

附录指南

注意：以斜体显示的选项更适用于高级用户。

文件	说明
新模型	关闭现行模型并且打开新模型
打开模型...	关闭现行模型并且打开原有模型
合并模型...	打开模型并将此模型包含进入选中的原子中或现行模型中。 这样原有模型不会被删除。
保存模型	将现行模型保存为同一文件名。
将模型保存为...	将现行模型保存为新的文件名。前一个模型名称不变。
添加原子至库...	选择一个原子并将其放置于库的底端。
将原子保存为...	显示窗口允许你从库中选择原子。然后，选中的原子可以保存为新名称。
导入	这一选项使你能够导入2D/VR (VR = 虚拟现实) 图标或者VR 声音。图标或声音被添加到用户可以分配至原子的图标声音列表中。
打印2D 版面...	在标准打印机上打印2D窗口。
打印设置	使你能够定义标准打印机的设置。
参数选择	这一选项显示了一些标签，用于定义Enterprise

	Dynamics的标准设置。
启动文本	使你能够修改启动文本。每次启动Enterprise Dynamics时都会执行此文本。修改必须以4DScript进行定义，4DScript是Enterprise Dynamics中的编程语言。
退出	关闭Enterprise Dynamics。

模型	说明
创建	显示库的树型图和模型设计窗口。你可以将原子拖放进入模型设计中，从而创建自己的模型。
设计窗口	显示模型设计窗口。在此窗口中可以创建原子，要么从库中拖放出原子，要么使用任务栏。
子设计窗口	显示另外一个设计窗口，但其等级层面较低，例如，一个组合容器的容量。
模型树型图	显示模型树型图。这个‘模型树型图’展示了模型中所显示原子的等级概况。
库树型图	这个选项显示了库中所有的原子。

仿真	说明
运行控制	开始，停止和修改仿真运行的速度。
时钟	显示时钟。
运行	
停止	
停止和重启	
历史	<p>使你能够通过一次运行生成图表和报告。</p> <p>这个选项‘综合历史’需要检查。它通过运行控制窗口与仿真联系起来。</p> <p>当需要某一具体原子的图表和报告时，此原子的历史必须被保存。最简单的做法是选择选项‘All on’。然而，这样会引起对所有原子进行历史收藏，形成庞大的数据文件！</p>
设置停止时间	<p>使你能够定义仿真运行的确切时间段。此时间段能以秒，分，时或天来定义。</p> <p>通过在仿真菜单中选择菜单选项“Reset + Run until Stoptime”（重启+运行直至停止时间），仿真过程就会在用户输入的确切时间段内停止。</p>
重启+ 运行直至停止时间	<p>重启仿真过程并让它一直运行到输入至菜单选项“设置停止时间”的时间结束为止。</p> <p>注意：在运行控制键旁的这个选项是第二种可用于执行一些仿真运行的方法。然而，在由几个运行组成的试验中，需要试验原子。</p>
重复	<p>当打开这个选项时，Enterprise Dynamics 中100个任意的发生器都会一直重复同样的随机数字。这样就可以执行几个完全一致的</p>

	仿真研究。
反向	当打开这一选项时，Enterprise Dynamics 中100个任意的发生器便设置为反向。这有利于减少极值对多次运行的影响。
种子值	使你能够定义100个任意的发生器中的每一个的起始值。

结果	说明
摘要报告	摘要报告展示了与模型中出现的所有原子相关的基本数据的概况，它基于单个运行之上。 注意：报告原子可以被拖放出模型，从而得到更详细的报告。
图表	显示出原子不同形式的图表，比如序列，柱状图和饼状图，均基于单个运行之上。只有当与所探讨的原子有关的历史选项处于开启状态时才能创建这些图表。

试验	说明
试验向导	帮助你先定义试验设置和输出变量。然后运行试验。
分析结果	关于一个试验的生成报告的定义。

工具	说明
原子编辑器	有了原子编辑器，你可以调整一个原子的功能和外观。 这个十分有效的工具使你能够改变现行原子的行为并创建你自己的原子。这里必须使用4DScript编程语言。
4DScript 界面	可以输入4DScript语言的窗口。接下来便是直接执行命令。
文本编辑器	一个简单的文本编辑器，其功能类似于微软中的记事本。
调试工具...	调试工具使你能够在4DScript代码正在被执行的过程中对其进行逐步分析。
CAD 导入向导	
GUI 建立...	GUI是图形用户界面的缩写。它使得用户能够创建自己的输入栏。
事件管理器	事件管理器帮助你依次进行多个仿真运行。有了这种方式，你只需启动事件序列，就可以回家，第二天清早便可以看到运行结果。
视窗原子标签	这个选项显示了所有选中原子（和包括在选中原子中的所有原子）的标签。 标签是变量和属性，用户可以将其分配至原子。
自动适合	自动适合功能分析一个数据集并寻找最适合的可能性分布。

显示	说明
2D 模型版面	打开2D建模窗口。
2D 模型视窗	打开2D视窗。 警告! 在此窗口中你不可在模型中添加原子或者重新配置原有的原子。若要进行这些操作你必须使用模型版面。

2D 模型子视窗	打开的窗口与2D模型视窗选项一样，但是这里只显示选中原子的内容。
2D 背景色...	这个选项使你能够改变现行2D建模窗口的背景色。
3D 模型视窗	打开3D视窗。
3D 模型子视窗	打开的窗口与3D模型视窗选项一样，但是这里只显示选中原子的内容。
3D 背景色	打开颜色选择窗口。这个选项使你能够定义3D窗口的背景色。此颜色也可在VR窗口中使用。

窗口	说明
关闭所有窗口	关闭所有已打开的窗口。
4DScript 概况	显示所有4DScript 表达的概况，并带有它们的句法说明。你也可以通过F2键打开此窗口。
误差监控器	打开显示4DScript中出现的误差的窗口。
跟踪	打开跟踪窗口。你可以在此窗口中进入4DScript表达。
层面	你可以在多个层面上创建模型，并且你可以锁定某些层面或者隐藏某些层面。在大的模型中，原子往往一层层地堆放在一起，这时此窗口会十分有用。
资源管理器	打开的窗口中显示了所有可用的原子图标。
图表窗口	打开最近创建的图表。在此窗口中不可能生成新图表。

帮助菜单	说明
帮助概况	你可以获得完整的使用说明手册，有以下三个子菜单组成。
快速启动	这些文件涵盖了Enterprise Dynamics中的一些新特色。它们可以帮助你理解这些特色的工作原理。
指南	包括了各种各样的指南可以指导你使用Enterprise Dynamics。
附加项	你可以从Enterprise Dynamics中得到一些附加软件包，这些附加帮助文件包含了与其相关的有用信息。
关于Enterprise Dynamics	显示了有关使用版本和Incontrol Enterprise Dynamics的信息。

附录 2 一些原子的说明

1	产品原子	11
2	源原子	13
3	服务器原子	19
4	序列原子	23
5	接收器原子	25
6	容器原子	27
7	集合原子	31
8	解包原子	35
9	多功能服务原子	37
10	锁定原子	39
11	接锁原子	41

1 产品原子



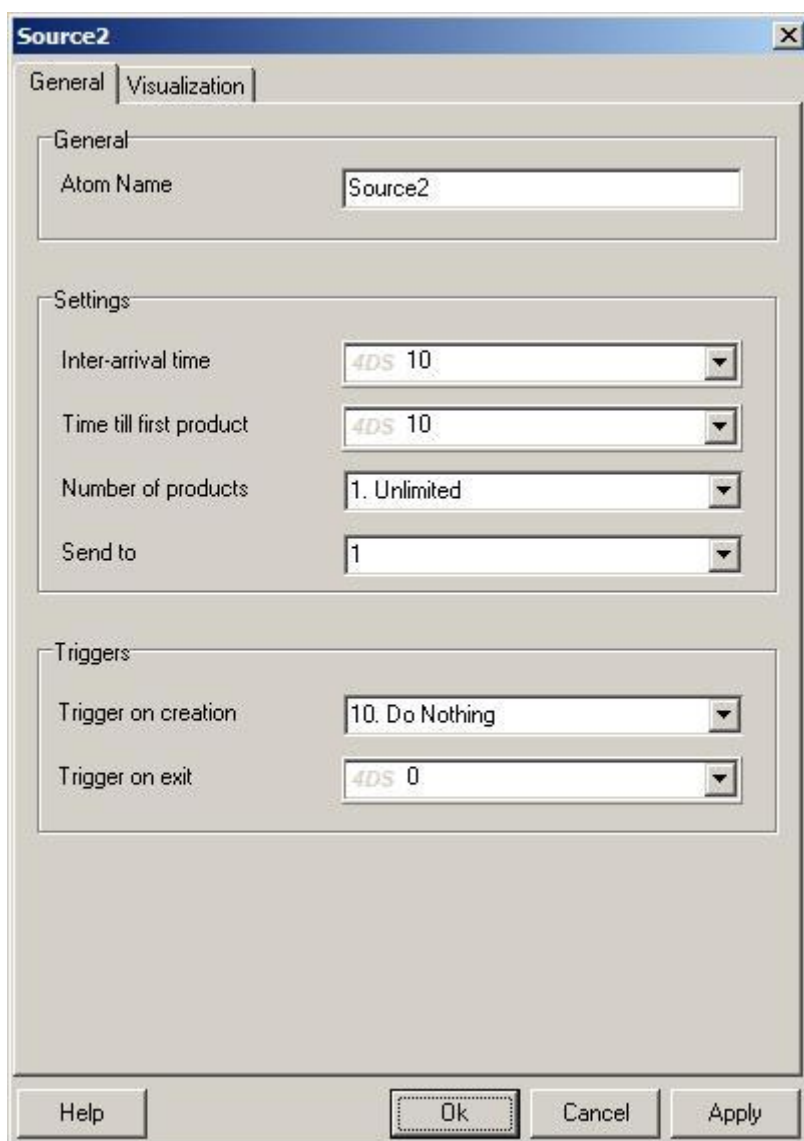
Picture 1-1: 产品原子

产品原子是用来模拟Enterprise Dynamics中的实物流动。这些流动可以由产品，货物，文件或人员组成。以下的原子设置定义如下：

- *原子名称*
给予原子的名称。
- *颜色*
给予原子的颜色。
- *2D图标*
2D窗口中用来代表产品原子的符号。

- *显示2D图标*
使有可能显示2D 图标。如果选项被选中（标准设置），便可以显示图标。
- *3D图标*
在3D窗口中用来代表产品原子的符号。
- *尺寸 X*
原子在x方向的尺寸 (长度以米计)。
- *尺寸 Y*
原子在y方向的尺寸(宽度以米计)。
- *尺寸 Z*
原子在z方向的尺寸(高度以米计)。

