PhUSE 2016

Paper PD10

Is the Statistical Programming Process LEAN Enough?

Simon Jennings, ICON Ltd, Marlow, UK

ABSTRACT

Right first time is a quality management concept that states defect prevention is more advantageous and cost-effective than defect detection and associated rework. The benefits of getting work right first time should be apparent in terms of customer satisfaction, staff satisfaction and cost control, yet can we honestly say that within the statistical programming domain we have resolved the issue? What is our reputation for quality and reliability? Is there a mentality that accepts rework is part of our normal day-to-day practice? How are we measuring whether we are getting our deliveries right first time? What are our customers saying about the quality?

INTRODUCTION

This paper will present the fundamental concepts of Lean Six Sigma and associated tools and show how these can applied to the Statistical Programming process to improve consistency and quality in outputs.

WHAT IS LEAN SIX SIGMA?

Wikipedia defines Lean Six Sigma as

> **…….** a methodology that relies on a collaborative team effort to improve performance by systematically removing waste, *combining lean manufacturing/lean enterprise and Six Sigma* to eliminate the eight kinds of waste (muda): Transportation, Inventory, Motion, Waiting, Over-production, Over-processing, Defects, and Skills (abbreviated as '**TIMWOODS**').

Lean manufacturing (lean production), often simply "lean", is a systematic method for the elimination of waste within a manufacturing/production system. Lean also takes into account waste created through overburden and through unevenness in workloads ("flow"). Working from the perspective of the client who consumes a product or service, waste is a process activity that doesn't add "value" - where value is any action or process that a customer would be willing to pay for. Essentially, lean is centred on *making obvious what adds value by reducing everything else* i.e. ***efficiency***.

Six Sigma is a set of techniques and tools for process improvement. Six Sigma seeks to improve the quality of the output of a process by identifying and removing the causes of defects and minimizing variability in manufacturing and business processes. It uses a set of quality management methods, mainly empirical, statistical methods, and creates a special infrastructure of people within the organization, who are experts in these methods. Each Six Sigma project carried out within an organization follows a defined sequence of steps and has specific value targets, for example: reduce process cycle time, reduce costs, increase customer satisfaction, or increase profits. The focus of Six Sigma is on organisational/process ***effectiveness***.

Lean management and Six Sigma are two concepts which share similar methodologies and tools. Both programs are of Japanese origin, but they are two different programs. Lean management is focused on eliminating waste and ensuring efficiency while Six Sigma's focus is on eliminating defects and reducing variability.

CONTINUOUS IMPROVEMENT

Lean Six Sigma is just one approach to the concept of continuous improvement of which there are five basic principles:

- People closest to the work know the issues and opportunities
- It is a journey of relentlessly seeking perfection
- Everything can be viewed as a process
- Performance is judged by the customer
- To manage performance you have to measure it

So when thinking about initiating a Lean Six Sigma project to statistical programming we have to first ask ourselves the following questions:

- Are we involving those closest to the issues?
- Are too many Programmers willing just to deliver something?

- Are we analyzing programming as a process?
- Are we taking on board the customer's perception of our performance? In this respect, who is the customer?
- Are we measuring performance? What measures are applicable?

SIX SIGMA TOOLKIT: DMAIC

When considering a project aimed at improving an existing process, Six Sigma uses a project methodology consisting of 5 phases under the acronym **DMAIC**.

