Paper 054-2013

Versatile Global Prompting for SAS® Web Report

Hong Jiang, PhD, Deloitte Consulting LLP

ABSTRACT

Prompts built into the information map are convenient tools for developers using SAS Web Report Studio (WRS). However, the parameter values set through these prompts are not able to populate to other web report sections or objects, limiting their usefulness. This paper describes a solution for creating versatile global prompts that support one-time user response for multiple objects or sections in a SAS web report. Both single-value and multiple-value selection features can be implemented by following the directions described in this paper.

INTRODUCTION

SAS Web Report Studio (WRS) is a flexible Business Intelligence (BI) tool that allows non-technical users to perform interactive data analysis by clicking and dropping variables, sorting and aggregating data as desired. These features are powered by the information map, which also prompts the user to either enter or select values for a subset of data fields. In such case, an information map is utilized as the data source of a SAS web report. This fact, however, imposes constraints for developing more sophisticated web reports in at least two ways:

First, an information map can only populate prompt parameter values to one report section. When a multiple-section report runs, the user-specified values are only recognized by the first section while other sections do not render the results even if the same information map serves as the data source. Therefore, one-time user response to the prompts does not filter data across the entire report sections; rather, repeated user responses are required to build out the full report.

Second, the limitation also manifests itself in single-section reports that include multiple objects driven by different data sources. Parameter values set through one information map are not able to populate to other objects. For example, if a report needs to present the user name and user-specified parameter values in one table, then information map alone does not support this design.

The solution to these two scenarios is to integrate an information map with stored processes that capture and populate parameter values *globally* to different sections or objects. There is more than one way to design the reports. Solutions presented in this paper are the simplest methods in terms of coding efforts. Solutions for single-value and multi-value prompts do not share exactly the same logics because the current SAS Information Map Studio only offers limited filter functions. More explanations will be provided in the sections to follow. Readers with basic knowledge of creating SAS web reports, SAS information maps and managing stored processes through SAS Management Console will find this paper easy to follow. Figure 1 is the logical workflow for developing a two-section SAS web report permitting single-value selection for prompts.

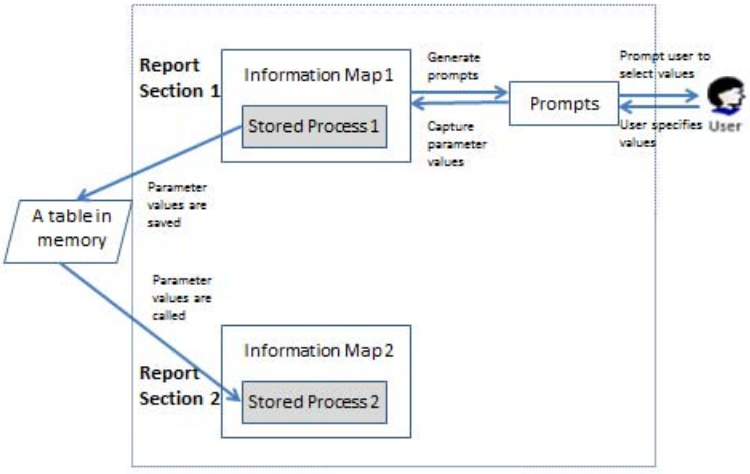

Figure 1. Logic workflow of developing a two-section SAS web report with single-value prompts

The example used throughout this paper manages a fictitious data set from the healthcare industry. In this scenario, health plans managed by different parent organizations are requested to either upload or download a document to a centralized system for review. The final report will provide users two (2) prompts to filter out the parent organization and Health Maintenance Organization ID (HMOID). Users are allowed to select only one value for each prompt in Example 1 and multiple values in Example 2.

EXAMPLE 1: PROMPTS PERMITTING SINGLE-VALUE SELECTION:

This section focuses on a simple scenario where users are only allowed to enter or select a single value for each prompt. The idea of the solution is to combine a stored process with a SAS information map so that the stored process captures prompt values while the information map retains the user interaction flexibility.

In this example, a first stored process creates prompts and captures parameter values in macro variables. The macro variables in turn define the filters of the SAS information map. The same macro variables will be called by a second stored process supporting multiple report sections or report objects. If the new section needs a SAS information map in place, the second stored process is incorporated into the information map for filtering with the parameters.