2 源原子



Picture 2-1:

源原子允许原子，大多数是产品，以一定的速度进入模型，因此它相当于是一个产品或客户发生器。这个原子通常是模型的第一个元素。

以下的设置可以调整：

- *原子名称*
给予原子的名称。
- *图标*
在2D窗口中用来代表源原子的符号。
- *进入间隔时间*
两个原子进入模型之间的时间。此时间以秒计并且可以是持续的，但是由

可能性分布定义。点击右边的三角显示下拉菜单，其中有一些可能的参数和值的可能性分布。

- **第一个产品进入时间**
第一个产品进入所用的时间。第一次进入之后便可应用进入间隔时间中的可能性分布。
默认值为10秒；如果你想所有产品的进入间隔时间都相同，那么在进入时间间隔中选择相同的表达。
- **产品数目**
有了这个选项你可以限制产品进入你的模型。有两种选择：
 1. 无限制 (默认)
 2. 最大产品数量为100有了第二个选项你可以选择你想要的进入数目。这与更广泛应用的锁定原子功能相似。
- **发送至**
显示出其他原子（大多数是产品）离开此原子时通过的输出路径的数目。这里会显示1到此原子输出路径总数之间的某个数字。如果结果是0，那么从来没有发生过发送的行为。如果原子被阻塞了是因为接受路径的输入路径被关闭了，那么只有当情况改变并且允许发送时‘发送至’的命令才会被重新考虑。

在选项发送至中，用户可以输入一个数字或者选择以下预先定义好的选项中的一个。在这些选项中，加粗部分（在屏幕上显示为蓝色）可以被用户修改：

- 1: **指定路径: 通常发送至路径 1。**
产品原子总是被发送至预先定义好的输出路径中。
- 2: **开放路径(第一个路径优先): 从第一个路径开始搜寻，将原子发送至搜寻所发现的第一个开放路径中。**
产品原子被发送至Enterprise Dynamics搜寻到的第一个开放路径中。它从第一个输出路径开始搜寻，然后到第二个依次下去。
- 3: **开放路径(最后一个路径优先): 从最后一个路径开始搜寻，将原子发送至搜寻所发现的第一个开放路径中。**
产品被发送至Enterprise Dynamics搜寻中遇到的第一个开放路径中，它从最后一个路径开始搜寻，依次前推。
- 4: **随机开放路径: 从所有开放的输出路径中随机选择一个路径。**
Enterprise Dynamics从所有开放的路径中随机选择一个路径。在较长的仿真运行中，这样选择使得一组服务器得到了均等的使用。
- 5: **按比例: 90% 的产品被发送至路径1, 剩余的比例为路径2。**
确定比例的产品被发送至某个指定的路径，其余的发送至另一个路径。用户可以定义路径和比例。
- 6: **按原子名称: 如果序列中第一个原子的名称与 AtomName 相应，那么将其发送至路径1 或2。**
原子是以名称排序的。如果名称和用户输入的名称一致，那么产品被发

送至路径1，否则发送至路径2。用户可以调整路径的数目和原子的名称。

- 7: **按标签值 (直接):** 路径数目直接写在标签上，标签被命名为序列中**第一个**原子的**LabelName**。如果标签值为0，那么发送至路径**1**。

原子在标签值的基础上排序。用户已经为标签定义了名称。变量值与输出路径值相对应。如果值为0，便使用预先定义好的退出。需要注意：搜寻不在原子上的标签时也会出现值为0的情况。

- 8: **按标签值 (有条件的):** 如果名称为序列中第一个原子的**LabelName**的标签值小于1，那么发送至路径 **1** 或 **2**。

同样在这里，某一特定的标签值决定了对输出路径的选择。如果原子值低于1，那么原子被发送至路径1，否则发送至路径2。可以编辑所有的值和比较方式（低于，高于，等于）。

- 9: **按标签文本:** 如果名称为序列中第一个原子的**LabelName**的标签上的文本与文本相匹配，那么发送至路径 **1** 或 **2**。

当已定义的标签值与指定文本相等时，原子被发送至路径**1**，否则至路径**2**。可以编辑文本和路径数字。

- 10: **条件申明:** 如果**1** > **0**，那么发送至路径 **1** 或 发送至路径**2**。

如果某一具体的值高于另一个值，那么将原子发送至路径**1**， 否则至路径**2**。可以编辑比较方式和路径数字。

- 11: **按图标名称:** 如果序列中第一个原子的图标名称与**IconName**相应，那么发送至路径**1** 或 **2**。

如果原子图标的名称与已定义的名称相应，原子被发送至路径1，否则至路径2。 可以明确图标名称和路径数字。

- 12: **按图标数字:** 如果序列中第一个原子的图标数字为 **1**，发送至路径 **1** 或 路径**2**。

如果原子的图标数字为**1**，那么将原子发送至路径**1**，否则至路径**2**。可以定义比较方式和路径数字。

- 13: **循环:** 所有的输出路径都是循环使用的。如果路径被关闭了，那么等到它打开为止。

所有的输出路径都是连续使用的。如果一个路径没有打开，Enterprise Dynamics 会等待直至它打开。

- 14: **最低序列:** 发送至与最低序列原子相连的路径。

原子被发送至序列最短的输出路径中。在同等长度下，选择个数最少的输出路径。

- 15: **最高序列:** 发送至与最高序列原子相连的路径。

原子被发送至序列最长的输出路径中。在同等长度下，选择个数最多的输出路径。

- 16: **查找表格:** 发送至指定为全部表格中**第一行第二列**的路径，表格名称为表格**1**。

将原子发送至已定义为表格**第一行第二列**的路径中。行和列的数字以及表格的名称都可以重新拟定。

注意表格必须在模型中以一个独立的原子出现！

- 17: **循环如果可用:** 所有输出路径都为循环使用, 只要路径可用。如果路径关闭, 则选中下一个可用的路径。

所有路径都是连续使用的, 但是当所需的路径关闭时, 便可以选择下一个可用的路径。

- 18: **搭配图标数字或为空:** 发送至包含同样图标产品的序列中。如果没有相匹配的图标, 则从最后一个输出路径开始寻找并发送至第一个空序列中。

将原子排序这样它们总是以包含同样图标数字原子的序列进入。如果没有找到包含同样图标数字原子的序列, ED便从与最后一个输出路径相连的序列开始寻找第一个空序列。

- 19: **下两个原子的最短序列:** 发送至与最短序列相连的输出路径, 这个最短序列包括了下两个原子。

Enterprise Dynamics 评估了与输出路径相连的每个原子和临近此原子的总序列。然后它会将下一个原子发送至与最小容量序列相连的路径中。比如, 一个原子可以被发送至三个不同的序列中, 每个序列后都有一个服务器。这个选项使得产品不会被发送至一个序列, 其中的服务器已经在运行中, 而其他的服务器仍然可用。

- 20: **按用户:** 输入自己的 4DScript 表达形成1 到路径数目之间的某个值: 1。你可以按下小按钮打开4DScript 编辑器。

用户写入4DScript 代码, 形成输出路径。点击文本旁小的方形按钮即可打开4DScript编辑器。

- 21: **随机路径:** 随机选择一个路径。如果路径处于打开状态则发送至此路径, 否则再次选择其他打开的路径。

Enterprise Dynamics 选择随机路径。如果此路径处于打开状态, 那么产品被发送至此路径, 如果关闭, 则再次选择另一个打开的路径。

- **开始创建**

当一个原子进入模型时执行这个命令。用户可以定义自己的4DScript 表达, 或者从以下选项中挑出一个:

- 1: **赋值标签:** 产品被赋予名称为**LabelName**并且值为1的标签。
产品被赋予一个有指定名称和特定值的标签。标签名和值可以重新定义。
- 2: **自动命名:** 计算器会在每个产品名称末尾添加数字。
会在产品名称末尾添加数字。比如, 第一个产品称为产品1, 第二个产品称为产品2。
- 3: **随机图标:** 产品会被随机赋予一个2到6之间的图标数字。

