# References


[1]. http://www.metroroadsystems.com/Parking-Management-Systems.shtml

[2]. http://www.asiri.lk/aboutus.htm

[3]. http://spinroot.com/spin/Doc/course/Standish_Survey.htm

[4]. Ian Sommerville (7th Edition), Software Engineering, Software Process Models

[5]. http://www.propark.com/

[6]. http://www.cs.umb.edu/~dqg/papers/uml.doc

[7]. http://www.delopt.co.in/downloads/Parking Management and Guidance System.pdf

[8]. http://ntl.bts.gov/lib/jpodocs/repts_te/14318_files/14318.pdf

[9]. http://onlinepubs.trb.org/onlinepubs/circulars/ec014.pdf

[10]. www.cs.ucl.ac.uk/staff/M.Sewell/stochastic-processes.pdfSimilar

**Activity Diagram of the Existing System**

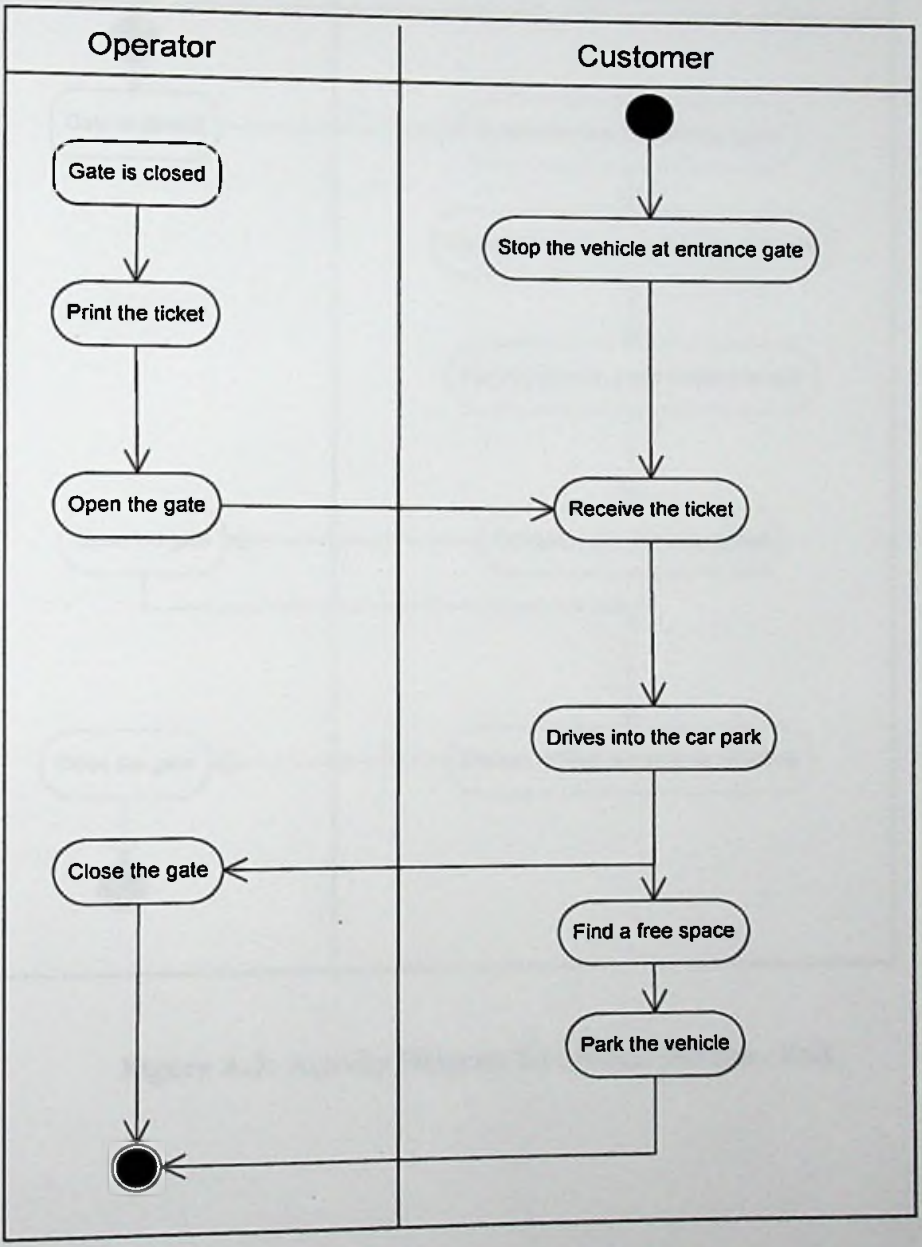


**Figure A.1**: Activity diagram for the manual process - Entrance

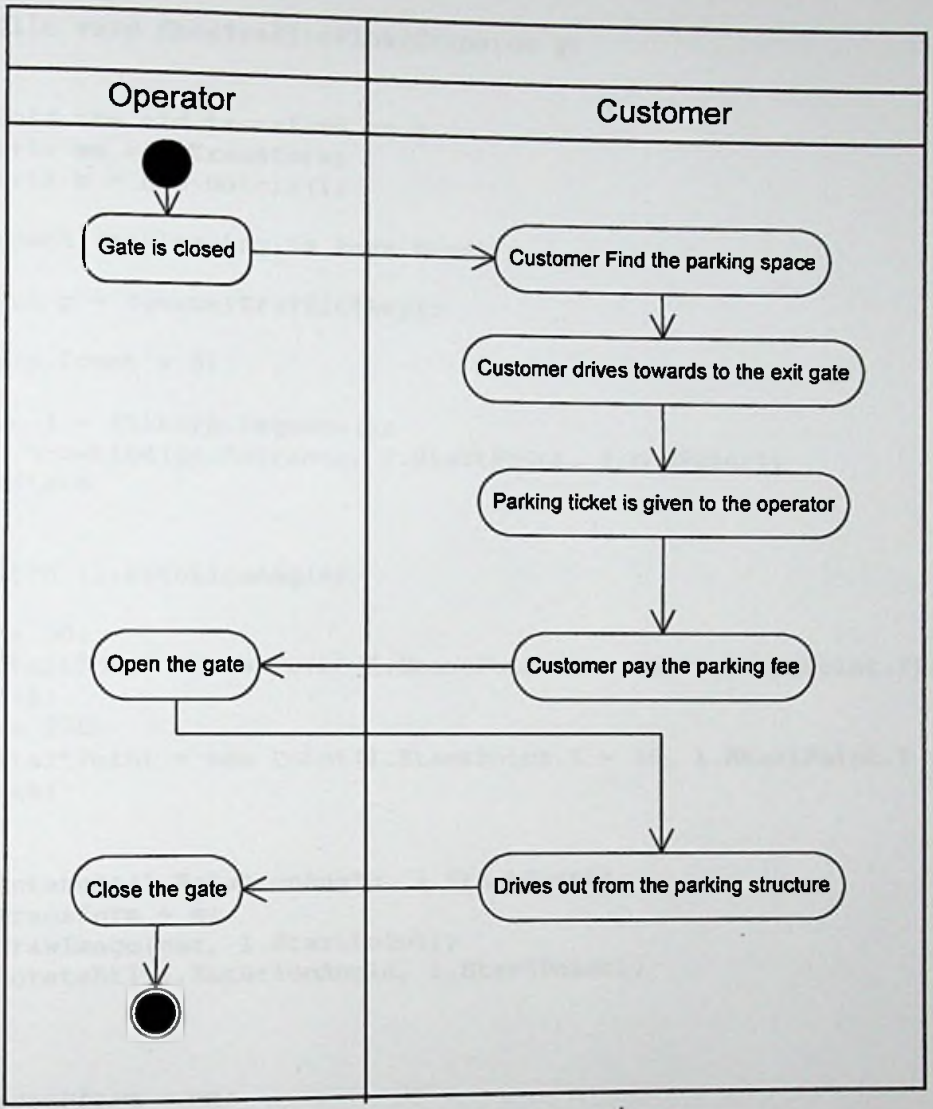


**Figure A.2:** Activity Diagram for manual process - Exit

**ShowTrafficFlow function**

```
public void ShowTrafficFlow(Graphics g)
{

//Hold the old transform of g
Matrix mm = g.Transform;
Matrix m = new Matrix();

foreach (string key in keys.Keys)
{
Queue p = (Queue)Traffic[key];

if (p.Count > 0)
{
Line l = (Line)p.Dequeue();
//g.DrawLine(penEntrance, l.StartPoint, l.EndPoint);
//rotate


switch (l.RotationAngle)
{
case 90:
l.StartPoint = new Point(l.StartPoint.X + 10, l.StartPoint.Y);
break;
case 270:
l.StartPoint = new Point(l.StartPoint.X - 10, l.StartPoint.Y + 10);
break;
}

m.RotateAt(l.RotationAngle, l.StartPoint);
g.Transform = m;
g.DrawImage(car, l.StartPoint);
