Paper 101-2009
Using Data Integration Studio as a Computer Aided Software Engineering (CASE) Tool

David Kiasi, Applications Alternatives

ABSTRACT

With SAS® DI Studio, a SAS developer can now design his/her application, fully documenting it on the front end, in a structured manner; then migrate that design to the application with minimal effort. DI Studio can, if used correctly, create a structured environment in a painless manner. This presentation will discuss the fundamentals of structured analysis and design, show how data flow diagrams and data dictionaries can be created with DI Studio, and show how easy it is to migrate that design to the application.

WHAT IS A SYSTEM LIFE CYCLE?

What is a System Life Cycle? Definition: "The course of a program or system from the inception of the original idea through development, implementation, and maintenance, until it is either replaced or no longer valid." 1 Essentially, these are the steps we go through to create a final computer application.

HOW SYSTEM LIFE CYCLES ARE DONE

1. **Instant Coding.** Features of this approach are:

- No paper trail
- Documentation only in the code (or worse yet, the code itself.)
- No one understands code but author.
- If the specs are interpreted wrong, then re-coding and re-testing must be done
- Limited testing-no hand off
- Changes often are made after production has started-introduces errors.
- No system design-it is an implied part of coding.
- Files are typically created on the fly-result is that some required fields could easily be left out or formatted wrong.

2. **Specs, Then Coding.**

- Specs are clearly defined- they serve as limited documentation
- Documentation only in the code (or worse yet, the code itself.)
- No one understands code but author.
- Limited testing-no hand off.
- Changes often are made after production has started- however, because specs were clearly defined , less errors are introduced.
- No system design-it is an implied part of coding.
- Files are typically created on the fly-result is that some required fields could easily be left out or formatted wrong. However, this is less likely to happen because the clear specs indicate exactly what fields should be the result of the application.
- A code walkthrough is the only time other project team members get to review any design or coding issues before the code is tested but at least there is a review.

3. **Structured System Life Cycle Development**

- Requirements Analysis involves Data Flow Diagrams
- Specs are clearly defined
- There is a formal design phase. The design acts as high level documentation
- Code comes after the design is completed and reviewed by peers.
- More time consuming
- No incentive to update design after development is done.

WHAT IS CASE?

What is Case? CASE is Computer-Aided Software Engineering. "A combination of techniques and tools aimed at building and maintaining software systems of all types--large and small, commercial and scientific on-line and read-time. CASE tools provide coverage of the software life cycle by providing auto analysis, design, implementation, and maintenance, as well as project management of software systems." 1

There are no popular CASE tools for SAS development.

Immediate advantages of a Case tool are that they:

- Allow the design to write the code.
- Combine design and development steps.
- Allow design documentation to be current-the design is part of the development.

SAS DATA INTEGRATION STUDIO

SAS Data Integration (DI) Studio is a SAS product that is part of the SAS Business Intelligence software. Data Integration Studio is primarily used for taking data out of a Data Warehouse and editing, transforming, and filtering data for use by end users. It is object oriented. Each step in the process is determined by process icons. Every data store is pre-defined. Although knowledge of coding SAS is helpful, one doesn't need to know how to code SAS programs to develop applications with DI Studio.

DI Studio has several features that lend itself to structured development. These features force a discipline on the developer that Base SAS code does not. DI Studio does this by:

- For each major step in the development process (DI Studio Job), forcing the user to define input and output data stores before any other development work is done. This is an intricate part of structured development, where data flows are the key component.
- When doing development, forces the user to create process icons within Jobs. These process icons are chosen from a Process Library. Examples are SQL, Extract, Transform, and Load. This forces the user to define each step by its function.

As I will show below, DI Studio can be used as a CASE tool for SAS development.

How can we use DI Studio as a CASE tool? My approach is to do a design with DI Studio first, ignoring the details that are necessary for development. Once the design is completed, one can generate documentation from DI Studio. This allows peers to review the documentation before development begins. Once development starts, it is not done from scratch because many of the objects needed for development have already been completed in the design phase.

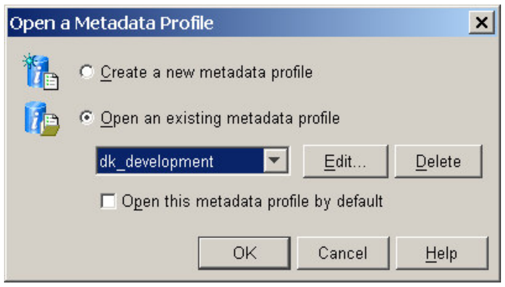
DI STUDIO IN THE DESIGN PHASE

DI Studio is based on data flows so it enforces good practices for structured development.

1. Designing source data stores.

DI Studio information is stored in Metadata. This Metadata is a part of the SAS BI environment. As good structured analysis and design teaches us, we must determine what data we're going to start with and how it flows toward the data we want to end up with. So first, we need to design our source data store. In this example, my source data consists of a simple SAS dataset.

a. When I first enter DI Studio, I must open a Metadata profile. Metadata is stored in a series of profiles. In this case, I'm opening my personal profile, "dk_development."



Open a Metadata Profile

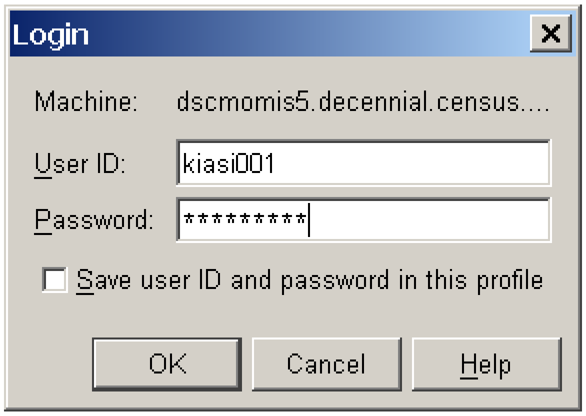
( ) Create a new metadata profile

(•) Open an existing metadata profile

dk_development

[ ] Open this metadata profile by default

b. Then, I must log onto the server through DI Studio.



Login

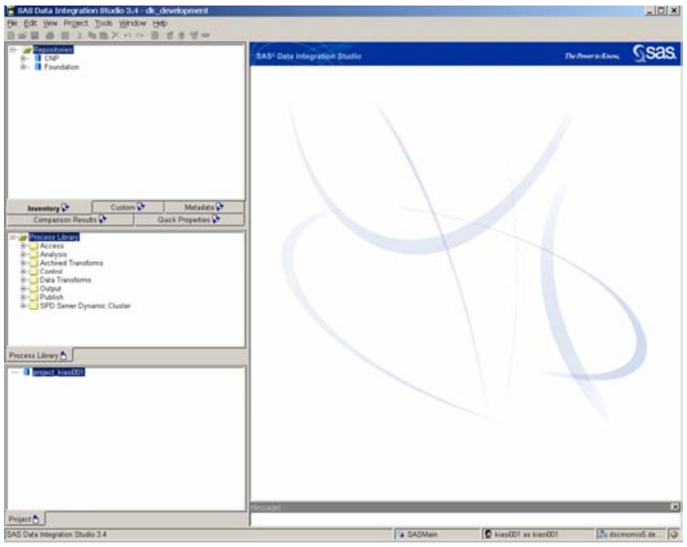
Machine: dscmomis5.decennial.census....

User ID: kiasi001

Password: *********

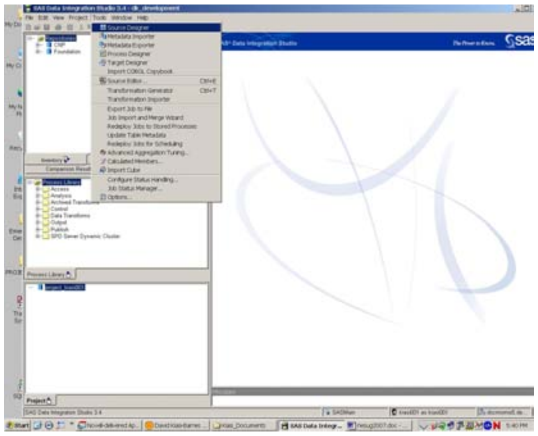
[ ] Save user ID and password in this profile

c. Below is the Metadata tree. This is what comes up after one logs on.

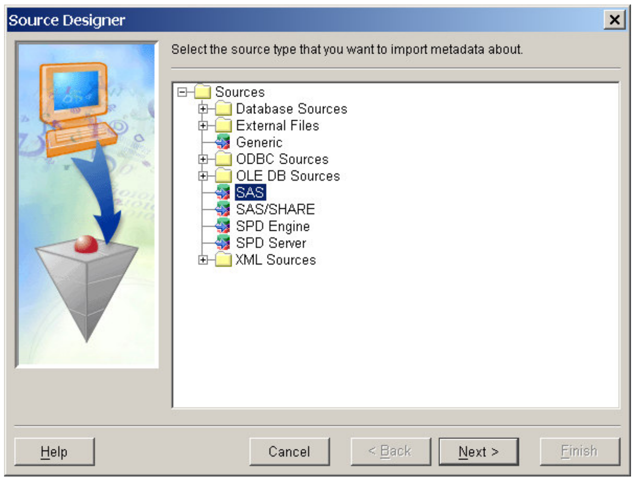


SAS Data Integration Studio 3.4 - dk_development

d. You click the “Tools” pull down menu. In order to create a source within the Metatdata, one clicks “Source Designer.”



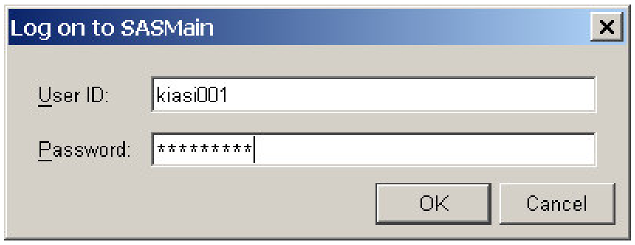
e. You must select the source type here. It is the structure of your source data store. In this case, I'll choose SAS because it is a SAS dataset.


Source Designer

Select the source type that you want to import metadata about.

