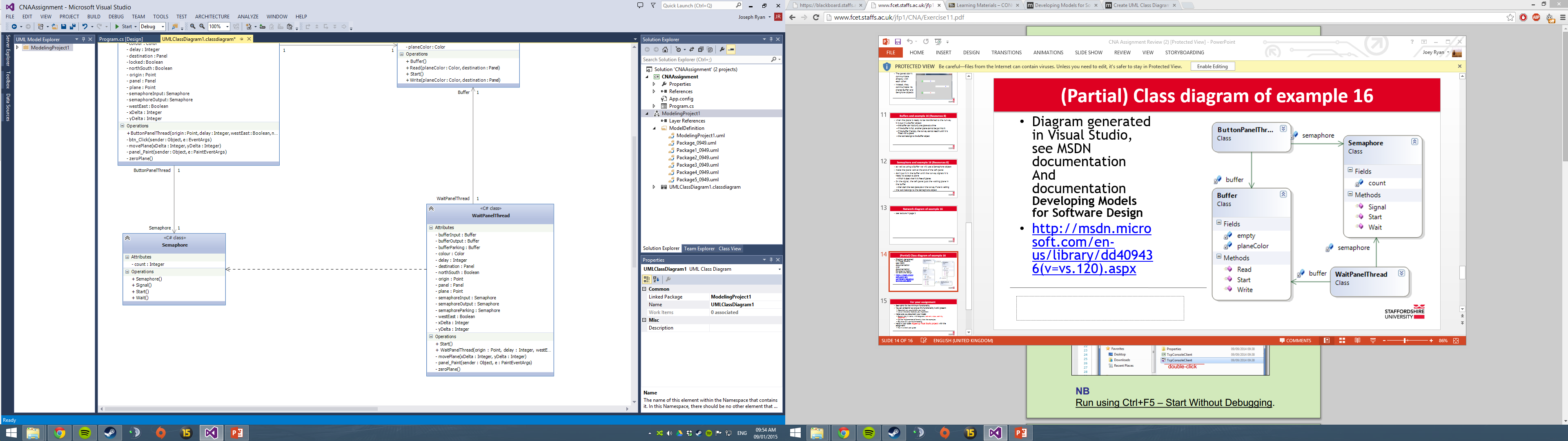
If a path is blocked, the plane will wait for the area to be empty before continuing.

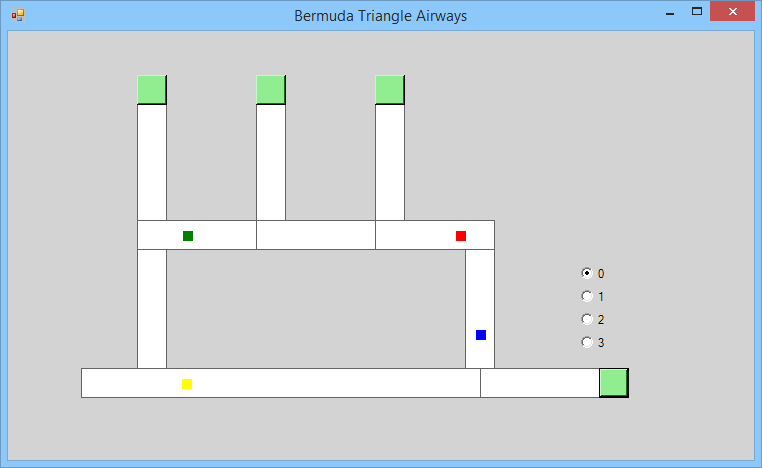
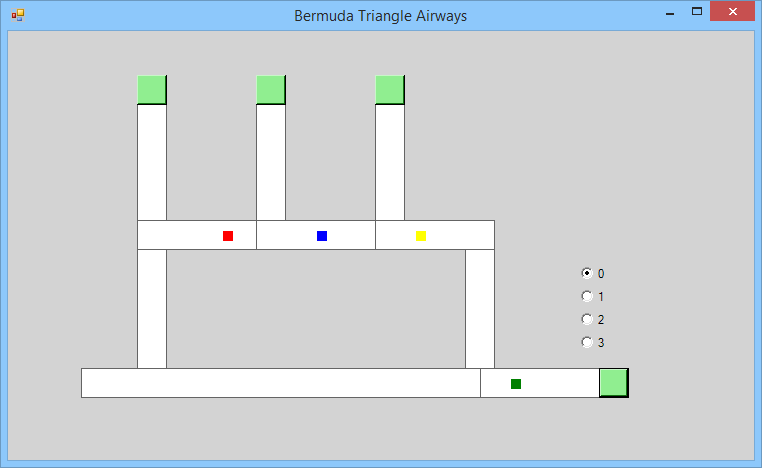
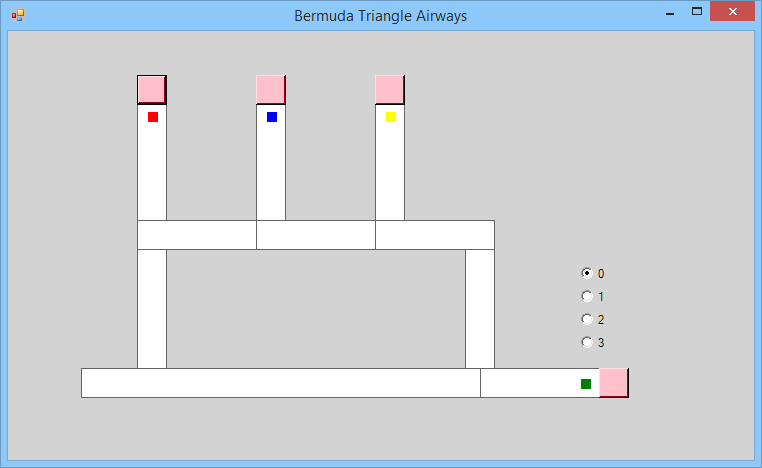
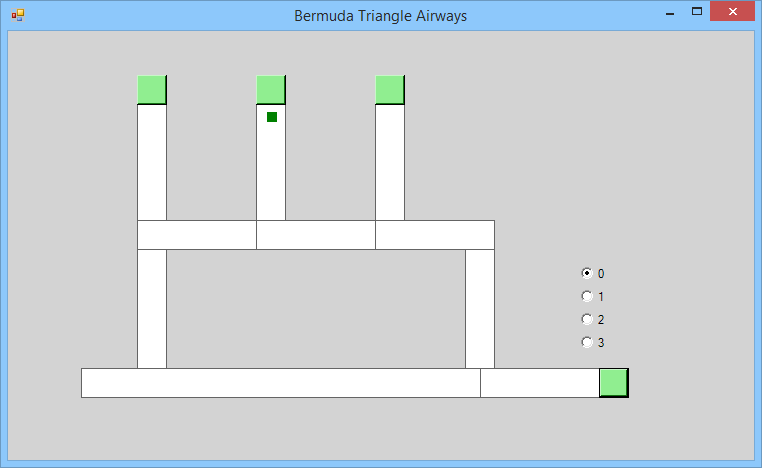
# Server

