Paper LI08

How Agile can you be –
Looking at statistical programming as a software development process

Oliver Wirtz, Bayer AG, Wuppertal, Germany
Peter Bonata, Bayer AG, Wuppertal, Germany

ABSTRACT
Agile methods and practices have been around in software development for more than 20 years. Surprisingly, agile principles have not found their way into day-to-day statistical programming work. Instead, statistical programming, especially in clinical studies, is still very much relying on planning work towards fixed milestones, following finalized specifications. This paper will challenge this common practice and outline how an agile mindset can be established in teams using software development tools such as Azure DevOps® and thus emphasizing team collaboration, incremental delivery, and continuous planning and learning with built-in quality.

INTRODUCTION
Statistical analysts across the industry encounter a myriad of challenges that can vary significantly based on organizational structure, team dynamics, and geographic considerations. Whether working within a country team or across different time zones, and utilizing tools such as SAS, R, or others, these analysts must navigate a complex landscape. The nature of clinical trials adds another layer of complexity. Analysts may find themselves managing highly standardized Phase 1 trials that are short in duration, or more intricate multi-center Phase 3 trials that involve multiple interim analyses over several years and simultaneous submissions to various regulatory agencies. Furthermore, the demands of various stakeholders outside the core analysis team can complicate matters, as they often seek regular updates and draft output documents. Teams frequently face the pressure of committing to timelines that may prove challenging to meet.

In the following sections, we will revisit the common clinical trial process and draw parallels between clinical trial analysis and software development methodologies. We will demonstrate how Agile methodology can facilitate the division of the overall work package into manageable increments, ultimately benefiting the study team as a whole. Finally, we will outline practical strategies for implementing these new practices in the day-to-day operations of analysis teams.

REVISITING THE CLINICAL TRIAL ANALYSIS PROCESS
Clinical studies typically adhere to a defined roadmap, where specific events are sequenced in a logical order. For instance, the study protocol outlines the statistical analysis plan (SAP), which serves as a foundational document guiding the analysis process. The tables, figures, and listings (TLF) are specified based on the SAP, detailing both the content and layout of the analysis datasets (ADaM). Ideally, these ADaM datasets should facilitate the derivation of TLFs with minimal effort - “one proc away.” When TLF have been delivered initially after database lock (DBL) there are additional documents such as intext tables or narratives that are based on TLF and/or finalized ADaM datasets.

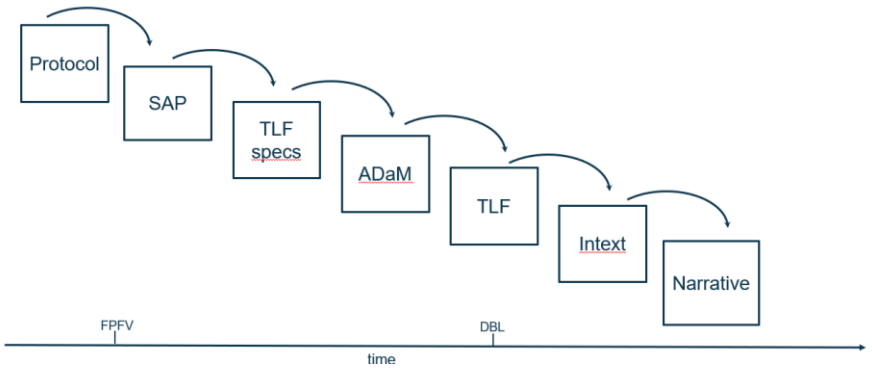

Figure 1: Example of sequential availability of study deliverables

The sequential nature of documents and deliverables is also reflected in standard operation procedures (SOP) which outline milestones during a trial, specifying when deliverables must be finalized. However, despite these structured dependencies, the reality is often far more intricate. Throughout the course of a trial, it is common for the protocol,

SAP, and TLF specifications to be revised. These changes can vary significantly, ranging from adapting to issues in source data to major protocol amendments that fundamentally alter the scope of the analysis, thereby impacting both SAP and TLF.
Unfortunately, SOP typically indicate when deliverables should be finalized and, even if subtle, suggest that modifying finalized deliverables is uncommon and should generally be avoided. This emphasis on milestones heavily influences how study teams approach their work, leading to a perception that change represents a setback. Consequently, teams may view modifications as indicative of poor planning rather than as necessary adaptations to evolving circumstances.