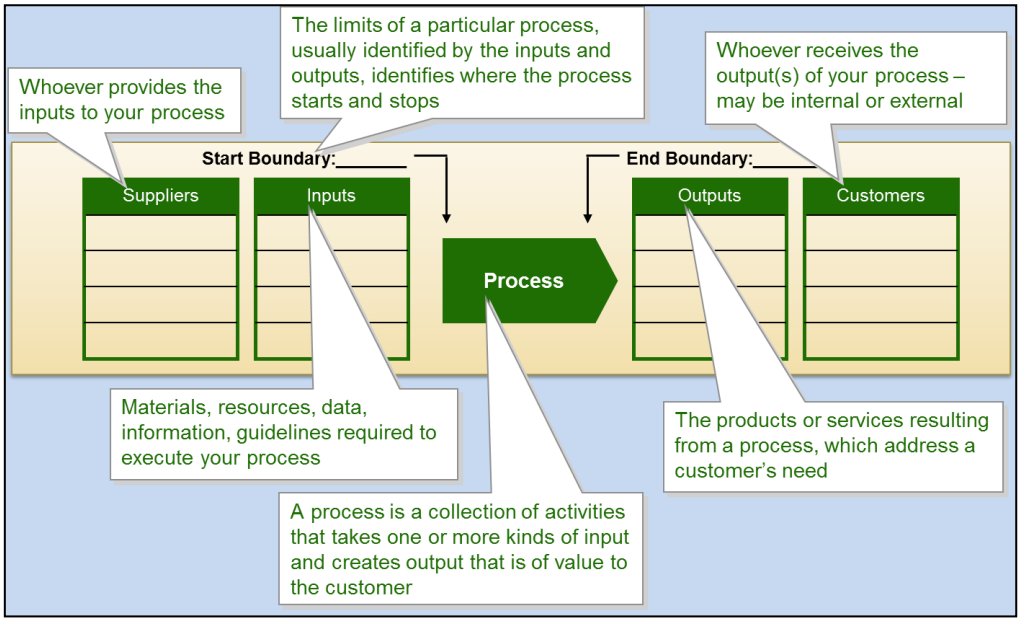
- ***D****efine* the process, the voice of the customer and their requirements, and the project goals, specifically.
- ***M****easure* key aspects of the current process and collect relevant data; calculate the 'as-is' Process Capability.
- ***A****nalyze* the data to investigate and verify cause-and-effect relationships. Determine what the relationships are, and attempt to ensure that all factors have been considered. Seek out root cause of the defect under investigation.
- ***I****mprove* or optimize the current process based upon data analysis using techniques such as design of experiments to create a new, future state process. Set up pilot runs to establish process capability.
- ***C****ontrol* the future state process to ensure that any deviations from the target are corrected before they result in defects. Implement control systems such as statistical process control, production boards, visual workplaces, and continuously monitor the process. This process is repeated until the desired quality level is obtained.

We will focus on some of the steps from the Design phase – namely mapping the process and voice of the customer.

'DEFINE' PHASE – PROCESS MAPPING

A process mapping tool called **SIPOC** can be used during the design phase to help visualize the full scope of a process and what influences it in terms of:

- **S**uppliers
- **I**nputs
- **P**rocess
- **O**utputs
- **C**ustomers



Several aspects of the SIPOC that may not be readily apparent are:

- Suppliers and customers may be internal or external to the organization that performs the process.
- Inputs and outputs may be materials, services, or information.
- The focus is on capturing the set of inputs and outputs rather than the individual steps in the process.

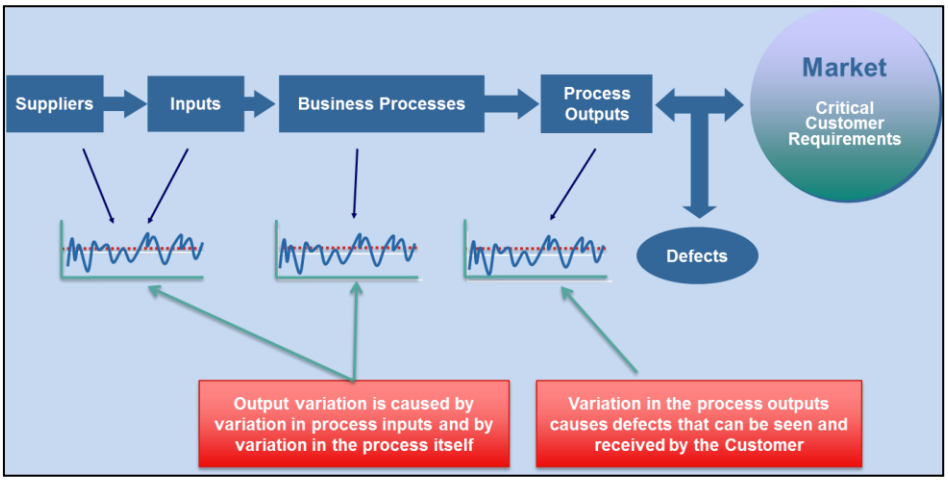
When mapping a process, one can take part of the process or the entire process but need to clearly define start and stop points.

