Chapter 2

# Release Notes

**This chapter contains information related to the SDL Suite and TTCN Suite 4.5 releases, in terms of:**

- **Included components**
- **New and changed functionality**
- **Known limitations**
- **Recommendation of use**

# Included Components

The SDL Suite 4.5 and TTCN Suite 4.5 release features the following tools and documents:

## Common SDL Suite and TTCN Suite Tools and Features

- Organizer
- Print utility
- Preference Manager
- Postmaster Libraries
- ASN.1 Utilities
- DOORS Integration

## The SDL Suite Tools and Features

- SDL Editor
- CIF Converters
- ispell integration
- Emacs integration **(UNIX only)**
- MS Word Integration **(Windows only)**
- Link Manager
- Type Viewer
- SDL Coverage Viewer
- Index Viewer
- SDL Analyzer
- SDL Access (not included in the standard distribution)
- CPP2SDL Utility
- SDL to C Compilers (Cbasic/Cadvanced/Cmicro)
- Targeting Expert
- SDL C Compiler Driver (SCCD)
- Simulation Library and GUI
- Validation Library and GUI
- Application Library
- Master Library
- Cmicro Package
- Threaded Integration
- ASN.1 BER/PER Encoding/Decoding
- OM Editor
- SC Editor
- MSC Editor

- HMSC Editor
- Deployment Editor
- Text Editor
- UML2SDL Utility
- OM Access

## The TTCN Suite Tools and Features

- Browser
- Finder **(Windows only)**
- Table Editor
- TTCN Link
- Autolink
- Data Dictionary
- Analyzer
- TTCN to C compiler
- Simulation Library and GUI
- TTCN Access
- TTCN Exerciser
- ASN.1 BER/PER Encoding/Decoding
- TCP/IP Module for adaptation

## User Documentation

- Installation Guide – on-line help and PDF files on the installation CD
- Release Guide – on-line help and PDF files on the installation CD
- Getting Started (tutorials) – on-line help and PDF files on the installation CD
- Methodology Guidelines – on-line help and PDF files on the installation CD
- User’s Manual – on-line help and PDF files on the installation CD

# New and Changed Functionality

This section contains a list of new and changed functionality in SDL Suite and TTCN Suite 4.5. The issues documented here are added or modified since the SDL Suite and TTCN Suite 4.4 (4.4.0) release.

## Common SDL Suite and TTCN Suite Tools Features

### Patches and Error Descriptions

- Patches to the SDL suite and TTCN suite software, as well as descriptions of known errors and corrected errors, are available on the Telelogic Help Desk web page.

  This page can be directly accessed with the Organizer *Help* menu command *Help Desk*. It also possible to manually login on the page http://www.telelogic.com/usersupp/custarea.asp

  Use your personal user name and password, or obtain a personal user name and password by using the information found in the dialog box Telelogic License Information, which is opened with the Organizer *Help* menu command *License Information*.

  **Note that this feature requires a valid maintenance agreement.**

## The SDL Suite Tools and Features

### SDL Editor

- Reference symbols

  In the SDL Editor it was not allowed to have reference symbols having syntax errors. This limitation does no longer exist. Also it is possible to use the same name having different case if the preference SDT*CaseSensitive is on. These names will also be correctly shown in the Organizer.

- SDL merge facility - Compare

  The text compare is integrated in the normal diagram compare. It is available regardless where the compare is started, in the Organizer or in the SDL Editor. When two diagrams are compared side by side a small dialog is used to navigate between the differences. This dialog has been updated to include an extra button "`Text Compare`". Normally this button will be greyed but when two text symbols differ it will be available and the user can enter the text compare dialog.

The text compare dialog is made up from two lists of text that are placed side by side with a "gutter" that will contain extra information about the differences that this information consists of:

– A "-" indicates that there is no corresponding line in the text compared to the other text.
– A "|" character indicates that the line differs in the two texts.

The current differences are indicated with a highlight. Buttons make it possible to navigate to "`Previous`" and "`Next`" difference item.

- SDL merge facility - Merge

  The merge will be conducted on "per page" basis, text symbols that can be selected from either of the compared diagrams will have the options to be merged together into a new text symbol.

  When manual merges are conducted the option to merge text symbols will always be available in the navigation dialog. An extra button will be added similarly to the compare functionality. As in compare the button will be mostly greyed. When a text symbol is selected the user has the option to select either symbol or merge the text in the symbols into a new symbol. If the text merge is selected a text merge dialog is present with three text lists, the two source texts and the result text. It is possible to navigate between the unresolved differences or between all differences. This works in the same way as in the normal merge. The buttons "<<" and ">>" will jump between the unresolved differences and the "`previous`" and "`next`" buttons will jump between all differences.

  The use of "`Accept 1/Reject 1`", "`Accept 2/Reject 2`" buttons will select neither, either or both the texts form merge. The selections from the source files are presented in the result list the "gutter" will show what version that is selected. It is also possible to edit the result line by line by selecting a line and edit it below the list. Clicking the "`Change`" button will replace the changed line.

- Print only changed parts of an entity

Print only changed entities of a compare operation. A “`Select and print differences`” options has been added to the “`Organizer::Tools::Compare SDL Diagrams`” dialog and this will print all pages with differences and is started from the Organizer.

The “`Select and print differences`” option in the “`SDL Editor Compare Diagrams`” dialog is updated to the same behavior as in the Organizer, that is to only print pages with differences.

The second “`Compare diagrams`” dialog is always used when doing a graphical compare. It does not matter if the compare operation was started from the SDL Editor, the Organizer or the command line. The second “`Compare diagrams`” dialog will be updated with a print button, to print the page or pages from the current page pair that has differences.

For symbol differences, the symbol background will be colored instead of the symbol border. This makes it easier to find the differences on the pages, both on screen and in print.

- Enable to move pages within an SDL process

  The up and down quick buttons in the Organizer can move pages within SDL diagrams. If the user clicks on the down button and an SDL diagram page is selected it will be moved down. The SDL Editor will be accessed to execute the command and the diagram will become dirty.

  In the SDL Editor the page move functionality is placed in the “`SDLE::Pages::Edit Pages...`” dialog and the “`Move Up`” and “`Move Down`” buttons will perform the operation.

  The “`Rename Page`” dialog accessed from the “`SDLE::Pages::Edit Pages...`” dialog is updated to allow the user to decide about auto-numbering of pages.

  The undo functionality in the Organizer and in the SDL Editor will have no effect on pages moves. However, it is easy to undo a page move operation just by moving the page back again.

- Possibility to view and work with a less detailed version of the SDL code

  It is now possible to work either on a less detailed level (“high-level view”) or the usual detailed SDL view. In the SDL Editor, a group of symbols and lines can be selected, and the user can then decide if they should belong to the detailed view only, or both views. By default, symbols belong to the detailed view only. Symbols that also should appear in the high-level view must be marked as important, with the menu choice `SDLE::Edit:Mark as Important`. There is also an `SDLE::Edit::Mark as Detail` menu choice to go back to only showing the selected symbols in the detailed view. In the editor, `CTRL+M` will mark a symbol or a group of symbols as important or detailed, depending on the current status.

It is possible to look at a diagram either in a detailed default view, or in a high-level view, and also to print either the detailed or the high-level view of a diagram.

The high-level view is invoked with `SDLE:View:Show High-Level View`". When the high-level view is shown, the diagram is read-only and unused space previously occupied by detailed symbols and lines is removed with the help of the tidy up operation. Thus, the diagram can only be edited in the detailed default view, where all symbols are visible.

### Text Editor

- An undo command has been added to the text window in the SDL Editor. By using this, it is possible to revert to how the text looked like, before one of the following events happened:

  – The text was selected for editing.
  – A cut/paste/clear operation was done.
  – An import operation was done.

  It is a one step undo/redo facility. When leaving the text window the undo information will be lost.

### SDL to C Compiler

- #CODE directives is supported in compound statements.

  See "#CODE directives in compound statements" on page 2657 in chapter 57, *The Cadvanced/Cbasic SDL to C Compiler, in the User's Manual* for more information.

- Optimized generated code

The SDL to C Compiler has been optimized so that temporary variables that are not used, no longer are generated.

