Decentralised Manufacturing as a Networked Cyber-Physical System: Formalising Free and Open Source Software Governance and ML Adaptation for Distributed Robustness

Bruno Dogančić *, Jurica Rožić, Marko Jokić, Marko Čeredar

Posted Date: 17 March 2026

doi: 10.20944/preprints202603.1360.v1

Keywords: decentralised manufacturing; cyber-physical systems; systemic robustness; open-source ecosystems; distributed control; digital fabrication; FOSS governance; disturbance rejection

 doi:10.20944/preprints202603.1360.v1

Article

Decentralised Manufacturing as a Networked Cyber-Physical System: Formalising Free and Open Source Software Governance and ML Adaptation for Distributed Robustness

Bruno Dogančić *, Jurica Rožić, Marko Jokić and Marko Čeredar

Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb, Ivana Lučića 5, 10000 Zagreb, Croatia
* Correspondence: bruno.dogancic@fsb.unizg.hr

Abstract

Decentralised manufacturing is expanding as digitally controlled fabrication tools become accessible to SMEs, independent operators, and community workshops outside traditional factory settings, but the resulting heterogeneous, autonomously operated network introduces systemic uncertainty that no central authority governs. This paper proposes a systems-theoretic framework in which Free and Open Source Software (FOSS) governance acts as the structural interoperability layer of a distributed cyber-physical manufacturing system (CPS), and node-local digital twins — each hosting a machine learning (ML) disturbance estimator — provide local adaptive compensation without centralised data aggregation. A defining property of the architecture is automatic improvement propagation: learned corrections distribute via federated learning to structurally similar nodes without operator intervention, and the open, observable FOSS ecosystem enables advances in one fabrication modality to transfer to others through shared interface standards. The framework is applied analytically to three disturbance classes: regulatory restriction, technical process variability, and supply-chain disruption. Across cases, the analysis shows how open modular interfaces and local adaptation preserve functional continuity under perturbations that would more strongly affect centralised architectures. The contribution is a unified mathematical basis for robustness analysis in decentralised manufacturing CPS and a foundation for future simulation and empirical validation.

Keywords: decentralised manufacturing; cyber-physical systems; systemic robustness; open-source ecosystems; distributed control; digital fabrication; FOSS governance; disturbance rejection

MSC: 93A14; 93D09; 93B70; 93E03; 68T05

JEL Classification: L17; O33; D85

1. Introduction

Manufacturing at the level of individuals, homes, and small-to-medium enterprises (SMEs) is increasingly accessible — encompassing desktop-scale additive manufacturing systems, CNC machines capable of manufacturing parts from lighter metals, laser cutters, and small-scale casting equipment. This shift is redistributing production capability away from centralised industrial facilities toward distributed workshops and small operators, fundamentally changing who can manufacture, and under what conditions. Such a change enables production to occur across geographically dispersed and technologically heterogeneous nodes, forming a decentralised manufacturing landscape in which digital designs, software toolchains, and local manufacturing capabilities interact to produce physical artifacts [1].

In this paper, decentralised manufacturing refers to the broader socio-technical shift in which manufacturing capability and decision-making move away from centralised industrial actors toward

 doi:10.20944/preprints202603.1360.v1

individuals and small entities. Within this paradigm, production occurs through distributed manufacturing systems, i.e., networks of geographically and technologically heterogeneous fabrication nodes. Throughout this paper, manufacturing refers to the broader socio-technical system — encompassing design, governance, and supply chain — while fabrication refers specifically to the physical production process and the equipment layer at individual nodes.

A crucial role in decentralising manufacturing was in large part made possible by desktop 3D printing. 3D printing was widely adopted due to its low cost, global material supply chains, easily exchanged design models, and mature open-source software ecosystem. The idea that individuals could operate digital fabrication systems outside traditional factory settings made it widespread in the maker community. Community-driven projects such as RepRap and Prusa were instrumental in establishing a global ecosystem of open-source 3D printing. The same idea has since been extended to other fabrication modalities, such as CNC machining and laser cutting, which are increasingly integrated into home and semi-professional environments. Each individual using these tools, regardless of intent, contributes to a distributed digital fabrication stack — such participation expands access to advanced production capabilities, lowers barriers to innovation, and enables rapid, localized customization of products. Recent analysis by Rayna and West is useful here precisely because it complicates the highly optimistic narrative — they show that 3D printing has not yet made the leap from prototyping to direct manufacturing at scale, despite a decade of hype — identifying 'reverse salients' that have slowed the translation of 3D printing technologies into broader direct manufacturing applications [2].

However, decentralising manufacturing capability also introduces new engineering challenges. Unlike centralised industrial systems characterized by tightly controlled environments and standardized processes, decentralised fabrication networks exhibit higher variability in materials, machine calibration, operator expertise, and process control. For most operators, that variability has no obvious fix — the expertise to diagnose and compensate for it takes years to develop, and no systematic route to it currently exists. In such a setting, functional performance emerges under greater uncertainty, and production outcomes depend on loosely coupled interactions between digital information, software control, and physical processes. From a systems engineering perspective, decentralised manufacturing can be interpreted as a network of heterogeneous cyber-physical production nodes with limited global observability and no single supervisory controller. As nodes expose more telemetry through open interfaces, system design still faces a "big observability, small controllability" tension, where compositional assume–guarantee contracts can localise diagnosability and coordinate corrective actions without centralised control [3].

