

# An Overview of Help

Help documents the features, operation, and internal workings of the software. It contains three main sections:

- **Part 1 - Basic Operations** documents the software's basic [controls](#), [menus](#), [toolbars](#), and palette items ([building blocks](#), [tools](#), and [objects](#)).
- **Part 2 - Deeper Details** covers details associated with the Map and Model layers. This includes discussions of the [Builtin functions](#), which can help you to accomplish a variety of specialized tasks as you construct the equation logic of your models. Part 2 also includes information on [importing and exporting model data](#); creating [Causal Loop Diagrams](#), working with [Sub-models](#); using the [cycle-time calculation](#) capabilities; working with [Arrays](#); using [modules](#) to create larger models out of smaller, self-contained models and running models on Windows machines using [command line options](#).
- **Part 3 - Technical Appendices** include information on [DT](#), the solution interval the software uses for making its calculations; the [numerical integration algorithms](#) employed by the software; the internal [rules of grammar](#) used by the software to control things like connections between building blocks and priorities of flow calculations; procedures for creating [spatial maps](#) from spatial data in your models; and publishing your models to the Internet with [isee NetSim](#). Part 3 concludes with set of technical tips and troubleshooting techniques.

# Introduction

This documentation applies to the ***iThink*®** and **STELLA®** software. As you can see by scanning the Table of Contents, our aim in preparing this documentation is to provide you with essential "how to" information concerning the use of the software. Use this documentation on an as-needed basis. Whenever you require more information about the features, operation, or inner workings of the software, look here.

This introductory chapter is intended to provide you with a big picture overview of how the software works. Once you have the big picture firmly in mind, you will be in a good position to learn more about the specific features and operational details of the software. This chapter contains the following sections:

- **An Overview of the Operating Environment** describes the software's four-layer operating environment.
- **Interacting with *iThink* and STELLA** describes how to use clicking, double-clicking, dragging, and right-clicking to build and interact with a model.
- **How to Use This Guide** provides some simple suggestions for using the documentation in the most effective manner.

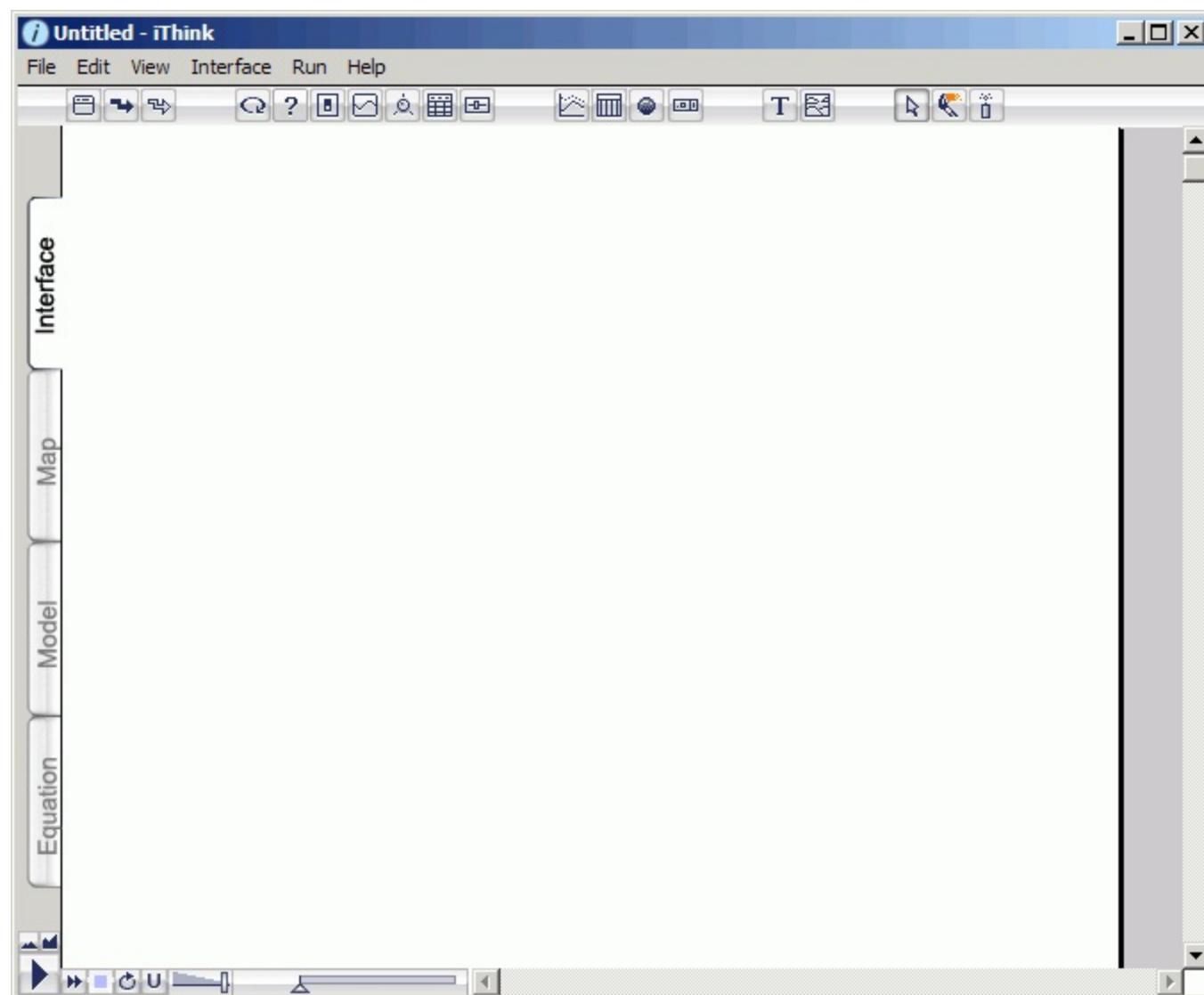
 [Related Topics](#)

# An Overview of the Operating Environment

The main **iThink®** and **STELLA®** window is divided into four tabbed pages: Interface, Map, Model, and Equation. Each tab represents a distinct layer in the model and each provides a different way of designing and presenting a model.

Figure 1-1 shows the Interface layer tab selected in the model window. The tabs along the left side of the window allow you to move to each layer. The [menu bar](#) and [toolbar](#) along the top of the window provide appropriate commands and tools for working with each layer.

*Figure 1-1 An Overview of the Operating Environment*



When you start **iThink** or **STELLA**, the software opens on the Map layer. This layer is where you will lay out your thinking in the form of a map.

The Model layer is below the Map layer and will transform maps into models that can be simulated on your computer. The Model layer is often referred to as the "engine room" for the models you create.

Above the Map layer, you'll find the Interface layer. As the name suggests, the Interface layer provides you with the tools needed for engaging end-user interfaces to your models. You'll use these Interface layer tools to create, for example, flight simulator cockpits in which users can interact with the model as the simulation progresses. The Interface layer makes it possible for you to transform a model into a compelling environment for learning.

Finally, below the Model layer you'll find the Equation layer. This layer gives you a list of all the equations that make up your model. Although you will seldom (if ever) need to visit the Equation layer, it's always good to know that the equations are accessible to you.

For you as model builder, the multi-layer design makes it significantly easier for you to manage visual and conceptual complexity. You've got a spot to lay out your thinking. You've got a place for the model structure. You've got a home for an equations list. And finally, you have a locus for input/output interactions with the model.

For your clients and colleagues, the benefits of the multi-layer approach are even more palpable. Separate layers make it possible to unravel the intricacies of your model in "mind-size bites." Potential for learning is maximized. Potential for confusion is minimized. The result: models with impact!

## [Related Topics](#)

# Interacting with *iThink* and STELLA

***iThink*** and **STELLA** provide the same interface tools and options you're used to using with other Windows and Macintosh applications, including clicking, double-clicking, dragging, and right-clicking:

- **Clicking** – Use clicking to select an item in any layer of a model. To select most items in a model, click the item once. When the item is selected, it is highlighted. To select a button, click its border. When a button is selected, selection handles appear around the button.

You also use clicking to indicate where to add a new element on the Interface, Map, or Model layers, to select commands from the [menus](#), to select a [toolbar button](#), and to move from one layer to another by clicking a navigation tab on the left side of the window.

- **Double-clicking** – Use double-clicking as a shortcut to view or edit the properties of most items in a model. For example, to view the details of a table, double-click the table pad icon in the model. To define the table's properties, double-click the table pad's page to view the Define Table dialog box.

Single- or double-clicking a button "pushes" the button so that it performs its associated action.

- **Dragging** – Use dragging to select one or more items by holding down the mouse button and then dragging to draw a rectangle around the items you want to select. When you release the mouse button, the items within the rectangle are selected.

You can also use dragging to move selected items to a new location on the layer and to change the size of an item by dragging a corner of the item to stretch it to a new size.

- **Right-clicking** – Right-click any element to see a shortcut menu of commands that apply to the element you clicked. When the shortcut menu appears, choose the command you want to use. For example, to view or change the properties of a button element, right-click the button and then choose **Open**. The Button dialog box appears.

 [Related Topics](#)

# How to Use This Guide

We strongly encourage you to work through the tutorial materials that come with the software, before jumping this documentation.

If, after working through the tutorials, you require information from this documentation, we'd suggest that you begin by scanning the Table of Contents. The contents will point you to the relevant section. Then, look on the first page of the section, where you will find an overview of what's within. Finally, go to the appropriate sub-section, and find what you need.

This Guide provides documentation for both the Macintosh and the Windows versions of the software. Essential operation of the software is the same on both platforms. Where clear differences in operation do exist, notes in the figures or in the text, will explain platform-specific differences.

 [Related Topics](#)



# Part 1 - Basic Operations

This portion of the Technical Documentation provides a detailed discussion of the essential features of the software. The sections in this Part 1 are organized around the Controls, Menus, Toolbars, Building Blocks, Tools, and Objects found in the software.

- **Controls** describes the basic window controls available in the ***iThink*** and **STELLA** windows and the Run Controller.
- **Menus** describes the commands available under each menu.
- **Toolbars** describes the toolbars available in the ***iThink*** and **STELLA** windows.
- **Building Blocks** describes the Building Blocks available on the Interface, Map and Model layers.
- **Tools** discusses the operation of the Tools in the software.
- **Objects** provides information on the operation of Objects: Loop Pad, Button, Input Devices, Graph Pad, Table Pad, Status Indicator, Numeric Display, Text Box, Sector Frame, and Graphics Frame.

This part of the Technical Documentation is designed to be used as reference material. When you need to know something about the basic operation of the software, the answer is likely to be somewhere within Part 1. For your convenience, the first topic of each section provides a summary of the topics covered within the section.

# Introduction to Controls

The topics in this section introduce you to the basic controls available in the software window. These controls allow you to navigate from layer to layer, change the zoom level in all views, and run model simulations.

The following topics describe:

- **[Window Controls](#)**, which gives you a visual overview of the location and use of the window tools
- **[The Run Controller](#)**, which describes the functions of each button in the Run Controller

---

**Note:** To learn about the controls associated with specific building blocks and objects, see [Introduction to Building Blocks](#) and [Introduction to Objects](#).

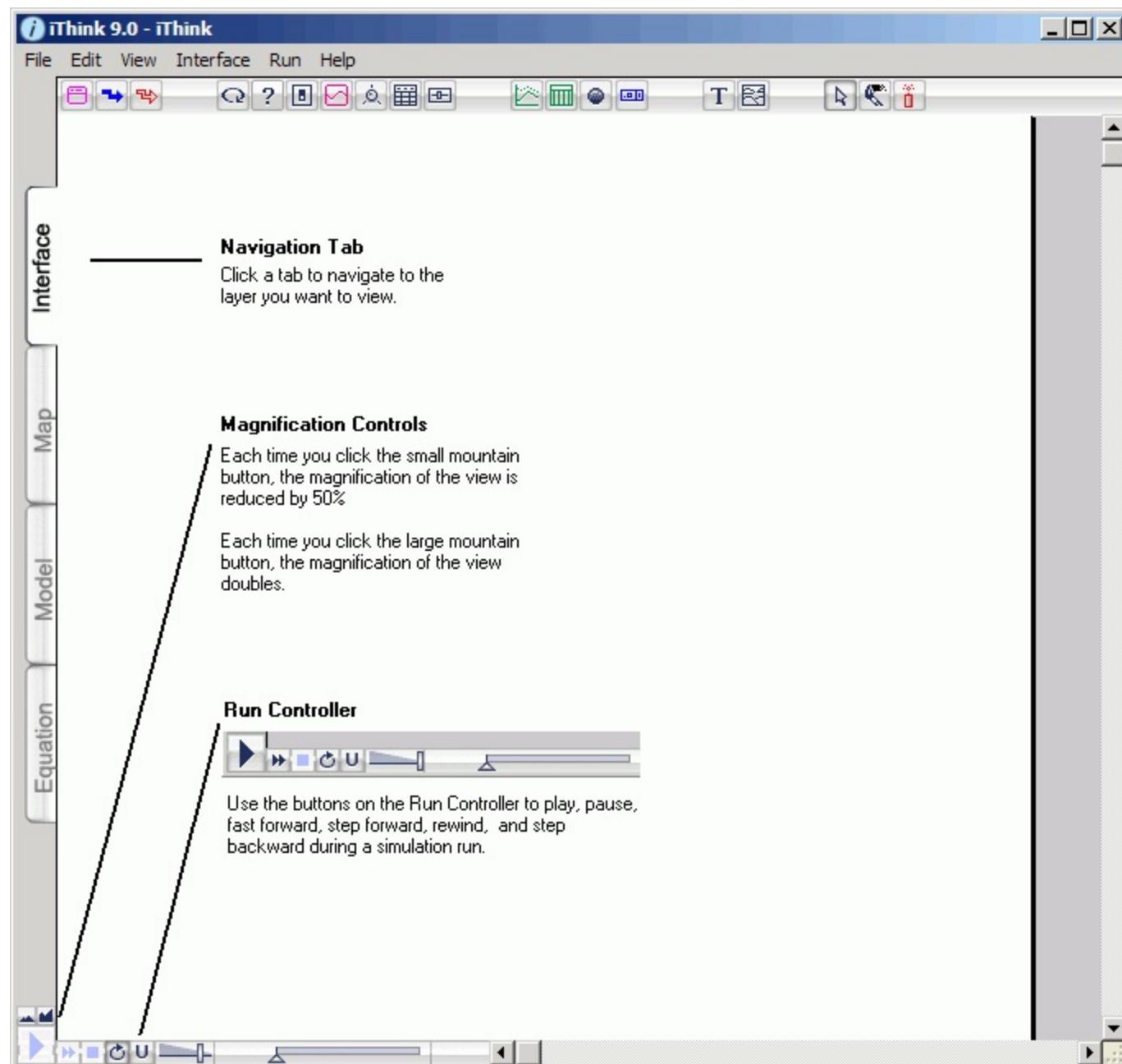
 [Related Topics](#)

# Window Controls

Figure 2-1 shows you the location of the standard controls available in the **iThink** and **STELLA** windows:

- **Navigation tabs**, which allow you to quickly switch from one layer to another
- **Magnification controls**, which allow you to change the magnification of the model view
- **Run Controller**, which allows you to control model simulations

*Figure 2-1 Controls on the Interface Layer*



 [Related Topics](#)

# The Run Controller

The Run Controller appears in the lower-left corner of the **iThink** and **STELLA** window. The Run Controller gives you quick access to many of the commands on the [Run menu](#). In addition, it gives you a visual indication of the progress of a simulation.

Figure 3-27 shows the Run Controller.

*Figure 3-27 Run Controller*



The parts of the Run Controller are described below.



## Run button

Click this button to run the simulation.

## **Pause button**

When a simulation is running, the Run button changes to the Pause button. Click the Pause button to pause the model simulation. When you click the Pause button, it changes back to the Play button.

## **Fast Forward button**

Click this button to push the model simulation forward as fast as possible. The Fast Forward button is available only when the simulation is playing.

## **Step Forward button**

When the simulation is paused, the Fast Forward button changes to the Step Forward button. Click the Step Forward button to step the model simulation forward by one DT. The Step Forward button is available only when the simulation is paused.

## **On/Off button**

Click this button to turn the simulation off. If you start the simulation again, it will start at the beginning of the simulation.

## **Loop button**

Click this button to turn on looping mode. When looping is turned on, the simulation starts running again as soon as it reaches the end of the simulation. When looping is turned off, the simulation stops when it reaches the end of the simulation. To turn off looping mode, click the Loop button again.

## **Restore All Devices button**

Click this button to restore all devices in the simulation, including Graphs, Tables, Sliders, Graphical Input Devices, Knobs, List Input Devices, Switches, Warning Devices and Numeric Displays.



## Speed Control

Use the Speed Control slider to control the speed of the simulation. Drag the bar on the slider to the left to slow down the simulation. Drag the bar to the right to speed up the simulation.

## Time Indicator

The Time Indicator shows you the progress of the simulation by showing you a clock and a slider.

### Sensi Run Number

If you have specified more than one run in the [Sensitivity Specs dialog box](#), the space to the right of the Time Indicator indicates the current run number.

#### [Related Topics](#)

# Introduction to Menus

The topics in this section describe the commands available under each menu:

- [iThink/STELLA Menu](#)
- [File Menu](#)
- [The Edit Menu](#)
- [The Interface, Model, and Equation Menus](#)
- [The Run Menu](#)
- [The Help Menu](#)

 [Related Topics](#)

# iThink/STELLA Menu

The iThink or STELLA menu appears in the Macintosh version only.

The commands on this menu give you information about the ***iThink*** or **STELLA** version you have installed, allow you to specify the software's default settings, allow you to exit from the software, and provide other standard Macintosh menu commands for accessing services and hiding/showing windows.

Each command on the iThink or STELLA menu is briefly described below.

## **About iThink / About STELLA**

Displays the version number of the software and your registration number.

---

*Note:* To view this information in the Windows version, choose the **AboutiThink** or **About STELLA** command from the [Help menu](#).

---

## Preferences

Preferences allows you to customize the default user interface and simulation settings in the Default Settings dialog. Changes made to Default Settings take effect immediately for all new models. For more information, see [Default Settings](#).

---

*Note:* You can also specify default settings by using the Default Settings command on the [File menu](#).

## **Services**

Gives you access to other service and utility applications.

## **Hide iThink / Hide STELLA**

Hides the ***iThink*** or **STELLA** window.

## **Hide Others**

Hides all other open windows on your desktop.

## **Show All**

Shows all previously hidden windows on your desktop.

## **Quit iThink/Quit STELLA**

Quits the application. If you have made any changes to your model since the last time it was saved, you will be asked if you want to save these changes before quitting.

---

*Note:* To exit from the application in the Windows version, use the **Exit** command on the [File Menu](#).

---

 [Related Topics](#)

# File Menu

The commands on the File menu allow you to create, open, close, print and save models. In addition, this menu provides commands for exporting models, locking models, specifying default settings, and exiting from the application.

Each command on the File menu is briefly described below.

## New

New creates a new model file. Because the software allows you to work with only one open file at a time, New is gray whenever a model file is open. Close any open models before you create a new one.

## **Open**

Open brings forth a standard Get File dialog. Navigate through the directory structure, select the model file you wish to open, and click the Open button (Macintosh) or the OK button (Windows).

## **Close**

Close closes an open model file as well as any unpinned Loop Pad, Graph Pad or Table Pad windows. If the open model file differs from what's saved on disk, a dialog will appear asking if you want to save your work.

## **Save**

Save saves an open model file in its current state to disk. For the Windows version, .itm is the extension added to all ***iThink*** model file names; .stm is the extension added to all **STELLA** model file names.

## **Save As**

Save As brings forth a dialog for saving the open model file under a new name in a selected directory or folder. For the Windows version, .itm is the extension added to all *iThink* model file names; .stm is the extension added to all STELLA model file names.

---

**Note:** By default, ***iThink*** and **STELLA** models created or saved in Version 9 are also compatible with Version 8. You can also choose to save a model as a Version 7 model so that you can open it in Version 7 or later of the software. To save a model as Version 7, select the appropriate option in the Save as type box in the Save model file as dialog box.

---

## Save As Runtime File

Save As Runtime File saves a copy of your model file to disk, in a format that can be read by the isee Runtime software. The isee Runtime software presents models with a professional look in full-screen mode; title bars, menus, scroll bars, window controls and tool bars will be hidden. As a result, you, as an author, can control user access to the various components of the model and interface.

When you choose this command, the Save model file as dialog box opens so that you can specify the name and location of the runtime file want to create. The file extension for a **STELLA** Runtime file is .str. The file extension for an **iThink** Runtime file is .itr.

---

**Note:** Before you choose this command, use the options in the Page Size and Runtime Options dialog box (available from the Interface Preferences dialog box) to select the page size and other runtime options for the saved model. For more information, see [Interface Preferences](#).

The isee Runtime software is an excellent vehicle for commercial distribution of your work. For details on commercial licensing, contact [isee systems](#) directly. To facilitate your work in the development of flight simulators and learning laboratories, we have included a developer's copy (i.e., not for distribution) of the isee Runtime software.

## Export for NetSim

Export for NetSim brings forth a dialog for exporting your model file to disk in a format readable by isee NetSim™, a separately available product for creating and interacting with **iThink** or **STELLA** models over the web (or any network). Examples of web-based models created with isee NetSim are available on the isee systems web site.

---

**Note:** Before you choose this command, use the options in the Page Size and Runtime Options dialog box (available from the Interface Preferences dialog box) to select the page size and home page for the exported model. For more information, see [Interface Preferences](#).

To conform to NetSim's required format, you must be able to run your model, and it cannot contain Queues, Ovens, cycle-time structures, or certain Built-in functions that are not supported by isee NetSim. The export process will let you know if your model fails to conform to the required format for isee NetSim. If your model fails to conform, the export process will be stopped. Other interface objects that are not supported will not cause the export or publication to fail, but you will receive a message that says that they have been excluded from the model. For a current list of unsupported interface

objects, go to <http://www.iseesystems.com/unsupportednetsim>.

A successful export of the model will result in an encrypted file with the .TXM extension.

For more information about isee NetSim, see [Introduction to isee NetSim™](#) in "Part 3 - Technical Appendices", and visit <http://www.iseesystems.com>

## **Export Text Boxes**

Choosing this menu item will cause the contents of all text boxes that have been marked for export to be copied to the clipboard. Additionally, the Export Text Boxes item gives you the opportunity to save the contents of "marked for export" text boxes as a text file. Contents of text boxes will be exported in alphabetical order based on the titles you provide.

Exporting the contents of text boxes is very useful when you desire to collect, for example, a set of responses to model output from users of your model. After users have entered their responses in text boxes, you can choose Export Text Boxes to collect responses in one convenient venue.

Marking a text box for export is a straightforward operation within the dialog of the text box. For details on the Export Text check box, see [Text Box](#).

## **Save As Image**

This command appears when the Interface, Map, or Model layer, or an unpinned Graph or Loop Pad is active. Save as Image allows you to create an image file of the contents of the active window (for an unpinned graph pad or loop pad, the currently-showing page of the Pad). You can choose to save the file in any of the following formats: PICT (\*.PCT), GIF (\*.GIF), Bitmap (\*.BMP), TIFF (\*.TIF), Jpeg (\*.JPG), or PNG (\*.PNG).

## **Save As Text**

This command appears when: (1) an unpinned Table Pad is the active window; or (2) the Equation layer is active. Save as TEXT allows you to create a text file of the currently-showing page of the Table Pad, or of the equations. Equations will be saved in accordance with the settings you have made in the Equation Preferences dialog (for more information, see [Equation Preferences](#)). The data will be saved in TEXT format. This file then can be opened by a variety of word processing, spreadsheet, and statistical packages. On Windows, the extension for a TEXT file is .txt.

## **Revert**

Revert causes an open model to revert to its state when last saved. Any model changes since the last save will be lost.

## **Print Setup** (Windows) or **Page Setup** (Macintosh)

The Print Setup or Page Setup menu item brings forth the Print or Page Setup dialog specific to your hardware/printer configuration.

## Print

This context-sensitive menu item depends on what layer is active, as well as what is selected within the layer. Choosing this item brings forth a print dialog specific to your printer, as described below.

- **Print Interface** – This item appears on the Interface layer. Print Interface allows you to print the contents of the Interface layer - including the top-most pages of pinned Graphs, Tables, and Loop Pads. Unpinned Graph, Table, and Loop Pads will not be printed.
- **Print Model** – This item appears on the Map and Model layers. Print Model allows you to print the contents of your model diagram - including the top-most pages of pinned graphs and tables. The structure within Sub-model spaces which have been closed into their icons, Unpinned Graph, Loop, and Table Pads will not be printed with the diagram.
- **Print Decision Diamond** – This item appears on the Map and Model layer, whenever one or more Decision Process Diamonds or Sub-model Spaces have been opened on the diagram. Print Decision Diamond allows you to print the contents of the opened Decision Process Diamonds or Sub-model spaces. Items showing on the main level of the diagram will not be printed.
- **Print Graph Pad** – This item appears on the Interface, Map, and Model layers when an unpinned Graph Pad is the active window. Print Graph Pad allows you to print all the pages of the Graph Pad in question.
- **Print Table Pad** – This item appears on the Interface, Map, and Model layers when an unpinned Table Pad is the active window. Print Table Pad allows you to print all the pages of the Table Pad in question.
- **Print Loop Pad** – This item appears on the Interface layer when an unpinned Loop Pad is the active window. Print Loop Pad allows you to print all pages of the Loop Pad in question
- **Print Equation** – This item appears when the Equation layer is showing. Print Equation allows you to print the equations, as they appear on the Equation layer. To alter the appearance of the equations, for example, to print documentation along with the list, visit the [Equations Preferences](#) dialog.

## **Lock Model**

Lock Model allows modelers to control access to their models. The default setting is Full Access. For more information, see [Setting Security Options for a Model](#).

## **Default Settings**

Default Settings allows you to customize the default user interface and simulation settings. Changes made to Default Settings take effect immediately for all new models. For more information, see [Default Settings](#).

---

**Note:** If you are using the Macintosh version, you can also access the Default Settings dialog by choosing the **Preferences** command from the [iThink or STELLA menu](#).

## **Exit (Windows)**

Exit quits the application. If you have made any changes to your model since the last time it was saved, you will be asked if you want to save these changes before quitting.

---

**Note:** To exit from the application in the Macintosh version, use the **Quit iThink** or **Quit STELLA** command on the [iThink or STELLA menu](#).

---

 [Related Topics](#)

# The Edit Menu

The commands on the Edit menu allow you to perform standard editing tasks on the currently displayed layer, including cutting, copying, and pasting text and images, and undoing the most recent deletion. In addition, the Edit menu allows you to create and manage import/export links between an Excel worksheet and your model.

Each command on the Edit menu is briefly described below.

## **Undo**

The Undo feature provides one level of undo for deletion. If you inadvertently dynamite or clear model structure, input devices or output devices (excluding graph and table pads), the Undo feature may be used to restore the structure or devices.

## Cut

Cut cuts the selected information in a window to the clipboard. For information about what can be cut from a model, see [Cut, Copy, Paste, and Clear Operations](#).

## **Copy**

Copy copies the selected information in a model to the clipboard. For information about what can be copied from a model, see [Cut, Copy, Paste, and Clear Operations](#).

---

*Tip:* Depressing the alt key (Windows) or option key (Macintosh) while copying a Table column (or row in Horizontal orientation) will cause the variable name to be copied with the data. Similarly, for a specific time slice containing data for all variables in a Table Pad page, the time will be copied if the alt key (Windows) or option key (Macintosh) is depressed when you choose copy. If you do not depress the alt key (Windows) or option key (Macintosh) while copying, the variable name (or time information) will not be copied with the data.

---

## Paste

Paste pastes information from the clipboard into a model. Note that pasting occurs only when consistent with the operation of the software. For example, you cannot paste a picture into an equation dialog. For information about what can be pasted in a model, see [Cut, Copy, Paste, and Clear Operations](#).

---

*Tip:* Pasting into an output column in a graphical function requires that the entire output column be selected. The easiest way to do this is to click on the word "Output" in the column header.

---

## **Delete** (Windows) **Clear** (Macintosh)

Delete/Clear clears the selected model element(s) from the Interface, Map, or Model layers as well as selected variables from Graphs and Tables. In Windows, pressing the "Delete" key is equivalent to using the Delete command. On the Macintosh, pressing the "Clear" key (or command-delete) is equivalent to the using the Clear command.

## Select All

Select All selects all elements on the Interface, Map, or Model layers, in the Equations list, and on the top page of the active Table Pad. Once all model elements are selected on the Map, Model, or Equation layer, you can rapidly review and edit equations by choosing Open Selection or by double-clicking on any selected element. The Cancel button will interrupt this rapid review process.

## Select All in Decision Diamond

When a Decision Process Diamond or Sub-model is open and its space has been selected on the Map or Model layer, this menu item changes to read Select All in Decision Diamond. When you choose Select All in Decision Diamond, the decision process diamond space (or sub-model space) will be de-selected, and all elements within the space will be selected.

## **Import Data**

Import Data allows you to set up an import link between an Excel spreadsheet and your model so that you can import data from the Excel worksheet into a model's variables, constants, and graphical functions. For more information, see [Setting Up Import Links](#).

## Export Data

Export Data allows you to set up an export link between a model and an Excel spreadsheet so that you can export data from your model to the Excel worksheet. You can choose to export all variables values in the model or only the data in a specified table in the model. For more information, see [Setting Up Export Links](#).

## Manage Persistent Links

Manage Persistent Links allows you to edit, turn on or off, or delete the persistent import and export links you have created. This command also allows you to initiate a manual import or export via your persistent links. For more information, see [Managing Persistent Import and Export Links](#) and [Manually Importing or Exporting Data](#).

### [Related Topics](#)

# The View menu

Use the commands on the View menu to change your view of the window by navigating to a different layer, show or hide elements, and change the zoom level in the window.

Each command on the View menu is briefly described below.

## **Interface Layer**

Interface Layer navigates to the Interface layer of the model. When you are viewing the Interface layer, this command has a check mark next to it.

## **Map Layer**

Map Layer navigates to the Map layer of the model. When you are viewing the Map layer, this command has a check mark next to it.

## **Model Layer**

Model Layer navigates to the Model layer of the model. When you are viewing the Model layer, this command has a check mark next to it.

## **Equation Layer**

Equation Layer navigates to the Equation layer of the model. When you are viewing the Equation layer, this command has a check mark next to it.

## Show Pads / Hide Pads

When Graph, Loop, and/or Table Pads are opened, but all unpinned ones are hidden behind the Map, Model or Equations, Show Pads will bring these pads to the front. When all opened Graph, Loop, and/or Table Pads are stacked in front of the Map/Model/Equations, Hide Pads will move these objects to the back. Note that Hide Pads is equivalent to a click on the Map, Model or Equations frame. When the Map or Model is between stacks of Graph, Loop, and/or Table Pads, Show Pads will put the remaining pads to the front.

---

*Tip:* To keep pads attached to their layer, pin them down. For more information about pinning and unpinning these objects, see [Loop Pad Surface Operations](#), [Graph Pad Surface Operations](#), and [Table Pad Surface Operations](#).

---

## Hide

Hide hides the element you choose from the Hide submenu:

- **Converters & Ghosts**- This command causes all converters and ghosts which are currently showing on the Map and Model layers to be hidden from view. It also hides any connectors which are attached to converters and ghosts. In so doing, the item provides you with a mechanism for simplifying the display of the model. This item is accessible under the Model menu, when one or more converters and/or ghosts are currently showing on the model
- **Connectors**- This command causes all connectors which are currently showing on the Map and Model layers to be hidden from view. In so doing, the item provides you with a mechanism for simplifying the display of the diagram. This item is accessible under the Model menu, when connectors are showing and converters and ghosts have been hidden.
- **Bundled Connectors**- This command causes all bundled connectors which are currently showing on the Interface layer to be hidden from view. In so doing, the item provides you with a mechanism for simplifying the display of your High-Level map. The item is accessible under the Interface menu, when bundled connectors are showing on the map.
- **Pad Icons**- This command causes all Graph Pad and Table Pad icons on the Interface, Map, and Model layers to be hidden from view. It is accessible under the Interface menu and the Model menu, whenever one or more Pad icons are showing on the surface.
- **Transparent Buttons**- This command causes all transparent buttons to become "invisible." When this happens, any objects, graphics, or text underlying the button will show, but the button will remain active. For more information about buttons, see [Button Dialog Operations](#).
- **Polarity**- This command hides all polarity labels (+/s or -/o) that are currently displayed for Connectors, Bundled Connectors, Flows, and Bundled Flows.

## Show

Show shows the element you select from the Show submenu:

- **Converters & Ghosts** – Choosing this command causes all converters and ghosts which have previously been hidden on the Map and Model layers to be displayed. It also causes all connectors on the Model diagram to be displayed. This item is accessible under the Model menu, when one or more converters and/or ghosts are currently hidden on the Model diagram.
- **Connectors** – This command causes all hidden connectors which are not connected to converters and/or ghosts to be displayed on the Model diagram. This item is accessible under the Model menu, when Hide Connectors has previously been invoked.
- **Bundled Connectors**– Choosing this command causes all hidden bundled connectors on the Interface layer to be displayed once again. This item is accessible under the Interface menu, when Hide Bundled Connectors has previously been invoked.
- **Pad Icons** – This command causes all hidden Graph Pad and Table Pad icons on the Interface, Map, and Model layers to be displayed once again. It is accessible under the Interface and Model menus, whenever Hide Pad Icons has previously been invoked.
- **Transparent Buttons**– This command causes all hidden transparent buttons to become visible again (the faint dotted line will show), making it possible to edit the button. It is available under the Map and Model menu whenever Hide Transparent Buttons has previously been invoked.
- **Polarity** – This command causes all hidden polarity labels (+/s or -/o) to become visible again for Connectors, Bundled Connectors, Flows, and Bundled Flows.

## **Zoom In**

Zoom In zooms the view of the current layer in to the next level of magnification.

## **Zoom Out**

Zoom Out zooms the view of the current layer out to the previous level of magnification.

**25%**

Zooms the view of the current layer to 25% of the normal size.

**50%**

Zooms the view of the current layer to 50% of the normal size.

**100%**

Zooms the view of the current layer to 100% of the normal size.

**200%**

Zooms the view of the current layer to 200% of the normal size.

## **400%**

Zooms the view of the current layer to 400% of the normal size.

 [Related Topics](#)

# The Interface, Model, and Equation Menus

The menu between the View and Run menus changes, depending upon the layer on which you are operating. On the Interface layer, you'll find the Interface menu. On the Map and Model layers, the menu reads Model. Finally, on the Equation layer, the menu reads Equations. Although most of the commands available on each menu is available, each menu contains a command specific to setting preferences for the layer you are viewing. Each command on the Interface, Model, and Equation menus is briefly described below.

## **Find**

Find is available on all three layers of the software. Choosing this command opens the Find Entity dialog, which contains a scrollable list of model variables. The Find Entity dialog lists all variables which are visible on the model alphabetically, by type. Stocks are listed first, followed by flows, converters, sub-model icons, sectors, and decision process diamonds. For each sub-model or decision process diamond whose space is showing on the diagram, its variables are listed alphabetically, by type.

By selecting a variable in the list, and then clicking on "Find," you'll be taken to the variable in question. From the Interface layer, you'll be taken down to the Model layer. On the Map or Model layers, you'll have the variable highlighted on the screen. On the Equation layer, you'll be taken to the appropriate place in the list of equations. A double-click on a variable in the scrollable list obviates the need for the click on "Find."

---

*Tips:* Up- and down-arrows on your keyboard can be used to scroll through the Find... dialog. Within the Find... dialog, typing a letter on your keyboard will take you to the next model variable in the scrollable list, whose name begins with that letter. Typing two or more letters in the variable name will narrow the search.

---

## **Find Next**

This menu item is enabled on the Map and Model layers, when you have selected a model variable which is ghosted elsewhere in your model. It is also active when you have selected a ghost of a variable. Choosing Find Next will take you to the next ghost of the variable, in order of creation. Find Next thus enables you to walk through your model diagram, locating all appearances of a variable on the diagram.

## **Find Controller**

Find Controller is enabled on the Map and Model layers when you have selected a model variable which has been assigned to a Slider, Knob, Switch, List Input Device or Graphical Input Device on the Interface layer. Variables which have been so assigned will display a small icon of the Slider, Knob, Switch, List Input Device or Graphical Input Device on their surface. Select the variable and choose Find Controller to navigate to the location of the Slider, Knob, Switch, List Input Device or Graphical Input Device on the Interface layer.

## **Align To Grid**

Align To Grid is accessible on the Interface, Map, and Model layers. Choosing Align To Grid causes the selected entities to be aligned to an invisible grid on the diagram. The size of the Model grid may be defined within the [Model Preferences dialog](#).

---

*Note:* Align To Grid does not apply to Sector Frames, Process Frames, Bundled Flows or Bundled Connectors.

## **Open Selection**

This command is available when you select an entity other than a Graph Pad, Loop Pad, or Table Pad.

The Open Selection command is an alternative to double-clicking as a means of opening a selected building block or object on either the Interface, Map, or Model layers. Open Selection also allows you to open a selected model equation from the Equation layer of the model.

For building blocks (Process Frame, Bundled Flow, and Bundled Connector on the Interface layer; Stock, Flow and Converter on the Map and Model layers), Buttons, Sliders, Graphical Input Devices, List Input Devices, Switches, Knobs, Sectors, Numeric Displays, and for Text Boxes, the Open Selection command opens the entity's associated dialog. For more information about the dialogs for building blocks, see [Introduction to Building Blocks](#). For information about the dialogs for Buttons, Sliders, Graphical Input Devices, List Input Devices, Switches, Knobs, Sectors, Numeric Displays, Warning Devices, Graphics Frames, and Text Blocks, see [Introduction to Objects](#).

## **Define Graph**

This command is available when you select a Graph Pad.

The Define Graph command is an alternative to double-clicking on a Graph Pad page within an open Graph Pad. The Define Graph command opens the dialog for defining a Graph Pad. This define dialog is described in [Graph Pad Dialog Operations](#).

## **Define Loop**

This command is available when you select a Loop Pad.

The Define Loop command is an alternative to double-clicking on a Loop Pad page within an open Loop Pad. Selecting the Define Loop command opens the dialog for defining a Loop Pad. This define dialog is described in [Loop Pad Dialog Operations](#).

## **Define Table**

This command is available when you select a Table Pad.

The Define Table command is an alternative to double-clicking on a Table Pad page within an open Table Pad. Choosing the Define Table command opens the dialog for defining a Table Pad. This define dialog is described in [Table Pad Define Dialog Operations](#).

## **Clear Second Position**

Second position is the position taken on by a model building block (stock, flow, converter, connector) whenever a decision process diamond or sub-model space is open. The Clear Second Position command removes all second position information from all building blocks that are not in sub-models or decision process diamonds. Clear Second Position thus is a useful command to employ whenever you find that opening decision process diamonds or sub-models causes your diagram to become very messy. By removing all second position information, Clear Second Position gives you a mechanism for regaining control of the display of the model. For more information on second position, see [Working with Two-Position](#).

## Restore

This commands under the Restore sub menu are accessible when Graphs, Tables, Numeric Displays, or Warning Devices are present, or when you are working with a model which contains input Sliders, Knobs, List Input Devices, Switches, and/or Graphical Input Devices. Each command under the Restore command is described below.

- **All Devices** – Choosing this command simultaneously restores Graphs, Tables, Sliders, Graphical Input Devices, Knobs, List Input Devices, Switches, Warning Devices and Numeric Displays.
- **Graphical Inputs** – Choosing this command causes the relationship for each Graphical Input Device to be restored to the relationship which exists in its associated graphical function.
- **Graphs & Tables** – Choosing this command clears the data from all Graph and Table Pad pages.
- **Knobs** – Choosing this command causes the value reported by each Knob on the Interface layer to be restored to the original numeric value entered in the equation box of its controlled entity.
- **Numeric Displays** – Choosing this command causes the value in each Numeric Display device to be cleared from the device. Numeric Displays thus are restored to a pristine state.
- **Sliders** – Choosing this command causes the value reported by each Slider on the Interface layer to be restored to the original numeric value entered in the equation box of its controlled entity. Restore Sliders has no effect on Sliders which are overriding an equation.
- **LIDs** – Choosing this command causes the value reported by each variable in the List Input Device (LID) on the Interface layer to be restored to the original numeric value entered in the equation box of its associated entities. Restore LIDs has no effect on variables in LIDs that are defined with an equation.
- **Switches** – Choosing this command resets the switch to 1 if the corresponding diagram converter is set to 1, or to 0 if the corresponding diagram converter is set to any other value.
- **Warning Devices** – Choosing this command causes the value in each Warning Device to be cleared from the device. Warning Devices thus are restored to a pristine state.

## **Interface Prefs / Model Prefs / Equation Prefs**

The Interface Prefs, Model Prefs, and Equation Prefs commands allow you to specify your preferences for visual characteristics of the specified layer. You can set preferences for a layer only when you are viewing the layer. For more information, see [Interface Preferences](#), [Model Preferences](#), and [Equation Preferences](#).

## **Array Editor**

The Array Editor is used to create dimensions and elements which you can then use to create arrayed variables. For more information, see [Array Editor](#).

## Unit Editor

The Unit Editor command displays the Unit Editor dialog. The Unit Editor dialog contains a list of all units of measure available to your model. In addition, the Unit Editor dialog provides a mechanism for creating new units of measure for your model. For more information, see [Working with Units](#).

 [Related Topics](#)

# The Run Menu

The commands on the Run menu enable you to launch, pause, resume, and stop regular runs, module runs, sector runs, and sensitivity runs. It also enables you to configure the time and simulation specifications for your model, and to set numerical scales for selected model variables.

Each command on the Run menu is briefly described below.

## Run

The Run command launches or resumes a simulation run conforming to the specifications given in the Module Specs, Sector Specs, and Sensi Specs dialog boxes. [Figure 3-21](#) shows the different permutations of this important menu item.

## **Pause**

Pause pauses a simulation run. Choosing Resume, or clicking the Run button on the Run Controller, while the simulation is paused, will resume the simulation.

---

*Tip:* To pause a simulation while it is in progress, you can also click once on the Pause button on the [Run Controller](#).

---

## **Stop**

Stop causes the simulation to cease its execution.

## Module Specs

The Module Specs command allows you to select which modules will be run in a simulation. You can select to run the entire model, including all modules (**Run This Model**), or just selected modules (**Run Selected Modules**). For more information, see [Running Modules](#).

## Sector Specs

Sector Specs takes you to the Sector Specs dialog. When the sector tool has been used to define model sectors, the Sector Specs dialog will enable you to specify the model sectors which will be run in a simulation. You can select to run the entire model, including all sectors (**Run Entire Model**), or just selected sectors (**Run Selected Sectors**). For more information, see [Sector Specs](#).

## **Sensi Specs**

Sensi Specs opens the Sensitivity Specs dialog box. This dialog enables you to establish the characteristics for a set of sensitivity runs. It also enables you to "turn on" sensitivity analysis. For more information, see [Sensi Specs](#).

## Run Specs

Run Specs opens the Run Specs dialog box, which enables you to specify the simulation length, the time step between calculations (DT), the interval between simulation pauses, the integration method, the time unit for the model, and the run mode (Normal or Cycle-time) for the model. For more information, see [Run Specs](#).

## **Range Specs**

The Range Specs command opens the Range Specs dialog box, which provides you with a listing of all model variables, and their most recent min/max global scales. For more information, see [Range Specs](#).

## Check Units

The Check Units command causes the software to perform a unit consistency check on all model variables. The software will highlight any model variables that fail the unit analysis check. If you specify units of measure in your models, it's a good idea to check units periodically during the model development cycle. Checking units in this fashion is an excellent way to minimize silly errors (algebraic or otherwise) that can be introduced into a model during its development.

For details on specifying units of measure, see [Working with Units](#).

 [Related Topics](#)

# The Help Menu

The commands on the Help menu give you access to **iThink** or **STELLA** Help and to other information that will help you use the software.

Each command on the Help menu is briefly described below.

## iThink Help / STELLA Help

Displays *iThink* or **STELLA** Help.

## **isee systems Website**

Navigates you to the isee systems Website.

## **Report a Problem**

Navigates you to the Problem Submission Form of the isee systems Website so that you can report a problem or bug you find with the software.

## About iThink / About STELLA (Windows)

Displays the version number of the software and your registration number.

---

Note: To view this information in the Macintosh version, choose the **About iThink** or **About STELLA** command from the iThink or STELLA menu.

---

 [Related Topics](#)

# Introduction to Toolbars

Each layer includes a toolbar at the top of the window that gives you access to the [building blocks](#) and [tools](#) you can use on the layer to create and modify your model.

The topics in this section describe the toolbar for each layer:

- [Interface Layer Toolbar](#)
- [Map and Model Layer Toolbar](#)
- [Equation Layer Toolbar](#)

 [Related Topics](#)

# Interface Layer Toolbar

Use the Interface layer toolbar to create end-user interfaces for your model. Each tool on the Interface layer toolbar is described below.

## **Process Frame**

Click this tool to [create a Process Frame](#). A Process Frame allows you to represent high-level processes. It facilitates a "top-down" approach to model construction. It also provides capabilities for navigating to the associated Sector Frame and its stock/flow structure on the Map and Model layers.

## **Bundled Flow**

Click this tool to [create a Bundled Flow](#). The Bundled Flow allows you to represent, at a high level, the material flows between processes in your model. Like the Process Frame, the Bundled Flow facilitates a "top-down" approach to model construction. It also provides navigational capabilities for finding sector to sector flows on the Model layer when "Link High-Level Map to Model" is checked in the Interface Prefs.

 **Bundled Connector**

Click this tool to [create a Bundled Connector](#). The Bundled Connector allows you to represent, in summary form on the Interface layer, any sector-to-sector connectors which exist in your model. Like the other mapping building blocks, the Bundled Connector facilitates a "top-down" approach to model construction, and provides navigational capabilities.



## Loop Pad

Click this tool to [create a loop diagram](#). Loop diagrams are simple pictures that identify the cause and effect processes that work to generate dynamic behavior patterns. The software's Loop Pad Object enables you to create these pictures based on the underlying model structure.



## Button

Click this tool to [create a button](#). Click and hold this tool to select from a menu of different button types. Buttons are designed to facilitate an end-user's interaction with your model by reducing the level of software skills and language fluency needed. When "pushed," buttons perform one of several operations such as navigating to a new location, executing a menu command, or providing pop-up information.

## Switch

Click this tool to [create a switch](#). As its name suggests, the switch enables you to turn things on or off in a model. Specifically, you can assign a switch to one or more converters in your model, to a sector in your model, or to the sensitivity analysis setup dialog. The switch thus enables you as a model builder to create a friendly device for users to control a model.

## **Graphical Input Device**

Click this tool to [create a Graphical Input Device](#). A Graphical Input Device enables model users to: (1) see, at a glance, the shape of a graphical function, (2) edit a graphical function from the Interface layer; (3) restore a graphical function to its author defined relationships; and (4) animate graphical functions during a simulation.



## Knob Input Device

Click this tool to [create a knob](#). The knob is useful principally for providing initial values for stocks. Knobs also can be used to adjust values for constants. Unlike the slider, the knob cannot be adjusted during the course of a simulation. It is set prior to the outset of a simulation and remains fixed throughout the simulation. Variables which contain equations or graphical functions can not be assigned to a knob.



## List Input Device

Click this tool to [create a List Input Device](#) (LID). The List Input Device is a simple spreadsheet-like input device. Users of your models can use the LID to set the values for converters and flows in the model. In addition, users can use the LID to set the initial values for stocks in the model. Multiple model variables can be assigned to a single LID. A single LID can contain multiple pages.



## Slider Input Device

Click this tool to [create a Slider Input Device](#). The Slider Input Device allows model users to adjust constant values, and to override equation logic (and graphical function relationships) with numerical inputs.

 **Graph Pad**

Click this tool to [create a Graph Pad](#). You use the Graph Pad as a repository for plotting data generated by model simulation runs. The software supports three basic types of graphs: time series plots (X, Y, Z... over time), scatter plots (X vs. Y), and bar graphs.



## Table Pad

Click this tool to [create a Table Pad](#). You use the Table Pad to display the numerical output from your simulations, and as a locus for exporting data to an Excel worksheet.

## Status Indicator

Click this tool to [create a status indicator](#). The status indicator is an Interface layer object that you can use to provide information about the status of key outputs in your simulation model. You can configure a status indicator to display as a simple lamp, or as a speedometer-type gauge. In either case, the indicator will light up using green, yellow, or a flashing red color to indicate the status of the key output.



## Numeric Display

Click this tool to [create a numeric display](#). You use a Numeric Display to display the current output associated with assigned model variables. The Numeric Display is useful for getting a precise reading of what's going on in the model, as a simulation unfolds.

## **Text Box**

Click this tool to [create a text box](#). The purpose of the Text Box is to hold text. The basic activities associated with Text Boxes are creating, moving, resizing, and typing text. Within each box's dialog you can exercise a variety of controls and options.

## **Graphics Frame**

Click this tool to [create a Graphics Frame](#). Graphics Frames are useful in framing objects such as input/output devices and buttons. They also let you import pictures, graphics and movies.

## **Arrow**

Click this tool to [use the Arrow tool](#). The Arrow is a general-purpose editing tool. You use it to select, move, open, and edit building blocks and objects. You also use it to manipulate Sub-model and Space Compression spaces on the Map and Model layers, and to select items on the Equation layer. In accomplishing these tasks, the point of the arrow is the "hot spot."



## Paintbrush

Click this tool to [use the Paintbrush tool](#). The purpose of the Paintbrush is to add color to model elements. With it, you can color building blocks, objects, backgrounds, and output from your model.



## Dynamite

Click this tool to [use the Dynamite tool](#). The Dynamite is used to clear building blocks and objects from the Interface, Map, and Model layers. It will also clear equations (with their associated building block representation) from the Equation layer.

[Related Topics](#)

# Map and Model Layer Toolbar

Use the Map and Model layer toolbar to lay out a map of your model (on the Map layer) and to build a model that can be simulated on your computer (on the Model layer). Note that the Map and Model layers have the same toolbar, although the options available for some tools differ depending on which layer you are currently viewing. Each tool on the Map and Model layer toolbar is described below.

 **Stock**

Click this tool to [create a stock](#). Click and hold this tool to select from a menu of different stock types. Stocks are accumulations. They collect whatever flows into them, net of whatever flows out of them.



## Flow

Click this tool to [create a flow](#). Click and hold this tool to select from a menu of different flow types. The job of flows is to fill and drain accumulations. The unfilled arrow head on the flow pipe indicates the direction of positive flow.

 **Converter**

Click this tool to [create a converter](#). Click and hold this tool to select from a menu of different converter types. The converter holds values for constants, defines external inputs to the model, calculates algebraic relationships, and serves as the repository for graphical functions. In general, it converts inputs into outputs.



## Action Connector

Click this tool to [create a connector](#). Click and hold this too lto select from a menu of different connector types. The job of the connector is to connect model elements.

## **Module**

Click this tool to [create a module](#). Modules are self-contained models that you can connect to other models. Modules allow you to break a single model into well-defined "chunks". Click and hold this tool to select to [create a Decision Process Diamond](#). The Decision Process Diamond (DPD) is a mechanism for managing the diagram complexity associated with the representation of decision processes within your models.



## Button

Click this tool to [create a button](#). Click and hold this tool to select from a menu of different button types. Buttons are designed to facilitate an end-user's interaction with your model by reducing the level of software skills and language fluency needed. When "pushed," buttons perform one of several operations such as navigating to a new location, executing a menu command, or providing pop-up information.

## **Sector Frame**

Click this tool to [create a Sector Frame](#). On the Map and Model layers, you'll use the Sector Frame to perform two major functions. Primarily, you'll use it to group together functionally related chunks of model structure. Secondarily, you can use it to display graphical images or to play QuickTime movies.

## **Graph Pad**

Click this tool to [create a Graph Pad](#). You use the Graph Pad as a repository for plotting data generated by model simulation runs. The software supports three basic types of graphs: time series plots (X, Y, Z... over time), scatter plots (X vs. Y), and bar graphs.

## **Table Pad**

Click this tool to [create a Table Pad](#). You use the Table Pad to display the numerical output from your simulations, and as a locus for exporting data to an Excel worksheet.

## **Status Indicator**

Click this tool to [create a status indicator](#). The status indicator is an Interface layer object that you can use to provide information about the status of key outputs in your simulation model. You can configure a status indicator to display as a simple lamp, or as a speedometer-type gauge. In either case, the indicator will light up using green, yellow, or a flashing red color to indicate the status of the key output.

## **Numeric Display**

Click this tool to [create a numeric display](#). You use a Numeric Display to display the current output associated with assigned model variables. The Numeric Display is useful for getting a precise reading of what's going on in the model, as a simulation unfolds.

## **Text Box**

Click this tool to [create a Text Box](#). The purpose of the Text Box is to hold text. The basic activities associated with Text Boxes are creating, moving, resizing, and typing text. Within each box's dialog you can exercise a variety of controls and options.

## **Graphics Frame**

Click this tool to [create a Graphics Frame](#). Graphics Frames are useful in framing objects such as input/output devices and buttons. They also let you import pictures, graphics and movies.

## **Arrow**

Click this tool to [use the Arrow tool](#). The Arrow is a general-purpose editing tool. You use it to select, move, open, and edit building blocks and objects. You also use it to manipulate Sub-model and Space Compression spaces on the Map and Model layers, and to select items on the Equation layer. In accomplishing these tasks, the point of the arrow is the "hot spot."

## **Paintbrush**

Click this tool to [use the Paintbrush tool](#). The purpose of the Paintbrush is to add color to model elements. With it, you can color building blocks, objects, backgrounds, and output from your model.



## Dynamite

Click this tool to [use the Dynamite tool](#). The Dynamite is used to clear building blocks and objects from the Interface, Map, and Model layers. It will also clear equations (with their associated building block representation) from the Equation layer.

 **Ghost**

Click this tool to [use the Ghost tool](#). The purpose of the Ghost tool is to make replicas, aliases, or shortcuts for individual stocks, flows, and converters. A Ghost of an entity has no independent identity - it is simply an image of the building block from which it was ghosted.

 [Related Topics](#)

# Equation Layer Toolbar

Use the Equation layer toolbar to view the equations that make up your model. Each tool on the Equation layer toolbar is described below.



## Arrow

Click this tool to [use the Arrow tool](#). The Arrow is a general-purpose editing tool. You use it to select, move, open, and edit building blocks and objects. You also use it to manipulate Sub-model and Space Compression spaces on the Map and Model layers, and to select items on the Equation layer. In accomplishing these tasks, the point of the arrow is the "hot spot."



## Paintbrush

Click this tool to [use the Paintbrush tool](#). The purpose of the Paintbrush is to add color to model elements. With it, you can color building blocks, objects, backgrounds, and output from your model.



## Dynamite

Click this tool to [use the Dynamite tool](#). The Dynamite is used to clear building blocks and objects from the Interface, Map, and Model layers. It will also clear equations (with their associated building block representation) from the Equation layer.

[Related Topics](#)

# Introduction to Building Blocks

Building blocks are the tools you use to construct the model on each layer.

The Interface layer provides three building blocks:

- [The Process Frame](#)
- [The Bundled Flow](#)
- [The Bundled Connector](#)

The Map and Model layers provide five building blocks:

- [The Stock](#)
- [The Flow](#)
- [The Converter](#)
- [The Connector](#)
- [The Module](#)

This section describes the mechanical aspects of using these building blocks.

At the end of this section, you'll also find sections that detail some of the special-purpose capabilities of the software's building blocks, including [The Message Poster](#), [Working with Units](#), and [Reading Numerical Values Directly from Model Elements](#).

 [Related Topics](#)

# Interface Layer Building Blocks

The topics in this section describe the basic operation of the Interface layer building blocks:

- [The Process Frame](#)
- [The Bundled Flow](#)
- [The Bundled Connector](#)

In the section, you'll learn:

- The purpose of each building block.
- How to select and place each building block.
- How to do building block operations on the map surface.
- How to operate the building block's dialog.

---

**Note:** When creating a High-Level Map, you may choose either to have the software enforce a one-to-one correspondence with the diagram or not. If "Link High-Level Map to Model" is checked in the Interface Prefs dialog or in the Default Settings, the software will not run a model until your Process Frame/Bundled Flow Connector Map on the Interface layer is in one-to-one correspondence with your Sector Frame/ stock/ flow/ converter/ connector diagram on the Model layer. When the link is not checked, you may build the structures independently of each other.

 [Related Topics](#)

# Process Frame

**Purpose:** The Process Frame allows you to represent high-level processes. It facilitates a "top-down" approach to model construction. It also provides capabilities for navigating to the associated Sector Frame and its stock/flow structure on the Map and Model layers.

## Selection and Placement:

Method 1:

1. Select the Process Frame by clicking once on the icon in the Building Block palette.
2. Move the mouse to the desired location on the Interface level.
3. Click once to deposit the Process Frame.

Method 2:

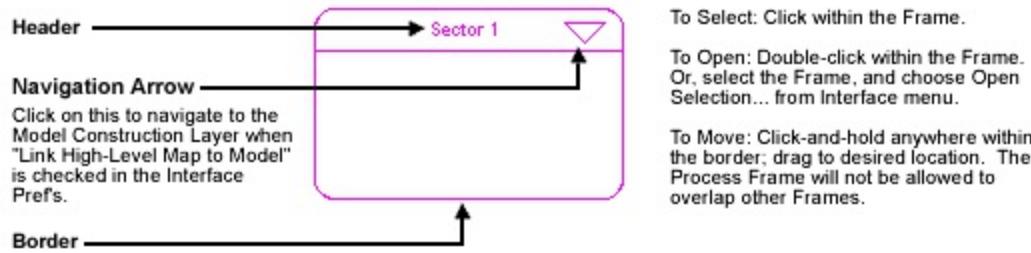
When "Link High-Level Map to Model" is checked in the Interface Prefs dialog:

- Create a Sector Frame on the Map and Model layers. A Process Frame will be created automatically on the Interface layer.

**Surface Operations:** Figure 4-3 details surface operations which may be performed on a Process Frame.

*Figure 4-3 Process Frame Surface Operations*

### (a) Unselected Process Frame

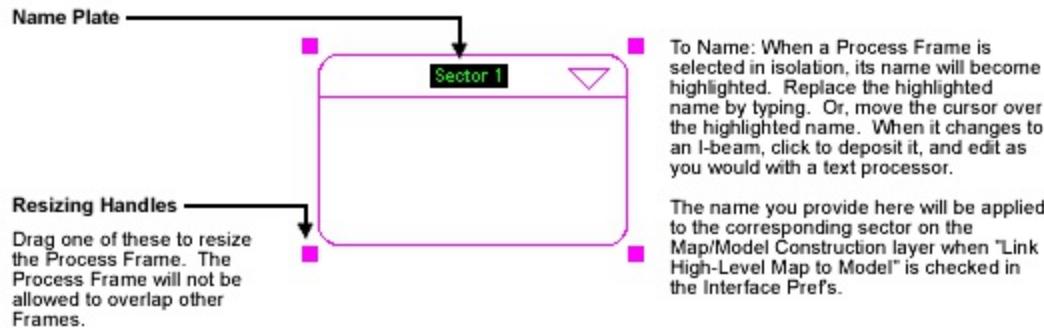


To Select: Click within the Frame.

To Open: Double-click within the Frame. Or, select the Frame, and choose Open Selection... from Interface menu.

To Move: Click-and-hold anywhere within the border; drag to desired location. The Process Frame will not be allowed to overlap other Frames.

### (b) Selected Process Frame



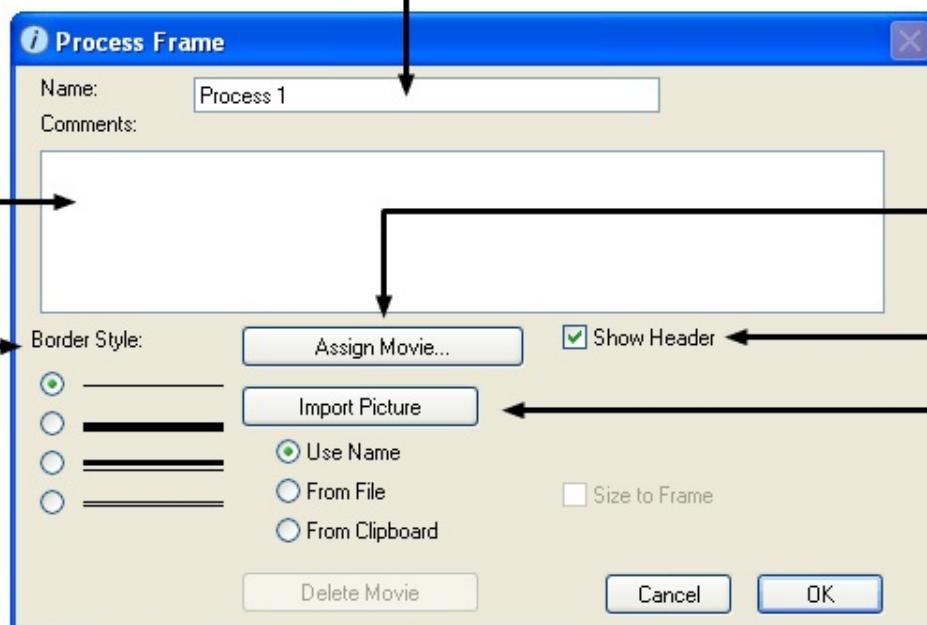
To Name: When a Process Frame is selected in isolation, its name will become highlighted. Replace the highlighted name by typing. Or, move the cursor over the highlighted name. When it changes to an I-beam, click to deposit it, and edit as you would with a text processor.

The name you provide here will be applied to the corresponding sector on the Map/Model Construction layer when "Link High-Level Map to Model" is checked in the Interface Pref's.

**Dialog Operations:** When you open a Process Frame, the Process Frame dialog appears. Figure 4-4 details the Process Frame dialog.

*Figure 4-4  
Process Frame Dialog*

Use this space to edit the name of the Process Frame. The name will also appear in the Sector Frame when the "Link High-Level Map to Model" check box is selected in Interface Prefs.



Available when a building block is in the associated Sector and your computer supports QuickTime. The operation is identical to assignment of movies in Sector Frames.

Select this check box to display the Process Frame header. The header displays the Process Frame name.

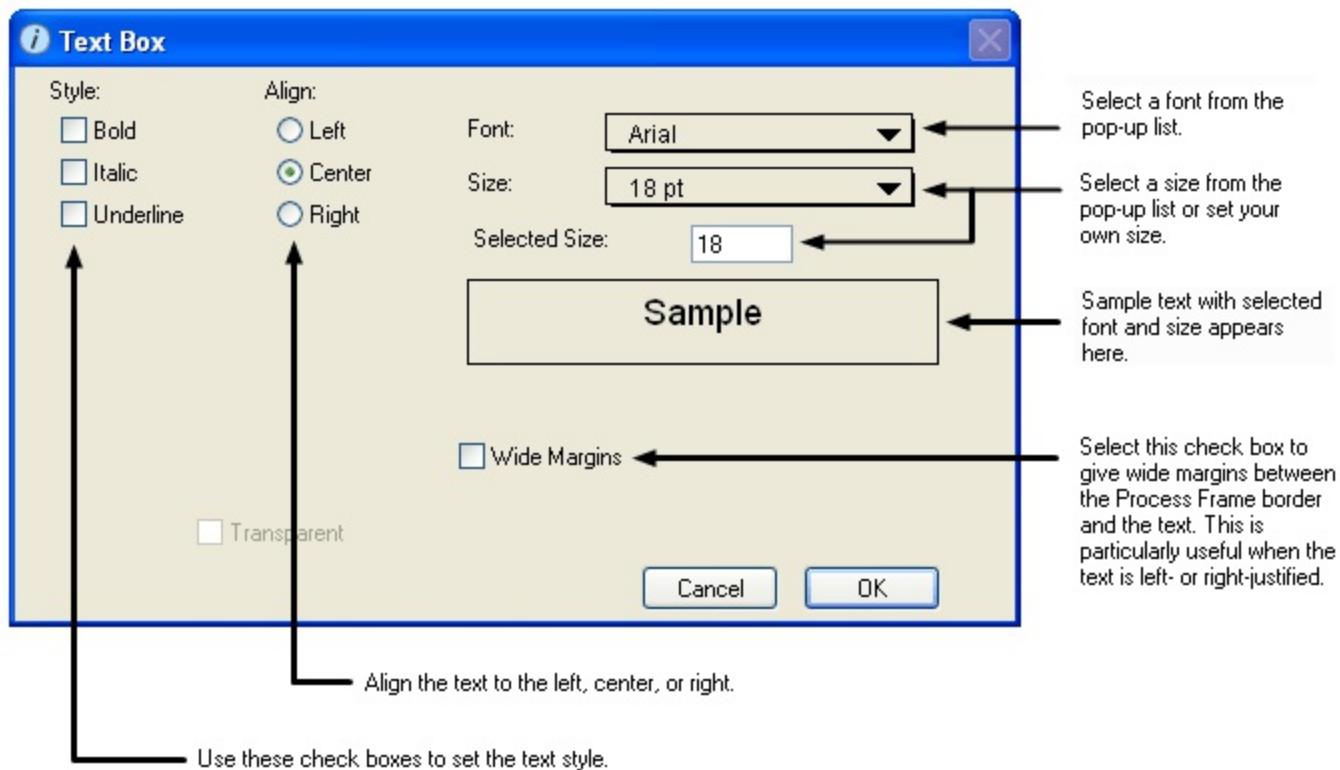
Click this button to import a picture onto the face of the Process Frame. You can use the name of the frame, import an image from the Clipboard, or import an image file. An asterisk (\*) appears on the button to indicate a successful import. See Figure 4-5 regarding the "Use Name" option.

Available when a movie or picture has been assigned. Deletes the imported picture or movie from the Process Frame.

Sizes the imported picture or movie to the Process Frame.

**Process Frame Dialog Notes:** The "Use Name" option can be quite helpful in displaying your high-level maps. When you have selected the "Use Name" option, a dialog like the one shown in Figure 4-5 will appear when you click the Import Picture button. This dialog allows you to set the font, size, and alignment characteristics of the sector's name, as it will appear within the main portion of the Process Frame.

*Figure 4-5  
Configuring Text when "Use Name" is in Effect*



 [Related Topics](#)

# Bundled Flow

**Purpose:** The Bundled Flow allows you to represent, at a high level, the material flows between processes in your model. Like the Process Frame, the Bundled Flow facilitates a "top-down" approach to model construction. It also provides navigational capabilities for finding sector to sector flows on the Map and Model layers when "Link High-Level Map to Model" is checked in the Interface Prefs.

## Selection and Placement:

Method 1:

1. Select the Bundled Flow by clicking once on the icon in the Building Block palette.
2. Move the mouse to the center of a Process Frame.
3. Click-and-hold. Drag the Bundled Flow into another Process Frame (the Process Frame border will turn gray on contact). Release the click.

Method 2:

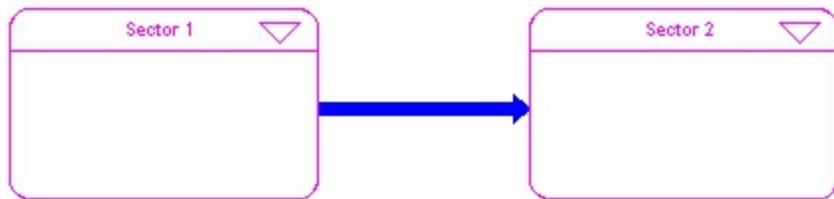
When "Link High-Level Map to Model" is checked in the Interface Prefs dialog:

- Drag a flow between two Sector Frames on the Map or Model layer diagram. A Bundled Flow will be created automatically on the Interface layer.

**Surface Operations:** Figure 4-6 details surface operations which may be performed on a Bundled Flow.

*Figure 4-6 Bundled Flow Surface Operations*

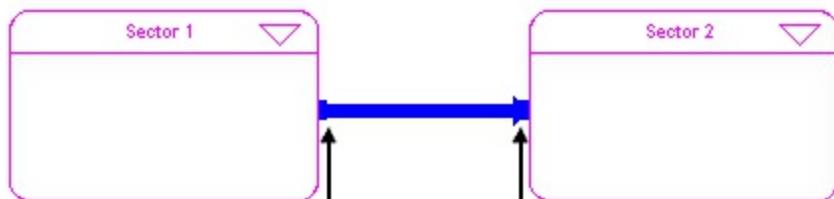
### (a) Unselected Bundled Flow



To Select: Click once on the Flow.

To Open: Double-click on the Flow. Or, select it, and choose Open Selection... from Map menu. When "Link High-Level Map to Model" is checked in Interface Prefs you will be able to access a list of all Model Construction layer flows which are "bundled" in the bundled flow, on the Interface level.

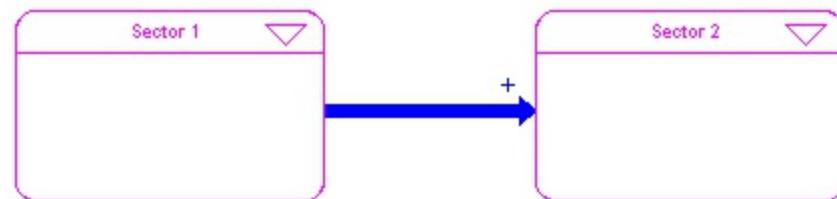
### (b) Selected Bundled Flow



#### Move Handles

Drag one of these to move the Bundled Flow. The movement will be constrained by the boundaries of the Process Frames to which it is attached.

### (c) Choose Polarity



To Choose Polarity: Right-click the Flow, choose Polarity, and then choose +/s, -/o, or None.

---

**Notes:** To bend a flow pipe, depress the shift key as you draw the flow. Each time you depress the shift key you will get a ninety degree bend.

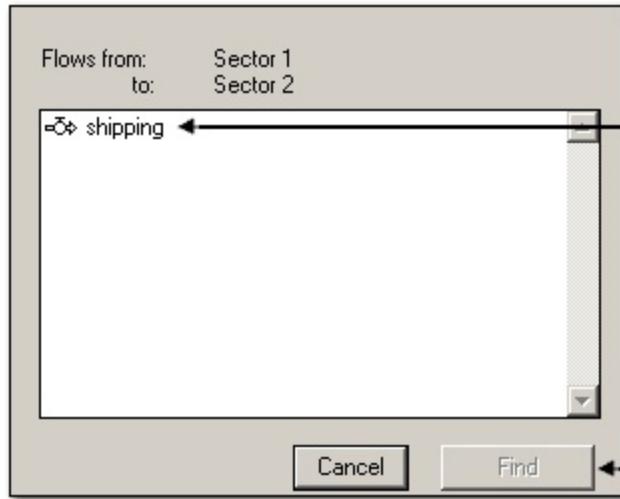
On the Interface layer, you can have a maximum of one Bundled Flow from Process Frame A to Process Frame B, and a maximum of one Bundled Flow from Process Frame B to Process Frame A.

This means that the software will enforce a one-to-one correspondence between sector-to-sector flows, and process-to-process Bundled Flows when "Link High-Level Map to Model" is checked in the Interface Prefs

---

**Dialog Operations:** When you open a Bundled Flow, you'll enter its associated dialog. As shown in Figure 4-7, the Bundled Flow lists sector-to-sector flows on the Map and Model layers, and provides a mechanism for navigating to those flows.

*Figure 4-7  
The Bundled Flow Dialog*



Scrollable list contains sector-to-sector flows which are represented in summary by the Bundled Flow only when "Link High-Level Map to Model" is checked in the Interface Pref's.

Select a flow in the list, and then click Find to navigate to the flow on the Model Construction layer.

## [Related Topics](#)

# Bundled Connector

**Purpose:** The Bundled Connector allows you to represent, in summary form on the Interface layer, any sector-to-sector connectors which exist in your model. Like the other mapping building blocks, the Bundled Connector facilitates a "top-down" approach to model construction, and provides navigational capabilities.

## Selection and Placement:

Method 1:

1. Select the Bundled Connector by clicking once on the icon in the Building Block palette.
2. Move the mouse to the center of a Process Frame.
3. Click-and-hold. Drag the Bundled Connector into another Process Frame (the Process Frame border will turn gray on contact). Release the click.

Method 2:

When "Link High-Level Map to Model" is checked in the Interface Prefs dialog:

- Drag a connector between variables in two Sector Frames on the Map or Model layers. A Bundled Connector will be created automatically.

---

**Notes:** To bend a connector, depress the shift key as you drag the connector. Each time you depress the shift key you will get a ninety degree bend.

On the Interface layer, you can have a maximum of one Bundled Connector from Process Frame A to Process Frame B, and a maximum of one Bundled Connector from Process Frame B to Process Frame A.

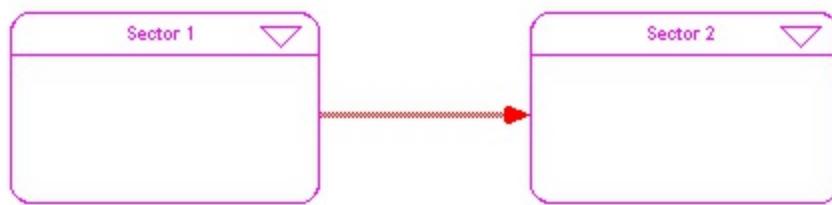
When "Link High-Level Map to Model" box is checked in the Interface Prefs dialog, the software will enforce a one-to-one correspondence between sector-to-sector connectors, and process-to-process Bundled Connectors. At run time, Bundled Connectors will automatically be created if they are needed. If connectors are needed on the Map or Model layer, an alert will list your options.

---

**Surface Operations:** Figure 4-8 details surface operations which may be performed on a Bundled Connector.

*Figure 4-8  
Bundled Connector Surface Operations*

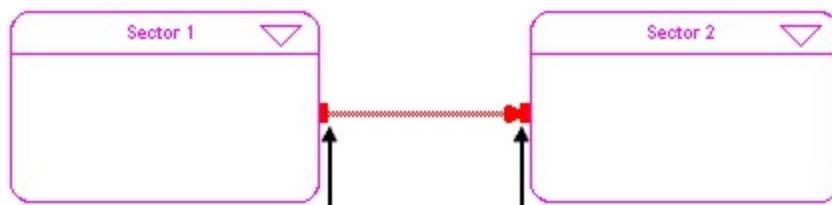
### (a) Unselected Bundled Connector



To Select: Click once on the Connector.

To Open: Double-click on the Connector. Or, select it, and choose Open Selection... from Map menu. When "Link High-Level Map to Model" is checked in Interface Pref's you will have access to a scrollable list of all sector-to-sector connections between entities on the Map/Model Construction layer.

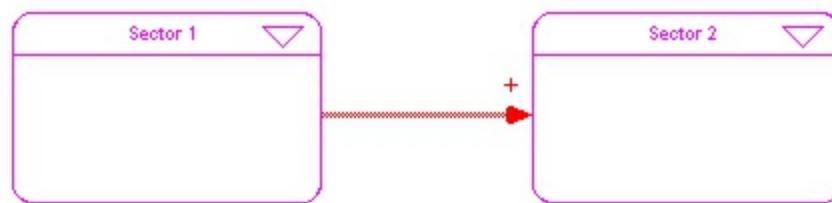
### (b) Selected Bundled Connector



#### Move Handles

Drag one of these to move the Bundled Connector. The movement will be constrained by the boundaries of the Process Frames to which it is attached.

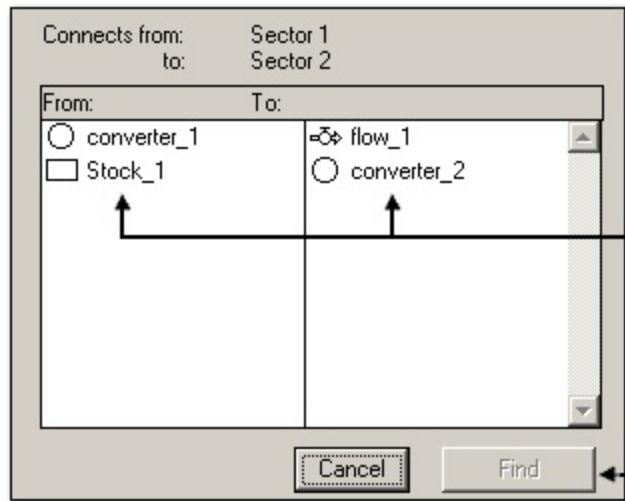
### (c) Choose Polarity



To Choose Polarity: Right-click the Connector, choose Polarity, and then choose +/s, -/o, or None.

**Dialog Operations:** When you open a Bundled Connector, you'll enter its associated dialog. As shown in Figure 4-9, the Bundled Connector lists sector-to-sector connectors on the Map and Model layers, and provides a mechanism for navigating to the entities to which the connectors are attached.

*Figure 4-9  
The Bundled Connector Dialog*



Scrollable list contains sector-to-sector From/To connections, represented in summary form by the Bundled Connector.

Select an entity in the list, and then click Find to navigate to it on the Model Construction layer.

## [Related Topics](#)

# Map and Model Layer Building Blocks

The topics in this section describe the basic operation of the Map and Model layer building blocks:

- [The Stock](#)
- [The Flow](#)
- [The Converter](#)
- [The Connector](#)
- [The Module](#)

In the section, you'll learn:

- The purpose of each building block.
- How to select and place each building block.
- How to do surface operations on the building block.
- How to operate the building block's dialog, on both the Map and Model layers.

Note that for stocks, the discussion is limited to the Reservoir stock type. Configuration templates for Conveyor, Queue, and Oven stock types (as well as for their associated flows) can be found in the sections on Conveyors, Queues, and Ovens.

 [Related Topics](#)

# Stocks

**Purpose:** Stocks are accumulations. They collect whatever flows into them, net of whatever flows out of them.

## Selection and Placement:

1. Select the stock icon by clicking once on the icon in the Building Block palette.
2. Move the mouse to the desired location on the diagram.
3. Click once to deposit the stock.

---

**Notes:** The default stock type is the Reservoir. Think of a Reservoir as a pool of water, or as an undifferentiated pile of "stuff." A Reservoir passively accumulates its inflows, minus its outflows. Any units which flow into a Reservoir will lose their individual identity. Reservoirs mix together all units into an undifferentiated mass as they accumulate.

The default, Reservoir, stock type is signified by a simple rectangle. To select a conveyor, sub-model, queue, or oven, click and hold the stock icon on the palette. Then, select the appropriate stock type from the drop-down list that appears

---

**Surface Operations:** Figure 4-11 details surface operations which may be performed on a stock.

*Figure 4-11  
Stock Surface Operations*

a) Unselected Stock

Name Plate → Noname 1



To Select: Click within the icon's border.

To Open: Double-click within the icon's border. Or, select the icon, and choose Open Selection from the Model menu. Opening will take you to the stock's Define dialog.

To Move: Click-and-hold within the icon's border; drag to the desired location.

To Move Name Plate: Click-and-hold the name plate of the icon. Drag the name plate around the border of the stock to the desired location. Hold Control key (Windows) or Command key (Macintosh) while dragging to constrain position of the name plate to North, South, East or West. Control-click (Windows) or Command-click (Macintosh) on the name plate to cause it to snap to the nearest cardinal position. To move the name plate to the center of the stock, simply drag it there.

b) Selected Stock

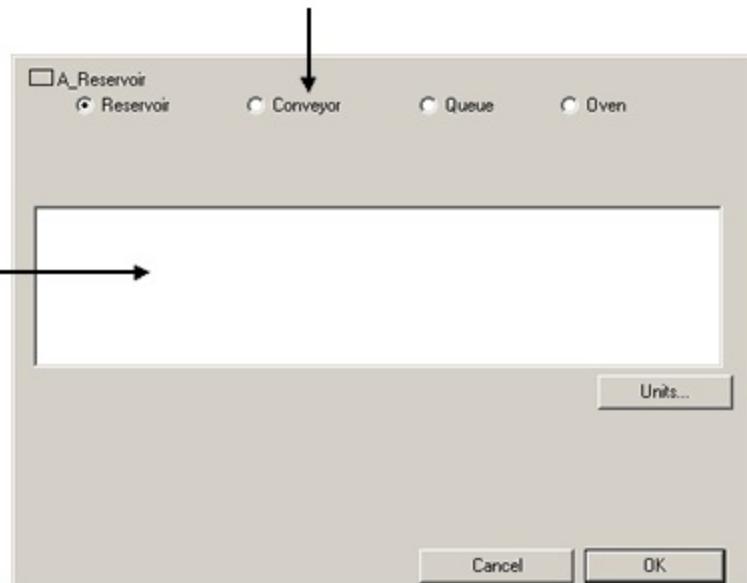


To Name: Click the name plate to select it; its name will become highlighted. Replace the highlighted name by typing. Or move the Arrow over the highlighted name. When the cursor changes to an I-beam, click to deposit it. Then edit as you would with a text processor. For multiple-line variable names, hit the carriage return key on your keyboard.

**Stock Dialog on Map Layer Operations:** When you open a Stock on the Map layer, you'll enter the map version of the Stock dialog. Figure 4-12 shows the options available in the Stock dialog on the Map layer when the Reservoir option is selected.

*Figure 4-12  
Stock Dialog - Reservoir - Map Layer*

Radio buttons allow you to choose among the different stock types [note that it is also possible to select a stock type directly from the drop-down list in the building block palette]. Reservoir is the default stock type. See the book on Conveyors, Queue and Ovens for a detailed discussion of each.



**Stock Dialog on Model Layer Operations:** When you open a stock on the Model Layer, the Stock dialog sports additional controls. Figure 4-13 shows the options available in the Stock dialog on the Model Layer when the Reservoir option is selected.

*Figure 4-13  
Stock Dialog - Reservoir - Model Layer*

Calculator: Click on keys to cause numbers and operators to appear in the equation box. Depress alt key (Windows) or option key (Macintosh) for additional operators.

When checked, Non-negative ensures that flows will not force the Reservoir to take on negative values.

Use this to replicate parallel details using a single iconic representation on the diagram. See Chapter 11 for details.

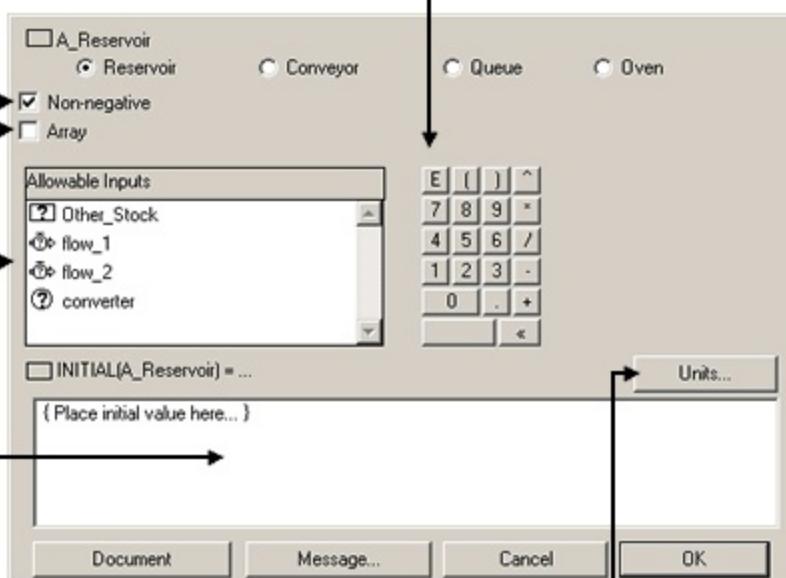
List of Allowable Inputs for defining the initial value of a Reservoir. Click on an item in the list, and it will appear in the equation box

Equation Box: Click on allowable inputs, numbers and operators to define initial value for Reservoir. Initial values can use a constant or an algebraic expression which evaluates to a single value. It is possible to enclose text within brackets {}.

Document button enables you to access the documentation field in the dialog. A \* will appear on the button when documentation is present.

Use the Message button to post messages. See appendix at the end of this Chapter for details.

Units button enables you to set the units of measure for each model entity. In addition you can perform a unit consistency check on the equation. See end of this chapter for details.



## Stock Dialog Notes:

- The expression in the equation box provides an initial value for the stock, and is evaluated only at the outset of a simulation run.
- You can generate initial and subsequent values for stocks via a link from an Excel spreadsheet by using the data import features. For more information, see [Introduction to Importing and Exporting](#).

 [Related Topics](#)

# Flows

**Purpose:** The job of flows is to fill and drain accumulations. The unfilled arrow head on the flow pipe indicates the direction of positive flow.

## Selection and Placement:

1. Select the flow icon by clicking once on the icon in the Building Block palette. Note that the default flow type is the uniflow, which flows in one direction only. To select a biflow from the drop-down palette, click and hold the flow icon, and select the bi-directional flow.
2. Move the mouse to the desired starting location on the page.
3. Click-and-hold.
4. Drag the cursor to the place where the flow will end. Release the click. As you drag, the flow will follow your cursor movement.

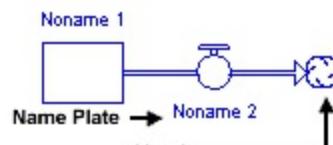
## Selection and Placement Notes

- *To bend a flow pipe*, depress the shift key and change the direction of mouse movement as you drag the flow. Each time you depress the shift key, a 90 degree bend will be put in the flow pipe.
- *To draw an outflow from a stock*, be sure to start with the cursor inside of the stock. If you are not within the boundary of the stock when you begin dragging, your flow will be drawn with a cloud at its source.
- *To draw an inflow to a stock*, make sure that your cursor makes contact with the stock before you release your click. The stock will turn gray on contact to let you know to release your click. If you release the click prematurely, a cloud will appear at the destination end of the flow pipe.
- *To replace a cloud with a stock*, select the stock with the Arrow tool. Drag the stock over the cloud. When the cursor (the tip of the index finger on the hand) is directly atop the cloud, the cloud will turn gray. Release your click, and the flow will be connected to the stock. The cloud will disappear.
- *To disconnect a stock from a flow*, select the flow with the Arrow tool, then click the move handle at the end of the flow that is connected to the stock and drag the handle away from the stock.

**Surface Operations:** Figure 4-14 details surface operations which may be performed on a flow.

Figure 4-14 Surface Operations on Flows

### (a) Unselected Flow



To Select: Click within the circle portion of the icon.

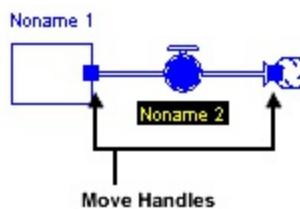
To Open: Double-click within the circle portion of the icon. Or, select the icon, and choose Open Selection... from the Model menu. Opening will take you to the Flow's define dialog.

To Select and/or Move Cloud: Click on the cloud and select it. Drag the cloud to the desired location. The Flow will remain attached.

To Change Direction of Flow: Control-click (Windows) or command-click (Macintosh) on flow arrow-head.

To Move Name Plate: Click-and-hold the name plate of the unselected icon. Drag the icon around to the desired location. Hold control key (Windows) or command key (Macintosh) while dragging to constrain position of the name plate to North, South, East or West. Control-click (Windows) or Command-click (Macintosh) on the name plate to cause it to snap to the nearest cardinal position.

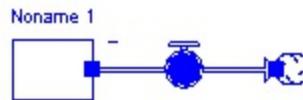
### (b) Selected Flow



To Name: When a single icon is selected, its name will become highlighted. Replace the highlighted name by typing. Or, move the Hand over the highlighted name. When the cursor changes to an I-beam, click to deposit it. Then edit as you would with a text processor. For multiple-line variable names, hit the carriage return key on your keyboard.

To Move: Click and drag a move handle to the new location. For handles attached to stocks, vertical movement is constrained by the boundaries of the stock. Moving the handle horizontally away from the stock disconnects the stock from the flow. For clouds, all movement is constrained by the cloud's boundaries. To avoid constraints, select and move the attached cloud or stock first; the flow moves, too.

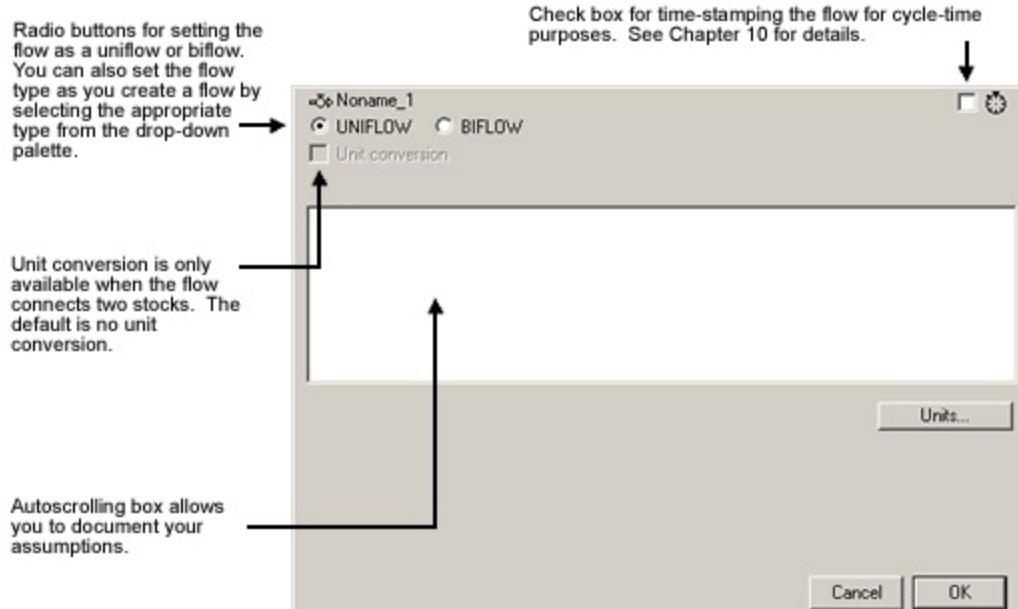
### (c) Choose Polarity



To Choose Polarity: Right-click the circle portion of the icon, choose Polarity, and then choose +/s, -/o, or None.

**Flow Dialog on Map Layer Operations:** When you open a flow on the Map layer, you'll enter the map version of the Flow dialog. Figure 4-15 details the operations within this dialog.

*Figure 4-15  
Flow Dialog - Map Layer*

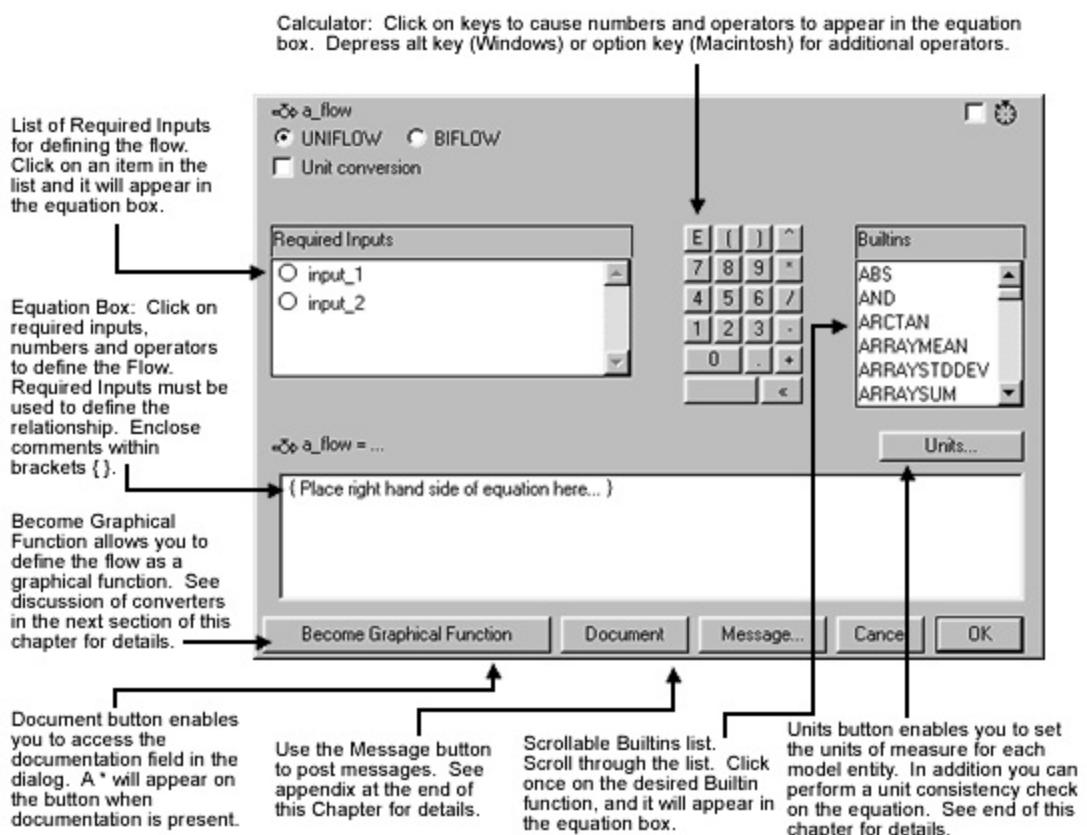


**Flow Dialog on Map Layer Notes:**

- **Uniflow vs. Biflow:** Uniflow means that the flow will flow in one direction only. With uniflows, the flow volume will take on non-negative values only. On the other hand, biflows can take on any value. If you specify a flow as a biflow, a second, shaded arrowhead will appear on the flow to point the direction of negative flow. It is not possible to have a biflow connected to a conveyor, queue, or oven. Note that you can select uniflows or biflows directly from the drop-down list in the building block palette.
- **Unit Conversion:** When the flow is conserved (i.e., it connects two stocks), the unit conversion check box is enabled. Unit conversion enables you to convert the units of measure for the flow, as material moves through the flow pipe. Unit conversion is useful in modeling processes such as assembly processes which transform raw materials into finished goods, or chemical processes which involve molecular transformations. When you specify a flow as unit converted, a shaded half-circle will appear in the flow regulator on the diagram to indicate that unit conversion is taking place.

**Flow Dialog on Model Layer Operations:** On the Model layer, the Flow dialog sports additional controls, as described in Figure 4-16.

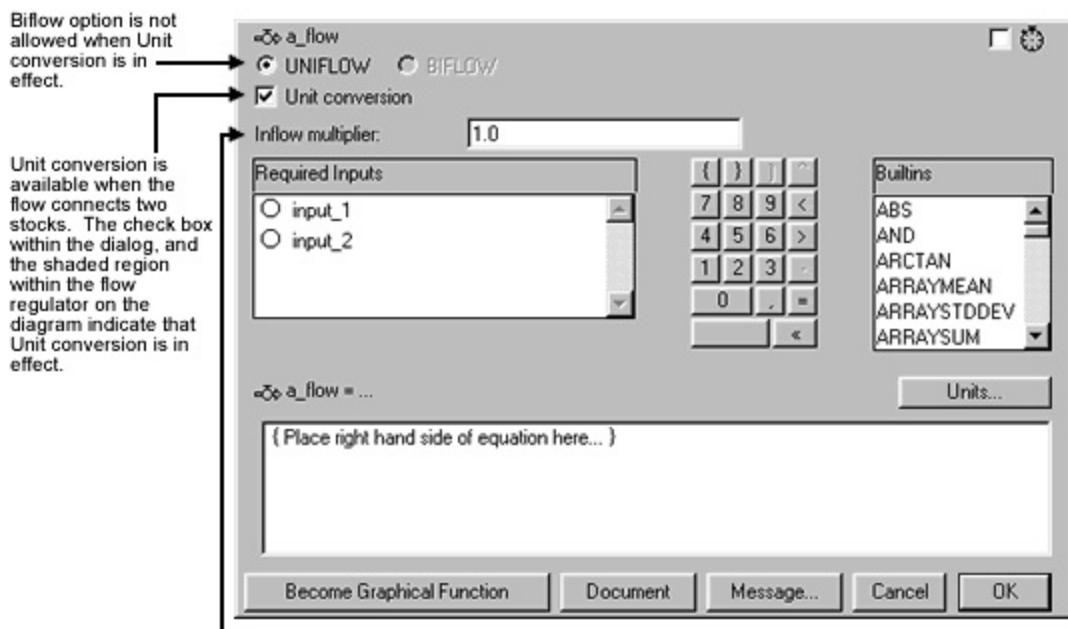
*Figure 4-16  
Flow Dialog - Model Layer*



**Flow Dialog on Model Layer Notes:**

- **Required Inputs:** The Required Inputs list contains a list of all inputs which have been connected to the flow using connectors. Anything in the Required Inputs list must be used in the equation definition for the flow.
- **Unit Conversion:** When Unit conversion has been checked, an additional field will appear in the flow dialog. As shown in Figure 4-17, the inflow multiplier can be a number. Alternately, it can be some model variable which appears in the Required Inputs list. What comes out of the upstream stock will be multiplied by the inflow multiplier, and then added to the downstream stock. When the model runs, the software will report the values for the flow, before unit conversion has taken place.

*Figure 4-17  
Flow Dialog - Unit Conversion*



[Related Topics](#)

# Converters

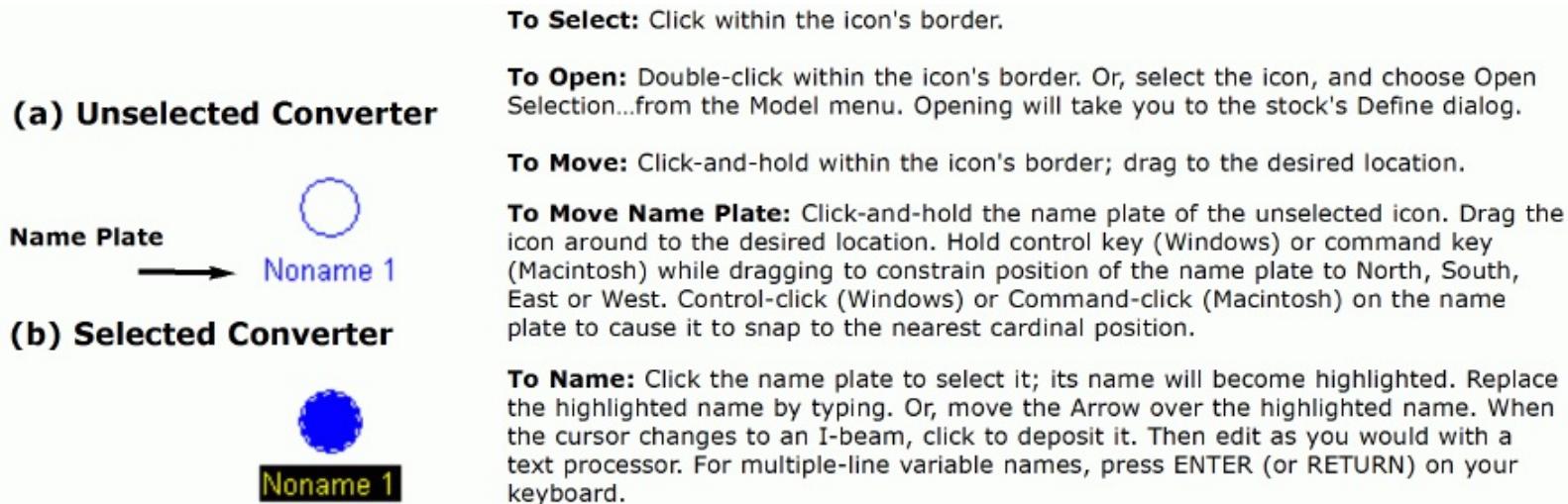
**Purpose:** The converter serves a utilitarian role in the software. It holds values for constants, defines external inputs to the model, calculates algebraic relationships, and serves as the repository for graphical functions. In general, it converts inputs into outputs. Hence, the name "converter."

## Selection and Placement:

- Select the converter icon by clicking once on the circle icon in the Building Block palette. (Note that the drop-down list also enables you to select a special type of converter called the [summer converter](#).)
- Move the mouse to the desired location.
- Click once to deposit the converter.

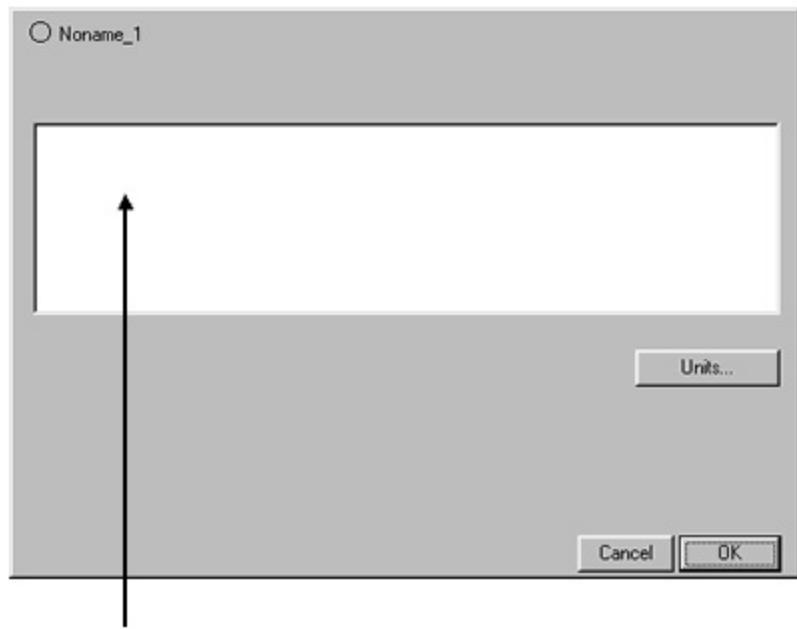
**Surface Operations:** Figure 4-18 details surface operations which may be performed on a converter.

Figure 4-18 Converter Surface Operations



**Converter Dialog on Map Layer Operations:** When you open a converter on the Map layer, you'll enter the Map version of the Converter dialog. Figure 4-19 details the operations within this dialog.

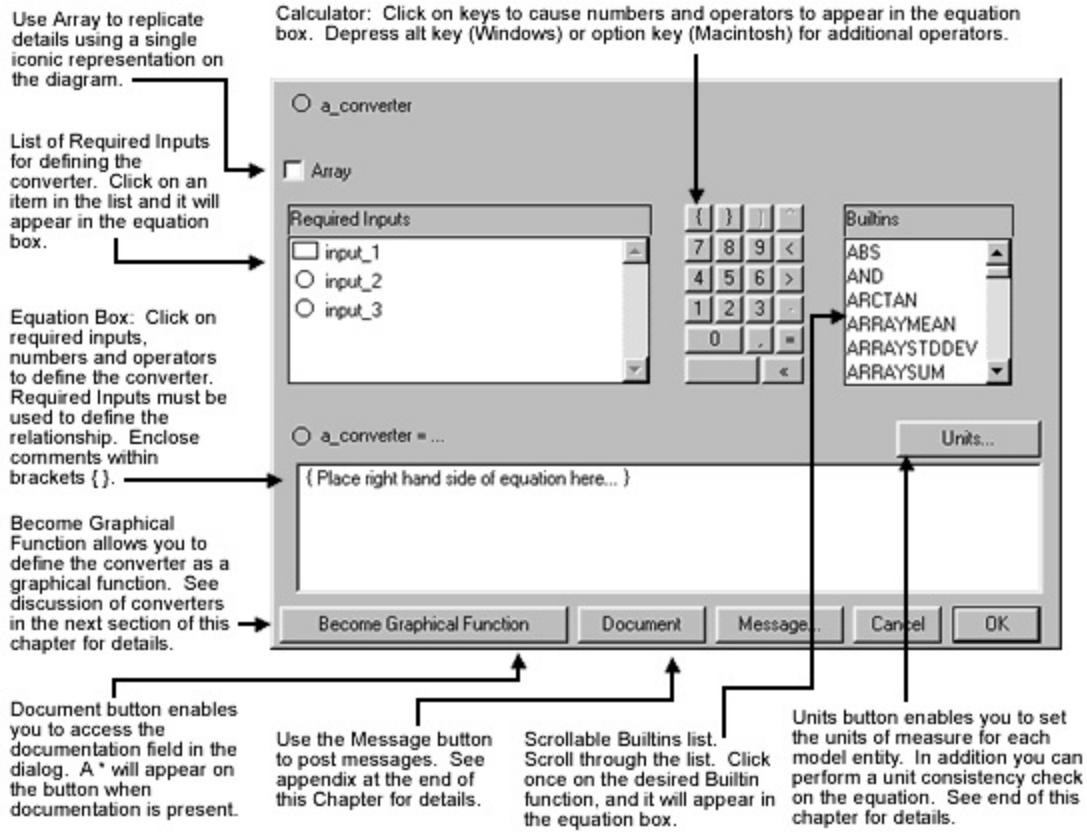
Figure 4-19  
Converter Dialog - Map Layer



Autoscrolling box allows you to document your assumptions.

**Converter dialog on Model Layer Operations:** On the Model layer, the Converter dialog provides a wealth of additional capabilities. Figure 4-20 shows the Converter modeling dialog when inputs have been drawn.

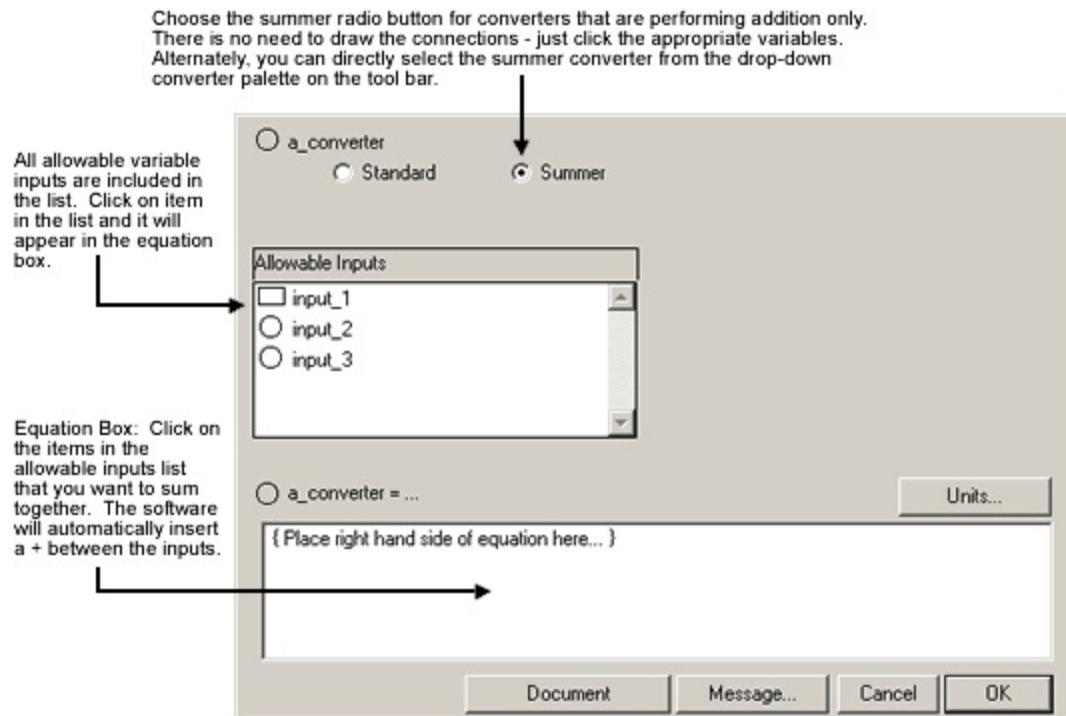
*Figure 4-20  
Converter Dialog - Model Layer - Inputs Drawn to Converter*



# Summer Converter

The Summer Converter dialog is only available when there are no inputs drawn into the converter. The summer converter enables you to add together a set of model variables, *without* the need of drawing inputs to the converter. This special converter type can be exceedingly helpful when you are creating summary report indices and the like. See Figure 4-21.

*Figure 4-21*  
*Summer Converter Dialog - No Inputs*



## Special Note: Graphical Functions

The Become Graphical Function button at the lower left of the dialog enables you to define the converter as a graphical function. Graphical functions also may be defined from within a flow dialog. A graphical function is a sketch of a relationship between some input (which itself can be an algebraic relationship defined from the Required Inputs list and/or Builtins list) and an output.

You can choose to define the graphical function's input by entering an equation in the equation box and then clicking the Become Graphical Function button, or you can use the default equation (Time) by leaving the equation box blank and then clicking the Become Graphical Function button.

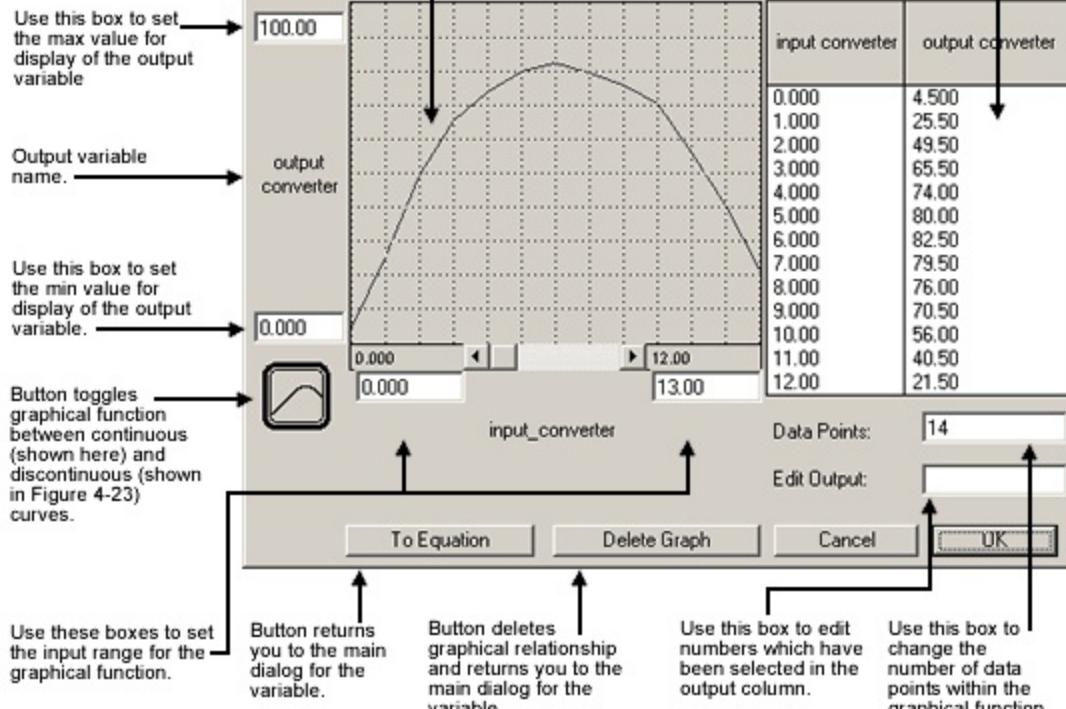
When you click the Become Graphical Function button, the Graphical Function dialog appears. Your tasks within the graphical function dialog are straightforward:

- *Set the min and max values for the X-axis.* Type numbers within the two fields below the graphical function grid. The number in the left field (the min value) must be less than the number in the right field (the max value). Hit the tab key to move from field to field. The minimum and maximum values you choose for the X-axis will be reflected in the non-editable left column (input), to the right of the axes.
- *Set the min and max values for the Y-axis.* Type numbers within the two fields to the left of the graphical function grid. The number in the bottom field (the min value) must be less than the number in the top field (the max value). Hit the tab key to move from field to field.
- *Set the number of data points.* Type a number into the Data Points field, found below the two columns of numbers. You must have at least two data points in a graphical function. You can have up to 1500 data points. When a graphical function contains more than 13 data points, the grid becomes scrollable. To view the entire function, depress the alt key (Windows) or option key (Macintosh).
- *Create a relationship.* Click-and-hold the cursor within the grid. As you drag the cursor, a curve will be drawn, following your mouse movements. In addition, the Y-axis coordinates of the curve will be displayed in the right column (output) at the far right of the dialog.

Figure 4-22  
Graphical Function Dialog

Grid: Click and drag within this region to sketch the graphical function. As you sketch, the vertical coordinates of the curve will appear in the right column.

Right Column: Click on the entity name to select entire column. Select individual numbers in column to alter their values. The curve will change to reflect your editing.



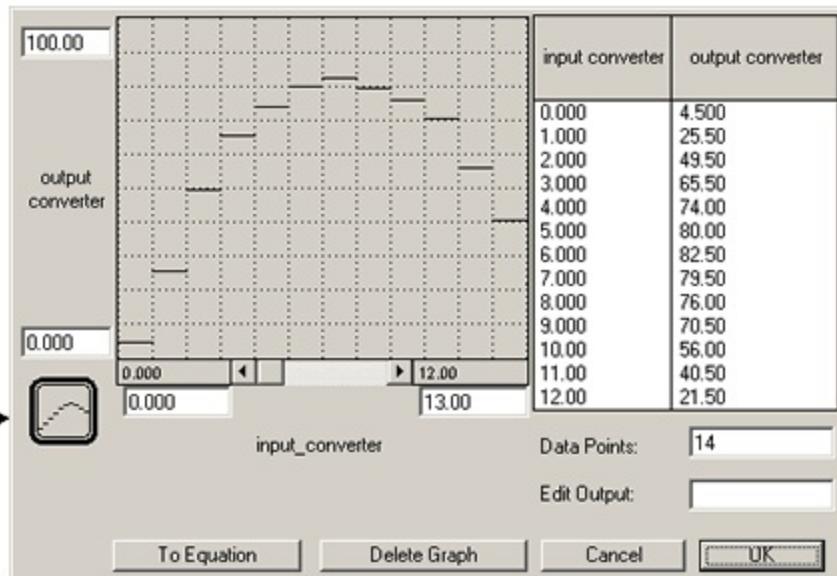
## Notes:

- *Editing numbers directly:* You can edit a graphical function by typing numbers into the right-hand column (the output column). Click on one of the numbers in the output column. The number you select will appear in the Edit Output box. Type the number you want; then hit the tab key on your keyboard. The point you have just defined will appear on the curve, and the next number down the output column will become selected.
- *Copy/Paste operations:* A graphical function's output column can be copied and/or pasted. To copy/paste data into the output column to/from the clipboard, select a number (or a set of numbers) in the output column (to select the entire column, click once on the entity name at the top). Then, under Windows, type control C to copy, or control V to paste. On the Macintosh, type command C to copy, or command V to paste. The number(s) you select will be replaced by the data you paste. The number of data points will be adjusted to reflect the size of the data set being pasted.
- *Data Links from other applications:* It is possible to create data links to your model from an Excel spreadsheet; these links allow you to import data from a spreadsheet and to export data to the same or different spreadsheet. For more information, see [Introduction to Importing and Exporting](#).
- *Continuous vs. Discontinuous Graphical Functions:* The small button near

the minimum boxes enables you to make the graphical function into either continuous or discontinuous line segments. A click on the button will make the line segments discontinuous - a set of flat line segments, as shown in Figure 4-23. Another click will toggle the curve back to the continuous mode. Note that in the discontinuous mode, the last number in the output column is not editable.

*Figure 4-23  
Graphical Function in Discontinuous Mode*

Button indicates that graphical function is operating in discontinuous mode. Click on it to make the function continuous once again.



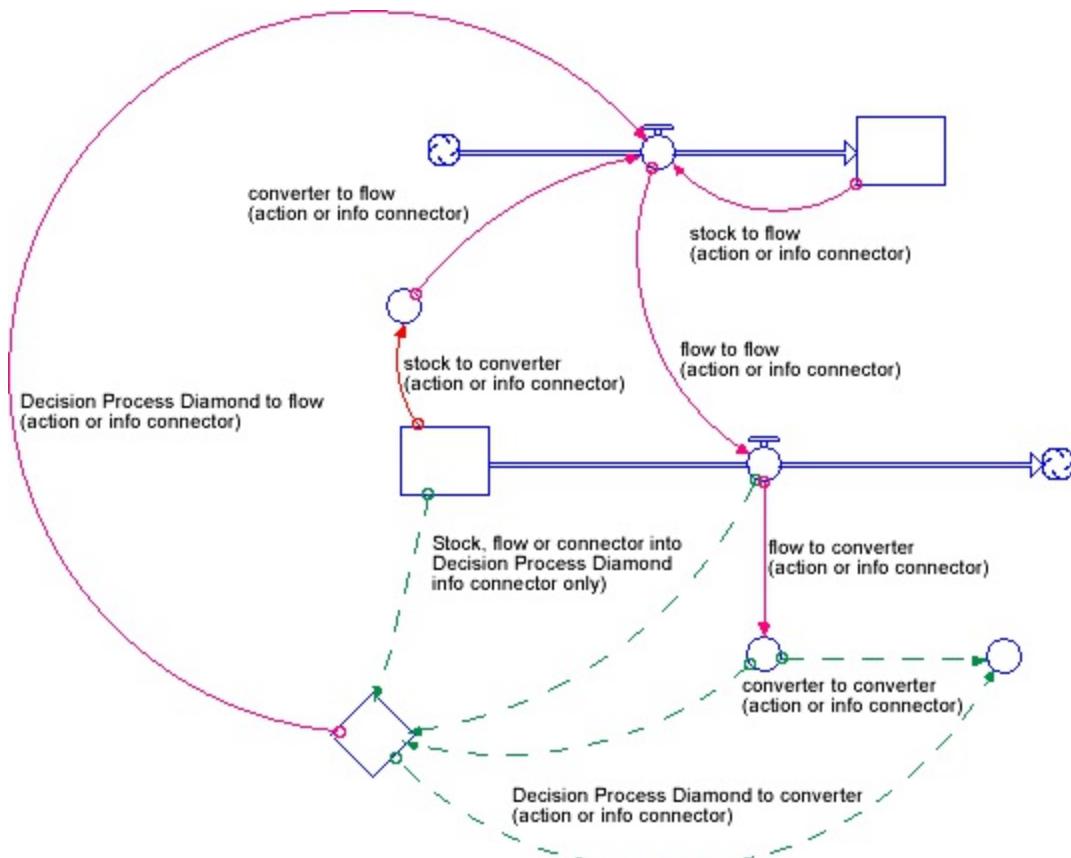
[Related Topics](#)

# Connectors

**Purpose:** As its name suggests, the job of the connector is to connect model elements. The software provides for two distinct types of connector: the action connector and the information connector. Action connectors are signified by a solid, directed wire. Information connectors are signified by a dashed wire. To select an info connector, click and hold the connector icon on the palette. Then select the info connector from the drop-down list that will appear.

Legal connector linkages are illustrated in Figure 4-24. As the diagram suggests, the software will not let you connect either an info or action connector directly into a stock. The only way to change the magnitude of a stock is through a flow.

Figure 4-24 Legal Connector Linkages



## Selection and Placement:

- Select the connector icon by clicking once on the icon in the Building Block palette. To obtain the info connector, select the dashed arrow icon from the drop down list.
- Slide the cursor (the point of the arrow) to the place where the connector will start. This starting place must be within the boundary of a stock, flow

regulator, converter, or decision process diamond on your screen.

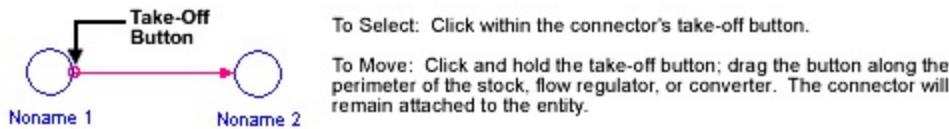
- Click-and-hold. Drag the cursor to the target converter, flow regulator, or decision process diamond. When you make contact, the target will turn gray. Release your click.

**Note on Circular Connections:** In drawing connector linkages, you may encounter an alert which tells you that circular connections are not allowed. Mechanically, this alert means that you have attempted to create a chain of converters and/or flows, such that one converter or flow ultimately depends upon itself. The software can not resolve the resultant simultaneous equations. To resolve the simultaneity, you must include a stock somewhere in the chain. The Introduction to Systems Thinking Guide provides a conceptual discussion of circular connections.

**Surface Operations:** Figure 4-25 details surface operations that may be performed on a connector.

*Figure 4-25  
Connector Surface Operations*

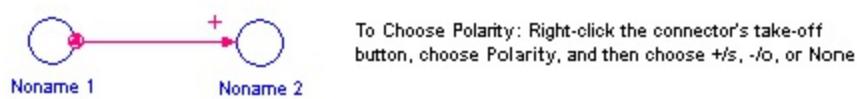
**(a) Unselected Connector**



**(b) Selected Connector**



**(c) Choose polarity**



**Dialog Operations:** Because the connector tool is used to indicate connections, it has no associated define dialog. The interrelationships between model variables are specified in the converter, flow, and stock building blocks.

 [Related Topics](#)

# Modules

**Purpose:** Modules are self-contained models that you can connect to other models. Modules allow you to break a single model into well-defined "chunks". Each module within a model is a cohesive model on its own, which you can run separately or within the larger model.

---

**Note:** The default module icon in the Building Block palette lets you create a self-contained model, as described above. You can also create structure in your models by using Decision Process Diamonds. To create a Decision Process Diamond, click and hold the module icon on the palette, and then select the Decision Process Diamond icon. For information about working with Decision Process Diamonds, see [The Decision Process Diamond](#).

## Selection and Placement:

1. Select the module icon by clicking once on the icon in the Building Block palette.
2. Move the mouse to the desired location on the diagram.
3. Click once to deposit the module.

## Operation:

For detailed information about working with modules, see [Introduction to Modules](#), in "Part 2 - Deeper Details".

 [Related Topics](#)

# The Decision Process Diamond

The Decision Process Diamond (DPD) is a mechanism for managing the diagram complexity associated with the representation of decision processes within your models. With DPDs, you can "bury" the intricacies of the decision rules that drive the flows into a "black box". On the surface, you and the users of your models can clearly see both the inputs and the outputs associated with a decision process. When the need arises, you can "drill down" into the detail of the decision process itself. As a result, your models can maintain a bi-focal perspective, displaying the macro- and micro-structure as needed.

This section describes the mechanical operation of the DPD.

**Creating a Decision Process Diamond:** To create a DPD, select the object from the palette on the Map or Model layer. As you move the cursor onto the diagram, it will change to a diamond-shaped DPD icon. Click once to deposit the icon in the desired location.

**Basic Manipulations: Opening, Closing, Selecting, Moving and Resizing:** When you open a DPD, you obtain a space within which you can place the decision logic that controls a flow. Figure 4-46 identifies the components of a DPD, and illustrates how to open, close, select, move and resize its space.

*Figure 4-46 Opening, Closing, Selecting, Moving and Resizing a DPD*

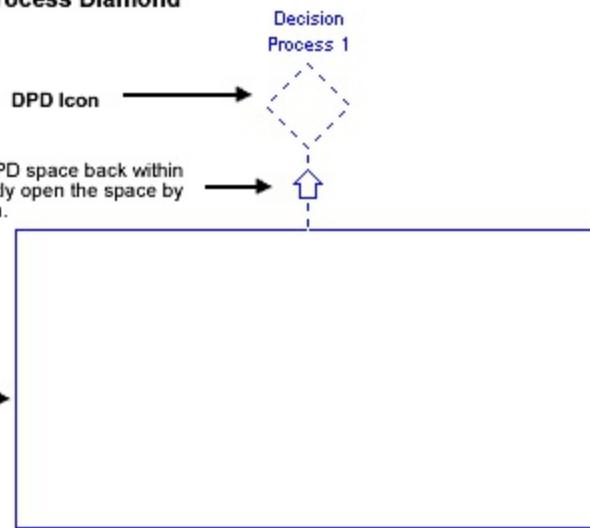
### (a) Unselected Decision Process Diamond

Decision  
Process 1



- To Select: Click within the icon's border.
- To Move the icon: Click-and-hold within the icon's border; drag to desired location.
- To Open the DPD and display the DPD space: Double-click within the icon's border. Or, select it and choose Open Selection... from the Model Menu.

### (b) Unselected Decision Process Diamond



To Select: Click on border. Cursor will change to an arrow when you're in the correct position.

To Move the Space: Click-and-hold border; drag to desired location. DPD space will be constrained by the position of the DPD's icon. The space must be below the icon. Note that the space can overlap other spaces.

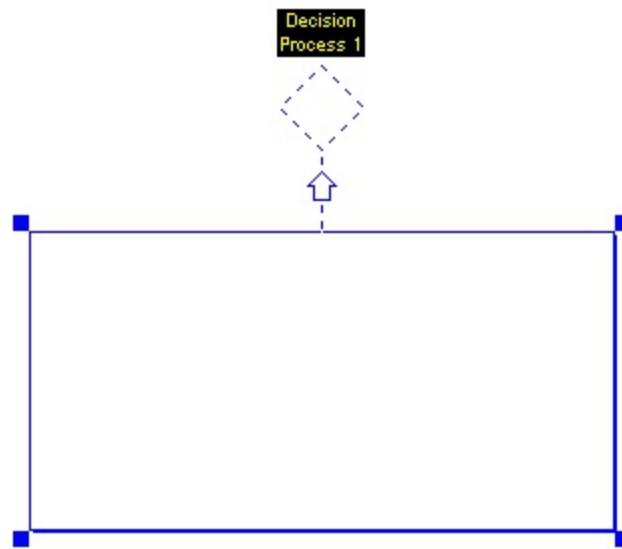
To Move the Icon and Space together: Hold down the Control (Windows) or Command (Macintosh) key and click on the DPD icon; drag to desired location.

### (c) Selected Open Decision Process Diamond

Resizing Handles →  
Drag one of these to resize the space.

**Important Note**  
When one or more DPD spaces are opened, entities on the main layer will become gray. In addition, the cursor will become a Ø when it is outside of the DPD spaces. To regain the hand or other cursor, depress the Control key (Windows) or Command key (Macintosh).

An option within the Model Prefs... dialog allows you to disable the gray display of diagram building blocks.



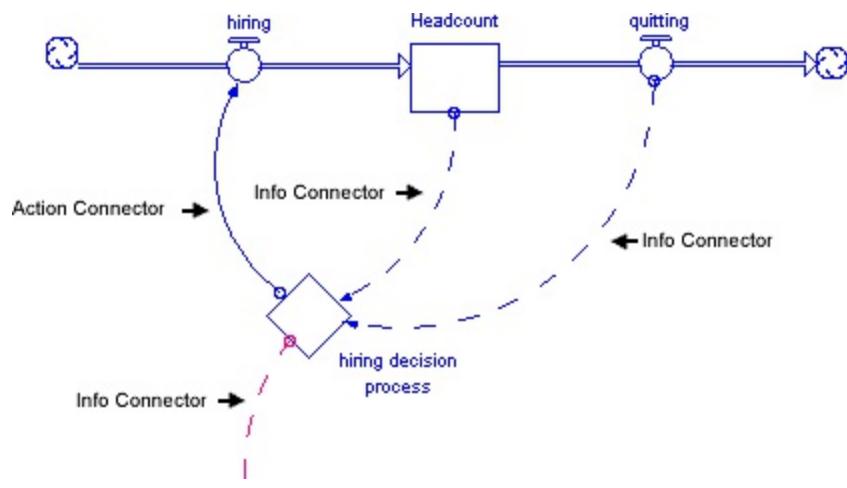
The software requires that Decision Process Diamond spaces be drawn below their icons. As a result, the software will enforce a minimum 40-pixel distance between the bottom of the DPD icon and the top of its associated space. In addition, the software will not allow you to move either the left or right vertical border of the space beyond the horizontal center of the DPD icon.

As you create DPDs, it is important to note that their associated spaces can overlap one another. While the software *will* allow Sub-model spaces to overlap one another, only *one* of the overlapping Sub-model spaces can be open at a time. When you open one overlapping sub-model space, all other overlapping sub-model spaces will be closed. As you move DPD spaces on the diagram, the software will display the outline of other spaces if a conflict is about to occur.

Finally, note that it's possible to import a picture and type comments into the DPD's icon. With the DPD space open, a control-double click (Windows) or command-double click (Macintosh) on the DPD icon will take you to the icon's dialog. There, if you wish, you can document the DPD.

**Incorporating DPDs into your models:** As you construct a model, you will often find that a 2-step process is the most effective vehicle for incorporating DPDs into the model structure. After you have placed a DPD on the diagram, the first step is to use information and action connectors to indicate the key linkages into and out of the decision process. As shown in Figure 4-47, only information connectors are allowed to go into a DPD (after all, the input to decisions is *always* information). Both information and action connectors can emanate from a DPD.

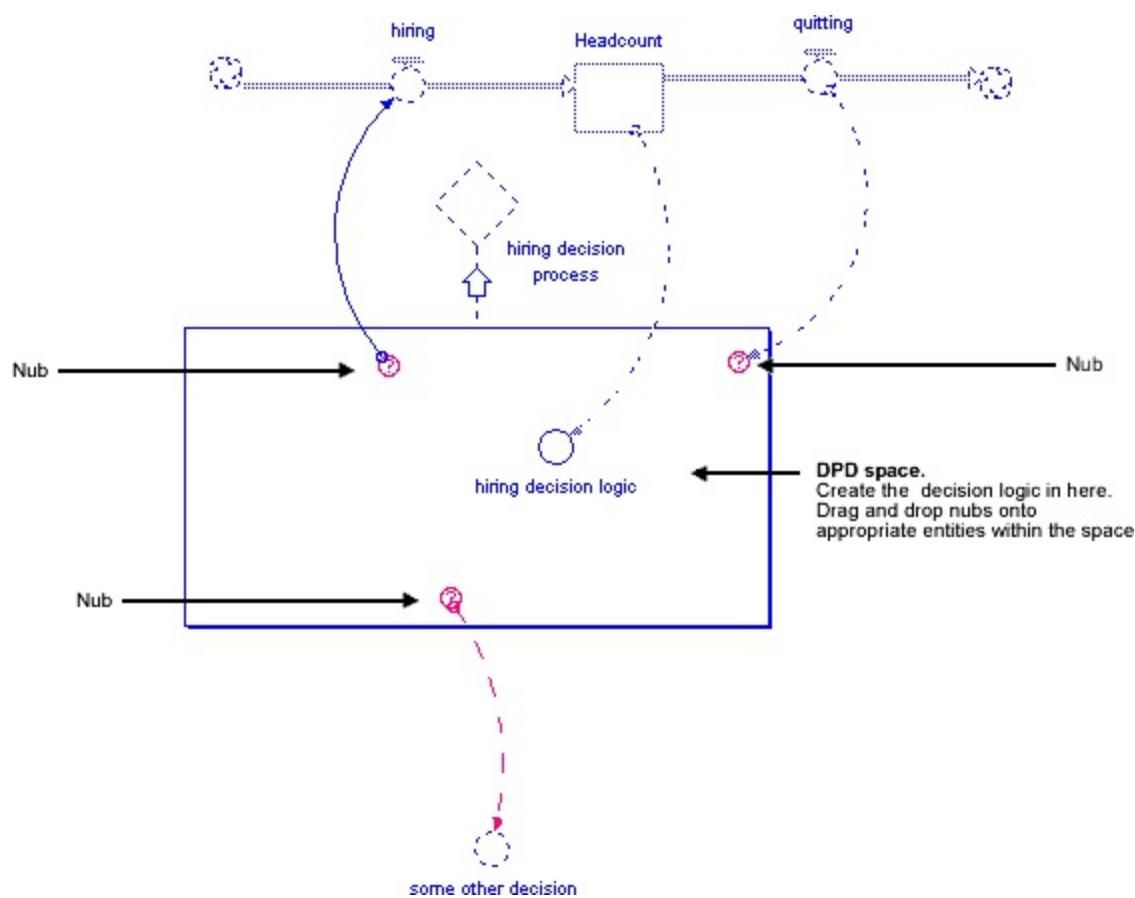
*Figure 4-47  
Main Diagram Layer Mapping With Decision Process Diamonds*



Once you have created the main level map, showing the key information inputs into and outputs from the DPD, you can drill down into the DPD and begin to construct the micro-level decision logic. As shown in Figure 4-48, within the DPD space you will have access to "nubs" that reflect the connections you have made on the main level. To connect the nubs to the relevant decision logic, simply drag and drop them on the appropriate DPD space model entities.

