SESUG Paper 123-2018

# Hardcoding In Clinical Trials: A (Sometimes) Necessary Evil

Gregory Weller, Rho®

## ABSTRACT

Hardcoding is generally considered to be a dirty word in the clinical trials programming world, especially in statistical programming. However, sometimes it is the only way to accurately report data that is known to be incorrect.

This paper will explore two examples from clinical trials where it was determined hardcoding post database lock was necessary to fix incorrect raw data. Every step of the hardcoding process will be explained, including identification of the problem, figuring out the appropriate solution, and the documentation involved. This paper will also explore an example of a programmer unintentionally hardcoding while trying to handle problematic raw data values. While unintentional hardcoding is not done with malice, it is nonetheless a serious problem because: 1) it bypasses the steps required to properly validate and approve hardcoding, and 2) the code resulting from an unintentional hardcoding is unlikely to be useful beyond the specific version of raw data it was created for.

## INTRODUCTION

Before proceeding, it is important to define exactly what hardcoding is. In her 2000 paper "I don't Look Good in Orange or Stripes," Susan Fehrer defines hard coding with respect to clinical databases as "…the programmatic changing of data in a database, without an audit trail, that had been incorrectly data entered from a Case Report Form or other data collection medium."

While the above definition of hardcoding is sufficient for clinical databases, it does not quite suffice for other types of clinical trials programming (though it does provide a good basis for the definition). For other types of clinical trials programs, including but not limited to SDTM, ADaM, Tables, Listings, and Graphs, a more applicable definition of hardcoding is: Altering source data within a program in a manner that fundamentally changes the information contained in the data.

A key piece of the above definition is "fundamentally changes the information contained in the data." As an example, consider a case where an SDTM LB program creates the original unit variable LBORRESU by directly pulling a variable called UNIT from the clinical dataset. For one test, the value of UNIT is reported as "10^6/mL" (million per milliliter). This unit is equivalent to "10^9/L" (billion per liter), which happens to be the proper controlled terminology value that should be presented. Thus, if a SAS® program contains the line:

```
if LBORRESU = "10^6/mL" then LBORRESU = "10^9/L";
```

this would not be considered a hardcode since the fundamental information (including the lab result) is not being changed.

## DECIDING TO HARDCODE

When should hardcoding be used to fix a data issue? The short answer is that it should be used only as an absolute last resort when there are no other options. Every effort should be made to correct the data issue at the source, but that is not always feasible. A good example of this in clinical trials is a data issue that is found after database lock (DBL). Due to the effort that goes into locking a database, and the effort that would be required to re-open the data for editing, the most effective solution to fixing data issues is to handle it in SDTM programs. However, this does not apply to all situations. Sometimes an issue is critical enough that it would justify unlocking the database to fix, but that needs to be decided on a case-by-case basis.

## DOCUMENTATION

Once the decision to hardcode has been made, it is critical to document the reasons why. Since the changes are being made outside of the usual audit trail, this documentation will provide a sufficient audit trail for future reference. A programmer should always follow their company's SOPs and other guidance for documenting hardcoding, and the steps listed below represent the bare minimum requirement.

The first piece of documentation that needs to be created is a "Note to File" (NTF, sometimes called a "Memo to File"). The Note to File is written in the style of a letter or memo, and for a clinical trial it will be placed in the Trial Master File (TMF). The NTF should provide a detailed explanation of: 1) what the data issue is, 2) how the programmer knows what the correct value of the data should be, 3) justification for why hardcoding is the best solution, and 4) any other supporting documentation deemed necessary or relevant. The NTF should be signed by all relevant stakeholders to indicate they are aware of the issue and the solution.

Second, the hardcoding must be documented in the program with a comment. The specific hardcoding and reasons for doing so must be made explicitly clear for any person who looks at the program in the future.. If the hardcoding is straightforward, the comment can be brief. Something along the lines of "<Action Taken> See the Note to File from yyyy-mm-dd" will usually suffice. Less straightforward instances of hardcoding will require more in-depth explanation. Again, a programmer should always follow their companies SOPs.

## EXAMPLES

This section will discuss 2 examples actual clinical trials where hardcoding was necessary to fix data issues.

### EXAMPLE 1

Company A started a large Phase III clinical trial, and midway through the trial the drug was bought by Company B. Since Company B would now be running the trial, they decided to keep using the same CRF and EDC that Company A had set up.

