Proposals for AI Use in Medical Device Software Development

A Regulatory & Technical Exploration

Gerald Rigdon | Fellow, Software Engineering

Boston Scientific Corporation | July 2026

Agenda

30-minute roadmap

01 **Background & FDA Landscape**

02 **Medical Device Software in Focus**

03 **Proposal 1 – LLM Code Generation**

04 **Proposal 2 – LLM Unit Test Generation**

05 **Proposal 3 – LLM as Independent Peer Reviewer**

06 **Summary, Risk Analysis & Conclusion**

Background: The FDA & AI

Section 01

50 Years of Evolving Oversight

- **1976** — Medical Device Amendments — FDA charged with device oversight
- **2019–21** — FDA discussion papers & proposed AI/ML frameworks
- **2021** — FDA + Health Canada + UK MHRA: 10 Guiding ML Principles
- **Oct 2023** — FDA bulletin: 171 AI/ML-enabled devices listed
- **Jan 2024** — GAO-24-106122: FDA found deficient — no AI/ML legislative guidance provided to Congress
- **Jun 2024** — FDA: ML Transparency Guiding Principles
- **Aug 2025** — FDA: Final guidance — AI-Enabled Device Software Functions

The GAO Wake-Up Call

GAO-24-106122 — January 2024

> *"The Commissioner of FDA should identify and document the specific changes to its statutory authorities that would enable FDA to take the actions it determines best to oversee AI/ML-enabled medical devices and then communicate these potential legislative changes to Congress."*

GAO Investigative Goals:

- Examine challenges & opportunities agencies face regulating emerging technologies
- Review government agency collaboration and cooperation activities
- Identify lessons from other governments' regulatory experiences

Medical Device Software in Focus

Section 02

Key Distinction: AI IN devices vs. AI FOR development

AI-Enabled Devices (Out of Scope)

AI models deployed IN the product
- Device software functions (AI-DSF)
- Patient-facing AI inference
- Covered by FDA 2025 Draft Guidance [11]

≠

AI in Development Tools (THIS PAPER)

AI used TO BUILD the device software
- Code generation
- Unit test generation
- Peer review assistance
- Direct Use Software (DUS)

(1) Identifying the Intended Use

The regulation requires manufacturers to validate software **that is used as part of production or the quality management system** for its intended use (see Subclauses 4.1.6, 7.5.6, and 7.6 of ISO 13485). This includes various cloud computing models related to computerized systems, such as IaaS, PaaS, and SaaS.

To determine whether the requirement for validation applies, manufacturers must determine whether the software is or will be used as part of production or the quality management system (whether directly or to support production or the quality management system).

Software with the following intended uses is considered to be used **directly** as part of production or the quality management system:

- Software intended for automating production processes, inspection, testing, or the collection and processing of production data; and
- Software intended for automating quality management system processes, collection and processing of quality management system data, or maintaining a quality record established under applicable quality management system obligations.

Software with the following intended uses is considered to be used to **support** production or the quality management system:

- Software intended for use as development tools that test or monitor software systems or that automate testing activities for the software used as part of production or the quality management system, such as those used for developing and running scripts or software embedded in the production equipment (e.g., firmware); and
- Software intended for automating general record-keeping for production or the quality management system that is not part of the quality record.

2026 Computer Software Assurance Guidance [13]

FDA CSA Framework: Is it DUS or SUS?

2026 Computer Software Assurance Guidance [13]

Step 1: Classify Intended Use

Category 1: Direct Use Software (DUS)

- Automating production or inspection
- Automating QMS processes
- Maintaining quality records

Category 2: Supporting Use Software (SUS)

- Development tools
- Test/monitor software systems
- General record-keeping automation

Step 2: Process Risk?

High Process Risk
Failure directly impacts production/QMS
→ Rigorous V&V required

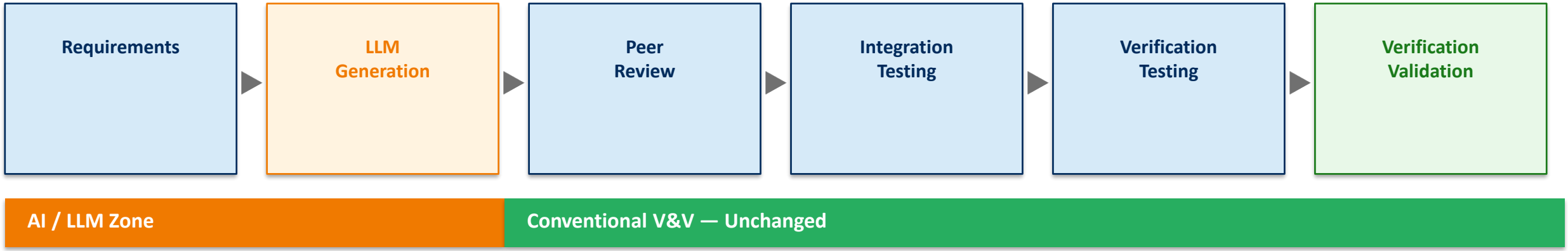
Not High Process Risk
Reduced but still proportional
assurance activities required

