

Could Enterprise Guide have been written in Java?

Author: Philip R Holland, Holland Numerics Ltd

Abstract

While the title of this paper may suggest a controversial topic, the intention is to review the functionality of Enterprise Guide®, the Windows-based thin-client front-end for SAS®, which uses COM/DCOM or the IOM Bridge to communicate with SAS® Integration Technologies. It is, however, possible to develop Java applications, which use the IOM Bridge to Java to communicate with SAS Integration Technologies, that can be run on any Java-compatible platform.

Comparing examples of the functionality available through the IOM Bridge for Java with the functionality of Enterprise Guide, it is hoped that this will explain how a thin-client front-end for SAS could have been written in Java, but also why Enterprise Guide itself was written by SAS to run only on the Windows platform.

Introduction

Having worked with Enterprise Guide since the release of Enterprise Guide version 1.1.1, the author has been able to watch how it has developed to become a very useful addition to the SAS product list. In particular, developing custom add-ins for Enterprise Guide version 2.0 has demonstrated its flexibility. However, the author is never happy with what he is given, and when he started looking at how external applications could communicate with SAS using Java through SAS Integration Technologies, he started to see some interesting parallels and, at the same time, realised that SAS were right to develop Enterprise Guide only for the Windows platform.

Comparison of Features: Enterprise Guide versus Java

The comparisons described below assume that you have the following software installed on the client SAS platform:

- Enterprise Guide version 2.0
- Microsoft Data Access Components release 2.7
- Client components for SAS Integration Technologies version 9.1
- Java JRE release 1.4.1

and on the server SAS platform:

- Base SAS and SAS Integration Technologies version 9.1

Platform Dependencies

Enterprise Guide has been developed using Microsoft Data Access Components and ActiveX DLL objects, which limit its installation to Windows platforms. The SAS installations it can communicate with are, however, not limited to Windows platforms, as Enterprise Guide can use a number of different methods of communication to access these installations, including COM for local SAS installations on Windows platforms, DCOM for remote SAS systems on Windows platforms, and the IOM Bridge for remote SAS systems on any platform.

If you wish to emulate the functionality of Enterprise Guide using a Java client, then your available platforms are only limited by the existence of Java environments for that platform. In this case there is currently only one method of communication with SAS Integration Technologies, which is the IOM Bridge for Java for accessing remote SAS systems on any platform. It should be noted that the Java client for SAS Integration Technologies in SAS version 9.1 requires the use of Java 2 Runtime Environment Standard Edition version 1.4.1, or above.

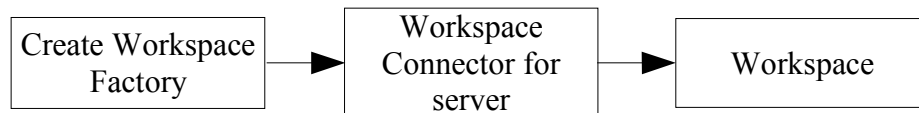
SAS Licensing Dependencies

While Enterprise Guide is limited to running only on Windows platforms, its available methods of

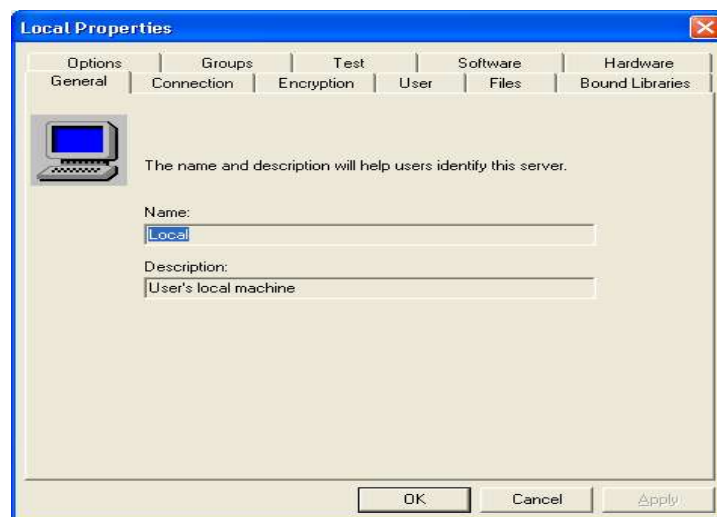
communication makes its SAS licensing less limited. DCOM and the IOM Bridge both require SAS Integration Technologies to be licensed on the remote SAS system before connections can be made to Enterprise Guide, but the COM connection to the locally installed SAS system can be made even if only Base SAS has been licensed.

In contrast, as a Java emulation of Enterprise Guide can only use the IOM Bridge for Java, it requires SAS Integration Technologies to be licensed on the remote SAS system before connections can be made.