### CPP2SDL

- New -extsyn flag

  The C code generator has been improved to handle CPP2SDL imported synonyms in a more optimal way. This flag overrides the optimization.

- New option to not use full path in include in generated header file

  There is a new option to the cpp2sdl tool. In Organizer a new check box is added in the graphical CPP2SDL menu named “Do not use full path in include in generated header file” and as default it is unchecked.

  Checking the box will prevent usage of path in the includes and make it easier to move files to other directory locations.

  When calling the cpp2sdl tool from the command line the options is “`-noabsolutepath`” and shortest text that will be understood is “`-no-abs`”.

- CPP2SDL options possible to use in sdtbatch option file

  There has been added a possibility when running sdtbatch to be able to change the CPP2SDL options used in an SDL system.

  Normally an SDL system’s CPP2SDL options are saved in one or several import specification (“`.is`”) files. However, these options will in batch be overwritten by eventual CPP2SDL options given in a batch “`.sdt`” option file.

  A section can be added for CPP2SDL options in the “`.sdt`” -file. Batch CPP2SDL options will be used instead of the options in the “`.is`” -files of the SDL system.

  Example of CPP2SDL section in a batch “`.sdt`” -file:

```
[CPP2SDLOPTIONS]
InputLanguageC=False
InputLanguageCpp=True
InputLanguageBorland=False
InputLanguageMicrosoft=False
InputLanguageGNU=False
```

```
RTTI=False
RecognizeSDLsorts=False
ObjectSlicing=False
Preprocessor=""
UsePreprocessorOptions=False
PreprocessorOptions=""
PtrPrefix=ptr_
KeywordPrefix=keyword_
ArrPrefix=arr_
IncompletePrefix=incomplete_
TplPrefix=tpl_
UscoreSuffix=uscore
NoAbsolutePath=True
GenerateCPPTypes=False
OptimizeClassPointers=False
```

The options reflect the ones that are available in the graphical CPP2SDL options menu in Tau Organizer and which also are in an import specification (“.is”) file.

### SDL Simulator

- Improved MSC test script usage

  It is now possible to select for which specific processes or blocks a cui script shall be generated, if an MSC test case describes several blocks. Through an instance translation table, that is a plain text file in the Organizer with the extension `*.itt`, it is decided which instances in the MSC that should be regarded as an environment. The file states for example that an instance A is instance env_2 and the signals to and from this environment will be included in the cui script to run the simulation.

- Checking that signals are not sent to the environment

  Running simulator scripts will now check for unexpected signals sent to the environment. If unexpected (unchecked) signals are detected **before** the **currently** checked signal the test case will fail and a message will be displayed:

```
Unexpected signal(s) to environment arrived before
currently checked signal:
* OUTPUT of DSignal to env:1
* OUTPUT of ASignal to env:1
```

  If unexpected (unchecked) signals are detected **after** the **last** checked signal in the script the test case will fail and a message will be displayed:

```
Unexpected signal(s) to environment arrived after
last checked signal:
* OUTPUT of CSignal to env:1
```

Make sure that the script will run enough for the unexpected signals after the last checked to arrive at the environment to make sure that it will be detected. This can for example be achieved with a finite number of “next-state” commands at the end of the test script. The statistics displayed after the test script has executed have been unexpected signals found in the test case as follows:

```
Command statistics:
        checked: 2
        failed: 0
        updated: 0
        unexpected: 2
```

- Possible to limit length of signal parameter to MSC trace

  When using Simulator with MSC trace it is now possible with environment variable SDTMSCMAXPARAMLENGTH to set a maximum parameter trace length.

  This can be useful when using very large parameters that can cause the MSC editor to work slowly.

  If a parameter exceeds the maximum length it will be truncated to the maximum length with an ending “...” to indicate that the parameter has been truncated.

  After the ending “...” there can be possible ending parenthesis, apostrophes or quotes to match possible initial ones to make the signal parameter syntactically correct.

## The TTCN Suite Tools and Features

### Table Editor

- Auto completion of identifiers.

  When editing tables, it is possible to auto-complete identifier table names while editing text in the Table Editor.

  Start to type any identifier (table with this name should already exist in the test suite). Type 1 or more letters, then press <CTRL-space>. TTCN Suite will find the tables that matches the text typed. If one match is found, the typed name will be completed with the found one. If several matches are found, a pop-up window with a list of names will be displayed. It is possible to select one by using keyboard (“Up”, “Down” arrows for navigate, “Enter” to select, “ESC” for cancel) or mouse (double-click to select, click outside the window for cancel).

# Known Limitations

The following sections contain a list of known limitations and restrictions for the tools and parts of SDL Suite and TTCN Suite 4.5.

## SDL Suite and TTCN Suite Limitations

### Operating System and Windowing Environment

- **In Windows**, SDL suite and TTCN suite have been built and tested using the following service packs:
  - Windows 98: Second edition
  - Windows NT 4.0: Service pack 6
  - Windows 2000
  - Windows XP

  You may encounter problems when running SDL suite and TTCN suite if you do not have these service packs installed on your system.

- When you run SDL suite and TTCN suite on X terminals or PCs with X server software, mouse-clicks are sometimes not transmitted quickly enough for SDL suite and TTCN suite to recognize it.

  The X resource `*multiClickTime` can be used to change the default threshold of 250 ms if you have a UNIX terminal with the `DISPLAY` environment variable set to your local X server. You can write:

  ```
  xrdb -merge
  *multiClickTime: 1000
  ^D
  ```

  The appropriate value (in milliseconds) may depend on your hardware/network.

- The X server delivered with HP-UX 10.20 need to be updated with patch PHSS_12470 from HP, to prevent the X server from crashing when you run SDL suite and TTCN suite.

### On-Line Help and Documentation

- When links are followed from the index in the on-line help, the positioning in the HTML help file may not always be exact. In some cases, the positioning is made a few lines down in the describing text. In the case of tables, the positioning may be after the table.

The workaround is to scroll the text up to find the heading, bulleted item or table row describing the corresponding topic.

- In chapter 78, *SOMT Tutorial, in the User's Manual*, it is assumed that you have basic knowledge of the Link Manager. However, this tool is not covered in the earlier tutorials. For information about the Link Manager, see "Implinks and Endpoints" on page 427 in chapter 10, *Implinks and Endpoints, in the User's Manual*.
- There are no tutorials on the State Chart and HMSC Editors.
- In the MSC Editor, the appearance of some MSC symbols and lines has been updated but all MSC diagrams depicted in the documentation have not been updated accordingly.
- All dialogs and diagrams in the documentation containing the previous release versions in paths, etc. have not been updated to reflect SDL Suite 4.5.
- If the last instance of Netscape was an application other than the browser, help files will not open. The work around is to change the preference for Help*NetscapeCommand to `netscape -noraise -remote 'openURL(URLTOOPEN,new-window)'`.

### Licensing

- The license timeout facility does not support the TTCN suite or tools that are run in stand-alone mode. In general, stand-alone tools do **not** release their licenses immediately when they do not need them any longer, with the exception of the Cadvanced/Cbasic SDL to C Compiler.

### ASN.1 Utilities

The ASN.1 Utilities handle all constructs of ASN.1 as defined in ITU-T recommendation X.680 with extensibility features, X.681 and X.682. Some of the X.683 construct are also supported. See restrictions below for unsupported constructs. There is no support for features defined in the old ASN.1 version X.208.

#### Restrictions to X.680

ASN.1 Utilities handle all constructs defined in X.680, the following semantic concepts of X.680 are not supported:

- Modules are distinguished from each other only by module name. Object identifier following the module name in GlobalModuleReference is ignored and module name is always considered definitive.
- Values can not be assigned to EmbeddedPDVType, ExternalType and UnrestrictedCharacterStringType, these types can not be constrained.
- The contents of character strings representing GeneralizedTime and UniversalTime types is not checked.

#### Restrictions to X.681

- Object class references (X.681, 7.1) can contain lower-case letters. Object class reference consist of a sequence of characters as specified for a type reference.
- Lower-case letters and digits can be included to Word item (X.681, 7.9). Word consists of a sequence of characters as specified for a type reference.

