Chapter

# 1 *Compatibility Notes*

**In this chapter you can read about compatibility issues between the Telelogic Tau 4.4 release and the 4.5 release, as well as between the 2.3 release and the 3.X releases. It also describes the compatibility issues for the supported languages in the SDL suite.**

**This chapter concentrates on issues of storage formats and language support. For the TTCN suite, this chapter describes how to upgrade an old TTCN suite database, and how to open a TTCN suite database with an old version of the TTCN suite.**

**For information regarding platform and product compatibility, see "Telelogic Tau Compatibility Matrix" on page 5 in chapter 1, *Platforms and Products, in the Installation Guide* and "Build and Certification Matrices" on page 6 in chapter 1, *Platforms and Products, in the Installation Guide*.**

# Licensing Compatibility

## FLEXlm License Keys

Telelogic Tau 4.5 is using the 8.0d version of FLEXlm.

The FLEXlm Product License Keys that come with the Telelogic Tau products in version 4.5 - 3.1X and 3.02 are not compatible with license keys used in versions earlier than 3.02 of ITEX and SDT.

This means that you cannot run earlier versions of ITEX and SDT (i.e. 3.0 and 3.01) using the FLEXlm FEATURES in the `license.dat` file in the Telelogic Tau 4.5 distribution.

If you want to be able to switch between versions 3.0/3.01 of ITEX and SDT and version 4.5 of Telelogic Tau, you must do as follows:

1. Save the existing (3.0 / 3.01) `license.dat` file before updating it with the new FEATURES that come in the 4.5 Product License Key from Telelogic Customer Support.
2. Assign the environment variable `LM_LICENSE_FILE` a value that designates the appropriate `license.dat` file before starting the tool set.

Telelogic Tau 3.2 and later versions use the same license feature names both on UNIX and in Windows. This was not true for SDT and ITEX of versions 3.1X and earlier.

You may experience some problems trying to get the Windows version 3.1X of ITEX and SDT to run together with later versions. If so, please contact Customer Support for help (see "How to Contact Customer Support" on page iv in this volume).

> **Note:**
>
> FLEXlm packages can only be handled by 3.1 and newer versions

## FLEXlm License Daemon

Telelogic Tau 4.5 comes with version 8.0d of the FLEXlm license daemon (`lmgrd`). If you are using FLEXlm licensing for other products than Telelogic Tau, or if you plan to use two different versions of Tele-

logic Tau in parallel, **you must use the latest version of the FLEXlm license daemon** that you have received (version 8.0d or later).

For more information, see the FLEXlm FAQ at
`http://www.globetrotter.com/support/index.html`

# File Name Compatibility

The following text concerns file name compatibility between UNIX and Windows using the Organizer in Telelogic Tau 3.3, 3.4, 3.5 and 3.6, as compared to earlier versions. Therefore, it does not apply to ITEX tools in Windows.

## Windows/UNIX Compatibility

The preference SDT*MixedPlatform (used in Telelogic Tau 3.2 and earlier versions) is no longer read by the Organizer. For systems using only relative paths, the Organizer now automatically converts file names between UNIX and Windows syntax, i.e. exchanging slashes and backslashes. The system file is always written in the format native to the platform used. Before version 3.3, the system file was always written in UNIX format.

For file names using absolute paths, the functionality of the Organizer's *PC Drives* dialog was extended in 3.3 to support general Windows paths instead of only drive letters, UNC paths (`\\hostname\path`), and quoted paths containing spaces. Files with absolute paths not found in the *PC Drives* dialog are presented in the Organizer as invalid diagram icons.

The special feature of always converting files to lowercase that the MixedPlatform preference used to provide must be enabled separately, using the preference SDT*UseLowerCaseInFileNames. This means that it is now possible to share systems between UNIX and Windows without forcing file names to lower case.

It is possible to make Telelogic Tau handle file names containing spaces. This feature has to be explicitly turned on as the possibility of using this feature may depend on the support for spaces in file names in other tools and executables integrated with Telelogic Tau. Support for spaces in file names is controlled by the preference SDT*AllowSpaceInFileNames.

### To Avoid Problems

If you are sharing systems between UNIX and Windows, and want to manipulate system files without using the Organizer, do not assume that paths are always written in UNIX format.

# Storage Format Compatibility

## System Files

**The file format for system files (`.sdt`) has changed in Telelogic Tau 4.5** as compared to Telelogic Tau 4.4. Telelogic Tau 4.5 can read older system files (3.0X - 4.4), but previous versions of Telelogic Tau cannot read the new 4.5 system files.

When saving older system files with the Telelogic Tau 4.5 Organizer, a conversion to the new 4.5 format is performed.

