PP

Volume 2: Programmer's Guide

Julian Onions
X-Tel Services Ltd

December 19, 1991
(Version 6.0)

Contents

List of Tables

List of Figures

Preface

This volume describes the PP programming interface. This covers:

- Submission of messages such as might be used by a user agent.
- Submission of messages as by a protocol channel.
- Writing channel programs.

It also explains the major data structures used in PP and the conventions such as logging and initialisation.

Chapter 1

General

1.1 Overview

There are certain approaches taken by PP processes. These are described here to facilitate emulation by new processes. These are not strict guidelines, but following existing practises will usually be easier and make upgrading the software less painful.

1.2 Initialisation

All PP programs generally start by calling one or more initialisation routines. This may vary depending on what the channel needs to know. This initialisation does various things, amongst which are:

- Reading global parameters from the tailor file.
- Initialising the logging packages.
- Handling of signals.

The normal method of initialising a process is to call the following routine very early on (usually one of the first things):

```
sys_init (name)
char *name;
```

This routine does most of the initialisation required. The argument is usually the value of `argv[0]`, which is used in the logging initialisation to record the name of this program.

For some programs however, this sort of initialisation is not appropriate. In this case another interface may be better.

```
int pp_initialise (name, flags)
char *name;
int flags;
```

This routine will read in the tailor file, but will optionally not configure certain aspects depending on the flags. The name is the same as in `sys_init`. The flags can be any of the following:

Flag	Effect
`TAI_NONE`	No special tailoring is done.
`TAI_LOGS`	Reset all logs to use the PP log directory.
`TAI_SIGNALS`	Set handlers for certain signals.
`TAI_ALL`	All of the above.

The flags may be combined by bitwise the bitwise or operator.

1.3 Runtime Tailoring

The runtime tailoring for PP is complex, as there are many parameters that can affect the operation of the system. The main things tailored are pathnames to locate commonly used programs; file names and directories; or the more complex tailoring to define such things as channels and tables. See Volume 1 for the full detail.

To add new items to be tailored, it is necessary to add some code to the parsing routines. For simple additions, it is necessary only to modify the tables and the switch statement in the file *pp/Lib/tai/tai_sys.c*. More complex tailoring is best done by creating a separate procedure, possible in a new file. Such additional tailoring should not be undertaken lightly as it will make the new version of PP incompatable with existing and new versions.

1.4 Logging

All logging works through a standard package. This package includes features to limit the log files and to notify administrators of errors. The logging has some structure too it and the following are the general rules used when programming PP.

1.4.1 Logging Levels

The logging package takes a bit mask of values to log. This is compared with the current setting of the logging level and then certain values are logged. The levels available are shown in Table 1.1. The values `LLOG_NONE` and `LLOG_ALL` are special: the first is no logging at all, the latter sets all bits in the bitmask.

1.4.2 Logging Structures

There are three log structures used in PP. The structures themselves are defined in the include file *isode/logger.h*. These structures are of type `LLog *` and are the following: