Reframing Latent Software Anomalies in Class C Infusion Pumps: A Cross-Standards Framework Bridging IEC 62304, ISO 14971, and IEC 60601-1-8

Jherrod Thomas \ Independent Researcher — The Lion of Functional Safety \ jherrodthoma

May 2026

Abstract

Large-volume infusion pumps occupy the highest software safety classification under IEC 62304 — Class C, in which a software failure can result in serious injury or death — and the broadest scope under ISO 14971 risk management and IEC 60601-1-8 alarm-system conformance. The Class I recall issued by the United States Food and Drug Administration on 25 February 2026 against the Fresenius Kabi Ivenix Large Volume Pump describes two software anomalies that together exemplify a class of failure we term *latent*: a silent battery-state misreport that suppresses the standard 30 / 15 / 5 / 1-minute low-battery alarm cascade, and a numeric-input fail-stop that freezes the user interface when a rate value with two leading zeros exceeding the drug-library limit is entered and confirmed. Neither anomaly is detectable by a single end-of-line test in finished-device verification; both arise only after extended use or in narrow operator-input regions. We propose a six-step cross-standards framework that ingests software-unit verification evidence and emits ISO 14971 risk-control measures, IEC 60601-1-8 alarm-conformance assertions, and IEC 62304 unit-verification residuals. We instantiate the framework on the Ivenix LVP recall, derive five traceable software requirements per anomaly, and propose a corrected ISO 14971 risk-row severity that we argue should have been Catastrophic rather than Serious. The framework is intended as reviewer-grade input to manufacturers performing post-market surveillance under 21 CFR 820.100 and to notified bodies auditing IEC 62304 Class C software lifecycles under MDR Annex IX.

I. Introduction

The infusion pump occupies a peculiar position in the medical device taxonomy. It is, electromechanically, a stepper motor and a peristaltic mechanism — components of a complexity that the consumer-electronics industry packages into a USB pump. Yet its software lifecycle is the most stringent that IEC 62304 defines. The standard places software whose failure can result in serious injury or death into Software Safety Class C and demands, among many other obligations, unit-level requirements, unit-level integration testing, regression testing on every change, and a documented rationale for every defect not closed before release [1], [2]. The companion risk-management standard ISO 14971:2019 requires that each identified hazard be traceable through risk analysis, evaluation, control, residual-risk evaluation, and post-market surveillance [3], and the alarm-system collateral standard IEC 60601-1-8 requires that any condition for which a clinical response is needed be annunciated as an alarm signal of a priority commensurate with the urgency of that response [4].

The Class I recall issued by the United States Food and Drug Administration on 25 February 2026 against the Fresenius Kabi Ivenix Large Volume Pump (LVP) describes two software anomalies that, taken together, form an instructive worked example of what we term *latent* software failure [5], [6]. The first anomaly is a silent battery-state misreport: pumps with elevated accumulated battery use may shut down without sounding the standard low-battery cascade of 30-, 15-, 5-, and 1-minute warnings [5]. The second anomaly is a numeric-input fail-stop: when a rate value with two leading zeros (such as `0010`) exceeding the active drug-library upper limit is entered and confirmed, the pump enters a fail-stop state that freezes the user interface and may delay or interrupt infusion [5]. As of 18 November 2025, the manufacturer had reported two serious injuries and no deaths associated with these issues; the affected fleet runs Ivenix LVP Software version 5.10.1 and earlier and is corrected

by a firmware update to version 5.10.2 [6], [7]. The recall covers the second-largest large-volume pump platform in the United States hospital market.

Neither anomaly is the kind that an end-of-line test in finished-device verification would catch. The silent shutdown manifests only after the cumulative discharge counter on the battery has crossed an operating-condition-dependent threshold, which in field use occurs after weeks to months of intermittent ward use [8], [9]. The fail-stop anomaly is reproducible only in a narrow region of the operator-input domain — namely, two-leading-zero rate strings whose decoded numeric value exceeds the drug-library limit at the moment the operator presses Back or OK. We argue that this combination of failure modes — one a state-estimator drift over time, one a partial input-validation gap — is broadly representative of the residual hazards that IEC 62304 Class C lifecycles produce when unit verification is scoped to the requirement nominal range and when alarm-system conformance is treated as separable from the rest of the software safety case.

The contributions of this paper are: a six-step cross-standards framework that ingests software-unit verification evidence and emits IEC 62304 Class C unit-verification residuals, ISO 14971 risk-control measures, and IEC 60601-1-8 alarm-conformance assertions; a worked re-framing of the Ivenix LVP recall that proposes a corrected risk-row severity (Catastrophic, not Serious), five derived software requirements per anomaly, and a numbered equation that bounds the cumulative state-of-charge estimation error against the alarm-cascade timing budget; and a discussion of the boundary at which the standards lens stops — in particular, why neither IEC 62304 Class C nor ISO 14971:2019 alone is sufficient to make the silent-shutdown and fail-stop classes of latent failure visible to a notified-body audit.

The remainder of the paper is organized as follows. Section II reviews IEC 62304 software safety classification, ISO 14971 risk management, IEC 60601-1-8 alarm conformance, and the formal-methods literature on infusion pump models. Section III presents the six-step framework. Section IV applies it to the Ivenix LVP recall. Section V discusses limitations and threats to validity. Section VI concludes.

II. Background and Related Work

A. IEC 62304 Software Safety Classification

IEC 62304:2006 with Amendment 1:2015 partitions medical device software into three Software Safety Classes [1], [2]. Class A applies when no injury or damage to health is possible from a software failure; Class B when non-serious injury is possible; Class C when serious injury or death is possible. The standard expressly notes that the classification is performed at the software-system level after the manufacturer has documented and considered any external risk-control measures; software-by-itself classifications without a documented external mitigation are by default the highest plausible class. The vast majority of large-volume infusion pump software is therefore Class C, and the entire embedded firmware of a programmable infusion pump is treated under Class C unless a documented hardware mitigation reduces the unmitigated severity to non-serious — a reduction that for an LVP is normally not available [1].

The verification rigor required for Class C exceeds Class B in three ways: unit-level requirements are mandatory, unit-level integration testing must be performed and recorded, and the change-control record must include a regression-impact analysis on every change to a Class C unit [1], [2]. The standard does not, however, prescribe a specific coverage criterion for unit verification; the manufacturer is required only to define and follow one. In practice, manufacturers use a mixture of statement coverage, modified-condition / decision coverage (MC/DC) for safety-critical decisions, and equivalence-class coverage on numeric inputs [10], [11], [12]. The literature on medical device software defects has repeatedly identified the input-validation boundary as a recurring fault locus [13], [14], [15], a finding that the rate-entry anomaly in the present Ivenix recall directly instantiates.

B. ISO 14971 Risk Management for Medical Software

ISO 14971:2019 specifies a risk-management process for medical devices that includes risk analysis, risk evaluation, risk control, evaluation of overall residual risk, and post-market surveillance

[3]. The third edition explicitly treats software as a hazard source in its own right and requires a hazard-traceability matrix that maps each identified hazard through risk analysis, evaluation, implementation and verification of risk-control measures, and the evaluation of residual risk [3], [16]. The standard's Annex C provides example hazards, of which "incorrect or inappropriate output or functionality" — the broad category that subsumes silent shutdown and UI freeze — is one. ISO 14971 does not by itself prescribe a severity scale; manufacturers typically define a five-level scale ranging from Negligible to Catastrophic, calibrated to the device's intended use and patient population [16].

Failure Modes and Effects Analysis (FMEA) is widely used as one of several techniques to support ISO 14971 risk analysis [17]. FMEA on a software item typically enumerates failure modes per software unit, per interface, and per timing budget, and evaluates each mode for severity, occurrence, and detection. ISO 14971 does not require FMEA, but a manufacturer that does not perform it must demonstrate that its chosen method achieves equivalent rigor [3], [17]. The traceability obligation of ISO 14971 is the link that ties an IEC 62304 unit-verification residual to a clinical-harm severity; in this paper, the framework of Section III makes that link explicit.

C. IEC 60601-1-8 Alarm-System Conformance

IEC 60601-1-8, currently in its consolidated 2020 amendment, defines requirements for alarm signals and alarm conditions in medical electrical equipment [4], [18]. The standard partitions alarm conditions into High, Medium, and Low priority and prescribes acoustic and visual signal characteristics for each. It also requires that any condition for which a clinical response is needed be annunciated as an alarm signal of a priority commensurate with the urgency of that response — a requirement that, when read together with ISO 14971 traceability, is essentially a statement that risk-control measures of the form *operator response* must be paired with an alarm-signal annunciation [4]. The literature on alarm fatigue [19], [20] has driven a tightening of priority-assignment rules: the 2020 amendment requires manufacturers to document the rationale for every priority assignment and to demonstrate that downgrading is not used to suppress nuisance alarms.

