



SDLTrados²⁰⁰⁶

S-Tagger User Guide

COPYRIGHT

S-TAGGER USER GUIDE

© Copyright 2006, SDL International.

All rights reserved. No part of this documentation may be duplicated in whole or in part or reproduced in any form without the express written permission of SDL International.

All trademarks are the property of their respective owners. The names of other companies and products mentioned herein may be the trademarks of their respective owners. Unless stated to the contrary, no association with any other company or product is intended or should be inferred.

Although SDL takes all reasonable measures to provide accurate and comprehensive information about the product, this guide is provided “as is” without warranty, either express or implied.

To access the latest documentation for this and other products, see the downloads page at www.sdl.com or www.translationzone.com.

This guide accompanies SDL Trados 2006

February 2006

TABLE OF CONTENTS

OVERVIEW

Chapter 1 Introducing S-Tagger

About this Guide	1-2
Related Documentation	1-2
What are the S-Taggers?	1-3
Key Terms and Definitions	1-4
General Software Information	1-6
Version Compatibility	1-6
Demo Mode.	1-7

Chapter 2 S-Tagger Basics

What is an STF File?	2-2
Working with Tagged Text Formats	2-2
STF File Contents	2-2
The Ancillary File.	2-3
Other S-Tagger Files	2-5
The ORG File	2-5
The CMP File	2-5
What is an S-Tag?	2-6
S-Tag Types	2-6
S-Tag Components	2-7
Project Structure	2-9
Creating a Folder Structure.	2-9
S-Tagger Workflow: An Overview.	2-10
Workflow Terminology	2-11
Choosing an Editor	2-13
TagEditor Workflow (TTX and RTF)	2-13
Word Workflow (RTF Only).	2-17

S-TAGGER FOR FRAMEMAKER

Chapter 3 Preparation and MIF to STF Conversion

Preparing FrameMaker Files	3-2
Preliminary Guidelines	3-2
Setting up the Project Structure	3-3
Required Tasks	3-5
Advised Tasks	3-6
Special Feature Tasks	3-9
Saving FM Files to MIF Format	3-12
Defining the Conversion Settings	3-13
Launching S-Tagger for FrameMaker	3-13
The Settings Tab	3-14
The Paths Tab	3-23
Converting MIF to STF	3-24
Testing the New STF Files	3-26

Chapter 4 STF File Translation

Translating STF files	4-2
General Translation Issues	4-3
Working with S-Tags	4-4
General S-Tag Guidelines	4-4
Manipulating Internal Tags	4-7
Manipulating External Tags	4-9
Asian and Eastern European Languages	4-14
Character Set Tags	4-14
Unsupported Characters	4-15
Index Marker Sort Levels	4-16

Chapter 5 S-Tag Verification and Clean Up

Verifying Translated STF Files	5-2
Verification Guidelines	5-2
About Verification	5-2
Checking your Conversion Settings	5-3
Verifier Report	5-4
Verifying the S-Tags	5-6
File Errors	5-8
S-Tag Verification Messages	5-9
Updating the Translation Memory	5-19

Chapter 6 STF to MIF Conversion and Post-Production

Converting STF to MIF	6-2
Correcting STF to MIF Conversion Errors	6-4
General Post-production	6-6
Opening the New MIF File	6-6
Checking the New MIF File	6-8
Saving the MIF File to FM Format	6-9
Asian and East European Language	
Post-production	6-10
Character Set Tags	6-10
Unsupported Characters	6-11
Using Font Mapper for FrameMaker	6-11

S-TAGGER FOR INTERLEAF

Chapter 7 Preparation and IASCII to STF Conversion

Preparing Interleaf Files	7-2
Preliminary Guidelines	7-2
Setting up the Project Structure	7-4
Required Tasks	7-6
Advised Tasks	7-7
Special Feature Tasks	7-11
Saving ILDOC Files to IASCII Format	7-14

Defining the Conversion Settings	7-15
Launching S-Tagger for Interleaf	7-15
The Settings Tab	7-16
The Paths Tab	7-23
Converting IASCII to STF	7-24
Testing the New STF Files	7-26

Chapter 8 STF File Translation

Translating STF Files	8-2
General Translation Issues	8-3
Working with S-Tags	8-4
General S-Tag Guidelines	8-4
Manipulating Internal Tags	8-7
Manipulating External Tags	8-9
Translating to Japanese	8-14
Character Set Tags	8-14
Reserved Character Tag	8-15
Unsupported Characters	8-15
String Length	8-15
Index Marker Sort Levels	8-16

Chapter 9 S-Tag Verification and Clean Up

Verifying Translated STF Files	9-2
Verification Guidelines	9-2
About Verification	9-2
Checking your Conversion Settings	9-3
Verifier Report	9-4
Verifying the S-Tags	9-6
File Errors	9-8
S-Tag Verification Messages	9-9
Updating the Translation Memory	9-18

Chapter 10 STF to IASCI Conversion and Post-Production

Converting STF to IASCI	10-2
Correcting STF to IASCI Conversion Errors	10-4
General Post-production	10-6
Setting Up the Book Structure for the Interleaf Files	10-6
Opening the New IASCI File	10-7
Checking the New IASCI File	10-8
Saving the IASCI File to ILDOC Format	10-9
Japanese Language Post-production	10-10
Character Set Tags	10-10
Unsupported Characters Alert	10-11
Language-specific Features	10-11
NO TAG References	10-11
Using Font Mapper for Interleaf	10-11

APPENDICES

Appendix A FrameMaker Tags and Examples

S-Tags in the STF Files	A-2
Anchored Frames	A-3
Character Formatting	A-5
Conditional Text	A-7
Cross-references	A-8
Footnotes	A-10
FrameMaker+SGML Elements	A-12
Markers	A-13
Paragraph Styles	A-17
Paragraph Numbering Formats	A-19
Tables	A-21
Unanchored Frames	A-25
Variables	A-25
S-Tags in the STF Ancillary File	A-26
Cross-reference Formats	A-27
Variable Formats	A-27
Paragraph Numbering Formats	A-27
Element Prefix/Suffix Formats	A-27
Footnote Prefix/Suffix Formats	A-28
Master Pages	A-28


List of S-Tags	A-29
----------------------	------

Appendix B Interleaf Tags and Examples

S-Tags in the STF Files	B-2
Anchored Frames	B-3
Attributes and Control Expressions	B-6
Cross-references	B-7
Footnotes	B-9
Inline Components	B-10
Markers	B-12
Paragraph Styles (Components)	B-15
Prefixes	B-17
Read-only Components	B-20
Shared Components	B-20
Shared Frames	B-22
Shared Named Objects	B-24
Special and Reserved Characters	B-25
Tables	B-25
Unknown Inline Components	B-28
S-Tags in the STF Ancillary File	B-29
Page Header/Footer Formats	B-30
Shared Frame Formats	B-30
Text in Shared Named Object Formats	B-30
Variable Formats	B-31
Prefix Formats	B-31
Autonumber Stream Formats	B-31
Page Number Stream Formats	B-31
List of S-Tags	B-32

Glossary

Index



SECTION 1: OVERVIEW

INTRODUCING S-TAGGER

Section 1 provides a product overview, applicable to both S-Tagger for FrameMaker and S-Tagger for Interleaf.

This particular chapter contains general application information.

Chapter sections include:

- ❑ a basic introduction to the S-Taggers
- ❑ information on new S-Tagger features
- ❑ general S-Tagger software information.

Chapter

1

ABOUT THIS GUIDE

This guide is aimed at users working with S-Tagger for FrameMaker or S-Tagger for Interleaf. The first section contains general information on the S-Taggers, while the second and third sections outline the different procedures used specifically for S-Tagger for FrameMaker or S-Tagger for Interleaf. The guide also contains information relating to the Font Mapper for FrameMaker and the Font Mapper for Interleaf.

Audience Profile

The S-Taggers are conversion tools, used to prepare FrameMaker MIF and Interleaf IASCII files for translation with Translator's Workbench and TagEditor or Word. S-Tagger users have a good knowledge of FrameMaker and/or Interleaf, which they need to prepare files for translation.

Related Documentation

You can also refer to the following guides for further information:

- ❑ *Translator's Workbench User Guide* – contains information on Translator's Workbench and the two editing environments (Word and TagEditor) that can be used in conjunction with the S-Taggers during the translation process. This guide also contains system requirements and installation information for the S-Taggers and the Font Mappers.
- ❑ *File Formats Reference Guide* – contains general information on using the S-Taggers to process FrameMaker and Interleaf files in an SDL Trados-based translation workflow.

S-Tagger for FrameMaker, S-Tagger for Interleaf, Font Mapper for FrameMaker and Font Mapper for Interleaf are all shipped with context-sensitive help systems.

Sample Files

If you are using this software for the first time, we recommend that you use the sample files to work through any examples on your own computer.

The default installation paths for the sample files are:

- ❑ `C:\Program Files\SDL International\T2006[FL]\Samples\STaggerF for FrameMaker`
- ❑ `C:\Program Files\SDL International\T2006[FL]\Samples\STaggerI for Interleaf.`

We recommend that you copy these files from the default installation location and paste them into a test translation project folder. It is important that you copy these files and do not overwrite the originals. If you encounter any problems, you can always start again by recopying the original sample files. You may create your own folder at whichever location best suits your work practices.

WHAT ARE THE S-TAGGERS?

The S-Taggers are a conversion utility for FrameMaker and Interleaf files. S-Tagger technology converts the FrameMaker MIF or Interleaf IASCII files into STF, a tagged text format that is compatible with SDL Trados. There are two separate S-Tagger components, S-Tagger for FrameMaker and S-Tagger for Interleaf.

All formatting information in the original source file is presented as translation-friendly tags, known as S-Tags. The STF file is comprised of these tags and the translatable text. You can translate files using TagEditor, a specialized editing interface for tagged text formats, or Microsoft Word.

After translation, you can use the tag verification features in TagEditor and in the S-Taggers to verify the tag content of your target STF files. Successful verification guarantees successful conversion to MIF/IASCII.

When your STF files have been verified, you can convert the STF files back to their original file format.

The S-Taggers are used in conjunction with the Font Mappers, which specify the fonts that you wish to map quickly and easily. This is necessary when working with Asian languages such as Japanese.

Key Terms and Definitions

The table below lists key terms that are used in the *S-Tagger User Guide*. It is important to familiarise yourself with these before proceeding.

Term	Definition
STF file	File produced when you convert a MIF/IASCII file to STF using the applicable S-Tagger. Formatting from a FrameMaker/Interleaf file is represented by brief coded statements (S-Tags). STF is a tagged text format that is compatible with Translator's Workbench.
TRADOS tag	XML-based file format for representing both DTP tagged text output and bilingual data. TRADOS tag files (*. ttx) provide a standard method for processing XML, HTML, SGML and DTP file formats.
Ancillary file	RTF, TXT or TTX file that is generated during the conversion of MIF/IASCII files to STF. It contains master property information such as header and footer text.
ORG file	File that contains structural information for the conversion of a given STF file back to the original format. A separate ORG file (*. org) is produced for each source file that is converted to STF. This file is then used to convert the translated STF file back to its original format.
CMP file	File generated when the S-Taggers verify the S-Tags in the target STF files. The CMP file (*. cmp) includes details of all tag errors, alerts, and warnings. The CMP file is also known as the verifier report.
S-Tags	Coded statements that the S-Taggers use to represent document formatting. These tags can be either external or internal.
External tags	Tags that represent formatting information about the source files that were converted. The information contained in the external tags is primarily structural. The external tags should not be modified, moved, or deleted during translation.
Internal tags	Tags that represent formatting information about the source files that were converted. The information contained in the internal tags describes simple formatting properties in the source documents like bolding or punctuation. The positioning of these tags will usually require modification during the translation process.
Original file format	Refers to the original FrameMaker (MIF) and Interleaf (IASCII) file format.

Term	Definition
Conversion	Refers to the conversion of MIF and IASCII files to STF format. It also refers to the conversion of the STF files back to their original file format after translation.
Verification	Refers to the comparison process where the tags in the target STF file are compared and verified against the tags in the source STF file. During and after translation, translators should verify that they have placed all the tags in the translated files in the correct sequence and position.

**FOR MORE INFORMATION**

For more information on any of the entries in the table above, see Chapter 2.

GENERAL SOFTWARE INFORMATION

This section contains information on version compatibility, copy protection and running the software in demo mode. System requirements and information on installing the S-Taggers is outlined in the *Translator's Workbench User Guide*.

Version Compatibility

This release of the S-Taggers is incremented to version 5.6. You can always refer to the S-Tagger **About** tab to check your software version number.

FrameMaker Compatibility

S-Tagger for FrameMaker version 5.6 does not work on files created using S-Tagger for FrameMaker version 1 or version 2. If you have any files created using version 1 or 2 of the software which you need to convert back to MIF or to verify, you must continue to use that version for these files. Translation memories containing S-Tags are backward compatible.

S-Tagger for FrameMaker supports MIF files generated by FrameMaker version 4.0 or later (this includes MIF files generated by FrameMaker 7), FrameBuilder and FrameMaker+SGML 5.0 or later.

MIF files generated by the use of customised tools or applications are not supported by S-Tagger for FrameMaker. Manually built or edited MIF files are not supported by S-Tagger for FrameMaker.

Interleaf Compatibility

S-Tagger for Interleaf version 5.6 supports IASCII and catalog files saved from Interleaf 5.2 or later. This release supports Interleaf 7, included as part of the QuickSilver XML application suite.

S-Tagger for Interleaf does not support IASCII files generated by any other applications or other versions of Interleaf. If you are using an earlier version of Interleaf, we recommend that you upgrade to Interleaf 7.

IASCII files generated by the use of customised tools or applications are not supported by S-Tagger for Interleaf. Manually built or edited IASCII files are also not supported.

Demo Mode

When the S-Taggers are running in demo mode the following file size restrictions apply for the conversion of files in original format:

S-Tagger for FrameMaker

- MIF files: 300 KB
- RTF files: 50 KB
- TXT files: 25 KB
- TTX: 50 KB

S-Tagger for Interleaf

- IASCII files: 300 KB
- RTF files: 50 KB
- TXT files: 25 KB
- TTX: 50 KB

These limitations on file size allow you to process only a few pages of documentation.

Font Mappers

Font Mapper for FrameMaker and Font Mapper for Interleaf can process MIF/IASCII files of up to 300 KB in size.



S-TAGGER BASICS

This chapter describes the basic concepts and features of S-Tagger for FrameMaker and S-Tagger for Interleaf.

Chapter sections include:

- ❑ an introduction to STF files
- ❑ information on the other files used with S-Tagger
- ❑ an introduction to S-Tags
- ❑ guidelines for your project folder structure
- ❑ an overview of the S-Tagger workflows.

Chapter

2

WHAT IS AN STF FILE?

An STF file is a tagged text file that contains translatable text from a FrameMaker or Interleaf file. STF is a text-only markup language which replaces the formatting from FrameMaker or Interleaf source files with brief coded statements (tags).

When S-Tagger converts the MIF/IASCII files to STF files, they can be saved as TRADOSTag (TTX), Workbench RTF, or TXT files depending on your conversion settings.

NOTE

The concepts outlined in this chapter are relevant to S-Tagger for FrameMaker and S-Tagger for Interleaf. All references to ‘S-Tagger’ throughout this chapter apply to both applications.

Working with Tagged Text Formats

After conversion with S-Tagger, there are two editing environments that can be used with Translator’s Workbench to carry out interactive translation of the STF file:

- TagEditor
- Word.

TagEditor supports translation of STF files in TRADOSTag and Workbench RTF format; Word supports translation of STF files in RTF format only.

When working with STF files, you translate the text as you would normally, then manipulate the necessary tags so that the translated text is formatted in the best way to suit your translation. Strict rules apply to moving, adding or deleting tags. You translate the text, never the tags. For more information on S-Tags, see “What is an S-Tag?” on page 2-6.

FOR MORE INFORMATION

TagEditor provides advanced tag display, protection, and verification features for tagged formats. For more information on using TagEditor or Word with Translator’s Workbench, see the *Translator’s Workbench User Guide*.

STF File Contents

Together with translatable text, the STF file contains a series of tags known as S-Tags. The S-Tags contain information about formatting and structure and can be either internal or external.

Graphics are not visible in the STF files during translation. Instead, the frame containing the graphic is represented by tags.

Internal Tags

An internal tag represents simple formatting information which often changes with translation. A font change is represented by an internal tag. Some punctuation and special characters are also represented by internal tags. You will often have to move these tags around to suit the translation. For complete information on which tags may be moved, added or deleted, see “Manipulating Internal Tags” on page 4-7.

External Tags

Paragraph styles and graphics are represented by external tags. You rarely need to change the paragraph styles or move the kind of graphic represented by an external tag. For more information on dealing with external tags during translation, see “Manipulating External Tags” on page 4-9.

The Ancillary File

The ancillary file is a special kind of STF file that can be created by S-Tagger during the conversion process, depending on your conversion settings. This file is always called `ancillary.rtf` or `ancillary.rtf.txt` or `ancillary.txt`, depending on the file format that you are working with.

Only one ancillary file is created for every batch of files that is converted. Instead of translating master property text again and again in multiple target files, the text can be extracted into the ancillary file and simply translated once. Text in the ancillary file is referred to as ‘ancillary text’.

After translation, as the files are being converted back to original file format, the translated text from the ancillary file is automatically re-inserted into the new file. The ancillary file must be present in the same folder as the STF files being converted back to original file format.

FrameMaker Ancillary Files

For FrameMaker sources, the ancillary file includes text found in:

- master pages
- paragraph numbering formats
- variables
- cross-reference formats
- element prefixes and suffixes (FrameMaker+SGML)
- footnote prefixes and suffixes.

If you have chosen to create an ancillary file, you can only translate such text within that file. The text within variables, for example, is visible in the other STF files, but is presented in tags which cannot be translated.

Interleaf Ancillary Files

For Interleaf sources, the ancillary file includes text found in:

- headers and footers
- prefixes
- shared components
- autonumber streams
- page number streams
- shared frames or shared named objects.

In addition to these items, any text from the categories above which is contained in a CAT (catalog reference) file and which is used in one or more of the files is included in the ancillary file.

When translation is complete and the STF files are converted back to Interleaf, the text from the translated ancillary file is automatically re-inserted into the new IASCII files, as long as the translated ancillary file and the CAT (catalog reference) files are present. For more information on CAT files, see “The CAT File” on page 7-5.

OTHER S-TAGGER FILES

The following section describes the ORG and CMP files that are used in S-Tagger translation projects.

The ORG File

For each MIF/IASCII file that is converted to STF, S-Tagger produces a second file called the ORG file. The ORG file for a given STF file uses the same file name but has an `*.org` extension. S-Tagger uses this file as the basis for converting the STF file back to its original format.

The S-Taggers allow you to choose whether or not to keep the original file name extension as part of the STF file name. If you choose this option, the file extension of the ORG file changes also. S-Tagger for FrameMaker produces a `*.mif.org` file. S-Tagger for Interleaf produces an `*.ildoc.org` or a `*.doc.org` file depending on the version of Interleaf you are working with. For more information on STF file naming conventions in S-Tagger for FrameMaker, see “STF File Name Extensions” on page 3-19 or for more information on S-Tagger for Interleaf, see “STF File Name Extensions” on page 7-21.



WARNING

The ORG file must never be deleted until after the project is complete and the files have been converted back to the original file format. You will not be able to convert STF files back to MIF or IASCII without the relevant ORG file.

The CMP File

Each time you use the verification feature, S-Tagger generates a verifier report, also known as the CMP file.

The CMP (or compare) file lists the number and type of errors, alerts or warnings found in the file being verified. The CMP file has the extension `*.cmp` and is always placed in the same folder as the translated files which are being verified. The CMP file always starts with a date and time stamp; it then shows you the name and path of the source file which was used as the basis for comparison, together with the name and path of the file which was being verified.

The S-Taggers allow you to choose whether or not to keep the original file name extension as part of the STF file name. If you choose this option, the file extension of the CMP file changes also. S-Tagger for FrameMaker produces a `*.mif.cmp` file. S-Tagger for Interleaf produces an `*.ildoc.cmp` or a `*.doc.cmp` file depending on the version of Interleaf you are working with. For more information on STF file naming conventions in S-Tagger for FrameMaker, see “STF File Name Extensions” on page 3-19 or for more information on S-Tagger for Interleaf, see “STF File Name Extensions” on page 7-21.

→ NOTE

- ❑ For more information on the verification process and the CMP file, see “Verifying the S-Tags” on page 5-6.


WHAT IS AN S-TAG?

An S-Tag represents information about formatting in the original FrameMaker or Interleaf file. A complete tag reference chart is available in Appendix A and B and in both the S-Tagger Helps.

S-Tag Types

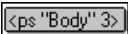
The following section describes the various tag types that are displayed in the STF files when opened in TagEditor (or Word).

→ NOTE


Use the **Complete** tag text setting to display the tag text fully in TagEditor. To set the tag text setting to **Complete**, simply click this icon in the Standard toolbar: 

Tags are classified as external or internal, depending on their function:

- ❑ external tags – these tags represent structural information. The text within these tags should remain as it is during translation because modifying it may cause errors when attempting to convert back to your original file format.

For example, `<ps "xxx" ?>` which indicates a paragraph style. In TagEditor, external tags are surrounded with a black border by default. For example, the paragraph style tag could be displayed as .



- ❑ internal tags – these tags can represent formatting information, surround in-document markers and appear inside the text to be translated. Most internal tags can be moved around within the segment to suit the translation. Depending on the file format, some internal tags can be added or deleted as required. TagEditor classifies unknown tags as internal and surrounds all internal tags with a red border by default.

For example, `<:t>` which indicates a tab. In TagEditor, internal tags are surrounded by a red border by default. For example, the tab tag could be displayed as .


Most tags can be classified as opening, closing or stand-alone tags, depending on how they work:

- ❑ opening and closing tags – these tags work in pairs to invoke and revoke an instruction. The opening tag indicates the start of a character format or structural element such as a heading.

The closing tag marks the end of a formatting or structural element. Text and other tag pairs may occur in between the opening and closing tags for a particular instruction.




For example, `<:b>` and `<:/b>` which indicate the scope of bold formatting. In TagEditor, these tag pairs would be displayed as  and .

- stand-alone tags – these tags function independently, which means that they do not need to be part of an opening and closing pair to function correctly.




For example, `<:hs>` which indicates a hard (non-breaking) space. In TagEditor, this tag would be displayed as .

During translation, TagEditor and Word insert tags to mark source and target segments and to provide information about translation memory match values.

- translation unit tags – these delimiting tags identify source segments, match value and target segment respectively, of a regular translation unit.

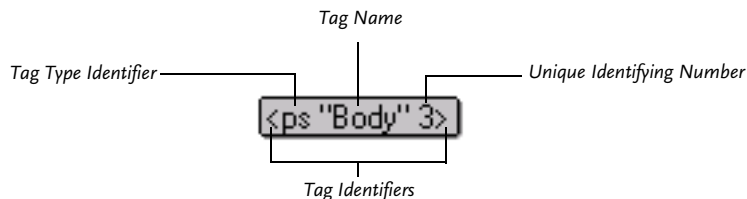
In Word, the delimiting tags are displayed as `{0>, < }100{>`, and `<0}`. In TagEditor, they are displayed as ,  and .

- Context TM unit tags – these delimiting tags identify the source and target segments of an Context TMunit. Because they are taken from previously reviewed bilingual documents (`*.bif` or `*.txt`), Context TM units are considered to be perfect matches from a segment and context point of view. For this reason, no match value is given.

In TagEditor, these tags are displayed as ,  and . Note that XTranslation units are not compatible with the Word editing environment. For more information, see “Context TM” on page 2-15.

S-Tag Components

An S-Tag consists of several components as illustrated below:



Tag Identifiers

The first part is the tag identifier. This consists of braces (< and >) which surround the tag. If the opening brace (<) is immediately followed by a colon (:), the tag is identified as an internal tag. If it is not, it is identified as an external tag.

TagEditor also uses colour formatting to distinguish internal and external tags. Internal tags are surrounded by a red border; external tags are surrounded by a black border.

Tag Type Identifiers

The second part of the tag is the tag type identifier. This is usually a single or double character, but in some cases there are more than two characters used. The tag type identifier characters immediately follow the opening brace or brace and colon. For example, a paragraph style tag type is identified by the characters *ps*; a character style tag type is represented by the characters *cs*.

Unique Identifying Number

The third part of the tag is its unique identifying number. Most tags are assigned a unique identifying number. This is because although you can apply a style with the same name to two separate paragraphs, the actual properties of that style may be completely, or even just slightly, different. To preserve all the formatting that has been applied, all tags which represent formatting features with any sort of properties applied to them (which is all tags except for symbols and special characters) have a unique identifying number.

You can ignore this number during translation as you will be using tags which already exist in the document. The majority of tags that you add will not have unique identifying numbers. Do not make up your own identifying number, as this will create an invalid tag.

Tag Name or Text

Some items, such as paragraph styles, have names. If an item has a name, this name will be inserted in between the tag type identifier and the unique identifying number. It always appears within quotes (“and”). Some items, such as variables, cross-references and prefixes, may contain text. When an item contains text, the text appears in quotes in between the tag type identifier and the unique identifying number. If an item has both a name and some text, only the text is displayed.



NOTE

Throughout the remainder of the *S-Tagger User Guide*, tags will generally be presented as text.

PROJECT STRUCTURE

It is important to create a set project structure before you begin the conversion process. An established project structure will ensure that a clean copy of the source STF file is kept for verification purposes later on in the process; it keeps the files used for translation separate from the vital source files.

S-Tagger never changes the name of the files it is working on. The converted STF file has the same name as the source MIF or IASCII file; only the extension is different. The STF file has an `*.rtf` or an `*.rtf.txt`, or `*.txt` extension, depending on the output file format that you select during conversion. If you do not set up a good project structure, you run the risk of overwriting, moving or deleting vital file information.

S-Tagger creates three new files for each MIF/IASCII file that is converted:

- ❑ a source STF file – to be used during and/or at the end of the translation process for verification
- ❑ an ORG file – to be used as the basis for converting the translated STF file back to its original format
- ❑ a target copy of the STF file – to be used as the file for translation.

When beginning to work with S-Tagger, you need to specify three file locations for:

- ❑ the MIF/IASCII files
- ❑ the source STF files (S-Tagger will also automatically place the ORG files in this folder)
- ❑ the target STF files.

These paths are all specified on the **Paths** tab in S-Tagger. For more information, see “The Paths Tab” on page 3-23.



NOTE

For more information on STF file naming conventions for S-Tagger for FrameMaker, see “STF File Name Extensions” on page 3-19, and for more information for S-Tagger for Interleaf, see “STF File Name Extensions” on page 7-21.

Creating a Folder Structure

We recommend that within your translation project folder, you create:

- ❑ a *Source* folder for the source STF and ORG files – the files that will be used for comparison when you are finished translating

- ❑ a *Target* folder for the target STF files – the files that will be translated
- ❑ a *MIF/IASCII* folder for the original format files – the files that will be converted to STF.

The target files are opened and edited in TagEditor (or Word) during the interactive translation process, but the source files must remain unchanged for comparison and verification purposes.

NOTE

- ❑ The complete folder structure must be set up for each target language. You cannot create a folder structure with subfolders for each target language as S-Tagger cannot process such folder structures.
- ❑ Because the file names do not change during translation, it is important to keep files apart in separate source and target folders.

S-TAGGER WORKFLOW: AN OVERVIEW

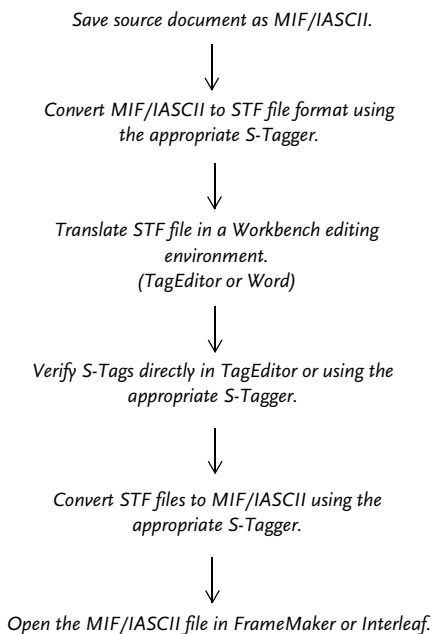
The translation processes for FrameMaker and Interleaf files have multiple steps. This section provides you with a good overview of the entire translation project workflow, from file preparation to saving the translated files back to FM/ILDOC file format. This section provides an overview of the TagEditor and Word workflows, using TRADOSTag (TTX) and RTF respectively as the STF file format.

ANSI Text Files

ANSI Text (plain text) files have the extension `*.txt` and can be opened in any text editor that supports ANSI text. The tags have no special formatting. If you are going to translate ANSI text files and the files are very large, you may not be able to edit them in Notepad or other small text editors. Always open and save the files as text only, without formatting. It is very important that you never open or save the files as MS-DOS text or in a DOS-based editor.

We recommend using the TRADOSTag workflow rather than TXT, because this allows you to use the batch tools in Translator's Workbench (analysis, pre-translation) to scope translation work and TagEditor as the editing environment (WYSIWYG display, tag protection and verification).

The diagram below illustrates a high-level overview of the translation process:



Workflow Terminology

It is important that you familiarise yourself with the terms that are used to describe the steps carried out during the translation process. The following table describes these steps or workflow events in chronological order.

Workflow Event	Description
Prepare the files in original format	Refers to the preparatory steps which must be carried out within the files in original format before saving them as MIF/IASCII. This work is carried out in FrameMaker or Interleaf.
Convert to STF	Refers to the conversion of the original format files to STF using S-Tagger. When this conversion takes place, the source and target RTF/TRADOSTag and ORG files are created.

Workflow Event	Description
Context TM	Process that compares updated source files to old bilingual documents rather than a translation memory. This is done by using the Context TM Wizard. This can be considered an optional additional step in the workflow, carried out after conversion to STF and before preparation and interactive translation.
Pre-translate	Process of pre-translating multiple files using the Translate command in Translator's Workbench. Translator's Workbench automatically inserts matches from the translation memory into the document for translation.
Translate/edit	Refers to the translation and editing that is carried out in the target STF files by a translator using TagEditor (or Word) as the editing environment.
Verify S-Tags	S-Tag verification allows you to verify the tag content of your target files during and after translation. Successful verification guarantees successful conversion back to the file format. This is carried out in TagEditor and/or S-Tagger.
Clean up	Refers to the Translator's Workbench feature which removes bilingual data from the target STF files and updates the translation memory. This step in the translation process is carried out before the target STF file is converted back to the original format.
Convert to original format	Refers to the conversion of STF files back to the original file format (MIF/IASCII) that was used as the initial source by S-Tagger during forward conversion.
Carry out post-production	Refers to the examination of the translated MIF/IASCII files and any required DTP work which is carried out after conversion back to the original file format.

Choosing an Editor

For the purposes of translation, both TagEditor and Word are integrated with Translator's Workbench, the translation memory system, and MultiTerm, the terminology management system. Both editing environments are capable of handling the STF tagged text files.

TagEditor however offers the following additional features:

- ❑ advanced tag protection – TagEditor can protect both external and internal tags, depending on the tag protection settings.
- ❑ flexible tag display – TagEditor uses colour formatting to differentiate between internal and external tags. It also provides you with a range of tag text display settings that can be customised to suit your requirements.
- ❑ powerful tag verification – this can be done at a segment or at a document level during and after the interactive translation process.
- ❑ Context TM compatibility – bilingual files from a previous project, that have been saved in the TRADOS tag (TTX) format with TagEditor, can be used as a basis for Context TM in future projects, cutting both translation time and costs.

Your choice of editor will determine the conversion options that you must select on the **Settings** tab in S-Tagger. If you are using TagEditor as your editor, you can select the TTX or RTF option; if you are using Word as your editor, you must select the RTF option.

TagEditor Workflow (TTX and RTF)

The following diagram represents the TagEditor translation workflow which can be considered the optimal workflow. This workflow offers maximum support for working with STF files.

Application	Workflow Event	Output
Windows Explorer	Create project structure	<i>MIF</i> or <i>IASCII</i> folder, <i>Source</i> and <i>Target</i> folder
FrameMaker or Interleaf	Save files as MIF/IASCII	Monolingual files in original format
S-Tagger	Convert files to STF	Monolingual STF and ORG files
Context TM Wizard	Context TM (optional)	Bilingual TTX file

Application	Workflow Event	Output
Translator's Workbench	Analyse against translation memory	File unchanged
Translator's Workbench	Pre-translate against translation memory (optional)	Bilingual TTX file
TagEditor with Translator's Workbench	Translate/edit	Bilingual TTX file
TagEditor with S-Tag Verifier plug-in	Verify S-Tags at document level	Bilingual TTX file
S-Tagger	Verify S-Tags in file batches or as single files	Bilingual TTX file
Translator's Workbench	Clean up bilingual TTX file and update translation memory	Monolingual RTF
S-Tagger	Convert to original format	Monolingual files in original format (MIF/IASCII)

Create Project Structure

This workflow event produces a *MIF* or *IASCII* folder and a *Source* and *Target* folder within your translation project folder. For more information, see “Project Structure” on page 2-9.

Save Files as MIF/IASCII

Using FrameMaker or Interleaf, this workflow event produces monolingual original format files, which are saved to the *MIF* or *IASCII* folder. For more information, see “Saving FM Files to MIF Format” on page 3-12 and “Saving ILDOC Files to IASCII Format” on page 7-14.

Convert Files to STF

This workflow event converts the MIF/IASCII files in the *MIF* or *IASCII* folders to STF format. The monolingual STF files are saved to the *Source* and *Target* folders that you created earlier. The *Source* folder now contains the source STF files (ancillary STF file optional) and the ORG file. The *Target* folder contains a copy of the STF files.

S-Tagger allows you to choose whether you want to convert files to RTF or TRADOS tag format. Both formats are equally acceptable. S-Tagger remembers the format that you choose during forward

conversion and expects to receive the same format for verification and backward conversion. This is important for the verification and conversion events described later in this section.

From this point in the workflow, all translation work is carried out on the target STF files. For more information, see “Converting MIF to STF” on page 3-24 and “Saving ILDOC Files to IASCII Format” on page 7-14.

NOTE

If you choose to convert to RTF file format, you will need to save the TTX file back to RTF after completing interactive translation. For more information, see “Verify S-Tags in File Batches or as Single Files” below.

Context TM

Context TM is only possible in the TagEditor workflow because the Context TM Wizard will only accept TTX or BIF bilingual files from earlier approved translations of your current project. If you have bilingual RTF files, you can open these in TagEditor and save them as TTX file format. Work can then be carried out using the Context TM Wizard

FOR MORE INFORMATION

For more information on this, see the *Context TM Wizard User Guide*.

Analyse against Translation Memory

The **Analyse** command in Translator’s Workbench analyses one or more files by comparing them to the currently selected translation memory. For more information on analysing the STF files against the translation memory, see the *Translator’s Workbench User Guide*.

Pre-translate against Translation Memory

As well as producing a bilingual target TTX file, under certain circumstances Translator’s Workbench can also produce a BAK file during this workflow event. The BAK file is a backup copy of the monolingual file with the same name as the original version.

The BAK file is only created when an RTF file is pre-translated in Translator’s Workbench and the **Save Workbench RTF as TRADOSTag** option is not selected.

The BAK file is not created when:

- ❑ a TTX file is being pre-translated
- ❑ an RTF file is being pre-translated and the **Save Workbench RTF as TRADOSTag** option is selected.

For more information on the translation memory settings, see the *Translator's Workbench User Guide*.

Translate/Edit

All files opened and translated in TagEditor are saved as TRADOSTag (TTX). This preserves all the bilingual data added to the files. When translation is complete, you should then verify the files in TagEditor, regardless of whether the files were originally in RTF or TTX format. Do not save the files back to RTF format before carrying out verification.

Verify S-Tags at Document Level

Using TagEditor, verification can be carried out at a segment and document level during and after translation of the files.

When verification is complete, you can save the file again as TTX or proceed directly to clean up, converting the file to monolingual RTF using the **Save Target As** command. You should save the file (in TTX format) before using the **Save Target As** command and retain a copy of this for future Xtranslation use, for correcting any tag errors that are highlighted during verification and for updating the translation memory. TagEditor does not overwrite the TTX file, but rather creates a new, additional RTF file.

This command removes all source segments and delimiting tags, leaving only the target language text. You should only use this command if you began the workflow with an RTF file. Note that the translation memory is not updated if you do not perform clean up in Translator's Workbench.

Verify S-Tags in File Batches or as Single Files

The target tag content has usually been already verified in TagEditor so this workflow event is really an additional QA step. The main advantage of the S-Tagger verification feature is that it can be used for batch verification; TagEditor verifies files one at a time.

In order for S-Tagger to successfully verify the files, the target files *must* be in the same format as the files that S-Tagger saved to the *Source* folder. For more information on the verification process, see "Verifying the S-Tags" on page 5-6.

Clean Up and Update Translation Memory

The main purpose behind this workflow event is to update the translation memory. The **Clean Up** command in Translator’s Workbench extracts the source language data from the target file and updates the translation memory. If you are performing clean up on a TTX file, you should ensure that you retain the most recent copy of this bilingual file as Translator’s Workbench will output a monolingual RTF after clean up. For more information, see the *Translator’s Workbench User Guide*.

Convert to Original Format

For this workflow event, you have the option of converting the bilingual TTX file or the monolingual RTF file that was created after clean up. If you select the TTX file for conversion, the S-Tagger backward conversion feature will remove any source language data while converting.

If you started the workflow with a TTX file, we recommend that you use the bilingual TTX file to convert back to the original file format. If you started the workflow with an RTF file, use the target monolingual RTF file for backward conversion.



NOTE

S-Tagger retains a bilingual TTX file of the translated documents after converting to original format. This means that your bilingual data is available for future use.

Word Workflow (RTF Only)

The following diagram represents the Word translation workflow using RTF as the file format. For more information, see “Choosing an Editor” on page 2-13.

Application	Workflow Event	Output
Windows Explorer	Create project folder structure	<i>MIF</i> or <i>IASCII</i> folder, <i>Target</i> and <i>Source</i> folder
FrameMaker or Interleaf	Save document files as <i>MIF/IASCII</i>	Original format files
S-Tagger	Convert to STF with STF format set to RTF	Monolingual STF and ORG files
Translator’s Workbench	Analysis against translation memory (optional)	File unchanged

Application	Workflow Event	Output
Translator's Workbench	Pre-translation against translation memory with the Save Workbench RTF as TTX option selected (optional)	Bilingual RTF and BAK file
Word with Translator's Workbench	Interactive translation	Bilingual RTF
S-Tagger	S-Tag verification (files can be processed singly or in batches)	Bilingual RTF
Translator's Workbench	Clean up and update translation memory	Monolingual target RTF (target language) and bilingual BAK file
S-Tagger	Convert to original format with STF format set to RTF	Monolingual files in original (MIF/IASCII)



SECTION 2:
S-TAGGER FOR FRAMEMAKER

PREPARATION AND MIF TO STF CONVERSION

Section 2 deals exclusively with S-Tagger for FrameMaker. This particular chapter explains the preparation and conversion of FrameMaker files.

Chapter sections include:

- ❑ preparing the FrameMaker files
- ❑ defining the conversion settings in S-Tagger
- ❑ converting from MIF to STF.

Chapter

3

PREPARING FRAMEMAKER FILES

This section contains information on features specific to the preparation of FrameMaker documents for conversion. You must complete a number of preparation steps before converting MIF files to STF using S-Tagger for FrameMaker.



NOTE

'S-Tagger' throughout this section refers to S-Tagger for FrameMaker.

The steps are divided into the following categories:

- ❑ preliminary guidelines – some guidelines to help you before beginning preparation
- ❑ project structure – preparation of your project folder which should be carried out before conversion in order to ensure smooth translation and separation of files
- ❑ required tasks – file preparation that *must* be carried out before converting files
- ❑ advised tasks – preparation that improves the quality of the STF files produced and minimises the DTP work required after translation
- ❑ special feature tasks – preparation that is only required when certain FrameMaker features have been used in the original FM files, such as SGML elements or Asian languages.



WARNING

You must have a good knowledge of FrameMaker and DTP if you are preparing FrameMaker files for translation.

Preliminary Guidelines

The following guidelines should be considered before beginning the preparation process:

- ❑ Remove any backup, crash, or corrupt files from the original document folder.
- ❑ Prepare the FrameMaker files on the platform that the files originated on. It is not necessary to port Macintosh or UNIX files to Windows to prepare them or save them as MIF.
- ❑ If you are working with FrameMaker files that have originated on a Macintosh, make sure that the **Remember Fonts** option is active. This will help avoid font conflicts between Macintosh and Windows.

- ❑ Convert only files that contain translatable text from MIF to STF.

Automatically generated files, such as table of contents and index files, should be converted only if there is text for translation in:

- ❑ variables or cross-reference formats
- ❑ paragraph numbering formats
- ❑ element or footnote prefixes
- ❑ suffixes.

This text is referred to as ancillary text and it is made available for translation in the ancillary file. The text in the generated files can be translated just once in the ancillary file and then re-inserted during conversion back to the original file format.

If the table of contents and/or the index have not been generated, but are manually built, the files should be included for conversion and translation.

- ❑ Before the files are translated, we recommend that you convert the new STF files back to MIF and open them again in FrameMaker to verify that the conversion process has worked correctly. To save time in porting, you can open the new MIF files in FrameMaker for Windows.

FrameMaker Version Information

S-Tagger for FrameMaker supports MIF files generated by FrameMaker version 4.0 or later, FrameBuilder and FrameMaker+SGML 5.0 or later. MIF files generated by the use of customised tools or applications are not supported by S-Tagger for FrameMaker. Manually built or edited MIF files are not supported by S-Tagger for FrameMaker.

Setting up the Project Structure

S-Tagger never changes the name of the files that it is working on. It generates files which have the same base name as the files being processed; only the extensions are different. Before you start working with S-Tagger, you must create a folder structure for your project. This facilitates file management and helps prevent files being overwritten.

During conversion, S-Tagger creates three new files for each MIF file:

- ❑ an STF file
- ❑ an ORG file
- ❑ a copy of the STF file.

S-Tagger can also create an ancillary file if you choose, saving it to the *Source* and *Target* folders. The ancillary file is described in detail in “The Ancillary File” on page 2-3.

FOR MORE INFORMATION

For more background information on all these files and project structuring, see “What is an STF File?” on page 2-2, “Other S-Tagger Files” on page 2-5, and “Project Structure” on page 2-9.

To set up your project structure using Windows Explorer:

- 1 Create a translation project folder in an appropriate location.
- 2 Copy all the FrameMaker and associated files (graphics or other referenced objects) to this folder. We recommend that you copy your original files so that if you make any mistakes, you will always be able to recopy the originals and start again.

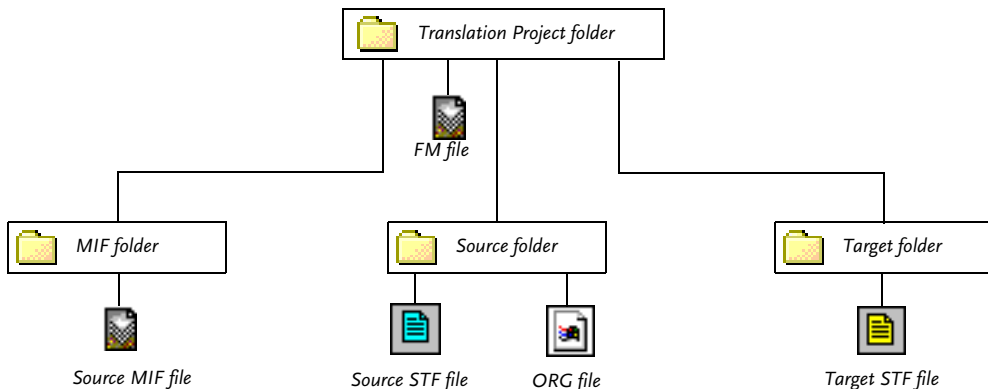
NOTE

You must maintain the same folder hierarchy as existed in the original FrameMaker project folder, otherwise, the FM files may not be able to locate all the referenced files. If your files contain graphics that need to be translated, you should deal with these separately and place them in a folder of the same name in the *Target* folder, maintaining the same folder structure. For example, if you have an *Art* folder in your original project folder, you must also have an *Art* folder in the translation project folder.

- 3 Create a *MIF* folder, a *Source* folder, and a *Target* folder within this project folder. The *MIF* folder will contain the MIF files saved from your original FM files and the *Source* and *Target* folders will contain the converted STF files.

When you have completed backward conversion of the MIF files, these new MIF files will be saved to the *Target* folder.

By the end of the conversion process, your folder structure and content will be something like this:



NOTE

- ❑ There may be an additional ancillary file in the *Source* and *Target* folders if you select the **Create ancillary file** option on the **Settings** tab. For more information, see “The Settings Tab” on page 3-14.
- ❑ For more information on STF file name extensions, see “STF File Name Extensions” on page 3-19.

Required Tasks

The following items can have an affect on MIF to STF conversion and should therefore be checked before attempting the conversion process. If you are unsure of how to carry out the steps, consult your FrameMaker manual or seek advice from qualified FrameMaker technical support staff.

Here is a brief checklist for you to refer to:

Issue	What to Check
FrameMaker Console messages	Are there any error messages informing you of font or language dictionary errors in the FM file?
Conditional text	Is conditional text set appropriately?
Change bars	Are the change bars turned off and cleared?
Cross-references	Are there any unresolved cross-references?
IXgen	Are all the markers collapsed?

FrameMaker Console Messages

The following messages may be displayed in your **FrameMaker Console** window.

- ❑ font substitution message – close the file without saving it. Locate the missing fonts and install them on your system. If the **Remember Missing Font Names** option is selected when the file is saved as **MIF**, you should also select the **Remember Fonts** option in your FrameMaker **Preferences** dialog box, to ensure that any font assignment is retained.
- ❑ missing language dictionaries – locate and install the missing language dictionaries on your system.

Conditional Text

If conditional text is to be translated, you must set it to **Show** so that it is presented for translation in the STF file. If the conditional text is not to be translated, it should be set to **Hide**.

Change Bars

Turn off **Automatic Change Bars** and select **Clear All Change Bars**. S-Tagger will not process MIF files which contain change bars. If you have turned on **Change Bars** in any character or paragraph format, you must turn them off in these formats. You may wish to globally turn off change bars in paragraph and character catalogs.

Cross-references

Check that there are no unresolved cross-references in the file. An unresolved cross-reference may indicate that the cross-referencing is not up to date. Update each unresolved cross-reference.

IXgen

If you are using IXgen to index your documents, check that all markers are collapsed before saving the file as MIF. If markers are expanded, you will not be able to convert the file to STF. If the file you wish to convert contains IXgen markers which are expanded, choose **Collapse Markers** from the **IXgen** menu on the menu bar. If you do not have a registered copy of IXgen, obtain another copy of the documents with markers collapsed.

Advised Tasks

Carrying out these preparation steps improves the quality of the STF files produced and minimises the DTP work required after translation.

Here is a brief checklist for you to refer to:

Issue	What to Check
Anchored frames	Are all the graphics contained in anchored frames?
Frame anchors	Are all the anchored frames which contain text boxes or text strings positioned at the beginning or end of a paragraph?
Hard returns	Have you checked the usage of hard returns in the FM file?
Hyphenation	Have all the suppress hyphenation and discretionary hyphenation characters been removed? Have you turned off automatic hyphenation?
Markers	Do any index markers need to be split? Are all the index markers at the end of a word or, preferably, at the end of a paragraph?

Anchored Frames

Check that all graphics and art files are contained within anchored frames. Graphics that are not in anchored frames may not retain their position in the translated files when the text expands during translation.

Frame Anchors

Ensure that all anchored frames which contain text boxes or text strings are positioned either at the beginning or the end of a paragraph. An anchored frame which is positioned in the middle of a paragraph breaks up the sentence, making translation of the segment difficult. Anchored frames which contain only graphics or rules may be positioned in the middle of a sentence, as they will be represented by internal tags.

Hard Returns

You have the option of selecting whether or not S-Tagger should insert a paragraph mark before a hard return tag (< :hr>) in the STF files. The **Insert line break before <:hr> tag** option is available in the **Settings** tab. For more information, see “The Settings Tab” on page 3-14.



NOTE

A hard return is known as a ‘forced return’ in FrameMaker.

- ❑ If you select **Insert**, you must check the document for instances of a hard return (manual line break), using the following guidelines:
 - ❑ If a hard return is *cosmetic* or has been inserted to make a paragraph look better in the source language, you should delete it. Otherwise, translators working with a translation memory system will see two segments instead of one sentence, making segment matching more difficult.
 - ❑ If a hard return has been inserted to create a new line without incurring new paragraph properties, there is no need to delete it.
- ❑ If you select **Don’t Insert** on the **Preferences** list, paragraph marks will *not* be inserted before the < :hr> tag. This is to facilitate translation of text when hard returns appear in the middle of a segment or in the middle of a word. You do not have to examine your documents for hard returns with this option.

Hyphenation

- ❑ automatic hyphenation – in FrameMaker you can turn off the automatic hyphenation in formats that have automatic hyphenation turned on. Alternatively, you can specify that hyphenation is turned off during conversion.

- ❑ suppress hyphenation characters – suppress hyphenation characters in FrameMaker are inserted into words that must not be hyphenated. Remove any such characters before converting the files.
- ❑ discretionary hyphens – remove any discretionary hyphens before converting the files.

Markers

FrameMaker markers are used to hold many types of FrameMaker text, for example, index content.

- ❑ maximum number of characters – FrameMaker supports a maximum number of 255 characters in the **Marker** dialog box. During translation the number of characters in a marker is likely to increase. For this reason, if markers contain more than 220 characters in the source language, you should split them into two or more markers.

You can find out if any of your index markers exceed 220 characters by running a test conversion from MIF to STF. S-Tagger reports long index markers in the MIF log file. For more information, see “Splitting Index Markers” on page 4-12.

- ❑ marker location – if an index marker, or any other marker, is placed inside a word, it will make the translation process less efficient. Ensure that all markers are placed outside individual words, and if possible at the end of the paragraph.



TIP

Use the following tips to search for certain items in FrameMaker’s **Find** dialog box:

- ❑ Enter \r to search for any hard returns.
- ❑ Enter \- to search for any discretionary hyphens.
- ❑ Enter \+ to search for any non-breaking hyphens.
- ❑ Enter _ to search for any suppress hyphenation symbols.

Special Feature Tasks

The following preparation is necessary if certain FrameMaker features have been used during the generation of the FM files. If you are using a standard edition of FrameMaker, refer to the following checklist. If you are using Asian languages with FrameMaker or FrameMaker+SGML, refer also to “FrameMaker+SGML Files” on page 3-II and “Asian Languages” on page 3-II.

Here is a brief checklist for you to refer to:

Issue	What to Check
OLE objects	Are all OLE objects contained in anchored frames?
Text insets	Do any text insets need to be translated and if so, have you dealt with them appropriately?
Text on reference pages	If text on the reference pages needs to be translated, have you saved it to a separate file?
PDF document information	Does your PDF Setup dialog box contain text that needs to be translated and if so, have you dealt with it appropriately?



NOTE

All symbols and special characters which can appear in FrameMaker are converted to either their ANSI text equivalent in the STF file or to an STF internal tag. A full list of all FrameMaker symbols and special characters can be found in your FrameMaker documentation.

OLE Objects

OLE objects are treated in a similar fashion to graphics. The anchored frame containing the OLE object is represented, but the actual object does not appear in the STF file. If there are any text boxes or text strings in the frame, the text within these is represented in the STF file.

Text Insets

S-Tagger always recognises text insets, and retains the inset properties. However, depending on the type of text inset, the text of the inset may or may not appear in the STF file.

- ❑ A text inset which is a reference to another document should be translated separately during translation. To prevent this text from appearing as text in the STF file, create a new condition in FrameMaker, apply the new condition to each referenced text inset and then hide the condition format before saving the file as MIF. When you convert the MIF file to STF, the text inset will appear as a hidden conditional text marker.

- ❑ A text inset copied into the FrameMaker document appears within text inset tags, but is treated as text for translation. The best way to prevent text insets from being translated is to insert them as OLE objects.

Text on Reference Pages

Translatable text on reference pages is not presented for translation in the STF file or the ancillary file. If you have any translatable text on reference pages, you should place it in a separate file for translation purposes.

PDF Document Information

FrameMaker 6.0 has a **PDF Setup** dialog box where you can specify the text that will appear in the **Document Information** dialog box in a PDF document. Each time you create a PDF document, the specified text appears in the **Document Information** dialog box. This text may require translation.

S-Tagger extracts the text from the **PDF Setup** dialog box and places it in the ancillary file for translation. Unlike the rest of the text in the document, the **PDF Setup** dialog box stores text in Unicode format. Extended characters (for example ‘á’) are inserted using Unicode character entities (for example, ¢). During MIF to STF conversion, S-Tagger converts the Unicode characters to regular text. Any character entities that cannot be represented as regular text are retained as character entities in the STF file.

If the **PDF Setup** dialog box contains characters that are not supported, conversion warning 310 appears:

```
This document contains PDF Document Info. S-Tagger cannot present this
text for translation unless you have language support for code page xxx
installed.
```

‘xxx’ refers to a code page.

S-Tagger supports the following code pages: 932, 936, 949, 950, 1250, 1251, 1252, 1253, 1254 or 1257. During the STF to MIF conversion process conversion warning 1450 appears if the target language is not supported:

```
The translated PDF Document Info cannot be inserted in the MIF file.
Your PC must support the language code page %d to be able to insert the
information.
```

In this case S-Tagger deletes the translated text and replaces it with the original text from the source files.

 **NOTE**

S-Tagger extracts **PDF Document Info** text that was inserted using the **PDF Setup** dialog box. Manually inserted **PDF Document Info** text is not supported.

