Paper 3461-2015

Integrating Data and Business Rules with a Control Data Set in SAS®

Edmond Cheng, CACI International Inc.

ABSTRACT

In SAS® software development, data specifications and process requirements can be built into a user-defined control data set functioning as components of ETL routines. A control data set provides comprehensive definition on the data source, relationship, logic, description, and metadata of each data element. This approach facilitates auto-generated SAS codes during program execution to perform data ingestion, transformation, and loading procedures based on rules defined in the table.

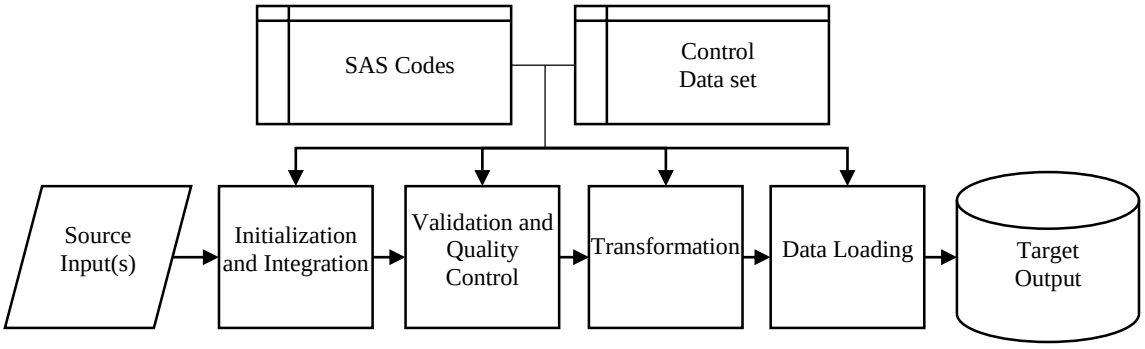
This paper demonstrates the application of using a control data set for the following: (1) data table initialization and integration; (2) validation and quality control; (3) element transformation and creation; (4) data loading; and (5) documentation. SAS programmers and business analysts would find programming development and maintenance of business rules more efficient with this standardized method.

INTRODUCTION

The use of control data set for integrating of multiple source data information and storing coding logics in data form has been around since the early development of computer programming. The control holds the data processing logics and rules designed specifically for the respective application it is written for. Therefore, control file is a series of low-level constructs readable by the associated program to perform meaningful functions.

For analysts, data control allows access to organization of data element definition and business rules, provided the flexibility to review, edit, and update logics, and integrating into automated processing without knowing the mechanical aspects of software engineering. For software developers, this technique greatly reduces the repetitive codes need to be written, and at the same time, increasing the accuracy of desired results and improving incorporation of subject matter knowledge into overall process. Use of a control file has proven to be effective collaboration technique during software development lifecycle, especially for requirements analysis, development, testing, and maintenance of the software..

Figure 1-A Overview Workflow



This paper walkthroughs an application of data control in performing a traditional Extract, Transform, and Load (ETL) operation relates to the SAS programming paradigm, shown in Figure 1-A. The example uses the Federal information technology investment data publicly available from the Federal IT Dashboard website (https://itdashboard.gov/). The website is designed to share with the public as where IT investment is being made and to measure budget performance relates to IT investment by Agency and Bureau. The particular data set chosen is the FY2015 IT Portfolio data. This file provides entry level information on the IT investment by category, spending type, dollar amounts, for Agency and Bureau voluntarily participated in reporting.

Note: the actual data contains over 8,000 reports with 40 plus variables. In subsequent examples used in this paper, the mock data being used has modified to display limited observations and data columns.

Also, selected reported values have been intentionally modified for illustration purposes. A snapshot of the mock data used looks as follows:

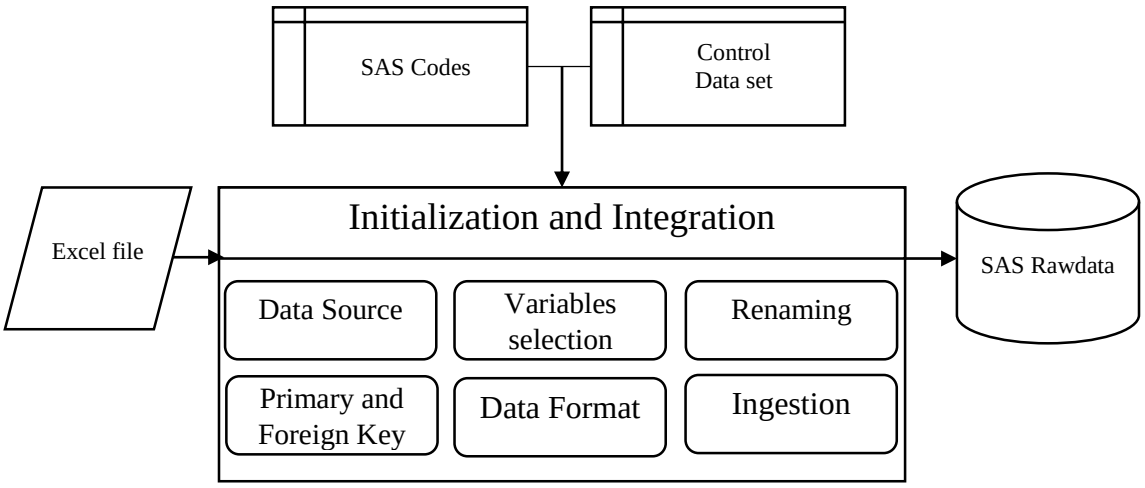
Figure 2-B Federal IT Portfolio data

	A	C	E	F	H	J	L	T	U	V	W	Z	AA	AB
1	Agency Code	Unique Identifier	Bureau Code	Part of Exhibit 53	Type of Investment	Investment Title	FEA BRM Services - Primary service area	DME CY Agency Funding	DME CY Contributions ($	DME BY Agency Funding	DME BY Contributions ($	O&M CY Agency Funding	O&M CY Contributions ($	O&M BY Agency Funding
2	5	005-000000095	68	01 - IT Investments for Mission D	02 - Non Major	FAS International Funds Control Reporting	129 - Reporting and	0	0	0	0	0.013	0	0
3	7	007-000100033	97	01 - IT Investments for Mission D	01 - Major	DoD Healthcare Management System Mod	248 - Health Care De	29.602	0	148.948	0	0	0	0
4	9	009-000000935	90		NA	OS ASA Perry Point Local Area Network Se	139 - Provide and Mi	0	0	0	0	0.3	0	-0.3
5	9		25	01 - IT Investments for Mission D	02 - Non Major	NIH NIGMS Sci Mission Research Knowled	577 - Knowledge Dis	0.25	0	0.24	0	0.09	0	0.07
6	9	009-000325766	10	02 - Infrastructure, IT Security, C	02 - Non Major	FDA - Email	659 - Email	0	0	0	0	5.557	0	5.557
7	10	010-000001462	12	01 - IT Investments for Mission D	02 - Non Major	USGS - WAT - Water	056 - Water Resourc	0	0	0	0	-11.8521	0	11.85206
8	11	011-000003434	10	01 - IT Investments for Mission D	02 - Non Major	FBI Collection Operations and Requiremer	045 - Criminal Invest					0		
9	12	012-000003020	20	01 - IT Investments for Mission D	02 - Non Major	Current Population Survey (CPS) Maintena	070 - General Purpo	0	0	0	0	1.539	0	1.732
10	14	014-000000019	0	01 - IT Investments for Mission D	01 - Major	Electronic Medical Record (EMR)	248 - Health Care De	2.1	0	8.3	0	5.63	0	3.825
11	15	15-000000255	5	01 - IT Investments for Mission D	02 - Non Major	Fiscal Projection System (FPS)	013 - Financial Secto	0	0	0	0	0.39	0	0
12	16	016-000002252	0	02 - Infrastructure, IT Security, C	01 - Major	Infrastructure - Office Automation	139 - Provide and Mi	122.777	0	94.802	0	72.936	0	44.904
13	18	018-000005555	80	01 - IT Investments for Mission D	02 - Non Major	Database Management Sys (Asset Mgmt) (	119 - Facilities, Flee	0	0	0.0745	0	0.4005	0	0.405
14	19	019-000000435	10	01 - IT Investments for Mission D	02 - Non Major	LM Property Management System (Sunflo	144 - Inventory Cont	0	0	0	0	0.061	0	0.062
15	20	020-999996060	0	01 - IT Investments for Mission D	01 - Major	eRulemaking	115 - Rule Publicatic	0.1	0	0	0	0.898	6.6	1
16	21	021-270139098	4	02 - Infrastructure, IT Security, C	01 - Major	DOTXX126: DOT Identity, Credential, and A	648 - Identification a	4.613233	0	1.4	0	5.46411	0	5.059831
17	23	023-000004050	30	01 - IT Investments for Mission D	01 - Major	National Electronic Accounting and Report	124 - Accounting	0	0	0	0	11.04367	0	11.58685
18	24	024-000000591	0	02 - Infrastructure, IT Security, C	01 - Major	DHS - Infrastructure Transformation Progra	139 - Provide and Mi	0	0	0	0	206.181	0	126.98
19	26	026-000001001	0	01 - IT Investments for Mission D	01 - Major	MSFC - ED - Payload Operations and Integr	026 - Scientific and T	1.054	0	0.243	0	20.318	0	20.466
20	28	028-000001001	0	01 - IT Investments for Mission D	01 - Major	OCFO: Oracle Administrative Accounting/J	124 - Accounting	0	0	0	0	0	0	0
21	422	422-000000032	0	02 - Infrastructure, IT Security, C	01 - Major	IT Infrastructure, Office Automation and Te	139 - Provide and Mi	0	0	0.492	0	31.398	0	31.658

DATA TABLE INITIALIZATION AND INTEGRATION

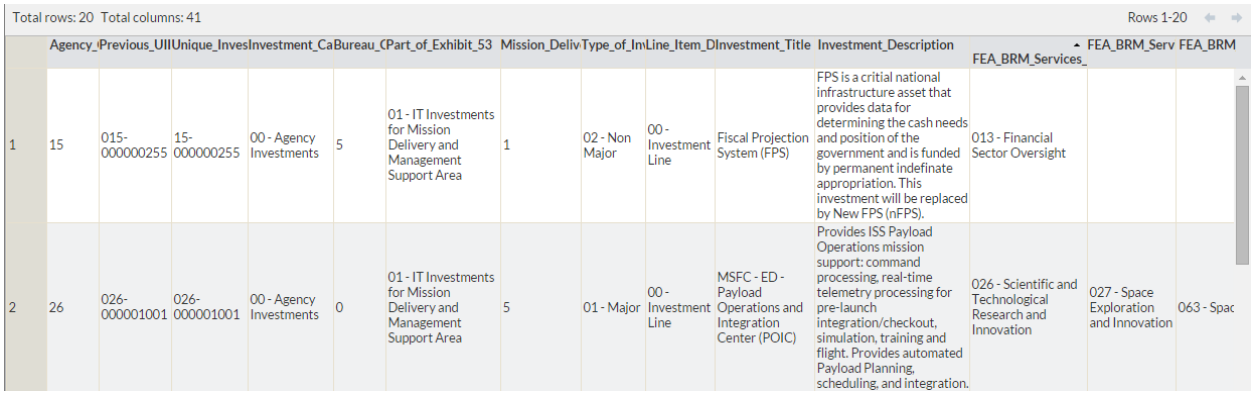
A traditional ETL process identifies the desired data from the original source(s) for which the software system to extract. With many choices of SAS PROCs, ACCESS engine, and programming techniques, SAS can handle common data format such as flat files, relational databases, XLM, JSON, and support for many other popular proprietary formats. Within SAS, you can view the organization metadata associated with the data by using PROC CONTENTS, PROC SQL, or properties task to get a description of the variables in the data. The paper chosen PROC CONTENTS to create the initial data dictionary. The data dictionary is the basis of the control data set which will further be modified to include logics parameters and constructs in conjunction with SAS programs to be used for code generation.