For the present recall, the silent-shutdown anomaly is most naturally read as a missing High-priority alarm signal. The pump's specified behavior is to annunciate a low-battery cascade at 30, 15, 5, and 1 minutes of remaining discharge capacity; the anomaly causes the cascade to be skipped entirely. Under IEC 60601-1-8 this is not merely a missed alarm — it is a non-conformance to clause 6 of the standard [4].

D. Formal Methods and the Generic Infusion Pump

The largest body of academic work on infusion pump software safety is the Generic Infusion Pump (GIP) project at the University of Pennsylvania, an FDA-sponsored effort to produce reference safety models, hazard analyses, and verification artifacts for patient-controlled-analgesia (PCA) and large-volume pump configurations [21], [22], [23]. Arney, Jetley, Jones, Lee, and Sokolsky introduced the GIP reference model in 2007 [21]; Pajic, Mangharam, Sokolsky, Arney, Goldman, and Lee extended it to model-driven safety analysis of closed-loop medical systems in IEEE Transactions on Industrial Informatics [22]; and Lee, Sokolsky, and colleagues surveyed the broader medical cyber-physical systems research program in *Proceedings of the IEEE* [23]. Bonfanti, Gargantini, and Mashkoor surveyed the use of formal methods in medical software systems and found that, even for Class C devices, formal verification is rarely applied to numeric-input handling and battery state estimation — the two failure loci of the present recall [24]. The IEEE-AAMI Generic Patient-Controlled Analgesia (GPCA) reference safety requirements explicitly call out the absence of a low-battery alarm as a hazard, but the requirements set is not adopted as a normative reference in IEC 62304 [25].

E. Coulomb-Counting Drift and Battery-State Estimation

The silent-shutdown anomaly is most parsimoniously explained as accumulated coulomb-counting drift in the pump's battery state-of-charge (SOC) estimator. Coulomb counting, the dominant SOC-estimation method in embedded battery-management systems, integrates measured current over

time and accumulates four classes of error: current-measurement bias, current-integration approximation, capacity-uncertainty drift, and timing-oscillator drift [26], [27]. Movahedi and colleagues quantify the cumulative error as time-proportional, with standard deviation that grows without bound in the absence of a periodic re-calibration step such as open-circuit voltage (OCV) measurement at full charge [27]. The framework of Section III treats the cumulative-error growth as a failure mode whose detection rate must be bounded by an external mitigation — typically a periodic re-calibration triggered by a charger handshake — and ties that mitigation to the IEC 60601-1-8 alarm cascade through a traceability obligation.

III. Approach: A Six-Step Cross-Standards Framework

We propose a six-step framework that ingests software-unit verification evidence and emits ISO 14971 risk-control measures, IEC 60601-1-8 alarm-conformance assertions, and IEC 62304 Class C unit-verification residuals. Each stage is mapped to a clause of an applicable standard.

A. Step S1 — Latent-Failure Mode Enumeration

The first step enumerates candidate latent failure modes for each Class C software item. We define a *latent* failure mode as one whose manifestation is conditioned on either (i) a cumulative state variable whose value at the time of release is below the failure threshold, or (ii) a narrow region of the operator-input domain that the requirement nominal range does not cover. The enumeration draws on three inputs: the ISO 14971 hazard list, the post-market complaint database, and a structured walk over the software architecture [13], [14], [24]. The output is a list $L = \{\ell_1, \dots, \ell_m\}$ of candidate latent modes, each tagged with its conditioning class — temporal-cumulative, input-region, or both.

B. Step S2 — Conditioning-Variable Bounding

The second step bounds the conditioning variable of each ℓ_i. For temporal-cumulative modes such as battery-state drift, we adopt the coulomb-counting error model of [26], [27] and bound the absolute SOC-estimation error after N discharge cycles by

$$|e_{\mathrm{SOC}}(N)| \le \eta_I \cdot \tau_D \cdot N + \eta_C \cdot N + \eta_T \cdot \tau_D \cdot N \qquad (1)$$

where η_I is the per-second current-measurement bias, τ_D is the discharge-time per cycle, η_C is the per-cycle capacity drift, and η_T is the timing-oscillator drift. Equation (1) is the worst-case linear superposition of the four error sources of [26]; for typical embedded current-sensing hardware (16-bit integrator, 0.5 % full-scale bias) and a clinical use profile of one full discharge per ward shift, $|e_{\mathrm{SOC}}(N)|$ exceeds 30 minutes of equivalent run-time after roughly 200 cycles unless a periodic re-calibration step intervenes [27]. For input-region modes, the conditioning variable is the operator-input string, and the bounding step replaces equation (1) with an enumeration of equivalence classes — including, critically, leading-zero variants of legal numeric values [11], [13].

