PharmaSUG 2019 - Paper LD-296

Time to COMPARE Programmer to Analyst: Examine the Differences and Decide Best Path Forward

Ginger Barlow and Carol Matthews, United Biosource LLC

ABSTRACT

There are almost as many titles for SAS® programmers in the pharmaceutical industry as there are programmers: Statistical Programmer, Clinical Programmer, Scientific Data Analyst, SDTM Implementer, Study Lead Programmer, and many more. Regardless of the label, one thing all of these roles have in common is that these people are writing code to turn data into knowledge. In contrast, the same title in different organizations can have very different expectations on how much any given programmer (or analyst) is expected to contribute to the overall scientific process. Programmers and analysts actually have very different, distinct roles. We will explore what makes each unique, how to know which one you are, how to know which one you may be interviewing, and what the future could hold for each. We will conclude with a discussion on strategies for developing programming teams to meet business quality and financial objectives.

INTRODUCTION

When most people start as programmers in the pharmaceutical industry, they are focused on learning how best to apply their programming language of choice (SAS, R, etc.) to the task at hand. In most cases, people just starting out don't even realize that there *are* different titles, much beyond Programmer and Senior Programmer, or that there are different expectations that accompany those titles. In many cases, companies do not distinguish nuances between a Senior Statistical Programmer and Senior Statistical Analyst. Regardless of what titles are or are not available within any given organization that you choose to work in, it is beneficial to every programmer and manager within the programming field to understand the differences between a "programmer" and an "analyst" so each can plan careers and workload accordingly.

DEFINITIONS AND DISTINCTIONS

So, what is the difference between a programmer and an analyst? What do they do differently? What work do they produce? What value does each add to their project teams?

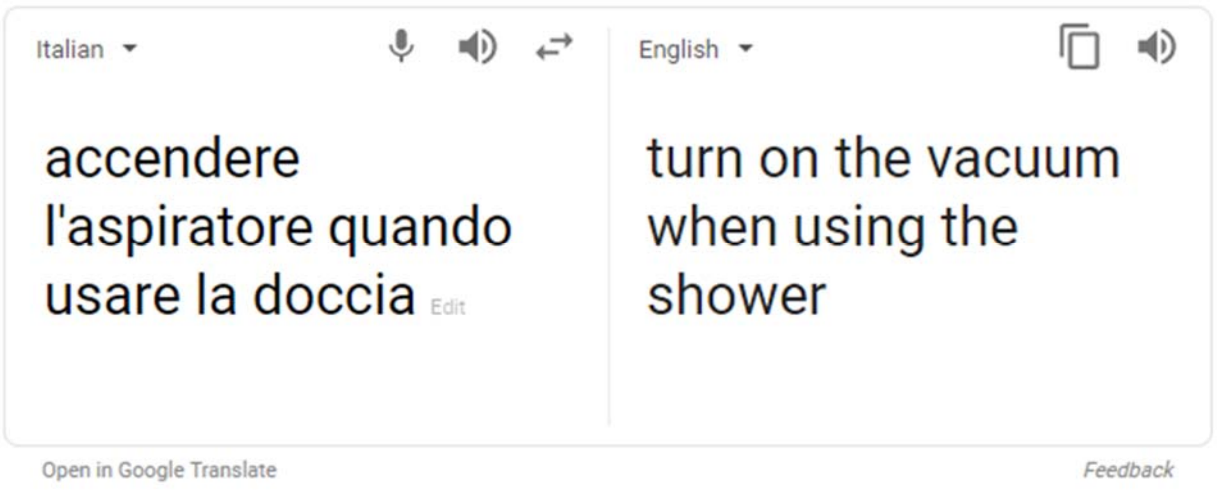
Programmer (noun): a person who writes code to have a computer perform a function, do calculations, produce a report, etc. Programmers receive detailed specifications (aka specs) and translate them into code the computer can understand. They will create a dataset or table that follows those specs and satisfies the reporting requirements.

Analyst (noun): a person who examines the assignment carefully and in detail to identify causes, key factors, possible results, etc. An analyst looks at what the request is, determines what question the request is designed to answer, authors or reviews the specs to answer those questions and then may write code too.

