SHORT FINAL REPORT OF S.M.A.R.T.[1,2]

Martha A. Centeno, Ph.D.
Department of Industrial and Systems Engineering
Florida International University
Miami, Florida 33199
Centeno@eng.fiu.edu

[1] Shop floor Modeling, Analysis, and Reporting Tool. Developed under a NASA / FAR Grant NAG10-0150
[2] Prepared by Martha A. Centeno, Marelys L. Garcia, Alicia C. Mendoza, Louis A. Molina, Daisy Correa, Steve Wint, Gregorie Dolce, M. Florencia Reyes

1. INTRODUCTION

This document presents summarizes the design and prototype of the Shop floor Modeling, Analysis, and Reporting Tool (S.M.A.R.T.) A detailed description of it is found on the full documentation given to the NASA liaison. This documentation is also found on the A.R.I.S.E. Center web site, under a protected directory. Only authorized users can gain access to this site.

2. OVERVIEW OF THE SFC/DC DATABASE

SFC is one of four clusters that make up the Integrated Work Control System (IWCS). The latter will integrate the shuttle processing databases at Kennedy Space Center (KSC). There are approximately 20 distinct databases within IWCS forming the four clusters of information. These clusters are ARMS, WPSS, CASPR, and SFC. SFC is the primary source of information for this effort. The SFC module has been operational since 1992.

SFC revolves around a DB/2 database with PFORMS acting as the database management system (DBMS). Tables in the SFC database can be one of three kinds: *pre-defined*, *statistical*, and *support*. *Pre-defined* tables are those defined by the developers of PFORMS. KSC systems analysts make copies of these tables and rename the fields as needed to meet specific requirements. *Statistical* tables are those that have been defined by KSC systems analysts to support the analysis effort. *Support* tables are those defined by KSC systems analysts to help maintain the integrity of the data.

Tables in the SFC database are populated either from the IOS database or directly by the user. The main dynamic entity in the SFC and IOS database is a task (WAD); thus, the physical storage location and update privileges are driven by the status of the WAD. A WAD can be updated only through the software system governing it; for example, if a WAD is in an *"IW"* status, all admissible changes to it are done through the SFC interface. These changes are transferred to IOS via a batch process written in COBOL and REXX/SQL. This protocol is executed in batch mode, and it enables *almost real time* updates to both databases. It falls short of real time because the protocol is executed either once a day (from IOS to SFC) or twice a day (from SFC to IOS). IOS is not a DB/2 database. Consequently, when transferring data to (and from) SFC, it is necessary to *extract* the data to a format neutral to both databases (ASCII or binary) and then *load* the data into the database unique format. This conversion is done automatically by a series of COBOL and REXX/SQL procedures. The data exchange process can also be executed *"on-demand"*.

The majority of changes to a WAD are done by the technicians as they execute them. Consequently, the SFC interface is utilized more frequently, and the SFC database records are updated in true real time. The tasks found in IOS are transferred to SFC with an 11-day window. Once this is done, they are *available* for the technicians to work on (table ***orderopn***). Upon assignment to a task, the technician logs in and records pertinent information on the WAD s/he is about to work on (table ***actvempl***). As the work progresses, the WAD may experience delays, so the technician keeps interacting with the SFC until the task (WAD) is completed. Delays are entered by the technician using pre-established delay code taxonomy. This taxonomy has been given to the technicians in the form of a bar-coded list. Once a week, a SQL procedure is run to filter information out of the ***actvempl*** table and put it into the ***tsf01_delay*** and the ***tsf02_task_worked*** tables. These last two tables are used to load and run EXCEL based procedures needed for *counts* reports. After the EXCEL procedures are done, records are added to the ***tsf03_perf_status*** for future usage.

3. S.M.A.R.T. DESIGN

3.1 GENERAL PHILOSOPHY

The Shop floor Modeling, Analysis, and Reporting Tool (S.M.A.R.T.) is a framework designed around a database that would contain statistical information regarding work time, delay duration, and other historical summaries for inferential analysis. This framework has been devised under the premises that it

1) *has to be easy to use and flexible*, so as to allow growth of NASA's modeling activities,
2) *must be pseudo intelligent*, so as to provide guidance to the end user, and
3) *must be an open system*, so as to avoid re-inventing the wheel.

Figure 3-1 provides a representation of S.M.A.R.T. It is rooted on a relational database, upon which a series of

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

commercial and customized software tools act. The various commercial tools are integrated through a customized user interface written in Visual Basic. Some capabilities of S.M.A.R.T. are expected to be fully implemented (coded) using Visual Basic, while others will be implemented by combining a specific commercial tool (e.g. STATMOST) and Visual Basic.

There are five major elements in the framework:

a) the users S.M.A.R.T. intends to serve,

b) the databases it uses,

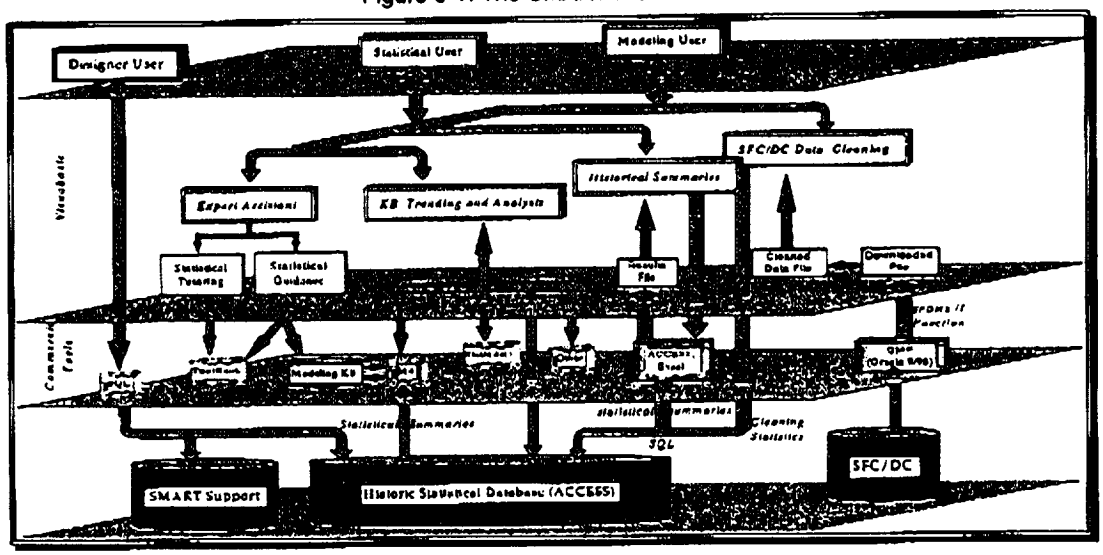
c) the processes through which SFC and S.M.A.R.T. interact,

d) the user interface, and

e) the tools used in developing S.M.A.R.T.

In the following paragraphs, these various elements are discussed from a design philosophy point of view. The technical descriptions of each one of them are given in other sections of this document.

Figure 3-1: The SMART Framework



3.2 The Users of S.M.A.R.T.

The various capabilities of S.M.A.R.T. have been devised to accommodate various data modeling needs of three types of users: 1) the designer user, 2) the statistical user, and 3) the modeling user.

