

1. Getting Started

The purpose of this document is provide a basic level introduction to the most important features of the SDT program. There may be further releases of this document. Start SDT as instructed on the web page of the SDT assignment. The SDT Organizer window appears.

2. Organizer

The SDT Organizer is the tool that assists you when working with SDL and MSC (Message Sequence Chart) diagrams. Organizer can also handle text files, object models and c-files. Organizer is also managing the other tools in SDT.

Some important commands within organizer are found in the next table:

Menu/Command	Action
File/New..	Creates a new system
File/Open..	Opens an old system
File/Save..	Saves the current settings in the system file
File/Print..	Prints the whole structure or parts of the structure
Edit/Add New..	Adds a new diagram/module/area to the contents of the organizer
Edit/Add Existing..	Adds a file to the contents of the organizer
Generate/Analyze..	Starts analyzing the selected system
Generate/Make..	Starts the make process, generates code and an executable Simulator/Application...
Generate/SDL Overview..	Starts to generate an SDL Overview diagram
Tools/Search..	Starts the search
Tools/SDL/Simulator UI..	Starts the Simulator user interface

The Organizer has also quick buttons which are:



You can also use pop-up menu which is the menu that comes when you press the right mouse button. "Move up" and "Move down" buttons move the selected diagram in the Organizer up or down. Some quick buttons make more than one task. For example, when you click "Simulate" button, SDT analyses the selected SDL system, makes simulator application and opens the application to the Simulator UI. "Validate" does just like "Simulate", but opens the Validator UI instead of Simulator UI.

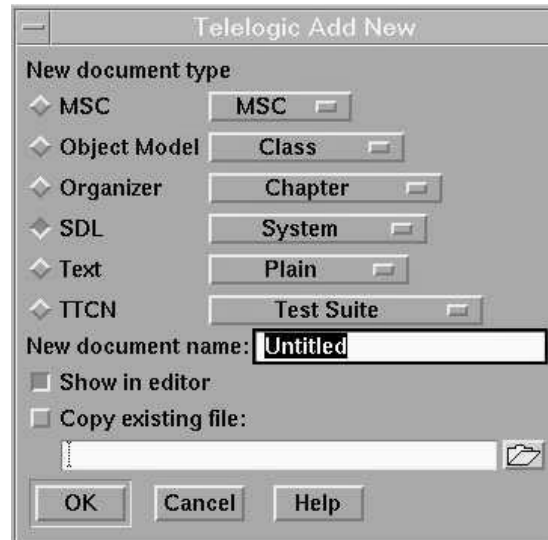
The drawing area of the Organizer is divided into several presentation areas containing icons of different statuses. You may freely add, remove, and rename a presentation area, as well as rearrange the order of the presentation areas. Default presentation areas are:

- the Analysis Model Area
- the Used Files Area
- the SDL System Structure Area
- the TTCN Test Specification Area
- the Other Documents Area.

3. SDL Editor

At first you can remove all those presentation areas that you don't need. Then you go to the presentation area what you want and start to build your own system structure.

First you select the command "Add New" when the following dialogue should appear:



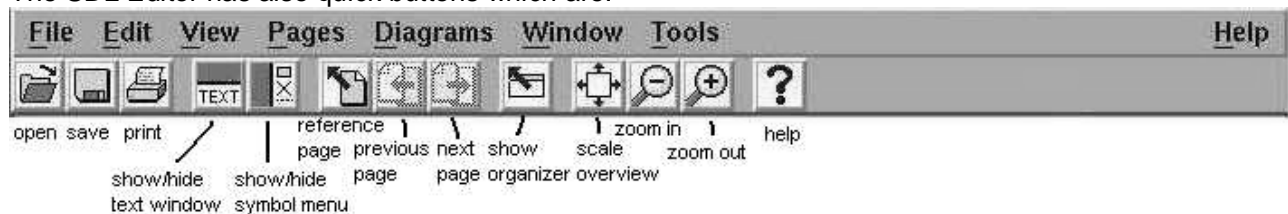
When you want to design an SDL System, you choose the option SDL and System (like in the picture) and then you write the system name for it. If you want to start doing it right away, you can choose "show in editor" and the SDL Editor should appear after you have pressed OK.

