# บทที่ 5

## การพัฒนาบรรณาธิกรสำหรับสร้างแผนภาพสเตทชาร์ทและตัวสร้างชุดคำสั่ง

เนื้อหาในบทนี้เป็นการแสดงรายละเอียดของคลาสที่พัฒนาขึ้นเพื่อใช้สร้างบรรณาธิกรสำหรับสร้างแผนภาพสเตทชาร์ทและตัวสร้างชุดคำสั่ง โดยอธิบายถึงรายละเอียดของตัวแปรสมาชิกและฟังก์ชันสมาชิกที่สำคัญของแต่ละคลาสต่อไปนี้

### 5.1 คลาส STDFrame

คลาส STDFrame คือคลาสหน้าจอหลักของบรรณาธิกรสำหรับสร้างแผนภาพสเตทชาร์ทซึ่งรายละเอียดของคลาสแสดงในรูปที่ 5.1 โดยมีองค์ประกอบที่สำคัญดังนี้

5.1.1 ตัวแปรสมาชิกที่เป็นปุ่มเครื่องมือวาดแผนภาพสเตทชาร์ท

เครื่องมือวาดแผนภาพสเตทชาร์ทคือตัวแปรสมาชิกที่มีชนิดเป็นคลาส JToggleButton ซึ่งเป็นปุ่มสำหรับเลือก ตัวแปรสมาชิกที่เป็นปุ่มเครื่องมือวาดได้แก่ selectButton stateButton transButton initButton และ finalButton เมื่อปุ่มถูกเลือกจะกำหนดสถานะการวาดให้กับตัวแปร cursorType ตามชนิดที่กำหนดของแต่ละปุ่ม

5.1.2 ตัวแปรสมาชิก statusBar

ตัวแปรสมาชิก statusBar คือแถบแสดงสถานะซึ่งมีชนิดเป็นคลาส JLabel แถบแสดงสถานะจะถูกกำหนดให้แสดงข้อความซึ่งเป็นการบอกสถานะการวาดเมื่อมีการเลือกที่ปุ่มเครื่องมือวาด

5.1.3 ปุ่มควบคุม

ปุ่มควบคุม คือตัวแปรสมาชิก save okey delSelected และ delAll ซึ่งมีชนิดเป็นคลาส JButton และทำหน้าที่ บันทึกแผนภาพ ปิดบรรณาธิกร ลบแผนภาพส่วนที่เลือก และลบแผนภาพทั้งหมดตามลำดับ

5.1.4 ตัวแปรสมาชิก displayPane

ตัวแปรสมาชิก displayPane คือพื้นที่แสดงแผนภาพซึ่งมีชนิดเป็นคลาส JPanel ทำหน้าที่เป็นตัวบรรจุแผนภาพสเตทชาร์ทและแสดงบนหน้าจอของบรรณาธิกร

5.1.5 ฟังก์ชันสมาชิก addRootPane

ฟังก์ชันสมาชิก addRootPane ทำหน้าที่นำวัตถุที่สร้างมาจากคลาส RootPane ให้แสดงในบรรณาธิกร

5.1.6 ฟังก์ชันสมาชิก setRootPane

ฟังก์ชันสมาชิก setRootPane ทำหน้าที่นำวัตถุที่สร้างมาจากคลาส StoreObject มาทำการแปลงเป็นวัตถุที่มีชนิดเป็น RootPane ซึ่งมีองค์ประกอบของแผนภาพสเตทชาร์ทบรรจุอยู่ภายในโดยเรียกใช้ฟังก์ชัน restoreDiagram ในการแปลง แล้วนำวัตถุ RootPaneที่ได้มาแสดงในบรรณาธิกร

5.1.7 ฟังก์ชันสมาชิก restoreDiagram

ฟังก์ชันสมาชิก restoreDiagram ทำหน้าที่แปลงวัตถุที่สร้างมาจากคลาส StoreObject เป็นวัตถุที่มีชนิดเป็น RootPane

5.1.8 ฟังก์ชันสมาชิก classifyObject

ฟังก์ชันสมาชิก classifyObject ทำหน้าที่นำวัตถุที่สร้างมาจากคลาส RootPane มาทำการแปลงเป็นวัตถุที่มีชนิดเป็น StoreObject เพื่อนำไปใช้จัดเก็บในแฟ้มข้อมูลต่อไป

```
public class STDFrame extends JFrame {
.....
String cursorType="select";
JToggleButton selectButton = new JToggleButton();
JToggleButton stateButton = new JToggleButton();
JToggleButton transButton = new JToggleButton();
JToggleButton initButton = new JToggleButton();
JToggleButton finalButton = new JToggleButton();
.....
JLabel statusBar = new JLabel();
JButton save = new JButton();
JButton okey = new JButton();
JButton delAll = new JButton();
JButton delSelected = new JButton();

JPanel displayPane = new JPanel();
.....
public void addRootPane(RootPane rootPane){
   if(this.rootPane!=null){
     this.displayPane.remove(this.rootPane);
   }
   this.rootPane=rootPane;
   rootPane.parent=this;
   this.displayPane.add(rootPane);
   this.displayPane.repaint();
 }
public void setRootPane(StoreObject obj){
    this.restoreDiagram(obj,this.pointer);
    this.rootPane=this.stoRootPane;
    this.addRootPane(this.rootPane);
 }
private void restoreDiagram(StoreObject obj,Container root){
    this.transitionVector.removeAllElements();
    this.constructStateTree(obj,root);                    // instantiate state
    for(int i=0;i<this.transitionVector.size();i++){      // instantiate ExternalTranstion
       StoreObject tmp =(StoreObject)this.transitionVector.elementAt(i);
       if(tmp.objectType.equals("ExTransition")){
         this.findState(tmp.sourceState,this.stoRootPane);
         State source = tempState;// assign the search result
         this.findState(tmp.targetState,this.stoRootPane);
         State target = tempState;
```