## SDL/GR and MSC/GR

The storage format for SDL diagrams did change in 4.1. This means that diagrams created with 4.1 cannot be opened in previous versions. SDL Suite 4.1 can still read all previous formats.

The storage format for SDL diagrams did change in 4.0. This means that diagrams created with 4.0 cannot be opened in previous versions.

The storage format for SDL and MSC diagrams did change in SDT 3.1X as compared to SDT 3.0X (and SDT 2.X). SDT 3.1X and later versions of Telelogic Tau can read older files, but previous versions of SDT cannot read the new format introduced in the 3.1X version.

In addition, **the storage format for MSC diagrams was changed in version 3.5**. Version 3.5 and 3.6 of the MSC Editor can handle the same formats as version 3.4 of the editor, but version 3.4 and earlier of the MSC Editor cannot read the new MSC format.

SDL and MSC diagrams stored in SDT 2.X and SDT 3.0X format may be opened as is into SDT 3.1X and later versions of the editors (using commands such as the SDL Editor's *Open*). To preserve the original diagram structure, you must however import them into the Organizer, by using the command *Import SDL* to build a diagram structure that may be stored on a system file.

When saving the opened SDT 2.X or 3.0X diagrams on file with SDT 3.1X and later versions of the editors, a conversion to the 3.1X format is performed. Conversion to the 3.1X format may also be performed automatically when importing the diagrams into the Organizer. The option *Save imported diagrams in SDT 3.X format* is used for this purpose.

**Caution!**

It is not possible to open files in the SDT 3.1X format in previous versions of SDT, e.g. SDT 3.0x or SDT 2.3, or to open MSC diagrams in the new 3.5 format in previous versions of the MSC Editor. Make sure you have backup copies of your diagrams.

## Generated SDL Simulators and Validators

Simulators and validators should be generated with the same version of the SDL suite as they are supposed to be executed in. They might not execute as expected when executed under a SDL Simulator Graphical User Interface of a different version. To re-generate, use the *Full Make* facility to make sure all C files are re-generated properly.

The SDL/GR trace in the SDL Editor requires the SDL diagrams to be saved in order to function properly. (The reference mechanism in SDT 3.X assumes the existence of information that is not stored in SDT 2.X files.)

## Deployment Diagrams

The 4.2 Deployment Diagram file format (*.sdp) was extended. The 4.2 Deployment Editor can read and convert .sdp files created with older versions of the Deployment Editor. However, older Deployment Editor versions can not read diagrams created with this new version.

In the 4.1 release, the Deployment Diagram file format (*.sdp) was extended. The Deployment Editor in the 4.1 release can read and convert .sdp files created with older versions of the Deployment Editor. However, older Deployment Editor versions can not read Deployment Diagrams created with the 4.1 version.

Buildscript generation is no longer supported by the Deployment Editor since the 4.1 release. Buildscripts are replaced by a feature which translates Deployment Diagrams into the Partitioning Diagram Models, which are used by the Targeting Expert. This makes it possible to utilize a partitioning configuration in a Deployment Diagram when building SDL systems using the Targeting Expert.

As buildscript generation is no longer supported, build settings on node and component level are ignored when an old Deployment Diagram is

converted into 4.1 format. Qualifier data on object level is automatically translated.

## The TTCN Suite Database Compatibility

This section describes the different versions of the ITEX database formats and how they are converted to each other. The following table shows the what formats are used in each ITEX version:

| ITEX version | Database version | Description |
|---|---|---|
| 2.0, 2.1 | 0 | Base version. |
| 2.2 | 1 | This version stores additional analyzer info. |
| 3.0x | 2 | Corrected order of DefaultsRef and Configuration in TestCase. |
| 3.1x | 3 | Support for Modular TTCN and encoding/decoding added. Changed to the compound format containing both the structure and the tables. |
| 3.2, 3.3, 3.4, 3.5, 3.6, 4.0 | 4 | Store some additional analyzer info and also restructured the content a little. |
| 4.1, 4.2, 4.3, 4.4 | 5 | Changed parse data structure. |
| 4.5 | 6 | Changed parse data structure. |

### The Database Upgrade

When the TTCN suite opens an old database, it checks if it is possible to convert the database to the current version via the TTCN suite TTCN-MP format. A conversion program for this old database version must be available. The name of this program has a suffix which indicates the version number of the database. For example the program which is used to convert a database of version 4 is called `mp-output-old4`.

This program is installed in the TTCN suite 4.5 release. It is used to convert databases of version 4 to MP format. The resulting MP file is then

converted to GR format as usual. This mechanism solves the problem of backward compatibility. The TTCN suite or a newer TTCN suite version can open an old version of the TTCN suite database.