- Sources
  - Database Sources
  - External Files
  - Generic
  - ODBC Sources
  - OLE DB Sources
  - SAS
  - SAS/SHARE
  - SPD Engine
  - SPD Server
  - XML Sources

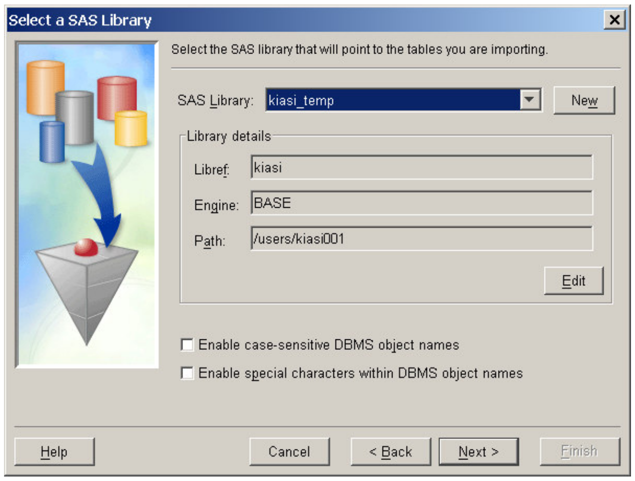
f. DI Studio prompts you to log on again. This is to gain access to the server where SAS metadata exists.


Log on to SASMain

User ID: kiasi001

Password: *********

g. Before I started DI Studio, I should have created a Metadata entry for the library that I am using in my personal area. This library, “kiasi_temp”, represents the directory on the UNIX workstation where my target dataset is stored.


Select a SAS Library

Select the SAS library that will point to the tables you are importing.

SAS Library: kiasi_temp

Library details

Libref: kiasi

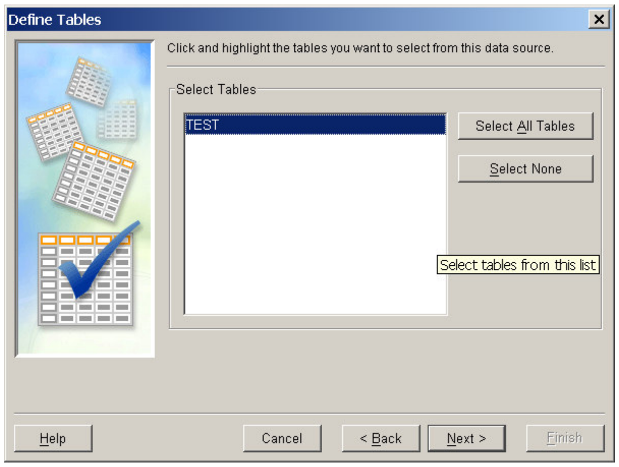
Engine: BASE

Path: /users/kiasi001

☐ Enable case-sensitive DBMS object names

☐ Enable special characters within DBMS object names

h. Within that directory, I am going to choose dataset “TEST”. This is my source dataset.


Define Tables

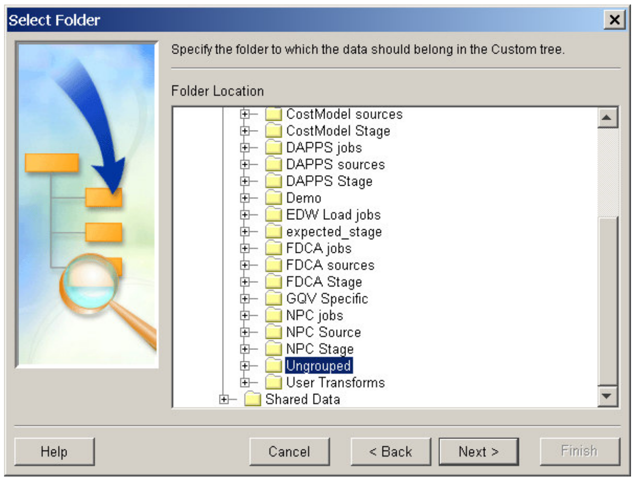
Click and highlight the tables you want to select from this data source.

Select Tables

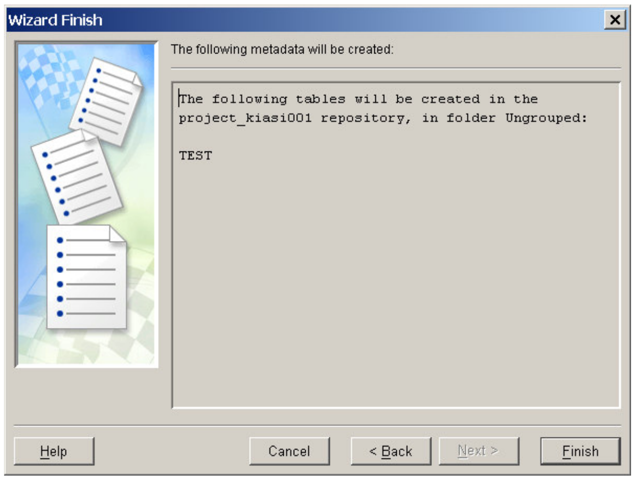
TEST

Select tables from this list

i. In the Metadata tree, I have to choose a folder where the metadata belongs. I choose, “Ungrouped”, since this is a demo and not a permanent entry for our project.

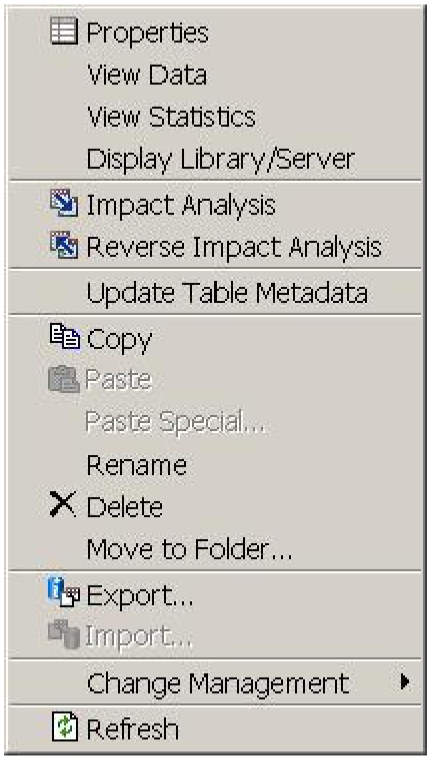


j. I click “Finish” on this window to complete the creation of the Metadata.

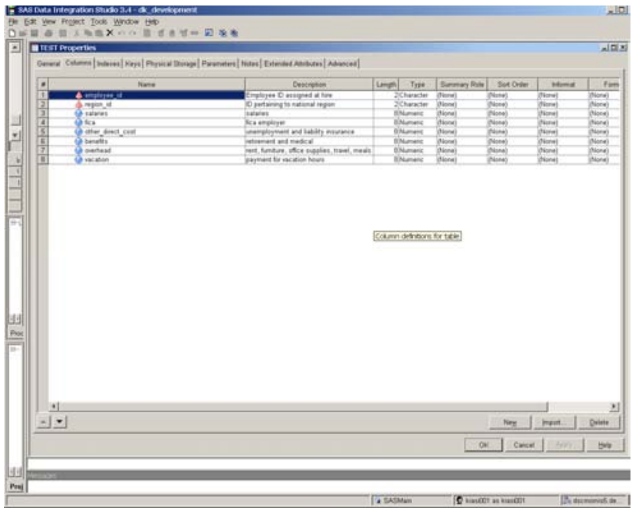


```
The following tables will be created in the
project_kiasi001 repository, in folder Ungrouped:

TEST
```

k. By right clicking the entry in the Metadata tree for “TEST”, I can pull up this pop-up, and choose, “Properties”, to view my Metadata for this dataset.



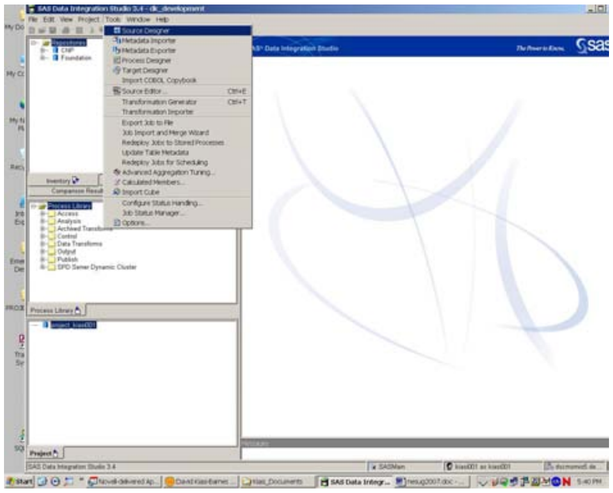
l. As you can see below, DI Studio has generated, from my SAS dataset, a well defined display of my data fields and their attributes.


SAS Data Integration Studio 3.4 - dk_development
File Edit View Project Tools Window Help
TEST Properties
General Columns Indexes Keys Physical Storage Parameters Notes Extended Attributes Advanced
Column definitions for table
New Import Delete
OK Cancel Apply Help

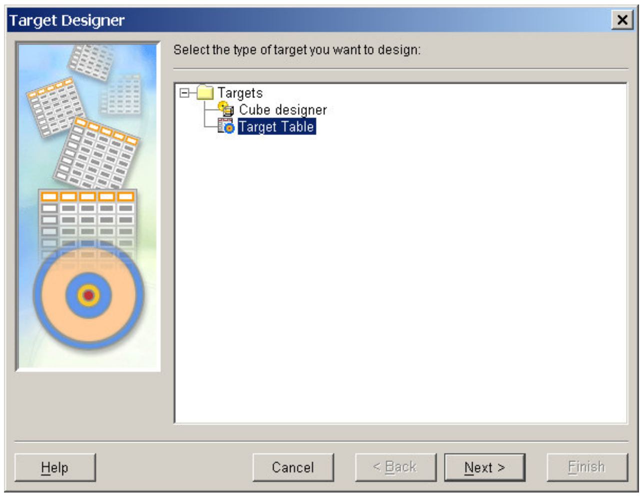
#	Name	Description	Length	Type	Summary Role	Sort Order	Informat	Form
1	employee_id	Employee ID assigned at hire	2	Character	(None)	(None)	(None)	(None)
2	region_id	ID pertaining to national region	2	Character	(None)	(None)	(None)	(None)
3	salaries	salaries	8	Numeric	(None)	(None)	(None)	(None)
4	fica	fica employer	8	Numeric	(None)	(None)	(None)	(None)
5	other_direct_cost	unemployment and liability insurance	8	Numeric	(None)	(None)	(None)	(None)
6	benefits	retirement and medical	8	Numeric	(None)	(None)	(None)	(None)
7	overhead	rent, furniture, office supplies, travel, meals	8	Numeric	(None)	(None)	(None)	(None)
8	vacation	payment for vacation hours	8	Numeric	(None)	(None)	(None)	(None)

2. Next, I will create my Target dataset. DI Studio requires you define your Target dataset in Metadata first. Then, when you populate the Target dataset in Metadata, it creates the actual SAS dataset on the server.

a. First, I click the Tools pull-down menu and click on "Target Designer."



b. The first window comes up for the Target Designer.

