Description

Title

Dynamic Personalized Drug Library and Closed-Loop Medication Control System

Field of the Invention

The present invention relates to medical infusion systems and patient-specific medication dosing. In particular, it concerns smart infusion pump technology with dynamic dose error-reduction libraries and closed-loop control algorithms for intravenous drug delivery. The system spans the fields of hospital pharmacy informatics, medical device control (especially IEEE 11073 infusion device integration), and adaptive patient monitoring, enabling real-time personalized medication management in critical care and other clinical settings.

Background of the Invention

Infusion pumps equipped with dose error reduction software (DERS) and drug libraries have become standard in hospitals to prevent programming errors. These drug libraries define standardized concentration units and dose limits for various medications. However, current smart-pump libraries are generally static and population-based, updated manually by pharmacy committees and lacking patient-specific customization. This can lead to **alert fatigue** and frequent overrides: clinicians often encounter non-actionable or overly conservative alerts when a patient's condition necessitates doses outside the static limits[3]. For example, raising certain drug dose limits in a static library was shown to reduce nuisance alerts by 35–100% in practice[3], indicating that many default limits are not optimally tuned to clinical realities. Yet, without dynamic adjustment, nurses must override or bypass the library in urgent cases, which undermines safety. Indeed, studies have found that bypassing the smart pump's library (e.g. by operating in "basic" mode or using unlisted drugs) contributes to about **10% of infusion errors or policy violations**[4]. This highlights a dilemma: rigid libraries protect against errors but can impede care if not tailored, whereas workarounds to deliver needed therapy introduce new risks.

Another limitation of current infusion systems is the lack of coordination between multiple infusions. In complex cases such as the ICU, patients receive **multiple concurrent drug infusions** (vasopressors, sedatives, insulin, electrolytes, etc.) that can interact. Traditional pumps function in isolation, and no commercial system dynamically prioritizes or adjusts doses when multiple closed-loop infusions might conflict. For instance, separate single-drug closed-loop

controllers (e.g. for insulin or analgesia) have been explored, but these operate independently and do not share safety constraints or knowledge of each other. There is **no published system** in the prior art that can automatically cancel, delay, or adjust one drug infusion in response to another to prevent harmful interactions. This shortcoming leaves clinicians to manually coordinate therapies and often react to adverse interactions after they occur (e.g. vasopressor-induced hypertension from an overly aggressive sedative dose). Additionally, maintaining safe inter-drug ratios (such as keeping vasopressors and vasodilators balanced, or electrolytes like calcium and phosphate within solubility limits) is left to manual calculation or broad rules of thumb, since infusion pumps do not enforce multi-drug "guard rails."

Lack of personalization is another critical gap. Every patient has unique pharmacokinetics and sensitivities – for example, sedation requirements can vary ten-fold among ICU patients, and tolerance can change over time. Static libraries cannot adapt to an individual's responsiveness, leading either to under-treatment (if limits are too strict for a resistant patient) or risk of overdose (if limits are too lenient for a sensitive patient). Currently, any personalization relies on clinician judgment and repeated adjustments; the pumps themselves do not learn from patient data. Some advanced systems (e.g. anesthesia closed-loop controllers or insulin pumps) use feedback from a single sensor (such as BIS for anesthesia or glucose for insulin) to adjust one drug, but these systems still use fixed safety thresholds and do not update the underlying library limits per patient. There is a need for a system that not only performs closed-loop control but **learns the patient's individual dose–response over time**, refining the acceptable dose range and optimal targets for that specific patient.

Furthermore, existing pumps lack robust integration of patient data and external safety checks. **Patient context** (such as weight, organ function, lab trends) is not automatically reflected in infusion limits – for example, dose limits might not adjust for a patient's weight or renal function unless manually calculated. There is also minimal integration with electronic medical records (EMR) to fetch real-time data or to document infusion events in detail. **Continuous Quality Improvement (CQI)** data from pumps (which record alerts and overrides) are typically analyzed only retrospectively, rather than actively feeding back into the system to improve safety limits in real time. As a result, opportunities to refine dosing rules or catch systematic issues are delayed.

On the hardware side, **safety interlocks** on current infusion devices are basic. Pump channels generally do not auto-recognize drug types or concentrations – they rely on users selecting the correct library entry or scanning a barcode. Wrong-drug or wrong-concentration errors (e.g. a high-concentration electrolyte solution accidentally loaded in place of a dilute one) can go undetected by the device. While connectors like color-coded ports or keyed syringes exist in niches, most IV pumps cannot electronically verify that the intended drug is what is actually connected. Additionally, once an infusion is started, pumps typically have no built-in mechanism to halt delivery if an unexpected patient condition arises unless a separate monitor triggers an alarm and a human intervenes. There is an absence of a **closed-loop safety supervision** that can directly stop or modulate infusions based on live patient data beyond simple pressure occlusion alarms or dose-limit alerts.

In summary, current smart infusion pumps and pharmacy library systems are limited by static parameters, one-size-fits-all dosing limits, lack of multi-drug coordination, inability to adapt to patient-specific responses, and incomplete integration with patient monitoring and hospital IT infrastructure. These limitations contribute to alarm fatigue, frequent library overrides, potential

medication errors, and suboptimal therapy. The present invention addresses these gaps by providing a dynamically updating drug library overlay and a closed-loop control system that together **personalize and safeguard infusion therapy** in ways not possible with existing technology.

Summary of the Invention

The invention provides a **Dynamic Personalized Drug Library (DPL) and Closed-Loop Medication Control System** that overcomes the above limitations and brings infusion therapy into a new era of precision and safety. The system architecture augments a conventional smart-pump dose error reduction system with a multi-layer control and learning framework, thereby embedding patient-specific intelligence into the infusion process.

At a high level, the system comprises: (1) one or more **smart infusion cartridges or channels** each associated with a medication, (2) a repository of drug dosing parameters that includes a **baseline library (G0)** of standard safe limits and a **patient-specific overlay (G1)** of dynamic limits, and (3) a hierarchical **controller** that manages drug delivery in real time using patient data feedback. The controller continuously monitors patient state via sensors and clinical data streams, compares therapy progress against desired targets and safety constraints, and automatically adjusts infusion rates or suspends therapy as needed. Importantly, the controller updates the G1 overlay parameters on-the-fly based on the patient's individual response and usage context, while ensuring that all adjustments remain bounded by fundamental safety rules (the G0 layer). This two-layer library approach allows personalization **without compromising the established safety envelope** of institutional DERS practices.

Multi-Tier Control Architecture: The infusion control logic is organized into a hierarchy of layers that separates critical safety enforcement from optimization and learning tasks. In one preferred embodiment, there are four main layers: - **L-0: Vital Override.** This top-priority layer acts as an emergency safety net. It monitors for immediately life-threatening conditions or extreme rule violations (e.g., a patient vital sign outside acceptable range, or an infusion command that would clearly overdose the patient) and can instantaneously halt or adjust infusions to avoid harm. L-0 essentially implements "hard coat" safety limits that cannot be overridden by any lower layer. For example, if arterial pressure falls below a set threshold or a pump's measured output exceeds the maximum hard limit, the L-0 layer will clamp or stop relevant infusions and trigger an alarm. - **L-1: Guard-Rail Arbiter.** This layer enforces the drug library constraints and interaction rules. It uses a **multi-vertex conflict graph** to manage multiple simultaneous infusions. If two therapies conflict (e.g., two drugs of the same class or physiologically opposite effects), L-1 will automatically scale back or temporarily suspend the lower-priority infusion to maintain overall safety. It also upholds ratio constraints (so that, for instance, electrolyte infusions keep critical ion ratios within safe bounds) and ensures cumulative dose limits (mg/kg per time) are not exceeded. L-1 operates continuously in the background, adjusting each pump's permissible range based on all active drugs' metadata and the patient's current physiologic state. - **L-2: Optimizer Control.** This layer handles goal-directed dosing adjustments using feedback control algorithms. For each drug or physiologic variable under control, L-2 may run a proportional–integral–derivative (PID) controller, model-predictive controller (MPC), or other algorithm to titrate the infusion rate towards a target (for example, keeping mean arterial pressure or sedation depth at a setpoint). L-2 generates candidate dose commands that achieve therapeutic objectives, but these commands are always subject to

modification or veto by the higher layers (L-0 and L-1). Thus, the optimizer focuses on efficacy (meeting clinical targets) within the **safety envelope** enforced above. - **L-3: Adaptive Learning (μ-Band Training).** This bottom layer is an innovation that allows the system to **learn patient-specific pharmacologic responses** over time. L-3 comes into play when the patient's parameters are within acceptable range ("micro-dose band") and no urgent corrections are needed. In these stable periods, the controller injects very small dose perturbations – on the order of a few percent of the current rate – to probe the patient's response without causing any clinical effect[5]. By analyzing how the patient's physiology changes with these tiny, safe "tick-tock" dosing variations (e.g., a slight increase then decrease in infusion rate), the system updates the **Jacobian matrix** of partial sensitivities between drugs and physiologic outputs for that patient[6][7]. In essence, L-3 refines the drug library overlay (G1) by learning how this particular patient reacts to each medication and combination, allowing subsequent doses to be tuned more accurately. The learning is constrained by safeguards: it starts from known population-average parameters (to ensure a safe baseline) and only adapts within a narrow band until sufficient confidence is gained (e.g., the controller requires that the learned parameter's confidence interval shrinks below 50% of initial uncertainty before trusting it fully[8]).

