

Progress Translation Manager Guide

© 2001 Progress Software Corporation. All rights reserved.

Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. This manual is also copyrighted and all rights are reserved. This manual may not, in whole or in part, be copied, photocopied, translated, or reduced to any electronic medium or machine-readable form without prior consent, in writing, from Progress Software Corporation.

The information in this manual is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear in this document.

The references in this manual to specific platforms supported are subject to change.

Progress, Progress Results, Provision and WebSpeed are registered trademarks of Progress Software Corporation in the United States and other countries. Apptivity, AppServer, ProVision Plus, SmartObjects, IntelliStream, and other Progress product names are trademarks of Progress Software Corporation.

SonicMQ is a trademark of Sonic Software Corporation in the United States and other countries.

Progress Software Corporation acknowledges the use of Raster Imaging Technology copyrighted by Snowbound Software 1993-1997 and the IBM XML Parser for Java Edition.

© IBM Corporation 1998-1999. All rights reserved. U.S. Government Users Restricted Rights — Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Progress is a registered trademark of Progress Software Corporation and is used by IBM Corporation in the mark Progress/400 under license. Progress/400 AND 400® are trademarks of IBM Corporation and are used by Progress Software Corporation under license.

Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Any other trademarks and/or service marks contained herein are the property of their respective owners.

.

May 2001



Product Code: 4539
Item Number: 81109;9.1C

Contents

Preface . . . . . **xi**
Purpose xi
Audience xi
Organization of This Manual xi
How to Use This Manual xiii
Typographical Conventions xiii
Syntax Notation xiv
Example Procedures xviii
Progress Messages xix
Other Useful Documentation xxi
Getting Started xxi
Development Tools xxiii
Reporting Tools xxiv
4GL xxiv
Database xxv
DataServers xxv
SQL-89/Open Access xxvi
SQL-92 xxvi
Deployment xxvii
WebSpeed xxvii
Reference xxviii
SQL-92 Reference xxviii

1. Preparing Your Application for Translation 1–1
1.1 Advantages of Planning Ahead 1–2
1.2 4GL Considerations 1–3
1.2.1 Defining Text as Untranslatable 1–3
1.2.2 Defining Alignment and Space 1–3
1.2.3 Defining Accelerator Keys 1–4
1.2.4 Using Default Buttons 1–4
1.3 Screen Layout Considerations 1–4
1.3.1 Explicitly Sizing and Positioning Widgets 1–5
1.3.2 Planning for Text Expansion 1–6
1.3.3 Creating Adjustable Text on Rectangles 1–8
1.3.4 Length and Width of Text Phrases 1–8
1.3.5 Creating Message Statements 1–9
1.3.6 Adjusting Database Field Justification 1–10
1.4 Where to Go for More Information 1–10

2. Overview 2–1
2.1 Overview of the Progress Translation Process 2–2
2.2 Tasks for the Project Manager 2–5
2.3 Tasks for the Translator 2–7

3. Getting Started 3–1
3.1 Upgrading from Version 8 to Version 9 3–2
3.2 Organizing a Directory Structure for the Source Code 3–2
3.3 Setting the PROPATH 3–3
3.4 Creating and Storing Environment (.ini) Files 3–4
3.5 Starting the Translation Manager 3–5
3.6 Connecting to the Application Databases 3–6

4. Managing Projects 4–1
4.1 What Is a Project? 4–2
4.2 Determining the Number of Projects to Create for an Application 4–3
4.3 Creating a Project 4–5
4.4 Opening a Project 4–7
4.5 Selecting the Current Project 4–7
4.6 Closing a Project 4–8
4.7 Selecting Source Procedures 4–8
4.7.1 Adding Files to the Procedures List 4–9
4.7.2 Removing Files from the Procedures List 4–12
4.7.3 Viewing and Editing a File in the Procedures List 4–13

5. Preparing Data for Translation 5–1
5.1 Extracting Text Phrases from Source Procedures 5–2
5.2 Specifying Filters 5–6
5.2.1 Exporting and Importing Filters 5–7
5.3 Loading Text Phrases into the Project Database 5–8
5.4 Generating Resource Procedures 5–12
5.5 Working on Subsequent Iterations of a Project 5–14
5.5.1 Procedure Subsets and Supersets 5–15
5.5.2 Using Procedure Subsets 5–16
5.6 Preparing Text from a Character Application 5–17
5.7 Using Third-party Tools 5–18

6. Adding Glossaries 6–1
6.1 Creating a Glossary 6–2
6.2 Importing a Glossary 6–4
6.3 Exporting a Glossary 6–8
6.4 Tracking Glossary Entries 6–8

7. Preparing a Kit 7–1
7.1 What Is a Kit? 7–2
7.2 Building a Kit 7–2
7.3 Deleting a Kit 7–3
7.4 Pretranslation 7–4
7.5 Changing a Kit's Source Language 7–7
7.6 Sending a Kit to the Translator 7–9

8. Incorporating a Translated Kit into the Project 8–1
8.1 Receiving the Kit from the Translator 8–2
8.2 Consolidating the Translated Kit into the Project Database 8–2
8.3 Compiling the Source Code with the Translated Phrases 8–4
8.3.1 Text Segment Expansion 8–6
8.4 Running the Compiled Code in a Target Language 8–8
8.4.1 Running Code from the Command Line 8–9

9. Updating a Project 9–1
9.1 Printing a Statistics Report 9–2
9.2 Editing Translations and Glossary Entries 9–3
9.2.1 Browsing Through the Tab Folders 9–3
9.2.2 Sorting Rows and Ordering Columns 9–3
9.2.3 Inserting and Deleting Rows 9–3
9.2.4 Cutting, Copying, and Pasting Text 9–3
9.3 Scanning for Changed Source Files 9–4

10. Tutorial 10–1
10.1 Preparing to Use This Tutorial 10–2
10.2 Creating a New Translation Manager Project 10–3
10.3 The Procedures Folder: Adding Files 10–4
10.4 The Data Tab Folder 10–6
10.4.1 Extracting Text Phrases 10–6
10.4.2 Specifying Filters 10–7
10.4.3 Loading Text Phrases 10–9
10.4.4 Creating Resource Procedures 10–10
10.5 The Glossary Folder 10–11
10.6 The Kits Folder 10–12
10.6.1 Creating a Kit 10–13
10.6.2 Creating a Zip File 10–13
10.7 Installing a Zip File and Consolidating a Kit Received from the Translator 10–14
10.8 Compiling Source Code with the Translated Phrases 10–15

11. Troubleshooting 11–1

A. Translation Manager Interface Reference A–1
A.1 Tab Folders A–2
A.1.1 Procedures Tab Folder A–3
A.1.2 Data Tab Folder A–5
A.1.3 Glossary Tab Folder A–7
A.1.4 Kits Tab Folder A–8
A.1.5 Statistics Tab Folder A–10
A.2 Menu Bar A–11
A.3 Toolbar A–11

Glossary Glossary–1

Index Index–1

Figures

Figure 2–1: Overview of Translating a Progress Application 2–3
Figure 2–2: The Translation Manager Translation Process 2–4
Figure 2–3: Projects and Kits 2–6
Figure 3–1: Organization of Ideal Directory Structure 3–3
Figure 3–2: Translation Manager Main Window 3–5
Figure 5–1: The Extract Process 5–2
Figure 5–2: Example STRING-XREF File 5–3
Figure 5–3: The Load Process 5–9
Figure 5–4: Data Tab Folder 5–11
Figure 5–5: Long String Translator Dialog Box 5–11
Figure A–1: Procedures Tab Folder A–3
Figure A–2: Data Tab Folder A–5
Figure A–3: Glossary Tab Folder A–7
Figure A–4: Kits Tab Folder A–9
Figure A–5: Statistics Tab Folder A–10

Tables

Table 1–1:	Standard Text Expansion Table for English Source Language	1–7
Table 1–2:	Documentation Cross Reference	1–10
Table 6–1:	Examples of Imported Glossaries	6–4
Table 6–2:	Example Glossary Entries	6–9
Table 8–1:	Translation Manager Internal Text Expansion	8–7
Table 8–2:	Text Expansion Examples	8–8
Table A–1:	Toolbar Buttons	A–11

Procedures

tmproc1.p 1–5
tmproc2.p 1–6

Preface

Purpose

This book describes how to use the Progress Version 9 Translation Manager tool to manage the process of translating a Progress 4GL application. It describes how the Translation Manager helps you through each step of the translation process, from the moment you receive the untranslated source code to the moment the completely translated application is running and ready to ship as a multi-lingual product.

Audience

This book is written for the translation project manager. The project manager is the person in a company who manages the entire process of translating Progress 4GL applications into multiple languages. The project manager's responsibilities include budgeting the translation process, hiring the translators, assembling the translation kits, monitoring the progress of the translators, answering questions regarding the application, reviewing translations, working with developers to create translatable applications, and building the final run-time code.

Organization of This Manual

Chapter 1, "Preparing Your Application for Translation"

Describes techniques that you can share with software developers to make your company's applications easier to translate.

Chapter 2, "Overview"

Provides an overview of the Translation Management System and the tasks involved in using the Translation Manager and Visual Translator tools.

Chapter 3, “Getting Started”

Describes how to organize a directory structure, set your PROPATH, start the Translation Manager tool, and connect to application databases.

Chapter 4, “Managing Projects”

Details how to create, open, and close a project database, select the current project, and select the source procedures you want to translate.

Chapter 5, “Preparing Data for Translation”

Explains how to prepare the data (text phrases and resource procedures) the translators will use to translate user-interface procedures.

Chapter 6, “Adding Glossaries”

Describes how to create, import, and export the lexical glossaries translators will use.

Chapter 7, “Preparing a Kit”

Describes how to build a language kit for the translator. It also describes how to send the language kit to the translators.

Chapter 8, “Incorporating a Translated Kit into the Project”

Details how to consolidate the translations and glossary entries from a translated language kit database into the project database.

Chapter 9, “Updating a Project”

Shows how to edit translations and glossary entries and scan for changed source files.

Chapter 10, “Tutorial”

Guides you through the steps of a sample translation project.

Chapter 11, “Troubleshooting”

Provides answers to some common questions and describes solutions to common problems.

Appendix A, "Translation Manager Interface Reference"

Provides reference information for the Translation Manager's tab folders, menu bar, and tool bar.

"Glossary"

How to Use This Manual

This book is a guide to the Progress Translation Manager tool. Although you can use it as a reference, it is structured so that each chapter builds on the previous chapters. Therefore, you will learn the most if you start at Chapter 1 and read through until the end.

Typographical Conventions

This manual uses the following typographical conventions:

- **Bold typeface** indicates:
  - Commands or characters that the user types
  - That a word carries particular weight or emphasis
- *Italic typeface* indicates:
  - Progress variable information that the user supplies
  - New terms
  - Titles of complete publications
- Monospaced typeface indicates:
  - Code examples
  - System output
  - Operating system filenames and pathnames

The following typographical conventions are used to represent keystrokes:

- Small capitals are used for Progress key functions and generic keyboard keys.

  END-ERROR, GET, GO
  ALT, CTRL, SPACEBAR, TAB

- When you have to press a combination of keys, they are joined by a dash. You press and hold down the first key, then press the second key.

  CTRL-X

- When you have to press and release one key, then press another key, the key names are separated with a space.

  ESCAPE H
  ESCAPE CURSOR-LEFT

Syntax Notation

The syntax for each component follows a set of conventions:

- Uppercase words are keywords. Although they are always shown in uppercase, you can use either uppercase or lowercase when using them in a procedure.

  In this example, ACCUM is a keyword:

  SYNTAX

  ```
  ACCUM aggregate expression
  ```

- Italics identify options or arguments that you must supply. These options can be defined as part of the syntax or in a separate syntax identified by the name in italics. In the ACCUM function above, the *aggregate* and *expression* options are defined with the syntax for the ACCUM function in the *Progress Language Reference*.

- You must end all statements (except for DO, FOR, FUNCTION, PROCEDURE, and REPEAT) with a period. DO, FOR, FUNCTION, PROCEDURE, and REPEAT statements can end with either a period or a colon, as in this example:

  ```
  FOR EACH Customer:
     DISPLAY Name.
  END.
  ```

- Square brackets (**[ ]**) around an item indicate that the item, or a choice of one of the enclosed items, is optional.

In this example, STREAM *stream*, UNLESS-HIDDEN, and NO-ERROR are optional:

SYNTAX

```
DISPLAY [ STREAM stream ] [ UNLESS-HIDDEN ] [ NO-ERROR ]
```

In some instances, square brackets are not a syntax notation, but part of the language.

For example, this syntax for the INITIAL option uses brackets to bound an initial value list for an array variable definition. In these cases, normal text brackets ([ ]) are used:

SYNTAX

```
INITIAL [ constant [ , constant ] ... ]
```

NOTE: The ellipsis (**. . .**) indicates repetition, as shown in a following description.

- Braces (**{ }**) around an item indicate that the item, or a choice of one of the enclosed items, is required.

  In this example, you must specify the items BY and *expression* and can optionally specify the item DESCENDING, in that order:

  SYNTAX

  ```
  { BY expression [ DESCENDING ] }
  ```

  In some cases, braces are not a syntax notation, but part of the language.

  For example, a called external procedure must use braces when referencing arguments passed by a calling procedure. In these cases, normal text braces ({ }) are used:

  SYNTAX

  ```
  { &argument-name }
  ```

- A vertical bar (|) indicates a choice.

  In this example, EACH, FIRST, and LAST are optional, but you can only choose one:

  SYNTAX

  ```
  PRESELECT [ EACH | FIRST | LAST ] record-phrase
  ```

  In this example, you must select one of *logical-name* or *alias*:

  SYNTAX

  ```
  CONNECTED ( { logical-name | alias } )
  ```

- Ellipses (**. . .**) indicate that you can choose one or more of the preceding items. If a group of items is enclosed in braces and followed by ellipses, you must choose one or more of those items. If a group of items is enclosed in brackets and followed by ellipses, you can optionally choose one or more of those items.

  In this example, you must include two expressions, but you can optionally include more. Note that each subsequent expression must be preceded by a comma:

  SYNTAX

  ```
  MAXIMUM ( expression , expression [ , expression ] ... )
  ```

  In this example, you must specify MESSAGE, then at least one of *expression* or SKIP, but any additional number of *expression* or SKIP is allowed:

  SYNTAX

  ```
  MESSAGE { expression | SKIP [ (n) ] } ...
  ```

  In this example, you must specify {*include-file*, then optionally any number of *argument* or &*argument-name* = "*argument-value*", and then terminate with }:

  SYNTAX

  ```
  { include-file
      [ argument | &argument-name = "argument-value" ] ... }
  ```

- In some examples, the syntax is too long to place in one horizontal row. In such cases, **optional** items appear individually bracketed in multiple rows in order, left-to-right and top-to-bottom. This order generally applies, unless otherwise specified. **Required** items also appear on multiple rows in the required order, left-to-right and top-to-bottom. In cases where grouping and order might otherwise be ambiguous, braced (required) or bracketed (optional) groups clarify the groupings.

  In this example, WITH is followed by several optional items:

  SYNTAX

```
WITH [ ACCUM max-length ] [ expression DOWN ]
  [ CENTERED ] [ n COLUMNS ] [ SIDE-LABELS ]
  [ STREAM-IO ]
```

  In this example, ASSIGN requires one of two choices: either one or more of *field*, or one of *record*. Other options available with either *field* or *record* are grouped with braces and brackets. The open and close braces indicate the required order of options:

  SYNTAX

```
ASSIGN {   { [ FRAME frame ]
               { field [ = expression ] }
               [ WHEN expression ]
           } ...
       |  { record [ EXCEPT field ... ] }
     }
```

Example Procedures

This manual provides numerous example procedures that illustrate syntax and concepts. Examples use the following conventions:

- They appear in boxes with borders.
- If they are available online, the name of the procedure appears above the left corner of the box and starts with a prefix associated with the manual that references it, as follows:
  - `e-` — *Progress External Program Interfaces*, for example, `e-ddeex1.p`
  - `lt-` — *Progress Language Tutorial*, for example, `lt-05-s3.p`
  - `p-` — *Progress Programming Handbook*, for example, `p-br01.p`
  - `r-` — *Progress Language Reference*, for example, `r-dynbut.p`

  If the name does not start with a listed prefix, the procedure is not available online.
- If they are not available online, they compile as shown, but might not execute for lack of completeness.

Accessing Files in Procedure Libraries

Documentation examples are stored in procedure libraries, `prodoc.pl` and `prohelp.pl`, in the `src` directory where Progress is installed.

You must first create all subdirectories required by a library before attempting to extract files from the library. You can see what directories and subdirectories a library needs by using the PROLIB -list command to view the contents of the library. See the *Progress Client Deployment Guide* for more details on the PROLIB utility.

Extracting source files from a procedure library involves running PROENV to set up your Progress environment, creating the directory structure for the files you want to extract, and running PROLIB.

1 ♦ From the Control Panel or the Progress Program Group, double-click the Proenv icon.

2 ♦ The Proenv Window appears, with the proenv prompt.

Running Proenv sets the DLC environment variable to the directory where you installed Progress (by default, `C:\Program Files\Progress`). Proenv also adds the DLC environment variable to your PATH environment variable and adds the bin directory (`PATH=%DLC%;%DLC%\bin;%PATH%`).

3 ♦ Enter the following command at the proenv prompt to create the `prodoc` directory in your Progress working directory (by default, `C:\Progress\Wrk`):

```
MKDIR prodoc
```

4 ♦ Create the `langref` directory under `prodoc`:

```
MKDIR prodoc\langref
```

5 ♦ To extract all examples in a procedure library directory, run the PROLIB utility. Note that you must use double quotes because "Program Files" contains an embedded space:

```
PROLIB "%DLC%\src\prodoc.pl" -extract prodoc\langref\*.*
```

PROLIB extracts all examples into `prodoc\langref`.

To extract one example, run PROLIB and specify the file that you want to extract as it is stored in the procedure library:

```
PROLIB "%DLC%\src\prodoc.pl" -extract prodoc/langref/r-syshlp.p
```

PROLIB extracts `r-syshlp.p` into `prodoc\langref`.

Progress Messages

Progress displays several types of messages to inform you of routine and unusual occurrences:

- Execution messages inform you of errors encountered while Progress is running a procedure (for example, if Progress cannot find a record with a specified index field value).
- Compile messages inform you of errors found while Progress is reading and analyzing a procedure prior to running it (for example, if a procedure references a table name that is not defined in the database).

- Startup messages inform you of unusual conditions detected while Progress is getting ready to execute (for example, if you entered an invalid startup parameter).

After displaying a message, Progress proceeds in one of several ways:

- Continues execution, subject to the error-processing actions that you specify, or that are assumed, as part of the procedure. This is the most common action taken following execution messages.
- Returns to the Progress Procedure Editor so that you can correct an error in a procedure. This is the usual action taken following compiler messages.
- Halts processing of a procedure and returns immediately to the Procedure Editor. This does not happen often.
- Terminates the current session.

Progress messages end with a message number in parentheses. In this example, the message number is 200:

```
** Unknown table name table. (200)
```

Use Progress online help to get more information about Progress messages. On the Windows platform, many Progress tools include the following Help menu options to provide information about messages:

- Choose Help→ Recent Messages to display detailed descriptions of the most recent Progress message and all other messages returned in the current session.
- Choose Help→ Messages, then enter the message number to display a description of any Progress message. (If you encounter an error that terminates Progress, make a note of the message number before restarting.)
- In the Procedure Editor, press the **HELP** key (**F2** or **CTRL-W**).

On the UNIX platform, you can use the Progress PRO command to start a single-user mode character Progress client session and view a brief description of a message by providing its number. Follow these steps:

1 ♦ Start the Progress Procedure Editor:

```
install-dir/dlc/bin/pro
```

2 ♦ Press **F3** to access the menu bar, then choose Help→ Messages.

3 ♦ Type the message number, and press **ENTER**. Details about that message number appear.

4 ♦ Press **F4** to close the message, press **F3** to access the Procedure Editor menu, and choose File→ Exit.

Other Useful Documentation

This section lists Progress Software Corporation documentation that you might find useful. Unless otherwise specified, these manuals support both Windows and Character platforms and are provided in electronic documentation format on CD-ROM.

Getting Started

Progress Electronic Documentation Installation and Configuration Guide (Hard copy only)

A booklet that describes how to install the Progress EDOC viewer and collection on UNIX and Windows.

Progress Installation and Configuration Guide Version 9 for UNIX

A manual that describes how to install and set up Progress Version 9.1 for the UNIX operating system.

Progress Installation and Configuration Guide Version 9 for Windows

A manual that describes how to install and set up Progress Version 9.1 for all supported Windows and Citrix MetaFrame operating systems.

Progress Version 9 Product Update Bulletin

A guide that provides a brief description of each new feature of the release. The booklet also explains where to find more detailed information in the documentation set about each new feature.

Progress Application Development Environment — Getting Started (Windows only)

A practical guide to graphical application development within the Progress Application Development Environment (ADE). This guide includes an overview of the ADE and its tools, an overview of Progress SmartObject technology, and tutorials and exercises that help you better understand SmartObject technology and how to use the ADE to develop applications.

Progress Language Tutorial for Windows and *Progress Language Tutorial for Character*

Platform-specific tutorials designed for new Progress users. The tutorials use a step-by-step approach to explore the Progress application development environment using the 4GL.

Progress Master Glossary for Windows and *Progress Master Glossary for Character* (EDOC only)

Platform-specific master glossaries for the Progress documentation set. These books are in electronic format only.

Progress Master Index and Glossary for Windows and *Progress Master Index and Glossary for Character* (Hard copy only)

Platform-specific master indexes and glossaries for the Progress hard-copy documentation set.

Progress Startup Command and Parameter Reference

A reference manual that describes the Progress startup commands and parameters in alphabetical order.

Welcome to Progress (Hard copy only)

A booklet that explains how Progress software and media are packaged. An icon-based map groups the documentation by functionality, providing an overall view of the documentation set. *Welcome to Progress* also provides descriptions of the various services Progress Software Corporation offers.

Development Tools

Progress ADM 2 Guide

A guide to using the Application Development Model, Version 2 (ADM 2) application architecture to develop Progress applications. It includes instructions for building and using Progress SmartObjects.

Progress ADM 2 Reference

A reference for the Application Development Model, Version 2 (ADM 2) application. It includes descriptions of ADM 2 functions and procedures.

Progress AppBuilder Developer's Guide (Windows only)

A programmer's guide to using the Progress AppBuilder visual layout editor. AppBuilder is a Rapid Application Development (RAD) tool that can significantly reduce the time and effort required to create Progress applications.*Progress Basic Database Tools* (Character only; information for Windows is in online help)

A guide for the Progress Database Administration tools, such as the Data Dictionary.

Progress Basic Development Tools (Character only; information for Windows is in online help)

A guide for the Progress development toolset, including the Progress Procedure Editor and the Application Compiler.

Progress Debugger Guide

A guide for the Progress Application Debugger. The Debugger helps you trace and correct programming errors by allowing you to monitor and modify procedure execution as it happens.

Progress Help Development Guide (Windows only)

A guide that describes how to develop and integrate an online help system for a Progress application.

Progress Visual Translator Guide (Windows only)

A guide that describes how to use the Progress Visual Translator tool to translate text phrases from procedures into one or more spoken languages.

Reporting Tools

Progress Report Builder Deployment Guide (Windows only)

An administration and development guide for generating Report Builder reports using the Progress Report Engine.

Progress Report Builder Tutorial (Windows only)

A tutorial that provides step-by-step instructions for creating eight sample Report Builder reports.

Progress Report Builder User's Guide (Windows only)

A guide for generating reports with the Progress Report Builder.

Progress Results Administration and Development Guide (Windows only)

A guide for system administrators that describes how to set up and maintain the Results product in a graphical environment. This guide also describes how to program, customize, and package Results with your own products. In addition, it describes how to convert character-based Results applications to graphical Results applications.

Progress Results User's Guide for Windows and *Progress Results User's Guide for UNIX*

Platform-specific guides for users with little or no programming experience that explain how to query, report, and update information with Results. Each guide also helps advanced users and application developers customize and integrate Results into their own applications.

4GL

Building Distributed Applications Using the Progress AppServer

A guide that provides comprehensive information about building and implementing distributed applications using the Progress AppServer. Topics include basic product information and terminology, design options and issues, setup and maintenance considerations, 4GL programming details, and remote debugging.

Progress External Program Interfaces

A guide to accessing non-Progress applications from Progress. This guide describes how to use system clipboards, UNIX named pipes, Windows dynamic link libraries, Windows dynamic data exchange, Windows ActiveX controls, and the Progress Host Language Call Interface to communicate with non-Progress applications and extend Progress functionality.

Progress Internationalization Guide

A guide to developing Progress applications for markets worldwide. The guide covers both internationalization—writing an application so that it adapts readily to different locales (languages, cultures, or regions)—and localization—adapting an application to different locales.

Progress Language Reference

A three-volume reference set that contains extensive descriptions and examples for each statement, phrase, function, operator, widget, attribute, method, and event in the Progress language.

Progress Programming Handbook

A two-volume handbook that details advanced Progress programming techniques.

Database

Progress Database Design Guide

A guide that uses a sample database and the Progress Data Dictionary to illustrate the fundamental principles of relational database design. Topics include relationships, normalization, indexing, and database triggers.

Progress Database Administration Guide and Reference

This guide describes Progress database administration concepts and procedures. The procedures allow you to create and maintain your Progress databases and manage their performance.

DataServers

Progress DataServer Guides

These guides describe how to use the DataServers to access non-Progress databases. They provide instructions for building the DataServer modules, a discussion of programming considerations, and a tutorial. Each DataServer has its own guide, for example, the *Progress DataServer for ODBC Guide*, the *Progress DataServer for ORACLE Guide*, or the *Progress/400 Product Guide*.

MERANT ODBC Branded Driver Reference

The Enterprise DataServer for ODBC includes MERANT ODBC drivers for all the supported data sources. For configuration information, see the MERANT documentation, which is available as a PDF file in *installation-path*\odbc. To read this file you must have the Adobe Acrobat Reader Version 3.1 or higher installed on your system. If you do not have the Adobe Acrobat Reader, you can download it from the Adobe Web site at: http://www.adobe.com/prodindex/acrobat/readstep.html.

SQL-89/Open Access

Progress Embedded SQL-89 Guide and Reference

A guide to Progress Embedded SQL-89 for C, including step-by-step instructions on building ESQL-89 applications and reference information on all Embedded SQL-89 Preprocessor statements and supporting function calls. This guide also describes the relationship between ESQL-89 and the ANSI standards upon which it is based.

Progress Open Client Developer's Guide

A guide that describes how to write and deploy Java and ActiveX applications that run as clients of the Progress AppServer. The guide includes information about how to expose the AppServer as a set of Java classes or as an ActiveX server.

Progress SQL-89 Guide and Reference

A user guide and reference for programmers who use interactive Progress/SQL-89. It includes information on all supported SQL-89 statements, SQL-89 Data Manipulation Language components, SQL-89 Data Definition Language components, and supported Progress functions.