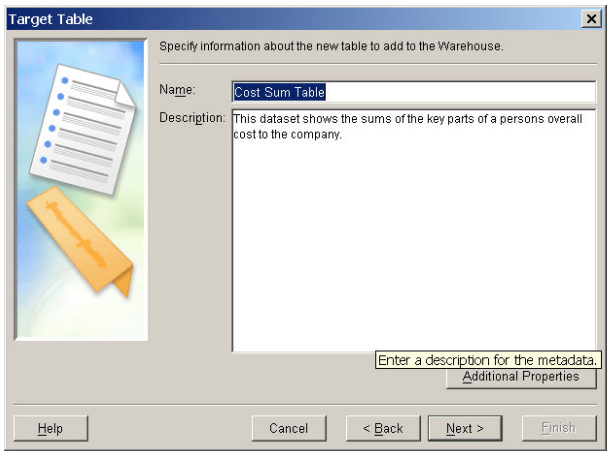

Target Designer

Select the type of target you want to design:

- Targets
  - Cube designer
  - Target Table

I click "Target Table" to create a target dataset. The other option is to create an OLAP Cube.

c. The next window, "Target Table", comes up.


Target Table

Specify information about the new table to add to the Warehouse.

Name: Cost Sum Table

Description: This dataset shows the sums of the key parts of a persons overall cost to the company.

Enter a description for the metadata.

On the Target table window I choose my Metadata data store name. Next to "Name", I choose, "Cost Sum Table".

Also, I have a chance to enter a description. Here, I make a design decision, by describing what the dataset's function will be.

d. The "Table Storage Information" window comes up next.

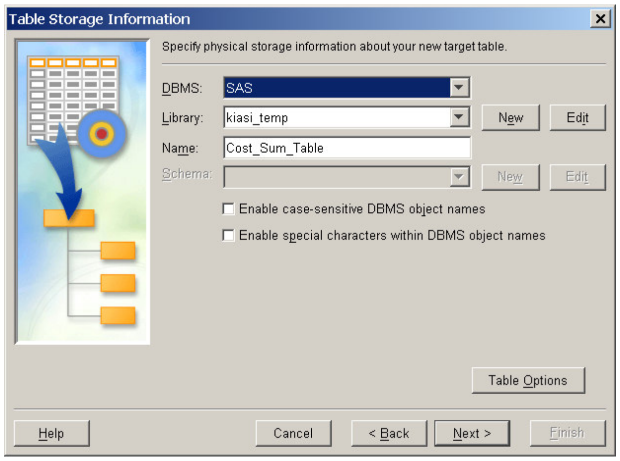


Table Storage Information

Specify physical storage information about your new target table.

DBMS: SAS

Library: kiasi_temp

Name: Cost_Sum_Table

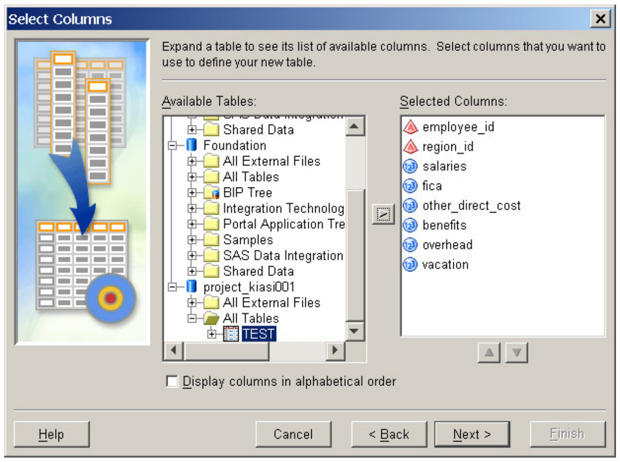
Schema:

☐ Enable case-sensitive DBMS object names

☐ Enable special characters within DBMS object names

Here, I declare my Metadata library, which is "kiasi_temp", which I created before. I can name my SAS dataset here. I call it "Cost_Sum_Table" which will be stored as Cost_Sum_Table.sas7bdat on the UNIX server.

e. The “Select Columns” window comes up.


Select Columns

Expand a table to see its list of available columns. Select columns that you want to use to define your new table.

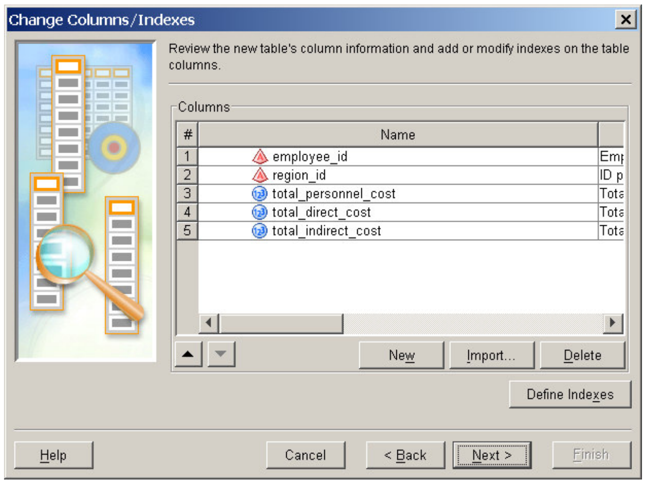
Available Tables:
- Shared Data
- Foundation
  - All External Files
  - All Tables
  - BIP Tree
  - Integration Technolog
  - Portal Application Tre
  - Samples
  - SAS Data Integration
  - Shared Data
- project_kiasi001
  - All External Files
  - All Tables
    - TEST

Selected Columns:
- employee_id
- region_id
- salaries
- fica
- other_direct_cost
- benefits
- overhead
- vacation

Display columns in alphabetical order

Here, DI Studio gives me the option of creating data fields from other datasets. If the fields are similar, then the amount of work that I have to do, particularly when it comes to field descriptions, can be minimized. In a typical design environment, field information is entered by typing text entries into a document.

f. The “Change Columns/Indexes” window appears.


Change Columns/Indexes

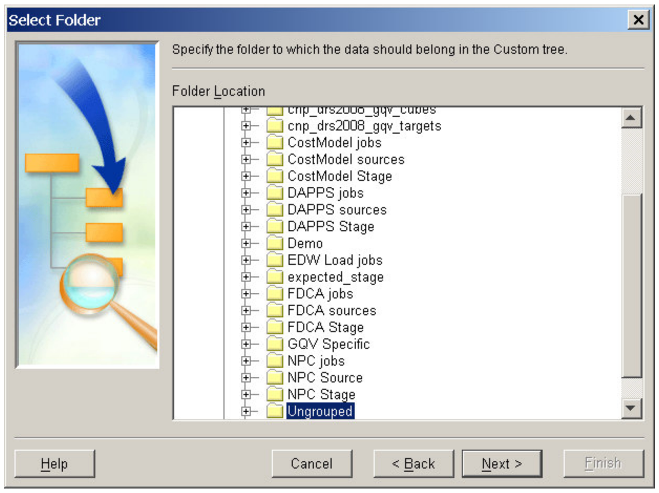
Review the new table's column information and add or modify indexes on the table columns.

Columns

#	Name	
1	employee_id	Emp
2	region_id	ID p
3	total_personnel_cost	Tota
4	total_direct_cost	Tota
5	total_indirect_cost	Tota

Here, I can create new fields for my dataset or I can modify the description or field names for the fields I have copied from other datasets’ Metadata.

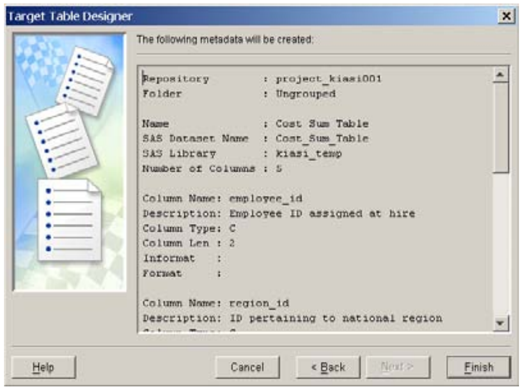
g. On the “Select Folder” window, one is asked where they want to put the target’s Metadata. These folders are set up in advance by the project team. “Ungrouped” is an exception and is used as a catch all folder.


Select Folder

Specify the folder to which the data should belong in the Custom tree.

Folder Location
- cnp_drs2008_gqv_cubes
- cnp_drs2008_gqv_targets
- CostModel jobs
- CostModel sources
- CostModel Stage
- DAPPS jobs
- DAPPS sources
- DAPPS Stage
- Demo
- EDW Load jobs
- expected_stage
- FDCA jobs
- FDCA sources
- FDCA Stage
- GQV Specific
- NPC jobs
- NPC Source
- NPC Stage
- Ungrouped

h. The “Target Table Designer” window, is the last window before the target data store is created. Again, this will not create the dataset on the UNIX server.