Example SIPOC: High-level Programming Process

Suppliers	Inputs	Process	Outputs	Customers
• DM • Clients/sponsor • Biostatistics • Clinical • Patients	• Protocol • Data • DM Plan • SAP • TLF Mocks • Programming conventions	• Training • People • Program development & testing • Program validation • Code review • Lead Programmer review	• TFLs • Datasets • Programs • Logs • Documentation	• Biostatistics • Medical Writing • Clients • Regulatory authorities • Patients

OUTPUT VARIATION

Ultimately any Lean Six Sigma project is looking for process improvement – reducing variation in the outputs and thereby reducing the number of defects (e.g. incorrect TLFs). It is variation in terms of suppliers, inputs and process that lead to variation in the outputs.



VARIATION IN INPUTS

What can/do we do to control variability in suppliers and inputs in the statistical programming process? If we look at some typical inputs to the programming process, most organizations already take steps to control some aspects of input variation:

Input	Methods to control variation
Data (quality)	• Standard CRFs e.g. CDASH • Standard edit/consistency checks • Data standards e.g. SDTM • Data transfer quality checks e.g. SDTM datasets • Handover acceptance criteria/clean data requirements
TLF mocks/shells	• Standard TLF mock templates • Standard formatting conventions e.g. significant digits, precision, display rules, treatment ordering, etc.

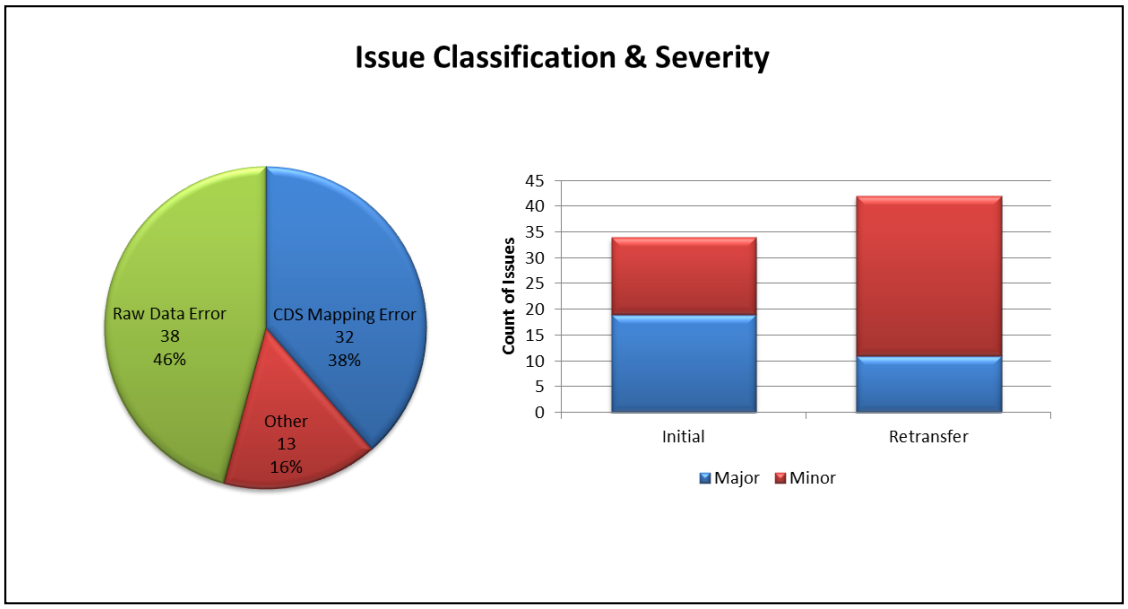
As you can see standardisation features in many of these; however, standardisation does not reduce the need for critical/robust thinking. Time expended upfront on inputs/planning is critical to reducing variation (process cycle time and/or quality). Nonetheless, even with standardisation and adequate planning time there is still residual variation in the inputs. Additional steps in a process are frequently added in because there is no confidence that earlier steps in the process (inputs) are working optimally. This is why steps such as review of TLF mocks are frequently performed by Programming Leads before programming commences. However, this goes against the idea of cutting out waste (remember lean?). As a side note, it is often the case that additional checks are added into a process following critical/major audit findings instead of performing a true root-cause analysis; this process of performing checks on top of checks often masks the true source of supplier/input variation.

