

Artificial Intelligence Adoption in Statistical Programming

A PHUSE Programming Heads Council Vision for the AI Era

phuse.global

Contents

Executive Summary 1

PHC Vision for Statistical Programming in the AI Era 1

Agentic Automation Use Cases in Programming 2

Human-in-the-Loop and Validation Models 2

How We Get There: Capability Roadmap and Operating Actions 3

Key Takeaways and Reflections 3

About the Authors 4



Revision History

Version	Date	Summary
1.0	October 2026	

Executive Summary

The PHUSE Programming Heads Council (PHC) positions AI as a defining leadership opportunity for statistical programming, not as a technology trend to observe from the sidelines. AI will automate routine implementation work, accelerate quality review, change how programming work is orchestrated, and create new expectations from clinical, statistical, medical writing, regulatory and data science partners. The PHC therefore positions statistical programming as an active architect of the future operating model for clinical evidence generation.

Our direction is clear: we will move from output production to evidence enablement; from efficiency alone to effectiveness; from downstream validation to upstream design; and from passive human-in-the-loop approval to accountable human leadership over data, analytics, interpretation and AI-enabled workflows. This paper shows the operational AI cases reality grounded: TFL generation and code assistance are the most mature near-term use cases; ADaM and end-to-end agentic workflows remain more difficult; and validation, consistency checking, standards compliance and knowledge retrieval may create value quickly. At the same time, the paper makes the leadership ambition explicit: statistical programming must shape how AI is embedded safely, productively and responsibly in the programming lifecycle, while keeping its core mandate firmly in focus – ensuring 100% quality, transparency and traceability in a complex data and technology environment.

PHC Vision for Statistical Programming in the AI Era

PHC Vision

Statistical programming is moving towards an AI-enabled operating model in which programmers act as data actors, data evidence architects, accountable reviewers and data workflow orchestrators. We will embed AI into programming flows, strengthen data lifecycle expertise, and build the next generation of professionals who can use AI while preserving scientific rigour, regulatory trust and deep understanding of clinical trial data.

AI is already reshaping drug development workflows, and we should proactively shape this shift. Statistical programmers have a unique combination of capabilities in the clinical development lifecycle. We understand clinical trial data, its sources, its limitations, its transformations, and its downstream use. We also understand the regulated environment in which transparency, reproducibility, traceability and accountability remain essential. These capabilities and the end-to-end accountabilities position programming leaders to drive responsible AI adoption rather than simply react to it.

The PHC envisions a future in which statistical programming is no longer defined by the manual creation of outputs. We will define our value by preparing trustworthy clinical trial data, shaping analytical questions, enabling efficient and auditable evidence generation, and supporting interpretation across the development lifecycle. AI will help us automate, delegate or support coding, information retrieval, synthesis, quality review, and pressure-testing of proposals. It will also allow us to investigate clinical data at unprecedented scale, enabling exploration across studies, programmes, indications and therapeutic areas to uncover trends, generate hypotheses, contextualise findings and accelerate evidence generation. Our unique understanding of clinical trial data – its provenance, transformations, limitations and intended use – positions statistical programmers as stewards of data-driven discovery and trusted partners in scientific and regulatory decision-making. We will use these capabilities to move closer to scientific and regulatory decision-making, not further away from it.

Statistical programmers occupy a unique position at the intersection of clinical trial data, analytics, standards and regulatory compliance. We understand where data originates, how it's transformed, the assumptions it contains, and how it supports regulatory decisions. As organisations adopt AI-enabled workflows, these capabilities make statistical programming leaders natural stewards of responsible AI adoption, ensuring innovation is accompanied by governance, inspection readiness and trust. This knowledge to define the provenance and traceability is our strongest and most defensible argument because it clearly differentiates statistical programmers from data scientists, software engineers and AI specialists, and directly links to regulatory expectations.

This vision requires a change in language and posture. We should not describe programmers as waiting to be impacted by AI. We are in the driver's seat. We will lead our role towards higher-value contributions: effectiveness over efficiency, architecture over execution, data stewardship over output production, and proactive design over reactive validation. AI adoption will not remove accountability; it will make accountable leadership with a holistic understanding of an entire product analytics and data package more important. The programmer of the AI era must understand data sources, data limitations, derivations, algorithms, review evidence, stakeholder needs, and the consequences of wrong answers.

