PhUSE 2016

Paper PD04

Continuous Improvement for TFL production

Caroline Terrill, CROS NT, Maidenhead, UK

ABSTRACT

A process was established to implement the ethos of continuous improvement within CROS NT. As a Contract Research Organization focused on data and analytics, the 5-step procedure was created so that it could be implemented across all aspects of the company, regardless of subject matter. Statistical programming was selected as one of the departments to pilot this continuous improvement process.

Programming tables was identified as the process that had the highest impact in terms of timelines and budget across the performance measures and therefore gave the biggest potential for meaningful improvement. We gave ourselves a target of 15% reduction in hours for table production.

In this paper I will explain the continuous improvement process with reference to how it worked within the programming department. I will go through the five steps to show the methodology and discuss the outcomes so far for this project.

INTRODUCTION

If a company isn't proactive, it will lose its competitive edge; improvements must be made to keep pace and stay in business. However, change can be time intensive to implement, and can cause disruption with employees, processes and projects. By implementing a process of continuous improvement, changes are made gradually and are always ongoing, making change and improvement part of the culture of the company. This also allows breakthrough changes to happen more spontaneously and be more widely accepted.

CONTINUOUS IMPROVEMENT PROCESS

Therefore, CROS NT implemented an action plan for a continuous development process within its operational departments. The 5-step procedure is adaptable to all aspects of the company, regardless of department or function.

The steps are:

- Determine current performance
- Define target objective
- Research current performance
- Identify solutions
- Implement and maintain the change

Statistical programming was selected as one of the departments to test this continuous improvement process and in this paper I will explain the process with reference to how it worked within the programming department.

DETERMINING CURRENT PERFORMANCE

Determining current performance involved looking at all the possible processes that are governed by our SOPs and deciding the most relevant process to start with. Improvement is a result that must be measured and therefore baselines based on current performance are key to ensuring the program of improvement is a success.

The following table shows an assessment of the impact the main programming tasks have on the business.

Performance measure	Process	Business Critical	High Cost Impact	High Time Impact	High Delivery Impact
Baseline cost estimates	Table Programming	High	High	High	High
	Listing Programming	Medium	Medium	Medium	High
	Figure Programming	Medium	Medium	Medium	High
	Derived dataset programming & Validation	High	High	High	Medium
	SDTM programming	High	High	High	Medium
	ADaM programming	High	High	High	Medium
	SDTM / ADaM documentation	Medium	Medium	Medium	Low
	Output delivery	High	Medium	Medium	High
Performance KPIs	Table Programming	High	High	High	High
	Derived dataset programming & Validation	High	High	High	Medium
Customer feedback	Output delivery	High	Medium	Medium	High
Employee feedback	Table Programming	High	High	High	High
	Figure Programming	Medium	Medium	Medium	High

Looking at the different performance measures that are used within CROS NT, table programming was the process with the highest potential impact across the categories assessed and therefore gave the biggest potential for meaningful improvement. Table programming was selected as the primary process for improvement, but if any updates could be made to the general programming process that could also have a positive impact on the programming of figures and listings, they were also to be considered.

DEFINE TARGET OBJECTIVE

To be able to determine if the improvement was successful, we needed to define the target objective. Potential targets are:

- Hours reduction
- Cost reduction
- Profit improvement
- Quicker delivery

The target chosen was hours reduction in table programming. This was the obvious choice of target because of the high impact of hours on the baseline cost estimates and KPIs. It was also the easiest to monitor on a regular basis and this was important in terms of the progress and perceived success of this project. In addition, if table programming hours can be reduced, this could lead to quicker delivery times, which is always important for the client. We gave ourselves a target of 15% reduction in hours by the end of 2016. The goal was to achieve this reduction in hours without changing other factors such as resources, and therefore we could not just simply select more experienced programmers.

To determine if we had achieved our objective, we needed to have a method of monitoring the number of hours per project on a regular basis. We used a proprietary enterprise resource planning system of our partner technology company to track this. This tool allowed us to monitor resources spent for each activity in the statistical programming function – in terms of hours and costs. As part of the statistical programming KPIs, the time needed to program tables (and listings and figures) is reviewed on a monthly basis to ensure any potential issues are highlighted and investigated. There is a large amount of variability in the hours needed for table programming by project, due to the complexity of the analysis, table shells and underlying data. In addition, different programmers at different seniority levels would have been used in previous projects and so we couldn't just use data from one particular project. Therefore, we decided to create a six-month moving average of the time needed to program a table. We realized that this would introduce a time delay in being able to see any improvement, but it was felt that this was the most consistent approach in the long term. In order to determine if the target would be achieved by the end of 2016, the average time needed to program a table for each project would be reviewed against the average for the past six months.

