

目录

- ❖ eM-plant仿真系统简介
- ❖ 物流系统基本建模对象
- ❖ simtalk仿真语言
- ❖ 三维仿真



eM-plant仿真系统简介

eM-plant仿真系统是德国 tecnomatix公司的产品，其前身为simple++，专门用于“离散系统”建模与仿真分析。

。

特点：

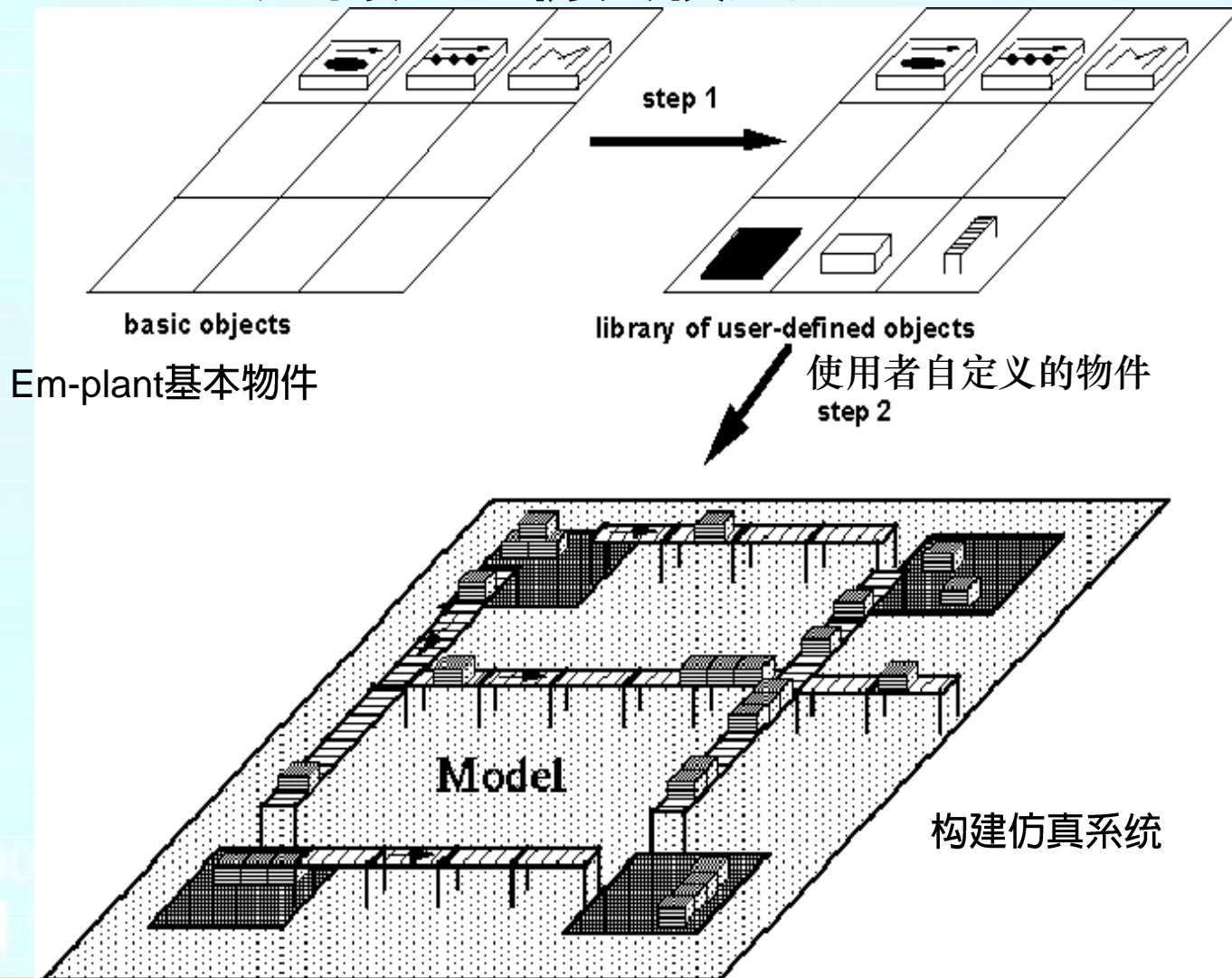
- 面向对象建模
- 集成仿真环境
- 仿真过程可视化
- 专用仿真语言
- 开放数据接口
- 2D+3D
- 提供如GA,ARIS,Gantt等模块



物流系统基本建模对象

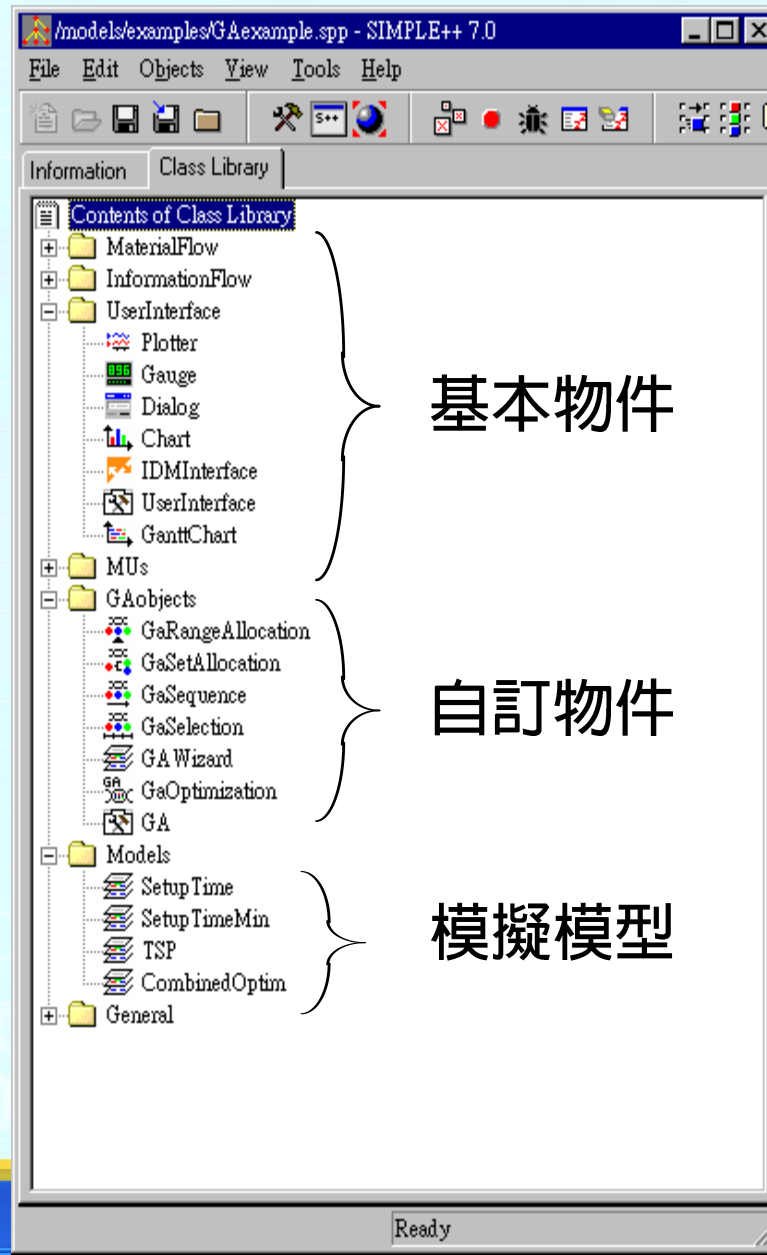
如何建立仿真模型

Modeling



树状结构的 物件库

Modeling



基本物件

自訂物件

模擬模型

§ 3.2 物流系统基本仿真要素分类

物流系统的功能要素：

- 运输、仓储、装卸搬运、包装、流通加工、配送和信息。

构成任何一个物流系统的仿真要素：

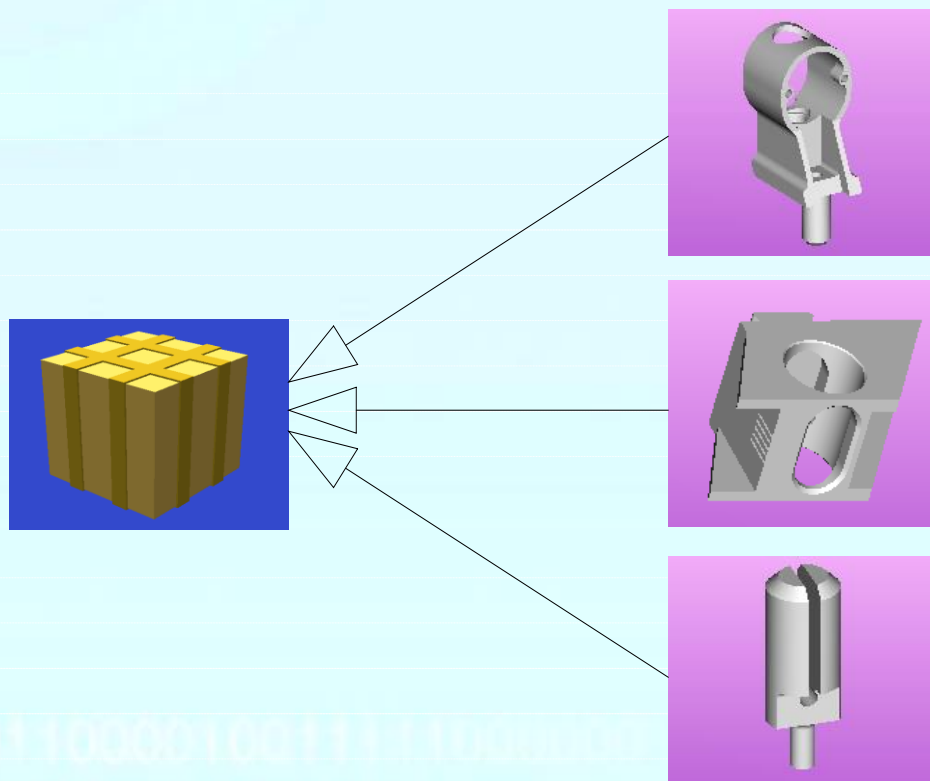
- 物料（流动实体/临时实体）
- 运输单元
 - 连续运载工具：辊道、悬挂、皮带、管道等
 - 离散运载工具：汽车、插车、火车、飞机、轮船等
- 加工单元：包装、流通加工等
- 仓库
- 信息流



§ 3.3 基本物流要素建模

1、物料（流动实体/临时实体）

- 在离散仿真系统中，不能表示流体或散料，只能是单元化的实体。  --Entity



物料的缺省属性只有“长度”，可以根据需要，用户自己定义其他属性（重量、类型、ID、甚至条码等）

物料是由专门的“对象source”按一定的规律产生,离开系统时由“drain”接收

- 一种物料，固定时间间隔（缺省）

- 一种物料，随机间隔

- 一种物料，按计划成批产生

- 多种物料，交替产生（固定、随机、成批）

示例：ex3_3_1

§ 3.3 基本物流要素建模

2、运输单元

- 在eM-plant中，运载工具分为：移动单元和固定单元
 - 移动单元--离散运输工具如：AGV、插车、汽车等
 - 固定单元--连续运载工具如：道路、辊道、链条、皮带等

(1) 移动单元

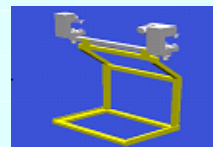
- 托盘（集装箱等） --container



- 运输车

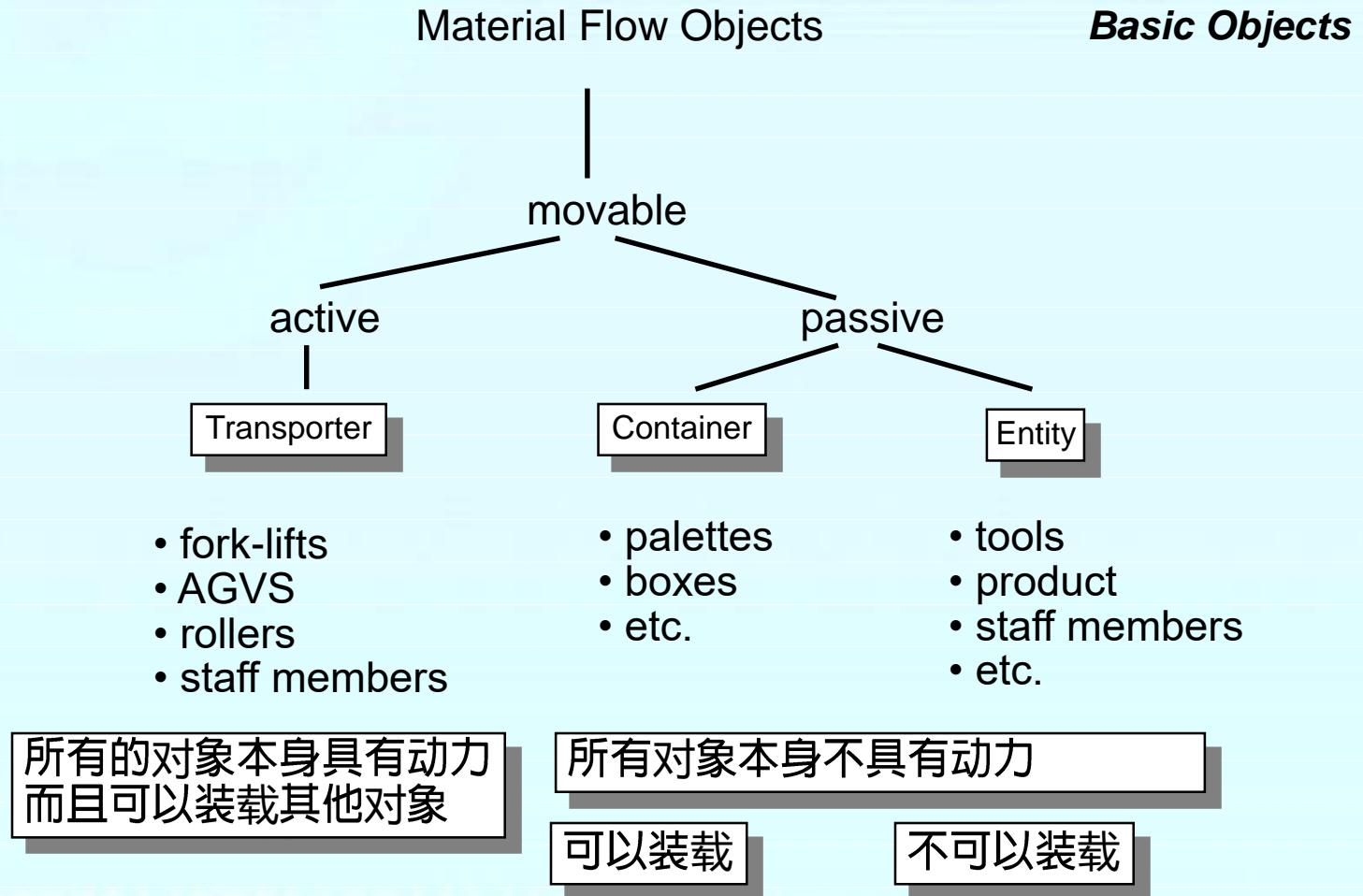


--transporter



container --自己不能移动，可以设定装载量。
transporter--可以设定其装载量和行走速度。

物流类对象中的可移动对象 (MU' s)



(2) 固定单元

道路

--track



辊道等

--line



track--可以设定：长度、容量和方向,只能用于
transporter




line --可以设定：速度、长度、容量和方向,自带动力

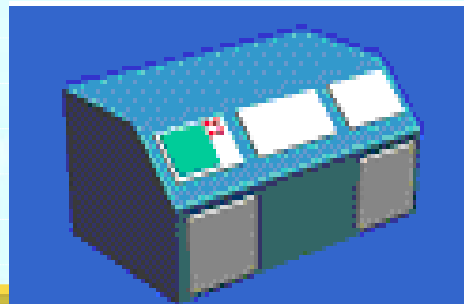
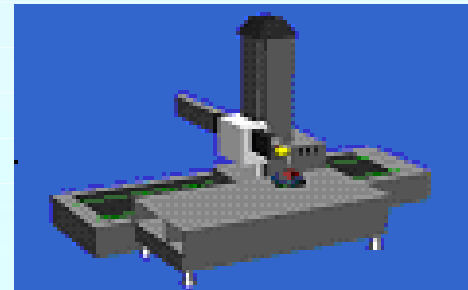
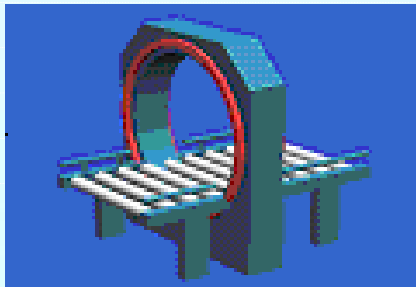
示例： **ex3_3_2**