The variability across decentralised nodes maps onto a class of problems that robust control theory already has tools for. Closed-loop control of process properties is well-established in conventional fabrication — metal forming being one example [4] — but decentralised networks extend this across heterogeneous nodes with no shared plant model and no central supervisor [5]. The challenge is not the control theory itself, but applying it to a system that spans technical, organisational, and informational layers simultaneously.

FOSS ecosystems provide the structural layer that makes this tractable [6,7]. Their modularity enables distributed innovation through standardised interfaces — a dynamic already visible in open-source fabrication communities: an operator identifies a source of drift in their machine, corrects it locally, and commits the fix back to the shared codebase, simultaneously reducing local uncertainty and increasing observability for every other node in the network. The transparency of open-source governance distributes the burden of quality assurance across all participants — deviations from expected behaviour are more likely to surface, and be resolved, the more operators observe the system — consistent with Benkler's account of commons-based peer production [8]. At the node level, ML formalises these same mechanisms: rather than relying on an operator to identify and correct process drift, a trained model does so automatically — with the degree of autonomy governed by operator policy [9–11], and federated learning [12] allows those corrections to propagate back through the network in the same way a committed fix would. For an individual node, this propagation is

 doi:10.20944/preprints202603.1360.v1

automatic: a calibration improvement contributed anywhere in the network reaches structurally similar nodes without each operator needing to discover, evaluate, or apply it independently — functionally equivalent to a package manager distributing updates to all devices sharing a common software base.

Distributed manufacturing, FOSS governance, and cyber-physical systems each have a substantial body of research behind them, yet **no existing work formalizes open-source governance as a structural layer within a CPS control architecture for manufacturing**. This paper takes a step toward treating open-source governance **not as background context but as a structural component** within the CPS control architecture itself. Baldwin and Clark's design-rules framework establishes that modular architectures enable independent development of subsystems through standardized interfaces [6], a principle that has held up across two decades of work on distributed innovation [13]. Baldwin's recent extension of this framework [7] explicitly addresses how open modular architectures enable self-organizing production ecosystems — the governance mechanism this paper argues decentralised manufacturing networks need. In this regime, FOSS ecosystems are the practical implementation of those design rules — their standardized interfaces and transparent governance are what allow heterogeneous nodes to interoperate.

Treating each fabrication unit as an autonomous agent in a networked cyber-physical system (CPS) offers a concrete way to address this uncertainty [14–16]. In this framing, CPS carries the physical-computational infrastructure and FOSS provides the governance layer. At each node, a **Cognitive Gateway** — a software layer co-located with the node's compute substrate, whose components are developed formally in Section 3 — acts as an automated substitute for the experienced operator: it tracks how the machine performs against a physics-informed nominal model, estimates and compensates for process drift, and governs which incoming FOSS updates are accepted. A dedicated disclosure function controls what leaves the node boundary, so only a privacy-preserving quality summary reaches the FOSS network while raw process data remains local. Nodes sharing similar machine architectures can then propagate learned corrections through federated learning [12], enabling *a priori* uncertainty reduction: a structurally similar node benefits from accumulated experience elsewhere in the network before collecting any data of its own [17]. This makes it possible to maintain quality standards without centralised supervisory control — and renders the system structurally resilient to both process variability and restrictive regulatory interventions such as those proposed in WA House Bill 2321 and similar legislation [18–20].

This paper presents an analytical systems engineering framework for treating decentralised manufacturing as a distributed cyber-physical production system (CPPS; abbreviated CPS throughout) under uncertainty. The argument developed in this paper is that decentralised manufacturing is to be understood as a distributed control problem — and that treating it as such opens up tools from systems engineering that the field has not yet applied. Using control theory metrics — primarily robustness and network connectivity — we can characterize the opportunities and constraints of decentralised manufacturing in terms of how it manages uncertainty across a network of heterogeneous nodes. This perspective also highlights the critical role of FOSS governance as a structural layer that enables coordination without centralised control, and the potential for ML/AI techniques to enhance local node performance while the architecture as a whole preserves global system robustness.

Contributions. This paper makes three primary analytical contributions:

1. **Formal CPS model:** A state-space formulation of each fabrication node as a stochastic cyber-physical system, with a unified taxonomy of governance (d_{gov}), technical (d_{tech}), and supply-chain (d_{sc}) disturbances acting at distinct layers of the network.
2. **FOSS-CPS architecture:** A structural framework in which FOSS governance functions as the interoperability constraint and coupling layer, combined with a node-local Cognitive Gateway — housing the digital twin nominal model, ML disturbance estimator ML_i, and Admission Controller Ω_i — as the adaptive compensation and update-governance mechanism.

 doi:10.20944/preprints202603.1360.v1

3. **Cross-disturbance analysis:** Three analytical case studies applying the framework to qualitatively distinct disturbance classes, revealing complementary rejection mechanisms and causal couplings across cases that single-domain analyses cannot capture.