Server Administration



The Enterprise Guide Administrator provides an interface to set up connection parameters for new servers and to update parameters for existing server connections.

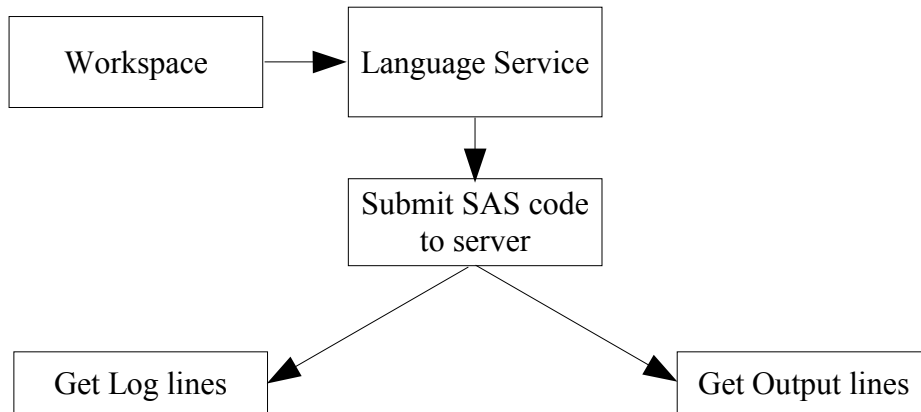


The following Java code demonstrates one method of connecting to a SAS workspace on a remote SAS server:

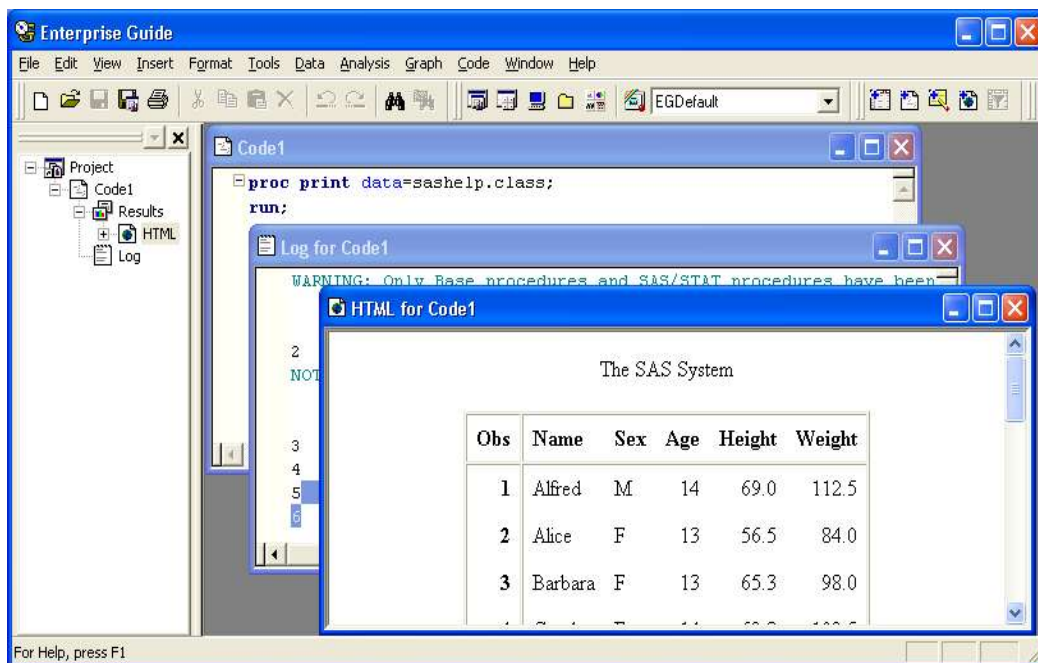
```
//set properties for server
Properties iomServerProperties = new Properties();
iomServerProperties.put("host", hostname);
iomServerProperties.put("port", portname);
iomServerProperties.put("userName", username);
iomServerProperties.put("password", password);
Properties[] serverList = {iomServerProperties};

//setup workspace factory
WorkspaceFactory wFactory = new WorkspaceFactory(serverList, null, null);
WorkspaceConnector connector = wFactory.getWorkspaceConnector(0L);
IWorkspace sasWorkspace = connector.getWorkspace();
```

Submitting SAS Code to the Server



Enterprise Guide provides Code, Log and Results nodes to hold the SAS program, Log and Output windows. The Code window can be used as a SAS code editor in the same way that the Enhanced Editor is used with the client SAS System.