EMBRACE CHANGE AS INTEGRAL PART OF THE PROCESS

Whatever the reason is for changing protocol, SAP, analysis datasets or any other deliverables throughout the study lifecycle, the entire study team - including the analyst team – must be prepared to implement necessary edits. Organization must recognize that change can occur at any stage of the trial, regardless of its current status. To address change effectively, teams need to shift their perspective. Instead of solely measuring success by reaching sequential milestones, they should focus on how to work on deliverables in iterative cycles. This approach fosters adaptability and allows teams to respond more effectively to evolving requirements.

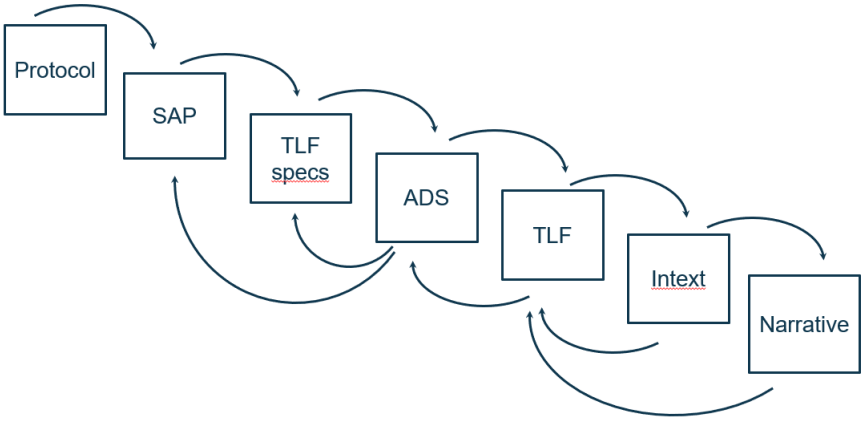

Figure 2 Iterative-incremental development of deliverables

The challenge of adapting to change is not unique to the pharmaceutical industry. In fact, this issue has been recognized for decades in software development, leading to the emergence of Agile methodology. This methodology has, in turn, given rise to a comprehensive set of tools and behaviors, such as DevOps and SCRUM, designed to help teams manage their work more effectively. Statistical programming and its stakeholders can particularly benefit from adopting these proven working practices.

EXTEND THE SCOPE TO INTERACTION WITH STAKEHOLDERS OUTSIDE STATISTICAL PROGRAMMING

Traditionally, statistical programming occurred behind the scenes, often obscured from stakeholders such as clinicians and medical writers. It was only during the database lock that programming efforts came into the spotlight—typically when delays arose. Overall, (SAS) programming carried an air of secrecy, with statisticians and programmers holding exclusive control over the data and possessing the specialized tools necessary for analysis and visualization.
However, this paradigm has shifted significantly in recent years. The emergence of low-code and no-code tools has made it possible for a broader range of stakeholders to access and interact with clinical databases, fostering greater collaboration and transparency. These tools have allowed individuals outside the statistical programming realm to gain insight into the day-to-day challenges faced by programmers, increasing the visibility of their work in the process. As a result, statistical programming is now better understood by non-programmers. Given this progress, why not capitalize on these advancements and involve stakeholders in the development process on a regular basis?
At first glance, this idea may seem counterintuitive, as we are accustomed to sharing study results only at specific milestones, such as database lock or draft TLFs. However, limiting updates to a few defined milestones carries significant risks. When results are shared only at these points, the resulting package can become quite large, reflecting weeks or even months of work. Managing such extensive packages can be overwhelming: developers must manage a substantial amount of information, while reviewers face the daunting task of digesting it all in a relatively short timeframe. This situation is stressful for both parties, often leading to frustration, especially when time is tight and systematic issues arise that could have been resolved much earlier in the process.