The PHC also aspires to a future for a broader professional identity: statistical programmers will remain technical experts, but the centre of gravity will shift to data fluency, applied judgement, cross-functional influence, and connecting clinical questions with reliable data and reproducible workflows. We will work alongside statisticians, data managers, data scientists, medical writers and regulatory colleagues as partners in evidence architecture. We will not merely validate what AI produces downstream but holistically provide inputs along the lifecycle. We will help design the questions, guardrails, metadata, tools and workflows that allow AI to produce value upstream.

Operating impacts will be significant. Routine implementation work will shrink; data interpretation, cross-study integration, scientific judgement, stakeholder engagement and evidence architecture will grow. The programmer becomes a data actor: someone who understands what data can support, where data limitations create risk, and how to make strategic data decisions to ensure the right data is available for the right decision. This identity sits between technical implementation and scientific decision-making and should become explicit in role descriptions, training, and career frameworks.

Agentic Automation Use Cases in Programming

The PHC sees agentic automation as a practical path to reshape programming work, starting with structured and repetitive activities. The strongest near-term candidates are TFL generation, code assistance, automated code review, standards checking, consistency review across outputs, process knowledge retrieval, and support for regulatory or ad hoc response preparation. These use cases help programming teams move from manual repetition to controlled orchestration. Programming organisations are moving business cases in AI from prototyping to production.

TFL generation is the most visible current use case because it draws on specifications, metadata, standards, templates, prior code and controlled packages. The target operating pattern is not blind automation. Programming organisations must support stepwise, accountable workflows in which AI interprets source materials, proposes code or outputs and programmers review the logic, context and fitness for purpose. We should position this as a capability that expands capacity and improves consistency while preserving sponsor accountability.

ADaM generation remains a higher-value but harder use case. It depends on derivation logic, data standards, disease-area conventions, endpoint definitions and study-specific assumptions. Programming teams should treat ADaM automation as a driving force for better metadata, clearer specifications, stronger standards and better machine-readable documentation. The greatest barriers to AI adoption are often not technological. They are fragmented standards, ambiguous specifications, disconnected metadata, undocumented assumptions and incomplete data lineage. Advanced AI use cases expose these weaknesses because they depend on information that is structured, consistent and machine-readable. Programming organisations should therefore use AI as a catalyst to improve standards, metadata, documentation and governance. When AI fails, the lesson is often not that the model is inadequate, but that our information ecosystem requires greater clarity, consistency and traceability.

Validation and quality review may provide faster value than end-to-end generation. AI can review code, compare outputs, detect inconsistencies, check regulatory and company standards adherence, and identify missing assumptions. These tasks do not replace human accountability; they allow human reviewers to focus on plausibility, scientific meaning, risk, and exceptions. We should therefore prioritise AI not only as a creator of artefacts, but as a quality amplifier embedded in the programming workflow.

Knowledge management is another strategic use case. Agents can help locate prior ad hoc requests, regulatory questions, historical outputs and relevant code patterns. This will only work if knowledge is documented and accessible. Programming organisations should therefore treat documentation, metadata and searchable repositories as part of AI strategy, not as administrative overhead.

Human-in-the-Loop and Validation Models

The PHC envisions a shift from repetitive QC to risk-aware QA, transforming QC from manual, fragmented checks to a smarter, risk-based, expert in the loop quality review ecosystem cross-functionally. Independent double programming has historically provided confidence, but AI creates new validation patterns: AI generation followed by human review, producer-validator agent models, AI-assisted output comparison, exception-based review, and risk-based validation intensity depending on criticality. The goal is not to reduce rigour. The goal is to apply rigour where it creates the most value.

Human accountability remains central. AI may generate code, perform checks or propose decisions, but programmers must still explain how results were produced, reviewed, approved and controlled. The PHC therefore supports traceability requirements that record source inputs, model or tool context, prompts where relevant, generated outputs, review evidence, approvals, change history, and the provenance of AI contribution. Inspection readiness must be designed into the workflow from the start.

The human-in-the-loop of the future cannot be a passive approver. Effective reviewers need to master clinical data and reporting expertise, understanding of analytical intent, enough programming logic to question outputs, awareness of model limitations, and the judgement to assess consequences of errors. We should create deliberate development paths within our organisations for this capability rather than assuming it will emerge naturally after routine work is automated.

The PHC has a clear vision of a future in which we need to protect and renew expertise. If AI performs more implementation work, young professionals may lose the traditional route through which they learned clinical trial data: working directly with study data, deriving variables, debugging issues, preparing outputs, responding to review comments, and seeing how downstream clinical or regulatory questions arise. Practical exposure to the clinical trial data lifecycle remains the best way to develop solid programming skills. Therefore, AI-enabled operating models must preserve meaningful exposure to real study data and real decision contexts.