Figure 2-A Initialization and Integration workflow



In the first part, the IT Portfolio data in Microsoft Excel format is imported into SAS using PROC IMPORT with XLSX engine. Depending on the field names, several of the imported variables had issues with special characters, blank spaces, over 32 characters limit, and incorrect formats.

Figure 2-B Initialization Stage: Input Data



Total rows: 20 Total columns: 41 Rows 1-20

	Agency_I	Previous_UII	Unique_Inves	Investment_Ca	Bureau_C	Part_of_Exhibit_53	Mission_Deliv	Type_of_Im	Line_Item_D	Investment_Title	Investment_Description	FEA_BRM_Services_	FEA_BRM_Serv	FEA_BRM
1	15	015-000000255	15-000000255	00 - Agency Investments	5	01 - IT Investments for Mission Delivery and Management Support Area	1	02 - Non Major	00 - Investment Line	Fiscal Projection System (FPS)	FPS is a critial national infrastructure asset that provides data for determining the cash needs and position of the government and is funded by permanent indefinate appropriation. This investment will be replaced by New FPS (nFPS).	013 - Financial Sector Oversight		
2	26	026-000001001	026-000001001	00 - Agency Investments	0	01 - IT Investments for Mission Delivery and Management Support Area	5	01 - Major	00 - Investment Line	MSFC - ED - Payload Operations and Integration Center (POIC)	Provides ISS Payload Operations mission support: command processing, real-time telemetry processing for pre-launch integration/checkout, simulation, training and flight. Provides automated Payload Planning, scheduling, and integration.	026 - Scientific and Technological Research and Innovation	027 - Space Exploration and Innovation	063 - Spac

The control data set is updated with new fields in organizing and cleansing of the ingested variables. Typically, not all source variables are required to be kept. Often time, you might be selecting different elements from multiple sources. The entries can be manually entered into the initial control file

- **INPUT** – arrange the order of the variables and use as a flag indicating the columns to keep
- **ELEMENT** – new element name of the variable

Figure 2-C Initialization Stage: Control File

Obs	Inpu	Element	Varnum	Variable	Label	Type	Length	Format
1	5	AgencyCode	1	AGENCY_CODE	Agency Code	Numeric	8	BEST12.
2	2	UII_Prev	3	PREVIOUS_UII	Previous UII	Char	209	$209.00
3	1	UII	4	UNIQUE_INVESTMENT_I	Unique Investment I	Char	13	$13.00
4	7	InvestmentCategory	5	INVESTMENT_CATEGORY	Investment Category	Char	31	$31.00
5	6	BureauCode	6	BUREAU CODE	Bureau Code	Numeric	8	BEST12.
6	8	ITPortfolioPart	8	PART OF EXHIBIT 53	Part of Agency IT I	Char	75	$75.00
7	9	MDMSACategory	9	MISSION_DELIVERY_AI	Mission Delivery ar	Numeric	8	BEST12.
8	10	InvestmentType	10	TYPE_OF_INVESTMENT	Type of Investment	Char	29	$29.00
9	11	LineItemDescriptor	11	LINE_ITEM_DESCRIPT(	Line Item Descripto	Char	20	$20.00
10	12	InvestmentTitle	12	INVESTMENT TITLE	Investment Title	Char	224	$224.00
11	13	InvestmentDescripti(	13	INVESTMENT DESCRIP	Investment Descript	Char	255	$255.00

The first SAS code example illustrates the use of PROC SQL to interact with the control file. The SQL code reads the control file for all observations where INPUT is not null, indicating the variables to keep. As each element is being read, macro variables are created:

- **&keepvar** – the list of new element names to store and order
- **&renamevar** - list of variables and syntax to rename variables

Figure 2-D Initialization Stage: SAS Code

```
/* Create the mvar for list of variables to keep and to rename
   for input is flagged */
PROC SQL noprint;
  select Variable, compress(Variable||'='||Element)
  into :keepvar separated by ' ', :renamevar separated by ' '
  from mydata.&controlfile
  where Input is not null
  order by Input;
QUIT;

%PUT List of Data Elements to keep: &keepvar;
%PUT List of Data Elements to rename (OLD->NEW): &renamevar;

DATA rawdata;
  retain &keepvar;
  set MYDATA.ITPortfolio;
  keep &keepvar;
  rename &renamevar;
RUN;
```

A DATA STEP recalls the macro variable values as constructs for the associated SAS functions to perform intended actions. The results are shown in Figure 2-E. Notice the primary key, UII, is now ordered as the first column of the data, and the old variable name Part_of_Exhibit_53 has been renamed to ITPortfolioPart. Not shown: other issues with variable names containing long variable names and special characters are now converted to SAS compatible names. Variables not in &keepvar are dropped.

Figure 2-E Initialization Stage: Output

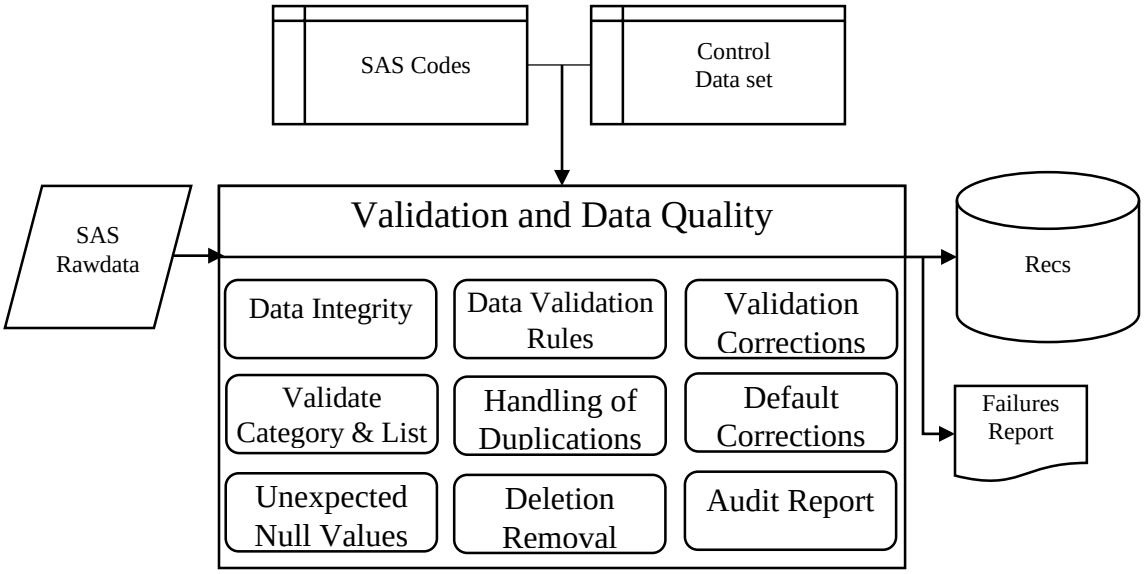
Total rows: 20 Total columns: 30 Rows 1-20

