UML全程实作

状态图

Think 

核心工作流

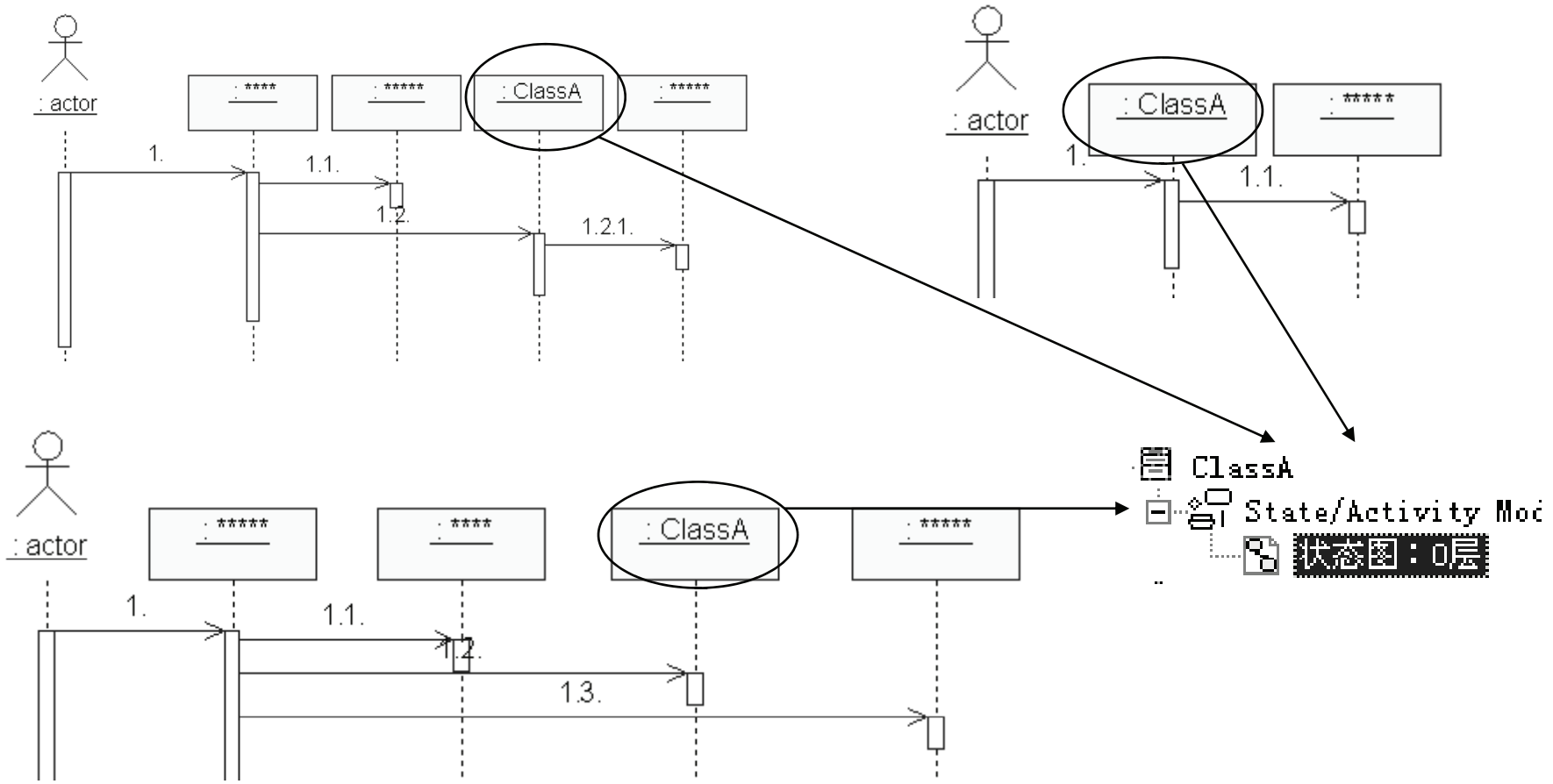
*愿景
*业务建模
    选定愿景要改进的业务组织
    业务用例图
    现状业务序列图
    改进业务序列图
*需求
    系统用例图
    书写用例文档

提升销售

*分析
    类图
    序列图
    状态图
*设计
    建立数据层
    精化业务层
    精化表示层

降低成本

状态图



把对象从所有的序列图中单独拿出来考察

对象能运转自如吗？

她没有说：不要去…

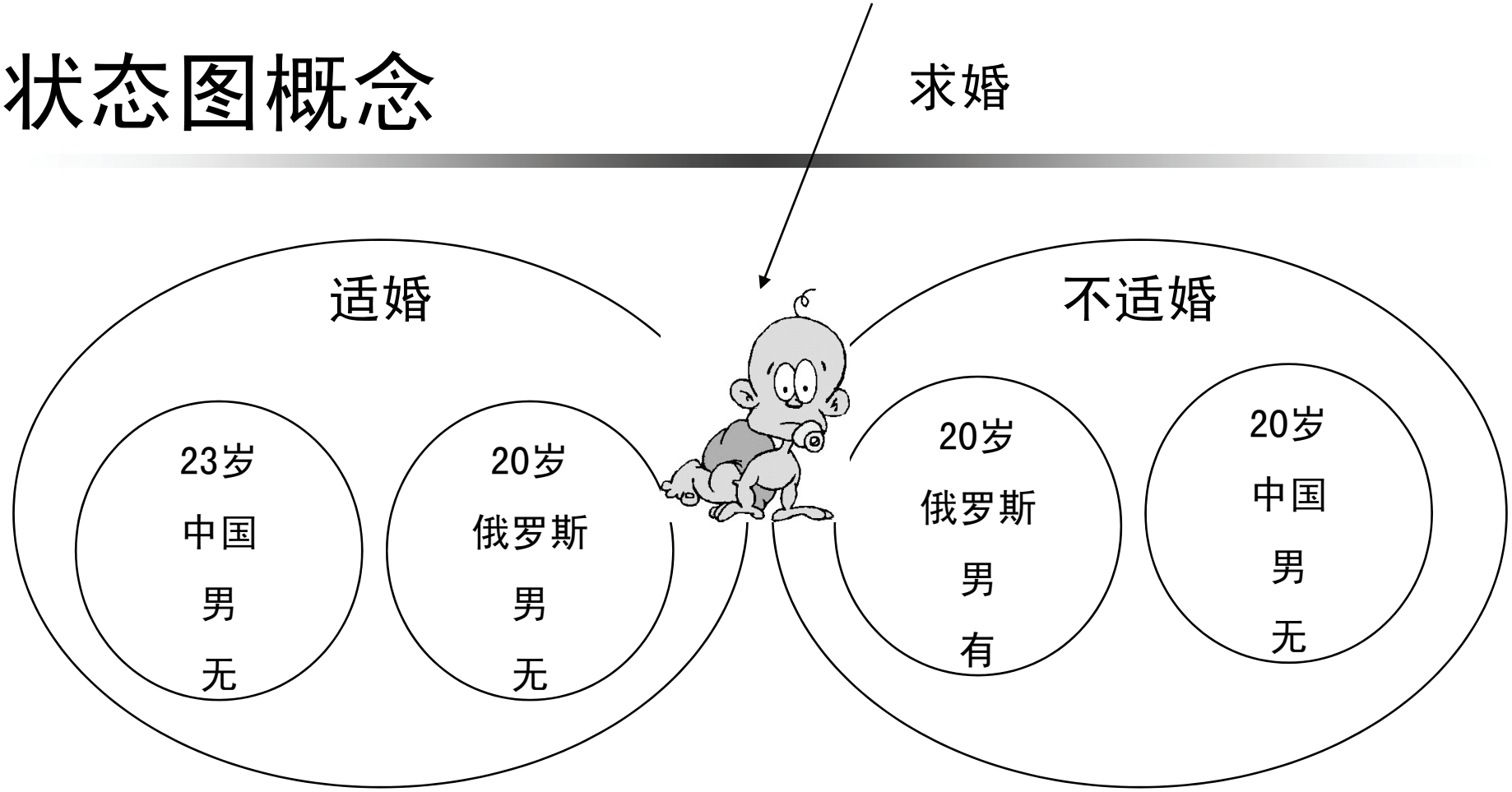
一名开发人员的一天

状态图概念

- 年龄：…6, 7, 8, 9…17, 18…21, 22…34, 35…
- 国籍：…中国，美国，俄罗斯，阿联酋…
- 性别：…男，女…
- 配偶：…有，无，多个…