SQL-92

Progress Embedded SQL-92 Guide and Reference

A guide to Progress Embedded SQL-92 for C, including step-by-step instructions for building ESQL-92 applications and reference information about all Embedded SQL-92 Preprocessor statements and supporting function calls. This guide also describes the relationship between ESQL-92 and the ANSI standards upon which it is based.

Progress JDBC Driver Guide

A guide to the Java Database Connectivity (JDBC) interface and the Progress SQL-92 JDBC driver. It describes how to set up and use the driver and details the driver's support for the JDBC interface.

Progress ODBC Driver Guide

A guide to the ODBC interface and the Progress SQL-92 ODBC driver. It describes how to set up and use the driver and details the driver's support for the ODBC interface.

Progress SQL-92 Guide and Reference

A user guide and reference for programmers who use Progress SQL-92. It includes information on all supported SQL-92 statements, SQL-92 Data Manipulation Language components, SQL-92 Data Definition Language components, and Progress functions. The guide describes how to use the Progress SQL-92 Java classes and how to create and use Java stored procedures and triggers.

Deployment

Progress Client Deployment Guide

A guide that describes the client deployment process and application administration concepts and procedures.

Progress Developer's Toolkit

A guide to using the Developer's Toolkit. This guide describes the advantages and disadvantages of different strategies for deploying Progress applications and explains how you can use the Toolkit to deploy applications with your selected strategy.

Progress Portability Guide

A guide that explains how to use the Progress toolset to build applications that are portable across all supported operating systems, user interfaces, and databases, following the Progress programming model.

WebSpeed

Getting Started with WebSpeed

Provides an introduction to the WebSpeed Workshop tools for creating Web applications. It introduces you to all the components of the WebSpeed Workshop and takes you through the process of creating your own Intranet application.

WebSpeed Installation and Configuration Guide

Provides instructions for installing WebSpeed on Windows and UNIX systems. It also discusses designing WebSpeed environments, configuring WebSpeed Brokers, WebSpeed Agents, and the NameServer, and connecting to a variety of data sources.

WebSpeed Developer's Guide

Provides a complete overview of WebSpeed and the guidance necessary to develop and deploy WebSpeed applications on the Web.

WebSpeed Version 3 Product Update Bulletin

A booklet that provides a brief description of each new feature of the release. The booklet also explains where to find more detailed information in the documentation set about each new feature.

Welcome to WebSpeed (Hard copy only)

A booklet that explains how WebSpeed software and media are packaged. *Welcome to WebSpeed* also provides descriptions of the various services Progress Software Corporation offers.

Reference

Pocket Progress (Hard copy only)

A reference that lets you quickly look up information about the Progress language or programming environment.

Pocket WebSpeed (Hard copy only)

A reference that lets you quickly look up information about the SpeedScript language or the WebSpeed programming environment.

SQL-92 Reference

(These are non-Progress resources available from your technical bookseller.)

A Guide to the SQL Standard

Date, C.J., with Hugh Darwen. 1997. Reading, MA: Addison Wesley.

Understanding the New SQL: A Complete Guide

Melton, Jim (Digital Equipment Corporation) and Alan R. Simon. 1993. San Francisco: Morgan Kaufmann Publishers.

1

Preparing Your Application for Translation

If you are going to translate your software, it is important to think about internationalization issues during the design phase. If your software can support a variety of cultural requirements (different alphabets, different date formats, translations of user-interface elements), you will have access to a larger worldwide market than if your software only works for a single audience. And if you have designed your software with translation issues in mind, the entire translation process will be simpler and less costly with a better end product.

Before using the Translation Manager, read the *Progress Internationalization Guide*, which has information on internationalization and localization issues that your developers should be aware of in order to make the translation process as smooth as possible

This chapter describes the following topics:

- Advantages of planning ahead
- 4GL considerations
- Screen layout considerations
- Where to go in the Progress documentation set for more information

1.1 Advantages of Planning Ahead

Software is being developed in many countries in many different languages, and you want to be sure that your product lends itself to translation so that you can compete in the worldwide marketplace. If your developers spend more effort designing an application with an international audience in mind, you will find that your internationalization efforts are much more efficient and less expensive in the end.

Translation Manager facilitates the process of translating Progress 4GL applications, but it does not eliminate the need for planning ahead. To successfully create a multi-lingual product, developers must design and create applications with translation in mind. For example, they must anticipate difficulties caused by the translation process. These might include text expansion difficulties, interface layout differences, behavioral differences, and different filtering requirements.

Often, software developers think about developing a user interface for the source language without considering the implications of translating the interface. When they complete an application, they send it to a translator. When the translation is finished, the developers realize they must go back and alter the source code and then return it to the translator. The practice is tedious and costly. It is usually easier to make changes during development than to retrofit changes after development is complete. By helping the developers who work on your project to plan ahead, you can help save time and money, as well as increase productivity.

If the application developers do not plan ahead and take translation into account as they develop source code, you are likely to encounter some of the following problems during translation:

- **Text expansion difficulties** — Translated text phrases must be abbreviated to fit in the interface or they appear truncated.

- **Interface layout differences** — Because of text expansion, the interface is not laid out properly in other languages. The interface layout looks good only in the source language. In the target languages, the layout shifts, and widgets, such as labels and messages, can overlap or text is clipped.

- **Program behavior differences** — If program behavior is dependent on text values, such as button labels, when you translate the labels, the behavior of the translated procedure can be defective.

- **Filter requirements** — Creating filters that extract only the strings that must be translated is time consuming and might require that developers use a coding technique that identifies strings to the filters. Translating unnecessary strings can be very costly.

1.2 4GL Considerations

This section describes the following programming techniques developers can use to ease translation:

- Defining text as untranslatable
- Defining alignment and space
- Defining accelerator keys
- Using default buttons

1.2.1 Defining Text as Untranslatable

When creating Progress applications, a developer can explicitly use the :U character-string literal after a text phrase to mark the text phrase as "Untranslatable." For example, the developer might not want to translate the name of your company or the name of a product, an internal key value, or a string used in determining program behavior. The developer can use the :U notation to ensure that the Translation Manager will not even display a given text phrase in the list of phrases to be translated.

When you extract text phrases in Translation manager, the Compiler automatically filters out and does not extract those phrases marked :U. You do not have to set this filter in the Translation Manager. You should encourage the developers to use the :U character string literal appropriately to make the task of selecting filters easier. See the *Progress AppBuilder Developer's Guide* for more information on filters.

1.2.2 Defining Alignment and Space

Progress allows developers to use alignment literals after a text phrase —:L, :C, :R, :T. Developers can also specify the amount of space (in characters) that should be reserved for the string's display. For example, specifying that side labels be right aligned can ease layout problems during translation. See the *Progress AppBuilder Developer's Guide* for more information on alignment of labels.

1.2.3 Defining Accelerator Keys

Developers should always use an ampersand (&) to define an accelerator key. As the project manager, you should decide and convey to your translators which of the following you want to do:

- Choose different accelerator keys based upon the target language. For example, if C is the accelerator key for the English word Close, use F for the French translation, Fermer, and use S for the German translation, Schliessen.
- Use the same accelerator key consistently for all languages to support portability. For example, always define C as the accelerator key for Close, no matter what the target language is.

1.2.4 Using Default Buttons

When creating Windows applications, developers often use the default buttons (OK and Cancel). This technique might not be the best for applications you want to translate. The default buttons display labels in the language of the operating system on which they are running. For example, if an application is running on Windows for French, the labels for the default buttons appear in French. Only if your application will always run in one language environment, should you use the default buttons.

If your application allows users to specify a language in which they will see the interface, you should avoid using default buttons, because you cannot programmatically specify the strings that the default buttons use as labels. Create your own buttons and mark the labels as translatable.

1.3 Screen Layout Considerations

A frequent problem that occurs when you translate an application is that the user-interface layout varies from one language to another due to text expansion. This section describes the following programming techniques you can discuss with your company's developers to help reduce the effects of text expansion on layout:

- Sizing and positioning widgets
- Planning for text expansion

- Creating adjustable text on rectangles
- Distinguishing between length and width of text phrases
- Creating message statements
- Adjusting justification on database fields

1.3.1 Explicitly Sizing and Positioning Widgets

If developers use explicit sizing and positioning when they define widgets, the application is easier to translate and the interface layout does not vary from one translation to another. Relative layouts, while easier for a developer to code from the Procedure Editor, have more problems compared to layouts made with the AppBuilder, which uses absolute sizes and coordinates.

The following code in `tmproc1.p` shows an example of code created in the Procedure Editor:

tmproc1.p

```
DEFINE VARIABLE a AS LOGICAL VIEW-AS TOGGLE-BOX
    LABEL "Replace if E&xists".
DEFINE VARIABLE b AS LOGICAL VIEW-AS TOGGLE-BOX
    LABEL "Check Syntax Now".

DEFINE FRAME myframe
  "Options" VIEW-AS TEXT AT ROW 1 COL 1 SKIP
  a SPACE(2)
  b
  WITH 1 DOWN.
```

This code example has the following translation problems:

- The widths of variables a and b are defined implicitly at compile time based upon properties such as the label. The label, “Replace if E&xists,” might have an implicit width of 18 characters which, depending on the font used, might not display the entire label text in the source language, let alone the target language.
- The width and height of the frame are not explicitly defined in the DEFINE FRAME statement. Typically, this means that the frame will not grow to accommodate translation space. Instead, the frame is clipped on the right side due to the implied widths of a and b.

- Progress positions the frame at row 1, column 1 because the frame's position is not explicitly defined.
- The SKIP and SPACE options might not properly align the free text, "Options," or the variables a and b.

To design for translatability, use the following code instead of the code in `tmproc1.p`:

tmproc2.p

```
DEFINE VARIABLE a AS LOGICAL VIEW-AS TOGGLE-BOX
    LABEL "Replace if E&xists" SIZE 30 BY 1.
DEFINE VARIABLE b AS LOGICAL VIEW-AS TOGGLE-BOX
    LABEL "Check Syntax Now" SIZE 30 BY 1.

DEFINE VARIABLE b AS LOGICAL VIEW-AS TOGGLE-BOX
    LABEL "Replace if E&xists":L25.

DEFINE FRAME myframe
  "Options" VIEW-AS TEXT SIZE 8 BY .85 AT ROW 1 Col 1
  a AT ROW 2 COL 1
  b AT ROW 3 COL 1
  WITH 1 DOWN SIZE 40 BY 10.
```

NOTE: Even when you use explicit sizing and positioning of widgets, you can still encounter text-expansion problems unless you also create margin space around widgets to accommodate expanded text phrases.

1.3.2 Planning for Text Expansion

When preparing your screen for translation, you should be aware that a translation can be much longer than the original text phrase. This is especially true when translating from English. Table 1–1 lists the standard formula for determining how much developers should plan for text expansion if English is the source language. This is the same table that the Translation Manager tool uses internally to calculate text expansion in the Compile dialog box. For example, if the length of the source text phrase is fourteen characters, you should plan to add space for eleven to fourteen additional characters, or a total of between twenty-five and twenty-eight characters. See Chapter 8, "Incorporating a Translated Kit into the Project," for information on how the Translation Manager handles text segment expansion.

NOTE: This table is meant as a guide. There is no one rule or formula that can accommodate all translations. Discussion with a translator about a particular language and its text expansion properties can help you develop an application that is usable in the target language.

Table 1–1: Standard Text Expansion Table for English Source Language

Length of English Text Phrase	Recommended Expansion for Target Phrase
Up to 10 characters	100–200%
11 to 20 characters	81–100%
21 to 30 characters	61–80%
31 to 50 characters	41–60%
51 to 70 characters	31–40%
Over 70 characters	30%

Developers should oversize the widgets in the interface layout and provide adequate space around labels and messages to accommodate text expansion in translation. For example, they should make buttons, list items, and radio options large enough to fit translations, otherwise the translator will need to abbreviate the translation.

For example, if you use the following code, there will not be enough space for the Spanish translations “Avion,” “Camion,” and “Barato”:

```
DEFINE c AS CHAR VIEW-AS SELECTION-LIST
  LIST-ITEMS "air", "truck", "boat"
  SIZE 14 BY 2.
```

Depending on the target language, the font for text phrases in a user interface can increase or decrease in size from the source language. Therefore, make sure the application developer leaves plenty of space around text and between objects so that when you translate the application, the user interface still appears in order. Often a developer creates an interface that looks perfect in the source language-with exact spacing. However, when you translate the text phrases to a language that uses a larger font, the text phrases in the user interface look crowded, overlap, or truncate.

The developer should also be aware of text expansion when sizing and positioning objects in the interface. A crowded interface in the source language could result in overlapping objects and text in the target language. Using the Text Expansion table will not always result in a perfect translated interface, and the translator will need to do some cleanup to achieve optimal results. However, properly planning for text expansion makes the translator's job smoother, results in a translated application that is clear and usable, and saves time that would be spent retrofitting the application.

1.3.3 Creating Adjustable Text on Rectangles

Developers often overlay text over rectangles in graphical interfaces, especially for three-dimensional applications or for radio sets where the rectangle creates a border for the radio options. If the developers use an absolute boundary to eliminate space to the right of the last letter of text for the source language, when you translate the application, the target text phrase might be truncated or have too much extra space. To avoid this problem, developers should use ru--time adjustments as illustrated in the following code example. In this example, the FONT–TABLE object allows the width of the object to grow or shrink as needed. This requires more coding but provides a uniform and predictable interface for translation, as follows:

```
DEFINE VARIABLE ThisLabel AS CHAR VIEW-AS TEXT SIZE 10 BY 1.
DEFINE RECTANAGLE Rect1 SIZE 1 BY 1 FONT 4.
DEFINE FRAME x
    Rect1 IN ROW 1.5 COL 1.5
    ThisLAbel IN ROW 1.25 COL 2
WITH 1 DOWN FONT 4.

ThisLabel = "options".
ThisLabel:SCREEN-VALUE=ThisLabel.
ThisLabel:WIDTH=font-table:get-text-width-chars(ThisLabel,ThisLabel:font).
```

1.3.4 Length and Width of Text Phrases

When you plan to translate an application, the developer should keep in mind not only the text-expansion factor of the target language, but also the font in which the target language will be displayed.

When you translate an application, you affect both the length and the width of the text phrases in the application. Translation Manager stores translations in terms of their length in bytes, but displays them in terms of their font's width. For example, if a button has the label "Hello," Translation Manager stores the label in a text segment that is five bytes long and uses 33 pixels to display "Hello" in Sans Serif 8-point font.

1.3.5 Creating Message Statements

You should be aware that some message statements might be difficult to translate. For example, message statements that contain compound phrases that follow the noun-verb-object rules of English do not translate well into languages that follow different grammatical rules. Sometimes developers create message statements with embedded SKIP statements to align the message statement evenly on the screen. While this hard-coded alignment might look attractive in the source language, it might look awkward when you translate the message into another language.

For example, the following code contains three text phrases that most likely will be translated out of context of one another:

- "The filename you entered,"
- ", is"
- "invalid under DOS."

```
DEFINE VARIABLE fname AS CHAR.
MESSAGE "The filename you entered," + fname + ", is"
    SKIP "invalid under DOS."
VIEW-AS ALERT-BOX WARNING.
```

When you translate these text phrases, the message will probably wrap awkwardly. To solve this problem, ask developers to avoid SKIP statements and to create message statements that do not split into text phrases that follow the grammar rules of a particular language. For example, in place of the above code, use one of the following code segments to display a similar message:

```
DEFINE VARIABLE fname AS CHAR.
MESSAGE fname SKIP(1) "is invalid under DOS. Please try again."
VIEW-AS ALERT-BOX WARNING.

DEFINE VARIABLE fname AS CHAR.
MESSAGE SUBSTITUTE (The filename you entered, &1, is invalid
    under DOS.", fname).
VIEW-AS ALERT-BOX WARNING.
```

1.3.6 Adjusting Database Field Justification

Many Progress applications use text labels from the Progress databases. The fields in the database might be left-justified. Left justification might work well on the user interface of the source language but might not work well on the user-interface after translation.

To avoid layout problems after translation, use the TRIM string attribute to trim the database fields. If you have multiple fields to trim, run the following Progress code to automatically set the TRIM attribute on all database labels:

```
FOR EACH _field WHERE NOT _field._field-name BEGINS "_":U
  AND_field._label-SA = ?:
    _field._label-SA = ""T:U.
```

1.4 Where to Go for More Information

Table 1–2 lists other locations in the Progress documentation set where you can learn more about creating applications for international audiences.

Table 1–2: Documentation Cross Reference

For More Information About . . .	See . . .
Using Progress's support for Internationalization, character sets, and code pages	*Progress Internationalization Guide*
Using character-string literals, such as :U and :T	*Progress Language Reference*
SUBSTITUTE Function	The Compile statement section of the *Progress Language Reference*
STRING–XREF files	The Compile statement section of the *Progress Language Reference*

2

Overview

The project manager must oversee the translation process and work to ensure that engineers develop a translatable application. The project manager also must ensure that the translator has the tools necessary to do the translation. This chapter describes the following topics:

- Overview of the Progress translation process
- Tasks for the project manager—the person who uses the Translation Manager tool
- Tasks for the translator—the person who uses the Visual Translator tool

2.1 Overview of the Progress Translation Process

Translation Manager facilitates the process of translating Progress 4GL applications, but it does not eliminate the need for planning ahead. To successfully create a multi-lingual product, developers must design and create applications with translation in mind. For more information on preparing a project for translation, be sure to read Chapter 1, "Preparing Your Application for Translation," and the *Progress Internationalization Guide*.

The Progress Translation Management system consists of two tools designed to work together—the Translation Manager and the Visual Translator. Together, these tools facilitate and manage the entire process of translating the text phrases in a Progress application into one or more languages. This book describes how to use the Translation Manager tool. For information on how to use the Visual Translator, see the *Progress Visual Translator Guide*.

The translation project manager uses the Translation Manager tool to prepare code for translation and then sends a kit to the translator, who uses the Visual Translator tool to translate the text phrases. The translator then sends the translated kit back to the project manager, who uses Translation Manager to consolidate the kit and produce the final translated application.

As the *translation project manager*—the person in your company who oversees the entire process of translating your company's software applications—you are responsible for:

- Determining the source code to translate. Note that if you use third-party applications or components, such as an ActiveX control, you must verify that the application is localized. A *localized application* is an application that is customized for a specific region. Do not assume that you can translate the third-party software.
- Developing the application with the appearance of the translated application in mind. Note that you must plan for text expansion and differences in layout that result from translation.
- Working with developers to prepare the source code for translation. Note that there might be behavioral differences and varying filter requirements as a result of translation.
- Defining the types of text phrases to translate.
- Estimating the volume of translation to be done.
- Providing the necessary information (such as text phrases and glossaries) to the translators, in a form that is usable to them.
- Reviewing the translated text phrases received from the translators.
- Incorporating the translated text phrases into the application.
- Compiling the source code with the translated text phrases to create a multi-lingual product.

The Translation Manager tool helps you, the project manager, complete these tasks.

The *translator*—the person who translates words and phrases from one language into another—uses the Visual Translator to translate an application's text phrases. The Visual Translator presents the translator with views of the text phrases as they actually appear in an application so that the translator can ensure that the translation's meaning is appropriate for the context and so that the translator can fit the translation into space allowed by the application's interface.

Figure 2–1 illustrates the technical concepts involved in translating the text phrases in a Progress 4GL application.

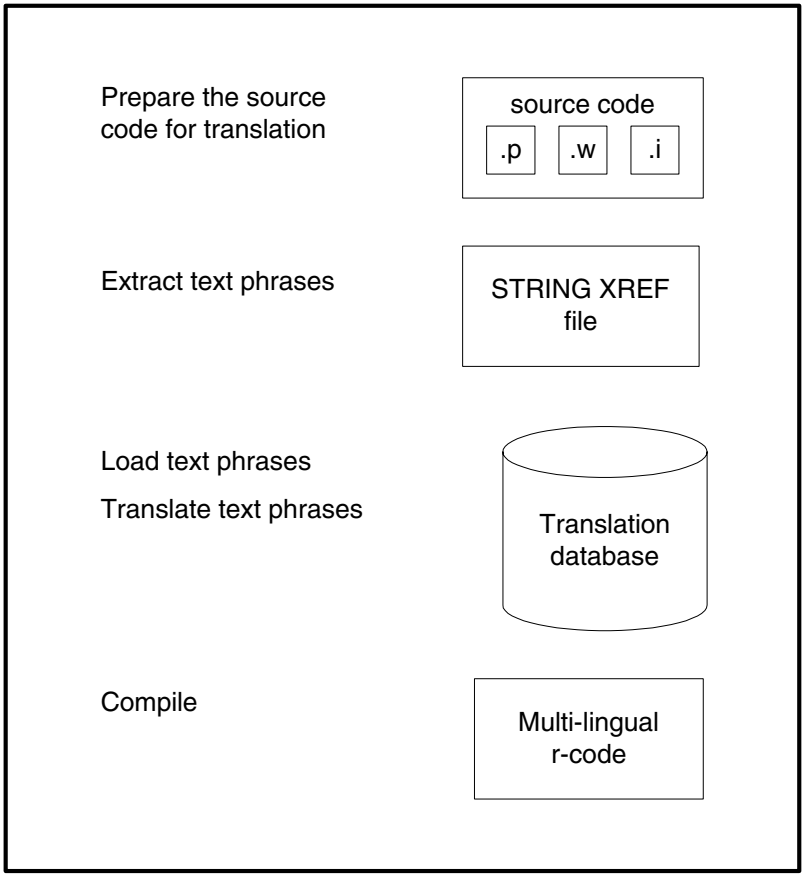


Figure 2–1: Overview of Translating a Progress Application

Figure 2–2 illustrates the steps the project manager follows with the Translation Manager tool and the interaction between the Translation Manager tool and the Visual Translator.

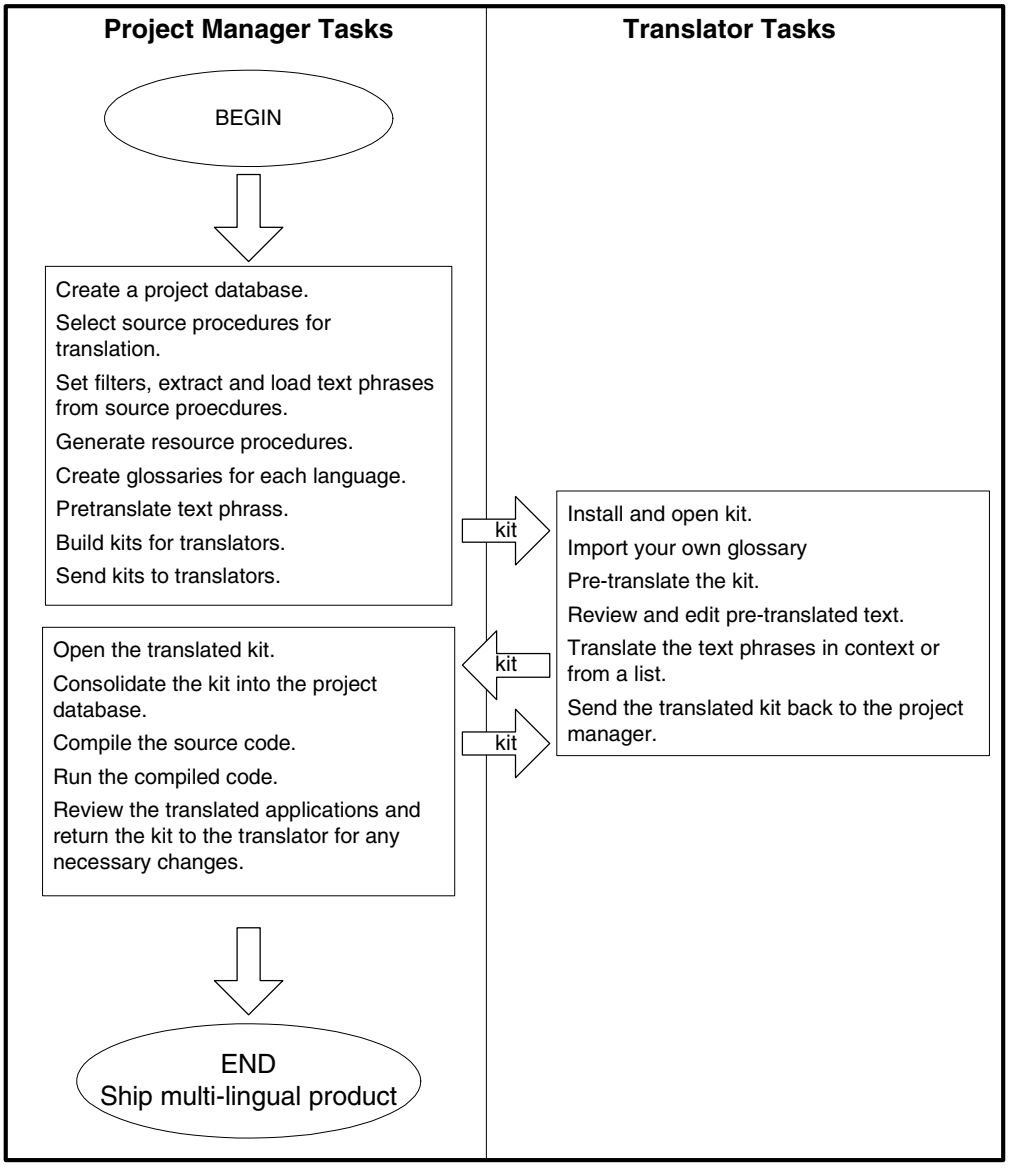


Figure 2–2: The Translation Manager Translation Process

2.2 Tasks for the Project Manager

The Progress Translation Manager tool helps you, as the translation project manager, organize and manage the process of translating the user interface of Progress 4GL applications. This section is an overview of the tasks you must complete using the Translation Manager tool.

The following outline describes the basic tasks involved in preparing an application for translation:

1. Use the Progress Translation Manager tool to create a *project database* that contains all the information needed to translate the application into one or more target languages.

2. Within the project database, create a list of *source procedures*—the Progress 4GL procedures in the application you want to translate.

3. Extract text phrases from the selected source procedures and place them in a STRING–XREF file. Specify filters to obtain only those text phrases you want to translate. Progress applications contain user-interface text (for example, label and ToolTip text) and text for 4GL logic. You usually want to translate only user-interface text in a source procedure but not string used in the 4GL logic. Then load the text phrases (through the filters to get only the text phrases you want to translate) into the project database.

4. Generate *resource procedures*—procedures that are shells of the original source procedures, containing the user-interface objects. Resource procedures do **not** contain any logic or database-access information. The translator will use these resource procedures (instead of the source procedures) to translate text phrases in a visual context.

5. Determine the target languages (the languages to which you want to translate) and create a *glossary*—a table in the project database that contains pairings of source and target phrases—for each target language. The Translation Manager tool includes Microsoft Language Glossaries that you can use to get started. You can also create glossaries specific to your application and to your customer's needs. For example, you can create a glossary of business or medical terms. You must specify a glossary for each language in the project.