§ 3.3 基本物流要素建模

3、加工单元

eM-plant的加工单元包括：

- single proc 
- paralle proc 
- assembly 装配站 
- dismantle station 拆站 



四个时间属性

processing time

set-up time

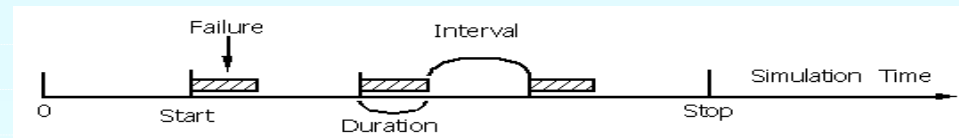
recovery time

cycle time

四种状态

- failed
- paused
- entrance locked
- exit locked

检修仿真



start--duration--interval-- stop

availability % 和 mean time to repair --MTTR

related to time

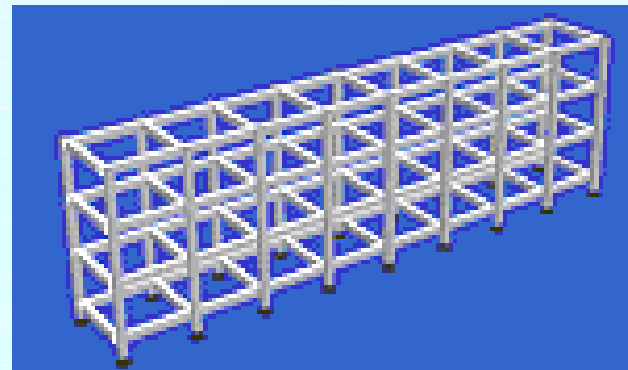
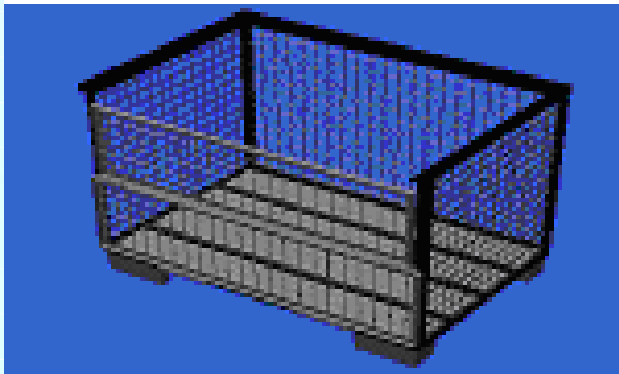
示例: ex3_3_3

§ 3.3 基本物流要素建模

4、存储单元

eM-plant的存储单元包括：

- buffer
- iobuffer
- sorter
- store



共有属性（有差别）

容量、状态、检修、时间

(1) **buffer**

缺省--先进先出

可细分存放单元--

```
entrance>buffer[1],buffer[2],...buffer[capacity]>  
exit
```

(2) **iobuffer**

不能细分存放单元

可以选择类型: **Queue stack**

示例: ex3_3_4

§ 3.3 基本物流要素建模

4、存储单元

(3) sorter

- 给每一个进入sorter的临时实体赋一个权值
- 按权值的大小，升序或降序确定离开顺序
- order --升序或降序
- time of sort --确定排序时间（有新实体进入或离开）
- sort criterion -- 赋权值的方式
 - 在sort中已经停留的时间，升序--先进先出；降序--先进后出
 - 临时实体的属性（长度、能力、速度、需要被加工的时间等等）
 - 方法---由用户自己确定

(4) store

- 能细分存放单元 (X--Y)
- 不能主动出入库
- store.pe(x,y) ---- store[x,y]

示例：ex3_3_4

其他

FlowControl对象是为实现物料流的分解和合并而设置的，它是物流控制对象。**FlowControl**对象并不对经过的**MU**进行加工，它只是按照既定的策略将经过的**MU**分配给其后续的其他物流对象上。

Broker对象和Exporter对象

Broker这个单词的本意是中间人，在EM-PLANT中Broker对象就起着这样一个“中间人”的作用。Broker为一些物流对象例如Singleproc、Paralleproc、Assembly等提供服务，这些物流对象都有一个Importer属性栏，Importer属性栏用于选择这些物流对象需要提供服务的“中间人”即Broker，而每一个Broker又有若干服务资源-----即Exporter对象，Broker接受Singleproc等物流对象的服务请求，按一定的原则去分配有限的Exporter对象，使物流系统在有限的服务资源下达到优良的整体效益。因此Broker对象和Exporter对象是必须搭配使用的，而Broker的Exporter分配原则必须由用户自己编写。

§ 3.4 基本信息流要素建模

基本信息流要素包括：

- 方法--method 
- 全局变量--variable 
- 表---tablefile 
- 卡片--cardfile 
- 堆栈/队列-- stackfile / queuefile 
- 时间序列--timesequence 
- 触发器--trigger 
- ShiftCalendar对象
- AttributeExplorer对象
- Generator对象
- FileInterface对象
- FileLink对象



§ 3.4 基本信息流要素建模

方法--method



方法模块--是物流与信息流的“接口”，它将物流的“control”属性和“simtalk”信息处理和控制程序连在一起，并为simtalk程序提供了一个“框架”，每一个方法相当于一个“函数”或“子程序”。








方法分为：

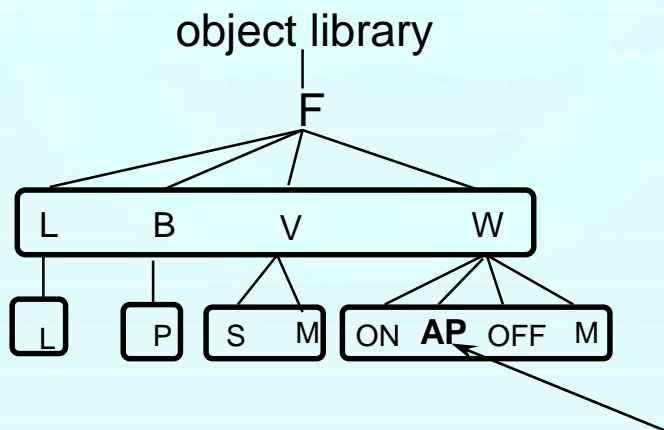
- 特殊方法：endsim、init、reset等
- 用户定义方法。

```
(x,y:integer):integer
is
    i:integer;
do
    i:=x+y;
    result:=i;
end;
```

示例：ex3_3_5(调试方法)



图标	名称	图标含义说明
	Default	Method 的默认图标
	ExitCtrl	物流对象的出口控制方法
	EntranceCtrl	物流对象的入口控制方法
	Init	仿真模型的启动控制方法
	Reset	仿真模型的重置控制方法
	EndSim	仿真结束的控制方法
	Error	调试出错的方法



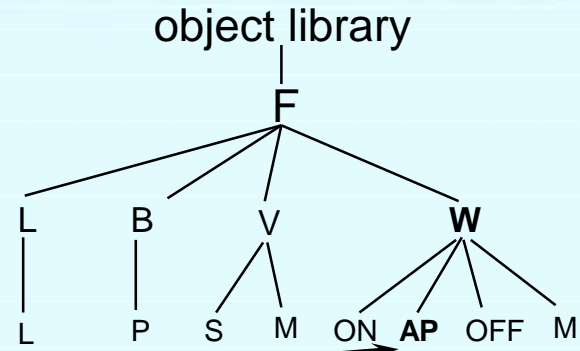
相对路径默认为从主对象所在的命名空间；例如，方法“M”（在“W”模型中，“W”模型又在“F”模型中）使用了“AP”则em-plant会自动找到当前命名空间的“AP”对象。

- 在对象object.F.W.M中键入如下代码：

```
is
do
  print AP;
end;
```

- 点“Apply”和Start“并在Console界面中观察结果。

绝对路径通常从对象结构树的最顶层开始，逐层向下，以分割符“.”分开。



例如需要调用图中的AP对象，则需从最顶层对象结构树开始，调用代码为：

. F. W. AP

§ 3.4 基本信息流要素建模

全局变量--variable



全局变量的作用域：frame

全局变量一般类型：integer,real,string,...

全局变量特殊类型：object,table,list,...

示例：ex3_3_6

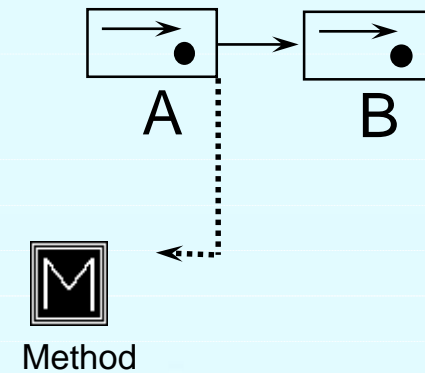


可移动对象的移动方法

Linking Material and Information Flow

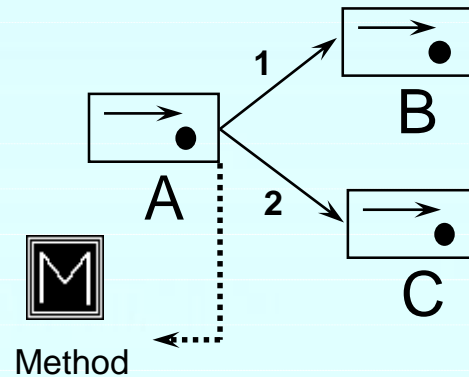
Move to B

```
@.move;  
@.move(B);  
@.move(1);
```



Move to C

```
@.move(C);  
@.move(2);
```



Comment对象Comment对象是EM-PLANT中提供辅助功能的对象，它用来记录模型开发人员对模型的注释，通过这些注释可帮助不同的用户能够更好地对模型进行理解。

示例：ex3_3_8

StackFile、**QueueFile**、**CardFile**、**TableFile**同是EM-PLANT中提供信息存储的对象，这四类对象各有不同的特点。**StackFile**对象是一纬的数据存储对象，其特点是采用后进先出（**LIFO**）的存储策略。

QueueFile对象也是一纬的数据存储对象，其特点是采用先进先出（**FIFO**）的存储策略。

CardFile是一类可自由存取的一纬数据存储对象，它类似一个文件柜，用户可增加、删除、读写存储在任一位置的的数据。

TableFile类似于数据库的表，是二纬的存储对象。**TableFile**对象可以用来收集、保存各种仿真数据和结果，**TableFile**由行、列组成，每列的数据类型可以根据需要来设置，在仿真过程中，用户可以添加、删除行、列的数目或读写任一单元格内的数据。

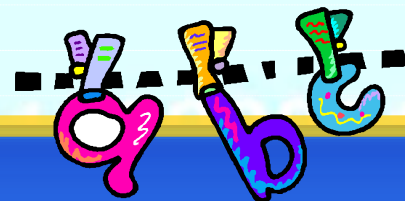
§ 3.4 基本信息流要素建模

触发器--trigger



Trigger的含义为触发器，它可在仿真运行过程中按照用户定义的模式来更改某一物流对象的属性值或Variable对象的值，它也可触发Method对象来执行预先编辑好的代码。例如工厂里的设备有统一的停机时间，每运行1小时就要休息5分钟，为了实现这个功能就可设置Trigger对象每隔55分钟发出讯息让设备停机，再隔5分钟发出讯息让设备启动。此外，Trigger对象的功能还有：

- 利用触发器控制source产生临时实体的时间、数量和类型
- 利用触发器改变全局变量的值
- 修改或设置物流对象的属性



Shiftcalendar:

ShiftCalendar对象是**EM-PANT**提供的一个对企业工作日志建模的非常有效工具，它可建立每一年、每一月、每一天、每一分钟的详细工作计划。例如有的物流系统周一到周五工作，五一、十一等法定节假日休息，每个正常的工作日中8:00到17:00上班，其中12:00到13:00休息，9:15到9:30、15:15到15:30倒班，这样一个复杂的企业工作日志可以在**ShiftCalendar**对象中迅速地完成建模工作。在同一个仿真模型中用户可建立多个**ShiftCalendar**对象，不同的**ShiftCalendar**对象可用于管理不同的物流对象，这样同一个仿真模型中的物流对象可按照不同的工作日志来工作。

示例：ex3_3_8

Attributexplorer:

AttributeExplorer对象用于管理某一物流对象的某一属性。**AttributeExplorer**对象可以将这些属性的值从它所属对象中读出，这些值可以浏览器的形式显示，并可将用户修改的值反馈回这些物流对象中去。

示例：ex3_3_8

Generator对象

在离散事件仿真中，有些事件之间存在着关联，例如“工件运送到空闲的设备上”这一事件完成后，“工件被加工”这一事件就要发生，而有些事件是与其它事件没有关联关系的，例如“某一工人有事不能来上班”这类事件则完全与系统内的其他事件没有任何关系，对于这类事件EM-PLANT专门设计了Generator对象来处理。

Generator对象可按固定的时间点来激发Method对象，它也可按固定的或随机分布的时间段周期性地重复激发，

示例：ex3_3_8

FileInterface对象

FileInterface对象是**EM-PLANT**提供的与外部文本文件的接口，它可在仿真过程中它可完成对指定的文本文件读写过程。其属性对话框主要用于选择所要操作的文本文件，选定文件后**EM-PLANT**就可将**FileInterface**对象当作选定的文本文件来操作。

示例：ex3_3_8

FileLink对象

FileInterface对象是**EM-PLANT**提供的与外部可执行文件的接口，它可在仿真模型需要的时候运行这个可执行文件。操作方法为在文档浏览器中选择文件拖入**Frame**框架即可。

示例：ex3_3_8

用户接口对象

Chart对象

Plotter对象

Gauge对象

Report对象

Dialog对象

Chart对象可以采用柱图、饼图等图表的方式将仿真全过程所需记录的数据集显示出来，因此它所显示的图形是动态的。**Chart**对象有两种方式显示动态的数据，一种方式是将其与**Table**对象连接起来，用**Table**对象来记录仿真所产生的数据；另一种方式是定义**Chart**对象的**Input Channels**，**Chart**对象会自动收集它本身所在**Frame**里所有物流对象的状态信息，用户可通过定义**Input Channels**来确定要显示那些物流对象的状态，**Chart**对象则会在仿真过程中将这些对象的状态加以显示。

示例：ex3_3_9

Plotter对象可用曲线图的方式显示一些数据在某一段仿真过程变化的情况，**Plotter**对象可在同一个图形中显示多条曲线，每一条曲线对应一个要显示的数据源，用户可定义每条曲线的颜色、线型以及每个数据源的零点。

Plotter对象可依据用户的指令来定义仿真的模式，包括 **Simple mode** 和 **Plot mode** 两种模式，**Simple mode** 是周期性地读数据源的数据并显示，**Plot mode** 是在每一个仿真事件结束后读数据源的数据并显示。

示例：ex3_3_9

Gauge对象可在整个仿真过程显示某一数据的值，这个数据可为某一对象的某一属性。当**Gauge**对象没有被激活时，**EM-PLANT**只显示它原始的图表，而当**Gauge**对象被激活时，**EM-PLANT**则显示它所对应数据的值。

Gauge对象有两种显示方式，**Sample**模式和**Watch**模式，**Sample**模式是**Gauge**对象周期性地更新它应显示的值，**Watch**模式是**Gauge**对象只在它所对应数据的值发生改变时才加以更新。**Gauge**对象可以文本、进度条、饼图三种方式来显示数据，其中文本方式用于显示字符型数据，进度条和饼图用于显示数字型数据。

示例：ex3_3_9

Report对象用于显示最终的仿真结果，它可将仿真结果以表格和图形的方式表现在**HTML**网页上。用户可将仿真结果打印、保存甚至发布到网络上去。仿真模型的开发用户也可通过编写**HTML**代码或**java**代码来改进**Report**对象所建立的原始的**HTML**程序结构。

示例：ex3_3_9

EM-PLANT之所以建立Dialog对象主要基于以下两个原因：

- 1) 为一个复杂的仿真模型建立简单易操作的用户接口，从而使其他对EM-PLANT并不熟悉的用户可以很方便地操作这个复杂的模型；
- 2) 可阻止其他用户查看仿真模型的结构和代码，Frame对象有一个“Argument for open”属性，该属性可右击Frame对象选择“Attributes”来更改，“Argument for open”属性可指向一个Dialog对象，这样设定以后再有用户双击Frame对象将不再打开Frame对象的建模框架，而是显示指向的Dialog对象。