每个产品都会随机得到一个图标。图标的数字在两个指定值之间。

- 4: **设置尺寸:** 产品尺寸设置为 $X=50\text{ cm}$, $Y=40\text{ cm}$, $Z=30\text{ cm}$ 。
产品的尺寸随着输入值而变化。
- 5: **随机尺寸:** 产品尺寸在如下范围内随机设置: $X=50\text{ 到 }100\text{ cm}$, $Y=50\text{ 到 }100\text{ cm}$, $Z=50\text{ 到 }100\text{ cm}$ 。
产品尺寸在已定义值的范围内随机变化。
- 6: **设置颜色:** 产品被设置为紫色。
产品颜色会随着用户的定义而变化。注意到在4DScript中, 选中的颜色必须带有前缀“color”。所以`colorpurple` 代表紫色的命令。除了可以输入命令`colorpurple`外, 你也同样可以输入颜色值。
- 7: **随机颜色:** 产品被随机赋予颜色。
产品被随机赋予颜色。
- 8: **随机尺寸和颜色:** 产品被随机赋予颜色并且其尺寸在下列范围内随机变化 $X=50\text{ 到 }100\text{ cm}$, $Y=50\text{ 到 }100\text{ cm}$, $Z=50\text{ 到 }100\text{ cm}$ 。
产品会随机得到颜色和尺寸(在指定范围内)。
- 9: **轮廓:** 只显示产品的简单轮廓, 而不是图标。
产品图标变得不可见, 只能看到轮廓。
- 10: **无进行事项。**
不会发生任何行为。这是标准系统设置。

- **开始退出**

当产品离开原子时便会执行这个命令。你要么使用4DScript 命令, 要么从预先定义好的表达中选择一个。问号表示用户必须在此处输入一个值。

在开始退出中预先定义好的表达可能有:

- 1: **setlabel([?],?,i).**

使用这个4DScript 命令可以给离开来源的原子添加标签。代码是: `setlabel([label name],value,i)`。

例子

为了给产品分配值为1的标签“complete”, 可以运用以下代码:

`setlabel([complete],1,i)`。字母 *i* 是指涉及到的原子, 即经历离开来源过程的原子。如果标签必须被放置在来源本身上, *i* 可以被*c*(现行的)代替。

例子

`Setlabel([cycletime], uniform(25,45),i)` 表示给产品定义的标签名为“cycletime”, 标签值为25秒到45秒之间均匀分布的某个值。稍后可以使用这个结果作为服务器的循环时间。

- 2: **set(name(i),[?]).**

改变离开来源原子的名称。问号‘?’这里必须被选中的名称取代。

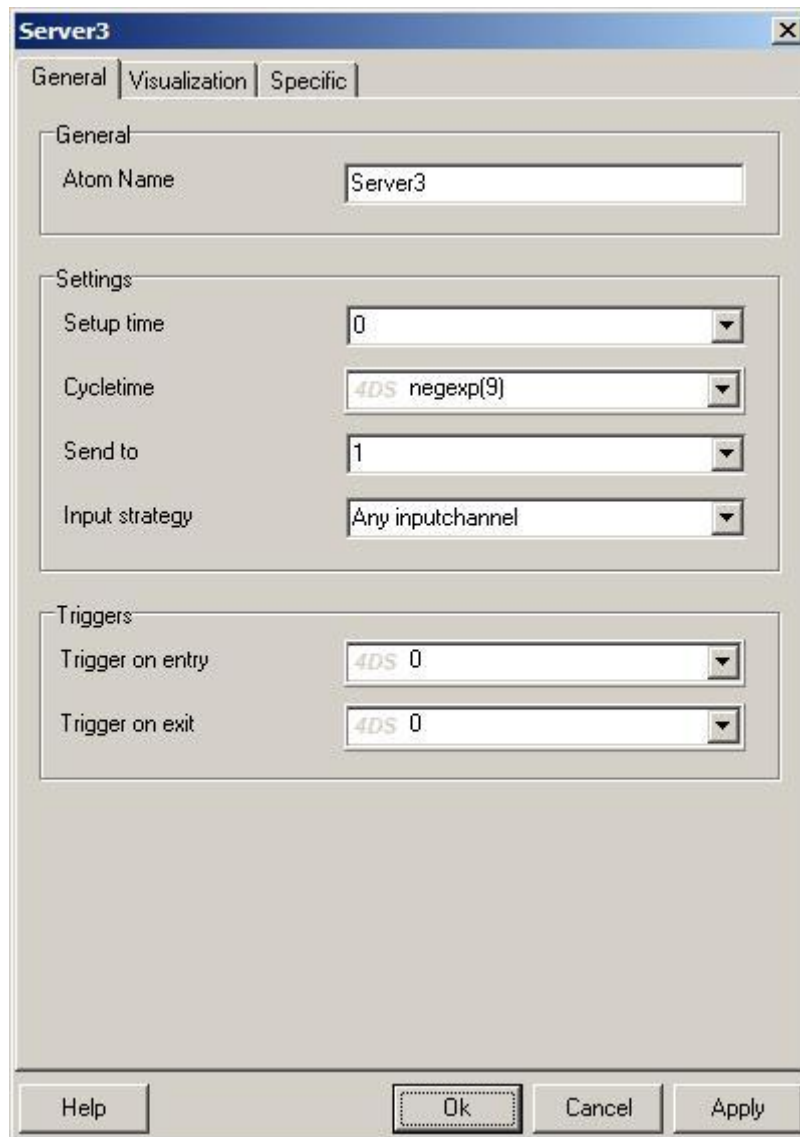
- 3: *set(icon(i),?)*.
将原子图标改变为带有数字‘?’的图标。
- 4: *set(icon(i),icobyname([?]))*.
将原子图标改变为带有名称‘?’的图标。
- 5: *set(color(i),coloryellow)*.
将原子颜色改变为已定义的颜色。在Enterprise Dynamics中，要么通过选择颜色的数字，要么在颜色前输入前缀“color”来定义颜色，比如colorred。
- 6: *setsize(?,?,?,i)*.
根据已定义的尺寸(x,y,z)改变原子尺寸。
- 7: *setloc(?,?,?,i)*.
根据命令中的定义(x,y,z)改变原子的位置。
- 8: *freeoperators(atombyname([Team],model),i)*.
允许重新使用操作原子。将‘Team’替换为Team原子的名称。这个选项只适用于高级用户。
- 9: *if(=(?,?),?,?)*.
有条件的比较。比如，输入以下代码：
`if(=(thesis1,thesis2),command1,command2)`

如果thesis1 和 thesis2 等同，那么会执行命令1，否则执行命令2。命令2也可以省略。
- 10: *if(=(label([?],i),?),?,?)*.
有条件的比较，其中考虑的是标签值。

例子
使用如下命令后，如果标签‘Reject’ 的值为1，那么所有不合格都会标注为红色，所有合格产品都会标注为绿色：
`if(=(label([Reject],i),1),set(color(i),colorred),set(color(i),colorgreen))`
- 11: *if(comparetext(name(i),[?]),?,?)*.
有条件的比较，其中用到了原子的名称。它的使用原理与选项10中一样。

为了了解更多关于4DScript命令的信息，可查看附录3或Enterprise Dynamics中包含的帮助文件。

3 服务器原子



Picture 3-1: 服务器原子

服务器用于需要耗费一定时间的模型操作，比如机器处理产品或者消费者在柜台结帐等。因此，服务器可以充当机器，柜台，助手或另一种类型的交易地点或设备。就像循环时间一样，其他参数，比如时间设置或是几个产品的同时处理等都可以重新定义。

以下值是可以编辑的：

- **原子名称**
给予原子的名称。
- **图标**
在2D窗口中用来代表服务器原子的符号。
- **3D图标**
在3D窗口中用来代表服务器原子的符号。
- **主色**
- **次色**
- **时间设置**
在实际操作开始之前所需的时间。比如，清洗机器，新产品的调整设置等。
- 点击三角形按钮，打开一系列选项，其中包括允许为每个产品定义设置或仅允许为不同类型的产品定义设置。
除了使用这些选项外，用户还可以创建自己的4DScript。
- **循环时间**
处理产品所需的时间。点击箭头，就会出现一个预先定义的4DScript命令列表。
重要：在共同处理产品（批处理）的情况下，循环时间是指整个一批产品而不是每件个体产品的循环时间。并且只有当一批产品齐全时处理才会开始。
- **发送至**
显示产品必须被送往的路径。
为了解更多详细的信息：见源原子中对‘发送至’的说明。
- **输入策略**
调整通过之前原子的输出路径到达这个指定原子的通道。输入策略有多重作用：它可以打开一个或更多路径并且它可以定义可用路径中的产品顺序。

你可以将输入策略比作交通灯中的先后顺序，一些交通灯从‘红色’到‘绿色’交替变换，为一条或几条支路调整秩序，它不顾实际的交通情况，并且已经定义好这些支路的处理优先级。

警告：前三个输入策略打开所有的输入路径而后两个每次只打开一个输入路径！

1: **任何输入路径。**

这个策略一旦激活，就会打开原子中所有的输入路径。如果通过输入路径连接的原子中有一个以上的原子可以被发送的话，那么通过最小数字输入路径进入的原子的优先级别最高。当产品不断地通过第一个路径进入时，其他的路径都将被阻塞。

2: *最大序列。*

这个策略一旦激活，就会打开原子中所有的输入路径。如果通过输入路径连接的原子中有一个以上的原子可以被发送的话，那么拥有最长序列或最大容量的原子的优先级别最高。注意到在有几个同等长度的序列的情况下，将选择最少数目的输入路径。

3: *最长等待。*

这个策略一旦激活，就会打开原子中所有的输入路径。如果通过输入路径连接的原子中有一个以上的原子可以被发送的话，那么平均等待时间最高的原子的优先级别最高。在有几个原子等待时间相等的情况下，通常会选择数字最低的输入路径。注意到这并不意味着象前一个选项中一样，序列的长度大约相等。

4: *循环。*

这个策略首先会打开第一个输入路径然后等待产品通过这个输入路径被发送出去。在第二个循环中，便轮到第二个输入路径，依此类推。当产品通过最后一个输入路径运行时，整个程序又会重新回到第一个路径并进行循环。