Both programmers and analysts can translate requirements into code, but an analyst is asked to carefully look at the requirements (either writing them herself or critically reviewing ones written by others), to understand the goal of the final product, and to determine whether the spec is appropriate and fits the overall purpose.

In simpler terms, the analyst develops/challenges a specification to be consumed by a programmer – the programmer produces code to be consumed by the computer.

To help understand the differences between programmers and analysts, let's look at a simple example. While travelling in Italy, both programmers and analysts might use Google Translate to interpret a sign outside a bathroom shower:



accendere l'aspiratore quando usare la doccia

turn on the vacuum when using the shower

A programmer would follow the instructions, find their vacuum cleaner, turn it on before getting into the shower and pat themselves on the back for being such a savvy traveler. An analyst would look at the translation, look at the context and consider whether the translation makes sense. Why would I turn on my vacuum cleaner? Is there something that should be different? They might ask an Italian friend what the sign really means or consult their travel guide for common phrases. Then they would turn on the exhaust fan in the bathroom and enjoy a long hot shower.

Although the translation is accurate following syntax and language rules (specifications) the intended meaning of the phrase is actually “Turn on the exhaust fan when using the shower.” In the realm of creating datasets and summary tables for clinical trials, the programmer would follow the specifications and create an adverse event table that satisfies those specifications and presents an accurate view of the data. In the same situation, the analyst would question whether the specifications make sense in the context of the current study and follow up to get those questions answered before finalizing specifications for themselves or a programmer to code from.

DEVELOPING YOURSELF AS A PROGRAMMER OR ANALYST

As a programmer, once you understand the differences between the role of programmer and analyst, it is ultimately up to you to decide what path to take your career. As with any skill, broadening your experience and abilities takes planning and work. Many of the skills required to be a good analyst are the same as those required to be a good programmer.

The path to becoming both a good programmer and good analyst starts with realizing that there is more to programming than just following specifications. You need to understand what you’re doing *technically* (e.g. how make SAS join two data sets appropriately), but also *holistically* (e.g. how joining these two data sets makes sense in the context of this study and how it is summarized). Whether you are mapping raw data files into SDTM data sets or programming complex efficacy figures, it is important to understand the study in which the data you are working with was collected. Once you understand the framework of the study, take note of what data you’re working with and how well it fits the specifications… or not.

The second step in the development of both programmers and analysts involves getting comfortable asking questions – beyond “how do I get SAS to create the output they want me to produce?” Get in the habit of looking for patterns and noticing when something doesn’t fit. More importantly, make sure you ask questions when you see anything that seems out of the ordinary or not quite in line with the specifications. Do the specs make sense in relation to the data and what the team wants to say about that data?

Programmers and Analysts both need to ask questions like…

- Why are we collecting this information? Does the data align with the purpose and the specs? If not, who do I ask – the data manager, the statistician or someone else?

- How is the data expected to look? Skip patterns, expected values/range of values, character vs numeric data (e.g. Are numbers stored in character fields? Do they need to be converted to numeric for analysis? How do you handle greater/less than signs in otherwise numeric fields?)
- How are the data and tables/listings/figures (TLFs) expected to relate to each other? Should every data point have a place in a table?
- Why are we producing this dataset/table/listing/figure? If a listing or figure should support a table, what do I need to cross-check?
- Why did another programmer approach a program in a particular way? Why are they checking particular data values but not others? Why did they put some code into a macro but not others? What can I learn from what someone else did to enhance my own skills?
- What more can I do? What other data domains could I create? What other types of output could I help produce?

Where Analysts start to differentiate themselves are the "extra" questions they ask that go beyond the basics required to create specific, high quality results.

- Based on my previous knowledge of the "customer," are there ways I could develop my code so that when they ask for subsets or other changes to analysis, they can be easily created?
- Can I develop a global macro to address this task/issue?
- Can I help the statistician develop table/listing mocks?
- What more can I learn? Can I attend team meetings with the project lead (or as the project lead)? Can I ask the Data Manager or Statistician if what I'm seeing in the data or analysis plan is "typical"; if not, what makes this project require something special? How will this impact what/how I program?
- What happens to the TLFs after they leave programming? Is output used as in-text tables or just added as an appendix? If included in-text, is there anything you can change about the output template that will make the programmed tables easier to use?