“人”应该有几种状态？

状态图概念

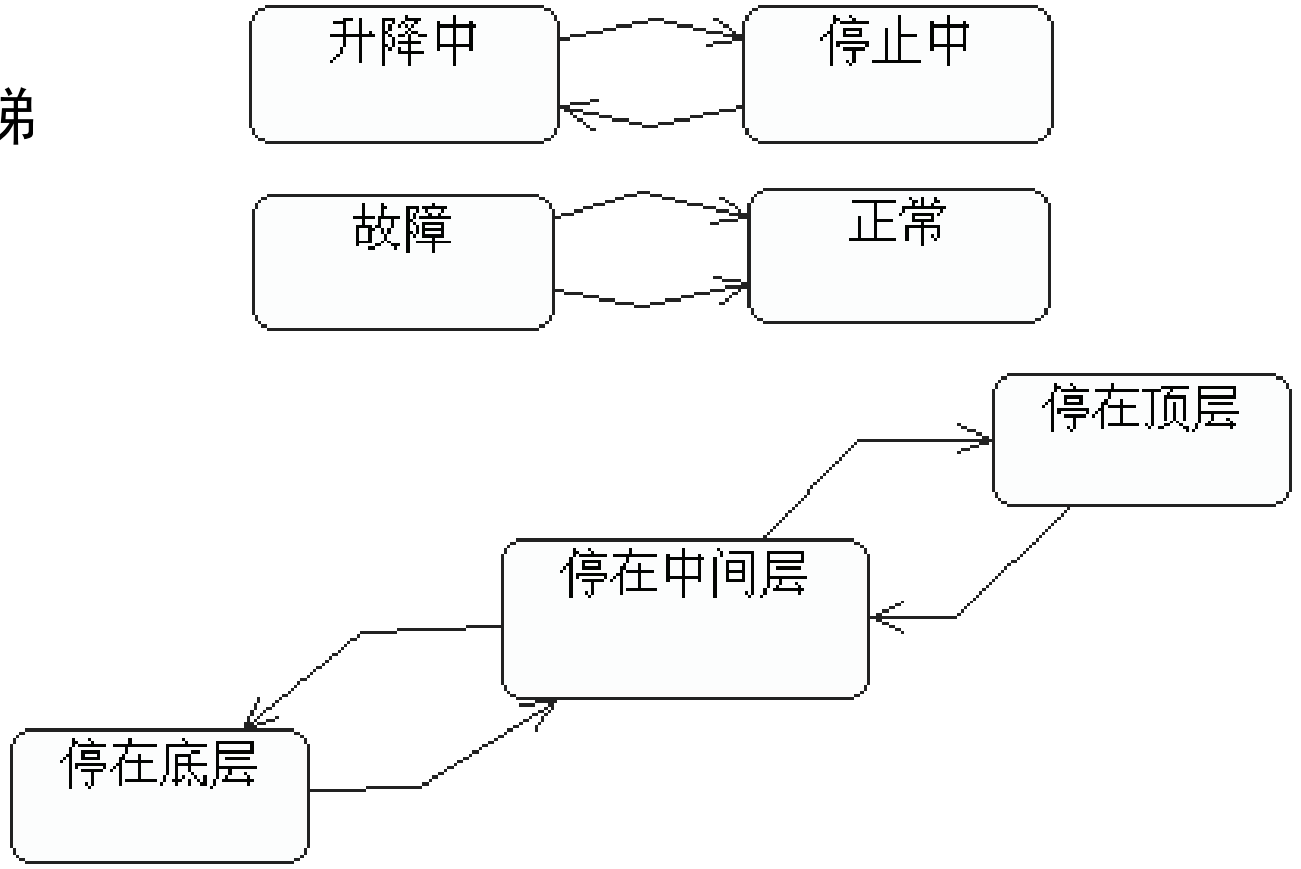


状态——在系统中表现出相同行为的属性值组合

状态！＝状态位

状态图概念

电梯

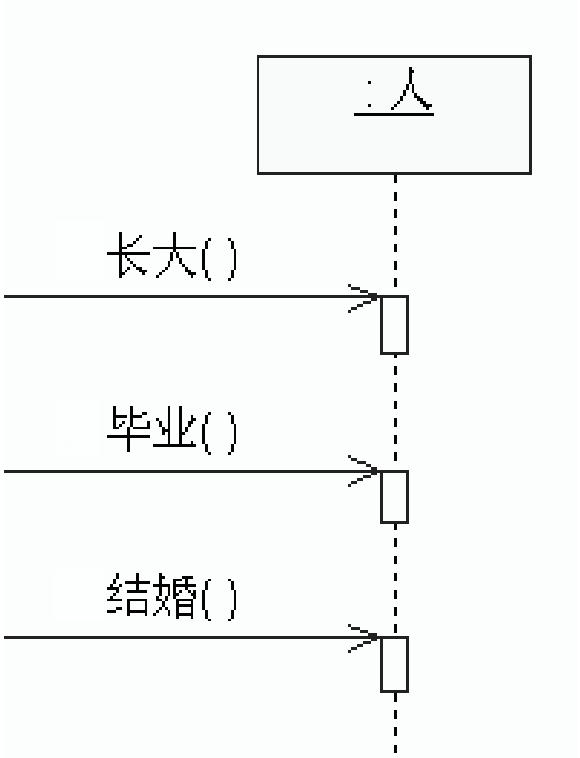


状态只是一种分组

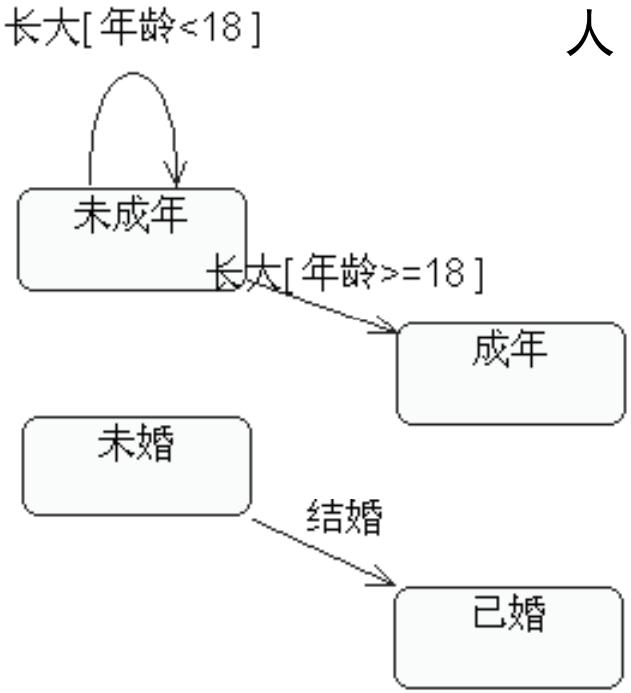
状态图概念

类图

序列图



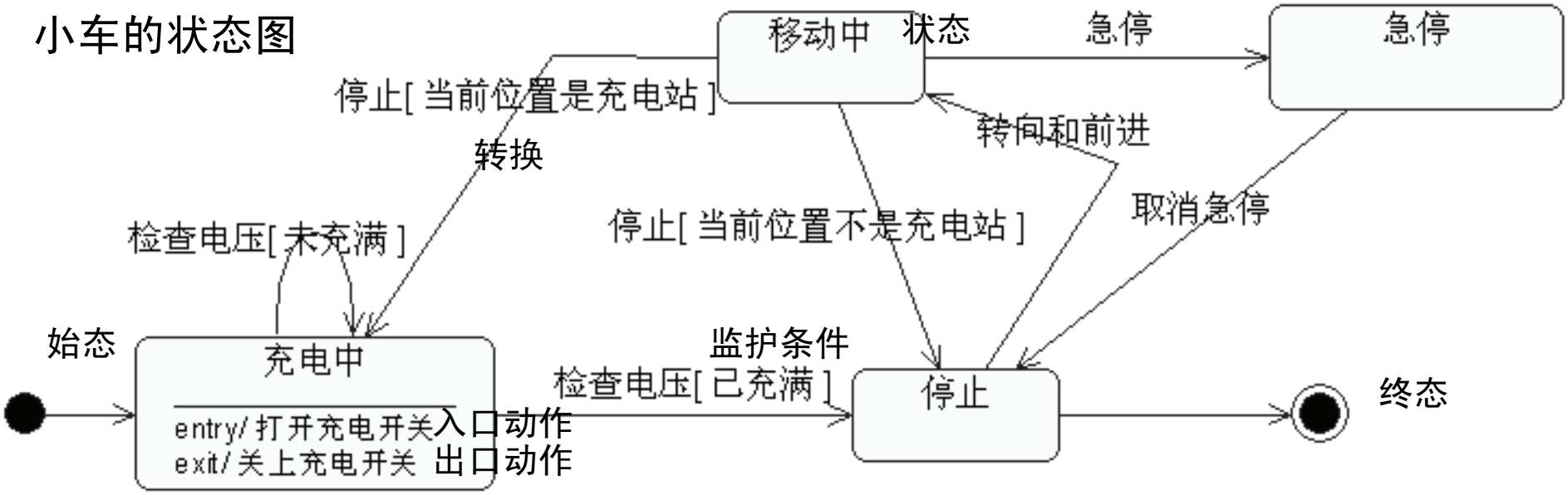
状态图



属性值变化导致行为发生变化—转换

状态图概念

小车的状态图

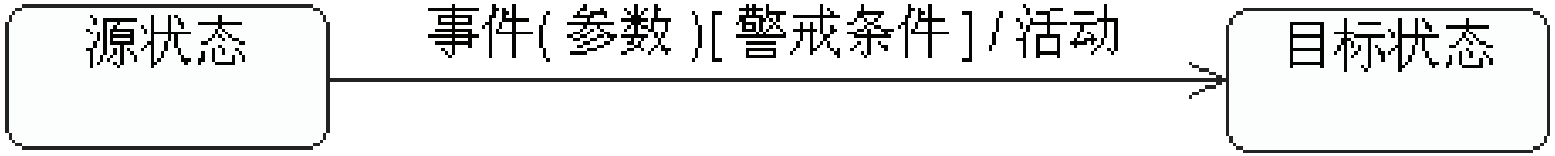


小车有几个操作？

状态图概念

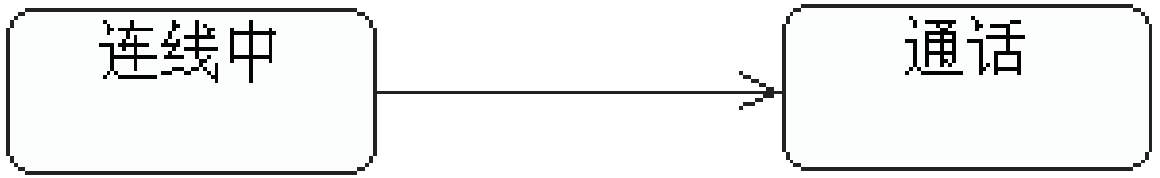
在源状态下，当事件发生时，

如果符合警戒条件，则执行活动，进入目标状态



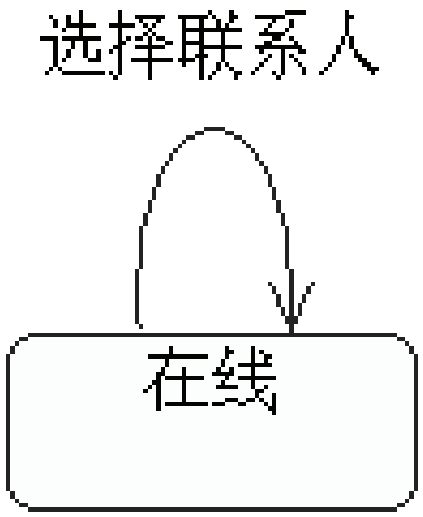
状态和转换

状态图概念



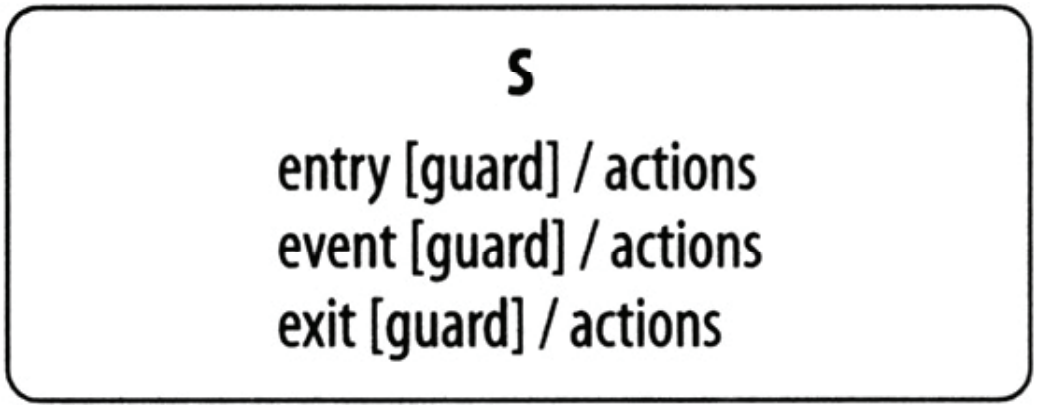
自动转换

状态图概念



自我转换

状态图概念



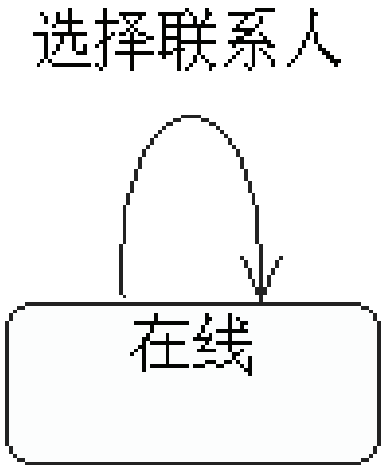
entry：进入时必须执行

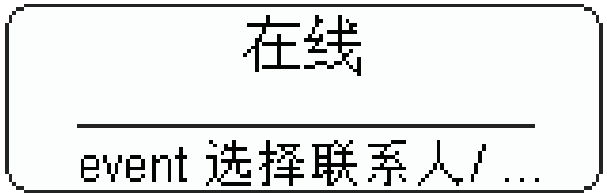
exit：离开时必须执行

event：发生event时内部执行

动作

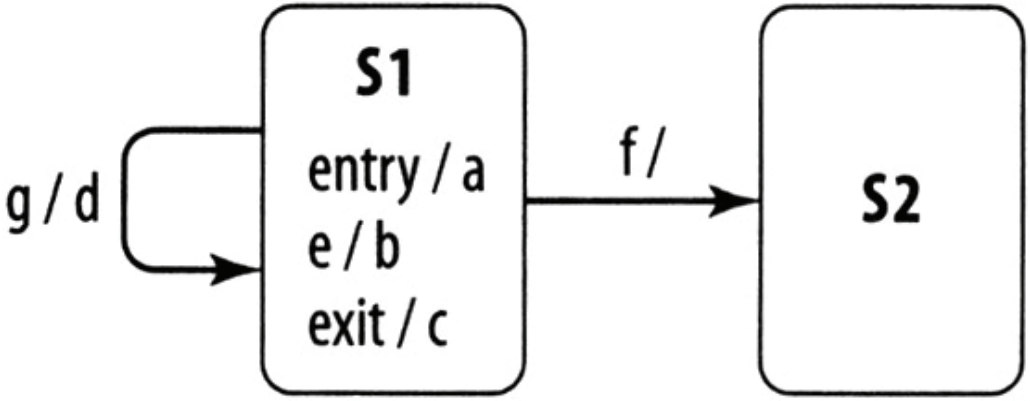
状态图概念





此二者有何区别？

状态图概念



当前S1：

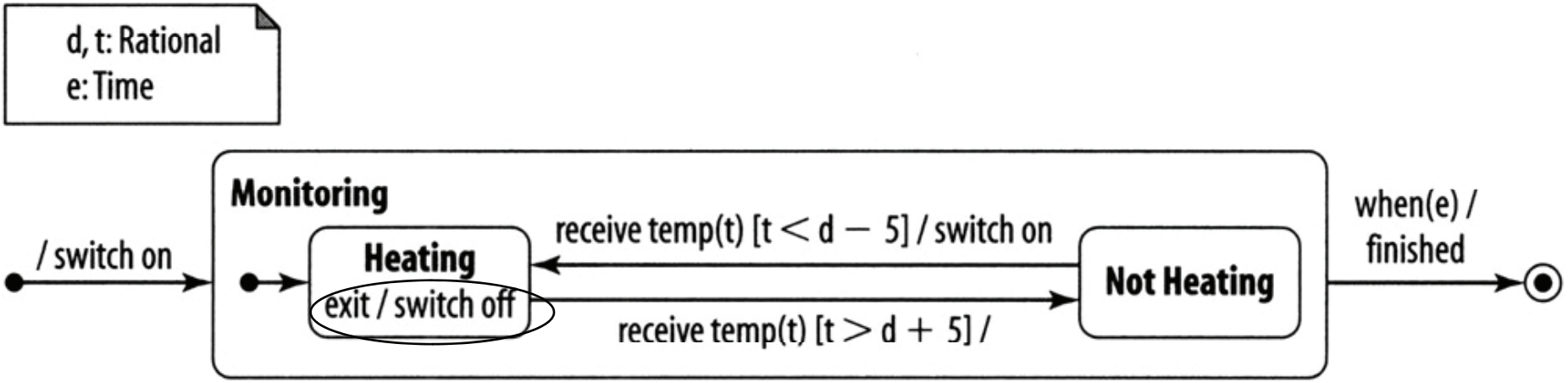
e发生：执行b，到S1

g发生：执行c, d, a，到S1

f发生：执行c，到S2

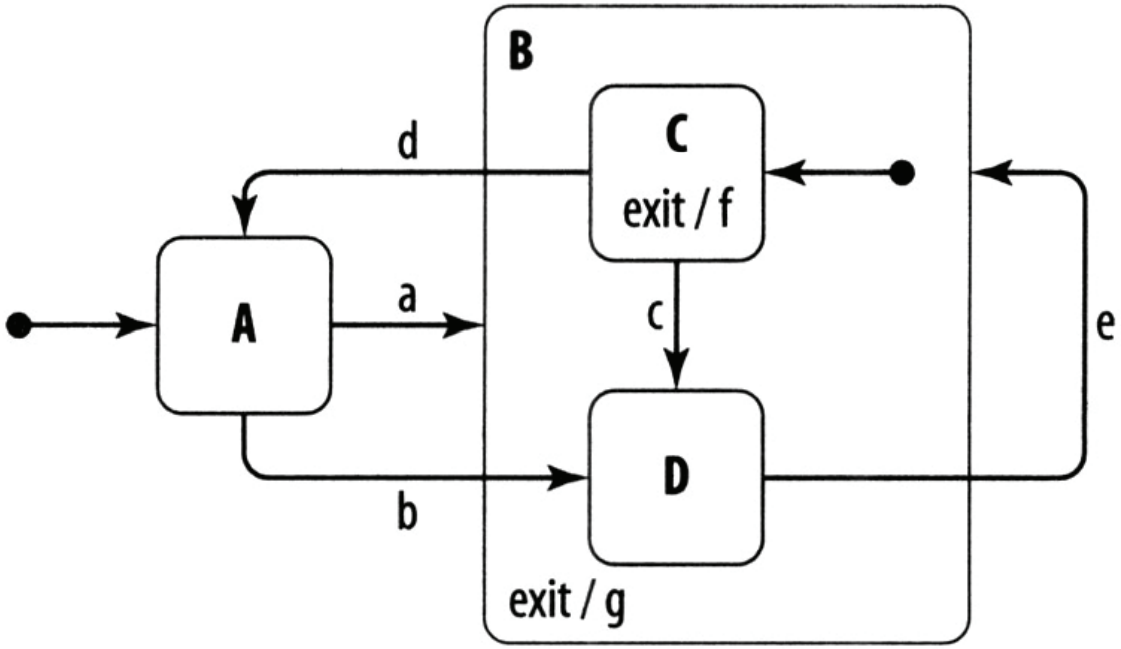
动作

状态图概念


d, t: Rational
e: Time
Monitoring
/ switch on
