FSB FINANCIAL STABILITY BOARD

Sound Practices for Responsible Adoption of Artificial Intelligence (AI): Consultation report

Response to Consultation

Queensbury Consulting

1. Do you agree with the benefits and risks of AI adoption by financial institutions described in this report? Are there any substantive benefits or risks not covered?

These answers are grounded in practice rather than offered on behalf of an industry body: a decade as the technical authority for risk and regulatory-capital analytics, including FRTB and CVA capital as well as ISDA SIMM, used by large internationally active banks, and since then the practice of governed AI-assisted engineering.

I agree with the benefits and risks as described. In particular, I welcome the recognition in the Agentic AI Risks section that an agent can err at any of hundreds of intermediate steps and that real-time human monitoring does not scale.

One area is under-represented on the risk side: coding assistance. It is listed under productivity enhancing applications in Section 1.2, and a case study reports that an insurer used an AI code-generation tool to write six million lines of code in 2025, with around 80% adoption among developers. The risk overview gives this only one brief mention, in Section 1.3, where defective AI-generated code is noted as a source of availability and data-integrity issues. Sound Practice 11 later addresses secure coding and code review through a cyber-security lens, but the broader development-governance implications are not examined. Relative to the reported adoption and output volume, that remains under-representation.

AI assistance in development spans a spectrum, from suggestions accepted interactively by a developer, through AI generating larger changes under direct developer control, to agents planning and executing development work with substantial autonomy. The risks below apply across that spectrum and intensify toward the agentic end. Where AI-generated code enters the systems that support critical operations and regulated functions, the report's materiality logic applies to the development activity itself. An assistant drafting non-sensitive internal notes will ordinarily be low materiality and low risk, while an agent changing systems that support critical operations may warrant substantially stronger controls, and materiality and risk still need separate assessment, as the report itself notes. Four risks specific to AI-assisted development at scale are not explicitly identified.

1. Review capacity. Code volume can grow faster than genuine review capacity. Section 1.3 names inadequate human oversight generally, and Sound Practice 10 rightly asks institutions to detect rubber-stamping. The development-specific version deserves explicit

treatment, because degradation of review over code that implements regulated functions bears on materiality and risk, not only on productivity.

2. Divergence between documentation and system. As AI raises the velocity of change, the gap between what an institution's documentation says its systems do and what they do can widen over time. That erodes the institution's ability to understand, maintain, and evidence how its systems implement approved requirements, and may impede audit, incident investigation, and supervisory engagement. The report's documentation controls are not expressly applied to maintaining alignment between software and its architectural or requirements documentation.

3. Lost decision provenance. Sound Practice 10 helpfully asks institutions to log agent process, including reasoning at decision points, which is execution-level provenance. Design-level provenance is the record of why a change was made, which requirement it serves, and what was rejected. Under human authorship that record accreted imperfectly through commit messages and review threads. Under agent-assisted authorship it may be lost or become substantially weaker unless captured deliberately. Logged model reasoning is not a substitute, since the report's own Annex 2 notes that a model's stated reasoning may not reflect its actual processing; observable records such as approved specifications and plans, changes, tests, and human approvals are the dependable part.

4. Erosion of the human fallback. Sound Practice 10 includes kill-switch mechanisms and Sound Practice 12 contemplates manual continuity as one element of an exit strategy. For development, the analogous continuity question is whether the institution could keep maintaining and changing its software safely if the AI capability became unavailable or had to be suspended. That ability may decay unless it is actively maintained as AI contributes a growing share of the system. I expand on this under Question 3.

One benefit could also be made explicit. The AI-in-the-loop concept in Sound Practice 10 and the AI-powered security tooling in Sound Practice 11 point at something that matters for development. AI participation changes the economics of verification. Continuous checking of code, documentation, and their mutual consistency can be made routine, at a scope and frequency that human effort alone could never sustain.

2. Are the sound practices sufficiently comprehensive and clear to enable financial institutions' responsible AI adoption?

They are comprehensive and clear for deployed AI use cases. I suggest one structural addition and two smaller refinements.

The addition is to state that, where institutions adopt agentic development at scale, the lifecycle sound practices apply within the development lifecycle itself. Documentation and lifecycle records in the manner of Sound Practice 3, explainability and transparency in the manner of Sound Practice 8, performance and drift monitoring in the manner of Sound Practice 9, and graduated oversight in the manner of Sound Practice 10 are all practicable inside the development lifecycle today. Although the report covers development and deployment generally, and reaches secure coding and code review through Sound Practice 11, it does not explicitly examine AI-assisted software engineering as a distinct application

domain in which the lifecycle practices should themselves be applied. A sentence or short paragraph making that application explicit would close the gap.

The first refinement concerns inventory scope. The Sound Practice 3 case study describes banks tracking risk at agent level and certifying agents for use within defined boundaries. The final report could make explicit that coding agents and agentic development pipelines fall within the scope of the AI inventory, with their documentation and controls proportionate to their materiality and risk, since their outputs can become part of the institution's own software. This extends practices the case study shows institutions already have.

The second is an observation on documentation posture. Sound Practice 3 frames documentation as records kept about AI use. For agentic development, documentation is more effective when it is an authoritative input the AI itself consumes, because divergence between recorded intent and a proposed change can be detected earlier, potentially at generation time rather than only in later review.