FrameMaker+SGML Files

Prepare FrameMaker+SGML files in exactly the same way as FrameMaker files. The main differences in the FrameMaker+SGML STF files are as follows:

- ❑ There are additional tags representing the SGML elements. FrameMaker+SGML elements and element prefixes and suffixes have special tags in STF. For an example, see “FrameMaker+SGML Elements” on page -12.
- ❑ The text of an attribute value, applied to an SGML element, is not available for translation in either the STF or ancillary file.

Asian Languages

S-Tagger fully supports the following Asian languages as both source and target languages: Japanese, Korean, Simplified and Traditional Chinese. Certain Asian-specific features must be taken into consideration when preparing FrameMaker files for translation either to or from Asian languages.

You should note the following:

- ❑ character sets – S-Tagger does not support multiple Asian character sets in the same document.
- ❑ Rubi characters – Japanese Rubi characters consist of small characters that appear on top of other characters. They annotate characters, providing information about their pronunciation. While FrameMaker provides support for Rubi characters, S-Tagger does not. Any Rubi text found in a FrameMaker file is treated as ordinary text. If Rubi text is required in a file being translated into Japanese, this must be done in FrameMaker.
- ❑ source marker length – when translating into an Asian language, we recommend that the **Source Marker Length** setting on the **Settings** tab is set to 150 characters. This is because Asian characters take up twice the space of Western characters.
- ❑ cross platform files – S-Tagger does not currently support MIF files created on a UNIX platform for Traditional Chinese. These MIF files are encoded using EUC-CNS (CNS11643-1992). As FrameMaker only supports the EUC-CNS characters that also exist in BIG5, it should be possible to convert the EUC-CNS MIF files to BIG5 using a text conversion program. This limitation does not affect translation into Traditional Chinese.

**NOTE**

Since italic formatting is not used in Asian languages, you may want to use Font Mapper for FrameMaker to remove it before converting the files. For more information on Font Mapper for FrameMaker, see “Using Font Mapper for FrameMaker” on page 6-11.

**TIP**

When translating from Asian languages, convert the Asian MIF files to STF, make a small change to them, save and check them. Open the new MIF files and examine them for any problem characters. If you find any, go back to the original MIF files, search for the characters and replace them with an alternative Western character that is lower than 128 in the ANSI character set.

Saving FM Files to MIF Format

After you have completed all preparatory work in the FrameMaker files, you can proceed with saving them as MIF files.

To save FrameMaker files as MIF:

- 1 Launch FrameMaker.
- 2 Open a file that you want to translate.
- 3 From the **File** menu, choose **Save As**. The **Save As** dialog box is displayed
- 4 Browse to the *MIF* folder that you created earlier in your translation project folder. For more information, see “Setting up the Project Structure” on page 3-3.
- 5 Select **MIF** from the **Save as type** drop-down list and click **Save**. The FrameMaker file is now saved to your *MIF* folder in MIF format.
- 6 Repeat for each file that you want to translate.

Checking the MIF Files

Errors in the MIF files can be caused by faulty graphic export filters or other non-standard MIF statements inserted by some customised Doc-to-Help tools.

To check the MIF files:

- 1 In FrameMaker, open each MIF file that you saved to the *MIF* folder.
- 2 If any error messages are displayed in the **FrameMaker Console** window, save them to a readme file.
- 3 Close the files again without saving them.

- 4 Open the readme and check the content of the message.
- 5 Include the readme with the delivery of ORG and source STF files.

In most cases, the message will not mention any errors and you can ignore it. However, somebody else may be converting the files back to MIF after translation and it is important that they know the error was not introduced by S-Tagger and is not relevant.

DEFINING THE CONVERSION SETTINGS

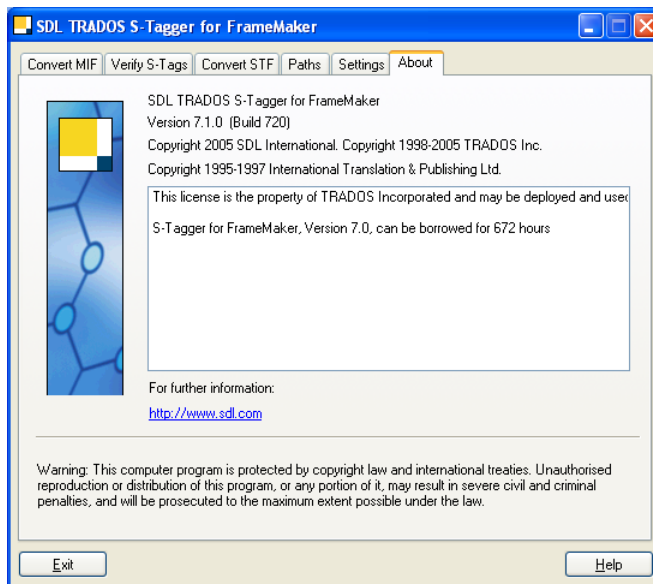
Before you convert the MIF files to STF, you must select the appropriate conversion settings. To do this, use the S-Tagger **Settings** tab and the **Paths** tab.

Launching S-Tagger for FrameMaker

To launch S-Tagger for FrameMaker:

- 1 From the Start menu, browse to *SDL International\SDL Trados 2006\Filters*.
- 2 Double-click on STagger for FrameMaker.

S-Tagger for FrameMaker opens with the **About** tab displayed.



S-Tagger for FrameMaker opens on the **About** tab by default. The **About** tab contains information about your licence and the S-Tagger version number.

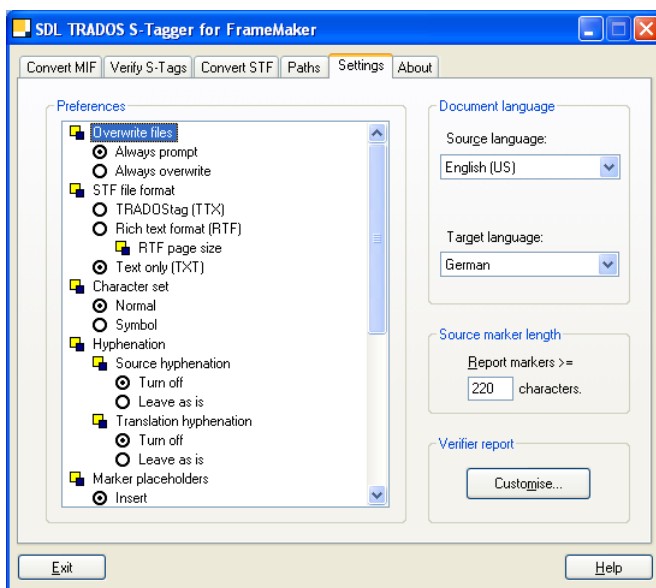
The Settings Tab

Before converting files to or from STF, or verifying the STF files, you must set the preferences for the file(s) you are processing. On the **Settings** tab, use the **Preferences** list to enter your conversion requirements. S-Tagger uses these settings during the conversion process to produce the STF files.

In this release of S-Tagger, you can convert directly to TRADOStag (TTX) format. Select this option under **STF file format** if you intend using TagEditor as your editing environment. For more information on choosing your editing environment, see “Choosing an Editor” on page 2-13.

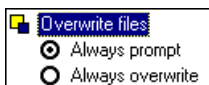
To enter your conversion settings:

- 1 In S-Tagger, open the **Settings** tab.



- 2 Select the appropriate options from the **Preferences** list. The various options are discussed in the following sections. Make sure that you examine all the options on the list before proceeding.
- 3 Select the correct source and target languages for the project. For more information, see “Document Language” on page 3-20.
- 4 Enter the maximum source marker length. For more information, see “Source Marker Length” on page 3-21.
- 5 Select the messages that you want to suppress in the verifier report. For more information, see “Verifier Report” on page 3-21.

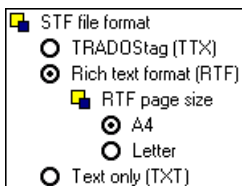
Overwrite Files



The default setting for this option is **Always prompt**.

- Select **Always prompt** if you wish to be prompted each time you are about to create new files which have the same names as existing ones.
- Select **Always overwrite** if you do not wish to receive this warning. Files with the same name are overwritten.

STF File Format



The default setting for this option is **Rich text format (RTF)**.

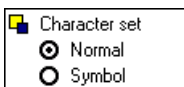
- Select **TRADOS tag (TTX)** to create TTX files suitable for translation directly in TagEditor.
- Select **Rich text format (RTF)** to create RTF files suitable for translation in Microsoft Word. Note that TagEditor accepts RTF files, but converts them to TTX format while you are working on them. For this option, you must also select the page size for the RTF files; **A4** is the standard page size for Europe and Asia and **Letter** is the American standard.
- Select **Text Only (TXT)** as the file format to create ANSI text files that can be opened and edited in any Windows text editor. The tags will not have any special formatting. ANSI text format is mainly used for communication with document management or machine translation systems and is not recommended for interactive translation.

To assist you with this selection, refer to “S-Tagger Workflow: An Overview” on page 2-10.

NOTE

When translating into Asian languages, you must choose **Rich text format (RTF)** or **TRADOS tag (TTX)** as the file format.

Character Set

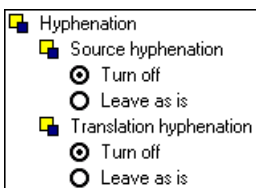


The default setting for this option is **Normal**.

- Select **Normal** if you are working with Western European languages.
- Select **Symbol** if you are working with Asian languages, either as the source or target files.

Hyphenation

Select the **Source** and **Target hyphenation** options to specify the hyphenation for the conversion from MIF to STF (**Source hyphenation**) and from STF back to MIF (**Translation hyphenation**).



The default setting for both the source and the translation hyphenation is **Turn off**.

Source hyphenation:

- Select **Turn off** to remove all hyphens. This is the default setting.
- If you select the **Leave as is** option, an `< :sh>` tag is inserted in the hyphenated word. For example, the following sentence appears in a FrameMaker MIF file:

This is a sentence with some hyphenation in it.

If you select **Leave as is** and convert the MIF file to STF, this sentence appears as follows in the STF file:

```
<ps "Indented" 2>This is a sentence with some hyphena< :sh>tion in it.
```

During translation, the sentence plus the internal tag representing the hyphen is added to the translation memory. Unless an identical sentence is found with matching hyphenation, leveraging is less likely. For this reason we recommend you turn off hyphenation.

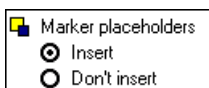
 **NOTE**

Whichever option you choose, when converting a MIF file to STF, S-Tagger warns you if any paragraph formats have hyphenation turned on.

Translation hyphenation:

- Select **Turn off** to switch off hyphenation for all paragraph formats in the target STF. This can be done even if hyphenation was turned on in the source files. As a result, the target text will not be hyphenated.
- Select **Leave as is** to preserve the original document's paragraph formats. If the original document did not include any paragraph formats with automatic hyphenation, the S-Tagger will not turn it on. This must be done manually, in the FrameMaker file.

Marker Placeholders



The default setting for this option is **Insert**. Under **Marker placeholders**, you can specify whether or not to insert marker placeholders at the marker location. This feature applies to index markers and other markers that may contain translatable text.

If a marker occurs within a sentence, S-Tagger moves the marker to the end of the paragraph. This allows you to finish translating a sentence before translating the text of the marker.

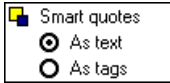
However, some markers may be placed strategically within the middle of a sentence, and should appear in a similar position within the translated sentence.

- Select **Insert** if you want to insert a marker placeholder at the point where the marker was originally found, while still placing the marker and its text at the end of the paragraph in the STF file. If marker placeholders have been inserted, the marker retains its position in the new translated file when the STF file is converted back to MIF.
- Select **Don't Insert** if no marker placeholders should be inserted. When the file is converted back to MIF, all markers will appear at the end of the paragraph in which they were originally found.

 **NOTE**

- If your document has markers that are placed in the middle of words and you select **Insert**, the placeholders will appear in the middle of words and you will have to translate around them.
- If you are converting FrameMaker+SGML files to STF, you must choose **Insert Marker Placeholders**.

Smart Quotes



The default setting for this option is **As text**. Under **Smart Quotes**, you can choose whether to have the smart quotation marks saved as **Text** or **Tags**. Examples of smart quotation styles are “English”, « French », „German“.

If you are using TagEditor as your editing environment, you should note that TagEditor does not display smart quotes – it will always use straight quotes. To produce smart quotes, insert the appropriate tags using the generic tag buttons (buttons 1 to 6) on the Tags toolbar. Alternatively, press Ctrl+6 for English, Ctrl+7 for French, or Ctrl+8 for German smart quotes.

If you are translating into languages that use straight quotes only, the **As text** option will suffice. However, for all other languages in TagEditor, we recommend that you select the **As tags** option.

If you are using Word as your editing environment:

- ❑ The **As text** option converts the quotes to the RTF keywords `\ldblquote` (left double quote) and `\rdblquote` (right double quote). In the French version of Word this is seen on screen as « » , and in the German version as „ “.

When the STF file is converted back to MIF these quotes are seen as the English smart quotes because they use the normal RTF keywords.

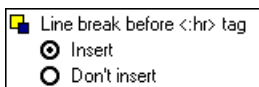
In a project where everyone is using the English version of Microsoft Word, you can use the **As Text** option and smart quotes can be typed the normal way for each language. This also applies to other languages that use the same smart quote characters as English.

- ❑ Select the **As tags** option to avoid losing the localised smart quotes. This converts all smart quotes to S-Tags. For example, `<:ldq>` for the English left double quote or `<:frq>` for a French right quote. When you want to insert or change a smart quote, you should also use the tags instead of the literal symbols. This way the smart quotes will remain as intended no matter what version of Microsoft Word you use.

➔ NOTE

- ❑ The **Smart quotes** setting does not apply to Asian languages as smart quote characters are among the characters above character position 127 which are not supported by S-Tagger.
- ❑ If the names of formats in FrameMaker have smart quotes in them, these are represented as straight quotes in the STF file if the **As text** option is selected, or as tags if the **As tags** option is selected.

Line Break Before <:hr> Tag



The default setting for this option is **Insert**. Using this option, you can select whether or not you would like S-Tagger to insert a new paragraph mark before each hard return (< :hr>) tag in the STF files.

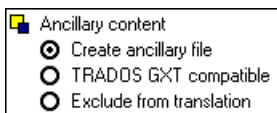
- Select **Insert** to insert a paragraph mark before each < :hr> tag.
- Select **Don't insert** to prevent paragraph marks being inserted before each < :hr> tag. This option facilitates a more layout-based translation.



NOTE

Hard returns are known as 'forced returns' in FrameMaker. For more information on S-Tagger's handling of hard returns, see "Hard Returns" on page 3-7.

Ancillary Content

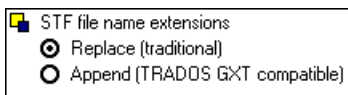


The default setting for this option is **Create Ancillary file**.

- Select **Create ancillary file** to create an ancillary STF file. S-Tagger extracts the ancillary text from the individual MIF files and saves it to a single ancillary file. The ancillary content can then be translated just once and is automatically re-inserted back into each MIF file during backward conversion.
- Select **Exclude from translation** to ignore the ancillary text. If you select this option, this text will not be made available for translation.

For more information on the ancillary file, see "The Ancillary File" on page 2-3.

STF File Name Extensions



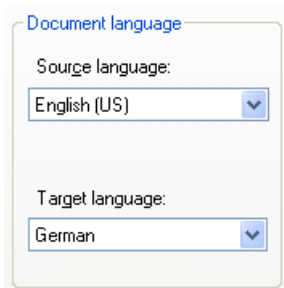
The default setting for this option is **Replace (traditional)**.

STF file name extensions allows you to specify whether you wish to use long or short STF file name extensions. In certain situations, the long file extension may help you to avoid inadvertently overwriting files.

- ❑ Select **Replace (traditional)** if you want S-Tagger for FrameMaker to apply the following file extensions to the converted STF files: `*.rtf`, `*.rtf.ttx`, or `filename.txt` (depending on your choice of STF file format), and `*.org` in your source folder.
 - ❑ The CMP file extensions are also influenced by this setting. If you select **Replace (traditional)**, the CMP file is `*.cmp`. The file extension of the ancillary file (if you have chosen to create one) is not affected by this setting.

Document Language

To enter the languages in the **Document language** box, simply select the appropriate languages from the **Source language** and **Target language** drop-down lists.



! WARNING

During conversion to STF and verification in S-Tagger, ensure that you set the source language correctly, otherwise, the spell-checking and other language-related features of FrameMaker may not work correctly.

The **Target language** should be set when converting the translated STF files back to MIF; this is to cater for projects dealing with multiple target languages. This setting allows you to change all instances of the source language found in the document to the target language. It will not change the language assignment of paragraph or character formats which have been assigned a different language to the specified source language, or none at all. If the source document language has been set incorrectly, the **Target language** setting will not work and the translated file will retain the language setting of the source file.

The languages listed in the **Document language** group box are those languages for which FrameMaker dictionaries are available. If the language you are translating into does not appear on the list, set the **Target language** to **None**.

Japanese (WinAlign) is included as a source language so that STF files can be generated that use <:so> tags instead of <ss> or <s1> tags in index markers. This is useful for alignment purposes because Japanese documents contain sort strings for all index entries and index levels. The internal <:so> tag ensures that the segment is not broken up. If you are converting Japanese MIF files to STF for the sole purpose of alignment, select Japanese (WinAlign) in the **Source language** list box.



FOR MORE INFORMATION

For more information on alignment, see the *WinAlign User Guide*.

Source Marker Length

To enter the source marker length, simply enter the appropriate amount in the **Report markers** text box.

The default value is 220 characters.

For Western languages we recommend you specify a maximum length of 220 characters, and for Asian languages, 150 characters.

In FrameMaker, index and other markers contain a maximum of 255 characters. During translation, the size of the text in a marker may increase. Use the **Source marker length** box to specify the maximum number of characters acceptable. If S-Tagger detects more characters than specified in this box, you will receive a warning during the conversion process.



NOTE

If you are translating FrameMaker+SGML files, you cannot insert new index markers in the STF file.

Verifier Report

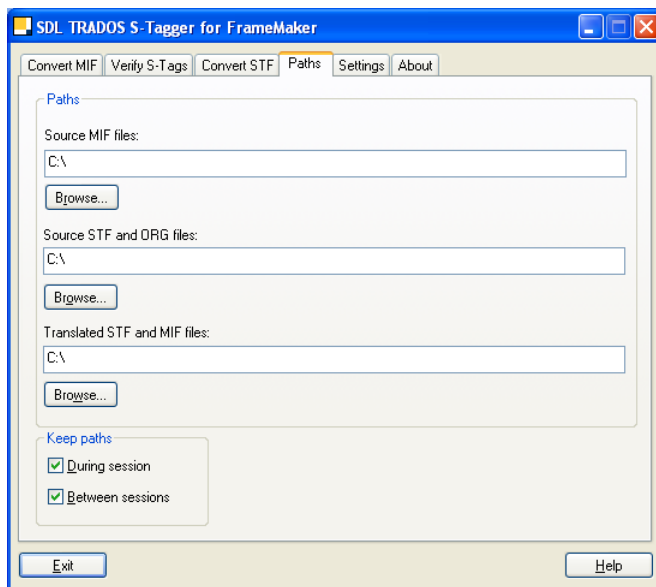
The verifier report is the verification information displayed in the **Results** window on the **Verify S-Tags** tab. For more information on the verifier report, see “Verifier Report” on page 5-4.

To use the **Customise** command button:

- 1 Click **Customise** to activate the **Customise Verifier Report** dialog box.
- 2 Select the alerts and warnings you wish to suppress during the tag verification process.

The Paths Tab

Use the **Paths** tab to set the paths for storing the S-Tagger project files before you start converting or verifying your files. You can also set or change the paths at each step in the process. The **Paths** tab consists of a group of text boxes for the various default paths. For more information on the recommended paths to use, see “Project Structure” on page 2-9.



To enter the paths for your translation project:

- 1 In S-Tagger, open the **Paths** tab.
- 2 Enter the paths to the folders you are using or click the **Browse** button underneath each option to enter a path.



NOTE

When you first open S-Tagger, the paths point to the C:\ drive by default.

- 3 Select the **During Session** option if you want S-Tagger to remember the file path for all project files. These paths are then used as the defaults until you close S-Tagger.
- 4 Select the **Between Sessions** option if you want the default paths to be written to the registry. This means that the next time you use S-Tagger, the same paths are displayed as defaults.

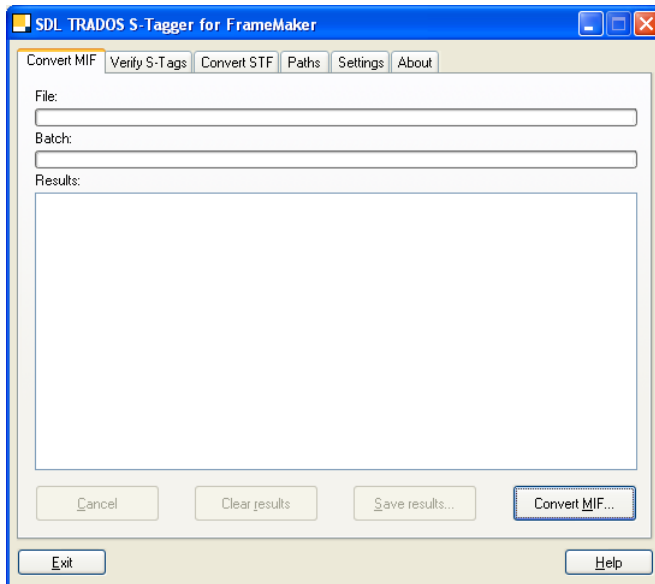
This option can only be selected if you have already selected **During Session**.

CONVERTING MIF TO STF

Once you are sure that your preferences are set correctly in the **Settings** and **Paths** tabs, you are ready to convert your MIF files to STF format.

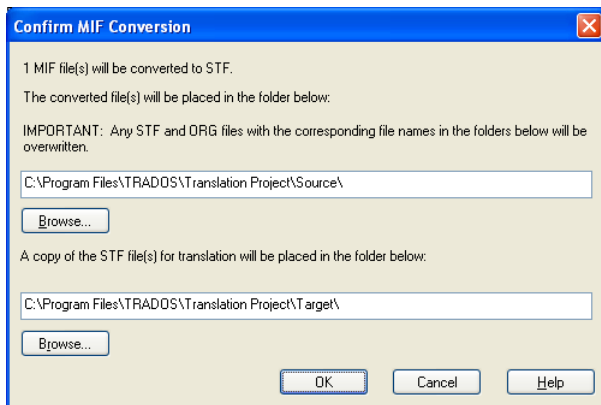
To convert your MIF files to STF:

- 1 In S-Tagger, open the **Convert MIF** tab.



- 2 Click the **Convert MIF** command button to select the file(s) you wish to convert. The **Select MIF File(s) to Convert** dialog box is displayed.
- 3 Select the MIF files that you want to convert and click **Open**.

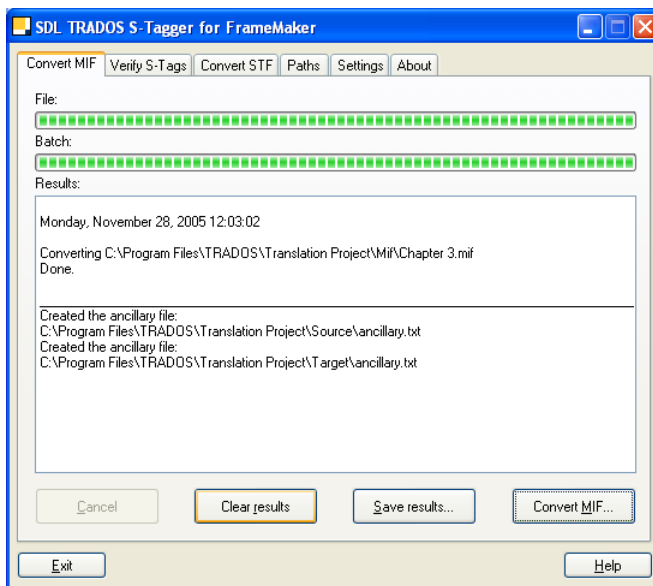
The **Confirm MIF Conversion** dialog box is displayed, prompting you to confirm the conversion.



The paths listed here are the same as those you entered on the **Paths** tab.

- 4 Click **OK** to accept the listed paths or enter new file locations by clicking the **Browse** button.

As each file is being converted, a series of messages is displayed in the **Results** window. These include any warnings about the file content or error messages from the file conversion. The **File** and **Batch** progress indicators show you how much of the process is complete.



When file conversion is complete, a message box is displayed which informs you that the file(s) converted correctly.

- 5 Click **OK** on the message box. Your MIF files have now been converted to STF format.
- 6 If you want to save the text in the **Results** window to a log file, you can do this by clicking **Save Results**. A dialog box is displayed, prompting you to specify the name of the file the results should be saved to. The log file is automatically given the extension `*.txt` and can be opened in any text editor.

Testing the New STF Files

After converting to STF, we recommend that you do the following:

- 1 In Windows Explorer, open your translation project folder.
- 2 Check that the source STF files and ORG files are in the *Source* folder.
- 3 Check that there is a copy of each STF file in the *Target* folder.
- 4 Make a copy of all the STF files in both folders and save them to a separate folder called *Test*.
- 5 Open each STF file in your chosen editing environment, make a small change to the text, convert the files back to MIF and open each one in FrameMaker. Be sure to make a similar change to the text in the ancillary file.

This test confirms that your STF files have not been corrupted during the conversion process and may be used safely for translation. Checking the STF files is especially recommended if you are translating multiple files into multiple languages.

- 6 When you are satisfied that the STF files are in order, you can delete the test copies and continue working with the files in the *Source* and *Target* folders.



WARNING

- ❑ *When making an artificial change to the STF files for the purpose of testing the files, be sure to change the text only, not the tags. If you change any of the text in a tag, the tag will no longer be valid and the file will not convert back to MIF.*
- ❑ *If you are using a search and replace routine in Microsoft Word, make sure that you set the search and replace to change only the default paragraph text format and not the `Tw4winInternal` or `Tw4winExternal` text formats.*



STF FILE TRANSLATION

This chapter guides you through the translation process.

Chapter sections include:

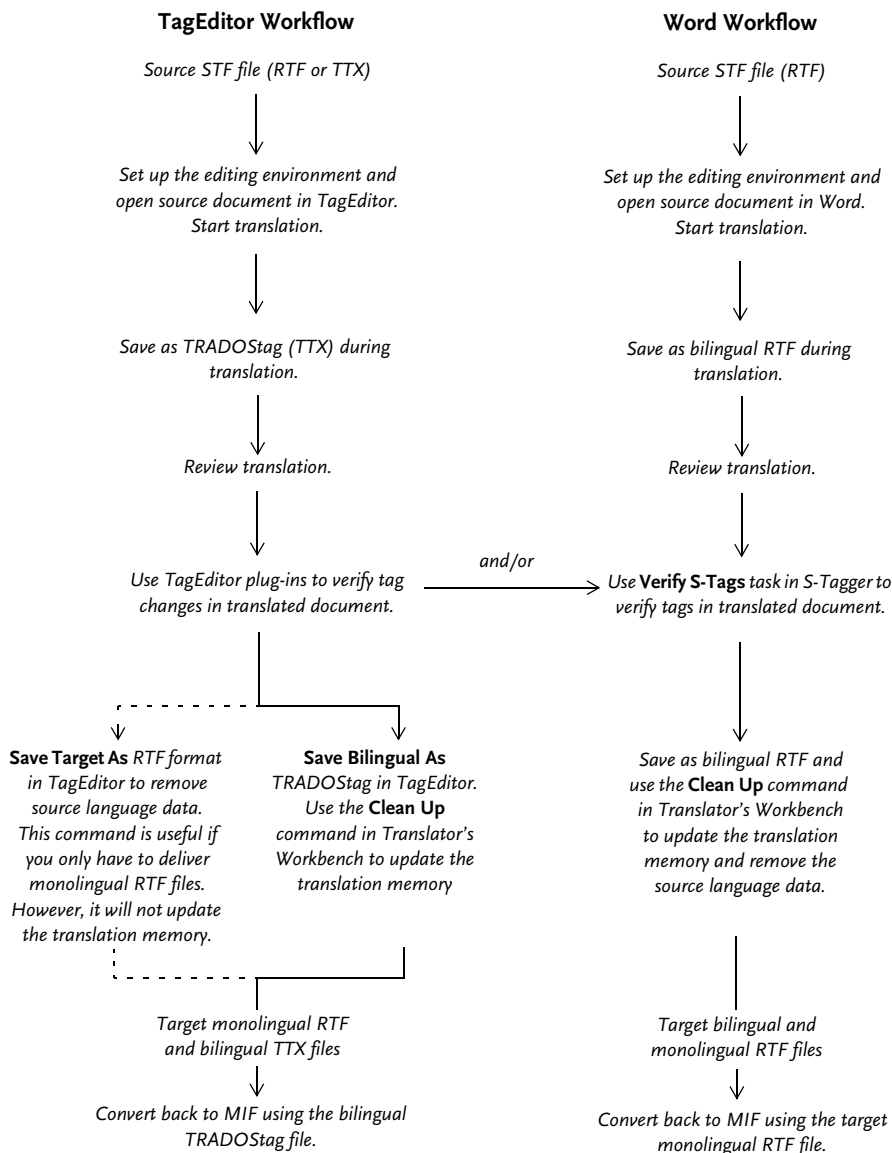
- ❑ translating STF files
- ❑ using S-Tags during translation
- ❑ translating to Asian and East European languages.

Chapter

4

TRANSLATING STF FILES

This section provides you with guidelines for translating STF files and using S-Tags. It also highlights some issues that may need particular attention during translation. The following diagram illustrates a high-level overview of the workflow for interactive translation in either of the editing environments. The workflow includes translation proper, tag verification and clean up of the translated STF files.



General Translation Issues

The following section describes some issues that may need your particular attention during translation, regardless of what source and target language you are using.

Localisation

The localisation issues that frequently arise during translation are as follows:

- ❑ formatting changes – some languages may require the addition of bolding or italics to some characters; in most translations, formatting tags need to be moved and/or deleted or added; in most translations, punctuation will be different to some degree.
- ❑ carriage returns and/or sentences or words are sometimes deleted – some translations require the merging of paragraphs or sentences, or the deletion of a word which has been inserted as, for example, a variable.
- ❑ paragraphs and/or sentences are sometimes added – some translations require a paragraph in the source language to be split into two paragraphs in the translation.
- ❑ paragraphs are often moved to positions in the translation that are different to their positions in the source.

Ancillary File

If you have chosen to create a separate ancillary file on the **Settings** tab, you must remember to translate this text also. This translated text is re-inserted into the files during the conversion back to MIF. For more information, see “S-Tags in the STF Ancillary File” on page A-26.

File Format

When you save the files after finishing translation, ensure that you save them in the customer’s required delivery format:

- ❑ If you have used TagEditor for translation and you are asked to deliver the target STF files back in RTF file format, you can do either of the following:
 - ❑ Use the **Clean Up** command in Translator’s Workbench and deliver the RTF file created during clean up. This option is recommended as it updates the translation memory.
 - ❑ Use the **Save Target As** command in TagEditor to save the bilingual file as a monolingual RTF file.
- ❑ If you have used Word for translation and you are asked to deliver the target STF files back in TTX file format, you can open the bilingual (uncleaned) RTF file in TagEditor and save it as TTX.

WORKING WITH S-TAGS

The following sections describe the S-Tags and how you can manipulate them during translation. Translator's Workbench ignores external tags and treats internal tags as placeables. This means that while they do not require translation, you must place them in the correct location in the target segment.

If you are using a translation memory from a previous project, remember that the tags in the translation memory may differ from the tags in the file for translation. You must check that the tags from the source segment in the file are included in the target segment.

At any stage during translation, you can check that you are working with the tags in the correct way by using the S-Tagger verification feature or TagEditor's S-Tag Verifier plug-in. Your verification options depend on the editing environment you are using.



FOR MORE INFORMATION

For more information, see:

- ❑ "S-Tagger Workflow: An Overview" on page 2-10
- ❑ "Choosing an Editor" on page 2-13
- ❑ the TagEditor tutorial in the *Translator's Workbench User Guide*
- ❑ the Word tutorial in the *Translator's Workbench User Guide*.

General S-Tag Guidelines

There are clearly defined rules for adding, deleting or moving tags. To successfully convert the STF files back to MIF, check the following:

- ❑ all the tags have correct opening and closing signs ('<' and '>')
- ❑ the tag pairs have both the opening and closing tags
- ❑ the correct tag is always used to start and end the STF file:
 - <stf "Fx.xx"> and </stf>, where x.xx is the version number of the STF file format.
- ❑ the following tags have the correct values:
 - ❑ <sourcecharset "xxx">
 - ❑ <sourcelanguage "xxx">

- `<sourcequotes "xxx">`
- `<sourcehyphenation "xxx">`
- `<sourcepath "xxx">`
- `<ancillarymode ?>` where ‘?’ can be a value between 0 and 2
- all the tags are valid (correct S-Tags)
- the correct sequence has been followed (some tags must follow a particular sequence; for example `<imk ?>` must be followed by at least one `<ie>` tag and a closing `</imk>`)
- all the tags are in the correct position, for example, external tags usually occur at the beginning of a paragraph.

Since TagEditor provides advanced tag protection and insertion capabilities, many of the following guidelines are more applicable to Word. Information that relates specifically to the Word editing environment is clearly identified.



FOR MORE INFORMATION

For more information on tag protection and insertion in TagEditor, see “Inserting Tags in TagEditor” on page 4-6 and the TagEditor tutorial in the *Translator’s Workbench User Guide*.

Formatting

To apply bold or italic formatting, or to apply small caps, insert the tag `< : b>` for bold, `< : i>` for italic, `< : bi>` for bolded italics or `< : s>` for small caps, in front of the word you wish to format. Place the closing tag (`< : /b>`, `< : /i>`, `< : /bi>` or `< : /s>`) at the point where you wish to return to the default font.

If you are using Word, you should always use the *Normal* font to enter tag text. Any formatting, apart from that done with tags, will be lost.

If you are using TagEditor, you can insert such common formatting tags by using the Tags toolbar or by right-clicking in the target segment and choosing the **Insert Tag** command from the shortcut menu.

Extended Characters and Symbols

Insert extended or accented characters using the keyboard, *not* the **Insert Symbol** command in Word or the Character Map utility in Windows. ALT sequences and country-specific keyboards can be used without any problem.

If symbols appear in the hard copy, they may appear either as anchored frame tags, as single characters surrounded by font change tags, or as internal tags in the STF files. For example,

<:fc 3>S<:/fc>. This implies that the letter 'S', when given the character style 'fc 3', appears as the symbol you see in the hard copy.

Special characters, like hard spaces or thin spaces, cannot be inserted in the ordinary way in Word. If you want to use these features, you must insert the corresponding S-Tag where you want the character to appear. For example, <:hs> for hard space or non-breaking space.

If you are using TagEditor, you can insert some of the common symbols by using the Tags toolbar or by right-clicking in the target segment and choosing the **Insert Tag** command from the shortcut menu.

Quotation Marks

You can insert quotation marks either by typing in the ANSI code or by inserting the S-Tags for quotation marks. If smart quotes are used in the document, English smart quotes appear as the tags <:ldq> for the opening quote and <:rdq> for the closing quote; French quotes appear as <:flq> and <:frq>; German quotes appear as <:glq> and <:grq>.

If you are using a localised version of Microsoft Word, you must insert smart quotes as tags when translating into German and French.

NOTE

Even if you select **Smart Quotes As Text** in the S-Tagger **Settings** tab, you can still use the tags to insert the localised smart quotes.

If you are using TagEditor, you can use the generic tag buttons (buttons 1 to 6) on the Tags toolbar to insert a variety of quotation marks. You can also right-click in the target segment and choose the **Insert Tag** command from the shortcut menu.

Inserting Tags in TagEditor

There are two methods of inserting tags in TagEditor:

- the Tags toolbar
- the **Insert Tag** dialog box.

The Tags toolbar in TagEditor allows you to quickly and easily insert tags and special characters into target segments during translation. The toolbar consists of a series of buttons; each button corresponds to a different tag, tag pair, or special character. The content of the toolbar is different for each file format that TagEditor supports.

The **Insert Tag** dialog box is accessed by right-clicking in the document and choosing the **Insert Tag** command from the shortcut menu. This displays the **Insert Tag** dialog box which contains a wide range of tags that you can insert into the target segment. This list displays the internal tags that you

can insert, but if you want to use the list to insert external tags, simply select the **Also show external tags** option.

Once you have inserted a tag using the **Insert Tag** command, that tag is displayed in a list on the shortcut menu. If you right-click in the target segment, you can select the tag directly from this shortcut menu list, without having to access the **Insert Tag** dialog box. The list can contain up to ten recently used tags.



FOR MORE INFORMATION

For more information, see the TagEditor tutorial in the *Translator's Workbench User Guide*.

Manipulating Internal Tags

An internal tag represents simple formatting information which often changes with translation. During translation, you will probably need to manipulate some of the internal tags. For example, some languages may require the addition of bolding to some characters. In most translations, formatting tags need to be moved, deleted or added; punctuation will also vary to some degree.

The following table outlines which internal tags may be added or deleted. If a tag is not listed here, it should never be added or deleted. Note that none of these tags contain numbers.

Internal Tag	Represents
<:hr>	Hard return (also known in FrameMaker as a forced return)
<:hs>	Hard (non-breaking) space
<:lt>	Less than sign
<:gt>	Greater than sign
<:sh>	Hyphen
<:t>	Tab
<:lq> and <:rq>	English left and right single quotes
<:ldq> and <:rdq>	English left and right double quotes
<:flq> and <:frq>	French left and right double quotes
<:glq> and <:grq>	German left and right double quotes
<:ems>	Em space
<:ens>	En space
<:dh>	Discretionary hyphen

Internal Tag	Represents
<code><:nh></code>	No hyphen
<code><:hh></code>	Hard (non-breaking) hyphen
<code><:ns></code>	Number space
<code><:ts></code>	Thin space

Frequently Used Formatting Tags

These tags are never found in the source STF file, but may be added to the target STF as required. They are used to cater for the additional bolding, italicising, capitalisation, and unique character usage that may be needed when translating to certain languages. For example, in German, bolding is used to emphasise words much more frequently than in English.

If such formatting does occur in the original source STF, it is represented by character style or font change tags, not by these tags.

Internal Tag	Represents
<code><:b></code> and <code><:/b></code>	Bold
<code><:i></code> and <code><:/i></code>	Italic
<code><:bi></code> and <code><:/bi></code>	Bold italic
<code><:s></code> and <code><:/s></code>	Small caps
<code><:c></code> and <code><:/c></code>	Character set
<code><:c1></code> and <code><:/c1></code>	
<code><:c2></code> and <code><:/c2></code>	
<code><:so></code>	Sort order. Note that where the source STF file was created using Japanese (WinAlign) as a source language, it may contain <code><:so></code> tags.



NOTE

If you are using FrameMaker+SGML, and you are adding any of the frequently used formatting tags, be sure to notify the DTP personnel. They will need to apply the correct structure to the additional formatting so that it is retained when the files are saved as SGML.

Other Internal Tags

These tags may be deleted to change the original formatting of a piece of text. You may add the character style, font change, variable and cross-reference tags where they already exist in the source text. For example, you may wish to use two instances of a variable tag. You cannot create new ones.

Internal Tag	Represents
<code><:fc ?></code> and <code><:/fc ?></code>	Font change
<code><:cs "xxx" ?></code> and <code><:/cs></code>	Character style
<code><:v "xxx" ?></code>	Variable
<code><:xr "xxx" ?></code>	Cross-reference

Manipulating External Tags

An external tag represents the kind of formatting that generally does not change, for example structural formatting. During translation it may be necessary to add or delete certain external tags. These tags should only be modified by experienced users of both FrameMaker and S-Tagger.

The only external tags that may be added or deleted are:

- paragraph styles
- paragraph numbering formats (prefixes)
- index markers.

If an external tag is to be deliberately added to the file, it must contain a '#' as its second character. For example, to add the external tag `<ps "Head 3" 2>` to a file, the tag must be written as `<#ps "Head 3" 2>`. The '#' characters are deleted during the conversion of the STF files back to original file format.

If you are adding tags, we recommend that you copy an existing tag and change the text, being very careful to keep any spaces before, after and within the opening and closing braces.

If an external tag from the allowed list is added without a '#' character placed in front of the tag type identifier, it is detected as an invalid tag and the file returns an error during verification.

**WARNING**

Do not invent tag numbers. If you do, this will prevent the translated STF file from being converted successfully back to MIF.

If you want to modify external tags in TagEditor, you must first remove all tag protection. To do this, use the **Protection** tab in the **Options** dialog box. When you have deleted or moved the external tag, we recommend that you reactivate the tag protection.

Paragraph Styles

Paragraph styles are represented as `<ps "xxx" ?>` in the STF files. If you add a paragraph style tag to an STF file, you must use a paragraph style that already exists within the document and insert a hash character in the additional tag so that the tag appears as `<#ps "xxx" ?>`. If you do not follow these conventions, the file will not convert back to MIF.

Never add a paragraph style tag to an STF file unless you are certain of the effect it will have on the formatting. Remember that paragraph styles are usually used to generate the table of contents of a FrameMaker document, so changing the style may have an unwanted effect on the table of contents.

Deleting a paragraph style tag and not replacing it with another tag means that all the text in the paragraph is moved to the end of the previous paragraph, without a new paragraph break. The style assigned to the previous paragraph is assigned to the merged paragraph text.

TagEditor does not allow you to insert paragraph style tags using the Tags toolbar or the **Insert Tag** command. You are only allowed to add those paragraph style tags that already exist in the document. This is why you must copy an existing paragraph style tag and paste it to the desired position in the document in order to apply that paragraph style.

If you have copied and pasted such a tag, you must follow these steps to ensure that S-Tagger will treat it as a valid custom external S-Tag:

- 1 In TagEditor, turn off the document and tag protection. For more information, see the *Translator's Workbench User Guide*.
- 2 Select the tag and right-click on it.
- 3 Choose **Toggle # in S-Tag** from the shortcut menu. This automatically adds a hash character to the tag.

**NOTE**

This command is only available when a `<ps "xxx" ?>` or `<pn "xxx" ?>` tag is selected.

- 4 When you have finished editing the tags, we recommend that you turn the document and tag protection back on.

Paragraph Numbering Formats (Prefixes)

Paragraph numbering formats (also known as prefixes) are used if certain text or symbols should always precede a particular paragraph style in FrameMaker.

Prefix text is presented as `<pn "xxx" ?>` in the STF files. A `<pn "xxx" ?>` tag may only be added to the STF file if it is accompanied by its relevant paragraph style tag. Each prefix tag must follow the paragraph style tag to which it belongs.

If you add or delete a `<ps "xxx" ?>` tag, make sure that you also add or delete any corresponding `<pn "xxx" ?>` tags. Deleting a `<pn "xxx" ?>` tag on its own will not delete the prefix from the paragraph in the FrameMaker document. When adding this tag, you must also insert the hash (#) character so that the inserted tag appears as `<#pn "xxx" ?>`.

If you are working in TagEditor, follow the same procedure as described in “Paragraph Styles” on page 4-10 to insert prefix tags and ensure that S-Tagger will treat them as valid custom external S-tags.



NOTE

Neither the `<ps>` nor the `<pn>` tag are available on the **Insert Tag** dialog box or the Tag toolbar. This is because you must use a tag that is already used elsewhere in the STF file.

Index Markers

You may also add and delete index markers. If you are adding an index marker, you must insert the '#' character; do not include a number. An `<imk>` tag must be followed by at least one `<ie>` tag and an `</imk>` tag.

If you add an index marker sequence to the file, the index marker is positioned at the end of the text paragraph immediately preceding the marker in the STF file. You cannot add an internal `<:imk ?>` marker to position the marker within the text.

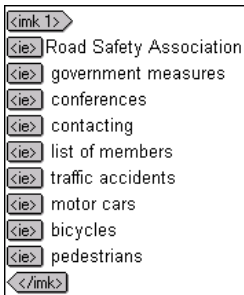
External Tag	Represents
<code><imk ?></code>	Index marker tag Deleting an <code><imk ?></code> tag deletes an index marker. If an <code><imk ?></code> tag is deleted, all following text and tags, up to and including the closing <code></imk></code> tag, must also be deleted.
<code><ie></code>	Index entry Deleting an <code><ie></code> tag deletes an index entry. Never delete an <code><ie></code> tag unless you are deleting an entire index marker. If you need to radically change the structure of an index entry and are confident in what you are doing, replace the existing index marker in its entirety with a new index marker, using the # syntax.

External Tag	Represents
<code><il></code>	Index entry subentry Deleting an <code><il></code> tag deletes a subentry to an index entry. Only delete <code><il></code> tags if you are sure you wish to delete the subentry.
<code><sl></code>	Sort level Deleting a sort level tag and its text replaces the sort level text with nothing. If you wish to delete a sort level entry, delete only the text and leave the tag in place.
<code><ss></code>	Sort string Deleting a sort string tag and its text replaces the sort string text with nothing. If you wish to delete a sort string entry, delete only the text and leave the tag in place.
<code></imk></code>	End index marker tag

Splitting Index Markers

If, during the verification process, you discover that an index marker contains more than 255 characters after you have translated it, you must split the index marker to create two index markers with less than 255 characters.

For example, the following index marker is found in an STF file in TagEditor:



To split this index marker into two markers:

- 1 End the `<imk 1>` after the index level *list of members*. Do this by inserting the `</imk>` tag on the next line.
- 2 Use the '#' syntax to create a new index marker. Start the index marker by inserting the `<imk>` tag. Do not place any number in this tag. Place a '#' at the start of the index entry tag `<#ie>` and at the start of each of the index level tags `<#il>`. Remember to end the index marker with a `<#/imk>` tag.

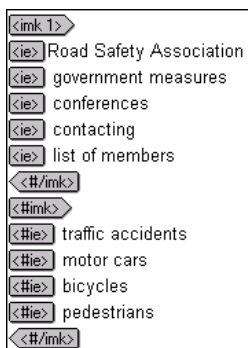
In TagEditor, you can use either the **<O>** button from the Tag toolbar, which includes a preconfigured `<#imk><#ie><#/imk>` tag combination or right-click in the document and choose the **Insert Tag** command from the shortcut menu. This opens the **Insert Tag** dialog box.

Select the **Also show external tags** option so that you are able to insert `<imk>`, `<ie>`, and `<il>` tags. After you have inserted a tag, this tag is listed in the **Insert Tags** command's submenu. This submenu displays a list of recently used tags.

NOTE

The inserted external tags are automatically prefixed with the hash character (#).

The split index marker will appear as follows:



WARNING

No external tag other than those described above should be added or deleted in the target STF. When working with STF files from FrameMaker+SGML, you must not add or remove paragraphs or index markers.

ASIAN AND EASTERN EUROPEAN LANGUAGES

If you are translating into or from an Asian language, this section is particularly relevant and should be examined thoroughly. If you are translating into an Eastern European, Greek, Cyrillic, Turkish, or Baltic language, first confirm whether the fonts being used require special treatment as symbol fonts. If they do, you may find the following section helpful.

The following issues may arise when translating into these languages:

- ❑ character set tags
- ❑ unsupported characters
- ❑ index marker sort levels
- ❑ markers that must contain both target and Western text.

Character Set Tags

To retain Western extended characters in the translated MIF file, insert character set tags (`< : c >` and `< : / c >`) around the character or word.

For example, if your documentation contains the product name ‘Qualité’ and you wish to retain the ‘é’ character in the product name in the final translation, use the character set tag as follows:
`< : c >Qualité< : / c >`.

If your Western text in the source document is formatted using one of the following fonts - Times New Roman, Courier New or Arial - you may wish to retain this font in the translated MIF file. Do so by adding `< : c >` and `< : / c >` to insert Times New Roman font, `< : c1 >` and `< : / c1 >` to insert Courier New font and `< : c2 >` and `< : / c2 >` to insert Arial font.



NOTE

You cannot place the `< : c >` tags in markers. Markers only support one character set, that of the language you are translating into.

If your source text is not formatted using Times New Roman, Courier New or Arial fonts, then use the `< : c >` tag syntax and map the font after translation. For more information on mapping fonts, see “Using Font Mapper for FrameMaker” on page 6-11.

Unsupported Characters

Some Asian characters which are available to translators working with TagEditor or Microsoft Word are not supported in FrameMaker. These characters exist for Japanese, Korean and both Simplified and Traditional Chinese. If an unsupported character is inserted during translation, S-Tagger will detect the character and you will receive a message when you verify the files.

Japanese

FrameMaker uses the EUC-JP character set on the UNIX platform and the Shift-JIS character set on the Macintosh and Windows platform. S-Tagger for FrameMaker supports MIF files generated on any of these platforms. Microsoft has added characters to the Shift-JIS standard and if any of these characters are used in FrameMaker, they will be displayed as nonsense in FrameMaker under UNIX and Macintosh. During translation, you will have access to these characters and there is no way of preventing their insertion into the translated STF file. When you verify the STF file(s), a warning message is generated, alerting you to the fact that the STF file contains unsupported characters.

Simplified Chinese

FrameMaker uses the GB2312-80 character set on the UNIX, Macintosh and Windows platforms. Under the Windows platform, the GBK character set is supported; this is a superset of GB2312-80. If any character in the GBK character set is used in FrameMaker, it will be displayed as nonsense in FrameMaker under UNIX and Macintosh. During translation, you will have access to the GBK character set and there is no way of preventing their insertion into the target STF file. When you verify the STF file(s), a warning message is generated, alerting you to the fact that the STF file contains unsupported characters.

Traditional Chinese

FrameMaker uses the BIG5 character set on the Macintosh and Windows platform. It uses EUC-CNS under UNIX. Under the Windows platform, there are Microsoft-specific extensions to the BIG5 character set called ETen. If any character in the ETen character set is used in FrameMaker, it will be displayed as nonsense in FrameMaker under UNIX and Macintosh. During translation, you will have access to the ETen extensions and there is no way of preventing their insertion into the target STF file. When you verify the STF file(s), a warning message is generated, alerting you to the fact that the STF file contains unsupported characters.

Korean

FrameMaker supports the basic KSC5601-1992 character set on all platforms. The Windows platform supports the full Johab character set, but FrameMaker does not. During translation, you will have access to the full Johab character set and there is no way of preventing their insertion into

the target STF file. When you verify the STF file(s), a warning message is generated, alerting you to the fact that the STF file contains unsupported characters.

Index Marker Sort Levels

You may need to add additional sorting information to markers when working with Asian languages, for example, Japanese. As Japanese is sorted phonetically, translation into the Japanese language requires that every index entry and index level has a corresponding sort string and sort level.

When working with Japanese, you may have to:

- ❑ add a sort string or sort level to markers
- ❑ adjust markers containing both Asian and Western text
- ❑ change the source sort order.

Adding a Sort String or a Sort Level to Markers

Where there is no sort string or sort level in the source text, you need to add them when translating into Japanese. Adding the internal tag `<:so>` allows you to assign sort levels to index markers.

In this example, the source index marker has no sort string or sort levels:

```
<imk 1>
<ie>File
<il>Open
</imk>
```

The index entry “File” has “Open” as an index level. To translate this into Japanese, you must insert a sort string. To insert this sort string, you add the internal tag `<:so>` as shown in the following example:

```
<imk 1>
<ie>ファイル<:so>ふぁいる
<il>開<<:so>ひら<
</imk>
```

By adding the internal tag `<:so>`, the sort information is stored with the translation in the translation memory. When you convert the STF file to MIF, S-Tagger preprocesses the marker and inserts the sort string.

 **NOTE**

S-Tagger only allows you to insert one `<:so>` tag per index entry or index level.

Adjusting Markers Containing both Western and Asian Text

Where markers contain both Western and Asian text, you can add a sort string for the Asian text.

For example:

```
<imk 3>  
<ie>Windows 2000  
<il>installing  
</imk>
```

You do not wish to translate Windows 2000, but you do require a sort string for “installing”. Add the internal tag `<:so>` as follows:

```
<imk 3>  
<ie> Windows 2000  
<il>インストール<:so>いんすとーる  
</imk>
```

S-Tagger preprocesses the text when converting from STF to MIF. The text “Windows 2000” becomes the sort string and the Japanese translation for “installing” becomes a sort level underneath Windows 2000.

 **NOTE**

The S-Tagger verification feature checks the length of the marker and generates a warning if it is too long. When calculating the length of the marker, this feature only considers text which should be included in the marker during conversion from STF to MIF.

Changing the Source Sort Order

If a marker in FrameMaker contains sort strings and sort levels, these will be presented using `<ss>` and `<sl>` tags in STF. When translating into Japanese, you will replace the source sort string and sort level with their Asian equivalent.

The following example illustrates how S-Tagger processes this type of index marker:

```
<imk 2>
<ie>Graphics
<il>inserting
<il>Deleting
<ss>Graphics
<sl>inserting
<sl>deleting
</imk>
```

When translating into an Asian language, you will need to modify the sort strings and sort levels to suit the translation. Do this by inserting the <:so> tag. Using this process, the preceding example becomes:

```
<imk 2>
<ie>画像<:so>がぞう
<il>挿入<:so>そうこゆう
<il>削除<:so>さくじょ
<ss>Graphics
<sl>inserting
<sl>deleting
</imk>
```

The <:so> tags will override the existing sort strings and sort levels. Before S-Tagger converts the STF file to MIF, it will preprocess the marker and replace the original sort string and sort order tags with the <:so> tags you have inserted. This preprocessing occurs internally in S-Tagger and is not visible to you. FrameMaker reads the index marker as if it had appeared as follows:

```
<imk 2>
<ie>画像
<il>挿入
<il>削除
<ss>がぞう
<sl>そうこゆう
<sl>さくじょ
</imk>
```

**NOTE**

During translation, where a source sort order is modified by using the `<:so>` tags, you can delete or ignore the source `<ss>` and `<s1>` tags in STF. If you delete the tags, S-Tagger will recreate the translated sort order from the `<:so>` tags during the STF to MIF conversion. If you ignore the `<ss>` and `<s1>` tags in STF, S-Tagger will overwrite them in the translated MIF file with the correct translated sort order.



