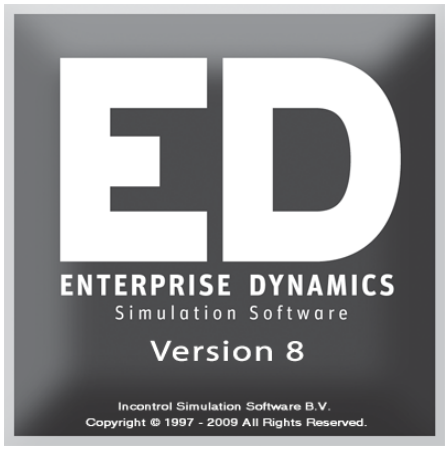
TUTORIAL ED 8



Simulation Software / TUTORIAL

Enterprise Dynamics®

Papendorpseweg 77, 3528 BJ Utrecht, The Netherlands
www.IncontrolSim.com

INCONTROL
Simulation Solutions

PREFACE

If you want to drive a car, you are legally required to have a driving license. This demonstrates that you have learned how to control a vehicle and that you are not a major threat to other road users. Equally, nobody would expect to be able to learn to drive a car simply by reading the manual supplied in the glove box.

As far as simulation software with its extensive possibilities is concerned, there is no driving license. But here too training is a prerequisite to understanding the software and its application to simulation problems. To make an analogy with the manual supplied with a new car: merely flipping through the pages is not likely to be sufficient to learn how to actually drive the car.

Enterprise Dynamics® (ED) from Incontrol Simulations Solutions is a software program for discrete event simulation. The software is supplied with an extensive online help system. However, the demand arose from beginners – especially in the educational field – for a more structured step-by-step approach that would help this group of users learn the first principles of ED quickly while working alone.

The result is this Tutorial and it represents a first step to fulfilling this demand. The Tutorial forms a significant part of Enterprise Dynamics. It teaches you the basic skills to create your own simulation model and interpret the results.

We expect that this tutorial will be adjusted and extended in the future, which will at least be necessary for new versions of ED. A word of thanks goes to all the people, who have contributed to the development of this tutorial, and our users who have made comments in how to improve our tutorial and help system.

We invite you to forward comments about this tutorial or the software, suggestions for improvement etc. to **Support@IncontrolSim.com**.

The Enterprise Dynamics Development and Support team and the Enterprise Dynamics Training & Education team.

INCONTROL Simulations Solutions, August 2011.

CONTENTS

1 TUTORIAL LAYOUT

1.1 The learning perspective

This tutorial teaches beginners to find their way through the Enterprise Dynamics (ED) simulation package.

The starting point in chapter 3 is simply ED's opening window. From there and through a progressive approach, you will get to know increasingly more about ED. This occurs mostly through examples: small problems in a specific context. You will learn about the functionality of ED by building and experimenting with models.

This approach has several advantages:

1. The user learns ED through modeling a simulation problem, which corresponds to the purpose for which the package is being used in practice.
2. The functionalities of a program are more effectively learned when they are connected to applications.
3. The user can refer back to several solutions in examples when he or she is confronted with a similar problem.

Briefly, this tutorial aims not only to show *what* a function of the software does, but also *how* and *when* it to be used. In addition, some parts of this tutorial can be used as a reference function, for example the overviews in the annexes in particular.

To be clear, this tutorial does not make the manual – which is to be found under the help menu (or when you press F1) – superfluous. Users who wish to reach a higher level in building simulation models will use the manual frequently, because it forms a complete documentation for all software functionalities.

1.2 Notation

When a submenu is referred to, the symbol | has been used. So File | Preferences refers to the submenu Preferences (consisting of a number of standard functions), which is to be found under the File menu. Emphasis on important matters is rendered by **bold** or *italic* writing of words. The case studies and the related questions are written in this font. When 4DScript code is to be found in a text, this font has been used.

A warning is preceded by: *Warning!* and an important tip, which can also be used in many other case studies, is preceded by: *Tip*.

1.3 Tutorial Structure

After a short general introduction regarding the ED program in chapter 2, the first contact with the package begins in chapter 3 in the form of an overview of the menu structure and of the sections to be found in ED's opening window.

The basic principles of the package are being demonstrated in chapter 4 by a simple queuing problem with just one server. Later on, two servers are added in order to show the concept of "channels". The atoms used in this instance are Source, Queue, Server and Sink.

The main topic in chapter 5 is the different methods to observe the results of a study. We use an example of a carpenter's factory, where the use of batches is explained. The user learns to apply the following items here as well: Monitor, the Summary Report, displaying various graphics, and the new Experiment Wizard. Furthermore, we practice at a beginner's level the steps to go through in a simulation study: model building, validation, experiment layout and analysis of results.

In Chapter 6, the emphasis is placed on observing the pre-programmed methods through which products get access to a next atom (input strategy), are placed in a queue (queue discipline), or are redirected after use (send to).

Chapter 7 introduces eight new atoms including the Assembler and the Conveyor atom, while Chapter 8 is dedicated to the link between ED and Excel. This chapter also provides a start in 4DScript and an introduction into labels.

The annexes play an important role as reference function. Annex 1 contains a short explanation of the menu structure, while Annex 2 contains a full description of the most important atoms. Annex 3, a first guide in 4DScript, forms the starting point for the underlying programming language in ED.

1.4 Learning Simulation

Simulation is not a simple technique. The sensible use of a simulation program requires the knowledge of the theoretical background to discrete simulation such as probability distributions, the model building process including validation techniques and the design of experiments. Although the tutorial brings knowledge in this field, it is not a textbook regarding simulation in itself. For this, one can refer to existing books or lecture notes about discrete simulation.

The development of a feeling for simulation and the building of models in particular are for the better part a matter of lots of practice.

Keep an eye on the website as we will be posting enhancements to Enterprise Dynamics from time to time. We also plan to make the first part of this tutorial available for downloading in a number of languages.

Examples of the models in this tutorial are also included in Enterprise Dynamics. Use them when in doubt whether your own models are correct.

2 ED'S BACKGROUND

Enterprise Dynamics is an object-oriented software program for modeling, simulation, visualization, and control of dynamic processes. The users can pick up objects – called atoms - from standard libraries in order to build their own model. ED is based on this concept of atoms as modeling objects in each model.

An atom can represent a machine, a counter or a product but can also have a non-physical character like a graph. As far as the difference in the type of atoms is concerned, we will mention *basic atoms* (five atoms often used: product, source, sink, server and queue), *transport atoms* (relating to transport such as conveyors or transporters), *results atoms*, etc.

Thanks to the open structure of ED, the advanced user can build and use own atoms, for example to model a machine with very specific characteristics. At this point, ED includes 100 standard atoms, but this number is ever increasing. The beginner may typically only need to select from around 30 frequently used atoms to have enough material for his or her applications.

Atoms thus are predefined modeling objects used to build models quickly and to carry out studies. ED also has a built-in programming language called 4DScript, which can be used for processing specific conditions from reality in the model. This language consists of approximately 1.100 words at this moment, but in this tutorial we will address 4DScript only briefly.

Thanks to the open structure of ED, the user can expand the package himself or design his own layout. Incontrol itself provides a few of the shelf products:

- ED Logistics for production, material handling and distribution
- ED Airport for airports and carriers
- ED Educational for teaching purposes, consisting of a combination of software of ED Logistics with a number of elaborated case studies which the teacher can use for student assignments
- ED Production that allows you to swiftly model processes that are common on a production environment
- ED Transport to model internal and external transport and information flow

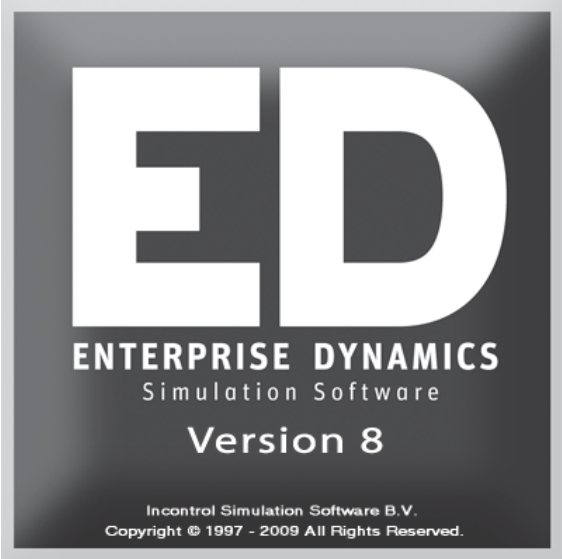
This tutorial can be used to learn the basic skills necessary to work with the ED software, but it also explains some atoms and features that are only part of ED Logistics. Furthermore it addresses topics and questions that typically arise in Simulation studies and is therefore an important part of ED Educational.

Incontrol Simulations Solutions is a Simulation Solution Provider: in addition to building, developing and selling the Enterprise Dynamics software, it also provides extensive services in the field of computer simulation, such as training and consulting. The consulting services involve carrying out simulation studies as well as application building.

3 GETTING FAMILIAR WITH ED

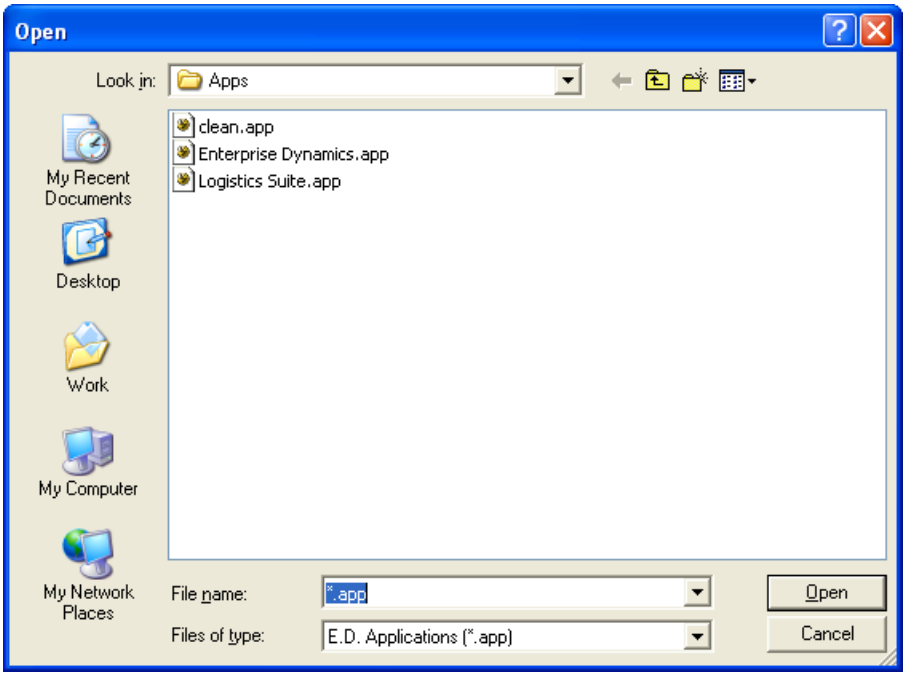
3.1 Starting Enterprise Dynamics

Enterprise Dynamics can be started using the Start Menu. During this phase, a splash screen (picture 3-1) first appears; you sometimes may be asked to choose an application to start (Picture 3-2). Here, an .app file has to be selected.



Picture 3-1: Splash screen

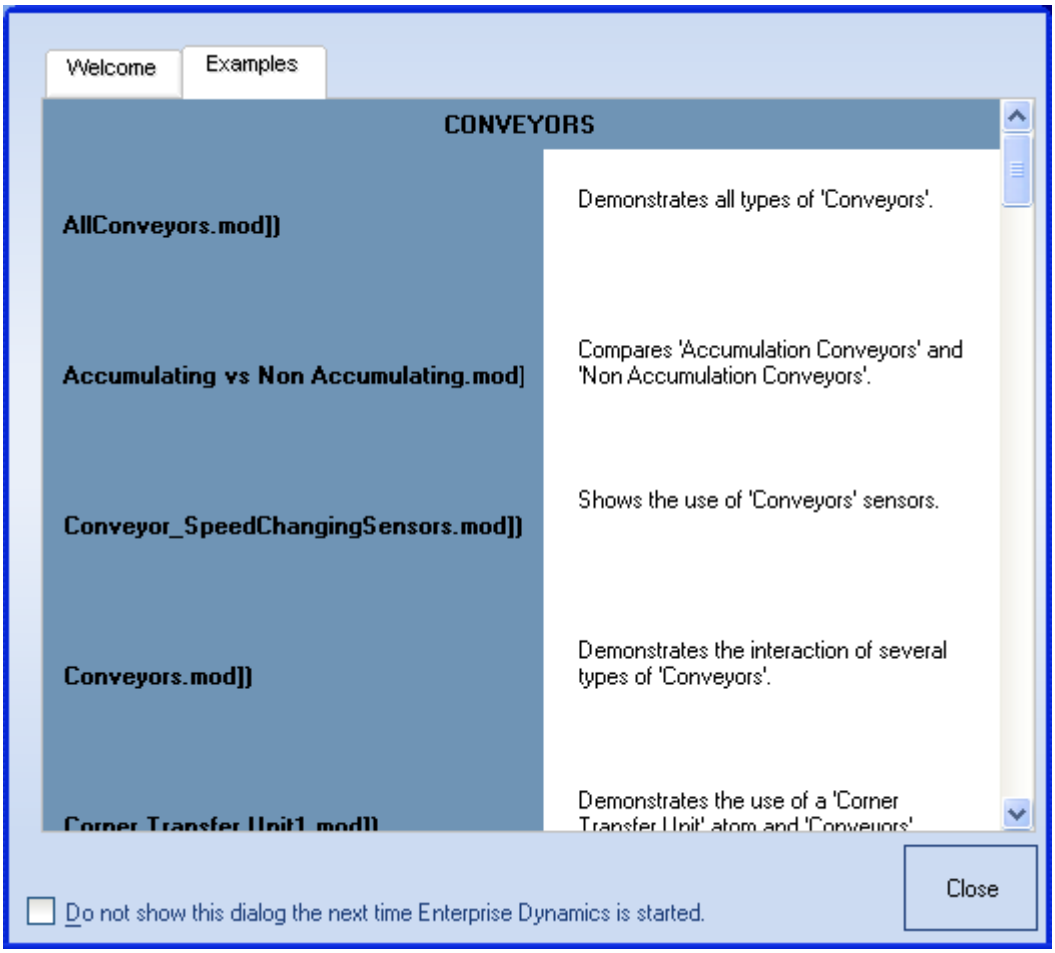
Enterprise Dynamics will often select the application file, in which case the user will not get to see the application selector window!



Picture 3-2: Application selector

The functions of Enterprise Dynamics itself are to be found in the application files. They can be programmed to determine which menus appear to the user and which atoms are directly available. The user can adjust these application files.

3.2 The Example Wizard

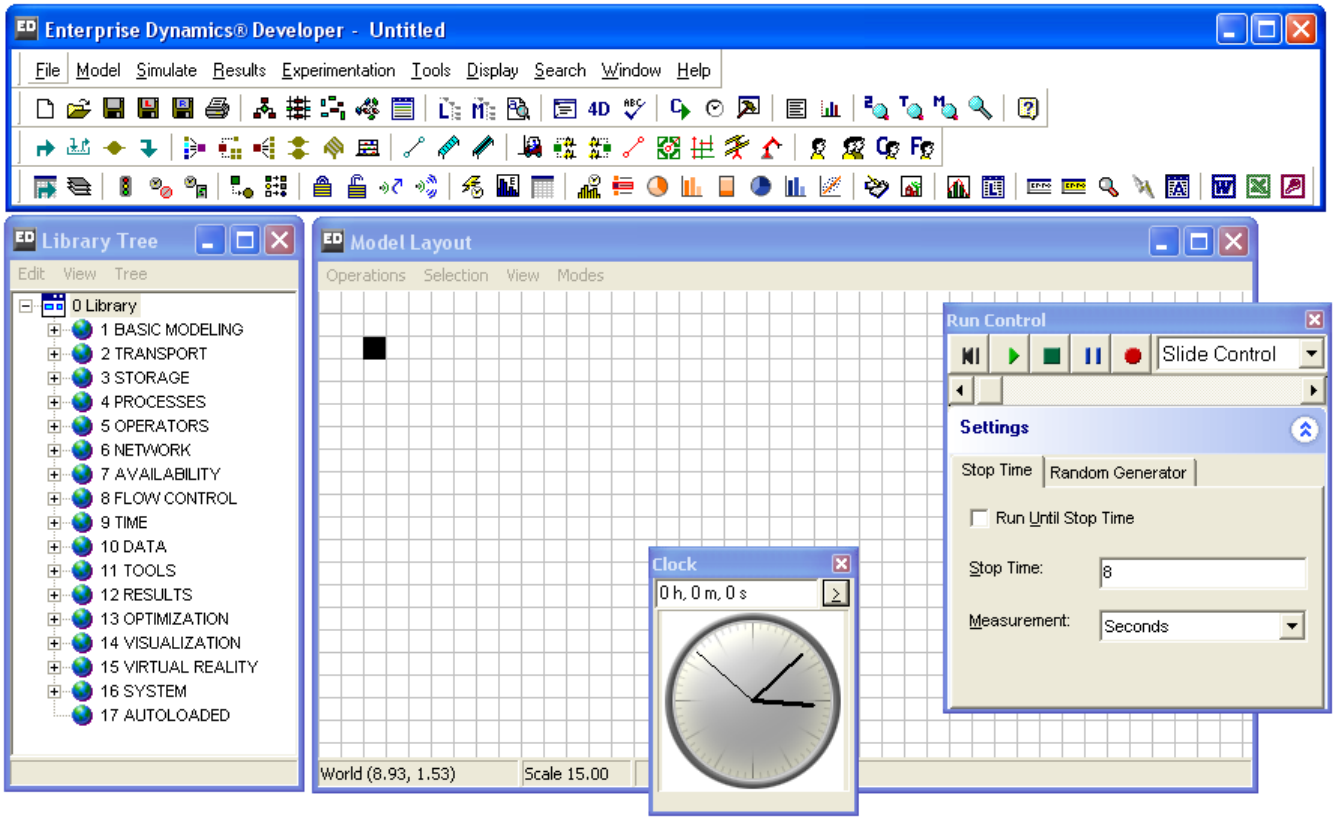


Picture 3-3: Example page of the Example Wizard

When you first start Enterprise Dynamics, you are presented with the Example Wizard, see Picture 3-3. The wizard offers some of the available example models that come with ED. Later on, you can also open it from the Help menu. Just click on one of the models to open the example.

3.3 The window sections

As soon as Enterprise Dynamics is started up completely, the opening window should be approximately similar to Picture 3-4.



Picture 3-4: Layout of the opening window in Enterprise Dynamics

The window is divided into the following sections:

- A *menu bar*: among others for opening and saving files. Section 3.3 covers the menu structure more thoroughly.
- The *speed buttons*: by using these buttons, specific atoms can be dragged into the model and commands can be given to Enterprise Dynamics (e.g. save a model). When clicking on a speed button, the atom automatically appears into the model or an action is started.
- The *library*: The library includes all atoms a user can place into a model. Each atom has a definite function and by combining the right atoms, it is possible to re-create ('model') a business process in Enterprise Dynamics. Model building is described in chapter 4.
- The *model layout window*: this is where the model is being built.
- The *run control*: use this to reset and start the model and to regulate its execution speed.
- The *clock* displays the simulated time already elapsed during the simulation (not the real time!).

3.4 The menu structure

The function and the appearance of the menus are similar to those in other Windows applications, such as Word and Excel. The most used menu options are explained in the table below.

The main menu is to be found in the menu bar, which is subdivided as follows:

File	Make, open or save files or to control standard functions such as printing or file location.
Model	Create and view models.
Simulate	To open a new Run Control or Clock window. From here you can also find the History settings.
Results	To generate reports and graphics of a *single* simulation run.
Experimentation	To design, perform and evaluate an experiment with *multiple* simulation runs.
Tools	Contains tools such as the Atom Editor for atom building, 4DScript interact and Autofit to fit a distribution to given data.
Display	Regulate the visualization of the model in 2D or 3D.
Search	To find atoms in the Model Tree that are selected in the Layout Window, and vice versa.
Window	Includes various windows such as the 4DScript words overview, the Tracer and the Resource Manager for icons and 3D models.
Help	Includes the complete guidebook as well as company and version information.

Note that the options Model, Simulate and Results follow the successive steps of a simulation study!

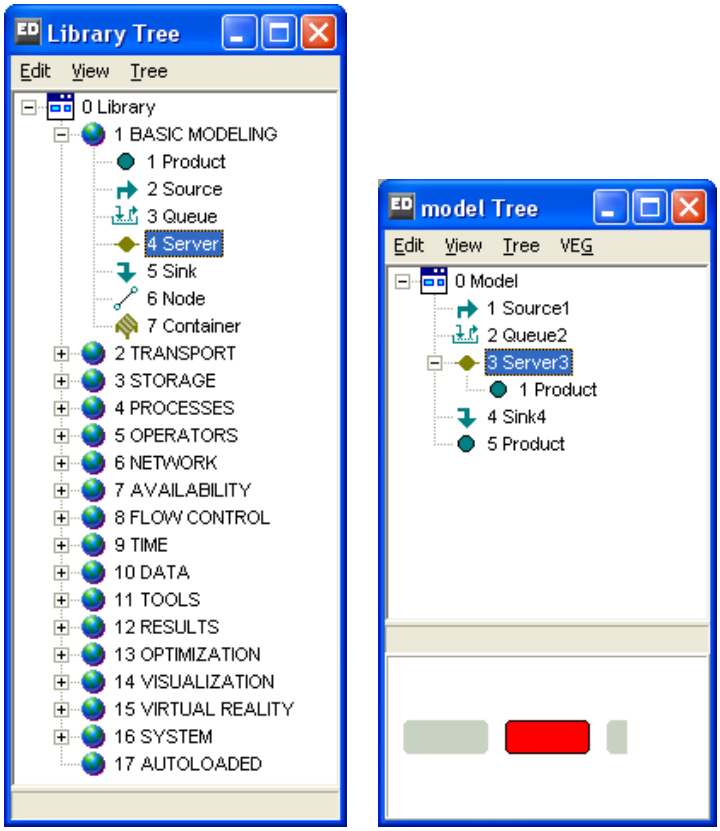
Behind each main menu item, submenus are hidden. A short explanation of this structure including a description of each submenu is mentioned in Annex 1. The menu options displayed in *italic* are only relevant for advanced users and can be skipped by beginners. This annex deliberately contains a total overview, because this part of the tutorial is intended to be used as a reference book as well. Read Annex 1 for a first insight into the program structure. A good knowledge of this structure is particularly useful when creating models individually at a later stage.

3.5 Structure of the library and the model

In ED we use a tree structure for the visualization of the organization of the atoms. We use it to indicate which atoms contain which other atoms. For example, the main tree gives you a complete overview of the application, the library and the opened model. Two other important trees are:

- The Library Tree (see Picture 3-5) which lists all atoms a user can insert into the model. The atoms are divided into groups, for example a transport group and an operators group. By selecting an atom and dragging it into the model window (the 'model layout'), the atom is added to the model.

- The Model Tree, where all the atoms used in the current model are listed. Press F5 to refresh the list.



Picture 3-5: The Library Tree & Model Tree of Example 1

The windows can be opened using menu options in the Model menu and with the speed buttons shown in Picture 3-6.

Picture 3-6: Speed buttons for the Library and Model Tree

4 MODEL BUILDING BASICS

In the previous chapter, only the theoretical aspects of Enterprise Dynamics have been discussed. In this chapter, we will start by building a simple model in Enterprise Dynamics. The objective here is to learn about ED, not to fully complete a simulation study.

At the end of this chapter, the user should be able to develop a model in which several machines are used simultaneously.

4.1 Dragging atoms into the model

The first step in the creation of a model is the placement of the right atoms into the model. In this section, we will start by building a simple model that consists of the following four parts (see Picture 4-1):

- **Source:** the function of this atom is to generate products into the model.
- **Queue:** this atom is a waiting area for customers or products.
- **Server:** the function of this atom is that of a machine or of a counter. Atoms entering a server undergo a process and remain in this atom for a certain time (the process time).
- **Sink:** the products or customers leave the model through this atom.

Picture 4-1: Speed Buttons: source, queue, server and sink

Example 1

An average of 20 customers an hour come into a post office and the assistant has on average two minutes to help a customer. Of course, the number of customers may vary from one hour to another.

The two-minute time that the assistant has to help a customer can vary as well. A customer who only wants to buy stamps will need less time than a customer who wants to open a new account. The customers are served on a first-in first-out basis.

A few customers complained about the queues and because the post office manager is very concerned about the service to his customers, he wants to inquire into this problem.

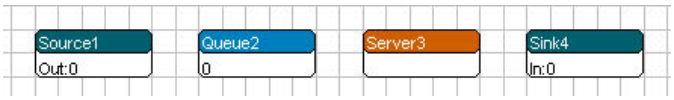
Questions and assignments

1. How high is the utilization of the post office assistant? What does that mean?
2. Can you make an estimate of the average queue length?
3. Which characteristics would you measure for the post office manager?

Simulation is a tool to gain insight into the average queue length. Because this model is the first one we are going to build, it will be dealt with step by step.

We therefore use the 4 atoms mentioned in the beginning of this section and put them into the model in the above-mentioned order. The first one is thus the **Source**, then the **Queue**, the **Server** and finally the **Sink**.

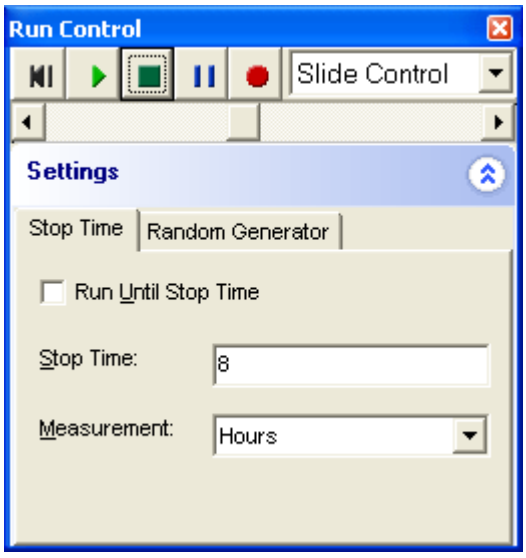
The selection of the atoms occurs by clicking on the Speed Buttons (see Picture 4-1) or by dragging the atoms out of the library. In the latter case click the "+" belonging to the first category "BASIC MODELING" in the library window. When this is done, you get to see Picture 4-2 on your screen:



Picture 4-2: the first model

In case of errors, click on the atom concerned and press 'delete' to remove the atom from the model.