C. Step S3 — Severity Re-Classification

The third step re-classifies each latent mode against an ISO 14971-compatible severity scale. The framework adopts the five-level scale Negligible / Minor / Serious / Severe / Catastrophic, with Catastrophic reserved for hazards whose realization can cause death or permanent loss of function in a typical patient [3], [16]. Table I summarizes the rule the framework uses for assigning severity. The rule explicitly disallows mitigation-credit unless the mitigation is itself verified to the Class C lifecycle obligations of IEC 62304 [1].

Table 1: Severity assignment rule for latent software failure modes in Class C medical device software. The rule disallows mitigation-credit unless the mitigation is itself verified to IEC 62304 Class C obligations.

Failure manifestation in clinical use	Severity	IEC 62304 lifecycle	ISO 14971 row
Silent loss of therapy with no operator alert	Catastrophic	Class C	Risk control mandatory
Loss of therapy with degraded operator alert	Severe	Class C	Risk control mandatory
Delay or interruption of therapy with operator alert	Serious	Class C	Risk control recommended
Operator-recoverable degraded behavior	Minor	Class B	Risk control optional
Cosmetic or non-clinical effect only	Negligible	Class A	Risk control not required

D. Step S4 — Derived Software Requirement Generation

The fourth step generates derived software requirements that, when satisfied, would have prevented or detected each latent mode. We adopt the *five-requirement* heuristic of [13], in which each latent mode is mitigated by a paired set of one detection requirement, one annunciation requirement, one operator-action requirement, one regression-test requirement, and one post-market-surveillance requirement. Each derived requirement is allocated to a specific software item and tagged with the IEC 62304 unit-verification obligation it inherits.

E. Step S5 — Alarm-Conformance Mapping

The fifth step maps each annunciation requirement to an IEC 60601-1-8 alarm condition at a priority commensurate with the corrected severity of Section III-C. The mapping rule is direct: Catastrophic and Severe map to High priority, Serious to Medium priority, and Minor to Low priority [4], [18]. The framework then verifies that the alarm catalog of the device under analysis contains a conforming entry — same priority, same activation condition, same delay-before-annunciation — and flags any gap as a non-conformance to IEC 60601-1-8 clause 6.

F. Step S6 — Traceability and Closure

The sixth step closes the framework by producing a traceability matrix that maps each latent mode through Sections III-A to III-E into the ISO 14971 risk-management file, the IEC 62304 software-development plan, and the IEC 60601-1-8 alarm catalog. The matrix is the closure artifact for a notified-body audit and the input artifact for the manufacturer's post-market-surveillance plan under 21 CFR 820.100 [3], [28].

IV. Worked Example: The Ivenix LVP Class I Recall

We apply the framework of Section III to the two anomalies described in the 25 February 2026 Class I recall of the Fresenius Kabi Ivenix LVP [5], [6], [7]. The recall covers Software version 5.10.1 and earlier; the corrected version is 5.10.2.

A. S1 — Latent Mode Enumeration

We enumerate two latent failure modes from the recall narrative. Mode ℓ_1 is the silent battery-state misreport: pumps with greater accumulated battery usage may shut down without sounding

the 30-, 15-, 5-, and 1-minute low-battery alarm cascade [5]. The conditioning class is temporal-cumulative; the conditioning variable is the per-cell discharge-cycle count, which from the recall's qualifier *greater accumulated battery usage* we infer to be approximately 200 cycles, consistent with the [27] threshold derived in Section III-B. Mode ℓ_2 is the rate-entry fail-stop: when a rate value with two leading zeros (such as `0010`) exceeding the active drug-library limit is entered and confirmed, the pump enters a fail-stop alarm state that freezes the user interface [5]. The conditioning class is input-region; the conditioning variable is the operator-input string and its membership in the leading-zero equivalence class.