S-TAG VERIFICATION AND CLEAN UP

This chapter explains the verification process and the updating of the translation memory during clean up.

Chapter sections include:

- ❑ verifying translated STF files
- ❑ updating the translation memory.

Chapter

5

VERIFYING TRANSLATED STF FILES

STF files can be verified using S-Tagger and/or TagEditor. This section describes the S-Tagger verification feature in detail; for more information on verification in TagEditor, see the *Translator's Workbench User Guide*.

Verification Guidelines

This section outlines some general guidelines for verification.

- ❑ Do not forget to include any ancillary files in the verification process.
- ❑ Run the file(s) through the S-Tagger verification feature again after fixing any errors, alerts or warnings that were generated the first time round.
- ❑ Continue verifying the file(s) until you are satisfied that any remaining alerts or warnings refer to changes which are intentional.
- ❑ Each time you run the file(s) through verification, save the CMP file with a new extension (for example, *.cm1 or *.cm2) to ensure the previous files are not overwritten.
- ❑ Save the last CMP files which show alerts or warnings about tag changes that you have examined and approved. Deliver these files with the finished translation, to facilitate smoother processing by technical and DTP personnel.

About Verification

Tag verification compares the tag content of the target STF files with the tag content of the source STF files and identifies any changes that were made. Changes in the target STF are acceptable provided that the syntax of the tags remains intact; if the tags are manipulated correctly, conversion back to MIF format will always be successful.

- ❑ If you are using TagEditor as your editing environment, you can verify during and after translation using the S-Tag Verifier plug-in.
- ❑ If you are using Word as your editing environment, you must use S-Tagger to verify your STF files.

The S-Tagger verification feature is particularly powerful when used as a batch tool. It can also be used as an additional QA step after verification in TagEditor.

During translation, it is often necessary to add tags to the translated text that were not present in the source document. For example, bold formatting is used more frequently in German texts than in English, so a German translation may contain more instances of bold text than its English source.



FOR MORE INFORMATION

For more information, see “Working with S-Tags” on page 4-4.

Similarly, you may have to manipulate tags to suit your translation. Verification detects every added or deleted tag.

Some tag changes do not prevent the STF file from being converted back to MIF; in such instances, S-Tagger generates alerts and warnings which inform you of such changes. It is recommended that you examine the alerts and warnings in order to determine which changes were intentional.

Verification in TagEditor

You can verify tags in TagEditor at segment or document level.

- ❑ segment level verification – during interactive translation, TagEditor automatically verifies the number, names, and order of internal tags in each target segment that you send to the translation memory. If there are changes, a message is returned.
- ❑ document level verification – TagEditor allows you to verify the tag content of whole documents, whether partially or fully translated, using the S-Tag Verifier plug-in. Document level verification looks at the internal and external tag content of the tag material and identifies any changes that have been made.

Checking your Conversion Settings

Before verifying the S-Tags, you must check that the current conversion settings in S-Tagger are correct.

To check your settings:

- 1 In S-Tagger, open the **Settings** tab.
- 2 Ensure that the **STF file format** option has been set to the correct format for the files you are working with, that is, **TRADOSTag (TTX)** for working with TRADOSTag files, **Rich text format RTF** for RTF files, **Text only (TXT)** for ANSI text files.
- 3 Check the STF file name extension setting is correct.
- 4 Change the **Target language** setting in the **Document language** group box to the language your file(s) has been translated into.

- 5 If either the source or the target language is an Asian language, set the character set to **Symbol** before verifying.
- 6 Open the **Paths** tab and ensure that the paths are still correct. If they are not correct, click **Browse** to point S-Tagger to the correct folders.

You do not need to change any other settings, except the settings in the verifier report described below.

Verifier Report

You can access the **Customise Verifier Report** dialog box through the verifier report **Customise** command button on the **Settings** tab. It contains a list of tags for which you can suppress the alert or warning that S-Tagger would otherwise generate during the verification process.

These alerts and warnings are displayed in the **Results** window on the **Verify S-Tags** tab and in the CMP file in more detail.

The CMP file is a comparison file that logs certain faults in the file being verified. This file is saved to the *Target* folder (the same folder as the files for verification), with the same base name as the file being verified and with the extension * .cmp. The CMP file lists the number and type of errors, alerts or warnings found in the file being verified.

Before running the verification process, you can determine which of the customisable alerts and warnings you wish to suppress. The S-Tagger verification feature detects these changes, but does not include detailed information about them in the CMP file. A list of all suppressed alerts and warnings appears at the end of the CMP file. For more information on using the CMP file, see “Using the CMP File” on page 5-10.

The effect of each of these report messages is described in the following table:

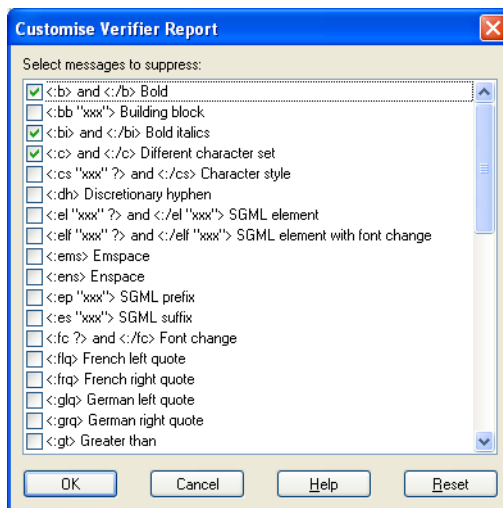
Message Type	Effect on Conversion
Error	Prevents an STF file from being converted to MIF
Alert	Does not prevent the STF file from being converted to MIF but may have a significant impact on the layout of the translated MIF file
Warning	Highlights tags that have been manipulated (added, deleted, or moved) during translation

Setting Verification Message Preferences

In the **Customise Verifier Report** dialog box, tags are divided into external and internal tag sections. Within each section, tags are listed in alphabetical order.

To open the **Customise Verifier Report** dialog box:

- 1 In S-Tagger, open the **Settings** tab.
- 2 Click **Customise**. The **Customise Verifier Report** dialog box is displayed.



- 3 Scroll down through the list and select the warnings and alerts you wish to suppress by checking the appropriate box. The messages that correspond to the tags that you select are suppressed.

For example, if you have translated an STF file from English into French and have substituted French smart quotes for English smart quotes, you may wish to suppress all warnings and alerts in relation to deleting English smart quotes and inserting French smart quotes. In this case, you would select English smart quote tags (<:ldq>, <:rdq>) and French smart quote tags (<:flq>, <:frq>).

By default, the following tags are selected:

Tag	Represents
<:b> and <:/b>	Bold
<:bi> and <:/bi>	Bold italics
<:c> and <:/c>	Different character sets
<:i> and <:/i>	Italics
<:s> and <:/s>	Small caps

A list of suppressed messages will appear at the end of the CMP file that is produced during verification. The number of suppressed alerts and warnings is also included in the **Results** window.

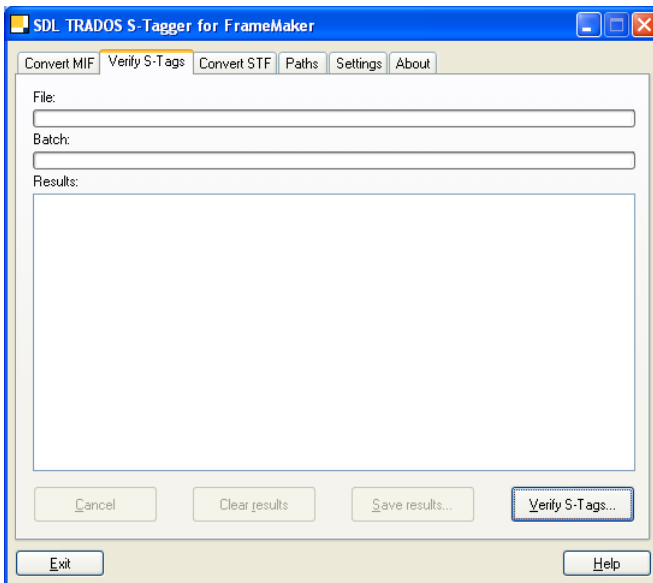
Now that you have set your verification preferences, you may proceed to verifying the S-Tags.

Verifying the S-Tags

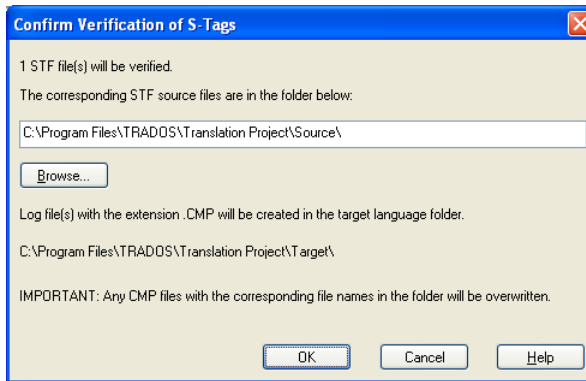
This section provides an overview of the verification process.

To verify the tags in your target STF file:

- 1 In S-Tagger, open the **Verify S-Tags** tab.

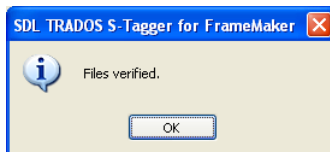


- 2 Click the **Verify S-Tags** command button and select the target STF files you wish to verify and click **Open**. A message box similar to the one below appears, prompting you to confirm that the path to the folder in which the source (untranslated) STF files are kept is correct:

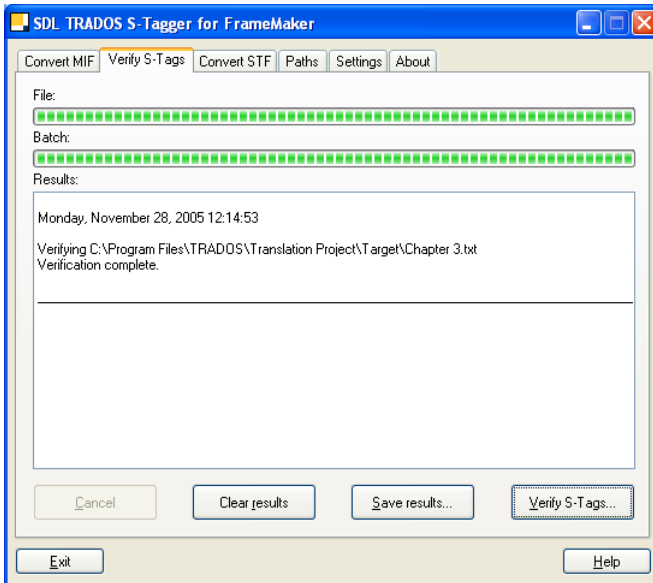


- 3 If the path that S-Tagger has listed here is not correct, click the **Browse** button and locate the correct directory.

During the verification process, the outcome of the comparison is written to the **Results** window. When the verification is complete, a message box similar to the one below appears. In this example, no errors were found.



The **Results** window also shows the results of the comparison.



- 4 If you want to save the text in the **Results** window to a log file, you can do this by clicking **Save Results**. A dialog box is displayed, prompting you to specify the name of the file the results should be saved to. The log file is automatically given the extension `*.txt` and can be opened in any text editor.

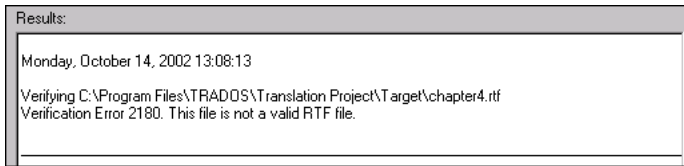
File Errors

File errors are only reported in the **Results** window, and a CMP file will not be generated. All other errors are reported in the CMP file.

These errors occur when a file cannot be opened or closed, or if a file has not been saved in the correct format. When a file error is encountered, a CMP file is not created for the file; the error message is written in the **Results** window.

In the following example, an RTF file had been saved as a `*.doc` file in Word by mistake. The user then changed the extension back to `*.rtf` in Windows Explorer to try and correct the mistake.

When the user attempted to verify the file, the following message was displayed in the **Results** window.



To correct this error:

- 1 Open the file in Microsoft Word and save it as RTF, making sure it has the extension * .rtf.
- 2 Verify the file again.

S-Tag Verification Messages

As well as the messages that appear in the **Results** window, a more detailed account of the tag changes is written to the CMP file. If there are no errors, alerts or warnings are generated (or if there are only file errors), no CMP file is created. Messages in the CMP file are usually clear and self-explanatory. If you do not understand a message, refer to the S-Tagger for FrameMaker Help.

There are four categories of messages:

- comparison errors
- tag errors
- tag alerts
- tag warnings.

In addition to comparison and tag errors, the verification feature informs you about other differences between the tags in the source file and those in the target file. These tag differences are categorised as tag alerts or tag warnings.

If you use TagEditor for translation, you will avoid some of the verification messages described in this section. This is because TagEditor's unique tag protection safeguards against issues that can easily occur if you have chosen Word as your editing environment.

NOTE

If you find any errors, alerts or warnings in the CMP file which you then fix in the STF file, you should save the CMP file with another extension, for example, * .cm1 as the original CMP file may be overwritten at a later stage.

Using the CMP File

A CMP file is created for each STF file that generates a verification message. The CMP file has the same file name as the respective STF file but with a *.cmp file extension. When correcting any verification errors, alert, or warnings, open the CMP file and its corresponding STF file in the editing environment that you are using for translation.

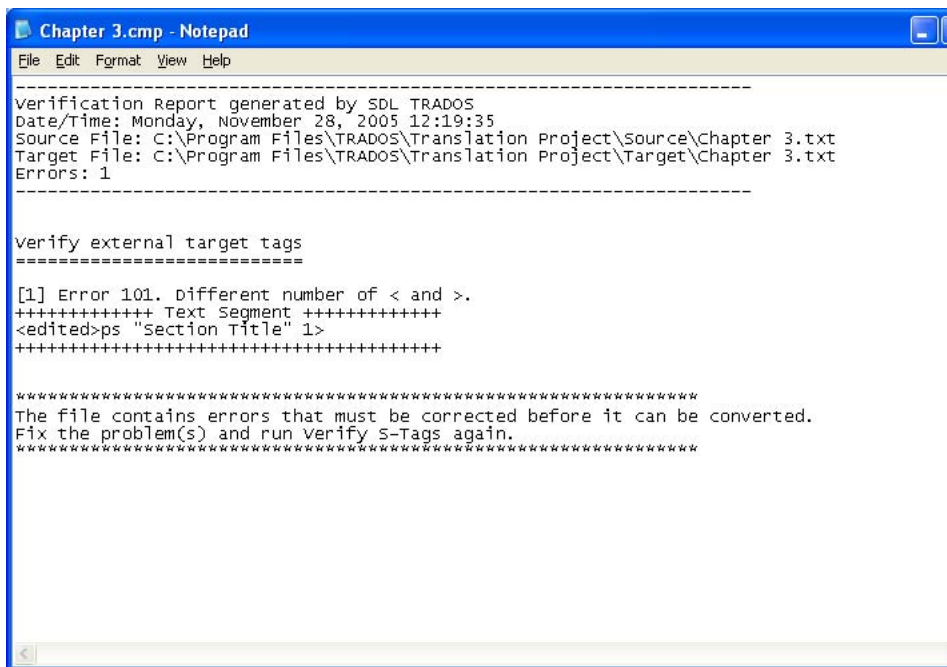
To use the CMP file:

- 1 Open the CMP file from the *Target* folder in the same editing environment as you have chosen for translating or editing the files.

Open the CMP file in Word or Notepad.

- 2 Open the STF file in a second window in your editing environment.

The CMP file always starts with a header message which date/time stamps the file. The date/time stamp is followed by a reference to the source and target files being verified and lists the total number of errors, alerts, warnings and suppressed alerts and warnings found in the file. The header message should be similar to the one below:



```

Chapter 3.cmp - Notepad
File Edit Format View Help
-----
Verification Report generated by SDL TRADOS
Date/Time: Monday, November 28, 2005 12:19:35
Source File: C:\Program Files\TRADOS\Translation Project\Source\Chapter 3.txt
Target File: C:\Program Files\TRADOS\Translation Project\Target\Chapter 3.txt
Errors: 1
-----

Verify external target tags
=====

[1] Error 101. Different number of < and >.
+++++++ Text Segment ++++++++
<edited>ps "Section Title" 1>
+++++++

*****
The file contains errors that must be corrected before it can be converted.
Fix the problem(s) and run Verify S-Tags again.
*****

```

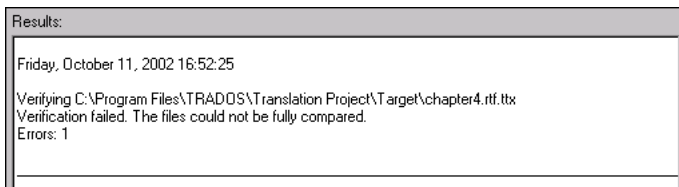
The header message is followed by a report of the procedures that S-Tagger has gone through. If an error, alert or warning is found during any of the procedures, it is listed in the section for that procedure. This can lead to the same error, alert or warning being reported several times.

Comparison Errors

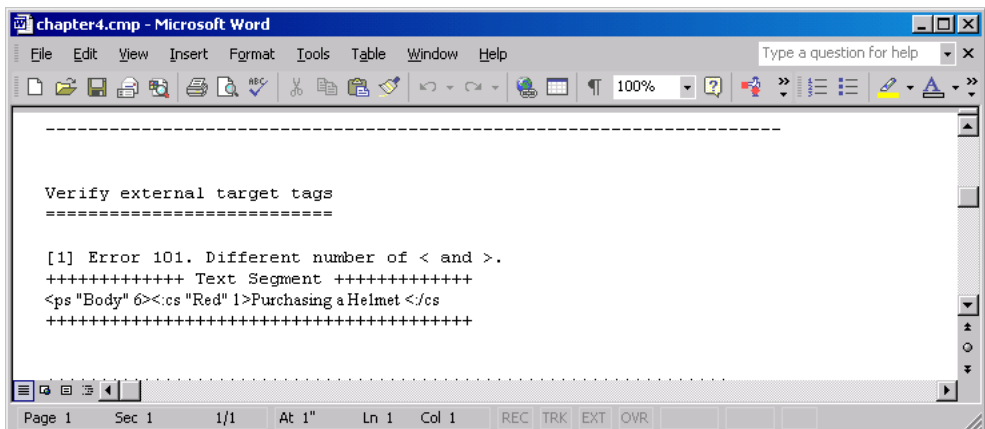
These are serious errors which prevent the files from being fully compared. In most cases, a CMP file is created, which details the errors found.

In the following example, a ‘>’ symbol was deleted accidentally. Note that this specific problem cannot occur if you use TagEditor for translation because TagEditor does not allow you to delete single characters from a tag. This example would only occur in a Word editing environment.

In the **Results** window, you should see a message similar to the one below for this file:



In the CMP file, you should see a message similar to the one below:



In this example, a ‘>’ sign has been deleted. This was the closing bracket for the internal tag < : /cs >.

To correct the error:

- 1 Open the target STF file in Word.
- 2 Insert a ‘>’ sign after the < : /cs statement in the target language RTF file.
- 3 Save the file and close it. Make sure that you close the CMP file as well.
- 4 In S-Tagger, run the STF file through the verification feature again.

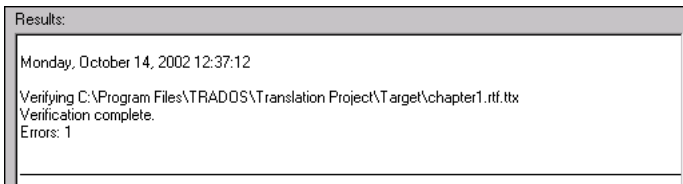
Tag Errors

Tag errors usually occur because an external tag has been added, deleted or changed. They can also occur because the position of some important tag has been changed. Adding or deleting some internal tags can also cause tag errors. There are strict rules for adding and deleting tags. For more information, see “Working with S-Tags” on page 4-4. These rules must be followed when adding or deleting tags in the translation.

If a tag error occurs in your file, it is not possible to convert it back to MIF. All errors must be corrected. When you have corrected the error, the files must be saved, closed, and verified again.

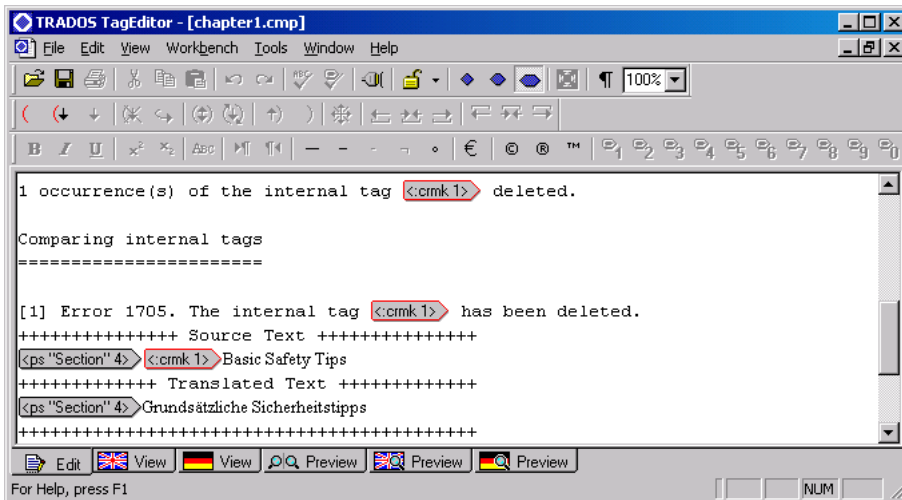
In the following example, an internal tag `<:crmk 1>` (representing a cross-reference marker) has been deleted accidentally. Note that this can happen frequently in Word. In TagEditor, this can only happen if tag protection has been turned off.

If this happens, the message in the **Results** window for this file will be similar to the one below:



In this instance, one error has been reported. The `<:crmk ?>` tags may never be deleted, which is why the level is error, rather than alert or warning.

Open the CMP file in TagEditor or Word (depending on your editing environment) and you will see a message similar to the following:



The error message shows you what the segment looks like in the source STF file and in the target STF file.

To correct the error:

- 1 Keep the CMP file open and open the relevant target file in TagEditor or Word.
- 2 Find the relevant text in the STF file (using TagEditor or Word's Find feature) and insert the correct tag, either from the CMP file or from the source STF file, into the translated STF file.
- 3 Save the file and close all open files.
- 4 In S-Tagger, verify the file again.



NOTE

Some Asian characters which are available to translators are not supported in FrameMaker. These characters exist for Japanese, Korean and both Simplified and Traditional Chinese. If an unsupported character is inserted during translation, S-Tagger will detect the character and you will receive a message when you verify the files. For more information, see "Unsupported Characters" on page 6-11.

Tag Alerts

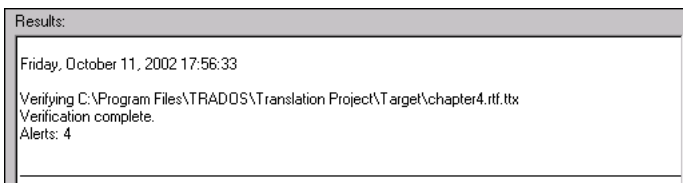
Tag alert messages refer to differences in the files which may have serious consequences for the file itself, but which will not prevent the file from being converted to MIF. Alerts also serve as higher level warnings for tag differences that may be intentional.

In this example, we want to delete an index marker. Note that you must turn off the document and the tag protection in TagEditor to delete such tags. For more information on document and tag protection in Tag Editor, see the *Translator's Workbench User Guide*.

To delete the index marker:

- 1 Open the target STF file in TagEditor or Word.
- 2 Delete the opening `<imk ?>` tag, each `<ie>` and `<il>` tag and the text of the index entry.
- 3 Delete any `<ss>` and `<sl>` tags, as well as the closing `</imk>` tag. If you wish to delete an index subentry, delete the relevant `<il>` tag and text only.
- 4 In S-Tagger, verify the file from which the marker was deleted.

The message in the **Results** window will be similar to the one below:



You can see from this **Results** window that there are four alerts reported. Consult the CMP file for full details of these alerts.

In the CMP file, the first section you encounter is the “External tags report”. This gives you summary information about which tags have been added or deleted. The external tags report will look similar to that shown below:

```
External tags report
=====

Errors reported in this section may correspond to problems reported below.
Correct the problems, using the information in this section as a reference, and
run Verify S-Tags again.

1 occurrence(s) of the external tag <imk 1> deleted.
2 occurrence(s) of the external tag <le> deleted.
1 occurrence(s) of the external tag </imk> deleted.
```

The next section in the CMP file, “Comparing External Tags”, lists the missing tags that have been detected. The section of the source STF that contains the relevant tags is shown, as is the section in the target STF from which the tags are missing. This helps you to quickly identify the issue and assess whether you need to fix anything.

You should see a message similar to the one below for each `<imk ?>`, `<ie>`, and `</imk>` tag deleted:

```
Assuming 4 external tag(s) deleted, none added.

[1] Alert 1511. The external tag <imk 1> has been deleted.
+++++++ Source Text ++++++
<pr "4 " 1>Bicycling safely<imk 1>
<imk 1>
<ie>Cycling safety
+++++++ Translated Text ++++++
<pr "4 " 1>Sicher Rad fahren
</tr>
+++++++
```

If the index marker was deleted in error, follow these steps to replace it:

- 1 Keep the CMP file open and open the relevant source STF file in TagEditor or Word.
- 2 Find the tag `<imk 1>`.
- 3 Copy the section from `<imk 1>` to `</imk>`.
- 4 Open the target STF file in the same editing environment as the CMP file.
- 5 Find the text where the marker should have been.
- 6 Insert a carriage return after the end of the last paragraph or tag.
- 7 Position the cursor on the empty line.
- 8 Paste the text from the source STF file into the target STF file.
- 9 Save the target STF file.
- 10 Translate the text of the index marker.
- 11 Close all files.
- 12 In S-Tagger, verify the file from which the marker was deleted.

Tag Warnings

These refer to tag differences between the files which usually relate to formatting or punctuation. Warnings can normally be ignored while the file is being converted, but a list of the warnings helps you to find formatting changes quickly.

There will often be many warnings in a translated file, as the formatting of the translation will probably be different to the formatting of the source language.

Two of the most common warnings are:

- ❑ Warning 1703. The internal tag `<:tag>` has been deleted.
- ❑ Warning 1707. The internal tag `<:tag>` has been added.

These warnings may appear quite often as you adapt the formatting of the text to fit the translation. If you add a `<:b>`, `<:i>`, `<:bi>`, `<:c>`, `<:c1>`, `<:c2>` or `<:s>` tag, you will not be informed of this, unless you have forgotten to add the closing tag or have turned on the reporting of these tag changes in the **Customise Verifier Report** dialog box.

You should check all warnings to ensure that the format change is intended. Include the CMP file, complete with suppressed alerts and warnings, with the delivery of the translated STF files.



NOTE

You can find a complete list of the alerts and warnings in the S-Tagger for FrameMaker Help.

Sorted Files

The content of translated files may be sorted in a different manner to the source files. This often occurs with glossary files. For example, the file could be a glossary or other alphabetically sorted list. This is allowed.

S-Tagger will not be able to compare the tags against the source file, as many paragraphs will be in different positions. However, it will verify that the tags are valid tags and that they appear in the correct order and in the right position, so that the files can be converted to MIF.

You may get alerts and warnings when you have sorted a file, even if there is nothing actually wrong with the positioning of the tags.



WARNING

You should never try and sort FrameMaker+SGML files.

Tag Formatting and Translator's Workbench

Some errors in the formatting of tags may be introduced by first-time users of Translator's Workbench. Most of these errors will be picked up by S-Tagger.

A common error occurs when the target text has been accidentally formatted as hidden text when working with Word and Translator's Workbench. Note that such a scenario does not occur if you are using TagEditor.

The example below illustrates a hidden text error that occurred using Word:

```
<ps "Body" *7> {0} With offices throughout the world, and a <:cs "TRADOS" *
2>TRADOS<:/cs> product on the desk of more than 20,000 professional translators, <:cs *
"TRADOS" *2>TRADOS<:/cs> is recognised as one of the market leaders in translation
technology. <0> With offices throughout the world, and a <:cs "TRADOS" *2>TRADOS<:/cs>
product on the desk of more than 20,000 professional translators, <:cs "TRADOS" *
2>TRADOS<:/cs> is recognised as one of the market leaders in translation technology. <0> ¶
```

Here you can see that the first piece of text in the file is the sentence which starts with "With offices throughout the world...". The source text is formatted as hidden text as indicated by the dotted underline.

If the *Hidden* attribute was accidentally applied to the target segment as well when the file was verified, a message similar to the one below would appear:

```
[1] · Warning · 1703 · The internal tag <:cs "TRADOS" *2> has been deleted. ¶
[2] · Warning · 1703 · The internal tag <:/cs> has been deleted. ¶
[3] · Warning · 1703 · The internal tag <:cs "TRADOS" *2> has been deleted. ¶
[4] · Warning · 1703 · The internal tag <:/cs> has been deleted. ¶
[5] · Warning · 1703 · The internal tag <:cs "TRADOS" *2> has been deleted. ¶
[6] · Warning · 1703 · The internal tag <:/cs> has been deleted. ¶
[7] · Warning · 1703 · The internal tag <:cs "TRADOS" *2> has been deleted. ¶
[8] · Warning · 1703 · The internal tag <:/cs> has been deleted. ¶
+++++ Source Text +++++¶
<ps "Body" *7> With offices throughout the world, and a <:cs "TRADOS" *
2>TRADOS<:/cs> product on the desk of more than 20,000 professional translators, <:cs *
"TRADOS" *2>TRADOS<:/cs> is recognised as one of the market leaders in translation
technology. <:cs "TRADOS" *2>TRADOS<:/cs> customers include such diverse organisations as
Microsoft, Oracle, Berlitz, Nortel Networks, SAP, Siemens, PeopleSoft and Parametric Technology
Corporation. Customers from the government and non-profit sectors include Amnesty International,
The International Monetary Fund, INTELSAT, European Investment Bank, the European Parliament,
the World Intellectual Property Organization, NATO and the Zollkriminalamt. More than 150
universities, from Finland to Chile, also use <:cs "TRADOS" *2>TRADOS<:/cs> products. ¶
+++++ Translated Text +++++¶
<ps "Body" *7> ¶
+++++
```

You can tell very quickly what has happened here, because the paragraph style tag in the translated text section is the same, but it is not followed by any text or tags.

To fix this error:

- 1 Open the target STF file in TagEditor or Word and check that non-printing characters are showing.
- 2 Remove the hidden formatting from the translated segment.



FOR MORE INFORMATION

Where an error occurs repeatedly and you cannot understand why, consult the online support centre at support.sdl.com.

UPDATING THE TRANSLATION MEMORY

After translation and verification, your target STF files are still in bilingual format. You must ensure that all the bilingual data that currently exists in each STF file is transferred to your translation memory.

To save your work to the translation memory, use the **Clean Up** command in Translator's Workbench. This command is available from the **Tools** menu. Before using this feature, make sure that the **Update TM** option is selected.

When working with S-Tagger, the **Clean Up** command is only used to update the translation memory. This is because you can perform backward conversion on bilingual as well as monolingual files using S-Tagger.

The **Clean Up** command also modifies the bilingual file in the following ways:

- ❑ It removes hidden source text and segment delimiting marks from the target STF files.
- ❑ It restores the original colouring of your text; this only applies if you have used the **Translated Text Colours** option in Translator's Workbench when using Word.
- ❑ It changes the file extension to RTF.

Translator's Workbench produces a monolingual RTF file after clean up. The production of the RTF is essentially a by-product of the translation memory update. If you have begun the translation workflow with a TRADOStag (TTX) file, then you should retain the most recent copy of this file before clean up.

You can use the monolingual RTF or the bilingual TTX file for the next workflow event, conversion back to the original file format (MIF). If you began your workflow with an RTF, we recommend that you use the monolingual target RTF file. However, if you began your workflow with a TRADOStag (TTX) file, we recommend that you use the bilingual TTX files for conversion back to MIF.



FOR MORE INFORMATION

- ❑ For more information on workflows, see "S-Tagger Workflow: An Overview" on page 2-10.
- ❑ For more information on cleaning your files, see the *Translator's Workbench User Guide* and the TagEditor Help.



STF TO MIF CONVERSION AND POST-PRODUCTION

This chapter explains the conversion and post-production tasks for translated FrameMaker files.

Chapter sections include:

- ❑ conversion to original format
- ❑ general post-production tasks
- ❑ Asian and East European language post-production.

Chapter

6

CONVERTING STF TO MIF

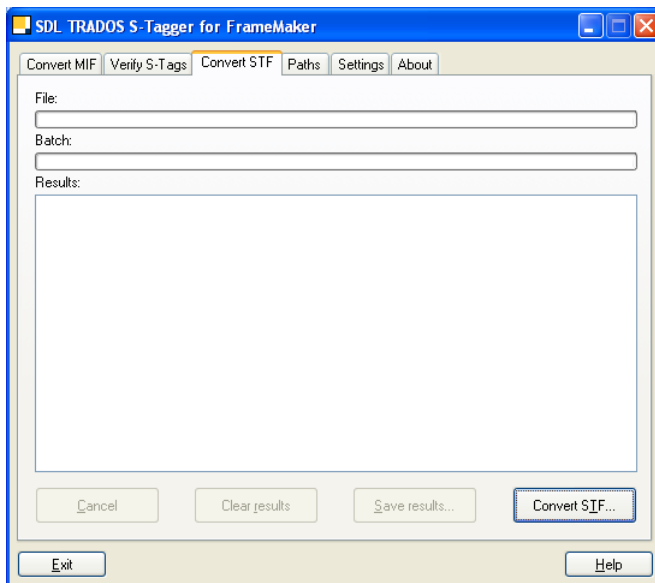
After translating and verifying your files, you can now convert them back to MIF format. This section describes how to use S-Tagger to convert the STF files, singly or in batches.

Converting STF to MIFMIF will only be successful if you have:

- ❑ corrected all the errors in the files noted during the verification process
- ❑ retained the original source STF and ORG files in their specified folder.

To convert your STF files to MIF:

- 1 In S-Tagger, open the **Convert STF** tab.



- 2 Click the **Convert STF** command button under the **Results** window.

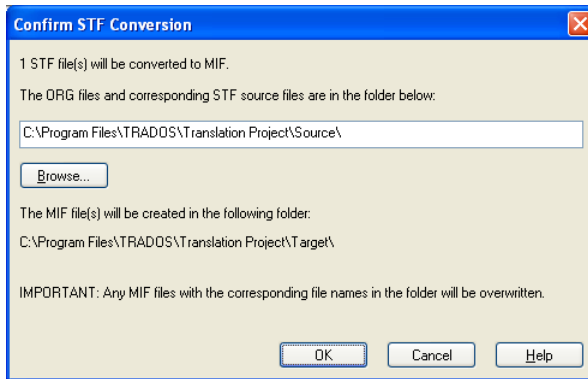
The **Select STF File(s) to Convert** dialog box is displayed, prompting you to select the translated STF file you wish to convert to MIF. Note that the type of file displayed in the dialog box depends on the **STF file format** option selected on the **Settings** tab. For example, if you have selected the **TRADOStag (TTX)** option, only TTX files are displayed here.

- 3 Select the file(s) to convert and click **Open**.

→ NOTE

You do not need to select the ancillary file as S-Tagger automatically inserts the text from this file into the correct files.

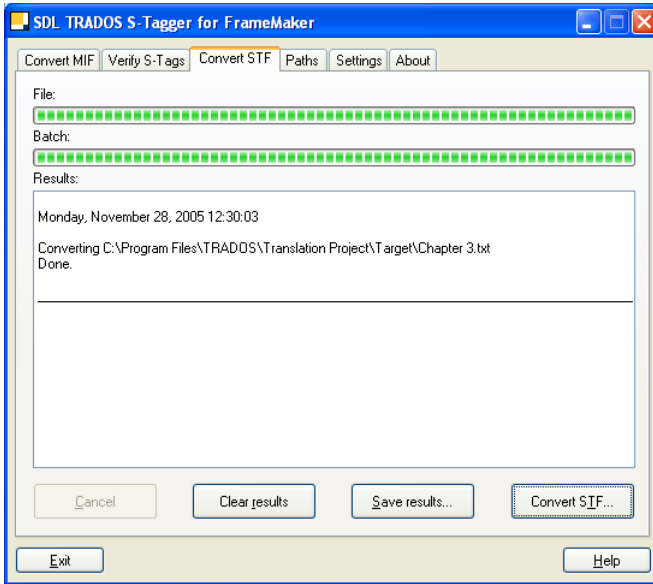
The **Confirm STF Conversion** dialog box is displayed, prompting you to confirm the file(s) to be converted to MIF and detailing the path where it expects to find the ORG files and the original source STF files.



If you have selected the **Keep Paths** option on the **Paths** tab, S-Tagger automatically finds the folder where it last located STF and ORG files. The **Keep Paths** settings information is not project-specific, but relates to the last use of the software.

- 4 If you wish to change the path that S-Tagger should follow to find the ORG files, click **Browse**. Locate the source STF and ORG files and click **OK** to start the conversion.

As the files are being converted, an analysis of the process appears in the **Results** window, similar to the one below:



- 5 If you want to save the text in the **Results** window to a log file, you can do this by clicking **Save Results**. A dialog box is displayed, prompting you to specify the name of the file the results should be saved to. The log file is automatically given the extension `*.txt` and can be opened in any text editor. By default, this file is saved in the same folder (*Target*) as the converted MIF files.

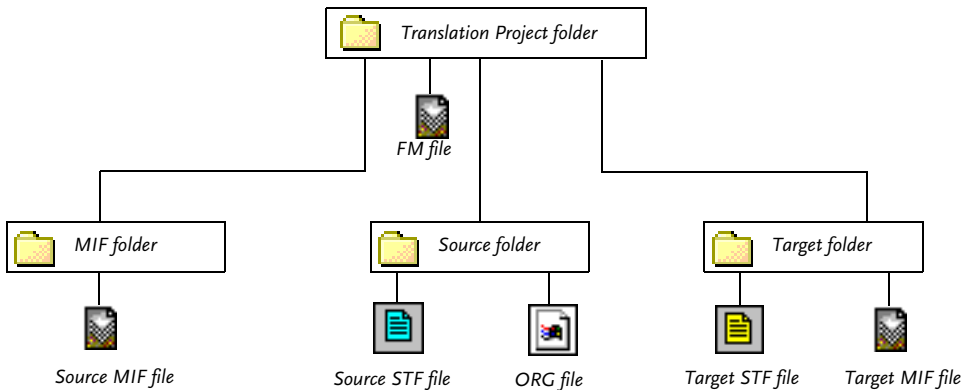
Correcting STF to MIF Conversion Errors

During conversion, errors may appear in the **Results** window. An error occurs when S-Tags have been moved or changed in a way that renders the converted MIF file invalid. For example, errors will occur if a closing tag has been deleted or if a tag has been invented and inserted into the STF file.

If an error occurs, run the verification feature to identify and locate the error before attempting to convert the STF files to MIF again. For more information on verification, see “Verifying the S-Tags” on page 5-6.

If no errors occur during the conversion process, the files are converted and appear as MIF files in the *Target* folder.

See the diagram below and compare it to the one provided in “Setting up the Project Structure” on page 3-3. Note that the translated, converted MIF files have now been added to the *Target* folder.



NOTE

- ❑ There may be an additional ancillary file in the *Source* and *Target* folders if you selected the **Create ancillary file** option on the **Settings** tab in S-Tagger.
- ❑ Your *Target* folder should also contain any subfolders, for example, *Art* (containing all the translated graphics), to replicate the structure of the translation project folder. This will ensure that all the references in the target files will function correctly.

After converting the files successfully to MIF, you should open the MIF file to check for any errors.

GENERAL POST-PRODUCTION

This section guides you through opening, checking, and saving the MIF file(s) after conversion from STF. The following is a high-level overview of the steps that you must complete:

- 1 Open the file in FrameMaker to check the error messages which may be displayed in the **FrameMaker Console** window.
- 2 When you have checked these error messages, you should then thoroughly examine each of the MIF files, using the checklist provided in “Checking the New MIF File” on page 6-8.
- 3 After checking the MIF files, you can then save them back to FM format.

Opening the New MIF File

Because S-Tagger for FrameMaker is a Windows application, it is recommended that, where possible, the new MIF file is opened in FrameMaker for Windows. Opening the new MIF file in FrameMaker for Windows rather than on any other platform reduces the amount of time needed for bug-fixing.

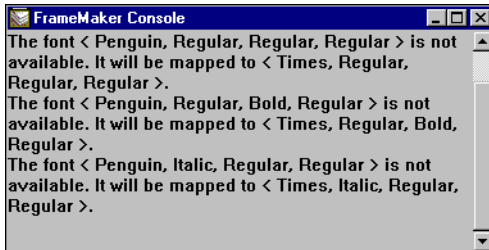
This section is written for users working in FrameMaker for Windows. If you are opening the files in FrameMaker for Macintosh or UNIX, the illustrations and instructions on the following pages may not be relevant to you.

The following is a list of common issues found in the newly translated MIF files:

- missing fonts
- missing languages
- unresolved cross-references
- S-Tagger character formats.

Missing Fonts

Missing font errors are due to missing fonts on your system, in which case the message will be similar to the one below. This message means that you need to locate and install the document fonts onto your system.



To correct this error, install the relevant fonts on your system and open the MIF file again. If you are in any doubt about missing fonts, consult your FrameMaker documentation or seek advice from qualified FrameMaker technical support staff.

Missing Languages

You may be alerted to the fact that the document calls for a language dictionary different to the one installed with your copy of FrameMaker. In this case, you should check the **Settings** tab in S-Tagger to see what language you set as the **Document Target Language**. You should, however, consider installing the additional language dictionaries on your system if you intend opening the FrameMaker files after translation and after they have been converted back to MIF.

If you are in any doubt about missing languages, consult your FrameMaker documentation or seek advice from qualified FrameMaker technical support staff.

Unresolved External Cross-references

If your MIF file contains external cross-references, you may be alerted to unresolved cross-references because your files now have the extension `*.mif` and are therefore not recognised as the source files for the cross-references. You can safely ignore this message until you are at the stage in the conversion process where all files have been renamed.

If you still receive this message after renaming all the files with the correct extensions and placing them within the correct folder structure, consult your FrameMaker documentation.

S-Tagger Character Formats

S-Tagger adds the following character formats to the catalog of the translated STF file: *STagNormal*, *STagCourierNew* and *STagArial*. You can either delete these or convert them to the character format of your choice.



NOTE

If any other messages appear in the **FrameMaker Console** window, consult your FrameMaker documentation.

Checking the New MIF File

After you have read the FrameMaker error messages and made any necessary changes, you should also check the following issues:

Issue	What to Check
Text	Is all the required text present in the file? Is the page count similar to the original FM file?
Formatting	Does the formatting in the new MIF file reflect the formatting in the original FM file?
Special characters/symbols	Have all the special characters or symbols converted correctly?
Art files and text	Are the art files and accompanying text in the correct position in the MIF file?
Position of markers	Are the markers in the correct position? For more information on this issue, see “Position of Markers”.
Other features	Have you checked all the other features that you think might have caused problems in the file?

Position of Markers

If you selected **Don't insert** under **Marker placeholders** on the **Settings** tab in S-Tagger before converting the MIF file to STF, the location of the index markers changes in the translated MIF file. Open the new MIF file in FrameMaker with **Text Symbols** selected in the **View** menu. You will notice that each marker has been moved to the end of the paragraph in which it originally occurred.

If you are concerned about the position of markers and the effect this will have on a generated index, generate the index and compare it against the original FM index. If you find that an index marker refers to a paragraph which breaks over a page, make a note that the index marker should be moved

to another place in the paragraph when final pagination is being done. Remember that the pagination will probably change with translation.

**NOTE**

If you encounter any difficulties other than those listed above, consult your FrameMaker documentation.

Saving the MIF File to FM Format

You can now save the translated MIF files as FM files to the correct folder structure.

To convert a MIF file to FM:

- 1 Open the MIF file in FrameMaker and make sure that any graphic references are pointing to your translated art folder.
- 2 Select the **Save As** command from the **File** menu and save as a FM file.
- 3 Update the cross-references, complete DTP tidy-up and pagination.
- 4 Repeat steps 1-3 for each target MIF file in the FrameMaker book (if you are working with a book structure).
- 5 Generate the table of contents and the index.

**NOTE**

For more information on files that have been translated into an Asian language, see “Asian and East European Language Post-production” on page 6-10.

ASIAN AND EAST EUROPEAN LANGUAGE POST-PRODUCTION

This section contains general tips for post-translation production. It is intended specifically for users working with Asian or East European languages. After you have checked your MIF files for the issues outlined below, you can follow the steps in “Saving the MIF File to FM Format” on page 6-9.

The following issues are examined in this section:

- ❑ character set tags
- ❑ unsupported characters
- ❑ font mapping.

Character Set Tags

To retain Western extended characters in a translated MIF file, insert character set tags (< : c > and < : / c >) around the character or word.

Where you add the character set tag, additional character formats are added to the translated MIF file when you convert it from STF to MIF. Three character formats are added to the MIF file; *STagNormal*, *STagCourierNew*, and *STagArial*.

The following table summarises this character set tag information:

Character Set Tag	Font	Character Format in FrameMaker
< : c > and < : / c >	Times New Roman	<i>STagNormal</i>
< : c1 > and < : c1 >	Courier New	<i>STagCourierNew</i>
< : c2 > and < : / c2 >	Arial	<i>STagArial</i>

These are character styles created by S-Tagger to allow you to retain the characteristics of Western European extended characters (those ANSI characters above 127, such as ß or à) within the STF file.

For example, you can use the character set tags to mark up a product name that is not for translation. The character style is then applied to the text surrounded by these tags when it is converted from STF to MIF.

Even if you do not insert character set tags into the STF file, the *STagNormal*, *STagCourierNew* and *STagArial* character formats still appear in the FrameMaker character catalog in the translated MIF

file. The fonts are not applied to any text in the file. You may wish to delete them before completing DTP on the translated document.

**NOTE**

If you find instances of characters in a translated FrameMaker file that do not make sense, check if the characters should have the normal font encoding applied to them in the STF file. Insert the `<:c>` tags where necessary.

**WARNING**

If you are working with FrameMaker+SGML documents, you must map these formats to relevant elements in your DTD, otherwise the formatting will be lost when you save as SGML.

Unsupported Characters

Some Asian characters which are available to translators may not be supported in FrameMaker, for example, certain Japanese, Korean and both Simplified and Traditional Chinese characters. If an unsupported character is inserted during translation, S-Tagger detects the character and generates a message during verification.

Using Font Mapper for FrameMaker

Font Mapper for FrameMaker facilitates the process of changing fonts in FrameMaker MIF documents that are translated into or from Asian languages. Font Mapper for FrameMaker maps, or replaces, the fonts in the original document with fonts that you specify. Other attributes, such as font size and some styles, can also be changed. These changes are referred to as font mappings.

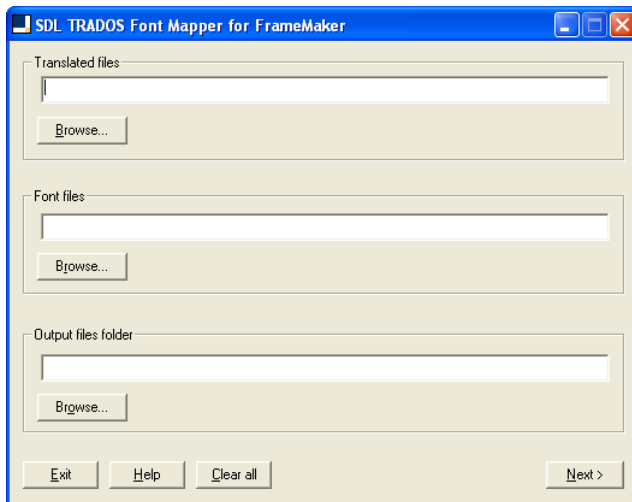
Using Font Mapper for FrameMaker you can:

- change the fonts in FrameMaker's two master catalogs where formats are defined, the Paragraph Catalog and the Character Catalog
- change fonts in any manual overrides made to paragraph and character formats
- map combined fonts – combined fonts are treated in the same way as any other font.

Mapping Fonts

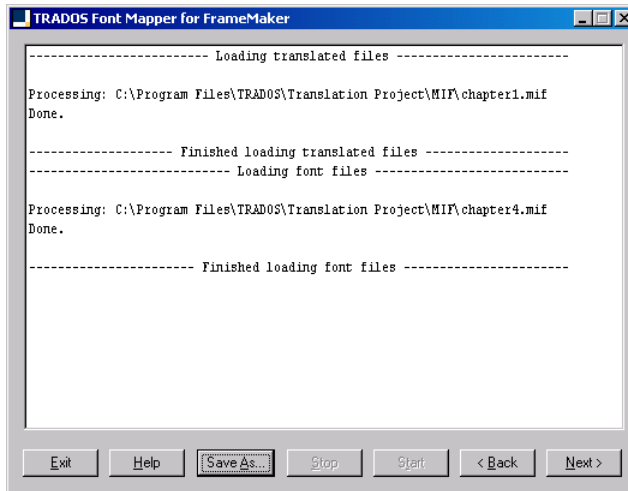
To map fonts:

- 1 Launch the Font Mapper for FrameMaker.

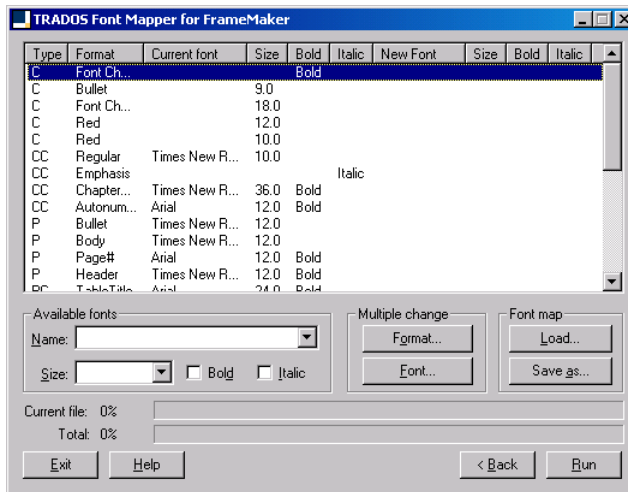


- 2 On this screen, specify the location of the following files and click **Next** to continue.
 - Under **Translated files**, browse to select the MIF files that require font changes.
 - Under **Font files**, browse to select the font files that contain examples of the fonts you wish to map. Font files may be either FrameMaker MIF files that contain the fonts you require or font map files that were created using Font Mapper for FrameMaker.
 - Under **Output files folder**, browse to select the output folder where Font Mapper will place the font mapped files. You can either select an existing folder or create a new one, specifically for that purpose.
- 3 On the second screen, click **Start**. Font Mapper loads and processes the files specified in the previous screen and records the results. The results are displayed on the screen.

- 4 Click **Next** to continue.

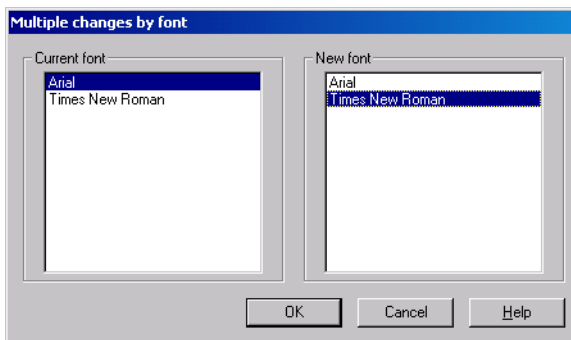


- 5 Use this third screen to specify the font mappings.



- ❑ The first two columns in the font mapping window show the types and formats in the translated files. There are two format types in FrameMaker: paragraph formats in the Paragraph Catalog (PC) and character formats in the Character Catalog (CC). Formats with manual overrides are indicated by 'P' and 'C', respectively.
- ❑ The next four columns in the window refer to the **Current Font** as used in the translated files. For each font type, the name of the font appears, together with its properties of **Size**, **Bold** and **Italic**.

- ❑ The last four columns in the window refer to the **New Font**. When a new font is specified, its name and properties of size, bold or italic are displayed in these columns.
- 6 You can now map the fonts globally or as they occur in the font mapping window.
- ❑ To map individual instances of a font, select the font or format that you wish to map from the font mapping window. Then select a font from the **Available Fonts** drop-down list. Specify a font size and face if required.
 - ❑ To map fonts in batches, for example to map all instances of a font to another font, click **Font** in the **Multiple Change** box. The **Multiple Changes by Font** dialog box is displayed.



Select the font that you wish to map from the **Current Font** list. Select the font that you wish to map to from the **New Font** list. Click **OK**. In this example, all instances of *Arial* will be mapped to *Times New Roman*.

- 7 When you have completed all your font mappings, click **Run**. Font Mapper creates a new set of MIF files in the output files folder. Open these files in FrameMaker.

FOR MORE INFORMATION

For more detailed information on Font Mapper for FrameMaker, see the Font Mapper Help.

You have completed mapping the fonts needed to display the translated MIF file in FrameMaker. You can now continue and save the MIF file to FM format.



SECTION 3:
S-TAGGER FOR INTERLEAF

PREPARATION AND IASCII TO STF CONVERSION

Section 3 deals exclusively with S-Tagger for Interleaf. This particular chapter explains the preparation and conversion of Interleaf files.

Chapter sections include:

- ❑ preparing the Interleaf files
- ❑ defining the conversion settings in S-Tagger
- ❑ converting from Interleaf ASCII to STF.