To check whether the customers walk through the model in the correct sequence, we will start a simulation run by using the Run Control window. If this window is not visible yet, use the option Simulate in the main menu. In the Run Control window, the sub-function Slide Control has to be selected (see Picture 4-3). This function enables you to adjust the speed of the simulation.



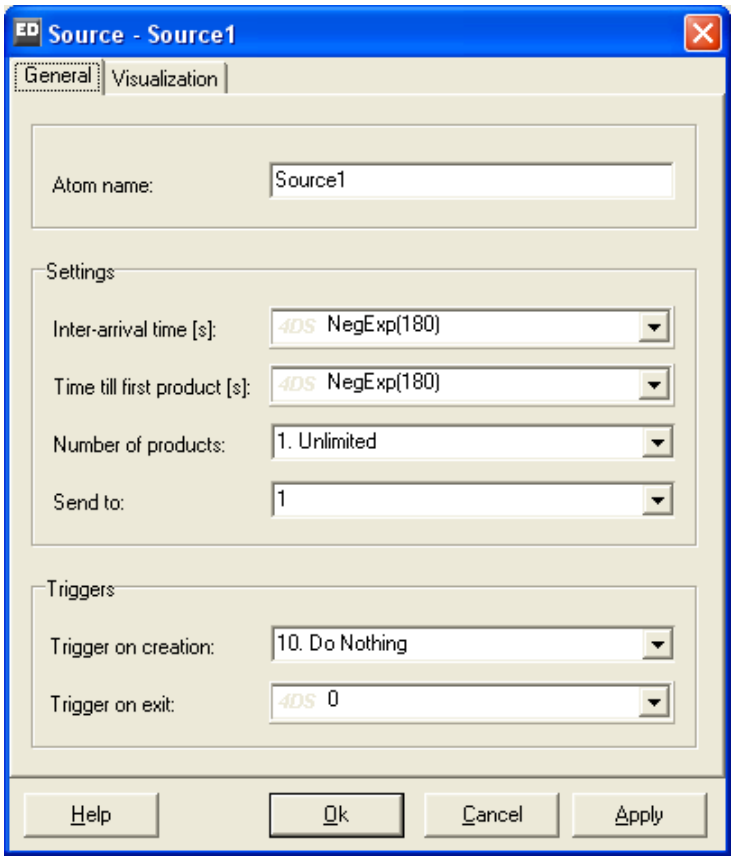
Picture 4-3: select Slide Control

Before the simulation can start, you first have to click on the Reset button. It is the leftmost button in the Run Control window. After clicking on the Reset button, a blue dot appears next to the Source atom. This is a product atom. By clicking on the start button (green triangle) the products, in this case customers will pass through the other atoms, in this case the post office. If necessary, adjust the speed by dragging the speed button in the Run Control window!

In this model no queue will appear before the counter (Server). As a result, the blue dot is only visible in the counter (Server). Furthermore, a percentage is displayed in the server. This percentage reflects the utilization of the Server so far. When the percentage is not visible, you might need to zoom in or zoom out the display by clicking on the left and right mouse buttons simultaneously and by moving the mouse forwards and backwards.

The atoms have been placed in the right order, but we still need to enter how many customers per hour are coming in and how long the assistant needs to help a customer. Before you can enter these cycle times in Enterprise Dynamics, you have to know that *all cycle times in Enterprise Dynamics are defined in seconds*. When the post office assistant needs two minutes to help a customer, you enter 120 seconds in Enterprise Dynamics. Equally, we must define how much time elapses between customer arrivals, called the Inter-arrival time, in seconds.

First, we change the parameters of the Source, so that an average of 20 new customers arrive at the post office per hour. By right-clicking (or double-clicking) the Source atom, an input window appears (see Picture 4-4). Normally this window will open with the "General" tab sheet. The time elapsing between two arrivals can be entered in the Inter-arrival time field. We use a probability distribution to model the uncertainty in the arrival process. In this example, the negative exponential distribution is used, which is a probability distribution often applied in unpredictable arrival or service processes. In Enterprise Dynamics, the following 4DScript code is to be used: NegExp(e1), where the parameter e1 stands for the average value or expectation of the exponential distribution. Now enter 180 as inter-arrival time and confirm with OK:

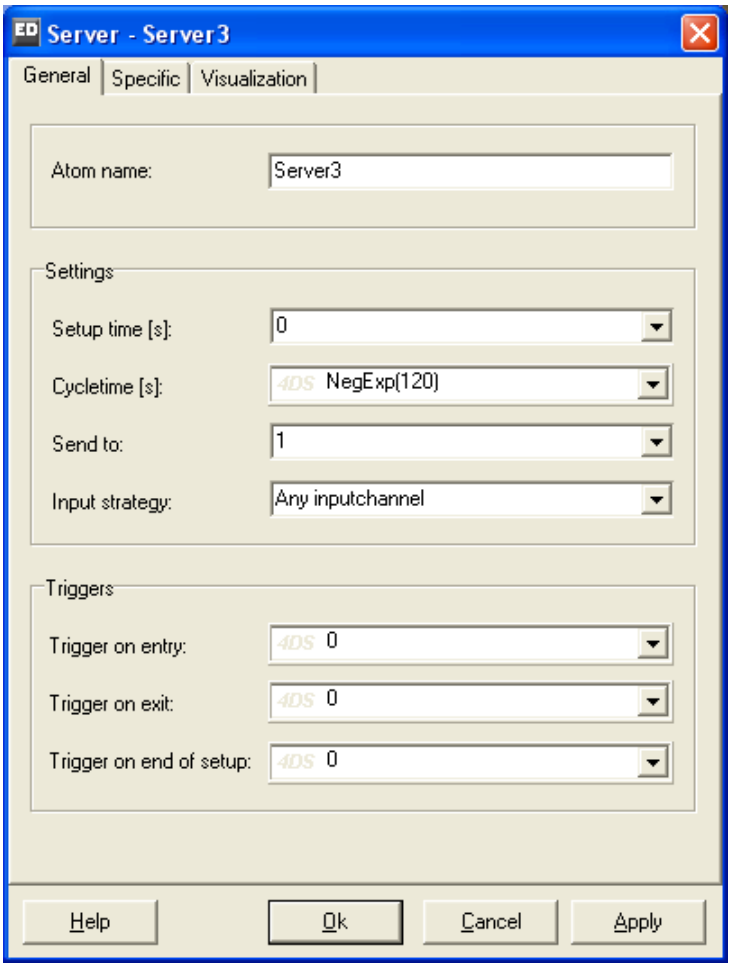


Picture 4-4: Input window Source

Important:

To the right of the input fields, you will often find a triangle pointing downwards. Clicking or double-clicking on this triangle opens a list with available and pre-defined options. Check first whether any of these options might be useful for your specific case! For an explanation of these options see Annex 2, this contains a full description of the most important atoms and their input fields. The grey letters '4DS' mean that this is where 4DScript commands, ED's programming language, can be entered.

Afterwards, the time that an assistant needs to help one customer has to be entered as well. The negative exponential distribution is used in this case too. Now, right-click or double-click the Server atom, so that the service time can be adjusted on the "General" tab sheet (see Picture 4-5). After selecting "NegExp(10)" click it (a 4DScript window will appear). Adjust the cycle time in such a way that the assistant needs an average of two minutes to help one customer.



Picture 4-5: Input window Server

Reset and restart the simulation with the Run Control. Because we have increased the time between two arrivals, it is a good idea to increase the simulation speed. By zooming in sufficiently, you can see how many customers are waiting in the queue.

You can replace the blue dot, representing a customer in our example, with another icon. To this end, double-click (or right-click) the blue dot to the left of the source. In the 2D Icon input field, with the "Visualization" tab sheet selected first, double-click once again the blue dot. The Resource Manager will appear, featuring (some of) the icons the users can select. Click on the icon you want to select and on OK in the Resource Manager: the blue dot in the Product input window is replaced by your new icon. Confirm this change with OK.
The icon that Enterprise Dynamics uses in the 3D display can be changed as well. Use for this the option 3D Icon and select the Person icon from the list. Now display the model in 2D and 3D. To open the 3D visualization: select the main menu option Display and then one of the 3D possibilities. To navigate through the 3D model, move the mouse and click on the mouse buttons.

Tip: There are many more icons available. Try to add an existing icon to the standard list (Resource Manager) with File | Import!

In this instance, the selection of a different icon is only to improve the display. However, if distinguishing between different types of customers is necessary, the use of several icons can bring additional clarity. For example, one group of customers could be represented by a green icon and another by a blue icon.

The model can now be saved, with File|Save as. For reference purposes, the model of this post office is to be found in the tutorial models under the name Postoffice1.mod.

4.2 The Channels

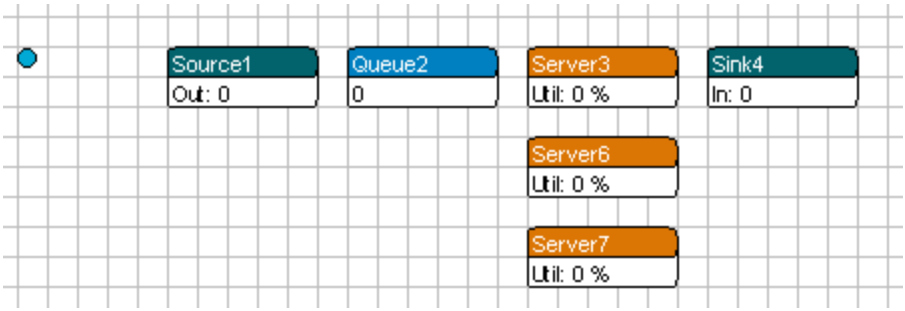
Example 1 (continued)
In the neighborhood, two post offices have been shut down. The manager hopes that many new customers are going to use his post office and even expects twice as many customers in the near future. As a result, an average of 40 customers will come into his post office per hour. In order to figure out if his post office can manage this flow with the existing counter, he decides to carry out a new simulation study.

Questions and assignments

4. Does it make any sense to simulate this situation? In order to answer this question, use the expected utilization and check your own supposition by modifying the model postoffice1.mod to the new situation and by carrying out a simulation run.

Example 1 (continued)
It seems that one counter will not be enough to manage the flow. The manager now wants to open 3 counters to be on the safe side. Once again, he wishes to see the effect of this situation on the queue.

To begin with, we put the two counters under the existing counter (select the right atom to model a counter). Then, the parameters for the counters' cycle times have to be changed in such a way that they correspond to the first counter. To this end, you might want to go back to chapter 4.1. The model should now look like Picture 4-6.

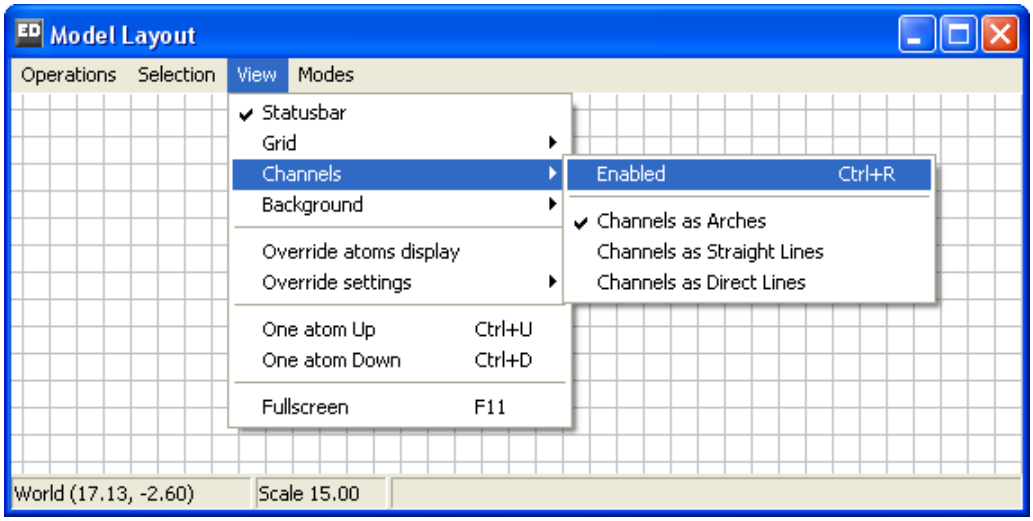


Picture 4-6: Post office with 3 counters

Tip: Atoms can simply be copied –including all entered fields and parameters-! For this, first click on the atom to be copied and then on F6: a duplicate of your atom is created.
To copy more than one atom right-click and select an area around these atoms or choose with the Ctrl-key your own selection of atoms: now F6 or Ctrl-V will do. Look out! With Ctrl-V the copy will be right behind the original, so you have to reposition your copy. For repositioning more than one atom you can also use one of the selection procedures above in combination with the arrow keys.

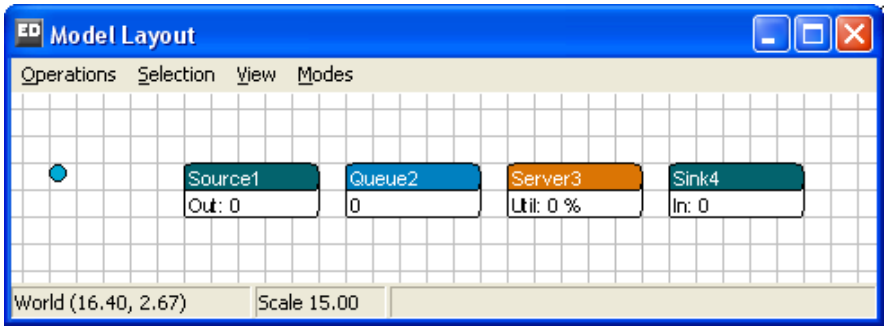
However, when the simulation is started, you will see that the customers are still sent to the first counter. In order to correct this, we have to look into the method that Enterprise Dynamics uses to direct the customers.

In the Model Layout window of the View Menu, select the option Channels (see Picture 4-7). By doing so, the channels of the atoms appear. The purpose of these channels in ED is to route products and to transmit information and they are therefore covered thoroughly in this chapter.



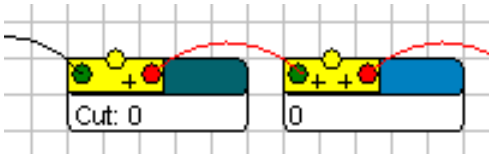
Picture 4-7: Switching on Channels

Switch on the option enabled to see the channels (see the result in Picture 4-8).



Picture 4-8: The original model of the post office with channels switched on

When the lines between two channels do not appear curved but squared, it might be helpful to switch on the option Channels as Arches in the View | Channels (see again Picture 4-7). As soon as the channels are switched on, a block is added in the upper left corner of all atoms. This yellow block contains the input and output channels of the concerned atom. (See Picture 4-9)



Picture 4-9: Connected channels

The block contains several dots. The left dot is an input channel and the right one an output channel. A channel can be *open* (in green) or *closed* (in red). When both the input and output channels are open, the connection between the input and output channels is *ready* (in green), otherwise, the connection is *not ready* (in red).

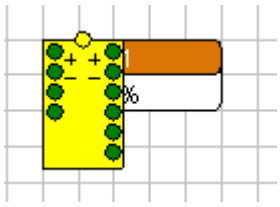
The dot in the middle (above the yellow block) is designed to receive (but not send!) information. In this tutorial, this channel is called the Central Channel, which is used to register information regarding an atom. There is always only one central channel present on an atom, but *several atoms can be connected to the same central channel*.

Products come into an atom through an input channel and leave the atom through an output channel. An input channel always has to be connected to an output or a central channel, whereas an output channel always has to be connected to an input or a central channel. Each input or output channel can only be connected to one other channel.

By clicking on the "+" sign next to a dot, the number of input or output channels can be increased and by clicking with the right mouse button on a channel dot, an overview of all connections of the atom appear. The "-" sign which appears as soon as the number of input and output channels is two or more speaks for itself.

Important: Although you can use channels for many purposes, the basic function of channels is the indication on an atom (read: queue, counter) of all possible next locations of products situated in this atom. In this way, the channels provide a (rough) routing of products through the model!

Picture 4-10 represents an atom with 4 input channels, 6 output channels and the central channel.

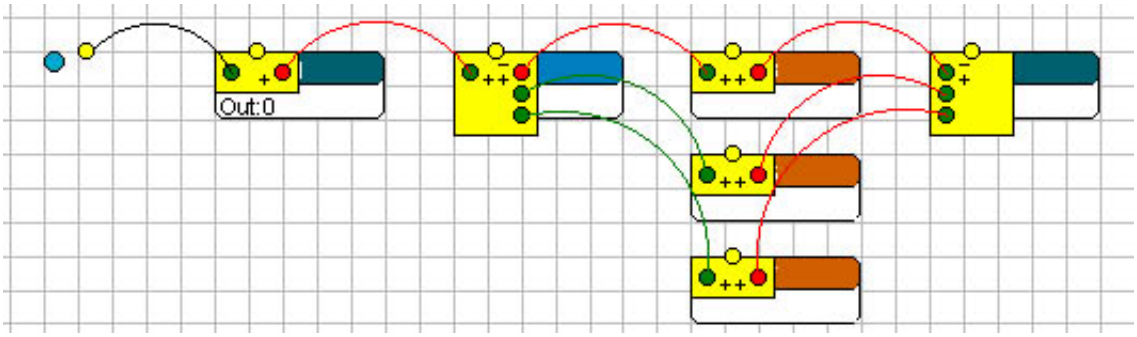


Picture 4-10: Atom with 4 input channels and 6 output channels

Remark: One of the revisers of this document happens to be color-blind, as are most laser printers and therefore has trouble distinguishing an open (green) and closed (red) channel. You can change these color settings in ED yourself by clicking File|Preferences …|Appearences. Change the color of "`2D -> Open channel`" and "`2D -> Closed channel`".

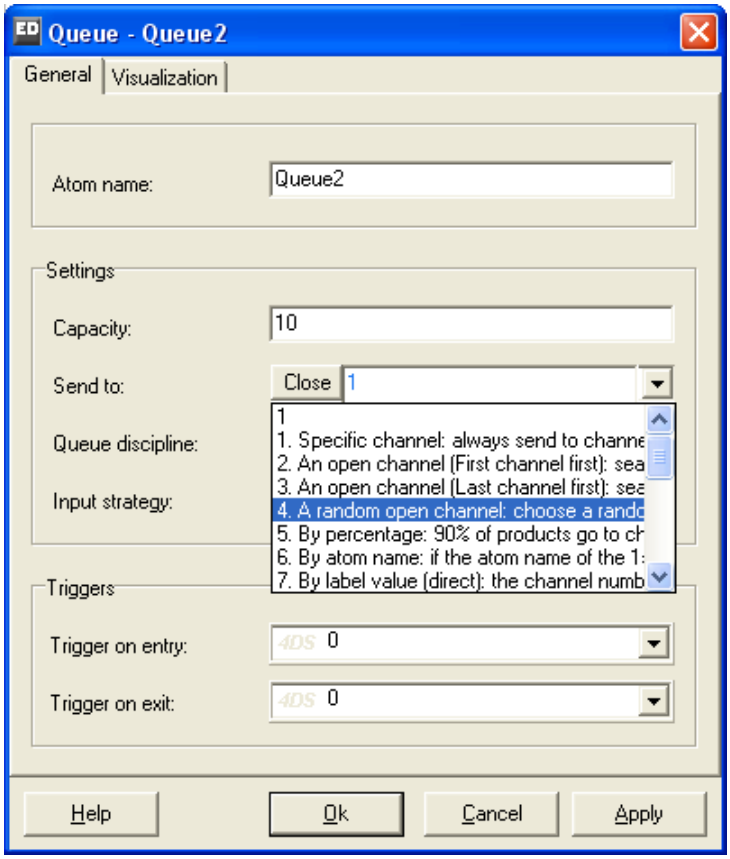
By dragging a line between an output and an input channel with the mouse cursor, two atoms are connected with each other. By dragging a line between an input or an output channel and the central channel of the same atom, a connection is broken.

Now connect the channels in the Enterprise Dynamics model in such a way that the customers only go to one of the three counters and afterwards from one counter to the exit. The result should look like Picture 4-11. Restart the simulation (do not forget to reset!) and check if the customers now go to the right counters.



Picture 4-11: Post office with connected channels

The reason why the customers are still going to the first counter only is that unless we tell ED otherwise, it sends all products through output channel 1. In the input window that appears by double-clicking or right-clicking an atom, you can specify the channel through which the products to leave the atom. By double-clicking on the Queue, we can alter the parameters of our model. In this instance, we have to adjust the value in the Send To field (see Picture 4-12: Queue: Send To): if we want that each open channel can be selected, the options 2, 3 and 4 are relevant.

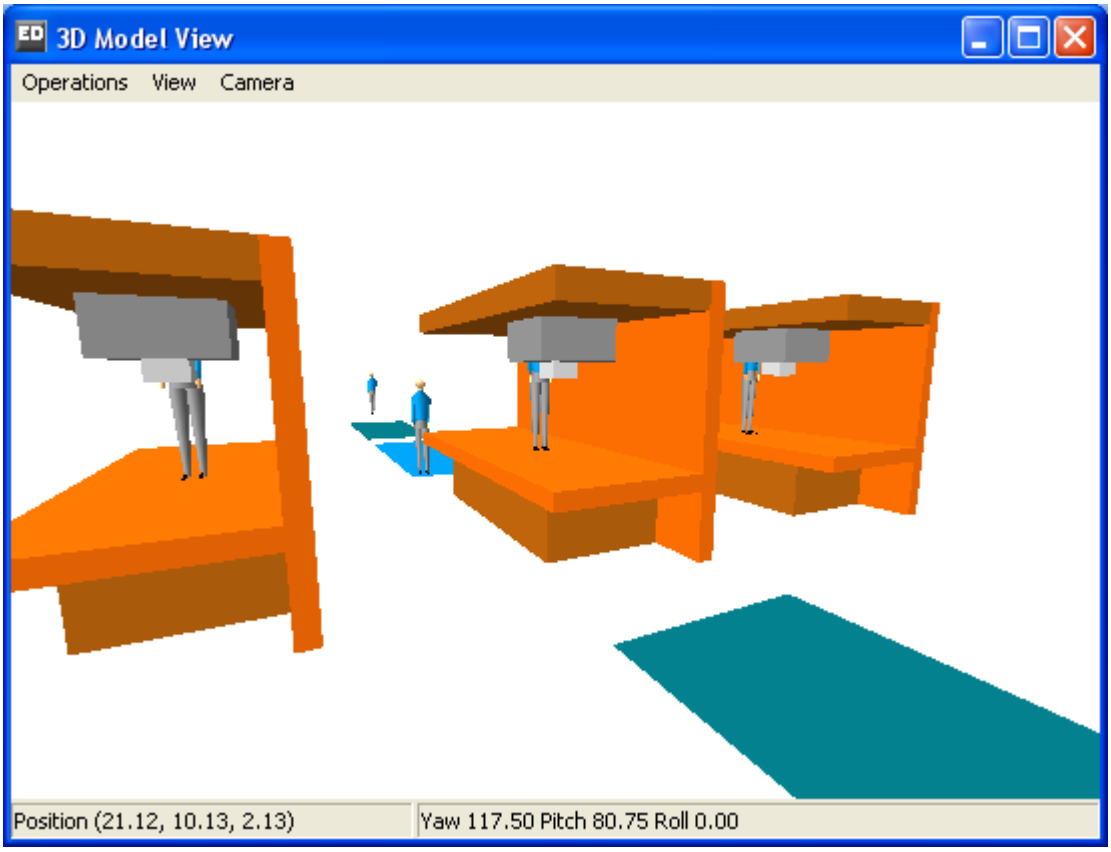


Picture 4-12: Queue: Send To

Explain why in option 2 or option 3, either the first or last server will get the highest utilization, while the servers' utilizations in option 4 will be more or less the same after the simulation has run for some time!

In the Send To field, you can either enter a number, write a short piece of 4DScript code that will result in the output channel number, or use predefined code. For beginners, the last option is definitely the easiest method. By clicking on the small down triangle in the input field, a list appears, showing the predefined options that the user can select. After a definite line is selected, text displayed in blue can be modified. Obviously, this possibility is applicable only if there is something left to change.

Now select a strategy where the utilization of the 3 counters is about the same. If the utilizations of the counters are not directly visible, use both mouse buttons in order to zoom in or zoom out. View the model also in 3D with Display | 3D Model View!



Picture 4-13: The 3D-model with the three counters modeled as machines

Questions and assignments

5. Is the layout with 3 counters sufficient?

The model can be saved now. For reference purposes, the model of this post office is to be found in the tutorial models under the name Postoffice2.mod.

5 ANALYSING THE RESULTS

In the last chapter we used a detailed and progressive method to simulate a post office. You learned how to place atoms in a model, and how to connect the channels between atoms. You also learned how to create a flow of product atoms into the model at a defined interval, and how to make them stay on a machine (counter) for the correct processing time. As a result, all the basic principles for building a model have been dealt with.

In this chapter, we are going to build a more complex model and above all observe how to follow and measure the results of the simulation study. We will once again start with a (simple) example and (after building the model) explain the different ways of measuring results. We will then practice these in the model.

Example 2: The carpenter's factory

General context:
Most of the time, a carpenter's factory has its own design department. This department is complex to manage because each order consists of a separate project, where the process times of each production step has to be estimated as accurately as possible. Large intermediate stocks, long throughput times, and variable bottlenecks often characterize such an environment.

The management of a carpenter's factory, which mainly produces windows and window frames, wish to gain insight into bottlenecks, production numbers and throughput times. For this purpose, they decide to have a simulation study carried out. To keep it simple, only the production of windows will be treated in this case.

The windows production process can be divided into several steps. First, long timbers come in, which are cut into 10 shorter lengths on a saw. Afterwards, these shorter pieces are cut into the desired pattern on two milling machines. When the parts have been milled, they go to the bench vice.

At this stage, four pieces at a time are placed in a frame and glued. There are two milling machines and two bench vices, which are in parallel in the production process, so the parts pass through milling and gluing only once. There are storage areas between all consecutive production steps, and there is a continuous supply of raw material. To control the stock levels, each intermediate storage space is limited to 100 parts.