原始的Dialog对象包含4种元素，静态文本、文本框、下拉菜单和按钮，用户可将这些元素设置在Dialog对象中合适的位置上完成Dialog界面的编辑，同时可在Method中编写这些元素的控制方法。升级后的Dialog对象除了以上四种元素外还包括选择框、列表框、列表视图等元素，利用这些元素可编辑出可与VC相媲美的对话框。

示例：ex3_3_9

Interface对象和Frame对象

二者通常结合起来表示不同的物流系统的不同层次。

示例：ex3_3_10

Exercise

把下列现实世界的物体转化为前面讲到的仿真对象，并把仿真对象的特点与下表对应。

	material flow elements	information flow elements	moveable	stationary	active	passive
conveyor belt						
fork lifting truck						
product						
work plan						
drilling machine						
AGV						
warehouse						
container						
assembly station						
worker						
bill of material						

Exercise

	material flow elements	information flow	moveable	stationary	active	passive
conveyor belt	X			X	X	
fork lifting truck	X		X		X	
product	X		X			X
work plan		X		X		X
drilling machine	X			X	X	
AGV	X		X		X	
warehouse	X			X		X
container	X		X			X
assembly station	X			X	X	
worker						
bill of material		X				X

simtalk仿真语言

一、常数

- 预定义常数: e 和 pi
- boolean : true false
- integer: -2147483648~2147483647
- real: 3.12, 6.12E2
- string: “abc”
- time,date,datetime 需要替代或转换:
 - real 或 integer
 - str_to_time days:hours:minutes:seconds
 - str_to_date year/month/day
 - str_to_datetime

ex4_1 (熟悉程序调试方法和breakpoint/debug)



二、数据类型和变量

- 数据类型

- boolean, integer, real, string, date, time, datetime
- length, speed, weight, money
 - real 国际标准单位，显示设置有关 (tool->options)
- list, quene, stack, table--与tablefile等功能相同，但不是“对象object”
- object
- **any**--在程序中可以表示任何类型，但是一旦确定为某种类型则不能改变。

ex4_2



二、数据类型和变量

- 变量说明

- 全局变量--object 选择类型
- 局部变量-- **V:integer;** 位于method中 is--do之间
- 参数和返回值-- **(v1:integer;v2:real):boolean** 位于method中 is之前

- 作用域

- 全局变量--所有folders,frames,methods
- 局部变量和参数--method

ex4_3



三、运算符

Priority	Operator	Description
highest	()	parenthesis
	~, NOT	negation
	*, /, //, \	multiplication, division, integer division, modulo
	+, -	addition, subtraction
	<, <=, =, /=, >=, >	smaller than, smaller or equal, equal, not equal, greater than equal, greater
	AND, OR	logic and, logic or
lowest	:=	assignment



四、库函数

- 算术函数
 - 基本算术函数
 - 三角函数
 - 分布函数
- 字符串函数
- 时间函数
- 系统函数
- 输入/输出函数
- 调试函数



四、

Function	Result
abs(x)	absolute value of x
beta(x,y)	beta function $B(x,y)$, $0 < x,y < 1$
betai(x,y,z)	incomplete beta function $I_z(x,y)$, $0 < x,y < 1$, $0 < z < 1$
ceil(x)	round up
exp(x)	exponential function
floor(x)	round down
gamma(x)	gamma function Γ
log(x)	natural logarithm
log10(x)	logarithm to the base 10
max(x,y)	maximum
min(x,y)	minimum
pow(x,y)	x^y When the basis is 0, the exponent has to be greater than 0. When the exponent is not an integer, for example -0.5, the basis has to be greater than or equal to 0.
round(x)	rounds from y on. Example: <code>print round(1.6666,2)</code> returns 1.67.
round(x,y)	rounds up or down from y on
sqrt(x)	square root. The argument of data type <i>real</i> has to be 0.

z_weibull(s,Alpha,Beta)

Weibull distribution



四、库函数

(2) 字符串函数

ascii, chr, copy, incl, omit, pos, strlen, toLower, toUpper, trim.

(3) 时间函数

day, dayOfWeek, dayOfYear, getDate, month, setDaylightSavingTime, timeOfDay, week, year.

(4) 系统函数

copyFile, copyTextToClipboard, currentEventCtl, execute, exitApplication, getEnv, getTextFromClipboard, messageBox, **ref**, sleep, sysDate, sysInfo,.....。

ex4_4 (ref)



四、库函数

(5) 输入/输出函数

输入函数: **prompt**, promptlist1, promptlistn.

输出函数: beep, bell, **getUnit**, **print**, **promptMessage**

ex4_5

(6) 调试函数

checkArguments, **debug**, deleteSuspendedMethods,
ignoreBreakpoints, setErrorStop,....。



simtalk 控制指令

一、注释和赋值语句

-- comment to the end of the line

price:= price* 1.16; -- value added tax

/* beginning of a long comment

that covers several

lines to document a feature

*/

二、分支语句

1、if__then__else__end;

ex4_6



二、分支语句

2、if__then__elseif.....else__end;

3、inspect

```
inspect number
  when 1 then
    print "not a prime number"
  when 2,5,7,3 then
    print "prime number"
  when 9,4 then
    print "square number"
  else
    print "no special number";
    print "or number greater than 9";
end;
```

ex4_7



三、循环语句

1、from__until

```
from i := 1;  
until i > 10 loop  
    print i;  
    i := i + 1;  
end;
```

2、while__loop

```
n:=6;  
while n>1 loop  
    m:=m*n;  
    n:=n-1;  
end;
```



三、循环语句

3、repeat__until

```
i := 0;  
repeat  
    i := i+1;  
until i > 10 ;
```

4、for__loop

```
for i:=1 to 10 loop  
    i:=i+1;  
next;
```



三、循环语句

5、waituntil__prio

```
waituntil ws.occupied =false prio 1 ;  
    @.move(ws);
```

6、stopuntil__prio

```
stopuntil ws.occupied =false prio 2 ;  
    @.move(ws);
```

两个语句所在的方法将被“挂起”，并赋予权值，1最小。当条件满足，可以激活一个以上被挂起的方法时，先激活权值大的方法。

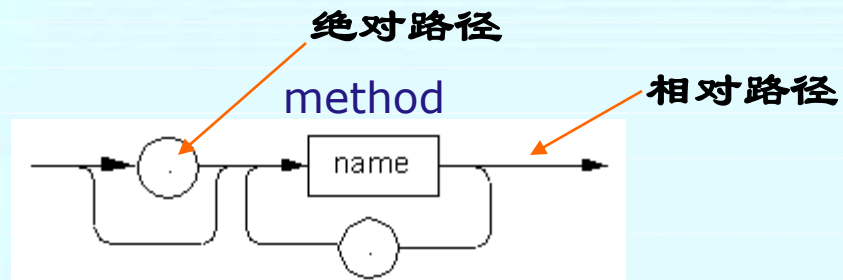
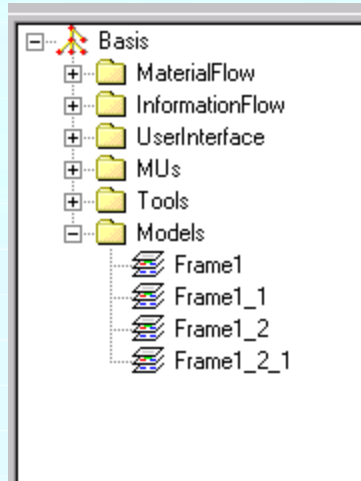
Wait 与 stop的区别：wait当权值高的方法结束后，重新分析上次满足条件由于权值低被“挂起”方法，看它们的条件，是否由于刚结束的方法引起的改变，是否依然满足被激活的条件；stop不重新检查。

ex4_8

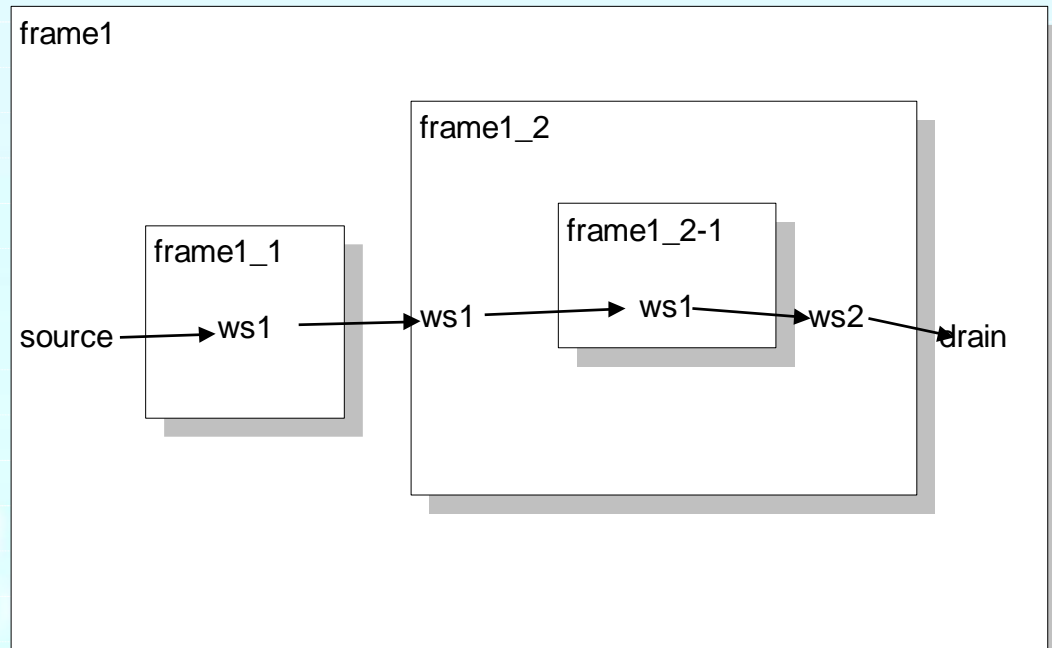


simtalk 对象控制方法

一、路径与定位



.models.frame1.buffer--method 在frame1
current.buffer 或 buffer

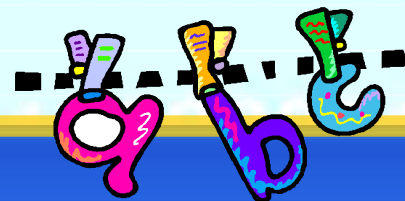


ex4_9

二、匿名标识

- @--表示触发物流对象control的MU
- basis--表示 class library
- current--表示method所在的frame
- ?---表示调用method的实体（物流对象或method）

ex4_10



三、常用物流对象的方法

1、创建mu

```
.mus.entity.create(line)  
.mus.entity.create(store[2,2])
```

2、移动mu

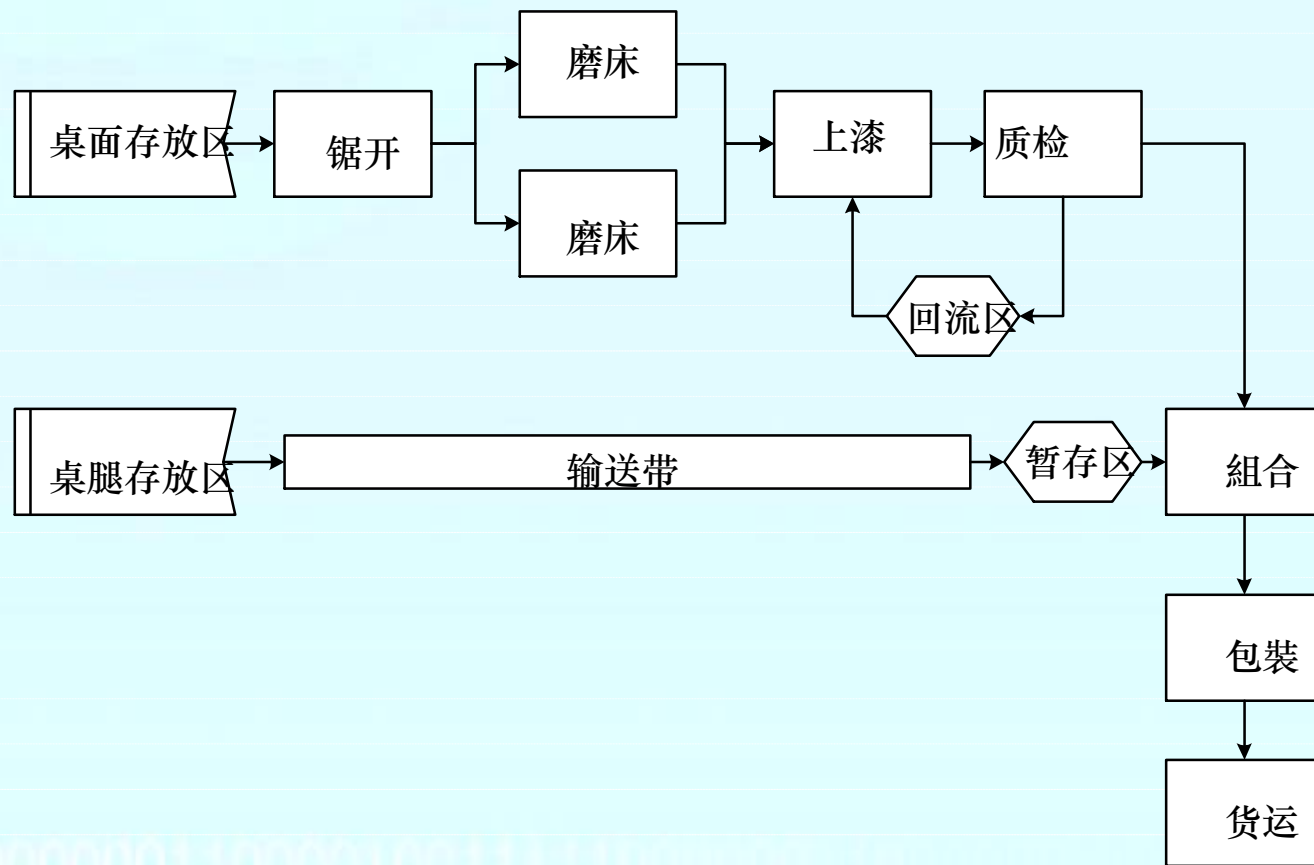
```
@.move;  
@.move(<mu_location>);  
@.move(<integer>);
```

