JOURNAL OF CLINICAL RESEARCH BEST PRACTICES

Vol. 4, No. 11, November 2008 "Can You Handle the Truth?"

Verifying and Validating Clinical Data Management Software

By Sarah Hammond and David Hammond

Most clinical research studies employ clinical data management (CDM) software for entering and verifying data from case report forms (CRFs). CDM software may capture data from paper CRFs, or it may include an Electronic Data Capture (EDC) (electronic Case Report Form (eCRF)) module. With an EDC module, the paper CRF usually becomes a web page. While some companies build their own CDM systems, most purchase "validated software packages." With these commercial products, the supplier has already validated the software. It also usually validates installation for a specific study.

Software testing is a complex field, and the rules and standards for testing software are, in many cases, overly detailed. When a clinical data management program is installed and validated, an assumption can be made that the essential functions work correctly. That is to say, when you build a table in the software, the table formats correctly, allocates memory correctly and retains data correctly. However, just because the software is validated, doesn't mean that what you build inside the software will work correctly. Much like a sterile operating room, if you bring in dirty tools, it doesn't matter how clean the room is, the patient will still end up with an infection. It is important that within the "clean" environment of a validated clinical data management system, the studies you build are "sterile" as well. To ensure that your studies function correctly in CDM systems (CDMS), it is necessary to institute a scaled-down version of verification and validation (V&V) for each individual protocol. In this article, we look at the broad landscape of software V&V and will help you design and document a test procedure for each new study, allowing you to focus on the items likely to cause problems and decrease time spent on non-critical path items. This documented V&V procedure will provide you and your company with:

- Assurance that critical functions have been documented and tested.
- Documentation of the database design and functionality pathways. This is important in those instances when the personnel involved in the design of the database are suddenly no longer available to the company.
- Documentation of the testing plan, the testing done, and the results of that testing. Having been involved in numerous audits, we find that most auditors find the presence of this documentation reassuring, and this initial work will often pay off in assuaged concerns at the completion of the study.
- An opportunity to correct issues, even minor ones, before the study is handed off to data entry personnel. This should speed data entry as you will spend less time fixing the data entry functions and delaying data entry while the corrections are made.

In 2002, the FDA released a guidance document on General Principles of Software Validation; Final Guidance for Industry and FDA Staff.[1] This guidance document, while really written for software in medical devices or as part of a quality system, provides a basic framework for testing any software. The guidance document focuses on the two key parts of testing software: verification and validation.

- Verification = did we build the thing right?
- Validation = did we build the right thing?

According to the IEEE Standard Glossary of Software Engineering Terminology, verification is defined as, "the process of evaluating a system or component to determine whether the

products of a given development phase satisfy the conditions imposed at the start of that phase." Simply put, verification demonstrates that the output conforms to the input.[2]

Validation is defined as, "the process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements." [3] Validation demonstrates that the system is operational and checks that the software conforms to the requirements that were defined at the beginning of the software development lifecycle.

Any time a new study is started or a new function is utilized in a piece of validated software, testing should take place to ensure that everything is working as it should. When the FDA comes to audit this study software, the auditor will want to see that you have tested the functionality adequately, corrected any mistakes, and documented the process. There is a lack of specific guidance on how this testing should be designed, executed or documented, but in evaluating the FDA's guidance documents on software testing, international standards for software verification and validation (V&V), and general quality system regulations (QSR) guidelines for documentation, a pathway emerges.

This probably all makes sense to someone who spends his or her life in software development and testing, but to be honest, the reason a company buys validated software is because it does not have ready access to the necessary software specialists. In the absence of such specialists, how does a clinical research associate or clinical data manager take these requirements and convert them into a set of activities that will help establish that the software works properly and document that the testing was done?