6. Build a kit for each target language:

    a. Create a kit database that contains copies of the text phrases to translate and a glossary table from the project database.

    b. Generate a compressed (.zip) file for each kit. The compressed file contains a kit database, resource procedures, image files, and an environment (.ini) file that the resource procedures use.

7. Send the kits to the translators.

Figure 2–3 illustrates the contents of and relationship between the project database and kits.

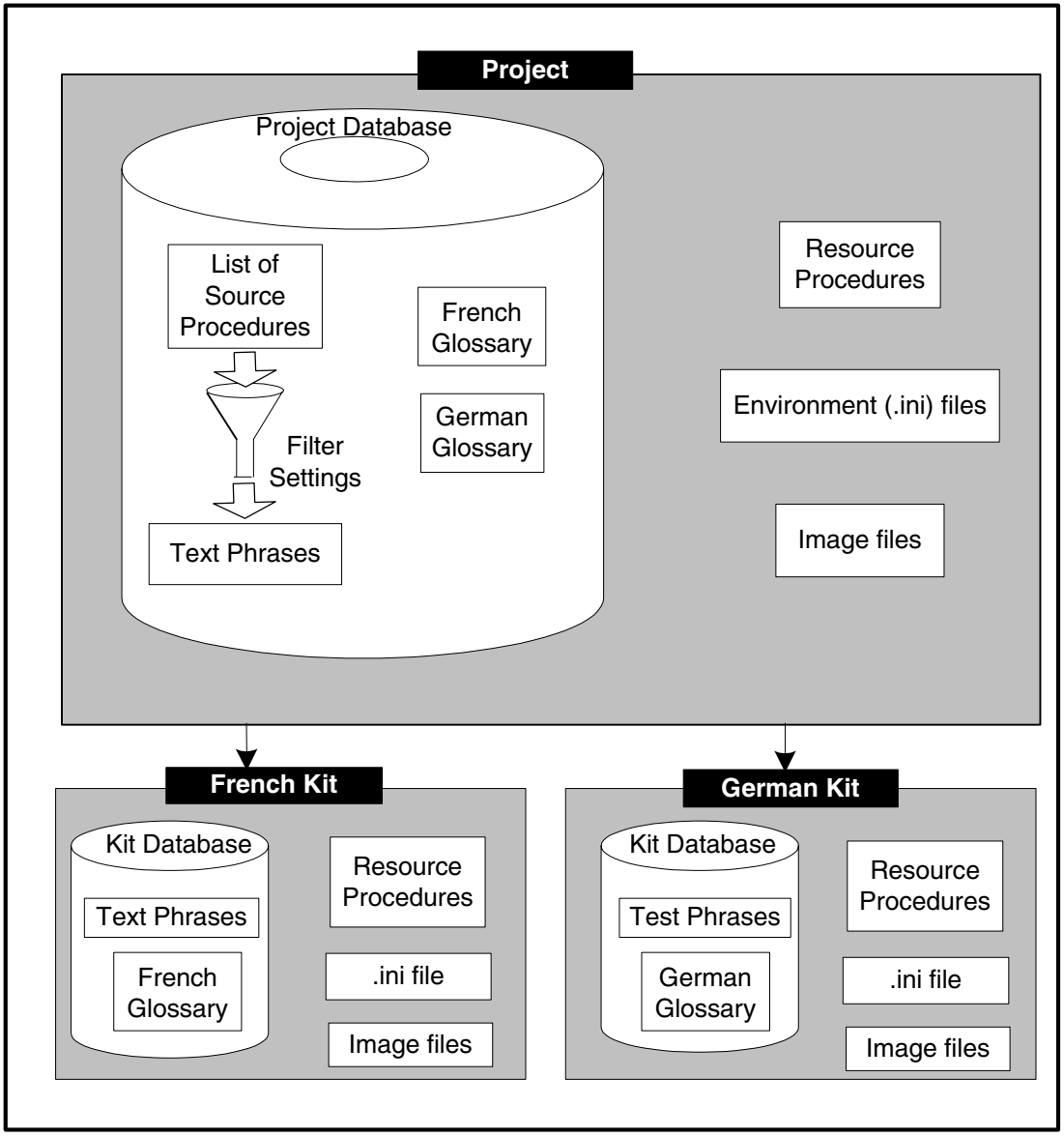


Figure 2–3:Projects and Kits

The translators translate the text phrases using the Progress Visual Translator tool and, when finished, return the kit database to you. See the “Tasks for the Translator” section for an overview of the translator’s tasks.

The following outline describes the basic tasks involved after the translator returns the kit database that has been translated using the Visual Translator tool:

1. Open (unzip) the returned kit database.
2. Merge the translations from the kit database into the project translation database. You can also merge the custom glossary entries (those entries added by the translator) with the original default glossary.
3. Specify how much you want to expand the space on the interface for the translated text phrases (optional) and compile the source code.
4. Run the source code to determine whether you need to discuss user-interface changes with the developers or return the kit to the translator to adjust the translations.
5. After making any necessary adjustments, recompile and ship the translated product. You might have a multi-lingual product or you might have multiple language versions of the same product.

2.3 Tasks for the Translator

The Visual Translator tool lets the translator translate user-interface text phrases in the context of the user interface or from a list.

The following outline describes the basic tasks involved in using the Progress Visual Translator to translate the text phrases in user-interface procedures:

1. The translator installs (unzips) and opens the kit you send.
2. The translator selects each procedure and translates text phrases in the context of the user interface or from a list.

   The translator can translate the application screen by screen. The Visual Translator presents each screen so that a translator can see the space available for each text phrase and can choose a translation that communicates the precise meaning of the original text as well as fits the available space.

   The Visual Translator also allows the translator to translate by reviewing lists of text phrases used in the application. This technique is useful for translating text that is not part of the user interface, for example, text in a report. This technique is also useful for reviewing similar phrases used on different screens, thereby ensuring a consistent translation.

3. The translator can create, maintain, and reference a glossary for consistency of translation within the application and for consistency across translations of documentation, online help, and marketing literature. A glossary can also help enforce consistency with industry standards for terminology.

4. After translating all the procedures, the translator saves the kit into a zip file and returns it to the project manager.

You review the translations and return them to the translator for modification if there are any changes to make.

For details on how to use the Visual Translator tool, see the *Progress Visual Translator Guide*

3

Getting Started

This chapter describes how to set up your system environment to use the Translation Manager tool. It includes the following topics:

- Upgrading from Version 8 to Version 9
- Organizing a directory structure for the source code
- Setting the PROPATH
- Creating and Storing Environment (.ini) Files
- Starting the Translation Manager
- Connecting to the application databases

3.1 Upgrading from Version 8 to Version 9

If you are upgrading your Progress Version 8 Translation Manager projects and database kits to Progress Version 9, follow these steps:

1 ♦ Back up your Version 8 project database by issuing the following command:

```
probkup db-name
```

2 ♦ Convert the single-volume database (Version 8) to a multi-volume database (Version 9) by using the following command:

```
prostrct convert db-name
```

3 ♦ Truncate the .bi file by using the following command:

```
proutil db-name -C truncate bi
```

4 ♦ Use Progress Version 9 to convert the database by using the following command:

```
proutil db-name -C conv89
```

You can now access the Version 9 project and kit databases.

3.2 Organizing a Directory Structure for the Source Code

You must make sure that the source code you want to translate (that is, all .p, .w, and .i files) is located in a directory tree with a single root directory. You must have read access to all the source code files.

You must also create the following directories:

- **Project directory** — To contain most of the information for your translation project. Progress Software recommends that you create this directory as a subdirectory of your current working directory and that you create multiple directories for multiple projects.
- **Translated r-code directory** — To store the r-code you create when you compile the application with the translated text phrases.

Figure 3–1 illustrates the outline of the directory structure you should create before you start the Translation Manager. You can have multiple project directories, as well as multiple directories for organizing your code.

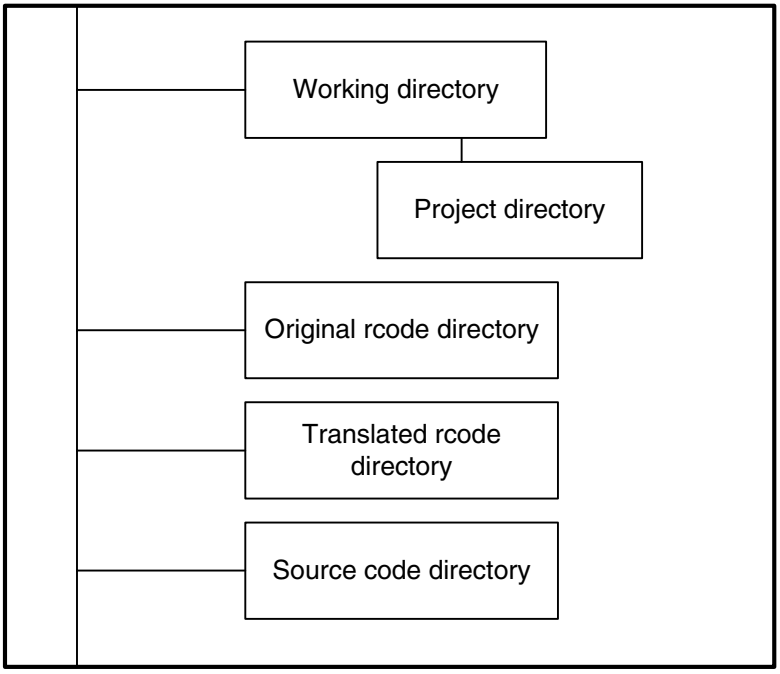


Figure 3–1: Organization of Ideal Directory Structure

3.3 Setting the PROPATH

The PROPATH is a Progress system environment variable that supplies a list of directories that Progress searches to find procedures and application files. For Windows environments, you specify the PROPATH environment variable in the registry or in the [Startup] section of the environment (.ini) file. The Translation Manager tool reads its configuration information from the registry. The Visual Translator tool relies on the .ini file that you send to your translators along with the kit.

Before you start the Progress Translation Manager tool, set PROPATH in the registry to the same setting you would use if you were developing the source code. You must add the full pathname of the source-code root directory to the PROPATH environment variable. One of the Translation Manager's functions is to compile translated source code, so you must re-create the directory structure that developers used when they wrote the original application. The PROPATH variable tells Progress which files to include when compiling and running an application. The PROPATH setting is stored within compiled r-code and is part of an

application's execution plan. Changing the location of r-code with relation to the directory structure in PROPATH prevents Progress from successfully running an application.

The PROPATH should contain the following directories in this order. Specifying a different order might cause source code to be compiled again:

1. The current working directory (the directory from which you will start Progress)
2. `$DLC/gui`
3. `$DLC`
4. The root directory that will contain the translated r-code
5. The root directory that contains the untranslated r-code for the source language
6. The source code root directory (.p, .w, and .i code)

NOTE: The root directory selection you make depends on whether you want source subroutines or target subroutines at run time.

Specifying these directories allows the Translation Manager to compile and run the translated code. For more information about the PROPATH environment variable file, see the *Progress Installation and Configuration Guide Version 9 for UNIX* or the *Progress Installation and Configuration Guide Version 9 for Windows* and the *Progress Client Deployment Guide*. When you deploy a translated application, you deploy only the translated r-code; the application does not require the original source code or the untranslated r-code to run. For information on deploying r-code, see the *Progress Programming Handbook* and the *Progress Internationalization Guide*.

3.4 Creating and Storing Environment (.ini) Files

Progress supports the use of the Windows registry or an existing environment (.ini) file to store configuration data. The registry is the default. You must decide which of the following is the most appropriate for you and your translators and specify one or the other:

- Use an existing environment (.ini) file if you already have specific .ini files for specific languages. For example, you might have a `german.ini` for the German translator and a `russian.ini` for the Russian translator. You can send these custom .ini files to the appropriate translators.
- Use the Windows registry. You specify that the Windows registry creates a .ini file for Visual Translator that you send with the kit to your translators.

For information on specifying one of these methods, seeChapter 5, "Preparing Data for Translation," for more information. For more information on .ini files, see the *Progress Installation and Configuration Guide Version 9 for Windows* and the *Progress Internationalization Guide*.

3.5 Starting the Translation Manager

The Translation Manager runs in single-user mode only. Use one of the following techniques to start the Translation Manager tool:

- From the ADE Desktop, choose the Translation Manager icon:



- From another ADE tool, such as the Procedure Editor, choose Tools→ Translation Manager.

Figure 3–2 shows the Translation Manager main window.

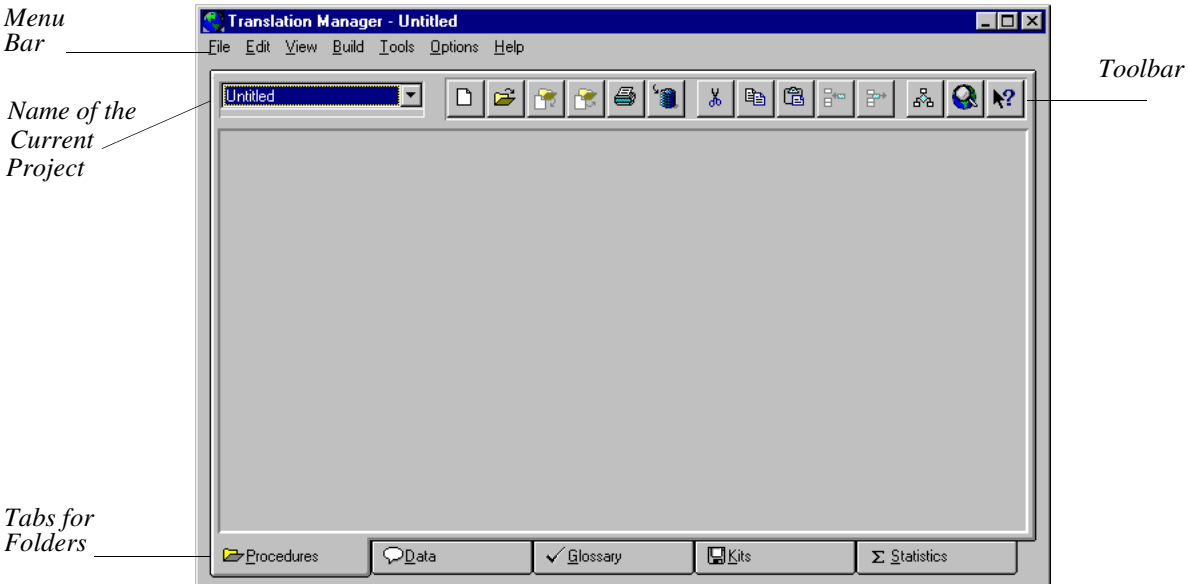


Figure 3–2: Translation Manager Main Window

See Appendix A, "Translation Manager Interface Reference," for information on the tab folders, menu options, and toolbar buttons of the Translation Manager tool.

3.6 Connecting to the Application Databases

The Translation Manager tool must compile the source procedures when it prepares translation data. Therefore, you must have read access to the databases that the source procedures use, and you must connect to those databases.

You can connect to the application databases on the command line when you start Progress from the Data Dictionary or Data Administration tools, or from the Translation Manager tool.

Follow these steps to connect to databases from the Translation Manager tool:

1 ♦ Choose the Database Connect button or choose File→ Database Connections. The Database Connections dialog box appears.

2 ♦ Choose Connect. The Connect Database dialog box appears.

3 ♦ Type the name of the database or choose Browse to select a database, then choose OK. The Translation Manager adds the database to the list in the Database Connections dialog box.

4 ♦ Repeat Steps 2 and 3 until you have connected all the necessary databases, then choose Close to return to the Translation Manager main window.

4

Managing Projects

This chapter describes how to manage projects. It includes the following topics:

- What is a project?
- Determining the number of projects to create for an application
- Creating a project
- Opening a project
- Selecting the current project
- Closing a project
- Selecting source procedures

4.1 What Is a Project?

A *project* is the main unit of information for the Translation Manager tool. It consists of:

- A Progress database (stored in the project directory) that contains:
  - A list of the source procedures you want to translate
  - Filter settings
  - Text phrases to translate (and eventually the translations of the text phrases)
  - A glossary for each target language
  - A list of the kits in the project
- Resource procedures (stored in the project directory)
- An environment (.ini) file (used to specify font settings)
- Image files (used with the resource procedures)

Each project has only one source language. The *source language* is the language (not necessarily English) in which the text phrases in the source procedures were written. A *target language* is the language to which you want to translate.

You can translate the source text phrases into many target languages. The target languages do not have to be national languages. You could define a language as Technical (contains software terminology), colloquial (contains informal expressions), or industry specific (contains manufacturing terminology). That way you can have translation variations of your product such that one version uses terminology appropriate for a specific industry, another uses more informal language that better suits ends users, and a third version uses vocabulary that meets the needs of a more technical audience. An application for small businesses with inexperienced computer users might require more general terms than the same application running in the Systems Operations department of a large multinational department. For example, an application for vehicle rental might have three variations—one that uses the term “vehicle” and allows you to lease both cars and trucks, one that allows you to lease cars, and one that allows you to lease trucks where “vehicle” is replaced with “car” or “truck,” as appropriate.

4.2 Determining the Number of Projects to Create for an Application

To determine the number of projects to create when you translate your application, you must understand the concepts of character sets, code pages, and language families.

A *character set* is the grouping of alphabetic, numeric, and graphic characters that are related to one another as defined by a standards organization, operating system vendor, or hardware manufacturer. Character sets also include communication and printer control codes—nonprinting characters like backspace, tab, and keyboard characters. For example, the `iso8859-1` code page is widespread in Western Europe, North America, and Latin America. Code pages often share characters.

Translation Manager supports three distinctly different types of character sets:

- *Single-byte character sets* contain characters that can be encoded into one-byte numeric code. Progress supports the following categories of single-byte character sets:
  - Western European/North American/Latin American, Australia, New Zealand
  - Eastern European
  - Russian
  - Hebrew
  - Arabic
  - Thai
- *Double-byte character sets* contain characters that must be encoded into two-byte numeric code. Progress supports double-byte character sets for the following languages:
  - Japanese
  - Traditional Chinese
  - Simplified Chinese
  - Korean
- *Multi-byte character sets* contain characters that are encoded into two or more bytes. Progress supports the multi-byte Unicode encoding standard, UTF–8. Unicode supports all languages currently used in commerce worldwide in a single character set.

A *code page* is a table that maps a numeric code to each character in a set of related characters. The computer uses these numeric codes to interpret and process textual data.

For more information on code pages and character sets, see the *Progress Internationalization Guide*.

You have various options when deciding how many projects to create for an application. You can choose to have all language versions supported by a single project. This allows you to create r-code with all languages. However, you can also work with multiple projects where each project supports a subset of all the languages being translated. Before deciding on the number of projects, take the following into consideration:

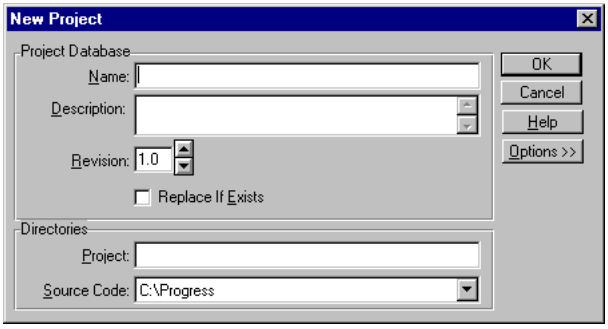
- Decide whether you require a single multilingual version or individual versions of the application where each version supports one or a few languages.
- The large size of a single multilingual version might require too much disk space or slow down the load process.
- If you have a large single-language application (1500 or more files), it might work better if you split it into multiple small projects. The Translation Manager processes smaller projects more quickly and you can divide the work among several translators. Choose logical units when dividing projects, such as discrete modules in an application. You must ensure that individual translators synchronize their glossaries regularly. An up-to-date glossary is essential for leveraging translated material.
- Decide if you will have the entire project translated simultaneously or if you will have the translations done in different phases of the application development cycle. If you are sending parts of the translation out at different times, it might be an advantage to have different projects coinciding with the different development phases.
- Since most systems cannot display characters from all parts of the world simultaneously, it will be impossible to view translations into languages that are not in the current character-set language family. You can avoid this by using different projects for languages that share the same character set. For example, you can handle Western European languages in one project, Eastern European languages in another, and an Asian language, such as Japanese, in yet another.
- If you are translating into more than one Asian language that requires double-byte character sets, each language (Simplified Chinese, Traditional Chinese, Korean, and Japanese) should have its own project.

4.3 Creating a Project

Follow these steps to create a new project:

NOTE: Before you create a project, make sure the root directory of the source code you want to translate is in your PROPATH. See Chapter 3, "Getting Started," for information about PROPATH.

1 ♦ Choose the New button or choose File→ New. The New Project dialog box appears:



2 ♦ Type the name of the project database. You can use a maximum of eleven characters. (Do **not** use blank spaces.)

NOTE: Progress Software recommends that you use the convention of starting all the names of your projects with the characters "p–." This naming convention allows you to quickly distinguish a project database from a kit database or an application.

3 ♦ Type a description of the project.

4 ♦ You can specify a revision number. The Translation Manager tool does not use this information; it is for your records only.

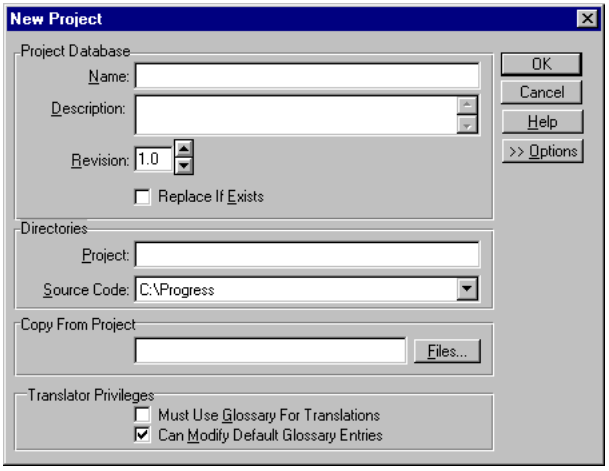
5 ♦ If the new project database has the same name as an existing database, and you want to overwrite the old database, activate the Replace If Exists toggle box.

6 ♦ Enter the name of the directory where you want to save the project database in the Project fill-in field. This directory will also contain the kit databases and the resource files. By default, the Translation Manager uses your current working directory.

7 ♦ Select the root directory that contains all the source code you want to translate.

NOTE: The Source Directory combo box lists only the directories in your PROPATH.

8 ♦ Choose Options to expand the New Project dialog box:



9 ♦ Specify whether to create a copy of an existing project database. If you have multiple projects you can use the same database for more than one project.

10 ♦ Select the privileges you want to give to all the translators for the project:

- **Must Use Glossary For Translations** — When you activate this option, the translators can use only the default glossary you supply with the translation kits. They cannot import their own glossary into the kit nor can they use their own words when the default glossary contains a translation. However, if the glossary does not contain a target phrase for the selected source phrase, translators can enter their own target phrase.
- **Can Modify Default Glossary Entries** — When you activate this option, the translators can override the pairings suggested by the default glossary you provide and modify existing translations. They can also import their own glossary into the kit.

All the language kits within a single project have the translator privileges you specify in this dialog box. If you want different translators to have different privileges, you must create several copies of the project.

You can only specify translator privileges when you create a new project. The only way to change translator privileges for an existing project is to create a new project with a new name, copy the files from the existing project, and specify new privileges.

11 ♦ Choose OK to create the project database in the working directory. The Translation Manager displays the Procedures tab folder.

See the "Selecting Source Procedures" section for a description of how to create a list of source procedures for the project.

4.4 Opening a Project

You can have more than one project open simultaneously. Follow these steps to open an existing project:

1 ♦ Choose the Open button or choose File→ Open. The Open Project dialog box appears.

NOTE: You can open only a project database in the Translation Manager tool. For example, you cannot use these steps to open the Sports database. However, you can connect to the Sports database using the File→ Database Connections option.

2 ♦ Select the location and name of the project database (.db) file you want to open, then choose OK.

4.5 Selecting the Current Project

The Translation Manager displays the name of the current project in the combo box located in the upper left corner of the Translation Manager main window:



If you have more than one project open, follow these steps to change the current project:

1 ♦ Choose the down arrow button. A list of the names of open projects appears.

2 ♦ Select the name of the project you want to display. The Translation Manager changes the current database to the project you select. It displays the name of the selected project in the combo box and the contents of the project in the tab folders.

4.6 Closing a Project

To close a project, choose File→ Close. The Translation Manager disconnects the current project database and displays the next project or, if no other project is open, an empty tab folder.

NOTE: The Translation Manager automatically saves changes as you make them. When you close a project, the Translation Manager retains these changes and automatically disconnects any connected project or kit databases. It does not disconnect any application (that is, nonkit or nonproject) databases.

4.7 Selecting Source Procedures

This section describes how to specify source procedures—the procedures you want to translate—for a project. When you select source procedures, the Translation Manager creates a list of the procedures to translate; it does not copy them into the project database.

You can use the following techniques to select source procedure files:

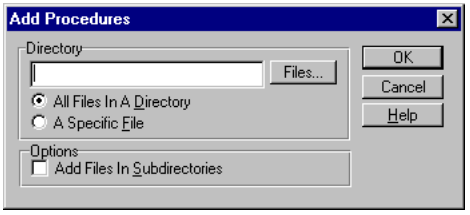
- To add a large number of files to the list, add the entire directory that contains the procedure files, then remove from the list those files that you do not want to translate.
- To add a small number of files to the list, add the files one at a time. Your project can consist of a single file.

NOTE: If the developers use SmartObjects in the application and you want to translate the text phrases in the SmartObjects, you must include the source code for those SmartObjects in the project's list of procedures. The source code for Progress SmartObjects is located in `DLC/src/adm2`. (For Version 9-compatible SmartObjects, the source code is in `DLC/src/adm/objects`.)

4.7.1 Adding Files to the Procedures List

Follow these steps to add files to the list of procedures you want translated:

1 ♦ From the Procedures tab folder, choose Add. The Translation Manager displays the Add Procedures dialog box:

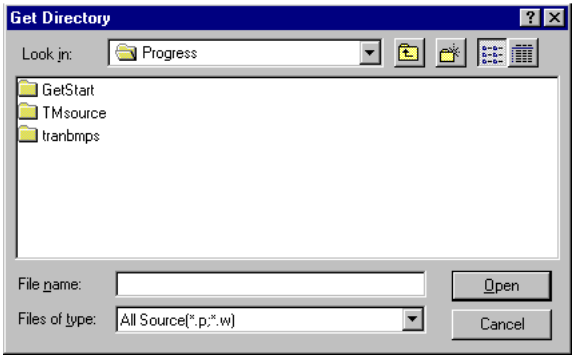


2 ♦ Specify whether to add all the files in a directory or a single file.