The next step in moving toward becoming an analyst involves volunteering for a little extra – ask to work on different types of projects, different tasks that you don't "usually" do – and making sure you learn from the experience. Learn what's unique about that project/task and what makes it unique so you can look at other projects critically, with that information as a frame of reference.

Think outside your box – look to learn more about the processes that occur both *before* and *after* your part of the process. Not just programming steps (like data import from other file formats, or preparing files for a submission), but key parts of the data management process as well as how writers typically incorporate output into documents. As you learn about these processes, you may see ways to improve processes that others don't even realize is possible. For example, if writers are including certain programmed tables in text and spending a lot of time reformatting the table, you may see an easy way to make changes to your table program that reduces the time required for reformatting (e.g., change font sizes, include table titles as a row in the table rather than as a page header, create superscript letters/numbers for footnotes per the report guidelines).

Finally, look to others you may work with currently who are senior leaders and objectively look at what makes them an analytical programming project lead instead of a senior programmer leading a study. What non-technical skills do they demonstrate? If there are areas you think you should develop, then ask for help or additional responsibilities that would help develop those areas. Areas to focus on would be:

- Adaptability – does "switching gears" often and quickly stress you out, or excite you?
- Initiative – do you reach out, or wait for others to ask you to step up?
- Professional integrity – dedication to getting things done the "right" way (not taking short-cuts that could undermine the quality of your work)

- External awareness – awareness and understanding of what is going on in the industry overall (e.g. latest regulatory guidelines, software innovations)
- Organizational awareness – awareness and understanding of company and department goals, and how what you do fits into achieving them

Contrary to what many may think, programming in the pharmaceutical industry is one job where the more questions you ask, the smarter people think you are! Make sure you pay attention to the answers/explanations people give, and that you truly understand the reasoning behind the decisions made and processes described. If you are unclear, make sure you ask follow-up questions until you are comfortable with the response. Don't ever assume that your current assignment is exactly like the one before – ask questions to ensure that you are working under the same general framework – it's another adverse event dataset, but is this an oncology study with dosing cycles or a cardiovascular study with one long-term dosing scheme? Is there anything special about certain adverse events? Do you need to calculate onset day relative to one reference point or several? By regularly questioning and absorbing the information around you, you will add valuable knowledge that enhances your skills as either a programmer or analyst.

HOW TO DETERMINE PROGRAMMER OR ANALYST IN AN INTERVIEW

When interviewing for people to add to your project team, people leaders should consider whether programmers or analysts are needed, or some mix of both. Do you have more routine work with well established procedures and standards? You might want programmers who can learn existing systems, produce consistent, quality results and be satisfied with their contributions. If your team needs people who can readily assess what needs to be done and develop novel solutions for unique projects, you may want to focus more on finding analysts.

But how can you determine whether someone is more of a programmer or an analyst? Here are some suggestions for interviewing that might help.

USE BEHAVIORAL QUESTIONS

If you have had any training on interviewing, you have probably heard about using behavioral questions over the more traditional approach. These kinds of questions help you assess past behavior which is usually a good indicator of future behavior. Many of these questions start with "Tell me about a time when…"

- Tell me about a time when you had to analyze information and make a recommendation or suggest a change.
- Have you had a situation when you were trying to communicate a technical idea to a non-technical coworker and they just weren't understanding? What did you do?

Avoid asking leading questions. Instead of saying "Tell me about a time when you had a difficult manager and what you did to make things better", ask "Tell me about a time when you had a difficult manager." This leaves it open for you to see how they dealt with the situation and different methods they tried, rather than indicating that you expect that they did something to improve the situation. If they say things like "I just tried to avoid pushing his buttons" or "I worked more overtime to try to meet their demands", then this individual is likely more suited to a programmer role. These responses show that they may be less likely to take initiative to search for solutions and be more likely to just work harder within existing parameters.