Every clinical dataset contained the variables SITEID and SUBJID, and USUBJID was derived in the usual way by concatenating STUDYID, SITEID, and SUBJID. A subject who failed screening was allowed to rescreen, but re-screen subjects were issued new identifying information. Unfortunately there was no mechanism set up in the CRF to link their old and new identifying information. Further, it was discovered that there were 4 subjects who transferred sites in the middle of the study and were issued new identifying information at their new site. All of the information regarding re-screening and transfer subjects were documented in individual Notes to File.

The FDA Study Data Technical Conformance Guide (page 9) is very clear that a subject can have only one value of USUBJID. In this case it was clear that the only way to meet this requirement was to hardcode the proper USUBJID values at some point along the way. It was decided that the re-screen subjects would maintain their latest value of identifying information, but the subjects who transferred sites would maintain the identifying information from their original site. The clinical demographics dataset contained one record per subject per screening attempt (the subjects who transferred sites had demographic information filled out again too), and in the SDTM DM program there were a series of "if-then" statements to identify the duplicate subjects and keep their appropriate record in DM. For the re-screen subjects, their original identifying information was placed in SUPPDM as is the standard approach (QNAM was assigned to be ORIGID). For the subjects who transferred sites, only their original identifying information was kept in DM and nothing was included in SUPPDM.

In the rest of the subject-level SDTM programs, properly assigning USUBJID required a separate approach for the re-screen and transfer subjects. The first step was to create a USUBJID variable from the clinical datasets by combining STUDYID, SITEID, and SUBJID. The rescreen subjects are identified by having an observation in SUPPDM with QNAM=ORIGID. If a derived USUBJID value matched a value of ORIGID, the USUBJID value was reassigned to the USUBJID value from SUPPDM. For the subjects who transferred sites (who do not have any other identifying information in DM/SUPPDM), a

macro to remap USUBJID within a WORK dataset was created for both the production and validation programmers to use. This macro looked for instances of their new USUBJID and remapped it to their old value. Here is an example code:

```
%macro remap;

  if USUBJID = "ABC01-004-0452" then USUBJID = "ABC01-002-0124";
  …

%mend remap;
```

Since the information about re-screening and transfer subjects was already contained in many separate notes to file, it was decided that the complete list of old and new subject identifying information would be compiled into one note to file. The original notes to file were maintained, but having all the information in one document was much more useful to the project team.

## EXAMPLE 2

In a randomized double-blind Phase II trial, subjects come in to the site for a visit every 2 weeks once they are randomized. The on-treatment period could last up to 12 weeks, so this meant up to 6 on-treatment visits. At each of these visits, they are dispensed a bottle of 28 tablets (active study drug or placebo) which they have to take twice daily.

Being a double-blind study, the containers that a site has are only identified by a 10 digit code. When a site needs to dispense a container to the subject, the randomization system (IWRS) tells them which container to dispense. The investigator also needs to record in the CRF that they dispensed the bottle to the subject.

In this study, the EX dataset was one record per subject per bottle dispensed. EXREFID was the investigator-reported bottle number that the subject was dispensed (which should match the value from the IWRS), EXSTDTC was the dispensed date, and EXENDTC was the returned date. Subjects were not required to keep a diary of when they took the medication each day, but returned any untaken pills to the site at their next visit. After all subject visits had concluded, but before unblinding, the database was finalized and locked. After the data was unblinded, the following observation appeared in EX:

| | USUBJID | EXREFID | EXTRT | EXDOSE | EXSTDTC | EXENDTC |
|---|---|---|---|---|---|---|
| 1 | ABC-001-001 | 1046837723 | | . | 2018-01-14 | 2018-01-28 |

**Table 1. EX Dataset Output**

Treatment and dosing information (EXTRT and EXDOSE) came from the IWRS system, so there was no reason for them to be missing when the database was finalized. The only explanation was that the reported container number (EXREFID) was not accounted for in the IWRS data.

After some investigation, it was determined that 1046837723 was in fact not a valid container ID number. However, the IWRS system data showed that the site was instructed to dispense container 1046837**6**23 to that subject on 2018-01-14. Reporting the correct study drug exposure for a subject is paramount for ensuring the legitimacy of a study, and the study team needed to determine the best course of action.

The first step was to determine whether the subject was actually dispensed a container on 2018-01-14. They did come in for a scheduled visit that day, and they were due to receive a new container of study medication. Further, the associated CRF page should have only been filled out if a subject received study drug. Given this information, the study team did conclude that the subject was dispensed a container, and the next step was to determine which container that was.

The consensus among study team members was that the most likely scenario was the subject got container 1046837623, but it was improperly reported in the CRF and nobody caught it. However, being the most likely scenario still doesn't guarantee this was what actually happened. The programming team did some more investigating to find evidence to corroborate this theory and found the following

information: 1) the subject was assigned to the Placebo treatment group, 2) container 1046837623 contained Placebo, and 3) no subject in the study was ever dispensed the wrong medication in error.