重要标语：在第一个产品之后这个策略就会被激活！因此，在有三个输入路径的情况下，这个策略会给出x,2,3, 1,2,3, 1,2,3 的安排，其中x可以为1,2或3！

5: *路径 1。*

在这种情况下，你可以输入一个具体的输入路径，所有产品都必须从中通过。如果输入了数字1，那么产品就只能通过输入路径1进入。注意到这条规则并不适用于第一个产品，因为在起始时所有路径都是打开的。

- *批 (B)*

可以在这里输入批的尺寸。标准设置是1。

- *批规则*

用来设置批量。有三种选择：

1: *批量进入，单个输出(第一个)。*

当进入服务器的产品数目与批大小相等时，在前面的产品就会被下一个原子超越。其他的产品会消失。

2: *批量进入，批量输出。*

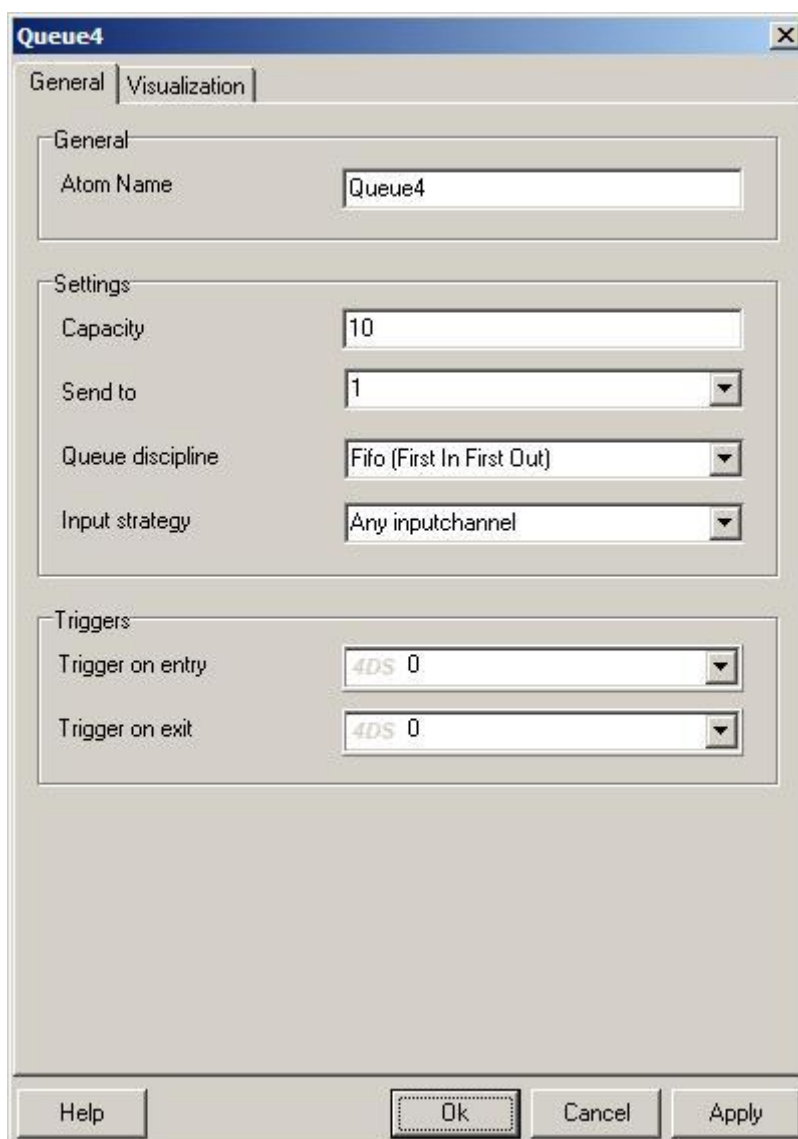
当进入服务器的产品数目与批大小相等时，产品会进入到下一个原子之前。只有当批中所有的产品都离开服务器时，服务器才会重新接收产品。

3: *单个进入，批量输出。*

每次当一个产品进入服务器时，批输入栏中所定义的所有数量的产品都会离开原子。产品都是进入服务器原子的复本。只有当所有其他产品都离开原子时，服务器才会重新接收产品。

- *开始进入*
这一栏中输入的命令会在产品进入服务器时被执行。
了解更多信息：见源原子中对‘开始退出’的说明。
- *开始退出*
这一栏中输入的命令会在产品离开服务器时被执行。
了解更多信息：见源原子中对‘开始退出’的说明。
- *忙碌时间*
当选中这个选项时，考虑进选项“故障间的平均时间”的时间仅仅为服务器实际使用的时间，而不是总的模拟时间。
- *MTTF*
此缩写的完整形式为Mean Time To Failure（故障之间的平均时间），指的是修复末期和下一个故障开始之间通过的平均时间。可以在输入栏中定义两个服务器故障之间的平均时间。此时间必须以秒输入。
- *MTTR*
此缩写的完整形式为Mean Time To Repair（修复的平均时间）。可以在输入栏中定义修复服务器所需的平均时间。
- *MCBF*
此缩写的完整形式为Mean Cycles Between Failure（故障之间的平均循环）。可以在输入栏中输入两个故障之间的平均循环次数。在MCBF中两个故障之间没有确定的时间，只有批的确定数字。
注意：当MTBF 栏和MCBF栏中均填入数字，那么故障的产生将由这两个定义决定。
- *MTTR 循环周期*
此修复平均时间（MTTR=Mean Time To Repair）输入栏适用于MCBF定义下的故障。

4 序列原子



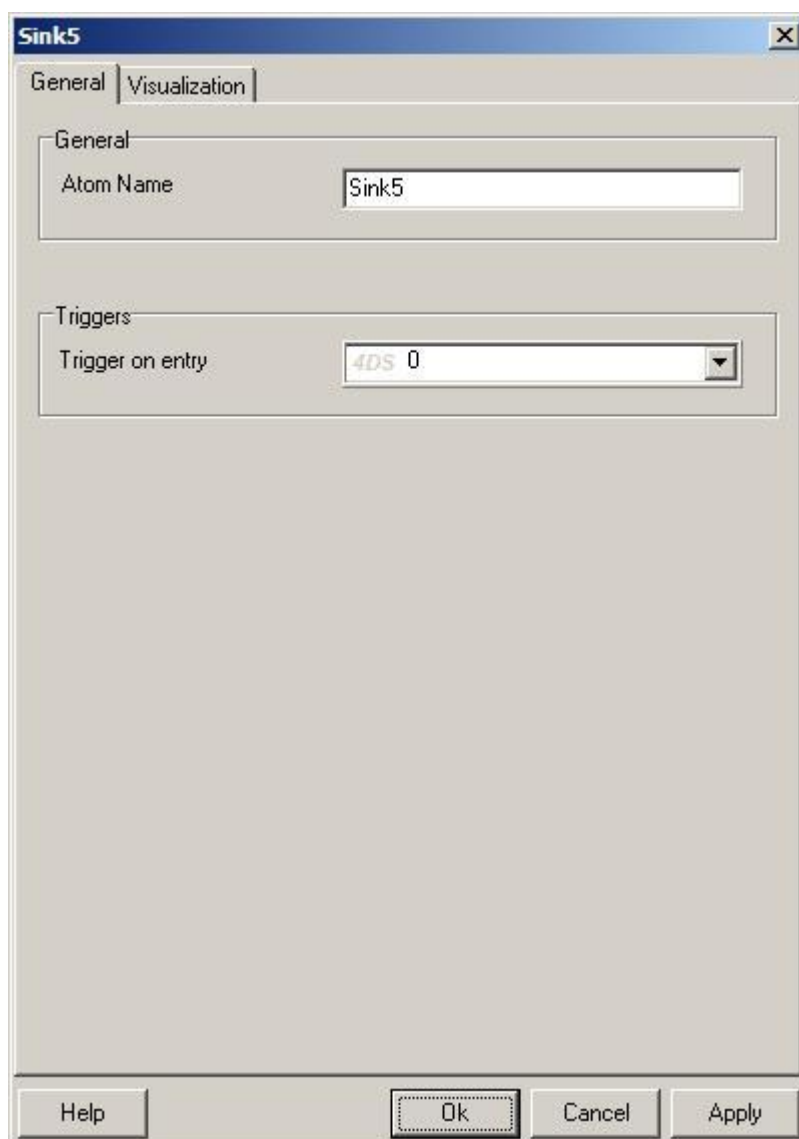
Picture 4-1: 序列原子

当下一个原子被占据时，序列原子将产品置于序列中。以下设置可以被调整：

- *原子名称*
赋予原子的名称。
- *图标*
在2D窗口中用来代表序列原子的符号。
- *3D图标*
在3D窗口中用来代表序列原子的符号。

- **容量**
序列的容量。当序列中存在的产品数目达到容量输入栏中定义的数目时，序列中就再也不能放入新的产品了。
- **发送至**
显示出产品被发送至的输出路径。
了解更多信息：见源原子中对发送至的说明。
- **序列原则**
产品在序列中被安排的方式。有以下一些可能的选项：
 - 1: **第一个进，第一个出。**
按照原子进入的顺序排列它们在序列中的先后位置。
 - 2: **最后一个进，第一个出。**
进入的原子被放置到序列最前端。因此，产品离开序列的顺序与其进入时相反。
 - 3: **随机。**
这种序列原则使得进入的产品在序列中随机排列。
 - 4: **按标签升序排列。**
指定的标签中值最低的产品被放置到序列的最前端。
警告!: 如果产品排列不恰当，结果可能是在标签名称前后出现空格。
 - 5: **按标签降序排列。**
指定的标签中值最高的产品被放置到序列的最前端。
警告!: 如果产品排列不恰当，结果可能是在标签名称前后出现空格。
 - 6: **用户定义。**
产品在序列中的位置取决于用户的定义。
- **输入策略**
这个输入窗口可以用来显示将要使用哪个输入路径。
了解更多信息：见源原子中对输入策略的说明。
- **开始进入**
当产品进入序列原子中时会执行此栏中输入的命令。
了解更多信息：见源原子中对‘开始进入’的说明。
- **开始退出**
当产品离开序列原子时会执行此栏中输入的命令。
了解更多信息：见源原子中对‘开始退出’的说明。

5 接收器原子

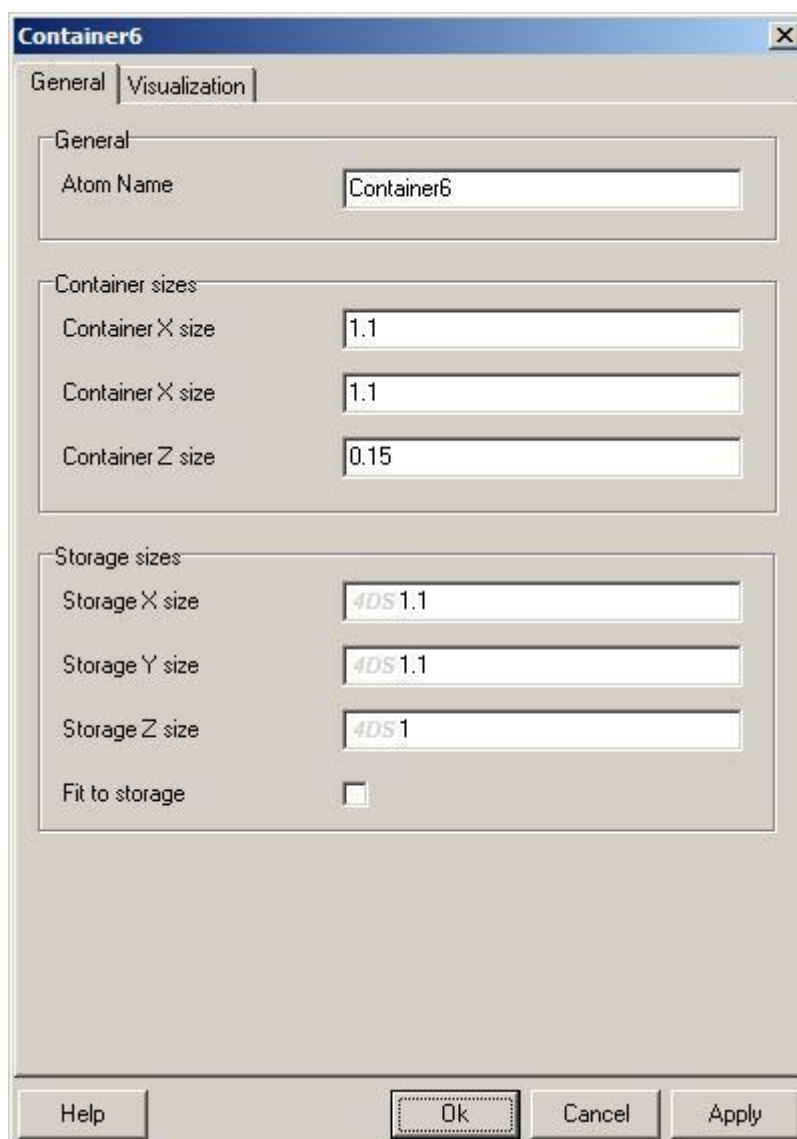


Picture 5-1: 接收器原子

此原子允许产品离开模型。以下设置可以被调整：

- **原子名称**
给予原子的名称。
- **开始进入**
一旦产品进入接收器原子就会执行此栏中输入的命令。
- **图标**
在2D窗口中用来代表接收器原子的符号。

- 3D 图标
在3D窗口中用来代表接收器原子的3D图标。



Picture 6-1: 容器原子

容器原子是特别为存储或堆放其他原子，比如盒子或货盘而创建的。对于容器原子而言，一些标准设置选项比如特殊的3D图标是为了改进视觉效果而设计的。而且，产品的大小可以自动调整为与容器的大小一样。原则上，容器原子和产品原子一样都是通过一个来源（或者通过进入列表原子）放置于模型中。使用集合原子便可以将产品放入容器中。

- **原子名称**
给予原子的名称。
- **颜色**
给予原子的颜色。

- **2D图标**
在2D窗口中用来代表容器原子的符号。
- **3D图标**
在3D窗口中用来代表容器原子的符号。
- **进程显现**
选中这个选项时会加快3D动画，但是可能导致3D图标突然消失（这取决于3D动画的大小和角度）。加速主要是由于没有绘制窗口以外的那些图标。尤其是在大型的模型中选中这个选项很重要。
- **容器 X 尺寸**
在x方向（长）测量的容器的尺寸。
- **容器 Y 尺寸**
在y方向（宽）测量的容器的尺寸。
- **容器 Z 尺寸**
在z方向（高）测量的容器的尺寸。
- **开始进入**
当产品进入容器原子时便会执行此栏中输入的命令。
了解更多信息：见源原子中对开始进入的说明。
- **开始退出**
当产品离开容器原子时便会执行此栏中输入的命令。
了解更多信息：见源原子中对开始退出的说明。
- **存储 X 尺寸**
标注存储一个进入的产品所需的尺寸（x方向）。
当输入的尺寸超过产品实际所需的空间时会使得进入的产品之间出现空格。
- **存储 Y 尺寸**
标注存储一个进入的产品所需的尺寸（y方向）。
当输入的尺寸超过产品实际所需的空间时会使得个体产品之间出现间隔。
- **存储 Z 尺寸**
标注存储一个进入的产品所需的尺寸（z方向）。
当输入的尺寸超过产品（见产品原子）实际所需的空间时会使得产品之间出现间隔。
- **适于存储**
选中这个选项会调整产品的尺寸使其恰好符合预先定义的存储尺寸。

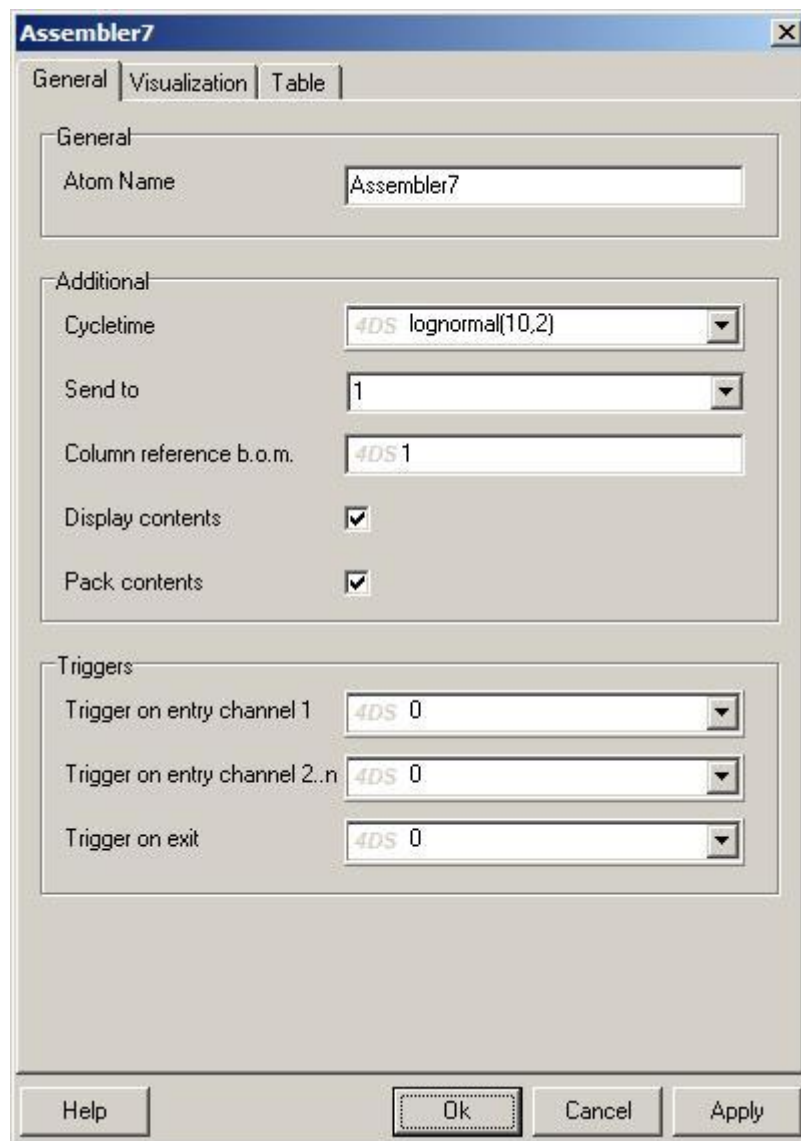
警告!