Figure 4-48

## Working With Nubs in the DPD Space



**Working Within a DPD Space:** When a DPD is open, all Map and Model layer building blocks are available to you, as are graphs, tables, Numeric Displays, and Text Boxes. To create a map within a DPD, simply connect stocks, flows, converters and connectors as you would on the main level. Your map can be as detailed as you'd like, although typically your DPD will house the "meatballs and spaghetti" of converter logic, and thus will contain few stocks and flows. As you create structure within a space, you should be aware of the following rules of grammar:

- The stock and flow structure within a DPD space must be self-contained. The software will not allow you draw a flow across the boundary of a DPD space. However, you can draw connections between a DPD space and the diagram, either directly or using Nubs.
- You cannot put a Sector Frame or a button within a DPD. These objects will be gray, as long as you have one or more DPD and/or Sub-models open.
- You cannot turn a stock within a DPD space into a Sub-model. You cannot place a DPD within another DPD space. The software supports only one level of drill-down.
- You cannot create an arrayed stock or flow within a DPD.

**Modifying Main Level When DPD Spaces Are Open:** Whenever one or more DPD spaces are open on the diagram, the software assumes that you want to work within one of them. As a result, all entities on the main diagram level are gray (this can be disabled within the Model Prefs dialog; for more information, see [Model Preferences](#)).

In addition, whenever you move the cursor outside of a DPD space, it will change to an international prohibition symbol ( $\emptyset$ ). When you try to accomplish something with the  $\emptyset$  cursor, all you'll get is a "beep" from the software. To override the  $\emptyset$  and gain the use of building blocks, tools and objects on the main level, depress the control key (Windows) or command key (Macintosh). For example, to place a converter on the main level while a DPD space is open, you can select the converter. Then, control-click (Windows) or command-click (Macintosh) somewhere on the diagram to deposit the converter. To open some main-level entity, control-double-click it (Windows) or command-double-click it (Macintosh).

**Brief Note on Two-Position:** Whenever one or more DPDs exist in the model, main level building blocks and objects can have two positions. The first position is the position that you set for the building block or object, when all Sub-models are closed. The second position is the one that you establish when one or more Sub-models is open on the diagram.

Establishing these two positions is easy. To establish the first position, simply use the Arrow tool to move the building block or object to the desired location while all DPDs are closed. To establish the second position, open the DPD and then control-click (Windows) or command-click (Macintosh) and move the main level building block or object. When all DPDs are closed, main level building blocks and objects will assume their first position. When one or more DPDs are open, the entities on the main level will move to their second position.

Two-position provides a nifty mechanism for keeping your diagram tidy. Occasionally, however, it can get out of control. To kill all second-position information, simply choose Clear Second Position from the model menu. For more detail on two-position, see [Working with Two-Position](#).

# Conveyors, Queues, and Ovens

The topics in this section describe the basic operation of the following stock types:

- [Conveyors](#)
- [Queues](#)
- [Ovens](#)

In this section, you'll learn:

- A summary of each stock type's operation.
- How to configure its modeling mode dialog.
- How to configure its outflow dialogs in the modeling mode.

 [Related Topics](#)

# Conveyors

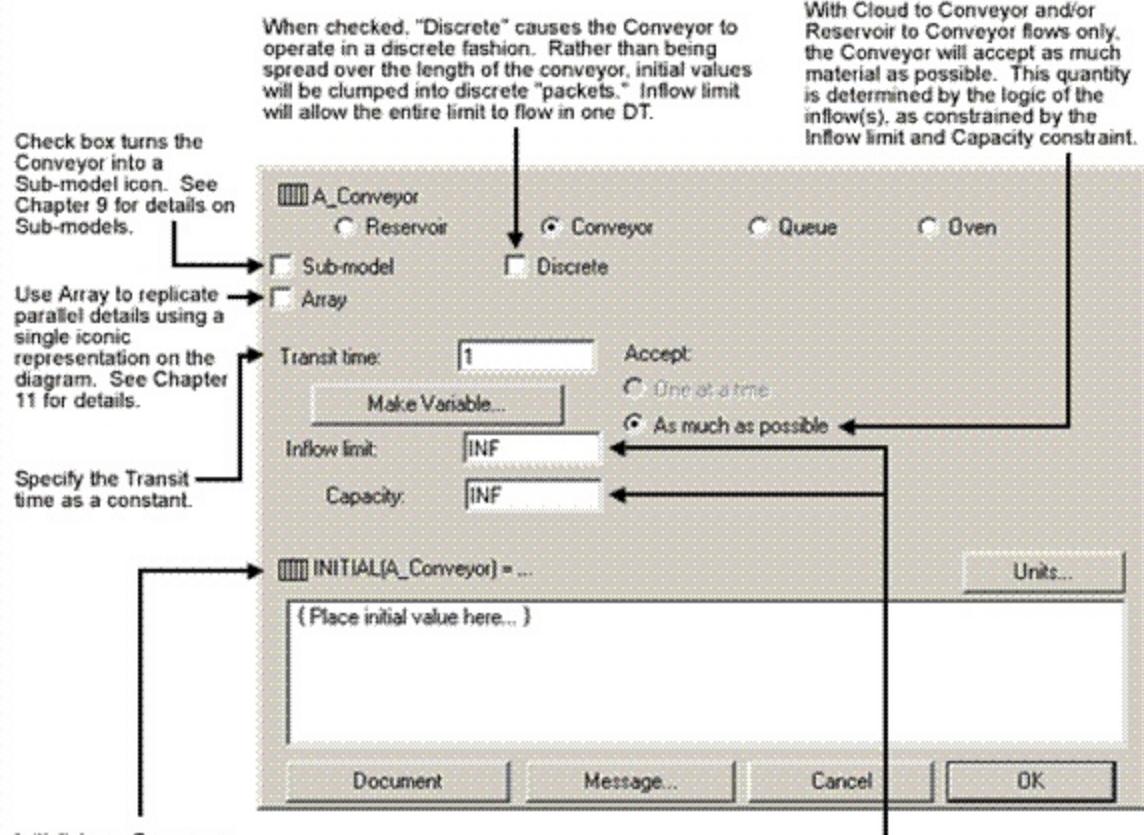
## Operation:

- Think of a Conveyor as a moving sidewalk or a conveyor belt.
- Material gets on the Conveyor, rides for a period of time, and then gets off.
- The transit time for a Conveyor can be either constant or variable.
- Operation of the Conveyor can be arrested.
- Multiple inflows to a Conveyor are allowed - Uniflows only.
- Both Capacity and Inflow Limit can constrain entry to a Conveyor.
- Leakage of Conveyor contents is possible.

**Modeling Dialog Operations:** The Conveyor's Modeling Dialog takes one of three forms, depending on the nature of the inflow(s) to the Conveyor:

- If inflows to the Conveyor come from Reservoirs and/or clouds, the dialog will look like Figure 4-27.
- If one or more inflows to the Conveyor come from Queues, and there are no inflows from Conveyors or Ovens, the dialog will look like Figure 4-28.
- If one or more inflows to the Conveyor come from a Conveyor or an Oven, or if the Conveyor is the first stock in an assigned Sub-model stock/flow structure, the dialog looks like the one shown in Figure 4-29.

*Figure 4-27 Conveyor Dialog - Inflows from Clouds and/or Reservoirs*



#### Initializing a Conveyor:

**Method 1:** Type a single number. It will be evenly spread over the length of the Conveyor.

**Note on Method 1:** When "Discrete" is checked, the Conveyor will be initialized so that material flows out in discrete "packets." The number you type will be spread over the Conveyor, but stacked in the last DT of each Conveyor slat.

**Method 2:** Type a series of numbers, separated by commas, one for each unit of time in the Conveyor's Transit time. Any slots not initialized will default to the last value provided. The first number you type will be the first to exit the Conveyor.

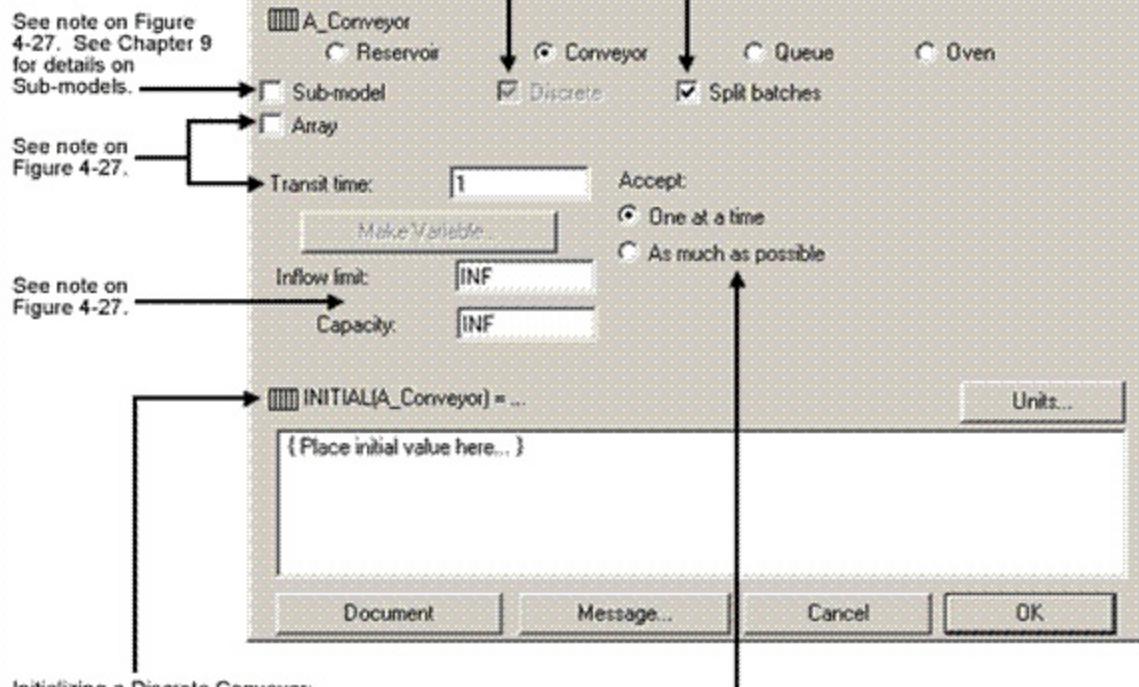
**Note on Method 2:** When "Discrete" is checked, the Conveyor will be initialized so that material flows out in discrete "packets." Otherwise, each initial value you provide will be spread over one time unit within the Conveyor.

Finite values for Inflow limit and Capacity can constrain entry to the Conveyor. Inflows per unit time will not be allowed to exceed the inflow limit. Nor can they be allowed to cause the magnitude of the Conveyor to exceed its capacity. Checking "Discrete" will allow the entire inflow limit to flow in one DT. Type INF to set either equal to infinity. Or, on the Macintosh, type option-5 to get  $\infty$ .

**Figure 4-28**  
**Conveyor Dialog - Inflows from Queues, No Inflows from Conveyors and/or Ovens**

Whenever a flow runs from a Queue to a Conveyor, "Discrete" is checked and grayed. The Conveyor will operate in a discrete fashion. Rather than being spread over the length of the Conveyor, initial values will be put into discrete "packets." If you want the conveyor to act in a continuous mode, use a Reservoir upstream instead of a Queue.

Split batches enables the Conveyor to take a portion of the "next" item in the Queue, whenever Inflow limit or Capacity constraint comes into play. When Split batches is turned "off," the flow from the Queue will be set to zero if it exceeds the Inflow limit or Capacity constraint. This can lead to Queue blockage or "gridlock."



#### Initializing a Discrete Conveyor:

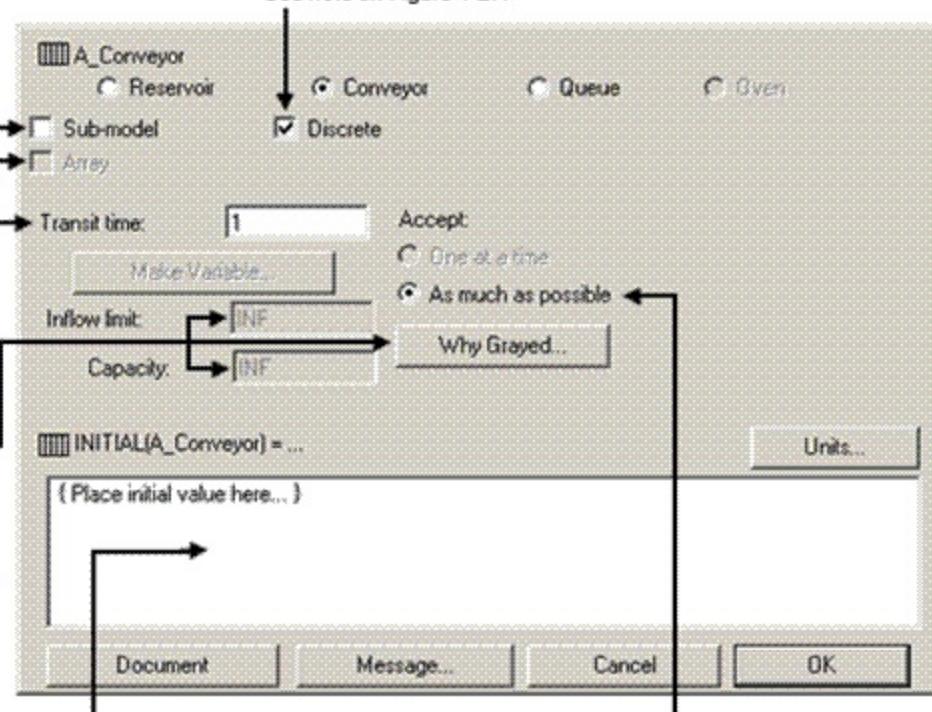
Method 1: Type a single number. It will be spread over the Conveyor, and stacked in the last DT of each slot of the Conveyor.

Method 2: Type a series of numbers, separated by commas, one for each Conveyor slot. Any slots not initialized will default to the last value provided. The first number you type will be the first to exit the Conveyor.

In either case, the conveyor will be initialized so that material flows out in discrete "packets."

"One at a time" will take the "next" item in the Queue, each DT, subject to the Inflow limit and the Capacity constraint. "As much as possible" will cause Conveyor to take as much as it can from the Queue, subject to the Inflow limit and Capacity constraint.

*Figure 4-29  
Conveyor Dialog - Inflows from Conveyor and/or Oven*



See note on Figure 4-27 for details on initializing Conveyors.

With Conveyor and/or Oven to Conveyor flows, the Conveyor will accept as much material as possible. This is because the inflows are determined entirely by the upstream Conveyor(s) and/or Oven(s).

## Special Note on Inflow Prioritization:

Whenever the Inflow limit or the Capacity constraint comes into play, the multiple inflows to the Conveyor will be prioritized. Flows will be allowed to enter the Conveyor in their order of priority. The inflows will show their priority number in their dialogs. For information about the inflow prioritization scheme used by the software, see [Flow Prioritization](#).

**Outflow Dialog Operations:** A maximum of *two* outflows are allowed-both must be Uniflows.

The first flow created is designated as the *flow-thru*. To learn about:

- Basic flow-thru dialog operations, see Figure 4-30.
- Sampling of transit time, see Figure 4-31.
- Arresting the operation of the Conveyor, see Figure 4-32.

The second flow created is designated as the leakage flow. To learn about its dialog see Figure 4-33.

*Figure 4-30  
Conveyor Flow-thru Dialog - Basic Operations*

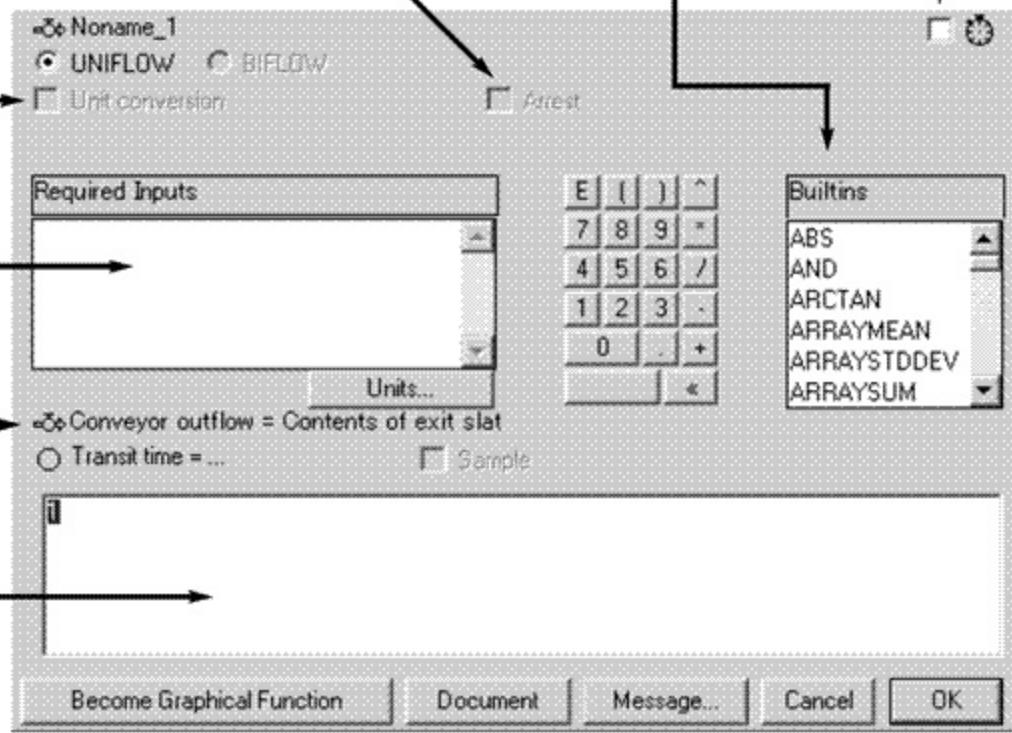
Arrest option is disabled whenever:

- The Required Inputs list is empty.
- An inflow to the Conveyor comes from a Conveyor.
- An inflow to the Conveyor comes from an Oven.

Builtins can be used to define the Transit time.

Check box for time-stamping the flow for cycle-time purposes.

Unit Conversion is available when the flow connects the Conveyor to another stock. See



Required Inputs can be used to define the Transit time or to control the Arrest or Sample logic.

Material will exit the Conveyor when its Transit time has expired.

#### Defining the Transit time:

**Constant Transit time:** Type a single number. All material, which enters the Conveyor, will use the number you specify for the Transit time. Note that a constant Transit time may also be specified from within the Conveyor dialog.

**Variable Transit time:** Create a relationship using Required Inputs, Builtins, algebraic operators, or a Graphical Function. If "Sample" is not checked, units will be assigned to the Transit time that is calculated as they enter the Conveyor.

**Important Note:** When using the software's Builtin Statistical functions to generate a variable Transit time, be sure to generate the random numbers within the Conveyor outflow dialog. DO NOT generate a sequence of random numbers in an external converter, and then use the converter to define the Transit time. The sequence of Transit times that will be used will not conform to the distribution you have specified in the converter.

**Figure 4-31**  
*Conveyor Flow-thru Dialog - Sampling Transit Time*

A variable named "sampling signal" has been created on the Diagram. Its logic was set using IF-THEN-ELSE logic to generate either a 0 or a 1. You can also use the SWITCH Builtin to generate a 0-1 sampling signal.

Material will exit the Conveyor when its sampled Transit time has expired.

Here, an exponential distribution is used to define the Transit time. The "next" number in the random number sequence will be sampled, and used for the transit time whenever the "sampling signal" converter generates a 1.

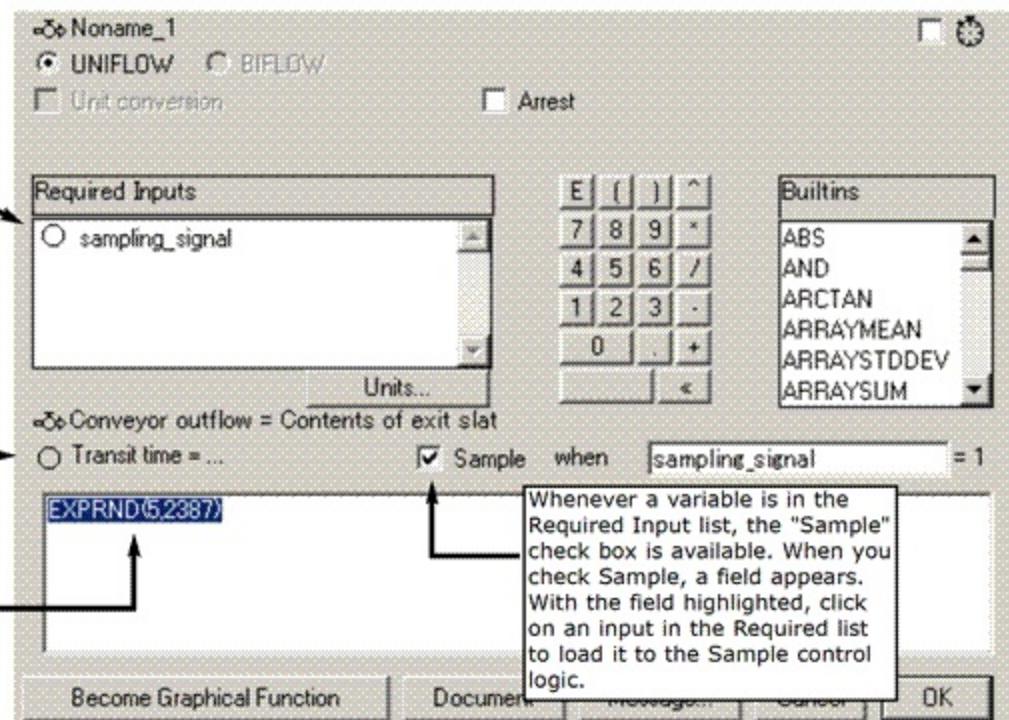


Figure 4-32  
Conveyor Flow-thru Dialog - Arrest Mode

Figure 4-33  
Conveyor Leakage Dialog

No-leak zone is that portion of the Conveyor's transit time, which is not available for leakage. The No-leak zone length is expressed as a percentage - the percentage of the transit time, which is applied to the inflow as it enters the Conveyor. Leakage always occurs at the downstream end of the Conveyor.

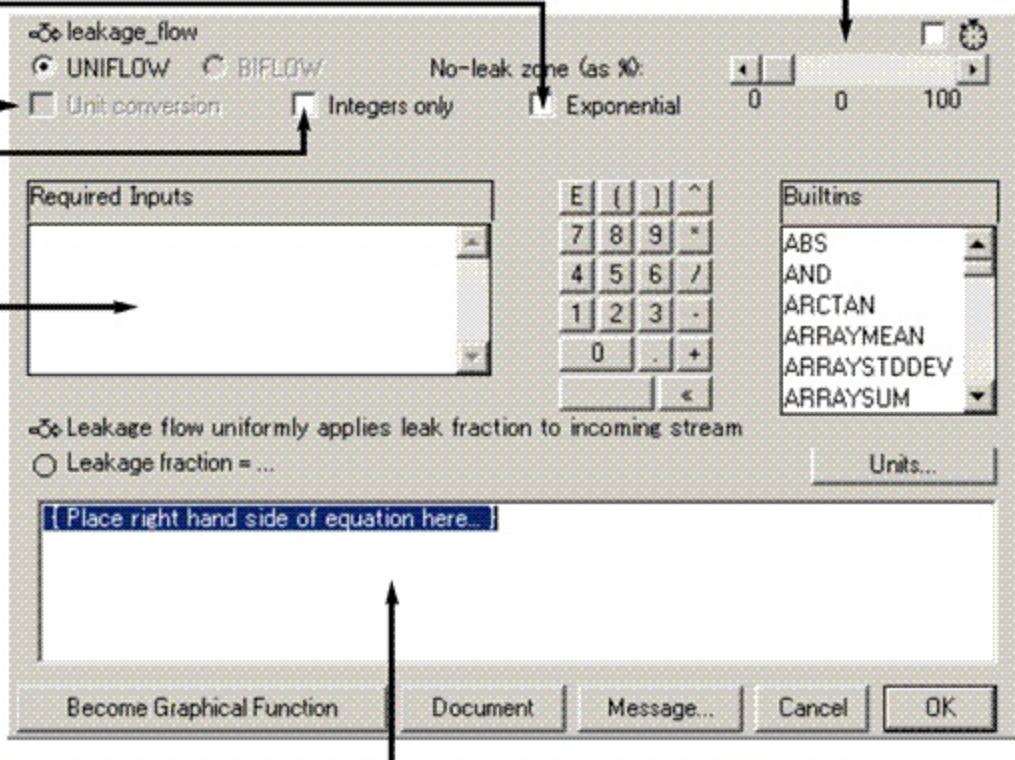
Move the slider to set the length of the No-leak zone. Click within the shaded area to move in increments of five percent. Click on the left or right arrows to move the slider in increments of one percent.

Check Exponential to create an exponentially distributed leakage in place of the default uniformly distributed leakage.

Unit Conversion is available when the flow connects the Conveyor to another stock.

Rounds leakage flow to nearest integer value - each DT if Discrete is checked in Conveyor dialog; each time unit otherwise.

Required Inputs can be used to define the leakage fraction.



Define a Leakage fraction using a constant, Required Inputs, Builtins, algebraic operators, or a Graphical function. Leakage fraction must be in the 0-1 range. Numbers outside this range will be forced to the nearest endpoint. The fraction you specify will be used to determine the leakage flow volume. Whatever enters the Conveyor, multiplied by the leakage fraction, will exit through the leakage flow. Leakage will flow in a uniform fashion once the material moves beyond the no-leak zone.

[Related Topics](#)

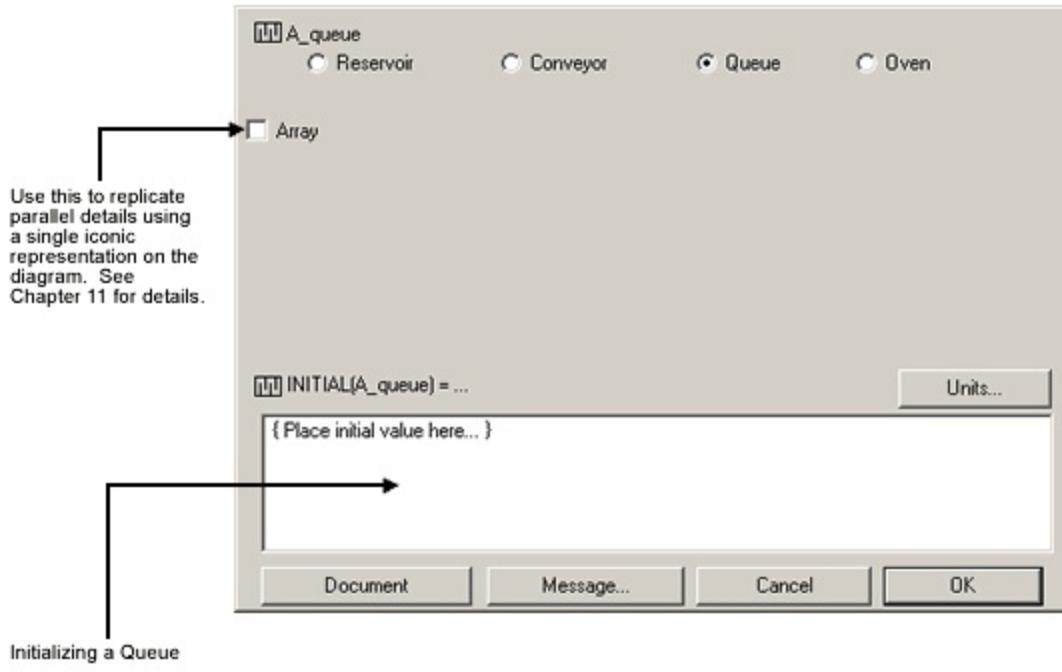
# Queues

## Operation:

- Think of a Queue as a line of items awaiting entry into some process or activity (e.g., grocery store checkout line, airport ticket counter line).
- Queues are FIFO (first in - first out) in their operation. Stuff enters the queue, and remains in line, waiting its turn to exit the Queue.
- Multiple inflows are allowed (Uniflows only). The contents of each inflow are given their own slot or element in the Queue.
- The Queue exerts no behind the scenes control over its inflows. Any control from a Queue to its inflows must be explicitly represented in the inflow logic.
- The software prioritizes multiple inflows into a Queue. For more information, see [Flow Prioritization](#).

**Modeling Dialog Operations:** The Queue's Modeling Dialog is easy to define, as shown in Figure 4-34.

Figure 4-34 Queue Dialog



Initializing a Queue

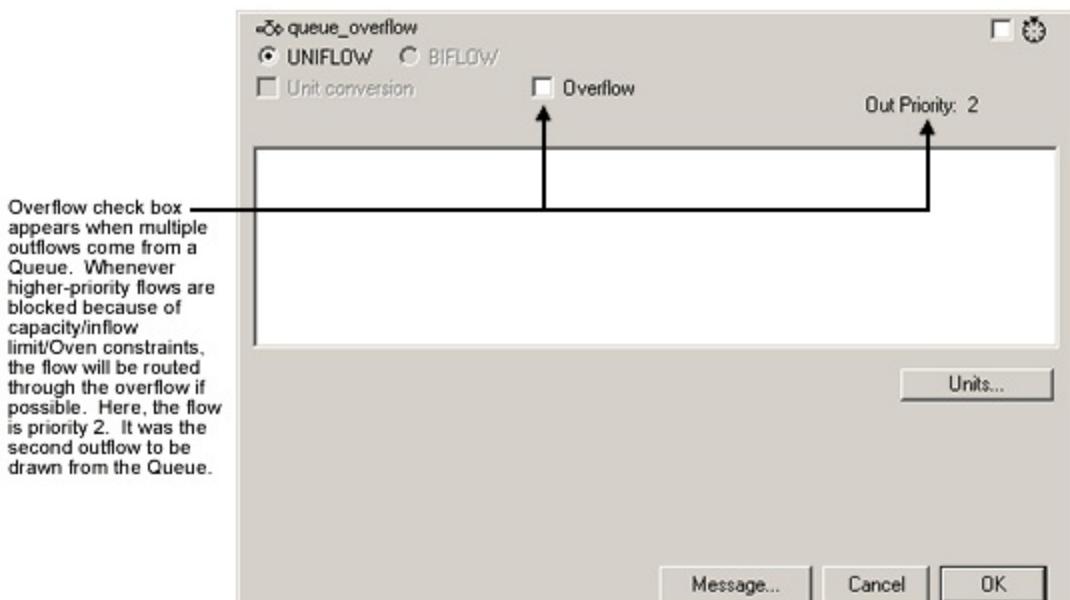
Method 1: Type a single zero to represent a Queue which is initially empty.

Method 2: Type a series of positive numbers, separated by commas, one for each Queue slot. The first number you type will be the first to exit the Queue.

**Outflow Dialog Operations:** Multiple Queue outflows are allowed-all must be Uniflows.

- The Queue "wants" to flow everything through its outflows. As such, its flows are constrained only when there is an Inflow limited and/or Capacity constrained Conveyor, or an Oven, immediately downstream.
- Unit conversion is possible in flows that connect Queues to other stocks. Unit conversion works as illustrated in Figure 4-17 (in [Flows](#)).
- When multiple outflows are drawn from a Queue, it is possible to designate one or more of the outflows as "overflow" in their dialogs. When a higher-priority flow is blocked because of capacity constraints, the flow will be routed through the overflow. A sample overflow designation is shown in Figure 4-35 (below).
- [Flow Prioritization](#) discusses the prioritization scheme employed by the software when there are multiple outflows from a Queue.

*Figure 4-35  
Queue Outflow Overflow Dialog*



[Related Topics](#)

# Ovens

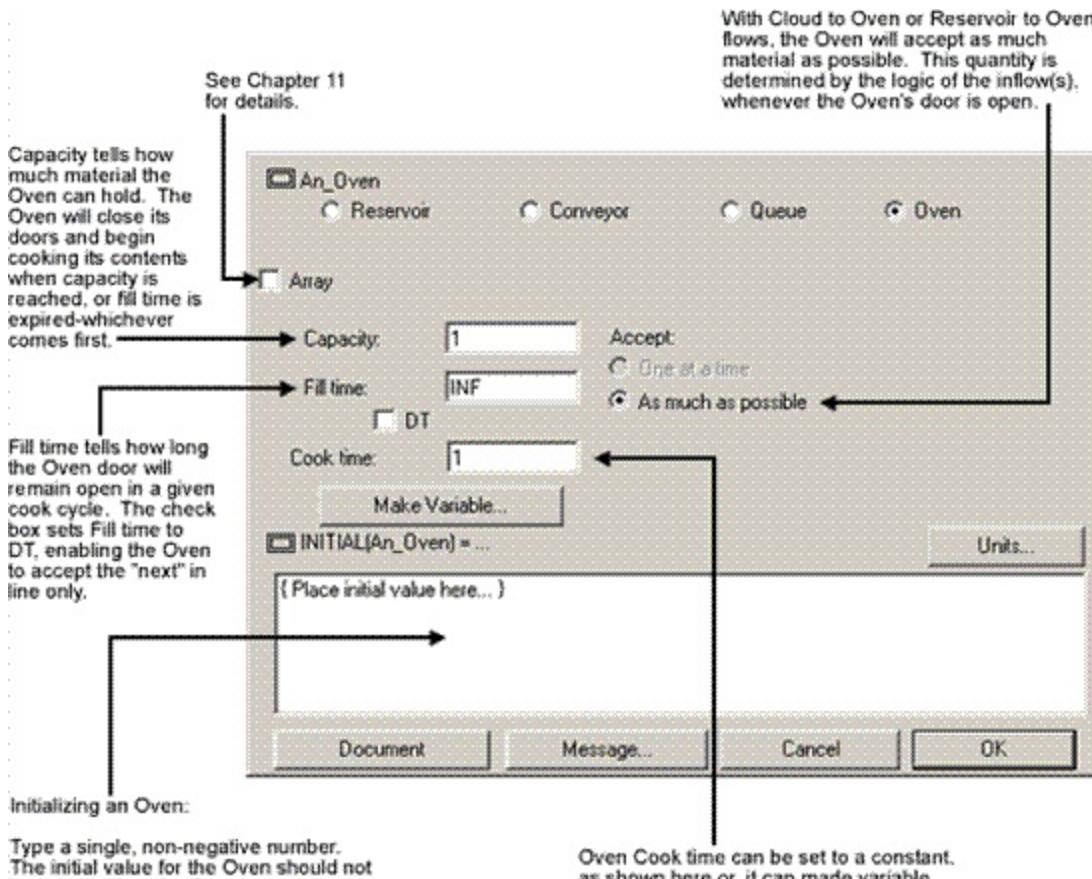
## Operation:

- Think of an Oven as a processor of discrete batches of stuff.
- The Oven opens its doors; fills (either to capacity or until it is time to close the door); bakes its contents for a time (as defined by its outflow logic); then unloads them in an instant.
- Operation of the Oven can be arrested.
- Only a single Uniflow can flow into an Oven.
- The inflow to an Oven can come from a cloud, a Reservoir or a Queue.

**Modeling Dialog Operations:** The Oven's Modeling Dialog takes one of two forms, depending on the nature of the inflow to the Oven.

If the inflow comes from a Reservoir or cloud, the dialog will look like Figure 4-36.

*Figure 4-36 Oven Dialog - Inflow from Reservoir or Cloud*

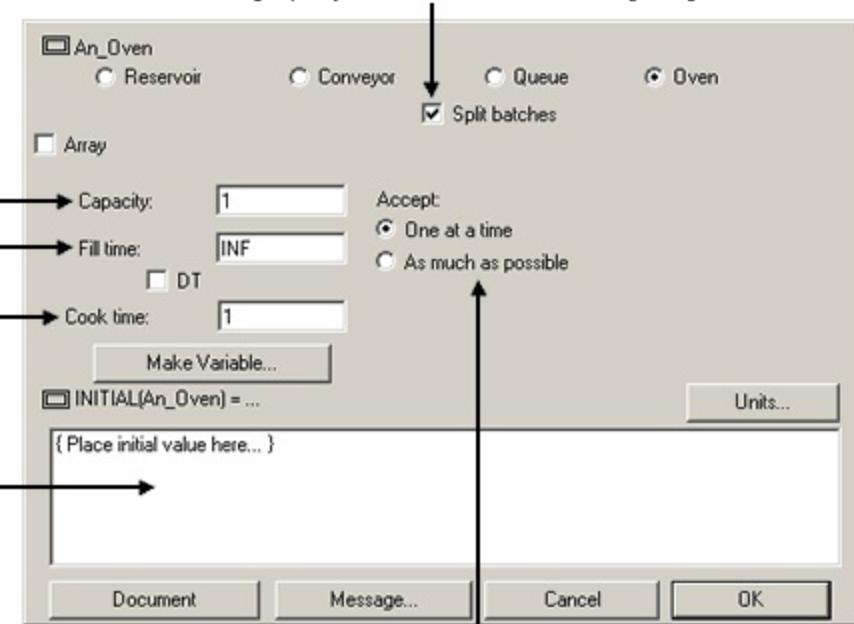


If the inflow comes from a Queue, the dialog will look like Figure 4-37.

*Figure 4-37*

## Oven Dialog - Inflows from Queue

Split batches enables the Oven to take a portion of the "next" item in the Queue, in order to fill itself to capacity. When Split batches is turned "off," the flow from the Queue will be set to zero if it is too large to fit in the remaining capacity. This can lead to Queue blockage or "gridlock."



"One at a time" will take the "next" item in the Queue, each DT, subject to Capacity constraints. "As much as possible" will cause the Oven to take as much as it can from the Queue, subject to its Capacity constraint.

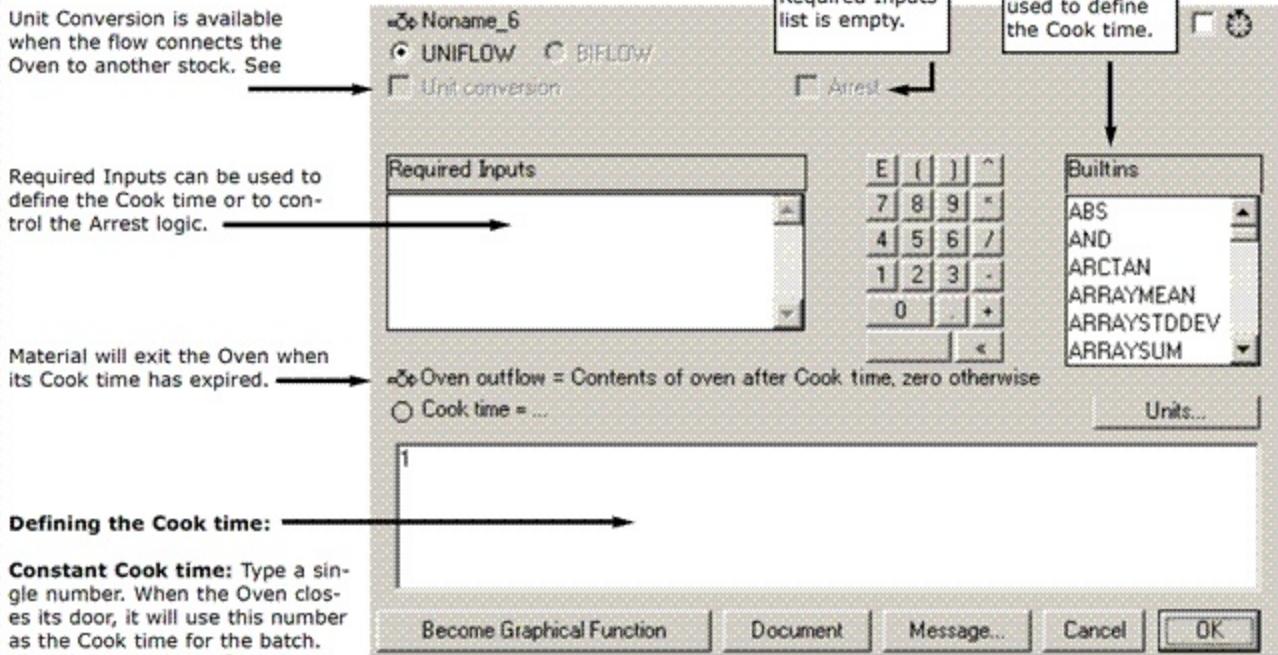
### Notes:

- Use long fill times, "One at a time," and small capacities to represent capacity constrained situations.
- Check the "DT" check box, select "One at a time," and use a large Capacity to represent a processor that takes only the "next" item from the upstream Queue.
- Outflows from Queues can be designated as "overflow." When gridlock has occurred in one Oven, the overflow can direct the contents of the Queue to a larger-capacity Oven.

**Outflow Dialog Operations:** A single, Uniflow outflow is allowed. Cook time is an attribute of the Oven outflow. Cook time is sampled each time the Oven closes its door. When Cook time has expired, the Oven will spit out its entire contents in an instant. At all other times the Oven's outflow will be zero.

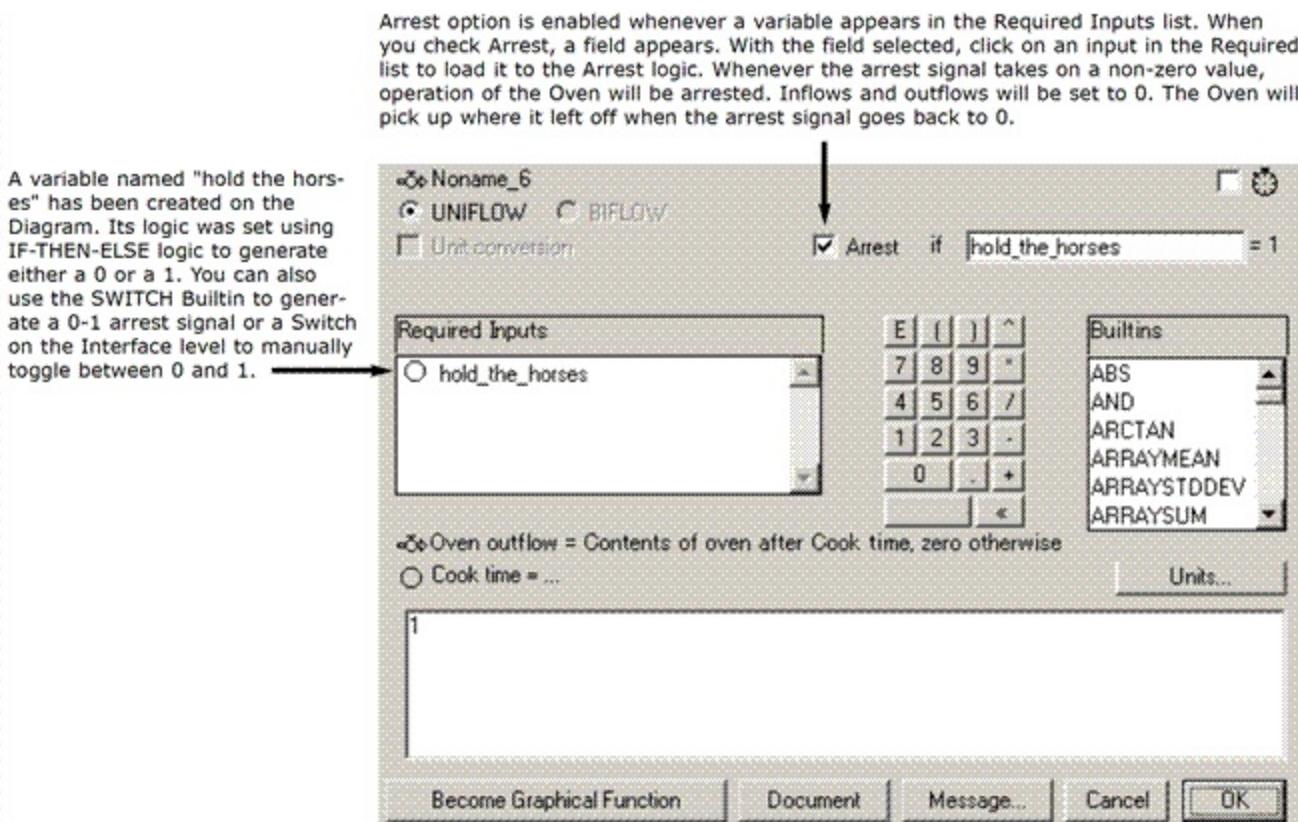
Basic operations in the Oven outflow dialog are shown in Figure 4-38. Figure 4-39 shows the Oven outflow dialog, when Arrest is in effect.

*Figure 4-38  
Oven Outflow Dialog - Basic Operations*



**Important Note:** When using the software's Built-in Statistical functions to generate a variable Cook time, be sure to generate the random numbers within the Oven outflow dialog. DO NOT generate a sequence of random numbers in an external converter, and then use this converter to define the Cook time. The sequence of Cook times that will be used will not conform to the distribution that you have specified in the converter.

**Figure 4-39**  
**Oven Outflow Dialog - Arrest Mode**



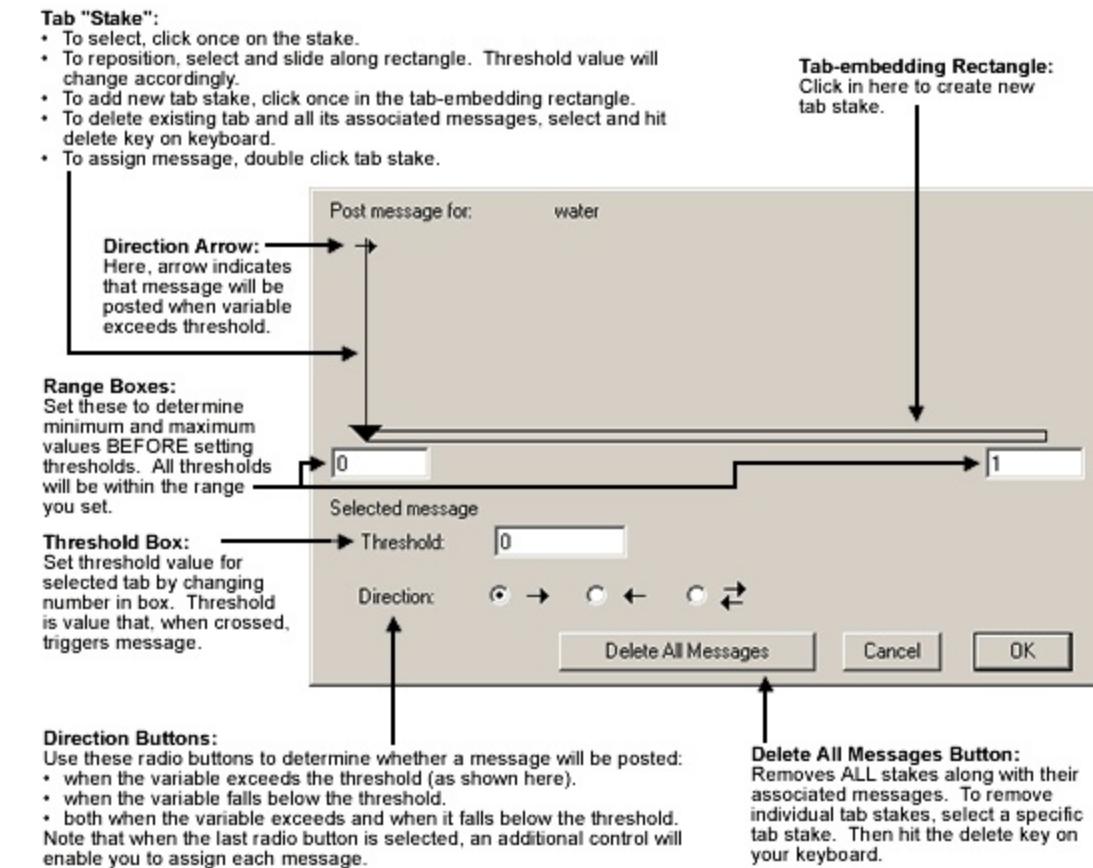
 [Related Topics](#)

# The Message Poster

**Purpose:** The software's message posting capability serves two purposes. First, message posting gives you, as model author, power to provide consumers of your models with visceral, concrete manifestations of model output. Second, message posting gives you a mechanism for coaching model consumers toward a deeper understanding of model structure and behavior. Message posting achieves these purposes by displaying author-defined messages (text, pictures, movies, and sound). You can even use message posting to navigate the user to a sequence of coaching screens!

**Basic Operations:** Message posting is activated from within the modeling mode dialog of stocks, flows and converters. A click on the "Message..." button within the dialog will take you to the message threshold dialog, as shown in Figure 4-40.

Figure 4-40 Message Threshold Dialog



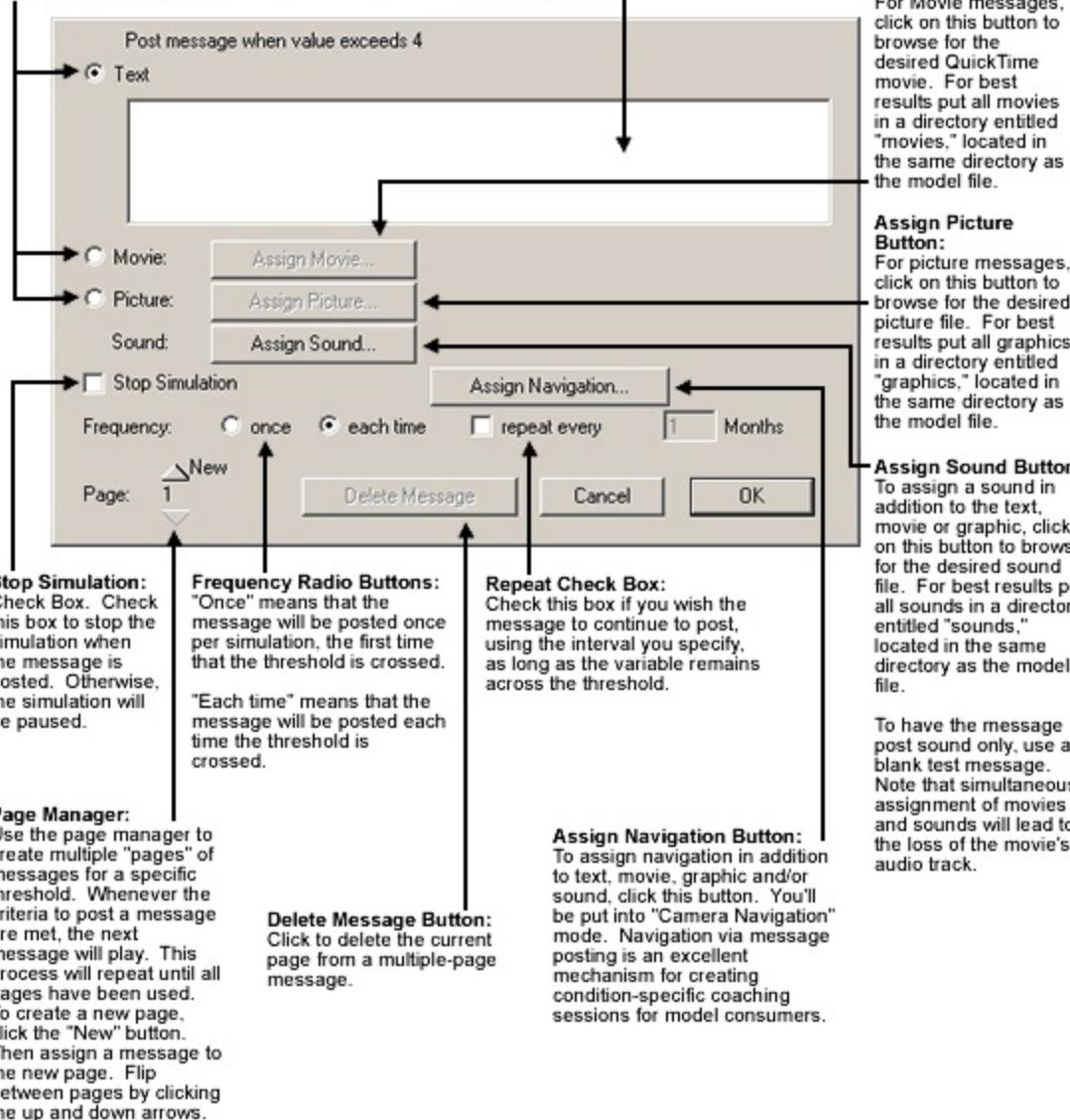
Your job in the message threshold dialog is to establish thresholds for message posting. A threshold is a value for a model variable which, when crossed, will cause a message to be posted. To establish thresholds for message posting, follow the procedure outlined below.

1. Establish the range. Type numbers in the minimum and maximum boxes, as indicated in Figure 4-40. Thresholds will fall in the range between the minimum and maximum values that you set.
2. Set a threshold. Select the default tab, or create a new tab by clicking somewhere in the tab-embedding rectangle. Then, move the tab to a position that reflects the desired threshold. Alternately you can set the threshold value directly by typing in the Threshold box.
3. Set the message direction. Use the radio buttons to determine whether a message will be posted (a) when the magnitude of the variable grows larger than the threshold; (b) when it falls below the threshold; or (c) both when it exceeds and when it falls below the threshold. Note that the variable must cross the threshold for a message to be triggered!
4. Assign the message. Double click the tab to enter the message assignment sub-dialog. As shown in Figure 4-41, a plethora of features await you there.

*Figure 4-41  
Message Assignment Dialog*

**Message Type Radio Buttons:**  
Message can be textual, a QuickTime movie clip, or a picture. Select the message type that best suits your needs.

**Text Box:**  
Textual messages can contain up to 256 characters. For longer textual messages, import a picture.



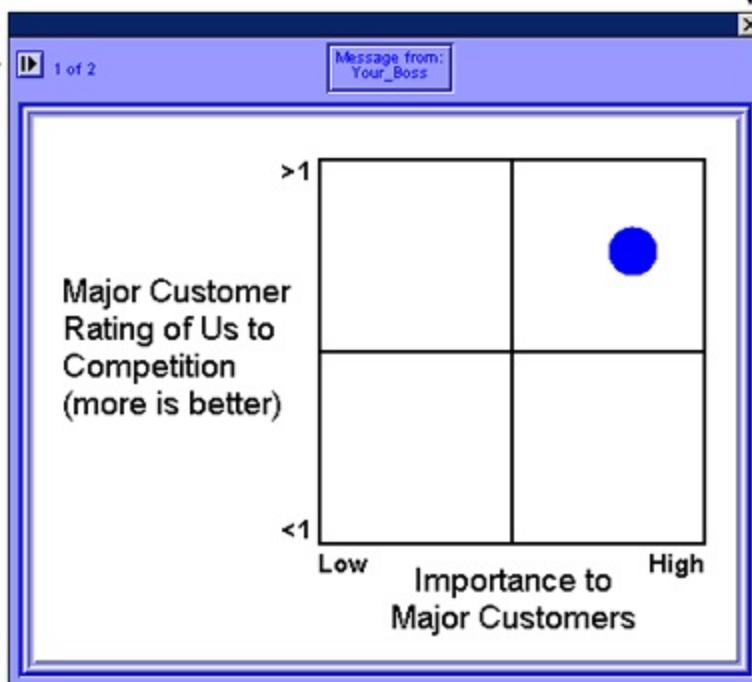
**Note on Message assignment:** When a tab contains multiple message pages, the pages will be accessed in order, until they have all been accessed. After all pages have been accessed, the tab will stop generating messages. To reset the tab's internal counter, simply close and re-open the model.

During the simulation run, any variables that have been outfitted with message posting capability will "post" their assigned messages, play their assigned sounds, or navigate to their assigned location upon satisfaction of their specific posting condition(s). All visible messages will be posted in the same place-in a message posting window that floats above the model surface. (Note: If you assign Sound only, the model will pause and play the sound, but no visible message box will appear.) Figure 4-42 details the salient features of the message poster window.

**Figure 4-42**  
*Floating Message Posting Window*

**Close Box:** \_\_\_\_\_  
Click to put away message.

**Advance Button:**  
Appears when multiple  
messages are  
triggered at the same  
time. Click to advance  
to the next message.



# Working with Units

Unit analysis is a simple way for you to stay in control of the process of building and simulating dynamic models. In order to assign a unit of measure to a stock, flow or converter in your model, you must think carefully about what the model element is, how it works, and how it relates to other model elements. Whenever you check the consistency of units – making sure that left- and right-hand sides of equations resolve to the same units of measure, or ensuring that stocks in a conserved flow chain all have the same units of measure – your thinking about how the process works is necessarily taken to a deeper level. Unit Analysis is a great vehicle for minimizing the number of silly mistakes in your models. More important, it is an excellent vehicle for maximizing the conceptual clarity of your models.

Unit Analysis is primarily a mental discipline. If you don't practice it in the construction of your models, no one will make you do it. That said, the software's unit analysis capabilities greatly facilitate the process of conducting unit analysis in your models.

There are three basic operations associated with unit analysis:

- [Assigning units](#) of measure to the model entities.
- [Creating units](#) (and creating equations for units)
- [Checking the units](#) to ensure that the left- and right-hand sides of equations have consistent units, and that conserved flow chains have consistent units.

The software provides you with many pre-defined units that you can assign to your model's entities. In addition, you can [create your own units](#).

Note that you can directly assign units for stocks and converters. Units for flows are assigned by the software based on the units you have assigned to their associated stocks. For example, if you assigned a unit of measure called "widgets" to a stock, and your model is running in months, any flows into or out of the stock would automatically be assigned the units "widgets per month".

The following sections discuss [assigning units](#), [creating units](#), and [checking unit consistency](#).

# Assigning units to entities

Begin by assigning units to the stocks in your model. Units for flows are automatically assigned by the software based on the units you have assigned to their associated stocks. For example, if you assigned a unit of measure called "widgets" to a stock, and your model is running in months, any flows into or out of the stock are automatically assigned the units "widgets per month." All stocks in a main chain are automatically assigned the same units that you assigned to any stock in the chain.

---

**Notes:** If you add stocks or flows to the chain after you assign units, the new entities will not be automatically assigned their units. To assign units to the new entities, assign units to any stock in the chain; all the other entities in the chain will be automatically assigned their units.

If there is a unit-converted flow (one for which the **Unit conversion** check box is selected) in the chain, any flows or stocks beyond that flow will **not** be automatically assigned the same unit of measure; you must manually assign units to those flows and stocks yourself.

---

After you have assigned units to the stocks, assign units to the converters in your model.

You can assign units to entities by right-clicking entities in the model diagram. You can also assign units while you are creating or editing entities. Both procedures are described below.

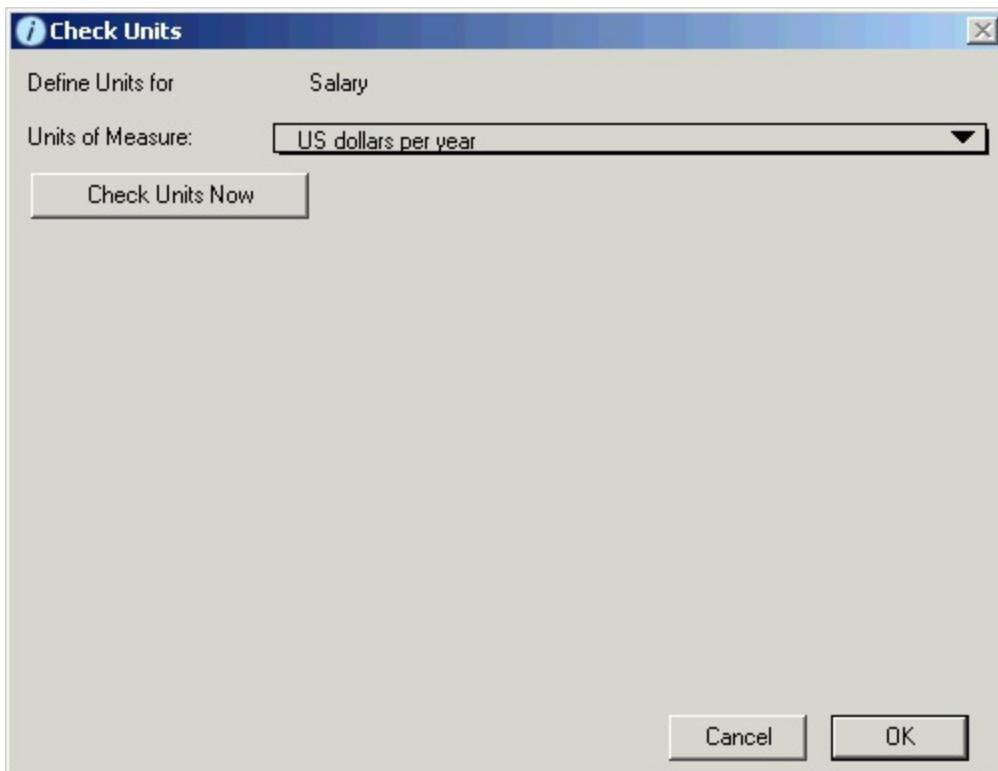
## To assign units to entities in the model diagram

- Right-click the entity to which you want to assign units, choose **Units**, and then choose the units that you want:
  - To choose units that are already defined, select the category that contains the units, and then select the units you want.
  - To choose units that have already been assigned to another entity in the model, choose **Units in This Model**, and then select the units you want.
  - To choose units that you have created previously, choose the **My Units** category.
  - To create a new unit of measure and then assign it to the entity, choose **Unit Editor**. For more information about creating units, see [Creating units](#), below.

The units you selected are assigned to the entity.

## To assign units while creating or editing entities

1. In the stock or converter entity dialog box, click the **Units** button. The Check Units dialog box opens.



The Units of Measure box displays the name of the currently assigned units. If no units are assigned, the box displays "Unspecified Units".

2. Click the Units of Measure box to select the units for the entity.

*Note:* If the unit of measure you want doesn't appear in the list, you can create a new unit of measure by selecting **Unit Editor**. For more information, see [Creating units](#), below.

3. Click **OK** to close the Check Units dialog box.
4. Click **OK** to close the entity's dialog box.

# Creating units

If the units you want to assign to an entity are not in the list of available units, you can create a new unit of measure by adding on to the list or by deriving a unit of measure from one or more existing units. Units you create appear under the "My Units" category when you are assigning units to entities.

You can also remove units that you create (those that appear under "My Units"). For more information, see [To remove units you've created](#), below.

The following sections describe how to create new units from scratch, how to create new units by deriving them from other units, and how to remove units you've created.

## To create new units from scratch

1. From the Interface, Model, or Equation menu, choose **Unit Editor**.

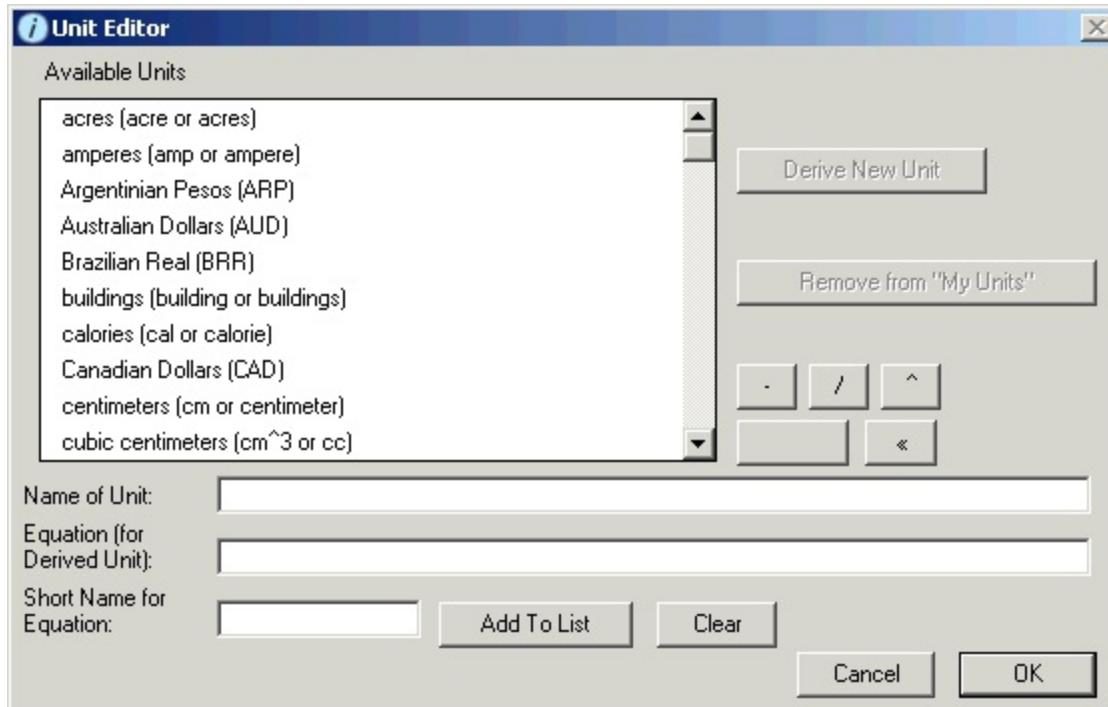
-or-

Right-click an entity, choose **Units**, and then choose **Unit Editor**.

-or-

In the Check Units dialog box, select **Unit Editor** in the Units of Measure box.

The Unit Editor dialog box opens.



The "Available Units" section displays a list of all units that can be used within the current model. Units are listed in alphabetical order, by name. If an equation and/or short name is specified for a unit of measure, that

information appears in parentheses following the unit name. An asterisk (\*) before the unit name indicates a unit of measure that you have created. Units you create are available in the My Units category and are available to all of your models.

2. In the Name of Unit box, type a unique name for the new unit of measure. The name appears in the "Available Units" list and in the shortcut menu that appears when you are assigning units to entities.
3. In the Equation (for Derived Unit) box, either type the equation for the new unit of measure, or type a short name that can be used when referring to this unit of measure in equations for units that are derived from this unit of measure. If you leave the box blank, the software assumes that the unit of measure's name and equation are the same. For more information about creating equations, see [Creating Equations for Units](#).

You can type the equation directly in the Equation (for Derived Unit) box. In addition, you can use the buttons above the Name of Unit box to edit the equation:

- Click  to add a multiplication sign to the equation (in the Unit Editor, "->" indicates multiplication).
- Click  to add a division sign to the equation. Note that you can have only one "/" in an equation.
- Click  to add a carat ("to the power of" sign to the equation).
- Click  to add a blank space to the equation.
- Click  to delete the previous character (or space) in the equation.

4. In the Short Name for Equation box, optionally type a short name for the unit of measure. This short name may also be used when [creating equations](#) for units that are derived from this unit of measure.
5. When you are finished specifying the new unit of measure, click **Add To List**. The new unit of measure is added to the list in alphabetical order, with an asterisk (\*) before its name.

## To derive units from existing units

1. From the Interface, Model, or Equation menu, choose **Unit Editor** (or right-click an entity, choose **Units**, and then choose **Unit Editor**). The Unit Editor dialog box opens.
2. In the "Available Units" list, select the existing unit of measure from which you want to derive the new unit of measure.

3. Click **Derive New Unit**. The name, [equation](#) (if any) and short name (if any) for the selected unit of measure appear in the boxes at the bottom of the dialog box.
4. Edit the name, equation, and short name for the new unit of measure.
5. When you are finished specifying the new unit of measure, click **Add To List**.

## To remove units you've created

You can remove units of measure that you have created (those that appear in the "My Units" menu and that are identified by an \* in the "Available Units" list). You cannot remove any units that are currently being used in the model; if entities are assigned to the unit of measure you want to remove, you must reassign those entities to other units of measure.

---

*Note:* If the unit of measure is used by entities in other models and you remove it in the current model, the unit of measure will not be removed from the other models unless you also reassign those entities to other units.

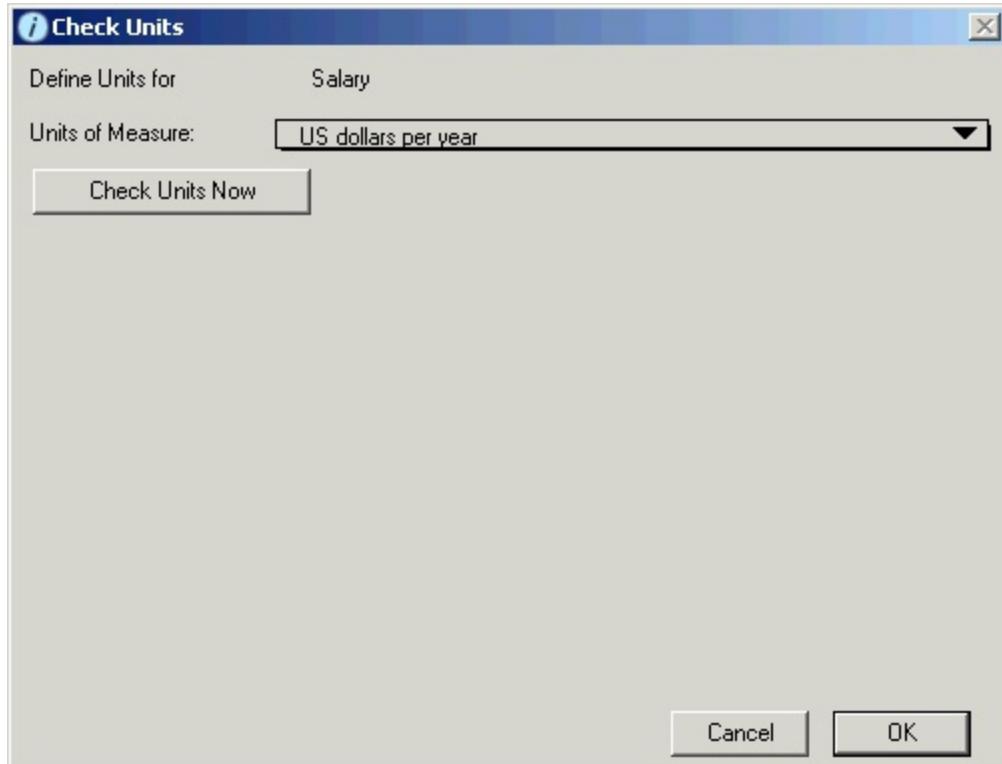
1. In the "Available Units" list in the Unit Editor dialog box, select the user-defined unit of measure that you want to remove.
2. Click the **Remove from "My Units"** button. The unit of measure is removed from the "Available Units" list. All model entities assigned to the unit of measure no longer have an assigned unit of measure.

# Checking unit consistency

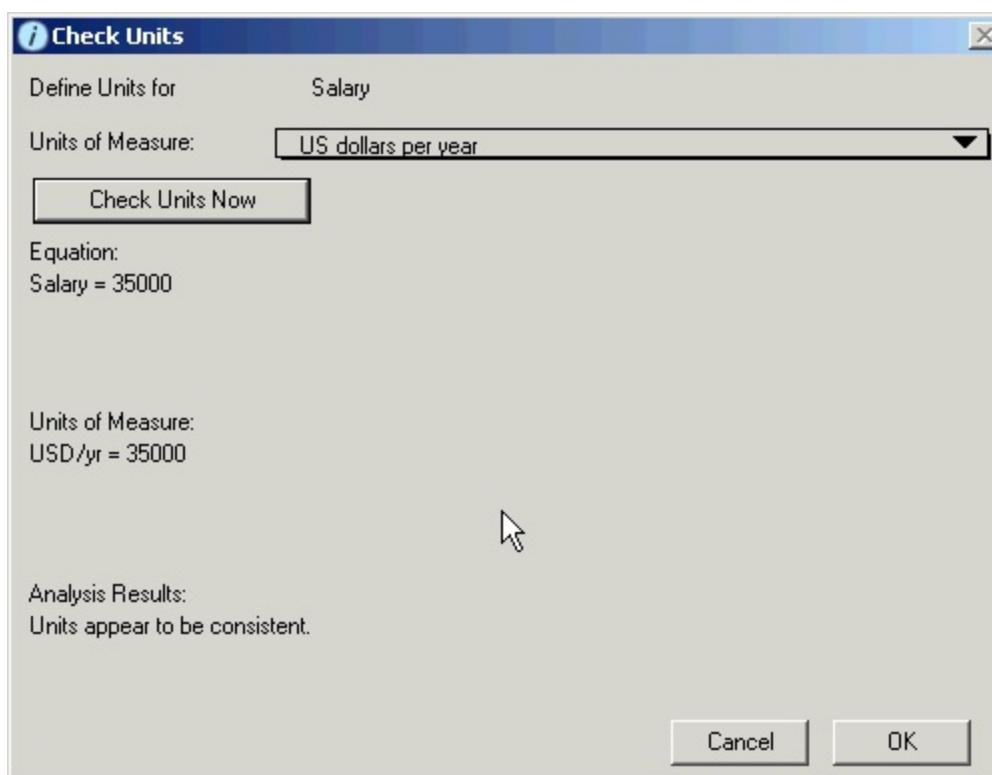
Checking unit consistency allows you to see if equations for units are defined properly and if assigned units are consistent across the model. You can check units for a specific entity or for the entire model.

## To check unit consistency for a single entity

1. In the stock or converter entity dialog box, click the **Units** button. The Check Units dialog box opens.



2. Click the **Check Units Now** button. The dialog box displays the unit check results.



- The Equation line shows the current equation for the relationship. You must have an equation defined for unit checking to operate.
- The Units of Measure line shows the left- and right-hand side of the equation, as units. If there are multiple terms in either side of the equation, each term is enclosed in parentheses.
- The Analysis Results line indicates whether the units are consistent.

## To check unit consistency for the entire model

- From the Run menu, choose **Check Units**. The software checks all units assigned to entities in the model.

If no problems are found, a message appears that tells you that all units in the model are consistent. Click **OK** to close the message.

If it finds any problems, a message appears that tells you that some units are inconsistent. Click **OK** to close the message. The entities that have inconsistent units are highlighted in the model diagram. To troubleshoot the inconsistencies, open each highlighted entity, click the **Units** button in the entity's dialog box, and verify that the correct unit of measure is assigned to the entity.

If the Enforce Unit Consistency check box is selected in the [Model Preferences dialog box](#), an "!" appears in any entity that fails the unit consistency check. You cannot run a simulation until all inconsistencies are resolved.

**Note:** The unit analysis capabilities in the software do not apply to the following model constructs: arrays, outflows from conveyors, outflows from ovens, and unit-converted flows.

---

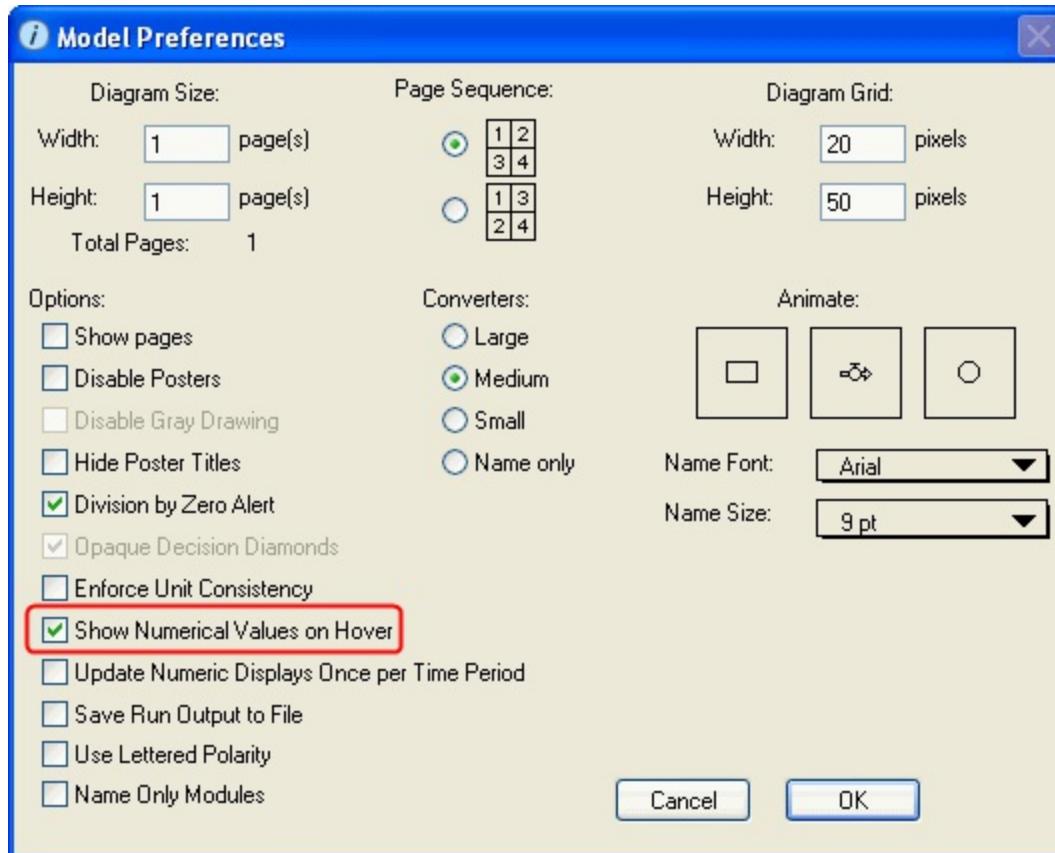
# Reading Numerical Values Directly from Model Elements

**Purpose:** When constructing a model, especially during de-bugging, it's often useful to be able to determine the numerical value taken on by a given variable at a particular point in time. The "hover and discover" feature enables you to do this. By "hovering" over any model variable while a simulation is paused or stopped, you will be able to see the numerical value of the variable at that point in time. Additionally, a small graph will be displayed for any variable that has been loaded into a table or graph.

**Basic Operations:** To activate this feature, choose "Model Prefs..." under the Model menu on the Map or Model layer. When the dialog appears, check the "Hover to Show Numerical Values" box as shown in Figure 4-45. OK the dialog. Whenever you pause a simulation, you will then be able to hover over any model variable to display its current numerical value.

*Tip:* To display a graph on hover for *all* model variables, check the Analyze Mode check box in the Run Specs dialog.

Figure 4-45 The "Hover & Discover" Feature





# Introduction to Tools

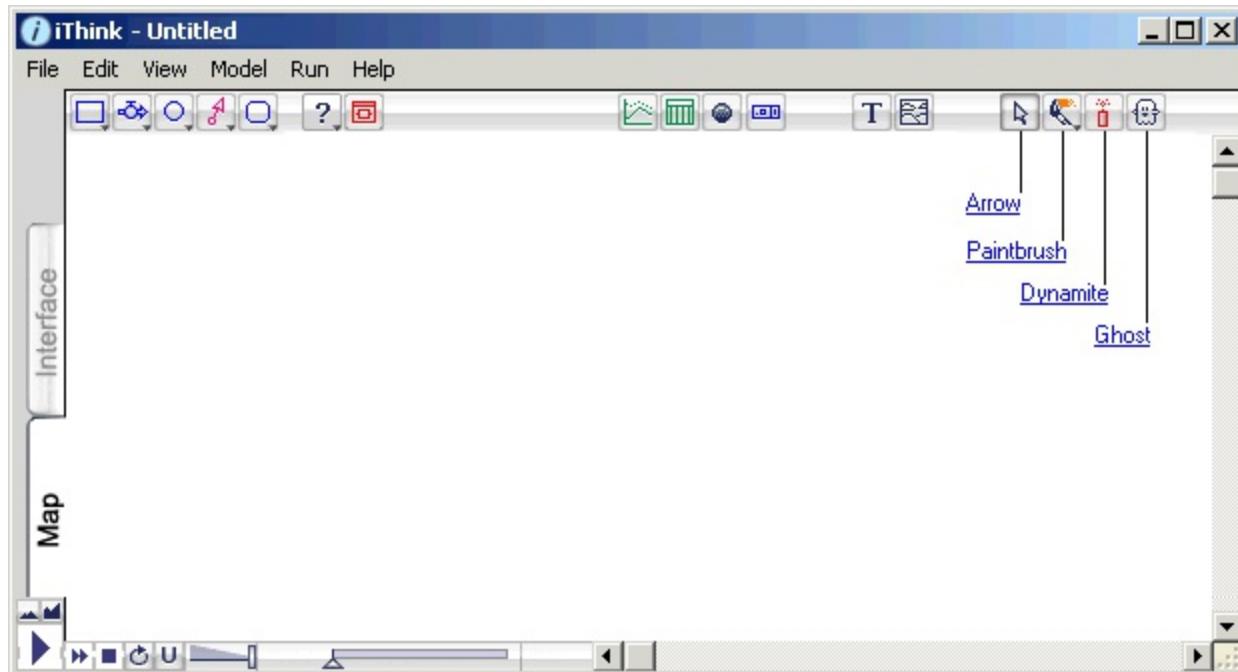
The topics in this section describe the four tools available to you within the software:

- [The Arrow](#)
- [The Paintbrush](#)
- [The Dynamite](#)
- [The Ghost](#) (Map and Model layer only)

These tools enable you to perform basic activities on your model entities, such as selecting, moving, coloring, deleting, and defining.

Figure 5-1 indicates the location of each of these tools. Click the icon or name of a tool in the figure to read about that tool.

*Figure 5-1 An Overview of Tools*



 [Related Topics](#)

# The Arrow

Click this button to use the Arrow tool. The Arrow is a general-purpose editing tool. You use it to select, move, open, and edit building blocks and objects. You also use it to manipulate Sub-model and Space Compression spaces on the Map and Model layers, and to select items on the Equations layer. In accomplishing these tasks, the point of the arrow is the "hot spot."

Because of its general utility, the Arrow is the default selection from among the palette elements. After you have clicked on the diagram while using any of the building blocks, tools, or objects, the software will revert you to the use of the Arrow.

---

*Tips:* By depressing-and-holding the alt key (Windows) or option key (Macintosh) after using a particular building block, tool, or object, you can re-use it without re-selecting it from the palette.

To multiple-select items, either drag-select or shift-click. Your Windows or Macintosh manual provides details on selection techniques.

---

 [Related Topics](#)

# The Paintbrush

The purpose of the Paintbrush is to add color to model elements. With it, you can color building blocks, objects, backgrounds, and output from your model. The process of coloring your model is straightforward. Begin by choosing a color. Then, color!

## Choosing colors

The first step in coloring anything is to decide which color you wish to use. To select a color, click-and-hold on the Paintbrush. A palette containing 2, 4, 16, or 256 color squares will drop-down (the number depends upon how many colors your computer allows, as well as the video driver you're using under Windows, or the setting you've made in the Monitors Control Panel on the Macintosh). Drag down and select a color. When you release your click, the palette will retract, and the Paintbrush will sport the color you have chosen. Click on the model element you'd like to color or drag-select multiple diagram elements which are to be the same color. The Paintbrush will remain selected until you click on a building block, tool, object, or surface. Be careful not to click on a surface such as the Diagram, unless you wish the surface to turn the color indicated by your current Paintbrush setting! When changing the color of the diagram surface, use a fast click.

## Changing Palette Colors

If you want to tear off the palette (so as to keep it in view for subsequent color selection), click-and-hold on the Paintbrush, then drag down until the cursor runs off the bottom of the palette. A transparent image of the palette will follow your cursor. Once you've positioned this image where you'd like the palette to sit, release your click. To put the palette away, click on its close box.

If you would like to substitute new colors for the ones in the palette, hold down the alt key (Windows) or option key (Macintosh), then click-and-hold on the Paintbrush. Drag down to a square whose color you would like to change. This square should be one of those located below the first row of colors. Then release your click. A dialog will appear, with the color that you have selected from the color palette. You then can create any new color by adjusting the hue, saturation, and brightness, or the red, green, and blue density.

Once you have chosen a color, the next task is... yes, to colorize. Figure 5-2 summarizes the coloring options available to you within the software.

*Figure 5-2 Paintbrush Operations*

**do this with the**

## To colorize this

## Paintbrush

## Notes

Process Frame	Click within the border	
Bundled Flow	Click on the flow	
Bundled Connector	Click on the connector	
Loop Pad Page	Click anywhere within the page except on Label	
Loop Pad Page Variables	Click on Label (R or C)	
Stock	Click within its borders	
Flow	Click on the flow regulator	color of attached clouds will change
Converter	Click within its borders	
Connector	Click on its take-off button (circle end)	
Cloud	Click on cloud	will change cloud's color only
Sector Frame	Click along Sector border or within its header	
Text Box and Enclosed Text	Click along Block border	if clicked where there is no text, color will spill to diagram
Graphics Frame	Click within the border	
Graph Pad/Loop Pad/Table Pad Icon	Click on Pad icon	
Graph Pad/Table Pad Page Background	Click anywhere within page except variable name	
Graph Pad/Table Pad Page Variables	Click on the variable's name	data associated with variable will be colored
Scatter Plot Trace	Click on Legend (e.g. "X vs. Y") at top of page	
Button, Graphical Input Device (GID), Knob, List Input Device (LID), Numeric Display, Slider, Switch, Warning Device	Click anywhere on object	
Decision Process Diamond / Sub-model	Click within its borders or background of open space	
Layer Background	Click on surface in unused location	
Equation icons in Equation layer	Click on equation	color will also change on diagram
Multiple Items on Diagram	Select color, then drag-select items	

Default Colors for Building Blocks/Object Icons on Toolbar

Select color, then hold Alt key (Windows) or option key (Macintosh) and click on the item on the toolbar



## The Eyedropper

There are times you would like to duplicate the color of an entity but you are not sure which shade you used. To identify the color of an entity, select the paintbrush; do a control-shift (Windows) or command-shift (Macintosh), and the Eyedropper will appear. Click on the entity and the paint brush will take on the color of that entity. Now you can color your model as before.

 [Related Topics](#)

# The Dynamite

The Dynamite is used to clear building blocks and objects from the Interface, Map, and Model layers. It will also clear equations (with their associated building block representation) from the Equations space. Because dynamite is a powerful tool, you should use it with caution. We strongly recommend that you always employ the "look-and-click" technique when using the Dynamite. Look-and-click means positioning the fuse of the Dynamite (the active region for this Tool) above the entity to be detonated. You should see the item highlight. If some other building block or object highlights instead, back off! Once you see the correct entity turn highlight, click away! [Boom!](#)

For purposes of dynamiting, the active regions for building blocks and objects are the same as those for coloring or selecting these entities. Position the fuse of the Dynamite, click-and-look, release. Figure 5-3 summarizes these locations for you.

On Loop, Graph and Table Pad pages, you have a few more dynamite options. These are illustrated in Figures 5-4, 5-5 and 5-6. Briefly, to dynamite variables on Graphs and Tables, position the fuse of the Dynamite on their name, then click-and-look...history! To dynamite a page on a Loop, Graph or Table pad, click on the page-up portion of the page-turn button. To dynamite an entire Pad, click on the Pad icon (located on the Diagram). The Dynamite also can be used to clear data from graphs or tables, and to reset scales on graphs. To clear data from a graph or table, position the Dynamite within the confines of the graph or table "data space" (i.e., not in the header, footer, etc.), then click. To remove the scale for a scaled variable on a graph, click with the Dynamite on either the upper or the lower scaled value for the variable. The variable then will become "locally unscaled," which means that it will not have any consciously-chosen scale on that particular page of the Pad. Instead, the variable will carry a "last simulation" scale (i.e., its scale will be defined by the minimum and maximum values taken on in the last simulation of the model). If two or more variables are set on the same scale, and are next to each other on the list, clicking the dynamite will descale them one at a time.

If you click on the lower scale value, it will remove the scale from the last variable first. If you click on the upper scale value, it will begin with the first variable.

You should note that it is not possible to remove a global scale for a variable by using the Dynamite. Detonating the scale of a globally-scaled variable will only knock the scale down to a "locally unscaled" status. For information about scaling variables in graphs, see [Scaling Variables: Within the Graph Pad's Define dialog, you can set the scale for any variable that has been loaded into](#)

[the Selected list of a Time Series, Bar, or Scatter Plot. If you don't set a scale, the software will automatically set one for you. For any variable or set of variables, you have four basic options: unscaled, locally scaled, globally scaled, and locally unscaled. A double-headed arrow will appear next to each variable in the Selected list, indicating its scaling status. Figure 6-54 illustrates these different options..](#)

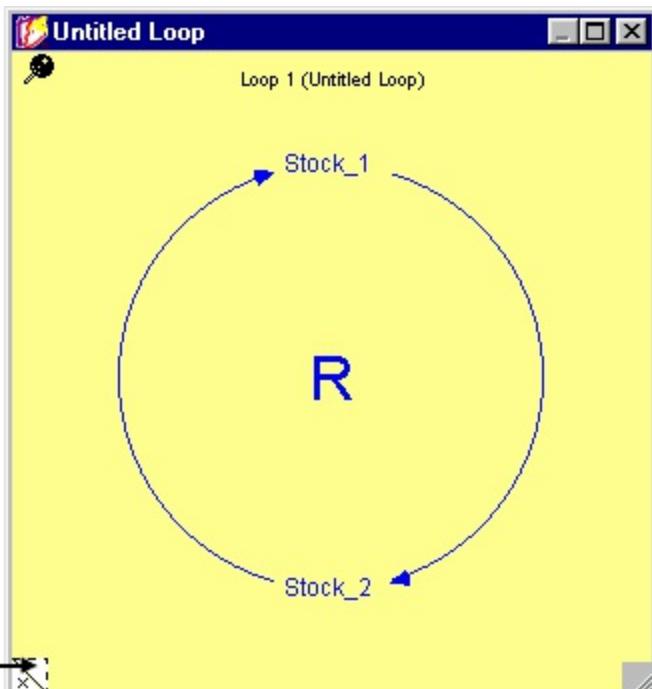
**Note:** If you delete a variable which has an associated controller (i.e., Slider, Switch, Knob, etc.), the associated controller will also be deleted.

*Figure 5-3 Dynamite Operations*

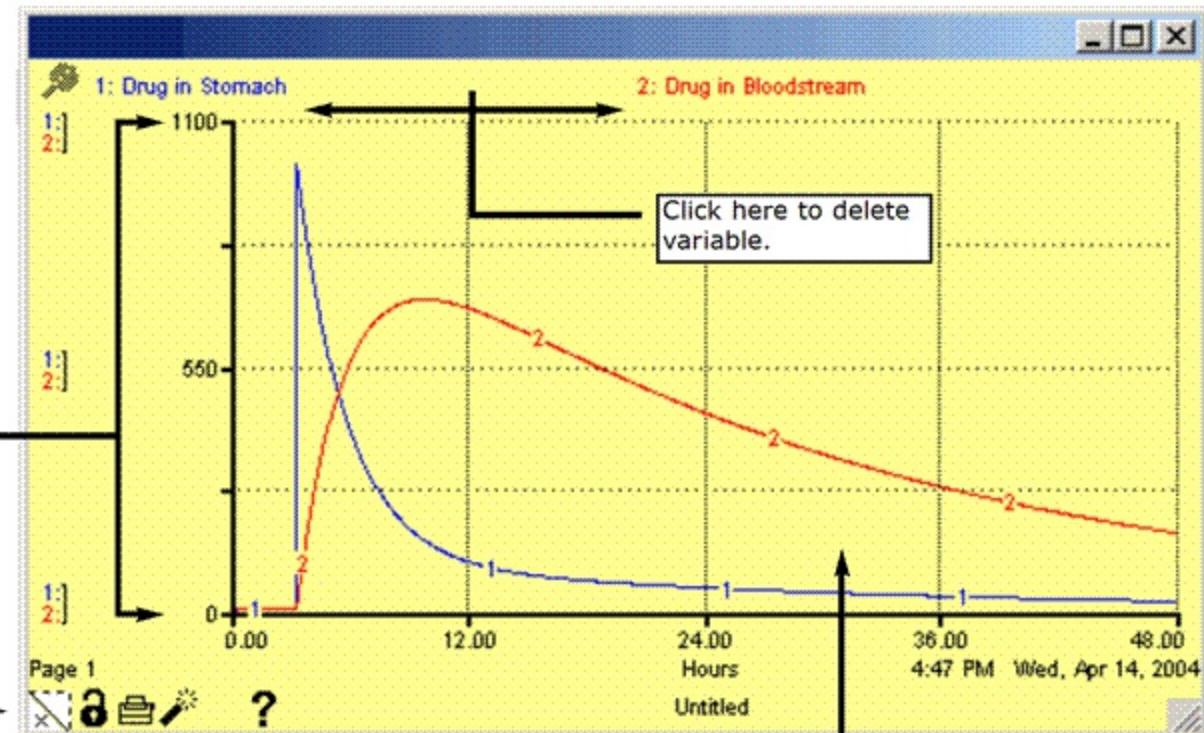
To Delete This	do this with the Dynamite	Notes
Process Frame	Click within the frame	
Bundled Flow	Click on flow	
Bundled Connector	Click on connector	
Button	Click anywhere on object	
Graphical Input Device (GID)	Click anywhere on object	
List Input Device (LID)	Click anywhere on object	
Slider	Click anywhere on object	
Stock	Click within its borders	
Flow	Click on the flow regulator	
Converter	Click within its borders	
Connector	Click in its take-off button	
Sector Frame	Click along sector border or within its header	
Numeric Display	Click anywhere on object	
Text Box	Click along block border	
Graphics Frame	Click anywhere on object	
Graph/Loop/Table/Pad Icon	Click on pad icon	
Graph Pad/Loop Pad/Table Pad page	Click on page-turn button	see Figures 5-4, 5-5, and 5-6
Graph Pad/Table Pad page variables	Click on variable's name	see Figure 5-5 and 5-6
Graph Pad page data	Click within axes or on graph dynamite icon	see Figure 5-5

Graph Pad page scale	Click on min or max value	see Figure 5-5
Table Pad page data	Click on a data point or table dynamite icon	see Figure 5-6
Equation in Equation layer	Click on equation	removes associated entity on diagram
Knob, Switch, Warning Device	Click anywhere on object	

*Figure 5-4  
Dynamite Operations on Open Loop Pads*



*Figure 5-5  
Dynamite Operations on Open Graph Pads*



**Figure 5-6**  
*Dynamite Operations on Open Table Pads*

Click here to remove variable.

Click here to clear all data

Click here to delete page.

OR

Click anywhere in here using Dynamite to clear all data.

Hours	Drug in Bloodstream	Drug in Stomach	absorbing
1.0000	0.85	2.13	0.59
2.0000	1.40	1.53	0.41
3.0000	1.74	1.13	276.22
4.0000	272.38	724.91	203.21
5.0000	469.68	521.70	138.74
6.0000	575.94	382.96	95.05
7.0000	644.52	287.91	65.43
8.0000	681.20	222.47	45.35
9.0000	696.66	177.13	31.71
10.0000	698.13	146.42	22.44
11.0000	690.46	122.98	16.13
12.0000	676.94	106.85	11.82

 [Related Topics](#)

# The Ghost

The last tool on the palette is the Ghost. The Ghost is available only on the Map and Model layers. Its purpose is to make replicas, aliases, or shortcuts for individual stocks, flows, and converters. You can also use the Ghost to [connect variables across modules and to automatically assign module inputs and outputs](#).

A Ghost of an entity has no independent identity - it is simply an image of the building block from which it was ghosted. The ghosted replica has no equation of its own. When you double-click on a ghosted replica, the dialog box that opens actually belongs to the original from which the replica was made. No matter how many ghosted replicas of a given building block you create, only one dialog box exists - because only one building block exists! A ghost adds no real structure to a model.

In particular, ghosted stocks can have no inflows or outflows; ghosted flows and ghosted converters (when you "Ghost" a flow, its Ghost will appear as a converter) can have no input connectors. Ghosts are thus read-only information holders. You can draw connectors from them. Nothing can go into them.

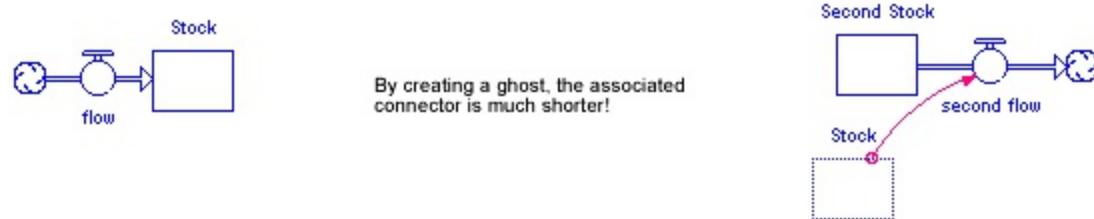
To "Ghost" a stock, flow, or converter, begin by selecting the Ghost tool. Click once on the stock, flow, or converter to be ghosted. The cursor will change to an image of that building block. Then, move the cursor to the desired location. Click once to deposit the Ghost. You're in business!

In your modeling efforts, Ghosts serve the primary role of keeping your diagram tidy. When connectors might otherwise run all over the screen, leading to diagram "spaghetti," ghosted images can help to the connections neat and clean. Figure 5-7, illustrates this role of Ghosts.

*Figure 5-7 Ghosting as an Antidote to Spaghetti*



Without ghosting, it's necessary to stretch a connector across an entire page ...



By creating a ghost, the associated connector is much shorter!

## [Related Topics](#)



# Introduction to Objects

The topics in this section document the objects available to you within the software. The objects available to you depend on the layer on which you are working.

The following objects are available on the Interface layer only:

- [Loop Pad](#)
- [Switch](#)
- [Graphical Input Device](#)
- [Knob](#)
- [List Input Device](#)
- [Slider](#)

The following objects are available on the Map and Model layers only:

- [Sector Frame](#)

The following objects are available on all layers:

- [Button](#)
- [Graph Pad](#)
- [Table Pad](#)
- [Status Indicator](#)
- [Numeric Display](#)
- [Text Box](#)
- [Graphics Frame](#)

Figures 6-1a and 6-1b indicate the location of each of these objects. Click the icon or name of an object in either figure to read about that object.

*Figure 6-1a Objects on the Interface Layer*

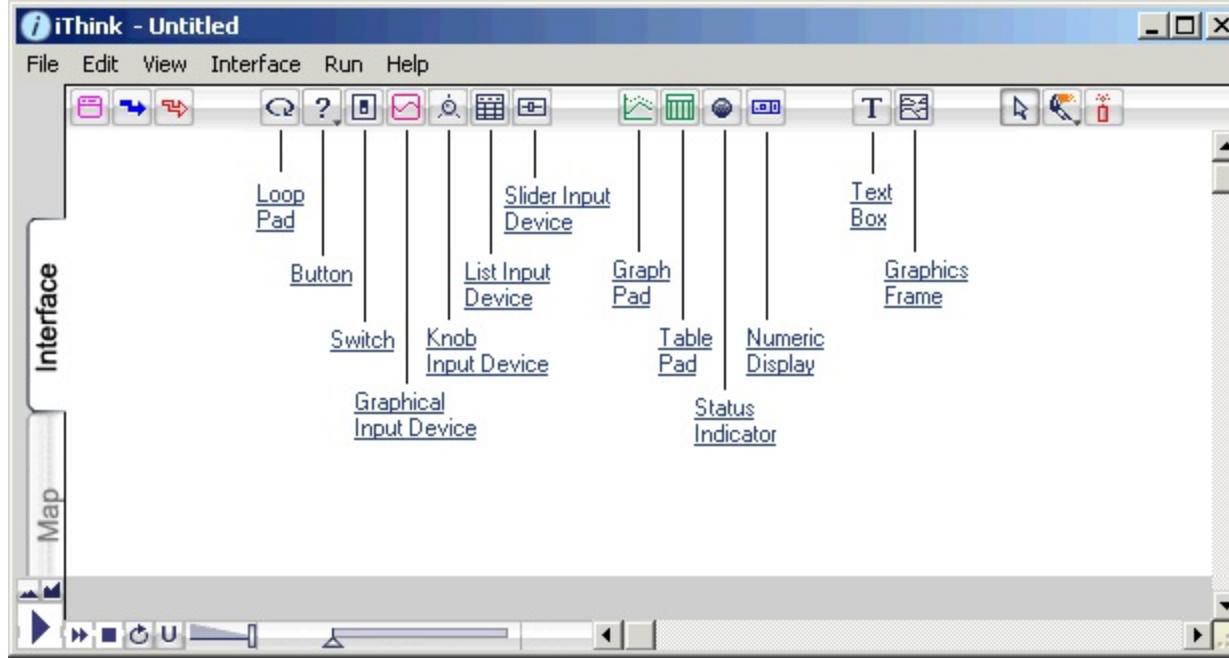
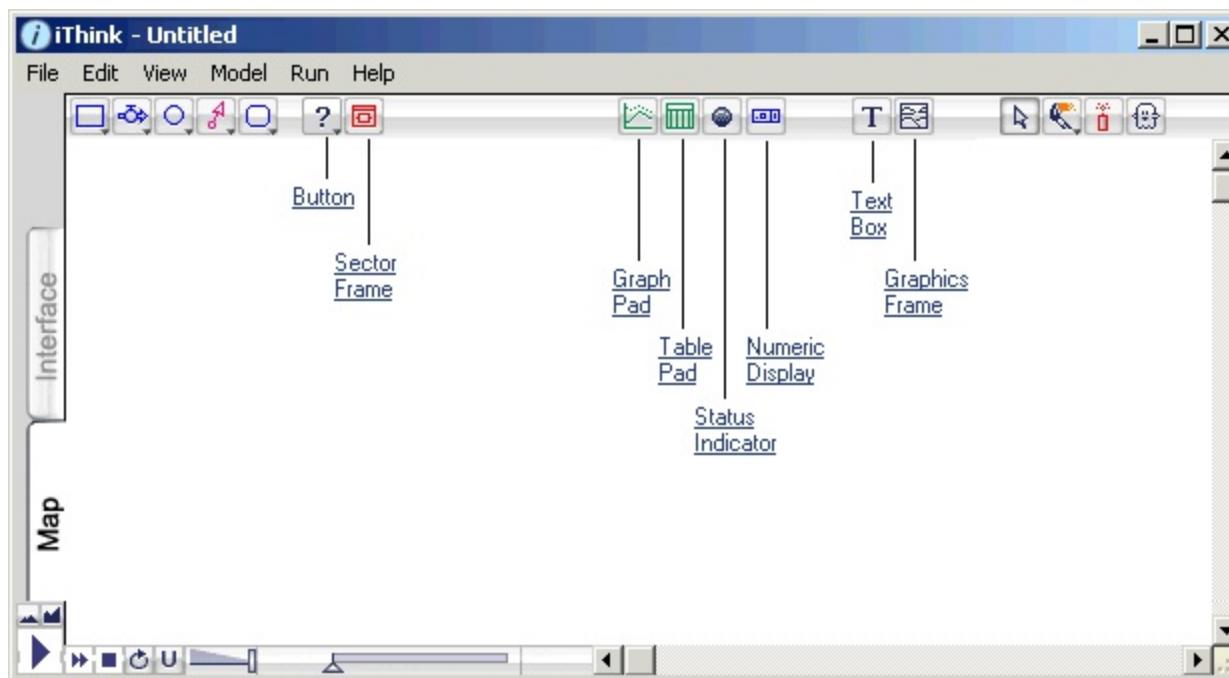


Figure 6-1b  
Objects on the Map and Model Layers



# Loop Pad

In many instances, you will want to provide consumers of your models with simple loop diagrams. Loop diagrams are simple pictures that identify the cause and effect processes that work to generate dynamic behavior patterns. The software's Loop Pad Object enables you to create these pictures based on the underlying model structure.

This section documents the operation of the Loop Pad object. It includes the following sections:

## [Loop Pad Surface Operations](#)

- [Creating/Moving/Resizing/Closing](#)
- [Pad Icon Operations](#)
- [Pinning and Unpinning](#)
- [Turning Pages](#)
- [Other Surface Operations](#)

## [Loop Pad Dialog Operations](#)

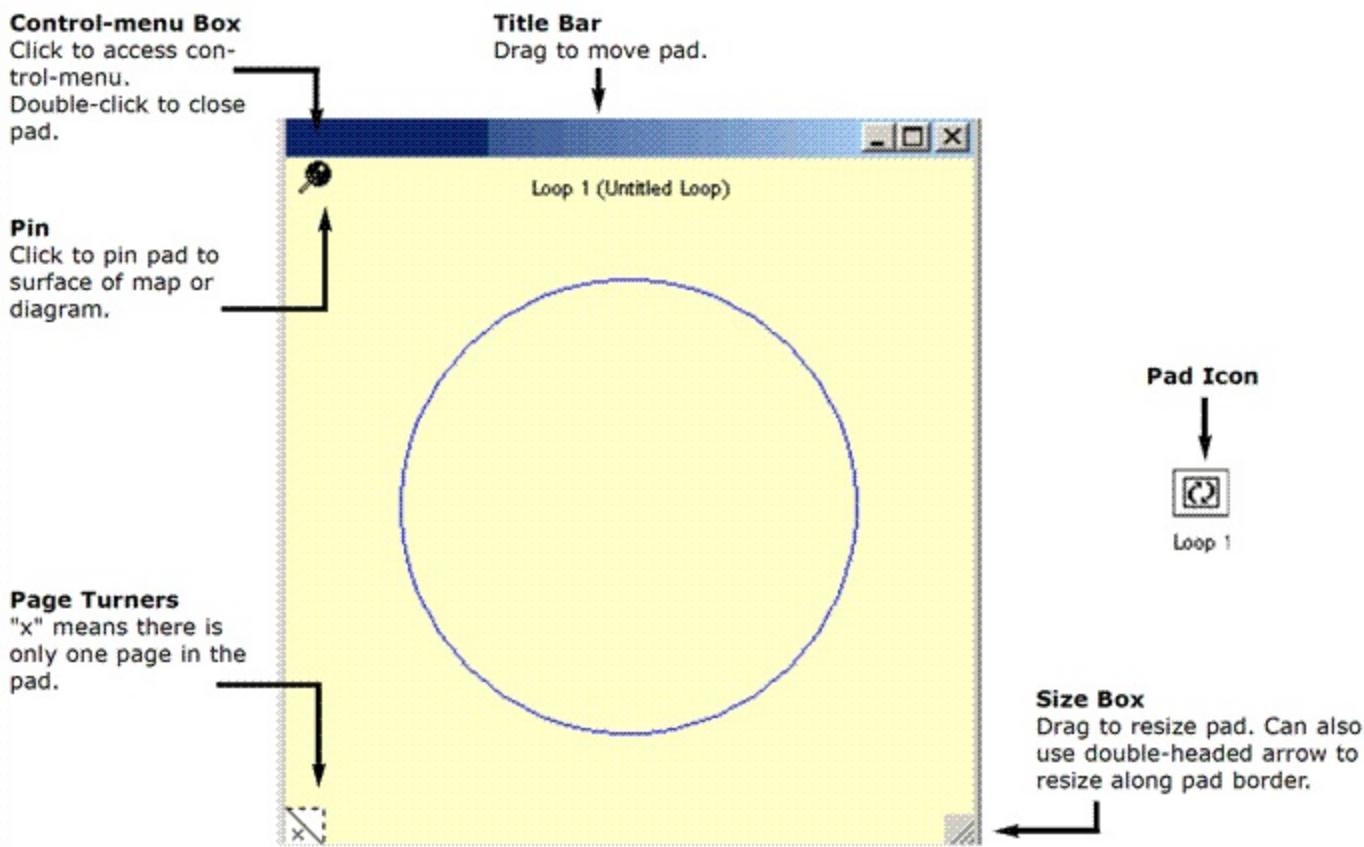
- [Loading/Removing/Exchanging Variables](#)
- [Choosing Loops](#)
- [Label Options](#)
- [Titling a Page](#)
- [Adding Pages](#)

## [Related Topics](#)

# Loop Pad Surface Operations

**Creating/Moving/Resizing/Closing a Loop Pad:** To create a Loop Pad, select the Loop Pad icon from the tool bar. Then, deposit the icon in the desired location by clicking once. When you create a Pad, you'll get a blank page such as the one shown in Figure 6-3. Note that Figure 6-3 shows where to click and/or drag to move, resize and close the Pad. It also identifies the salient features of the default Loop Pad page.

Figure 6-3 Default Loop Pad Page



**Pad Icon Operations:** As shown in Figure 6-4, associated with the Loop Pad is a Pad icon. The icon exists on the surface of the Interface layer. When the Pad is accessible (either because you have closed the Pad or because you have made active the layer containing the Pad), it is amenable to a variety of operations. These are summarized in Figure 6-4.

Figure 6-4  
Loop Pad Icon Operations

**(a) Unselected Loop Pad Icon**



To Select: Click within icon's border. To Open: Double-click within the icon's border. Or, select the icon, and choose Open Selection... from Map menu. Opening will take you to the last page viewed on the Loop Pad.

To Move: Click-and-hold within icon's border; drag to desired location.

To Move Name Plate: Click-and-hold name plate of unselected Pad icon. Drag name around icon to desired location. Hold Control key (Windows) or command key (Macintosh) while dragging, to constrain position of name plate to North/South/East/West. Control-click (Windows) or command-click (Macintosh) on name plate to cause it to snap to nearest cardinal position.

**(b) Selected Loop Pad Icon**



To Name: When a single Pad icon is selected, its name will become highlighted. Replace the highlighted name by typing. Or, move the Hand over the highlighted name. When the cursor changes to an I-beam, click to deposit it. Then edit as you would with a text processor.

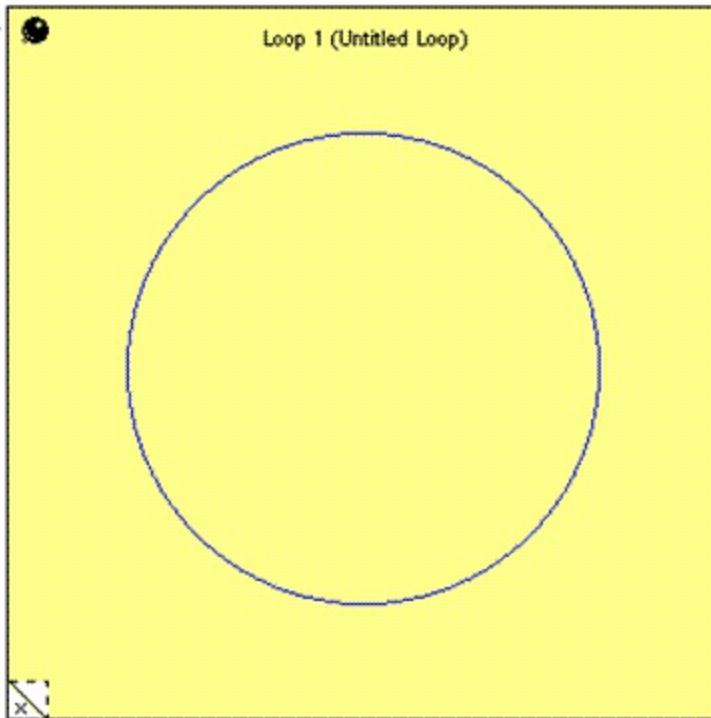
**Pinning and Unpinning:** The push-pin icon on the Loop Pad is used to pin and unpin the entire pad to the surface of the Interface layer. Pinned pads will move along with the layer to which they are affixed, when you click, drag, scroll, etc., on the layer. Unpinned pads will "float," either above or below the layer. Click once on the push-pin to pin the pad to the surface. Click again on the push-pin to unpin the pad. Figure 6-5 illustrates a pinned Loop Pad. By default, Loop Pads are unpinned when you first add them.

**Notes:** When a pad is pinned, its title bar and resizing controls disappear. To move or resize, you must first unpin the pad. Because a pinned pad is attached to its layer, access to building blocks and objects is retained while the pad is extant.

The Pin will be grayed out unless the entire area of the pad is within the borders of the Interface layer.

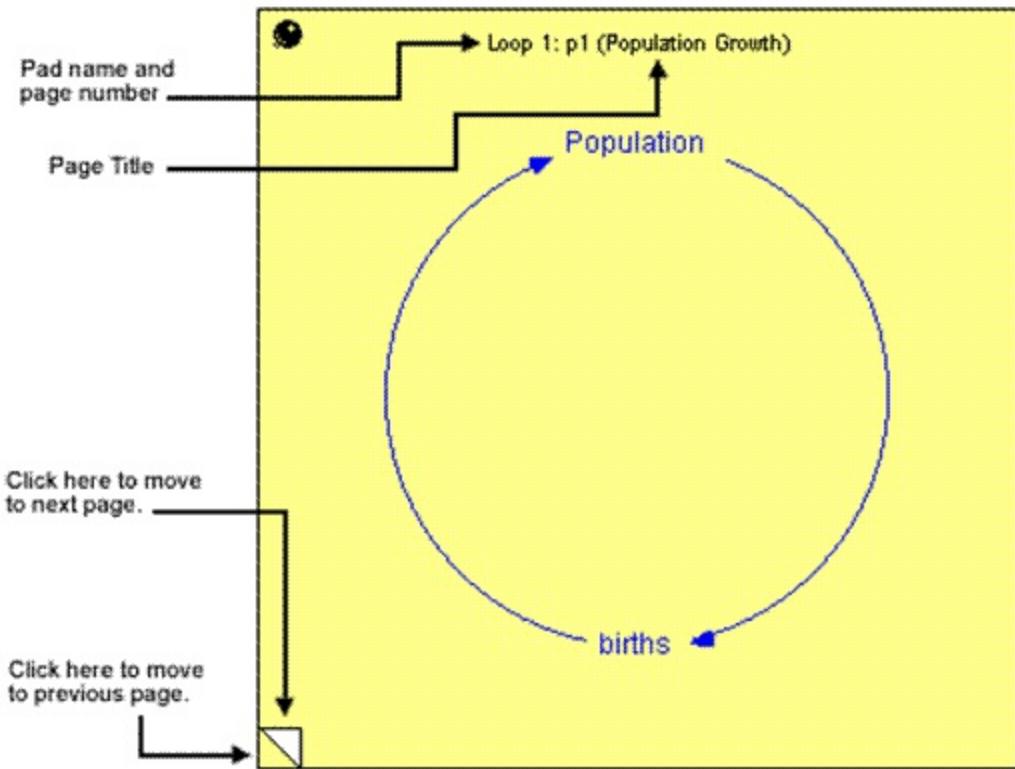
*Figure 6-5  
A Pinned Loop Pad*

Note that the pin is in the downward position. Click on it to unpin the pad.



**Turning Pages:** When a Loop Pad contains more than one page, the page-turning apparatus on its lower left corner will become active. The Pad will also tell you what page of the Pad you are on. As shown in Figure 6-6, a click on the upper triangle will move you to the next page in the pad. A click on the lower triangle will move you to the previous page.

*Figure 6-6  
Turning Loop Pad Pages*



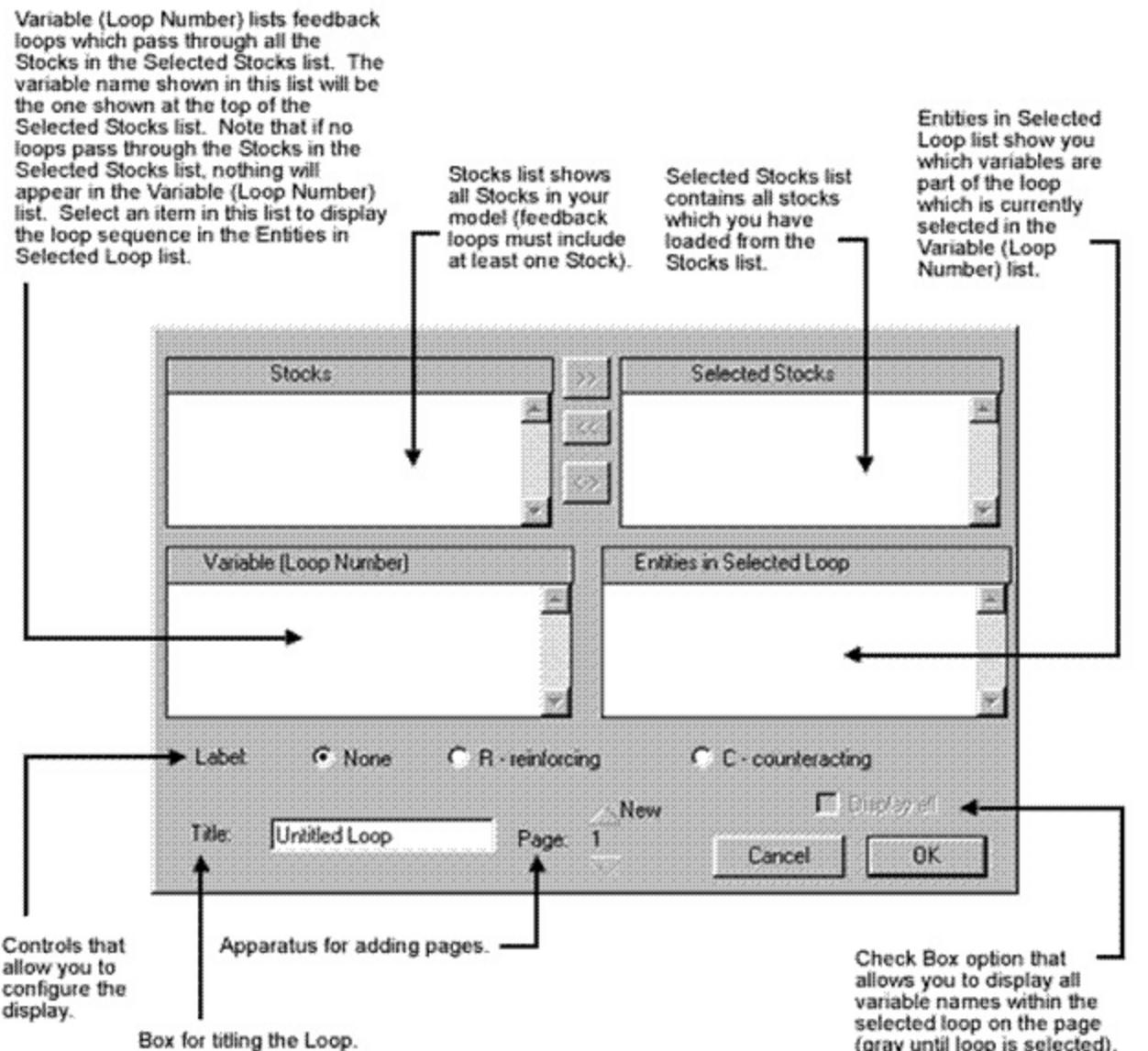
**Other Surface Operations:** Using the Paintbrush tool, you can change the color of backgrounds, variable names, and Pad icons. For more information, see [The Paintbrush](#). You can delete pages using the Dynamite tool. For more information, see [The Dynamite](#).

 [Related Topics](#)

# Loop Pad Dialog Operations

To enter the Loop Pad's Define dialog, use the hand tool to double-click anywhere within the Loop Pad (except the title bar). Or, with an unpinned, active pad, choose Define Loop... from the Interface menu. The Loop Pad Define dialog is shown in Figure 6-7. Define dialog operations are described in the text which follows.

Figure 6-7 The Loop Pad Dialog



**Loading/Removing/Exchanging Variables:** The two lists at the top of the dialog, along with the buttons between the lists, are used to load, remove, and exchange the variables to be displayed in the Loop Pad page. All stocks will appear in the Allowable list. Stocks to be displayed in the loop appear in the Selected list. When a variable has been loaded to the Selected list, it will

appear gray in the Allowable list. Here's how to move variables between the two lists:

- *Loading Variables:* To load a variable to the Selected list, double-click the variable name in the Allowable list. Or, select it, and click the >> button. To select multiple variables, simply drag-select, shift-click (for adjacent variables in the list), or control-click (Windows) or command-click (Macintosh) for noncontiguous variables.
- *Removing Variables:* To remove a variable from the Selected list, double-click the variable name in the Selected list. Or, select it and click the << button between the two lists. To select multiple variables, simply drag-select, shift-click (for adjacent variables in the list), or control-click (Windows) or command-click (Macintosh) for noncontiguous variables.
- *Exchanging Variables:* To exchange one or more variables in the Allowable list for one or more variables in the Selected list, select the desired variable(s) in each list. Then, click on the <-> button. The variables will be exchanged.

**Choosing Loops:** The "Variable (Loop Number)" list displays the number of loops which "pass through" the variables that are currently selected in the "Selected" list. When you select one of the loops in the "Variable (Loop Number)" list, the variables included within that loop are displayed to the right in the "Entities in Selected Loop" list.

---

*Notes:* Once you have selected a loop and clicked OK, only the Stocks within the loop will be displayed. If you want to view all of the variables in the loop you'll need to check the "Display All" check box.

If there are two or more variables in the Selected list, only the loops that pass through all of the variables are displayed.

---

**Label Options:** As shown in Figure 6-7, the Loop Pad dialog provides you with three radio button display options. None is selected by default. The other two options let you label the Loop as either a reinforcing or counteracting loop by displaying the letters "R" or "C" within the center of the loop.

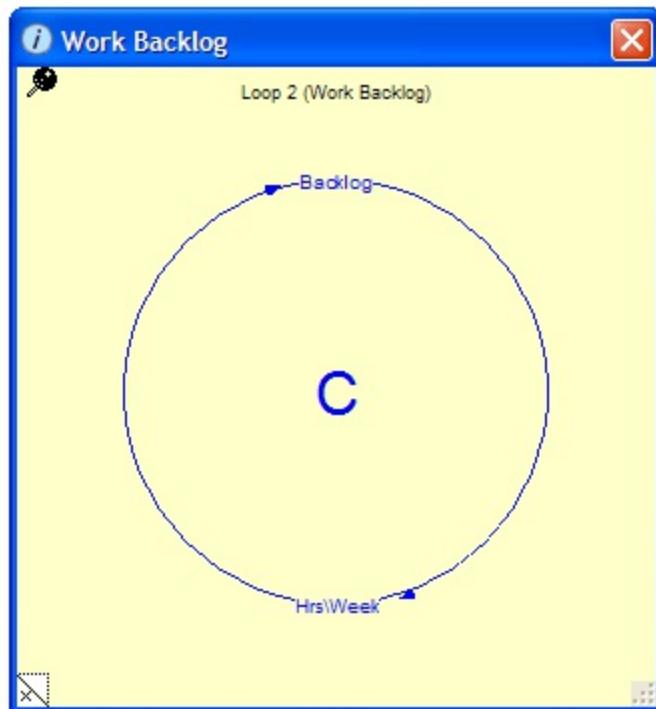
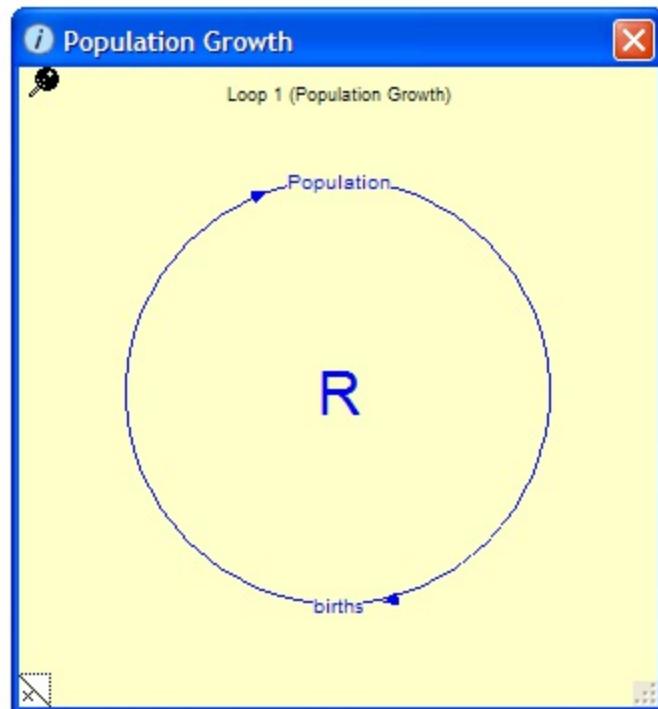
---

*Note:* A counteracting ("C") loop is also known as a balancing ("B") loop.

---

An example of the different labeling options is shown in Figure 6-8.

*Figure 6-8*  
*Loop Pad Labeling Options*



**Titling a Page:** To title a specific page of the Loop Pad, select the text in the Title field of the Loop Pad dialog. Then, type the desired title. The title will appear in the title bar of unpinned Pads. It will appear at the top of pinned loops along with the Pad icon name and the page number.

**Adding Pages:** To the right of the Title Field is the apparatus for adding pages to a Loop Pad. Once you have loaded variables into the Selected list of the first page of the Pad, the up-arrow in the apparatus will become active. Click on the arrow (next to the word "New") to create a new page. You can add as many pages to a Pad as you'd like. You can click on the up- or down-arrows to move between pages from within the dialog. Or, you can click on the page-turn "buttons" (lower left corner) on the Loop Pad surface.

[Related Topics](#)

# Button

Buttons are designed to facilitate an end-user's interaction with your model by reducing the level of software skills and language fluency needed. When "pushed," buttons perform one of several operations such as navigating to a new location, executing a menu command, or providing pop-up information. As you seek ways of simplifying the access to and increasing the impact of your models, it is entirely possible that the Button object will become your new best friend.

This section documents the operation of the Button object. It includes the following sections:

[Button Surface Operations](#)

[Button Dialog Operations](#)

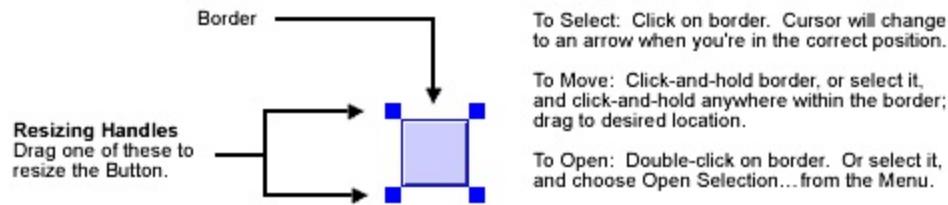
- [Information Buttons](#)
- [Navigation Buttons](#)
- [Menu Buttons](#)
- [Tracing Buttons](#) (available on the Interface layer only)
- [Storytelling Buttons](#) (available on the Interface layer only)
- [Hyperlink Buttons](#)
- [Play Movie Buttons](#)
- [Button Sound](#)
- [Button Appearance](#)

 [Related Topics](#)

# Button Surface Operations

To create a Button, select the Button icon from the Interface or Model toolbars. Then, deposit the icon in the desired screen location by clicking once. You will see the Button, with its resizing handles, displayed as shown in Figure 6-10.

*Figure 6-10 Button Surface Operations*

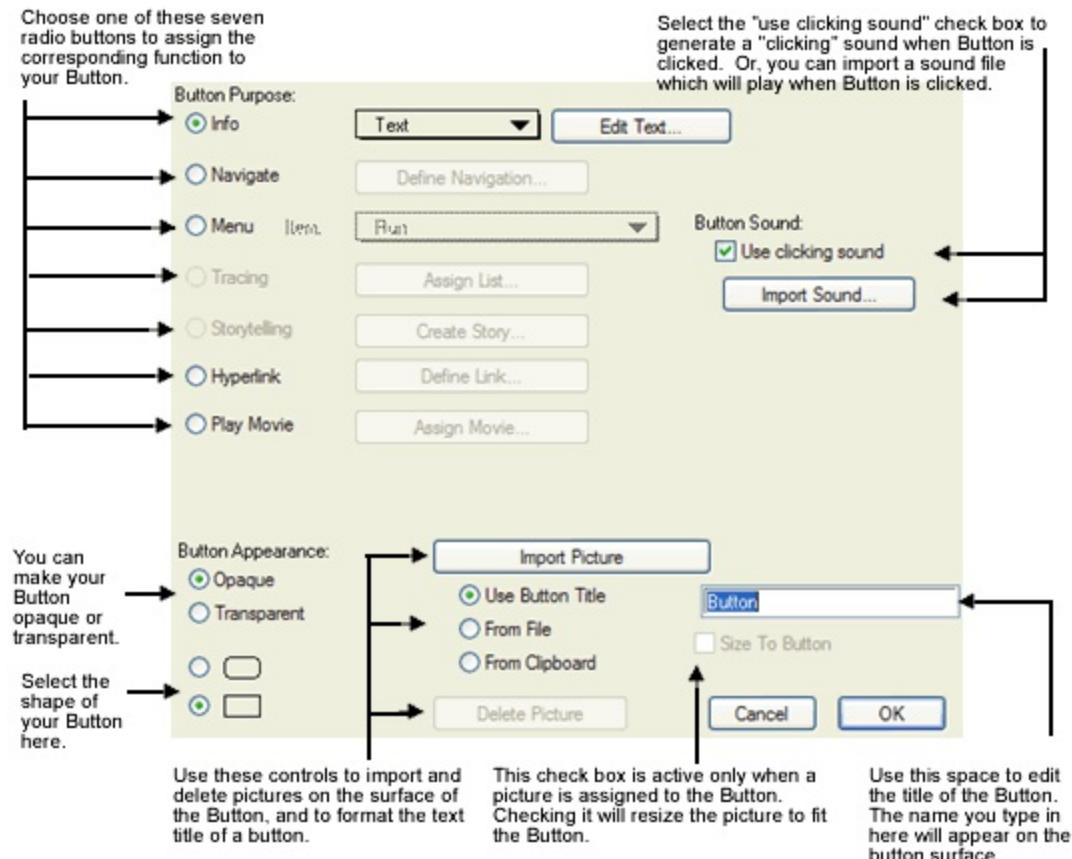


 [Related Topics](#)

# Button Dialog Operations

Double-click on the Button's border to open its Define Dialog box as shown in Figure 6-11 below. Notice that the dialog box has two major sections: Button Purpose and Button Appearance.

Figure 6-11 Primary Button Dialog Box



You can configure a button to accomplish any one of seven major purposes. Your choices are: Information, Navigate, Menu Item, Tracing, Storytelling, Hyperlink or Play Movie. Note that each of the button purposes is unique. A single button can not be used to accomplish multiple purposes. However, in addition to accomplishing a one of the seven purposes, a button can be configured to play a sound when it is clicked.

The following sections describe each type of button you can create and the options available for each button purpose.

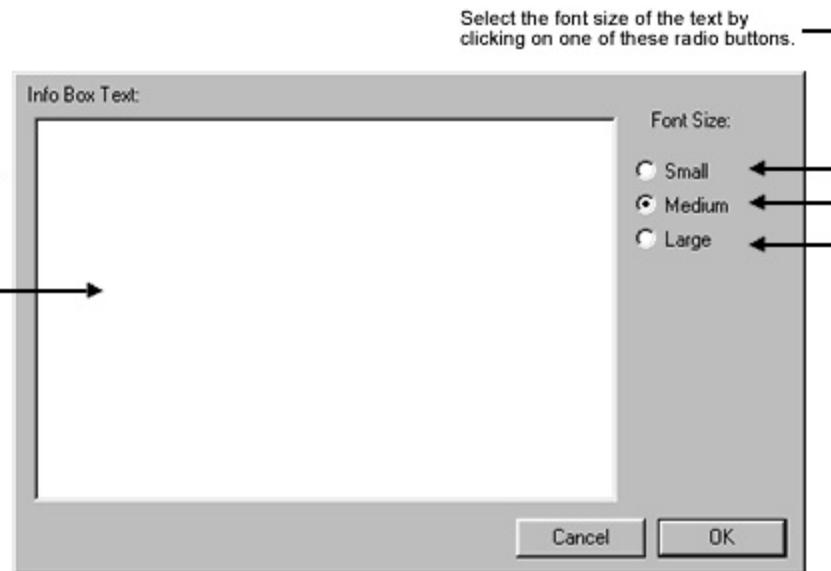
## Information button

An information button is useful whenever you want to provide the users of your model with a handy repository for instructions, background information, or summary insights. When you click on an information button, a small "windoid" pops up. The info windoid can contain text, a graphic, or even a

QuickTime movie. A close box on the windoid enables the user to put it away. Setting up an information button is a straightforward process. After creating a button, begin by entering its dialog and selecting the "Info" button purpose (it's the default, so this step is very easy!). Then, from the pop-up list to the right of the "Info" radio button, select the type of info that you want to display. The pop-up list gives you three choices: text, graphic, and movie. Depending on your choice, a button labeled "Edit text..." or "Import..." will appear next to the pop-up list. Click this button to enter/edit text, or to import a graphic or movie from disk.