รูปที่ 5.1 รายละเอียดของคลาส STDFrame

```
        ExTransition tran = new
          ExTransition(source,tmp.sourcePoint,target,tmp.targetPoint,this.stoRootPane);
        tran.setStoreObject(tmp);

      }else if(tmp.objectType.equals("InTransition")){          // instantiate InternalTranstion
        this.findState(tmp.sourceState,this.stoRootPane);
        State source = tempState;// assign the search result
        InTransition tran = new  InTransition(source);
        tran.setStoreObject(tmp);
      }
    }
  }
public void classifyObject(Container obj){

   if(obj.getComponentCount()<1){
     if(obj instanceof RootPane){
      RootPane tmp=(RootPane)obj;
      this.rootObject=tmp.getStoreObject();
     } else if(obj instanceof RootState){
      RootState tmp=(RootState)obj;
      RootPane parent=(RootPane)obj.getParent();
      StoreObject stoParent=parent.getStoreObject();
      stoParent.add(tmp.getStoreObject());
     } else if(obj instanceof State){
      State tmp=(State)obj;
      BasicState parent=(BasicState)obj.getParent();
      StoreObject stoParent=parent.getStoreObject();
      stoParent.add(tmp.getStoreObject());
     } else if(obj instanceof Transition){
      Transition tmp=(Transition)obj;
      if(obj.getParent()instanceof RootPane){
      RootPane parent=(RootPane)obj.getParent();
      StoreObject stoParent=parent.getStoreObject();
      stoParent.add(tmp.getStoreObject());
      } else if(obj.getParent()instanceof BasicState){
       BasicState parent=(BasicState)obj.getParent();
       StoreObject stoParent=parent.getStoreObject();
       stoParent.add(tmp.getStoreObject());
      }
     }}else{
      ..........
      // Code like above
     for(int i=0;i<obj.getComponentCount();i++){
      this.classifyObject((Container)obj.getComponent(i));
     }
   }
  }
.....
}
```

รูปที่ 5.1 รายละเอียดของคลาส STDFrame (ต่อ)

**5.2 คลาส RootPane**

คลาส RootPane คือคลาสที่ทำหน้าที่เป็นตัวบรรจุองค์ประกอบทั้งหมดของแผนภาพสเตทชาร์ท ซึ่งรายละเอียดของคลาสแสดงในรูปที่ 5.2 โดยมีองค์ประกอบที่สำคัญดังนี้

5.2.1 สถานะราก

สถานะรากคือวัตถุที่สร้างจากคลาส RootState ทำหน้าที่เป็นสถานะตั้งต้นที่อยู่ในระดับสูงสุด ดังนั้นสถานะรากจึงเป็นตัวบรรจุของวัตถุอื่นที่ใช้สร้างเป็นแผนภาพสเตทชาร์ทเช่น สถานะพื้นฐาน หรือการเปลี่ยนแปลง เป็นต้น คลาส RootPane จะมีสถานะรากได้เพียงหนึ่งเดียวเท่านั้น

5.2.2 ตัวแปรสมาชิก methodName และ methodDetail

เนื่องจากบรรณาธิกรสำหรับสร้างแผนภาพสเตทชาร์ทถูกออกแบบให้สามารถกำหนดรายละเอียดให้กับฟังก์ชันสมาชิกได้ ดังนั้นจึงกำหนดให้มีตัวแปรสมาชิก methodName และ methodDetail สำหรับเก็บข้อมูลของชื่อฟังก์ชันสมาชิกและรายละเอียดของฟังก์ชันสมาชิกตามลำดับ ตัวแปรสมาชิกทั้งสองเป็นวัตถุที่สร้างจากคลาส Vector

5.2.3 ฟังก์ชันสมาชิก setStoreObject

ฟังก์ชันสมาชิก setStoreObject ทำหน้าที่นำข้อมูลของวัตถุ StoreObject มากำหนดคุณสมบัติให้กับวัตถุ RootPane

5.2.4 ฟังก์ชันสมาชิก getStoreObject

ฟังก์ชันสมาชิก getStoreObject ทำหน้าที่นำข้อมูลของวัตถุ RootPane มากำหนดให้วัตถุ StoreObject ที่เป็นตัวแปรสมาชิกแล้วส่งวัตถุ StoreObject นี้ให้กับวัตถุที่ร้องขอ

```
public class RootPane extends JPanel {
.....
RootState rootState = new RootState(this);
Vector methodName=new Vector();
Vector methodDetail=new Vector()
StoreObject storeObject = new StoreObject();

public StoreObject getStoreObject(){
   this.storeObject.objectType="RootPane";
   this.storeObject.setBounds(this.getBounds());
   this.storeObject.methodName=this.methodName;
   this.storeObject.methodDetail=this.methodDetail;
   return this.storeObject;
}
public void setStoreObject(StoreObject sto){
   this.setBounds(sto.getBounds());
   this.methodName=sto.methodName;
   this.methodDetail=sto.methodDetail;
   this.storeObject=sto;
}
.....
}
```

รูปที่ 5.2 รายละเอียดของคลาส RootPane

**5.3 คลาส State**

คลาส State คือคลาสนามธรรมที่กำหนดคุณสมบัติพื้นฐานให้กับสถานะซึ่งรายละเอียดของคลาสแสดงในรูปที่ 5.3 โดยมีองค์ประกอบที่สำคัญดังนี้

5.3.1 ตัวแปรสมาชิก name

ตัวแปรสมาชิก name คือตัวแปรสมาชิกที่ใช้เก็บชื่อสถานะซึ่งมีชนิดเป็นคลาส String

5.3.2 ตัวแปรสมาชิก entryT

ตัวแปรสมาชิก entryT ใช้สำหรับเก็บตำแหน่งของทุกการเปลี่ยนแปลงที่เป็นการเปลี่ยนแปลงเข้าสู่สถานะนี้ ตัวแปรสมาชิก entryT มีชนิดเป็น Vector

5.3.3 ตัวแปรสมาชิก exitT

ตัวแปรสมาชิก exitT ใช้สำหรับเก็บตำแหน่งของทุกการเปลี่ยนแปลงที่เป็นการเปลี่ยนแปลงออกจากสถานะนี้ ตัวแปรสมาชิก exitT มีชนิดเป็น Vector

