

Written and researched by:
Dr. Kirk Mousley
and Karl Mousley

The Importance of Specifications

EDC Today is an independent publication on current information and issues in Electronic Clinical Systems (ECS) strategies and technologies for the Biotechnology and Pharmaceutical (Biopharma) industry. Each month we examine topics related to ECS theory, technology, practice, or implementation.

EDC Management has long been aware that clinical trial-specific configurations of an EDC solution, especially those offered under the Application Service Provider model where the external vendor sets up the configuration, have placed a heavy burden on communicating the trial's requirements between the study's sponsor and the developers of the EDC solution. Since the developers are not usually familiar with the sponsor's way of doing things, many details, such as object naming conventions, codelist contents and styles, and other standards implicitly assumed to be known by in-house staff, are no longer a "given". Furthermore, often overlooked, is the need for communicating the post data capture requirements; that is, how, when, and in what format, the captured clinical trials data will be sent to the sponsor or the group performing the study analysis.

In this issue, we discuss the importance of specifications and how they entail planning and thinking ahead about various requirements as well as potential problems and issues centering on how a project's goals are to be met. Equally important is the necessity to have those involved in the project carefully review those specifications and understand that the specifications are a "contract" for the work to be done, which is especially critical when project members are geo-physically dispersed as well as organizationally separated.

In the next issue of
EDC Today:

The Role & Functionality
of Clinical Management
Systems

About EDC Management:

EDC Management is the leader in Clinical and Data Management and Electronic Data Capture (EDC) consulting services for the biopharmaceutical industry. EDC Management publishes well-researched and timely information about Electronic Data Capture technologies and processes through EDC Today[®] and EDC In Depth. We do not sell or endorse any specific EDC software application or vendor. Improve process today; position for tomorrow.

EDC Management

P.O. Box 23
Conshohocken, PA 19428
484-530-0300 (voice)
610-567-0357 (fax)
info@edcmanagement.com
www.edcmanagement.com



Introduction

Most Electronic Data Capture (EDC) study-specific configurations (as well as many other software-related) projects include a body of information that describes the application, desired results, or output of the project's work effort; this information deals with the objectives of the final application, defined in the project requirements, and any rules for creating the application, defined in the project specifications. Any well-conceived and reasonable project must have requirements that define what the project is ultimately supposed to achieve. The term “deliverable,” overused and misused, is often used to express the goal of a project.

There are two important roles for the collection of documents that describe requirements and specifications. First, these documents are a communication tool that convey to those who configure or build the software what the users need and expect. Second, these documents are a necessary part of the Food and Drug Administration’s (FDA) software validation regulations. The developed software must be proven to function as intended, and the intention of the software is documented in the requirements and specifications.

Any organization that believes they don’t have time to properly develop the requirements and specifications for a software project is mistaken. A well thought out specification will save more time in the end. The most expensive and time-consuming part of any project is making changes to software that has been previously developed and tested.

This issue of EDC Today® will discuss the important differences between requirements and specifications. It will stress how one must address several important areas in terms of setting up a clinical trial-specific application. Keeping requirements simple and understandable but sufficiently complete is a fine balancing act. Many times written prose is not appropriate; tables and flow charts can help make the specifications more visual. Moreover, sometimes the format of the specifications does not permit easy printing and subsequent review. Finally, this issue will discuss how standards (i.e., standard objects such as page layouts, items, and tables) can make the specification process less difficult if the standards already exist in the EDC solution being used to collect the data. If the EDC solution is new to the Clinical Data Management (CDM) group, standards developed for previous Clinical Data Management Systems (CDMS) application can be a useful and reasonable starting point.

A key point is that there should be no assumptions (or implicit understandings) made about how the application should be built or configured. Many projects performed within a single CDM group often use assumptions concerning naming conventions, codelist conventions, and other style expectations. The only reason that an in-house project can succeed with these assumptions is that they are taught as part of employee training, are documented in the sponsor’s Standard Operation Procedures (SOPs), or are otherwise communicated to internal employees. External parties most definitely do NOT know these assumptions.