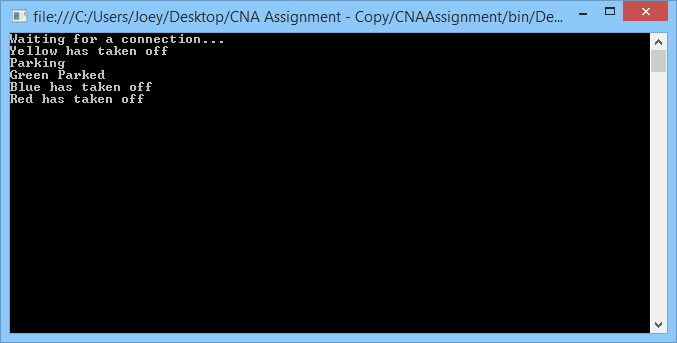
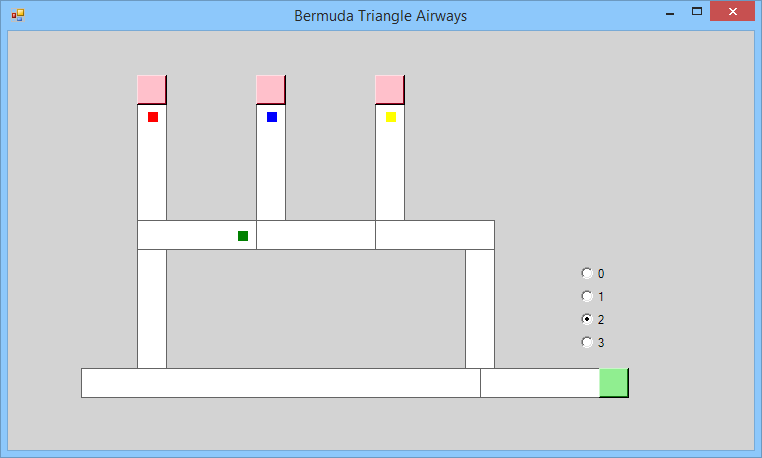
The server must be opened before the client can connect otherwise, the client will crash and will need to be rebooted.

# Documentation

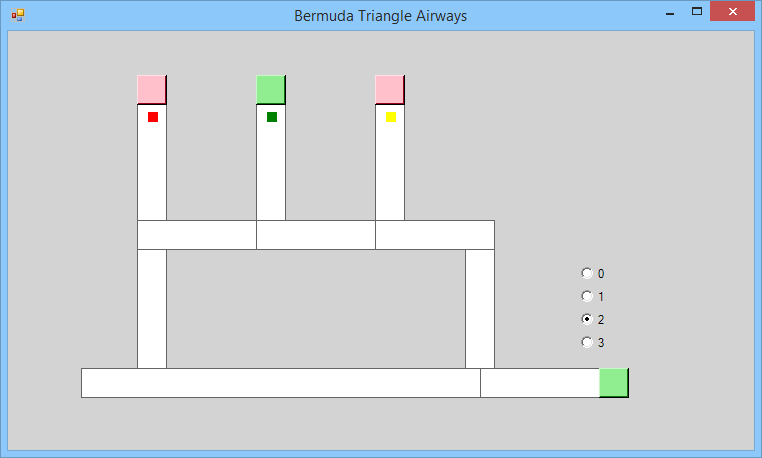




Logging of movements

Waiting for the hub to be emtpy



# Code

using System;

using System.Windows.Forms;

using System.Threading;

using System.ComponentModel;

using System.Collections;

using System.Data;

using System.Drawing;

using System.IO;

using System.Net;

using System.Net.Sockets;

using System.Text;

public class Form1 : Form

{

private Container components = null;