1. *Designer User*: S/he interacts with S.M.A.R.T. through the various commercial tools integrated under it to:

   (a) *Modify the historical and support database of S.M.A.R.T.*: As new analysis requirements are defined, new relational tables may be needed. Although the relational model allows for easy expansion of the database, only the *designer user* should be allowed to add tables to it. Otherwise, chaos will show up. This user must be someone who understands the structure and relationships of the entire database, so that data duplication does not occur and critical tables are not deleted.

   (b) *Develop new "canned" reports*: After the three-year grant period, the need for a new type of reports will arise. The *designer user* will be called upon to design and develop the necessary forms and procedures

(using the **report** tool of ACCESS or Visual Basic) to generate these new reports.

(c) *Develop new data entry forms*: In the future, the support database of S.M.A.R.T. and the support knowledge bases will need to be updated as new knowledge and needs arise. For instance, at the moment, a support table has only a subset of the fields of SFC ACTVEMPL table. This means that only data pertaining to those fields may be effectively handled by S.M.A.R.T. Such a decision has been made based on the current analysis and modeling requirements. However, these requirements may change forcing the inclusion/exclusion of new or existing SFC fields.

(d) *Add new users of S.M.A.R.T.*: Because S.M.A.R.T. will maintain a historic database from which inferential patterns will be drawn, this user must keep a certain level of control on who interacts with the system. The level of access and privileges provided is up to the discretion of this main liaison user.

2. *Statistical User*: This user may be a "student" of S.M.A.R.T., or it may be a user who needs to generate a descriptive and/or inferential summary by using one of the capabilities already automated within the framework. S.M.A.R.T. requires the interaction with this user at strategic points in time (e.g. end of a flow) to populate its historical statistical database.

3. *Modeling User*: S/he will be an engineer who uses existing S.M.A.R.T. canned procedures in conjunction with customized ones to support a new modeling activity. An example would be the use of S.M.A.R.T. procedures to feed probabilistic inputs to a queueing model, SCRAM, or SIMAN models.

3.3 THE S.M.A.R.T. DATABASES

The framework is seating on two main relational databases: a support database and a historical database. Each of these databases contains tables to store data for diverse capabilities. For instance, there is a table called ***descriptive*** that is used to store descriptive statistics of a flow segment (flow cluster). Later, these summaries would be retrieved to conduct inferential analysis such as regressions, correlation, and hypotheses testing. Results from these analyses would also be stored in the historical database. Another example of the functionality of the databases is the table ***invcodes*** which stores data regarding delay codes that are no longer in use. As time passes, the delay codes structure changes. Some codes are added, while some codes are deleted. S.M.A.R.T. needs to "know" when a code is no longer in use, so that it can handle downloaded data properly.

3.4 SFC/DC - S.M.A.R.T. INTERACTION

S.M.A.R.T. takes its inputs from SFC/DC in an indirect fashion. Data from SFC/DC must be downloaded and transferred to a PC text file. The downloaded file contains a series of printer codes that must be removed before S.M.A.R.T. can process the actual data. In addition, it is necessary to check each individual record for completeness and usefulness. These two record operations are accomplished through a Visual Basic routine. The cleaned files are kept in ASCII format. A set of Visual Basic routines interact with both the cleaned data files and commercial tools to summarize and manipulate the data to find patterns and relevant statistical models of it. These activities are the ones that will populate the historical database of S.M.A.R.T.; therefore, it is important to download SFC data in a meaningful fashion. A later Section of this document describes the way in which SFC data would be clustered for effective and meaningful analysis.

3.5 TOOLS TO DEVELOP S.M.A.R.T.

We have decided not to re-invent the wheel; thus, we are using a variety of commercial tools. Whenever these commercial tools fall short for our needs, we use Visual Basic to complement it or to take over. These tools include Visual Basic, ToolBook, ACCESS, SQL, STATMOST, Statistica, Eclipse, M4, and ForecasterPro.

4. SFC/DC DATA CLUSTERING PHILOSOPHY

SFC/DC contains delay and work data collected at various locations at Kennedy Space Center (KSC) and Edwards Air Force Base. Thus, each record is labeled as either a work record or a delay record. In addition, each record contains fields to identify the physical location where the record was entered as well as the number of the flow. For the purposes of S.M.A.R.T., we need these data to be clustered in a meaningful way from the modeling and analysis point of view. Each cluster will be downloaded as a set, and it will be reduced to summaries, so as to

populate parts of the historic database. The hierarchy in Figure 4-1 is used to organize the way SQL/QMF queries are constructed (Table 4-1), and to name the various ASCII files that S.M.A.R.T. needs to generate as various record processes are done.

Three levels of clustering have been identified:

- *Flow Level*: (1st level) Most general grouping of SFC data is on a per flow basis. e.g. STS-62, STS-51. A flow is broken down into several non-serial *stages* (Figure 4-2).
- *Stage Level*: (2nd level) Intermediate grouping is done per stage, e.g. OP, VAB, PAD. SFC data will not be clustered per department (DEPCH) because our main goal is to enable mechanisms to improve the processes, not to point out the "guilty" party.
- *Record Type Level*: (3rd level) Most specific grouping is done on a per type of record basis, within each location for each flow. e.g. delay record, work record. There are other types of records in the SFC database; however, they are beyond the scope of this effort.

Level 3 in the hierarchy is the level at which the SFC/DC data must be extracted for usage in S.M.A.R.T. This enables analysis in various ways for each type of record (Work (acttrnid = '31') and Delay (acttrnid = '37')), including on a per flow basis, on a per flow stage (flow segment) basis, on a per code basis, and on a per wad/wad type basis

Figure 4-1: SFC/DC Data Clustering Hierarchy

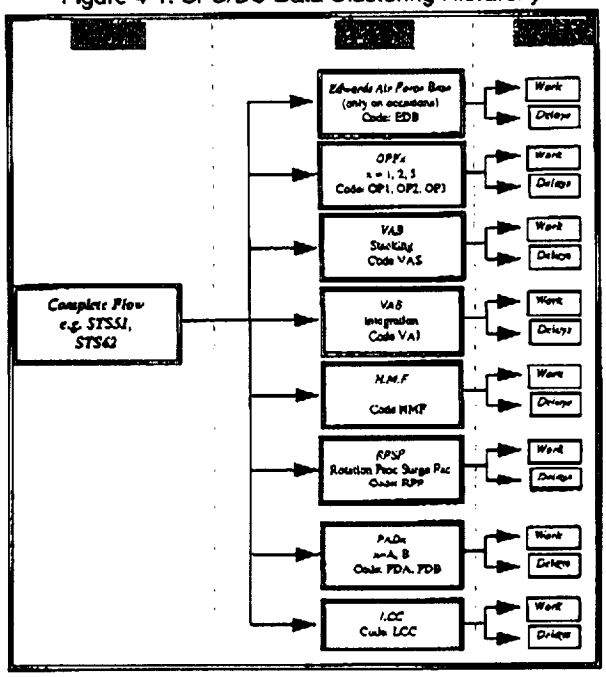


Table 4-1: Codes for the various clusters

Department Description	OPF 1	OPF 2	OPF 3	HMF	VAB Integration	VAB Stacking	Pad A	Pad B	RPSF	EAFB
Department Code	23-11	23-12	23-13	23-25	26-32	81-50				

We have developed a nomenclature that encapsulates the actual origin of the data. The nomenclature is limited in the number of characters by the restrictions imposed by the DOS operating system. In general, the nomenclature distinguishes each file as a work file (W) or as a delay file (D). It states the STS number to which the file belongs, it tells the cluster the file is coming from, and it tells which subset of the original data the file represents. From each data set, we can form subsets as needed. For example, it may be desirable to examine only the delays experienced

by OMI wads. In those cases, OMI records can be physically stored in a different ASCII file. Other subsets may be created afterwards by using the appropriate option in S.M.A.R.T. Figure 4-3 gives the nomenclature and Table 4-2 gives a list of the various types of wads.