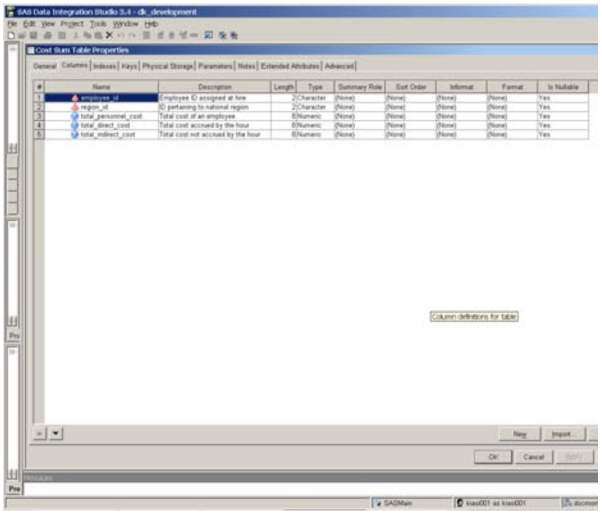

Target Table Designer

The following metadata will be created:

```
Repository          : project_kiasi001
Folder              : Ungrouped

Name                : Cost Sum Table
SAS Dataset Name    : Cost_Sum_Table
SAS Library         : kiasi_temp
Number of Columns : 5

Column Name: employee_id
Description: Employee ID assigned at hire
Column Type: C
Column Len : 2
Informat   :
Format     :

Column Name: region_id
Description: ID pertaining to national region
```

Our target is now created in the Metadata.

i. Target Properties Window.



SAS Data Integration Studio 3.4 - dk_development

Cost Sum Table Properties

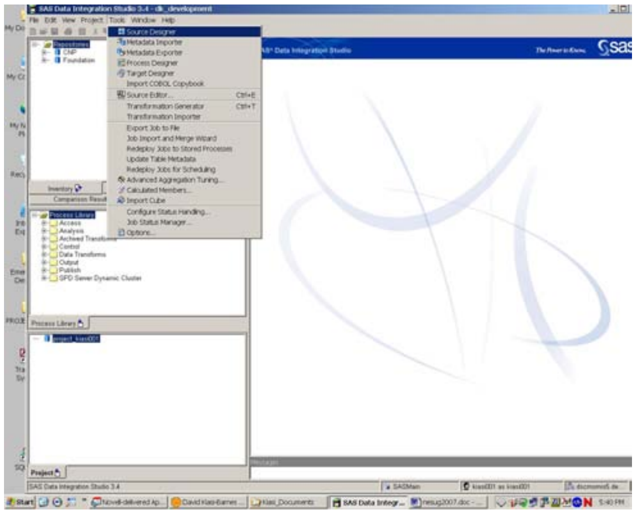
General | Columns | Indexes | Keys | Physical Storage | Parameters | Notes | Extended Attributes | Advanced

#	Name	Description	Length	Type	Summary Role	Sort Order	Informat	Format	Is Nullable
1	employee_id	Employee ID assigned at hire	2	Character	(None)	(None)	(None)	(None)	Yes
2	region_id	ID pertaining to national region	2	Character	(None)	(None)	(None)	(None)	Yes
3	total_personnel_cost	Total cost of an employee	8	Numeric	(None)	(None)	(None)	(None)	Yes
4	total_direct_cost	Total cost accrued by the hour	8	Numeric	(None)	(None)	(None)	(None)	Yes
5	total_indirect_cost	Total cost not accrued by the hour	8	Numeric	(None)	(None)	(None)	(None)	Yes

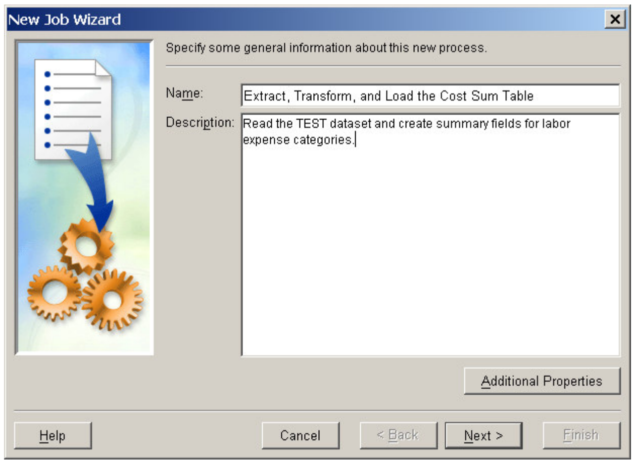
By right clicking the entry in the Metadata tree for this target dataset, we can pull up the "Columns" tab, showing us the fields that make up our target.

3. Now, we want to create our Data Flow Diagram using DI Studio.

a. We first go back to the "Tools" pull down menu and this time we click on "Process Designer".



b. The "New Job Wizard" window appears.



New Job Wizard

Specify some general information about this new process.

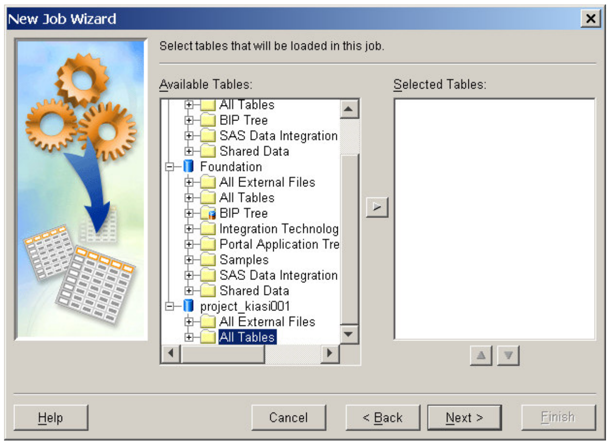
Name: Extract, Transform, and Load the Cost Sum Table

Description: Read the TEST dataset and create summary fields for labor expense categories.

Here we name our "Job". A DI Studio Job is a set of one or more Processes that make up a flow of data from one permanent data store to another. We may have interim data stores but these are not saved anywhere. This structure makes it easy to create a design. Natural breaks in the processing are determined by where data is stored.

Also, we can use the Description field to describe the overall function of the "Job". This description is a part of the design and these will be used by developers to understand how development should proceed.

c. Select Tables Window.

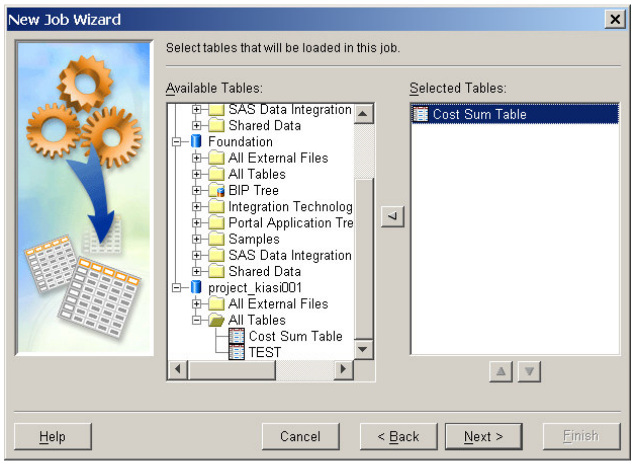


New Job Wizard

Select tables that will be loaded in this job.

On this window, we define what table we are loading as a result of the Job that we're creating. I will go through the Metadata tree to find the dataset that I will populate.

d. Select Tables Window, 2.


New Job Wizard

Select tables that will be loaded in this job.

Available Tables:

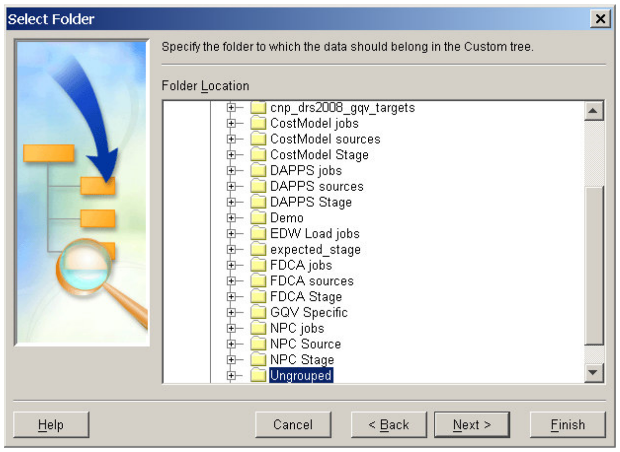
- SAS Data Integration
- Shared Data
- Foundation
  - All External Files
  - All Tables
  - BIP Tree
  - Integration Technolog
  - Portal Application Tre
  - Samples
  - SAS Data Integration
  - Shared Data
- project_kiasi001
  - All External Files
  - All Tables
    - Cost Sum Table
    - TEST

Selected Tables:

- Cost Sum Table

Seeing the "Cost Sum Table", the target data store that I've just defined, I click it and then click the arrow to bring it over to the Selected Tables list.

e. Select Folder Window.


Select Folder

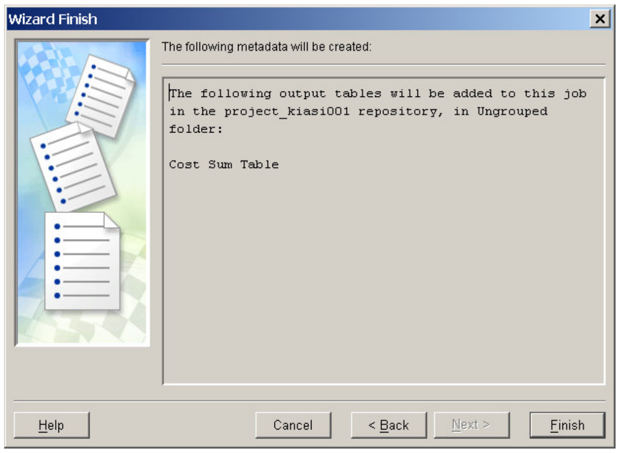
Specify the folder to which the data should belong in the Custom tree.

Folder Location

- cnp_drs2008_gqv_targets
- CostModel jobs
- CostModel sources
- CostModel Stage
- DAPPS jobs
- DAPPS sources
- DAPPS Stage
- Demo
- EDW Load jobs
- expected_stage
- FDCA jobs
- FDCA sources
- FDCA Stage
- GQV Specific
- NPC jobs
- NPC Source
- NPC Stage
- Ungrouped

Next, we determine where the Job should reside once created. One finds a folder in the Metadata tree.

g. The Finish Window.