*Text* - The text entry/edit dialog shown in Figure 6-12.

*Figure 6-12*  
*Info Button Dialog Box*



In the Info Box, type or paste in the text you want to pop up when the button is clicked. The font will be determined by that specified in the Model Prefs dialog for the layer on which you place the button. After you have entered your text, OK the dialogs. Then, click your button and a window, containing the text you've just entered, will pop up. You can then size and re-position the window as your requirements dictate.

*Graphic* - To import a graphic from disk, select graphic from the pop-up list, click the Import... button and navigate through the directory structure to locate the target graphic. Note that with QuickTime installed on your computer, you have access to a wide variety of graphics formats. For a list of Windows graphic file formats you can import, see [Graphic File Formats - Windows Only](#). You'll find it most efficient to place disk-based graphics in a directory entitled "graphics", placed in the same directory as your model file. Doing so makes it

easy to find the files, and also makes it easy to transport your model and the associated disk-based graphic files to other machines with different directory structures.

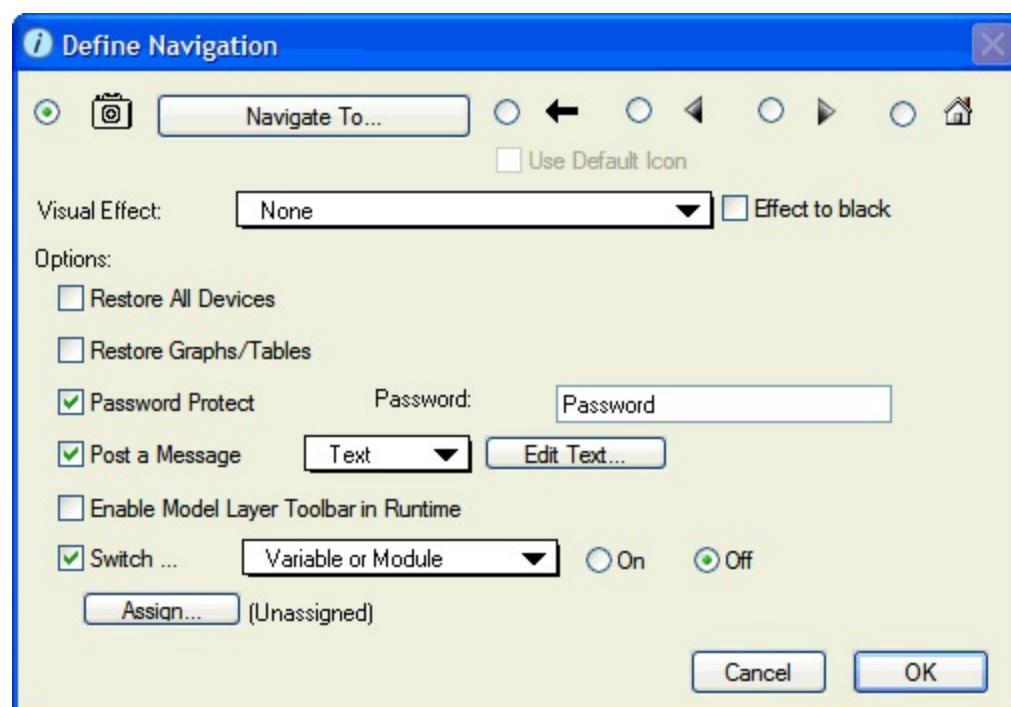
**Movie** - To import a movie from disk, select Movie from the pop-up list, click the Import... button and navigate through the directory structure to locate the target movie. It's a good idea to place all disk-based movies in a directory entitled "movies", placed in the same directory as your model file. Doing so goes far to ensure that your model and its associated movies will be easily transportable from your machine to the machines of your clients and colleagues.

## Navigation button

Navigation buttons can form an essential component of the user interface to your model. Navigation Buttons eliminate the need to scroll from one screen to the next. Button navigation is far more convenient than scrolling, and it is far friendlier to users.

To assign a navigation function to your button, click on the radio button next to Navigate in the Button Dialog (refer to Figure 6-11, above). Then click on the "Define Navigation" button. The define navigation dialog will appear as shown in Figure 6-13. Within this dialog you have three essential tasks: Setting up the navigation itself, configuring the visual effects that will go along with the navigation, and choosing among the various options that can accompany a navigation. We'll look at each task in turn.

*Figure 6-13  
Navigation Secondary Dialog*



**Selecting Navigation Type:** In the Define Navigation dialog, you have three options for navigation: Camera Navigation, Return Navigation, and Page Navigation.

- **Camera Navigation** – Camera Navigation allows you to assign a specific location in your model to the button. When the user clicks the navigation button is clicked, the user is navigated the location you specified for the button. To assign Camera Navigation to the button, select the  option. To determine the destination for the navigation button, click the **Navigate To** button. The cursor turns into a camera image. Use the scroll bars (or the page up/down keys on your keyboard) to move to the desired location on the Interface, Map, or Model layer. To change layers, click the up or down arrow on the left-hand frame of the diagram. Be sure to center the screen just the way you want it to appear. When you are satisfied with the location, click the camera icon anywhere on the screen to assign the associated coordinates to the button. You will then be returned to the main Button Dialog. Click **OK** exit the button dialog and to complete the navigation assignment.

---

**Notes:** When using the Camera Navigation option, it is important to use the scroll bars or page up/down keys to move to the new location to which you want to assign the button. Any click on the diagram itself while the camera is displayed will immediately assign the associated coordinates to the button. Also, note that there is a "secret feature" associated with camera navigation. If you depress the control (command on Macintosh) key and click within a graphics object when you create a camera navigation assignment, the subsequent camera navigation will take you to the top-left coordinate of the graphics object.

If you select the Camera Navigation option and have selected to show pages on the Interface layer (with the **Show Pages** check box in the [Interface Preferences dialog box](#)), the button will navigate users to the top-left of the Interface layer page where you set the navigation (by clicking the camera icon), rather than the exact location on the page that you selected when you assigned the navigation.

---

**Deleting or Changing camera navigation assignment:** To change a button's previously assigned camera navigation, you must first click once on the Delete Navigation button within the Define Navigation dialog, then click **Navigate To** to make the re-assignment.

- **Return Navigation** – Return Navigation retraces Camera Navigation, one

step at a time, much like the "Back" button in a Web browser. To assign Return Navigation to the button, select the option. When a user clicks the navigation button, the model will retrace Camera Navigation, one step at a time. When you select the Return Navigation option, the **Use Default Icon** check box becomes available. When you select this check box, the navigation button displays the default symbol for the action () on the button's face. If you use a standard icon, users of your model will become familiar with this symbol and the Return function associated with it. Alternately, you may wish to assign text or a different symbol to the button (for more information, see [Button Appearance](#)).

- **Page Navigation** – Page Navigation allows you to specify that the button navigates to the previous page on the current layer, the next page in the set of pages, or to the Home Page (as defined in the Page Size and Runtime Options dialog box). To assign Page Navigation to the button, select the (Previous page), (Next page), or (Home page) option. When a user clicks the navigation button, the model navigates to the page you specify here. When you select one of the Page Navigation options, the **Use Default Icon** check box becomes available. When you select this check box, the navigation button displays the default symbol for the action you selected (, , ) on the button's face. If you use a standard icon, users of your model will become familiar with this symbol and the function associated with it. Alternately, you may wish to assign text or a different symbol to the button (for more information, see [Button Appearance](#)).

---

*Note:* The Home page is always on the Interface layer. If you create a navigation button on the Model layer and choose the (Home page) option for the button, you will be navigated to the Interface layer. By default, the Home page is page 1 on the Interface layer. If the model has multiple pages, you can select a different Interface layer page as the Home page in the Page Size and Runtime Options dialog box. For more information, see [Interface Preferences](#).

**Assigning Visual Effects:** You can assign a Visual Effect to a Navigation Button to heighten interest and engage your audience. Visual effects use visual cues for smooth transition from one screen to the next.

To assign a Visual Effect, click-and-hold on the Visual Effect pop-up menu box and select from among the variety of effects. If you click the optional "Effect to black" box, the visual effect will transition to a black screen before it transitions to the new screen.

**Selecting Navigation Button Options:** The following options are available in the Define Navigation dialog box:

- **Restore All Devices** – Select this check box to restore ALL devices (graphs, tables, sliders, etc.) as navigation takes place.
- **Restore Graphs/Tables** – Select this check box (and don't select the Restore All Devices check box) to restore ONLY graphs and tables as navigation takes place.
- **Password Protect** – Select this check box to password-protect camera navigation, When you select this check box and type a password in the Password box, users of your model will be unable to navigate to a location with the navigation button you are defining unless they know the password. Note that passwords are case-sensitive.
- **Post a Message** – Select this check box to post a message to the user when he or she clicks the navigation button. Navigation button-based messages work just like information buttons. The message you post to the user can be text, a graphic, or a QuickTime movie. To post a message, select this check box, then select the type of message you want to post (**Text**, **Graphic**, or **Movie**), and then either click the **Edit Text** button to specify the text of the message, or click the **Import** button to navigate to and select the [graphic](#) or movie (.MOV)file you want to use.
- **Enable Model Layer Toolbar in Runtime** – Select this check box to display a Model layer toolbar in the Runtime version. By default, when viewing the Map or Model layer in the Runtime version, the toolbars are hidden to prevent the user from modifying the model structure. Selecting this check box reveals a subset of the tools, which allows the user to map out model structure.
- **Switch** – Select this check box to switch a model variable, module or a sector "on" or "off" in association with a camera navigation. To use this feature, select the **Switch** check box, choose **Variable or Module** or **Sector**, click the Assign button to select the variable, module or sector that you want to assign to the navigation, and then select whether you want navigation to turn the variable/module/sector **On** or **Off**. Each time navigation occurs with this button, the associated model variable/module/sector will be switched on or off accordingly. For variables, numerically "On" is equivalent to a value of 1; "Off" is equivalent to a value of 0.

## Menu button

You can associate Menu items with buttons by clicking once on the "Menu" radio button. You must then choose the desired Menu item you want performed when the end-user clicks the button. Click-and-hold on the menu

box, and choose the menu item you want from the scrollable list; when you have made your choice, release your click to assign the specific operation. Your choice appears in the bar next to "Menu Item." Figure 6-14 shows the list of possible Menu items that you can assign to a button.

*Figure 6-14  
Menu Items Available to Assign to Buttons*

<b>File Menu</b>	<b>Map/Model Menus</b>	<b>Run Menu</b>
Open	Find	Run
Close	Restore Sliders	Pause
Save	Restore Graphical Inputs	Stop
Save As	Restore Numeric Displays	Resume
Export Text Boxes	Restore Graphs/Tables	Sector Specs
Save As Image	Restore Knobs	Sensi Specs
Revert	Restore LIDs	Run Specs
Print Setup	Restore Switches	Range Specs
Print	Restore Warning Devices	
Quit/Exit	Restore All Devices	

For more information about the menus and menu items, see [Introduction to Menus](#) and the sections describing each menu in the "Menus" chapter.

In addition to the menu items listed above, you can assign the following commands to a Menu button:

- **Close Window** – The Close Window option closes the current model window.
- **Run/Restore** – The Run/Restore option will prevent the end user from beginning a new simulation until they have Reset the model (usually via a "Reset" button). This will prevent them from accidentally clicking the Run button after the completion of the run before they have had a chance to examine the graph and table output. If you use this option, you should provide a "Reset" button to set up the model for another run. A "Reset" button is a Menu button with "Restore all Devices" as the assigned function.
- **Print Exports** – The Print Export option prints a report containing any Text Boxes, Graphs, Tables and Sector Frames that have been marked for export.
- **Print Screen** – The Print Screen option prints the portion of the model that is currently shown on the computer screen.
- **Import Now** – Imports data from an Excel file to a model with a persistent import link.
- **Export Now** – Exports data from a model with a persistent export link to the corresponding Excel file.

## Tracing button

A model structure's logic can be difficult to swallow in a single bite. And yet, a comprehensive diagram of the process by necessity will show the whole enchilada. Often, one doesn't know where to start. Also, the feedback loop structure of a model can be difficult to recognize when a model consumer (or builder, for that matter) is looking at a diagram that is rich with Stocks and Flows. It is difficult to identify these loops by inspection.

Tracing can be used to sequentially unfold the logic of a particular model, working back through the inputs to a given model variable. By displaying the model structure in mind-sized bites, tracing makes it far easier to understand how the model is put together. Tracing can also be used to step through the feedback loops associated with a given model variable. In so doing, it provides a more visceral sense for the feedback processes that are at work in the model.

Using the Tracing option you can define buttons to perform Logic or Loop Tracing. Each type of tracing "walks" you through relevant structure. Loop

Tracing displays the loops that impact a particular model element. Starting at a single element, Logic Tracing unfurls structure - each time you click on a dot, you reveal the inputs to the dotted element.

---

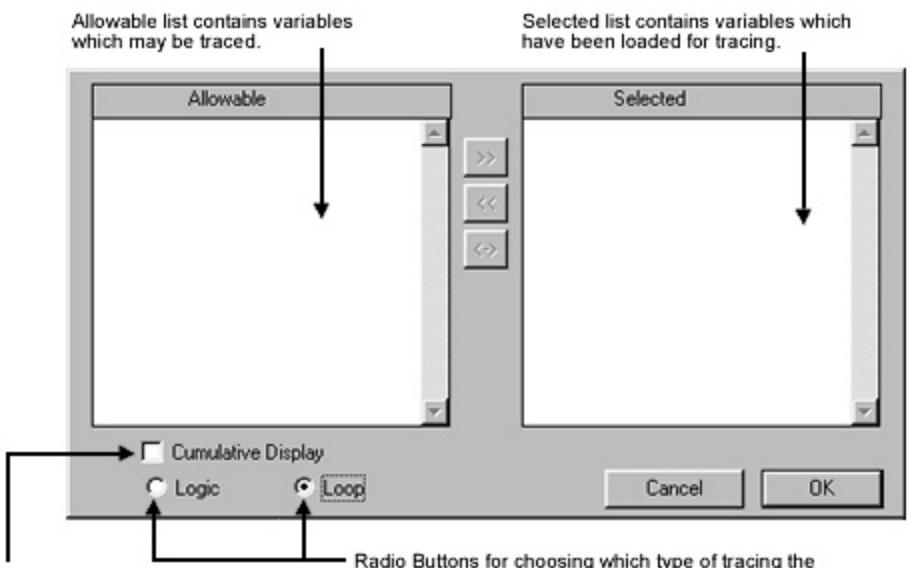
**Note:** The tracing button is accessible on the Interface layer only. You cannot create a tracing button on the Map or Model layers.

---

*Loading/Removing/Exchanging Variables:* The two lists within the dialog, shown in Figure 6-15, along with the buttons between the lists, are used to load, remove, and exchange the variables to be traced. All model variables (stocks, flows, converters) will appear in the Allowable list. Variables to be traced appear in the Selected list. When a variable has been loaded to the Selected list, it will appear gray in the Allowable list. Here's how to move variables between the two lists:

- *Loading Variables:* To load a variable to the Selected list, double-click the variable name in the Allowable list. Or, select it, and click the >> button. To select multiple variables, simply drag-select or shift-click (for adjacent variables in the list), or control-click (Windows) or command-click (Macintosh) for noncontiguous variables.
- *Removing Variables:* To remove a variable from the Selected list, double-click on the variable's name in the Selected list or select it and click on the << button between the two lists. To select multiple variables, simply drag-select, shift-click (for adjacent variables), or control-click (Windows) or command-click (Macintosh) for noncontiguous variables.
- *Exchanging Variables:* To exchange one or more variables in the Allowable list for one or more variables in the Selected list, select the desired variable(s) in each list. Then, click on the <-> button. The variables will be exchanged.

*Figure 6-15*  
*Tracing Button Dialog*



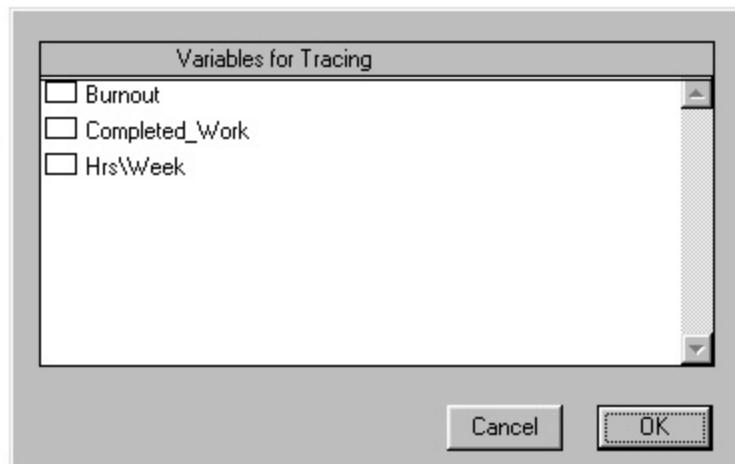
Checking this option will cause loop diagrams to be drawn cumulatively. This option only appears when the Loop radio button option is selected.

Radio Buttons for choosing which type of tracing the button will perform.

If you have loaded more than one model element into the Tracing Button you will get the dialog similar to the one shown in Figure 6-16 when you click on the button. To trace a variable, select the variable by clicking on its name and click **OK**. Or, you can simply double-click on the desired variable's name. If you have only one model element loaded into the Tracing button, the software will begin tracing the element when you click on the button.

*Figure 6-16  
Variables for Tracing Dialog*

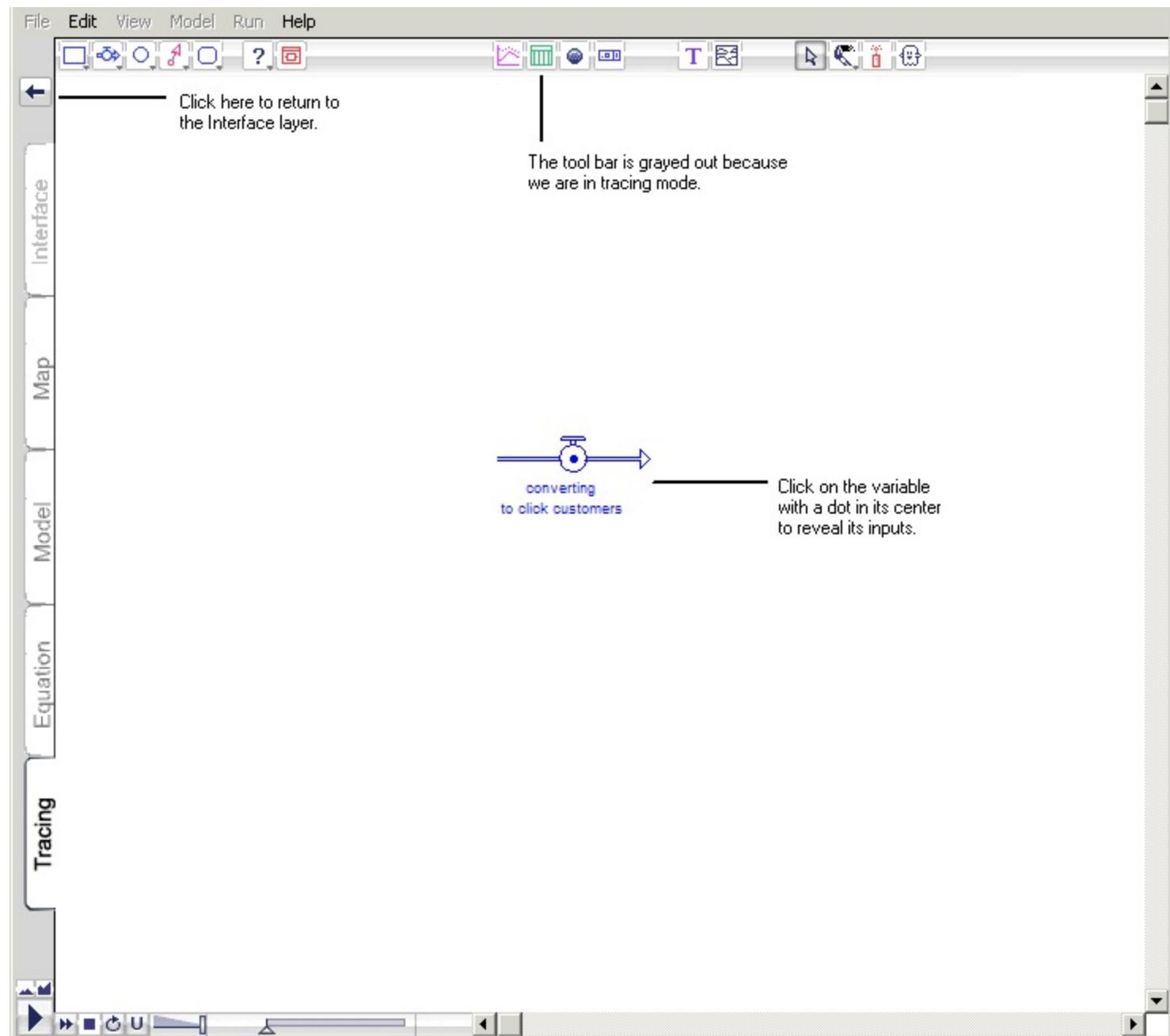
Click on one of the variables in the list to select it and click OK, or double-click the name of a variable to initiate a tracing sequence.



If the button has been set up to do Logic tracing, you will be taken to the Tracing tab, where the selected tracing variable will be displayed with a big dot in its center. Click on that dot to reveal the inputs to that variable. Continue clicking on the dotted variables to sequentially unfold model structure. When

you're done tracing, click the return arrow above the navigation tabs to return to the Interface layer. See Figure 6-17.

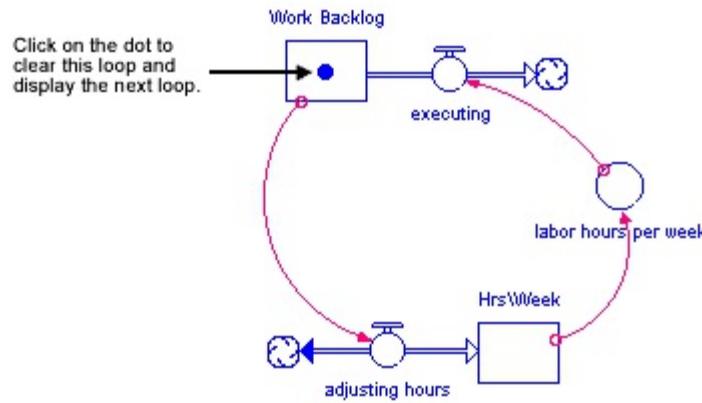
*Figure 6-17  
Clicking the Tracing Button*



If the button has been set up for Loop tracing, each feedback loop that passes through the selected variable will be displayed. After each loop is drawn, a dot will appear within the selected variable. Click on the dot to clear the current loop and display the next loop. See Figure 6-18.

If you have checked the Cumulative Display option within the Loop Button define dialog, clicking the Loop Button will cause all of the loops that pass through the stock to be variable (i.e., it will not clear existing loops before drawing the next loop that passes through the variable).

**Figure 6-18**  
**Loop Tracing**



## Storytelling button

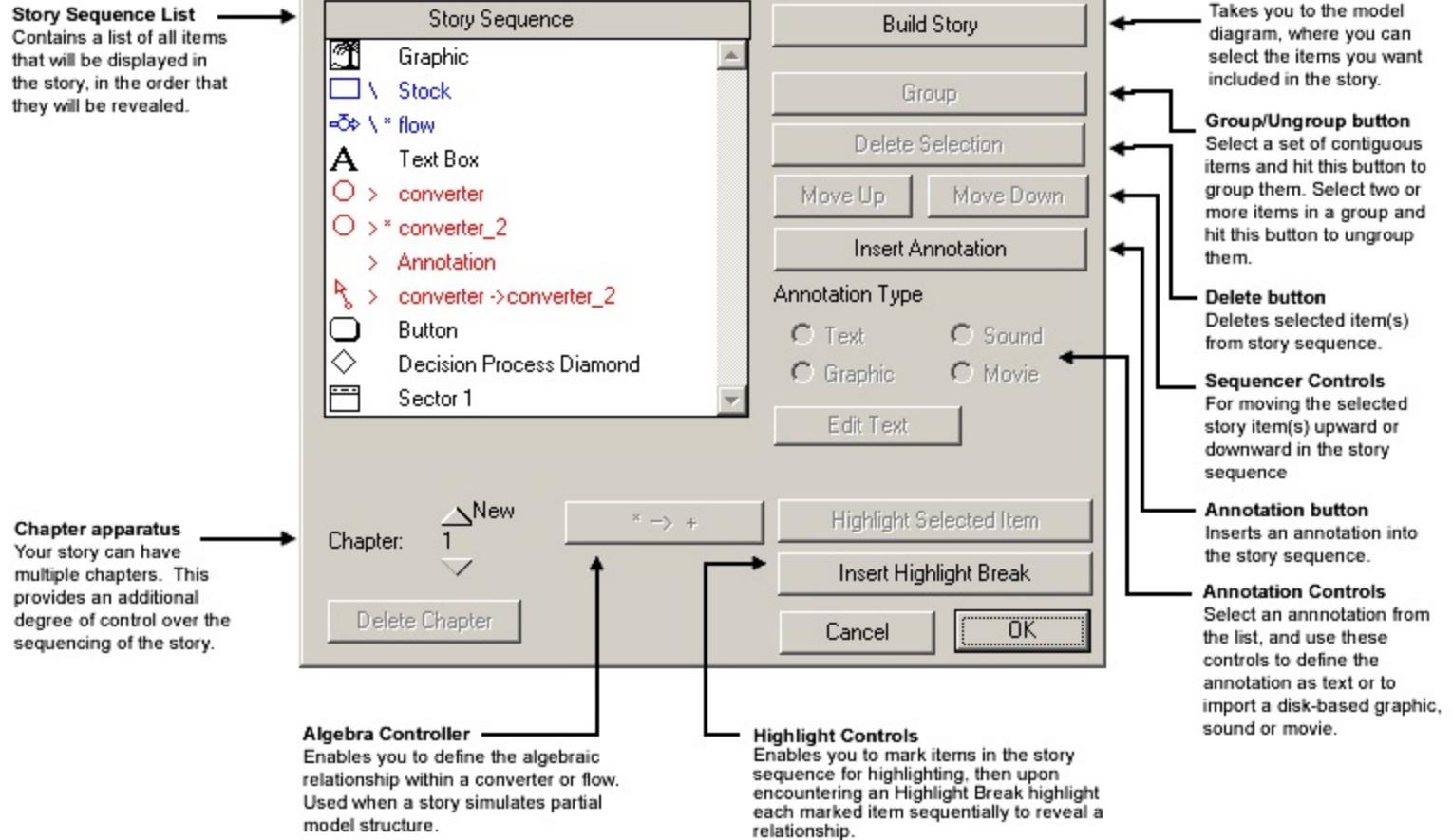
A storytelling button allows you to customize the sequential unfurling of the elements in your map. You choose both which elements are revealed, and in what order. The feature also allows you to intersperse text, graphics, sounds, or movies between each element that is being unfurled. Finally, you can simulate the structure that is showing on the diagram, in mid story. Taken as a whole, storytelling enables you to breathe life into the communication of your model and its insights.

---

**Note:** The Storytelling button is accessible from the Interface layer only. You cannot create a storytelling button on the Map or Model layers.

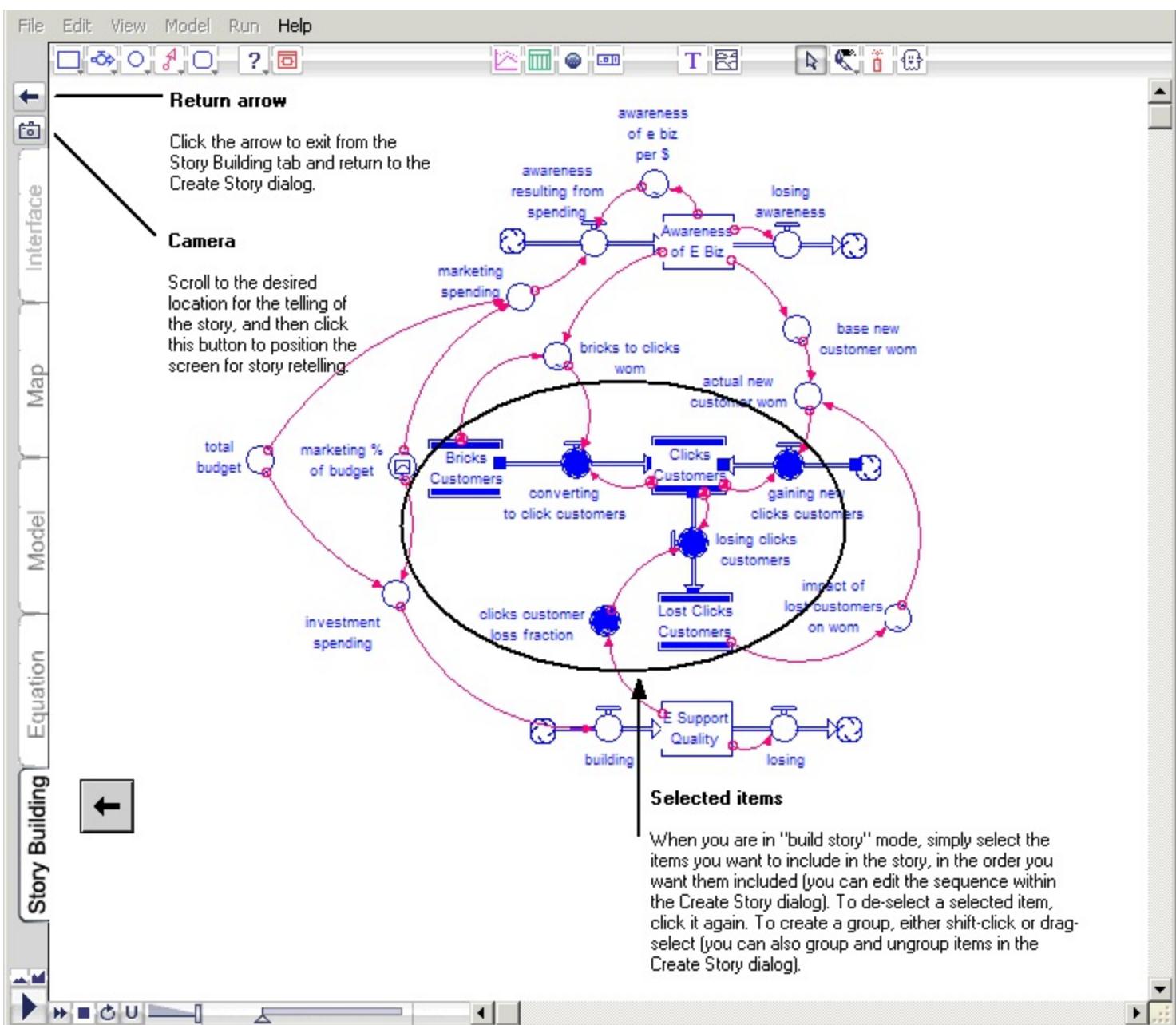
To create a storytelling button, select the "Storytelling" radio button in the Button dialog box for a button that you have created on the Interface layer. Then click the "Create Story..." button that becomes available. When you do, the Create Story dialog box appears, as shown in Figure 6-19a. The Create Story dialog is the locus for the building and editing of your story line.

**Figure 6-19a**  
**The Create Story Dialog**



**Building a story.** To build a story, the first step is to click the "Build Story" button within the Create Story dialog. When you do, you'll be taken to the Story Building tab, which displays your model as you see it on the Model layer. There, you can click on diagram elements to add them to the story sequence. The initial sequence will be determined by the order in which you select diagram elements. You can subsequently modify the sequence from within the Create Story dialog. As you build the story, you can group items by depressing the shift key as you click on diagram elements, or you can drag select items. To remove an item from the story simply click it again to de-select it. Figure 6-19b, below, illustrates the process of building a story.

**Figure 6-19b**  
*Building a Story on the Story Building Tab*



Refining and testing a story. Once you have built the initial story sequence on the diagram, you'll probably want to refine it. Here, you'll work within the Create Story dialog to group items, to change the sequencing, and to add annotations. Figure 6-19a, above, provides a description of the controls available to you.

An essential component of this refining process is to test the story. To do so, simply OK your way out of the Storytelling and Button dialogs, and then use the Arrow to click on the storytelling button you created. You'll be taken into storytelling mode on the Story Telling tab. There, each time you tap on the spacebar you will see the next step in the story sequence (this process analogous to "custom animation" within Microsoft PowerPoint). Highlighting will indicate the new structure in each step of the story sequence.

To "back up" in the story sequence, tap the backspace key (On the Macintosh, tap the delete key).

As you test the story, re-size and re-position annotation as desired. The software will retain the positions and sizes you set. To end a storytelling sequence, click the return arrow above the Navigation tabs to return to the Interface layer. If your model has a navigation button already created, you can include it in your story to take the user back to the Interface layer.

Of special note in the context of refining and testing your story is the camera icon shown in Figure 6-19b. The camera icon is something you can access when you visit the diagram in "build story" mode. It provides you with a mechanism for defining the diagram space in which the story will be told. When you click the camera, the software will use the associated screen coordinates as the initial location for the display of the story. In refining your story, you will want all the main story elements to be showing on your screen when you click the camera. If the story contains elements that are "off screen" from this location, you will need to include a navigation button in your storytelling sequence, and instruct the end-user to click it.

Running your model while in Storytelling mode. One of the truly amazing features associated with storytelling is the ability to simulate portions of the model while you are in the midst of a storytelling sequence. Because menus and toolbars are unavailable within a story sequence, you'll need to incorporate a run button in the story sequence. (You also might want to incorporate a pinned graph or table, or perhaps a numeric display or two. Doing so enables the end user to see clearly the results of the simulation.) The sample models that accompany the software make extensive use of this "run while in storytelling" feature. We encourage you to take a look at these models to see what's possible.

The software's simulation algorithms are modified somewhat when you are running portions of the model in the midst of a story. The approach is to evaluate all diagram entities that are displayed on the diagram at the time the simulation is run. Those entities that are not showing (either because they have not been included in the story sequence or because they have not yet been unfurled in the story) are neutralized in the simulation. The table below summarizes the rules used by the in evaluating equations within a storytelling sequence.

*Figure 6-19c  
Building a Story on the Model Layer*

Type of equation logic within the entity	How the entity is displayed within the story	What happens
		A constant value is returned.

Constant in converter or flow	It is showing	The user can change the value by hovering over the entity, depositing the cursor in the hover value box, typing a new value, and hitting the enter key on the keyboard.
Algebraic relationship in flow	No inputs to the flow are displayed	A constant value equal to 10% of the attached stock is returned. The user can change the value by hovering over the flow regulator, depositing the cursor in the hover value box, typing a new value, and hitting the enter key on the keyboard.
Algebraic relationship in converter or flow	All inputs to the converter or flow are displayed	The full equation is evaluated using all input values. The user cannot directly change the value returned by the flow or converter.
Algebraic relationship in converter or flow	Some but not all of the inputs to the converter or flow are displayed	The equation is evaluated using the inputs that are showing on the diagram. The software attempts to neutralize the inputs that are not showing. If "+" is displayed to the left of the entity in the story sequence, all non-displayed inputs to the entity are given a value of 0 in the evaluation of the entity's equation logic. If "*" is showing next to the entity in the story sequence, all non-displayed inputs to the entity are given a value of 1 in the evaluation of the entity's equation logic. Use the algebra controller button

within the storytelling dialog to change the evaluation of non-displayed inputs.

## **Hyperlink button**

A hyperlink button allows you to link to a web page or a file from within your model. To assign a link to a button, click on the Hyperlink radio button in the Button dialog. Then click on the "Define Link..." button. The Define Hyperlink dialog will appear. Within this dialog, type or browse to the location of the link and click on the File or URL radio button depending on the type of link you are creating. It is a good idea to place all linked files in a directory entitled "Links", placed in the same directory as your model file. Doing so will help to ensure that your model and its associated links will be easily transportable from your machine to the machines of your clients and colleagues.

## **Play Movie button**

You can assign a movie to a button by clicking once on the Play Movie radio button, then on the "Assign Movie" button. You will view a standard Get File dialog; select the file containing your movie. To complete the assignment, click OK. Now when the end user clicks the button, the movie assigned will play. QuickTime must be installed in your system to play movies in the software.

When you select the Play Movie radio button, an option called "Iris Visual Effect" will appear below in the dialog. Click in the box next to "Iris Visual Effect" to create a visual effect transition to your movie.

## Button Sound

The Button Purpose choices described are mutually exclusive. Sound, however, can be used in conjunction with any of the six button purposes. Though the default sound assigned to a button is a simple clicking sound, you can opt to import a sound you've recorded (music, audio text, sound effect) or use one from an audio library. Or, you can choose to have no sound associated with your button by unchecking the "Use clicking sound" box.

To assign a sound other than the clicking sound, click on the "Import Sound" button. You will be prompted by a standard "Get" dialog box to find and select your file. After you select the file, the name of the sound file will appear below the button (which will read "Delete Sound").

*Deleting sounds from a button:* To delete or reassign a button's sound, click on the "Delete Sound" button to remove the current sound file. At this point, you may reassign a new sound file.

# Button Appearance

There are three major attributes that dictate the appearance of your button: the opacity, the shape and the image (if any) assigned to the button's surface. The features described in this section are located in the lower half of the dialog box, described in [Figure 6-11](#).

*Opaque/Transparent:* Select whether you want users to be able to see the button. To make the button visible, select the **Opaque** option. By default, an opaque button appears as a blank gray button, but you can change it by importing a picture or adding text to the button (see below). To make the button invisible, select the **Transparent** option. When you select this option, the button appears on the diagram with a faint, dotted outline. When this outline is displayed, you can select the button, double-click its border to access the Button dialog, and otherwise manipulate the button as usual.

While opaque buttons usually look like buttons that will perform an action, you can format a transparent button so that it looks like an image, plain text, or a hyperlink that navigates the user to another area in the model or that provide more information. A transparent button can perform any function that an opaque button can, but it doesn't have to look like a button.

To make the border of a transparent button invisible, go to the Interface menu (on the Interface layer) or to the Model menu (on the Map or Model layer), choose "Hide", and then choose "Transparent Buttons". Your button will disappear but will remain on top of the graphic or text over which you've placed it. When the end-user clicks it, it will perform the operation you defined, but, to the user it will appear that they clicked on the visible object. To return the dotted outline of transparent buttons (to access the dialog box), select "Show" from the Map or Model menu and choose "Transparent Buttons."

*Shape:* Select whether you want the button to have rounded corners or squared (right-angled) corners.

*Picture:* If you have created an opaque button, you can place a picture on the button by choosing either the **From File** option or the **From Clipboard** option, and then clicking the **Import Picture** button. If you selected **From File**, the Open Image File dialog box opens so that you can select the image file you want to use for the button. If you selected **From Clipboard**, clicking the **Import Picture** button imports the current image on the clipboard onto the button.

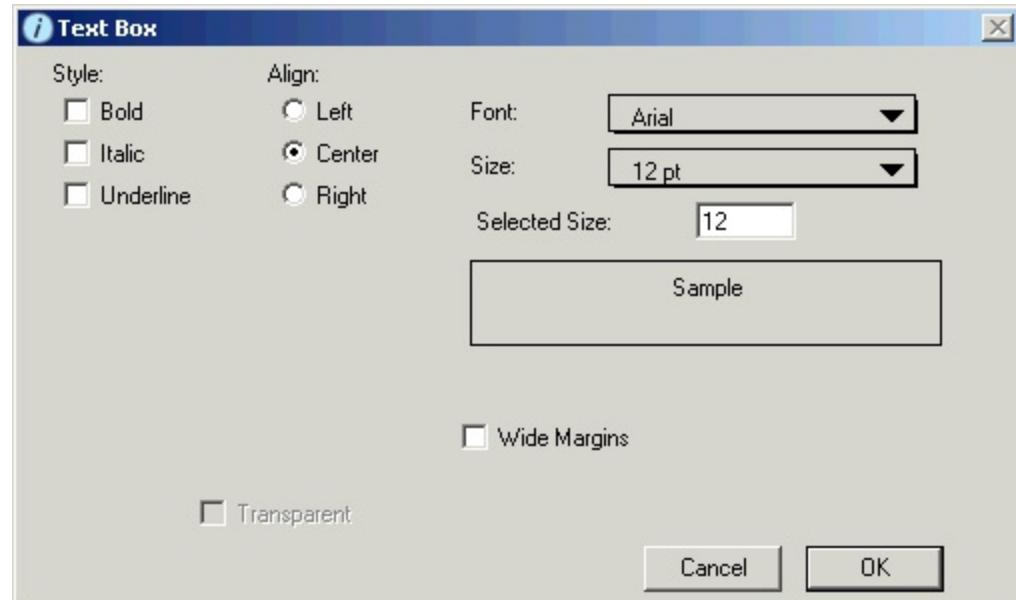
---

*Note:* For Macs, you can import PICT files; for Windows machines, see [Graphic File Formats - Windows Only](#) for list of valid Windows image formats.

*Button Title:* For either type of button, you can have text appear on the

button. To specify text, select the **Use Button Title** option, type the text you want in the box to the right of the **Import Picture** button, and then click the **Import Picture** button. The Text Box dialog box appears, as shown in Figure 6-19.

Figure 6-19  
"Format Button Title" Dialog Box



Use the Text Box dialog box to format the text that will appear on the button:

- **Style** – Use the check boxes in this section to specify the text style on the button: **Bold**, **Italic**, or **Underline**.

*Tip:* To make the text on a transparent button look like a hyperlink, select the **Underline** check box.

- **Align** – Use the options in this section to select how the text should be aligned on the button: **Left**, **Center**, or **Right**.
- **Font** – Click this box to select the font type for the text.
- **Size** – Click this box to select the font size (in points) for the text. The size you select is displayed here and in the Selected Size box.
- **Selected Size** – Displays the currently selected font size for the text (in points). You can change the size by typing a font size in this box.
- **Sample** – Displays sample text that shows your current text settings (style, alignment, font type, and size).
- **Wide Margins** – Select this check box to provide extra margin space between the text and the side borders of the button.

When you are finished selecting text options, click **OK** to return to the Button dialog box ([Figure 6-11](#)). Notice the "\*" that appears next to the **Import Picture** button. This indicates that a title/picture is assigned to the button.

**Delete Picture:** To delete a previously assigned picture (picture file or text), click the **Delete Picture** button; you can now assign a new picture.

**Size To Button:** Select the **Size To Button** check box to scale the imported picture so that it fits within the button.

 [Related Topics](#)

# Switch

As its name suggests, the switch enables you to turn things on or off in a model. Specifically, you can assign a switch to one or more converters in your model, to a sector or a module in your model, or to the sensitivity analysis setup dialog. The switch thus enables you as a model builder to create a friendly device for users to control a model.

When a switch is assigned to a converter, the converter will take on the value of 1 when the switch is on, and 0 when the switch is off. When a switch is assigned to a Sector, Run by sectors for the sector will be turned on when the switch is turned on. When a switch is assigned to a Module, Run by modules for the module will be turned on when the switch is turned on. Finally, when the switch is assigned to do sensitivity analysis, the current sensi setup will be activated when the switch is turned on. The model will run in normal single-run mode when the switch is turned off.

Whenever you assign a switch to a converter, that converter will be unavailable to connect to other input devices. Switches may only be associated with converters defined as constants. You cannot assign a switch to a stock, a flow or a converter containing equation logic. Finally, a special type of switch, the "chained switch", makes it possible for you to turn "on" one variable while simultaneously turning "off" all other variables in the chain.

This section includes the following sections:

[Switch Surface Operations](#)

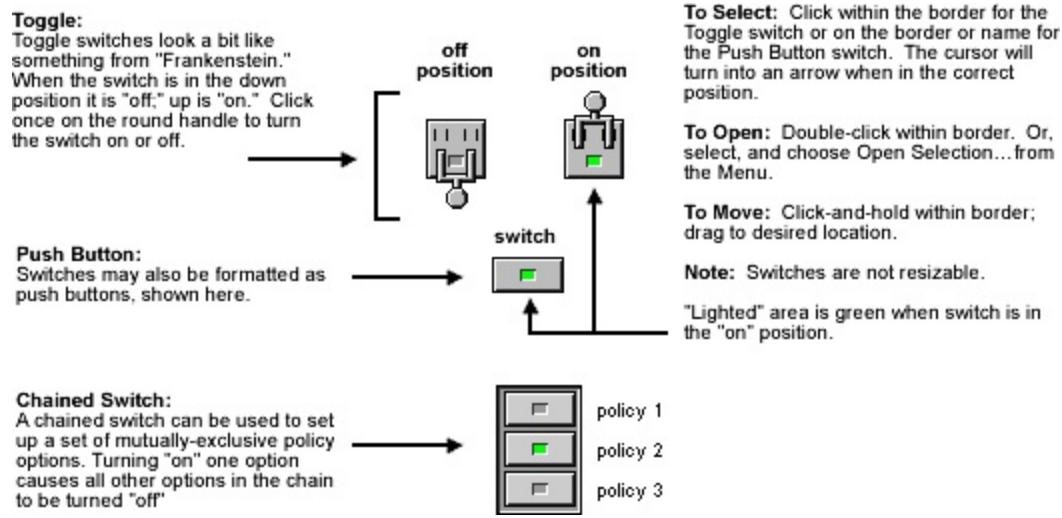
[Switch Dialog Operations](#)

 [Related Topics](#)

# Switch Surface Operations

Click once to select the Switch from the Tool Bar at the Interface layer; click once to place the switch on the screen in the spot you choose. Figure 6-20 outlines the surface operations of the Switch.

*Figure 6-20 Switch Types - Toggle and Push Button*

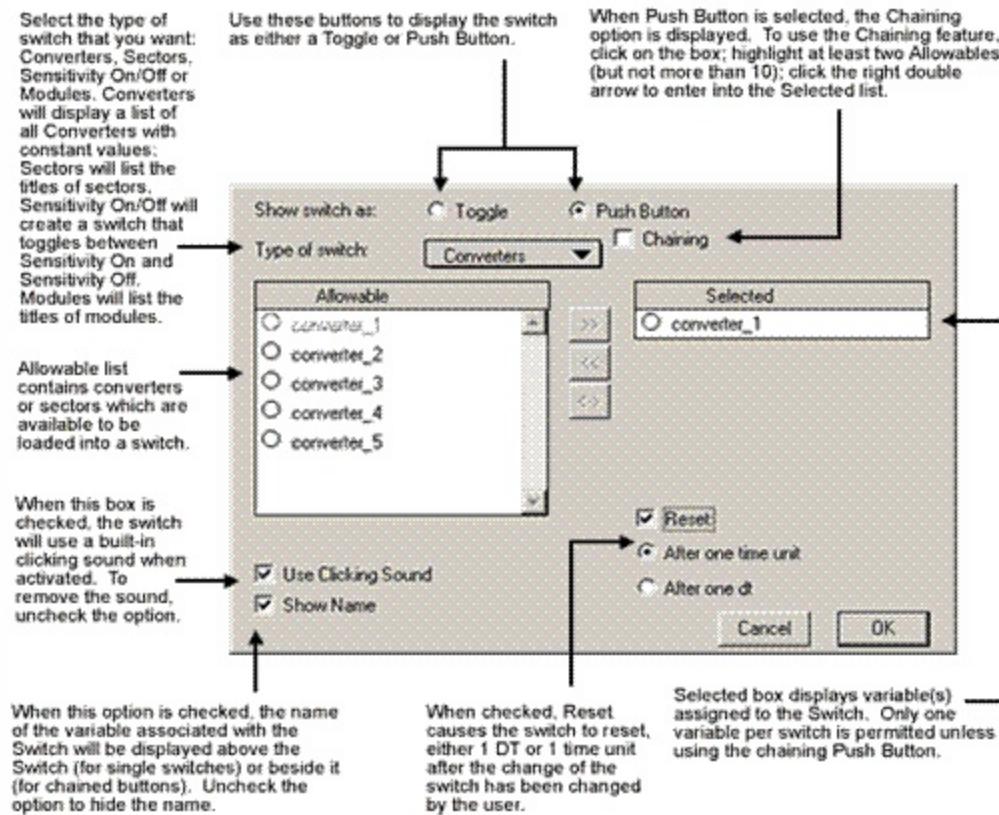


[Related Topics](#)

# Switch Dialog Operations

When you open the Switch, you will gain access to the Switch define dialog, illustrated in Figure 6-21 below.

Figure 6-21 Switch Dialog Box



**Show Switch as Toggle vs. Push Button:** You can choose to show the switch as either a toggle switch or as a push button switch. A decision to use one type over the other is based mostly on aesthetics—whether your audience will appreciate the simple Push Button design, or the more humorous "Frankenstein-esque" Toggle. If you want a single switch to link two or more on/off variables so that only one is "on" at any given time, select the Push Button option and then check the Chaining box (discussed later in this section). The Chaining feature is not available with the Toggle.

**Assign Variable:** To move a variable from the Allowable list to the Selected list, either double-click the variable name or select it and click the >> button. This will automatically replace a previously selected variable. Note that whenever a converter is loaded to the selected list, it becomes grayed out in the allowable list, indicating that it cannot be assigned to any other input device.

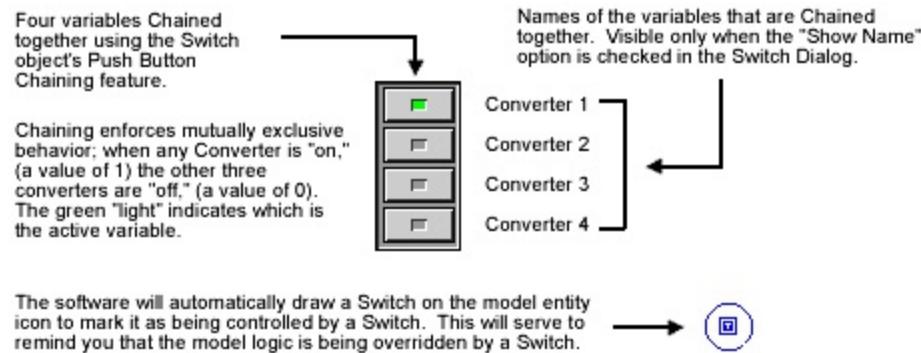
Switches have a built-in clicking sound when turned "on"; to remove the

sound, click once on the check box.

You have the option of displaying the associated variable's name. Click once on the "Show Name" box to have the name appear on the screen. When you have assigned the variable(s) to the Switch, click OK.

**Chained Switches:** When you have more than one decision variable (but not more than 10) that are mutually exclusive in their on/off behavior, set up a chained switch. Within the Switch dialog, select Push Button, and then check the "Chaining" check box. Then, move the variables you wish to chain to the Selected list (double-click or highlight and click on the >> button). Click OK. Figure 6-22 shows the appearance of an assigned Chained Switch.

*Figure 6-22  
Chained Switch*



[Related Topics](#)

# Graphical Input Device

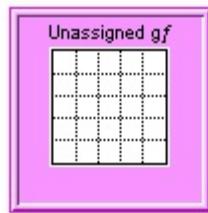
The Graphical Input Device enables model users to: (1) see, at a glance, the shape of a graphical function, (2) edit a graphical function from the Interface level; (3) restore a graphical function to its author defined relationships; and (4) animate graphical functions during a simulation.

---

**Note:** You are allowed only one Graphical Input Device per graphical function; once you've assigned a graphical function to a Graphical Input device, you can't assign it to a second one.

**Basic Operations:** Click once to select a Graphical Input Device from the toolbar on the Interface layer. Click once again to deposit it. An unassigned Graphical Input Device looks like the one shown in Figure 6-23.

*Figure 6-23 Unassigned Graphical Input Device*

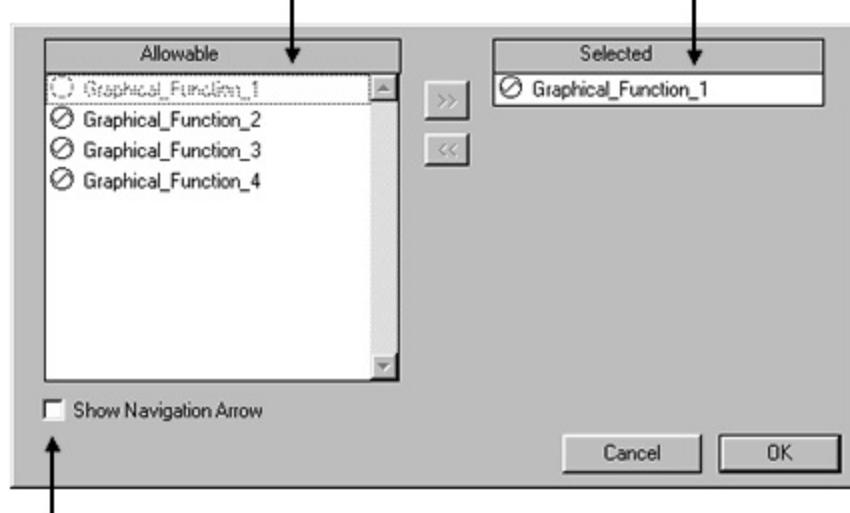


Double-click on the face of the device to enter its Define dialog, shown in Figure 6-24.

*Figure 6-24  
Graphical Input Device Define Dialog*

Allowable list contains a list of all graphical functions within a model.

Selected list contains the name of the graphical function that has been entered into the Graphical Input Device.



When this option is checked, a navigation arrow will be displayed on the Graphical Input Device.

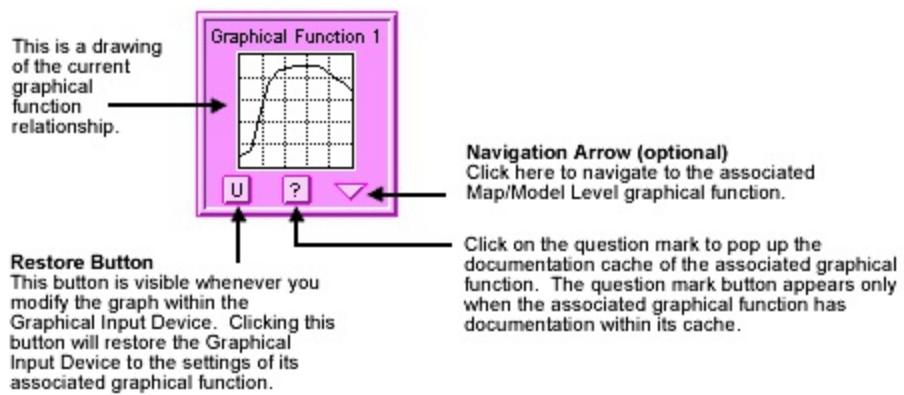
The dialog will contain a scrollable list of all graphical functions in the model. Double-click on any graphical function in the Allowable list to move it into the Selected Box. Alternately, select and click >> button to make the transfer from Allowable to Selected. A click on the << button will move the variable out of the Selected box.

If you select an arrayed graphical function, what happens next depends on whether or not Apply to All has been selected in the graphical function. If you select an arrayed graphical function where Apply to All has been turned off, you will be given the opportunity to select the element of the graphical function that the Graphical Input Device will control. If you select an arrayed graphical function where Apply to All has been turned on, the resulting graphical input device will control all elements of the graphical function.

Once a graphical function has been associated with the Graphical Input Device, a click on OK will give you what you see in Figure 6-25.

*Figure 6-25  
Defined Graphical Input Device*

Double-click anywhere within the Graphical Input Device housing to open the dialog of the associated graphical function. However, any editing done within this dialog will not modify the actual graphical function relationship.



If you double-click a Graphical Input Device that has a graphical function associated with it, the dialog associated with that graphical function will open. You'll then have the ability to edit the graphical function. However, you are not really editing the existing graphical function relationship. That relationship is being preserved! When you OK the dialog, the Graphical Input Device will display a Restore button as illustrated in Figure 6-25.

A click on the Restore button will cause the graphical function that exists in the diagram-level variable (the one that was being preserved while you did your editing) to be restored. If you wish to edit the actual graphical function relationship, you must do so by entering the graphical function dialog on the model diagram. This feature enables end-users to preserve the original relationships for subsequent restoration, while exploring alternative relationships during a scenario analysis.

---

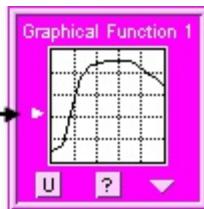
**Note:** Editing and Restoring of the Graphical Input Device can occur only when the model simulation is stopped. When the model is running or paused, the Graphical Input Device cannot be opened, and its Restore button will be inactive.

---

**Animating the Graphical Input Device:** Once you've associated a graphical function with a Graphical Input Device, when you run the model, you will see the position of the "Y" value of the graphical function (as it varies over the course of the simulation) indicated by an arrow-pointer. An example of what you'll see is shown in Figure 6-26.

*Figure 6-26  
Animated Graphical Input Device*

The arrow-pointer slides up and down the "Y" axis indicating the value taken on by the dependent variable in the graphical function relationship.



The animation capability makes the Graphical Input Device a hybrid input and output device. Not only is animation useful for end-users who are trying to gain an understanding of what's going on during a simulation, it is also helpful when you are de-bugging your models. It enables you to see at a glance which of the graphical function relationships in your model are operating "within bounds," and which are slamming into and out of bounds "instantly."

**Re-assigning a Graphical Input Device:** To re-assign a Graphical Input Device, double-click on the Graphical Input device you wish to re-assign. When you do, the associated graphical function dialog will appear. In the resulting graphical function dialog, click the Delete Graph button. Doing so will have no impact on the original graphical function associated with the diagram-level variable. It simply will disassociate the Graphical Input Device from the particular graphical function to which it is tied. You then are free to re-assign the Graphical Input Device to any other graphical function that has not yet been assigned (or if you no longer want the device, you can Dynamite it).

# Knob

The Knob is useful principally for providing initial values for Stocks. Knobs also can be used to adjust values for constants. Unlike the Slider, the Knob cannot be adjusted during the course of a simulation. It is set prior to the outset of a simulation and remains fixed throughout the simulation. Variables which contain equations or graphical functions can not be assigned to a Knob.

Knobs work well for selecting one of multiple scenarios available to the model user. The model user may "dial up" a scenario number at the start of the simulation. You may only assign one variable per Knob.

This section includes the following sections:

[Knob Surface Operations](#)

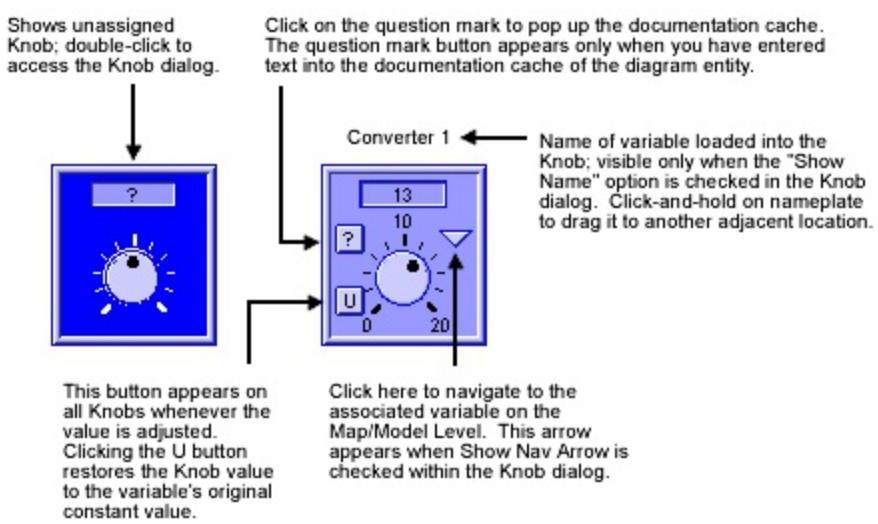
[Knob Dialog Operations](#)

 [Related Topics](#)

# Knob Surface Operations

On the Interface layer palette, click once on the Knob icon; then, click once to place it on the screen in the desired location. Figure 6-27 shows a picture of an unassigned Knob.

Figure 6-27 Knob Surface Operations



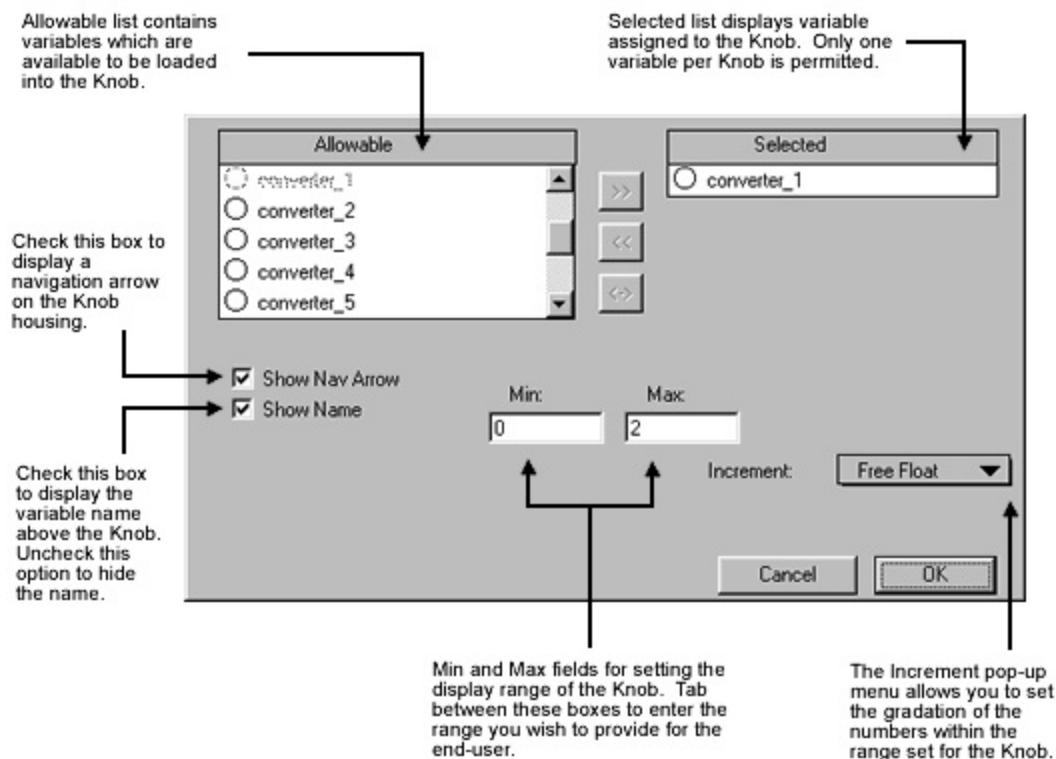
 [Related Topics](#)

# Knob Dialog Operations

To access the Knob Dialog box shown in Figure 6-28, double-click on the Knob icon. Variables which are eligible for assignment to the Knob appear in the Allowable list (items which appear to be "grayed out" have already been assigned to another input device or have non-constant values). Double-click on the variable name in the allowable column you wish to assign to the Knob or select the item and click on the >> button to move it to the Selected list.

Once a variable has been selected, Min and Max fields will appear. You can set the display range for the Knob by entering values in these fields. The Increment pop-up menu allows you to set the increments between values that can be selected on the Knob. Click-and-hold on the down-arrow to view the Increment options.

Figure 6-28 Knob Dialog Box



Other display options for the Knob include Show Nav Arrow and Show Name. Showing the navigation arrow will put a small arrow on the Knob which, when clicked, will navigate a user to the associated variable on the Model layer. If Show Name is checked, the variable name will be displayed. Figure 6-27 shows an example of an assigned Knob with name and navigation button both displayed.

When a user changes the value of a variable associated with a Knob by moving the dial, a small "U" button appears in the lower left corner. To restore the

Knob to its original value, click once on the "U" button (to undo).

---

**Note:** When a variable has been assigned to a Knob, a Knob icon appears on that variable at the Model layer. This makes the author and model users aware that the variable's value is being set at the Interface layer, and not in the entity's dialog box.

---

 [Related Topics](#)

# List Input Device

The List Input Device (LID) is a simple spreadsheet-like input device. Users of your models can use the LID to set the values for converters and flows in the model. In addition, users can use the LID to set the initial values for stocks in the model. Multiple model variables can be assigned to a single LID. A single LID can contain multiple pages.

This section includes the following sections:

[List Input Device Surface Operations](#)

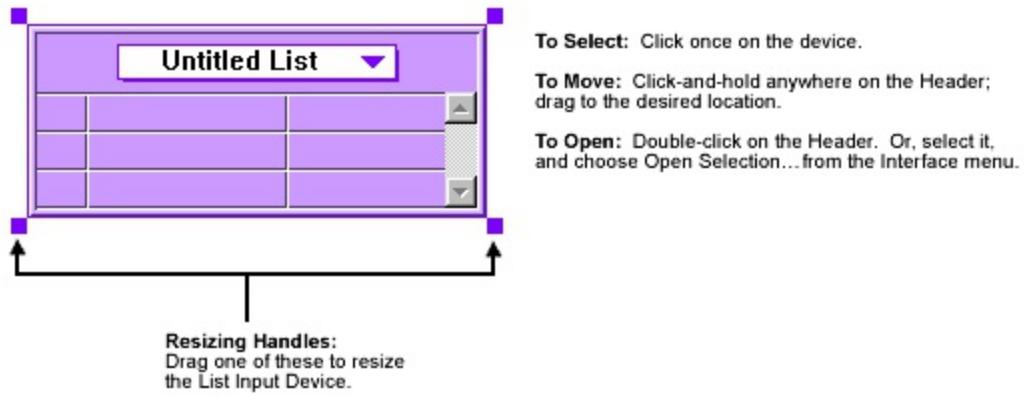
[List Input Device Dialog Operations](#)

 [Related Topics](#)

# LID Surface Operations

To create a LID, click once on the LID icon on the Tool Bar; then click once to deposit it on the screen on the Interface layer. An unassigned LID looks like the one shown in Figure 6-29.

Figure 6-29 Unassigned List Input Device (LID)



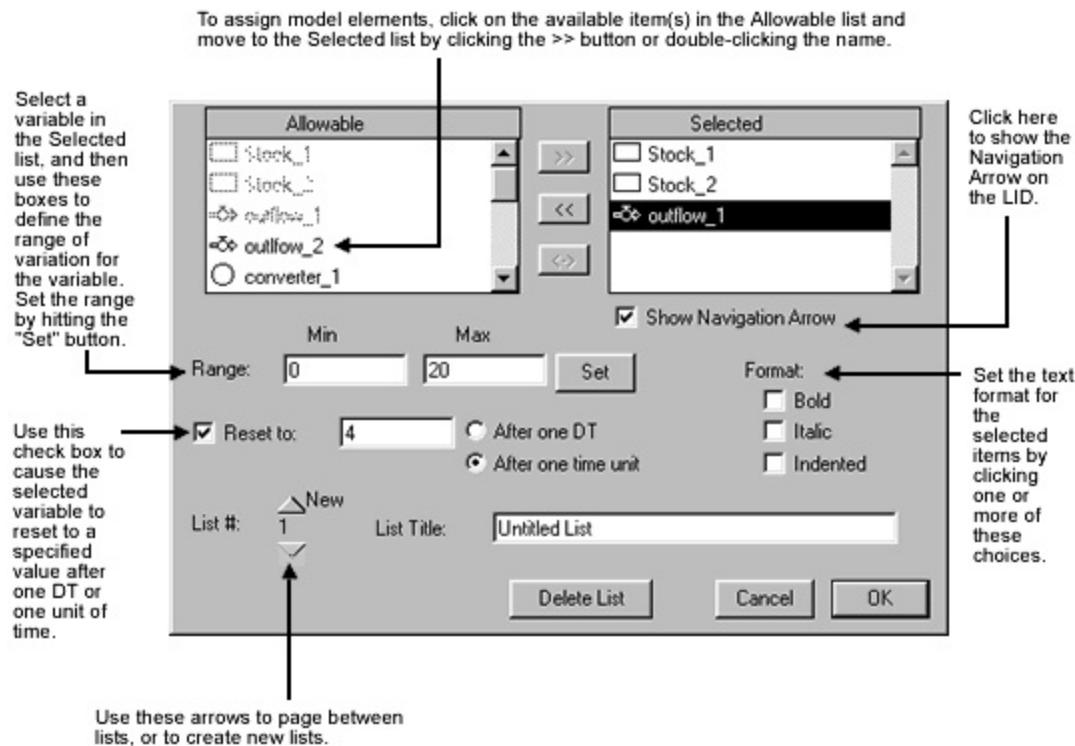
You can resize the LID by selecting it and then dragging one of the Resizing Handles. Note, however, that the scrolling capabilities of the LID mean that you can load multiple variables to a small LID. Users of your model can scroll through the device to view and modify any variable that has been loaded to the LID.

 [Related Topics](#)

# LID Dialog Operations

To enter the LID's Define dialog, use the Arrow to double-click anywhere within the confines of the LID. Or, select the LID and choose Open Selection... from the Interface Menu. The LID Define dialog is shown in Figure 6-30. Define dialog operations are described in the text which follows.

Figure 6-30 List Input Device (LID) Define Dialog



**Loading/Removing/Exchanging Variables:** The Allowable and Selected lists, along with the buttons between the lists, are used to load, remove, and exchange the variables to be displayed in the LID page. All model variables (stocks, flows, converters) will appear in the Allowable list. Variables that you have loaded to the LID will appear in the Selected list. When a variable has been loaded to the Selected list, it will appear gray in the Allowable list. Here's how to move variables between the two lists:

- **Loading Variables:** To load a variable to the Selected list, double-click the variable name in the Allowable list. Or, select it, and click the >> button. To select multiple variables, simply drag-select or shift-click (for adjacent variables in the list), or control-click (Windows) or command-click (Macintosh) for noncontiguous variables.

---

**Tips:** If a variable is selected within the Selected list, any variables that you load to the selected list will be placed above the selected variable.

You can put as many variables as you want into a given list.

---

- **Removing Variables:** To remove a variable from the Selected list, select it. Then, click on the << button between the two lists. To select multiple variables, simply drag-select or shift-click for adjacent variables, or control-click (Windows) or command-click (Macintosh) for noncontiguous variables.
- **Exchanging Variables:** To exchange one or more variables in the Allowable list for one or more variables in the Selected list, select the desired variable(s) in each list. Then, click on the <-> button. The variables will be exchanged.

**Setting the Range for LID Variables:** Within the LID Define dialog, you should set the input range for any variable that has been loaded into the Selected . The input range will define the bounds of user input. The user will not be able to assign a value to the variable that is outside of the range that you have set. Setting the range is easy.

- *Select variable in the selected list:* Step 1 is to select the variable whose range you want to set from within the Selected list. When you do, the minimum and maximum values will appear in the Min and Max boxes next to the word "Range" in the dialog.
- *Set the Range:* In the Min and Max boxes, type the desired minimum and maximum values for the range of the variable. After entering these values, click the Set button. You're set!

**Reset To check box:** The Reset To check box is available when you have selected either a converter or flow in the selected list. As illustrated in Figure 6-30, when Reset to is checked, you can configure the selected variable to reset to a pre-determined value, either after one DT or after one time unit. Reset to is a great way to prevent users of your models from inadvertently repeating a decision (such as buying a factory) that is best thought of as a one-time event.

**Show Navigation Arrow:** When this is checked in the LID dialog, a navigation "down" arrow will appear on the LID surface, enabling the user to navigate from the LID to the corresponding model variable.

**Formatting Options:** As shown in Figure 6-30, the LID Pad Dialog provides you with three simple formatting options. Simply select a variable in the selected list, and check the desired options.

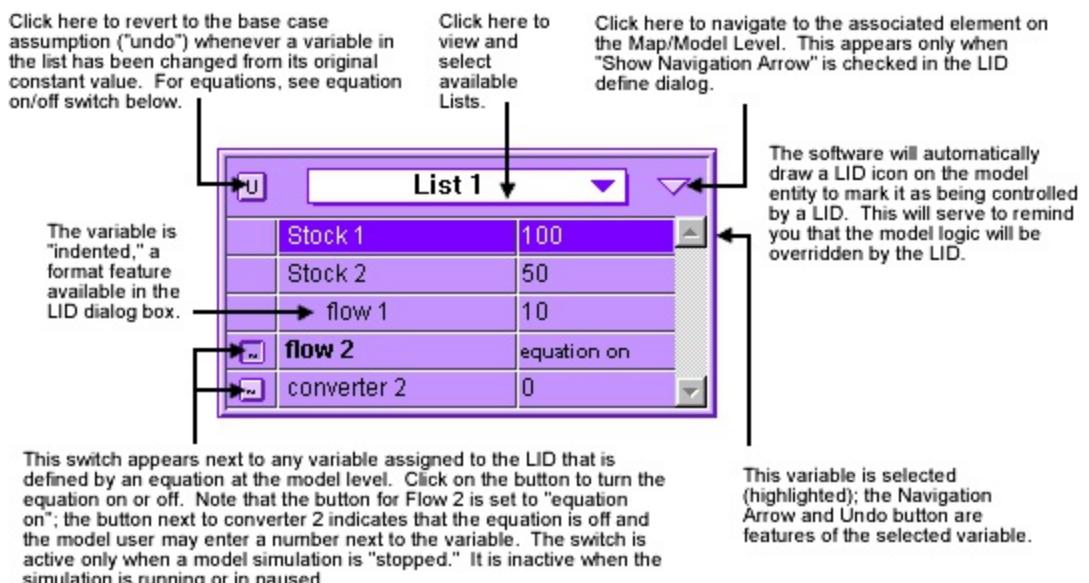
**Adding Pages:** At the bottom-left of the LID dialog is the apparatus for adding pages to the LID. Once you have loaded variables into the Selected list of the

first page of the LID, the up-arrow in the apparatus will become active. Click on the arrow (next to the word "New") to create a new page. You can add as many pages to a Pad as you'd like. You can click on the up- or down-arrows to move between pages from within the dialog.

**Titling a Page:** To title a specific page of the LID, select the text in the Title field of the LID dialog. Then, type the desired title. The title will appear at the top of the LID.

Once you have configured a LID and hit OK, the LID will look something like what's shown in Figure 6-31:

*Figure 6-31  
Assigned List Input Device*



[Related Topics](#)

# The Slider Input Device

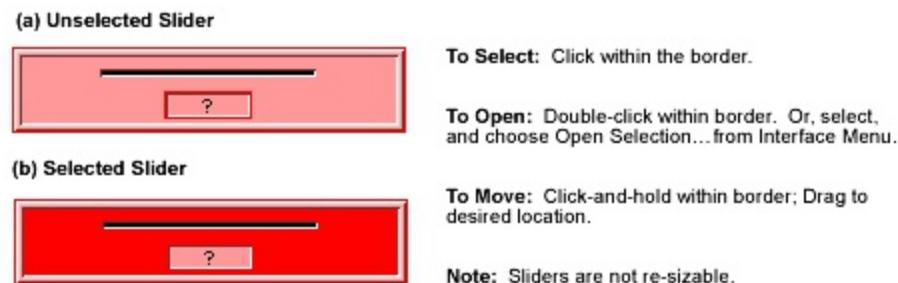
The Slider Input Device allows model users to adjust constant values, and to override equation logic (and graphical function relationships) with numerical inputs.

*Important to note:*

- You are allowed only one Slider per variable; if you've assigned a variable to a Slider, it won't be available to assign to another Slider.
- If a Slider is associated with a graphical function, that graphical function will not be available for association with a Graphical Input Device (and vice versa).
- Sliders cannot be associated with: Reservoirs, Conveyors, Queues, Ovens, Conveyor outflows, Queue outflows, Oven outflows, Sub-model icons, Sub-model roll-up Flows, and roll-down Flows within Sub-models.

**Basic Operations:** Click once to select the Slider from the Interface Layer Tool Bar. Click once to deposit it. An unassigned Slider looks like the one shown in Figure 6-32.

*Figure 6-32 Arrow Operations on Unassigned Slider*



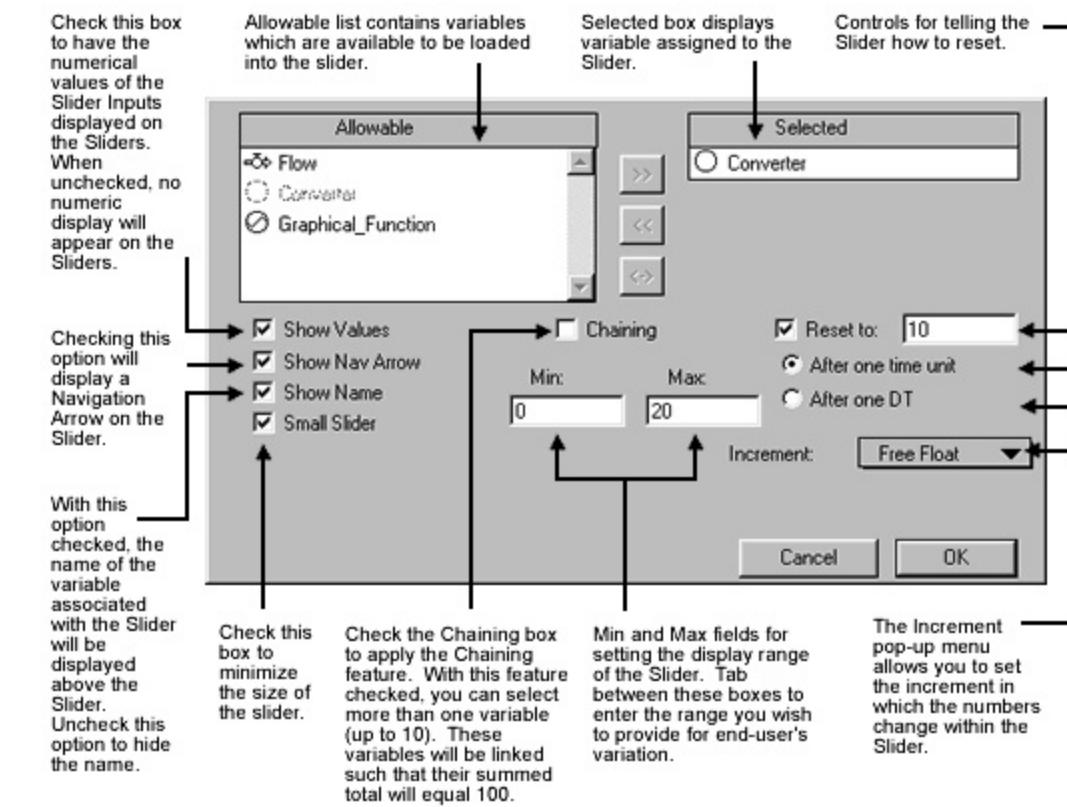
Open the Slider either by double-clicking within its border, or by selecting it and choosing Open Selection... from the Interface menu. You will see the Slider's Define dialog, shown in Figure 6-33.

To move a variable from the Allowable list to the Selected box, either double-click it, or select it and click the >> button. This will automatically replace a previously selected variable.

If you check the "Reset to:" box, another box will appear within which you can provide a numeric value to which the Slider will re-set either "After one time unit" or "After one DT" depending on your radio button selection. These features enable the modeler to save end-users from repeating themselves, by

preventing them from inadvertently buying 10 houses, or hiring 10 people a month for 2 years, when they really only wanted to execute each of these actions once.

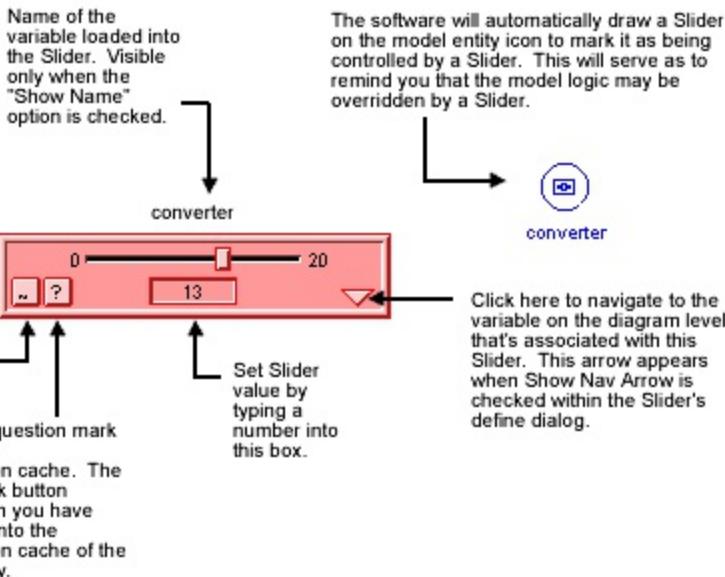
*Figure 6-33  
Slider Define Dialog*



When you click OK within the dialog, you'll see something similar to what's shown in Figure 6-34.

*Figure 6-34  
Assigned Slider*

This switch appears whenever the variable you've associated with the Slider is defined by an equation or a graphical function. Clicking on the switch will cause the equation or graphical function logic to be "turned back on" making the Slider inoperable. The switch is active only when a model simulation is "stopped." It is inactive when the simulation is running or paused.



As shown in Figure 6-35, sliders associated with model constants (i.e., as opposed to variables that were defined using an algebraic equation or as a graphical function), will display a Restore button in the lower left corner of the Slider housing. The Restore button will appear whenever the Slider's knob is moved. Clicking the button will cause the Slider's value to be restored to the numerical value that appears in the variable's equation box.

*Figure 6-35  
Slider Assigned to Constant*

This button appears on all Sliders which do not display switches whenever the Slider knob is moved. Clicking the button restores the Slider value to that of the variable's equation box.



*Tip:* The **Restore: Sliders** menu item on the Interface and Model menus will reset all sliders with constant values to their original values.

It is possible for the value set for a slider to be overridden by the constraints of Stocks. Under the following conditions, what you set won't be what you get:

- Slider assigned to outflow from a non-negativity restricted reservoir, when the non-negativity constraint comes into play.
- Slider assigned to inflow of capacity constrained or inflow limited Conveyor, when the constraint/limit would otherwise be exceeded.
- Slider assigned to inflow of Oven, when it is cooking.
- Slider assigned to inflow of Oven/Conveyor, when it is arrested.

For a discussion of the "rules of grammar" associated with stocks and flows, see [Introduction to Implicit Stock/Flow Rules of Grammar](#).

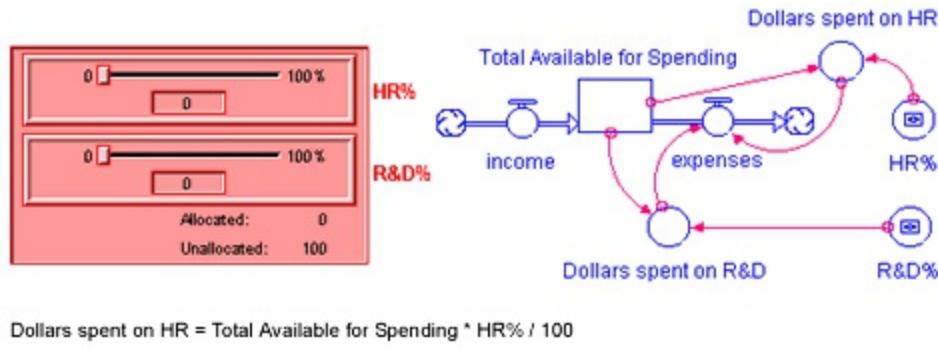
 [Related Topics](#)

# The Chained Slider

The Chained Slider builds on the basic functions and principles of operation found in the Slider object. It is useful when you have interdependent decisions - i.e., you want to permit the user to manipulate the values of each decision variable, but restrict the combined total value of the variables. For example, allocating a fixed capital budget among three major spending categories would be a good use for a Chained Slider.

The Chained Slider limits the assigned variables to a total maximum value of 100, distributed between the variables that are assigned to it. The distribution between variables is generated by the current slider settings, as determined by the model user. Frequently you will use the generated numbers as percentages applied to a resource in your model. For instance, if you want to allocate company spending between Human Resources and Research & Development, apply the Chained Slider values (number between 0 and 100) to the Total Amount available - this will calculate the real dollar amount for each of the two departments. Figure 6-36 demonstrates this concept using a sample diagram and the corresponding Chained Slider.

Figure 6-36 Chained Slider with Diagram Structure

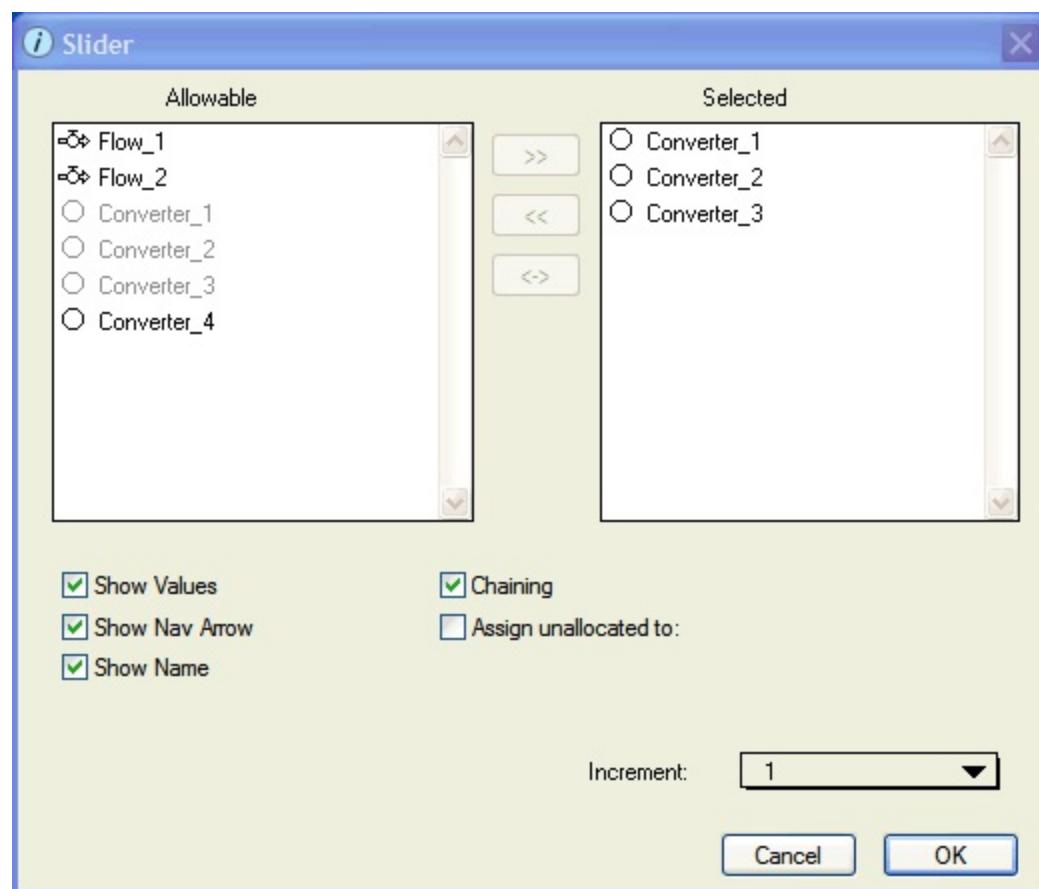


Note: Since a Pause in the simulation will occur following a full round of calculations (i.e., all Converters and Flows will be calculated then the Stocks updated), it is advisable to use a Stock as the Available Resource. Then the user can view the value of the Stock on a table or numeric display during a pause to determine the adjustment to the Chained Slider. If placed in a Converter or Flow, the Available Resource value will be updated at the same time as the slider value (after the simulation is resumed).

**Basic Operations:** After depositing an unassigned Slider on the Interface layer, double-click on the icon to access the Slider dialog. Then, check the **Chaining** check box. The dialog will be transformed into something that looks

like Figure 6-37, below.

Figure 6-37  
Chained Slider Dialog Box



Select two or more (up to 10) variables from the Allowable list and load them into the Selected list. Use the following options in the Chained Slider dialog to achieve the desired effect.

**Show Values:** Select this check box to have the numerical values of the Slider inputs displayed on the Sliders. When unchecked, no numeric display will appear on the Sliders.

**Show Nav Arrow:** Select this check box to display the Navigation Arrow on the Sliders which will navigate the user to the variables on the Model Layer.

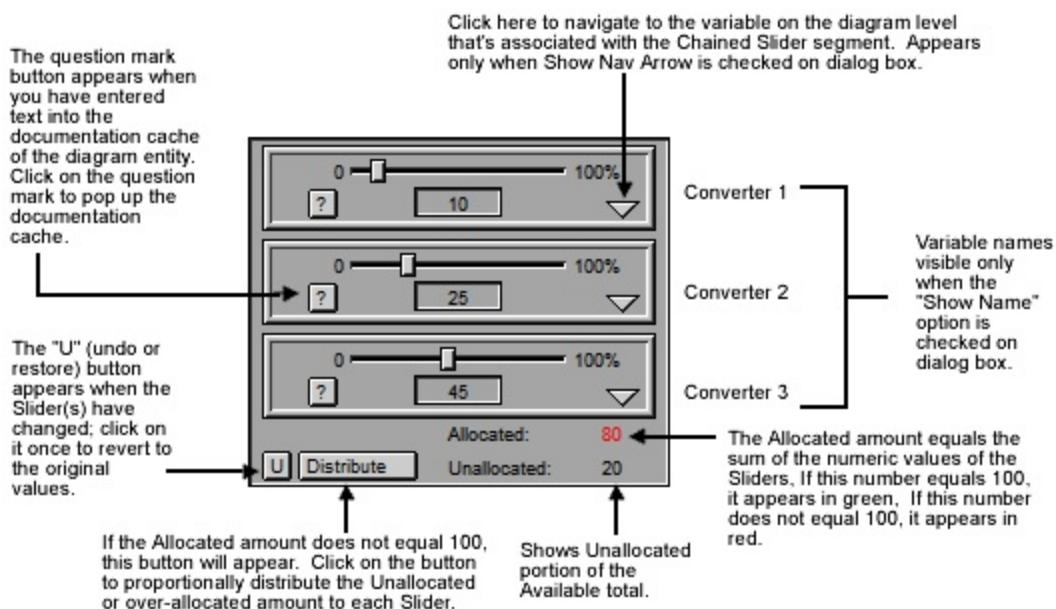
**Show Name:** Select this check box to display the variable names to the right of the Chained Slider. Uncheck this option to hide the names.

**Assign unallocated to:** Select this check box to assign unallocated values to another model variable. When this box is checked another entry field will appear in the dialog. Type the variable name in this field or click on the variable in the Allowable list.

**Increment:** Select the increment value of the Slider from the drop down menu. Selecting tenths (.1) or hundredths (.01) allows the user to type in values in tenths or hundredths.

Figure 6-38 illustrates an assigned Chained Slider.

*Figure 6-38  
Chained Slider with Unallocated Portion*



In the lower right corner of the Chained Slider casing, you will see two values marked "Allocated" and "Unallocated." These two values will always sum to 100. Allocated will always equal the sum of the numerical values of the component Sliders. Think of these as doing "scorekeeping" for the Chained Slider total. The Allocated amount appears in red if the amount does not equal 100 and green if it does.

If the Allocated amount does not equal 100, click on the "Distribute" button to proportionally distribute the Unallocated amount or over allocated amount to each Slider.

If the Allocated amount does not equal 100 and the user tries to run the model, they will be prompted with a message. The message presents options to do one of the following:

- Click on a Distribute button to have the software proportionately distribute the amount to each Slider
- Click on a Navigate button to navigate to the Chained Slider so they can manually fix it.
- Cancel the run

To restore the elements of a Chained Slider to their original values, click on the "U" button of the bottom left of the Sliders. These "U" buttons appear

whenever the Slider's button has been moved. Alternately, choose Restore then Sliders from the Map or Model menu.

 [Related Topics](#)

# Sector Frame

On the Map and Model layers, you'll use the Sector Frame to perform two major functions. Primarily, you'll use it to group together functionally related chunks of model structure. Secondarily, you can use it to display graphical images or to play QuickTime movies.

This section describes how to use the Sector Frame. It includes the following sections:

- [Creating/Naming/Moving/Resizing Sector Frames](#)
- [Operating Sector Frame Controls](#)
- [Using Sector Frame Dialog Operations](#)
- [Importing Pictures to Sector Frames](#)
- [Assigning QuickTime Movies to Sector Frames](#)
- [Playing QuickTime Movies](#)

 [Related Topics](#)

# Creating, Naming, Moving & Resizing Sectors

The Sector Frame enables you to create sectors. A sector is a grouping of functionally-related elements in a model. For example, in a model of a business organization, you might use a sector to represent each of the major processes under consideration. You might have a manufacturing sector, a marketing sector, a human resources sector, and a financial sector in the model. In a model of a food web, each of the major trophic levels might be represented by different sectors.

**Creating a Sector Frame:** There are two ways to create a Sector Frame on the Map or Model layer. First, if the "Link High-Level Map to Model" option is checked in the [Interface Prefs...](#) or [Default Settings... dialog](#), a Sector Frame will be created automatically for you whenever you deposit a Process Frame on the Interface layer.

Second, you can create a Sector Frame manually on the Map or Model layer. Simply select the Sector Frame object from the Tool Bar. As you move the cursor onto the diagram, it will change to a Sector Frame icon. Click once to deposit the Frame in the desired location. If the "Link High-Level Map to Model" option is checked in the [Interface Prefs...](#) or [Default Settings... dialog](#), you'll also get a Process Frame on the Interface layer.

---

**Note:** The Sector Frame object can not be placed within a sub-model or decision process diamond and is unavailable to you whenever one or more decision process diamond spaces are open on the Map or Model layer. To obtain the Sector Frame cursor, you must first choose Close Decision Diamonds from the Interface menu.

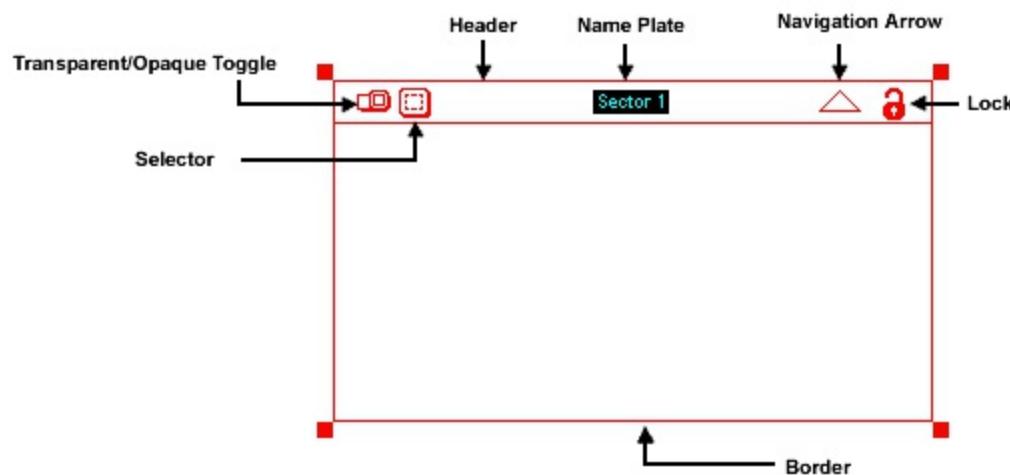
---

**Tip:** If you attempt to deposit a Sector Frame such that it will overlap another Sector Frame, the software will alert you that there is not enough room to place the Sector at the location you have chosen. When you get this alert, you can move or resize other sector frames to gain some room. Or, you can deposit the Frame in a different location.

---

Deposit a Sector Frame, and you get what's shown in Figure 6-40. The features of the Sector Frame have been identified in the Figure.

*Figure 6-40 Sector Frame Components & Controls*



**To Select:** Click on Border, or within Header. Cursor will change to an arrow when you're in the correct position.

**To Open:** Double-click on Border, or within Header. Or, select it and choose Open Selection...from Interface Menu.

**To Move:** Click-and-hold Border or Header; drag to desired location. Sector Frame will not be allowed to overlap other Frames.

**Naming a Sector Frame:** Whenever a Sector Frame is selected in isolation, its name plate will become highlighted. You can replace the highlighted name by typing. Or, you can move the cursor over a highlighted name. When the cursor changes to an I-beam, click to deposit it. Then, edit as you would with a text processor. You also can edit the Sector name from within its dialog, as indicated in Figure 6-42.

**Note:** Whenever you edit the name of a Sector Frame, the results of your editing will be reflected in the name of the corresponding Process Frame on the Interface layer if "Link High-Level Map to Model" is selected.

**Moving a Sector Frame:** To re-position a Sector, click-and-hold, either in a blank space within the header or on the border, and drag to the desired position. In moving the Sector, the software will not allow you to overlap other model Sectors.

**Resizing a Sector Frame:** To resize the Sector, first select it. Then drag on one of the four resizing handles, to make the Sector larger or smaller as required. As you resize the Sector, the software will prevent you from overlapping the boundaries of other Sectors.

# Operating Sector Frame Controls

Figure 6-40 identified the four controls on the header of the Sector Frame: the Transparent/Opaque Toggle, the Selector, the Navigation Arrow, and the Lock. The operation of each is described below:

**Transparent/Opaque Toggle:** This switch is the left-most of the controls on the Sector Frame. When you toggle from transparent (the position shown in Figure 6-40) to opaque, an opaque fill will cover whatever lies within the confines of the Sector Frame. If you have imported a picture or assigned a QuickTime movie to the Sector Frame, it will be displayed when you make the sector opaque (in the case of a movie, only a single preview frame will be displayed). Click the switch when the Sector Frame is opaque, and it once again will become transparent.

---

**Note:** When you toggle the Sector Frame to the opaque state, the Selector disappears (you would not be able to see any entities within the sector, even if they were selected), and the lock automatically locks and grays. This ensures that any entities that are within the Sector Frame remain there if you resize or move the Frame while it is opaque.

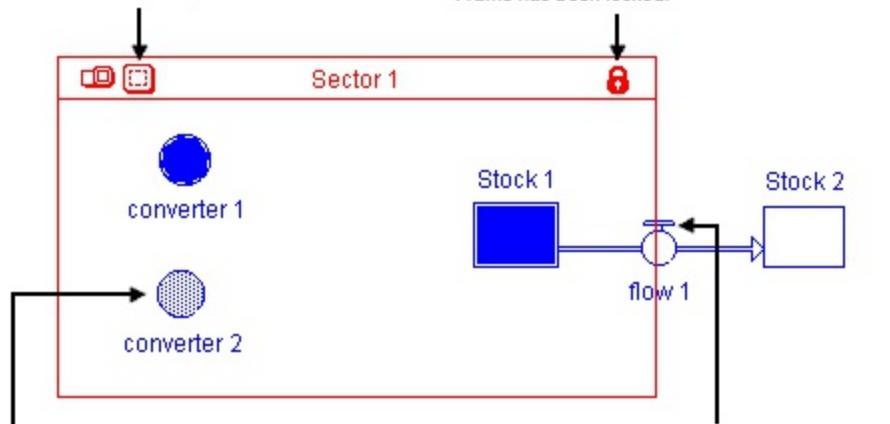
**Selector:** The second control is the Selector. Click the button to select all building blocks that the software recognizes as part of the Sector. Figure 6-41 illustrates the use of the Selector.

It's important to know which building blocks are recognized as part of a particular Sector. Whenever you set up your model to "Run Selected Sectors" (for more information, see [The Run Menu](#)), only those building blocks which are part of the selected Sectors will be run. All others will be held constant at their initial values. As Figure 6-41 indicates, there are two cases in which a building block that appears to be in the Sector will not actually be part of the Sector. First, if the Stock, Converter, or circle portion of the Flow is not completely enclosed within the Frame, the building block will not be considered part of the Sector. Second, if a locked Sector has been moved over a building block, that building block will not be considered part of the Sector. To remedy the first situation, move the building block into the Sector. To remedy the second situation, you either can move the building block out of the Sector, and then back in again, or, unlock and re-lock the Sector.

*Figure 6-41 Rules for Selector and Locked Sector*

Click here to select all building blocks which are part of the sector

Note that the Sector Frame has been locked.



The Frame was locked and then moved over this converter. the converter is gray, indicating that it is not considered part of the sector.

This flow regulator is not completely enclosed within the Frame, and thus is not considered part of the sector.

**Navigation Arrow:** The third control on the Sector Frame's header is the Navigation Arrow. This control is available to you if the "Link High-Level Map to Model option" is checked within the [Interface Prefs...](#) or [Default Settings dialogs](#). A click on the Navigation Arrow will take you from the Model layer up to the corresponding Process Frame on the Interface layer.

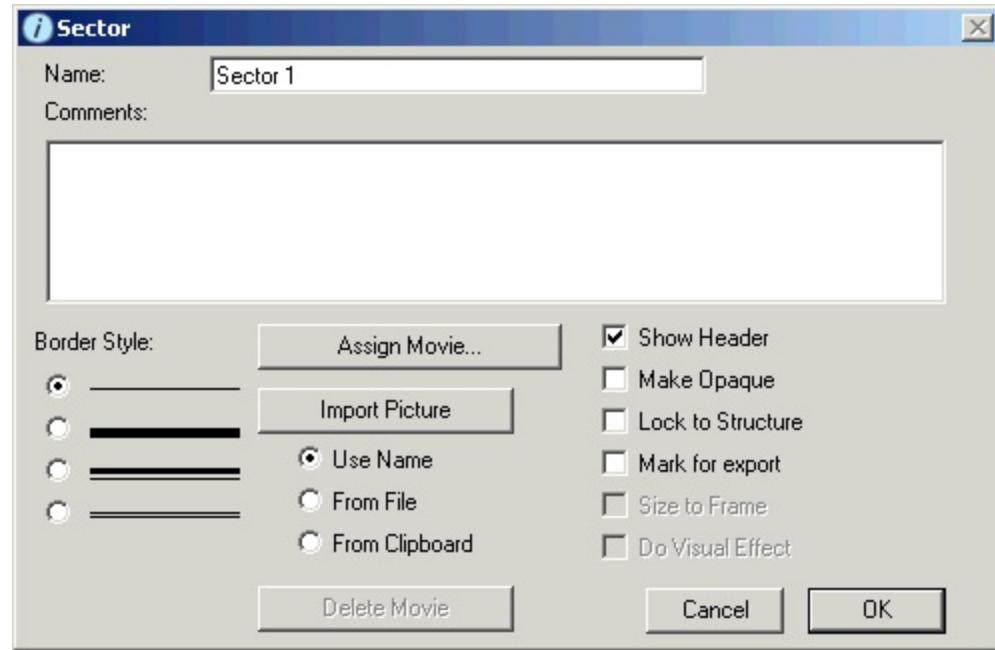
**Lock:** The final control, located on the right side of the header, is the Lock. The default position for the lock is "open." A click on the lock closes, or locks, the lock. When a Sector Frame is locked, any building block considered part of the Sector will retain its position relative to the frame. If you move or copy the Frame, all building blocks within it also will move or be copied. In addition, if you attempt to shrink a locked Sector Frame, the software will not allow you to decrease its size to the point where any of the building blocks contained within its boundary are "ejected." If you encounter difficulty in attempting to shrink a Sector Frame, this "anti-eject" feature is most likely the cause.

Sector Frames are automatically locked - and their locks become gray - whenever they become opaque, or whenever you open a decision process diamond. If you subsequently close all decision diamonds or make the Sectors transparent, the Frames will remain locked. Simply click on their locks to unlock them.

# Sector Frame Dialog Options

In addition to its surface-level controls, the Sector Frame provides controls and options within its dialog. To enter the dialog, double-click along the border, or within the header. Or, select the Sector Frame and choose Open Selection from the Model menu. Figure 6-42 shows what you'll find.

Figure 6-42 Sector Frame Dialog



As Figure 6-42 shows, the Sector Frame dialog provides a field for the Sector Name, a Comments box, Border Style options, Controls for assigning movies and for importing pictures, and a set of Check Box options. Right now, we'll look at each item in the dialog, except for the assign and import controls. For information about importing pictures and movies, see [Importing Pictures into Sector Frames](#) and [Importing QuickTime Movies into Sector Frames](#).

**Sector Name:** Type the name of the sector. The name you type here appears in the header of the Sector Frame, as well as in the corresponding Process Frame on the Interface layer if the **Link High-Level Map to Model** check box is selected in the [Interface Level Preferences dialog box](#).

**Comments:** The large, auto-scrolling Comments box is a place for you to record any thoughts, assumptions, or notes that you wish to retain regarding the Sector.

**Border Style:** The software provides you with several border style options for the Sector Frame. Choose the style that you desire by clicking on the appropriate radio button.

**Show Header:** Select this check box to display the Sector Frame's header. If

this check box not selected, the header will be hidden.

**Make Opaque:** Select this check box to have an opaque fill cover whatever lies within the confines of the Sector Frame. This check box has the same functionality as the Transparent/Opaque Toggle in the Sector Frame header (for more information, see [Operating Sector Frame Controls](#)). This check box gives you access to the option when you select to hide the header.

**Lock to Structure:** Select this check box to lock the Sector Frame so that any building block considered part of the Sector will retain its position relative to the frame. This check box has the same functionality as the Lock in the Sector Frame header (for more information, see [Operating Sector Frame Controls](#)). This check box gives you access to the option when you select to hide the header.

**Mark for Export:** Select this check box to include the Sector Frame in a report generated by a button that has been assigned the Print Exports action.

**Size to Frame:** Select this check box to have the assigned picture (or movie) re-sized so that it fits within the borders of the frame. When you re-size the frame, the picture will re-size correspondingly.

**Do Visual Effect:** Select this check box to cause a picture to undergo an "iris open" when shifting from the opaque to the transparent state, and an "iris close" when moving from transparent to opaque. This is as opposed to the picture simply disappearing (zap!) and re-appearing (plink!).

 [Related Topics](#)

# Importing Pictures into Sector Frames

To import a picture into a Sector Frame, you can drag and drop an image file onto the Sector Frame, or you can use the Import Picture button in the Sector dialog box.

**Importing Pictures with Drag and Drop:** To import a picture with Drag and Drop, drag the image file from your desktop, a browser, the Finder or Windows Explorer, or any other location on your computer onto the Sector Frame. When the Sector Frame is highlighted with a border and the pointer changes to a "+", release the mouse button to drop the picture into the Sector Frame.

**Importing Pictures with the Import Picture Button:** From within the Sector Frame dialog, the software provides three ways to import a picture, designated by the radio buttons labeled "From File," "From Clipboard," and "Use Name." The button that is "on," at the time you click the Import Picture button, determines where the picture will come from. If "From File" is on, a click on the Import Picture button will produce a standard "Get File" dialog. Navigate through the directory to locate the graphics file you wish to import (PICT files for Macintosh; for a list of Windows formats, see [Graphic File Formats - Windows Only](#)). Once you have found what you want, import the file (using the Open/OK button or a double-click on the name of the file in the list) and you're in business. The Import Picture button now will include an "\*" to let you know that you were successful in importing the picture.

If you select the "From Clipboard" option, whatever is currently in the Clipboard will be imported when you click the Import Picture button. Once again, look for the "\*" to ensure that a successful import has occurred.

---

*Tip:* There is a shortcut for importing pictures from the Clipboard. Select a Sector Frame on the Map or Model layer, then choose Paste from the Edit Menu.

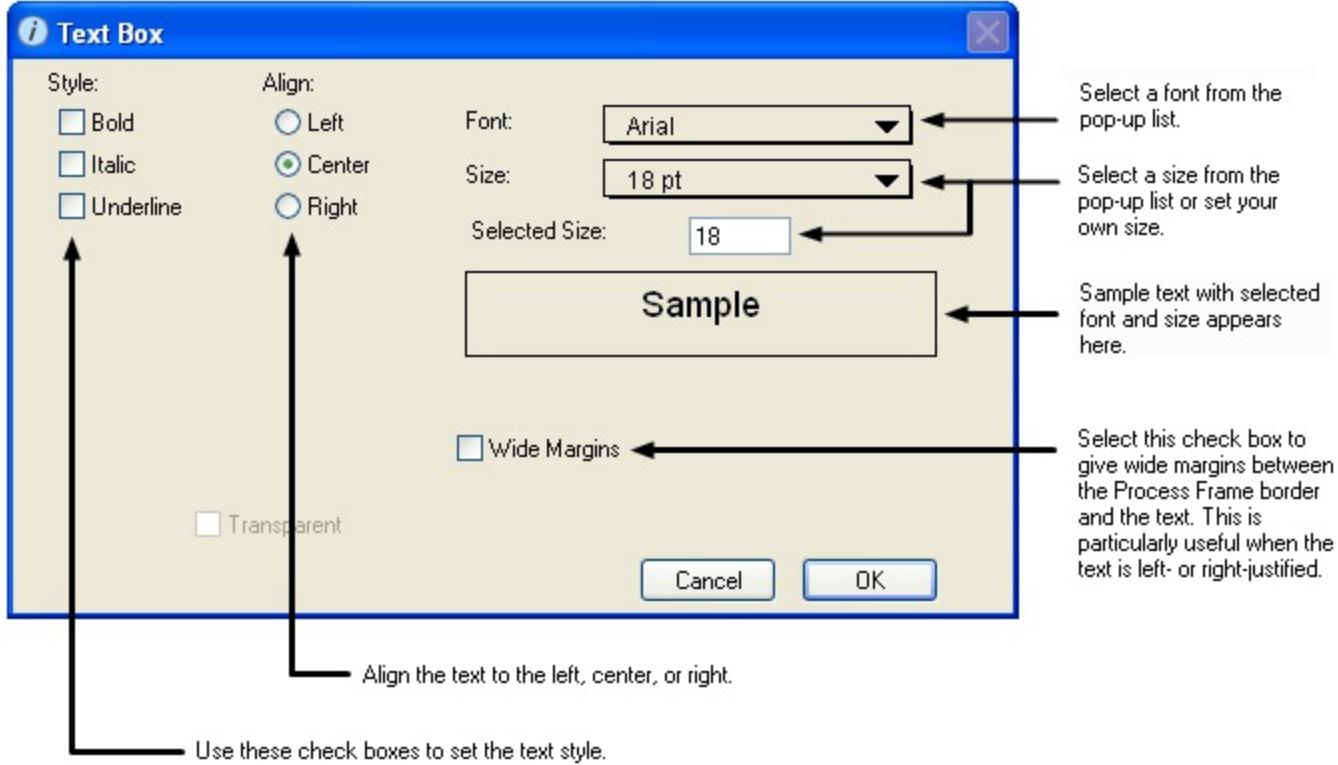
Finally, if you have selected the "Use Name" option, a dialog like the one shown in Figure 6-43 will appear when you click the Import Picture button. This dialog allows you to set the font, size, and alignment characteristics of the Sector's name, as it will appear within the main portion (not the header) of the Sector Frame.

---

*Note:* The text that is displayed within the Sector Frame will not be directly editable. To change the text, edit the name of the Sector within its dialog or header.

---

*Figure 6-43 Configuring Text When "Use Name" Option is Selected*



## [Related Topics](#)

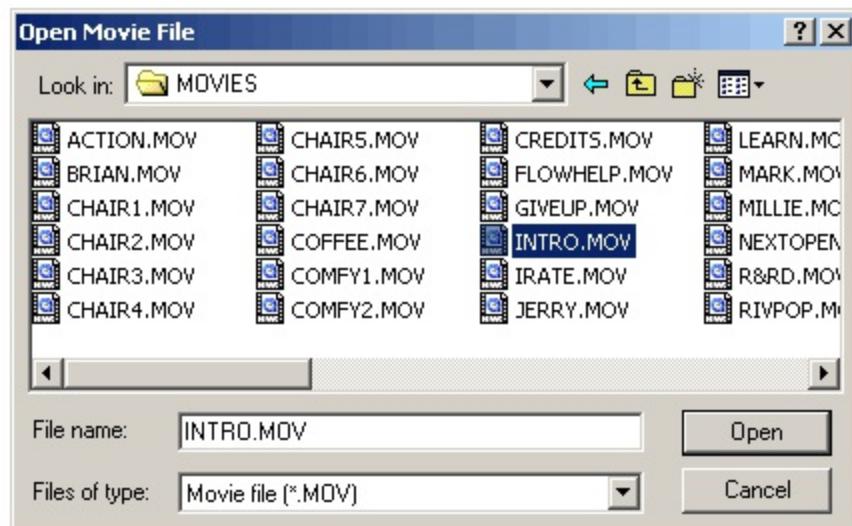
# Importing QuickTime Movies into Sector Frames

In order to assign QuickTime movies to a Sector, your machine must have QuickTime capability. To have this capability, your computer must support color, and also have QuickTime installed (be sure to restart your machine after installing QuickTime). Assign Movie will be gray if QuickTime is not installed on your machine.

To assign a movie to a Sector Frame, click on the Assign Movie button within the Sector's dialog. A standard Find File dialog will be displayed (shown in Figure 6-44).

*Figure 6-44 Find File Dialog - Choosing a QuickTime Movie*

## Windows Version



## Macintosh Version



Locate a QuickTime movie on your hard drive. The first frame of the assigned QuickTime movie will be displayed in the center of the Sector Frame as shown in Figure 6-45.

*Figure 6-45*

*Sector with a QuickTime Movie Assigned*



---

*Note:* You should note that you can use the "Size to Frame" option with movies. If the movie is smaller than the Sector Frame within which it resides, you can check the Size to Frame option (within the Sector Frame dialog) in order to stretch the movie so that it fully occupies the interior of the Sector frame. However, you should be aware that whenever you change the size of the movie by using Size to Frame, the subsequent movie playback is likely to be "jagged." QuickTime is optimized to play best using its default size.

---

*Tip:* Size to Frame allows you to center a QuickTime movie within a sector, if you have imported a movie and subsequently changed the size of the Sector Frame. Simply check the Size to Frame option in the Sector dialog and then click OK. Then, re-open the Sector dialog and un-check Size to Frame. When you click OK, the movie will be centered in the Sector Frame.

---

# Playing QuickTime Movies

To begin playing a movie, double-click within the movie frame. This movie will not have any connection to model performance. To pause the movie, click once on the movie. To resume playing, double click on the movie again. To stop playing the movie, click outside the frame.

You can include movies in as many Sector Frames as you'd like (and as memory permits). You cannot initiate the playing of more than one movie at a single instant in time. However, you can begin them playing one right after the other by double-clicking on each movie frame in rapid succession. Depending upon how much horsepower your CPU has, the resulting cascade of movie play may be just right, or painfully slow (with attendant loss of sound synchrony). You'll have to experiment to determine the performance limits of your unique hardware configuration.

 [Related Topics](#)

# Graph Pad

You'll find the Graph Pad Object on the Interface, Map, and Model layers. You'll use the Graph Pad as a repository for plotting data generated by model simulation runs. The software supports three basic types of graphs: time series plots (X, Y, Z... over time), scatter plots (X vs. Y), and bar graphs. Each graph type can be displayed in a standard format (in which the output from a single simulation run is displayed) or in a comparative format (in which the results from multiple simulation runs are superimposed atop one another).

This section documents the following Graph Pad object operations:

## Graph Pad Surface Operations

- [Creating/Moving/Resizing/Closing](#)
- [Pad Icon Operations](#)
- [Operating the Lock](#)
- [Print Button](#)
- [Dynamite Button](#)
- [Pinning and Unpinning](#)
- [Turning Pages](#)
- [Navigating to Associated Model Variables](#)
- [Activating Logic Tracing from Interface Layer Graphs](#)
- [Annotating Graph Pad Pages](#)
- [Accessing Sensitivity Setups](#)
- [Other Surface Operations](#)

## Graph Pad Dialog Operations

- [Loading/Removing/Exchanging Variables](#)
- [Scaling Variables](#)
- [Unscaling Variables](#)
- [Titling a Page](#)
- [Exercising Display Options](#)
- [Adding Pages](#)
- [Setting the Display Range](#)
- [Setting the Graph Type](#)

[The Sketchable Graph](#)

[Reading Numbers Directly from Graphs](#)

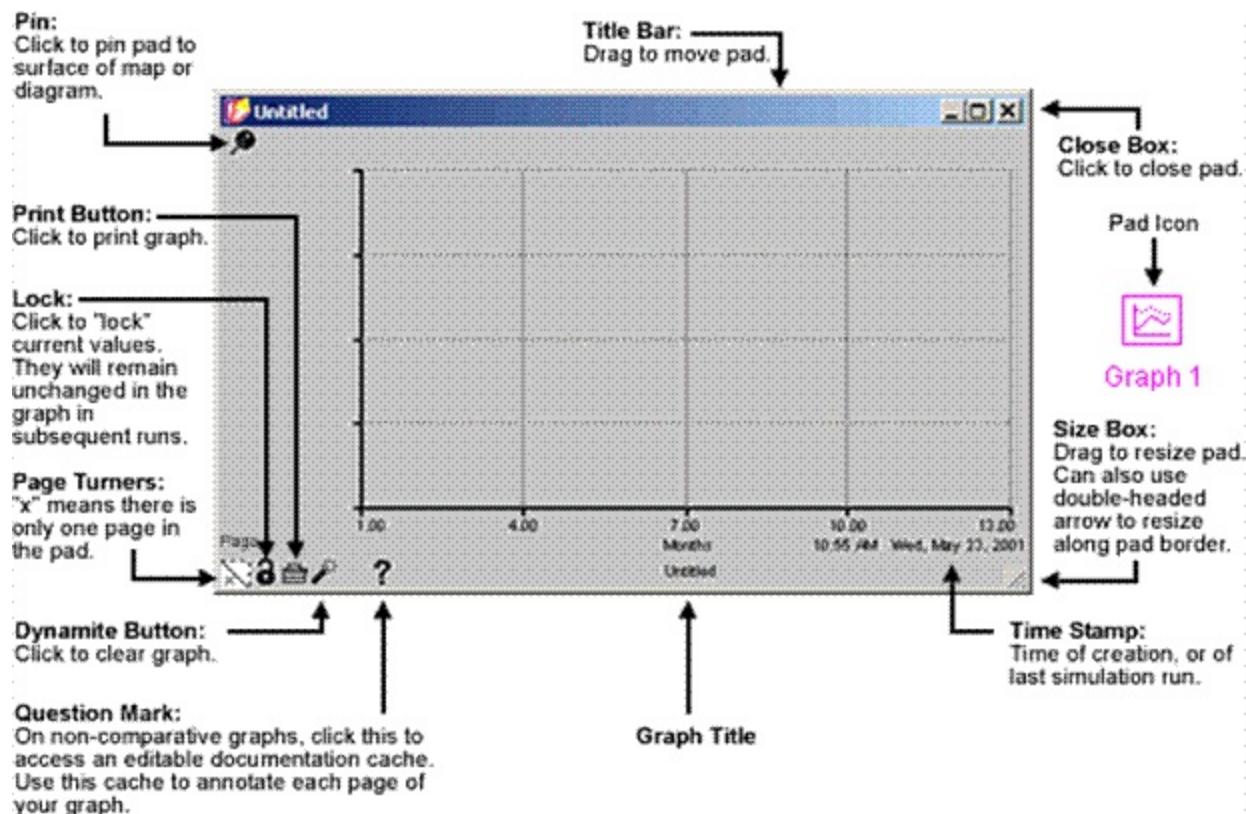
[Setting the Format for Numbers on Graphs](#)

 [Related Topics](#)

# Graph Pad Surface Operations

**Creating/Moving/Resizing/Closing a Graph Pad:** To create a Graph Pad, select the Graph Pad icon from the tool bar. Then, deposit the icon in the desired location (on the Interface, Map, or Model layer) by clicking once. When you create a Pad, you'll get a blank page such as the one shown in Figure 6-48. Note that Figure 6-48 shows where to click and/or drag to move, resize and close the Pad. It also identifies the salient features of the default Graph Pad page.

Figure 6-48 Default Graph Pad Page



**Pad Icon Operations:** As shown in Figure 6-48, associated with the Graph Pad is a Pad icon. The icon exists on the surface of the Interface, Map, or Model layer. When the Pad icon is accessible (either because you have closed the Pad or because you have made the layer containing the Pad active), it is amenable to a variety of operations. These are summarized in Figure 6-49.

Figure 6-49  
Graph Pad Icon Operations

### (a) Unselected Graph Pad Icon



Graph 1

To Select: Click within the icon's border.

To Open: Double-click within the icon's border. Or, select the icon, and choose Open Selection... from Interface Menu. Opening will take you to the top-most page of the associated pad.

To Move: Click-and-hold within icon's border; drag to desired location.

To Move Name Plate: Click-and-hold name plate of unselected Pad icon. Drag name around icon to desired location. Hold Control key (Windows) or command key (Macintosh) while dragging to constrain position of name plate to North / South / East / West. Control-click (Windows) or Command-click (Macintosh) on name plate to cause it to snap to nearest cardinal position.

### (b) Selected Graph Pad Icon



Graph 1

To Name: When a single Pad icon is selected, its name will become highlighted. Replace the highlighted name by typing. Or, move the Hand over the highlighted name. When the cursor changes to an I-beam, click to deposit it. Then, edit as you would with a text processor.

**Operating the Lock:** As shown in Figure 6-48, each Graph Pad page contains a small lock in its lower left-hand corner. A click on an open lock will close it. Another click will open the lock. When a Graph Pad page is locked, it will not be overwritten as a simulation unfolds. The lock thus provides a convenient mechanism for retaining the results from a particular simulation run.

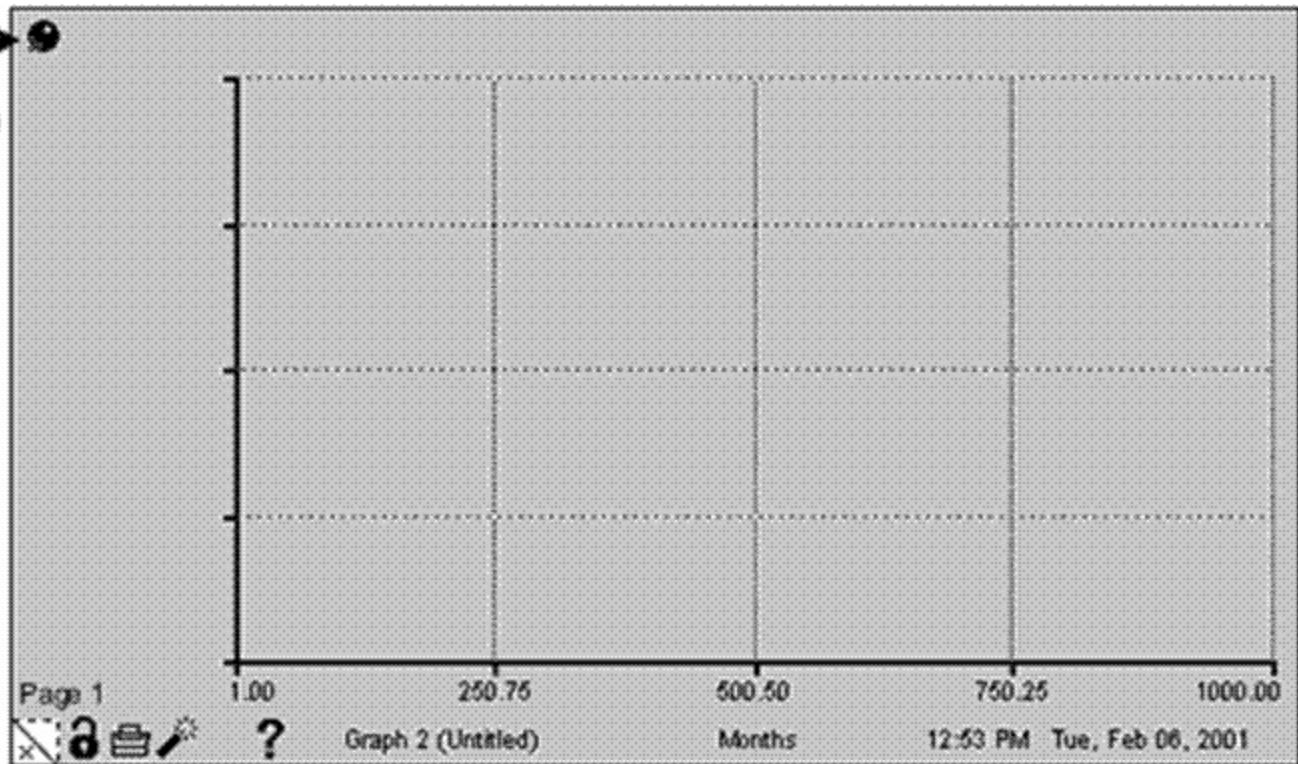
**Print Button:** Also located in the lower left-hand corner of the Graph is a small Print button. Click this button to print out a copy of the entire Graph Pad or specific pages within the Pad.

**Dynamite Button:** The last icon in the lower left corner of the Graph is the small Dynamite button. Clicking this button will clear the current page of an unlocked Graph Pad.

**Pinning and Unpinning:** The push-pin icon on the Graph Pad is used to pin and unpin the entire pad to the surface of the Interface, Map, or Model layer. Pinned pads will move along with the layer to which they are affixed, when you click, drag, scroll, etc., on the layer. Unpinned pads will "float," either above or below the layer. Click once on the push-pin to pin the pad to the surface. Click again on the push-pin to unpin the pad. Figure 6-50 illustrates a pinned Graph Pad. By default, Graph Pads are unpinned when you first add them.

Figure 6-50  
Pinned Graph Pad

Note that the pin is in the downward position. Click on it to unpin the pad.



**Notes:** When a pad is pinned, its title bar and resizing controls disappear. To move or resize, you must first unpin the pad. Because a pinned pad is attached to its layer, access to building blocks and objects is retained while the pad is extant.

Unless the entire area of the pad is within the borders of the Interface, Map, or Model layer, the Pin will be grayed out.

**Turning Pages:** When a Graph Pad contains more than one page, the page-turning apparatus in its lower left corner will become active. The Pad will also tell you what page of the Pad you are on near the lower border. As shown in Figure 6-51, a click on the upper triangle will move you to the next page in the pad. A click on the lower triangle will move you to the previous page.

*Figure 6-51  
Turning Pages, Navigating to Associated Model Variables, and Activating Logic Tracing*

Select a variable by clicking on its name

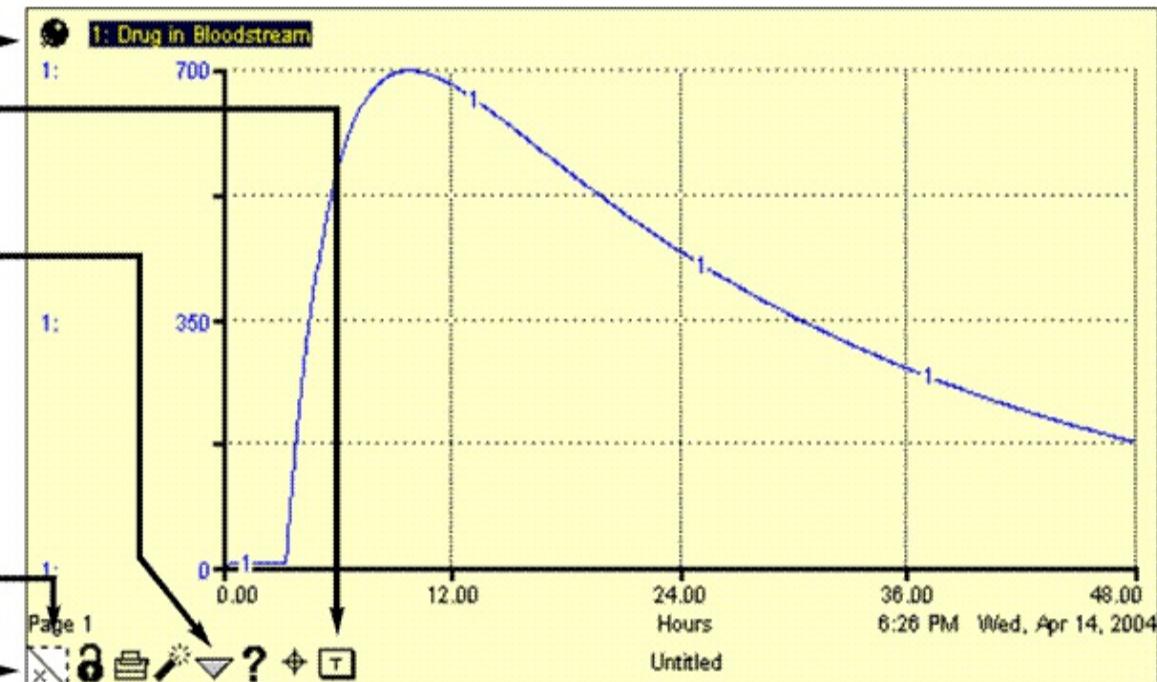
Then, on Interface Layer, click the T button to initiate a logic tracing sequence beginning with the selected variable.

Or click the down arrow to go directly to the selected variable.

[Click here to move to next page.](#)

[Click here to move to previous page.](#)

[Click here to access and annotate the page's documentation cache.](#)



Click and drag here to add an annotation event to the plot of the selected variable. Double-click on the annotation event to assign a name and annotation text. Unlock the graph using the padlock icon to clear the annotation event.

**Activating Logic Tracing from Interface Layer Graphs:** Model logic tracing is possible for graph pads that have been created on the Interface layer of the software. Select a variable by clicking on its name on the graph surface, and then click the "T" button that appears along the bottom of the page. When you do, you'll be taken to the Model layer, and will be able to sequentially traverse through the inputs to the model variable. When you're done tracing, click the left-pointing arrow beneath the Tracing tab name to exit Logic tracing. [Figure 6-17](#) shows what tracing looks like.

*Tip:* When you turn on "Analyze Mode" in the Run Specs dialog, Logic tracing can be an incredibly powerful tool for model analysis. If you encounter "interesting" model behavior on a graph, you can initiate a tracing sequence. On the Map/Model layer, as you roll the cursor over each model variable you'll see a small graph of its output along with its current value. You'll very quickly be able to diagnose and understand model behavior by traversing its structure.

**Navigation to Associated Model Variable:** From any graph on any model layer, you can navigate directly to a selected model variable. First, select a variable within the graph by clicking on its name. Then click on the down arrow that appears at the bottom of the graph (next to the Dynamite button). You'll be taken to the associated model variable.

**Annotating Graph Pad Pages:** It's often useful to annotate a graph. The annotation can help you to refresh your memory when you come back to the

graph. In addition, it can help users of your model to better understand what you were thinking as you created the graph. For all graphs except comparative graphs, you can provide text annotation via the "?" button that appears at the bottom of each graph pad page. Click the ?, and type in text in the editable documentation cache. When you're done, simply close the cache. Nothing could be easier.

**Annotating Points on a Curve:** It is often useful to draw attention to important events, shifts in feedback loop dominance, or other important dynamics by marking notable points on the curve of a time series graph. In order to mark, label, and annotate an important point, first highlight the name of that curve. The annotation point icon will appear among the icons in the lower left of the graph pad. Then, click the annotation point tool to add an annotation point to the graph. Drag the point along the graph to the desired position. Double-clicking any point will open a dialog that allows you to name and annotate that point.

