Paper AS17

Beyond Box Checking:
Five Best Practices for Embedding QC in Modern SCEs

Matthew Tendler, Domino Data Lab, San Francisco, USA

ABSTRACT

Quality control (QC) remains a cornerstone of regulatory confidence in clinical trial analyses, but too often it is handled as a separate, downstream activity. This paper shares best practices for integrating QC directly into statistical computing environments (SCEs), ensuring quality is built into workflows rather than added at the end. Drawing on lessons learned from global clinical teams and informed by industry survey data, we show how organizations can embed QC across data, code, and environments to reduce rework, strengthen reproducibility, and improve submission readiness. We present five concrete best practices: Git-first workflows, automation by default, flexible risk-based categorization, end-to-end collaboration models, and AI-ready architecture. By moving QC upstream and embedding it in daily practice, these approaches go beyond checklists to deliver continuous quality assurance in clinical analytics.

1. INTRODUCTION

Statistical computing environments have evolved significantly over the past decade, yet quality control practices at many pharmaceutical organizations remain anchored to patterns established when SAS was the only programming language, studies were smaller, and regulatory expectations for traceability were less explicit. The result is a persistent gap: powerful modern tools exist, but much of the time spent by statistical programmers goes to stitching them together, managing trackers, and navigating disconnected QC processes.

In a 2025 presentation, we introduced a maturity model for SCE modernization and outlined a set of foundational principles. This paper extends that framework with five prescriptive best practices, informed by direct experience working with global pharmaceutical organizations on SCE modernization initiatives, and grounded in survey data collected from statistical programmers working across pharma, biotech, and CRO environments.

Four converging forces make this work urgent. First, regulatory agencies increasingly expect full traceability - not just final outputs, but the lineage that produced them. Second, data complexity is exploding as real-world data, decentralized trials, and complex endpoints overwhelm legacy validation approaches. Third, the industry is undergoing a structural transition from SAS-only environments to multi-language SCEs, with a meaningful and growing proportion of statistical programmers now working in R alongside SAS. Fourth, there is a significant gap between AI ambition and production readiness: while many organizations are experimenting with AI-assisted programming internally, very few have included AI-generated outputs in regulatory submissions. These forces create both urgency and opportunity for the best practices described in this paper.

2. THE CURRENT STATE OF QC IN STATISTICAL PROGRAMMING

To ground this work in empirical observation rather than opinion alone, we reviewed survey data from statistical programmers working across pharma, biotech, and CRO environments. The findings paint a consistent picture of an industry where QC practices have not kept pace with the tools and complexity of modern clinical development.

2.1 QC Tooling and Traceability

A significant proportion of statistical programming teams operate without a dedicated QC tracking system. Many rely on email-based approvals, spreadsheet trackers, or ad hoc methods to manage quality workflows. When asked to rate the accessibility of end-to-end traceability from SDTM through ADaM into TFLs and CSR outputs, respondents generally rated their organizations as moderate at best, with pharma-based teams tending to score lower than their CRO and biotech counterparts. More than half of those surveyed reported relying on manual documentation as their primary traceability method - a finding that underscores the gap between what regulators expect and what most teams can deliver today.

2.2 The Prevalence of Default Double Programming

Perhaps the most striking finding is the prevalence of blanket double programming. A large majority of respondents reported that their organization defaults to 100% independent double programming for ADaM datasets, regardless of the criticality of the output. The pattern holds across TFLs and SDTM as well, with roughly half of respondents reporting that they double-program all three domains by default. When asked about barriers to improving QC, respondents described this default as entrenched in SOPs and resistant to change - even among teams that recognize it as over-conservative.

2.3 Where QC Effort Concentrates

When asked which stages of QC require the most time and effort, statistical programmers consistently identified ADaM dataset QC as the top bottleneck, followed by TFL QC and SDTM dataset QC. Cross-team coordination - handoffs between programmers, biostatisticians, and medical writers - also emerged as a significant effort area. The features rated as most critical for QC tracking were **version history**, **ease of use**, **audit trail**, and **dependency tracking** - a set of requirements that maps directly to the best practices described in this paper.

2.4 AI Readiness

The gap between AI aspiration and production use is substantial. Very few organizations have incorporated AI-generated outputs into regulatory submissions, and a large proportion of respondents indicated uncertainty about whether their organization permits it at all. When asked what would increase confidence in AI-based outputs for submissions, the top responses centered on industry standards, validation frameworks, clearer regulatory guidance, and - critically - better audit trail and traceability. This last point is significant: it suggests that the foundational practices described in this paper are themselves prerequisites for responsible AI adoption.

3. THE MODERNIZATION MATURITY MODEL

Before prescribing solutions, it is important to diagnose where an organization currently stands. We propose a four-stage maturity model that describes the typical journey from fragmented QC to embedded, AI-enabled quality assurance.