#### Restrictions to X.682

- At-notation is not checked when assigning values to the types with component relations.
- Multiple table constraints on one type are not allowed, for example "MY-CLASS.&TypeField ( {MY-SET1} ) ( {MY-SET2} )" is not allowed.

#### Restrictions to X.683

- Parameterized object classes, objects and object sets and value set types are not supported; only parameterized types and values can be specified, though all of the above entities can be used as parameters themselves.
- Dummy governors are not supported.

#### Restrictions to X.690 and X.691

- Only one reference to identifier component is supported by ASN.1 coders. For the components restricted by two or more component relations in the AtNotationList (see X.681, 10.7), only the first restriction will be handled by the ASN.1 coders.

- X.690 8.11.3 restriction. The order of data values in a set value decoding restricted by canonical order (X.680 8.4) as default compile configuration. Use CODER_BER_CANONICAL_OFF compile switch to avoid of this restriction.
- The following ASN.1 types defined in X.680 are NOT supported by ASN.1 encoders and decoders:
  - BMPString
  - GeneralString
  - GraphicString
  - ISO646String
  - TeletexString
  - T61String
  - UniversalString
  - UTF8String
  - VideotexString
  - UnrestrictedCharacterStringType
  - EmbeddedPDVType
  - ExternalType
  - InstanceOfType
  - ObjectDescriptor

#### Restrictions to Z.105

- The following concepts of Z.105 are not supported by the ASN.1 Utilities:
- ASN.1 definitions can not be put in text symbols in SDL diagrams; they can only be placed in separate ASN.1 modules, whereas Z.105 allows free mixing of SDL and ASN.1 definitions.
- Value notations for types defined in another module are not supported. For example; val X ::= 3 is only allowed if X is defined in the same module. External value references should be used instead.

- Use of the ASN.1 value notation within SDL expressions is not supported for bit string, octet string, set, set of, sequence, and sequence of.
- Value notation for SEQUENCE/SET where optional/default components have been assigned a value cannot be mapped to SDL unless SDL optional support in make operator is switched ON.
- UniversalString, GraphicString, and EXTERNAL are translated to SDL types with the same name, but no implementation for these types is available (have to be provided by you). The reason is that Telelogic considers the mapping in Z.105 as insufficient (does not really allow extended character sets).
- PLUS-INFINITY and MINUS-INFINITY are translated to some large numbers. If these are not large enough for a specific platform/application, you must define these SDL synonyms yourself.
- If there are name clashes between named bits, named numbers or ASN.1 values, only the first occurrence of the synonym will be generated to SDL, for example, for ASN.1 definitions `T1 ::= BIT STRING { a(3) }`, `b INTEGER ::= 11` and `T2 ::= INTEGER { a(4), b(5) }` only synonyms `a` from BIT STRING and `b` for INTEGER value will be generated to SDL with warning messages because generation of all synonyms according to the Z.105 standard will result in an SDL analyzer error.
- In SDL (Analyzer) literals and synonyms with the same name will clash if they have the same name. So, for the following ASN.1 definitions `T ::= ENUMERATED {x, y}    x INTEGER ::= 15 val T ::= x` generated SDL `newtype T literals x,y operators ordering; endnewtype; synonym x Integer = 15; synonym val T = x;` will not compile: *`ERROR 375 Type mismatch for synonym or literal: synonym val T = ? x INFO 352 x is one of the visible definitions`*. Although a simple workaround for this problem is to add a qualifier before value `x` to the generated SDL : `synonym val T = <<type T>>x;` .

### Print

- When printing to EPS files (one per page) the files will be named after the page in a case sensitive way. On Windows, when pages have the same name only differed by case, only one file will be printed containing the last page, due to the limitation in Windows concerning case.

- Colors are not supported for FrameMaker or Interleaf printing.

## SDL Restrictions

This section describes the major SDL restrictions with respect to the ITU-T Z.100 recommendation of 1992. The restrictions lie in one or several specific tools. This means that even if a concept is not fully supported, it may sometimes be possible to use it in a limited part of the tool set.

This listing deals with restrictions in the following tools:

- SDL Editor
- GR to PR Converter
- SDL Analyzer
- SDL to C Compiler (Cbasic and Cadvanced)

For more detailed information on the SDL restrictions in the Analyzer and the SDL to C Compiler, see “SDL Analyzer” on page 37 and “SDL to C Compiler” on page 40.

For a description of SDL restrictions regarding the Simulator, the Validator, the TTCN Link kernels and the Cmicro SDL to C Compiler, please see the following sections in the User’s Manual:

- *“Restrictions” on page 2163 in chapter 50, The SDL Simulator*.
- *“Restrictions” on page 2318 in chapter 53, The SDL Validator*.
- “SDL Restrictions” on page 1381 in chapter 36, *TTCN Test Suite Generation*.
- “SDL Restrictions” on page 3358 in chapter 66, *The Cmicro SDL to C Compiler*.

| Z.100 Chapter | Concept and Restriction | Affected Tools |
|---|---|---|
| **2.2.2 Visibility rules, names and identifiers** | <name>: spaces in <name>s are not supported (a space is a shorthand for <underline>) | SDL Editor, Analyzer, SDL to C Compiler |

| Z.100 Chapter | Concept and Restriction | Affected Tools |
|---|---|---|
| **2.4.1.2 Package** | <interface> definitions, and <definition selection list> in use statements, are not supported. | Analyzer, SDL to C Compiler |
| **2.5.1 Channel** | **nodelay:** channels with delay and channels with nodelay are treated the same. | SDL to C Compiler |
| **2.7.4 Output** | **via all** to a <block instance set> only sends a single signal. | SDL to C Compiler |
| **2.9 Internal input and output** | The <internal input symbol> and the <internal output symbol> are not available. | SDL Editor |
| **3.2 Partitioning** | <channel substructure definition> is not supported in generated code. | SDL to C Compiler |
| | In substructures with both process definitions and block definitions, the process definitions will be ignored. | Analyzer, SDL to C Compiler |
| **3.3 Refinement** | <signal refinement> is not supported. | SDL to C Compiler |
| **4.2 Macro** | <macro definition>: macros and macro calls are transformed to PR and expanded in PR form, which restricts the usability of macros. | Analyzer |
| | <any area>: graphical macros can only contain flow symbols in one single flow. | SDL Editor, GR to PR converter |
| | <macro call>: graphical macro calls are not allowed in a higher scope than processes. | SDL Editor, GR to PR converter |
| **4.3.3 Optional definition** | <select definition>: textual optional definitions are supported, but not graphical. Any (external) synonyms used in the select expression must be defined in the system definition which may not be type based. | SDL Editor, Analyzer, SDL to C Compiler |

| Z.100 Chapter | Concept and Restriction | Affected Tools |
|---|---|---|
| **4.3.4 Optional transition string** | <alternative question>: question part in transition option must not contain external synonyms. | Analyzer, SDL to C Compiler |
| **5.2.3 Axioms** | <axioms>: not supported. | Analyzer, SDL to C Compiler |
| **5.2.4 Conditional equations** | <conditional equation>: not supported. | Analyzer, SDL to C Compiler |
| **5.3.1.14 Name class literals** | <name class literal>: not supported. | Analyzer, SDL to C Compiler |
| **5.3.1.15 Literal mapping** | <literal mapping>: not supported. | Analyzer, SDL to C Compiler |
| **5.4.4.5 Timer active expression** | <timer active expression>: cannot be used for timers with parameter. | SDL to C Compiler |
| **6.2 Context parameter** | <formal context parameters> and <actual context parameters> are not supported. | Analyzer, SDL to C Compiler |

## The SDL Suite Tool Limitations

### SDL Editor

- Diagram names may not exceed 255 characters.
- If the colors red and blue are used for some symbols, you may find it hard to interpret the graphical view resulting from the Diff operation, as these colors are used for highlighting the differences between two diagrams. To avoid this problem, do not use the colors red and blue on symbols when you compare diagrams.
- The special graphical symbols for internal input and output are not available, as the use of them is explicitly discouraged in Z.100. The same behavior can be achieved with the normal input and output symbols.
- Graphical macros (macro diagrams) can only contain flow symbols in one single flow, without branches. Graphical macro calls are not allowed in a higher scope than process (system, block, etc.).

