Balancing the requirements of high quality data and timely completion of a clinical study is thankless task. On one hand the data must accurately reflect the trial conducted and on the other hand are the business needs that must be considered. Given enough time, all clinical trials would be able to have very high quality data with few if any errors. However, this is not a luxury that many will ever experience. Rather there are pressures to get the trial completed, with attention towards quality, but with the define pressure of deadlines that are often set outside the clinical realm. To ease the burden of cleaning and correcting the data in clinical trials of long duration and those that use the same personnel, a different style of handling data was put forth in order to redress the problem of recurring systemic problems.

What is outlined below is not a panacea to cure all the data ills of clinical trials, but rather one approach towards the problem of achieving, maintaining, and enhancing the quality of data at all phases of the clinical trial. The requirements of this approach are not insignificant, and include the willingness to critically analyze all segments of the trial, disseminate, and act on that information.

The term ‘data discrepancy’ is used here rather than ‘data error’ for several reasons. Its definition in this context is quite simple yet far reaching: Any datum that is changed from initial entry on the CRF to the final reports is held to be an data discrepancy, and thus covers a broader range of events than a data error. This definition demands accountability at each step of the trial or trials, enabling the use of various statistical methodologies to determine the existence of variances between individuals, sites, processes, CRF pages, and so forth. In short, it allows the process of clinical trials to be monitored in the same fashion that events in a clinical trial are monitored. For example, if instructions to the clinicians are vague and thus lead to multiple queries in order to resolve a particular datum correctly, it is better to be able to detect that data discrepancy early in the trial, and also be able to restate the instructions so as to remove any possible ambiguity for other clinicians than to wait until all the data in house and then start querying all of the clinicians on events that may only then be several months or years old.

The basic hypothesis of this paper is that by monitoring the performance and production of the processes at all stages in a trial, the points at which data discrepancies are commonly introduced can be identified and then rectified if deemed advisable. Monitoring here is not meant as x number of items per hour but rather the overall rate of clean data coming through the various stages of the trial. An individual (or process) who has a greater throughput of data per given amount of time than others may not be as effective as the others if the time required to correct the data discrepancies outweighs the increase in throughput. This leads to the concept of effective datum time, or the time required for the average datum of a particular type requires from the conception of the trial to the completion of the trial and all documents that use the datum. This includes not only the documents for and NDA, but also the follow-up ISS reports, interim adverse events reports, among others. This can be represented by:

\[
\text{Effective Datum Time} = \text{Process Time} + (\text{Data discrepancy rate} \times \text{Rectification Time})
\]

Viewed in this manner, there are two ways to reduce the amount of time a trial consumes, either reduce the data discrepancy rate or reduce the rectification time. As many of the issues surrounding rectifying an data discrepancy may be out of the control of trial management it is often more practical to reduce the data discrepancy rate.

In light of the concept of the effective datum time, the factors controlling the delivery of quality data in the basic processes of the clinical trial were investigated. The analysis of both SAS and Clintrial led to the conclusion that while both products allow for considerable control over the quality of the data, what was required to decrease the number of data discrepancies was needed was data on the characteristics of the data discrepancies in question. In this analysis, the strength of SAS is its analytical capabilities and that of Clintrial is its auditing/logging features. Using the two together affords the analysis of changes in the data. The focus then shifted to maximizing data quality through monitoring the causes of data discrepancies so as to be better able to determine and correct them. By enhancing the integration of the SAS and Clintrial using the table layouts of Clintrial, relevant internal variables in Clintrial, the auditing abilities of Clintrial and the analytical abilities of SAS, it was determined that the majority of data discrepancies could be tracked, although there would be significant overheads in computer resources and data collected incurred in doing so.

Using these features of SAS and Clintrial, a systematic report structure was designed that would help illuminate...
the sources and frequencies of discrepancies in data handling from CRF design to data collection through data entry and finally through analysis. The benefits of such a report system tracking data discrepancy genesis by personnel, site, CRF, CRF item, as well as by date and time, enhancing the management of the clinical trial process, both in post mortem and in vivo analyses. Deficiencies in the clinical trial process had previously been being identified but had not been corrected as there was insufficient information available to permit resolution. In order to address this lack of information, the development of a mechanism with capabilities similar to those used to track and analyze the adverse events of a double blind double crossover study was initiated. This capabilities of this mechanism would include:

- assisting in the tracking of events such as changes in the content of tables and listings as a trial progressed, so that identifying why a particular change (frequency of a given AE) occurred could be done with ease and certainty.
- enabling the definitive identification of the root cause of a change in the output, whether it be additional data that had been added, editing of the data used, changes in the program being run, and so forth, and to do so without a great deal of effort.
- monitoring the efficacy of the people involved in a trial, so as to be able to reduce the number and severity of data discrepancies generated.
- the system had to show not just a decrease in the clinical trial misery index, but a more tangible result, a decrease in cost and/or time.

Given this mechanism as our Holy Grail, the events that had made previous projects tortuous were logged, whether or not the trial had used Clintrial (all had used SAS). A small host of these were identified, and included data entry data discrepancies, illegible scrawl on CRFs, incomplete data, mismatched variable types between Clintrial and SAS, and more. The results of these data discrepancies is well known: “Go back, do not pass go, do not go home at five o’clock.” Having found a number of data discrepancy types, a rough calculation of the cost of data discrepancy rectification was performed as follows:

1) Identify the amount of time per item that an item takes to be generated, logged, entered, and checked. Ex: Visit 4 has 134 items, visit time is 1 hour, logging time 5 minutes, entering time 15 minutes, checking time 20 minutes.

2) Calculate the amount of time to rectify item. Writing exception report 5 minutes, preparation, typing and sending of letter to site 15 minutes, time spent explaining letter contents to site personnel, time spent asking site personnel for nth time, receiving letter from site, and so forth.

3) Add time delay for transmission of information, vacations, and so forth.

It is fairly easy to see that rectifying a data discrepancy can take significantly longer than the original entry in the first place. As the length of time to rectify a data discrepancy and the discrepancy rate increase, the effective datum time becomes far larger than the entry time.

<table>
<thead>
<tr>
<th>Rectification Time (Units)</th>
<th>1</th>
<th>4</th>
<th>16</th>
<th>64</th>
<th>256</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discrepancy Rate (%)</td>
<td>.1</td>
<td>1.001</td>
<td>1.004</td>
<td>1.016</td>
<td>1.064</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1.010</td>
<td>1.040</td>
<td>1.160</td>
<td>1.640</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>1.050</td>
<td>1.200</td>
<td>1.800</td>
<td>4.200</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1.100</td>
<td>1.400</td>
<td>2.600</td>
<td>7.400</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>1.200</td>
<td>1.800</td>
<td>4.200</td>
<td>13.80</td>
</tr>
</tbody>
</table>

From this simple table it is obvious that controlling either the rectification time or the discrepancy rate will limit the time damage. The combined effect of both the discrepancy rate and the rectification time being large is tremendous, especially if the time lag in discovery is large. The data discrepancy system attempts to minimize all these effects by analysis incoming data and addressing the problems immediately. Similar calculations can be done for costing analysis.

The mechanism that allows the implementation of data discrepancy control is comprised of six Clintrial variables, SAS/ACCESS for ORACLE, and the audit tables in the Clintrial database. It should be noted that any system that contains full audit information can provide the information necessary for this system. The six Clintrial variables are CT_RECID, RID, ENTRY_ID, ENTRY_DATETIME, MERGE_DATETIME, and STATUS. How each of these variables can be used follows:

CT_RECID contains the unique identifier for a record in Clintrial. This allows loading files back into Clintrial with changed values while retaining audit trails (either explicitly or implicitly), and actions upon records to be traced. The audit trail ability is key for the identification of data discrepancies. It provides, in conjunction with the other five variables, the ability to identify a record that has changed, who changed it, and ultimately why the record was changed. CT_RECID can also be carried
through listing and report programs and used as positive id for the source of information.