RESEARCH CURRENT PERFORMANCE

To determine the current performance we needed to break down the table programming process into each individual step that is required. This needed to include any preparation activities, waiting time or movement of items within the process and deliverables produced. For table programming we included the following activities:

Create utility programs in the study folder:

- Copy general programs from program store to study folder
- Update program header for all programs
- Update directory setting program
- Create required formats

Create general macros for the study:

- Copy general macros from macro store to study folder
- Update program header for all programs
- Update general macros as needed for study
- Create new macros as needed for outputs required

Create general table programs for study:

- Copy general table programs from program store to study folder
- Update program header for all programs
- Update general table programs to be study specific
- Create new table programs as needed

Produce production output

- Update production program for all outputs required in first draft
- Run production program ensuring copies of relevant programs, outputs and logs are saved if to be delivered to client

While producing the step-by-step process, we needed to determine if there were any inefficiencies that can be identified in the following categories: over production, inventory, motion, transportation, waiting, rework, over processing and talent.

Inefficiencies in over production can happen if there are any tasks that we are completing sooner than needed, producing deliverables faster than needed or producing more deliverables than the customer requires. Generally customers want the outputs as soon as possible, so I don't believe we are producing deliverables faster than needed. We usually assume we are going to provide 2 drafts and 1 final output, while customers for smaller projects might only need 1 draft and 1 final output, however, this would be specified in the contract and accounted for in the hours for the project. No inefficiencies were identified in over production

Inventory inefficiencies can arise if you are gathering more information than needed or producing documents that are not used – for example, if data are collected on the CRF that are not needed in the analysis. For table programming, the statistical analysis plan (SAP) contains the information the programmers need to be able to create the required outputs. Table shells are included in the SAP with additional notes for the programmer as needed, e.g. for sort order. Any additional clarifications are discussed with the statistician / customer and documented in an internal tracking spreadsheet. Although additional documents are created for the specification of datasets, no additional documents are created for table programming and so no inefficiencies were identified for inventory.

Motion inefficiencies are where we have unnecessary movement of people, documents, tasks, information within a particular process. Looking at our process for table programming, we do have motion inefficiencies in that programs and macros are copied from the store to the study folder, those macros and programs are then updated for the requirements of that particular study. The time taken to copy is minimal, but updating the programs and macros for the study can be time-consuming. This can be repetitive and potentially wasted if similar updates are needed for different studies. We always try to minimize the number of programmers on a project and to maintain the same programmers throughout the entire project to minimize any movement of information, but this is not always possible due to timelines and resource availability which would be another example of motion inefficiencies.

Transportation inefficiencies can arise where there is unnecessary movement of people, documents, tasks or information between different areas of work, e.g. offices and departments. CROS NT has programmers based in several locations, some office based and some home based which requires the appropriate IT set-up to enable this to work across multiple locations and time zones. Although this presents some challenges to documentation management, there are no transportation inefficiencies identified in table programming.

There are many types of waiting inefficiencies within the programming environment. Waiting for the SAP to get finalized, waiting for the data to be cleaned and DB locked, waiting for the datasets to be programmed and validated, waiting for the program to run in SAS®, waiting for comments from the statistician and sponsor, etc. Many of these inefficiencies can be managed by effective resource planning so that programmers work on other tasks while they are waiting. If a programmer is waiting for their program to run in SAS, this may indicate they are not writing the most efficient programs. The production program can take some time to run depending on the number and length of the outputs to be produced for a study, this waiting time can be managed by the programmer working on other tasks while the production program is running.

If the validation programmer is waiting until all the outputs are programmed prior to conducting any validation, this can cause an increase in the amount of rework that is needed. For example, all of the outputs could potentially need an update for minor findings which could have been prevented if detected earlier. Again, effective resource planning can minimize any rework so that programmers are working on the same project during the same period. If a programmer has started on another task and then has to go back to work they completed previously, it will take them longer to get back in to the programming task. There can also be rework inefficiencies if programmers are making the same updates to macros for different projects. We identified that there was rework inefficiency within our process for table programming. In particular, if the general macros didn't have the required parameters to produce the output as desired for a study, the programmer could then amend the macro, but if this style of output was needed for a different study, this may have needed to be done again. In addition, this can cause rework if different programmers end up working on the project as they will need to learn what the particular macro did for this study.