Through this hierarchical approach, the system achieves **closed-loop control** that is at once safe, responsive, and personalized. The upper layers guarantee that no action of the lower layers can violate core safety rules. For instance, if the optimizer (L-2) suggests increasing a vasopressor dose but the patient's heart rate is dangerously high, the guard-rail layer (L-1) or vital override (L-0) will override or limit that command, thereby preventing compounding of risk. Conversely, the lower layers provide continuous fine-tuning and adaptation that the rigid upper layers alone could not accomplish. This division of responsibilities ensures both **immediate safety** and **long-term adaptation**.

Dynamic Drug Library Overlay: At the heart of the invention is the dynamic drug library (DPL) which overlays patient-specific learned limits on the static hospital library. The system maintains a data structure for each drug being infused that includes: 1. **Drug Identification and Class Code:** A unique identifier for the drug and its pharmacologic class or category. This can be used to detect duplications or conflicts (e.g., two drugs from the same class running concurrently). 2. **Standard Dose Limits (Base Library, G0):** The institution-approved soft and hard limits for that drug (dose rate, concentration, bolus size, cumulative dose per 24h, etc.), as would be found in a traditional drug library. 3. **Patient-Specific Limits (Overlay, G1):** Adjusted upper and lower limits for that drug's infusion parameters, which are recalibrated continuously. These might tighten or relax the range based on the patient's observed tolerance. For example, if a patient consistently requires higher doses, the system might raise the soft limit to reduce nuisance alarms (while never exceeding an absolute maximum)[9][10]. Conversely, if a patient is sensitive, the system lowers the limits to prevent overshoot. 4. **Micro-dose Band (μ-band) Parameters:** A small dose-rate interval (or percentage of the normal dose range) designated for probing. Within this band, dosing oscillations are considered too minor to cause clinical effect but sufficient for system identification. The overlay stores the current μ-band boundaries for each drug or control variable. 5. **Learned Patient Model Coefficients:** The current best estimates of the patient's response sensitivities (Jacobian elements) relating this drug to various physiological metrics. These coefficients are updated by the L-3 learning process and used by L-1 and L-2 to anticipate effects of dose changes. For example, the system might learn that "for this patient, increasing Drug A by 1 μg/min raises blood pressure by 5 mmHg" and incorporate that into the

guard-rail logic to avoid overshoot on blood pressure if another drug also affects it. 6. **Cumulative Dose Counters:** Running totals of how much drug has been delivered (over certain time frames, e.g., per hour and per 24 hours). These counters feed into both safety limits (ensuring daily maxima are not exceeded) and analytics (for documentation and trend analysis). 7. **Override and State History:** Flags or records indicating if/when manual overrides were invoked for this drug, and any recent state changes (such as the patient entering a different care phase). This can influence how aggressively the overlay adapts (e.g., after an override, the system may temporarily widen limits or require confirmation before re-narrowing). 8. **Authenticity and Configuration Data:** Each cartridge or channel carries a cryptographic serial number or certificate, calibration constants (if applicable), expiration date of the drug, and any manufacturer-set parameters. These are used to ensure the system recognizes the cartridge and trusts its content. A **tamper-evident design** ensures that if a cartridge is removed or altered, its authentication is revoked[11][12].

All the above data is preferably stored in a non-volatile memory (such as an EEPROM or FRAM) on the infusion cartridge or pump module itself, and is read by the controller upon cartridge insertion and continuously re-checked during operation[13][14]. Storing critical parameters on the cartridge enables **plug-and-play safety**: the system can instantly configure itself to the drug's properties and enforce the correct limits without manual input. It also prevents using an **expired or incompatible drug**, since the cartridge can signal lockout conditions. For instance, if a cartridge's expiration date has passed or if its drug class is on an incompatibility list with another running drug, the system will refuse to start that infusion or will shut it down automatically[15][16]. This effectively makes certain errors (like wrong drug in line or decimal misplacement in programming) **physically impossible**, because the firmware will not energize a pump that does not pass the metadata safety checks.

Closed-Loop Operation: In use, the system creates a feedback loop for each controlled physiological parameter. Consider an ICU patient on a vasopressor medication to maintain blood pressure. A blood pressure sensor provides real-time readings to the controller. The DPL overlay for the vasopressor has initial dose ranges but is ready to adapt. The controller's optimizer layer (L-2) periodically adjusts the infusion rate to keep the blood pressure at the target mean (for example, 65–75 mmHg). If the blood pressure drifts low, L-2 suggests an increase in vasopressor rate. Before executing, the guard layer L-1 checks this against the patient's current drug library overlay: if the requested rate would exceed the patient-specific limit (perhaps recently adjusted upward as the patient has shown tolerance), L-1 will cap it or require a smaller increment. Simultaneously, L-1 looks at other infusions and rules: if the patient is also on a vasodilator that is due to be decreased or an inotrope that interacts, it may suppress those to allow the vasopressor to work without conflict. Once vetted, the command is sent to the pump (L-0 would intercept only if it were an extreme aberration that threatens immediate harm). The pump delivers the dose, and the cycle continues with new sensor feedback each second or minute. Over time, as the patient stabilizes near target, the micro-dose learning layer L-3 kicks in: the system may introduce a tiny 5% oscillation in infusion rate for a short period and measure the blood pressure response. Suppose it finds the patient's blood pressure is very sensitive to the drug (rises quickly even with small increments) – the controller will update the G1 overlay to narrow the safe range (lower the maximum recommended rate for this patient) and adjust the PID parameters to avoid overshooting. These changes are logged and will persist, so if a new nurse

takes over or the pump is restarted, the library overlay retains the knowledge of the patient's requirements.

If the patient's condition changes (e.g., develops tachycardia or renal impairment), the system can also incorporate **physiologic state rules** into the overlay adjustments. For example, certain drug limits might dynamically change if lab results indicate organ dysfunction (the system can receive such data via HL7 from the EMR). The controller can periodically query or accept pushed data like serum drug levels, creatinine, etc., and use that to modify dosing or trigger alerts. All such adaptations are done under the governance of hospital-approved rules and recorded for transparency.

Override Workflow: The system is designed to cooperate with clinicians, not to replace them. Therefore, a robust override and authorization workflow is built in, ensuring that a human can always intervene but in a controlled, logged manner. If a clinician needs to administer a dose outside even the personalized limits (for example, an extraordinary bolus during a code situation), they can invoke an **override mode**. In one embodiment, the nurse would authenticate (e.g., by scanning a staff ID badge or entering a code) at the pump or central station. The controller then checks the credentials and role against its policy (for instance, only an attending physician or authorized critical care nurse can override certain high-risk limits)[17][18]. If approved, the system will allow the infusion beyond normal constraints for a bounded time window (e.g., 5 minutes) or dose amount, during which it will suspend alarm sounds but continue monitoring[19][20]. The override state is indicated clearly (e.g., the pump channel turns amber indicating temporary override[21]). Once the timer expires or the clinician manually ends the override, the system automatically reverts to closed-loop mode and re-enables the normal limits[18]. All actions in this process – the request, grant, identity of who overrode, duration, and doses given – are logged to the audit ledger[22]. Multiple successive overrides are tracked; the system can require escalating authorization if, say, more than three overrides occur in a shift for the same parameter (prompting a physician review)[23]. The override workflow thus provides a **"safety valve"** for clinicians to exercise judgment in unique cases, while ensuring such deviations remain **explicit, temporary, and reviewable**[24][25]. This mechanism also contributes to quality improvement by identifying situations where the standard or learned limits might need refinement for future patients.