Stage 1 – Manual: QC tracking happens through email approvals, spreadsheet-based trackers, and isolated tools. Reproducibility depends on individual programmers rather than systems. Audit confidence is low.

Stage 2 – Fragmented: Some automation exists but is inconsistent across studies and teams. Partial version control adoption. Standards exist on paper but enforcement is manual.

Stage 3 – Unified: A governed SCE provides a single environment for SAS, R, and Python. QC is embedded in workflows with Git-based traceability, automated validation, and real-time dashboards. Risk-based QC policies are versioned and traceable.

Stage 4 – AI-Enabled: Risk-based planning is informed by AI. Coding agents assist with routine programming tasks. End-to-end traceability extends from protocol through submission. Cross-study intelligence enables reuse at scale.

Based on industry survey data and direct experience with large pharmaceutical organizations, most teams currently sit between Stages 1 and 2. Critically, the order of progression matters - most modernization failures occur when organizations attempt to skip stages, particularly when they pursue AI capabilities before establishing the foundational governance and traceability infrastructure required by Stages 2 and 3.

4. BEST PRACTICE #1: GIT-FIRST WORKFLOWS

Version control is the single highest-ROI modernization step a statistical programming team can take. When every script, specification, and configuration file lives under Git, reproducibility becomes structural rather than aspirational. This practice directly addresses the most frequently cited feature needs among statistical programmers: version history and audit trail.

4.1 What Git-First Means in Practice

- All statistical programs, macros, and configuration files are stored in Git repositories with enforced branching strategies.
- Production branches are protected; exploratory work happens on feature branches, enabling compliance without constraining scientific exploration.
- Commit history provides an immutable audit trail, eliminating the need for separate lineage documentation.
- Code review becomes a natural QC gate integrated into the development workflow, rather than a bolt-on step.

4.2 Lessons from Large Pharma Deployments

In extensive SCE deployments at large pharmaceutical organizations, Git-based traceability has proven to be a foundational requirement. A key capability is the ability to trace any output back to the specific dataset snapshot and code version that produced it. The implementation requires connecting batch execution with dependency tracking, so that if upstream data changes, downstream outputs are automatically flagged as potentially stale. This structural approach to traceability replaces manual lineage tracking and substantially reduces audit preparation effort - addressing a pain point felt acutely by the majority of statistical programmers who still rely on manual documentation.

5. BEST PRACTICE #2: AUTOMATION BY DEFAULT

The principle of automation by default means that outputs land where they need to - every time, without manual file management. This eliminates an entire category of errors and delays that plague traditional SCE workflows. Industry data suggests that current automation efforts are delivering only modest returns, likely because they are piecemeal rather than systemic.

5.1 Key Automation Capabilities

- Batch execution with full dependency tracking: users define job chains once, and the system executes them in order with automatic snapshot management.
- Metadata-driven TFL generation pipelines that configure outputs from study specifications rather than manual programming.
- Automatic dataset snapshots tied to the jobs that produced them, enabling one-click dependency visualization.
- Reusable pipeline stages configured once and applied across studies, ensuring consistent, validated execution patterns.

5.2 Why Current Automation Falls Short

Many teams have adopted some degree of automation - macro libraries, template-based generation, custom validation scripts - yet report only marginal improvement in QC efficiency. The root cause is typically that automation is layered on top of fragmented processes rather than embedded into a unified workflow. Dependency tracking, one of the most frequently cited critical features among statistical programmers, requires system-level integration that piecemeal tools cannot provide. Automation should be invisible: when it works well, programmers simply do not think about file management, versioning, or output routing.

6. BEST PRACTICE #3: FLEXIBLE, RISK-BASED CATEGORIZATION

One of the most significant inefficiencies in current QC practice is the default to double programming for all deliverables, regardless of their criticality. Industry data consistently shows that a large majority of teams apply 100% independent double programming to ADaM datasets, TFLs, and often SDTM as well - with no differentiation between primary efficacy endpoints and exploratory listings. This over-conservative approach consumes enormous resources and, paradoxically, can reduce overall quality by spreading reviewer attention equally across high-risk and low-risk outputs.

6.1 A Risk-Based Framework

We propose categorizing deliverables by criticality - for example, CSR primary endpoints versus supportive analyses versus exploratory listings - and scaling QC depth accordingly. Critical outputs warrant full double programming; supportive outputs may require independent code review; exploratory outputs may need only automated validation checks. This framework must be paired with full version history of QC policy decisions, so that changes to categorization are traceable and auditable. Statistical programmers themselves recognize the need for this evolution - many describe the current default as entrenched in SOPs but no longer reflective of best practice.

6.2 AI-Driven Risk Categorization

Looking forward, AI-driven risk categorization represents a natural extension of this practice. By analyzing study metadata, historical defect patterns, and output characteristics, AI systems can suggest appropriate risk categories for new deliverables. This does not replace human judgment but augments it - study leads review and approve AI suggestions rather than creating categorizations from scratch. The result is faster, more consistent planning that still maintains human oversight and accountability.