m.RotateAt(-l.RotationAngle, l.StartPoint);

}

g.Transform = mm;
}
}
```

**InboundTraffic function**

```
public void InboundTraffic(string slotName , string trafficKey)
{
Path = new Queue();
//A1
Point p = new Point(0, 0);
if (!slotName.Equals(string.Empty))
{
p = (Point)SlotInfo[slotName];
if (!allocatedTemp.Contains(slotName))
allocatedTemp.Add(slotName, p);
}
SolidBrush sb = new SolidBrush(Color.LimeGreen);

if (LevelSettings.IsInOut)
{
//Path A
if (slotName.StartsWith("A"))
{
Path.Enqueue(new Line(new Point(pointIn.X - 100, pointIn.Y + 15), new
Point(pointIn.X , pointIn.Y + 15)));
Path.Enqueue(new Line(new Point(pointIn.X, pointIn.Y + 15), new
Point(pointIn.X, pointIn.Y - levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointIn.X, pointIn.Y -
levelSettings.Width / 4), new Point(p.X, pointIn.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(p.X, pointIn.Y - levelSettings.Width
/ 4), new Point(p.X, p.Y)));
}
else
{
//Path B
Path.Enqueue(new Line(new Point(pointIn.X - 100, pointIn.Y + 15), new
Point(pointIn.X, pointIn.Y + 15)));
Path.Enqueue(new Line(new Point(pointIn.X, pointIn.Y + 15), new
Point(pointIn.X, pointIn.Y - levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointIn.X, pointIn.Y -
levelSettings.Width / 4), new Point(pointUp.X, pointIn.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointUp.X, pointIn.Y -
levelSettings.Width / 4), new Point(pointUp.X, pointIn.Y - 3 *
levelSettings.Width / 4)));

if (slotName.Equals(string.Empty))
{