Figure 4-2: Stages in a Flow

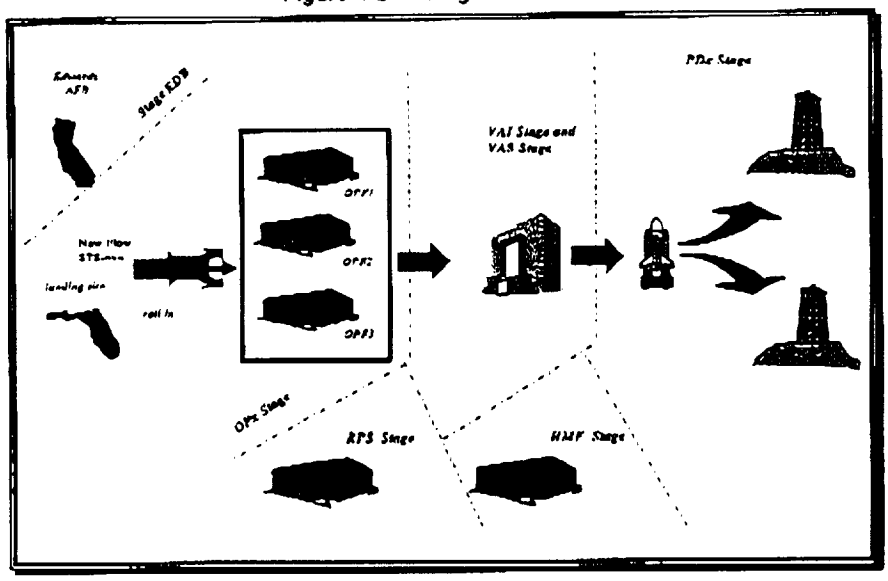


Figure 4-3: File Naming Nomenclature

Delay records:	*Dnnnxxij.ext*	Work records:	*Wnnnxxij.ext*	
where	*nnn* = { 030, ...,051,...,072,...,101,...999}		*xx* = { op, va, pd, hm, mx }	
i(xx) =	{ op: 1, 2, 3 va: I, S, pd: A, B, hm: F}		op1 = OPF1	op2 = OPF2
		vai = VAB - Integration	op3 = OPF3	hmf =HMF
		vas = VAB - Stacking	pda =Pad A	pdb=Pad B
j =	{ a, ..., z }			
ext =	{ cln, scr, tsh, tmp}	tmp = Temporary file	cln = Cleaned file	
		tsh = Rejected record	scr = SCRAM related file	

Table 4-2: Wad types

Wad Type	Wad Description	Wad Type	Wad Description
TPSA	Test Preparation Sheet: Permanent Change	IPR	Interim Problem Report
TPSB	Test Preparation Sheet: Temporary Change	PR	Problem Report
OMI	Operation Maintenance Instruction	DR	Discrepancy Report
ROMI	Repetitive Operation Maintenance Instruction	JC	Job Card
OM	Operation Manual	WA	Work Authorization

5. FILES LOCATION HIERARCHY

The file hierarchy of S.M.A.R.T. is given in Figure 5-1. The ***smart*** sub-directory is cinsered the root directory. It contains the executable and supporting files that make up S.M.A.R.T. All other sub-directories are hanging from ***smart***.

The ***cleandat*** sub-directory contains the files cleaned during the cleaning process. If the files in this sub-directory are deleted, they can be regenerated from the original downloaded files. The ***cleangral*** sub directory has been created to place cleaned files that are cleaned by S.M.A.R.T. but that will not be statistically processed by it.

The ***stats*** sub-directory contains files specially created for statistical analyses. a "cleaned" file may be the source of many "statistical" files. Files in this sub-directory will be customized to meet the format requirements of the statistical tool to use. The ***scram*** sub directory contains the files to feed **SCRAM**, and other SCRAM related files such as the files with discrepancies.

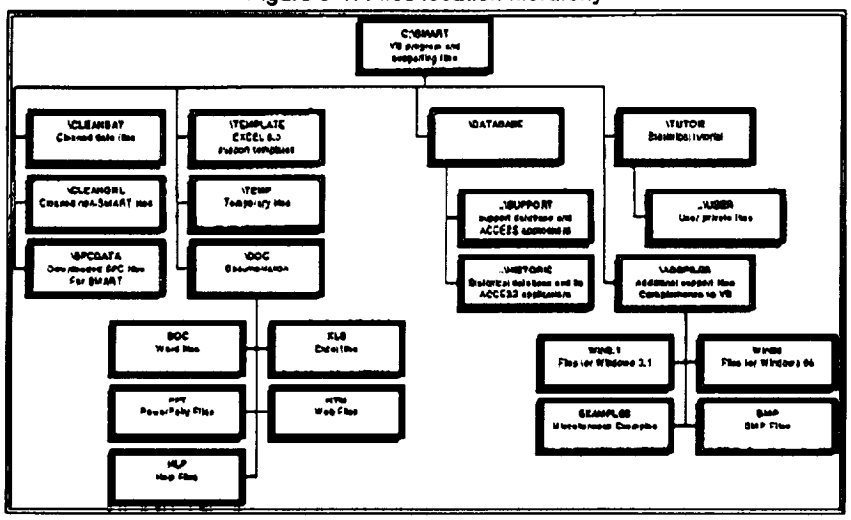
The ***sfcdata*** sub-directory contains the original files downloaded from the mainframe. This sub-directory may be empty if the end user chooses not to transfer the files from the floppy diskette to the hard disk. It is recommended that the files be transferred to the hard disk to speed up execution of the cleaning routine.

The ***templates*** sub-directory contains Excel 5.0 templates needed to support analysis and modeling activities. These templates will exists to avoid re-inventing the wheel in as much as possible. The ***tutor*** sub-directory contains supporting files for the ToolBook statistical tutor. Under it, there is a ***user*** directory to store tutoring-related user files, such as those files needed to track the progress of the student.

The ***database*** sub-directory contains supporting files for the ACCESS database. Under it, there are the ***historic*** and the ***support*** directories. The ***historic*** contains the historic database S.M.A.R.T. builds up as files are processed, whereas the ***support*** contains files that can be updated by the user, but are critical for S.M.A.R.T. If the files in ***support*** are deleted, S.M.A.R.T. will not work properly.

Sometimes, while doing an operation with files, S.M.A.R.T. needs to create intermediate files. These files are stored in the ***temp*** sub directory. All intermediate files are automatically deleted when no longer needed. However, if you see files in the sub-directory (when S.M.A.R.T. is NOT running) you may delete them. The ***doc*** sub directory contains all the documentation of S.M.A.R.T., including this document. Changes in hardware and software technology causes that sometimes the tools chosen for the development of applications like S.M.A.R.T. do not interact quite smoothly. Vendors of hardware and software, in most instances, provide the means to overcome such incompatibilities in the form of software that must accompany the application as extras. S.M.A.R.T. has several of these software files, and they are found in the sub directory ***addfiles***. Under this sub directory, there are two sub directories, one for Windows 3.1 and one for Windows 95. This is necessary because the software technology is in transition to Windows 95 (or Windows NT).