(continued on page 3)



Requirements versus Specifications

It is important to understand the difference between requirements and specifications. Both are needed. One way to view the difference is to say that requirements detail WHAT is to be done and specifications detail HOW it is going to be done.

Quick Definitions:

Requirements are generally high-level business-oriented objectives and generally independent of implementation. Specifications are more detail-oriented and are normally implementation specific.

As can be seen in Table 1, requirements are high-level objectives. In EDC Management's experience, it has been noticed that at a significant number of clinical research organizations, many high-level requirements are often assumed. In fact, in many cases, requirements are not explicitly stated at all, and project documentation often starts with specifications.

Once high-level requirements have been thoroughly thought out and clearly stated, specifications can be developed that bridge the requirements to the actual implementation. As can be seen in Table 2, specifications might detail how laboratory normal ranges are stored, how laboratory values (i.e., results) are compared to the normal range, what the out of range flag(s) will be, and so forth.

Table 1. Examples of Requirements

1. All of the data described in the protocol must be collected and analyzed.
2. Blood samples will be sent to a central lab and lab values will be electronically merged with the collected clinical data.
3. Lab values will be compared to normal ranges, and will be identified as out of range as appropriate.
4. Adverse Events will be encoded using the Medical Dictionary for Regulatory Activities (MedDRA).
5. Concomitant medications will be encoded using MedDRA.
6. Ease of use is imperative as investigators are not computer savvy.

Table 2. Examples of Specifications

1. The visit (event) schedule for this protocol will be Screening, Administration at Day 1, One week post dosing follow up, Thirty day follow up, Adverse Events, and Concomitant Medications.
2. The demographic information will be collected on a page named DEMOG.
3. The Demographic Table will include the items date of birth, gender, height, weight, and ethnicity.
4. The items in the Demographics Table must be entered into the data (i.e., they cannot be missing at the end of the study).

(continued on page 4)



The stated requirements should be the guiding principles for how the application should function. A thorough understanding of these principles will make the specification process go more smoothly.

Derek Sisson, the Director of Quality Assurance & Usability at Grassroots Enterprise in San Francisco, sums up requirements and specifications quite nicely on his website:

Any coherent and reasonable project must have requirements that define what the project is ultimately supposed to do. Putting aside the particular document for now, requirements are instructions describing what functions the software is supposed to provide, what characteristics the software is supposed to have, and what goals the software is supposed to meet or to enable users to meet.

I prefer to use the term requirements to refer to the general set of documents that describe what a project is supposed to accomplish and how the project is supposed to be created and implemented. Such a general set of requirements would include documents spelling out the various requirements for the project — the "what" — as well as specifications documents spelling out the rules for creating and developing the project — the "how".¹

Communications

Both requirements and specifications are communications tools. These tools facilitate communications between the users or “owners” of the applications, and the developers or “providers” of the applications. These tools also facilitate communications within a group. For the creation of most sizeable software applications, as an EDC protocol configuration would be, many users and developers are involved.

William Wilson, of the National Aeronautics and Space Administration (NASA), said the following in his presentation on “Writing Effective Requirements Specifications” at the Software Technology Conference in Utah in April 1997:

The desirable characteristics for requirements specifications are:

- Complete
- Consistent
- Correct
- Modifiable
- Ranked
- Traceable
- Unambiguous
- Verifiable²

While the interested reader is encouraged to read all of Wilson’s comments on the above list, what he says about being “complete” follows:

A complete requirements specification must precisely define all the real world situations that will be encountered and the capability’s [i.e., the application] responses to them. It must not include situations that will not be encountered or unnecessary capability features.³

(continued on page 5)



While protocol configurations are not rocket science, Wilson makes a good point as to what should be considered when putting a specification together.