Integration and Deployment: The invention is designed to integrate into existing hospital infrastructure. The controller hardware can be a dedicated bedside device or incorporated into an infusion pump module. It communicates with hospital systems using standards-based protocols – for example, it can receive patient data (e.g., weight, lab results, prescribed orders) from the electronic health record via **HL7 messaging** interfaces, and it can report infusion status and events back to the EHR for charting. For direct device interfacing, the system uses the IEEE 11073 family of standards for medical device communication, enabling it to control third-party infusion pumps or to ingest data from patient monitors on the network. In one embodiment, the controller implements the IHE Infusion Pump Event Communication (IPEC) profile, so that any infusion start, rate change, alarm, or completion event is automatically sent to the clinical information system. This not only streamlines documentation but also provides additional context to the control system (for example, recognizing an "end of infusion" event could trigger it to adjust other drips accordingly). By adhering to these interoperability standards, the dynamic library controller can function as a **middleware layer** atop existing infusion pumps if needed – effectively upgrading legacy pumps with adaptive capabilities without hardware replacement.

The system's architecture is modular and scalable. In a multi-infusion setup (e.g., an ICU patient on many drips), multiple smart cartridges (or channels) plug into a central control bus. As shown in **FIG. 5**, a manifold (500) may house several cartridge slots (510) sharing common fluid supply lines (520) and a data communication bus (530) for hot-plug detection[26][27]. Each cartridge mechanically and electrically mates via a standardized connector, and the controller auto-discovers the cartridge type and loads its library data from the on-board memory (830, see FIG. 8)[28]. Cartridges can be swapped in and out without stopping the entire system – a **hot-swap** operation illustrated in **FIG. 4**. When a cartridge is removed from a slot (440) and a new one inserted (430), sensors detect the change (e.g., a leaf switch or reed sensor, 720 in FIG. 7) and trigger the controller to pause flow through that slot by opening a bypass valve (410) within milliseconds[29][30]. The new cartridge is authenticated (via its cryptographic ID 730) and initialized in a few seconds, including a brief automatic priming sequence (pumping fluid through a purge line 420 to eliminate air)[29][31]. This design minimizes disruption – continuous infusion is maintained through alternate pathways while swapping, and the closed-loop logic either holds steady or smoothly transfers control to the new cartridge's parameters. Such capabilities make the system flexible and **reduce downtime** for cartridge changes (for example, switching out an empty drug reservoir or replacing a sensor cartridge) without compromising sterility or control continuity[29][32].

Overall, the DPL and closed-loop control system transforms the traditional smart pump into an **adaptive therapeutic device**. It achieves the twin goals of enhancing patient safety (through vigilant, multi-layered safeguards and removal of human error opportunities) and improving efficacy (through individualized therapy that can respond to minute-by-minute changes in patient condition). The system's novel combination of features – dynamic per-patient drug libraries, real-time conflict arbitration across infusions, micro-dose learning of patient parameters, and rigorous override and logging workflows – creates a closed-loop medication platform that is self-correcting, transparent, and deeply integrated with clinical practice.

Brief Description of the Drawings

- **FIG. 1** – Exploded Smart-Cartridge Hardware Assembly. An exemplary infusion cartridge is shown in an exploded view, including housing and internal components. Key elements (with reference numerals) include the cartridge housing (100), a built-in sensor patch (110), on-board microcontroller (120), micro-pump actuator (130), drug reservoir (140), electrical connector interface (150) for data/power, unique ID memory chip (160), fluid quick-connect port (170) with O-ring seal (180), and a tamper-evident locking ring (190)[33].
- **FIG. 2** – Control-Loop Sequence (Swim-Lane Diagram). This diagram outlines the timing and data flow of the closed-loop control process for one channel. It shows the sensor reading (200) being acquired each cycle, the edge-AI or predictive module (210) optionally running, the main control engine decision step (220) comparing the sensor value to target and thresholds, the actuator command (pump actuation 230) delivering a micro-dose if needed, and the audit logger (240) recording the event[34][35]. Steps 1–5 of a typical control iteration are labeled.
- **FIG. 3** – Audit-Trail Hash-Chain Flow. A block diagram illustrating how each infusion event (JSON record 300) is hashed (310), linked to a running Merkle chain (nodes 320, root 330), and anchored in a secure ledger block (header 340). This provides a

tamper-evident log of all dosing decisions and overrides for compliance purposes[36][37].

- **FIG. 4** – Hot-Swap Cartridge Replacement Sequence. This schematic sequence shows the steps when exchanging an infusion cartridge without stopping system flow. It features the manifold socket (400), bypass valve (410) opening to shunt flow during swap, purge drain line (420) for priming, removal of the spent cartridge (old cartridge 440) and insertion of a new cartridge (430). Arrows indicate the fluid path and control signals ensuring continuous infusion and maintaining sterility during the <30 second swap procedure[29][38].
- **FIG. 5** – Five-Slot Manifold Assembly. A hardware diagram of an infusion manifold that can dock multiple smart cartridges. Five cartridge bays are depicted in a common body (500), each with an electrical/fluid connector (510). Shared components include a common fluid supply line (520), a data bus (530) connecting all slots to the main controller, and a drain port (540) for purging or common waste. This assembly allows parallel operation of multiple infusions with centralized control[26][27].
- **FIG. 6** – Digital-Twin Validation Flow. A flowchart illustrating the optional predictive analytics and model validation layer. Sensor data is fed into a data lake (600) and replay engine (610) that runs a digital twin or historical simulation. A machine-learning model (620) forecasts future parameters; a residual calculator (630) computes prediction error, which is compared via a gamma (γ) tolerance comparator (640) to decide if the ML model is acceptable. A retraining loop (650) and deployment gate (660) manage updating the model if error stays within limits[39][40]. This figure shows how the system can validate advanced predictive models against real data before using them in control.
- **FIG. 7** – Physical Tamper-Evidence Mechanism. An illustration of the cartridge tamper-evident seal. The cartridge cap (700) has a frangible tear line (710); opening it triggers a leaf switch sensor (720) and invalidates the UID chip (730) via firmware. A collar (740) on the housing holds the cap in place. Once broken, the cartridge's certificate or ID cannot be reused, ensuring one-time use and security[41][42].
- **FIG. 8** – Chelator Counter-Measure Cartridge and Pump Integration. This figure shows a special cartridge (e.g., an antidote or chelating agent cartridge) installed in slot 5 of the manifold, and its fluid routing. The dedicated micro-pump (800) for this cartridge delivers antidote into the main line when triggered. A pair of driver circuits (810) can provide iontophoresis current if the cartridge is a sensing/removal type. A status LED (820) indicates μ-band probe activity (green/amber). The cartridge's EEPROM pad (830) and an independent dose-budget watchdog microcontroller (840) are highlighted, illustrating on-cartridge safety electronics[43][28].
- **FIG. 9** – Iontophoresis Extraction Timing Diagram. A graph showing the electrical current profile (e.g., 400 μA pulses) over time (60-second duty cycles) for an iontophoresis-based analyte extraction or delivery, as might be used in a sensor patch. It demonstrates the pattern of microcurrent application and rest periods to draw molecules through the skin in a controlled manner[44][45].
- **FIG. 10** – Critical-Factor Sensor Daughterboard and Auxiliary Pump Cassettes. A schematic of an add-on sensor board that carries additional sensors (for critical biomarkers like zinc, copper via aptamer arrays) and two small supplementary pump cassettes. It stacks onto the main bus, with features labeled: slots for sensor inputs and

mini-pumps. This shows how the system can be expanded with extra sensing and actuation for special use-cases[46][47].