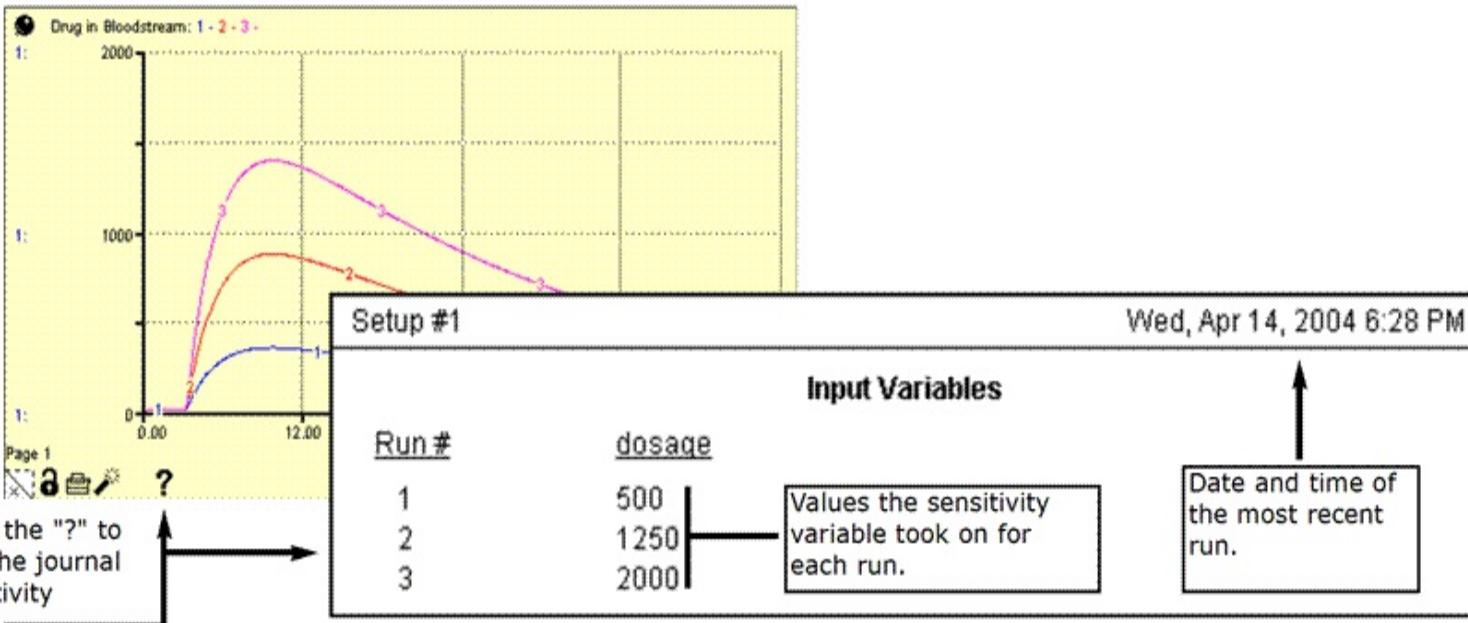
---

**Note:** Adding an annotation to a curve automatically locks the graph. This preserves the integrity of the graph and associated points by preventing a new simulation run from changing the graphed curve. Unlocking the graph will delete the annotations.

---

**Accessing Sensitivity Setups:** Whenever you have defined a Graph Pad page as a comparative graph, and have generated output using the software's sensitivity analysis capabilities, the "?" will appear in the lower left of the page. As illustrated in Figure 6-52, a click on the "?" will pop up a journal of the most recent sensitivity setup.

*Figure 6-52  
Comparative Graph Journaling*



**Note:** To learn more about using sensitivity analysis, see [The Run Menu](#).

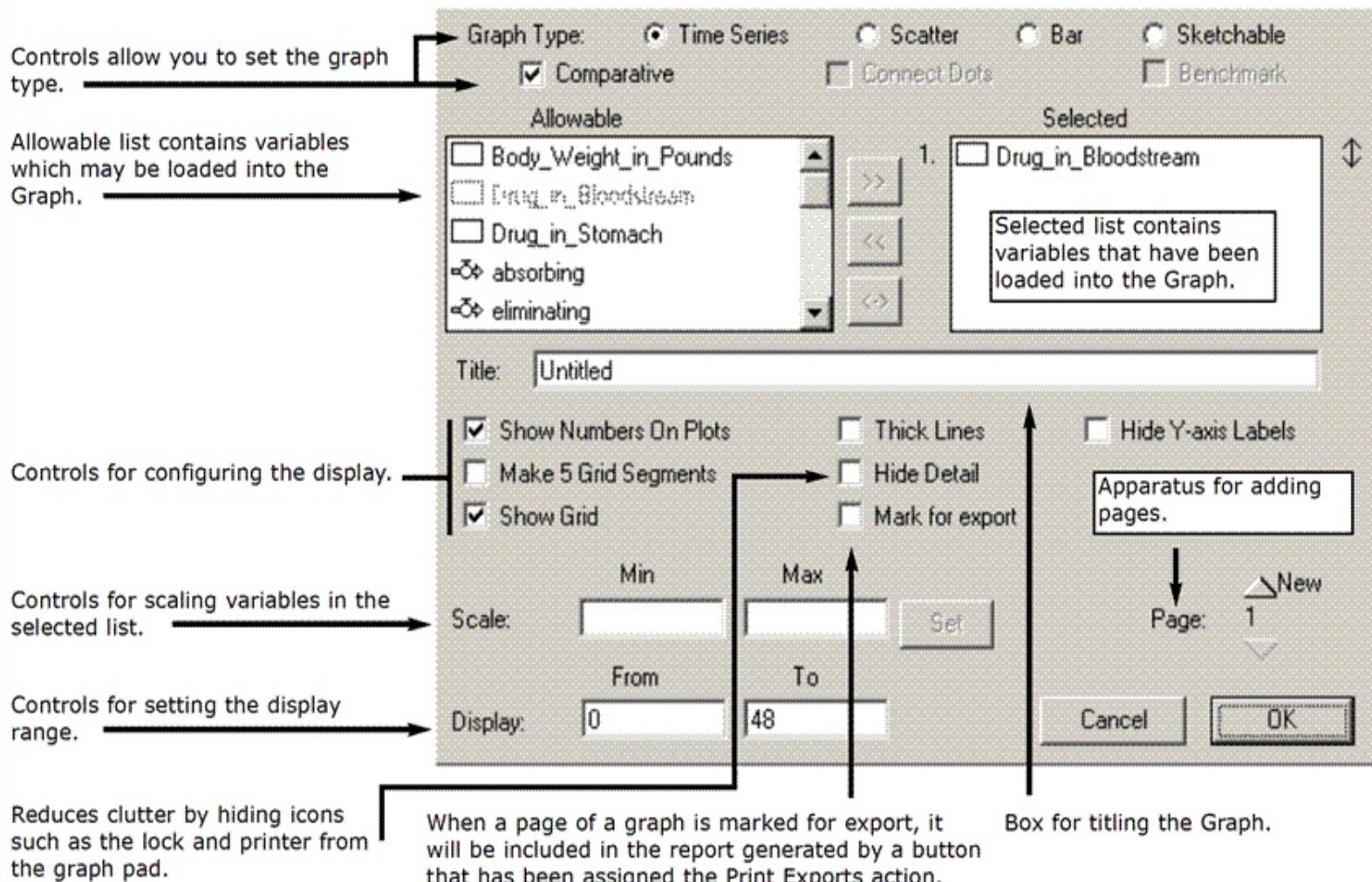
**Other Surface Operations:** Using the Paintbrush tool, you can change the color of plots, backgrounds, and Pad icons. For more information, see [The Paintbrush](#). You can clear data, specific variables, scales, and entire pages using the Dynamite tool. For more information, see [The Dynamite](#).

[Related Topics](#)

# Graph Pad Dialog Operations

To enter the Graph Pad's Define dialog, use the Arrow to double-click anywhere within the Graph Pad (except the title bar). Or, with an unpinned, active pad, choose Define Graph from the Interface or Model menu. The Graph Pad Define dialog is shown in Figure 6-53. Define dialog operations are described in the text which follows.

Figure 6-53 Graph Pad Dialog



**Loading/Removing/Exchanging Variables:** The Allowable and Selected lists, along with the buttons between the lists, are used to load, remove, and exchange the variables to be displayed in the Graph Pad page. All model variables (stocks, flows, converters) will appear in the Allowable list. Variables to be plotted appear in the Selected list. When a variable has been loaded to the Selected list, it will appear gray in the Allowable list. Here's how to move variables between the two lists:

- **Loading Variables:** To load a variable to the Selected list, double-click the variable name in the Allowable list. Or, select it, and click the >> button.

To select multiple variables, simply drag-select or shift-click (for adjacent variables in the list), or control-click (Windows) or command-click (Macintosh) for noncontiguous variables.

---

Tips: If a variable is selected within the Selected list, any variables that you load to the selected list will be placed above the selected variable.

There is a limit of 5 variables within a Time Series or Bar Graph, 2 variables on a Scatter Plot, and 1 variable within a Comparative Graph.

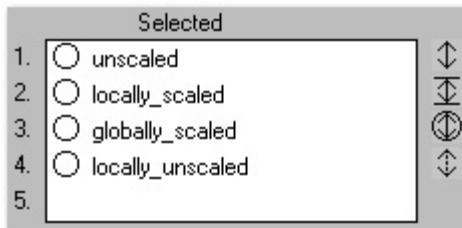
You can also easily load variables into a Graph by dragging and dropping a stock, flow, or converter icon on the Map or Model layer directly onto the Graph icon. When you view the Graph Pad Dialog for the Graph, the entities you dragged onto it are listed in the Selected list. You can also drag and drop variables directly onto a pinned Graph Pad.

---

- *Removing Variables:* To remove a variable from the Selected list, select it. Then, click on the << button between the two lists. To select multiple variables, simply drag-select or shift-click for adjacent variables, or control-click (Windows) or command-click (Macintosh) for noncontiguous variables.
- *Exchanging Variables:* To exchange one or more variables in the Allowable list for one or more variables in the Selected list, select the desired variable(s) in each list. Then, click on the <-> button. The variables will be exchanged.

**Scaling Variables:** Within the Graph Pad's Define dialog, you can set the scale for any variable that has been loaded into the Selected list of a Time Series, Bar, or Scatter Plot. If you don't set a scale, the software will automatically set one for you. For any variable or set of variables, you have four basic options: unscaled, locally scaled, globally scaled, and locally unscaled. A double-headed arrow will appear next to each variable in the Selected list, indicating its scaling status. Figure 6-54 illustrates these different options.

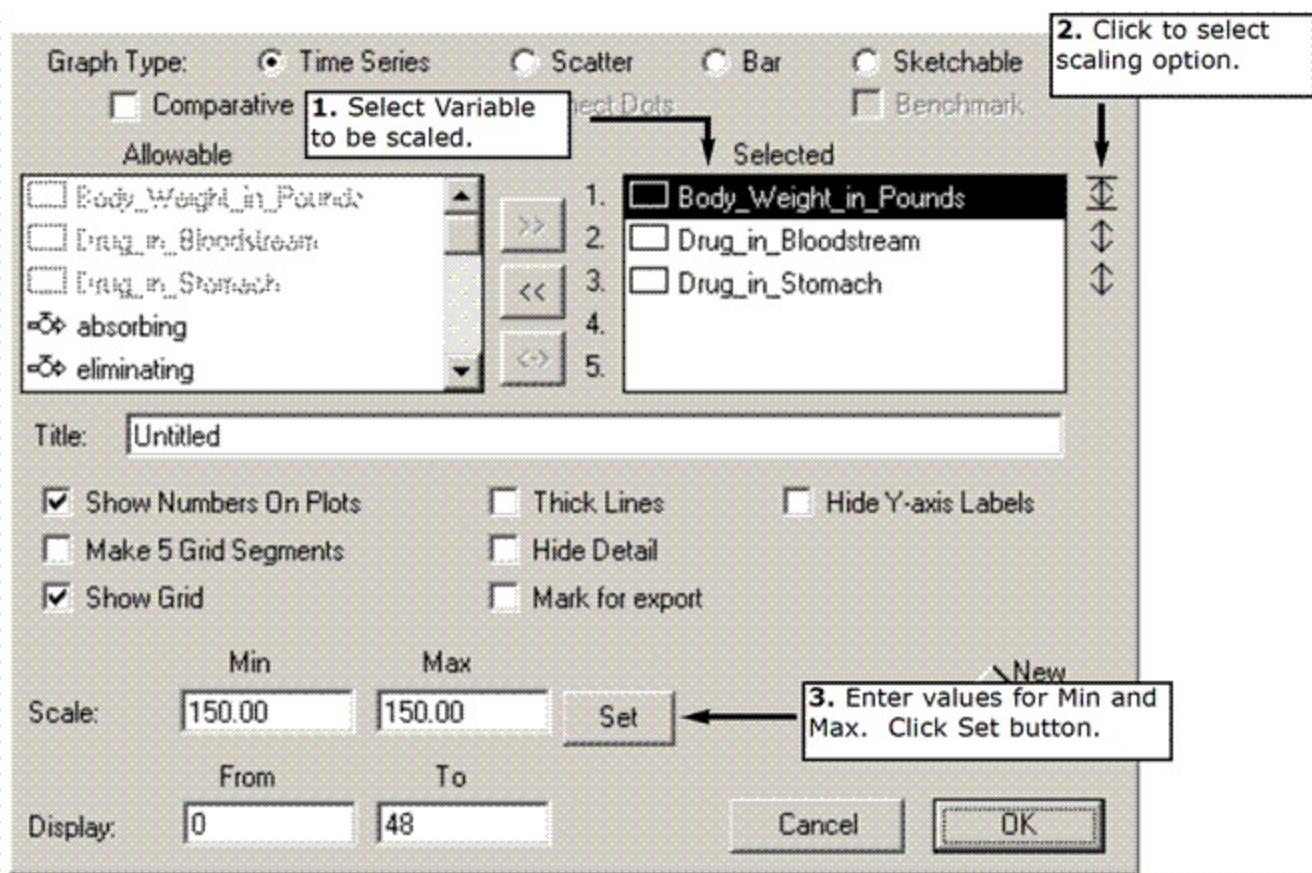
*Figure 6-54  
Variable Scaling Options*



To scale a variable from within the Graph Page dialog, you can follow a simple three-step process. First, in the Selected list, select the variable(s) to be scaled. Second, click on the scaling arrow to select a scaling option. Third, set the scale using the Scale boxes and the Set button. The details of this process are illustrated in Figure 6-55, and detailed in the text below.

- *Select variables to be scaled:* Step 1 is to select the variable(s) you want to scale from within the Selected list. When you do, the minimum and maximum values, according to the current scaling status, will appear next to the word "Scale" in the dialog.

*Figure 6-55  
Scaling Variables Within the Graph Dialog*



- *Select the desired scaling option:* Step 2 is to select a scaling option. You can set a local scale, which will apply to the current page only. Or, you can

set a global scale, which will apply to all pages (as well as Diagram animation), unless it is overridden locally.

To select local scaling, click on the two-headed vertical arrow located to the right of first selected variable in the Selected list. When horizontal lines (a "floor" and "ceiling") appear above and below each arrowhead, you're ready to set the local scale. To select global scaling, click again on the two-headed arrow. Now, the floor and ceiling will encircle the two-headed arrow like a globe. The encirclement means the variable is ready to be globally-scaled. Note that a subsequent click on the scaling icon will return it to its unscaled state - no lines, no circles.

- **Set the Scale:** As soon as a scaling option is selected, boxes will appear around the Min and Max values (indicating that you now may edit these values), and the Min value will be selected. To execute Step 3, type the desired minimum and maximum values for the scale of the variable(s). After entering these values, click the Set button. You're set!

**Unscaling Variables:** To remove a scale that you've set for a variable, select the variable in the Selected list, toggle through the scaling options until you return to the original two-headed arrow (no floor and ceiling, no circle), or to the dashed two-headed arrow (indicating globally scaled, but locally unscaled) then click the De-set button. On the page surface, you can un-scale variables which have previously been scaled by using the Dynamite tool. Click on the upper or lower value in the variable's scale (arrayed on the Y axis of the graph). If the variable had been globally scaled, it will become locally unscaled.

**Titling a Page:** To title a specific page of the Graph Pad, select the text in the Title field of the Graph Pad dialog. Then, type the desired title. The title will appear in the title bar of unpinned Pads and at the bottom of pinned and unpinned graphs. Note that titles will appear at the bottom of the graph when you print a Graph Pad.

**Exercising Display Options:** As shown in Figure 6-55, the Graph Pad Dialog provides you with four check box display options. These display options are local to the Graph Pad page.

- **Show numbers on plots** assigns a number to each variable in a time series plot. Showing numbers on plots is useful when printing graphs. Even when you are using a color monitor, numbers can help the user to differentiate the variables being plotted on a Graph Pad page.
- **Show Grid** places a grid on the background of the Graph Pad page. A grid

can make it easier for users to determine numerical values as they inspect a plot visually.

- *Make 5 Grid Segments* divides the X axis into five uniform segments as opposed to the default four segments.
- *Thick Lines* will cause the lines of a time series graph to be drawn more boldly.
- *Hide Detail* hides the date and time stamp on the graph.
- *Mark for export* will include the graph in a report generated by a button that has been assigned the Print Exports action.
- *Hide Y-axis Labels* hides the graph's Y-axis labels.

**Adding Pages:** Below the Title field is the apparatus for adding pages to a Graph Pad. Once you have loaded variables into the Selected list of the first page of the Pad, the up-arrow in the apparatus will become active. Click on the arrow (next to the word "New") to create a new page. You can add as many pages to a Pad as you'd like. You can click on the up- or down-arrows to move between pages from within the dialog. Or, you can click on the page-turn "buttons" (lower left corner) on the Graph Pad surface.

**Setting the Display Range:** The Display boxes at the lower left corner of the dialog give you an opportunity to display a portion of the data associated with a simulation run (Time Series only). The default display range is the entire simulation.

By adjusting the From and To values, you can focus on the details associated with a specific part of the simulation run. The From value must be greater than or equal to the From time you have specified in the Time Specs dialog. The To time must be less than or equal to the To time you have set in the Time Specs dialog. The display range setting you make will be local to the Graph Pad page.

**Setting the Graph Type:** At the very top of the Graph Pad dialog, four radio buttons and a check box allow you to set the Graph type. You can select between Time Series, Scatter, Bar and Sketchable graphs. For each type, the graph can be generated in a standard or a comparative mode. Comparative graphs superimpose the results of one simulation run atop those generated by previous runs.

- *Time Series* is the default type. In its standard mode, Time Series will graph a set of variables over simulation time (i.e., time is on the X axis). You can display up to five variables on a standard time series graph. In the comparative mode, the output from multiple runs will be displayed for a

single variable.

- *Scatter plots* chart the values of one variable versus another, as these values change over time. The Scatter type allows you to load only two variables to the Selected list, one per axis. The Connect Dots display option available for Scatter plots causes line segments to be drawn between plotted points.
- *Bar graphs* let you generate output from your model in a dynamic bar graph display. You can display up to five variables within a bar graph. In the comparative mode, as with the Time Series graph, the output from multiple runs will be displayed for a single variable.
- *Sketchable graphs* allow a user to "sketch out" the expected performance of a variable for comparison versus the actual performance over a simulation run. For details, see [The Sketchable Graph](#).
- *Comparative graphs*, an available option for each graph type, are often used in conjunction with Sensitivity Analysis. When comparative graphs are generated via a sensitivity analysis, a "?" will appear just to the right of the Dynamite in the lower left of the Graph Pad page. A click on the "?" will pop up a journal of the most recent sensitivity setup used to create the graph. This process was illustrated in Figure 6-52.

---

*Tip:* Comparative graphs accumulate data. Periodically, you may wish to clear this data. To do so, click the Dynamite icon in the lower left corner of the graph or position the Dynamite tool somewhere within the confines of the graphing region and click once. For an illustration, and further details, see [The Dynamite](#). Alternately, choose **Restore Graphs & Tables** from the [Model menu](#).

---

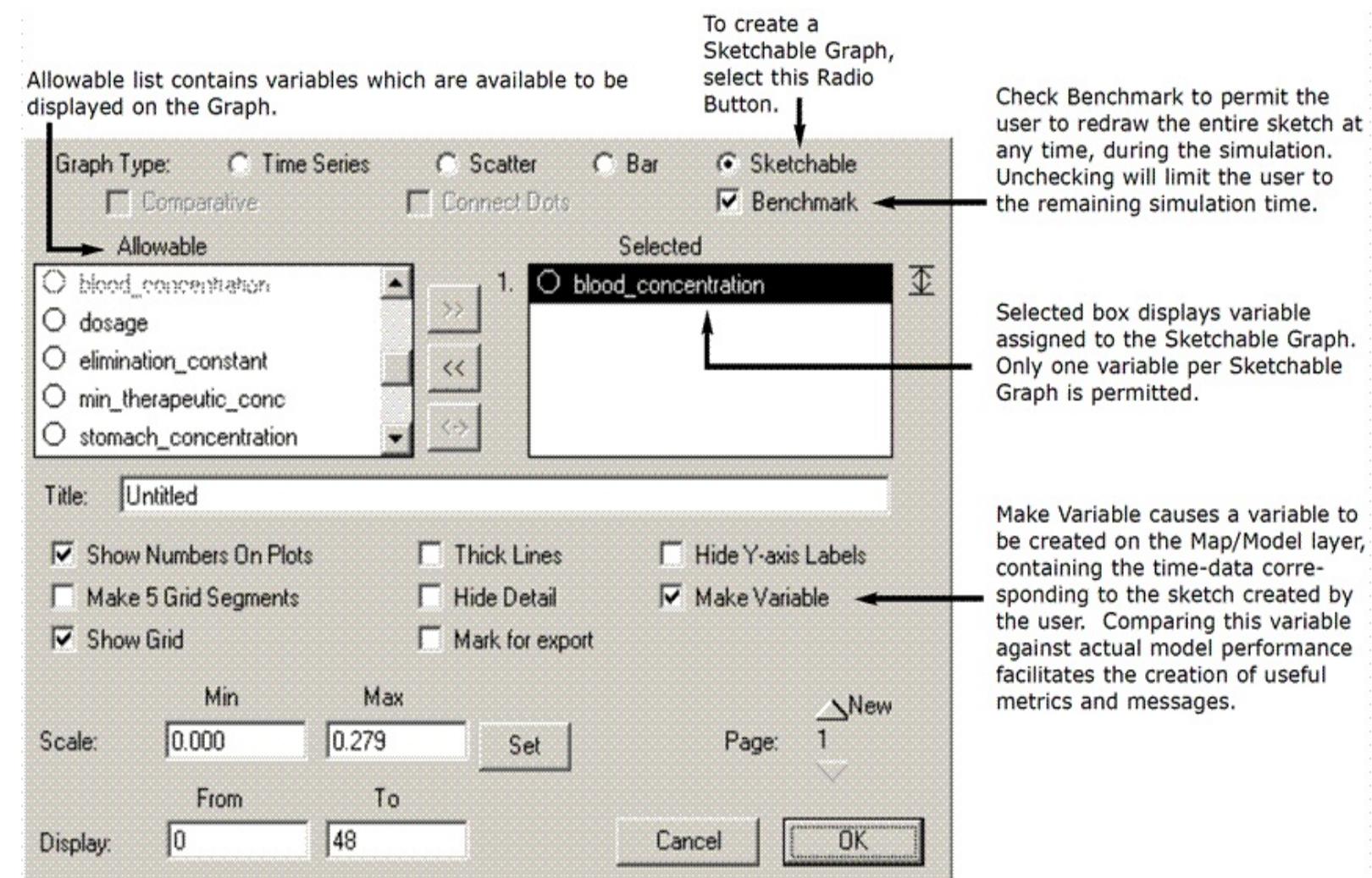
 [Related Topics](#)

# The Sketchable Graph

A Sketchable graph enables users of your model to create a sketch of the expected behavior for some model variable, over the course of a simulation run. When the simulation is run, the user will be able to compare their expected behavior against the actual behavior of the variable. Sketchable graphs come in handy when you want users of your model to "put a stake in the ground" regarding the performance of the model. Discrepancies between expected and actual performance can motivate effective, efficient learning experiences.

To create a sketchable graph, open a graph pad's Define dialog, and then select the Sketchable graph type option. When you do, the dialog will look something like what's shown in Figure 6-56.

Figure 6-56 Sketchable Graph Dialog



Note that you can load only one variable to a sketchable graph. The graph will display output for the variable, along with a user-provided sketch for the variable. The variable and its sketch will share a common scale. You *must*

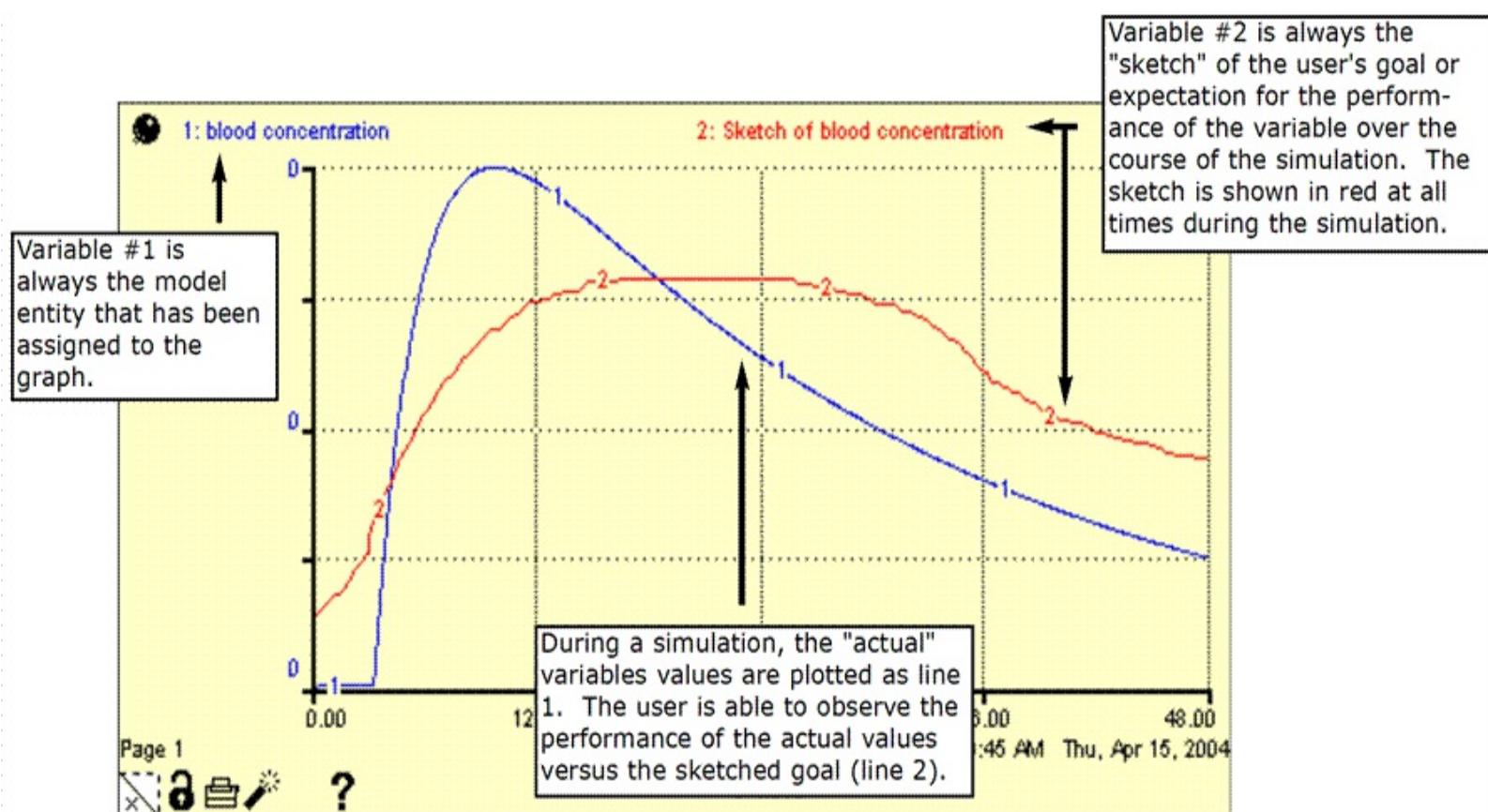
provide a scale. The process outlined in the [Scaling Variables](#) section in [Graph Pad Dialog Operations](#) shows you how to scale variables.

Whenever you create a sketchable graph, the Benchmark check box is automatically checked for you. Benchmark enables the user of your model to re-draw an entire Sketch during pauses mid-simulation, or even while the simulation is running. When Benchmark is not checked, the user can only redraw that portion of the graph that is beyond the current simulation time. No "History" (the part of the simulation that has already run) can be "re-written."

To create a sketch, click within the axes and drag in the desired pattern over time. Figure 6-57, below, shows the resultant sketchable graph after a simulation run has been completed. Note that both the actual model variable, and the sketch of expected behavior, appear on the graph.

*Tip:* When you use sketchable graphs, it's always a good idea to provide the user of your model with some instructions. Use an info button, a text box, a sound, or even the annotation cache that's part of the graph pad page to provide the user with a brief set of instructions.

*Figure 6-57*  
*Sketchable Graph Output*



 [Related Topics](#)

# Reading Numbers Directly from Graphs

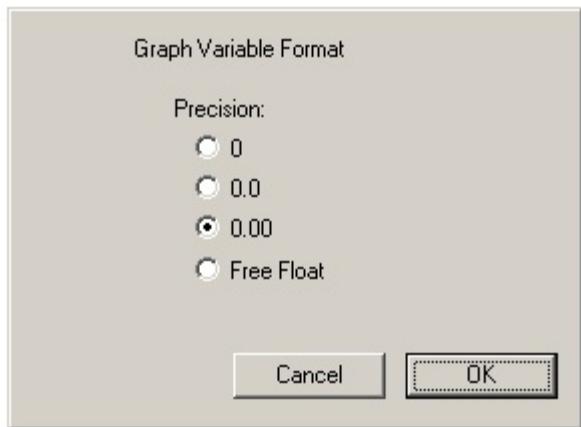
On a time series graph, click and hold along the curve associated with a variable. As you do, the magnitude of the variable at that point on the graph will be displayed underneath the variable's name at the top of the graph pad page.

 [Related Topics](#)

# Setting the Format for Numbers on Graphs

When you double-click on the name of any variable on a graph pad page, a Formatting dialog that looks like what you see in Figure 6-58 will appear. Within the formatting dialog, you can select the desired numerical precision for the selected variable. You may also set the X-axis numerical precision by double-clicking the time label.

*Figure 6-58 Formatting Dialog for Graph Variables*



## Tips:

- If the Y-axis for a graph pad appears to have no variability in its range, you need to increase the precision for the variable in question.
- When multiple variables in a graph pad page that share a common scale, the highest-precision format among the variables will be used for all variables.
- The default Y-axis format is 0. Greater precision is desirable when you need to view fractional values for variables.

[Related Topics](#)

# Table Pad

You'll find the Table Pad Object on the Interface, Map, and Model layers. You'll use the Table Pad to display the numerical output from your simulations, and as a locus for exporting data to an Excel worksheet. Linking operations are described in the [Importing & Exporting Data](#) chapter. Here, we document all other operations of the Table Pad.

This section documents the following Table Pad object operations:

## [Table Pad Surface Operations](#)

- [Creating/Moving/Resizing/Closing](#)
- [Pad Icon Operations](#)
- [Annotating Table Pad Pages](#)
- [Print Button](#)
- [Operating the Lock](#)
- [Changing Column Width](#)
- [Pinning and Unpinning](#)
- [Turning Pages](#)
- [Navigating to Associated Model Variables](#)
- [Dynamite Button](#)
- [Accessing Sensitivity Setups](#)
- [Other Surface Operations](#)

## [Table Pad Dialog Operations](#)

- [Loading/Removing/Exchanging Variables](#)
- [Setting the Table Type](#)
- [Setting the Table Orientation](#)
- [Titling a Page](#)
- [Adding Pages](#)
- [Setting the Report Interval](#)
- [Use Month Names](#)
- [Reporting Beginning vs. Ending Balances](#)
- [Reporting Instantaneous vs. Summed Flows](#)
- [Using the C --> F Button](#)

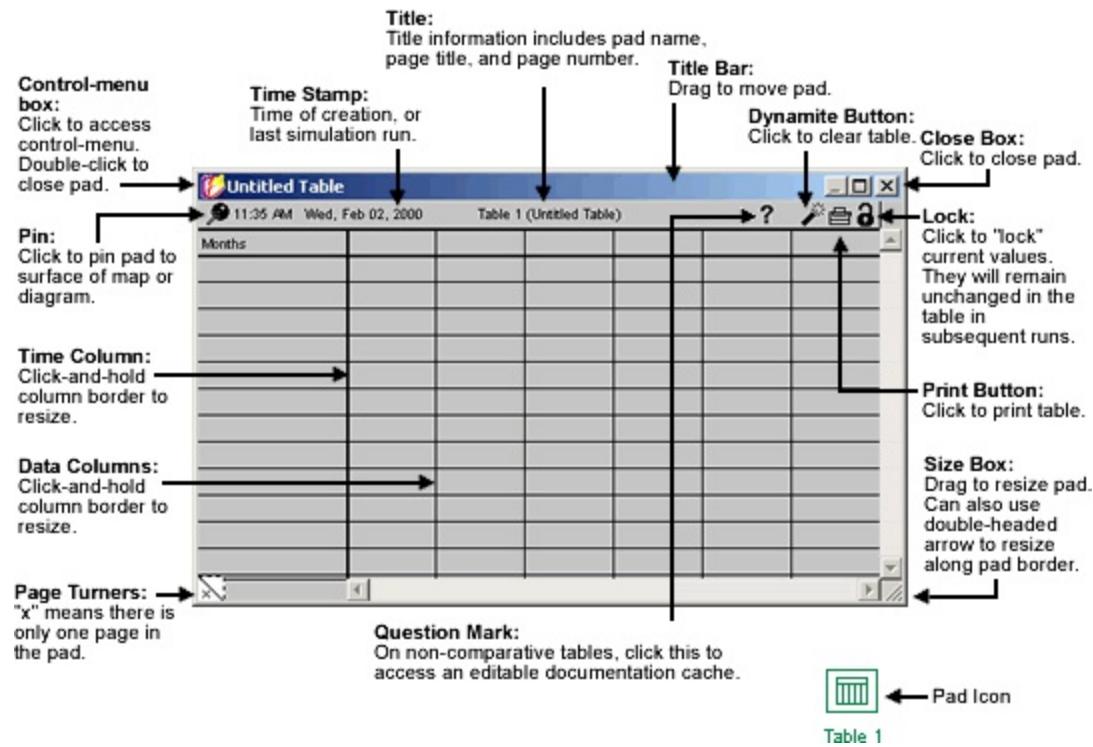
## Format Dialog Operations

 [Related Topics](#)

# Table Pad Surface Operations

**Creating/Moving/Resizing/Closing a Table Pad:** To create a Table Pad, select the Table Pad icon from the Objects palette. Then, deposit the icon in the desired location (on the Interface, Map, or Model layer) by clicking once. When you create a Pad, you'll get a blank page such as the one shown in Figure 6-59. Note that Figure 6-59 identifies the salient features of the default Table Pad page.

Figure 6-59 Default Table Pad Page



**Tip:** Running your model with an open Table will slow down the execution of a simulation. If you wish to increase the speed at which your simulation results are generated, simply close all Table Pads before choosing Run. Once the simulation is complete, re-open the objects as needed.

**Pad Icon Operations:** As shown in Figure 6-59, associated with the Table Pad is a Pad icon. The icon exists on the surface of the Interface, Map, or Model layer. When the Pad icon is accessible (either because you have closed the Pad or because you have made the layer containing the Pad active), it is amenable to a variety of operations. These are summarized in Figure 6-60.

Figure 6-60  
Table Pad Icon Operations

### (a) Unselected Table Pad Icon



Table 1

To Select: Click within the icon's border.

To Open: Double-click within the icon's border. Or, select the icon, and choose Open Selection... from Interface Menu. Opening will take you to the top-most page of the associated pad.

To Move: Click-and-hold within icon's border; drag to desired location.

To Move Name Plate: Click-and-hold name plate of unselected Pad icon. Drag name around icon to desired location. Hold Control key (Windows) or command key (Macintosh) while dragging to constrain position of name plate to North / South / East / West. Control-click (Windows) or Command-click (Macintosh) on name plate to cause it to snap to nearest cardinal position.

### (b) Selected Table Pad Icon



Table 1

To Name: When a single Pad icon is selected, its name will become highlighted. Replace the highlighted name by typing. Or, move the Hand over the highlighted name. When the cursor changes to an I-beam, click to deposit it. Then, edit as you would with a text processor.

**Annotating Table Pad Pages:** It's often useful to annotate a Table. The annotation can help you to refresh your memory when you come back to the table. In addition, it can help users of your model to better understand what you were thinking as you created the table. For all tables except comparative tables, you can provide text annotation via the "?" button that appears at the top left of each table pad page. Click the ?, and type in text in the editable documentation cache. When you're done, simply close the cache. Nothing could be easier.

**Dynamite Button:** As shown in Figure 6-59, each Table Pad page contains a small Dynamite icon in its upper right-hand corner. Clicking this button will clear the data from the current page of the Table Pad.

**Print Button:** Also located in the upper right-hand corner of the Table is a small Print button. Click this button to print out a copy of the entire Table Pad or specific pages within the Pad.

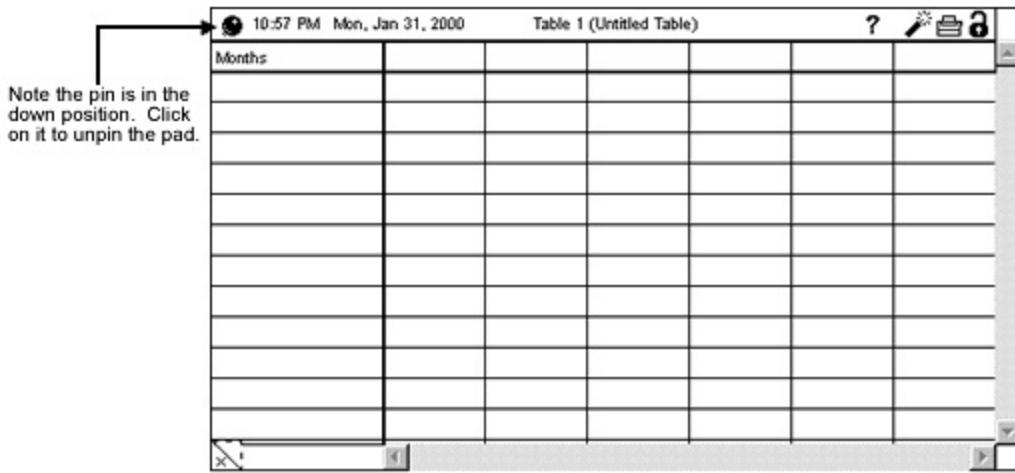
**Operating the Lock:** The last icon in the upper right corner of the Table is a small Lock. A click on an open lock will close it. Another click will open the lock. When a Table Pad page is locked, it will not be overwritten as a simulation unfolds. The lock thus provides a convenient mechanism for retaining the results from a particular simulation run.

**Changing Column Width:** To change the width of the first column (the time column) in a Table Pad page, click-and-hold on the first column divider. Then drag the divider to the desired width. The width of the time column will be changed independently of other columns. To change the width of all other columns in a Table Pad page, drag any column divider (except the first) to the desired width.

**Pinning and Unpinning:** The push-pin icon on the Table Pad is used to pin and unpin the entire pad to the surface of the Interface, Map, or Model layer.

Pinned pads will move along with the layer to which they are affixed, when you click, drag, scroll, etc., on the layer. Unpinned pads will "float," either above or below the layer. Click once on the push-pin to pin the pad to the surface. Click again on the push-pin to unpin the pad. Figure 6-61 illustrates a pinned Table Pad. By default, Table Pads are unpinned when you first add them.

*Figure 6-61  
Pinned Table Pad*



---

**Notes:** When a pad is pinned, its title bar and resizing controls disappear. To move or resize the pad, you must first unpin it. Because a pinned pad is attached to its layer, access to building blocks and objects is retained while the pad is extant.

---

The pin will be grayed out unless the entire area of a pad is within the border of the Interface, Map, or Model layer.

---

**Turning Pages:** When a Table Pad contains more than one page, the page-turning apparatus on its lower left corner will become active. At the top of the Table, the Pad will also tell you what page of the Pad you are on. As shown in Figure 6-62, a click on the upper triangle will move you to the next page in the pad. A click on the lower triangle will move you to the previous page.

*Figure 6-62  
Turning Table Pad Pages, Navigating to Associated Model Variables*

Months	price
Initial	42.00
1	42.00
2	42.00
3	42.00
4	42.00
5	42.00
6	42.00
7	42.00
8	42.00
9	42.00
10	42.00
11	42.00

When variable is selected within a Table,  
the Navigation Arrow is enabled.

**Navigating to Associated Model Variable:** You can navigate to any of the model variables within your table from directly within the table. To do this, first select a variable within the table by clicking on its name. A navigation arrow will appear at the top of the table next to the Dynamite button as shown in figure 6-62. Click the navigation arrow to navigate to the selected variable on the Map or Model layer.

**Accessing Sensitivity Setups:** Whenever you have defined a Table Pad page as a comparative table, and have generated output using the software's sensitivity analysis capabilities, a click on the "?" at the top-right of the pad will pop up a journal of the most recent sensitivity setup. Figure 6-63 illustrates this "Journaling" feature.

---

*Note:* To learn more about using sensitivity analysis, see [Sensi Specs](#).

*Figure 6-63  
Comparative Table Journaling*

The screenshot shows two windows side-by-side. The top window is titled 'Untitled Table' and contains a table titled 'Table 1 (Untitled Table)'. The table has columns labeled 'Months', '1: model output', '2: model output', '3: model output', and an empty column. Rows 1 through 7 show data: Month 1 (7.50, 15.00, 22.50), Month 2 (10.00, 20.00, 30.00), Month 3 (12.50, 25.00, 37.50), Month 4 (15.00, 30.00, 45.00), Month 5 (17.50, 35.00, 52.50), Month 6 (20.00, 40.00, 60.00), and Month 7 (22.50, 45.00, 67.50). A callout arrow points from the question mark icon in the top right of the table window to the text 'Click on the "?" to access the journal of sensitivity setups.' located in the bottom right of the same window. The bottom window is titled 'Sensitivity Setup' and shows 'Setup #1'. It has a header 'Input Variables' and a table with columns 'Run #' and 'sensi variable'. Three rows are listed: Run 1 (2.00), Run 2 (4.00), and Run 3 (6.00). A callout arrow points from the text 'The value the sensitivity variable took on for each run.' located below the table to the 'sensi variable' column. Another callout arrow points from the text 'Date and time of the most recent run.' located below the table to the date/time header 'Mon, Jan 31, 2000 10:44 PM'.

Months	1: model output	2: model output	3: model output	
1	7.50	15.00	22.50	
2	10.00	20.00	30.00	
3	12.50	25.00	37.50	
4	15.00	30.00	45.00	
5	17.50	35.00	52.50	
6	20.00	40.00	60.00	
7	22.50	45.00	67.50	

Setup #1

Input Variables

Run #	sensi variable
1	2.00
2	4.00
3	6.00

Mon, Jan 31, 2000 10:44 PM

The value the sensitivity variable took on for each run.

Date and time of the most recent run.

**Other Surface Operations:** Using the Paintbrush tool, you can change the color of specific variables, backgrounds, and Pad icons. For more information, see [The Paintbrush](#). Using the Dynamite tool, you can clear data, specific variables, and Table Pad pages. For more information, see [The Dynamite](#).

[Related Topics](#)

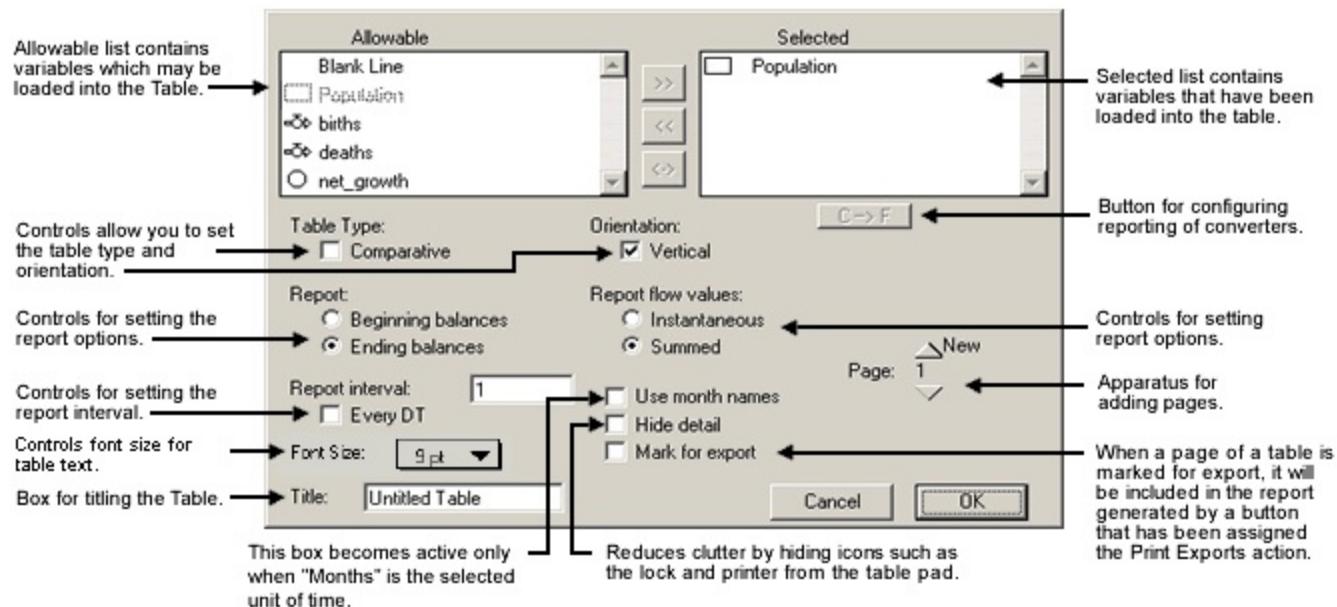
# Table Pad Define Dialog Operations

To enter the Define dialog for a Table Pad, double-click anywhere within the confines of the page (except on a variable name or on the title bar). Or, with an unpinned, active pad, choose Define Table from the Model menu. The Table Pad Define dialog is shown in Figure 6-64. Define dialog operations are described in the text which follows.

**Loading/Removing/Exchanging Variables:** The two lists at the top of the dialog, along with the buttons between the lists, are used to load, remove, and exchange the variables to be displayed in the Table Pad page. All model variables (stocks, flows, converters) will appear in the Allowable list. Variables to be plotted appear in the Selected list. When a variable has been loaded to the Selected list, it will appear gray in the Allowable list. Here's how to move variables between the two lists.

- **Loading Variables:** To load a variable to the Selected list, double-click the variable name in the Allowable list. Or, select it, and then click the **>>** button. To select multiple variables, simply drag-select or shift-click to move adjacent variables, or control-click (Windows) or command-click (Macintosh) to move noncontiguous variables.

Figure 6-64 Table Pad Dialog



*Tips: Selected list will be placed above the selected variable.*

You can also easily load variables into a Table by dragging and dropping a

stock, flow, or converter icon on the Map or Model layer directly onto the Table icon. When you view the Table Pad Dialog for the Table, the entities you dragged onto it are listed in the Selected list. You can also drag and drop variables directly onto a pinned Table Pad.

---

- **Removing Variables:** To remove a variable from the Selected list, double-click the variable name. Or select it and click on the << button between the two lists. To select multiple variables, simply drag-select or shift-click to move adjacent variables, or control-click (Windows) or command-click (Macintosh) to move noncontiguous variables.
- **Exchanging Variables:** To exchange one or more variables in the Allowable list for one or more variables in the Selected list, select the desired variable(s) in each list. Then, click on the <-> button. The variables will be exchanged.

---

*Tip:* As you load variables to Table Pad pages, note the words "Blank Line" at the top of the Allowable list. You may enter as many Blank Lines (to separate variables for formatting purposes) between variables as you'd like.

**Setting the Table Type:** Below the Allowable list, you'll find a check box entitled "Comparative." When you check the Comparative box, the page will maintain a cumulative record of the results for one variable, from each of a series of consecutive simulations, rather than displaying anew the results of each simulation. When a comparative table is generated in conjunction with Sensitivity Analysis, a "?" will appear in the top-right corner of the page. A click on the "?" will pop up a journal of the most recent sensitivity setup used to create the page. This process was illustrated in Figure 6-63.

---

*Tip:* To clear comparative data, click on the Dynamite icon in the upper right-hand corner of the Table or position the Dynamite tool within the confines of the numerical display portion of the page, then click. For more information, see [The Dynamite](#).

**Setting the Table Orientation:** The Orientation check box gives you the option of arraying your table output in vertical columns or in horizontal rows.

**Setting the Font Size:** Use the Font Size option to select the size of the font used in all table text.

**Titling a Page:** To title a specific page of the Table Pad, select the text in the Title field of the Table Pad dialog. Then, type the desired title. The title will appear in the title bar of unpinned Pads. It also will appear with the Pad name and page number at the top of the page. Note that titles will be printed at the bottom of the Table when you print a Table Pad.

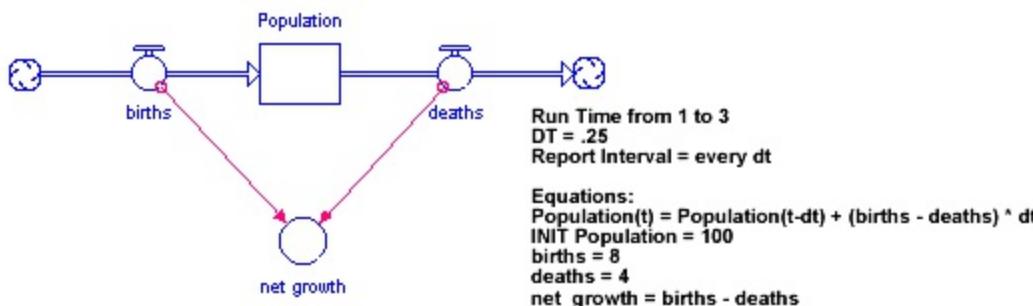
**Adding Pages:** Above the OK button is the apparatus for adding pages to a Table Pad. Once you have loaded variables into the Selected list of the first page of a Pad, the up-arrow in the apparatus will become active. Click on the arrow (next to the word "New") to create a new page. You can add as many pages to a Pad as you'd like. You can click on the up or down-arrows to move between pages from within the dialog. Or, you can click on the page-turn buttons on the Table Pad surface.

**Setting the Report Interval:** The Report interval determines at what interval the software will report numerical values on the table during a simulation. You can use the Report interval box to set the Report interval to any value greater than or equal to DT, and less than or equal to the time at which you've specified your simulation to end. In addition, you can check the Every DT box to cause the software to report numerical values each time a round of calculations is made. Such a setting is a good choice when debugging your model, because you can see the results of the calculations, as the calculations are made. If you subsequently un-check Every DT, the Report interval box will reappear. Report interval is global. It applies to all pages in all Table Pads.

**Use Month Names:** When "Months" is selected as the unit of time in the Run Specs... dialog, the Use Month Names check box becomes available to you. Check the box to cause time headings in the table to be displayed textually as month names rather than numerically. The software will map time 1 to the month January, time 2 to February, etc.

**Reporting Beginning vs. Ending Balances:** There are several options for, and issues associated with, reporting numerical values in Tables. To illustrate these options and issues, we'll make use of the simple system shown in Figure 6-65. The system consists of Population, births, deaths, and net growth. You may wish to construct the model, to work through these issues.

*Figure 6-65  
Simple Population Model*



In choosing whether to report Beginning or Ending balances, the issue has to do with when the values of stocks and converters are displayed, relative to the

time at which flow values are displayed. The issue arises because of an ambiguity in the use of numbers to reflect points in time. For example, when you receive an Annual Report dated 1992, it is understood that this means "for the year 1992." However, as a number, 1992 means 1992.0. 1992, as used in "for the year 1992" really means 1992.999... (i.e., it's the end of the year 1992). Although the spoken language glosses over this ambiguity, computers can't! They need to know: do you mean end of 1992 or beginning of 1992?

The software allows for either interpretation. The Ending balances option will attach values for all stocks, flows, and converters to the end of each report interval. If you opt for Beginning balances, values will be attached to the beginning of each report interval. Whichever Beginning/Ending Balances choice you choose will apply to all pages of all Table Pads.

We'll use the model outlined in Figure 6-65 to illustrate the differences between Beginning and Ending Balances reporting. The results displayed in Figure 6-66 show values for Population (the stock) under both Beginning and Ending Balances options. First, note that under the Ending balances option there is no Month 2.0. Instead, there is "end of time period 1." The value for Population at "1:end" corresponds exactly with the value for Population at 2.00. Note also that when the Ending Balances option is in effect, an "initial" value is displayed for Population. This is the value the stock takes on at the very outset of the simulation. It's the same value that's displayed for Time 1.00 in the Beginning Balances case. Likewise, a "Final" value appears in the Beginning Balances case.

*Figure 6-66  
Ending vs. Beginning Balances - Summed Reporting of Flows*

Ending Balances..				Beginning Balances..			
Months	Population	births	deaths	Months	Population	births	deaths
Initial	100.00			1.00	100.00	2.00	1.00
1:.25	101.00	2.00	1.00	1.25	101.00	2.00	1.00
1:.50	102.00	2.00	1.00	1.50	102.00	2.00	1.00
1:.75	103.00	2.00	1.00	1.75	103.00	2.00	1.00
1: end	104.00	2.00	1.00	2.00	104.00	2.00	1.00
2:.25	105.00	2.00	1.00	2.25	105.00	2.00	1.00
2:.50	106.00	2.00	1.00	2.50	106.00	2.00	1.00
2:.75	107.00	2.00	1.00	2.75	107.00	2.00	1.00
2: end	108.00	2.00	1.00	Final	108.00		

One other thing occurs when the Ending Balances option is in effect. Flow values are reported out one report interval later than where they appear when Beginning Balances is in effect. The shift in the reporting of flow values is suggested in Figure 6-66, and illustrated clearly in Figure 6-67 where deaths

is defined with a step increase at time 2.0.

*Figure 6-67  
Shifting of Flow Value Display*

Ending Balances..				Beginning Balances..			
Months	Population	births	deaths	Months	Population	births	deaths
Initial	100.00			1.00	100.00	2.00	1.00
1: .25	101.00	2.00	1.00	1.25	101.00	2.00	1.00
1: .50	102.00	2.00	1.00	1.50	102.00	2.00	1.00
1: .75	103.00	2.00	1.00	1.75	103.00	2.00	1.00
1: end	104.00	2.00	1.00	2.00	104.00	2.00	2.00
2: .25	104.00	2.00	2.00	2.25	104.00	2.00	2.00
2: .50	104.00	2.00	2.00	2.50	104.00	2.00	2.00
2: .75	104.00	2.00	2.00	2.75	104.00	2.00	2.00
2: end	104.00	2.00	2.00	Final	104.00		

The step increase in deaths raises it up to equal births. Note that the step-increase is displayed at the end of the first DT of Month 2 (i.e., Month 2:.25) in the Ending balances case, and at Month 2.00, which is one DT earlier, in the Beginning balances case. In the Beginning balances case, births are equal to deaths at Month 2.0. Yet the value of the stock has incremented from 103 to 104. In the Ending balances case, the step-increase is displayed at the end of the first DT of Month 2. This is the value of the flow which prevailed over the interval from the last DT of Month 1 to the end of the first DT of Month 2. The stock therefore is not incremented during the first DT of Month 2.

You'll have to decide which display option makes most sense for your particular simulation. In general, when you are working with financial variables which often are reported in end-of-time-interval fashion, it probably makes most sense to use the Ending balances option.

**Reporting Instantaneous vs. Summed Flows:** The next issue associated with the display of numerical values in Tables is with respect to flows. Should you report them as instantaneous values, or should you sum them over the report interval? The option you choose applies to all pages of all Table Pads. When summed reporting is in effect, the values for flows are displayed as the amount which actually increments (or decrements) the stock over the Report interval you have specified. In Summed mode, with a report interval larger than DT, the flow values that are displayed will be the sum of the flow values that would have been displayed for each DT during the Report interval. To illustrate, shown in Figure 6-68 are values for births and deaths displayed in Summed mode, first with a Report interval of DT, and then with a Report interval of 1 (Ending balances is in effect).

*Figure 6-68*

## Comparing Summed Values for Flows with Differing Report Intervals

**Report Interval = DT...**

Months	Population	births	deaths
Initial	100.00		
1: .25	101.00	2.00	1.00
1: .50	102.00	2.00	1.00
1: .75	103.00	2.00	1.00
1: end	104.00	2.00	1.00
2: .25	105.00	2.00	1.00
2: .50	106.00	2.00	1.00
2: .75	107.00	2.00	1.00
2: end	108.00	2.00	1.00

**Report Interval = 1.0...**

Months	Population	births	deaths
Initial	100.00		
1	104.00	8.00	4.00
2	108.00	8.00	4.00

With a Report interval equal to DT, births - which are defined as equal to a constant 8 per month - appear as a constant 2 per DT (since DT = 0.25 and there are 4 DT's in a month). Similarly, deaths - defined as equal to a constant 4 per month - are reported as a constant 1 per DT. The net of births over deaths therefore is 1 per DT, which is exactly the amount by which the stock (Population) is incremented per DT. In the second Table, the Report interval is set to 1.0. This means that the flow values are cumulated over 1 month (or 4 DTs). Flow values do not usually remain constant from DT to DT, as they do in this illustration. Hence, the cumulative value displayed for flows usually will not be equal to  $(1/DT)$  times the flow value in the first DT of a time unit.

It is also possible to display the instantaneous values of flows. When you use Instantaneous reporting, the actual calculated flow value, as determined from the flow's equation logic, is displayed. In Figure 6-69, for example, during each DT the births flow is reported as 8. The deaths flow is reported as 4. The difference between the two is 4. Yet Population increases by only 1 in each DT. This is due to the fact that before the net difference between births and deaths is added to the previous value of Population to produce a new value for Population, the difference is multiplied by DT (for this illustration, DT is set to 0.25). When reporting flow values in the instantaneous mode, this multiplication is not considered (for display purposes). Obviously, however, it is being considered when making the calculation to update the value of the stock.

*Figure 6-69  
Instantaneous Reporting of Flows --  
Ending Balances, Report Interval = DT*

Months	Population	births	deaths
Initial	100.00		
1: .25	101.00	8.00	4.00
1: .50	102.00	8.00	4.00
1: .75	103.00	8.00	4.00
1: end	104.00	8.00	4.00
2: .25	105.00	8.00	4.00
2: .50	106.00	8.00	4.00
2: .75	107.00	8.00	4.00
2: end	108.00	8.00	4.00

**Using the C -> F Button:** Located below the **Selected** list in the Table Pad define dialog, the C -> F button allows you to make converters act like flows in their Table Pad display. The issue arises because, for Table display purposes, converters are not always treated in the same way as flows. The default is to report converters as instantaneous values. When you are reporting flows as Summed, there can be a mismatch between the reporting of a flow value, and the reporting of a converter value which depends upon that flow.

For example, as the left panel of Figure 6-70 illustrates, the value displayed for net growth (a converter) is not equal to the difference between the displayed values for births and deaths (both flows). This is because net growth is being considered to be a converter (for display purposes), and hence its instantaneous value is being displayed. (If you have been building this example as we have progressed you will notice the software automatically C->F'd net growth. Any converter defined by flows is automatically C->F'd on tables.)

When net growth is C -> F'd, you obtain the values displayed in the Table (shown in the right panel of Figure 6-70) - which are the values you'd expect.

*Figure 6-70  
Displaying Converter Values without and with C --> F --  
Ending Balances, Summed Reporting of Flows*

Without C → F..

Months	Population	births	deaths	net growth
Initial	100.00			4.00
1: .25	101.00	2.00	1.00	4.00
1: .50	102.00	2.00	1.00	4.00
1: .75	103.00	2.00	1.00	4.00
1: end	104.00	2.00	1.00	4.00
2: .25	105.00	2.00	1.00	4.00
2: .50	106.00	2.00	1.00	4.00
2: .75	107.00	2.00	1.00	4.00
2: end	108.00	2.00	1.00	4.00

With C → F..

Months	Population	births	deaths	net growth
Initial	100.00			
1: .25	101.00	2.00	1.00	1.00
1: .50	102.00	2.00	1.00	1.00
1: .75	103.00	2.00	1.00	1.00
1: end	104.00	2.00	1.00	1.00
2: .25	105.00	2.00	1.00	1.00
2: .50	106.00	2.00	1.00	1.00
2: .75	107.00	2.00	1.00	1.00
2: end	108.00	2.00	1.00	1.00

Whenever a converter is a function of one or more flows (but no stocks, or other converters) - and unless that converter is a graphical function - it will be automatically C → F'd upon entry into the Selected list in the Table dialog. You may of course override this automatic conversion. To do so, select the converter, then click the button labeled C <- F (the arrow is now reversed) which appears below the Selected list in the dialog. You also may C → F any converter you choose simply by selecting it and then clicking once on the C → F button. Whenever a converter has been C → F'd, an "\*" will appear to the left of the converter's name in the Selected list.

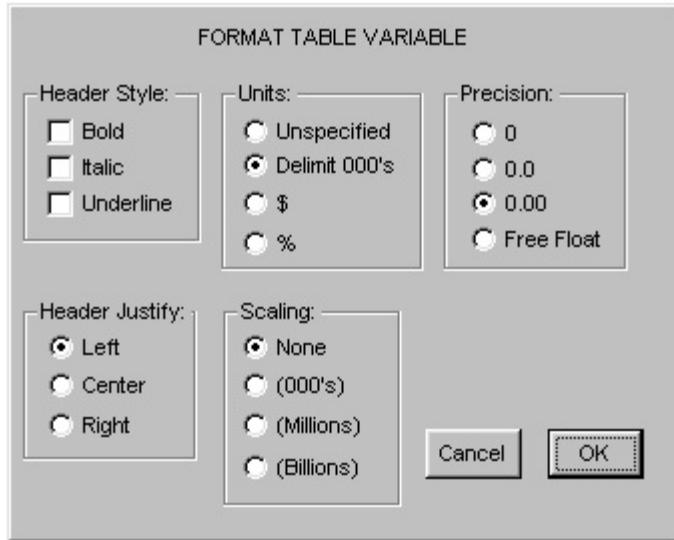
Note that the C → F operation is local to the Table Pad page you are defining. In general, if you have converters that are playing the role of flow concepts in your model, make sure that they are using the C → F option! If you are getting what you consider to be "strange" numerical results displayed in your Tables for converters, you should check to see if these converters really should be treated as flows (for display purposes). If they should, C → F them. But then remember that you have done it! The "\*" in front of the variable's name should help to jog your memory.

 [Related Topics](#)

# Format Dialog Operations

To format a variable in a Table, double-click on the variable's name on the table surface (you also can multiple-select variables before double-clicking). When you do, the dialog shown In Figure 6-71 will appear. Use it to make your formatting selections, then click OK. Your Precision, Units, or Scaling choices will change only the display of the variable in the Table. Your choices will not change the way the variable is represented either for graphing or internal calculation purposes.

Figure 6-71 Table Format Dialog



For example, if a variable takes on a value of 1 million during a simulation, and you've formatted it in (millions), its value will be displayed as 1.00 in the Table. However, for purposes of graphing, or for performing any calculations with the variable, the value of 1 million will be used.

 [Related Topics](#)

# Status Indicator

Use the status indicator to provide information about the status of key outputs in your simulation model. You can configure a status indicator to display as a simple lamp, or as a speedometer-type gauge. In either case, the indicator will light up using green, yellow, or a flashing red color to indicate the status of the key output. As you create the status indicator, you can set a range of values for green, yellow, or red display. Status Indicators thus are an excellent mechanism for telling users of your models the status of the system at a glance.

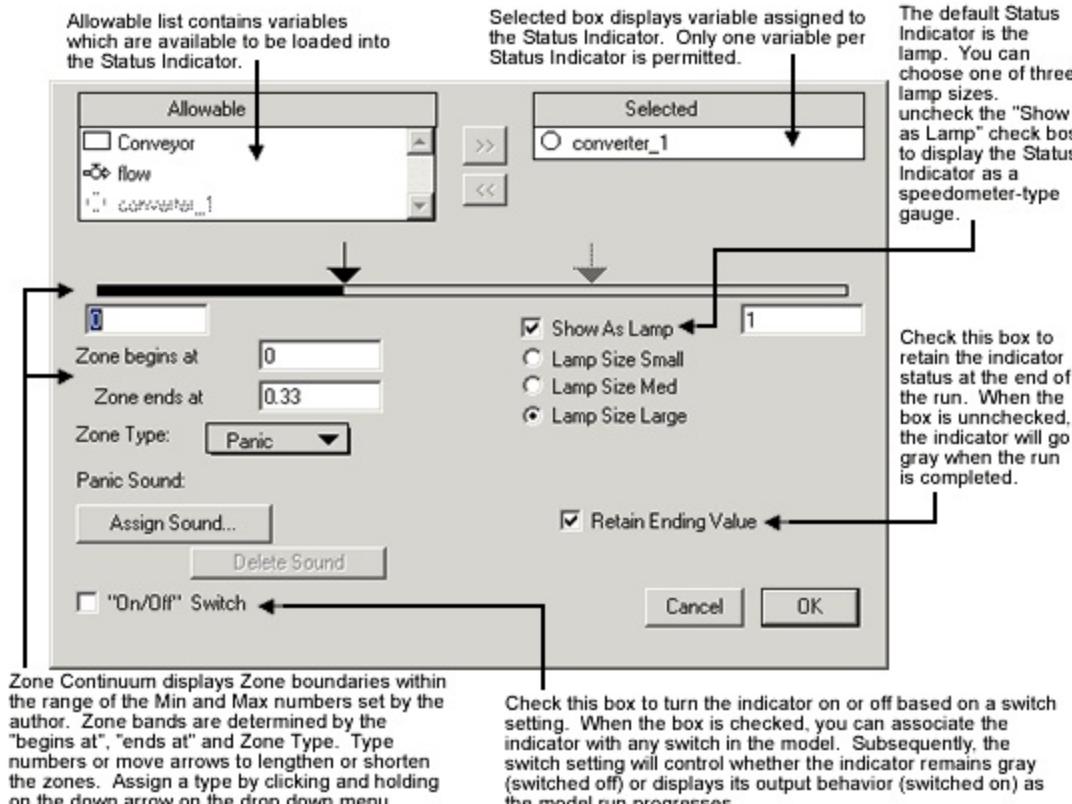
## Surface Operations

To create a Status Indicator, click once to select the icon from the object palette. Move the cursor to the desired location on the interface layer, and click to deposit the device. Double-click on the object to enter its define dialog. The default dialog looks something like what you see in Figure 6-73.

# Dialog Operations

Within the dialog, there are three basic tasks. First, you need to determine whether to display the indicator as a lamp, or as a speedometer. Second, you need to assign a variable to the indicator. Finally, you need to configure the device.

Figure 6-73 Status Indicator Dialog Box



**Lamp vs. Speedometer Display:** Within the dialog, your first decision is whether to display the indicator as a lamp or as a speedometer gauge. The default display option for the status indicator is a lamp. The lamp indicator will be displayed as a smallish circle. The lamp will use color only - green, yellow, or flashing red - to report out the status of the assigned variable. If you uncheck the "Show as Lamp" checkbox within the dialog, the indicator will be displayed as a speedometer-like instrument. The speedometer shows the magnitude of the assigned variable using both a needle gauge and color. In our experience, Lamps do an excellent job of letting users know whether there is a problematic behavior in the model. They also do an excellent job of conserving screen real estate. Speedometer gauges, on the other hand, provide a bit more data to the user at the cost of a larger amount of screen real estate.

**Assigning a variable to the Device:** To assign a variable to the status indicator, double-click on its name in the Allowable list. Or select the variable name and click the >> button. When you do, the variable will be loaded to the

selected list. In addition, as indicated in Figure 6-73, the bottom half of the dialog will become chockablock full of controls and options. You'll use these controls and options to configure the indicator.

**Configuring the Indicator:** Once you've determined whether to display the indicator as a Lamp or as a speedometer, and you have assigned a variable to the indicator, you need to configure the it. There's lots to do here, but fortunately, the basic process is very straightforward. Here's what you need to do:

- *Set the display range for the Status Indicator.* Immediately below the allowable list you will see the "Zone Continuum Bar." This bar stretches from the left to the right of the dialog. Below the bar, at its beginning and end, you'll find number boxes that you can use to set the display range for the status indicator. Use these boxes to set minimum and maximum values for the display range. Does the speedometer go from 0 to 140 mph? Or does it go from 10 to 62 mph? In Figure 6-73, the display range has been set from 0 to 20
- *Configure each zone.* Select a Zone by clicking along the zone continuum bar. When you do, the zone will become highlighted (i.e., will turn black) and the "stakes" at the end(s) of the selected zone will become highlighted. In addition, the numbers in the "Zone begins at" and "Zone ends at" boxes will change to reflect the zone you have selected. Use the numbers to set the boundaries for the selected zone (alternately, drag the stakes to the left or right) Then, use the pop-up Zone Type list to assign Normal (green), Caution (yellow), or Panic (flashing red) to the zone. Assign a sound to the Panic zone if desired.

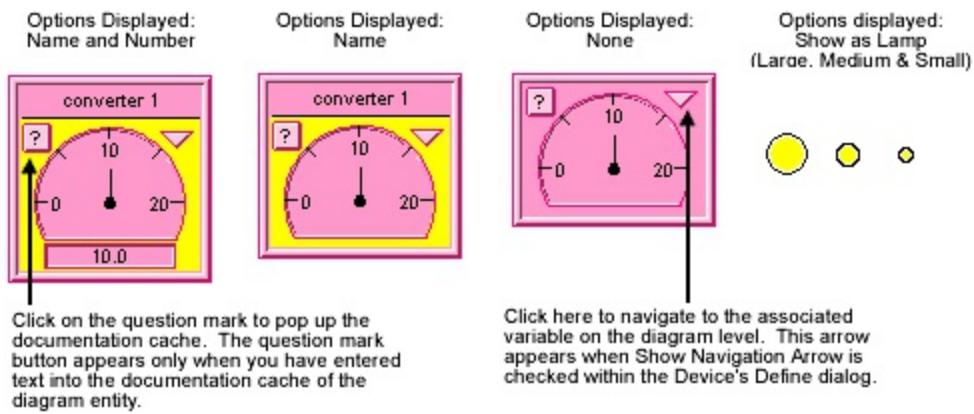
As you work through the configuration process for each zone, think carefully about the manner in which zones are associated with model performance. For example, low cash values might engender "panic." On the other hand, low levels of toxic waste in groundwater might be a very "green" thing to have while high levels of waste might imply a "panic" display. In setting up status indicator, context is key!

- *If desired, associate the Status Indicator with a Switch.* In some instances, you may wish for status indicators to "do their thing" only when switched on by the user of the model. For example, indicators that show the magnitude of wolves in the Greater Yellowstone ecosystem may be relevant *only* after the predators have been introduced into the ecosystem. The optional "On-Off" switch check box gives you this capability. Check the box, and associate the indicator with a switch from the drop-down list. As

users interact with your model, when the switch is in the off position the associated indicator will be grayed out.

- *Complete the configuration.* The remaining controls within the Status Indicator dialog relate to the display of the resultant object on the Interface Layer. Lamps can be configured as small, medium or large in size. Speedometer gauges can display the name of the associated variable, the numeric value associated with the output, and a down-arrow that can take the user directly to the model variable. Finally, retain ending value causes the ending value of the variable to be displayed after the simulation is complete. These different configurations are detailed in Figure 6-75.

*Figure 6-75*  
*Status Indicator Display Options*



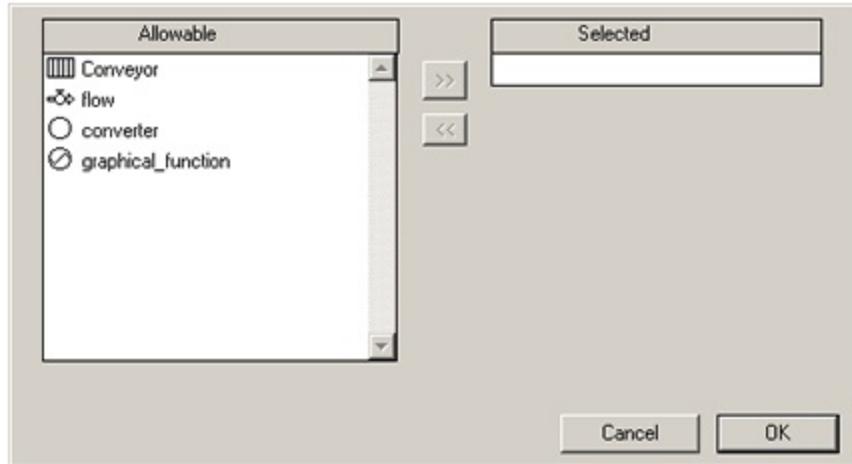
# Numeric Display

On the Interface, Map, and Model layers, you'll use the Numeric Display to display the current output associated with assigned model variables. The Display is useful for getting a precise reading of what's going on in the model, as a simulation unfolds.

**Surface Operations:** Click to select the Numeric Display from the Objects palette. Click once to deposit it on the Diagram. Numeric Displays are not resizable. They can be re-positioned by clicking-and-holding, then dragging them to a new location.

**Dialog Operations:** Double-click to open the Numeric Display's Define dialog. An unassigned display dialog looks like the one in Figure 6-76.

*Figure 6-76 Numeric Display Dialog - Unassigned Variable*



To move a variable from the Allowable to the Selected list, either double-click it, or select it, then click the >> button. This will automatically replace a previously selected variable. To remove the variable in the Selected list, select it and then click the << button.

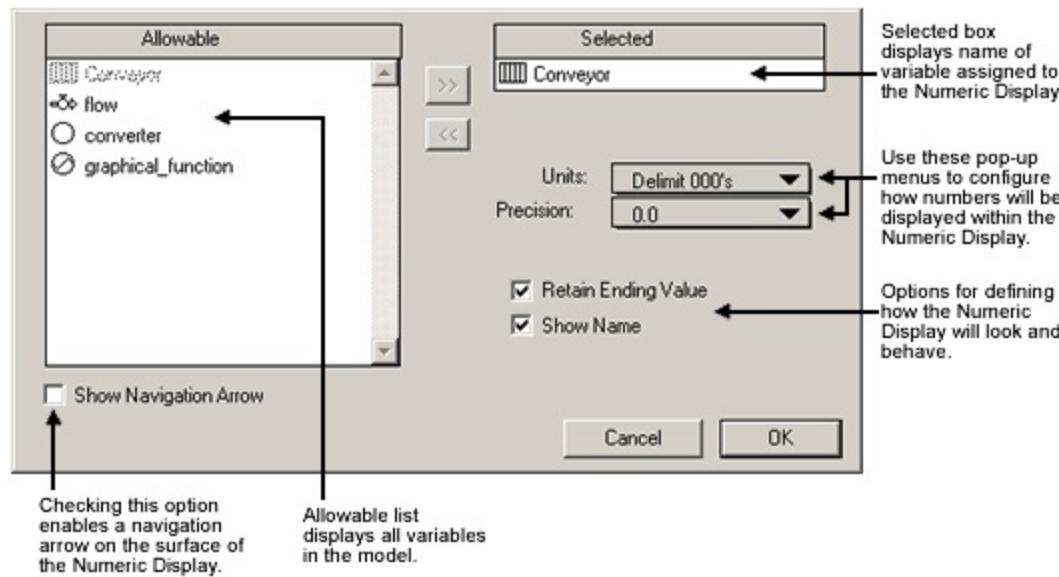
Once you've assigned a variable to the Numeric Display, the dialog will transform itself to look like Figure 6-77.

Below the Selected box are two pop-up formatting menus. Click-and-hold, then choose the formatting option(s) as desired.

A check box in the dialog offers you the ability to have the Numeric Display "Retain Ending Value." If you select this option, the ending value for the variable associated with the Display will continue to be displayed after the simulation has ended. If the option is not elected, the Display will "go blank" at the end of the simulation.

Figure 6-77

## Numeric Display Dialog - Assigned Variables



If you select the "Retain Ending Value" option, and you decide that you'd like to clear the data from Displays, you can do so by choosing Restore Numeric Displays from the Interface menu.

The "Show Name" check box in the dialog allows you to selectively show the name of the variable that has been assigned to the numeric display. When Show Name is not checked, the name of the variable will not appear in the numeric display. You can then place the display next to the model variable to create an elegant diagram-level output device.

The final check box in this dialog is located under the Allowable list. When the "Show Navigation Arrow" check box is checked, a triangular, downward pointing arrow will appear on the right side of the Numeric Display surface. Clicking on this arrow will navigate you down to the display's corresponding variable on the Model layer.

# **Text Box**

The purpose of the Text Box is to hold text. The basic activities associated with Text Boxes are creating, moving, resizing, and typing text. Within each Text Box's dialog you can exercise a variety of controls and options. We'll begin by looking at the activities. We'll end the section by looking at the controls and options in the Text Box dialog.

# Creating, Resizing & Typing Text in Text Boxes

**Creating a Text Box:** Creating a Text Box is easy on the Interface, Map, or Model layer. Simply select the Text Box object from the palette. As you move the cursor onto the diagram, it will change to a Text Box icon. Click once to deposit the Text Box in the desired location. The top-left corner of the Text Box will be in the location of your click. When you release your click, the insertion point cursor will blink at you from the top-left corner of the Text Box, indicating that the Text Box is ready for you to begin typing.

*Tip:* If Sub-model spaces or Decision Process Diamond spaces are open on the Map or Model layer, you will encounter the Ø cursor after you select the Text Box from the palette. To deposit a Text Box on the main level, depress the control key (Windows) or command key (Macintosh). To deposit a Text Box within the Sub-model space, move the cursor just inside the left edge of the space. When the Text Box cursor appears, click to deposit the Text Box.

Frame (a) of Figure 6-78 shows a Text Box immediately after it has been deposited. Frame (b) shows a Text Box in its selected state.

*Figure 6-78 Basic Activities with Text Boxes*

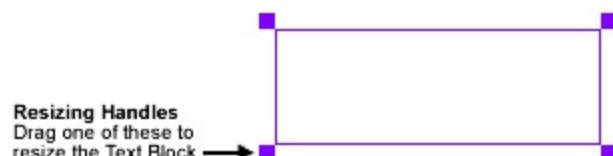
(a) Newly Created Text Block



To Select Block: Click on border. Cursor will change to an arrow when you're in the correct position.  
To Move Box: Click-and-hold border; drag to desired location.  
To Open: Double-click on border. Or, select it, and choose Open Selection... from Interface Menu.

To Edit Text: Move cursor within block and click. Cursor will change to an I-beam when it is in position. Click, drag, and type as you would with a text processor.

(b) Selected Text Block



**Moving a Text Box:** To move a Text Box to a new location, click-and-hold on the Text Box's border. Then, drag the block to its new home.

**Resizing a Text Box:** To resize a Text Box, first select it by clicking along the border. Then click-and-drag on one of the four resizing handles, to make the box larger or smaller as required.

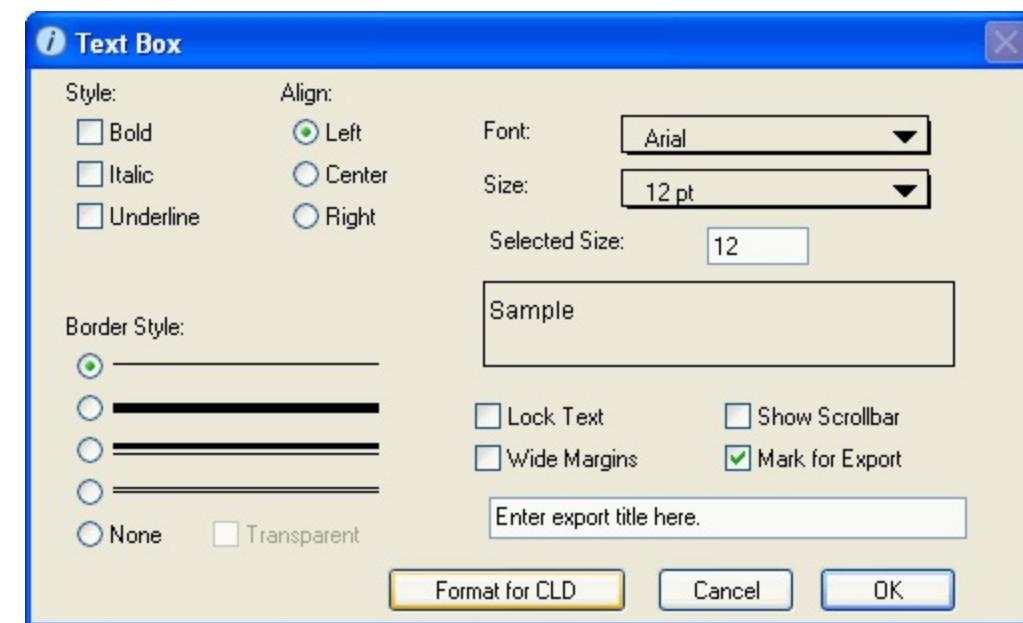
**Typing Text in Text Boxes:** To type text within a Text Box, move the pointer over the box until it changes to an I-beam. Then, click to deposit the cursor

within the box. Click, drag, and type as you would with a text processor.

# Text Box Dialog Operations

By double-clicking along the Text Box's border or by selecting the Text Box and choosing Open Selection from the Interface or Model menu, you'll gain access to the Text Box dialog. As shown in Figure 6-79, this dialog allows you to control the display of the text that appears on the surface of the Text Box.

Figure 6-79  
Text Box Dialog



Use the Text Box dialog box to format the text in the Text Box:

- **Style** – Use the check boxes in this section to specify the text style: **Bold**, **Italic**, or **Underline**.
- **Align** – Use the options in this section to select how the text should be aligned in the Text Box: **Left**, **Center**, or **Right**.
- **Font** – Click this box to select the font type for the text.
- **Size** – Click this box to select the font size (in points) for the text. The size you select is displayed here and in the Selected Size box.
- **Selected Size** – Displays the currently selected font size for the text (in points). You can change the size by typing a font size in this box.
- **Sample** – Displays sample text that shows your current text settings (style, alignment, font type, and size).
- **Border Style** – Select the style of border you want for the Text Box. If you select **None**, the **Transparent** check box becomes enabled. If you select the **Transparent** check box, the Text Box becomes transparent so that you can see what's behind it. Note that only the first line of text is displayed.

while using this option.

- **Lock Text** – Select this check box to lock the text so that it cannot be changed.
- **Wide Margins** – Select this check box to provide extra margin space between the text and the side borders of the Text Box.
- **Show Scrollbar** – Select this check box to add a vertical scroll bar to the Text Box so that you can easily scroll the text up and down if there is more text than can be displayed at once in the Text Box.
- **Mark for Export** – Select this check box to "mark" the text for export. When you select this check box, a new box appears so that you can type a title for the Text Box. Subsequently, when you choose "Export Text Boxes" from the File menu, the contents of all marked Text Boxes will be copied to the clipboard and optionally saved to disk as a text file.

---

*Note:* Marked text will also be included in the report generated by a button that has been assigned the Print Exports action.

- **Format for CLD** – If you are using the Text Box to add labels to a [Causal Loop Diagram](#) (for example, a "(B)" label to indicate a balancing CLD), click this button to automatically reformat the Text Box for use in a Causal Loop Diagram. When you click this button, the Text Box is resized so that it is smaller, the font size is changed to be slightly larger, and the border is set to transparent. If you have not already entered text into the Text Box when you click this button, the software displays a plus (+) sign in the Text Box.

# Graphics Frame

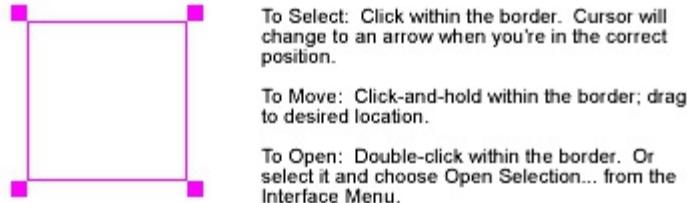
Graphics Frames will spice up your user interface in a number of ways. They are useful in framing objects such as input/output devices and buttons. And they let you import pictures, graphics and movies. This section will discuss the Graphics Frame object's Surface Operations and Basic Operations via the Graphics Frame dialog box.

## Surface Operations:

To create a Graphics Frame, click once on the icon in the palette; select a location on the screen and click once to place the object. The default size of the Graphics frame is determined by the target screen size as defined in the Runtime Options dialog box, which appears when you choose the [Save As Runtime File](#) command from the [File menu](#). A selected Graphics Frame has the appearance shown in Figure 6-80a below.

*Figure 6-80a Graphics Frame Surface Operations*

Click and drag on the Resizing Handles to change the shape / size of the Graphics Frame.



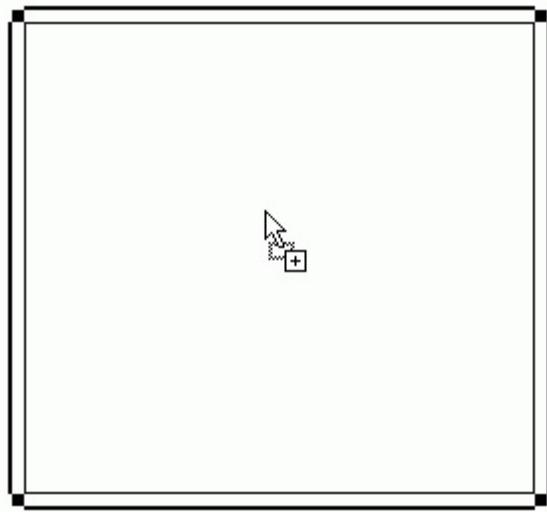
**Moving:** To re-position a Graphics Frame, click-and-hold within the border, and drag to the desired position.

**Resizing:** To resize the Graphics Frame, select it, then drag on one of the Resizing Handles to increase or decrease the size to your specifications. In addition, within the graphics frame dialog, you can resize the graphics frame to full screen for a number of target screen sizes as defined by Runtime Options dialog box, which appears when you choose the [Save As Runtime File](#) command from the [File menu](#).

**Importing Pictures:** To import a picture, drag it from your desktop, a browser, the Finder or Windows Explorer, or any other location on your computer onto the Graphics Frame. When the Graphics Frame is highlighted with a border and the pointer changes to a "+" (as shown in Figure 6-80b), release the mouse button to drop the picture into the Graphics Frame.

*Figure 6-80b*

## *Import Picture with Drag and Drop*



### **Dialog Operations:**

To open the Graphics Frame and access its Dialog Box, select the object and either double-click on the border or choose Open Selection... from the Interface menu. The Dialog Box will appear like the one in Figure 6-81.

**Object Type.** There are two types or "styles" of Graphics Frames: Graphics and 3D Housing. These determine the display of the Graphics Frame only. The basic capabilities of the frame are independent of the object type.

The Graphics incarnation has a 2-dimensional appearance, no fill options and several border styles (including no border at all)-the dialog is shown in Figure 6-81. On the other hand, as shown in Figure 6-82, the 3D Housing has a 3-dimensional appearance, the option to fill or not fill, and no border style choices.

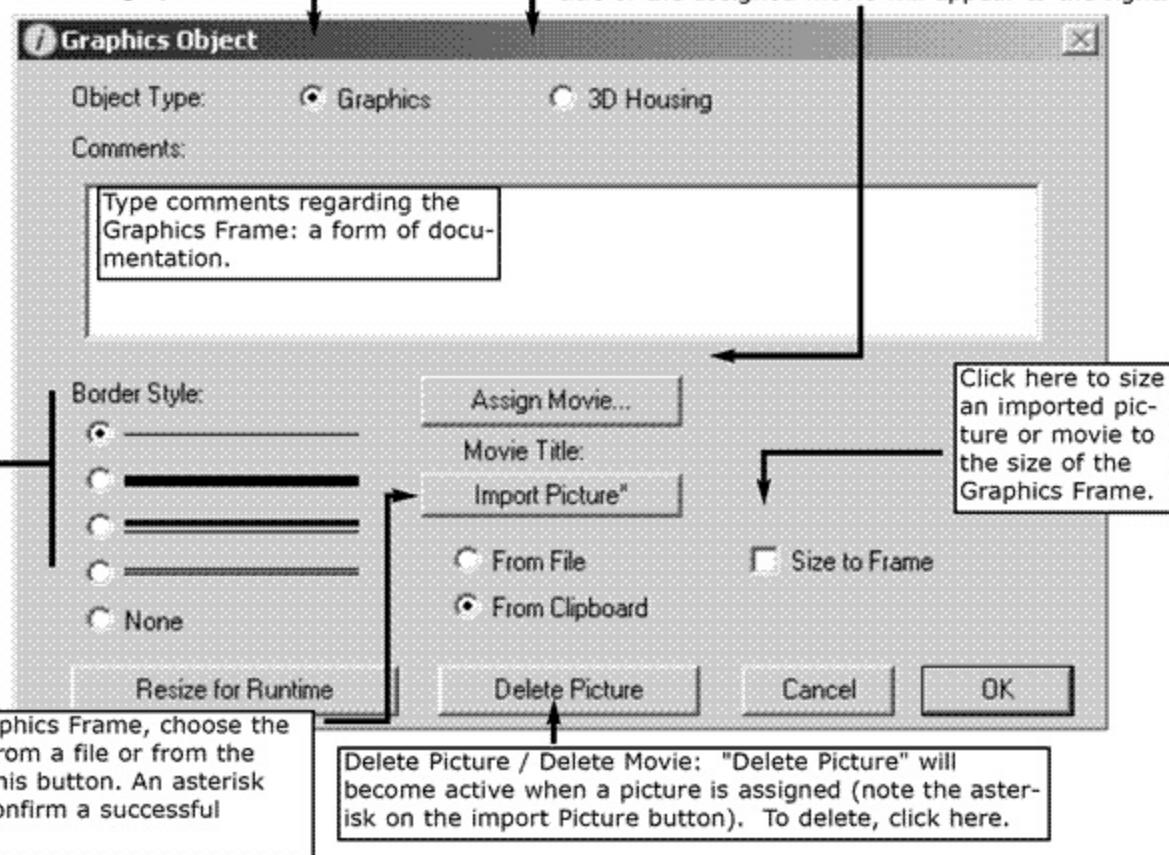
*Figure 6-81  
Graphics Frame Dialog Box with Picture Assigned*

Choose between a standard Graphics Frame, using Border Style settings below, and a 3D Housing which provides a frame with a three dimensional appearance. Figure 6-82 shows the 3D Housing options.

To assign a movie to the Graphics Frame, click once on this button. Locate the file you wish to assign using this standard "Get File" dialog. The title of the assigned movie will appear to the right.

When Graphics is the chosen Object Type, select from four border styles or choose none for no border.

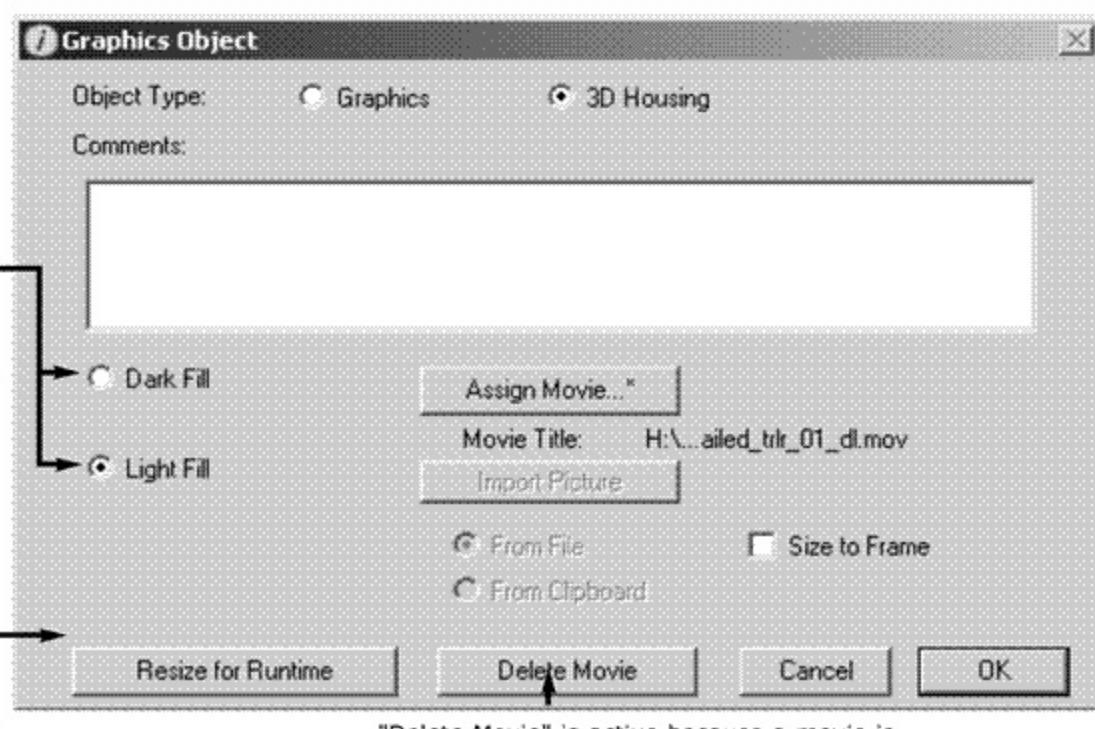
To assign a picture to the Graphics Frame, choose the source of the picture, either from a file or from the clipboard, and click once on this button. An asterisk will appear in the button to confirm a successful import.



**Figure 6-82**  
*Graphics Dialog Box with 3D Hosing Style and Movie Assigned*

When 3D Housing is the selected Object type, choose either Dark Fill or Light Fill to determine the tone of the fill color for the Graphics Frame. Choosing Dark Fill will fill the Graphics Frame with the darker of the color chosen or the 3D accent color. Choosing Light Fill will fill the Graphics Frame with the lighter of the color chosen or the 3D accent color.

Click this button to resize the graphics frame to fill the screen for a number of target screen sizes (e.g. 640x480, 800x600, 1024x768). Doing so is very useful when you are creating a stand-alone application, and you wish to use a single graphic as the backdrop for a screen on the interface or model layer. After you click the Resize for Runtime button, the graphics frame will be resized when you exit the dialog with OK. The resultant size of the graphics frame will be determined by the screen size settings that you are currently using within the Runtime... dialog.



"Delete Movie" is active because a movie is assigned (note the asterisk on the Assign Movie... button). To delete, click here.

**Functionality:** Both Types of Graphics frames permit assignment of movies and importing of pictures.

**Importing Movies:** In Order to assign QuickTime movies to a Graphics Frame, your machine must have QuickTime capability; i.e., your machine must have color and have the QuickTime software installed. The Assign Movie button will appear gray if the QuickTime software is not resident on your computer.

To assign a movie to the Graphics Frame, click on the Assign Movie button visible in Figure 6-82. A standard Get File dialog will appear. Locate and select the movie that you wish to assign. The first frame of the assigned QuickTime movie will be displayed in the center of the Graphics Frame.

Assigning movies to Graphics Frames is very similar to assigning movies to Sector Frames. To read additional details and learn helpful hints about this process, see [Importing QuickTime Movies into Sector Frames](#).

**Importing Pictures:** To import a picture into a Graphics Frame, click on the Import Picture button in the Graphics Frame dialog box. You will view a standard Get File dialog. Locate the graphics file that you want to import into the Graphics Frame. For a list of graphic file formats that can be imported into your graphics frame, see [Graphic File Formats - Windows Only](#).

---

*Tip:* You can also import a picture into a Graphics Frame by dragging and dropping a picture onto the Graphics Frame. For more information, see the [Importing Pictures](#) section under "Surface Operations", above.

Graphics Frames expand the creative possibilities for the model interface. Figure 6-83 shows side-by-side comparison of Graphics Frames using different options that are available. Experiment liberally and don't be afraid to have fun!

*Figure 6-83  
Different Style Options for Graphics Frames*



Graphics Frame  
with Standard Border



3D Housing  
with Dark Fill



Graphics Frame  
with No Border

# Graphic File Formats - Windows Only

The following file formats can be imported into the Button, Graphics Frame, Process Frame, and Sector Frame on Windows when QuickTime is installed:

- Amiga IFF
- BMP
- GIF
- JPEG
- PCX
- Photoshop
- PICT (Raster only)
- Sun Raster
- TARGA
- TIFF
- WMF (Raster only)





# Part 2 - Deeper Details

This portion of the Technical Documentation provides deeper information regarding the operation and features of the software.

- **Builtins** describes each of the Builtin functions that the software makes available to you. For each Builtin, you'll learn what it does, how it works, and when to use it.
- **Importing and Exporting Data** explains how to importing and exporting data in your models by using the Copy and Paste commands and by creating import and export links between the model and one or more Microsoft® Excel worksheets.
- **Presenting Models as Causal Loop Diagrams** describes how to create Causal Loop Diagrams to show only the dominant feedback loops or selected causal connections between entities in your model.
- **Sub-models** provides a set of how-to directions for creating and working with Sub-models on the Map and Model layers. It also describes the rules the software uses for the manipulation of diagram objects, whenever Sub-models or Decision Process Diamonds exist within a model.
- **Cycle-Time** details the cycle-time capabilities of the software. There you will learn how to set up a model to collect cycle-time metrics. You'll also learn a number of ways to use cycle-time metrics to determine system performance.
- **Arrays** provides the necessary details for using arrays to manage complexity in your models. You will find the details needed for setting up one and two dimensional arrays and the internal rules that are enforced by the software.
- **Modules** describes how to build and use modules to create hierarchical models that are constructed out of smaller, self-contained models.
- **Command Line Options** describes the options available to run a model outside of the software using the Windows command line.

This part of the Technical Documentation is designed to be used as reference material. You may want to skim the topics in this part now. Then, when you need an in-depth technical treatment of one of the topics covered, take some time to work through the appropriate section.

# Builtins

This section describes the format and use of the software's Builtin functions. These Builtin functions fall into ten categories: **Test Input, Mathematical, Trigonometric, Logical, Statistical, Financial, Discrete, Cycle-time, Array, and Special Purpose.**

Builtin Name	Category	Builtin Name	Category
<a href="#">ABS</a>	Mathematical	<a href="#">LOOKUP</a>	Special
<a href="#">AND</a>	Logical	<a href="#">LOOKUPXY</a>	Special
<a href="#">ARCCOS</a>	Trigonometric	<a href="#">MAX</a>	Mathematical
<a href="#">ARCSIN</a>	Trigonometric	<a href="#">MEAN</a>	Mathematical
<a href="#">ARCTAN</a>	Trigonometric	<a href="#">MIN</a>	Mathematical
<a href="#">ARRAYMAX</a>	Array	<a href="#">MOD</a>	Mathematical
<a href="#">ARRAYMAXIDX</a>	Array	<a href="#">MONTECARLO</a>	Statistical
<a href="#">ARRAYMEAN</a>	Array	<a href="#">NORMAL</a>	Statistical
<a href="#">ARRAYMIN</a>	Array	<a href="#">NORMALCDF</a>	Statistical
<a href="#">ARRAYMINIDX</a>	Array	<a href="#">NOT</a>	Logical
<a href="#">ARRAYRANK</a>	Array	<a href="#">NPV</a>	Financial
<a href="#">ARRAYSTDDEV</a>	Array	<a href="#">OR</a>	Logical
<a href="#">ARRAYSUM</a>	Array	<a href="#">OSTATE</a>	Discrete
<a href="#">ARRAYVALUE</a>	Array	<a href="#">PAUSE</a>	Special
<a href="#">CAP</a>	Discrete	<a href="#">PCT</a>	Mathematical
<a href="#">CGROWTH</a>	Special	<a href="#">PI</a>	Mathematical
<a href="#">COOKTIME</a>	Discrete	<a href="#">PMT</a>	Financial
<a href="#">COS</a>	Trigonometric	<a href="#">POISSON</a>	Statistical
<a href="#">COSWAV</a>	Trigonometric	<a href="#">PULSE</a>	Test Input
<a href="#">COUNTER</a>	Special	<a href="#">PV</a>	Financial
<a href="#">CTFLOW</a>	Cycle-time	<a href="#">QELEM</a>	Discrete
<a href="#">CTMAX</a>	Cycle-time	<a href="#">QLEN</a>	Discrete
<a href="#">CTMEAN</a>	Cycle-time	<a href="#">RAMP</a>	Test Input
<a href="#">CTMIN</a>	Cycle-time	<a href="#">RANDOM</a>	Statistical
<a href="#">CTSTDDEV</a>	Cycle-time	<a href="#">REWORK</a>	Special

<a href="#">CYCLETIME</a>	Cycle-time	<a href="#">ROUND</a>	Mathematical
<a href="#">DELAY</a>	Discrete	<a href="#">RUNCOUNT</a>	Special
<a href="#">DELAY1</a>	Special	<a href="#">SIN</a>	Trigonometric
<a href="#">DELAY3</a>	Special	<a href="#">SINWAVE</a>	Trigonometric
<a href="#">DELAYN</a>	Special	<a href="#">SMTH1</a>	Special
<a href="#">DERIVN</a>	Mathematical	<a href="#">SMTH3</a>	Special
<a href="#">DT</a>	Special	<a href="#">SMTHN</a>	Special
<a href="#">ELSE</a>	Logical	<a href="#">SOUND</a>	Special
<a href="#">ENDVAL</a>	Special	<a href="#">SQRT</a>	Mathematical
<a href="#">EXP</a>	Mathematical	<a href="#">STARTTIME</a>	Special
<a href="#">EXPRND</a>	Statistical	<a href="#">STEP</a>	Test Input
<a href="#">IF</a>	Logical	<a href="#">STOPTIME</a>	Special
<a href="#">FV</a>	Financial	<a href="#">SUM</a>	Mathematical
<a href="#">FORCST</a>	Special	<a href="#">SWITCH</a>	Special
<a href="#">HISTORY</a>	Special	<a href="#">TAN</a>	Trigonometric
<a href="#">INIT</a>	Special	<a href="#">THEN</a>	Logical
<a href="#">INT</a>	Mathematical	<a href="#">THROUGHPUT</a>	Cycle-time
<a href="#">LOGNORMAL</a>	Statistical	<a href="#">TIME</a>	Special
<a href="#">LOGN</a>	Mathematical	<a href="#">TRANSTIME</a>	Discrete
<a href="#">LOG10</a>	Mathematical	<a href="#">TREND</a>	Special

Documentation for each Builtin function begins with the name of the Builtin, followed by the arguments that are used to define the Builtin. Arguments to Builtins must be enclosed within parentheses ( ). Note that each Builtin function argument is denoted by < >. When there are multiple arguments, they must be separated by a comma. Occasionally, there will be opportunity to include optional arguments when defining the Builtin function. Optional arguments are denoted by square brackets [ ]. Thus, the form of a Builtin function in the following documentation is:

**Builtin(<argument 1>, <argument 2> [,<optional argument>])**

When it is appropriate, an example is provided.

# Test Inputs

This section describes the following Builtins:

- [PULSE](#)
- [RAMP](#)
- [STEP](#)

The software's test input Builtins enable you to conduct controlled experiments on your model. Typically, a PULSE, RAMP or STEP is appended to an inflow or outflow equation. When the Builtin is activated (at the time you specify) it will knock the system out of its previous state. You can then observe how your model responds to this idealized test.

Conducting controlled experiments using idealized inputs such as PULSE, RAMP and STEP is an excellent way to build understanding of the inner workings of your model.

## PULSE(<volume>[,<first pulse>,<interval>])

The PULSE function generates a pulse input of a specified size (*volume*). In using the PULSE function, you have the option of setting the time at which the PULSE will first fire (*first pulse*), as well as the interval between subsequent PULSES (*interval*). Each time that it fires a pulse, the software pulses the specified volume over a period of one time step (DT). Thus, the instantaneous value taken on by the PULSE function is *volume*/DT. Volume can be either variable or constant. First pulse and interval should be specified as constants.

If you do not provide a value for first pulse, the software will generate the first pulse at the outset of your simulation run. If you do not provide an interval, the software will generate subsequent pulses each DT of the model simulation. Setting interval = 0 will yield a single pulse that will not repeat; it is the equivalent of specifying an interval that is greater than the length of the simulation.

Example:

Figure 7-1 shows the structure, equations, and behavior pattern for a simple PULSE stream that accumulates in a stock.

Figure 7-1 Structure and Behavior of PULSE Function

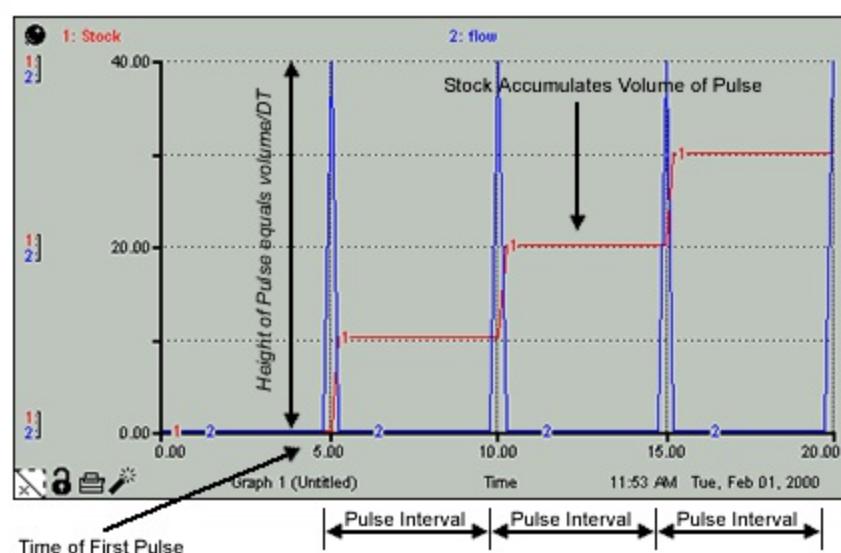
Structure That Accumulates Pulse Stream



where:

Flow = PULSE(10, 5, 5)  
INIT(Stock) = 0  
and  
DT = 0.25

Stock/Flow Pattern Generated By Simple Pulse Stream



## RAMP(<slope>[,<time>])

The RAMP function generates a linearly increasing or decreasing input over time with a specified slope (*slope*). Optionally, you may set the time at which the ramp begins. Slope and time can be either variable or constant.

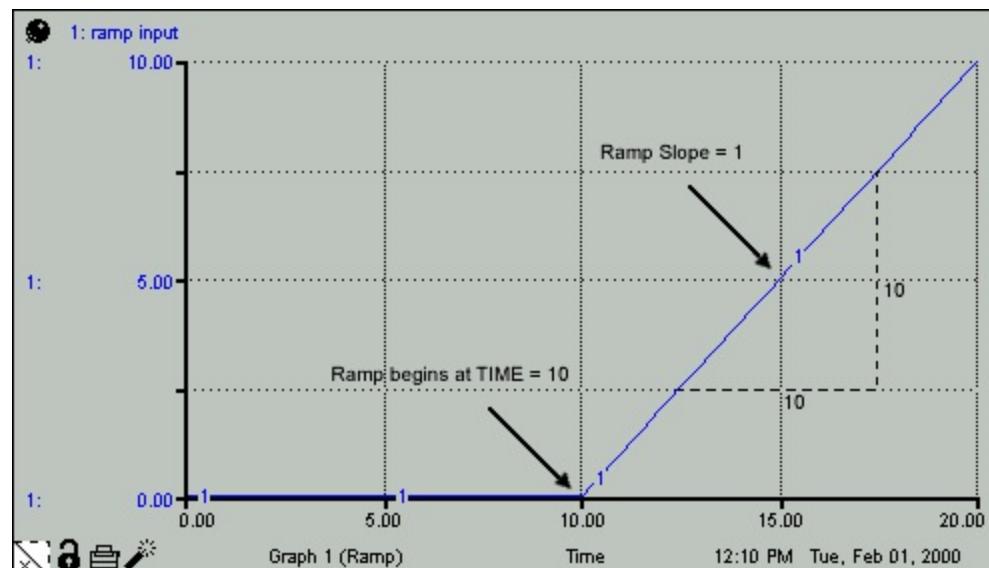
As the simulation progresses, the RAMP will return a 0 before its time to begin has been reached. If you do not set the RAMP's time, it will begin at the outset of the simulation.

Example:

Ramp\_Input = RAMP(1,10) generates the pattern shown in Figure 7-2.

Figure 7-2

Ramp Input With Slope of 1, Beginning at Time 10



## **STEP(<height>,<time>)**

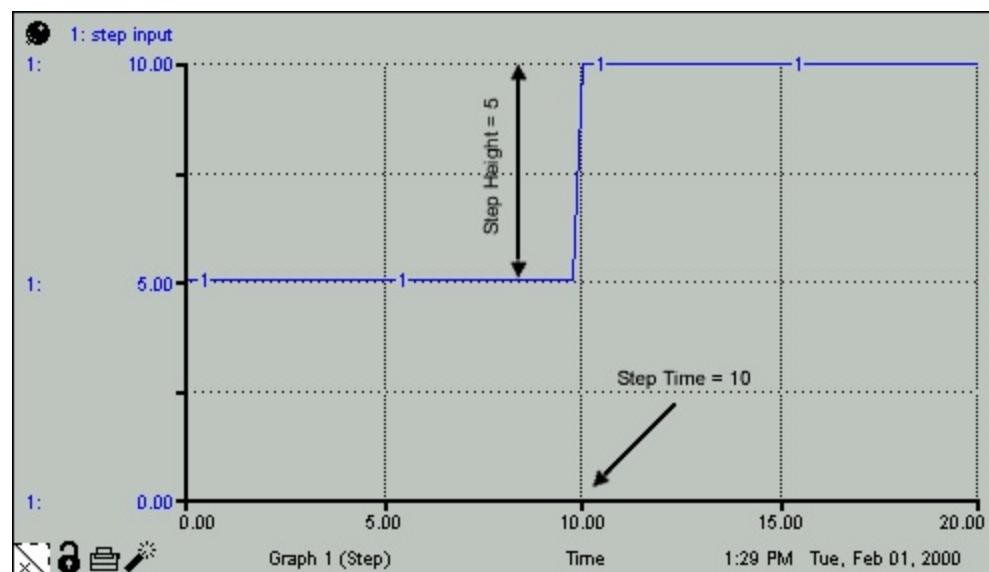
The STEP function generates a one-time step change of specified height (*height*), which occurs at a specified time (*time*). Height and time can be either variable or constant.

Example:

Step\_Input = 5 + STEP(5,10) generates the behavior pattern shown in Figure 7-3.

*Figure 7-3*

*Step Input With Step Height of 5, Occurring at Time 10*



[Related Topics](#)

# Mathematical Functions

This section describes the following Builtins:

- [ABS](#)
- [DERIVN](#)
- [EXP](#)
- [INT](#)
- [LOG10](#)
- [LOGN](#)
- [MAX](#)
- [MEAN](#)
- [MIN](#)
- [MOD](#)
- [PCT](#)
- [PI](#)
- [ROUND](#)
- [SQRT](#)
- [SUM](#)

The mathematical Builtins perform mathematical operations on their input expressions, generating a result. Most commonly, you will use the mathematical functions to define converters which form the micro-logic of your model. Mathematical functions do a wide variety of operations, as detailed below.

## **ABS(<expression>)**

The ABS function returns the absolute value of expression. Expression can be variable or constant.

Example:

ABS(-1) equals 1

## **DERIVN(<input>,< order>)**

The DERIVN function calculates the nth-order time derivative of *input*. Input can be variable or constant. *Order* must be a non-negative integer. The software uses a recursive finite difference technique to calculate time derivatives. The basic equation used in the calculation is:

$$dx/dt = (x_t - x(t-dt))/dt$$

where *t* is current simulation time, and *dt* is DT, the simulation solution interval.

Because the DERIVN function uses previous values of input for its calculations, the function will return an initial value of 0.

Calculus cognoscenti will note that it is possible to use the chain rule to calculate  $dy/dx$ , where *y* and *x* are time-dependent model variables. Simply use the formula:

$$dy/dx = dy/dt \div dx/dt$$

Examples:

DERIVN(1,0) equals 1 (the zeroth-order derivative)

DERIVN(1,1) equals 0 (the first-order derivative)

DERIVN(SINWAVE(1,1),3) calculates the third-order derivative of a sine wave with period of 1 and amplitude of 1.

## **EXP(<expression>)**

The EXP function gives e raised to the power of *expression*. *Expression* can be variable or constant. The base of the natural logarithm, e is also known as Euler's number. e is equal to 2.7182818... EXP is the inverse of the LOGN function (natural logarithm). To calculate powers of other bases, use the exponentiation operator (^).

Examples:

EXP(1) equals 2.7182818 (the value of Euler's constant)

EXP(LOGN(3)) equals 3

2^3 equals 8 (2 raised to the third power)

## **INT(<expression>)**

The INT function gives the largest integer less than or equal to *expression*.  
*Expression* can be variable or constant.

Because of the internal calculations associated with the Runge-Kutta methods, the INT function will not always return integer values when you use the 2nd- or 4th-order Runge-Kutta computation methods. When your model constructs rely on the INT function, be sure to use Euler's method.

Examples:

INT(8.9) equals

INT(-8.9) equals -9

## **LOG10(<expression>)**

The LOG10 function gives the base 10 logarithm of *expression*. *Expression* can be variable or constant. For LOG10 to be meaningful, *expression* must evaluate to a positive value. LOG10 is the inverse of the letter E in scientific notation, or base 10 exponentiation (10 raised to the power of *expression*).

Examples:

LOG10(10) equals 1

LOG10(1E5) equals 5

LOG10(8)/LOG10(2) equals 3 (the base 2 logarithm of 8)

LOG10(-10) returns ? (undefined for non-positive numbers)

## **LOGN(<expression>)**

The LOGN function calculates the natural logarithm of *expression*. *Expression* can be variable or constant. Natural logarithms use Euler's constant e, 2.7182818..., as a base. For LOGN to have meaningful results, expression must be positive. LOGN is the inverse of the EXP function, e raised to the power of *expression*.

Examples:

LOGN(2.7182818) equals 1

LOGN(EXP(3)) equals 3

LOGN(8)/LOGN(2) equals 3 (the base 2 logarithm of 8)

LOGN(-10) returns ? (undefined for non-positive numbers)

## **MAX(<expression>,<expression>,...)**

The MAX function gives the maximum value among the expressions contained within parentheses.

Because of the internal calculations associated with the Runge-Kutta methods, the MAX function will not always return the expected value when you use the 2nd- or 4th-order Runge-Kutta computation methods, and your inputs to the MAX calculation are variables. When your model constructs rely on the MAX function, be sure to use Euler's method.

Example:

Purchases = MAX(0,Desired\_Purchases) returns the larger value among Desired Purchases and 0. In this example, the MAX function will prevent Purchases from taking on negative values.

## **MEAN(<expression>,<expression>,...)**

The MEAN function returns the arithmetic mean of the expressions contained within parentheses. The expressions can be numbers or model variables.

Examples:

MEAN(1,1,1,1,1,1) equals 1

MEAN(1,2,3,4,5,6,7,8,9) equals 5

## **MIN(<expression>,<expression>,...)**

The MIN function gives the minimum value among the expressions contained within parentheses.

Because of the internal calculations associated with the Runge-Kutta methods, the MIN function will not always return the expected value when you use the 2nd- or 4th-order Runge-Kutta computation methods, and your inputs to the MIN calculation are variables. When your model constructs rely on the MIN function, be sure to use Euler's method.

Example:

Spending = MIN(Desired\_Spending,Allowable\_Spending) returns the smaller value among Desired Spending and Allowable Spending.

## **MOD(<expression>,<modulus>)**

The MOD function computes the remainder (modulo) when *expression* is divided by the *modulus*.

Examples:

Excess\_Components=MOD(Components,components\_per\_machine)  
calculates the number of components that will remain in inventory after machines have been assembled.

Time\_of\_Day=MOD(TIME,24) converts simulation time (in hours) to a 24-hour metric of time. Each 24 hours of simulation time, Time of Day will reset itself to 0. See the [COUNTER](#) Built-in for a more generalized way to create a cyclical clock based on simulation time.

## **PCT(<fraction>)**

The PCT function gives the value of *fraction*, expressed as a percentage.  
*Fraction* can be variable or constant.

Examples:

PCT(.65) equals 65

PCT(1.22) equals 122

## PI

The PI function gives the number 3.14159..., an approximation of the constant  $\pi$ .

Example:

`10*SIN(2*\pi*TIME/12)`

generates a sinusoidal fluctuation with an amplitude of 10 and a period of 12. See also [SIN](#) and [COS](#) (see [Trigonometric Functions](#)).

## **ROUND(<expression>)**

The ROUND function rounds *expression* to its nearest integer value.

Because of the internal calculations associated with the Runge-Kutta methods, the ROUND function will not always return integer values when you use the 2nd- or 4th-order Runge-Kutta computation methods. When your model constructs rely on the ROUND function, be sure to use Euler's method.

Examples:

ROUND(9.4) equals 9

ROUND(9.64) equals 10

## **SQRT(<expression>)**

The SQRT function gives the square root of *expression*. *Expression* can be variable or constant. For meaningful results, expression must be greater than or equal to 0.

Examples:

SQRT(144) returns 12

SQRT(-242) returns ? (undefined for negative numbers)

## **SUM(<expression>,<expression>,...)**

The SUM function returns the arithmetic summation of the expressions contained within parentheses.

Examples:

SUM(1,2,3,4,5,6,7,8,9) equals 45

SUM(a,b,c,d,e,f) equals (a+b+c+d+e+f)

 [Related Topics](#)

# Trigonometric Functions

This section describes the following Builtins:

- [ARCCOS](#)
- [ARCSIN](#)
- [ARCTAN](#)
- [COS](#)
- [COSWAVE](#)
- [SIN](#)
- [SINWAVE](#)
- [TAN](#)

Trigonometric Builtins perform basic trigonometric calculations. The trigonometric functions can be useful in modeling scientific problems (for example, kinematics in two or three dimensions). They also can be helpful if you wish to drive an input to the model using a periodically fluctuating signal (e.g., sine or cosine wave).

## **ARCCOS (<expression>)**

The ARCCOS function gives the arccosine of *expression*. The arccosine is the angle in radians whose cosine is *expression*. The angle will be in the range 0 to  $\pi$ . To convert measurement between degrees and radians, use the identity:  $\pi$  (radians) = 180 (degrees). Expression can be constant or variable.

Examples:

ARCCOS(0.5) equals 1.047 (radians)

ARCCOS(0.5)\*180/ $\pi$  equals 60 (degrees)

## **ARCSIN (<expression>)**

The ARCSIN function gives the arcsine of *expression*. The arcsine is the angle in radians whose sine is *expression*. The angle will be in the range  $-\pi/2$  to  $\pi/2$ . To convert measurement between degrees and radians, use the identity:

$\pi$  (radians) = 180 (degrees). Expression can be constant or variable.

Examples:

ARCSIN(0.866) equals 1.047 (radians)

ARCSIN(0.866)\*180/ $\pi$  equals 60 (degrees)

## **ARCTAN(<expression>)**

The ARCTAN function gives the arctangent of *expression*. The arctangent is the angle in radians whose tangent is *expression*. The angle will be in the range  $-\pi/2$  to  $\pi/2$ . To convert measurement between degrees and radians, use the identity:

$\pi$  (radians) = 180 (degrees). Expression can be constant or variable.

Examples:

ARCTAN(1) equals 0.785 ( $\pi/4$  radians)

ARCTAN(1)\*180/ $\pi$  equals 45 (degrees)

## **COS(<radians>)**

The COS function gives the cosine of *radians*, where *radians* is an angle in radians. To convert measurement between degrees and radians, use the identity:  $\pi$  (radians) = 180 (degrees). *Radians* can be constant or variable.

Examples:

COS(1.047) equals 0.5

COS(60\*( $\pi/180$ )) equals 0.5

## **COSWAVE(<amplitude>,<period>)**

The COSWAVE function returns a time-dependent cosine wave, with the specified *amplitude* and *period*. To generate the cosine wave, the COSWAVE function will use the absolute value of the amplitude you specify. To produce meaningful wave results, choose a DT that is significantly smaller than the period of the wave. A DT equal to a quarter of period gives triangle waves. A smaller DT gives results which better approximate a continuous curve.

Examples:

COSWAVE(10,5) generates a cosine wave with an amplitude of 10 unit and a period of 5 time units.

COSWAVE(-5,2) generates a cosine wave with an amplitude of 5 units and a period of 2 time units.

## SIN(<radians>)

The SIN function gives the sine of *radians*, where *radians* is an angle in radians. To convert measurement between degrees and radians, use the identity:  $\pi$  (radians) = 180 (degrees). *Radians* can be constant or variable.

Examples:

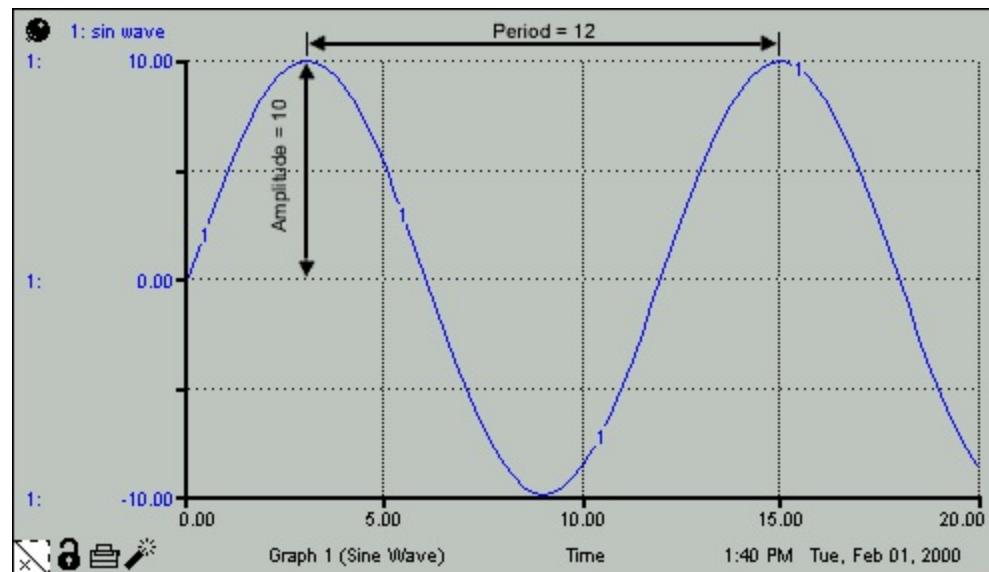
SIN(1.047) equals 0.866

SIN(60 $\pi$ /180) equals 0.866

`sin_wave = 10*SIN(2 $\pi$ *TIME/12)` generates a sinusoidal fluctuation with an amplitude of 10 and a period of 12, as shown in Figure 7-4.

Figure 7-4

Sine Wave With Amplitude of 10 and Period of 12



## **SINWAVE(<amplitude>,<period>)**

The SINWAVE function returns a time-dependent sine wave, with the specified *amplitude* and *period*. The SINWAVE function will use the absolute value of the amplitude you specify to generate the sine wave. To produce meaningful wave results, choose a DT that is significantly smaller than the period of the wave. A DT equal to a quarter of the period gives triangle waves. A smaller DT gives results which better approximate a continuous curve.

Examples:

SINWAVE(10,5) generates a sine wave with an amplitude of 10 units and a period of 5 time units.

SINWAVE(-5,2) generates a sine wave with an amplitude of 5 units and a period of 2 time units.

## TAN(<radians>)

The TAN function gives the tangent of *radians*, where *radians* is an angle in radians. To convert measurement between degrees and radians, use the identity:  $\pi$  (radians) = 180 (degrees). Radians can be constant or variable.

Examples:

TAN(0.785) equals 1

TAN(45\* $\pi$ /180) equals 1

 [Related Topics](#)

# Logical Functions

This section describes the following Builtins:

- AND
- ELSE
- IF
- NOT
- OR
- THEN

The logical functions - IF, THEN, ELSE, AND, OR, NOT - are used to create expressions, and then give values based upon whether the resulting expressions are TRUE or FALSE. When you have multiple conditions, the expressions to be evaluated must be enclosed in parentheses ( ).

When using IF-THEN-ELSE logic to generate 0 - 1 switches or flags, be sure to use Euler's method. Because of the internal calculations associated with the Runge-Kutta methods, the IF-THEN-ELSE constructs designed to return integer values will not always do so when you use the 2nd- or 4th-order Runge-Kutta computation methods.

Examples:

Bonus\_Payments = IF(TIME=5) OR (Sales>5000) THEN Bonus ELSE 0

This statement sets Bonus Payments to the value of Bonus at simulated time 5, or whenever the value of Sales is greater than 5000. When neither condition is met, the statement gives the value 0. Sales and Bonus are defined elsewhere in the model.

Capital\_Acquisition = IF (IRR > = 0.1) AND (Cash > = 1000000) THEN 100000 ELSE 0

This statement sets Capital Acquisition at the value 100000 when IRR (internal rate of return) is greater than or equal to 0.1 and Cash is greater than or equal to 1000000. When one of the conditions is not met, Capital Acquisition is set to 0.

Thermo\_Switch = IF (Temperature < Set\_Point) THEN 1 ELSE 0

This statement sets Thermo Switch (a thermostatic switch on a furnace) at 1 - the "on" position - when Temperature is less than Set Point. The statement sets Thermo Switch at 0 - the "off" position - otherwise.

Thermo\_Switch = IF NOT(Temperature > = Set\_Point) THEN 1 ELSE 0

This statement sets Thermo Switch at 1 when Temperature is not greater than or equal to Set Point. The statement sets Thermo Switch at 0 otherwise.

The software is capable of supporting extremely complex conditional constructs. You should be careful, however, if you find a single equation becoming larger than the size of the equation box. Complex conditional statements can be exceedingly difficult to "debug." They also can be difficult to explain to your colleagues. Good modeling practice dictates that you divide such complex statements into logical groupings, using a converter to evaluate each.

 [Related Topics](#)

# Statistical Functions

This section describes the following Builtins:

- [EXPRND](#)
- [LOGNORMAL](#)
- [MONTECARLO](#)
- [NORMAL](#)
- [NORMALCDF](#)
- [POISSON](#)
- [RANDOM](#)

The statistical Builtins generate sequences of random numbers, each conforming to its specific distribution. The statistical functions thus enable you to introduce randomness into your models.

The software will sample from the distribution you have specified, each DT in the model simulation. Unless you have specified a seed for the distribution as a non-negative integer, the sample set drawn from the distribution will not be replicable.

## **EXPRND(<lambda>[,<seed>])**

The EXPRND function generates a series of exponentially distributed random numbers with a mean of *lambda*. EXPRND samples a new random number in each iteration (that is, each DT) of a model run. If you wish to replicate the stream of random numbers, specify *seed* as an integer between 1 and 32767. To replicate a simulation, all variables that use statistical functions must have seeds specified. Each variable using a statistical function should use a separate seed value.

### Examples:

EXPRND (1) gives a non-replicable exponentially distributed stream of numbers (mean=1)

EXPRND (2,456) gives a replicable exponentially distributed stream of numbers (mean=2)

## **LOGNORMAL(<mean>,<std>[,<seed>])**

The LOGNORMAL function generates a series of random numbers that conform to a Log-Normal distribution (that is, the log of the independent variable follows a normal distribution) with a specified *mean* and *standarddeviation*. LOGNORMAL samples a new random number in each iteration of a simulation. If you wish to replicate the stream of random numbers, specify *seed* as an integer between 1 and 32767. To replicate a simulation, all variables that use statistical functions must have seeds specified. Each variable using a statistical function should use a separate seed value.

Examples:

LOGNORMAL(0,1) gives a non-replicable stream of numbers following a log-normal distribution (mean=0, standard deviation=1)

LOGNORMAL(0,1,147) gives a replicable stream of numbers following a log-normal distribution (mean=0, standard deviation=1)

LOGNORMAL(5,1,12) gives a replicable stream of numbers following a log-normal distribution (mean=5, standard deviation=1)

LOGNORMAL(10,5,102) gives a replicable stream of numbers following a log-normal distribution (mean=10, standard deviation=5)

## **MONTECARLO(<probability>[,<seed>])**

The MONTECARLO function randomly generates a series of zeros and ones, based on the *probability* you have provided. The *probability* is the percentage probability of an event happening per unit of simulation time. *Probability* can be a variable or a constant, but should evaluate to a number between 0 and 100 (numbers outside the range will be set to 0). Note that the *probability* you specify, divided by 100, is the mean number of occurrences per unit time.

MONTECARLO is equivalent to the following logic:

```
IF (RANDOM(0,100,<seed>) <= probability*DT THEN 1 ELSE 0
```

MONTECARLO samples a new random number in each iteration of a model. If you wish to replicate the stream of random numbers, specify *seed* as an integer between 1 and 32767. To replicate a simulation, all variables that use statistical functions must have seeds specified. Each variable using a statistical function should use a separate seed value.

Examples:

MONTECARLO(20,1577) gives a replicable stream of numbers which are equal to 1 20% of the time and equal to 0 the remaining 80% of the time. This means that the mean number of occurrences generated by this distribution is 20/100, or 0.2 per unit time.

## **NORMAL(<mean>,<std>[,<seed>])**

The NORMAL function generates a series of normally distributed random numbers with a specified *mean* and *standard deviation*. NORMAL samples a new random number in each iteration of a simulation. If you wish to replicate the stream of random numbers, specify *seed* as an integer between 1 and 32767. To replicate a simulation, all variables that use statistical functions must have seeds specified. Each variable using a statistical function should use a separate seed value.

Examples:

NORMAL(0,1) gives a non-replicable normally distributed stream of numbers (mean=0, standard deviation=1)

NORMAL(0,1,147) gives a replicable normally distributed stream of numbers (mean=0, standard deviation=1)

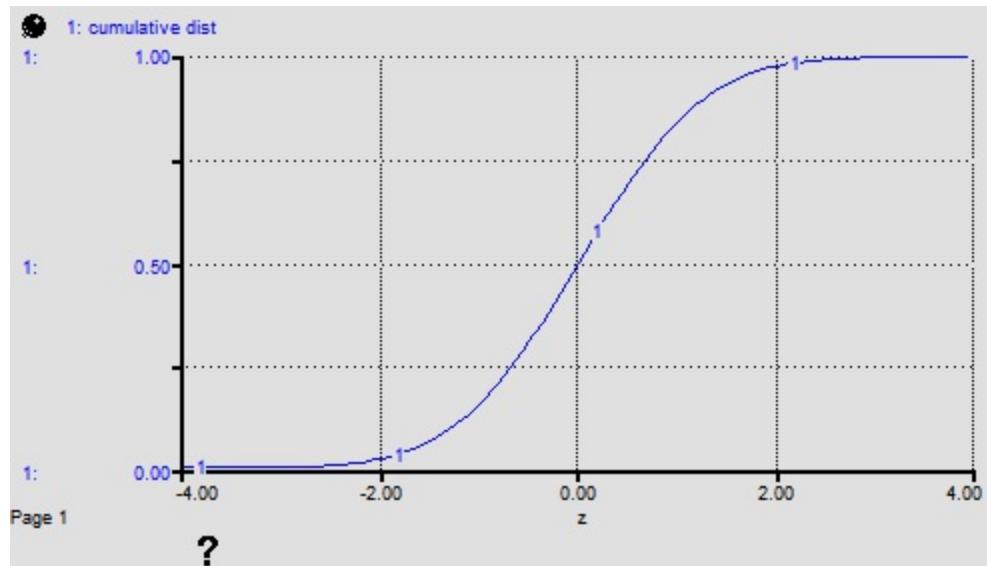
NORMAL(5,1,12) gives a replicable normally distributed stream of numbers (mean=5, standard deviation=1)

NORMAL(10,5,102) gives a replicable normally distributed stream of numbers (mean=10, standard deviation=5)

## **NORMALCDF(<left>, <right>[, <mean>, <stddev>])**

The NORMALCDF calculates the cumulative Normal distribution function between the specified z-scores, or, when the mean and standard deviation are given, between two data values. The cumulative distribution function integrates (sums) all values of the Normal probability distribution function between the left and right endpoints. To sum from negative infinity, use a z-score of -99. To sum to positive infinity, use a z-score of 99.

The graph below shows the behavior of the NORMALCDF function for  $-4 = z = 4$ .