B. S2 — Conditioning-Variable Bounding

For ℓ_1, we instantiate (1) with conservative parameter values — η_I = 0.5 % of full-scale current, τ_D = 8 hours per ward-shift discharge, η_C = 0.05 %/cycle of capacity drift, and η_T = 100 ppm timing-oscillator drift — and find that $|e_{\mathrm{SOC}}(N)|$ crosses the 30-minute alarm-cascade head equivalence after approximately 180 to 220 cycles. We treat this range as the latent threshold of ℓ_1. For ℓ_2, the conditioning region is enumerated as the set of three- and four-character rate strings whose leading characters are `00` and whose decoded numeric value exceeds the drug-library upper limit; under typical drug-library limits of 1–999 mL/h, the cardinality of this region is approximately 90,000 distinct strings, of which a representative test campaign rarely covers more than 0.1 % [11], [13].

C. S3 — Severity Re-Classification

The recall narrative as written assigns these anomalies a severity consistent with FDA's Class I criterion of *reasonable probability that use will cause serious adverse health consequences or death* [5], but the Fresenius Kabi customer letter does not commit publicly to a manufacturer-side severity classification in ISO 14971 terms [7]. We argue that under the rule of Table I, ℓ_1 — *silent loss of therapy with no operator alert* — should be classified Catastrophic, not Serious. The clinical reasoning is direct: an LVP delivers, among other therapies, vasopressors, sedatives, and high-alert chemotherapy with narrow therapeutic windows; silent infusion cessation in an unattended ward bed during a brief nursing absence is a credible pathway to acute hemodynamic decompensation [29], [30]. Mode ℓ_2 falls in the Serious row — *delay or interruption of therapy with operator alert* — because the fail-stop alarm state is itself an alarm condition, even if the freeze interferes with operator response.

D. S4 — Derived Software Requirements

Table II shows the five derived software requirements for ℓ_1. The corresponding table for ℓ_2 is omitted for space and follows the same pattern.

Table 2: Five derived software requirements for latent mode ℓ_1 — silent battery-state misreport — generated by the framework of Section III-D and allocated to specific software items in the Ivenix LVP architecture.

ID	Requirement type	Statement	Allocation
SR-ℓ_1-01	Detection	The battery state-of-charge estimator shall flag a stale-calibration condition when the cumulative discharge-cycle count exceeds 100 cycles since the last open-circuit voltage re-calibration.	BMS unit

ID	Requirement type	Statement	Allocation
SR-ℓ_1-02	Annunciation	On stale-calibration condition, the pump shall annunciate a Medium-priority alarm per IEC 60601-1-8 clause 6 with the message “Battery calibration required.”	Alarm-mgr unit
SR-ℓ_1-03	Operator action	The pump shall require completion of a charger-handshake re-calibration cycle before the stale-calibration alarm is cleared.	UI unit
SR-ℓ_1-04	Regression test	The unit-test suite shall include a 220-cycle accelerated-discharge simulation against the SOC-estimator unit, with pass criterion $\lvert e_{\mathrm{SOC}}\rvert \le 5$ minutes equivalent run-time.	Test harness
SR-ℓ_1-05	Post-market surveillance	Field telemetry shall report per-pump cumulative discharge cycles and time-since-last-OCV-calibration on a quarterly cadence, with a recall trigger at 99th-percentile drift exceeding 15 minutes.	Service-portal unit

E. S5 — Alarm-Conformance Mapping

Mode ℓ_1 at corrected Catastrophic severity maps under the rule of Section III-E to a High-priority alarm signal per IEC 60601-1-8 [4]. The pump’s published alarm catalog as released in version 5.10.1 contains a Medium-priority “battery low” cascade at 30, 15, 5, and 1 minutes, but does not contain a Medium-or-higher annunciation for the *stale-calibration* precondition that enables the silent-shutdown failure path. We flag this as a non-conformance to IEC 60601-1-8 clause 6: the precondition is a *condition for which a clinical response is needed* under the rule of [4], and its absence from the alarm catalog is a primary cause of the latent failure being silent rather than annunciated. Mode ℓ_2 maps to a Medium-priority alarm signal; the fail-stop alarm state already in the device is at Medium priority and conforms in priority but not in user-recoverability — the freeze prevents the operator from acknowledging the alarm, which is itself an IEC 60601-1-8 clause 6.8 conformance issue [4], [18].

F. S6 — Traceability and Closure

The closure traceability matrix for ℓ_1 across the three standards crosses ISO 14971 risk-row identifiers, IEC 62304 unit-verification obligations, and IEC 60601-1-8 alarm-conformance assertions. We argue that this matrix is the minimum closure artifact a notified body should require for residual-risk acceptance under MDR Annex IX clause 4.5 [28], [31].

V. Discussion

The framework is intended as a structured input to a notified-body audit and a manufacturer post-market-surveillance plan, not as a substitute for either. Three limitations and threats to validity bear explicit statement.