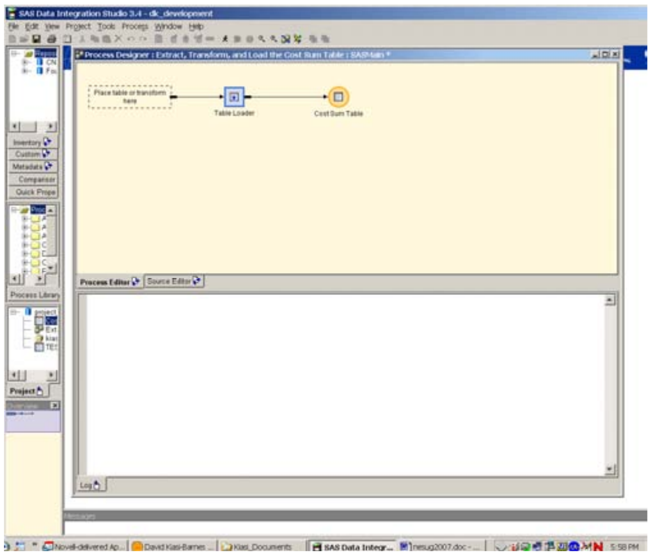

Wizard Finish

The following metadata will be created:

```
The following output tables will be added to this job
in the project_kiasi001 repository, in Ungrouped
folder:

Cost Sum Table
```

This window completes the addition of the target data store to the Job.

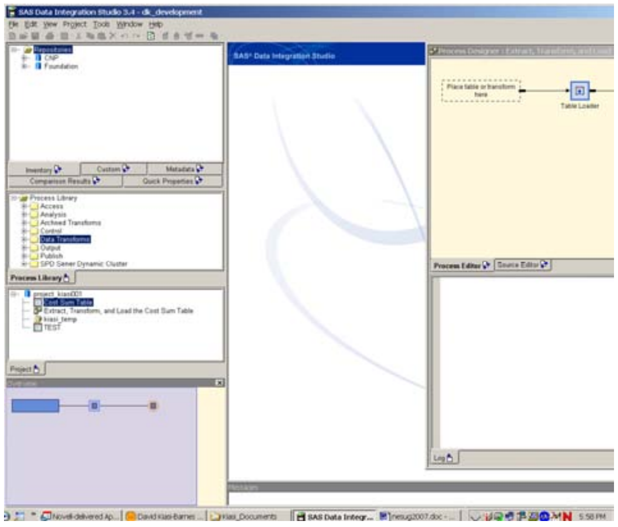
h. Back to the Process Designer Window.


SAS Data Integration Studio 3.4 - dk_development
File Edit View Project Tools Process Window Help
Process Designer : Extract, Transform, and Load the Cost Sum Table : SASMain *
Place table or transform here
Table Loader
Cost Sum Table
Inventory
Custom
Metadata
Comparison
Quick Props
Process Library
Project
Process Editor
Source Editor
Log

With this step in the process completed, we can see that we've started to create our data flow diagram. DI Studio always generates a Load process before a target is loaded.

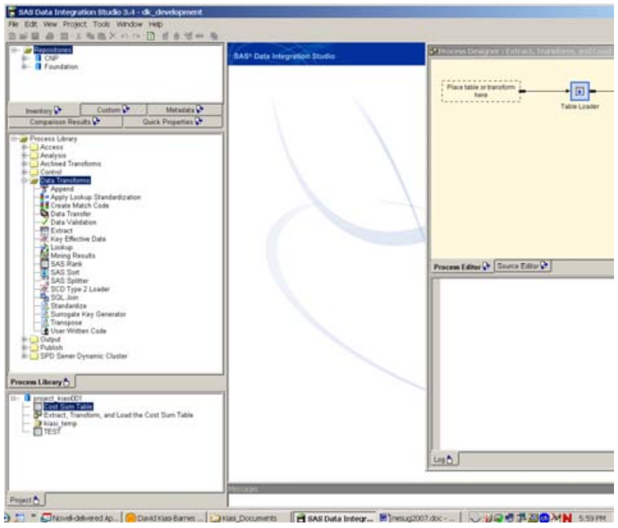
Now, we need to continue to create the Job. Since the target is already in the diagram, I will choose to work backward from the target to the source.

i. Choosing a Process From the Process Library.


SAS Data Integration Studio 3.4 - dk_development
Process Library
Data Transforms
Place table or transform here
Table Loader

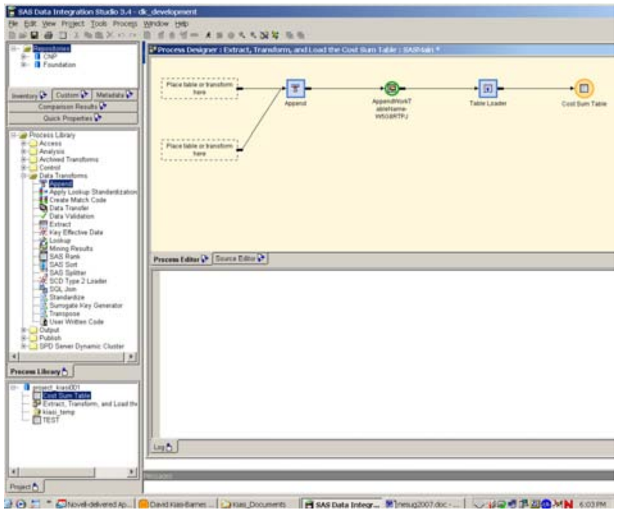
With DI Studio, each process icon is created from a list in the Process Library. The Process library contains any number of operations. Most of the ones that developers will use are in the Data Transforms directory.

j. Displaying the Processes available.

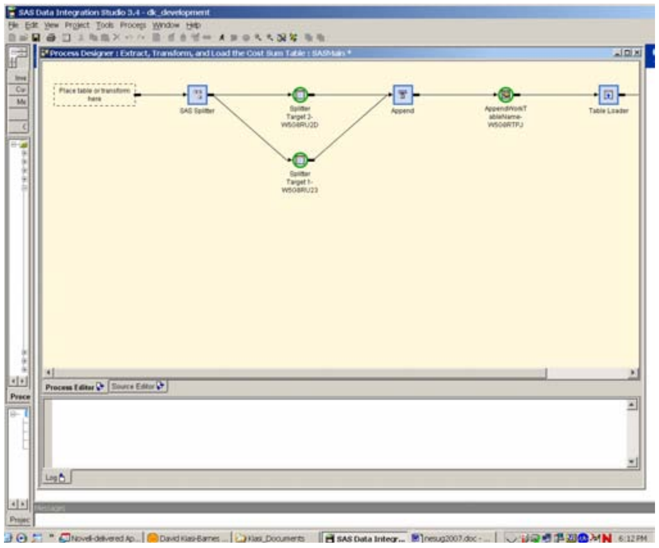

SAS Data Integration Studio 3.4 - dk_development
Process Library
Data Transforms
Append
SAS Splitter
Place table or transform here
Table Loader

By clicking the folder, we can see the list of processes.

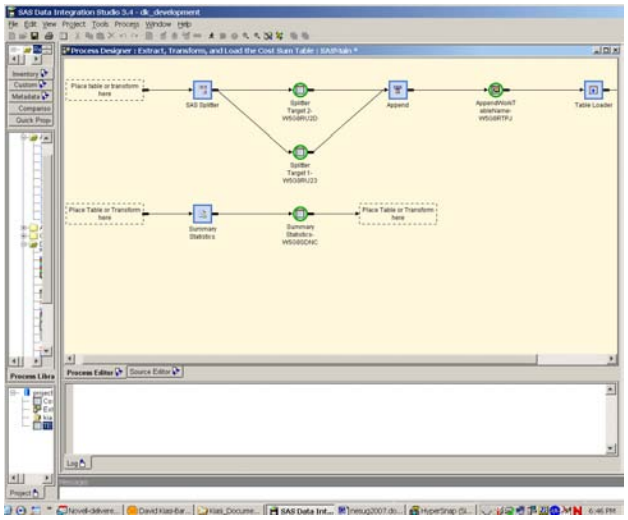
k. Here’s my problem. First, I want to separate my data into two parts, processing each part separately. Then, I want to do special operations on each set of data. After that, I want to combine the data together again. Working from right to left, I click on the “Append” process and move it over to the dotted box to the left of the Table Loader icon. What results can be seen below.


Process Designer : Extract, Transform, and Load the Cost Sum Table : SASMain *
Place table or transform here
Append
Table Loader
Cost Sum Table

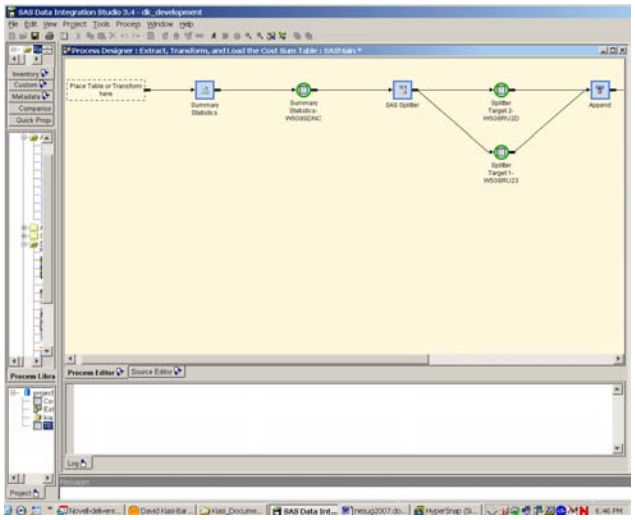
l. Next I use a SAS Splitter icon and move the two green temporary dataset icons over the empty boxes of the Append icon. The result is found below.