The equation `NORMALCDF(-99, TIME)` was used to make this graph. Note that the NORMALCDF very closely approximates the logistic function, or S-shaped growth. It can be used to formulate, using an equation, several types of common graphical functions:

S-shaped growth – Use the entire function

Exponential growth – Use the left side of the function

Saturation curve – Use the right side of the function

---

*Note:* A z-score can be calculated with the formula:  $(\text{data\_value} - \text{mean})/\text{stddev}$ .

## **POISSON(<mu>[,<seed>])**

The POISSON function generates a series of random numbers that conform to a Poisson distribution. The mean per unit time for the distribution is given by *mu*. POISSON samples a new random number in each iteration of a model run (i.e., every DT). If you wish to replicate the stream of random numbers, specify *seed* as an integer between 1 and 32767. To replicate a simulation, all variables that use statistical function must have seeds specified. Each variable using a statistical function should use a separate seed value.

### Examples:

POISSON (1) gives a non-replicable Poisson sample set with mean of 1, per unit of simulation time.

POISSON (10, 256) gives a replicable Poisson sample set with mean of 10, per unit of simulation time.

## **RANDOM(<min>,<max>[,<seed>])**

The RANDOM function generates a series of uniformly distributed random numbers between *min* and *max*. RANDOM samples a new random number in each iteration of a model run. If you wish to replicate the stream of random numbers, specify *seed* as an integer between 1 and 32767. To replicate a simulation, all variables that use statistical functions must have seeds specified. Each variable using a statistical function should use a separate seed value.

### Examples:

RANDOM(0,1) creates a non-replicable uniformly distributed stream of numbers between 0 and 1.

RANDOM(0,1,147) creates a replicable uniformly distributed stream of numbers between 0 and 1.

RANDOM(0,10,12) gives a replicable uniformly distributed stream of numbers between 0 and 10.

RANDOM(10,13,12) gives a replicable uniformly distributed stream of numbers between 10 and 13.

# Sampling from Non-uniform Distributions

In many cases you may wish to draw samples from distributions other than the ones explicitly provided by the software. A straightforward technique, known as inversion, allows you to transform a sample drawn from RANDOM into any distribution whose density function can be integrated and then inverted.

For example, it is possible to follow this technique to use the RANDOM function to generate Weibull variates. The expression below provides the transformation:

$$\text{Weibull} = b * (-\text{LOGN}(\text{RANDOM}(0,1,1575)))^{(1/c)} + a$$

where

$a \geq 0$  the location parameter

$b > 0$  the scale parameter

$c > 0$  the shape parameter

Transformations of this ilk can be quite useful, but are best left to the analytically inclined. For an in-depth coverage of techniques for sampling from non-uniform distributions, see: Naylor, T.H., J. Balintfy, D.S., and K. Chu, Computer Simulation Techniques, Wiley, 1966.

 [Related Topics](#)

# Financial Functions

This section describes the following Builtins:

- [FV](#)
- [PMT](#)
- [PV](#)
- [NPV](#)

**FV(<rate>,<nper>,<pmt>,<pv>)**

**PMT(<rate>,<nper>,<pv>,<fv>)**

**PV(<rate>,<nper>,<pmt>,<fv>)**

Present value (*pv*), future value (*fv*), periodic payment (*pmt*), number of periods (*nper*), and interest rate per period (*rate*) are standard financial parameters in cash flow problems involving constant payments. The functions FV, PMT, and PV calculate future value, periodic payment, and present value based upon the values of the other financial parameters.

Use the following cash flow conventions in depicting PV, FV, and PMT: cash received is represented by a positive number, cash paid out is represented by a negative number. See the example below for an illustration. Rate and nper must refer to the same period. For example, if nper is the number of months, then rate must be the effective monthly interest rate. Finally, for each of the Built-in financial functions, the first payment is assumed to occur at the end of the first period.

The software uses the following equations to solve for one financial argument in terms of the others:

$$pv * (1 + rate)^{nper} + (pmt / rate) * ((1 + rate)^{nper} - 1) + fv = 0$$

(for *rate*  $\neq$  0)

$$pv + pmt * nper + fv = 0$$

(for *rate* = 0)

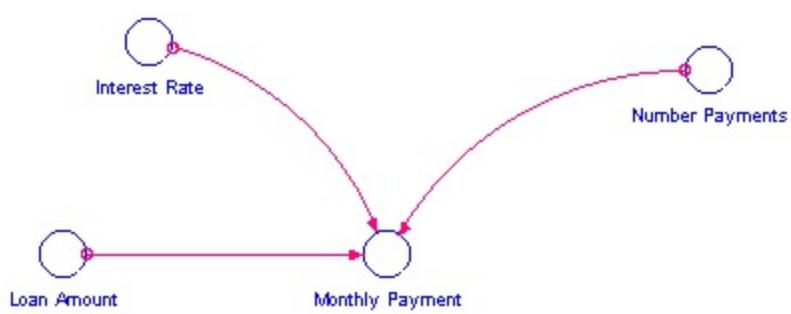
Example:

You received an automobile loan of \$10,000 at an effective monthly interest rate of 1%. You must repay the loan in 36 monthly payments made at the end of each month. What are your monthly payments?

Your structural diagram, and the corresponding equations are shown in Figure 7-5.

The software returns the value for Monthly Payment as -332.14, indicating that the payment is an expense.

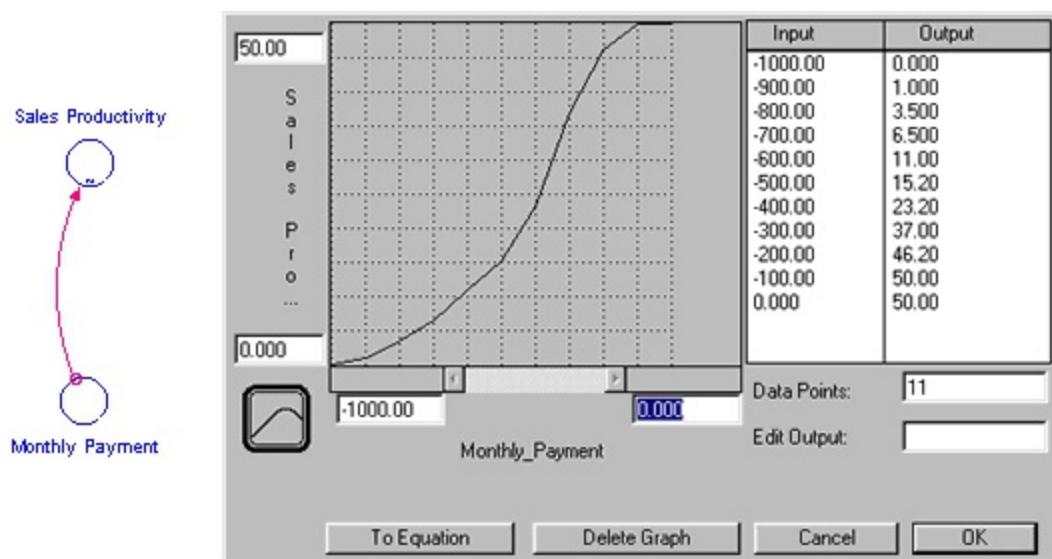
*Figure 7-5 Calculating Monthly Payments With the PMT Function*



$\text{Monthly Payment} = \text{PMT}(\text{Interest\_Rate}, \text{Number\_Payments}, \text{Loan\_Amount}, 0)$   
 $\text{Interest\_Rate} = 0.01$   
 $\text{Number\_Payments} = 36$   
 $\text{Loan\_Amount} = 10000$

In this example, financial functions are used to make a one-time, spreadsheet-like evaluation. Typically you will use the software to continuously evaluate financial parameters based on other variables within your model. For example, you might want to relate monthly payments to monthly sales of an auto dealership. You could do this via a graphical function, as shown in Figure 7-6. Your model thus would continuously show the effects of changing financial parameters on the productivity of the sales staff.

*Figure 7-6  
Relating Monthly Payments to Sales Productivity*

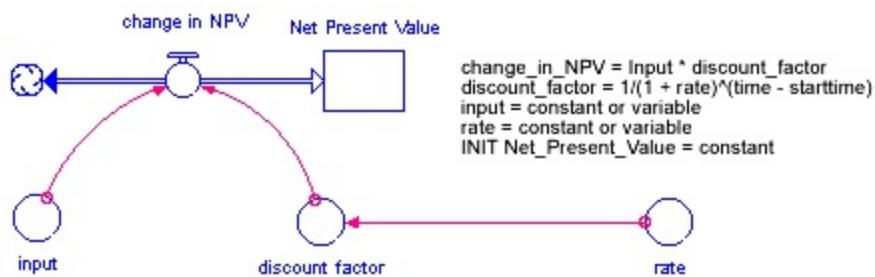


## **NPV(<input>,<rate>[,<initial>])**

The NPV function calculates the net present value of a stream of *input*, using a discount rate of *rate*. *Rate* refers to the effective per-period discount rate for your model simulation. The value returned by NPV is the present value (at the outset of your simulation run) of the stream of inputs from model start time to a given simulation time. You may specify an optional *initial* value initial for the calculation, otherwise the initial value of input will serve as the initial value.

The NPV function, when used in a converter, will produce the same results as the stock in the structure and equations shown in Figure 7-7. The discount factor is normalized to the Time Specs setting for start time so that the initial condition of the denominator evaluates to a value of 1.

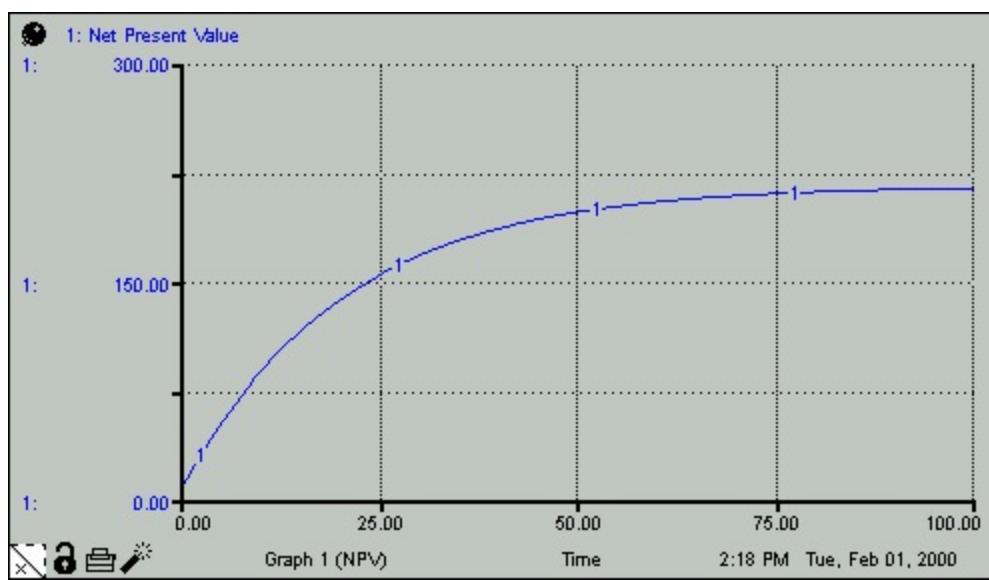
*Figure 7-7*  
*Structure of NPV Calculation*



Examples:

*Net\_Present\_Value* = NPV(10, .05) generates the pattern shown in Figure 7-8. The values for Net Present Value represent the present value - at time 0 - of the constant stream of benefits, given the effective per-period discount rate of 5%.

*Figure 7-8*  
*Net Present Value Calculation*



[Related Topics](#)

# Discrete Functions

This section describes the following Builtins:

- [CAP](#)
- [COOKTIME](#)
- [DELAY](#)
- [OSTATE](#)
- [QELEM](#)
- [QLEN](#)
- [TRANSTIME](#)

The majority of the discrete functions provide information about the internal workings of the Queue, Conveyor and Oven stock types. These Builtins (CAP, COOKTIME, OSTATE, QELEM, QLEN, TRANSTIME) tell you what a particular discrete stock is doing, as the simulation progresses. The DELAY function, on the other hand, simply delays its input signal by a fixed amount of time.

## **CAP(<oven>) or CAP(<conveyor>)**

The CAP function returns the capacity associated with a conveyor or an oven, where *conveyor* is defined as a "Conveyor" stock type, and *oven* is defined as an "Oven" stock type. Often, you will want to use an Oven's capacity as an input to other model variables. Similarly, you may want to use a Conveyor's capacity as an input to other model variables. The CAP function allows you to accomplish these tasks.

Example:

Capacity = CAP(Mixer) will return the capacity you have specified for the Oven Mixer.

## **COOKTIME(<oven>)**

The COOKTIME function samples the cook time associated with an oven, where *oven* is a discrete "Oven" stock. Often, you will want to know the cook time of a given Oven in your model. The COOKTIME function allows you to do so. It samples and holds the Oven's cook time, while the Oven is cooking. While the oven is filling and emptying, COOKTIME returns a value of 0.

Example:

`processing_time= COOKTIME(Processor)` will sample the cook time being used for the Oven Processor.

## **DELAY(<input>,<delay duration>[,<initial>])**

The DELAY function returns a delayed value of *input*, using a fixed lag time of *delay* duration and an optional initial value *initial* for the delay. If you do not specify an initial value *initial*, DELAY assumes the value to be the initial value of input. If you specify delay duration as a variable, the DELAY function will use the initial value for its fixed lag time.

Example:

Shipments = DELAY(Orders,5) will cause shipments to lag behind orders by 5 time units. For the first 5 time units of the simulation, the delay will return the initial value for orders - since no initial value was specified.

## **OSTATE(<oven>)**

The OSTATE function returns the status of an Oven, where *oven* is a discrete "Oven" stock. If the Oven is filling, OSTATE returns a "0." If it is "cooking," OSTATE returns a "1." If it is "emptying," OSTATE returns a "2." When the Oven is simultaneously emptying current contents, and filling with new contents, OSTATE will return a "2". For more information about Ovens, see [Stocks](#).

Example:

Status = OSTATE(Mixer) will return the status of Mixer, where Mixer is a discrete Oven stock.

**QELEM(<queue>,<element>)**

-or-

## **QELEM(<conveyor>,<element>)**

The QELEM function returns the value contained in the specified element, or "slot," of the specified *queue* or *conveyor* discrete stock. Elements are numbered sequentially, beginning with the "next" element to flow out of the Queue or Conveyor. For more information on Queues and Conveyors, see [Stocks](#).

Example:

Next\_to\_be\_served = QELEM(Service\_Queue,1) returns the value of the first element in the discrete stock Service Queue.

**QLEN(<queue>)**

-or-

## **QLEN(<conveyor>)**

The QLEN function returns the total number of elements or slots contained in the specified *queue* or *conveyor* discrete stock. For more information on Queues and Conveyers, see [Stocks](#).

Example:

Line\_Length = QLEN(Checkout\_Queue) will return the length - in terms of the number of orders - in the discrete stock Checkout Queue.

## **TRANSTIME(<conveyor>)**

The TRANSTIME function returns the transit time associated with the flow of material that is currently entering a conveyor, where *conveyor* is a discrete "Conveyor" stock type. The TRANSTIME function thus tells you how much time will be spent in the Conveyor, by stuff that is currently entering the Conveyor. When the inflow goes to zero, the TRANSTIME function will hold the value of transit time from the most recent inflow.

Example:

`Transit_time = TRANSTIME(Work_in_Process)` will return the transit time associated with material that is currently entering the Conveyor Work in Process.

 [Related Topics](#)

# Cycle-time Functions

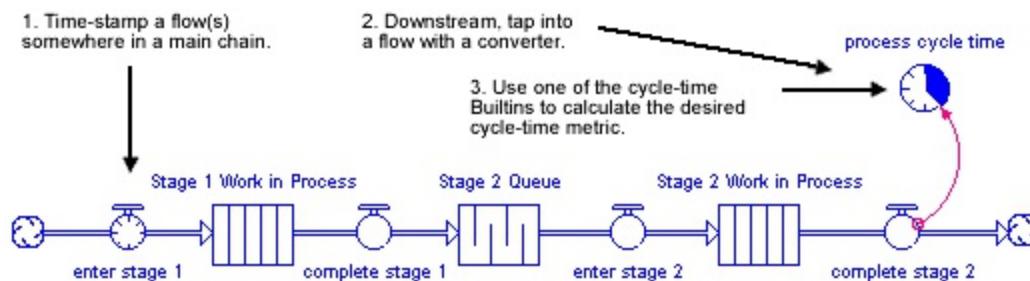
This section describes the following Builtins:

- [CTFLOW](#)
- [CTMAX](#)
- [CTMEAN](#)
- [CTMIN](#)
- [CTSTDDEV](#)
- [CYCLETIME](#)
- [THROUGHPUT](#)

Imagine that you and three of your colleagues are asked to run an obstacle course. You start at 1:00 PM, and complete the course at 1:35 PM. The first of your colleagues begins at 1:05 PM, and completes the course at 1:42 PM. The second colleague begins at 1:10 PM, and, being somewhat more athletic than you, completes the course at 1:35. The third starts at 1:15 PM, gets stuck somewhere, and exits the course at 2:15 PM. Because you know the start and completion time for each person (or batch) in the group, it's possible to calculate a variety of useful cycle-time metrics.

The cycle-time functions work in an analogous fashion. They rely on the time-stamping of flows at some point in a conserved-flow process, either on the Map or Model layer or within a Sub-model space. Downstream, flows can subsequently provide this time-stamping information to the Builtins, which in turn will calculate metrics regarding the time that material has spent in process, since it was last time-stamped. Figure 7-9 provides an overview of how to introduce cycle-time metrics into your models. For more information and examples of cycle-time configurations, see [Introduction to Cycle-Time](#).

*Figure 7-9 An Overview of Cycle-time Capabilities*

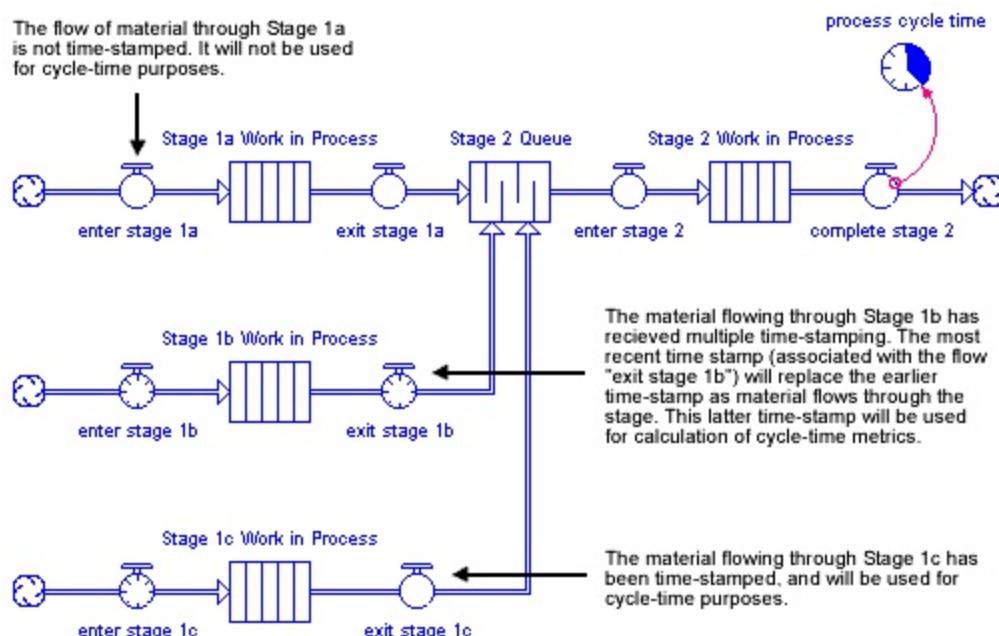


As suggested by the diagram, the procedure associated with making a cycle-time calculation is straightforward. First, time-stamp a flow (or a set of flows)

at some point in a stock/flow chain (to time-stamp a flow, check the check box next to the small stopwatch in the flow's dialog). Then, determine from which flow, downstream in the same stock/flow chain, you wish to calculate the cycle-time metric(s). Use a connector and a converter to tap into the downstream flow. Then, define the converter using the appropriate cycle-time Builtin.

The cycle-time Builtins are accessible only within converter dialogs, when a connection runs from a downstream flow to the converter, and one or more upstream flows have been time-stamped. The resultant cycle-time calculations apply only to the material coming from those upstream flows which have been time-stamped. Whenever the material gets time-stamped at more than one flow within a chain of stocks and flows, the downstream cycle-time calculation will be based on the most recent time-stamping. When multiple inflows enter a conserved flow chain, cycle-time calculations will be based on all flows which have been time-stamped. Figure 7-10 shows these rules in graphical form. The text which follows details the capability of each cycle-time function.

*Figure 7-10  
Rules of Cycle-time - Multiple Flows*



## **CTFLOW(<flow>[,<initial>])**

The CTFLOW function returns the volume of flow associated with time-stamped batches of material moving through the *flow*. CTFLOW can be important to report when, in a conserved flow chain, flows which have been time-stamped are mixed with flows which have not been time-stamped. When this mixing occurs, the resulting downstream flow volume will be greater than the volume of flow used for cycle-time calculations. CTFLOW will tell you how much of the flow volume has been time-stamped, and thus is being used for cycle-time purposes.

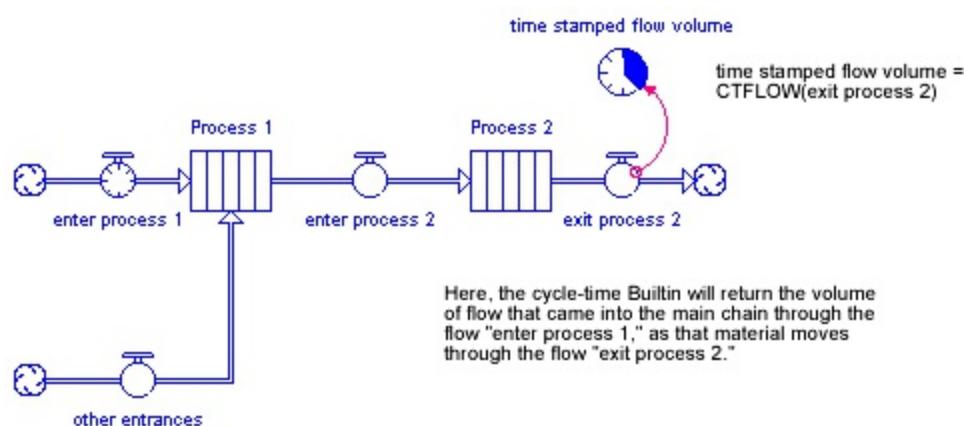
The optional *initial* parameter will allow you to provide an initial volume for the associated flow. If *initial* is not specified, CTFLOW will return a zero for the initial value.

When you are operating in the Normal simulation mode (in the Time Specs dialog, under the Run menu), or when time stamping has been turned off for all upstream flows in a conserved flow chain, CTFLOW will return a zero. Any initial values of material in a conserved stock/flow chain will not be time-stamped, and hence will not be used for cycle-time purposes.

Example:

In Figure 7-11, the converter time stamped flow volume calculates the time-stamped flow volume associated with stuff that is moving through the flow exit process 2.

*Figure 7-11  
Illustrating the Use of the CTFLOW Builtin*



## **CTMAX(<flow>[,<initial>, <instantaneous>])**

The CTMAX function returns the maximum cycle-time associated with time-stamped batches of material moving through a flow. CTMAX will operate in one of two distinct modes, depending upon whether the optional instantaneous argument is set to 1. When instantaneous is not specified (or set to zero), CTMAX will return the maximum cycle time associated with time-stamped batches of material which have moved through a flow, over the course of a simulation. When instantaneous is set to 1, CTMAX will return the maximum cycle-time associated with time-stamped material which is currently moving through the flow.

The optional *initial* parameter is applicable only when instantaneous is not specified (or set to zero). In this case, the initial value will be the value that is returned by the function until the time-stamped portion of the flow volume becomes positive. If initial is not specified, CTMAX will return a zero until the time-stamped portion of the flow volume is positive.

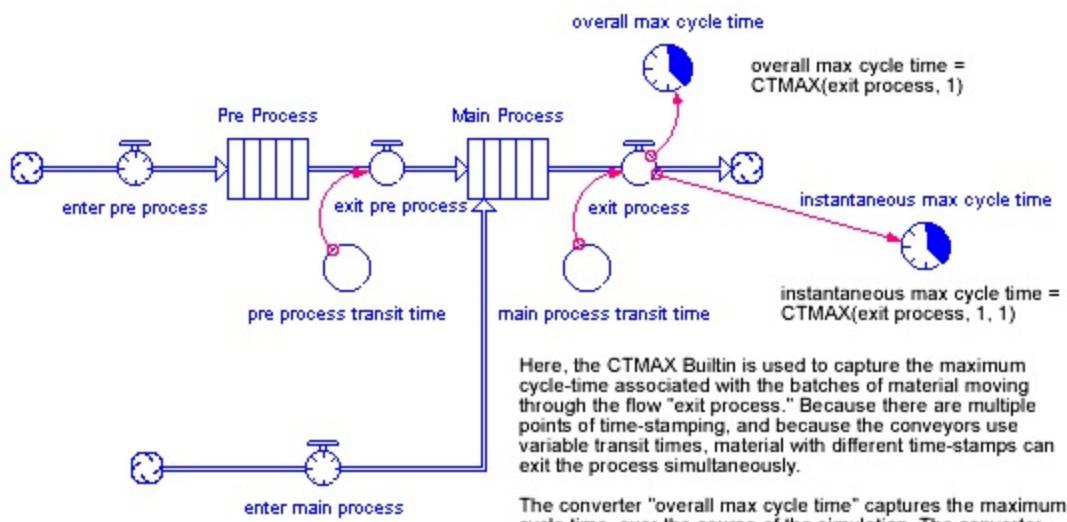
CTMAX becomes particularly relevant when more than one time-stamped flow enters a conserved flow chain, or when different batches get mixed together somewhere in a process (for example, in a Conveyor with variable transit time). In either case, material with different time-stamps can exit the process simultaneously. CTMAX will provide you with the maximum cycle-time associated with stuff which is moving through the flow.

When you are operating in the Normal simulation mode (in the Time Specs dialog, under the Run menu), or when time stamping has been turned off for all upstream flows in a conserved flow chain, CTMAX will return a zero. Any initial values of material in a conserved stock/flow chain will not be time-stamped, and hence will not be used for cycle-time purposes. When operating in its *instantaneous* mode, CTMAX will return a zero when the time-stamped portion of the flow volume is zero.

Example:

In Figure 7-12, the converter overall max cycle time calculates the maximum cycle-time of the stuff which is moving through the flow exit process, over the course of the simulation. The converter instantaneous max cycle time calculates the maximum cycle-time of the batches which are currently moving through exit process. Note that time-stamped batches enter the process at two entry points, and that the conveyors in the process each have variable transit times.

*Figure 7-12  
Illustrating the Use of the CTMAX Built-in*



Here, the CTMAX Builtin is used to capture the maximum cycle-time associated with the batches of material moving through the flow "exit process." Because there are multiple points of time-stamping, and because the conveyors use variable transit times, material with different time-stamps can exit the process simultaneously.

The converter "overall max cycle time" captures the maximum cycle time, over the course of the simulation. The converter "instantaneous max cycle time" captures the maximum cycle time for the batches that are flowing through "exit process," as the simulation progresses.

In each case, the initial value returned by the Builtin is 1.

## **CTMEAN(<flow>[,<initial>,<weighted>])**

The CTMEAN function returns the cumulative average cycle-time, since the start of a run, associated with the time-stamped portion of the flow of material moving through a conserved flow chain. CTMEAN will perform its cumulative average cycle-time calculation in one of two distinct ways, depending upon whether the optional *weighted* argument is set to 1.

When *weighted* is not specified, CTMEAN calculates the algebraic (i.e., per batch) cycle-time, over the course of the simulation. When *weighted* is not specified, the CTMEAN calculation thus is independent of the volume of the time-stamped flow. Consider a simulation in which two batches pass through a process. The first has a volume of 10 and a cycle-time of 1, while the second has a volume of 2 and a cycle-time of 5. If *weighted* is not specified, the resultant CTMEAN calculation is:

$$(1 + 5) / 2 = 3 \text{ (time units per batch)}$$

On the other hand, when the optional argument *weighted* is set to 1, the resultant cycle-time calculation will be weighted by the magnitude of the flow volumes. Thus, the weighted CTMEAN calculation provides a metric of the cycle time per unit of flow volume. Using the same two batches, a weighted CTMEAN calculation would give:

$$(10 * 1 + 2 * 5) / (10 + 2) = 1.67 \text{ (time units per unit volume)}$$

The optional *initial* value is the value that is returned by the Built-in function until the time that the time-stamped flow volume first becomes positive. When the initial value is not specified, the Built-in will initially return a 0. Specifying an initial value is useful when initializing a continuous system to steady-state.

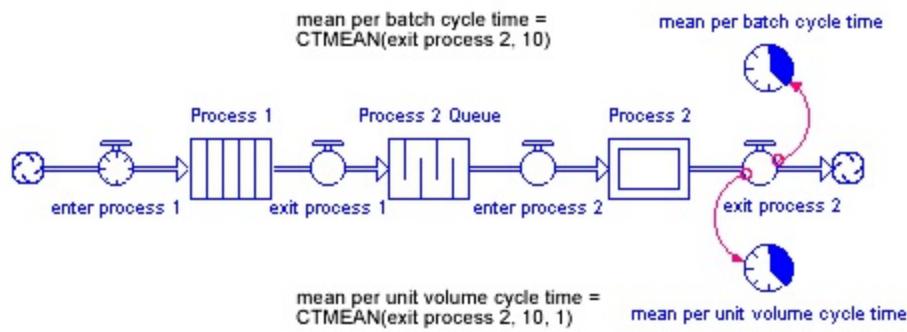
When you are operating in the Normal simulation mode (in the Time Specs dialog, under the Run menu), or when time stamping has been turned off for all upstream flows in a conserved flow chain, CTMEAN will return a zero. Initial values of material in a conserved stock/flow chain will not be time-stamped, and hence will not be used for cycle-time purposes.

### Example:

In Figure 7-13, the converter mean per batch cycle time calculates the cumulative average cycle-time (i.e., the average time per batch, since the start of the simulation run, spent in Process 1 and Process 2) for material which has entered the Process 1 stock through the flow enter process 1. mean per unit volume cycle time calculates the cumulative average cycle-time on a unit volume basis. Both calculations initially return a value of 10.

Figure 7-13

Illustrating the Use of the CTMEAN Builtin



## **CTMIN(<flow>[,<initial>,<instantaneous>])**

The CMIN function returns the minimum cycle-time associated with time-stamped batches of material moving through a *flow*. CMIN will operate in one of two distinct modes, depending upon whether the optional *instantaneous* argument is set to 1. When *instantaneous* is not specified, CMIN will return the minimum cycle time associated with time-stamped batches of material which have moved through a flow, over the course of a simulation. When *instantaneous* is set to 1, CMIN will return the minimum cycle time associated with time-stamped material which is currently moving through the flow.

The optional *initial* parameter is applicable only when *instantaneous* is not specified (or set to zero). In this case, the initial value will be the value that is returned by the function until the time-stamped flow volume becomes positive. If *initial* is not specified, CMIN will return a zero until the time-stamped portion of the flow volume is positive.

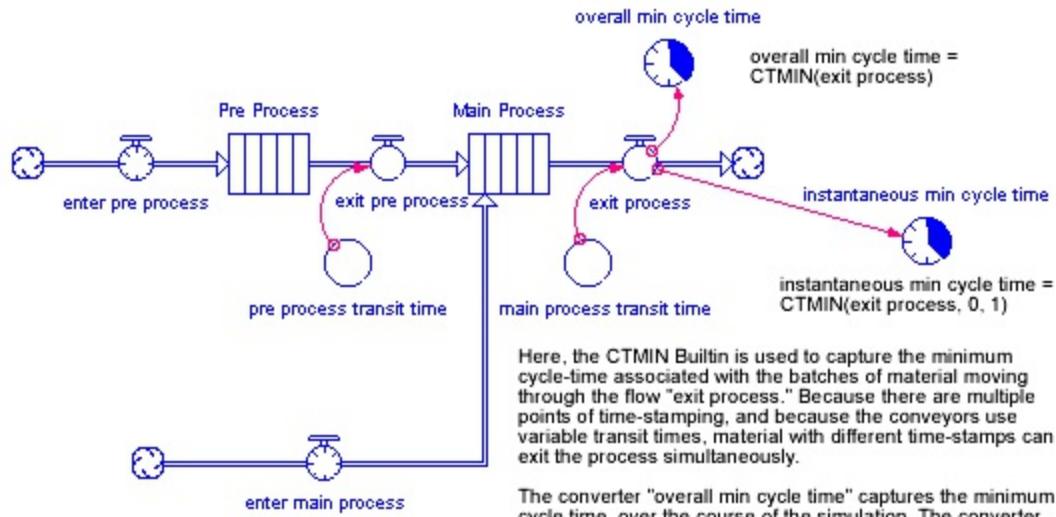
CTMIN becomes particularly relevant when more than one time-stamped flow enters a conserved flow chain, or when different batches get mixed together somewhere in a process (for example, in a Conveyor with variable transit time). In either case, material with different time-stamps can exit the process simultaneously. CMIN will provide you with the minimum cycle-time associated with stuff which is moving through the flow.

When you are operating in the Normal simulation mode (in the Time Specs dialog, under the Run menu), or when time stamping has been turned off for all upstream flows in a conserved flow chain, CMIN will return a zero. Any initial values of material in a conserved stock/flow chain will not be time-stamped, and hence will not be used for cycle-time purposes. When operating in its instantaneous mode, CMIN will return a zero when the time-stamped portion of the flow volume is zero.

### **Example:**

In Figure 7-14, the converter overall min cycle time calculates the minimum cycle-time of the stuff which is moving through the flow exit process, over the course of the simulation. The converter instantaneous min cycle time calculates the minimum cycle-time of the batches which are currently moving through exit process. Note that time-stamped batches enter the process at two entry points, and that the conveyors in the process each have variable transit times.

*Figure 7-14  
Illustrating the Use of the CMIN Built-in*



Here, the CTMIN Builtin is used to capture the minimum cycle-time associated with the batches of material moving through the flow "exit process." Because there are multiple points of time-stamping, and because the conveyors use variable transit times, material with different time-stamps can exit the process simultaneously.

The converter "overall min cycle time" captures the minimum cycle time, over the course of the simulation. The converter "instantaneous min cycle time" captures the minimum cycle time for the batches that are flowing through "exit process," as the simulation progresses.

In each case, the CTMIN function returns an initial value of 0.

## **CTSTDDEV(<flow>[,<weighted>,<initial>])**

The CTSTDDEV function returns the standard deviation in cycle-times, since the start of the simulation run, associated with time-stamped material moving through the *flow*. Only non-zero cycle-times are used for the CTSTDDEV calculation. The calculation assumes that cycle-times are normally distributed. When two or more time-stamped batches move simultaneously through a flow, the software uses the average cycle-time in its standard deviation calculation.

The CTSTDDEV function is particularly useful when you want to obtain a measure in the variability of cycle-times, over the course of a simulation. CTSTDDEV performs its calculations in one of two distinct ways, depending upon whether the optional *weighted* argument is set to 1.

When *weighted* is not specified, CTSTDDEV calculates an algebraic, non-weighted metric of standard deviation. This metric is independent of the magnitude of time-stamped flow volume. It looks only at the cycle-time associated with the material moving through the flow. In this sense, the non-weighted CTSTDDEV provides you with a measure of the per-batch variability in cycle-times.

On the other hand, when the *weighted* argument is set to a value of 1, CTSTDDEV weights its calculations by the magnitude of time-stamped flow volumes. Thus, the weighted CTSTDDEV calculation provides a metric of the variability in cycle time per unit volume.

The optional *initial* parameter will allow you to provide an initial value for the standard deviation of the cycle-time. This initial value will be returned until the first time-stamped units pass through the flow. If no initial value is specified, CTSTDDEV will return a zero until the time-stamped portion of the flow volume is positive.

When you are operating in the Normal simulation mode (in the Time Specs dialog, under the Run menu), or when time stamping has been turned off for all upstream flows in a conserved flow chain, CTSTDDEV will return a zero. Any initial values of material in a conserved stock/flow chain will not be time-stamped, and hence will not be used for cycle-time purposes.

---

*Note:* Because the average cycle-time of material moving through a flow is used in the standard deviation calculation, the standard deviation reported by the CTSTDDEV Built-in may vary slightly from a distribution set upstream.

## **CYCLETIME(<flow>[,<weighted>,<initial>])**

The CYCLETIME function returns the cycle-time associated with time-stamped batches of material moving through the *flow*. CYCLETIME thus tells you how much time material has spent "in process," since it was last time-stamped. Whenever more than a single time-stamped flow enters a conserved flow chain, or when different batches get mixed together somewhere in a process (for example, in a Conveyor with variable transit time), material with different time-stamps can exit the process simultaneously. In these cases, what's returned by CYCLETIME will depend upon whether you have specified the optional *weighted* argument to a value of 1.

When *weighted* is not specified, the CYCLETIME Built-in returns a simple algebraic measure of cycle-time. This measure can be thought of as a per batch metric for cycle-time, since it is independent of the volume of time-stamped flow, whenever material with different time-stamps simultaneously moves through the flow. Consider a simulation in which two batches exit a process simultaneously. The first has a volume of 10 and a cycle-time of 6. The second has a volume of 100 and a cycle-time of 50. If *weighted* is not specified, the CYCLETIME calculation would result in a value of:

$$(6 + 50) / 2 = 28 \text{ (time units per batch)}$$

On the other hand, when the optional argument *weighted* is set to 1, the CYCLETIME Built-in will weight its calculation by the magnitude of the respective flow volumes in the combined batch. Thus, the weighted CYCLETIME calculation provides a metric of the cycle-time per unit of flow volume. Using the same two batches, a weighted CYCLETIME calculation would give:

$$(10 * 6 + 100 * 50) / (10 + 100) = 46 \text{ (time units per unit volume)}$$

The optional *initial* parameter will allow you to provide an initial value for the cycle-time. This initial value will be returned until the first time-stamped units pass through the flow. If no initial value is specified, CYCLETIME will return a zero until the time-stamped portion of the flow volume is positive.

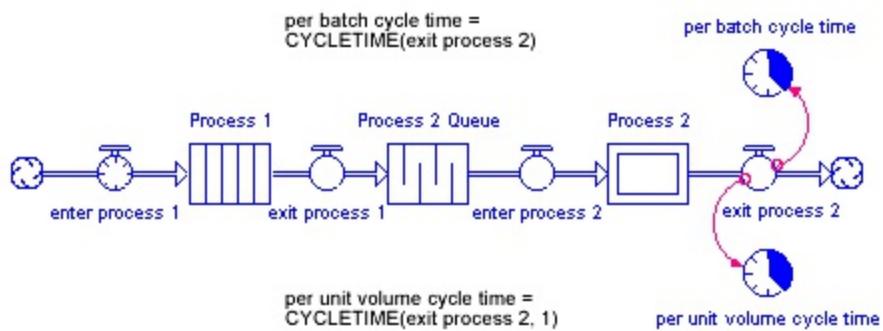
When you are operating in the Normal simulation mode (in the Time Specs dialog, under the Run menu), or when time stamping has been turned off for all upstream flows in a conserved flow chain, CYCLETIME will return a zero. Any initial values of material in a conserved stock/flow chain will not be time-stamped, and hence will not be used for cycle-time purposes.

Example:

In Figure 7-15, the converter per batch cycle time calculates the cycle-time (i.e., the average time per batch, spent in Process 1 and Process 2) for material which is currently exiting the process through the flow exit

process 2. per unit volume cycle time calculates the cycle-time on a unit volume basis.

*Figure 7-15  
Illustrating the Use of the CYCLETIME Builtin*



## THROUGHPUT(<flow>[,<initial>])

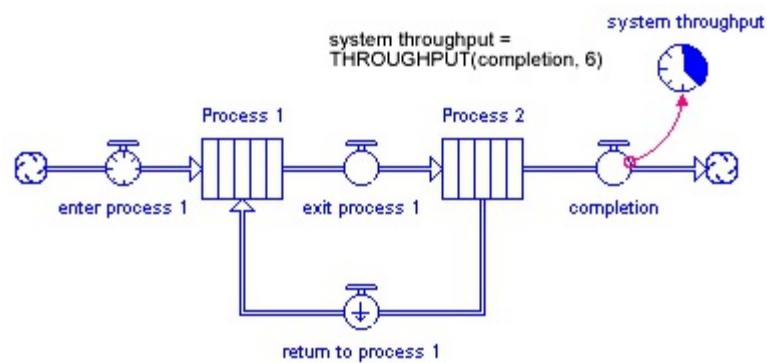
The THROUGHPUT function returns the cumulative average throughput (measured in units/time) of the time-stamped flow volume, over the course of the simulation run. THROUGHPUT provides a measure of the average rate of time-stamped flow through the system. THROUGHPUT performs its calculation by summing the time-stamped flow volumes over the course of the simulation, and then dividing this quantity by the current simulation time minus the time that the first time-stamped flow moved through the flow.

When you are operating in the Normal simulation mode (in the Time Specs dialog, under the Run menu), or when time stamping has been turned off for all upstream flows in a conserved flow chain, THROUGHPUT will return a zero. Any initial values of material in a conserved stock/flow chain will not be time-stamped, and hence will not be used for cycle-time purposes. Finally, if you have not specified an optional *initial* value, THROUGHPUT will return a zero until the time-stamped portion of the flow volume is positive.

Example:

In Figure 7-16, the converter system throughput calculates the average throughput (since the start of the run) of the time-stamped stuff moving through the completion flow. In this case, the throughput calculation provides a metric for the average rate of movement of material (in units/time) through a discrete process. The initial value returned by the converter is 6.

Figure 7-16  
Illustrating the Use of the THROUGHPUT Builtin



 [Related Topics](#)

# Array Functions

This section describes the following Builtins:

- [ARRAYIDX](#)
- [ARRAYMAX](#)
- [ARRAYMAXIDX](#)
- [ARRAYMEAN](#)
- [ARRAYMIN](#)
- [ARRAYMINIDX](#)
- [ARRAYRANK](#)
- [ARRAYSTDDEV](#)
- [ARRAYSUM](#)
- [ARRAYVALUE](#)
- [DIMSIZE](#)

The functions in this section are designed to supplement the Array capabilities of the software. Using these functions you can find the sum, the standard deviation, or the mean across dimensions of arrayed variables. The details for using these Builtin functions follow.

## **ARRAYIDX([<dimension index>])**

The ARRAYIDX Builtin gives the numerical index of the current array element in the array holding the equation. The <dimension index> is 1 for the first dimension (default when not specified) or 2 for the second dimension.

## **ARRAYMAX(<arrayed variable>)**

The ARRAYMAX Builtin finds the maximum value in the array. The calculation can be done in either an arrayed or a non-arrayed variable.

When using the ARRAYMAX Builtin to report the maximum value of a one-dimensional arrayed variable, you must choose [\*] from the secondary dialog, which indicates that you want the value for all the elements within the variable.

When using the ARRAYMAX Builtin to report the maximum value for a two-dimensional arrayed variable, it is possible to calculate it in three ways:

To report the maximum value across both dimensions of the arrayed input variable, select [\*,\*] in the secondary dialog. You can report the arrayed variable's maximum value in either an arrayed or a non-arrayed converter or flow.

Example:

Maximum Value for Salary = Salaries[\*,\*]

With this Builtin, you can also report a value across one dimension of the arrayed input variable, or roll up a value across (or down) one dimension of a two-dimensional arrayed variable into a one-dimensional arrayed variable. For examples, see [Other Array function examples](#).

## **ARRAYMAXIDX(<arrayed variable>[, <dimension index>])**

The ARRAYMAXIDX Builtin gives the index of the maximum-valued element in the array, where 1 refers to the first element, 2 to the second element, and so on. The optional <dimension index> specifies which dimension is desired. The <dimension index> is 1 for the first dimension and 2 for the second dimension (two-dimensional arrays only). The <dimension index> is 1 if it is not specified.

Examples:

For a two-dimensional array (called "array"), get the row number of the largest-valued element in the array:

```
ARRAYMAXIDX(array[*,*], 1)
```

For a two-dimensional array (called "array"), get the column number of the largest element in the array:

```
ARRAYMAXIDX(array[*,*], 2)
```

With this Builtin, you can also report a value across one dimension of the arrayed input variable, or roll up a value across (or down) one dimension of a two-dimensional arrayed variable into a one-dimensional arrayed variable. For examples, see [Other Array function examples](#).

## **ARRAYMEAN(<arrayed variable>)**

The ARRAYMEAN Built-in calculates the arithmetic mean across the elements in a one or two-dimensional Array. The calculation is defined as the sum of the elements, divided by the number of elements in the Array.

When using the ARRAYMEAN Built-in to report the mean of a one-dimensional arrayed variable, you must choose [\*] from the secondary dialog, which indicates that you want the mean for all the elements within the variable. The ARRAYMEAN calculation can be done in either an arrayed or a non-arrayed converter or flow.

When using the ARRAYMEAN Built-in to report the mean of a two-dimensional arrayed variable, it is possible to calculate the mean three ways:

To report the mean across both dimensions of the arrayed input variable, select [\*,\*] in the secondary dialog. You can report the arrayed variable's mean in either an arrayed or a non-arrayed converter or flow.

Example:

Average Salary for All Employees = Salaries[\*,\*]

With this Built-in, you can also report a value across one dimension of the arrayed input variable, or roll up a value across (or down) one dimension of a two-dimensional arrayed variable into a one-dimensional arrayed variable. For examples, see [Other Array function examples](#).

## **ARRAYMIN(<arrayed variable>)**

The ARRAYMIN Builtin finds the minimum value in the array. The calculation can be done in either an arrayed or a non-arrayed variable.

When using the ARRAYMIN Builtin to report the minimum value of a one-dimensional arrayed variable, you must choose [\*] from the secondary dialog, which indicates that you want the value for all the elements within the variable.

When using the ARRAYMIN Builtin to report the minimum value for a two-dimensional arrayed variable, it is possible to calculate it in three ways:

To report the minimum value across both dimensions of the arrayed input variable, select [\*,\*] in the secondary dialog. You can report the arrayed variable's minimum value in either an arrayed or a non-arrayed converter or flow.

Example:

Minimum Value for Salary = Salaries[\*,\*]

With this Builtin, you can also report a value across one dimension of the arrayed input variable, or roll up a value across (or down) one dimension of a two-dimensional arrayed variable into a one-dimensional arrayed variable. For examples, see [Other Array function examples](#).

## **ARRAYMINIDX(<arrayed variable>[, <dimension index>])**

The ARRAYMINIDX Builtin gives the index of the minimum-valued element in the array, where 1 refers to the first element, 2 to the second element, and so on. The optional <dimension index> specifies which dimension is desired. The <dimension index> is 1 for the first dimension and 2 for the second dimension (two-dimensional arrays only). The <dimension index> is 1 if it is not specified.

Examples:

For a one-dimensional array (called "list"), find the location of the smallest-valued element in the array:

```
ARRAYMINIDX(list[*])
```

For a two-dimensional array (called "array"), find the row number of the smallest-valued element in the array:

```
ARRAYMINIDX(array[*,*], 1)
```

For a two-dimensional array (called "array"), find the column number of the smallest element in the array:

```
ARRAYMINIDX(array[*,*], 2)
```

With this Builtin, you can also report a value across one dimension of the arrayed input variable, or roll up a value across (or down) one dimension of a two-dimensional arrayed variable into a one-dimensional arrayed variable. For examples, see [Other Array function examples](#).

## **ARRAYRANK(<arrayed variable>, <rank number>[,<secondary sort variable>])**

The ARRAYRANK Built-in gives the numerical index of the <arrayed variable> with the given <rank number> when the array is sorted in ascending order. The optional <secondary sort variable> specifies the secondary sort field for variables with the same value. The <secondary sort variable> must be the same size as the <arrayed variable> or the secondary sort will not occur (this is only enforced when you simulate the model, so you will not be warned).

Example:

If array A contains the values { 6, 2, 8, 5 }

ARRAYRANK(A, 2) will equal 4 (i.e., the fourth element of array A is the second largest value)

Using ARRAYRANK causes the specified array to be sorted once each time step (DT). By definition, ARRAYRANK only works on one-dimensional arrays. For two-dimensional arrays, you must pass a row or column of the array in order to sort it.

Typical uses of ARRAYRANK as the Apply to All equation in an array are shown below.

In the following examples, array A is a one-dimensional array of size X, and array C is a two-dimensional array of size [X, Y].

ARRAYRANK(A[\*], ARRAYIDX()): Fills the given array, which must be of size X, with the indices of the sorted elements of A in ascending order. If this array is called B, using ARRAYVALUE(A[\*], B[X]) in another array of size X will fill that array with the sorted values of A in ascending order.

ARRAYRANK(A[\*], DIMSIZE() - ARRAYIDX() + 1): Fills the given array, which must be of size X, with the indices of the sorted elements of A in descending order. If this array is called B, using ARRAYVALUE(A[\*], B[X]) in another array of size X will fill that array with the sorted values of A in descending order.

Note: You can use the formulation, DIMSIZE() - rank + 1, in any of the following examples to get the results in descending order, rather than ascending order.

ARRAYVALUE(A[\*], ARRAYRANK(A[\*], ARRAYIDX())): Fills the given array, which must be of size X, with the sorted values of array A in ascending order.

ARRAYVALUE(C[\*,\*], ARRAYRANK(C[\*, 2], ARRAYIDX()), 2): Fills the

given array, which must be of size X, with the sorted values of column 2 of array C in ascending order. Note the column number 2 must be specified both in ARRAYRANK() and ARRAYVALUE(). This can be avoided if an intermediate one-dimensional array variable is used to hold C[\*, 2].

ARRAYVALUE(C[\*,\*], 3, ARRAYRANK(C[3, \*], ARRAYIDX())): Fills the given array, which must be of size X, with the sorted values of row 3 of array C in ascending order. Note the row number 3 must be specified both in ARRAYRANK() and ARRAYVALUE(). This can be avoided if an intermediate one-dimensional array variable is used to hold C[3, \*].

ARRAYVALUE(C[\*,\*], ARRAYIDX(), ARRAYRANK(C[X, \*], ARRAYIDX(2))): Fills the given array, which must be of size [X, Y], with sorted values of each row of C.

ARRAYVALUE(C[\*,\*], ARRAYRANK(C[\*, Y], ARRAYIDX()), ARRAYIDX(2)): Fills the given array, which must be of size [X, Y], with sorted values of each column of C.

ARRAYRANK(A[\*], ARRAYIDX(), B[\*]): Fills the given array, which must be of size X, with the indices of the sorted elements of A in ascending order. Any ties between values in A will be broken using corresponding values from B.

**IMPORTANT NOTE:** After using ARRAYRANK, subscript names may no longer make sense. For example, if Sales[Location] is an array with sales figures for { NY, Chicago, LA }, when Sales is sorted into another array A of size Location, the sales figures in A, although they are indexed by location name, will no longer correspond to corresponding locations. For this reason, it is generally preferred to only use this function on arrays with numbered subscript names.

## **ARRAYSTDDEV(<arrayed variable>)**

The ARRAYSTDDEV Builtin calculates standard deviation for all the elements in a given arrayed variable. The calculation can be done either in an arrayed or a non-arrayed variable.

When using the ARRAYSTDDEV Builtin to report the standard deviation of a one-dimensional arrayed variable, you must choose [\*] from the secondary dialog, which indicates that you want the standard deviation of all the elements within the variable.

When using the ARRAYSTDDEV Builtin to report the standard deviation of a two-dimensional arrayed variable, it is possible to calculate it in three ways:

To report the standard deviation across both dimensions of the arrayed input variable, select [\*,\*] in the secondary dialog. You can report the arrayed variable's standard deviation in either an arrayed or a non-arrayed converter or flow.

Example:

Standard Deviation in Salary = Salaries[\*,\*]

With this Builtin, you can also report a value across one dimension of the arrayed input variable, or roll up a value across (or down) one dimension of a two-dimensional arrayed variable into a one-dimensional arrayed variable. For examples, see [Other Array function examples](#).

## **ARRAYSUM(<arrayed variable>)**

The ARYSUM Builtin calculates the sum of all the elements in a given one or two dimensional Array. The calculation can be done in either an arrayed or a non-arrayed variable.

When using the ARYSUM Builtin to report the sum across a one-dimensional arrayed variable, you must choose [\*] from the secondary dialog, which indicates that you want the sum across all the elements within the variable.

When using the ARYSUM Builtin to report the sum across a two-dimensional arrayed variable, it is possible to calculate it in three ways:

To report the sum across both dimensions of the arrayed input variable, select [\*,\*] in the secondary dialog. You can report the arrayed variable's sum in either an arrayed or a non-arrayed converter or flow.

Example:

Total Salary for All Employees = Salaries[\*,\*]

With this Builtin, you can also report a value across one dimension of the arrayed input variable, or roll up a value across (or down) one dimension of a two-dimensional arrayed variable into a one-dimensional arrayed variable. For examples, see [Other Array function examples](#).

## **ARRAYVALUE(<arrayed variable>, <row number>[, <column number>])**

The ARRAYVALUE Builtin gives the value that is in the specified element of the given array. Both <row number> and <column number> start at 1. If an array is one-dimensional, only the row number can be specified. If an array is two-dimensional, both the row number and the column number must be specified. This Builtin can be used directly with the ARRAYMINIDX and ARRAYMAXIDX Builtins.

Example:

For parallel one-dimensional arrays (called "list" and "attribute"), find the attribute associated with the largest element in the array "list":

```
ARRAYVALUE(attribute[*], ARRAYMAXIDX(list[*]))
```

## **DIMSIZE([<dimension index>])**

The DIMSIZE Built-in gives the number of elements in the array holding the equation. The <dimension index> is 1 for the first dimension (default when not specified) or 2 for the second dimension.

# Other Array function examples

The following examples apply to all of the above Array Built-in functions except **ARRAYVALUE**:

- To report a value across one dimension of the arrayed input variable, select the element name from the appropriate Dimension in the secondary dialog, and then select the [\*] for the other Dimension.

For example, for the [ARRAYSUM](#) function, to find the Total Employees in Boston where your input variable is **Employees**, your Dimensions are **Location** and **Function**, use

`ARRAYSUM(Employees[Boston,*])`

This setup will return a scalar value, consisting of the total Employees in Boston. You can make this calculation in either an arrayed or a non-arrayed converter or flow.

- You can roll up a value across (or down) one dimension of a two-dimensional arrayed variable into a one-dimensional arrayed variable. In defining the one-dimensional variable, select the Dimension that you want to have reflect the roll-up. Define the function by selecting the Dimension name to define the dimension the variables have in common. Select [\*] for the other dimension.

For example, for the [ARRAYSTDDEV](#) function, to find the Standard Deviation in Salaries of all Employees across **Location**, where your input variable is **Salaries** and your Dimensions are **Location** and **Function**, use

`ARRAYSTDDEV(Salaries[Location,*])`

This setup will return the standard deviation in salary for all Employees in Boston in the **Salaries[Boston]** element of the variable, the standard deviation for Chicago in the **Salaries[Chicago]** element, etc.

The following example demonstrates how the [ARRAYIDX](#) and [DIMSIZE](#) functions can be used with the [ARRAYVALUE](#) function to shift data across an array.

Example:

The following equation on the stock's inflow moves material from one element in that stock to the next:

`in[x] = ARRAYVALUE(out[*], ARRAYIDX(out[x])%DIMSIZE(out[*]) + 1)`

where "out" is the outflow of the stock, and "x" is the dimension name for the single dimension used by "in" and "out". The above formulation uses the "%" operator, which is equivalent to the MOD built-in function. It could also be written using MOD:

```
in[x] = ARRAYVALUE(out[*], MOD(ARRAYIDX(out[x]),  
DIMSIZE(out[*])) + 1)
```

Assuming "x" has elements named 1, 2, 3, and 4, this equation is equivalent to the separate equations:

```
in[1] = out[2]  
in[2] = out[3]  
in[3] = out[4]  
in[4] = out[1]
```

A two-dimensional example, which moves elements to both the next row and column appears below:

```
in_2D[x,y] = ARRAYVALUE(out_2D[*,*],  
ARRAYIDX(out_2D[x,1])%DIMSIZE(out_2D[*,*]) + 1,  
ARRAYIDX(out_2D[1,y], 2)%DIMSIZE(out_2D[*,*], 2) + 1)
```

If both "x" and "y" have elements 1, 2, 3, 4, this is equivalent to the separate equations:

```
in_2D[1, 1] = out_2D[2, 2]  
in_2D[1, 2] = out_2D[2, 3]  
in_2D[1, 3] = out_2D[2, 4]  
in_2D[1, 4] = out_2D[2, 1]  
in_2D[2, 1] = out_2D[3, 2]  
in_2D[2, 2] = out_2D[3, 3]  
in_2D[2, 3] = out_2D[3, 4]  
in_2D[2, 4] = out_2D[3, 1]  
in_2D[3, 1] = out_2D[4, 2]  
in_2D[3, 2] = out_2D[4, 3]  
in_2D[3, 3] = out_2D[4, 4]  
in_2D[3, 4] = out_2D[4, 1]  
in_2D[4, 1] = out_2D[1, 2]  
in_2D[4, 2] = out_2D[1, 3]  
in_2D[4, 3] = out_2D[1, 4]  
in_2D[4, 4] = out_2D[1, 1]
```

 [Related Topics](#)

# Special Functions

This section describes the following Builtins:

- [CGROWTH](#)
- [COUNTER](#)
- [DELAY1](#)
- [DELAY3](#)
- [DELAYN](#)
- [DT](#)
- [ENDVAL](#)
- [FORCST](#)
- [HISTORY](#)
- [INIT](#)
- [LOOKUP](#)
- [LOOKUPXY](#)
- [PAUSE](#)
- [REWORK](#)
- [RUNCOUNT](#)
- [SMTH1](#)
- [SMTH3](#)
- [SMTHN](#)
- [SOUND](#)
- [STARTTIME](#)
- [STOPTIME](#)
- [SWITCH](#)
- [TIME](#)
- [TREND](#)

As the name of the category suggests, the special functions were designed to be used in a variety of different circumstances. You will find them helpful in the smoothing of noisy data streams (the SMTH functions), in setting up specific scenarios for a model simulation (COUNTER, SWITCH, ENDVAL), in doing simple trend analysis and extrapolation (FORCST, TREND), in

establishing relationships which require knowledge of the simulation specs or of specific model variables (CGROWTH, DT, INIT, REWORK, STARTTIME, STOPTIME, TIME, RUNCOUNT), in reusing or defining data sets for graphical functions (LOOKUP, LOOKUPXY), and in providing visual and auditory feedback to the user (PAUSE, SOUND). Together, the special functions offer a wealth of capabilities which can fill out the details of your model.

## CGROWTH(<percentage>)

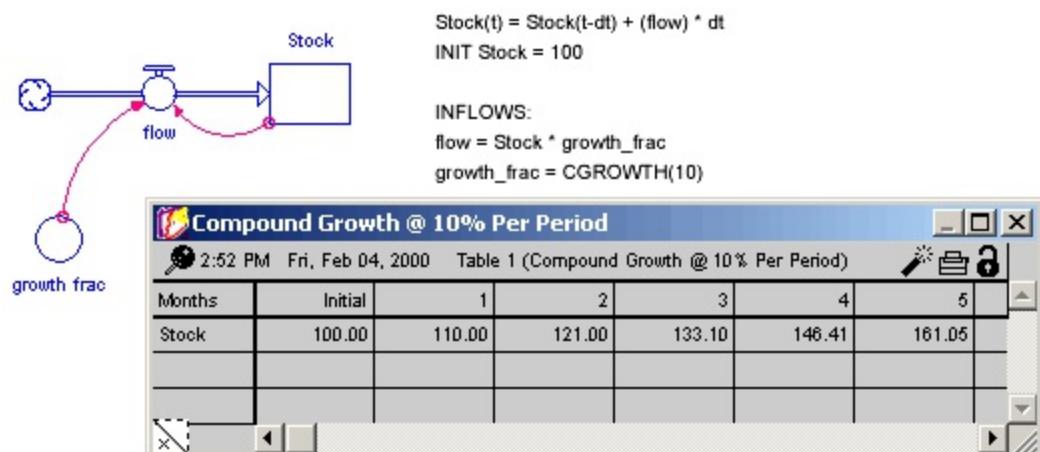
In many instances, you will want a stock to grow in compounding fashion, at a certain percentage rate per unit of time. You'd like to input the percentage growth rate, and to have the results of the process be independent of the DT which is being used in the model simulation.

The CGROWTH function enables you to define such DT-independent growth rates. Simply provide CGROWTH with a per-period percentage growth rate. When embedded in a compounding process, CGROWTH will ensure that the stock grows at the per-period rate you have specified, independent of the DT which is being used.

Example:

Growth Fraction = CGROWTH(10) produces 10% per unit time compound growth for the Stock in Figure 7-17. The specific numerical results of this compound growth process will be independent of the DT being used for the simulation.

Figure 7-17 Using the CGROWTH Function



## COUNTER(<start>, <end>)

Activities in a simulation model often are driven by external cycles. A financial department, for example, runs on a 30-day billing cycle. A business concern may have seasonal demand cycles. When the simulation is to run over several cycles, it is useful to know where in the cycle you are, at any point in time.

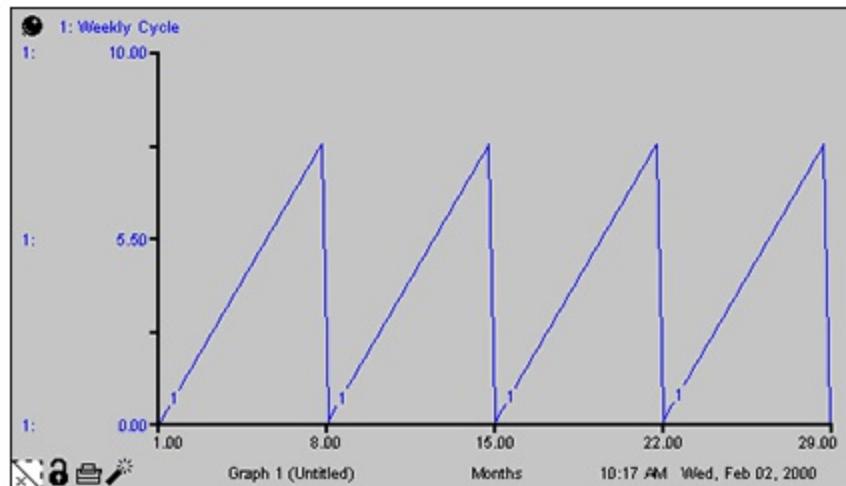
The COUNTER function enables you to define such time-dependent cycles. You provide COUNTER with starting and ending values for the cycle. COUNTER will map *start* to the From time you have specified in the Time Specs dialog. It will subsequently return linearly increasing values, as time progresses. Once COUNTER has counted up to the *end*, it will reset itself to *start* and begin the cycle anew.

Example:

Weekly Cycle = Counter(1,8) produces a linearly increasing cycle which begins at 1, runs up to 8, and then repeats itself. The cycle thus translates simulation time into days of the week. Its behavior is shown in Figure 7-18. In the example, From time has been set to 1.

Figure 7-18

Generating a Weekly Cycle Using the COUNTER Builtin

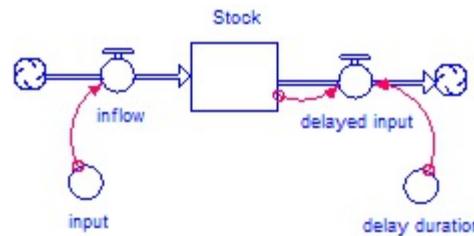


## **DELAY1(<input>, <delay duration>[, <initial>])**

The DELAY1 function calculates a first-order material delay of input, using an exponential delay time of delay duration, and an optional initial value initial for the delay. If you do not specify an initial value initial, DELAY1 assumes the value to be the initial value of input.

The DELAY1 function is equivalent to the structure and equations shown in Figure 7-18a. This structure is a draining process.

*Figure 7-18a  
Structure of a First-Order Material Delay Process*



$$\text{Stock} = \text{Stock} + DT * (\text{inflow} - \text{delayed\_input})$$

$$\text{INIT Stock} = \text{input} * \text{duration\_delay} \text{ or specified initial value} * \text{duration\_delay}$$

$$\text{inflow} = \text{input}$$

$$\text{delayed\_input} = \text{Stock} / \text{delay\_duration}$$

$$\text{input} = \text{some variable or constant}$$

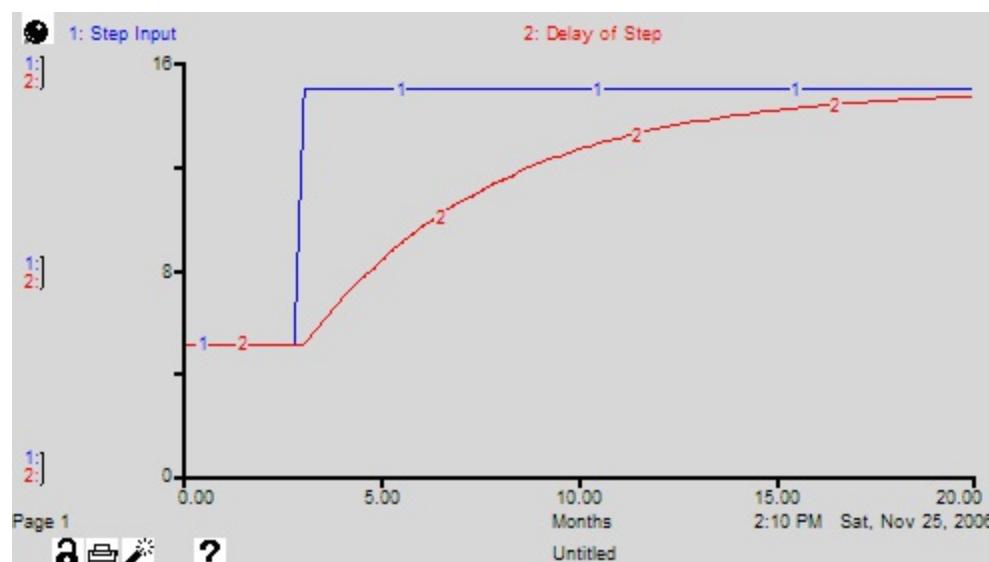
$$\text{delay\_duration} = \text{some variable or constant}$$

Example:

$\text{Delay\_of\_Step} = \text{DELAY1}(\text{Step\_Input}, 5)$  where  $\text{Step\_Input} = 5 + \text{STEP}(10,3)$  produces the pattern shown in Figure 7-18b.

*Figure 7-18b*

*Response of a First-Order Material Delay to STEP Input*



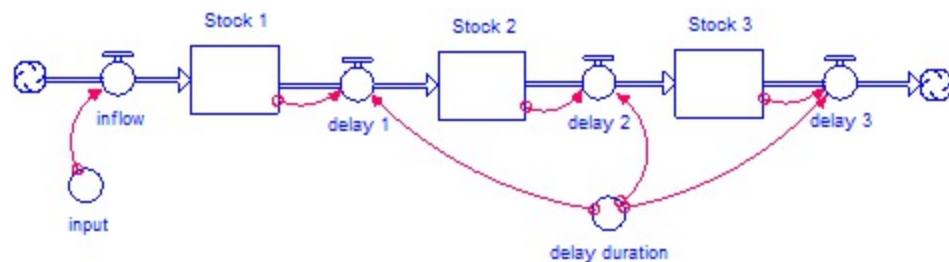
**Note:** The dynamic behavior of DELAY1 is identical to SMTH1, except when the delay time changes. In this case, DELAY1 will conserve material and SMTH1 will not. DELAY1 is conceptually a flow and SMTH1 is conceptually a stock. Because DELAY1 is a flow concept, you must C-to-F any converter that uses this function if you wish to see the correct reported results in a table (or when exported).

---

## **DELAY3(<input>, <delay duration>[, <initial>])**

The DELAY3 function calculates a third-order material delay of input, using an exponential delay time of delay duration, and an optional initial value initial for the delay. DELAY3 does this by setting up a cascade of three first-order material delays, each with a delay duration of delay duration/3. DELAY3 returns the value of the final delay in the cascade. If you do not specify an initial value initial, DELAY3 assumes the value to be the initial value of input. The DELAY3 function will return the value of delay 3 in the structure and equations shown in Figure 7-18c.

*Figure 7-18c  
Structure of a Third-Order Material Delay Process*

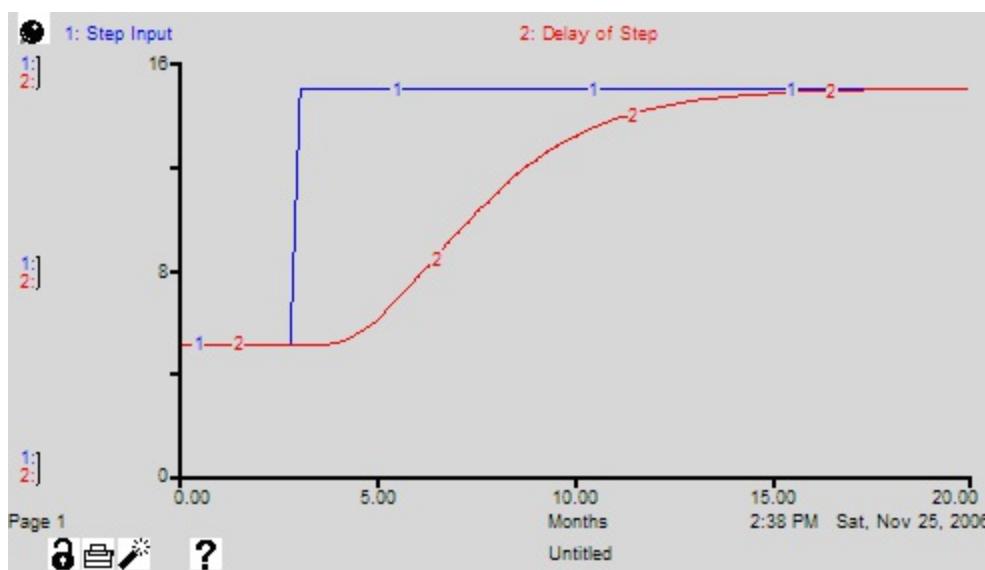


```
INIT Stock_1 = INIT Stock_2 = INIT Stock_3 = input or specified initial value times duration_delay/3
inflow = input
delay_1 = Stock_1/(delay_duration/3)
delay_2 = Stock_2/(delay_duration/3)
delay_3 = Stock_3/(delay_duration/3)
input = some variable or constant
delay_duration = some variable or constant
```

Example:

Delay\_of\_Step = DELAY3(Step\_Input, 5) where Step\_Input = 5 + STEP(10,3) produces the pattern shown in Figure 7-18d.

*Figure 7-18d  
Response of a Third-Order Material Delay to STEP Input*



Note: The dynamic behavior of DELAY3 is identical to SMTH3, except when the delay time changes. In this case, DELAY3 will conserve material and SMTH3 will not. DELAY3 is conceptually a flow and SMTH3 is conceptually a stock. Because DELAY3 is a flow concept, you must C-to-F any converter that uses this function if you wish to see the correct reported results in a table (or when exported).

## **DELAYN(<input>, <delay duration>, <n>[, <initial>])**

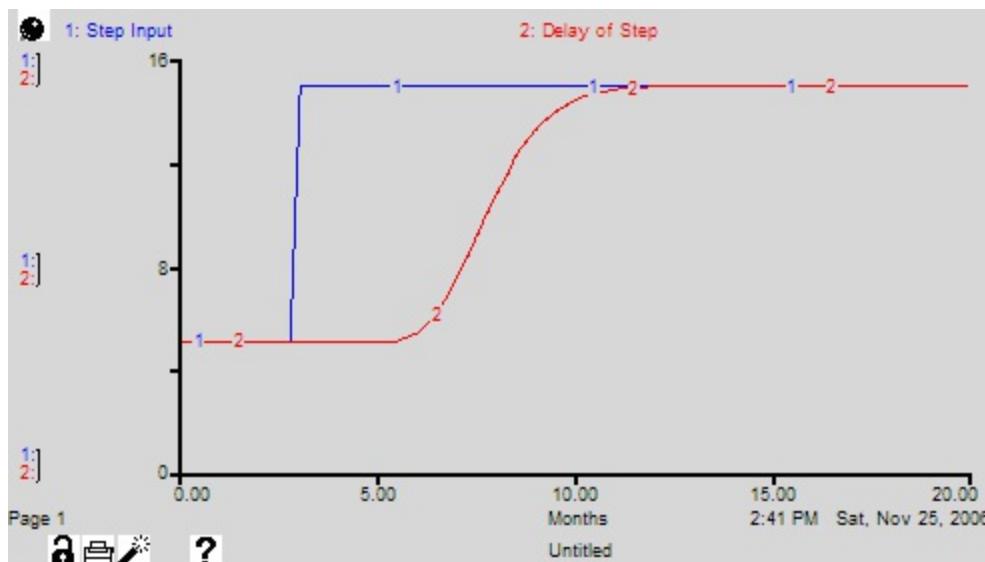
The DELAYN function calculates an nth-order material delay of input, using an exponential delay time of delay duration, and order number of n, and an optional initial value initial for the delay. DELAYN does this by setting up a cascade of n first-order material delays, each with a delay duration of delay duration/n. DELAYN returns the value of the final delay in the cascade. If you do not specify an initial value initial, DELAYN assumes the value to be the initial value of input. n should be specified as an integer.

Example:

Delay\_of\_Step = DELAYN(Step\_Input, 5, 9) where Step\_Input = 5 + STEP(10,3) produces the pattern shown in Figure 7-18e.

*Figure 7-18e*

*Response of a Ninth-Order Material Delay to STEP Input*



**Note:** The dynamic behavior of DELAYN is identical to SMTHN, except when the delay time changes. In this case, DELAYN will conserve material and SMTHN will not. DELAYN is conceptually a flow and SMTHN is conceptually a stock. Because DELAYN is a flow concept, you must C-to-F any converter that uses this function if you wish to see the correct reported results in a table (or when exported).

## **DT**

DT is the time increment for calculations in a model simulation. DT is found in Run Specs... under the Run menu. For more information about DT, see [Introduction to DT](#).

## **ENDVAL(<input>,[<initial>])**

The ENDVAL function returns the ending value of *input*, from the most recent simulation run in a session with a model. The first time you run the model after opening it, ENDVAL will return the *initial* value you have specified. If you do not specify an initial value initial, ENDVAL assumes the first-run value to be the initial value of input.

Example:

```
Is_Performance_Improving = Current_Performance_Indicator -  
ENDVAL(Current_Performance_Indicator,0)
```

enables your model to look at current performance relative to the ending value of performance in the last simulation run of a given simulation session. In so doing, ENDVAL provides a mechanism for looking at run-to-run performance changes, and thus can provide a vehicle for triggering "as-needed" coaching to the model consumer.

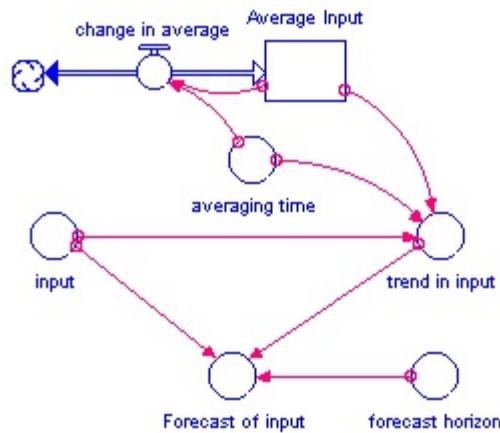
## **FORCST(<input>,<time>,<horizon>[,<initial>])**

The FORCST function performs simple trend extrapolation. Here's how it works. First, FORCST calculates the trend in input, based upon the value of *input*, the first order exponential average of *input*, and the averaging time. (Think of the averaging time as the time over which you wish to calculate a trend.) Then FORCST extrapolates the trend into the future - you specify the distance into the future by providing a value for *horizon*. If you do not specify *initial*, FORCST substitutes 0 for the initial value of the trend in *input*.

The FORCST function is equivalent to the structural diagram and equations shown in Figure 7-19.

*Figure 7-19*

*Structure for Forecast Based on Trend Extrapolation*



```
Forecast_of_Input = input*(1+trend_in_input*forecast_horizon)
trend_in_input = (input-Average_Input)/(Average_Input*averaging_time)
input = some variable or constant
Average_Input = Average_Input+(dt)*(change_in_average)
INIT (Average_Input) = input or specified initial value
change_in_average = (input-Average_Input)/averaging_time
averaging_time = some variable or constant
Forecast horizon = some variable or constant
```

**Example:**

Sales\_Forecast = FORCST(Sales,10,15,0) produces a forecast of sales 15 time units into the future. The forecast is based on current sales, and the trend in sales over the last 10 time units. The initial growth trend in sales is set to 0.

---

**Tip:** If you encounter noise in the variable to be forecasted, you may wish to filter the randomness by basing your forecast on an exponential smooth of the variable. To do this, use the [SMTH1](#) or the [SMTH3](#) function (described later in this section).

## **HISTORY(<variable>,<time>)**

The HISTORY function returns the value of a variable at a prior time in the simulation.

---

*Note:* HISTORY (stock, TIME-1) is the same as DELAY (stock,1).

## **INIT(<stock>) or INIT(<flow>) or INIT(<converter>)**

The INIT function takes the initial value of the stock, flow, or converter, where the initial value for the entity has been calculated at the outset of a simulation. The INIT function will accept only a single model variable name within its parentheses.

Examples:

Distance\_Traveled = Position - INIT(Position)

Computes Distance Traveled as the difference between current position and the initial value of position.

Debt\_Ratio = Debt/INIT(Debt)

Computes Debt Ratio as the ratio of Debt to its initial value. When creating graphical functions, it often is useful to normalize the input to the graphical function in this manner.

## **LOOKUP(<graphical variable>,<expression>)**

The LOOKUP function evaluates the <graphical variable> at the given <expression> (versus using the equation stored in the graphical function itself).

## **LOOKUPXY(<x graphical variable>, <y graphical variable>, <expression>)**

The LOOKUPXY function evaluates a graphical function described by (x,y) pairs at the <expression>. <x graphical variable> is a graphical function that contains the desired x-coordinates in the graphical function's y-values. <y graphical variable> is a graphical function that contains the corresponding y-coordinates in the graphical function's y-values. In both graphical functions, the x-value is ignored; the only thing that is used from the graphical function is the number of data points and the y-values. These two fields constitute the (x, y) pairs of the graphical function (and will necessarily be sorted in x-ascending order at the start of each run).

The <expression> is evaluated and found along the x-axis. The corresponding y-coordinate is returned if the value matches an x-coordinate. Linear interpolation is performed between x-coordinates. The last y-value is used at either end if the x-value is out-of-range of the given set of points.

The <y graphical function> can be controlled with a Graphical Input Device by itself, but if the <x graphical function> is controlled by a Graphical Input Device, the <y graphical function> must also be controlled.

## **PAUSE**

When it is called, the PAUSE function causes the computer to execute the Pause command from the Run menu. The PAUSE function thus allows you to pause your simulation run based on model conditions.

Example:

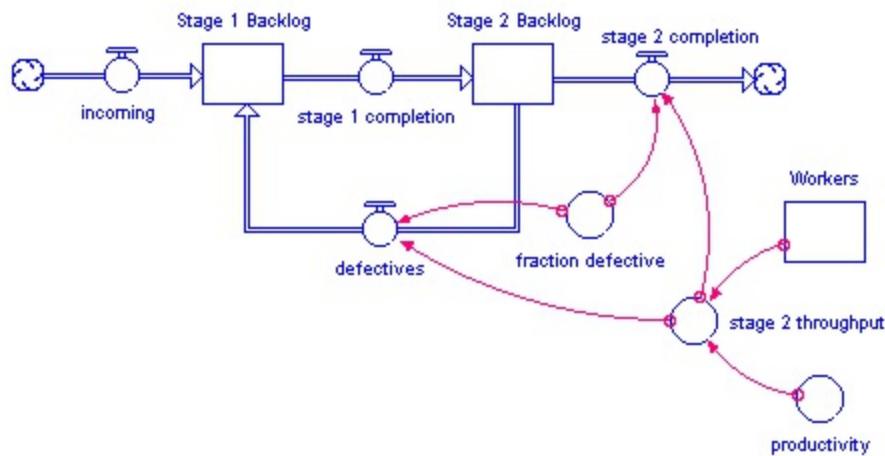
Sim\_Pause = IF US\_Deficit > IBM\_Revenues THEN PAUSE ELSE 0 causes the simulation to pause its execution whenever the federal deficit becomes larger than the revenues of this large computer manufacturer.

## REWORK(<percentage>)

In many instances, you will want to represent a rework process. In Figure 7-20, for example, a production process is used to draw down a work backlog. In the Figure, a portion of the work (defectives) is shunted back to an earlier stage in the process to be reworked. In Figure 7-21, a Conveyor represents an inspection activity. The leakage flow from the activity moves rejected material to an earlier stage in the process, where it will subsequently be re-worked.

In either situation, using a simple fraction to represent the rework percentage will overstate the cumulative flow of material through the rework process. Each time that material passes through the inspection activity or production process, a fraction of it will be sent back to be re-worked. Double-counting can ensue. For example, if 100 units are sent through the process initially, a 10% defective fraction would send 10 units back to be re-worked in the first round. In the second round, 1 unit (10% of the 10 units) would be sent back. And, so on. After the fact, more material than 10% has been re-worked! In most cases, this is not what you intended.

*Figure 7-20  
A Simple Rework Process*



To get around this double-counting phenomenon, the software provides the REWORK Builtin. Use it only to represent a rework flow which deposits material at a point somewhere upstream in the main chain. Simply specify the percentage of total work flow that you desire to flow back upstream, to be reworked. *percentage* should be a value between 0 and 100.

Important Notes: (1) When using a draining process to represent the rework flow, the REWORK Builtin is not an appropriate choice. (2) When the defectives flow goes to a cloud, double-counting is not an issue. Hence, REWORK is not required.

## Example:

Figure 7-21 shows the results of using REWORK(10) to define the leakage fraction for the leakage flow from the Inspection Activity. With a total of 100 units of material entering the system through the entering wip flow, a total of 10 units of work flow through the failing inspection flow, over the course of the simulation.

*Figure 7-21  
Using the REWORK Builtin*

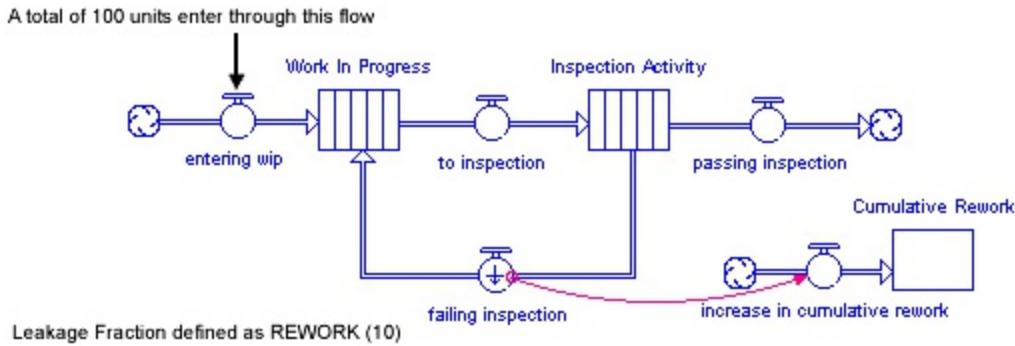


Table 1 (Untitled Table)					
Time	.00	5.00	10.00	15.00	Final
Cumulative Rework	0.00	7.95	9.94	10.00	10.00

## RUNCOUNT

The RUNCOUNT function accumulates the number of runs that a model has performed since it was created/opened. Once the model is closed, the function will reset and start at one the next time the model is opened. This is useful if you are building an interactive interface for your model and you want some change to take place after a few runs (e.g., turn on a piece of structure or post a message).

## **SMT1(<input>,<averaging time>[,<initial>])**

The SMT1 function calculates a first-order exponential smooth of *input*, using an exponential averaging time of *averaging time*, and an optional initial value *initial* for the smooth. If you do not specify an initial value *initial*, SMT1 assumes the value to be the initial value of input.

The SMT1 function is equivalent to the structure and equations shown in Figure 7-22.

This structure is a stock-adjustment process. Smooth of Input seeks the goal Input.

Example:

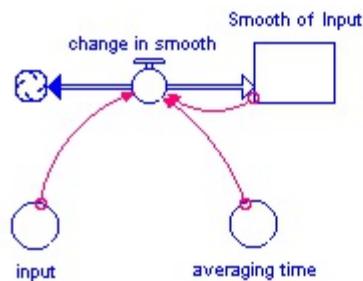
Smooth\_of\_Step = SMT1(Step\_Input,5)

where

Step\_Input = 5 + STEP(10,3) produces the pattern shown in Figure 7-23.

*Figure 7-22*

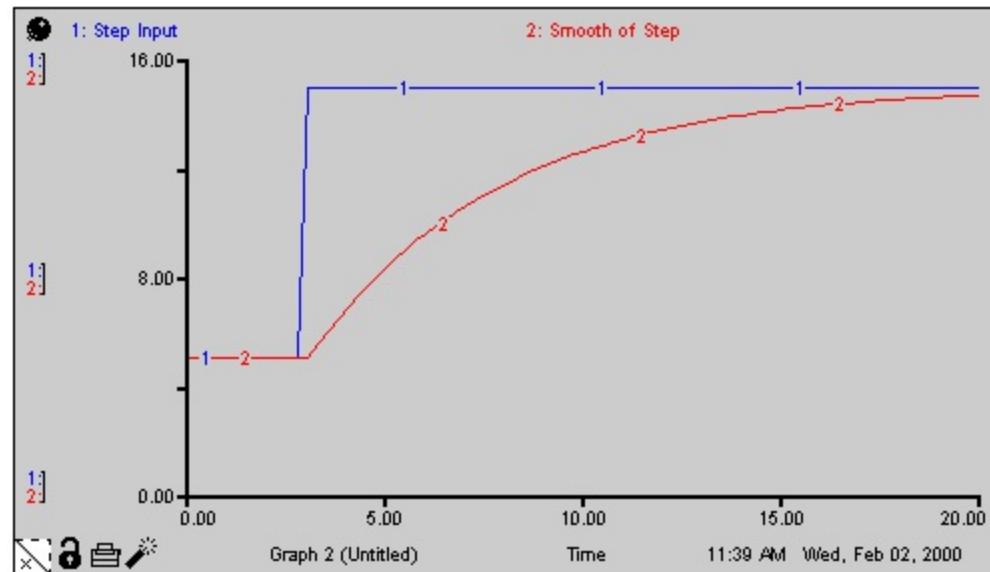
*Structure of First-Order Exponential Smoothing Process*



```
Smooth_of_Input= Smooth_of_Input + (DT)*(Change_In_Smooth)  
INIT Smooth_of_Input = Input or specified value  
Change_in_Smooth = (Input - Smooth_of_Input)/Averaging_Time  
Input = some variable or constant  
Averaging_Time = some variable or constant
```

*Figure 7-23*

*Response of First-Order Exponential Smooth to STEP Input*



---

Note: The dynamic behavior of SMTH1 is identical to DELAY1, except when the averaging time changes. SMTH1 is conceptually a stock and DELAY1 is conceptually a flow.

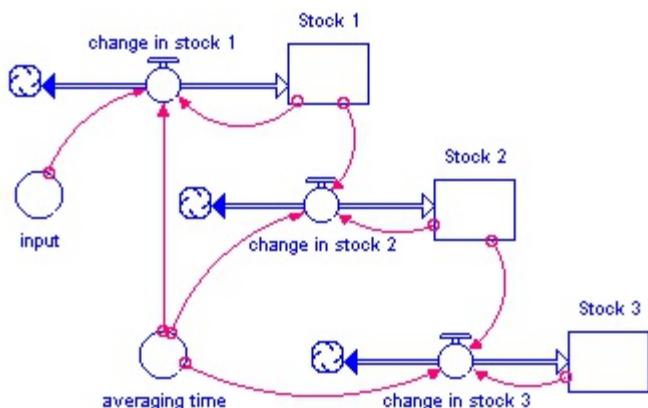
## SMT3(<input>,<averaging time>[,<initial>])

The SMT3 function performs a third-order exponential smooth of input, using an exponential averaging time of averaging time, and an optional initial value initial for the smooth. SMT3 does this by setting up a cascade of three first-order exponential smooths, each with an averaging time of averaging time/3. SMT3 returns the value of the final smooth in the cascade. If you do not specify an initial value initial, SMT3 assumes the value to be the initial value of input.

The SMT3 function will return the value of Stock 3 in the structure and equations shown in Figure 7-24.

Figure 7-24

Structure of Third-Order Exponential Smoothing Process



$$\text{Change\_in\_Stock\_3} = (\text{Stock\_2} - \text{Stock\_3}) / (\text{Averaging\_Time} / 3)$$

$$\text{Change\_in\_Stock\_2} = (\text{Stock\_1} - \text{Stock\_2}) / (\text{Averaging\_Time} / 3)$$

$$\text{Change\_in\_Stock\_1} = (\text{Input} - \text{Stock\_1}) / (\text{Averaging\_Time} / 3)$$

Average\_Time = some variable or constant

Input = some variable or constant

INIT(Stock\_3) = INIT(Stock\_2) = INIT(Stock\_1) = input or specified value

Examples:

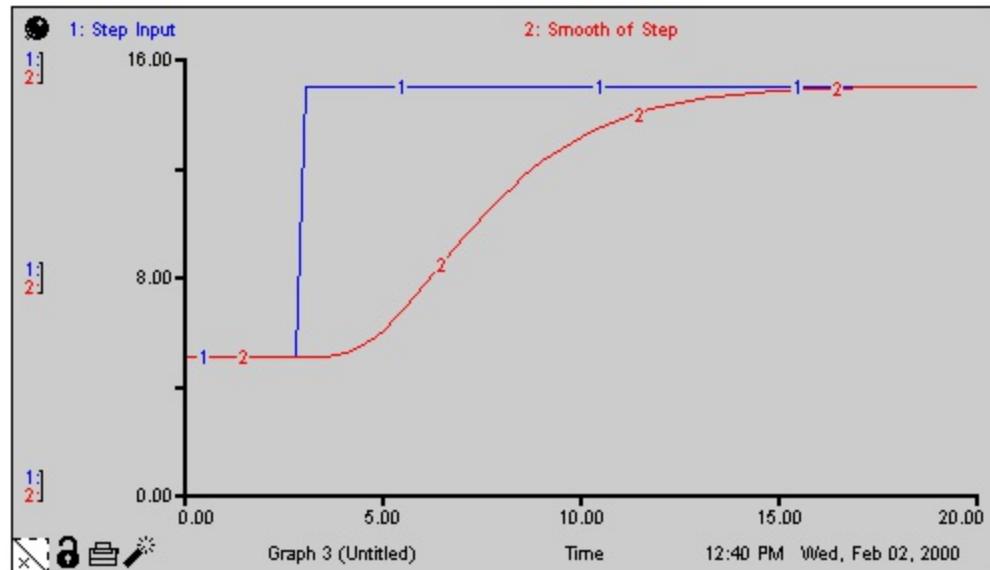
Smooth\_of\_Step = SMT3(Step\_Input,5)

where

Step\_Input = 5 + STEP(10,3) produces the pattern shown in Figure 7-25.

Figure 7-25

Response of Third-Order Exponential Smooth to STEP Input



Note: The dynamic behavior of SMTH3 is identical to DELAY3, except when the averaging time changes. SMTH3 is conceptually a stock and DELAY3 is conceptually a flow.

## **SMTHN(<input>,<averaging time>,<n>[,<initial>])**

The SMTHN function performs an  $n$ th-order exponential smooth of *input*, using an exponential averaging time of *averaging time*, an order number of *n*, and an optional initial value *initial* for the smooth. SMTHN does this by setting up a cascade of *n* first-order exponential smooths, each with an averaging time of *averaging time/n*. SMTHN returns the value of the final smooth in the cascade. If you do not specify an initial value *initial*, SMTHN assumes the value to be the initial value of *input*. *n* should be specified as an integer.

Examples:

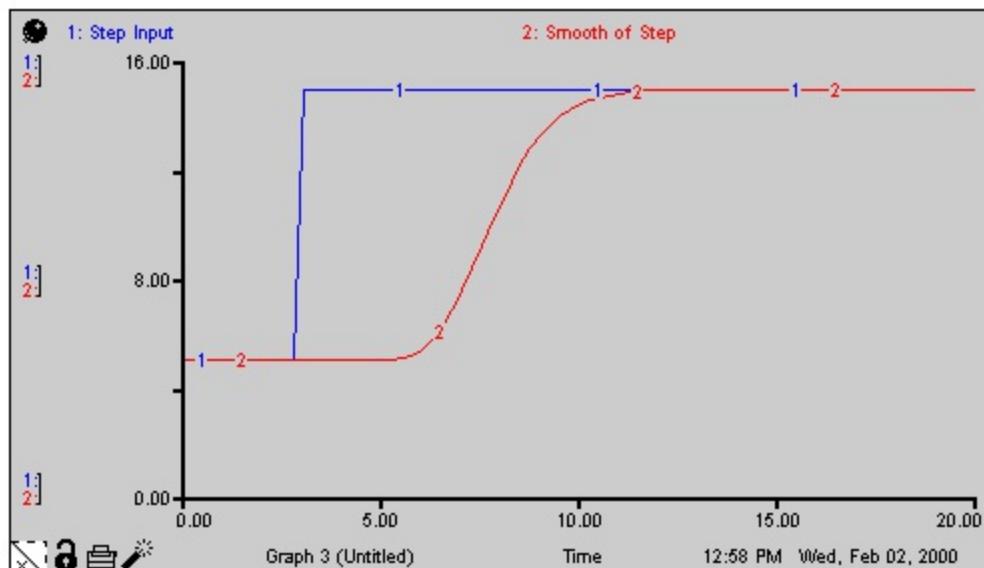
Smooth\_of\_Step = SMTHN(Step\_Input,5,9)

where

Step\_Input = 5 + STEP(10,3) produces the pattern shown in Figure 7-26.

*Figure 7-26*

*Response of Ninth-Order Exponential Smooth to STEP Input*



---

*Note:* The dynamic behavior of SMTHN is identical to DELAYN, except when the averaging time changes. SMTHN is conceptually a stock and DELAYN is conceptually a flow.

## **SOUND(<expression>)**

The SOUND function causes the computer to play the system "beep" sound, when *expression* is  $> 0$ . When SOUND is active, it takes on a numeric value of 1. Otherwise, SOUND assumes a numeric value of 0.

Example:

Warning\_Sound = SOUND(US\_Deficit - IBM\_Revenues) causes the computer to play the system "beep" sound, each DT of the model simulation, as long as the federal deficit is larger than the revenues of this large computer manufacturer.

## **STARTTIME**

STARTTIME returns the value that you have specified in the Run Specs dialog for the "From" time in your model.

## **STOPTIME**

STOPTIME returns the value that you have specified in the Run Specs dialog for the "To" time in your model.

## **SWITCH(<Input1>,<Input2>)**

The SWITCH function is equivalent to the following logic:

If *Input1* > *Input2* then 1 else 0.

## TIME

TIME is the current time within a model simulation. TIME often is used as an argument to logical functions, trigonometric functions, and graphical functions.

Examples:

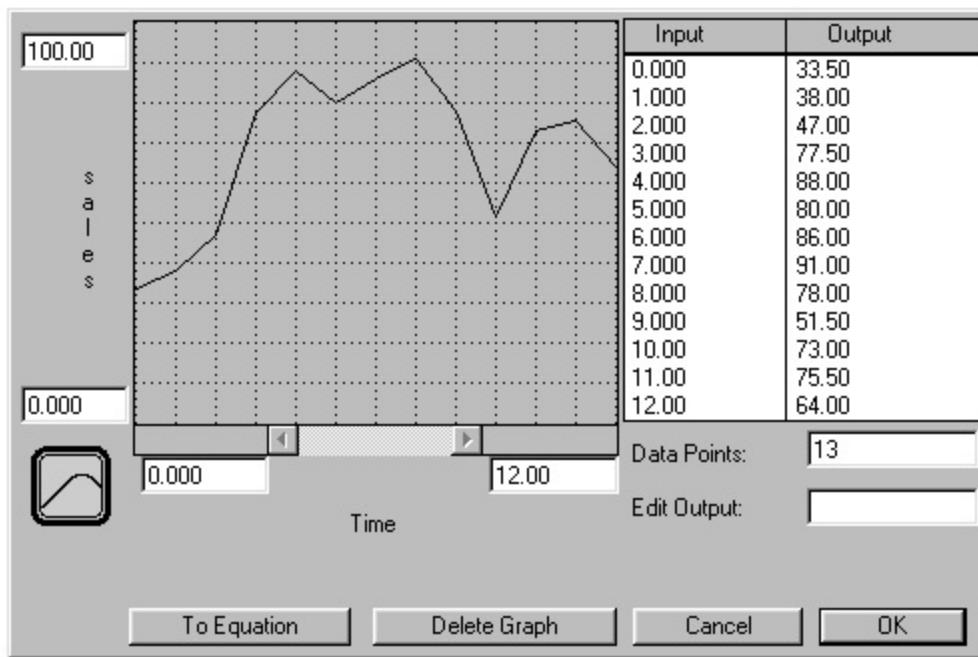
$10 * \text{SIN}(2 * \text{PI} * \text{TIME} / 12)$  generates a sinusoidal fluctuation with an amplitude of 10 and a period of 12.

My\_Bonus = IF(TIME=5) OR (Sales>5000) THEN Bonus ELSE 0

This statement sets My Bonus to the value of Bonus at simulated time 5, or whenever the value of Sales is greater than 5000. When neither condition is met, the statement gives the value 0. Sales and Bonus are defined elsewhere in the model.

As Figure 7-27 illustrates, TIME can also be used as the independent variable in a graphical function which defines a set of historical data.

*Figure 7-27  
A TIME-Dependent Graphical Function*



Because of the internal calculations associated with the Runge-Kutta methods, the TIME function will not return values equal to simulation time when you use the 2nd- or 4th-order Runge-Kutta computation methods. When your model constructs rely on the TIME function being exactly equal to simulation time, be sure to use Euler's method.

## TREND(<input>,<averaging time>[,<initial>])

The TREND function calculates the trend in input, based upon the value of *input*, the first order exponential average of input, and the exponential averaging time *averaging time*. TREND is expressed as the fractional change in input per unit time. If you do not specify *initial*, TREND substitutes the value 0 for the initial value of the trend.

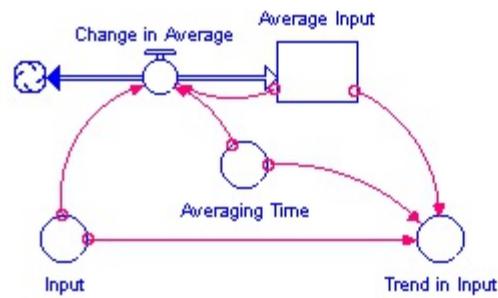
The TREND function is equivalent to the structural diagram and equations shown in Figure 7-28.

Example:

$$\text{Yearly\_Change\_in\_GNP} = \text{TREND}(\text{GNP}, 1, .04)$$

This equation calculates the annual change in the input GNP. It starts with an initial value of .04 (4% per year).

Figure 7-28  
Structure for Calculating TREND in Input



Trend\_in\_Input = (Input - Average\_Input)/(Average\_Input\*Averaging\_Time)  
Input = some variable or constant  
Average\_Input = Average\_Input+(DT)\*(Change\_in\_Average)  
INIT (Average\_Input) = Input or specified initial value  
Change\_in\_Average = (Input - Average\_Input)/Averaging\_time  
Averaging\_Time = some variable or constant

[Related Topics](#)

# Introduction to Importing and Exporting

Many models require imported numerical data from external applications such as spreadsheets. Or, you may need to export numerical data from a model to another application.

The software provides two basic mechanisms for importing and exporting data:

- By manually transferring data between a model and another application by using the [Copy and Paste commands](#).
- By creating [import](#) and [export](#) links between the model and one or more Microsoft® Excel worksheets so that you can automatically and manually import data from the spreadsheet into the model or export model data to the spreadsheet.

Linking your models to Excel worksheets provides an easy way to continually update your model with new values for use in your simulations, or to update the worksheets with the values of the latest run of your model. It is generally the most practical method when you are bringing in data from a constantly changing source, or when you want to continually update another document with the output from your models.

 [Related Topics](#)

# Setting Up Import Links

The import feature allows you to import data from an Excel worksheet into a model's variables, constants, and graphical functions.

---

**Note:** You can import data only into graphical functions or into entities whose values are defined by a constant. You cannot import data into any other entity whose values are defined by an equation.

To use the import feature, you need to create a link between the model and an Excel worksheet. The Excel worksheet contains the names of the model variables into which you want to import data and the values that you want to import.

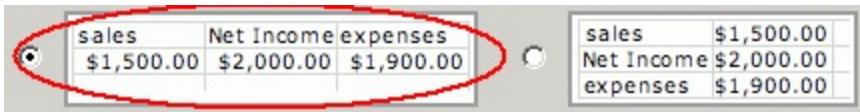
When you set up an import link, you can choose to create a temporary link or a persistent link. Temporary links allow you to import values one time only. Persistent links are saved and can be reused later to manually import values (On Demand links) or to automatically import values (Dynamic links). Dynamic persistent links import values every time you open the model, start a model run, or resume a model run. On Demand persistent links import values only when you perform a [manual import](#). Whether the import is dynamic or on demand, values are imported only if the data in the Excel file have been changed since the last import.

You can create as many import links as you want per model. You can set up export and import links to use the same or different Excel files. For example, you can choose to import values from one Excel file into your model, and then export model run values to a different Excel file to retain the "before" and "after" values in separate files. Or, if you want a single Excel file to always reflect the most current data in the model, you can import values from an Excel file and then export run values to the same Excel file.

---

**Note:** Since the software performs an import when you set up a link, you must have created a Excel worksheet with [valid data formats](#) **before** you set up the link and you must have saved all changes to the Excel file in order to import its data.

# To set up an import link

1. Open the model for which you want to set up an import link.
2. From the Edit menu, choose **Import Data**. The Import Data dialog box appears.
3. Under "Import Type", select the type of link you want to create:
  - To import data once, select the **One Time** option. Data from the Excel file will be imported into the model once, when you click **OK** in the Import Data dialog box.
  - To import data now and to use the established link to manually or automatically import data in the future, select the **Persistent** option.
4. If you selected the **Persistent** option, select how you want to import data via the link:
  - To import data only when you choose to import it, select the **On Demand** option. Data from the Excel file will be imported when you first create the link and, thereafter, only when you click the **Import Now** button in the Manage Persistent Links dialog box.
  - To import data automatically, select the **Dynamic** option. Data from the Excel file will be imported whenever you open the model and anytime the data changes in the Excel file, as long as the model isn't running at the time. If the model is running at the time the data change in the Excel file, the import will happen the next time the model run stops or pauses.
5. Under "Import Data Source", specify the Excel file from which you want to import data by clicking the **Browse** button to navigate to and select the Excel file.
6. In the Worksheet Name box, select the worksheet in the Excel file that contains the data you will import.
7. Under "Data Orientation", select the option next to the image that shows how the data in the worksheet is set up:
  - If the variable names are specified as column headings in the Excel file, select left-most option:

sales	Net Income	expenses
\$1,500.00	\$2,000.00	\$1,900.00

sales	\$1,500.00
Net Income	\$2,000.00
expenses	\$1,900.00
  - If the variable names are specified as row headings in the Excel file, select the right-most option:

	sales	Net Income	expenses
<input type="radio"/>	\$1,500.00	\$2,000.00	\$1,900.00

	sales	Net Income	expenses
<input checked="" type="radio"/>	\$1,500.00	\$2,000.00	\$1,900.00

8. Click **OK**. The data are imported into the model.

---

*Note:* If the worksheet you selected specifies variable names that are not included in the model, a warning message appears that tells you which variables were not able to be imported. All other variables are imported.

 [Related Topics](#)

# Setting Up Export Links

The export feature allows you to export data from your model to an Excel worksheet. You can choose to export all variable values in the model or only the values for variables specified in a table in the model.

To use the export feature, you create a link between the model and an Excel worksheet. The worksheet can initially be empty. The export process will automatically overwrite any information in the worksheet and set it up in the correct format.

When you set up an export link, you can choose to create a temporary link or a persistent link. Temporary links allow you to export values one time only. Persistent links are saved and can be reused later to manually export values (On Demand links) or to automatically export values (Dynamic links). Dynamic persistent links export values that have changed when the model run pauses or stops. On Demand persistent links export values only when you perform a [manual export](#).

The export link also allows you to specify the export interval. You can export one set of values (the current values for the model), or values from a run at the interval you define.

You can create as many export links as you want per model. You can set up export and import links to use the same or different Excel files. For example, you can choose to import values from one Excel file into your model, and then export model run values to a different Excel file to retain the "before" and "after" values in separate files. Or, if you want a single Excel file to always reflect the most current data in the model, you can import values from an Excel file and then export run values to the same Excel file.

---

**Note:** Since the software performs an export when you set up a link, you must have created an Excel worksheet to receive the data **before** you set up the link.

# To set up an export link

1. Open the model for which you want to set up an export link.
2. From the Edit menu, choose **Export Data**. The Export Data dialog box appears.

*Note:* If you have already created export links for all tables in the model and have created an export link that specifies the **Export all model variables** option, a message appears that tells you that you cannot create a new export link. To create a new export link, you must first [delete an existing link](#) of the type you want to create (table or "all variables").

3. Under "Export Type", select the type of link you want to create:
  - To export data once, select the **One Time** option. Data from the model will be exported to the Excel file when you click **OK** in the Export Data dialog box, and will not be exported again.
  - To export data now and to use the established link to manually or automatically export data in the future, select the **Persistent** option.
4. If you selected the **Persistent** option, select how you want to export data via the link:
  - To export data only when you choose to export it, select the **On Demand** option. Data from the Excel file will be exported when you first create the link and, thereafter, only when you click the **Export Now** button in the Manage Persistent Links dialog box.
  - To export data automatically when a model runs, select the **Dynamic** option.
5. Under "Export Data Source", select which data you want to export and how often:
  - To export all variables in the model to the Excel file, select **Export all model variables**.

*Note:* You can create only one export link that has this option selected.

- To export only the variables specified in a table, in the order in which they appear in the table, select **Export variables in table**, and then choose the table that contains the variables you want to export.

*Note:* You can create only one export link per model table. If you have already created an export link for a table, that table is not available for other another export link unless you [delete the existing export](#)

[link](#) for that table.

6. Under "Interval," select how often you want to export the selected variables:
  - To export the values as they are in the model right now, select the **One set of values** option. The Excel worksheet will be formatted so that the data can be imported back into the model later.
  - To export the values in the model after an interval in relation to the model's DT, select the **Every \_\_\_\_ <Unit of Time>** option, and then enter the interval value. For example, if the model's time unit is Months, you can export a value for every other Month by entering **2** in the box.
  - To export the values in the model for every DT in the next simulation run, select the **Every DT - Export every intermediate value during the run** option.
  - To export the values exactly as they appear in the table whose values you are exporting, select the **Use table settings** option. The exported values will use the report interval, orientation, and other settings as specified in the Define Table dialog box for the table.

*Note:* The **Use table settings** option is available only if you selected the **Export variables in table** option.

7. Under "Export Destination", specify the Excel file to which you will export the data by clicking the **Browse** button to navigate to and select the Excel file.
8. In the Worksheet Name box, select the worksheet to which you will export the data.
9. Under "Data Orientation", select the option next to the image that shows the data format used in the worksheet:

*Note:* If you selected the **Use table settings** option in step 6, the "Data Orientation" option is disabled. The exported table will use the orientation specified in the Define Table dialog box for the table.

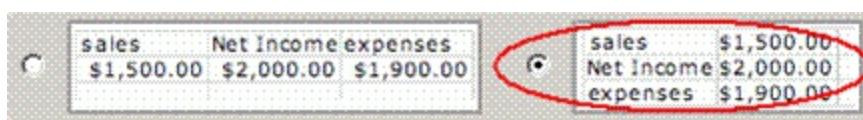
- If the variable names are specified as column headings in the Excel file, select left-most option:

<input checked="" type="radio"/>	sales	Net Income	expenses
	\$1,500.00	\$2,000.00	\$1,900.00

<input type="radio"/>	sales	\$1,500.00
	Net Income	\$2,000.00
	expenses	\$1,900.00

- If the variable names are specified as row headings in the Excel file, select the right-most option:



	sales	Net Income	expenses
<input type="radio"/>	\$1,500.00	\$2,000.00	\$1,900.00

	sales	\$1,500.00
<input checked="" type="radio"/>	Net Income	\$2,000.00
	expenses	\$1,900.00

10. Click **OK**. The selected data is exported from the model into the Excel worksheet.

 [Related Topics](#)

# Managing Persistent Import and Export Links

Once you have created one or more persistent links between your model and Excel worksheets, you can manage the links by editing them, enabling or disabling them, or deleting them.

- **Editing** a link allows you to change any aspect of the link, including its type, its associated Excel File, and its data orientation.
- **Disabling** a link allows you to temporarily turn off the link so that it doesn't import or export data while you are making changes to or experimenting with your model. **Enabling** a link turns the link back on without making any other changes to the link.
- **Deleting** a link allows you to permanently remove a link that you no longer use.

Each of these procedures is described below.

## To edit a persistent link

1. Open the model for which you want to edit import or export links.
2. From the Edit menu, choose **Manage Persistent Links**. The Manage Persistent Links dialog box appears and displays the existing persistent links for the model.
3. Click the **Edit** button for the link that you want to edit. The Edit Import Link or Edit Export Link dialog box appears, where you can make your edits.
4. Make your edits to the link.
5. When you are finished making edits, click **OK** to close the Edit Import Link or Edit Export Link dialog box.

## To disable or enable a persistent link

1. Open the model for which you want to disable or enable an import or export link.
2. From the Edit menu, choose **Manage Persistent Links** The Manage Persistent Links dialog box appears and displays the existing links for the model.
3. Disable or enable the link:
  - To disable a link, clear the **Enabled** check box for the link.
  - To enable a link, select the **Enabled** check box for the link.
4. When you are disabling or enabling links, click **OK**.

## To delete a persistent link

1. Open the model for which you want to delete import or export links.
2. From the Edit menu, choose **Manage Persistent Links**. The Manage Persistent Links dialog box appears and displays the existing links for the model.
3. Click the **Delete Link** button for the link that you want to delete. The selected link is deleted from the link list in the Manage Import Export dialog box.
4. To complete the deletion, click **OK**.

 [Related Topics](#)

# Creating Excel Import and Export Files

In order to import data from an Excel file into a model, you need to set up the Excel file so that the data is properly formatted. To export data from a model to an Excel file, you can use a blank Excel file. The export process will automatically overwrite any information in the worksheet and set it up in the correct format.

---

**Notes:** To import data from an Excel file, you must save the Excel file. Unsaved changes in an Excel file will not be imported.

If you are sharing a model that has a persistent import or export link to an Excel file, create a folder called "Data" within the folder that contains the model file. This ensures that the connection between the model and the Excel file stays active, regardless of the computer that you are running the model on.

---

The [import file format](#) and [export file format](#) are described in detail below.

# Import file format

To import values from an Excel file into a model, the Excel file must contain the names of the variables in the model into which you want to import values, and the values that you want to import.

The variable names in the Excel file must exactly match the variable names used in the model. You do not need to include all model variables in your Excel file; include only the variables whose values you want to import to the model. Model variables that do not appear in the Excel file are not affected during the import.

---

*Note:* If any of the variables in the Excel file do not match those in the model, you will receive a warning message during the import to tell you what data could not be imported.

You can enter the variable names as column headings or row headings (you will indicate the format you choose when you [set up the link](#) to your model). For example, the following Excel worksheet defines the initial values for *population* (a stock), *birth rate* (a converter), and *death rate* (a graphical function), with the variable names in the column headings:

	A	B	C	D
1	population	birth rate	death rate	
2	100	0.2	0.1	
3			0.102	
4			0.105	
5			0.113	
6			0.13	

The name of each variable appears as a column heading and the values to import for each variable appear immediately beneath it. All values for the graphical function (*death rate*), ordered from minimum  $x$  to maximum  $x$ , appear under the variable heading. If there are not enough values for the graphical function, the last value will be repeated to fill it out. If there are too many values, the excess values will be ignored.

The following example shows the same data with the variables as row headings:

	A	B	C	D	E	F
1	population	100				
2	birth rate	0.2				
3	death rate	0.1	0.102	0.105	0.113	0.13
4						

Note that the data in the spreadsheet do not have to start at row 1, column A. The import process searches the first 20 rows and 10 columns (200 cells) for a variable name, so you can begin entering data anywhere within that area. This allows you to enter header data or other documentation in the file that will not

affect the import process. You can also enter descriptive information between columns (if you are using a column format) or between rows (if you are using a row format).

Between each variable, you can have up to five columns (or rows, if you are using row format) that have a blank header line. Data can appear in any other row (or column, in row format) than the header row and these columns (or rows) will be ignored during the import.

## Import file format for arrays

**One-dimensional arrays.** To import values in an Excel file into a one-dimensional array in a model, the Excel file must contain one column (or row) of data for the array. The column (or row) heading specifies the array name. All subsequent columns (or rows) contain the data.

The following example shows a one-dimensional array ("OneD array") set up with a column heading, and with seven specified values.

OneD array	4	5	6	7	8	9	10
------------	---	---	---	---	---	---	----

**Two-dimensional arrays.** To import values in an Excel file into a two-dimensional array in a model, the Excel file must contain more than one column (or row) of data for the array. The first column (or row) specifies the array name. All subsequent columns or rows for the array contain an ellipsis ("...").

In the following example, there are three variables for the array "TwoD array", set up with column headings. For each variable, the Excel spreadsheet provides three values for each variable.

TwoD array	...	...	...
	4	2	-1
	5	3	-7
	6	4	0

If there are not enough values for the array, the last value will be repeated to fill it out. If there are too many values, the excess values will be ignored.

---

**Note:** For two-dimensional arrays, the data in Excel rows always map to rows in the model array, and the data in Excel columns always map to columns in the model array, regardless of whether there are row headings or column headings in the Excel file. The following example shows the same data as in the "TwoD array" example above, with row headings rather than column headings. Notice that the data is in the same order in both examples.

TwoD array	4	2	-1
...	5	3	-7
...	6	4	0

**One-dimensional graphical functions.** Format the data in the same way as you would for a two-dimensional array, with an ellipsis ("...") in subsequent rows or columns. For one-dimensional graphical functions, however, your specification of row vs. column headings determines how the values are read into the graphical function.

The following example shows how you would format the Excel file for a one-dimensional graphical function (called "OneD gf array") with four elements (in column headings).

OneD gf array ...	...	...	...
5	6	5	5
6	3	2	6
3	4	1	6
2	2	7	6
2	1	5	3
1	6	8	2
6	9	9	2
7	3	2	1
9	2	1	1

If there are not enough values for the elements, the last value will be repeated to fill it out. If there are too many values, the excess values will be ignored.

**Two-dimensional graphical functions.** To import values in an Excel file into a two-dimensional graphical function, format the values in row-major order (values are imported from the first row, moving from left to right, then from the second row, left to right, and so on). Succeeding columns in a row start with an ellipsis (...). To indicate the start of a new row, use "\*\*\*\*" in the first column of the row.

The following example shows the format for a 4x3 array (called "TwoD gf array") of graphical functions (1 column for each cell in row-major order).

TwoD gf array ...	...	***	...	...	***	...	...	***	...	...	...
4	5	6	5	3	5	3	2	4	6	1	5
3	6	7	4	4	4	4	4	5	4	3	3
4	4	5	2	2	3	5	5	3	2	4	2
3	3	4	3	3	2	6	6	2	2	5	2
3	2	4	2	2	3	7	7	3	4	3	3
3	3	3	3	3	2	8	8	4	5	5	2
2	4	2	4	2	3	5	9	5	6	6	2

## Import file format for conveyors and queues

To import values in an Excel file into a conveyor or queue, enter all values for the conveyor or queue as a comma-separated list in a single cell of the Excel file. For conveyors, you specify one value per unit time in the conveyor. For queues, you specify one value per element in the queue.

The following example shows the format for specifying three values to import

for a conveyor.

Conveyor	1, 2, 3
----------	---------

# Export file format

If the **One set of values** option is selected as the interval for the export link, data exported from the model to the Excel file appears in the same format as described for the import file. This allows you to use the exported values as import values at another time. If one of the other interval options is selected for the export link, data exported from the model matches is in the same format as table output. If you are using the **One set of values** option with a table, do not include more than one element from each array as every occurrence of an array element will cause the entire array to be exported.

As with the import file, you decide whether you want the variables to be listed as column headings or as row headings when you [set up the export link](#).

When you export data, the contents of the Excel worksheet are always erased before the export starts. Any text or formatting specified in the worksheet is lost when the export happens.

 [Related Topics](#)

# Manually Importing or Exporting Data

Once you have created one or more persistent [import](#) or [export](#) links, you can manually import or export data via the links at any time. Although only On Demand links require you to manually import or export data, you can also manually import or export data with Dynamic links. The manual import feature allows you to import or export data at any time you choose.

*Note:* To manually import or export data, you must have already set up a persistent [import link](#) or one or more persistent [export links](#) for the model.

## To manually import or export data

1. Open the model into which you want to manually import data or from which you want to manually export data.
2. From the Edit menu, choose **Manage Persistent Links**. The Manage Persistent Links dialog box appears.
3. Select whether you want to manually import or export data:
  - To manually import data, click the **Import Now** button. Data is imported from the file specified in the link in the Import list.
  - To manually export data, click the name of the export link you want to use in the Export list, and then click the **Export Now** button. Data is exported to the file specified in the selected link.
4. Click **OK**.

 [Related Topics](#)

# Importing and Exporting Module Variables

When you import or export data from a model that contains modules, the import and export links you set up for the model apply to all modules in the model. You do not set up separate import/export links for individual modules in the model.

To import a variable into a module, you must specify the variable's fully qualified name (*modulename.variablename*) in the [Excel file](#).

 [Related Topics](#)

# **Copy and Paste**

Copy and Paste provides the most simple method for moving data into and out of your models. It is generally the most practical when the exchange of data is a one time event. The operation of these commands is outlined below, followed by a discussion of the necessary formats.

# Exporting Data Using Copy and Paste

Use the following procedure to export data from your model by using Copy and Paste commands.

1. Set up a table in the model with the desired variables and format.
2. Run the model.
3. Click the column header to select the column of data to be exported as illustrated in Figure 8-1 (hold down the shift key to select multiple contiguous columns or the control key (Windows) or command key (Macintosh) to select noncontiguous columns).

*Figure 8-1 Selecting Table Variables for Copying*

Months	Water	rainfall
1.00	100.00	19.50
2.00	121.48	30.50
3.00	151.77	37.00
4.00	186.60	40.00
5.00	230.26	60.50
6.00	287.05	64.00
7.00	347.30	70.50
8.00	415.61	84.50

4. Choose **Copy** from the Edit menu. The data is automatically stored on clipboard included in your system.

---

*Note:* If you hold down the alt key (Windows) or option key (Macintosh) when choosing **Copy**, you will also copy the variable name along with the data in the column

---

5. Open the document which is to receive the data.
6. Paste the data into the recipient application.

# Importing Data using Copy and Paste

Use the following procedure to import data into your model by using Copy and Paste commands.

1. Copy the data set from another application following the appropriate directions. The data is automatically stored on the clipboard included in your system.
2. Switch to your model, open the variable dialog which is to receive the data.
3. Select all pre-existing data that is to be replaced by the pasted set.
4. Use the key commands CTRL+V (Windows) or command-V (Macintosh) to deposit the data.

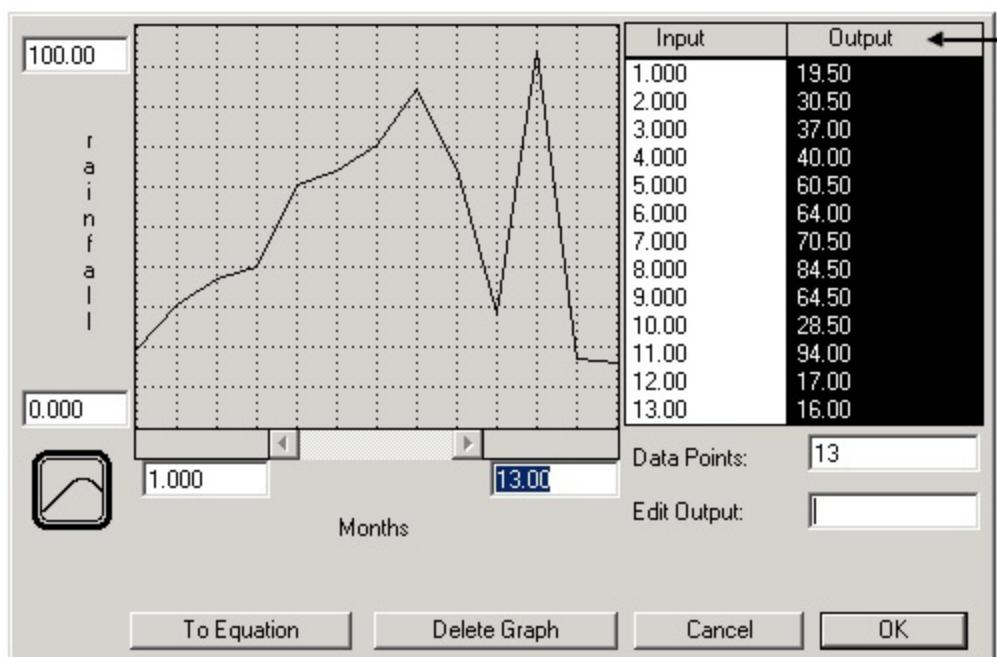
---

*Note:* Copy and Paste are grayed out in the Edit menu when a variable dialog is opened, necessitating the use of key commands.

You can paste numeric data into any model variable. However you must ensure that the number of data points is appropriate to the location. The necessary formats are as follows:

- A graphical function created within a flow or converter can hold up to 1500 data points (numbers only). For more information about creating graphical functions with flows and converters, see Flows and Converters. The data will be pasted into the Output column on the top, right side of the dialog. You can select all or part of the column. To select the entire column at once, click once on the word Output at the top of the column, as illustrated in Figure 8-2.

*Figure 8-2  
Pasting Data into a Graphical Function*



Click on the word Output to highlight the entire Output column.  
Paste the data into the Output column by typing control-V (Windows) or command-V (Macintosh).

- Reservoir, flow, and converter dialogs can be recipients of an equation or constant. All pre-existing contents of the equation box should be highlighted first, if they are to be replaced by the incoming data.
- Queues and Conveyors can be the recipient of single or multiple constants for their initial values, but multiple numbers must be separated by commas. All pre-existing contents of the equation box should be highlighted first, if they are to be replaced by the incoming data.

*Note:* If the data is being imported from a spreadsheet, a column of commas can be created alongside the column of data. Copying them together will bring the data in the necessary format.

	A	B
1	100,	
2	200,	
3	300,	
4	400,	
5	500,	
6	600,	
7	700,	
8		
9		

- Ovens can be the recipients of constant initial values. All pre-existing contents of the equation box should be highlighted first, if they are to be replaced by the incoming data.

 [Related Topics](#)

# Pasting Data into Arrayed Variables

Note: Importing data into an arrayed variable in your model via Paste follows the same basic process as for non-arrayed variables. There are several exceptions to note. For more information about working with Arrays, see [Introduction to Arrays](#).)

- One data set can be applied to all elements of a graphical function if pasted while Apply To All is checked. It is not possible to Paste into all Array elements in the graphical function with one Paste command when Apply To All is not checked - you will need to paste into each element separately.
- A data set can be pasted into the equation box of an arrayed Reservoir or Oven, or an arrayed flow or converter without Required Inputs (one value will be put in each element's equation box), if:
  - Apply To All is checked in the arrayed variable's dialog.
  - There are the same number of data points in the data set to be pasted as there are cells in the arrayed variable (number of cells = number of elements in column \* number of elements in row).
  - The data are enclosed in square brackets [] and all data points are separated by commas. The correct format for a column of data is illustrated here:

	A	B
1	100	,
2	200	,
3	300	,
4	400	,
5	500	,
6	600	,
7	700	]

Note: If the data are being imported from a spreadsheet, a column of commas can be created alongside the column of data. Square brackets should be put in front of the first number and after the last number. Copying them together will bring the data in the necessary format.

- A data set that defines the initial value(s) of an individual element within an arrayed Conveyor or Queue can be pasted if the data are separated by commas, as seen here. If Apply To All is selected, the initial values will apply to all elements within the Conveyor or Queue.

	A	B
1	100	,
2	200	,
3	300	,
4	400	,
5	500	,
6	600	,
7	700	
8		
9		

 [Related Topics](#)

# Introduction to Causal Loop Diagrams

A useful way to explain or communicate your model is to present it as a Causal Loop Diagram (CLD). A CLD allows you to show only the dominant feedback loops or selected causal connections between entities in your model.

By using a CLD, others can quickly see the overall causal relationships in the model without being distracted by the other details that are necessary to simulate the model. The software allows you to create two types of CLDs:

- **Hybrid CLDs** are used to communicate the feedback loops in an existing model; they require at least one Stock.
- **Regular CLDs** are used to create maps of the high-level causal relationships by showing words (names) and arrows that indicate the direction of causality. Regular CLDs do not require a Stock.

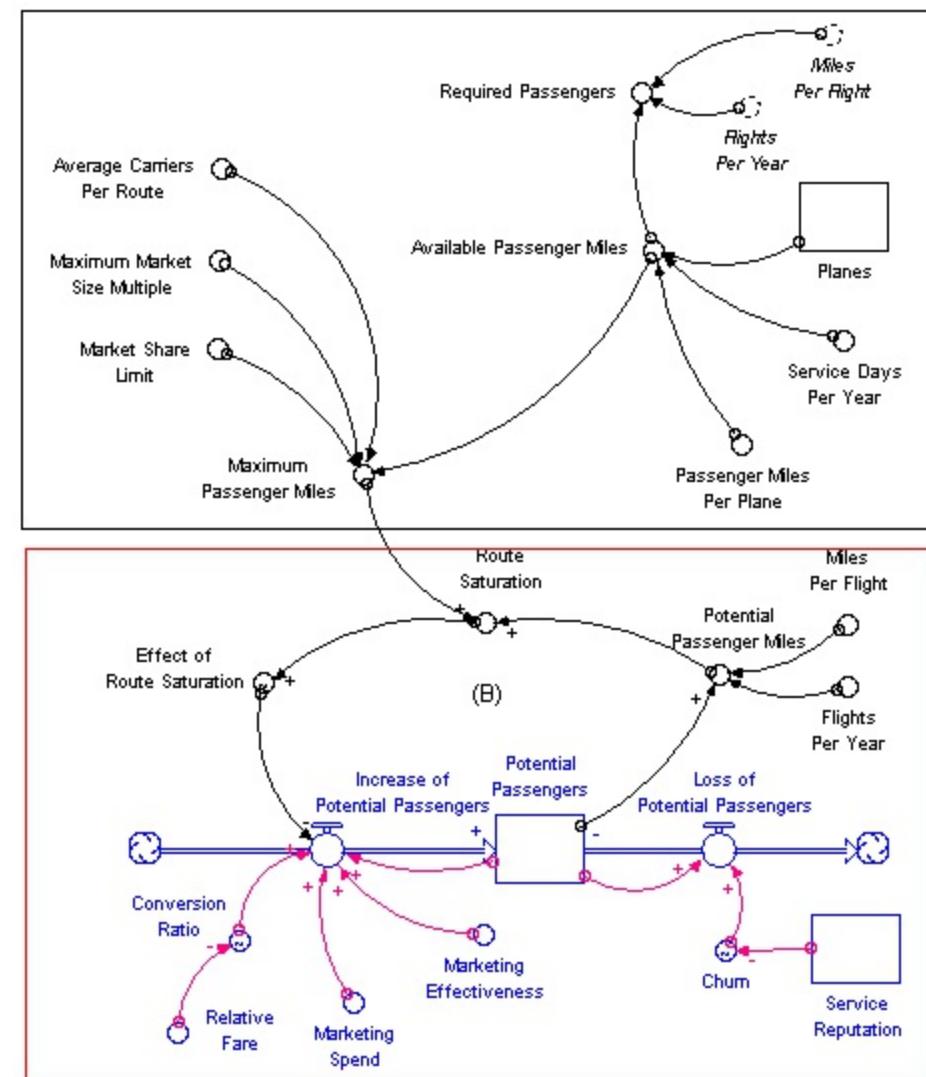
This section provides an overview of the CLDs you can create with **iThink** and **STELLA**, and it provides step-by-step procedures for creating CLDs.

 [Related Topics](#)

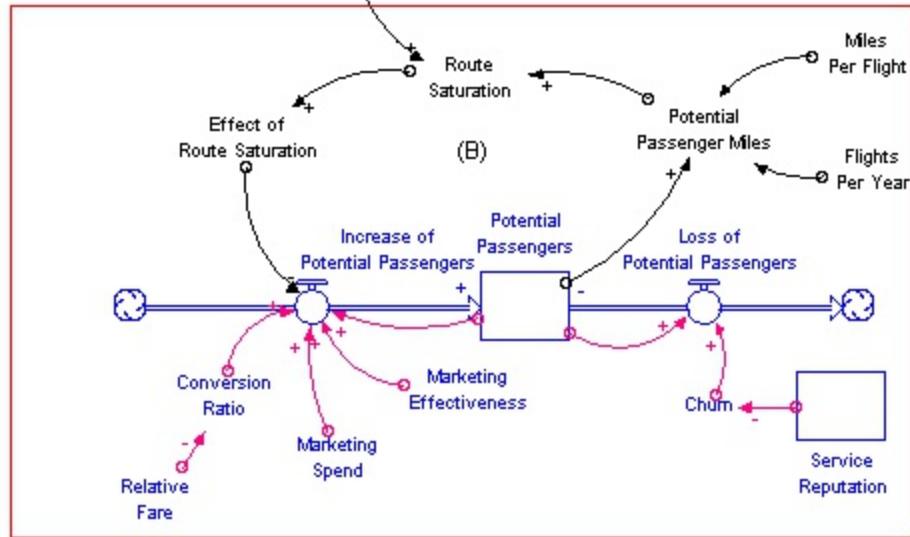
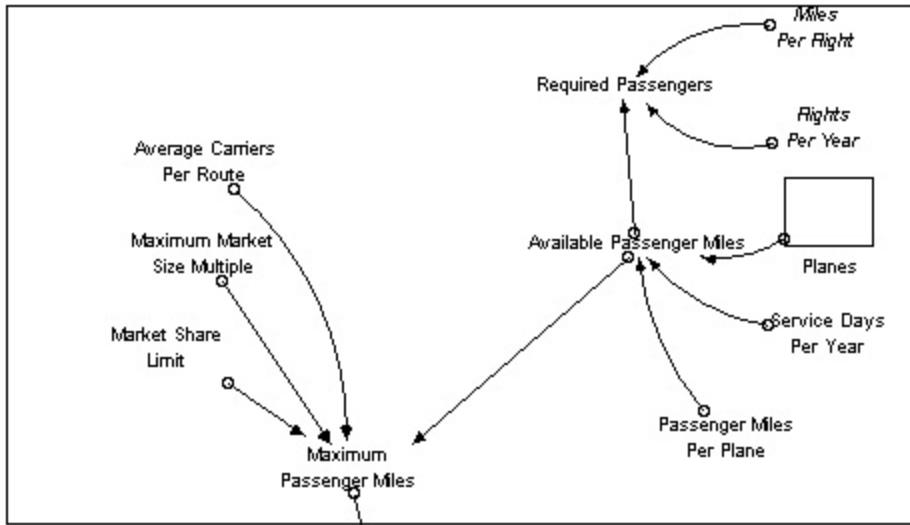
# Overview of Causal Loop Diagrams

The following images illustrate how you can transform an existing model into a Causal Loop Diagram.

