Ada Information Server (AIS)

Overview

The Ada Information Server is a utility that facilitates the enhancement of simple editors to be become more Ada aware.

The problem with simple editors is that they only provide limited macro and scripting facilities that by themselves are insufficient to process the complex syntax and semantics found in Ada source code.

To workaround this limitation, the Ada Information Server does the work as a separate process. An editor or programming environment merely has to submit its request to the server and to process the resultant output.

As a demonstration of this technique the following describes how to enhance the popular simple editor UltraEdit such that pressing a specific key combination whilst the cursor is positioned within an Ada source file will attempt to determine where the symbol at the current cursor position was defined.

If the reference can be resolved, AIS will return the file name along with the row and column.

The interface to AIS is through a duplex named pipe which is specified when AIS is started as its first command line parameter with the an optional second parameter denoting the pipe length which specifies the longest possible command that can be sent to the server or reply received from the server. If not supplied it defaults to 1000 bytes which is adequate for most purposes unless exceeding long path names are used.

For example:

```
C:\Tools\Ais_95 Ais_Pipe 2000
```

Would start the AIS server for Ada 95 located in the folder C:\Tools and use the named pipe Ais_Pipe to communicate with its client with a maximum transfer size of 2000 bytes.

The Client/Server protocol is described at the end of this document. However in order to promote the use of AIS, White Elephant GmbH has written a sample client which is provided as part of this project.

Configuring UltraEdit v10 for AIS using the supplied client Ais_Client.

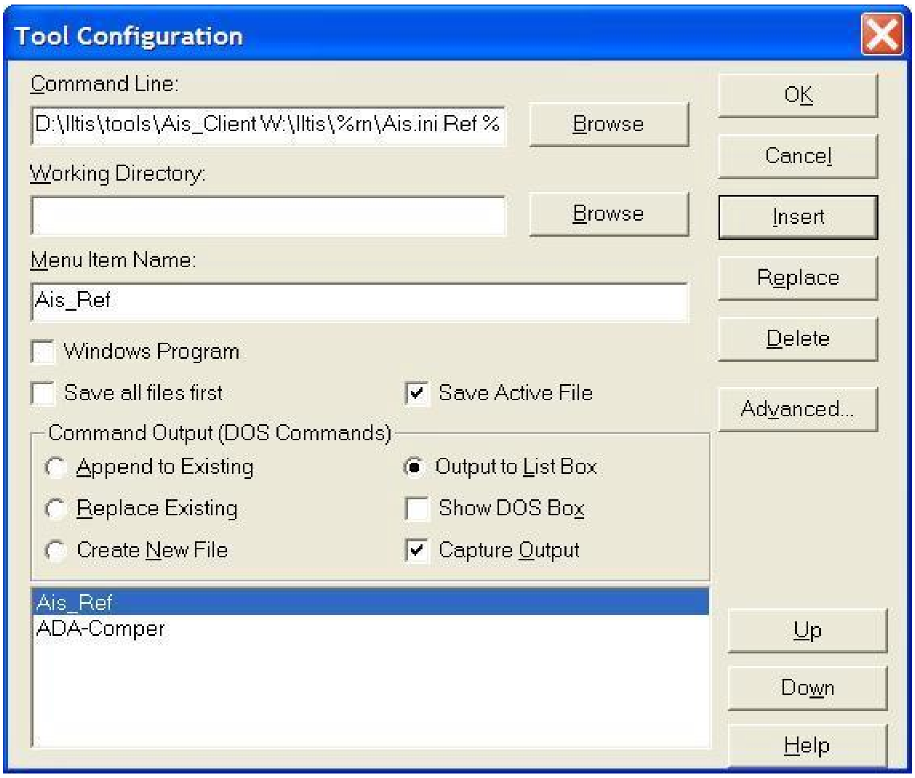
Within UltraEdit define a tool using "*Tool configuration*" from the *Advanced* menu item.
In theCommand Line window enter something similar to

D:\Iltis\tools\Ais_Client W:\Iltis\%rn\Ais.ini Ref %f %line% %col%

to call the Ais_Client with five parameters:

1. The name of the configuration file.
   Typically this will be project specific, hence the use of Ultra edit projects and %fn in the path name of the configuration file.
2. The keyword *Reference*
3. The name of the file to be analyzed. %f% indicates the file currently being edited
4. The line number containing the symbol to be referenced. %line% resolves to the line number the current cursor position.
5. A column position that identifies the symbol to be referenced. %col% resolves to the column number of the current cursor position.