5.3.4 ตัวแปรสมาชิก internalT

ตัวแปรสมาชิก internalT ใช้สำหรับเก็บตำแหน่งของทุกการเปลี่ยนแปลงที่เป็นการเปลี่ยนแปลงภายในของสถานะนี้ ตัวแปรสมาชิก internalT มีชนิดเป็น Vector

5.3.5 ตัวแปรสมาชิก selected

ตัวแปรสมาชิก selected เป็นตัวแปรชนิด boolean ใช้แสดงถึงสถานะการถูกเลือกของสถานะ โดยระบุความหมายดังนี้ ค่าเป็น false หมายถึงสถานะนี้ไม่ถูกเลือก ค่าเป็น true หมายถึงสถานะนี้ถูกเลือก

5.3.6 ฟังก์ชันสมาชิก removeAllTransition

ฟังก์ชันสมาชิก removeAllTransition คือฟังก์ชันที่ทำหน้าที่ลบการเปลี่ยนแปลงที่เกี่ยวข้องกับสถานะทั้งหมดออกจากรายการที่จัดเก็บคือ รายการของการเปลี่ยนแปลงที่เข้าสู่สถานะ รายการของการเปลี่ยนแปลงที่ออกจากสถานะ และรายการของการเปลี่ยนแปลงภายใน

5.3.7 ฟังก์ชันสมาชิก removeSelf

ฟังก์ชันสมาชิก removeSelf คือฟังก์ชันที่ทำหน้าที่ลบตนเองออกจากสถานะพ่อแม่

5.3.8 ฟังก์ชันสมาชิก removeAllSubstate

ฟังก์ชันสมาชิก removeAllSubstate คือฟังก์ชันที่ทำหน้าที่ลบสถานะย่อยทั้งหมดของตนเองออก

5.3.9 ฟังก์ชันสมาชิก setStoreObject

ฟังก์ชันสมาชิก setStoreObject ทำหน้าที่นำข้อมูลของวัตถุ StoreObject มากำหนดคุณสมบัติให้กับวัตถุซึ่งมีชนิดเป็นคลาส State

5.3.10 ฟังก์ชันสมาชิก getStoreObject

ฟังก์ชันสมาชิก getStoreObject ทำหน้าที่นำข้อมูลของวัตถุ State มากำหนดให้วัตถุ StoreObject แล้วส่งวัตถุ StoreObject นี้ให้กับวัตถุที่ร้องขอ

```
public abstract class State extends JPanel {
.....
String name;
Vector entryT = new Vector();
Vector exitT = new Vector();
Vector internalT = new Vector();
Boolean selected=false;
 StoreObject storeObject=new StoreObject();
.....

void removeAllTransition(){
   Transition t;
    for(int i=0;i<=this.entryT.size()-1;i++){
     t =(Transition)this.entryT.elementAt(i);
     t.removeSelf();
    }
    for(int i=0;i<=this.exitT.size()-1;i++){
     t =(Transition)this.exitT.elementAt(i);
     t.removeSelf();
    }
    for(int i=0;i<=this.internalT.size()-1;i++){
     t =(Transition)this.internalT.elementAt(i);
     t.removeSelf();
    }
 }

 public void removeAllSubstate(){
  Vector temp=new Vector();
    for(int i=0;i<=this.getComponentCount()-1;i++){
     if(this.getComponent(i) instanceof State){
       State s =(State)this.getComponent(i);
       temp.add(s);
      }
    }
    for(int i=0;i<=temp.size()-1;i++){
       State s =(State)temp.elementAt(i);
       s.removeSelf();
    }
 }

public StoreObject getStoreObject(){
   this.storeObject.name=name;
   this.storeObject.color=color;
   this.storeObject.setBounds(this.getBounds());
   return this.storeObject;
 }
 public void setStoreObject(StoreObject sto){
   this.name=sto.name;
   this.color=sto.color;
   this.setBounds(sto.getBounds());
   this.storeObject=sto;
 }
```

รูปที่ 5.3 รายละเอียดของคลาส State

```
.....
public void removeSelf(){
 if(this.getComponentCount()<=1){
     this.removeAllTransition();
     Container parent=this.getParent();
     parent.remove(this);
     parent.repaint();
     if(parent instanceof RootPane){
       RootPane temp=(RootPane)parent;
       StoreObject stoParent=temp.getStoreObject();
       stoParent.remove(this.getStoreObject());
    }else{ BasicState temp=(BasicState)parent;
       StoreObject stoParent=temp.getStoreObject();
       stoParent.remove(this.getStoreObject());
    }
 }else{
   for(int i=0;i<=this.getComponentCount()-1;i++){
      if(this.getComponent(i) instanceof State){
        State s =(State)this.getComponent(i);
        s.removeSelf();
      }
    }
    this.removeAllTransition();
    Container parent=this.getParent();
    parent.remove(this);
    parent.repaint();
    if(parent instanceof RootPane){
     RootPane temp=(RootPane)parent;
     StoreObject stoParent=temp.getStoreObject();
     stoParent.remove(this.getStoreObject());
    }else{
     BasicState temp=(BasicState)parent;
     StoreObject stoParent=temp.getStoreObject();
     stoParent.remove(this.getStoreObject());
    }
  }
}
.....
}
```

รูปที่ 5.3 รายละเอียดของคลาส State (ต่อ)

**5.4 คลาส BasicState**

คลาส BasicState คือคลาสซึ่งสืบทอดคุณสมบัติจากคลาส State ทำหน้าที่เป็นคลาสของวัตถุสถานะพื้นฐาน ซึ่งรายละเอียดของคลาสแสดงในรูปที่ 5.4 โดยมีองค์ประกอบที่สำคัญดังนี้

5.4.1 ตัวแปรสมาชิก nameLabel

ตัวแปรสมาชิก nameLabel คือป้ายแสดงสถานะซึ่งมีชนิดเป็นคลาส ExLabel ทำหน้าที่ในการแสดงชื่อของสถานะ

5.4.2 ตัวแปรสมาชิก historyType

ตัวแปรสมาชิก historyType เป็นตัวแปรที่มีชนิดเป็น int ใช้แสดงถึงการกำหนดประเภทของสถานะครั้งก่อนโดยระบุความหมายดังนี้ ค่าเป็น 0 หมายถึงไม่มีการระบุสถานะครั้งก่อน ค่าเป็น 1 หมายถึงมีการระบุสถานะครั้งก่อน ค่าเป็น 3 หมายถึงมีการระบุสถานะลึกครั้งก่อน