The macro variable is stored in a user-specific data set located at the SAS library 'wrstemp' which must be set up during SAS BI platform configuration. This library uses a query cache method which improves WRS performance. For reports that contain multiple data-driven objects, the query cache maximizes efficiency by building a temporary common data table that store filter parameters for all data-driven objects in the report. Also, the use of the cache is determined on a per-session basis; parameters set at one session will not interrupt other sessions. The following steps walk through the process of implementing the solution. More details on creating SAS stored processes, information maps and reports are skipped here assuming readers are already familiar with them.

Step 1: Creating a first stored process

The purpose of the first stored process is to capture and store user-specified parameter values. In SAS Enterprise Guide (EG), create a stored process and use the prompts to capture one-time user-specified parameters ❶. The captured parameter values are stored in a data set that corresponds to each user❷. In the example code below, 'hojiang' is the value of the macro variable &_metauser, which is used to distinguish users' input❸. The user's (i.e. hojiang's) parameter value selection is then saved in wrstemp.hojiang.

```
*ProcessBody;
%stpbegin;

%global parent_org HMOID; ❶
options symbolgen;

%let _metauser = %scan(%sysfunc(getoption(METAUSER)),1,'@'); ❷

data wrstemp.&_metauser; ❸
    length Parent_org HMOID $20.;
    parent_org= &parent_org;
    HMOID=&HMOID;
run;

%stpend;
```

Figure 2 illustrates the steps of execution at-a-glance: Follow the stored process manager in SAS EG to name the stored process, insert the example SAS code above, set up execution options and create the prompts. In this example, the stored process is named as 'STP_SetData_Paper'.

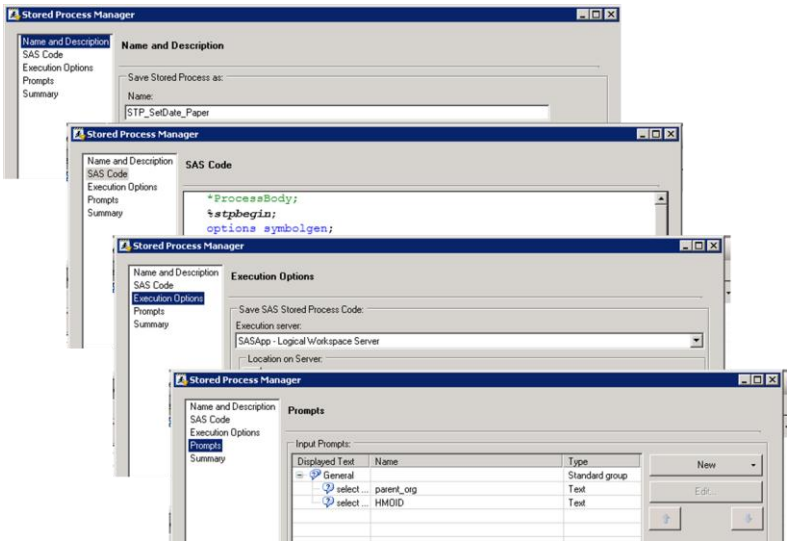

Figure 2. Create a first stored process in SAS EG

Name the prompts exactly the same as the macro variables in the first stored process❶. In this example, the macro variables are “parent_org” and “HMOID”; therefore, the prompts have to be named parent_org and HMOID, although they are not case sensitive. To ensure that the stored process is usable by an information map, make the stored process compatible through SAS Management Console. To complete this step, locate and right click the stored process, then select ‘Make Compatible’ from the drop-down menu.

Step 2: Creating an information map

Create an information map that uses the stored process-STP_SetData_Paper. Instead of building prompts into the information map, add a filter and link each data item to an existing macro variable in STP_SetData_Paper. Figure 3 illustrates how to engage the stored process STP_SetData_Paper and how to configure the information map to recognize the parameter values passed through the macro variables using the ‘expression’ feature. Name this information map as ‘Info_SAS_Paper_1st’.