The following Java code demonstrates how to submit a SAS program to a remote SAS server, and then retrieve the SAS Log and Listing files:

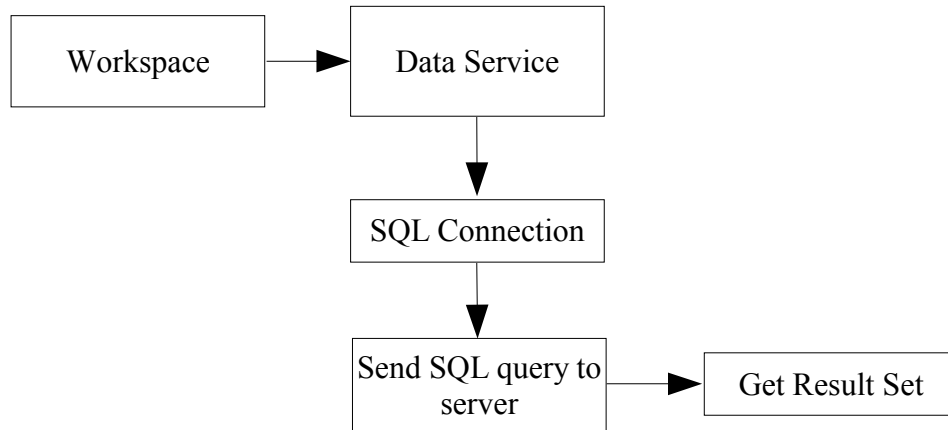
```
//use workspace factory to get reference to workspace stub
ILanguageService sasLanguage = sasWorkspace.LanguageService();

//submit SAS code to server
sasLanguage.Submit("proc means data=sashelp.class; class sex; var age height weight; run;");

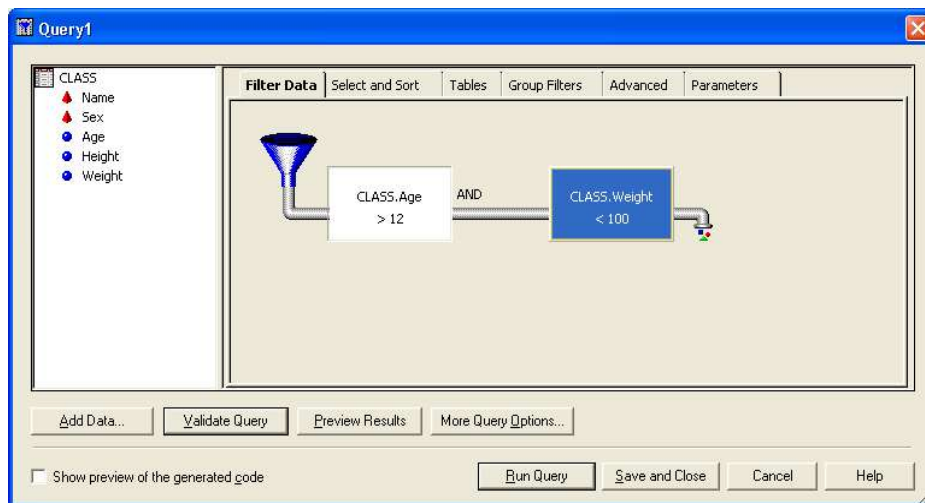
//get Log from server
CarriageControlSeqHolder logCCHldr = new CarriageControlSeqHolder();
LineTypeSeqHolder logLTHldr = new LineTypeSeqHolder();
StringSeqHolder logHldr = new StringSeqHolder();
sasLanguage.FlushLogLines(Integer.MAX_VALUE, logCCHldr, logLTHldr, logHldr);
String[] logLines = logHldr.value;
```

```
//get Listing from server
CarriageControlSeqHolder listCCHldr = new CarriageControlSeqHolder();
LineTypeSeqHolder lstLTHldr = new LineTypeSeqHolder();
StringSeqHolder lstHldr = new StringSeqHolder();
sasLanguage.FlushListLines(Integer.MAX_VALUE, listCCHldr, lstLTHldr, lstHldr);
String[] lstLines = lstHldr.value;
```

Reading SAS Tables from the Server



It is possible to open a SAS table directly by inserting the data into a Project using the Insert...Data menu option. However, SQL queries can also be built to select data from SAS tables which are then automatically added to the current Project.



It is not necessary to execute the generated query to find out whether any records have been selected, as the results can be previewed using the Preview Results button.