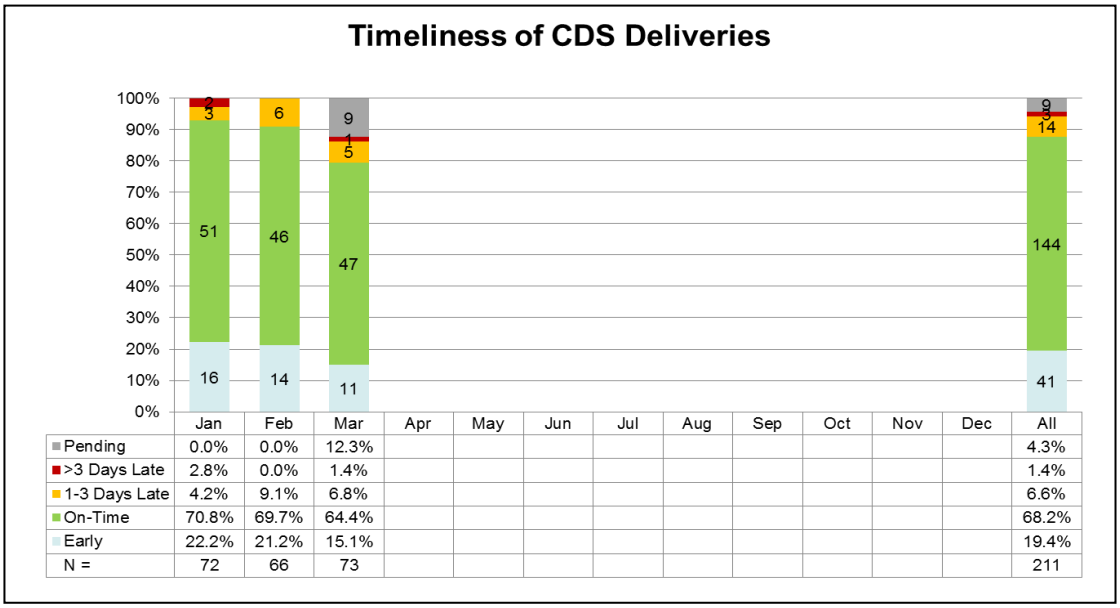
MEASUREMENT OF VARIABILITY

Objective measurement of process variability is critical to ensure evidence-based decision making. If we don't measure it then we cannot determine if any change in process has an impact on variability or a particular step is non-value added. For example, in the case of the Programming Lead reviewing the mock TLFs before programming starts, if 100% of the mocks are completely standard then it is clear that the review step becomes superfluous.

Example: Input Measurement of data quality

At ICON, all data quality issues impacting development of program code are systematically logged and captured. These can be CRF data or structural/mapping issues e.g. SDTM mapping. The combined data are analysed monthly to look at trends in performance of B&P's primary supplier.





	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	All
Pending	0.0%	0.0%	12.3%										4.3%
>3 Days Late	2.8%	0.0%	1.4%										1.4%
1-3 Days Late	4.2%	9.1%	6.8%										6.6%
On-Time	70.8%	69.7%	64.4%										68.2%
Early	22.2%	21.2%	15.1%										19.4%
N =	72	66	73										211

VARIATION IN PROCESS

Variation in process outputs is driven by variation in the process itself. The first step in analyzing process variation is to map the process as completely as possible, measure the variation of the relevant process steps (e.g. quality, cycle time, etc.) and then to investigate the different sources that contribute to this variability e.g. geography/region, job level, qualifications, experience, therapeutic area, etc. It is important to ensure that the people closest to the actual day-to-day work are involved in process mapping to ensure that all of the steps, inputs and outputs are captured.

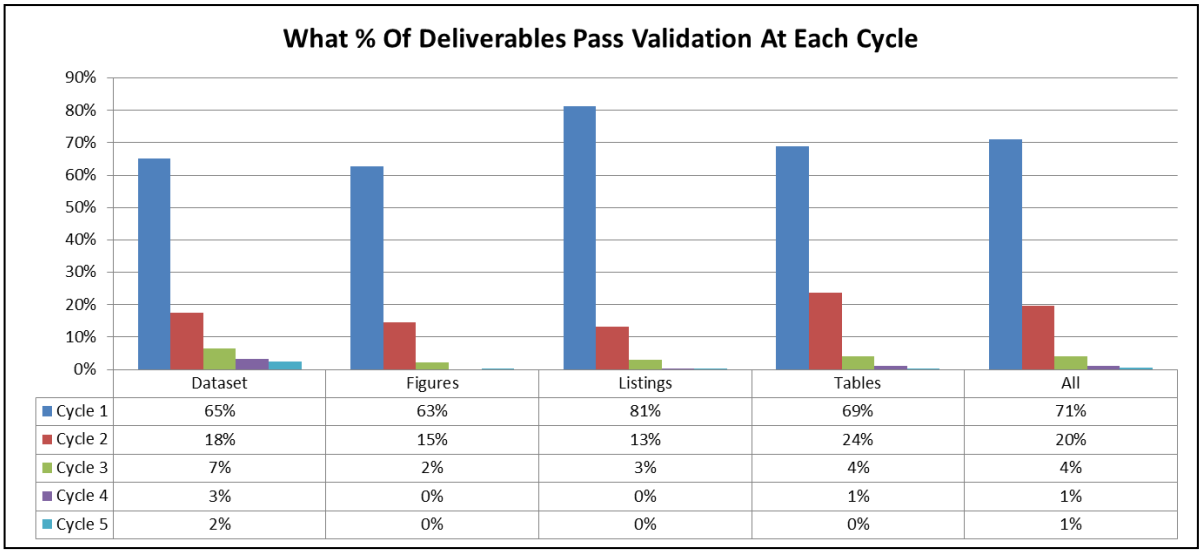
In the statistical programming process, there are many approaches applied to control process variation, some of which are listed below:

Methods to control variation
• SOPs • Training (process, technical, therapeutic, study-specific) • Defined roles and responsibilities • Standard macros/tools, standard directory structures • Global, harmonized programming environment • Template code; code libraries • Metadata-driven reporting systems; dependency-management systems • Programming guidelines/conventions • Best practices/guidelines • Subject matter experts

Possible metrics to measure sources of statistical programming process variation include: team turnover (% key member turnover/annum); number of SOP deviations/waivers, usage of standard macros/tools/utilities, number of iterations to produce validated dataset, table, figure, listing per release; % datasets, tables, figures, listings right-first-time (i.e. one development/validation cycle).

Example: Measuring 'Right-First-Time'

All findings arising from the independent validation process (usually independent programming) are captured in a systematic manner in a Programming Quality Tracker and are analyzed regularly to monitor the quality of production outputs at the point of handover to validation.



	Dataset	Figures	Listings	Tables	All
Cycle 1	65%	63%	81%	69%	71%
Cycle 2	18%	15%	13%	24%	20%
Cycle 3	7%	2%	3%	4%	4%
Cycle 4	3%	0%	0%	1%	1%
Cycle 5	2%	0%	0%	0%	1%

VOICE OF THE CUSTOMER

Another concept from the 'Define' phase of DMAIC is the 'Voice of the Customer' (VOC). VOC is a process used to capture the requirements/feedback from the customer (internal or external) to provide the best in class service/product quality. This process is all about being proactive and constantly innovative to capture the changing requirements of the customers with time.

The "voice of the customer" is the term used to describe the stated and unstated needs or requirements of the customer.

This data is used to identify the quality attributes needed for a supplied component, material or service to incorporate in the process or product i.e.

- What does the customer need or want? What are the attributes and features of the service, product, etc. that are Critical to Quality (CTQ)?
- How well does the existing product/service meet the needs of the customer?
- What suggestions does the customer have about ways to better meet their needs?

Essentially, we are trying to answer the question: What is the customer willing to pay for?

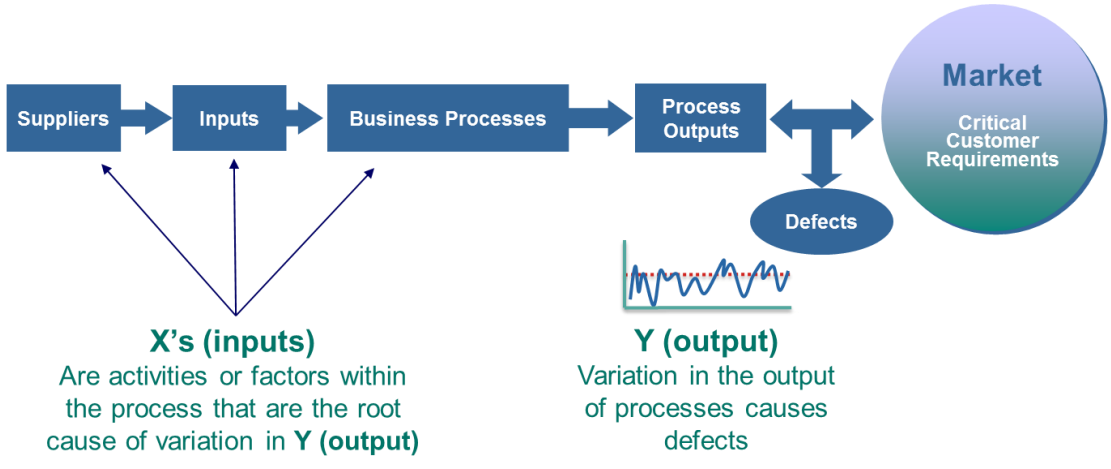
From a programming perspective customers want consistency, "added-value", expertise, but what do these concepts mean for your customer and how are they defined? At what timepoints in the process do you measure these? Are they looking at this per output, per deliverable, per study, per program?