Path.Enqueue(new Line(new Point(pointUp.X, pointIn.Y - 3 *
levelSettings.Width / 4), new Point(pointUp1.X, pointIn.Y - 3 *
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointUp1.X, pointIn.Y - 3 *
levelSettings.Width / 4), new Point(pointUp1.X, pointUp1.Y)));
}
else
{
Path.Enqueue(new Line(new Point(pointUp.X, pointIn.Y - 3 *
levelSettings.Width / 4), new Point(p.X, pointIn.Y - 3 *
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(p.X, pointIn.Y - 3 *
levelSettings.Width / 4), new Point(p.X, p.Y)));
}
}
```

```
}
else if (LevelSettings.IsLastFloor)
{
if (slotName.StartsWith("A"))
{
Path.Enqueue(new Line(new Point(pointUp2.X, pointUp2.Y), new
Point(pointUp2.X, pointUp2.Y + levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointUp2.X, pointUp2.Y +
levelSettings.Width / 4), new Point(p.X, pointUp2.Y +
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(p.X, pointUp2.Y + levelSettings.Width
/ 4), new Point(p.X, p.Y)));
}
else
{
Path.Enqueue(new Line(new Point(pointUp2.X, pointUp2.Y), new
Point(pointUp2.X, pointUp2.Y + levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointUp2.X, pointUp2.Y +
levelSettings.Width / 4), new Point(pointUp.X, pointUp2.Y +
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointUp.X, pointUp2.Y +
levelSettings.Width / 4), new Point(pointUp.X, pointUp2.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointUp.X, pointUp2.Y -
levelSettings.Width / 4), new Point(p.X, pointUp2.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(p.X, pointUp2.Y - levelSettings.Width