The screenshot shows the 'Query Results for "Query1"' window. It contains a table with the following data:

	Name	Sex	Age	Height	Weight
1	Alice	F	...	56.5	84
2	Barbara	F	...	65.3	98
3	Jeffrey	M	...	62.5	84
4	Judy	F	...	64.3	90

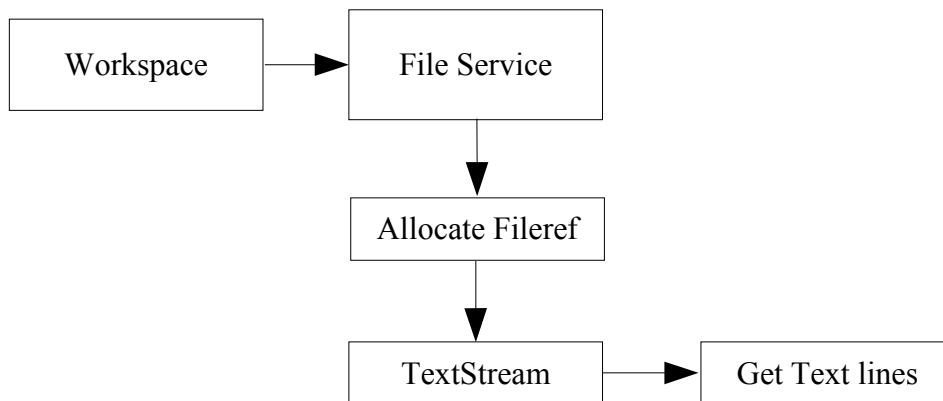
The following Java code demonstrates how to read a SAS table using SQL on a remote SAS server, and then copy the contents into a local store:

```
//get Dataset list
IDataService sasData = sasWorkspace.DataService();

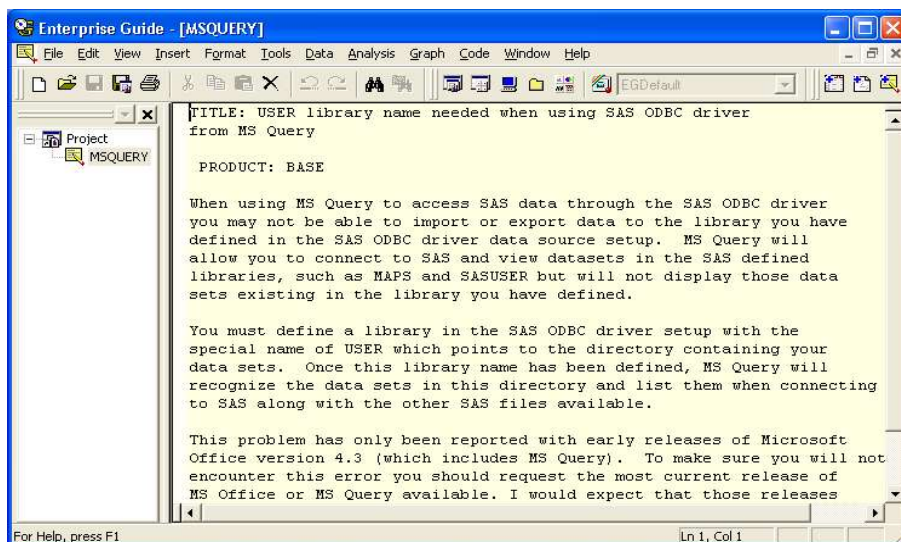
//get Resultset - open SQL connection
Connection sqlConnection = new MVAConnection(sasData, new Properties());
Statement sqlStatement = sqlConnection.createStatement
    (ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY);
ResultSet sqlResultSet = sqlStatement.executeQuery("SELECT c.*, r.* FROM
    sashelp.retail r, sashelp.class c WHERE r.year EQ 1982 AND c.name EQ
    'Thomas'");

//get the ResultSet MetaData. This can be used for the column headings
ResultSetMetaData sqlRSMetadata = sqlResultSet.getMetaData();
```

Reading Text Files from the Server



Enterprise Guide can insert a Note node into a Project using the Insert...Note menu option. The Note node can be any of a number of file types, including Text Files (*.txt, *.csv, *.asc, *.tab), Web Files (*.htm, *.html, *.htx, *.asp, *.alx, *.stm, *.shtml), Microsoft Word Document (*.doc), and Microsoft PowerPoint Presentation (*.ppt).