This paper takes the conservative approach: High Process Risk – DUS

Keeping Validation Intact — Upstream vs. Downstream

Proposals 1 & 2 foundation

The core principle: LLM involvement is UPSTREAM. Verification & Validation remain UNCHANGED downstream.



Key insight: The same software verification and validation procedures operate identically regardless of whether upstream code was written by a human engineer or an LLM.

> *"Assume the most conservative approach: High Process Risk – DUS. The proposals do not fundamentally alter the medical device software development process."*

Proposal 1: LLM Code Generation

Section 03

01 Use LLMs to generate medical device implementation code

The Central Question:

> *"If the implementation satisfies peer review, passes static analysis, and passes all tests — does the origin of the code matter?"*

Process Flow (Figure 4 in paper):

Benefits:

- Satisfies all downstream requirements
- Significant reduction in coding development time
- Frees human engineers for higher-value tasks

Proposal 2: LLM Unit Test Generation

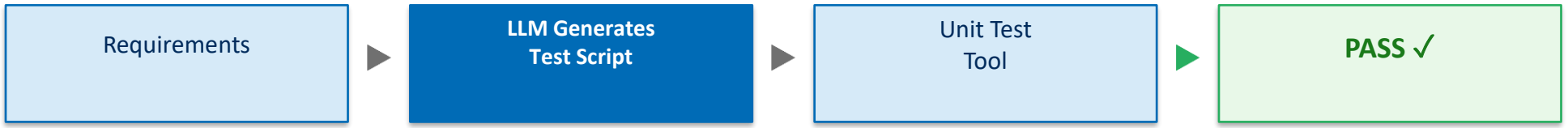
Section 04

02 **Use LLMs to generate unit test scripts, refined via automated tooling loops**

Narrowed Scope — Unit Testing Only:

- Grey-box Integration Testing → remains conventional (no LLM)
- Black-box Verification Testing → remains conventional (no LLM)
- Proposal 1 validation chain remains intact

Reflection Loop (Figure 5 in paper):



- Tool enforces coverage & syntax requirements regardless of script author
- Non-deterministic generation is offset by deterministic tool validation
- Reduces unit test development time; frees engineers for higher-value work

Proposal 3: LLM as Independent Peer Reviewer

Section 05

03 **Deploy LLMs in the independent reviewer role of the peer review process**

FDA 2002 Validation Guidance [12] — Independence of Review:

"Validation activities should be conducted using the basic quality assurance precept of 'independence of review.' Self-validation is extremely difficult. When possible, an independent evaluation is always better, especially for higher risk applications."

Questions This Raises:

- Can an LLM trained on vast software engineering principles serve as an independent reviewer?
- How would an LLM reviewer be validated — and does one validate a human reviewer the same way?
- Is an LLM not unlike a static analysis tool — recommended by FDA precisely because computers excel at tasks the human brain doesn't?

Proposal 3: Review Process with LLM

Human SME stays in the loop

Review Process (Figure 6 in paper):

Arm 1 — Human SME

- Subject matter expert review
- Domain knowledge & context
- Coding standard compliance
- Remains mandatory & unchanged

+

Arm 2 — LLM Independent Reviewer

- RAG over requirements, design docs, DDC constraints
- Architectural analysis
- Logic & edge case detection
- Integration analysis
- Replaces independent human reviewer

Combined Output → Peer Review Report ✓ Human SME anchors the loop

LLM = Next-Generation Static Analysis Tool

The analogy that makes Proposal 3 defensible

The FDA recommends static analysis tools precisely because computers exceed human capability for certain tasks.

Traditional Static Analysis

- MISRA C compliance
- Naming conventions
- Complexity metrics
- Memory usage patterns
- Rule-based enforcement

Complementary

not competing

LLM Analysis (with RAG)

- Architectural soundness
- Domain logic correctness
- Cross-component integration
- Requirement traceability
- Context-aware observations

Both arms are valuable. Static analysis catches rule violations; LLM catches domain, architectural and integration issues that rules cannot codify.