The voice of the customer can be captured in a variety of ways. One can use so-called passive systems (complaints, sales data, system alerts, etc.) to elicit customer data where information comes to you whether you like it or not. Alternatively, proactive systems are used where effort is required to gather the information on customer requirements e.g. surveys, focus groups, brainstorming. When using proactive methods to gather VOC data, a common mistake is failing to probe for customer needs with sufficient precision or failing to ask for additional information to qualify a response e.g. why, how often, when?

The following table provides three key attributes for effective customer questions – specific, measurable and prioritised.

Three features of *effective* customer questions:	
1. Specific	• You said our outputs are poor quality. How do you define quality? What specific features do you look for?
2. Measurable	• You said that we are often late. What is late for you? • You said we produce too many tables. What number do you want?
3. Prioritised	• You listed five features. Could you rank them? • What are your non-negotiable requirements? . . . What else is important? . . . What would surprise and delight you? • What do you like most? Least?

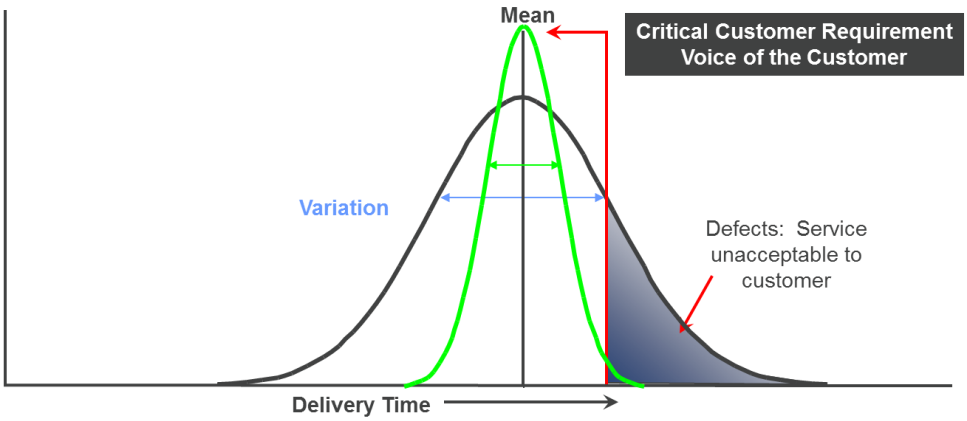
Ultimately, defects in product/service lead to customer dissatisfaction and poor business performance. Defects arise out of variation in outputs which is a (non-specified and non-linear) function of activities or factors due to suppliers, inputs and process.



$$Y = f(X_1, X_2 \ldots, X_k)$$

The fundamental equation of Six Sigma

The goal is to reduce process output variation (according to whatever metric(s) are relevant to the customer) to minimize – and ideally eliminate – defects. Defects arise as a consequence of excess variation in performance beyond tolerance thresholds as articulated by the customer e.g. >1 major programming error (non-cosmetic) in final planned analysis package.



MEASURING OUTPUT VARIATION

Measurement of process output variability is no different from measurement of variability in inputs or process but should focus on what is important to the customer and what level of service is considered unacceptable (under-performance). As mentioned above, if possible it is also desirable to ascertain what the customer would consider as exceptional or over-performance but in many cases customers are reluctant to articulate this or set unrealistic targets e.g. no programming errors in all deliverables in all studies.

