

Paper 51-28

Developing Custom Analytic Tasks for SAS® Enterprise Guide®

Joe Carter, Stephen McDaniel, Mike Porter, SAS® Institute Inc., Cary, NC

ABSTRACT

With the capability of creating custom add-in tasks being made available in SAS Enterprise Guide version 2.0, it is possible to develop and deploy application tasks that enhance and extend the existing capabilities of Enterprise Guide.

INTRODUCTION

Enterprise Guide version 2.0 allows you to create add-ins that can be added to the repertoire of existing Enterprise Guide tasks, thus leveraging the project management, data management, reporting, analytic, and graphic capabilities of Enterprise Guide, while adding custom functionality for specific application requirements.

GOALS

The goals for developing this analytic custom task add-in with Enterprise Guide were:

- **Functional goal:** deliver an application that generates a large volume of high-quality forecasts
- **Design goal:** provide the user with an intuitive, easy-to-user, point-and-click user interface for interacting with the forecasting application
- **Development goal:** utilize our existing Microsoft Visual Basic programming skills and experience
- **Implementation goal:** make it easy to deliver this custom forecasting application with a minimum of time and effort

APPROACH

After establishing our decision criteria for achieving these goals, we decided to implement this forecasting application with SAS® High-Performance Forecasting software (HPF) and to deliver this application as a custom task add-in in Enterprise Guide. Further, it was decided to use Microsoft Visual Basic version 6.0 as the development environment for building this add-in. While Enterprise Guide custom task add-ins can be developed with any development tool that supports the COM interface, such as Visual C++ or Visual Basic .NET, the authors' familiarity with Visual Basic was the main determinant in choosing Visual Basic as the development tool for creating this add-in.

SAS HPF

The forecasting capabilities we wanted to implement for this Enterprise Guide task add-in are contained in HPF, a SAS software product that includes capabilities for:

- transforming transactional data into time series formats
- automatically selecting among multiple time series models that best explain historical data
- optimizing all model parameters
- generating large numbers of high-quality forecasts
- providing numerous metrics for evaluating forecast performance, with exceptions flagged for further analysis

CREATING A CUSTOM TASK WITH ENTERPRISE GUIDE

At a high-level, each task add-in project requires the following steps:

1. Define the goals of the project
2. Identify the functionality to be delivered through the application
3. Deconstruct the functionality
4. Implement the functionality in a COM development environment, such as Microsoft Visual Basic
5. Compile a dynamic-link library (DLL)
6. Install the DLL on a Microsoft Windows client machine
7. Load the custom task into Enterprise Guide
8. Utilize the custom task

After defining the goals of the project, the next step in building an add-in to Enterprise Guide is to determine which functionality you wish to make available through your custom task. The first decision point in this process of identifying the functional requirements of the custom task is to determine whether there is existing SAS functionality that can be used, as opposed to developing custom SAS code. In this case, the required functionality to be delivered is found in HPF. Once this determination was made, the next step was to decide which features of HPF should be implemented for this application.

HPF SYNTAX

HPF delivers its functionality through the HPF procedure. The PROC HPF syntax was analyzed to determine the required and optional parameters. This examination of the syntax for PROC HPF resulted in our understanding that the required elements of PROC HPF are:

- PROC HPF
- DATA=
- RUN;

All other components of the PROC HPF syntax are optional.

After analyzing the PROC HPF syntax, we decided that the custom task-in would provide the user with the following features:

- Choose which variables to forecast
- Accumulate records across time period (default is none)
- Transformations (default is none)
- Model specification (default is none)
- Statistic of fit (default is Root Mean Square Error (RMSE))
- Missing values (default is no replacement)
- Forecasting lead (default is 12 periods)
- Confidence level (default is $\alpha = .05$)
- Printed output (default is none)
- Plots (not supported directly by HPF)

Now that the syntax of PROC HPF has been evaluated for required and desirable elements, and the feature list is determined, it is possible to proceed with mapping out the desired functionality in a Visual Basic application. But before you begin implementing the HPF functionality in an Enterprise Guide add-in, you have to set up your development environment.