What about the candidate who responds with a list of things they tried (communicating differently, clarifying expectations, asking the manager for key things to focus on) but ends up saying "I tried all those things but eventually decided I wanted to pursue a different opportunity"? These answers will highlight how they tried different methods, how they communicated, how deeply they thought of possible solutions, their problem solving strategies, etc. This is an indication that you are talking to an analyst – this person is an active problem solver.

USE HYPOTHETICAL SCENARIOS

Hypothetical questions are similar to behavioral questions but these questions are more focused on assessing a candidate's thought processes when faced with challenges. An analyst will evaluate risks and options. They will look for what might fail rather than how to get something done. They might take a longer time to answer and ask questions to clarify. In contrast, a programmer might jump right into an answer with more straightforward responses – describing what they will do step by step and you won't hear them stop to say words like "evaluate" or "review". Some examples of hypothetical scenarios are:

- When you are told to copy code from an older study to use for a new one, what do you do? How do you test your work?
- How do you self-validate code?
- How do you know when to solve a problem on your own or to ask for help?
- Pretend I've given you an assignment to validate a dataset I've written. What steps will you take? What is it that you're trying to accomplish?

ASK FOR EXAMPLES

This is a more common interview technique for technical roles, but still can be effective and give you more clues to whether someone has more of a programmer or analyst mindset.

- Ask them to bring code to the interview. Ask them to explain what it does, and why.
- Provide sample data and scenarios and ask them to produce something.
- Provide code and ask them to explain what it does and how to make it better or more efficient.

COACHING PROGRAMMERS WHO WANT TO BECOME ANALYSTS

While it is up to an individual to decide where they want their career path to lead, it is a manager's responsibility to find out where their staff want their career path to lead, weave it into their department's objectives, and create a plan to help both the individual and the company achieve their goals.

As noted above, there are many aspects to both the programmer and analyst career path that overlap. As a manager, the key to helping both develop is to allow both to feel empowered to pursue their chosen path in a direction that both suits the individual and aligns with company goals. Both programmers and analysts should understand the impact of their work, feel appreciated and take ownership of their contributions to the overall organization.

An atmosphere of empowerment **"**is achieved by encouraging creativity, individuality, problem solving, and an open and honest exchange of ideas among all employees in a non-threatening environment." (1)

How can you, as a manager, build up a team of empowered programmers and analysts within your department?

- Cultivate a culture of ownership. Build teams that both support each other and "call each other out" respectfully when problems arise
- Encourage programmers to be ready to justify what they have done and the thinking process behind their decision. If they followed a line of logic and took a wrong turn somewhere, they can learn from that mistake. Be willing to listen when they explain why the mistake happened, so you both can learn what could have been done differently. Then build in processes that reduce the risk of recurrence.
- Abandon the "command and control" style of leadership. You are limited to your knowledge, your experience and your intellectual capabilities. Listen when others contribute.
- Set a goal to reserve a percentage of your time to developing staff – make it one of your objectives as a people leader.
- Encourage senior technical programmers/analysts to mentor more junior programmers.

In this environment, individuals can work to develop distinct skill sets that interest them most. While programmers may want to focus on becoming the subject matter expert in one particular area, analysts may be more interested in working on a wide variety of projects where their innate sense of curiosity, self-motivation and creativity can be developed. As a people leader, it is important to realize that both are valuable and work to ensure all members of your team know they are valued.

In most cases, development of analytical skills takes longer and involves more diverse skill sets beyond those required to be a good programmer. How can you, as a manager, help someone move from being a good programmer to being a strong analyst who contributes value to his project team?

- Clarify expectations for individual performance. Communicating expectations is the same as developing specs for a dataset – the more specific and detailed they are, the more likely they are to be successful.
- Be specific on what types of behaviors an analyst is expected to show. Identify areas that might need more focus, like communicating technical ideas to non-technical team members from other departments.
- Include those who are interested in developing their analytical skills in project meetings they may not normally attend in their current role, along with the analyst assigned to the project so the trainee can observe and learn expected behaviors.
- Encourage safe failure. Give someone the opportunity to try something new, even if you don't think it will be successful. Regardless of success or failure, discuss the result in detail so it is a positive learning experience.