Process Designer : Extract, Transform, and Load the Cost Sum Table : SASMain *
Place table or transform here
SAS Splitter
Append
Table Loader

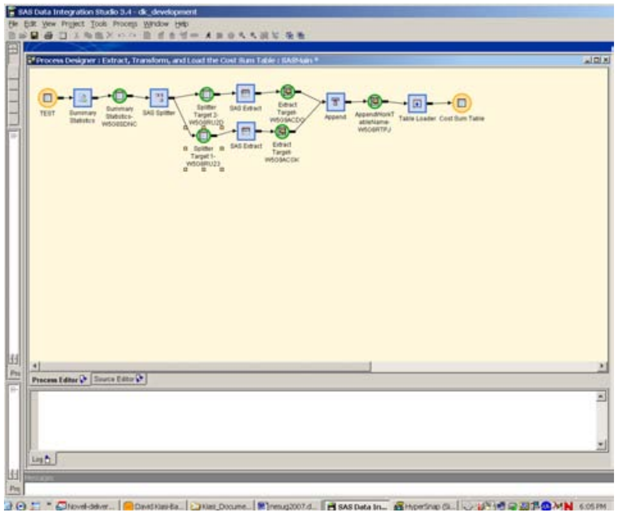
m. Before the data is split, I want to summarize it by Region and Employee ID. To do this, I click and drag over the Summary Statistics icon. The result is below:


SAS Data Integration Studio 3.4 - dk_development
Process Designer : Extract, Transform, and Load the Cost Sum Table : SASMain *
Place table or transform here
SAS Splitter
Splitter Target 2- W5O8RU2D
Splitter Target 1- W5O8RU23
Append
AppendWorkTableName- W5O8RTPJ
Table Loader
Place Table or Transform here
Summary Statistics
Summary Statistics- W5O8SDNC
Place Table or Transform here
Process Editor
Source Editor
Log

n. Now I need to move the green temporary table to the dotted box to the right of the SAS Splitter icon. The result is found below:


SAS Data Integration Studio 3.4 - dk_development
Process Designer : Extract, Transform, and Load the Cost Sum Table : SASMain *
Place Table or Transform here
Summary Statistics
Summary Statistics- W5O8SDNC
SAS Splitter
Splitter Target 2- W5O8RU2D
Splitter Target 1- W5O8RU23
Append
Process Editor
Source Editor
Log

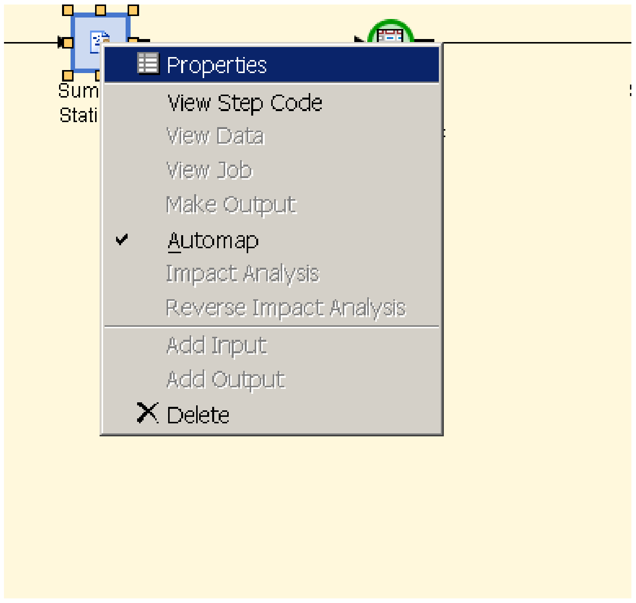
o. I now realize that I need special processes to occur once the data has been split. For Region One, one kind of operation is needed. For the other regions, another operation is needed. Dragging over two SAS Extract icons, splicing them between the SAS Splitter and the Append icons, I get the following Job. Later on, I will delete the SAS Extract for the non-Region One data because I don't need to do anything to it.


SAS Data Integration Studio 3.4 - dk_development
Process Designer : Extract, Transform, and Load the Cost Sum Table : SASMain *
TEST
Summary Statistics
Summary Statistics- W5O8SDNC
SAS Splitter
Splitter Target 2- W5O8RU2D
Splitter Target 1- W5O8RU23
SAS Extract
Extract Target- W5O9ACDO
SAS Extract
Extract Target- W5O9ACDK
Append
AppendWorkTableName- W5O8RTPJ
Table Loader
Cost Sum Table
Process Editor
Source Editor
Log

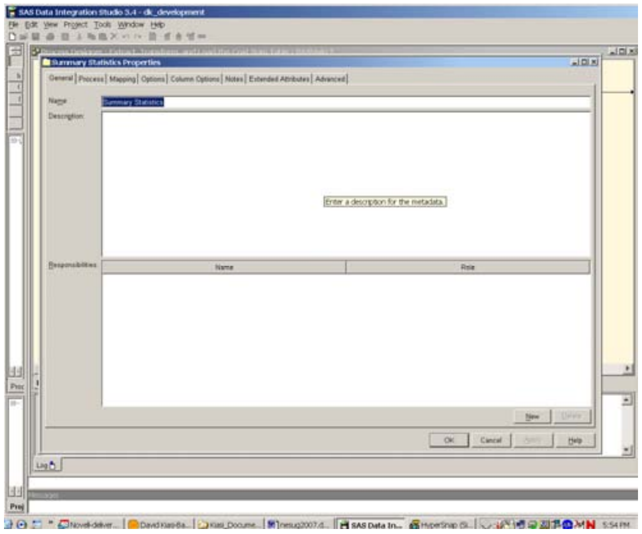
As you can see, DI Studio has allowed me to create a Data Flow diagram. This same Data Flow Diagram will be used by the developer when he/she adds the actual process parameters, mapping, and filtering that will be used in the final application.

Now, we need to embellish our icons with documentation, so that we can tell the developer what to do when it is time for him/her to complete the application.

p. I right click the Summary Statistics icon and get this pop-up window. I click the properties selection.



q. Process Properties Window.

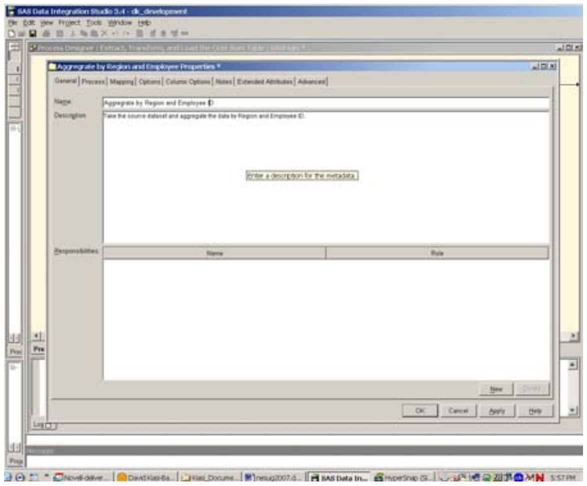


Name: Summary Statistics

The properties window appears. It defaults to the "General" tab.

r. On the window, I will first change the name of my icon from "Summary Statistics" to "Aggregate by Region and Employee ID", which is a much more specific way to label the process.

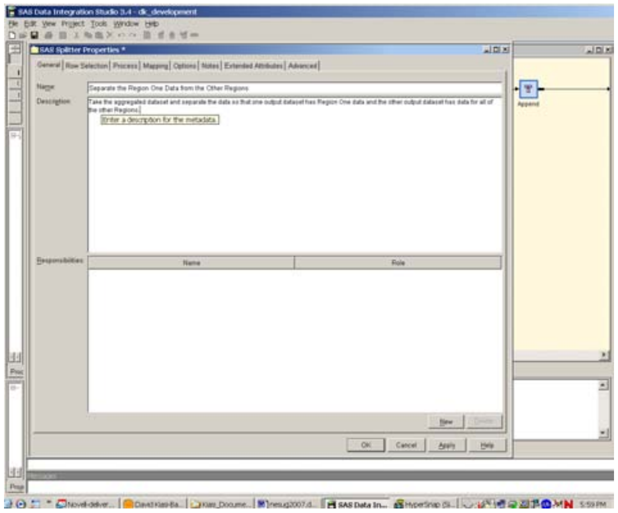
Then, I will use the Description field to enter a brief description of the process so that a developer will have specifications to complete development. This description will be part of the design.



Name: Aggregate by Region and Employee ID

Description: Take the source dataset and aggregate the data by Region and Employee ID.

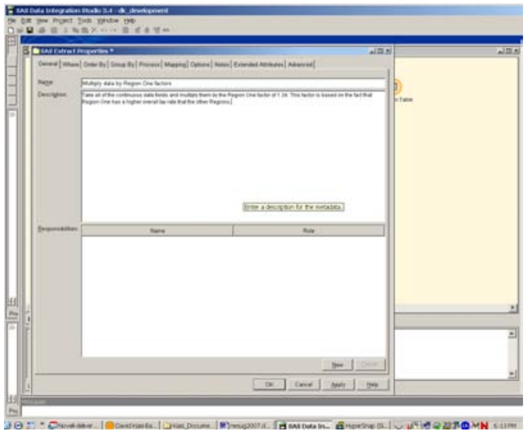
s. Next, I'll use the same process to rename the SAS Splitter icon to "Separate the Region One Data from the Other Regions." Under the description field, I describe what the process does.



Name: Separate the Region One Data from the Other Regions

Description: Take the aggregated dataset and separate the data so that one output dataset has Region One data and the other output dataset has data for all of the other Regions.

t. For the SAS Extract icon, I rename it “Multiply data by the Region One factors”. Under the description, I type the factor and how it will be applied to the continuous data variables.

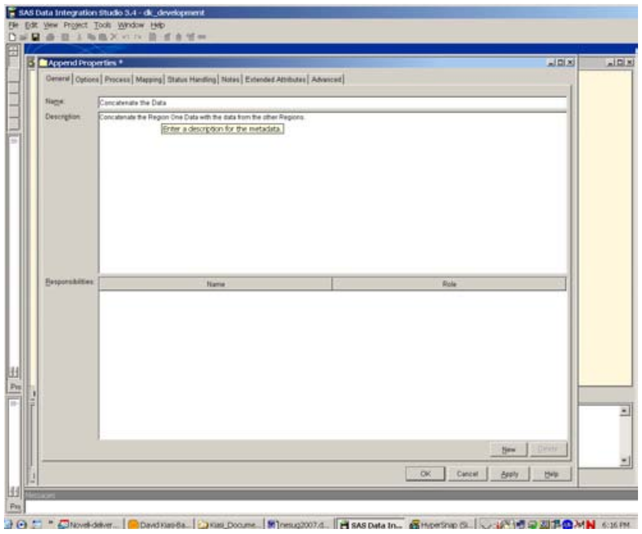

SAS Data Integration Studio 3.4 - dk_development
SAS Extract Properties *
General | Where | Order By | Group By | Process | Mapping | Options | Notes | Extended Attributes | Advanced

Name: Multiply data by Region One factors

Description: Take all of the continuous data fields and multiply them by the Region One factor of 1.36. This factor is based on the fact that Region One has a higher overall tax rate that the other Regions.