Example output measures that can be used for the programming process:

Process Output	Measure	Defect (examples)
Datasets & TLFs	# of major/minor programming errors (per deliverable)	1+ major programming errors and/or 10+ minor programming errors (major/minor as per agreed definition)
	% of deliverables not fit for purpose (i.e. with at least one defect)	>5%
Documentation	# of critical/major audit findings	>0 critical and/or >1 major finding
Cost	Average cost/TFL	
Time	# days deviation from agreed delivery plan (per deliverable)	>1 day late
	% of deliverables on (or before) time	≤90%
Customer satisfaction	Meets/exceeds expectations (e.g. rated on 5 point scale)	<85% of studies rated as meets/exceeds expectations

Some of the metrics are completely independent of one another and can tell the whole story; however, most alone only tell part of the story. Reviewing and assessing all of the metrics as a package will give you a more complete picture of performance and what is occurring within your process.

The following example dashboard measures performance in terms of quality and timeliness and the combination of both as well as looking at “key” deliverables only:

Measure	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	YTD	Last 3 mos.
All Deliverables	**29**	**35**	**25**	**19**	**25**	**26**	**21**	-	-	-	-	-	**180**	**72**
On Time	29 (100.0)	35 (100.0)	25 (100.0)	19 (100.0)	25 (100.0)	25 (96.2)	21 (100.0)	-	-	-	-	-	179 (99.4)	71 (98.6)
Traffic Light	G	G	G	G	G	G	G						G	G
Fit for Purpose	27 (93.1)	35 (100.0)	25 (100.0)	19 (100.0)	25 (100.0)	25 (96.2)	21 (100.0)	-	-	-	-	-	177 (98.3)	71 (98.6)
Traffic Light	Y	G	G	G	G	G	G						G	G
On Time & w/Quality	27 (93.1)	35 (100.0)	25 (100.0)	19 (100.0)	25 (100.0)	25 (96.2)	21 (100.0)	-	-	-	-	-	177 (98.3)	71 (98.6)
Traffic Light	Y	G	G	G	G	G	G						G	G
Key Deliverables	**8**	**12**	**7**	**3**	**6**	**13**	**4**	-	-	-	-	-	**53**	**23**
On Time	8 (100.0)	12 (100.0)	7 (100.0)	3 (100.0)	6 (100.0)	12 (92.3)	4 (100.0)	-	-	-	-	-	52 (98.1)	22 (95.7)
Traffic Light	G	G	G	G	G	Y	G						G	G
Fit for Purpose	8 (100.0)	12 (100.0)	7 (100.0)	3 (100.0)	6 (100.0)	12 (92.3)	4 (100.0)	-	-	-	-	-	52 (98.1)	22 (95.7)
Traffic Light	G	G	G	G	G	Y	G						G	G
On Time & w/Quality	8 (100.0)	12 (100.0)	7 (100.0)	3 (100.0)	6 (100.0)	12 (92.3)	4 (100.0)	-	-	-	-	-	52 (98.1)	22 (95.7)
Traffic Light	G	G	G	G	G	Y	G						G	G

Notes:
Performance Thresholds: Green-95.0+, Yellow-85.0-94.9, Red-<=84.9
Key Deliverables: Clinical Study Report, DMC/DSMB, Top Line Report, Interim Analysis or Summary of Clinical Safety, Efficacy or Pharmacology TLFs

Balancing meeting the needs of the customer and eliminating the waste in the process are sometimes competing priorities; frequently organizations will put in additional quality steps to control for the need to meet stringent quality requirements, rather than have better up-front planning and input processes. As discussed earlier, these additional quality checks are a safety net for upstream flaws in inputs and earlier steps in the process whereas more effort should be expended on understanding how to reduce the variation that leads to excessive and repeat checking and potential revision at the end of the process.

CONCLUSION

The programming process has all of the characteristics of any other service and product production process and in order to manage – and improve – performance it is critical to measure and analyse each and every step in the process (including inputs and outputs). Lean six sigma provides the methodology and tools to support process improvement.

Lean Six Sigma reminds us of the importance of:

- Metrics
- Voice of the customer
- Standardisation and the elimination of waste
- Planning vs fixing
- Defining what is critical to quality

In general, there is significant room for improvement in terms of making the programming process more lean and ensuring that all sources of waste are eliminated from the process. As has been shown, additional process steps are frequently added because we fail to deliver right-first-time quality in upstream inputs into and within the programming process.

ACKNOWLEDGMENTS

Many thanks to Pam Howard for the opportunity to reuse her material presented at a recent PhUSE SDE in Chennai.

CONTACT INFORMATION

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

Simon Jennings
ICON Clinical Research UK Ltd
2 Globeside Business Park
Marlow SL7 1HZ
Work Phone: +44 1628 496489
Email: simon.jennings@iconplc.com
Web: www.iconplc.com