DEVELOPING THE HPF ADD-IN COMPONENT

The HPF task add-in was developed with Microsoft Visual Basic version 6.0. The first step was to create an ActiveX EXE project. For this add-in, the project was named SASHPF. A public class named Custom was defined. Once the project was established, the project references were specified in order to indicate the appropriate type library for this add-in. To create an Enterprise Guide 2.0 add-in, you must implement a set of interfaces that allow Enterprise Guide to connect to, communicate with, and disconnect from an add-in. These interfaces are the plumbing that allows your add-in to interact with Enterprise Guide. These interfaces are delivered in the SASAIOBJ type library. To enable this type library, you need to select the following type library in the Project → References dialog box:

SAS: Enterprise Guide 2.0 Add-in Object Model Type Library

Once the desired type library has been identified, you can implement the required interfaces, which are:

- AddIn Object
- Task Object
- TaskInfo Object
- ErrorSink Object
- DataSink Object

These interfaces were entered in the Declarations section of the Custom class module, as shown:

```
Option Explicit

Implements SEGAddIn
Implements SEGTask
Implements SEGTaskInfo
Implements SEGErrorSink
Implements SEGDataSink
```

Once these interfaces have been implemented, each object's methods became available for use in developing the add-in.

INTERFACES

While the purpose of this paper of this paper is not to provide a tutorial on the Visual Basic code that was used to implement the add-in, it should prove useful to review the role of each of the required interfaces (objects) and some of their associated methods.

AddIn Object

The AddIn object is implemented by the add-in control, which in this case is a Visual Basic form. Enterprise Guide uses the AddIn object to determine whether it can support the add-in component. Methods supplied by the AddIn object include:

- Connect – establishes an initial connection to the add-in task
- Disconnect – unloads the task from memory
- GetName – returns the logical name of the task
- GetDescription – returns the logical description of the task

Task Object

The Task object is called from the Enterprise Guide internal framework and allows the add-in to interact with the other tasks and functionality that is already part of Enterprise Guide. Methods supplied by the Task object include:

- GetProperty – saves task-specific state information in the project file
- SetProperty – gets task-specific state information from the project file
- GetSubmitCode – provides the generated code to the Enterprise Guide application framework for submission to a SAS session

Error Sink Object

This optional interface allows Enterprise Guide to communicate an error and detailed information about it back to the task for error handling. The Error Sink object provides the following method:

- ExecutionError - invoked if a problem occurred while the task was executing

Data Sink Object

This optional interface is called by the Enterprise Guide internal framework when a change has occurred to the input data or to notify it to update information about output data. One of the methods supplied by the Data Sink object is:

- ActiveDataChanged – called the very first time the add-in is invoked as well as to notify the add-in that the active data has changed

Application Object

This interface is implemented by Enterprise Guide, not by an add-in. The Application object exposes core functionality in the Enterprise Guide internal framework that is used with an add-in. One of the key services that the Application object provides to an add-in is that it retrieves the SAS code to be submitted by the add-in and submits the code to a SAS session.

HPF ADD-IN TASK FORMS

Now that the infrastructure has been laid for developing an add-in, it was time to develop that Visual Basic forms that provides a user interface for the HPF add-in functionality. The HPF add-in provides two forms:

- PROCHPF form
- Preview Code form

PROCHPF Form

The PROCHPF form (see figure 1) contains the main functionality in the HPF add-in task. The form was designed with two tabs:

- Columns – provides a variable selector that allows you to drag and drop columns into their respective roles
- Options – allows you to configure the PROC HPF options, including printed output

Preview Code Form

The Preview Code form (see figure 2) allows you to view the code that is about to be submitted by the add-in.