Figure 5-1: Files location hierarchy



6. DATABASES

6.1 SUPPORT DATABASE

S.M.A.R.T. requires the existence of some information at runtime. This information is stored in the support database (ACCESS format). This database is to be updated by the *design user* only. A list of the current tables in this database is given below, followed by a description of each table. The estimated size of the entire support database is 1 megabyte.

- CrossFields
- FieldsDef
- IndexTable
- InvCodes
- Messages
- Missions
- QueryFields
- QueryNames
- SFCTables
- Shops
- Users
- UserSession
- WadTypes

CrossFields contains fields found within the SFC database that may appear in a file used by SMART. This table basically renames each field in SFC, so that the new name can be used directly as labels for the statistical package. These new names are more intuitive.

FieldsDef enables the creation of ACCESS tables at run time. In some instances, S.M.A.R.T. may need to re-arrange the results of a SQL query. This is achieved by playing around with temporary tables. Such temporary tables need to be created during run time. ***IndexTable*** holds the name and number of the support tables needed by S.M.A.R.T. to perform various operations. An important piece of information is that this table also records a unique number for each one of the tables. ***Messages*** contains the text of the messages displayed on the status bar of the **activity** form. These messages are displayed as a specific request during an operation.

InvCodes holds the list of invalid codes. Because there may be several types of codes used within S.M.A.R.T., this table has a field that distinguishes the various classes of codes. At the moment, **type = 37** is the only one being used, and it is reserved for ***delay codes***. A delay code may become invalid as the list of delays codes is consolidated. The cleaning process of S.M.A.R.T. uses the date stored in this table to decide whether the invalid code is applicable to the tuples resulting from a SQL query. All records that have one of these codes are considered rejected records, and they are saved in a separate file (extension ".tsh").

Missions holds information pertinent to each flow. It includes information about stages, the STS Number, the orbiter, and the dates for each of the stages. Most of the information in this table has been collected from the NASA on-line information. This table will be used to conduct correlation and trend analyses among flights of the same kind; for instance, analyses among *spacelab* flights only or among *TDRS* flights only. In addition, orbiter and facility driven correlation can be implemented using the information in this table. Also, this table includes a **complexity** factor which will be determined by S.M.A.R.T. based on the number of wads processed. An analysis of the data regarding the number of wad as well as the variety of wads should be used to establish a base-line for what will constitute an "average level" of complexity

QueryNames contains the references to the names of the standard queries. The queries are associated to the entries in ***QueryFields*** through a 1:many relationship. ***QueryFields*** contains the fields on each of the standard query. It includes the name of the query, the type of the record and its position within a particular query. A standard query is a query that is used repetitively. All queries needed to populate the historical database are standard queries.

SFCTables contains the name of the SFC/DC tables of interest to S.M.A.R.T. This table also assigns an arbitrary number to each table. This table is visited by S.M.A.R.T. whenever the user wishes to use the "general records" capabilities of S.M.A.R.T. ***Shops*** contains the various codes for shops of interest. ***Users*** contains information pertaining to each user of SMART. It includes the user's identification number, last name, first name, type, password and the date of the user's last session. This table will be used along with the ***UserSession*** table to gather statistics on of the users of SMART. These two tables are associated through a 1:many relationship.

WadTypes holds the prefixes of some wad names and their corresponding wad type. This table enables S.M.A.R.T. to repair some of the records downloaded from SFC in case the records have no wad type. The prefixes have been set by cross referencing the wad names and the wad types, and identifying the minimum set of common characters that uniquely relates the prefix to a wad type. Table 6-4 gives a list of the rules that have been implemented in S.M.A.R.T. as it pertains to repairing the wad type. Other rules have been formulated, but they have not been implemented in S.M.A.R.T. (Table 6-3) because of knowledge not encapsulated in the *wadname* itself: *experiential knowledge from technicians.*

6.2 HISTORICAL DATABASE

The historical database is the centerpiece of S.M.A.R.T. In it, one finds summaries of the original SFC data. This database does not have the original raw data for two main reasons. First, the raw data does not need to be duplicated, it is stored in ASCII files for reusability, and the size of it is very large. Second, most of the continuous improvement models require statistical summaries rather than the original set of data. This database is also in ACCESS format. Updates to it are done automatically as the statistical user interacts with S.M.A.R.T. Appropriate

data integrity protocols have been embedded in the Visual Basic routines to preserve the consistency and reliability of the summaries. In the following paragraphs the actual contents of these tables is discussed in more details.

- CleanStats
- Desc_Delay
- Desc_Flow
- Desc_Flow2
- Desc_Wad
- Desc_Wad2
- Desc_Work
- Desc_Work_Wad

Table 6-3: Rules to repair wad types - not implemented

(1) If wadname begins with ECL and the last 5 characters in wadname have the format "-####" AND the numeric value of the last 4 characters in wadname is < 100, Then wadtype = DR

Example: ECL-#-##-0083 and ECL-#-##-0067 type DR
ECL-#-##A0083 not type DR because of "A" not "-"

reason: the rule above looks "pretty good" for data that is over a year and a half old. The last 4 digits of a DR used to return to "0000" at the beginning of each flow, the PR did not do this. Therefore, a Dr would have a smaller number than PR. But, nowadays, DR's and PR's number are pulled from a unique sequential list of numbers.

comments: need to run some more analysis on wad names and wad types to see if a different pattern can be found.

(2) A rule to distinguish between TPSA and TPSB can not be formulated.

reason: there is no way to distinguish between TPSA and TPSB. The FRCS TPS numbers that are formatted like FRC4-xx-xxx have no relationship to a vehicle. The "4" means that it is FRC #4, and the "xx" means it is that flight number for that particular FRCS, not for a particular orbiter. A deferred TPS will always have an "A" preceding the last 3 digits in the wad name.

comments: run some analysis to see if a definite rule can be formulated for a TPS wad type. Ask Amanda if there is any particular interest in keeping TPSA and TPSB as two distinct wad types.

Table 6-4: Rules to repair wad types - implemented

1. If *wadname* begins with an STS number, Then *wadtype = IPR*
Example: 062V.... ; 047V.....

2. If *wadname* begins with P, Then *wadtype = ROMI*
where:
P = {V1047, V1171, V1264, V1269, V1270, V3570, V5057, V5067, V5069, V9001, V9002, V9019, V9023, V9028, V9045 }

3. If *wadname* begins with P AND the *wadname* does not have a "/" in it, Then *wadtype = OMI*
where:
P = {V1047, V1171, V1264, V1269, V1270, V3570, V5057, V5067, V5069, V9001, V9002, V9019, V9023, V9028, V9045 }

4. If *wadname* begins with I, Then *wadtype = OMI*
where:
I = {V1007, V1008, V1009, V1010, V1011, V1015, V1017, V1019, V1022, V1026, V1032, V1037, V1041, V1042, V1048, V1052, V1053, V1058, V1062, V1065, V1076, V1078, V1082, V1086, V1091, V1093, V1097, V1098, V1105, V1113, V1134, V1153, V1158, V1164, V1165, V1175, V1176, V1177, V1179, V1180, V1181, V1196, V1200, V1215, V1223, V1224, V1226, V1230, V1238, V1240, V1262, V1263, V1274, V1276, V1278, V1283, V1288, V1291, V1304, V1321, V3575, V5005, V5006, V5008, V5011, V5012, V5018, V5032, V5046, V5050, V5058, V5074, V5079, V5083, V5098, V5101, V5114, V5124, V5129, V5138, V5139, V5144, V5147, V5148, V5151, V5158, V6005, V6012, V6018, V6029, V6049, V6044, V6054, V7253, V9048, V9046, V9021 }