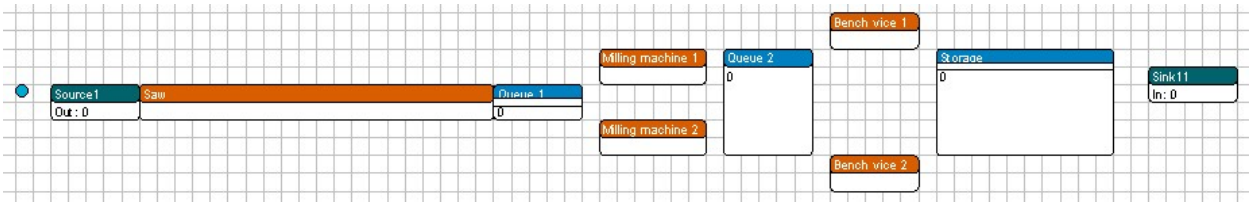
The time to saw the incoming timber into shorter lengths is uniformly distributed between two and three minutes and the processing time on the milling machine is normally distributed with an average of 36 seconds and a standard deviation of 2 seconds.

The bench vice takes exactly two minutes to glue a window together. The production process takes place continuously between 09:00 to 17:00 hours. The products that are not completed at the end of the day remain until the next day.

Questions and assignments: Analysis beforehand

1. Make a drawing of the process with the capacity per hour in raw lengths, milled lengths and frames for each production stage.
2. How many window frames are expected to leave the factory each day?

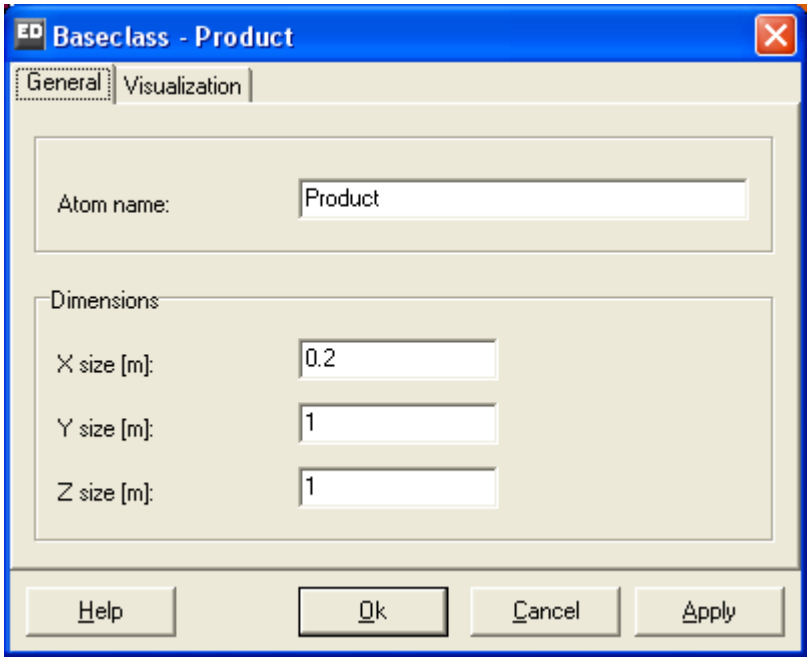
Now build the model in Enterprise Dynamics. The probability distributions are to be found in the pull down menu of the cycle times. Note that for the normal distribution, the formula max(0, normal(36,2)) is used to prevent negative service times from appearing. Use the Batch Rule "1 in, B out" with batch size 10 on the servers (on the tab sheet "Specific") for the 1:10 operation from timbers to short lengths and the batch rule "B in, 1 out" with batch size 4 for the 4:1 operation from short lengths to frames. Change the name of the Servers, so that the model is easier to read. If everything went well, your model should look like the model in Picture 5-1.



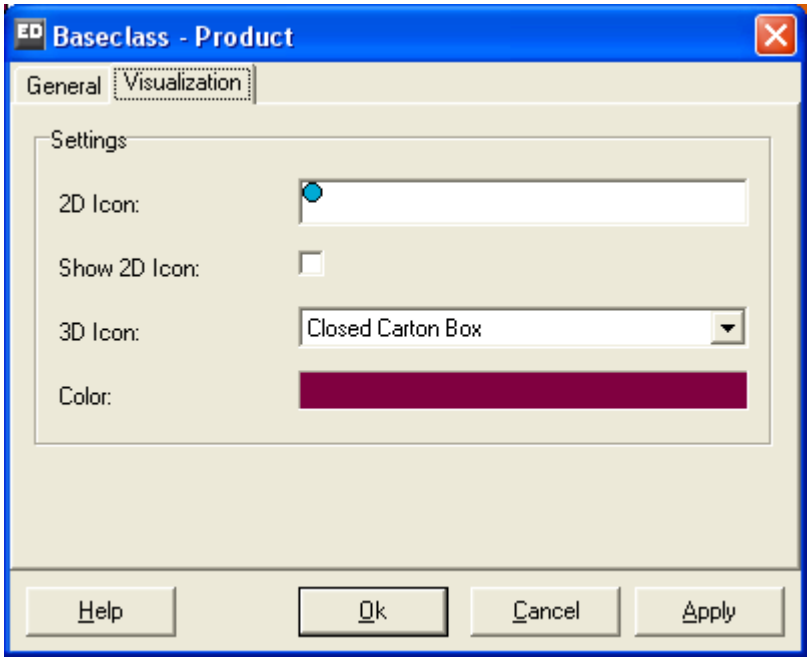
Picture 5-1: Layout of the carpenter's factory

Changing the size of the atoms can be done by clicking the atom, and then dragging the upper left corner or the lower right corner to the place you want.

The Product atom (the blue dot) can be changed into a brown `timber': double-click or right-click on the product and change the input field as indicated below.



Picture 5-2: Changing the product's general settings



Picture 5-3: Changing the product's display

Save the changes you have made. What has occurred?

The 2D icon with the blue dot is still there, but clearing *Show 2D icon* leaves only the brown background visible. It has a length of 0.2 meters and a width of 1 meter, similar to a timber in upright position. In 3D, the height (size z) of 0.2 meters plays another role. Note that the product icons can differ in 2D and 3D. In 3D, select e.g. the pallet or another icon!

In 2D the grid that is visible behind the atoms functions as a grid for coordinates, with each square the size of 1 x 1 meters. The visible black square matches the coordinates (0,0). Physical dimensions do not play a role in this abstract model of a carpenter's factory, but they definitely do play a role in other applications such as warehouses with short walking distances, or fork lifts having to cover a certain distance!

If you are not sure that you got your own model correct, you can also use the supplied model timber1.mod in the next sections.

5.1 Techniques to measure the results

Now that we have a working model, we can start with the measuring results. Within Enterprise Dynamics, there are several techniques we can use to achieve this.

1. Information indicated on the atoms.
   Each Queue shows how many products are in the queue, each Server shows its utilization and each Source or Sink shows how many products have entered or left the model. This information is particularly useful to see if a model is working logically during the simulation (which is part of the verification of the model).
2. The Result Atoms.
   The Results atoms can be found in the Results section of the library. For example, the

StatusIndicator and StatusMonitor atoms display the current status of an atom and the fraction of time that the atom is Busy, Idle or Blocked.
The Generic Monitor Atom can graphically display other information about an atom. The user can select which information is to be shown, and in which form. This atom is particularly useful to validate the model.

3. The Summary Reports and the Graphs from the Results menu.
   The intermediate results of a simulation run are shown here. This method is especially convenient for a quick overview of the system's status and a rapid feedback of the effect of changing various parameters.
4. Setting up an experiment with the Experiment Wizard.
   This technique is quite different from the last three and is used for the actual study. The length of a measurement period and how often this period is to be simulated, e.g. 10 times half a year, is set beforehand. We must also specify what variables we want to measure. At the end of the experiment the mean values of the variables are given within confidence limits.

5.2 Measuring the results

In section 5.1, four techniques have been presented, through which the results of a simulation study can be shown. In this section, we are going to apply these four methods to the carpenter's factory. For this, the carpenter's factory model has to be opened in Enterprise Dynamics.

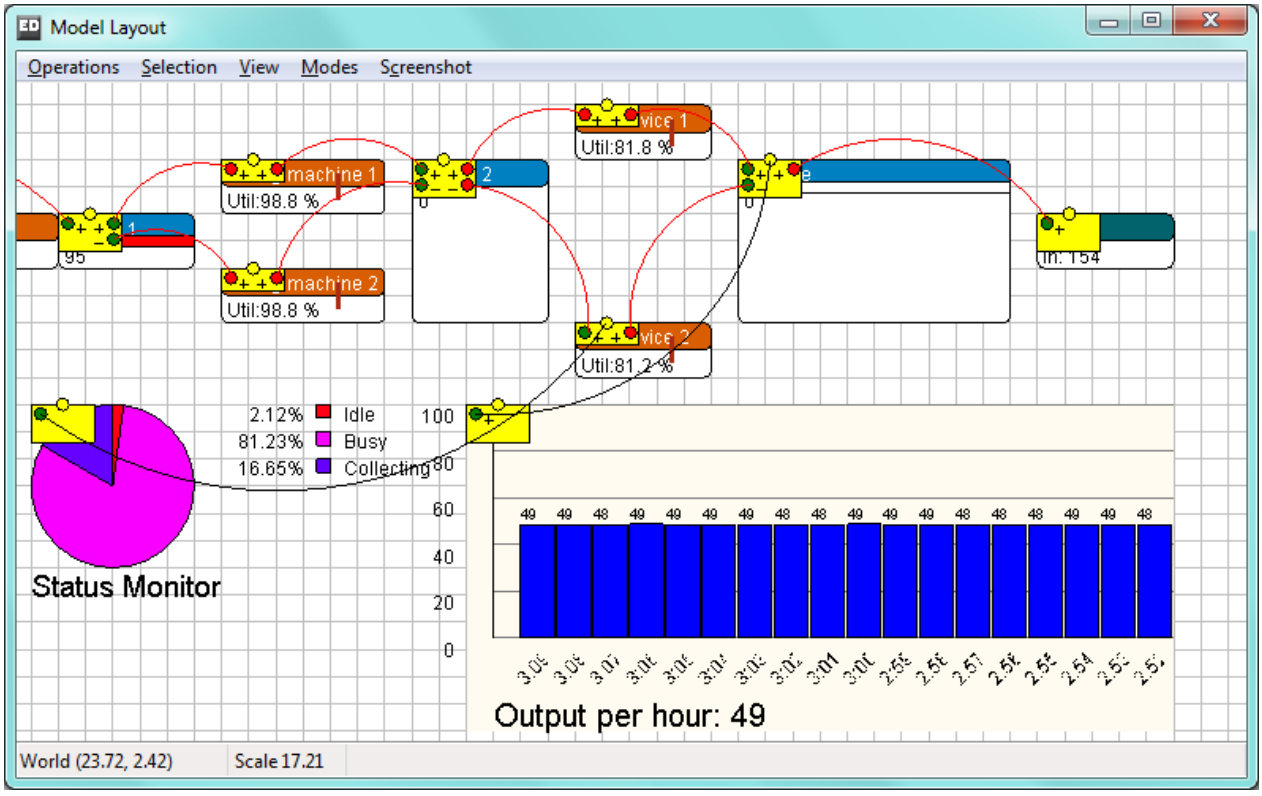
1. Information shown on the atom
   The clock function in Enterprise Dynamics has to be visible on the screen (submenu of Simulate). Start the simulation and see how many products leave the factory within eight hours. In the Run Control window check the option "Run until stop time". Set the stop time on 8 and measurement on hours. If you now start the simulation the will stop after exactly 8 hours.

Questions and assignments

3. Perform this a few times and note the number of windows produced. Does this correspond to the analysis carried out previously? What is the bottleneck in this process?

2. Results atoms
   The use of the results atoms is quite simple. The Library tree (and not the Model Tree) has to be opened. Look for the StatusMonitor atom and the Monitor atom in the Results group and drag them into the model. If the channels are switched on, the atom should look like Picture 5-4: Part of the model with (already connected) Monitor atoms.

   To use the StatusMonitor, it is sufficient to connect its input channel to the central channel of the atom to be monitored. When the simulation is reset and started, the status monitor atom will automatically begin displaying the status statistics. In this case connect it to one of the bench vices, reset the model and press Start. Examine which proportion of its time the bench vice spends on waiting for timber to arrive, on collecting timber after the first arrived, and on actually processing the timber.



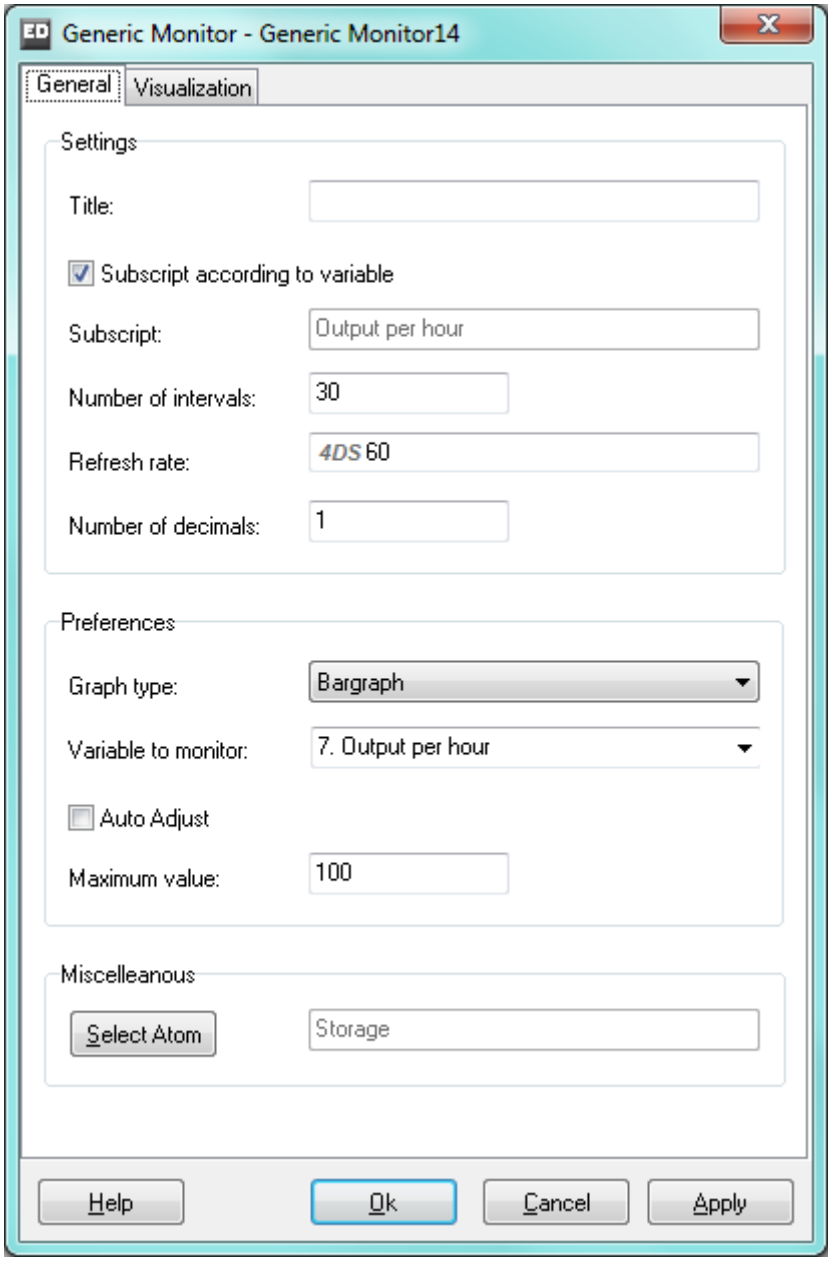
Picture 5-4: Part of the model with (already connected) Monitor atoms

The Generic Monitor atom needs more work. Again, the first step is to connect the monitor with the atom you want to observe. In this case, we want to know how many windows are leaving the factory per hour. As a result, the monitor has to be connected with the Queue before the Sink. Connect the input channel of the monitor to the central channel (the information channel) of the Queue. You can also double-click or right-click the monitor atom and then press the button "Select Atom" select from the list that appears the atom that is going to be observed.

Secondly, you have to enter what is going to be observed. For this, double-click or right-click once again on the monitor atom and adjust the Variable to Monitor. Select the option "Output per hour" out of the list of predefined options.

If the requested variable does not appear in the list, 4DScript code has to be entered. To this end, select the option 4DScript Expression from the list and click the square button to open a 4DScript Editor. You can now enter your own code. This is however not recommended for beginners.

Now you see why we added the Queue before the Sink: to use the pre-defined statement Output per hour (a Sink has no output!).



Picture 5-5: Monitor variable

The standard setting for the *Sample rate* is 5 seconds. Change it to 60: measurements will now be taken every (simulated) minute. Because the average output is not likely to be restricted to whole numbers change the number of decimals to 1. Try *Auto adjust*, *Maximum value,* to develop a feeling for the various settings!

We choose 100 as maximum value as you can see in Picture 5-5. Start the simulation now.

Tip: Using a large number of monitor atoms with the sample rate set to the default value of 5 seconds can be the cause of models running very slowly! Remember that the monitor atom is being displayed every 5 **simulated** seconds, and is thus taking up a lot of processing time! This phenomenon even occurs when the model layout window is closed. In many instances, a sample rate of 60 seconds or even 3600 seconds is preferable. Therefore use the default sample rate of 5 seconds in monitor atoms with caution.

Questions and assignments

4. How many windows are leaving the factory on average per hour?

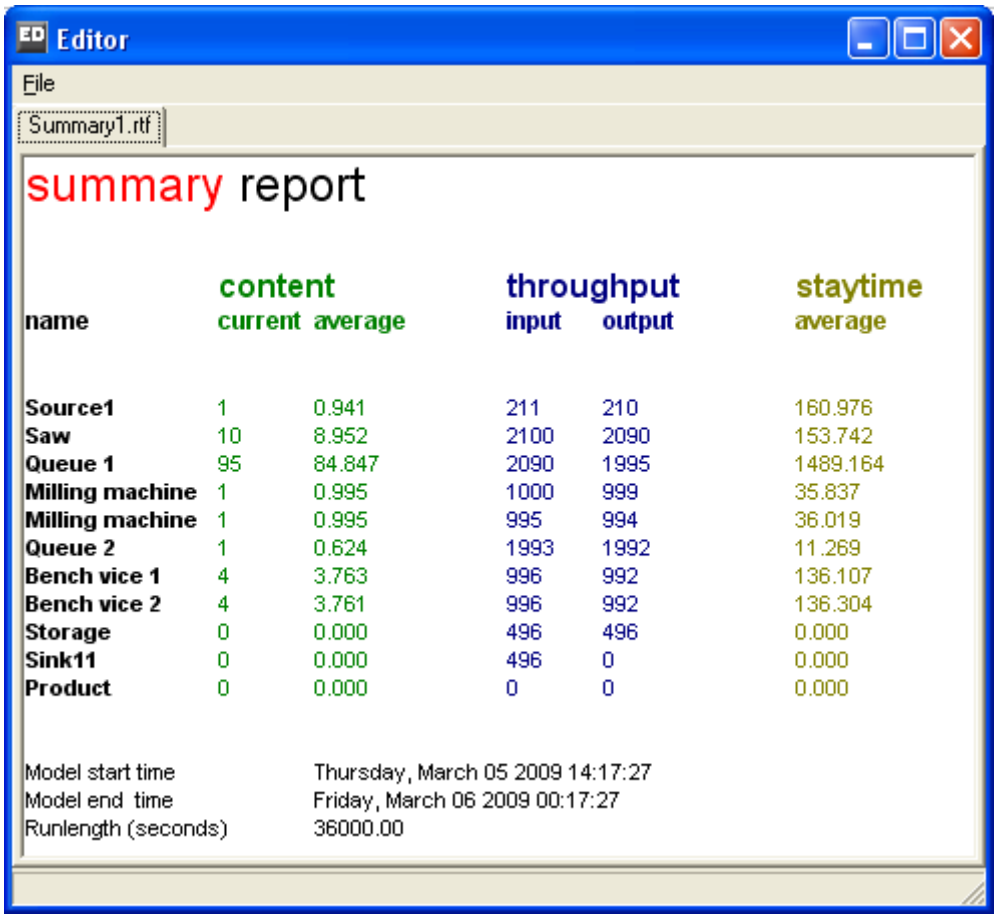
5.2.1 Reports and graphs

A third method to measure results is to use the option Summary reports and Graphs from the Results menu.

We will first deal with the option Summary reports. It enables you to get an overview of the status of the model, where a distinction is made between the following features:

- Current Content: the number of products present in an atom at the moment when the report is made.
- Average Content: the average number of products that were present in the atom.
- Throughput Input: the number of products that have entered the atom.
- Throughput Output: the number of products that have left the atom.
- Staytime average: the average time the products are spending in the atom.

By selecting the option Summary Report in the Results menu, a report such as in Picture 5-6 appears.



Picture 5-6: Summary report

Accordingly, you can read out of Picture 5-6 that a product is present on average 35.837 seconds in milling-machine 1, that 992 products have been produced in bench vice 2 and that 4 products are still present there.

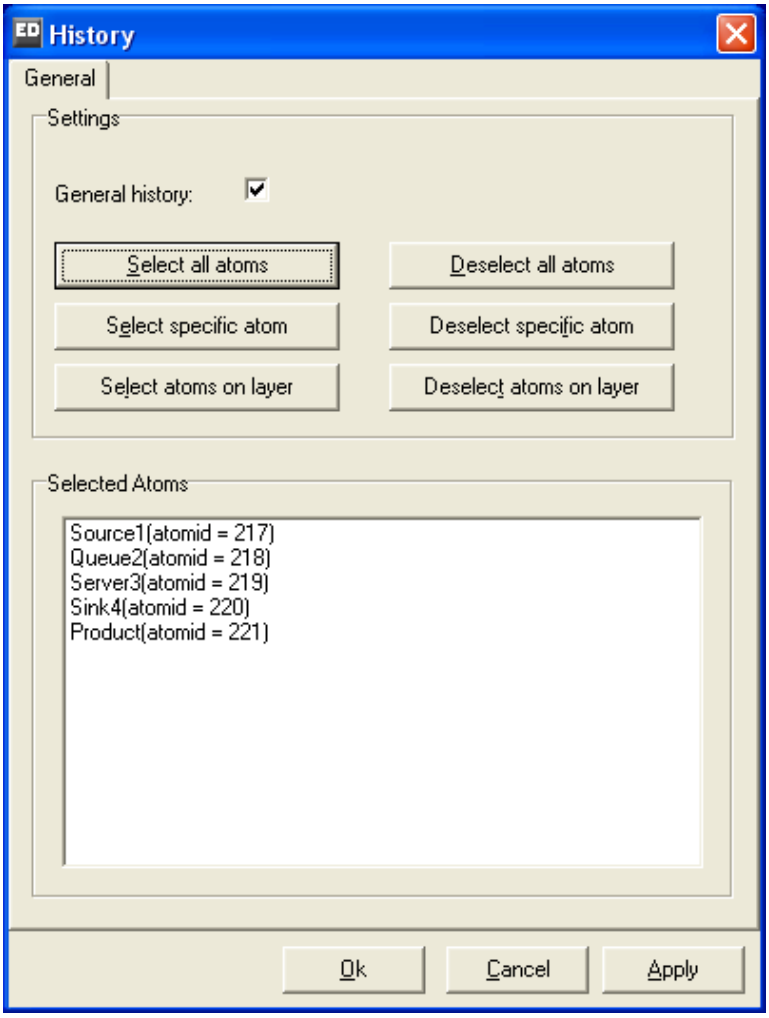
As you can see in the lines of the saw and the two bench vices, *all* measurements on servers with batch transitions are performed in the "smaller" unit, i.e. the units each batch consists of. For instance, look at the output of Source1 of 210, and the input of the Saw of 2100. The measurements on the saw therefore apply to short timbers rather than long timbers, and the measurements on the bench vices apply to the milled short timbers rather than window frames. This is a new measurement policy introduced with ED7. (The reasons are of no interest to anyone new in ED, but for the more experienced reader: it has something to do with the status Collecting for the B:1 servers, and the status Blocked for the 1:B servers. After thorough examination it was found that this policy was the only one that leads to consistent results under all circumstances. By the way, it *would* have been possible that the output (not the input) of the saw had *not* been a multiple of 10 (like 2098 instead of 2100), because Queue1 is full and is therefore blocking the entry of additional finished short timbers. In order to understand this, please realize that Queue1 will send short timbers to the two milling machines one by one).

Based on this report, measure that 49.6 windows on average per hour have been produced!

The option Graphs gives you a graphical representation of a variable. To apply this method, you need to switch on the History option for the atom that you want to represent in a graph. This is done by using the History option in the Simulation menu.

Warning! The option 'General history' (by checking) as well as the individual atom (via option 'One on') has to be selected (see also Picture 5-7). At first it might seem a good idea to turn history on for all atoms, 'All on'. However, recording history this way, especially for large models, will have the effect of slowing down the simulation unnecessarily.

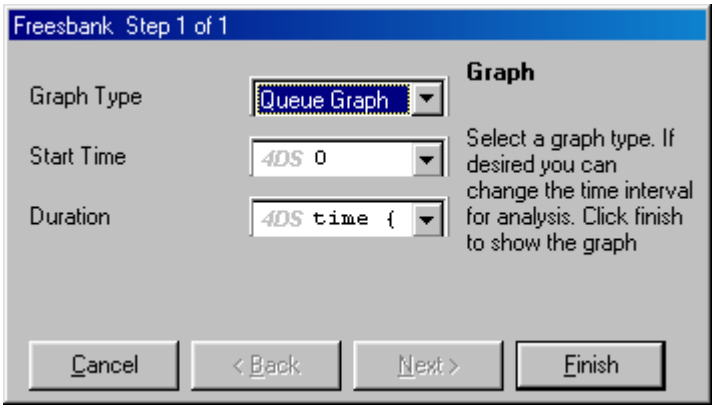
At this stage, we will only focus on the possibility of creating graphs by means of the menu option Graphs, because the atom Graph often needs to be programmed. The Graph atom might be convenient for advanced users, because all possible variables can be defined by using 4DScript.



Picture 5-7: History

If the general History is switched on and some of the atoms are selected in the history window each run data is collected of these atoms. This data can be visualized with the Graph option in the results menu.

By clicking on the option Graphs in the Results menu, a window appears where you can select the atom of which you want to make a graphical representation. If an atom is selected and history for that specific atom was recorded, a window such as in Picture 5-8 appears. If no atom is selected a selector window appears. In that case first choose one of the atoms for which you have recorded history.