CHANGE OF MINDSET TO ACTIVELY ENCORAGE AGILE BEHAVIOUR(S)

While acknowledging that documentation is subject to change and that work results can be exchanged with stakeholders on an ongoing basis, let's redefine the pharmaceutical programming process. We should recognize

statistical programming work as an ongoing interaction between development and operations and organize this interaction using DevOps and Agile tools.

In software development, DevOps is a cultural and technical movement aimed at improving collaboration between development (Dev) and operations (Ops) teams. When we translate development into the statistical programming team and see medical writers, medics, statisticians and others as operations team, the goal of collaboration remains the same in a clinical trial context. The product, however, shifts from a software application to deliverables such as tables, listings, figures. Despite this change in product, the approach to development remains consistent. The collaboration between stakeholders and the development team follows a predefined series of phases that the team navigates through:

PLAN
In the planning phase, the team collaboratively discusses the upcoming deliverable. With all key stakeholders involved, it becomes much easier to manage expectations and address challenges or known issues before the actual work begins.

DEVELOP
The development phase encompasses the actual statistical programming work, built upon clearer and clarified specifications. The team iterates in time-boxed increments, where in-built feedback loops ensure better quality throughout the process (more about this in the next chapter)

DELIVER
The development team will deliver the agreed-upon increment to the stakeholders. Since this deliverable represents only a portion of the overall package, it is much easier and faster to review.

OPERATE
Stakeholders will review the increment and provide feedback as needed. Features discussed during the planning phase and included in the deliverable can be marked off the list. Any new or unresolved issues can be addressed in the next iteration.

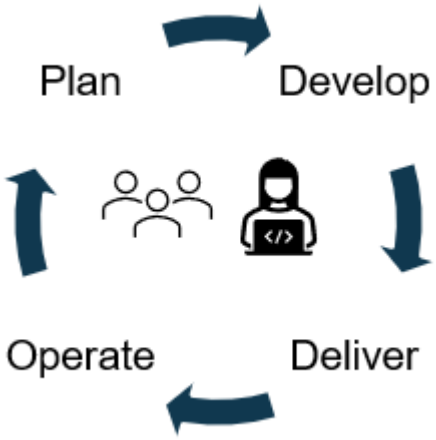

Figure 3 DevOps in a simplified TLF lifecycle

In essence, utilizing DevOps means that teams are encouraged to collaborate more closely and, importantly, more frequently than in traditional working models. This approach helps to foster team spirit, as everyone on the team participates in planning the work and is more connected to where the actual work happens. Stakeholders transition from only receiving a finished product to actively participating in shaping the product they expect to see on an ongoing basis. Dividing the process into multiple rounds or iterations also allows for continuous refinement of the product, enabling stakeholders to see results evolve much sooner.
On the other hand, developers will have the opportunity to discuss open questions directly with stakeholders during planning sessions. The team either reaches a solution, or if there is no consensus on how to address pending issues, they can determine which points can be tackled in the current iteration and which need to be deferred. The key point here is that the team strives to resolve all issues that can be addressed with the current knowledge as early as possible, even if some issues remain unresolved. This approach makes a significant difference, as waiting for all issues to have a solution could lead to unnecessary procrastination. It also forces everybody on the team to re-think current approaches early, when there is still time to change direction and/or handle exceptions in the documentation.
This leads to perhaps the greatest advantage a team can have when starting early: having the time to fail. While this may sound trivial, it is crucial. Only when a team can experiment with new ideas without the fear of failure will they be willing to challenge the status quo and introduce innovative concepts.