- The option symbol (dashed polygon) is not available.
- Nested diagrams are not supported, that is, diagrams must contain reference symbols to diagrams at lower levels. For instance, a system with a block must be drawn as two diagrams, where the system diagram contains a block reference symbol to the block diagram.
- When the preference Editor*AlwaysNewWindow is used and for example the search command is used, new windows can be shown. As long as the dialog are still opened the menu bar for the new window should be dimmed but some commands are available. Choosing these commands should be avoided when the dialog is shown.

### Print

- The FrameMaker or Interleaf integration do not support colors.
- It is only possible to change the font used to print text files by using the preference SDT*PrintFontFamily.
- When printing to html, diagram page names containing “\“ are not supported.

### Emacs Integration (UNIX only)

- To function properly, the SDL suite assumes GNU Emacs 19.31 or later.
- Undo is not supported for the link handling commands.
- If Emacs is started from SDL suite and TTCN suite and SDL suite and TTCN suite is shut down, but not Emacs, the Emacs session cannot be re-connected to another SDL suite and TTCN suite session.

### Microsoft Word Integration (Windows only)

- MS Word 7.0, MS Word 8.0 and MS Word 2000 are supported, see certification matrix.
- When a MS Word document becomes modified, its icon is not marked as “modified” in the Organizer.
- A new MS Word document is always named “Document<x>”, where <x> is an integer, independently of what name it was given in the Organizer.

- Undo is not supported for the link handling commands.
- Endpoints in MS Word documents sometimes need to be synchronized with the Link Manager manually: Use the *Update* menu choice.
- In the Organizer, multiple instances of MS Word cannot be handled.
- Due to problems with recognizing read/write security permissions on NTFS files/directories, some odd behavior in the communication with MS Word may occur. This shows up as two error boxes with almost the same contents, one in MS Word and one in the Organizer. In the Organizer, the access rights on such files are incorrectly marked as “rw”.
- Since part of the implementation is done with MS Word macros, conflicts may arise when combined with other templates/macros. This must be considered on a case-to-case basis and possibly the different macros may have to be merged. The implementation does not work with for example “Macro Virus Protection Tool” from Microsoft.
- If MS Word is started from SDL suite and TTCN suite and SDL suite and TTCN suite is shut down, but not MS Word, the MS Word session cannot be re-connected to another SDL suite and TTCN suite session.

## SDL Analyzer

### Implementation Restrictions

The present version of the Analyzer is an SDL-92 analyzer. The restrictions and the implementation limits are described in this section. When possible, a work-around is described.

- Macros and macro calls are transformed to PR and expanded in PR form. This restricts the usability of macros, for instance the number of inlets and outlets must be 0 or 1. A common case that fails is a macro with states that is called from a branch of a decision. It fails because the GR to PR converter does not know the contents of the macro and treats the call like a task symbol. Other things not supported because of this, are variable declarations and type definitions.

- The Analyzer assumes the layout of external properties to follow the example below. That is, each external formalism name should be on the same line as the alternative keyword and the external data descriptions should be on lines of their own.

**Example 1**

```
alternative a,b;
   abc abc t5;
   abc abc t5-2;
endalternative;
b;
```

#### The PR to GR Converter

- The PR system will be changed to have all its definitions referenced to match the way the SDL suite handles diagrams. A referenced definition needs a qualifier if name and type are not unique in the system. Such qualifiers are not supplied automatically: use the *Update Heading* command in the Organizer.
- Comments in PR files `/* comment text */` may sometimes be skipped during conversion into GR format.

#### Restrictions on Syntactic Analysis

- The Analyzer cannot handle spaces in names (a space is a shorthand for an underline). Replace spaces with explicit underlines.

#### Restrictions on Semantic Analysis

- The semantics of virtual types is unclear.
- Gate constraints on implicit signal routes are not checked.
- Implicit gate connections are not checked.
- Signals of global procedures are not properly checked.
- Only limited tests on output via.
- It is not checked that instantiated types have signals in the gate constraints.
- Context parameters are not implemented.

- Conditional equations are not implemented.
- Some checks may be lost in nested virtual definitions.
- Package interfaces and definition selection list in USE statements are not implemented.
- Remote procedure calls are not checked against imported procedure specifications.
- Imported procedure specifications are not checked against exported procedures.
- Import expressions are not checked against imported variable specifications.
- Imported variables are not checked against exporting processes.
- The check that substructure name in subtype and supertype must be the same is not implemented, since it would make the use of “block in block” useless for inheritance.
- Operator definitions are not checked against operator signatures.
- Name class literals in user defined data types cannot be handled in the Semantic Analyzer. A nameclass literal is a shorthand for writing a (possibly infinite) set of literal names defined by a regular expression. List the literals explicitly or use inheritance from the predefined data types.
- Question and answers in decisions are always considered formal, that is, informal text is considered to be a character string.
- Inheritance from data type definitions containing qualifiers referencing that type definition is not allowed, since qualifiers are copied and not changed during the expansion of inheritance.
- When analyzing part of a system, checks depending on the omitted parts will not be performed.
- When a substructure contains both processes and blocks, it should be possible to select which implementation that is preferred. This is not possible; the process definitions will not be analyzed.
- Signal refinement is not supported.

#### Semantic Checks that are not Performed

Analysis of the following semantic SDL rules is not performed:

- The consistent partitioning subset must be a consistent refinement subset.
- The decision answers must be mutually exclusive. This is not tested for all possible data types.
- An exported variable must exist for each imported variable.
- The type check and the evaluation and check of the equivalence classes in axioms is not performed.
- Generator names and generator formal parameters must not be used in a qualifier, qualified, followed by an exclamation, or used in a default assignment.

### SDL to C Compiler

- Channel substructures are not allowed.
- Signal refinements are not supported.
- Context parameters are not supported.
- Timer active expressions for timers with parameters are not allowed.
- In generated code there will be no difference between a channel with delay and a channel with NODELAY.
- Integer and Real are restricted in range and precision, as they are implemented using the C int (32 bit integers) and double types. Charstrings may not contain the character NUL, as Charstring is implemented using `char *` in C, and `char *` is terminated by NUL (=0).
- Axioms and literal mappings may be part of an abstract data type definition, but that information is not used in any way by the SDL to C Compiler.
- In abstract data types, the following concepts cannot be handled:
  - Name class literals
  - Naming a literal using a Charstring literal

- A sort definition may not refer to itself, directly or indirectly.
- A component of a struct will, in the generated code, have the same name as in SDL. Characters that are not letters, numerals or underscore characters will, however, be removed. The remaining part of the name must be a valid C identifier to be accepted by the C compiler, that is:
  - It should start with a letter.
  - It should not coincide with a C reserved word.
  - It should be unique within the C struct representing the struct.
- Multiple paths from a decision for a certain decision expression value are not checked. A simulation program will simply choose one of the existing paths.
- Overflow of integer and real values are checked at the C level if the actual C system performs these checks.
- A process that exports procedures cannot call (directly or indirectly) a global procedure containing states.
- Output VIA ALL C, where C leads to a block instance set, is not handled correctly. One signal should be sent to each block instance, but only a single signal will be sent.
- The `.ifc` file contains `#define` definitions for the kind of SDL units for which this is possible. All the SDL to C Compilers follow this principle. This can, depending on the C compiler, lead to compilation errors/warnings or to runtime errors in some cases. The following example shows how the problem may occur:

```
newtype x
   literals green, blue;
endnewtype x;

newtype y
   literals blue, green;
endnewtype y;
```

The both literals `green` and `blue` will then be created like this:

```
/* For newtype x */
#define green  0
#define blue   1

/* For newtype y */
#define blue   0
```

```
#define green  1
```

The C compiler then usually complains that the macros `blue` and `green` are redefined with a different value. Some C compilers will continue in any case and produce an executable. Runtime errors are the consequence on this, because when `green` and `blue` are used together with the first `newtype x`, the literal values from the second `newtype y` are used by the C compiler. In order to avoid this problem, a prefix for each literal should be manually added, like this:

```
newtype x
   literals x_green, x_blue;
endnewtype x;

newtype y
   literals y_blue, y_green;
endnewtype y;
```

The `green` and `blue` example is not a good one, because it is possible to avoid the problem by using the inherit construct. The literals from the `newtype x` could be inherited into the `newtype y`. More problems will occur if several entity classes are mixed up from the C compiler, like if there would for example be a signal called `green`, which is also created as a `#define`. You are strongly recommended to avoid name conflicts in the `.ifc` file by manually adding prefixes.