SDL Editor Overview:

The SDL Editor is the tool that lets you create, view and edit your SDL diagrams. The editor has different modes, depending on the kind of edited diagram. The modes differ with respect to on-line syntax and symbols in the symbol menu. A few commands are also affected by the mode. The most used commands in the SDL Editor are viewed in the next table:

Menu/Command	Action
File/Save	Save the current diagram
File/Print..	Print the current diagram
Edit/Cut,Copy,Paste	Graphical clipboard features
Edit/Redirect	Changes the direction of a channel or a signal route
Edit/Bidirect	Changes a channel or a signal route to be bi-directional
Edit/Drawing Size..	Set the size of the diagram drawing area
Pages/Edit..	Editing features for diagram pages: Add pages, Copy pages, Edit pages....
Diagrams/...	Switch between the different diagram buffers in the editor
Window/New Window	Open a new SDL Editor window
Window/Grammar Help	Open the Grammar Help window:

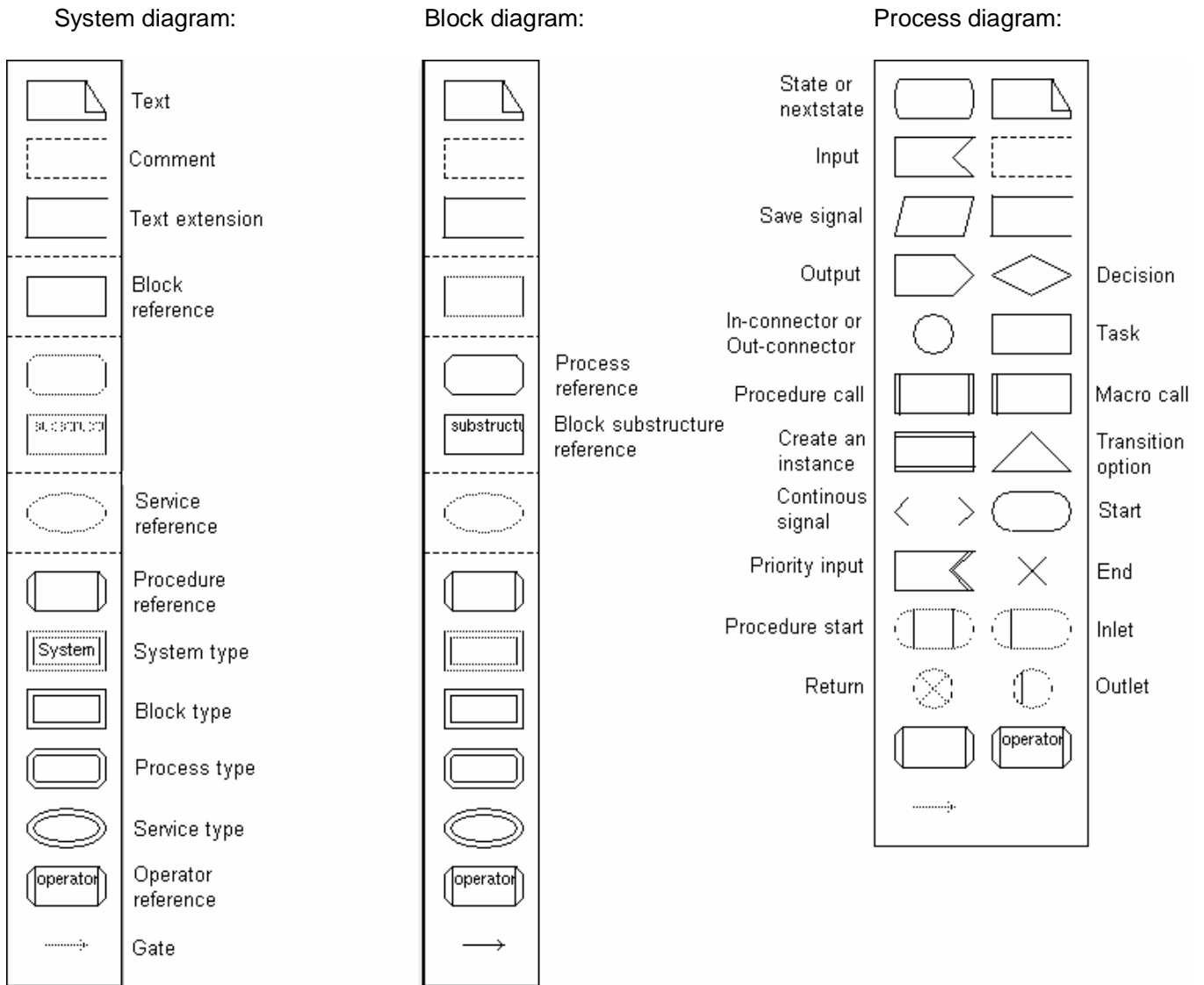
The SDL Editor has also quick buttons which are:



Commands for some of these quick buttons are:

Command	Action
Show/Hide Text Window	Toggle the text window
Show/Hide Symbol Menu	Toggle the Symbol Menu
Reference Page	Edit the diagram where this diagram is referenced
Previous/Next Page	Show the previous/next page (if there is)
Show the Organizer	Moves back to the Organizer
Scale Overview	Set the scale to fit the diagram into the size of the window

Symbol options in different diagrams:



Some of these symbols can only be used, for instance, when doing procedures or when there are Block/Process types involved in the system. Symbols that can't be used are shown blurred.

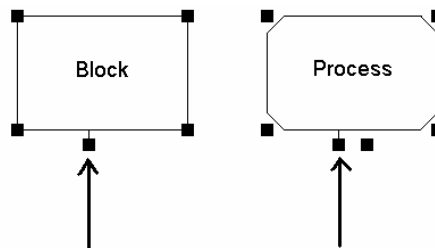
Designing diagrams:

When you want to place new symbols to the diagrams, you can either double-click the symbol in the symbol menu or you can grab the symbol from the menu and move it to the drawing area.

You can resize the symbol by grabbing one of the corners of the symbol and drag.

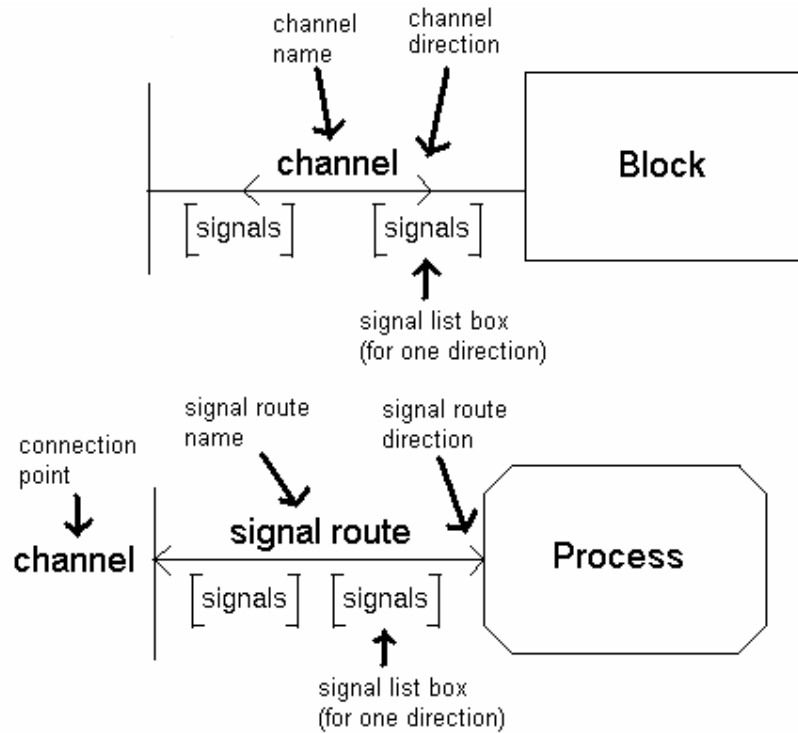
Channels can be placed, if you select a block symbol, grab the handle on the bottom of the symbol and drag.

Signal routes can be placed, if you select the left handle and drag. Handles are shown in the next figure.



When you have connected your channels (signal routes) to the place you want, you can change the direction by using command redirect from Edit menu. You can also change it to bi-directional by using bidirect command.

Components of a channel and signal route are shown in the next figure.



When you connect channels to environment in the system level, you don't have any connection points. In the figure above we assume that the Block includes the Process. So, when you want to send signals from process to environment (or vice versa) you have to connect the signal route to the channel like in the picture.

Text editor:

You can write texts into symbols when the symbol is selected and your mouse pointer is pointing the symbol or you are pointing the text area below the drawing area. Here is a figure about the text area and also some editing commands that you can use.



When you have started to design your own system and your SDL Editor is in the system level, you might want to design some block(s) there. After you have connected these blocks to environment or to another blocks with channels you go to the next level to design the blocks. By double-clicking the block reference symbol in the drawing area the same dialogue than in the page 2 appears. Now you choose SDL and Block and press the button OK. Now you are in the block level and you can start to design processes (or blocks). When you have connected these processes to right channels or to another processes you can

start to design the functionality that happens inside the processes. By double-clicking the process in the block level you get again the same dialogue than in page 2. Choose the right options and press OK. If you didn't save the diagrams yet, there is "unconnected" text next to the System/Block/Process names in the Organizer window. After you have saved the diagrams, you will see the file name and "rw", which are your rights to that file, instead of "unconnected".