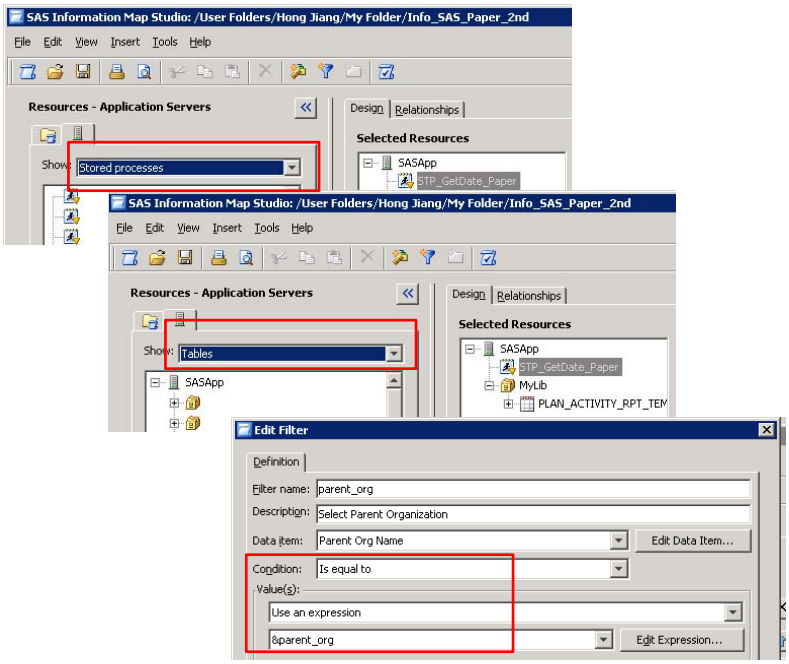

Figure 3. Create an information map by combining a stored process.

***Step 3:* Developing a second stored process**

Develop a second stored process to call the global parameter values when creating a separate report section or displaying another report object within the same section. This stored process calls a parameter value back from a macro variable, and associates the data value with an information map or use it in other SAS statements. This second stored process does not need to build any prompt as it calls parameter values that have been captured by the first stored process and stored in the table of wrstemp.hojiang.

A sample code below, name it as 'STP_GetData_Paper', accomplishes the purpose described above:

```
*ProcessBody;
%stpbegin;

%global parent_org HMOID;

%let _metauser = %scan(%sysfunc(getoption(METAUSER)),1,'@');

data _NULL_;
   set wrstemp.&_metauser;
   call symput("parent_org",parent_org); ❹
   call symput("HMOID",HMOID);
run;

*Insert SAS Procedures for generating a report object as necessary;❺
…
%stpend;
```

If the new report section or object does not require an information map, then simply develop the extra procedures to complete the design❺. Alternatively, if the new report section or object is based on an information map, then follow Steps 4 and 5 to get the job done.

Step 4: Creating a second information map

Create a second information map to filter the parameter values. This step is similar to Step 2 except that the associated stored process is 'STP_GetData_Paper' rather than 'STP_SetData_Paper.' Name this information map as 'Info_SAS_Paper_2nd'.

Step 5: Developing a SAS web report with two sections

Now it is ready to complete an integrated web report with multiple sections and objects using stored processes and information maps. In the first section of the web report, select the information map 'Info_SAS_Paper_1st' as the data source. To enable the filters being built in the information map, click through WRS 'Data', 'Section Filters …' and check the filters that apply to the section.

In order to build a second section, click 'Options' in the 'Table of Contents' panel, then select ''Insert a New Section…". Choose the second information map 'Info_SAS_Paper_2nd' as the source data. Build a table or a chart based on the information map. Then save and test the report.

The result will show that the two sections share the same subset of data because the filters in both sections are governed by the same prompts and the user is required to respond only once.

EXAMPLE 2: PROMPTS PERMITTING MULTI-VALUE SELECTION:

It gets slightly more complicated if a prompt allows multiple selections because more than one data values are tied up to a single prompt parameter. Multiple macro variables can be used to capture all the values. However, SAS Information Map Studio does not have 'in' condition which allows assigning multiple values to a data item as shown in Figure 4.