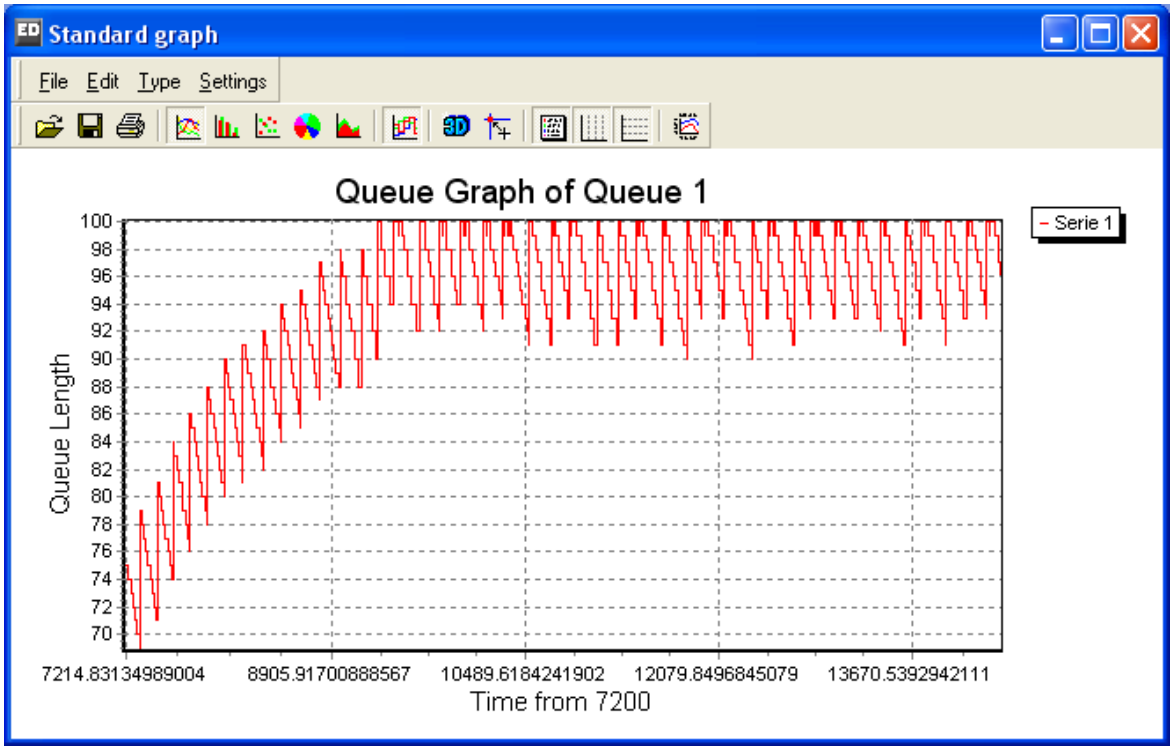
Picture 5-8: Input window Graph

We will first go over the last two input fields:

- Start time
  The starting time of the graph is to be entered here. If 0 is indicated, the graph will start from the beginning of the simulation run. If the value 10 is entered, then the graph will not include the first 10 seconds of the run.
- Duration
  In this input field, you indicate how long the graph will be recorded. For example, if the value 3600 is entered here, the first hour after the starting time of the graph will be shown. The default value is “time”, the number of simulated seconds that have passed when the simulation run was stopped.

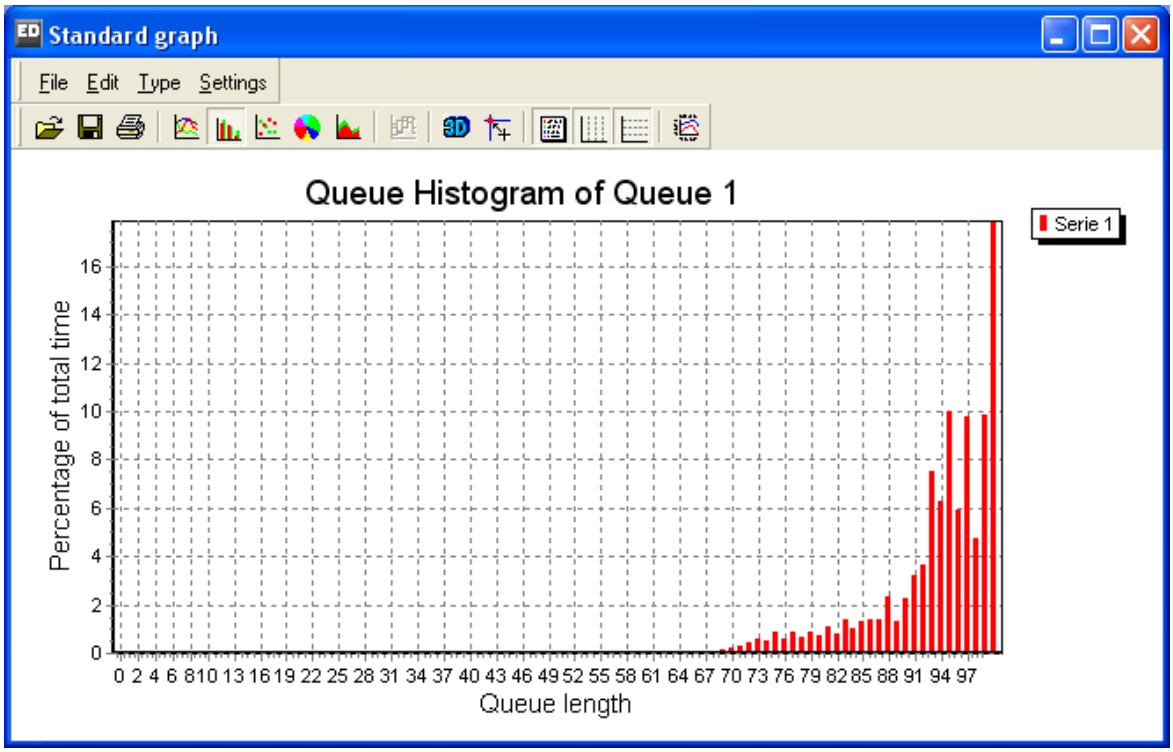
The first input field, Graph Type, is where you can select the type of graph desired. There are 5 possibilities:

1. Queue Graph
   With this function, you can make a graph of the number of products present in an atom. In a Server atom, the number of products will as a rule never be higher than 1, but in a Queue atom, the queue is being conveyed in the graph. Picture 5-9 represents a graph of the first buffer from the carpenter’s factory case study, measured between 2 and 4 simulated hours after the start of the simulation.



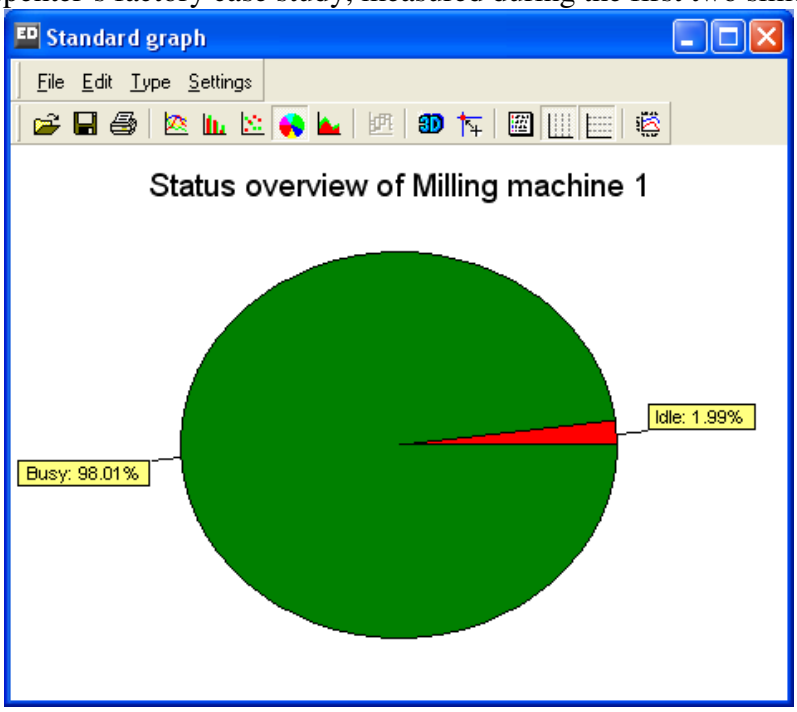
Picture 5-9: Queue graph

2. Queue Histogram
   The percentage of time each 'storage level' takes up is reflected here.
   See Picture 5-10, displaying the distribution of different queue lengths of Queue1 measured between 2 and 4 hours after the start of the simulation.



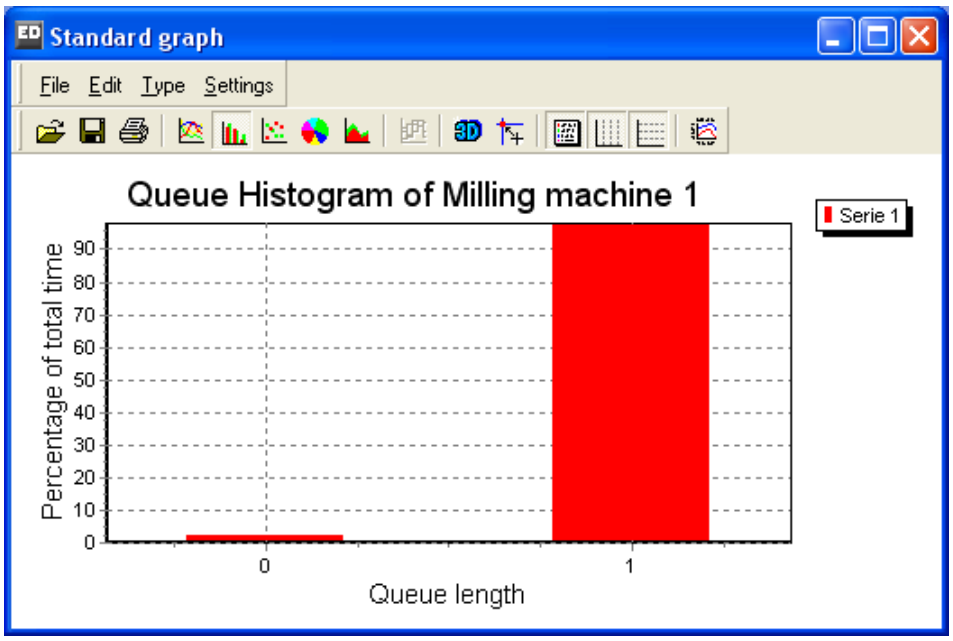
Picture 5-10: Queue histogram

3. Status Pie
   This function enables you to represent the status of an atom in a pie chart, such as in Picture 5-11, where a Status Pie reflects the status of the first milling machine from the carpenter's factory case study, measured during the first two simulated hours.



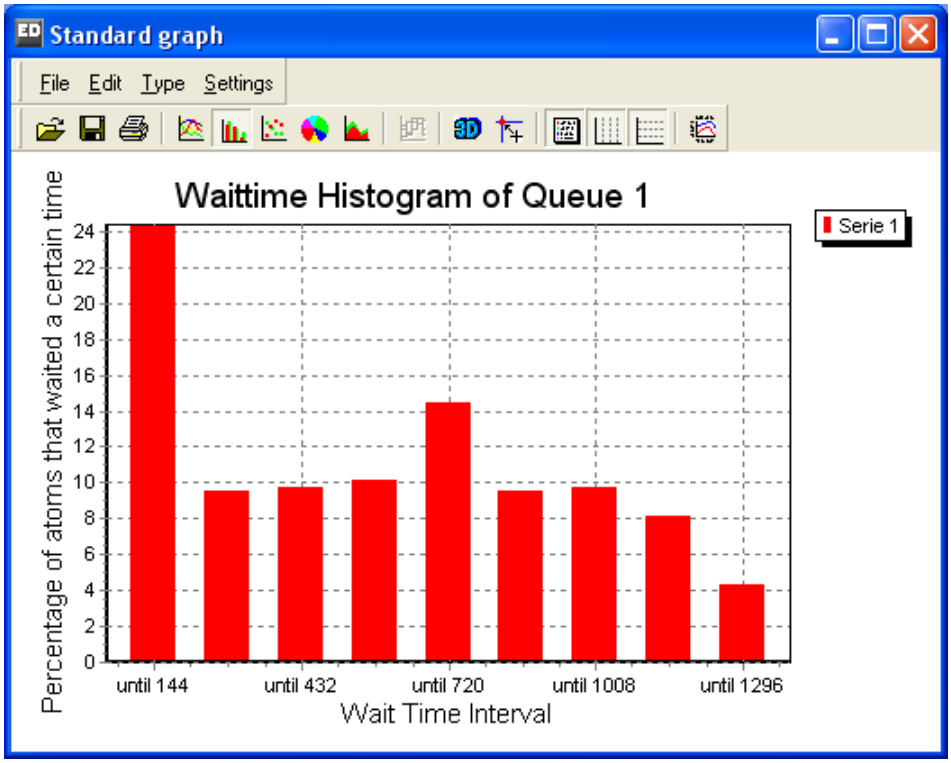
Picture 5-11: Status Pie

4. Status Bar
   With the option Status Bar, the status of an atom is also reflected, but this time in a bar chart. See Picture 5-12, which corresponds to the measurements displayed in Picture 5-11.



Picture 5-12: Status Bar

5. Wait Histogram
   The Wait Histogram function produces a histogram about a product's waiting time in the atom. See Picture 5-13, which displays the distribution of the waiting times of the short timbers in Queue1 during the first two simulated hours.



Picture 5-13: Wait Histogram

5.2.2 Setting up an experiment with the Experiment Wizard

The three above mentioned methods for measuring results are particularly useful to display results directly during the simulation. For drawing conclusions about the results of the modeling, these techniques are less appropriate. In this case, an experiment is the appropriate method to use. An experiment can be created with the Experiment Wizard, which can be found in the main menu under Experimentation.

Refer to Annex 4 or the Help | Tutorials for a full explanation on the Experiment Wizard and Analyze Results.

It is important to understand the major differences between the options Simulate and Results on one hand and Experimentation on the other hand: We use the first two options in the process of building and testing our model and for getting first results on *single* runs, while Experimentation is used later on in the process when we more or less trust our model and want results on *multiple* runs, because we can't draw conclusions from the outcome of a single run. For who is to say if the outcomes of another run (under the same conditions) will be about the same?

Case study assignment (continued)

Questions and assignments

5. Design an experiment with a warm-up period of 10 hours and a measurement period of 100 hours, in which the average window production per unit of time (hour, day or week or simulation period) can be determined with a 95% reliability. Does it correspond to your calculations made beforehand and to the results gained with the other techniques to measure results?

We will demonstrate how to solve this assignment step by step.

Basically there are four steps in experimentation:

Step1 *Definition of the Experiment Settings*
Defines experiment settings such as the number of runs, the length of the observation period and the warm-up period, etcetera

Step 2 *Definition of the Performance Measures*
Define the performance measures of atoms or a group of atoms

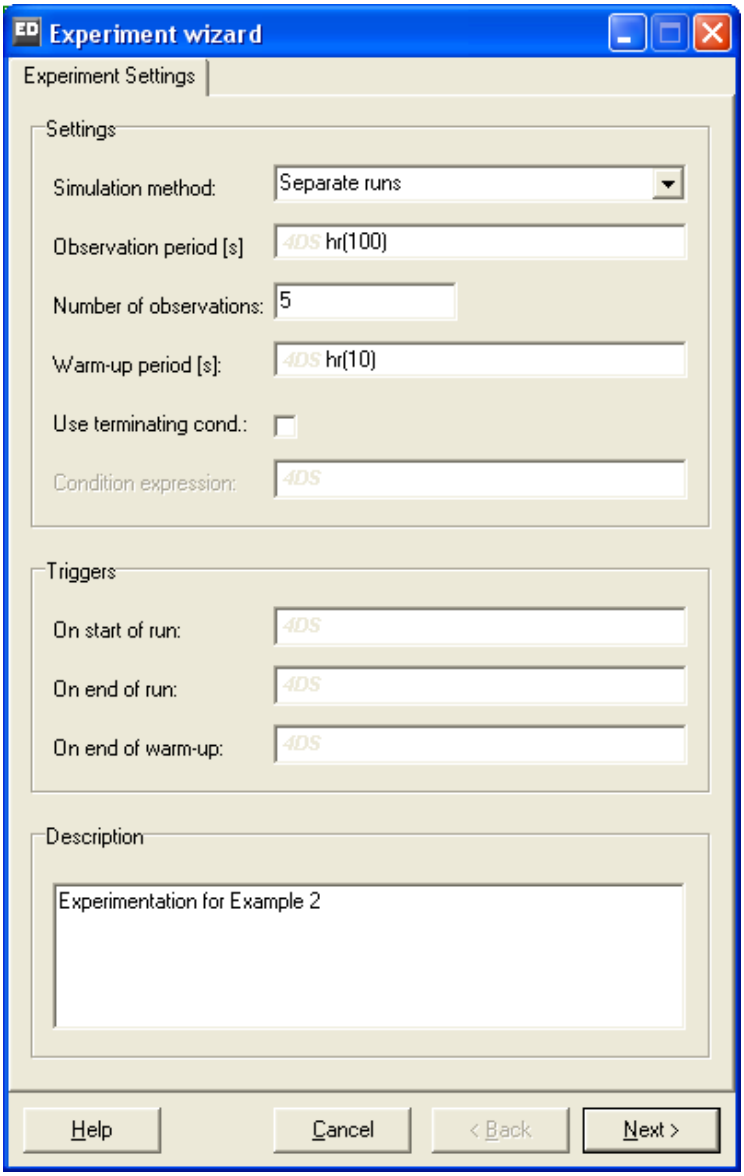
Step 3 *Experimentation*
Running the experiment according to its definition. This step requires no action from the user!

Step 4 *Report definition and analysis*

If the sample rate of your monitor atom is still set to 60 seconds, or even worse 5 seconds, this might be a good time to set it to 36000 seconds (10 hours), or even 1E9 (1 billion seconds, about 32 years), since you will not need the monitor atom during the experiment. It will make your model run considerably faster during the experiment.

Open the Experiment Wizard by clicking Experimentation in the menu bar: we start with giving the required figures for the observation period, number of runs and the warm-up period in the Experiment Setting (see Picture 5-14). If you are finished you can click on the 'Next' button.

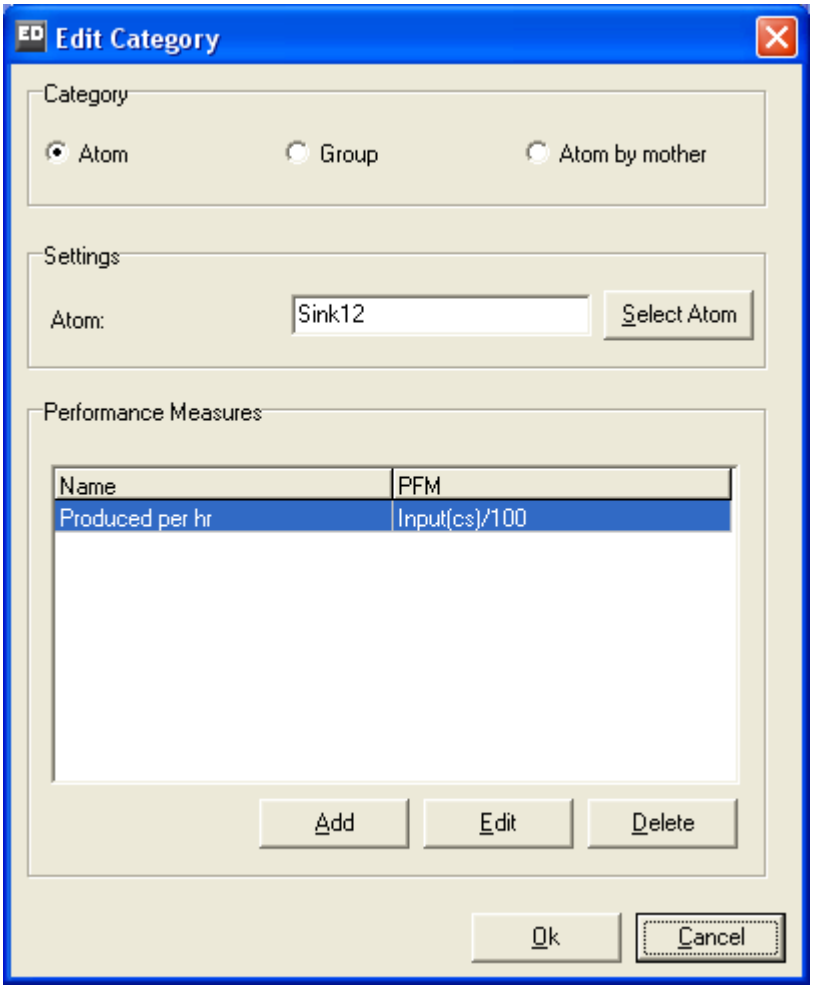
We are now ready to define the Performance Metrics (PFMs). In the 'Edit Category Window', see Picture 5-15, click the 'Add' button to create a new performance metric. We are interested in the capacity per hour of the production system. We can measure this by measuring the number of products that enter the Sink. Thus the output of the system can be measured by measuring the input of the Sink. To define a PFM on the Sink first press 'Select Atom' and select the Sink. We still need to define what we would like to measure at the Sink. Therefore click the 'Add' button in the Performance Metric section. We can now enter a name and select a performance metric from the list.



Picture 5-14: Experiment Settings

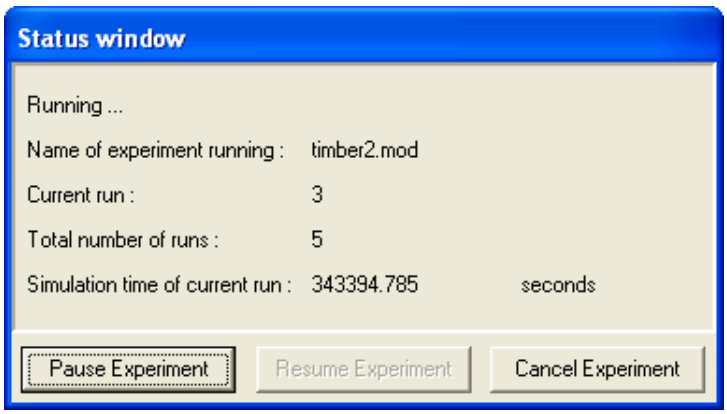
Define a PFM on the Sink named `Produced per hr' (see Picture 5-15). The input during the 100 hours is divided by 100 to get the number of windows per hour. Make sure you select the "[USER DEFINED]" option for your PFM, and do not forget to remove the string "[USER DEFINED] ", and type "Input(cs)/100". "cs" is the 4DScript code for the current selection, which can also be a group of atoms.

If both the name and the PFM are entered click the 'Ok' button. We can click the 'Add' button again to define another PFM on the Sink. In this case we only need a single PFM thus we can click again on the 'Ok' button. We can click the 'Add' button again if we need to define another PFM on a different atom. Click 'Next' after all PFM have been entered.



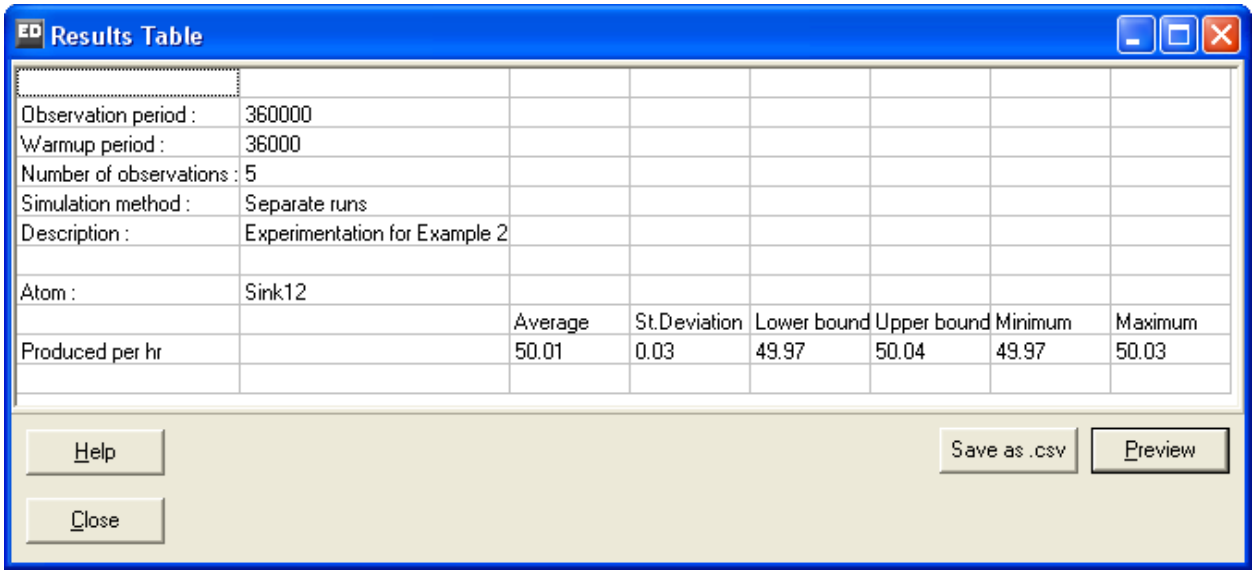
Picture 5-15: definition of the PFM

The actual running of the model –after start experiment- is done in step 3 (see Picture 5-16) and may take some time: In every run of 100 hours about 500 windows have to be made from 2000 short timbers and 200 long timbers!



Picture 5-16: The experiment in progress

After running the experiment it's easy to make a report. See Picture 5-17 for an example of a report. From this table the results can be exported to Excel (save Table as .csv) or put in a more fashionable way via QuickReport (see Report Preview).



Picture 5-17: The Results Table

This concludes our example about experimenting. For a thorough examination of what is possible we refer to our document on experimentation (see Help | Tutorials).