- **FIG. 11** – Pump-Bay Map (Slots 1–16). A layout diagram mapping a multi-channel pump system's bay numbering and class-code assignments. Each slot 1–16 is labeled with its intended cartridge type or class (for example, slots designated for vasopressors, analgesics, etc.), illustrating one possible configuration of a large pump manifold. This map helps prevent insertion of a wrong cartridge in a slot by hardware keying or software checks (each slot can reject cartridges whose class code doesn't match the map)[48][49].
- **FIG. 12** – Multi-Analyte Sensor Daughtercard. A schematic depiction of an integrated sensor card capable of measuring multiple analytes simultaneously (e.g., a 6-electrode electrochemical panel). It highlights the electrode array and its connection to the controller. This card can be used in-line with blood or fluid samples to provide real-time lab results (such as INR, lactate, drug levels) feeding the closed-loop system[49][50].
- **FIG. 13** – Flex-Pump Sub-Manifold (Slots 21–36). An illustration of a flexible micro-pump array that can be attached as an extension to the main manifold, providing additional channels (lanes) for micro-infusions. Sixteen micro-pumps are shown in a compact arrangement, feeding into a common output. This allows the system to scale up the number of controlled infusions without a large footprint[51][52].
- **FIG. 14** – Multi-Infectious-Antigen Sensor Pad. A perspective view of a layered lateral-flow pad (like a lab-on-chip) that can detect multiple infectious disease markers from a small fluid sample. The figure illustrates the stacked construction enabling multiplexed pathogen antigen sensing. This optional module can plug into the system to provide triggers for certain therapies (e.g., detection of a sepsis marker could prompt antibiotic dosing)[52][53].
- **FIG. 15** – Endotoxin & HMGB-1 Sorbent Cartridge in CRRT Loop. A diagram showing a sorbent cartridge placed inline with a continuous renal replacement therapy (CRRT) circuit to remove endotoxins or inflammatory mediators (like HMGB-1) from blood. It depicts how this cartridge (with its special media) is connected and can be controlled by the system (e.g., engaged or bypassed via pumps and valves) to provide extracorporeal detoxification as part of critical care therapy[54][55].
- **FIG. 16** – Hierarchy & Knock-Down Conflict Graph. A schematic summarizing the system's three-layer arbitration hierarchy and the conflict resolution graph (knock-down matrix) used to prioritize therapies[45]. It visually represents high-priority loops suppressing lower-priority ones, with nodes as control loops and edges indicating inhibitory relationships. This figure illustrates how a vital override (L-0) or guard-rail rule (L-1) can "knock down" a lower loop's command to prevent mutual interference, thereby maintaining overall stability in a 140-loop network.
- **FIG. 17** – Smart-Cartridge EEPROM Memory Map. A diagram of the memory layout on each infusion cartridge's EEPROM/FRAM chip[56][57]. It shows fields for the cartridge's unique ID and class code, expiration date, calibration constants, dose limit parameters (soft/hard), μ-band threshold values, incompatibility list (drug-class conflict codes), and log counters. This map defines how the cartridge stores all metadata needed for the controller to safely manage that drug.
- **FIG. 18** – μ-Band Probe Cycle Timeline. A plot illustrating the adaptive μ-band probing process and the convergence of learned patient-specific coefficients over time[57][58].

The graph displays how the controller periodically applies tiny dose perturbations (the μ-band "probe pulses") and measures the corresponding changes in a physiological signal, refining the Jacobian matrix. It also indicates how the confidence interval of the learned parameter shrinks as more probe cycles occur, eventually leveling off once the patient's parameter is identified (J^patient converges)[59][8].

- **FIG. 19** illustrates a **Sniffer-Tap data-path overview** for the passive-capture embodiment. An infusion pump **1920**forwards all serial/Ethernet traffic through the in-line **Sniffer Tap Module 1900**, which mirrors the frames to the original host gateway **1930** while simultaneously streaming a normalized event feed to the SmartStop real-time analytics cloud **1940**. Solid arrows indicate the direction of primary data flow (pump → tap → host) and the secure uplink from the tap to the cloud scheduler.

- **FIG. 20** is an **exploded hardware view** of the Sniffer Tap Module. From top to bottom the stack comprises: a protective enclosure **1950**; a dual-port Ethernet hub or Y-cable interface **1960** that presents two transparent pass-through ports; an FPGA line-sampler board **1970** equipped with dual-port SRAM for lossless capture at up to 1 Gb s^{-1}; an RTOS micro-controller board **1980** handling protocol decode, MQTT/TLS publishing, and OTA updates; and a FIPS-rated secure crypto chip **1990** that stores device certificates and signs hash-chain audit entries. The downward arrows depict the assembly sequence of these layers into the finished tap.

Detailed Description of Embodiments

System Architecture Overview

Referring to **FIG. 1**, the core hardware of the system is a **smart infusion cartridge** or module that interfaces with a central controller. Each cartridge contains the necessary components to autonomously manage a single drug infusion under supervision of the controller. These components include a cartridge body or **housing** (100) that encloses a small **reservoir** (140) of the liquid medication, a miniaturized **micro-pump** (130) (for example, a piezoelectric micropump capable of precise μL-scale dosing), and an integrated **sensor patch** (110) or interface relevant to that drug (which could be, for instance, a pressure sensor to detect line occlusion, or a chemical sensor if the cartridge doubles as a monitoring module)[33]. Each cartridge also has embedded electronics: a low-power **microcontroller** (120) dedicated to that cartridge, responsible for local tasks like pump control and sensor sampling, and a non-volatile memory (**UID/FRAM chip** 160) that stores the cartridge's identity and drug-specific data[33]. The cartridge connects to the rest of the system via a standardized **quick-connect interface** (170) that includes fluid connectors (e.g., a Luer lock with O-ring seal 180 for the drug inlet/outlet) and electrical contacts (often spring-loaded pogo pins, part of connector block 150)[33]. A **tamper-detect mechanism** (such as a frangible ring 190 or seal) is built into the cartridge; if broken (during unauthorized opening), a switch triggers and the cartridge's certificate in memory is invalidated[60][42], thereby preventing any use of a compromised cartridge.

The cartridges are designed to be **plug-and-play and hot-swappable**. They slide into slots on a manifold (see FIG. 5) where the quick-connect (170) mates with a socket (400 in FIG. 4). The central controller can accommodate multiple cartridges concurrently, communicating with each via a bus (e.g., an RS-485 multi-drop network 530 on the manifold[26]). When a cartridge is inserted, the controller reads its memory (160) to identify the drug and load all relevant parameters into the system's active drug library data structure. When removed, the controller marks that infusion as inactive and can physically isolate the slot's fluid line by closing a valve or using a manifold bypass (410) to prevent interruption of flow to other cartridges[29]. **FIG. 4** illustrates this hot-swap process, where a bypass valve (410) opens immediately upon cartridge ejection (detected via a sensor) to maintain fluid continuity, and a brief purge occurs when a new cartridge is inserted to eliminate air before resuming dosing[29][38].

The **central controller** itself can be implemented as an embedded computing device (for example, a 32-bit microcontroller or an ARM Cortex processor) with both real-time capabilities and communication interfaces. It runs the hierarchical control algorithm and coordinates all cartridges. In some embodiments, critical real-time functions (like L-0 safety checks and L-1 guard-rails) are executed on a dedicated safety microcontroller, whereas higher-level functions (optimization, learning, UI/communications) run on a companion processor. This **dual-processor design** (primary controller plus safety co-processor) ensures that even if the main controller is busy or needs an update, the safety layer remains operational and immutable[61]. The system also includes a power management unit (with battery backup for a limited period, to handle power failures gracefully by pausing infusions and preserving logs) and network interfaces (e.g., Wi-Fi or Ethernet for connecting to hospital networks, and possibly Bluetooth or BLE for wearable sensors if any).

User interaction with the system can be via a touchscreen or connected workstation that displays the status of each infusion (with color-coded indicators per channel, e.g., green for normal run, amber for override active, red for any hard-limit lockout[62]). Clinicians can input commands like starting a new infusion (by inserting a cartridge and selecting a protocol or target), adjusting setpoint goals (e.g., changing the target blood pressure), or initiating an override. The user interface is integrated with the hospital's medication order system such that when a physician orders a medication in the EMR, the DPL system can pre-configure a cartridge slot to expect that drug (flashing an indicator at the correct slot to guide the nurse, for instance). Conversely, any infusion event the system performs is documented back into the EMR automatically as an electronic medication administration record entry.

Data Model and Drug Library Schema

As introduced, each drug in the system is represented by a rich data object comprising static and dynamic fields. **FIG. 17** provides a schematic memory map of these fields on the cartridge's EEPROM[45][56]. The key fields are:

- **Drug Class Code and Identifier:** This is typically a coded value (for example, an 8- or 16-bit code) that uniquely identifies the medication or fluid. It may encode both the drug and concentration. The class code is used by the system to detect conflicts (for instance, two different brand-name cartridges carrying the same active drug would share a class code, or two drugs known to interact might have codes that map to an incompatibility matrix). The identifier also ties into pharmacy databases for verification.