5.4.3 ฟังก์ชันสมาชิก paint

ฟังก์ชันสมาชิก paint ทำหน้าที่วาดภาพสัญลักษณ์ของสถานะ

5.4.4 ฟังก์ชันสมาชิก initHistoryState

ฟังก์ชันสมาชิก initHistoryState ทำหน้าที่สร้างสถานะครั้งก่อนให้กับสถานะพื้นฐานและกำหนดค่าให้กับตัวแปรสมาชิก historyType เป็น 1

5.4.5 ฟังก์ชันสมาชิก initDeepHisState

ฟังก์ชันสมาชิก initDeepHisState ทำหน้าที่สร้างสถานะลึกครั้งก่อนให้กับสถานะพื้นฐานและกำหนดค่าให้กับตัวแปรสมาชิก historyType เป็น 2

5.4.6 ฟังก์ชันสมาชิก removeHisState

ฟังก์ชันสมาชิก removeHisState ทำหน้าที่ยกเลิกการกำหนดสถานะครั้งก่อนหรือยกเลิกการกำหนดสถานะลึกครั้งก่อน ตามชนิดที่กำหนด

5.4.7 ฟังก์ชันสมาชิก this_focusGained

ฟังก์ชันสมาชิก this_focusGained จะถูกประมวลผลเมื่อโฟกัสมายังสถานะโดยการคลิกเมาส์ที่สถานะ การประมวลผลจะทำการกำหนดค่าให้ตัวแปร selected เป็น true เพื่อแสดงว่าสถานะนี้ถูกเลือก

5.4.8 ฟังก์ชันสมาชิก this_focusLost

ฟังก์ชันสมาชิก this_focusLost จะถูกประมวลผลเมื่อเมาส์คลิกที่องค์ประกอบอื่นโดยการประมวลผลจะทำการกำหนดค่าให้ตัวแปร selected เป็น false เพื่อแสดงว่าสถานะนี้ไม่ได้ถูกเลือกแล้ว

5.4.9 ฟังก์ชันสมาชิก this_mouseClicked

ฟังก์ชันสมาชิก this_mouseClicked จะถูกประมวลผลเมื่อเมาส์คลิกที่สถานะพื้นฐาน การประมวลผลจะทำการตรวจสอบค่าของตัวแปร cursorType ของคลาส STDFrame ซึ่งเป็นการตรวจสอบสถานะการวาดแผนภาพสเตทชาร์ท

```
public class BasicState extends State {
.....
public void paint(Graphics g){
   g.setColor(color);
   g.fillRoundRect(0,0,this.getWidth()-1,this.getHeight()-1,30,30);
   g.setColor(Color.black);
   g.drawRoundRect(0,0,this.getWidth()-1,this.getHeight()-1,30,30);
   if(selected){
     g.setColor(Color.red);
     g.fillRect(0,0,5,5);
     g.fillRect(this.getWidth()-5,0,5,5);
     g.fillRect(0,this.getHeight()-5,5,5);
     g.fillRect(this.getWidth()-5,this.getHeight()-5,5,5);
     g.fillRect((this.getWidth()/2)-5,(this.getHeight()/2)-5,5,5);
   }
     this.paintChildren(g);
  }
public void removeHisState(){
   if(this.historyType==1){
     this.remove(this.hisState);
     this.hisState=null;
   }else if(this.historyType==2){
     this.remove(this.deepHisState);
     this.deepHisState=null;
   }
     this.historyType=0;
  }
  public void initHistoryState(){
   this.historyType=1;
   hisState = new HistoryState(new Point(10,30),this.rootPane);
   hisState.name="hisOf"+this.name;
   this.add(hisState);
   if(this.deepHisState!=null){
     this.remove(this.deepHisState);
     this.deepHisState=null;
   }
  }
  public void initDeepHisState(){
   this.historyType=2;
   deepHisState = new DeepHisState(new Point(10,40),this.rootPane);
   deepHisState.name="deepOf"+this.name;
   this.add(deepHisState);
   if(this.hisState!=null){
     this.remove(this.hisState);
     this.hisState=null;
   }
  }
  void this_focusGained(FocusEvent e) {
     this.selected=true;
  }
  void this_focusLost(FocusEvent e) {
     this.selected=false;
  }
```

รูปที่ 5.4 รายละเอียดของคลาส BasicState

```
public class BasicState extends State {
.....
void this_mouseClicked(MouseEvent e) {
.....
   if(rootPane.parent.cursorType.equals("init")){
      rootPane.parent.cursorType = "select";
      rootPane.parent.setCursor(Cursor.getDefaultCursor());
      rootPane.parent.selectButton.setSelected(true);
      rootPane.parent.selectButton.requestFocus();
      rootPane.parent.statusBar.setText("Ready");
      InitState init = new InitState(e.getPoint(),this.rootPane);
      this.add(init);
   }else if(rootPane.parent.cursorType equals("final")){
      rootPane.parent.cursorType = "select";
      rootPane.parent.setCursor(Cursor.getDefaultCursor());
      rootPane.parent.selectButton.setSelected(true);
      rootPane.parent.selectButton.requestFocus();
      rootPane.parent.statusBar.setText("Ready");
      FinalState init = new FinalState(e.getPoint(),this.rootPane);
      this.add(init);
   }else if(rootPane.parent.cursorType.equals("trans")){
      rootPane.parent.cursorType = "targetstate";
      rootPane.parent.source = this;
      rootPane.parent.sourcePoint = new Point(e.getX(),e.getY());
      rootPane.parent.statusBar.setText("Select target state");
   }else if(rootPane.parent.cursorType.equals("targetstate")){
      rootPane.parent.cursorType = "select";
      rootPane.parent.setCursor(Cursor.getDefaultCursor());
      rootPane.parent.selectButton.setSelected(true);
      rootPane.parent.selectButton.requestFocus();
      rootPane.parent.target = this;
      rootPane.parent.targetPoint = new Point(e.getX(),e.getY());
      rootPane.parent.statusBar.setText("Ready");
      ExTransition t = new
         ExTransition(rootPane.parent.source,rootPane.parent.sourcePoint,rootPane.parent.target,rootPane.parent.tar
getPoint,this.rootPane);
   }else if(rootPane.parent.cursorType.equals("state")){
      rootPane.parent.cursorType = "select";
      rootPane.parent.setCursor(Cursor.getDefaultCursor());
      rootPane.parent.selectButton.setSelected(true);
      rootPane.parent.selectButton.requestFocus();
      rootPane.parent.statusBar.setText("Ready");
      rootPane.parent.source=this;
      BasicState state = new
         BasicState("No name",this.rootPane,new Rectangle(e.getX(),e.getY(),80,60));
      this.add(state);
   }
   this.rootPane.parent.change=true;
}
.....
}
```