Paper structure. Section 2 reviews foundational work on decentralised manufacturing, CPS, FOSS ecosystems, and ML in distributed production. Section 3 develops the formal framework: node-level stochastic dynamics, the FOSS interoperability constraint, the Cognitive Gateway architecture, and systemic robustness and resilience metrics. Section 4 applies the framework to the three disturbance classes through Case A (governance), Case B (technical process variability), and Case C (supply-chain disruption), followed by a cross-case comparison. Section 5 synthesises the findings, discusses design and policy implications, and identifies limitations and future directions. Section 6 concludes.

2. Background and Related Work

This section reviews the foundational concepts that underpin the proposed framework: the evolution of decentralised manufacturing paradigms, the systems-theoretic treatment of cyber-physical systems, the role of open-source ecosystems as coordination mechanisms, and the emerging application of machine learning in distributed production environments.

2.1. Decentralised Manufacturing and Industry 4.0

The traditional manufacturing paradigm is predicated on centralised control: a single entity governs the design, production, and quality assurance pipeline from end to end. Industry 4.0 has introduced a competing vision in which production is distributed across networked, autonomous agents — a shift that has been extensively analysed through the lens of systems theory [5]. Digital fabrication technologies — such as additive manufacturing (FDM, SLA, SLS), CNC machining, laser cutting, and small-scale forming and casting — have collectively lowered the capital barrier to entry, enabling geographically dispersed individuals and small enterprises to participate in manufacturing networks that were previously the domain of large-scale factories.

Digital twins — physics-informed virtual models of physical assets, continuously updated with operational data — have established themselves in model-based systems engineering (MBSE) as tools for lifecycle management, fault diagnosis, and predictive maintenance [14,21,22]. Classical DT architectures, however, presuppose centralised data aggregation: a single authoritative model ingests telemetry from all assets and coordinates responses globally. This incompatibility with decentralised manufacturing — where nodes are independently owned, heterogeneous, and under no obligation to share raw process data — motivates assigning digital twins a *node-local* role rather than a network-orchestrating one. In the framework developed here, each node's **Cognitive Gateway** hosts a physics-informed digital twin nominal model alongside an ML disturbance estimator and an Admission Controller, with a dedicated Disclosure Function producing a privacy-preserving quality summary as the node's sole output to the FOSS governance layer [15]. CPS infrastructure and FOSS governance handle the network-level coordination that classical DT architectures achieve through centralisation.

Several recent contributions have moved toward treating distributed manufacturing as a structured system rather than a purely economic phenomenon. Kostakis et al. proposed the "Design Global, Manufacture Local" model, in which digital commons of knowledge and design are combined with localised fabrication, but analysed it as a political-economic paradigm without formal systems modelling [23]. Bonvoisin et al. mapped modularity types in open-source hardware, extending Baldwin and Clark's design-rules theory to tangible artefacts, yet stopped short of modelling how that modularity governs a manufacturing network [24]. Mariscal-Melgar et al. proposed a node-based conceptual architecture for distributed open-source hardware production — the closest existing work to a structural model — but did not invoke cyber-physical systems formalism or control theory [25]. Collectively, to our knowledge, existing literature treats FOSS governance either as a social coordination mechanism or as an innovation-management variable, with limited formal treatment as a **structural layer** within a CPS control architecture. This paper addresses that gap by formalising open-source governance as a

 doi:10.20944/preprints202603.1360.v1

coupling and disturbance-rejection mechanism within a control-theoretic framework for decentralised manufacturing.

2.2. Cyber-Physical Systems and Distributed Control

Cyber-physical systems (CPS) integrate computational and physical processes through networked feedback loops [26]. The systems engineering literature has developed mature frameworks for analysing CPS properties such as observability, controllability, and robustness — properties that translate directly to the decentralised manufacturing context but have rarely been applied to it.

Across a decentralised manufacturing network, nodes differ in process class, mechanical architecture, calibration state, and operational history, constituting a heterogeneous multi-agent system in which maintaining output consistency becomes structurally analogous to the distributed consensus problem [27]. Unlike classical consensus settings, however, the heterogeneity here is not merely topological but parametric: each node's dynamics are governed by a distinct operating point, disturbance distribution, and process-class-specific uncertainty structure [28,29]. This motivates modelling each node as a linear state-space system at its nominal operating point, with residual variability absorbed into the stochastic disturbance term and addressed by the node-local ML disturbance estimator.

Structure-preserving robustness analysis for interconnected dissipative systems has demonstrated that modelling uncertainty at the subsystem level yields less conservative robustness guarantees than monolithic approaches [30]. The decentralised manufacturing network considered here extends this principle from the physical domain (spatially distributed mechanical systems) to a cyber-physical domain where the interconnection structure is governed by open-source software protocols rather than physical coupling elements. In this case, the structure-preserving decomposition maps naturally to per-node uncertainty in process parameters, firmware configurations, calibration states, and process-class-specific variability sources.

Akkaya et al. demonstrate that distributed heterogeneous CPS require aspect-oriented modularity to manage intersecting concerns — firmware, communication protocols, and physical process models — without a unified supervisory architecture [31]. This architectural requirement — separation of concerns with well-defined interfaces — is precisely what FOSS governance systems provide at the ecosystem level, enabling different types of fabrication nodes to be developed and integrated independently.