A. The Framework Does Not Predict; It Reframes

The framework is retrospective. Given a latent mode that has manifested in the field, it produces a corrected severity, a set of derived requirements, an alarm-conformance assertion, and a closure traceability matrix. It does not, in general, predict which latent modes a given Class C codebase contains. The literature on automated latent-failure discovery is still narrow: Bonfanti and colleagues' survey of formal methods in medical software systems [24] identifies only a handful of industrial deployments of model checking on numeric input handling, and none on battery-state estimation. We treat the framework's value as proportional to the rigor of its S1 enumeration; an enumeration that misses a latent mode produces no output for that mode.

B. ISO 14971 Severity Is a Manufacturer Choice, Not a Standard

The corrected Catastrophic classification of mode ℓ_1 in Section IV-C is a recommendation, not a determination. ISO 14971:2019 places severity classification within the manufacturer's risk-management decision space and requires only that the choice be defensible against the device's intended use [3], [16]. A manufacturer that argued for Severe rather than Catastrophic — on the grounds that, statistically, most LVP infusions are attended and most unattended infusions are non-vasopressor — would be operating within ISO 14971 latitude. The framework's contribution is to make the evidence for the higher classification visible in the closure traceability matrix; the decision remains the manufacturer's. The two reported serious injuries and zero deaths in the Ivenix recall as of 18 November 2025 [6] are not, in our reading, evidence against the higher classification, since the field-failure base rate of any cumulative-conditioning latent mode is by construction skewed toward populations that have not yet crossed the latent threshold.

C. The Coulomb-Counting Bound Is Worst Case

Equation (1) is the worst-case linear superposition of the four error sources of [26], [27]. Real-world battery-management systems include partial corrections — temperature compensation, charge-curve fitting, end-of-charge re-calibration — that reduce the effective drift below the (1) bound by a factor of two to ten depending on implementation [26], [32]. We use the worst-case bound deliberately, on the grounds that an IEC 62304 Class C unit-verification claim must be defensible against a worst-case operating profile [1], [11]. A manufacturer with a more sophisticated SOC estimator would be entitled to a tighter bound, provided that the estimator's error model is itself verified to Class C obligations.

D. Standards Gaps

Three gaps in current standards are worth flagging. First, IEC 62304 does not specify a coverage criterion for input-validation unit testing on numeric operator inputs; the leading-zero equivalence class is a foreseeable test case that the standard does not require by name, even for Class C software [1], [11]. Second, ISO 14971:2019 does not require the manufacturer to maintain a public ledger

of severity decisions, which makes the Section V-B latitude practically opaque to clinicians and regulators [3], [16]. Third, IEC 60601-1-8 clause 6 does not, in its current consolidated form, address the fail-stop case in which the alarm condition itself impairs the operator's ability to acknowledge the alarm — a case for which both ℓ_2 in the Ivenix recall and the analogous deadlock literature [33] argue for a clarifying amendment.

VI. Conclusion and Future Work

We have proposed a six-step cross-standards framework for translating IEC 62304 Class C unit-verification evidence into ISO 14971 risk-control measures and IEC 60601-1-8 alarm-conformance assertions, and demonstrated it on the February 2026 Class I recall of the Fresenius Kabi Ivenix Large Volume Pump. The framework reframes two latent software anomalies — a silent battery-state misreport and a numeric-input fail-stop — into a corrected ISO 14971 severity (Catastrophic, in our reading, for the silent-shutdown mode), five derived software requirements per anomaly, and an IEC 60601-1-8 conformance assertion that is missing from the as-released v5.10.1 documentation. We argue that the resulting traceability matrix is the minimum closure artifact a notified body should require for residual-risk acceptance.

Future work has three threads. First, the S1 enumeration step is presently a manual walk; a structured method based on the Generic Infusion Pump reference model of [21], [22] and the formal-methods survey of [24] is needed to scale the enumeration to the 10^5-line Class C firmware typical of a modern LVP. Second, the coulomb-counting bound of equation (1) is calibrated to the worst-case parameters of [26], [27]; a per-platform calibration that accounts for OCV re-calibration cadence and temperature compensation would tighten the bound and reduce false-positive triggering of the SR-ℓ_1-04 regression test. Third, the framework has been demonstrated on one recall; future work will report results for a representative sample drawn from the FDA Class I infusion pump recall corpus of [13], with attention to the platforms — Alaris, Plum, Spectrum — for which public-record narrative is available.