ex4_11

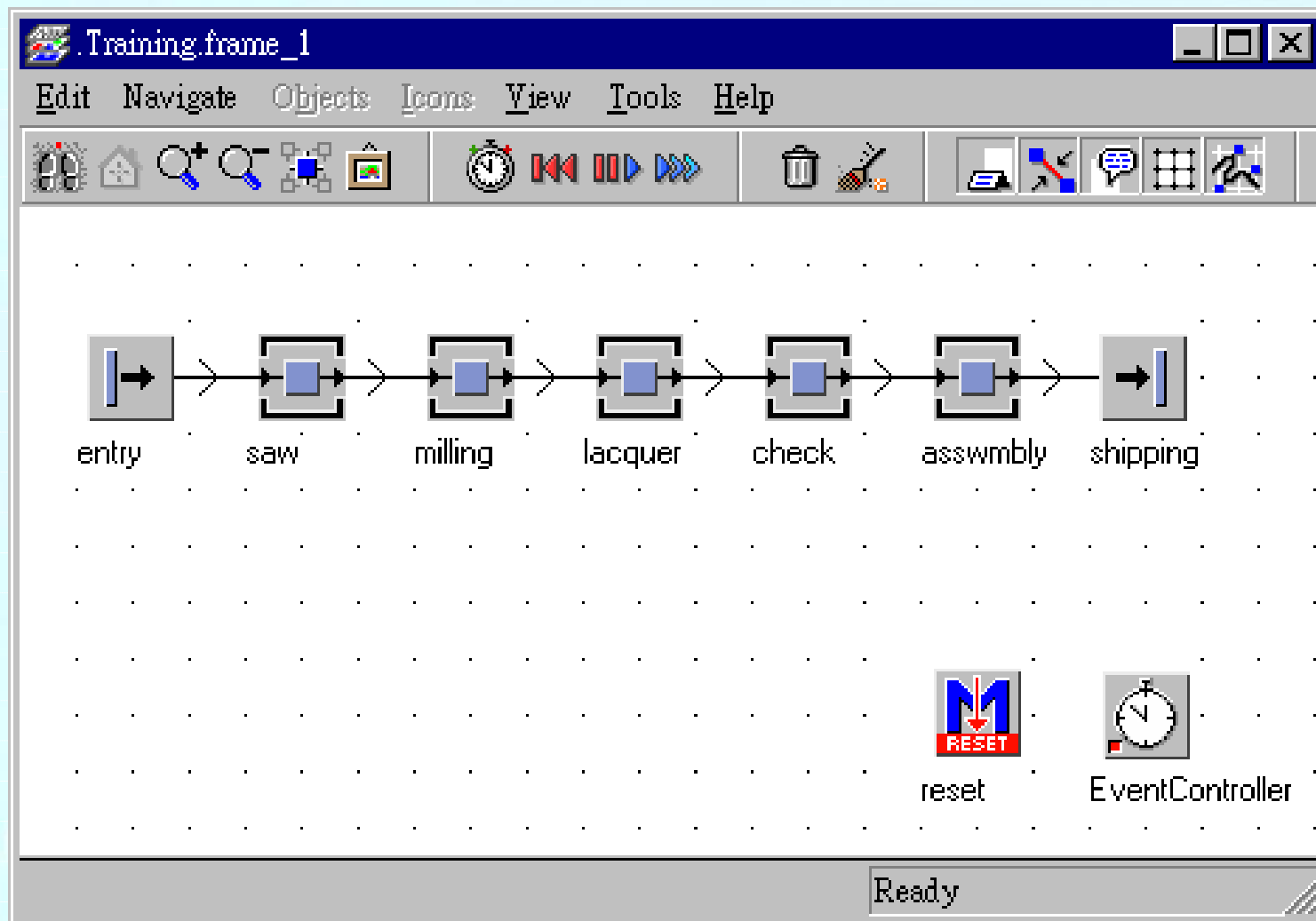


练习

办公桌生产流程图

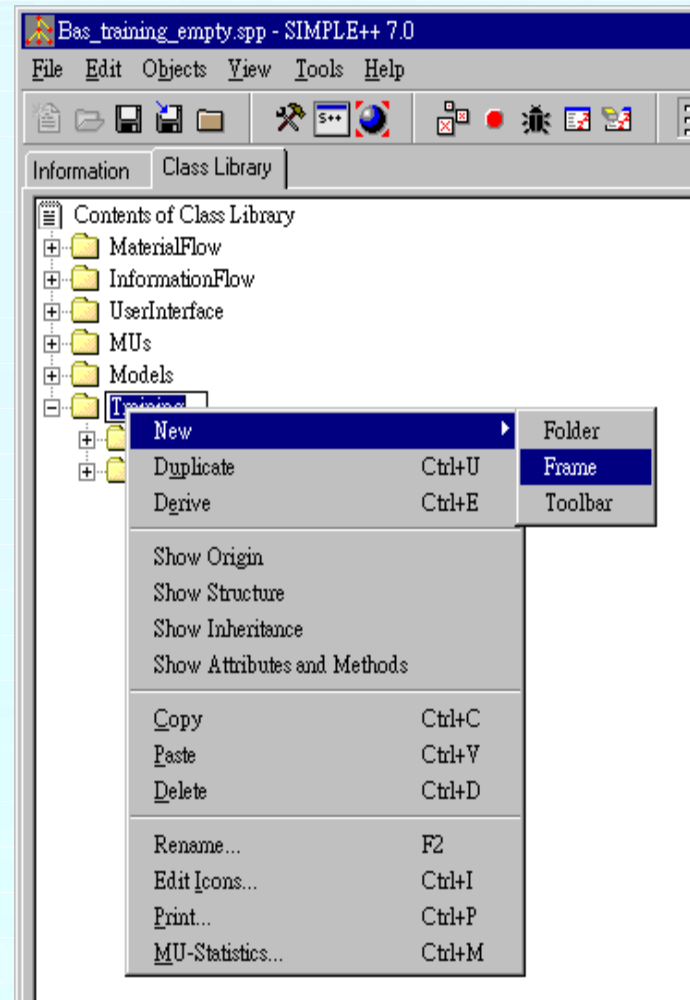


我们将建立以下的模型：



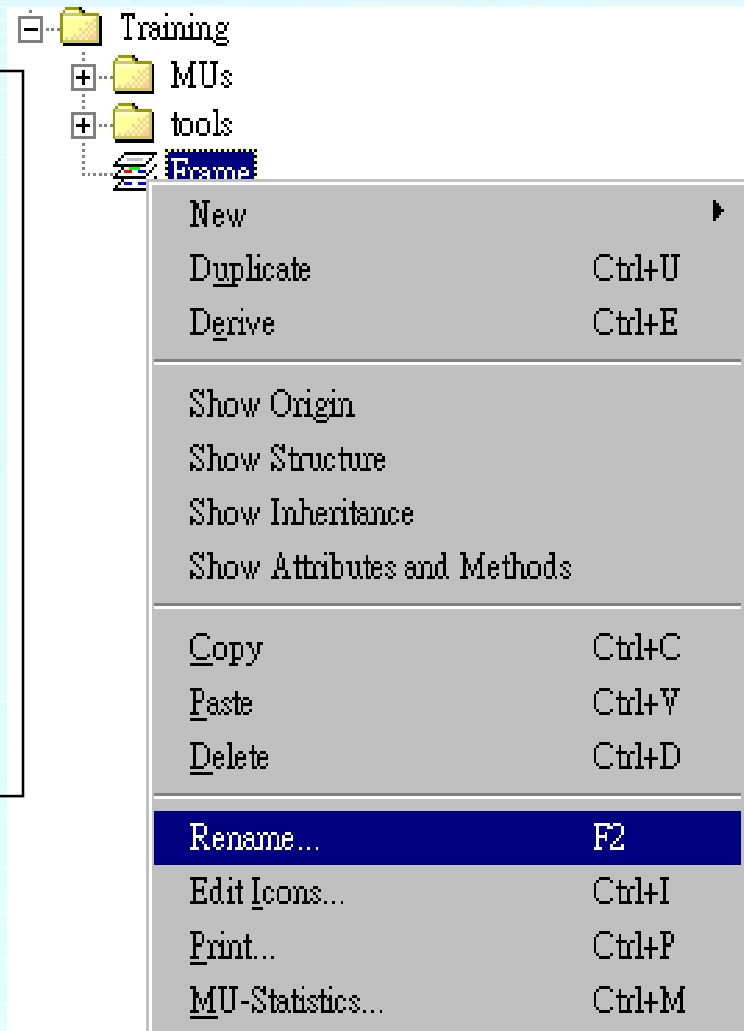
Step 1 :

新增一个 Empty *Frame*



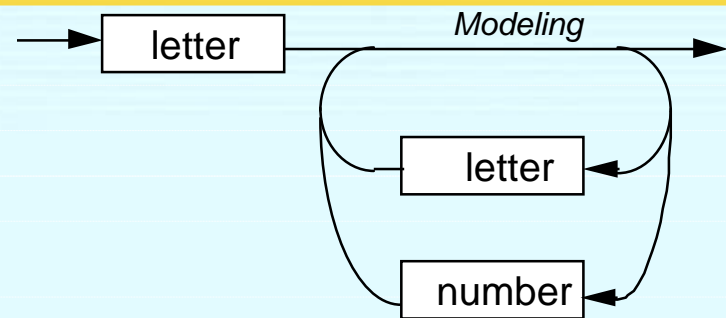
Step 2:

- 在Object Library上更改对象名称
- 可用鼠标选择对象
- 或是按鼠标右键选择“Rename”
- 将Model改名为“Frame_1”

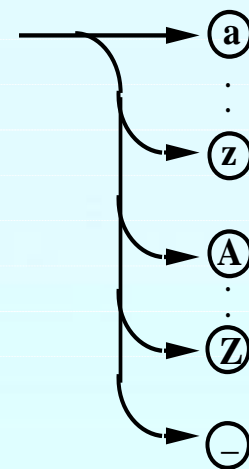


自定对象的名称

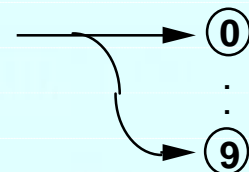
- 任何一个模型你都可以指定其名称
- 使用者自定的名称必须由字母开始紧接着才可以是数字,特殊的文字不被允许. 不可以输入汉字,但可以在label内输入
- 不可以指定其对象名称超过20个字符
- 不可以指定其保留字,如sin, cos, ...if ,then ,else ,end,...等等
- 不能重复指定对象的名称
- 大小写没有区别,如 singleProc = SiNGLEproC



letter:




number:

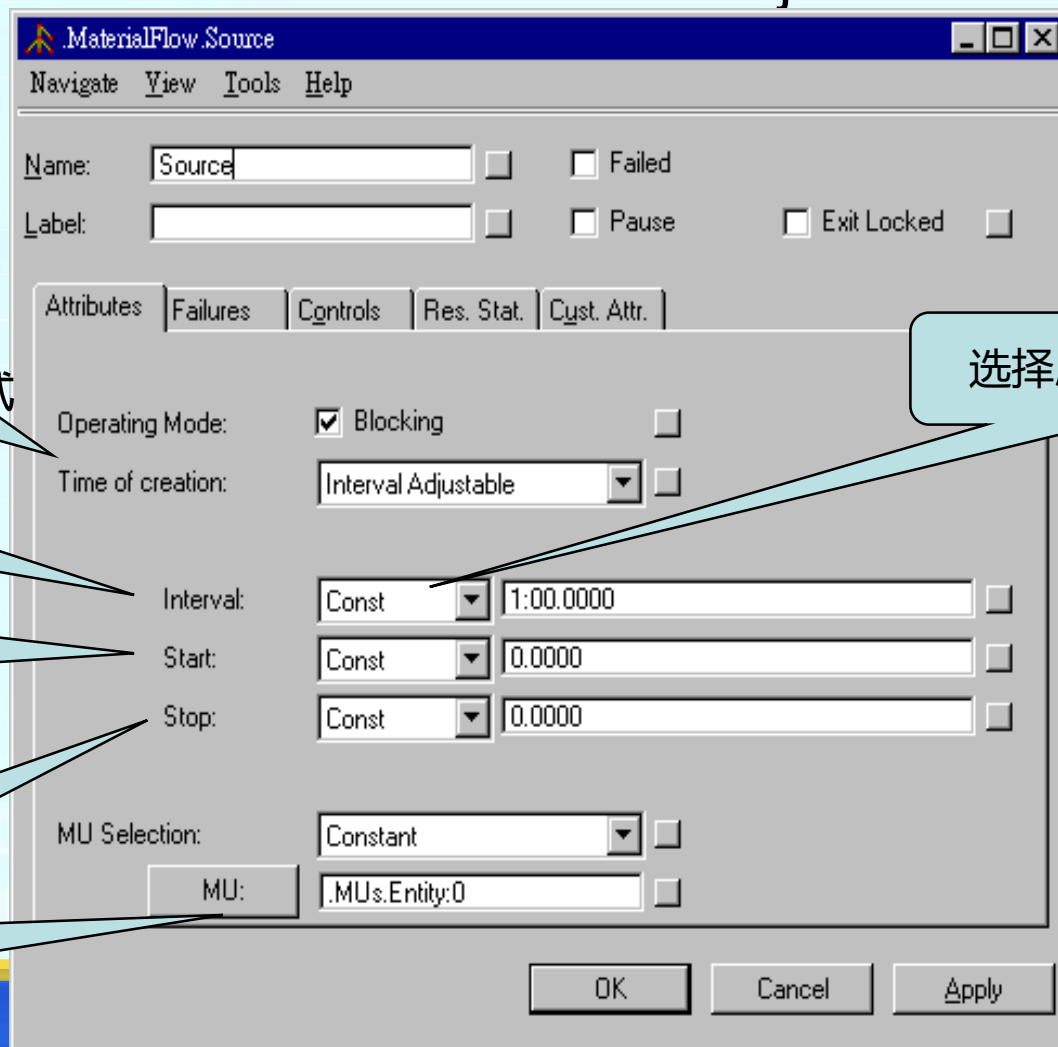


基础对象 Source

Modeling

Features:

- icon: 
- capacity: 1
- active material flow basic object



MaterialFlow.Source

Navigate View Tools Help

Name: Source ☐ Failed

Label: ☐ Pause ☐ Exit Locked ☐

Attributes Failures Controls Res. Stat. Cust. Attr.

Operating Mode: ☒ Blocking ☐

Time of creation: Interval Adjustable ☐

Interval: Const ☐ 1:00.0000 ☐

Start: Const ☐ 0.0000 ☐

Stop: Const ☐ 0.0000 ☐

MU Selection: Constant ☐

MU: ☐ .MUs.Entity:0 ☐

OK Cancel Apply

移动对象产生的方式

间隔时间

开始时间

停止时间


MU的选择

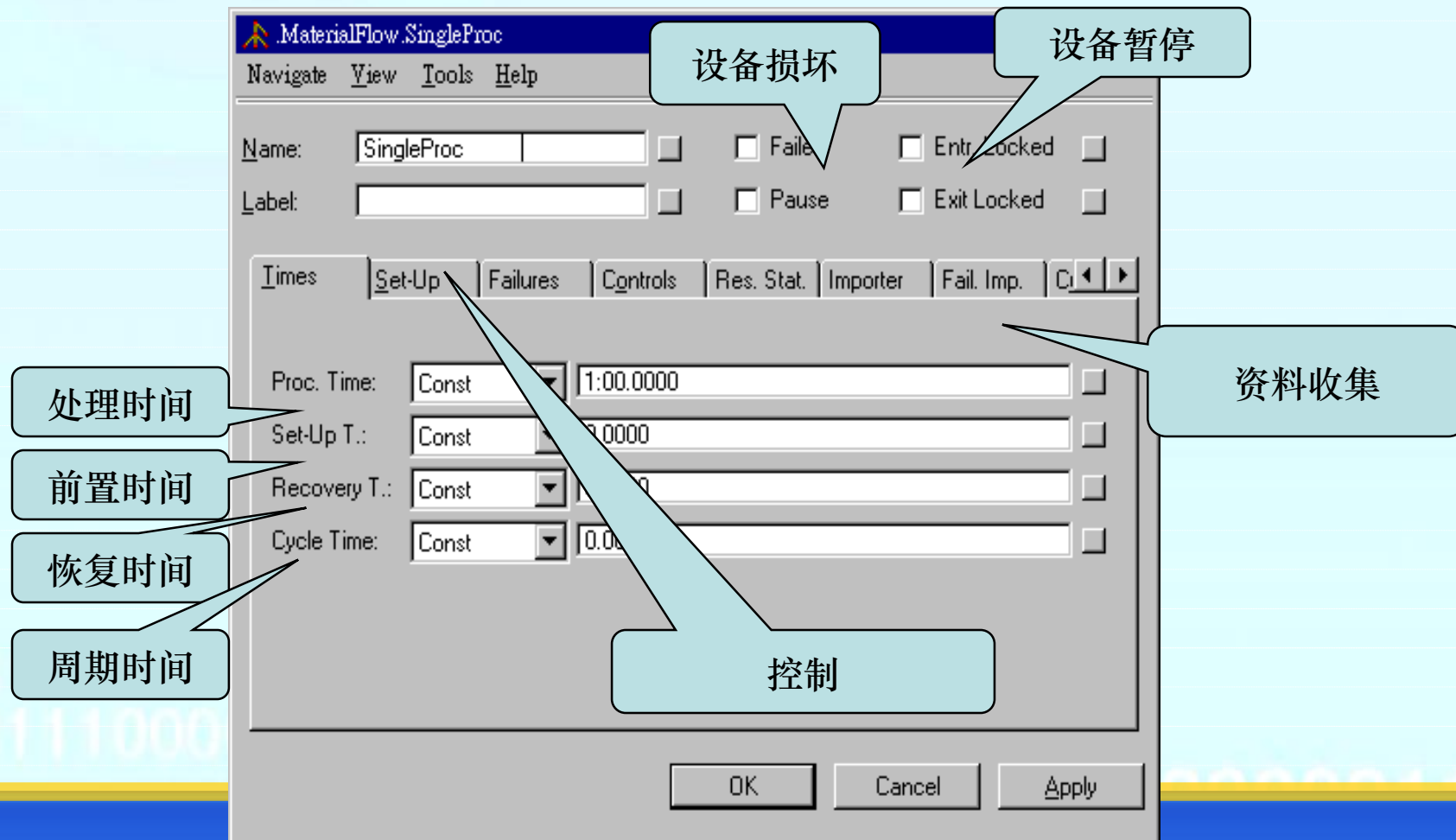
选择属于什么概率

基础对象 *SingleProc*

Modeling

Features:

- : capacity: 1
- active material flow basic object



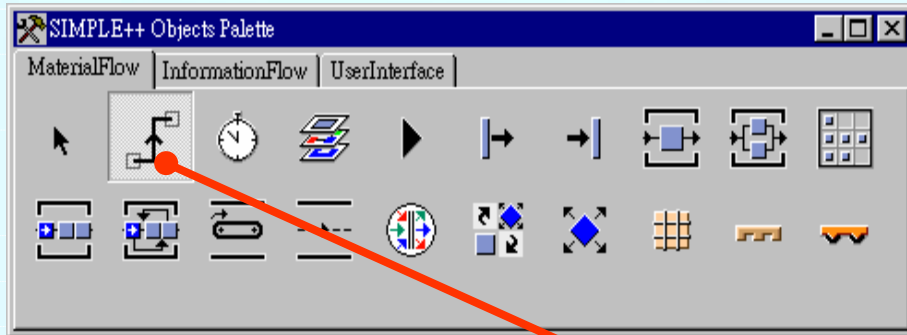
插入对象

- 点选对象库中的对象
- 移动鼠标放到准备放该对象的frame上
- 按鼠标左键
- **连续插入对象模式**
- 点选对象库中的对象后，按Ctrl键，此时可连续插入对象。
- 要放弃时放开Ctrl键即可。

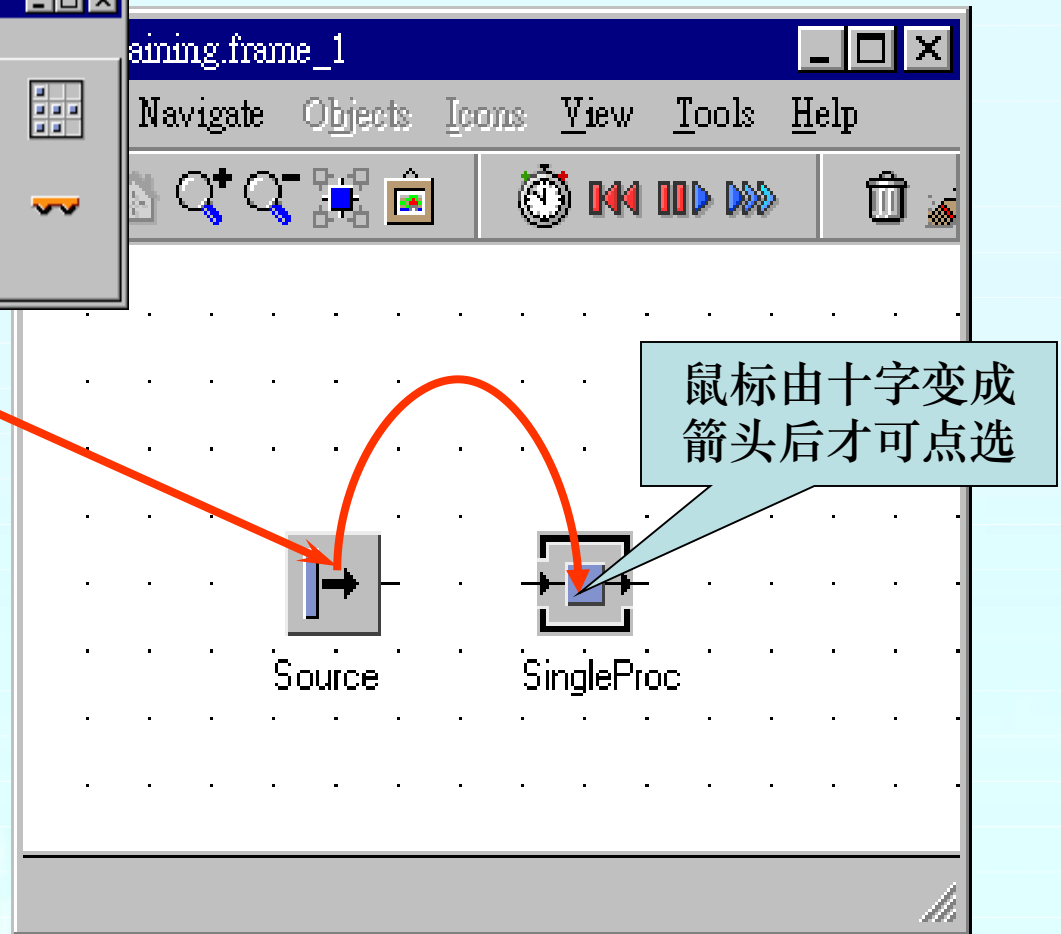
手动连接对象

Modeling

source object → destination object

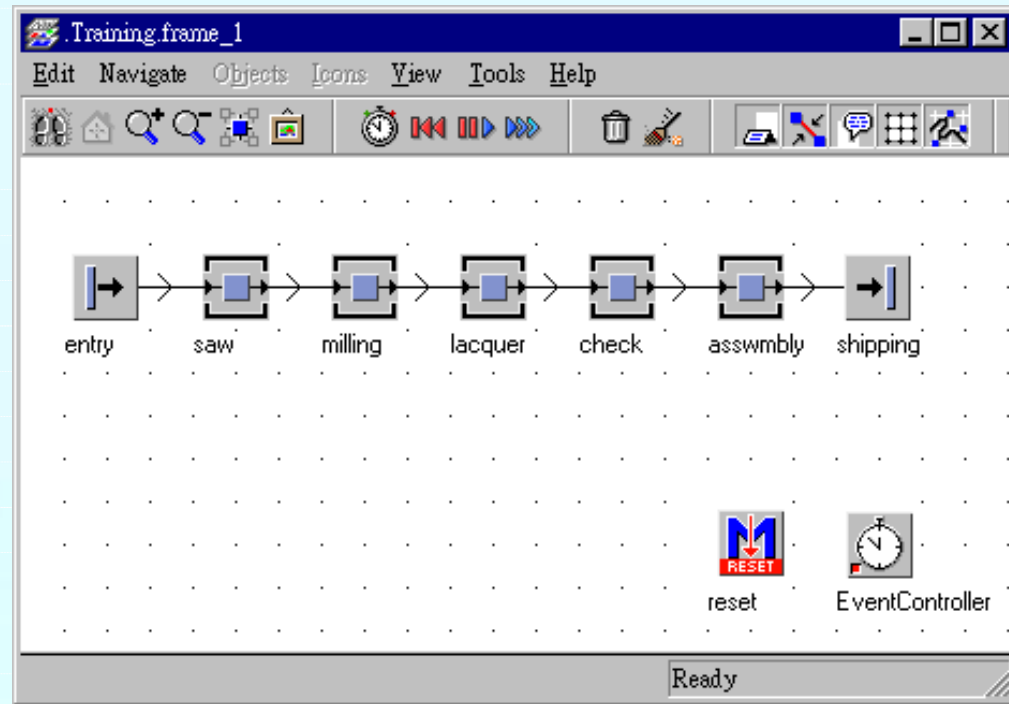


- 先点选Connect对象
- 再点选 Source 对象
- 然后再点选目标对象
- 完成一个连线
- 按住Ctrl可连续点选

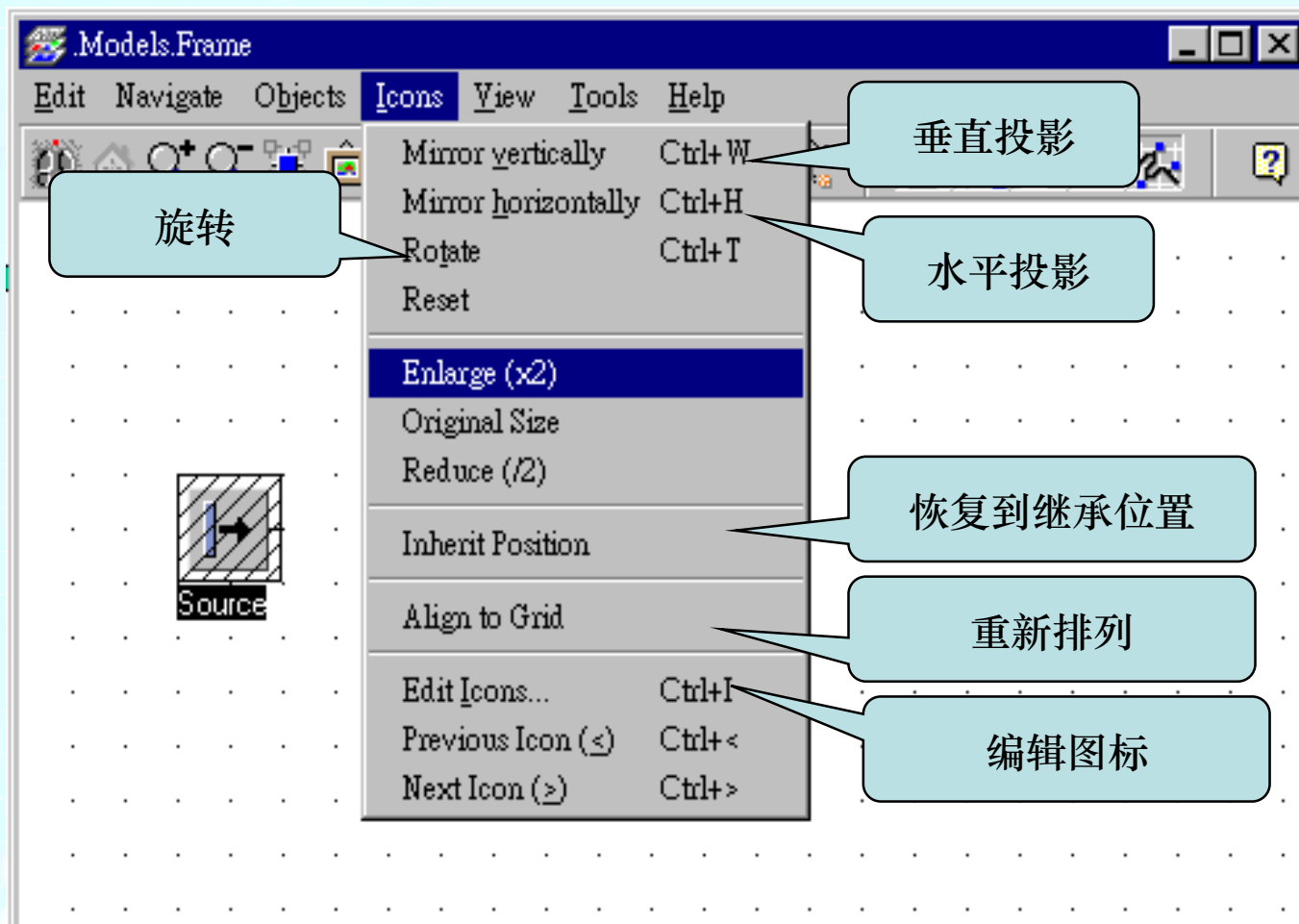


构建一个简单的模型 (Exercise 1)

- 更改这个 *Frame* 名称为 “frame_1”
- 插入一个 Source, 五个 SingleProcs, 一个 Drain, 一个 EventController 和 Method, 并依照图上修改对象名称
- 利用 *Connector* 对象将每个对象连接起来
- 双击 milling 设备, 打开对话框, 设定其 processing time (处理时间) 为 8 分钟
- 双击 *EventController*, 打开对话框, 按下 start 按钮, 启动仿真.

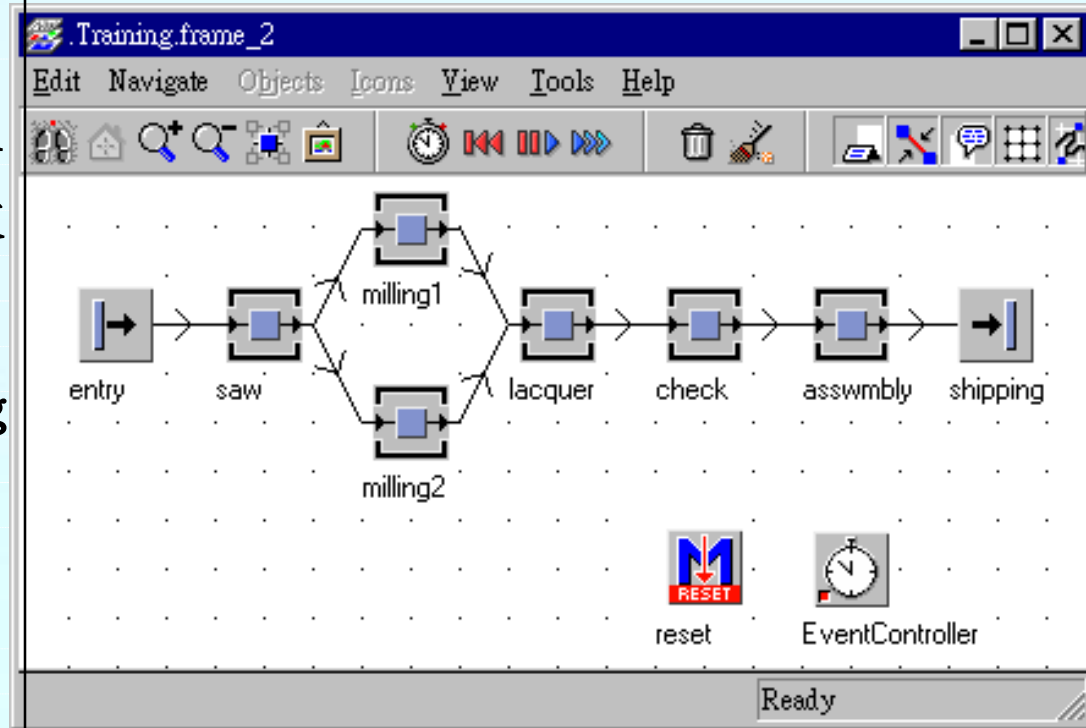


图标操作的技巧




Exercise 2: 物料分流与合并

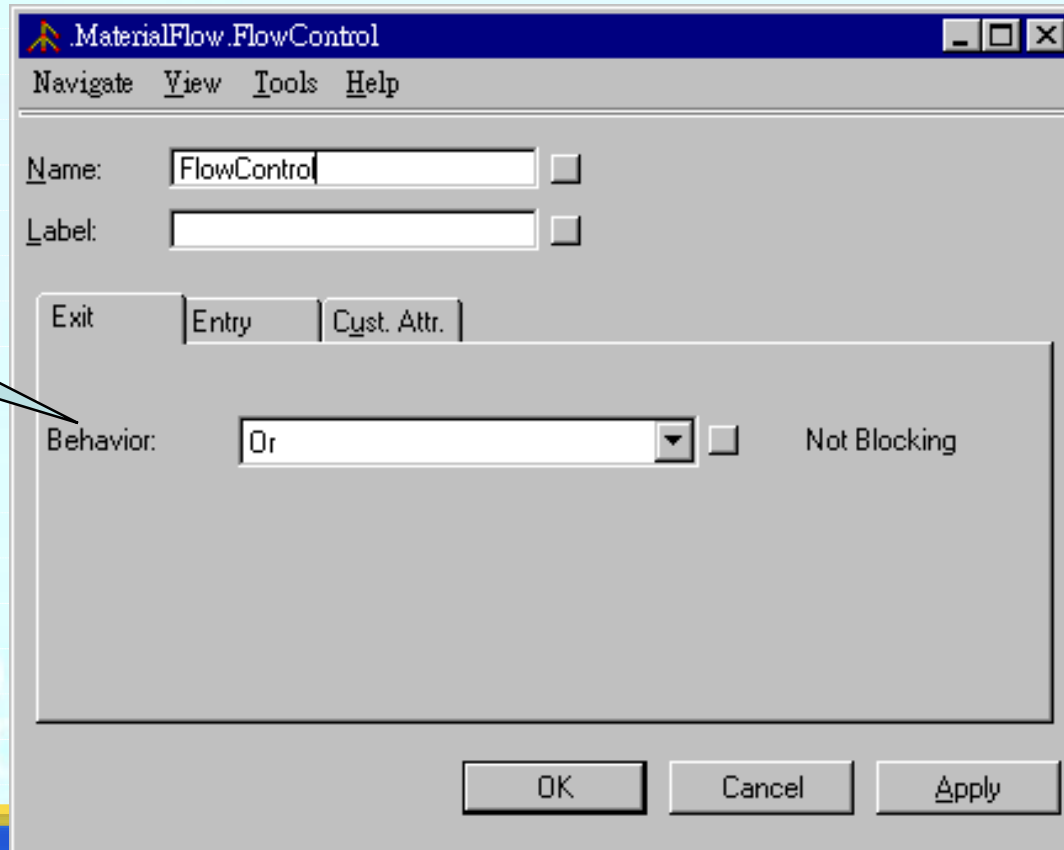
- 将先前的 frame_1 复制, 取另一个名称 “frame_2”
- 再多加一个 *SingleProc* 到这个 frame 上并连接, 依据图上修改名称
- 将 “milling1” 的 processing time 修改为8分钟
- 将 “milling2” 的 processing time 修改为4分钟
- 启动仿真模型, 观察分流情况



FlowControl的用法

Features:

- icon: 
- capacity: 0
- information flow basic object



MaterialFlow.FlowControl

Navigate View Tools Help

Name: FlowControl ☐

Label: ☐

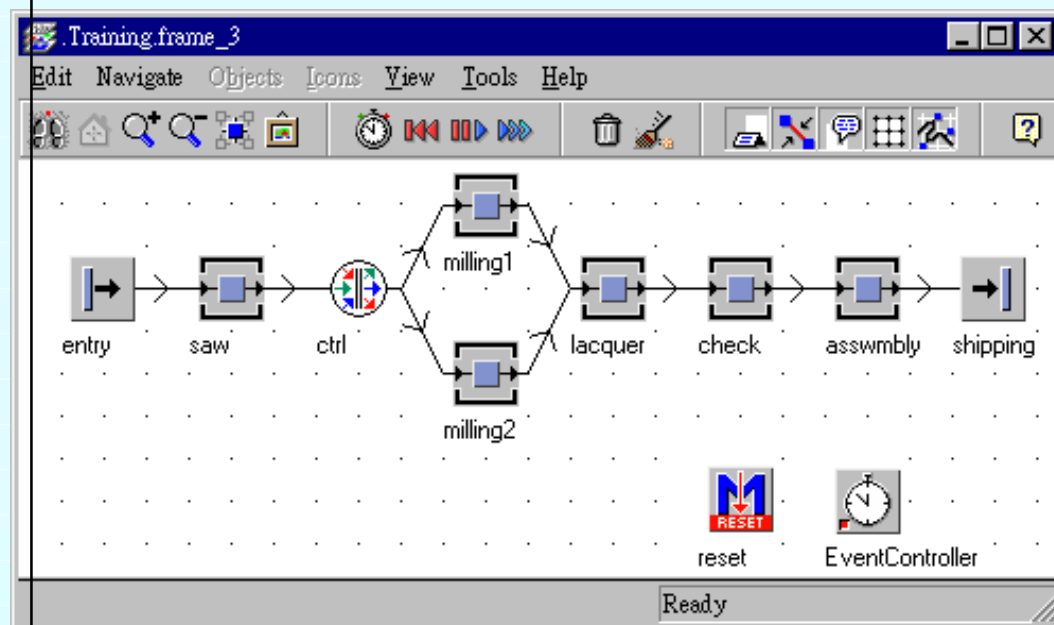
Exit Entry Cust. Attr.