Given all the evidence, the study team could say with a high level of confidence that the subject actually received bottle 1046837623 and nobody discovered this data error until it was too late. The decision was made to hardcode the proper EXREFID value for this observation in the SDTM EX program. To comply with documentation requirements, the following Note To File was written:

*To Whom It May Concern,*

*On 2018-01-14, CRF data shows subject ABC-001-001 was dispensed study drug container 1046837723. However, this is not a valid container ID in this study. The IWRS system reports this subject was assigned container 1046837623 on 2018-01-14. Subject ABC-001-001 was assigned to the Placebo treatment group, and container 1046837623 is identified by the IWRS as containing Placebo. Further, no subjects were found to have been assigned an incorrect treatment container during this study. For these reasons, 1046837623 has been hardcoded to the EXREFID variable in the EX dataset for this observation.*

## UNINTENTIONAL HARDCODING

The previous examples illustrate times when hardcoding is necessary to correct bad data. However, programmers occasionally use hardcoding as a shortcut when it is not appropriate to do so. Sometimes this is done with malice by a lazy programmer who does not want to deal with a complex data situation. Other times a well-intentioned programmer will commit hardcoding without realizing it. As a general rule of thumb, code should be made as robust as possible to handle all data in a generalized manner and not targeted to specific data points.

Consider the following example of unintentional hardcoding from an LB (Lab) SDTM program. One of the results reported in the clinical data is the ratio of neutrophils to leukocytes (LBTESTCD = “NEUTLE”). These results are supposed to be reported as a percent, with a value ranging from 0 to 100, but for an unknown reason one of these observations was reported as a fraction instead of a percent in the clinical dataset (hence the result was off by a factor of 100). An excerpt of the LB dataset with the standard results read in directly from the clinical data is provided below. One NEUTLE observation is in the proper unit (%), and the other observation is in the undesired unit of a fraction.

| | USUBJID | LBSEQ | LBTESTCD | LBTEST | LBSTRESN | LBSTRESU |
|---|---|---|---|---|---|---|
| 1 | ABC01-012-0067 | 52 | NEUTLE | Neutrophils/Leukocytes | 58.7 | % |
| 2 | ABC01-028-0144 | 18 | NEUTLE | Neutrophils/Leukocytes | 0.612 | fraction of 1 |

**Table 2. Example LB Dataset**

Since all of the NEUTLE fraction results needed to be converted to a percent, and there were only 3 instances of this happening in the clinical data, the programmer used the following code in their LB program:

```
data lbremap;
  set lb;
  if usubjid=“ABC01-028-0144” and lbtestcd=“NEUTLE” and lbseq=18 then do;
    lbstresn=61.2;
    lbstresu=“%”;
  end;
run;
```

Upon first glance there might not appear to be any problem with the above code. The programmer is taking a problem value (.612) and correcting it to the proper value and unit (61.2%). However, what if in the next data cut there is another NEUTLE record that is reported in fractions? Or what if in a future data cut the problem observation is no longer LBSEQ=18? As you can see, hardcoding the fix by focusing on

this one specific observation is not a robust solution. A better approach without hardcoding would have been to set up the program to handle any instances of fraction NEUTLE values. For example:

```
data lbremap;
  set lb;
  if lbtestcd="NEUTLE" and lbstresu="fraction of 1" then do;
    lbstresn=(100)*lbstresn;
    lbstresu="%";
  end;
run;
```

## CONCLUSION

Every effort should be made to ensure that clinical data is correct at the source. When data issues are discovered and fixing the source data is not possible or practical, fixing the data downstream via hardcoding is usually the simplest approach. When deciding to use hardcoding to fix data issues, the reasons for doing so and the process itself must be thoroughly documented. Programmers should also be careful to avoid unintentional hardcoding.

## REFERENCES


Study Data Technical Conformance Guide v4.1, U.S. Food & Drug Administration, Available at https://www.fda.gov/downloads/forindustry/datastandards/studydatastandards/ucm384744.pdf

Fehrer, Susan M. 2000. "I Don't Look Good in Orange or Stripes Subtitled, 'Hard Coding Clinical Data is not Permissible.'" *PHARMASUG 2000 Conference,* Seattle, WA. Available at https://www.lexjansen.com/pharmasug/2000/DMandVis/dm14.pdf


## CONTACT INFORMATION

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


Gregory Weller
Rho, Inc.
919-595-6436
greg_weller@rhoworld.com