private ButtonPanelThread h1, h2, h3, pArrival;

private Button btn1;

private WaitPanelThread p1, p2, p3, pExit, pEntrance, pRunway;

private Thread thread1, thread2, thread3, thread4, thread5, thread6, thread7, thread8, thread9, thread10;

private Semaphore semaphore\_h1, semaphore\_h2, semaphore\_h3, semaphore\_p1, semaphore\_p2, semaphore\_p3, semaphore\_pExit, semaphore\_pEntrance, semaphore\_pRunway, semaphore\_pArrival;

private Thread semaphore\_h1thread, semaphore\_h2thread, semaphore\_h3thread, semaphore\_p1thread, semaphore\_p2thread, semaphore\_p3thread, semaphore\_pExitthread, semaphore\_pEntrancethread, semaphore\_pRunwaythread, semaphore\_pArrivalthread;

private Buffer bufferp1, bufferp2, bufferp3, bufferh1, bufferh2, bufferh3, bufferpExit, bufferpEntrance, bufferpArrival, bufferpRunway;

private Thread buffThread1, buffThread2, buffThread3, buffThread4, buffThread5, buffThread6, buffThread7, buffThread8, buffThread9, buffThread10;

private Panel hub1, pnl1, pnlArrival;

private Button btn2;

private Panel hub2;

private Button btn3;

private Panel pnlExit;

private Panel pnl2;

private Panel panel4;

private Panel pnl3;

private Panel hub3;

private Panel pnlRunway;

private Panel pnlEntrance;

public RadioButton rad0;

public RadioButton rad1;

public RadioButton rad2;

public RadioButton rad3;

private Panel pRadio;

private Button btnArrival;

public TcpServer host;

public Thread hostThread;

private Panel destination;

private Panel arrivalDestination;

public Form1()

{

InitializeComponent();

semaphore\_h1 = new Semaphore();

semaphore\_h2 = new Semaphore();

semaphore\_h3 = new Semaphore();

semaphore\_p1 = new Semaphore();

semaphore\_p2 = new Semaphore();

semaphore\_p3 = new Semaphore();

semaphore\_pExit = new Semaphore();

semaphore\_pEntrance = new Semaphore();

semaphore\_pRunway = new Semaphore();

semaphore\_pArrival = new Semaphore();

bufferp1 = new Buffer();

bufferp2 = new Buffer();

bufferp3 = new Buffer();

bufferh1 = new Buffer();

bufferh2 = new Buffer();

bufferh3 = new Buffer();

bufferpExit = new Buffer();

bufferpEntrance = new Buffer();

bufferpRunway = new Buffer();

bufferpArrival = new Buffer();

destination = pnlRunway;

if (rad0.Checked)

arrivalDestination = pnlRunway;

if (rad1.Checked)

arrivalDestination = hub1;

if (rad2.Checked)

arrivalDestination = hub2;

if (rad3.Checked)

arrivalDestination = hub3;

arrivalDestination = hub2; // <<<<<<<< Change this to alter landing destination

///Button Thread Order

///(

/// Point origin,

// int delay,

// bool westEast,

// bool northSouth,

// Panel panel,

// Color colour,

// Panel destination,

// Semaphore semaphoreInput,

// Semaphore semaphoreOutput,

// Buffer bufferInput,

// Buffer bufferOutput,

// Button btn,

// Server host

//)

///

h1 = new ButtonPanelThread(new Point(10, 10),

100, false, true, hub1,

Color.Red,

pnlRunway,

semaphore\_h1,

semaphore\_p1,

bufferh1,

bufferp1,

btn1,

host);

h2 = new ButtonPanelThread(new Point(10, 10),

100, false, true, hub2,

Color.Blue,

pnlRunway,

semaphore\_h2,

semaphore\_p2,

bufferh2,

bufferp2,

btn2,

host);

h3 = new ButtonPanelThread(new Point(10, 10),

100, false, true, hub3,

Color.Yellow,

pnlRunway,

semaphore\_h3,

semaphore\_p3,

bufferh3,

bufferp3,

btn3,

host);

pArrival = new ButtonPanelThread(new Point(100, 10),

100, false, false, pnlArrival,

Color.Green,

arrivalDestination, //set to pnlArrival, add switch to detect radioButton selected

semaphore\_pArrival,

semaphore\_pRunway,

bufferpArrival,

bufferpRunway,

btnArrival,

host);

//Wait Thread Order

////( Point origin,

// int delay,

// bool westEast,

// bool northSouth,

// Panel panel,

// Color colour,

// Panel destination,

// Semaphore semaphoreInput,

// Semaphore semaphoreOutput,

// Semaphore semaphoreParking,

// Buffer bufferInput,

// Buffer bufferOutput,

// Buffer bufferParking,

// Server host

// )