ORGANIZE CHANGE

The previous sections highlight that change is inherent in the clinical trial process. Teams that embrace change can turn necessity into opportunity by leveraging DevOps and Agile methodologies. However, leadership must facilitate this transition by fostering an environment conducive to new behaviors.

Firstly, it is beneficial for the leadership team to reach a consensus on adopting agile practices without exception. It should be clear that this is the direction the organization is going and that teams will see full support from leadership. Ideally, standard operating procedures (SOPs) related to clinical trial analytics should be thoroughly reviewed to identify and address gaps that hinder time-boxed increments. This approach sets the foundation to ensure teams can transition safely to working adaptively. Let´s assume these external factors have been sorted out and teams are ready to working in agile teams – how can we instill changing mindset towards an iterative/incremental working style?

YOU´RE NOT STARTING FROM SCRATCH

The good news is that much of what is described in Agile and DevOps is already happening in projects that run well. The team may not be aware, but chances are high that they already practice an iterative/incremental work style, e.g.

- Everybody on the team knows their role
- The team has established an efficient communication style, off record, as needed
- The team can react fast to changes
- The team is self-organizing and works cross-functional
- The team front loads tasks
- The team provides regular updates to stakeholders
- The team pro-actively tries out new/alternative ways to solve issues and is innovative
- …and many more

So, in general successful team already know how to efficiently run a trial and it is straight forward to introduce the overall concept of an iterative-incremental workflow such as:

1. break down overall workload into work items/tasks
2. discuss in team which items can be completed with the available information
3. divide overall time until final deliverable into smaller intervals of e.g. 2 weeks
4. commit to work items that can be completed during the time interval
5. (discuss prioritization, if needed)
6. after two weeks share the outcome with the team
7. accepted outcomes are seen as completed
8. changes and comments are addressed in next iteration
9. repeat!

The list of tasks above describes a complete cycle the team goes through during a time-boxed iteration and many of the items may look familiar – even without any knowledge about Agile and are also quite common in software development. In order to handle these recurring tasks in a structured way, Scrum was built as a framework based on Agile methodology. The basic idea of the Scrum framework is "*to get work done as a team in small pieces at a time, with continuous experimentation and feedback loops along the way to learn and improve as you go*"[1]

Although Scrum was originally developed for software development, it has increasingly been adopted in many fields that involve complex processes. Scrum defines roles (such as Scrum Master, Product Owner, and Development Team), events (like Sprints, Sprint Planning, Daily Stand-ups, and Sprint Reviews), and artifacts (such as the Product Backlog and Sprint Backlog) that guide teams in delivering work incrementally. The following schema from the official Scrum homepage shows an overview of the Scrum framework.

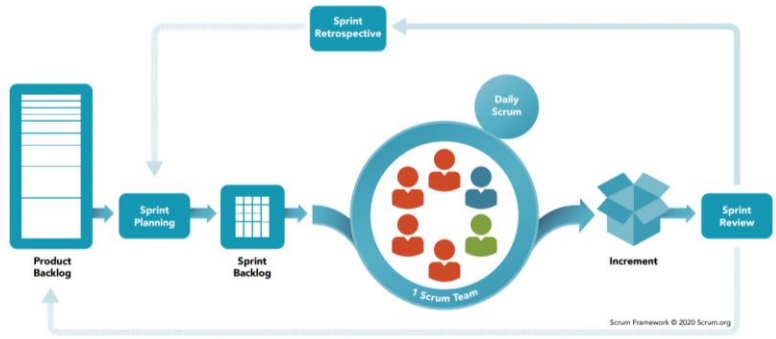

Figure 4:Scrum framework, an iterative-incrementatl workflow

However, directly translating the Scrum framework into the realm of statistical programming can present challenges due to Scrum's rigid structure regarding roles, responsibilities, and the timing of various events. The specificity of Scrum can sometimes feel constraining when applied to the more fluid nature of statistical programming tasks. Fortunately, it is not essential to fully grasp every aspect of Scrum to leverage its technical infrastructure effectively.