- any(Sort) where Sort is a syntype is only implemented if the syntype contains at most one range condition which is of the form a:b, that is one limited range. If it is a syntype of a real type, e.g. Real or Time, with a range condition it is not implemented.
- any(Real) does not generate random decimal values, only whole numbers.
- When the preference SDT*RelativeSDTREF is on, and Edit Separation is used with a separate file name, compile errors may occur requesting a file that does not exist. This can be resolved by either not using a separate name or turning the Relative SDT References off.

### Targeting Expert

- The automatic and manual configuration/scaling for the Cadvanced libraries will not be used per default in all pre-defined integrations.

The compiler flag `-DUSER_CONFIG` must be set to use configurations.

- Targeting Expert is built to comply with Borland Builder, not Borland 5.02. The Alignment settings for these two compilers differs.

  The solution is to change the alignment to 8 for all types in the “Host Connection” section in Targeting expert. I.e “Target data coding: alignment” should have all entries set to 8. These values are read from the file sdtmt.opt.

### CPP2SDL

- The following C/C++ dialects are supported by the tool:
  - ANSI C/C++
  - Microsoft Visual C/C++
  - Borland C/C++
  - Gnu C/C++
- The tool performs a complete syntactic analysis of the input C/C++ program, but only parts of the semantic analysis is done.
- C++ exceptions are not supported.
- C++ template declarations are not translated to SDL directly. Instead, instantiations of the templates must be provided, and these will be translated.
- The support for function pointers is limited. Function pointer types are represented in SDL (as untyped pointers), but it is for example not possible to call the functions pointed at.
- Expressions containing usages of the `sizeof` operator are not always correctly evaluated.
- C++ cast expressions are not fully supported.
- Members inherited from a virtual base class are sometimes not accessible in the SDL translation.
- Classes that inherit from multiple template instantiations of the same template are not fully supported.
- "`sizeof`" is not correctly handled under certain conditions cause problems.

- There can be problems when using anonymous types for example:

```
typedef struct {
  int i;
  float f;
}mess;
```

instead it should be:

```
typedef struct mess {
  int i;
  float f;
}mess;
```

### SDL Simulator

- If you have already generated an SDL simulator from the Organizer, and want to generate a new one with other options or with another selection, you should perform a Full Make, as changes in options or selection is not handled by the build process. Otherwise compilation or link errors might be the result of the build process.

- There are no range checks implemented for general arrays. General arrays are implemented as linked-lists because the code generator can not determine whether or not it can be implemented as an ordinary array. This occurs if a synonym translated to a variable is used in a range condition of a syntype and the syntype is used as an index sort in an array or powerset instantiation. The reason for this is that the length of the array cannot depend on a variable in C.

  An example:

```
SYNONYM Lo NATURAL = 1;
SYNONYM Hi NATURAL = 10;
SYNTYPE BufferType = NATURAL
CONSTANTS Lo , Hi
ENDSYNTYPE;
NEWTYPE MessageBufferType
  Array( BufferType, CharString)
ENDNEWTYPE;
```

### Validator/Autolink

- Bitfields in struct that also contains pointers are not handled.

- C unions are not allowed to contain pointers.

- Encoding/Decoding functionality cannot be used together with the Validator kernel.

- Validation of MSCs containing a signal sending a parameter of type ptr_char, or Charstar will give the following error: “No reference to dynamically allocated memory.”

  The RefError occurs because when you read in the MSC diagram there is a parameter of type Charstar and reading this will create the dynamic memory which can not be handled correctly.

## Examples

- **On UNIX**, the `phone` example cannot be used together with the graphical SimUI if the Phone application is compiled and linked with the ApplicationDebug kernel.

## Real-Time Operating System Support

### Light Integration

The same restrictions as for the SDL to C Compiler apply. See “SDL to C Compiler” on page 40.

### Threaded Integration

The same restrictions as for the SDL to C Compiler apply. See “SDL to C Compiler” on page 40

- Threaded Integration can only be generated with the Deployment Editor and Targeting Expert i.e. the make feature in the Organizer can NOT be used.
- The Threaded Integration for VxWorks uses POSIX library functions i.e. the VxWorks kernel must be compiled with the POSIX libraries.
- In a Threaded Integration, the xInEnv function is invoked when a state transition occurs in the SDL system. This means that intervals between the xInEnv calls are irregular and depend on the properties of the running SDL system. When writing external code for input into an SDL system, it is recommended that the code is run in a thread of its own, using xGetSignal and SDL_Output for inserting signals into the system.
- On-line MSC trace under VxWorks softkernel does not work.

#### TCP/IP Communication Module

- On VxWorks, the TCP/IP communication module has been tested on a softkernel running on Solaris. In order to connect to a Vxworks softkernel application, the loopback IP address of the host machine must be used. In most cases, this IP address is “127.0.1.0”.
- On OSE, the TCP/IP communication module has been tested on a softkernel running on Solaris. In order to connect to a remote component from an OSE softkernel application, the IP address of the remote host must be given explicitly. A regular hostname, e.g. “the_computer.the_company.com”, will not suffice.

#### Tight Integration

- The tight integrations in the SDL suite are only applicable for Cadvanced – not for Cmicro.
- Only Solaris (Posix 4), Win32, VxWorks, and OSE delta integrations are available as downloads.
- Priority input is not handled.
- Continuous signals are not handled.
- Services are not handled.
- An RPC call of value returning procedure without parameters does not work.
- The SDL `ANY` construct is not properly implemented.
- The timer set construct `SET(5,T1)` is not supported. Instead use `SET(NOW+5,T1)`
- ADT Library files located in `$telelogic/sdt/include/ADT`:
  - file.pr:

    Should work if supported in the RTOS environment. It might be necessary to enclose the function `xGetValue` with the following code to be able to compile with some compilers.

```
#ifdef XMONITOR
...
#endif
```

  - list1.pr and list2.pr:

Should not be used since they are implemented with dynamic pointers which could lead to memory leaks.

- byte.pr:

  Works OK.

- unsigned.pr, unsigned_long.pr, longint.pr:

  Works OK.

- pidlist.pr:

  Works OK but only with the version supplied in the RTOS delivery:
  `$telelogic/sdt/sdtdir/RTOS/INCLUDE/pidlist.pr`.

- idnode.pr:

  The functions `FirstPId` and `NoOfProcesses` may work under some conditions, the rest of the functions do not.

- pointer.pr:

  Should not be used since there is a big risk of memory leaks.

* Support for Kernighan & Ritchie C is discontinued.
* The integration code does not check the upper limit for the number of process instances as declared in the SDL system.
* The same data type is used to represent TIME and DURATION; in most operating system this is an unsigned integer. Thus TIME or DURATION expressions may not evaluate to negative values. This also applies to intermediate expression results.
* The SDL error term is not supported in Cadvanced Tight Integrations. E.g. “return error;” can not be used in Cadvanced Tight Integrations.
* REVEAL and VIEW are not supported. Applications using this concept may crash or behave in an undefined way due to concurrency problems.
* EXPORT and IMPORT are not supported. Applications using this concept may crash or behave in an undefined way due to concurrency problems.

- In addition, the same restrictions as for the SDL to C Compiler also apply. See “SDL to C Compiler” on page 40.

### SDL C Compiler Driver (SCCD)

- RESTRICTION: SCCD under DOS/Windows does not handle include paths in double quotes.