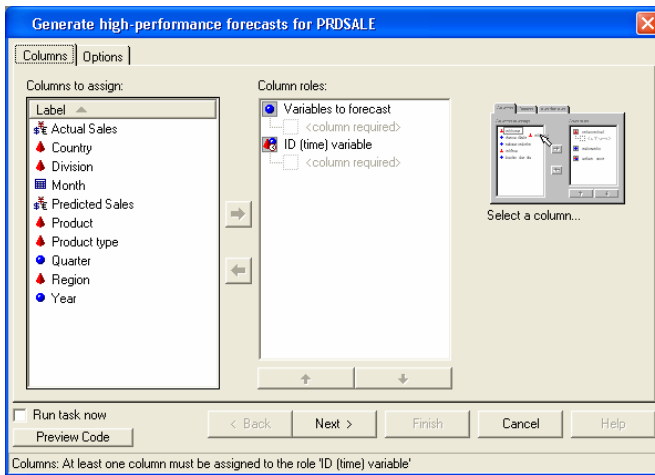


Figure 1: PROC HPF Form - Columns Tab

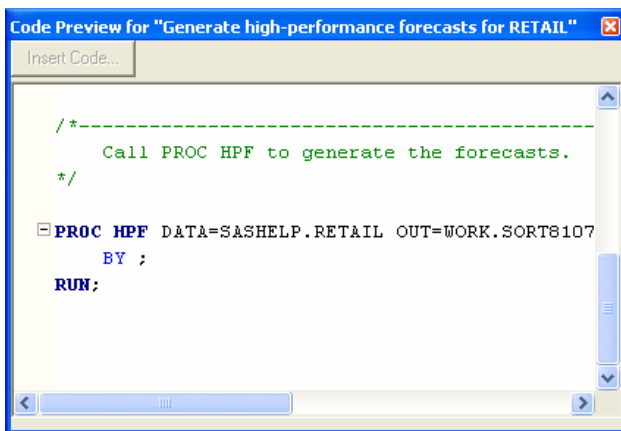


FIGURE 2: Code Preview Form

COLUMNS TAB

The Columns tab (see figure 1) of the PROC HPF form contains a variable selector control that lets you view the available columns in the selected data set and drag these variables into specific roles for the analysis. For the HPF add-in, there are two types of roles:

- Forecast variables – numeric columns that contain data to be forecasted into the future
- ID variable – time column that is used to define the timeline of the data

OPTIONS TAB

The Options tab (see figure 3) of the PROC HPF form provides a way for you to configure PROC HPF options settings. The Options tab consists of 10 frames:

- Accumulate Records
- Transformations
- Model Specification
- Statistic of Fit
- Missing Values
- Forecasting Lead
- Confidence Limit
- Output data set
- Print Options
- Scatter Plots

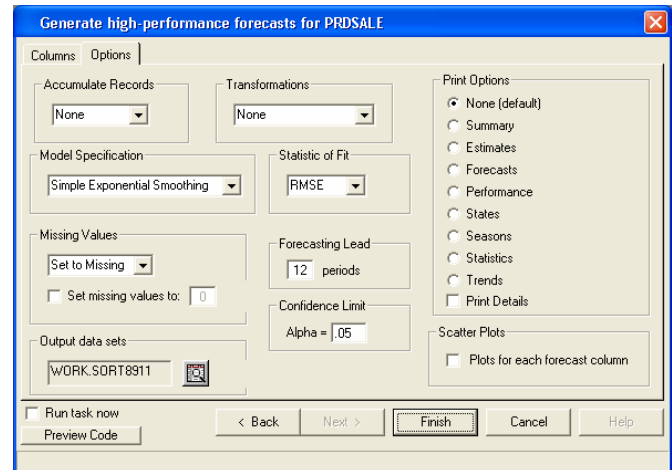


Figure 3: PROC HPF Form - Options Tab

Accumulate Records

The Accumulate Records combo box lets you specify how the records are accumulated within each time period. This sets the ACCUMULATE= option on the FORECAST statement in PROC HPF. The default is NONE, which indicates to PROC HPF that the ID (time) variable is equally spaced with respect to the frequency. Other options made available from the Accumulate Records combo box are: TOTAL, AVERAGE, MINIMUM, MEDIAN, MAXIMUM, N, NMISS, NOBS, FIRST, LAST, STDDEV, CSS, USS.

Transformations

The Transformations combo box allows you to apply a specific transformation to the time series (forecast) columns. This option lets you set the value of the TRANSFORM= option on the FORECAST statement. The default setting is NONE, which means that no transformations will be applied. The other options supplied in the combo box are: LOG, SQRT, LOGISTIC, BOXCOX, AUTO.