/////

p1 = new WaitPanelThread(new Point(10, 10),

100, true, false, pnl1,

Color.White,

destination,

semaphore\_p1,

semaphore\_p2,

semaphore\_h2,

bufferp1,

bufferp2,

bufferh2,

host);

p2 = new WaitPanelThread(new Point(10, 10),

100, true, false, pnl2,

Color.White,

destination,

semaphore\_p2,

semaphore\_p3,

semaphore\_h3,

bufferp2,

bufferp3,

bufferh3,

host);

p3 = new WaitPanelThread(new Point(10, 10),

100, true, false, pnl3,

Color.White,

destination,

semaphore\_p3,

semaphore\_pExit,

null,

bufferp3,

bufferpExit,

null,

host);

pExit = new WaitPanelThread(new Point(10, 10),

100, false, true, pnlExit,

Color.White,

destination,

semaphore\_pExit,

semaphore\_pRunway,

null,

bufferpExit,

bufferpRunway,

null,

host);

pRunway = new WaitPanelThread(new Point(380, 10),

100, false, false, pnlRunway,

Color.White,

destination,

semaphore\_pRunway,

semaphore\_pEntrance,

null,

bufferpRunway,

bufferpEntrance,

null,

host);

pEntrance = new WaitPanelThread(new Point(10, 100),

100, false, false, pnlEntrance,

Color.White,

destination,

semaphore\_pEntrance,

semaphore\_p1,

semaphore\_h1,

bufferpEntrance,

bufferp1,

bufferh1,

host);

host = new TcpServer();

hostThread = new Thread(new ThreadStart(host.Start));

hostThread.Start();

semaphore\_h1thread = new Thread(new ThreadStart(semaphore\_h1.Start));

semaphore\_h2thread = new Thread(new ThreadStart(semaphore\_h2.Start));

semaphore\_h3thread = new Thread(new ThreadStart(semaphore\_h3.Start));

semaphore\_p1thread = new Thread(new ThreadStart(semaphore\_p1.Start));

semaphore\_p2thread = new Thread(new ThreadStart(semaphore\_p2.Start));

semaphore\_p3thread = new Thread(new ThreadStart(semaphore\_p3.Start));

semaphore\_pExitthread = new Thread(new ThreadStart(semaphore\_pExit.Start));

semaphore\_pEntrancethread = new Thread(new ThreadStart(semaphore\_pEntrance.Start));

semaphore\_pRunwaythread = new Thread(new ThreadStart(semaphore\_pRunway.Start));

semaphore\_pArrivalthread = new Thread(new ThreadStart(semaphore\_pArrival.Start));

buffThread1 = new Thread(new ThreadStart(bufferp1.Start));

buffThread2 = new Thread(new ThreadStart(bufferp2.Start));

buffThread3 = new Thread(new ThreadStart(bufferp3.Start));

buffThread4 = new Thread(new ThreadStart(bufferh1.Start));

buffThread5 = new Thread(new ThreadStart(bufferh2.Start));

buffThread6 = new Thread(new ThreadStart(bufferh3.Start));

buffThread7 = new Thread(new ThreadStart(bufferpRunway.Start));

buffThread8 = new Thread(new ThreadStart(bufferpExit.Start));

buffThread9 = new Thread(new ThreadStart(bufferpEntrance.Start));

buffThread10 = new Thread(new ThreadStart(bufferpArrival.Start));

thread1 = new Thread(new ThreadStart(h1.Start));

thread2 = new Thread(new ThreadStart(h2.Start));

thread3 = new Thread(new ThreadStart(h3.Start));

thread4 = new Thread(new ThreadStart(p1.Start));

thread5 = new Thread(new ThreadStart(p2.Start));

thread6 = new Thread(new ThreadStart(p3.Start));

thread7 = new Thread(new ThreadStart(pExit.Start));

thread8 = new Thread(new ThreadStart(pEntrance.Start));

thread9 = new Thread(new ThreadStart(pArrival.Start));

thread10 = new Thread(new ThreadStart(pRunway.Start));

this.Closing += new CancelEventHandler(this.Form1\_Closing);

semaphore\_p1thread.Start();

semaphore\_p2thread.Start();

semaphore\_p3thread.Start();

semaphore\_pExitthread.Start();

semaphore\_pEntrancethread.Start();

semaphore\_pRunwaythread.Start();

semaphore\_pArrivalthread.Start();

buffThread1.Start();

buffThread2.Start();

buffThread3.Start();

buffThread4.Start();

buffThread5.Start();

buffThread6.Start();

buffThread7.Start();

buffThread8.Start();

buffThread9.Start();

buffThread10.Start();

thread1.Start();

thread2.Start();

thread3.Start();

thread4.Start();

thread5.Start();

thread6.Start();

thread7.Start();

thread8.Start();

thread9.Start();

thread10.Start();

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

if (components != null)

components.Dispose();

}

base.Dispose(disposing);

}

private void InitializeComponent()

{

this.hub1 = new System.Windows.Forms.Panel();

this.btn1 = new System.Windows.Forms.Button();

this.pnl1 = new System.Windows.Forms.Panel();

this.pnlArrival = new System.Windows.Forms.Panel();

this.btn2 = new System.Windows.Forms.Button();

this.hub2 = new System.Windows.Forms.Panel();

this.btn3 = new System.Windows.Forms.Button();

this.pnlExit = new System.Windows.Forms.Panel();

this.pnl2 = new System.Windows.Forms.Panel();

this.panel4 = new System.Windows.Forms.Panel();

this.pnl3 = new System.Windows.Forms.Panel();

this.hub3 = new System.Windows.Forms.Panel();

this.pnlRunway = new System.Windows.Forms.Panel();

this.pnlEntrance = new System.Windows.Forms.Panel();

this.btnArrival = new System.Windows.Forms.Button();

this.rad0 = new System.Windows.Forms.RadioButton();

this.rad1 = new System.Windows.Forms.RadioButton();

this.rad2 = new System.Windows.Forms.RadioButton();

this.rad3 = new System.Windows.Forms.RadioButton();

this.pRadio = new System.Windows.Forms.Panel();

this.pnl2.SuspendLayout();

this.pRadio.SuspendLayout();

this.SuspendLayout();