The following Java code demonstrates how to allocate a Fileref to a text file on a remote SAS server, and then copy the text from the file line by line:

```
//allocate File to new Fileref
IFileService sasFile = sasWorkspace.FileService();
StringHolder listHldr5 = new StringHolder();
IFileref sasRef = sasFile.AssignFileref("newfile", "DISK",
    "/home/phil/SAS/objspawn9.sh", "", listHldr5);

//read File
StreamOpenMode modeRef = StreamOpenMode.StreamOpenModeForReading;
ITextStream textRef = sasRef.OpenTextStream(modeRef, 1024);

//get lines until end of the file
String textLine = textRef.Read(132);
while(textLine != "") {
    textLine = textRef.Read(132);
}
```

Converting External Files to SAS Tables

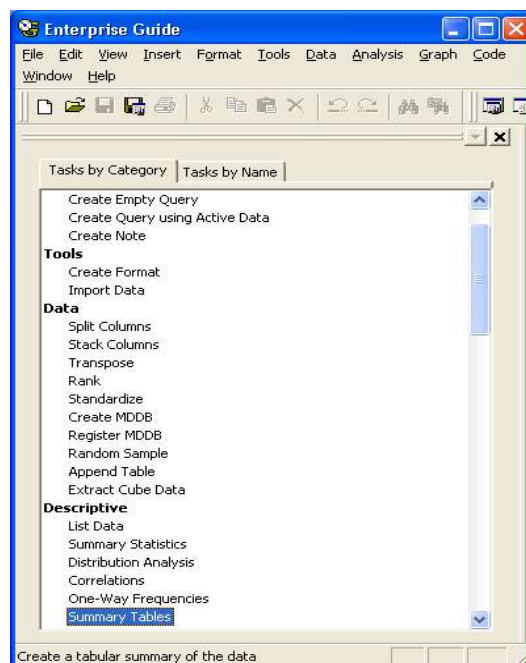
While Enterprise Guide includes Import facilities similar to client SAS, it is also possible, using the Microsoft Data Access Components to automatically convert spreadsheets and databases to SAS tables automatically by just inserting them into the current Project. This works if the spreadsheet or database is accessible from the Windows platform, and is not dependent on any licensing of SAS/ACCESS® for PC Files component, which would be required if the conversion were to be carried out by SAS itself.

As the Java client could be running in an environment where spreadsheets and databases cannot be converted directly, their conversion to SAS tables has to be carried out using the SAS/ACCESS for PC Files component, which must, therefore, be licensed in the remote SAS installation.

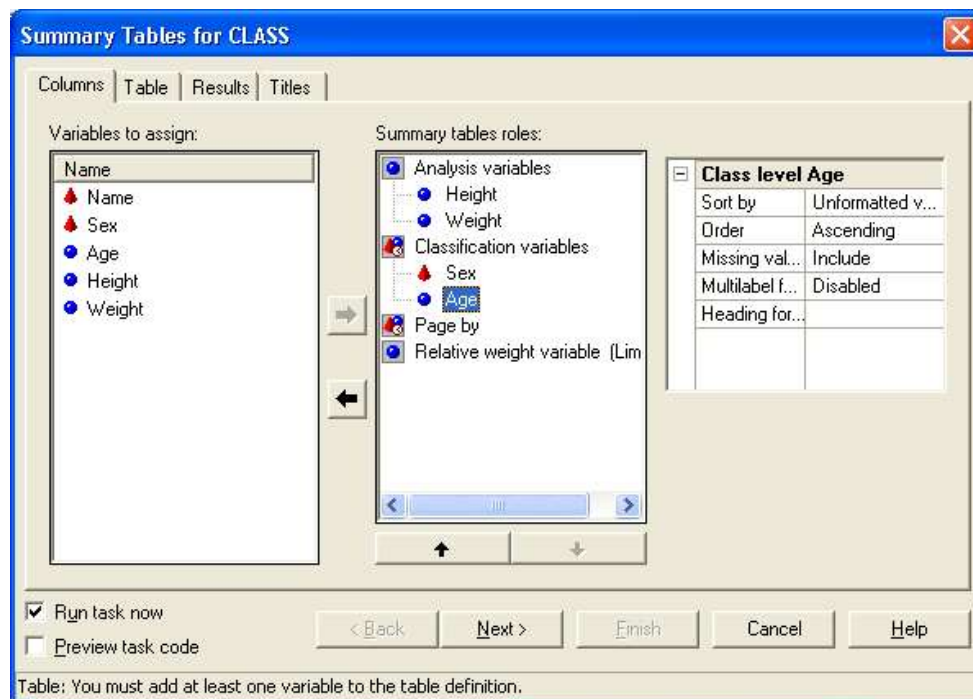
Using and Developing Tasks

While the functions that has been described and compared so far are necessary for the operation of Enterprise Guide or a Java client, the most important benefit of Enterprise Guide is the collection of Tasks supplied with the product. These Tasks are add-in modules that are used to generate SAS code based on selections made using the menus.