Distributed control theory offers well-established tools for this class of problem. Consensus protocols, formation control, and cooperative tracking all build on the principle that agents can coordinate on shared objectives without central supervision [27]. These ideas have already been applied to manufacturing through the cyber-physical production systems (CPPS) paradigm: Monostori et al. survey how embedded intelligence and networked feedback enable factory-level self-configuration [28], and Leitão et al. show multi-agent architectures in which production units negotiate schedules and adapt to disruptions in real time [29].

The limitation of this body of work is its scope. CPPS research almost invariably assumes a **smart factory** setting — a single facility with unified ownership, standardised equipment, and centralised data governance. The coordination problems it addresses remain within the factory boundary. The progression from passive to autonomous fabrication nodes mirrors the digital twin maturity spectrum identified in recent reviews, from disconnected model-only twins to fully autonomous, self-optimising units [16,32] — yet even such comprehensive taxonomies remain confined to the factory paradigm. **Decentralised manufacturing networks look fundamentally different**: nodes are independently owned, heterogeneous, and geographically dispersed, with no single supervisory authority, no guaranteed communication topology, and no uniform equipment baseline.

2.3. FOSS Ecosystems as Design Rules for Distributed Manufacturing

The role of Free and Open Source Software (FOSS) in manufacturing extends beyond providing free tools. From a systems perspective, FOSS ecosystems function as **Design Rules** in the sense of Baldwin and Clark [6]: they define the modular architecture that enables independent agents to

 doi:10.20944/preprints202603.1360.v1

produce interoperable outputs without hierarchical coordination. This architectural role has been underappreciated in the manufacturing systems literature, which has tended to treat open source as an implementation detail rather than a structural governance mechanism.

In the 3D printing ecosystem, for example, the full fabrication pipeline — from parametric design (OpenSCAD, FreeCAD) through process planning (PrusaSlicer, Cura) to machine control (Klipper, Marlin) — is governed by FOSS with standardised interchange formats (STL, 3MF, G-code) at each interface boundary. Analogous open-source tool-chains exist for other fabrication modalities: for CNC machining, the pipeline spans parametric CAD (FreeCAD) through computer-aided manufacture (FreeCAD Path workbench, CAMotics) to motion control (LinuxCNC, Grbl); for laser cutting, vector toolpaths generated in Inkscape are executed by open controllers such as Grbl or Smoothieware. A curated reference collection of FOSS tools constituting this governance layer — spanning CAD, CAM, firmware, machine control, simulation, federated learning, and open hardware designs — is maintained as companion material to this paper [33]

This layered, modular structure exemplifies Baldwin and Clark's Design Rules: each component can be independently developed and substituted without disrupting the pipeline, provided it respects the interface specifications. The importance of such ecosystems for enabling distributed manufacturing has been noted in the literature [23–25], yet their role as a formal structural layer within a CPS control architecture remains unexplored. Notably, even comprehensive reviews of distributed manufacturing and CPS do not consider FOSS ecosystems as a structural enabler — a gap that underscores why this paper formalises open-source governance as a first-class layer within the CPS control architecture.

A structural property of this ecosystem is the asymmetry between contribution and benefit: a single quality improvement — a calibration profile, a firmware patch, a slicer parameter set — propagates immediately to every node drawing from the shared repository, irrespective of whether those nodes actively participate in development. Most network participants benefit from advances they had no part in producing, without any explicit coordination, making the effective uncertainty reduction across the network substantially larger than the active contributor base alone would suggest.

The choice of FOSS as the governance layer follows from the specific conjunction of properties the framework requires. Proprietary coordination platforms can enforce interface standards, but at the cost of single-point control and opacity. Decentralised but proprietary protocols can achieve redundancy, but without the transparency that distributes verification across participants. FOSS ecosystems provide all three simultaneously: open interface standards enabling interoperability across independently developed components, distributed hosting eliminating single points of control, and transparent community-governed codebases making quality assurance a shared rather than a managed function. Only their simultaneous presence qualifies FOSS as the governance layer this framework requires.

2.4. Machine Learning in Distributed Manufacturing

Machine learning has been widely applied to manufacturing quality control, predictive maintenance, and process optimisation in centralised factory settings. Federated learning [12] offers a framework for training ML models across distributed nodes without centralising sensitive data — a property that aligns naturally with the privacy-preserving, autonomous character of decentralised manufacturing networks. Under such a framework, each fabrication node can act as a local agent within a multi-agent system, learning to compensate for its own process variability while contributing to a collective performance objective through consensus-based model aggregation [34].

Within additive manufacturing, Gaussian processes and neural networks have proven effective for layer-wise quality prediction, thermal distortion modelling, and in-situ defect correction [9–11]. The same uncertainty-reduction principle extends across manufacturing modalities: ML-based thermal error compensation has been extensively demonstrated for CNC machine tools [35–37], hybrid ML models have been applied to process parameter optimisation in laser-assisted machining [38,39], and closed-loop feedback control has been validated in metal forming [4]. In each case the trained model replaces conservative tolerances with data-driven estimates of actual process variability at a specific machine.