Let's take walk through the process. You need to set up a new study in your company's CDM software. How can you build this study while ensuring that your set-up works correctly? This can be done in a simple five-step process:

1. Build
2. Document
3. Verify
4. Validate
5. Document

1. Build

There are a couple of tasks that need to happen prior to simply building the study. The first, and most important, is to define what the study is supposed to look like in the CDMS and create the specification requirements. Documentation of the requirements is vital because it gives you a solid foundation on which to plan, design and execute the test activities. In other words, if you don't know what the software is supposed to do, it is hard to verify that the software works.

Most software companies have defined processes for the creation, documentation and approval of the specifications. However, these processes probably do not exist in your company. The "Build Plan" doesn't need to be overly complex. Keep it simple: Document what you want the study (as a whole) to look like; where the data should go (separate fields, different tables); how the data should go in (data entry screens, autofills) and what the data should look like (numbers, text, dates and images); and how the data should come out (in reports or downloads). Once these specifications are approved by all parties, the study can be built in the CDM software.

2. Document

The documentation step appears twice in this process. Before you launch into testing your new study in the CDM software, you need to construct a test plan. The list of elements for a software test plan according to the IEEE Standard Glossary of Software Engineering Terminology is listed in Table 1.

However, for this type of testing, the list can be shortened to:

1. Introduction
2. Test Items
3. Features and fields to be tested
4. Features and fields not to be tested
5. Testing methodology
6. Item pass/fail criteria

Table 1. IEEE Test Plan Elements

1. Test plan identifier
2. Introduction
3. Test items
4. Features to be tested
5. Features not to be tested
6. Approach
7. Item pass/fail criteria
8. Suspension criteria and resumption requirements
9. Test deliverables
10. Testing tasks
11. Environmental needs
12. Responsibilities
13. Staffing and training needs
14. Schedule
15. Risks and contingencies
16. Approvals

Introduction

- The name and version of the protocol being used in creating the study in the CDMS.
- The version of the case report forms being used in creating the study in the CDMS.
- The estimated time period during which the testing will take place
- Who will participate in the testing. It is important that the person setting up the testing bring a variety of people together to do the testing. A cross section of personnel from data management (both the designers and the data entry personnel), clinical operations (the CRA and/or project manager), and biostatistics will cover most of the groups with a stake in the collection and reporting of the data from the system. Miller et al suggest that you think of your testing (or oversight) group as an IRB, with representatives for each group providing feedback from his or her own perspective.[4]

Test Items

- Test items and their version
- Characteristics of their transmittal data
- References to related documents, such as requirements specification, design specification, user's guide, operations guide, installation guide
- References to defect reports that describe the defects identified

Features and Fields To Be Tested.

- A list of any features in the software or fields in the study that are being tested
- In general, fields involving primary endpoints, subject safety (adverse events and SAEs), protocol deviations, and product tracking should ALWAYS be included in the features to be tested
- Any field utilizing any kind of internal logic or calculations should ALWAYS be included in the fields to be tested

Features and Fields Not To Be Tested

- A list of any features in the software or fields in the study that are not being tested
- Why these features are not being tested

Testing Methodology

- How the testing will be done (e.g., manually or using automated scripts)
- How information will be collected (e.g., where you write that the date field does not accept alpha characters)
- Any dummy CRFs used to enter data into the CDMS

Item Pass/Fail Criteria

- The specific criteria that will determine whether each test item has passed or failed, e.g., 95% of all test cases must pass and there are no unresolved critical defects

3. Verify

Once the specifications are documented and approved, the test team uses them to begin planning the test cycle. In the verification phase, you create and conduct tests that tell you the study is built correctly. For instance, part of the study may include electronic case report forms. Most of the fields on the form require only alpha characters; however, there are a few fields on each screen that require alphanumeric characters. A suitable set of verification tests might include the following:

1. The 'Name' field is an alpha field. Enter "A" in the 'Name' field.
2. The 'Name' field is an alpha field. Enter "9" in the 'Name' field.
3. The 'Name' field is an alpha field. Enter "%" in the 'Name' field.
4. The 'Comment' field is an alphanumeric field. Enter "A" in the 'Name' field.
5. The 'Comment' field is an alphanumeric field. Enter "A7" in the 'Name' field.
6. The 'Comment' field is an alphanumeric field. Enter "A7#" in the 'Name' field.
7. The 'Phone' field is a numeric field. Enter "6" in the 'Phone' field.
8. The 'Phone' field is a numeric field. Enter "P" in the 'Phone' field.
9. The 'Phone' field is a numeric field. Enter "#" in the 'Phone' field.

For testing CRF data fields in CDM software, the simplest method is to create a series of dummy CRFs with fake data to enter into the database. So, from the list above, if a field is designed to hold a subject number that must be numerical, try entering letters, symbols like "#," and a combination of letters, numbers and symbols.

Data limits and pre-selected values are data type checks (filters) that allow only certain data to be entered into a field.[9] Data type checks verify, for example, that date fields allow only valid dates, and yes/no fields allow only "Yes" or "No." Numeric data limits allow any number that falls within a specified range. Pre-selected values provide a list of acceptable answers and do not allow manual entry of a different answer. These field characteristics reduce errors in the data by preventing obviously incorrect answers, such as a patient height of 5.9 inches or a gender of "None." The risk is that if a legitimate possibility is excluded, research sites will be unable to enter the data. When generating any data-limited or pre-selected value field, be absolutely sure that the limitations are correct. To play it safe, include an 'Other' field. Consult with the medical affairs staff and people knowledgeable about the condition under study, the investigational product, and medicine in general.

The next step is to verify that the data are entered into the correct fields in the database table. This verification requires opening and inspecting data sets at the table level.

Misplaced data can often be traced back to incorrectly labeled fields. For example, on the data entry screen, each field has two parts, a label and a box into which the data is entered. When the data screens are set up, it is a simple mistake to reverse two labels so that a field labeled as 'Initials' is actually the 'City of Birth' field.

Many of these tests can be reused from field to field and from study to study. The number of test cases in a study depends on the design of the study and the amount and type of data. The tests should thoroughly cover primary endpoints, safety (adverse events), and product accountability. Less important data requires less comprehensive testing. For a typical study, 10 or fewer dummy CRFs are required.

The next step in verification is testing the output of data from the database. This step involves generating reports using the data entered in the database. Reports may include calculated data along with site-entered data. Write a set of test cases that track entered data through to the reports. Manually verify that data values, calculations and labels are correct and located in the right places in the reports.

As more hospitals move to electronic medical records (EMR), it becomes more likely that study data will be downloaded from their database and uploaded into yours. Testing this functionality requires dummy downloads from the source database (like fake case report forms). Upload this fake data and verify that each data item ended up in the correct data field with the correct subject identifier and date. Frequently, EMR data is lab data. Labs often use different units for certain tests (e.g., mg/ml vs. µg/ml or mmHg vs. kPa), so careful review by knowledgeable people is required.

A complete verification test suite provides test coverage on every aspect of functionality expected to be used during the study.

4. Validate

Validation tests employ use cases that "put the system through its paces." Use cases are based on the specifications, but tend to be broader and less specific than verification tests. For instance, instead of focusing on the fields themselves, the tests for validation of a CDM form might follow this user scenario:

Subject Quincy John Adams, III came in for his first visit in a drug trial. Populate the following fields with the stated values and press [Enter] to proceed to the next form.