For our purposes, it is entirely sufficient to adopt the relevant terminology and tools associated with Scrum while maintaining flexibility in how we implement these tools within the context of statistical programming. This approach allows teams to benefit from Scrum's organizational advantages without being bound by its structures.

Let's consider how we can apply the Scrum framework to manage a clinical trial, exploring how the elements depicted in the figure above can be aligned with the various phases and activities involved in trial conduct. By doing so, we can create a "tailored version" of Scrum that meets the unique demands of statistical programming in clinical research.

PRODUCT BACKLOG

The Product Backlog serves as a comprehensive repository that encompasses all programs e.g. required to generate the Tables, Listings, and Figures (TLF) needed for the clinical study. This dynamic log evolves continuously as the project progresses. Any modifications—such as the addition of new tables, the introduction of new parameters, or changes to existing algorithms—can be seamlessly integrated into the backlog. This is achieved either by adjusting existing backlog items or by adding new items, as well as removing those that are no longer relevant.

As the single source of truth for the project, the Product Backlog ensures that all team members have access to the most up-to-date information regarding project requirements and priorities. It fosters transparency and alignment within the team, as everyone can refer to this centralized document to track progress and understand the current focus.

In comparison to traditional tracking documentation such as Excel trackers, the Product Backlog functions similarly but offers enhanced capabilities. Unlike static spreadsheets, the backlog provides a more interactive and flexible framework that allows for real-time updates and prioritization, team discussion and more, thereby facilitating better project management and collaboration.

SPRINT BACKLOG

During a sprint planning meeting, the programming team collaboratively reviews the backlog to determine which items or tasks they can realistically complete within the upcoming sprint, typically lasting two weeks. This meeting is not just a routine check-in; it is a critical opportunity for the development team to engage with other stakeholders, such as statisticians or even medics or medical writers, to align on priorities and expectations.

Together, the team assesses backlog items, considering factors such as complexity, dependencies, priority, and resource availability. This collaborative effort leads to the creation of a Sprint Backlog, which outlines the specific tasks to be undertaken during the sprint. The team discusses any pending issues, potential analysis strategies, and clarifies any uncertainties regarding the tasks at hand.

By the end of the sprint planning meeting, the Sprint Backlog serves as a comprehensive plan for the next two weeks, ensuring that all team members share a mutual understanding of their objectives and responsibilities. This alignment enhances team cohesion and sets a clear direction, enabling the team to work efficiently and effectively toward their goals.

DAILY SCRUM/DAILY STANDUP

Once the team has identified and prioritized the backlog items, they begin working on these tasks collaboratively. Depending on the study phase, it can be highly advantageous to hold a daily stand-up meeting. During this brief, focused gathering, programmers and team members convene to discuss their progress. Each participant shares what they accomplished the previous day, outlines their goals for the current day, and identifies any impediments they may be facing.

Daily stand-ups serve as a crucial communication tool, fostering transparency and accountability within the team. While some may mistakenly perceive these meetings as a form of micro-management, they are, in fact, an effective strategy for the lead analyst to monitor the trial's progress in real-time. By facilitating open dialogue, daily stand-ups enable the team to swiftly address challenges, adjust priorities, and maintain momentum, ultimately contributing to the successful execution of the clinical trial.

INCREMENT

The outcome of the sprint is the sprint increment, which in our example can be subset of TLF coming as result from the agreed list of backlog items. It is also a measurable outcome of the team´s effort and can now be shared with other stakeholders, if applicable. In a clinical trial setting this marks the point where an intermediate product can be sent out for review. Since it´s a small subset of the overall package, it can be reviewed faster and more thoroughly by both the development team and external stakeholders. Reviewers can give their feedback and either accepting the increment as it is or ask for changes that can lead to updates in the backlog and changes will be done in a later sprint. Overall, these reviews enable the development team to review a smaller package before it´s sent out within their team and use this as an additional layer of QC before the deliverable is shared with others.