Chapter

7

PREPARING INTERLEAF FILES

This section contains information on features specific to Interleaf documents and the preparation of Interleaf documents for conversion to STF. You must complete a number of preparation steps before converting Interleaf ASCII (IASCII) files to STF using S-Tagger for Interleaf.

NOTE

- ❑ 'S-Tagger' throughout this section refers to S-Tagger for Interleaf.
- ❑ Information provided regarding ILDOC and ILSTY files also applies to DOC and STY files created using earlier versions of Interleaf.

The steps are divided into the following categories:

- ❑ preliminary guidelines – some guidelines to help you before beginning preparation
- ❑ project structure – preparation of your project folder which should be carried out before conversion in order to ensure smooth translation and separation of files
- ❑ required tasks – file preparation that *must* be carried out before converting files
- ❑ advised tasks – preparation that improves the quality of the STF files produced and minimises the DTP work required after translation
- ❑ special feature tasks - preparation that is only required when certain Interleaf features have been used in the original ILDOC files. Issues that arise when working with Japanese are also described in this section.

WARNING

You must have a good knowledge of Interleaf and DTP if you are preparing Interleaf files for translation.

Preliminary Guidelines

The following guidelines should be considered before beginning the preparation process:

- ❑ Remove any backup, crash, or corrupt files from the original document folder.
- ❑ Prepare the Interleaf files on the platform that the files originated on. It is not necessary to port UNIX Interleaf files to Interleaf for Windows to either prepare them or save them in IASCII format.

! WARNING

When moving IASCII files between UNIX and Windows platforms using FTP, make sure you use a binary file transfer or your IASCII files may become corrupted.

- ❑ If your project has been set up with duplicate file names in separate books which contain separate chapters, you must rename each of these files.
- ❑ If catalogs are used to export master properties into the files, you must ensure that these catalogs are positioned correctly within your document's book container.
- ❑ While S-Tagger recognises long file names, version 6 of Interleaf for Windows (or earlier) truncates the file names outside of the Interleaf desktop to the DOS file-naming convention of an 8-character name with a 3-character extension. The long file name is detailed in the attribute file. It is advisable to stick to the 8.3 file-naming convention.
- ❑ Convert only files that contain translatable text from IASCII to STF.

Automatically generated files, such as table of contents and index files, should be converted only if they contain text for translation, for example text in headers and footers.

Such text is referred to as ancillary text and it is made available for translation in the ancillary file. The text in the generated files can be translated just once in the ancillary file and then re-inserted during conversion back to the original file format.

If the table of contents and/or the index have not been generated, but are manually built, the files should be included for conversion and translation.

- ❑ If your book contains catalogs which export headers/footers, shared components, frames or named objects containing translatable text, prefixes, autonumber or page number streams, you will need to convert the catalog files as well as the *.doc or *.ildoc files.

→ NOTE

S-Tagger will only convert IASCII files which have the extension *.doc or *.ildoc and catalog files with the extension *.sty or *.ilsty.

- ❑ Before the files are translated, we recommend that you convert the new STF files back to IASCII and open them again in Interleaf to verify that the conversion process has worked correctly.

Interleaf Version Information

S-Tagger for Interleaf supports IASCII and catalog files saved from Interleaf 5.2 or later. This release supports Interleaf 7, included as part of the QuickSilver XML application suite. IASCII files generated by the use of customised tools or applications are not supported by S-Tagger for Interleaf. Manually built or edited IASCII files are also not supported.

Setting up the Project Structure

S-Tagger never changes the name of the files that it is working on. It generates files which have the same base name as the files being processed; only the extensions are different. Before you start working with S-Tagger, you must create a folder structure for your project. This facilitates file management and helps prevent files being overwritten.

During conversion, S-Tagger creates three new files for each IASCII file:

- ❑ an STF file
- ❑ an ORG file
- ❑ a copy of the STF file

➔ NOTE

If you have included catalog files in the conversion, S-Tagger produces STY and CAT files and saves them in the *Source* folder. A copy of the CAT file is saved in the *Target* folder. For more information, see “The CAT File” on page 7-5.

S-Tagger can also create an ancillary file if you choose, saving it to the *Source* and *Target* folders. The ancillary file is described in detail in “The Ancillary File” on page 2-3. For more background information on all these files and project structuring, see “What is an STF File?” on page 2-2, “Other S-Tagger Files” on page 2-5, and “Project Structure” on page 2-9.

To set up your project structure using Windows Explorer:

- 1 Create a translation project folder in an appropriate location.
- 2 Copy all the Interleaf and associated files (graphics or other referenced objects) to this folder. We recommend that you copy your original files so that if you make any mistakes, you can always recopy the originals and start again.

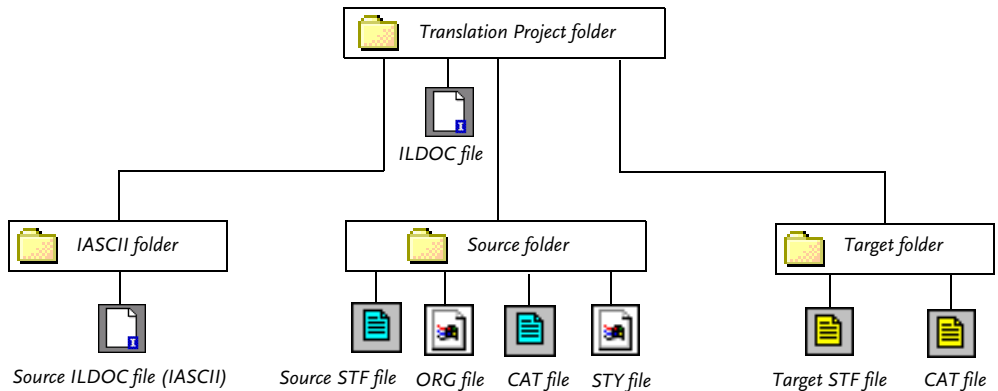
➔ NOTE

You must maintain the same folder hierarchy as existed in the original Interleaf project folder, otherwise, the Interleaf files may not be able to locate all the referenced files. If your files contain graphics that need to be translated, you should deal with these separately and place them in a folder of the same name in the *Target* folder, maintaining the same folder structure. For example, if you have an *Art* folder in your original project folder, you must also have an *Art* folder in the translation project folder.

- 3 Create an *IASCII* folder, a *Source* folder, and a *Target* folder within this project folder. The *IASCII* folder will contain the IASCII files saved from your original ILDOC files and the *Source* and *Target* folders will contain the converted STF files.

When you have completed backward conversion of the IASCII files, the new IASCII files will be saved to the *Target* folder.

By the end of the conversion process, your folder structure and content will be something like this:



➔ NOTE

- ☐ There may be an additional ancillary file in the *Source* and *Target* folders if you select the **Create ancillary file** option on the **Settings** tab. For more information, see “The Settings Tab” on page 7-16.

The CAT File

CAT files are created by S-Tagger from Interleaf catalog files. The original catalog files have the extension `*.sty` or `*.ilsty` but when converted by S-Tagger, this extension is changed to `*.cat`.

Catalog files in Interleaf function as master style sheets. The author of the original Interleaf files sets up components, shared components, page numbering, and headers and footers in the catalog and exports these components to other files in the Interleaf book.

Any ancillary text from the catalog file will be listed last in the ancillary file.

The CAT files are provided for information purposes only. There is no translatable text in them. You can use them as a reference to check whether the file you are translating is importing text from a given catalog. Note that you will not necessarily receive any CAT files.

Required Tasks

The following items can have an affect on IASCII to STF conversion and should therefore be checked before attempting the conversion process. If you are unsure of how to carry out the steps, consult your Interleaf/QuickSilver manual or seek advice from qualified Interleaf/QuickSilver technical support staff.

Here is a brief checklist for you to refer to:

Issue	What to Check
Opening/saving messages	Are there any error messages informing you of font or language dictionary errors in the ILDOC file?
Catalogs	Are the relevant catalogs in the correct position? Are the catalog files saved in IASCII format?
Control expressions	Are the relevant control expressions set?
Revision tracking	Is revision tracking turned off and cleared?
Cross-references	Are all the 'NO TAG' references been updated in the files?

Opening/Saving Messages

Open the first file in the book. The **Opening/Saving Messages** window may appear and display a message about one of the following:

- font substitution – if a font substitution message is displayed, close the file without saving it. Locate the missing fonts and install them on your system.
- missing language dictionaries – if a missing language dictionaries message is displayed, you can ignore it. However, you may wish to locate and install the missing language dictionaries on your system.

If you do not know how to install fonts or language dictionaries on your system, consult your Interleaf/QuickSilver manual or seek advice from qualified Interleaf/QuickSilver technical support staff.

Catalogs

Catalogs that are used to export components, graphics, headers and footers, or any other formats imported by the files must be in the correct positions.

If you want to translate text within the catalog files, you must save the catalog as a *.doc or *.ildoc file as well as a *.sty or *.ilsty file. You can then save the file in IASCII format.

Control Expressions

Relevant control expressions should be set before you save the files as IASCII. Any text hidden using a control expression will not appear for translation in the STF file.

Revision Tracking

Revision tracking must be turned off in the ILDOC files. Revisions in the document should also be removed. S-Tagger will not process IASCII files with revision tracking on.

Cross-references

All 'NO TAG' references in the file must be updated. A 'NO TAG' reference may indicate that the cross-referencing is not up-to-date.

Advised Tasks

Carrying out these preparation steps improves the quality of the STF files produced and minimises the DTP work required after translation.

Here is a brief checklist for you to refer to:

Issue	What to Check
Attribute references	Do the attribute references contain text that must be translated? If so, have you extracted the text from the references?
Autonumber stream length	Have you shortened all lengthy prefixes or suffixes in the stream lengths for translation?
Catalog files	Do you need to translate all the ancillary text in your catalog files? If so, have you created a dummy file for translation of this text?
Cross-reference variable length	Have you shortened all lengthy cross-reference suffixes for translation?
Hard returns	Have you checked the usage of hard returns in the ILDOC file?
Hyphenation	Have you adjusted the hyphenation to meet your requirements for the target ILDOC files?
Index tokens	Have you shortened all lengthy marker text to allow for translation?

Issue	What to Check
IASCII instructions in index entries	Have you determined whether there are IASCII instructions in the index entries that need to be translated?
Page number stream length	Have you shortened all lengthy page number stream prefixes?
Read-only components	Have you determined whether there is read-only content that needs to be translated in your files?
Table of contents and index	Have you determined whether any generated files contain text that needs to be included in the ancillary file for translation?

NOTE

All symbols and special characters which can be used in Interleaf will be converted to either their ANSI equivalent in the STF file or to an S-Tag. A full list of all Interleaf symbols and special characters can be found in your Interleaf/QuickSilver manual or Online Help file.

Attribute References

If you have defined a value for an attribute reference and the text of that value is to be translated, you must extract this information separately. S-Tagger does not present attribute reference value text for translation. If the file contains any attribute references, a warning is listed in the log file after you have converted the files from IASCII to STF.

Autonumber Stream Length

Prefixes and suffixes in autonumber streams may contain a maximum of 15 characters in Interleaf. You may wish to shorten the text of any autonumber stream to ensure that the translation will not exceed 15 characters.

Catalog Files

S-Tagger extracts ancillary text from the catalog files. The ancillary text in a catalog will only be extracted and placed in the ancillary file if at least one instance of the ancillary text item is found in at least one *.doc or *.ildoc file in the batch being converted. If you want to translate all ancillary text in your catalog, create a dummy file which contains at least one instance of all the ancillary text and include it in the batch to be translated.

Cross-reference Variable Length

Suffixes in cross-reference variables (variable content) may contain a maximum of 20 characters. You may wish to shorten the text of any cross-reference variable suffix to ensure that the translation will not exceed 20 characters.

Hard Returns

You have the option of selecting whether or not S-Tagger should insert a paragraph mark before a hard return tag (< :hr>) in the STF files. The **Insert line break before <:hr> tag** option is available in the **Settings** tab. For more information, see “The Settings Tab” on page 7-16.

- ❑ If you select **Insert**, you must check the document for instances of a hard return (manual line break), using the following guidelines:
 - ❑ If a hard return is *cosmetic* or has been inserted to make a paragraph look better in the source language, you should delete it. Otherwise, translators working with a translation memory system will see two segments instead of one sentence, making segment matching more difficult.
 - ❑ If a hard return has been inserted to create a new line without incurring new paragraph properties, there is no need to delete it.
- ❑ If you select **Don't Insert**, paragraph marks will not be inserted before the < :hr> tag. This is to facilitate translation of text when hard returns appear in the middle of a segment or in the middle of a word. You do not have to examine your documents for hard returns with this option.

Hyphenation

- ❑ automatic hyphenation – Interleaf does not save automatic hyphens when creating an IASCII file.

If you want automatic hyphenation to apply to the target ILDOC file:

- ❑ In Interleaf, leave automatic hyphenation turned on and,
- ❑ In S-Tagger, select the **Leave as is** option under **Hyphenation** on the **Settings** tab.

The hyphenation will be reactivated when you save the target IASCII file as ILDOC.

If you do not want automatic hyphenation to apply to the target ILDOC file:

- ❑ In Interleaf, turn off the automatic hyphenation or,
- ❑ In S-Tagger, select the **Turn off** option under **Hyphenation** on the **Settings** tab.

The hyphenation will be turned off when you save the target IASCII file as ILDOC.

- ❑ non-breaking hyphenation – in some cases, Interleaf converts manually inserted hyphens into non-breaking hyphens depending on the context in which they are used. These hyphens are represented by the same hyphen tag in STF as a regular hyphen tag, (< :sh>). There is no need to change these.

Index Tokens

S-Tagger will not allow you to convert a file which contains an index token with 255 characters or more. During translation it is likely that the number of characters will increase. For this reason, S-Tagger warns you if you try to convert an IASCII file containing a marker which has more than a specified number of characters in it. Specify the number of characters under **Source Marker Length** in the **Settings** tab in S-Tagger.

IASCII Instructions in Index Entries

Any IASCII instructions in index entries are presented as text entries for translation. If your documents contain index entries with IASCII instructions within them, this must be clearly noted for translation purposes.

Page Number Stream Length

Prefixes in page number streams may contain a maximum of 9 characters. You may wish to shorten the text of any page number stream prefix to ensure that the translation will not exceed 9 characters.

Read-only Components

A read-only attribute may be applied to components during the editing process. On the **Settings** tab in S-Tagger, you can choose whether or not to make components with read-only content available for translation. Check all read-only components and read-only rows in tables and decide which setting is applicable to your documents.

Table of Contents and Index

If your book contains generated lists such as table of contents and index files, you can include the text (such as the words 'Contents' and 'Index') in the ancillary file. To do this, you convert the table of contents and index along with the rest of the files in the book, as if they were normal chapter files containing translatable text. Do not attempt to translate the generated text in these files. After translation is complete and the files are converted back to Interleaf, update the table of contents and index files as normal.

Special Feature Tasks

The following preparation is necessary if certain Interleaf features have been used during the generation of the ILDOC files. If you are using a standard edition of Interleaf, refer to the following checklist. If you are using Japanese with Interleaf, refer also to “Working with Japanese” on page 7-13.

Issue	What to Check
Frame anchors	Have all anchored frames that were in the middle of a word been moved?
Graphics	Have you compressed any large Interleaf graphics?
Outline text	Have you replaced any text objects that require translation with text strings or microdocuments?
Fixed width microdocuments	Have you determined whether all fixed width microdocuments contain enough space for text expansion in translation?
Text strings and chart labels in hidden named objects	Have you determined whether any hidden named objects contain text strings or chart labels that need to translated?
Printing locks	Have you checked that all text objects with a printing lock applied to them do not need to be translated?

Frame Anchors

Anchored frames that contain microdocuments, text strings or chart labels should not be placed in the middle of words. An anchored frame, positioned in the middle of a word, breaks up the word. This makes translation of the word more difficult and time-consuming, and will make the segment less re-usable.

Graphics

In some versions of Interleaf, the IASCII file size may be much greater than the binary file size. This may be caused by a lack of graphic compression. Some versions of Interleaf do not compress graphics by default when saving documents as IASCII.

To fix this, create a LISP script, based on the example below, and place it in the profile drawer in the custom cabinet in the system cabinet on your desktop.

Sample Lisp script:

```
(lisp-set-implementation "Interleaf Lisp" "2.0")
(im-set-vars
 :ascii-compress t
 :binary-compress t)
```

Quit Interleaf and re-launch the application. Save a file that was huge before as IASCII again and you will see a significant difference.

If changing the compression default property does not work, you should check the master properties of components which contain graphics. If the master contains a graphic, the file size will also be excessive because all of the graphic information in the master component will be repeated in the declarations section of the IASCII file. Remove the graphic from the master, unless it is in a shared frame which is used more than once. Consider placing all shared frames in a catalog if possible.

Outline Text

When you convert the IASCII files to STF, the S-Tagger generates a list of any outline text objects in the files. Isolate these text objects and see if they need to be translated. Replace text objects that require translation with text strings or microdocuments.

Fixed Width Microdocuments

As text usually expands during translation, check all fixed width microdocuments to see if there is enough space for text expansion. If not, modify the microdocument as required.

Text Strings and Chart Labels in Hidden Named Objects

S-Tagger does not distinguish between hidden and shown text objects in shared named objects. If your documents contain text strings or chart labels in hidden shared named objects, you may need to flag these instances for the translators or find an alternative method of presenting the text.

Printing Locks

Text objects which have had a printing lock applied to them will not be presented for translation in the STF file.



NOTE

If your documents contain text strings and chart labels in hidden named objects and/or printing locked items, a warning is listed in the log file during the IASCII to STF conversion process.

Working with Japanese

If you are working with Japanese as either a source or target language, there are some specific issues that you need to consider:

- ❑ unsupported characters – some Japanese characters which are available to translators are not supported in Interleaf. These are characters which Microsoft has added to the Shift-JIS standard. Translators should be instructed not to insert these characters into the STF files.

You can find a list of the unsupported characters in the Character Map on a system running Japanese Windows. Characters in the ranges 0xea3-0xee4, 0xed40-0xeefc and 0xfa40-0xfc4b are not supported.

- ❑ read-only components – if there are any read-only components in the document which have been translated into Japanese, you will have to remove the read-only property before you can apply the correct font.
- ❑ footnotes – if there are footnotes in the document, you may need to change the numbering system if it is set up as alphabetic. You do this in the Interleaf file and not in the STF file.
- ❑ string length – when the STF files are being verified, some strings, such as the prefixes and suffixes in autonumber streams, are checked for length. Interleaf only allows a limited number of characters in these strings. The warnings are based on the single byte characters in the string, so if you are warned that a string is two characters too long, shortening it with one double byte character will give the correct result.
- ❑ Japanese (WinAlign) as a source language – Japanese (WinAlign) has been added as a source language to generate STF files that use <:so> tags instead of <ss> and <sa> tags in index markers. This is useful for alignment purposes because Japanese documents often contain sort strings for all index entries. If you are converting Japanese IASCII files to STF for the sole purpose of alignment, select **Japanese (WinAlign)** from the **Source Language** list on the **Conversion** tab of the **Tools Settings** page.



FOR MORE INFORMATION

For more information on alignment, see the *WinAlign User Guide*.

- ❑ reserved character tags – when you convert Japanese IASCII files to STF, the STF files may contain reserved character tags (<:r?>) where there are European extended characters in the Japanese text. The reserved characters may be left or replaced with their relevant European characters.

Saving ILDOC Files to IASCII Format

After you have completed all preparatory work in the ILDOC files, you can proceed with saving them as IASCII files.

To save ILDOC files as IASCII:

- 1 Launch Interleaf.
- 2 Open a file that you want to translate.
- 3 From the **File** menu, choose **Save As**. The **Save As** dialog box is displayed
- 4 Browse to the *IASCII* folder that you created earlier in your translation project folder. For more information on this, see “Setting up the Project Structure” on page 7-4.
- 5 Select **ASCII** from the **Save as type** drop-down list and click **Save**. The Interleaf file is now saved to your *IASCII* folder in IASCII format.
- 6 Repeat for each file that you want to translate.

If each file has been set up with its own container and catalog, copying all the IASCII files into the one folder (*IASCII*) will speed up the conversion process greatly. When the files have been translated, place them back into their original book structures to ensure that all referenced objects in the files (for example, graphics) link successfully.

Checking the IASCII Files

From time to time, errors in the IASCII files can occur and it is important that you check the files before proceeding to convert them to STF format.

To check the IASCII files:

- 1 In Interleaf, open each IASCII file that you saved to the *IASCII* folder.
- 2 If any error messages are displayed in the **Opening/Saving Messages** window, save them to a readme file.
- 3 Close the files again without saving them.
- 4 Open the readme and check the content of the message.
- 5 Include the readme with the delivery of ORG and source STF files.

In most cases, the message will not mention any errors and you can ignore it. However, somebody else may be converting the files back to IASCII after translation and it is important that they know the error was not introduced by S-Tagger and is not relevant.

DEFINING THE CONVERSION SETTINGS

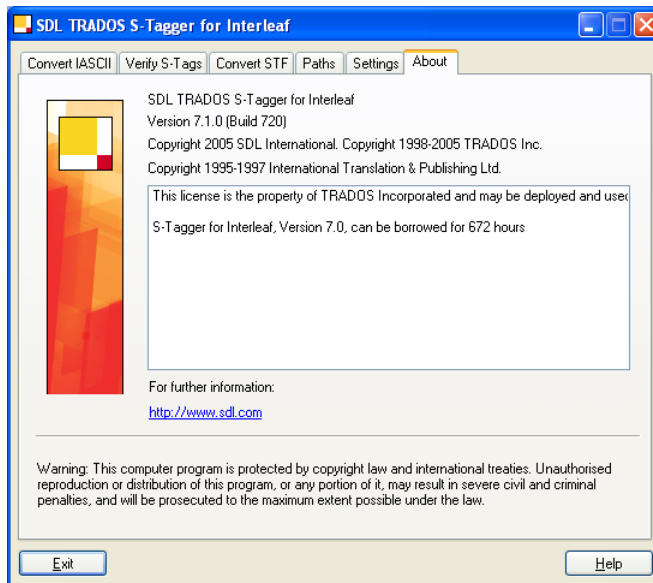
Before you convert the IASCII files to STF, you must select the appropriate conversion settings. To do this, use the S-Tagger **Settings** tab and the **Paths** tab.

Launching S-Tagger for Interleaf

To launch S-Tagger for Interleaf:

- 1 From the Start menu, browse to *SDL International>SDL Trados 2006>Filters>STagger for Interleaf*.

S-Tagger for Interleaf opens with the **About** tab displayed.



S-Tagger for Interleaf opens on the **About** tab by default. The **About** tab contains information about your licence and the S-Tagger version number.

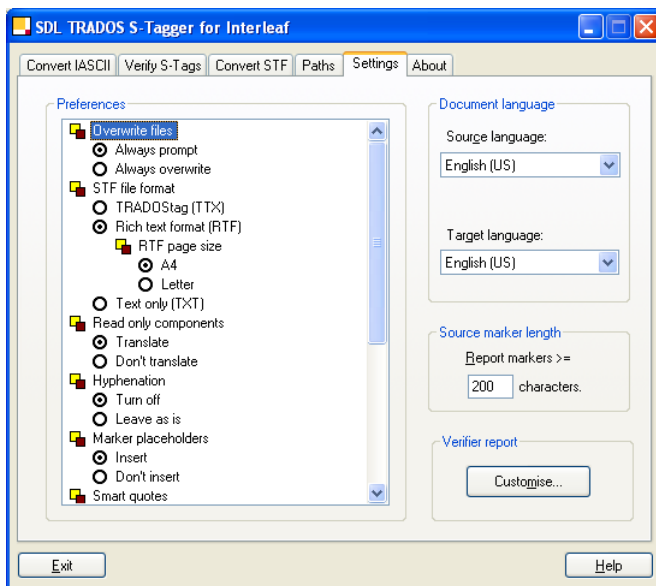
The Settings Tab

Before converting files to or from STF, or verifying the STF files, you must set the preferences for the file(s) you are processing. On the **Settings** tab, use the **Preferences** list to enter your conversion requirements. S-Tagger uses these settings during the conversion process to produce the STF files.

In this release of S-Tagger, you can convert directly to TRADOSTag (TTX) format. Select this option under **STF file format** if you intend using TagEditor as your editing environment. For more information on choosing your editing environment, see “Choosing an Editor” on page 2-13.

To enter your conversion settings:

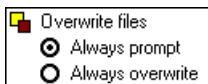
- 1 In S-Tagger, open the **Settings** tab.



- 2 Select the appropriate options from the **Preferences** list. The various options are discussed in the following sections. Make sure that you examine all the options on the list before proceeding.
- 3 Select the correct source and target languages for the project. For more information, see “The CMP file extensions are also influenced by this setting. If you select **Replace (traditional)**, the CMP file is `filename.cmp`. Document Language” on page 7-22
- 4 Enter the maximum source marker length. For more information, see “Source Marker Length” on page 7-23.

- 5 Select the messages that you want to suppress in the verifier report. For more information, see “Verifier Report” on page 7-23.

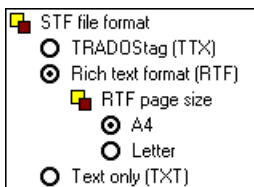
Overwrite Files



The default setting for this option is **Always prompt**.

- Select **Always prompt** if you wish to be prompted each time you are about to create new files which have the same names as existing ones.
- Select **Always overwrite** if you do not wish to receive this warning

STF File Format



The default setting for this option is **Rich text format (RTF)**.

- Select **TRADOS tag (TTX)** to create TTX files suitable for translation directly in TagEditor.
- Select **Rich text format (RTF)** to create RTF files suitable for translation in Microsoft Word. Note that TagEditor accepts RTF files, but converts them to TTX format while you are working on them. For this option, you must also select the page size for the RTF files; **A4** is the standard page size for Europe and Asia and **Letter** is the American standard.
- Select **Text Only (TXT)** as the file format to create ANSI text files that can be opened and edited in any Windows text editor. The tags will not have any special formatting. ANSI text format is mainly used for communication with document management or machine translation systems and is not recommended for interactive translation.

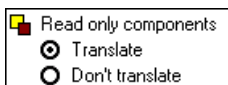
To assist you with this selection, refer to “S-Tagger Workflow: An Overview” on page 2-10.



NOTE

When translating into Japanese, you must choose **Rich text format (RTF)** or **TRADOS tag (TTX)** as the file format.

Read Only Components



The default setting for this option is **Translate**.

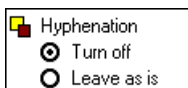
- Select **Translate** if you want S-Tagger to ignore the read-only property in all components, making these components available for translation. On conversion back to IASCII, the read-only property will still apply to any component in the translated files which had that property in the source file.
- Select **Don't Translate** to convert any file component with a read-only attribute to the tag `<:r "xxx" ?>`, where `xxx` is the text of the component. The read-only tag can be either internal or external, depending on whether the component is an inline component or not. This setting also applies to read-only rows in tables which are presented as `<:rrow "xxx" ?>`.



NOTE

If there are any read-only components that have been translated into Japanese, you will have to remove the read-only property before you can apply the correct font.

Hyphenation



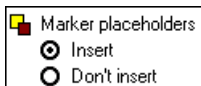
The default setting for this option is **Turn off**.

If automatic hyphenation is turned on in Interleaf, any hyphenation properties applied to the components will be active. If it is turned off, then no hyphenation properties are active. Automatically inserted hyphens are not imported by Interleaf to IASCII; this means that they will never appear in the STF file. This setting is relevant for conversion of STF files back to IASCII.

- Select **Turn off** if you wish to turn off automatic hyphenation during the conversion from STF back to IASCII.
- Select the **Leave as is** option if you want a target IASCII file to have the same hyphenation settings as the source IASCII file. If the original document did not have automatic hyphenation turned on, S-Tagger will not turn it on. This must be done manually, in the Interleaf file.

 **NOTE**

This setting will apply to the entire document, including all microdocuments. Manually inserted hyphens of any sort are not affected by this setting.

Marker Placeholders

The default setting for this option is **Insert**. Under **Marker placeholders**, you can specify whether or not to insert marker placeholders at the marker location. This feature applies to index markers and other markers that may contain translatable text.

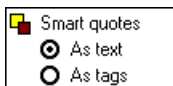
If a marker occurs within a sentence, S-Tagger moves the marker to the end of the paragraph. This allows you to finish translating a sentence before translating the text of the marker.

However, some markers may be placed strategically within the middle of a sentence, and should appear in a similar position within the translated sentence.

- Select **Insert** if you want to insert a marker placeholder at the point where the marker was originally found, while still placing the marker and its text at the end of the paragraph in the STF file. If marker placeholders have been inserted, the marker retains its position in the new translated file when the STF file is converted back to IASCII.
- Select **Don't Insert** if no marker placeholders should be inserted. When the file is converted back to IASCII, all markers will appear at the end of the paragraph in which they were originally found.

 **NOTE**

If your document has markers that are placed in the middle of words and you select **Insert**, the placeholders will appear in the middle of words and you will have to translate around them.

Smart Quotes

The default setting for this option is **As text**. Under **Smart Quotes**, you can choose whether to have the smart quotation marks saved as **Text** or **Tags**. Examples of smart quotation styles are “English”, « French », „German“.

If you are using TagEditor as your editing environment, you should note that TagEditor does not display smart quotes – it will always use straight quotes. To produce smart quotes, insert the

appropriate tags using the generic tag buttons (buttons **1** to **o**) on the Tags toolbar. Alternatively, press Ctrl+6 for English, Ctrl+7 for French, or Ctrl+8 for German smart quotes.

If you are translating into languages that use straight quotes only, the **As text** option will suffice. However, for all other languages in TagEditor, we recommend that you select the **As tags** option.

If you are using Word as your editing environment:

- The **As text** option converts the quotes to the RTF keywords `\ldblquote` (left double quote) and `\rdblquote` (right double quote). In the French version of Word this is seen on screen as « » , and in the German version as „ “.

When the STF file is converted back to IASCII these quotes are seen as the English smart quotes because they use the normal RTF keywords.

In a project where everyone is using the English version of Microsoft Word, you can use the **As Text** option and smart quotes can be typed the normal way for each language. This also applies to other languages that use the same smart quote characters as English.

- Select the **As tags** option to avoid losing the localised smart quotes. This converts all smart quotes to S-Tags. For example, `< : l d q >` for the English left double quote or `< : f r q >` for a French right quote. When you want to insert or change a smart quote, you should also use the tags instead of the literal symbols. This way the smart quotes will remain as intended no matter what version of Microsoft Word you use.



NOTE

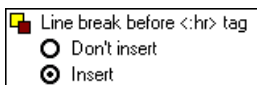
The **Smart quotes** setting does not apply to Asian languages as smart quote characters are among the characters above character position 127 which are not supported by S-Tagger.



TIP

Smart quotes can be turned on and off in Interleaf via the **Preferences** dialog box.

Line Break Before <:hr> Tag



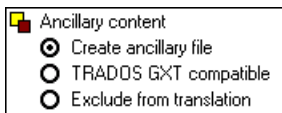
The default setting for this option is **Insert**. Using this option, you can select whether or not you would like S-Tagger to insert a new paragraph mark before each hard return (`< : hr >`) tag in the STF files.

- Select **Insert** to insert a paragraph mark before each `< : hr >` tag.
- Select **Don't insert** to prevent paragraph marks being inserted before each `< : hr >` tag. This option facilitates a more layout-based translation.

FOR MORE INFORMATION

For more information on S-Tagger's handling of hard returns, see "Hard Returns" on page 7-9.

Ancillary Content

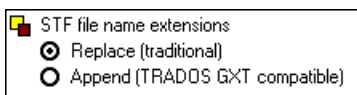


The default setting for this option is **Create Ancillary file**.

- Select **Create ancillary file** to create an ancillary STF file. S-Tagger extracts the ancillary text from the individual IASCII files and saves it to a single ancillary file. The ancillary content can then be translated just once and is automatically re-inserted back into each IASCII file during backward conversion.
- Select **Exclude from translation** to ignore the ancillary text. If you select this option, this text will not be made available for translation.

For more information on the ancillary file, see "The Ancillary File" on page 2-3.

STF File Name Extensions

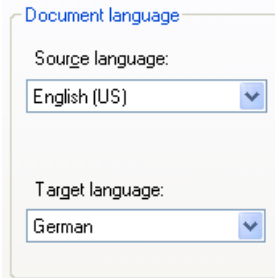


The default setting for this option is **Replace (traditional)**. STF file name extensions allows you to specify whether you wish to use long or short STF file name extensions. In certain situations, the long file extension may help you to avoid inadvertently overwriting files.

- Select **Replace (traditional)** if you want S-Tagger for FrameMaker to apply the following file extensions to the converted STF files: *.rtf, *.rtf.ttx, or *.txt (depending on your choice of STF file format), and *.org in your source folder.

The CMP file extensions are also influenced by this setting. If you select **Replace (traditional)**, the CMP file is `filename.cmp`. **Document Language**

To enter the languages in the **Document language** box, simply select the appropriate languages from the **Source language** and **Target language** drop-down lists.



The image shows a dialog box titled "Document language". It contains two dropdown menus. The first is labeled "Source language:" and has "English (US)" selected. The second is labeled "Target language:" and has "German" selected.

WARNING

During conversion to STF and verification in S-Tagger, ensure that you set the source language correctly, otherwise, the spell-checking and other language-related features of Interleaf may not work correctly.

The **Target language** should be set when converting the translated STF files back to IASCII; this is to cater for projects dealing with multiple target languages. This setting allows you to change all instances of the source language found in the document to the target language. It will not change the language assignment of paragraph or character formats which have been assigned a different language to the specified source language, or none at all. If the source document language has been set incorrectly, the **Target language** setting will not work and the translated file will retain the language setting of the source file.

The languages listed in the **Document language** group box are those languages for which Interleaf dictionaries are available. If the language you are translating into does not appear on the list, set the **Target language** to **None**.

Japanese (WinAlign) is included as a source language so that STF files can be generated that use `<:SO>` tags instead of `<SS>` tags in index markers. This is useful for alignment purposes because Japanese documents contain sort strings for all index entries and index levels. The internal `<:SO>` tag ensures that the segment is not broken up. If you are converting Japanese IASCII files to STF for the sole purpose of alignment, select Japanese (WinAlign) in the **Source language** list box.

FOR MORE INFORMATION

For more information on alignment, see the *WinAlign User Guide*.

Source Marker Length

To enter the source marker length, simply enter the appropriate amount in the **Report markers** text box.

The default value is 200 characters.

For Western languages we recommend you specify a maximum length of 200 characters, and for Asian languages, 150 characters.

In Interleaf, index and other markers contain a maximum of 255 characters. During translation, the size of the text in a marker may increase. Use the **Source marker length** box to specify the maximum number of characters acceptable. If S-Tagger detects more characters than specified in this box, you will receive a warning during the conversion process.

Verifier Report

The verifier report is the verification information displayed in the **Results** window on the **Verify S-Tags** tab. For more information on the verifier report, see “Verifier Report” on page 9-4.

To use the **Customise** command button:

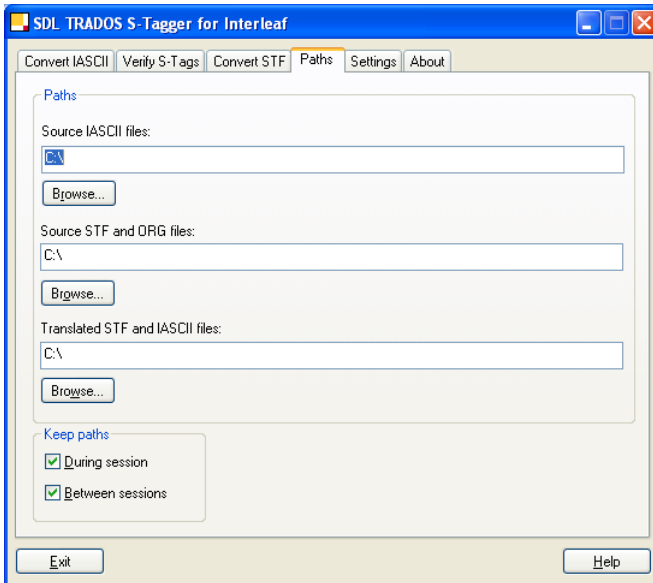
- 1 Click **Customise** to activate the **Customise Verifier Report** dialog box.
- 2 Select the alerts and warnings you wish to suppress during the tag verification process.

The Paths Tab

Use the **Paths** tab to set the paths for storing the S-Tagger project files before you start converting or verifying your files. You can also set or change the paths at each step in the process. The **Paths** tab consists of a group of text boxes for the various default paths. For more information on the recommended paths to use, see “Project Structure” on page 2-9.

To enter the paths for your translation project:

- 1 In S-Tagger, open the **Paths** tab.



- 2 Enter the paths to the folders you are using or click the **Browse** button underneath each option to enter a path.

→ NOTE

When you first open S-Tagger, the paths point to the C:\ drive by default.

- 3 Select the **During Session** box if you want S-Tagger to remember the file path for all project files. These paths are then used as the defaults until you close S-Tagger.
- 4 Select the **Between Sessions** option if you want the default paths to be written to the registry. This means that the next time you use S-Tagger, the same paths are displayed as defaults.

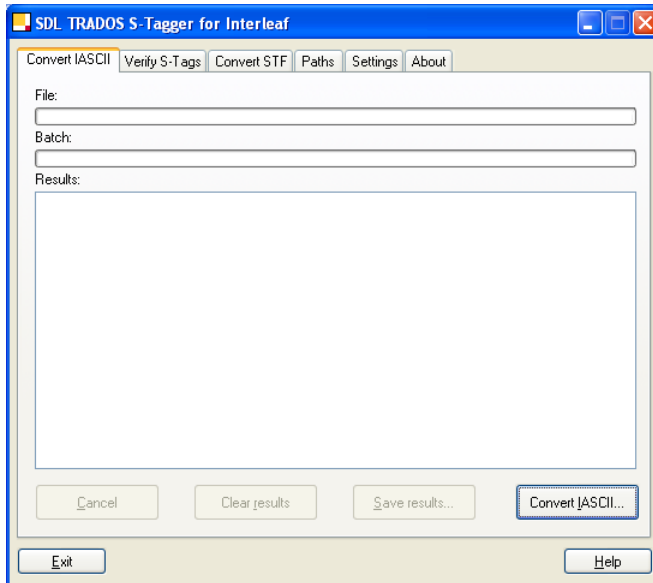
This option can only be selected if you have already selected **During Session**.

CONVERTING IASCIID TO STF

Once you are sure that your preferences are set correctly in the **Settings** and **Paths** tabs, you are ready to convert your IASCIID files to STF format.

To convert your IASCIID files to STF:

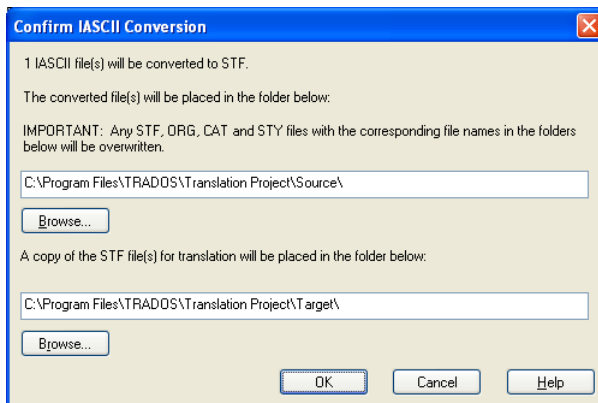
- 1 In S-Tagger, open the **Convert IASCII** tab.



Click the **Convert IASCII** command button to select the file(s) you wish to convert. The **Select IASCII File(s) to Convert** dialog box is displayed.

- 2 Select the file or files you wish to convert. Each file for conversion must have the extension *.doc or *.ildoc; *.sty or *.ilsty. Click **Open**.

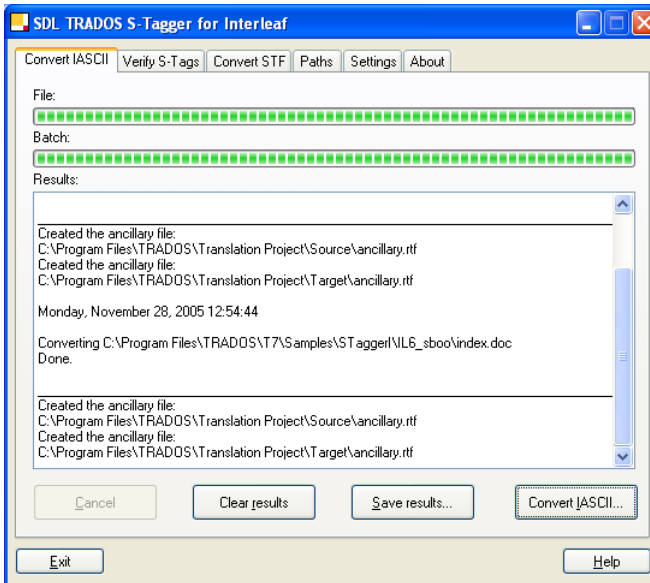
The **Confirm IASCII Conversion** dialog box is displayed, prompting you to confirm the conversion.



The paths listed here are the same as those you entered on the **Paths** tab.

- 3 Click **OK** to accept the listed paths or enter new file locations by clicking the **Browse** button.

As each file is being converted, a series of messages are displayed in the **Results** window. These include any warnings about the file content or error messages from the file conversion. The **File** and **Batch** progress indicators show you how much of the process is complete.



When file conversion is complete, a message box is displayed which informs you that the file(s) converted correctly.

- 4 Click **OK** on the message box. Your IASCII files have now been converted to STF format.
- 5 If you want to save the text in the **Results** window to a log file, you can do this by clicking **Save Results**. A dialog box is displayed, prompting you to specify the name of the file the results should be saved to. The log file is automatically given the extension *.txt and can be opened in any text editor.

Testing the New STF Files

After converting to STF, we recommend that you do the following:

- 1 In Windows Explorer, open your translation project folder.
- 2 Check that the source STF files and ORG files are in the *Source* folder.
- 3 Check that there is a copy of each STF file in the *Target* folder.

- 4 Make a copy of all the STF files in both folders and save them to a separate folder called *Test*.
- 5 Open each STF file in your chosen editing environment, make a small change to the text, convert the files back to IASCII and open each one in Interleaf. Be sure to make a similar change to the text in the ancillary file.

This test confirms that your STF files have not been corrupted during the conversion process and may be used safely for translation. Checking the STF files is especially recommended if you are translating multiple files into multiple languages.

- 6 When you are satisfied that the STF files are in order, you can delete the test copies and continue working with the files in the *Source* and *Target* folders.



WARNING

- ❑ *When making an artificial change to the STF files for the purpose of testing the files, be sure to change the text only, not the tags. If you change any of the text in a tag, the tag will no longer be valid and the file will not convert back to IASCII.*
- ❑ *If you are using a search and replace routine in Microsoft Word, make sure that you set the search and replace to change only the default paragraph text format and not the Tw4winInternal or Tw4winExternal text formats.*



STF FILE TRANSLATION

This chapter guides you through the translation process.

Chapter sections include:

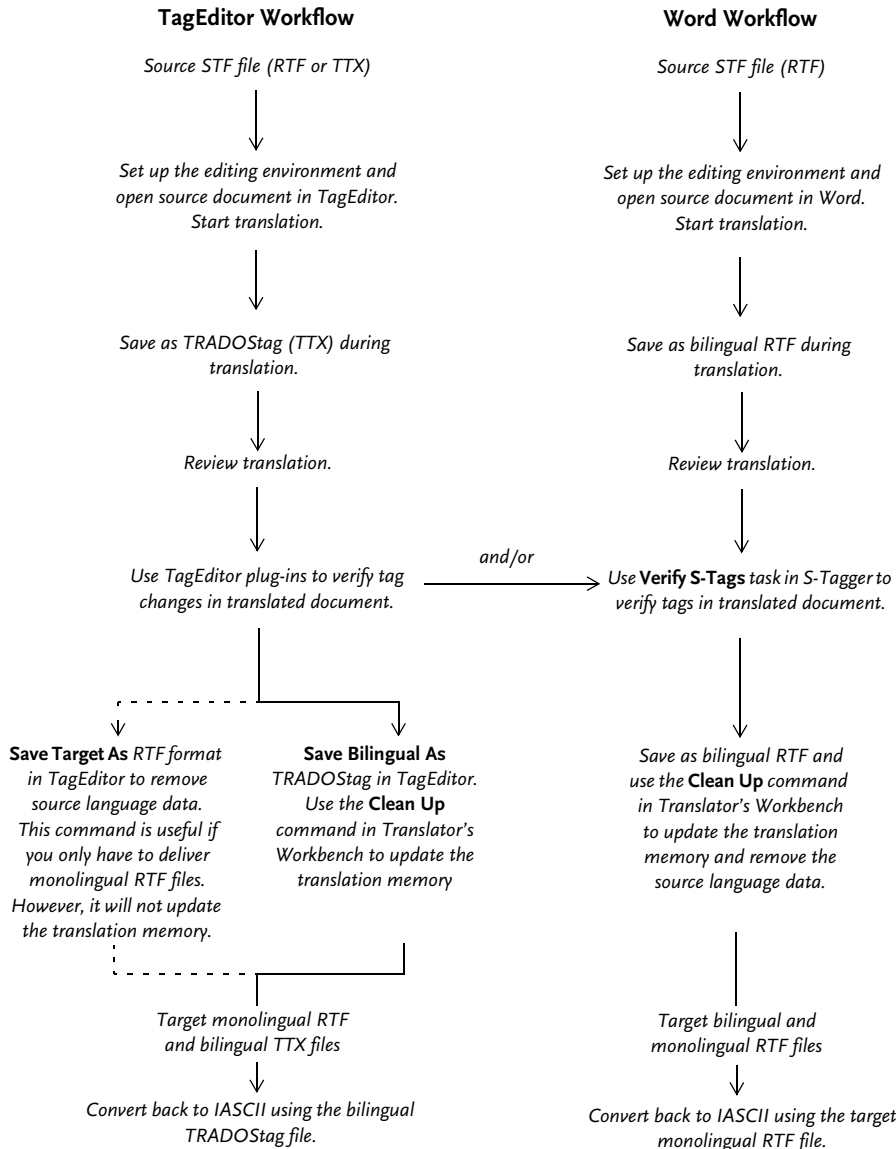
- ❑ translating STF files
- ❑ using S-Tags during translation
- ❑ translating to Japanese.

Chapter

8

TRANSLATING STF FILES

This section provides you with guidelines for translating STF files and using S-Tags. It also highlights some issues that may need particular attention during translation. The following diagram illustrates a high-level overview of the workflow for interactive translation in either of the editing environments. The workflow includes translation proper, tag verification and clean up of the translated STF files.



General Translation Issues

The following section describes some issues that may need your particular attention during translation, regardless of what source and target language you are using.

Localisation

The localisation issues that frequently arise during translation are as follows:

- ❑ formatting changes – some languages may require the addition of bolding or italics to some characters; in most translations, formatting tags need to be moved and/or deleted or added; in most translations, punctuation will be different to some degree
- ❑ carriage returns and/or sentences or words are sometimes deleted – some translations require the merging of paragraphs or sentences, or the deletion of a word which has been inserted as, for example, a variable
- ❑ paragraphs and/or sentences are sometimes added – some translations require a paragraph in the source language to be split into two paragraphs in the translation
- ❑ paragraphs are often moved to positions in the translation that are different to their positions in the source.

Ancillary File

If you have chosen to create a separate ancillary file on the **Settings** tab, you must remember to translate this text also. This translated text is re-inserted into the files during the conversion back to IASCII.

File Format

When you save the files after finishing translation, ensure that you save them in the customer's required delivery format:

- ❑ If you have used TagEditor for translation and you are asked to deliver the target STF files back in RTF file format, you can do either of the following:
 - ❑ Use the **Clean Up** command in Translator's Workbench and deliver the RTF file created during clean up. This option is recommended as it updates the translation memory.
 - ❑ Use the **Save Target As** command in TagEditor to save the bilingual file as a monolingual RTF file.
- ❑ If you have used Word for translation and you are asked to deliver the target STF files back in TTX file format, you can open the bilingual (uncleaned) RTF file in TagEditor and save it as TTX.

WORKING WITH S-TAGS

The following sections describe the S-Tags and how you can manipulate them during translation. Translator's Workbench ignores external tags and treats internal tags as placeables. This means that while they do not require translation, you must place them in the correct location in the target segment.

If you are using a translation memory from a previous project, remember that the tags in the translation memory may differ from the tags in the file for translation. You must check that the tags from the source segment in the file are included in the target segment.

At any stage during translation, you can check that you are working with the tags in the correct way by using the S-Tagger verification feature or TagEditor's S-Tag Verifier plug-in. Your verification options depend on the editing environment you are using.



FOR MORE INFORMATION

For more information, see:

- ❑ "S-Tagger Workflow: An Overview" on page 2-10
- ❑ "Choosing an Editor" on page 2-13

General S-Tag Guidelines

There are clearly defined rules for adding, deleting or moving tags. To successfully convert the STF files back to IASCII, check the following:

- ❑ all the tags have correct opening and closing signs ('<' and '>')
- ❑ the tag pairs have both the opening and closing tags
- ❑ the correct tag is always used to start and end the STF file:
 - <stf "Ix.xx"> and </stf>, where x.xx is the version number of the STF file format.
- ❑ the following tags have the correct values:
 - ❑ <sourcelanguage "xxx">
 - ❑ <sourcequotes "xxx">
 - ❑ <sourcereadonly "xxx">
 - ❑ <sourcepath "xxx">
 - ❑ <ancillarymode ?> where '?' can be a value between 0 and 2

- ❑ all the tags are valid (correct S-Tags)
- ❑ the correct sequence has been followed (some tags must follow a particular sequence; for example `<imk ?>` must be followed by at least one `<ie>` tag and a closing `</imk>`)
- ❑ all the tags are in the correct position, for example, external tags usually occur at the beginning of a paragraph.

Since TagEditor provides advanced tag protection and insertion capabilities, many of the following guidelines are more applicable to Word. Information that relates specifically to the Word editing environment is clearly identified.



FOR MORE INFORMATION

For more information on tag protection and insertion in TagEditor, see “Inserting Tags in TagEditor” on page 8-6 and the TagEditor tutorial in the *Translator's Workbench User Guide*.

Formatting

To apply bold or italic formatting, or to apply small caps, insert the tag `< :b>` for bold, `< :i>` for italic, `< :bi>` for bolded italics or `< :s>` for small caps, in front of the word you wish to format. Place the closing tag (`< /b>`, `< /i>`, `< /bi>` or `< /s>`) at the point where you wish to return to the default font.

If you are using Word, you should always use the *Normal* font to enter tag text. Any formatting, apart from that done with tags, will be lost.

If you are using TagEditor, you can insert such common formatting tags by using the Tags toolbar or by right-clicking in the target segment and choosing the **Insert Tag** command from the shortcut menu.

Extended Characters and Symbols

Insert extended or accented characters using the keyboard, *not* the **Insert Symbol** command in Word or the Character Map utility in Windows. ALT sequences and country-specific keyboards can be used without any problem.

If symbols appear in the hard copy, they may appear either as anchored frame tags, as single characters surrounded by font change tags, or as internal tags in the STF files. For example, `< :fc 3>S< /fc>`. This implies that the letter ‘S’, when given the character style ‘fc 3’, appears as the symbol you see in the hard copy.

Special characters, like hard spaces or thin spaces, cannot be inserted in the ordinary way in Word. If you want to use these features, you must insert the corresponding S-Tag where you want the character to appear. For example, `< :hs>` for hard space or non-breaking space.

If you are using TagEditor, you can insert some of the common symbols by using the Tags toolbar or by right-clicking in the target segment and choosing the **Insert Tag** command from the shortcut menu.

Quotation Marks

You can insert quotation marks either by typing in the ANSI code or by inserting the S-Tags for quotation marks. If smart quotes are used in the document, English smart quotes appear as the tags <:ldq> for the opening quote and <:rdq> for the closing quote; French quotes appear as <:flq> and <:frq>; German quotes appear as <:glq> and <:grq>.

If you are using a localised version of Microsoft Word, you must insert smart quotes as tags when translating into German and French.

NOTE

Even if you select **Smart Quotes As Text** in the S-Tagger **Settings** tab, you can still use the tags to insert the localised smart quotes.

If you are using TagEditor, you can use the generic tag buttons (buttons 1 to 9) on the Tags toolbar to insert a variety of quotation marks. You can also right-click in the target segment and choose the **Insert Tag** command from the shortcut menu.

Inserting Tags in TagEditor

There are two methods of inserting tags in TagEditor:

- the Tags toolbar
- the **Insert Tag** dialog box.

The Tags toolbar in TagEditor allows you to quickly and easily insert tags and special characters into target segments during translation. The toolbar consists of a series of buttons; each button corresponds to a different tag, tag pair, or special character. The content of the toolbar is different for each file format that TagEditor supports.

The **Insert Tag** dialog box is accessed by right-clicking in the document and choosing the **Insert Tag** command from the shortcut menu. This displays the **Insert Tag** dialog box which contains a wide range of tags that you can insert into the target segment. This list displays the internal tags that you can insert, but if you want to use the list to insert external tags, simply select the **Also show external tags** option.

Once you have inserted a tag using the **Insert Tag** command, that tag is displayed in a list on the shortcut menu. If you right-click in the target segment, you can select the tag directly from this shortcut menu list, without having to access the **Insert Tag** dialog box. The list can contain up to ten

recently used tags. For more information, see the TagEditor tutorial in the *Translator's Workbench User Guide*.

Manipulating Internal Tags

An internal tag represents simple formatting information which often changes with translation. During translation, you will probably need to manipulate some of the internal tags. For example, some languages may require the addition of bolding to some characters. In most translations, formatting tags need to be moved, deleted or added; punctuation will also vary to some degree.