- **Concentration and Calibration Data:** The exact concentration of the drug in the cartridge (e.g., 4 mg/mL norepinephrine) is stored, as well as any calibration factors for the pump (to ensure the volume delivered per pump stroke is accurate). This ensures dosing calculations are precise and device-specific.
- **Dose Limits – Hard and Soft:** The baseline (hospital-defined) dosing limits are stored here. These include the *upper hard limit (UHL)* beyond which the pump should never deliver (often a value that would cause toxicity), *upper soft limit (USL)* which is a threshold that triggers an alert or requires confirmation, *lower soft limit (LSL)* which might trigger an alert if the dose is unusually low (to catch programming errors), and any *lower hard limit* if applicable (for example, TPN may have a minimum safe flow to avoid line clotting). These base limits correspond to the G0 library values and are typically derived from pharmacy guidelines or literature.
- **Cumulative Dose Ceiling:** A parameter representing the maximum total dose allowed over a certain period (e.g., per 24 hours or per lifetime of a cartridge/patient). For example, for certain drugs like chemotherapeutics or digoxin, the system might enforce a 24-hour total not to be exceeded. This value is used along with the running tally in the cartridge to halt infusions that would exceed safe accumulation[63].
- **μ-Band (Micro-dose Band) Definition:** This defines the threshold below which dose changes are considered negligible (for learning purposes). It can be stored as an absolute range or a percentage of typical dose. For instance, μ-LOW and μ-HIGH values might demarcate a ±5% band around the current steady dose. The controller will only perform probing maneuvers within this band.
- **Incompatibility List / Multi-Vertex Vector:** This is a data structure encoding which other drug classes conflict or interact with this drug. It might be a bitmask or list of class codes that cannot run concurrently (hard incompatibility) or must have constrained ratios. In some implementations, this is represented as a vector of coefficients relating this drug to principal interaction "vertices"[64]. For example, a vasopressor might have a vector entry that maps it to a "vasodilation" vertex to indicate it conflicts with vasodilators, or insulin might map to a "blood glucose" vertex to indicate synergy or need for glucose supplementation. The controller uses these vectors to compute on-the-fly whether any combination of active drugs produces a prohibited sum. Essentially, this field feeds the conflict graph (FIG. 16) logic.
- **Patient-Specific Adjustment Factors:** This section of memory is writeable by the controller and stores any current adjustments made to the soft limits or targets for this drug. For example, if the system has learned that a patient needs higher dosage, it might store a factor like "USL multiplier = 1.5x" meaning the soft limit was raised by 50% for now (still within absolute max). Or it could store updated PID gains or model parameters specific to this patient's response.
- **Usage Counters and Logs:** The cartridge tracks volume dispensed, time in use, number of times inserted, etc. A **dose budget** register might count down from the maximum volume; if it hits zero (meaning the cartridge delivered its full allowed dose), the system knows the cartridge is spent or should be replaced[43][28]. There may also be a small circular log of key events (like the last few alarms or overrides) for audit redundancy.
- **Security and Integrity Fields:** This includes the cartridge's digital certificate or authentication hash, manufacturing lot and expiration date, and possibly a firmware version or allowed controller compatibility list. There may be a field that the controller

sets to mark the cartridge as "used" or locked once removed, to prevent reuse on another patient without reconditioning.

All these fields are structured with defined offsets, as depicted in FIG. 17, so that both the cartridge's microcontroller and the main controller have a common understanding of the data layout[45]. Whenever a cartridge is connected, the system reads all relevant fields and checks them. If any check fails (e.g., wrong drug for that slot as per FIG. 11 mapping, expired cartridge, or a conflict with an active drug's incompatibility list), the controller will refuse to arm that infusion channel and will alert the user to the issue[15][16]. This creates a robust **safety lockout** at the hardware level, preventing many infusion errors at the earliest stage.

Control Layers and Hierarchical Arbitration (G0/G1 Integration)

The control software orchestrates the multi-layer hierarchy introduced earlier (L-0 through L-3). At runtime, these layers continuously communicate and override each other as necessary in a top-down priority fashion. **FIG. 16** illustrates the relationship: higher-priority layers (emergency safety, guard-rails) sit above and can preempt lower layers (optimization, logging)[45][56]. In implementation, this may be realized as a stack of decision filters: L-3 suggests a baseline action (often no change, just learning), L-2 computes an optimized infusion command, then L-1 modifies or vetoes it based on safety rules, and L-0 finally has the ability to final-check and abort if needed. Only after passing through L-0 is a command actually issued to an actuator.

G0 vs G1 Libraries: The terms G0 and G1 can be used to conceptually separate the source of constraints in this hierarchy. G0 refers to the fixed global library constraints – essentially the hard boundaries and institution-set rules (many enforced at L-0/L-1). G1 refers to the dynamic overlay – the patient-specific constraints and learned parameters (which influence L-1 and also L-2 behavior). For example, if the hospital library (G0) sets a vasopressor dose hard limit at 30 μg/min, that is a non-negotiable L-0 cap. Meanwhile, the patient-specific overlay (G1) might determine that for this patient a more appropriate soft limit is 25 μg/min to avoid tachyarrhythmia, and L-1 will enforce that as the working limit. In contrast, for another patient who is very resistant, the overlay might raise the soft limit to 35 μg/min, but L-0 would still cap absolute max at 30 – unless an authorized override temporarily lifts it. Thus, G0/G1 interact such that **G1 can tighten but not break G0 rules** under normal operation.

The guard-rail layer (L-1) is essentially where G0 and G1 libraries are applied. It runs a **solver or rule engine** that takes all active infusions and their candidate commands (from L-2) and computes the adjusted commands. It solves inequalities representing limits (e.g., dose $\leq$ min{G0_limit, G1_limit}) and relational constraints (like maintaining a ratio A:B between two drugs within a range, or ensuring if drug X is on, drug Y must not exceed Z). This can be formulated as a constrained optimization or simply a priority rule set. One approach described in our system is a **multi-tier signal selector**: for any actuation decision, if multiple layers provide input, the system selects the one from the highest-priority layer that is "safe"[65]. In effect, L-1 provides a limiting function: it will pass through the L-2 command only if it lies within the safe window defined by the library and conflict rules; if not, it will substitute a scaled-back value or even zero (to pause) from L-1's own logic. Meanwhile, L-0 is watching for extreme conditions (for instance, if a measured variable is in a red zone, L-0 might generate its own override command – e.g., set infusion to zero). These arbitration rules guarantee that **safety dominates over performance** in all decisions.

A concrete example of the conflict arbitration: Suppose the patient is receiving both an insulin infusion to control blood glucose and a potassium infusion to correct hypokalemia. It is known physiologically that insulin drives potassium into cells, so giving insulin can worsen hypokalemia. The system's conflict graph encodes this interaction. If both insulin and K^+ cartridges are running in closed-loop (insulin trying to lower glucose, K^+ trying to raise serum potassium), the guard-rail layer will recognize the antagonistic relationship. If the insulin loop (perhaps higher priority to avoid immediate glucose danger) increases insulin delivery, L-1 may proactively **knock down** the K^+ infusion rate to prevent a crash in K^+ levels. This happens automatically according to the predefined incompatibility graph edge between those therapies. Conversely, if potassium is being rapidly infused, L-1 might clamp the insulin to minimal until potassium is safer. Without this, independent controllers might both act strongly and overshoot both targets. The three-layer arbiter in FIG. 16 encapsulates such logic, ensuring one loop can dominate if needed while the other yields.

Micro-Dosing "Slug" Mechanism for Learning

A highlight of this system is the **micro-dose slug** or probe mechanism (layer L-3) that enables active learning of patient response in real time. Traditional control systems would either require manual tuning or risk oscillation to learn a patient's dynamics. Here, by confining the learning input to a microscopically small dose change, the system obtains signal information without causing a clinical deviation.

FIG. 18 shows how these probe cycles occur over time[57]. Typically, when the controlled variable (say blood pressure or glucose) is well within target and all drugs are steady (no acute errors to correct), the controller initiates a probe: for a short duration, it will modulate one drug's infusion rate slightly up and down in a symmetric "tick-tock" pattern[5]. For example, it might increase vasopressor by +5% for one minute, then decrease it to -5% (below baseline) for one minute, then return to baseline. During this, it monitors the patient's blood pressure response. If the pressure rises by a measurable amount in the +5% phase and falls accordingly in the -5% phase, the system can estimate the derivative (slope) of pressure vs dose. This slope is one element of the Jacobian matrix (∂BP/∂Vasopressor). Repeating such tests at spaced intervals can refine the estimate and also capture non-linearities or time-lag (if any). The system uses a **recursive least squares** or similar adaptive filter to update the Jacobian continuously[66][67].