### Cmicro SDL to C Compiler and Library

- The names of processes in different blocks must not be the same. The reason for this restriction is the missing symbol tree in Cmicro. It is impossible to address for example two processes with the same name in the same C module (in the SDL environment). As a workaround, you should always have different names for the different processes in the system. You can easily accomplished this by adding a prefix to each process name.
- Predefined sort carray:

  It is impossible to return variables of the predefined sort carray from an SDL procedure because the ANSI C language cannot handle this. It is however possible to enclose the carray into a newtype struct value as a workaround.
- Timers with parameters and the preemptive Cmicro Kernel do not work together.
- If a procedure returns a value of a sort that is passed as address, an assignment like `call prd(invar, retvar);` is not supported. Instead use `retvar := call prd(invar);` .
- You shouldn’t have syntypes named as the following:

```
int
unsigned_int
long_int
unsigned_long_int
short_int
unsigned_short_int
char
signed_char
unsigned_char
float
double
ptr_void
bool
wchar_t
```

### SDL Target Tester

- When the target system is suspended (by sending the suspend command), any signal that is sent to the target will be discarded.
- When, with the Output-PAR/NPAR command, you want to send a signal to the target system, a list with all the signals that are defined in the system is presented. You should keep in mind that you always have to address the right process instance when specifying the parameters of the SDL Target Tester's output command.
- It is not possible to send charstrings or pointers with the Output-PAR command to the target.
- Several compilers, e.g. the Tasking 80166, are giving incorrect information about the message length.

  This error is detected by the SDL Target Tester and displayed once. The trace of SDL events will be correct as only the CRC check is not involved in the following communication.
- The SDL Target Tester should only be started from the Targeting Expert's *Tools* menu since there may be problems when the SDL Target Tester is invoked standalone.
- Page file and convert file in SDL Target Tester does not work.
- **On Unix**: The Cmicro Postmaster leaves temporary files below `/tmp`. You should remove these files from time to time by hand (`rm /tmp/.sdt.cmicro.*`)
- It is impossible to specify a breakpoint on a timer input.
- If the debug mode of Cmicro Postmaster is used, there might be too long messages, in which case the Postmaster will crash with a core dump.
- If the line numbering in the SDL Target Tester is switched on and off again, sometimes the first columns in the text area are split. As a work around and to prevent this, line numbering always should be switched on at the beginning and never switched off again.
- The context sensitive help sometimes indicates that a command may be used, though the command cannot be used in current context (setting a breakpoint is allowed although no target system is connected and running).

- **On UNIX**: If the `sdtmt.btn` file contains too many entries that are marked as “:MENU”, the SDL Target Tester is not able to scale its window size when it is invoked. If you resize the window manually, the size will be correct.
- The Parametertrace is restricted to Predefined Sorts, Syntypes, Enums, Newtype Structs and Newtype Array.
  If the SDL Target Tester encounters an unknown data type, the rest of the parameters is shown as hex-buffer.
  Limitations:
  It is not possible to show the contents of a Charstring.
  If you use wrong values for `LENGTH_`, `ALIGN_` and `ENDIAN_` in the file `sdtmt.opt` it is possible that the Tester decodes all parameters, but the shown values are wrong.
- Searching the Text trace in the Target Tester with regular expressions is disabled by default. The Tester crashes if reaching the end of the trace, i.e. there is no more regular expression found in the trace.
  The feature can be enabled by setting the environment variable TT_REGEXP to 1 before starting the Target Tester.

### Application Generation with Cmicro

- Not all possible errors are checked by the Targeting Expert. In most cases, a compilation error is the result you will get from an incorrect configuration. The on-line help will support you to configure correctly.
- If you try to `#include "systemname.ifc"` in `ml_typ.h`, or for some other reason, there may be further naming conflicts, for example during preprocessing:

```
sctpred.c: In function 'yAss_SDL_Object_Identifier':
     ml_pred.c:2109: parse error before '0'
     ml_pred.c:2109: parse error before '0'
```

  The reason for the error above (and other behavior that is impossible to foresee) was that there was a signal named “`SIn`”. This will be generated as a `#define SIn 0`. In the `sctpred.c` file, there is a variable called `SIn` (in the macro `yAss_SDL_Object_Identifier`). The C preprocessor then replaces the variable name with the `#define` value of `SIn`, which ended up in the parse error above.

In order to avoid that, you should never include code generated header or `.ifc` files in any part of the Cmicro Library itself or in the generated C code. The `.ifc` file should only be included in the SDL environment files (like `env.c`).

- In the generated C code, the following warning is repeated for each variable that is unused:

```
"Unused Variable : <variablename>"
```

These warnings in generated C code are not that easy to remove because it is difficult to calculate this during code generation. These warnings should not produce real problems (except that it is sometimes impossible to read the large C compiler output). The following variables are sometimes referred:

```
warning: unused variable 'yOutputSignal'
warning: unused variable 'yVarP'
warning: unused variable 'ySVarP'
```

The same warning as above is thrown from C compilers when signals without parameters are used.

This warning is accepted in the `mk_outp.c` module:

```
mk_outp.c:warning: unused variable 'P_Parameter'
```

A similar warning (accepted also) is within `mk_stim.c`:

```
mk_stim.c:warning: 'SystemTime' defined but not used
```

- Warnings like: “Control reaches end of non-void function”

As the SDL to C Compiler basically creates a `switch()` construct in C, and every return from the process activity description function (PAD function) occurs in all the case branches of that switch, the PAD function contains no return statement after the switch. This is the reason why some C compilers complain that the “Control reaches end of non-void function”. This is an accepted warning. It is possible for you to remove that warning in the `ml_typ.h` or `user_cc.h` file by modifying the `END_PAD` macro like this:

```
#define END_PAD(PrsNameWithPrefix) return (SDL_DASH_NEXTSTATE);
```

- Warnings: missing braces around initializer for `xSTATE_INDEX_zyyyy_xxxxxxx[0]`

This is an accepted error so far, because each known C compiler handles this correctly. The reason why the SDL to C Compiler generates like this is that is still should be conform to not just only ANSI C, but also Kernighan & Ritchie C.

- Some C compilers fail in preprocessing of very long macro definitions (like all the C Compilers available on the market for the 8051 family of CPU's). Such long macro calls occur in the predefined generators (String, Powerset, Bag).

  If the C compiler (or specifically the C preprocessor) complains about a macro definition that is too long, or produces erroneous output, it is not possible to use these kind of generators. The macro `X_LONG_MACROS` should be set with the Targeting Expert.

- It could also be impossible for some C compilers for embedded systems to compile recursive macro calls. Recursive macro calls appear for SDL statements like for example:

  ```
  a:=b and c and d;
  ```

  If such a problem occurs, it is strongly recommended to avoid such statements for example with the following work around:

  ```
  tmp:=b and c, a := tmp and d;
  ```

- When it comes to target applications, you always have to specify how memory is allocated (see C module `mk_cpu.c` in the template directory) and what should happen if there is no more memory available. As the template C files does not know what you want to do, there is a C macro called `NO_MORE_MEMORY` that you must define. Any reaction on such a fatal error might be appropriate, like “Hang up the program”, or “Reset”, or “Print out an error message”. Printing out error messages may however fail as well, if the print function allocates memory.

- The default values for variables are not set in a procedure when `XMK_USE_KERNEL_INIT` is defined. This is a problem which cannot be solved easily.

There are two work-arounds:

– Do not define `XMK_USE_KERNEL_INIT`

– Do not use default values for variables in a procedure, e.g. this is forbidden in a procedure (in a process it does work very well):

```
DCL a integer;
DCL b integer;
   task b:=a+b; /* non deterministic results */
```

This is better:

```
DCL a integer = 0;
DCL b integer = 0;
   task b:=a+b; /* deterministic result ! */
```

### Use of ADTs from the SDL suite in Cmicro

The packages and ADTs delivered with the SDL suite are mainly to be used in Cbasic/Cadvanced applications. Some of these ADTs may however also be useful for Cmicro applications. Others cannot be used together with Cmicro as they contain references to C code from Cbasic/Cadvanced. A list of restrictions and recommendations can be found in “Abstract Data Types” on page 3314 in chapter 66, *The Cmicro SDL to C Compiler, in the User’s Manual*.

### Combining Cadvanced / Cmicro Code

Mixing C code from different SDL to C compilers is not possible as they use their own runtime model and runtime data structures. Trying to mix up the C code will sooner or later lead to compilation errors. This restriction is true for any kind of combination of C code, including sdth2sdl.

## TTCN Restrictions

- The TTCN language supported in the TTCN suite is according to an interim version of the new TTCN standard labeled Delivery 8.3, August 2, 1996, with the exception of the Modular TTCN features that are according to the Delivery 9.6 version.