Behavior: Or ☐ Not Blocking

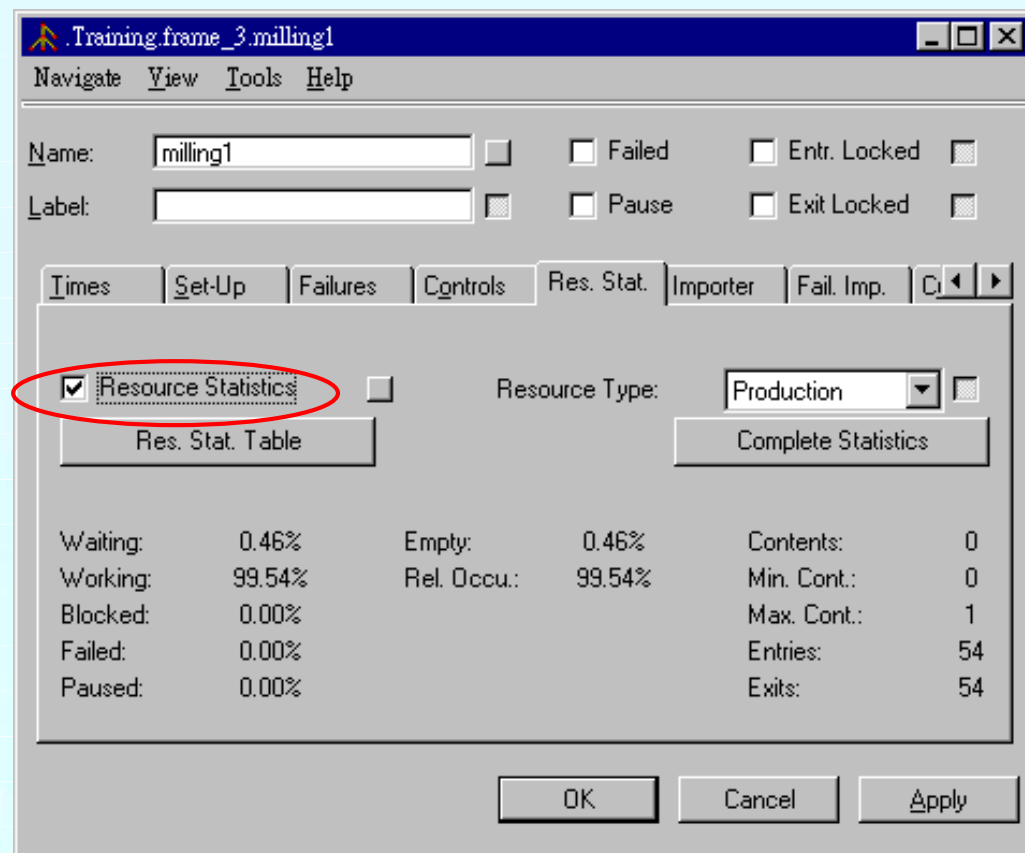
OK Cancel Apply

分流处理法则

- 复制 frame_2, 改名为“frame_3”。
- 插入一个 *FlowControl* 对象, 并修改名称。
- 打开 *FlowControl* 并点选“Percentage”并点选“block”
- 点 “Open” 按钮, 出现表格之后, 在第一栏输入10第二栏输入90
- 启动仿真, 观察MU的分流状况



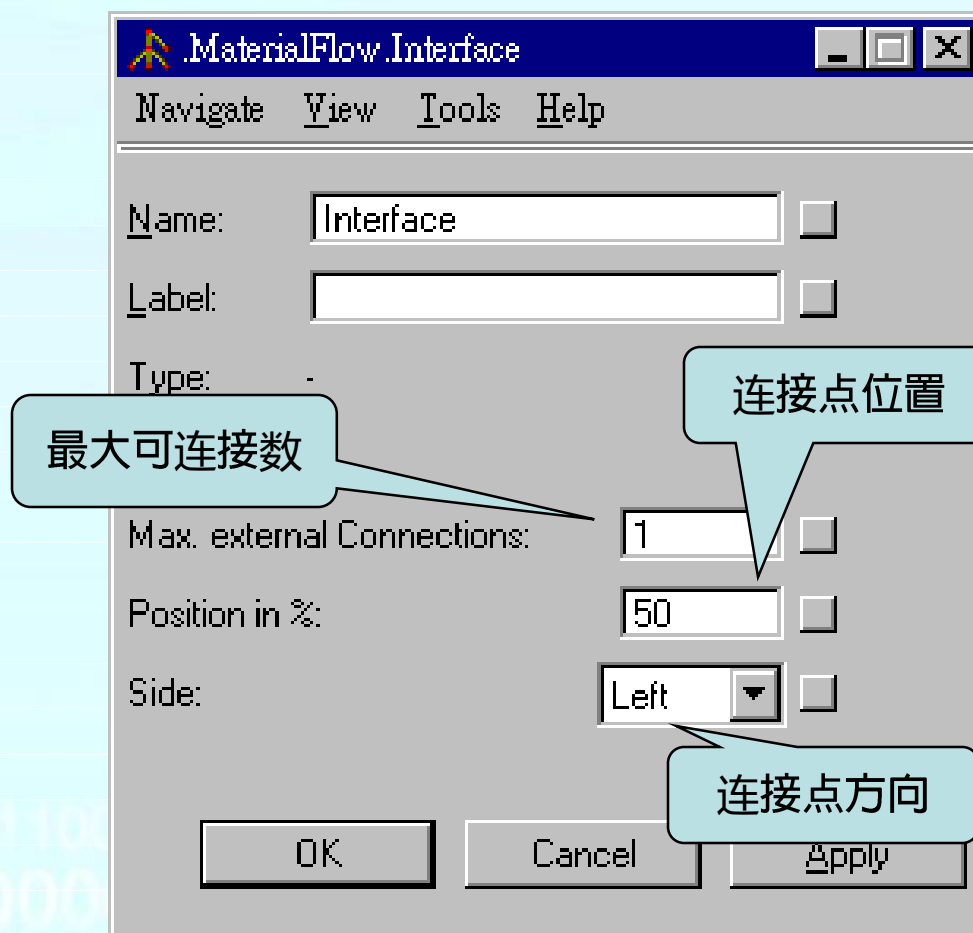
- 打开 “milling1” 和 “milling2”
- 将 “Res. Stat.” 页, Resource Statistics启动, 并按Apply。
- 启动仿真模型
- 点取 “Res.-Stat.-Table” 按钮,观察统计数据



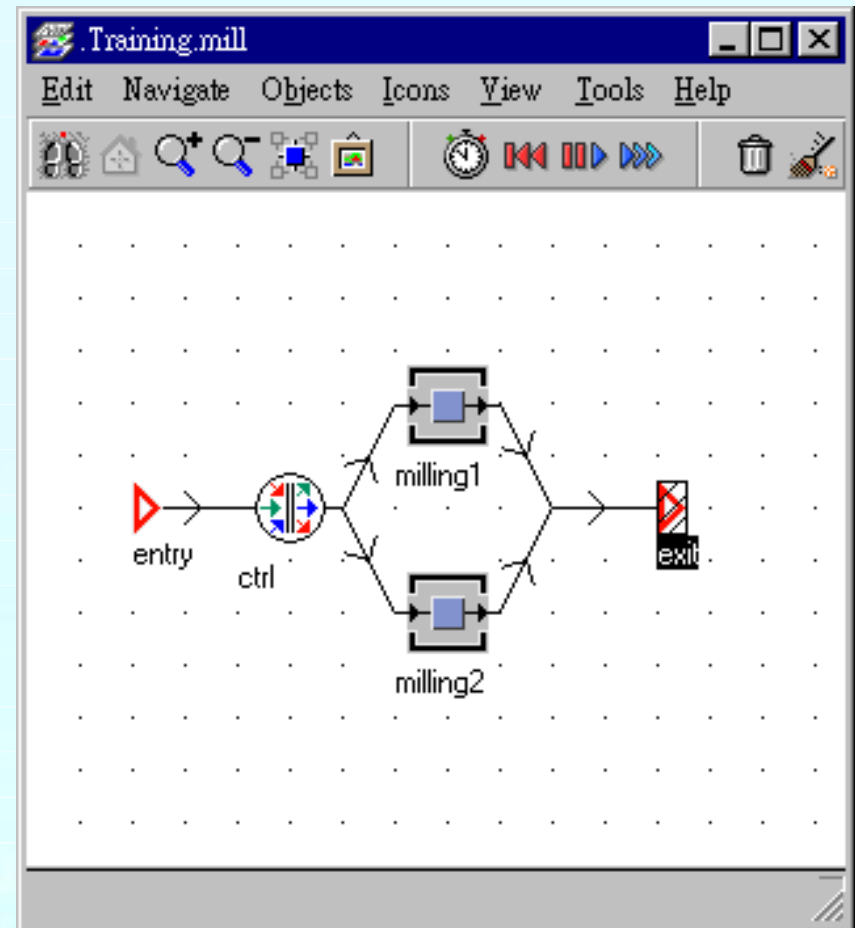
Interface的用法

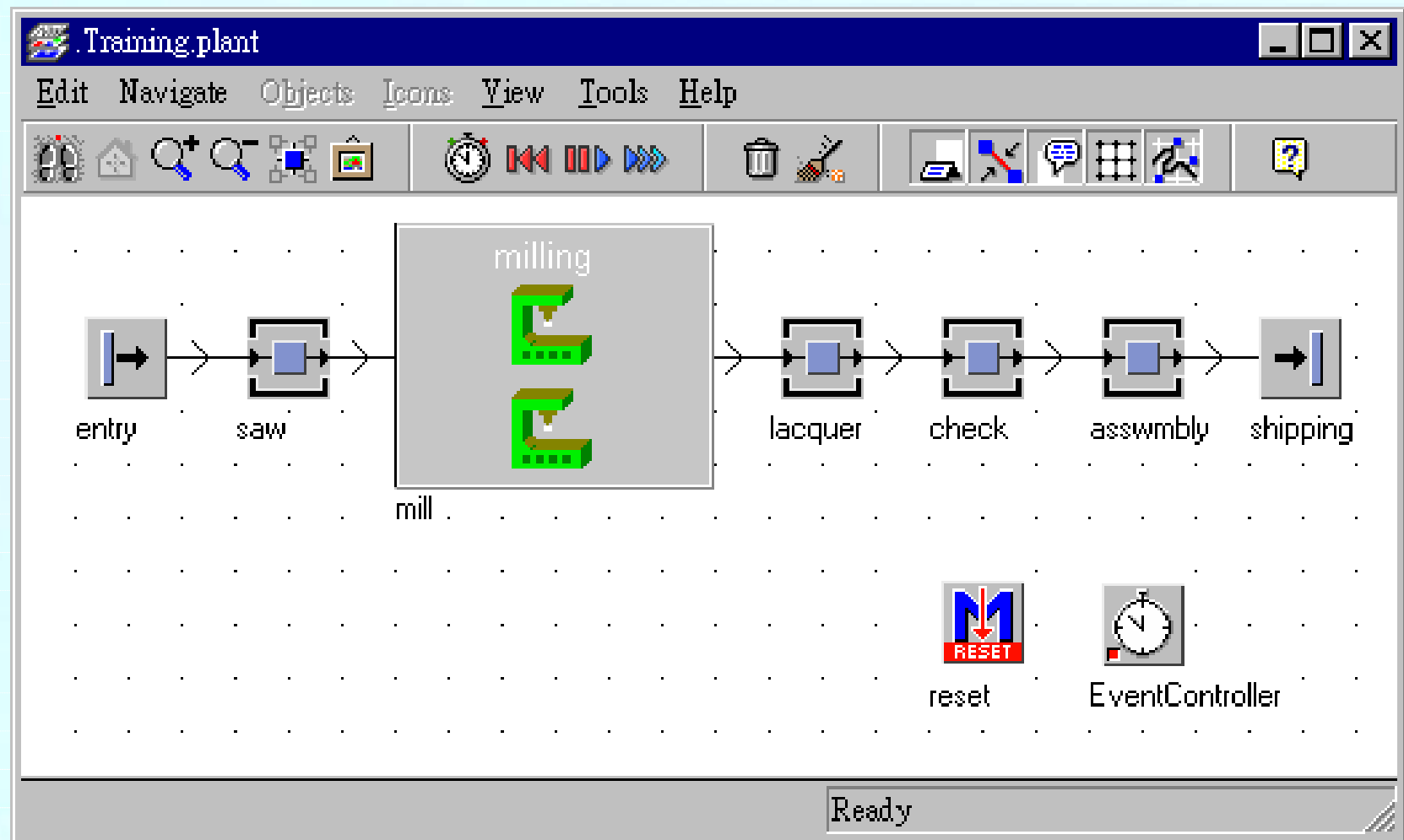
Icon: 