### Forward Compatibility

A special problem occurs when an old ITEX version needs to open a new version of the ITEX database (forward compatibility). This problem is not relevant between database versions 0, 1 and 2 since the transfer format (MP) is the same. But the MP format which is used in 3.1 and later releases have extensions. Another problem occurs because the 3.1 and later databases are stored in a single file (compound format) with extension `.itex` instead of the `.itex/.itex-tables` combo of previous versions.

In the TTCN suite 4.5, the database contains documents of one of these types:

- Test suite
- Modular test suite
- Module

Conversion of a new database of type Module to an old 3.0x or earlier database is not possible. The only forward compatibility which is supported is for the databases of type Test Suite or Modular Test Suite.

This conversion mechanism is possible to retrofit into an older existing 3.X installation by copying the programs `mp-output-old6` and `convert-to-old6` from the 4.5 installation's binary directory for the architecture(s) installed and put these copied files into the corresponding directory in the old 3.X installation tree (these files are present in the current release for primarily this reason).

To make a 3.0x version able to convert from 4.5 databases it also needs the companion file with the extension `.itex-tables` to exist beside the 3.6 database file. Fortunately this file may be a dummy file containing nothing at all.

# Code Generation Compatibility

The default way of generating code for SDL operators was changed in Telelogic Tau 4.2. This method replaces generated functions for assignments, equal test, free and predefined operators in generators with generic functions.

This means that parameter passing has changed. Parameters are passed as addresses not as values, and data structures describing the data types are needed.

This method is backward incompatible with SDL systems that uses in-line C code. The function calls must be manually changed.

An old-style code generation for SDL operators option is available, enabling compatibility with SDL operator code generation of previous Tau releases.

# Compatibility with ITU SDL

The SDL language support in the SDL suite includes nearly all of SDL'92, as defined in the ITU-T Z.100 recommendation.

The main divergencies between SDL supported in the SDL suite and the Z.100 recommendation are:

- The concept of a generic system is only partially supported.
- Macro calls are only allowed within flowchart diagrams.
- The names of diagrams and diagram pages must conform to the lexical rules of SDL, except that spaces are not allowed.

More detailed limitations can be found in "SDL Restrictions" on page 33 and "SDL Editor" on page 35 in chapter 2, *Release Notes*.

The SDL Editor supports a few extensions to the SDL/GR notation:

- An *additional heading* symbol, intended for defining inheritance, specialization, formal parameters, signalsets, etc.
- An *operator reference* symbol, making operators visible in the SDL structure handled by the Organizer.

The SDL suite supports several new SDL concepts, some of which are defined in an addendum to the latest Z.100 Recommendation, known as *SDL-96*, and some others which are Telelogic-specific extensions to SDL. The extensions are supported in all relevant tools.

## SDL-96 Extensions

- **Remote procedures on channels and signal routes:** This extension allows names of remote procedures to be put in signal lists of

channels and signal routes. This has the main advantage that it enables processes to call remote procedures implemented in the environment of the SDL system, and also the other way around; remote procedures can be called from the environment of the SDL system.

- **External procedures:** External procedures are called like normal SDL procedures, but they are implemented in the target language (C). This makes it easier to use C functions in SDL.
- **External operators:** External operators are called like normal SDL operators, but they are implemented in the target language (C). This makes it easier to use C functions in SDL.

Other SDL-96 features are not supported in the SDL suite.

## Telelogic-Specific Extensions

The following extensions to SDL are Telelogic-specific. A detailed description of these extensions can be found in chapter 2, *Data Types* and chapter 3, *Using SDL Extensions, in the SDL Suite Methodology Guidelines*.

- **Choice:** With this construct you can define types with values where only one of a number of components can be present. Choice types replace the `#UNION` code generator directive (although `#UNION` is still supported for backward compatibility reasons).
- **Optional/default components in structs:** It is now possible to specify that fields of struct types are optional, or have a default value. This feature is useful in the design and implementation of protocols.
- **Size constraints:** With this construct it is possible to define types that should have a limited length.
- **Bit fields:** Integer fields in struct types can be specified as bit fields, very similar to C. This is useful when reusing C data types that contains bit fields in SDL.
- **Z.105 types and generators:** The following new predefined data types and generators from Z.105 have been added: Bit, Bit_String, Octet, Octet_String, Object_Identifier, IA5String, NumericString, PrintableString, VisibleString, Null, and the Bag generator.