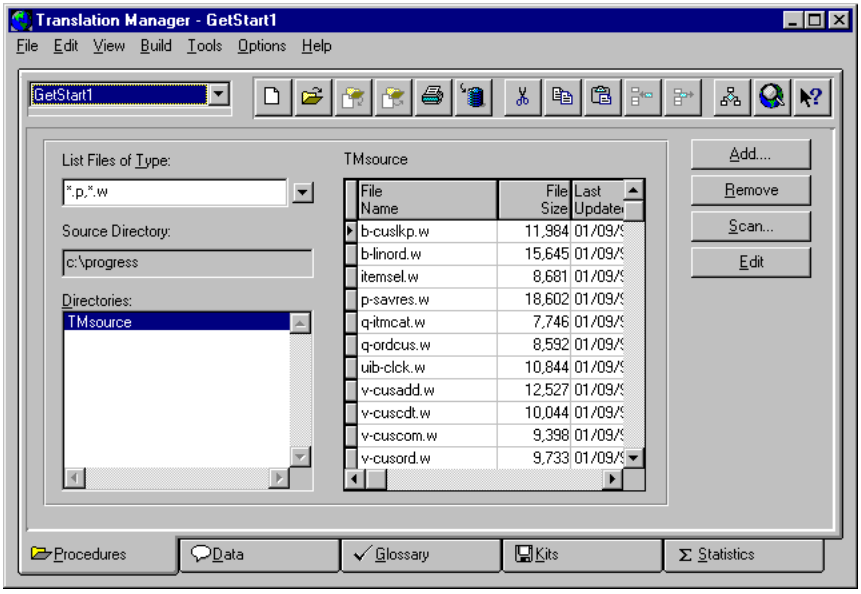
3 ♦ If you want to add all the files from the subdirectories of the directory you specify, activate the Add Files in Subdirectories toggle box.

4 ♦ Enter the name of the directory or file you want to add, then choose OK.

NOTE: If you activate the All Files in a Directory radio button and choose the Files button to select a directory, Progress displays the Get Directory dialog box with a caret (^) in the File Name fill-in field. When you select a directory and close the dialog box, the Translation Manager adds all the files in the directory you selected to the list of procedures in the project database:



The directories and files display as follows on the Procedures tab folder:

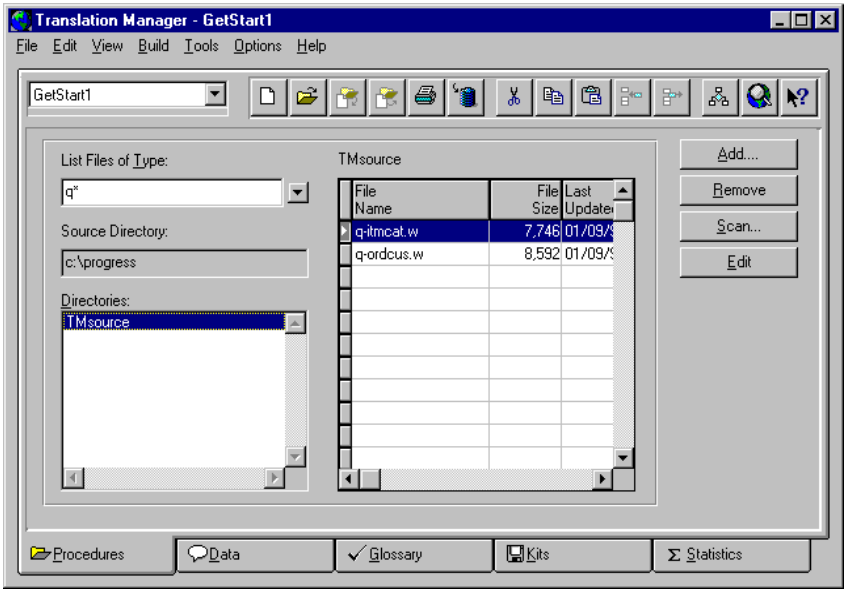


By default, the Translation Manager lists only the procedure (.p) and window (.w) files; however, you can use a wild card (*) expression in the List Files of Type field to select specific types of files (or specific filenames) for translation.

Follow these steps to use a wildcard expression to select specific files for translation:

1 ♦ Enter a wild card expression in the List Files of Type field. For example, enter q* to add query files. Press the Add button. The Add Procedures dialog box appears.

2 ♦ Press OK. Only the procedures you specify appear:



NOTE: You can only use a wild card expression in the List Files of Type field of the Procedures tab folder. You cannot use a wild card expression in the Files of Type field of the Get Directory dialog box.

If you are an experienced user of Translation Manager and you want to speed up the process of adding files to a project, you can suppress messages and statistics that are reported during the process. To suppress messages, choose Options→ Preferences and select the degree of suppression you want. You can suppress messages during extract, resource procedure generation, and when adding procedures.

See the "Removing Files from the Procedures List" section for information on how to remove individual files that you do not want to translate.

If you want to translate all of the files listed without removing any, see Chapter 5, "Preparing Data for Translation," for information on specifying the filters you want to use when you extract the text phrases from the procedures.

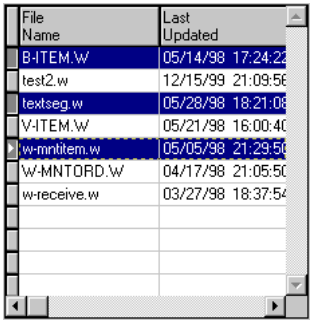
4.7.2 Removing Files from the Procedures List

If there are files in the project that you do not want the translator to translate, you must remove the files.

Follow these steps to remove a file you do not want translated from the procedure file list:

1 ♦ From the Procedures tab folder, select the files that you want to remove from the list:

- To select a single file, choose the row marker for that file.
- To select multiple consecutive files, drag the cursor over the row markers for the files you want to delete.
- To select multiple nonconsecutive files, press and hold **CTRL** as you select or deselect individual files:



2 ♦ Choose Remove. The Translation Manager deletes the selected files from the list of source procedures in the Procedures tab folder.

NOTE: The Translation Manager does not delete the files. The files are only removed from the list. If you remove a file by mistake, just add the file to the list again.

4.7.3 Viewing and Editing a File in the Procedures List

From the Procedures and Data tab folders, you can peruse a source procedure to determine whether you want to translate it.

Follow these steps to view, edit, or run the 4GL code of a file in your source-procedure list:

1 ♦ Select the row marker for the file you want to view, then choose Edit. The Translation Manager displays the procedure code in a Procedures window.

2 ♦ View, edit, or run the code.

3 ♦ Choose File→ Close Window to return to the Translation Manager tool.

You must verify that you have permission to edit code before you perform these steps.

5

Preparing Data for Translation

This chapter describes how to prepare text for the translators. It discusses the following topics:

- Extracting text phrases from the source procedures
- Specifying filters through which you can load text phrases
- Loading the text phrases (through the filters) into the project database
- Creating resource procedures that the translators will use to translate user-interface objects in context
- Working on subsequent iterations of a project
- Preparing text from a character application
- Using third-party tools

5.1 Extracting Text Phrases from Source Procedures

When you extract text phrases from source procedures, the Translation Manager compiles the source procedures and copies the text phrases into a temporary (STRING-XREF) file. It does not remove the text phrases from the procedures. Figure 5–1 illustrates the process the Translation Manager follows when it extracts text phrases.

NOTE: Because the Translation Manager compiles the source procedures when it extracts text phrases, you must be connected to the databases that the source procedures use during the extract process.

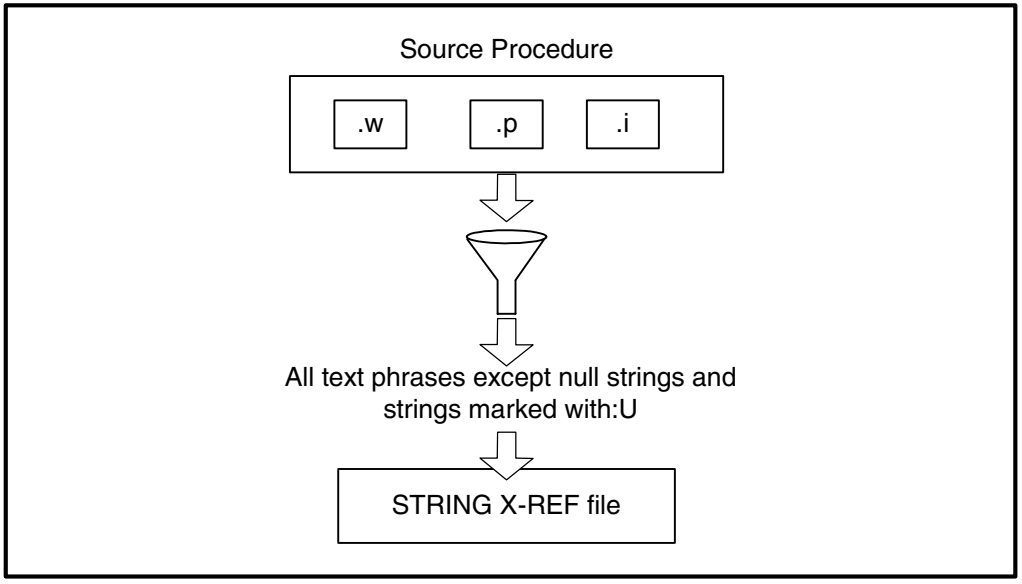


Figure 5–1: The Extract Process

By default, the STRING-XREF filename is the project name with an .xrf extension. The STRING-XREF file contains a list of translatable text phrases; it does not include null text strings (" ") or any text the developer marked as untranslatable (:U) in the source code; it does include initial values. The STRING-XREF file includes details such as the name of the source procedure, line number, item, and the 4GL statement that contains each text phrase. Figure 5–2 shows the contents of an example STRING-XREF file.

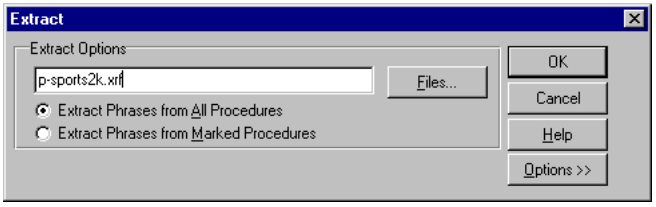
```
String Xref Version 4.0 adm\objects\p-updsav.w ibm850
158 ? "c:\apps\adm\objects\p-updsav.w should only be
RUN PERSISTENT." 61 NONE MESSAGE
101 ? "You must complete the open transaction
before exiting." 54 NONE MESSAGE
388 ? "YES" 3 NONE IF EXPR
591 ? "adm-" 4 NONE IF EXPR
779 ? "?" 1 NONE ASSIGN EXPR
790 ? "?" 1 NONE ASSIGN
934 ? "=" 1 NONE IF EXPR
936 ? "Invalid element in set-attribute call:" 38 NONE MESSAGE
936 ? "in" 2 NONE MESSAGE
225 ? "&Save" 5 NONE ASSIGN
244 ? "&Update" 7 NONE ASSIGN
258 ? "&Save" 5 NONE ASSIGN
308 ? "&Update" 7 NONE IF EXPR
311 ? "&Save" 5 NONE ASSIGN
317 ? "&Update" 7 NONE ASSIGN
431 ? "&Update" 7 NONE ASSIGN
626 "Btn-Save" "&Save" 5 TRIM DEF-BUTTON LABEL
626 "Btn-Reset" "&Reset" 6 TRIM DEF-BUTTON LABEL
626 "Btn-Add" "&Add" 4 TRIM DEF-BUTTON LABEL
626 "Btn-Copy" "&Copy" 5 TRIM DEF-BUTTON LABEL
626 "Btn-Delete" "&Delete" 7 TRIM DEF-BUTTON LABEL
626 "Btn-Cancel" "Ca&ncel" 7 TRIM DEF-BUTTON LABEL
```

Figure 5–2: Example STRING-XREF File

Follow these steps to extract text phrases from the source procedures:

NOTE: To extract text phrases or generate resource procedures, you must be connected to the application databases that the source procedures use. See Chapter 3, "Getting Started," for information on how to connect to databases from Translation Manager.

1 ♦ From the Data tab folder, choose Extract. The Extract dialog box appears:

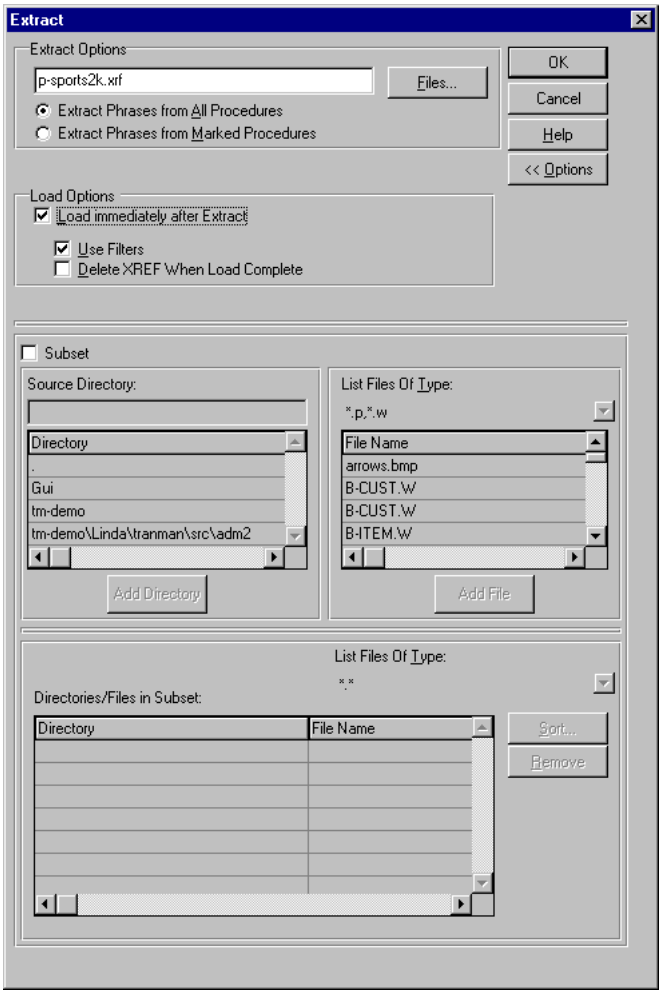


2 ♦ Type a name for the STRING-XREF file or accept the default.

3 ♦ Specify one of the following extract options:

- **Extract Phrases from All Procedures** — Extracts text phrases from all the procedures in the procedures list of the project. Activate this radio button the first time you extract text phrases for a project.
- **Extract Phrases from Marked Procedures** — Extracts text phrases from only those procedures that you mark when you use the Scan dialog box to update the procedures list for the project.

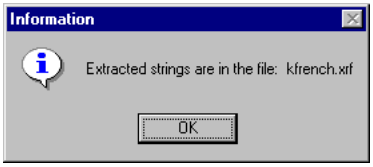
4 ♦ If you are an experienced Translation Manager user, you might want to extract the text phrases and load them immediately into the project database. Choose Options. The dialog box expands:



5 ♦ Check the Load immediately after Extract option. You can also specify to use filters and to have the STRING-XREF file deleted after Translation Manager loads the phrases into the project database. This option is useful if you have a proven set of filters and want to extract and load text phrases without tuning filters as an intermediary step.

6 ♦ Choose OK.

When the Translation Manager finishes extracting the text phrases, an alert box appears indicating the pathname of the STRING-XREF file that contains the extracted text phrases:



The STRING-XREF file is located, by default, in the current working directory. You might want to translate only some of the text phrases in the STRING-XREF file. The next section describes how to set filters so that you load only those text phrases you want to translate into the project database.

5.2 Specifying Filters

Usually, you will want to translate only a few of the text phrases in a source procedure; that is, only user-interface text, not 4GL logic text. The Translation Manager lets you set filters so that the Translation Manager loads into the project database only those text phrases that you want to translate.

Because it is easier to remove text strings than add text strings, you should include text strings that you are not sure about. You can remove them later.

Follow these steps to specify filters:

1 ♦ From the Data tab folder, choose Filters. The Filter Wizard window appears. The Filter Wizard consists of thirteen panels that lead you through the process of specifying filters.

The first ten panels let you specify *include filters* — filters that determine the 4GL text phrases you want to place in the project database for translation. The last three panels let you specify *exclude filters* for Progress keywords or specific text strings (for example, a company logo) that you do not want to translate. You can specify multiple filters—the number of filters you can specify is limited only by the size of the character string your system resources support. See the "Loading Text Phrases into the Project Database" section for details on how the Translation Manager uses these filters.

2 ♦ To accept the defaults and skip to the end of the Filter Wizard, choose Finish. The Filter Wizard displays a summary of the filters you defined.

If you would like to redefine your filters, use the Previous and Next buttons to navigate through the Filter Wizard and make changes. Then choose Finish to display an updated summary.

3 ♦ Choose Apply to set the filters.

The Translation Manager saves your filter settings in a table in the project database. You can redefine them by repeating Steps 1 through 3.

NOTE: The Filter Wizard assumes you are familiar with the Progress 4GL language.

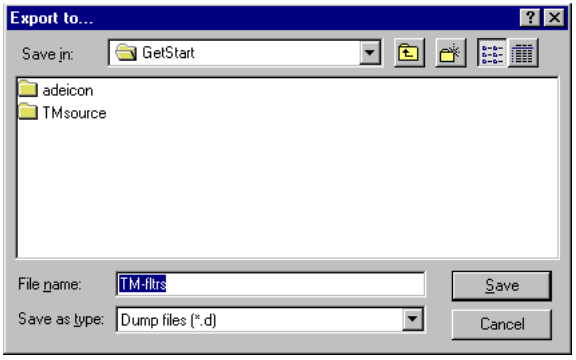
The following section describes how you can reuse your filters for multiple projects.

5.2.1 Exporting and Importing Filters

You can export a project filter to a text file so that you can use it in another project. When you export filters, the Translation Manager saves it as a table dump file (.d).

Follow these steps to export a filter:

1 ♦ Choose Tools→ TranMan Utilities→ Export Filters. The Export to... dialog box appears:



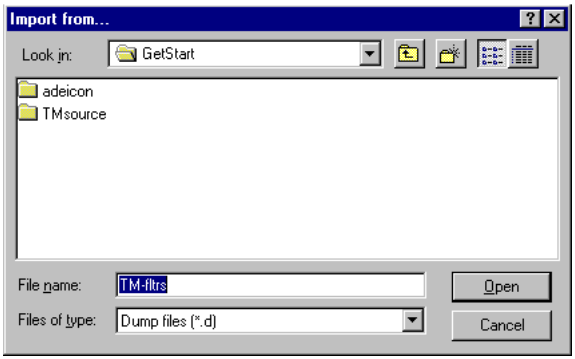
2 ♦ Choose the directory that contains the filters that you want to export.

3 ♦ Enter the filter filename (the default filename is TM-fltrs) with a table dump file (.d) extension, then choose the Save button.

You can customize your filters, then import the filters into multiple projects. The filter file you import must be a text file with the source and target phrases separated by commas or spaces.

Follow these steps to import new filters into a project:

1 ♦ Choose Tools→ TranMan Utilities→ Import Filters. The Import from... dialog box appears:



2 ♦ If you import new filters, you replace the old filters. Choose OK to verify the import of new filters.

3 ♦ Choose the directory that contains the filters that you want to import.

4 ♦ Enter the filter filename (the default filename is TM-fltrs) with a table dump file (.d) extension, then choose the Open button.

5 ♦ Enter the code page for the glossary.

The "Loading Text Phrases into the Project Database" section describes how to load the text phrases from the STRING-XREF file into the project database through the filters you just defined.

5.3 Loading Text Phrases into the Project Database

After you define filters, your next step is to load the text phrases into the project database.

When you load text, the Translation Manager passes the text phrases in the STRING-XREF file through the three types of filters you defined with the Filter Wizard: Progress 4GL, Progress Keywords, and Custom. The Translation Manager then loads the text phrases into the project database. Figure 5–3 illustrates the process the Translation Manager follows to load text phrases.

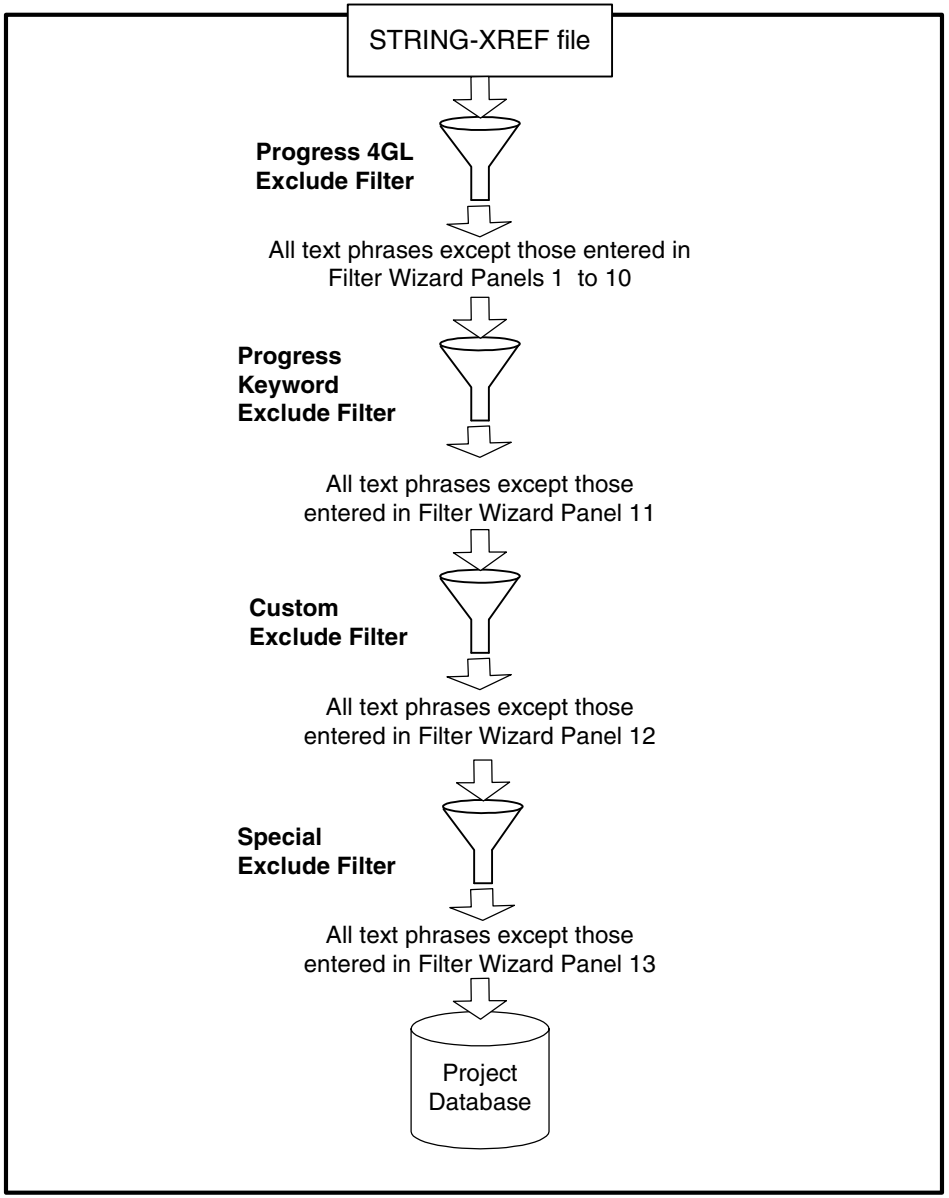
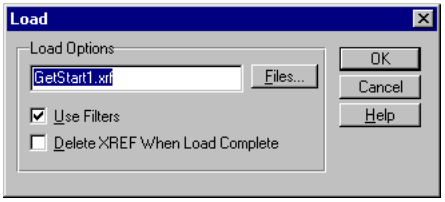


Figure 5–3: The Load Process

After you load the text phrases, you can view them in the Data tab folder. You can then redefine your filters and load the text phrases again. The Translation Manager compares the new load information to the text phrases that are already in the database and updates the database with the newest information.

Follow these steps to load the text phrases from the STRING-XREF file into the project database:

1 ♦ From the Data tab folder, choose Load. The Load dialog box appears:



2 ♦ Specify the following load options, then choose OK:

- **Use Filters** — Loads the text phrases from the STRING-XREF file through the filters you selected in the Filter Wizard.

  If you deactivate this option, the Translation Manager loads all the text phrases in the STRING-XREF file into the project database.

- **Delete XREF When Load Complete** — If you accept the default (deactivated), you can reuse the STRING-XREF file later to avoid re-extracting if you change the filter settings.

  If you activate this option, the Translation Manager deletes the temporary STRING-XREF file after it loads the text phrases into the project database. You will then need to re-extract the text phrases if you change the filter settings.

Progress loads the text phrases and displays a statistics alert box summarizing the number of text phrases it extracted and loaded into the project database. The alert box also indicates whether the Translation Manager deleted the STRING-XREF file:



Figure 5–4 shows the Data tab folder where you can view the text phrases Progress loaded into the project database. For example, if you accept the default filter settings and load the text phrases, the Translation Manager does not include combo-box list items. To include the list items for combo boxes, open the Filter Wizard to the fourth panel and activate the Combo-box

List-items toggle box. When you choose Load, the Translation Manager loads the text phrases through the newly specified filters and this time includes the list items for combo boxes.

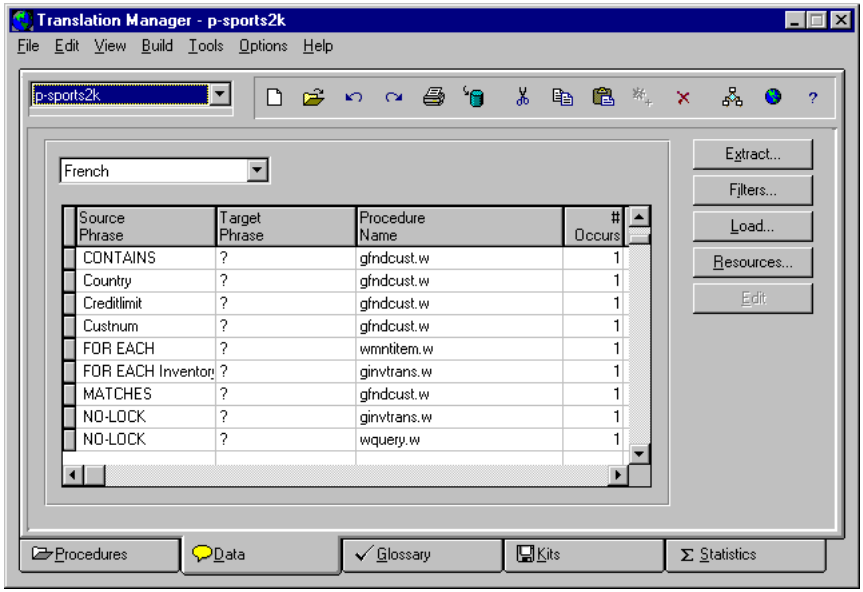


Figure 5–4: Data Tab Folder

Review the text phrases carefully and remove any text phrases you do not want the translator to translate. You can view any text phrase in a separate, larger dialog box by double-checking on any field of the Data tab folder. Figure 5–5 shows the Long String Translator dialog box.

