

Curso de ProModel

Professores:

Diolino José dos Santos Filho

Paulo Eigi Miyagi

Newton Maruyama

1. Modelo

Segundo Seila [1995], um sistema é um conjunto de componentes ou entidades interagindo, isto é, os componentes trabalham juntos para atingir algum objetivo. Neste contexto, um modelo é uma abstração e representação simplificada do sistema. O modelo representa os componentes mais importantes do sistema e a forma como eles interagem. Um modelo estocástico é um modelo cujo comportamento não pode ser predito com valores fixos, mas que é sujeito a aleatoriedades.

A maioria dos modelos aproximados usa o paradigma do “entity-attribute-set” (conjunto de atributos da entidade), onde um sistema é considerado como sendo composto por entidades, as quais possuem atributos que são itens de informação a respeito da entidade.

2. Modelagem

Um modelo deve conter apenas os detalhes estritamente necessários para capturar a essência do sistema, dentro dos propósitos para o qual foi concebido, ou, em outras palavras, não se deve prender a detalhes desnecessários.

Para uma melhor modelagem, sugere-se a seguinte sequência de execução dos modelos:

- fazer uma análise do sistema a ser modelado e definir os objetivos que se quer alcançar com a simulação;
- definir o modelo;
- coletar os dados;
- construir, verificar e validar o modelo;
- analisar.

3. Simulação

De acordo com Seila [1995], o termo simulação é usualmente empregado para se referir à representação da dinâmica de um objeto ou sistema para análise de alguma atividade deste, considerada muito grande e/ou complexa.

Toda simulação se utiliza de um modelo para descrever o comportamento do objeto/sistema, que pode ou não existir e que é geralmente muito maior, custoso e complexo que o modelo. A idéia chave é que a simulação é uma realização alternativa que se aproxima do objeto/sistema e em muitos casos o propósito da simulação é analisar e entender o comportamento deste em função de ações e decisões alternativas.

A simulação é uma metodologia que não é específica para uma área em particular de aplicação, pelo contrário, pode ser aplicada para qualquer objeto/sistema que possa ser devidamente modelado.

Ela possui vantagens sobre outras alternativas (como por exemplo a análise matemática que é limitada a um certo número de objetos/sistemas cujo modelo matemático é conhecido). Como algumas de suas vantagens, pode-se citar: a capacidade de analisar modelos de complexidade arbitrária; a geração de resultados quantitativos e qualitativos para os responsáveis pela tomada de decisões; a flexibilidade do arranjo de métodos de simulação que pode ser usado para analisar sistemas estocásticos, etc.

Segundo Kelton [1995], a simulação de sistemas pode ser de dois modos: DIDO (entrada determinística – saída determinística: “deterministic input and output”) ou RIRO (entrada aleatória - saída aleatória: “random input and output”). Os sistemas que se comportam como o primeiro caso são chamados determinísticos e, não importa quantas vezes sejam simulados, os resultados serão sempre os mesmos. Já os que se assemelham ao segundo são chamados estocásticos e, quando simulados, fornecem uma “distribuição” de resultados onde se pode considerar um intervalo de confiança.

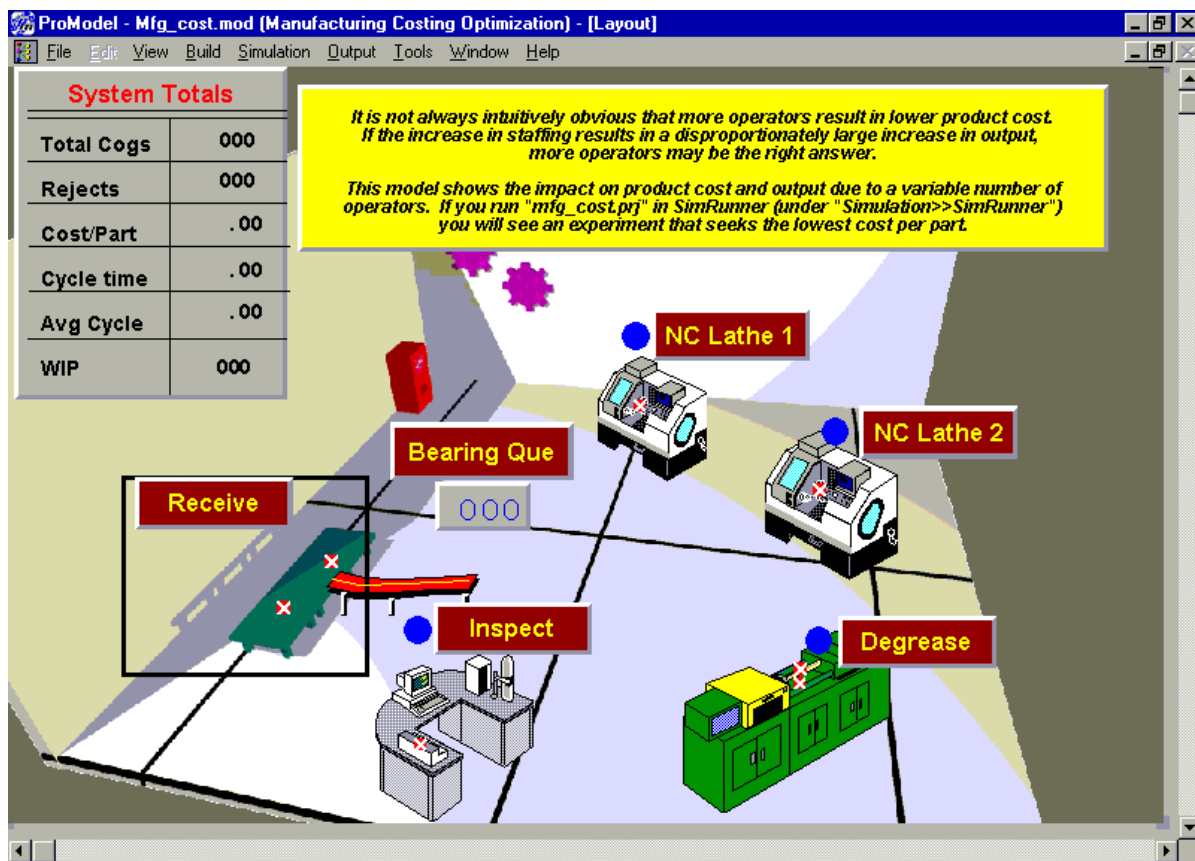
A simulação é a ferramenta adequada para se modelar um sistema no qual a variação estatística e a interdependência de seus elementos são tais que a programação linear e outros métodos de otimização não se mostram práticos ou aplicáveis.

4. ProModel

ProModel é uma poderosa ferramenta de simulação para modelar todos os tipos de sistemas de manufatura, abrangendo desde pequenos job shops e células de manufatura até produção em grande escala e sistemas de manufatura flexível. O ProModel é um software

desenvolvido para Windows com uma interface gráfica intuitiva e orientada a objeto, o que reduz em muito a necessidade de programação.

O ProModel permite o trabalho em grupo, possibilitando que diferentes partes do modelo sejam implementadas por diferentes grupos e posteriormente sejam agrupadas em um grande modelo final.



exemplo de modelo de simulação

5. Elementos de modelagem

O processo de se executar a modelagem de um sistema usando ProModel começa por definir o que o software denomina *elementos de modelagem*. A seguir, tem-se uma lista dos elementos possíveis no software, bem como, quando útil, uma referência ao sistema simples presente no tutorial que o acompanha, em que um operário utiliza em seqüência duas máquinas de usinagem para fabricar um tipo de peça:

- Locais:** local é um lugar imóvel de processamento ou armazenamento num sistema, para o qual *entidades* (ver mais adiante) são movidas para serem armazenadas, processadas ou

tomar novo roteiro. Os locais podem ser simples (comportam apenas uma entidade de cada vez) ou múltiplos (comportam mais de uma entidade simultaneamente). Os locais podem possuir regras de entrada e saída: as regras de entrada são usadas para selecionar qual a próxima entidade a ser manipulada caso exista mais de uma destas com essa possibilidade, ao passo que as de saída são usadas num local de capacidade múltipla para determinar a ordem com que deixarão o mesmo.

No exemplo do tutorial do ProModel, as máquinas de usinagem, o buffer de entrada, o buffer de saída e a esteira transportadora são modelados como locais.

- b) **Entidades:** entidade é um item, como um produto em fabricação, que é processado no modelo. A dinâmica conferida pelo ProModel às entidades permite que estas sofram operações cujos resultados são novas entidades, como reunião, divisão e conversão. As entidades podem receber *atributos* (ver mais adiante), que podem ser testados para a tomada de decisão ou para se obter estatísticas específicas. A imagem que representa uma entidade pode ser trocada como resultado de uma operação para se ilustrar uma mudança física da entidade durante uma simulação.

No exemplo do tutorial do ProModel, a peça fabricada e a matéria-prima desta são modelados como entidades.

- c) **Rotas:** rotas são elementos opcionais e definem o caminho utilizado por entidades e recursos para se moverem no sistema. As rotas são constituídas de nós conectados por segmentos (que são definidos graficamente através de simples cliques de mouse) e de interfaces desses nós com locais. Várias rotas podem ser definidas, e várias entidades e/ou recursos (ver mais adiante) podem compartilhar a mesma rota. Os movimentos de entidades e recursos ao longo da rota podem ser definidos em termos de comprimento da mesma e velocidade de movimento, ou simplesmente pelo tempo de percorrimto da mesma. As distâncias são automaticamente computadas baseando-se na escala de layout definido pelo usuário, mas podem ser redefinidas manualmente.

- d) **Recursos:** recursos podem ser pessoas, ferramentas, veículos ou qualquer outro objeto que possa ser usado para transportar materiais entre dois locais, realizar uma operação sobre um material em um local, ou realizar manutenção em um local ou em outro recurso que esteja quebrado. Recursos podem ser dinâmicos ou estáticos. O que difere estes dois tipos de recursos é o fato de o primeiro ser vinculado a uma rota e o segundo não. Regras

de decisão podem ser utilizadas para alocar os recursos e priorizar os carregamentos e entregas. Características de movimentação dos recursos, como velocidades quando cheio e quando vazio, aceleração, desaceleração, tempo de carga e descarga, entre outros, podem ser especificados.

No exemplo do tutorial do ProModel, o operário é modelado como um recurso.

- e) **Processos:** a lógica de um processo define qual a operação e o roteamento para cada tipo de entidade em cada local do sistema. Os tempos de operação ou serviço, requisições de recursos, lógicas de processamento, relações de entrada e saída, condições de roteamento e tempos de movimentação podem ser descritos. Os tempos de operação podem ser descritos por formas variadas como constantes, distribuições probabilísticas, resultados de funções, valores de atributos, resultados de sub-rotinas, etc., ou por uma expressão contendo uma combinação destas formas. Operações lógicas como IF-THEN-ELSE, malhas (loopings) e chamadas a subrotinas podem ser incluídas. Expressões relacionadas a recursos como GET, USE, JOINTLY GET juntamente com expressões booleanas e palavras reservadas como ACCUM, JOIN, GROUP simplificam a lógica de programação e utilização dos recursos.

- f) **Chegadas:** Neste elemento é definido o mecanismo de determinação dos momentos em que uma entidade é introduzida no sistema. uma chegada indica como um entidade é introduzida no sistema. Um registro de chegada é composto de: número de novas entidades por chegada, frequência das chegadas, locais das chegadas, o instante da primeira chegada e o número total de ocorrências de chegadas. Chegadas podem ser determinísticas, condicionais ou estocásticas; neste último caso, podem-se utilizar tanto distribuições probabilísticas pré-definidas quanto definidas pelo usuário para definir os horários e as quantidades das chegadas.

- g) **Horários de trabalho:** Este elemento, determinando horários de funcionamento e paradas agendadas do sistema, é definido pela seleção de horas no dia e dias na semana. A cada horário de trabalho podem ser associados recursos e locais.

- h) **Atributos:** Entidades e locais podem receber atributos de valor inteiro ou real. Pode-se definir atributos para entidades e locais e podem assumir tanto valores inteiros como reais. Nomes de locais, recursos e entidades podem ser associados a atributos. Estes

atributos podem ser relacionados aos do E-MFG e são definidos para uma família de entidades, por exemplo, possuindo cada uma os seus respectivos atributos. Os atributos são geralmente empregados para representar características das entidades tais como cor, número de vezes que foi retrabalhada, tempo de operação da entidade para locais específicos, etc. Tanto para variáveis como para atributos é possível incrementar, decrementar ou executar qualquer operação matemática, com a diferença de que os atributos estão associados a entidades e locais específicos. (REESCREVER)

- i) **Variáveis:** As variáveis são usadas para tomadas de decisão e cálculo de estatísticas. O valor da variável pode ser monitorado a todo momento e mostrado ao fim da simulação em forma de gráfico ou histograma. Estas, assim como os atributos, podem assumir valores reais ou inteiros. Quando se utiliza um contador, é necessário definir uma variável e vincular o contador a esta.

- j) **Vetores / matrizes:** São matrizes de variáveis, podendo ser uni ou multidimensionais.

- k) **Macro:** É uma expressão complexa, ou um conjunto de expressões, que pode ser definida apenas uma vez e utilizada depois várias vezes. Macros são extremamente úteis quando pedaços de lógica se repetem em vários pedaços do modelo. (As macros assemelham-se à utilização da palavra reservada “#define” na linguagem C de programação. Em C encontraríamos: #define MAX 5.) Pode-se definir uma macro como uma interface de tempo de execução (**RTI Macro**); ao se definir uma macro como RTI, o conteúdo de seu campo **Text** torna-se automaticamente o valor da lógica que a mesma está substituindo. As macros RTI permitem ao usuário trocar facilmente os parâmetros do modelo antes de executá-lo, além de permitir a preparação de cenários múltiplos para serem executados em lotes. As RTI podem ser acessadas através das opções **Model Parameters** ou **Scenario** existentes dentro do menu de simulação.