Heating
exit / switch off
receive temp(t) [t < d − 5] / switch on
receive temp(t) [t > d + 5] /
Not Heating
when(e) /
finished

嵌套状态

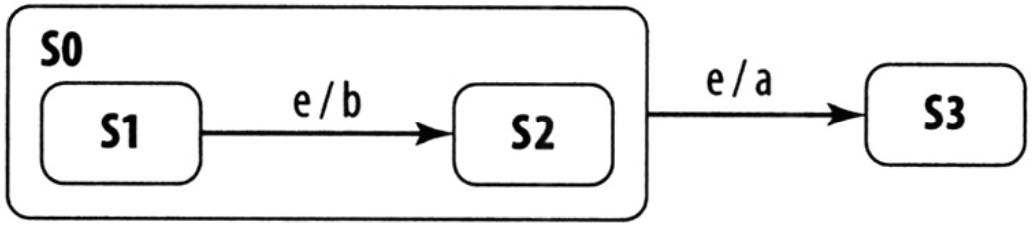
状态图概念



源状态	发生	动作	到达状态
A	a		B, C
B, C	c	f	B, D
B, D	e	g	B, C
B, C	d	f, g	A

嵌套状态

状态图概念



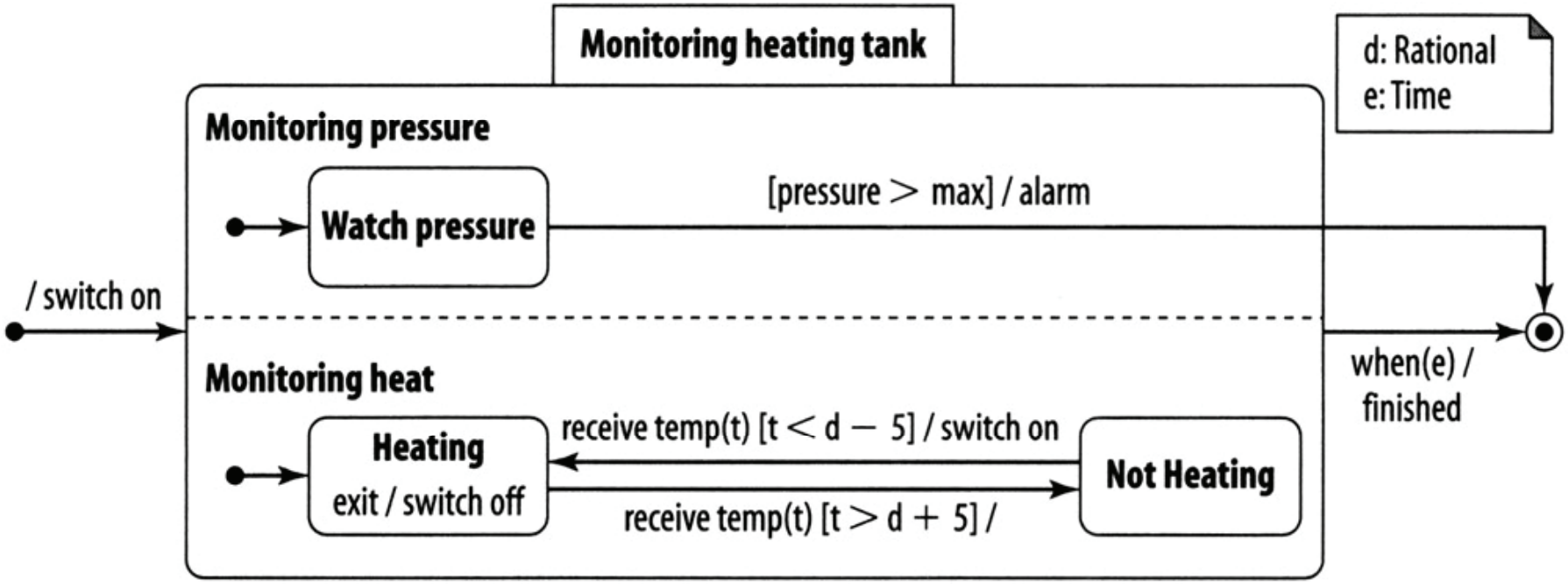
当前S1：

e发生会怎样？

子状态覆盖父状态，到S2

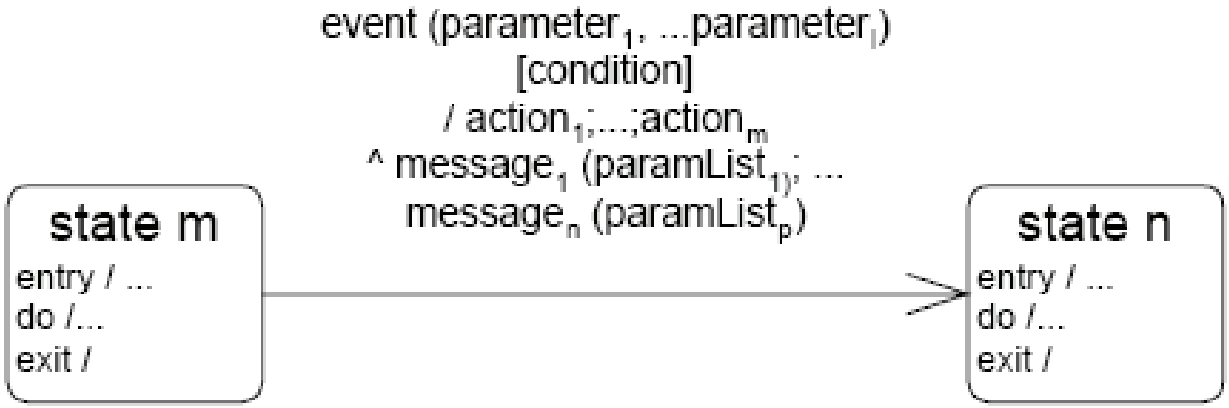
嵌套状态

状态图概念



并行状态—AND

状态图



1. 事件
2. 检查当前状态是否能接受事件
3. 如果可以，检查转换条件
4. 如果条件为真

   状态m的出口活动

   action1 – actionm

   发送message1 – messagen

   改变状态

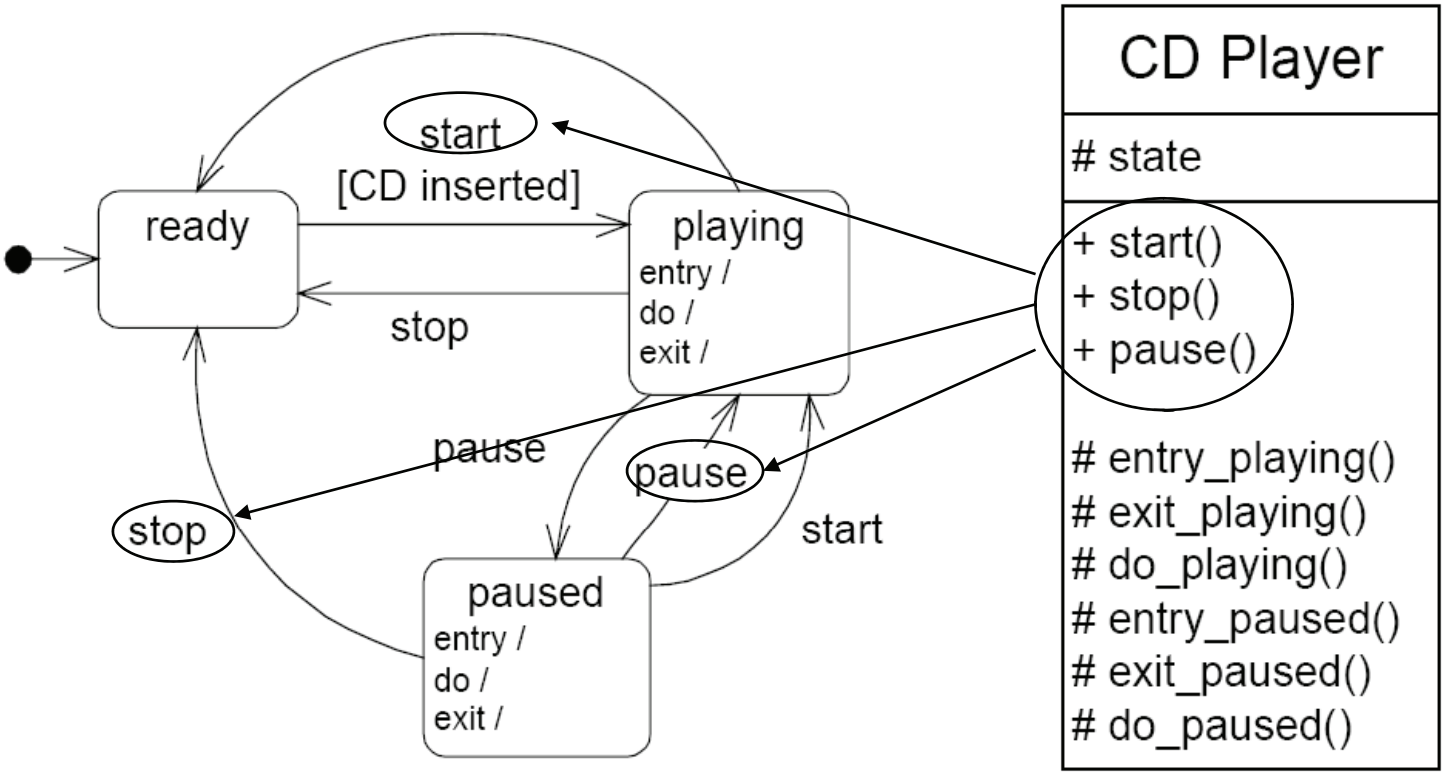
   状态n的入口活动

   状态n的do活动

代码映射

状态图

——代码映射



以CD机为例

状态图

——代码映射

```
class CD_Player {
protected:
enum {INITIAL, READY, PLAYING, PAUSED, FINAL}    stateValue;
stateValue  m_state;
virtual void setState (stateValue newValue)
{
     if ((newValue >= INITIAL) && (newValue <= FINAL))
     {
     m_state = newValue
     }
     else
     {
     throw badStateChange;    // exception
     }
};
```