When requirements are compiled, it is useful to involve an analyst who knows how to elicit requirements from the users. Most users have an incomplete sense of what is needed. The job of the analyst is to help users develop a full and specific set of requirements and specifications. A good analyst is a tremendous asset to a company.

Karl Wieggers, in his October 2000 article in *Software Development* entitled “Usable Requirements: Habits of Effective Analysts” has this to say about analysts:

An analyst provides a specialized capability that can make the difference between a project that succeeds and one that struggles. ... There is no substitute for experience. One of my consulting clients discovered that they could inspect requirements specifications written by experienced analysts twice as fast as those written by novices because they contain fewer defects.⁴

Important Considerations

As mentioned above, specifications should be complete and should not assume the reader knows anything about the project beforehand. To delve into all the aspects concerning a specification would take several issues of EDC Today,[®] so this issue only discusses some areas that are, in EDC Management’s experience, often not adequately covered in documentation.

These include:

- details about what data points are critical for the statistical analysis plan
- when and what data (as well as in what format) should be received from a vendor
- what naming conventions for pages, tables, items, and so forth should be followed
- how partial dates should be handled
- what is expected in the way of Codelists (i.e., format, style, and contents).

EDC Management believes that every protocol should have an associated detailed listing of the criticality of data points as far as the statistical analysis program is concerned. This would be in the format of a requirement, and it should indicate which data points are absolutely required, which should be present but could be missing, and lastly which data points are not essential. With this list, specifications for edit checks and data entry field validations can more easily be developed. Obviously if a data point is critical to analysis, not only will an edit check be written to make sure the data point is properly captured and stored, but it is very likely that a data entry check (i.e., an entry form-level check) will be supplied in the EDC application to immediately alert the investigator site that the data point must be supplied and the format and value that is expected. For non-critical data points, this additional effort may not be worth the programming (depending on other considerations such as the size the clinical trial and how often the data point is collected).

(continued on page 6)



When it comes to working with external vendors such as Contract Research Organizations (CROs), central laboratories, or EDC vendors, sponsors must state clearly what data they want delivered on an interim basis and at the end of the study. Far too often, EDC Management has been involved in projects where specifications denoting what data, and the format and timeframe in which it needs to be delivered, has not been specified in contracts with external vendors. As a result, the sponsor, at a late stage in the clinical trial, has to work out a data transfer/transform process and often gets data in formats that are difficult to work with. In particular, contracts (which are a form of specifications) with central labs should clearly state what the lab data format is, what the units are for each lab test, and whether lab data transfers are complete “all the data-to-date” transfers, or are incremental transfers (i.e., updates) in nature.

Naming conventions are almost never included in specifications. This is unfortunate since other departments within a clinical research organization (e.g., sponsor, CRO, EDC Vendor) may have programs and processes that are built upon a certain naming structure. Internal to the clinical research organization, many of the application developers are probably aware of the conventions and make use of them without second thought. However, an external vendor most likely has seen many different naming conventions, and may in fact have a naming convention it normally follows in the case where its client does not explicitly specify any. As a result, the resultant datasets and data items may likely have different names than the client desired and/or expected.

EDC Management has also encountered little in the way of specifications on how a clinical research organization expects dates to be handled, especially in the case where different parts of the dates can be (and often are) missing. Many clinical research organizations will have developed some method for handling these sorts of dates. It appears, however, that these methods are either not documented, or are documented apart from the requirements and specifications for the particular clinical trial. Unfortunately, EDC management has dealt with several clinical research organizations that appear not to have developed a partial date strategy at all.

Lastly, many clinical research organizations do not have adequate documentation for their codelist conventions. Some organizations use numeric codes and some use character codes. Some organizations attempt to use somewhat descriptive codes within their codelists, for example using M for Male and F for Female while others do not. Even beyond whether the codes are numbers or letters are the issues of size, their decode values, special codes (e.g., codes not really members of the list but included to cover cases such as “missing”, “not applicable” and/or other situations). It appears that since codelists are often developed “globally” in the sense that they are used by applications developed for many different clinical studies, this information about codelists is assumed to be known by all involved and is often not included in specifications and is not passed on to the external vendor.