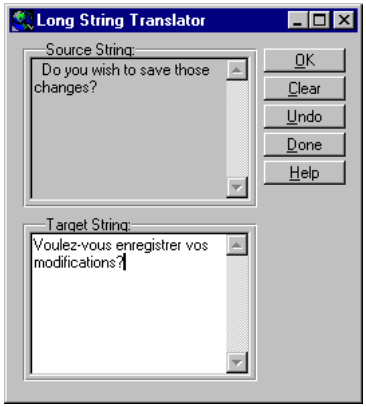


Figure 5–5: Long String Translator Dialog Box

The Long String Translator dialog box allows you to view and translate text phrases. If a text phrase is longer than 34 characters, you cannot see all of the text in the Translation tab folder.

Double-check on the text to display the Long String Translator dialog box. The source string displays in the Source String viewer. The target string displays in the Target String viewer. You cannot edit the source string, but you can edit the translation in the Target String viewer.

The next section describes how to create the resource procedures that translators use for in-context translation.

5.4 Generating Resource Procedures

You can create resource procedures if you want the translator to translate text in the visual context of the user interface. A *resource procedure* is the user-interface shell of an original source procedure. When you create a resource procedure, the Translation Manager removes all database-access and logic information so that the resource procedure does not function like a source procedure. For example, if the translator presses a delete button on a resource procedure interface, the Visual Translator tool does not delete anything. Instead, the Visual Translator displays the Properties window that allows the translator to translate "delete" into the target language.

NOTE: The Translation Manager creates the layout of the resource procedure files based upon the compile-time layout of the corresponding source procedure. The resource procedures layout does not reflect any layout modifications that occur when the source procedure is run.

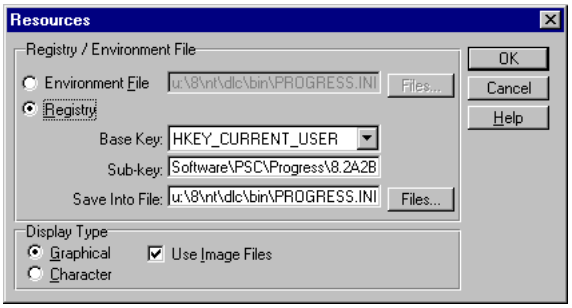
To create a resource procedure, the Translation Manager creates a copy of the source procedure and makes the following changes:

- Adds the code from any called include (.i) files to the main procedure. That is, the Translation Manager does not create a separate resource file for an include file. It copies the contents of the include file into every procedure that calls the include file.
- Maps DISPLAY statements to DEFINE FRAME statements.
- Changes internal text in FORM, DISPLAY, and DEFINE FRAME statements to fill-in field widgets with VIEW-AS TEXT properties.
- Removes all the 4GL logic. For example, the Translation Manager removes all IF . . . THEN statements.
- Removes all bindings to databases.
- Removes all triggers and internal procedures and replaces them with its own triggers and procedures.
- Replaces the file's .p or .w extension with a .rc extension.

Follow these steps to create resource procedures from the source procedures listed in the Procedures tab folder:

NOTE: Before you generate resource procedures you must connect to the application databases that the source procedures use.

1 ♦ From the Data tab folder, choose Resources. The Resources dialog box appears:



2 ♦ Choose either the Environment File radio-set button or the Registry radio-set button. This option determines how Translation Manager stores and how you maintain font and color settings for the target language. For example:

If you choose the Environment File radio-set button, enter the pathname of the environment (.ini) file that specifies the font and color settings for the target language or choose the Registry. For example, you might specify an environment file that is similar to the files included in *install-path*\dlc\prolang\.

NOTE: If you do not enter a filename or you enter the wrong filename, the translator can view resource procedures, but the Visual Translator might display the resource procedures with the wrong font and color mapping, thereby giving the wrong feedback to the translator. For example, if the Visual Translator cannot find the proper font file, it uses the default 10-point font, no matter what the font should be.

If you choose the Registry radio-set button (the default), enter the Windows registry basekey or subkey and the pathname for the environment (.ini) file. Instead of using a .ini file that already exists, the Windows registry creates a .ini file. If the basekey is not specified, Progress searches the registry.

Translation Manager sends an environment (.ini) file to the translator whether you specify an existing environment (.ini) file or you specify the registry to create a .ini file.

For more information on environment (.ini) files and the Windows registry, see Chapter 3, "Getting Started," and the *Progress Installation and Configuration Guide Version 9 for Windows*.

3 ♦ Specify whether you want Progress to display the resource procedures as character or graphical interfaces.

4 ♦ Activate the Use Image Files option if you want the Translation Manager to display any image files when it generates the resource procedures. The Translation Manager will also include the image files in the kits you send to the translators.

5 ♦ Choose OK.

The Translation Manager deletes any existing resource procedures and generates new resource procedures in the project directory. The Translation Manager displays an alert box stating how many resource procedures it attempted to process and how many it actually processed.

5.5 Working on Subsequent Iterations of a Project

You are likely to have to go through each project's translation process multiple times as new modules are added, new versions of the application are created, or bugs are fixed. The first time you go through the translation cycle, you must complete all of the following tasks:

- Create a new project.
- Add the application procedures.
- Extract the strings from procedures into a .xrf file.
- Filter and load the strings.
- Create Resource Procedures for procedures in the project.
- Create a glossary.
- Create a kit to send to the Translator.
- Consolidate the translated kit into the project.
- Compile the application into a multilingual executable.

In subsequent iterations of a project, you do not need to repeat all the steps above or to work with all the procedures. You can choose to work with one or more subsets of the procedures to complete the following processes:

- Choose to scan all the application directories and procedures or to scan a subset of them. You can mark procedures manually.
- Extract the strings from marked procedures into a string-xref file.
- Filter and load the strings from that string-xref file or a subset of the string-xref file.
- Create Resource Procedures for all procedures or a subset of the procedures in the project.
- Create a kit to send to the Translator with all the application procedures or a subset of the application procedures.

5.5.1 Procedure Subsets and Supersets

Each one of the processes listed above allows the use of procedure subsets, and each subset is derived from a different superset, depending on the process. Most of the time the superset contains all the procedures in the project (from now on referred to as project procedures).

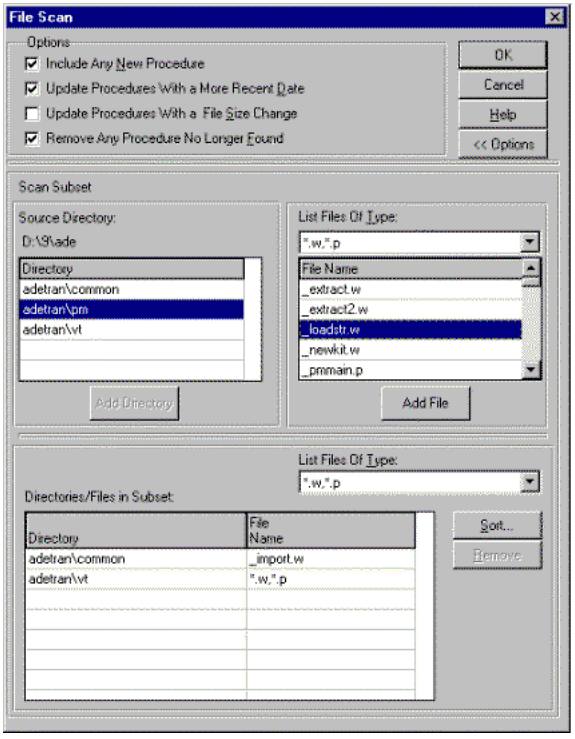
Each subset's superset is explained as follows:

- **Scanning directories/procedures (Procedure tab, Scan button)** — The superset is all the project procedures and all the procedures that are in the directories that are part of any of the project procedures.
- **Extracting strings from procedures (Data tab, Extract button)** — The superset is all the project procedures that are marked as needing extracting. Note that before Version 9.1, the only way to mark the project procedures for extracting was by the Scan process. The scan process now allows manual marking of procedures. The extract process creates an `.xrf` file.
- **Loading strings (Data tab, Load button)** — The superset is the procedures contained in a `.xrf` file that are a subset of the project procedures.
- **Creating resource procedures (Data tab, Resources button)** — The superset is the project procedures.
- **Creating a kit (Kit tab, Add button)** — The superset is the project procedures.

5.5.2 Using Procedure Subsets

Follow these steps to specify which project procedures to work on:

1 ♦ Choose the Options button. The dialog box expands by adding subset sections:



2 ♦ Click on the Subset box.

3 ♦ Select a directory and choose Add Directory to add a source directory.

4 ♦ Select a filename and choose Add File to add a file.

5 ♦ Select a directory from the Directories/Files in Subset area and choose Sort.

6 ♦ Select a file and choose Remove to remove that file from the list.

5.6 Preparing Text from a Character Application

You can use the Translation Manager to translate character applications that were developed for UNIX. Follow these steps to prepare the text for translation:

1 ♦ On the platform where the source code exists, compile the source code with the STRING-XREF option.

2 ♦ Transfer the STRING-XREF output (the .xrf file) to the machine where the Translation Manager is running. You must also transfer the source files.

3 ♦ On the Translation Manager, re-create the same directory structure for your source files as you had on UNIX. Your working directory must be the same as the one you were using when you generated the STRING-XREF file.

4 ♦ In Translation Manger, create a project, but do not choose Extract on the Data tab folder.

5 ♦ When you load the files into the platform, select the .xrf file that you created in Step 1.

6 ♦ Continue with the translation process as you would for any other application.

7 ♦ Verify that the translation is complete by building and compiling the translated application in the Translation Manager.

8 ♦ Using the Progress Data Administration tool, dump the data definitions (.df) and data (.d) files from the project database that the Translation Manager created.

9 ♦ Transfer the .df and .d files to the platform where the translated application will run.

10 ♦ Create an empty Progress database on the target platform with the same name as the project database that the Translation Manager created.

11 ♦ Load the .df and .d files into the new database.

12 ♦ Using the Progress Procedure Editor or Application Compiler, connect to the new database (you must use xlatedb as the Logical Database Name) and to the application database and compile the original source code with the LANGUAGES option.

13 ♦ When running the compiled code, you must use the Language (-lng) startup parameter and specify the appropriate language.

5.7 Using Third-party Tools

You can use the Translation Manager tool in conjunction with third-party tools such as spell checkers by exporting the contents of the Data tab to the tool, using the tool, and then importing the material back into the Data Tab.

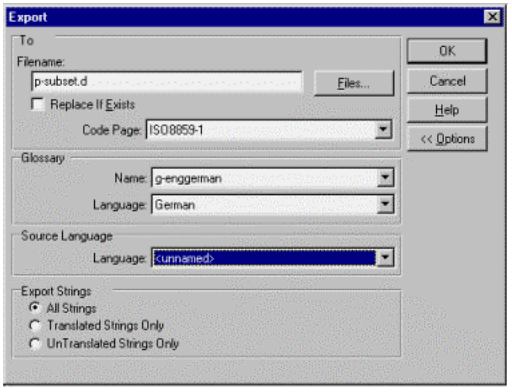
Exporting Data Strings

To prepare a Translation Manager project for export to a third-party tool, you must load and extract procedures, define filters, load strings, and create a glossary as you would for any project.

NOTE: The Export and Import menu choices are only available if there are strings in the project.

Follow these steps to export data to a third-party tool:

1 ♦ From the Data tab folder, choose File→ Export. The Export dialog box appears:



2 ♦ Enter the name of the file to which you want to export the data. Progress recommends that you use the file extension of `.csu` for all your export files. If you use this convention, you will later find it easier to distinguish data files from other types.

3 ♦ Check the Replace If Exists box if you want to replace an existing file with another file of the same name.

4 ♦ Specify the code page for the translation entries you want to export. If you need help determining the code page, contact the Project Manager.

5 ♦ Specify the name and language of the glossary.

6 ♦ Specify the target language of the strings that you want to export.

7 ♦ Specify the source language of the strings you want to export.

8 ♦ Optionally, specify which strings you want to export:

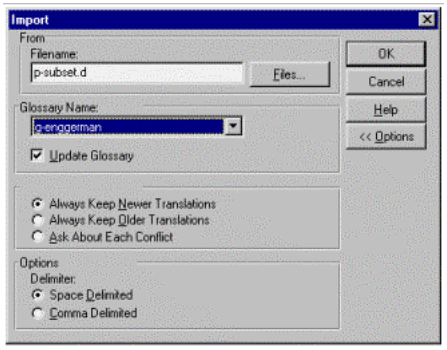
- All strings
- Translated strings only
- Untranslated strings only

You can now export the project to a third-party tool.

Importing Data Strings

When you have completed your work using the third-party tool, follow these steps to import the strings into the Data tab:

1 ♦ From the File Menu, choose Import. The Import dialog box appears:



2 ♦ Enter the name of the text file that contains the data you want to add to the project. Select a file, then choose OK.

3 ♦ Specify the name of the glossary that you want to import. Activate the Update Glossary toggle box to update the glossary.

4 ♦ Choose one of the following options for reconciliation:

- **Always Keep Newer Translations** — The Translation Manager adds all the translations for all source phrases from the import text file, even if the project database already has a translation. If you activated the Update Glossary toggle box, the Translation Manager adds all new (custom) entries. If the translator modified any default entries, the Translation Manager adds the modified entry and retains the existing default entry.

- **Always Keep Older Translations** — If the project database already has a translation for a source phrase, the Translation Manager does not overwrite it with the translation from the import text file. The Translation Manager only includes a translation from the import text file if the project database does not already have one. If you activated the Update Glossary toggle box, the Translation Manager adds all new (custom) entries but keeps all existing default entries.

- **Ask About Each Conflict** — For each source phrase that has a different translation in the project database from that in the import text file, the Resolve Conflict dialog box appears and lets you decide which translation to save in the project database. This option does not affect the glossary.

5 ♦ Specify whether the text file that contains the translation entries you want to import separates the data phrases with a space (Space Delimited) or a comma (Comma Delimited).

Once you have imported your project back into Translation Manager, you are ready to proceed with the translation project.

6

Adding Glossaries

A *glossary* is a table in the project database that consists of pairings of source and target language phrases. The Translation Manager tool lets you create glossaries that provide the translators with suggested translations. The translators can use the Visual Translator tool to add their own glossary entries and modify the target phrases offered in the glossary you prepare for them.

This chapter describes the following topics:

- Creating a glossary
- Importing a glossary
- Exporting a glossary
- Tracking glossary entries

See Chapter 8, “Incorporating a Translated Kit into the Project,” for information on consolidating a glossary into a project when it comes back from the translator.

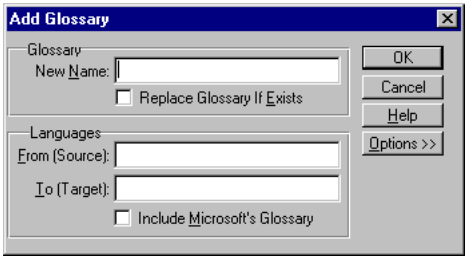
6.1 Creating a Glossary

If you do not have glossary terms to add, you must still create an empty glossary that can accept the translator's entries.

Follow these steps to add a glossary that contains suggested pairings of words for the source and target languages to the current project:

1 ♦ Choose the Glossary tab to open the Glossaries tab folder.

2 ♦ Choose Add. The Add Glossary dialog box appears:



3 ♦ Type a name for the glossary in the New Name field. You should use a name that indicates the source and target languages. For example, if you are creating a glossary containing English source phrases and French target phrases, call the glossary "English-French."

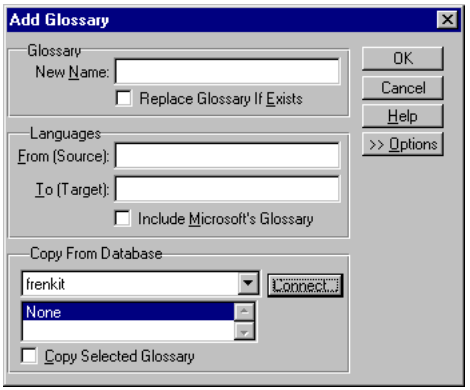
4 ♦ Type the name of the source and target languages.

5 ♦ Specify whether you want to populate the new glossary with terms from a Microsoft Technical Glossary.

If you activate the Include Microsoft Glossary toggle box, a dialog box appears from which you can select a specific Microsoft Technical Glossary:



6 ♦ Choose Options to expand the Add Glossary dialog box and specify whether to create a copy of a glossary from another project database. You must connect to the project database from which you want to copy the glossary:



7 ♦ Choose OK to add the glossary to the current project. The Translation Manager displays a list of the source phrases and target phrases in the Glossaries folder.

You can add source and target text phrases manually into the Glossary tab folder. However, the Translation Manager does not accept text phrases that begin with a question mark (?).

6.2 Importing a Glossary

You can import the glossary data from a text file. The glossary data you want to import must be in a text file, with the source and target phrases separated by commas or spaces.

Table 6–1 shows examples of comma-separated and space-separated glossaries. In both examples, the glossaries order the text phrases first by source language (English) then by target language (French).

Table 6–1: Examples of Imported Glossaries

Comma Delimited	Space Delimited
add,ajouter	add ajouter
application,programme	application programme
apply,appliquer	apply appliquer
blank space,espace	"blank space" "espace"
browse,parcourir	browse parcourir
cancel,annuler	cancel annuler
capitalize,mettre en majuscules	"capitalize" "mettre en majuscules"
change,modifier	change modifier
chart,graphique	chart graphique
check,cocher	check cocher
choose,choisir	choose choisir
click,cliquer	click cliquer
"city, state","ville, département"	"city, state" "ville, département"

You must use quotes in an imported glossary under the following circumstances:

- If you have a comma-delimited glossary and you want to include a comma as part of the text, you must use quotes around the text. For example, to include the comma in an address, you must use quotes:

  ```
  "street, city","calle, ciudad"
  ```

- If you have a space-delimited glossary and you want to include a phrase that has more than one word, you must use quotes around the text:

  ```
  "capitalize" "mettre en majuscules"
  ```

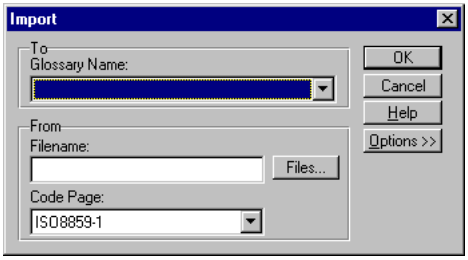
- If the string has quotes, the quotes must be quoted with a tilde (~) as shown:

  ```
  "He said, ~"Hello.~"" "Il a dit, ~"Bonjour.~""
  ```

Follow these steps to import glossary entries from a file to an existing glossary in the project database:

1 ♦ Choose the Glossary tab to open the Glossaries tab folder.

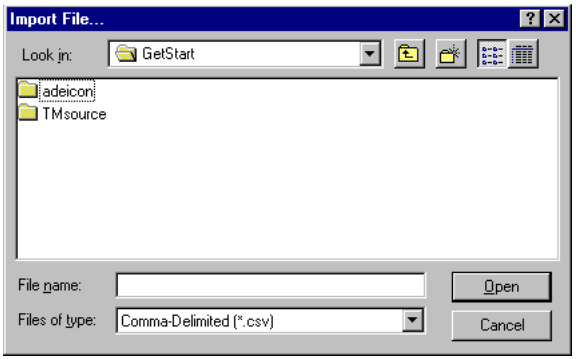
2 ♦ Choose the Import button or choose File→ Import. The Import dialog box appears:



3 ♦ Select the name of the glossary where you want to import the glossary entries.

4 ♦ Enter the name of the text file that contains the glossary.

If you choose Files, the Import File... dialog box appears. Select a file, then choose OK:



5 ♦ Enter the code page for the glossary entries you want to import so that the Translation Manager can properly decode them.

By default, the project and kit databases you create with the Translation Manager use the "undefined" code page.

The code page you specify for the glossary depends on the code pages of the source and target languages of the glossary, and the values you used for the Stream Code Page (-cpstream) and Internal Code Page (-cpinternal) startup parameters when you started Progress.

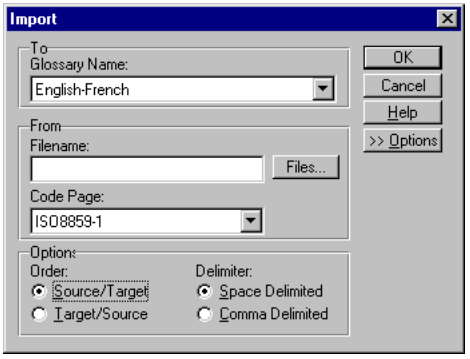
Apply the following rules to help decide which code page to specify for the glossary:

- If the source and target languages in the glossary have the same code page but -cpstream has another code page, select the code page for the source/target languages.
- If the source and target languages in the glossary and -cpstream have the same code page, accept the default.
- If the source and target code pages in the glossary are different, make sure the source code page is the same as the value of -cpinternal, then select the undefined code page in the Import dialog box.

NOTE: You can determine the values of -cpstream and -cpinternal by using the SESSION:CPSTREAM and SESSION:CPINTERNAL attributes, respectively.

For more information on -cpstream and -cpinternal, see the *Progress Internationalization Guide*.

6 ♦ Choose the Options button to expand the Import dialog box:



7 ♦ Specify the order in which the glossary organizes the phrases.

8 ♦ Specify whether the glossary uses a space or a comma as a delimiter between source and target phrases, then choose OK.

6.3 Exporting a Glossary

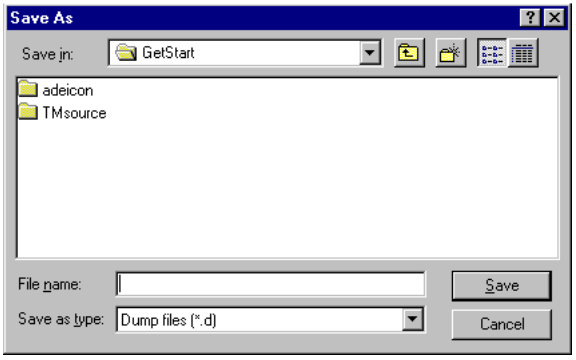
You can export a project glossary to a text file to use in another project. When you export a glossary, the Translation Manager saves it as a space-delimited data (.d) file.

NOTE: When you export a glossary you created in the Translation Manager, the resulting data file consists of data that is in the -cpstream code page (converted from the -cpinternal code page).

Follow these steps to export a glossary:

1 ♦ Choose the Glossaries tab to open the Glossaries tab folder.

2 ♦ Choose the Export button or choose File→ Export. The Save As dialog box appears:



3 ♦ Specify the directory where you want to store the file.

4 ♦ Type the name of the file with a .d extension.

5 ♦ Save the File as a Dump file type, then choose OK.

6.4 Tracking Glossary Entries

When you create a glossary, the Translation Manager marks all the glossary entries you add with a D (for default) in the Type column of the Glossaries tab folder.

When you create a new project, you define whether the translators can alter the default glossary. (See Chapter 4, "Managing Projects," for information on setting translator privileges.) By default, the translators can use the Visual Translator to add their own glossary entries to the glossary, modify the default glossary entries, and enter their own translations.

The Translation Manager keeps track of whether a glossary entry was created in the Translation Manager (default glossary entry) or by the translator in the Visual Translator tool (custom glossary entry). When the translator returns a kit and you merge the kit's glossary back into the project glossary, the Translation Manager displays either a D for default or a C for custom in the Type column of the Glossaries folder to indicate where each glossary entry originated. If the translator altered a default entry, the Translation Manager indicates this with a **yes** in the Modified by Translator column.

Table 6–2 lists some examples of glossary entries you might see in an English/Spanish glossary. You can determine from looking at the Type column that both the Add/Agregar and the Bitmap/Mapa de bits glossary entries were added in the Translation Manager tool and the Close/Cerrar glossary entry was added by the translator in the Visual Translator tool. By looking in the Modified by Translator? column, you can determine that the Bitmap/Mapa de bits default glossary entry was modified by the translator.

Table 6–2: Example Glossary Entries

Source Phrase	Target Phrase	Modified By Translator?	Type
Add	Agregar	No	D
Bitmap	Mapa de bits	Yes	D
Close	Cerrar	Yes	C

7

Preparing a Kit

This chapter describes the following topics:

- What is a kit
- Building a kit for each target language
- Deleting a kit from the project database
- Changing a kit's source language
- Sending a kit to a translator

7.1 What Is a Kit?

You create a kit for each target language. A *kit* consists of:

- A kit database that contains:
  - Text phrases that were extracted from the source code, then filtered and loaded into the project database
  - A default glossary with recommended pairings between the source language and the target language
- Resource procedures so the translators can view the text phrases in context
- An environment (.ini) file so that the translators view the resource procedures in the same user-interface layout as the project manager
- Any image files used in the resource procedures

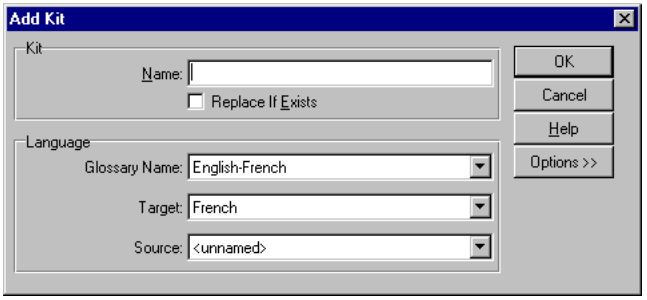
The kit does not include source code or the databases associated with the source code.

7.2 Building a Kit

Follow these steps to build a kit:

NOTE: These steps assume you have already filtered, extracted, and loaded text phrases, generated resource procedures, and created a glossary (even if the glossary is empty) for the project.

1 ♦ From the Kits tab folder, choose Add. The Add Kit dialog box appears:



2 ♦ Type a name for the kit. When naming the kit be aware that:

- You can only use a maximum of eleven characters for the kit name. Translation Manager uses the first eight characters for the name of the kit database. The combined total number of characters for the directory name, the kit name, and the translation string cannot exceed 100 characters.
- Progress Software recommends that you use the convention of starting all the names of your kits with the letter "k." If you use this convention, you will later find it easier to distinguish between a kit and a project database.
- The kit name does not have to be language specific. For example, you could use "k-engfr" to name a kit.
- You cannot use spaces in the kit name.

3 ♦ Select the glossary you want to put into the kit.

4 ♦ Select the language of the glossary. This is the source language in which the glossary was created.

5 ♦ Select a source language for the kit. This is the language the translator will translate from. By default, the source language is unnamed.

6 ♦ Choose OK. The Translation Manager displays the kit's information in the Kits tab folder.

7.3 Deleting a Kit

If you make changes to a project and want to update a kit, you can can re-create the old kit and activate the Replace If Exists toggle box in the Add Kit dialog box. You can also delete the kit and then re-create it.

Follow these steps to remove a kit from the project database:

1 ♦ In the Kits tab folder, select the language kit you want to remove.

2 ♦ Choose Remove. The Translation Manager prompts you to verify that you want to remove the kit.

3 ♦ Choose Yes. The Translation Manager prompts you to specify whether you want to delete the corresponding kit database.