状态和设置状态

状态图

——代码映射

```
virtual void stop (void)
{
      switch (m_state)
      {
            case PLAYING:
                  exit_playing();
                  setState (READY);
                  break;

            case PAUSED:
                  exit_paused();
                  setState (READY);
                  break;

            default:    break;  // event ignored

      }  // switch
};
```

Stop

状态图

```
virtual void start (void)
{
      switch (m_state)
      {
            case READY:
                  if (CD inserted)
                  {
                        exit_ready();
                        setState (PLAYING);
                        entry_playing();
                        do_playing();
                  }
            break;

            case PAUSED:
                  exit_paused();
                  setState (PLAYING);
                  entry_playing();
                  do_playing();

            default:    break;  // event ignored

      }  // switch
};
```

Start

状态图

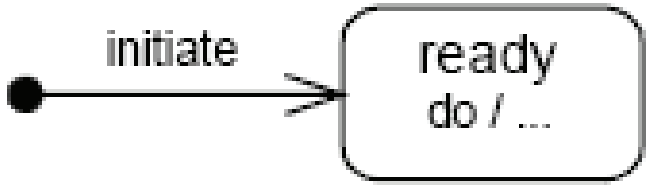
```
virtual void pause (void)
{
      switch (m_state)
      {
            case PLAYING:
                  exit_playing();
                  setState (PAUSED);
                  entry_paused();
                  do_paused();
            break;

            case PAUSED:
                  exit_paused();
                  setState (PLAYING);
                  entry_playing();
                  do_playing();
            break;

            default:    break;  // event ignored

      }  // switch
};
```

Pause

状态图

——代码映射

```
CDPlayer::CDPlayer (void)     // constructor
{
      setState (INITIAL);
      // user code
};


void CDPlayer::initiate (void)
{
      switch (m_state)
      {
            case INITIAL:
                  setState (READY);
                  do_ready();
                  break;

            default:    break; // event ignored
      }
};
```

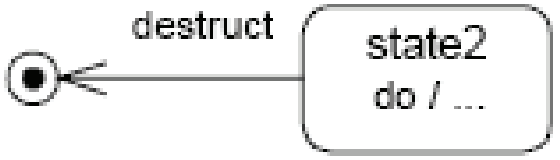


初始

状态图

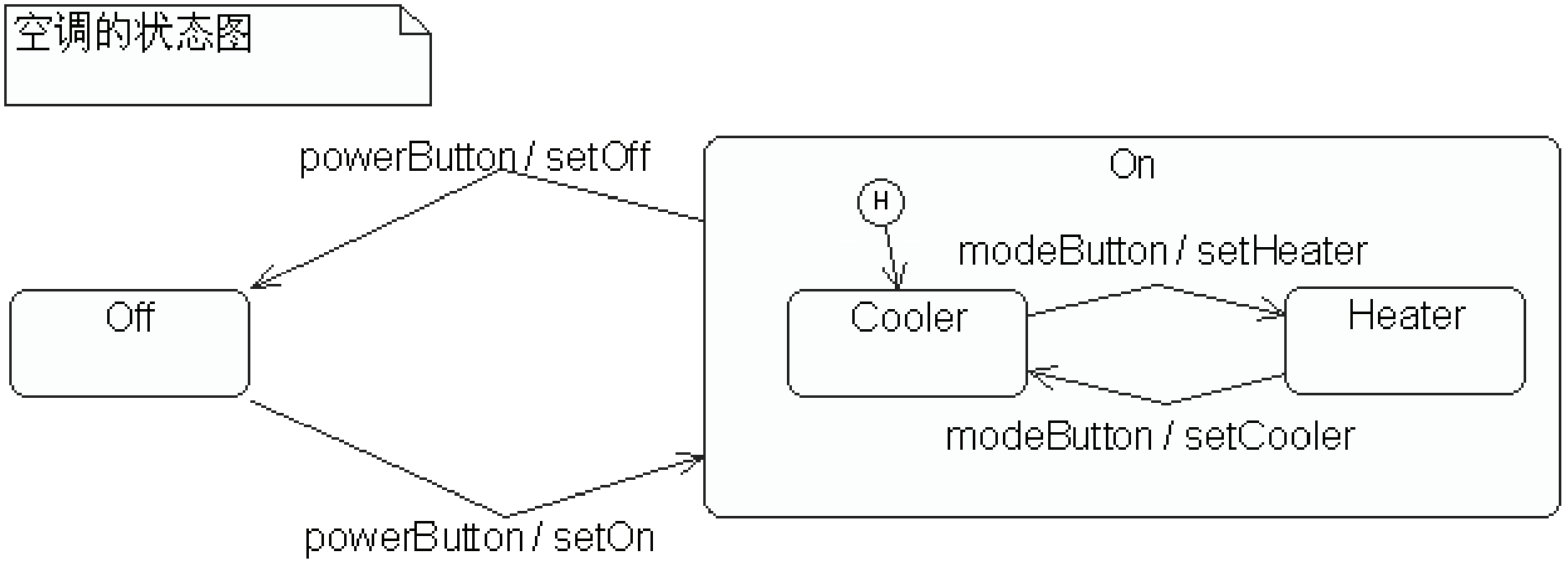
——代码映射

```
void CDPlayer::destruct (void)
{
      switch (m_state)
      {
            case STATE2:
                  setState (FINAL);
                  delete this;
                  break;

            default:    break; // event ignored
      }
};
```



终止

请写出powerButton()的代码



请写出powerButton()的代码

```
class AirCon {
    public static final int off = 1;
    public static final int on = 2;
    public static final int cooler = 3;
    public static final int heater = 4;
    public int state; // state variable
    public int on_subState;

    ……
```

```
public void powerButton() {
   switch (__①__ ) {
      case __②__ :
         setOn;
         __③__ = on_subState;
        break;
      case __④__ :
         setOff;
         state = __⑤__ ;
         break;

     ……..
   }
}
```

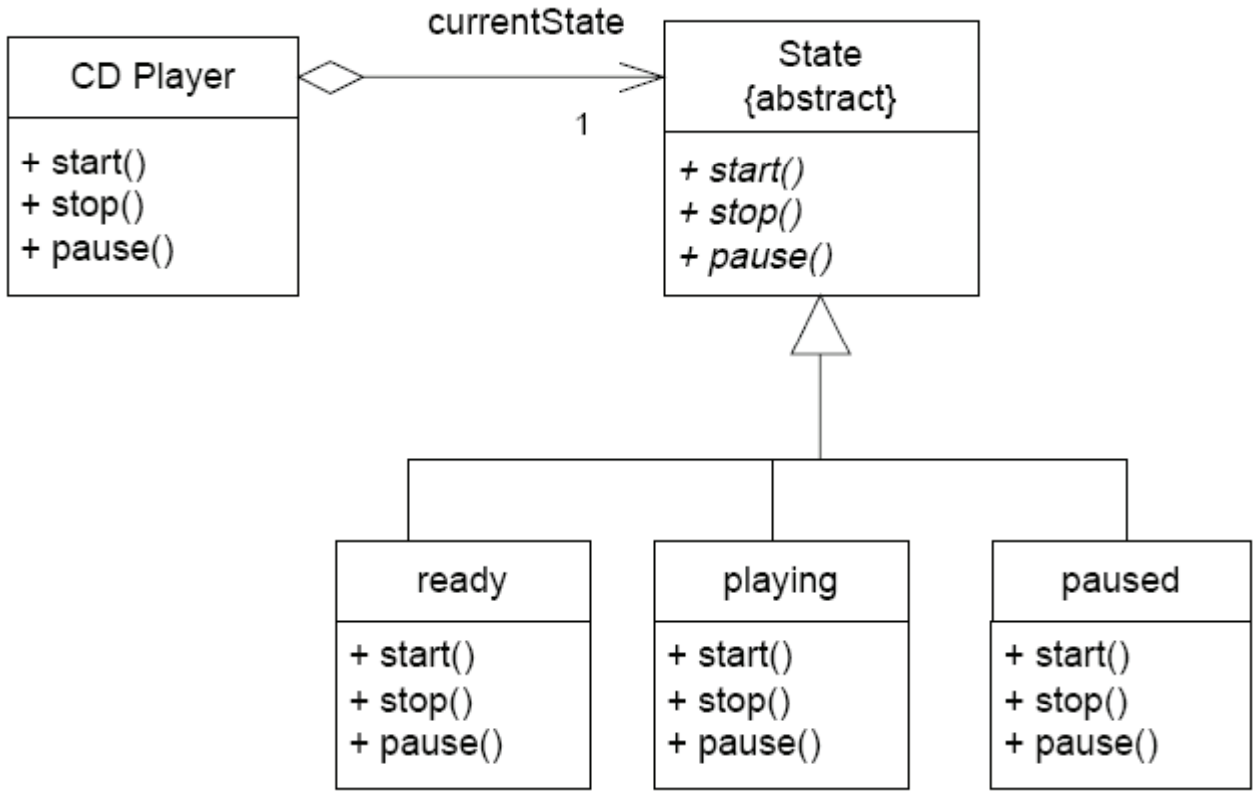
modeButton()的代码？

```
AirCon() {  //constructor
 state = off;
 on_subState = cooler;
}
public void modeBut() {  // event method
  switch (state) {
        case off :
              break;
        case cooler :
              setHeater;    // action
              // exit actions
              on_subState = Heater;
              state = on_subState;
              // entry actions
              break;
        case heater :
              setCooler;   // action
              // exit actions
              on_subState = Cooler;
              state = on_subState;
              // entry actions
              break;
        default :
              break;
}
```