รูปที่ 5.4 รายละเอียดของคลาส BasicState (ต่อ)

**5.5 คลาส RootState**

คลาส RootState คือคลาสที่ทำหน้าที่เป็นสถานะราก คลาส RootState สืบทอดคุณสมบัติจากคลาส BasicState ดังนั้นคุณสมบัติของคลาส RootState จึงเหมือนกับกับคลาส BasicState ซึ่งอธิบายไว้แล้วในข้อ 5.4 มีเพียงการกำหนดสีในการแสดงผลเท่านั้นที่แตกต่างจากสถานะพื้นฐาน

**5.6 คลาส Transition**

คลาส Transition คือคลาสนามธรรมที่กำหนดคุณสมบัติพื้นฐานให้กับการเปลี่ยนแปลงแผนภาพคลาสของคลาส Transition ซึ่งรายละเอียดของคลาสแสดงในรูปที่ 5.5 โดยมีองค์ประกอบที่สำคัญดังนี้

5.6.1 ตัวแปรสมาชิก name

ตัวแปรสมาชิก name คือตัวแปรของชื่อการเปลี่ยนแปลง ซึ่งมีชนิดเป็น String ใช้สำหรับเก็บค่าของชื่อการเปลี่ยนแปลง

5.6.2 ตัวแปรสมาชิก transitionString

ตัวแปรสมาชิก transitionString คือตัวแปรของข้อความการเปลี่ยนแปลง ซึ่งมีชนิดเป็น String ใช้สำหรับเก็บข้อความการเปลี่ยนแปลง

5.6.3 ตัวแปรสมาชิก eventType

ตัวแปรสมาชิก eventType คือตัวแปรแสดงชนิดของเหตุการณ์ ซึ่งมีชนิดเป็น int ใช้สำหรับเก็บข้อมูลของชนิดเหตุการณ์ที่กระตุ้นให้เกิดการเปลี่ยนแปลง โดยกำหนดความหมายดังนี้ ถ้าค่าของ eventType เท่ากับ 0 หมายถึงไม่มีการระบุเหตุการณ์ ถ้าเท่ากับ 1 หมายถึงเป็นเหตุการณ์การเปลี่ยนค่าของตัวแปรพร้อมทำงาน ถ้าเท่ากับ 2 หมายถึงเป็นเหตุการณ์แบบเวลา ถ้าเท่ากับ 3 หมายถึงเป็นเหตุการณ์เมื่อเข้าสู่สถานะ ถ้าเท่ากับ 4 หมายถึงเป็นเหตุการณ์เมื่อออกจากสถานะ

5.6.4 สถานะตั้งต้นและสถานะเป้าหมาย

สถานะตั้งต้น คือตัวแปรสมาชิก source มีชนิดเป็น State ทำหน้าที่เป็นพอยท์เตอร์ชี้ตำแหน่งของสถานะตั้งตนของการเปลี่ยนแปลง สถานะเป้าหมาย คือตัวแปรสมาชิก target มีชนิดเป็น State ทำหน้าที่เป็นพอยท์เตอร์ชี้ตำแหน่งของสถานะเป้าหมายของการเปลี่ยนแปลง

5.6.5 ตัวแปรสมาชิก av

ตัวแปรสมาชิก av คือรายการของตัวแปรพร้อมทำงาน ซึ่งมีชนิดเป็น String ทำหน้าที่เก็บค่าของรายการของตัวแปรพร้อมทำงานซึ่งกำหนดในสัญลักษณ์ของเหตุการณ์การเปลี่ยนค่าของตัวแปรพร้อมทำงาน

5.6.6 ตัวแปรสมาชิก delay

ตัวแปรสมาชิก delay คือตัวแปรสำหรับเวลาหน่วง ซึ่งมีชนิดเป็น String ทำหน้าที่เก็บค่าเวลาหน่วงที่กำหนดในสัญลักษณ์ของเหตุการณ์ประเภทเวลา

5.6.7 ตัวแปรสมาชิก triggerT และตัวแปรสมาชิก removeT

ตัวแปรสมาชิก triggerT คือรายการของการเปลี่ยนแปลงเริ่มต้นเหตุการณ์แบบเวลาซึ่งมีชนิดเป็น String ทำหน้าที่เก็บรายการของการเปลี่ยนแปลงเริ่มต้นเหตุการณ์แบบเวลาที่กำหนดใน

สัญลักษณ์ของเหตุการณ์แบบเวลาที่ถูกเพิ่มเติม ตัวแปรสมาชิก removeT คือรายการของการเปลี่ยนแปลงยกเลิกเหตุการณ์แบบซึ่งมีชนิดเป็น String ทำหน้าที่เก็บรายการของการเปลี่ยนแปลงยกเลิกเหตุการณ์แบบเวลาที่กำหนดในสัญลักษณ์ของเหตุการณ์แบบเวลาที่ถูกเพิ่มเติม

5.6.8 ตัวแปรสมาชิก condition

ตัวแปรสมาชิก condition คือตัวแปรของเงื่อนไขซึ่งมีชนิดเป็น String ทำหน้าที่เก็บเงื่อนไขของการเปลี่ยนแปลงที่จะต้องถูกตรวจสอบเมื่อการเปลี่ยนแปลงถูกกระตุ้นให้เกิดขึ้น