First Name = Quincy

Last Name = Adams

Middle Initial = J

Suffix = 3

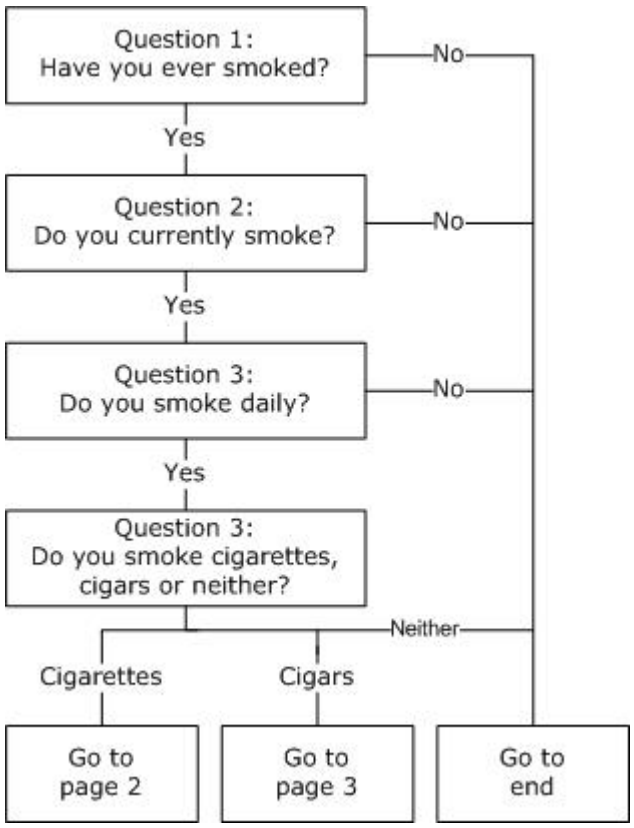
Medication = Aspirin

Reaction = Rash

Comment = Patient took 3 tablets, and then broke out in a rash 20 minutes later. Offered investigator $50 in cash for a tube of hydrocortisone.

The above test follows a process from start to finish and thereby validates whether the system can handle a real-life usage scenario. This testing is best done by the people who will be entering the data into the CDM software. It also ensures, for example, that the forms flow correctly from one field to the next, that the fields are large enough to visualize all the information without scrolling, that all data points necessary for any calculations are present, and that any logic, data limits, or preselected values function as intended.

Figure 1. Skip Logic Flowchart



Let's focus on these last two activities: the presence of necessary data points and the skip-logic, data limits, and preselected value functions. The design and testing of CDMS often falls down in these areas.[5]

One of the advantages of using CDMS instead of paper CRFs is that the data entry system can automatically calculate values, such as converting inches to centimeters to calculate more complex scores like a Body Mass Index.[6] However, if the CDMS calculates any values, validation must ensure that the input data is present. Let's take a simple example, the Apgar score for evaluating the health of newborn babies. The Apgar score is based on five criteria: **A**ppearance (skin color), **P**ulse (heart rate), **G**rimace (reflex irritability), **A**ctivity (muscle tone), and **R**espiration (breathing).[7] Each of these criteria is scored from 0-2 for a total of 0-10. If your study in the CDMS only collects Pulse, Activity and Respiration, you will be unable to calculate Apgar scores. In more than one study, one or more of these criteria have been missing from a case report form prior to validation. Verification and validation tests may seem obvious, but they often find errors that would be very costly – and embarrassing – if not identified before case report forms are designed and the database is built.

Electronic Data Capture (EDC) forms often include skip logic, data limits, and preselected values. Skip logic is the electronic version of the instruction, "If yes, skip to question 3."[8] It speeds data entry by allowing users to automatically move to the correct field instead of having to tab through irrelevant data fields or click on "Next Page." However, if the skip logic takes the user to the wrong field, then the right data can end up in the wrong place. Skip logic should therefore always be tested. Testing is simple and can be documented using a flowchart, as in Figure 1. Questions 1, 2 and 3 are all Yes/No questions (e.g., inclusion/exclusion criteria). If you answer "Yes" to any one of these questions, the software will lead you to the next one. If you answer "No," it will lead you to Question 4. Question 4 is multiple choice; each choice either leads you to a different page or ends the questions.