The following table outlines which internal tags may be added or deleted. If a tag is not listed here, it should never be added or deleted. Note that none of these tags contain numbers.

Internal Tag	Represents
<:hr>	Hard return
<:fs>	Figure space
<:hls>	Hairline space
<:hs>	Hard (non-breaking) space
<:lt>	Less than sign
<:gt>	Greater than sign
<:sh>	Soft hyphen
<:t>	Tab
<:t.>	Tab, leader character dot
<:t->	Tab, leader character dash
<:t-d>	Tab, leader character short dash
<:t_>	Tab, leader character underscore
<:lq> or <:rq>	English left and right single quotes
<:ldq> or <:rdq>	English left and right double quotes
<:flq> or <:frq>	French left and right double quotes
<:glq> or <:grq>	German left and right double quotes

Frequently Used Formatting Tags

These tags are never found in the source STF file, but may be added to the target STF as required. They are used to cater for the additional bolding, italicising, capitalisation, and unique character usage that may be needed when translating to certain languages. For example, in German, bolding is used to emphasise words much more frequently than in English.

If such formatting does occur in the original source STF, it is represented by character style or font change tags, not by these tags.

Internal Tag	Represents
<code><:b></code> and <code><:/b></code>	Bold
<code><:i></code> and <code><:/i></code>	Italic
<code><:bi></code> and <code><:/bi></code>	Bold italic
<code><:s></code> and <code><:/s></code>	Small caps
<code><:c></code> and <code><:/c></code>	Character set
<code><:c1></code> and <code><:/c1></code>	
<code><:c2></code> and <code><:/c2></code>	
<code><:so></code>	Sort order. Note that where the source STF file was created using Japanese (WinAlign) as a source language, it may contain <code><:so></code> tags.



WARNING

You may not add these tags to autonumber or page number streams, cross-reference variable suffixes, to text strings, to index, sort string or See also entries.

Other Internal Tags

You may delete character style, font change, shared content and cross-reference tags. Remember that deleting font change or character style tags will affect the formatting of the text. You may add character style, font change, shared component and cross-reference tags where the tags already exist in the source text. You cannot create new ones.

Internal Tag	Represents
<code><:fc ?></code> and <code><:/fc ?></code>	Font change
<code><:cs "xxx" ?></code> and <code><:/cs></code>	Character style

Internal Tag	Represents
<code><:sc "xxx" ?></code> and <code><:/sc></code>	In-line component An in-line component may indicate a font change.
<code><:v "xxx" ?></code>	Variable (shared component)
<code><:xr "xxx" ?></code>	Cross-reference

Manipulating External Tags

An external tag represents the kind of formatting that generally does not change, for example structural formatting. During translation, it may be necessary to add or delete certain external tags. These tags should only be modified by experienced users of both Interleaf and S-Tagger.

The only external tags which may be added or deleted are:

- paragraph styles
- paragraph numbering formats (prefixes)
- Index markers.

If an external tag is to be deliberately added to the file, it must contain a '#' as its second character. For example, to add the external tag `<ps "Head 3" 2>` to a file, the tag must be written as `<#ps "Head 3" 2>`. The '#' characters are deleted during the conversion of the STF files back to original file format.

If you are adding tags, we recommend that you copy an existing tag and change the text, being very careful to keep any spaces before, after and within the opening and closing braces.

If an external tag from the allowed list is added without a '#' character placed in front of the tag type identifier, it is detected as an invalid tag and the file returns an error during verification.

If you add an external tag, other than the ones listed above, even using the '#' syntax, it will generate an error and the file will not convert to IASCII. No external tag, other than those described in this section may ever be added or deleted.



WARNING

Do not invent tag numbers. If you do, this will prevent the translated STF file from being converted successfully back to IASCII.

If you want to modify external tags in TagEditor, you must first remove all tag protection. To do this, use the **Protection** tab in the **Options** dialog box. When you have deleted or moved the external tag, we recommend that you reactivate the tag protection.

Paragraph Styles

Paragraph styles are represented as `<ps "xxx" ?>` in the STF files. If you add a paragraph style tag to an STF file, you must use a paragraph style that already exists within the document and insert a hash character in the additional tag so that the tag appears as `<#ps "xxx" ?>`. If you do not follow these conventions, the file will not convert back to IASCI.

Never add a paragraph style tag to an STF file unless you are certain of the effect it will have on the formatting. Remember that paragraph styles are usually used to generate the Table of Contents of an Interleaf document, so changing the style may have an unwanted effect on the Table of Contents.

Deleting a paragraph style tag and not replacing it with another tag means that all the text in the paragraph is moved to the end of the previous paragraph, without a new paragraph break. The style assigned to the previous paragraph is assigned to the merged paragraph text.

TagEditor does not allow you to insert paragraph style tags using the Tags toolbar or the **Insert Tag** command. You are only allowed to add those paragraph style tags that already exist in the document. This is why you must copy an existing paragraph style tag and paste it to the desired position in the document in order to apply that paragraph style.

If you have copied and pasted such a tag, you must follow these steps to ensure that S-Tagger will treat it as a valid custom external S-Tag:

- 1 In TagEditor, turn off the document and tag protection. For more information, see the *Translator's Workbench User Guide*.
- 2 Select the tag and right-click on it.
- 3 Choose **Toggle # in S-Tag** from the shortcut menu. This automatically adds a hash character to the tag.



NOTE

This command is only available when a `<ps "xxx" ?>` or `<pn "xxx" ?>` tag is selected.

- 4 When you have finished editing the tags, we recommend that you turn the document and tag protection back on.

Paragraph Numbering Formats (Prefixes)

Paragraph numbering formats (also known as prefixes) are used if certain text or symbols should always precede a particular paragraph style in Interleaf.

Prefix text is presented as `<pn "xxx" ?>` in the STF files. A `<pn "xxx" ?>` tag may only be added to the STF file if it is accompanied by its relevant paragraph style tag. Each prefix tag must follow the paragraph style tag to which it belongs.

If you add or delete a `<ps "xxx" ?>` tag, make sure that you also add or delete any corresponding `<pn "xxx" ?>` tags. Deleting a `<pn "xxx" ?>` tag on its own will not delete the prefix from the paragraph in the Interleaf document. When adding this tag, you must also insert the hash (#) character so that the inserted tag appears as `<#pn "xxx" ?>`.

If you are working in TagEditor, follow the same procedure as described in “Paragraph Styles” on page 8-10 to insert prefix tags and ensure that S-Tagger will treat them as valid custom external S-Tags.

**NOTE**

Neither the `<ps>` nor the `<pn>` tag are available on the **Insert Tag** dialog box or the Tag toolbar. This is because you must use a tag that is already used elsewhere in the STF file.

Index Markers

You may also add and delete index markers. If you are adding an index marker, you must insert the '#' character; do not include a number. An `<imk>` tag must be followed by at least one `<ie>` tag and an `</imk>` tag.

If you add an index marker sequence to the file, the index marker is positioned at the end of the text paragraph immediately preceding the marker in the STF file. You cannot add an internal `<:imk ?>` marker to position the marker within the text.

**NOTE**

Index markers are known as index tokens in Interleaf.

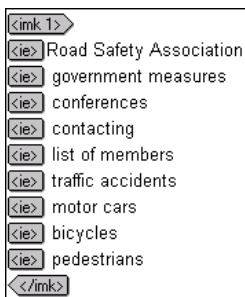
External Tag	Represents
<code><imk ?></code>	Index marker tag Deleting an <code><imk ?></code> tag deletes an index marker. If an <code><imk ?></code> tag is deleted, all following text and tags, up to and including the closing <code></imk></code> tag, must also be deleted.
<code><ie></code>	Index entry Deleting an <code><ie></code> tag deletes an index entry. Never delete an <code><ie></code> tag unless you are deleting an entire index marker. If you need to radically change the structure of an index entry and are confident in what you are doing, replace the existing index marker in its entirety with a new index marker, using the # syntax.
<code><il></code>	Index entry subentry Deleting an <code><il></code> tag deletes a subentry to an index entry. Only delete <code><il></code> tags if you are sure you wish to delete the subentry.

External Tag	Represents
<ss>	Sort string Deleting a sort string tag and its text replaces the sort string text with nothing. If you wish to delete a sort string entry, delete only the text and leave the tag in place.
<sa>	See also entry
</imk>	End index marker tag

Splitting Index Markers

If, during the verification process, you discover that an index marker contains more than 255 characters after you have translated it, you must split the index marker to create two index markers with less than 255 characters.

For example, the following index marker is found in an STF file in TagEditor:



To split this index marker into two markers:

- 1 End the <imk 1> after the index level *list of members*. Do this by inserting the </imk> tag on the next line.
- 2 Use the '#' syntax to create a new index marker. Start the index marker by inserting the <imk> tag. Do not place any number in this tag. Place a '#' at the start of the index entry tag <#ie> and at the start of each of the index level tags <#i1>. Remember to end the index marker with a <#/imk> tag.

In TagEditor, you can use either the <o> button from the Tag toolbar, which includes a preconfigured <#imk><#ie><#/imk> tag combination or right-click in the document and choose the **Insert Tag** command from the shortcut menu. This opens the **Insert Tag** dialog box.

Select the **Also show external tags** option so that you are able to insert `<imk>`, `<ie>`, and `<i1>` tags. After you have inserted a tag, this tag is listed in the **Insert Tags** command's submenu. This submenu displays a list of recently used tags.



NOTE

The inserted external tags are automatically prefixed with the hash character (#).

The split index marker will appear as follows:



TRANSLATING TO JAPANESE

If you are translating to or from Japanese, you will find the information in this section helpful.

The following issues may arise when translating into Japanese:

- ❑ character set tags
- ❑ unsupported characters
- ❑ index marker sort levels
- ❑ markers that must contain both Japanese and Western text.

Character Set Tags

To retain Western extended characters in the translated IASCII file, insert character set tags (<:c> and <:/c>) around the character or word.

Where you add the character set tag, additional character formats are added to the translated IASCII file when you convert it from STF to IASCII. Three character formats are added to the IASCII file; *STFWesternFont*, *STFCourier*, and *STFHelvetica*.

The following table summarises this character set tag information:

Character Set Tag	Font	Character Format in Interleaf
<:c> and <:/c>	Times	<i>STFWesternFont</i>
<:c1> and <:/c1>	Courier	<i>STFCourier</i>
<:c2> and <:/c2>	Helvetica	<i>STFHelvetica</i>

For example, if your documentation contains the product name ‘Qualité’ and you wish to retain the ‘é’ character in the product name in the final translation, use the character set tag as follows:

```
<:c>Qualité<:/c>.
```

You will need to take special care with individual extended characters, such as the ™ or © symbols. If you wish to retain these characters in your translation, you will need to insert the <:c> to <:/c> tags around them. If you are using Translator’s Workbench and one of these characters is on a line by itself, Workbench will not offer the segment to you automatically; you must check for this manually.



FOR MORE INFORMATION

For more information, see “Using Font Mapper for Interleaf” on page 10-11.



NOTE

You cannot place the < : c> tags in markers. Markers only support one character set, that of the language you are translating into.

Reserved Character Tag

When you convert Japanese IASCII files to STF, the STF files may contain reserved character tags (< : r?>) where there are European extended characters in the Japanese text. The reserved characters may be left as they are or replaced with their relevant European characters.

Unsupported Characters

Some Japanese characters which are available to translators are not supported in Interleaf. These are characters which Microsoft have added to the Shift-JIS standard. You must not insert these characters into the STF files.

You can find a list of the unsupported characters in the Character Map on a system running Japanese Windows. Characters in the ranges 0xea3-0xeea4, 0xed40-0xeefc and 0xfa40-0xfc4b are not supported.

If you have inserted any of the unsupported characters in the STF file, you will receive an alert message when you verify the files. The message will be similar to the one below:

```
Alert 1201. The following Japanese character does not exist in
Interleaf and may therefore not be used: "character".
```

If you receive this message, you must replace the unsupported character with a supported one.

String Length

When the STF files are being verified, some strings, such as the prefixes and suffixes in autonumber streams, are checked for length. Interleaf only allows a limited number of characters in these strings. The warnings are based on the single byte characters in the string, so if you are warned that a string is two characters too long, shortening it with one double byte character will give the correct result.

Index Marker Sort Levels

You may need to add additional sorting information to markers when working with Japanese. As Japanese is sorted phonetically, translation into the Japanese language requires that every index entry and index level has a corresponding sort string and sort level.

When working with Japanese, you may have to:

- ❑ add a sort string or sort level to markers
- ❑ adjust markers containing both Japanese and Western text
- ❑ change the source sort order.

Adding a Sort String or a Sort Level to Markers

Where there is no sort string or sort level in the source text, you need to add them when translating into Japanese. Adding the internal tag `<:SO>` allows you to assign sort levels to index markers.

In this example, the source index marker has no sort string or sort levels:

```
<imk 1>
<ie>File
<il>Open
</imk>
```

The index entry 'File' has 'Open' as an index level. To translate this into Japanese, you must insert a sort string. To insert this sort string, you add the internal tag `<:SO>` as shown in the following example:

```
<imk 1>
<ie>ファイル<:SO>ふぁいる
<il>開<:SO>ひらく
</imk>
```

By adding the internal tag `<:SO>`, the sort information is stored with the translation in the translation memory. When you convert the STF file to IASCII, S-Tagger for Interleaf preprocesses the marker and inserts the sort string.

NOTE

S-Tagger only allows you to insert one `<:SO>` tag per index entry or index level.

Adjusting Markers Containing both Western and Japanese Text

Where markers contain both Western and Japanese text, you can add a sort string for the Japanese text.

For example:

```
<imk 3>
<ie>Windows 2000
<il>installing
</imk>
```

You do not wish to translate Windows 2000, but you do require a sort string for “installing”. Add the internal tag `<:so>` as follows:

```
<imk 3>
<ie> Windows 2000
<il>インストール<:so>いんすとーる
</imk>
```

S-Tagger preprocesses the text when converting from STF to IASCII. The text “Windows 2000” becomes the sort string and the Japanese translation for “installing” becomes a sort level underneath Windows 2000.

NOTE

The S-Tagger verification feature checks the length of the marker and generates a warning if it is too long. When calculating the length of the marker, this feature only considers text which should be included in the marker during conversion from STF to IASCII.

Changing the Source Sort Order

If a marker in Interleaf contains sort strings, these will be presented using the `<ss>` tag in STF. When translating into Japanese, you will replace the source sort string text with its Japanese equivalent. The following example illustrates how S-Tagger processes this type of index marker:

```
<imk 2>
<ie>Graphics
<ss>Graphics
<il>inserting
<ss>inserting
<il>deleting
<ss>deleting
</imk>
```

When translating into Japanese language, you will need to modify the sort strings and sort levels to suit the translation. Do this by inserting the `<:so>` tag. Using this process, the preceding example becomes:

```
<imk2>
<ie> 画像<:so> がぞう
<ss>Graphics
<il> 挿入<:so> そうこゅう
<ss>inserting
<il> 削除<:so> さくじょ
<ss>deleting
</imk>
```

The `<:so>` tags will override the existing sort strings and sort levels. Before S-Tagger converts the STF file to IASCII, it will preprocess the marker and replace the original sort string and sort order tags with the `<:so>` tags you have inserted. This preprocessing occurs internally in S-Tagger and is not visible to you. Interleaf reads the index marker as if it had appeared as follows:

```
<imk2>
<ie> 画像
<ss> がぞう
<il> 挿入
<ss> そうこゅう
<il> 削除
<ss> さくじょ
</imk>
```

NOTE

During translation, where a source sort order is modified by using the `<:so>` tags, you can delete or ignore the source `<ss>` tags in STF. If you delete the tags, S-Tagger will recreate the translated sort order from the `<:so>` tags during the STF to IASCII conversion. If you ignore the `<ss>` tags in STF, S-Tagger will overwrite them in the translated IASCII file with the correct translated sort order.



S-TAG VERIFICATION AND CLEAN UP

This chapter explains the verification process and the updating of the translation memory during clean up.

Chapter sections include:

- ❑ verifying translated STF files
- ❑ updating the translation memory.

Chapter

9

VERIFYING TRANSLATED STF FILES

STF files can be verified using S-Tagger and/or TagEditor. This section describes the S-Tagger verification feature in detail; for more information on verification in TagEditor, see the *Translator's Workbench User Guide*.

Verification Guidelines

This section outlines some general guidelines for verification.

- ❑ Do not forget to include any ancillary files in the verification process.
- ❑ Run the file(s) through the S-Tagger verification feature again after fixing any errors, alerts or warnings that were generated the first time round.
- ❑ Continue verifying the file(s) until you are satisfied that any remaining alerts or warnings refer to changes which are intentional.
- ❑ Each time you run the file(s) through verification, save the CMP file with a new extension (for example, *.cm1 or *.cm2) to ensure the previous files are not overwritten.
- ❑ Save the last CMP files which show alerts or warnings about tag changes that you have examined and approved. Deliver these files with the finished translation, to facilitate smoother processing by technical and DTP personnel.

About Verification

Tag verification compares the tag content of the target STF files with the tag content of the source STF files and identifies any changes that were made. Changes in the target STF are acceptable provided that the syntax of the tags remains intact; if the tags are manipulated correctly, conversion back to IASCII format will always be successful.

- ❑ If you are using TagEditor as your editing environment, you can verify during and after translation using the S-Tag Verifier plug-in.
- ❑ If you are using Word as your editing environment, you must use S-Tagger to verify your STF files.

The S-Tagger verification feature is particularly powerful when used as a batch tool. It can also be used as an additional QA step after verification in TagEditor.

During translation, it is often necessary to add tags to the translated text that were not present in the source document. For example, bold formatting is used more frequently in German texts than in English, so a German translation may contain more instances of bold text than its English source.



FOR MORE INFORMATION

For more information, see “Working with S-Tags” on page 8-4.

Similarly, you may have to manipulate tags to suit your translation. Verification detects every added or deleted tag.

Some tag changes do not prevent the STF file from being converted back to IASCII; in such instances, S-Tagger generates alerts and warnings which inform you of such changes. It is recommended that you examine the alerts and warnings in order to determine which changes were intentional.

Verification in TagEditor

You can verify tags in TagEditor at segment or document level.

- ❑ segment level verification – during interactive translation, TagEditor automatically verifies the number, names, and order of internal tags in each target segment that you send to the translation memory. If there are changes, a message is returned.
- ❑ document level verification – TagEditor allows you to verify the tag content of whole documents, whether partially or fully translated, using the S-Tag Verifier plug-in. Document level verification looks at the internal and external tag content of the tag material and identifies any changes that have been made.

Checking your Conversion Settings

Before verifying the S-Tags, you must check that the current conversion settings in S-Tagger are correct.

To check your settings:

- 1 In S-Tagger, open the **Settings** tab.
- 2 Ensure that the **STF file format** option has been set to the correct format for the files you are working with, that is, **TRADOSTag (TTX)** for working with TRADOSTag files, **Rich text format (RTF)** for RTF files, **Text only (TXT)** for ANSI text files.
- 3 Check the STF file name extension setting is correct.
- 4 Change the **Target language** setting in the **Document language** group box to the language your file(s) has been translated into.

- 5 If you are working with Japanese translations, it is essential that the **Target Language** in the **Document Language** group box is set. This is to ensure that the correct font is used to represent any segments in the CMP file.
- 6 Open the **Paths** tab and ensure that the paths are still correct. If they are not correct, click **Browse** to point S-Tagger to the correct folders.

You do not need to change any other settings, except the settings in the verifier report described below.

Verifier Report

You can access the **Customise Verifier Report** dialog box through the verifier report **Customise** command button on the **Settings** tab. It contains a list of tags for which you can suppress the alert or warning that S-Tagger would otherwise generate during the verification process.

These alerts and warnings are displayed in the **Results** window on the **Verify S-Tags** tab and in the CMP file in more detail.

The CMP file is a comparison file that logs certain faults in the file being verified. This file is saved to the *Target* folder (the same folder as the files for verification), with the same base name as the file being verified and with the extension `*.cmp`. The CMP file lists the number and type of errors, alerts or warnings found in the file being verified.

Before running the verification process, you can determine which of the customisable alerts and warnings you wish to suppress. The S-Tagger verification feature detects these changes, but does not include detailed information about them in the CMP file. A list of all suppressed alerts and warnings appears at the end of the CMP file. For more information on using the CMP file, see “Using the CMP File” on page 9-10. The effect of each of these report messages is described in the following table:

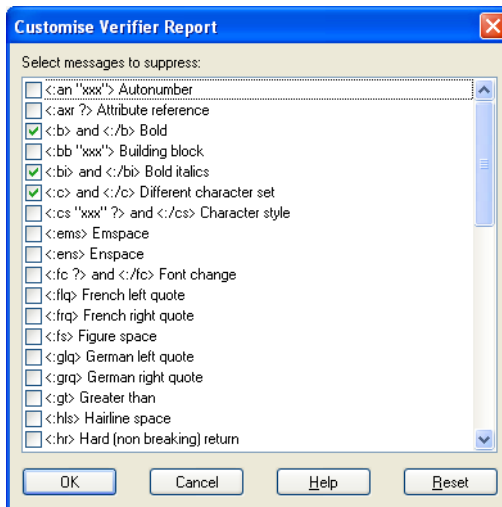
Message Type	Effect on Conversion
Error	Prevents an STF file from being converted to IASCII
Alert	Does not prevent the STF file from being converted to IASCII but may have a significant impact on the layout of the translated IASCII file
Warning	Highlights tags that have been manipulated (added, deleted, or moved) during translation

Setting Verification Message Preferences

In the **Customise Verifier Report** dialog box, tags are divided into external and internal tag sections. Within each section, tags are listed in alphabetical order.

To open the **Customise Verifier Report** dialog box:

- 1 In S-Tagger, open the **Settings** tab.
- 2 Click **Customise**. The **Customise Verifier Report** dialog box is displayed.



- 3 Scroll down through the list and select the warnings and alerts you wish to suppress by checking the appropriate box. The messages that correspond to the tags that you select are suppressed.

For example, if you have translated an STF file from English into French and have substituted French smart quotes for English smart quotes, you may wish to suppress all warnings and alerts in relation to deleting English smart quotes and inserting French smart quotes. In this case, you would select English smart quote tags (<:ldq>, <:rdq>) and French smart quote tags (<:flq>, <:frq>).

By default, the following tags are selected on the **Customise Verifier Report** dialog:

Tag	Represents
<:b> and <:/b>	Bold
<:bi> and <:/bi>	Bold italics
<:c> and <:/c>	Different character sets
<:i> and <:/i>	Italics
<:s> and <:/s>	Small caps

A list of suppressed messages will appear at the end of the CMP file that is produced during verification. The number of suppressed alerts and warnings is also included in the **Results** window.

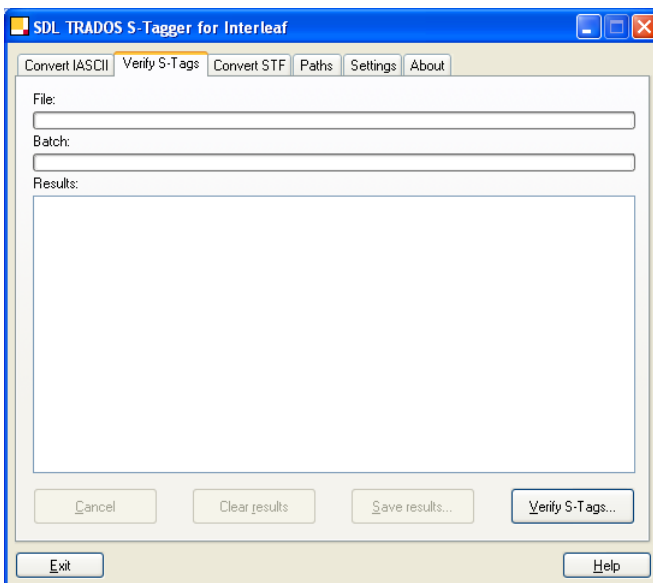
Now that you have set your verification preferences, you may proceed to verifying the S-Tags.

Verifying the S-Tags

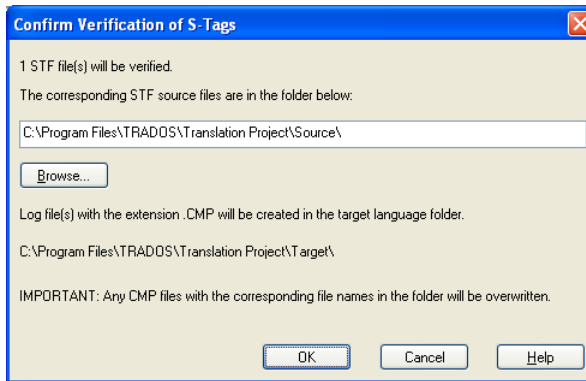
This section provides an overview of the verification process.

To verify the tags in your target STF file:

- 1 In S-Tagger, open the **Verify S-Tags** tab.

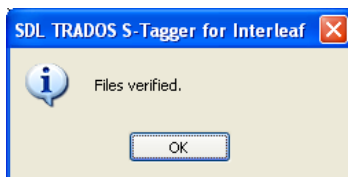


- 2 Click the **Verify S-Tags** command button and select the target STF files you wish to verify and click **Open**. A message box similar to the one below appears, prompting you to confirm that the path to the folder in which the source (untranslated) STF files are kept is correct:

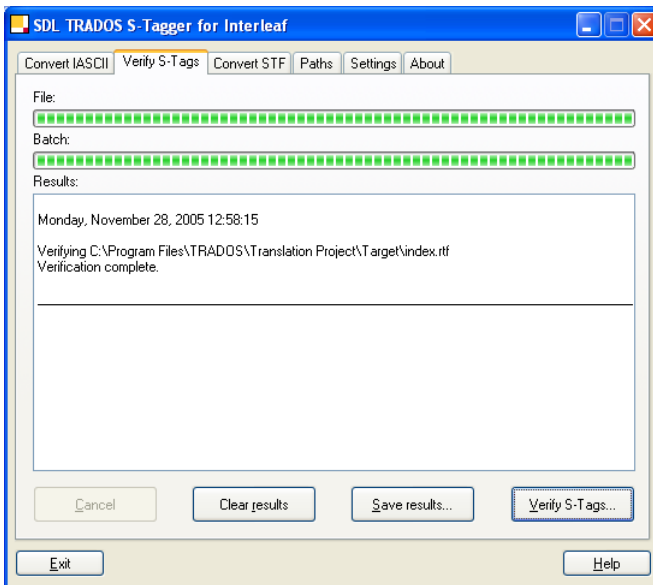


- 3 If the path that S-Tagger has listed here is not correct, click the **Browse** button and locate the correct directory.

During the verification process, the outcome of the comparison is written to the **Results** window. When the verification is complete, a message box similar to the one below appears. In this example, no errors were found.



The **Results** window also shows the results of the comparison.



- 4 If you want to save the text in the **Results** window to a log file, you can do this by clicking **Save Results**. A dialog box is displayed, prompting you to specify the name of the file the results should be saved to. The log file is automatically given the extension `*.txt` and can be opened in any text editor.

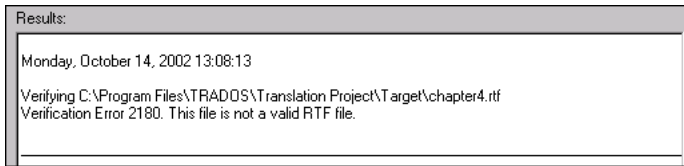
File Errors

File errors are only reported in the **Results** window, and a CMP file will not be generated. All other errors are reported in the CMP file.

These errors occur when a file cannot be opened or closed, or if a file has not been saved in the correct format. When a file error is encountered, a CMP file is not created for the file; the error message is written in the **Results** window.

In the following example, an RTF file had been saved as a `*.doc` file in Word by mistake. The user then changed the extension back to `*.rtf` in Windows Explorer to try and correct the mistake.

When the user attempted to verify the file, the following message was displayed in the **Results** window.



To correct this error:

- 1 Open the file in Microsoft Word and save it as RTF, making sure it has the extension `*.rtf`.
- 2 Verify the file again.

S-Tag Verification Messages

As well as the messages that appear in the **Results** window, a more detailed account of the tag changes is written to the CMP file. If there are no errors, alerts or warnings are generated (or if there are only file errors), no CMP file is created. Messages in the CMP file are usually clear and self-explanatory. If you do not understand a message, refer to the S-Tagger for Interleaf Help.

There are four categories of messages:

- comparison errors
- tag errors
- tag alerts
- tag warnings.

In addition to comparison and tag errors, the verification feature informs you about other differences between the tags in the source file and those in the target file. These tag differences are categorised as tag alerts or tag warnings.

If you use TagEditor for translation, you will avoid some of the verification messages described in this section. This is because TagEditor's unique tag protection safeguards against issues that can easily occur if you have chosen Word as your editing environment.

NOTE

If you find any errors, alerts or warnings in the CMP file which you then fix in the STF file, you should save the CMP file with another extension, for example, `*.cm1` as the original CMP file may be overwritten at a later stage.

Using the CMP File

A CMP file is created for each STF file that generates a verification message. The CMP file has the same file name as the respective STF file but with a *.cmp file extension. When correcting any verification errors, alert, or warnings, open the CMP file and its corresponding STF file in the editing environment that you are using for translation.

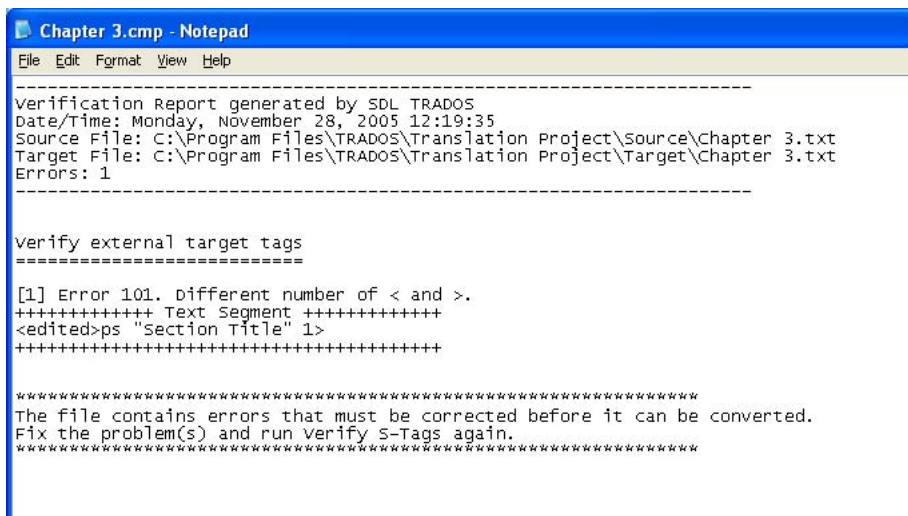
To use the CMP file:

- 1 Open the CMP file from the *Target* folder in the same editing environment as you have chosen for translating or editing the files.

You can open the CMP file in Notepad or Word.

- 2 Open the STF file in a second window in your editing environment.

The CMP file always starts with a header message which date/time stamps the file. The date/time stamp is followed by a reference to the source and target files being verified and lists the total number of errors, alerts, warnings and suppressed alerts and warnings found in the file. The header message should be similar to the one below:



```

Chapter 3.cmp - Notepad
File Edit Format View Help
-----
Verification Report generated by SDL TRADOS
Date/Time: Monday, November 28, 2005 12:19:35
Source File: C:\Program Files\TRADOS\Translation Project\Source\Chapter 3.txt
Target File: C:\Program Files\TRADOS\Translation Project\Target\Chapter 3.txt
Errors: 1
-----

Verify external target tags
=====

[1] Error 101. Different number of < and >.
+++++++ Text Segment ++++++++
<edited>ps "Section Title" 1>
+++++++

*****
The file contains errors that must be corrected before it can be converted.
Fix the problem(s) and run verify S-Tags again.
*****

```

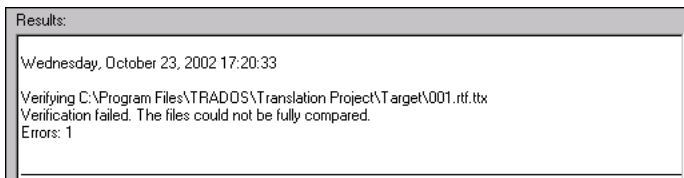
The header message is followed by a report of the procedures that S-Tagger has gone through. If an error, alert or warning is found during any of the procedures, it is listed in the section for that procedure. This can lead to the same error, alert or warning being reported several times.

Comparison Errors

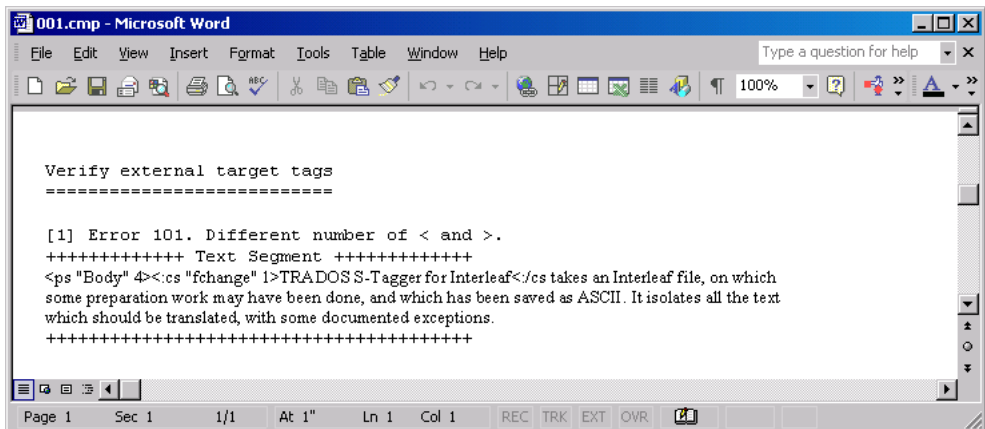
These are serious errors which prevent the files from being fully compared. In most cases, a CMP file is created, which details the errors found.

In the following example, a ‘>’ symbol was deleted accidentally. Note that this specific problem cannot occur if you use TagEditor for translation because TagEditor does not allow you to delete single characters from a tag. This example would only occur in a Word editing environment.

In the **Results** window, you should see a message similar to the one below for this file:



In the CMP file, you should see a message similar to the one below:



In this example, a ‘>’ sign has been deleted. This was the closing bracket for the internal tag < : /cs>.

To correct the error:

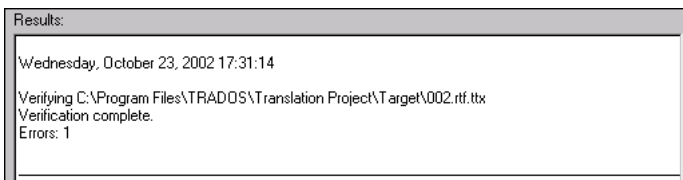
- 1 Open the target STF file in Word.
- 2 Insert a ‘>’ sign after the < : /cs statement in the target language RTF file.
- 3 Save the file and close it. Make sure that you close the CMP file as well.
- 4 In S-Tagger, run the STF file through the verification feature again.

Tag Errors

Tag errors usually occur because an external tag has been added, deleted or changed. They can also occur because the position of some important tag has been changed. Adding or deleting some internal tags can also cause tag errors. There are strict rules for adding and deleting tags. For more information, see “Working with S-Tags” on page 8-4. These rules must be followed when adding or deleting tags in the translation.

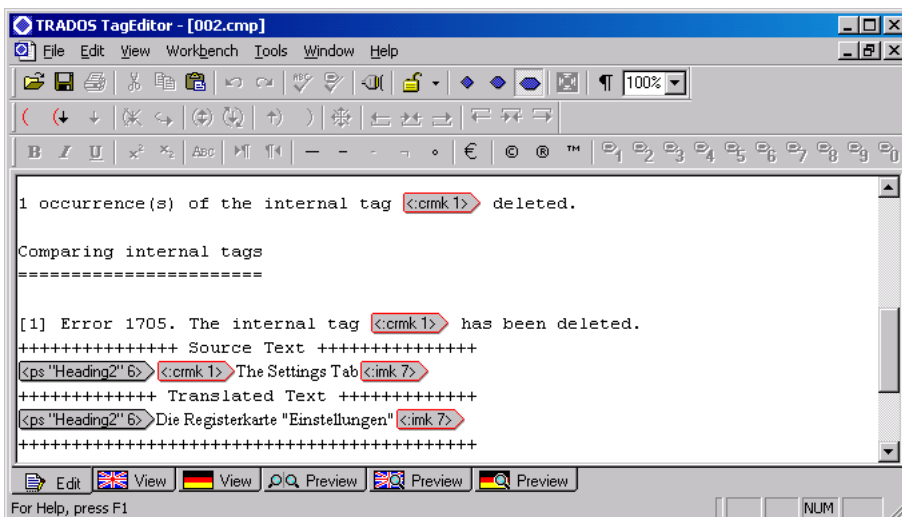
If a tag error occurs in your file, it is not possible to convert it back to IASCI. All errors must be corrected. When you have corrected the error, the files must be saved, closed, and verified again.

In the following example, an internal tag `<:crmk 1>` (representing a cross-reference marker) has been deleted accidentally. Note that this can happen frequently in Word. In TagEditor, this can only happen if tag protection has been turned off. If this happens, the message in the **Results** window for this file will be similar to the one below:



In this instance, one error has been reported. The `<:crmk ?>` tags may never be deleted, which is why the level is error, rather than alert or warning.

Open the CMP file in TagEditor or Word (depending on your editing environment) and you will see a message similar to the following:



The error message shows you what the segment looks like in the source STF file and in the target STF file.

To correct the error:

- 1 Keep the CMP file open and open the relevant target file in TagEditor or Word.
- 2 Find the relevant text in the STF file (using TagEditor or Word's Find feature) and insert the correct tag, either from the CMP file or from the source STF file, into the translated STF file.
- 3 Save the file and close all open files.
- 4 In S-Tagger, verify the file again.



NOTE

Some Japanese characters which are available to translators are not supported in Interleaf. If an unsupported character is inserted during translation, S-Tagger will detect the character and you will receive a message when you verify the files. For more information, see "Unsupported Characters" on page 8-15.

Tag Alerts

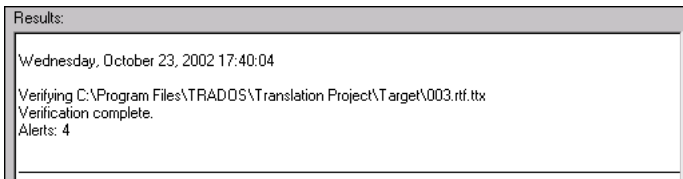
Tag alert messages refer to differences in the files which may have serious consequences for the file itself, but which will not prevent the file from being converted to IASCII. Alerts also serve as higher level warnings for tag differences that may be intentional.

In this example, we want to delete an index marker. Note that you must turn off the document and the tag protection in TagEditor to delete such tags. For more information on document and tag protection in Tag Editor, see the *Translator's Workbench User Guide*.

To delete the index marker:

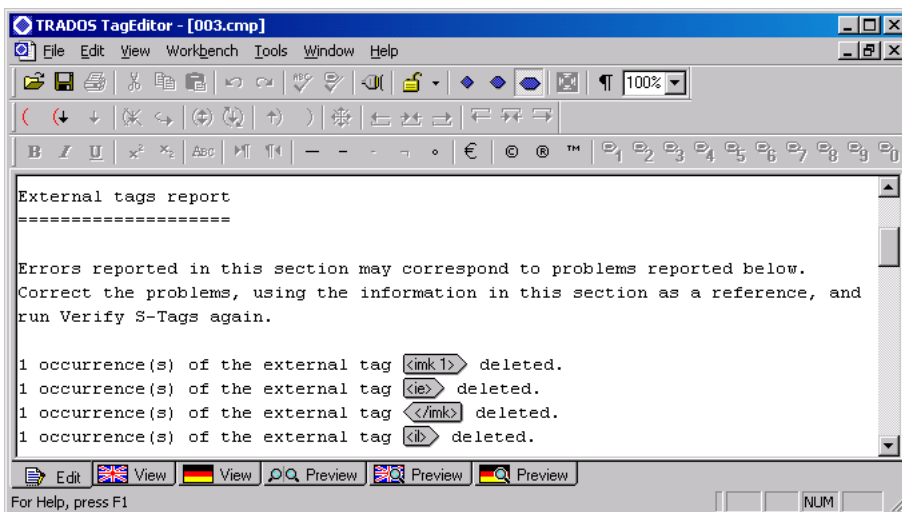
- 1 Open the target STF file in TagEditor or Word.
- 2 Delete the opening `<imk ?>` tag, each `<ie>` and `<il>` tag and the text of the index entry.
- 3 Delete any `<ss>` tags, as well as the closing `</imk>` tag. If you wish to delete an index subentry, delete the relevant `<il>` tag and text only.
- 4 In S-Tagger, verify the file from which the marker was deleted.

The message in the **Results** window will be similar to the one below:



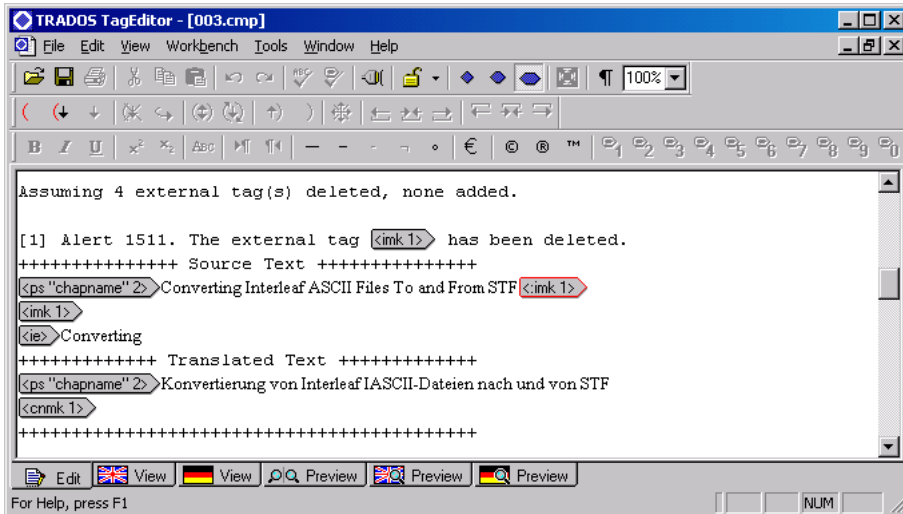
You can see from this **Results** window that there are four alerts reported. Consult the CMP file for full details of these alerts.

In the CMP file, the first section you encounter is the “External tags report”. This gives you summary information about which tags have been added or deleted. The external tags report will look similar to that shown below:



The next section in the CMP file, “Comparing External Tags”, lists the missing tags that have been detected. The section of the source STF that contains the relevant tags is shown, as is the section in the target STF from which the tags are missing. This helps you to quickly identify the issue and assess whether you need to fix anything.

You should see a message similar to the one below for each `<imk ?>`, `<ie>`, and `</imk>` tag deleted:



If the index marker was deleted in error, follow these steps to replace it:

- 1 Keep the CMP file open and open the relevant source STF file in TagEditor or Word.
- 2 Find the tag `<imk 1>`.
- 3 Copy the section from `<imk 1>` to `</imk>`.
- 4 Open the target STF file in the same editing environment as the CMP file.
- 5 Find the text where the marker should have been.
- 6 Insert a carriage return after the end of the last paragraph or tag.
- 7 Position the cursor on the empty line.
- 8 Paste the text from the source STF file into the target STF file.
- 9 Save the target STF file.
- 10 Translate the text of the index marker.
- 11 Close all files.
- 12 In S-Tagger, verify the file from which the marker was deleted.

Tag Warnings

These refer to tag differences between the files which usually relate to formatting or punctuation. Warnings can normally be ignored while the file is being converted, but a list of the warnings helps you to find formatting changes quickly.

There will often be many warnings in a translated file, as the formatting of the translation will probably be different to the formatting of the source language.

Two of the most common warnings are:

- ❑ Warning 1703. The internal tag <:tag> has been deleted.
- ❑ Warning 1707. The internal tag <:tag> has been added.

These warnings may appear quite often as you adapt the formatting of the text to fit the translation. If you add a <:b>, <:i>, <:bi>, <:c>, <:c1>, <:c2> or <:s> tag, you will not be informed of this, unless you have forgotten to add the closing tag or have turned on the reporting of these tag changes in the **Customise Verifier Report** dialog box.

You should check all warnings to ensure that the format change is intended. Include the CMP file, complete with suppressed alerts and warnings, with the delivery of the translated STF files.

➔ NOTE

You can find a complete list of the alerts and warnings in the S-Tagger for Interleaf Help.

Tag Formatting and Translator's Workbench

Some errors in the formatting of tags may be introduced by first-time users of Translator's Workbench. Most of these errors will be picked up by S-Tagger.

A common error occurs when the target text has been accidentally formatted as hidden text when working with Word and Translator's Workbench. Note that such a scenario does not occur if you are using TagEditor.

The example below illustrates a hidden text error that occurred using Word:

```
<ps "Body" 7>{0}With offices throughout the world, and a<:cs "TRADOS"
2>TRADOS<:/cs> product on the desk of more than 20,000 professional translators.<:cs "
TRADOS" 2>TRADOS<:/cs> is recognised as one of the market leaders in translation
technology.<0>{0}With offices throughout the world, and a<:cs "TRADOS" 2>TRADOS<:/cs>
product on the desk of more than 20,000 professional translators.<:cs "TRADOS"
2>TRADOS<:/cs> is recognised as one of the market leaders in translation technology.<0> ¶
```

Here you can see that the first piece of text in the file is the sentence which starts with "With offices throughout the world...". The source text is formatted as hidden text as indicated by the dotted underline.

If the *Hidden* attribute was accidentally applied to the target segment as well when the file was verified, a message similar to the one below would appear:

```
[1] ·Warning·1703·.·The·internal·tag·<:cs·"TRADOS"·2>·has·been·deleted·¶
[2] ·Warning·1703·.·The·internal·tag·<:/cs>·has·been·deleted·¶
[3] ·Warning·1703·.·The·internal·tag·<:cs·"TRADOS"·2>·has·been·deleted·¶
[4] ·Warning·1703·.·The·internal·tag·<:/cs>·has·been·deleted·¶
[5] ·Warning·1703·.·The·internal·tag·<:cs·"TRADOS"·2>·has·been·deleted·¶
[6] ·Warning·1703·.·The·internal·tag·<:/cs>·has·been·deleted·¶
[7] ·Warning·1703·.·The·internal·tag·<:cs·"TRADOS"·2>·has·been·deleted·¶
[8] ·Warning·1703·.·The·internal·tag·<:/cs>·has·been·deleted·¶
+++++·Source·Text·+++++¶
<ps·"Body"·7>With·offices·throughout·the·world,·and·a·<:cs·"TRADOS"·
2>TRADOS<:/cs>·product·on·the·desk·of·more·than·20,000·professional·translators,·<:cs·
"TRADOS"·2>TRADOS<:/cs>·is·recognised·as·one·of·the·market·leaders·in·translation·
technology.·<:cs·"TRADOS"·2>TRADOS<:/cs>·customers·include·such·diverse·organisations·as·
Microsoft,·Oracle,·Bertiz,·Nortel·Networks,·SAP,·Siemens,·PeopleSoft·and·Parametric·Technology·
Corporation.·Customers·from·the·government·and·non-profit·sectors·include·Amnesty·International,·
The·International·Monetary·Fund,·INTELSAT,·European·Investment·Bank,·the·European·Parliament,·
the·World·Intellectual·Property·Organization,·NATO·and·the·Zollkriminalamt.·More·than·150·
universities,·from·Finland·to·Chile,·also·use·<:cs·"TRADOS"·2>TRADOS<:/cs>·products·¶
+++++·Translated·Text·+++++¶
<ps·"Body"·7>¶
+++++¶
```

You can tell very quickly what has happened here, because the paragraph style tag in the translated text section is the same, but it is not followed by any text or tags.

To fix this error:

- 1 Open the target STF file in TagEditor or Word and check that non-printing characters are showing.
- 2 Remove the hidden formatting from the translated segment.



FOR MORE INFORMATION

Where an error occurs repeatedly and you cannot understand why, consult the online support center at support.sdl.com.

UPDATING THE TRANSLATION MEMORY

After translation and verification, your target STF files are still in bilingual format. You must ensure that all the bilingual data that currently exists in each STF file is transferred to your translation memory.

To save your work to the translation memory, use the **Clean Up** command in Translator's Workbench. This command is available from the **Tools** menu. Before using this feature, make sure that the **Update TM** option is selected.

When working with S-Tagger, the **Clean Up** command is only used to update the translation memory. This is because you can perform backward conversion on bilingual as well as monolingual files using S-Tagger.

The **Clean Up** command also modifies the bilingual file in the following ways:

- ❑ It removes hidden source text and segment delimiting marks from the target STF files.
- ❑ It restores the original colouring of your text; this only applies if you have used the **Translated Text Colours** option in Translator's Workbench when using Word.
- ❑ It changes the file extension to RTF.

Translator's Workbench produces a monolingual RTF file after clean up. The production of the RTF is essentially a by-product of the translation memory update. If you have begun the translation workflow with a TRADOSTag (TTX) file, then you should retain the most recent copy of this file before clean up.

You can use the monolingual RTF or the bilingual TTX file for the next workflow event, conversion back to the original file format (IASCII). If you began your workflow with an RTF, we recommend that you use the monolingual target RTF file. However, if you began your workflow with a TRADOSTag (TTX) file, we recommend that you use the bilingual TTX files for conversion back to IASCII.



FOR MORE INFORMATION

- ❑ For more information on workflows, see "S-Tagger Workflow: An Overview" on page 2-10.
- ❑ For more information on cleaning your files, see the *Translator's Workbench User Guide* and the TagEditor Help.



STF TO IASCI2 CONVERSION AND POST-PRODUCTION

This chapter explains the conversion and post-production tasks for translated Interleaf files.

Chapter sections include:

- ❑ conversion to original format
- ❑ general post-production tasks
- ❑ Japanese language post-production.

Chapter

10

CONVERTING STF TO IASCI

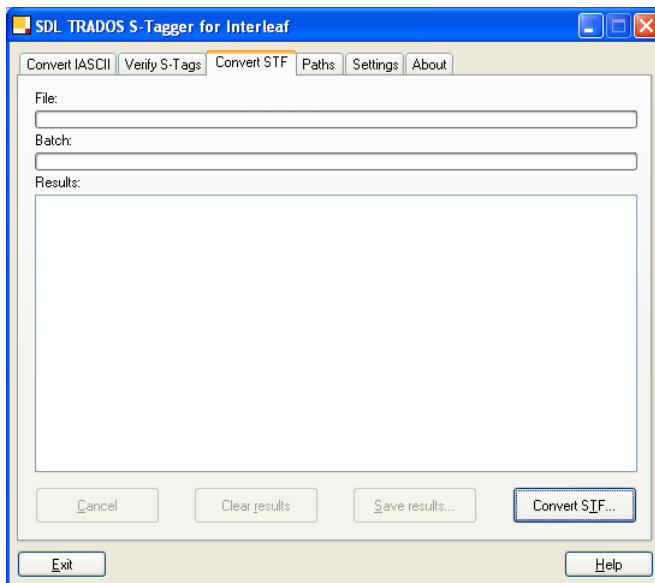
After translating and verifying your files, you can now convert them back to IASCI format. This section describes how to use S-Tagger to convert the STF files, singly or in batches.

Conversion will only be successful if you have:

- ❑ corrected all the errors in the files noted during the verification process
- ❑ retained the original source STF and ORG files in their specified folder.

To convert your STF files to IASCI:

- 1 In S-Tagger, open the **Convert STF** tab.



- 2 Click the **Convert STF** command button under the **Results** window.

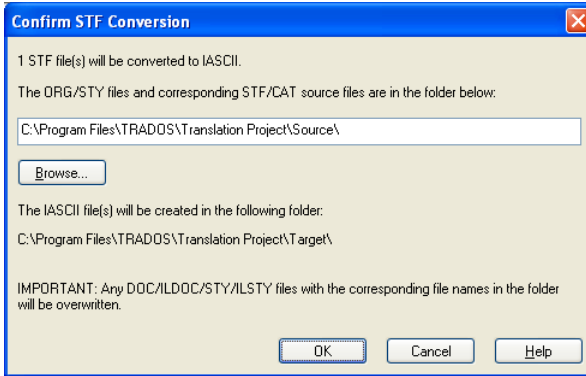
The **Select STF File(s) to Convert** dialog box is displayed, prompting you to select the translated STF file you wish to convert to IASCI. Note that the type of file displayed in the dialog box depends on the **STF file format** option selected on the **Settings** tab. For example, if you have selected the **TRADOS tag (TTX)** option, only TTX files are displayed here.

- 3 Select the file(s) to convert and click **Open**.

→ NOTE

You do not need to select the ancillary file as S-Tagger automatically inserts the text from this file into the correct files.

The **Confirm STF Conversion** dialog box is displayed, prompting you to confirm the file(s) to be converted to IASCII and detailing the path where it expects to find the ORG files and the original source STF files.



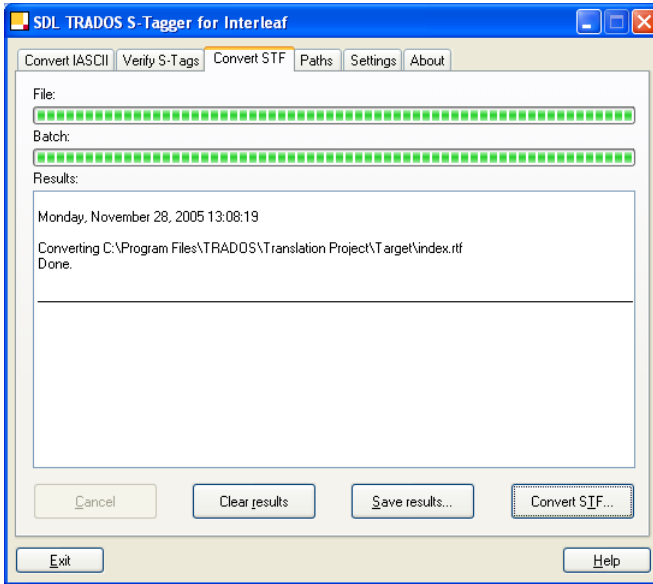
If you have selected the **Keep Paths** option on the **Paths** tab, S-Tagger automatically finds the folder where it last located STF, ORG, STY and CAT files. The **Keep Paths** settings information is not project-specific, but relates to the last use of the software.