5.6.9 ตัวแปรสมาชิก actions

ตัวแปรสมาชิก actions คือรายการของการกระทำซึ่งมีชนิดเป็น Vector ทำหน้าที่เก็บรายการของการกระทำและการกระทำในอนาคตที่ระบุในข้อความการเปลี่ยนแปลง

5.6.10 ฟังก์ชันสมาชิก setStoreObject และฟังก์ชันสมาชิก getStoreObject

ฟังก์ชันสมาชิก setStoreObject ทำหน้าที่นำข้อมูลของวัตถุ StoreObject มากำหนดคุณสมบัติให้กับวัตถุซึ่งมีชนิดเป็นคลาสย่อยของ Transition ฟังก์ชันสมาชิก getStoreObject ทำหน้าที่นำข้อมูลของวัตถุของคลาส Transition มากำหนดให้วัตถุ StoreObject แล้วส่งวัตถุ StoreObject นี้ให้กับวัตถุที่ร้องขอ

## 5.7 คลาส ExTransition

คลาส ExTransition คือคลาสของการเปลี่ยนแปลงภายนอก ซึ่งรายละเอียดของคลาสแสดงในรูปที่ 5.6 โดยมีองค์ประกอบที่สำคัญดังนี้

5.7.1 ตัวแปรสมาชิก arrowLine

ตัวแปรสมาชิก arrowLine คือเส้นหัวลูกศร มีชนิดเป็นคลาส GeneralPath ทำหน้าที่เป็นวัตถุซึ่งวาดเป็นเส้นหัวลูกศร

5.7.2 ตัวแปรสมาชิก coParent

เนื่องจากการเปลี่ยนแปลงภายนอกสามารถเปลี่ยนแปลงไปยังสถานะที่อยู่ต่างระดับของโครงสร้างลำดับชั้นได้เช่น การเปลี่ยนแปลงสู่สถานะย่อย เป็นต้น ดังนั้นจึงกำหนดตัวแปร coParent มีชนิดเป็น Container ให้เป็นตัวบรรจุร่วมของสถานะตั้งต้นและสถานะเป้าหมายที่อยู่ต่างระดับกัน แล้วทำการบรรจุการเปลี่ยนแปลงภายนอกสู่ตัวบรรจุร่วมนี้

5.7.3 ฟังก์ชันสมาชิก findCoParent

ฟังก์ชันสมาชิก findCoParent ทำหน้าที่หาตัวบรรจุร่วมของสถานะตั้งต้นและสถานะเป้าหมายที่กำหนด

5.7.4 ฟังก์ชันสมาชิก removeSelf

ฟังก์ชันสมาชิก removeSelf ทำหน้าที่ลบตนเองออกจากสถานะซึ่งเป็นตัวบรรจุ ลบตนเองออกจากการอ้างอิงของรายการการเปลี่ยนแปลงที่เข้าสู่สถานะ และการอ้างอิงรายการการเปลี่ยนแปลงที่ออกจากสถานะ

5.7.5 ฟังก์ชันสมาชิก setStoreObject และฟังก์ชันสมาชิก getStoreObject

ฟังก์ชันสมาชิก setStoreObject และฟังก์ชันสมาชิก getStoreObject เป็นการเขียนทับ (Override) ฟังก์ชันสมาชิก setStoreObject และฟังก์ชันสมาชิก getStoreObject ของคลาส Transition เพื่อเพิ่มเติมรายละเอียดสำหรับการเปลี่ยนแปลงภายนอก

```
public abstract class Transition extends ExLabel {
.....
String name;
String transitionString;
int eventType;
State source;
State target;
String av;
String delay;
String triggerT,removeT;
String condition;
String actions;
.....
public StoreObject getStoreObject(){
   this.storeObject.setBounds(this.getBounds());
   this.storeObject.name=this.name;
   this.storeObject.transitionString=this.transitionString;
   this.storeObject.eventType=this.eventType;
   this.storeObject.av=this.av;
   this.storeObject.delay=this.delay;
   this.storeObject.triggerT=this.triggerT;
   this.storeObject.removeT=this.removeT;
   this.storeObject.condition=this.condition;
   this.storeObject.actions=this.actions;
   return this.storeObject;
  }

  public void setStoreObject(StoreObject sto){
   this.setBounds(sto.getBounds());
   this.name=sto.name;
   this setTransitionLabel(sto.transitionString);
   this.eventType=sto.eventType;
   this.av=sto.av;
   this.delay=sto.delay;
   this.triggerT=sto.triggerT;
   this.removeT=sto.removeT;
   this.condition=sto.condition;
   this.actions=sto.actions;

  }
.....
}
```

รูปที่ 5.5 รายละเอียดของคลาส Transition

```
public class ExTransition extends Transition {

GeneralPath arrowLine;
Container coParent;
.....
public ExTransition(State s,Point sp,State t,Point tp,RootPane root) {
   this.rootPane=root;
   source =s;
   target =t;
   source.exitT.add(this);
   target.entryT.add(this);
   this.sp=sp;
   this.tp=tp;
   this.eventType=0;
   StoreObject stoSource=source.getStoreObject();
   stoSource.exitT.add(this.storeObject);
   StoreObject stoTarget=target.getStoreObject();
   stoTarget.entryT.add(this.storeObject);
.....
}

public StoreObject getStoreObject(){
   this.storeObject = super.getStoreObject();
   this.storeObject.objectType="ExTransition";
   this.storeObject.sourcePoint=this.sp;
   this.storeObject.targetPoint=this.tp;
   this.storeObject.sourceState=this.source.getStoreObject();
   this.storeObject.targetState=this.target.getStoreObject();
   return this.storeObject;
  }

  public void setStoreObject(StoreObject sto){
   super.setStoreObject(sto);
   this.sp=sto.sourcePoint;
   this.tp=sto.targetPoint;
   this.storeObject=sto;
  }

private void findCoParent(JPanel s1,JPanel s2){

   if(s1.getParent().isAncestorOf(s2)|| s1.getParent().equals(s2)){
      if(s1.getParent().equals(s2)){
      coParent= (JPanel)s2.getParent();
      }else{
      coParent= (JPanel)s1.getParent();;
      }
   }else {
      findCoParent((JPanel)s1.getParent(),s2);
   }
  }
```

