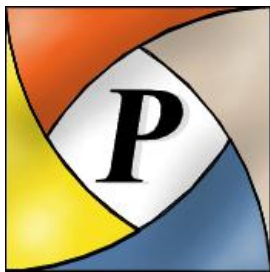




Institute for Energy Technology
OECD Halden Reactor Project



ProcSee

Graphical User Interface Management System

Plugins

Reference Manual

1.0



This document will be subjected to revisions in the future as the development of ProcSee continues. New versions will be issued at new releases of the ProcSee system.

The information in this document is subject to change without notice and should not be construed as a commitment by Institute for Energy Technology.

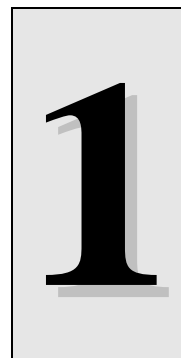
Institute for Energy Technology, OECD Halden Reactor Project, assumes no responsibility for any errors that may appear in this document.

Published by : Institute for Energy Technology, OECD Halden Reactor Project
Date : June 2010
Revision : 1.0



Contents

Chapter 1: Introduction	6
About ProcSee Plugins	6
About this manual.....	6
Using Plugins	7
Chapter 2: Plugins	8
OPC Plugin.....	8
PlaySound Plugin	9



Introduction

About ProcSee Plugins

The ProcSee Run Time Manager (RTM) can be expanded with plugins. By using plugins, platform specific functionality can be provided without including platform specific code in the RTM. Plugins can also be used to provide functionality that is only relevant for specific applications.

At the moment the plugin interface, i.e. the software interface between any plugin and the RTM, is undocumented and subject to change. However, plugins developed by the ProcSee team will be released as part of ordinary ProcSee releases.

About this manual

This manual describes the ProcSee plugins available at the moment. Additional plugins may be added in later versions.

Using Plugins

To use the functionality provided by a plugin, the plugin must be loaded into the RTM. This is done by adding the plugin to the application or library where the plugin is needed.

On the Microsoft Windows platform, plugins are loaded by right clicking the application or library and selecting **Plugins...** in the context menu. In the window that opens, select the plugin in the list of available plugins, and click on the load button, ref Figure 1. The plugin will then be displayed in the list of loaded plugins.

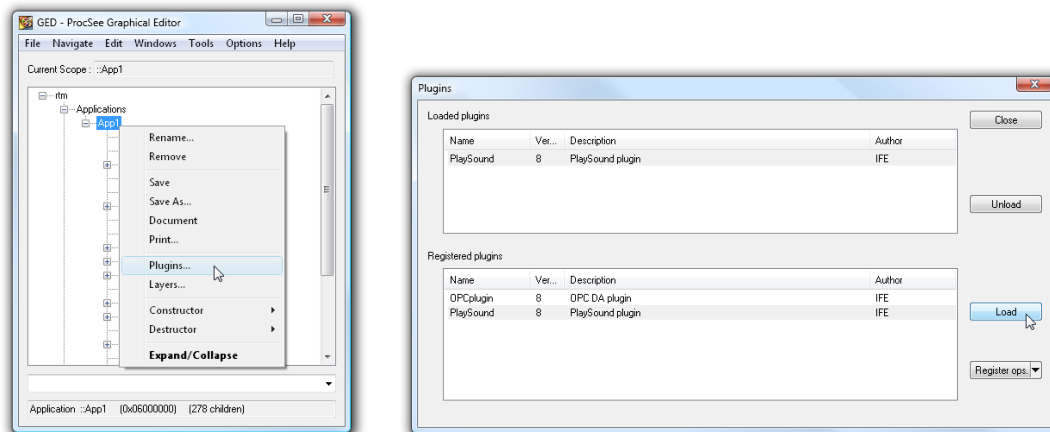


Figure 1 Loading a plugin in an application using GED.

On the other platforms, the addition of plugins must be done by editing the application or library .Tdoc file.

In the .Tdoc file, the use of plugins is indicated by the keyword **plugin** followed by the name of the plugin, ref Figure 2. The plugin must be located in the **plugins/\$(ARCH)** directory of the ProcSee installation, and be registered in the **plugins.pprd** file. The plugins.pprd file contains information about the plugins available, and what version of the plugin-interface they use. The version number of the plugin interface of the plugin must match the version number expected by the RTM.

```

application App1
{
    plugin "PlaySound";
    ...
}

```

Figure 2 Including a plugin in an application, .Tdoc syntax.

Plugins are loaded together with an application or library, and normally register functions that can be called directly from pTalk.



Plugins

OPC Plugin

ProcSee's OPC plugin enables the RTM to act as an OPC Data Access client. The plugin uses OPC Data Access 2.0, allowing the RTM to access any server compliant with the OPC Data Access 2.0 specification.

The OPC plugin is available in the Microsoft Windows version only.

The OPC plugin is documented in a separate document: [OPCpluginConfig.pdf](#).

PlaySound Plugin

The PlaySound plugin is a very simple plugin, which makes the Microsoft Windows Multimedia API function PlaySound available to pTalk. This is for playing simple event sounds. At the moment this plugin is available only on Windows, since the PlaySound function is a Windows function.

The PlaySound plugin makes the following two functions available to pTalk:

```
int playSound( char* fileName, char* options=0 );
int stopSound();
```

The **playSound** function plays a **.wav** sound file. The sound-file is played asynchronously by default.

The *fileName* is either an absolute file-name, or a file-name relative to the application that it is called from, or a name of some of the event sounds in Windows, like "SystemAsterisk" (Check the Windows documentation of PlaySound for details).

The *options* argument can contain:

- **loop** to make the sound play repeatedly. To stop the playback, use the stopSound function or play another sound.
- **sync** to make the playSound function not return until the sound has been played (the return value is then 0 if there was any errors, in other cases the function returns 1).
- **async** to make the playSound return immediately (before the sound is played). This is the default.
- **nodefault** to make the system not play a default sound if the fileName is not found.
- **alias** to only look for system event sounds.
- **filename** to only look for sound-files.

The **stopSound** function stops the currently playing sound.

Example of usage:

```
playSound( "sounds/tada.wav" ); // plays the wave file.
playSound( "SystemAsterisk", "loop" ); // plays the event sound.
stopSound();
```