尺寸有三种类型：产品尺寸（见产品原子），容器尺寸和存储尺寸。存储尺寸可以对包装方式进行（视觉）调整。

例子：

一个容量大小为1x1x0.15的货盘可以存放一个尺寸为1x1x1 4 高的产品，如果集合程序将4个产品置于货盘中。

但是如果选择了适于存储的选项，并将存储尺寸设置为0.5x0.5x0.5，它将刚好能把四个盒子放在一层中！



Picture 7-1: 集合原子

此原子从几个来源聚集原子。被放入另一个原子的原子可能仍然保存着，可能已经损坏。除了模拟真实的集合工作外，这个原子还对用盒或货盘包装产品，甚至对堆放顺序都十分有用。

货盘，盒子或顺序指令通常是通过第一个输入路径进入的，而产品则通过其他路径进入。根据设置，这些产品在通过第一个输入路径进入的原子中要么被损坏，要么仍然保存着。

- **原子名称**
给予原子的名称。
- **图标**
在2D窗口中用来代表集合原子的符号。
- **循环时间**
采集产品所需的时间。这个时间是从所有需要的原子进入集合原子的时刻开始计算的，并且指的是整个集合过程而不是每个单独的产品！点击箭头可以打开一些预先定义的4DScript 表达。
- **发送至**
显示产品被发送至哪个出口路径。
了解更多信息：见源原子中对‘发送至’的说明。
- **开始进入路径1**
当产品通过第一个输入路径进入集合原子时会执行此栏中输入的命令。
了解更多信息：见源原子中对‘开始退出’的说明。
- **开始进入路径 2..n**
当产品通过其他输入路径中的一个（也就是说不是路径1）进入集合原子时便会执行此栏中输入的命令。
了解更多信息：见源原子中对‘开始退出’的说明。
- **开始退出**
当产品离开集合原子时便会执行此栏中的命令。
了解更多信息：见源原子中对‘开始退出’的说明。

材料清单

集合原子同样有材料清单(b.o.m.)表格, (双击原子即可见)。材料清单通过输入路径表明需要多少原子用于集合终端产品。

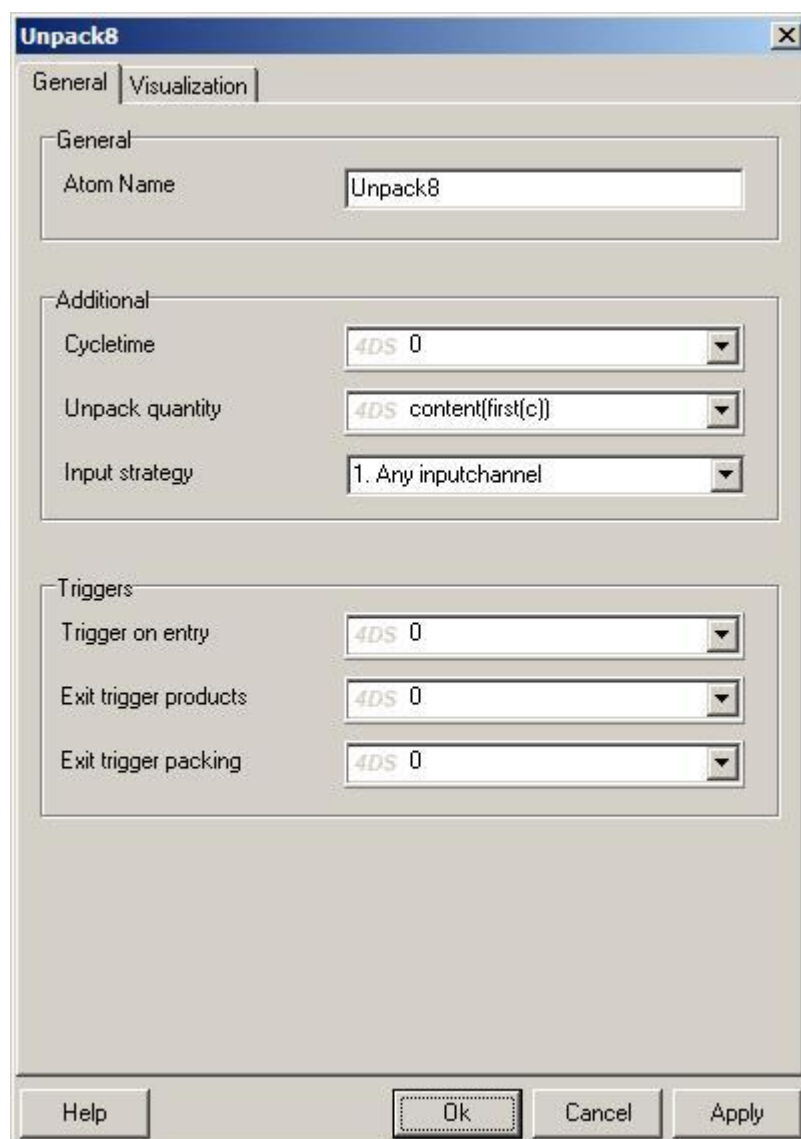
这个材料清单表格有许多行和列。列数与输入路径的数目一致。尽管默认值为1，但是用户可以自己定义列数。

这表明对于所集合的每一种产品类型都可以创建单独的一列。当产品通过第一个输入路径进入时，用户可以定义材料清单中哪一列是用来存放进入的其他原子。

- **材料清单列数。**
定义材料清单中的列数。
- **材料清单中的列数参考。**

表明将使用材料清单中的哪些列。用户可以输入一个数字，也可以输入 4DScript 命令从而产生一个数字。当第一个原子通过输入路径1进入时会评估此栏，并且它会定义出当其他原子进入集合原子时将使用哪些栏。

- *显示内容*
选中这个选项将显示集合原子的内容。
- *包的内容*
选中这个选项时，原子（即：产品）被放置在通过第一个路径进入的原子（即：主要产品或容器）中。如果没有选择这个选项，所有没有通过第一个路径进入的原子都会被损坏。



Picture 8-1: 解包原子

解包原子被用来清除容器原子中的产品。解包以后，容器原子便通过第二个输出路径发送出去并且产品通过第一个输出路径发送出去。

- **原子名称**
给予原子的名称。
- **图标**
在2D窗口中用来代表解包原子的符号。
- **循环时间**
解包容器原子所需的时间。这个时间从容器原子进入解包原子的时刻开始计量，并且指的是每一个容器原子的‘卸载时间’（不是每个单个产品）。

点击箭头将会打开一些预先定义好的4DScript表达。

了解更多信息：见服务器原子中对‘循环时间’的说明。

- **解包数量**

定义将被解包的原子数量。当达到这个数量时，容器原子便被发送出去。

默认设置容量(`first(c)`) 保证了所有的产品都会从容器原子中清除。

三个其他预先定义好的选项是：

```
duniform(1,10)
label([?],first(c))
if(=(?,?),?,?)
```

- **输入策略**

此栏可用于表明应该使用哪个输入路径。

了解更多信息：见服务器原子中对‘输入策略’的说明。

- **开始进入**

当容器原子进入解包原子时将会执行此栏中输入的命令。

了解更多信息：见源原子中对‘开始退出’的说明。

- **退出产品**

当产品离开解包原子时将会执行此栏中输入的命令。

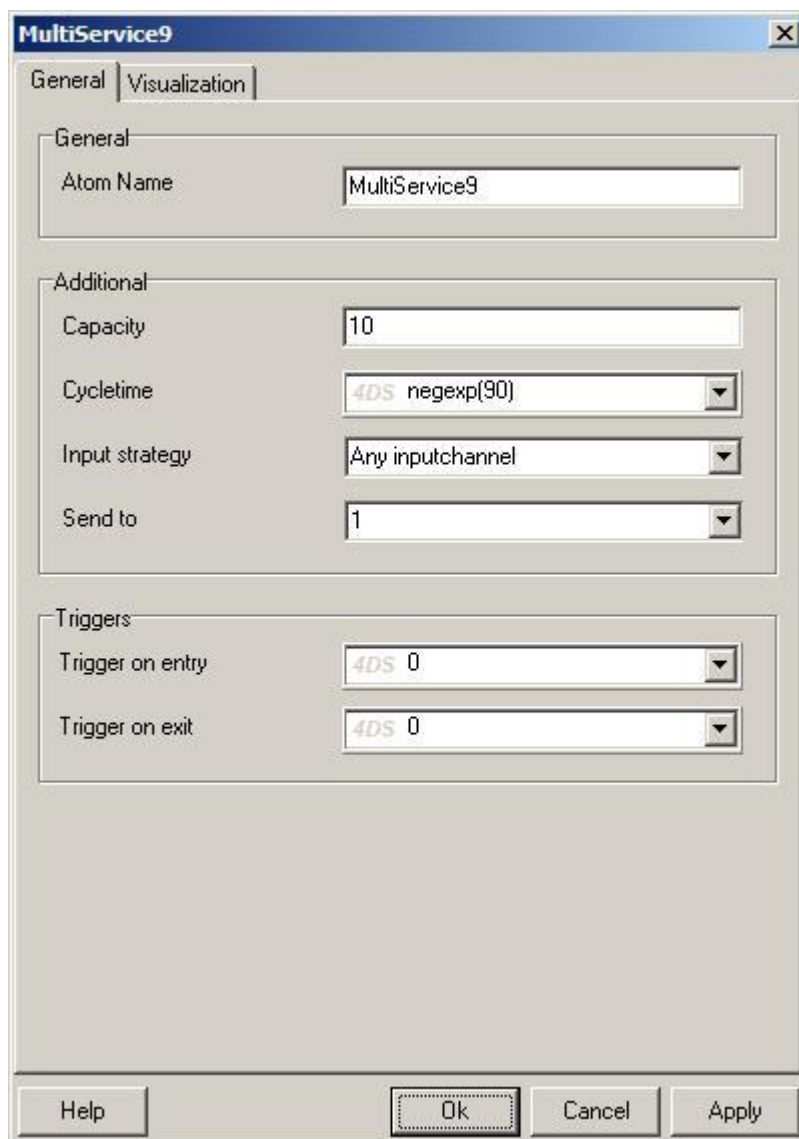
了解更多信息：见源原子中对‘开始退出’的说明。

- **退出打包**

当容器原子离开解包原子时将会执行此栏中输入的命令。

了解更多信息：见源原子中对‘开始退出’的说明。

9 多功能服务原子



Picture 9-1: 多功能服务原子

多功能服务原子和一些单独的服务器原子的基本功能相同。服务器原子一次只能处理一个产品，而多功能服务原子可以同时处理几个产品。设置选项不象一般的服务器中那样细致，但是所有基本的功能都是可用的。