 doi:10.20944/preprints202603.1360.v1

A common thread across this body of work is the assumption of centralised data access, which is well-suited to homogeneous factory settings but ill-fitted to independently operated distributed nodes. More fundamentally, none of these works treat ML as an uncertainty-reducing mechanism governed jointly by local process data and a global FOSS policy layer. In such a setting, each node trains locally to reduce its own process uncertainty, yielding models that are more accurate and less conservative; the federated aggregation objective is simultaneously shaped by community-set quality requirements and, where applicable, regulatory constraints — both propagated as policy signals through the same tool-chain API stack (design software, slicer, firmware) to individual agents. Policy compliance and quality improvement are thus two expressions of the same governance mechanism rather than separate concerns — a gap that motivates the framework developed in Section 3. There, each node's ML disturbance estimator is housed within the node's **Cognitive Gateway**: its digital twin nominal model isolates unmodelled process effects as a measurable residual (the twin gap $\delta_i[k]$) [14,15], giving the estimator a well-defined learning target rather than raw sensor streams. Nodes sharing the same machine architecture can then propagate learned corrections via federated learning [12], enabling *a priori* uncertainty reduction at the network level before a new node has collected any process data of its own [17] — provided that process characteristics are sufficiently similar across the federation, communication quality is adequate, and training data distributions are not excessively heterogeneous.

3. Proposed Framework: Decentralised Cyber-Physical Governance

Anyone who has hand-calibrated a 3D printer — tweaking flow rates, deciding whether to pull a community firmware patch, printing temperature towers, posting settings to a forum — has in part already run this framework manually: intuition as the disturbance estimator, judgement as the admission controller, and community profiles as federated learning at human speed. What follows names those roles, defines the interfaces between them, and makes the manual ones automatable — as presented in Table 1.

Table 1. Mapping between familiar FOSS manufacturing practice and the formal framework components.

What an experienced FOSS operator does	Proposed framework equivalent
Compensates for machine quirks by feel or accumulated experience	Digital Twin nominal model + ML disturbance estimator ML_i
Decides whether to apply a community firmware or configuration update	Admission Controller Ω_i
Posts calibration settings to forums or commits to a shared repository	Disclosure Function $\mathcal{F}$ and federated update $\Delta\eta_i$
Downloads a community profile before printing a new material	*A priori* uncertainty reduction via peer model updates
Uses PrusaSlicer / Klipper / G-code as the shared toolchain	FOSS governance layer $\Phi[k]$

Even though the practice exists, the unifying framework still does not. Without formal treatment, none of the above scales beyond a single operator or a small group. Most operators never reach the level of expertise the left column describes — and even those who do cannot scale their judgement to a heterogeneous network, derive computable robustness guarantees, or preserve node privacy in the process. For the newcomer who buys a first CNC machine and encounters systematic dimensional error, there is no clear path from observation to compensation without hard-won experience or costly proprietary support. The framework replaces each row with an automated, formally grounded equivalent: the expert's toolkit, available to any operator at any node, at network scale as a structural property of the system rather than an individual skill.

The right column of Table 1 names an automated equivalent for each manual step; it does not yet identify what performs those steps. That role falls to the **Cognitive Gateway** — a software layer co-located with the node's computing substrate that automates what the experienced operator currently

 doi:10.20944/preprints202603.1360.v1

does by hand, and makes that capability a structural feature of the network rather than an individual skill.

Figure 1 establishes the network-level architecture. N heterogeneous fabrication nodes — spanning additive, subtractive, small-scale casting equipment, and laser processes — are coupled through the FOSS governance layer $\Phi[k]$, which distributes design references, quality requirements, and process-selection guidance through open-standard APIs. External disturbances enter at distinct layers: governance pressure $d_{gov}[k]$ and supply-chain disruption $d_{sc}[k]$ act on the FOSS layer, while process variability $d_{tech}[k]$ enters each node individually. At each node, a **Cognitive Gateway** sits between the FOSS layer and the physical plant, providing the automated intermediary role that currently falls to the operator: receiving design references, compensating for local process variability, and reporting quality telemetry back to the network.