รูปที่ 5.6 รายละเอียดของคลาส ExTransition

```
public void removeSelf(){// remove itself from its parent
   this.source.exitT.remove(this);
   this.target.entryT.remove(this);
   Container parent=this.getParent();
   parent.remove(this);
   parent.repaint();
   if(parent instanceof RootPane){
    RootPane temp=(RootPane)parent;
    StoreObject stoParent=temp.getStoreObject();
    stoParent.remove(this.getStoreObject());
   }else{
    BasicState temp=(BasicState)parent;
    StoreObject stoParent=temp.getStoreObject();
    stoParent.remove(this.getStoreObject());
   }
  }
.....
}
```

รูปที่ 5.6 รายละเอียดของคลาส ExTransition (ต่อ)

### 5.8 คลาส InTransition

คลาส InTransition คือคลาสของการเปลี่ยนแปลงภายใน ซึ่งรายละเอียดของคลาสแสดงในรูปที่ 5.7 โดยมีองค์ประกอบที่สำคัญดังนี้

5.8.1 คอนสตักท์เตอร์

คอนสตักท์เตอร์ของคลาส InTransition ทำหน้าที่กำหนดค่าเริ่มต้นให้กับวัตถุโดยระบุวัตถุที่สร้างเป็นการเปลี่ยนแปลงภายในนี้ให้กับรายการการเปลี่ยนแปลงภายในของสถานะตั้งต้น

5.8.2 ฟังก์ชันสมาชิก removeSelf

ฟังก์ชันสมาชิก removeSelf ทำหน้าที่ลบตนเองออกจากสถานะตั้งต้น

5.8.3 ฟังก์ชันสมาชิก setStoreObject และฟังก์ชันสมาชิก getStoreObject

ฟังก์ชันสมาชิก setStoreObject ของคลาส InTransition ได้เขียนทับฟังก์ชัน setStoreObject ของคลาส Transition เพื่อเพิ่มเติม และฟังก์ชันสมาชิก getStoreObject ได้เขียนทับฟังก์ชัน getStoreObject ของคลาส Transition เพื่อเพิ่มเติมเช่นกัน

```
public class InTransition extends Transition {
.....
public InTransition(State s) {
   this.source = s;
   source.add(this,0);
   source.internalT.add(this);
   this.setBounds(20,10,50,25);
  }

  public StoreObject getStoreObject(){
   this.storeObject = super.getStoreObject();
   this.storeObject.objectType="InTransition";
   this.storeObject.sourceState=this.source.getStoreObject();
   return this.storeObject;
  }
  public void setStoreObject(StoreObject sto){
   super.setStoreObject(sto);
   this.storeObject=sto;
  }

  public void removeSelf(){// remove itself from its parent
   this.source.internalT.remove(this);
   Container parent=this.getParent();
   parent.remove(this);
   parent.repaint();
   if(parent instanceof RootPane){
    RootPane temp=(RootPane)parent;
    StoreObject stoParent=temp.getStoreObject();
    stoParent.remove(this.getStoreObject());
   }else{
    BasicState temp=(BasicState)parent;
    StoreObject stoParent=temp.getStoreObject();
    stoParent.remove(this.getStoreObject());
   }
  }.....
}
```

รูปที่ 5.7 รายละเอียดของคลาส InTransition

### 5.9 คลาส StoreObject

คลาส StoreObject คือคลาสที่ใช้สร้างวัตถุสำหรับใช้เก็บข้อมูลของทุกองค์ประกอบในแผนภาพสเตทชาร์ท ตัวอย่างเช่น สถานะพื้นฐาน สถานะครั้งก่อน การเปลี่ยนแปลงภายนอก การเปลี่ยนแปลงภายใน เป็นต้น องค์ประกอบของคลาส StoreObject ประกอบด้วยตัวแปรสมาชิกชนิดต่างๆ ที่ทำหน้าที่เก็บข้อมูลที่แตกต่างกันในแต่ละประเภท เมื่อผู้ใช้ทำการบันทึกแผนภาพสเตทชาร์ทเป็นแฟ้มข้อมูล วัตถุชนิด StoreObject นี้จะถูกจัดเก็บในรูปของแฟ้มข้อมูลด้วยวิธีการซีเรียลไลเซชัน (Serialization) เมื่อทำการเรียกใช้แฟ้มข้อมูล วัตถุชนิด StoreObject ถูกเรียกกลับมาในหน่วยความจำและถูกแปลงเป็นวัตถุที่เป็นองค์ประกอบของแผนภาพสเตทชาร์ทดังเดิม รายละเอียดของคลาส StoreObject แสดงในรูปที่ 5.8

```
public class StoreObject extends Container {
        public String objectType;
        public String name="";
        public Color color;
        public int historyType;
        public Point sourcePoint;
        public Point targetPoint;
        public StoreObject sourceState;
        public StoreObject targetState;
        public String transitionString;
        public int eventType;
        public String av;
        public String delay;
        public String triggerT;
        public String removeT;
        public String condition;
        public Vector actions;
        public Vector methodName;
        public Vector methodDetail;
        public Vector entryT=new Vector();
        public Vector exitT=new Vector();
        public String functionName;

        public StoreObject() {

        }
}
```

รูปที่ 5.8 รายละเอียดของคลาส StoreObject

### 5.10 คลาส StateDialog

คลาส StateDialog คือคลาสของหน้าจอแสดงคุณสมบัติของสถานะ (State properties) โดยคุณสมบัติที่แสดงคือ ชื่อสถานะ การเปลี่ยนแปลงภายใน และสถานะย่อย วัตถุชนิด StateDialog ถูกสร้างโดยการคลิกที่วัตถุที่มีชนิดเป็นคลาส BasicState ดังรายละเอียดที่แสดงในรูปที่ 5.9 หน้าจอแสดงคุณสมบัติของสถานะสามารถแสดงรายละเอียดได้โดยฟังก์ชัน setStateInfo ซึ่งทำการอ่านคุณสมบัติของสถานะแล้วแสดงบนหน้าจอ