- l) **Subrotinas:** São blocos definidos pelo usuário que recebem valores quando chamadas e podem retornar resultados após sua execução. Operações complexas desenvolvidas em várias partes do modelo podem ser definidas apropriadamente por uma única subrotina.

- m) **Distribuições do usuário:** Estas podem ser definidas pelo usuário para substituir as distribuições probabilísticas fornecidas pelo software.

Para uma visão prática do significado de várias das definições acima, recomenda-se seguir o tutorial do ProModel, que leva aproximadamente uma hora para ser executado.

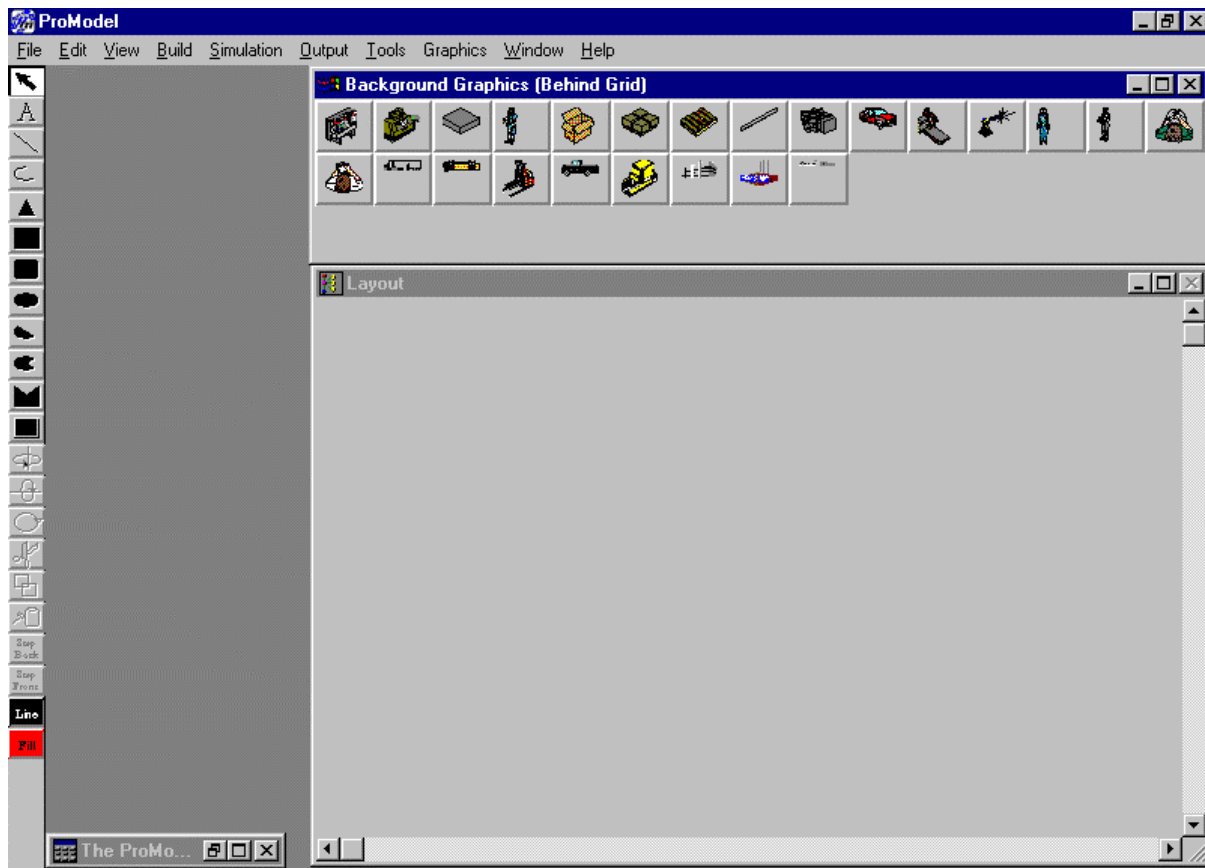
6. Instruções para geração de modelo

Tem-se a seguir instruções detalhadas para a criação de um projeto, do modelo e de seus elementos:

6.1. Criar novo projeto:

Acessar o menu **File** e selecionar **New**. O software criará um novo projeto e abrirá a janela de informações gerais (**General Information**) para que se defina:

- título do modelo;
- padrões de unidades adotadas (tempo e distância);
- qual biblioteca gráfica será adotada;
- cor de fundo do layout;
- escala e cor o reticulado (grid) no layout;
- pano de fundo (se for o caso), importado ou criado a partir do editor gráfico do software.



janela para edição de cenário

6.2. Definir locais (locations):

Para se inserir um local no modelo, basta selecionar um elemento na janela **Graphics** e em seguida clicar na janela **Layout**. O objeto será inserido na janela, podendo ser redimensionado, e simultaneamente será inserido um novo elemento na tabela **Locations**.

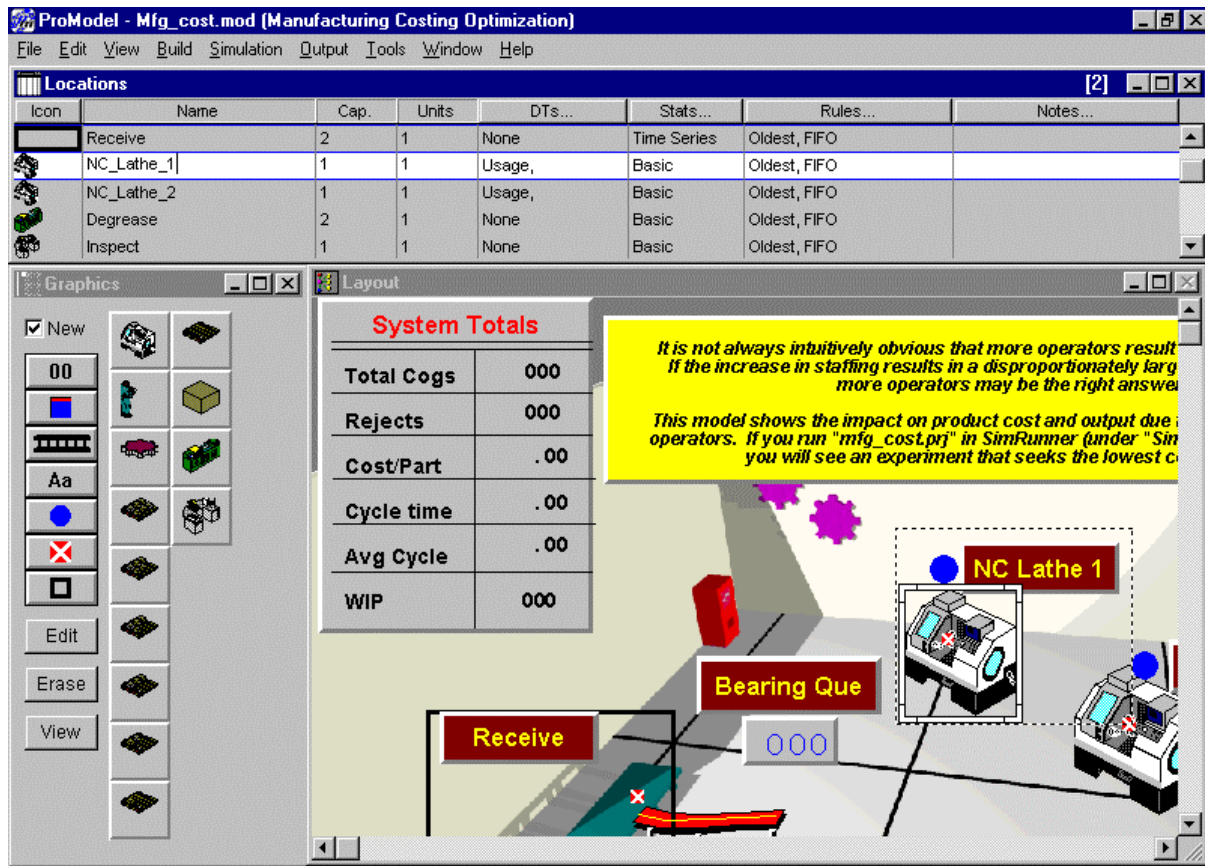
Caso seja necessário apagar um lugar, selecione o lugar na janela de **Locations**, selecione **Edit** no menu e em seguida **Delete** (este procedimento é válido para todos os elementos do ProModel).

Nesta tabela pode-se:

- redefinir o nome do local (**Name**);
- qual a capacidade que cada local comporta (**Cap**);
- definir o número de réplicas da unidade local em questão (**Units**);
- determinar os tempos mortos (**DT**) – tempos mortos são usados para simular falhas aleatórias de equipamentos, manutenção de rotina, ajustes, etc., e podem ser definidos tanto para os locais (locais de capacidade múltipla só aceitam tempos mortos definidos por **clock**) quanto para os recursos. Como parâmetros, define-se: frequência com que

estes tempos mortos ocorrem, o primeiro momento em que um deles ocorre, a prioridade, lógica (se houver), e se este está ativado ou não;

- definir o tipo de estatística a ser coletado (**Stats**);
- definir as regras de decisão (**Rules**).



janela de edição de locais.

6.3. Definir entidades (entities):

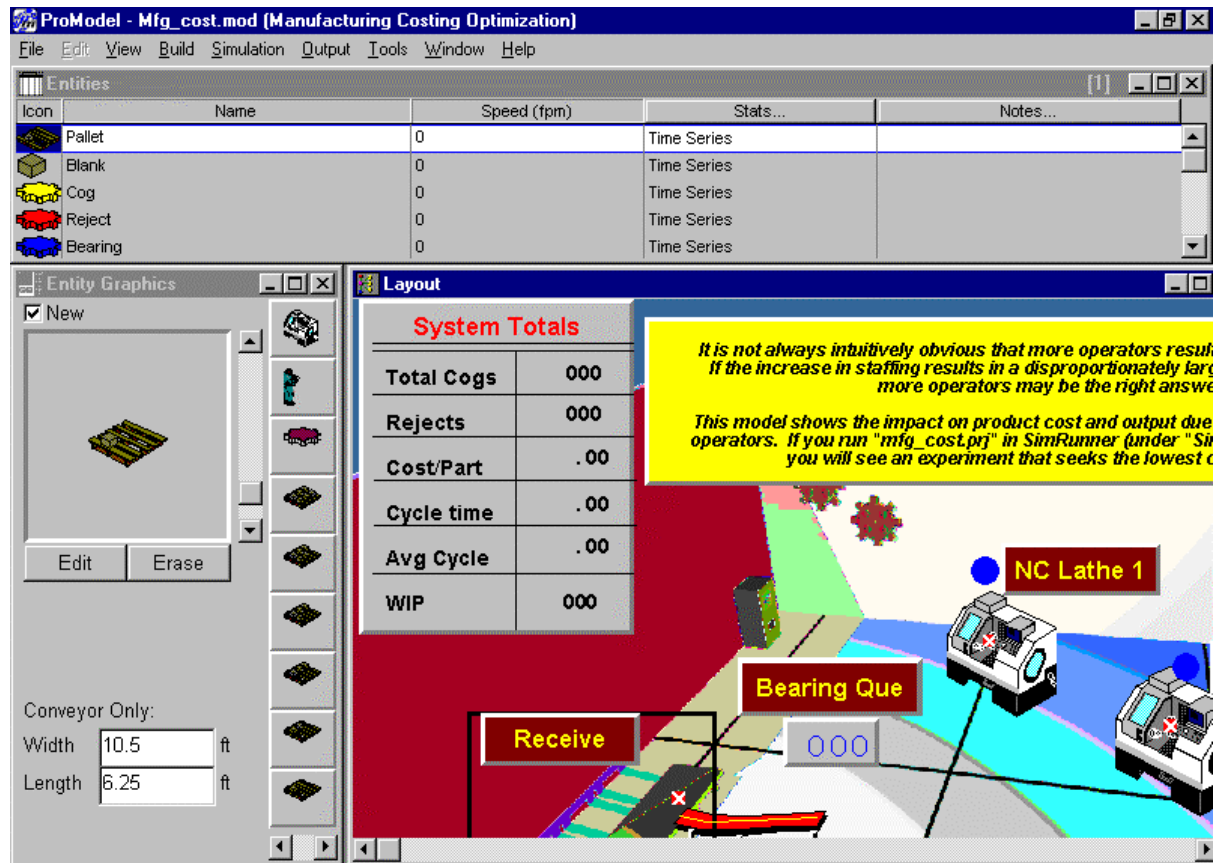
Para se inserir uma entidade, basta selecionar um elemento na janela **Entity Graphics** e aparecerá uma imagem no quadro de edição da mesma janela e um nome correspondente a este elemento na janela **Entities**. Observação: aqui as entidades são apenas definidas; elas só aparecem na simulação quando inseridas através da tabela de chegadas (**Arrivals**).

Na janela **Entities** é possível definir:

- nome da entidade (**Name**);
- escolher a velocidade com a qual a entidade se movimenta no sistema (**Speed**);
- escolher o tipo de estatística que se deseja coletar ao final da simulação (**Stats**).

Na janela **Entity Graphics**, pode-se:

- editar a imagem (botão **Edit**);
- criar um vetor de figuras desmarcando-se a opção **New** (canto superior esquerdo da janela).



janela de edição de entidades

6.4. Definir rotas (path networks):

As rotas são os elementos de mais fácil construção; basta clicar (não é necessário manter o botão pressionado) sobre um ponto desejado da janela **Layout** e movimentar o mouse. Dando um clique é possível mudar a direção do segmento de reta e, com um duplo clique, inserir o nó final – como consequência surgirá um segmento de rota biorientado (flechas em ambas as pontas). Pode-se observar que surge a indicação deste segmento de rota na janela **Paths**, com origem, destino, comprimento (que é determinado automaticamente com base na escala pré-estabelecida para o **Layout**, mas pode ser editado manualmente) e se é uni ou bidirecional.