- Due to the Delivery 9.6 Modular TTCN features, the support for the TTCN Package document type has been dropped.

- The import/export mechanism of the Modular TTCN feature is currently not able to handle import that uses the “xxx[yyy]” construct needed for importing named numbers, and neither able to handle the “xxx::yyy” construct for importing objects with name collisions.

More general restrictions and modifications to the TTCN and ASN.1 support in the TTCN suite is described in chapter 39, *Languages Supported in the TTCN Suite, in the Telelogic Tau 4.5*.

## The TTCN Suite Tool Limitations

### TTCN Browser / Desktop

- If two identifiers in the TTCN differ by a final letter, D, TestCase (Test Suite Variable and TestCaseD) then TTCN Suite could generate code with name conflicts. Example:

```
Name of Test Suite Variable: TestCase

Name of Test Case: TestCaseD
```

### Color Problems on UNIX

- On computers where the display hardware limits the number of concurrent displayed colors, the TTCN suite might run out of colors. It will in this case start to use what appears to be random colors. This should have no effect on the usability of the TTCN suite, but can be somewhat annoying.

### Table Editor on UNIX

- The Table Editor can only display up to approximately 1000 rows. This limitation is only present in the Table Editor, so conversion to or from MP, printing, analysis, etc. is not affected.

### Table Editor on Windows 98

- It is not possible to edit texts larger than 32766 bytes.

### Analyzer

- Special characters in strings, for example “CR” and “ESC”, are not checked during analyze.

- The use of the `R` variable in expressions is only allowed in the dynamic part, not in declarations. The TTCN Analyzer does not detect the use of `R` in declarations.
- The TTCN Analyzer check of send value is incomplete, wild cards in send values are not detected.
- The types BMPString and UniversalString are not supported.
- The Analyzer currently only implements a very basic analysis of the imports and externals tables used for Modular TTCN, and in addition the export tables are not analyzed at all.
- The Analyzer does not support encoding/decoding variations.
- ASN.1 enumerated type identifiers are allowed to start with both upper and lower case characters. This applies only for the TTCN Suite tool and not for the ASN.1 utilities tool.

### External ASN.1 Reference

The definition of the ASN.1 type that is referred is limited:

- References of the form `Module.Type` are not allowed in the definition.
- References to types/values in the same or other ASN.1 module is not allowed in the definition, unless separately defined.
- The limitations for the types and values referenced from TTCN also include those described in "ASN.1 Utilities" on page 29.
- When using ASN.1 PDU Type Definitions By Reference, you cannot employ BER or PER ASN.1 encoding/decoding if you are mapping the PDU names to other names local to the test suite. (The PDU names will not be recognized by the runtime system.)

### TTCN ASN.1 BER Encoding/Decoding

- The following composite ASN.1 types are not supported:
  - EXTERNAL
  - EMBEDDED PDV
  - CHARACTER STRING
  - ObjectDescriptor
  - UTCTime
  - GeneralizedTime
  - TeletexString
  - VideotexString
  - GraphicString
  - GeneralString
  - T61String
  - ISO646String
  - BMPString
  - UniversalString
  - UTF8String (from X.680/1997)

  It is not possible to Encode/Decode values of these types.

### The TTCN to C Compiler

The language supported by the TTCN to C compiler is a subset of the language covered by the Analyzer. This means that in some cases, constructions that have been analyzed correctly are not supported in the code generation phase. The following list describes the known limitations in the current TTCN to C compiler.

- Single value restrictions of SEQUENCE OF, SET OF and CHOICE are not supported.
- The `INTEGER` data type is restricted in range and precision, as it is implemented using the C int data type (usually 32 bit signed integers).
- The use of choice values in brace lists in the constraints or in assignments is not allowed. The way to work-around this is to do the assignments field-wise in the constraints.
- `SUBSET`, `SUPERSET` and `COMPLEMENT` are not supported as values for fields in any structured type.
- The `EXTERNAL` type is not supported.

- The `MIN/MAX` and the inclusive/exclusive (that is `<`) syntax in ASN.1 value range sub types are not supported.
- `INFINITY`, in all its uses, is not supported.
- `COMPONENTS OF` is not supported.
- Selection type and their values are not supported.
- Values of the `REAL` data type is not supported.
- To specify values of the data type `BIT STRING` as a list of named bits is not supported.
- Value length restrictions are not supported.
- Encoding/decoding variations are not supported.
- The maximum number of parameters to a PTC is restricted to 8.

### MSC Logging

The MSC generation part of an ETS is limited in scope and applicability to some particular applications. This list of limitations may be circumvented by manually editing the code in the files `mscgen.h`, `mscgen.c`, `static.c` and `gci.c`. Such edits are not supported by Telelogic, though suggestions of improvements are welcome.

- Distributed concurrent test component logs are not supported.

  The reason for this is that it would require additional communication in-between the distributed test components. The required features for a common log of any kind are not available through the architecture of TTCN or the GCI Interface. The result of applying MSC logs for truly distributed ETSs is not defined. Note that the Composed mode may work but it will produce logs only of one component at a time.

  The internal concurrency mode of the TTCN suite is fully supported by the MSC Generator.

- Re-start of test components

  Concurrent test suites in which one given component is created more than once, will result in MSC/PR that may not be supported by the MSC Editor.

The generated MSC log will contain multiple instances with the same name, though presumably not at the same time. It is conceivable to write a text processing script which renames instances that are created multiple times. The limitation applies only to the Decomposed mode.

- Value notation limitations

  These are limitations to the value encoding for MSC/PR form. For information on how the value encoding can be completely disabled, see "Compiling an ETS with MSC Generation" on page 1297 in chapter 33, *The TTCN to C Compiler (in Windows), in the User's Manual*.

  – Some types may not be supported for the logging of values with the events of the MSC log. If they are encountered, a warning message will be appended to the event log. See the `mscgen.c:MscEncodeValue` function for details.

  – Some minor type encodings may differ from the one produced by the SDL Simulator. These differences can be fixed by editing the `mscgen.c:MscEncodeValue` function. Note that such changes are not supported by Telelogic.

  – Some types that are not supported by the SDL suite are handled. The HEXSTRING and derived TTCN types are not recognized by the SDL suite.

  – The tool has a fixed size buffer for MSC value encoding. The result is that some values may be truncated (values that print to strings longer than approximately 4000 characters). See "Compiling an ETS with MSC Generation" on page 1297 in chapter 33, *The TTCN to C Compiler (in Windows), in the User's Manual* for a description of the definitions that need be changed for increasing the buffer size. If the buffer is filled up, syntactic errors in the generated MSC/PR may be the result.

  – The use of message types that have names equal to MSC keywords may result in incorrect MSCs. There is no functionality for detection of this in the MSC Generator or TTCN Analyzer. The MSC Editor will detect the problem if trying to read MSC/PR form files where the problem is present. It is recommended to avoid ASP, PDU and CM names that are such names.

Examples of these names include “reset”, “RESET”, “stop”, “msc”, “TimeOut” etc.

- Concurrent TTCN configurations
  - The instances used in a generated MSC always include all the PCOs declared in the test suite. As a consequence, even though a PCO may not be defined to be used in a test case with a configuration, it will still be part of the generated MSC.
- The MSC Editor logging lacks some error control and may also result in excessive memory consumption by the MSC Editor in long test runs. Also, it slows down the tester significantly since it performs an extra 2 RPCs for each event that is logged. It is recommended to use this option only when developing or testing test scripts.
- It is possible to “reverse” the components used in the “composed mode” by editing the MscIn and MscOut functions such that they in the composed mode do not switch the “from” and “to” parameters. The reversed view may be conceptually more natural, though it may not be possible to process with some of the other tools of the SDL suite and TTCN suite family.
- There is an additional compile-time definition that is for debugging purposes: DEBUG_POST_COMMUNICATION. If set, there will be printouts to `stderr` of messages sent and received from the SDL suite and TTCN suite Postmaster. See also ***“POSTDEBUG” on page 494 in chapter 11, The PostMaster, in the User’s Manual***. If events seem to be missing in an MSC Editor log, these may be used to determine if there is an unhandled error condition.
- The MSC Editor has a built-in maximum number of simultaneously open diagrams. Since each test case generates a separate MSC diagram, this limit may eventually be reached when large numbers of test cases are run. The result of reaching this limit is not defined. As a consequence, it is recommended not to use MSC Editor logging when running large numbers of test cases.