//

// hub1

//

this.hub1.BackColor = System.Drawing.Color.White;

this.hub1.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;

this.hub1.Location = new System.Drawing.Point(129, 70);

this.hub1.Name = "hub1";

this.hub1.Size = new System.Drawing.Size(30, 120);

this.hub1.TabIndex = 0;

//

// btn1

//

this.btn1.BackColor = System.Drawing.Color.Pink;

this.btn1.Location = new System.Drawing.Point(129, 44);

this.btn1.Name = "btn1";

this.btn1.Size = new System.Drawing.Size(30, 30);

this.btn1.TabIndex = 0;

this.btn1.UseVisualStyleBackColor = false;

//

// pnl1

//

this.pnl1.BackColor = System.Drawing.Color.White;

this.pnl1.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;

this.pnl1.Location = new System.Drawing.Point(129, 189);

this.pnl1.Name = "pnl1";

this.pnl1.Size = new System.Drawing.Size(120, 30);

this.pnl1.TabIndex = 1;

//

// pnlArrival

//

this.pnlArrival.BackColor = System.Drawing.Color.White;

this.pnlArrival.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;

this.pnlArrival.Location = new System.Drawing.Point(472, 337);

this.pnlArrival.Name = "pnlArrival";

this.pnlArrival.Size = new System.Drawing.Size(120, 30);

this.pnlArrival.TabIndex = 2;

//

// btn2

//

this.btn2.BackColor = System.Drawing.Color.Pink;

this.btn2.Location = new System.Drawing.Point(248, 44);

this.btn2.Name = "btn2";

this.btn2.Size = new System.Drawing.Size(30, 30);

this.btn2.TabIndex = 3;

this.btn2.UseVisualStyleBackColor = false;

//

// hub2

//

this.hub2.BackColor = System.Drawing.Color.White;

this.hub2.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;

this.hub2.Location = new System.Drawing.Point(248, 70);

this.hub2.Name = "hub2";

this.hub2.Size = new System.Drawing.Size(30, 120);

this.hub2.TabIndex = 4;

//

// btn3

//

this.btn3.BackColor = System.Drawing.Color.Pink;

this.btn3.Location = new System.Drawing.Point(367, 44);

this.btn3.Name = "btn3";

this.btn3.Size = new System.Drawing.Size(30, 30);

this.btn3.TabIndex = 1;

this.btn3.UseVisualStyleBackColor = false;

//

// pnlExit

//

this.pnlExit.BackColor = System.Drawing.Color.White;

this.pnlExit.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;

this.pnlExit.Location = new System.Drawing.Point(457, 218);

this.pnlExit.Name = "pnlExit";

this.pnlExit.Size = new System.Drawing.Size(30, 120);

this.pnlExit.TabIndex = 2;

//

// pnl2

//

this.pnl2.BackColor = System.Drawing.Color.White;

this.pnl2.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;

this.pnl2.Controls.Add(this.panel4);

this.pnl2.Location = new System.Drawing.Point(248, 189);

this.pnl2.Name = "pnl2";

this.pnl2.Size = new System.Drawing.Size(120, 30);

this.pnl2.TabIndex = 2;

//

// panel4

//

this.panel4.BackColor = System.Drawing.Color.White;

this.panel4.Location = new System.Drawing.Point(146, 0);

this.panel4.Name = "panel4";

this.panel4.Size = new System.Drawing.Size(144, 30);

this.panel4.TabIndex = 5;

//

// pnl3

//

this.pnl3.BackColor = System.Drawing.Color.White;

this.pnl3.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;

this.pnl3.Location = new System.Drawing.Point(367, 189);

this.pnl3.Name = "pnl3";

this.pnl3.Size = new System.Drawing.Size(120, 30);

this.pnl3.TabIndex = 6;

//

// hub3

//

this.hub3.BackColor = System.Drawing.Color.White;

this.hub3.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;

this.hub3.Location = new System.Drawing.Point(367, 70);

this.hub3.Name = "hub3";

this.hub3.Size = new System.Drawing.Size(30, 120);

this.hub3.TabIndex = 5;

//

// pnlRunway

//

this.pnlRunway.BackColor = System.Drawing.Color.White;

this.pnlRunway.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;

this.pnlRunway.Location = new System.Drawing.Point(73, 337);

this.pnlRunway.Name = "pnlRunway";

this.pnlRunway.Size = new System.Drawing.Size(400, 30);

this.pnlRunway.TabIndex = 6;

//

// pnlEntrance

//

this.pnlEntrance.BackColor = System.Drawing.Color.White;

this.pnlEntrance.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;

this.pnlEntrance.Location = new System.Drawing.Point(129, 218);

this.pnlEntrance.Name = "pnlEntrance";

this.pnlEntrance.Size = new System.Drawing.Size(30, 120);

this.pnlEntrance.TabIndex = 3;

//

// btnArrival

//

this.btnArrival.BackColor = System.Drawing.Color.Pink;

this.btnArrival.Location = new System.Drawing.Point(591, 337);

this.btnArrival.Name = "btnArrival";

this.btnArrival.Size = new System.Drawing.Size(30, 30);

this.btnArrival.TabIndex = 2;

this.btnArrival.UseVisualStyleBackColor = false;