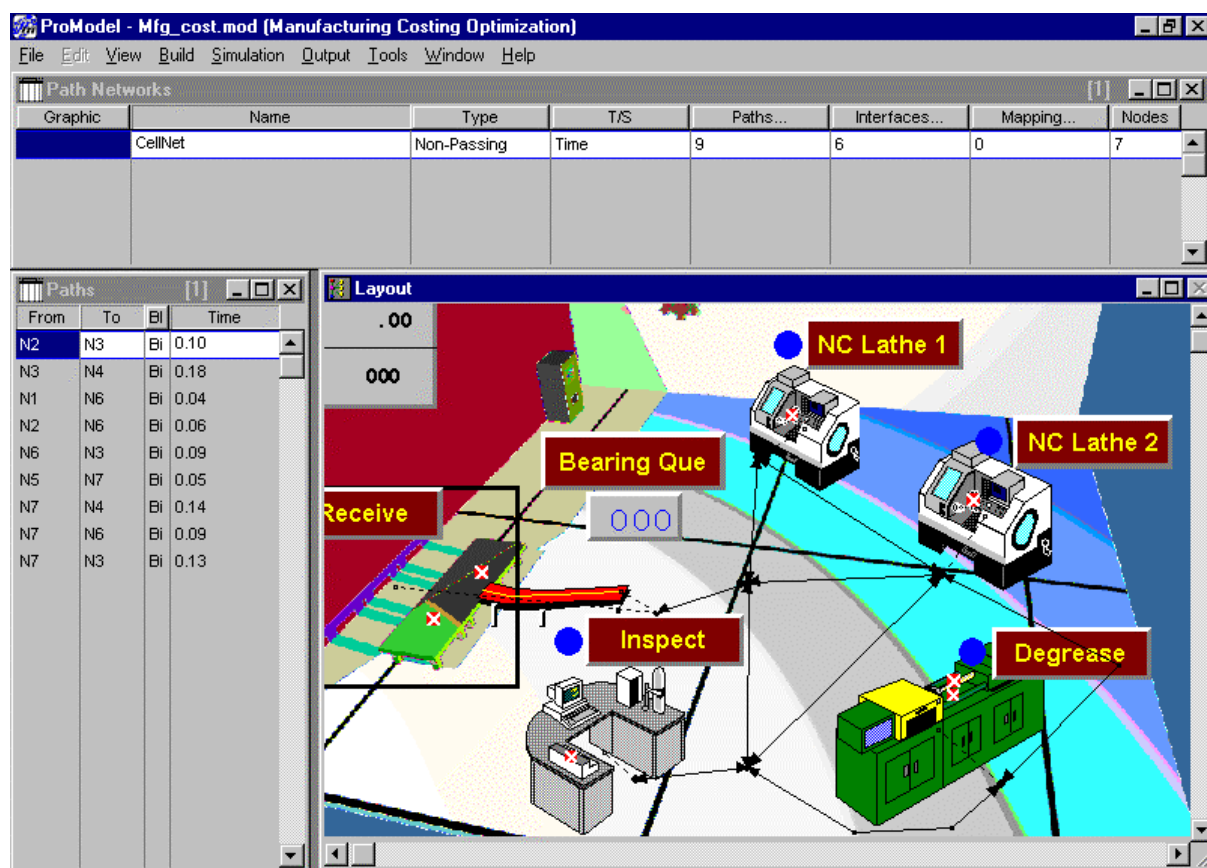
Na janela **Path Networks** pode-se definir:

- cor da rota (**Graphic**);

- nome da rota (**Name**);
- tipo (se é passante ou não) (**Type**);
- se o tempo de passagem é calculado baseado num valor de tempo ou em valores de velocidade e distância (**T/S**).

Deve-se ter em mente que rotas são inúteis sem a criação de interfaces, que são a conexão entre os locais e as rotas.

Para se criar uma interface, basta selecionar **Interfaces** dentro da janela **Path Networks**, dar um clique sobre um nó da rota e em seguida clicar sobre o local ao qual se deseja relacionar o nó.

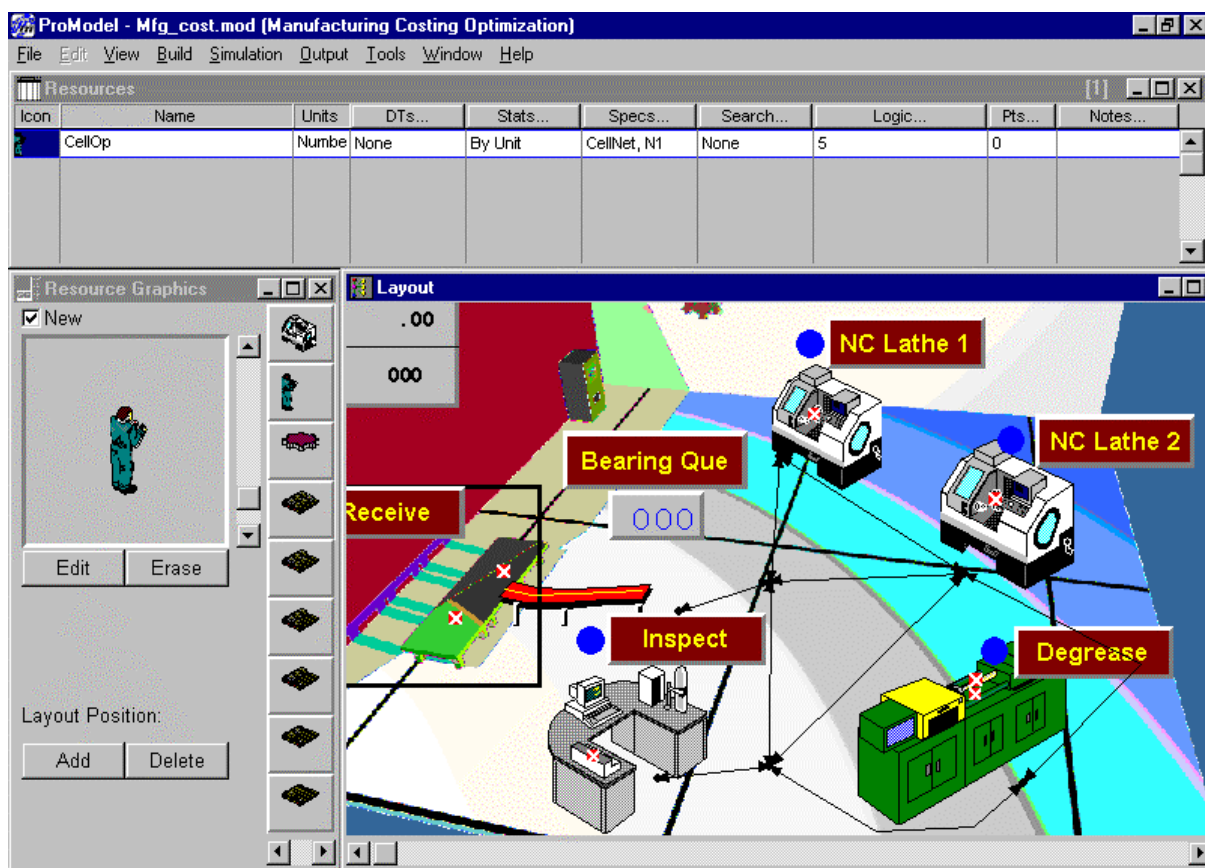


janela de edição de rotas e interfaces

6.5. Definir recursos (resources):

Para se inserir um recurso deve-se proceder da mesma forma do que para as entidades, havendo também a possibilidade de se editar os elementos gráficos. Ao se inserir um novo recurso, pode-se, na janela **Resources**:

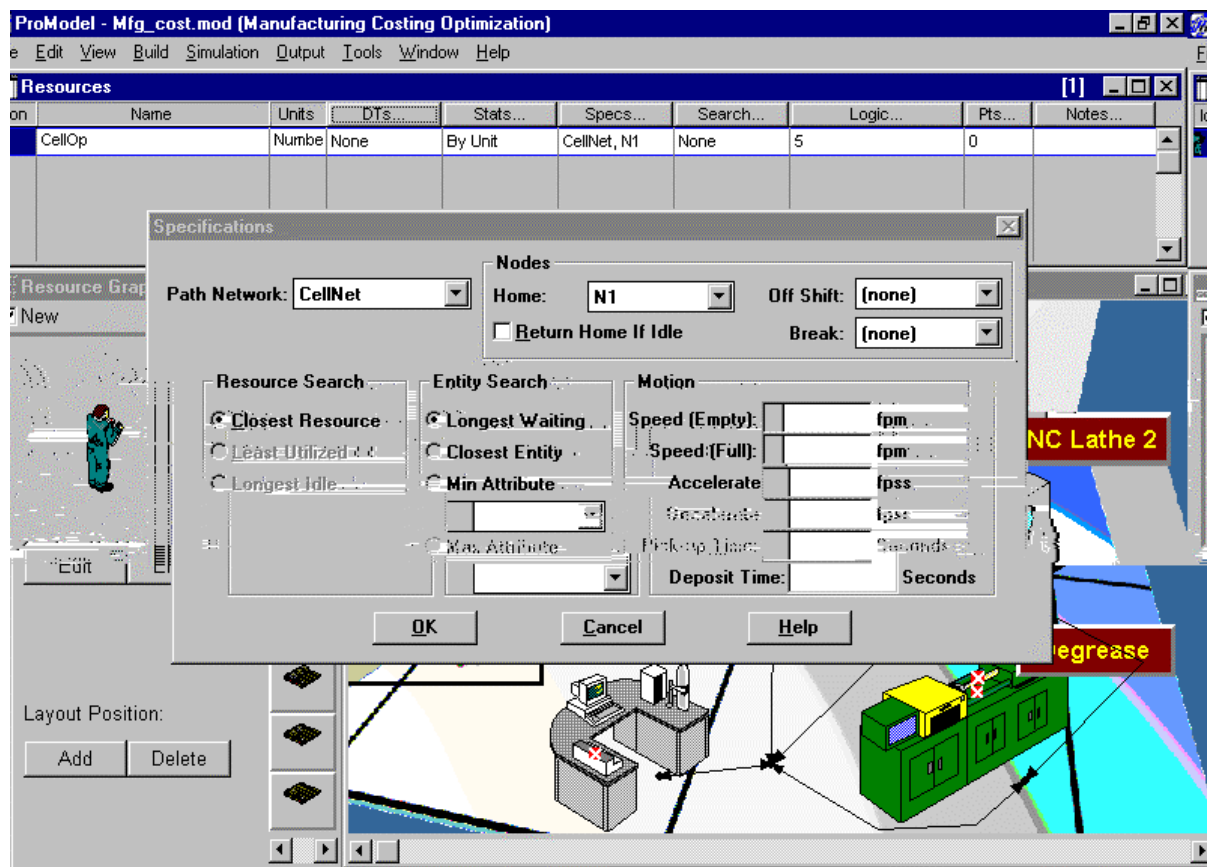
- mudar o nome do recurso (**Name**);
- definir a quantidade de recursos (**Units**);
- definir paradas / quebras (**DTs**);
- definir tipo de dados estatísticos serão coletados durante a simulação (**Stats**);
- definir características do recurso (**Specs...**);
- definir busca: **Work Search** (busca de trabalho – trata-se de uma lista de locais onde os recursos irão buscar trabalho tão logo ele deposite a entidade que ele estiver processando) ou **Park Search** (busca para estacionar – uma lista de nós onde o recurso pode estacionar tão logo ele deposite a entidade que ele esteja carregando. Esta só é realizada após todas as buscas de trabalho terem sido esgotadas);
- definir alguma lógica para o recurso (**Logic**), como, por exemplo, o elemento gráfico que será utilizado para representar o recurso.



janela para inserção e edição de recursos

Selecionando-se o item **Specs...** pode-se definir ainda:

- se o recurso é estático ou dinâmico e em que rede de caminhos ele trafegará se for dinâmico (**Path Network**);
- em que nó o recurso inicia as suas atividades (**Home**), para onde ele deve se dirigir fora do seu horário de trabalho (**Off shift**) e para onde deve se dirigir quando estiver quebrado e necessitar de parada (**Break**);
- lógica para escolha do próximo recurso desse tipo a ser alocado: *o mais próximo, o menos utilizado* ou *o que está a mais tempo ocioso*;
- lógica para escolha da próxima entidade a ser atendida: a que está *esperando há mais tempo, a mais próxima, a de menor atributo* (escolhido à parte) ou a de *maior atributo* (idem);
- características do recurso: *velocidade sem carga, velocidade com carga completa, aceleração, desaceleração, tempo para carregar e tempo para descarregar.*