A flowchart like Figure 1 shows you which pathways should be tested and which ones are unnecessary. For example, if you answer "No" to question 1, it should take you directly to question 4; therefore, you would never need to test what happens with questions 2 and 3 after you have answered "No" to question 1.

Skip-logic testing can be facilitated with a scenario table. Figure 2 is an example scenario table. This table starts with all the possible permutations of "Yes" and "No" answers and then grays out any combinations that should never occur in the study. As you can see, you start with 48 possibilities, but after removing the ones that your workflow should avoid, you reduce the number to 22.

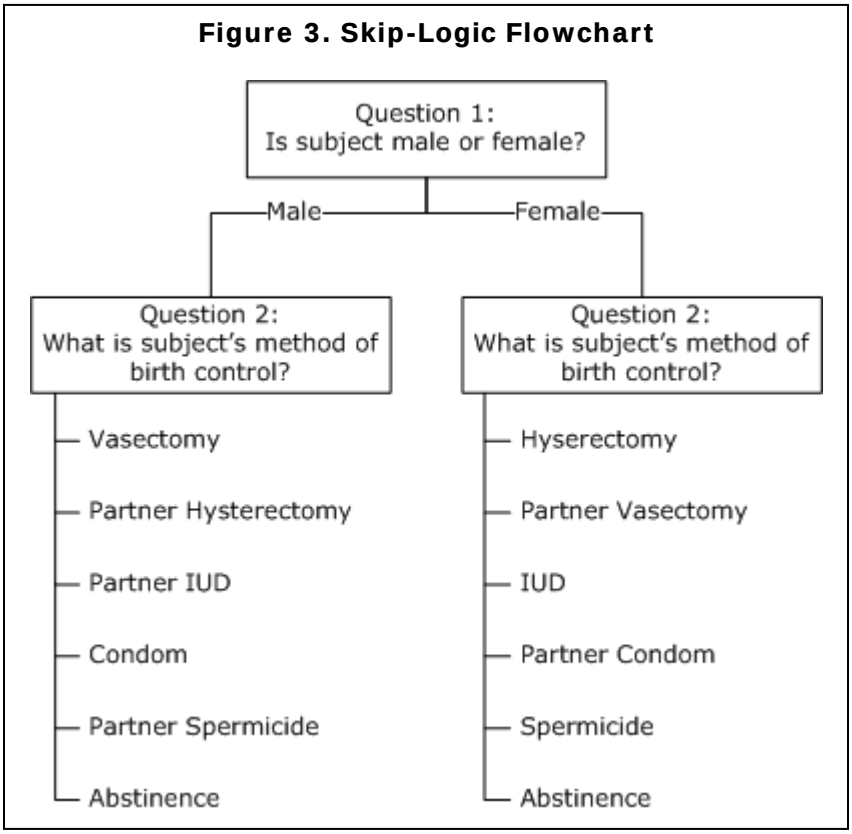
Figure 2. Skip-Logic Scenario Table

	Male		Female	
	Y	N	Y	N
Abstinence	x	x	x	x
Barrier	x	x		
Condom	x	x		
Hysterectomy			x	x
UD			x	x
Spermicide			x	x
Partner Condom			x	x
Partner Hysterectomy	x	x		
Partner IUD	x	x		
Partner Spermicide	x	x		
Partner Vasectomy			x	x
Vasectomy	x	x		

Verification gets more complicated with intra-form or cross-form validation checks or cascading fields.[10] With these types of fields, an answer to a prior question will limit responses to a later question (but not block the question entirely). Figure 3 shows a flowchart that is similar to the flowchart above for skip-logic. This flowchart will help you document all the possible permutations. Depending on whether the subject is male or female, the birth control choices change.