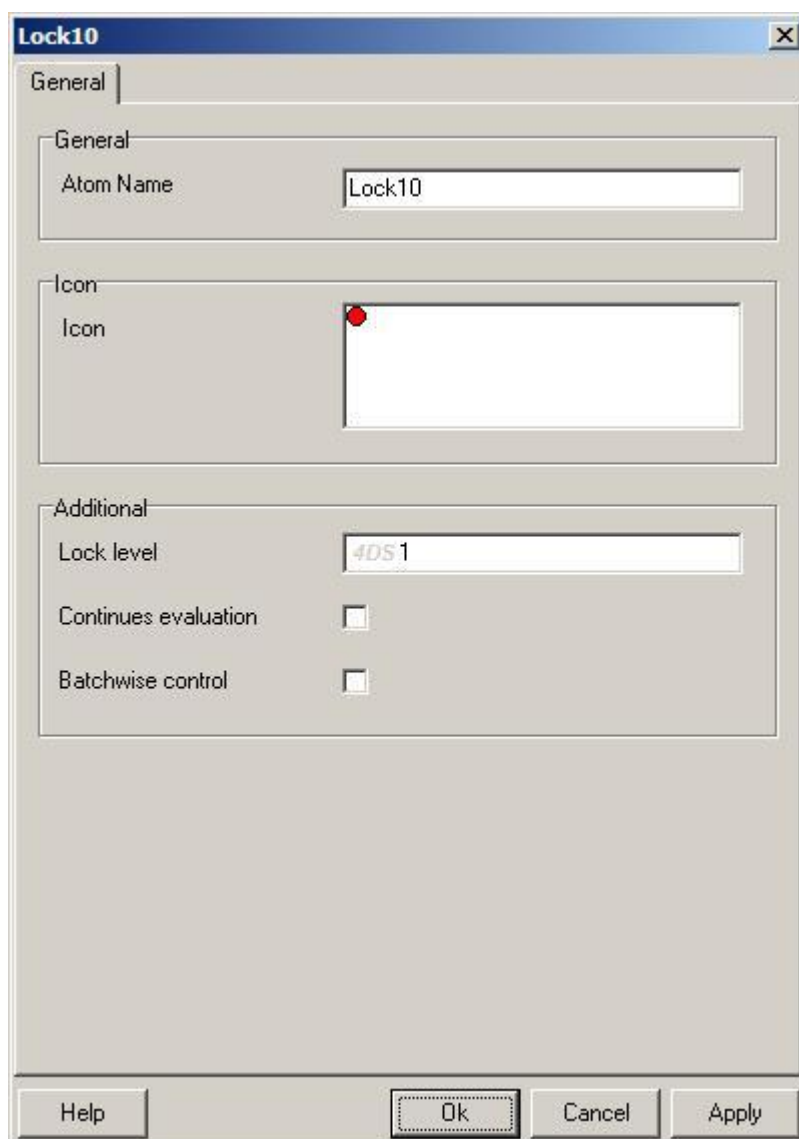
- *原子名称*
给予原子的名称。
- *图标*
在2D窗口中用来代表多功能服务原子的符号。

- **2D 显示**
在2D窗口中产品在原子中的显示方式：
 - 1: **标准显示**
多功能服务原子中的产品是不可见的，但是有文本显示存在着多少产品。
 - 2: **从左向右移动**
产品可见并且从左向右移动。
 - 3: **从右向左移动**
产品可见并且从右向左移动。
 - 4: **垂直方向**
当更多产品进入多功能服务原子时，产品是不可见的，可见的是有色区域，其高度增加。
 - 5: **水平方向**
当更多产品进入多功能服务原子时，产品是不可见的，可见的是有色区域，其宽度增加。
 - 6: **线形容量**
产品是可见的并且呈线性上下排列。
- **容量**
列出可以同时处理的产品数目。

对于以下的选项，我们给出了服务器原子的描述：

- **循环时间**
- **输入策略**
- **发送至**
- **开始进入**
- **开始退出**

10 锁定原子



Picture 10-1: 锁定原子

当仅单独运行锁定原子时，它的功能是作为一个门户只允许锁定水平中所指示的一些产品穿过。

当锁定原子和解锁原子结合起来使用时，锁定原子能够控制模型某个部分的产品的最大数目。由于锁定原子和解锁原子而解除限制的模型中，这一过程中的工作量不会超过所谓的锁定水平。

由于两个原子之间的联系，解锁原子如下所述。

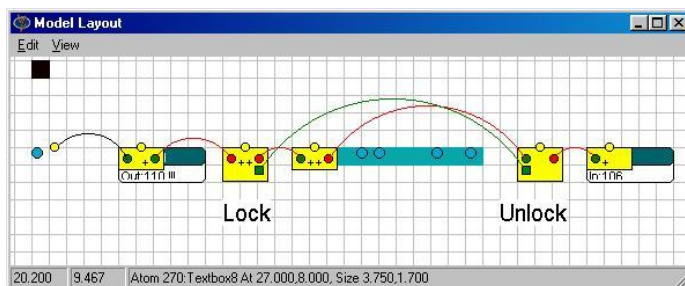
- *原子名称*
给予原子的名称。

- *图标*
在2D窗口中用来代表锁定原子的符号。
- *锁定水平*
当锁定和解锁之间的原子数目达到了此栏输入的具体水平时，便会关闭锁定原子的进入，因而阻塞了通道。当没有选择解锁原子时，输入通道关闭后将不会重新打开。
- *继续评估*
当选中这个选项时，每次产品进入锁定原子时都会评估锁定水平这一栏，而不是等到模型重启时才被评估。
只有当锁定水平包含了4DScript命令时 (不仅仅是一个数字)，才有必要启动这个选项。
- *批控制*
当达到锁定水平时，进入锁定原子的通道被阻塞，并且只有在路径中的所有产品通过解锁原子消失之后才会重新打开。这样便创建了一种形式的批处理，并将锁定水平作为批数量。

这个原子没有自己的设置，只能和锁定原子一起使用。每次产品离开解锁原子时，锁定原子便会允许一个新产品进入系统（除非“批控制”选项被启动）；在那种情况下，所有的产品都必须事先就通过了解锁原子）。

在下面的例子中，锁定原子和解锁原子控制着传送机上的产品数量。从中可以看到两个原子是怎样‘以一种正常的方式’放入到产品路径中的以及它们是怎样各自通过它们的第二个输出或输入路径连接起来的。

解锁原子中没有窗口需要填入。锁定原子可以和几个解锁原子连接在一起。



附录 3 4DScript 中的第一步

4DScript是Enterprise Dynamics的编程语言。在Enterprise Dynamics中执行的所有命令都是通过4DScript完成的。

这个文件为初学者提供了4DScript的结构和一个经常使用的命令列表，并有例子说明。

我们从句法规则，数学和逻辑运算入手。第二步，将用例子说明参考的概念。第三步是ED中最常见的命令。

1. 4DScript的基本概念

4DScript 的基本句法很简单并且在你输入4DScript 的任何位置都有效：

- 1 语言包含了一些词（命令）；
- 2 这些4DScript命令可以有(最多255) 参数；
- 3 参数放置在插入语()中；
- 4 参数可以为值，字符串或表达(其他4DScript 词条)；
- 5 参数之间总是要用逗号隔开；
- 6 如果参数为字符串，那么字符串参数总是要放入到方括号[]中。如果参数以4Dscript代码执行，那么参数就以正常的形式输入即可；
- 7 备注可以放在花括号{ }中间。

这些规则处处适用并且长期有效。然而，对于某些陈述而言这些规则并不能总是产生可读的代码。为了使得其他人能更容易地理解代码，一些 4DScript 句法在一些情况下可以例外。这在数学和逻辑符号中尤为适用。

数学符号

考虑一下有效的陈述：

一种更自然的表达：

+(12,7) 结果为 19

12+7

+/(100,10),6) 结果为 16 100/10+6 or 6+100/10

所有特殊的数学运算按照如下顺序进行：

*	=	乘法
/	=	除法
+	=	加法
-	=	减法

这些运算的正常优先原则仍然适用。在有疑问的情况下，使用插入语()!