- **Powerset operators:** Powerset has been extended with Length and Take operators, which make it possible to iterate over powersets and thereby make powersets more useful.
- **Pointers:** There are syntactic extensions that make it easier to work with pointers in SDL. In order to use pointers, the Ref generator from the package `ctypes` should be used.
- **Own and ORef:** The generators Own and ORef support safe pointers, making it possible to speed up applications by sending pointer values instead of copying data during signal sending. Allocation and deallocation of memory is handled automatically.
- **Algorithms in SDL:** A number of new constructs have been added to allow algorithms to be expressed within a Task symbol, e.g. if-then-else, loops and decisions.
- **Extended operator:** Added support for operators without parameter, operators without result, and in/out parameters to operators (both operator signatures and diagrams). As an alternative to an assignment statement an operator application statement may invoke a non value returning operator.
- External variables supported.
- **Latin One:** ISO-Latin-1 supported in predefined sort character.
- **I18n:** Predefined sort charstring supports multi-byte character literals.
- **Names:** Underscore may be used without restrictions in names.
- A make operator has been added to the predefined items String, Powerset, Bag, and Object_identifier. The operator takes a variable number of arguments (0 and up). It is invoked by the "(. .)" notation. The signatures for the operators are:

  ```
  make! : * Itemsort -> String;
  make! : * Itemsort -> Powerset;
  make! : * Itemsort -> Bag;
  make! : * Natural -> Object_identifier;
  ```

  The * above indicate a variable argument (0 or more occurrences allowed).
- New operator in predefined type Object_identifier:

  ```
  append    : in/out Object_identifier, Natural;
  ```

## SDL-2000 Extensions

- Binary and hexadecimal literals for Integer, Bit_string, Octet, and Octet_string sorts. Examples ‘’b ‘1001’B ‘’H ‘Fe’h.
- Mix of block and process:
  - System may contain process (without intervening block).
  - Block may contain block (without intervening substructure).
  - Channels may be connected to processes.

# Compatibility with ITU MSC

The MSC language support in the SDL suite includes full support for *basic MSCs* (MSC'93), and also parts of MSC'96, as defined in the ITU-T Z.120 recommendation.

The key points for the MSC'96 support are:

- Roadmaps, or High-level Message Sequence Charts (HMSC), are supported in the HMSC Editor with the exception of the comment symbol and the parallel frame symbol.
- MSC diagrams support the reference symbol with MSC reference expressions, allowing references to other (H)MSC diagrams.
- MSC diagrams support qualifiers.
- MSC diagrams support inline expressions, allowing several scenarios to be expressed in the same diagram. Since the MSC Editor does not support gates, the inline expressions always cover all instances.
- Separate timer symbols are supported in MSC diagrams.
- MSC diagrams support lost and found messages (black and white holes).

Limitations to the MSC support can be found in "UML and MSC Restrictions" on page 62 and "MSC Editor" on page 64 in chapter 2, *Release Notes*.

The MSC Editor also supports a few extensions to Z.120:

- Messages and timers may be assigned additional statuses that map the Z.100 recommendation, allowing to express a graphical trace in a more intuitive way.
- Messages are allowed to be *sent to self*, where the out event and the in event resides on the same instance (Z.120 does neither explicitly allow or forbid this).

## MSC'96 Reserved Words

The following words are reserved in MSC'96:

**action, all, alt, as, before, begin, block, by, comment, concurrent, condition, connect, create, decomposed, empty, end, endconcurrent, endmsc, endexpr, env, exc, expr, external, found, from, gate, in, inf, inline, inst, instance, loop, lost, msc, mscdocument, msg, opt, order, par, process, reference, related, reset, service, seq, set, shared, stop, subst, system, text, timeout, tim, to, via**

The reserved words may not be used as names/identifiers even if the MSC Editor does not support the corresponding language construct; this to avoid compatibility problems either when the editors support more of the MSC'96 language or if the MSC/PR is to be exported to other tools with a different subset of the language supported.

## MSC'93 Reserved Words

The following words are also treated as reserved words in the (H)MSC Editors but are mapped to (corresponding) MSC'96 reserved words.

| MCS'93 | MSC'96 |
|---|---|
| instancehead | instance |

This mapping is done to give old scripts and MSC/PR generators a chance to work.

## Unsupported Reserved words

The following reserved words are not supported by the (H)MSC Editors (i.e. they are treated as reserved words with respect to names/identifiers but the Editors have no support for the constructs that can be created with these words).

**before, begin, external, found, gate, lost, order, related, via**

In addition, the Validator does not support the following reserved words:

**subst, by, msg**

## Parameter Lists

Note that MSC/PR import in Telelogic Tau allows empty parameter lists and an empty duration name when setting a timer, due to backward compatibility. This is not in accordance with either MSC'93 or MSC'96.

# Chapter 1 Compatibility Notes