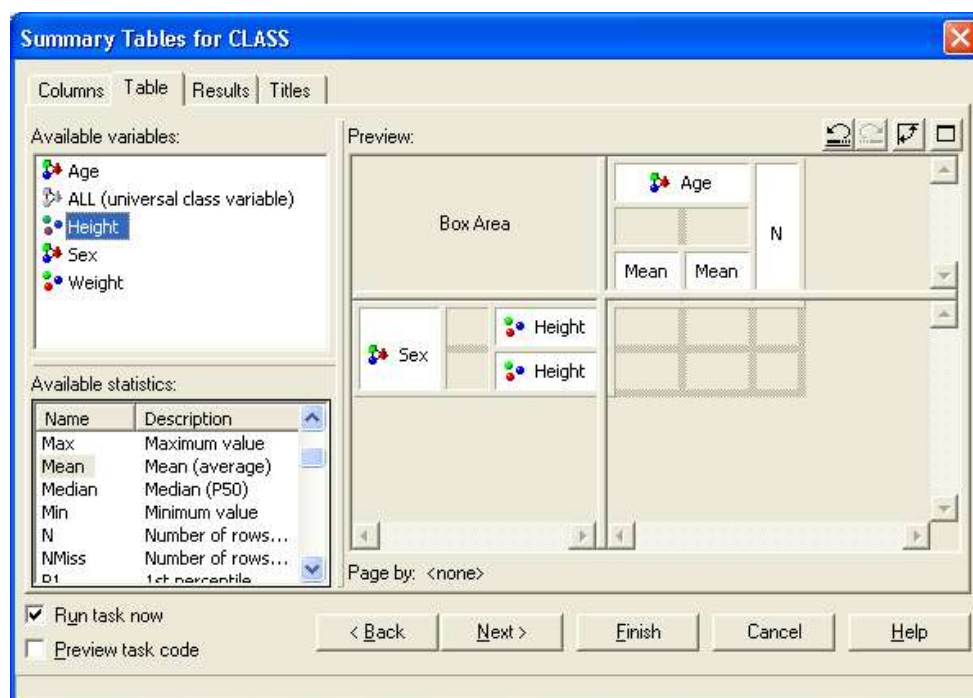
The Tasks supplied with Enterprise Guide range from a simple Query module (see “Reading SAS Tables from the Server” above) to graphical and data analysis modules like Bar or Summary Tables.



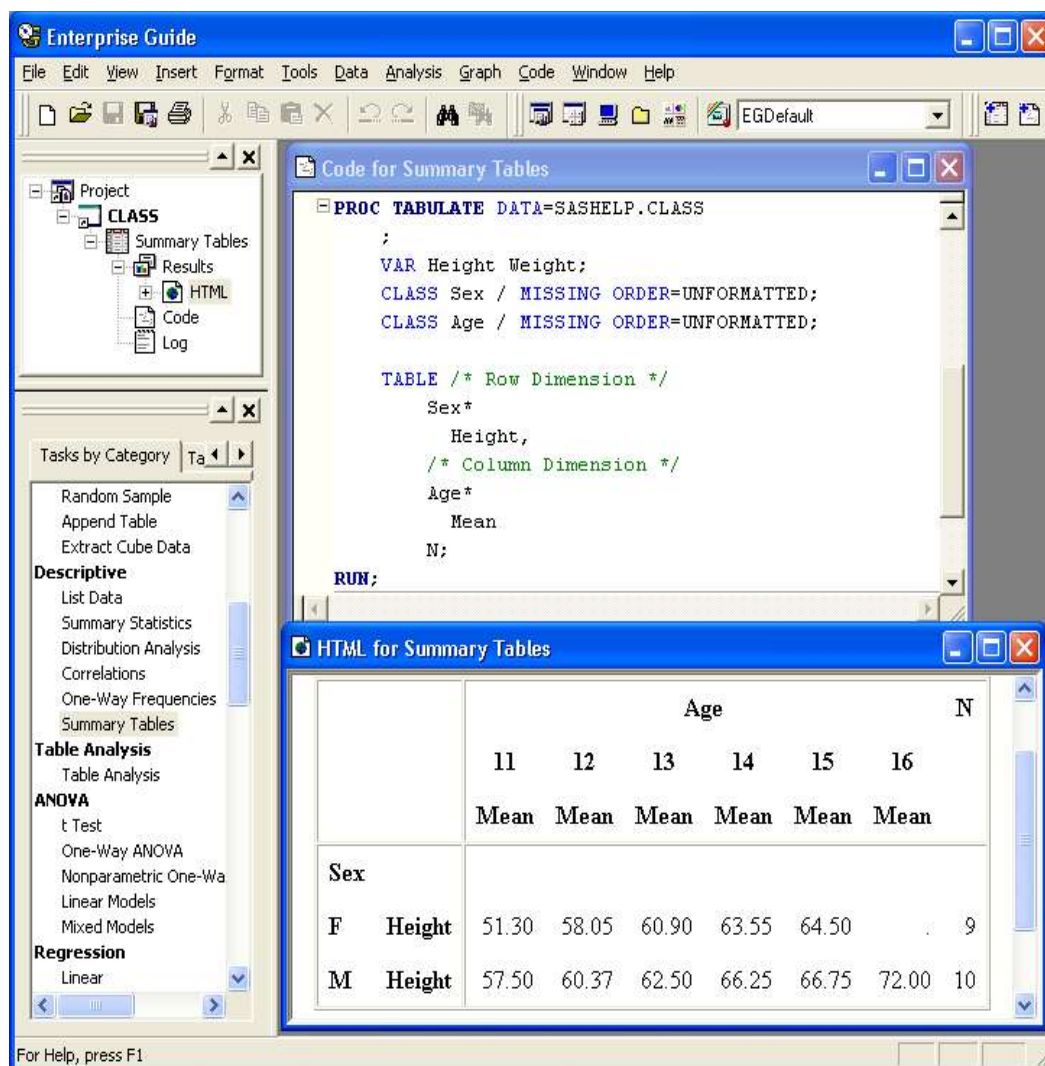
Selecting a Task from the list will open a new window where data columns can be selected and formatted without any SAS coding.