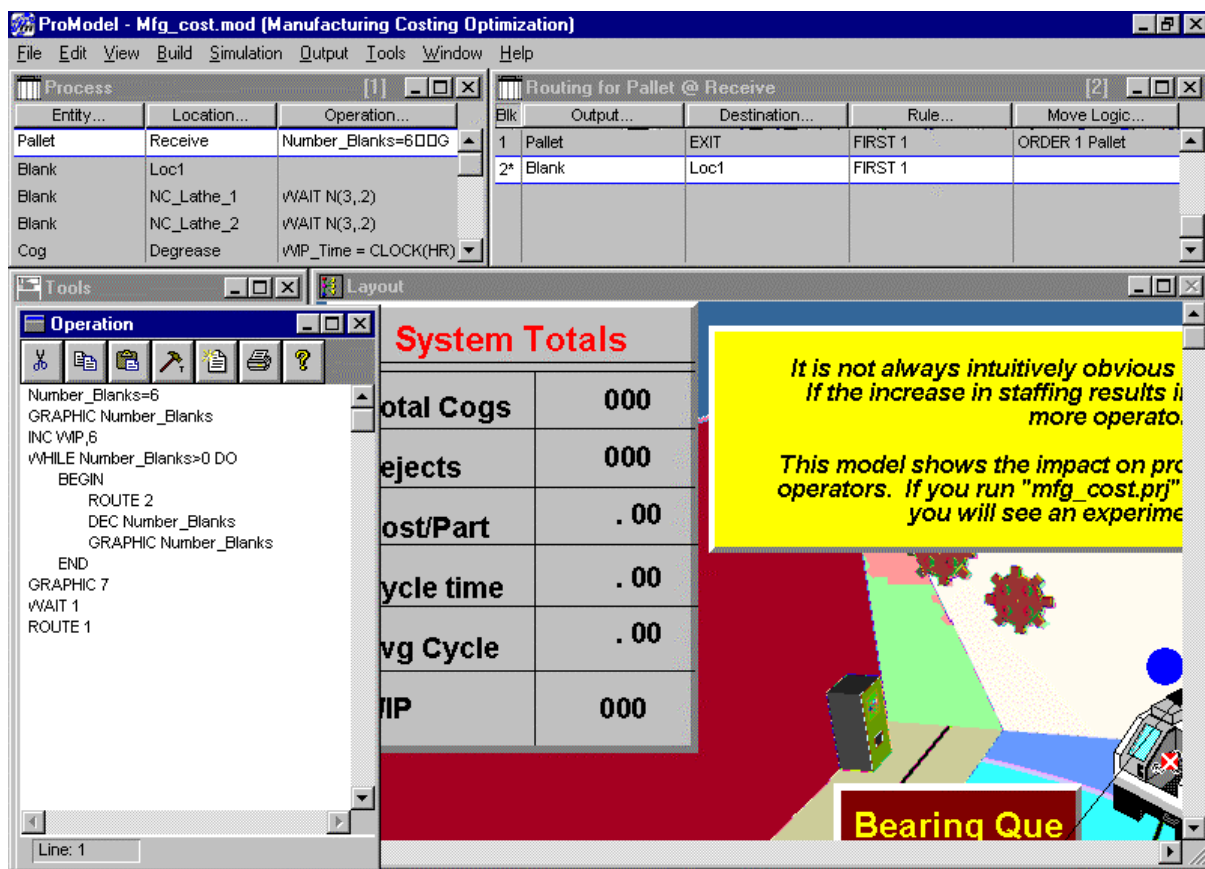
janela de edição das propriedades dos recursos

6.6. Definir processos (processing):

O ambiente para definição de processos é composto por quatro janelas: **Layout**, **Process**, **Tools** e **Routing for X**, onde X é o nome do campo selecionado na janela **Process**.

Na janela **Process**, têm-se as opções:

- **Entity**: seleciona a entidade que sofrerá a ação (todos os processos são definidos para as entidades);
- **Location**: é o local onde a entidade sofrerá a ação;
- **Operations**: pressionando-se este botão, abre-se uma janela de programação onde se define o que deve ser feito com a entidade no lugar especificado.



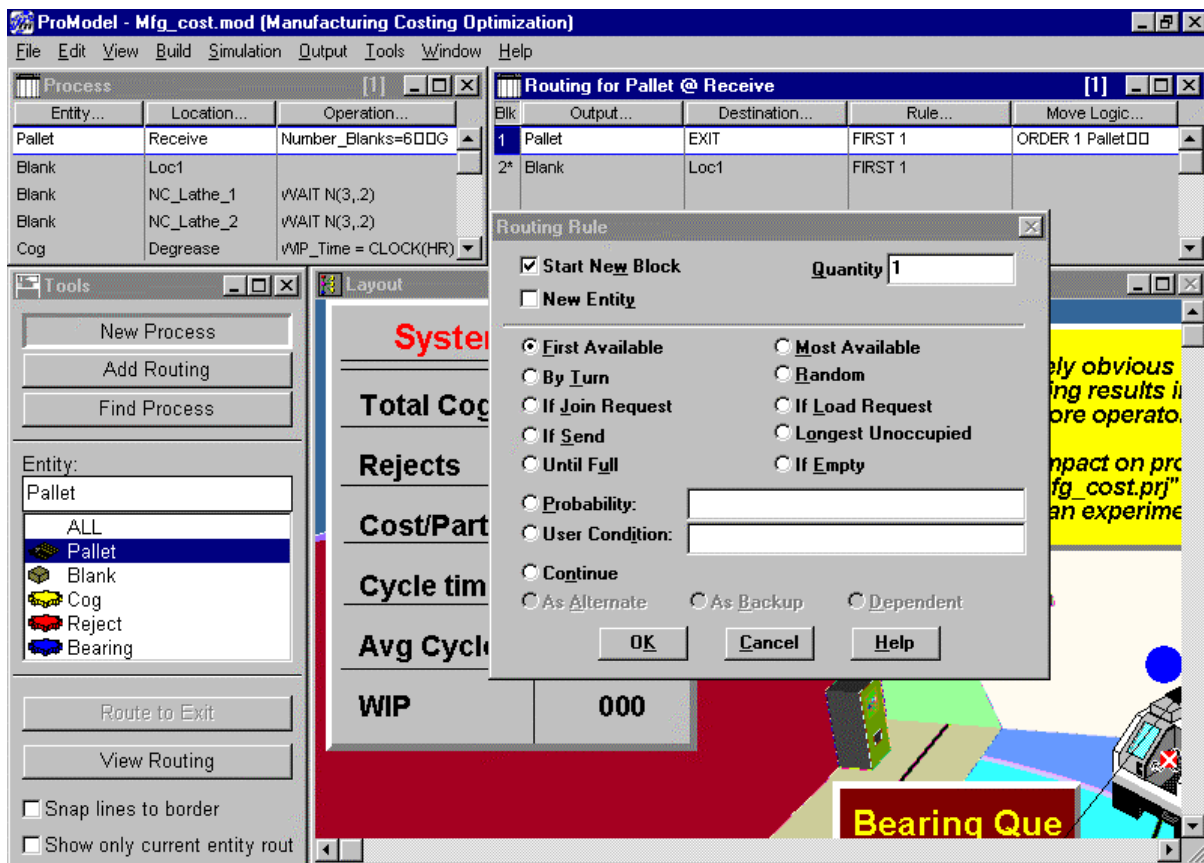
exemplo de operação

Na janela **Tools**, seleciona-se a entidade para a qual se pretende definir uma lógica.

Na janela **Routing for X**, define-se:

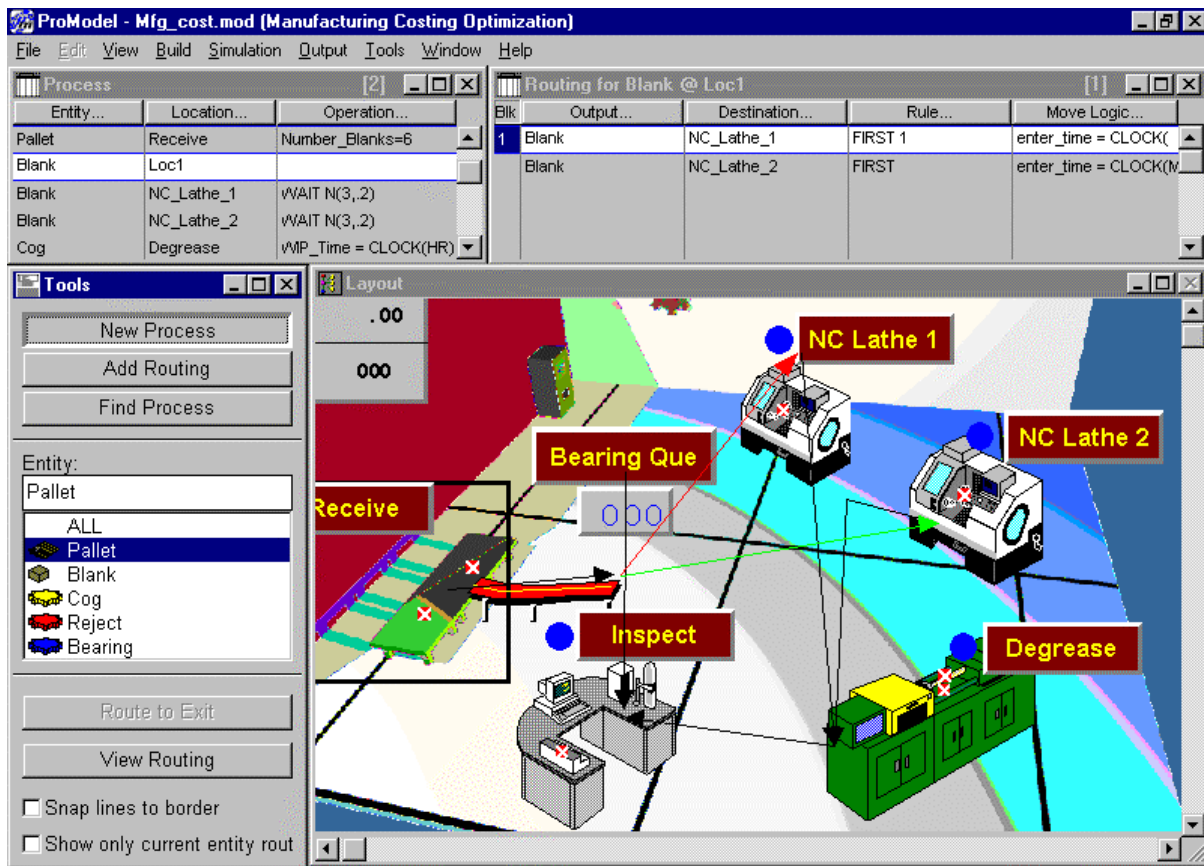
- **Output**: o tipo de entidade que se obtém ao término da operação;
- **Destination**: o local de destino para onde seguirá após a operação a entidade obtida;

- Rule:** regra para escolha do local de destino (uma mesma entidade pode ter vários destinos de acordo com seus atributos, por exemplo). A janela de regras de roteamento (**Routing Rules**) fornece uma lista de regras e ramificações pré-definidas que a entidade pode selecionar para seguir até o próximo local dentro do processo estabelecido. A regra de roteamento **First Available** faz com que seja selecionado o primeiro local disponível dentro do bloco de roteamento de saída. Trata-se também da opção “default” caso haja apenas um local de destino. A opção **Start New Block** indica a formação de um novo bloco de roteamento e a opção **Quantity** se refere ao número de entidades que aparecerão no local de destino selecionado (quantidades maiores que um são usadas quando desfazemos lotes ou quando são criadas entidades adicionais);

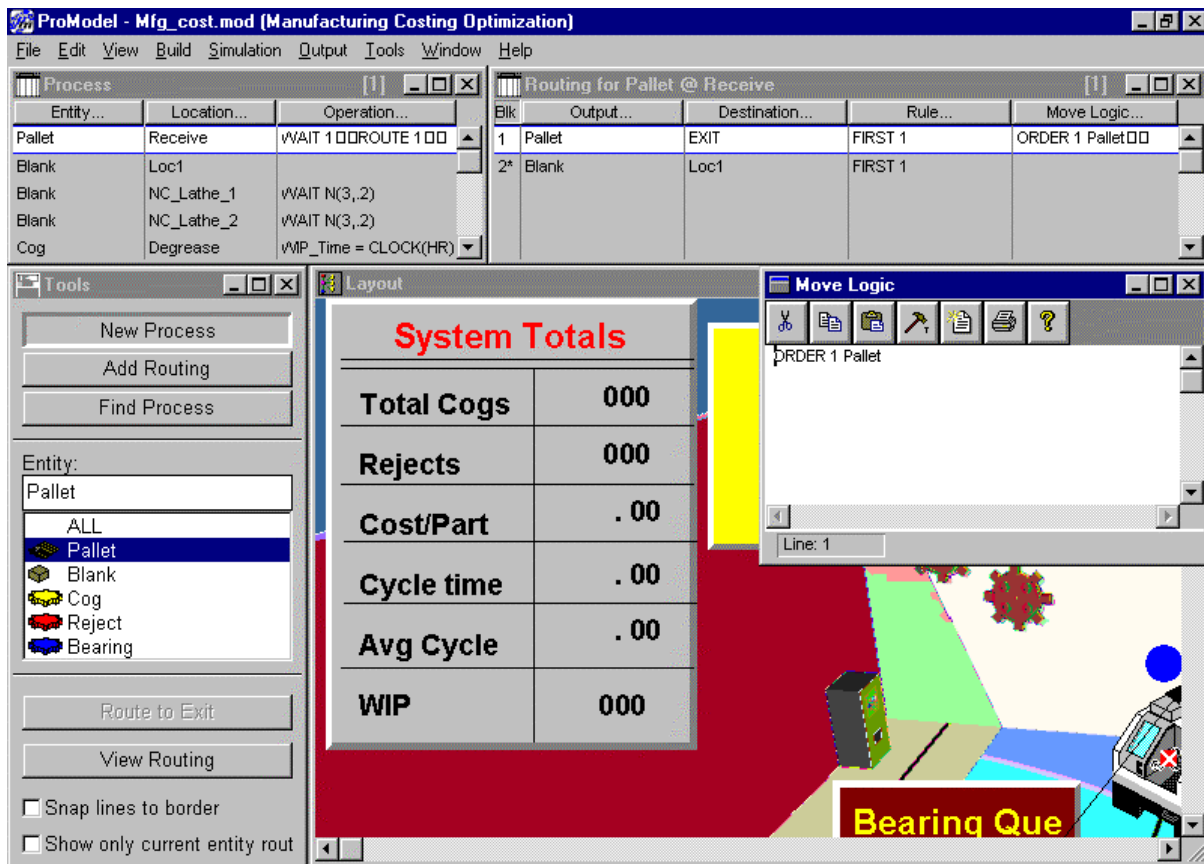


exemplo de regra de roteamento

- Move Logic:** contém as regras de movimentação da entidade; qual recurso utilizar, qual o elemento gráfico que a representará, qual o tempo de movimentação, se ela utilizará o recurso e o reterá ou se o liberará para outra atividade, etc.