4 ♦ Choose Yes to remove the kit from the project and delete the kit database. Choose No to remove the kit from the project without deleting the kit database.

7.4 Pretranslation

Before you send the kit, you can perform a *pretranslation* on the text phrases. Pretranslation is a task that matches each text phrase with the first possible glossary entry. Translation Manager translates all text phrases at once. This task allows you to review and edit suggested translations before you send the kit to the translators. The translator can also perform pretranslation before starting the manual translation process. To facilitate the review process, you can generate a hardcopy of the Translation Manager PreTranslation exceptions report or print the information to a text file.

You can generate a Translation Manager PreTranslation exceptions report that provides you with the following information about each string within a procedure file:

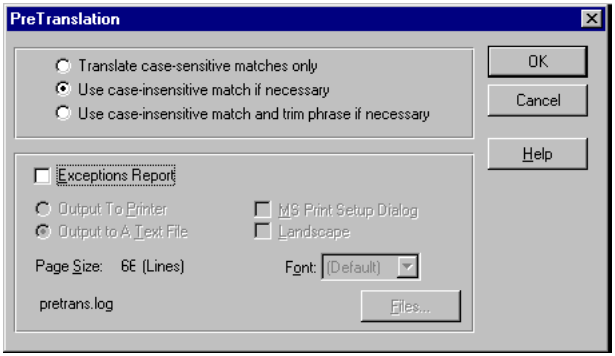
- Untranslated source strings
- Translated strings with no glossary match
- Translated strings that have more than one possible translation

The Translation Manager PreTranslation exceptions report also provides the following summary information, by procedure and by project:

- The total number of translation strings
- The number of new translation strings
- The number of new translation strings that have more than one possible translation
- The number of previously translated strings

Follow these steps to pretranslate text phrases in a kit:

1 ♦ Choose Tools→ TranMan Utilities→ PreTranslation. The PreTranslation dialog box appears:



2 ♦ Select one of these options for pretranslating phrases:

- **Translate case-sensitive matches** — The Translation Manager translates only the text phrases that match a glossary entry exactly. For example, the Translation Manager will not translate “Customer Name” if the glossary entry is “customer name.” Although the text phrase is the same, the case does not match the glossary entry.
- **Use case-insensitive match if necessary** — The Translation Manager searches for an exact match with a glossary entry first. If there is no glossary entry that matches the text phrase exactly, the Translation Manager searches for a match regardless of case. You should select this button if you do not want leading and trailing blanks ignored.
- **Use case-insensitive match and trim phrase if necessary** — The Translation Manager first searches for exact matches with glossary entries, then it searches for a match regardless of case. Finally, it ignores leading and trailing blanks by trimming them from the phrase before looking for a match.

3 ♦ Activate the Exceptions Report toggle box if you want a hard-copy report (or to save the information to a text file).

4 ♦ Select your print options. If you select the Output To Printer radio-set button, you do not need to enter a filename. The default filename is `pretrans.log`. If you select the Output to A Text File radio-set button, you must either enter a filename or use the default filename, `pretrans.log`.

NOTE: The Pretranslation process does not place the accelerator key symbol (&) on the target string. As the project manager, you must decide what the accelerator key will be for the translated phrase and convey that information to the translators.

5 ♦ Choose OK. A Translation Manager PreTranslation report similar to the one shown below prints:

```
Translation Manager PreTranslation Report                              Page 1
Project: GetStart1 02/21/97  10:32 pm
Procedure: TMsource\b-cuslkp.w
     City                                                    Untranslated
     Country                                                 Untranslated
     Current record has been changed.                        Untranslated
     Name                                                    Untranslated
     Search On Customer Name                                 Untranslated
     State                                                   Untranslated
     You must complete or cancel the update before leaving t... Untranslated
     c:\progress\TMsource\b-cuslkp.w should only be RUN PERS... Untranslated
     on-top                                                  Untranslated
       SUMMARY:       10 Strings Total                    100 %
                       0 Exisitng Translations             0 %
                       1 Strings Translated                10 %
                       9 Strings Untranslated              90 %
                      0 Strings with Multiple Translations  0 %Procedure:
TMsource\w-ordtrk.w
     &Update Customer                                        Untranslated
     3,1                                                     Untranslated
     Customer Order Tracking                                 Untranslated
     Customer|Shipment|Order lines|Catalog                   Untranslated
       SUMMARY:        4 Strings Total                     100 %
                       0 Exisitng Translations              0 %
                       0 Strings Translated                 0 %
                       4 Strings Untranslated             100 %
                       0 Strings with Multiple Translations  0 %
GRAND SUMMARY:
                      14 Strings Total                     100 %
                       4 Exisitng Translations              8 %
                       1 Strings Translated                 1 %
                      13 Strings Untranslated              91 %
                       0 Strings with Multiple Translations  0 %
```

7.5 Changing a Kit's Source Language

When you create a kit in Translation Manager, you specify the source language of the application. If the application's source language is one that makes finding translators difficult or costly, you might decide to translate the application into a more common language from which you then launch the remaining translation kit. Translation Manager allows you to change the source language of a kit.

For example, you might have a Spanish application that you want translated into English, French, Greek, and Japanese. You determine that the best strategy is to translate the application into English first, then from English into French, Greek, and Japanese. Use Translation Manager to create the kit, specifying Spanish as the source language. Once the application has been translated into English, use Translation Manager to create additional kits (French, Greek, Japanese) with the source language of English.

When translators open the kit in Visual Translator, they see the pretranslated English phrases as the source phrases, not the Spanish phrases. However, if a phrase was not translated in the original Spanish project, Visual Translator displays the original Spanish phrase.

The Spanish application includes the following phrases:

- abreviatura
- Acerca de
- ventana activa

If the English translator leaves one of the phrases untranslated, when you look at the Data tab for the English translation, you see:

- abreviatura — abbreviation
- Acerca de — About
- ventana activa — ?

When you are ready to pass the English source to your translators and create a kit choosing English as the source language, French as the target language, and the g-engFrench glossary, the untranslated Spanish phrases are also included in the kit. The translator who is translating the application from English to French, sees the following phrases in Visual Translator:

- abbreviation
- About
- ventana activa

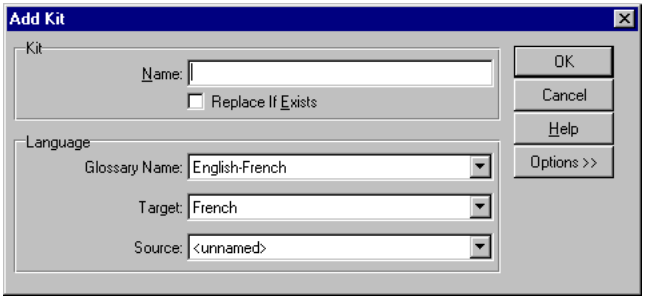
The translation project manager should give the translators clear guidelines for dealing with phrases that remain in the original language. If the translator creates glossary entries for the Spanish and the English phrases, the Data tab for the French translation shows:

- abreviatura — abreviation
- Acerca de — A propos de
- ventana activa — fenetre active

Once you create the kit and translate it into the common language, follow these steps to create new versions of the kit with a new source language:

1 ♦ Choose the Kits tab to open the Kits tab folder.

2 ♦ Choose Add. The Add Kit dialog box appears:



3 ♦ Type a name for the kit. Do not give it the name of the original kit.

4 ♦ Select the new source language.

5 ♦ Select the new glossary you want to put into the kit, then choose OK. The Translation Manager displays the kit's information in the Kits tab folder.

6 ♦ Repeat steps 1–5 until you have created the number of kits you need.

7.6 Sending a Kit to the Translator

This section describes how to send a kit to the translator. To provide the translators with all the data they need, you should create a zip (compressed) file for each kit. The zip file contains all the elements of a kit:

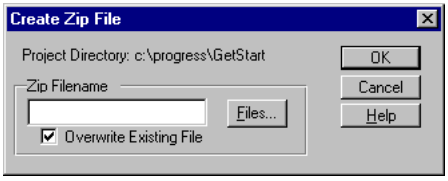
- A kit database
- Resource procedures so the translators can view the text phrases in context
- The environment (.ini) file that you used to create the resource procedures
- Any image files that the resource procedures use

The Preferences dialog box contains zip file options. You should make sure that you have your options set correctly before you create a zip file.

Follow these steps to create a zip file for a kit:

1 ♦ In the Kits tab folder, select the kit you want to package in a zip file.

2 ♦ Choose Create Zip file. The Create Zip File dialog box appears:



3 ♦ Type a filename for the zip file with a .zip extension, then choose OK. The Translation Manager saves the kit in the zip file to the working directory.

You can then send the compressed file to the translator on a diskette or by e-mail.

NOTE: The translator must have the Visual Translator tool to decompress, open, and translate the kit.

8

Incorporating a Translated Kit into the Project

After you send a kit to a translator, the application is translated using either the Visual Translator tool or a third-party translation tool. Then the translator returns the translated kit to the project manager. This chapter describes the following topics:

- Receiving the translated kit from the translator
- Consolidating the translated kit into the project database
- Compiling the source code with the translated text phrases
- Running the compiled code in a target language

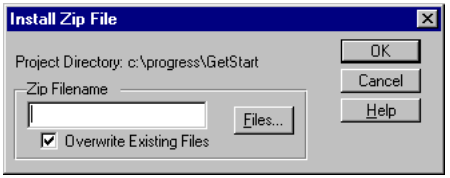
8.1 Receiving the Kit from the Translator

When you receive a translated kit from the translator it contains only the kit database. It does not include the resource procedures or the environment (.ini) file that you sent with the kit.

Follow these steps to unzip a translated kit database into the project directory:

1 ♦ Make sure your working directory and PROPATH are the same as when you created the kit zip file.

2 ♦ In the Kits tab folder, choose Install Zip file. The Install Zip File dialog box appears:



3 ♦ Activate the Overwrite existing Files toggle box if you want to overwrite an existing kit database that has the same name as the kit database you are installing.

4 ♦ Enter the pathname for the file that contains the translated kit, then choose OK.

The Translation Manager unzips the zip file into the project directory.

8.2 Consolidating the Translated Kit into the Project Database

This section describes how to consolidate (merge) the kit translation data and glossary into the project database.

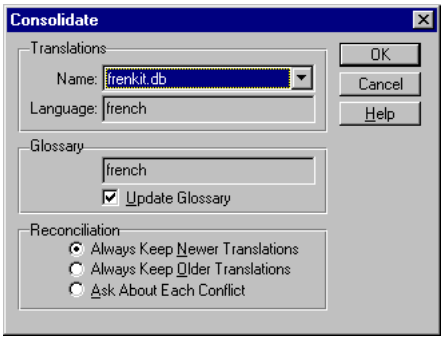
When you consolidate translations, you add the target text phrases the translator created to the project database. You can then compile the source code with the translated text phrases and run the application in the target language.

When you consolidate the glossary, you update the project glossary with the modifications the translator made to the glossary you sent with the kit. The project glossary grows with each translated kit, thereby making each subsequent translation effort easier. You can then export the project glossary to use in a related project.

Follow these steps to consolidate the translations and the glossary from the kit into the project database:

1 ♦ Choose the Open button or choose File→ Open to open the project.

2 ♦ In the Kits folder, choose Consolidate. The Consolidate dialog box appears:



3 ♦ Select the name of the kit you want to consolidate. The Translation Manager displays the name of the target language in the Language field and the name of the glossary for the kit in the Glossary field.

4 ♦ If you want to update the project's glossary with changes and additions from the kit's glossary, activate the Update Glossary toggle box.

5 ♦ Activate one of the following reconciliation radio buttons, then choose OK:

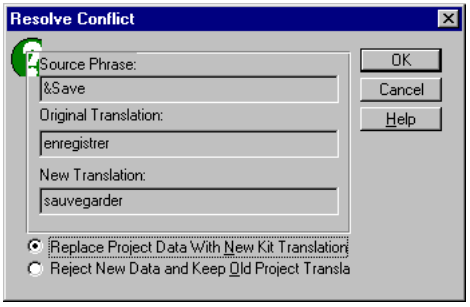
- **Always Keep Newer Translations** — The Translation Manager adds all the translations for all source phrases from the kit, even if the project database already has a translation.

  If you activated the Update Glossary toggle box, the Translation Manager adds all new (custom) entries. If the translator modified any default entries, the Translation Manager adds the modified entry and retains the existing default entry.

- **Always Keep Older Translations** — If the project database already has a translation for a source phrase, the Translation Manager does not overwrite it with the translation from the kit database. The Translation Manager only includes a translation from the kit if the project database does not already have one.

  If you activated the Update Glossary toggle box, the Translation Manager adds all new (custom) entries but keeps all existing default entries.

- **Ask About Each Conflict** — For each source phrase that has a different translation in the project database from that in the kit database, the Resolve Conflict dialog box appears and lets you decide which translation to save in the project database:



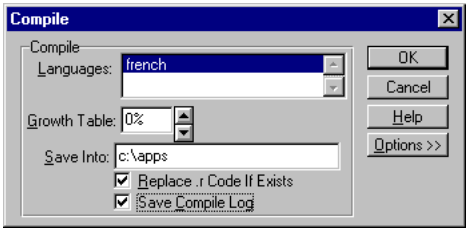
This option does not affect the glossary.

8.3 Compiling the Source Code with the Translated Phrases

This section describes how to recompile the source code to create new run-time (.r) code to include the translated text phrases from one or more languages. The compiled .r code contains a separate text segment for each language. For example, if you translate an application from English into German, French, and Spanish, the resultant .r code contains four text segments (including the English source).

You must have already consolidated the kits to compile the source code with the translation data. Follow these steps to compile the source code:

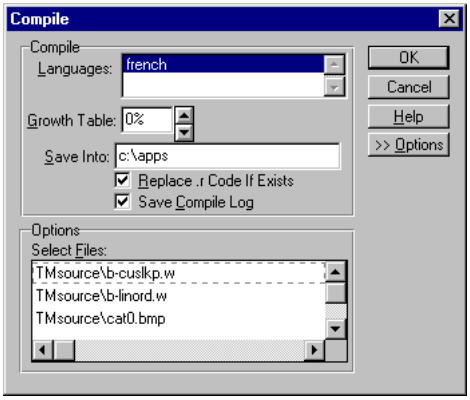
1 ♦ From any folder, choose the Compile button or choose Build→ Compile. The Compile dialog box appears:



2 ♦ Select the languages you want to compile. Press and hold **CTRL** to select more than one language.

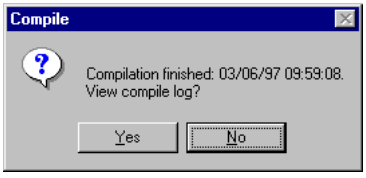
3 ♦ If you want to expand the length of the text phrases for the target languages, specify the percentage of the internal text segment growth table you want to use in the Growth Table combo box. For more information on the process the Translation Manager uses for text segment growth, see the "Text Segment Expansion" section.

4 ♦ Choose Options if you want to compile only certain procedures:

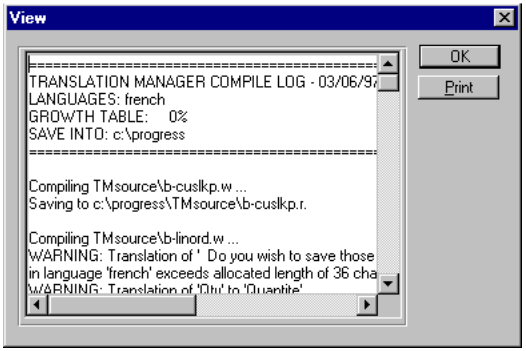


5 ♦ Choose OK to start the compile.

A status meter appears indicating the Compiler's progress. When the Compiler is finished, a message appears indicating the time and date the Compiler completed the compilation and prompts you to specify whether you want to view the compile log file:



6 ♦ Choose Yes to view the compile log:



See Chapter 11, "Troubleshooting," for information on the common error messages and warnings you might view in the compile log.

8.3.1 Text Segment Expansion

This section describes the Translation Manager text segment expansion process.

Text segment growth is the process of expanding source code text segment allocation at compile time so that translations can fit in their text segments without truncation. Text segment growth is important when you translate a compact language, such as English, into other less compact languages. For example, the English word "file" is shorter than the French translation "fichier."

The purpose of text segment expansion is to avoid truncation of text phrases. Text phrase truncation occurs when the Progress Compiler allocates a given number of characters for a text phrase in the text segment but a translation of the text phrase exceeds that number. The translated text phrase is truncated in the text segment and appears truncated on the screen. To overcome this common difficulty, use text segment expansion to add a number of extra characters to each text phrase.

A drawback to the text expansion process is that additional characters might disturb the layout of the interface.

The best solution is for developers to create source code that specifies maximum-length options for character strings to provide adequate space for truncation. For more information on how to hard code the length of a text string, see the character-string literal section of the *Progress Language Reference*.

The Translation Manager applies text segment growth to a text phrase only if it has a translation and does not have a hard-coded string-length attribute. If a text string is marked untranslatable (:U), or if it is explicitly set to a maximum length (for example, “hello” R10), the Translation Manager does not apply the text-segment growth expansion factor to that individual text phrase. For more information on the TRIM string attribute, see Chapter 1, “Preparing Your Application for Translation.”

The Translation Manager uses the algorithm in Table 8–1 to set the growth factor. For short strings, the Translation Manager increases strings by 200%, meaning that the new strings are three times as long as the original strings (original + 200%).

Table 8–1: Translation Manager Internal Text Expansion

Length of Source Text Phrase	Text Segment Expansion Factor
Up to 10 characters	200%
11 to 20 characters	100%
21 to 30 characters	80%
31 to 50 characters	60%
51 to 70 characters	40%
Over 70 characters	30%

The Compile dialog box lets you enter a value in the Growth Table field. This value specifies the percentage of the internal table’s growth factor that the Translation Manager uses to expand text strings.

Table 8–2 shows examples of how the Translation Manager determines the length of the target phrase.

Table 8–2: Text Expansion Examples

Value of Growth Table Field in the Compile Dialog Box	Length of Source Text Phrase	Corresponding Internal Table Expansion Factor	Expansion Factor Translation Manager Uses	Length of Target Text Phrase
0%	7	200%	0% of 200 = 0%	7
100%	5	200%	100% of 200 = 200%	15
50%	10	200%	50% of 200 = 100%	20
150%	50	60%	150% of 60 = 90%	95

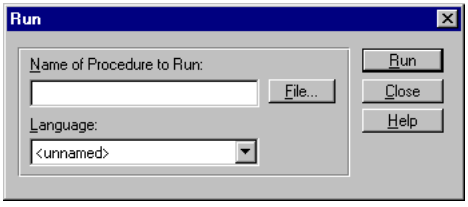
8.4 Running the Compiled Code in a Target Language

When you are ready to test the translated r-code, use a computer that is set up to run in the target language you want to test. For example, to see Russian text, run the application on Russian Windows.

Follow these steps to run the compiled code in a target language:

1 ♦ Before you run the compiled code, make sure that the directory that contains the compiled code is listed in your PROPATH before any directory containing similarly named source code files. See Chapter 3, "Getting Started," for more information on PROPATH.

2 ♦ Choose Build→ Run. The Run dialog box appears:



3 ♦ Enter the name of the procedure you want to run.

4 ♦ Select the language in which you want to run the procedure, then choose OK.

8.4.1 Running Code from the Command Line

If you decide to run the compiled code (r-code) directly from the command line, you should consider the following:

- When you change the value of the CURRENT-LANGUAGE statement, you affect all the text phrases in the r-code that Progress subsequently runs. Any r-code that is already on the run-time stack still runs with the previous CURRENT-LANGUAGE setting. Also, CURRENT-LANGUAGE is a global value. If you return to a previously running piece of r-code from the place where you changed the CURRENT-LANGUAGE, the value of CURRENT-LANGUAGE is the changed value, not the language with which the "old" r-code is running.

- To query an r-code session to see the available languages, use the following code, where *my.p* is the name of the procedure you are running:

```
RCODE-INFO:FILE-NAME = "my.p".
DISPLAY RCODE-INFO:LANGUAGES FORMAT "X(50)".
```

Chapter 1, "Preparing Your Application for Translation," describes how developers can change their source code to make it easier to translate. Chapter 9, "Updating a Project," describes how to update a project if the developers change any of the source code procedures and how to edit target phrases in the Translation Manager tool. Chapter 11, "Troubleshooting," describes how to resolve some common error messages.

9

Updating a Project

This chapter describes the following topics:

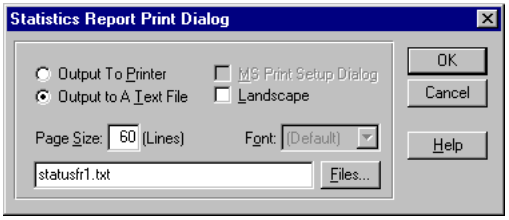
- Printing a Statistics report
- Editing translations and glossary entries
- Scanning for changed source files

9.1 Printing a Statistics Report

The Statistics report provides you with information on the current project database. The report contains information that helps you keep track of your translations, for example, the name of the project, creation date, date of last update, and the number of procedures.

You can generate a hardcopy of a Statistics report or send a report to a text file from the Statistics tab folder. Follow these steps:

1 ♦ Choose the Print button or choose File→ Print from the Statistics tab folder. The Statistics Report Print Dialog box appears:



2 ♦ Select your print options. If you select the Output To Printer radio-set button, you do not need to enter a filename. The filename defaults to `status.txt`. If you select the Output to A Text File radio-set button, you must either enter a filename or use the default filename, `status.txt`.

3 ♦ Choose OK. A Statistics report similar to the following prints:

```
Project:GetStart1              03/10/97          Page1
Project Name                                  kfrench
Description
Revision                                      1.0
Create Date                                   01/10/97
Last Updated                                  03/06/97
Database Size (in Bytes)                      458.8K
Display Type                                  Graphical
Number of Procedures                          16
Number of Phrases                             164
Number of Unique Phrases                      109
Number of Words                               347
Number of Unique Words                        188
Current Procedure                             None
Current Object                                None
Number of Frames                                0
Number of Objects                               0
% Translated                                  9%
# Items In Glossary                           1063
```

9.2 Editing Translations and Glossary Entries

This section describe how to edit the text phrases and glossary entries in a project. It discusses the following options:

- Browsing through the tab folders
- Sorting rows and ordering columns
- Inserting and deleting rows
- Cutting, copying, and pasting text

9.2.1 Browsing Through the Tab Folders

You can browse through the contents of the Data and Glossaries tab folders using the following techniques:

- **Use the vertical scrollbars** — To scroll up and down the rows of information and view all the text phrases or glossary entries.
- **Use the horizontal scrollbars** — To scroll right and left to view all the columns of information in the tab folder.
- **Choose a column label and type a letter** — To automatically scroll down to the row of information to find the next occurrence of a phrase that begins with the letter you typed.

9.2.2 Sorting Rows and Ordering Columns

You can change the order of rows or columns in the tab folders with the Sort Rows and Order Columns options of the View menu. See online help for more information on these options.

9.2.3 Inserting and Deleting Rows

You can insert or delete rows in the Glossaries tab folder when you use the Insert or Delete toolbar buttons or Edit menu options. You can also delete rows in the Data tab folder.

9.2.4 Cutting, Copying, and Pasting Text

You can edit the text in a cell of the tab folder when you use the Cut, Copy, and Paste toolbar buttons or Edit menu options. You can also edit any text phrase by double-clicking on the text to display the Long String Translator dialog box. You can view source string text in the Source String viewer or view and edit target string text in the Target String viewer.

9.3 Scanning for Changed Source Files

When the software developers in your company update the application that is being translated, you should check the source procedures listed in the project database for changes. The developers might have deleted, added, or otherwise changed source procedures. When you use the File Scan option, the Translation Manager compares the list of source procedures in the project database against the statistics of the source procedures in the source code directory, then marks (internally in the Translation Manager) any changed procedures in your list of source procedures.

You can then extract text phrases, create resource procedures for only the changed files, and avoid repeating the extraction process for the majority of the procedures.

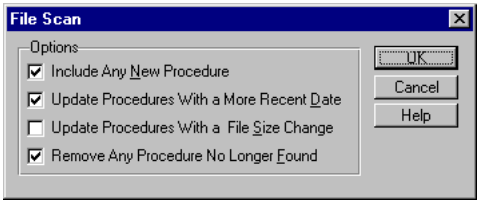
NOTE: The File Scan option scans the broad file characteristics of files, such as last-modified date and file size. It does not compare the actual contents of files.

You can specify the following options for the Translation Manager to use when it scans files:

- **Include Any New Procedure** — If a procedure exists in the directory but is not on your list, the Translation Manager adds it to your list and marks it as changed.
- **Update Procedures With a More Recent Date** — If a developer modifies a procedure and alters its contents, the Translation Manager marks it on the list as changed. Later, during the extraction process, the Translation Manager tool extracts and loads only those marked procedures.
- **Update Procedures With a File Size Change** — If the size of a file is different than it was when you added the filename to the source procedure list, the Translation Manager marks it on the list as changed.
- **Remove Any Procedure No Longer Found** — If the Translation Manager cannot find a source procedure in the directory, it removes the procedure name from the list.

Follow these steps to update your list of source procedures by scanning for changed files:

1 ♦ From the Procedures folder, choose Scan. The File Scan dialog box appears:



2 ♦ Select the options you want to scan for, then choose OK.

You can now extract the text phrases for the marked procedures, filter them, and load them into the project database. You can then re-create the kits and send them to the translators for translation.