Signals that you are using in the System and Block level have to be introduced there in the same level or in the upper level where you want to use them. If you introduce a signal in the system level, you can use it inside the blocks in that system without introducing that signal inside that block. When you introduce a signal inside a block you can only use that signal inside that block and not in the system level. How to introduce signals are shown in the next figure.

```
signal
input_signal,
output_signal(Integer);
```

Here you have introduced two signals. One named input_signal with no parameters and other named output_signal which has one parameter and type for that parameter is Integer. When you want to use more parameters than one, you just introduce this signal like this:

```
output_signal(Integer,Integer,Charstring)
```

Note, that there is semicolon in the end of introduction of signals and comma between signals.

SDL has 9 predefined data sorts that you can use. These are:

- Integer
- Natural
- Real
- Boolean
- Character
- Charstring
- PId
- Duration
- Time

You don't have to know more about these. If you want to know more, you can search more information from SDT online help (Organizer window) which is also quite good source when you want to know any other information about SDL/SDT. You can also create new types using these predefined data types and so on, but you are spared for this kind of information so far.

Designing functionality:

Now you are in the process level and starting to design the functionality of the system. All variables you are using have to be introduced. Here is one example:

```
DCL
var1 Integer,
var2 Charstring,
var3 Duration;
```

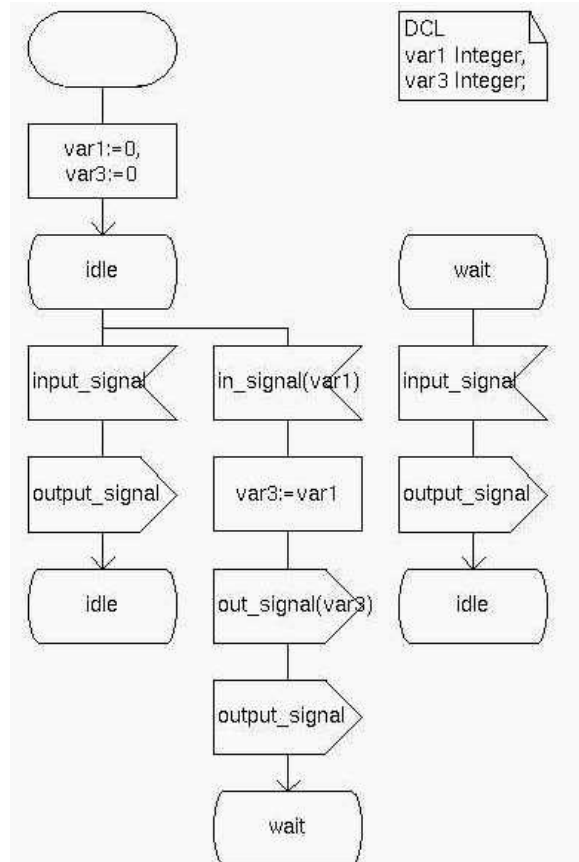
DCL comes from declaration. In the end of declarations there must be semicolon like in the case of signal introduction.

Timers have to be introduced in a bit different way:

```
Timer
Clock;
```

The first symbol that has to be in every process is Start symbol (page 3). After that there could be some initial tasks before the initial state. For instance, set initial values for variables. After initial state there is input signal(s) and tasks what should be done when that input arrives. Then (usually) there is output

signal(s) after tasks and next state that can be either same state or another state than the initial state and so on. Here is one simple example that shows, how does this work:

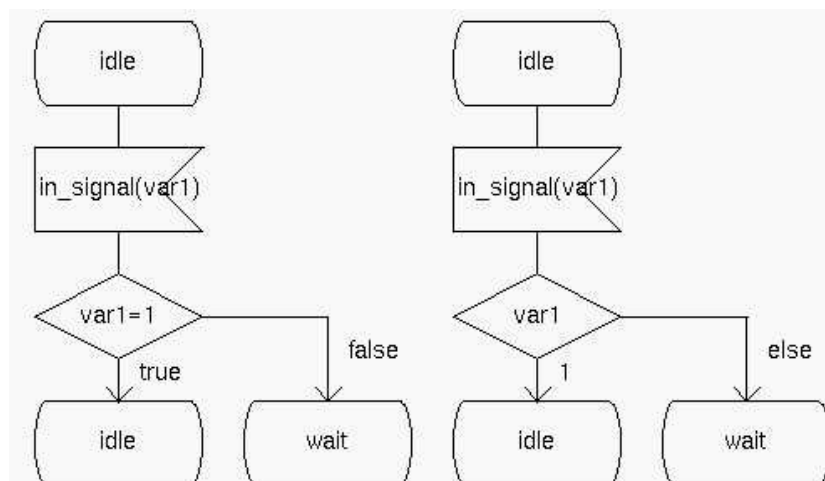


In this example var1 (Integer) and var3 (Integer) are initialised so, that they both have value 0. Initial state here is idle. When the process is in the state idle, signals that could come are input_signal and in_signal that has one parameter (Integer). If the next signal is input_signal, output_signal is sent out from the process and next state is the same than the initial state. If the next signal is in_signal with one Integer parameter, variable var3 gets this parameter value and signals out_signal with var3 and output_signal is sent out from the process. In this case next state will be wait. When process is in state wait, the only signal that could affect the process is input_signal. So, process is in state wait until it receives signal input_signal and transmits output_signal and goes to the state idle again.

As you can see, this works just like a state machine.

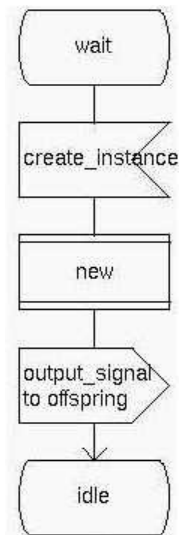
Assignment var3 gets Integer value var1, which is expressed in SDL with marks := (like in the picture).

Another useful symbol is Decision symbol. Here is a little example of that:



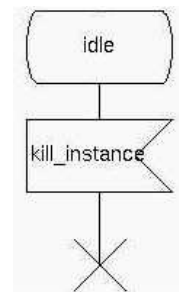
As you can see there is more than one way to do different decisions.

There is also possibility to create processes dynamically. Here is an example of that:



In this case, when create_instance signal arrives, process called new will be created. Also in this example signal output_signal will be sent to that created process. As you can see from the picture, you can use preposition "to" when sending signals to exact process. Sometimes it's the only way to send signals to right processes. You can also use instead of preposition "to" word "via", but in this case you have to name the channel or gate where to send the signal. This is also more assured way to send signals than addressing signals at all. Offspring in the figure means the address of the process new and type of this is PId (process id), which is one of the predefined data types.

You can also end the processes so, that they won't "exist" anymore. Here is an example of that:



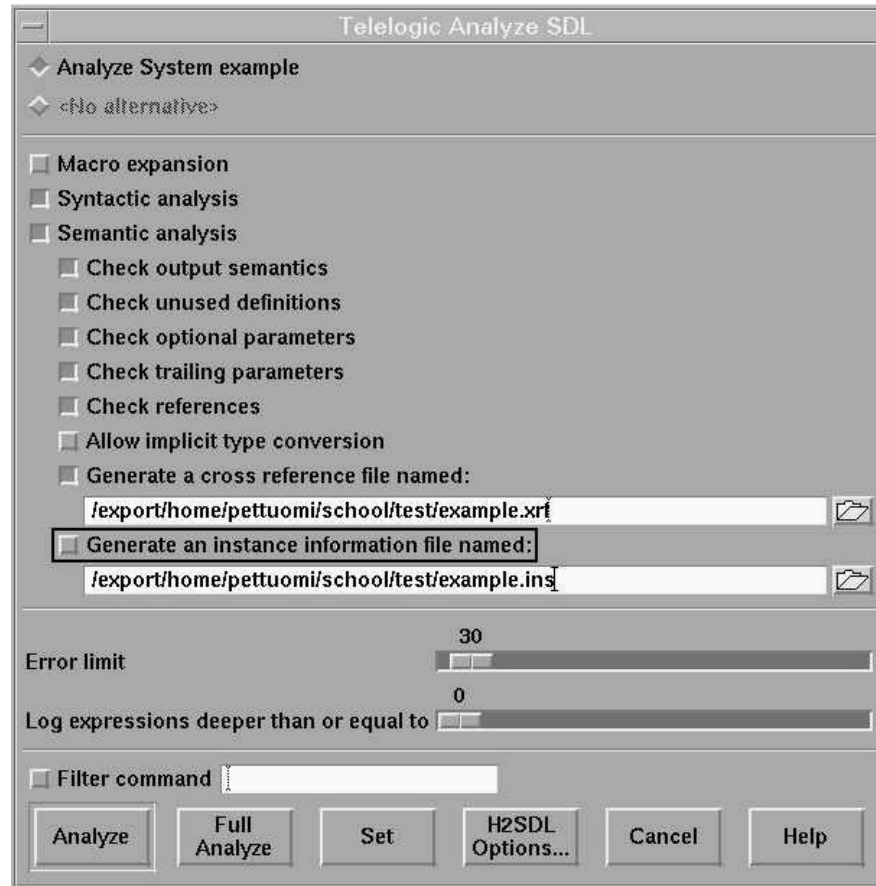
Process have to receive signal before ending the process. Should be quite easy to understand.