Model Specification

The Model Specification combo box lets you select which forecasting model to use, which corresponds to the MODEL= option on the FORECAST statement of PROC HPF. The default setting for PROC HPF is NONE, but for the HPF add-in, the default is set as SIMPLE, which performs simple (single) exponential smoothing. Other options available are: DOUBLE, LINEAR, DAMPTREND, SEASONAL, WINTERS, ADDWINTERS, BEST, BESTN, BESTS, CROSTON, BESTALL.

Statistic of Fit

The Statistic of Fit combo box allows you to indicate the model selection criterion, which corresponds to the SELECT= option on the FORECAST statement. The default value in PROC HPF and for the HPF add-in is RMSE (Root Mean Square Error). The other available values are: SSE, MSE, UMSE, URMSE, MINPE, MAXPE, MPE, MAPE, MDAPE, GMAPE, MINPPE, MAXPPE, MSPPE, MAPPE, MDAPPE, GMAPPE, MINSPE, MAXSPE, MSPE, SMAPE, MDASPE, GMASPE, MINRE, MAXRE, MRE, MRAE, MDRAE, GMRAE, MAXERR, MINERR, ME, MAE, RSQUARE, ADJRSQ, AADJRSQ, RWRSQ, AIC, SBC, APC.

Missing Values

The Missing Values frame allows you to determine how to handle missing values and what value to set them to. This corresponds

to the SETMISSING= option | number setting on the ID statement of PROC HPF. The default option setting is MISSING, which sets the values to missing values. The default number setting is zero (0). Other available settings for the option are: AVERAGE, MINIMUM, MEDIAN, MAXIMUM, FIRST, LAST, PREVIOUS, NEXT.

Forecasting Lead

The Forecasting Lead frame allows you to specify the number of future periods to forecast. This corresponds to the LEAD= option on the PROC HPF statement. The default in PROC HPF and the HPF add-in is 12.

Confidence Limit

The Confidence Limit text box allows you to specify the alpha value for setting the significance level for computing the confidence limits of the forecast. This corresponds to the ALPHA= option on the FORECAST statement. The default in PROC HPF and the HPF add-in is $\alpha = .05$.

Output Data Set

The Output Data Set frame lets you specify the name of the output data set that is generated by PROC HPF. This corresponds to the OUT= option on PROC HPF. The default value in the HPF add-in is to create a temporary data set to store the output data set.

Print Options

The Print Option frame allows you to configure the type and quantity of output. The default option for PROC HPF and the HPF add-in is NONE, which means that no output is generated. Available options are: SUMMARY, ESTIMATES, FORECASTS, PERFORMANCE, STATES, SEASONS, STATISTICS, TRENDS. Also, a Print Details check box is supplied, which allows you to specify that the output be generated with more detailed information. This corresponds to the PRINTDETAILS option on the PROC HPF statement.

Scatter Plots

The Scatter Plots frame allows you to generate scatter plots for each forecast variable by its ID (time) variable, for pre- and post-forecasted time series data.

HOW THE CODE IS GENERATED

The SAS code that is submitted to a SAS server is created by splicing together SAS syntax with the values of controls on the forms. In this example, the `strWork` string is used to assemble the SAS code that is ultimately submitted to a SAS session. In the following code fragment, the PROC HPF statement is concatenated with strings that contain the value of the current `libref` and `member` name.

```
strWork = "PROC HPF DATA=" & Libref & "." &
Member
```

For each of the controls on the forms, the value property of each control is inspected, and appropriate SAS code is generated. For example, the code for retrieving the value of the Accumulate Records combo box looks like this:

```
strWork = strWork & "ACCUMULATE = " &
combAccumulate.Value
```

After the tokens of information have been strung together into a single string, this string value is passed to the `GetSubmitCode` subroutine, which in turn is submitted to the SAS session.

```
Private Sub SEGTask_GetSubmitCode(pbstrCode
As String)
    pbstrCode = TaskUI.Code()
End Sub
```

GENERATING THE HPF ADD-IN DLL

Now that the HPF add-in forms have been created and the code has been written, the next step is to compile the HPF add-in and move it into production. The first step in preparing the HPF add-in for availability is to compile the SASHPF code into a DLL. Provided that no compile-time errors occur, the resulting file is called SASHPF.DLL.