Once the data columns have been selected, then the report can be built up on the screen, again with no requirement that the user is able to code in SAS.



Finally the generated SAS code can be run on the specified SAS server, and the Code, Log and Results nodes added to the current Project.



Tasks in Enterprise Guide look like complex modules, but can be broken down into a number of sub-modules, which are much easier to understand:

1. Data selection and formatting, where the columns that exist in the input table can be dragged (selected) into corresponding categories and their formatting changed, can be carried out by a Java client. Earlier the reading of SAS tables from the server was discussed, and this Java code can be adapted to supply a list of available data columns with their attributes.
2. Code generating, where the SAS code to be run is built up using the information obtained from the user. Anyone who has written SAS programs to write new programs based on simple input data will confirm that it is not a difficult problem to solve, as it is usually based on a collection of IF..THEN..ELSE statements. The Java language can be used to write out the generated SAS code in the same way, and, as the Task modules are designed to perform very tightly defined processes, most of the SAS code can actually be hard-coded in advance.
3. Code submission, where the generated SAS code is submitted to the specified SAS server. Again the submitting of SAS code to the server was discussed earlier, and this Java code can be reused here.
4. Storing Task instructions for reuse, where the information given by the user through the menus and windows can be stored and reused to regenerate the report. This is probably the most complex problem to solve, as a storage format would have to be designed for the Java client that would allow the Task parameters and any generated Code, Log, Results and Data that the user wished to retain to be stored in a single Project file. Enterprise Guide uses its own Project file

(* .seg) format, which holds all these items specific to the Enterprise Guide Tasks until the next time the Project is opened.

Conclusions and Speculations

From the comparison of the existing Enterprise Guide features with functionality provided by the IOM Bridge for Java it can be seen that a Java version of Enterprise Guide could be developed, but with the following significant differences:

- A Java version would not be restricted to a Windows platform, but only limited to those platforms that support Java.
- A Java version can only communicate with a SAS System where SAS Integration Technologies has been licensed, whereas Enterprise Guide can also communicate with a locally installed SAS System having only Base SAS licensed.
- A Java version would require an API based on Java Beans, whereas Enterprise Guide uses an API based on ActiveX DLLs. Methods in ActiveX DLLs can pass multiple values to calling programs, whereas methods in Java Beans can only pass a single value, so instead they must pass a pointer to an array holding all the values to be passed to the calling program.

So we can now speculate on why SAS developed Enterprise Guide to run only on the Windows platform:

- As a thin-client front-end to the SAS System, it would be most used if it could be installed on the most widely used client platform at the time it was first released, i.e. Windows. The ability to run on other platforms would not have significantly increased its use at that time.
- The inability of the Java version to communicate with a locally installed SAS System without SAS Integration Technologies licensed vastly increases its start-up cost, and therefore reduces its market.
- When Enterprise Guide was first released, the current version of SAS was 8, which did not have many features included in the corresponding release of IOM Bridge for Java. However, the COM/DCOM interface that existed in Microsoft Windows at that time could be fully utilised.

Future Development

If anyone reading this paper, whether inside or outside SAS, is considering the development of a platform-independent Java version of Enterprise Guide, then the author would be very interested in discussing a collaborative project.