初始状态

状态图

——代码映射



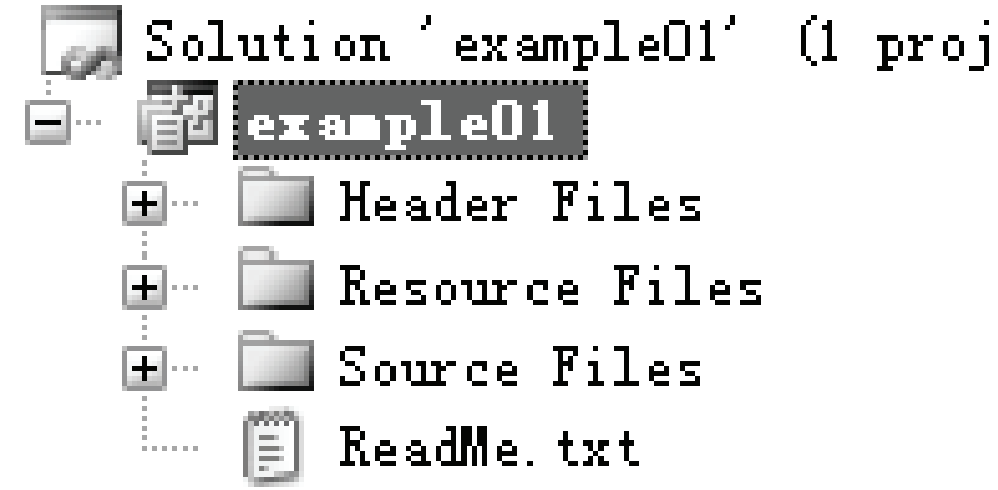
GoF的State模式

进一步重构

VS＋Statewizard演示

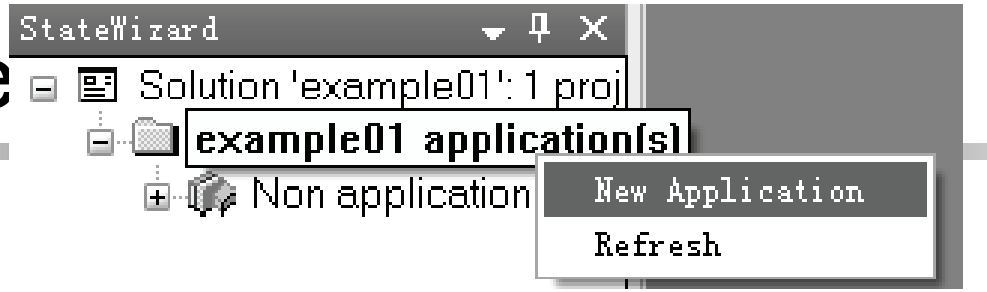
UML StateWizard

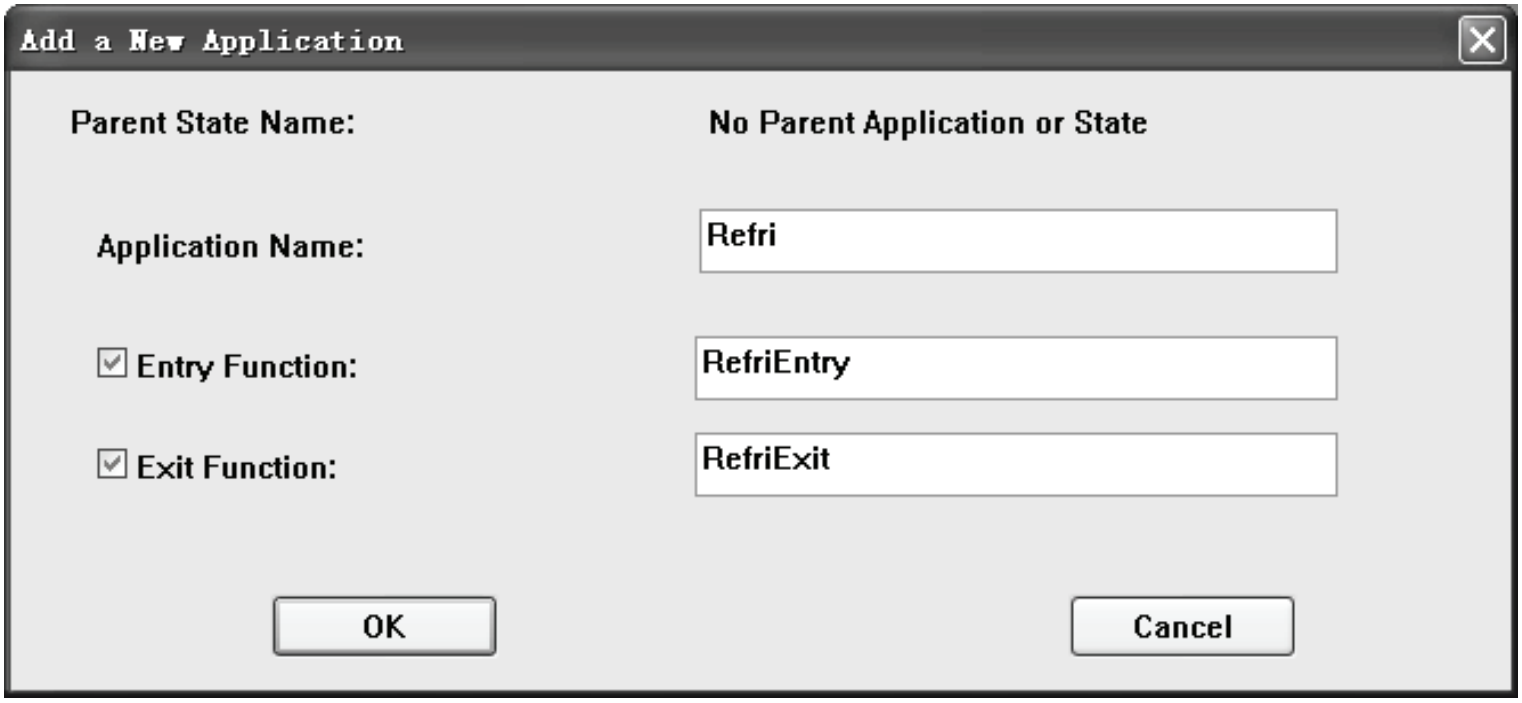
VS＋Statewizard演示



新建一个Dialog-based项目

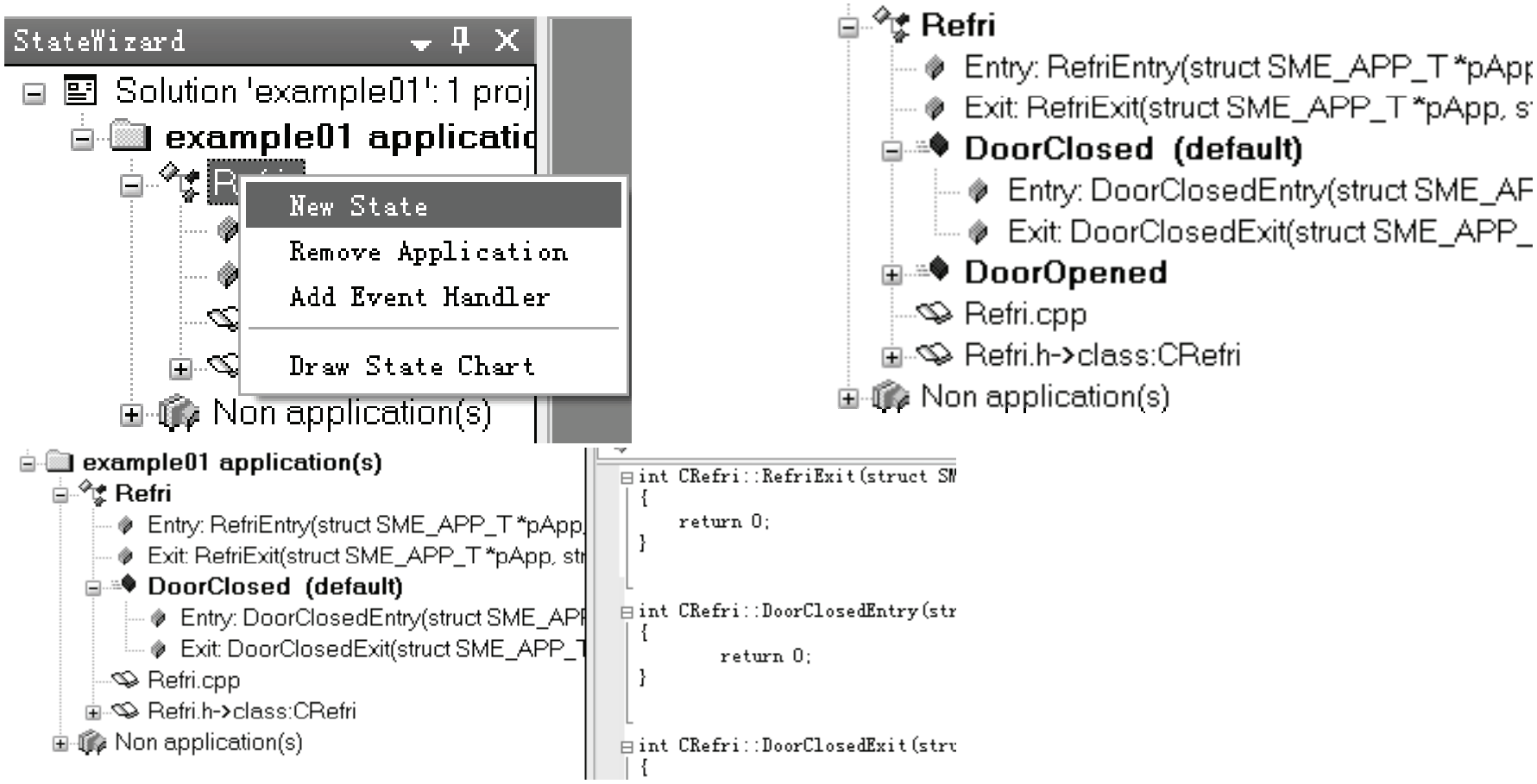
VS＋State





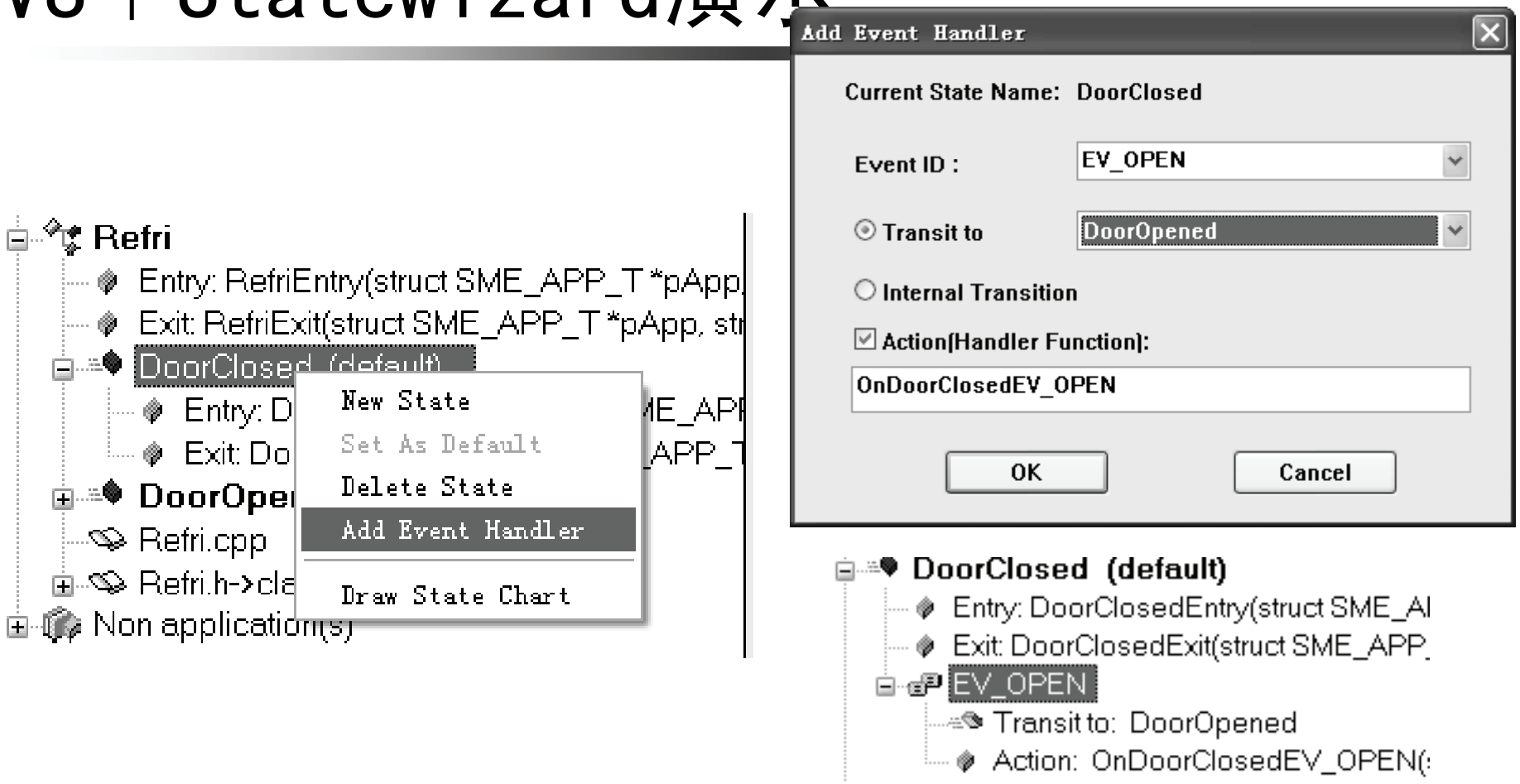
建立冰箱状态树

VS＋Statewizard演示

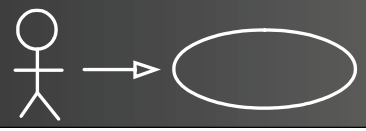


建立冰箱状态树

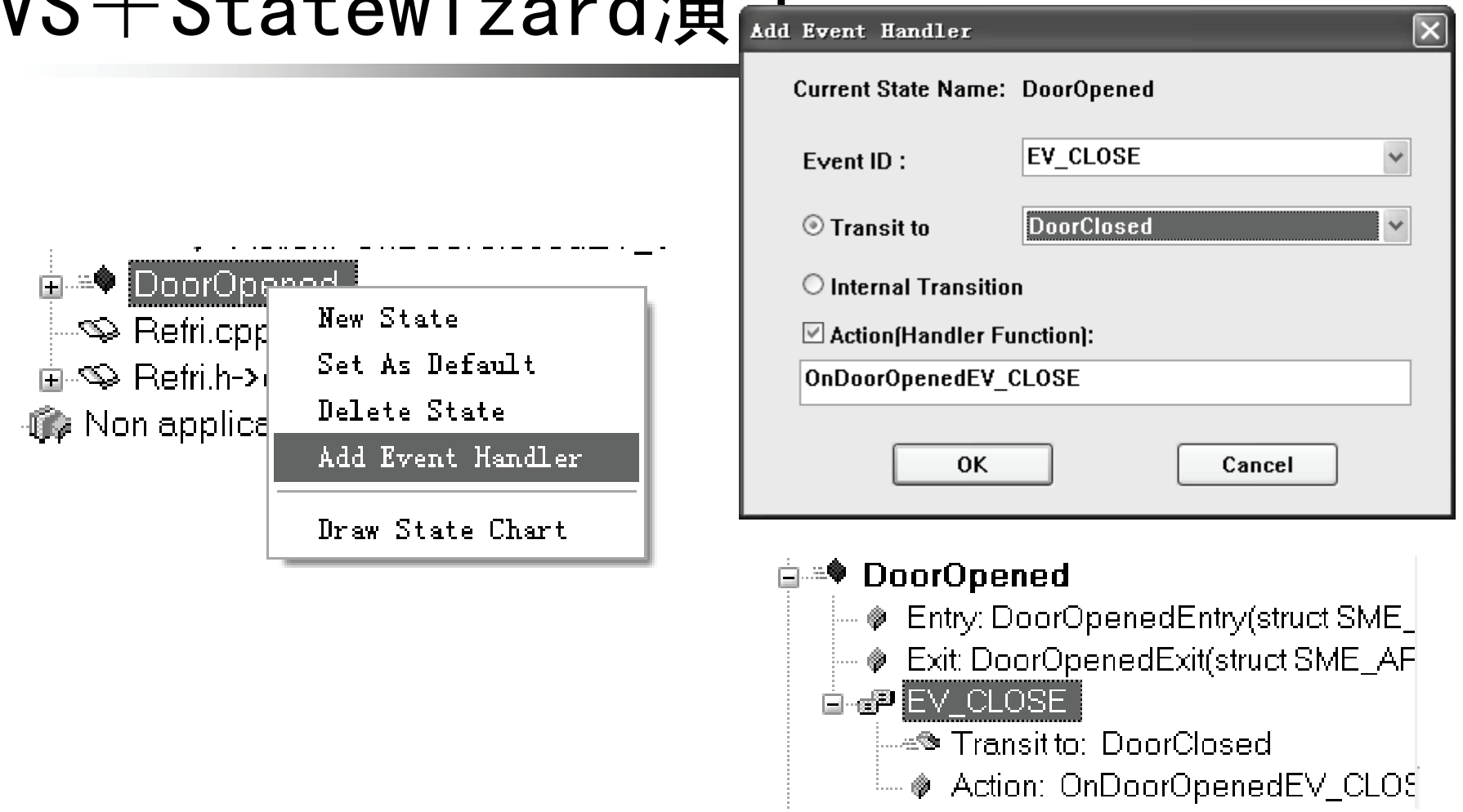