SPRINT REVIEW

As previously mentioned, the sprint review serves as a vital platform for exchanging results with stakeholders on an ongoing basis. This meeting not only facilitates communication but also acts as an essential quality control (QC) loop, allowing members of the development team to present their work from the past few weeks. But there is even more use of the sprint review: junior programmers have the opportunity to show their progress, demonstrating what they have learned. It´s an excellent opportunity to show new analysis strategies or innovative algorithms.
This collaborative environment encourages the sharing of insights and experiences, enabling team members to discuss challenges they faced and solutions they implemented. Overall, the sprint review provides a structured opportunity for the team to reflect on their accomplishments and lessons learned from the previous sprint.

SPRINT RETROSPECTIVE

The sprint retrospective in essence is meant give the team the chance to reflect overall about how the recent sprint(s) went, if the team needs to improve or, also to celebrate success. A significant benefit of these built-in feedback loops is that they help institutionalize a culture of continuous feedback and learning, which can often be challenging to incorporate into the daily workflow. By making feedback a regular part of the process, the team fosters an atmosphere of growth and improvement, ultimately enhancing both individual and collective performance.

DISCUSSION

The Scrum toolbox offers a wealth of resources and may initially appear complex to those unfamiliar with its intricacies. However, the core principles of Scrum—such as maintaining a backlog, organizing work into sprints, executing tasks, and delivering incremental results—should be easily grasped by statistical programming teams, even if the terminology is new to them. Setting aside additional practices like daily standups, it is important to note that a typical clinical trial is already analyzed in a (very short) series of sprints, where these sprints extend much longer than desirable. When teams recognize that adopting shorter, more frequent sprints, coupled with established feedback loops, can significantly enhance their current workflows, this realization will rapidly translate into their day-to-day operations. As teams integrate these practices into their daily routines, they will likely notice increased collaboration, improved communication, and a greater ability to respond to changes or obstacles. It is, however, also essential to recognize that an implementation, e.g. designed for long-running Phase 3 studies, may have different requirements compared to a more ambitious Phase 1 development project that spans only a few months. Ultimately, the creativity of team members, along with their unique team dynamics, will shape the implementation of Agile practices by selecting the most appropriate tools from the array of options available.

CONCLUSION

Agile methodologies, DevOps practices, and the Scrum framework are well-established concepts in software development that have been meticulously refined over the years. When it comes to project management, the challenges faced in software development share many similarities with those encountered in clinical trials. Although the final products may differ—software applications versus clinical study outcomes—both fields rely heavily on specification documentation that is subject to change and must adapt swiftly to evolving requirements. Furthermore, statistical programming involves skill sets that are comparable to those in IT, making it easier to translate roles, responsibilities, and methodologies from a software development context to a clinical trial setting. There is a plethora of free resources available, with a proven track record, to assist organizations in taking their initial steps toward adopting Agile work behaviors.
With the rise of cloud-based Agile project management tools, such as Azure DevOps®, it has become increasingly straightforward to kickstart Agile practices with a comprehensive suite of software solutions to support these initiatives. Teams will soon discover areas where existing Scrum blueprints need to be adjusted to align with their organizational landscape. Leadership teams should foster an environment that encourages experimentation and learning through trial and error, allowing teams to find the best interpretation of Agile for their specific circumstances.

REFERENCES

[1] What is Scrum? | Scrum.org: https://www.scrum.org/learning-series/what-is-scrum/ last accessed on 26SEP2024

CONTACT INFORMATION (HEADER 1)

Author Name: Oliver Wirtz, Peter Bonata
Company: Bayer AG, Wuppertal, Germany
Address: Aprather Weg 18a, 42113 Wuppertal
Email: oliver.wirtz@bayer.com
Website: bayer.com