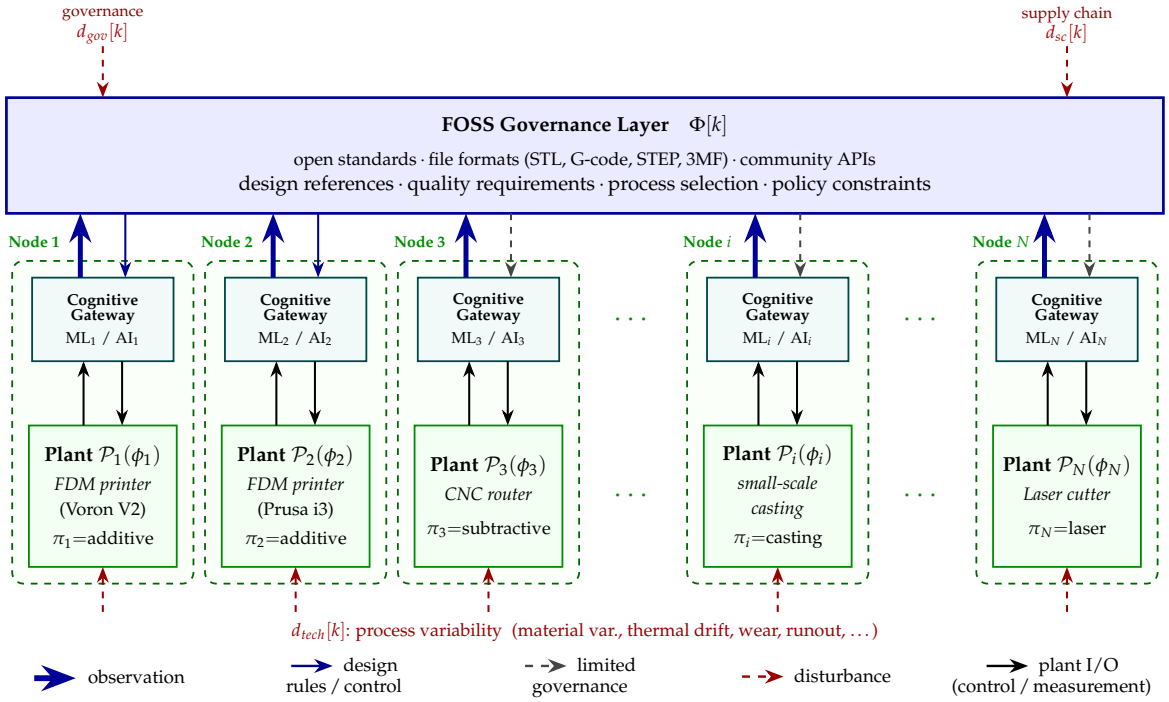

Figure 1. Network view of N heterogeneous fabrication nodes under FOSS governance. Each dashed outline is one node — comprising a Cognitive Gateway (ML/AI) and the physical Plant $\mathcal{P}_i(\phi_i)$. The Cognitive Gateway, whose internal structure is detailed in Figure 2, acts as the automated intermediary between the FOSS layer and the physical process. Thick blue upward arrows carry process observations to the FOSS layer; solid blue downward arrows deliver active governance signals (design rules and control) — the asymmetry between rich upward observability and variable downward control reach reflects the "big observability, small controllability" character of current open-source fabrication networks. Dashed downward arrows indicate nodes with limited or indirect governance. Black arrows between the Cognitive Gateway and Plant carry the adapted control signal $u_i[k]$ downward and the measurement $y_i^{\mathrm{m}}[k]$ upward; their internal routing is detailed in Figure 2. Process heterogeneity is made explicit through the process-class labels π_i.

The architecture visible in Figure 1 rests on a precise characterisation of the uncertainty each node faces. Every fabrication node has its own unique process variability — thermal drift, feedstock variance, mechanical wear, spindle runout, beam focus instability — that causes its actual output to deviate from any nominal model. If there is no measurement data, a controller has to assume a worst-case uncertainty bound and design conservatively: wide tolerances, careful process parameters, lower throughput. The effective uncertainty set gets smaller as a node collects local process data and fits a model to it. The controller can then exploit this tighter characterisation to achieve better performance without losing robustness. The proposed framework organises this intuition at network scale.

 doi:10.20944/preprints202603.1360.v1

Across a heterogeneous network this logic operates at two levels. The aggregate uncertainty space spanning all candidate process types and parameter regimes is vast; however, selecting the appropriate process for a given task collapses the majority of that uncertainty before any local compensation is needed. A flat metal profile is better laser-cut than FDM-printed; a threaded bore requires a lathe or CNC rather than a slicer. Each such allocation decision — reached by the operator or the node's Cognitive Gateway, drawing on the shared design rules, capability specifications, and community practice that the FOSS layer makes available — reduces the uncertainty envelope that the local ML controller must subsequently handle. The framework treats this two-level structure — process selection informed by the FOSS layer, residual compensation at the ML layer — as its core mechanism.

Figure 2 zooms into the internal structure of a single fabrication node i and details how these three layers interact at the signal level.