The model with this experiment is supplied under the name timber2.mod

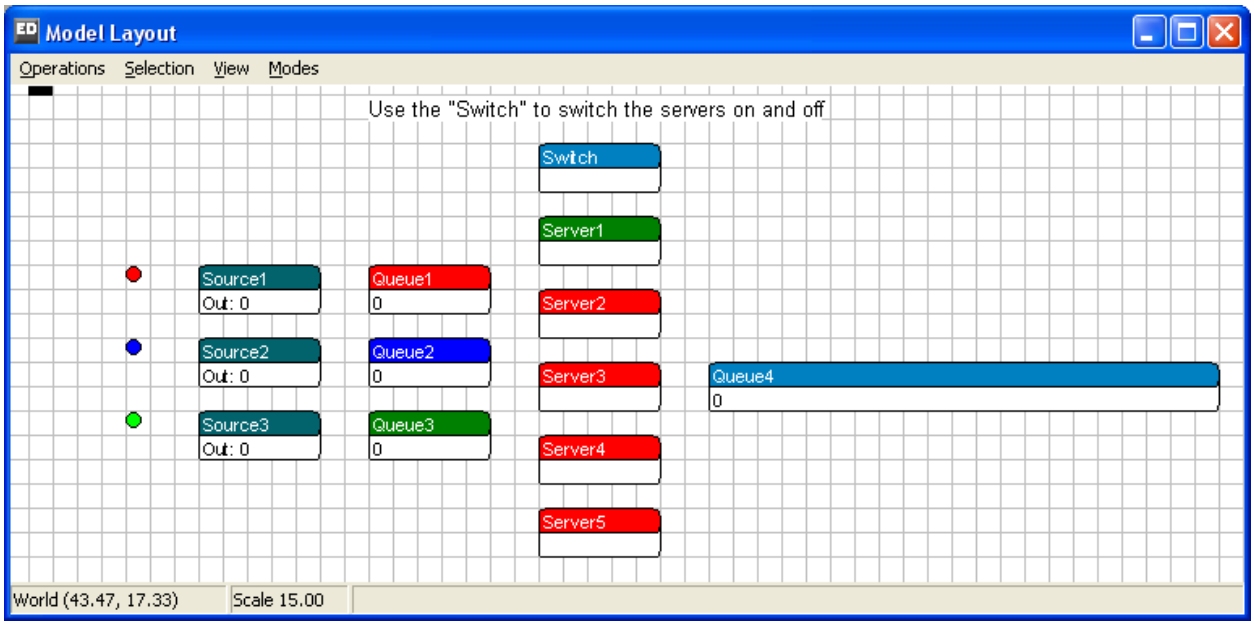
6 PLAYING WITH STRATEGIES

In this chapter, the emphasis lies on the pre-defined strategies with which products get access to a next atom (input strategy), are selected out of a queue (queue discipline) and are re-directed after processing (send to).

In order to demonstrate this, a simple model has been built. It consists of three Sources, three Locks (an atom regulating the products' supply), five Servers, four Queues and one Switch, which is a self-built atom (Enterprise Dynamics offers this flexibility). The purpose of this atom is to switch the Servers on and off. There is no Sink; consequently all products remain in the last Queue (see also Picture 6-1).

The model is supplied under the name strategy.mod.

Tip: The most important atoms and their functionalities, including these strategies, are fully described in Annex 2. Take a look at this Annex for this illustration and use it later as a reference when building your own models!



Picture 6-1: The layout of strategy.mod

Warning: Because the source code of the last queue atom has been adjusted and a new designed atom is used, beginners will not be able to build this model on their own. Advanced users can however build this model by using the standard atom and adjusting the code on the 2D Draw event handler on the last queue so that it can show up to 100 products in it. The atom behavior can be changed from the Atom Editor. Because the Switch is also not a standard atom, advanced users will have to switch the servers on and off manually. This is possible with the Availability Control atoms, among others.

From the three sources, three types of customers arrive at the queues. Their arrival is regulated in such a way that the three queues will get filled rather quickly. We have set the option 'Number

of products' on each of the sources to allow only 10 of each type of customer to leave the source. From the queue, the customers are helped by a server and after that end up in the last queue. The role of the *switch* is to switch the servers on and off.

In principle, all servers can be reached from the three queues, the first server can always be reached through channel 1, the second through channel 2, etc. However, in the original settings of strategy.mod, only server 1 has been switched on (by means of the switch). As a result, during a simulation, the first server is colored green while the others turn red. In addition, the default parameters are applied, which means that:

1. all input strategies are still on *Any input channel,*
2. all queue disciplines are still on *First in first out,*
3. all send to on the Queues are set to 4. A random open channel: choose a random channel from all the open output channels.

This means every Server available can be the next station from a Queue.

The products and their icons are visible in the last queue. As a result, the order in which the products have entered the queue is accurately visible (provided the queue discipline is set on FIFO).

Because the products receive a *label* when leaving the source, they can also be sorted out in the last queue on the basis of this label. Labels can be thought of as tags hanging on a product. They can represent a color, a weight, a size or something similar. They are defined in 4DScript, Enterprise Dynamics' programming language, and play an important role in most models.

Look at the other parameters of strategy.mod and carry out a few simulation runs to find out more about it.

After that, we are now really ready for experimenting with strategies…

6.1 Adjusting the input strategy

The input strategy regulates the access to an atom from preceding atoms (i.e. those with an output channel connected to one of its input channels). The role of the input strategy is to define the order in which products will be accepted from the available channels.

You can think of the input strategy to the sequencing of traffic lights where some roads have a greater proportion of time at Green than more minor roads.

The first three input strategies open all input channels and the last two open one input channel each time.

As an exercise, we are now going to look at the effect of the input strategy change on server 1 in the unaltered initial model. Change the input strategy as specified below and consider what is happening to the performance of the systems after observing a few runs.

1. *Any inputchannel*
   When activated, this strategy opens all input channels of an atom. If more than one of the atoms that are connected via the input channel can be forwarded, the atom with the

lowest number as input channel will have priority. As long as the products keep entering through the first channel, the other channels will be blocked.

2. *Largest queue*
   When activated, this strategy opens all input channels of an atom. If more than one of the atoms that are connected via the input channel can be sent, the atom with the longest queue or largest contents will have priority. Note that in the case of several equally long queues, the input channel with the lowest number is always chosen.
3. *Longest waiting*
   When activated, this strategy opens all input channels of an atom. If more than one of the atoms that are connected via the input channel can be sent, the atom with the highest waiting time will have priority. In the case of several atoms with an equal waiting time, the input channel with the lowest number is always chosen. Note that it does not mean that the queues get approximately equally long, as is the case in the previous option.
4. *Round robin*
   This strategy first opens the first input channel and then waits for a product to be sent through this input channel. In the second cycle, it is the turn of the second input channel etc. When the products have run through the last input channel, the procedure is resumed with the first one.

   Important remark: this strategy becomes active after the first product! So, in case of three input channels this strategy gives x,2,3, 1,2,3, 1,2,3 where x can be 1,2 or 3!
5. *Channel **1***.
   In this case, you can enter a specific input channel through which all products must enter. If **1** is entered, the products may enter only through input channel 1. Note that this rule is not valid for the first product entering as all channels are open initially.

6.2 Changing the Queue Discipline

The purpose of this section is to change the Queue Discipline of the last queue and to observe the consequences.

Change the input strategy on the first server to Largest Queue. This will ensure that the products enter with as much disorder as possible. Leave the other four servers switched off, because when five servers are switched on, it is harder to see how the atoms are moving. The six possible parameters are mentioned below (see also Annex 2: the queue atom):

1. *First in first out*
   The atoms are put in the queue according to their order of entry.
2. *Last in first out*
   The entering atoms are placed at the front of the queue. Consequently, the products leave the queue in reverse of their order of entry.
3. *Random*
   This queue discipline places the incoming products in a random spot in the queue.
4. *Sort by **Label** Ascending*
   The products with the lowest value for a specific label are placed ahead in the queue. Test this option by sorting according to 'product' label, which is specific to all product atoms of this example. *Warning:* if the products are not sorted properly, the cause might be a space before or after the label name.

5. *Sort by **Label** Descending*
   The products with the highest value for a specific label are placed ahead in the queue.
6. *User defined*
   The products are placed in the queue according to a position defined by the user. Try this out for the value 5.

6.3 Adjusting the Send to Statement

In this third part of the case, we will discuss a number of 'Send to' statements for the three queues towards the servers. With the 'Send to' statement, we define the output channel through which a product is to be sent. The user can enter a figure, or select one of the 21 (!) pre-defined options. For a complete overview of the Source atom description, we refer you to annex 2.

First, we switch on the five servers; otherwise there will be little to send. This will give them the color 'green' and allow the products to pass through.

Try out the following settings on the queues:

1. *Specific channel: always send to channel **1**.*
   The Product Atom will always be sent to a defined output channel. Enter for example channel 3.
2. *An open channel (First channel first): search, starting from the first channel, and send to the first open channel found.*
   The Product Atom is sent to the first open channel that Enterprise Dynamics finds. The search starts from the first output channel (channel number 1).
3. *An open channel (Last channel first): search, starting from the last channel, and send to the first open channel found.*
   The search starts from the last channel and the product is sent to the first open channel Enterprise Dynamics comes across.
4. *A random open channel: choose a random channel from all the open output channels.*
   Enterprise Dynamics selects a random channel. With long simulation runs, it results in equal utilizations of e.g. a group of servers.
5. *By percentage: **90**% of products go to channel **1**, the remaining percentage go to channel **2**.*
   A definite percentage of the products are sent to a specific channel and the rest to another channel. The user can enter the channels and the percentage. Try to send 75% to server 1 and the rest to server 5.
6. *By user: enter your own 4DScript expression resulting in a value between 1 and the number of channels:1.You can press the small button for the 4DScript editor.*
   The user writes a 4DScript code that results in the output channel. By clicking on the small triangle next to the text, the 4DScript editor appears. For example, enter 2.

This concludes the example relating to the strategies. Of course, there are many other possibilities, which can be defined using 4DScript, ED's underlying programming language.

7 MORE ATOMS: FROM ASSEMBLER TO UNPACK

In the previous chapters, we described the basic atoms. Subsequently we practiced model building and experimentation with simple models. In this chapter, we will introduce eight new atoms, with which more complex situations can be modeled. Furthermore, we will address in more detail the visualization possibilities in ED.

The atoms described in this chapter are:

1. *Assembler*
   This atom is used when several atoms are assembled in a new atom. The old atoms can remain stored (packed) or destroyed (assembly).
2. *Unpack*
   This atom can disassemble previously assembled atoms. If, for example, a container was filled using an Assembler, the Unpack atom can unpack it again.
3. *Container*
   The Container is an atom for storing or packing products. Examples are a box or pallet.
4. *Accumulating conveyor*
   The Accumulating conveyor is a transport system that also functions as a buffer. If the load in front of the conveyor is blocked, then loads will continue to move forward until their progress is blocked by another load that has accumulated. The Accumulating conveyor is used for modeling roller conveyors.
5. *Non-Accumulating conveyor*
   The Non-Accumulating conveyor is very similar to the Accumulating conveyor, but does not continue to move products forward if progress is blocked. The distance between products thus always stays the same. The Non-Accumulating conveyor is used for modeling chain conveyors.
6. *MultiService*
   The MultiService atom functions as a group of parallel Servers: it has the basic functionalities of the Server atom and allows the simultaneous processing of several products or individual product processing.
7. *Lock*
   The Lock atom lets only a pre-defined number of products through. All following products are blocked.
8. *Unlock*
   When a product exits the Unlock atom, the Lock atom can allow a new product through. This makes it possible to control the amount of work in process in the model or the number of transport means.

All the above atoms are described in the following example. A full description of these atoms can be found in Annex 2.

Example 3: Stacking and wrapping

In the dispatch department of a factory, products are stacked on a pallet. The products and pallets arrive via two separate conveyors at this pallet stacker or pallet-stacking robot. This robot can stack the products on top of each other as well as next to each other, irrespective of their size.

From the robot, the pallet stacked with products moves via the next conveyor to a wrapping machine, which wraps the products in plastic. The wrapping machine can wrap the products from several pallets simultaneously or independently of each other.

The system is made up of the following elements:

1. Every 5 seconds, a product arrives at the pallet stacker via a roller conveyor.
2. Via a second roller conveyor, one pallet on average arrives every 40 seconds, according to the negative exponential distribution.
3. The pallet-stacking robot always places 8 products on the pallet and needs exactly 20 seconds per pallet, provided all products are present.
4. A chain conveyor transports the full pallets to the wrapping machine.
5. The wrapping machine can wrap a maximum of 4 pallets simultaneously in plastic foil. For each pallet, an average of 120 seconds is needed (using negative exponential distribution) to wrap the products on the pallet in plastic.
6. The length of each conveyor has no bearing on this matter: select one of about 10 meter's length. The speed is always 1 m/s. The products are 50 cm in length, width and height and the pallets are 1 meter long and wide.

We want to model this system in Enterprise Dynamics: for the roller conveyor we use an Accumulating conveyor, for the chain conveyor, we use a Non-Accumulating conveyor. Further, for the pallet stacker we use the Assembler, for the wrapping machine, the MultiService atom, and for the pallet we use the Container.

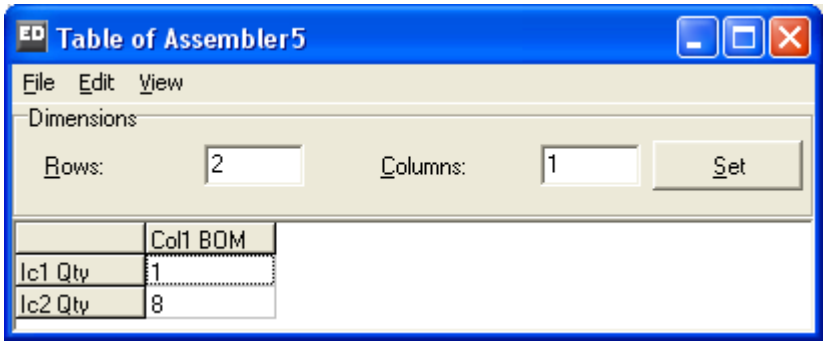
But... before you start, work out quickly whether this system would be able to handle the incoming load!

Most atoms can be placed in the model without problems because they are connected with other atoms in a normal manner.

With the Assembler, the pallets must enter the atom via the first channel (visualized by a small square) and the products must enter via the second channel.

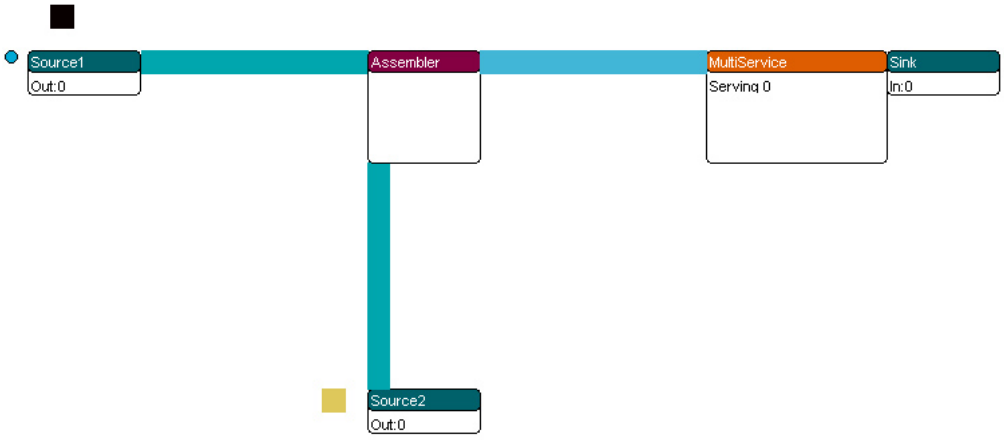
Double-click on the Assembler to define the number of products per pallet. This will display an input window with various tab sheets. The General tab sheet allows you to enter, for each entry channel, how many atoms are to be assembled by pushing the button "Edit B.O.M. table" (see Picture 7-1). See Annex 2 for a full description of all atoms used, including the Assembler itself!

The Assembler stacks the products on the pallet automatically. The space reserved for each product is defined in the Container atom. In order to avoid having all products stacked on top of each other in one high column, the size of the product must be adjusted in the Product atom. To this end, see the solution worked out in Example 2!



Picture 7-1: Bill of Material of the Assembler

When all atoms are placed in the model, the layout of the model will look like the one shown in Picture 7-2.



Picture 7-2: Layout of the conveyor system

Save your own model. If you are not sure that your own model is correct, you can also use the model supplied under the name conveyor1.mod in the following sections.

Let the system run for a while and see how it works in 3D. Play around with the sizes of the products or the Container and see what happens! Do the same with the speed of the conveyors or the processing times: for example, defining a much longer processing time of the MultiService atom will cause a buildup for the wrapping machine, and give you a clear picture of how the Non-Accumulating conveyor works.

Questions and assignments

1. For controlling the workload, we do not want to work with more than 5 pallets at a time in this system. Please adjust the model accordingly.

This is achieved by placing a Lock and an Unlock atom in the model. The Lock and Unlock atoms are inserted in the middle of the production process: the Lock atom is inserted at the spot where the restriction starts, the Unlock atom, where it ends. The Lock atom shuts down by itself after a pre-defined number of atoms have gone through. The Unlock atom reverses the blockade.

By inserting a Lock atom alone, no more than e.g. five atoms can enter the system. By inserting an Unlock atom as well, we make sure that no more than five atoms are present in the system at the same time. The Lock atom is always placed behind the Source, because the Source would otherwise regard the Lock atom itself as a product and send it into the system. The Unlock atom is always inserted at the spot where the restriction of the number of atoms has to end again, e.g. behind a certain machine or just in front of the Sink. In this example, the Unlock would thus have to be placed directly in front of the Sink.

By inserting an Unlock atom, the Lock atom will open automatically when an atom leaves the Unlock atom. The Lock and Unlock atoms are connected automatically with another. However, should it become necessary to connect them manually, then the second output channel of the Lock atom must be connected with the second input channel of the Unlock atom. To see more clearly how the Lock and Unlock atoms work, the Lock atom can be set temporarily to close after 1 or 2 pallets already. See Annex 2 for a more detailed description!

Adapt the model now or open the file named conveyor2.mod

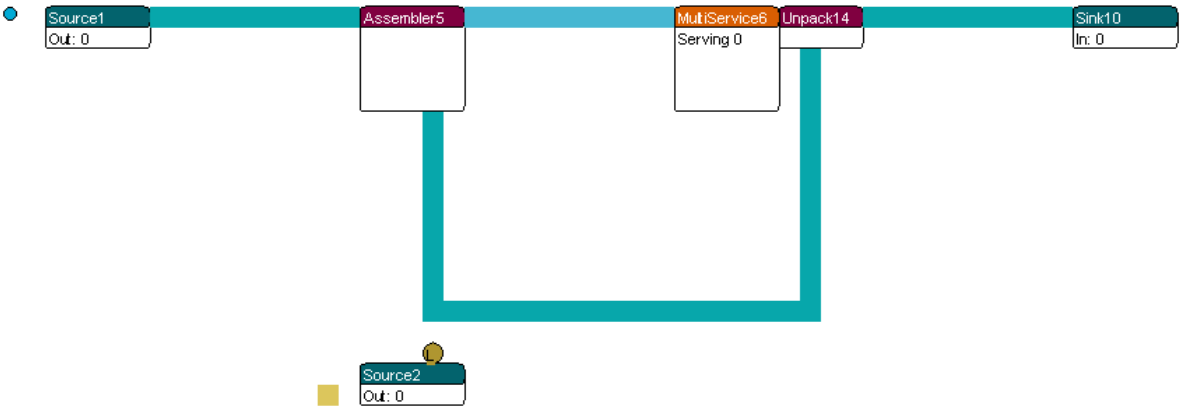
Questions and assignments

2. The company now wants to separate the pallets from the products after the wrapping process and move them back to the pallet stacker using an additional conveyor system. The number of pallets in the system stays limited to 5. Adjust the model accordingly.

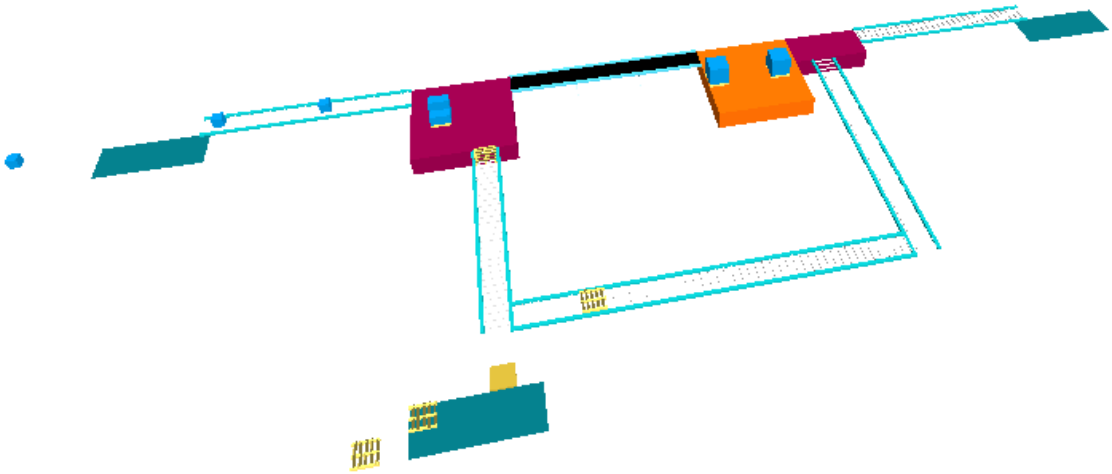
To make re-use possible, two major adjustments have to be made: a) separation of product and pallet, and b) transporting the pallets to the Assembler. For separating the pallet from the products, the Unpack atom is needed. Placed behind the MultiService atom, it sends the products to the first output channel and the pallets to the second output channel. By connecting a roller conveyor to this second output channel and connecting it to the existing roller conveyor, the pallets can then be re-used.

As the pallets no longer leave the system, the Unlock atom is superfluous. We could also leave the Lock out and set Number of Products to 5 on the Source. When the extra roller conveyors are included in the model, it could look like the one illustrated in Picture 7-3 and
Picture 7-4 (conveyor3.mod).

The length of the conveyors is set in a way, which facilitates an immediate visualization of the pallets' cycle.



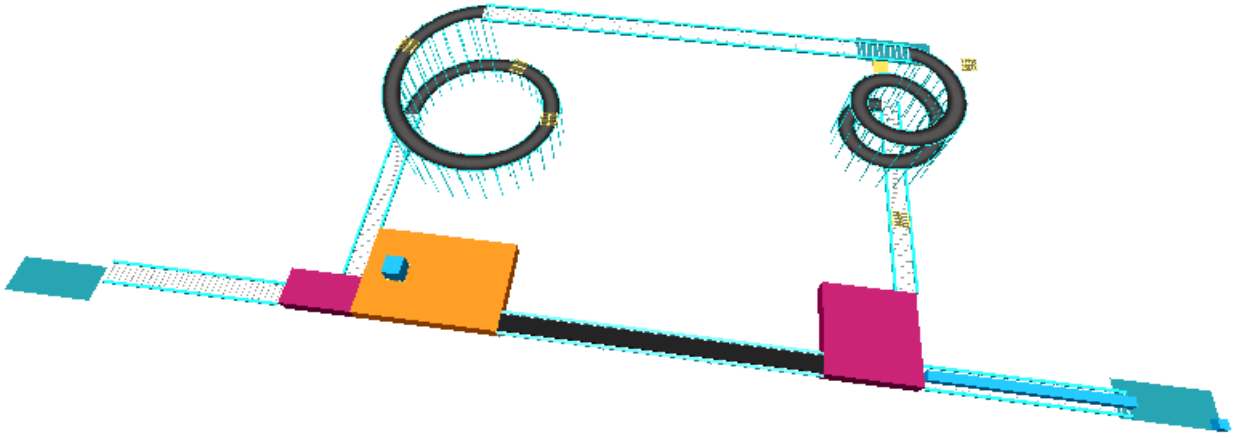
Picture 7-3: Re-use of pallets in 2D



Picture 7-4: Re-use of pallets in 3D

Instead of square conveyors you could, of course, also use curved conveyors. In Picture 7-5 the layout is mirrored and depicts curved, height-adjustable conveyors.

Both models can be found under the name conveyor3.mod and conveyor4.mod respectively.



Picture 7-5: Height-adjustable conveyors

We end with a few questions regarding conveyor3.mod:

Questions and assignments

3. How much time does an average pallet cycle take (excluding waiting times)?
4. Too few pallets in the system limit the hourly production, while – from a certain moment – extra pallets have little, if any, effect on the hourly production. Explain!

Determine the optimal number of pallets in the system through experimentation. Hint: Examine the average hourly production via a Generic Monitor on the conveyor before the Sink and vary the number of permitted pallets with the Lock or the Source.

8 ENTERPRISE DYNAMICS AND EXCEL®

From within Enterprise Dynamics, we can access other software programs such as Excel, Word or Access for importing or writing files to disk. In this chapter, we will discuss a frequently used link between ED and Excel by building on the earlier post office example.

We will further introduce a number of 4DScript commands and the important concept of *labels* in ED.

By the end of this chapter, the user will have learned to access Excel and, moreover, will master the first basics of 4DScript.

8.1 The bank

Example 4
A bank in Rotterdam functions as experimental subject for testing new customer processing concepts. There are several customer types with different service times.

For illustration purposes, we will limit the number of customer types to two in this example, so that 50 customers on average per hour, with a service time of 1 minute, arrive at the bank. We will indicate these customers as type A customers. In addition, another 5 type B customers arrive on average, with a service time of 10 minutes.

There is one single queue for both customer types, which is serviced by two counters, according to the 'first come, first served' principle.

The assumption is that all arrival processes are exponentially distributed and that all service times are constant.

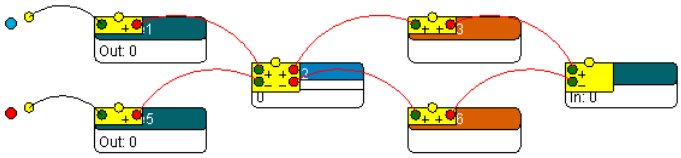
Questions and assignments

1. How high is the utilization of this system?
2. Applying the queuing theory, measure the average waiting times and queues, if both customer types had their own queue and counter (server).
3. Can you also make an estimate of these characteristics for the system with the single queue? Why so, or why not?
4. Which of the two systems would have the shorter average waiting time? Support your answer with arguments!

Try to answer these questions, which are of interest from a theoretical viewpoint; we shall return to them later. Users who are not familiar with the queuing theory on M/D/1 and M/D/2 systems can skip question 2 and 3.