//

// rad0

//

this.rad0.AutoSize = true;

this.rad0.Checked = true;

this.rad0.Location = new System.Drawing.Point(16, 3);

this.rad0.Name = "rad0";

this.rad0.Size = new System.Drawing.Size(31, 17);

this.rad0.TabIndex = 0;

this.rad0.TabStop = true;

this.rad0.Text = "0";

this.rad0.UseVisualStyleBackColor = true;

//

// rad1

//

this.rad1.AutoSize = true;

this.rad1.Location = new System.Drawing.Point(16, 26);

this.rad1.Name = "rad1";

this.rad1.Size = new System.Drawing.Size(31, 17);

this.rad1.TabIndex = 1;

this.rad1.Text = "1";

this.rad1.UseVisualStyleBackColor = true;

//

// rad2

//

this.rad2.AutoSize = true;

this.rad2.Location = new System.Drawing.Point(16, 49);

this.rad2.Name = "rad2";

this.rad2.Size = new System.Drawing.Size(31, 17);

this.rad2.TabIndex = 2;

this.rad2.Text = "2";

this.rad2.UseVisualStyleBackColor = true;

//

// rad3

//

this.rad3.AutoSize = true;

this.rad3.Location = new System.Drawing.Point(16, 72);

this.rad3.Name = "rad3";

this.rad3.Size = new System.Drawing.Size(31, 17);

this.rad3.TabIndex = 3;

this.rad3.Text = "3";

this.rad3.UseVisualStyleBackColor = true;

//

// pRadio

//

this.pRadio.Controls.Add(this.rad3);

this.pRadio.Controls.Add(this.rad0);

this.pRadio.Controls.Add(this.rad1);

this.pRadio.Controls.Add(this.rad2);

this.pRadio.Location = new System.Drawing.Point(557, 231);

this.pRadio.Name = "pRadio";

this.pRadio.Size = new System.Drawing.Size(64, 100);

this.pRadio.TabIndex = 7;

//

// Form1

//

this.BackColor = System.Drawing.Color.LightGray;

this.ClientSize = new System.Drawing.Size(746, 419);

this.Controls.Add(this.btn1);

this.Controls.Add(this.hub1);

this.Controls.Add(this.btn3);

this.Controls.Add(this.pnlRunway);

this.Controls.Add(this.pnlEntrance);

this.Controls.Add(this.btnArrival);

this.Controls.Add(this.hub3);

this.Controls.Add(this.pnl3);

this.Controls.Add(this.pnlExit);

this.Controls.Add(this.pnl2);

this.Controls.Add(this.btn2);

this.Controls.Add(this.hub2);

this.Controls.Add(this.pnl1);

this.Controls.Add(this.pnlArrival);

this.Controls.Add(this.pRadio);

this.Name = "Form1";

this.Text = "Bermuda Triangle Airways";

this.Closing += new System.ComponentModel.CancelEventHandler(this.Form1\_Closing);

this.pnl2.ResumeLayout(false);

this.pRadio.ResumeLayout(false);

this.pRadio.PerformLayout();

this.ResumeLayout(false);

}

//public Panel getSelectedRadioButton

//{

// get

// {

// if (rad0.Checked)

// return pnlRunway;

// if (rad1.Checked)

// return hub1;

// if (rad2.Checked)

// return hub2;

// if (rad3.Checked)

// return hub3;

// return hub3;

// }

//}

private void Form1\_Closing(object sender, CancelEventArgs e)

{

// Environment is a System class.

// Kill off all threads on exit.

Environment.Exit(Environment.ExitCode);

}

}// end class form1

public class Buffer

{

private Color planeColor;

private Panel destination;

private bool empty = true;

public void Read(ref Color planeColor, ref Panel destination)

{

lock (this)

{

// Check whether the buffer is empty.

if (empty)

Monitor.Wait(this);

empty = true;

planeColor = this.planeColor;

destination = this.destination;

Monitor.Pulse(this);

}

}

public void Write(Color planeColor, Panel destination)

{

lock (this)

{

// Check whether the buffer is full.

if (!empty)

Monitor.Wait(this);

empty = false;

this.planeColor = planeColor;

this.destination = destination;

Monitor.Pulse(this);

}

}

public void Start()

{

}

}// end class Buffer

public class Semaphore

{

private int count = 0;

public void Wait()

{

lock (this)

{

while (count == 0)

Monitor.Wait(this);

count = 0;

}

}

public void Signal()

{

lock (this)

{

count = 1;

Monitor.Pulse(this);

}

}

public void Start()

{

}

}// end class Semaphore

public class ButtonPanelThread

{

private Point origin;

private int delay;

private Panel panel;

private bool westEast;

private bool northSouth;

private Color colour;

private Point plane;

private int xDelta;

private int yDelta;

private Panel destination;

private Semaphore semaphoreInput;

private Semaphore semaphoreOutput;

private Buffer bufferInput;

private Buffer bufferOutput;

private Button btn;

private bool locked = true;

private TcpServer host;

public ButtonPanelThread(Point origin,

int delay,

bool westEast,

bool northSouth,

Panel panel,

Color colour,

Panel destination,

Semaphore semaphoreInput,

Semaphore semaphoreOutput,

Buffer bufferInput,

Buffer bufferOutput,

Button btn,

TcpServer host)

{

this.origin = origin;

this.delay = delay;

this.westEast = westEast;

this.panel = panel;

this.colour = colour;

this.destination = destination;

this.plane = origin;

this.panel.Paint += new PaintEventHandler(this.panel\_Paint);

if (northSouth)

this.yDelta = +5;

else

this.xDelta = westEast ? +5 : -5;

this.semaphoreInput = semaphoreInput;

this.semaphoreOutput = semaphoreOutput;

this.bufferInput = bufferInput;

this.bufferOutput = bufferOutput;

this.btn = btn;

this.host = host;

//if (this.panel.Name == "pnlArrival")

//{

// destination = getSelectedRadioButton();

//}

this.btn.Click += new System.