These were the most important features concerning your forthcoming exercise. If you want to know more about the other symbols or other features, the best way again is to search from the Organizer's online Help menu.

4. Analyzing the design

When you have finished your design, the next procedure is to find errors, if there are any, and generate an application.

- From Generate Menu, choose Analyze and the following dialogue appears:



These are the default options for the analyzer. Here you use the same options.

- Press button Analyze or Full Analyze.

You can follow the Analyzing process (or Make process) from the Organizer Log window.

5. Organizer Log Window:

The Organizer Log window works as a console for the Telelogic Tau tools. The window can be visible (raised or iconified) or not visible on the screen. All log information is output to this window independently of whether the window is visible or not.

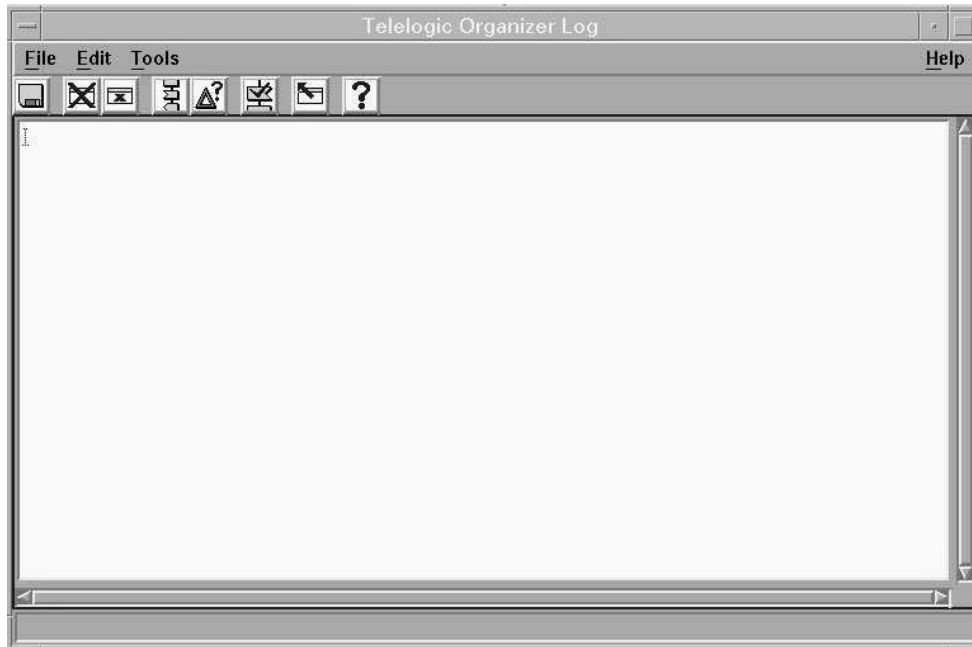
The window is used in the following situations:

- To log information from the Analyze and Make process.
- To log the activities when an SDL system is imported.
- When the Search List Manager issues the command User Command.
- When files are checked during the Open command. Inconsistent files are reported.
- To show status information from the Generate tools.
- Other tools, such as ITEX and tools started with dynamic menus, may also use the window to output textual information.

The window is opened or raised when the menu choice Organizer Log is selected from the Tools Menu, or when something is written to the log. The preference ShowLogLevel controls which output to the window should cause a raise of the window.

There is only one Organizer Log window. The Organizer main window is not locked for user input when the Organizer Log window is visible.

The Organizer Log window is a sub window to the Organizer's Main window.



Meaning of the quick buttons:



Show Error shows the error in the SDL diagram. Help on Error shows the description of that error referring to online Help. Clear cleans the Log window.