For now, let's first build the model with the single queue. After that we will access Excel. See Picture 8-1 for the layout of the model with the channels.

To distinguish between the customers, type A customers are represented with a blue dot and type B customers with a red one.



Picture 8-1: Layout of the bank

The time needed by the bank assistant to help a customer must now be set on the Server. This service time, however, depends on the customer type, meaning that there is a choice between several formulas!

This problem has been solved by using *labels* of the kind that are – usually – hanging on a product (tags). These labels can represent a color, a weight, a barcode or a service time. A product can have several labels attached and the number of labels per product can differ as well.

The idea is now to set the service time at an early stage and 'attach' this time to a label that also indicates when the service time starts. The great thing about this concept is that it is not necessary to check for customer type on the Server, but only for customer label. Moreover, this concept can easily be applied to three or more customer types!

We hang a label named servicetime on each product on the 'Trigger on Exit' from the first and second Source respectively. This label contains a value for the correct servicetime:

```
Label([servicetime], i) := mins(1)      for type A customers
Label([servicetime], i) := mins(10)     for type B customers
```

Important!
The need for a closer study of the underlying structure of ED and the programming language 4DScript is clear. In Annex 2, under 'the Server', the use and syntax of the commands Label is explained.

However … for more insight into the syntax and the structure of 4DScript, this would be the time to work your way through the English version of Annex 3!

Tips!

1. Keep referring to this annex later in order to systematically increase your knowledge of 4DScript. Taking the step to pick up the manual with over 1000 commands will then become a good deal easier.
2. Double-clicking on fields into which 4DScript can be entered, activates the 4DScript editor. The F2 key gives access to a list with 4DScript terms and a short explanation of each command.

The required label is called up in the cycle time of the Servers and used as service time: Label([servicetime], First(c)). So, instead of entering a number directly, a 4DScript command is used which produces a number.

See Annex 3 for the syntax of the Label command and an explanation of the meaning of First(c).

The complete model can be found under the name bank1a.mod. This model can easily be converted to a system of separate queues (bank1b.mod).

Questions and assignments

5. Estimate the average waiting times in both systems based on a number of 100-hour runs. Do the answers correspond with your intuition?

8.2 The link with Excel

Instead of relying solely on the statistics in ED, we export the raw data such as waiting times and throughput times to Excel. With the analyses options in Excel we can then determine e.g. averages.

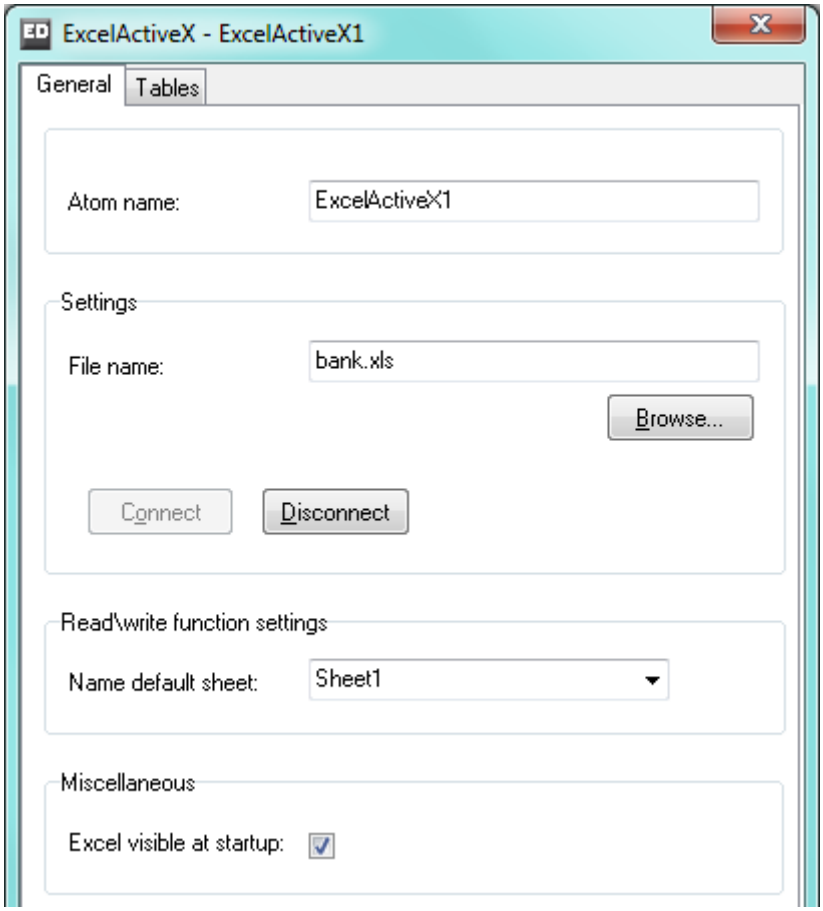
General procedure

1. Create a new Excel file in the same directory as your model.
   In most cases this will be *C:\Program Files\Enterprise Dynamics\Work*.

   In this example, the file is named bank.xls.

2. Place the ExcelActiveX atom into the model. This atom can be found in the library under the category 'Data'. This atom then 'organizes' the link between ED and an Excel workbook. . The ExcelActiveX atom does not need to be connected via channels with other atoms.

3. Double-click on the ExcelActiveX atom. The following screen appears:

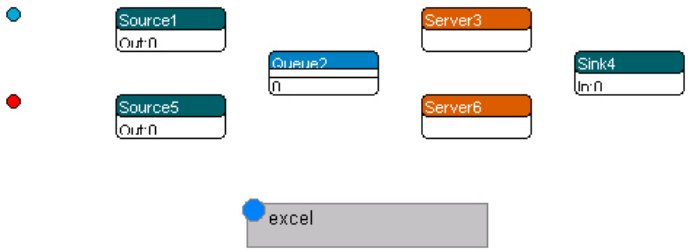


Picture 8-2: The Excel Communication window

Browse to select the newly created Excel file or to look up an old one.. Once you're ready, click on Connect and the Excel file opens and the Excel icon appears on the taskbar. Then you can select a default worksheet of the Excel file. The name of the ED worksheet must correspond with the name of one of the worksheets in the Excel file! The *ExcelActiveX_Read* and *ExcelActiveX_Write* function will read from and write to the default sheet if no sheet is given as a parameter. Switch `Excel visible at startup' on to see Excel on the background of Enterprise Dynamics.

The link has now been established and – when zoomed out far enough – can be seen in the Excel atom. Closing the model also disconnects the link with Excel. Re-opening the model automatically establishes the link; steps 1 to 3 therefore do not have to be repeated manually each time!

If everything has gone correctly, the following layout has been created:



Picture 8-3: The bank with an Excel link

In case of any problems with the link, refer to **Error! Reference source not found.** 'Troubleshooting' first!

8.3 Writing data to Excel

To see how the link works in this example, we write each customer's waiting time to Excel. This is done using the following command on the 'Trigger on Exit' of the Queue: ExcelActiveX_Write(Output(c), 1, Age(i)).

ExcelActiveX_Write(a,b,c) writes the result of expression c to cell(a,b) of the default sheet in the Excel file. The *Age* command keeps track of a product's age from the time of its arrival in the model. Here, this arrival time is measured from the moment the product leaves the Queue and therefore matches the waiting time of the product! The *Output* command indicates how many products have gone through the respective atom at that moment and functions as a teller. Summarized: the waiting time of customer i is written to row i, column 1 of the Excel file bank.xls.

Now set the speed of the simulation to a low rate and reset the simulation. Then start a run and use the taskbar to switch from ED to the Excel file. If everything went correctly, something 'wonderful' will happen: *parallel to the process in ED, the waiting times are written to the first column of bank.xls!*

Just think about that for a moment: what is possible with *one* bit of data can also be done with a lot more data. This means that it is possible, quite easily, to store large amounts of raw data and write the data to Excel, and subsequently unleash the entire arsenal of available Excel tools.

For those users whose screens shows nothing or merely display a large number of #####. This problem sometimes occurs when the column width is too small for displaying all the data in a specific cell. Making that particular column wider will solve this problem.

For writing throughput times to the second column, the following command suffices on the Trigger on Entry of the Sink: *ExcelActiveX_Write(Input(c), 2, Age(i)).*

Warning: the waiting time of customer 100 is shown in row 100, column 1, but the throughput time of the same customer 100 does not have to be shown in row 100, column 2. Explain why!

Questions and assignments

6. In Excel, measure the average waiting and throughput times of the first 1000 customers.

The models with the Excel link can be found under bank2a.mod and bank2b.mod. The Excel file is named bank.xls.

Questions and assignments (For enthusiasts)

7. Enter the waiting times of type A customers in column 3, the waiting times of type B customers in column 4, and measure – with sufficient customers – the average waiting time. Do the results correspond with earlier outcomes?

8.4 Reading data from Excel into ED

We have succeeded in linking ED and Excel and writing output data to Excel but … reading data from Excel into ED is something that also occurs frequently. How this is done is the last subject we shall illustrate now.

In the context of our example, we may well have actually measured customer inter-arrival times available. This arrival pattern is what we want to read into ED and use for a simulation run.

Using a formula, generate 100 (positive!) figures in the first and second column of postoffice.xls. You can, of course, also use the waiting times that have just been generated …

In both cases, it is useful to remove the earlier *ExcelActiveX_Write* commands.

In the Source input window under Inter-arrival time, enter the following for type A customers:
ExcelActiveX_Read(Output(c)+1,1).
For the Trigger on Entry of the Queue write: if(Output(c) = 100, CloseOutput(c)).

The ExcelActiveX_Read(a,b) command reads the value in cell(a,b) of the default worksheet of the Excel file, while the CloseOutput command ensures that the output of the Source atom shuts down after 100 customers, thereby ending the 'reading-in data process' as well. If this command is not performed, ED will continue to read in data and interpret empty cells as an inter-arrival time of 0.

Figure out yourself how the statements for type B customers should read!
The model can be found under the name bank3.mod.

In conclusion: arrival patterns are generated with a Source. For fixed production schedules or timetables, ED uses the ArrivalList atom. This atom can also be used for cutting and pasting (large) schedules in the proper format from Excel. The data is then available in an external table, which makes an open link with Excel, at least for data input, superfluous.

Reading data into ED through a table results in a significant increase in speed and is therefore highly recommended for most situations!

8.5 To pool or not to pool?

The results of the new post office show something remarkable: the model with a *single* queue (bank1.mod) has a longer average waiting time than the model with *separate* queues (bank2.mod)!

We found the following results:

average waiting times:	single queue (in minutes)	separate queues (in minutes)
type A customers	6.18	2.45
type B customers	6.18	25.7
average of all customers	**6.18**	**4.56**

Picture 8-4: Results of simulation runs

Just how is such a result possible? Aren't we taught all the time that combining means improvement?

After all, in the case with the separate queues it is quite possible that a queue with type A customers will form in front of the first counter, while the assistant behind the counter for type B customers twiddles thumbs. The answer is: *variation*!

'Combining' reduces average waiting times when there is one type of customers, but not when there are two types of customers, as is the case here.

Without going too deeply into the queuing theory: in the system with separate queues there are two independent M/D/1 models. For that purpose, a set of closed formulas is available for the most important queuing characteristics.

These formulas give the average waiting time (Wq) of type A and type B customers as:
Wq = 2.5 minutes and Wq = 25 minutes respectively.

The expected average waiting time of all customers therefore is 50/55 x 2.5 + 5/55 x 25= 4.54 minutes.

In the system with a single queue, a M/G/k system with two customer types is formed. In order to apply the formulas from the queuing theory we 'merge' the customers into one customer type. They arrive at an average rate of 55 per hour with an expected service time of E[S]=50/55 x 1 + 5/55 x 10 = 20/11 or 1.81 minutes.

The variance Var[S] of this service time is 6.69 and the variance coefficient C, defined as $Var[S]/(E[S])^2$, is 2.025.

Based on that, the waiting time is roughly – formulas are not given – Wq = 6.25 minutes.

The results of our runs pretty much correspond with these theoretical values:

The average waiting time in the separate queues is reduced by 27%!

The explanation? In separate queues, the irregular customers, whose service time is 10 times longer than that of the regular customers, no longer interrupt the waiting time of the regulars. Even though this results in a considerable increase in the expected waiting time of the irregular customers, the average waiting time of all customers is significantly reduced.

Questions and assignments

1. Adapt the separated queue model such that type A customers are allowed to jockey. This means that the first person in the queue for type A customers may use the counter for type B customers, provided there are no type B customers waiting. You can, of course, also experiment with jockeying between both queues.

   How does this affect the average waiting time of customers?

9 THE OPERATOR

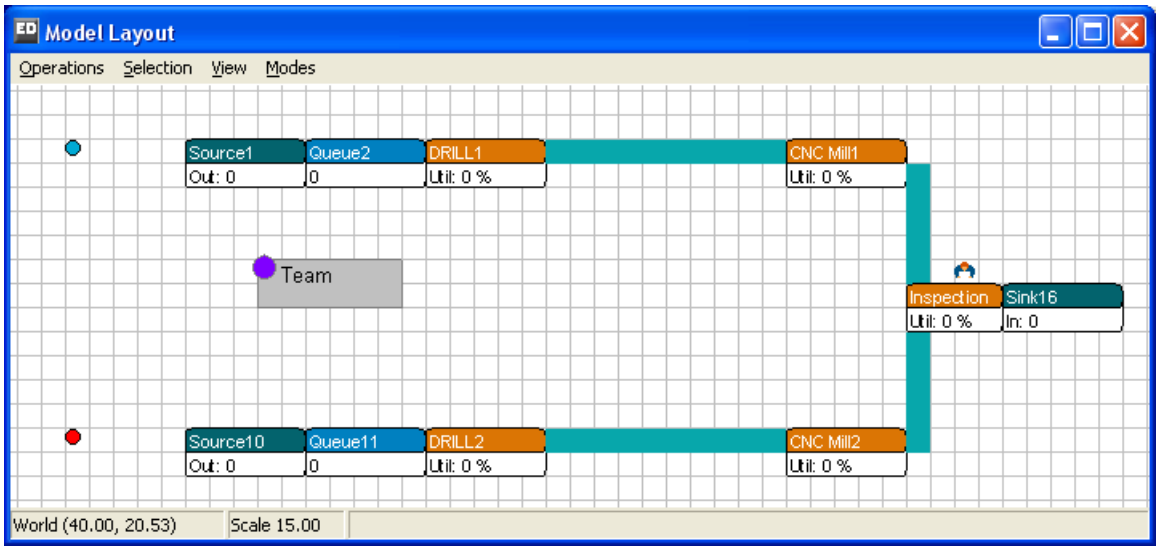
In this chapter the Operator takes central stage. The Operator is a capacity source needed on several processing sites. Though usually a man of flesh and blood, the Operator can also be a piece of equipment. We begin with an example where we need the Operator but where the time required for moving up and down between machines is negligible. Subsequently we shall deal with the different ways in which distances and movement is simulated in Enterprise Dynamics. The example will then be expanded with the free movement of the Operator within the flat grid and along mapped paths.

9.1 Getting started with the Operator

Example 5: The Operator for Red and Blue

Two semi-finished products, HalfBlue and HalfRed, arrive at two separate but identical production lines. Each production line consists of a drilling machine preceded by a storage facility. Once past the drilling machine, a ten-meter long conveyor transports the products to a CNC machine. From the CNC machine, the products continue on to final inspection on a five-meter long conveyor. Here both the red and the blue products are inspected according to the arrival sequence. After that the products leave the system. The semi-finished products arrive independently of each other with an average inter-arrival time of one hour. Drilling takes 20 minutes on average and the inspection takes an average of 6 minutes. All times follow an exponential distribution.
The CNC machines, on the other hand, are very constant and have an operating time of 10 minutes.

An Operator carries out the final inspection as well as the drilling of both products, which means that the Operator is constantly moving up and down between the drilling machines and final inspection. For now, let's assume that these walking times are negligible.



Picture 9-1: Production line for semi-finished products

In Enterprise Dynamics the layout of this production system can be seen in Picture 9-1. Go ahead and build it yourself!

Questions and assignments

1. How high is the (theoretical) utilization rate of the Operator?
2. Verify that the utilization level of the drilling machine(s), the CNC machine(s) and inspection (as site) is 33%, 16% and 20% respectively.

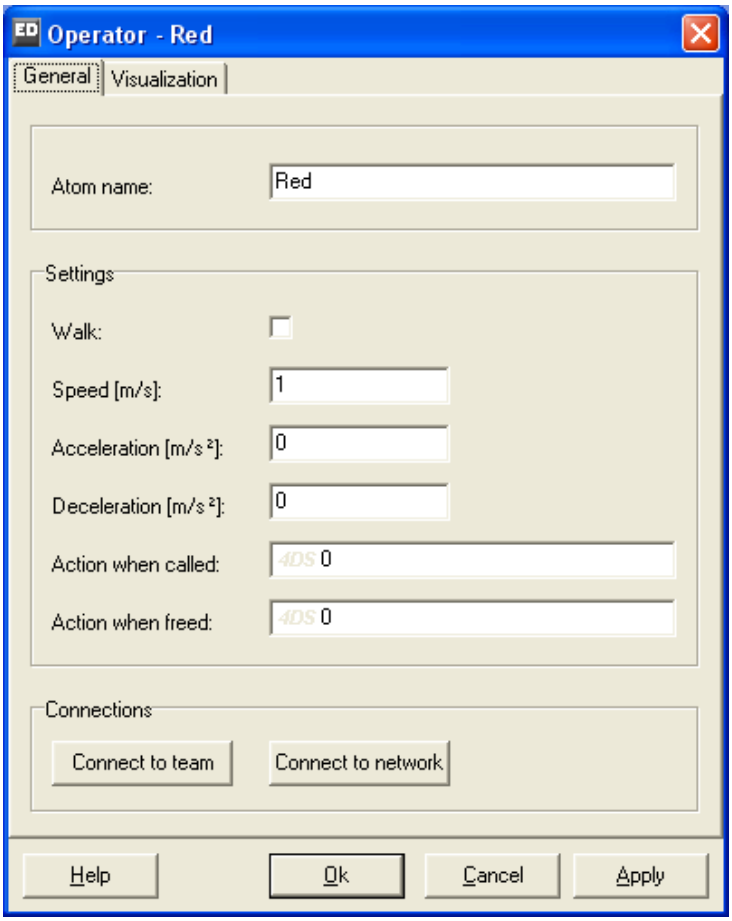
Check these results by letting the model run for quite some time: the results you get should roughly look like that!

The processing sites for both drilling and inspection are represented as Servers. These are, after all, operations that take a certain amount of time.
But it is insufficient nonetheless: the layout is right, but you still are left with one person walking between two drilling machines and inspection.

So the idea is to simply model the sites where the operation takes place in physical terms; in this case as a Server. When a product arrives at the drilling or inspection site, a call goes out for the assistance of an Operator. Only when the Operator arrives on the scene the operation can start. Once the Operator is no longer needed, he is released.
In Enterprise Dynamics this issue has been solved by means of the Operator and the Team atom. Each Operator is part of a Team. A Team can consist of one or more Operators.

We now drag an Operator and a Team atom into the model. Both atoms can be found in the library in the Operators category. By right-clicking or clicking on the Operator, a window opens (see Picture 9-2).

Push the Connect to Team button and then select the Team atom. The Operator is now part of the Team. Also the name of the Team has changed from e.g. Team23 to Team. This is not necessary, but it is useful for later references.



Picture 9-2: Input window for the operator

The location of both atoms in the Model Layout, by the way, has no bearing …

We still have to arrange that the Operator is called for inspection and drilling. Since this is necessary on arrival of a product, the obvious thing to do is to set this up in the Triggers. On every Server atom the default list is by

Trigger on Entry: CallOperators(AtomByName([Team], Model), 1)
Trigger on Exit: FreeOperators(AtomByName([Team], Model), i)

AtomByName searches in the model for the Team atom with the corresponding name; so if another name is being used it may have to be adapted. With *CallOperators* one Operator is called (by default); the user can change this number. With *FreeOperators*, Trigger on Exit releases all Operators (the atom(s) involved) again that are attached to the product.

Since both commands play a big role, the full syntax is listed below:

CallOperators(e1,e2,{e3,...,e24})
This command can contain 24 variables; variables 3 to 24 are optional. The following applies:
e1 = the reference to the team
e2 = the number of Operators required for a task
e3 = the priority of a task, where a higher number represents a higher priority
e4..e24 = the names of the required Operators

Example: CallOperators(AtomByName([Team],Model),2,1,[John]) requests 2 Operators with priority 1, one of whom has to be John. If there are more than 2 names, ED selects two names from that specific little list. The Operators are allocated to the first product of the atom on which this call statement is written. All events on this atom are delayed until the required number of Operators is present.

FreeOperators(e1,e2)
e1 = the reference to the team
e2 = the reference to the atom on which the task takes place

Example: FreeOperators(in(1,c),i) releases all Operators from the first product in the atom that is connected to the first input channel of the Team.

Open operator1.mod in the Tutorial Models or correctly adjust the Server atoms in your own model. Run the model to ensure that the Operator hops constantly from one drilling machine to the other or to the inspection site. Processing sites that want to begin with an operation but haven't an Operator at their disposal yet, turn red in color!

Requests to the Operator are processed according to their arrival sequence. This can lead to the model getting stuck in a so-called deadlock, as a drilling machine can no longer get rid of its products. Make a long simulation run in order to study this situation. Later we shall learn how such a deadlock can be prevented.

The utilization rate of the Operator can be established with the Experiment Wizard and will be in excess of 80% (see Busy status). Does this match with your own calculation?

In short; it seems as though the model can handle the flow fine, but that the Operator will have to work hard even though walking does not (yet) take time. And 'walking' will be discussed in the following chapter …

9.2 Moving in the model's dimension

The Operator can be moved in 3 ways

1. Relocating without time expense (default)
2. Walking
3. Walking by means of a network

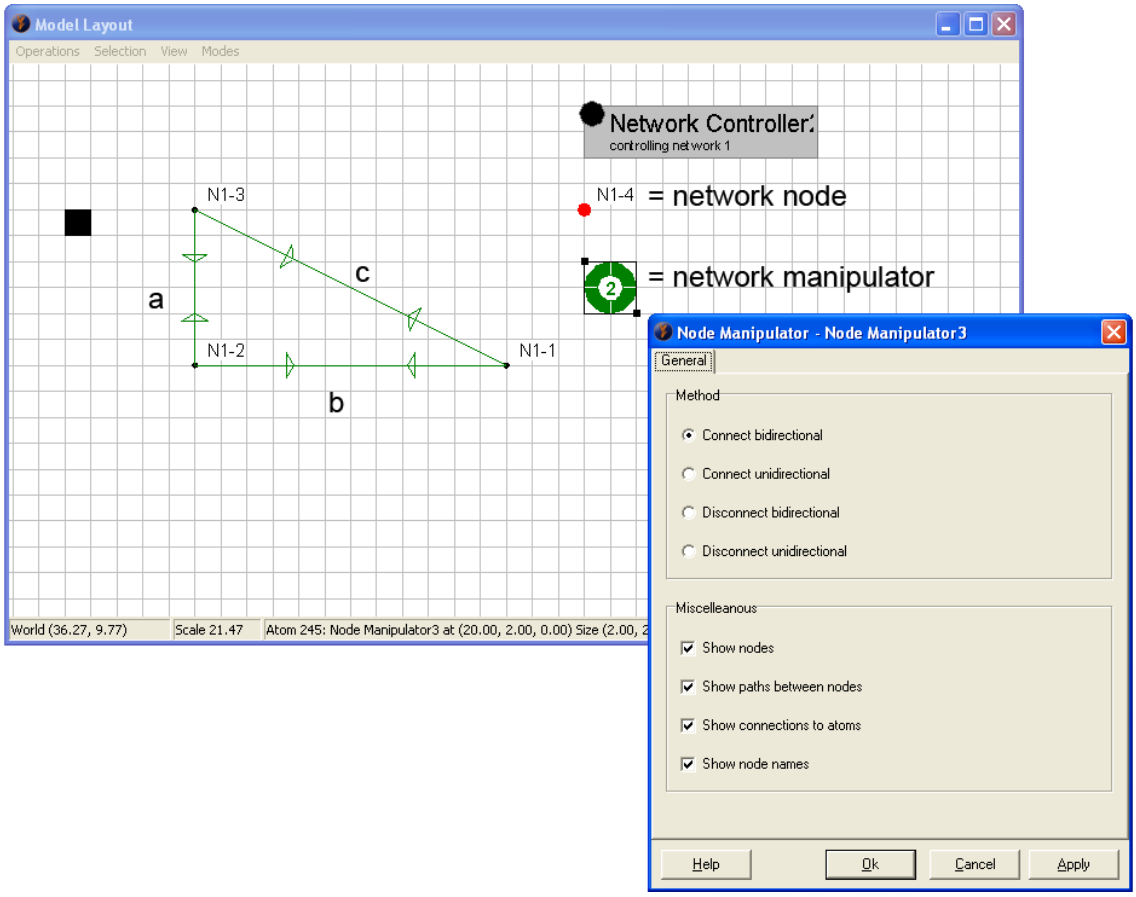
The first option is used when walking times or transport times are negligible in the problem. When using the second option, we introduce time used for transport as a consequence of the distance between two locations without worrying about how the route is covered. When using the last option, the Operator moves over a network of paths and junctions. This is intended for situations such as order pickers in warehouse aisles, but also for vehicles such as forklifts and AGV's, which are simulated in Chapter 10 with Transporters.

The complexity of the behavior of movement clearly increases with option one to three. As designer of simulation models, therefore always question whether adding transport times and transport networks are relevant within the framework of the problem!

Prior to studying the movement of the Operator in more detail, first some more information about distances and the use of dimension in Enterprise Dynamics. The sample models in this tutorial, with the exception of the conveyor in example 4, have so far not made use of the physical dimension. So what happens, when location, size and distances between places do play a role?

In Picture 9-3, we see the grid – always visible behind the atoms – in addition to a number of other atoms, which we shall deal with later. This grid functions as the grid for coordinates; each square measuring 1 by 1 meter. The visible black square corresponds with the coordinates (0,0) the upper left. This means that Enterprise Dynamics considers the north-eastern corner as reference point for the location of an atom.
Move around with the cursor in Enterprise Dynamics in order to get a feel for the coordinate system!

In Picture 9-3, we also see three points, N1-1, N1-2 and N1-3, represented in a triangle. Verify that (5,6) form the coordinates of N1-2.