VS＋Statewizard演示



建立冰箱状态树

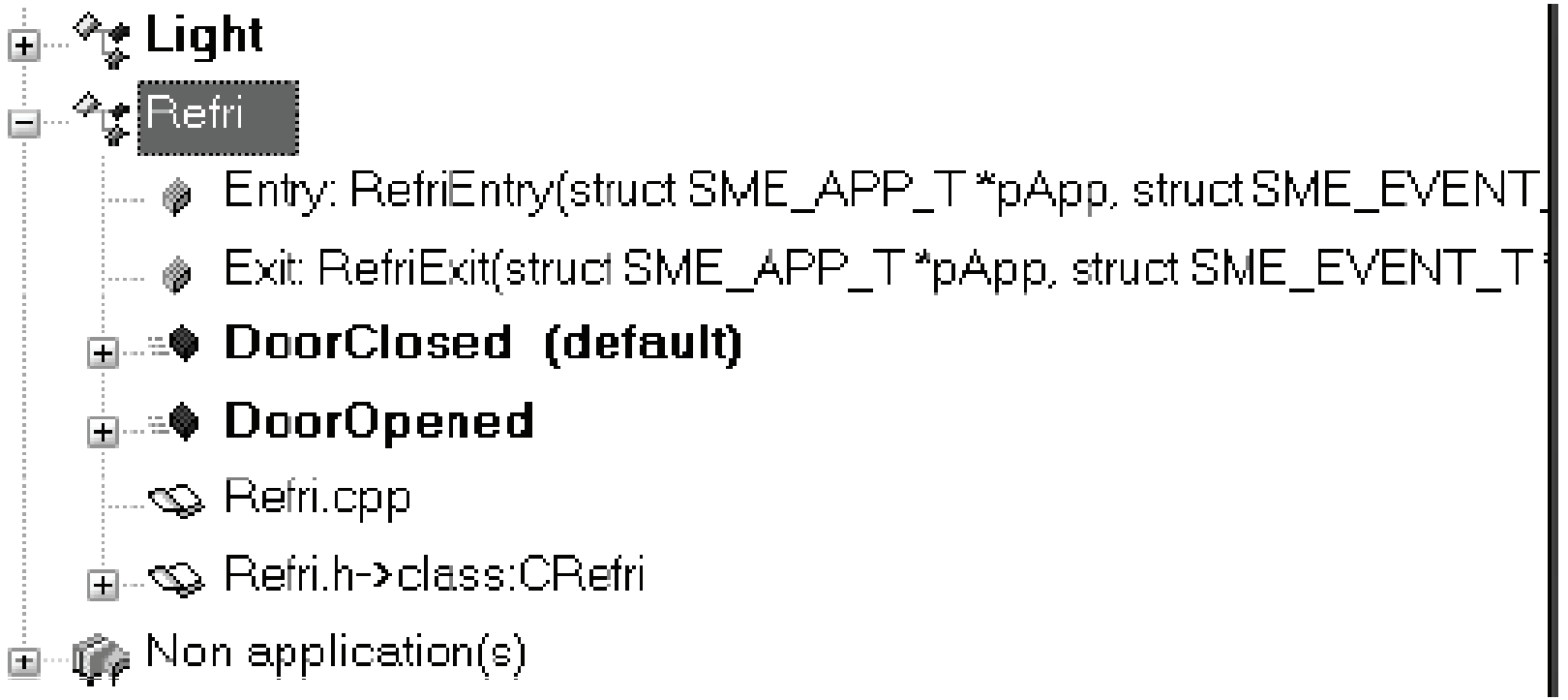


VS＋Statewizard演示



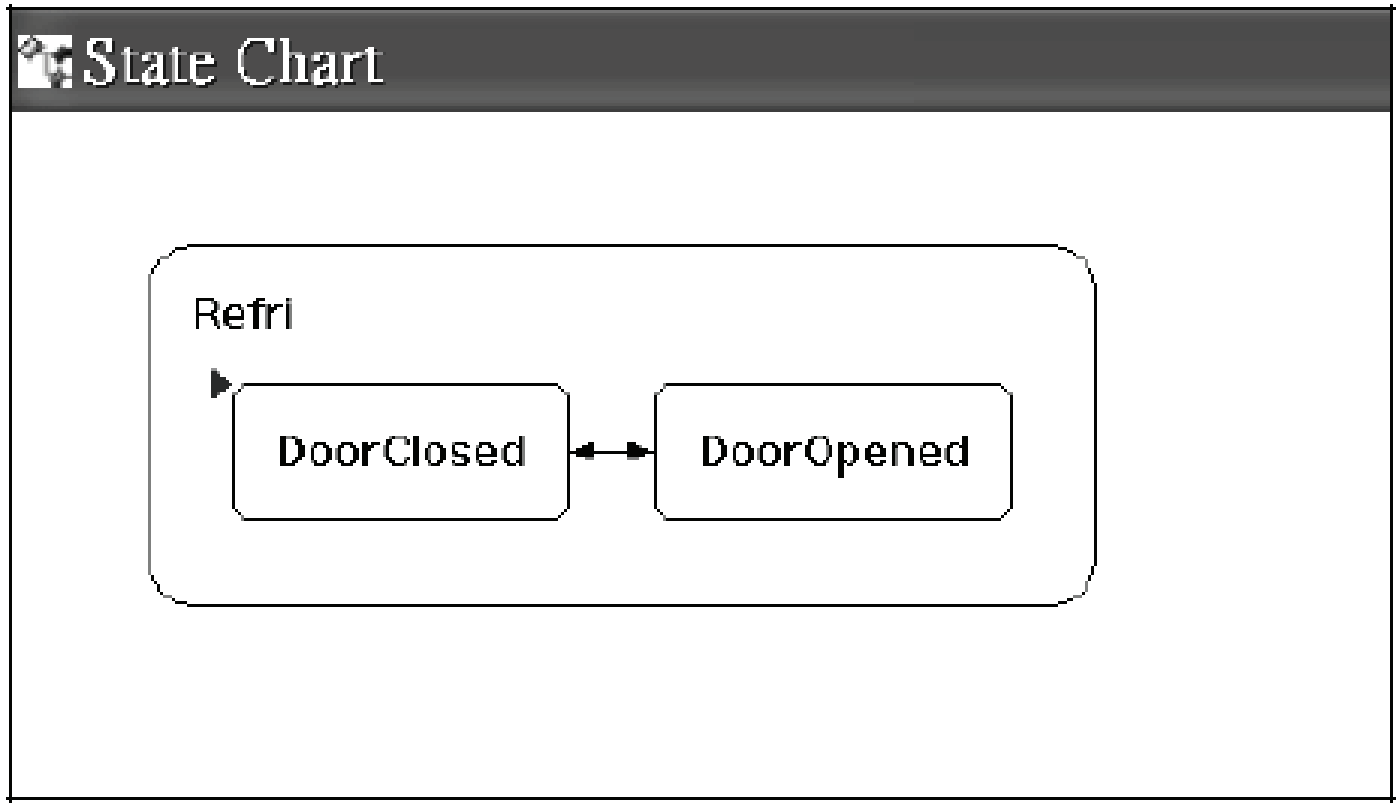
建立冰箱状态树

VS＋Statewizard演示



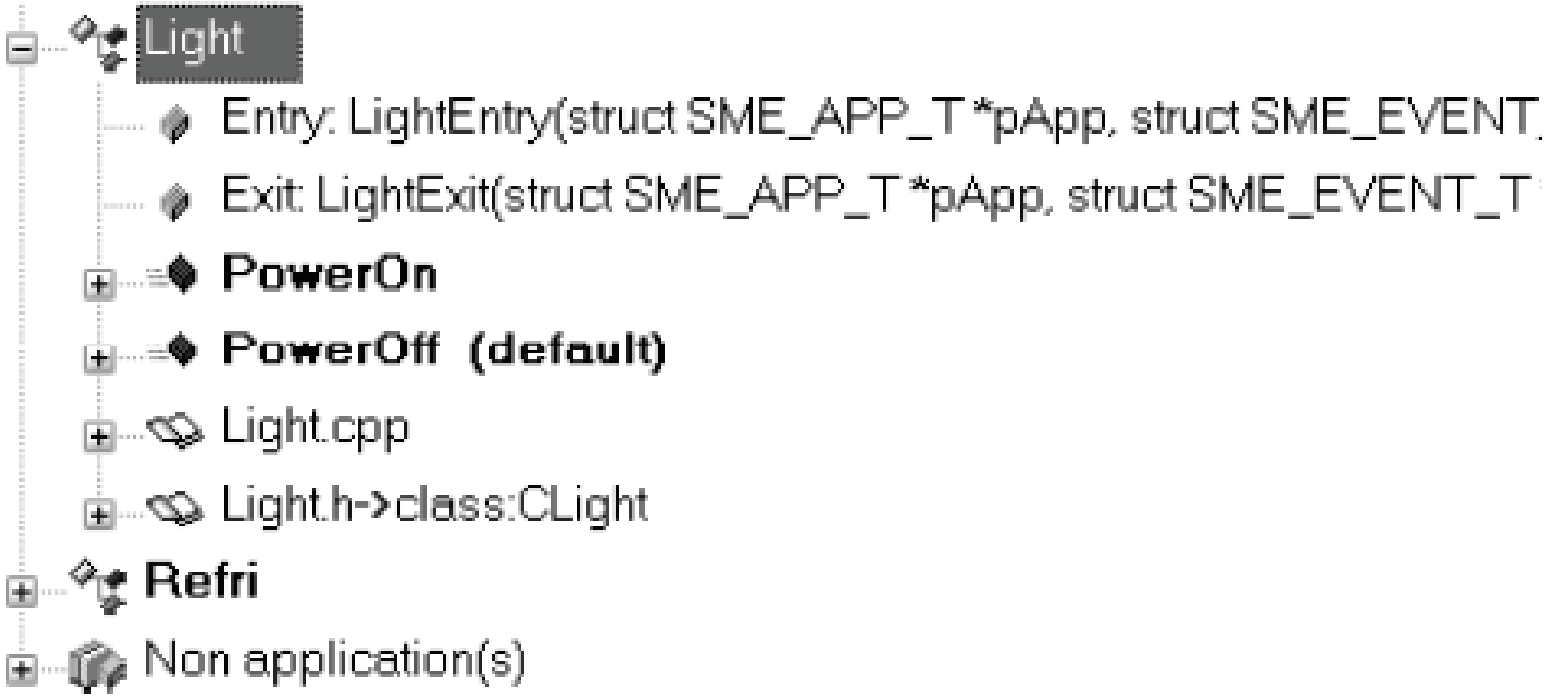
Refri状态树

VS＋Statewizard演示



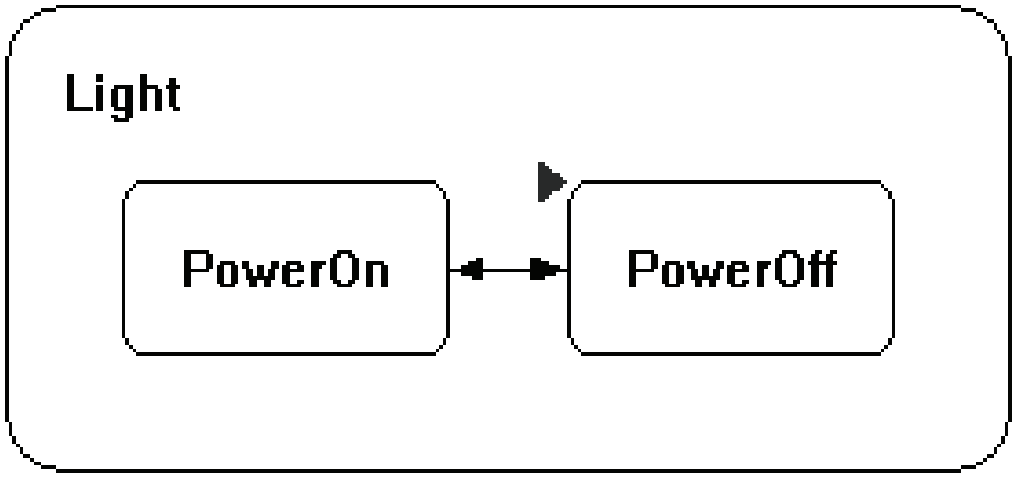
Refri状态图（StateWizard）

VS＋Statewizard演示



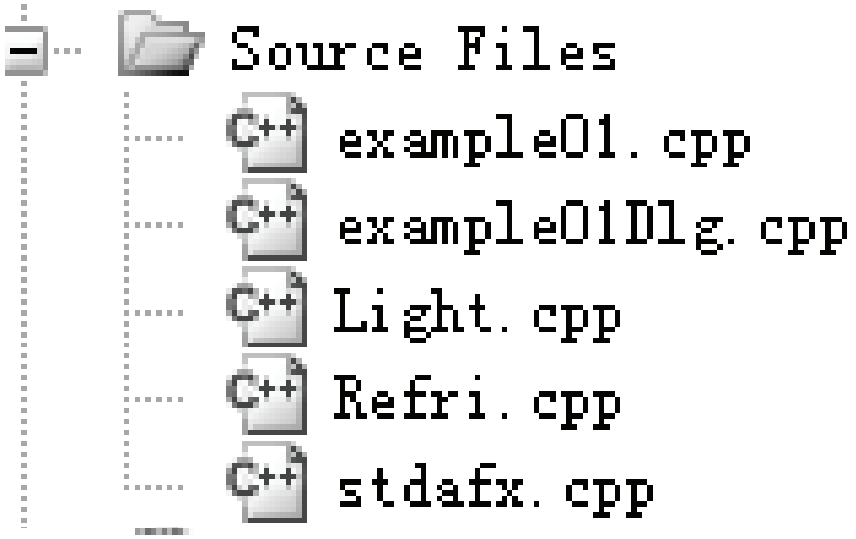
Light的状态树

VS＋Statewizard演示



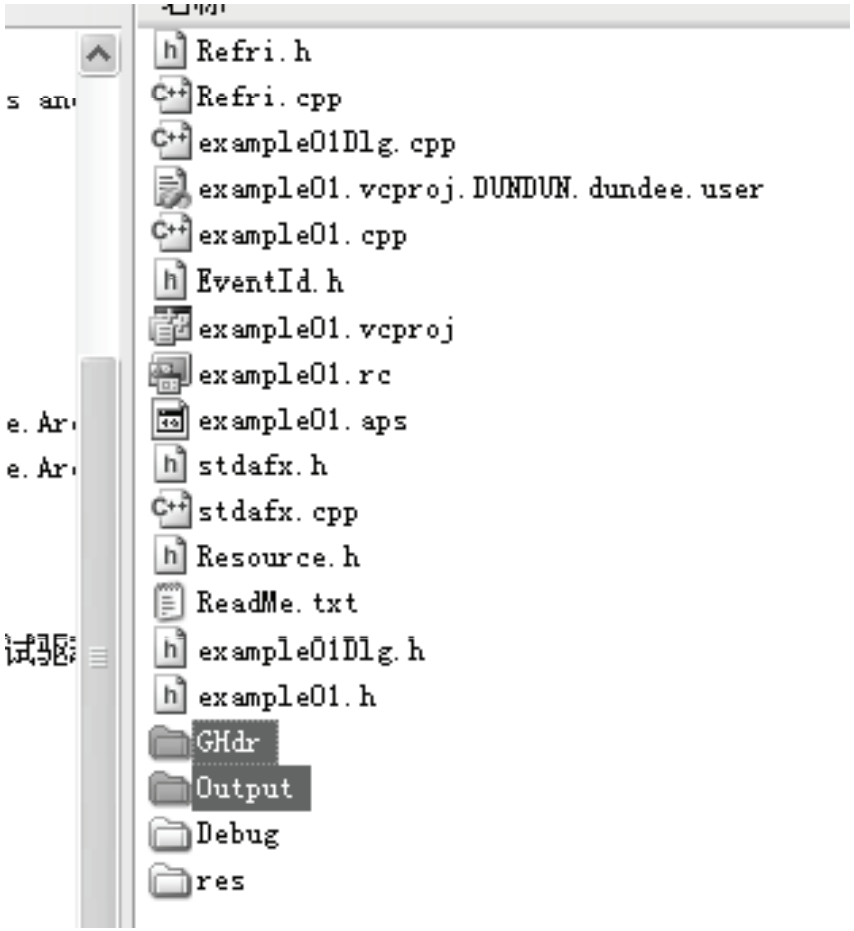
Light的状态图（StateWizard）