6. MSC Editor

You can create a new MSC diagram by choosing "Add New" from edit menu at the Organizer. Now similar dialogue as with SDL Editor should appear (see section 3. SDL Editor).

Choose option MSC using radio buttons and input a name for your MSC diagram.

If you don't want to start designing immediately unchoose option Show in editor, otherwise simply press OK. Another way to start MSC Editor, is to choose "Tools->Editors->MSC Editor" from Organizer. This will start MSC Editor with new diagram.

MSC Editor Overview:

The MSC Editor is a tool for creating, viewing and editing your MSC diagrams. The most used commands in the MSC Editor are:

Menu/Command	Action
File/Save,Open,...	Save, open or print your diagram
Edit/Cut,Copy,Paste	Cut, Copy or Paste from/to clipboard
Edit/Rerect	Changes the direction of message
Edit/Connect	Lets you to connect current diagram to existing one
Edit/Drawing size..	Lets you to adjust the size of the drawing area
Edit/Make space	Creates space for additional objets to be inserted to selected point
Diagrams/*	You can switch between diagrams open in the editor
Window/Info Window	Opens a info Window that shows information about selected obejt
Tools/Tidy Up...	Automatically makes a new layout for the diagram
Tools/Generate SDL	Generates a template SDL of the drawn MSC diagram

The quick button bar in MSC Editor is largely similar to the bar in SDL Editor. There are however few (three) new buttons (starting from the third on the left). Meaning of these new buttons are listed in the table below.



Command	Action
Add space for item	Adds space for new object to selected position
Remove space	Removes space from selected position
Show/hide instance ruler	Toggle the instance ruler

It is also possible to generate MSC diagrams from SDL diagrams (or vice versa). This is done by converting SDL or MSC diagrams to the PR Format (Textual Syntax for SDL) and back to the graphical format. This generation might be quite uncertain when doing with large models. You can find more information concerning this from online help.

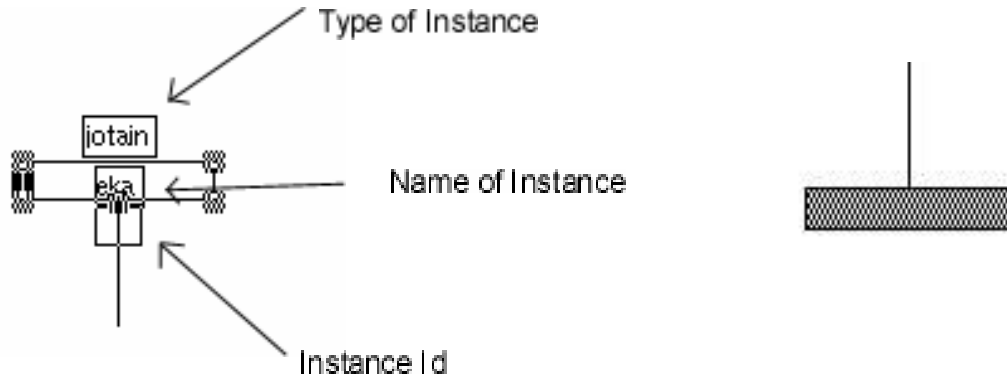
Symbols used to design an MSC diagram:

	Text
	Comment
	Instance Head
	Instance End
	Message
	Condition
	Timer
	Separate Timer
	Action
	Create Process
	Process Stop
	Coregion
	MSC Reference
	Inline Expression Begin
	Inline Expression Separator
	Inline Expression End

Designing diagrams:

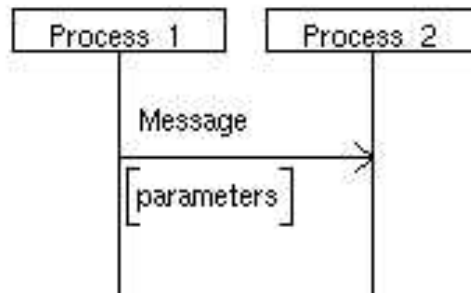
Similar to SDL Editor, new symbols can be placed to diagram by double-clicking the symbol or grabbing and moving it to desired position.

Components for Instance Head symbol are shown in the next figure together with instance end.