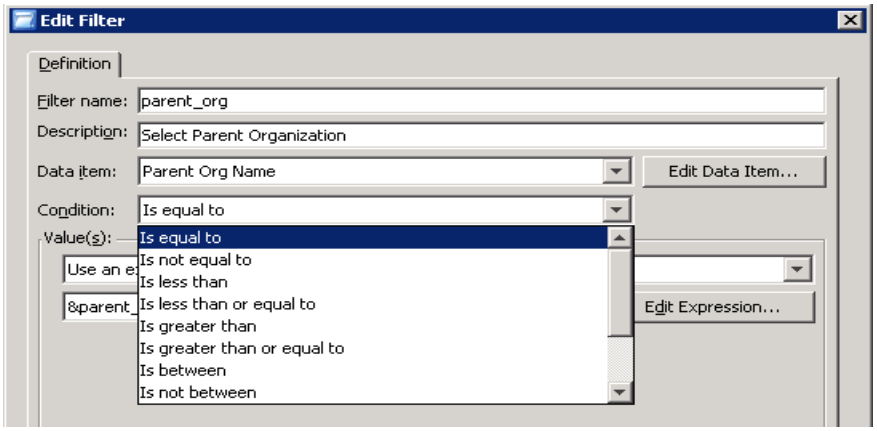

Figure 4. No 'in' condition in SAS Information Map Studio.

To work around this limitation, one solution is to build the information map based on the subset of data filtered out by the prompt selections. In order to know the exact number of values entered for a prompt, use the automatically generated macro variable &PARAMETER_Count to report the number of values for the prompt 'PARAMETER'. Use Macro variables &PARAMETER1, &PARAMETER2, and so on to store the first, the second and subsequent values entered in the prompt 'PARAMETER', respectively. For example, if a user selects two (2) values from the prompt 'parent_org', the macro variable &parent_org_count is equal to 2. Then the macro variable &parent_org1 captures the first and &parent_org2 captures the second value entered by the user. Detailed solution is presented in the following steps.

Step 1: Creating a first stored process

Create a stored process to capture multiple values from the user's response to the prompt. Then select a subset from the original data set according to the user's specification. The sample code below illustrates this process.

```
*ProcessBody;
%stpbegin;
```

```
%global parent_org HMOID;
options symbolgen;
%let _metauser = %scan(%sysfunc(getoption(METAUSER)),1,'@');

*save multi-value parameter into a user specific table
%macro SetData;
data wrstemp.&_metauser;
       %do i=1 %to &parent_org_count; ❻
              var='parent_org';
              val="&&parent_org&i";
              output;
       %end;
run;
%mend;
%SetData

*subset data with the prompt values saved in the user specific table;
proc sql;
create table libname.plan_subset as
       select * from libname.plan
              where parent_org in (select val from wrstemp.&_metauser where
              var='parent_org'); ❼
quit;
%stpend;
```

The do loop in the code captures multiple values that the user responded to the prompt parent_org ❻. The parameter values are saved in an interim table in the library wrstemp. The 'where' clause allows to filter data with all parent_org values specified by the user.❼

Complete the stored process by building a front-end prompt enabling multi-value processing.

Step 2: Building an information map

Similar to the single-value selection case, build an information map using the stored process and the global parameter values generated in Step 1. The difference here is that the interim table is the data source feeding the information map.

Steps 3 through 5 described in the single-value selection case are not necessary for multi-value selection as long as the interim table is the source data for all report sections and/or objects.

CONCLUSION

While prompts built into the information map provide convenient tools for SAS web report developers, they have limitations due to the inflexibility of value assignments across multiple web report sections or objects.

This paper offers a solution for handling scenarios with prompts permitting either single or multi-value selections. By creating stored processes and associating them with the information map or other data sources, developers are able to create a more efficient and versatile SAS web report. Users only need to respond to the prompt once and parameter values will be captured and populated to multiple report sections or multiple report objects. The technique presented here will be useful for many industries such as health care where SAS web reports are widely used.

REFERENCES

1. Sample 40758: Using one set of prompt values to subset all sections in a SAS® Web Report Studio report. http://support.sas.com/kb/40/758.html

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Hong Jiang
Deloitte Consulting LLP
1919 N. Lynn Street,

Arlington, VA 22203
hojiang@deloitte.com