Suppose you have created the following model:



If you have a complex model like this, you might want to simplify it to show only the dominant feedback loops as a Causal Loop Diagram (CLD). You can do this by first selecting **Name only** as the converter type in the Model Preferences dialog. This simplifies your model to look like this:



You may also want to more clearly indicate the feedback loops by assigning polarity (+/s or -/o) to the connectors and flows, as shown above. The center of the balancing feedback loop in the above model is indicated by inserting a Text Box that contains a "B" (for "balancing") and clicking the **Format for CLD** button in the [Text Box dialog box](#).

For a simple model, just using Name only converters and polarity may be enough to create a CLD. For a more complex model like the one shown above, however, you may want to select only the dominant feedback loops to present as a CLD. The first step to converting a model with multiple feedback loops into a CLD is to create a storytelling button on the Interface layer and to build a story that contains only the elements you want to include in the CLD.

The following image shows what the Create Story dialog box looks like for the model shown above.

**Note:** Selected elements in the image below are indicated with an asterisk (\*).

## Create Story

### Story Sequence

- \ \* Maximum\_Passenger\_Miles
- \ Maximum\_Passenger\_Miles ->Route\_Saturation
- \ \* Potential\_Passengers
- \ \* Increase\_of\_Potential\_Passengers
  - \ Potential\_Passengers ->Potential\_Passenger\_Miles
  - \ Potential\_Passenger\_Miles ->Route\_Saturation
  - \ Route\_Saturation ->Effect\_of\_Route\_Saturation
  - \ Effect\_of\_Route\_Saturation ->Increase\_of\_Potential\_Pa...
- \ \* Route\_Saturation
- \ \* Potential\_Passenger\_Miles
- \ \* Effect\_of\_Route\_Saturation
  - \ Flights\_Per\_Year ->Potential\_Passenger\_Miles
  - \ Miles\_Per\_Flight ->Potential\_Passenger\_Miles
- \ \* Miles\_Per\_Flight
- \ \* Flights\_Per\_Year

Build Story

Group

Delete Selection

Move Up

Move Down

Insert Annotation

### Annotation Type

- Text       Sound  
 Graphic     Movie

Edit Text

Insert Highlight Break

Highlight Selected Item

Chapter:

New

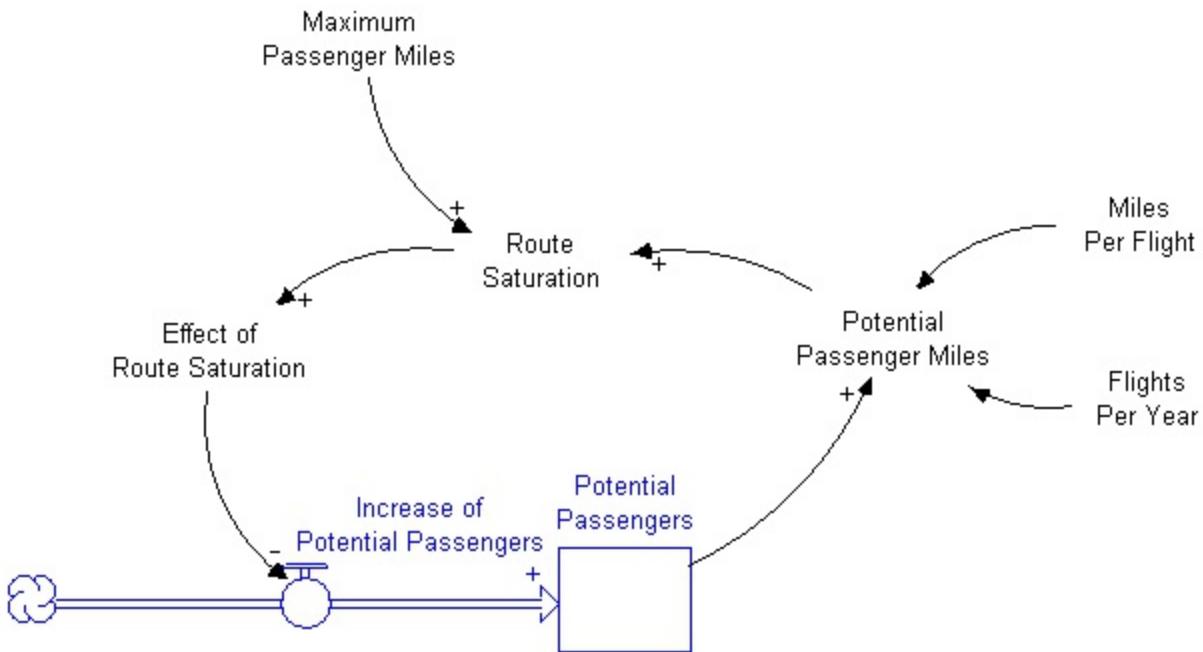
\* -> +

Delete Chapter

Cancel

OK

When you click the storytelling button you've created, the resulting CLD appears on the Storytelling tab:



[Related Topics](#)

# Creating Causal Loop Diagrams

You can create a Causal Loop Diagram from scratch, by using Name Only Modules to map out the high-level causal relationships. You can also create a Hybrid Causal Loop Diagram from an existing model by changing all converters in the model to Name only converters and then using the storytelling feature to select the elements that you want to include in the CLD.

Both procedures are described below.

# Creating a Causal Loop Diagram from scratch

1. Open a new model file.
2. From the Model menu, choose **Model Prefs**. The Model Preferences dialog box appears.
3. Under "Options", select the **Name Only Modules** check box.
4. Click **OK**.
5. On the Model or Map layer, click the Module tool in the toolbar and then click the diagram surface to place modules. Rename the modules to text that describes the connections you want to show. Use connectors to link the modules.
6. Choose the polarity for connectors and flows in your CLD, by right-clicking each connector and then selecting the appropriate polarity (+/s or -/o).
7. Place a Text Box in the center of the CLD. Open the Text Box dialog box, click the **Format for CLD** button to reformat the Text Box appropriately for a CLD, and then click **OK**. In the Text Box, type "R" for reinforcing, "B" for balancing, or any other symbol you want to describe the CLD. If you do not type anything, the software places a "+" sign in the Text Box.

# Creating a Hybrid Causal Loop Diagram from an existing model

1. Create or open the model from which you want to create a Causal Loop Diagram.
2. Go to the Model or Map layer.
3. From the Model menu, choose **Model Prefs**. The Model Preferences dialog box appears.
4. Under "Converters", select **Name only**.
5. Click **OK**. The converter icons disappear and only the names associated with the converters remain.
6. Go to the Interface layer.
7. In the Interface layer toolbar, click the  button and hold the mouse button down to see the other available button types.
8. Select the  button. The pointer changes to .
9. Click the Interface layer to place the new storytelling button on the layer. The Create Story dialog box appears.
10. Click the **Build Story** button. The model appears on the Story Building tab.
11. Select the elements that you want to include in the CLD by using one of the following methods:
  - Click and drag over all element icons in the model to select them all. Using this method automatically groups the elements into a single group.
  - Click the icon for each element that you want to include, one at a time.
  - Use SHIFT+click to select multiple elements.
    - When an icon is selected, it is highlighted in blue.
12. To deselect any selected element, click its icon again.
13. When you are finished selecting elements, click  above the navigation tabs. The Story Building tab disappears, the Create Story dialog box appears again, and the selected elements appear in the Story Sequence box.
14. In the Story Sequence box, select all of the listed elements by using SHIFT+click or CTRL+click.

- If you used only the click-and-drag method to select elements, the elements are already in a single group. Go to step 16.
- If you used any of the other methods to select elements, you need to group them into a single group.

15. Click the **Group** button to group the elements. For more information about using the options in the Create Story dialog box, see the [Storytelling](#) section in [Button Dialog Operations](#).

---

*Note:* If the **Group** button does not appear, it means that you have multiple groups of elements in the story sequence. To create a single group, select all of the listed elements, click the **Ungroup** button, select all of the listed elements again, and then click the **Group** button.

---

16. Click **OK** to close the Create Story dialog box
  17. Click **OK** to close the Button dialog box.
  18. Choose the polarity for connectors and flows in your CLD, by right-clicking each connector and flow and then selecting the appropriate polarity (+/s or -/o).
  19. You can now view and share your CLD by clicking the storytelling button you created.
- 

*Tip:* You can also share your CLD by copying and pasting it into a Microsoft Word or PowerPoint document. To copy a CLD, view the CLD by clicking the storytelling button, choose **Select All** from the Edit menu, and then choose **Copy** from the Edit menu. You can then paste the copied CLD into another document.

---

 [Related Topics](#)

# Introduction to Sub-Models

Sub-models provide an excellent mechanism for managing diagram complexity. Sub-models enable you to "drill down" into the various stages of a process, to represent sub-processes with a greater degree of detail. The space within which each Sub-model is represented can be selectively shown on the diagram, or hidden from view. As a result, you can control the pace at which diagram detail is revealed. By managing detail, Sub-models can keep you (as well as users of your models) in control of the complexity.

The topics in this section provide a practical guide to the creation and use of Sub-models. Throughout the section, you'll find hints and tips which will help you to make most effective use of the software's Sub-model capabilities.

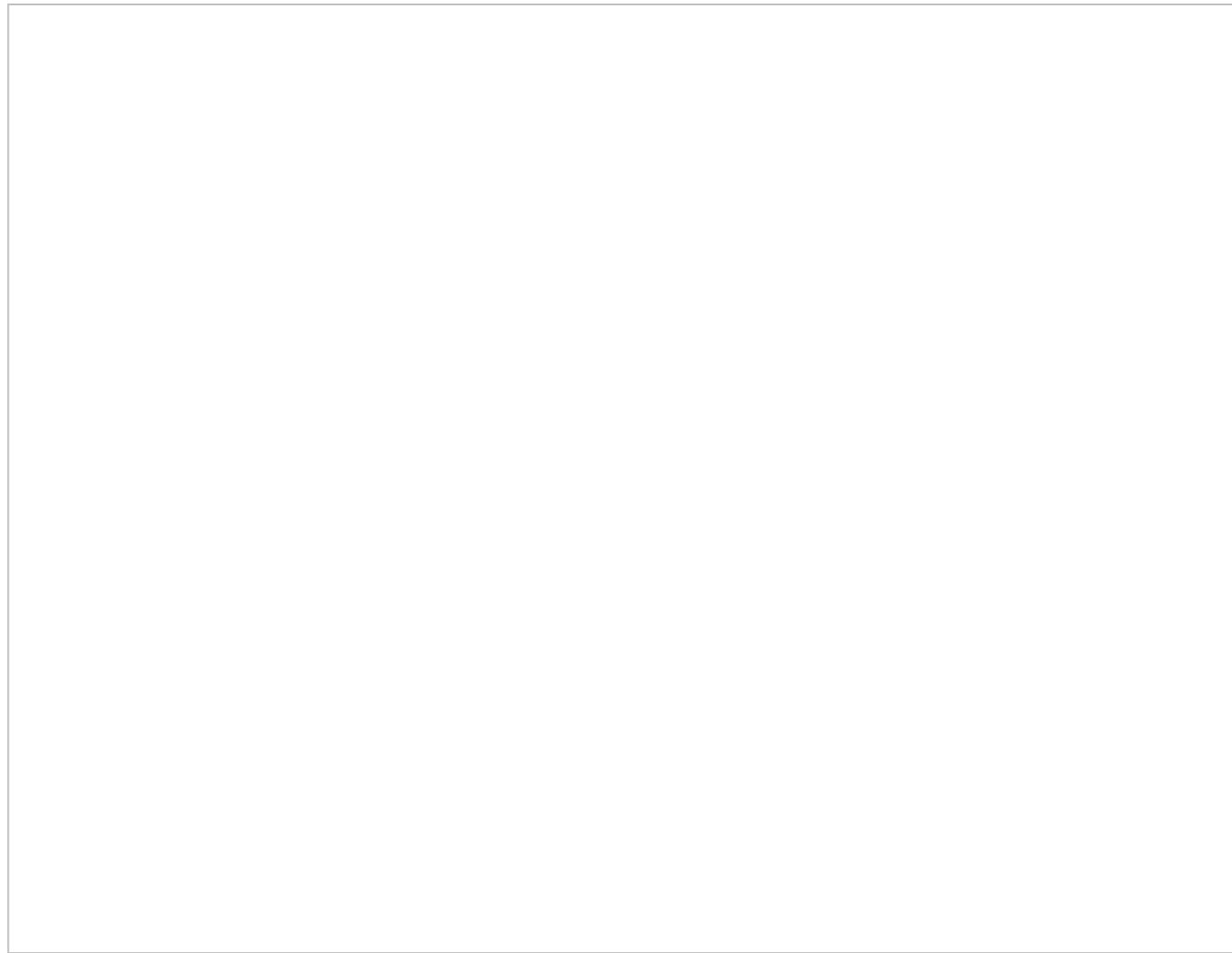
 [Related Topics](#)

# Overview of Sub-models

This section provides a brief introduction to Sub-model capabilities. In it, you'll learn about what a Sub-model is, and how one operates. You'll also learn about how the diagram display can change when you open, and subsequently close, a Sub-model.

A Sub-model is a device for encapsulating detailed structure within a model. Without Sub-models, the stock and flow structure of a model will have a "flat" appearance on the diagram. As suggested by Figure 9-2, flatness can make a model difficult to "read."

*Figure 9-2 Without Sub-models, the Diagram can be Visually Overwhelming*

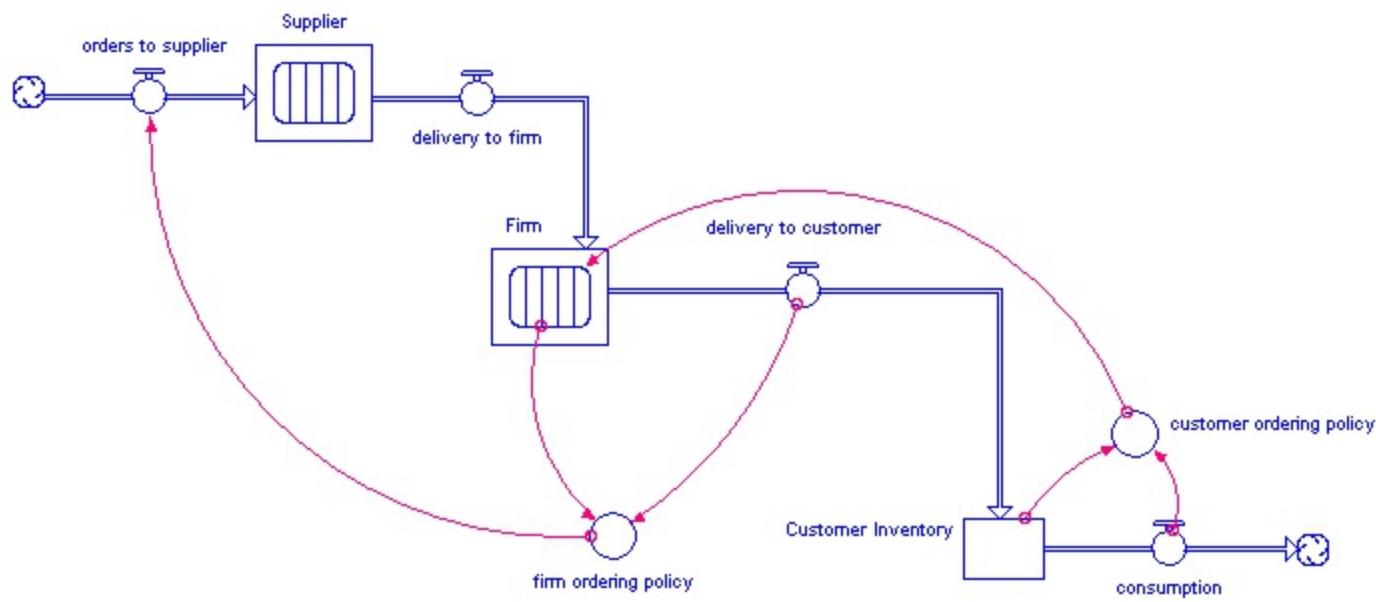


In Figure 9-2, it's hard to know where to start. Because the entire model exists on a single layer, there is no way to differentiate between the details and the essence. With a highly detailed model that spans several pages, the problem can become even more severe.

By encapsulating structure, Sub-models add layering to the diagram. In so

doing, Sub-models make it much easier to separate the essence from the details. As an illustration, Figure 9-3 takes advantage of Sub-models to represent the same process as shown in Figure 9-2. This simpler representation captures the heart of material flow from the supplier, through the firm, and to the customer.

*Figure 9-3  
Sub-models Simplify Diagram Display*



Note the large, conveyor-like icons for the Supplier and the Firm. These are called Sub-model icons. Within each icon, you'll find a Sub-model. Sub-models can be opened or closed selectively, providing you with access to the details of the underlying process.

For example, imagine that you wanted to drill down into the logic of the Supplier order fulfillment process. Double-clicking on the Supplier icon will open its Sub-model, as illustrated in Figure 9-4. After exploring the Sub-model, you can click on the up-arrow that connects the Sub-model space with its icon. The Sub-model will be closed once again.

*Figure 9-4  
Selectively Display a Sub-model*

Layering introduces the potential for Sub-models to obscure main-level structure. For example, consider the map in Figure 9-5. The stock of Workers is used to drive the production activities within the Work in Process Sub-model.

*Figure 9-5  
A Typical Sub-model Configuration*

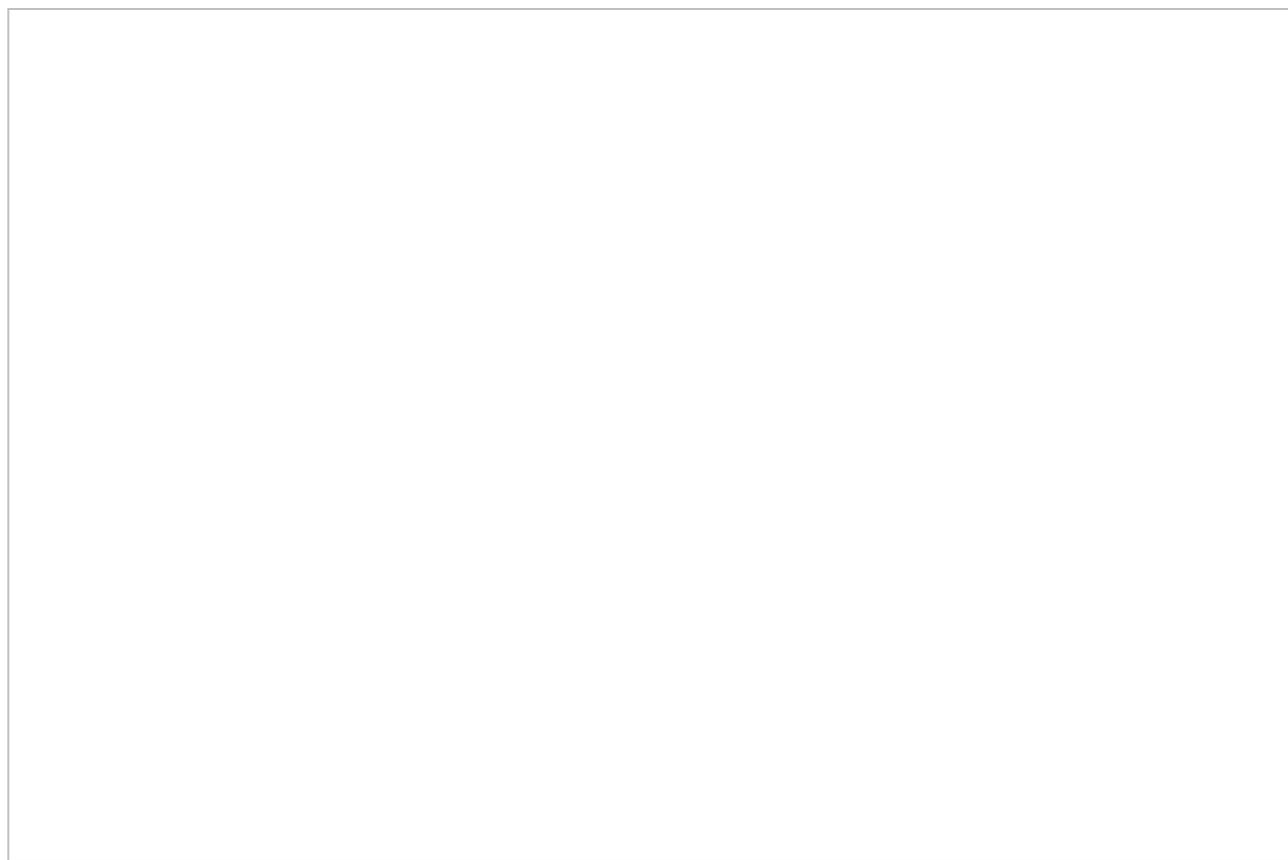
Clearly, there is potential for a topological difficulty with this diagram. When the Sub-model icon Work in Process is opened, the Sub-model that will be ejected will lay right on top of Workers.

Or will it? In designing the Sub-model feature, we anticipated the potential for

Sub-models to obscure main level structure. To get around the problem, the software allows all main level building blocks and objects to have two positions. Each building block and object will remember the two positions you have set for it. The first position is the position you establish when no Sub-model spaces are open. The second is the one you establish when one or more Sub-model spaces are open. When you open any Sub-model, all main level building blocks and objects will move to their second position. When you close all Sub-models, all main level building blocks and objects will locate themselves in their first position.

Continuing with the illustration, Figure 9-6 shows the result of opening Work in Process.

*Figure 9-6  
Two-position Display of Diagram Elements*



Notice that the structure for Workers, hiring and layoffs, and target inventory was "pushed aside" when the Sub-model was opened. These building blocks were "pushed aside," because a second position had been previously set for them while the Sub-model was open. Now, the structural elements know both their first (all Sub-models closed) and second (any Sub-models open) positions. Closing the Sub-model will cause the structure to move back to its previous position. Opening the Sub-model will push the structure aside once again.

 [Related Topics](#)

# **Creating a Sub-model Space and Basic Sub-model Operations**

In this section, you'll learn:

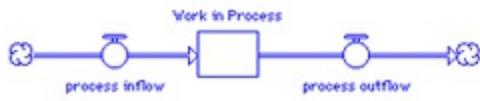
- How to create a Sub-model space
- How to open, close, select, move and resize a Sub-model space
- Constraints to moving and resizing a Sub-model space

# Creating a Sub-model Space

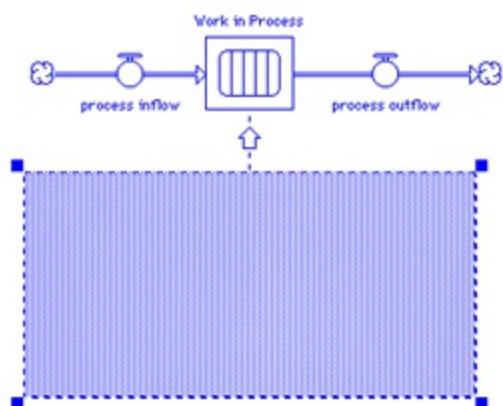
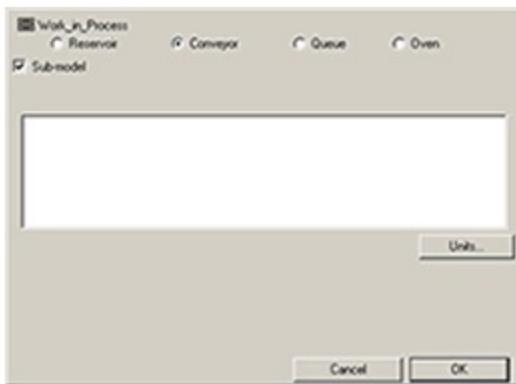
As illustrated in Figure 9-7, creating a Sub-model space is easy, involving four simple steps.

*Figure 9-7 Creating a Sub-model Space*

1. Create a Stock with at least one inflow and one outflow.



2. Define the stock as a Conveyor.  
3. Click the Sub-model check box.



4. When you click OK, the stock's icon will change to a Sub-model icon. The main level structure will turn gray. The Sub-model space will be ejected beneath the icon.

---

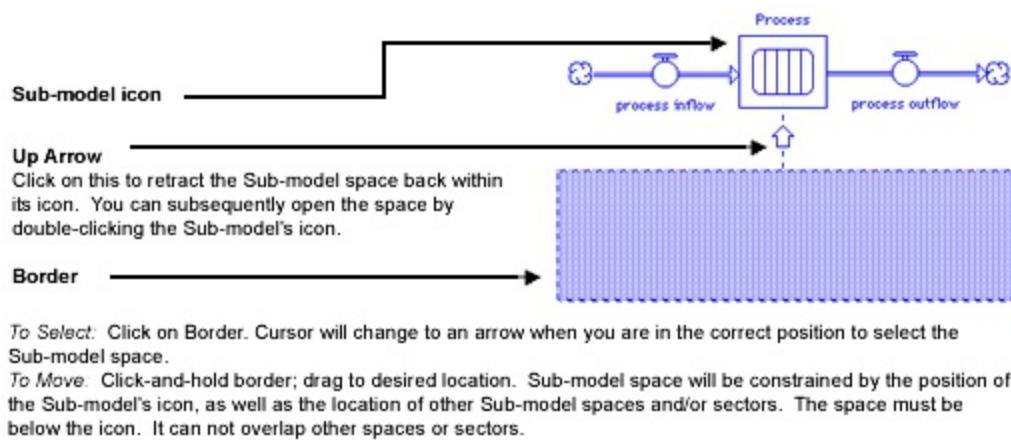
**Note:** If you create Sub-models on the Map layer (as illustrated in Figure 9-7), you'll not get distracted by dialog details. It's possible to create Sub-model spaces when no flows are attached to the stock. However, you'll find it easier in the long haul if you think through the high-level stock and flow structure of your model, before you drill down to create Sub-models.

# Opening, Closing, Selecting, Moving and Resizing

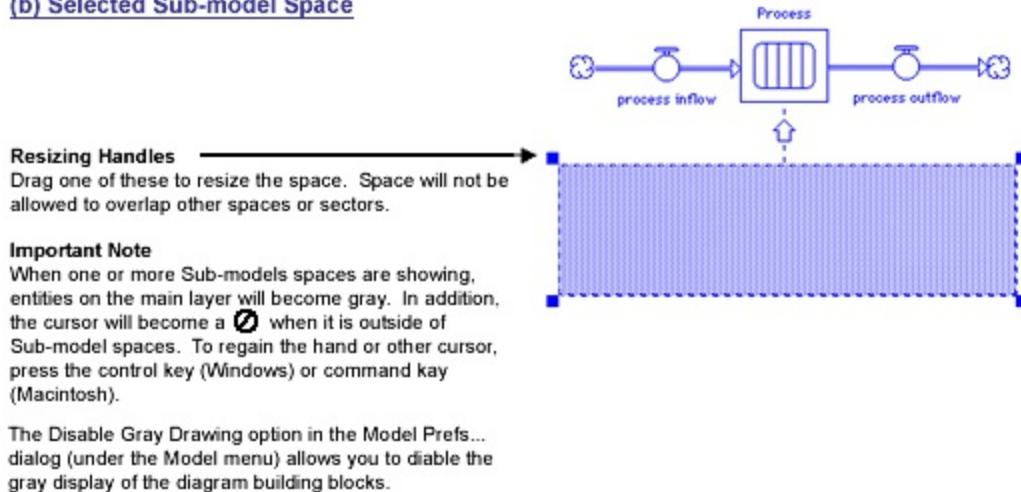
Figure 9-8 identifies the components of a Sub-model, and illustrates how to open, close, select, move, and resize a Sub-model space.

*Figure 9-8  
Opening, Closing, Selecting, Moving and Resizing a Sub-model Space*

## (a) Unselected Sub-model Space



## (b) Selected Sub-model Space



# Constraints on Moving and Resizing a Sub-model Space

The software requires Sub-model spaces to be drawn below their icons. As a result, the software will enforce a minimum 40-pixel distance between the bottom of the Sub-model icon, and the top of its associated space. In addition, the software will not allow you to move either the left or right vertical border of the Sub-model space beyond the horizontal center of the Sub-model icon.

The positioning of a given Sub-model space will also be constrained by the position of other Sub-model spaces on the diagram. Specifically, while the software *will* allow Sub-model spaces to overlap one another, only *one* of the overlapping Sub-model spaces can be open at a time. When you open one overlapping sub-model space, all other overlapping sub-model spaces will be closed.

 [Related Topics](#)

# Mapping in Sub-model Spaces

In this section, you'll learn about:

- Creating maps in a Sub-model space
- Modifying main level maps, when one or more Sub-model spaces is open
- Drawing connectors from main level building blocks to building blocks within a Sub-model space
- Drawing connectors from building blocks within a Sub-model space to main level building blocks
- Using objects when a Sub-model space is open

## Creating Maps in a Sub-model Space

When a Sub-model space is open, all Map and Model layer building blocks are available to you. To create a map within a Sub-model space, simply connect stocks, flows, converters and connectors as you would on the main level of the Map or Model layers. Your Sub-model map can be as detailed as you require. As you create structure within a Sub-model space, you should be aware of four important rules of Sub-model grammar. These are listed below.

- The stock and flow structure within a Sub-model space must be self-contained. The software will not allow you draw a flow across the boundary of a Sub-model space. However, it will allow a connector to cross the boundaries.
- The Sub-model space must contain at least one stock, with both an inflow and an outflow attached. You'll need these to make the Sub-model runnable when you shift to modeling mode.
- You cannot put a Sector Frame within a Sub-model space. The Sector Frame object will be gray as long as you have one or more Sub-models open.
- You cannot turn a stock within a Sub-model space into another Sub-model. Similarly you cannot place a Decision Process Diamond inside a Sub-model. The software supports only one level of drill-down.

# Modifying Main Level Maps

Whenever one or more Sub-model spaces are open on the diagram, the software assumes that you want to work within one of them. As a result, entities on the main level are gray (this can be overridden within the [Model Prefs... dialog](#)).

In addition, whenever you move the cursor outside of a Sub-model space, it will change to an international prohibition symbol ( $\emptyset$ ). When you try to accomplish something with the  $\emptyset$  cursor, all you'll get is a "beep" from the software. To override the  $\emptyset$  and gain the use of building blocks, tools and objects on the main level, depress the control key (Windows) or command key (Macintosh). For example, to place a converter on the main level while a Sub-model space is open, you can select the converter. Then, control-click (Windows) or command-click (Macintosh) somewhere on the diagram to deposit the converter. To open a main-level entity, control-double-click (Windows) or command-double-click (Macintosh).

# Drawing Connectors from Main Level to Sub-model

As you create a map within a Sub-model space, you'll often find that you need to make a connector linkage from a building block on the main level, to some building block within the Sub-model space. To achieve this end, you have two basic options.

## *Option 1: Use a Ghost*

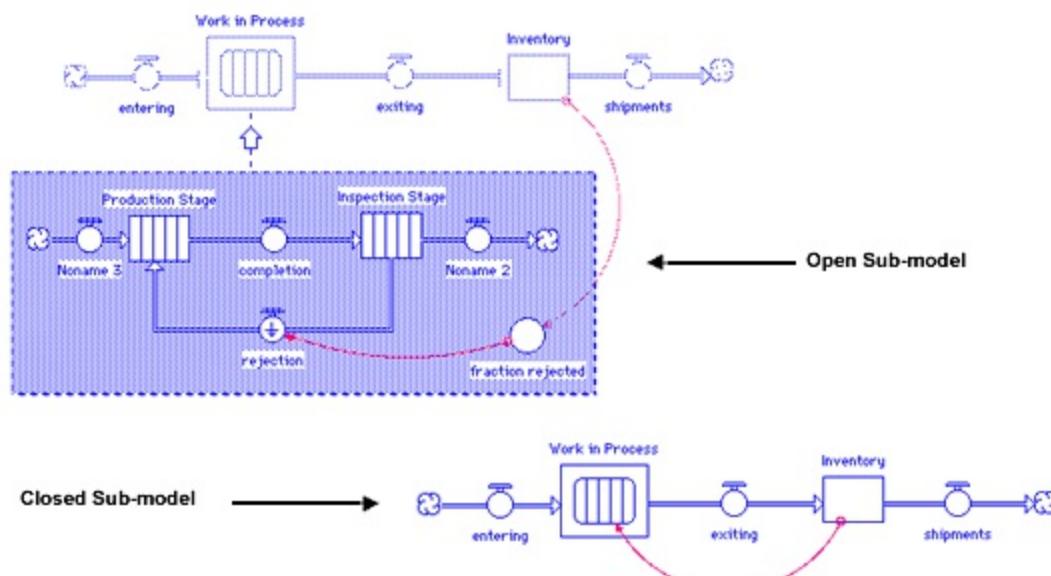
- Select the Ghost tool.
- Control-click (Windows) or command-click (Macintosh) on the main level building block.
- Click within the Sub-model space to deposit the ghosted image.
- Select the connector, and connect the ghosted image to the desired Sub-model building block.

## *Option 2: Make a direct connection*

- Select the connector.
- Depress the control key (Windows) or command key (Macintosh).
- Make the desired connection.

As shown in Figure 9-9, the connection is displayed, both when the Sub-model is open and when it is closed.

*Figure 9-9 Drawing Connectors from Main Level to Sub-Model*



**Drawing Connectors from Sub-model to Main Level:** When making a connector linkage from a Sub-model to a building block on the main level, you again have two options.

*Option 1: Use a Ghost*

- Select the Ghost tool.
- Click once on the desired Sub-model building block.
- Move the cursor out of the Sub-model space.
- When you get near the target, depress the control key (Windows) or command key (Macintosh), and click to deposit the ghost of the entity.
- Select the connector, and control-drag (Windows) or command-drag (Macintosh) to connect the ghost to the desired building block.

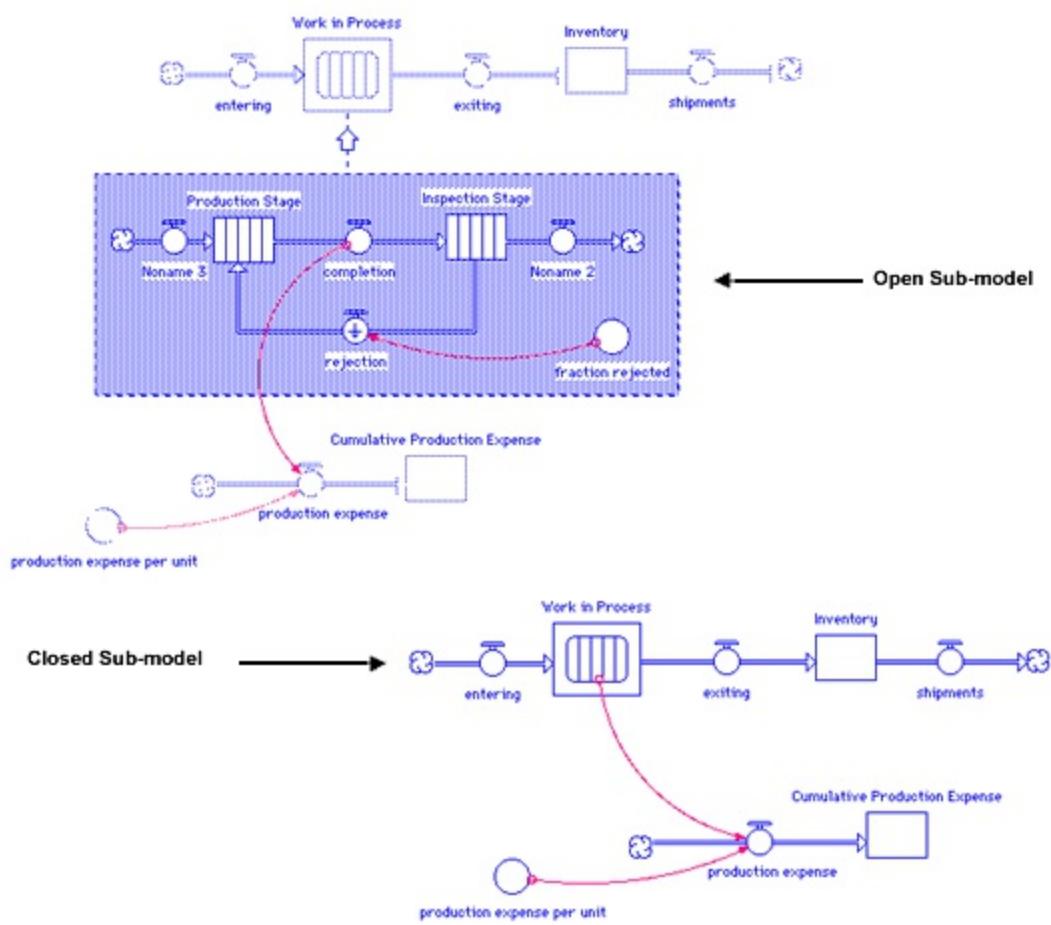
*Option 2: Forge a direct connection*

- Select the connector, and forge the connection as you normally would.

As illustrated in Figure 9-10, the software will show the connection between the Sub-model and the main level, both when the Sub-model is open and when it is closed.

*Figure 9-10*

*Drawing Connectors from Sub-model to Main Level*



**Using Objects when a Sub-model Space is Open:** The operation of Map and Model layer objects (the Sector Frame, the Decision Process Diamond, the Graph Pad, the Table Pad, the Numeric Display, and the Text Box) varies, depending upon whether a Sub-model space is open. It's important for you to be aware of these differences in operation. We'll look briefly at each object.

- **Sector Frame** – The Sector Frame is grayed out whenever a Sub-model is open. Sectors can exist only on the main level. To create a new Sector Frame (or to paste a copy of an existing one), you must first close all Sub-models.
- **Decision Process Diamond** – You cannot place a Decision Process Diamond within a sub-model space. The software supports only one level of drill-down.
- **Graph and Table Pads** – The Graph Pad and Table Pad objects are accessible, both when a Sub-model is open and when it is closed. A click within a Sub-model space will deposit the Pad icon within the space. A control-click (Windows) or command-click (Macintosh) will deposit the Pad icon on the main level. When a Pad icon is within a Sub-model space, it will take on the selective display characteristic of Sub-models. Opening the Sub-model will show the Pad icon, while closing the Sub-model will hide

the Pad icon.

Note, however, that Graph and Table Pads themselves will not take on the selective display characteristic. Unpinned Pads will exist in their own windows. When you open an unpinned pad, it will "float" above the surface of the diagram. Pinned pads will be pinned to the main level of the diagram, not the Sub-model space. If a Sub-model contains a Pad icon, and the Pad is pinned to the diagram, closing the Sub-model will leave the Pad showing on the diagram.

- **Numeric Display** – The numeric display can be selected from the object palette while a Sub-model is open. After you select the Numeric Display, you must position the cursor away from the Border of the Sub-model space to allow the entire display to fit within the space. As you move the cursor towards the middle of the space, the cursor will turn into the Numeric Display icon. Then click to deposit it in the Sub-model space.
- **Text Block** – The Text Box can be selected from the object palette while a Sub-model is open. However, the software will allow you to deposit a Text Box within a Sub-model space only when the default size of the Block will fit within the space. As a result, if you select a Text Box when a Sub-model is open, you will encounter the Ø cursor when you slide the cursor onto the diagram - unless you move the cursor toward the top-left edge of the Sub-model space. When you see the cursor turn to that of the Text Box, click to deposit the block within the space. To deposit the Text Box on the main level, you must first depress the control key (Windows) or command key (Macintosh).
- **Button** – You cannot place a button within a Sub-model space. You can press a button that appears on the main level by simultaneously pressing the control key (Windows) or the command key (Macintosh). If the button is configured to navigate to the Interface layer, the software will close any open Sub-models before leaving the Map or Model layer.

 [Related Topics](#)

# Working with Two-Position

This section looks at the two positions associated with main level building blocks and objects, when Sub-models exist in the model. The section begins by showing how to establish the first and second position for building blocks and objects. Second, the section provides a set of tips for working in the two-position environment.

**Establishing the First and Second Positions:** Whenever one or more Sub-models exist in the model, main level building blocks and objects (stocks, flows, converters, connectors, Sub-model icons, Sector Frames, Decision Process Diamonds, Pad icons, Numeric Displays, Text Boxes) can have two positions. The first position is the position that you set for the building block or object, when all Sub-models are closed. The second position is the one that you establish when one or more Sub-models is open on the diagram. Note that pinned Graph and Table Pads have only one position, which is a part of the main level. This position is maintained both when Sub-models are open and when they are closed. Unpinned Graph and Table pads are not considered to be part of the diagram. Instead, when an unpinned pad is open, it "floats" above the diagram.

Establishing these two positions is easy. To establish the first position, simply use the Hand tool to move the building block or object to the desired location. Selecting and moving building blocks and objects is discussed extensively in [The Arrow](#). To establish the second position, control-click (Windows) or command-click (Macintosh) and move the main level building block or object, while one or more Sub-models are open. When all Sub-models are closed, main level building blocks and objects will assume their first position. When one or more Sub-models are open, the entities on the main level will move to their second position.

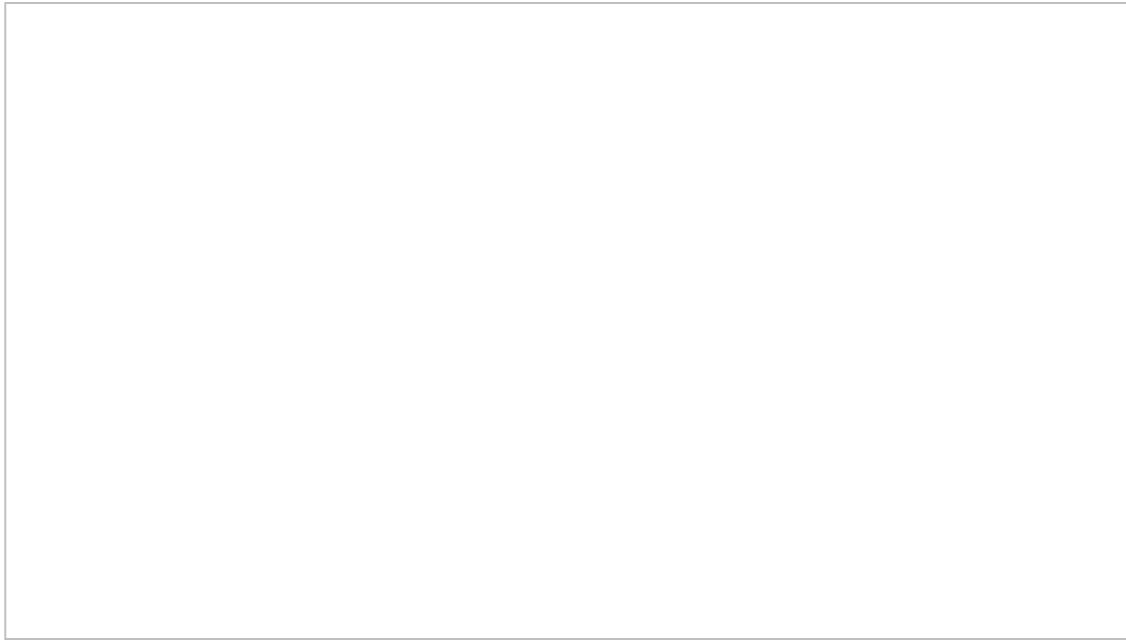
As an illustration, consider the simple map shown in Figure 9-11. The map contains one Sub-model. With the Sub-model closed, main level diagram entities have been organized so as to present a tidy picture.

*Figure 9-11 A Simple Map With One Sub-model*

If the second position has not been established for the structure associated with Receivables and Cash, the diagram will become quite messy when the Sub-model is opened. Such a result is shown in Figure 9-12, below. This result is unsatisfactory - the Sub-model obscures the main-level structure.

*Figure 9-12*

*Without the Second Position, the Diagram Can Easily Become a Mess*



Fortunately, the problem can be easily rectified by setting a second position for main-level diagram entities. In this case, by depressing the control key (Windows) or command key (Macintosh), you can use the Hand to drag-select the structure which lies beneath the Sub-model space. Then, you can control-drag (Windows) or command-drag (Macintosh) to move the structure below the space, as shown in Figure 9-13. Once you establish the second position, building blocks will remember it. Thus, when the Sub-model space is closed the diagram will look like Figure 9-11. When you open the Sub-model, the diagram will look like Figure 9-13.

*Figure 9-13*

*Establishing a Second Position*

**Tips for Operating in the Two-Position Environment:** Two-position provides you with a powerful tool. To use the tool effectively, you must exercise caution. Below we offer a set of tips which will maximize the effectiveness of your efforts in using two-position.

- Straighten out the main level map as you go, rather than waiting until all Sub-models are in place. If you strive to keep your main level map tidy at all times, you'll establish a good first position for each main level building block and object. On the other hand, if you wait until "later," you'll generally need to straighten things out twice.
- Close all Sub-models before you set the first position for main level entities. If you don't you'll be setting the second position! Make it a habit to choose Close Decision Diamonds from the Edit menu, before you set the first position.
- Open all Sub-models before you set the second position for main level entities. Unless all Sub-models are open, the second position you establish may conflict with the position associated with another closed Sub-model. Make it a habit to choose Open Decision Diamonds from the Edit menu before you set the second position.
- Remember that whenever you deposit a building block or object, the location of deposition will become both its first and second position -

regardless of whether a Sub-model is open. If you subsequently move the entity, its new position will be either the first or second position - depending upon whether a Sub-model is closed or open.

- The position of a name relative to its icon will remain fixed, regardless of the position of the icon. Thus, if you move the position of a variable name from the top to the bottom of a converter, while the converter is in its first position, the variable name will remain on the bottom when you open a Sub-model. If you then move the variable name back to the top, it will remain there when you close the Sub-model.
- If you mess things up, it's easy to kill the second position of all main level objects. Choose Clear Second Position from the Model Menu to remove the second position information for all main level objects.

 [Related Topics](#)

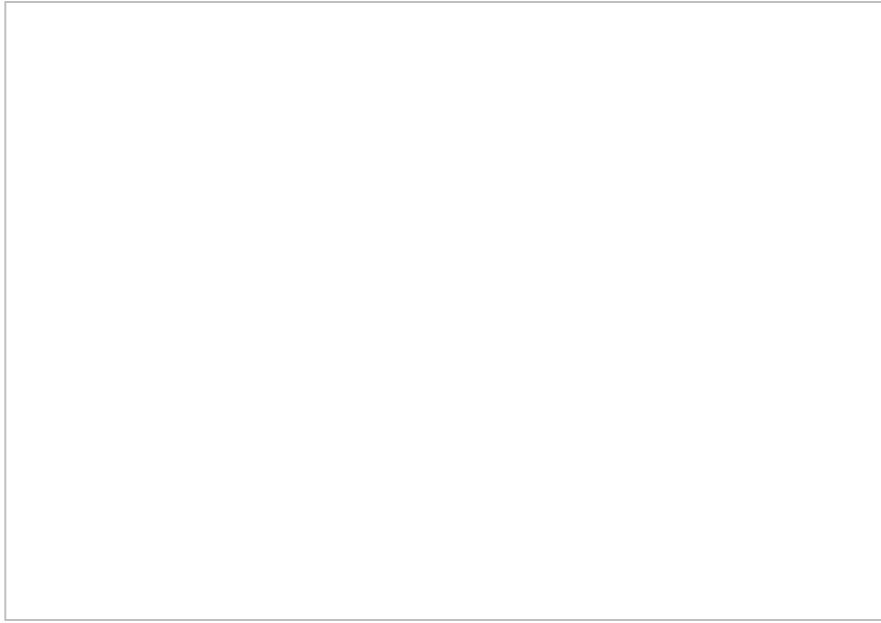
# Assigning Roll-downs and Roll-ups

Once a map which contains at least one stock/flow structure has been created within a Sub-model space, it is necessary to forge equation linkages between a portion of the Sub-model structure and main level counterpart. To forge these linkages, you must first enter the modeling mode. You'll face three tasks:

1. Rolling up a single stock/flow structure in the Sub-model space to the Sub-model icon.
2. Rolling up outflows from the Sub-model structure, to outflows from the Sub-model icon.
3. Rolling down inflows to the Sub-model icon, to inflows to the Sub-model structure.

In this section, you'll learn how to accomplish each task. After working through the basics, you'll learn about two special cases which might otherwise give you pause. The simple structure shown in Figure 9-14 provides the basis for illustrating each task.

*Figure 9-14 Assigning Roll-down and Roll-ups*



**1) Rolling up a Sub-model Stock/Flow Structure:** In this task, you'll be assigning a stock/flow within the Sub-model space to the Sub-model icon. Although you can have as much structure as you'd like within a Sub-model, only one stock/flow structure (in which flows come from and go to clouds) can be rolled up into the Sub-model icon.

To accomplish the roll-up, control-double-click (Windows) or command-double-click (Macintosh) on the Sub-model icon, while the Sub-model is open and

you're in the modeling mode. When you do, you'll enter the icon's Define dialog. Figure 9-15 shows the Define dialog associated with the Sub-model icon Work in Process.

*Figure 9-15  
Sub-model Icon Roll-up Dialog*



Within the dialog, you'll find a list of Allowable Sub-model stock/flow structures. Click on the desired structure from the Allowable list, to load it into the roll-up equation box. Then, click OK to exit the dialog. When you do, the "?" in the Sub-model icon will be extinguished.

*Notes:*

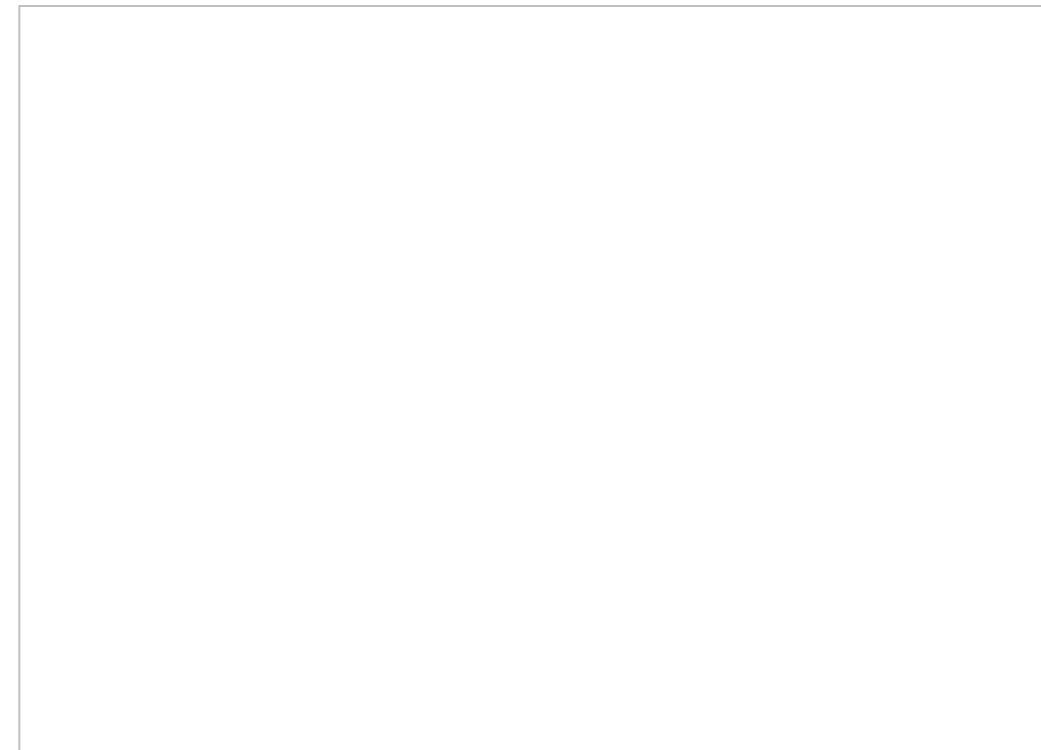
- To be "allowable," a stock/flow structure must have at least one inflow coming from a cloud, and at least one outflow going to a cloud.
- The software allows only one Sub-model stock/flow structure to be rolled up to the Sub-model icon.
- Each Sub-model stock/flow structure is given its own line in the Allowable list.
- When the model runs, the Sub-model icon will report out the sum of its associated stock/flow structure stock values.

**2) Rolling Up Outflows from the Stock/Flow Structure:** In the second task, you'll be forging linkages between the Sub-model stock/flow structure outflow(s), and outflow(s) from the Sub-model icon. In forging these connections, you must make sure to roll up all outflows from the Sub-model stock/flow structure (i.e., all outflows which go to clouds). All outflows from the

Sub-model icon must be defined as roll-ups.

To roll up the outflows, control-double-click (Windows) or command-double-click (Macintosh) on an outflow from the Sub-model icon, while the Sub-model is open and you're in the modeling mode. You'll enter the outflow's Define dialog. Figure 9-16 shows the Define dialog associated with the exit process flow.

*Figure 9-16  
Sub-model Icon Outflow Roll-up Dialog*



Within the dialog, you can select the desired flow from the Allowable Inputs list to load it into the roll-up equation box. When you click OK to exit the dialog, the "?" will be extinguished in the Sub-model icon's outflow. The Sub-model stock/flow structure's outflow will be re-named to reflect the name of the Sub-model icon's outflow, as shown in Figure 9-17.

*Notes:*

- Multiple outflows from a Sub-model stock/flow structure may be rolled up to a single outflow on the main level. Clicking multiple Sub-model flows in the Allowable list, will cause a "+" sign to be put between them in the equation box.
- Once a flow has been rolled up, it will become gray in the Define dialog of each outflow from the Sub-model icon. This prevents you from rolling up a single Sub-model flow to many main level outflows.
- You can never have more outflows from a Sub-model icon, than from the assigned Sub-model stock/flow structure.

- Rolling up outflows will automatically lead to the rolling up of the associated Sub-model stock/flow structure. Thus, if you start with this second task, the first task will be accomplished automatically for you.

*Figure 9-17  
Roll-ups Assigned*



**3) Rolling Down Inflows to the Sub-model icon:** In the third task, which must be done after you have done the roll-up of the Sub-model stock/flow structure, you'll be forging a linkage between each inflow to the Sub-model icon, and the associated inflow to the Sub-model stock/flow structure (i.e., the inflows which come from clouds). The software enforces a one to one correspondence between inflows on the main level and roll-down inflows. This means that you must make a unique roll-down assignment for each inflow to the Sub-model stock/flow structure.

To roll down an inflow to the Sub-model icon, open the inflow to the Sub-model stock/flow structure. The inflow's define dialog will appear, with an allowable list of main level flows to roll down. For our example, Figure 9-18 shows the Define dialog for the flow Noname 1.

In the dialog, click on one flow from the Allowable list. When you click OK and exit the dialog, you'll find the diagram has changed in two ways. First, the inflow to the Sub-model stock/flow structure will be renamed to reflect the name of its associated main level inflow. Second, the display of its icon will change to reflect that it is a roll-down flow. These two changes in the diagram are shown in Figure 9-19.

*Figure 9-18  
Roll-down Dialog*

*Figure 9-19  
Roll-down assigned*

---

**Note:** Each inflow to the Sub-model icon can be rolled down to one (and only one) inflow to the Sub-model stock/flow structure.

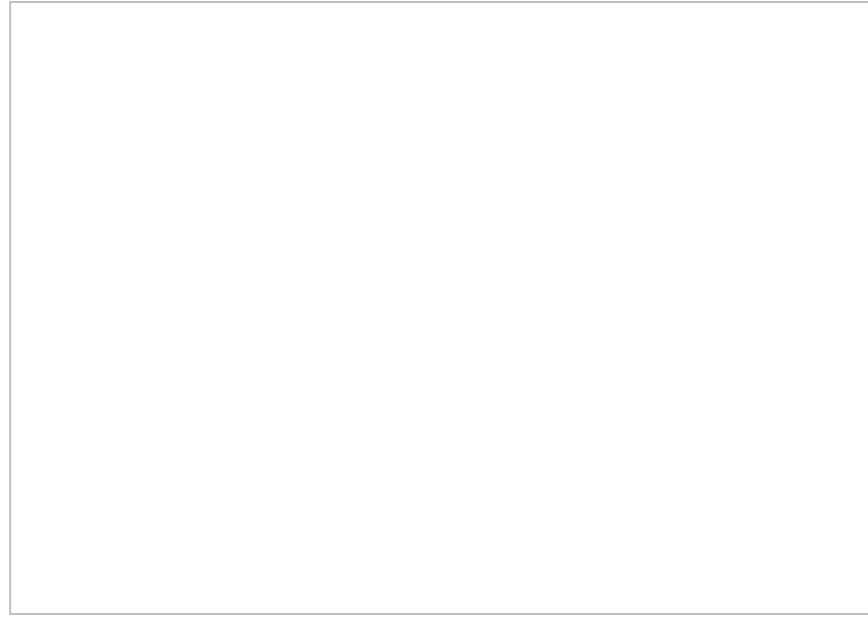
---

# Special Cases Requiring Special Care

Two special situations, each requiring special care, can arise as you engage in the process of linking a Sub-model stock/flow structure to its associated main level structure. Each arises when there is a mismatch between flows on the main level, and the flows in the Sub-model. First, we'll look at the case in which an outflow from the Sub-model stock/flow structure has not been rolled up into an outflow from the Sub-model icon. Second, we'll look at the case in which there is not a one to one correspondence between the number of inflows to the Sub-model icon, and the number of flows to the Sub-model stock/flow structure.

**Case 1: The un-rolled up outflow:** Consider the Sub-model configuration shown in Figure 9-20.

Figure 9-20 Sub-model Stock/Flow Structure with Undefined/Unassigned Outflow

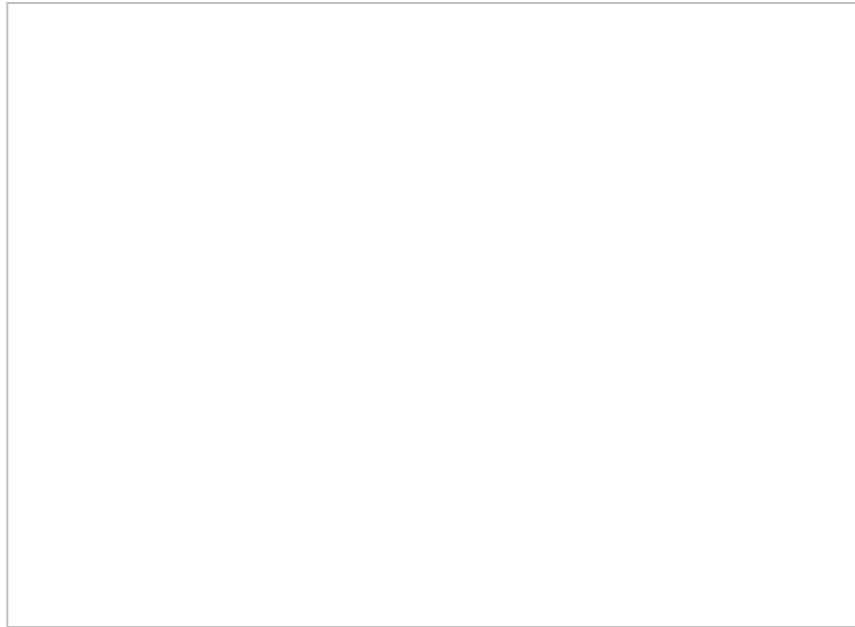


In this configuration, the stock/flow structure has been rolled up to the Sub-model icon. The flow *exit process*' has been rolled up to the main level outflow *exit process*. Finally, the main level inflow *enter process* has been rolled down to the Sub-model inflow *enter process*'. Note, however, the "?" in the outflow from the stock/flow structure, Noname 1. This outflow has not been rolled up to the main level. Nor has it been defined in its own right.

A natural response to the "?" would be to open Noname 1 and define its logic. Doing so leads to a somewhat puzzling result, as shown in Figure 9-21. Although the "?" is (correctly) extinguished for Noname 1, a new "?" has appeared in the Sub-model icon. At this point, you might ask, "What's going on here?"

*Figure 9-21*

*A ? in the Sub-model Icon - Unassigned Flow*



What's going on is this: The Sub-model stock/flow structure has not been completely assigned to the main level structure. Specifically, not all outflows from the stock/flow structure have been rolled up into outflows from the Sub-model icon.

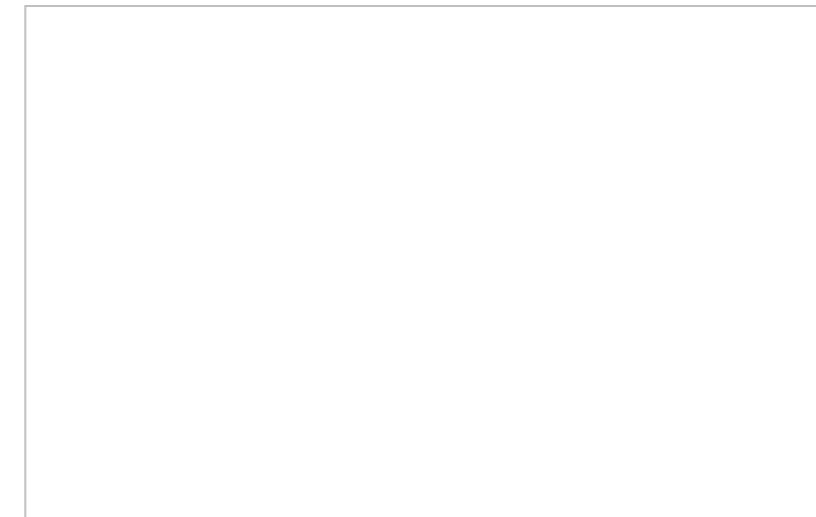
Control-double-clicking (Windows) or command-double-clicking (Macintosh) on the Sub-model icon, after attempting to run the model, would bring forth an alert which summarizes the problem. To remedy it, you have two options. First, you could control-double-click (Windows) or command-double-click (Macintosh) on the *exit* process flow. Within the dialog, you could add Noname 1 to the roll-up definition. Second, as illustrated in Figure 9-22, you could create a second outflow from the Sub-model icon, and use it to roll up the Noname 1 flow. Exercising either option would put the model into a runnable state.

*Figure 9-22*

*Adding a Second Outflow to the Main Level*

**Case 2: The mismatched inflows:** A second case requiring special care can occur when there is a mismatch between the number of inflows to a Sub-model icon, and the number of flows into the corresponding stock/flow structure. As an illustration of this phenomenon, consider Figure 9-23.

*Figure 9-23  
A Simple Sub-model Configuration*

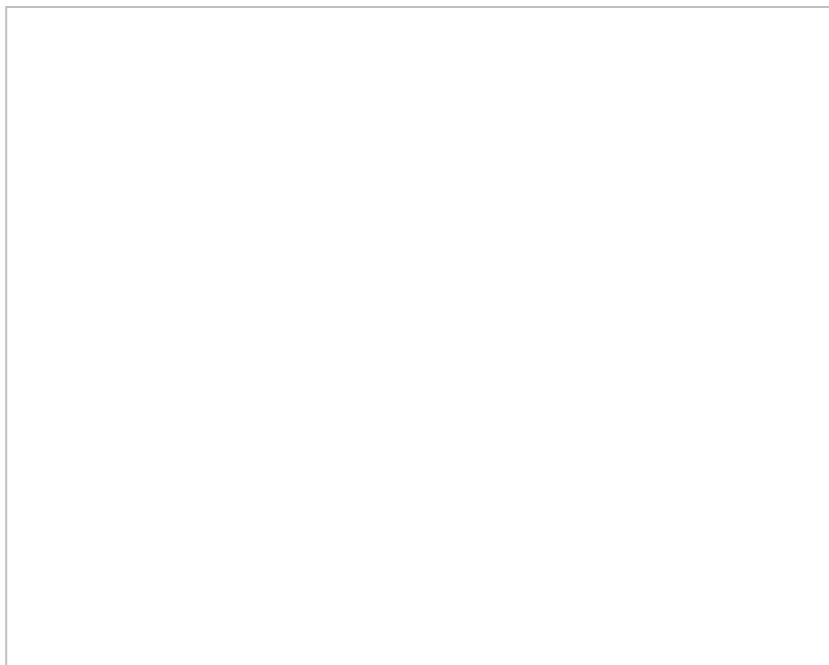


This configuration has been completely defined, with all roll-ups and roll-downs assigned. Watch what happens, however, when a second inflow is drawn into the Sub-model icon on the main level. Figure 9-24 shows the result.

*Figure 9-24  
Adding a Second Inflow to the Main Level*

Again, the question mark which appears in the Sub-model icon is an indication that your work is not finished. (You'll receive an alert to this effect if you open the Sub-model icon, or if you try to run the model.) The software requires that each inflow to the Sub-model icon is rolled down to a unique inflow in the stock/flow structure. This means that you must draw another inflow into the Sub-model stock/flow structure, if you wish to retain the second inflow to the Sub-model icon. The inflow to the stock/flow structure may be drawn into Stage 1. Or, as shown in Figure 9-25, you can draw the required inflow into Stage 2.

*Figure 9-25  
Rolling Down a Second Inflow to the Sub-model Stock/Flow Structure*



 [Related Topics](#)

# Introduction to Cycle-Time

The cycle-time functionality enables you to generate and collect a variety of useful metrics regarding the amount of time that material spends "in process," as it moves through a conserved flow chain. Such metrics can be valuable in a variety of modeling contexts. In particular, cycle-time metrics can be of value when you are engaged in a process re-engineering or process improvement effort. The clear, unambiguous cycle-time metrics provided by the software can help you, both in diagnosing process problems, and in testing and evaluating proposed alternatives.

The topics in this section document the cycle-time functionality. It is divided into three parts:

- [An Overview of Cycle-Time Concepts](#) provides an overview of the cycle-time concepts incorporated into the software.
- [Incorporating Cycle-Time into Your Model](#) walks you through a simple process for putting cycle-time into a model. In so doing, the second section provides an overview of the Built-in cycle-time functions.
- [Cycle-Time Rules](#) details the "rules" of cycle-time for a main chain with multiple flows and for a Sub-model, and briefly discusses the changes in internal calculations which are brought about when cycle-time is brought into play.

Each cycle-time Built-in function is documented extensively in [Cycle-time Functions](#) in the Builtins section..

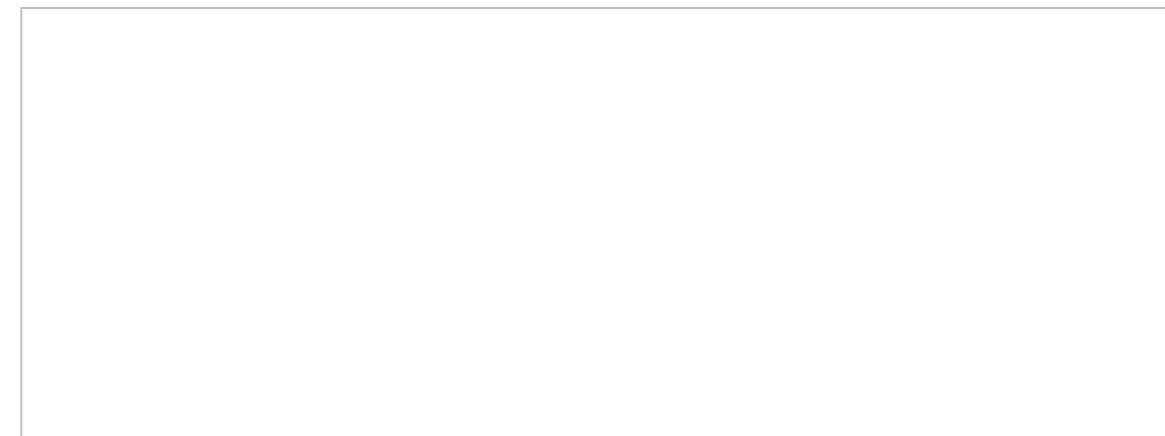
 [Related Topics](#)

# An Overview of Cycle-Time Concepts

Imagine that you and three of your colleagues have been asked to run an obstacle course. You start at 1:00 PM, and complete the course at 1:35 PM. The first of your colleagues begins at 1:05 PM, and completes the course at 1:42 PM. The second colleague begins at 1:10 PM, and, being somewhat more athletic than you, completes the course at 1:35 PM. The third starts at 1:15 PM, gets stuck somewhere, and exits the course at 2:15 PM. Because you know the start and completion time for each person in the group, you can calculate a wealth of useful information about the time that individuals (or groups of individuals) have spent "in process."

This little obstacle course example can be represented nicely as a stock/flow map. Figure 10-1, for example, maps a four-stage course. As the Figure suggests, incorporating cycle-time metrics into a model is quite easy.

*Figure 10-1 Representing an Obstacle Course*



You'll note two distinctive features in the map of Figure 10-1. First, the flow starting the course looks a bit like the face of a clock. The clock face on the flow indicates that the flow has been time-stamped. Whenever material moves through the flow, it will be stamped with the current simulation time. Time-stamping establishes a starting time, for the purposes of generating cycle-time metrics.

Second, the downstream flow completing the course has been connected to the converter called course completion time. This converter also displays the face of a clock - but the face is partially filled. This partially-filled clock face indicates that the converter contains a cycle-time Built-in function. When material moves through the downstream flow, the Built-in will use the material's starting time information, in conjunction with the current simulation time, to calculate a cycle-time metric.

 [Related Topics](#)

# Incorporating Cycle-Time into Your Model

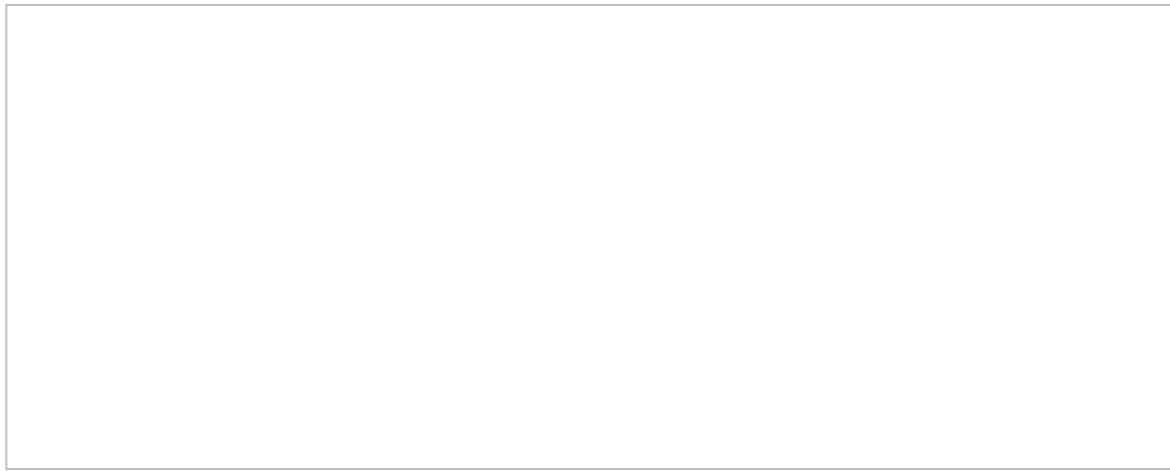
This section presents a process for incorporating cycle-time into your models. At its essence, the process is quite simple. The process involves four basic steps, as outlined below:

1. Determine the "starting line" in a conserved flow chain.
2. Time-stamp the flow(s) which represent the starting line.
3. Determine the "finish line" in the conserved flow chain.
4. Tap into the flow associated with the finish line using a connector and converter. Define the converter using one of the cycle-time Built-in functions.

Let's take a closer look at each step in the process.

**1. Determine the starting line:** In this step, you're deciding what flow(s) in the process should be considered as the starting line, for the purpose of generating cycle-time metrics. Any flows in a conserved flow chain can be used as the starting line. In Figure 10-2, for example, one could choose from among many potential starting lines. Any flows downstream from the starting line you choose can be used for the generation of cycle-time metrics.

*Figure 10-2 Choosing a "Starting Line"*



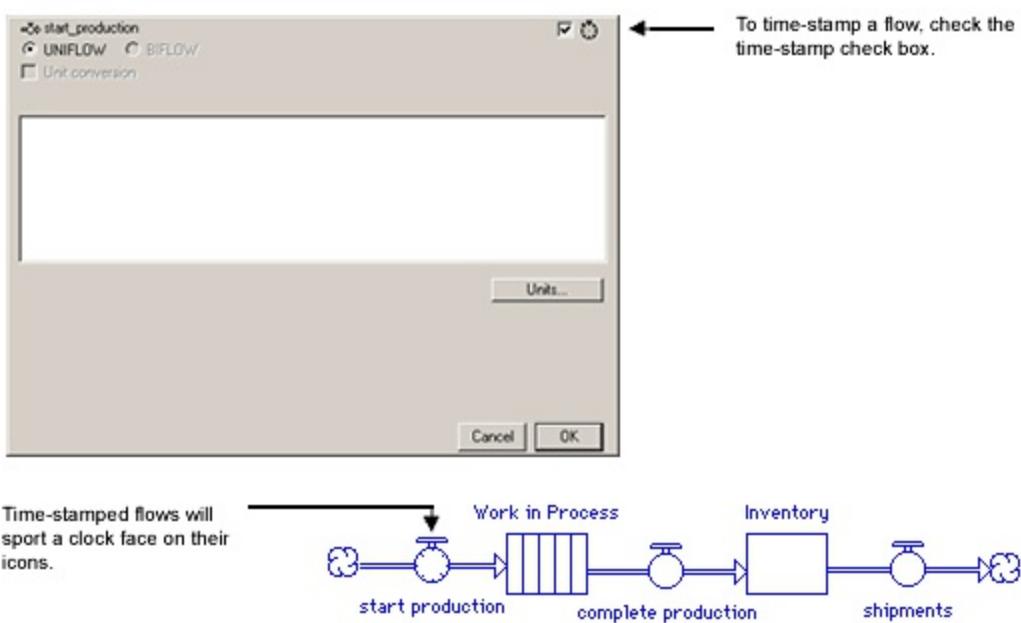
Thus, if you identified the flows enter queue 1 and enter queue 2 as the starting lines, you then could collect cycle-time data such as the wait time for process 1, the total time for process 1, and the cycle-time for the overall process. If, on the other hand, you decided to use enter process 3 as your starting line, the associated cycle-time calculation would apply to Process 3 only.

Notes:

- When you have multiple entry points to a main chain, you will want to consider making each entry point a "starting line." In Figure 10-2, if enter queue 1 is designated as the starting line but enter queue 2 is not, only the material which moves through the main chain via enter queue 1 will be considered in the subsequent cycle-time calculation.
- It's possible to designate successive stages in a main chain as starting lines. In Figure 10-2, for example, you might want to designate both enter process 1 and enter process 3 as starting lines. However, if both flows are time-stamped, material that moves through enter process 1 will be time-stamped again, when it moves through enter process 3. This material will lose its earlier time-stamping.

**2. Time-stamp the flows which represent the starting line:** To time-stamp a flow, begin by entering the flow's define dialog. As indicated in Figure 10-3, you'll find a check box next to a small stopwatch icon at the top-right of the dialog. This check box is the time-stamping check box. Check the box. Then, click OK. When you return to the diagram, the flow's appearance will change to indicate that it has been time-stamped. An illustrative time-stamped flow is also shown in Figure 10-3.

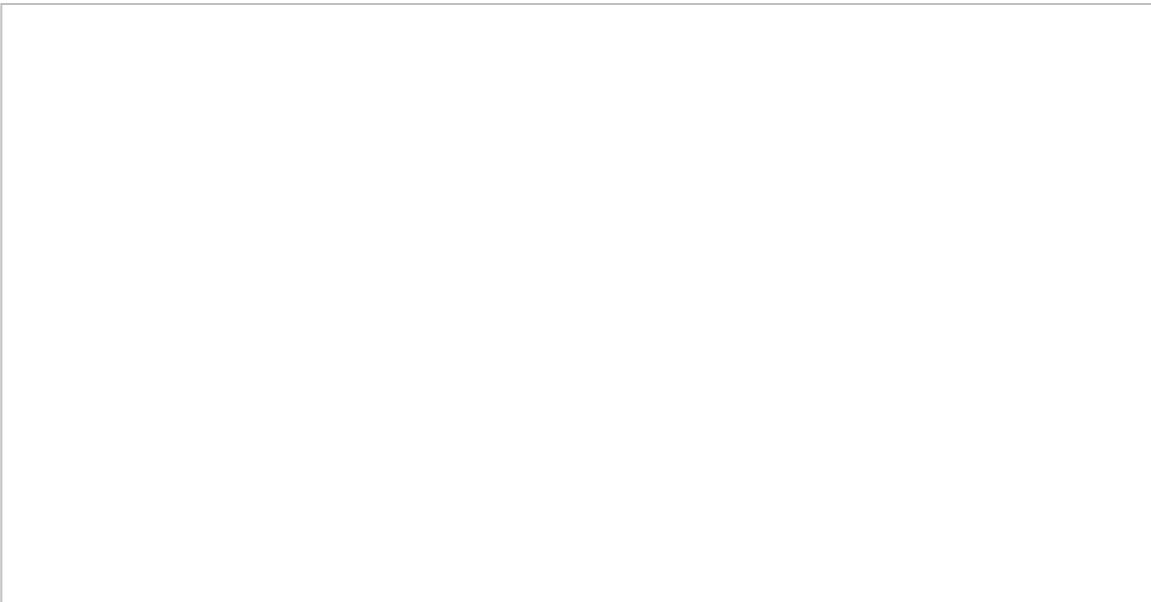
Figure 10-3  
Time-Stamping a Flow



**3. Determine the "finish line":** The finish line is the point in the conserved-flow chain where you want to collect the cycle-time information. As shown in Figure 10-4, the finish line can be any flow which is downstream of the starting line in the chain. There must be a time-stamped flow upstream in the

conserved flow chain, if cycle-time calculations are to have meaning.

*Figure 10-4  
Choosing a "Finish Line"*



**4. Tap into the flow associated with the finish line:** To wrap up your work, you'll use a connector and a converter to tap into the finish line flow. In the modeling mode, you can define the converter using one of the cycle-time Built-in functions. As illustrated in Figure 10-5, this step is straightforward. When you exit the converter's define dialog, the converter will sport a partially-filled clock face. This is your visual indication that the converter is generating cycle-time data. When you subsequently run your model, as the time-stamped flow volume crosses the finish line its time-stamp will be used to generate the desired cycle-time data.

*Figure 10-5  
Tapping into a Downstream Flow*

## Notes:

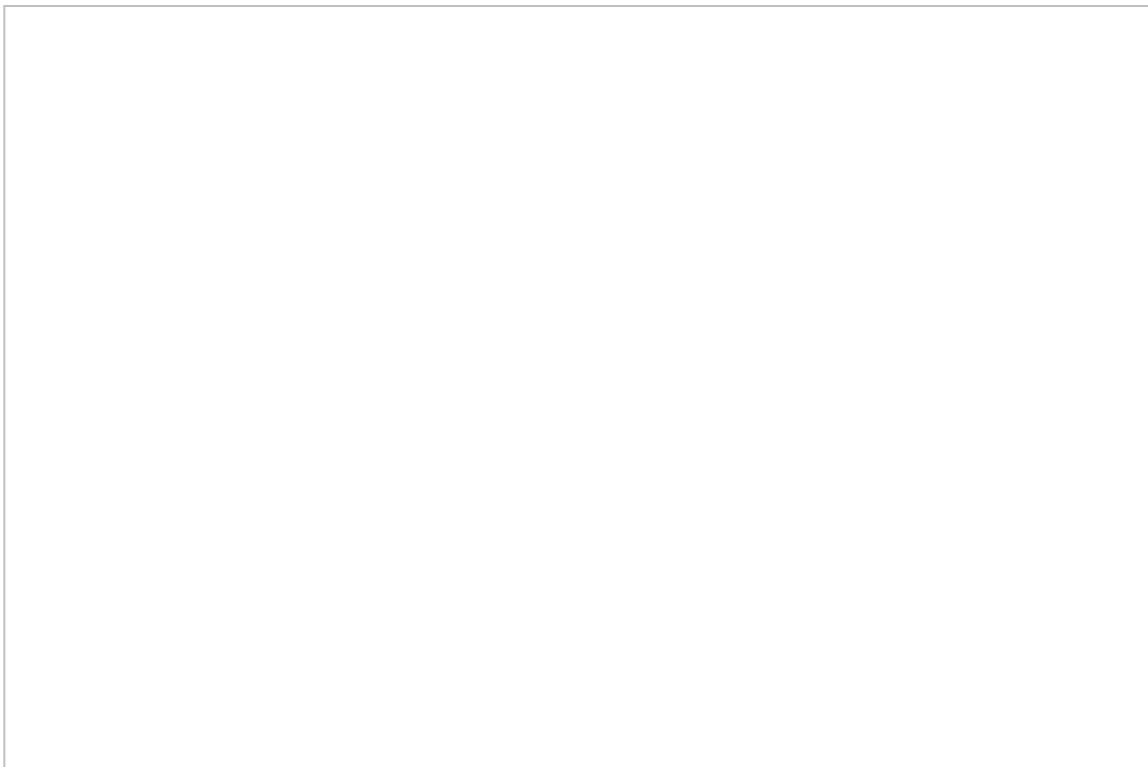
- It's possible to tap into a conserved flow chain in multiple places, so as to collect cycle-time information from multiple-stages in a conserved-flow chain. Each metric will be generated based upon the closest upstream time-stamp.
- By using multiple converters tapped into the same downstream flow, you can calculate a number of different cycle-time metrics at the same point in a process.
- Cycle-time Builtins are available only in converters. They will be gray unless the converter is tapped into a flow in a conserved flow chain, and you have time-stamped a flow somewhere upstream in the same conserved flow chain.

 [Related Topics](#)

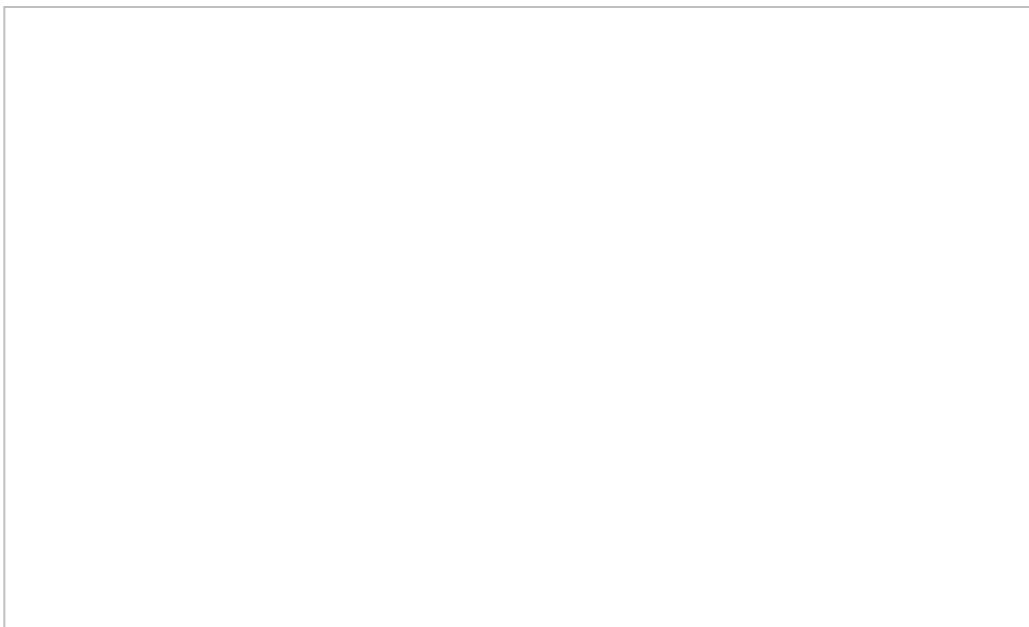
# Cycle-Time Rules

This section uses graphics to present the rules of cycle-time, in an easily accessible form. Figure 10-7 shows how the software handles multiple time-stamps within a conserved-flow chain. Figure 10-8 shows how the software deals with cycle-time calculations when a Sub-model is involved.

*Figure 10-7 Rules of Multiple Time-Stamping*



*Figure 10-8  
Rules of Time-Stamping with Sub-models*





# Internal Considerations

Whenever a flow is time-stamped in a model, the software will automatically shift from the "Normal" to the "Cycle-time" Run mode. The change in the Run mode is required for the software to collect cycle-time information. The Run mode radio buttons can be found within the Time Specs dialog, under the Run menu.

Internally, the following changes occur as a result of shifting the Run mode from Normal to Cycle-time.

- The model will be set to run using Euler's integration method. The Runge-Kutta 2 and Runge-Kutta 4 methods will be disabled.
- Inflows to reservoirs will be "lined up" within the reservoirs, and will exit via outflows on a first in, first out basis. In this respect, reservoirs will act like queues.
- Non-negativity will be turned "on" for all reservoirs, and will not be accessible within the reservoir dialog.

Practically speaking, you should notice only two behavioral changes when the Run mode shifts to Cycle-time. First, it is possible that the execution of a simulation run will be somewhat slower than might otherwise be the case. This is because of the additional calculational overload that is brought about by the generation of cycle-time metrics. Second, any reservoirs that have taken on negative values in the Normal mode (because Non-negativity has been turned "off" in their dialogs), will be kept from taking on negative values.

It's always possible for you to override the shift from Normal to Cycle-time Run mode. To do so, simply enter the Time Specs dialog and click on the "Normal" radio button. If you do shift to the Normal Run mode, cycle-time calculations will be disabled. As a result, cycle-time Builtins will return a value of zero. All time-stamped flows will lose the clock face on their icons. Their check boxes will be turned "off." Similarly, all cycle-time converters will lose the clock face on their icons.

You can shift back to the Cycle-time mode by clicking on the Cycle-time radio button within the Time Specs dialog. Or, you can time-stamp any flow in your model. When you shift back to the Cycle-time mode, the software will remember all of your previous work with cycle-time. The clock face will return on the icons for all flows which had previously been time-stamped, as well as for all converters which contain cycle-time Builtins.

 [Related Topics](#)

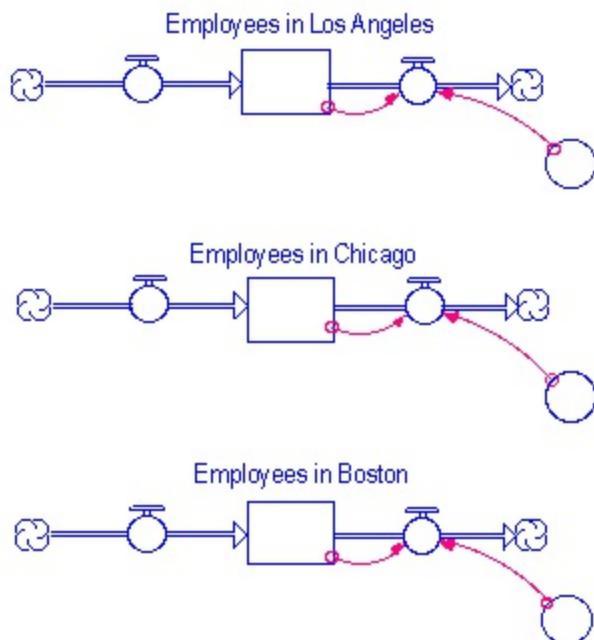
# Introduction to Arrays

In many modeling efforts, you will find yourself using copy and paste to replicate a model structure multiple times. For example, a demographic model might treat the aging of a population, but be replicated for multiple ethnic groups in multiple areas. Conceptually, the model is very straightforward. Visually, the structure might take up a lot of diagram space and therefore be a bit difficult to handle. Similarly, you might model a supply chain in which a supplier ships out of a single inventory to multiple customers. Conceptually, this structure is very straightforward. Diagrammatically, it can be a bit difficult to swallow. In a marketing model, you might repeatedly use a simple structure to keep track of customer affinity for several different product lines. In an ecological model, you might repeatedly use a simple structure to represent the different serial stages associated with plant succession.

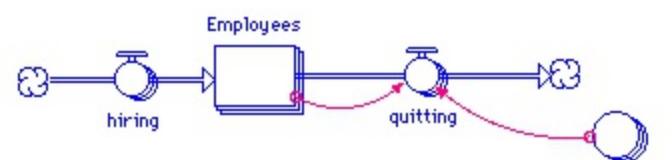
In each of these cases, copy and paste is the most straightforward way to represent the multiple parallel model structures involved. Unfortunately, the associated visual complexity of the resulting model diagram can become hard to manage, both for the builder of the model and the user of the model. Arrays provide a simple yet powerful mechanism for managing this visual complexity. By "encapsulating" parallel model structures, arrays can help you to present the essence of a situation in a simple diagram. Beneath the scenes, of course, arrays retain the richness of the disaggregated structure. Figure 11-1 provides a picture of what Arrays buy you.

Figure 11-1 Arrays Encapsulate Parallel Model Structures

This...



becomes this...



- Arrayed variables simplify the model display (3-D effects denote arrayed variables).

- The generic nature of the underlying equations can simplify construction and modification.

Introducing arrays into a model is conceptually straightforward. The process consists of three simple steps.

1. *Use the Array Editor to define one or more dimensions* - A dimension is simply a category. Each dimension contains a set of elements. For example, you might create a dimension entitled "Cities." This category might contain the elements "London," "Sarajevo," and "Tokyo." A category of "Species" might include "lion," "tiger," and "bear." You will find the Array Editor in the Interface menu, the Model menu, or the Equations menu. In addition, you can get at the Array Editor from within the define dialog of any arrayed variable.
2. *Transform model variables into arrayed variables* - After you have defined one or more Dimensions, you can transform non-arrayed variables into arrayed variables. To make the transformation, simply check the Array check box within the define dialog of any stock or converter. You needn't worry about transforming flows - whenever you transform a stock into an arrayed stock, any attached flows are automatically transformed into arrayed entities as well.
3. *Define the equation logic for arrayed variables* - Within each arrayed variable, you need to define the equation logic for each element within the array. It's possible to apply a single, generic equation to all elements within the array. Alternately, you can "page through" the individual elements in an arrayed variable and provide unique numerical values or equation definitions for each element.

This process is illustrated for one- and two-dimensional arrays in [Working with One-Dimensional Arrays](#) and [Working With Two-Dimensional Arrays](#). The rest of the topics in this section address many of the issues and special situations that can arise as you work with arrays in your models.



## [Related Topics](#)

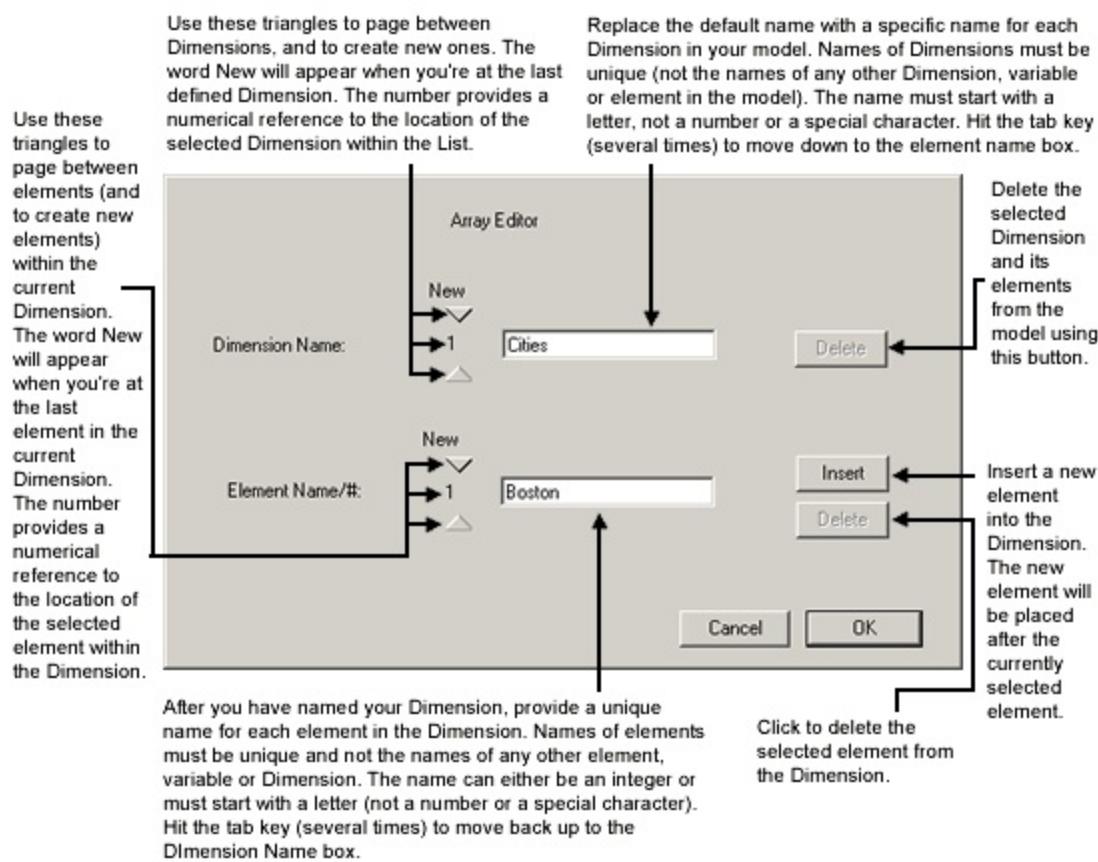
# Working with One-Dimensional Arrays

One-dimensional arrays are far simpler to understand than two-dimensional arrays. The simple illustration here shows how to incorporate a one-dimensional arrayed structure into your model. The illustration looks at company with offices in Boston, Chicago, and Los Angeles.

## 1. Use the Array Editor to define one or more dimensions

Enter the Array Editor through either the Map, Model, or Equation menu. When you do, you'll see a dialog that looks like the one in Figure 11-3. Within the dialog you can create new dimensions. For each dimension, you can define the associated elements. As the annotations within Figure 11-3 suggest, there is great power within the Array Editor dialog.

Figure 11-3 Array Editor Dialog

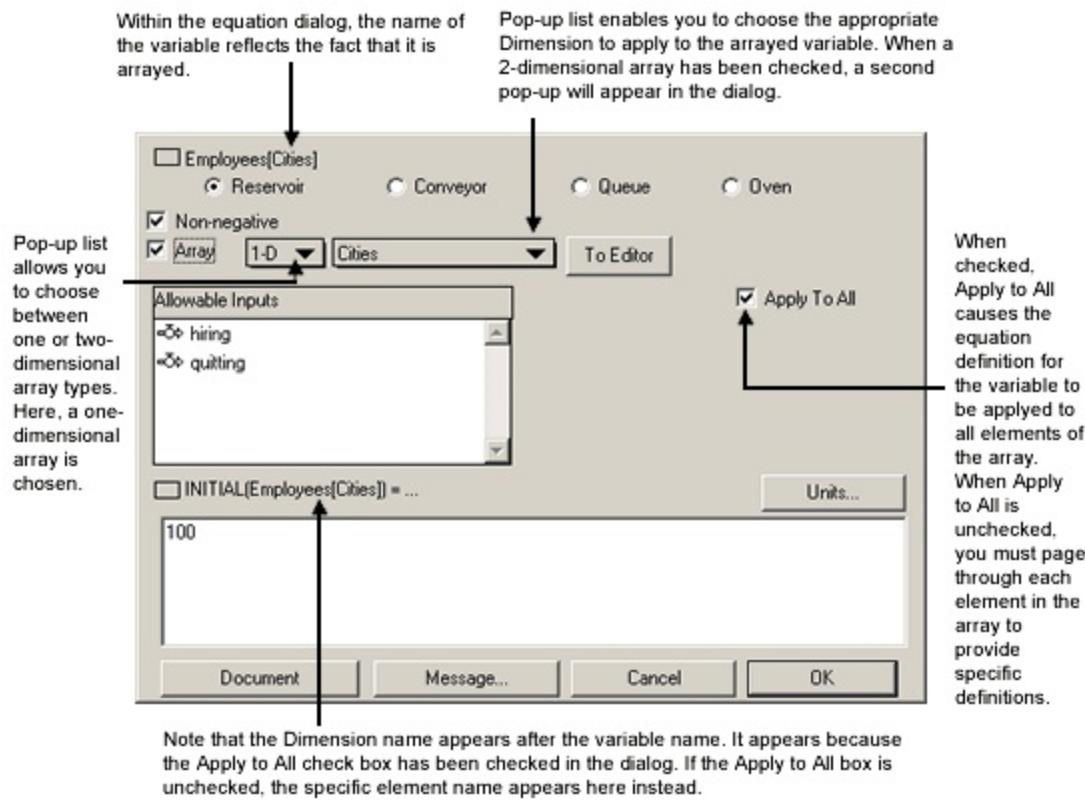


## 2. Transform model variables into arrayed variables

Once you have created one or more Dimensions, you can begin to transform model variables into arrayed variables. Note that whenever you transform a stock into an arrayed stock, its attached flows are *automatically* arrayed for you. In modeling mode, open a stock or converter dialog and check the Array check box. Then, use the pop-up list of dimensions to select the Dimension

that you wish to apply to the variable. Figure 11-4 illustrates the process of transforming a reservoir into an arrayed reservoir. The process of transforming converters into arrayed converters is analogous.

*Figure 11-4  
Transforming a reservoir into an arrayed reservoir*



### 3. Define the equation logic for arrayed variables

To define the equation logic for an arrayed variable, you must engage in two distinct activities. First, you must determine whether you decide whether to apply a single generic equation to all elements within the arrayed variable. Then, you need to click in the equation logic - either a single generic equation or a specific equation for each element of the array. As you'll see, the application of a single generic equation can minimize both the analytic complexity and the creation time associated with defining the equation logic for arrayed variables.

*To Apply to All, or Not to Apply to All...That is the question* When *Apply To All* is checked within an equation dialog, you are telling the software to use the same generic equation for all elements within the arrayed variable. For example, imagine that you have an arrayed stock of entitled Animal Population. If *Apply To All* is checked in the reservoir's dialog, the single number that you enter for the initial value will be used to define the initial value for all the elements in the array. If you type 100 as the initial value, each population within the arrayed variable will have an initial value of 100.

Similarly, if the arrayed variable is a flow or converter production, with Required Inputs of Workers and productivity, you could click in the equation Workers \* productivity. This equation would be used to define all elements in the arrayed variable. Figure 11-4 (above) illustrated a stock with Apply To All selected.

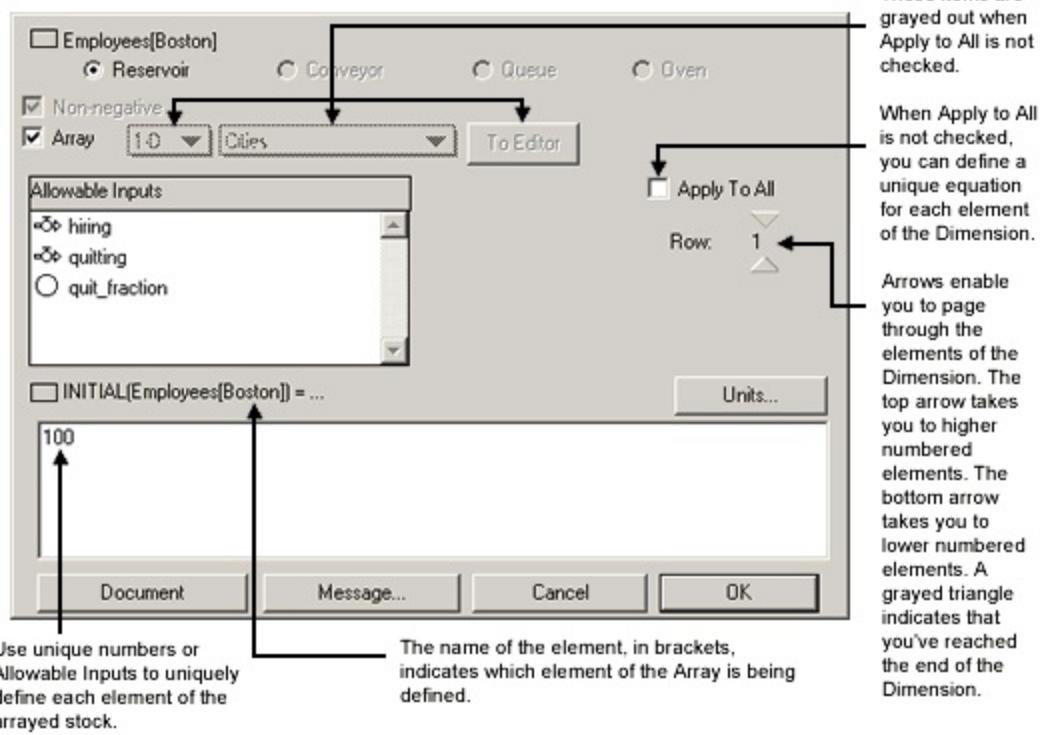
When Apply to All is checked, you have access to all stock, flow, and converter dialog settings (for example, stock type, non-negative, flow time-stamping, unit conversion, biflow vs. uniflow, 1-D vs 2-D, Dimension Names) should be selected while Apply To All is checked. If Apply To All is unchecked, these options will become inaccessible.

If you want to create an arrayed graphical function, Apply to All *must* be checked within the main equation define dialog. Once you have defined the input to the graphical function, click the Become Graph button at the bottom of the dialog. You can then define a single graphical relationship that will be applied to all elements within the Array. (Within the graphical function dialog itself, you can uncheck Apply to All and sketch individual curves for each model element)

When you uncheck the Apply To All box in an arrayed variable's dialog, you must provide a separate definition for each element within the arrayed variable. Use the paging arrows shown in Figure 11-5 to walk through each element in the arrayed variable.

*Figure 11-5*

*Defining Individual Elements Within an Arrayed Variable - Apply to All Not Checked*

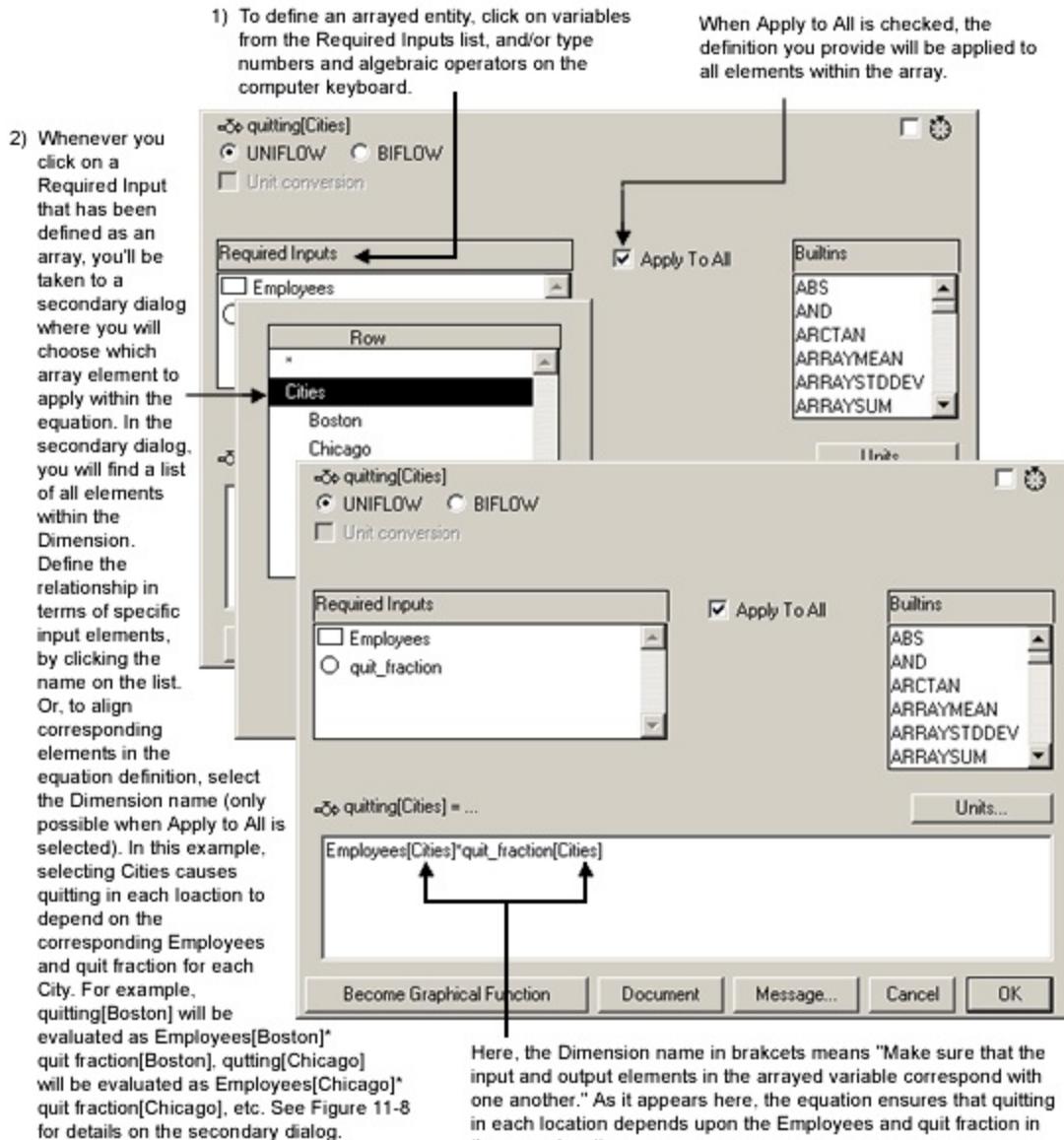


*Clicking in the equation logic* Once you have determined whether to use Apply to All option, you are ready to specify the equation logic for the arrayed variable. If you are using constants or have non-arrayed variables on the Required Inputs list or Allowable Inputs list, define each element or use the Inputs just as you would non-arrayed variables. With constants or scalar inputs, defining the equation logic is easy.

When the inputs to your arrayed variable are themselves arrayed variables, clicking in the equation logic is almost as easy. After you click on the input, a secondary dialog will emerge. You'll use this dialog to tell the software which element of the arrayed input to use to define the elements within the arrayed variable. Figure 11-6 shows how to click in the equation logic for an arrayed variable, when Apply To All is checked. Note that in this instance, one generic equation is used to define multiple elements in an appropriate fashion.

*Figure 11-6*

*Defining Logic Using One-Dimensional Arrayed Input Variables - Apply to All Checked*

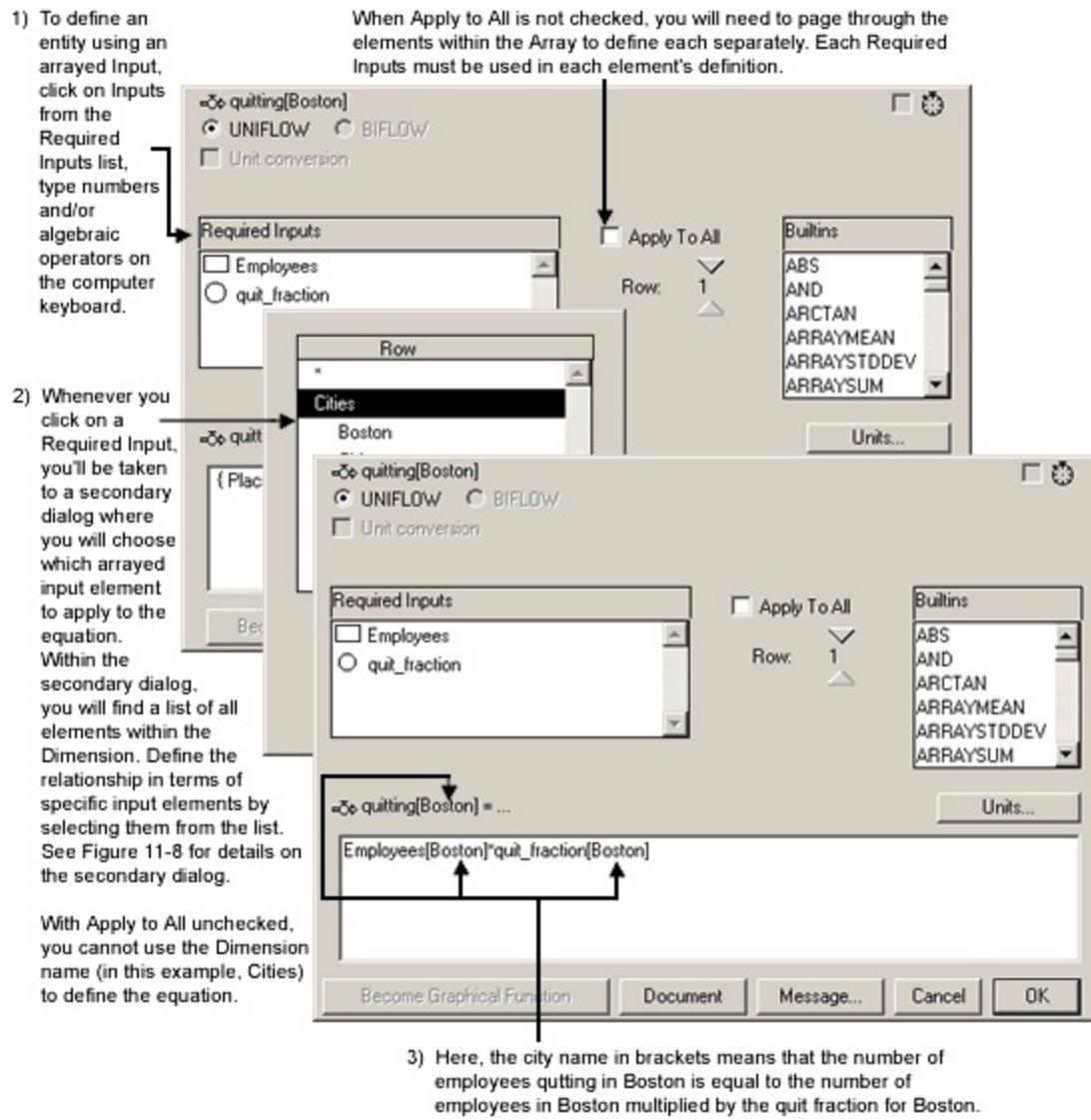


Here, the Dimension name in brackets means "Make sure that the input and output elements in the arrayed variable correspond with one another." As it appears here, the equation ensures that quitting in each location depends upon the Employees and quit fraction in the same location.

When you uncheck the Apply to All box, you'll need to provide a specific equation for each element within the arrayed variable. Use the paging arrows to navigate through each element in the arrayed variable to enter the unique definition for each. Remember that *all* Required Inputs must be used to define each element in the arrayed variable. Operations for this process are outlined in Figure 11-7

*Figure 11-7*

*Defining Logic Using One-Dimensional Arrayed Input Variables - Apply to All Not Checked*

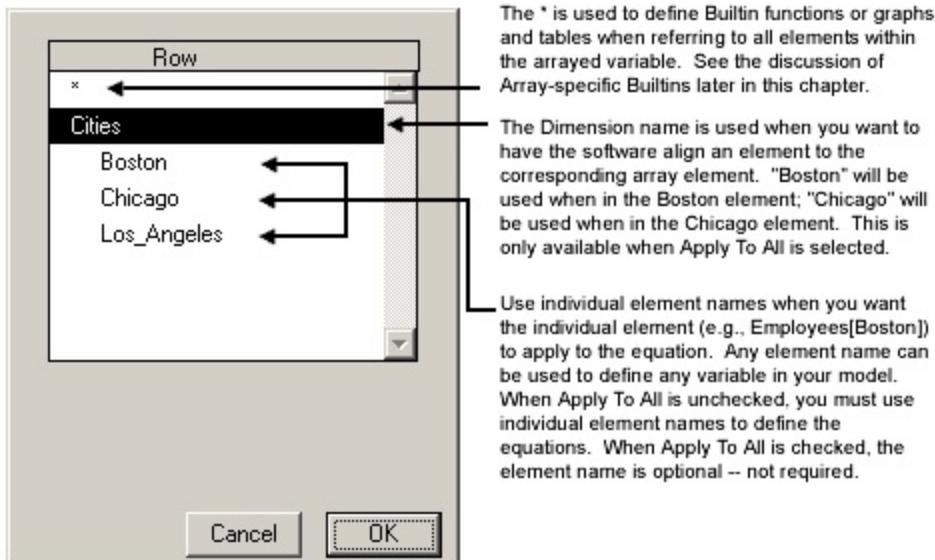


3) Here, the city name in brackets means that the number of employees quitting in Boston is equal to the number of employees in Boston multiplied by the quit fraction for Boston.

Figure 11-8 summarizes the options with an arrayed input variable's secondary dialog.

*Figure 11-8  
Secondary Dialog - One-Dimensional Arrayed Variable*

This secondary dialog provides options for defining equations when arrayed Required or Allowable Inputs are used. It will appear when you click on a one-dimensionally arrayed variable name in the Required or Allowable Input list.



## [Related Topics](#)

# Working With Two-Dimensional Arrays

Working with two-dimensional arrays involves the same basic process as working with one-dimensional Arrays. The only wrinkle is that instead of using one Dimension within the arrayed variable, you will be working with two. For example, suppose that in your model you need to represent Employees both by where they are located (e.g. Boston, Chicago, and Los Angeles) and by the function that they perform within the organization (e.g. Administration, Human Resource, or Finance). The matrix in Figure 11-9 gives a picture of the type of situation that can be captured by a two-dimensional arrayed variable.

Figure 11-9 Two-Dimensional Matrix - Employees

		Dimension 1: Location		
		Boston	Chicago	Los Angeles
Dimension 2: Function	Administration			
	Human Resources			
	Finance			
	In this example, we can track both location and function for our employees.			

## 1. Use the Array Editor to define two or more dimensions

When you are ready to set up a two-dimensional Array, the first step is to revisit the Array Editor dialog to define Dimension Names (See [Figure 11-3](#) for details on setting up a Dimension). To set up a two-dimensional Array, you need at least two Dimensions within your model.

## 2. Transform model variables into arrayed variables

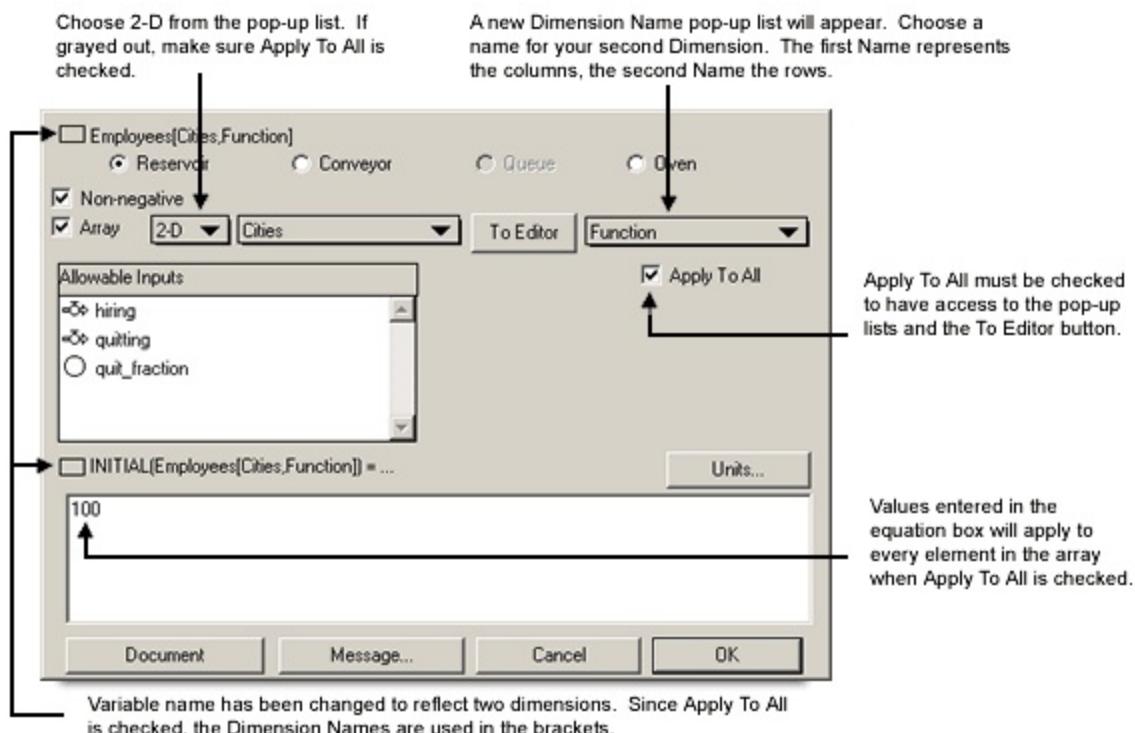
The arrayed variables need to be defined with two Dimensions. To create a 2-D arrayed variable, open the variable's equation dialog and check the Array check box. Then, pick 2-D from the adjacent pop-up list. Finally, use the two dimension name lists to pick the appropriate dimensions for the arrayed variable. Figure 11-10 illustrates how to set up a two-dimensional arrayed stock.

## 3. Define the equation logic for arrayed variables

As with one-dimensional Arrays, when Apply To All is checked, the constant or algebraic definition you supply *once* will apply to *all* elements within the arrayed entity. Figure 11-10 details this process. When Apply To All is *not* checked, you must work your way through the columns (Dimension 1) and rows (Dimension 2) of the matrix to assign unique values or equations to each element within the arrayed variable. Figure 11-11 illustrates this process.

**Figure 11-10**

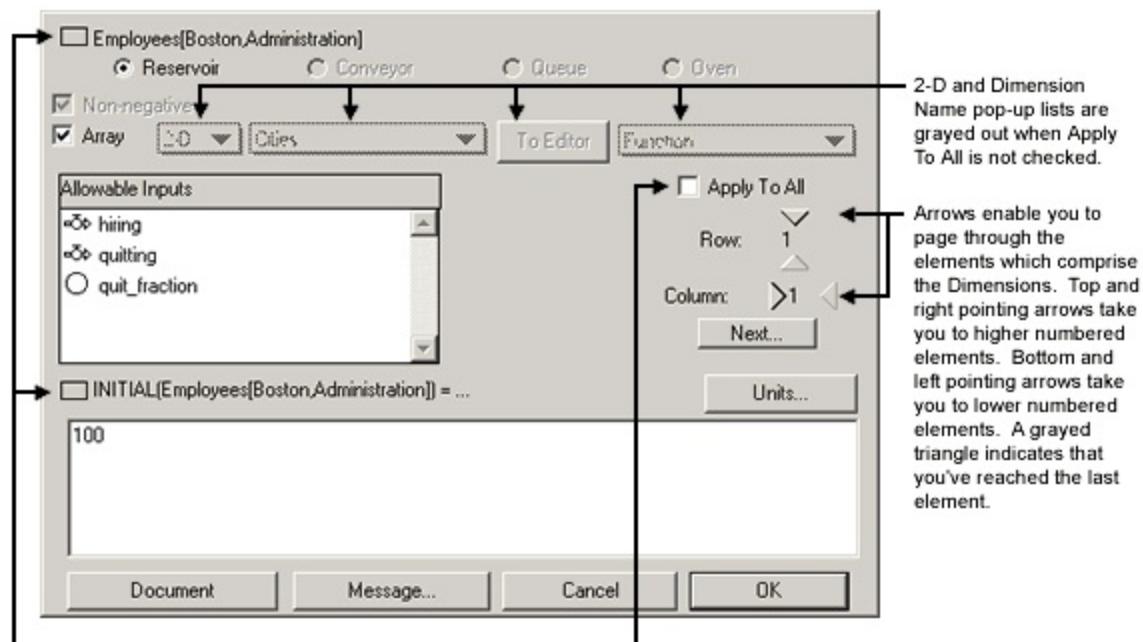
## Defining Logic of Two-Dimensional Arrays - Apply To All Checked



Variable name has been changed to reflect two dimensions. Since Apply To All is checked, the Dimension Names are used in the brackets.

**Figure 11-11**

## Defining Logic of Two-Dimensional Arrays - Apply To All Not Checked



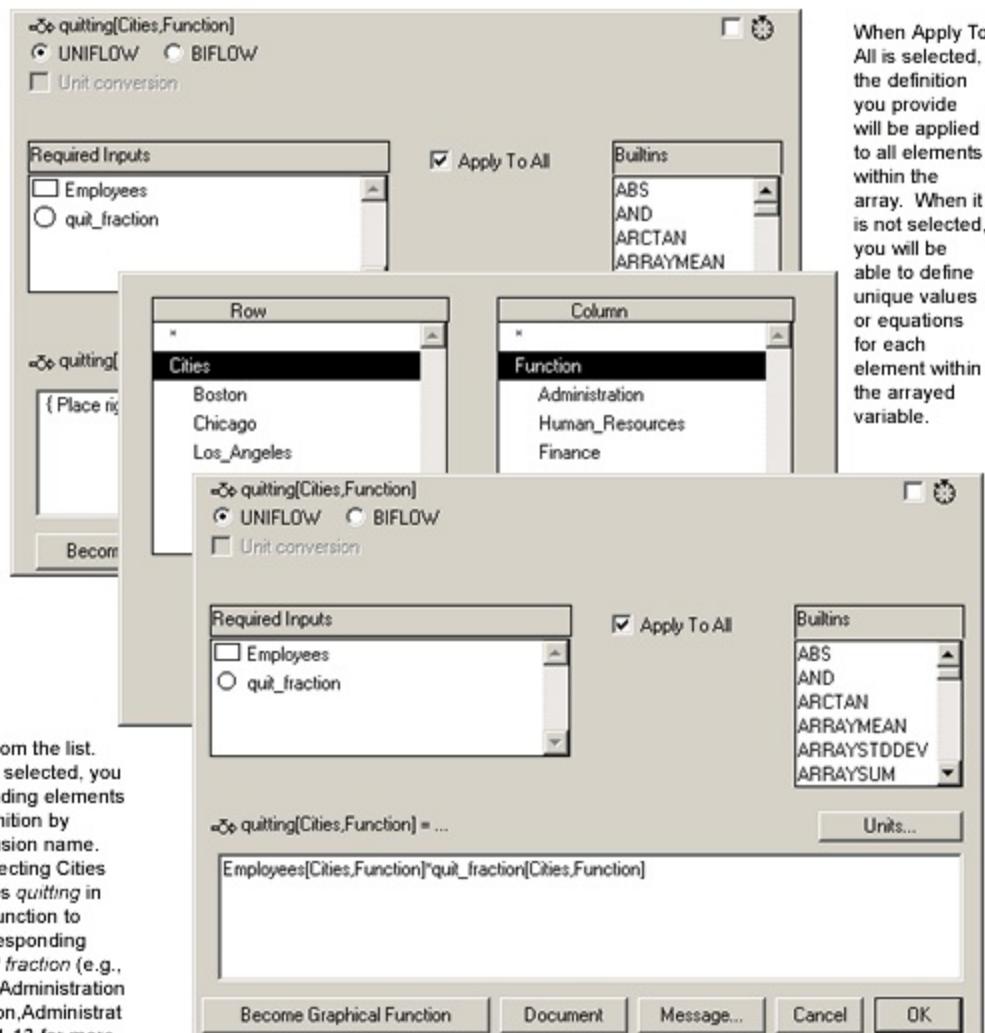
Whenever your arrayed variable has arrayed required inputs (or if you initialize a stock using arrayed allowable inputs), defining the equation logic

involves an additional step. When you click on the input name to move it to the equation, a secondary dialog will appear. Figure 11-12 illustrates the process of defining a variable using a two-dimensional arrayed input variable.

Figure 11-12

Defining Logic Using Two-Dimensional Arrayed Input Variables - Apply To All Checked

Whenever you click on a two-dimensional arrayed Required or Allowable Input variable, you'll be taken to a secondary dialog where you will choose which Array element to apply to the equation. Within the secondary dialog, you will find two lists containing all elements within the two dimensions of the Input. Define the relationship in terms of specific input elements, by selecting them from the list. Or, if Apply To All is selected, you can align corresponding elements in the equation definition by selecting the Dimension name. In this example, selecting Cities and Function causes *quitting* in each location and function to depend on the corresponding employees and *quit fraction* (e.g., Employees[Boston,Administration]"quit\_fraction[Boston,Administration]). See Figure 11-13 for more details on the secondary dialog.



When Apply To All is selected, the definition you provide will be applied to all elements within the array. When it is not selected, you will be able to define unique values or equations for each element within the arrayed variable.

The Dimension names in brackets mean "Make sure that the input and output elements in the arrayed variable correspond with one another." As it appears here, the equation ensures that *quitting* in each location depends upon the corresponding *Employees* and *quit fraction*. Dimension Names in the equation definition are only available when Apply To All is selected.

When Apply To All is not checked, use the paging arrows to navigate through each element in the arrayed variable and enter a unique value for each.

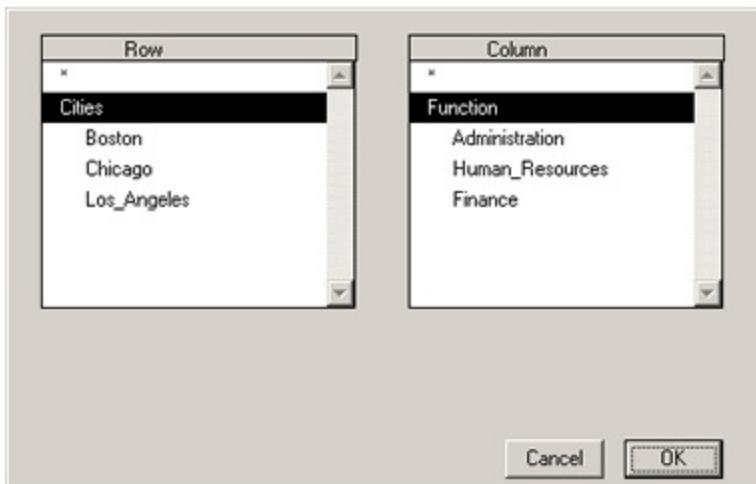
More detail on the secondary dialog for a two-dimensional Array is found in Figure 11-13.

Figure 11-13

Secondary Define Dialog - Two-Dimensional Arrays

This secondary dialog provides options for defining equations using two-dimensional inputs from the Required or Allowable Inputs lists. It will appear when you click on a variable name in the Required or Allowable Input lists. You must choose one selection from each of the two dimension lists to define the input.

Use the \* to define Array builtins, or graphs and tables, when you want to incorporate all the elements in the array. `Employees[*,*]` will return the sum of all Employees from both Dimensions of the Array. See the discussion of Array-specific builtins later in this chapter and in the Builtins chapter. `Employees[*,*]` will place all cells from the matrix into the graph/table.



The Dimension Name is used when you want to have the software align an element to the corresponding array element. For example, [Boston, Administration] will be used when in the [Boston, Administration] element; [Chicago, Administration] will be used when in the [Chicago, Administration] element. You can only use the Dimension Name when Apply To All is checked in the variable dialog.

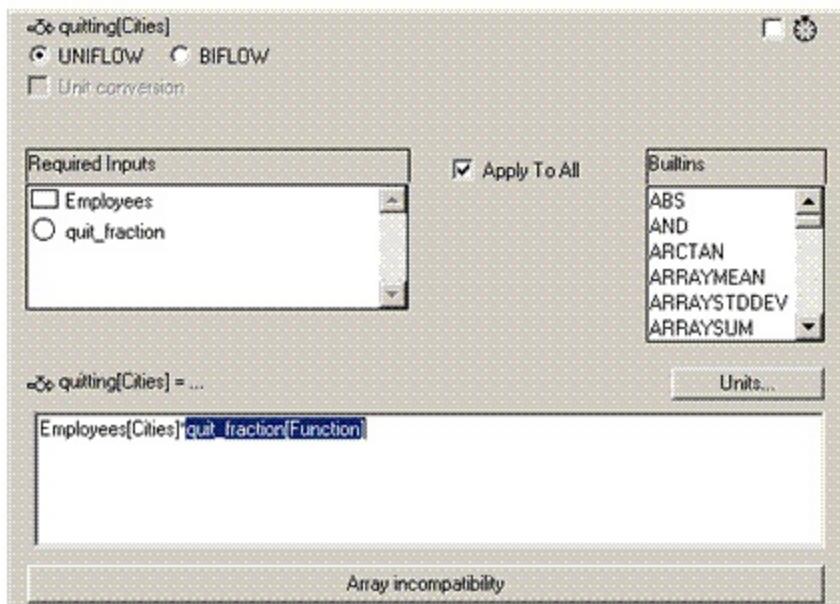
Use individual element names when you want only the specific element to apply to the equation. When Apply To All is *not* checked, you must use individual element names to define the equations. When Apply To All *is* checked, the use of individual element names is an option, but not required.

## [Related Topics](#)

# Resolving Incompatible Arrays

As you work with arrayed variables within your models, the software will help you ensure that the dimensions associated with inputs are consistent with the dimensions associated with an arrayed variable. Internally, the software will form a consistency check whenever you OK an equation dialog. The software will generate an "Array incompatibility" error message whenever arrayed variables have inconsistent dimensions. Three instances are shown in Figures 11-14a-c. For each, we note both the cause of the incompatibility and its cure.

*Figure 11-14a  
Incompatible Arrays: Causes and Cures*



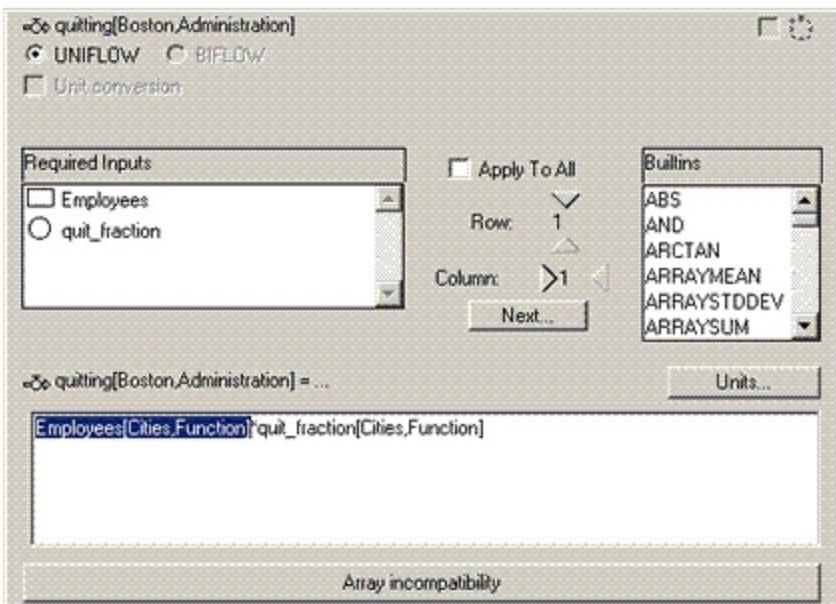
**Cause:** Two arrayed inputs are used to define an equation using Dimension Names, with Apply To All selected -- one Input's Dimension is different than the variable's Dimension. The software reports Array Incompatibility because the two do not make sense relative to each other.

**Cure:** To resolve this incompatibility, you can:

- Change the Dimension of the inconsistent Input to Match the variable (e.g., `Employees[Cities]*quit_fraction[Cities]`).
- Define the variable as a two-dimensional array that contains the Two Dimensions found in the Inputs (e.g., `quitting[Cities,Function]=Employees[Cities]*quit_fraction[Function]`).
- Clear the **Apply To All** check box and define each equation individually, using individual element names.