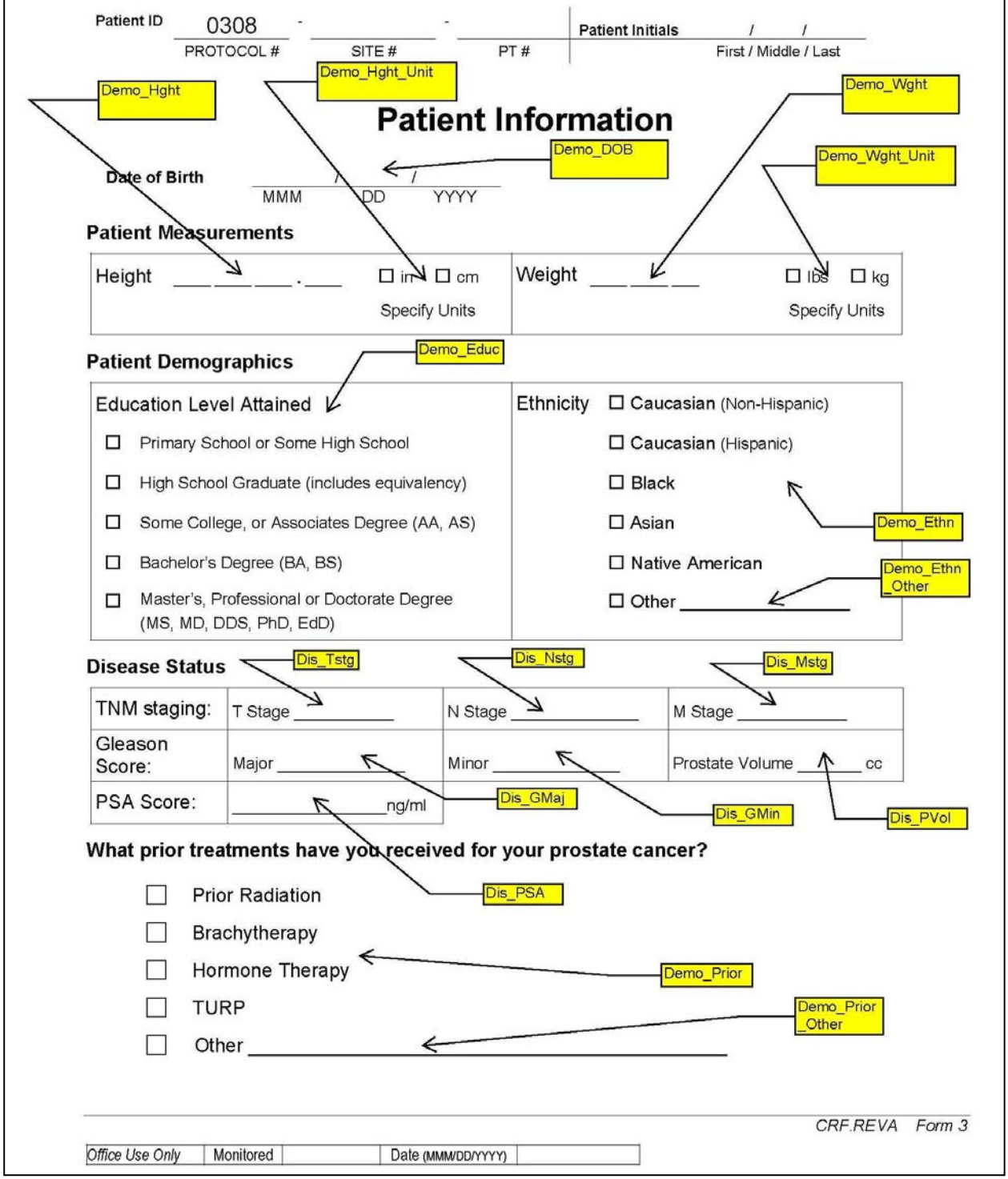
Now that you have ensured that the features work correctly, how do you establish that the study you've set up in Clinical Data Management Software has places for all of the study data, and none extra? The easiest way is to create an annotated CRF. As shown in Figure 4, an annotated CRF is a version of the CRF in which every field on the CRF is matched to its corresponding database field.