The term **μ-band** refers to the range within which these probes are done. The logic is that changes inside the μ-band are considered too small to harm the patient. How do we ensure that? The band is set based on known pharmacology and safety margins – usually a few percent of normal dose or a threshold below any effect level. Additionally, the controller monitors the patient during a probe; if any sign of deviation appears (like the variable drifting more than a tiny amount or any other vitals change), it aborts the probe and reverts. Over multiple probes, as the system becomes confident in the patient's sensitivity (confidence gating logic), it can reduce probe frequency or amplitude[8][68]. Eventually, when the model is very accurate (say the 95% confidence interval of the learned parameter is sufficiently tight), the system might stop probing that parameter and just maintain the learned value, only re-probing occasionally to verify no change. In this way, the controller **"locks on" to the patient's true response characteristics** typically within a few hours for fast-acting loops, or a couple of shifts for slower ones[59][69], without ever leaving safe operating regions.

For multi-variable interactions, the system can also leverage natural or induced variations in one drug to learn effects on multiple outputs. For example, if two drugs share effects on a variable (like sedative and opioid both affect respiratory rate), the system's multi-vertex learning can use one probe to gauge combined effect. The design even contemplates a **multi-vertex algebra** trick: if two pumps share a principal effect, alternating changes in one can indirectly inform the coupling of the other[70] (thus learning cross-effects faster). This is advanced and not required for basic operation, but underscores that the system actively seeks information to improve itself.

The **benefit of this micro-dose learning** is that over time, the G1 library overlay becomes highly tuned: the safe dose ranges and the expected effect per dose for each patient are known with increasing certainty. This means fewer alerts and more precise control. For instance, the controller can anticipate that "if I raise Drug A by X, variable Y will change by Z" for this patient, and thus avoid overshooting the target or triggering a secondary issue. It also allows detection of changes in patient condition – if the patient suddenly becomes less responsive to a drug (e.g., developing tolerance or a new metabolic issue), the micro-dose tests will start to show a different slope, and the system will detect this and adapt the therapy or alert the team to possible drug resistance.

Throughout this learning, **safety constraints are paramount**. The system uses multiple safeguards ("six tricks" as internally described) to ensure learning never compromises safety[71][8]: - It begins with a **population prior** – default control parameters drawn from a large dataset of patients, ensuring initial behavior is reasonable even if the patient-specific model is blank[71]. - It employs **confidence gating** – it does not allow the learned parameters to significantly alter therapy until they reach a certain confidence level[8]. - It uses **micro-probing** only within μ-bands as described. - It might incorporate **multiple models or sanity checks** (e.g., a mechanistic pharmacokinetic model echo alongside the learned model for cross-verification). - It has **fail-safes** where if sensors are noisy or the system cannot get a reliable reading (due to noise or artifact), it doesn't adapt at that time (thus avoiding chasing false data). - And crucially, the higher layers (L-0/L-1) remain in charge at all times – meaning even if the learning suggested something risky, the guard-rails would block it. In other words, the learning algorithm can only operate in the realm allowed by the base library and immediate safety monitors (hence the term **"hard-coat" safety** – a protective layer that is not altered by learning).

Continuous Quality Improvement (CQI) and System Feedback

Beyond individual patient adaptation, the system contributes to broader quality improvement. All infusion events (starts, stops, rate changes), alerts (limits hit, overrides), and outcomes are logged in a structured format. Hospitals can aggregate this data to identify patterns – for example, if a particular drug's soft limit is overridden frequently across many patients, it signals that the base library limit might be set too low for clinical reality[9][72]. Unlike current pumps that require manual retrieval of log data, this system can provide **real-time CQI reports**. It could automatically flag to pharmacy committees that "Drug X had 12 overrides this week in the ICU – consider adjusting the library." Additionally, because the system learns per patient, it can anonymize and feed this back into a central database to improve population priors. Over time, the institution builds a refined model of, say, "septic shock patients on Drug X typically need this dose range," which can be fed into future patients' starting settings.

In terms of immediate CQI feedback, the system also has the earlier mentioned override counter and escalation. This ensures that repeated deviations trigger review. The controller might even prompt the user after multiple overrides: "Would you like to adjust the limits for this patient going forward?" – essentially learning from the clinician if they consistently push beyond the recommended range. If the clinician confirms, the system can incorporate that as a semi-automated library update for that patient (under proper authorization).

All logs are cryptographically secured (see FIG. 3) so that they are admissible for compliance and auditing. This trustworthiness means the data can be used in incident investigations or for regulatory submissions. For example, should an adverse event occur, the system's ledger can show exactly what happened (each action hashed and time-stamped, forming an immutable chain[36][37]). This provides not only clinical value but also helps meet regulatory requirements for traceability (21 CFR Part 11 in the US, etc.).

Example Use-Cases

To illustrate the operation, consider a specific **ICU scenario**: A patient with septic shock is on multiple infusions: norepinephrine for blood pressure support, insulin to maintain tight glucose control, a sedative (propofol) for ventilation comfort, and intermittent doses of an analgesic. The patient also has sensors including an arterial line for blood pressure, a CGM for glucose, and periodic blood gases. The dynamic library system is deployed with cartridges for each infusion: - The **norepinephrine cartridge** has class "vasopressor", with G0 limits of, say, 0.01 to 0.5 μg/kg/min (soft) and 1.0 μg/kg/min (hard). The patient is currently needing 0.3 μg/kg/min to maintain MAP ~70. The system has learned overnight that increasing norepinephrine tends to also drop the patient's heart rate (due to baroreflex), and that the patient's sensitivity to norepinephrine on MAP is moderate. It adjusted the soft limit up to 0.6 because the patient needed 0.55 at one point. Now as the patient improves, it's titrating down while ensuring no abrupt hypotension. - The **insulin cartridge** has class "insulin", with standard sliding scale. The system coordinates insulin and dextrose (if available) to avoid hypoglycemia. It learned the patient's insulin sensitivity after a few hours of micro-boluses – for instance, how much 1 unit of insulin drops glucose. So it predicts and prevents overshoot. It also notes that when vasopressor dose is high (indicating stress), insulin requirement increases, so it started preemptively adjusting. - The **propofol (sedative) cartridge** is managed to keep a target sedation level (maybe by EEG or clinical score). However, propofol lowers blood pressure, which conflicts with norepinephrine's goal. The conflict graph in L-1 gives vasopressor maintenance higher priority than deepening sedation. So if blood pressure is at risk, L-1 will limit propofol increments or even lighten sedation if needed to preserve perfusion. It effectively finds a balance: the sedation target might be temporarily compromised to ensure a vital override criterion (MAP) is met. Once stable, sedation can increase again. - If the patient experiences an acute change (say a bleeding event causes blood pressure to drop sharply), L-0 triggers: It sees MAP below emergency threshold and instantly commands norepinephrine to max allowable and alerts for transfusion, etc., while possibly stopping propofol to avoid any further drop. This buys time for clinicians to respond, potentially saving the patient from cardiac arrest.

In such a use-case, the system acts like an ever-vigilant secondary clinician: adjusting drips every few seconds in a way no human realistically could, learning the patient's nuances, avoiding known pitfalls (like propofol causing too much hypotension or insulin causing hypoglycemia by coordinating with dextrose or reducing insulin if needed), and yet always allowing the human

team to step in and steer when they need to. The net effect is a more stable patient with fewer dangerous swings in vitals, fewer alarms ringing, and more confidence that the "routine" aspects of titration are handled.

Another use-case is in **operating room anesthesia control**: A patient under anesthesia might have automated delivery of anesthetic drugs using this system. The dynamic library overlay can personalize how much anesthetic is needed to keep a certain depth (perhaps using BIS or EEG). If surgical stimulation increases (heart rate spikes), L-2 increases analgesic or anesthetic accordingly, but L-1 ensures it doesn't overshoot to cause hypotension. If multiple anesthetics (like opioid and hypnotic) are used, the system ensures they stay in safe ratio (avoiding too much of both causing loss of blood pressure or respiration). This could operate as an autonomous anesthesia system with the anesthesiologist overseeing and available to intervene (and with an easy override knob as described).