Interface 是Frame与其他对象之间的桥梁



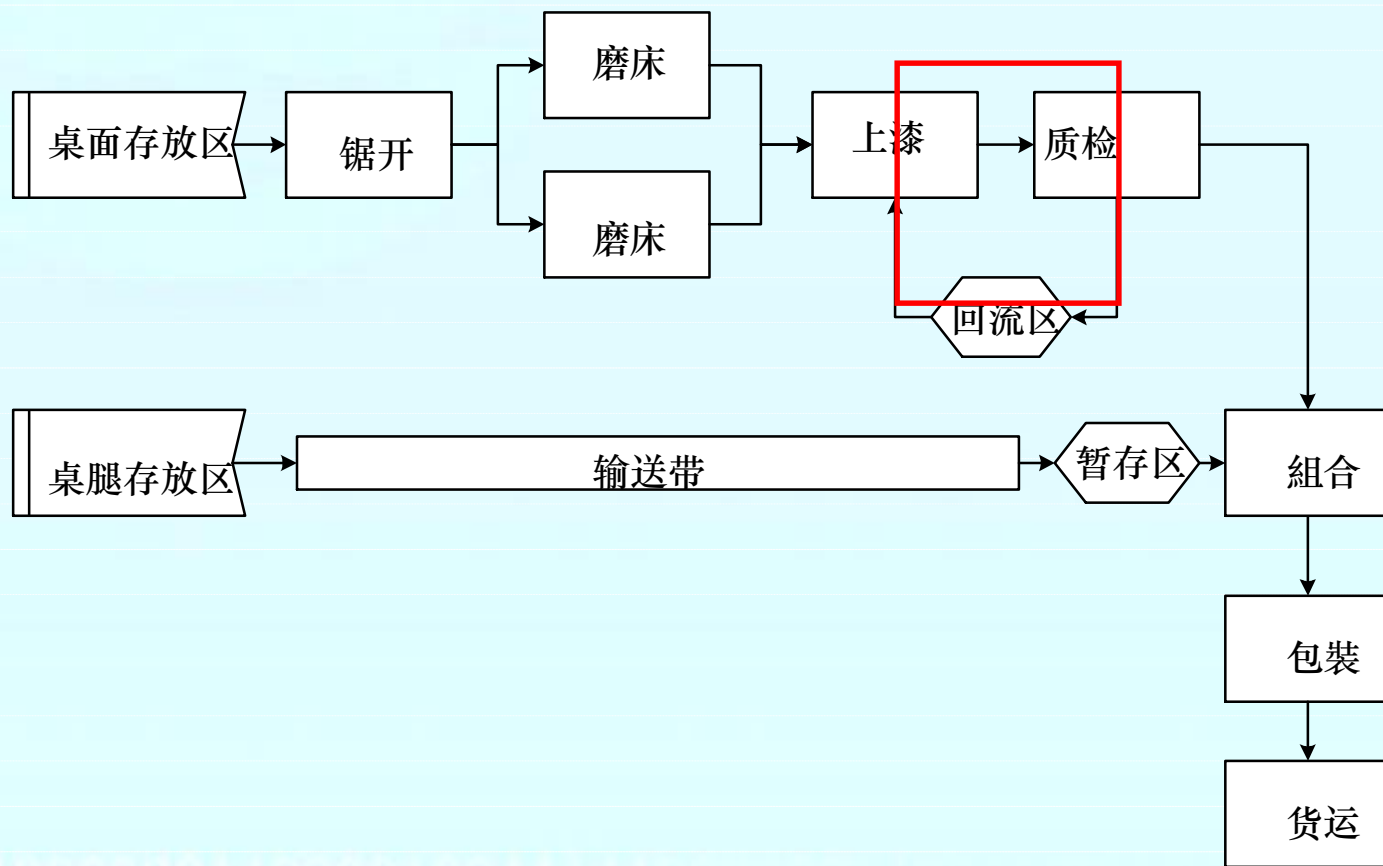
- 打开一个新的 *Frame*
- 将其名字改为 “mill”
- 将control1依照百分比法, 10%到milling1, 90%到milling2





练习

办公桌生产流程图



用户自定义属性

Customized Attributes

- ❑ 可让MU携带属性
- ❑ 可设定无限多的 customized attributes
- ❑ Customized attribute 是由 name、data type与value 构成

创建Customized Attribute

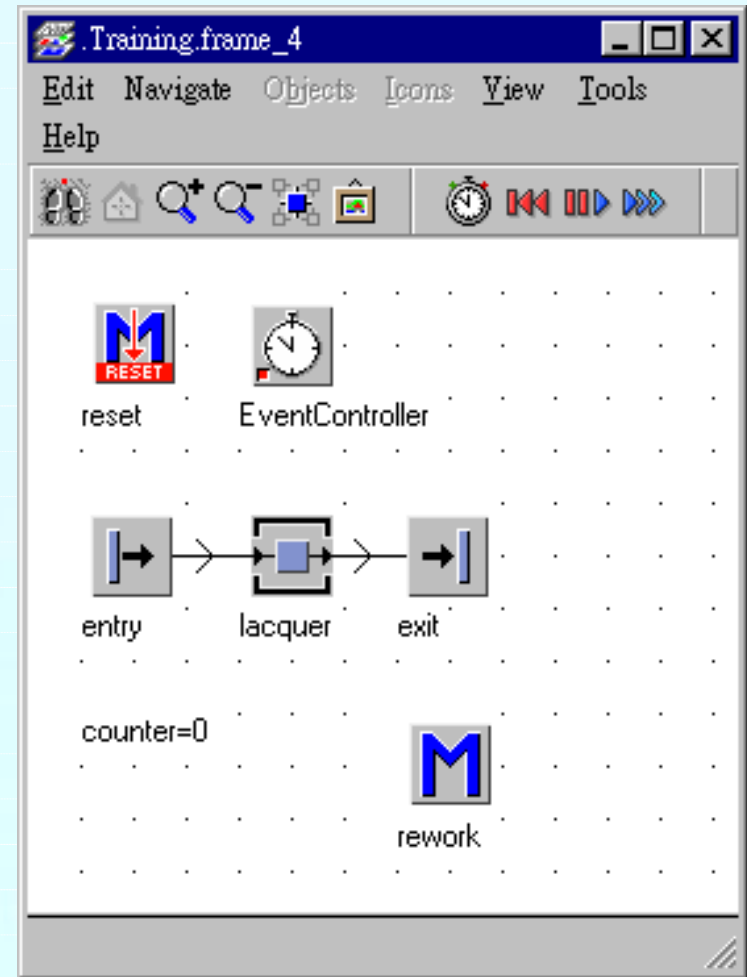
- 双击 “entity” 对象.
- 单击 “Cust. Attr.” 属性页 .
- 点击 “Insert” 按钮.
- 按下图更新属性

The screenshot shows a Windows-style dialog box titled "Cust. Attributes". It has a standard title bar with minimize, maximize, and close buttons. The main area contains the following fields and controls:

- Name:** A text box containing "quality" with an unchecked checkbox to its right.
- Value** and **Communication** tabs. The "Value" tab is currently selected.
- Data Type:** A dropdown menu showing "string" with an unchecked checkbox to its right.
- Value:** A text box containing "GOOD" with an unchecked checkbox to its right.
- Buttons:** "OK", "Cancel", and "Apply" buttons are located at the bottom right of the dialog.

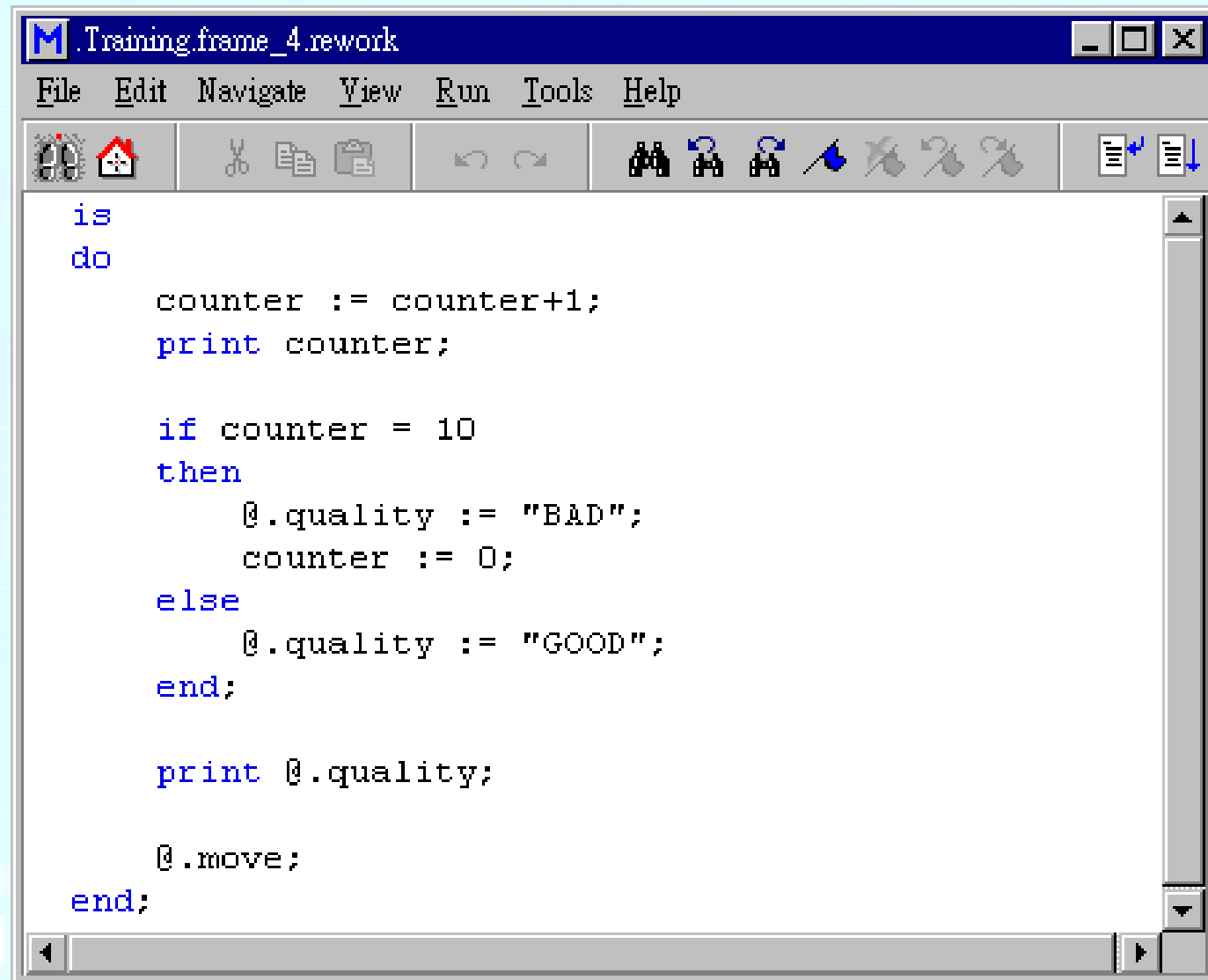
Linking Material and Information Flow

- 创建一个新的frame并命名为 frame_4.
- 如图所示在frame中建立仿真模型。
- 加入variable 和method对象
- 把method对象作为“lacquer”对象exit的控制方法。
- 把variable对象命名为“counter”。



Rework的代码

Linking Material and Information Flow



The screenshot shows a software window titled ".Training.frame_4.rework". The window has a menu bar with "File", "Edit", "Navigate", "View", "Run", "Tools", and "Help". Below the menu bar is a toolbar with various icons for file operations, navigation, and editing. The main area is a code editor with the following code:

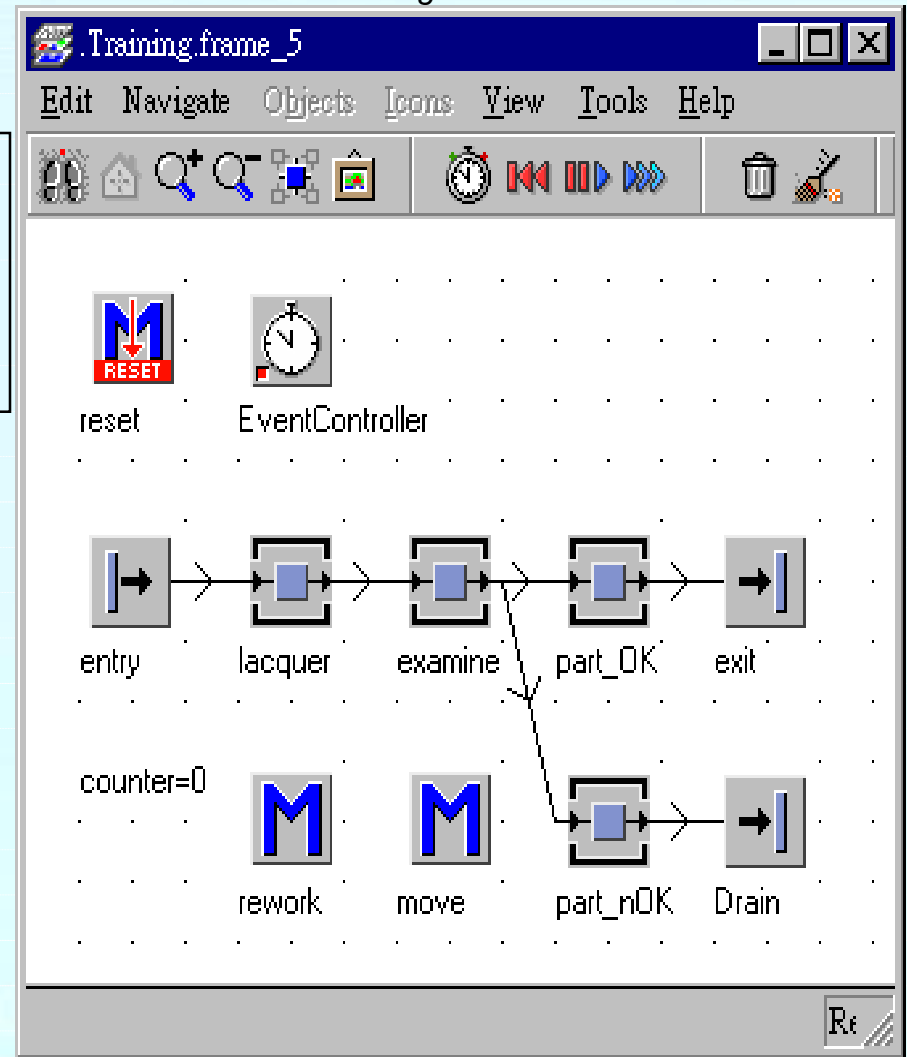
```
is
do
    counter := counter+1;
    print counter;

    if counter = 10
    then
        @.quality := "BAD";
        counter := 0;
    else
        @.quality := "GOOD";
    end;

    print @.quality;

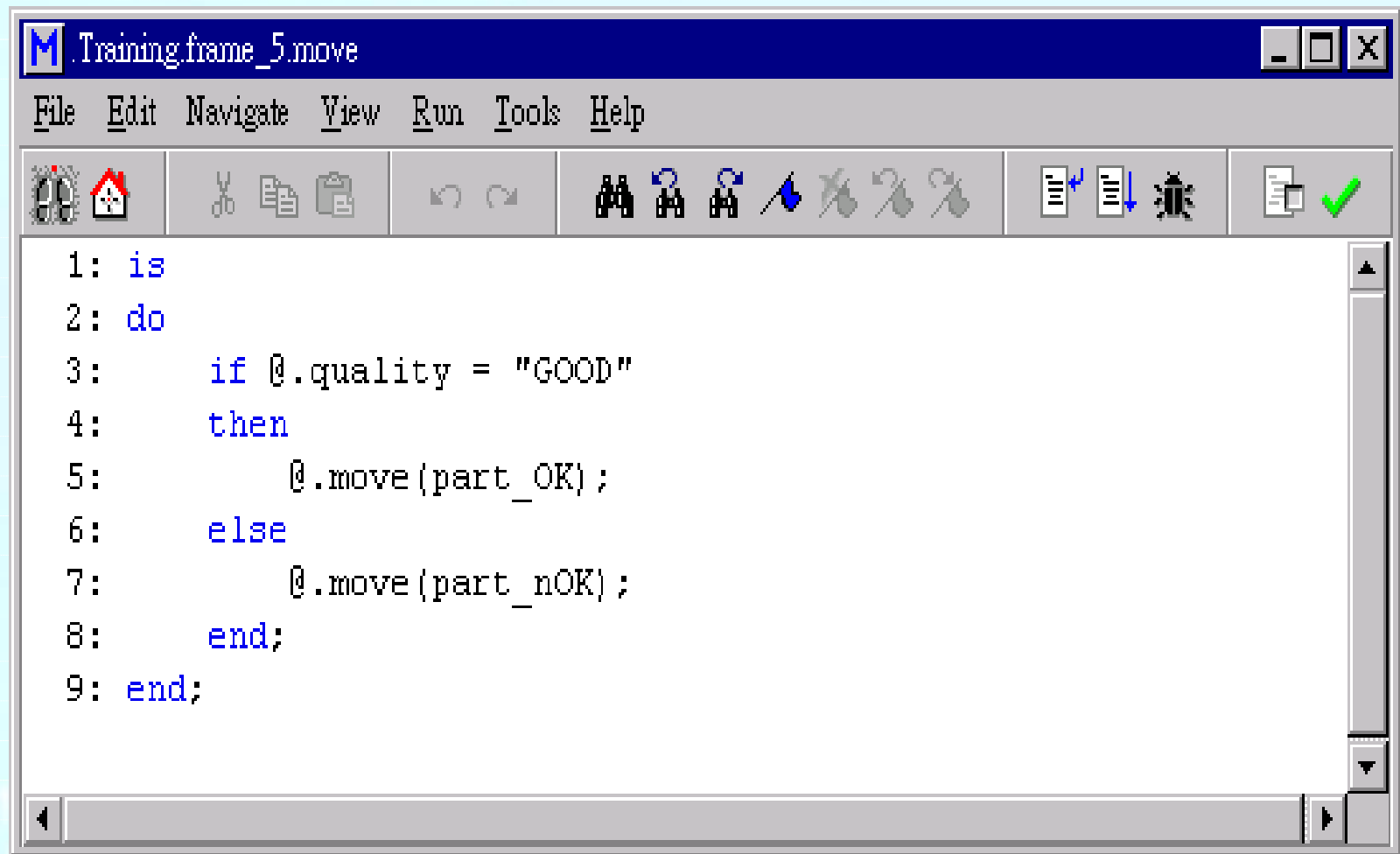
    @.move;
end;
```