Next give the tool a name (Eg. Ais_Ref) then select *Save Active File*, *Output to List Box* and *Capture Output* options and press the *Insert* button to save the tool definition.



Command Line: D:\Iltis\tools\Ais_Client W:\Iltis\%rn\Ais.ini Ref %

Working Directory:

Menu Item Name: Ais_Ref

☐ Windows Program

☐ Save all files first ☑ Save Active File

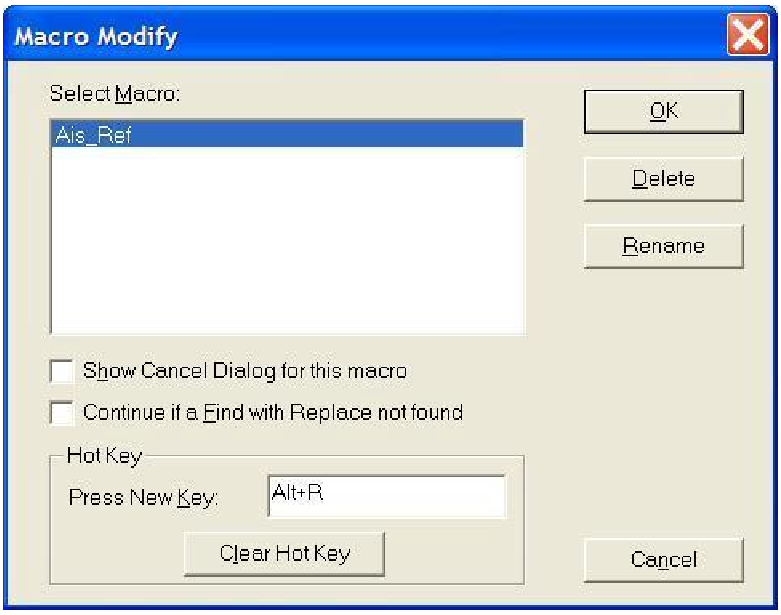
○ Append to Existing ● Output to List Box

○ Replace Existing ☐ Show DOS Box

○ Create New File ☑ Capture Output

Ais_Ref
ADA-Comper

Next a macro should be defined using *Edit Macro…* from the *Macro* menu item.
Press the *New Macro* button and in the resultant pop-up give the macro a name, disable *Show Cancel Dialog*, bind it to a hot key (for example Alt-R) and then press the *OK* button