### Simulator on UNIX

The entire list of SDL to C Compiler limitations applies also to the Simulator, see “TTCN ASN.1 BER Encoding/Decoding” on page 56.

- Monitoring of other test components can only be viewed together in the main screen log.
- Test suite variables and timers cannot be modified or viewed.
- The contents of channels cannot be modified.
- An entire test suite cannot be selected for execution. You have to do this manually, for example by selecting all test groups or all test cases.
- If you abort the execution of the TTCN-SDL Co-simulator when the SDL Simulator is running, further communication may be inhibited. Always restart the SDL Simulator when you abort the execution of the TTCN-SDL Co-simulator.

### TTCN-SDL Co-Simulator

- ASN.1 modules that have been joined/merged within the SDL system will not always work for SDL-TTCN Co-simulation, as the same join/merge functionality does not work the same way on the TTCN side. This means that name clashes will be solved differently on the SDL and the TTCN sides respectively.

### TTCN Exerciser

#### Timer Duration Limitations

- If a timer has a duration exceeding approximately 2000000000 (2^31-1) time units – as defined with the timer declaration – the condition may not be noticed, and it may result in erroneous behavior.
- If a timer duration exceeds the theoretical maximum period, the condition may not be noticed, and it may result in erroneous behavior.

#### Value Encoding and Decoding Limitations

In general, the value encoding and decoding has been designed to be a general mechanism for reading and writing almost any value, including ones that are not part of the type system of a generated test suite. This is for being able to input totally unexpected or irregular messages. The consequence of this is also that it may be easy to “break” test suites or even the kernel by providing wildly unexpected message contents. In order to get valuable results of the simulation runs, there may be a bit of effort needed to specify the messages. While doing that, the following notes and limitations may be helpful to avoid some pitfalls:

- There is limited error checking in the value encoding and decoding, which may result in ambiguous messages being encoded or decoded, or even possibly crashes if some invalid combinations are used.

  For example, if `asp1` is an ASP (base type is sequence), then these value encodings are likely to cause a runtime error or crash if fed into the kernel:

  ```
  asp1 24
  IA5String TRUE
  ```

  This error situation is not handled, since in general the kernel is designed to allow for arbitrary objects to be input. In the general case, the situation above will be created if assigning a refined type to an object with a different base type.

- For simple types, the actual type name is not part of the encoding. This particular choice of implementation was made to improve the readability of encoded values, but it may result in ambiguities when decoding the same message.

  An example: The value `VideotexString "Hello"` is encoded as `"Hello"` only, without the actual type name. Decoding of the same message will result in an object with the default string type IA5String. In order to specify the actual type, use a prefix of the actual type, such as `VideotexString "Hello"`.

- ENUMERATED values are not encoded or decoded by name, but rather as integer values.

- CHOICE values must be possible to distinguish by value, rather than by tag since tagging is not incorporated in the encodings and decodings of the TTCN Exerciser.

- The type OBJECT IDENTIFIER is not supported.

#### Concurrent TTCN

- There is a built-in limit in the TTCN Exerciser to how many concurrent test components may be simultaneously active. The limit is more than 10 but less than 25. The reason for this limit is the implementation of thread scheduling in which there is a built-in overhead which grows by the number of threads. There should be no problems running up to maybe 6–8 simultaneous test components, but more may seriously degrade the performance of the kernel. To work around this for real-time testing, do consider an alternative adaptation strategy at this time (using a deterministic and fair real-time system scheduler).

- Cancelling a test case from the context of a parallel test component may lead to unexpected behavior (such as hanging or livelocked PTC threads). It is recommended to always run a concurrent test case to a SNAPSHOT state before cancelling.
- Multiple readers/writers on a PCO or CP is not detected by the kernel, though it is a test case error according to TTCN.
- The result of terminating the MTC prior to all PTCs have terminated is undefined. The condition should be reported by the TTCN Exerciser, but the test result may be invalid. Terminating the MTC prior to all MTCs are terminated is illegal according to TTCN.
- There is no way to list the contents of a CP with the TTCN Exerciser.

#### TTCN Link

- Encoding/Decoding functionality cannot be used together with the TTCN Link kernel.
- ASN.1 modules that have been joined are not supported by TTCN Link.

#### Auto Link

- ASN.1 modules that have been joined are not supported by Auto Link.

## UML and MSC Restrictions

This section lists the major UML and MSC restrictions in the SDL suite, with respect to the OMG specification for UML, and the ITU-T recommendation for MSC.

References below are to sections in the OMG document “ad/99-06-08” (UML Notation Guide, version 1.3, June 1999), and the Z.120 recommendation.

### UML Class Diagrams

The following features of UML 1.3 Static Structure (Class) diagrams are not supported:

- Packages (section 3.13), but package references are allowed in class names
- Interfaces (section 3.28)
- Parametrized class, or template (section 3.29)
- Composite objects (section 3.39)
- Xor-associations (section 3.41.5)
- Navigability in associations (section 3.42.2)
- N-ary associations (section 3.46)
- Constraints in generalizations (section 3.49.2)
- Dependency, i.e. dashed arrows (section 3.50)

### UML Statechart Diagrams

The following features of UML 1.3 Statechart diagrams are not supported:

- Hidden decomposition indicator icon (section 3.76)
- Substate compartment with concurrent states (section 3.76)
- Transition times (section 3.78)
- Concurrent transitions (section 3.79)
- History state indicator (section 3.80)
- Stubbed transitions (section 3.80)
- Factored transition paths (section 3.81)
- Submachine states (section 3.82)
- Synch states (section 3.83)

### UML Implementation Diagrams

The notation used in the DP Editor diagrams is based on UML 1.3 Implementation diagrams. The major differences are the following:

- No dashed-arrow dependencies (on nodes and components)
- No interfaces (on components)
- No migration (of components from node to node or of objects from component to component)
- No graphical nesting (only composite aggregation) to show containment
- The thread concept has been added

### MSC and HMSC

- General ordering for events is not supported (Z.120 chapter 4.5).
- Gates are not supported (Z.120 chapter 4.4).
- Comment symbols in HMSC diagrams are not supported.
- Parallel frames in HMSC diagrams are not supported (Z.120 chapter 5.5).

## Tool Limitations

### All Editors

- Diagram names may not exceed 255 characters.

### MSC Editor

- The inline expressions always cover all instance axis. The reason for this limitation is that gates (and specifically inline gate interfaces) are not supported.
- The column-form of instances and coregions is not supported, only the line-form.
- A syntax error occurs when logging to MSC if an SDL page name contains two consecutive underscores, like “pg__name”. Error message should begin, ‘`MSCE: Service INSERTOBJECT failed.`’

### OM Editor

- Use *Browse&Edit Class* dialog with care if the symbol contains syntax errors.

  The syntax checker on attributes and operations have error recovery functionality, and can sometimes “recover” in an unwanted way. Use the Browse&Edit Class dialog for **editing** purposes only with special care if a syntax error is detected. In the drawing area, syntax errors are highlighted with a red underlining in the symbol’s corresponding text compartments. The browsing functionality of the Browse&Edit Class dialog can however be used without considerations even if syntax errors are detected.

### UML2SDL utility

- Currently there is no special recognition of the use of an SDL predefined type in the UML diagrams. Thus when having an attribute of type Pid in the UML Suite an (erroneous) newtype definition for the type Pid will be generated.

### Text Editor

- When you edit large portions of text containing many endpoints, updates can become slow in the Text Editor.

  As a work-around, turn off endpoint display if you encounter performance problems.

- On Windows, when using double-click to select words and then typing something to replace the selected word, a mismatch will occur between the text shown in the window and the actual text handled internally in the editor. The space shown after the new word is not recognized internally and is therefore missing when the text is saved and later reopened. The same happens if any text selected ends by the newline character. When replacing the selected characters with new text the ending newline will be gone when the text is saved.