Addressing Objections to Proposal 3

Anticipating the hard questions

Objection	Response
Proposal 1 uses LLMs to write code AND Proposal 3 uses LLMs to review it — isn't that circular?	Peer review is anchored by a human SME. Static analysis tools enforce coding standards independently. The LLM reviewer adds a second opinion, not self-review.
How do you validate the LLM reviewer?	Option: Evaluate LLM review validity via a separate human-reviewed process. Or contend code generation is not fully autonomous — human engineers vet during development.
Non-determinism is a concern.	The same concern applies to assigning any human reviewer. The LLM's base training on a vast corpus of software engineering is a defensible qualification.

Risk-Based Analysis Summary

Figures 7 & 8 in paper — Conservative posture

All proposals evaluated under: High Process Risk – DUS (most conservative classification)

Proposal 1
Code Generation

- SUS classification arguable
- Downstream V&V unchanged
- LLM is upstream input only
- Human engineer reviews output

Proposal 2
Unit Test Gen

- Narrow scope: unit tests only
- Tool enforces pass/fail criteria
- Integration & Verification: conventional
- Reflection loop self-corrects

Proposal 3
Peer Review

- Human SME anchors process
- LLM = validated tool (not person)
- Complementary to static analysis
- Independence of review maintained

Summary

Section 06

Three proposals for LLM use in medical device software development — all defensible under existing FDA risk-based frameworks, ISO 13485 alignment, and the 2026 CSA guidance.

■ **Proposal 1**	**LLM Code Generation**	Upstream only; validated by full downstream V&V pipeline
■ **Proposal 2**	**LLM Unit Test Generation**	Narrow scope; tool-enforced correctness via reflection loop
■ **Proposal 3**	**LLM Peer Review**	Human SME anchors; LLM as RAG-augmented static analysis complement

These proposals improve efficiency and reduce resource strain while remaining compliant through documented processes, targeted internal checks, and risk-based validation.

Conclusion & Path Forward

2026 and beyond

There is little doubt that AI will challenge the FDA's balancing act of protecting public health without impeding innovation in ways that are not easily anticipated.

Industry Engagement	As engineers in medical device development, active participation in regulatory dialogue is essential — not optional.
Research Roadmap	The three proposals presented will be the focus of upcoming MDM research in 2026, with a goal to publish findings.
Democratic Policymaking	As citizens and patients, we must engage in the formation of policy around AI in healthcare systems.
Next Steps	Validate Proposal findings through structured internal studies; contribute to FDA guidance evolution.

The goal: publish MDM research findings for these three proposals to further the conversation on the use of AI in medical device software development.

References

Selected key sources

[1] GAO-24-106122 Report to Congressional Requesters (Jan 2024) — gao.gov

[4] Good ML Practice for Medical Device Development: 10 Guiding Principles (2021) — fda.gov

[7] Marketing Submission Recommendations for AI-Enabled Device Software Functions (Aug 2025) — fda.gov

[10] IMDRF: Good ML Practice Guiding Principles (Jan 2025) — imdrf.org

[11] AI-Enabled Device Software Functions: Lifecycle Mgmt & Market Submission. Draft Guidance (Jan 2025) — fda.gov

[12] General Principles of Software Validation (Jan 2002) — fda.gov

[13] Computer Software Assurance for Production & Quality Management System Software (Feb 2026) — fda.gov

[14] Medical Devices; Quality System Regulation Amendments (Feb 2022) — federalregister.gov

[15] ISO 13485:2016 — Medical Devices Quality Management Systems

Proposal Paper

Proposals for AI Use in Medical Device Software Development

Gerald Rigdon
Fellow, Software Engineering
Boston Scientific Corporation
St. Paul, USA
gerald.rigdon@bsci.com
May 2026

Abstract

Since the Medical Device Amendments were added to the Federal Drug and Cosmetic Act in 1976, the FDA has been charged with the oversight of an industry with substantial and ever-emerging technological change over the past 50 years. Medical devices have evolved from mechanical devices to sophisticated electro-mechanical products operated by complex software. With the advent of connected medical devices in recent years, the FDA has had to deal with a new era where devices are now subject to cybersecurity breaches. And now on the heels of responses to pressing security concerns, the agency faces the expansive frontier of AI. This paper offers an MDM perspective and identifies important proposals for the use of AI in medical device software development.

Key Terms:

AI – Artificial Intelligence
DUS – Direct Use Software
FDA – Food and Drug Administration
LLM – Large Language Model
MDM – Medical Device Manufacturer
ML – Machine Learning
SME – Subject Matter Expert
SUS – Supporting Use Software

Thank You

Questions & Discussion

Gerald Rigdon | Fellow, Software Engineering

Boston Scientific Corporation

gerald.rigdon@bsci.com