逻辑运算

对于逻辑运算(比如 >, <, =)而言，可以不遵守‘先输入4DScript 命令再输入参数’的规则。

考虑一下有效的陈述：

>(10,6) 也可以写成‘正常的’的形式： 10>6
<=(23,12) 也可以写成‘正常的’的形式： 23<=12

这个例子可以延伸至更多的4DScript逻辑符号：

=	=	等于
>	=	大于
<	=	小于
>=	=	大于或等于
<=	=	小于或等于
<>	=	不等于

and

句法: `and(e1,e2)`

如果e1 and e2 为真则返回值1，否则返回值0。如果e1为假，那么 e2 不再被评估。用户可以不写成and(e1,e2)，而写成e1 **and** e2。

可见 **or**也同样如此。

所以，这两个命令可以照常使用...

max

句法: `max(e1,e2)`

返回e1 and e2的最大值。 同样可见**min**。

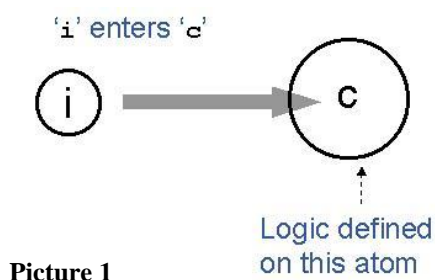
本节的末尾我们将讲述时间和时间转化的一些命令。记住ED是以秒来计量的，但是转化时间很简单：

时间 返回模型当时的时间，以秒计。
分(e1) 返回e1 (分) ， 以秒计： e1 乘以 60。
小时(e1) 返回e1 (小时) ， 以秒计： e1 乘以 3600。

2. 参考

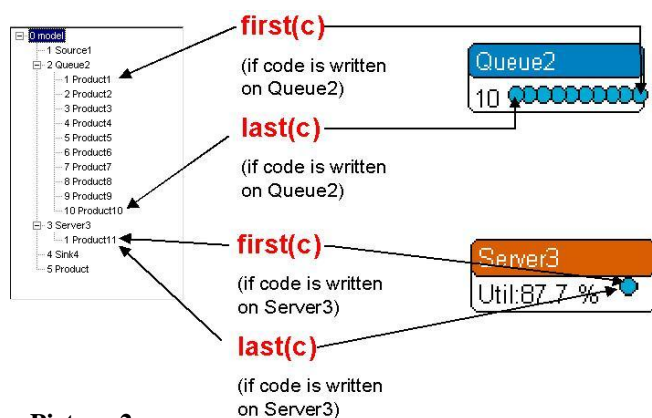
在ED 中一个十分重要的对象便是参考：如果我们讨论‘下一个’原子时，我们的观点是什么？如果有原子包含在其他原子中比如序列中的产品，那么情况会更加复杂。

我们在不同的陈述中看见了字母c and i，现在我们对它们给予说明：‘c’是指现在的原子，而‘i’指的是涉及到的原子，正在进入或离开现在原子的原子。



简单地说：‘i’ 只用于 Trigger on Entry（开始进入）或者 Trigger on Exit（开始退出）中！

理解参考这个概念中的关键可见下一个例子：



在图2 的左侧你可以看到一个简单序列系统的模型树，在某一个时刻处于冻结状态：

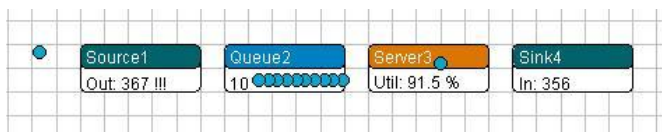


图 3

我们发现源原子，序列原子，服务器原子，接收器原子以及源原子之前的产品原子（模型树中的第五个，因为大多数情况下都是在你设置好并重启模型之后添加此原子的）都处于同一个层次。

序列2 中包含了十个产品，分别命名为产品1到产品10。与产品原子，源原子，序列原子，服务器原子和接收器原子相比，他们形成了第二个层次。当然产品1和产品5是位于同一层次的。服务器3中包含了一个产品，名称为产品11。

最高层次是模型本身。在图1中你在模型树中可以看到这一点！你能预测当你重启这个模型时有多少模型树会变化吗？

如果你理解了模型中的这个层次安排，你就可以理解参考的概念。

首先讲述一些在参考中经常使用的命令：

first(e1) 是指e1原子内部的第一个原子
last(e1) 是指e1原子内部的最后一个原子
next(e1) 是指与e1原子处于同一层次的下一个原子
prev(e1) 是指与e1原子处于同一层次的前一个原子

所以first(c)的意思是现在原子内部的第一个原子。如果现行原子是一个序列，它就可能是序列中的第一个产品。现在再次看图1以及关于*first* 和*last*的例子。

所以， first 和 last 通常处于陈述输入的原子的下一个层次！

关于next and previous的命令见如下说明：

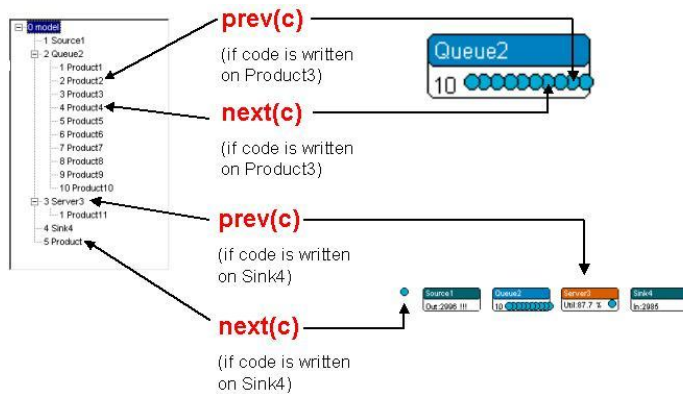


图 4

现在看图3和关于 *prev* and *next* 的例子。

所以, *prev* 和 *next* 通常处于陈述输入的原子同一个层次!

以下是一些常用的可用于流入控制并与路径相关的命令:

out(e1,e2) 是指与原子e2的输出路径e1相连接的原子

in(e1,e2) 是指与原子e2的输入路径e1相连接的原子

openoutput(e1) 打开原子e1的总输出

closeoutput(e1) 关闭原子e1的总输出。一旦关闭, 路径不能畅通, 不论个体用户对路径采用何种设置。

以相同的方式你可以定义 **openinput** 和 **closeinput**。

图5中有 *in* 和 *out* 的说明:

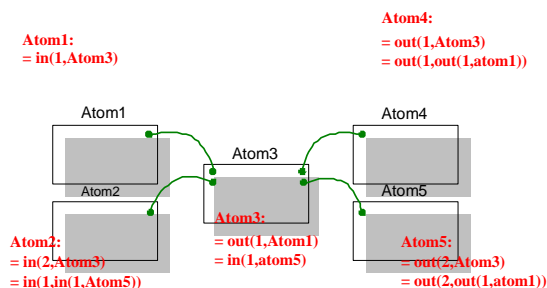


图 5

例 1:

所以将`in(1,c)` (写在原子3上, 观点)写入原子1, 但是将写在原子4上同样的陈述写入原子3!

例 2:

`content(in(3,c))` 返回与现行原子的输入路径3相连接的原子中包含的容量(原子数目)。

例 3:

`closeoutput(in(2,c))` 关闭与现行原子的第二个输入路径相连接的原子的输出。

注意命令嵌套的方式...

3. 重要的命令

在每一种编程语言中都有重要的命令, 执行着类似于标准块元件的功能。

3.1 条件语句: **if**

句法: `if(e1,e2 {,e3})`

如果e1为真(1)则执行e2; 否则执行e3 (如果规定的话)。 返回e2 或 e3的结果。

例 1:

`if(time>3600, msg[more than an hour], msg[less than an hour])`

意义为: 如果 `time>3600` 秒那么给出信息`more than an hour' (大于一个小时), 否则给出信息`less than an hour' (小于一个小时)

例 2:

`if(content(c)>10, closeinput(in(1,c))`

意义为: 如果现行原子的容量大于10则停止输入与第一个路径连接的原子, 否则不作任何处理。

不要着急一些使用过的命令, 比如`msg`, `content` 或者 `closeinput`。我们稍后作说明!

3.2 执行一个以上的语句: **do**

句法: `do(e1,e2,...,e25)`

依次执行e1, e2, 等命令。返回最后一个表达的结果。你最多可以使用25个参数。如果需要更多的参数, 几个do的循环可以嵌套在一起。

例子:

```
do( set(color(i),colorred),  
    set(icon(i),2),  
    setlabel([temperature],uniform[20,40],i)
```

所涉及原子(大多数是产品)中完成的三件事: 颜色被设置为红色, 图标改变为图标数字2并且
产品上印有含有名称`temperature` 的标签, 其值在20到40之间随机选择。

3.3 标签

使用标签是用户储存局域(暂时)变量的方式。它们大多数附加在产品上并且代表重量, 客户号, 生产时间等等。它们可以通过命令`setlabel` 来设置定义并通过`label`复制。

a. `setlabel(e1,e2,e3)`

定义原子`e3`上的标签, 其中`e1`是名称, `e2` 是标签的新内容。.
标签不需要在其录入为存在的时刻被创建。`Sddb` 命令功能相同。

例 1:

```
setlabel([Weight],10.24,i)
```

首先选择一个原子并用命令`setlabel`创建一个标签(并赋值)。.
假设标签名称为`v1`且值为100:

例 2:

```
setlabel([v1],100,animatom)
```

`label([v1],animatom)` 现在返回选中原子的具体值 (100)。

b. `label(e1,e2,{e3})`

返回对原子`e2`定义的名为`e1`的标签内容。标签不需要在其录入为存在的时刻被创建。标签名`e1` 通常是一个字符串并且敏感。

结果是字符串或是一个值取决于容量和`e3`: 如果`e3` 没有指定或者`e3=0`, 那么结果是一个值, 否则结果是一个字符串。如果`e3=1`, 那么结果通常都是一个值。如果容量不能转化为一个值那么值为0。如果`e3=2`, 那么结果通常都是字符串。如果你使用长的名称和许多标签时, 速度便会减慢。

3.4 以一些其他的重要命令结束:

age

句法: `age(e1)`

返回原子e1的年龄。年龄是现行时间与创建时间之间的差值。当运行重新设置时，所有原子的创建时间都变为0。

同样可见 **maxage, minage, avgage**

content

句法: `content(e1)`

返回原子e1的容量(包含的最高水平的原子数目)。

同样可见 **maxcontent, avgcontent**

avgstay

句法: `avgstay(e1)`

返回原子e1在总的运行时间之间的平均停留时间。

input

句法: `input(e1)`

返回原子e1的输入。输入被定义为至今为止已经进入原子e1的原子数目。相反的时原子的输出。

同样可见 **output**