	UII	AgencyCo	BureauCc	ITPortfolioPart	InvestmentType	InvestmentTitle	FEA_BRM_Primary	DMEFunding_FY2014	DMECont	DMEFunding_FY2015	DMEContri
1	005-000000095	5	68	01 - IT Investments for Mission Delivery and Management Support Area	02 - Non Major	FAS International Funds Control Reporting System (IFCRS)	129 - Reporting and Information	0	0	0	0
2	007-000100033	7	97	01 - IT Investments for Mission Delivery and Management Support Area	01 - Major	DoD Healthcare Management System Modernization	248 - Health Care Delivery Services	29.602	0	148.948	0
3	009-000000935	9	90		NA	OS ASA Perry Point Local Area Network Services	139 - Provide and Maintain IT Infrastructure	0	0	0	0
4	009-000325766	9	10	02 - Infrastructure, IT Security, Office Automation, and Telecommunications	02 - Non Major	FDA - Email	659 - Email	0	0	0	0

VALIDATION AND QUALITY CONTROL

Validation ensures the data quality must meet the requirements for data to be useful for processing and analysis. Data validation contributes to the success, or failures, of the overall output product. Validation stages include the logics, elements, parameters, and handling control for the validating data to be useful for further processing. Quality control helps identifying completeness and accuracy, as well as performing correction and producing audit report for further assessing ways to improve quality. It is not atypical to see focus in development effort in this ETL stage in addressing quality deficit items.

Figure 3A. Validation and Data Quality workflow



The original IT Portfolio data is well maintained and blessed with beyond excellent data quality. This can be accredited to the ETL process behind the collection, processing, storage, and user documentation attributable in its data processing. The example data used in Figure 3-B has been manually modified to generate data quality issues. These include missing required UII identifier key, Investment category not reported or reported in a non-standard, improper coding, proper abbreviated agency or bureau code, valid dollars amounts, missing values reported, and so forth.

Figure 3-B Initialization Stage: Input Data

Total rows: 20 Total columns: 30 Rows 1-20

	UII	AgencyCo	BureauCc	ITPortfolioPart	InvestmentType	InvestmentTitle	FEA_BRM_Primary	DMEFunding_FY2014	DMECont	DMEFunding_FY2015	DMEContri
1	005-000000095	5	68	01 - IT Investments for Mission Delivery and Management Support Area	02 - Non Major	FAS International Funds Control Reporting System (IFCRS)	129 - Reporting and Information	0	0	0	0
2	007-000100033	7	97	01 - IT Investments for Mission Delivery and Management Support Area	01 - Major	DoD Healthcare Management System Modernization	248 - Health Care Delivery Services	29.602	0	148.948	0
3	009-000000935	9	90		NA	OS ASA Perry Point Local Area Network Services	139 - Provide and Maintain IT Infrastructure	0	0	0	0
4	009-000325766	9	10	02 - Infrastructure, IT Security, Office Automation, and Telecommunications	02 - Non Major	FDA - Email	659 - Email	0	0	0	0
5		9	25	01 - IT Investments for Mission Delivery and Management Support Area	02 - Non Major	NIH NIGMS Sci Mission Research Knowledge Dissemination Tools	577 - Knowledge Distribution and Delivery	0.25	0	0.24	0

Looking at Figure 3-C, the control file from previous stage is now including four new columns:

- **Logic_Rule_Description** – a meaningful description to broad users
- **Logic_Rule_Type** – a given function type to be used in conjunction with SAS codes to perform particular function
- **Logic_Rule_Condition** – the SAS syntax construct for which the condition is being checked for the particular logic in order
- **Logic_Rule_Return** – the SAS syntax construct for which the function should return in the event if the condition is met

Figure 3-C Initialization Stage: Control File

Obs	Input Element		Logic_Rule_Description	Logic_Rule_Type	Logic_Rule_Condition	Logic_Rule_Return
1	1	UII	Validate the format as ###-#########	val_if_then_else	length(UII)=13 and notdigit(substr(UII,1,3))=0 and substr(UII,4,1)='-' and	Delete
4	4	Status	Auto assign status information if missing	val_if_then	Status = ''	Status = 'No change in status'
5	5	AgencyCode	Check Agency code in existing codes	val_macrocall	%gencode(runmode=validate, codetype=agencycode)	Delete
8	8	ITPortfolioPart	Check the 2-digitPart of Agency IT Portfolio code in range 01-06	val_if_then_else	substr(ITPortfolioPart,1,2) in ('01', '02', '03', '04', '05', '06')	ITPortfolioPart = '01 - IT Investments for Mission Delivery and
19	19	DMEFunding_FY2014	Validate reported amount as postive total; if failed, assign zero dollars	val_if_then_else	DMEFunding_FY2014 >= 0	DMEFunding_FY2014 = 0

The values stored in these new 'Logic' columns hold the critical construct for each processing rule and condition not only in the validation stage, but as well as the transformation stage to be described later. The **Logic_Rule_Description** is intended for documentation purposes. The **Logic_Rule_Condition** and **Logic_Rule_Return** are SAS codes stored in the control file for the given validation tests to be performed. As for differentiating the multiple types of validation test, the **Logic_Rule_Type** are identified by the three separate types, shown in Figure 3-C. The actual functions are designed and coded by the Developer in the programing portion.