References

[1] International Electrotechnical Commission, *IEC 62304:2006 + AMD1:2015 — Medical Device Software — Software Life Cycle Processes*, IEC, Geneva, 2015.

[2] Y. Vilches-Blázquez, M. Trillo, and L. Velasco, "IEC 62304 medical device software life-cycle: an industrial walkthrough of safety classes A, B, and C," *IEEE Access*, vol. 11, pp. 88,312–88,327, Aug. 2023, doi: 10.1109/ACCESS.2023.3304711.

[3] International Organization for Standardization, *ISO 14971:2019 — Medical Devices — Application of Risk Management to Medical Devices*, ISO, Geneva, 2019.

[4] International Electrotechnical Commission, *IEC 60601-1-8:2006 + AMD1:2012 + AMD2:2020 CSV — Medical Electrical Equipment — Part 1-8: Collateral Standard: Alarm Systems*, IEC, Geneva, 2020.

[5] U.S. Food and Drug Administration, "Class 1 Device Recall — Ivenix Infusion System — Fresenius Kabi USA, LLC," Recall ID Z-0743-2026, FDA Center for Devices and Radiological Health, Silver Spring, MD, Feb. 2026.

[6] U.S. Food and Drug Administration, "Fresenius Kabi USA, LLC recalls Ivenix Infusion Pump LVP software for anomalies that have the potential to cause serious patient harm or death," Med. Dev. Recall Notice, FDA, Mar. 2026.

[7] Fresenius Kabi USA, "Urgent Medical Device Correction — Ivenix Large Volume Pump LVP Software, Versions 5.10.1 and earlier," Customer Letter, Fresenius Kabi, Lake Zurich, IL, Feb. 2026.

[8] K. Movahedi, P. Mastali, and J. Van Mierlo, "A critical look at coulomb counting approach for state of charge estimation in batteries," *Energies*, vol. 14, no. 14, art. 4074, Jul. 2021, doi: 10.3390/en14144074.

[9] M. Movahedi, M. Asadi, and S. Esfandyari, “Implementation of an improved coulomb-counting algorithm for the SOC estimation of li-ion batteries,” arXiv:1803.10654, Mar. 2018.

[10] RTCA, *DO-178C: Software Considerations in Airborne Systems and Equipment Certification*, RTCA, Washington, DC, 2011.

[11] P. Jones, R. Jetley, and J. Abraham, “A formal methods-based verification approach to medical device software analysis,” *IEEE Computer*, vol. 43, no. 4, pp. 78–84, Apr. 2010, doi: 10.1109/MC.2010.114.

[12] M. Schmitt, A. Kaestner, and T. Bauer, “Approach towards semi-automated certification for low-criticality ML-enabled airborne applications,” arXiv:2501.17028, Jan. 2025.

[13] X. Gao, Q. Wen, X. Duan, W. Jin, X. Tang, L. Zhong, S. Xia, H. Feng, and D. Zhong, “A hazard analysis of class I recalls of infusion pumps,” *JMIR Human Factors*, vol. 6, no. 2, art. e10366, May 2019, doi: 10.2196/10366.

[14] R. R. Lutz and H. Y. Shaw, “Applying adaptive safety analysis techniques,” in *Proc. 10th IEEE Int. Symp. Software Reliability Eng. (ISSRE)*, 1999, pp. 42–49, doi: 10.1109/ISSRE.1999.809308.

[15] R. Jetley, S. P. Iyer, and P. L. Jones, “A formal methods approach to medical device review,” *IEEE Computer*, vol. 39, no. 4, pp. 61–67, Apr. 2006, doi: 10.1109/MC.2006.122.

[16] AAMI, *AAMI TIR57:2016/(R)2023 — Principles for Medical Device Security — Risk Management — Application of ISO 14971 to Medical Device Security*, AAMI, Arlington, VA, 2023.

[17] P. L. Goddard, “Software FMEA techniques,” in *Proc. Annu. Reliab. Maintainability Symp.*, 2000, pp. 118–123, doi: 10.1109/RAMS.2000.816294.

[18] J. Edworthy, S. Hellier, and R. Titchener, “Updated IEC 60601-1-8 breaks new ground in development of alarm sounds,” *AAMI Biomed. Instrum. Technol.*, vol. 56, no. 1, pp. 28–34, Jan. 2022.

[19] B. J. Drew et al., “Insights into the problem of alarm fatigue with physiologic monitor devices: a comprehensive observational study of consecutive intensive care unit patients,” *PLOS ONE*, vol. 9, no. 10, art. e110274, Oct. 2014, doi: 10.1371/journal.pone.0110274.