Select Macro: Ais_Ref

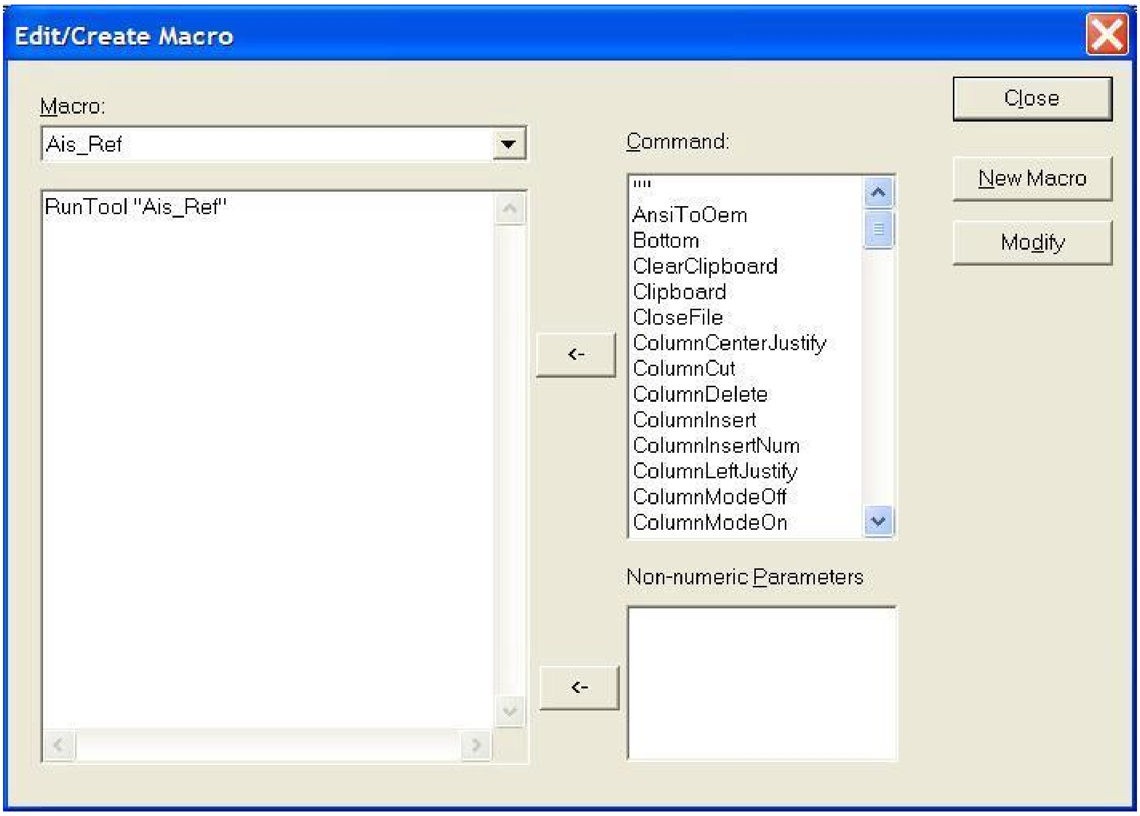
☐ Show Cancel Dialog for this macro

☐ Continue if a Find with Replace not found

Hot Key

Press New Key: Alt+R

Then in the Edit/Create Macro window edit the macro to run the previously defined tool



Macro: Ais_Ref

```
RunTool "Ais_Ref"
```

Press the *Close* button then save the macro definitions by selecting *Save All* from the *Macro* menu item and then cause these to be loaded on startup by using *Set Auto Load…* also from the *Macro* menu item.

Configuring the sample client Ais_Client

The configuration file supplied to Ais_Client is a standard windows ini file. If only the filename and extension is provided the file is assumed to be either within the normal search path of windows initialization files or in the same folder as the client executable.

The structure of the file consists of a single section called Ais_Client followed by a number of keywords assigned a value.

For Example:

```
[Ais_Client]
Server=D:\Iltis\tools\Ais_95.exe
Pipename=Ais_Pipe
Pipelength=2000
Timeout=60
Compiler=Aonix
Path=W:\Iltis\Mx1\Src;T:\Src;B:\Libraries\Iltis\Src;B:\Libraries\Bindings\Src;S:\Aonix;
Errors=1
Units=Matching
Editor=D:\iltis\tools\ultraedit\uedit32.exe
Format=%f/%r/%c
```

Keyword definitions

- Server - Full file specification of the AIS server to use.
- Pipename - Name of the Windows named pipe to be used in Client/Server communication.
- Pipelength - Maximum transfer size using the named pipe. If not provided a default length of 1000 is used.
- Timeout - Number of minutes the AIS server should remain active after being started or following a service request. Maximum 2'184 minutes.
  If not specified then the server will remain active until explicitly closed by the client. A value of zero specifies that the server should not remain active after use, however this value is not recommended.
- Compiler - Specifies the compiler being used.
  This indicates which compiler extensions should be recognized.
  - Aonix
  - Gnat
- Path - Details the search path the server should use to locate files.
  Directories are specified in the order they are to be searched, separated by semicolons (;). A directory specification terminated by * indicates all files in all nested subdirectories. For example D:\Ada\*
  Note: The last folder in the path should be the compiler specific directory (For Aonix, these files are available from the White Elephant web site)
- Errors - Maximum number of errors to display in the output window.
  If not provided only the first three errors will be reported
- Units - Defines how Ada packages are stored in files
  - Matching - Packages are stored one per file with the file name corresponding to the file name
  - Single - Packages are stored one per file however the file name may differ from the package name
  - Multiple - Standard Ada. There are no restrictions on how and where packages are stored.