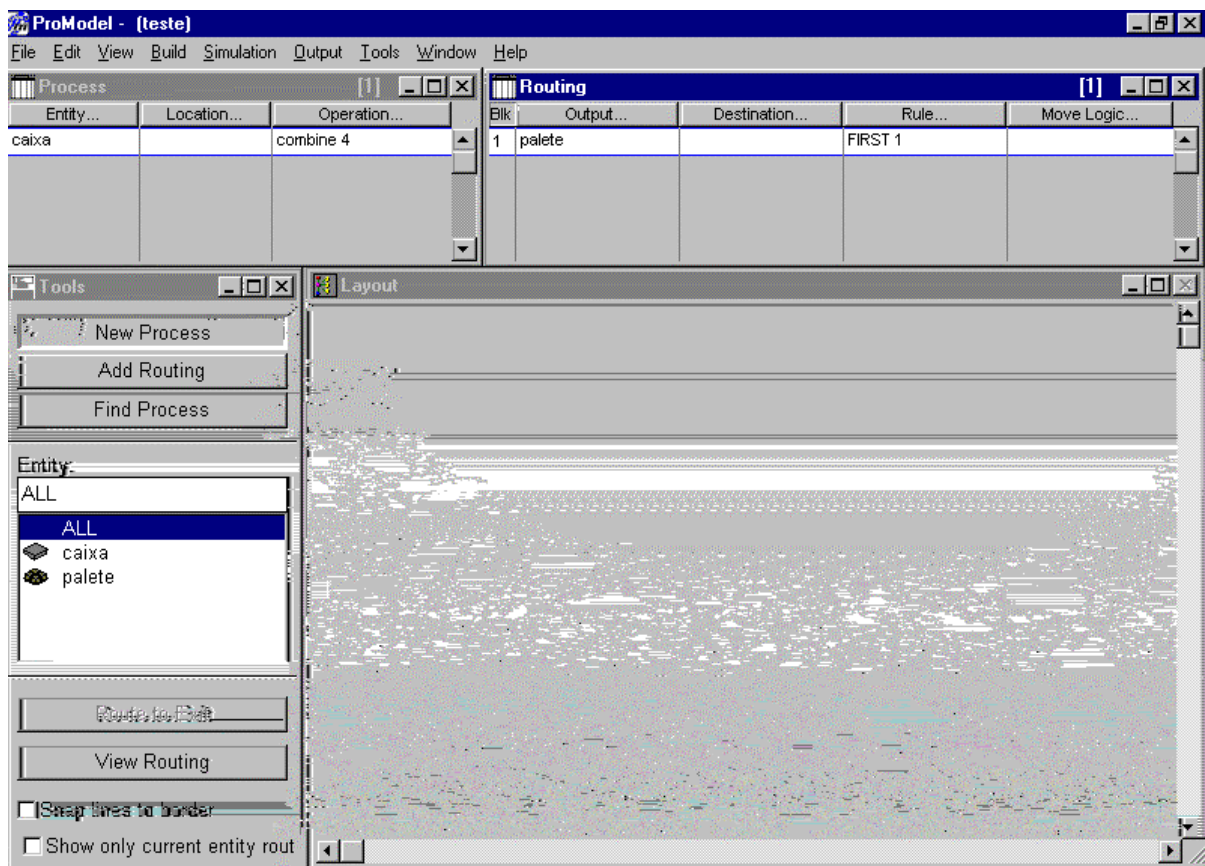
janelas para definição dos processos



exemplo de comando em Move Logic

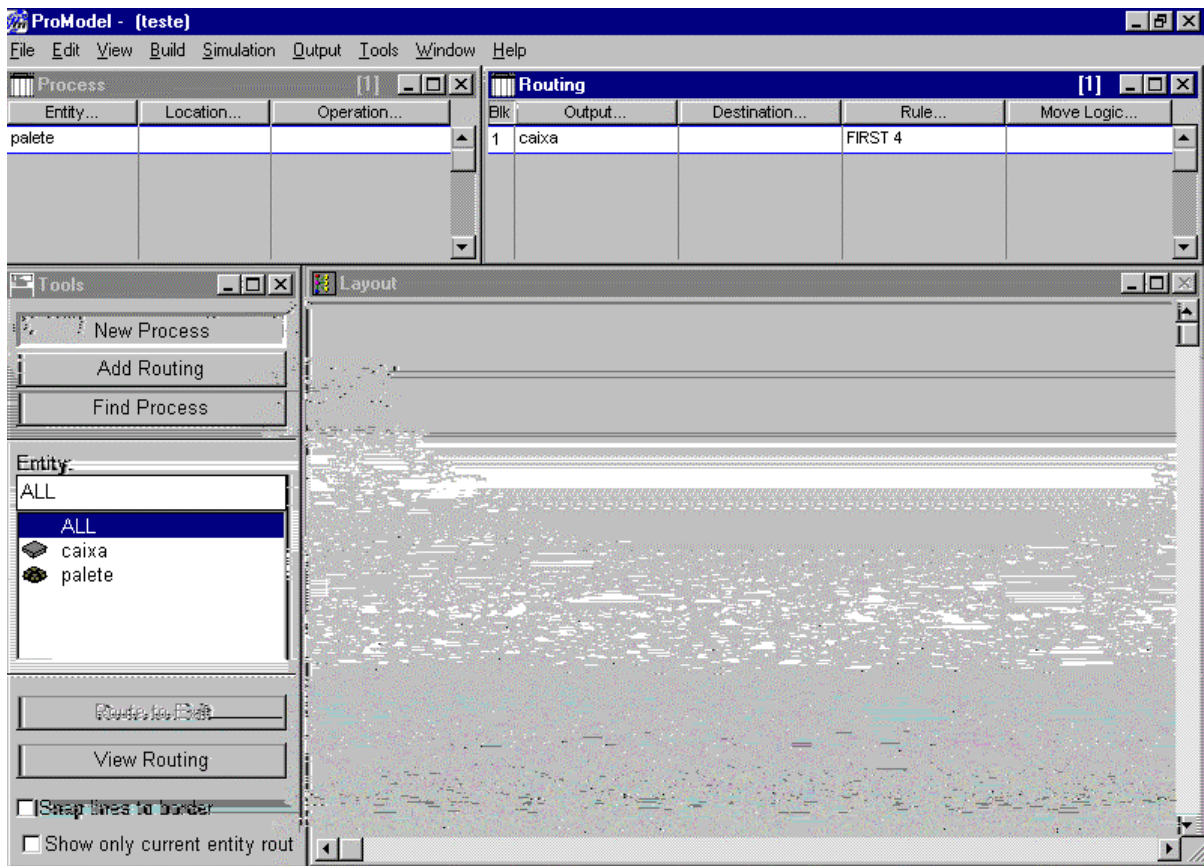
Seguem-se alguns comandos e construções úteis para se programar os processos no ProModel. Com exceção da construção para desfazer lotes e de um dos usos do comando **join**, todos eles devem ser utilizados na janela **Operations** :

- **combine**: usado para fazer lotes (paletização) e serve para juntar várias entidades e transformá-las em uma só (seja por algum tempo ou permanentemente). Um exemplo é combinar 4 caixas para formar um palete:



exemplo do uso do comando “combine”

- *desfazer lotes*: ao invés de se utilizar um comando, basta especificar no campo **Rule...** a quantidade de entidades que sairão do local:



exemplo de desmanche de lote

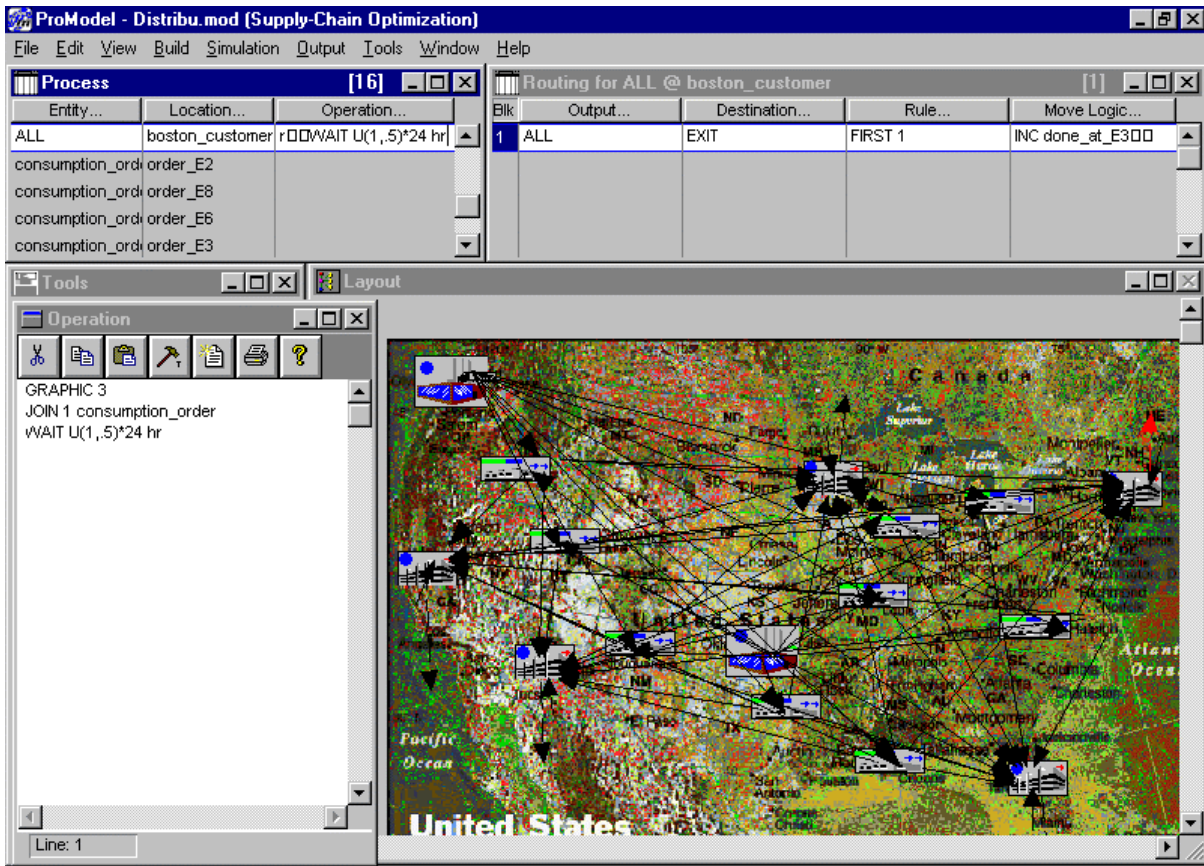
- **INC / DEC:** usado para incrementar e decrementar variáveis respectivamente. Para incrementar utiliza-se o comando **INC variável, valor do incremento** e para decrementar, **DEC variável, valor do decremento** (valores de incremento e decremento iguais a 1 podem ser omitidos):

INC x é o mesmo que se escrever $x = x + 1$

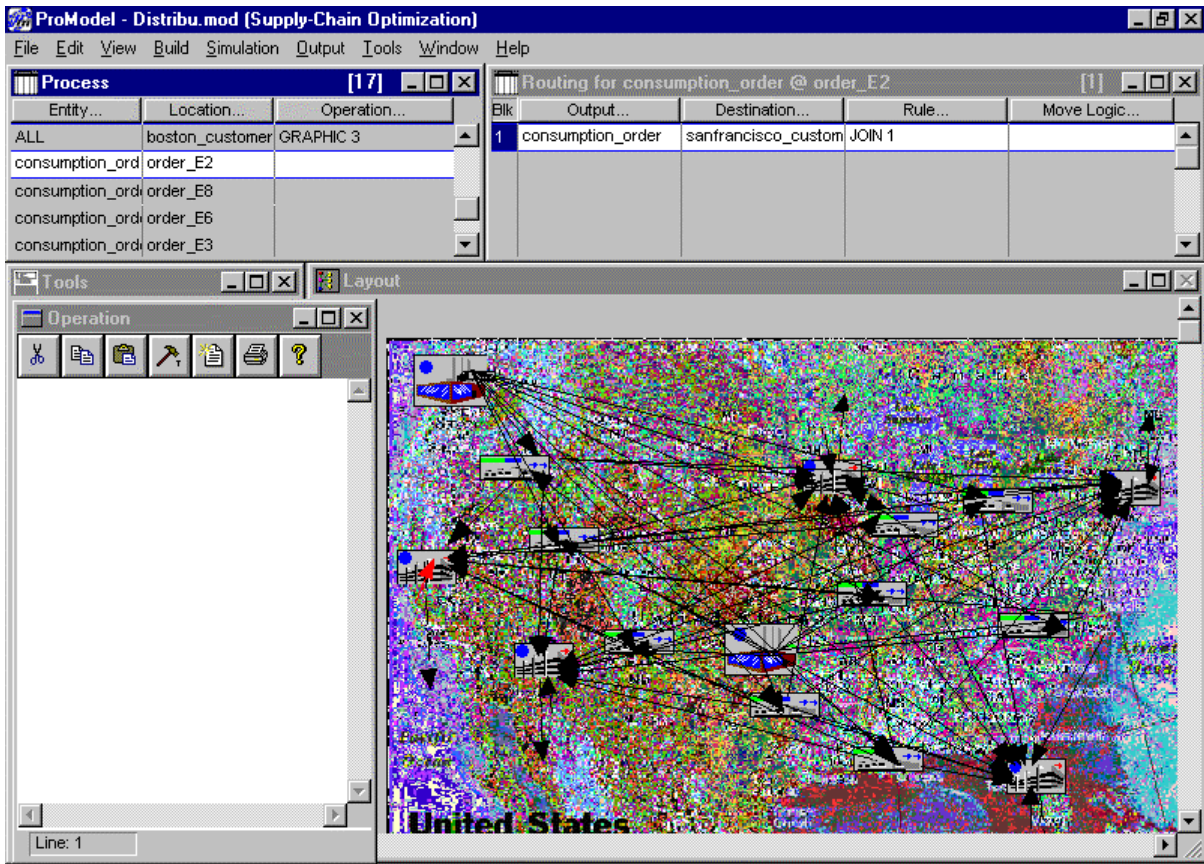
DEC x é o mesmo que $x = x - 1$

INC x, 2 é o mesmo que $x = x + 2$

- **join:** comando usado para unir duas ou mais peças. Ele deve ser usado no campo **Operation** para a entidade que está solicitando as outras entidades e no campo **Rule** para as entidades a serem unidas:



exemplo de uso do comando “join” para o requisitante da união



exemplo de uso do comando “join” para a peça requisitada

- **get / free**: usados respectivamente para capturar e liberar um recurso, no caso em que várias linhas de lógica precisam ser executadas desde o instante em que o recurso é capturado até o instante em que ele é liberado:

ex: **get** *recurso*
 wait 3 min
 free *recurso*

- **use**: usado para capturar um recurso por um período específico de tempo, após o qual ele é automaticamente liberado:

Ex: **use** *recurso* **for** 3 min

- **if- then**: é semelhante em sintaxe ao encontrado em PASCAL, com a diferença de que os comandos não são separados por “;” e sim escritos um em cada linha. A sintaxe segue o padrão IF <expressão booleana> THEN <processo>

Ex1: **IF** var1 = 5 **THEN** **wait** 2 min

ex2: **IF** attr1 <> 4 **OR** var1 > 0 **THEN**

```

BEGIN
var1 = 5
wait 2 min
END

```

- **if-then-else**: idem ao anterior só que no lugar de **BEGIN** e **END** pode-se usar “{“ e “}”.
- **while-do** (enquanto-faça): também semelhante ao PASCAL. Sua sintaxe é **WHILE** <condição> **DO** <processo>.

ex: **WHILE** FreeCap (Loc1) > 5 **DO**

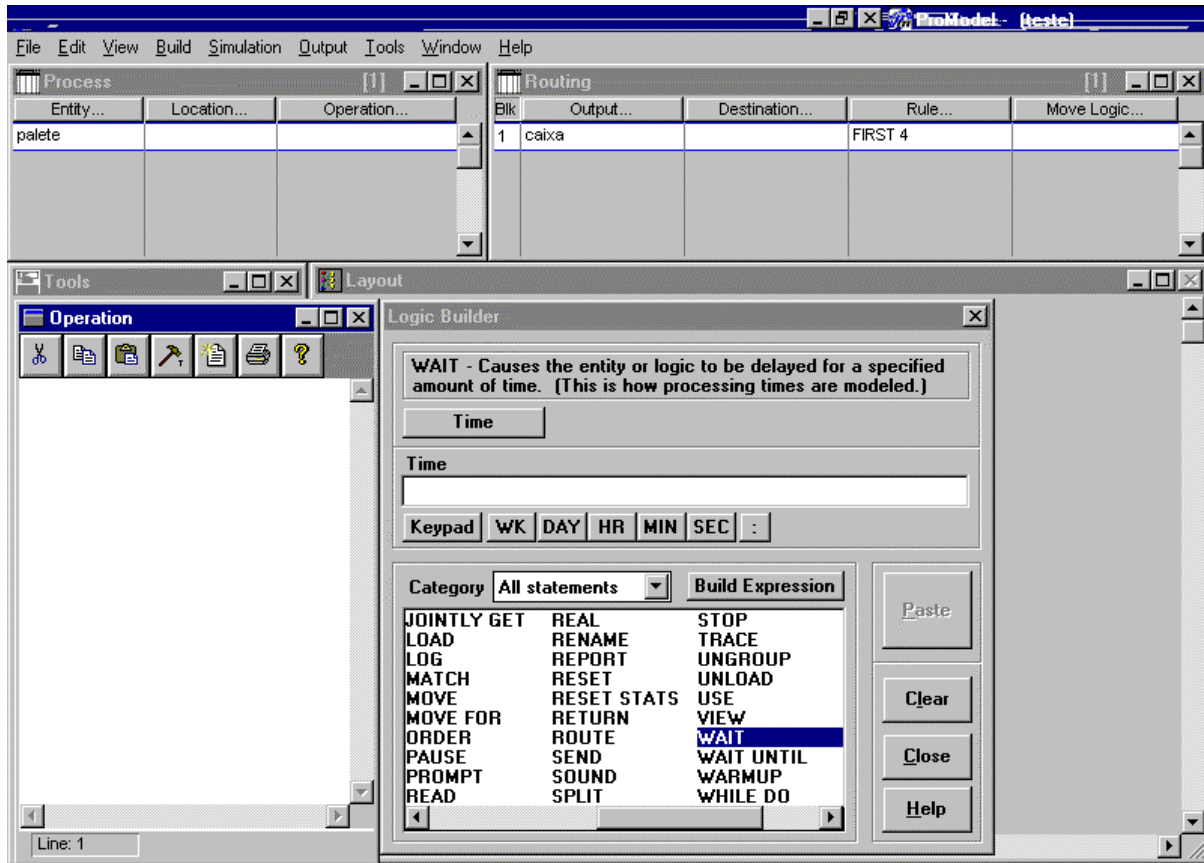
```

BEGIN
inc var2, 5
wait 5 sec
END

```

Obs: a função **FreeCap** verifica a capacidade disponível de um **Location**.

Mais comandos podem ser obtidos na janela **Logic Builder**, aberta acionando-se o botão **Operation** e em seguida selecionando-se o ícone representado por um martelo, ou clicando-se com o botão direito do mouse sobre o campo **Operation**.

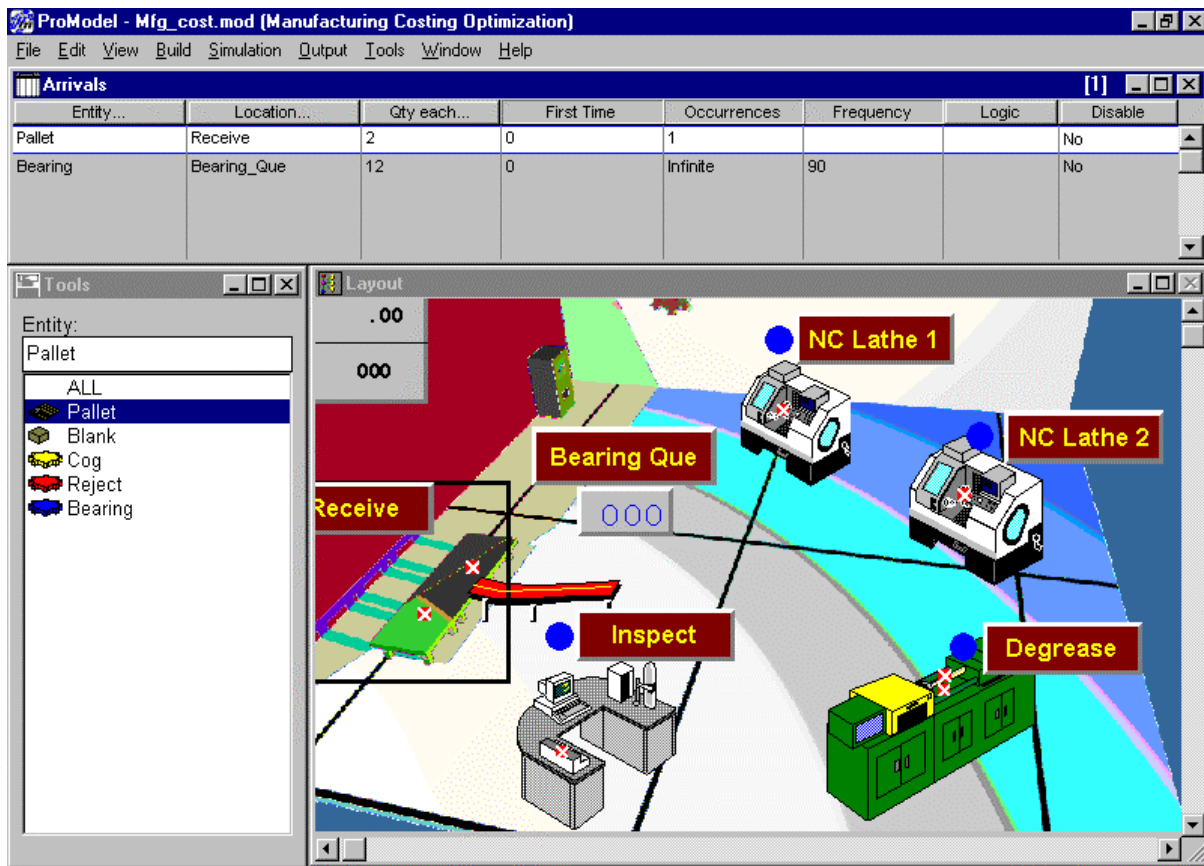


janela “Logic Builder” com lista de comandos

6.7. Definir chegadas ou agendamento da produção (arrivals):

Para se inserir uma nova chegada, escolher uma entidade já definida na janela **Tools** e, na janela **Arrivals**, definir:

- local onde esta entidade aparecerá (**Location**);
- quantas entidades aparecerão a cada vez (**Qty each**);
- quando será a primeira vez que esta entidade aparecerá (**First time**);
- quantidade de eventos de chegada (de 1 a infinito) (**Occurrences**);
- tempo transcorrido entre cada chegada (**Frequency**);
- lógica que seja necessária (**Logic**);
- habilitar ou desabilitar a entrada de entidades no sistema (**Disabled**).



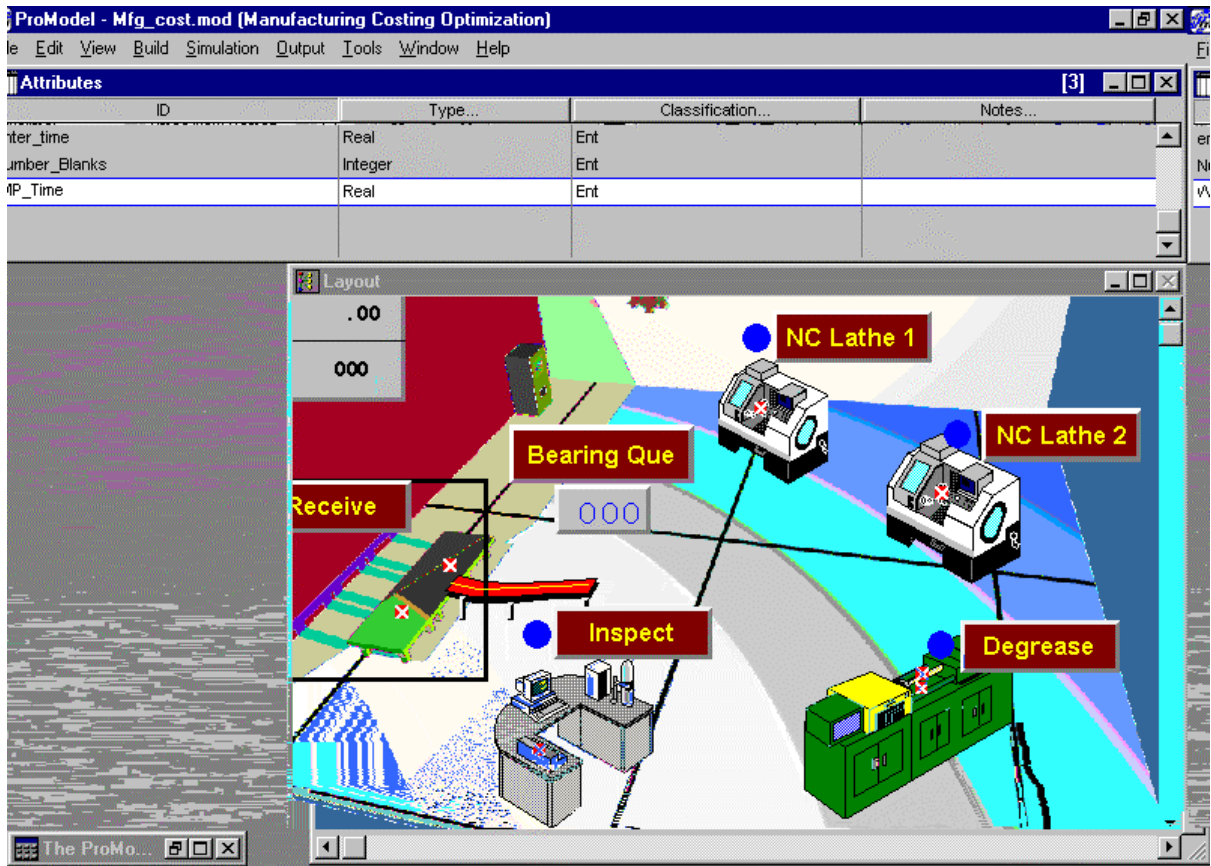
janela para definição das chegadas

6.8. Definir horário de trabalho (shifts):

Para definir os horários de trabalho é só selecionar no menu o elemento **Shifts** e selecionar **Define** - caso se deseje definir um mesmo horário para vários elementos do sistema - ou **Assign** - este tanto para associar elementos a horários pré-definidos ou para criar horários específicos para o elemento em questão.