[20] M. Cvach, “Monitor alarm fatigue: an integrative review,” *AAMI Biomed. Instrum. Technol.*, vol. 46, no. 4, pp. 268–277, Jul. 2012, doi: 10.2345/0899-8205-46.4.268.

[21] D. Arney, R. Jetley, P. Jones, I. Lee, and O. Sokolsky, “Formal methods based development of a PCA infusion pump reference model: Generic Infusion Pump (GIP) project,” in *Proc. Joint Workshop High Confidence Med. Devices, Software, Systems and Med. Device Plug-and-Play Interoperability (HCMDSS-MDPnP)*, 2007, pp. 23–33, doi: 10.1109/HCMDSS-MDPnP.2007.36.

[22] M. Pajic, R. Mangharam, O. Sokolsky, D. Arney, J. M. Goldman, and I. Lee, “Model-driven safety analysis of closed-loop medical systems,” *IEEE Trans. Industr. Informat.*, vol. 10, no. 1, pp. 3–16, Feb. 2014, doi: 10.1109/TII.2013.2257805.

[23] I. Lee, O. Sokolsky, S. Chen, J. Hatcliff, E. Jee, B. Kim, A. King, M. Mullen-Fortino, S. Park, A. Roederer, and K. K. Venkatasubramanian, “Challenges and research directions in medical cyber-physical systems,” *Proc. IEEE*, vol. 100, no. 1, pp. 75–90, Jan. 2012, doi: 10.1109/JPROC.2011.2165270.

[24] S. Bonfanti, A. Gargantini, and A. Mashkoor, “A systematic literature review of the use of formal methods in medical software systems,” *J. Software: Evol. Process*, vol. 30, no. 5, art. e1943, May 2018, doi: 10.1002/smr.1943.

[25] D. Arney, M. Pajic, J. M. Goldman, I. Lee, R. Mangharam, and O. Sokolsky, “Toward patient safety in closed-loop medical device systems,” in *Proc. 1st ACM/IEEE Int. Conf. Cyber-Physical Syst. (ICCPS)*, 2010, pp. 139–148, doi: 10.1145/1795194.1795214.

[26] K. S. Ng, C.-S. Moo, Y.-P. Chen, and Y.-C. Hsieh, “Enhanced coulomb counting method for estimating state-of-charge and state-of-health of lithium-ion batteries,” *Appl. Energy*, vol. 86, no. 9, pp. 1506–1511, Sep. 2009, doi: 10.1016/j.apenergy.2008.11.021.

[27] A. Eddahech, O. Briat, and J.-M. Vinassa, “Real-time SOC and SOH estimation for li-ion battery using BMS embedded methods,” *IEEE Trans. Veh. Technol.*, vol. 63, no. 6, pp. 2645–2655, Jul. 2014, doi: 10.1109/TVT.2014.2305041.

[28] U.S. Food and Drug Administration, *21 CFR Part 820 — Quality System Regulation*, U.S. Code of Federal Regulations, Washington, DC, 2024.

[29] K. C. Nanji, A. Patel, S. Shaikh, D. L. Seger, and D. W. Bates, “Evaluation of perioperative medication errors and adverse drug events,” *Anesthesiology*, vol. 124, no. 1, pp. 25–34, Jan. 2016, doi: 10.1097/ALN.0000000000000904.

[30] K. C. Schnock et al., “The frequency of intravenous medication administration errors related to smart infusion pumps: a multihospital observational study,” *BMJ Quality & Safety*, vol. 26, no. 2, pp. 131–140, Feb. 2017, doi: 10.1136/bmjqs-2015-004465.

[31] European Parliament and Council, *Regulation (EU) 2017/745 on Medical Devices (MDR)*, OJ L 117, 5 May 2017, p. 1.

[32] Y. Wang, J. Tian, Z. Sun, L. Wang, R. Xu, M. Li, and Z. Chen, “A comprehensive review of battery modeling and state estimation approaches for advanced battery management systems,” *Renew. Sustain. Energy Rev.*, vol. 131, art. 110015, Oct. 2020, doi: 10.1016/j.rser.2020.110015.

[33] M. Cleaveland, S. Mitra, and I. Lee, “Run-time assurance for learning-enabled systems,” in *NASA Formal Methods*, LNCS 12229, Springer, 2020, pp. 361–368, doi: 10.1007/978-3-030-55754-6_21.