3. **Do the sound practices strike an appropriate balance between managing risks relating to all forms of AI, and addressing some of the risks relating to emerging and new complex forms of AI, such as GenAI and agentic AI?**

The balance is right, and the realism of Sound Practice 10 about agentic oversight is particularly welcome. Approval of every agent decision does not scale, boundary-setting with monitoring is the workable frame, and accountability stays with institutions and individuals even when active monitoring is largely machine-performed. That matches my operating experience. I offer two contributions from practice, both specific to development.

The first concerns approval points. In an agentic development lifecycle, two particularly useful approval points are the specification and the implementation plan. Human approval of an explicit plan, derived from documented intent, before execution makes the approved scope more explicit and allows many deviations to be detected automatically. Execution can then be governed through automated boundaries, logging, validation, and proportionate human review rather than continuous observation of every intermediate step. And a corresponding control afterwards is verification that the record still describes the system, which helps keep the institution's technical and governance record aligned with the implemented system, rather than allowing documentation to remain a point-in-time artefact. Plan-level review concentrates scarce human attention on intent and approach rather than on raw output volume, a practical counter to the automation bias and rubber-stamping Sound Practice 10 warns about. Accountability remains with the institution through a clearly identified owner, with the approving engineer responsible for approval and acceptance decisions within that allocation.

The second extends the kill switch and the exit-strategy concept by analogy. For development, the continuity question is whether the institution could keep maintaining and changing its software safely if the AI capability became unavailable or had to be suspended, whether by the institution's choice or a provider's. That ability may decay unless it is actively maintained as AI contributes a growing share of the system. Important elements in preserving it include engineers who stay genuinely engaged through substantive review rather than rubber-stamping; a current, human-readable record of intent, decisions, and structure; and retained skills and the ability to build, test, and release without the provider.

Without these, suspending the AI may leave the codebase with engineers who have insufficient working knowledge of it, exactly when an incident or a vendor failure has made time short. The practical refinement is to treat this continuity as an arrangement that is maintained and periodically exercised, in the spirit of the scenario testing in Sound Practice 11 and the exit strategies in Sound Practice 12. A loss-of-AI-development-capability scenario, through vendor termination, model withdrawal, export-control change, or sustained outage, is testable in the way disaster recovery is tested.

4. **Are the sound practices sufficiently flexible to accommodate and address newer types of AI and responsible adoption over time?**

They are rightly technology-neutral. I suggest calling out an explicit durability mechanism. Controls anchored in artefacts and evidence (what must be documented, what must be traceable, and what must be verified) are more likely to remain usable when a model, agent, or vendor changes. Controls expressed as the configuration of a particular tool inherit that tool's lifespan, and AI tools and vendors can change rapidly. This is the development-lifecycle analogue of Sound Practice 12's expectations of exit strategies, substitutability, and data portability, and it addresses the risk the report flags of vendors changing underlying models without notice. Portable, tool-independent repository artefacts support substitutability, reduce switching costs, and form an important part of an exit strategy, and the continuity question discussed under Question 3 is the operational test of the same idea. Section 3.1 recognises that lifecycle stages are illustrative and may be adapted to an institution's environment. The same outcome-focused posture, applied to where controls are anchored, would keep the practices durable as AI evolves.

5. **Do the case studies in this report sufficiently highlight how different types of financial institutions can benefit from responsible AI adoption? Are there additional case studies for inclusion in the report? If so, please provide such case studies, particularly for nonbanks.**

Partly. The case studies are useful, but they remain weighted toward large banks, notwithstanding the insurer example, so further nonbank cases would strengthen the final report. On content, the strongest example is the agentic fraud-detection study, because it names its control, human review and approval of every proposed rule before implementation. The report also contains an enterprise-level case study describing governance around AI embedded in software-development functions. The specific case reporting six million lines of code and 80% developer adoption, however, gives no corresponding information about development controls, review arrangements, testing, traceability, or deployment acceptance. I suggest that case studies state their guardrails alongside their gains, and that the final report include one software-development case study that shows those arrangements.

6. **Do the case studies in this report provide actionable insights for financial institutions in their responsible AI adoption?**

Partly. The case studies are actionable where they describe the relevant control as well as the outcome, as the fraud-detection study does. Actionability would improve if the software-development examples did the same, for the reasons given under Question 5.

7. **Are the definitions in the glossary clear and aligned with industry sound practices, including recent developments in AI?**

   The definitions are generally clear and well aligned, and the existing entries for data drift and model drift are clear as far as they go. One suggestion concerns the report's use of "scope creep" for agentic AI. The glossary might define "agentic scope creep" as expansion of an agent's observed actions, tool use, or data access beyond its approved objectives or boundaries.

8. **Are there any comments you would like to make on this report? Please fill in.**

   By way of context, since the form has no field for it. I write as an individual practitioner rather than on behalf of an industry body. I have spent over two decades in software engineering, including a decade as the technical authority for risk and regulatory-capital analytics used by large internationally active banks, and I now work on governed AI-assisted engineering, which I have practised across two complex software projects.

   Across my answers there is a single suggestion. The report already includes AI-assisted software development in its scope, as a productivity use case, as a case study, and as a cyber risk. What the final report could add, in a sentence or a short paragraph, is that the lifecycle sound practices (documentation and provenance, explainability and transparency, performance and drift monitoring, and graduated human oversight) apply within the development lifecycle where institutions adopt agentic development for material or higher-risk systems. AI-generated code that is accepted into the codebase becomes part of the institution's own systems, including those that support critical operations, and effective application of the sound practices presupposes that an institution can describe and explain its own systems.

   The sound practices are a welcome contribution, and the non-prescriptive, proportionate framing is the right posture for a fast-moving field. I would be glad to discuss any of these points further.