Enter a description for the metadata.
Responsibilities
Name
Role
New
Delete
OK
Cancel
Apply
Help

u. For the Append Icon, I rename it “Concatenate the Data” and in the description I describe this process.

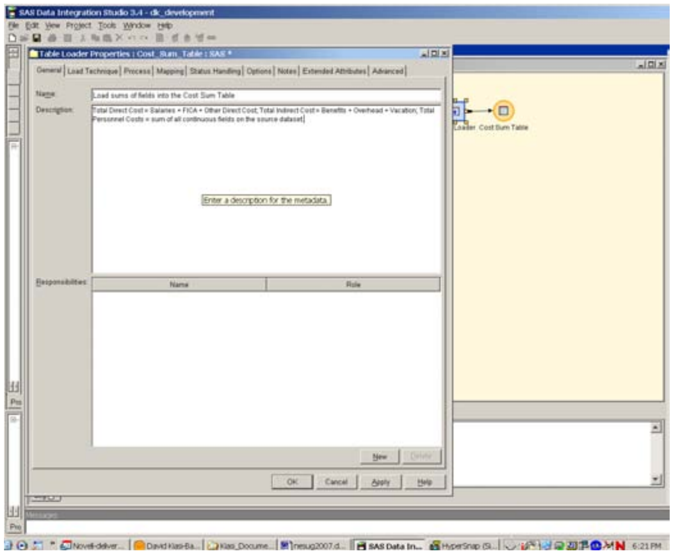

SAS Data Integration Studio 3.4 - dk_development
Append Properties *
General | Options | Process | Mapping | Status Handling | Notes | Extended Attributes | Advanced

Name: Concatenate the Data

Description: Concatenate the Region One Data with the data from the other Regions.

Enter a description for the metadata.
Responsibilities
Name
Role
New
Delete
OK
Cancel
Apply
Help

v. Finally, I modify the Loader icon so that I describe how the final variables are created from the input variables. I rename the icon, “Load sums of fields into the Cost Sum Table.” In the decription, I describe the equation that will be used in the icon to create the new variables.

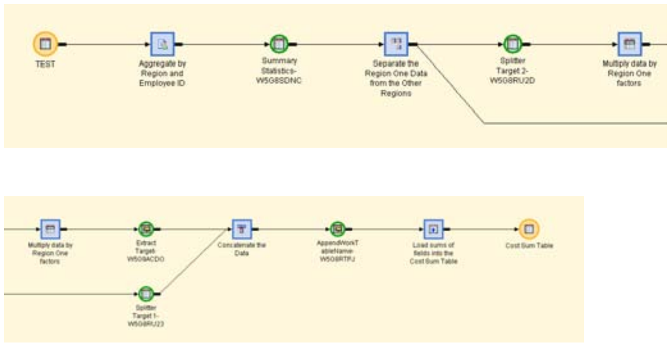

SAS Data Integration Studio 3.4 - dk_development
Table Loader Properties : Cost_Sum_Table : SAS *
General | Load Technique | Process | Mapping | Status Handling | Options | Notes | Extended Attributes | Advanced

Name: Load sums of fields into the Cost Sum Table

Description: Total Direct Cost = Salaries + FICA + Other Direct Cost; Total Indirect Cost = Benefits + Overhead + Vacation; Total Personnel Costs = sum of all continuous fields on the source dataset

Enter a description for the metadata.
Responsibilities
Name
Role
Loader
Cost Sum Table
New
Delete
OK
Cancel
Apply
Help

w. Completed Data Flow Diagram


TEST
Aggregate by Region and Employee ID
Summary Statistics-W5G8SDNC
Separate the Region One Data from the Other Regions
Splitter Target 2-W5G8RU2D
Multiply data by Region One factors
Multiply data by Region One factors
Extract Target-W5G8ACDO
Splitter Target 1-W5G8RU23
Concatenate the Data
AppendWorkTableName-W5G8RTPJ
Load sums of fields into the Cost Sum Table
Cost Sum Table

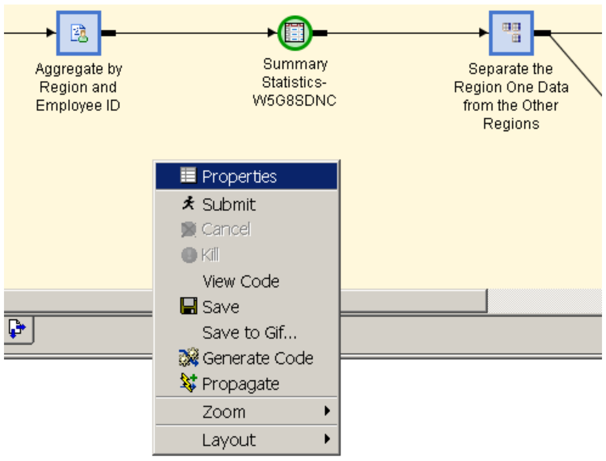
Now, by looking at the final Data Flow Diagram, you can get a pretty good idea of what the Job is doing.

GENERATING A DESIGN DOCUMENT

Now, we’re ready to create our design document from our Job.

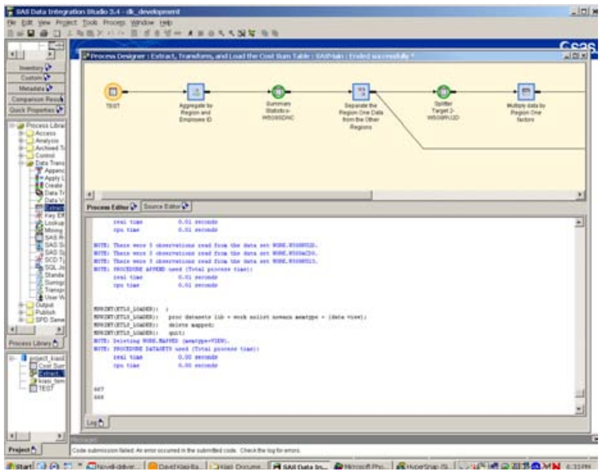
The amazing thing about DI Studio is that after moving a bunch of icons together into a diagram, our diagram is now ready to execute. DI Studio automatically generates SAS code that can be executed, even before we’ve developed the detailed features of the application.

a. So, we go back to the Job window, and right click. The following comes up.


Aggregate by Region and Employee ID
Summary Statistics-W5G8SDNC
Separate the Region One Data from the Other Regions
Properties
Submit
Cancel
Kill
View Code
Save
Save to Gif...
Generate Code
Propagate
Zoom
Layout

By clicking "Submit", our Job runs.

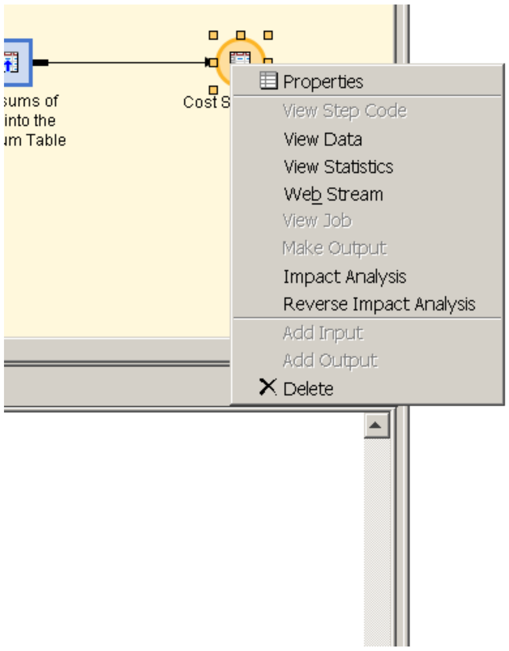
b. Below, we can see that the job ran successfully.


SAS Data Integration Studio 3.4 - dk_development
Process Designer : Extract, Transform, and Load the Cost Sum Table : SASMain : Ended successfully
TEST
Aggregate by Region and Employee ID
Summary Statistics-W5G8SDNC
Separate the Region One Data from the Other Regions
Splitter Target 2-W5G8R1ID
Multiply data by Region One factors
Process Editor
Source Editor
Log

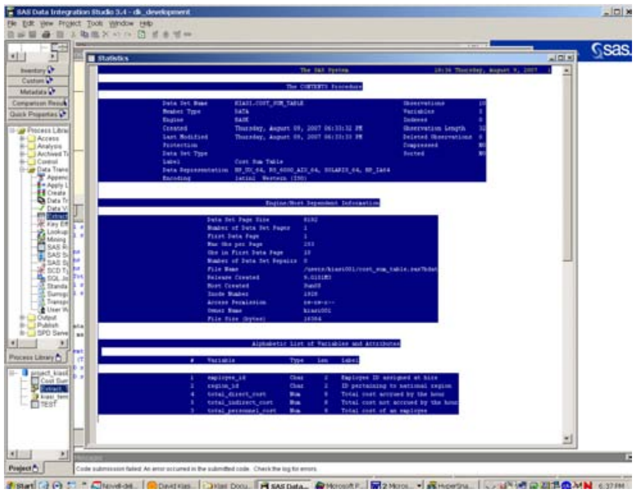
Why did we have to execute the Job? The only way to create the actual SAS dataset on your server is to execute a Job. Only until the target Metadata is actually loaded is the physical dataset created.

c. Once created, we can generate a PROC CONTENTS that can be copied directly into our design document's Data Dictionary section. We didn't have to hand create a table with Word.

We generate the PROC CONTENTS by using the "View Statistics" option from the pop up window that appears when we right click the target dataset in the Metadata tree or on the Job diagram.