### 5.11 คลาส TransitionDialog

คลาส TransitionDialog คือคลาสของหน้าจอแสดงคุณสมบัติของการเปลี่ยนแปลง (Transition properties) โดยคุณสมบัติที่แสดงคือ ชื่อการเปลี่ยนแปลง ชนิดของเหตุการณ์ที่กระตุ้น เงื่อนไข และรายการของการกระทำ ดังรายละเอียดที่แสดงในรูปที่ 5.10 หน้าจอแสดงคุณสมบัติของการเปลี่ยนแปลงสามารถแสดงรายละเอียดได้โดยฟังก์ชัน setTransitionInfo ซึ่งทำการอ่านคุณสมบัติของการเปลี่ยนแปลงแล้วแสดงบนหน้าจอ

```
public class BasicState extends State {
.....
void this_mouseClicked(MouseEvent e) {
    if(rootPane.parent.cursorType.equals("select")){
     if(e.getClickCount()==2){
          new StateDialog(this);
      }
    }
.....
}

public class StateDialog extends JDialog {
.....
public StateDialog(BasicState state) {
   try {
    this.currentState=state;
    jbInit();
    this.setStateInfo(this.currentState);
    this.setCenter();
   }
   catch(Exception e) {
    e.printStackTrace();
   }
  }
public void setStateInfo(BasicState state){
   this.stateName.setText(state.name);// set name
   // display history state type
   if(state.getHistoryState()==1){
    this.hisState.setSelected(true);
   }else if(state.getHistoryState()==2){
    this.deepHisState.setSelected(true);
   }
   // display internal transition
   this.internalT=state.getInternalTransition();
   this.inTransString.removeAllElements();
   for(int i=0;i<internalT.size();i++){
    InTransition tmp =(InTransition)internalT.elementAt(i);
    this.inTransString.add(tmp.transitionString);
   }
   this.inTransitionList.setListData(this.inTransString);
   // display substate
   this.subState=state.getSubstate();
   this.substateName.removeAllElements();
   for(int i=0;i<this.subState.size();i++){
    BasicState tmp =(BasicState)this.subState.elementAt(i);
    this.substateName.add(tmp.name);
   }
   this.subStateList.setListData(this.substateName);
  }.....
}
```

รูปที่ 5.9 รายละเอียดของฟังก์ชันของคลาส StateDialog

```
public class TransitionDialog extends JDialog {
.....
public TransitionDialog(Transition t) {
   try {
    this.currentTransition=t;
    this.parentFrame=t.source.rootPane.parent;
    jbInit();
    if(t instanceof ExTransition){
     this.entryState.setVisible(false);
     this.exitState.setVisible(false);
    }else if(t instanceof InTransition){
     this.noEvent.setEnabled(false);
    }
    this.setTransitionInfo(this.currentTransition);
    this.setCenter();
   }
   catch(Exception e) {
    e.printStackTrace();
   }
 }
public void setTransitionInfo(Transition t){
   this.transitionName.setText(t.name);
   this.eventType=t.eventType;
   // set event type
   if(this.eventType==0){
      this.noEvent.setSelected(true);
   }else if(this.eventType==1){
      this.changeOf.setSelected(true);
      this.av.setText(t.av);
   }else if(this.eventType==2){
      this.timeEvent.setSelected(true);
      this.delay.setText(t.delay);
      this.triggerTrans.setText(t.triggerT);
      this.removeTrans.setText(t.removeT);
   }else if(this.eventType==3){
      this.entryState.setSelected(true);
   }else if(this.eventType==4){
      this.exitState.setSelected(true);
   }
   // set condition
   this.condition.setText(t.condition);
   // set actions into action vector
   this.actionVector.removeAllElements();
   if(t.actions!=null){
    for(int i=0;i<t.actions.size();i++){
      this.actionVector.add(t.actions.elementAt(i));
    }
      this.addActionToList(this.actionVector);
   }
 }
.....
}
```

รูปที่ 5.10 รายละเอียดของฟังก์ชันของคลาส TransitionDialog

**5.12 คลาส CodeGenerator**

คลาส CodeGenerator คือคลาสของตัวสร้างชุดคำสั่งของวัตถุพร้อมทำงานซึ่งถูกใช้โดยบรรณาธิกรสำหรับสร้างเค้าร่าง ตัวสร้างชุดคำสั่งเดิมสามารถสร้างชุดคำสั่งได้เพียงโครงของชุดคำสั่งเท่านั้นซึ่งไม่มีส่วนรายละเอียดของฟังก์ชัน ดังนั้นจึงได้ทำการพัฒนาเพิ่มเติมให้ตัวสร้างชุดคำสั่งมีความสามารถในการแปลงจากแผนภาพสเตทชาร์ทเป็นชุดคำสั่งของวัตถุพร้อมทำงานได้ โดยฟังก์ชันสมาชิกที่สำคัญของคลาส CodeGenerator มีดังต่อไปนี้

5.12.1 ฟังก์ชันสมาชิก writeModelFile

ฟังก์ชันสมาชิก writeModelFile ทำหน้าที่สร้างแฟ้มข้อมูลซึ่งเป็นชุดคำสั่งในส่วนคลาสโมเดลของวัตถุพร้อมทำงาน ชุดคำสั่งในส่วนคลาสโมเดลนี้เป็นชุดคำสั่งที่แปลงจากแผนภาพสเตทชาร์ทโดยตรงดังนั้นชุดคำสั่งในส่วนคลาสโมเดลจึงเป็นส่วนกำหนดพฤติกรรมของวัตถุพร้อมทำงาน

5.12.2 ฟังก์ชันสมาชิก writeEditFile

ฟังก์ชันสมาชิก writeEditFile ทำหน้าที่สร้างแฟ้มข้อมูลซึ่งเป็นชุดคำสั่งในส่วนคลาสวิวของวัตถุพร้อมทำงาน ชุดคำสั่งในส่วนคลาสวิวนี้เป็นชุดคำสั่งที่ใช้แสดงผลการทำงานบนหน้าจอ สำหรับฟังก์ชันสมาชิกของคลาสวิวสามารถกำหนดรายละเอียดได้ด้วยบรรณาธิกรสำหรับสร้างแผนภาพสเตทชาร์ท