EventHandler(this.btn\_Click);

}

private void btn\_Click(object sender,

System.EventArgs e)

{

locked = !locked;

this.btn.BackColor = locked ? Color.Pink : Color.LightGreen;

lock (this)

{

if (!locked)

Monitor.Pulse(this);

}

}

public void Start()

{

Color signal = Color.Red;

Thread.Sleep(delay);

this.zeroPlane();

panel.Invalidate();

lock (this)

{

while (locked)

{

Monitor.Wait(this);

}

}

for (int i = 1; i <= 16; i++)

{

this.movePlane(xDelta, yDelta);

Thread.Sleep(delay);

panel.Invalidate();

}

semaphoreOutput.Wait();

bufferOutput.Write(this.colour, this.destination);

this.colour = Color.White;

panel.Invalidate();

//wait for plane to park

semaphoreInput.Signal();

////semaphoreOutput.Wait();

//Thread.Sleep(delay);

bufferInput.Read(ref this.colour, ref this.destination);

//this.zeroPlane();

//panel.Invalidate();

for (int i = 1; i <= 16; i++)

{

this.movePlane(-xDelta, -yDelta);

Thread.Sleep(delay);

panel.Invalidate();

}

}

private void zeroPlane()

{

plane.X = origin.X;

plane.Y = origin.Y;

}

private void movePlane(int xDelta, int yDelta)

{

plane.X += xDelta; plane.Y += yDelta;

}

private void panel\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

SolidBrush brush = new SolidBrush(colour);

g.FillRectangle(brush, plane.X, plane.Y, 10, 10);

brush.Dispose(); // Dispose graphics resources.

g.Dispose(); //

}

}// end class ButtonPanelThread

public class WaitPanelThread

{

private Point origin;

private int delay;

private Panel panel;

private bool westEast;

private bool northSouth;

private Color colour;

private Panel destination;

private Point plane;

private int xDelta;

private int yDelta;

private Semaphore semaphoreInput;

private Semaphore semaphoreOutput;

private Semaphore semaphoreParking;

private Buffer bufferInput;

private Buffer bufferOutput;

private Buffer bufferParking;

private TcpServer host;

public WaitPanelThread(Point origin,

int delay,

bool westEast,

bool northSouth,

Panel panel,

Color colour,

Panel destination,

Semaphore semaphoreInput,

Semaphore semaphoreOutput,

Semaphore semaphoreParking,

Buffer bufferInput,

Buffer bufferOutput,

Buffer bufferParking,

TcpServer host

)

{

this.origin = origin;

this.delay = delay;

this.westEast = westEast;

this.panel = panel;

this.colour = colour;

this.destination = destination;

this.plane = origin;

this.panel.Paint += new PaintEventHandler(this.panel\_Paint);

if (this.panel.Name == "pnlEntrance")

{

this.yDelta = -5;

}

else if (northSouth)

this.yDelta = +5;

else

this.xDelta = westEast ? +5 : -5;

if (this.panel.Name == "pnlRunway")

{

this.xDelta = -20;

}

this.semaphoreInput = semaphoreInput;

this.semaphoreOutput = semaphoreOutput;

this.semaphoreParking = semaphoreParking;

this.bufferInput = bufferInput;

this.bufferOutput = bufferOutput;

this.bufferParking = bufferParking;

this.host = host;

}

public void Start()

{

//Thread.Sleep(delay);

this.colour = Color.White;

for (int k = 1; k <= 100; k++)

{

semaphoreInput.Signal();

this.zeroPlane();

bufferInput.Read(ref this.colour, ref this.destination);

for (int i = 1; i <= 18; i++)

{

panel.Invalidate();

this.movePlane(xDelta, yDelta);

Thread.Sleep(delay);

}

//check if the plane has arrived

if (this.panel.Name == this.destination.Name)

{

Console.WriteLine(this.colour.Name+" has taken off");

}

else

//check it it's the next panel

{

if (

(this.panel.Name == "pnlEntrance" && this.destination.Name.Equals("hub1")) ||

(this.panel.Name == "pnl1" && this.destination.Name.Equals("hub2")) ||

(this.panel.Name == "pnl2" && this.destination.Name.Equals("hub3"))

)

{

Console.WriteLine("Parking");

semaphoreParking.Wait();

bufferParking.Write(this.colour, this.destination);

Console.WriteLine(this.colour.Name+ " Parked");

}

else

{

semaphoreOutput.Wait();

bufferOutput.Write(this.colour, this.destination);

}

}

this.colour = Color.White;

panel.Invalidate();

}

this.colour = Color.Gray;

panel.Invalidate();

}

private void zeroPlane()

{

plane.X = origin.X;

plane.Y = origin.Y;

}

private void movePlane(int xDelta, int yDelta)

{

plane.X += xDelta; plane.Y += yDelta;

}

private void panel\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

SolidBrush brush = new SolidBrush(colour);

g.FillRectangle(brush, plane.X, plane.Y, 10, 10);

brush.Dispose(); // Dispose graphics resources.

g.Dispose(); //

}

}// end class WaitPanelThread

public class TheOne

{

public static void Main()//

{

Application.Run(new Form1());

}

}// end class TheOne

public class TcpServer

{

public void Start()

{

TcpListener server = null;

try

{

IPAddress localAddr = IPAddress.Parse("127.0.0.1");

// Set the TcpListener to port 5000.