10

Tutorial

This tutorial guides you through the steps of a sample Translation Manager project. This project uses an English/French glossary to translate an application from English to French and includes the following sections:

- Creating a new Translation Manager project
- Adding and removing files
- Extracting, filtering, and loading files; and creating resource procedures
- Adding glossaries
- Creating a kit and a zip file to send to the translator
- Installing a zip file and consolidating a kit received from the translator
- Viewing statistics

10.1 Preparing to Use This Tutorial

Before you start working on this tutorial, you need to:

- Have Progress and the Translation Manager tool installed on your system. You must do a complete install. See the *Progress Installation and Configuration Guide Version 9 for Windows* for information in installation options.
- Make sure that your PROPATH contains the following files:

  <current directory>

  `C:\Sports2000\trans-r`

  `C:\Sports2000\trans-r\gui`

  `C:\Sports2000\src`

  `C:\Sports2000\src\gui`
- Use `C:\Progress` as your working directory. The directory structure must be as follows:

  `C:\Sports2000`

  `C:\Sports2000\trans-r`

  `C:\Sports2000\trans-r\gui`

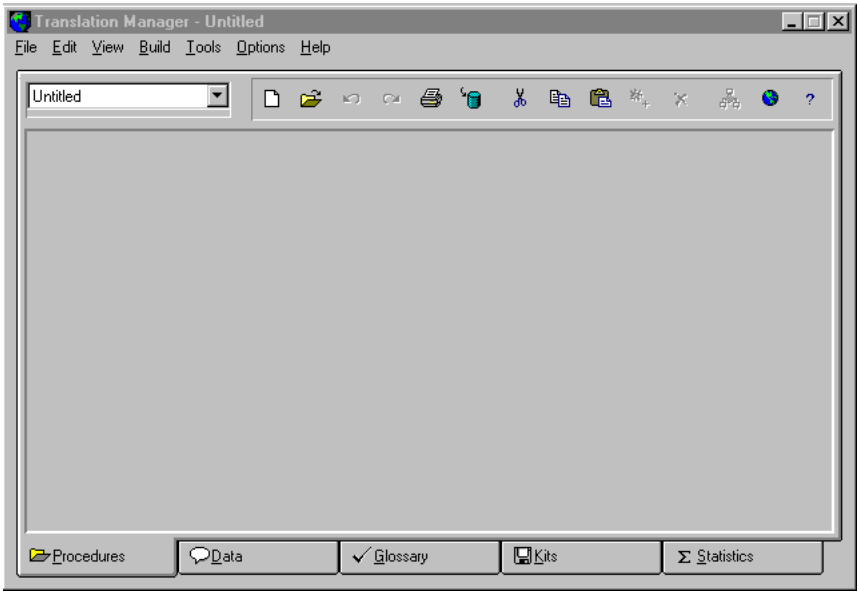
  `C:\Sports2000\src`

  `C:\Sports200\src\gui`
- Copy the sample files from `C:\Program Files\Progress\prolang\samples\tranman` to your `C:\Sports2000\src\gui` directory

10.2 Creating a New Translation Manager Project

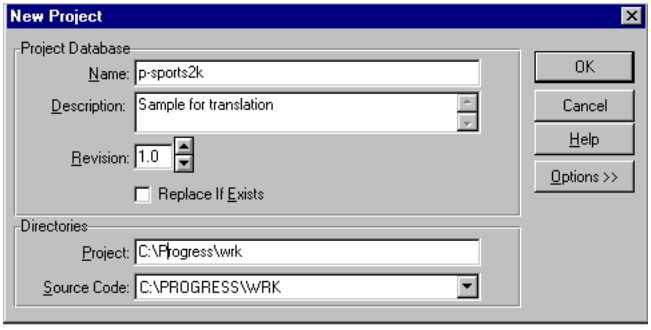
Follow these steps to prepare a new project for translation:

1 ♦ Click on the Start button and choose Programs→ Progress→ Translation Manager. The Translation Manager window appears:



2 ♦ Choose File→ Database Connections and connect to the Sports2000 database.

3 ♦ Choose File→ New to create a new project. The New Project dialog box appears:



4 ♦ Type **p-sports2k** in the Name field.

5 ♦ Type **sample for translation** in the Description field.

6 ♦ Type **C:\Sports2000\project** in the Project field.

7 ♦ Select `C:Sports2000\src` from the Source Code list.

8 ♦ Choose OK.

You have created a new sample Translation Manager project called `p-sports2k`.

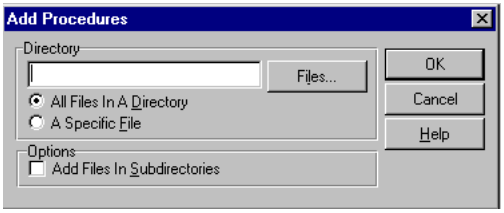
NOTE: There are five tabs along the bottom of the Translation Manager window. You work from left to right from the Procedures tab to the Statistics tab. Within each tab you work from top to bottom with the buttons on the right of the window.

10.3 The Procedures Folder: Adding Files

In the Procedures tab you add procedures that you want to translate, delete procedures that you do not want to include, and scan procedures that were translated earlier to see if there have been any changes.

In this section you will add files to include in your translation project. Follow these steps to add files to your project:

1 ♦ Choose the Add button. The Add Procedures dialog box appears:

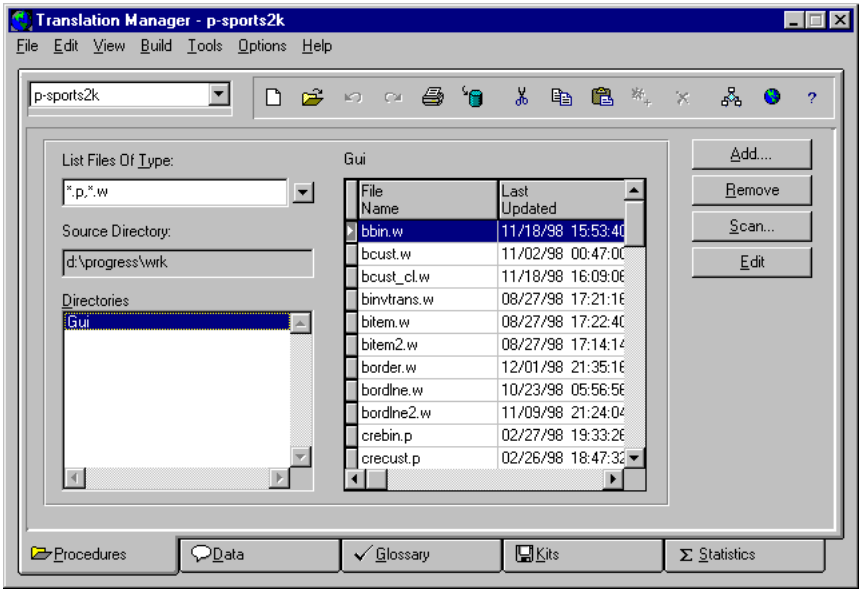


2 ♦ Type **C:\Sports2000\src\gui** in the Directory box.

3 ♦ Select the All Files in a Directory and Add Files in Subdirectories buttons

4 ♦ Choose OK. An Information box appears reporting on the number of procedures added to the project.

5 ♦ Choose OK. The Translation Manager Procedure window appears with the list of files added:



You have completed work in the Procedures tab and have added the source files you want to include in your translation project.

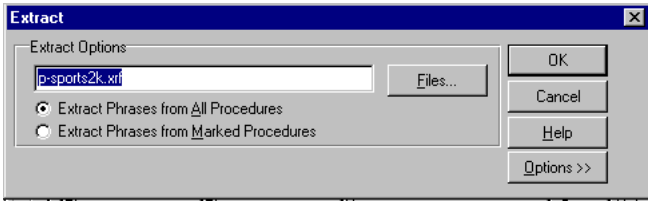
10.4 The Data Tab Folder

You are now ready to work in the Data tab section, which is where you prepare text for the translator. In order to prepare text for translation, you must first extract text phrases from the source procedure, filter and load files for translation, and create resource procedures.

10.4.1 Extracting Text Phrases

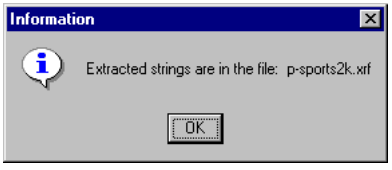
Follow these steps to extract text phrases for translation:

1 ♦ In the Data tab folder, choose Extract. The Extract dialog box appears with the `p-sports2k.xrf` file highlighted:



2 ♦ Choose Extract Phrases from All Procedures and choose OK.

Translation Manager extracts phrases from source procedures by compiling the source procedures and copying the text phrases into a temporary .xrf file. All text phrases except null strings and strings marked with :U (untranslatable) are extracted. An Information box appears telling you the name of the file that contains the extracted strings:

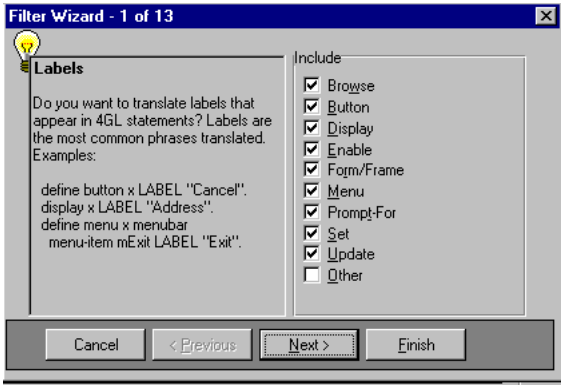


3 ♦ Choose OK.

10.4.2 Specifying Filters

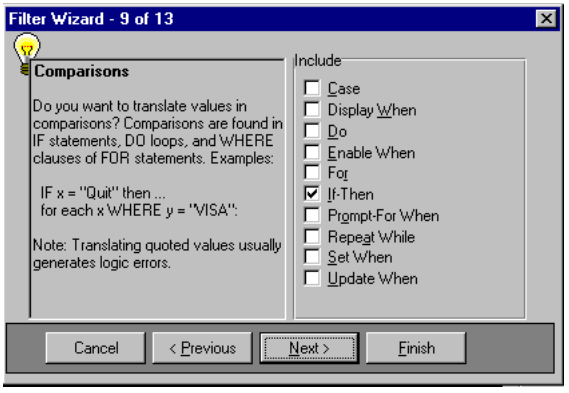
You are now ready to specify which filters to use. Filters allow you to load only those text phrases you want to translate. Follow these steps to specify filters:

1 ♦ In the Data tab folder, choose Filters. The Filter Wizard appears:

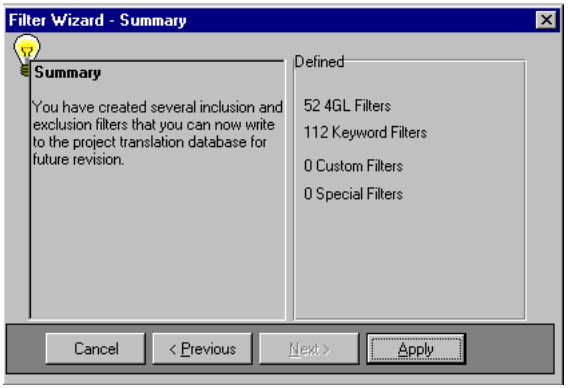


2 ♦ Choose Next on pages 1–8 to accept the default filters.

3 ♦ On page 9, select to include “If-Then” statements:



4 ♦ Choose Next on pages 10–13. The Filter Wizard Summary dialog box appears:



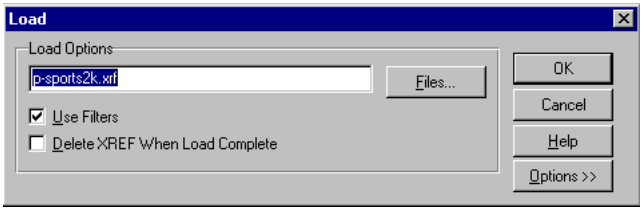
5 ♦ Choose Apply.

You have defined your filters and can now load the text phrases into the project database.

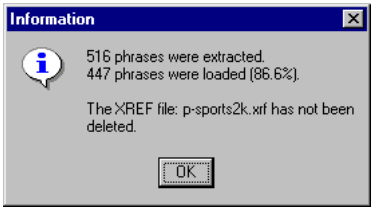
10.4.3 Loading Text Phrases

Translation Manager passes the text phrases in the `.xrf` file through the types of filters defined with the Filter Wizard and then loads the text phrases into the project database. Follow these steps to load the text phrases:

1 ♦ Choose Load. The Load dialog box with the `p-sports2k.xrf` file in the Load Options box appears:



2 ♦ Choose OK. An Information box appears reporting the number of phrases that were extracted and loaded. Choose OK:

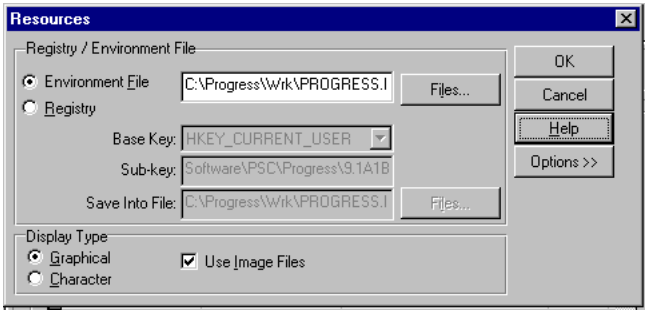


You can now view the text phrases that were loaded into the project database.

10.4.4 Creating Resource Procedures

Now you are ready to create resource procedures that Translation Manager uses to translate text phrases. Follow these steps to create resource procedures:

1 ♦ Choose Resources. The Resources dialog box appears:

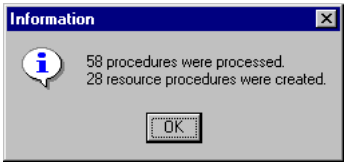


2 ♦ Type **C:\Sports2000\progress.ini** in the Files box.

3 ♦ Check the following boxes:

- Environment File
- Graphical
- Use Image Files

4 ♦ Choose OK. An information box with a report on the Resource Procedures created appears:



You have created the resource procedures that the Translation Manager uses to translate text phrases in the context of the user interface.

10.5 The Glossary Folder

Now you are ready to move to the Glossary tab where you create a glossary for the target language. Follow these steps to add a glossary to your project:

1 ♦ In the Glossary tab, choose Add. The Add Glossary dialog box opens:



2 ♦ Type **English-French** in the New Name field.

3 ♦ Type **English** as the From (Source) language and **French** as the To (Target) language.

4 ♦ Check the Include Microsoft's Glossary box. The Microsoft Glossaries dialog box appears:

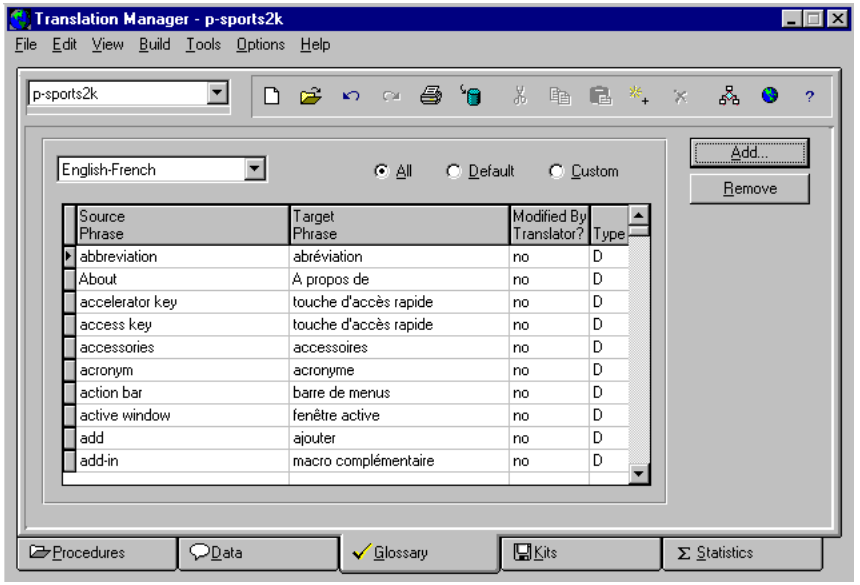


5 ♦ Select English/French and choose OK.

6 ♦ Choose OK in the Add Glossary dialog box. An Information box appears after the glossary is loaded:



7 ♦ Choose OK. The glossary appears in the Glossary tab:



CAUTION: The Remove button removes the entire glossary.

You have defined the glossary that the translator will use. When there are many translation projects, it is important that translators use the same glossary to be sure that they translate phrases consistently.

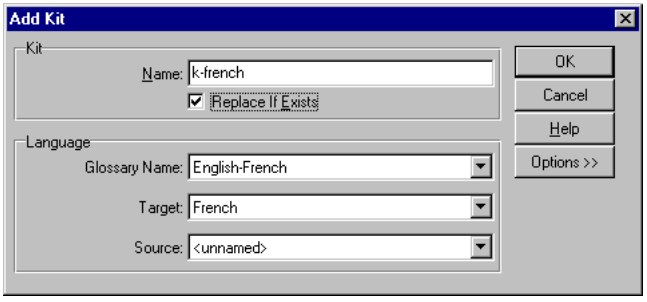
10.6 The Kits Folder

Now you are ready to work in the Kits folder of the Translation Manager. You must create a kit for your project. The kit contains the text phrases, resource procedures, and glossary that you have created. After creating the kit, you will create a zip file to send to the translator.

10.6.1 Creating a Kit

To create a kit for your French translation, move to the Kits tab and follow these steps:

1 ♦ Choose the Add button. The Add Kit dialog box appears:



2 ♦ Type **k-french** in the Name box.

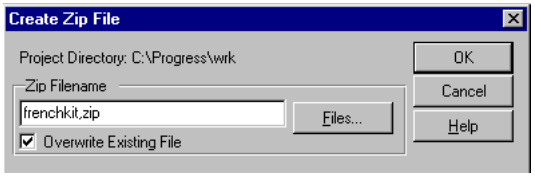
3 ♦ Check the Replace If Exists box and choose OK. The `k-french.db` kit is added to the list of kits.

Translation Manager copies text phrases, glossary entries, resource files, environment files, and images. You are now ready to create a zip file to send to the translator.

10.6.2 Creating a Zip File

Follow these steps to create a zip file:

1 ♦ Choose Create Zip File. The Create Zip File dialog box appears:



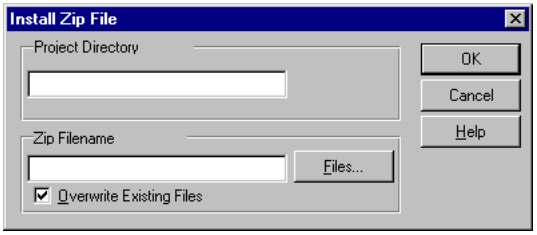
2 ♦ Type **frenchkit.zip** in the Zip Filename field and choose OK. Translation Manager creates the zip file that you send to the translator.

At this point the Project Manager sends the zip file to the translator who will use the Progress Visual Translator tool to translate the project. When the translation is complete, the translator sends the zip file back to the Translation Manager project manager for final work.

10.7 Installing a Zip File and Consolidating a Kit Received from the Translator

Now you have received the translated zip file from the translator. To install the zip file and consolidate the kit, follow these steps:

1 ♦ In the Kits tab choose Install Zip File. The Install Zip File dialog box appears:

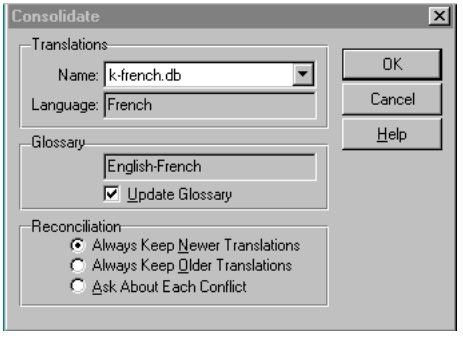


2 ♦ Type **C:\sports2000\src** in the Project Directory box.

3 ♦ Type **frenchtran.zip** in the Zip Filename box, and choose OK to unzip the file into your project directory.

Follow these steps to consolidate your kit:

1 ♦ Choose Consolidate. The Consolidate dialog box appears:



2 ♦ Select the **k-kitname.db** file from the Name dropdown list.

3 ♦ Check the Update Glossary toggle box.

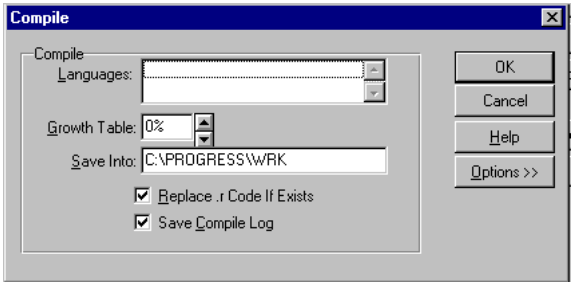
4 ♦ Select Always Keep Newer Translations and choose OK.

This process incorporates the information in the kit into the project database.

10.8 Compiling Source Code with the Translated Phrases

Follow these steps to compile the program with the translated data:

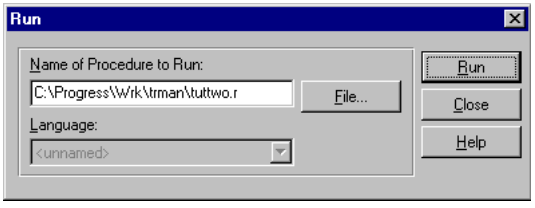
1 ♦ Choose Build→ Compile from the Translation manager menu bar. The Compile dialog box appears:



2 ♦ Choose OK. Translation Manager compiles the code. The Compile box appears. Choose yes to view the compile log. The View box with the compile log appears.

3 ♦ Choose OK to return to the TM window.

4 ♦ Choose Build→ Run from the TM menu bar. The Run dialog box appears:



5 ♦ Enter **wmain.r** in the Run dialog box and choose Run. The Translated sample application appears.

You have successfully completed a sample Translation Manager project.

11

Troubleshooting

This chapter answers common questions about the Translation Manager tool and describes solutions to common problems.

Why can't I start the Translation Manager?

You must install a copy of Progress Version 9 that includes the Translation Manager tool before you can start the tool. See the *Progress Installation and Configuration Guides* for *UNIX* or *Windows* for information on system requirements and how to install Progress.

Why do I receive error messages when I try to extract text phrases?

Before you extract text phrases, you must connect to the databases that the source procedures typically use and make sure they are connected with the proper logical names. Also, make sure the PROPATH includes the source procedures and their associated include (.i) files.

Why do I receive a "cannot find a file" error message?

You probably do not have your PROPATH set properly. See Chapter 3, "Getting Started," for information about setting your PROPATH.

Why do I receive an "open database table is full" error message when creating a kit?

By default, Progress allows you to open five databases simultaneously. If you are creating more than three kit databases, or you have many application databases connected, you might receive an error indicating that the open database table is full (three kit databases, one project database, and one application). To open more than five databases, exit Progress. Then, when you restart Progress, specify the Number of Databases (-h) startup parameter with a value greater than five.

How do I include the text phrases in SmartObjects?

If the application you want to translate contains SmartObjects and you want to translate the text phrases in the SmartObjects, you must include the source code for those SmartObjects in the project's list of procedures. The source code for Progress SmartObjects is located in `$DLC\gui\adm\objects`. For more information on selecting source procedures for a project, see Chapter 4, "Managing Projects."

Why can't the tool display resource procedures?

Before you generate resource procedures, you must connect to the databases that the source procedures typically use. Also, make sure the databases are connected with the proper logical names. Make sure that the PROPATH is set up correctly.

Why does the Visual Translator tool display the text phrases in an undesired font in the resource procedures?

If you do not enter an environment (.ini) filename or you enter the wrong .ini filename, the translator can view resource procedures, but the Visual Translator displays the resource procedures with the wrong font and color mapping, thereby giving the wrong feedback to the translator. For example, if the Visual Translator cannot find the proper font file, it will use the default 10-point font for a window, no matter what the font should be.

How do I avoid common errors and warnings in the Compiler's log file?

To avoid warnings in the compile log file about text phrases that exceed the allocated length, you can increase the percentage of the Growth Table field in the Compile dialog box and repeat the compile.

To avoid errors about fill-in fields that do not fit in frames, you can talk with the application developer who probably needs to change the source code or talk to the translator about using an abbreviation for the translation. You must fix these errors before you can run the compiled code.

Why do I receive error messages at run time?

If you have any error about fill-in fields that do not fit in frames, check your compile log file. You should talk with the application developer who probably needs to change the source code. You must fix these errors before you can run the compiled code.

A

Translation Manager Interface Reference

This appendix provides reference information for the following parts of the Translation Manager interface:

- Tab folders
- Menu bar
- Toolbar

For online reference information about a dialog box, choose the Help button in the dialog box.

A.1 Tab Folders

The main interface component of the Translation Manager tool is the tab folder. When you select one of the tabs shown below, the Translation Manager displays the corresponding folder:



The tabs allow you to view the following folders:

- **Procedures** — Lets you specify the source files you want translated. You can specify entire directories or individual files. Also, it lets you peruse and edit source procedures.
- **Data** — Lets you extract text phrases from the source procedures you selected in the Procedures folder, specify filters, and load the extracted text phrases through the filters into the project database. Lets you create *resource procedures*—the user-interface procedures that the translators will use to translate text phrases and user-interface objects in context. Also, it lets you import and export data strings to a third-party translation tool.
- **Glossary** — Lets you create and maintain glossaries with the option of using Microsoft Glossaries provided with the Translation Manager tool. Also, you can import and export glossary data.
- **Kits** — Lets you create a language kit for each translator. Each kit, when zipped (compressed), includes a database that contains text phrases to be translated and a glossary, resource procedures, image files, and an environment (`progress.ini`) file for the resource procedures. Also, you can consolidate a translated kit into the project database.
- **Statistics** — Lets you view various statistics for the current project including the date you created the project, the size of the project, the code page, the number of procedures and text phrases, and the number of words. You can also generate a hardcopy of the statistics report or save the report to a text file.

A.1.1 Procedures Tab Folder

Use the Procedures tab folder to create a list of the source procedures you want to translate. Chapter 4, "Managing Projects," describes how to use this folder. Figure A–1 shows the Procedures tab folder.

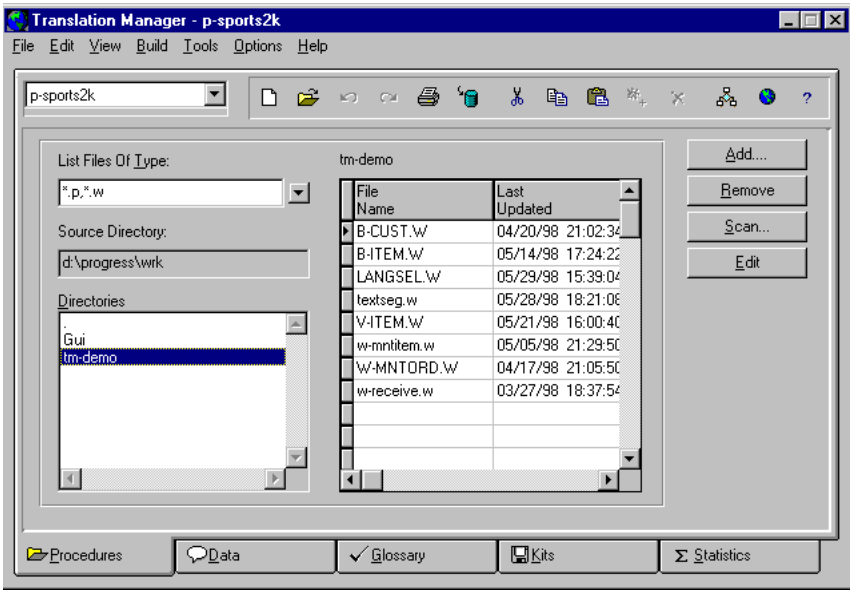


Figure A–1: Procedures Tab Folder

The Procedures tab folder contains the following user-interface elements:

- **Source Directory** — Displays the pathname of the root directory for the source procedures.
- **Directories** — Displays the subdirectories within the source directory that contain the source procedures.
- **Browse Viewer:**
  - **File Name** — Displays the name of the source procedure (.p or .w) file.
  - **File Size** — Displays the size of the source procedure, in bytes.
  - **Last Updated** — Specifies the date and time the source procedure was last changed by a developer.

- **Status of Last Kit** — Specifies the number of translated text phrases and the total number of text phrases in the kit that you last consolidated into the project database.
- **Resources Generated?** — Indicates whether you have generated a resource procedure (.rc) file from the source procedure.
- **Comments** — Lets you enter comments about the procedure for the translator.