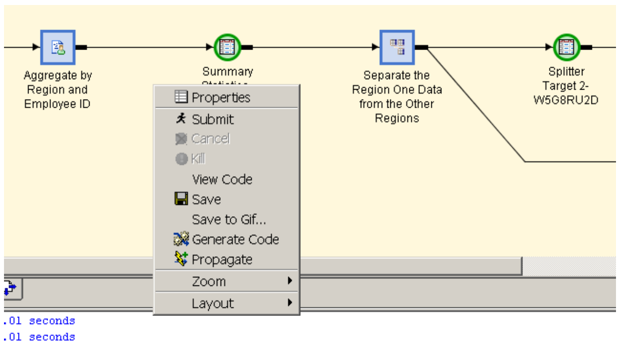
d. The PROC CONTENTS appears in a window:



```
                                    The SAS System          18:36 Thursday, August 9, 2007

                          The CONTENTS Procedure

Data Set Name        KIASI.COST_SUM_TABLE                              Observations
Member Type          DATA                                              Variables
Engine               BASE                                              Indexes
Created              Thursday, August 09, 2007 06:33:32 PM             Observation Length
Last Modified        Thursday, August 09, 2007 06:33:33 PM             Deleted Observations
Protection                                                             Compressed
Data Set Type                                                          Sorted
Label                Cost Sum Table
Data Representation  HP_UX_64, RS_6000_AIX_64, SOLARIS_64, HP_IA64
Encoding             latin1  Western (ISO)

                     Engine/Host Dependent Information

Data Set Page Size          8192
Number of Data Set Pages    1
First Data Page             1
Max Obs per Page            253
Obs in First Data Page      10
Number of Data Set Repairs  0
File Name                   /users/kiasi001/cost_sum_table.sas7bdat
Release Created             9.0101M3
Host Created                SunOS
Inode Number                1928
Access Permission           rw-rw-r--
Owner Name                  kiasi001
File Size (bytes)           16384

              Alphabetic List of Variables and Attributes

#    Variable               Type    Len    Label

1    employee_id            Char    2      Employee ID assigned at hire
2    region_id              Char    2      ID pertaining to national region
4    total_direct_cost      Num     8      Total cost accrued by the hour
5    total_indirect_cost    Num     8      Total cost not accrued by the hour
3    total_personnel_cost   Num     8      Total cost of an employee
```

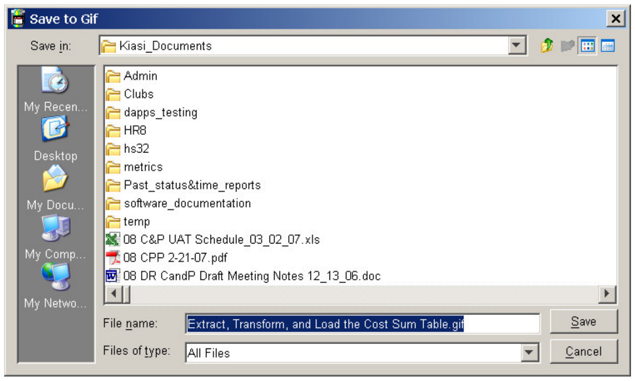
Code submission failed: An error occurred in the submitted code. Check the log for errors.

e. We can export our Data Flow Diagram by using the "Save to GIF" option. To bring this up, we need to have the Job diagram showing. On the diagram we right click anywhere and the following pop-up appears.



By clicking the "Save to Gif" option, we then bring up the next window.

f. This window allows us to save the GIF diagram.

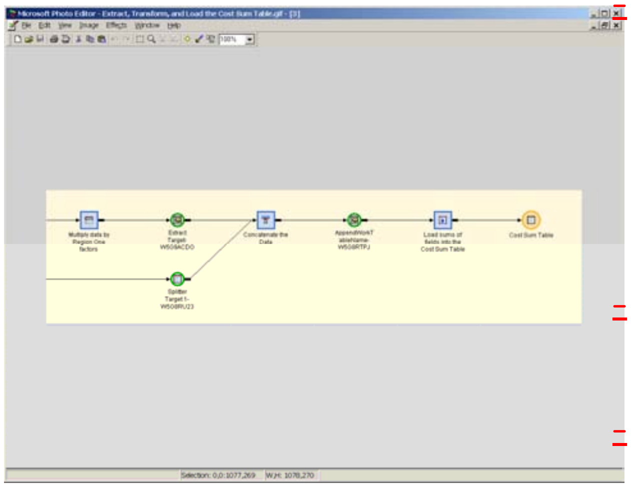


Save in: Kiasi_Documents

File name: Extract, Transform, and Load the Cost Sum Table.gif

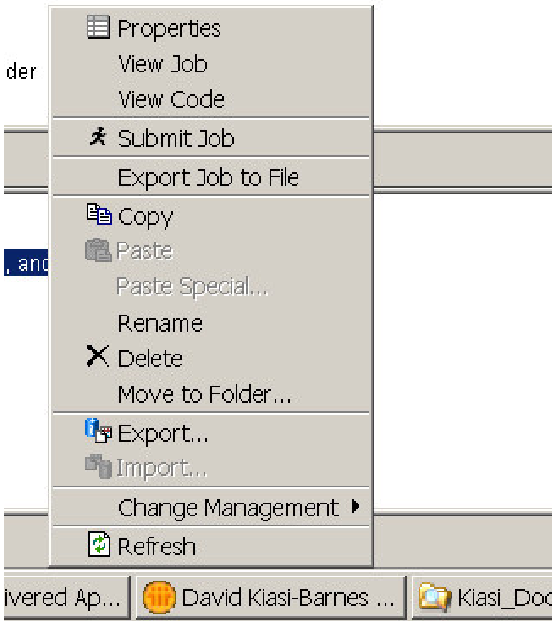
Files of type: All Files

g. Below is the GIF diagram.



h. Now, you can also export your descriptions. However, an XML file is generated when you do this. In order to use this in a document, you'll have to write a SAS program to parse the XML file.

To generate the XML file, you right click on the Job name in your Metadata tree. The following pop-up appears.



The option you want to choose is "Export Job to File."

i. Below is a part of the XML file that is generated by this feature.

```
 <?xml version="1.0" encoding="UTF-8" ?>
- <Export Type="Job" Version="9.1.3">
- <AddMetadata>
- <Metadata>
- <FeatureMap IsUserDefined="0" IsActive="1"
    Desc=""
    MetadataCreated="09Aug2007:22:07:42"
    Name="newmap"
    MetadataUpdated="09Aug2007:22:07:42"
    LockedBy="" TransformRole="ONETOONE"
    Id="$147">
  <SourceCode />
  <PropertySets />
  <SubstitutionVariables />
- <AssociatedClassifierMap>
  <Select ObjRef="$156" />
    </AssociatedClassifierMap>
  <Properties />
- <FeatureSources>
  <Column ObjRef="$64" />
    </FeatureSources>
  <TransformationSources />
- <FeatureTargets>
  <Column ObjRef="$101" />
    </FeatureTargets>
    </FeatureMap>
- <Column SASColumnType="C" BeginPosition="0"
     SASAttribute="" ColumnLength="2"
     SASFormat="" SASScale="0.0"
     SASColumnLength="2"
     ColumnName="region_id" SASInformat=""
     Desc="ID pertaining to national region"
     MetadataCreated="03Aug2007:22:05:48"
     IsDiscrete="0" IsNullable="1"
     SASPrecision="0.0"
     SASColumnName="region_id"
     SummaryRole="" Name="region_id"
     MetadataUpdated="03Aug2007:22:05:48"
     LockedBy="" ColumnType="0" SortOrder=""
     SASExtendedColumnType=""
     SASExtendedLength="0.0" EndPosition="0"
     Id="$158">
- <SourceFeatureMaps>
  <FeatureMap ObjRef="$171" />
    </SourceFeatureMaps>
  <PropertySets />
- <Table>
  <PhysicalTable ObjRef="$74" />
    </Table>
- <SpecSourceTransformations>
  <Transformation ObjRef="$94" />
    </SpecSourceTransformations>
```

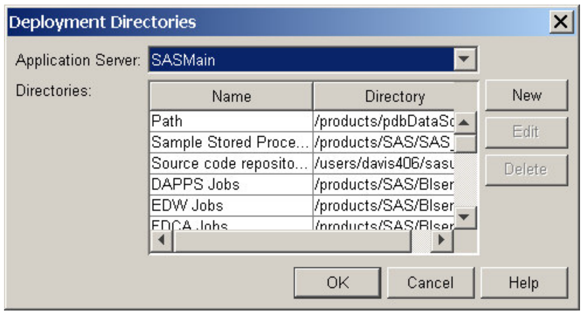
NEXT STEPS

You now have the components to create your design document.

Your peers can now review your design. Not only will they be able to provide valuable comments about your approach but they will also become familiar with the application you are building. If there was no time to create the document or there was no will to do so by management, at least your design is available for others to look at using DI Studio.

When several DI Studio Jobs have been created that depend on each other, before any development is done, one can use the DI Studio scheduler to act as a sequence structure diagram.

First, your deployment directory needs to be established. From the ????? you bring up the Deployment directories window:



Name	Directory
Path	/products/pdbDataSc
Sample Stored Proce...	/products/SAS/SAS_
Source code reposito...	/users/davis406/sasu
DAPPS Jobs	/products/SAS/BIser
EDW Jobs	/products/SAS/BIser
EDCA Jobs	/products/SAS/BIser

When you click new, the following window appears:

Here. you choose “Redeploy Jobs for Scheduling.”

DEVELOPMENT

The design phase is over. Now development begins.

The developer can use the design to develop the application. He/she will use the layouts of the datasets, the Data Flow Diagrams, and the descriptions within each process icon to finish the application. He/she will now:

1. Go into each process icon and create the mapping of the fields.

2. Add the necessary filters.

3. Put the code in the windows where fields are modified, combined, summed, within each row of data.

4. Add SQL statements for joins in the menus.

5. Refine the scheduler.

CONCLUSION

DI Studio can not only be used for Extract, Transform, and Load applications when building a data warehouse. It can also be used as a structured design tool for any SAS application. The advantage of this approach is that design of applications, which in many shops is ignored entirely, becomes painless. Design and Development are interconnected. What was created for the design can be directly applied to development. This ends up saving time and also allows for a less error prone and well documented application.

ACKNOWLEDGEMENTS

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.

I would like to acknowledge my client, the DBIA Team of the Decennial Management Division of the U.S. Census Bureau, who provided the SAS Business Intelligence software that this presentation was based on. Also my other clients, Centech, Inc. and Qualex, Inc.

REFERENCES

1 Webster's New World Dictionary of Computer Terms, Fourth Edition (1992).

AUTHOR CONTACT

David Kiasi
Applications Alternatives
P.O. Box 4238
Upper Marlboro, MD, 20775

(301) 350-4752

E-mail: david.kiasi@appalt.com

Web page: www.appalt.com