RID contains the unique identifier for a record in a particular Clintrial table. This variable is populated when data is merged to the data table. It starts at 1 and is incremented with each record that is added to the data base. As such, it gives the order in which items were entered into the database and is more easily identified (to the human eye) than CT_RECID. RID can also be used to group the data by percentiles from start of study, allowing investigations of the rate of data discrepancies as a trial progressed.

ENTRY_ID on the update and data tables contains the id of the person who last touched the record. This permits the identification of the individual who initially entered the data or changed the data. In the audit table it contains the ID of the person who either entered the original or who made the modification. As such, it is the keystone of any system of accountability.

ENTRY_DATETIME contains the date and time of the last action performed on the record in the update table. It is this variable that along with the records from the audit table, gives the ability to recreate the condition of the database at any point in time during a trial, and so to identify the cause of changes.

MERGE_DATETIME contains the date and time that the record was merged from the update table into the data table or the last time the record was changed or viewed as long as the viewing of the record was not aborted.

STATUS contains values that identify where a record is in the Clintrial process. Status indicates whether the record has been entered (and how), verified, validated, modified or deleted and appears on the update, data, and audit table. It is useful for answering questions about how many record items are how far into the process.

With the six Clintrial variables in hand, the ability to extract the audit tables from Clintrial using SAS/ACCESS provides with some formidable tools to ease the clinical trial process. These variables and the table layouts give the programmer ability to run SAS programs on the data outside of Clintrial and load the results back into Clintrial with the Clintrial audit trail being complete. As a result of being able to maintain the audit trail, the data presented to SAS programs can be tailored and analyzed. It is then possible to present the clinical data as it existed on any particular day of the trial, thus permitting the checking of program changes through the trial. This allows the results that were obtained with program version 1.0 and data version from May 9, 1997 which were sent to client/FDA to be checked against the results from program version 3.5 on the same data, in order to see if that little bug/feature that was found impacted anything. Problem resolution then becomes much easier when the various versions of data and programs are available.

The audit trails also allows checking on the records that have been changed. When the query for a listing of all patients who ever had a particular COSTART code(s), due to new interpretations of results, by looking for said code values in the data and audit tables, records which do not have that value in the data table can be found in the audit table.

With these tools we can produce a myriad of reports covering all aspects of the data experience. As the identification of data exception types is made possible by the Clintrial variables ENTRY_ID and CT_RECID and the audit table, comparing the information in the table and its audit table will determine who made the original data entry, what date it was entered, what time it was entered, and what item(s) were changed. This allows the monitoring of data exception generation by day of week, time of day, study, item, daily volume, as well as study related items as clinician/center, CRF page, and so forth. Monitoring these types of items allows for the fine tuning of CRFs, motivating personnel, allocating resources more efficiently.

Among the first areas identified for monitoring reports were data entry (single and double data entry), data verification, quality control checks, and data exception checks. The scope of these problems ranges from the design of the CRF through the data entry phase to final quality assurance. If data entry data discrepancy rate was too high, and discrepancies were randomly distributed, then that would indicated that the data entry operators would need further/better training/supervision. If the discrepancies centered around a few personnel, that would indicate that only those should be attended to. However, clustering of data discrepancies by panel or panel item would indicate a less than optimal CRF design or incomplete instructions for the data entry personnel. Data discrepancy clustering by investigator is obvious.

The ability to run SAS programs on the data as existed on any given date can be easily generated through the creation of a data warehouse in SAS containing all versions of the records. The merging of the data table with its audit table allows the construction of a SAS dataset that can then be indexed by date, any time slice generated simply by taking the last record on or before the date in question. This feature allows a direct
comparison of the results between two versions of a program, a program on two different data dates, and so forth with ease. Retaining the CT_RECID in the datasets used for production of reports generates the positive id needed to track discrepancies down. To track an issue, simply run in the debug mode which places the CT_RECID, with any other variables that might help to the right of the standard table or listing and print on legal size paper. This makes the positive identification of the values used in a report much easier.

In conclusion, being able to retain the full record of the trial allows for the generation of a complete system with positive identification of all data discrepancies, whether generated by data modification or program modification. The increase in overhead that accompanies this particular approach is significant but with increasingly powerful computing resources becoming available, the deacon to implement is becoming easier and easier to justify.