Picture 9-3: Coordinates, distances & several network atoms

The distance between N1-1 and N1-2 is 5 meters, so if an Operator moves between these points with a default speed of 1 m/s, it will take him 5 seconds. Pythagoras' theorem applies to distance c: $c^2 = a^2 + b^2$. Verify that c =13!

9.3 Walking (free movement)

When distances play a role for the Operator, we can change the default setting `relocate without time expense' to `walk'.

By right-clicking on the Operator the input window for the operator appears (see Picture 9-2). Checking the option Walk makes the Operator move at a user-definable speed (default is 1 m/s) between his present location and a new location. The distance formula applied is the formula given by Pythagoras, meaning that the distance is determined in a straight line, disregarding any objects in between. This is 'free range' movement.

Attention!
Atoms differ in size. The decisive coordinates are always those of the point in the upper left corner of an atom!

A detailed discussion of the other fields of the Operator atom will follow later.

Example 5 (continued): The Operator for Red and Blue

Now adapt operator1.mod such that the Operator walks as described or open operator2.mod.

Questions and assignments: Run the model and note that:

3. The Operator does indeed move slantwise across the screen from one drilling machine to inspection and vice versa.
4. That the animation of the Operator at the Server differs from the animation in the previous model.

Calculate and check that the mentioned relocation in the model takes a little less than 21 seconds (√436 seconds, to be exact).

The utilization level of the Operator will rise also: one hour on average will be taken up by 40 minutes of drilling, 12 minutes of inspection time, but also by 1 to 2 minutes of walking time. This can be checked with the Experiment Wizard …

9.4 Networks in general

In the previous chapter we have looked at free-range movement in a straight line. But free-range movement is not always possible due to walls or other obstacles, and so movement often occurs by means of mapped paths.

In Enterprise Dynamics this has led to the concept of a *network*. A network is a collection of points and lines. It is possible to connect Operators or Transporters to such a network, and, once connected, they can move through this network only. The points form the junctions in the network; the lines represent the paths. These paths can be used one-directional or bi-directional.

For creating and embedding a network, we proceed along the following steps:

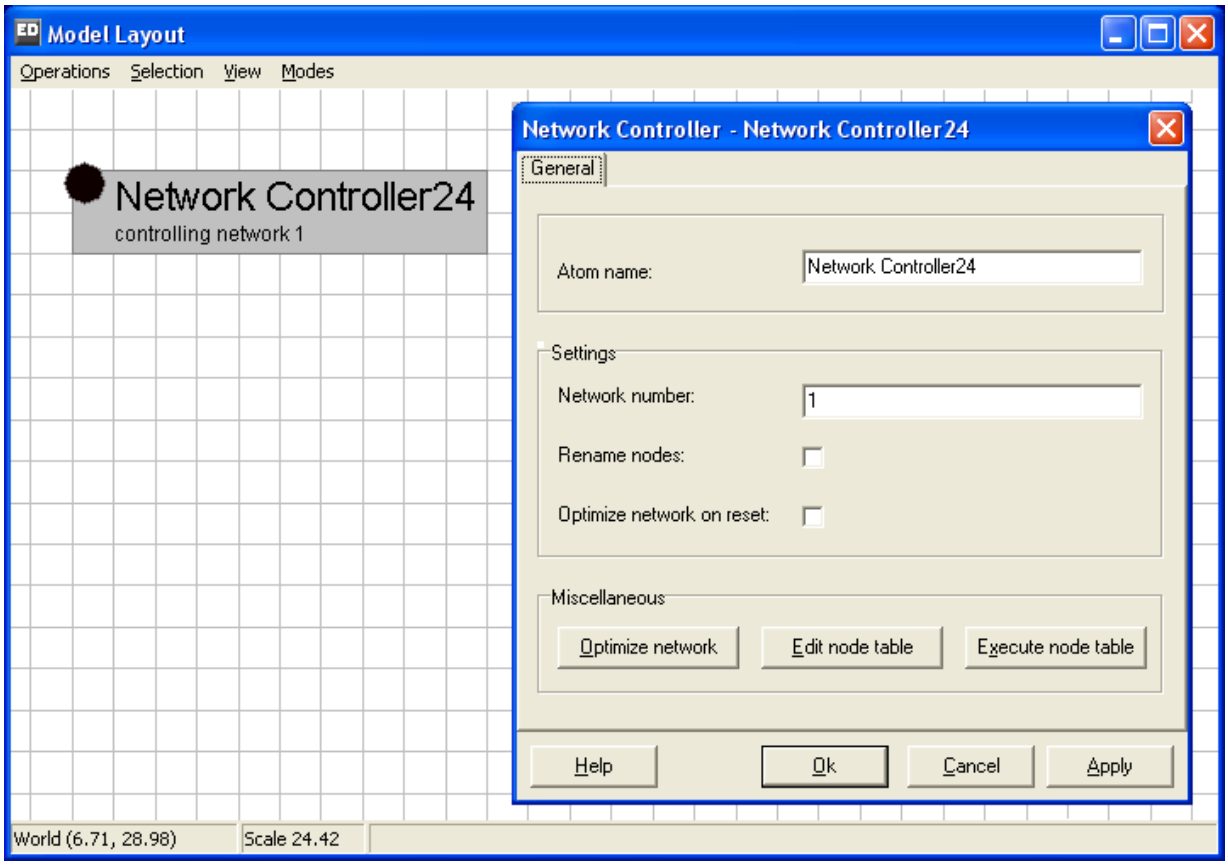
1. Drag a number of **Network Nodes** and a **Node Manipulator** into the Model Layout. Both can be found in the Transport category and are shown in Picture 9-3.
2. Connect two selected points by
   i. placing the Node Manipulator over the first point
   ii. right-clicking (the Node Manipulator turns completely green)
   iii. placing the Node Manipulator over the second point
   iv. right-clicking again

   A green link will appear between the 2 points, which, depending on the user's selection on the Node Manipulator, can be used one-directional or two-directional.
   Repeat this process for all connections you wish to create.
3. Using the same technique, connect each of the atoms where the Operator is needed with a point in the network (placed nearby).
   Attention: right-clicking with the Node Manipulator on the upper left corner of such an atom will show that connection as a blue line.
4. Connect the Operator with the network by right-clicking, with the cursor on the Operator, and selecting Connect to network (see Picture 9-2).
5. Drag a **Network Controller** into the Model Layout, (right-)click, and push button Optimize Network.

The names printed in bold refer to new atoms and can all be found under category Transport. The Network Node hardly requires explaining. Picture 9-3 shows all options of the Node Manipulator. To delete connections, repeat step 2; this time using the Disconnect option.

If you use Connect 2 way or Connect 1 way, the Node Manipulator turns green, showing 2 or 1 respectively as legend. With Disconnect 2 way or Disconnect 1 way, the Node Manipulator turns red, showing 2 or 1 respectively as legend.

Picture 9-4 contains all options of the Network Controller. Perform exercise 5 in order to familiarize yourself with these atoms, as well as with the steps mentioned above, and read the Help file for these atoms!



Picture 9-4: Options of the Network Controller

Questions and assignments:

5. Experiment with a new model consisting of a Source, a Server, a network that connects them, and a Conveyor followed by a Sink. Try to add a few one-directional routes, and delete these connections again, etc.. Make sure that the shortest route there differs from the way back. Connect the Server and the Conveyor by means of a network. Are blue connection lines showing? Add a Team with an Operator. The Operator assists the Server and the Conveyor. Connect the Operator with the network, as described above. Finally, optimize the network with the Network Controller.

   Does the Operator move nicely up and down between the various routes?

Important!
Optimize Network with the Network Controller allows users of the network to always take the shortest path (the algorithm of Dijkstra is applied to the internal network). *This is why it is necessary to perform this step for every change made in the network.*

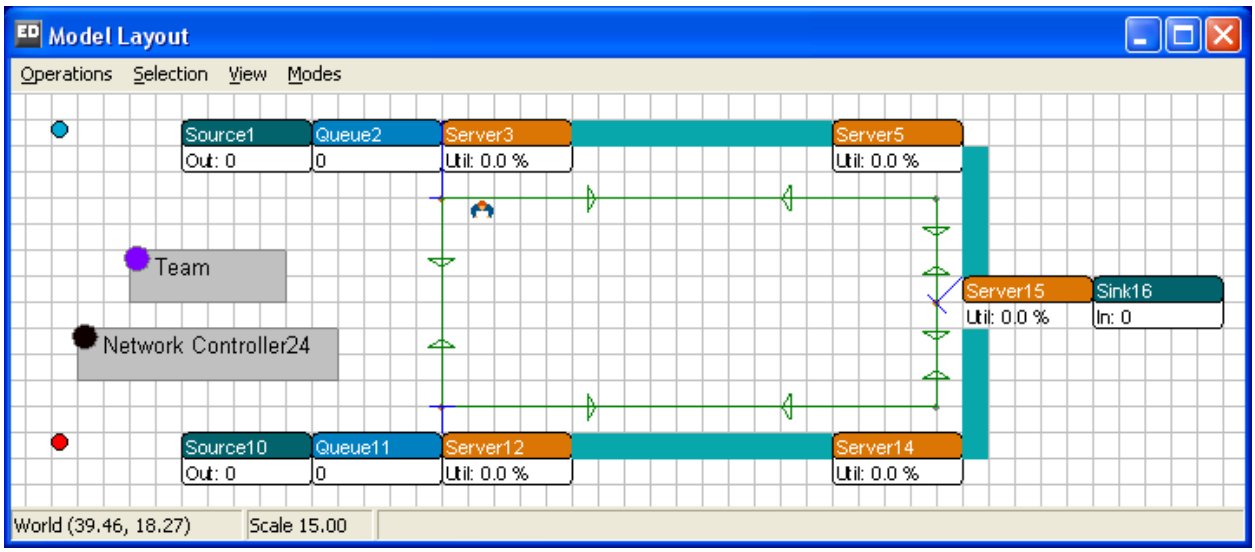
9.5 Walking by means of a network

Example 5 (continued): The Operator for Red and Blue

Add a network with 4 points and 5 connections to the earlier models, as can be seen in Picture 9-5. The blue lines represent the connections between the network and the atoms where the Operator is needed. The speed of the Operator remains 1 m/s. Perform all steps described in Chapter 9.4 and applied in exercise 5 in a logical manner.

Questions and assignments

6. Verify that the Operator walks by means of the network and picks the shortest route from one location to the next. Note that the Operator positions himself at the junction leading to the atom, where the Operator is called.
7. Verify that walking from the inspection point to a drilling machine takes 23 seconds and that walking between the drilling machines takes 8 seconds. This means that the time needed to walk along the blue connections is not counted!



Picture 9-5: The Operator in the Network

This model can be found under operator3.mod.

9.6 Allocation of priorities

Those of you, who simulate one of these three models long enough, will observe that the conveyors occasionally do fill up, creating a situation where the Operator wants to deliver a product to a conveyor, but can't, because the conveyor is full. This is what we refer to as a *deadlock*: the system hangs because an action is no longer possible.

This situation can occur here, since the Operator, by default, processes all requested orders according to the FIFO principle. Let's say that there are queues in front of the drilling machines as well as the inspection point and that the Operator is finished at time 100 with assisting in the inspection. Drilling machines 1 and 2 contain a product and have already been 'asking' for an Operator for a while: this is evident in visual terms, as the atoms turn red in color. Let's assume further that the order reached drilling machine 1 at time 90 and drilling machine 2 at time 95.

What now happens in what sequence at time 100? First, the inspection order is completed (uncoupling of the Operator), then a new product is brought to inspection, subsequently the Operator checks his list containing three potential orders and, according to the FIFO principle, selects the oldest [longest waiting]. Make sure that this list does not contain *two*, but *three* potential orders. That this is crucial will become clear a little later in the priority rules!

So, if the Operator makes a new choice at time 100, his list will show:

Call	From	Time of call
1	Drilling machine 1	90
2	Drilling machine 2	95
3	Inspection	100

Picture 9-6: Call order sequence for the Operator without priorities

Based on the FIFO principle the Operator now goes to drilling machine 1, then to drilling machine 2 and then back to inspection. So it is clear that in very busy periods the Operator continues to walk around in circles and that the system can deadlock!

We now want to assign priority to the inspection task. The syntax of the call-statement in Chapter 9-1 offers help. On inspection, change the Trigger on Entry:

CallOperators(AtomByName([Team], Model), 1)
to:
CallOperators(AtomByName([Team], Model), 1, **2**)

The calls from Inspection now have priority 2, while the calls from the drilling machines retain (default) priority 1. A '1' can be added to this last statement, but it is not necessary. If the calls in Picture 9-6 are now sorted by priority first, and only then by time of call, an entirely different picture emerges:

Call	From	Time of call	Priority
1	Inspection	100	2
2	Drilling machine 1	90	1
3	Drilling machine 2	95	1

Picture 9-7: Call sequence for the Operator with priorities

This, in fact, means that the row of inspection orders will be processed first. Why?

Those, who want to see this phenomenon in action, best take one of the three operator models without priority, watch out for when the model is in danger of deadlocking, stop the run and adjust the priority of the inspection, and then continue the run.

The models with priority are included: operator1 with priority.mod, and so on.

9.7 More Operator options

Much more is possible with Operators. Imagine a task that requires two Operators, e.g. lifting a heavy plate together. An Operator can also remain with a product longer. These possibilities are incorporated in an assignment.

Questions and assignments

Add a second Operator to operator1.mod and realize the following:

8. Inspection requires 2 Operators at the same time.
9. Drilling will always require an (arbitrary) Operator, but he will now accompany the product as far as the CNC machine.
10. Same as previous; except that Operator Red now handles the red products and Operator Blue handles the blue products.

These three modifications together can be found in operator4.mod.

10 THE TRANSPORTER

In practice, the transport of products over distances occurs with forklifts, AGV's, Reach Trucks or other vehicles. In Enterprise Dynamics, these vehicles are simulated with the Transporter or the Advanced Transporter.

The Advanced Transporter can either move freely on the pane or be restricted to paths in a network. This instantly reminds us of the Operator described in the previous chapter. In addition, indeed, for the free-range movement of the Advanced Transporter on the pane or movement restricted to paths, we are using the same concepts as for the Operator. We therefore advise you to work through Chapter 9 first!

However, there also are clear differences between the Advanced Transporter and the Operator. The Operator only moves himself from point to point, but Advanced Transporters transport one or more products, have load and unload points and sometimes cannot catch up with or overtake each other. In short, an Advanced Transporter has possibilities that are more complex. This has resulted in *two* Transporters being included in the Transport category, namely the Transporter and the Advanced Transporter, but only the latter can drive over the paths of a network.

In Chapter 10-1 we list the possibilities of both Transporters, followed by an example entailing the Transporter in Chapter 10-2. In Chapter 10-3 we will first repeat this example for the Advanced Transporter, followed by expanding the subject by adding a network. Chapter 10-4 describes in more detail the ideas behind and the connection with the Dispatcher, the Advanced Transporter and the Destinator. In Chapter 10-5, we discuss load and unload strategies with regard to the Advanced Transporter, while Chapter 10-6 describes the simultaneous use of several Advanced Transporters.

10.1 Functionalities of the Transporter(s)

Transporters can be used for the following:

1. Moving *one* product from pick-up locations to destinations.
2. *Automatically* determining driving times between pick-up and destination coordinates based on distance and indicated speed (free-range movement without a network).
3. Adding acceleration and deceleration for gathering speed and breaking, while retaining the possibility to automatically determine the correct driving times.
4. Adding unload and load times when picking up and delivering products.

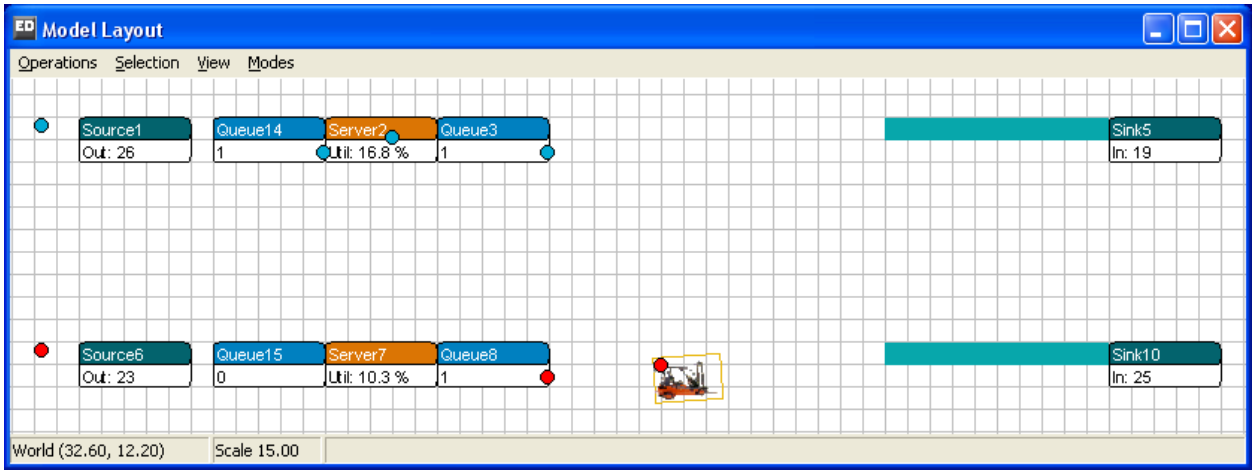
The Advanced Transporter has all the possibilities of the Transporter plus:

5. The capacity to move *several* products simultaneously from pick-up to destination points;
6. The capacity to move by means of the paths of a network, having set the maximum number of vehicles on a connection;
7. The capacity to lift products, that is to say to place goods at a certain height onto racks as is customary in warehouses;
8. Allows setting of (x,y,z) coordinates where products can be picked up from or stacked.

10.2 Working with the Transporter

Example 6: Pick-up and delivery

Identical to the situation in example five, here we have two parallel production lines where blue or red products, after completion of the machine process, are waiting on the factory floor for a forklift. The forklift transports the product to a conveyor some distance away, at which point the products leave the system. The straight-line distance from the machine to the conveyor behind it is about 15 meters. The distance between both machines is roughly 10 meters. Blue and red products are arriving at the machines with an average inter-arrival time of 2 minutes. The machines have an average production time of 20 seconds. Arrival and production times follow an exponential distribution. All buffers have a standard capacity of 10, the conveyor a standard speed of 1 m/s. The layout of this system can be seen in Picture 10-1.



Picture 10-1: Layout of the production line

The forklift has a speed of 1 m/s and free-range movement on the pane. To begin with, let's assume that the products are sent to one of the two conveyors with an equal chance.

Questions and assignments

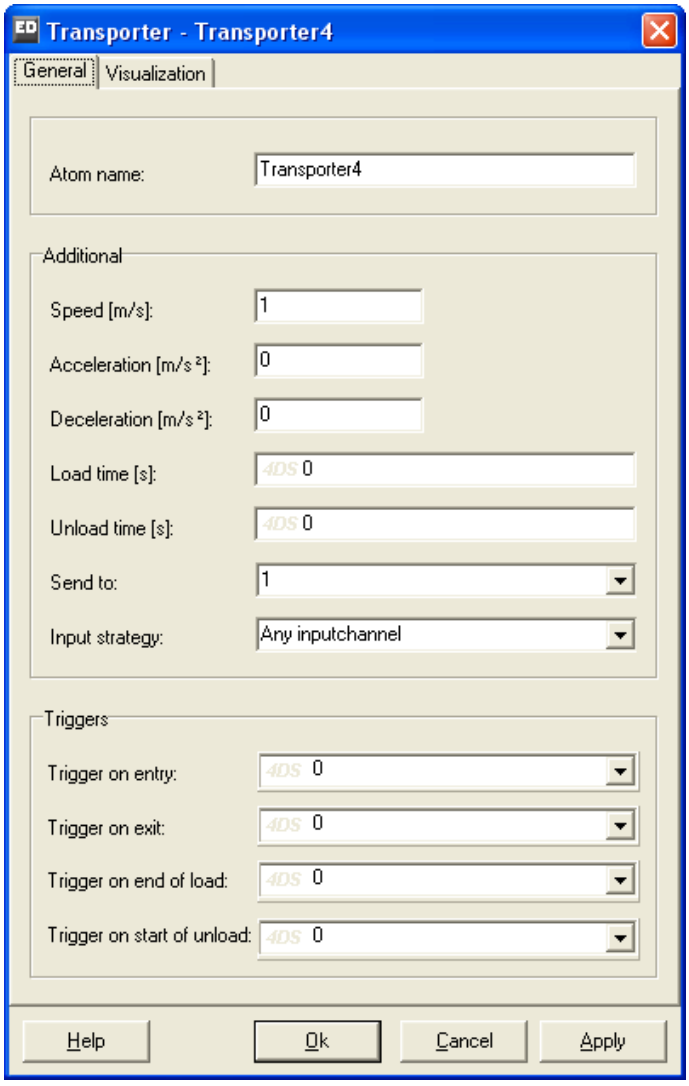
1. Approximately how high should the forklift's theoretical utilization level be?
2. How high will this utilization level become, when picking up and placing a product takes 5 seconds each?
3. Answer both questions given a situation in which the products are sent to the nearest conveyor only: blue products go to the upper conveyor, red products to the lower conveyor.

Hopefully, it will not be too difficult to answer these questions! Let us now review our answers by building this model with a Transporter.

Drag the Transporter (not the Advanced Transporter!) from the Transport category into the model. Connect the Transporter's input channel 1 or 2 respectively with the output channel of the buffer behind the machine of the first or second production line. Connect the Transporter's

output channel 1 or 2 respectively with the input channel of the conveyor of the first or second production line. Double-clicking on the Transporter opens the input window: see Picture 10-2. The default setting of the speed is 1 m/s. Change the Send To statement such that products are selected from both conveyors at random. As Input Strategy, select Largest Queue (well visible on the screen) or 'the Longest Waiting'.

Input Strategy and Send To statements are discussed in Chapter 5!



Picture 10-2: Input window of the Transporter

Make a few simulation runs with your own model now, or with transporter1.mod. Define 5 8-hour runs in the Experiment Wizard with output variable Status on the Transporter. Do the answers regarding the utilization level match your own calculations?

Then add 5 seconds for loading and unloading on Load Time and Unload Time. Run the experiment again. The utilization level of the forklift will increase from 55% to 71%.

In order to answer question 3, the model will have to be adapted. This can be done very neatly with a label; let's call it `Destination'. This label is given value 1 for the blue products and value

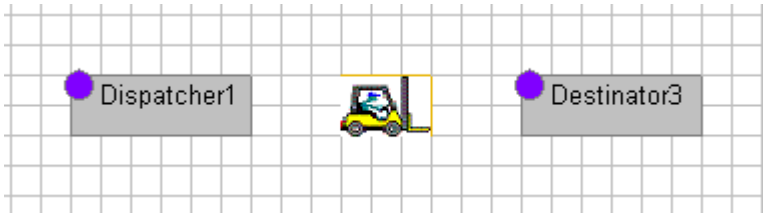
2 for the red products, and then used on the Transporter's Send to Statement to identify the correct conveyor.

Make a few more runs with your own model or with transporter2.mod. Compared to the previous model, the utilization level will have dropped by a few percent. This applies both to the variant with unload time as well as to the variant without unload time. Explain why!

10.3 The (Advanced) Transporter in the network

In Chapter 10-1 we claimed that the Advanced Transporter could do everything the Transporter can do. We will therefore, before expanding example 6, first replace the Transporter in model 1 with the Advanced Transporter and compare the results. These results will naturally have to be the same!

The Advanced Transporter will *virtually always* require two additional atoms: the Dispatcher and the Destinator. Both can be found in the Transport category. The Dispatcher handles the pick-up of the products (to dispatch = send) by means of the Advanced Transporter, the Destinator deals with getting them to their destinations. Picture 10-3 depicts the three atoms in the Model Layout. A different icon for each makes it easy to distinguish between the Advanced Transporter and the Transporter!



Picture 10-3: The Dispatcher, the Advanced Transporter and the Destinator

The Dispatcher's input channels are linked to the output channels of the pick-up sites, and the output channels are linked with one or more Advanced Transporter(s). The logic of the pick-up process is written on the Dispatcher.

The Destinator's input channels are connected with the output channels of one or more Advanced Transporter(s), and the output channels are linked to the input channels of the destinations. Aside from adjusting channels, not much else takes place on the Destinator.

Therefore, the Advanced Transporter is located in the middle and contains, among other things, the parameters that relate to the subsequent routing, speed, and loading and unloading.

Now go ahead, remove the Transporter from transporter1.mod, and drag the three above-mentioned atoms into the model. Connect the channel in the proper manner. In order to compare the Advanced Transporter with the Transporter, you still have to select the smallest queue from the queue of red and blue products.

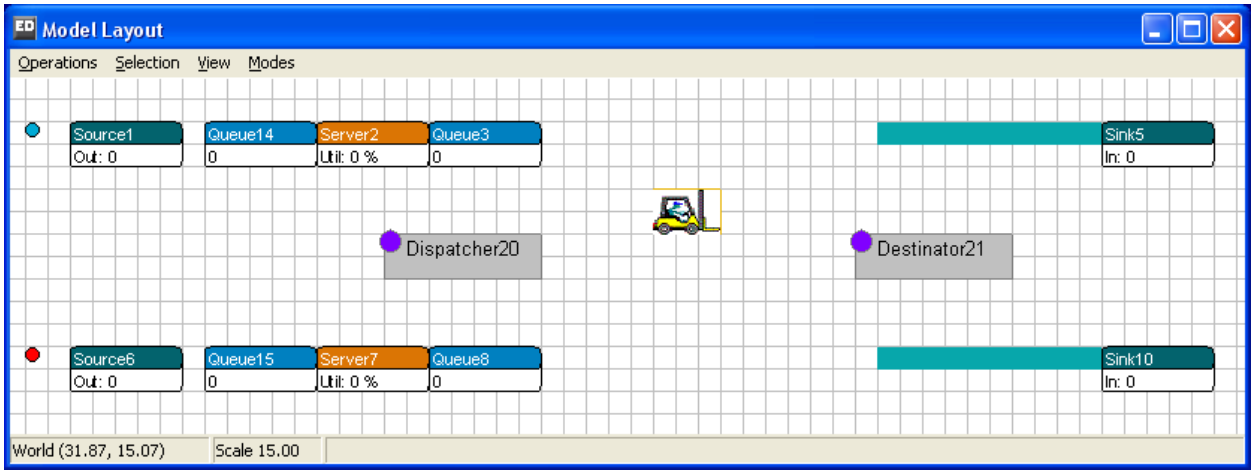
Therefore, on the Dispatcher, Sort Tasks By. Select:
3. Content minimum --> atoms in the container with the smallest content are dispatched first.