The PHC encourages organisations to establish development pathways that evolve beyond traditional programming roles. Future early-career professionals could serve as clinical data and evidence analysts, combining data investigation, analytical reasoning, AI-assisted exploration, standards knowledge and scientific understanding. These roles would focus on understanding how data is generated, transformed, interpreted and used to support decisions within and across studies, indications and therapeutic areas. They would learn to challenge assumptions, investigate anomalies, assess data quality, understand analytical provenance, and translate evidence into actionable insights for stakeholders.

Critically, the role should not be reduced to reviewing and approving AI-generated outputs. A future model where professionals merely perform repetitive quality-control checks or ‘rubber-stamp’ AI results would neither develop expertise nor provide sustainable career growth. Instead, AI should augment learning by automating routine implementation activities while creating greater opportunities for data exploration, problem framing, scientific investigation, evidence synthesis, and engagement with clinical, statistical, medical and regulatory stakeholders. In this way, organisations can preserve critical expertise, accelerate onboarding and continuously renew the capabilities needed for the next generation of statistical programming leaders.

Enablers for Scale: Platforms, Data Readiness and Operating Impact

AI scale is not only a model-selection exercise. Agentic AI tools must become part of analysis and reporting platforms and the broader data science ecosystem rather than disconnected chat interfaces. Programmers should be able to invoke AI within approved workflows, with access controls, audit trails, secure data access, reusable code libraries, metadata, validated packages and governed repositories.

Data and document readiness will determine how far AI can go. Machine-readable protocols, SAPs, specifications, metadata, standards, derivation rules, controlled terminology and linked repositories are strategic assets. Ambiguous specifications, undocumented assumptions, fragmented metadata or inaccessible prior knowledge will constrain AI and create false confidence. Programming organisations should make data readiness a leadership agenda, not because AI needs clean inputs only, but because better data foundations improve quality even without AI. Every company should invest in this foundational work.

How We Get There: Capability Roadmap and Operating Actions

The roadmap of the change of role of statistical programming must be supported by leadership expectations. Managers should not only ask whether a programmer delivered outputs; they should ask whether the programmer understood the study, challenged assumptions, used AI responsibly, communicated risk, and contributed to better decisions. Individuals must also own their own development.

AI itself is not the threat. Irrelevance emerges when professionals do not adapt their capabilities to the work that remains valuable.

Action area	What organisations should promote	Example practices
AI capability building	Build data lifecycle expertise and AI literacy together.	Rotations through study setup, derivation, output review, data issue triage and regulatory/ad hoc response. AI training should use clinical trial data scenarios, not generic prompts only.
Manoeuvre-style learning	Create realistic practice environments where early-career professionals experience the lifecycle quickly and safely.	Simulation exercises, study read-out rehearsals, dry-run review of data and outputs, scenario-based debugging, and mock authority questions.
Study read-outs	Make programmers explain the science, statistics, data and limitations behind outputs.	Require programmers to present what the data shows, what assumptions matter, what could be wrong, and what decisions the output supports.
Effectiveness mindset	Mentor teams to ask why work is needed, not only how to produce it faster.	Review requests through value, decision impact, reproducibility, stakeholder need, and whether AI-enabled alternatives provide better evidence.
AI in workflow	Embed AI into approved programming flows rather than treating it as a side tool.	AI-assisted specification review, code generation, standards checking, output comparison, metadata query and review documentation inside governed SCE workflows.
Personal ownership	Make development a shared responsibility between leaders and individuals.	Each programmer maintains a development plan covering AI literacy, data lifecycle mastery, stakeholder communication and evidence interpretation skills.

Key Takeaways and Reflections

- The profession should move from output factory to data solution provider and evidence architect.
- Effectiveness over efficiency: AI will consume many efficiency roles, so our distinctive value must shift to better questions, better data, better interpretation and better decisions.
- Architect over executor: programmers should help shape workflows, data flows, questions, standards, tools and guardrails rather than only execute downstream specifications.
- AI must be embedded in governed programming flows, with traceability, provenance, review evidence and accountability.
- Data lifecycle expertise remains the foundation of programming judgement. We must preserve exposure to clinical trial data for young professionals, even as AI automates implementation work.

About the Authors

This white paper was authored by members of the EU region of the Programming Heads Council, comprised of leaders and experts responsible for statistical programming across the bio-pharmaceutical industry.

There were many common themes during our journey as we explored what AI adoption means for our profession, and we have examined how best to shape a future in which Statistical Programming leads rather than reacts.

We are an industry of talented and dedicated professionals, and we are confident that, together, we can lead the responsible adoption of AI in clinical evidence generation.