7. BEST PRACTICE #4: END-TO-END COLLABORATION MODELS

Traditional QC tracking is scoped too narrowly. When QC tracking begins and ends with the programming team, critical handoff points to adjacent stakeholders become sources of delay

and error. Cross-team coordination consistently appears as a significant effort area in industry surveys, and the stages that consume the most QC effort - ADaM and TFL QC - are precisely the stages that require the most collaboration between programmers, biostatisticians, and medical writers.

7.1 Stakeholders Beyond Programming

- Medical writers need real-time visibility into TFL status to plan CSR drafting, rather than relying on email-based status updates.
- Biostatisticians should review and approve QC plans before programming begins, not after outputs are already produced.
- Clinical data managers need to see how upstream data changes propagate through to downstream outputs.
- Self-service dashboards replace status meetings and tracker spreadsheets, transforming the SCE from a development environment into a collaboration platform.

When asked to describe their ideal QC system, statistical programmers consistently describe something integrated, automated, and connected to the broader deliverable lifecycle - not a standalone tracker that requires manual input. The SCE must evolve from a place where code is written to a platform where clinical deliverables are collaboratively managed.

8. BEST PRACTICE #5: AI-READY ARCHITECTURE

The final best practice is forward-looking: building SCE architecture that can support AI capabilities as they mature. This is deliberately the last practice because AI only works well on a solid foundation of governance, traceability, and standardized workflows.

8.1 Where AI Pays Its Dividend

The most transformative AI dividend in clinical programming will not come from coding assistants alone. While LLMs are already proving valuable for high-friction, low-risk tasks like code translation (SAS to R), specification drafting, log summarization, and cross-study knowledge retrieval, the real value lies in end-to-end orchestration: from protocol to SAP to metadata to output generation, with AI accelerating the entire pathway.

It is critical to be honest about the current state. Deterministic statistical packages - SAS and R - remain essential and non-negotiable for regulatory submissions. LLMs are not replacing validated statistical procedures. The opportunity for AI lies in three areas: acceleration of routine tasks, intelligent assistance (such as AI-suggested QC risk categorizations and automated impact analysis), and knowledge management across studies and programs.

8.2 The Foundation Connection

A key insight from industry data is that better audit trail and traceability is among the top factors that would increase confidence in AI-based outputs for regulatory submissions. This means that Best Practices 1 through 4 - Git, automation, risk-based QC, and collaboration - are not merely independent improvements. They are the foundation that makes AI outputs trustworthy. Building AI readiness is not about purchasing AI tools; it is about establishing the governance, traceability, and workflow infrastructure that makes AI outputs auditable, reproducible, and defensible.

8.3 The Evolving Role of the Statistical Programmer

For statistical programmers, the role is evolving from writing every line of code to orchestrating, validating, and governing AI-assisted workflows. This is not a threat - it is an amplification. The domain expertise that statistical programmers carry in clinical data structures, regulatory requirements, and statistical methodology is irreplaceable. AI amplifies that expertise by handling routine execution while programmers focus on judgment, quality governance, and scientific interpretation. The organizations that invest in this transition now will see compound returns over the next two to three years as AI foundations mature.

9. CONCLUSION

Quality control in clinical programming has too long been treated as a downstream, parallel process. Industry data confirms what many have long suspected: QC practices have not kept pace with the tools and data complexity of modern clinical development. A significant proportion of teams lack dedicated QC tools, rely on manual traceability, default to double programming regardless of criticality, and are far from incorporating AI into submission workflows. These are experienced professionals who know the current state falls short.

The five best practices presented in this paper - Git-first workflows, automation by default, flexible risk-based categorization, end-to-end collaboration models, and AI-ready architecture - provide a concrete, sequential framework for closing this gap. They are informed by direct experience with large pharmaceutical SCE modernization initiatives and validated by consistent patterns in industry data. The sequence is deliberate: each practice builds on the last, and the foundational steps of version control and automation are prerequisites for the risk-based planning and AI capabilities that follow.

By moving QC upstream and embedding it in daily practice, organizations can reduce rework, strengthen reproducibility, improve submission readiness, and accelerate clinical timelines. The organizations that invest in these SCE foundations today will be best positioned to lead in clinical development efficiency as the industry continues to evolve.

ACKNOWLEDGMENTS

The author would like to thank the statistical programmers who contributed survey data, the SCE teams at partner organizations for their collaboration, and the broader PHUSE community for ongoing dialogue on SCE modernization. The insights in this paper reflect the collective experience of many clinical programming teams working to advance the state of the art in statistical computing environments.

CONTACT INFORMATION

Matthew Tendler

Head of Life Sciences Product Management

Domino Data Lab

matthew.tendler@dominodatalab.com