VS＋Statewizard演示



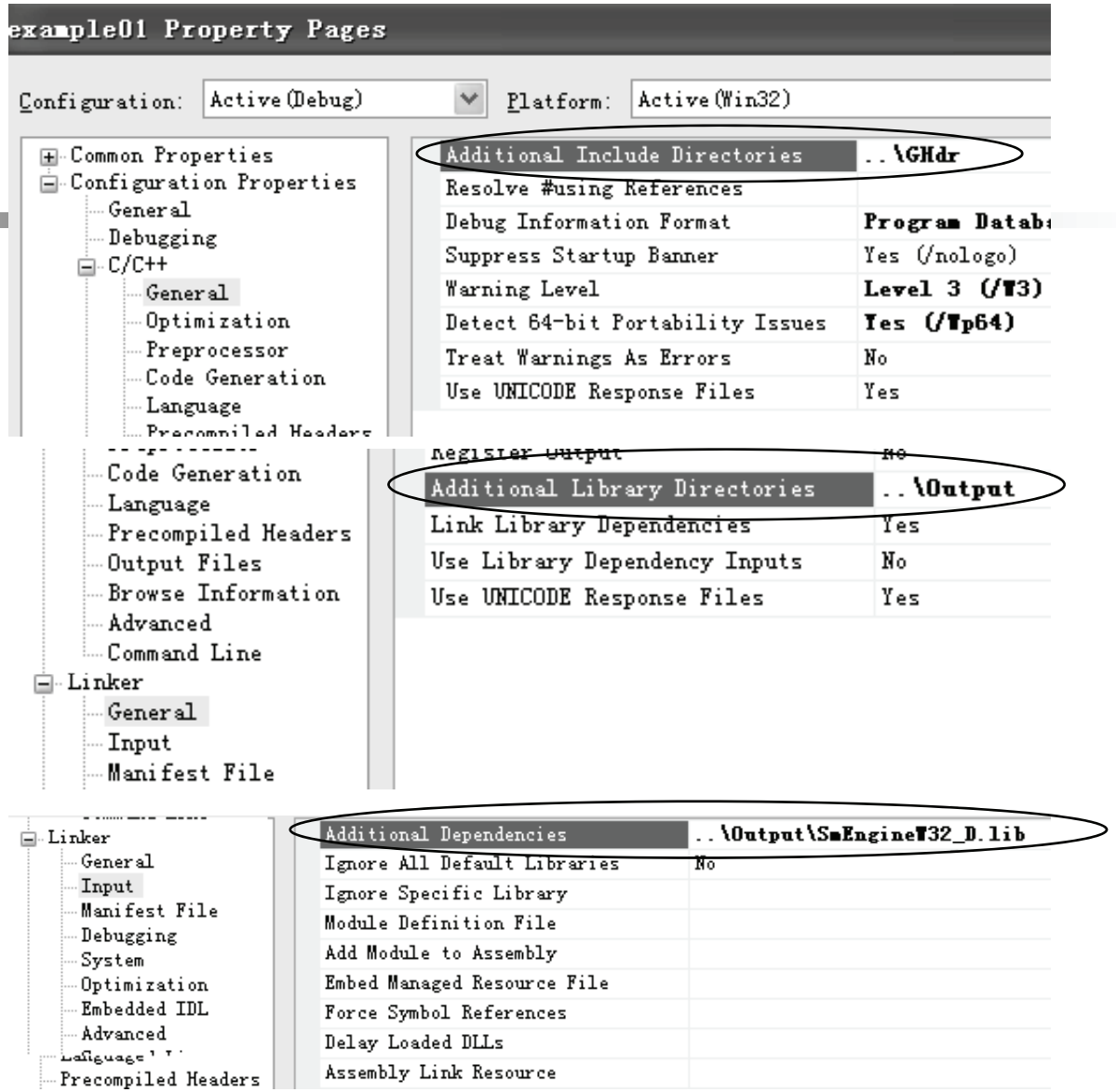
三个模块的代码文件

VS＋Statewizard演示



把Output和GHdr拷贝到项目文件夹

VS＋Statewi



设置项目属性

VS＋Statewizard演示

```
FILE:EventId.h
NOTE: The StateWizard will add mapping
*/

#ifndef EVENTID_H
#define EVENTID_H

#include "sme.h"

#ifdef __cplusplus
extern "C" {
#endif

enum
{
    /*{{SME_EVENT_ID_LIST_DECLARE*/
    EV_OPEN,
    EV_CLOSE,
    EvTurnOn,
    EvTurnOff
    /*}}SME_EVENT_ID_LIST_DECLARE*/
};
```