DEPLOYING THE HPF ADD-IN

Once the DLL has been created, it must be installed on a Windows client machine. After it is installed, the HPF add-in must be registered on the Windows machine before it can be loaded into Enterprise Guide. To register the DLL, you can use the `regsvr32` utility. For example:

```
Regsvr32 SASHPF.DLL
```

LOADING THE ADD-IN INTO ENTERPRISE GUIDE

Once the SASHPF.DLL has been registered, you can add it to Enterprise Guide through these steps:

1. Open Enterprise Guide version 2.0 (any project) and select Tools → Customize
2. Click the Add-Ins tab.
3. Click the Add button.
4. In the Register Add-in dialog box, type in the ProgID and click OK. The ProgID for the HPF add-in is SASHPF.Custom. If you install an add-in, the ProdID should be supplied with the documentation for the add-in. You can also use the GetProgID tool from the www.sas.com web site to get the ProgID from an existing DLL.
5. Load the Add-in by clicking the check box next to it. You must load an Add-in task into memory so that you can use it within Enterprise Guide. If you do not want to have this add-in automatically loaded into memory, press the Unload check box.
6. Click OK to close the Customize dialog box.
7. Re-invoke Enterprise Guide to cause your changes to take effect. The custom add-in task should now be available.

RUNNING THE HPF ADD-IN

The HPF Add-in is now available as the High-Performance Forecasting task in the Time Series folder in the Task window (see figure 4). To run this add-in, you must first select a data set that contains time series data. In the following example, you should use the SASHELP.PRDSALE data set. Once this data set has been added to the Enterprise Guide project and selected, open up the HPF Add-in and select SALES as the variable to forecast and select MONTH as the ID (time) column in the Columns tab (see figure 1). Select the Options tab (or use the Next > button) to navigate to the next panel. On the Options tab (see figure 3), you can specify various settings for running the

analysis. In this example, you should select the Forecasts option from the Print Options frame. You should also enable the Scatter Plots check box.

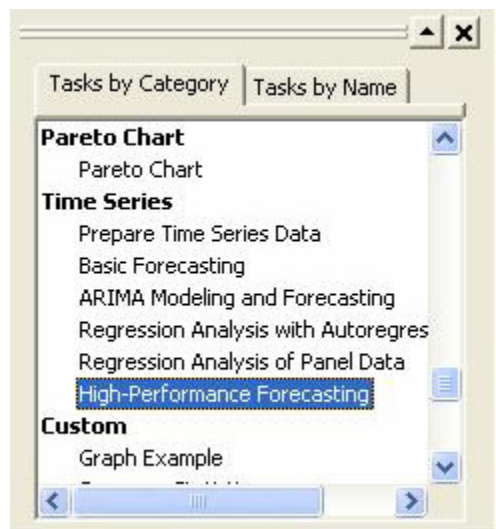


Figure 4: HPF Add-in within the Enterprise Guide Time Series Folder

PROCEDURE RESULTS

In this example, the printed output generated from the HPF add-in consists of two parts:

- Report of forecasted values for the Sales variable (see figure (see figure 5))
- Scatter plots of historical (see figure 6) as well as forecasted sales (see figure 7)