5. If *wadname* begins with R AND the 4th character in the *wadname* is a "-", Then *wadtype = JC*
where:
R = {V00, V02, V11, V30, V31, V32, V33, V34, V35, V37, V41, V42, V45, V60, V63, V66, V75, V76, V80 }

Example: V35-400... type JC
V3570.... not type JC because of "7" not "-"

CleanStats is the table in which the cleaning process stores statistics regarding the number of useful records kept and the number of records rejected. Centeno and Mitskevich (1994) established that many records type '31' and '37' were done as trial records; hence, some of them were not logged off until months later or never logged off. In some other instances, the records were not updated appropriately when converted from a type '31' record to a type '37' record. Yet other records were entered accidentally, or were entered as "NOTES" or "REMARKS". Each file extracted using a historic standard query is cleaned up and appropriate statistics are computed and stored, to be able to track the data entry process. In the early stages of SFC, the data entry process was erratic; however, as SFC gained maturity, this process has reached reasonable steady state conditions. The information in this table will help the user to keep an eye on any unusual behavior in the data entry process. It will also help establish the source of the erratic behavior because it keeps information regarding the reasons why a record is rejected. This table begins its collection with STS-51. The decision to begin with STS-51 is based on a flow having no more than 10% of the original records rejected by the cleaning criteria.

Desc_Delay contains the summaries for delay records; specifically, it contains descriptive statistics per delay category, per each segment in a flow. The values in this table are computed by STATMOST and transferred to ACCESS via a Visual Basic procedure. The individual observations used in computing these summaries are the actual entries found in SFC. A *flow* (*j*) is processed at *n* different clusters. At each cluster, a WAD may experience any of the delays in the taxonomy. In fact, the same WAD may experience the same delay more than once. Every time a WAD experiences a delay, an entry is posted against SFC. This entry is treated as *an observation* when computing the values that go into ***descriptive***. The actual equations for each value are given below.

Let d_{ilkj} = *i*th delay duration for the *l*th delay code, in the *k*th cluster of the *j*th flow

m_{lkj} = total number of delay duration for the *l*th delay code, in the *k*th cluster of the *j*th flow

Then $d_{ilkj} = DelayEndTime\&\ DelayEndTime - DelayStartDate\&\ DelayStartTime$

$$m_{lkj} = count\{d_{ilkj} \quad \forall i\} \qquad mcount_{lkj} = m_{lkj} \qquad msum_{lkj} = \sum_{i=1}^{m_{lkj}} d_{ilkj} \qquad m\,var_{lkj} = (stdv_{lkj})^2$$

$$mavg_{lkj} = \frac{msum_{lkj}}{mcount_{lkj}} \qquad m\max_{lkj} = \max\{d_{ilkj} \quad \forall i\} \qquad m\min_{lkj} = \min\{d_{ilkj} \quad \forall i\}$$

$$stdv_{lkj} = \frac{\sum_{i=1}^{m_{lkj}} (d_{ilkj} - mavg_{lkj})^2}{mcount_{lkj} - 1} \qquad mcv_{lkj} = \frac{stdv_{lkj}}{mavg_{lkj}} \qquad kurtuosis_{lkj} = \sum_{i=1}^{m_{lkj}} d_{ilkj}^4$$

$$ckurt_{lkj} = \frac{kurtuosis_{lkj}}{\left(m\,var_{lkj}\right)^2} \qquad skewness_{lkj} = \sum_{i=1}^{m_{lkj}} d_{ilkj}^3 \qquad cskew_{lkj} = \frac{skewness_{lkj}}{\left(stdv_{lkj}\right)^3}$$

Desc_flow is the table in which one finds descriptive statistics per delay category for the entire flow; thus, the summaries in it are computed using all the files for each cluster in a flow. In other words, this operation requires the opening of all the cleaned files for each and every cluster of a given flow. The individual observations used in computing these summaries are the same as those used to populate the ***descriptive*** table. The resulting summaries confound (disregard) the differences that may exist among clusters; however, such difference needs to be detected and not guessed. The information in this table will be compared to that of table ***desc_flow2*** in an attempt to establish such difference. Per every flow/cluster combination, this table will have no records.

Let d_{ilkj} = *i*th delay duration for the *l*th delay code, in the *k*th cluster of the *j*th flow

$m_{:kj}$ = total number of delay duration for the *l*th delay code, in the *k*th cluster of the *j*th flow

n_j = total number of clusters in the jth flow

Then $d_{ilkj} = DelayEndTime\& DelayEndTime - DelayStartDate\& DelayStartTime$

$$m_{lkj} = count\{d_{ilkj} \quad \forall i\} \qquad mcount_{lj} = \sum_{k=1}^{n_j} m_{lkj} \qquad m\max_{lj} = \max\{d_{ilkj} \quad \forall i \quad \forall k\} \qquad msum_{lj} = \sum_{k=1}^{n_j}\sum_{i=1}^{m_{lkj}} d_{ilkj}$$

$$mavg_{lj} = \frac{msum_{lj}}{mcount_{lj}} \qquad m\min_{lj} = \min\{d_{ilkj} \quad \forall i \quad \forall k\} \qquad stdv_{lj} = \frac{\sum_{k=1}^{n_j}\sum_{i=1}^{m_{lkj}} (d_{ilkj} - mavg_{lj})^2}{mcount_{lj} - 1}$$

$$m\,var_{lj} = (stdv_{lj})^2 \qquad mcv_{lj} = \frac{stdv_{lj}}{mavg_{lj}} \qquad kurtuosis_{lj} = \sum_{k=1}^{n_j}\sum_{i=1}^{m_{lkj}} d_{ilkj}^4 \qquad ckurt_{lj} = \frac{kurtuosis_{lj}}{\left(m\,var_{lj}\right)^2}$$

$$skewness_{lj} = \sum_{k=1}^{n_j}\sum_{i=1}^{m_{lkj}} d_{ilkj}^3 \qquad cskew_{lj} = \frac{skewness_{lj}}{\left(stdv_{lj}\right)^3}$$

Desc_flow2 is the table in which one finds *clustered* descriptive statistics per delay category for the entire flow. The contents of this table differed from those in ***Desc_flow*** in the sense that these summaries use as individual observations the entries found in the ***desc_delay*** table, specifically the average found in each entry (a_{lkj}). Consequently, the average in this table is an average of averages, and the variance is measuring the dispersion of the average delay duration among the various clusters.