  If not provided Matching Units is assumed.

Editor	-	The full file specification name of the editor to be called in response to a successful reference response. If not provided reference details will be sent to standard output.
Format	-	Text that specifies how AiS should format its replies. Composite symbols %f, %r and %c are replaced by the full file name, row and column respectively. If not specified then the default format %f/%r/%c is used.

Commands supported by the sample client Ais_Client

Key bindings may be defined for the following commands:

Reference - Request the definition of the symbol identified by three additional parameters: File, Row and Column

Location - Request a list of all the locations where the identified symbol has been referenced. The symbol identified by three additional parameters: File, Row and Column

Unused - Show all the declarations made in the file specified as the command's only parameter and all its dependencies that are unused. Performing this action on a main package will effectively show all the unused declarations for a specific program.

Close - Shutdown the AIS server

Syntax - Check the syntax of the specified file.

AIS Notes:

1. Only Units=Matching is currently implemented.
   Ie. For a package to be located it is assumed to be contained in a file of the same name. Specifications in files with extensions .ads, .spc or .dat and implementations in files with extensions .adb or .bdy.
   For files with the extension .ada if the filename ends with an underscore it is assumed to contain a specification otherwise an implementation.
   The dot notation for Child packages is replaced with a minus.
   For example the specification of Win32.Winbase is assumed to be contained in the file Win32-Winbase.ads
2. The message "Pipename not specified in configuration file" means that the definition of the pipename could not been found in the configuration file. This is often because it could not find the specified configuration file.
3. Subunits are not supported.
4. Files supplied by Aonix have been modified to conform with Units=Matching and these together with the package Standard.ads are provided as part of this project. These files should be copied into a folder and the folder included in the source search path.
5. Standard.ads defines specifications that are implicitly defined by the compiler and are therefore Aonix specific.
6. As error and diagnostic messages are sent to the output window, it is recommended that this be open for display when Ais is used.
7. Setting the sever timeout value too small will result in the server perhaps restarting unnecessarily and thereby having to redo a lot of preparation work that could otherwise have been avoided. On the other hand setting the value too high will cause the server to stay around long after it has stopped being used. As the server can potentially consume a lot of memory this might be undesirable. Setting the timeout value is therefore a very personal choice and depends very much on the way a particular person chooses to work.
   Note however that another key binding could be defined to issue the client close command. This could then be used to explicitly close the server without recourse to the Windows Task Manager.
8. Commands that result in a list of information use the program *file_index.exe* to display the information in a separate window. AIS expects to find this program in the same directory as itself.

AIS Protocol

- The AIS Client and Server communicate in ASCII over a duplex named pipe.
- When started, the server opens the pipe and waits for commands from the client.
- The server replies to each command with one or more messages.
- The last message for any particular command is a single character full stop ('. '= 16#2E#)
- The Client may only send further commands after it has received the full stop message.
- If the first character of a reply is an exclamation mark ('!' = 16#21#) the Server is requesting that it be shutdown – usually because of some terminal problem.
- If the first character of a reply is the character "greater than" ('>' = 16#3E#) this indicates that the editor should be invoked and the remainder of the line passed as a parameter
- Positions within files are indicated to AiS by two numbers: row and column, separated by a comma and the whole expression enclosed within angle brackets. For example: <10,5> for row 10 column 5

Recognised commands

Command	Parameters	Action
A	Filename	Show all unused declarations
C		Close the connection
E	Number	Define maximum number of errors to report
I	Editor	Parameter sent to the indexer program used to display lists. Currently this is the full file specification of the editor the indexer should use to display an item if it is double clicked.
L	Filename, Location	Locations where the symbol (found in file at row and column) is used.
P	String	Search path. List of directory names separated by ;
R	Filename, Location	Reference symbol found in file at row and column
S	Filename	Check syntax of named file.
T	Number	Define Server timeout in minutes
U	Keyword	Units. Defines how Ada packages are stored.
X	Keyword	Indicates what compiler specific extensions should be recognised.