ROLES FOR EVERYONE IN TODAYS MARKETPLACE

While it may seem that a company's goal would be to have a department full of analysts, in reality there is a place for both good programmers and good analysts. Just as there is a need for individuals who look at the "big picture" and tackle unusual tasks, the there is also a real need for "heads down" programmers who enjoy working through the everyday tasks of generating output. An optimally functioning programming department would ideally have a mix of programmers and analysts where the ratio of one to the other would vary depending on the workload and type of project work being done.

Programmers	Analysts
Attributes	
Tend to be most comfortable working with explicit specifications, discussing items needing clarification with the statistician or project lead and working on teams with well-defined processes and guidelines	Often enjoy working with loose requirements and actively collaborating with the overall project team to refine specifications as they go.
Usually prefer to work on and be proficient at common/standard tasks; may get stressed if asked to switch projects often or work on very unusual tasks.	Tend to prefer working on one-off, unusual tasks or being involved with planning for a new project or something novel; may become bored with "day to day" tasks, lose focus, and become prone to errors.
Typical Tasks	
Creation of SDTM safety dataset specifications (define files); generation of these datasets; programming custom domains from well-defined specs	Creation of efficacy and custom SDTM dataset specifications; generation of these datasets; reviewing to ensure all data is mapped into SDTM datasets

Programmers	Analysts
Creation of standard ADaM safety domain dataset specifications; generation of these datasets	Creation of efficacy and custom ADaM dataset specifications; generation of these datasets
Develop programs for data import/export and rerun as needed throughout the course of the study	Look for efficiencies and build code libraries / macro systems that can be used across studies or globally
Subject Matter Expert (SME) on specific data domains	Risk Evaluation and Mitigation Strategy (REMS) metrics – develop table specifications and execute them
Generating TLFs from well-defined specifications, with an eye for consistency and fit to the actual data	Other “RWE” projects where the study design and/or data sources are *not* clinical trials, and/or tend to be “one-off”

Everyone starts as a programmer – learning how to make your software of choice do what is required to execute the tasks you’ve been assigned. As you work to master your chosen language, you will likely get the opportunity to observe a variety of tasks and roles within your department. You will eventually be given more challenging assignments and you need to pay attention to what aspects of your job you enjoy, and those that you may find to be a chore. Some may enjoy the challenge of figuring out how to fit data into SDTM structures, mastering define files and creating datasets that fit those specifications. They enjoy mastering the complexities of the CDISC guidelines and are most happy when working in that space. Others may find that same work frustrating and tiresome, preferring the challenge of working on projects that fall outside the purview of CDISC. To them, the challenge of designing datasets without specific guidelines, where the custom data structures simply need to enable the most efficient generation of summary tables, is the most rewarding work.

The reality is, there’s a little analyst in every good programmer and vice versa. The key is to decide which you truly prefer, and work with your management to tailor your work assignments as much as possible to allow you to grow into the best possible programmer or analyst that you can be.

CONCLUSION

It is important for Biometrics professionals in the pharmaceutical industry to understand the difference between programmers and analysists. As an individual contributor, understand what parts of your job you enjoy most and work to become the best you can be at those tasks. Evaluate what areas you need to grow in and develop a plan to improve your skills. As a people leader, clarify what your department goals are and create a team that blends programmers and analysts in the right proportion to meet your business objectives. If you are at a large company with well-developed macro systems, you may need a higher ratio of programmers to analysts – people who will learn those systems and be satisfied working in that environment with a focus on quality and consistency. If your department works primarily with post-marketing data and/or studies that are “one off” then you’ll likely want a higher ratio of analysts – those people who enjoy tackling new challenges on a regular basis and can quickly adapt to new situations. In either case, understanding the differences between programmers and analysts, and maximizing that blend on your team is the key to achieving your individual, team, department and company goals.

REFERENCES

(1) Michaels, Sharon A. “Empowerment.” Reference for Business. Available at https://www.referenceforbusiness.com/encyclopedia/Eco-Ent/Empowerment.html .

CONTACT INFORMATION

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

Ginger Barlow
267.470.1294
Ginger.Barlow@UBC.com

Carol Matthews
215.390.2236
Carol.Matthews@UBC.com