6.9. Definir atributos (attributes):

Para se criar um novo atributo é só selecionar uma célula **ID** da janela **Attributes** e fornecer um nome. Em seguida, define-se o tipo (**Type** – real ou inteiro) e com que tipo de elemento este atributo está relacionado (**Classification** - Entidade ou Local).

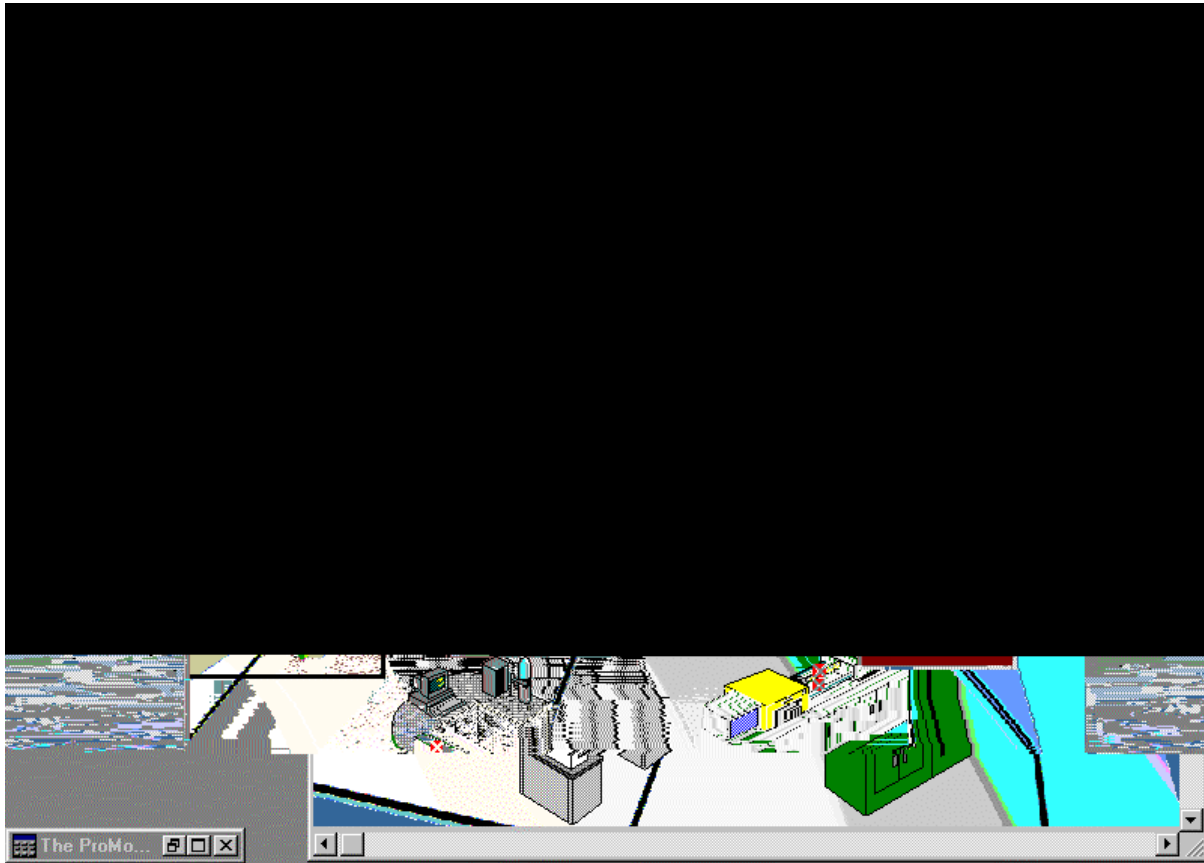


janela de definição de atributos

6.10. Definir variáveis (variables):

Para se inserir uma nova variável, basta definir um novo nome na janela **Variables** e definir:

- se existe representação gráfica ou não (**Icon**);
- tipo (**Type**) – inteiro ou real;
- valor inicial (**Initial value**);
- tipo de dados estatísticos que serão coletados (**Stats**).



exemplo de variáveis

6.11. Definir vetores / matrizes (arrays):

Assim como os elementos anteriores, é só definir o nome, as dimensões (números inteiros separados por vírgulas) e o tipo (real ou inteiro).

6.12. Definir macros (macros):

No ProModel, basta abrir a janela **Macros**, dar um nome no campo **ID** e um valor, texto ou conjunto de comandos no campo **Text**.

6.13. Definir subrotinas (subroutines):

Para se definir uma subrotina, basta abrir a janela **Subroutines** e:

- dar um nome a ela (**ID**),
- seleccionar o tipo de dados que esta trabalha (**Type**) – real, inteiro, interativo ou nenhum destes;

- definir parâmetros para a subrotina (**Parameters...**, se necessário);
- definir a lógica (**Logic**).

```

top:
WAIT 24 hr

ORDER N(8, 1) consumption_order TO order_E2
ORDER N(8, 1) consumption_order TO order_E3
ORDER N(8, 1) consumption_order TO order_E6
ORDER N(8, 1) consumption_order TO order_E7
ORDER N(8, 1) consumption_order TO order_E8

SEND done_at_E2 product_1 TO minneapolis_customer
SEND done_at_E3 product_1 TO boston_customer
SEND done_at_E6 product_1 TO phoenix_customer
SEND done_at_E7 product_1 TO sanfrancisco_customer
SEND done_at_E8 product_1 TO tampa_customer

IF raleigh_warehouse_level-CONTENTS(raleigh_warehouse)>0 THEN SEND raleigh_warehouse_level-CONTENTS(raleigh_warehouse) product_1 TO raleigh_ware
IF stlouis_warehouse_level-CONTENTS(stlouis_warehouse)>0 THEN SEND stlouis_warehouse_level-CONTENTS(stlouis_warehouse) product_1 TO stlouis_wareh
IF detroit_warehouse_level-CONTENTS(detroit_warehouse)>0 THEN SEND detroit_warehouse_level-CONTENTS(detroit_warehouse) product_1 TO detroit_wareh
IF slc_warehouse_level-CONTENTS(slc_warehouse)>0 THEN SEND slc_warehouse_level-CONTENTS(slc_warehouse) product_1 TO slc_warehouse
IF dallas_warehouse_level-CONTENTS(dallas_warehouse)>0 THEN SEND dallas_warehouse_level-CONTENTS(dallas_warehouse) product_1 TO dallas_warehou
IF neworleans_warehouse_level-CONTENTS(neworleans_warehouse)>0 THEN SEND neworleans_warehouse_level-CONTENTS(neworleans_warehouse) produ
IF chicago_warehouse_level-CONTENTS(chicago_warehouse)>0 THEN SEND chicago_warehouse_level-CONTENTS(chicago_warehouse) product_1 TO chicago
IF boise_warehouse_level-CONTENTS(boise_warehouse)>0 THEN SEND boise_warehouse_level-CONTENTS(boise_warehouse) product_1 TO boise_warehouse
IF albuquerque_warehouse_level-CONTENTS(albuquerque_warehouse)>0 THEN SEND albuquerque_warehouse_level-CONTENTS(albuquerque_warehouse) proc

ORDER oklahomacity_production_level product_1 TO oklahomacity_production
ORDER seattle_production_level product_1 TO seattle_production

done_at_E2 = 0
done_at_E3 = 0
done_at_E6 = 0

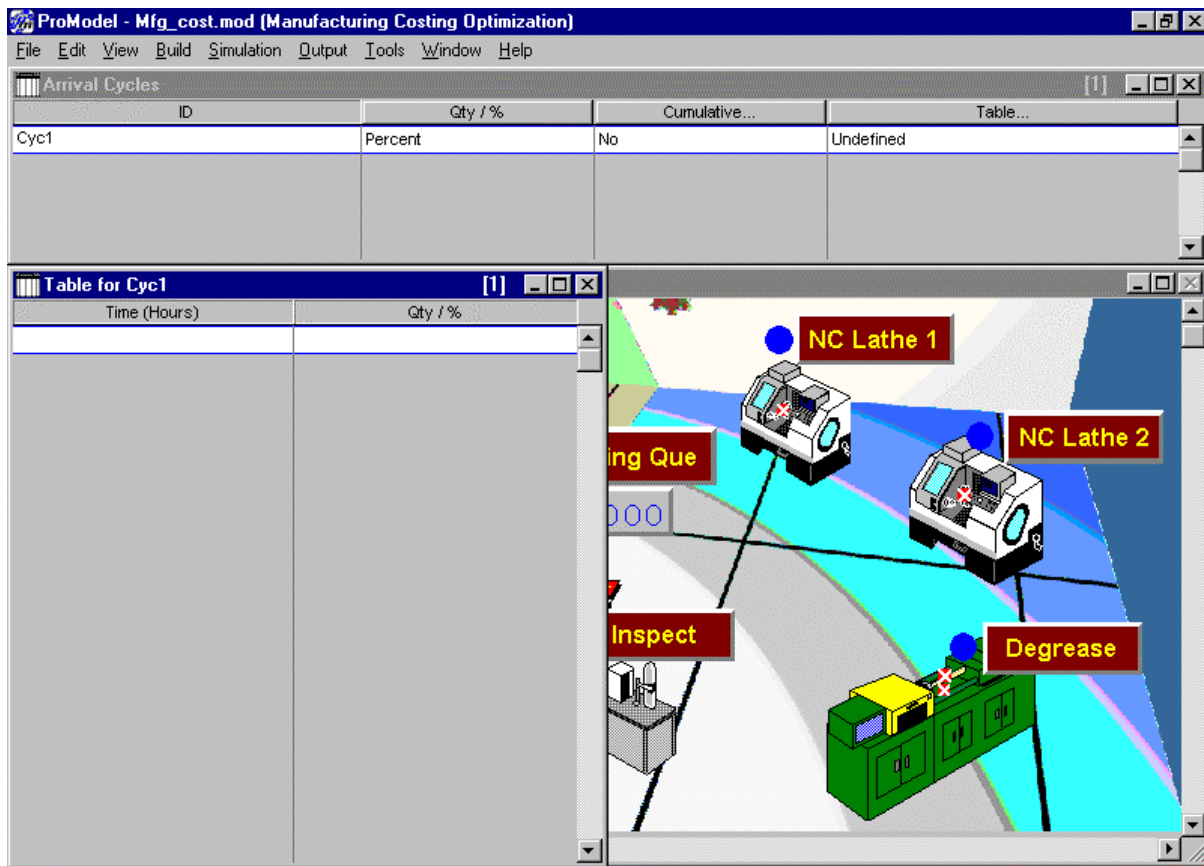
```

exemplo de subrotina

6.14. Definir ciclos de chegada (arrival cycles):

Para se definir os ciclos de chegada basta abrir a janela **Arrival Cycles** e definir:

- nome (**ID**);
- se será em quantidade ou porcentagem (**Qty/%**);
- se se trata de uma distribuição cumulativa ou não (**Cumulative**);
- intervalo e a quantidade (ou porcentagem) (**Table**).

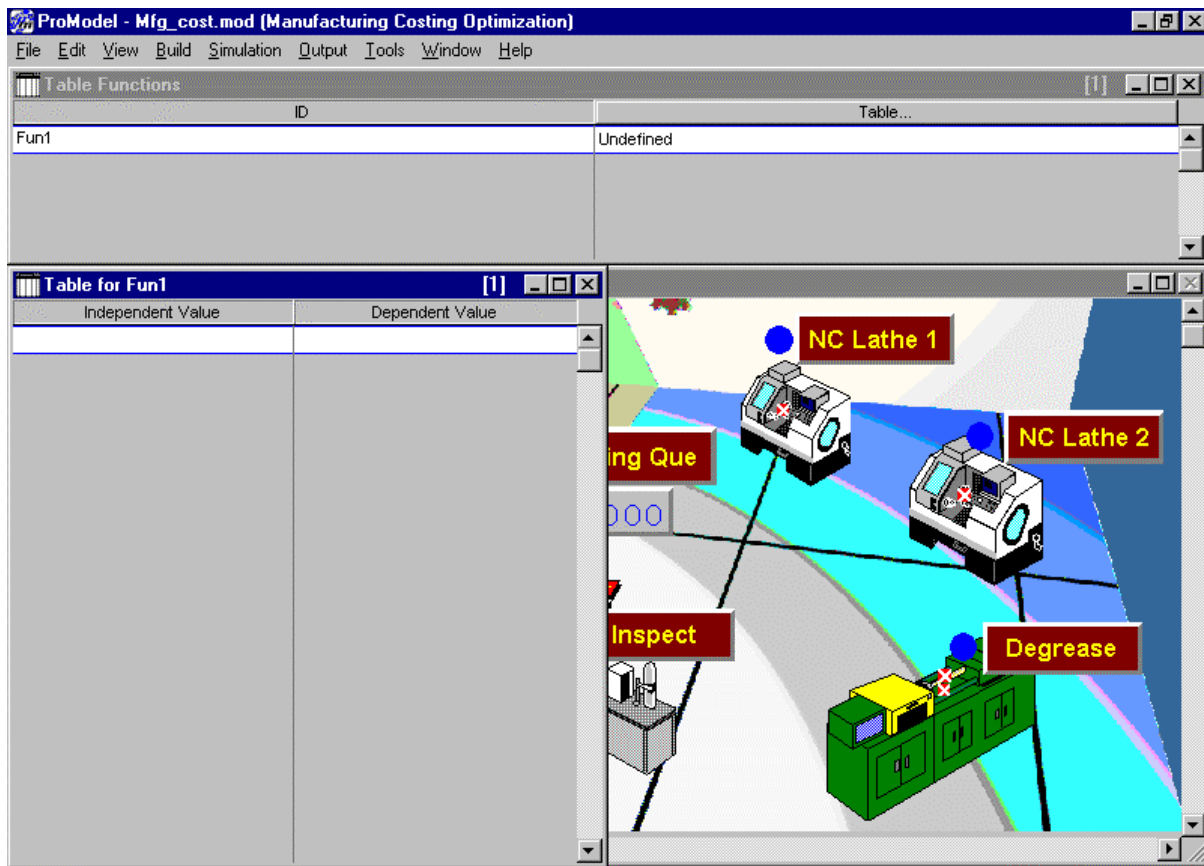


exemplo de ciclo de chegada

6.15. Definir funções tabeladas (*table functions*):

Para se definir uma função tabelada, basta abrir a janela **Table Functions** e:

- dar um nome para a função (**ID**);
- escrever uma tabela de variáveis dependentes e independentes (idêntico às tabelas que se escreve para regressão linear) (**Table...**) – a idéia por trás disto é que o software se utiliza destes parâmetros para gerar uma “curva de distribuição” com os dados.



exemplo de função tabelada

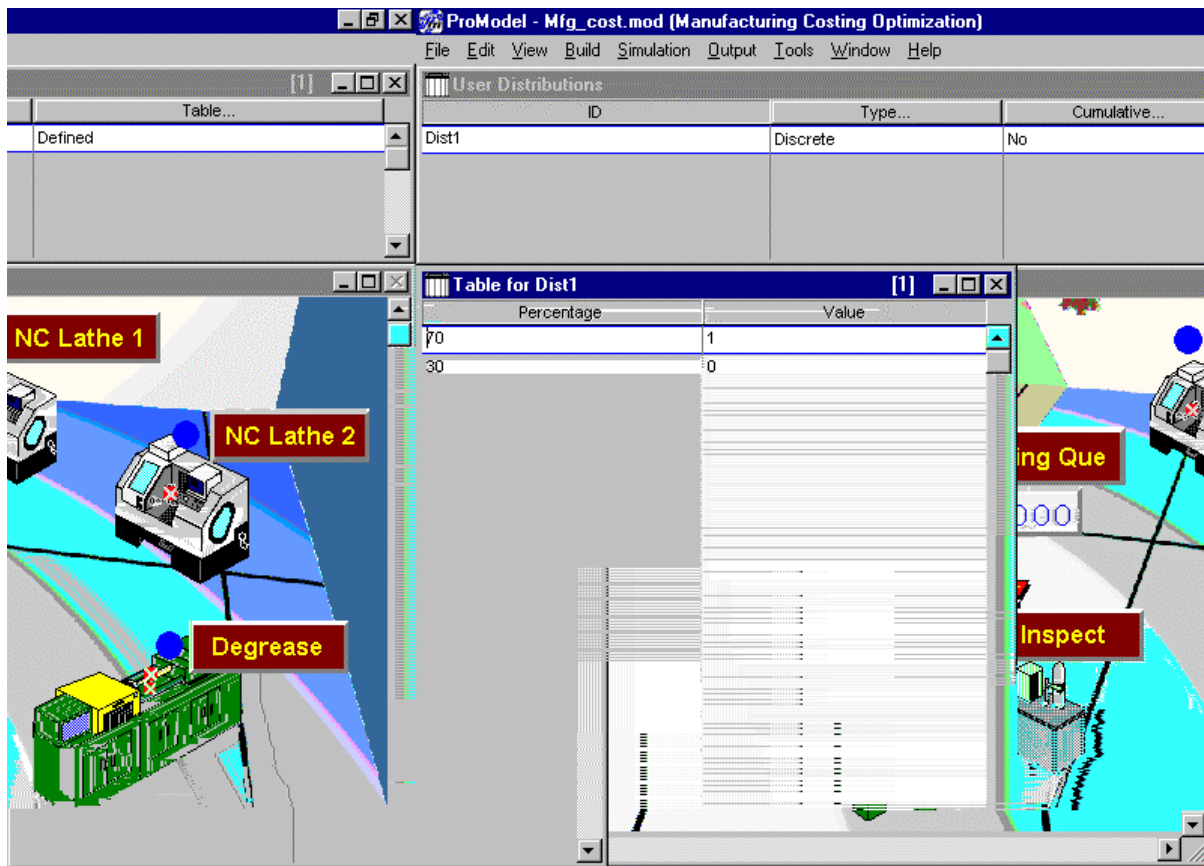
6.16. Definir distribuições do usuário (*user distributions*):

Para se definir uma distribuição do usuário basta abrir a janela **User Distributions** e:

- dar um nome para a distribuição (**ID**);
- definir se é uma distribuição contínua ou discreta (**Type**);
- se é ou não uma distribuição cumulativa (**Cumulative...**);
- definir o modelo de distribuição (**Table...**) como na figura abaixo.

As distribuições podem ser escritas através do **Logic Builder** (Processos → **Operation** → botão direito do mouse → botão **Build Expression** → **Distribution Functions**).

Ajustes de curvas podem ser feitos através do software **Stat::Fit**, que acompanha o ProModel.



exemplo de distribuição do usuário

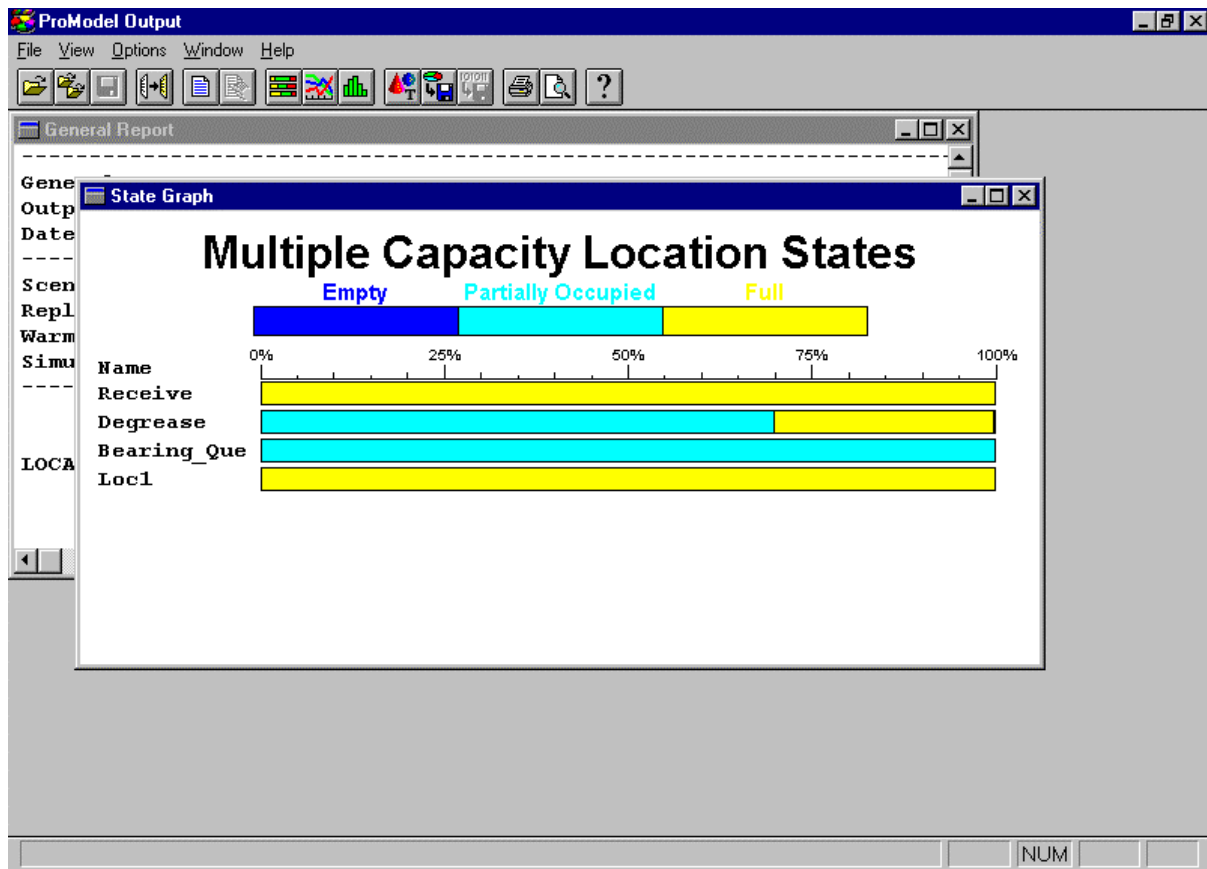
7. Visualização das estatísticas do modelo

Após uma simulação realizada com sucesso, o usuário pode fazer a coleta estatística de dados para posterior análise.

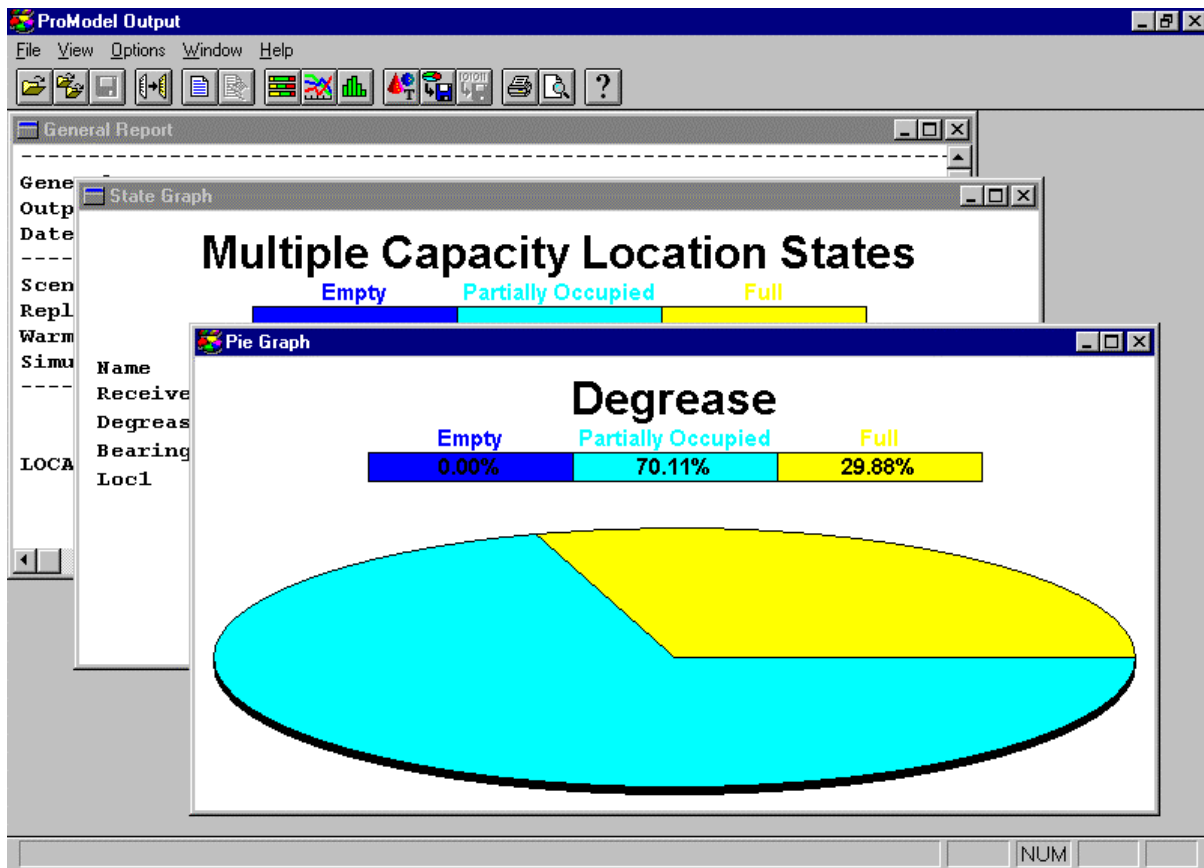
As estatísticas básicas coletadas para os locais produzem duas séries de dados tabulados: uma para locais de capacidade unitária e outra para locais de capacidade múltipla. Para se coletar estatísticas detalhadas de qualquer local, deve-se selecionar o registro do local em questão na tabela de edição encontrada dentro do módulo de locais e então selecionar **Stats**, e, em seguida, **Detailed**.

A visualização gráfica dos elementos pode ser feita selecionando-se o ícone onde se encontram desenhadas três barras horizontais (em vermelho, amarelo e verde) e selecionar o tipo de objeto que se deseja visualizar. Caso se queira uma visualização em forma de pizza,

basta clicar sobre a barra desejada que um novo gráfico aparecerá. Os botões ao lado fornecem, respectivamente, gráficos estatísticos e histograma dos dados de saída.



janela de saída de dados



detalhamento de um dos itens do grupo de locais

Demais estatísticas:

- para as entidades encontramos informações do tipo: nome da entidade, quantas saíram do sistema e a quantidade final que permanece no sistema. Analogamente aos locais, pode-se coletar estatísticas detalhadas;
- os resultados reportados para os recursos são divididos em duas categorias: resumo (relatório sumário sobre todas as unidades do recurso) e por unidade (estatísticas coletadas individualmente para cada uma das unidades do recurso)
- para as variáveis encontra-se o nome da variável, quantas vezes esta teve o seu valor alterado durante a execução da simulação, qual o tempo médio entre trocas, qual o seu valor final e qual o seu valor médio. Quando se opta por estatísticas detalhadas, esta acrescenta o desvio padrão sobre o valor médio do relatório;
- ressalta-se que as estatísticas podem ser obtidas para todos os objetos disponíveis pelo ProModel.

Bibliografia

BAIRD, SCOTT P., LEAVY, JEFFREY J. Simulation Modeling Using ProModel for Windows. Proceedings of the 1994 Winter Simulation Conference.

KELTON, W.D. A tutorial on design and analysis of simulation experiments. Proceedings of the 1995 Winter Simulation Conference.

SEILA, A.F. Introduction to simulation. Proceedings of the 1995 Winter Simulation Conference.