Let a_{lkj} = kth average delay duration for the lth delay code, in the jth flow

q_{lj} = total number of *average* delay duration for the lth delay code, in the jth flow ≤ Number of clusters

Then

$$a_{lkj} = mavg_{lkj} \qquad q_{lj} = count\{a_{lkj} \quad \forall k\} \qquad mcount2_{lj} = q_{lj} \qquad m\max 2_{lj} = \max\{a_{lkj} \quad \forall k\}$$

$$msum2_{lj} = \sum_{k=1}^{q_{lj}} a_{lkj} \qquad mavg2_{lj} = \frac{msum2_{lj}}{mcount2_{lj}} \qquad m\,var\,2_{lj} = (stdv2_{lj})^2 \qquad m\min 2_{lj} = \min\{d_{ilkj} \quad \forall k\}$$

$$stdv2_{lj} = \frac{\sum_{k=1}^{q_{lj}} (a_{lkj} - mavg2_{lj})^2}{mcount2_{lj} - 1} \qquad mcv2_{lj} = \frac{stdv2_{lj}}{mavg2_{lj}}$$

Desc_Wad is the table in which summaries of the descriptive statistics per delay category, per wad, per flow are stored for future analysis. There could be up to 183 delay categories per wad name. However, not all wads may experience all of the delays. ***Desc_Wad2*** is the table in which summaries of the descriptive statistics per **wad name**, per flow are stored. These summaries confound all delay categories per wad. ***Desc_Work*** is the table in which summaries of the descriptive statistics on work duration, per wad per flow are stored. ***Desc_Work_Wad*** is the table in which summaries of the descriptive statistics on work duration per **wad type**, per flow are stored.

7. THE RECORD CLEANING AND ASSESSMENT PROCESS

S.M.A.R.T. requires that records extracted from the SFC database be free of any nuisance values, and that they be complete as per the SQL/QMF query issued. Thus, each SFC record is inspected for completeness and for unusual values. Previous assessment of the SFC database contents revealed that in the early implementation stages, records were input as trial records or by mistake. Furthermore, the application program did not collect all the fields that are currently being collected. Consequently, many records have null values in one or more fields. To overcome these problems, S.M.A.R.T. applies the criteria given in Table 7-1 to each record, with the support of the contents of the ***InvCodes***. Completeness of the record is checked by breaking down each record into its various fields. The size of each field is read off from the downloaded file, which has a line of dashes that separates the field labels and the fields values. S.M.A.R.T. has the capability of counting these series of dashes and establishing the size for each individual field as downloaded from SFC. Statistics on the number of invalid records are kept in the historic database.

Criterion 1 and criterion 2 of the cleaning criteria for "WORK" records are only applicable when a grouping operation (**group by**) was done through the SQL/QMF query. Criterion 1 for "DELAY" records includes a null value for the delay code. The criteria for "GENERAL" records have been established using the same guidelines used for "DELAY" records in criterion 2. It is also important to point out that, as of 6/25/97, S.M.A.R.T. does not know when a code became invalid. As new codes become invalid, they shall be added through the ACCESS application developed for this purpose; thus, the date will be automatically capture and fed to S.M.A.R.T.

Table 7-1: Cleaning criteria for each type of record

Criteria for tasks worked records (type '31')		
1.	Time elapsed between technicians clocking out of the task is more than 7 hours: (max(sdate+stime) - min(sdate+stime)) > 7 hours	
2.	Time elapsed between technicians clocking in for the task is more than 7 hours: (max(actcdate+actctime)-min(actcdate+actctime))>7 hours	
3.	Worked time is less than 10 minutes or negative: (max(sdate+stime) - min(actcdate+actctime)) <0.167 hours	
4.	Worked time is more than 60 days: (max(sdate+stime) - min(actcdate+actctime)) > 1440 hours	
5.	Record is a trial record. Trial records have a number as the first character and a "-" as the 2nd character of partn: Example: 2-111692-5	
6.	The clock out date is 2 or more days after the roll over/roll out/ lift off date: max(sdate+stime)>2 + (roll over/roll out/lift off) date	
Criteria for delay records (type '37')		
1.	Delay code is invalid:	Examples: null, CD, blank space, SQ, C24, ACT
2.	Delay time is less than 5 minutes:	(sdate+stime) - (actcdate+actctime)) < 0.083 hours
3.	Delay time is more than 60 days:	(sdate+stime) - (actcdate+actctime)) > 1440 hours
4.	The clock in date is after the roll over/ roll out/ lift off date: (actcdate+actctime)) > roll over/roll out/lift off date	
5.	The record is a trial record:	See explanation given for type '31' records
Criteria for general records (type '99')		
1.	First time interval is less than 5 minutes	
2.	Second time interval is less than 5 minutes	

The current outputs of the record cleaning and assessment process are four files in a comma delimited, text format. These files are stored in the **cleandat** sub directory, from where they are retrieved as needed. The first file (suffix A.cln) contains all the records that met the cleaning criteria. The second file (suffix B.cln) contains the same records as the first file (suffix A.cln), except that some records have been modified. Those records that have WADTYPE = 'ROMI' and the WAD name has a suffix of the form "/A-ddmmyy-N (something that looks like a date) have been modified. The modification consists of replacing such suffix with "/R". This has been done to avoid the problem encountered of one observation for a given wad just because it was processed at different times for different flows. The third file (suffix C.cln) contains records that met the cleaning criteria and have WADS with a "V" as their

initial character. The latter file is the only subset file generated during the record cleaning and assessment process. Other subsets may be generated by using of the options in S.M.A.R.T. The fourth file (extension .TSH) contains the original SFC records that did not meet the cleaning criteria, in one or more criterion. The first three files are comma delimited, whereas the last one is written with two spaces between each field. Records in all of these files may have one or more fields repaired.

The layout for the first three files (suffixes A, B and C) is the same if the type of record is either "WORK" or "DELAY". The generic layout for these files is given in Figure 7-1. Notice that S.M.A.R.T. adds one or two columns, namely the time interval. For delay records, the interval is computed using equation (1), whereas for work records the interval is computed using equations (2) and/or (3). The units of the time interval are days. S.M.A.R.T. has the capability of identifying whether the necessary fields to compute the time intervals are part of the downloaded data. "WORK" records are liable to have up to two time intervals because grouping operations may be done on them. For "DELAY" records, it does not make sense to perform any grouping operation; thus, this type of records will have only on time interval. "GENERAL" records are treated in a similar fashion as to "WORK" records.

Figure 7-1: Layout of cleaned work or delay records

Line 1:	"RecType", NumFields, AddFields, STSNumber, "Location", #UBDate#, "Extraction Criterion"
Line 2:	LabelField1, LabelField2, LabelField3, . . . ,LabelFieldn, Interval1[,Interval2]
Line 3:	ValueField1, ValueField2, ValueField3, . . . ,ValueFieldn, ValueInterval1[,ValueInterval2]
:	
Line m:	ValueField1, ValueField2, ValueField3 . . . . ValueFieldn, ValueInterval[,ValueInterval2]

$$Interval = DelayEndDate + DelayEndTime - (DelayStartDate + DelayStartTime) \quad (1)$$

$$Max(Interval) = MaxEndDate + MaxEndTime - (MinStartDate + MinStartTime) \quad (2)$$

$$Min(Interval) = MinEndDate + MinEndTime - (MaxStartDate + MaxStartTime) \quad (3)$$

The meaning of the identifiers of the first line (Figure 7-1) is as follows:

- ☑ ***RecType*** is a string that can be either "DELAY" or "WORK", and it identifies whether the data set in the file are delay or work records.
- ☑ ***NumFields*** is an integer value that tells the number of fields included in the SQL query posted against SFC (as given by the user during the cleaning process).
- ☑ ***AddFields*** is an integer value that tells the number of fields added by S.M.A.R.T. These fields will be either 1 or 2, namely the time intervals computed for the records.
- ☑ ***STSNumber*** is an integer value that represents the number of the flight (e.g. 51, 76, 78).
- ☑ ***Location*** is a three character string representing the cluster from which this data set originated. See Section SFC/DC Data Clustering Philosophy for an explanation of these codes.
- ☑ ***#UBDate#*** is a date value that represents the ending date of operations for the stage or cluster; the format of this datum is of a Variant Type 7 (Date) in Visual Basic. The actual date value must be enclosed in "#"
- ☑ ***Extraction Criterion*** is a multi valued string (all values separated by a blank space), which indicates the size of the file with respect to the original file (all records or only a subset). It also indicates whether any modifications where done on the records. For example, files containing only those records where Wad name has a "V" as its first character, and the records where modified on the WAD field based on the wad type being ROMI, would have the value "MODI WAD WADTYPE ROMI SUBSET V". Figure 7-2 provides the generic hierarchy to establish the value for this item, whereas Table 7-2 gives possible values for this item.