/ 4), new Point(p.X, p.Y)));
}
}
else
{
if (slotName.StartsWith("A"))
{
Path.Enqueue(new Line(new Point(pointUp1.X, pointUp1.Y), new
Point(pointUp1.X, pointUp1.Y + levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointUp1.X, pointUp1.Y +
levelSettings.Width / 4), new Point(p.X, pointUp1.Y +
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(p.X, pointUp1.Y + levelSettings.Width
/ 4), new Point(p.X, p.Y)));
}
else
{
Path.Enqueue(new Line(new Point(pointUp1.X, pointUp1.Y), new
Point(pointUp1.X, pointUp1.Y + levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointUp1.X, pointUp1.Y +
levelSettings.Width / 4), new Point(pointUp.X, pointUp1.Y +
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointUp.X, pointUp1.Y +
levelSettings.Width / 4), new Point(pointUp.X, pointUp1.Y -
levelSettings.Width / 4)));

if (slotName.Equals(string.Empty))
{
Path.Enqueue(new Line(new Point(pointUp.X, pointUp1.Y -
levelSettings.Width / 4), new Point(pointUp1.X, pointUp1.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointUp1.X, pointUp1.Y -
levelSettings.Width / 4), new Point(pointUp1.X, pointUp1.Y)));
}
```

```
else
{
Path.Enqueue(new Line(new Point(pointUp.X, pointUp1.Y -
levelSettings.Width / 4), new Point(p.X, pointUp1.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(p.X, pointUp1.Y - levelSettings.Width
/ 4), new Point(p.X, p.Y)));
}
}

}

Path = GetLineSegments(Path);
keys.Add(trafficKey, trafficKey);
Traffic.Add(trafficKey, Path);
}
```