If we go beyond what is required by the customer or regulatory authority we can have over processing inefficiencies. This can happen if we apply the same validation requirements to all studies, regardless of their intended use. For example, the validation requirements for a Phase III pivotal study will be different from an analysis that is needed to support a manuscript. At CROS NT the level of validation is agreed with the client depending on their requirements. The information about the validation strategy needs to be communicated to the entire team to ensure this is followed.

At a CRO, programmers work on many projects across many therapeutic areas, and inefficiencies can occur in which people's talents are not being utilized effectively. A particular programmer may be especially skilled in oncology studies and so could produce the required outputs quicker and to a higher standard than another programmer. It is not often possible to factor this in to the resource planning to reduce this inefficiency. However, maintaining the same team on multiple projects for the same client is always preferred as that team can get to know the expectations of the client.

In summary, motion and rework were identified as the types of inefficiencies within our processes.

IDENTIFY SOLUTIONS

The inefficiencies and suggested solutions identified above, and those that were separately identified by the programmers themselves, were ranked by the ratio of likely improvement in time to the time needed for the implementation. This formed the basis for the order of the improvements and those with the greatest potential saving per hour of effort were implemented first.

In evaluating our rework inefficiencies, we found that our standard macro programs were not flexible enough for the current sponsor requirements so macros were often updated for each individual study. The programming department had grown significantly over the last few years in in terms of the number of resources and locations, and new layouts and formats were needed to satisfy the growing diversity of the customer base.

Some of the other inefficiencies identified were in how we stored our macros and the copying and updating that was routinely completed for each study. In addition, having a central repository of macros that are not modified would mean it would be easier for all the programmers to be very familiar with these macros. This would also have a positive impact on figures and listings programming as well.

IMPLEMENT AND MAINTAIN THE CHANGE

Once the potential solutions were identified, we needed to plan and implement the change. To ensure the changes were quickly and successfully accepted by the team, they needed to be planned and communicated in a positive way. The programmer who had originally developed the current suite of macros at CROS NT was assigned the task of updating the macros to add in additional parameters to cover table style, sorting options and formatting preferences. Additional table programs were created to cover standard baseline and safety tables. Other senior programmers were involved in the development of an autocall library so that the macros would be stored centrally and be accessed across all the different studies. During this time, the department was kept up to date on a regular basis so that they knew the progress that was being made and when it was likely to be implemented.

Performance controls were put into place to ensure that the interested parties can monitor the performance. This was then incorporated into the monthly review of the projects that forms part of the KPI analysis. The results of this will be shared with the department over the coming months to further encourage the team and to strengthen internal support for the process. The acceptance of the process will enable more opportunities for improvement in the future.

For this project, it is currently too early to know if the target in hours reduction will be achieved. The new macros have been implemented and are being used and have received positive feedback from the programmers. However, we don't yet have a project that has been completed to a sufficient level to see if there has been a reduction in hours. This will be possible by the end of 2016.

While working through this project for table programming, it has been clear that there are other processes that have a high impact on the costs, timelines and deliverables that could benefit from going through this same process.

CONCLUSION

An ethos of continuous improvement within a company can allow change to be implemented more quickly and effectively. Statistical programming served as a model of success for the implementation of such a process in the CRO environment. The particular process selected was table programming due to its high impact on costs, timelines and deliverables while also being critical to the business strategy and growth. The target was defined as a 15% reduction in hours for table programming. The process was broken down in to small steps to determine where any inefficiencies might be. Some inefficiencies were identified and a plan was put in place to make the necessary updates on a gradual and ongoing basis. Many of the general macros were updated and an autocall library was created. The training on the new macros has been completed and they are being used on all appropriate projects.

It is currently too early to know if the target hours reduction has been achieved, but the next process to go through this continuous improvement program is currently being identified. In addition, even though it is too early to assess progress against the target, the process has been a success for the programming department in terms of planning, teamwork and motivation.

ACKNOWLEDGMENTS

I would like to acknowledge the support of programmers in my department for the hard work that has gone in to this project along with Jo Marshall, CROS NT UK Country Manager and Continuous improvement leader for her help and support with this project.

CONTACT INFORMATION

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

Author Name	Caroline Terrill
Company	CROS NT
Address	Beechwood, Grove Park
City / Postcode	White Waltham SL6 3LW
Work Phone:	+44 (0) 1628 290141
Fax:	
Email:	caroline.terrill@crosnt.com
Web:	www.crosnt.com