On the Advanced Transporter under General Parameters, Send To, select:

5. By percentage: 50% of the products go to channel 1, the remaining percentage goes to channel 2

Alternatively, select option 4, where an output channel is chosen at random.

This model can be found under transporter3.mod and is portrayed in Picture 10-4.



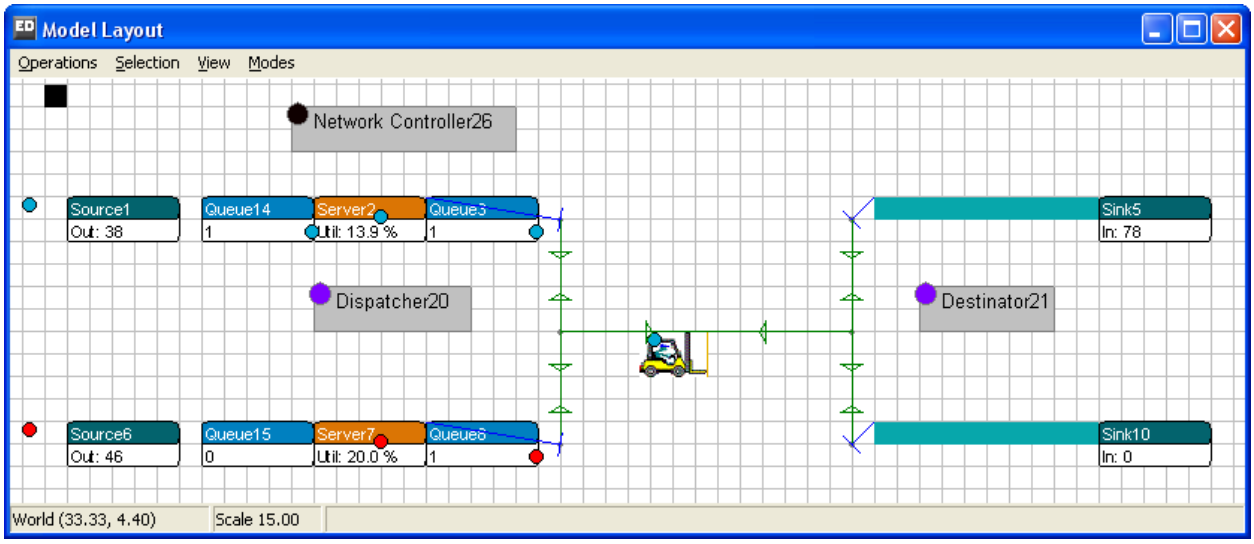
Picture 10-4: The Advanced Transporter without network

When we run this model, we find, by using the Experiment Wizard, a utilization level of 55% on the Advanced Transporter too.

The following elements pertain to the Advanced Transporter only. Since this is clear from the context, we will drop the prefix `advanced'.

Example 6 (first continuation)

The production lines and the running conveyors appear to be located in two different halls of the factory. A path connects both halls and the movement in each of the halls proceeds by way of mapped paths as shown in Picture 10-5.



Picture 10-5: The Advanced Transporter in a network

Two points are connected with a pick-up site at the end of a production line, 2 points are connected with a destination site for a running conveyor, and two points form the end points of the path between the two halls. All connections can be used bi-directional.

Questions and assignments

4. Using transporter3.mod as basis, add the above network with 6 points.
5. How high is the utilization level of the Transporter then?
6. How high is the utilization level when adding 5 seconds for loading and unloading?

Important!
Always use the steps described in Chapter 9 for building a network. Start by connecting the Transporter to the network before optimizing the network!

First, check if the model works, as it should on the screen. Does the Transporter move along the connections? Are the transport times correct? When in doubt, use transporter4.mod!

If everything works out, the answer to question 2 will show a utilization level of 77%, given that one up-and-down cycle takes 46 seconds. On average, a product enters a buffer once every minute: 46/60 rounded is 77%. With loading and unloading taking 5 seconds, we add 10 seconds to the cycle time: the utilization level rises to 56/60 or 93%.

10.4 The connection between Dispatcher, Advanced Transporter and Destinator

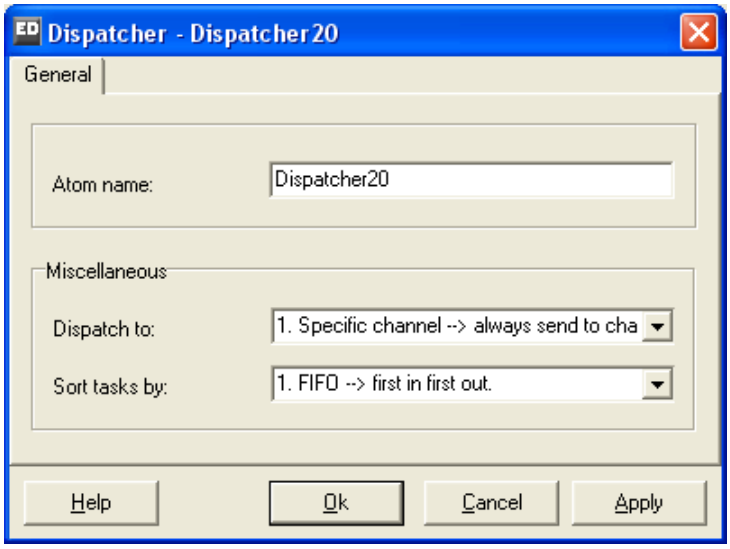
Example 6 has helped to demonstrate a number of the properties of both these Transporters. The 'ordinary' Transporter is more or less discussed in full. Acceleration and deceleration is all that has been skipped, but is easily built into the model – for both Transporters – and watched on the screen. What's more, we showed how the Advanced Transporter moves by means of a network. But … so much more is possible with this atom!

First, we shall describe the Dispatcher and Destinator in greater detail. Both can be used only in combination with the Advanced Transporter.

10.4.1 The Dispatcher

The Dispatcher generates transport tasks (no products!) for the atoms connected to the input channels of the Dispatcher. These tasks are allocated to the Advanced Transporters attached to the output channels of the Dispatcher: this means there can be more than one!

There are 9 Dispatch To rules, with as default: 1. Specific channel --> always send to channel 1.
This is the Transporter attached to the first output channel of the Dispatcher.

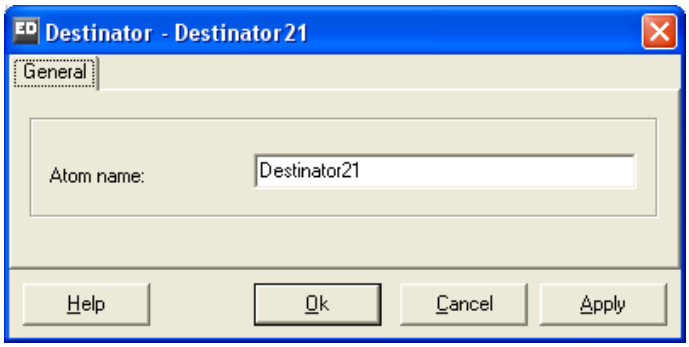


Picture 10-6: The input window for the Dispatcher

These transport tasks can be sorted (`Sort tasks by') according to 10 pre-defined sorting rules, with FIFO as default. Study all pre-defined allocation and sorting rules!

10.4.2 The Destinator

The input channels of the Destinator are linked to the output channels of one or more Advanced Transporters. The output channels of the Destinator are linked to all possible destination sites. Since all the logic of delivering is placed on the Advanced Transporter, not much else is to be seen in the Destinator window (see Picture 10-7).



Picture 10-7: The Destinator

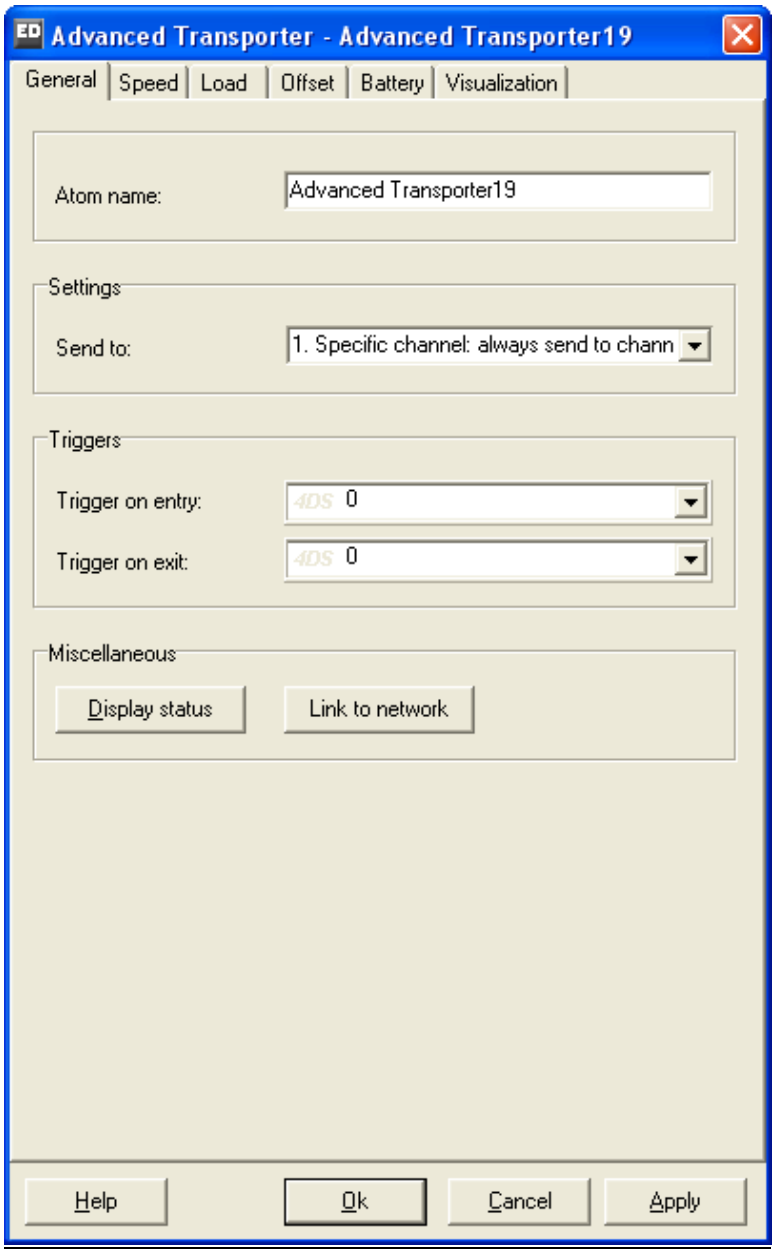
10.4.3 The Advanced Transporter

The input window of the Advanced Transporter has five different tab sheets in which parameters can be set:

- General parameters
- Speed parameters
- Load parameters
- Offset parameters
- Link to network

Here, we will discuss the General Parameters and the Load Parameters. The Speed Parameters are self-explanatory, as is the allocation of the Transporter to the network. The Offset Parameters will be dealt with in a later chapter, together with the Warehouse Atom.

The General Parameters (see Picture 10-8) show no new functionality. However, it is important to know that the Send To Statement is performed via the output channels of the Destinator.



Picture 10-8: Input window for General Parameters of the Advanced Transporter

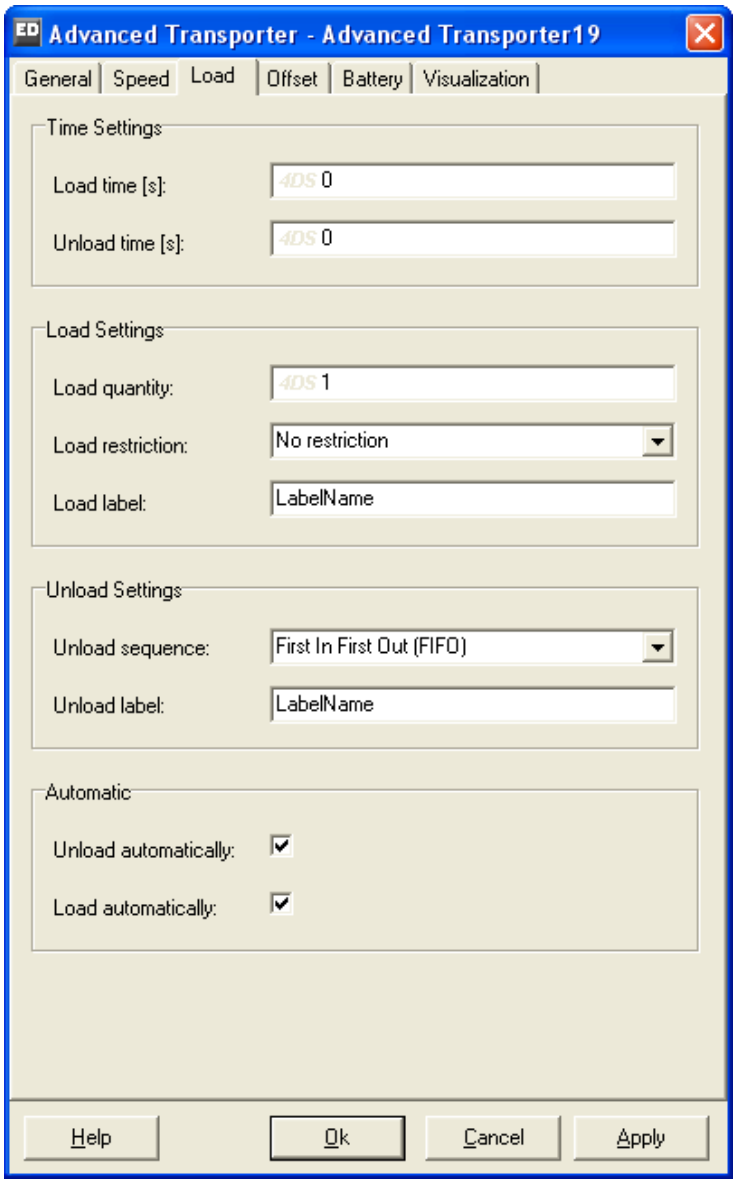
The Load Parameters (see Picture 10-9), however, do offer many new options. First of all, it is possible to adjust the load quantity. Only when that quantity has been picked up, the Transporter will start transporting products!

When no load restrictions are set, loading will take place based on the arrival sequence of the transport tasks, meaning that the Transporter may be running up and down between load sites. Six settings are possible:

- No restriction (default)
- Same name
- Same label
- Same label text
- Same mother
- Same container

Some of these options, we shall explain.

The load quantity with the corresponding restriction becomes active only after the first product is loaded. Let's say, for example, we have a load quantity of 5 and a Transporter makes a transport that first takes him to a product named `radio' in Queue10.



Picture 10-9: Load Parameters of the Advanced Transporter

Then the load restriction:

Same name	leads to	waiting and collecting another 4 radios from all pick-up sites;
Same container	leads to	waiting and picking up another 4 radios from Queue10.

In both of these situations, these radios must still be able to appear; so watch out for deadlocks!

10.5 Load and unload strategies of the Advanced Transporter

Let's build the following options into our example problem:

1. Collecting and transporting two products to the conveyors;
2. Collecting and transporting two *identical* products to the conveyors;
3. Collecting and transporting two *identical* products to a specific conveyor: the blue products to the first conveyor, the red products to the second conveyor.

Option 1 is easy to realize: change the load quantity to 2. Verify that the Transporter is now transporting both red-blue as well as any other product combination, depending on their arrival sequence in the buffers.
This means that when a blue product follows a red product, the Transporter will move to red only after the red product has arrived in the buffer. Subsequently the Transporter will move to the blue product, possibly after waiting. Then it's off to the conveyors. Note that the last product, which was picked up, appears in the icon of the Advanced Transporter.

Unloading too has many variants: red and blue placed on the same conveyor and on different conveyors. This was the idea of option 1 as well, but why does it take place the way it does? The answer lies in the Send to Statement: a product will go to the first conveyor with a 50% chance or will go to the second conveyor with the same probability. Evidently, some kind of draw goes on for each product inside the buffer, establishing the 'preferred' conveyor. Option 1 can be seen in transporter5a.mod.

For option 2, all that we do is change the Load Restriction to Same Container from option 1. This process can be seen in transporter5b.mod. But delivery still occurs in an unstructured way.

For Option 3, we revert to the Destination label as we have done before in Chapter 10-2: blue products receive a Destination label with value 1, red products receive a Destination label with value 2. Then, from Option 2, adjust the Send to Statement on the Transporter: 7. By label value (direct): Use the value stored in the Label named Destination and send to the corresponding channel.

This solution can be found in transporter5c.mod.

Come up with alternative questions yourself and try to solve them…

10.6 The use of several Transporters

Several (advanced) Transporters with different properties on the same network: how does that work? And what if they are not allowed to overtake? Let's examine these problems again by means of our example.

Example 6 (second continuation)

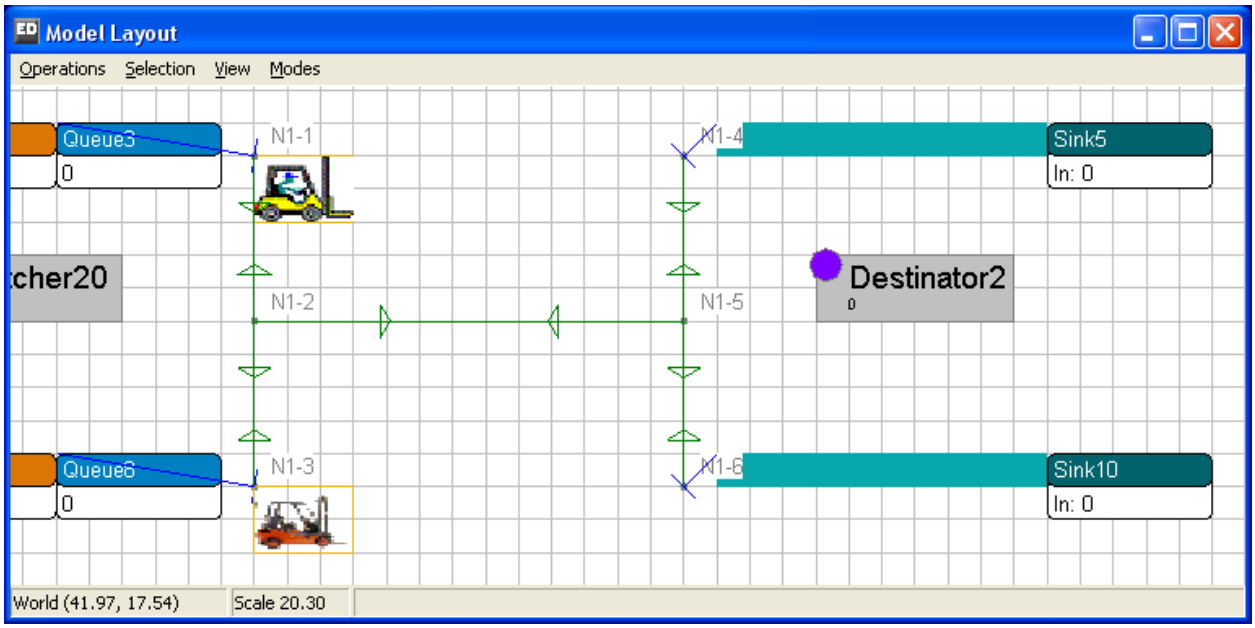
Due to the high customer demand the production flow of both product lines has doubled. Our one Transporter cannot handle this flow and a second Transporter has been acquired. Both Transporters are still transporting one product at a time to an arbitrarily selected conveyor; however, the second Transporter is significantly faster and has a speed of 2 m/s. When both Transporters are free to pick up a product, the product is collected by the faster Transporter.

Questions and assignments

7. Using transporter4.mod as basis, add a second Transporter to the network with the required properties and adjust the product supply. Don't forget to optimize the network!
8. Study the behavior of both Transporters in your own model or, when in doubt, open transporter6.mod. Can Transporter 1 overtake Transporter 2 without problems?
9. In order to check things, answer the questions concerning the utilization levels of both Transporters again.

In transporter6.mod, we have added a second Advanced Transporter to the network. To keep them apart, the icon of the second Transporter has been changed into the icon of the 'normal' Transporter. That of course does not change the functionality of our new Advanced Transporter!

Picture 10-10 shows a section of the layout with both Transporters.



Picture 10-10: Two Transporters in one network

With the Node Manipulator and the Show Nodes option, the names of the nodes are made visible. As home base for the yellow or red Transporter, node N1-1 or N1-3 respectively was selected. These adjustments can be seen when the model is reset.

The allocation of a product to the fast Transporter, when both Transporters are free, is done on the Dispatcher with Dispatch to:
3. A free Transporter (LTF) --> look for a free Transporter. Check the Last Transporter connected to the dispatcher First.

This is done because the fast Transporter is connected to the second output channel of the Dispatcher. Study the other Dispatch To rules in order to get an idea of the possibilities.

Question 9 is answered with the Experiment Wizard. With priority given to Transporter 2, we observed a utilization level of 48% (Status Idle 0.52) for the Transporters as a group, while the utilization level for Transporter 1 with priority rises to 56% (Status Idle 0.44).

With an equal distribution of the load, this works out in theory at 57.5%: 2 orders per minute, one of which is carried out by Transporter 1 (46 seconds) and one by Transporter 2 (23 seconds). This comes to an average of 34.5 seconds work out of 60 seconds.

Find an explanation why these 57.5% are still higher than the measured 56% with priority for the slow Transporter!

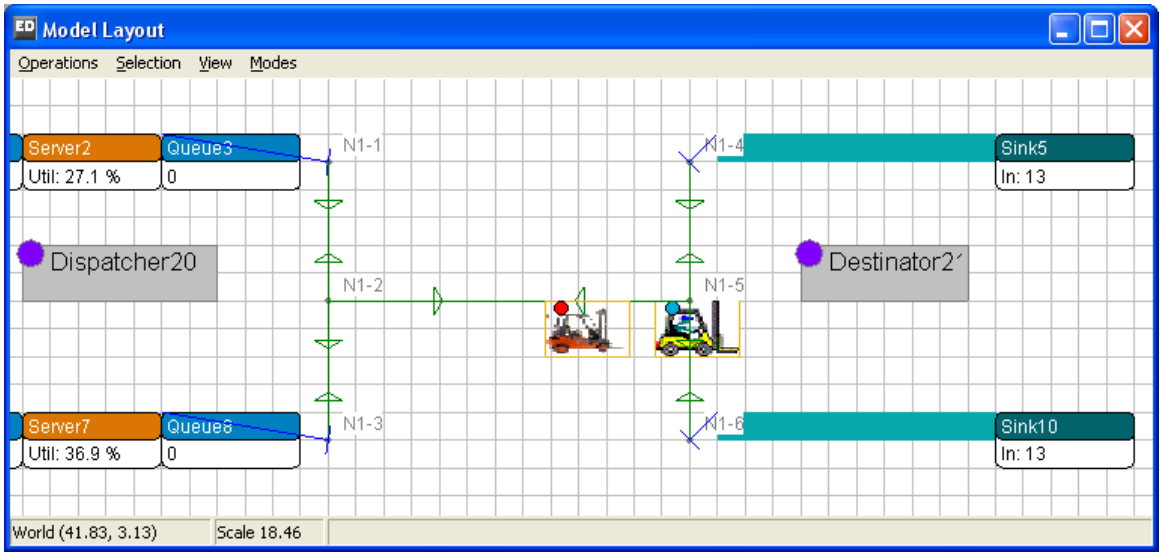
Example 6 (third continuation)

Due to security considerations, only one Transporter may use the center aisle path in both directions. A second Transporter has to wait until the respective lane is free again. Overtaking on this stretch therefore is impossible,

Questions and assignments

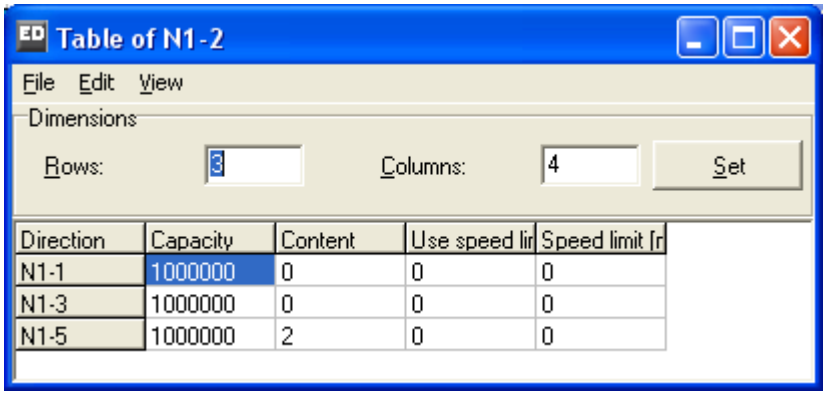
10. Realize the above situation with the two Transporters in the network.

In order to answer question 11, we examine how we can realize capacity restrictions on sections of a network: In Picture 10-11, we see 2 Transporters moving on the connection N1-2 to N1-5, a situation that now must be ruled out.



Picture 10-11: Two Transporters on the connection N1-2 to N1-5

Furthermore, point N1-2 has been selected. This is evident from the black border to the lower right of N1-2. Double-clicking on that point opens the table of Picture 10-12
.



Picture 10-12: The network table of N1-2

This table contains the three connections from N1-2 to N1-1, N1-3 and N1-5. *Capacity* shows the permitted number of vehicles allowed to move simultaneously on the indicated connection. The default setting is 1000000, or better still: infinite. *Content* indicates how many vehicles are moving on the indicated connection at that moment. The content of 2 vehicles from N1-2 to N1-5 is part of the snapshot in Picture 10-11.

Let the model run for a while to see how the numbers in the right-hand column of Picture 10-12 continuously keep changing. Examine this behavior at the other points of the network also.

By changing the capacity from N1-2 to N1-5 to 1, answering question 11 has now become easy. Do the same from N1-5 to N1-2!

Now study how the Transporters behave on the stretch from N1-2 to N1-5. Are the vehicles waiting for each other? Note that the connections turn red when the capacity limit is reached!

This model can be found under transporter7.mod.

For now, we stop explaining the application of Transporters with or without network for the time being.

It is clear that we are now able to model a number of complex problems in warehouse environments.