Figure 11-14b

### Incompatible Arrays: Causes and Cures



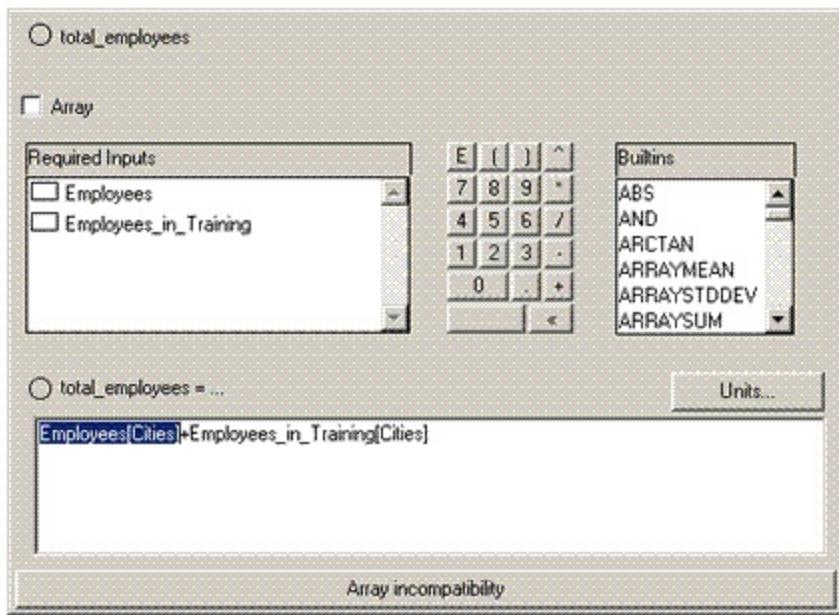
**Cause:** When **Apply To All** is not selected in an arrayed variable, you cannot define the equation of an individual element within the variable using the Dimension name when selecting variables from the Required or Allowable Input lists.

**Cure:** To resolve this incompatibility, you can:

- Remove all Dimension names from the equation definition and replace them with specific elements within the Dimension; instead of `[Employees[Cities,Function]*quit_fraction[Cities,Function]`, use `Employees[Chicago,Administration]*quit_fraction[Chicago,Administration]`.
- Select the **Apply To All** check box.

Figure 11-14c

### Incompatible Arrays: Causes and Cures



**Cause:** When arrayed variables are being used to define a non-arrayed variable, the Dimension name cannot be used.

**Cure:** To resolve this incompatibility, you can:

- Use only individual elements to define the equation. Instead of *Employees[Cities]+Employees\_In\_Training[Cities]* use *Employees[Boston]+Employees\_In\_Training[Boston]*.
- Use [Array Builtin](#) functions to operate on Inputs (e.g., *ARRAYSUM(Employees[\*])+ARRAYSUM(Employees\_In\_Training[\*])*).
- Change total employees into an arrayed variable with Cities as the Dimension.

 [Related Topics](#)

# Rules for Using Arrayed Variables

As you begin the process of defining the arrayed variables in your models, there are certain rules you need to keep in mind. The software will not allow you to create arrayed stocks or flows that violate the following rules.

## General Rules for Arrayed Variables

- Arrayed variables can be one-dimensional or two-dimensional only.
- The array status of a flow is determined by the stock(s) to which the flow is attached.
  - A flow must be attached to at least one stock before it can be arrayed.
  - An inflow from a cloud to an arrayed stock will be given the same dimension(s) as the stock.
  - An outflow from an arrayed stock to a cloud will be given the same dimension(s) as the stock.
  - When a flow runs from one arrayed stock to a second arrayed stock, *and* the stocks have identical specification of their dimensions, the flow will be assigned the same dimensions as the stocks.
  - When a flow runs from a one-dimensional stock to a second one-dimensional stock both upstream and downstream, but the stocks are *not* of the same dimension, the flow will become two-dimensional. The Dimensions will be assigned the order [upstream Dimension, downstream Dimension].
  - When a flow runs between a one-dimensional stock and a two-dimensional stock, *and* the Dimension for the one-dimensional stock is the same as *one* Dimension in the two-dimensional stock, the flow will take on identical dimensions to the two-dimensional stock.
  - When a flow runs between two one-dimensional stocks *and* the stocks use the same Dimensions, the Cross Flows check box becomes accessible within the flow dialog. When cross flows is check, this flow will become two-dimensional, using the same Dimension Name twice. As a result, you will be able to distribute the entire contents of the upstream stock across the elements of the downstream stock.
- An arrayed graphical function cannot be created unless Apply To All is checked in the arrayed variable's main equation box. After you click "Become Graph," you can uncheck Apply To All within the graphical function dialog. Then, separate curves can drawn for each element in the arrayed graphical function.

## **Rules for Arrayed Variables that do not share the same dimensions**

- Whenever two one-dimensional arrayed Inputs to an arrayed variable do not share a the same dimension, you need to take special care. In order to use Dimension names to define the third variable, the third variable must be two-dimensional, using the Dimensions of the two Inputs. This will allow for the full range of possible equations. If the variable is one-dimensional arrayed or non-arrayed, specific elements from the Inputs can be included in an equation. Array Builtins also are available in this situation.
- Dimension names can be used to define a two dimensional arrayed variable which contains both two-dimensional arrayed Inputs and one-dimensional arrayed Inputs as long as each two-dimensional Input shares both Dimensions with the variable and each one-dimensional arrayed Input shares at least one Dimension with the variable.

## **Rules for Arrayed Conveyors, Ovens, and Queues**

- All rules for non-arrayed discrete building blocks apply when working with Arrays. For more information about these rules, see [Conveyors, Queues, and Ovens](#). For information about flow prioritization, see [Flow Prioritization](#). For information about implicit connectors, see [Implicit Connectors](#). Some Array-specific situations are listed in [Array Tips and Tricks](#).
- Flows out of a Conveyor must use the same Dimension(s) as the Conveyor.
- Flows into and out of an Oven must use the same Dimension(s) as the Oven.
- A non-arrayed Conveyor or Oven cannot be created immediately upstream from an arrayed stock of any type. This is because, unless it is arrayed, there is no way to distribute the contents of a Conveyor or Oven to the multiple elements of the arrayed downstream stock. Placing a non-arrayed Reservoir or Queue between the non-arrayed Conveyor or Oven and the arrayed stock will enable you to put both into the same main chain. (Note: Also, an arrayed stock cannot be created immediately downstream from a non-arrayed Conveyor or Oven.)
- When Apply To All is unchecked in an arrayed Conveyor or Oven, constant values for Transit time (Conveyor) or Cook time (Oven) must be defined from the outflow dialog. If Apply To All is then unchecked in the outflow dialog, access to the Transit time or Cook time will be enabled within the Conveyor or Oven dialog.
- If you uncheck Apply To All in the define dialog for Transit time (Conveyor)

or Cook time (Oven), constant values for Transit time and Cook time will not be available within the Conveyor or Oven dialog. If you uncheck Apply To All in the Conveyor or Oven dialog, then access to constant values for Transit time or Cook time will be enabled within the Conveyor or Oven dialog.

## Rules for Prioritization of Outflows

- a. If you have a two-dimensional outflow from a non-arrayed Queue or non-negative Reservoir, the priorities of the flows are determined by the order of the Dimensions in the two-dimensional arrayed variable - the first Dimension gets first priority. In this case, if you have two Dimensions in the flow, where Dimension 1 = {a,b} and Dimension 2 = {x,y}, then flow priority will be [a,x], [a,y], [b,x], [b,y].
- b. If you have an outflow from a one-dimensional Queue or non-negative Reservoir into a two-dimensional stock (the flow will be two-dimensional), the priorities of the flow out of any particular element are determined by the Dimension the two have in common. If there are not sufficient units in the one-dimensional stock and its inflow(s) to cover the demand for the two-dimensional outflow, the software will meet the demand as follows. Suppose the one-dimensional stock uses Dimension 1 (Dimension 1 contains elements {a,b}) and the two-dimensional flow contains Dimension 1 and Dimension 2 (Dimension 2 contains elements {x,y}). The priority for flows out of the one-dimensional stock's element a will be [a,x] then [a,y]. The priority for flows out of element b will be [b,x] then [b,y].

## Using Arrays with Other Features

- The basic Cycle-Time operations will be the same with Arrays as with non-arrayed variables. However, you must keep in mind that time stamped material will be measured within each element, not the entire Array. Figure 11-15 illustrates this concept.

*Figure 11-15 Cycle-Time Measurement with Arrays*

- When diagram animation is turned on in the Model Prefs... dialog, the animation you see on the diagram will be of the first element in the arrayed variable.
- Arrayed stocks (and thus flows) are not permitted inside Sub-model spaces or Space Compression spaces. Arrayed converters can be created in or pasted into Sub-model and Decision Process Diamond spaces.
- The summer converter can accept individual array elements in its equation box or if you select the \* it will automatically insert the [ARRAYSUM](#) Built-in.
- When you click on an arrayed variable to load it into a graph, if you choose the \*, you will get all the elements in that variable in a table, and the first 5 elements in that variable in the graph. You cannot use the Dimension name to define a Graph or Table.
- When defining Numeric Displays, Sliders, Graphical Input Devices, Loop Objects, Warning Devices, Switches, Knobs, and LIDs, you can only choose one element from within the Array to define the object (i.e., you cannot define the object using the Dimension name).
- When defining the Loop tracing button, you will click the variable name into the dialog. The individual elements will not be available to define the Loop Button.

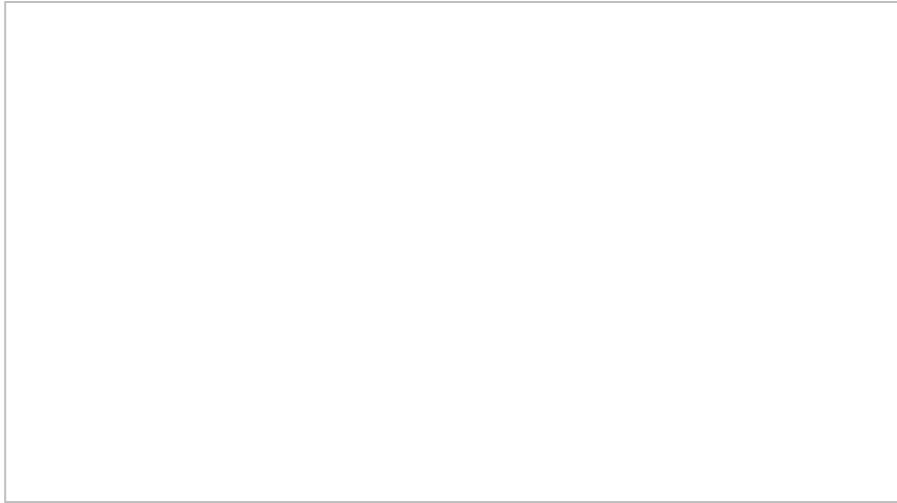
 [Related Topics](#)

# Array Tips and Tricks

In many situations, a tip from us might help you set up an Array in just the right way for your situation. Below you will find a few tips and tricks that you may find useful.

- Here's how to move material from one element in an arrayed stock, back into another element in the same arrayed stock. Draw an outflow into a cloud, and then draw an inflow out of a cloud back into the same stock. Connect the two flows using a connector. Then, with Apply To All unchecked, you can define the inflow in terms of the outflow to move material from one element to the other. Figure 11-16 illustrates this concept.

*Figure 11-16 Moving Material from One Element in an Arrayed Stock to Another*



- If you find you want to use the sum of all the elements within an arrayed variable as an input into another variable, use the ARRAYSUM Built-in to capture the sum of the elements.
- While creating and defining your two-dimensional arrayed variables, you might find it useful to keep a chart on paper (a matrix similar to what you saw in Figure 11-9). This will help keep a clear picture of the elements and cells you are modifying.
- Since Arrays reduce diagram clutter, the temptation is to overload the structure with more detail than you need. Avoid this temptation. Keep your models as simple as possible. As a practical matter, ask yourself if you'd use copy and paste to create the structure if arrays were not available. If the answer to this question is "no," the structure you are contemplating is

probably too detailed.

- Before arraying variables, create and test the non-arrayed structure. Doing so will help you to manage the analytic complexity of working with arrays.
- Use short names when creating arrayed variables. Since the software will list the name plus the Dimension name(s) on input and output devices, shorter names will help you know what is assigned to what device.

 [Related Topics](#)

# Array-Specific Builtins

The software includes Builtin functions to support your use of Arrays. These Builtins are ARRAYIDX, ARRAYMAX, ARRAYMAXIDX, ARRAYMEAN, ARRAYMIN, ARRAYMINIDX, ARRAYRANK, ARRAYSTDDEV, ARRAYSUM, ARRAYVALUE, and DIMSIZE. The basic definition of these Builtins is summarized here. For more information about how to use these Builtins, see [Array Functions](#).

## **ARRAYIDX([<dimension index>])**

The ARRAYIDX Builtin gives the numerical index (starting at 1) of the current array element in the array holding the equation. The <dimension index> is 1 for the first dimension (default when not specified) or 2 for the second dimension.

## **ARRAYMAX(<arrayed variable>)**

The ARRAYMAX Builtin finds the maximum value in the array. The calculation can be done in either an arrayed or a non-arrayed variable.

## **ARRAYMAXIDX(<arrayed variable>[, <dimension index>])**

The ARRAYMAXIDX Builtin gives the index of the maximum-valued element in the array, where 1 refers to the first element, 2 to the second element, and so on. The optional <dimension index> specifies which dimension is desired. The <dimension index> is 1 for the first dimension and 2 for the second dimension (two-dimensional arrays only). The <dimension index> is 1 if it is not specified.

## **ARRAYMEAN(<arrayed variable>)**

The ARRAYMEAN Builtin calculates the arithmetic mean of the elements in a one or two-dimensional Array. The calculation is defined as the sum of the elements, divided by the number of elements in the Array. The calculation can be done in either an arrayed or a non-arrayed converter or flow.

## **ARRAYMIN(<arrayed variable>)**

The ARRAYMIN Builtin finds the minimum value in the array. The calculation can be done in either an arrayed or a non-arrayed variable.

## **ARRAYMINIDX(<arrayed variable>[, <dimension index>])**

The ARRAYMINIDX Built-in gives the index of the minimum-valued element in the array, where 1 refers to the first element, 2 to the second element, and so on. The optional <dimension index> specifies which dimension is desired. The <dimension index> is 1 for the first dimension and 2 for the second dimension (two-dimensional arrays only). The <dimension index> is 1 if it is not specified.

## **ARRAYRANK(<arrayed variable>, <rank number>[,<secondary sort variable>])**

The ARRAYRANK Built-in gives the numerical index of the <arrayed variable> with the given <rank number> when the array is sorted in ascending order. The optional <secondary sort variable> specifies the secondary sort field for variables with the same value. The <secondary sort variable> must be the same size as the <arrayed variable> or the secondary sort will not occur (this is only enforced when you simulate the model, so you will not be warned).

## **ARRAYSTDDEV(<arrayed variable>]**

The ARRAYSTDDEV Builtin calculates standard deviation for all the elements in a given arrayed variable. The calculation can be done either in an arrayed or a non-arrayed variable.

## **ARRAYSUM(<arrayed variable>]**

The ARYSUM Builtin calculates the sum of all the elements in a given one or two-dimensional Array. The calculation can be done in either an arrayed or a non-arrayed variable.

## **ARRAYVALUE(<arrayed variable>, <row number>[, <column number>])**

The ARRAYVALUE Builtin gives the value that is in the specified element of the given array. Both <row number> and <column number> start at 1. If an array is one-dimensional, only the row number can be specified. If an array is two-dimensional, both the row number and the column number must be specified. This Builtin can be used directly with the ARRAYMINIDX and ARRAYMAXIDX Builtins.

## **DIMSIZE([<dimension index>])**

The DIMSIZE Built-in gives the number of elements in the array holding the equation. The <dimension index> is 1 for the first dimension (default when not specified) or 2 for the second dimension.

 [Related Topics](#)

# Introduction to Modules

Modules are self-contained models that you can connect to other models. Modules allow you to break a single model into well-defined "chunks". Each module within a model is a cohesive model on its own, which you can run separately or within the larger model.

By using modules, you can:

- Build small, self-contained portions of a model, one-at-a-time.
- Test a single portion of a large model.
- Represent a hierarchical system in a model, by making each level of the hierarchy into a separate module, with each module linked to one or more levels above it
- Work with teams to build a complex model by having each team member build separate modules that are later linked together.
- Reuse portions of a model in as many models as you want.
- Create very large models (models that may have previously exceeded size limits).
- Incorporate locked models into other models.

You can create as many modules as you need, and you can incorporate as many modules as you want into a single model. This allows you to create very large or complex models that are broken down into cohesive, self-contained pieces.

---

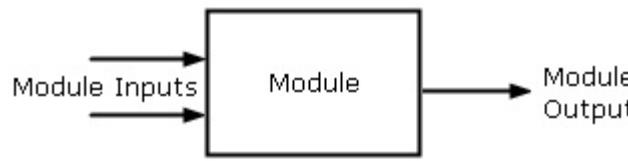
*Note:* When you import or export data from a model that contains modules, all module data in the model are also imported or exported. For more information, see [Importing and Exporting Module Variables](#).

# Understanding module concepts

To understand how to use modules to build a hierarchy, you need to understand the basic concepts of working with modules:

**Module** – A module is a self-contained model that specifies the inputs it needs to run and the output it will generate when the module is run. Once a module is built, its inputs and outputs can be connected to variables in other models (or modules). These variables provide the input values for the module and accept the output values from the module.

You can think of a module as a "black box" that has inputs and outputs. To connect the module to other modules, you need to specify what information needs to come from other modules ("Module Inputs") and what information is sent out to other modules ("Module Outputs").



**Module input** – A module input is a variable (for example, a stock or a converter) that contains information that the module needs in order to run. When you build a module, you decide what information in the module needs to come from outside the module and you create an entity to hold this information. These are the module inputs. When you are ready to use the module, you select the variable outside the module that will provide the value for the module input by *assigning* the module input to a variable.

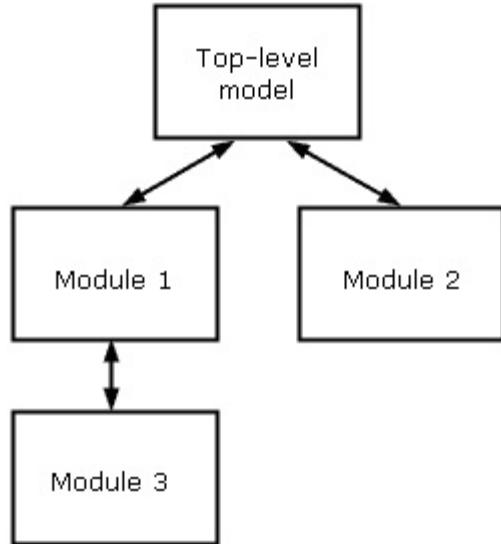
If you think of a module as a mathematical function, unassigned module inputs (and outputs) correspond to the *formal parameters* of a function. When you assign the input to an entity outside the module, you are assigning the *actual parameter*.

You can assign a module input to any variable that is connected to the module that contains the module input. For the module to run, an entity defined as a module input must be assigned to an entity outside the module that will provide the input value.

**Module output** – A module output is a variable in the module that contains information generated by the module. A module's outputs can be assigned to other entities when that module is connected to an entity or another module (thus, the module output value is assigned to a module input). This allows you to have the results of one module become the input value to another entity or module. You can manually define a variable as a module output, or you can assign it as a module input and it will be automatically defined as a module

output.

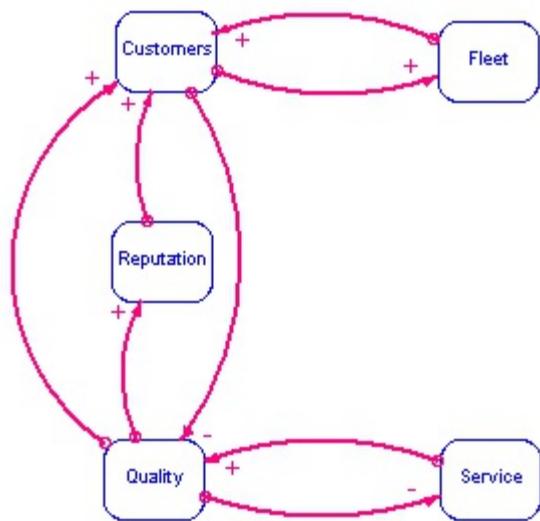
**Levels** – Modules within a hierarchy exist in levels. The hierarchy begins with a top level, which is a model. The top-level model contains connections down into one or more modules. Although several modules can exist on the same level, you can view only one module at a time. You can navigate up and down levels to view modules, but you cannot navigate sideways.



[Related Topics](#)

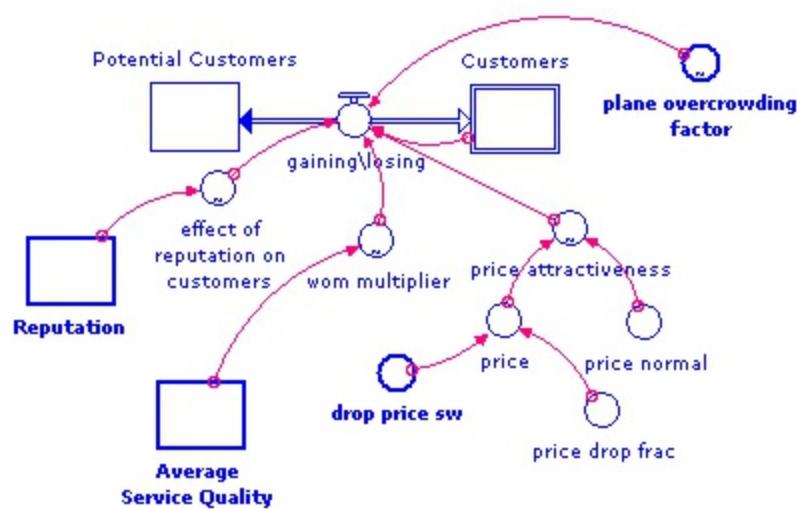
# A Module Example

Imagine you are building a business model for an airline. The causal loop diagram for the model (the “dynamic hypothesis”) might look like this:



This is the top level of the model, which comprises five modules (Customers, Fleet, Reputation, Quality, and Service). The connectors between modules indicate how values flow from one module to another. For example, you can see that variables flow in both directions between the Customers and Fleet modules; however, variable values flow only in one direction from the Quality to Reputation modules, and from the Reputation to Customers modules. Similarly, the connectors show you that there is no direct connection between the Service and Customers modules.

You develop each underlying module as a separate model. For example, this is what the “Customers” (customer acquisition) module might look like:

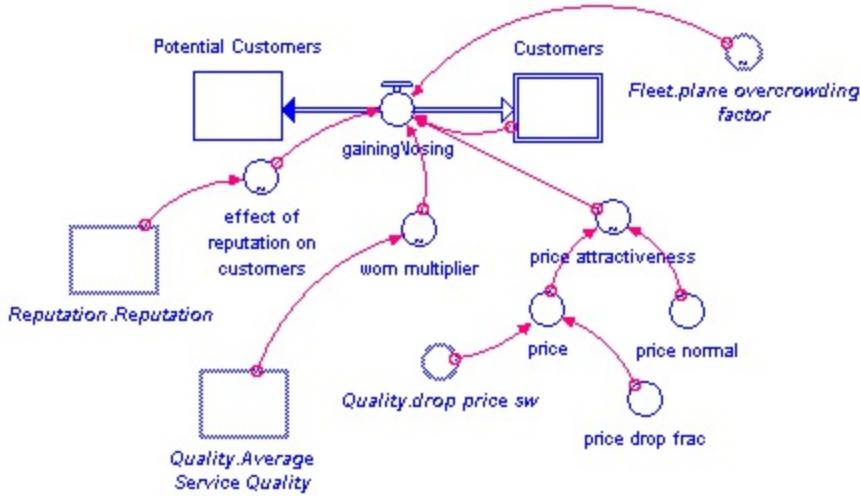


Note that this module depends on several inputs from elsewhere in the model and will not be able to run until those inputs are assigned. The unassigned module inputs are indicated visually by a double-thick border for their entity

icons and a bold font for their labels:

- **plane overcrowding factor**
- **Reputation**
- **Average Service Quality**
- **drop price sw**

When these module inputs are [assigned to variables](#) elsewhere in the model, the module is fully defined and able to simulate as it stands in isolation.



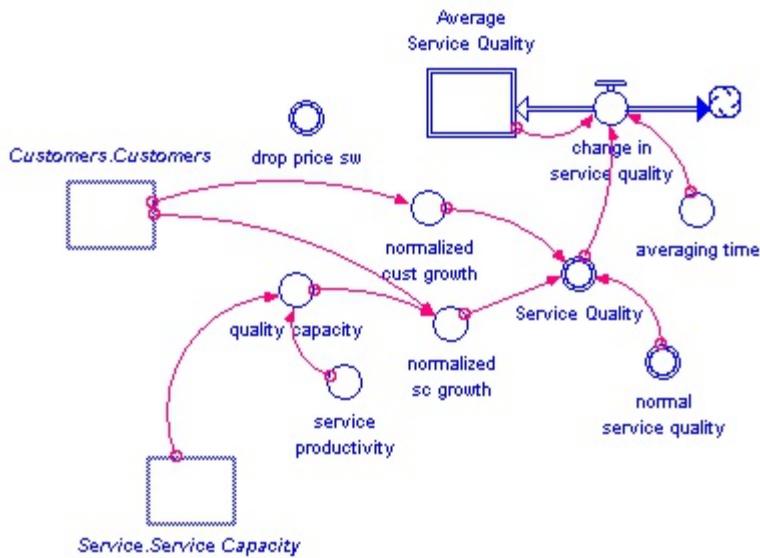
Notice that the previously unassigned module inputs are now named this way:

- *Fleet. plane overcrowding factor*
- *Reputation. Reputation*
- *Quality. Average Service Quality*
- *Quality. drop price sw*

The part of the name before the period indicates the module where the variable is defined (for example, “Fleet”, “Reputation”, and “Quality”). The part of the name after the period indicates the variable’s actual name, as defined in the module where the variable is defined (for example, “plane overcrowding factor” and “Average Service Quality”). Notice that the assigned module inputs are now indicated visually by an italic font for the entity’s label, and a double-thick gray border for the entity’s icon.

This module also contains a variable that is defined as a module output (remember that the Customers module had connectors indicating that it had inputs and outputs to both the Fleet and Quality modules). In the Customers module, the “Customers” entity is defined as an output. Outputs are indicated visually by a double-line border (which looks like an O in converters and flows). The name for outputs appears in a regular font.

Now, let's look at the Quality module, which tracks service quality. This is the way the module looks after the module inputs have been assigned:



In this module, there are two module inputs ("Customers.Customers" and "Service.Service Capacity") and three module outputs ("drop price sw", "Average Service Quality" and "normal service quality").

A module input can be assigned to

- Variables that are directly connected to this module with an incoming connector.
- Variables that are defined as module outputs in any module that is connected to this module via an incoming connector.
- Variables defined as module outputs in any submodule that has a connector into the module input.

If you look back at the top-level of the module, you'll see that the Customers and Service modules are both connected to the Quality module via incoming connectors, so module outputs from both of those modules are available to be assigned as inputs to the Quality module.

[Related Topics](#)

# Introduction to Working with Module Inputs and Outputs

To prepare a module for use in a hierarchy, you need to decide which variables in the module get their values from other modules, and which variables in the module (if any) output the module's results. In most cases, you explicitly indicate which variables are module inputs and which are module outputs by *defining module inputs and module outputs*.

When you are ready to use the module in a hierarchy, you need to assign each module input to an actual variable. To make variables available for assigning to a module input, you first need to make explicit connections (with [connectors](#)) between the module and other modules, or between the module and a variable in the model. Once those connections are established, you can assign module inputs.

You can also use the [Ghost tool](#) to connect variables. Using the Ghost tool allows you to automatically assign module inputs and outputs as well as make connections between modules and other modules or variables.

## What do you want to do?

[Define module connections](#)

[Define variables as module inputs or module outputs](#)

[Assign module inputs](#)

[Use the Ghost tool to assign module inputs and outputs](#)

 [Related Topics](#)

# Defining Module Connections

Before you can assign inputs in your modules, you first need to establish the connections that determine the source of data flowing into the module, and the recipient of data flowing out of the module. To define the connections, you use [connectors](#) to draw the relationship between modules and other modules, or between modules and variables in other models.

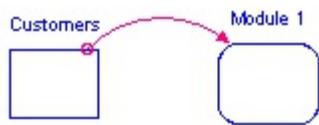
A completely defined module (one with all module inputs assigned and all entities defined) can be run independently of the model that contains it and any other modules it is connected to.

# To define module connections

1. On the Map or Model layer, draw a connector from the module or entity that will provide variables as outputs to the module that will accept those variables as input. In the following example, variables in "Module 1" are available as inputs for variables in "Module 2":

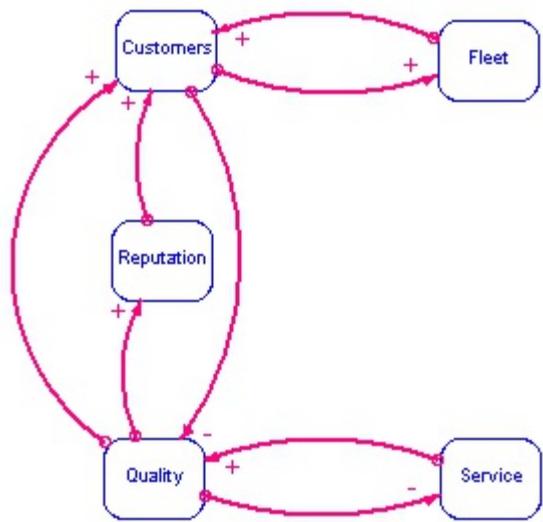


In the next example case, the "Customers" variable is available as an input for variables in "Module 1".



2. Continue drawing connectors to define the input/output relationships among all modules and entities in the model.

The following example (from the sample hierarchy discussed in [A Module Example](#)) shows how some modules are related by both inputs and outputs, where others are related by only one or the other, and still others are not directly related at all.



You can change the relationships at any time by adding or removing connectors. If you make changes to the connectors after you [assign input variables](#) in each module, you may need to reassign inputs to their proper outputs to be able to run the model.

 [Related Topics](#)

# Defining Module Inputs and Module Outputs

To allow model variable values to be used as values in a module, you must define module variables as either module inputs or module outputs.

- A module **input** is a variable in the module (a stock, a flow, or a converter) that contains information that the module needs in order to run. Although you can explicitly define a variable as a module input, you are not required to; when you [assign a variable as a module input](#) (by using the **Assign Input To** command), that variable is automatically made a module input.
- A module **output** is a variable in the module that contains information generated by the module and that can supply a value to other connected entities or modules. Variables directly connected into a module (via a connector from the variable to the module) can be assigned to a module input in the connected module without explicitly defining them as outputs. These variables are providing data into the connected module. On the other hand, variables that are providing data out of the module (into the model that contains the module, or into another module) must be explicitly defined as outputs.

---

**Note:** You can also define a variable to be neither a module input nor a module output by selecting the **Neither** option.

# To define a variable as a module input or module output

1. Navigate into the module for which you want to define a variable as a module input or output.
2. Right-click the variable that you want to define, choose **Module**, and then choose **Accept Input**, **Provide Output**, or **Neither**.

---

*Note:* Changing a variable's input/output definition disconnects any existing assignments made between that variable and another variable.

---

 [Related Topics](#)

# Assigning Module Inputs

In order to use a module in a hierarchy, you need to assign each module input to an actual variable. You can assign a module input to

- Variables that are directly connected to this module with an incoming connector.
- Variables that are defined as module outputs in any module that is connected to this module via an incoming connector.
- Variables defined as module outputs in any submodule that has a connector into the module input.

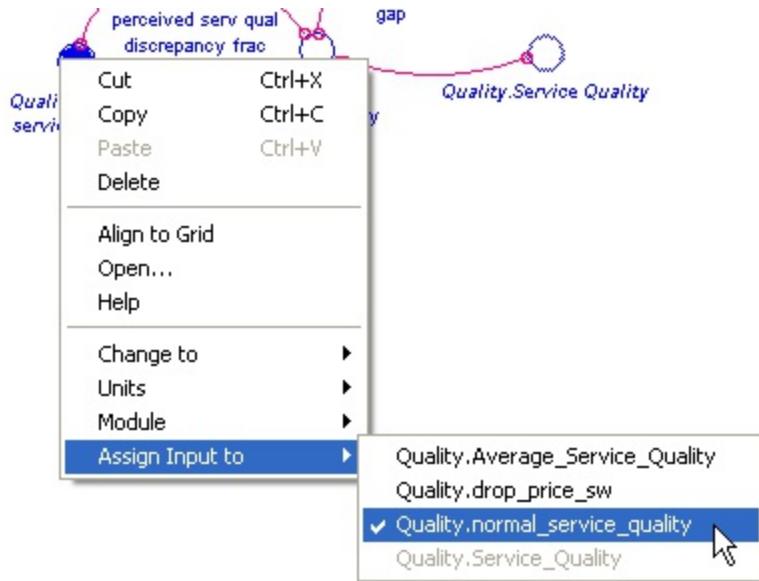
There are two ways to assign module inputs:

- From within a module, by selecting an available variable that is outside the module.
- From outside of a module, by selecting an entity that has a connector into or out of a module and assigning it to a variable that is inside the module.

Assigning module inputs from inside the module is possible only when you have access to the module. If the module is locked, you can only assign module inputs from outside the module.

# To assign module inputs from inside a module

1. Navigate into a module to which you want to assign inputs by double-clicking the module entity on the Map or Model layer.
2. In the module, right-click the entity to which you want to assign an input and then choose **Assign Input to**. A list of variables that can be assigned to this entity appears.



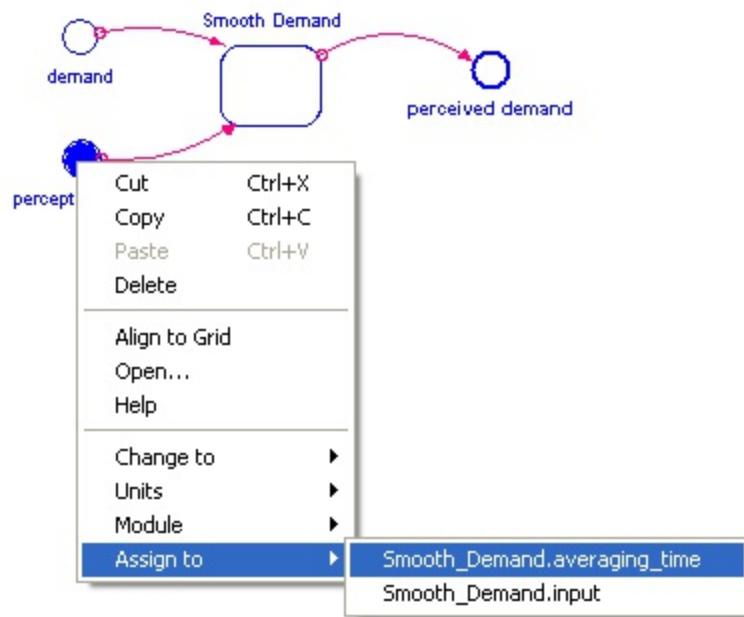
- If the module input has already been assigned, a check mark appears next to the variable assigned to it ("Quality.normal\_service\_quality", in the above example).
  - If a variable is assigned to another entity in the module, it appears grayed out ("Quality.Service\_Quality" in the above example).
  - If a variable is a flow, it can only be assigned to a flow if the module output flows into a cloud (sink) and the module input originates from a cloud (source). If these conditions are not met, the variable appears grayed out.
  - If a variable is a flow and has already been assigned to a flow in the model, it appears grayed out for all other flows in the model.
- Note:* Although module outputs that are flows can only be assigned to one other flow in the model, they can still be assigned to as many converters as you wish.
- All other listed outputs have not yet been assigned and can be assigned to inputs.
3. Select an available variable to assign to the module input. When an input is already assigned, choosing a different output reassigns the input to the new variable. When you assign a module input, the entity's name is

automatically changed to that of the output.

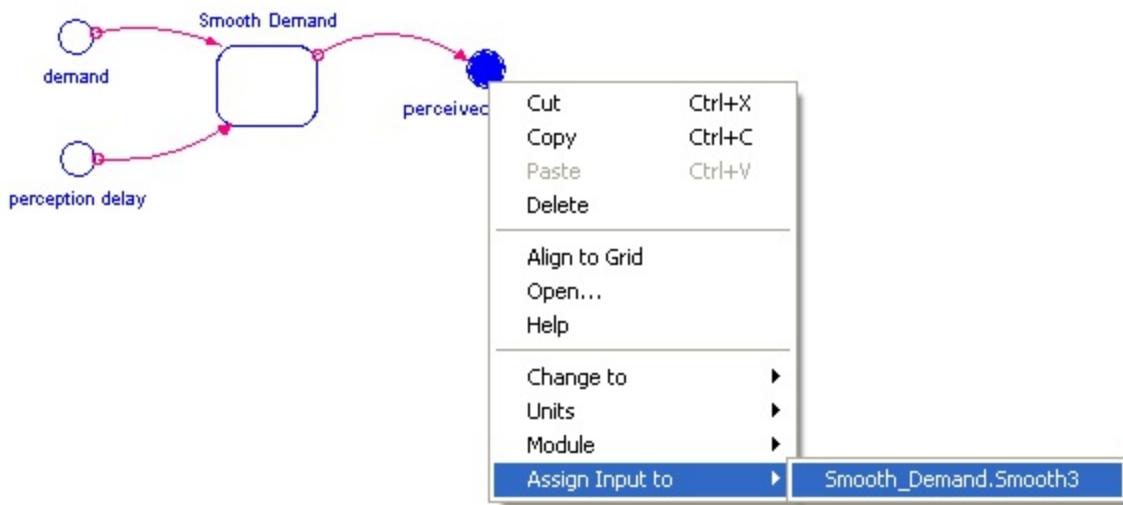
# To assign module inputs from outside a module

When you are assigning module inputs from outside a module, you can assign both input variables to the connected module as well as return values from within the module.

1. To assign an entity as an input value to a module input in a connected module, right-click the entity and then choose **Assign to**. A list of module inputs to which you can assign the entity appears:



- If the entity has already been assigned to a module input, a check mark appears next to the variable assigned to it.
  - If a variable is assigned to another entity, it appears grayed out.
  - All other listed module inputs have not yet been assigned and can be assigned to the entity.
2. Select an available module input to assign to the entity. When an entity is already assigned to a module input, choosing a different module input reassigns the entity to the new variable.
  3. To assign a module output as a return value to an entity connected to the module, right-click the entity and then choose **Assign Input to**. A list of module outputs that can be assigned to the entity appears.



4. Select an available module output to assign to the entity. When an entity is already assigned to a module output, choosing a different module output reassigns the entity to the new variable. When you assign an entity to a module output, the entity's name is automatically changed to that of the module output.

 [Related Topics](#)

# Using Ghosts to Assign Module Inputs and Outputs

Using the [Ghost tool](#) is the simplest and fastest way to assign module inputs and outputs. When you use the Ghost tool to define which module variables are inputs and outputs to other modules, the software automatically defines the appropriate module connections (that is, which modules are connected to which other modules) and assigns the appropriate variables as either module inputs or module outputs.

---

*Note:* You can use this method between modules that are one level up or down from each other or between modules across any level in the hierarchy. You cannot use the Ghost tool between modules that are multiple levels up or down from each other.

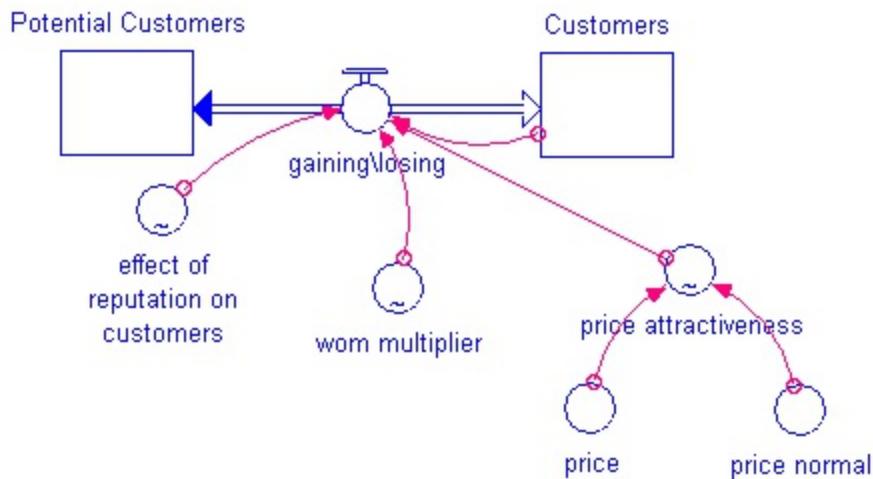
---

# To assign module inputs and outputs with the Ghost tool

1. In a model that contains one or more modules, navigate into a module that contains an entity whose value will be output to another module. In the following example, the model contains two modules: Customers and Fleet.

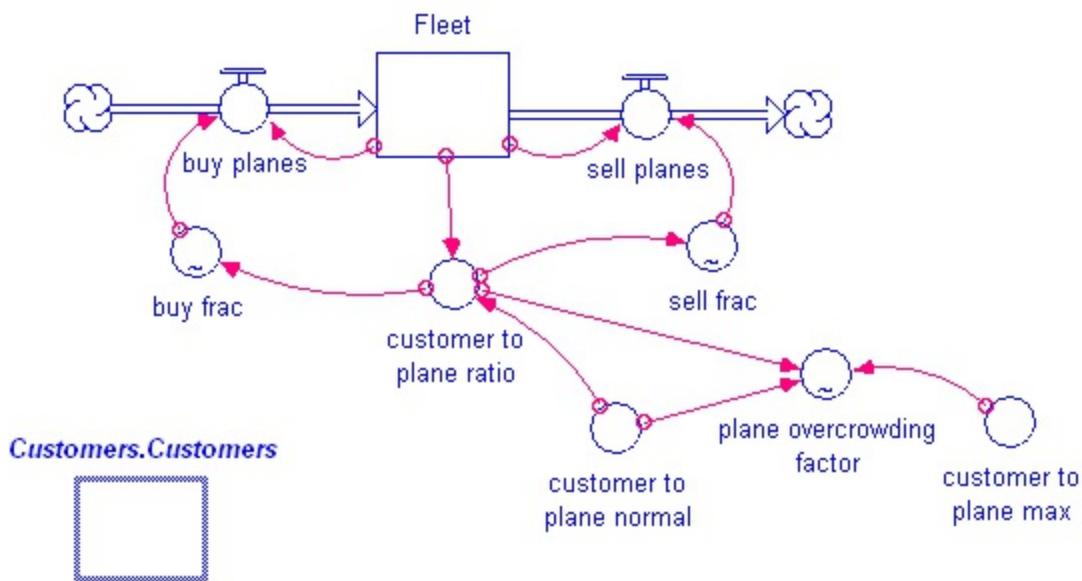


In this example, the Customers module contains a Stock called "Customers" that will serve as an input to the Fleet module. We'll begin by navigating into the Customers module:

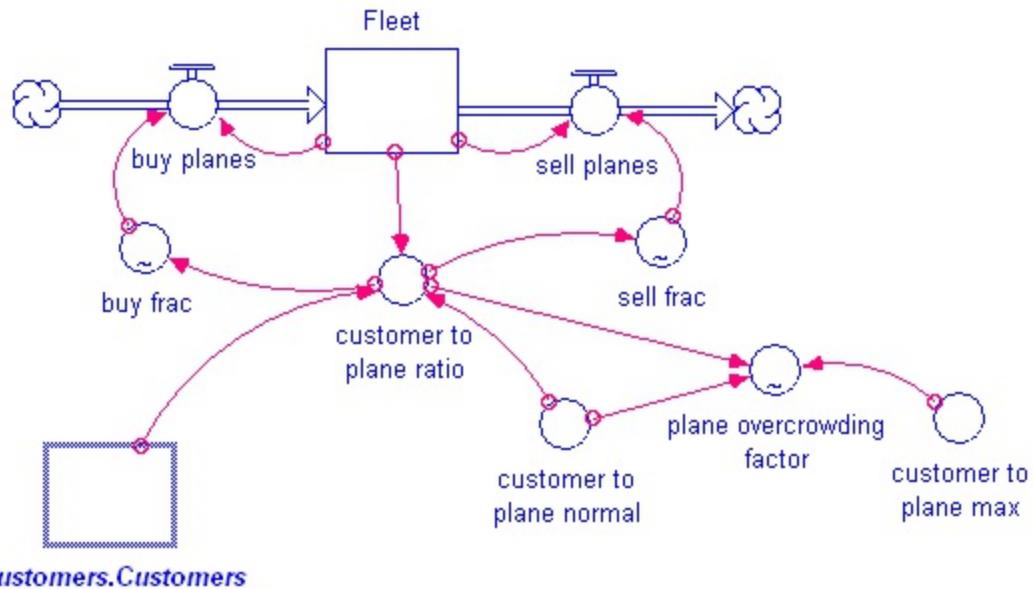


2. Select the Ghost tool.
3. Click the center of the entity that will serve as a module input to another entity or module. The cursor changes to a small version of the ghosted building block's icon. In this example, we'll be defining the "Customers" Stock as an input to the Fleet module, so we would click the center of the "Customers" Stock and the cursor changes to a small Stock icon.
4. Navigate to the module that will accept the ghosted entity as an input (either up or down one level from the current module, or at the same level as the current module. In our example, we'll navigate up one level and then back down one level into the Fleet module.
5. Click anywhere on the module diagram to place the ghosted entity. The ghosted entity appears and its name includes the name of the module that is providing the value. In our example, the ghosted entity is the "Customers.Customers" entity (that is, the "Customers" Stock from the

Customers module):



6. Use a connector to draw the connection between the new module input and the entity to which it provides an input value. In our example, we'll draw a connection between the new "*Customers.Customers*" entity and the "customer to plane ratio" converter:



7. If you right-click the "*Customers.Customers*" entity and choose the **Module** command, you will see that it has been automatically set to **Accept Input**. When you navigate back out of the module, you will see the connection between the modules has been automatically drawn:



If you navigate into the module that is providing the input value to the

module (the Customers module in this example), right-click the entity that you ghosted into the other module, and choose the **Module** command, you will see that it has been automatically set to **Provide Output**.

8. Repeat steps 1 - 6 to assign module inputs and outputs between modules in your model.

---

*Note:* If you delete a ghosted variable and it is not being used anywhere else in the model, the variable that served as the module output no longer serves as a module output. If there are no other variables shared between the modules (in the same direction), the software also automatically deletes the connection between modules.

---

 [Related Topics](#)

# Testing and Troubleshooting Modules

When a hierarchy is correctly defined, you can run the entire model and all modules connected to the model will run. A model will not be able to run, however, if any of the following conditions exist:

- Entities (or modules) within the model are invalid.
- A module within the model has inputs in it that are not assigned, either because they have not yet been assigned or there are not enough connectors going into the module to provide input values.
- A module within it has connectors into it that are not used.

If a module is correctly defined, with all module inputs assigned and variables defined, you can run the module on its own to test it. When you run a module in isolation, module inputs are held constant using the same rules as the **Run Selected Sectors** option.

 [Related Topics](#)

# **Creating Modules**

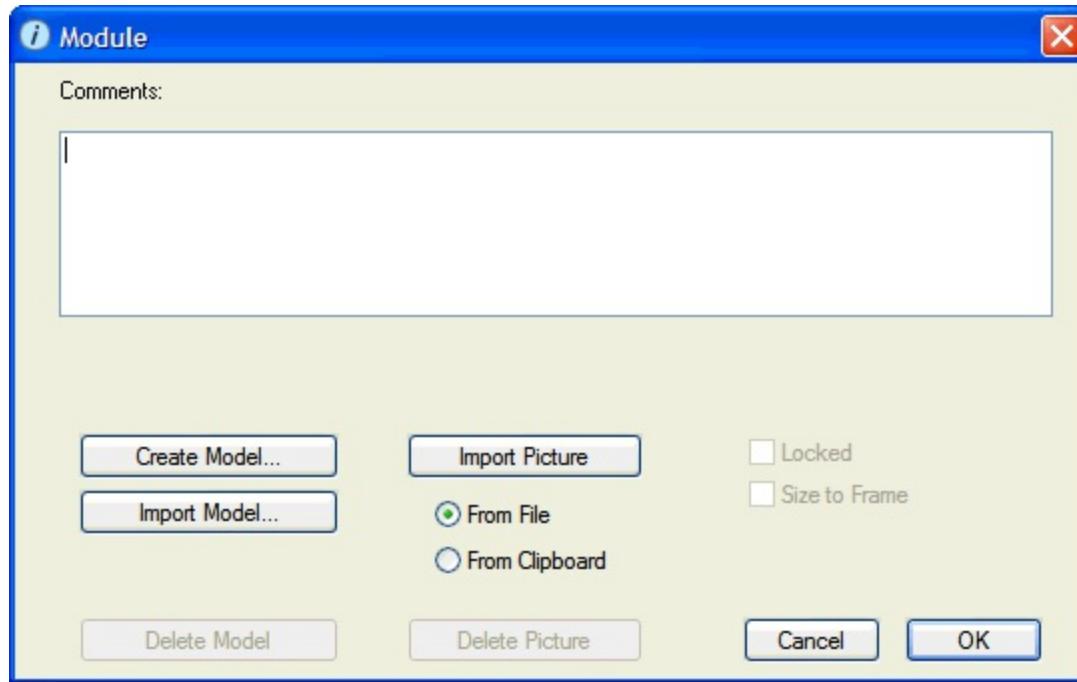
You can create a module either by creating a model, saving it, and then importing it into a module; or by inserting a module into a model and then building the module directly within the model. Both procedures are described below.

# Creating a module from an existing model

1. Create a model as you ordinarily would, but also [define its module inputs and outputs](#).
2. Save the model.
3. [Import the saved model into the module](#).

# Creating a module from within a model

1. In an existing or new model, click the  button on the Map or Model toolbar.
2. Click the diagram surface to place the module entity on the layer.
3. Name the module entity as you would any other entity.
4. On the Model layer, double-click the new module entity. The Module dialog box opens.

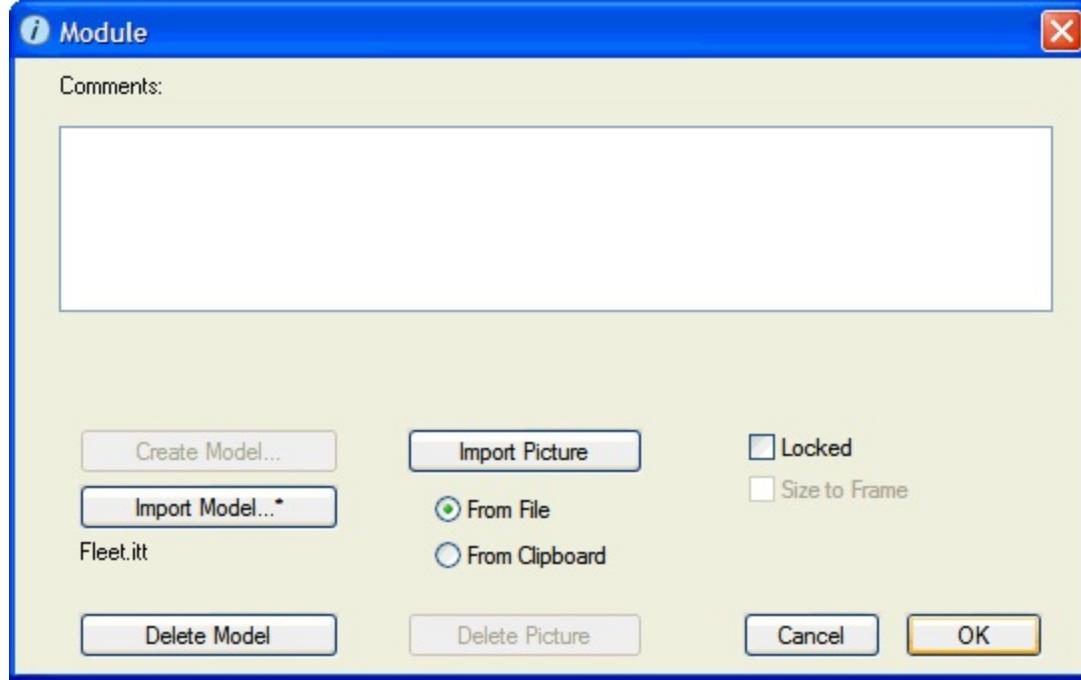


- 
5. Click **Create Model**. The Save As Template dialog box opens.

Note: You can also add a module to the model by importing an existing model as a module. For more information, see [Importing Models as Modules](#).

---

6. In the File name box, type the name of the new model.
7. Leave the default selections for the file location (**Modules** directory) and file type (**iSee Template doc (\*.ITT)**).
8. Click **Save**. The model's name appears beneath the **Import Model** button in the Module dialog box.



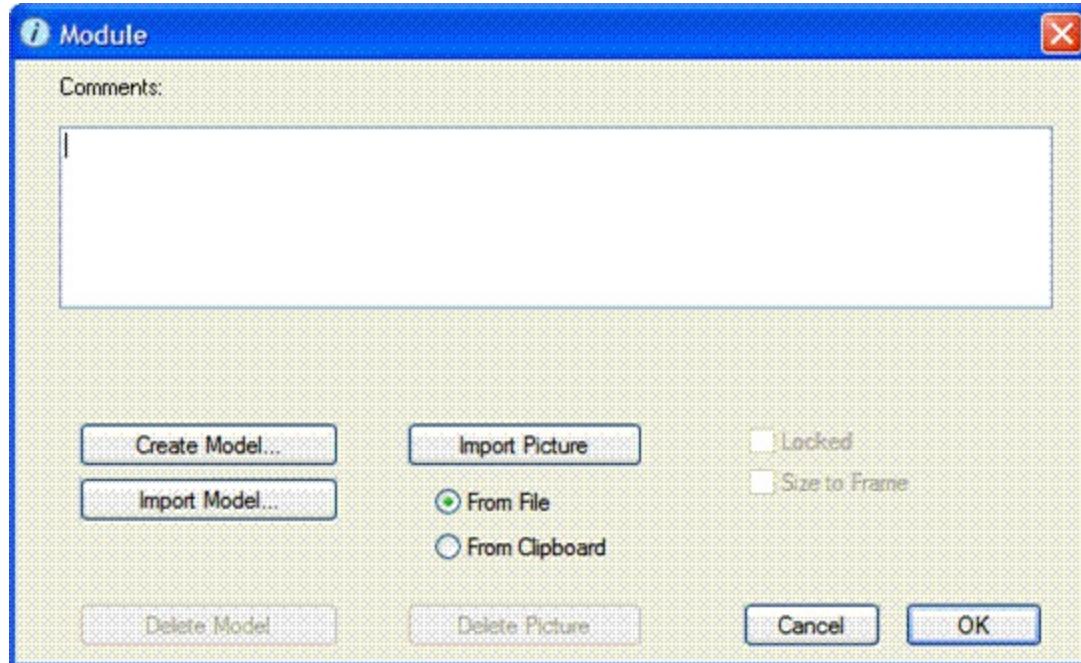
9. Optionally, add a comment, import a picture, or lock the module. For more information, see [Specifying Module Properties](#).
10. Click **OK**.
11. On the Map or Model layer, double-click the new module icon. The Map or Model layer appears for the new module.
12. Build the new module as you would any model.
13. Save your changes as you work by choosing **Save** from the File menu. This saves all changes you made to the top-level model and any modules in the model.
14. Repeat this procedure to add as many modules as you need to the model. You can add modules to the top-level model, or to any other modules within the model.

When you having finished creating the module, [define the module inputs and outputs for the module](#), and then [assign the module inputs](#). The model, as a whole, cannot run until all module inputs have been assigned (each module, though, can run in isolation). For more information, see [Running Modules](#).

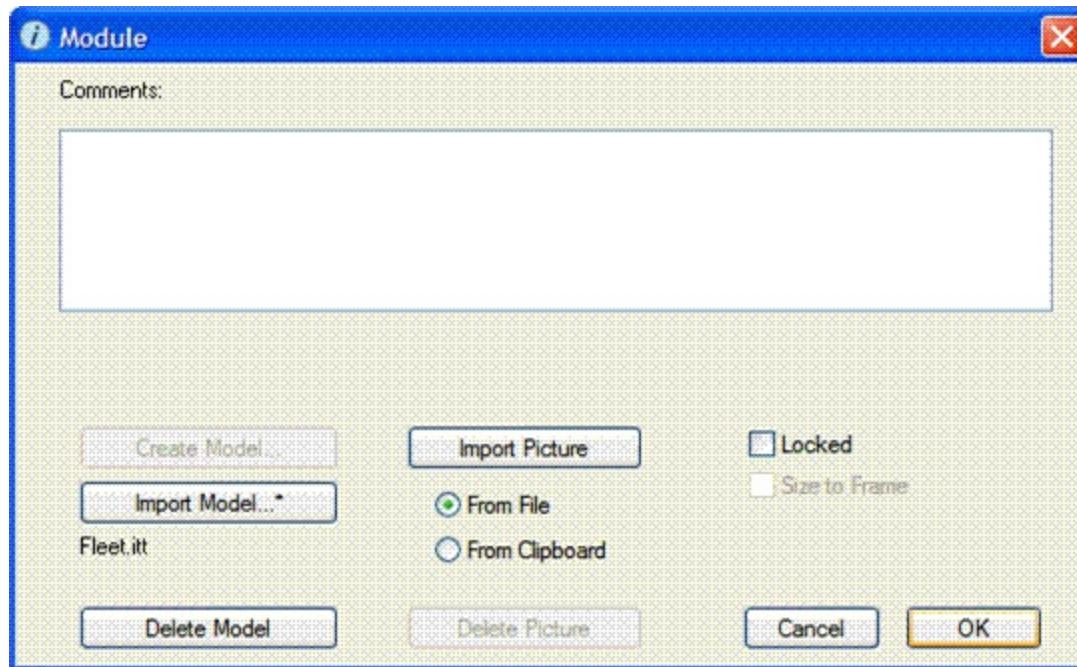
# Importing Models as Modules

Use the following procedure to include any existing model as a module within another model.

1. If you have not already done so, add a module entity to your model by clicking the  button on the Map or Model toolbar and then clicking the diagram surface to place the module entity on the layer.
2. On the Model layer, double-click the module entity. The Module dialog box opens.



3. Click the **Import Model** button. The Open File dialog box opens.
4. Navigate to and then select the model that you want to import.
5. Click **Open**. The name of the module appears beneath the **Import Model** button.



6. Optionally, add a comment, import a picture, or lock the module. For more information, see [Specifying Module Properties](#).
7. Click **OK**.

You can now navigate into the module by double-clicking the module icon.

 [Related Topics](#)

# Saving Changes to Modules

When you save a model that contains modules, all changes made to the entire model (including changes made to modules connected to it) are saved. Each module is saved in its own file, in a subdirectory named "Modules" within the directory where the model is saved. Modules are saved with an .ITT file extension.

You can save a copy of a model that contains modules by using the **Save As** command. When you save a model with a new name, all information defined for the model (including modules connected to it) are saved under the new name.

---

*Note:* The **Save As** command allows you to specify a new name for the top-level model only. All connected modules keep their original names. Therefore, to make a copy of the entire model (including connected modules), you must save the model to a different directory than the original model's directory; otherwise, the new top-level model will be connected to the same modules as are connected to the original model.

## Saving changes to models that contain modules

- From the File menu, choose **Save**. All changes made anywhere in the model, including changes made to modules, are saved.

## Saving a copy of a model that contains modules

1. From the File menu, choose **Save As**. The Save model files as dialog box opens.
2. Navigate to the location where you want to save the new model file.
3. In the File name box, type a name for the new model file.
4. In the Save as type box, select **iThink document (\*.ITM)** or **STELLA document (\*.STM)**.
5. Click **Save**. A copy of the model is saved in the new location under the new name.

## Saving modules as models

1. From the File menu, choose **Open**, and then select the module file (.ITT) that you want to save as a model. A message appears that says that the template model will be converted to a model file.
2. Click **OK**.
3. From the File menu, choose **Save**. The Save model file as dialog box opens.
4. Navigate to the location where you want to save the new model file.
5. In the File name box, type a name for the model file.
6. Click **Save**. The module is saved as a model. If the module contained other modules, the connections to the submodules are retained.

 [Related Topics](#)

# Specifying Module Properties

A module's properties include the name of the model file that contains the module's contents, the picture (if any) that appears on the module icon in the model, and any comments you want to include about the module.

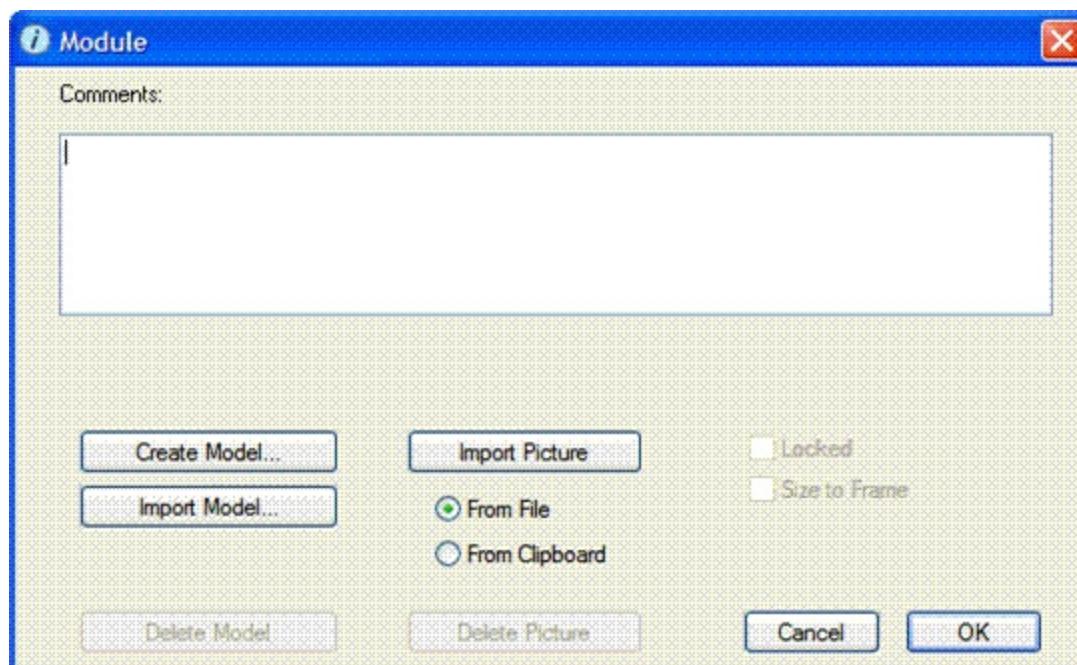
When you [create a module](#), you can specify the properties for the module. You can also place a module icon on the diagram surface and specify the properties later (you cannot navigate into the module until you have assigned a model to the module). You can change the properties for a module at any time.

# To specify module properties

1. Select the module icon and then, from the Model menu, choose **Open Selection**.

*Tip:* If the module does not yet have an assigned model, you can also open the Module dialog box by double-clicking the module icon. If the module has an assigned model, you can also open the Module dialog box by right-clicking the icon and then choosing **Open**, or by using **CTRL+double-click** (Windows) or **Command-double-click** (Mac).

The Module dialog box opens.



2. Optionally, include a comment for the module by typing the comment in the Comments box. Comments are for informational purposes only and do not appear anywhere but in this dialog box.
3. If necessary, assign a model to the module or unassign the currently assigned model:
  - **To assign a model to the module by creating a new model**, click **Create Model**, and then specify a name for the new model. For more information, see "[Creating a module from within a model](#)" (in [Creating a module from within a model](#)).
  - **To assign a model to the module by importing an existing model**, click **Import Model**, and then and specify the model that you want to import as a module. For more information, see [Importing Models as Modules](#).
  - **To unassign the currently assigned model** for the module, click the **Delete Model** button.

4. Optionally, add or remove a picture for the module's icon:

- **To include a picture on the module's icon**, first choose the source of the image file (**From File** or **From Clipboard**) and then click **Import Picture**. If you select **From File**, you are prompted to specify the image file you want to use. If you select **From Clipboard**, the image currently copied to the clipboard will be used. If there was an existing picture on the model's icon, the picture is replaced by the new picture you specify.
  - **To scale the imported picture** so that it fits within the module icon, select the **Size to frame** check box.
  - **To delete the selected picture** from the module's icon, click **Delete Picture**.
5. To lock the module, select the **Locked** check box. Enter a password to access the locked module and click **OK**.
6. When you are finished specifying the module's properties, click **OK**.

 [Related Topics](#)

# Navigating Through Modules

Each module appears on its own level of a model. When you view the top level of a hierarchy, you see the Interface, Map, Model, and Equation layers for the model as a whole, with icons for modules indicated on the Map and Model layers.

When you navigate into a module within the hierarchy, the Map and Model layers show you only the information contained within the module (including icons for any other modules that are contained within the module).

The Equation layer always shows information for the current level and for all modules contained inside or below the current level. On the Equation layer, you can view equations sorted by module by selecting the **Module** option in the [Equation Preferences dialog box](#).

---

*Note:* The Interface layer is not available when you are viewing a module. Only the top-level model in a hierarchical model has an Interface layer. If you create a module by [importing a model](#) that has an interface, the interface is not visible when you view the module, but the interface is visible if you open the module as a stand-alone model.

# To navigate down one level in the hierarchy

- On the Map or Model layer, double-click the module icon  .  
The module appears on the module's Map or Model layer. The module's name appears in square brackets after the model's name in the window's title bar.

Module 1



---

**Note:** You cannot navigate into a module until you have [created](#) or [imported](#) a model for it.

---

## To navigate up one level in a hierarchy

- On the left side of the window, click the  button. The next higher level in the hierarchy appears.

---

*Tip:* You can also use CTRL+double-click (Windows) or Command-double-click (Mac) on the diagram's surface to navigate up one level.

---

## To navigate to the top level of a hierarchy

- On the left side of the window, click the  button. The top level of the model hierarchy appears.

 [Related Topics](#)

# Running Modules

You can run modules in three ways:

- By running the model that contains the module (with the **Run** command). When you run a model, all of its modules automatically run.
- By choosing to run selected modules within the model (with the **Module Specs** command). When you run a module as part of the model that contains it, the module uses the model's run specifications (as defined in the [Run Specs dialog box](#)).
- By opening the module (.ITT) file and running the module in isolation (for example, in order to test it or use it as a stand-alone model). When you run a module in isolation, the module uses its own run specifications (as defined in the [Run Specs dialog box](#)). When you run one or more selected modules (without running the entire model), module inputs are held constant using the same rules as those used by the **Run Selected Sectors** option.

---

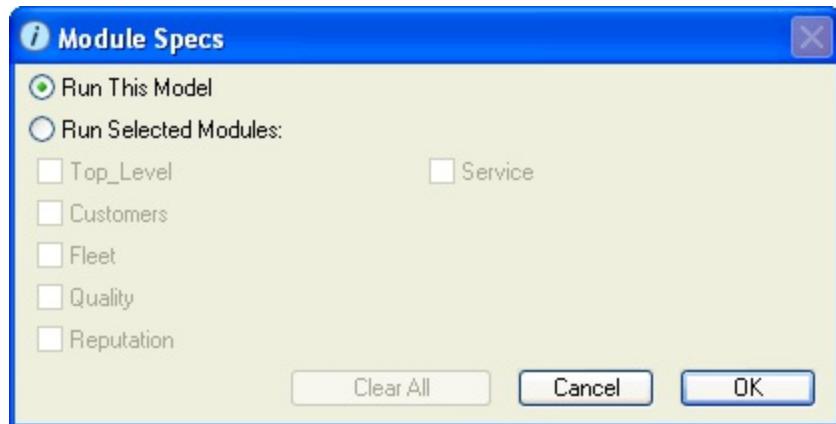
**Note:** A model that contains modules will not run until all inputs in each of its modules have been assigned to variables in the model. If you receive an error message when you try to run the model, check all connections for the modules and make sure that all inputs are assigned and that there are no unused connectors. For more information, see [Testing and Troubleshooting Modules](#).

---

The following procedure describes how to run selected modules within a model.

# To run one or more modules in a model

1. From the Run menu, choose **Module Specs**. The Module Specs dialog box opens.



2. Select the modules that you want to run:

- To run the entire model, including all modules, select **Run This Model**.
- To run one or more modules within the model, select **Run Selected Modules**, and then select the check boxes for the modules that you want to run.

---

Note: Modules whose names are grayed out cannot run because they are not fully defined (do not have all variables defined) or have errors.

---

3. To clear your selections so that you can make another choice, click the **Clear All** button.
4. When you have finished making your selections, click **OK**.
5. Run the selected modules:
  - If you selected **Run This Model**, choose **Run** from the Run menu.
  - If you selected **Run Selected Modules**, choose **Run Module(s)** from the Run menu.

[Related Topics](#)

# Storytelling with Modules

You can use the [Storytelling feature](#) to walk through a hierarchy, revealing only the entities and modules that you want to show, in the order in which you want to show them. When building stories with modules, it is a good idea to create a new chapter in the story each time you navigate from one module into another in the story. If you create a new chapter, you must delete all entities from the chapter before building it (as described below).

# To build a story that includes modules

1. On the Interface layer of the model that contains modules, add the Storytelling button and open the Button dialog box. For more information, see [Storytelling button](#).
2. In the Create Story dialog box, click the **Build Story** button. The model appears in the Storytelling tab.
3. In the model, navigate to the level where you want to begin the story (if the story does not begin on the top level of the model):
  - To navigate down a level, double-click the module's icon.
  - To navigate up a level, CTRL+double-click (Windows) or Command-double-click (Mac) the diagram surface.
4. Build the story by including entities on the current level as you normally would.
5. When you want to move to a different module, click  above the navigation tabs to return to the Create Story dialog box.
6. If you are navigating from one module into another and want to create a new chapter, click the **New** chapter button. Otherwise, skip to step 8.



The number under the **New** button changes to the next sequential number ("2", if this is the first chapter you've created in the story). The "Story Sequence" box displays a copy of the list of entities included in the previous chapter of the story.

7. Select all of the entities displayed in the "Story Sequence" list and then click **Delete Selection** to delete them.
8. Repeat steps 2-7 to continue composing the story.
9. When you are finished building the story, click **OK** to close the Create Story dialog box.

# Introduction to Using Modules

One way is to use modules is to start at the top by determining the connections between modules and then creating the contents of each module. This method is called "developing the dynamic hypothesis". With this method, you first look at the connections between modules, envisioning the overall flow of information from one module to the other. Only after you've made a plan for how each module is connected to the other do you start building the contents of each module. The [airline module example](#) shows a module that is built this way.

Another way is to use modules is to use (or re-use) an already-built structure in your model, or to hide the details of a complex process. With this method, you are using the module to perform a function in your model. You may not have built the module yourself, and you may not even be able to see the details of the module if it is locked, but you have access to the module by providing input values to the module and retrieving output values generated from the module.

## What do you want to do?

[Use modules to communicate between modules](#)

[Use modules to re-use a common structure or hide a complex process](#)

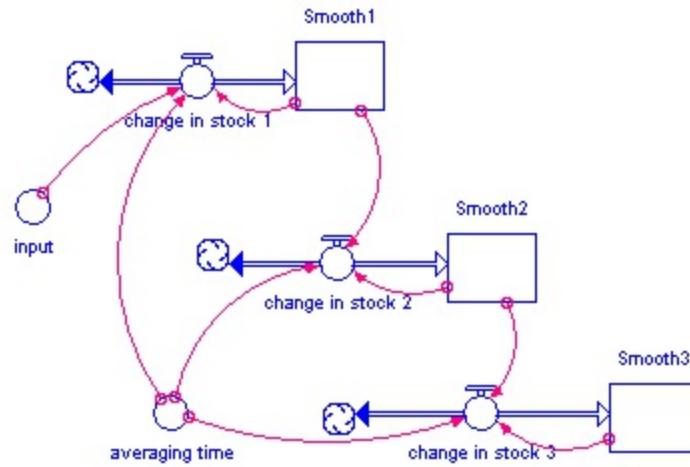
 [Related Topics](#)

# Using Modules to Re-use a Common Structure

The following procedure describes the overall process for using an already-built structure in your model to provide input values to the model. For more information about each step, click the link provided in each step.

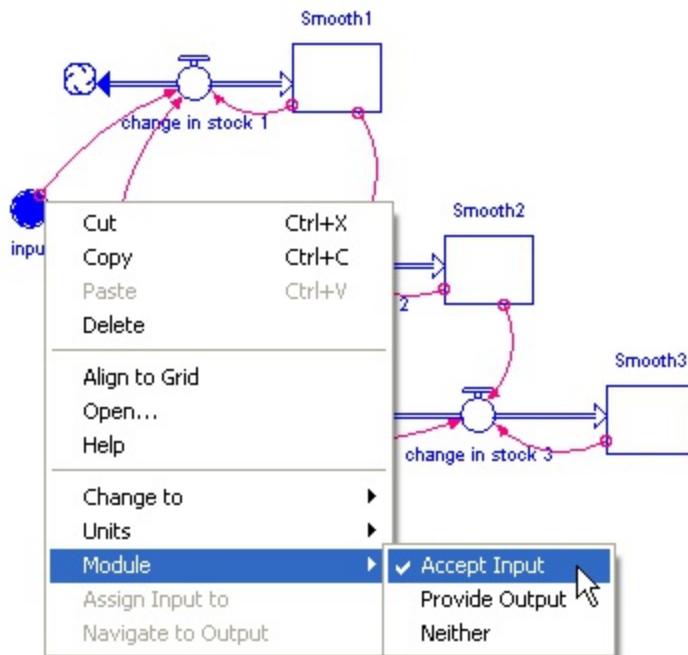
1. Create and test a model for the process that you want to be able to re-use in other models.

The [Smooth3 model](#) is an example of a process that you might want to build once to use as a part of other models.

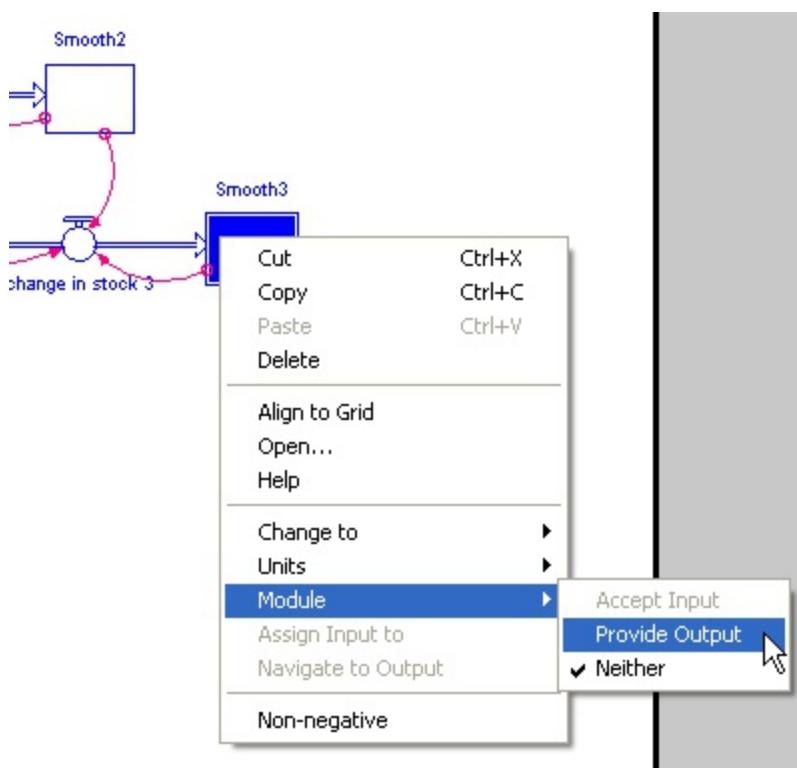


2. [Define the inputs and outputs](#) of the model.

In the Smooth3 example, there are two inputs: the converter named "input" (shown below) and the converter named "averaging time".

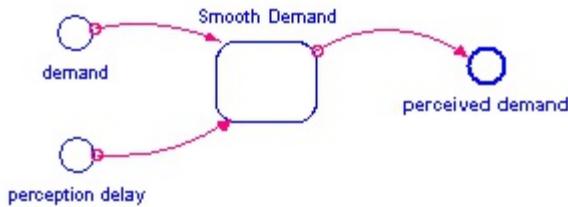


The output is the stock named "Smooth3":



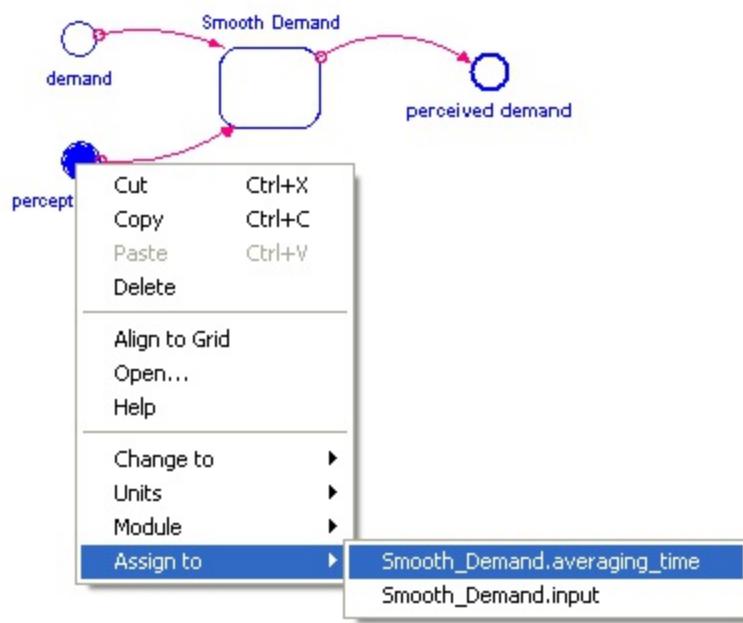
3. When you are finished creating the process model, save it.
4. Create or open the model that will use the process model.
5. Use the  building block to add a module to the model.
6. [Import the process model as a module](#) by double-clicking the module icon on the Model layer and then clicking the **Import Model** button in the Module dialog box.
7. Use connectors to [create connections](#) between the model entities and the module.

In the Smooth3 example, the module is defined as having two inputs and one output, so we could draw the connections this way, with "demand" and "perception delay" providing the input value to Smooth3, and "perceived demand" accepting the returned output value from Smooth3:

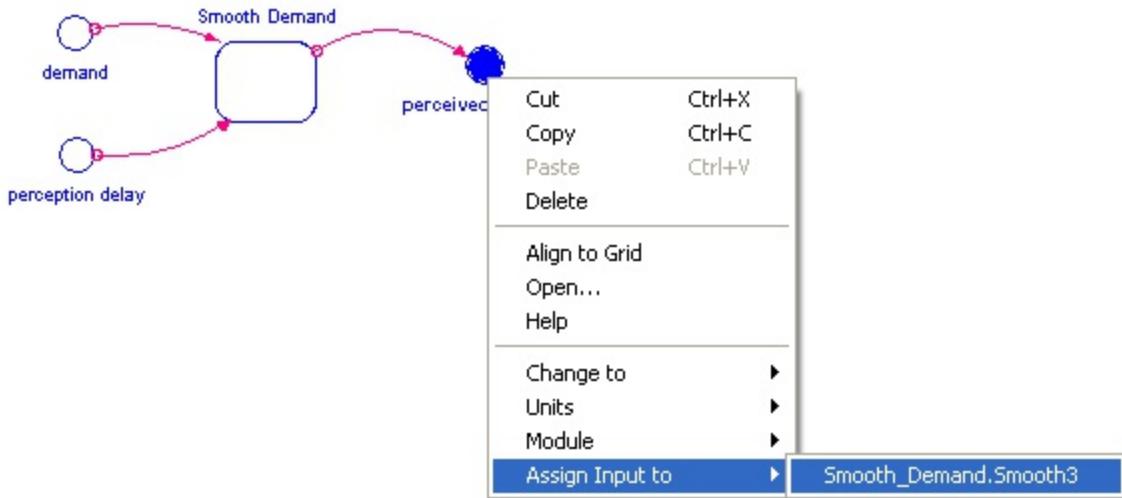


8. [Assign the parameters and return values](#) in the model to the defined module inputs and module outputs in the module.

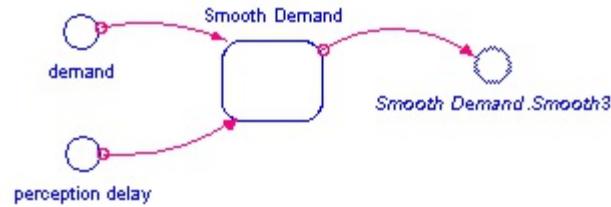
In the Smooth3 example, the two module inputs ("input" and "averaging time") are the variables available to assign to the "demand" and "perception delay" variables in the model.



Similarly, the module output "Smooth3" is the only variable available to be assigned to "perceived demand".



This is what the model looks like with all variables assigned:



# Using Modules to Communicate Between Modules

The following procedure describes the overall process for building a hierarchical module by first determining the connections between modules and then creating or importing the modules. For more information about each step, click the link provided in each step.

1. Click  on the Map or Model toolbar and then click the diagram surface to place module icons.



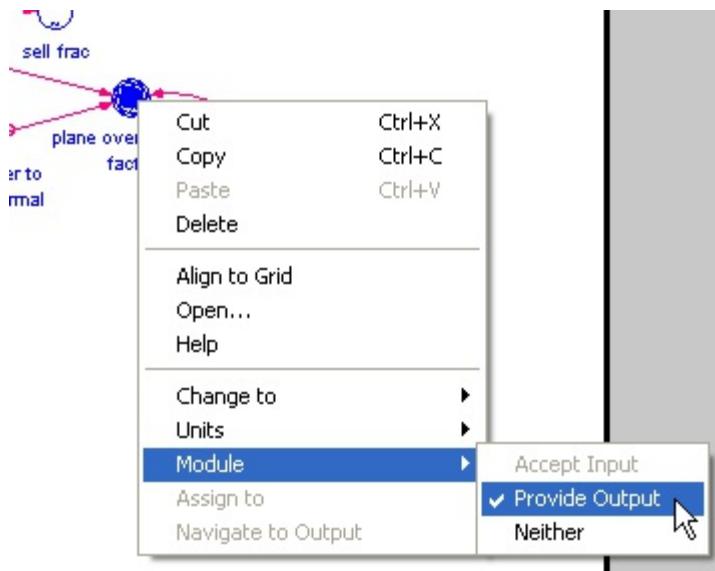
2. Use connectors to [create connections](#) between the module icons.

In the following example, connections indicate that both modules supply and receive variable values from each other.



3. Create and test the contents of each module. You can do this by [building the module](#) directly within the model that contains it, or by creating a separate model.
4. In each module (or separate model to be used as a module), [define the module outputs](#).

In the following example, the "plane overcrowding factor" in the Fleet module is defined as an output.

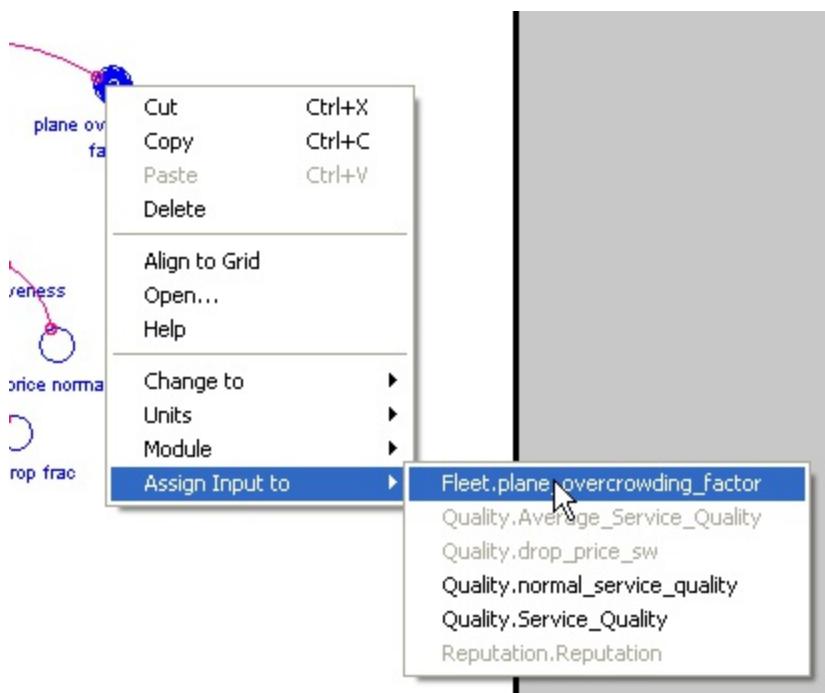


5. If you created a separate model to be used as module, [import it into the](#)

[module.](#)

## 6. [Assign the module inputs for each module.](#)

In the following example, the "plane\_overcrowding\_factor" variable from the Fleet module is assigned as an input to the "plane\_overcrowding\_factor" variable in the Customers module.



 [Related Topics](#)

# Command Line Options

On Windows machines, models can be run from outside of **STELLA** or **iThink** using command line controls. For example, to open the model "test.itm", run it and then quit you would use the following command:

```
"iThink.exe" -r test.itm
```

The following describes the command line options that are supported:

- r** Run the model ballistically and then quit (don't save)
- rn N** Run the model ballistically N times and then quit (don't save)  
[N > 0]
- rs** Run sensitivity analysis as defined in the model and then quit (don't save)
- rd** Restore all devices after opening (before the first run)
- s** Save the model at the end of the last run
- i** Import Now before each run
- x** Export Now after each run
- nq** Do not quit after the last run
- 0 var** Force the specified variable to zero after opening (before the first run)
- 1 var** Force the specified variable to one after opening (before the first run)

## Example:

To use the command line options to run a model more than 32,000 time steps, follow these steps:

1. Add a persistent link to export the values of the model to a spreadsheet.
2. Connect the last values exported to a persistent import sheet.
3. Save the model.
4. Create a shortcut to **STELLA** or **iThink** using the -rn N option.

For example:

"iThink 9.1.exe" -rn 3 model.itm

This will run model.itm 3 times, picking up the last run's values for the next run, and then quit. If the time specs for the model are set to 32,000, it will run a total for 96,000 time steps and the final results will be in the Excel spreadsheet.

To run comparative runs for different scenarios, use the -0 and -1 options to set the proper configuration switches, and the -s option to save after the run.

NOTE: You do not have to quit the software in order to change configuration settings, however, quitting is the default behavior.



# Part 3 - Technical Appendices

This portion of the Technical Documentation provides technical information regarding the software.

- **DT: What, Why, & Wherefore** discusses the time interval between calculations in the software. You'll learn more about what DT is, and how it is used. You'll be provided with a set of practical guidelines for choosing an appropriate value for DT in your models.
- **Simulation Algorithms** discusses the simulation algorithms employed by the software. You'll learn how the software does its internal calculations under the Euler, 2nd-order Runge-Kutta, and 4th-order Runge-Kutta integration methods.
- **Implicit Stock/Flow Rules of Grammar** looks at the internal rules of grammar used by the software. You'll learn about topics such as prioritization rules for non-negative stocks and allowable stock-flow connections.
- **Creating Spatial Maps** explains how to map array data in your model to color values so that you can view your data as three-dimensional, spatial information.
- **Using isee NetSim** describes how to publish your models to Internet or any Web server running the isee NetSim Web Service.
- **Tips, Tricks and Troubleshooting** gives you answers to frequently asked questions and a range of techniques that will help you optimize performance and troubleshoot problems in your models.

This part of the Technical Documentation is designed to be used as reference material. You may want to skim the topics in the part now. Then, when you need an in-depth technical treatment of one of the topics covered, take some time to work through the appropriate section.

# Introduction to DT

How can something so small be such a big problem for so many people? Yes, two little letters ... D ... T. Enough to bring grown men and women to their knees, swooning in fits of frustration and despair. But now, right here, we're going to fix all this. This section will teach you almost everything you ever wanted to know about that deceptively tricky little rascal known only as ... "DT" (though Delta Time is the official name on the birth certificate).

 [Related Topics](#)

# What is DT?

Many have asked, few have understood. DT, or dt (depending on your level of disdain), is the interval of time between calculations. DT is expressed in whatever time unit you've chosen for your model. Therefore, DT answers the question: Is my model having its numerical values re-calculated once every time period, twice, three times...?

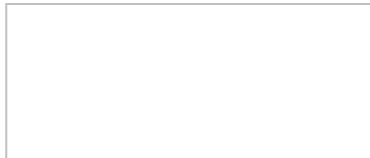
If you do not make a conscious choice of time unit for your model, the default choice will be "Months" (**iThink**) or "Time" (**STELLA**). To see this, visit the Run Specs... dialog located under the Run menu. Your choice of time unit provides the denominator of the units-of-measure for all of the flows in your model. For example, if you have flows of widgets, people, and dollars (and you are using the default time unit of "Months"), then the units-of-measure for your flows will be widgets/month, people/month, and \$/month. If DT in this model is 1.0, then a round of calculations will be performed once each month. If DT is 0.25, then a round of calculations would be performed every 1/4 of a month (or, four rounds of calculations would be performed per month). And, so on.

 [Related Topics](#)

# How does DT work?

Let's illustrate a few simple calculations so you can see how DT works in the calculation process. Consider the simple model, whose diagram is shown in Figure 12-1.

Figure 12-1 Simple Cash Model

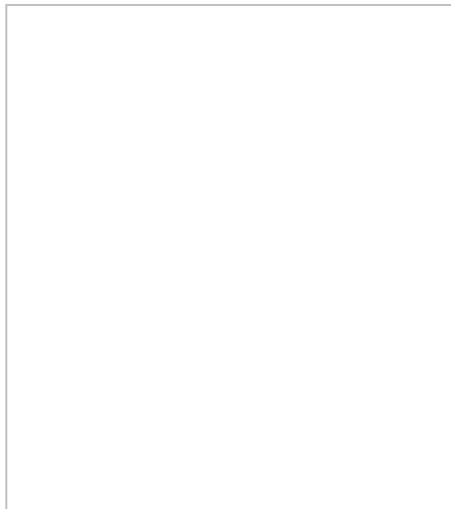


In this model, Cash is initialized at \$0. *income* is set at a constant of \$100/month. Reproduced in Figure 12-2 is a Table of numerical values generated by the model. In the first run, DT is set at its default value of 0.25. In the second, DT is set at 1.0. In each case, numerical values are printed every DT, stocks are reported as beginning balances, flows are reported as summed, and From Time has been set to 0. Three months' worth of data appear in the Table.

Figure 12-2  
Comparison of Cash Model Results with Differing Values for DT

**DT = 0.25**

**DT = 1.0**



Beginning balances reporting of stocks, summed reporting of flows

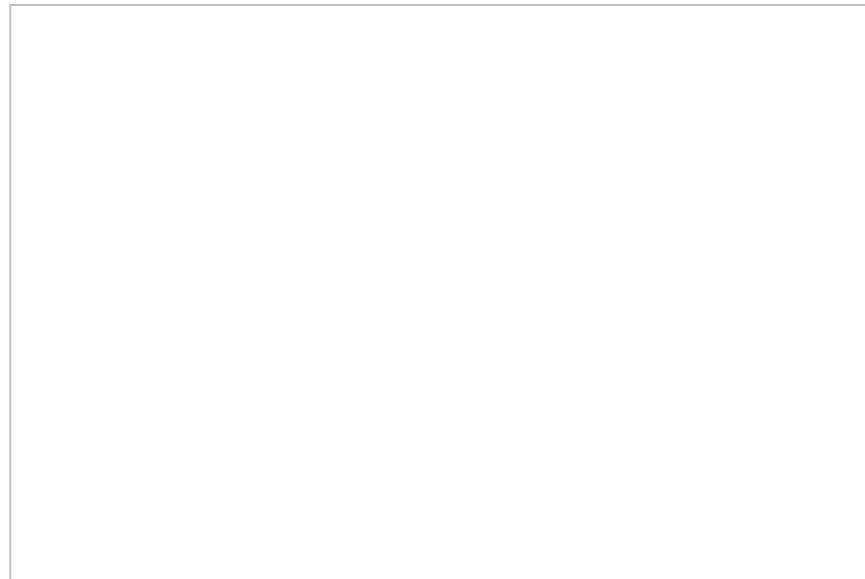
In both runs, by Month 3, Cash totals \$300. The only difference is the path followed in the two runs. In one case, \$25 was added 12 times. In the other, \$100 was added 3 times. In the first run, DT simply breaks the constant monthly flow of income into four equal chunks. Why would anyone want to do this?

As you'll soon see, there are a number of good reasons for breaking a time unit's worth of flow volume into smaller chunks. For now, a good first reason is that the division makes it possible to capture changes which occur over an interval of time less than the time unit you've picked for your model. When we picked a DT of 0.25, we would have been able to capture changes that occurred over an interval as short as 1/4 of a month (or roughly one week). So, DT represents the smallest time interval over which a change can occur.

DT thus determines how "chunkily" change will unfold in your model. A larger DT means chunkier changes in variables within your model. A smaller DT means smoother, more continuous changes. Imagine creating a cartoon animation by riffling through a deck of static pictures. If you have a large deck of pictures, in which the change is small from one picture to the next, the animation will unfold smoothly as you riffle through the deck. By contrast, if you have a deck with only a few pictures, in which each picture represents a fairly big change relative to its predecessor, riffling through the deck will produce a jerky animation. DT works the same way with the numerical values in your model.

To illustrate the notion of change occurring within a unit of time, let's modify our little Cash model to allow income to vary over a month. To implement variation, we'll convert income to a graphical function (using the discontinuous line segment option). The graphical function is reproduced in Figure 12-3. The input to the graphical function is Months (running from 0 to 1.0 in increments of 0.25). The output of the graphical function is income, which now declines over the month.

*Figure 12-3  
Income as a Time-Dependent Graphical Function*



We'll make two runs of the modified model. In the first, DT will be set to 0.25.

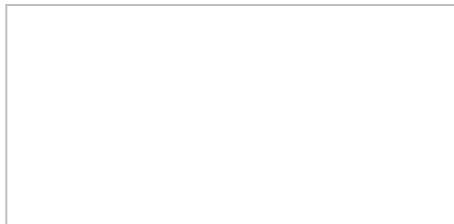
In the second, we'll set DT at 1.0. Both runs will be one month in duration. Before looking at the results, do you think the value of Cash will be the same at the end of the two runs? If not, in which run do you think Cash will achieve the greater value?

The results from the two runs are presented in Figure 12-4. Surprised?

*Figure 12-4*

*Modified Cash Model: Comparison of Results with Differing DTs*

**DT = 0.25**



**DT = 1.0**



Beginning balances reporting of stocks, summed reporting of flows

With DT set at 0.25, the value of income varies from DT to DT. Four calculations are made, each yielding a different value for income (each value is simply the value of income taken from the graphical function, and then multiplied by DT). Cash climbs to \$85 by the end of the run. With DT set at 1.0, only one calculation is made. And hence, only one value for income can be used. Given the way the graphical function works, that value is the value during Month 0. Cash thus grows to \$100.

In summary, values of DT less than 1.0 enable your model to capture changes that occur within the course of a time unit. By adjusting DT, you can cause the changes in the variables in your model to unfold quite jerkily or very smoothly. The choice is up to you. The trade-off, for the most part, is speed against smoothness and numerical precision. As DT gets smaller, changes get smoother and results become more precise numerically. However, as DT gets smaller, more calculations will need to be made, and it will take longer to complete a run. As DT gets larger, results will be choppier and less precise numerically. But, a larger DT leads to fewer calculations and therefore faster simulation speed.

# Some Practical Considerations

All of this said, under normal circumstances, there's a small range of values for DT which virtually always yields acceptable results (both smoothness and precision-wise), but which also does not cause your model to become unduly bogged down in calculations (with two exceptions which will be discussed below). This range is from 0.0625 to 1.0. DTs smaller than 0.0625 are rarely justified (though we'll indicate how it could be that even smaller values would be needed in the "[1/2 test](#)" below). The default value for DT (0.25) works very well in the majority of the models that we've seen or created. So, for the most part, you need not even worry about fiddling with DT.

However, if you do fiddle, we recommend you do so with numbers taken from the sequence  $(1/2)^n$ . Because of the binary arithmetic that the computer uses, numbers not taken from this sequence can lead to nasty round-off errors. The  $(1/2)^n$  sequence produces the following "nice" numbers: 1.0, 0.5, 0.25, 0.125, 0.0625 ...

## 2 Exceptions to the $(1/2)^n$ Sequence

As mentioned above, there are two exceptions to choosing DT from this "nice" set of values. The first happens when it is important to report values that do not fall on the  $(1/2)^n$  of a time unit. A typical example is when your model runs in years, and you want to report monthly values. In cases such as this, simply choose the DT as fraction option in the Time Specs... dialog and enter the denominator desired for your DT.

The second exception to the DT settings recommended above occurs when the length of the simulation is very long in the denominated units of time, and change unfolds slowly relative to a time unit (e.g., the movement of a glacier over a few thousand years). In these cases, it is appropriate to choose a DT larger than 1.0. Any integer value up to one million is allowed by the software to define a DT larger than 1.0.

## The 1/2 Test

A good first approximation for choosing DT is to choose a step size one-half of the shortest time delay in your model. Many time delays may be apparent, such as average days payable, a shipping lead time, a conveyor transit time, or a training delay. If you can determine the shortest time delay, you've got a good starting point for selecting the step size.

As a check on the value of DT that you've picked, it's always a good idea to execute the "1/2 test." Re-run your model with DT set equal to 1/2 of the value that was used to produce your results. Compare the results generated by the model to those generated before the value of DT was halved. Look at the results both in terms of numerical differences as well as any qualitative differences in the pattern of behavior that the model might be exhibiting. For example, has an oscillation disappeared? Is growth now occurring much more rapidly? Did the value of some stock which used to go negative cease going negative? If the answer to any of these type of questions is "Yes," you should take half the value of DT again and repeat the test. Continue in this fashion until your results remain essentially the same for two runs. Then, set DT equal to the highest of the last two values.

## Brief Re-Cap

To re-cap, DT is the amount of time between calculations. No change in the numerical value of any element in your model can occur in a unit of time smaller than DT. For most models, a DT of 0.25 works well. With DT set at this value, the running of the model is not unduly slowed down (due to too many calculations), and yet numerical precision and smoothness in most cases are ensured.

 [Related Topics](#)

# DT Situations Requiring Special Care

Two types of situations require special care in choosing the value for DT. The first situation concerns the creation of "artifactual delays" - delays which are artifacts of the way the software makes its calculations, rather than real delays resulting from the interplay of relationships in the system that you're modeling. The second situation results in "artifactual dynamics" - dynamic behavior patterns that are artifacts of the calculation process, rather than the result of relationships in the real system. If you understand these two types of situations, and you also make a practice of running the "[1/2 test](#)," you should have minimal difficulties with DT.

# Artifactual Delays

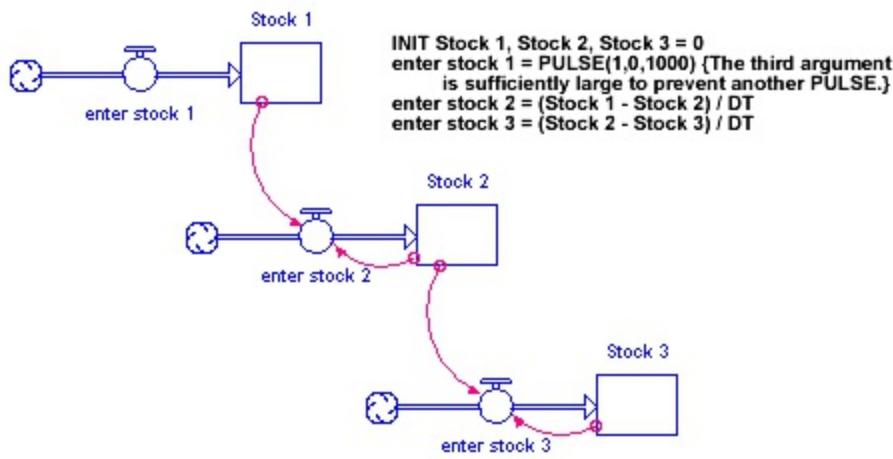
Artifactual delays result from the fundamental conceptual distinction, incorporated into the software, between stocks and flows. Stocks exist at a point in time. Flows exist between points in time. As such, the two cannot co-exist at the same instant in time.

Look at the first table of numbers in the "Comparison of Cash Module Results with Differing Values for DT" example (see [Figure 12-2](#) in [Figure 12-2 Comparison of Cash Model Results with Differing Values for DT](#)). Notice that Cash is 0 at TIME = 0. The first value of income that appears in the table is \$25. It appears at TIME = 0. But it really exists between TIME = 0 and TIME = 0.25! This is why Cash does not equal \$25 until TIME = 0.25. It takes one DT to integrate (in a mathematical sense) the value of the flow to yield a new value for the stock. There is no way to get around this integration delay. And, conceptually, it's real. It reflects the inevitable delay that exists between the time a decision to take some action is made, and the time it takes for the resulting action to make itself felt in one or more accumulations. For example, if you decide to eat that cookie beckoning to you from the plate, it will take an "instant" before you can seize it. You cannot decide to seize and seize in exactly the same instant. This "instant" is the software's artifactual delay.

As long as DT is defined as a relatively small value, artifactual delays usually are insignificant. It's usually only when DT becomes too large that such delays become a problem. We say "usually" because, in some cases (even with a small DT), small integration delays can "add up." In these cases, lots of little delays make one big delay. And, that big delay can distort the model's dynamics. Let's look at a case in which this distortion occurs.

Suppose that we have three stocks arranged such that the first transmits information about its magnitude to the inflow which feeds the second. The second, in turn, transmits similar information to the inflow which feeds the third. A diagram of this arrangement appears in Figure 12-5.

*Figure 12-5 Simple Model Showing Stock-Flow-Stock Connections*



The inflow to *Stock 1*, *enter stock 1*, is defined with the [PULSE Builtin function](#). The PULSE fires off a 1.0 at TIME = 0. Once the 1.0 arrives in *Stock 1*, this becomes a signal to fire a 1.0 into *Stock 2*. Once Stock 2 is "loaded," it triggers a 1.0 to flow through *enter stock 3*. We'll simulate this system twice, once with DT = 0.25 and once with DT = 1.0. When we do, we get the results shown in Figure 12-6.

*Figure 12-6*  
*Artifactual Delays Increase with Larger DT*

**DT = 0.25**

**DT = 1.0**

Beginning balances reporting of stocks, summed reporting of flows

Tracing the initial pulse through the system, we see from the Table that with DT = 0.25, the pulse does not arrive in *Stock 3* until TIME = 0.75, three DT's after it is fired. The delay in arriving is due to the implicit one DT "perception

lag" built into any such stock-to-flow-to-stock connection. Since there are two such connections in this system, we should expect to wait two DT's for the message to complete its journey. The third DT lag is contributed by the integration delay between *enter stock 1* and *Stock 1*. These artifactual delays become a bit more of an issue when DT is 1.0 or larger. In the case of DT=1.0, a three DT lag means three months!

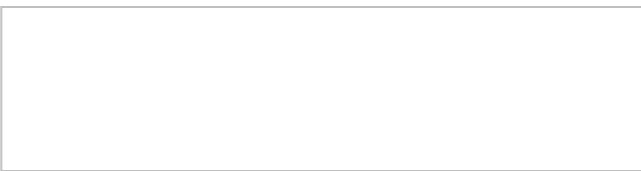
If you have sequences in which one stock is generating the input to a flow which feeds a second stock (and so on), you'll need to use care in selecting a value for DT. In these instances, you'll probably want to make DT less than 1.0 to avoid the cumulative effect of such perception lags.

## Artifactual Dynamics

The second area requiring special care in the choice of DT arises when your model generates a dynamic pattern of behavior caused by the way the software is performing its calculations. What happens in these cases is that because the value of DT is too big, the flow volumes remain too big, and "pathological" results are generated. In effect, the flow overwhelms the stock to which it's attached.

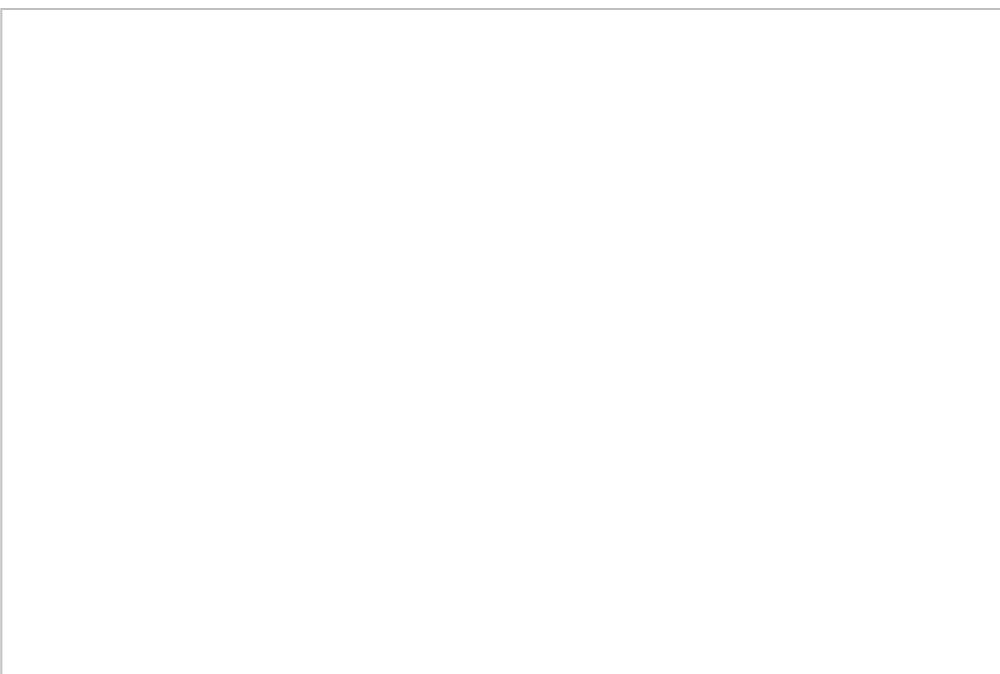
To illustrate how "overwhelming" can come about, let's return to the "Comparison of Cash Module Results with Differing Values for DT" example (see [Figure 12-2](#) in [Figure 12-2Comparison of Cash Model Results with Differing Values for DT](#)). In that example, we had a stock of Cash being fed by a flow of income. Adding an outflow yields the diagram in Figure 12-7.

*Figure 12-7  
Modified Cash Flow Model*



Cash will be drained by a flow of spending. Spending is defined as a graphical function which takes Cash as its input. The graphical function used to specify spending is shown in Figure 12-8.

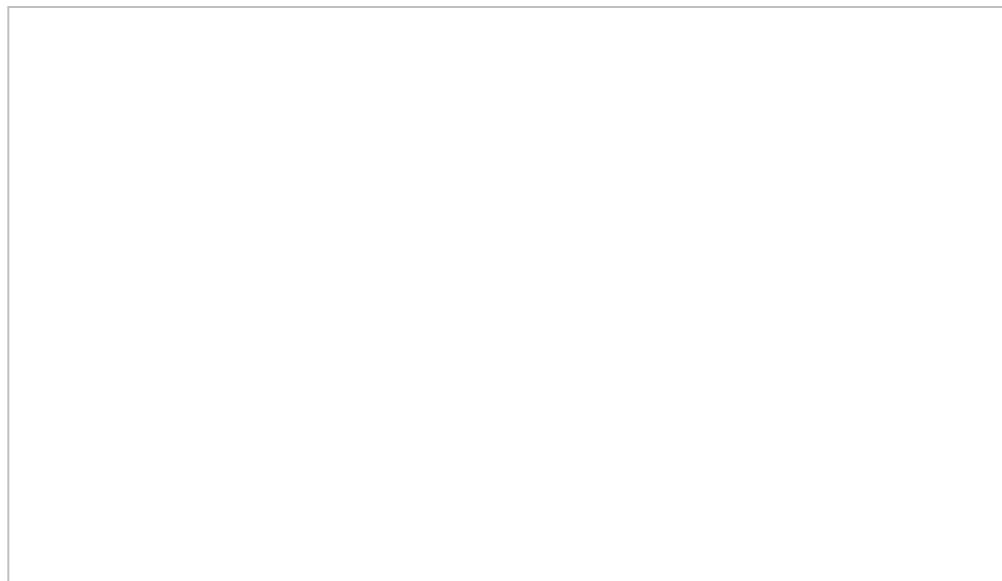
*Figure 12-8  
Spending as a Function of Cash*



The initial value of the Cash stock is \$100, and income remains constant at \$90/month. DT is set at 1.0.

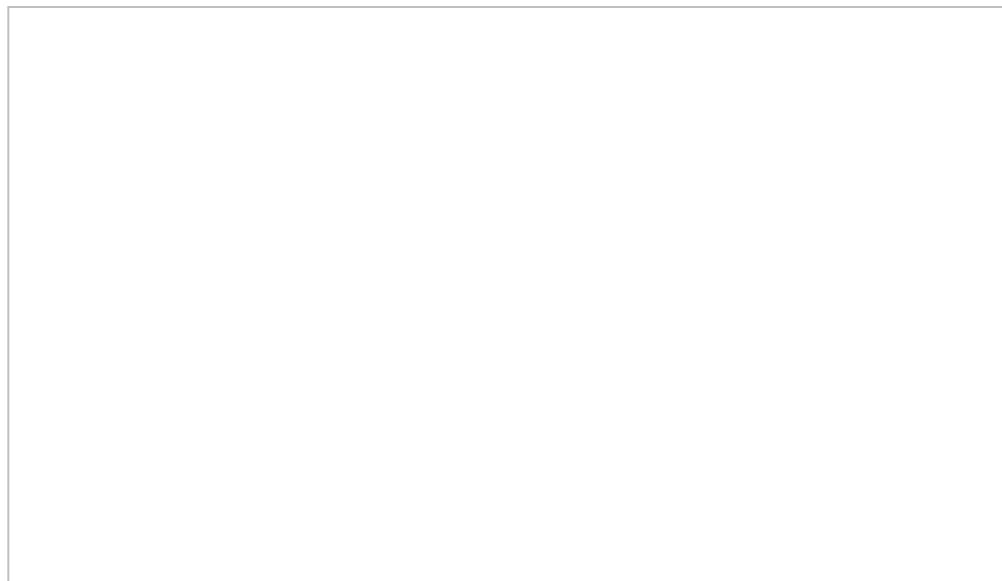
The graph in Figure 12-9 presents the results of a simulation of this model. The results indicate that Cash will drain down for awhile, and then "saw-tooth" about a steady-state value - which appears to be somewhere around \$12 to \$15. Spending remains constant at \$100 per month until Cash dips below \$20 (see the graphical function), then it too, begins to saw-tooth.

*Figure 12-9  
With Large DT, Pathological Results Ensue*



Whenever you see such a saw-tooth pattern, always run the "[1/2 test](#)." The results of this test appear in Figure 12-10.

*Figure 12-10  
A Smaller DT Removes the Pathology*



As you can see, only the barest hint of a saw-tooth pattern remains. This "hint" would completely disappear were we to cut DT in half once again (Try it!). The pattern would disappear because it is being generated as an artifact of a calculation problem, not by the interplay of the relationships in the system!

Just what is this calculation problem? As soon as Cash dips below \$20, spending begins to decline. What should happen is that as spending falls, Cash should begin to decline at a decreasing rate. This is because the rate at which Cash is declining is simply equal to the difference between income and spending. And, since spending is falling (and income is constant), then the difference between the two is getting smaller. As such, Cash should decline more and more slowly as spending gets closer and closer to income. Cash, therefore, should gradually approach a value where it will generate a value for spending that is exactly equal to income (i.e., \$90). This is essentially what happens when DT is cranked down to 0.5, and exactly what happens for values of DT less than 0.5. But it's not what happens when DT is 1.0.

When DT is "too big," the cut back in spending does not occur soon enough. In fact, spending remains too high for one DT. This causes Cash to drop below its steady-state value (i.e., the value at which spending will decline to just equal income). Because Cash falls below its steady-state value, it generates a value of spending that's less than the value of income. With income in excess of spending, Cash increases. But, once again, because DT is too big, the rebound carries Cash past its steady-state value. This means that spending is again in excess of income. So, once again Cash must fall. And, we're back into the cycle. Not until DT is reduced is Cash able to directly home in on its steady-state value without overshoot or undershoot. This is because with a smaller DT, the incremental changes in spending are smaller (since calculated flow values are multiplied by DT). As such, spending can gradually approach income, rather than "hopping over" it.

It may help you to see what's going on in the model to generate values for Cash, income and spending in a Table for different values of DT. It will also be useful to create a graph of Cash. For small values of DT, you will see it smoothly home right in on its steady-state value with no overshoot whatsoever.

Picking a value for DT that's too big can lead to all sorts of mayhem. This is particularly true when a Reservoir, with Non-negative turned off, has a bi-directional flow whose value is determined as some function of the Reservoir. What happens in this case is that when the biflow's outflow volume "overwhelms" the stock, the stock goes negative. Then, because the flow depends on the stock, it too goes negative. But a negative outflow is equivalent to an inflow! So, the flow reverses, putting stuff back into the

stock. This enables the outflow to be positive once more - re-initiating the cycle. Once again, the manifestation of the problem is a saw-tooth oscillation. In this case, the stock oscillates between a positive and negative value (sometimes the amplitude of the oscillation will expand).

The bottom line here is that if you see a saw tooth oscillation, or any oscillation for that matter, it's always a good idea to run the "1/2 test." Make sure that the oscillation is real, and not an artifact of your choice of a value for DT. Real oscillations will persist even with the tiniest values for DT.

# Summary

Practically speaking, if you keep DT at, or below, 0.25, it is unlikely that you'll ever encounter any of the problems discussed here. However, it is important that you be aware of what DT is doing in your models. This will keep you from being puzzled when strange behaviors appear in your model results. It also will equip you for correcting such strangeness if it does appear. If you encounter unexpected delays in your models, or you see a puzzling behavior pattern (particularly, a saw tooth oscillation), check DT. Cut it in half and re-run your model. If the results persist, try cutting it in half one more time. If the results continue to persist, it's very likely that the behavior you're seeing is real. If the behavior disappears, it was a calculation artifact.

You now should be DT savvy. Simulate on!

 [Related Topics](#)

# Introduction to Simulation Algorithms

As you run a model, the software uses standard numerical methods to solve the system of equations that comprise your model. The following sections describe the inner workings of the three numerical methods at your disposal:

- [Conceptual Overview](#) and [Technical Overview](#) provides an overview of how the software solves the equations in a model.
- [Choosing the Appropriate Algorithm and DT](#) gives guidelines for choosing an appropriate step size and integration method in your models.
- [Accuracy of the Simulation Algorithms](#) and [Euler's Method](#) and [The Runge-Kutta Methods](#) show how the three simulation algorithms (Euler's method, 2nd-order Runge-Kutta, and 4th-order Runge-Kutta) are applied to model equations.

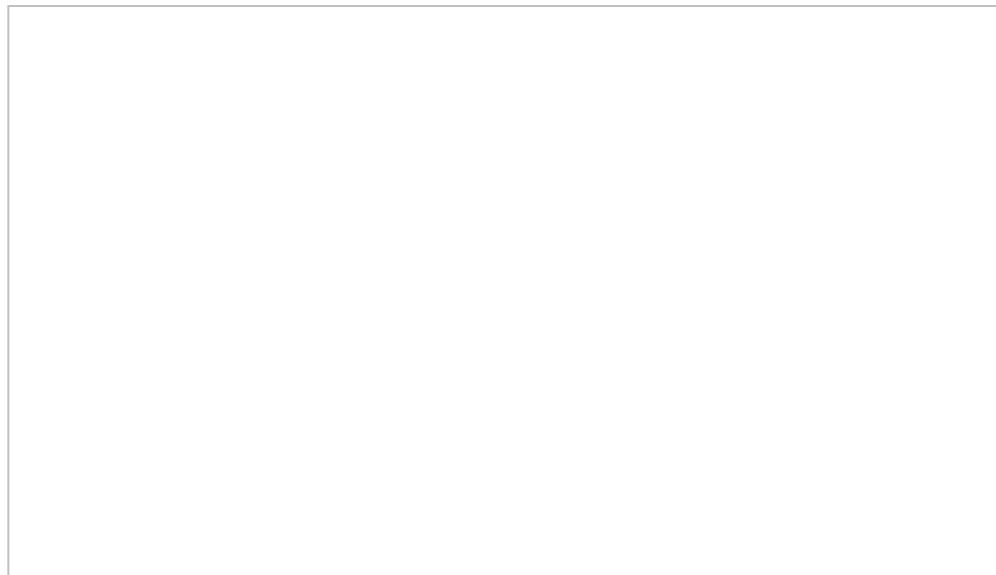
 [Related Topics](#)

# Simulation Algorithm Conceptual Overview

Consider the plot and model structure shown in Figure 13-1. The plot shows the idealized behavior of a real-world system - in this case, the temperature of a cooling cup of coffee. The model is idealized as well - if it was solved analytically, it would yield exactly the behavior exhibited by the plot.

When the computer is used to simulate systems such as this, it is not possible to get an exact solution. Instead, a set of discrete calculations (the only kind that digital computers are capable of producing) is used to approximate the idealized curve. The software divides the time axis into equally-spaced intervals, each with a width of DT (or "delta time"). Then, calculations are performed at discrete intervals, as indicated in [Figure 13-2](#).

*Figure 13-1 Idealized Behavior of Cooling Model*



*Figure 13-2  
Simulated Behavior of Cooling Model*

## Related Topics

# Simulation Algorithm Technical Overview

In the software, the equation structure underlying the model diagram is of vital importance. The equations created behind the scenes as you hook together stocks and flows are known as "Finite Difference Equations." The equations window of the cooling model, for example, would yield a set of equations that look like the ones below.

Temperature(t) = Temperature(t-dt) + (-Cooling)\*dt

INIT Temperature = 100

Cooling = Temperature \* Constant

Constant = 0.5

In a model, each stock equation (in this case, the equation for Temperature) is a finite difference equation. Conceptually, solving finite difference equations is straightforward. It involves a two step initialization phase, and a three step iterative evaluation phase:

## Initialization Phase

**Step 1.** Create a list of all equations in required order of evaluation.

**Step 2.** Calculate initial values for all stocks, flows and converters (in order of evaluation).

## Iteration Phase

**Step 1.** Estimate the change in stocks over the interval DT. Calculate new values for stocks based on this estimate.

**Step 2.** Use new values of stocks to calculate new values for flows and converters.

**Step 3.** Update simulation time by an increment of DT. Stop iterating when Time >= simulation To Time.

Step 1 of the iteration phase is a critical one: How does one estimate the change in the value of stocks over the interval DT? The software provides three algorithms for doing this estimation: [Euler's method](#), [2nd-order Runge-Kutta method](#), and [4th-order Runge-Kutta method](#). (To change the integration method, open the Run Specs... dialog under the Run menu.) We'll deal with each in turn.

 [Related Topics](#)

# Accuracy of the Simulation Algorithms

Compared to Euler's method, both Runge-Kutta methods require additional evaluations within each iteration. The 2nd-order method requires two additional evaluations and the 4th-order method requires four additional evaluations. If these two methods were no more accurate than Euler's, it certainly would not be worth the cost of the additional calculations.

A good way to compare the three methods is with an example. The temperature model presented in the [Simulation Algorithm Conceptual Overview](#) and [Simulation Algorithm Technical Overview](#) was used to estimate values of temperature using each of the simulation algorithms. Since the Runge-Kutta methods require additional calculations in each iteration, a different step size (DT) was used so that the total number of calculations performed was the same in each case. The results are shown in Figure 13-5.

*Figure 13-5 Comparison of Euler's, 2nd-Order, and 4th-Order Runge-Kutta*

Time	Exact Value $(100 * e^{-0.5*time})$	Euler's Method $(dt = 0.025)$	2nd-Order Runge-Kutta $(dt = 0.05)$	4th-Order Runge-Kutta $(dt = 0.1)$
0	100.000000	100.000000	100.000000	100.000000
0.1	95.122942	95.092971	95.123447	95.122943
0.2	90.483742	90.426732	90.484702	90.483742
0.3	86.070798	85.989466	86.072168	86.070798
0.4	81.873075	81.769938	81.874813	81.873076
0.5	77.880078	77.757464	77.882145	77.880079
0.6	74.081822	73.941883	74.084181	74.081823
0.7	70.468809	70.313533	70.471427	70.46881
0.8	67.032005	66.863228	67.034851	67.032006
0.9	63.762815	63.58223	63.765861	63.762817
1.0	60.653066	60.462232	60.656285	60.653068

Notice that the step size (DT) for 2nd-order Runge-Kutta is twice that of the Euler's estimate, while the 4th-order Runge-Kutta estimates use a step size 4 times larger than Euler's. However, even with the larger step sizes, the errors produced with the Runge-Kutta methods are significantly smaller than those produced using Euler's.

 [Related Topics](#)

# Choosing the Appropriate Algorithm and DT

Each of the simulation algorithms employed by the software has its own niche. The appropriate choice for an algorithm depends largely on the constructs that you employ in a given model. Similar considerations exist when you choose DT. Below, we provide a brief set of guidelines for choosing an appropriate simulation algorithm, and an appropriate step size (DT) for your simulation.

- If your model uses any of the software's discrete objects, or if you use Built-in functions to generate integer values (e.g., IF-THEN-ELSE logic to set 0-1 flags, INT, ROUND, SWITCH, etc.), use [Euler's method](#). [Runge-Kutta methods](#) were designed to deal with continuously varying systems. As such, they do not deal well with queues, conveyors, and ovens. Nor do they deal well with integer values. Check to make sure that there are no discrete objects or Builtins that generate integer values in your model if you are using either of the Runge-Kutta methods.
- Use one of the Runge-Kutta methods if the system you're modeling is continuous and has inherent oscillatory tendencies. Because the integration error of Euler's method is cumulative, you will get specious results if you use Euler's method. A system that you know to be a sustained oscillator, for example, will always generate expanding oscillations if you're using Euler's method.
- Use a step size (DT) which achieves a good compromise between accuracy of results and speed of simulation. The smaller your simulation time step, the more accurate your simulation results. Unfortunately, the price of this accuracy is a long turnaround time for your simulation. Choosing a DT that achieves a good compromise may take a few trials. In general, a good first approximation would be to choose a step size one-half of the shortest time delay in your model. Many time delays may be apparent, such as average days payable, a shipping lead time, an oven cook time, or a training delay. If you can determine the shortest time delay, you've got a good starting point for selecting the step size.
- Always test your model for sensitivity to step size. Begin by running your model for the full length of a simulation. Plot and print some variables. Cut the step size in half and repeat the simulation. Compare the results from the two runs. Did the results change significantly? If they did, cut the step size in half and test again. Continue this process until the improvement in results becomes negligible.

For more information about DT, see [Introduction to DT](#).

 [Related Topics](#)

# Euler's Method

By far the simplest algorithm used by the software is Euler's method. In this method, the computed values for flows provide the estimate for the change in corresponding stocks over the interval DT. In the simple cooling model, the initialization and iteration phases look like this:

## Initialization Phase (Euler's Method)

**Step 1.** Create a list of all stocks, flows and converters in required order of evaluation.

**Step 2.** Calculate initial values of all stocks, flows and converters (in order of evaluation).

Time = From Time

--->Time = 0

Stock<sub>t=0</sub> = f(initial values, stocks, converters, flows)

--->INIT Temperature = 100

Converters = f(stocks, converters, flows)

--->constant = 0.5

Flows = f(stocks, converters, flows)

--->cooling = Temperature \* constant

## Iteration Phase (Euler's Method)

**Step 1.** Estimate change in stocks over the interval DT.

$\Delta stock = dt * flow$

Calculate new value for Stocks based on this estimate.

Stock<sub>t</sub> = Stock<sub>t-dt</sub> +  $\Delta stock$

--->Temperature = Temperature + (-cooling) \* dt

**Step 2.** Calculate new values for flows and converters (in order of evaluation).

Converters =  $f(\text{stocks}, \text{converters}, \text{flows})$

--->constant = 0.5

Flows =  $f(\text{stocks}, \text{converters}, \text{flows})$

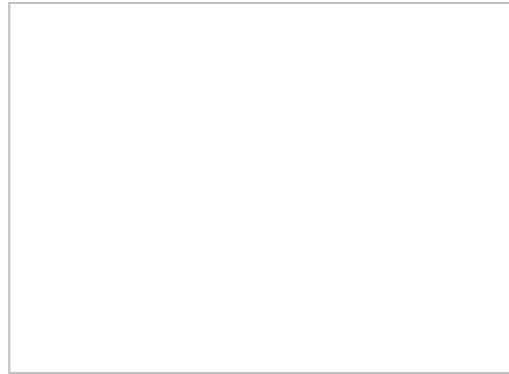
--->cooling = Temperature \* constant

**Step 3.** Update simulation time. Stop iteration when Time  $\geq$  simulation stoptime.

Time = Time + dt

The numeric output generated by the cooling model, for the first four time units, is shown in the Figure 13-3, below. The output was produced using Euler's method, with DT = 0.5, beginning balances, and summed reporting of flows.

*Figure 13-3 Numerical Output of Cooling Model*



As the data show, cooling declines in proportion to Temperature. The data set for Temperature approximates an exponential decay pattern, such as the one shown in the [Simulation Algorithm Conceptual Overview](#).

At this point, however, it is useful to ask about the accuracy of this approximation. As Figure 13-4 shows, the model output is not very accurate when compared with the analytical solution to the equivalent equation ( $\text{Temperature} = 100e^{-0.5 \cdot \text{time}}$ ). Because Euler's method simply uses the value calculated for the flow as its estimate for the change in the stock over the interval DT, and because the interval is a large one (relative to the continuous case), an integration error is introduced. This integration error is the difference between the two curves on the plot.

*Figure 13-4  
Analytic vs. Model Solution for Cooling Model*

## Related Topics

# The Runge-Kutta Methods

It always is possible to reduce this integration error by making the DT of your models smaller. As DT approaches zero, Euler's approximation approaches the exact solution. But as you make DT smaller, you also increase the number of calculations (and the time) required to conduct your simulation - clearly not a desirable outcome. Another way to reduce the integration error is to use a more sophisticated procedure for estimating the change in stocks over a given DT. The two Runge-Kutta algorithms give you a higher degree of sophistication.

To aid in the exposition of Runge-Kutta, a bit of notation will prove helpful. Let " $x$ " represent any stock and " $f(t, x)$ " represent the functional notation for flows that depend on time " $t$ " and stock " $x$ ".

# 2nd-Order Runge-Kutta

Of the two Runge-Kutta methods, 2nd-order is the simpler. Basically, this algorithm uses two flow calculations within a given DT to create an estimate for the change in a stock over the DT. As with Euler's method, the solution of equations involves both an initialization and an iteration phase.

## Initialization Phase (Runge-Kutta 2)

**Step 1.** Create a list of all stocks, flows and converters in required order of evaluation.

**Step 2.** Calculate initial values of all stocks, flows and converters (in order of evaluation).

Time = From Time

Stock<sub>t=0</sub> = f(initial values, stocks, converters, flows)

Converters = f(stocks, converters, flows)

Flows = f(stocks, converters, flows)

## Iteration Phase (Runge-Kutta 2)

### Step 1.

(a) Estimate change in stocks over the interval DT. Use two points to create this estimate:

- (1) F1 = dt \* f(t, x) [the same as Euler's flow estimate]
- (2) F2 = dt \* f(t+dt, x + F1)
- (3)  $\Delta\text{stock} = 1/2 * (F1 + F2)$

(b) Calculate new value for stocks based on this estimate.

Stock<sub>t</sub> = Stock<sub>t-dt</sub> +  $\Delta\text{stock}$

### Step 2. Calculate new values for flows and converters (in order of evaluation).

Converters = f(stocks, converters, flows)

Flows = f(stocks, converters, flows)

### Step 3. Update simulation time. Stop iteration when Time is $\geq$ simulation To Time.

Time = Time + dt

The intermediate calculations in Step 1 of the iteration phase enable the 2nd-order Runge-Kutta method to create a more accurate estimate for the change in stocks during a given DT. This estimate is based on an average of flow

values computed at the beginning, and at the end, of the DT interval.

# 4th-order Runge-Kutta

4th-order Runge-Kutta works just like its 2nd-order counterpart, except that it uses four rather than two flow calculations within a given DT to create an estimate for the change in a stock over the DT. A weighted average of these calculations is used as the estimate for the change in the stock:

## Initialization Phase (Runge-Kutta 4)

**Step 1.** Create a list of all stocks, flows and converters in required order of evaluation.

**Step 2.** Calculate initial values of all stocks, flows and converters (in order of evaluation).

Time = From Time

Stock<sub>t=0</sub> = f(initial values, stocks, converters, flows)

Converters = f(stocks, converters, flows)

Flows = f(stocks, converters, flows)

## Iteration phase (Runge-Kutta 4)

### Step 1.

(a) Estimate change in stocks over the interval DT. Use four points to create this estimate:

$$(1) F1 = dt * f(t, x)$$

$$(2) F2 = dt * f(t+dt/2, x + 1/2F1)$$

$$(2) F3 = dt * f(t+dt/2, x + 1/2F2)$$

$$(4) F4 = dt * f(t+dt, x + F3)$$

$$(5) \Delta stock = 1/6 * (F1 + 2F2 + 2F3 + F4)$$

(b) Calculate new value for Stocks based on this estimate.

$$Stock_t = Stock_{t-dt} + \Delta stock$$

### Step 2. Calculate new values for flows and converters (in order of evaluation).

Converters = f(stocks, converters, flows)

Flows = f(stocks, converters, flows)

### Step 3. Update simulation time. Stop iteration when Time is >= simulation To Time.

$$Time = Time + dt$$

The intermediate calculations in the first step of the iteration phase are the key to the 4th-order Runge-Kutta method.  $F_1$  represents the rate of change of the stock, i.e., the flow, at time "t".  $F_2$  is a new calculation of the flow at  $1/2$  an interval into the future,  $t+dt/2$ . In order to calculate  $F_2$ ,  $F_1$  is used to update the stocks. These then are used to re-calculate the flows at  $t+dt/2$ . One-half of  $F_1$  is used in estimating  $F_2$  since the calculation is made over only  $1/2$  the interval.  $F_3$  is a new calculation of the flow at  $t+dt/2$ . To calculate  $F_3$ ,  $F_2$  is used to update the stocks, which then are used to re-calculate the flows at  $t+dt/2$ . Again, only  $1/2$  of  $F_2$  is used to update the stocks since the calculation is made over  $1/2$  an interval.  $F_4$  is a calculation of the flow at one full interval in the future,  $t+dt$ . To calculate  $F_4$ ,  $F_3$  is used as the flow over the full interval. The stocks are updated using  $F_3$ , then  $F_4$  is estimated.

Finally, the four intermediate flow calculations are averaged together, yielding an estimate of the flow. It is this average flow that is used to update the stock before the next iteration begins.

 [Related Topics](#)

# Introduction to Implicit Stock/Flow Rules of Grammar

The topics in this section look at the implicit stock/flow rules of grammar employed by the software. [Implicit Connectors](#) describes the implicit connectors which exist between the various stock types and their associated inflows and outflows. [Flow Prioritization](#) looks at the rules of flow prioritization which come into play whenever multiple flows are associated with a given stock.

 [Related Topics](#)

# Implicit Connectors

This section identifies the implicit connections generated by the software between different stock/flow configurations. Implicit connections are important because they can limit your ability to create explicit connections between model variables. If an implicit connection exists between model variable "A" and model variable "B," the software will prevent you from drawing a connector from variable "B" back to variable "A" in most cases. Instead, you'll get a message that circular connections are not allowed, and a notification of the implicit connection which is at the root of the problem. As noted on page 14-3, in a few cases the software will override its implicit rules to allow connections between variables.