- 4 If you wish to change the path that S-Tagger should follow to find the ORG files, click **Browse**. Locate the source STF, ORG, STY and CAT files and click **OK** to start the conversion.

! WARNING

If you do not select any CAT files, no new catalogs will be created and some shared items may be left untranslated.

As the files are being converted, an analysis of the process appears in the **Results** window, similar to the one below:



- 5 If you want to save the text in the **Results** window to a log file, you can do this by clicking **Save Results**. A dialog box is displayed, prompting you to specify the name of the file the results should be saved to. The log file is automatically given the extension `*.txt` and can be opened in any text editor. By default, this file is saved in the same folder (*Target*) as the converted IASCII files.

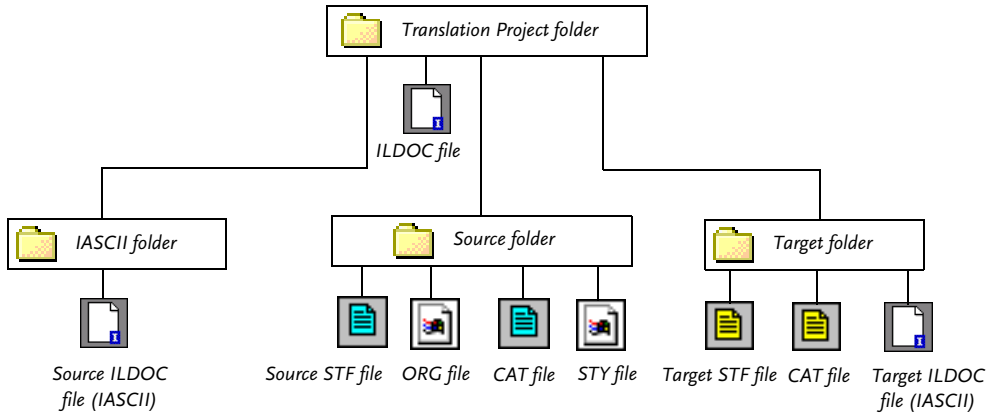
Correcting STF to IASCII Conversion Errors

During conversion, errors may appear in the **Results** window. An error occurs when S-Tags have been moved or changed in a way that renders the converted IASCII file invalid. For example, errors will occur if a closing tag has been deleted or if a tag has been invented and inserted into the STF file.

If an error occurs, run the verification feature to identify and locate the error before attempting to convert the STF files to IASCII again. For more information on verification, see “Verifying the S-Tags” on page 9-6.

If no errors occur during the conversion process, the files are converted and appear as IASCII files in the *Target* folder.

See the diagram below and compare it to the one provided in “Setting up the Project Structure” on page 7-4. Note that the translated, converted IASCII files have now been added to the *Target* folder.



➔ NOTE

- ❑ There may be an additional ancillary file in the *Source* and *Target* folders if you selected the **Create ancillary file** option on the **Settings** tab in S-Tagger.
- ❑ Your *Target* folder should also contain any subfolders, for example, *Art* (containing all the translated graphics), to replicate the structure of the translation project folder. This will ensure that all the references in the target files will function correctly.

After converting the files successfully to IASCII, you should open the IASCII file to check for any errors.

GENERAL POST-PRODUCTION

This section guides you through opening, checking, and saving the IASCII file(s) after conversion from STF. The following is a high-level overview of the steps that you must complete:

- 1 Open the file in Interleaf to check the error messages which may be displayed in the **Opening/Saving Messages** window.
- 2 When you have checked these error messages, you should then thoroughly examine each of the IASCII files, using the checklist provided in “Checking the New IASCII File” on page 10-8.
- 3 After checking the IASCII files, you can then save them back to ILDOC format.

Setting Up the Book Structure for the Interleaf Files

Before you open and save the new IASCII files, you will need to check the following:

Issue	What to Check
Folder structure	Is the folder structure correctly set up for the Interleaf files?
Files	Does the target book container contain all the summary and attribute files that the source book container contained? If not, you may get duplicate autonumber tokens and catalogs may not export correctly.
Book containers	Is the target book container positioned at the same level on the Desktop as the source book container?

Once you have checked the issues described in the table:

- 1 Copy the translated *.doc and *.sty, or *.ildoc and *.ilsty, files from the *Target* folder to the book (or other) container that the files were originally found in.

WARNING

If you copy and paste within Interleaf, you may get duplicate files rather than overwriting the existing files. Always use a file management utility to copy or move files.

- 2 Perform an update in Interleaf and then open the new IASCII files in Interleaf. Some of the errors described in “Opening the New IASCII File” on page 10-7 may occur.

Opening the New IASCII File

Because S-Tagger for Interleaf is a Windows application, it is recommended that, where possible, the new IASCII file is opened in Interleaf for Windows. Opening the new IASCII file in Interleaf for Windows rather than on any other platform reduces the amount of time needed for bug-fixing.

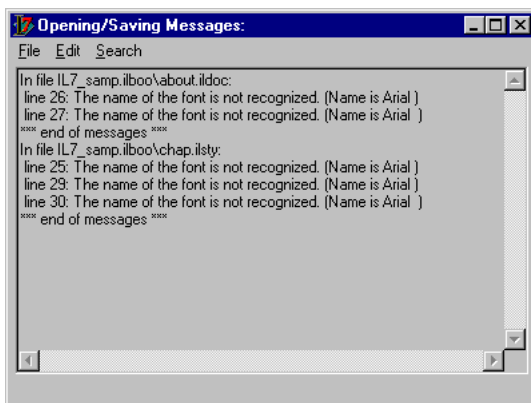
This section is written for users working in Interleaf for Windows. If you are opening the files in Interleaf for DOS or UNIX, the instructions on the following pages may not be relevant to you.

The following is a list of common issues found in the newly translated IASCII files:

- missing fonts
- missing languages

Missing Fonts

The errors may be due to missing fonts on your system. This message is not due to an error caused by S-Tagger.



If you are opening the IASCII files for any other purpose and you get this message, you should note the fonts which are missing from the **Opening/Saving Messages** window and close the IASCII file without saving it. Use the **LeafWare FontCheck** utility if it is available on your system to determine which fonts are used in the document.

Install the relevant fonts on your system and open the file again.

Missing Languages

You may be alerted to the fact that the document calls for a language dictionary different to the one installed with your copy of Interleaf. You should check the **Settings tab** to see what language you set as the document **Target Language**. You may need to install the additional language dictionaries on your system if you are opening the Interleaf files after translation.



NOTE

If any other messages appear in the **Opening/Saving Messages** window, consult your Interleaf/QuickSilver manual or Help file.

Checking the New IASCII File

After you have read the Interleaf error messages and made any necessary changes, you should also check the following issues:

Issue	What to Check
Text	Is all the required text present in the file? Is the page count similar to the original Interleaf file?
Formatting	Does the formatting in the new IASCII file reflect the formatting in the original Interleaf file?
Special characters/symbols	Have all the special characters or symbols converted correctly?
Art files and text	Are the art files and accompanying text in the correct position in the IASCII file?
Position of markers	Are the markers in the correct position? For more information on this issue, see “Position of Markers”.
Cross-references	Are there any NOTAG references in the IASCII file? If so, were they also present in the original Interleaf file? If not, recreate the original folder structure exactly to remedy this problem. You should also refer to the Interleaf/QuickSilver manual or Help.
Other features	Have you checked all the other features that you think might have caused problems in the file?

Position of Markers

If you selected **Don't insert** under **Marker placeholders** on the **Settings** tab in S-Tagger before converting the IASCII file to STF, the location of the index markers changes in the translated IASCII file. Open the new IASCII file with **All Default Markers** set to view. You will notice that each marker has been moved to the end of the paragraph in which it originally occurred.

If you are concerned about the position of markers and the effect this will have on a generated index, generate the index and compare it against the original Interleaf index. If you find that an index token refers to a paragraph which breaks over a page, make a note that the index token should be moved to another place in the paragraph when final pagination is being done. Remember that the pagination will probably change with translation.

Saving the IASCII File to ILDOC Format

You can now save the translated IASCII files as Interleaf ILDOC files to the correct folder structure.

To convert a IASCII file to ILDOC:

- 1 Open the IASCII file in Interleaf and make sure that any graphic references are pointing to your translated art folder.
- 2 Select the **Save As** command from the **File** menu and save the file, using the **Fast** option.
- 3 Update the cross-references, complete DTP tidy-up and pagination.
- 4 Repeat steps 1-3 for each target IASCII file in the book (if you are working with a book structure).
- 5 Generate the table of contents and the index.

JAPANESE LANGUAGE POST-PRODUCTION

This section contains general tips for post-translation production. It is intended specifically for users working with Japanese.

The following issues are examined in this section:

- ❑ character set tags
- ❑ unsupported characters
- ❑ language-specific features
- ❑ NO TAG references
- ❑ font mapping.

Character Set Tags

You may find instances of the inline component *STFWesternFont*, *STFCourier*, or *STFHelvetica*. These are character styles created by S-Tagger to allow you to retain the characteristics of European extended characters (those ANSI characters above 127, such as ‘ß’ or ‘à’) within the STF file. To retain Western extended characters in the translated IASCII file, insert character set tags (< : c> and < : /c>) around the character or word.

For example, if your documentation contains the product name ‘Qualité’ and you wish to retain the ‘é’ character in the product name in the final translation, use the character set tag as follows:
< : c>Qualité< : /c>.

Where characters above 127 appear formatted in either Courier or Helvetica fonts, apply < : c1> < : /c1> and < : c2> < : /c2>, respectively, to the text to retain the character and the font in the translated document. In the translated and converted Interleaf file, this text will have **STFCourier** and **STFHelvetica** in-line components applied to it.

If you find instances of characters which are appearing as nonsense in a file, check if the characters should have had the normal font encoding applied to them in the STF file and insert the < : c> tags where necessary.

Unsupported Characters Alert

If you have inserted any unsupported characters in the STF file, S-Tagger generates an alert message when you verify the files. The message will be similar to the following:

```
Alert 1201. The following Japanese character does not exist in
Interleaf and may therefore not be used: "character".
```

If you receive this message, you must replace the unsupported character with a supported one.

Language-specific Features

When working with Japanese, language-specific features, such as the component **Dictionary** or **Hyphenation** specifications, will be assigned “No Language”, since there is no Japanese Dictionary.

The index document sort order will have to be applied in the Interleaf file, according to the required style of the index document.

NO TAG References

If there are any ‘NO TAG’ references in the documents when they are opened in a Japanese version of Interleaf, the ‘NO TAG’ references will not be correctly displayed until a Japanese font has been applied.

Using Font Mapper for Interleaf

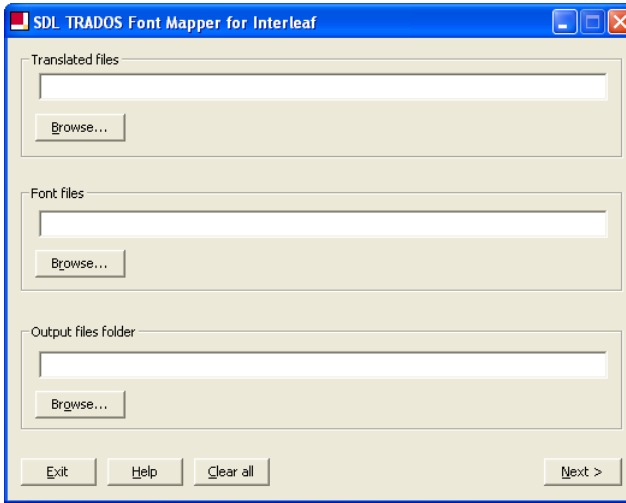
Font Mapper for Interleaf facilitates the process of changing fonts in IASCII files that are translated into or from Japanese. Font Mapper for Interleaf maps, or replaces, the fonts in the original document with fonts that you specify. Other attributes, such as font size and some styles, can also be changed. These changes are referred to as font mappings.

You can map fonts for one document, save the settings to a font map file and load the file for the other documents in your book.

Mapping Fonts

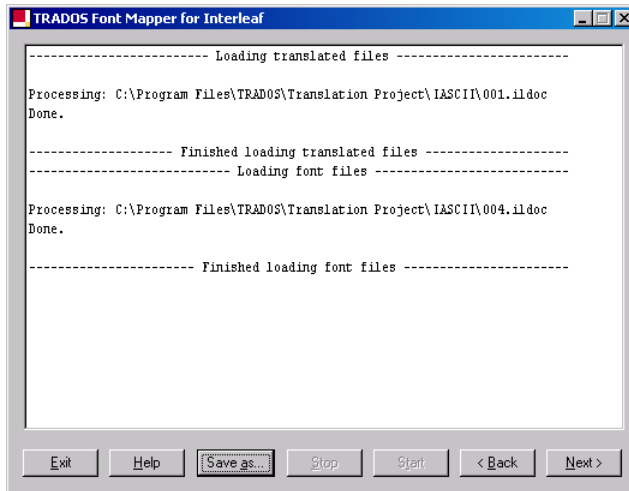
To map fonts:

- 1 Launch the Font Mapper for Interleaf.

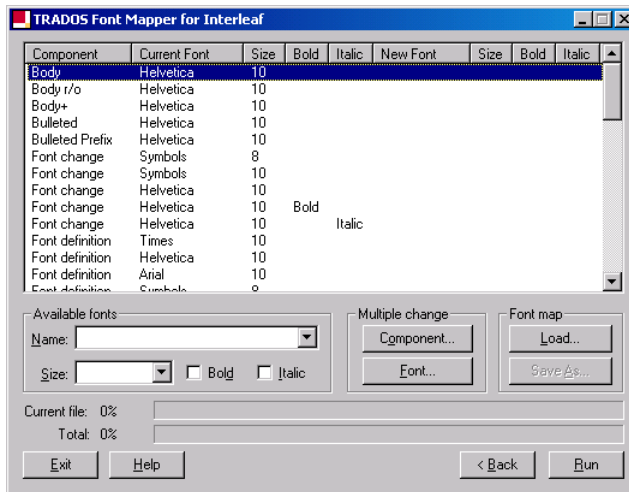


- 2 On this screen specify the location of the following files and click **Next** to continue:
 - Under **Translated files**, browse to select the IASCII files that require font changes.
 - Under **Font files**, browse to select the font files that contain the fonts that you wish to map. Font files may be either IASCII files that contain the fonts that you require or font map files that were created using Font Mapper for Interleaf.
 - Under **Output files folder**, browse to select the output folder where Font Mapper will place the font mapped files. You can either select an existing folder or create a new one specifically for that purpose.
- 3 On the second screen, click **Start**. Font Mapper loads and processes the files specified in the previous screen and records the results. The results are displayed on the screen.

- 4 Click **Next** to continue.

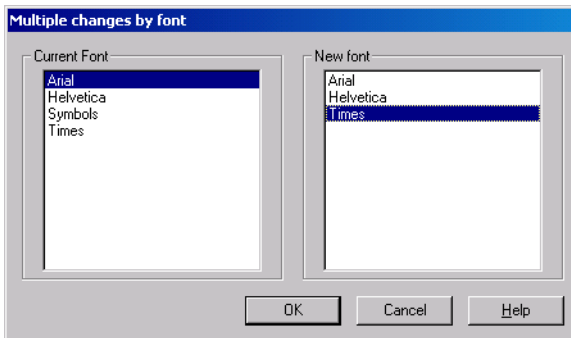


- 5 Use this third screen to specify the font mappings.



- The first column in the font mapping window shows you the name of the component. The second column, **Current Font**, shows the font name as it is in the translated file. This is followed by the font's other property columns: **Size**, **Bold** and **Italic**.
- The last four columns are for the **New Font**, with its **Size**, **Bold** and **Italic** properties. When a font has been mapped and these properties change, they are displayed in these columns.

- 6 You can now map the fonts globally or as they occur in the font mapping window.
 - ❑ To map individual instances of a font, select the font or format that you wish to map from the font mapping window. Then select a font from the **Available Fonts** drop-down list. Specify a font size and face if required.
 - ❑ To map fonts in batches, for example to map all instances of a font to another font, click **Font** in the **Multiple Change** box. The **Multiple Changes by Font** dialog box is displayed.



Select the font that you wish to map from the **Current Font** list. Select the font that you wish to map to from the **New Font** list. Click **OK**. In this example, all instances of *Arial* will be mapped to *Times*.


- 7 When you have completed all your font mappings, click **Run**. Font Mapper creates a new set of IASCII files in the output files directory. Open these files in Interleaf.



FOR MORE INFORMATION

For more detailed information on Font Mapper for Interleaf, see the Font Mapper Help.

You have completed mapping the fonts needed to display the translated IASCII file in Interleaf. You can now continue and open the IASCII file in Interleaf to complete DTP.



SECTION 4: APPENDICES

FRAMEMAKER TAGS AND EXAMPLES

This section presents examples of S-Tags as they are used within the STF files. If you are preparing or translating and editing STF files, you will find this reference information useful.

This appendix contains examples of FrameMaker features, their functions and the equivalent S-Tags.

Appendix sections include:

- ❑ S-Tags in the STF files
- ❑ S-Tags in the STF ancillary file
- ❑ a list of S-Tags used in STF files generated from MIF.

Appendix

A

S-TAGS IN THE STF FILES

S-Tagger uses brief coded statements known as S-Tags to represent formatting from the original FrameMaker MIF document in the converted STF file.



NOTE

'S-Tagger' throughout this section refers to S-Tagger for FrameMaker.

This section covers the S-Tags used within the STF files. The presentation of S-Tags within the ancillary file is described separately in “S-Tags in the STF Ancillary File” on page A-26.

The S-Tags and examples are described in the following order:

- ❑ anchored frames
- ❑ character formatting
- ❑ conditional text
- ❑ cross-references
- ❑ footnotes
- ❑ FrameMaker+SGML elements
- ❑ markers
- ❑ paragraph styles
- ❑ paragraph numbering formats (prefixes)
- ❑ tables
- ❑ unanchored frames
- ❑ variables.

Anchored Frames

Text in anchored frames can be in a text box or a text string. Text boxes can contain text that is formatted similarly to the rest of the document, but text strings are more limited.

The illustration below shows an excerpt from one of the SDL Trados FrameMaker sample files.



*Speed kills
– the message we need to pass on to the
children.*

50% of motorists ignore signals.



The text “Speed kills – the message we need to pass on to the children” is in a text box, with the text “50% of motorists ignore signals” appearing in a text string. This anchored frame is used in the following examples.

How an Anchored Frame Appears in an STF File

Anchored frames can be represented in a number of ways, depending on the frame content:

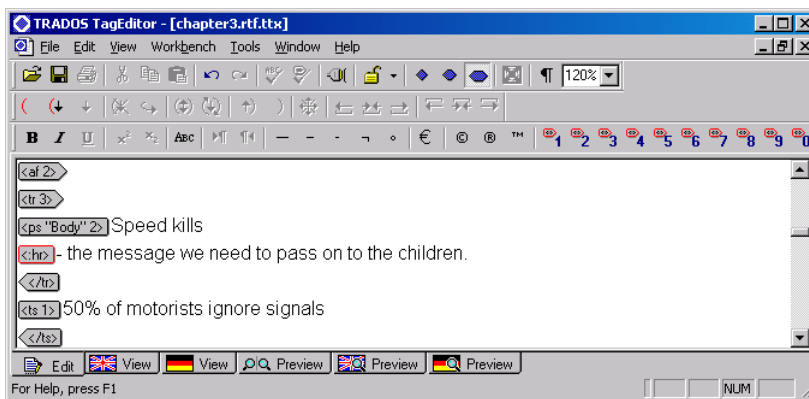
- ❑ If the anchored frame contains a text box, text string or both, it is represented with the external tag pair `<af ?>` `</af ?>`. An `<af ?>` tag must always be followed by its corresponding `</af >` tag in the appropriate position.
- ❑ If the anchored frame does not contain any text, it is represented with the internal tag `<:iaf ?>`.

The number assigned to each `<af ?>` and `<:iaf ?>` tag is unique within the file. There are never two `<af ?>` or `<:iaf ?>` tags with identical numbers.

Anchored frames can contain several items which must be represented in the STF files:

Anchor Frame Content	Representation in the STF File
Text boxes	<p><code><tr ?> </tr></code></p> <p>Paragraph style tags and all other tags allowed within text can appear within a <code><tr ?> </tr></code> sequence. Text placed in a text box inside an anchored frame, is treated by FrameMaker in exactly the same way as text in any other part of the document.</p>
Text strings	<p><code><ts ?> </ts></code></p> <p>Only character formatting tags can appear within a <code><ts ?> </ts></code> sequence. A text string can only contain one line of text. Text which is a series of text strings, joined to look like a paragraph, can cause problems in localisation as you need to be careful of the number of characters in each line. Many FrameMaker features, such as markers or footnotes, cannot be used in text strings.</p>

This illustration shows how the example anchored frame is represented in the STF file:



In this example, only the text “50% of motorists ignore signals” is a text string; the rest of the text is contained within a text box. In the STF file, you see the text box for the source FrameMaker file (`<tr 3>`), and then the text string (`<ts 1>`).

➔ NOTE

The text boxes and text strings are presented in this order because of their physical position in the frame. S-Tagger places the text item according to the co-ordinates of the items, starting with top left.

Character Formatting

There are two main ways in which character formatting can be changed within FrameMaker:

- ❑ character styles – you can create a character style which is given a name and properties. Every time a particular type of formatting is required, the originator can apply the character style to the text.
- ❑ font changes – you can make changes to character formatting by clicking the bold, italic, or underline buttons on the toolbar in FrameMaker or by changing the text properties of the individual characters.



FOR MORE INFORMATION

For more information, see “Character Set Tags” on page 6-10 and “Extended Characters and Symbols” on page 4-5.

How a Character Style Appears in an STF File

Because there are two ways of applying character formatting in FrameMaker, there are also two ways of representing it in STF.

When a character style is applied to a character or characters, the characters affected are preceded by the internal tag `<:cs "xxx" ?>` (where xxx is the name of the style). A character style tag is always followed by its closing tag `<:/cs>` in the appropriate place.



NOTE

Character style tags in index and other markers do not have a unique identifying number.

Example

The following sentence is found in a FrameMaker file:

The man likes **chocolate** and the woman likes *cake*.

The words ‘chocolate’ and ‘cake’ have character styles applied to them, using the character style *Bold* and the character style *Italic*. In the STF file, the sentence appears as:

```
<ps "para" 6>The man likes <:cs "Bold" 3>chocolate<:/cs> and the woman  
likes <:cs "Italic" 3>cake<:/cs>
```

If you wish, you can change where the character formatting begins and ends, making sure that an opening tag is always followed by a closing tag.

How a Font Change Appears in an STF File

Where a font change is made to a character(s) by changing the text properties or clicking a formatting button on the toolbar, S-Tagger inserts the internal tag `<:fc ?>`. The end of the formatting is represented by the corresponding closing tag `<:/fc>`. An `<:fc ?>` tag must always be followed by its closing tag in the appropriate position.

An `<:fc ?>` tag does not have a name. It is identified only by its number. Be careful if you are adding, deleting or moving an `<:fc ?>` tag. Ensure you have the correctly numbered tag and be sure to insert the closing `<:/fc>` tag when you want the text to return to the default font of the paragraph style.

Example

The following sentence is presented in a FrameMaker file:

The man likes **chocolate** and the woman likes *cake*.

A font change has been applied to the words 'chocolate' and 'cake', making them bold and italic respectively, compared to the rest of the text in the sentence. This font change is presented in the STF file as follows:

```
The man likes <:fc 2>chocolate<:/fc> and the woman likes  
<:fc 3>cake<:/fc>.
```

The tags in the sentence are internal opening and closing tag pairs, which represent the character formatting applied to the two words.

You can also add font changes using frequently used formatting tags. The following sentence is presented in a FrameMaker file:

The man walks the dog.

To apply bold and italic formatting to the words 'man' and 'dog', use the `<:b>` and `<:/b>`; `<:i>` and `<:/i>` formatting tags. The sentence appears in the STF file as follows:

```
The <:b>man<:/b> walks the <:i>dog<:/i>.
```

Conditional Text

FrameMaker allows you to generate multiple versions of a document or documents from one source by using conditional text. Text with a condition applied to it can be hidden or shown.



NOTE

If text is hidden in the FrameMaker file, it will not be available for translation.

How Conditional Text Appears in an STF File

- ❑ Conditional text that is set to **Show** in FrameMaker will appear in the STF file, preceded by the internal tag `<:cns "xxx" ?>`, where 'xxx' is the name of the conditional text style. A `<:cns "xxx" ?>` tag must always be followed by its closing tag `<:/cns>` in the appropriate position.
- ❑ Conditional text that is set to **Hide** does not appear in the STF file. The position of the hidden conditional text is marked by the internal tag `<:cnmk ?>`.

Example

A book is being written for two print versions, one for men and one for women. Most of the content is the same, some words or phrases will change slightly depending on the version. The sentence “Give the cake to the man” is presented in the male version and “Give the cake to the woman” is presented in the female version. Two conditional text styles have been set up: one is called *male*, the other is called *female*. To print or edit the male version, you hide the style *female* and vice versa. However, to translate both versions at the same time, both styles must show.

In FrameMaker, the sentence might be presented as:

Give the cake to the manwoman.

In STF, this would be presented as:

```
<ps "para" 6>Give the cake to the <:cns "male" 3>man<:/cns>
<:cns "female" 3>woman<:/cns>.
```



TIP

When creating such a sentence, copy the article (“the”) so that it is contained within the conditional text style tags, as in some languages the article is different for masculine and feminine. The correct sentence in the STF file would then read:

```
<ps "para" 6>Give the cake to <:cns "male" 3>the man<:/cns>
<:cns "female" 3>the woman<:/cns>.
```

Cross-references

In FrameMaker you can create a cross-reference to a page, or to text on a page. You can also create some more sophisticated cross-reference formats. These formats can pose problems for translation as the structure of the translated sentence may differ from that of the source sentence. Cross-references can be internal (to an item within the same file) or external (making reference to another file in a book).

FOR MORE INFORMATION

For more information, see “Unresolved External Cross-references” on page 6-7.

How Cross-references Appear in an STF File

Cross-references appear in the STF files within the internal tags `<:xr "xxx" ?>`, where ‘xxx’ is the actual text of the cross-reference as it appears in the FrameMaker file. The text that is being referenced is translated during the translation process and the cross-references are updated when the files are opened in FrameMaker after translation.

NOTE

When you open a MIF file, cross-references are not automatically updated. You must save your file as a FrameMaker document with an *.fm extension. For more information on updating cross-references in FrameMaker, see the FrameMaker Help.

Internal cross-references are automatically updated; external cross-references are not automatically updated until the files have been saved in FrameMaker format and are organised back into the original file structure.

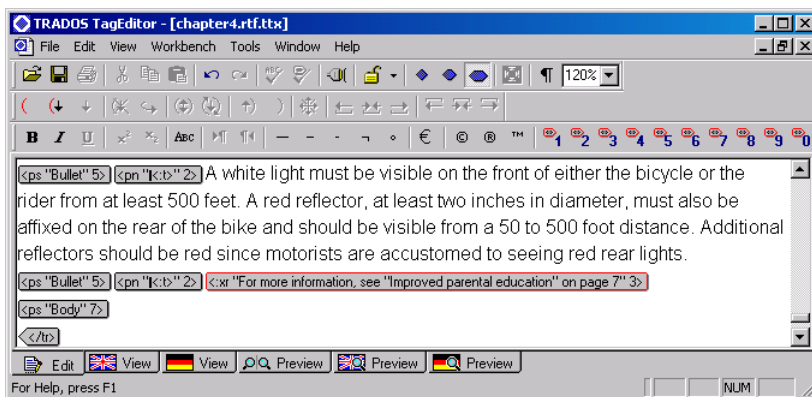
Any text within a cross-reference format, such as ‘See’ or ‘on page’, is presented in the ancillary file. You can translate the text of the cross-reference format in the ancillary file; this is then automatically inserted into the new MIF file when the STF file is converted back to MIF.

Example

In the illustration below, the sentence "For more information, see “Improved Parental Education” on page 7 " is a cross-reference to a section which appears later in the file:

- For more information, see "Improved parental education" on page 7

The following illustration shows how the cross-reference is presented in the STF file:



The entire cross-reference is contained within the `<:xr "xxx" ?>` tag. The referenced text, “Improved parental education”, is translated as a heading in the STF file. When the file is converted back to MIF and opened in FrameMaker, this text is automatically updated to show the translated text in the cross-reference.

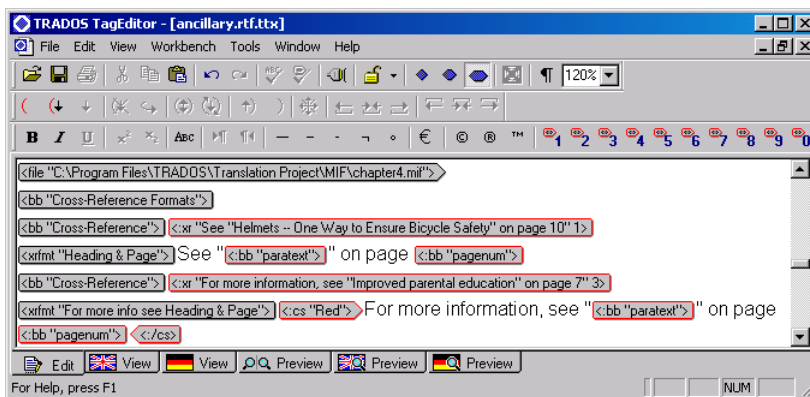
The text used in the cross-reference format must be translated separately in the ancillary file. You may also need to change the structure of the sentence.

If this is the first time a cross-reference of this particular type is encountered, you can search through the ancillary file for the text as it appears in the STF file.

In the ancillary file, go to the section of the file which is relevant to the STF file you are translating. You can find this by searching for the file name.

Next, go to the cross-references section, which starts with the tag `<bb "Cross-Reference Formats">`.

There, you will find the cross-reference description and text as shown below:



The tags represent building blocks used by FrameMaker to build the cross-reference. In this example, the building blocks 'paratext' and 'pagenum' have been used. You can translate the text not contained within these tags and then move the tags to change the structure of the cross-reference.

It is always better if the cross-reference is an entire sentence in itself because the entire cross-reference can be translated as a single unit, preserving the structure. However, this is not always the case. You need to visualise how the entire sentence will look after you have moved the tags around and be aware that you will need to change the sentence structure each time the cross-reference format appears in a sentence later on in the file.

Footnotes

Footnotes are created in FrameMaker by inserting a footnote reference marker into the text and then typing the text of the footnote where it appears at the bottom of the page.

How Footnotes Appear in an STF File

Footnotes are represented in the STF files with two types of tags:

- ❑ an internal tag < : fn ? > at the point where the footnote reference occurs in the text, and
- ❑ a series of external tags, surrounding the actual text of the footnote and positioned at the end of the paragraph in which the footnote reference has been placed.

The number '?' found in the internal tag matches the number of its corresponding external tag <fn ?>. The footnote text can contain any FrameMaker features, such as paragraph styles, character styles, index markers, and even graphics.

→ NOTE

You cannot add a footnote to an STF file. If you wish to create a new footnote, you must do this in FrameMaker after the translated files have been converted to MIF.

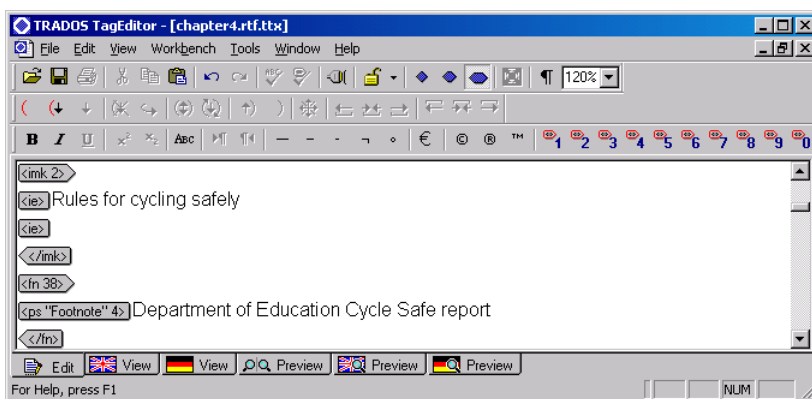
Example

In the example below, there are two footnotes in the paragraph:

10 Rules for Safe Cycling¹

In addition to wearing a helmet to ensure safety, bicycle riders should abide by some simple safety rules. Following these rules can decrease the risk of injury.²

The first footnote, which appears after the text, is presented as follows in the STF file:



When you are translating this sentence, you may need to move the `< : fn ? >` tag in the translated sentence to the correct position within the translation.

! WARNING

Do not move the `<fn ?>` to `</fn>` tags from their position at the end of the paragraph which contains the footnote reference.

FrameMaker+SGML Elements

FrameMaker+SGML is a version of FrameMaker which is used when you want to apply a strict structure to the FrameMaker documents and/or save the documents in SGML format.

How FrameMaker+SGML Elements Appear in an STF File

In STF, SGML elements are presented within the tags like this:

Tag	Represents
<code><el "xxx" ?></code> or <code><:el "xxx" ?></code>	An element without a font change
<code><elf "xxx" ?></code> <code><:elf "xxx" ?></code>	An element which is followed by a font change
<code><:ep "xxx"></code>	An element prefix
<code><:es "xxx"></code>	An element suffix

The different `<el "xxx" ?>` and `<elf "xxx" ?>` tags have been created to minimise the number of tags found in the document. All element structures are maintained. When translating, you are restricted in the number and type of tag changes that can be made in an STF file generated from a FrameMaker+SGML file.

Example

The example below shows a paragraph from a FrameMaker+SGML file:

```
<ps "Body" 1>
<el "paragraph" 3>The zoo was full of <:el "animal" 5> Elephants
<:/el "animal"> and <:el "animal" 5> Giraffes<:/el "animal">.
</el "paragraph">
```

If any of the elements are followed by a font change, S-Tagger includes the font change in the element tag. For example, a font change occurs after the element tag `<:el "animal" 5>`. When this happens, the element tag will appear as `<:elf "animal" 5>` instead of `<:el "animal" 5>`. The closing tag will also be `<:/elf "animal">`.

 **NOTE**

- ❑ Closing tags do not have a unique number.
- ❑ A closing tag should always follow an opening element tag.

Element Prefixes and Suffixes

As in normal FrameMaker files, there can be text within paragraph numbering formats in FrameMaker+SGML files. In addition to this, text may appear in an element prefix or suffix.

The format of the prefix or suffix is taken from the **Element Catalog**. You translate the text in the ancillary file, in the `Element Prefix/Suffix Formats` section. SGML element tags can be internal and external. The element name is included in both the opening and closing tag.

Markers

Markers are used in FrameMaker to automatically generate a list or index from a set of FrameMaker files. The most common markers that appear in FrameMaker files are index markers.

Index and other markers have a maximum length of 255 characters. During translation, the number of characters in the marker normally increases. For this reason, S-Tagger warns you if you try to convert a MIF file containing a marker which has more than a specified number of characters in it.

The default setting is 220 characters, which is satisfactory for most Western European languages. We recommend 150 characters for Asian languages. If a warning appears, you should isolate the marker, ascertain if it contains translatable text and, if so, split it into two or more markers.

 **FOR MORE INFORMATION**

For more information, see:

- ❑ “Index Markers” on page 4-11
- ❑ “Splitting Index Markers” on page 4-12
- ❑ “Index Marker Sort Levels” on page 4-16
- ❑ “Position of Markers” on page 6-8.

How Index Markers Appear in an STF File

Index markers are represented in STF by a series of tags:

Tag	Represents
<code><imk ?></code>	The beginning of an index marker is represented by the tag <code><imk ?></code> . This is followed by at least one <code><ie></code> tag.
<code><ie></code>	The <code><ie></code> tag represents an index entry. There may be multiple <code><ie></code> tags within any one marker. The <code><ie></code> tag is followed by the text of the marker.
<code><il></code>	Subentries are represented by the tag <code><il></code> , for index level. There may be multiple <code><il></code> tags following each <code><ie></code> tag; <code><ie></code> and <code><il></code> tags do not have closing tags.
<code><s1></code>	Sort level. Deleting a sort level tag and its text replaces the sort level text with nothing. If you wish to delete a sort level entry, delete only the text and leave the tag in place.
<code><ss></code>	If a sort string has been defined for an index marker, this is represented by an <code><ss></code> tag which is followed by the actual sort string. The sort string may be one or many characters. If there is a sort string for an index level, it appears after an <code><s1></code> tag in STF.
<code><:bb "xxx"></code>	Any other index properties, such as page range, are represented by the internal tag <code><:bb "xxx"></code> . The <code><:bb "xxx"></code> tag represents a building block. When an index entry spans across a page range, you will see the tag <code><:bb "startrange"></code> at the point where the page range begins and the tag <code><:bb "endrange"></code> at the point where it ends. Other popular building blocks used in markers include "nopage" and "singlepage".
<code><:cs "xxx"></code>	Character style changes within markers are represented by the internal tags <code><:cs "xxx"></code> to <code><:/cs></code> , similar to character style tags within the text but without the unique identifying number. Always remember to insert the closing tag.

→ NOTE

- ❑ If you have chosen to insert marker placeholders, the placeholder will be an internal tag `<:imk ?>`, whose number corresponds to the number of the `<imk ?>` tag following the paragraph.
- ❑ If you are assigning a sort string or a sort level to any entry, you should try to sort under an entire word rather than a single character. If you assign a single character, this character may have to change when the text is translated.

Example 1: Simple Index Marker

This is an example of an index marker in its simplest form in an STF file.

```
<imk 3>
<ie>Road safety
</imk>
```

The first part of the index marker is the `<imk ?>` tag. This shows you that an index marker is about to start.

The second part of the index marker is the `<ie>` tag. This is followed immediately by the actual text of the index entry (`<ie>` represents index entry). There can be multiple entries in any index marker.

The third part of the index marker illustrated above is the `</imk>` tag, which represents the end of that index marker.

This index marker would result in a generated index like the following:

```
R
Road Safety
```

Example 2: Index Marker with Subentries

An index entry can include subentries. In the STF files, each subentry is presented with an `<il>` tag in front of it. An `<il>` tag followed by another `<il>` tag means that the text following the second `<il>` tag is a subentry to the text of the first `<il>` tag. If you wish to make a second subentry to an index entry, you must insert a new `<il>` tag and text.

```
<imk 1>
<ie>Road safety
<il><:cs "italic">education</cs>
</imk>
```

This index marker would result in a generated index like the following:

```
R
Road safety
  education, 3
```

The `<il>` entry shows that the word ‘education’ is a subentry of the entry ‘Road safety’. The `<:cs "italic">` tag applies italic formatting to the subentry.

Example 3: Index Marker with Sort String Entries

If an `<ie>` tag and text is followed by the tag `<ss>` and additional text, this means that the index entry should be sorted under some letter other than the first letter of the index entry.

For example, if the index entry ‘Road safety’ should always be sorted under S for safety, then the sort string for the index entry ‘Road safety’ would be ‘safety’. Such an index entry would be represented in the STF file as:

```
<imk 1>
<ie>Road safety
<ss>Safety
</imk>
```

The resulting index would contain the entry ‘Road safety’ under S instead of R.

Sort string entries should consist of an entire word rather than a letter. However, if the author has entered only a letter, check the hard copy of the index to determine the letter under which the entry should be sorted in the target language.

Index Marker Sort Levels

When translating index markers into an Asian language, you may need to insert sort strings and sort levels. Alternatively, you may need to alter an existing sort order to suit the translation. For more information, see “Index Marker Sort Levels” on page 4-16.

Other Markers

Text in cross-reference, hypertext and conditional text markers is not presented for translation nor are the markers moved from their position within the text, whether or not marker placeholders are inserted.

All other marker types follow the rules for index markers. Other markers are represented by the internal tag `<:mk ?>` when marker placeholders are inserted. The actual text of the marker is found in between the tags `<mk "xxx" ?>` and `</mk>` at the end of the paragraph in which the marker

was originally placed. The 'xxx' in the `<mk "xxx" ?>` tag is the name of the marker. From FrameMaker 5.5 onwards, you can create user-defined markers and name them.

**WARNING**

When working with STF files generated from original FrameMaker+SGML documents, you may never add or remove paragraphs or index markers.

Paragraph Styles

In FrameMaker, each paragraph is assigned a paragraph style. The properties of each style are defined in the Paragraph Designer. Each paragraph style has a name and associated properties. Properties may be changed for any paragraph without giving the paragraph style a new name. This type of change is referred to as a local override.

Specific characters can be formatted differently inside a paragraph using character styles and font changes. Applying individual formatting in this way does not affect the paragraph style properties.

**FOR MORE INFORMATION**

For more information, see "Paragraph Styles" on page 4-10.

How Paragraph Styles Appear in an STF File

Paragraph styles are represented by the tag `<ps "xxx" ?>` where 'ps' is the tag type, 'xxx' is the paragraph style name, and '?' is the unique identifying number for the paragraph. For example, a paragraph style called 'Heading 1' might have the tag `<ps "Heading 1" 2>`.

The unique identifying number at the end of the tag is different when local overrides are applied to the individual paragraph. Sometimes, you may find a different unique identifying number when there is no obvious local override. This can be caused by FrameMaker's method of writing the MIF statement; the paragraph will not be affected when the file is converted back to FrameMaker format.

A paragraph style tag always starts at the beginning of a line. The text that follows it has that style applied to it, until another paragraph style tag is encountered.

When translating with paragraph styles in an STF file:

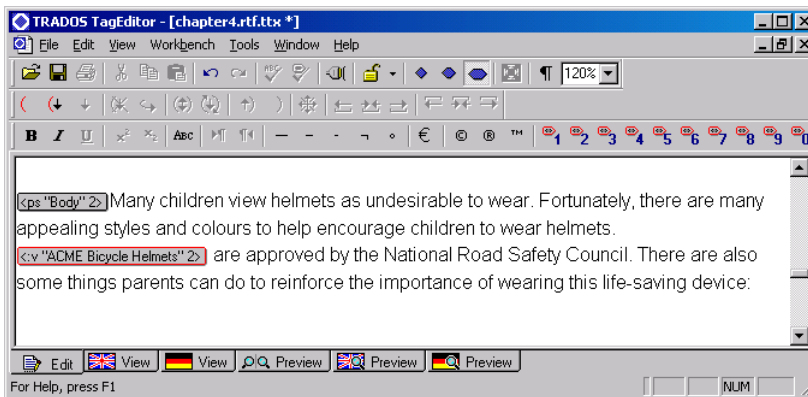
- ❑ Do not translate, or change, the text within the tag.
- ❑ Do not create a new paragraph style tag of your own. If you need to insert a paragraph style, you *must* copy an existing paragraph style tag from a different location in the document. This is because you may only insert paragraph styles that are already used in the document.
- ❑ Always leave the tag where it is and translate the text that follows it.

If you are using Word as your editing environment, do not forget to insert a '#' character at the beginning of an added paragraph style tag. If you forget to do this, this tag will be invalid and S-Tagger will not be able to convert the STF file back to MIF.

If you are using TagEditor as your editing environment, refer to the procedure outlined in "Paragraph Styles" on page 4-10. This will ensure that the tags you insert are treated as valid S-tags during verification.

Example

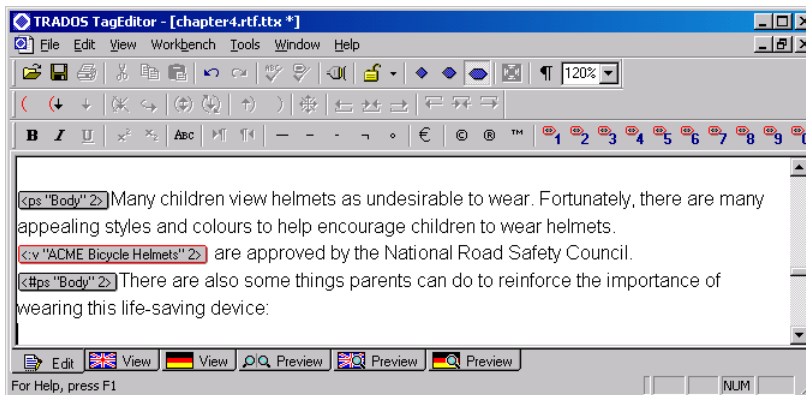
A paragraph similar to the one below is presented in the STF file:



To split this paragraph into two, create a new paragraph with the same paragraph style as the original paragraph.

As you can see from the illustration below, the paragraph has been split into two parts: the paragraph tag `<ps "Body" 2>` has been placed in front of the new paragraph and a '#' character has been placed at the beginning of the tag.

The '#' character is removed during the conversion back to MIF format.



Paragraph Numbering Formats

A paragraph style tag is often followed by a prefix tag. If you want a piece of text or a symbol always to precede the text of a particular paragraph style in FrameMaker, you insert the text in the paragraph numbering format. The text of a prefix is a property of the paragraph style and only needs to be translated once, which is why it is contained within a tag. The text of the prefix is translated in the ancillary file.

Prefixes often contain a number or a symbol, like a bullet. They are also used for note or warning text.



FOR MORE INFORMATION

For more information, see “Paragraph Numbering Formats (Prefixes)” on page 4-11.

How Prefixes Appear in an STF File

When a paragraph format includes a prefix, or text in the paragraph numbering format, the prefix is converted into the external tag `<pn "xxx" ?>`, where ‘pn’ is the tag type, ‘xxx’ is the actual text visible on the hard copy of the FrameMaker file and ‘?’ is the unique identifying number for the prefix.

When translating with prefixes in an STF file:

- Do not translate, or change, the text within the tag.
- Do not create a new prefix tag of your own. If you need to insert a paragraph style, you *must* copy an existing prefix tag from a different location in the document. This is because you may only insert prefixes that are already used in the document.

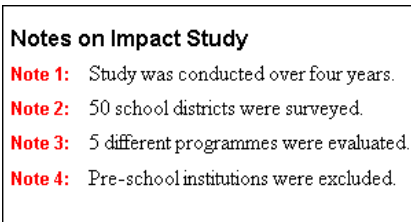
- ❑ If you are adding a paragraph which also has a prefix, you must add the prefix tag as well as the paragraph tag.
- ❑ Always leave the tag where it is in the STF file and translate the text that follows it.
- ❑ Translate any translatable text in the ancillary file. After translation is complete, the translated text in the ancillary file is automatically inserted into the new MIF file during conversion.

If you are using Word as your editing environment, make sure that you add a '#' character in front of both the <pn "xxx" ?> tag and the <ps "xxx" ?> tags. Otherwise, they will be invalid and will prevent S-Tagger from converting the STF files back to MIF.

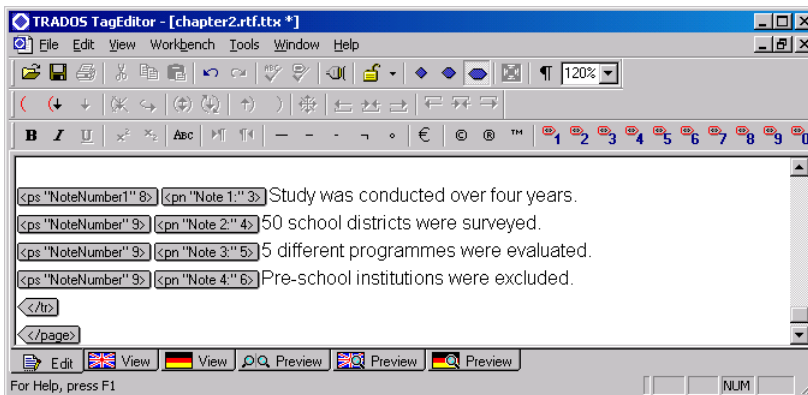
If you are using TagEditor as your editing environment, refer to the procedure outlined in "Paragraph Styles" on page 4-10. This will ensure that the tags you insert are treated as valid tags during verification.

Example

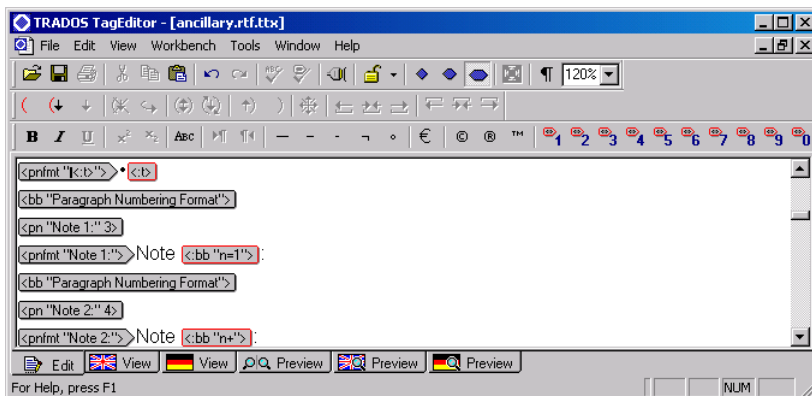
The illustrations below show how a bulleted list appears in a FrameMaker file and its corresponding STF file:



The text 'Note' in the <pn "xxx" ?> tag appears in the FrameMaker file as part of the prefix associated with the *NoteNumber1* and *NoteNumber* paragraph styles.



In the ancillary file, in the Paragraph Numbering Format section, you will see something like this:



The tag `<pnfmt "Note 1:">Note <:bb "n=1">:` represents the prefix property associated with the *NoteNumber1* paragraph style.

The second tag `<pnfmt "Note 2:">Note <:bb "n+">:` represents the prefix property associated with the *NoteNumber* paragraph style.

In this example, you would translate the text that is not contained within the tags, which in this case is the word 'Note'.

Tables

Tables in FrameMaker are user-defined. They may have custom ruling and shading applied to individual cells and/or rows and the text may straddle rows or cells. The text within the cells is exported to and presented in the STF file row by row and cell by cell. When translating tables, you only need to translate the text within the cells.

How Tables Appear in an STF File

A table begins with the opening tag `<tb ?>`, where '?' is the unique number for that table. It then continues with a sequence which includes at least one of the tag pairs in the table below until it reaches its corresponding `</tb>` closing tag.

Note that the tags in the STF files appear in the order in which they are listed.

Tag	Represents
<code><tblt> </tblt></code>	Title of the table
<code><tblh> </tblh></code>	Header row
<code><tblb> </tblb></code>	Body of the table
<code><hrow ?></code>	Hidden table row. Any rows hidden by means of a hidden conditional text style are represented by this tag. The text of hidden rows is not translatable.
<code><row ?> </row></code>	Table row
<code><ct ?> </ct></code>	Table cell. Each <code><ct ?></code> tag is followed by a paragraph style tag and then text, just like a regular body paragraph. You only translate the text that follows a <code><ct ?></code> tag.
<code><tblf> </tblf></code>	Table footer

Text within the table cells is formatted just like body text, with paragraph and character styles applied. Tables can also contain markers, graphics and footnotes within the cells and these are treated just like markers, graphics and footnotes in the main body text.

If custom ruling and shading has been applied to any cells or rows, this does not appear in the STF file, but is retained within the tag for when the file is converted back to MIF.

Tables which span pages are presented in the STF file as one continuous table with no page breaks.

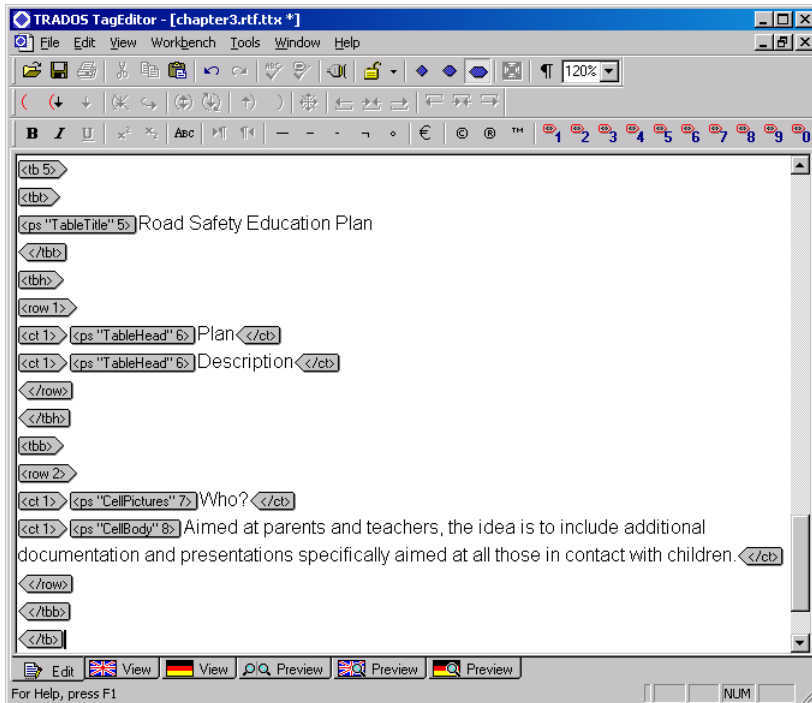
Example

The following table (abridged here for example purposes) appears in a FrameMaker file:

Road Safety Education Plan

Plan	Description
Who?	Aimed at parents and teachers, the idea is to include additional documentation and presentations specifically aimed at all those in contact with children.