定义事件常数

VS＋Statewizard演示

```
#include "stdafx.h"
#include "sme.h"
#include "Light.h"
#include "EventId.h"

//=======================
#include "example01.h"
//=======================
```

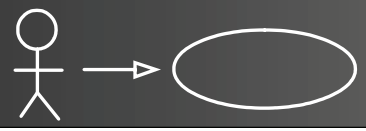
```
SME_APPLICATION_DEF(Light, "Light")

//=======================
extern HWND g_hwndMain;
//=======================

int CLight::LightEntry(struct SME_APP_T *pApp, struct SME_EVENT
int CLight::PowerOnEntry(struct SME_APP_T *pApp, struct SME_EVENT_T *pEvent)
{
    ::PostMessage(g_hwndMain, WM_DRAW_YELLOW_LIGHT, pApp->nAppState,0);
    return 0;
}

int CLight::PowerOffEntry(struct SME_APP_T *pApp, struct SME_EVENT_T *pEvent)
{
    ::PostMessage(g_hwndMain, WM_DRAW_GRAY_LIGHT, pApp->nAppState,0);
    return 0;
}
```

Light模块代码

VS＋Statewizard演示

```
int CRefri::DoorClosedEntry(struct SME_APP_T *pApp, struct SME_EVENT_T *pEvent)
 {
     SME_EVENT_ID_T nEventId = EvTurnOff;
     SME_EVENT_T *pKeyEvent;
     pKeyEvent = SmeCreateIntEvent(nEventId, 0, 0, SME_EVENT_CAT_OTHER, NULL);
     SmePostEvent(pKeyEvent);
     return 0;
 }
int CRefri::DoorOpenedEntry(struct SME_APP_T *pApp, struct SME_EVENT_T *pEvent)
 {
     SME_EVENT_ID_T nEventId = EvTurnOn;
     SME_EVENT_T *pKeyEvent;
     pKeyEvent = SmeCreateIntEvent(nEventId, 0, 0, SME_EVENT_CAT_OTHER, NULL);
     SmePostEvent(pKeyEvent);
     return 0;
 }
```

Refri模块代码



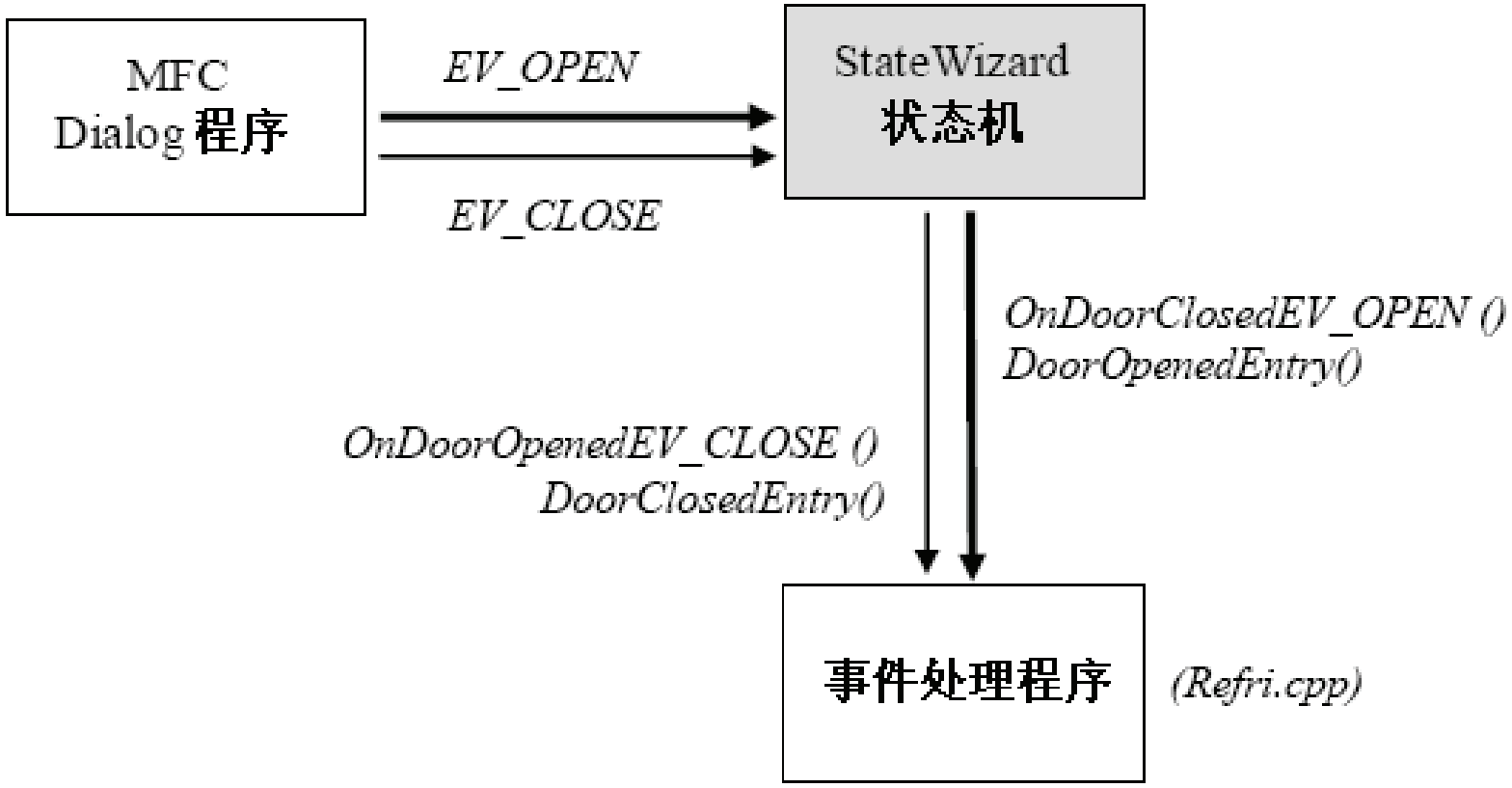
VS＋Statewizard

```
//=============================================
#include "resource.h"
#define WM_DRAW_YELLOW_LIGHT  (WM_APP+1)
#define WM_DRAW_GRAY_LIGHT  (WM_APP+2)
//=============================================

    // 生成的消息映射函数
    virtual BOOL OnInitDialog();
    afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
    afx_msg void OnPaint();
    afx_msg HCURSOR OnQueryDragIcon();
//=================================================
    afx_msg LRESULT OnLightOn(WPARAM, LPARAM);
    afx_msg LRESULT OnLightOff(WPARAM, LPARAM);
//=================================================
    DECLARE_MESSAGE_MAP()
};
    //}}AFX_MSG_MAP
    ON_BN_CLICKED(IDC_BUTTON1, &CexampleO1Dlg::OnBnClickedButton1)
    ON_BN_CLICKED(IDC_BUTTON2, &CexampleO1Dlg::OnBnClickedButton2)
    ON_MESSAGE(WM_DRAW_YELLOW_LIGHT, OnLightOn)
    ON_MESSAGE(WM_DRAW_GRAY_LIGHT, OnLightOff)
END_MESSAGE_MAP()


// CexampleO1Dlg 消息处理程序

//=====================================================
SME_THREAD_CONTEXT_T g_AppThreadContext;
SME_DEC_EXT_APP_VAR(Refri);
//=====================================================
//=============================================
SME_DEC_EXT_APP_VAR(Light);
//=============================================
```

```
// exampleO1Dlg.cpp : 实现文件
//

#include "stdafx.h"
#include "exampleO1.h"
#include "exampleO1Dlg.h"
#include "EventId.h"

//==============================
#include "Refri.h"
#include "SrvAgent.h"
//==============================

//====================
#include "Light.h"
//====================

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

//==============================
HWND g_hwndMain=NULL;
//==============================
  //启动状态机
  g_AppThreadContext.nAppThreadID = 0;
  SmeInitEngine(&g_AppThreadContext);
  MfcHookWnd(GetSafeHwnd());
  SmeActivateApp(&SME_GET_APP_VAR(Refri),NULL);
  //=================================================
  SmeActivateApp(&SME_GET_APP_VAR(Light),NULL);
  //=================================================

   SetIcon(m_hIcon, TRUE);          // 设置大图标
   SetIcon(m_hIcon, FALSE);         // 设置小图标

   //==============================
   g_hwndMain = m_hWnd;
   //==============================
```

MFC Dialog模块代码

VS＋Statewizard演示



运行模拟

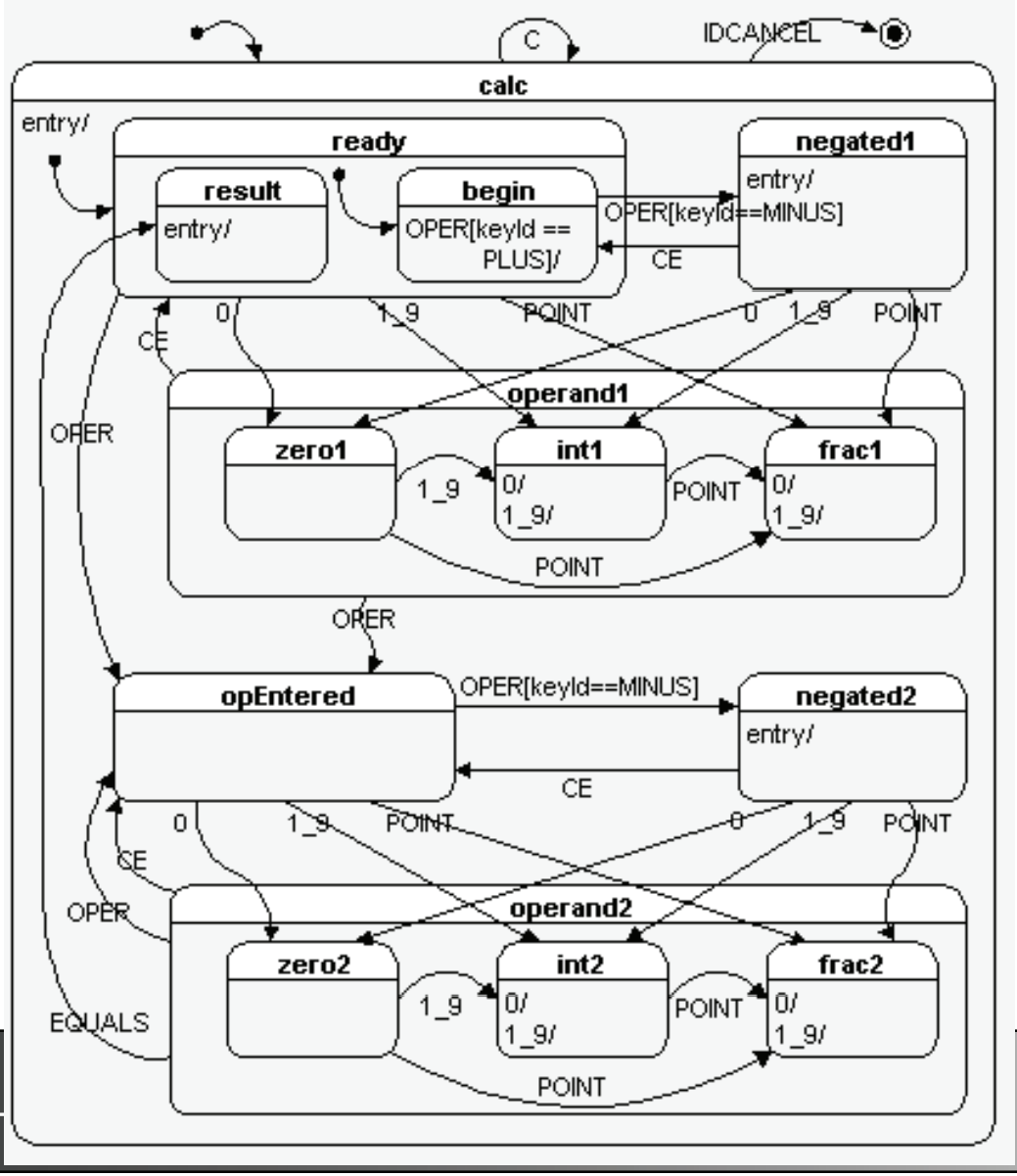
VS＋Statewizard演示

操作界面

VS＋Statewizard演示

按下关门按钮

状态图



计算器

量子框架实现