**OutboundTraffic function**

```
public void OutboundTraffic(string slotName, string trafficKey)
{

Path = new Queue();

Point p = new Point(0, 0);
if (!slotName.Equals(string.Empty))
{
p = (Point)SlotInfo[slotName];
}

SolidBrush sb = new SolidBrush(Color.Red);

if (LevelSettings.IsInOut)
{
//Path A
if (slotName.StartsWith("A"))
{
Path.Enqueue(new Line(new Point(p.X, p.Y),new Point(p.X, pointOut.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(p.X, pointOut.Y - levelSettings.Width
/ 4), new Point(pointOut.X, pointOut.Y - levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointOut.X, pointOut.Y -
levelSettings.Width / 4), new Point(pointOut.X, pointOut.Y + 35)));
Path.Enqueue(new Line( new Point(pointOut.X, pointOut.Y + 35), new
Point(pointOut.X - 100, pointOut.Y + 35)));

}
else
{
//Path B

if (slotName.Equals(string.Empty))
{
Path.Enqueue(new Line(new Point(pointDown1.X, pointDown1.Y), new
Point(pointDown1.X, pointOut.Y - 3 * levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointDown1.X, pointOut.Y - 3 *
levelSettings.Width / 4), new Point(pointDown.X, pointOut.Y - 3 *
levelSettings.Width / 4)));
```

```
}
else
{
Path.Enqueue(new Line( new Point(p.X, p.Y), new Point(p.X, pointOut.Y
- 3 * levelSettings.Width / 4)));
Path.Enqueue(new Line( new Point(p.X, pointOut.Y - 3 *
levelSettings.Width / 4), new Point(pointDown.X, pointOut.Y - 3 *
levelSettings.Width / 4)));
}

Path.Enqueue(new Line(new Point(pointDown.X, pointOut.Y - 3 *
levelSettings.Width / 4), new Point(pointDown.X, pointOut.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointDown.X, pointOut.Y -
levelSettings.Width / 4), new Point(pointOut.X, pointOut.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointOut.X, pointOut.Y -
levelSettings.Width / 4), new Point(pointOut.X, pointOut.Y + 15)));
Path.Enqueue(new Line(new Point(pointOut.X, pointOut.Y + 15), new
Point(pointOut.X - 100, pointOut.Y + 15)));
}
}
else if (LevelSettings.IsLastFloor)
{
if (slotName.StartsWith("A"))
{
Path.Enqueue(new Line(new Point(p.X, pointDown2.Y +
levelSettings.Width / 4), new Point(p.X, p.Y)));
Path.Enqueue(new Line(new Point(pointDown2.X, pointDown2.Y +
levelSettings.Width / 4), new Point(p.X, pointDown2.Y +
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointDown2.X, pointDown2.Y), new
Point(pointDown2.X, pointDown2.Y + levelSettings.Width / 4)));
}
else
{
Path.Enqueue(new Line(new Point(p.X, pointDown2.Y -
levelSettings.Width / 4), new Point(p.X, p.Y)));
Path.Enqueue(new Line(new Point(pointDown.X, pointDown2.Y -
levelSettings.Width / 4), new Point(p.X, pointDown2.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointDown.X, pointDown2.Y +
levelSettings.Width / 4), new Point(pointDown.X, pointDown2.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointDown2.X, pointDown2.Y +
levelSettings.Width / 4), new Point(pointDown.X, pointDown2.Y +
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointDown2.X, pointDown2.Y), new
Point(pointDown2.X, pointDown2.Y + levelSettings.Width / 4)));
}
}
else
{
if (slotName.StartsWith("A"))
{
Path.Enqueue(new Line(new Point(p.X, p.Y), new Point(p.X, p.Y -
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(p.X, p.Y - levelSettings.Width / 4),
new Point(pointDown1.X, pointDown1.Y + levelSettings.Width / 4)));
```

```
Path.Enqueue(new Line(new Point(pointDown1.X, pointDown1.Y +
levelSettings.Width / 4) , new Point(pointDown1.X, pointDown1.Y)));
}
else
{
if (slotName.Equals(string.Empty))
{
Path.Enqueue(new Line(new Point(pointDown1.X, pointDown1.Y) , new
Point(pointDown1.X, pointDown1.Y - levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointDown1.X, pointDown1.Y -
levelSettings.Width / 4), new Point(pointDown.X, pointDown1.Y -
levelSettings.Width / 4)));

}
else
{
Path.Enqueue(new Line(new Point(p.X, p.Y) , new Point(p.X,
pointDown.Y - levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(p.X, pointDown.Y -
levelSettings.Width / 4) , new Point(pointDown.X, pointDown.Y -
levelSettings.Width / 4)));
}

Path.Enqueue(new Line(new Point(pointDown.X, pointDown.Y -
levelSettings.Width / 4), new Point(pointDown.X, pointDown.Y +
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointDown.X, pointDown.Y +
levelSettings.Width / 4), new Point(pointDown.X, pointDown1.Y +
levelSettings.Width / 4)));
Path.Enqueue(new Line(new Point(pointDown1.X, pointDown1.Y), new
Point(pointDown1.X, pointDown1.Y + levelSettings.Width / 4)));
}

}

Path = GetLineSegments(Path);
keys.Add(trafficKey, trafficKey);
Traffic.Add(trafficKey, Path);
}
```