- 复制 frame_4 并命名为 frame_5.
- 如图所示建立仿真模型.
- 将move方法和“examine”对象的exit连接。



Move的代码

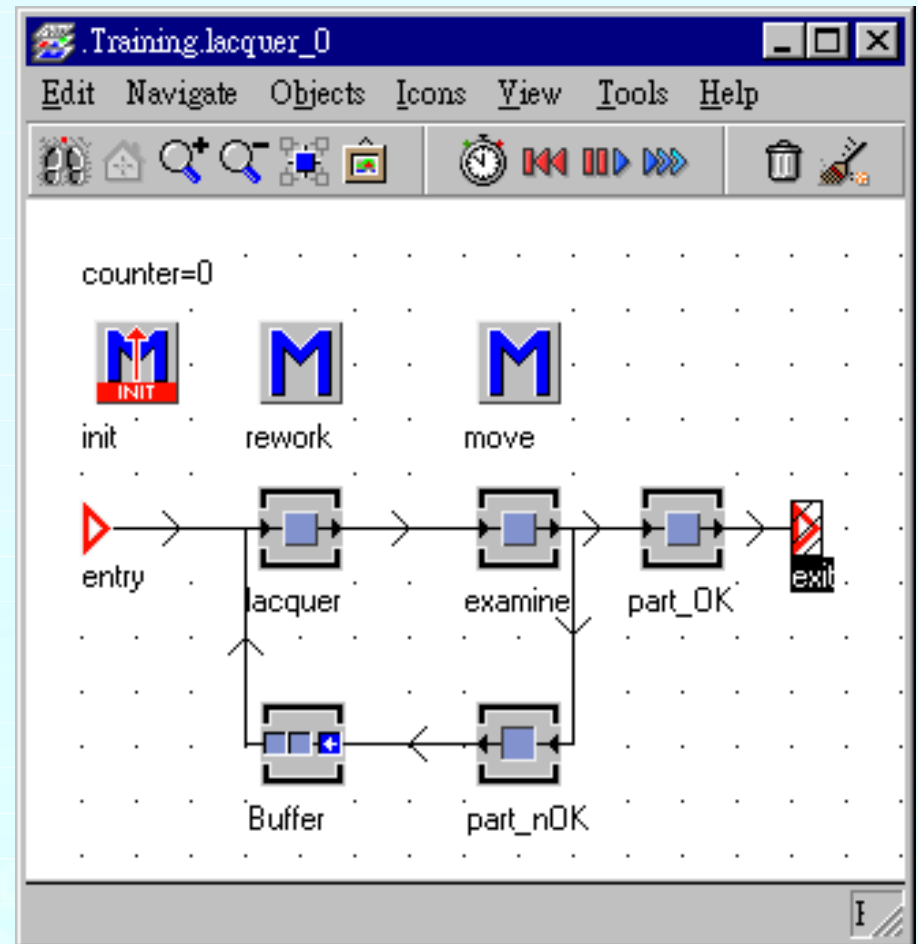
Linking Material and Information Flow



The screenshot shows a code editor window with a menu bar (File, Edit, Navigate, View, Run, Tools, Help) and a toolbar with various icons. The code is as follows:

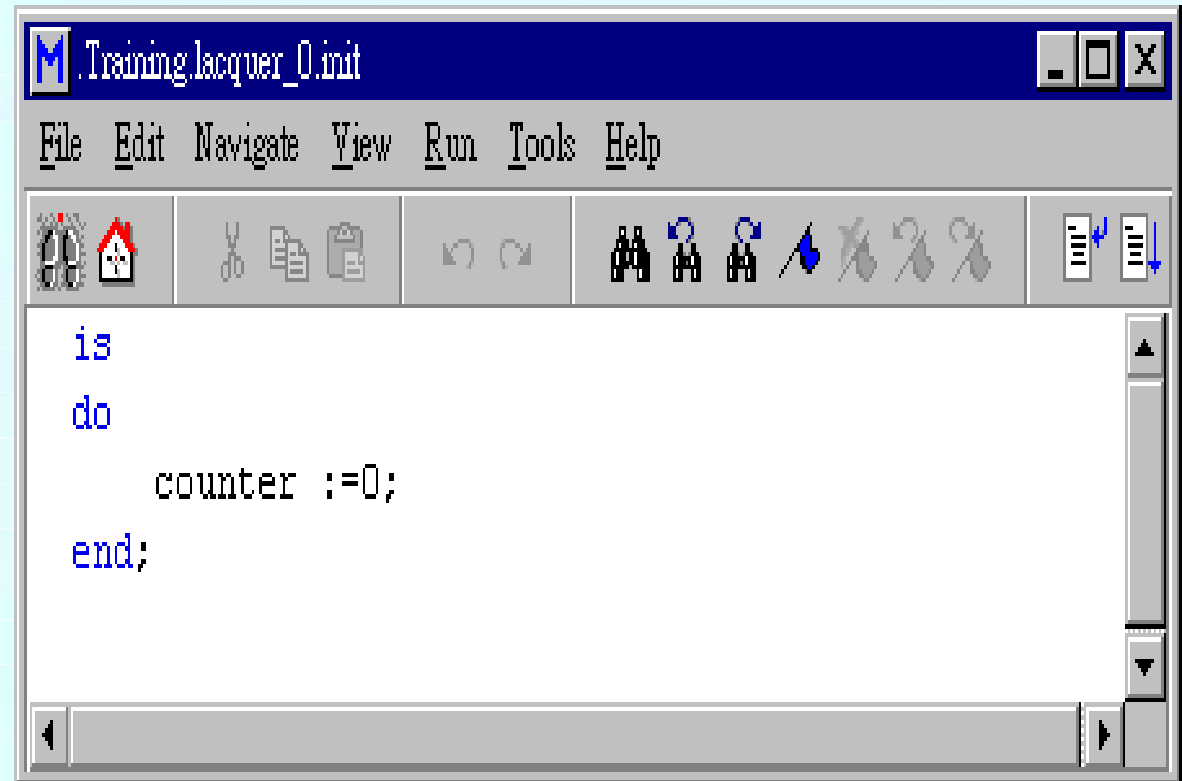
```
1: is
2: do
3:     if @.quality = "GOOD"
4:     then
5:         @.move(part_OK);
6:     else
7:         @.move(part_nOK);
8:     end;
9: end;
```

- 如图所示构建仿真模型
- 将”rework”和”move”方法的代码补充完整
- 将该frame和原模型通过entrance对象连接起来。



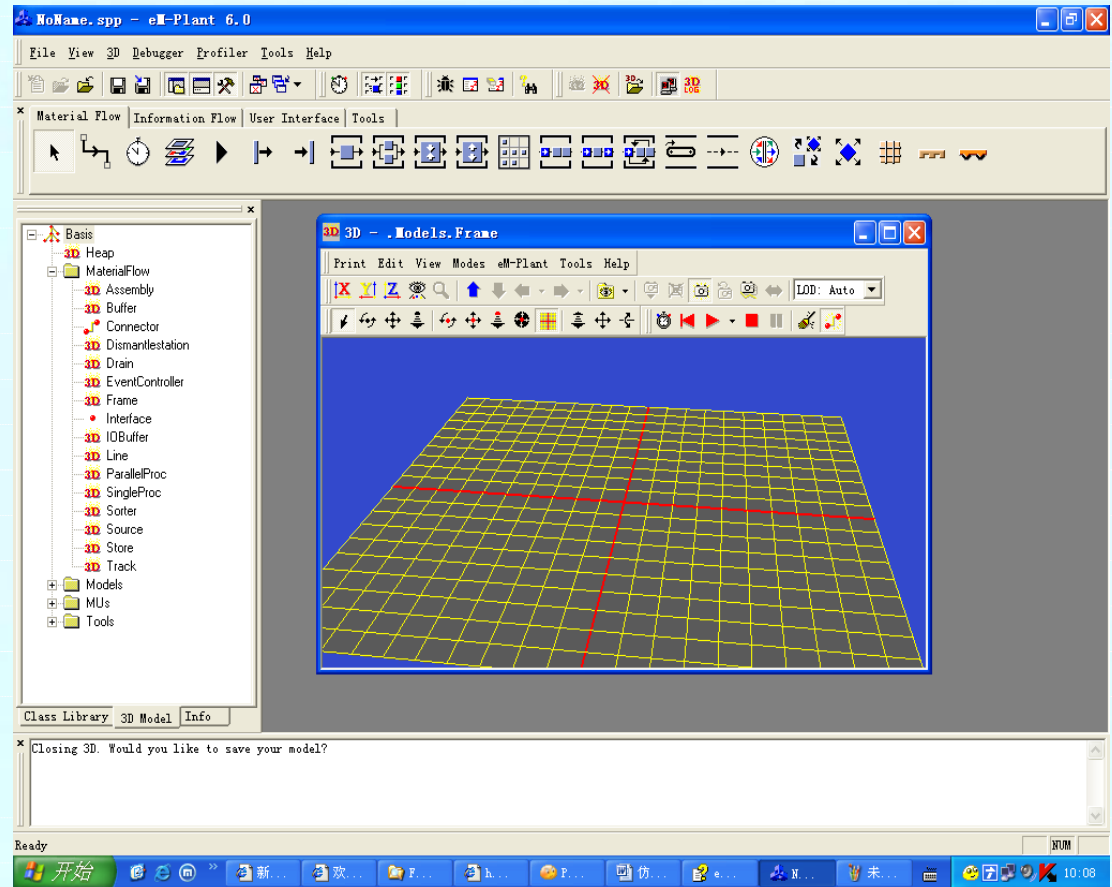
“init”方法

- “init”方法在每次仿真开始时会自动调用。
- 在这里每次仿真开始时需要将”counter”置为0，因此添加一个”init”方法，并设为0。

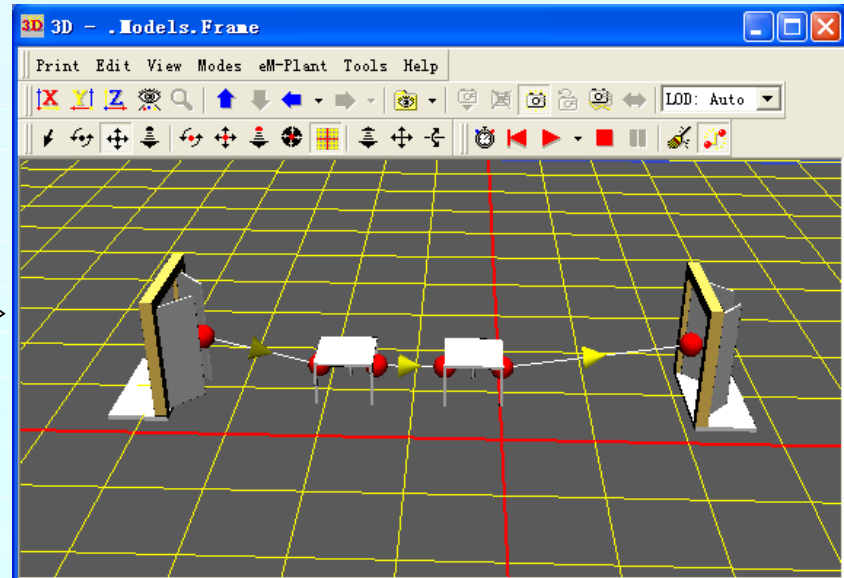
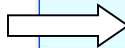
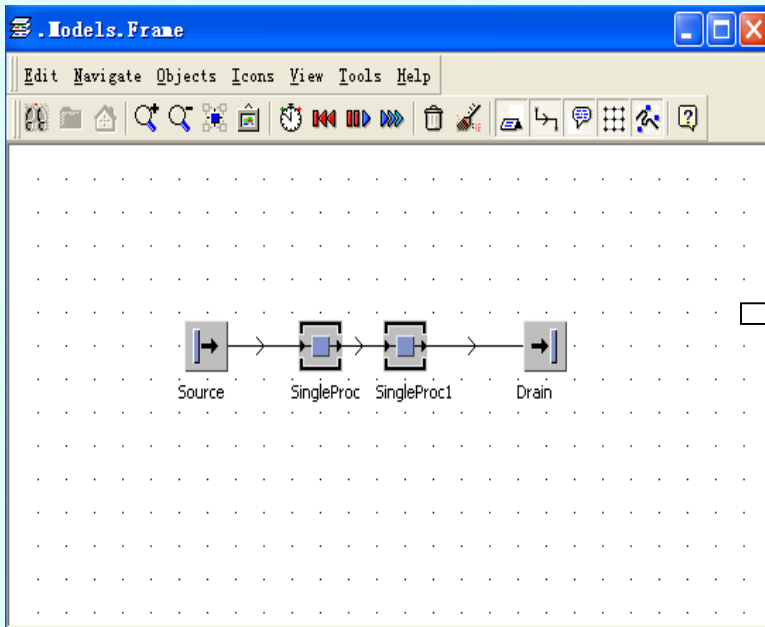


三维仿真

打开点击3D > Start 3D Viewer菜单来启动3D Viewer，3D Viewer启动后会弹出Frame建模框架的3D界面，并在对象结构视图中的3D Tab中显示与二维建模环境下相同的类库结构

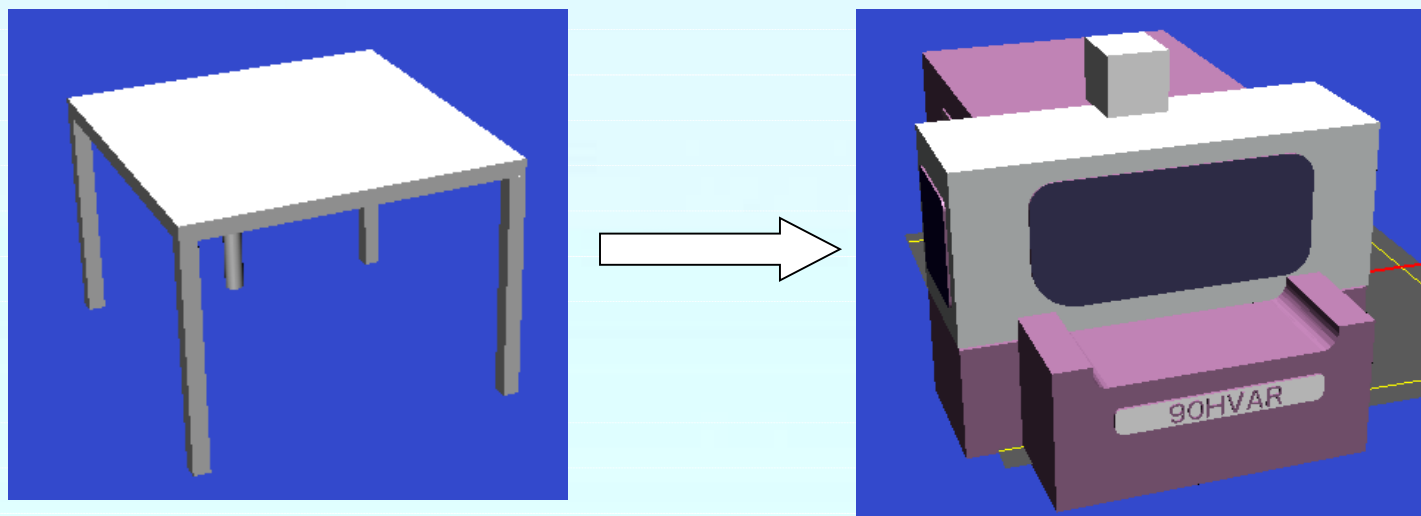


在Frame的三维框架中选择Em-Plant菜单下的2D→3D菜单就可实现二维模型向三维模型的转换，反之，选择3D→2D菜单可实现三维模型向二维模型的转换。由于二维模型建立起来相对较为简单，因此在建模时多先建立二维模型再使用转换工具将其转换为三维模型



使用专业三维建模软件建模，将该模型保存为VRML格式文件，再由**EM-PLANT**转换为其专用的**S3d**格式。

更改物流对象三维模型的方法是右键单击要更改的物流对象，在菜单中选择**Edit→Exchange Graphic**菜单，然后选择保存的**S3d**格式文件。



放映结束！