- **Val_if_then** – validation test: if condition is met, then perform the return function
- **Val_if_then_else** – validation test: if condition is not met, then perform the return function
- **Val_macrocall** – validation test: execute the specified macro per the parameters specified and return results

The full SAS code is available in the Appendix section of the paper, the selected SAS code shows the interaction between the SAS program and data control file.

Figure 3-D Initialization Stage: SAS Code

```
/* Create the mvar for list of validation tests based on
   logic_rule_type beginning with 'val' indicating validation tests */
%PUT Reading Control file &controlfile;
DATA _null_;
    set mydata.&controlfile end=eof;
    call symput('Element'||compress(put(put(_N_,2.),2.)), strip(Element));
    call symput('Logic_Rule_Type'||compress(put(put(_N_,2.),2.)), strip(Logic_Rule_Type));
    call symput('Logic_Rule_Condition'||compress(put(_N_,2.)), strip(Logic_Rule_Condition));
    call symput('Logic_Rule_Return'||compress(put(_N_,2.)), strip(Logic_Rule_Return));
    if eof then call symput('Logic_Rule_Order', strip(put(_N_,2.)));
    where Logic_Rule_Type =: 'val' and Input is not null;
RUN;

DATA Recs (drop=Fail_Flag )        /* Interim Output passed validation corrections */
     Audit_Report_Failure;         /* Audit Report tracking records and validation failures */
  Length Fail_Flag $500.;
  set rawdata;
  %do n = 1 %to &Logic_Rule_Order; /* &n iterates thru each validation tests */
    /* Validation IF-THEN-ELSE */
    %IF "&&Logic_Rule_Type&n" = "val_if_then_else" %THEN %DO;
      IF not(&&Logic_Rule_Condition&n) then do;
        Fail_Flag = "Data Element failed: &&Element&n";
        output Audit_Report_Failure;
        &&Logic_Rule_Return&n;
      end;
    %END;

    /* Additional Codes */
    /* Additional Codes */
```

DATA STEP and CALL SYMPUT is used in this example to generate the macro variables by iteratively reading through the control for each validation tests as indicated by any Logic_Rule_Type beginning with ‘val’. A series of macro variables, such as Logic_Rule_Condition1, Logic_Rule_Condition2, …, Logic_Rule_Condition’N’, are created. The iterative macro variables are passed as constructs into the series of validation tests syntax for each the Logic_Rule_Type function, prior to execution of the DATA STEP. In another words, the SAS program generates a series of validation test codes based on the defined functions for each unique Logic_Rule_Type and the snippet of SAS codes stored in the Logic_Rule_Condition and Logic_Rule_Return from the control data set. Like putting pierces of puzzles together, the full list of validation tests are generated for the DATA STEP to execute.

Comparing the same UII keys shown in earlier Figure 3-B, you can see the post-validation tests results reflected in Figure 3-E. For example, the entry which failed valid UII key has been deleted from the output data, missing or invalid investment category have been default to a standard value, and negative or missing investment figures have set to zero dollars. Not shown: any part of quality control, failed validation tests for each UII and elements are recorded in a separate data set which facilitates audit report purposes.

Figure 3-E Initialization Stage: Output

Total rows: 18 Total columns: 30 — Rows 1-18

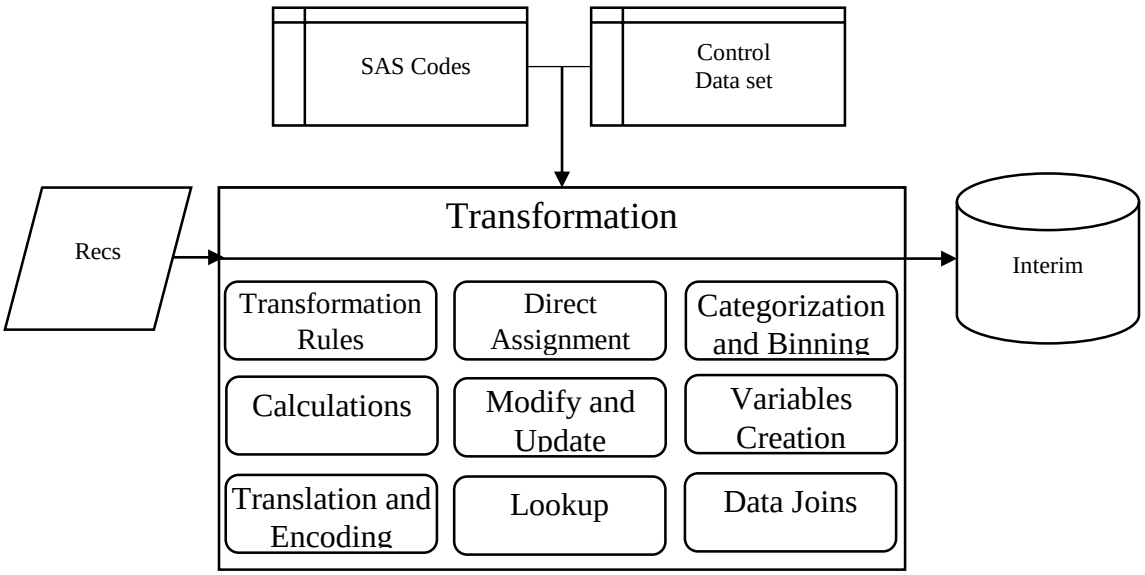
	UII	AgencyC	BureauCc	ITPortfolioPart	InvestmentType	InvestmentTitle	FEA_BRM_Primary	DMEFunding_FY2014	DMECo	DMEFunding_FY2015	DMECc	OMFunding_FY2014
1	005-000000095	5	68	01 - IT Investments for Mission Delivery and Management Support Area	02 - Non Major	FAS International Funds Control Reporting System (IFCRS)	129 - Reporting and Information	0	0	0	0	0.013
2	007-000100033	7	97	01 - IT Investments for Mission Delivery and Management Support Area	01 - Major	DoD Healthcare Management System Modernization	248 - Health Care Delivery Services	29.602	0	148.948	0	0
3	009-000000935	9	90	01 - IT Investments for Mission Delivery and Management Support Area	01 - Major	OS ASA Perry Point Local Area Network Services	139 - Provide and Maintain IT Infrastructure	0	0	0	0	0.3
4	009-000325766	9	10	02 - Infrastructure, IT Security, Office Automation, and Telecommunications	02 - Non Major	FDA - Email	659 - Email	0	0	0	0	5.557
5	010-000001462	10	12	01 - IT Investments for Mission Delivery and Management Support Area	02 - Non Major	USGS - WAT - Water	056 - Water Resource Management	0	0	0	0	0