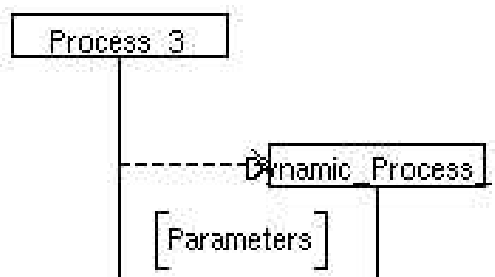
Instance head identifies the object and therefore is the first symbols for every object. When instance head is placed, an axis will appear. In this axis is placed all the other symbols mentioned in this section.

Information exchange and synchronization between entities is handled by messages. Messages are illustrated in the MSC diagram by arrows. The use of message between two processes to carry information is shown below.



The condition symbol is used to illustrate certain case when following actions are performed. The case when this condition is performed is named inside the hexagon.

Create process symbol is used to indicate that certain dynamic process is created by another process at a certain time at runtime. The procedure to create dynamic processes is shown in the next figure. Similar to messages, create commands can carry parameters to processes.



Action symbol is used to specify what kind of actions are performed at certain place/time. For example action symbol can contain straight forward insert expression $a := b$.

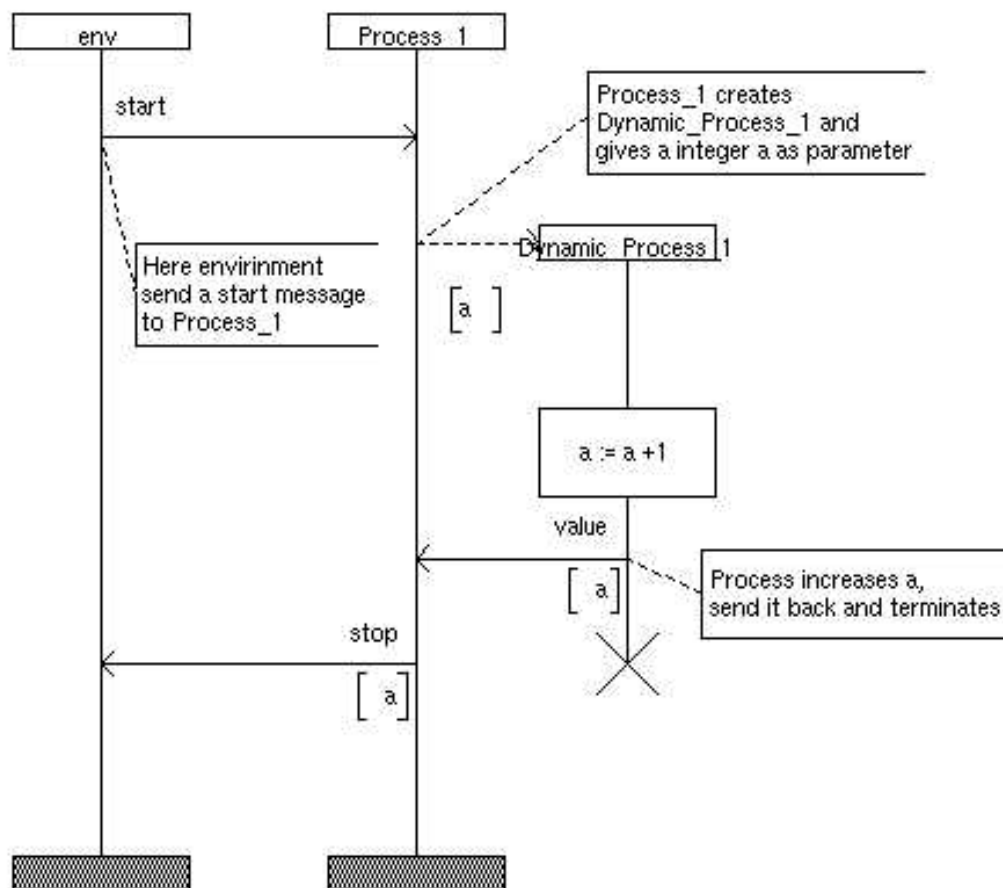
Text Editor window:

Use of text editor was explained earlier in this document.

Small example MSC:

A simple example MSC is shown in the next figure to make it easier to understand the syntax of MSC language. The described example system does the following:

- At start-up only one process (plus environment process) is running
- Environment sends a message to Process_1 at start-up
- After receiving start signal, Process_1 creates dynamically Dynamic_Process_1
- Dynamic_Process_1 gets a integer parameter from Process_1 at creation
- Dynamic_Process_1 increases this value and sends it back to Process_1 in a message and stops
- After receiving integer value, Process_1 sends it to environment



These are the most used symbols in MSC Editor. Information about other symbols can be found from the online help of Telelogic SDT.