In the STF file, this table appears something like this:



The tags used in the preceding example STF file are described in the following table:

S-Tag and Text

Represents

<tb 5>	Start of the table
<tbt>	Start of table title
<ps "TableTitle" 5>Road Safety Education Plan	Table title paragraph format and text of table title
</tbt>	End of table title
<tbh>	Start of the table heading rows
<row 1>	Start of the first (and only) heading row
<ct 1>	Start of the first cell in the heading row
<ps "TableHead" 6>Plan	Text in the first cell in the heading row with paragraph style "TableHead"

S-Tag and Text	Represents
<code></ct></code>	End of the first heading row cell
<code><ct 1></code>	Start of the next cell
<code><ps "TableHead" 6>Description</code>	The text in the cell
<code></ct></code>	End of the cell
<code></row></code>	End of the first (and only) heading row
<code></tbh></code>	End of the table's header rows
<code><tbb></code>	Start of the table's body rows
<code><row 2></code>	Start of the first body row
<code><ct 1></code>	Start of the first body row cell
<code><ps "CellPictures" 7>Who?</code>	The text in the cell with paragraph style "CellPictures"
<code></ct></code>	End of the cell
<code><ct 1></code>	Start of the next cell
<code><ps "Cellbody" 8>Aimed at parents and teachers, the idea is to include additional documentation and presentations specifically aimed at all those in contact with children.</code>	Text in the cell
<code></ct></code>	End of the cell
<code></row></code>	End of the row
<code></tbb></code>	End of the table's body rows
<code></tb></code>	End of the table

**TIP**

Use a hard copy of the original FrameMaker file for reference, so that you can easily see which cells you are translating.

Unanchored Frames

Frames are usually anchored to text, but unanchored frames may be used on reference pages to contain graphics or on body pages, to crop graphics that will not move with the text.

Unanchored frames are represented by the tag `<ufo ?>`.

Variables

In FrameMaker, a variable is a user-defined building block. When you create a variable, you enter text into a dialog box and then every time you insert that variable in the document, the same text is used. A common use of variables is in the creation of headers and footers.

How Variables Appear in an STF File

Both system and user variables appear in the STF file within the internal tag `<:v "xxx" ?>`, where 'xxx' is the text of the variable which is visible on the hard copy.

When translating with variables in an STF file:

- Do not translate the text within a variable tag, simply move it to another part of the sentence if required.
- Do not replace a variable with text unless it is absolutely necessary. Usually variables are used for items like software names or version numbers or even internal code names before software is shipped. If you replace the variable with text, you will make updating the file very difficult.

The text of the variable is translated in the ancillary file, and the text from the ancillary file is inserted into the new MIF file during conversion from STF to MIF.

Example

A variable called *Product Name* has been used to represent the name of a product throughout the FrameMaker files. In this case, the variable contains the text "ACME Bicycle Helmets".

The following text appears in a FrameMaker file:

```
ACME Bicycle Helmets are approved by the National Road Safety Council.
```

In the STF file, this text appears as:

```
<:v "ACME Bicycle Helmets" 2> are approved by the National Road Safety Council.
```

When you translate this sentence, move the variable tags `<:v "ACME Bicycle Helmets" 2>` so that it appears in the correct position in the translation.

In this case, the company would probably want to keep the product name (ACME Bicycle Helmets) in the translated versions, so this text does not have to be translated in the ancillary file.

S-TAGS IN THE STF ANCILLARY FILE

This section describes the use of S-Tags in the ancillary file. For more information on the ancillary file, see “The Ancillary File” on page 2-3.

When translating text in the ancillary file, follow these guidelines:

- ❑ Remember to translate only the necessary text, and not the tags in the ancillary file. If you are using TagEditor as your editing environment, the tag protection feature will not allow you to translate the text within tags.
- ❑ The ancillary file must always be verified, along with the rest of the STF files, to ensure that the tag structure and usage is correct.
- ❑ We recommend that you translate the text of the ancillary file before translating the rest of the text in the documentation set. This way, all translators working on a project will be able to see the translations in the ancillary file and the translations will be contained in the translation memory, if one is being used.

The ancillary file creates a separate section for each of the files that you convert to STF. The beginning of each file section is identified with a file name tag. For example:

```
<file "C:\Program Files\Translation Project\MIF\Chapter 2.mif">
```

Within each of these file sections, the ancillary file contains the following subsections:

- ❑ Cross-reference Formats
- ❑ Variable Formats
- ❑ Paragraph Numbering Formats
- ❑ Element Prefix/Suffix Formats
- ❑ Footnote Prefix/Suffix Formats
- ❑ Master Pages.

As S-Tagger converts the MIF files to STF, the appropriate ancillary text is inserted into each section. If there is no ancillary text in any of the files that were converted corresponding to a particular section, the section will remain empty.

Cross-reference Formats

The cross-reference formats section contains the cross-reference format for each type of cross-reference used within the particular file. For more information, see “Cross-references” on page A-8.

Variable Formats

With variables, translate the text you find after any `<vfmt "xxx">` tags. For more information, see “Variables” on page A-25.

Paragraph Numbering Formats

The first character of a paragraph numbering format is often a single character. This character, followed by a colon (:), is the flow specification and should never be changed. For prefixes, translate the text after any `<pnfmt "xxx">` tags. For more information, see “Paragraph Numbering Formats” on page A-19.

Element Prefix/Suffix Formats

Element prefixes or suffixes will only be present if the files have been converted from FrameMaker+SGML format. If they are present, translate the text after the `<epsfmt "xxx">` tags. If there are multiple ‘If’ statements or arguments within an element definition which contains a prefix or suffix, the alternative text will be presented for translation along with the text of the prefix or suffix which appears in the STF file. All text belonging to an element prefix/suffix format should be translated.

For example:

An element has the text ‘Discussion and example’, as a prefix. This prefix text is specified in the EDD. The EDD designer is also aware that there may not always be an example to follow the discussion and so allows the writer to insert only ‘Discussion’. All instances of this element in the document use the full prefix, rather than just the ‘Discussion’ prefix.

But in the ancillary file, you will see something like the following:

```
<bb "Element Prefix/Suffix"><:ep "Discussion and example">  
<epsfmt "Discuss">  
<bb "Prefix">Discussion and example  
<bb "Prefix">Discussion
```

Both options can be translated, but you need to be aware of the difference between them.

Footnote Prefix/Suffix Formats

Tags representing footnote prefixes and suffixes always appear in the ancillary file even if there are no footnotes in the MIF document. For more information, see “Footnotes” on page A-10.

Master Pages

If the master pages contain translatable text, it appears in the master page section

```
<bb "Master Pages">.
```

In this section, each master page is listed (where there is more than one) with the text that appears on that page. Translatable text appears as normal text. Translate any text that requires translation and save the file. When you convert the STF file to MIF, translated text will be inserted automatically in the correct location on the correct master page.



NOTE

Sometimes all text on the master page is generated text, for example, running headers and footers where the text derives from headings on body pages. In this case, there will be no master page text for translation in the ancillary file.

LIST OF S-TAGS

You will get an error message if you add or delete any tags from the following list. Note that the tag `<stf "F? . ??">` is not included in the list. This is because the S-Tagger for FrameMaker does not recognise the file as a valid STF file without the opening `<stf "F? . ??">` tag and therefore will not attempt to process it.

Tag	Closing Tag	Represents
<code><:cnmk ?></code>		Hidden conditional text marker
<code><:cns "xxx" ?></code>	<code><:/cns></code>	Conditional text style
<code><:crmk ?></code>		Cross-reference marker
<code><:fn ?></code>		Footnote reference marker
<code><:fmk1 ?></code>		Header/footer 1 marker
<code><:fmk2 ?></code>		Header/footer 2 marker
<code><:hmk ?></code>		Hypertext marker
<code><:iaf ?></code>		Internal anchored frame
<code><:ti ?></code>	<code><:/ti></code>	Text inset
<code><ac></code>	<code></ac></code>	Start of ancillary content
<code><af ?></code>	<code></af></code>	Anchored frame
<code><ancillarymode></code>		Ancillary mode setting in STF files
<code><oa "xxx" ?></code>	<code></oa></code>	Object attribute, used primarily inside MIF objects saved in FrameMaker 7.0.
<code><bb "xxx"></code>		Building block, used in the ancillary file
<code><ct ?></code>	<code></ct></code>	Table cell
<code><e "xxx"></code>		Element (tables and footnotes)
<code><el "xxx" ?></code>	<code></el "xxx"></code>	External element
<code><elf "xxx" ?></code>	<code></elf "xxx"></code>	External element with font change
<code><epsfmt "xxx"></code>		Element prefix/suffix format in the ancillary file

Tag	Closing Tag	Represents
<file "xxx">	</file>	File name, used in the ancillary file
<fn ?>	</fn>	Footnote
<hrow ?>		Hidden table row
<mk "xxx" ?>	</mk>	Marker of type xxx
<mk "Header/Footer#" \$>	</mk>	Header/footer marker Note: '#' is a number between 1 and 12. This number indicates the hierarchy level of the header/footer marker. '\$' is a number that refers to the sequence of the headers/footers in the file. Values can be 1 to n.
<page "xxx">	</page>	Page number xxx
<pnfmt "xxx">		Text in paragraph numbering format, used in the ancillary file
<row ?>	</row>	Table row
<sourcecharset "xxx">		Source document character set
<sourcelanguage "xxx">		Source document language
<sourcehyphenation "xxx">		Source document hyphenation
<sourcequotes "xxx">		Source document smart quotes
<sourcepath "xxx">		Source MIF file path
</stf>		End of STF file
<tb ?>x	</tb>	Table
<tbb>	</tbb>	Table body
<tbf>	</tbf>	Table footer
<tbh>	</tbh>	Table header
<tbt>	</tbt>	Table title
<tr ?>	</tr>	Text box

Tag	Closing Tag	Represents
<code><ts ?></code>	<code></ts></code>	Text string
<code><ufo ?></code>	<code></ufo></code>	Unanchored frame
<code><vfmt "xxx"></code>		Variable format, used in the ancillary file
<code><xrfmt "xxx"></code>		Cross-reference format, used in the ancillary file

You will get an alert if you add or delete any of the tags from the following list:

Tag	Closing Tag	Represents
<code><:bb "xxx"></code>		Building block
<code><:el "xxx" ?></code>	<code><:/el "xxx"></code>	Internal SGML element
<code><:elf "xxx" ?></code>	<code><:/elf "xxx"></code>	Internal SGML element with font change
<code><:imk ?></code>		Index marker placeholder
<code><:mk ?></code>		Marker placeholder
<code><:r1> - <:r39></code>		FrameMaker reserved characters
<code><:v "xxx" ?></code>		Variable
<code><:xr "xxx" ?></code>		Cross-reference
<code><ie></code>		Index entry
<code><il></code>		Index sub-entry
<code><imk ?> and </imk></code>		Index marker
<code><sl></code>		Sort level
<code><ss></code>		Sort string

Adding or deleting any other tag will generate a warning.



FOR MORE INFORMATION

For more information, see “Manipulating Internal Tags” on page 4-7 and “Manipulating External Tags” on page 4-9.



INTERLEAF TAGS AND EXAMPLES

This appendix contains examples of Interleaf features, their functions and the equivalent S-Tags.

Appendix sections include:

- ❑ S-Tags in the STF files
- ❑ S-Tags in the STF ancillary file
- ❑ a list of S-Tags used in STF files generated from IASCII.

Appendix

B

S-TAGS IN THE STF FILES

S-Tagger uses brief coded statements known as S-Tags to represent formatting from the original Interleaf IASCII document in the converted STF file.



NOTE

'S-Tagger' throughout this section refers to S-Tagger for Interleaf.

This section covers the S-Tags used within the STF files. The presentation of S-Tags within the ancillary file is described separately in “S-Tags in the STF Ancillary File” on page B-29.

The S-Tags and examples are described in the following order:

- ❑ anchored frames
- ❑ attributes and control expressions
- ❑ cross-references
- ❑ footnotes
- ❑ inline components
- ❑ markers
- ❑ paragraph styles (components)
- ❑ prefixes
- ❑ read-only components
- ❑ shared components
- ❑ shared frames
- ❑ shared named objects
- ❑ special and reserved characters
- ❑ tables
- ❑ unknown inline components.

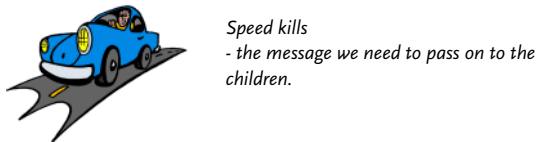
Anchored Frames

In Interleaf, as in most DTP packages, graphics are contained within frames. Callouts are also sometimes contained within frames.

In Interleaf, frames are often used to anchor text outside of the main text flow. You may see text in the hard copy which is not an obvious graphic, but which is represented in the STF file by a frame or a shared anchored frame tag. For more information, see “Shared Frames” on page B-22.

Text in anchored frames can be in a microdocument (text box), in a chart label or a text string. Microdocuments can contain text that is formatted similarly to the rest of the document, but chart labels and text strings are more limited.

The illustration below is a good example of the use of microdocuments in Interleaf:



50% of motorists ignore signals



The text “Speed kills – the message we need to pass on to the children” is included in microdocuments, with the text “50% of motorists ignore signals” appearing in a text string.

How an Anchored Frame Appears in an STF File

Anchored frames can be represented in a number of ways, depending on the frame content:

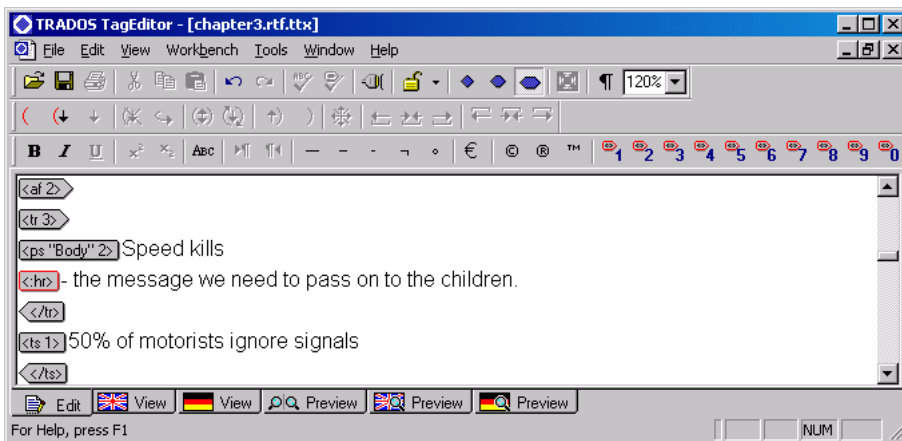
- ❑ If the anchored frame contains a text box, text string or both, it is represented with the external tag pair `<af ?>` `</af ?>`. An `<af ?>` tag must always be followed by its corresponding `</af ?>` tag in the appropriate position.
- ❑ If the anchored frame does not contain any text, it is represented with the internal tag `<:iaf ?>`.

The number assigned to each `<af ?>` and `<:iaf ?>` tag is unique within the file. There are never two `<af ?>` or `<:iaf ?>` tags with identical numbers.

Anchored frames can contain several items which must be represented in the STF files:

Anchor Frame Content	Representation in the STF File
Microdocuments	<p><code><tr ?> </tr></code></p> <p>Paragraph style tags and all other tags allowed within text can appear within a <code><tr ?> </tr></code> sequence. Text placed in a microdocument inside an anchored frame, is treated by Interleaf in exactly the same way as text in any other part of the document.</p>
Text strings	<p><code><ts ?> </ts></code></p> <p>Only character formatting tags can appear within a <code><ts ?> </ts></code> sequence. A text string can only contain one line of text. Text which is a series of text strings, joined to look like a paragraph, can cause problems in localisation as you need to be careful of the number of characters in each line.</p>
Chart labels	<code><t1 ?> </t1></code>

This illustration shows how the anchored frame above is represented in STF:

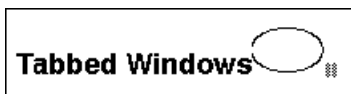


Anchored Frames without Text Objects

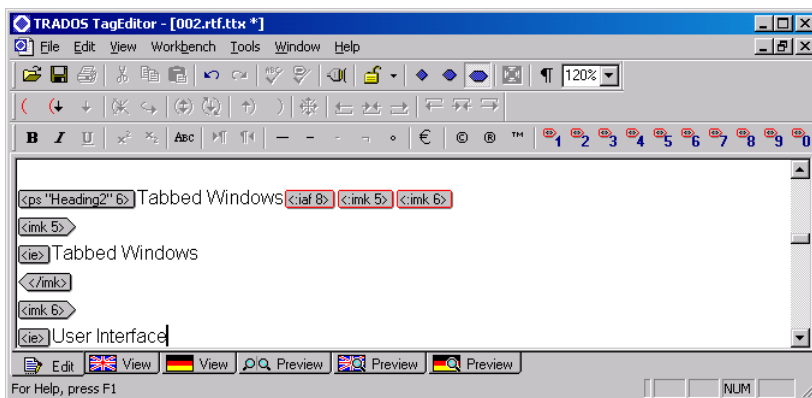
When an anchored frame does not contain any text objects or when it only contains text objects which are print locked, or outline text or equations, the frame will be assigned the internal tag `<: i:af ?>`. The tag is inserted at the point in the text where the frame anchor was found.

Example

The graphic below shows how an anchored frame with no text appears in Interleaf:



In the STF file, the anchored frame is represented by the `<:iaf 8>` tag:

**Anchored Frames Which Are Shared**

Shared anchored frames are represented by the tag `<:saf "xxx" ?>` or `<saf "xxx">`, where 'xxx' is the name of the frame. The `<:saf "xxx" ?>` tag is an internal tag and is inserted at the point in the text where the frame anchor was found. Any translatable text in the shared frame is found in the ancillary file, under the section `Text in Shared Frames`.

Interleaf allows the user to apply a shared property to a frame as a local override. Any frame with a shared property is treated as a shared frame.

FOR MORE INFORMATION

For more information, see "Shared Frames" on page B-22.

Attributes and Control Expressions

User attributes are applied when the Interleaf author wishes to show some text under certain conditions and different text under other conditions. The concept of user attributes corresponds to FrameMaker's conditional text features and is often referred to as 'effectivity' in Interleaf. User attributes are displayed or hidden by using a control expression.

Control expressions define which elements you want to include in a particular edition of your document. Always set the control expressions before saving the file to IASCI format. If the control expression is applied via a catalog, ensure that the catalog is in place and exporting the correct properties before saving the files.

How Control Expressions Appear in an STF File

- ❑ Control expressions that are set to **Show** in Interleaf will appear in the STF file, preceded by the internal tag `< :cns "xxx" ?>`, where 'xxx' is the name of the user attribute applied. If there are many user attributes applied to the subcomponent, only the last applied name will be shown.

Where the user attribute is applied on a component level and the component is showing, the fact that the component has a user attribute applied to it will not be identifiable in the STF file. Only inline components with user attributes applied will be identified as control expressions.

A `< :cns "xxx" ?>` tag must always be followed by its closing tag `< :/cns>` in the appropriate position.

- ❑ Control expressions that are set to **Hide** do not appear in the STF file. Instead, the position of the hidden text is marked by the tag `<cnmk ?>` if it is a component, or `< :cnmk ?>` if it is an inline component.

NOTE

The characters 'cns' are used because the concept of using attributes to show or hide text under certain conditions corresponds to FrameMaker's conditional text feature. S-Tagger for FrameMaker uses the tag `< :cns "xxx" ?>` to indicate conditional text, and the `< :cnmk ?>` tag to present conditional text markers. With S-Tagger for Interleaf, the same syntax is used to avoid confusion for translators working with both types of STF files.

Example

A book is being written for two print versions. One version is for men and one is for women. While most of the content is the same, some words or phrases will change slightly depending on the version.

The sentence "Give the cake to the man" is presented in the male version, and "Give the cake to the woman" is presented in the female version. Two conditional text styles have been set up; one is called

male, the other is called *female*. To print or edit the male version, you hide the style *female* and vice versa. However, to translate both versions at the same time, both styles must show.

In Interleaf, the sentence might be presented as:

Give the cake to the «man»«woman».

In STF, this would be presented as:

```
<ps "para" 6>Give the cake to the <:cns "male" 3>man<:/cns>
<:cns "female" 3>woman<:/cns>.
```



TIP

When creating such a sentence, copy the article ('the') so that it is contained within the conditional text style tags, as in some languages the article is different for masculine and feminine. The correct sentence in STF would then read:

```
<ps "para" 6>Give the cake to <:cns "male" 3>the man<:/cns>
<:cns "female" 3>the woman<:/cns>.
```

Attribute References

You can use an attribute reference to add a *continued* or other statement to a table header or footer, or place an attribute reference at any other point in an Interleaf file. If you use either method, the value of the attribute reference will not be available for translation in either the STF file or the ancillary file. The attribute reference will be represented by the tag `<:axr ?>` and you will be warned of its existence as you convert the file from IASCII to STF. Any attribute reference values must be translated in the Interleaf file.

Cross-references

In Interleaf, cross-references are made to auto or page numbers. They are represented in the STF files within the internal tags `<:xr "xxx" ?>`, where 'xxx' is the actual text of the cross-reference as it appears in the Interleaf file. If there is any text in the prefix or suffix of the auto or page number that is being referenced, it will be translated in the ancillary file and the cross-references will be updated when the files are opened in Interleaf after translation.

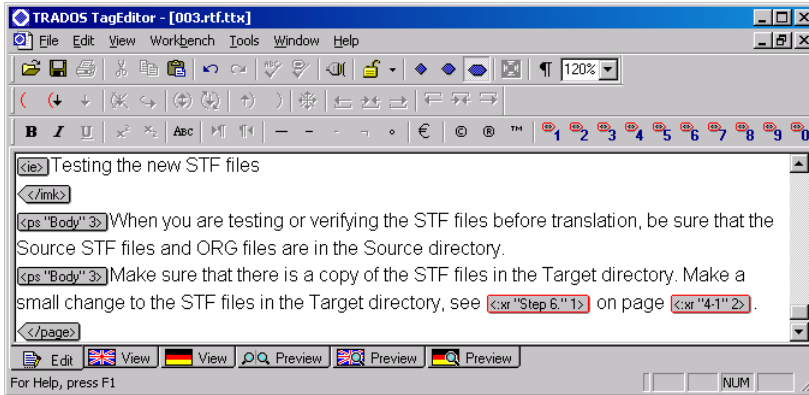
Cross-reference markers are not presented for translation, nor are they moved from their position within the text, even if marker placeholders are inserted. A cross-reference marker is represented by the tag `<:crmk ?>`.

Example

The following sentence is presented in an Interleaf file:

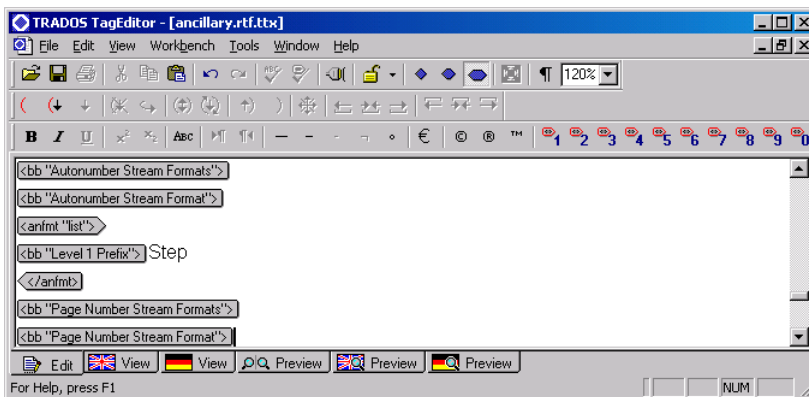
Make a small change to the STF files in the Target directory, see Step 6. on page 1-1

The following illustration shows how the cross-reference is presented in the STF file:



The text used in the cross-reference format must be translated separately in the ancillary file. You may also need to change the structure of the sentence.

In the ancillary file, go to the section of the file which is relevant to the STF file you are translating. You can find this by searching for the file name. In this example, you would then search for the word 'Step'. Under Autonumber Stream Format, you will see something like this:



The word 'Step' is translated during translation of the ancillary file and when the file is converted back to IASCII, the autonumbering format prefix is automatically updated.

Cross-reference Variables

Cross-reference variables (also known as variable content) are represented in STF by the external tags and text `<xrv "xxx" ?>suffix</xrv>`, where 'xxx' is the text of the component being referenced and 'suffix' is the text of the suffix applicable to that particular cross-reference variable. These cross-reference variables can have a suffix appended to them, for example, to put the word "continued" after the variable when the component that is being referenced spans more than one page.

Cross-reference variables are sometimes used in headers and footers. The cross-reference is made to the text of a component which is automatically updated when the translated file is converted back to IASCII. You only translate the suffix text.

Do not delete the text of the suffix just because it does not appear on the hard copy; it may not be necessary in the source language, but it may be required in the translation.

The text of the suffix may be a maximum of 20 characters only. Ensure that your translation fits within this limit.

The text of the suffix can be different for each instance of a cross-reference variable in any given file. For this reason, it is presented in the STF file, rather than the ancillary file.

Cross-reference Markers

Cross-reference markers will not be presented for translation, nor will they be moved from their position within the text, even if marker placeholders are inserted. A cross-reference marker is represented by the tag `<:crmk ?>`.

Footnotes

Footnotes appear in anchored frames in Interleaf. They are represented in the STF files with two types of tags:

- ❑ an internal tag `<:fn "x" ?>` at the point where the footnote reference occurs in the text
- ❑ a series of external tags, surrounding the actual text of the footnote and positioned at the end of the paragraph in which the footnote reference has been placed. The text 'x' and the number '?' found in the internal tag will match the text and the number of its corresponding external tag `<fn "x" ?>`.

Example

The Interleaf file contains a sentence with a footnote:

National Road Safety Report^I.

The text of the footnote, “2001 edition” is found at the bottom of the page. In the STF file, the paragraph and footnote would be represented as:

```
<ps "Body" 2>National Road Safety Report<:fn "1" 1>.  
<fn "1" 1>  
<af 8>  
<tr 8>  
<ps "micro:ftnote" 12><:xr "1" 1>.<:t>2001 edition  
</tr>  
</af>  
</fn>
```

Inline Components

Any component, or structural element, which is found within a paragraph (or component) is referred to as an inline component. Inline components are also sometimes referred to as subcomponents.

Inline components can be used to:

- ❑ control character formatting
- ❑ create a style other than a paragraph style which is included in the table of contents
- ❑ control conditional text content. For more information, see “Attributes and Control Expressions” on page B-6.

There are two main ways in which character formatting can be changed within Interleaf:

- ❑ character styles (inline components) – you can create a character style which is given a name and properties. Every time a particular type of formatting is required, you can insert the relevant inline component into the text.
- ❑ font changes – you can make changes to character formatting by clicking the bold, italic, or underline buttons on the toolbar in Interleaf or by changing the text properties of the individual characters.

**FOR MORE INFORMATION**

For more information, see “Character Set Tags” on page 10-10 and “Extended Characters and Symbols” on page 8-5.

How a Character Style Appears in an STF File

Because there are two ways of applying character formatting in Interleaf, there are also two ways of representing it in STF.

When a character style is applied to a character or characters, the characters affected are preceded by the internal tag `<:cs "xxx" ?>` (where xxx is the name of the style). A character style tag is always followed by its closing tag `<:/cs>` in the appropriate place.

Example

The following sentence is found in an Interleaf file:

The man likes **chocolate** and the woman likes *cake*.

The words 'chocolate' and 'cake' have character styles applied to them, using the inline component *Bold* and the inline component *Italic*. In the STF file, the sentence appears as:

```
<ps "para" 6>The man likes <:cs "Bold" 3>chocolate<:/cs> and the woman  
likes <:cs "Italic" 3>cake<:/cs>
```

If you wish, you can change where the character formatting begins and ends, making sure that an opening tag is always followed by a closing tag.

How a Font Change Appears in an STF File

Where a font change is made to a character(s) by changing the text properties or clicking a formatting button on the toolbar, S-Tagger inserts the internal tag `<:fc ?>`. The end of the formatting is represented by the corresponding closing tag `<:/fc>`. An `<:fc ?>` tag must always be followed by its closing tag in the appropriate position.

An `<:fc ?>` tag does not have a name. It is identified only by its number. Be careful if you are adding, deleting or moving an `<:fc ?>` tag. Ensure you have the correctly numbered tag and be sure to insert the closing `<:/fc>` tag when you want the text to return to the default font of the paragraph style.

Example

The following sentence is presented in an Interleaf file:

The man likes **chocolate** and the woman likes *cake*.

A font change has been applied to the words 'chocolate' and 'cake', making them bold and italic respectively, compared to the rest of the text in the sentence.

This font change is presented in the STF file as follows:

```
The man likes <:fc 2>chocolate<:/fc> and the woman likes  
<:fc 3>cake<:/fc>.
```

The tags in the sentence are internal opening and closing tag pairs, which represent the character formatting applied to the two words.

You can also add font changes using frequently used formatting tags. The following sentence is presented in an Interleaf file:

```
The man walks the dog.
```

To apply bold and italic formatting to the words ‘man’ and ‘dog’, use the `<:b>` and `<:/b>`; `<:i>` and `</i>` formatting tags. The sentence appears in the STF file as follows:

```
The <:b>man</b> walks the <:i>dog</i>.
```



WARNING

You may not add these tags to autonumber stream formats, page number stream formats, cross-reference variable suffixes, to text strings, to chart labels, or to index, sort string or ‘see also’ entries.

Markers

Markers are used in Interleaf to automatically generate an index or other automated list from a set of Interleaf files.

Index tokens and other markers have a maximum length of 255 characters. During translation, the number of characters in the marker normally increases. For this reason, S-Tagger warns you if you try to convert an IASCII file containing a marker which has more than a specified number of characters in it.

The default setting is 200, which is satisfactory for most Western European languages. We recommend 150 characters for Asian languages. If a warning appears, you should isolate the marker, ascertain if it contains translatable text and, if so, split it into two or more markers.



FOR MORE INFORMATION

For more information, see:

- ❑ “Index Markers” on page 8-11
- ❑ “Splitting Index Markers” on page 8-12
- ❑ “Index Marker Sort Levels” on page 8-16
- ❑ “Position of Markers” on page 10-9.

How Index Markers Appear in an STF File

Index markers are represented in STF by a series of tags:

Tag	Represents
<code><imk ?></code>	Index marker tag Deleting an <code><imk ?></code> tag deletes an index marker. If an <code><imk ?></code> tag is deleted, all following text and tags, up to and including the closing <code></imk></code> tag, must also be deleted.
<code><ie></code>	Index entry There may be multiple <code><ie></code> tags within any one marker. The <code><ie></code> tag is followed by the text of the marker.
<code><il></code>	Index entry subentry There may be multiple <code><il></code> tags following each <code><ie></code> tag; <code><ie></code> and <code><il></code> tags do not have closing tags.
<code><ss></code>	Sort string Deleting a sort string tag and its text replaces the sort string text with nothing. If you wish to delete a sort string entry, delete only the text and leave the tag in place.
<code><sa></code>	See also entry



NOTE

- ❑ If you have chosen to insert marker placeholders, the placeholder will be an internal tag `<:imk ?>`, whose number corresponds to the number of the `<imk ?>` tag following the paragraph.
- ❑ If you are assigning a sort string to any entry, you should try to sort under an entire word rather than a single character. If you assign a single character, this character may have to change when the text is translated.

Example 1: Simple Index Marker

This is an example of an index marker in its simplest form in an STF file.

```
<imk 3>
<ie>Road safety
</imk>
```

The first part of the index marker is the `<imk ?>` tag. This shows you that an index marker is about to start.

The second part of the index marker is the `<ie>` tag. This is followed immediately by the actual text of the index entry (`<ie>` represents index entry). There can be multiple entries in any index marker.

The third part of the index marker illustrated above is the `</imk>` tag, which represents the end of that index marker.

This index marker would result in a generated index like the following:

```
R
Road Safety, 3
```

Example 2: Index Marker with Sub-entries

An index entry can include sub-entries. In the STF files, each sub-entry is presented with an `<il>` tag in front of it. An `<il>` tag followed by another `<il>` tag means that the text following the second `<il>` tag is a sub-entry to the text of the first `<il>` tag. If you wish to make a second sub-entry to an index entry, you must insert a new `<il>` tag and text.

```
<imk 1>
<ie>Road safety
<il><:cs "italic">education<:/cs>
</imk>
```

This index marker would result in a generated index like the following:

```
R
Road safety
  education, 3
```

The `<il>` entry shows that the word ‘education’ is a subentry of the entry ‘Road safety’. The `<:cs "italic">` tag applies italic formatting to the subentry.

Example 3: Index Marker with Sort String Entries

If an `<ie>` tag and text is followed by the tag `<ss>` and additional text, this means that the index entry should be sorted under some letter other than the first letter of the index entry.

For example, if the index entry ‘Road safety’ should always be sorted under S for safety, then the sort string for the index entry ‘Road safety’ would be ‘safety’. Such an index entry would be represented in the STF file as:

```
<imk 1>
<ie>Road safety
<ss>Safety
</imk>
```

The resulting index would contain the entry 'Road safety' under S instead of R.

Sort string entries should consist of an entire word rather than a letter. However, if the author has entered only a letter, check the hard copy of the index to determine the letter under which the entry should be sorted in the target language.

Example 4: Index Marker with See Also Entries

An <sa> tag denotes a 'see also' entry. This means that the index entry should be followed by an entry directing the user to another entry.

For example, if the index entry 'Road Safety Education' should also direct the reader to the index entry 'National Reports', then the index entry would be represented in the STF file as:

```
<imk 1>  
<ie>Road Safety Education  
<sa>National Reports  
</imk>
```

The translation of the text 'see also' or 'see' is specified in the Interleaf dictionary file according to the language of that dictionary. If you wish to change the translation from the default, you must instruct the person responsible for generating the index in Interleaf to change the translation in the 'Sort' file in Interleaf.

Paragraph Styles (Components)

In Interleaf, each paragraph is assigned a paragraph style. Paragraph styles are called 'components' in Interleaf. The properties of each paragraph style (such as the default font for the paragraph, indents and spacing before) are specified in the **Component Properties** dialog box in Interleaf.

Each paragraph style has a name and properties. The properties can be changed for any paragraph in Interleaf, without giving the paragraph style a new name. This type of a change is called a local override.

Specific characters can be formatted differently inside a paragraph using character styles and font changes. Applying individual formatting in this way does not affect the paragraph style properties.

How Paragraph Styles Appear in an STF File

A paragraph style is represented in STF by the external tag <ps "xxx" ?>, where 'ps' is the tag type, 'xxx' is the paragraph style name, and '?' is the unique identifying number for the paragraph style. For example, a paragraph style called 'Heading 1' might have the tag <ps "Heading 1" 2>.

A paragraph style tag always starts at the beginning of a line. The text that follows it has that same style applied to it, until another paragraph style tag is encountered.

When translating with paragraph styles in an STF file:

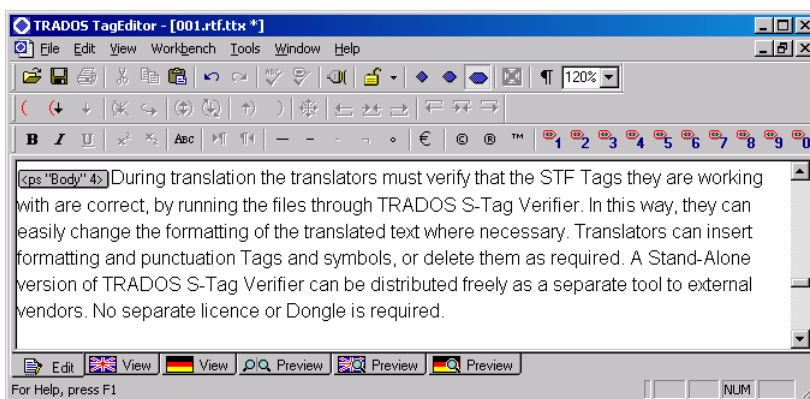
- ❑ Do not translate, or change, the text within the tag.
- ❑ Do not create a new paragraph style tag of your own. If you need to insert a paragraph style, you *must* copy an existing paragraph style tag from a different location in the document. This is because you may only insert paragraph styles that are already used in the document.
- ❑ Always leave the tag where it is and translate the text that follows it.

If you are using Word as your editing environment, do not forget to insert a '#' character at the beginning of an added paragraph style tag. If you forget to do this, this tag will be invalid and S-Tagger will not be able to convert the STF file back to IASCII.

If you are using TagEditor as your editing environment, refer to the procedure outlined in "Paragraph Styles" on page 8-10. This will ensure that the tags you insert are treated as valid tags during verification.

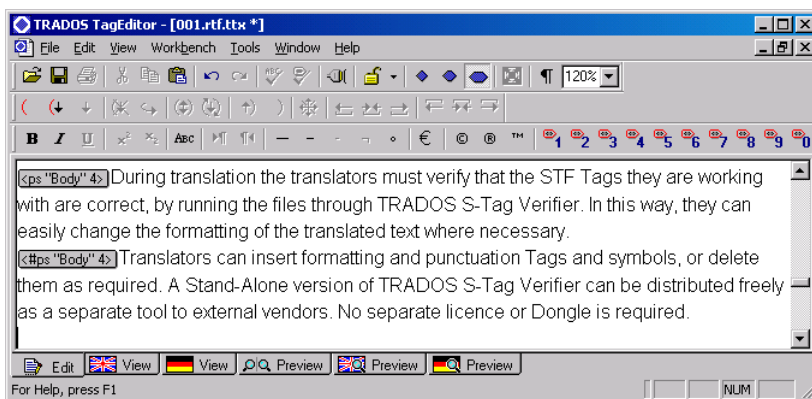
Example:

A paragraph similar to the one below is presented in an STF file:



To split this paragraph into two, create a new paragraph with the same paragraph style as the original paragraph and insert it.

As you can see from the illustration below, the paragraph has been split into two parts, the paragraph tag `<ps "Body" 4>` has been placed in front of the new paragraph and a '#' character has been inserted at the front of the tag.



The '#' character is removed automatically during the conversion back to Interleaf.

Prefixes

A paragraph style tag is often followed by a prefix tag. If you want a piece of text or a symbol always to precede the text of a particular paragraph style in Interleaf, you insert a prefix. The prefix contains this common text. The text of a prefix only needs to be translated once, which is why it is contained within a tag. It is a property of the paragraph style. The text of the prefix is translated in the ancillary file.

Prefixes often contain a number or a symbol, like a bullet. Prefixes are also often used for note or warning text. Prefixes are represented in STF files so that you can see, as you are translating, that you are in a numbered or bulleted list, according to the type of prefix.

How Prefixes Appear in an STF File

A prefix tag is represented in STF by the external tag `<pn "text" ?>`, where 'pn' is the tag type, 'text' is the actual text you will see on the hard copy of the Interleaf file, and '?' is the unique identifying number for the prefix.

When translating with prefixes in an STF file:

- ❑ Do not translate, or change, the text within the tag.
- ❑ Do not create a new paragraph style tag of your own. If you need to insert a paragraph style, you *must* copy an existing paragraph style tag from a different location in the document. This is because you may only insert paragraph styles that are already used in the document.
- ❑ If you are adding a paragraph which also has a prefix, you must add the prefix tag as well as the paragraph tag.
- ❑ Always leave the tag where it is in the STF file and translate the text that follows it.
- ❑ Translate any translatable text in the ancillary file. After translation is complete, the translated text in the ancillary file is automatically inserted into the new IASCII file during conversion.

If you are using Word as your editing environment, make sure that you add a '#' character in front of both the <pn "xxx" ?> tag and the <ps "xxx" ?> tags. Otherwise, they will be invalid and will prevent S-Tagger from converting the STF files back to IASCII.

If you are using TagEditor as your editing environment, refer to the procedure outlined in "Paragraph Styles" on page 8-10. This will ensure that the tags you insert are treated as valid tags during verification.

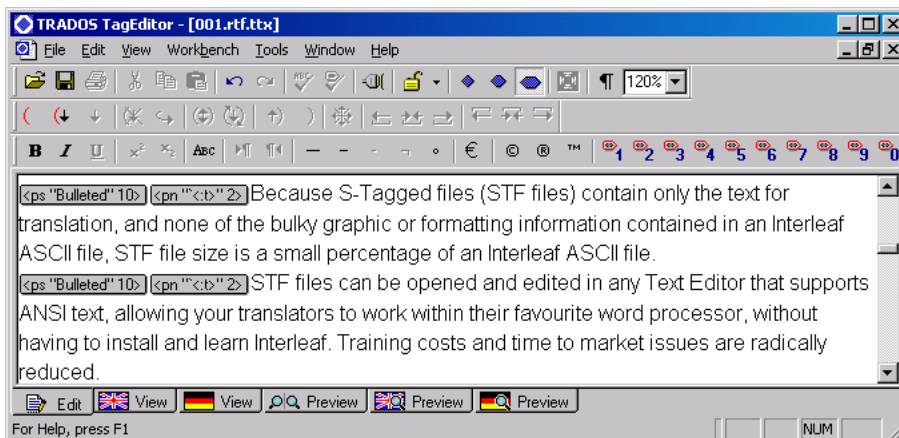
Example

The illustration below shows how a bullet symbol prefix, associated with a paragraph style, appears in an Interleaf file:

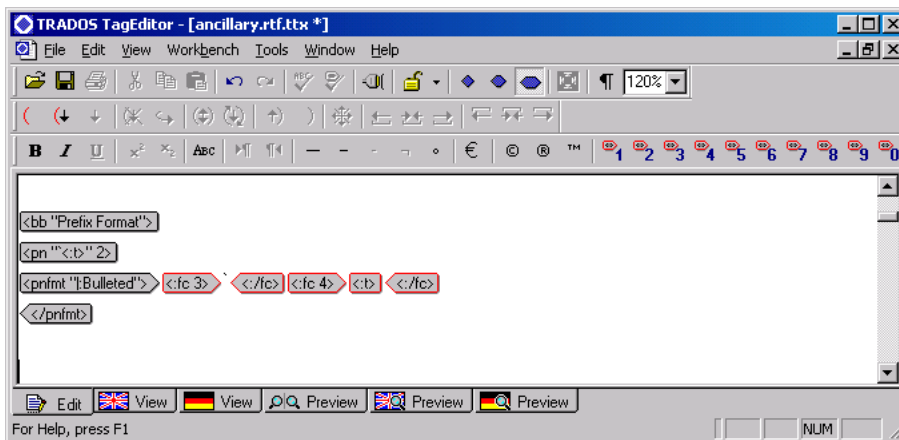
Component with Prefix <pn "xxx" ?> tags

- Because S-Tagged files (STF files) contain only the text for translation, and none of the bulky graphic or formatting information contained in an Interleaf ASCII file, STF file size is a small percentage of an Interleaf ASCII file.
- STF files can be opened and edited in any Text Editor that supports ANSI text, allowing your translators to work within their favourite word processor, without having to install and learn Interleaf. Training costs and time to market issues are radically reduced.

In the example below, the `<pn "xxx" ?>` tag represents the prefix bullet symbol. The apostrophe before the `<:t>` tag represents the bullet in *Symbol* font. The `<:t>` tag represents a tab character.



In the ancillary file under the section `Prefix Formats` you will see an example of the prefix: `<pn "`<:t>" 1>`.



Following the example will be the name of the prefix in a `<pnfmt "xxx">` tag. The prefix name will usually be the same as the name of the paragraph style.

You would translate the text following the `<pnfmt "xxx">` tag, if it was meant to be translated. The text in this example does not need to be translated, so you would not do anything to it.

Read-only Components

In Interleaf, a read-only property may be assigned to a component or inline component. This prevents the user from editing the content or properties of the component.

You have two choices as to how read-only components are treated in the STF file:

- ❑ If you wish to make all components with a read-only property translatable, in S-Tagger, select the **Translate** option under **Read only components** in the **Preferences** list on the **Settings** tab.

If you select this option, the components will be treated in the same manner as the rest of the document. During translation, you will have access to all the text in the component. When the STF file is converted back to IASCII, the translated component will still have the read-only property applied to it.

- ❑ If you wish to prevent translation of all components with a read-only property, select the **Don't translate** option on the **Settings** tab.

If you select this option, the read-only component is represented by tags, which cannot be translated.

How Read-only Components Appear in an STF File

If you select the **Don't translate** option, the read-only component is represented by the tag `<r "xxx" ?>` (if external) or `<:r "xxx" ?>` (if internal), where 'xxx' is the text of the component.

`<r "xxx" ?>` tags can contain other tags, such as `<:af>` tags to indicate the beginning of a frame containing text. Text in any such frame, which is found in a read-only component, will not be available for translation either in the STF file or the ancillary file. It will be visible as a visual aid only.

Shared Components

In Interleaf, a shared component is a special kind of building block. When you create a shared component, you place some text and/or graphics into a component. Each time you insert that component into the text, the same text and/or graphics appear. Shared components are often used for product names or company names.

How Shared Components Appear in STF

Shared components are referred to as variables in the Interleaf STF files, to correspond with FrameMaker STF files. They are presented in the STF file within the tag `<v "xxx" ?>` (if external) or `<:v "xxx" ?>` (if internal). The 'xxx' here refers to the actual text of the shared component as it appears in the Interleaf file, and not the component name. You cannot change this text within the

STF file. It is available in the ancillary file, where it can be translated, if necessary. A shared component can be both internal and external in the same file.

Example

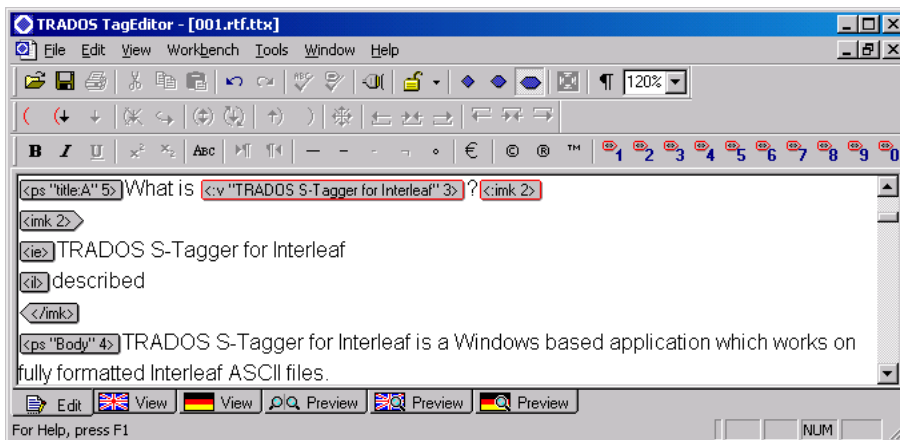
The product name 'TRADOS S-Tagger for Interleaf' is used in a documentation set, but it is known in advance that this name will change before shipping the documentation. In Interleaf, the product name is inserted by means of a shared component, so that when it is time to change it to the new name, the name will only have to change once in each file, and be automatically updated.

The following is presented in an Interleaf file:

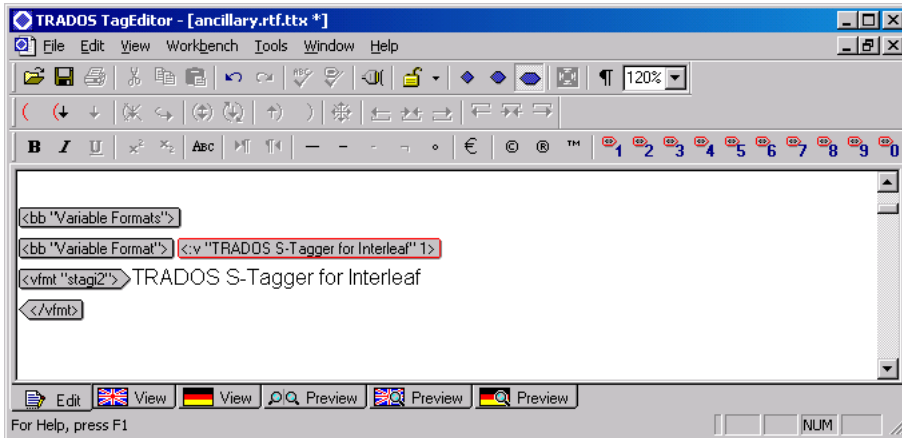
What is **TRADOS S-Tagger for Interleaf?**

TRADOS S-Tagger for Interleaf is a Windows based application which works on fully formatted Interleaf ASCII files.

In the STF file, this would be presented as:



To translate the text contained within the variable tag, open the ancillary file and find the correct file's `Variable Formats` section.



You can now translate the product name within the ancillary file. S-Tagger automatically updates the variable as used throughout the STF file when converting back to IASCI.

Shared Frames

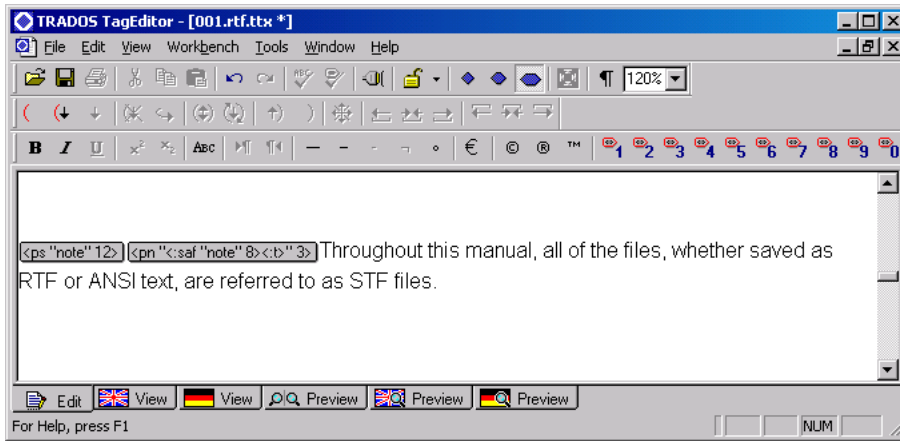
Any frame containing shared content and text objects will be represented in the STF file with the tag `<:saf "xxx" ?>` (if internal) or `<saf "xxx" ?>` (if external). The text contained in the shared frame is presented for translation in the ancillary file.

Example

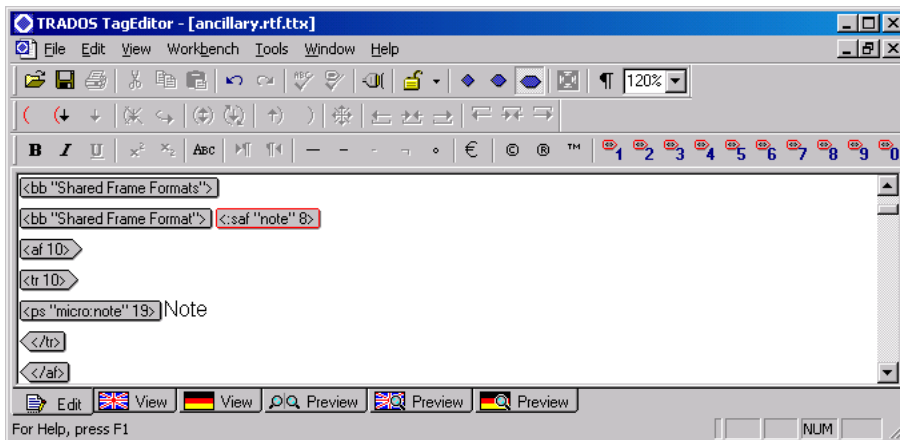
In an Interleaf file, the text 'Note' is contained in a shared frame within a prefix.

Note Throughout this manual, all of the files, whether saved as RTF or ANSI text, are referred to as STF files.

In the STF file, you will see a paragraph similar to the one below:



To translate the text contained within the tag, open the ancillary file and find the correct file's Shared Frame Formats section.



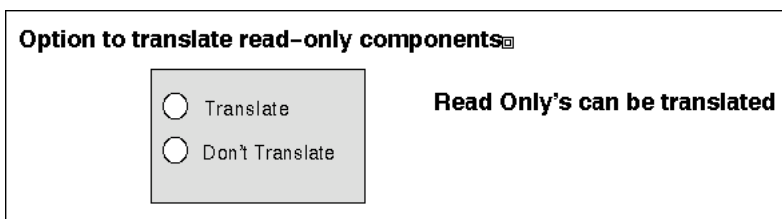
You can now translate the shared frame text within the ancillary file. S-Tagger automatically updates the frame text when converting back to IASCI.

Shared Named Objects

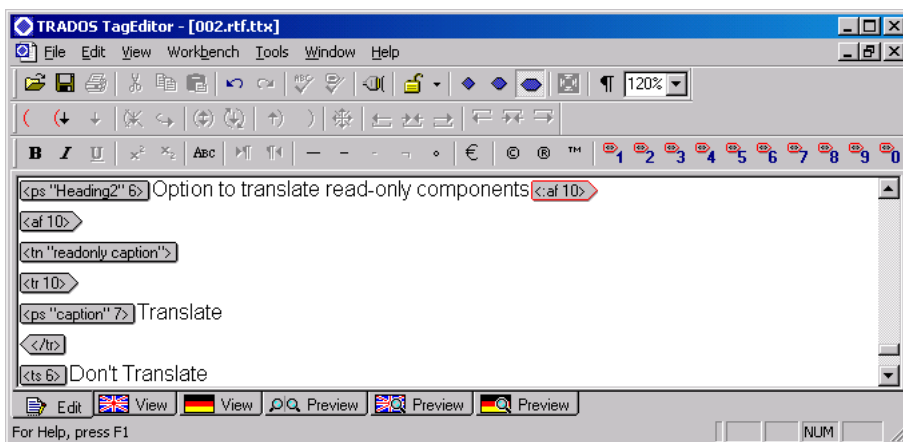
Any named object containing shared content and visible text objects, will be represented in the STF file with the tag `<tn "name" ?>`. The text contained in the shared frame will be presented for translation in the ancillary file.