Packet-Sniffing Shim for Legacy Infusion Pumps

Non-Intrusive Network-Tap Interface and Real-Time Infusion-Safety Middleware for Closed-Loop Medication Control

---

1. Field of the Invention

The present continuation relates to medication-delivery safety systems and, more particularly, to passive network-tap ("sniffer") hardware and software that retrofit existing infusion pumps without altering pump firmware or replacing drug-delivery cartridges, while still enabling the adaptive dose-error-reduction features, real-time rule-enforcement, and immutable audit-trail functions described in the parent application *Dynamic Personalized Drug Library and Closed-Loop Medication Control System*.

---

2. Background (Problem to Solve)

Hospitals have tens of thousands of installed legacy smart pumps whose proprietary firmware cannot be modified.
Although these pumps often expose standards-based data ports (e.g., HL7 IHE-IPEC, IEEE 11073), in practice many institutions have connectivity disabled due to cybersecurity concerns or license fees, leaving pharmacy departments blind to real-time programming violations (e.g., running high-alert medications in basic-rate mode).

Thus, there is a need for a *drop-in* retrofit that:

- Requires zero cooperation from pump vendors.
- Passively observes pump traffic—serial (RS-232), USB, or Ethernet—by means of a mirror-port, Y-cable, or optical tap.
- Decodes that traffic in real time, publishes a standards-compliant event stream to the SmartStop™ analytics engine, and enforces the adaptive guard-rails and closed-loop PID tuning disclosed in the parent spec.

3. Summary of the Invention

A packet-sniffing shim (tap module 1900) interposes transparently between the pump (P) and its downstream host (H) (e.g., bedside gateway or CIS). The shim:

1. Mirrors or forwards all bytes unmodified, preserving pump operation.
2. Filters only the infusion-related PDUs.
3. Decodes & normalizes those PDUs into a canonical schema.
4. Publishes the normalized events over TLS to the SmartStop cloud or on-prem broker.
5. Receives any rule-violation alerts or dose-adjustment commands and (optionally) injects them upstream as a standards-compliant message or bedside visual alert.

Because the tap is passive, installation is a five-minute cable swap; no pump recertification is required.

4. Brief Description of the Drawings

- FIG. 19 – High-level block diagram of the packet-sniffing shim and data path.
- FIG. 20 – Exploded view of the tap hardware showing microcontroller, FPGA-based line-sampler, and secure element.
- FIG. 21 – Real-time processing pipeline (capture → filter → decode → publish).
- FIG. 22 – Timing diagram showing round-trip latency < 150 ms from pump event to SmartStop alert.
- FIG. 23 – Alternate embodiment using an *optical splitter* on a gigabit uplink with no electrical break in the chain.
  (Reference numerals 1900-1999 are reserved for these new figures.)

5. Detailed Description of Preferred Embodiments

Ref. No.	Component	Function
1900	Tap Body	DIN-rail or bedside clip enclosure with ingress/egress connectors.
1910	Micro-controller / SoC	Runs lightweight RTOS; orchestrates capture and TLS publishing.
1920	Line Sampler	Dual-port SRAM & FPGA fabric for lossless 1 Gb s-1 capture with time-stamping (±1 µs).

1930	Protocol-Decode Engine	Plug-in modules for HL7 v2, ISMP R-IPEC, IEEE 11073-10101, proprietary vendor frames, etc.
1940	Crypto Co-Processor	Stores x.509 certs; signs outgoing audit frames; decrypts OTA policy updates.
1950	Secure Pub-Sub Gateway	MQTT 5 over TLS 1.3; publishes *InfusionStarted*, *DoseEdited*, *KVO*, *LibraryBypassed* topics.
1960	Real-Time Analytics Node (SmartStop Cloud)	Runs ML weight engine (parent numerals 40/60) and rule-violation logic.
1970	Alert & Enforcement Module	Generates FHIR-R4 *DetectedIssue* resources, SMS pager alerts, or HL7 NMD messages back to pharmacy.
1980	Optional VLAN Bridge	For sites that wish the tap to also *inject* soft-interlocks (stop/pause) back to pump if vendor API permits.
1990	Immutable Ledger Sink	Hash-chains (parent numerals 300-340) persist all captured frames ± alerts.

5.1 Passive Capture Modes

- Serial Y-cable – classic RS-232 RX line fan-out (no load on TX).
- USB-C "T-capture" – inline hub presenting as billboard class; host and device negotiate normally.
- RJ-45 Mirror-Port – switched 10/100/1000 with FPGA doing in-place packet replication (Rx) while forwarding untouched traffic.
- Optical Split – 50/50 LC duplex splitter; no power cycle required.

5.2 Processing Flow (FIG. 21)

graph LR

A[Wire Capture] --> B[Frame Filter 1920]

B --> C[Protocol Decode 1930]

C --> D[Canonical JSON]

D --> E[Secure Publish 1950]

E --> SmartStop[SmartStop ML Engine 1960]

SmartStop -->|Alerts| F[Pharmacy Dashboard / Mobile App]

Latency budget:

- Capture & timestamp ≤ 2 ms
- Decode & JSON marshal ≤ 15 ms
- TLS handoff & broker hop ≤ 50 ms (edge)
- ML rule evaluation ≤ 50 ms
- Push notification ≤ 30 ms
- **Total** ≤ 150 ms worst-case (meets "near-real-time").

5.3 Security & Compliance

- Tap is *read-only* by default; write-capable firmware requires cryptographically-signed unlock.
- FIPS 140-3 L2 secure element (1940).
- Syslog over TLS exports to SIEM.
- NIST SP 800-53 "AU-6(1)" immutable audit-trail satisfied via parent hash-chain.

5.4 Alternate Embodiments

1. **Wi-Fi Bridge** – tap relays over WPA3 network when no Ethernet drop exists.
2. **BLE Beacon Mode** – for home-health pumps; beacon only sends *LibraryBypassed* events to patient's companion app.
3. **Cloudless Edge-Inference** – ML weight engine pushed to on-tap TensorSoC; useful in air-gapped wards (oncology, pediatrics).
4. **Auto-Learning Decoder** – uses Bayesian grammar induction to reverse-engineer proprietary frames; continuously improves field decoding accuracy.

---

7. Advantages Over Prior Art

- **Zero pump downtime** – install during routine line-check; no need for vendor technician.
- **Vendor-agnostic** – same tap works on Pump A, B, or C; hospital stocks one SKU.
- **Regulatory friendly** – pump remains inside existing FDA 510(k) clearance; only the tap must meet IEC 60601-1-2 EMC and UL 60601 leakage.
- **Scalable** – one SmartStop instance supervises thousands of pumps across VLANs.
- **Enhances parent invention** – delivers the closed-loop adaptive benefits to legacy fleets years before refresh cycles.

---

8. Conclusion

This specification supplements the parent application by **fully enabling a passive, network-tap retrofit that unlocks SmartStop's adaptive safety logic for any infusion pump—no hardware swap required.** The embodiments, figures, and claim scaffolding herein are offered by way of example and may be combined, re-ordered, or substituted without departing from the scope defined by the appended claims.

These examples demonstrate how the invention can be applied to improve safety and outcomes in various clinical scenarios. The detailed description above, along with the figures, provides a full disclosure enabling those skilled in the art to implement the system in practice.

Industrial Applicability

The primary domain of applicability is in **healthcare settings** – especially hospital environments such as intensive care units (ICUs), operating rooms, emergency departments, and general wards where IV medications are administered with infusion pumps. The invention allows hospitals to enhance existing infusion systems with adaptive, patient-specific dosing control. Pharmacy departments can integrate the dynamic drug library into their medication safety workflow, using it to reduce dosing errors and alert fatigue while still enforcing institutional policies. Critical care patients stand to benefit from more stable vital signs and fewer adverse drug events due to the closed-loop, coordinated control (for example, maintaining optimal blood pressure and oxygenation with minimal manual titration). The system can also be applied in outpatient infusion settings (such as chemotherapy suites or home infusion devices) where patient conditions vary widely and safety is paramount.