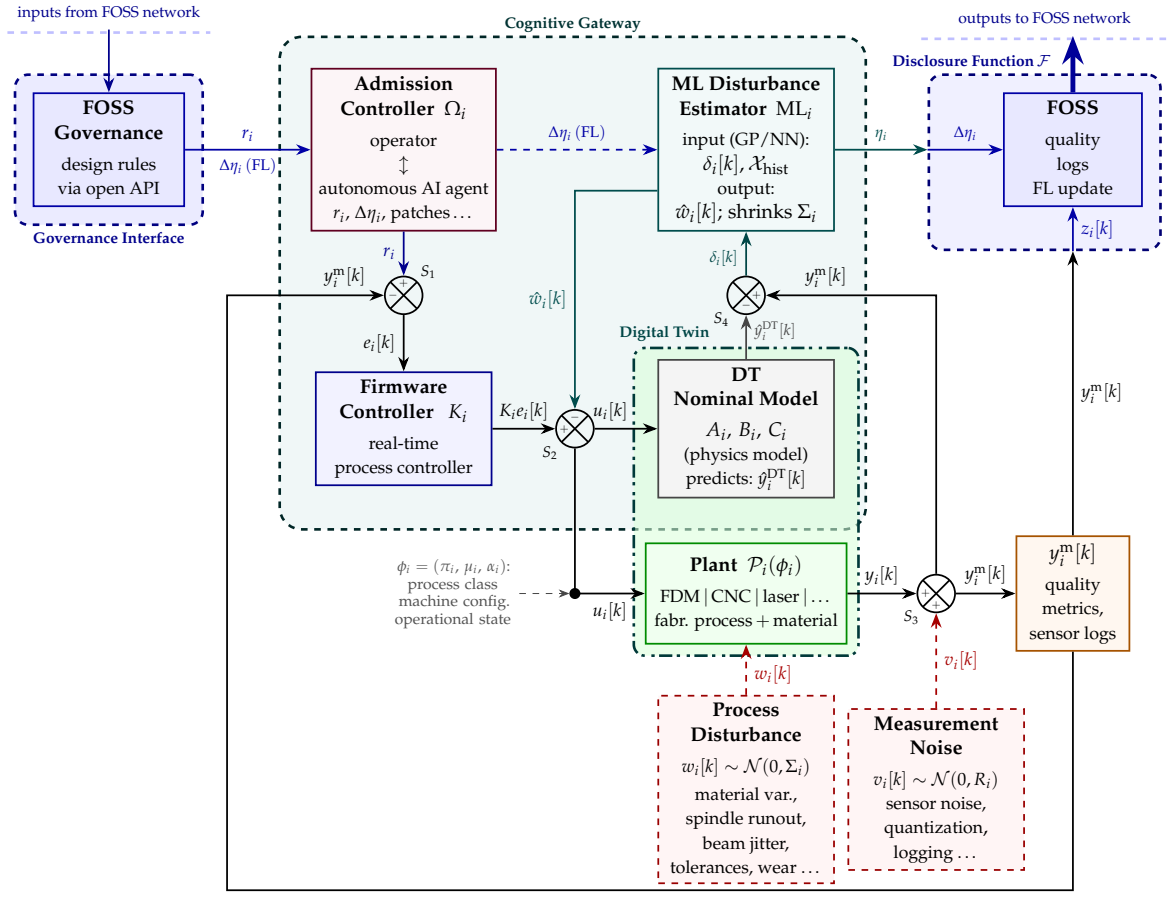

Figure 2. Single-node closed-loop architecture for fabrication node i. The **Plant** $\mathcal{P}_i(\phi_i)$ (solid green) is the sole hardware element; all other blocks are software co-located on the node's computing substrate. The **Cognitive Gateway** (outer teal dashed boundary) encloses the Admission Controller Ω_i, Firmware Controller K_i, ML Disturbance Estimator ML_i, and DT Nominal Model; the **Digital Twin** boundary (dot-dash-dot) spans the Nominal Model and the Plant. Junctions S_1–S_4 are the tracking-error, compensation, measurement, and twin-gap summing points, respectively. The **Governance Interface** and **Disclosure Function** $\mathcal{F}$ define the node's sole input and output channels to the FOSS network. Governing equations: (3)–(11).

The framework organises node behaviour across three interacting layers: the **FOSS governance layer** informs process selection and distributes design references; the **Cognitive Gateway** performs local adaptive compensation; and the **plant** executes the physical fabrication. The FOSS governance layer $\Phi[k]$ is rich in shared design rules, open interface standards, and quality criteria, yet issues no commands to individual nodes — coordination by protocol rather than by authority. At the lowest level, the physical plant — the fabrication machine itself — is subject to process-class-specific disturbances:

 doi:10.20944/preprints202603.1360.v1

thermal gradients and filament variance for FDM nodes, spindle runout and fixturing compliance for CNC, beam instability for laser cutting.

The intermediate layer — the **Cognitive Gateway** — is where the framework's core contribution lies. It is software running on the node's co-located compute substrate, automating what an experienced human operator would otherwise do manually: diagnosing process drift, fitting compensation models, and closing the gap between nominal design intent and physical output. Its ML disturbance estimator learns continuously from local process data and receives peer-network calibration knowledge via federated learning; the Admission Controller Ω_i governs a configurable spectrum from passive human review to a fully autonomous AI agent that applies accepted updates without intervention — progressively reducing the effective uncertainty Σ_i at each node.

The key architectural distinction in Figure 2 is between the sole hardware element — the **Plant** $\mathcal{P}_i(\phi_i)$ — and the surrounding software layer, the **Cognitive Gateway**, running on the node's computing substrate. The Plant is the only element subject to process disturbance $w_i[k]$ and measurement noise $v_i[k]$; the Cognitive Gateway mediates between the FOSS network and the physical process.

Signals enter through the **Governance Interface** (left boundary). The **Admission Controller** Ω_i gates every inbound FOSS signal along its configurable policy spectrum: the design reference r_i is admitted to the tracking-error junction S_1, while the federated update $\Delta\eta_i$ updates the ML Disturbance Estimator parameters as $\eta_i \leftarrow \eta_i + \Delta\eta_i$, pre-conditioning it with peer-network calibration knowledge. On the output side, the Plant produces $y_i[k]$, which is corrupted by measurement noise at junction S_3 to yield $y_i^{\mathrm{m}}[k] = y_i[k] + v_i[k]$. From this measured signal, three distinct feedback paths radiate.