The hierarchy in Figure 7-3 is a 6-level hierarchy. At each level, a string or null is added to the final value of **extraction criterion**.

- ➲ **Level 1** is the *type of record*: valid or rejected. This level adds nothing to the value of **extraction criterion**.
- ➲ **Level 2** is the *record modification*: was any records modified on any of its fields? (TRASHED, MODI, or

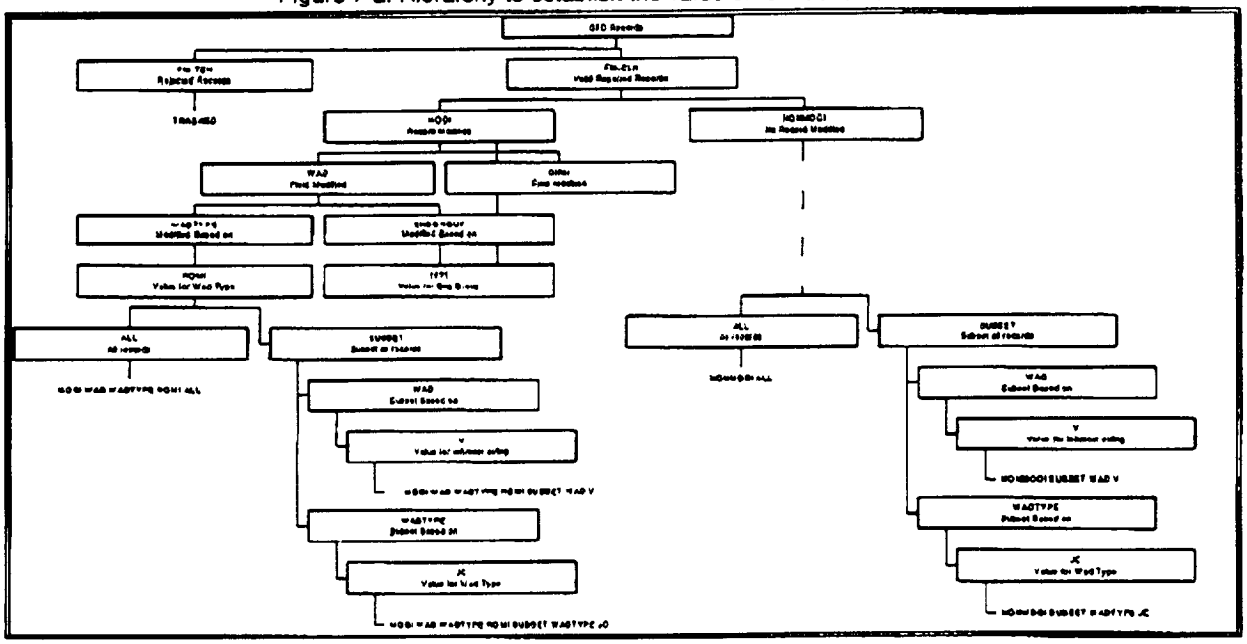
NONMODI). The value of **extraction criterion** begins to form at this level.

- **Level 3** is *field modified*: what field was modified, if any. For non modified records, this level adds a null value (not a blank space).
- **Level 4** is the *field modification was based on*. This level adds two strings: the field name and the field value. For non modified records, this level adds a null value (not a blank space).
- **Level 5** is the *set size* (ALL or SUBSET).
- **Level 6** is the *field record selection was based on*. This level adds two strings: the field name and the field value or field sub-string. For non modified records, this level adds a null value (not a blank space).

Table 7-2: Possible values for the item "extraction criteria"

Value	Meaning
NONMODI ALL:	File has all records that met the cleaning criteria. Some fields may have been repaired, e.g. stsnumber or in its wadtype, but none of the fields in the record were modified. (Suffix A file)
MODI WAD WADTYPE ROMI SUBSET WAD V:	File has only those records which had "V" as the first character in the wad name and met the cleaning criteria. Some fields may have been repaired. Some records may have been modified on the WAD field if the wad type was ROMI. (Suffix C)
MODI WAD WADTYPE ROMI SUBSET WAD A:	File has only those records which had "A" as the first character in the wad name and met the cleaning criteria. Some fields may have been repaired. Some records may have been modified on the WAD field if the wad type was ROMI.
MODI WAD WADTYPE ROMI SUBSET WADTYPE JC:	File has only those records which had "JC" as the value for wad type and met the cleaning criteria. Some fields may have been repaired. Some records may have been modified on the WAD field if the wad type was ROMI.
MODI WAD WADTYPE ROMI ALL:	File has all records that met the cleaning criteria. Some fields may have been repaired. Some records may have been modified on the WAD field if the wad type was ROMI. (Suffix B file)
TRASHED:	File has those records that did not meet the cleaning criteria. Some fields may have been repaired. (Extension TSH file)

Figure 7-2: Hierarchy to establish the value of "extraction criteria"



The second line in these files contains the labels of the fields included in each record. These labels are written as per our cross-referencing of SFC fields. The field labels (names) are not enclosed in quotes, but instead are only separated by commas, as is the rest of the information in the file. Lines 3 to *m* contain the actual values of each field. Notice that there are up to two additional fields, namely the time intervals.

In addition to assessing files for S.M.A.R.T., the framework has the capability of removing the printer codes for any file extracted from SFC. These other files are referred to as **general** records files. The records in the resulting file are not examined for completeness; however, the fields of these records are examined to determine if a time interval can be computed. If a data set is supposed to have the necessary fields to compute at least one time interval, all records must have the values to compute it; otherwise, the record is rejected and placed in the trashed records files. Another reason to place a general record in the trashed file is that the resulting time interval comes out to be negative. In summary, for "GENERAL" records, only two files are created: cleaned records and rejected records (See Figure 7-3). Notice that the difference between this layout and that of the WORK/DELAY records layout is the absence of the origin information.

Figure 7-3: Layout of cleaned general records

Line 1:	LabelField1, LabelField2, LabelField3, . . . ,LabelFieldn, Interval1[,Interval2]
Line 2:	ValueField1, ValueField2, ValueField3, . . . ,ValueFieldn, ValueInterval1[,ValueInterval2]
:	
Line m:	ValueField1, ValueField2, ValueField3 . . . , ValueFieldn, ValueInterval

The layout of a rejected records file is similar to those of a cleaned files, except that the second line of a cleaned file becomes the third line in a rejected records file (Figure 7-4). Why? because the second line in this type of file now has the size of each field as determined during the cleaning process. It is very important to point out, that the first field in a rejected record is always the "reason" why a record was rejected. This field has been given a size of 4 by the designer. In the case of **general** records, the first line of the rejected records file (Figure 7-5) has a modified version of the origin information. Specifically, we have decided to keep some information "learned" while cleaning the file (e.g. number of fields), so that the **modeling** user would be able to use other S.M.A.R.T. capabilities to review these rejected records.