The obvious advantage is minimizing the coding written in the SAS programs, especially for business rules and logics which require higher maintenance effort. This minimizes the amount of black box coding in the programs and also organizes the logic condition in a readable fashion. The control file facilitates a nice technique for SAS program to auto generate codes.

ELEMENT TRANSFORMATION AND CREATION

The data transformation stage transforms existing elements and create new elements relevant for consumption of the output data, such as analysis and reporting. In most business situations, the Subject Matter Experts or Requirement Analysts should provide data details to Developers for engineering into the software or ETL process. A challenge many offices has faced is maintenance of the business rules when requirement have changed, data sources are added, data elements have updated. In many environments, for good and bad reasons, accessing the source codes or updating business logics might either be inaccessible or untimely. A goal of the control data set is storing and maintaining 80% to 90% of these functions in an accessible repository minizing hardcoding in the source programs.

Figure 4-A. Transformation workflow



This part will take the validated data from previous stage and run through the new transformation rules which are now been added to the existing control file. The expected results after completing the transformation is new variables on agency and bureau name, calculation of various IT function levels and metrics, and category flag for subject of interest.

Figure 4-B Initialization Stage: Input Data

Total rows: 18 Total columns: 30 Rows 1-18

	UII	AgencyC	BureauCc	ITPortfolioPart	InvestmentType	InvestmentTitle	FEA_BRM_Primary	DMEFunding_FY2014	DMECo	DMEFunding_FY2015	DMECc	OMFunding_FY2014
1	005-000000095	5	68	01 - IT Investments for Mission Delivery and Management Support Area	02 - Non Major	FAS International Funds Control Reporting System (IFCRS)	129 - Reporting and Information	0	0	0	0	0.013
2	007-000100033	7	97	01 - IT Investments for Mission Delivery and Management Support Area	01 - Major	DoD Healthcare Management System Modernization	248 - Health Care Delivery Services	29.602	0	148.948	0	0
3	009-000000935	9	90	01 - IT Investments for Mission Delivery and Management Support Area	01 - Major	OS ASA Perry Point Local Area Network Services	139 - Provide and Maintain IT Infrastructure	0	0	0	0	0.3
4	009-000325766	9	10	02 - Infrastructure, IT Security, Office Automation, and Telecommunications	02 - Non Major	FDA - Email	659 - Email	0	0	0	0	5.557

The partial look of the control file in Figure 4-B shows the addition of transformation logics similiarly entered to ones for validation tests. The new transformation logics are prefixed with 'cal' in the Logic_Rule_Type. Again, the logic rule type are called in the SAS programs along with the matching Logic_Rule_Condition and Logic_Rule_Return construct to form a complete SAS syntax to perform following.functions:

- **Cal_assign** – transformation rule: assign the return value per the specified logic
- **Cal_if_then** – transformation rule: if condition is met, then perform the return function
- **Cal_macrocall** – transformation rule: call the specified macro per the parameters specified and return results

Figure 4-C Initialization Stage: Control File

Obs	Element	Logic_Rule_Description	Logic_Rule_Type	Logic_Rule_Condition	Logic_Rule_Return
6	Agency	Auto assign name by Agency Code	cal_macrocall	%gencode(runmode=assign,codetype=agencycode)	
16	FEA_BRM_Description	Concatenate Primary and Secondary PEA BRM	cal_assign		catx(' | ', FEA_BRM_Primary, FEA_BRM_Secondary1, FEA_BRM_Secondary2,
22	ITSpending_FY2014	Calculate the sum of CY DME and O&M amount	cal_assign		sum(DMEFunding_FY2014,DMEContributions_FY2014,OMFunding_FY2014,OMContributions_FY2014)
24	ITSpending_OTYC	Calculate the CY to BY level change	cal_assign		ITSpending_FY2015 - ITSpending_FY2014
36	Flag_Oracle	Search text 'Oracle' in Investment Title and Description	cal_if_then	index(upcase(InvestmentTitle),'ORACLE') or index(upcase(InvestmentDescription),'ORACLE')	Flag_Oracle = 1

The full SAS code is available in the Appendix section of the paper, the selected SAS code shows the interaction between the SAS program and data control file for the element transformation and creation stage.

Figure 4-D Initialization Stage: SAS Code

```
/* Create the mvar for list of calculated and derived Data Elements based on
   logic_rule_type beginning with 'cal' indicating calcuation or derived logics */
%PUT Reading Control file &controlfile;
DATA _null_;
    set mydata.&controlfile end=eof;
    call symput('Element'||compress(put(put(_N_,2.),2.)), strip(Element));
    call symput('Logic_Rule_Type'||compress(put(put(_N_,2.),2.)), strip(Logic_Rule_Type));
    call symput('Logic_Rule_Condition'||compress(put(_N_,2.)), strip(Logic_Rule_Condition));
    call symput('Logic_Rule_Return'||compress(put(_N_,2.)), strip(Logic_Rule_Return));
    if eof then call symput('Logic_Rule_Order', strip(put(_N_,2.)));
    where Logic_Rule_Type =: 'cal';
RUN;

DATA Recs_Calc;        /* Interim Output with new elements created */
 set Recs;
 length FEA_BRM_Description $300. Agency Bureau $60.;
    %do n = 1 %to &Logic_Rule_Order; /* &n iterates thru each calcuation logics */
    /* Validation IF-THEN-ELSE */
    %IF "&&Logic_Rule_Type&n" = "cal_assign" %THEN %DO;
      &&Element&n = &&Logic_Rule_Return&n;
    %END;

    /* Additional Codes */
    /* Additional Codes */
```

It can be quickly noted the similarity of the coding in parsing and executing the transformation logics demonstrated earlier in previous stage. The design and programming using control data set encourages re-use of modular programming without rewriting repetitive codes. One might further macroize the modularized code as function oriented macros in a pseudo object oriented programming scheme. The shown SAS code generates the complete syntax with the calculation condition and results for each transformation rule for each element specifications.