Example

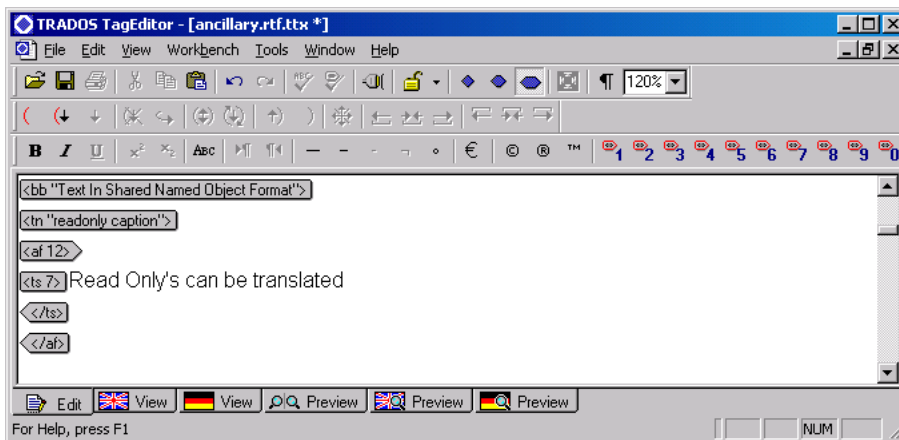
In the Interleaf graphic below, the text “Read-only’s can be translated” is a shared named object:



This text is presented as the tag `<tn "readonly caption">` in the STF file:



To translate the shared object text, open the ancillary file and find the correct file's Text In Shared Named Object Formats section.



You can now translate the shared object text within the ancillary file. S-Tagger automatically updates the object text when converting back to IASCI.

Special and Reserved Characters

Some special characters, such as hard or non-breaking spaces, em and en spaces and hard returns, are represented in STF with internal tags. This is to ensure that the translator does not remove them inadvertently and can easily insert them where necessary. For example, in French translations, a hard space is often inserted before a colon.

Some Interleaf characters are reserved for system use and do not have an equivalent value that can be interpreted and displayed in STF. These characters are represented by a reserved character tag, `<:r1>` to `<:r16384>`. Reserved characters are also used when a Japanese source text contains European extended characters.

Tables

In Interleaf, you create a table by creating a new table master or by creating an instance of an existing table master. Tables are used for any other text and graphics that you organize in columns and rows.

How Tables Appear in an STF File

A table begins with the opening tag `<tb ?>`, where '?' is the unique number for that table. It then continues with a sequence which includes at least one of the tag pairs in the table below until it

reaches its corresponding `</tb>` closing tag. Note that the tags in the STF files appear in the order in which they are listed.

Tag	Represents
<code><hrow ?></code>	Hidden table row This text is not translatable.
<code><rrow "xxx" ?></code>	Table row with a read-only attribute applied The 'xxx' represents the text of the row.
<code><row ?> </row></code>	Table row
<code><ct ?> </ct></code>	Table cell Each <code><ct ?></code> tag is followed by a paragraph style tag and then text, just like a regular body paragraph. You only translate the text that follows a <code><ct ?></code> tag.
<code><ctart ?></ctart></code>	Table cell with graphics editor applied

Tables which span pages will be presented in the STF file as one continuous table, with no page breaks.

Example

The following table is presented in an Interleaf file:

Product	Developed by	Description
SuperKid Seat	Child Car Seats Inc.	This is our base model for infants.

In the STF file, this table appears as follows (the different elements are explained in the second column):

S-Tag and Text	Represents
<code><tb 13></code>	Indicates the start of the table
<code><row 1></code>	Start of the first (and only) heading row
<code><ct 1></code>	Start of the first cell in the heading row
<code><ps "cellheading" 1>Product</code>	Text in the first cell in the heading row
<code></ct></code>	End of the first heading row cell

S-Tag and Text	Represents
<code><ct 1></code>	Start of the next cell
<code><ps "cellheading" 1>Developed by</code>	The text in the cell
<code></ct></code>	End of the cell
<code><ct 1></code>	Start of the next cell
<code><ps "cellheading" 1>Description</code>	The text in the cell
<code></ct></code>	End of the cell
<code></row></code>	End of the first (and only) heading row
<code><row 2></code>	Start of the first body row
<code><ct 1></code>	Start of the first body row cell
<code><ps "cellbody" 1>SuperKid Seat</code>	The text in the cell
<code></ct></code>	End of the cell
<code><ct 1></code>	Start of the next cell
<code><ps "cellbody" 1> Child Car Seats Inc.</code>	The text in the cell
<code></ct></code>	End of the cell
<code><ct 1></code>	Start of the next cell
<code><ps "cellbody" 1>This is our base model for infants.</code>	The text in the cell
<code></ct></code>	End of the cell
<code></row></code>	End of the row
<code></tb></code>	End of the table

**TIP**

Use a hard copy of the original Interleaf file for reference, so that you can easily see which cells you are translating.

Graphics in Tables

A `<ct ?>` tag represents a table cell. A `<ctart ?>` tag represents a table cell with a graphics editor applied. `<ctart ?>` cells are closed by `</ctart>` tags. Text within the table cells will be formatted just like text anywhere else, with paragraph and character styles applied. There can be markers within the cells, but graphics and text objects can only appear in `<ctart ?>` type cells.

Example

In the table below, the cell containing the text “Feature” has a graphics editor applied.

Feature	Page
Component <ps "xxx" ?>	

This appears in the STF file as follows:

S-Tag and Text	Represents
<tb 1>	Indicates the start of the table
<row 1>	Start of the first row
<ctart 1>	Start of cell containing a graphic
<af 8>	Anchored frame
<tr 8>	Start of text
<ps "cellhead" 4>Feature	Text in the cell
</tr>	End of text
</af>	End of anchored frame
</ctart>	End of cell

Unknown Inline Components

Interleaf allows the user to insert a subcomponent or inline component for other purposes than a change in character formatting. When one of these subcomponents is encountered and S-Tagger cannot determine what the formatting instruction is, the tag is converted to an unknown inline component tag, <:sc "name" ?>.

The text following the tag until the closing tag <:/sc> will have that format applied to it.

**WARNING**

Never delete the unknown inline component tags or replace them unless you are certain of the effect this will have on the Interleaf file.

S-TAGS IN THE STF ANCILLARY FILE

This section describes the use of S-Tags in the ancillary file. For more information on the ancillary file, see “The Ancillary File” on page 2-3.

When translating text in the ancillary file, follow these guidelines:

- ❑ Remember to translate only the necessary text, and not the tags in the ancillary file.
- ❑ The ancillary file must always be verified, along with the rest of the STF files, to ensure that the tag structure and usage is correct.
- ❑ We recommend that you translate the text of the ancillary file before translating the rest of the text in the documentation set. This way, all translators working on a project will be able to see the translations in the ancillary file and the translations will be contained in the translation memory, if one is being used.

The ancillary file creates a separate section for each of the files that you convert to STF. The beginning of each file section is identified with a file name tag. For example:

```
<file "C:\Program Files\Translation Project\IASCII\001.ildoc">
```

Within each of these file sections, the ancillary file contains the following subsections:

- ❑ Page Header/Footer Formats
- ❑ Shared Frame Formats
- ❑ Text in Shared Named Object Formats
- ❑ Variable Formats
- ❑ Prefix Formats
- ❑ Autonumber Stream Formats
- ❑ Page Number Stream Formats.

Each catalog file is listed at the end of the ancillary file. If no catalogs are included in the conversion, there will be no catalogs in the ancillary file.

As S-Tagger converts the IASCII files to STF, the appropriate ancillary text is inserted into each section. If there is no ancillary text in any of the files that were converted corresponding to a particular section, the section will remain empty.

Example

A document contains a left page header with the text “Road Safety Information”. This text does not appear in the corresponding STF file generated by SDL Trados. To translate the text and all other instances of it from the shared content component in the file, open the ancillary file.

First, search for the file name tag, then scroll down until you come to the tag
<bb "Variable Format">.

Next, scroll down until you see the following tags:

```
<v "Road Safety Information" 4>  
<vfmt "headerL">Road Safety Information  
</vfmt>
```

Following the name will be the variable format which shows you the name of the format. In this case, <vfmt "headerL">, followed by the actual text of the variable, in this case, “Road Safety Information”.

You would then translate the relevant words and save the file.

For prefixes, translate the text after any <pnfmt "xxx"> tags, and for page number streams, translate the text after the <bb "prefix1"> or <bb "prefix2"> tags, if any are present in the ancillary file.

Page Header/Footer Formats

The page header/footer section contains the formats for each header and footer used within the particular file.

Shared Frame Formats

The shared frame format section contains any text and formatting associated with shared frames used in a particular file. For more information, see “Shared Frames” on page B-22.

Text in Shared Named Object Formats

The text in shared named object formats section contains any text that is associated with a named object used in a particular file. For more information, see “Shared Named Objects” on page B-24.

Variable Formats

The variable formats section contains any text that is inserted as a shared component throughout a file. For more information, see “Shared Components” on page B-20.

Prefix Formats

The prefix formats section contains any text that is inserted as a prefix. This is usually associated with autonumber streams or cross-references. For more information, see “Paragraph Styles (Components)” on page B-15 and “Cross-references” on page B-7.

Autonumber Stream Formats

The autonumber stream format section contains any text that is associated with an autonumber stream.

In Interleaf, text can be associated (as a prefix or a suffix) with an autonumber stream. When the text is an autonumber, it is represented by the tag `< : an "xxx" ? >` where ‘xxx’ is the text of the prefix, the autonumber and the text of the suffix. The actual text of the prefix and/or suffix is translated in the ancillary file.

When a reference is created to an autonumber, the reference is treated as a cross-reference. For more information, see “Cross-references” on page B-7.

Page Number Stream Formats

The page number stream format section contains any text that is associated with a page numbering stream. In Interleaf, page numbering streams are similar to autonumbering streams.

When the text is a page number, it is represented by the tag `<page "xxxx" >?</page>` where ‘xxx’ is the text that needs to be translated in the ancillary file. In the ancillary file, you will find something like the following:

```
<bb \"Page Number Stream Format\">
<pgfmt "xxx">
<bb "Prefix1">prefix text to be translated
<bb "Prefix2">more prefix text to be translated
</pgfmt>
```

LIST OF S-TAGS

The table below presents a comprehensive list of tags. The list is arranged alphabetically and is divided into internal tags and external tags. If a tag needs a closing tag, this is indicated in the second column.

Consult the specific rules on each one. Note that 'xxx' represents any text. '?' represents a number.

Tag	Closing Tag	Represents
<code><:af ?></code>		Anchored frame marker
<code><:an "xxx" ?></code>		Autonumber
<code><:axr ?></code>		Attribute reference
<code><:b></code>	<code><:/b></code>	Bold
<code><:bb "xxx"></code>		Building block
<code><:bi></code>	<code><:/bi></code>	Bold italic
<code><:c></code>	<code><:/c></code>	Use different character set for font encoding. (Times font)
<code><:c1></code>	<code><:/c1></code>	Use different character set for font encoding. (Courier font)
<code><:c2></code>	<code><:/c2></code>	Use different character set for font encoding. (Helvetica font)
<code><:cnmk ?></code>		Hidden conditional text marker
<code><:cns "xxx" ?></code>	<code><:/cns></code>	Conditional text style
<code><:crmkn ?></code>		Cross-reference marker
<code><:cs "xxx" ?></code>	<code><:/cs></code>	Character style
<code><:ems></code>		Em space
<code><:ens></code>		En space
<code><:fc ?></code>	<code><:/fc></code>	Font change
<code><:flq></code>		French left quote (<< symbol)
<code><:fn "xxx" ?></code>		Footnote reference marker
<code><:frq></code>		French right quote (>> symbol)

Tag	Closing Tag	Represents
<:fs>		Figure space
<:glq>		German left quote („ symbol)
<:grq>		German right quote (“ symbol)
<:gt>		Greater than symbol (>)
<:hls>		Hairline space
<:hr>		Hard (non-breaking) return
<:i>	<:/i>	Italic
<:iaf ?>		Internal anchored frame
<:imk ?>		Index marker placeholder
<:ldq>		Left double quote
<:lq>		Left single quote
<:lt>		Less than symbol (<)
<:pn "xxx" ?>		Internal prefix
<:r "text" ?>		Internal read only
<:r1> - <r:16384>		Interleaf reserved characters
<:rdq>		Right double quote
<:rq>		Right single quote
<:s>	<:/s>	Small caps
<:saf "xxx">		Shared anchored frame
<:sc "xxx" ?>	<:/sc>	In-line component
<:sh>		Soft hyphen
<:so>		Internal sort order
<:t>		Tab
<:t.>		Tab (leader character dot)
<:t_>		Tab (leader character underscore)
<:t->		Tab (leader character dash)

Tag	Closing Tag	Represents
<code><:t-d></code>		Tab (leader character short dash)
<code><:ts></code>		Thin space
<code><:v "xxx" ?></code>		Internal variable (shared component)
<code><:xr "xxx" ?></code>		Cross-reference
<code><ac></code>	<code></ac></code>	Start of ancillary content
<code><af ?></code>	<code></af></code>	Anchored frame
<code><ancillarymode></code>		Ancillary mode setting in STF files
<code><anfmt "xxx"></code>	<code></anfmt></code>	Autonumbering stream format, used in the ancillary file
<code><bb "xxx"></code>		Building block
<code><cnmk ?></code>		Hidden conditional text marker
<code><ct ?></code>	<code></ct></code>	Table cell
<code><ctart ?></code>	<code></ctart></code>	Table cell with graphics editor
<code><file "xxx"></code>	<code></file "xxx"></code>	File name, used in the ancillary file
<code><fn "xxx"></code>	<code></fn></code>	Footnote
<code><hrow ?></code>		Hidden table row
<code><ie></code>		Index entry
<code><il></code>		Index entry sub-entry
<code><imk ?></code>	<code></imk></code>	Index marker
<code><page "xxx"></code>	<code></page></code>	Page number
<code><pgfmt "xxx"></code>	<code></pgfmt></code>	Page numbering stream format, used in the ancillary file
<code><pn "xxx" ?></code>		Text in shared prefix
<code><pnfmt "xxx"></code>	<code></pnfmt></code>	Prefix format, used in the ancillary file
<code><ps "xxx" ?></code>		Paragraph style
<code><r "xxx" ?></code>		Read only paragraph
<code><row ?></code>	<code></row></code>	Table row

Tag	Closing Tag	Represents
<code><rrow "xxx" ?></code>		Read only row
<code><sa></code>		'See also' entry in index marker
<code><saf "xxx"></code>		Shared anchored frame
<code><sourcelanguage "xxx"></code>		Source document language setting
<code><sourcepath "xxx"></code>		Path to source file
<code><sourcequotes "xxx"></code>		Source document smart quotes setting
<code><sourcereadonly "xxx"></code>		Source document read only setting
<code><ss></code>		Sort string in index marker
<code><stf "xxx"></code>	<code></stf></code>	Declaration of STF file type and version
<code><tb ?></code>	<code></tb></code>	Table
<code><t1 ?></code>	<code></t1></code>	Chart label
<code><tn "xxx" ?></code>		Text in shared named object
<code><tr ?></code>	<code></tr></code>	Text box
<code><ts ?></code>	<code></ts></code>	Text string
<code><v "xxx" ?></code>		Variable (shared component)
<code><vfmt "xxx"></code>	<code></vfmt></code>	Variable format, used in the ancillary file
<code><xrv "xxx" ?></code>	<code></xrv></code>	Cross-reference variable



FOR MORE INFORMATION

For more information, see “Manipulating Internal Tags” on page 8-7 and “Manipulating External Tags” on page 8-9.

GLOSSARY

A

Alignment

Process of building a translation memory from previously translated material. WinAlign is the alignment tool.

Analysis

Feature in Translator's Workbench that allows you to assess the number of translation matches in a document for translation by comparing the document to an existing translation memory.

Anchored Frame

Used to contain a graphic, text strings and/or text boxes within a column of text. Anchored frames move with the text flow in which they are placed; they are anchored to the text, not their position on the page.

Ancillary File

TTX, RTF or text-only file that is generated during the conversion of MIF/IASCI documents to STF. The ancillary file contains shared text that cannot be accessed in the STF file but that requires translation.

ANSI Text

ANSI (American National Standards Institute) text is used by Windows.

B

Batch Processing

Processing multiple files at one time is referred to as batch processing.

Book File

Used by FrameMaker and Interleaf to assemble any number of documents into a unified document with continuous page, figure, and list numbering. A book file allows you to graphically see and manage the organisation of documents within it. This organisation can include generating files like table of contents and an index, and assigning page numbering.

C

Callout

Caption that appears beside a graphic, explaining items in the graphic.

Catalog File

Interleaf allows you to set up the styles for a document or set of documents in one file which can then be imported into all following documents. A catalog is a type of style sheet.

CAT File

File created by S-Tagger for Interleaf from catalog files.

Character Set

FrameMaker has two ways of internally encoding characters; Normal and Symbol. Normal is used to encode most Western European languages. Symbol is generally used for Asian characters, as well as some Eastern and Central European languages.

Character Style

Font attribute (typeface, size, bold, italic) that can be given a character style and applied to individual or sets of characters.

In Interleaf, character style formatting can be applied through the use of inline components.

Clean Up

Feature in Translator's Workbench that removes source segments from translated documents. The source segments have been stored in the translated document during the translation process. The clean up feature also allows you to update the translation memory in accordance with the latest changes in the target files.

Concordance

Feature that allows you to search the translation memory for text fragments during interactive translation. The **Concordance** tab in the **Translation Memory Options** dialog box in Translator's Workbench defines the parameters for concordance searching.

Context TM

The Context TM process compares updated source files to old bilingual documents rather than a translation memory. Segment matches, known as Context TM units (XUs,) are checked for context and extracted from the old bilingual documents and transferred

to the updated source files. This results in a new set of bilingual documents that have the extension *.ttx.

CMP File

File generated by the S-Tagger verification feature. It details all errors, alerts and/or warnings about tags.

D

DBCS

Double-byte Character Set. DBCS languages such as Japanese, Korean and Chinese require special DBCS fonts.

Demo Mode

Applications, including the S-Taggers and the Font Mappers can be used in demonstration mode when no dongle (or copy protection) is attached. File size limitations and feature restrictions may apply.

DTD

Document Type Definition. An ASCII file that defines the structure, elements and conventions to which an SGML, XML or HTML document must conform. The DTD file is separate from the document it defines. An example of a widely used DTD is the HTML 4.0 DTD; all HTML 4.0 documents are based on this DTD.

Tag Settings File

Document that contains information necessary for processing documents that conform to a particular DTD. For example, a Tag settings file to process and format HTML, SGML, and XML documents for translation purposes. DTD settings files (also known as initialisation files) have the extension `*.ini`.

In SDL Trados, the Tag Settings Wizard is used to create new Tag settings files or to edit existing ones. The wizard is available in TagEditor, WinAlign and the Localization Manager module of Translator Studio.

E

Extended Characters

Accented letters and symbols which do not appear in the standard ASCII character set, for example, 'à' or 'é'.

External Tag

Tag which represents structural formatting in an STF file. Paragraph styles, anchored frames and tables are represented by external tags.

F

Font Mapper for FrameMaker

Font Mapper for FrameMaker facilitates changing the fonts in FrameMaker MIF documents translated into or from Asian languages. Font Mapper for FrameMaker maps, or replaces, the fonts in the original document with fonts that you specify.

Font Mapper for Interleaf

Font Mapper for Interleaf facilitates changing the fonts in Interleaf ASCII documents translated into or from Japanese. It maps, or replaces, the fonts in the original document with fonts that you specify.

Footers

Running footers. Where the same text needs to be inserted at the bottom of each page of a particular type, a footer is inserted.

FrameMaker

FrameMaker is a powerful desktop publishing and book-building package. It runs on several platforms; UNIX, Macintosh and Windows.

Frequently Used Formatting

Certain types of formatting, such as applying bold or italics to individual characters, is referred to as frequently used formatting. Special S-Tags can be inserted to represent this type of formatting.

H**Hard Return**

Manual line break or a non-breaking carriage return. When you want a new paragraph to begin, but do not want to incur paragraph attributes or the end of previous paragraph attributes, you insert a hard return.

Hard returns are known as forced returns in FrameMaker.

I**IASCII**

Interleaf ASCII. IASCII is a text file format which allows you to exchange information between Interleaf and other applications. All formatting, file and page layout is retained in the IASCII file.

InDesign

A desktop publishing package

Index Marker

Index entries are stored in index markers. The cursor is placed at the point in the text where you want an index entry to be referenced to and the text is typed into the index marker. In STF files, index markers are moved to the ends of the paragraphs they were originally contained in.

Interleaf

A desktop publishing package, which has some similar features to FrameMaker. The STF produced from Interleaf files is similar to the STF produced from FrameMaker files. Interleaf runs on several platforms; UNIX, DOS and Windows. S-Tagger for Interleaf supports IASCII files from all Interleaf platforms and from Interleaf versions 5.2 or later.

Internal Tag

Tag which is used in STF files for formatting commands that do not affect the overall structure of the file, such as character styles, variables special characters, and non-breaking spaces.

M**MIF**

Maker Interchange Format, a text format used by FrameMaker which allows you to exchange information between FrameMaker and other applications. All formatting and page layout information is retained in the MIF document.

MultiTerm

The terminology management system. The dynamic interface between MultiTerm and Translator's Workbench means that terms stored in MultiTerm are automatically suggested to translators during translation.

O**ORG File**

File containing structural information for the conversion of STF files to original format. The S-Taggers produce an ORG file

for each MIF/IASCII file that is converted to STF. The ORG file is used to convert the translated STF file back to its original MIF/IASCII format.

P

Page Break

You can set up a component/paragraph style so that it always starts at the top of a page or always stays with the following paragraph. Most documents will contain some components/paragraph styles whose pagination attributes are overridden, for example, when a forced page break has to be inserted.

PageMaker

PageMaker is a desktop publishing package from Adobe. It is essentially a page layout program.

Paragraph Style

Used with FrameMaker, paragraph style properties, such as indents, line spacing, tab stops, font attributes (typeface, size or colour), pagination, hyphenation and word spacing, are all assigned within the Paragraph Designer.

In Interleaf, paragraph styles are known as components and the style properties are assigned within the component's master property sheet.

Pre-translation

Process of translating multiple files using the **Translate** command on the **Tools** menu in Translator's Workbench. Translator's Workbench automatically inserts matches from the translation memory into the document for translation.

Public Entity Set

A standardised set of character entities. Typically, a character entity represents a special character not available in the ASCII character set. For example, the public entity set ISO Added Latin 1 contains representations for all umlaut and accented characters, as used in Western European alphabets.

Q**QuarkXPress**

QuarkXPress is an electronic publishing software package. It allows you to create, design, and deliver publications in both print and electronic media.

QuickSilver

QuickSilver is Interleaf's XML content management system. It uses Interleaf 7 as its editor.

R**Read-only**

Interleaf allows you to apply a read-only property to any component or subcomponent. Components with read-only properties assigned to them cannot be edited in any way in the Interleaf file. S-Tagger for Interleaf allows you to choose whether or not these components are to be treated as translatable text.

RTF

Rich Text Format. This is an interchange file format invented by Microsoft.

See also "Workbench RTF".

S**SGML**

Standard Generalized Markup Language, a language used for designing tagged text formats.

Shift-JIS

A standard for Japanese character encoding, used mainly under Windows and Macintosh.

Smart Quotes

Curly left and right quotation marks instead of straight quotation marks. Smart quotes can have different formats, depending on the language they are used in; for example, German smart quotes are different to English smart quotes.

Special Characters

Symbols like '®', '™', '©', as well as accented characters such as á and ì. Special characters and symbols must be inserted into the STF file using the Alt keyboard sequence. They may *not* be inserted using the **Insert Symbol** command in Microsoft Word for Windows.

S-Tagger for FrameMaker

The conversion solution for FrameMaker. S-Tagger for FrameMaker converts MIF files to STF. All FrameMaker page layouts and character formatting are preserved during translation.

S-Tagger for Interleaf

The conversion solution for Interleaf. S-Tagger for Interleaf converts Interleaf ASCII files to STF. All Interleaf page layouts and character formatting are preserved during translation.

S-Tags

Brief coded statements in STF files that represent formatting in FrameMaker and Interleaf documents. S-Tags are divided into external tags and internal tags.

STF File

File produced when you convert MIF/IASCII files to STF using the Convert to STF feature. Formatting from the FrameMaker/Interleaf documents is represented by brief coded statements (S-Tags). STF files are translated using Translator's Workbench in the TagEditor or the Microsoft Word editing environment.

STF File Format

When using the S-Taggers to convert FrameMaker (MIF) and Interleaf (IASCII) files to STF, you choose the file format of the STF files. They can be saved in TTX (TRADOSTag), RTF (Rich Text Format) or as ANSI text.

Story Collector for InDesign

Story Collector for InDesign facilitates the export of tagged text from InDesign and the subsequent re-import of text after translation.

Story Collector for Pagemaker

Story Collector for PageMaker facilitates the export of tagged text from PageMaker and the subsequent re-import of text after translation.

Story Collector for QuarkXPress

Story Collector for QuarkXPress facilitates the export of tagged text from QuarkXPress and the subsequent re-import of text after translation.

T

TagEditor

The translation editor for tagged text formats including HTML, SGML, XML, STF (Workbench RTF and TRADOSTag) and DTP file formats including QuarkXPress, PageMaker and Ventura. TagEditor also includes the DTD Settings Wizard, tag verification and spelling checker plug-ins.

Termbase

Database used to store terminology and related information.

TRADOSTag

An XML-based file format for representing both DTP tagged text output and bilingual data. TRADOSTag (TTX) provides a standard method for processing XML, HTML, SGML and DTP file formats and replaces BIF as the file format for bilingual documents in SDL Trados.

TTX is one of the STF file format options that you can select in the S-Taggers.

Translation Memory

Database of segment or sentence pairs. Each source language segment is paired with its corresponding target language segment.

Translation Memory Server

Server component in the Workbench client/server system. The Translation Memory Server handles all communication between the server-based translation memory and clients of the system, including Localization Manager.

Translator's Workbench

Translation memory database and management system with document

analysis, pre-translation and clean up features. Translator's Workbench can be used in conjunction with a variety of editing environments including Microsoft Word and TagEditor. Translator's Workbench is also integrated with the MultiTerm terminology management system.

TTX

See "TRADOStag".

V

Validation

In TagEditor, validation refers to the process of validating an XML document after translation using the XML Validator plug-in.

Ventura

Ventura is a desktop publishing package from Corel.

Verification

Refers to the comparison process where the tags in the target STF file are compared and verified against the tags in the source STF file. During and after translation, translators should verify that they have placed all the tags in the translated files in the correct sequence and position.

Verification can be carried out using TagEditor's S-Tag Verifier plug-in and/or the S-Taggers' verification feature.

Verification Plug-in

The verification plug-ins provide advanced tag verification functionality for whole documents that have been translated in TagEditor. Verification plug-ins include the Generic Tag Verifier, the XML Validator and the S-Tag Verifier, QA Checker,

W

Warning

Message which is generated during tag verification. Certain warnings may be suppressed during the verification process by customising the verifier report. Warnings identify changes in tags which do not affect the structure of the file, only the formatting of the text within the file.

Warnings do not prevent backward conversion of the target file but should be checked in order to avoid undesirable effects in the finished document.

WinAlign

The visual alignment tool that allows you to create translation memory import files from previously translated source and target texts. WinAlign also includes the DTD Settings Wizard.

Workbench RTF

Workbench RTF documents (also referred to as tagged RTF documents) are files that comply with the Translator's Workbench standard for tagged RTF. These files contain tags that have been marked up with the *tw4winExternal* and *tw4winInternal* tag styles.

RTF is one of the STF file format options that you can select in the S-Taggers.

X**XML Validator**

The plug-in that allows you to validate XML documents that have been translated in TagEditor.

INDEX

A

- About tab, S-Tagger for FrameMaker 3-13
- About tab, S-Tagger for Interleaf 7-15
- Alerts, FrameMaker
 - correcting 5-10
 - sorted files 5-16
 - suppressing, displaying 5-4
- Alerts, Interleaf
 - correcting 9-10
- Analysis, Translator's Workbench 2-15
- Anchored frames, FrameMaker
 - graphics 3-7
 - positioning 3-7
 - preparing for conversion to STF 3-7
- Ancillary file
 - about 2-3
 - definition of 1-4
- Ancillary file S-Tags B-29
 - variable formats A-27
- Ancillary file S-Tags, FrameMaker
 - cross-reference formats A-27
 - element prefix and suffix formats A-27
 - footnote prefix and suffix formats A-28
 - master pages A-28
 - paragraph numbering formats A-27
- Ancillary file S-Tags, Interleaf
 - autonumber stream formats B-31
 - page header and footer formats B-30
 - page number stream formats B-31
 - prefix formats B-31
 - shared frame formats B-30
 - variable formats B-31
- Ancillary file S-Tags, text in shared named object formats B-30
- Ancillary file, FrameMaker 2-3
 - creating 3-4
 - translating 4-3
- Ancillary file, Interleaf
 - about 2-4
 - creating 7-4
 - translating 8-3
- Ancillary text, *see* Ancillary content
- ANSI text files
 - using with the S-Taggers 2-10
 - using with Translator's Workbench 2-10

- Asian and Eastern European Languages, FrameMaker 4-14
- Asian languages, FrameMaker
 - character sets and conversion to STF 3-11
 - cross platform files and conversion to STF 3-11
 - font mapping 4-14
 - preparing for conversion to STF 3-11
 - Rubi characters and conversion to STF 3-11
 - selecting file format for conversion to STF 3-15
 - smart quotes 3-18
 - source marker length and conversion to STF 3-11
 - testing MIF and STF files 3-12
 - types of language supported 3-11
- Asian languages, Interleaf
 - smart quotes 7-20
- Automatic hyphenation, FrameMaker
 - turning off, turning on 3-7
- Automatic hyphenation, Interleaf
 - turning off, turning on 7-9

B

- Backward conversion, *see* Converting to original format
- BAK file
 - creating 2-16
- BAK file, Translator's Workbench 2-15
- Between session option, S-Tagger for FrameMaker 3-23
- Between session option, S-Tagger for Interleaf 7-24

C

- CAT file
 - saving in IASCII format for translation 7-6
- Catalogs, Interleaf
 - preparing for conversion to STF 7-6
- Change bars, FrameMaker
 - preparing for conversion to STF 3-6
- Character set option, S-Tagger for FrameMaker 3-16
- Character set tags, FrameMaker
 - Asian languages 4-14
 - Eastern European languages 4-14
- Character set tags, Interleaf
 - Japanese 8-14

- Character styles, FrameMaker
 - S-Tags for 4-9
 - Checking the IASCII files 7-14
 - Clean up 2-17
 - definition of 2-12
 - Clean Up command, Translator's Workbench
 - using 5-19
 - using to update the translation memory 5-19
 - CMP file
 - about 2-5
 - and the verifier report 2-5
 - definition of 1-4
 - CMP file, FrameMaker
 - comparison errors 5-11
 - contents of 5-9
 - tag alerts 5-13
 - tag errors 5-12
 - tag warnings 5-16
 - using 5-10
 - CMP file, Interleaf
 - comparison errors 9-11
 - contents of 9-9
 - tag alerts 9-13
 - tag errors 9-12
 - tag warnings 9-16
 - using 9-10
 - Comparison errors, FrameMaker 5-11
 - Comparison errors, Interleaf 9-11
 - Compatibility, FrameMaker 1-6
 - Compatibility, Interleaf 1-6
 - Conditional text, FrameMaker
 - preparing for conversion to STF 3-5
 - Confirming STF Conversion 6-3, 10-3
 - Context TM
 - about 2-15
 - definition of 2-12
 - Context TM in TagEditor 2-13
 - Context TM unit tags 2-7
 - Context TM Wizard 2-15
 - Control expressions, Interleaf
 - preparing for conversion to STF 7-7
 - Conversion settings, S-Tagger for FrameMaker 3-13
 - checking 5-3
 - Conversion settings, S-Tagger for Interleaf
 - checking 9-3
 - Convert STF tab, FrameMaker 6-2
 - Convert STF tab, Interleaf 10-2
 - Converting
 - definition of 1-5
 - IASCII to STF 7-24
 - MIF to STF 3-24
 - ORG file 2-5
 - to original format 2-17
 - to STF, about 2-14
 - Converting STF to IASCII 10-2
 - confirming conversion 10-3
 - rules 10-2
 - save results 10-4
 - selecting STF files to convert 10-2
 - Converting STF to MIF 6-2
 - confirming conversion 6-3
 - rules 6-2
 - save results 6-4
 - selecting STF files to convert 6-2
 - Converting, FrameMaker
 - RTF file format 3-15
 - TRADOSTag file format 3-15
 - TXT file format 3-15
 - Converting, Interleaf
 - RTF file format 7-17
 - TRADOSTag file format 7-17
 - TXT file format 7-17
 - Correcting STF to IASCII conversion errors 10-4
 - Correcting STF to MIF Conversion Errors 6-4
 - Cross-references, FrameMaker
 - preparing for conversion to STF 3-6
 - S-Tags for 4-9
 - Customise Verifier Report dialog, S-Tagger for FrameMaker
 - default settings 5-6
 - opening 5-5
 - Customise Verifier Report dialog, S-Tagger for Interleaf
 - default settings 9-6
 - opening 9-5
 - Customising
 - verifier report settings 3-21, 7-23
-
- D**
 - Demo mode
 - about 1-7
 - file size limitations 1-7
 - Discretionary hyphenation, FrameMaker
 - preparing for conversion to STF 3-8
 - Doc-to-Help
 - errors in MIF files 3-12
 - Document language options, S-Tagger for FrameMaker 3-20
 - Documentation
 - useful reading material 1-2
 - During Session option, S-Tagger for FrameMaker 3-23
 - During Session option, S-Tagger for Interleaf 7-24

E

Editing

- STF files, about 2-16

Editing environments

- choosing an editor 2-13
- effects on conversion options 2-13
- options 2-2

Errors, FrameMaker 5-4

Extended characters and symbols, FrameMaker

- in TagEditor 4-6
- in Word 4-6
- S-Tags for 4-5
- translating 4-5

Extended characters and symbols, Interleaf

- in TagEditor 8-6
- in Word 8-5
- S-Tags for 8-5
- translating 8-5

External tags 2-3

- definition of 1-4
- presentation of 2-6
- tag identifiers 2-8

External tags, FrameMaker

- index entries 4-11
- index markers 4-11
- index sort levels 4-12
- index sort strings 4-12
- paragraph numbering formats 4-11
- paragraph styles 4-10
- prefixes 4-11
- using 4-9

External tags, Interleaf

- index entries 8-11
- index markers 8-11
- index sort strings 8-12
- paragraph numbering formats 8-10
- paragraph styles 8-10
- prefixes 8-10
- using 8-9

F

File errors, FrameMaker 5-8

File errors, Interleaf 9-8

File formats, FrameMaker

- creating monolingual RTF files in TagEditor 4-3, 8-3
- translating 4-3
- using Translator's Workbench Clean Up command 4-3

File formats, Interleaf

- translating 8-3
- using Translator's Workbench Clean Up command 8-3

File size 1-7

Files

- BAK file 2-15
- sample 1-2
- separation of 2-9, 2-10

Folder structure

- creating 2-9

Folder structure, FrameMaker 3-4

Folder structure, Interleaf 7-4

Font changes, FrameMaker

- S-Tags 4-9

Font Mapper for FrameMaker

- removing italics for Asian languages 3-12

Font Mappers

- using with the S-Taggers 1-3

Font mapping, FrameMaker

- for Asian languages 4-14

Font mapping, Interleaf

- Japanese 8-15

Font substitution messages, FrameMaker 3-5

Font substitution messages, Interleaf 7-6

FontMappers

- file size in demo mode 1-7

Forced returns, *see* Hard returns

Formatting information

- internal tags 2-6

Formatting tags, *see* S-Tags

Formatting, FrameMaker

- S-Tags for 4-5
- STF files in TagEditor 4-5
- STF files in Word 4-5

Formatting, Interleaf

- S-Tags for 8-5
- STF files in TagEditor 8-5
- STF files in Word 8-5

Frame anchors, FrameMaker

- preparing for conversion to STF 3-7

FrameBuilder 1-6

FrameMaker

- graphics 3-4
- preparing files for conversion to STF 3-2
- project folder structure 3-4
- referenced files 3-4
- symbols and special characters 3-9
- versions supported 1-6
- workflow 2-10

FrameMaker+SGML

- frequently used formatting tags 4-8
- index markers 4-13

paragraph markers 4-13
 preparing files for conversion to STF 3-11
 versions supported 1-6

Frequently used formatting tags, FrameMaker
 in FrameMaker+SGML files 4-8
 using 4-8

Frequently used formatting tags, Interleaf
 using 8-8

G

General post production 6-6, 10-6
 Graphics, FrameMaker 3-4
 Graphics, Interleaf 7-4

H

Hard returns, FrameMaker
 preparing for conversion to STF 3-7

Hidden named objects, Interleaf
 preparing for conversion to STF 7-12

Hyphenation option, S-Tagger for FrameMaker 3-16
 source hyphenation 3-16
 translation hyphenation 3-17

Hyphenation option, S-Tagger for Interleaf 7-18
 leave as is 7-18
 manually inserted hyphens 7-19
 turn on 7-18

Hyphenation, FrameMaker
 automatic 3-7
 discretionary 3-8
 preparing for conversion to STF 3-7
 searching for 3-8
 suppress hyphenation characters 3-8

Hyphenation, Interleaf
 automatic 7-9

I

IASCII
 checking files 7-14
 file size in demo mode 1-7
 folder 7-4
 types of file not supported 1-6

ILDOC
 saving files as IASCII 7-14

Index entries, FrameMaker
 translating 4-11

Index entries, Interleaf
 translating 8-11

Index marker sort levels, FrameMaker
 Asian languages 4-16
 Eastern European languages 4-16
 translating 4-12

Index marker sort levels, Interleaf
 Japanese 8-16

Index markers
 in FrameMaker+SGML files 4-13
 preparing for conversion to STF 3-8

Index markers, FrameMaker
 IXgen 3-6
 maximum number of characters in 3-8
 replacing 5-15
 splitting 4-12
 translating 4-11

Index markers, Interleaf
 replacing 9-15
 splitting 8-12
 translating 8-11

Index sort strings, FrameMaker
 translating 4-12

Index sort strings, Interleaf
 translating 8-12

Insert line break option, S-Tagger for FrameMaker 3-7

Insert line break option, S-Tagger for Interleaf 7-9

Interleaf
 graphics 7-4
 preparing files for conversion to STF 7-2
 project folder structure 7-4
 referenced files 7-4
 versions supported 1-6
 workflow 2-10

Internal tags
 about 2-3
 definition of 1-4
 presentation of 2-6
 tag identifiers 2-8

Internal tags, FrameMaker
 using 4-7

Internal tags, Interleaf
 using 8-7

Italics
 using with Asian languages 3-12

IXgen
 preparing markers for conversion to STF 3-6

J

- Japanese WinAlign 3-21, 7-22
- Japanese, Interleaf
 - in markers 8-15
 - character set tags 8-14
 - font mapping 8-15
 - footnotes 7-13
- Japanese (WinAlign) as a source language 7-13
- reserved character tag 8-15
- reserved character tags 7-13
- selecting file format for conversion to STF 7-17
- string length 7-13, 8-15
- unsupported characters 7-13, 8-15

L

- Licence information, S-Tagger for FrameMaker 3-13
- Licence information, S-Tagger for Interleaf 7-15
- Line break before hard return tag option, S-Tagger for FrameMaker 3-19
- Line break before hard return tag option, S-Tagger for Interleaf 7-20
- Line breaks, FrameMaker
 - inserting, not inserting in the STF files 3-19
- Line breaks, Interleaf
 - inserting, not inserting in the STF files 7-20
- List A-29
- List of S-Tags A-29
- List of S-Tags, FrameMaker A-29
- List of S-Tags, Interleaf B-32
- Localisation, FrameMaker
 - added paragraphs, sentences 4-3
 - carriage returns 4-3
 - deleted sentences, words 4-3
 - formatting changes 4-3
 - moved paragraphs 4-3
- Localisation, Interleaf
 - added paragraphs, sentences 8-3
 - carriage returns 8-3
 - deleted sentences, words 8-3
 - formatting changes 8-3
 - moved paragraphs 8-3

M

- Manual line break, *see* Hard returns
 - marker length, Interleaf
 - Asian language limitation 7-23
 - Marker placeholders option, S-Tagger for FrameMaker 3-17
 - Marker placeholders option, S-Tagger for Interleaf 7-19
 - Marker placeholders, FrameMaker
 - in FrameMaker+SGML files 3-17
 - inserting, not inserting in STF files 3-17
 - Marker placeholders, Interleaf
 - inserting, not inserting in STF files 7-19
 - Markers, FrameMaker
 - location of 3-8
 - preparing for conversion to STF 3-8
 - using Western and Asian text in 4-17
 - Markers, Interleaf
 - using Western and Asian text in 8-17
 - Messages, FrameMaker
 - suppressing 5-5
 - Messages, Interleaf
 - suppressing 9-5
 - MIF
 - checking files 3-12
 - converting to STF 3-24, 7-24
 - demo file size 1-7
 - file errors 3-12
 - folder 3-4
 - saving FM files as MIF 3-12
 - testing Asian language files 3-12
 - types of file not supported 1-6
 - MIF file
 - opening 6-6
 - Missing fonts, FrameMaker 3-5
 - Missing language dictionaries, FrameMaker 3-5
 - Missing language dictionaries, Interleaf 7-6
 - Multiple target languages, FrameMaker
 - settings for 3-20
 - Multiple target languages, Interleaf
 - settings for 7-22
-
- ## O
- OLE objects, FrameMaker
 - preparing for conversion to STF 3-9
 - Opening and closing tags
 - about 2-6
 - Opening the new IASII file
 - missing fonts 10-7
 - Opening the New MIF file
 - S-Tagger character formats 6-8
 - unresolved external cross-references 6-7

- Opening the new MIF file 6-6
 - missing fonts 6-6, 6-7
 - missing languages 6-7
- ORG file
 - about 2-5
 - definition of 1-4
- Original file format
 - definition of 1-4
- Overwrite files option, S-Tagger for FrameMaker 3-15
- Overwrite files option, S-Tagger for Interleaf 7-17

P

- Paragraph marks
 - FrameMaker+SGML files 4-13
- Paragraph marks, FrameMaker
 - inserting, not inserting in STF files 3-7
- Paragraph marks, Interleaf
 - inserting, not inserting in STF files 7-9
- Paragraph numbering formats, FrameMaker
 - translating 4-11
- Paragraph numbering formats, Interleaf
 - translating 8-10
- Paragraph styles, FrameMaker
 - in TagEditor 4-10
 - translating 4-10
- Paragraph styles, Interleaf
 - translating 8-10
- Paths tab, S-Tagger for FrameMaker 3-23
- Paths tab, S-Tagger for Interleaf 7-23
- Paths, S-Tagger for FrameMaker
 - remembering 3-23
 - setting the defaults 3-23
- Paths, S-Tagger for Interleaf
 - remembering 7-24
 - setting the defaults 7-24
- PDF document information, FrameMaker
 - preparing for conversion to STF 3-10
 - types of text not supported 3-11
- Post production, general 6-6, 10-6
- Post-production
 - definition of 2-12
- Preferences list, S-Tagger for FrameMaker 3-14
- Prefixes, FrameMaker
 - translating 4-11
- Prefixes, Interleaf
 - translating 8-10
- Preparing, FrameMaker 3-2
 - avoiding font conflicts 3-2

- generated files 3-3
 - platforms 3-2
- Preparing, Interleaf 7-2
 - catalogs 7-3
 - generated files 7-3
 - platforms 7-2
 - renaming files 7-3
- Pre-translation 2-15
 - definition of 2-12
- Printing locked items., Interleaf
 - preparing for conversion to STF 7-12
- Process overview 1-3
- Project structure 2-9
 - MIF/IASCII folder 2-10
 - Source folder 2-9
 - Target folder 2-10
- Project structure, FrameMaker 3-3
 - setting up folders 3-4
- Project structure, Interleaf 7-4
 - setting up folders 7-4

Q

- QuickSilver 1-6
- Quotation marks, FrameMaker
 - S-Tags for 4-6
- Quotation marks, Interleaf
 - S-Tags for 8-6

R

- Read only components option
 - don't translate 7-18
 - Japanese 7-18
 - translate 7-18
- Read only components option, S-Tagger for Interleaf 7-18
- Reference page text, FrameMaker
 - preparing for conversion STF 3-10
- References file, FrameMaker 3-4
- References file, Interleaf 7-4
- Results window, S-Tagger for FrameMaker 5-4
 - file errors 5-8
 - MIF to STF conversion messages 3-25
 - saving text from 3-26
- Results window, S-Tagger for Interleaf 9-4
 - file errors 9-8
 - IASCII to STF conversion messages 7-26
 - saving text from 7-26

Revision tracking, Interleaf
preparing for conversion to STF 7-7

RTF
creating monolingual RTF files 2-16
creating RTF files during clean up 5-19, 9-18
file size in demo mode 1-7

S

S B-29

Sample files 1-2

Saving
in TagEditor 2-16

SDL GXT
settings in S-Tagger for Interleaf 7-21

Settings tab, S-Tagger for FrameMaker 3-14
Preferences list 3-14

SGML elements
preparing for conversion to STF 3-11

Smart quotes
in TagEditor 3-18, 7-19
in Word 3-18, 7-20

Smart quotes option, S-Tagger for FrameMaker 3-18

Smart quotes option, S-Tagger for Interleaf 7-19

Smart quotes, FrameMaker
format names 3-18
in TagEditor 4-6
in Word 4-6

Smart quotes, Interleaf
in TagEditor 8-6
in Word 8-6

Sort levels, FrameMaker
adding 4-16

Sort levels, Interleaf
adding 8-16

Sort strings, FrameMaker
adding 4-16

Sort strings, Interleaf
adding 8-16

Sorted files, FrameMaker
verification alerts, warnings 5-16

Source language, FrameMaker
setting 3-20

Source marker length option, S-Tagger for FrameMaker 3-21

Source marker length option, S-Tagger for Interleaf 7-23

Source marker length, FrameMaker
Asian language limitation 3-21
Western language limitation 3-21

Source sort order, FrameMaker

changing 4-17

Source sort order, Intelleaf
changing 8-17

Source STF files 2-9

S-Tag
list A-29

S-Tag, Interleaf
attributes and control expressions B-6
inline components B-10
markers B-12
prefixes B-17
read only components B-20
shared components B-20
shared named objects B-24
special and reserved characters B-25
tables B-25
unknown inline components B-28

S-Tagger for FrameMaker

About tab 3-13
Convert MIF tab 3-24
launching 3-13
Paths tab 3-23
Verify S-Tags tab 5-4

S-Tagger for Interleaf

About tab 7-15
Convert IASCIITab 7-24
launching 7-15
Paths tab 7-23

S-Tags

about 2-6
as placeables 4-4, 8-4
components 2-7
definition of 1-4
displaying in TagEditor, Word 2-6
types of 2-6

S-Tags in the STF ancillary file B-29

S-Tags, FrameMaker

adding, deleting, moving 4-4
anchored frame appearance in STF file A-3, B-3
anchored frames A-3
character formatting A-5
character style 4-9
cross-references 4-9
extended characters, symbols 4-5
font changes 4-9
formatting 4-5
internal tags 4-7
list A-29
quotation marks 4-6
rules 4-4
using with previous translation memories 4-4
variables 4-9
verifying 5-6

S-Tags, Interleaf

- adding, deleting, moving 8-4
- anchored frame appearance in STF file B-3
- anchored frames B-3
- ancillary file B-29
- attribute references B-7
- character style appearance in STF file B-11
- control expression appearance in STF file B-6
- cross-reference variables B-9
- cross-references B-7
- extended characters, symbols 8-5
- footnotes B-9
- formatting 8-5
- index marker appearance in STF file B-13
- internal tags 8-7
- list B-32
- paragraph style appearance in STF file B-15
- paragraph styles (components) B-15
- prefix appearance in STF file B-17
- quotation marks 8-6
- read only component appearance in STF file B-20
- rules 8-4
- shared anchored frame appearance in STF file B-5
- shared component appearance in STF file B-20
- table appearance in STF file B-25
- verifying 9-6

S-tags, Interleaf

- shared frames B-22

S-Tags, Interleaf

- anchored frame without text object appearance in STF file B-4

Stand-alone tags 2-7

STF file

- anchored frame A-3, B-3
- anchored frame without text object B-4
- character style B-11
- control expressions B-6
- index markers B-13
- paragraph styles B-15
- prefixes B-17
- read only components B-20
- shared anchored frames B-5
- shared components B-20
- tables B-25

STF file format option, S-Tagger for FrameMaker 3-15

STF file format option, S-Tagger for Interleaf 7-17

STF files

- about 2-2
- ancillary file 2-3
- creation of 2-9
- definition of 1-4
- extensions 2-9
- file formats 2-2
- naming conventions 2-9
- overview 1-3
- saving 2-16

- source files 2-9
- target STF files 2-9
- translation of 2-2
- See also* Ancillary file

STF files, FrameMaker

- testing 3-26

STF files, Interleaf

- testing 7-26

STF tags

- list of B-32--??

String length, Interleaf

- Japanese 8-15

Structural information

- external tags 2-6

Suppress hyphenation characters, FrameMaker

- preparing for conversion to STF 3-8

Symbols and special characters, FrameMaker

- converting to STF 3-9

System requirements 1-6

T

Tag alerts, FrameMaker 5-13

Tag alerts, Interleaf 9-13

Tag errors, FrameMaker 5-12

Tag errors, Interleaf 9-12

Tag identifiers 2-8

- presentation of 2-8

Tag identifying number 2-8

Tag name 2-8

Tag pairs

- about 2-6

Tag text 2-8

Tag type identifiers 2-8

Tag warnings, FrameMaker 5-16

Tag warnings, Interleaf 9-16

TagEditor

- advantages of 2-13
- Context TM 2-13
- copying and pasting tags 4-10, 8-10
- creating monolingual RTF files 4-3, 8-3
- Insert Tag dialog box 4-6, 8-6
- inserting tags 4-6, 8-6
- S-Tag Verifier plug-in 5-2, 9-2
- tag protection and verification 5-9, 9-9
- Tags toolbar 4-6, 8-6
- workflow 2-13

Target language setting, FrameMaker

- multiple target languages 3-20

Target language setting, Interleaf
multiple target languages 7-22

Target STF files 2-9, 2-10

Text files 2-10

Text insets, FrameMaker
inserting as OLE objects 3-10
preparing for conversion to STF 3-9

TRADOS tag
definition of 1-4

TRADOS tag, FrameMaker
converting to 3-15

TRADOS tag, Interleaf
converting to 7-17

Translated MIF file 6-6

Translating
STF files, about 2-16

Translating to Japanese, Interleaf 8-14

Translating, FrameMaker
ancillary file 4-3
file format 4-3
formatting tags 4-5
quotation marks 4-6
special character tags 4-5
workflow 4-2

Translating, Interleaf
ancillary file 8-3
file format 8-3
formatting tags 8-5
quotation marks 8-6
special character tags 8-5
workflow 8-2

Translation hand-off, FrameMaker
delivering RTF files 4-3

Translation hand-off, Interleaf
delivering RTF files 8-3

Translation match values
in TagEditor 2-7
in Word 2-7

Translation memory
match values 2-7
updating 2-17, 5-19, 9-18

Translation unit tags 2-7

Translator's Workbench
analysis 2-15
BAK file 2-15
Clean Up command 5-19
editing environments 2-2
for translating STF files 1-3
handling S-Tags 4-4, 8-4
pre-translation 2-15
translation memories 4-4, 8-4
updating the translation memory 5-19, 9-18

TXT, file size in demo mode 1-7

TXT, FrameMaker
converting to 3-15

TXT, Interleaf
converting to 7-17

U

Unique identifying number 2-8

Unsupported characters, FrameMaker
Asian languages 4-15
Eastern European languages 4-15

Unsupported characters, Interleaf
Japanese 8-15

Updating
translation memory 2-17

V

Variables, FrameMaker
S-Tags for 4-9

Verification messages, FrameMaker 5-9

Verification messages, Interleaf 9-9

Verifier report
about 2-5

Verifier report, FrameMaker 3-21

Verifier report, Interleaf 7-23

Verifier report, S-Tagger for FrameMaker
customising 5-4
errors, warning, alerts 5-4
Results window 5-4
suppressing messages 5-4

Verifier report, S-Tagger for Interleaf
customising 9-4
errors, warning, alerts 9-4
Results window 9-4
suppressing messages 9-4

Verifying
definition of 1-5
using TagEditor 2-16

Verifying the S-Tags 9-6

Verifying, FrameMaker
about 5-2
checking the conversion settings 5-3
process overview 5-6
rules 5-2
selecting files for verification 5-7
suppressing alerts, warnings 3-22
TagEditor 5-3

- TagEditor plug-in 5-2
- Word errors 5-17
- Verifying, Interleaf
 - about 9-2
 - checking the conversion settings 9-3
 - process overview 9-6
 - selecting files for verification 9-7
 - suppressing alerts, warnings 7-23
- TagEditor 9-3
- TagEditor plug-in 9-2
- Word errors 9-16
- Version compatibility
 - FrameMaker 1-6, 3-3
 - Interleaf 7-3

W

- Warnings, FrameMaker
 - correcting 5-10
 - sorted files 5-16
- Warnings, Interleaf
 - correcting 9-10
- Warnings, S-Tagger for FrameMaker
 - suppressing 3-21
 - suppressing, displaying 5-4
- Warnings, S-Tagger for Interleaf
 - suppressing 7-23
- Western characters, FrameMaker
 - retaining in translated files 4-14
- Western characters, Interleaf
 - retaining in translated files 8-14
- Word
 - hidden text 5-17, 9-17
 - tag formatting errors 5-17, 9-16
 - workflow 2-17
- Workflows
 - overview of 2-10
 - TagEditor 2-13
 - Word 2-17
- Workflows, FrameMaker
 - translating workflow 4-2
- Workflows, Interleaf
 - translating workflow 8-2