Figure 7-4: Layout of file for rejected work or delay records

Line 1:	"RecType", NumFields, AddFields, STSNumber,"Location", #UBDate#, "Extraction Criterion"
Line 2:	4, size1, size2, size3, ... *sizen*
Line 3:	"reason", LabelField1, LabelField2, LabelField3, . . . ,LabelFieldn
Line 4:	reason ValueField1 ValueField2 ValueField3 . . . ValueFieldn
:	
Line m:	reason ValueField1 ValueField2 ValueField3 . . ..ValueFieldn

Figure 7-5: Layout of file for rejected general records

Line 1:	"GENERAL", NumFields, , " ", # #, "Trashed"
Line 2:	4, size1, size2, size3, ... sizen
Line 3:	"reason", LabelField1, LabelField2, LabelField3, . . . ,LabelFieldn
Line 4:	reason ValueField1 ValueField2 ValueField3 . . . ValueFieldn
:	
Line m:	ValueField1, ValueField2, ValueField3 . . . , ValueFieldn

All files, except the one of rejected records, are sorted in ascending order by wad name. This has been done to facilitate some of the analysis functions which require sorted data, so instead of having to sort at each analysis point, we sort only once: here!☺. In summary, the record cleaning and assessment process for files to be used by S.M.A.R.T. consists of:

1. **Clean File**: Remove printer codes from file. The SFC file has about 19 lines of printer codes and each record also has about the first 18 columns being printer code.

2. **Assess usefulness of record**:
   (a) Check the fields included in the SQL/QMF query to ensure that at least one time interval can be computed.
   (b) Repair fields as needed.
   (c) Apply cleaning criteria to each record.
3. **Generate files**:
   (a) Generate suffix A file and rejected records file.
   (b) Sort suffix A file.
   (c) Modify records as needed.
   (d) Generate suffix B and suffix C files.
4. Update cleaning statistics in **historic.mdb** as appropriate.
5. Report results of cleaning process to user.

whereas for a general file, it consists of examining records to establish if a time interval can be computed, apply cleaning criteria to each record, repair fields as needed, and generate files.

1. **Clean File**: Remove printer codes from file. The SFC file has about 19 lines of printer codes and each record also has about the first 18 columns being printer code.
2. **Assess usefulness of record**:
   (a) Check the fields included in the SQL query to check if at least one time interval can be computed.
   (b) Repair fields as needed.
   (c) Apply cleaning criteria to each record.
3. **Generate files**:
   (a) Generate *filename* file and rejected records file.
   (b) Sort *filename* file by wad name if wad name was part of the fields included in the SQL query.
4. Report results of cleaning and assessment process to user.

This procedure must be done for each cluster once a flow is completed. Once the files are cleaned and assessed, they need to be processed through various of the statistical options of S.M.A.R.T. to populate the historic database. Details on how to proceed in this regard are given in the section *Populating the Historic Database*.

8. A WEB SITE FOR SMART

We have designed a web site for this effort. Contents of this site are on secure site in A.R.I.S.E. Center's web site. The appearance of the web site is so dynamic that attempting to document it here is almost a "mission impossible." The URL of the web site is http://arise.eng.fiu.edu/smartweb/smart.htm

9. STATISTICAL TUTOR

One of the every day challenges that engineers face is to handle raw data for decision making. These data must be converted and reduced to meaningful indicators by using statistical and graphical tools. Unfortunately, engineers are not traditionally trained in probability and statistics, except for those who studied Industrial Engineering. This situation is no different at K.S.C.; thus, as part of S.M.A.R.T. we have designed and developed a ToolBook module that will help engineers to gain competence in probability and statistical analysis.

The main objective of this tutorial is to provide a comprehensive review of statistical concepts in several fashions:

1. *Conceptual*: The tutorials will enable the navigation from one concept to the other in the same sense as a human tutor would guide a student in the classroom.

2. *Problem Solving*: The tutorial will enable the end user to identify a specific set of techniques that could be used to solve a specific problem.
3. *Testing*: The tutorial will enable the end user to test his or her knowledge by solving a variety of exercises.

Consequently, the tutorial works in several modes: a) *self instruction*, b) *interactive*, and c) *self testing*. Under the ***Self instructional*** mode, the user navigates through different concepts and is allowed to select topics of interests for his/her review. The review is done by reading the concepts associated with the topic, or by observing an example of an application (real life problem) and how the concept is applied through solving a sample problem. Under the ***Interactive*** mode, the user interacts with the tutorial by providing information about the problem at hand and the type of data available. The tutorial reacts to the inputs of the user and request appropriate additional information. Under the ***Self Test*** mode the user has the choice to test how much s/he knows about a series of concepts. Usually the user would have first gone through the self instructional steps, either conceptually or through a sample, and then s/he would use this mode to check out how useful the session was by taking a test that covers the information just reviewed. The tutorial selects at random different samples from a database and uses them for the user to "examine" his/her knowledge. The tutorial provides feedback to the user as s/he progresses in the test by correcting mistakes made by the user, and providing an explanation for the right answer. At every stage of the testing routine, the user has the choice to return to the portion of the tutorial that covers the topic being tested on and be able to brush up on some concepts. To facilitate the documentation effort, we have baptized the tutorial with the name **C.P.T.** (Concepts, Problems, Tests).

C.P.T. offers several benefits over a regular textbook, besides those offeedr by the media employed (favoring quick access to information and quick review), include the advantages associated with the self test feature. Also, the medium in which the tutorial is being developed facilitates for examples to be renewed over time without changing the program code. Examples can be simply added or deleted from an examples database. Another benefit is the interactive facility of the program, which allows the user to receive feedback on performance as he/she is performing. This can potentially increase user retention; thus, providing a proactive approach to learning. The contents of the tutorial are documented in the full report.

10. THE S.M.A.R.T. PROTOTYPE

Once S.M.A.R.T. is installed, there shall be a group called SMART in your computer desktop, and inside it there will be an icon that resembles the space shuttle. Double click on the latter, and S.M.A.R.T. will ask you to enter your assigned user name and password. If S.M.A.R.T. is able to establish that you are authorized to use it, the it will be up and running.

If S.M.A.R.T. determines that you are not an authorized user, it will display an appropriate message, and it will go back to ask you for the user name and the password again. There is no limit to the number of times a user can entered an unauthorized user name / password.

The prototype is documented in the full report. For the purposes of this report, several capabilities have been implemented in the prototype. These capabilities include:

- assessing a work records file,
- assessing a delay records file,
- assessing a general records file.
- opening a single cluster for analysis,
- closing opened clusters,
- reviewing the origin information of a clean file,
- creating a new subset of a clean file,
- merging several clean files, and
- updating the support database.

11. OVERALL PROJECT STATUS

This project was officially extended by one year at no cost. Several factors contributed to delay the progress of the report including changing the NASA liaison several times, students leaving the program of studies, and health related problems. However, significant progress was made. To this date, although the project has been officially closed, we are still making improvements to the prototype. Two master's theses are being completed using the results of this effort. Several articles have been written, published and presented. Depending on the results of the two theses being completed by the end of summer 1999, other venues of inquiry shall be explored.

The full documentation of the effort is three binders. As enhancements are made to the prototype, the documentation is updated. The latest version is always maintained on a restricted web site.