| Forecasts for Variable SALES | | | | | |
|------------------------------|--------|-----------|----------------|-----------------------|-----------|
| Obs | Time | Forecasts | Standard Error | 95% Confidence Limits | |
| 59 | 1994:3 | 992.7691 | 19.8698 | 953.8249 | 1031.7133 |
| 60 | 1994:4 | 1061.8903 | 24.1078 | 1014.6398 | 1109.1408 |
| 61 | 1995:1 | 936.4521 | 25.7530 | 885.9772 | 986.9269 |
| 62 | 1995:2 | 1040.2098 | 30.3272 | 980.7695 | 1099.6500 |
| 63 | 1995:3 | 1042.9886 | 33.0764 | 978.1600 | 1107.8172 |
| 64 | 1995:4 | 1114.9355 | 36.8867 | 1042.6390 | 1187.2320 |
| 65 | 1996:1 | 982.6542 | 35.5007 | 913.0741 | 1052.2343 |
| 66 | 1996:2 | 1090.9057 | 40.4487 | 1011.6277 | 1170.1838 |
| 67 | 1996:3 | 1093.2082 | 42.4123 | 1010.0816 | 1176.3347 |
| 68 | 1996:4 | 1167.9807 | 46.3272 | 1077.1810 | 1258.7803 |
| 69 | 1997:1 | 1028.8563 | 43.1517 | 944.2805 | 1113.4322 |
| 70 | 1997:2 | 1141.6017 | 48.5602 | 1046.4255 | 1236.7779 |

Figure 5: Forecasts of Sales Variable

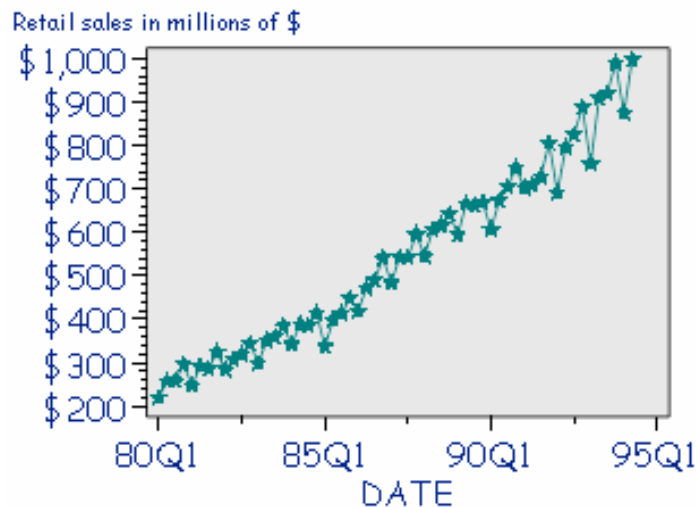


Figure 6: Historical Sales Data (Pre-forecast)

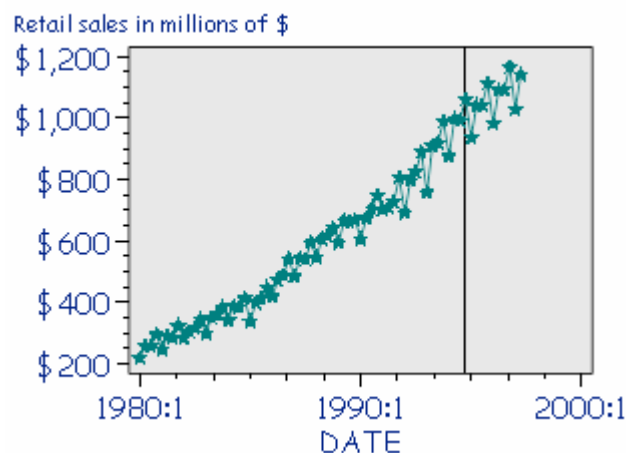


Figure 7: PROC HPF Forecasted Sales

The generated report and scatter plots can be presented and manipulated in the same way you can handle procedure output for any other Enterprise Guide task. One of the great benefits of creating an add-in task is that you do not have to develop custom functionality to handle procedure output.

CONCLUSION

The authors were able to achieve their functional, design, development, and implementation goals for this project by implementing a custom task add-in in Enterprise Guide. Visual Basic version 6.0 was the development environment for this application and met our needs for a development tool. The add-in interface for Enterprise Guide version 2.0 provides an architectural platform for implementing custom functionality and providing richer capabilities to users of Enterprise Guide.

ACKNOWLEDGMENTS

The authors wish to thank the following SAS colleagues for their help and support for this SUGI paper: Gail Kramer, David McNamara, Kecia Serwin, and Eric Hunley.

CONTACT INFORMATION

Your comments and questions are welcome. Please contact the author at:

Joe Carter
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
Joe.Carter@sas.com

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.

Other brand and product names are trademarks of their respective companies.