- **Action Buttons:**
  - **Add** — Displays the Add Procedures dialog box, which lets you add the names of the source procedures you want to translate to the project database.
  - **Remove** — Removes the selected procedure or directory from the list of procedures in the project database. Note that you can only remove a directory after deleting all the procedures it contains.
  - **Scan** — Displays the File Scan dialog box, which lets you compare the source procedures in the project database list to the source procedures in the source code directory and internally marks files so you can extract them separately.
  - **Edit** — Opens a Procedure Editor window in which you view and edit the selected source procedure.

A.1.2 Data Tab Folder

Use the Data tab folder to extract text phrases from the source procedures, set filters, load text phrases into the project database, generate resource procedures, and view or edit the source procedures. Chapter 5, "Preparing Data for Translation,"discusses how to use this folder. Figure A–2 shows the Data tab folder.

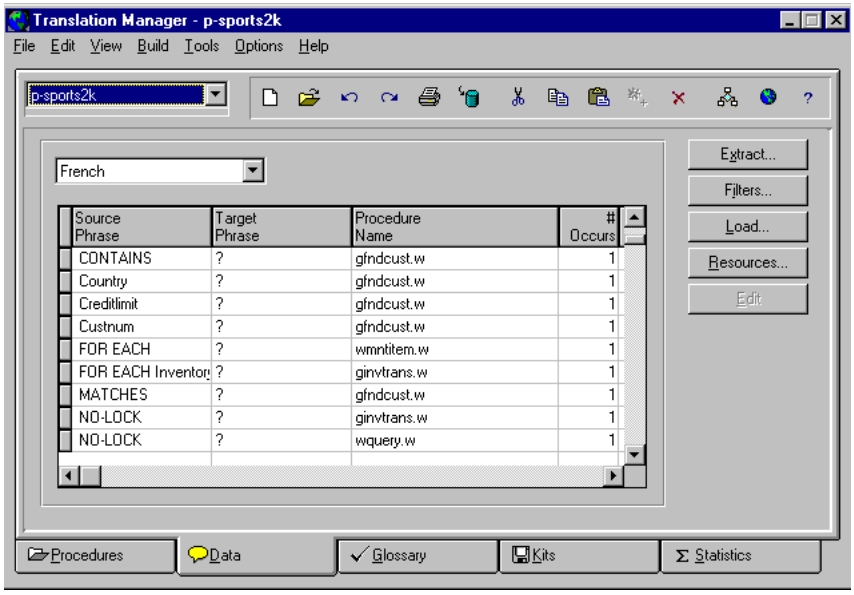


Figure A–2: Data Tab Folder

The Data tab folder contains the following user-interface elements:

- **Language Combo Box** — Displays the name of the current target language.
- **Browse Viewer:**
  - **Source Phrase** — Displays the text phrase in the original, source language.
  - **Target Phrase** — Displays the text phrase in the current target language.
  - **Procedure Name** — Displays the name of the procedure file that contains the text phrase.
  - **# Occurs** — Displays the number of times the text phrase occurs in the procedure file.

- **Line Number** — Indicates the line numbers on which the text phrase occurs in the procedure file or its associated include (.i) file. Displays the unknown value (?) if the text phrase occurs on more than one line number in the procedure file.
- **Justification** — Provides the justification format for the text phrase in the user interface: Center, Right, Left, or Trim.
- **Length** — Provides the number of characters allocated for the text phrase.
- **Object Name** — Indicates the type of user-interface object that contains the text phrase.
- **Statement** — Indicates the Progress 4GL statement associated with the text phrase.
- **Item** — Provides a Progress 4GL keyword that clarifies the Progress 4GL statement.
- **Comments** — Lets you enter comments about the text phrase for your own use. The translators will not see these comments; they can only view the comments you write in the Procedures tab folder.

NOTE: You can double-click on any field to display the Long String Translator dialog box. You can view the source string text in the Source String viewer. You can view and edit target string text in the Target String viewer.

- Action Buttons:
  - **Extract** — Displays the Extract dialog box, which lets you extract text phrases from the source procedures into a temporary STRING-XREF file.
  - **Filters** — Displays the Filter Wizard, which lets you specify filters where you load the text phrases from the STRING-XREF file to the project database. The Filter Wizard contains fourteen panels. The first ten let you specify the 4GL text phrases you want to include. Panels eleven, twelve, and thirteen let you specify words you want to exclude. The final panel provides a summary of the filters you specified.
  - **Load** — Displays the Load dialog box, which lets you load the text phrases from the STRING-XREF file, through the filters, into the project database.
  - **Resources** — Displays the Resources dialog box, which lets you generate resource procedures that the translators will use to visualize the application as they translate the text phrases.

A.1.3 Glossary Tab Folder

Use the Glossary tab folder to add glossaries to the project database. See Chapter 6, "Adding Glossaries," for instructions on using this tab folder. Figure A–3 shows the Glossary tab folder.

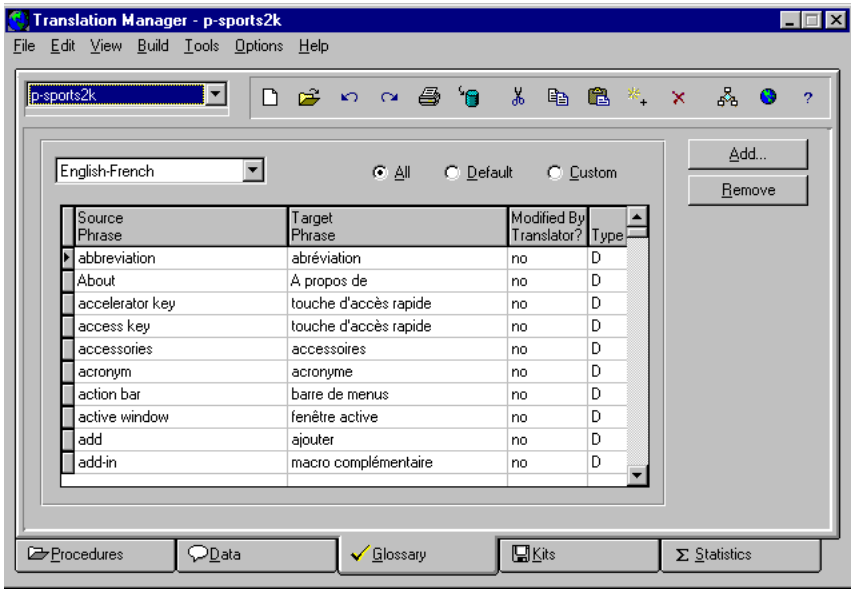


Figure A–3: Glossary Tab Folder

The Glossary tab folder contains the following user-interface elements:

- **Glossary combo box** — Displays the name of the current glossary.
- **All radio-set button** — Displays all the glossary entries.
- **Default radio-set button** — Displays only the glossary entries you entered with the Translation Manager tool.
- **Custom radio-set button** — Displays only the glossary entries entered by the translator with the Visual Translator tool.
- **Browse Viewer:**
  - **Source Phrase** — Displays the text phrase in the original, source language. You can edit this field.
  - **Target Phrase** — Displays the text phrase in the current target language. You can edit this field.

- **Modified by Translator?** — Displays yes if the translator altered a default glossary entry or if the translator added the entry; displays no if the translator did not alter the glossary entry.

- **Type** — Displays D (for default) if the entry was added with the Translation Manager tool; displays C (for custom) if the entry was added with the Visual Translator tool.

NOTE: You can double-click on any field to display the Long String Translator dialog box. You can view the source string text in the Source String viewer. You can view and edit translation text in the Target String viewer.

- **Action Buttons:**

  - **Add** — Displays the Add Glossary dialog box and lets you add a glossary table to the project database.

  - **Remove** — Deletes the entire selected glossary table from the project database.

A.1.4 Kits Tab Folder

Use the Kits tab folder to create kit databases and zip files. Also use it to unzip a kit you receive from the translators. Chapter 7, "Preparing a Kit," and Chapter 8, "Incorporating a Translated Kit into the Project," discuss how to use this folder. Figure A–4 shows the Kits tab folder.

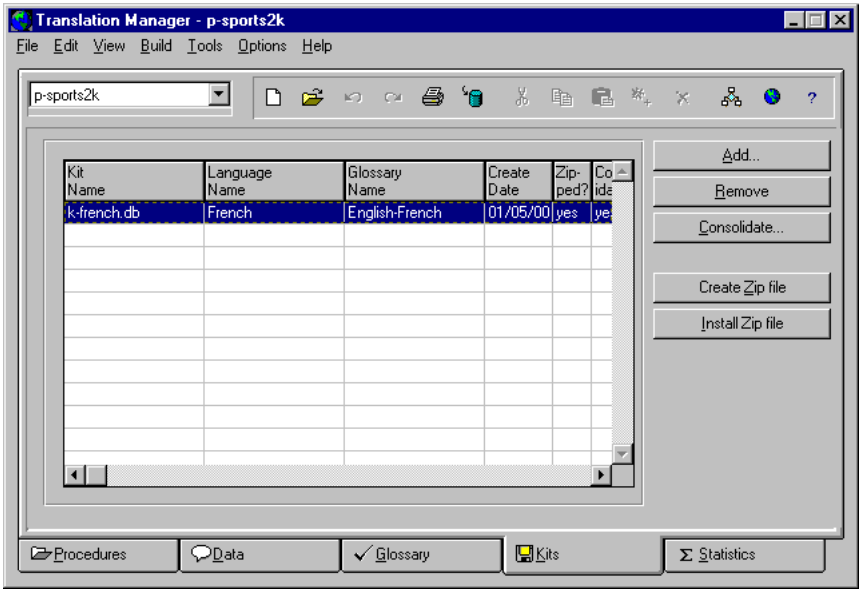


Figure A–4: Kits Tab Folder

The Kits tab folder contains the following user-interface elements:

- **Browse Viewer:**
  - **Kit Name** — Displays the name of the current kit.
  - **Language Name** — Displays the name of the current target language.
  - **Glossary Name** — Displays the name of the current glossary.
  - **Create Date** — Indicates the date you created the kit database.
  - **Zipped?** — Indicates whether you created a zip file for the kit.
  - **Consolidated?** — Indicates whether the kit has come back from the translator and you have consolidated it into the project database.
  - **Phrases Translated?** — Indicates the total number of phrases translated in the kit.
  - **Percent Translated** — Indicates the percentage of text phrases translated in the kit.

- **Action Buttons:**
  - **Add** — Displays the Add Kit dialog box and lets you add a kit to the project database.
  - **Remove** — Deletes the selected kit name from the project database. Also lets you delete the corresponding kit database.
  - **Consolidate** — Displays the Consolidate dialog box and lets you consolidate the translated text phrases and custom glossary entries into the project database.
  - **Create Zip file** — Displays the Create Zip File dialog box and lets you compress the kit into a zip file.
  - **Install Zip file** — Displays the Install Zip File dialog box and lets you decompress a kit zip file returned from the translator.

A.1.5 Statistics Tab Folder

Use the Statistics tab folder to view the statistics for the current project database. Figure A–5 shows the Statistics tab folder.

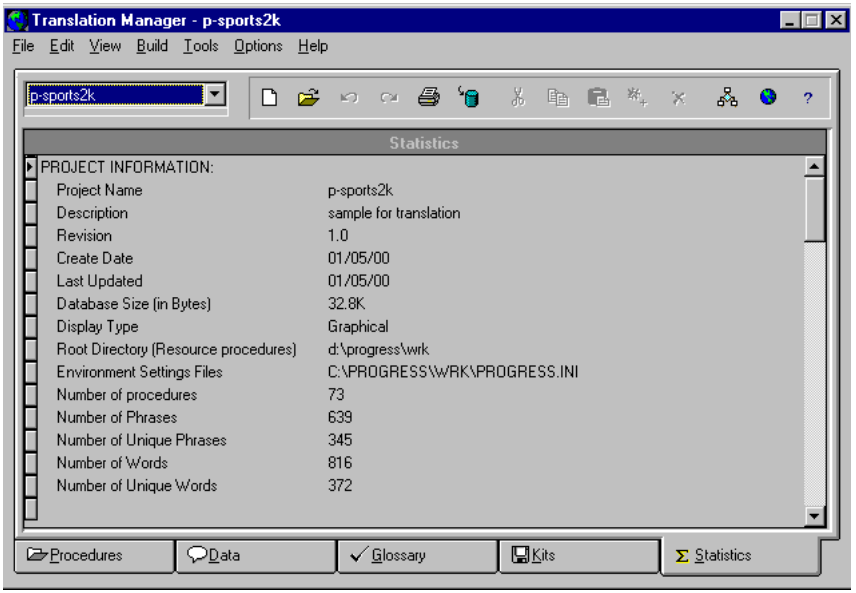


Figure A–5: Statistics Tab Folder

The Statistics tab folder contains information divided into sections for the current project, the kits in the current project, and any glossaries in the current project that are not part of a kit. You can generate a hard-copy report or send the report to a text file. For more information on the Statistics report, see Chapter 9, "Updating a Project."

A.2 Menu Bar

The Translation Manager tool's menu bar consists of the following menu options:

- File
- Edit
- View
- Build
- Tools
- Options
- Help

For more information on menu options, select Help→ Help Topics→ Translation Manager: Interface Reference→ Menu Commands.

A.3 Toolbar

The toolbar contains buttons that provide quick access to some of the most commonly used menu options. When a button or a menu option is grayed out, it is disabled and you cannot use it. Table A–1 provides a description of the toolbar buttons and lists their corresponding menu options and the tab folders in which they are enabled.

Table A–1: Toolbar Buttons *(1 of 3)*

Button	Menu Option	Description	Tab Folders
	File→ New	Creates a new project.	All
	File→ Open	Opens an existing project.	All

Table A–1: Toolbar Buttons *(2 of 3)*

Button	Menu Option	Description	Tab Folders
–	File→ Export	Exports a glossary.	Glossary
–	File→ Import	Imports a glossary.	Glossary
	File→ Print	Prints a screen capture of the interface or prints a hard-copy Statistics report.	All
	File→ Database Connections	Connects or disconnects from a Progress database.	All
	Edit→ Cut	Cuts selected text.	Procedures, Data, Glossary
	Edit→ Copy	Copies selected text.	Procedures, Data, Glossary
	Edit→ Paste	Pastes text you cut or copied.	Procedures, Data, Glossary
	Edit→ Insert	Inserts a row.	Glossary
	Edit→ Delete	Deletes a row.	Data, Glossary
	Build→ Compile	Compiles source code.	All

Table A–1: Toolbar Buttons *(3 of 3)*

Button	Menu Option	Description	Tab Folders
–	Tools→ Visual Translator	Opens the Visual Translator tool.	All
?	–	Provides online help about buttons and folders.	All

Glossary

Accelerator Keys

Key combinations you press to select a menu item.

Application Database

A database containing application schema and perhaps some data. You must connect to the application databases when you compile the source code, whether or not you include translations.

Character Set

The set of characters you use to read and write a language.

Code Page

A mapping of characters and symbols to internal numeric values.

Compressed File

A file that contains compressed data. You usually compress a kit to send to the translator. Also known as a zip file because it has a .zip extension.

Custom Glossary Entry

A glossary entry that the translator adds to a kit.

Default Glossary Entry

A glossary entry provided by the project manager that contains a list of suggested pairings from which the translator can choose.

Filter

The 4GL code that the Translation Manager uses to determine the subset of text extracted from the source procedures to load into the project database. The first ten panels in the Filter Wizard let you specify *include filters*-filters that determine the 4GL text phrases you want to place in the project database for translation. The last two panels let you specify

exclude filters-Progress keywords or words you specify that you do not want to translate.

Folder

The main interface component of the Translation Manager and Visual Translator tools.

4GL

See **Progress 4GL**.

Glossary

A list of the text-phrase pairings for two spoken languages, for example German and French. A kit usually includes two types of glossary entries: Default and Custom. The translator uses the pairings in the glossary to translate text phrases.

In-context Translation

Translates text phrases as you see them in the context of the user-interface of a procedure. Also known as screen-by-screen translation.

Folder Translation

Translates a single text phrase from a list of text phrases.

Kit

Contains a kit database, resource procedures, an environment (.ini) file, and image files. You create a kit for each target language. You create a language kit with the Translation Manager tool.

Kit Database

A Progress database that contains text phrases and a default glossary with recommended pairings.

Language Family

Consists of the languages that use the same character set.

Menu Bar

Contains menu options that allow you to perform various tasks.

Privileges

Permissions you assign to the translator.

Progress 4GL

The fourth-generation programming language in which source code procedures are written.

Project

See **Project Database**.

Project Database

The largest container unit in the Translation Manager tool. A database that contains the elements required to translate an entire software application into one or more spoken languages. It includes a list of source procedures, text phrases, glossaries, and one or more language kits.

Resource Procedures

Shells of the original source procedures that do not contain any functionality. For example, if you choose a delete button on a resource procedure, you will not delete anything. The translator uses the resource procedures to translate text phrases in the context of the user interface.

Source Language

The language in which the source procedure is written.

Source Procedures

The Progress 4GL application procedures that contain the text phrases to be translated.

Tab

The user-interface component that opens a tab folder. For example, you choose the Procedures tab to open the Procedures tab folder.

Target Language

The language into which the translator translates the source-code text phrases. The target language does not have to be a spoken language; it can be an alternate language of business or technical terms.

Text Phrase

One or more characters enclosed in quotes in a Progress 4GL application.

Text Segment Growth

The process of expanding source code text segment allocation at compile time so that translations can fit in their text segments without truncating. Text segment growth is important when you translate a compact language, such as English, into other less compact languages. For example, the English word “file” is shorter than the French translation “fichier.”

Toolbar

Contains buttons that provide quick access to some of the most commonly used menu options.

TRANMAN 2.0

See Translation Management System 2.0.

Translation Data

The resource procedures, text phrases, and user-interface objects to be translated.

Translation Management System 2.0

The Progress Version 9 Translation Management System (TRANMAN 2.0) consists of two tools-the Translation Manager and the Visual Translator.

Translation Manager

A Progress application tool that lets you manage the entire process of translating the text phrases in Progress applications. This tool works in partnership with the Progress Visual Translator tool.

Translation Project Manager

The person who oversees the entire process of translating a company's software applications.

Translator

The person who uses the Visual Translator tool to translate text phrases in user-interface procedures.

Visual Translator

A Progress application tool that lets you translate text phrases from procedures into one or more spoken languages. This tool works in partnership with the Progress Translation Manager tool.

Index

A

Accelerator keys
  defining 1–4
  translating 1–3
Adding
  glossaries 6–2
  kits 7–2
  source procedures 4–8
Additional Progress information 1–10
Application databases 3–6
ASCII 4–3
Attributes
  :T 8–6
  :U 1–3, 5–3
Audience xi

B

Blanks
  trimming 7–5
Bold typeface
  as typographical convention xiii
Buttons A–11

C

Changes
  saving 4–8
  updating 9–4
Character applications 5–17
Character sets 4–3
Closing projects 4–8
Code pages 4–3
  defaults 6–6
Columns
  ordering 9–3
Comma-delimited glossaries 6–4
Comments
  entering A–3, A–6
Compilation log file 8–5
Compiling code 8–4
Compressed kit files 7–9
Connecting to databases 3–6
Consolidating kits 8–2
Copying
  existing projects 4–5
  text 9–3
Creating
  glossaries 6–2
  kits 7–2
  projects 4–5
  zip files 7–9

Current directory 3–3
CURRENT-LANGUAGE
statement 8–9
Custom glossary entries 6–8
Cutting text 9–3

D

Data tab folder A–5
Databases
  connecting 3–6
  disconnecting 4–8
  project 4–2
Default buttons
  Windows 1–4
Default glossary entries 6–8
Defining filters 5–6
Deleting files 4–12
Dialog boxes
  Add Glossary 6–2
  Add Kit 7–2
  Add Procedures 4–8
  Compile 8–4
  Consolidate 8–2
  Create Zip File 7–9
  Database Connection 3–6
  Extract 5–4
  File Scan 9–5
  Get Directory 4–9
  Import 6–5
  Install Zip File 8–2
  Load 5–10
  Long String Translator 5–11
  New Project 4–5
  Resolve Conflict 8–3
  Resources 5–13
  Run 8–8
  Save As 6–8
Differences in layout 1–2
Directories
  source code 4–8
  translated r-code 3–2
Directory structure
  recommended 3–2
Disconnecting databases 4–8
Double-byte character sets 4–3

E

Editing
  a source file 4–13
  text phrases 9–3
Environment files 3–3
Environment (.ini) files 3–4
Error messages 11–1
  displaying descriptions xix
  writing for translation 1–9
Example procedures xviii
Exceptions Report 7–5
Exclude filters 5–6
Exporting data strings 5–18
Exporting glossaries 6–8
Extract options 5–4
Extracting
  batch mode 5–5
Extracting text phrases 5–2

F

Files
  adding to a project 4–8
  Compilation log 8–5
  image 5–14
  importing 6–5
  include 11–2
  pretrans.log 7–6
  removing 4–12
  scanning 9–5
  STRING-XREF 5–2
  viewing contents 4–13
  zip 7–9
Filter Wizard
  see also Filters
Filters
  importing and exporting 5–7
  redefining 5–11
  specifying 5–6
Fonts 1–6, 11–3

G

Glossaries 2–5
  consolidating 8–2
  copying 6–3
  creating 6–2
  custom compared to default 6–8
  defined 6–1
  example entries 6–8
  exporting 6–8
  importing 6–4
  Microsoft 6–2
  naming conventions 6–2
  updating 8–2
Glossaries tab folder A–7
Growth table 8–7

H

Help
  Progress messages xix

I

Image files 5–14
Importing data strings 5–18
Importing glossary entries 6–4
Include files 11–2
Include filters 5–6
Information sources 1–10
Initial values 5–3
Installing
  the Translation Manager 3–5
  zip files 8–2
Internal growth table 8–7
Italic typeface
  as typographical convention xiii
Items A–6

J

Justification of text phrases A–5

K

Keystrokes xiii
Keywords 5–8
Kits
  adding 7–2
  changing source language 7–7
  consolidating into the project 8–2
  contents 7–1
  definition 7–2
  deleting 7–3
  naming conventions 7–2
  receiving from translator 8–2
  related to projects 2–6
  replacing 7–3
  updating 7–3
Kits tab folder A–8

L

Language
  changing a kit's 7–7
Language families 4–3
Languages 4–2
Layout
  considerations 1–4
  problems 1–2
Length of text phrases A–5
  compared to width 1–8
Line numbers A–5
Load dialog box 5–10
Loading text phrases 5–8

M

Main window 3–5
Manual
  syntax notation xiv
Manual, organization of xi
Marked procedures 5–4
Menu bar A–11
Merging kits into the project 8–2
Message statements
  writing for translation 1–9

Messages
displaying descriptions xix
suppressing 4–11
Microsoft language glossaries 2–5, 6–2
Monospaced typeface
as typographical convention xiii
Multi-byte character sets 4–3

N

Naming conventions
glossaries 6–2
kits 7–2
projects 4–5
New projects 4–5
Null strings 5–3
Number of users 2–2

O

Object names A–6
Opening
projects 4–7
tab folders A–2
Options
for extracting 5–4
Ordering columns 9–3

P

Pasting text 9–3
Permissions for translator 4–6
Phrases
see also Text phrases
Planning ahead 1–2
Position of widgets 1–4
Pretranslation 7–4
pretrans.log file 7–6
Print option A–11
Printing a pretranslation exceptions report 7–4
Privileges for translator 4–6
Problems 11–1
avoiding 1–2
Procedures
adding 4–9
examples of xviii
removing 4–12
Procedures tab folder A–3
Progress keywords 5–8
progress.ini file 3–3
locations 5–13
Project directory 3–2
Project manager
responsibilities 2–2
tasks 2–5
Projects
closing 4–8
copying 4–5
creating 4–5
defined 4–2
determining how many 4–3
opening 4–7
related to kits 2–6
selecting 4–7
updating 9–4
PROPATH setting 3–3
PRO*Tools
accessing 3–3

Q

Question mark
unknown value 6–3

R

R-code
compiling 8–4
running 8–8
testing 8–9
Rectangles
adjusting text 1–8
Redefining filters 5–11
Removing procedure files 4–12
Reports
pretranslation exceptions 7–4

Resource procedures 2–5
compared to source procedures 5–12
creating 5–12
Root directory 3–3
Row markers 4–12
Rows
sorting 9–3
Run dialog box 8–8
Running translated code 8–8

S

Saving changes 4–8
Scanning for changes 9–4
Scrollbars 9–3
Selecting procedure files 4–12
Selecting the current project 4–7
Setting filters 5–6
Setting your PROPATH 3–3
Setup 3–1
Single-byte character sets 4–3
Single-user mode 2–2
Size of widgets 1–4
SmartObjects 4–8
Sorting rows 9–3
Source code
compiling 8–4
directory structure 3–2
updating 9–4
Source code directory 3–3
Source language
changing a kit's 7–7
Source languages 4–2
Source procedures
compared to resource procedures 5–12
selecting for a project 4–8
viewing and editing 4–13
Space-delimited glossaries 6–4
Specifying filters 5–6
Starting the Translation Manager 3–5
Statistics
suppressing 4–11
Statistics tab folder A–10
String attributes
:T 8–6
:U 1–3, 5–3
Strings
see also Text phrases
STRING-XREF file 5–2
deleting 5–10
example 5–3
location 5–4
Subsets 5–14
Supersets 5–15
Suppressing messages 4–11
Syntax notation xiv

T

Tab folders A–2
browsing 9–3
Target languages 4–2
Tasks
for project manager 2–5
for translator 2–7
Testing r-code 8–8
Text expansion 8–6
examples 8–7
planning ahead 1–6
Text in rectangles 1–8
Text phrases
editing 9–3
expansion 8–6
extracting from source procedures 5–2
filtering 5–6
length compared to width 1–8
loading into project 5–8
translating 2–3
viewing in Data tab folder 5–11
Text segment growth 8–6
Third-party translation tools 5–18
Tool bar A–11
TRANMAN 2.0 3–5
Translated kits
consolidating into the project 8–2
receiving 8–2

Translated r-code directory 3–2
Translating text phrases
  preliminary translation 7–4
Translation Manager 2–2
  starting 3–5
Translation process 2–3
Translator
  defined 2–2
  tasks 2–7
Translator privileges 4–6
Trim string attribute (
  T) 8–6
Trimming blanks 7–5
Troubleshooting 11–1
Typographical conventions xiii

U

UNIX-based applications 5–17
Untranslatable attribute (
  U) 1–3, 5–3
Unzipping kits 8–2
Updating kits 7–3
Updating projects 9–4
Upgrades
  Version 8 to Version 9 3–2
User-interface problems 1–2, 1–4
Users 2–2

V

Viewing a source file 4–13
Visual Translator 2–2
  translator's tasks 2–7

W

Widgets
  sizing and positioning 1–4
Width of text phrases
  compared to length 1–8
Wizard 5–6
Working directory 3–3

Z

Zip files 7–9