Beyond human medical use, the architecture is **industrially applicable to analogues in process control**. The concept of intervention cartridges with sensor-feedback loops and hierarchical control is directly translatable to industries like biopharmaceutical manufacturing, chemical processing, water treatment, and so on[73]. For instance, in a bioreactor system, each "drug" cartridge might correspond to a reagent or nutrient feed; the dynamic library overlay becomes a recipe that adapts to process conditions (culture growth, pH, etc.), and the closed-loop control maintains optimal conditions. The multi-layer safety architecture (vital override, guard-rails, etc.) ensures that process variables stay within safe limits to protect the product and equipment. In fact, the system described herein was conceived to be broadly applicable – it can **miniaturize for medical therapeutics or scale up for industrial fluid control**[74]. Thus, any process that requires automated, adaptive dosing of multiple inputs based on sensor feedback can leverage this invention. Specific examples include: - **Hospital Pharmacy Automation:** Automated compounding systems or drug dispensing robots that adjust formulations in real-time for personalized medicine. - **Bioreactor Control:** As mentioned, dynamically dosing feeds, antifoams, pH adjusters in fermentation based on live sensor data (pH, dissolved oxygen, optical density) with a conflict graph ensuring, e.g., anti-foam addition pauses nutrient feeds to avoid interactions. - **Water Treatment Plants:** Dosing coagulants, chlorine, pH buffers adaptively to maintain water quality, with safety layers preventing overshoot that could violate regulations. - **Chemical Pilot Plants:** Multi-cartridge skids (as in some examples provided) where catalysts, stabilizers, inhibitors are added to a reaction in closed-loop, with a library of chemicals and rules

to avoid incompatible reagent mixes[75][76]. - **Food and Beverage Processing:** Precise addition of enzymes, flavors, or preservatives based on sensor feedback (quality metrics), using the hierarchical control to ensure product safety limits (e.g., regulatory maxima for additives) are never exceeded.

In summary, the invention finds use anywhere precise, safe, and adaptive control of fluidic dosing is needed. Its design is robust and flexible enough to handle human physiological systems as well as industrial processes, demonstrating industrial applicability across biotechnology, chemistry, environmental engineering, and manufacturing domains[73]. The descriptions and claims herein therefore extend to **both medical and non-medical implementations**, recognizing that the underlying control architecture and safety features function analogously once the "patient" is generalized to a "process" and "drug" to a "reagent"[77][78].

Enablement and Best Mode Implementation

The inventors have built and tested a prototype of the system, which they consider the best mode of carrying out the invention as of this filing. In this embodiment, each smart cartridge is built using readily available medical-grade components, and the control logic is implemented on a combination of firmware and edge computing hardware:

Hardware Configuration (Best Mode): A central control unit is implemented on an NVIDIA® Jetson Xavier NX module (or similar edge AI computer) running a Linux OS (Ubuntu 22.04) and a real-time extension for critical tasks. The Jetson NX provides substantial computing power to run predictive models (for L-2 optimization and L-3 learning)[79][80]. It communicates with up to 16 smart cartridges via a CAN bus or RS-485 bus using a custom protocol built on top of Modbus-TCP for reliability[79][81]. Each smart cartridge in this prototype uses a Bartels Mikrotechnik mp6 micropump (5 µL per stroke, calibrated to ±2% accuracy)[82]. The cartridge body is machined from USP Class VI PEEK plastic and includes a dual LED optochemical sensor (PreSens brand PSt3 patch) that measures pH and dissolved oxygen in the fluid stream[82][83]. This sensor data is timestamped by an onboard STM32G0 microcontroller on the cartridge, which also monitors a small inline flow sensor for redundancy. The cartridge's 2-kB FRAM (ferroelectric RAM) stores the drug library fields as described, and a 13.56 MHz NFC tag (NTAG 424 DNA) is integrated to hold the cryptographic certificate for authentication[84][85]. The quick-connect is an ISO 80369-7 compliant Luer lock with an integrated electrical pogo-pin array, allowing for one-step mechanical and electrical connection[86][87]. A small lithium-polymer battery on the controller provides ~15 minutes of backup power in case of mains failure, during which the system can maintain sensing and logging (and possibly minimal critical infusions)[88].

Firmware and Software: The cartridge firmware (STM32) handles low-level safety: it has a watchdog that counts pump strokes and verifies expected flow vs actual (from the flow sensor); if a deviation >5% persists for more than a few seconds, it raises a fault to the controller[89][90]. It also will automatically close a tiny pinch valve in the cartridge if it ever loses communication with the main controller or if a tamper event is detected, as a fail-safe to stop infusion. The main controller software is layered. A real-time thread cycles at 10 Hz to perform sensor reads and control updates for each active channel. PID control loops are implemented for fast-acting parameters (like blood pressure or pH) to provide smooth control within each second. For slower parameters or those better served by anticipative control (like glucose, which has delays), a

model-predictive control algorithm runs every few minutes using a lightweight TensorFlow Lite model that was pre-trained and continuously updated with the patient's data[79][80]. This **predictive model** (bi-directional LSTM in the prototype) looks 60 seconds ahead for pH control in the lab test scenario and triggers micro-doses preemptively if it predicts a deviation beyond tolerance[39][91]. The model is periodically validated against stored data via a digital twin routine (as per FIG. 6) to ensure it doesn't drift beyond reality[39].

The hierarchical arbiter is coded as a state machine: it has states corresponding to S0 (normal operation), S1 (minor alarm or soft limit exceeded), S2 (soft override active or moderate alarm), and S3 (lockout hard alarm)[92][93]. Depending on state, different layers are allowed to output. For instance, in S0 all layers function normally; in S1 or S2, some optimization commands are ignored and only guard-rail adjustments are allowed; in S3, everything is halted by L-0 and only manual override (L-5 in some descriptions) can release it[94][95]. The override interface in the prototype is a single physical rotary push-button on the device, combined with an RFID badge reader. A clinician taps their badge (the system reads it via NFC and verifies credentials against an internally stored list or via the hospital LDAP through network)[96][97]. The clinician then rotates or presses the knob to select an override option (e.g., extend an infusion limit temporarily or silence an alarm)[98][19]. The system's Policy Decision Point (PDP) logic checks the request (role, reason, current state) and either grants or denies within 200 ms[99][100]. If granted, a timer is started and the state moves to OVERRIDE-T (timed override) for that channel, visual indicators update (screen and an LED on the pump glows amber), and the system automatically re-arms (returns to normal) when the timer expires[18][101]. We found this single-knob override design with timed auto-rearm effectively prevented "alarm fatigue" overrides from being left in an unsafe state – caregivers liked that the system would gently beep and revert unless they actively extended the override.

For security, every data packet from the cartridges is signed using HMAC-SHA256 in the cartridge microcontroller before sending, and the main controller verifies it. The audit log (FIG. 3 process) is implemented using a local SQLite database on the Jetson and an off-device Hyperledger Fabric blockchain. Each infusion event JSON includes fields for timestamp, cartridge ID, action taken, previous hash, etc., which are hashed and the hash stored on the blockchain ledger maintained by a hospital server[36][35]. If network is unavailable, the system buffers logs and ensures to sync once reconnected (and as an extra, the device will not allow a software update or library change unless it can sync to the ledger to prove nothing was tampered in the interim). This meets FDA's expectations for medical device data immutability in post-market surveillance and cybersecurity.

In testing this best-mode setup on a simulated patient (in silico models and in vitro fluid testbeds), the system successfully maintained key parameters within target ranges >95% of the time, and responded to induced disturbances (like sudden change in "patient" sensitivity or setpoint) without any safety limit violations[102][103]. For example, in a closed-loop pH control of a 100 L bioreactor (an analog test), the system held pH 7.20 ± 0.03 and reduced buffer consumption by ~12% compared to a PID-only scenario[104]. These results demonstrate the efficacy of the layered approach: the optimizer and learning layers improved performance, while the guard and safety layers ensured tight safety bounds.

The best mode described above, including specific components and protocols, is an illustration of how the invention can be implemented with current technology. It is not intended to be

limiting; those skilled in the art could substitute equivalent components (e.g., use a different microprocessor or pump type) or adapt the software to different programming languages or operating systems. The inventors have emphasized the modularity of the design to allow future **"plug-and-play" upgrades**: for instance, adding new sensor types to the system simply involves introducing a new cartridge with that sensor and updating the library metadata for how it feeds into control logic, without redesigning the whole system[105][106]. This flexibility is considered part of the best mode, as it future-proofs the platform for evolving medical and industrial needs.

In conclusion, the Dynamic Personalized Drug Library and Closed-Loop Medication Control System has been described in detail through its architecture, components, algorithms, and a working embodiment. It fully enables any person skilled in the relevant art to make and use the invention, providing the best mode as a practical example. This system represents a significant advancement in infusion technology, marrying the reliability of engineered safety constraints with the adaptability of modern AI-driven control, to deliver smarter and safer medication therapy.