Figure 4-E shows the partial output due from the transformation and creation stage. The official Agency and Bureau name is created with standardized naming convention, which is helpful for data consumers. The FEA_BRM_DESCRIPTION is now a concatenation of multiple FEA_BRM fields that was previously stored under four separated fields, the ITSpending field is calculated from the sum of DME and OM funding and contributions, and along with the calculation of over-the-year level and percentage changes. Not shown: a category flag is created to show investment item associated with 'Product X' through text search.

Figure 4-E Initialization Stage: Output

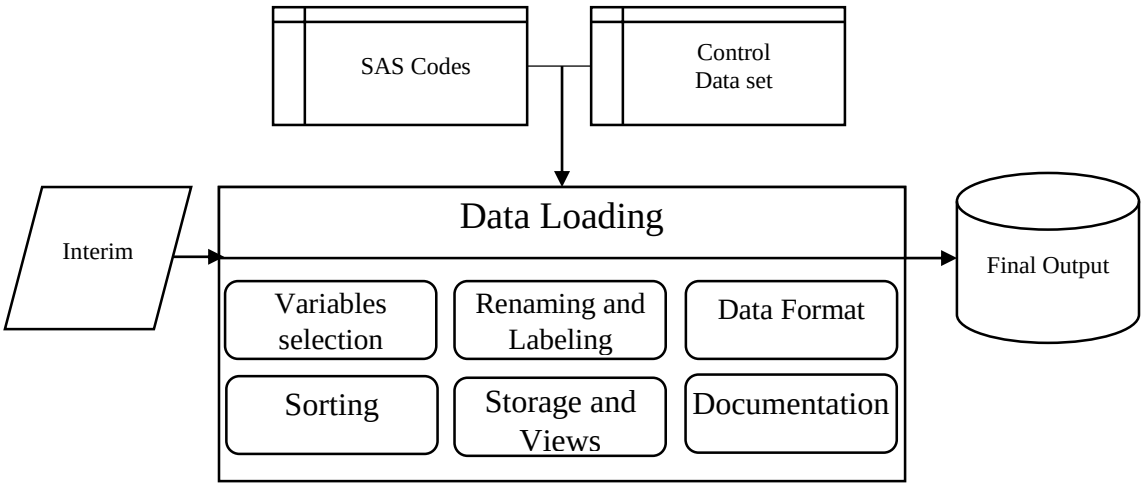
Total rows: 18 Total columns: 40 Rows 1-18

	UII	AgencyCAgency		BureauCBureau		ITPortfolioPart	InvestmentInvestmentTitle		FEA_BRM_DESCRIPTIO	FEA_BRM_PI	TSpending_FY2014	ITSpending_FY2015	ITSpending_OT
1	005-000000095	5	Department of Agriculture	68	Foreign Agricultural Service	01 - IT Investments for Mission Delivery and Management Support Area	02 - Non Major	FAS International Funds Control Reporting System (IFCRS)	129 - Reporting and Information	129 - Reporting and Information	0.013	0	-0.013
2	007-000100033	7	Department of Defense	97	Defense-wide	01 - IT Investments for Mission Delivery and Management Support Area	01 - Major	DoD Healthcare Management System Modernization	248 - Health Care Delivery Services	248 - Health Care Delivery Services	29.602	148.948	119.346
3	009-000000935	9	Department of Health and Human Services	90	Departmental Management	01 - IT Investments for Mission Delivery and Management Support Area	01 - Major	OS ASA Perry Point Local Area Network Services	139 - Provide and Maintain IT Infrastructure	139 - Provide and Maintain IT Infrastructu	0.3	0	-0.3
4	009-000325766	9	Department of Health and Human Services	10	Food and Drug Administration	02 - Infrastructure, IT Security, Office Automation, and Telecommunications	02 - Non Major	FDA - Email	659 - Email	659 - Email	5.557	5.557	0

DATA LOADING AND DOCUMENTATION

The use of control file facilitates the loading of the validated and transformed data to the target data warehouse. The benefit of using a control file allows the flexibility for both the developers and end users to control the elements and output source location especially for complex outputs. While a data warehouse might wish to keep all variables if efficiency and storage space is not at conflict. However when the data are used for reporting or BI dashboards, probably storing a separate data set or maintaining a data view could improve efficiency. Also in conjunction with data views, control file can store information on access permission which could enforce security authorization model for user access group (such as by Departments, by functions, by role) to complement enterprise database security permission model.

Figure 5-A Data Loading workflow



In the last part, the goal is to perform final variable selection, renaming and labeling, and applying data formats for the calculated output from the Transformation stage. The last additions are manually entered to the final version of the control file, shown in Figure 5-C, to include selection of output variables and proper formats. The control file includes a new column:

- **Output** – the order of the output variables and a flag indicating the columns to output

Figure 5-B Initialization Stage: Input Data

Total rows: 18 Total columns: 40 Rows 1-18

	UII	AgencyC	Agency	BureauC	Bureau	ITPortfolioPart	Investment	InvestmentTitle	FEA_BRM_DESCRIPTIO	FEA_BRM_PI	TSpending_FY2014	ITSpending_FY2015	ITSpending_OT
1	005-000000095	5	Department of Agriculture	68	Foreign Agricultural Service	01 - IT Investments for Mission Delivery and Management Support Area	02 - Non Major	FAS International Funds Control Reporting System (IFCRS)	129 - Reporting and Information	129 - Reporting and Information	0.013	0	-0.013
2	007-000100033	7	Department of Defense	97	Defense-wide	01 - IT Investments for Mission Delivery and Management Support Area	01 - Major	DoD Healthcare Management System Modernization	248 - Health Care Delivery Services	248 - Health Care Delivery Services	29.602	148.948	119.346