Figure 3. Skip-Logic Flowchart



Annotated CRFs can support a fine level of detail. For example, if you need to verify which patients enrolled between June and August 2008 received study medication during their three-month follow-up visit, being able to quickly find the necessary fields will make creating the necessary query a snap. You will notice in Figure 4 that each place to enter data on the CRF mentions an associated data field in a database table. These notes will allow you to quickly assess whether any fields are missing or whether two fields on the CRF write to the same field in the database.

Figure 4. Annotated CRF



The focus of validation is not only on the case report forms. Data can also come from other sources like lab reports and subject diaries. Document and annotate these sources like the case report forms.

5. Document

When the testing is complete, document what was done in a test report. The test report should include the desired functionality, test cases used to verify and validate the functionality, the outcome of those tests, and any changes made to the functionality before the study was launched. When an FDA investigator arrives for a post-study audit, use this document to reassure him or her that your study CDMS is correctly designed and that you tested all the critical functionality. For some of the simpler functionality, the entire report (including the test plan, test cases, and outcome) may only be a page or two long.

When testing large, complex software systems, you may generate multiple test reports with hundreds of pages of documentation for a single project. When testing functions in a validated CDMS system, the test plan and test report each may be only a few pages long, supported by documentation of test cases and other details.

Conclusion

After you have a validated installation of Clinical Data Management Software, the true work begins for the CRA or clinical research personnel in charge of building the study in the data management software and handling the study data. It is important to evaluate each study, evaluate which data will be stored in the CDMS, and develop a plan for creating a place in the CDMS for the data and for testing that the software functions the way you planned. For each additional step handled by the software, whether a calculation, skip-logic or fields with data limits, additional scrutiny will need to be given to those fields. The layout of the study within the CDMS, the testing, and the results should be documented in a verification and validation plan. The effort made up front should speed your data entry, provide documentation about your study in the CDMS just in case, and provide evidence to your management and any auditors that you've made an effort to ensure that your data is handled in an acceptable manner.

References

1. "General Principles of Software Validation; Final Guidance for Industry and FDA Staff" http://www.fda.gov/cdrh/comp/guidance/938.html. Issued Jan 11, 2002. Accessed August 11, 2008.
2. C. U. Smith, L.G. Williams, "Performance Engineering Evaluation of Object Oriented Systems with SPE-ED", Computer Performance Evaluation: Modeling Techniques and Tools (LNCS 1245), Springer-Verlag, 1997
3. Ibid.
4. Miller RA, Gardner RM. Recommendations for responsible monitoring and regulation of clinical software systems. J Am Med Inform Assoc. 1997 Nov-Dec;4(6):442-57.
5. Brandt CA, Argraves S, Money R, Ananth G, Trocky NM, Nadkarni PM. Informatics tools to improve clinical research study implementation. Contemp Clin Trials. 2006 Apr;27(2):112-22. Epub 2006 Jan 4.
6. Ibid.
7. Apgar, Virginia. "A proposal for a new method of evaluation of the newborn infant". Curr. Res. Anesth. Analg. 32 (4): 260–267
8. Brandt CA, Argraves S, Money R, Ananth G, Trocky NM, Nadkarni PM. Informatics tools to improve clinical research study implementation. Contemp Clin Trials. 2006 Apr;27(2):112-Epub 2006 Jan 4.
9. Ibid.
10. Ibid.

Authors

Sarah Hammond is a Lead Software Test Engineer at GE Healthcare and an independent CDMS consultant. Contact her at sarahdhammond@gmail.com.

David Hammond is the Clinical Affairs Manager at Calypso Medical Technologies, Inc. and an Affiliate Instructor in the School of Pharmacy at the University of Washington. Contact him at 1.206.650.7258 or david.t.hammond@gmail.com.