(continued on page 7)



Balance

For both specifications and software, the KISS (Keep It Simple, Stupid) method is almost always the best approach. The simpler one can keep the application and have it meet a majority of his or her requirements, the more likely the software can be developed within timelines and budgets. In addition, the simpler the software, the less expensive it will be to maintain.

Keeping a balance between too little detail and too much detail within specifications is necessary, but difficult. On the one hand, specifications need to be complete. On the other hand, specifications that are too detailed become intractable and expensive in terms of time and effort. Developers need a certain amount of flexibility to create a workable system and overwhelming them in details can strangle this flexibility.

Thus there are two ends of the spectrum. At one end, there are insufficient specifications, which usually means the developers end up making many guesses at what the user expects to be done and perhaps leads to a failed implementation. At the other end, there are specifications that detail too many things, which often leads to a cumbersome, and perhaps unsuccessful, implementation. In addition, extremely detailed specifications are often hard for people to adequately review. People often assume that sheer volume of specifications means that they are complete and well thought out, and do not do a good job of reviewing and critiquing them.

An approach to specifications that one might view as “standards plus guidance” may be the best way to find a good balance between too little and too much. In such an approach one would specify what standard components are to be included in the application as well as broad, generalized rules that apply to all pages, tables, data items (i.e., fields), and so forth. Then one would list the exceptions to the above rules. This approach allows completeness as long as the broad rules address all of the important requirements. It also allows flexibility for the developer to decide how to apply the rules to the individual objects in the implementation.

Specification Media

One limitation of specifications is the media that is used to capture them. It is unfortunate that the most often used forms to express specifications, Microsoft Word Documents and Excel files, are two-dimensional and are mainly prose based. As a result, specifications are often filled with cumbersome (both to write and to read) text, require difficult formatting, and are often difficult to manage in terms of both document and version control. To add to the document user’s woes, updates to the specifications (e.g., changes between versions) are often hard to indicate, find, and to track, and printing the document for the purpose of review (especially the case of multi-worksheet Excel file) can be an additional and unwanted complication.

EDC Management believes that other forms of representation of requirements and specifications (such as visual methods) are necessary to fully communicate what is desired. When it comes to setting up an EDC application, a paper CRF is still a useful method for determining how a page should be laid out. In the event that a paper CRF is not going to be produced for a protocol, form/page mockups should be completed in their place.

Furthermore, certain parts of the applications could be prototyped to aid in the development of the specifications, and to help those who are reviewing the specifications envision what is needed, intended and workable in practical terms.

(continued on page 8)



Changes to Specifications

While specifications need to allow flexibility, it should be obvious that changing the specification has a rippling affect throughout the whole project. This is especially true when parts of the specification have already been through the design process and have been or are being implemented.

For those not familiar with software development, it is usually easier to design the system and build it from the beginning, than it is to retrofit functionality into an existing system. Furthermore, changes to the specifications after development has started often mean that what has already been accomplished has to be reviewed at minimum, and in the worse case situation, discarded forcing the developer to start anew. Starting over is almost always the result of a specification change in regards to testing and validation. Once the software is changed, its testing has to be completed all over again.

An organization should endeavor to make its absolute best effort to complete carefully considered and reviewed specifications before software implementation starts. This requires a lot of heavy duty thinking, collaborating, reviewing, and even prototyping before the actual implementation process should begin. And once implantation begins, any further work on requirements and specifications should be performed with a different framework of thinking. The new framework should use a much larger cost factor, perhaps as much as double or triple the original cost factor, when evaluating whether a particular feature is needed.

Standards

Developing requirements and specifications is time consuming and should involve much thought. Clearly the quality of the specifications is an important part of implementing a successful EDC application. Specifications may undergo several revisions before the final application is completed.

However, once this process is done for pages that are used in multiple studies, these pages should be re-useable as they are, and the specification process need not be repeated. This is part of the practical benefits of having and using standards.