Contact Details

The author is a consultant for Holland Numerics Ltd and can be contacted at the following address:

	Philip R Holland
address:	Holland Numerics Ltd 94 Green Drift Royston Herts. SG8 5BT UK
e-mail:	<phil.holland@bcs.org.uk>
web:	http://www.hollandnumerics.com/
tel. (mobile):	+44-(0)7714-279085

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Appendix 1: Summary of Features

<i>Enterprise Guide implementation</i>	<i>Mark</i>	<i>Feature</i>	<i>Mark</i>	<i>Java implementation</i>
Yes, on Windows 98, Me, NT, 2000 and XP.	+	Installation on Windows client platform.	+	Yes, if suitable Java runtime libraries installed.
No.	=	Installation on non-Windows client platform.	+	Yes, if suitable Java runtime libraries installed.
Only requires Base SAS licence on the SAS client system.	+	Access local SAS client system.	=	Requires SAS Integration Technologies licence on the SAS client system, and a local TCP/IP network address defined for the client environment.
Requires SAS Integration Technologies licence on the SAS server system.	+	Access remote SAS server system.	+	Requires SAS Integration Technologies licence on the SAS server system.
Use Enterprise Guide Administrator to set the properties of SAS servers available to Enterprise Guide users.	+	Administration of SAS servers available to thin-client users.	?	NB. Need to develop an application to set the properties of available servers, and store these properties in a form that can be read into the thin-client application on start-up.
Use Data window.	+	Viewing SAS tables.	+	Use IDataService and Java SQL interfaces.
Use Code window.	+	Editing text files.	+	Use IFileService interface to read data from text files into Java AWT or Swing component text objects.
Application menus and pop-ups are used to select Local or Remote Server submission of SAS code.	+	Write and submit SAS code to run on the SAS system.	+	Use ILanguageService interface to submit single code statement, or array of code statements.
Use Microsoft Data Access Component.	+	Importing external data sources into SAS tables.	?	Requires SAS/ACCESS for PC Files licence on the SAS server system.
Use Microsoft Data Access Component.	+	Exporting SAS tables into external data sources.	?	Requires SAS/ACCESS for PC Files licence on the SAS server system.
Use Log window.	+	Display of SAS Log.	+	Use ILanguageService interface to return array of Log lines.

<i>Enterprise Guide implementation</i>	<i>Mark</i>	<i>Feature</i>	<i>Mark</i>	<i>Java implementation</i>
SAS Output is treated as another report type and displayed in the Results window (see below).	+	Display of SAS Output.	+	Use ILanguageService interface to return array of Output lines.
Use Results window, where specific report types can be associated with external browsers, e.g. Internet Explorer, Word, Acrobat Reader.	+	Display of reports from SAS ODS, e.g. HTML, RTF, PDF, etc.	?	Use IFileService interface to read data from server-based report files. <u>NB.</u> <i>Need to develop a method to transfer data from the SAS system into local files of the same type, so that they can be read by their associated external browsers.</i>
Use Task window.	+	Point-and-click SAS programming interface.	?	Use Java AWT or Swing components to develop individual modules.
Use Enterprise Guide API for interface within ActiveX DLL components, which can be written in C++, C#, Visual Basic, etc., and registered with Windows and Enterprise Guide.	+	User-written tasks.	?	<u>NB.</u> <i>Need to develop an API for integrating the supplied and user-written tasks into the application.</i>
Use Project file (*.seg) format.	+	Project file format.	?	<u>NB.</u> <i>Need to develop an Project file format suitable for use with Java clients.</i>

Appendix 2: Java Packages

The SAS and CORBA Java packages below can be found in the following Jar files:

```
sas.svc.connection.jar  
sas.core.jar
```

Server Administration

The Java code requires the following packages:

```
import com.sas.iom.WorkspaceConnector;  
import com.sas.iom.WorkspaceFactory;  
import com.sas.iom.SAS.IWorkspace;  
import java.util.Properties;
```

Submitting SAS Code to the Server

The Java code requires the following packages:

```
import com.sas.iom.SAS.ILanguageService;  
import com.sas.iom.SAS.ILanguageServicePackage.CarriageControlSeqHolder;  
import com.sas.iom.SAS.ILanguageServicePackage.LineTypeSeqHolder;  
import com.sas.iom.SAS.IOMDefs.StringSeqHolder;
```

Reading SAS Tables from the Server

The Java code requires the following packages:

```
import com.sas.iom.SAS.IDataService;  
import com.sas.rio.MVACConnection;  
import java.sql.*;
```

Reading Text Files from the Server

The Java code requires the following packages:

```
import com.sas.iom.SAS.IFileService;  
import com.sas.iom.SAS.IFileref;  
import com.sas.iom.SAS.ITextStream;  
import com.sas.iom.SAS.StreamOpenMode;  
import org.omg.CORBA.StringHolder;
```