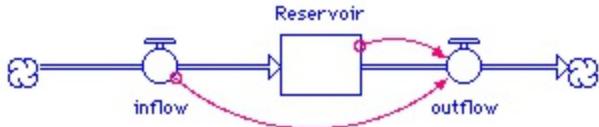
Implicit connections exist with:

- Non-negative Reservoirs
- Queues
- Conveyors
- Ovens
- Sub-models

Figures 14-1 and 14-2 identify these implicit connections, and briefly describe their implications.

*Figure 14-1 Implicit Connectors: Reservoirs, Queues, Conveyors*

### 1. Non-negative Reservoir



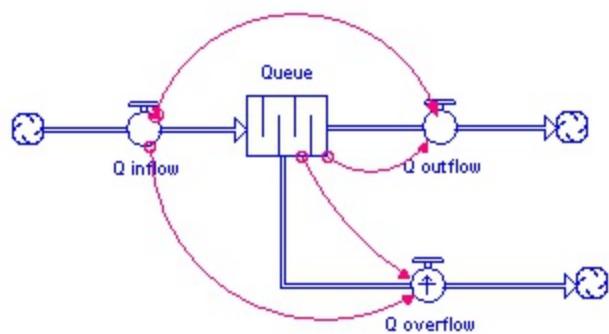
#### Implicit Connections:

- Inflow(s) to outflow(s)
- Reservoir to outflow(s)

#### Implications:

- Implicit connection from inflow to outflow broken if you connect outflow to inflow

### 2. Queue



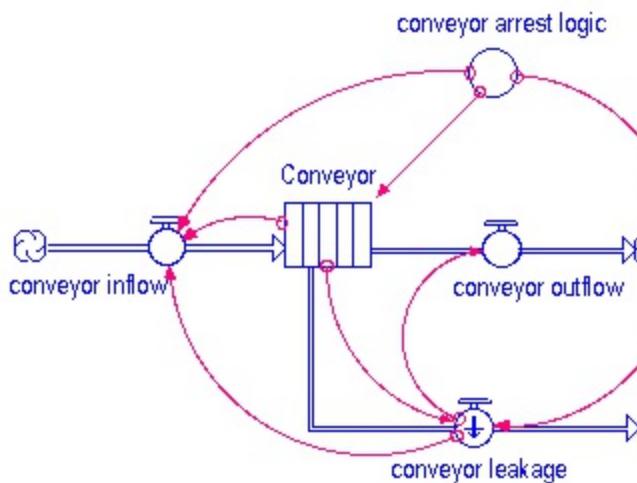
#### Implicit Connections:

- Inflow(s) to outflow(s)
- Queue to outflow(s)
- Higher-priority outflows to lower-priority overflows

#### Implications:

- Implicit connection from inflow to outflow broken if you connect outflow to inflow

### 3. Conveyor



#### Implicit Connections:

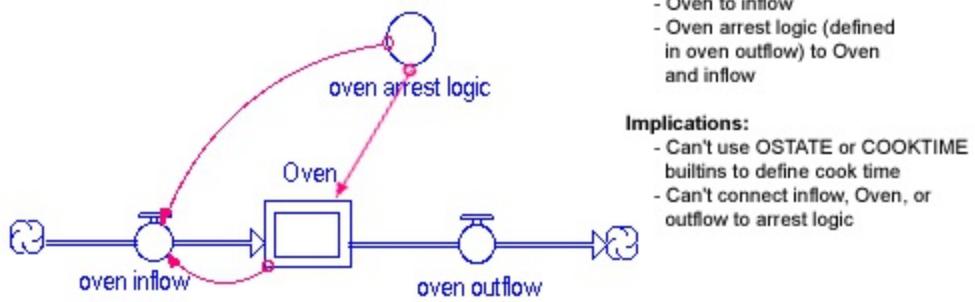
- Conveyor to inflow(s)
- Conveyor to leakage flow
- Leakage flow to flow-thru flow
- Leakage flow to inflow, when conveyor capacity is finite
- Arrest logic (specified in conveyor's outflow) to conveyor inflow, and leakage

#### Implications:

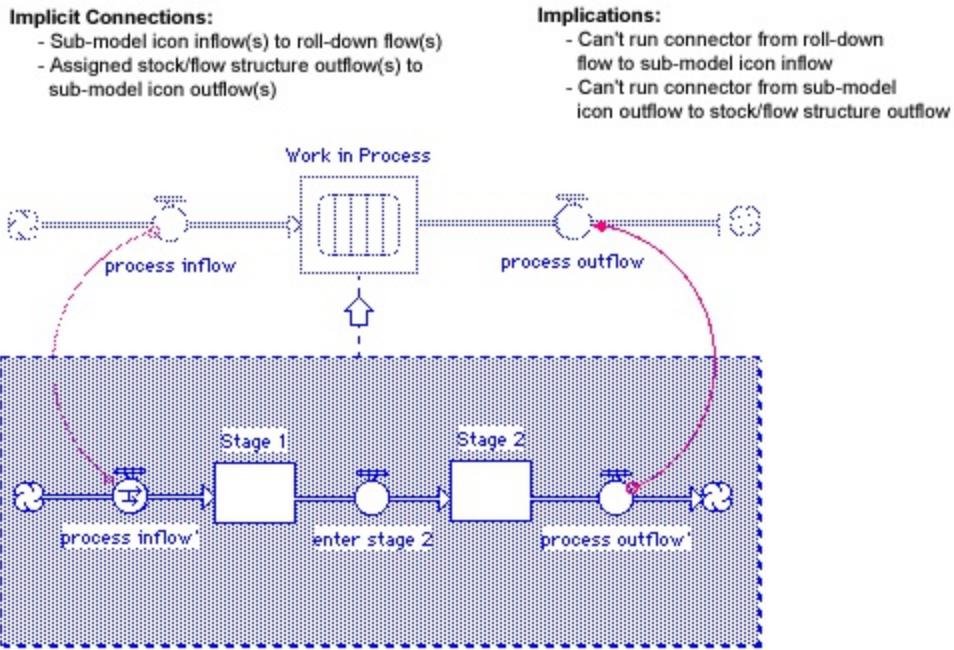
- Can't run connection from flow-thru flow to leakage flow
- Can't run connection from conveyor to flow-thru flow
- Can't connect inflow to leakage flow, when have finite conveyor capacity
- Can't connect inflow, conveyor, leakage to arrest logic
- Can't use QELEM, QLEN TRANSTIME to define transit time

**Figure 14-2**  
*Implicit Connectors: Ovens, Sub-models*

#### 4. Oven



#### 5. Sub-model



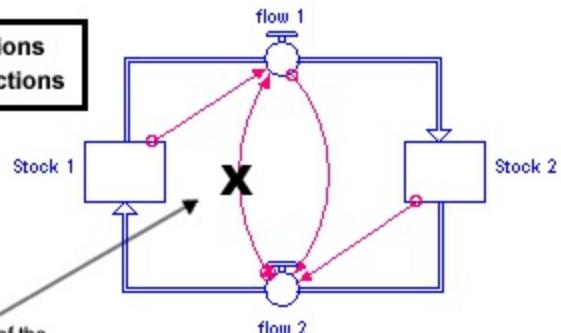
## Situations in which the software overrides its implicit connection rules

In some cases, the implicit connection rules shown in Figures 14-1 and 14-2 could lead to circular connections. To prevent this from happening, the software will systematically "break" implicit connections in order to prevent circular connections. In all cases, the software still maintains the dependency relationships that you have made with explicit connectors.

Figure 14-3 shows the most common instances in which such "breaking" occurs. The figure also notes the run-time implications of the breaking. Run-time implications will appear only when non-negativity constraints come into play.

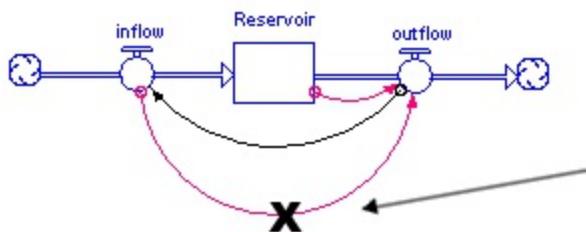
*Figure 14-3  
Instances where Implicit Connectors are Broken*

**Red Connectors = Implicit connections**  
**Black Connectors = Explicit connections**



Here the software will break the implicit connection from flow 2 to flow 1, because of the non-negative constraint on the stocks. The result is that 1 DT's worth of flow will pile up in Stock 1.

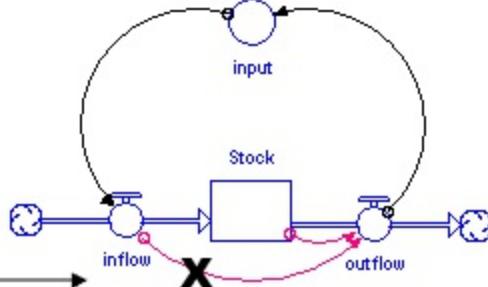
*Flow from a Stock into a second Stock, plus a flow back into the first Stock*



Here the software will break the implicit connection from inflow to outflow. 1 DT's worth of flow will pile up in the stock when non-negativity comes into play.

*Explicit connector from the outflow to the inflow of a non-negative Stock*

In this instance, the software will break the implicit connection where it would otherwise lead to a circular connection. The non-negative stock for which the connection is broken will hold 1 DT's worth of inflow.



*Indirect connection between the outflow and the inflow through other model structure*

[Related Topics](#)

# Flow Prioritization

In the software, certain stock/flow configurations will cause stocks of various types to exert control over their flows. With multiple flows, the stock's control over its flow necessitates a Prioritization scheme. When multiple flows are involved and prioritization is an issue, an inflow and outflow priority number will appear in the dialog for each flow. This section provides an in-depth look at the control and prioritization rules the software uses for:

- [Outflows from a Reservoir when non-negative is activated](#)
- [Inflows to an Inflow limited Conveyor](#)
- [Inflows to a Capacity-constrained Conveyor](#)
- [Inflows to a Queue](#)
- [Outflows from a Queue](#)

We'll look at each in turn.

# Non-negativity and Reservoir Outflow Prioritization

When the Non-negative option is checked in the Reservoir dialog, the explicit logic of outflows will be overridden whenever it will otherwise cause the Reservoir's magnitude to become negative. In the simple case, with only one outflow from a Reservoir, the non-negative logic says that whenever the magnitude of the Reservoir, plus the total inflow over the next DT, is insufficient to support the explicitly-calculated outflow over the next DT, flow the entire magnitude of the Reservoir, plus the inflow, during a single DT. Internally, the software uses a calculation of the form:

$$\text{outflow volume} = \text{MIN}(\text{Reservoir}/\text{DT} + \text{total inflow volume}, \text{explicit outflow volume})$$

With multiple outflows, the non-negative logic gets a bit trickier. Essentially, the Reservoir will apply the non-negative logic to each outflow as it apportions its contents among the outflows. The apportionment is done on a priority basis, each DT in the model simulation. The priority 1 outflow gets first crack at the contents of the Reservoir. What's left over in the Reservoir (if anything), gets allocated to the priority 2 outflow. What's left after priority 1 and priority 2 have had their turn is allocated to the priority 3 outflow, etc.

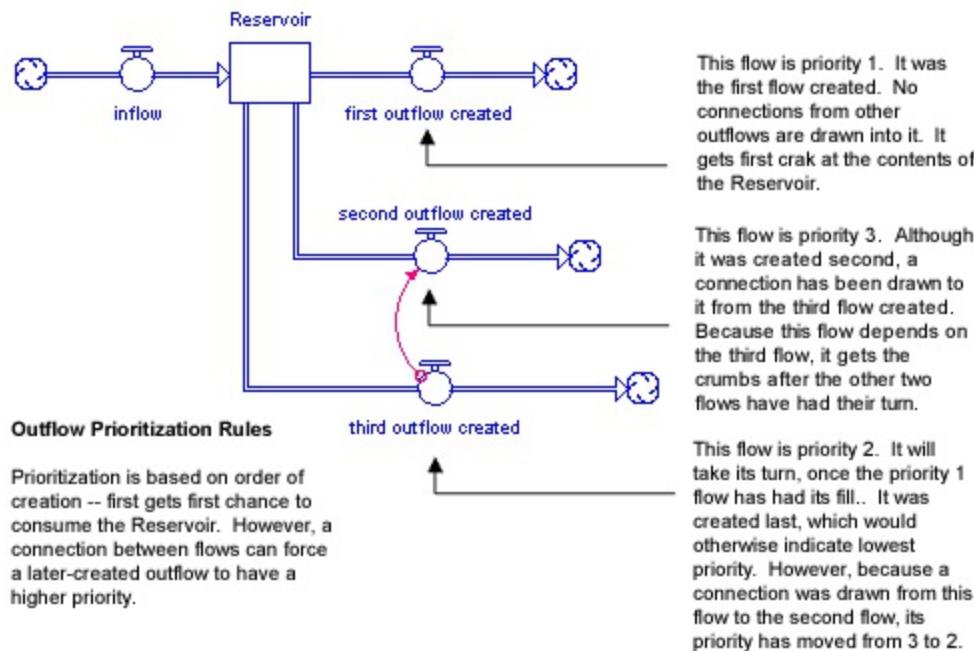
The assignment of priorities for multiple outflows is determined primarily by the inner workings of the software. However, when there are interdependencies among the flows, the flow priority will reflect the dependency relationship. Any outflow whose volume depends upon the calculation of another outflow from the same Reservoir will assume a lower priority, as shown in Figure 14-4.

*Figure 14-4  
Non-Negativity and Reservoir Outflow Prioritization*

**Non-negativity means:**

total outflow volume = MIN(Reservoir/DT + inflow, explicit outflow volume)

(explicit outflow volume = first outflow created + second outflow created + third outflow created)



# Inflow Limited Conveyors and Inflow Prioritization

In Conveyors, a finite Inflow limit can constrain inflows. The Inflow limit field is disabled whenever a Conveyor has inflows emanating from either another Conveyor or from an Oven. When inflows come from Reservoirs, clouds, or Queues, however, the Inflow limit may be set to a finite value. The underlying Inflow limit logic works in one of two ways, depending upon whether the "Discrete" check box is checked in the Conveyor dialog.

*Case 1: Discrete is not checked in Conveyor dialog (possible with cloud and/or Reservoir to Conveyor flows only).*

When the inflows to the Conveyor come from clouds and/or Reservoirs only, and the "Discrete" box is not checked, the Inflow limit will ensure that the total flow volume into the Conveyor will never exceed the limit. This means that the explicit inflow logic will be overridden whenever it yields a volume that exceeds the Conveyor's Inflow limit. For Reservoir to Conveyor or cloud to Conveyor flows, the Inflow limit is equivalent to the following:

$$\text{flow volume} = \text{MIN}(\text{Inflow limit}, \text{explicit inflow volume})$$

Each DT in the simulation, the software will check to ensure that the explicit flow volume does not exceed the Inflow limit. If it does, the total inflow will be "clipped" to the value that you have set for the inflow limit.

The software will prioritize multiple inflows from clouds and/or Reservoirs to an Inflow-limited Conveyor. The first priority inflow gets first crack at the Inflow limit. Any remaining space gets allocated to the second priority inflow, and so on. Priority is established by the order in which inflows are created. The priority of flows is the order they were connected to the Conveyor. As with outflows from Reservoirs, the prioritization is altered to reflect dependency relationships among inflows to a given Conveyor. Figure 14-4 shows how a dependency between flows will change the prioritization of the flows.

*Case 2: Discrete is checked in Conveyor dialog (optional with cloud and/or Reservoir to Conveyor flows only; set as the default when Queue to Conveyor flows exist).*

When "Discrete" is checked, the Inflow limit will ensure that no more will flow into the Conveyor in a given unit of time than you have specified in the Inflow limit box. The software will keep track of how much has flowed into the Conveyor over the unit of time. If the total inflow volume would otherwise

cause the Inflow limit to be exceeded, it will be "clipped" to the appropriate value. Each unit of time (i.e., at time = 1, time = 2, etc.) the internal tracking mechanism will reset itself.

### ***Special Notes on Queue to Conveyor flows:***

- Whenever a flow runs from a Queue to a Conveyor, the "Split batches" option becomes enabled. Split batches enables the Conveyor to chip away at the next element in the Queue, taking only what's needed to satisfy the Inflow limit constraint. If Split batches is not checked, the Conveyor has the potential to become gridlocked whenever the next element in the Queue is larger than the Inflow limit. To prevent gridlock, either designate a second outflow from the Queue as an "Overflow," or ensure that Split batches is checked in the Conveyor dialog.
- Whenever multiple flows come into a Discrete, Inflow-limited Conveyor, prioritization is required. The prioritization scheme used by the software is essentially the same as described on the previous page for non-discrete, Inflow-limited Conveyors. However, whenever a flow exists from a Queue to a Conveyor, it will always assume top priority. Subsequent flows from Queues will be prioritized in their order of creation.

# Capacity-Constrained Conveyors and Inflow Prioritization

In Conveyors, a finite Capacity can constrain inflows. The Capacity field is disabled whenever a Conveyor has inflows emanating from either another Conveyor or from an Oven. When inflows come from Reservoirs, clouds, or Queues, however, the Capacity may be set to a finite value.

In words, a Capacity-constrained Conveyor will ensure that the total inflow to the Conveyor will never cause the magnitude of the Conveyor to exceed the Capacity you have set. Flow logic will be overridden whenever nominal inflow volume (i.e., what "wants" to flow from clouds, Reservoirs and Queues) would cause Capacity to be exceeded. Internally, the software uses equations of the form:

$$\text{inflow volume} = \text{MIN}((\text{Capacity} - \text{Conveyor})/\text{DT} + \text{total outflow volume, explicit inflow volume})$$

Each DT in the simulation, the software will check to ensure that the nominal inflow volume will not cause the capacity to be exceeded. If it will, the total inflow will be "clipped" to the appropriate value.

## ***Special Note on Queue to Conveyor flows:***

- Whenever a flow runs from a Queue to a Conveyor, the "Split batches" option becomes enabled. Split batches enables the Conveyor to chip away at the next element in the Queue, taking only what's needed to satisfy the Capacity constraint. If Split batches is not checked, the Conveyor has the potential to become gridlocked whenever the next element in the Queue would cause the Conveyor to exceed its Capacity constraint. To prevent gridlock, either designate a second outflow from the Queue as an "Overflow," or ensure that Split batches is checked in the Conveyor dialog.

The software will prioritize multiple inflows from clouds and/or Reservoirs to a Capacity-constrained Conveyor in the same way that it prioritizes flows under Inflow limits. As with Inflow-limited Conveyors, a Queue to Conveyor flow will always have a higher priority than a cloud or Reservoir to Conveyor flow.

## Inflows to a Queue

In the simple case with one inflow to a Queue, the Queue will create a new element or slot each DT to hold the inflow. With multiple inflows to the Queue, however, the software must create multiple slots - one for each of its inflows. The software prioritizes the inflows to a Queue to resolve the issue of which flow gets to enter the Queue first, in a given DT. As with inflows to Conveyors, prioritization is established by order of flow creation. The prioritization is altered to reflect implicit and explicit dependency relationships in the model. Queue outflows to the Queue will always be of highest priority, in their order of creation. A Conveyor leakage flow will always have higher priority than the flow-thru from the same Conveyor, when both flow into the same Queue. A flow whose calculation depends upon another flow will always have a lower priority than that other flow, when both flow into the same Queue.

# Outflows from a Queue

When multiple outflows are attached to a Queue, the software must first prioritize the outflows to resolve the issue of which flow gets to leave the Queue first, within a given DT. Once again, the priority is based on the order of creation. The first outflow created gets to flow first. What's left in the Queue (after the first flow has been calculated) is then available to the second flow that was created, etc.

Given the flow prioritization, the software uses the following control rules to determine the volume of each flow, each DT:

- Queue to Oven: Flow logic is controlled by Oven.
- Queue to Conveyor: Flow logic is controlled by Conveyor.
- Queue to Reservoir: Flow entire contents of Queue, plus whatever is flowing into the Queue - material will pass through the Queue.
- Queue to Cloud: Flow entire contents of Queue, plus whatever is flowing into the Queue - material will pass through the Queue.
- Queue to Queue: Flow next element in Queue, plus whatever is flowing into the Queue - material will pass through the Queue.

# The Bottom Line

In terms of flow control and prioritization, the important information will be displayed to you within the flow dialogs. When relevant, the flow dialogs will tell you both the outflow and inflow priority of a given flow. In many cases, this can be overridden by making explicit dependencies between flows.

 [Related Topics](#)

# Introduction to Spatial Maps

Spatial Map is a stand-alone application that allows you to map array data in your model to color values or images so that you can view your data as three-dimensional, spatial information. In the spatial map, the x and y axes are defined by the bounds of the array and z is represented by a color or image. (Elevation and temperature are examples of z values.) There is a forced 1-to-1 mapping between the array dimensions and the spatial area you are viewing.

In Spatial Map, you specify the ranges of data values to be assigned and the colors or images that will represent each value range in the spatial map. The color ranges can either be discrete ranges (transitioning abruptly from one color to the next), or they can blend (transitioning smoothly from one color to the next).

Spatial Map allows both static and dynamic mapping. With static mapping, you specify a fixed set of values from your model that you want to view as a spatial map, and then use Spatial Map to define the color or image mapping for the ranges you specify. With dynamic mapping, you can use **iThink** or **STELLA** to automatically refresh the data in the spatial map with updated values as the model runs.

Regardless of the type of mapping you want to perform, you need to create a .CSV (comma-separated values) file that will contain the data values. You export your data from the model to the .CSV file and then specify the same .CSV file in Spatial Map to create a link between the model and Spatial Map.

# Creating spatial maps

The following outline provides an overview of the basic steps you need to perform to create a spatial map from your **iThink** or **STELLA** model data:

1. Build a model that includes spatial data, which is contained in an [array](#).
2. Create an empty .CSV file. You can create a .CSV file by using a spreadsheet program like Microsoft Excel®, or you can create a blank text file with any text editor and name the file with the .CSV file extension.
3. [Create a table](#) in your model and place **one** element of the desired array in the table.
4. [Export one set of values from the table to the .CSV file you created](#).
5. [Specify the .CSV file in Spatial Map](#).
6. [Set up color or image range configurations in Spatial Map](#).

 [Related Topics](#)

# Exporting Model Data to Use with Spatial Map

In order to use Spatial Map with your model, you first need to create a .CSV (comma-separated values) file that contains the values from the model variable that you want to view as a spatial map. The .CSV file serves as the data file for Spatial Map, and you will [specify that same .CSV file](#) when you want to [create color or image range configurations](#) for the exported data in Spatial Map.

To create a .CSV file that contains model variable values, your model must include a table that contains a single element of the array for which you want to view data spatially. Once you create the table, you can export the table's values to the .CSV file.

---

**Notes:** To create a spatial map from model data, the table that you create must contain only one element of the array. That is, the table can contain only one specific row-column pair in the array, not the entire array.

If you will be creating a dynamic map (where the spatial map is updated with new values as the model runs), you *cannot* export stock variables to the .CSV file. The export process will export only the initial values for the stock variable, but no other values. To create a dynamic map with stock values, you must first copy the stock to a converter, then put the converter into a table, and export the converter values from the table by using the following procedure.

---

# To export model data to use with Spatial Map

1. Use Microsoft Excel or another program to create a blank .CSV file.
2. In **iThink** or **STELLA**, open the model that contains the array variable that you want to view spatially.
3. Create a table and place one element of the desired array in the table.
4. From the Edit menu, choose **Export Data**. The Export Data dialog box opens.
5. Under "Export Type", select the **Persistent - Export data from the model, establishing a link** option, and then select the **Dynamic - Update when data changes** option.
6. Under "Export Data Source", select **Export variables in table**, and then select the table that you want to export.
7. Under "Interval", select **One set of values**.
8. Under "Export Destination", click the **Browse** button and then navigate to and select the .CSV file you created in step 1.
9. The Export Data dialog box should look like this:

## Export Data

Export Data to an Excel Worksheet

### Export Type

- One Time - Export data from the model without establishing a link
- Persistent - Export data from the model, establishing a link
  - On Demand - Update when requested by user
  - Dynamic - Update when data changes

### Export Data Source

- Export all model variables
- Export variables in table

### Interval:

- One set of values
- Every  Years
- Every DT - Export every intermediate value during the run

### Export Destination

#### Excel File Name:

C:\Projects\isee\Spatial Map\Admin\rebecca test\rebecca test\island 2.csv

Worksheet Name:

#### Data Orientation

sales	Net Income	expenses
\$1,500.00	\$2,000.00	\$1,900.00

sales	\$1,500.00
Net Income	\$2,000.00
expenses	\$1,900.00

10. Click **OK**. The values are exported to the .CSV file.
11. If you will be creating a static map (where the spatial map reflects a fixed set of values from your model), you can exit from **iThink** or **STELLA**. If you will be creating a dynamic map (where the spatial map is updated with new values as the model runs), **iThink** or **STELLA** open while Spatial Map is open.

You can now [specify the .CSV file in Spatial Map](#).

 [Related Topics](#)

# **Specifying .CSV Files in Spatial Map**

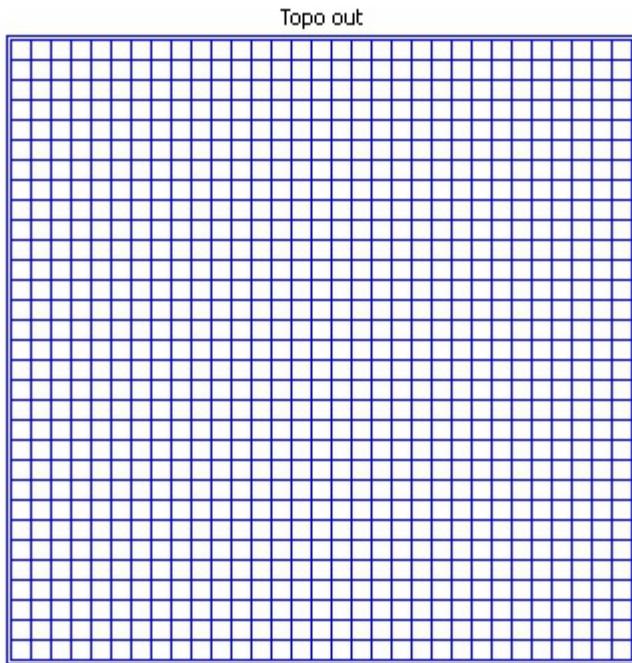
In order to use Spatial Map with your model, you need to specify a .CSV file that [contains the data that you want to map](#).

Use the following procedure to specify the .CSV file that you created with your model data.

# To specify a .CSV file in Spatial Map

1. Start Spatial Map.
2. Click the **Set CSV** button. The Find File dialog box opens.
3. Navigate to and select the .CSV file to which you [exported the model data](#) that you want to spatially map.
4. Click **Open**. The file name appears in the box next to the **Set CSV** button.
5. One empty spatial map appears in the Spatial Map window for each variable in the associated table, with the name of the variable appearing as the name of the spatial map.

The following example shows an empty spatial map for a single variable called "Topo out":

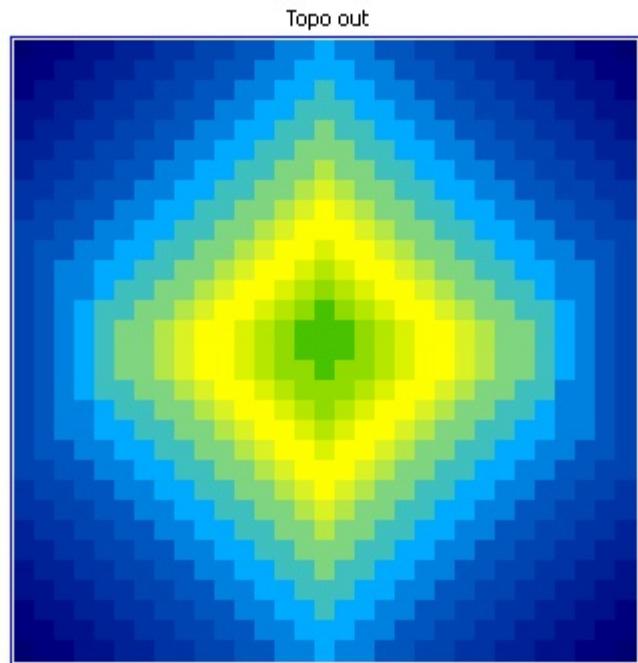
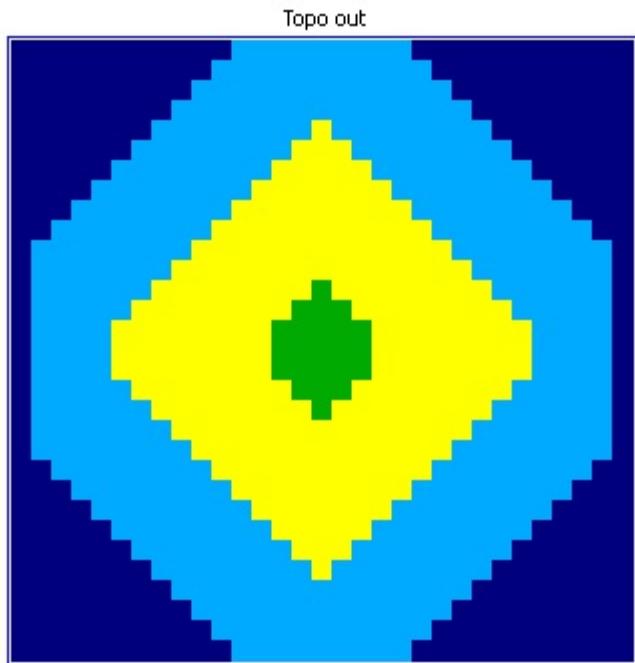


You can now [create one or more color or image configurations](#) for the spatial map.

# Working with Color and Image Configurations

You can create one or more color or image configurations, and you can apply a color or image configuration to one or more spatial maps in a single .CSV file.

Each configuration specifies the ranges of values to be assigned a color or image. You can choose to have the color ranges be discrete ranges (transitioning abruptly from one color to the next), or to have the color ranges blend (transitioning smoothly from one color to the next), as shown in the following examples.

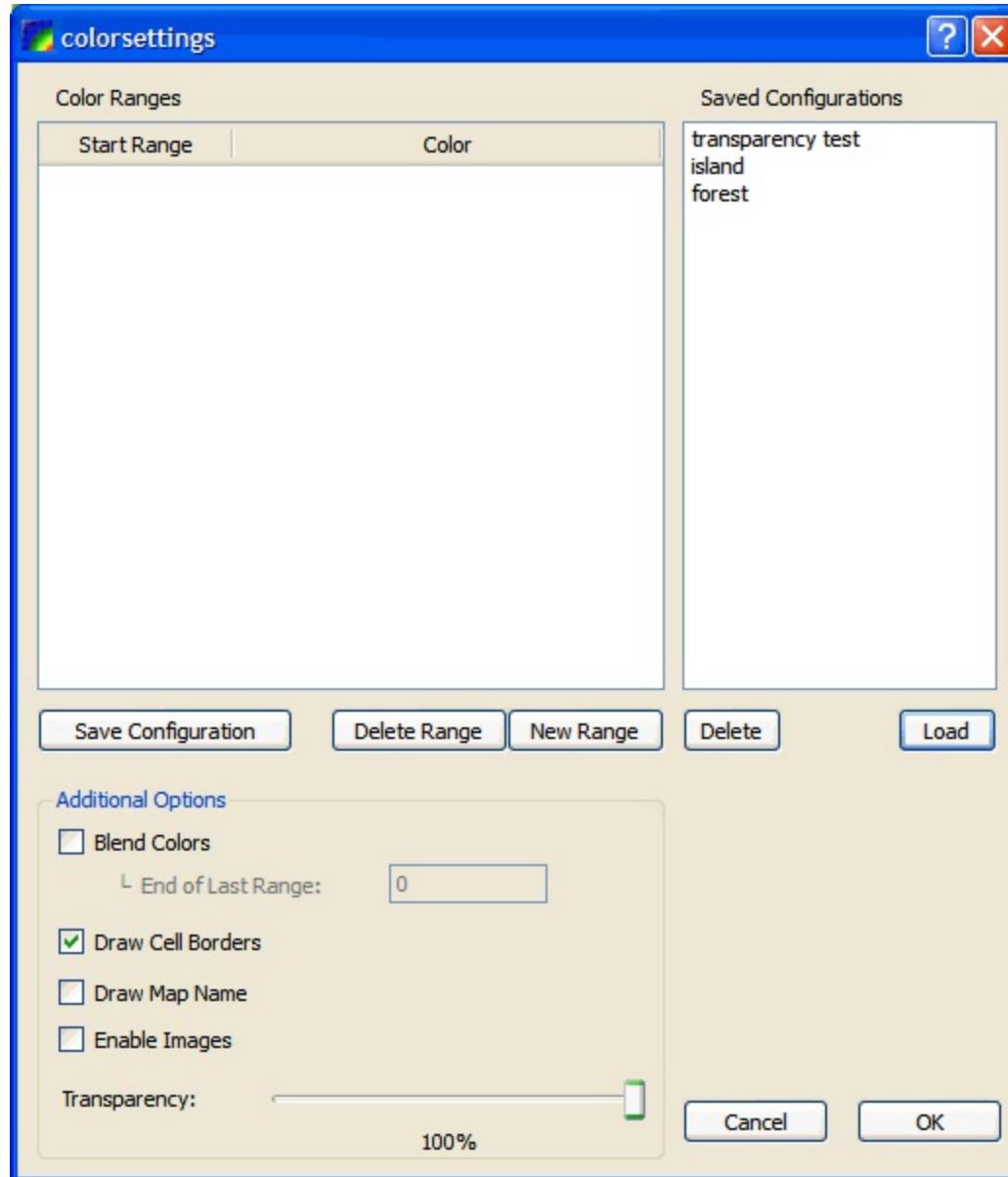


# To create a color or image configuration

1. In Spatial Map, [specify a .CSV file](#).
2. Click the  button, or double-click a spatial map to create a color or image configuration.

*Note:* If more than one spatial map is displayed in the Spatial Map window, the configuration you create automatically applies to all of the displayed spatial maps. To create different color or image configurations for one or more displayed maps, first select the maps for which you want to create a configuration (use SHIFT+click to select multiple maps) and then click the  button.

The colorsettings dialog box opens.



3. Use the "Color Ranges" table on the left side of the dialog box to specify the numerical ranges you want to use and the color for each range:

- Click the **New Range** button. A new row appears in the table.
- In the first Start Range box, type the value where the range begins and then press ENTER. The value that you enter here is the *lowest* possible value that will be colored with the color you specify. For example, if you type 5 in the first Start Range box, only values 5 and above will be colored with the color you select; colors below 5 will not be given a color.
- Click the Color box to the right of the Start Range box. The Select Color dialog box opens. Choose the color you want to assign to values that begin at the number you specified in the corresponding Start Range box and then click **OK**. The color you chose appears in the Color box.
- To add another range, click **New Range** and specify a value and color for the new range. The value you specify is the start of the new range. All values below this number will use the color specified for the preceding range.
- Repeat this process until you have specified ranges for all of the data that you want to represent in the color map.

When you have finished specifying ranges, the "Color Ranges" section will look something like this:

Color Ranges		
	Start Range	Color
1	-20	[Dark Blue]
2	-15	[Medium Dark Blue]
3	-12	[Medium Blue]
4	-8	[Light Blue]
5	-5	[Yellow]
6	-3	[Light Green]
7	0	[Green]
8	5	[Dark Green]

4. To delete a listed range, click the Start Range box for the range you want to delete and then click the **Delete Range** button
5. To edit an existing color range, click the appropriate cell to change the value or color.
6. To blend the colors from one range to the next, select the **Blend Colors**

check box (under "Additional Options" at the bottom of the dialog box). When you select this check box, each value in range is displayed by blending a proportional amount of the color for the previous range with the color for the next range. This produces a smoother transition between colors in the spatial map.

7. If you selected to blend colors, use the End of Last Range box to specify the end value for the last range. This is the maximum value you expect to see in the data and is used to smoothly blend the color for the last specified range from its start value to the end value. Spatial Map blends to white below the lowest value in the configuration and above the highest value in the configuration. (Note: The color specified for the range is the middle of the range.)

8. By default, white cell borders appear around each cell in the spatial map:



To remove the cell borders, clear the **Draw Cell Borders** check box.

9. To include the variable name above each map, select the **Draw Map Name** check box.
10. To assign an image to value ranges, select the **Enable Images** check box. Specify the images you want to use for each range:

- Click the Image box to the right of the Color box. The Find File dialog box opens. Navigate to the image file that you want to assign to values that begin at the number in the corresponding Start Range box and then click **OK**. The image and path you chose appears in the Image box.
- Repeat this process until you have specified images for all of the data that you want to in the map.

Note: When assigning an image to a value range, it is not necessary to specify a color. If you choose to specify both a color and an image for a value range, the color will be used for the background of the image.

When you have finished assigning images, the "Color Ranges" section will look something like this:

Color Ranges		
Start Range	Color	Image
1 -1		C:/deadtree.png
2 0.001		C:/firetree.png
3 1.3		C:/livetree.png

1. To make the colors transparent, adjust the Transparency slider. Moving the slider all the way to the right makes the colors 100% transparent.
2. When you are finished creating the configuration, click **OK** to apply the configuration to the selected maps.

# To save a configuration

You can save one or more color configurations to use with any spatial map. This allows you to generate different effects for a set of data to highlight specific changes or variations in the data.

1. In Spatial Map, [specify a .CSV file](#).
2. In the colorsettings dialog box, [create the color or image range configuration](#) you want to save.
3. When you are finished, click the **Save Configuration** button. The Save As dialog box opens.
4. Type a name for the configuration and then click **OK**. The save configuration's name appears in the "Saved Configurations" list on the right side of the dialog box and is available for use with any spatial map. For information about changing the specified configuration for a spatial map, see [To apply a saved configuration](#), below.

# To apply a saved configuration

Once you have saved a configuration, you can apply that configuration to any spatial map.

1. In Spatial Map, [specify a .CSV file](#).
2. Select the spatial maps to which you want to apply a saved configuration:
  - To apply a configuration to all spatial maps in the file, click the  button.
  - To apply a configuration to a single spatial map in the file, select the spatial map and then click the  button, or double-click the spatial map.
  - To apply a configuration to multiple spatial maps in the file, select the spatial maps by using SHIFT+click, and then click the  button.

The colorsettings dialog box opens.

3. Under "Saved Configurations", select the saved configuration that you want to use and then click **Load** (or double-click the saved configuration's name). The configuration's details appear in the "Color Range" table.
4. Click **OK** to apply the configuration to the selected spatial maps.

 [Related Topics](#)

# Adding Pictures to Spatial Maps

You can add a picture behind your spatial maps to help provide context for the spatial maps. For example, you might want to include an aerial or satellite image map of the location where the data represented in the spatial maps is taking place.

You can add any picture that is in the following formats:

- .BMP
- .GIF
- .JPG
- .JPEG
- .PNG
- .PBM
- .PGM
- .PPM
- .XBMX11

# To add a picture to spatial maps

1. Click the  button. The Find File dialog box opens.
2. Navigate to and select the image file for the picture that you want to add.
3. Click **Open**. The picture appears in the middle of the Spatial Map window, behind the spatial maps.
4. To move the picture to a different location in the window, click and drag it to the location you want.
5. To resize the picture, click and drag the bottom-right corner of the picture.
6. To delete the picture, click the picture and then press DELETE.

 [Related Topics](#)

# **Zooming In and Out of Spatial Maps**

While you are viewing spatial maps, you can easily zoom in or out to view more or less detail. Zooming in allows you to see more detail, but presents a blockier view of the map. Zooming out presents a smoother view of the map.

## To zoom in or out of Spatial Map

- Drag the bar at the bottom of the window to zoom in (right, towards the + sign) or to zoom out (left, towards the - sign).

 [Related Topics](#)

# **Copying Spatial Maps**

You can copy one or more spatial maps from the Spatial Map window and then paste them in any other document that accepts pictures (for example, a word processing document, an email message, or a presentation document created with an application like Microsoft PowerPoint®). This allows you to share your spatial maps by including them in other documents and presentations.

# To copy spatial maps

1. In the Spatial Map window, select the spatial maps that you want to copy:
  - To copy all displayed spatial maps, click the  button.
  - To copy selected spatial maps, use SHIFT+click to select the maps that you want to copy, and then click the  button.

Spatial Map copies the selected maps to the clipboard.

---

*Note:* If there is a name displayed for the spatial map, the name is not included in the copy.

2. Paste the map from the clipboard into the document in which you want it to appear.

 [Related Topics](#)

# Introduction to isee NetSim™

isee NetSim™ is a utility that allows you to publish your **iThink** and **STELLA** models to the Internet. Publishing models to the web allows you to easily share your models with anyone, even if they don't have **iThink** or **STELLA**. All they need is a browser with Adobe Flash Player 9 support and a connection to the location where you will publish the models (either an Internet connection, or a connection to a Web server that is running isee NetSim Server software).

To publish models with isee NetSim, you first prepare the model by [laying out the Interface layer in pages](#) (you may have already performed this step if you built the model with publication in mind), then you [export the model](#) from **iThink** or **STELLA** as a NetSim (.TXM) file, and then you use the isee NetSim wizard to [publish the model](#) to the location you select.

To use isee NetSim, you must purchase a license for it, and you must be running **iThink/STELLA** version 9.1 or higher. You must also have access to a web site where you can publish models. If you do not already have access to a web site, isee NetSim allows you to open a free account with Forio Broadcast™. isee NetSim includes a free Forio Broadcast account that can be created the first time you publish a model. For more information about Forio Broadcast, visit [www.forio.com/isee](http://www.forio.com/isee).

 [Related Topics](#)

# Laying Out Model Pages

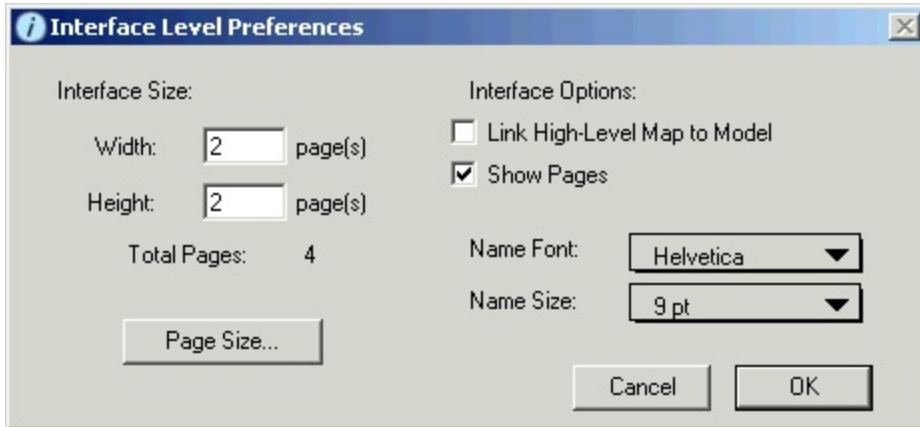
If you are publishing your model with isee NetSim or are saving the model so that it can be viewed with isee Runtime, you need to carefully lay out the portions of your model that will be seen by other people so that the page size is appropriate for your viewers and that all of the model's objects fit inside those pages:

- For models being published with isee NetSim, people viewing your model will have access to only the Interface layer of your model, so you only need to lay out the Interface layer in pages.
- For models being saved as isee Runtime files, people viewing your model will have access to the Interface layer of your model and can have access to the Model layer if you include navigation buttons on the Interface layer that navigate to Model layer. If you provide access to the Model layer, you will want to lay out both the Interface and Model layers in pages.

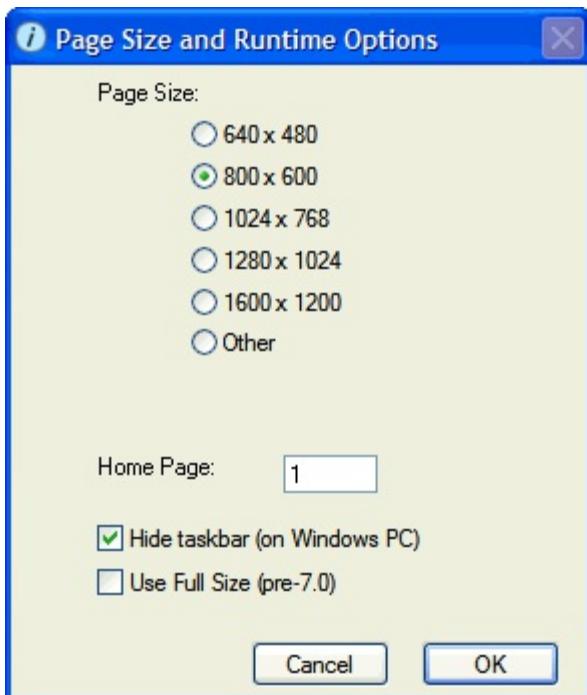
If you are building a model that you intend to make available with isee NetSim or isee Runtime, you should plan the lay out of the Interface layer (and Model layer, if applicable) as you build the model. If you are publishing an existing model, you may need to make some minor changes to the layout to be sure that the objects appear correctly on pages before you export or publish the file.

# To lay out model pages

1. In **iThink** or **STELLA**, open the model whose pages you want to publish.
2. Go to the Interface layer.
3. From the Interface menu, choose **Interface Prefs**. The Interface Level Preferences dialog box opens.



4. To enable pages on the Interface layer, select the **Show Pages** check box.
5. Use the Width and Height boxes to specify the size of the Interface layer in pages. For example, to have the Interface layer be divided into just two pages across, type **2** in the Width box and **1** in the Height box. The Total Pages value updates to indicate the total number of pages you've specified.
6. Click the **Page Size** button. The Page Size and Runtime Options dialog box opens.

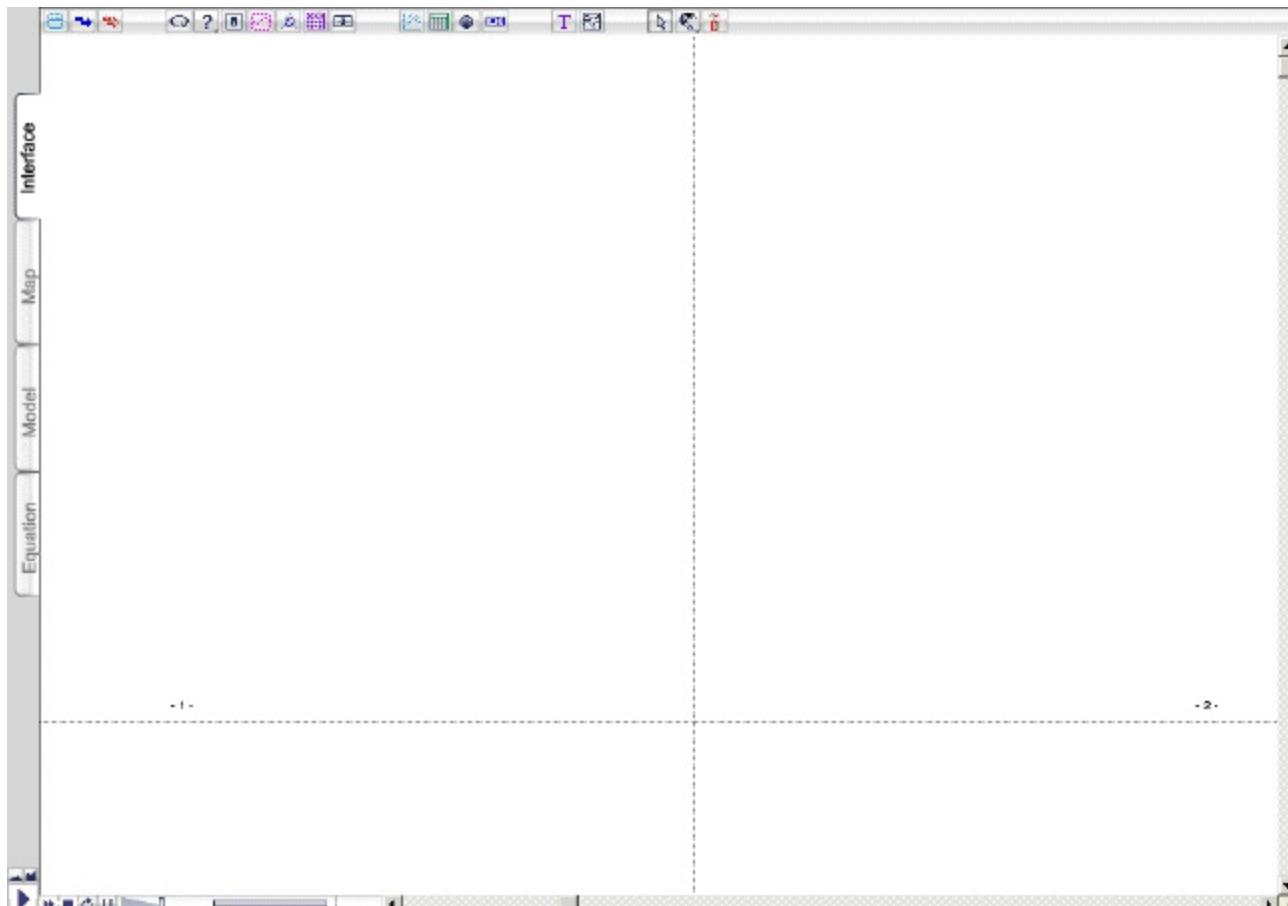


7. Select the appropriate Page Size option. This option determines how the pages will be formatted based on the screen resolution you select. In general, we recommend that you select the **800 x 600** option, unless you

are sure all of your model viewers will be using a different screen resolution.

8. Use the Home Page box to specify which page is the starting page when users view your published model. The Home Page is also that pages that users will be navigated to if you include a Home on the Interface.
9. If you will be saving this model as an isee Runtime file and you want to hide the Windows taskbar when users view the model, select the **Hide taskbar (on Windows PC)** check box.
10. Click **OK**.

The Interface layer now shows dashed lines that indicate the boundaries of the pages you specified.



11. Create and/or arrange interface objects on the pages. When you define navigation buttons, you can define specify camera, return, or page navigation (previous page, next page, and home page). For more information about defining navigation buttons, see [Button Dialog Operations](#).

*Tip:* To easily move many objects at once, zoom out on the model and then use drag-select to select multiple objects at once, then drag one selected object to move them all the same direction and distance.

12. If you will be saving this model as an isee Runtime file and you want to

allow users to view the Model layer via a navigation button from the Interface layer, go to the Model layer of the model.

13. From the Model menu, choose **Model Prefs**. The [Model Preferences dialog box](#) opens.
14. To enable pages on the Model layer, select the **Show Pages** check box.
15. Use the Width and Height boxes to specify the size of the Model layer in pages.
16. Click **OK**.
17. Create or arrange your model on the Model layer pages.

 [Related Topics](#)

# Exporting Models for Publication

To publish models with isee NetSim, you need to export the model from **iThink** or **STELLA** as a NetSim (.TXM) file.

**Notes:** Before you export this command, make sure you have first laid out the model's Interface layer in pages. For more information, see [Laying Out Model Pages](#).

To conform to NetSim's required format, you must be able to run your model, and it cannot contain Queues, Ovens, cycle-time structures, or certain Built-in functions that are not supported by isee NetSim. The export process will let you know if your model fails to conform to the required format for isee NetSim. If your model fails to conform, the export process will be stopped. Other interface objects that are not supported will not cause the export or publication to fail, but you will receive a message that says that they have been excluded from the model. For a current list of unsupported interface objects, go to <http://www.iseesystems.com/iseenetsim/help>.

# To export a model file for publication

1. Open the model file that you want to publish in **iThink** or **STELLA**.
2. From the File menu, choose **Export for NetSim**. The Save NetSim file as dialog box opens.
3. Navigate to the location where you want to save the file.
4. In the File name box, type the name you want to give the file (by default, the name is the same as the model file's name).
5. In the Save as type box, make sure that **NetSim file (\*.TXM)** is selected.
6. Click **Save**.
7. You can now publish the TXM file with isee NetSim.

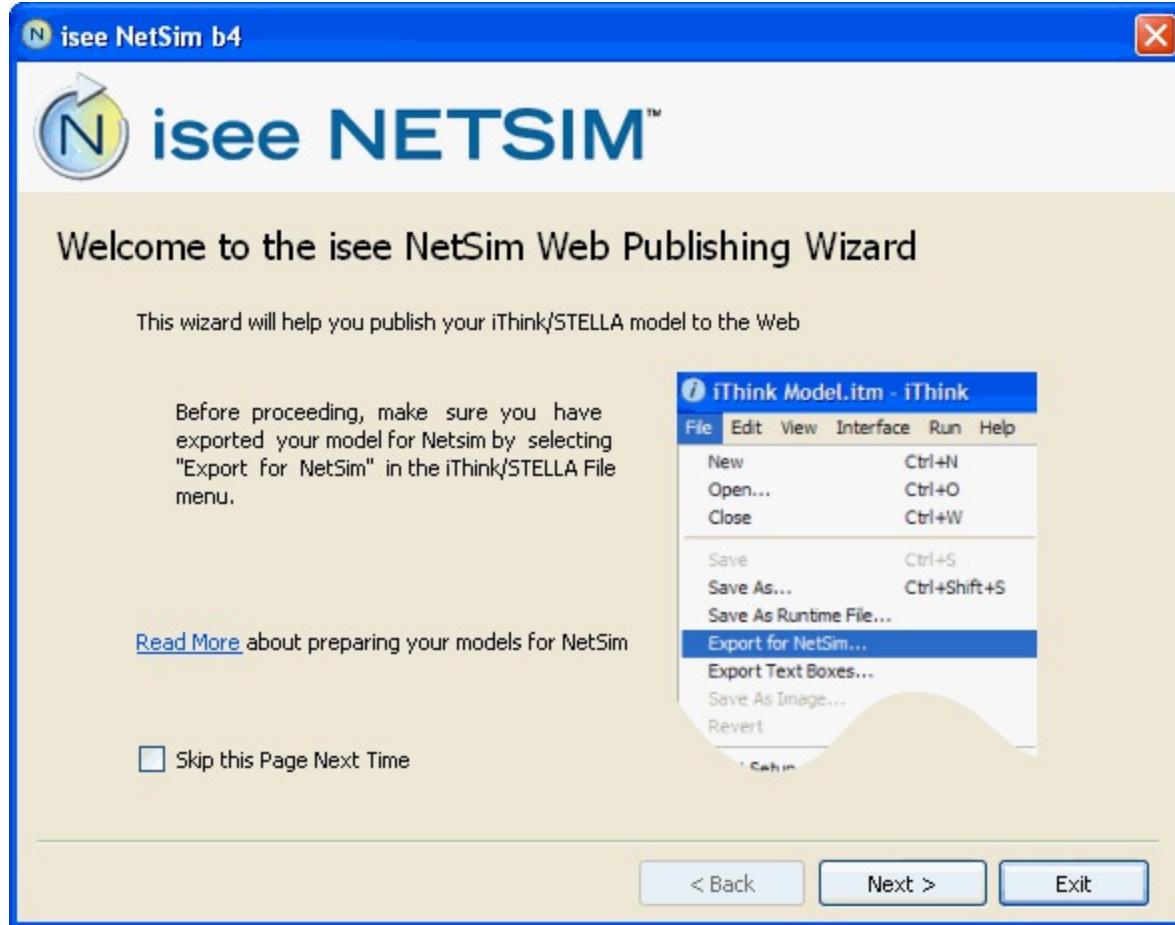
 [Related Topics](#)

# **Publishing Models with isee NetSim**

Use the following procedure to publish an exported model with isee NetSim.

# To publish a model with isee NetSim

1. Start isee NetSim by double-clicking the  icon on your desktop. The first page of the wizard opens.

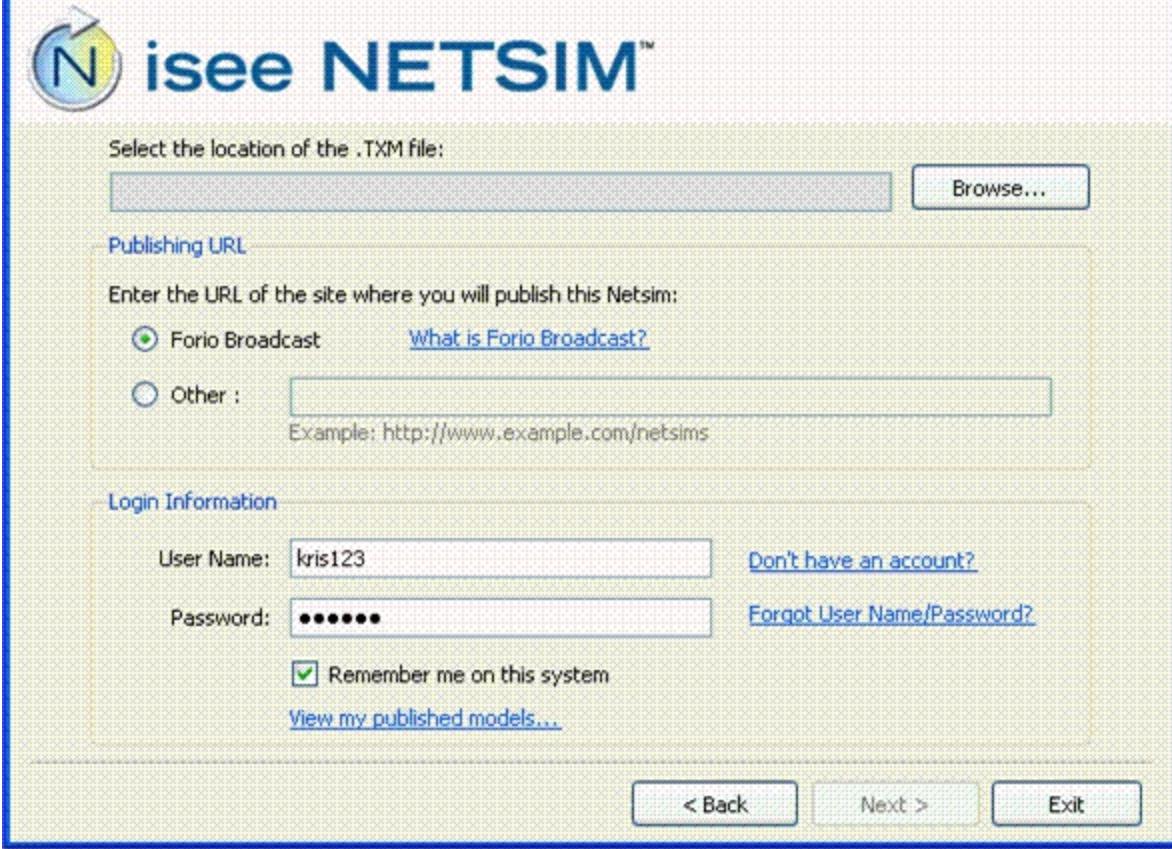


This page provides an introduction to the process and reminds you that you need to first [export your model](#) in order to publish it with isee NetSim.

---

*Note:* You can hide this page in the future by selecting the **Skip this Page Next Time** check box.

2. Click **Next**. The second page of the wizard appears.



3. Click the **Browse** button at the top of the page to navigate to and select the .TXM file that you want to publish.
4. Under “Publishing URL”, select where you want to publish the model:
  - To publish the model to Forio Broadcast, select **Forio Broadcast**. Forio Broadcast provides free access for **iThink** and **STELLA** users to publish their models on a web site.
  - To publish the model to any other web site, select **Other** and then type the full URL of the web site where you want to publish the model.
5. Under “Login Information”, type your user name and password for the location where you want to publish the model. If you selected **Forio Broadcast** and do not yet have an account, click the **Don't have an account?** link to create an account by providing a username, password, and contact information.

---

*Note:* If you have already created an account but have forgotten your password, click the **Forgot User Name/Password?** link. You will be prompted to enter identifying information so that your account name or password can be emailed to you.

---

6. To save your login information for the future, select the **Remember me on this system** check box.
7. If you have already published one or more models, you can see a list of

your published models by clicking the **View my published models** link. A web page appears that displays a list of your published models. Use this page to view or delete your published models.

8. Click **Next**. The next page of the wizard appears, which displays the status of the publication process. isee NetSim logs in to the specified site and publishes the model.

Once isee NetSim has finished publishing the model, the simulation automatically launches in your browser.

9. When you are finished publishing models with isee NetSim, click **Exit** in the wizard.

 [Related Topics](#)

# **Introduction to Tips, Tricks, and Troubleshooting**

This section provides a set of tips, tricks, and troubleshooting techniques that may be of value to you as you use the software. Here, you'll find solutions to some common technical challenges. For additional technical information, be sure to visit our web site: <http://www.iseesystems.com>. Or, contact us directly using the information in [Getting Help from isee systems](#).

 [Related Topics](#)

# Moving Files from Mac to PC

Use the following procedure to move ***iThink*** or **STELLA** files from a Mac to a PC.

# Moving model files from Mac to PC

1. Give the file the appropriate extension:

- .stm – **STELLA**
- .str – **STELLA** Runtime
- .itm – *iThink*
- .itr – *iThink* Runtime

2. Move the file to your PC. Use removable media, email, FTP or any other method you normally use.

---

Note: If you are using FTP or email attachments to move files from one machine to another, it's a good idea to compress your files using .zip or stuffit compression technology. Doing so will minimize the potential that files will become corrupted in transit.

---

 [Related Topics](#)

# Cross-platform Keyboard Equivalents

Many special keyboard characters are available on the Macintosh, but do not exist on Windows machines. Below are the common key equivalents for the software on the two platforms.

*Figure 15-1 Characters or Keys by platform*

Windows	Macintosh	Notes
$\geq$	$\geq$ or $\geq$	greater than or equal to operator
$\leq$	$\leq$ or $\leq$	less than or equal to operator
$\neq$	$\neq$ or $\neq$	does not equal operator
inf	$\infty$ or inf	infinity symbol
ctrl-del	command-delete or clear	clears single selected model element

 [Related Topics](#)

# Rules For Naming Diagram Entities

Occasionally you will try to name a Stock, Flow or Converter on the diagram and the name will keep popping back to its previous name. Since the names you choose for Stocks, Flows and Converters will be used by the software as part of equations, the software will enforce rules about which characters can be used for naming them.

- No Stock, Flow or Converter name can begin with a number of any of the following: ! # % &. You can use a number or one of these characters in the middle of the name or place them at the end, but do not use them at the beginning of the name.
  - You cannot use the same name twice in the same model.
  - You cannot use the following characters anywhere in a Stock, Flow or Converter name:  
. , < > - - / + = \* ( ) ^ [ ] { }
- However, you can use the following characters: \ @ \$

 [Related Topics](#)

# Out of Memory Alerts

Whenever you attempt to do something that requires more RAM than has been allocated to the software, you will receive an alert which states that you are out of memory. When you run out of memory, there is no recourse but to quit the program. You have the option of saving your model under a different name, thus preserving any work that you have done since the last Save command. Below are listed typical causes of this alert, as well as some memory-expanding options.

## Causes of Out of Memory Alert

- Multiple-element arrayed structures. Arrays can use up memory while taking up relatively little diagram structure.
- Too much output in comparative graphs/tables. The output displayed in comparative tables and graphs is stored in RAM. As a result, a multiple-run sensitivity analysis, with several comparative graphs/tables, can quickly consume much of the memory allocated to the software.
- Conveyors with long constant or variable transit times. Conveyors with variable transit times are deceptively simple looking. Beneath the scenes, however, such conveyors require a lot of RAM.
- Long simulations with small DT: Memory requirements for a simulation increase proportionally with the length of the simulation, and inversely with the size of DT. Longer simulations and smaller DTs will require more memory.
- Analyze mode. When this option is turned on in the Run Specs dialog, model output for *all* model variables is stored in RAM. With various combination of a large model, a long simulation, and a small DT, memory can very quickly be consumed.
- Graphic images: Some graphic images can be exceedingly memory intensive. If you do not have sufficient memory to display a graphic image within your model, the image will not be drawn and the graphics object will provide you with a text message.
- DELAY functions with long delay times.
- DERIVN function with a large number for the order.
- SMTHN functions with a large number for N.
- POISSON or EXPRND functions with a large mean: Because these functions use a recursive technique to calculate the next number in the sequence, large values for the mean can slow the execution of the simulation and can

require deceptively large amounts of memory.

## **Memory-Expanding Options**

- Save your work early and often. In so doing, you will minimize the amount of work you will lose if your machine ever “crashes.”
- Periodically clear the data displayed in comparative graphs and/or tables. Choose Restore Graphs and Tables from the Interface or Model menu to clear the data. This will free a corresponding amount of memory in RAM.
- Get more RAM. These days, RAM is perhaps the most cost-efficient computer investment you can make. It’s cheap. It’s readily available. And, it’s easy to install. It’s probably the best way to relax memory constraints.

 [Related Topics](#)

# Speeding up Simulations

Simulation speed is dependent on several factors in your models. If you choose a long simulation time or a small value for DT the number of calculations the software has to do increases. Also, as your model increases in complexity, as measured in number of Stocks, Flows, and Converters use in the model, the software will have more calculations per time step. Here are some things you can do to increase simulation speed in your models:

- Close all graphs and tables while the simulation is running. Then open them up after the simulation is complete to review the results. This removes the need for the software to redraw the changing graphs and tables during the run.
- Make sure Animation is turned off (click on each of the Stock, Flow, and Converter icons, in the Model Prefs under the Model Menu, to remove the box around the icon). This will remove the need to redraw the changes to the diagram during the run.
- Limit the use of numeric displays. The software will need to update each numeric display at each time step and this can slow down the operation.
- Visit the Run Specs dialog and ensure that the Sim Speed box is set to 0. Values larger than 0 will cause the simulation to slow down.
- If you observe that your computer has to use Virtual Memory to increase RAM for running the simulation, buying more RAM will speed things up. Generally you can hear the hard drive in use during the run and this would be the signal that Virtual Memory is required to run the model.

 [Related Topics](#)

# Reducing File Size on Disk

If you have a large number of graphs and tables or make use of comparative graphs and tables, you will find that the model size increases when you save your model following a simulation. To combat this increase in file size, choose Restore Graphs and Tables under the Model or Interface menus. This will clear all stored data from table and graph pads, and thereby will reduce the size of file on disk.

 [Related Topics](#)

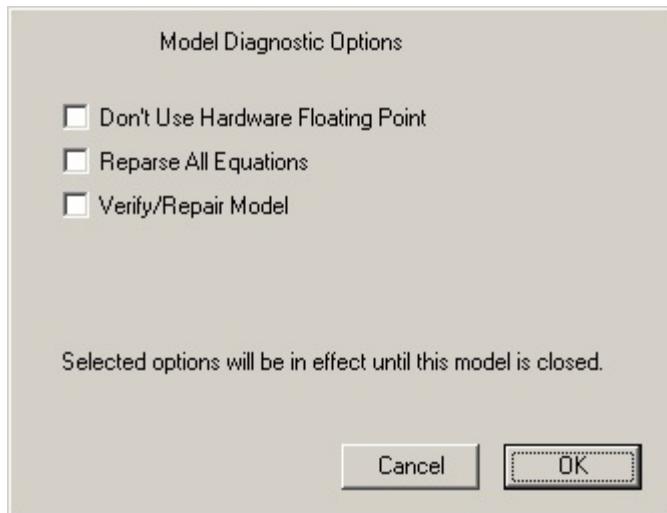
# Using Model Diagnostics

When you save a model, the software keeps track of the diagram layout, equations, table and graph data, and other important information by storing lists within the model file. Occasionally a glitch or some other interruption to the save process can cause these lists to become out of sync. Although this is very unusual, if you suspect that this may have occurred, you can check for and usually correct such damage using the software's Model Diagnostics feature.

Before using any of the Model Diagnostics features, we urge you to make a backup copy of your model.

# To open a model in Model Diagnostics mode

1. Press the SHIFT key while opening the model. The Model Diagnostic Options dialog box opens.



2. Select the diagnostic options that you want:

- **Reparse All Equations** - Select this check box to force the software to examine all the equations in the model. If, for some reason, the model contains an invalid equation that does not have a question mark on its corresponding building block, a question mark will be added so that you can locate and correct the offending equation.
- **Verify/Repair Model** - Select this check box to see if there are any errors in the model, and also to see the total count of all entities in the model.

---

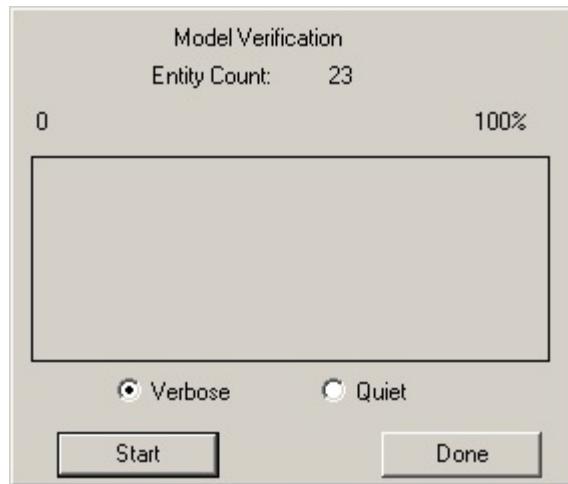
Note: The **Don't Use Hardware Floating Point** option does not work in this version of the software.

3. Click **OK**. The options you selected are in effect until you close the model.

# To check for errors in the model

Use this procedure to have the software examine all of the equations in the model, report any errors it finds, and correct any errors it can.

1. Press the SHIFT key while opening the model. The Model Diagnostics Options dialog box opens.
2. Select the **Verify/Repair Model** check box.
3. Click **OK**. The model opens.
4. From the Run menu, choose **Verify/Repair Model**. The Model Verification dialog box opens.



5. Select whether you want to run the diagnostics in verbose or quiet mode:
  - **Verbose** – Select this option to have the software display a message every time it encounters an error.
  - **Quiet** – Select this option to have the software report all errors at the end of the verification process.
6. Click the **Start** button. The software scans through the structure of the model. If you selected **Verbose** mode and an error is encountered, the Model Verification dialog box displays a message describing the error. Click the **Continue** button to repair the error and continue with the diagnostic process. If you selected **Quiet** mode, any error messages are displayed at the end of the verification process.
7. When the verification process is complete, click **Done**.
8. If any errors were detected, you should save your model with a new name (as a precaution, the software will not allow you to overwrite the original model).
9. To confirm that the problems have been corrected, close and reopen the new model file and then run it.

If the procedure highlighted any errors that could not be corrected, or if you continue to experience problems loading or running your model, we recommend that you contact our Customer Services Group for assistance.

# To view the total number of entities

An entity is a component in your model (Stock, flow, converter, etc.). If you have an “arrayed” component, each element of the array counts as an entity. It can be helpful to see the total number of entities if you are having problems with your model and suspect that the problems are caused by approaching model size limitations.

1. Press the SHIFT key while opening the model. The Model Diagnostics Options dialog box opens.
2. In the Model Diagnostics Options dialog box, select the **Verify/Repair Model** check box.
3. Click **OK**. The model opens.
4. From the Run menu, choose **Verify/Repair Model**. The Model Verification dialog box opens and display the total number of entities in the model at the top of the dialog box.
5. Click **Done**.

 [Related Topics](#)

# Getting Help from isee systems

## Online Resources

isee systems, inc. provides technical assistance to all users through free resources available on our Website. Visit [www.iseesystems.com](http://www.iseesystems.com) to access frequently asked questions, free pre-built models, and Systems Thinking based articles.

## Contacting isee systems, inc. Technical Support

Live Technical Support via phone and email is available to end users who have active support plans.

- To receive help via email, submit your inquiry through the Technical Support Request form on the isee systems Website: [www.iseesystems.com](http://www.iseesystems.com)
- To receive help via phone, contact isee systems Technical Support at: 603-448-4990. Please have your software registration number ready.

 [Related Topics](#)

# Copyright Information, Trademarks and Conditions of Use

**STELLA** software and ***iThink*** software © 1985-2009 isee systems, inc. All rights reserved.

**STELLA** and ***iThink*** Technical Documentation © 1997-2009 isee systems, inc. All rights reserved.

Mac2Win Porting Technology © 1990-2008, Altura Software, Inc. All rights reserved.

Mac2Win Porting Technology™ is licensed from Altura Software, Inc. It is against the law to copy the Mac2Win software for distribution or use by a third party without a valid license agreement from Altura Software, Inc.

It is against the law to copy the **STELLA** software, or the ***iThink*** software for distribution without the prior written consent of isee systems, inc. Under the law, copying includes translation of the software into another language or format. Licensee agrees to affix to, and present with, all permitted copies, the same proprietary and copyright notices as were affixed to the original, in the same manner as the original.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, printed, recording, or otherwise, without prior written permission from isee systems, inc.

**STELLA** and ***iThink*** are registered trademarks of isee systems, inc. Macintosh is a trademark of Apple Computer, Inc. Windows is a trademark of Microsoft Corporation. Other brand names and product names are trademarks or registered trademarks of their respective companies.

isee systems, inc's Licensor makes no warranties, express or implied, including without limitation the implied warranties of merchantability and fitness for a particular purpose, regarding the software. isee systems, inc's Licensor does not warrant, guaranty, or make any representations regarding the use or the results of the use of the software in terms of its correctness, accuracy, reliability, currentness, or otherwise. The entire risk as to the results and performance of the software is assumed by you. The exclusion of the implied warranties is not permitted by some states. The above exclusion may not apply to you.

In no event will isee systems, inc's Licensor, and their directors, officers, employees, or agents (collectively isee systems, inc's Licensor) be liable to you for any consequential, incidental, or indirect damages (including damages for

loss of business profits, business interruption, loss of business information, and the like) arising out of the use of, or inability to use, the software even if isee systems, inc's Licensor has been advised of the possibility of such damages. Because some states do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitations may not apply to you.

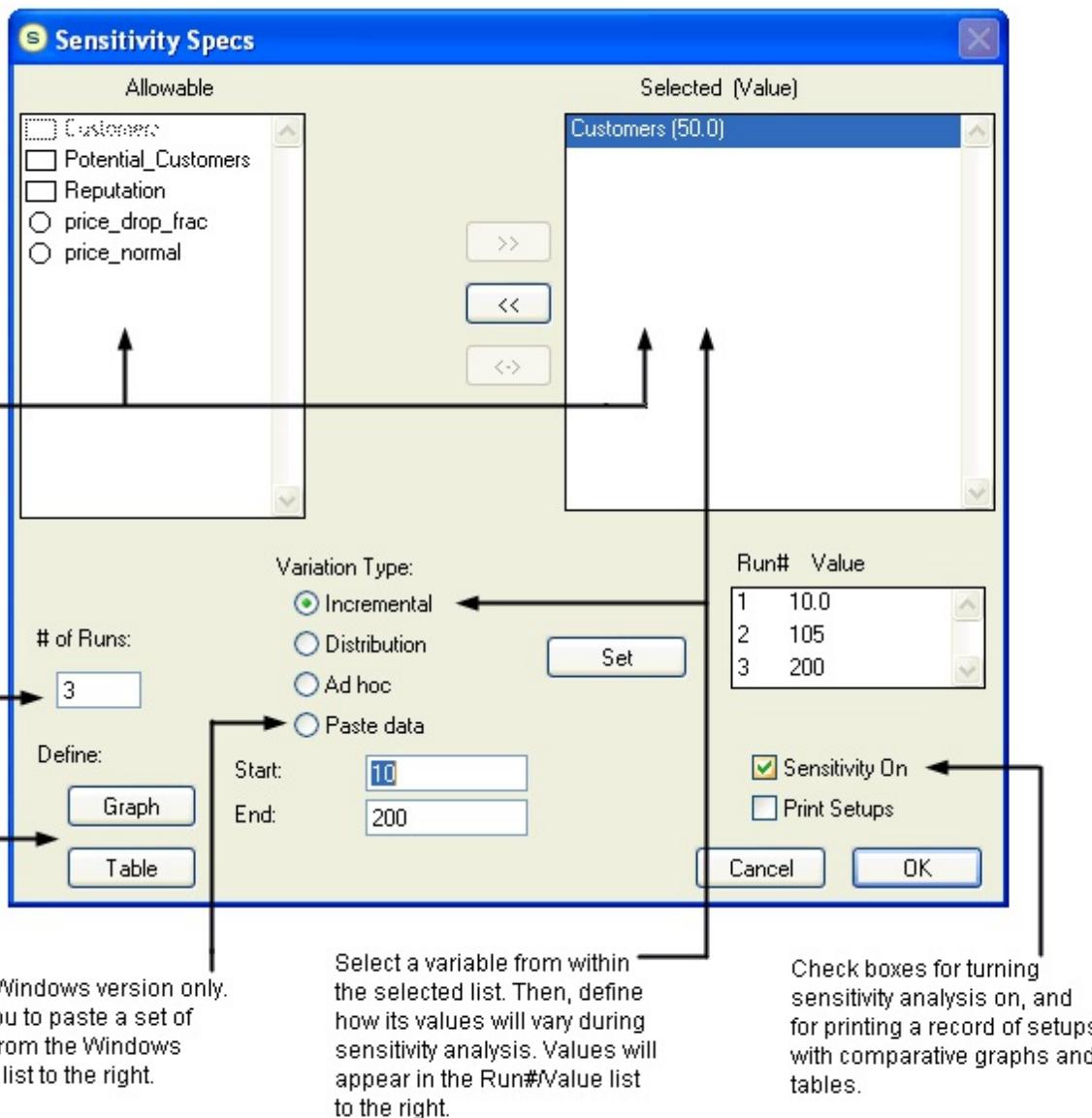
May 2009

***iThink*** and **STELLA** version 9.1.3

# Sensi Specs

The Sensi Specs command opens the Sensitivity Specs dialog box. This dialog enables you to establish the characteristics for a set of sensitivity runs. It also enables you to "turn on" sensitivity analysis.

Figure 3-23 Sensi Specs Dialog



The top of this dialog contains an Allowable list and a Selected list, as well as arrowed Enter (>>), Remove (<<), and Replace (<->) buttons. In the middle of the dialog is a box for specifying the number of sensitivity runs you want to make, a set of controls for varying a selected variable in the Selected list, a Set button, and a list of values associated with the selected variable. In the bottom left of the dialog are buttons which automatically take you to the Graph Pad and Table Pad objects when you leave the dialog. In the bottom right is a check box for turning Sensitivity On on, a check box for printing sensitivity setups along with comparative graphs and tables, a Cancel button,

and an OK button.

Entering, removing, and replacing variables works exactly as it does in the [Define Graph...](#) and [Define Table... dialogs](#). Simply select the target variable(s), and click the appropriate button between the Allowable list and Selected list. Once in the Selected list, variables are available to have their values changed from run to run in the sensitivity analysis. Note that when you enter a variable into the Selected list, its current initial value is shown within parentheses.

When you select any variable that has been entered into the Selected list, you can then define how its values will vary for the purpose of sensitivity analysis. You have three choices in the Macintosh version: Incremental, Distribution, and Ad hoc; and a fourth in the Windows version: Paste data. Once you have selected the variation type you wish to use, you will be prompted to provide specifics of the variation. Different Variation Types are shown in Figure 3-24.

*Figure 3-24  
Sensitivity Analysis Variation Types*

Variation Type:

- Incremental
- Distribution
- Ad hoc
- Paste data

Start:

End:

For Incremental variation, type in values for Start and End. Then click the "Set" button. The software will assign the appropriate intermediate values.

Variation Type:

- Incremental
- Distribution
- Ad hoc
- Paste data

Ad hoc Values:

For ad hoc variation, type in an ad hoc value and then click the "Set" button. Repeat this procedure to specify all the values you want to use.

Variation Type:

- Incremental
- Distribution
- Ad hoc
- Paste data

Mean:

S.D.:

Seed:

To specify a normal distribution for sensitivity analysis, type in the mean, the standard deviation, and a randomizing seed. Then click the "Set" button. To toggle to a uniform distribution, click the bell curve button.

Variation Type:

- Incremental
- Distribution
- Ad hoc
- Paste data

Min:

Max:

Seed:

To specify a uniform distribution for sensitivity analysis, type in the minimum, the maximum, and a randomizing seed. Then click the "Set" button. To toggle to a normal distribution, click the distribution button.

For distributions, the seed must be a non-negative integer. A seed of 0 will cause the software to choose a random seed, which means the sequence of random numbers will not be repeatable if you ever change the setup for the variable. When you use a positive integer for the seed, you ensure the ability to replicate a particular random number sequence in subsequent sensitivity runs.

Variation Type:

- Incremental
- Distribution
- Ad hoc
- Paste data

Run#	Value
1	2.00
2	4.00
3	6.00

#### Paste data (Windows only):

To use this option, you must have first copied a numeric data set (comma-separated values) to the Windows clipboard. Then select the Paste data option. A "Paste" button appears to the right of the option. Click the "Paste" button to paste the data into the Run#/Value list. The number of runs will be automatically set to match the number of values pasted in.

When you change the "# of Runs" you wish to make, the values of the variables that you have specified as candidates for variation will automatically be changed (except in the case of ad hoc specification, in which case the last value specified will be used for any additional runs you specify). "# of Runs" cannot be selected after copying data from the clipboard with the Paste data option (Windows) because that number is determined by the number of values pasted into the sensitivity setup.

**Tip:** If you have created a sensitivity setup, depressing the alt key (Windows) or the option key (Macintosh) as you choose the Run menu will change the display of the Sensi Specs... item to Disable Sensi Run or Enable Sensi Run. Thus, by depressing the alt or option key while choosing the Sensi Specs... item from the Run menu, you can toggle between Sensitivity On and Sensitivity Off without entering the Sensi Specs... dialog.

# **Default Settings**

The Default Settings command allows you to customize the default user interface and simulation settings. Changes made to Default Settings take effect immediately for all new models. In current or previously saved models, changes made to Default Settings are either global or local in nature. Global Default Settings take effect immediately for the current model, but will not affect any previously saved models. Local Default Settings take effect immediately for all additions to current and previously saved models, but will not affect existing model entities. Figures 3-6 through 3-11 indicate which settings are global and which are local.

Within Default Settings, you can specify defaults for the following arenas:

- Model (Figure 3-6)
- Object (Figure 3-7)
- Stock/Flow/Converter (Figure 3-8)
- Table/Graph (Figure 3-9)
- Text box (Figure 3-10)
- Run Specs (Figure 3-11)

*Figure 3-6 Default Settings - Model*

Use this check box to show page numbers and boundaries on the Interface and Map/Model levels as the default setting. This is a global setting.

Use this check box to establish a one-to-one correspondence between the High-Level Map on the Interface level and the Map/Model level as the default. This is a global setting.

This Default Setting, when checked, causes a simulation run to stop and an alert message to pop up when division by zero occurs. This is a global setting.

This check box regulates access to the Interface level. If the box is checked, the Interface level is accessible. If unchecked access to the Interface level will be unavailable. This is a global setting.

Checking this box will cause a unit analysis of all model entities when you click "OK" and exit an equation dialog. If there is any unit inconsistency, the software will place a "!" on the inconsistent icon on the diagram. You must then resolve the inconsistency before you can run the model.

Checking this box activates tooltip functionality within model entities during a pause in a running simulation. With the model paused, hover the cursor over model entities to see their current value.

Use this pop-up menu to select the group of features for which you are to establish default settings.

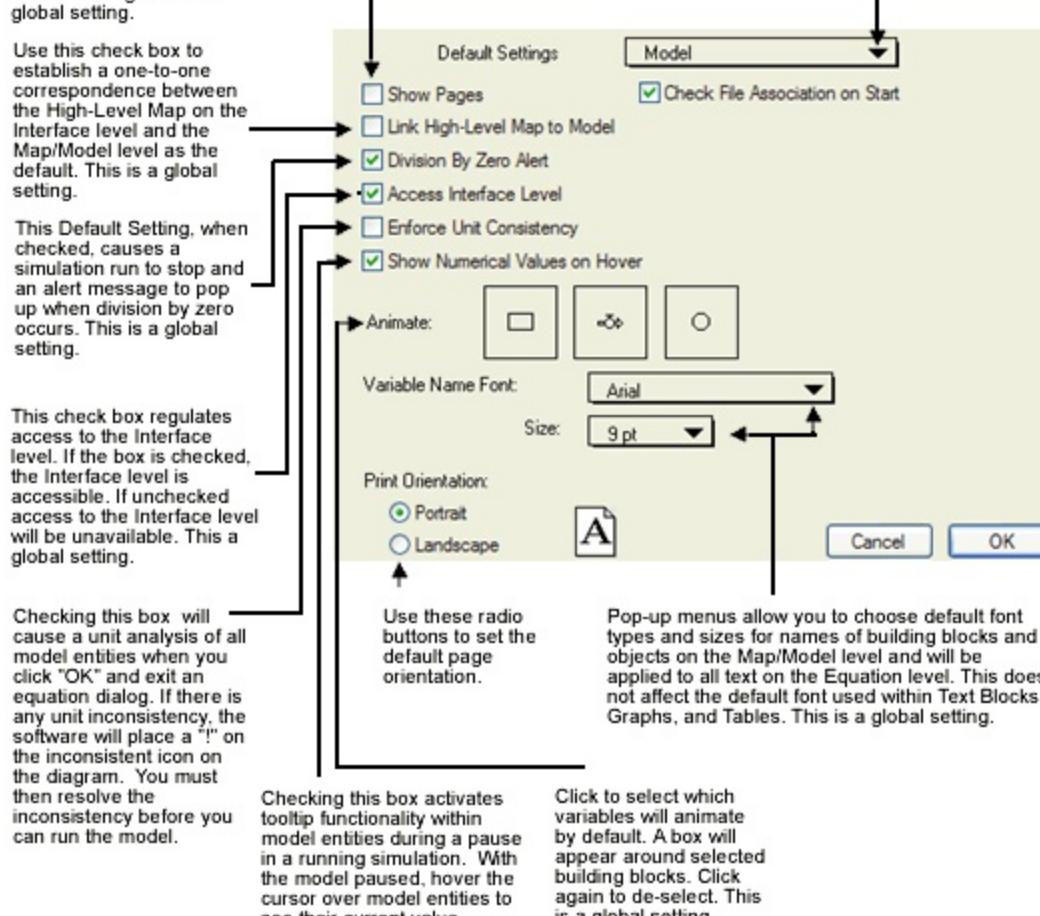
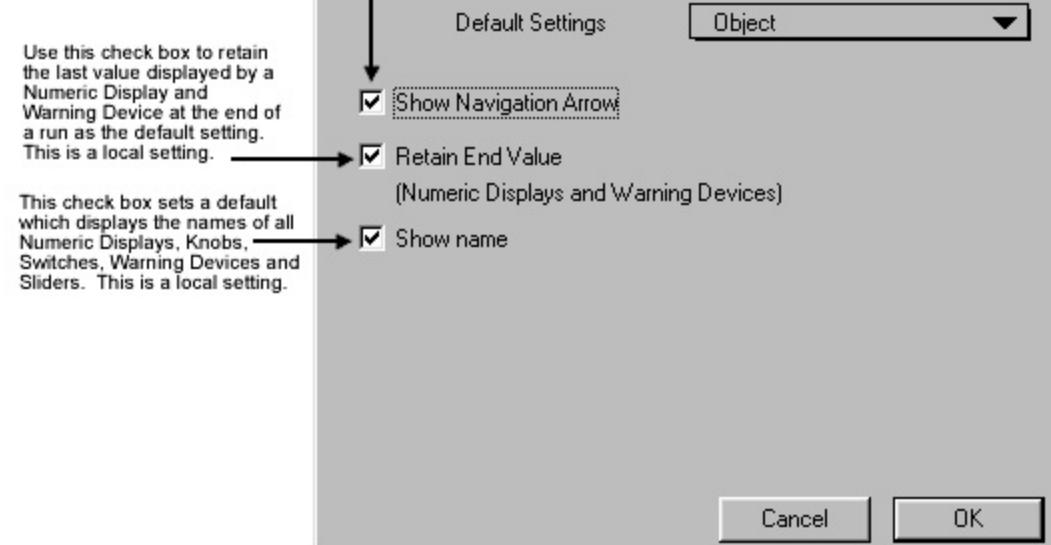


Figure 3-7  
Default Settings - Object

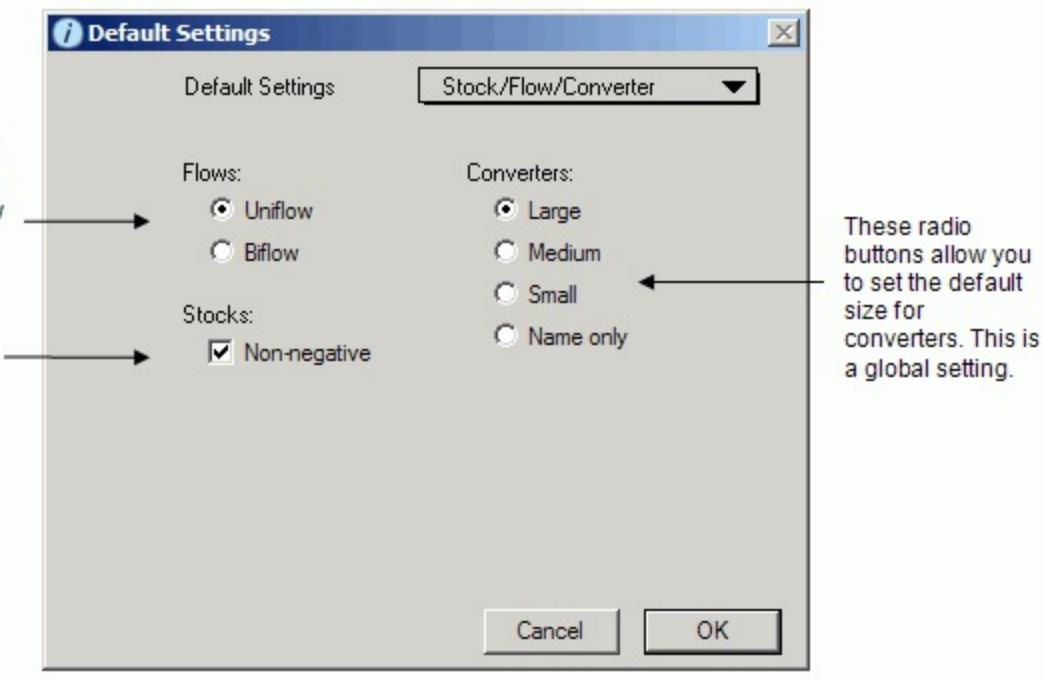
Use this check box to retain the last value displayed by a Numeric Display and Warning Device at the end of a run as the default setting. This is a local setting.

This check box sets a default which displays the names of all Numeric Displays, Knobs, Sliders, Warning Devices and Switches. This is a local setting.

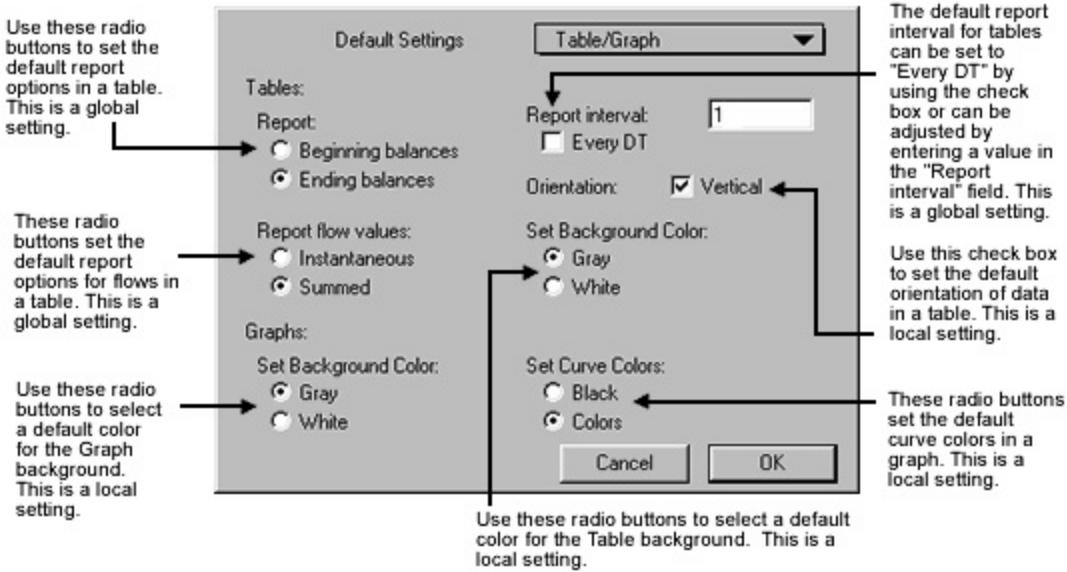
Use this check box to show the Navigation Arrow of the Knob, List Input Device, Slider, Graphical Input Device, Numeric Display, and Warning Device to the corresponding building block as the default setting. This is a local setting.



**Figure 3-8**  
**Default Settings - Stock / Flow / Converter**



**Figure 3-9**  
**Table / Graph**



**Figure 3-10**  
**Default Settings - text box**

Pop-up menus let you set default font and size for the text. This is a local setting.

Radio buttons allow you to set default borders for text blocks. This is a local setting.

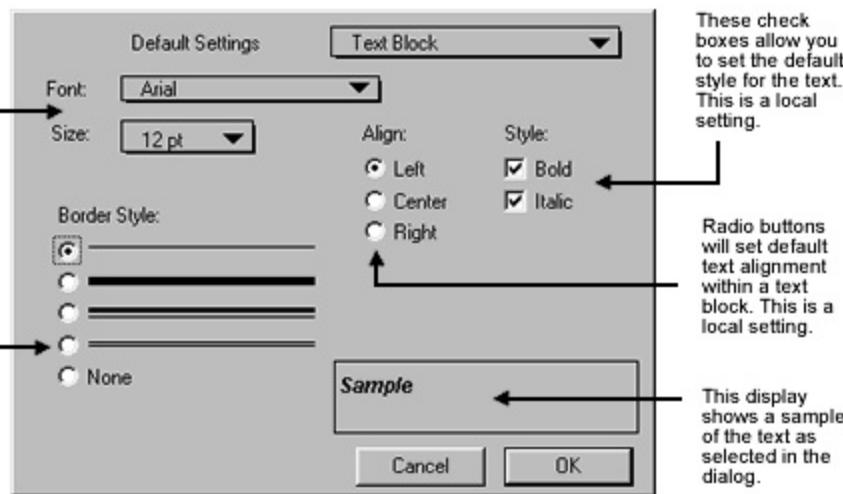
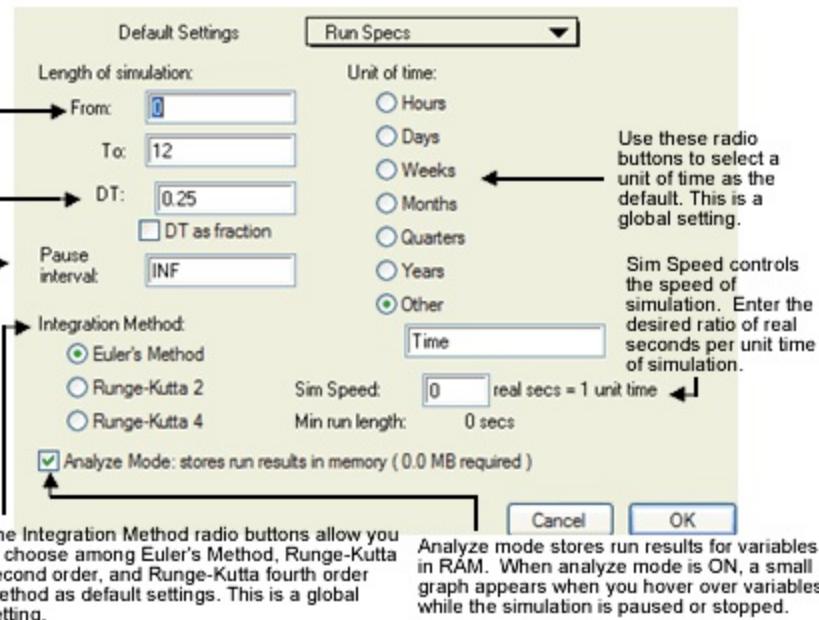


Figure 3-11  
Default Settings - Run Specs

Indicate the default length of simulation by setting values for "From" and "To." This is a global setting.

Set the default simulation time step (DT) by adjusting the number or by checking the check box marked "DT as fraction." This is a global setting.

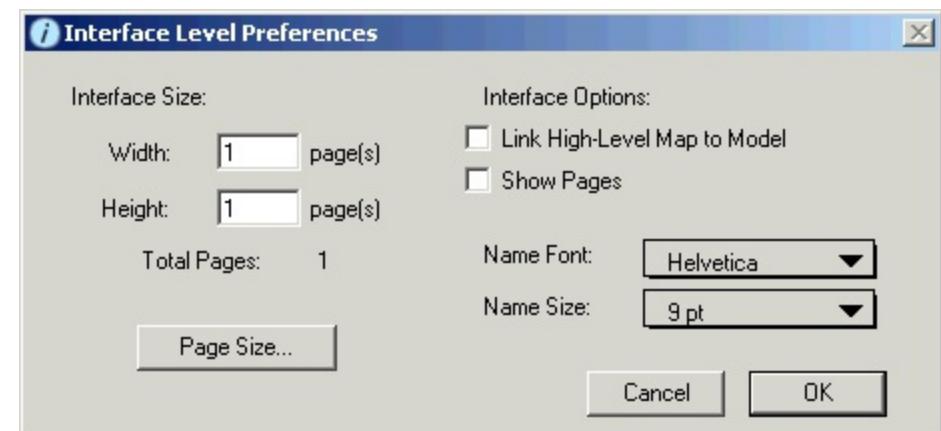
Model simulations will pause by default according to the interval you specify. This is a global setting.



# Interface Preferences

The Interface Prefs command (available on the Interface menu when you are viewing the Interface layer) displays the Interface Preferences dialog box. Use the Interface Preferences dialog box to set the visual characteristics of the Interface layer. The Interface Level Preferences dialog is shown in Figure 3-16.

Figure 3-16 Interface Level Preferences Dialog



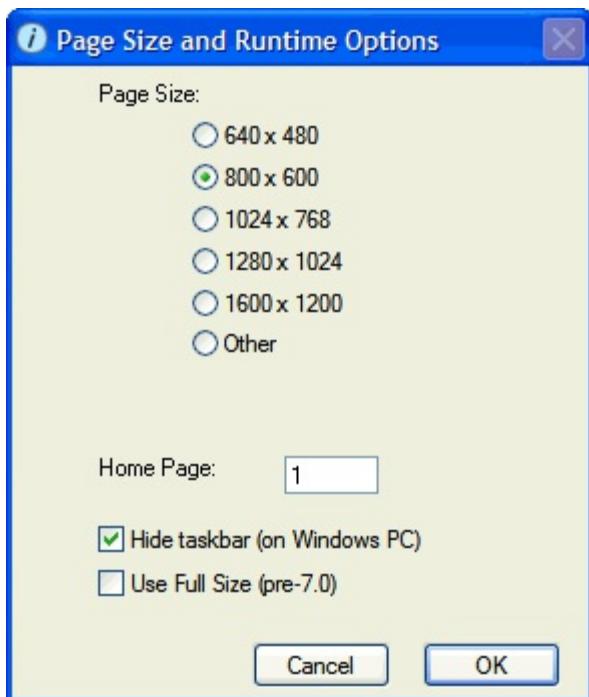
## **Interface Size**

Use the Width and Height boxes to specify the size (in pages) of the Interface Layer. The Total Pages value updates to indicate the total number of pages you've specified.

## Page Size button

Click this button to select the screen resolution and other page-size related options in the Page Size and Runtime Options dialog box, as shown in Figure 3-16a.

Figure 3-16a  
Page Size and Runtime Options dialog box



- **Page Size** – If you selected the **Show Pages** check box, use these options to select the page size for pages in your model. This option determines how the pages will be formatted based on the screen resolution you select. In general, we recommend that you select the **800 x 600** option, unless you are sure all of your model viewers will be using a different screen resolution. Select the **Other** option to use the current printer page settings or to enter a custom width and height in pixels.
- **Home Page** – Use this box to specify the page number of the Home Page in your model. The Home Page is the page to which users will be navigated when they click a Home ( ) navigation button on the model's Interface layer.
- **Hide taskbar (on Windows PC)** – If you will be saving this model as an isee Runtime file, select this check box to hide the Windows taskbar when users view the model with isee Runtime on a computer running windows.
- **Use Full Size (pre-7.0)** – Select this check box to specify the size of the pages on the Interface instead of the device (screen or printer). If you will be saving this model as an isee Runtime file and the model was created with a version of the software that is older than version 7, you can select

this check box to preserve the pre-7.0 sizing (which is slightly larger than the current version's page sizes).

## **Link High-Level Map to Model**

Select this check box to enforce a one-to-one correspondence between High Level mapping entities on the Interface layer and entities on the Map and Model layers. These include Process Frames, Bundled Flow, Bundled Connectors, and their corresponding Map and Model layer counterparts (i.e., Sectors and their connecting Flows and Connectors). When you select this check box, the software will create the corresponding structures automatically on the Interface layer as you construct the model on the Map and Model layers. If this check box is not selected, there is no logical connection enforced between the levels, and any previously linked High-Level map will be deleted.

## Show Pages

Select this check box to show page numbers and page boundaries (displayed as dashed lines) on the Interface layer.

---

**Note:** If you select this check box and have navigation buttons that are using Camera Navigation, the buttons will navigate users to the top-left of the page where you set the navigation (by clicking the camera icon) rather than the exact location on the page that you selected when you defined the button. For more information about defining navigation buttons and using Camera Navigation option, see the "[Navigation](#)" section in [Button Dialog Operations](#).

---

## **Name Font**

Select the font style for labels that appear on the Interface layer.

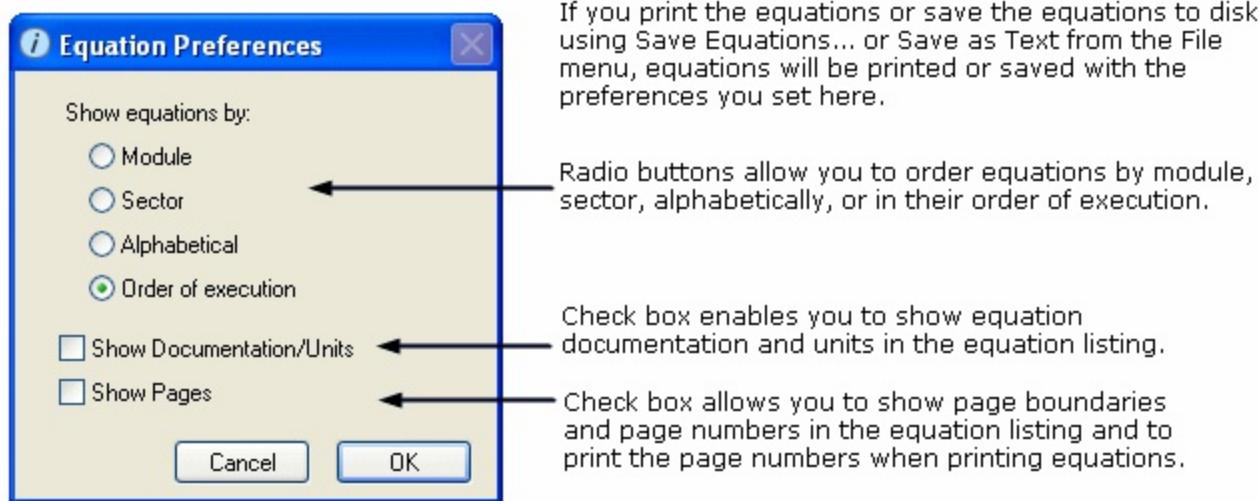
## **Name Size**

Select the font size for labels that appear on the Interface layer.

# Equation Preferences

The Equation Prefs command (available on the Equation menu when you are viewing the Equation layer) displays the Equation Preferences dialog. Use the Equation Preferences dialog to set the visual characteristics of the Equations list. The Equation Preferences dialog is shown in Figure 3-18.

Figure 3-18 Equation Preferences Dialog



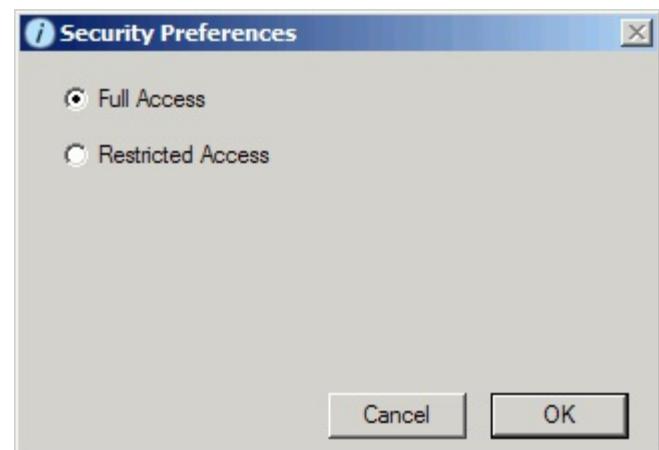
If you want to create a different web interface for your **iThink** or **STELLA** model and publish it to the Internet using Forio Broadcast simulation hosting service, follow these steps:

1. In the Equation Preferences dialog, select Show equations by **Order of execution**.
2. From the File menu, choose **Save As Text**.
3. Visit <http://www.folio.com/isee>.
4. Sign up or Log in to your Forio Broadcast Express or Broadcast Pro Account
5. Click on the **Import/Edit Model >> Model Import Wizard** menu item.
6. On the Import Model page, click the **Browse** button, select the text file that you created, and click **Next**
7. Follow the instructions in the next set of screens to complete your model import to Forio Business Simulations.

# Setting Security Options for a Model

The **Lock Model** command allows modelers to control access to their models. The default setting is **Full Access**. When you choose Lock Model from the File menu, the Security Preferences dialog box appears.

*Figure 3-4 Security Preferences Dialog*



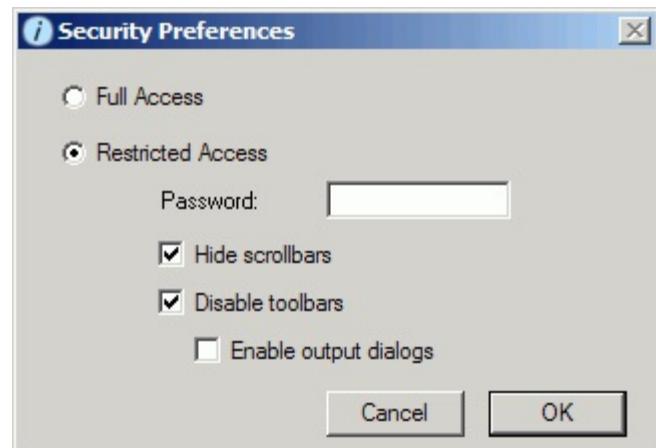
## **Full Access**

The default setting. Provides complete access to all aspects of the model.

## Restricted Access

Choosing Restricted Access allows you to password protect your model. When you select this option, options for specifying a password and further restricting access appear. Figure 3-5 shows the options available when you select the Restricted Access option.

*Figure 3-5  
Restricted Access*



## **Password**

Enter your password in this box, keeping in mind the following rules:

- Always write down the password you enter in case you forget it!
- The password can be as long as 256 characters or as short as one, including all keyboard characters. Passwords are case sensitive.
- By not entering a password, you allow unrestricted access to the Security Preferences dialog, allowing users to change the level of security.
- Restrictions take effect immediately upon clicking OK to close the dialog. Subsequent access to the Security Preferences dialog requires the password.

## **Hide scrollbars**

Selecting this check box disables access to the scroll bars and other window controls, including navigation tabs, magnification controls, and the Run Controller. When this check box is selected, model users have a limited workspace in which to manipulate model structure or work with an interface control panel.

## **Disable toolbars**

Selecting this check box disables access to the toolbar and restricts access to the Interface and Mapping layers. Only documentation caches are available when a user double-clicks a Building Block.

## **Enable output dialogs**

Selecting this check box enables access to output device dialogs while in Restricted Access mode.

# Cut, Copy, Paste, and Clear Operations

The following table indicates which types of elements can be cut, copied, pasted, or cleared with the Cut, Copy, Paste, and Clear commands.

*Figure 3-13 Cut, Copy, Paste and Clear Operations*

ELEMENT	Cut	Copy	Paste	Clear	Notes
<b>Building Blocks</b>					
Process Frame	no	no	no	yes	
Graphic to selected process frame	no	no	yes	no	bitmap or Mac PICT format
Bundled Flow	no	no	no	yes	
Bundled Connector	no	no	no	yes	
Stock	yes	yes	yes	yes	
Flow	yes	yes	yes	yes	
Converter	yes	yes	yes	yes	
Graphical function output column	no	yes	yes	no	data only; data link client/subscriber capable; use command/Ctrl keys to execute commands within dialogs
Connector	yes	yes	yes	yes	need entities on either side for cut, copy, and paste
Equations within dialogs	yes	yes	yes	yes	text only; data link client/subscriber capable; use command/Ctrl keys to execute commands within dialogs
Documentation within dialogs	yes	yes	yes	no	text only; use command/Ctrl keys to execute commands within dialogs
Ghosts of building blocks	yes	yes	yes	yes	
<b>Equation Listing</b>					
<b>Objects</b>					
Sector Frame	yes	yes	yes	yes	
Graphic to selected Sector Frame	no	no	yes	no	bitmap or Mac PICT format
Documentation w/in sector dialog	yes	yes	yes	no	text only
text box	yes	yes	yes	yes	
Text within Text Box	yes	yes	yes	yes	text only

Graph/Loop Pad/Table icon	no	yes	no	yes	copy/paste will operate on icon only; clear deletes entire pad
Unpinned Graph Pad page	no	yes	no	no	picture of graph is copied to clipboard
Selected variables w/in Graph	no	no	no	yes	
Unpinned Loop Page	no	no	no	no	copy/paste will operate on icon only; clear deletes entire pad
Selected variables w/in Table	yes	yes	no	yes	hold option/alt key to copy variable name along with data; data link is server/publisher capable
Selected time slices w/in Table	no	yes	no	no	hold option/alt key to copy time along with output data
Knob, Switch, Numerical Display, LID, GID, Warning Device, Slider, Graphics Frame, Button	yes	yes	yes	yes	after paste, input devices must be re-configured.

# Model Preferences

The Model Prefs command (available on the Model menu when you are viewing the Map or Model layer) displays the Model Preferences dialog. Use the Model Preferences dialog to set the visual characteristics of the model diagram on the Map and Model layers. The Model Preferences dialog is shown in Figure 3-17.

*Figure 3-17 Model Preferences Dialog*



Each option in the Model Prefs dialog is described below.

## Diagram Size

Adjust the width and height values to change the size (in pages) of the Model.

## **Page Sequence**

Use these radio buttons to determine page sequencing.

## **Diagram Grid**

Adjust the width and height values to change the size of the Diagram Grid. Then, choose the Align to Grid command from the Model menu to align selected Model elements to the grid.

## Options

- **Show pages** – Select this check box to show page numbers and boundaries on the diagram and to print page numbers on the diagram.
- **Disable Posters** – Select this check box to disable any message posters built into your model.
- **Disable Gray Drawing** – When Sub-models are present, selecting this check box will cause main level building blocks to be drawn in a solid color (rather than in a gray scale) whenever one or more Sub-models are open.
- **Hide Poster Titles** – Select this check box to hide poster titles.
- **Division by Zero Alert** – Select this check box to cause an alert message to pop up when division by zero occurs and to reveal the identity of the offending model variable. When this check box is not selected, the model will continue to generate data when division by zero occurs.
- **Opaque Decision Diamonds** – When Sub-models are present, selecting this check box causes opaque backgrounds to be drawn for all Sub-model spaces. If you do not select this check box, the background will be transparent.
- **Enforce Unit Consistency** – Select this check box to enforce a unit analysis of all model entities when you click OK to exit an equation dialog or when you [check unit consistency](#) for the model. If there is any unit inconsistency, the software will place a "!" on the inconsistent icon in the diagram. You must then resolve the inconsistency before you can run the model.
- **Show Numerical Values on Hover** – Select this check box to activate Tooltip functionality for model entities when a model simulation is paused or stopped. With the model paused or stopped, roll the cursor over model entities to see their current value. In addition, small graphs will be displayed for all entities that have been loaded into table pad or graph pad pages. To display hover graphs for all model variables, select the Analyze Mode check box in the Run Specs dialog.
- **Update Numeric Displays Once per Time Period** – Select this check box to cause numeric displays to update once per simulation time period. Checking this box minimizes the frequency of screen redraw as a simulation progresses, thus speeding the unfolding of simulation results. Recommended on the Macintosh platform when there are multiple numeric displays in a simulation control panel on the Interface layer.
- **Save Run Output to File** – Select this check box to have the software save a model's output for all variables for all time periods to an external

file that can then be opened by spreadsheet and other applications. This makes it possible to do post-processing of model output using, say, statistical routine. It also means if you neglected to enter a variable into a Graph or Table Pad page prior to doing a simulation, you may chart the variable without having to re-run a simulation.

- **Use Lettered Polarity** – Select this check box to have the software display s/o labels rather than +/- labels when polarity is shown.
- **Disable Discrete Animation** – Select this check box to disable the discrete animation for Conveyors and Queues.
- **Name Only Modules** – Select this check box to have the software display module names only, instead of both module names and icons. This option is useful when you are creating [Causal Loop Diagrams](#) that contain modules.

## Converters

Use these radio buttons to control the size of the Converters on the diagram.

## **Animate**

Click an icon to select variable types for animation. A box will appear around selected types. Click again to de-select.

## **Name Font / Name Size**

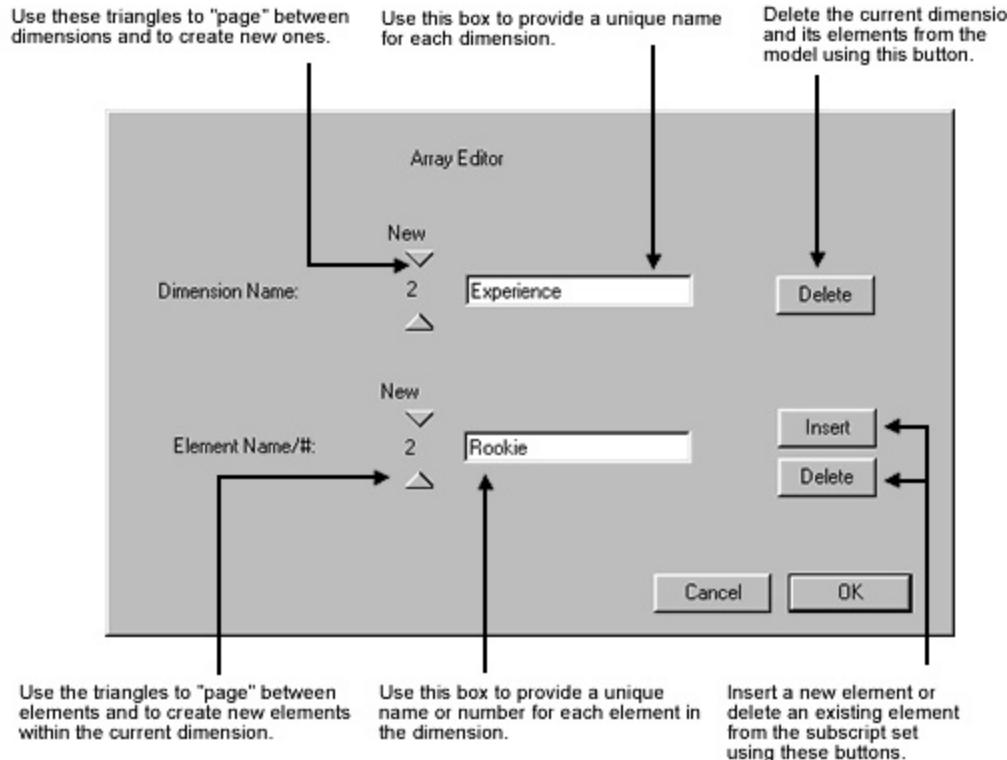
Pop-up menus allow you to choose different font types and sizes for the names of building blocks and objects.

# Array Editor

Use the Array Editor dialog (shown in Figure 3-19) to create dimensions and elements which you can then use to create arrayed variables. The Array Editor dialog appears when you choose Array Editor from the [Interface, Model, or Equation menu](#).

For a detailed treatment of array capabilities, see [Introduction to Arrays](#).

Figure 3-19 Array Editor



# Run Command Configurations

Figure 3-21 shows the different permutations of the Run command.

*Figure 3-21 Different Configurations of the Run Command*

If you see this	Run Selected Sectors is	Run Selected Modules is	Sensitivity is	Model is
Run	off	off	off	stopped
Resume	off	off	off	paused
Run Sector(s)	on	off	off	stopped
Resume Run Sector(s)	on	off	off	paused
S-Run	off	off	on	stopped
Resume S-Run	off	off	on	paused
S-Run Sector(s)	on	off	on	stopped
Resume S-Run Sector(s)	on	off	on	paused
Run Module(s)	off	on	off	stopped
Resume Run Module(s)	off	on	off	paused
S-Run Module(s)	off	on	on	stopped
Resume S-Run Modules(s)	off	on	on	paused

# Sector Specs

The Sector Specs command opens the Sector Specs dialog. When the sector tool has been used to define model sectors, the Sector Specs dialog will enable you to specify the model sectors which will be run in a simulation. The software shows a maximum of 120 sectors in a given model, when the main monitor of your computer is 13 inches or larger. For smaller monitors, the software shows a maximum of 65 sectors. The Sector Specs dialog is illustrated in Figure 3-22.

Figure 3-22 Sector Specs Dialog



As the Figure shows, the dialog contains two radio buttons. The default is to run the entire model. When Run Selected Sectors is activated, all model sectors which have been fully defined are available to be run. A fully defined sector will have all variables within it defined, all flows in and out of the sector defined, and all variables on originating ends of connectors coming into the sector defined. Simply check those sectors which you want to run.

Clear All unchecks all selected model sectors. Cancel exits the dialog without registering any changes. OK registers the changes you make, and then exits the dialog.

Whenever you run a model using the Run Selected Sectors option, the software will apply the following rules to isolate the selected sectors from the rest of the model:

- Building blocks which are entirely within a running sector will be run dynamically in the sector's simulation.
- Flows between running sectors will be run dynamically.
- Stocks, flows and converters which are not part of a running sector will be held constant at their initial values. Stocks and converters will report this initial value in Tables and in Numeric Displays.
- If the flow regulator is within a running sector, an outflow from a running sector (either to the model diagram or to a non-running sector) will be

treated as flowing into a cloud.

- If the flow regulator is outside of a running sector, an outflow from a running sector (either to the model diagram or to a non-running sector) will be held constant at its initial value unless it is defined by a variable within an active sector, in which case it could be dynamic. Flows from Queues will be set to zero.
- If the flow regulator is within a running sector, an inflow to a running sector (either from the model diagram or from a non-running sector) will be run according to its explicit flow logic unless it is dependent on variables outside the running sector(s), in which case the flow will be held constant at its initial value.
- If the flow regulator is outside of a running sector, an inflow to a running sector (either from the model diagram or from a non-running sector) will be held constant at its initial value, subject to any active (i.e., dynamically changing) logic coming from the running sector(s).

---

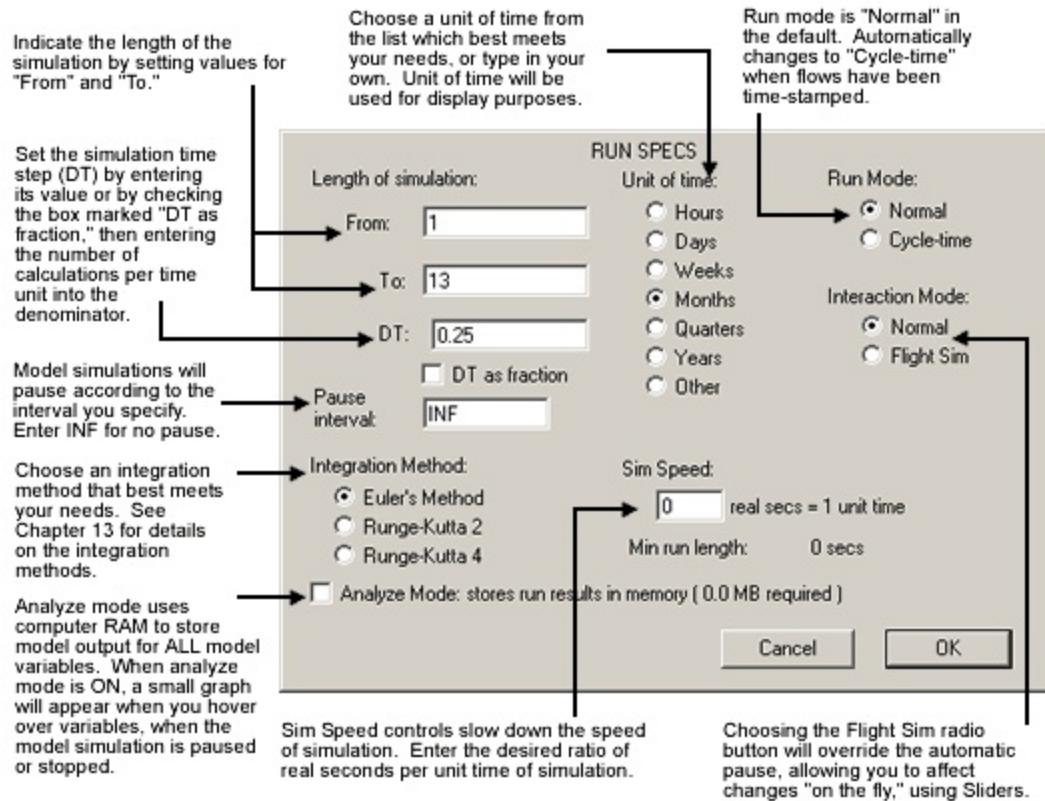
*Tip:* If you have selected a set of model sectors to be run in a sector-by-sector analysis, depressing the alt key (Windows) or the option key (Macintosh) as you choose the Run menu will change the display of the Sector Specs... item to Disable Sector Run or Enable Sector Run. Thus, by depressing the alt or option key while choosing the Sector Specs... item from the Run menu, you can toggle between Run Selected Sectors and Run Entire Model without entering the Sector Specs... dialog.

---

# Run Specs

The Run Specs command opens the Run Specs dialog box, which enables you to specify the simulation length, the time step between calculations (DT), the interval between simulation pauses, the integration method, the time unit for the model, and the run mode (Normal or Cycle-time) for the model. The Run Specs dialog is shown in Fig 3-25.

Figure 3-25 Run Specs Dialog



## **From / To**

Simulation length is defined by entering numbers for "From" and "To." "To" must always be greater than "From." If DT is greater than 1, and is not evenly divisible into the length of simulation, the software will choose the closest length of simulation which is a multiple of DT.

## DT

The Run Specs dialog also affords you the opportunity to set the simulation time step (DT). DT can be set by entering the desired time step directly. You can enter any number less than or equal to one or any integer greater than one. Alternately, you can check the "DT as fraction" box. Then set the number of calculations which will occur per unit time. In either case, choose a DT which is small enough to enable you to capture the shortest time frame of interest in your model. The trade-off here is speed versus accuracy. The more calculations the computer has to make per unit of simulation time, the longer it will take to complete any given simulation. For more about DT, see [Introduction to DT](#).

## **Pause Interval**

The default setting for Pause interval is infinity, designated by the letters INF on Windows machines or the infinity symbol (option-5 on your keyboard) on Macintosh machines. When the Pause interval is set at a value less than the length of the simulation, model simulation runs will pause as you have specified. Users then can inspect results. They also can change model constants, and then resume the simulation run by choosing Run from the Run menu. Thus, your models can be set up in a gaming mode for users.

## Integration Method

The Integration Method radio buttons allow you to choose among Euler's, Runge-Kutta second-order, and Runge-Kutta fourth-order methods. Briefly, use Euler's integration method unless you anticipate that your model will generate an oscillation. If you feel an oscillation is likely, and you are not using Queue, Conveyor or Oven stock types in your model, choose either RK 2 or RK 4 (try RK 2 first). For more information about integration methods, see [Introduction to Simulation Algorithms](#).

## Analyze Mode

Analyze mode uses computer RAM to store model output for ALL model variables. When analyze mode is ON, a small graph will appear when you hover over variables, when the model simulation is paused or stopped.

## **Unit of Time**

Within Run Specs, you can define the time unit for model simulations. Simply select one of the pre-defined time units, or select Other and type in your own.

## Run Mode

Radio buttons allow you to specify the run mode for your model as "Normal," or "Cycle-time." Cycle-time is turned on automatically whenever you time-stamp a flow in order to generate and collect cycle-time metrics. When operating in the Cycle-time mode, the integration method is set to Euler's. This setting cannot be altered, as long as Cycle-time is on. For more information about Cycle-time calculations and the workings of time-stamping, see [Introduction to Cycle-Time](#).

## **Interaction Mode**

Interaction mode gives you a choice between the Normal mode and Flight Sim mode. If you elect Flight Sim mode, the Pause interval box will disappear, and the Pause interval will gray. The options for causing the simulation to pause, when in this mode, are to choose Pause or Stop from the Run Menu. The posting of a message will also cause the simulation to Pause, even when in Flight Sim mode.

In Flight Sim mode, when you move a Slider Input, the simulation will continue running as it incorporates your input. This contrasts with the Normal interaction mode, in which clicking on a slider will pause the simulation. Note that Flight Sim mode is not available when Sensitivity Analysis is turned on. Flight Sim will always be turned off when Sensitivity Analysis is turned on.

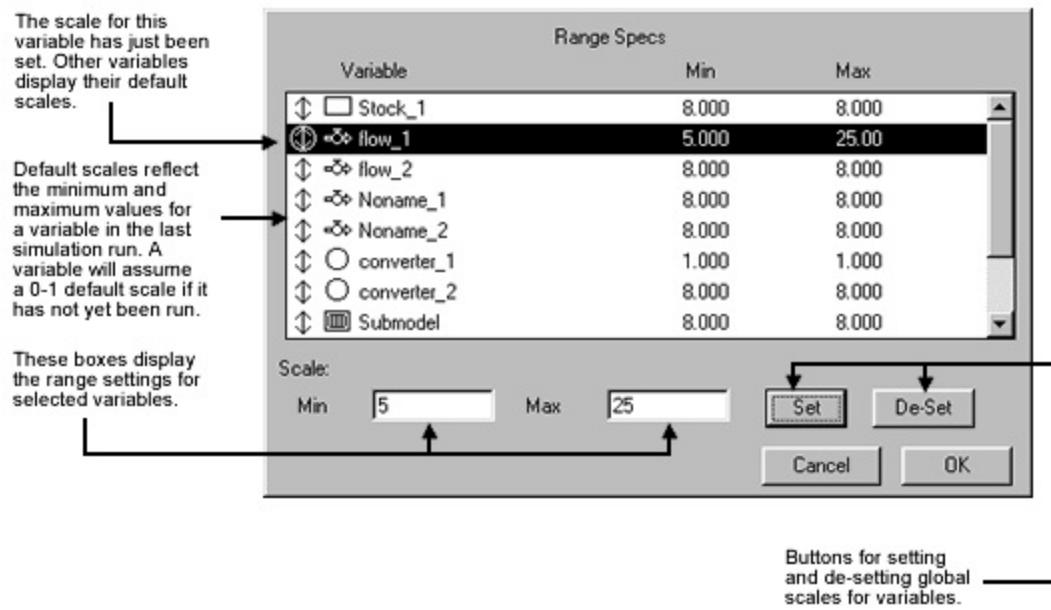
## **Sim Speed**

When Flight Sim mode is selected, the Sim Speed control will appear within the Run Specs dialog. This control allows you to slow down the speed of simulation to offer the user time to react to changes in the model. This feature cannot be used to increase the speed of simulation.

# Range Specs

The Range Specs command opens the Range Specs dialog box, which provides you with a listing of all model variables, and their most recent min/max global scales. The Range Specs dialog is shown in Figure 3-26. Within it, you can set the scale of model variables.

Figure 3-26 Range Specs Dialog



Note that variables may also be scaled from within the Define Graph dialog. For more information about scaling variables, see [Graph Pad Dialog Operations](#).

Model variables are initially assigned a default scale from zero to one. Once a simulation has been conducted, each variable will take on its actual min and max as its scale (unless you've already set its scale). These values appear in the scrollable list.

Setting the scale for one or more variables is easy. First, select the variable(s) in the "Variable" list. Then, type in the min and max values that you would like to have applied. Finally, click the "Set" button. A circle will appear around the arrows, to the left of each variable you have scaled. The scales you set in Range Specs are global, and will be used for plotting and animation purposes, until you change or remove them. To remove the scale that's been set for one or more variables, simply select the variable(s) in the "Variable" list. Then click on the "De-Set" button.

# Creating Equations for Units

When you create a unit of measure, you can define an equation that tells the software how to calculate the unit of measure.

In some cases, the equation for the unit of measure is the unit of measure itself. For example, the equation for "people" is "person" (or "people"). In other cases, however, you may need to define an equation by referencing other existing units and one or more operators, or by referencing other existing units, one or more operators, and one or more numerical values. For example, to create a unit of measure called "kilometers per day", you would define an equation ("km/day") that references two other existing units ("kilometers", which has an equation of "km", and "days", which has an equation of "day") with the operator "/". For a "cubic centimeters" unit of measure, the equation ("cm<sup>3</sup>") is created by referencing one existing unit of measure ("centimeters", which has an equation of "cm") and specifying that the equation should be cubed ("<sup>3</sup>"). To create a unit of measure that has an equation that references another unit of measure, you must first make sure that the referenced unit of measure exists (and create it if it does not).

To create equations, you must observe the following rules:

- Simplify all equations
- Use only one division ("/") in each equation
- Define equations so that they are calculated in the order you intend

These rules are described in more detail below.

# Simplify equations

Say you want to have a unit of measure called "people per person per year". If you wrote this equation on paper, you would write it this way:

$$\frac{\text{people}}{\text{person}} \\ \hline \text{year}$$

When typing this in the Equation (for Derived Unit) box, you might want to write it this way:

people/person/yr

however, unit-of-measure equations can have only one division, so the above equation won't be calculated correctly.

In order to define this equation for a unit of measure, you need to simplify the equation. In this case people/person really equals 1, so the simplified equation is 1/yr.

# Use only one division ("/") in each equation

Unit-of-measure equations can contain only one division ("/") and this division represents the fraction bar in the equation. If you have an equation that you can't simplify so that it has only one division, you may be able to rewrite the equation so that it uses multiplication rather than division.

For example, say you want to create a unit of measure called "resources per person per year". If you wrote this equation on paper, you might write it this way:

$$\frac{\text{resources}}{\text{person}} \\ \hline \text{year}$$

Unlike the above example, you cannot simplify this equation because "resources" and "person" are not equivalent. You can, however, rewrite this equation so that it uses multiplication in place of division by multiplying both the numerator and the denominator by the same fraction (1/year):

$$\frac{\text{resources}}{\text{person}} \\ \hline \text{year} \quad \times \quad \frac{1}{\text{year}}$$

Since this multiplication cancels out the denominator, you are left with the following equation:

$$\frac{\text{resources}}{\text{person} * \text{year}}$$

In the Unit Editor dialog box, you use "-" to represent multiplication and parentheses. That is, "x-y" is the same as writing "(x\*y)". Therefore, you would define this equation in the Unit Editor dialog box like this:

resources/person-year

Note that you could also write the equation the following way, because the unit of measure "year" has an Equation of "yr" and a short name of "year":

resource/person-yr

# **Define equations so that they are calculated in the order you intend**

The software calculates unit-of-measure equations in the following order:

1. All terms in the numerator are multiplied together
2. The numerator is divided by the denominator
3. All terms in the denominator are multiplied together