It behooves businesses to develop standards. EDC Management believes standard pages, that is, pages that have been configured and used in at least one study in the current data collection application, can result in huge savings of effort, time, and money. These pages have not only been specified and developed, but they have also been validated and real-life tested. For new studies that use these pages, it is almost sufficient to just include these pages in the new study.

However, if the clinical research organization is moving to a new EDC solution, the existing standards can at best, provide a good start for a new specification. The reason existing standards are only a good start for a specification is that the implementation details such as the metadata (e.g., item names and types; table names and structure, page names and layouts) as well as any programming that was created to support desired functionality of these existing standard pages, usually will not port directly to the new system, and will require re-implementation and testing. It may be necessary to reverse-engineer any edit check and derivation code. Hopefully, at a minimum, the original specifications used to develop the standard pages can to be reused.

(continued on page 9)



Conclusion

Ultimately, good requirements and specifications can help make the entire software development process go smoothly. When all groups involved have a good understanding of what is needed and how it should be accomplished, the resulting software has a good chance of being successful – where success is measured as the application meets the requirements of the user, is useable in practical terms, and was developed in time and within budget.

However, poor requirements and specifications usually lead to lengthy and reiterative development cycles and hit and miss software. Even a great programmer cannot overcome a poor specification.

Tyson Gill in his book *Project Planning for Successful Software Development*, sums up this point nicely:

Is the developer the blameless victim in all this? Certainly not! There are times when developers fail to deliver. ... Quite often, however, the developers are not the problem, or at least not the only problem. No developer, however talented and conscientious, can develop a project efficiently from a bad plan. The rule of “garbage in, garbage out” is universal.⁵

In addition to its role in communicating project needs and expectations, the collection of documents that describe requirements and specifications serves as the basis for regulatory compliance. The FDA requires software to be validated against its intended use. Without good specifications, a thorough and complete validation script cannot be developed. Without a good validation script, one will be hard pressed to show that the application is working as intended. Therefore, without good specifications, one is at risk that the FDA will not accept the use of the application as part of the clinical program.

Resources

¹ <http://www.philosophie.com/design/requirements.html>

² http://satc.gsfc.nasa.gov/support/STC_APR97/write/writert.html

³ *Ibid*

⁴ http://www.sims.berkeley.edu/academics/courses/is290-4/s03/readings/Usable_Requirements_Habits_of_Effective_Analysts.htm

⁵ <http://www.phptr.com/articles/article.asp?p=26397&r=1>



Who's behind the research?

Our lead researcher, Kirk Mousley, PhD received BS and MS degrees in Electrical Engineering from MIT and a PhD in Computer Science from Lehigh University. He has been the President of Mousley Consulting, Inc. since its founding in 1993 and has directed the company's efforts in the areas of clinical database design, data editing/cleaning, document management, and submissions.

Karl Mousley received his BS in Mechanical Engineering from Rose-Hulman Institute of Technology and a MS in Computer Science from Villanova University. He has been a senior member of the technical staff at Mousley Consulting, Inc. since 1993. Among his significant accomplishments are the investigation, evaluation, and implementation of new computer technologies for clinical data management systems and developing strategic plans for integrating these technologies into current systems. He has extensive experience preparing Standard Operating Procedures (SOPs).



EDC Today and EDC In Depth

EDC Management publishes well-researched, timely information about EDC technologies and processes.

EDC Today is a free electronic technical bulletin.

Each month we examine topic areas related to Electronic Clinical Systems (ECS) theory, technology, practice, or implementation.

Each *EDC In Depth* research report comes with an executive summary and may be purchased individually for \$395 or as a group of related reports for \$975. Available via downloadable electronic version or paper version sent via mail.

To subscribe to ***EDC Today*** or purchase a specific ***EDC In Depth*** research report:

Order online at
www.edcmanagement.com

Email us at
info@edcmanagement.com

Call us at
1-484-530-0300