Int32 port = 5000;

server = new TcpListener(localAddr, port);

// Start listening for client requests.

server.Start();

// Buffer for reading data

Byte[] bytes = new Byte[256];

String data = null;

// Enter the listening loop.

while (true)

{

Console.WriteLine("Waiting for a connection... ");

// Perform a blocking call to accept requests.

// server.AcceptSocket() could also be called here.

TcpClient client = server.AcceptTcpClient();

Console.WriteLine("Got connection!");

data = null;

// Get a stream object for reading and writing

NetworkStream stream = client.GetStream();

int i;

// Loop to receive all the data sent by the client.

while ((i = stream.Read(bytes, 0, bytes.Length)) != 0)

{

// Translate data bytes to a ASCII string.

data = System.Text.Encoding.ASCII.GetString(bytes, 0, i);

Console.WriteLine("Received: {0}", data);

// Process the data sent by the client.

data = data.ToUpper();

byte[] msg = System.Text.Encoding.ASCII.GetBytes(data);

// Send back a response.

stream.Write(msg, 0, msg.Length);

Console.WriteLine("Sent: {0}", data);

}

// Shutdown and end connection

client.Close();

}

}

catch (SocketException e)

{

Console.WriteLine("SocketException: {0}", e);

}

finally

{

// Stop listening for new clients.

server.Stop();

}

Console.WriteLine("\nServer exiting");

Console.ReadLine();

}

public void Send(String msg)

{

}

}

public static void Main(String[] args)

{

Thread client = new Thread(new ThreadStart(StartClient));

Console.WriteLine("Client started");

client.Start();

//Application.Run(new ConsoleClient());

Console.ReadLine();

}

private void InitializeComponent()

{

this.listBox1 = new System.Windows.Forms.ListBox();

this.label1 = new System.Windows.Forms.Label();

this.label2 = new System.Windows.Forms.Label();

this.label3 = new System.Windows.Forms.Label();

this.checkBox1 = new System.Windows.Forms.CheckBox();

this.checkBox2 = new System.Windows.Forms.CheckBox();

this.checkBox3 = new System.Windows.Forms.CheckBox();

this.SuspendLayout();

//

// listBox1

//

this.listBox1.FormattingEnabled = true;

this.listBox1.Location = new System.Drawing.Point(32, 12);

this.listBox1.Name = "listBox1";

this.listBox1.Size = new System.Drawing.Size(400, 134);

this.listBox1.TabIndex = 0;

//

// label1

//

this.label1.AutoSize = true;

this.label1.Location = new System.Drawing.Point(488, 33);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(57, 13);

this.label1.TabIndex = 1;

this.label1.Text = "Red Plane";

//

// label2

//

this.label2.AutoSize = true;

this.label2.Location = new System.Drawing.Point(488, 72);

this.label2.Name = "label2";

this.label2.Size = new System.Drawing.Size(58, 13);

this.label2.TabIndex = 2;

this.label2.Text = "Blue Plane";

//

// label3

//

this.label3.AutoSize = true;

this.label3.Location = new System.Drawing.Point(488, 112);

this.label3.Name = "label3";

this.label3.Size = new System.Drawing.Size(66, 13);

this.label3.TabIndex = 3;

this.label3.Text = "Green Plane";

//

// checkBox1

//

this.checkBox1.AutoSize = true;

this.checkBox1.Location = new System.Drawing.Point(582, 32);

this.checkBox1.Name = "checkBox1";

this.checkBox1.Size = new System.Drawing.Size(15, 14);

this.checkBox1.TabIndex = 4;

this.checkBox1.UseVisualStyleBackColor = true;

//

// checkBox2

//

this.checkBox2.AutoSize = true;

this.checkBox2.Location = new System.Drawing.Point(582, 71);

this.checkBox2.Name = "checkBox2";

this.checkBox2.Size = new System.Drawing.Size(15, 14);

this.checkBox2.TabIndex = 5;

this.checkBox2.UseVisualStyleBackColor = true;

//

// checkBox3

//

this.checkBox3.AutoSize = true;

this.checkBox3.Location = new System.Drawing.Point(582, 111);

this.checkBox3.Name = "checkBox3";

this.checkBox3.Size = new System.Drawing.Size(15, 14);

this.checkBox3.TabIndex = 6;

this.checkBox3.UseVisualStyleBackColor = true;

//

// ConsoleClient

//

this.ClientSize = new System.Drawing.Size(793, 239);

this.Controls.Add(this.checkBox3);

this.Controls.Add(this.checkBox2);

this.Controls.Add(this.checkBox1);

this.Controls.Add(this.label3);

this.Controls.Add(this.label2);

this.Controls.Add(this.label1);

this.Controls.Add(this.listBox1);

this.Name = "ConsoleClient";

this.ResumeLayout(false);

this.PerformLayout();

}

}