The SAS code resembles the similar code from the Initialization and Integration stage with the use of PROC SQL code to parse through the element selection indicated in the control file and storing them as macro variables. The DATA STEP recalls the macro variables for the associated functions. The SAS codes picks up the output selection variables where OUTPUT is not null, indicating the variables to keep. As each element is being read, macro variables are created:

- **&keepvar** –list of new output variables to keep and ordering of variables
- **&labelvar** - list of elements with desired in proper syntax
- **&formatvar** - list of elements and formats in with proper syntax

Figure 5-C Initialization Stage: Control File

Obs	Input	Output	Element	Label	Format	Logic_Rule_Description	Logic_Rule_Type	Logic_Rule_Condition	Logic_Rule_Return	Type	Length	InputElement
1	1	1	UII	Unique Investment Identifier	$13.	Validate the format as ###-#########	val_if_then_else	length(UII)=13 and notdigit(substr(UII,1,3))=0 and substr(UII,4,1)='-'	Delete	Char	13	UNIQUE_INVESTMENT_IDENTIFIER
2	2	2	UII_Prev	Previous UII	$209.					Char	209	PREVIOUS_UII
3	3	3	BudgetYear	Budget Year	BEST12.	Validate budget year as 2015; if failed, assign 2015	val_if_then_else	BudgetYear = 2015	BudgetYear = 2015	Numeric	8	BUDGET_YEAR
4	4	4	Status	Derived Status	$28.	Auto assign status information if missing	val_if_then	Status = ''	Status = 'No change in status'	Char	28	DERIVED_STATUS
5		5	Agency	Agency	$60.	Auto assign name by Agency Code	cal_macrocall	%gencode(runmode=assign,codetype=agencyc		Char	60	

Prior to the execution of the DATA STEP, the macros are resolved to the list of actual data elements names, formats, and labels. During the DATA STEP processing, the final output data gets the proper formatting and labeling treatment. Variables not in use are dropped. This greatly enhance the final data to an organized and user-friendly structure to interpolate with reporting, tabulations, and visualization solutions.

Figure 5-D Initialization Stage: SAS Code

```
/* Create the mvar for list of variables to keep, format, and label
   for output is flagged */
PROC SQL noprint;
  select Element, catx(' ', Element, Format),compress(Element||"='"||Label||"'")
  into :keepvar separated by ' ', :formatvar separated by ' ', :labelvar separated by ' '
  from mydata.&controlfile
  where Output is not null
  order by Output;
QUIT;

%PUT List of Data Elements to keep: &keepvar;
%PUT List of Data Elements for formatting: &formatvar;
%PUT List of Data Elements for labeling: &labelvar;

DATA mydata.itportfolio_final;  /* Final output in analysis-ready format for perman
  format &formatvar;
  set recs_calc;
  keep &keepvar;
  label &labelvar;
RUN;
```

The final control data set has evolved from a simple data set metadata information table to a full data dictionary, documenting the input, outputs, and business rules as used throughout the ETL process. The control file is a self-reference documentation material. For some industries, the data dictionary might supplement compliance documents on modeling parameters, calculation methods, formulas for variables, and inputs/outputs. The data dictionary can be versioned to track the changes over software development cycle and requirement changes. The control file is an efficient medium to share business rules and understand ETL functions between developers and business group.

Figure 5-E Initialization Stage: Output

Total rows: 18 Total columns: 20 Rows 1-18

	UII	UII_Prev	BudgetYear	Status	Agency	Bureau	InvestmentCateg	ITPortfolioPart	MDMSAC	InvestmentTy	InvestmentTitle	InvestmentDescription
1	005-000000095	005-000000095	2015	No change in status	Department of Agriculture	Foreign Agricultural Service	00 - Agency Investments	01 - IT Investments for Mission Delivery and Management Support Area	1	02 - Non Major	FAS International Funds Control Reporting System (IFCRS)	Supports FAS' Strategic plan to implement Pr supplements USDA/OCFO's FMMI to meet FA of budgetary and programmatic control of fun accuracy of financial reporting.
2	007-000100033		2015	New	Department of Defense	Defense-wide	00 - Agency Investments	01 - IT Investments for Mission Delivery and Management Support Area	17	01 - Major	DoD Healthcare Management System Modernization	DHMSM program will acquire and support de sustainment of a commercial electronic health replaces DoD legacy Military Health System (I

CONCLUSION

Designing and developing an ETL process with high-degree of maintainability is becoming increasing challenging as data continues to grow in size and complexity. The application of control data set in particular useful for SAS programming language, especially for data processing routines and sequential processing. Paring control data set and efficient SAS codes can improve modularized programming and reduce writing repetitive codes. Subject domain logics becomes a part of a self-documented file readable by the processing system, users can self-maintain and perform update as requirement changes. Developer and Analyst can leverage each other comparative strengths in building higher quality products and quicker development cycle. The control data set also supports end users in understanding the products and methodology behind.

REFERENCES

IT Dashboard. 2015, "IT Dashboard FY2015 Edition." Accessed March 1, 2015. https://itdashboard.gov/.

Cheng, Edmond. 2008. "Better, Faster, and Cheaper SAS® Software Lifecycle." *Proceedings of the NESGU 2008.* Pittsburgh, PA. Northeast SAS User Group. http://www.nesug.org/proceedings/.

Hughes, Troy. 2014. "Reliably Robust: Best Practices for Automating Quality Assurance and Quality Control Methods." *Proceedings of the SESUG 2014.* Myrtle Beach, SC. Southeast SAS User Group. http://www.sesug.org/SESUG2014/.

Schlegelmilch, Gary E. 2014. "How to Build a Data Dictionary – In One Easy Lesson." *Proceedings of the SESUG 2014.* Myrtle Beach, SC. Southeast SAS User Group. http://www.sesug.org/SESUG2014/.

CONTACT INFORMATION

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

Edmond Cheng
CACI International Inc.
edmondcheng@outlook.com