The *hardware loop* feeds $y_i^{\mathrm{m}}[k]$ back to S_1, forming the tracking error $e_i[k] = r_i - y_i^{\mathrm{m}}[k]$ that drives the **Firmware Controller** K_i. At S_2, $K_i e_i[k]$ combines with the ML disturbance estimate $\hat{w}_i[k]$ to give $u_i[k] = K_i e_i[k] - \hat{w}_i[k]$ (Equation (8)) — the error-tracking structure present in any conventional closed-loop controller. The *twin-gap loop* routes $y_i^{\mathrm{m}}[k]$ to junction S_4, where it is compared against $\hat{y}_i^{\mathrm{DT}}[k]$ produced by the **DT Nominal Model** (A_i, B_i, C_i) running in parallel on the same input $u_i[k]$ (Equation (6)). The resulting twin gap $\delta_i[k] = y_i^{\mathrm{m}}[k] - \hat{y}_i^{\mathrm{DT}}[k]$ (Equation (7)) gives ML_i a well-defined learning target, progressively reducing Σ_i. The **Digital Twin** boundary wraps both the Nominal Model and the Plant, reflecting that the twin is a software–hardware pair. The *network feedback loop* routes $y_i^{\mathrm{m}}[k]$ and η_i to the **Disclosure Function** $\mathcal{F}$, which privacy-filters them into quality telemetry $z_i[k]$ and federated update $\Delta\eta_i$, giving $(z_i[k], \Delta\eta_i) = \mathcal{F}(y_i^{\mathrm{m}}[k], \eta_i)$ (Equation (11)) and propagating telemetry and model updates back to the FOSS network. The subsections that follow develop each layer mathematically.

3.1. Node-Level Stochastic Dynamics

Throughout this framework, *node* refers to the autonomous production unit participating in the FOSS-governed network. In the most direct reading this is a single fabrication machine — a desktop FDM printer, a CNC router, a laser cutter — with its Cognitive Gateway running on co-located compute, and the formal treatment that follows develops this single-machine case in full. The structural elements that define a node, however, are not inherently machine-scoped: the Admission Controller Ω_i governing inbound updates, the Disclosure Function $\mathcal{F}$ bounding outbound telemetry, and the federated learning participation channel are all defined at the *operator's production boundary* rather than the machine boundary. An operator running a small print farm, a workshop combining CNC and laser processes, or a FOSS-governed factory each participates in the network as a single node — one admission policy, one privacy boundary, one aggregate contribution to shared calibration knowledge. Provided the aggregate state of such a production unit admits a consistent state-space description, the framework applies at any of these scales; whether a hierarchical two-level formulation, in which sub-nodes exist within a node, better captures the relationship between machine-level and facility-level dynamics is left as a direction for future work.

The local physical fabrication process at node i is modelled as a discrete-time stochastic state-space system [40,41]:

 doi:10.20944/preprints202603.1360.v1

$$x_i[k+1] = A_i(\phi_i)\, x_i[k] + B_i(\phi_i)\, u_i[k] + w_i[k], \tag{1}$$

where:

- $x_i[k] \in \mathbb{R}^n$ represents the **plant state** of the i-th node (e.g., toolpath precision and thermal distribution for additive nodes; spindle load and surface-normal deviation for subtractive nodes; beam focus and kerf width for laser nodes);
- $u_i[k] \in \mathbb{R}^m$ is the **control input** derived from the digital design and software toolchain;
- $A_i(\phi_i)$ and $B_i(\phi_i)$ are the system matrices parameterised by $\phi_i = (\pi_i, \mu_i, \alpha_i)$, where π_i is the **process class** (e.g., additive, subtractive, forming, joining), μ_i is the **machine configuration** (kinematic architecture, actuator resolution, working envelope), and α_i is the **operational state** (age, calibration history, material system); together these encode the full node heterogeneity within a unified parameter vector;
- $w_i[k] \sim \mathcal{N}(0, \Sigma_i)$ is the **stochastic process noise** representing local uncertainties whose character is process-class-dependent (e.g., filament diameter variance and thermal gradients for FDM; spindle runout and fixturing compliance for CNC; beam power fluctuations and assist-gas pressure for laser cutting).

The measured output of each node is given by:

$$y_i^{\mathrm{m}}[k] = C_i\, x_i[k] + v_i[k], \tag{2}$$

where C_i is the output matrix and $v_i[k] \sim \mathcal{N}(0, R_i)$ is the measurement noise — the linear-Gaussian structure of (1)–(2) is the standard setup for optimal state estimation [42]. This linearisation is most accurate for process regimes that are locally smooth and disturbance-dominated rather than bifurcation-dominated; highly nonlinear phenomena such as thermal runaway, layer delamination, or abrupt tool failure lie outside its formal scope and would require extended or switching formulations. The measured output $y_i^{\mathrm{m}}[k]$ represents the observable quality characteristics of the fabricated artefact (e.g., dimensional accuracy, surface roughness, mechanical properties). Throughout the case studies, Φ (without time index) denotes the structural state of the FOSS ecosystem when its evolution is slow relative to node-level dynamics and can be treated as quasi-static; $\Phi[k]$ is used when its time-variation is under analysis.

3.2. FOSS as the Interoperability Constraint (Design Rules)

The interaction between nodes is governed by Free and Open Source Software (FOSS) ecosystems, which function as the "Design Rules" in the sense of [6,7]. Mathematically, these ecosystems enforce a **Common Interface Constraint** $\mathcal{I}$:

$$r_i = \mathcal{I}(D, \theta_i), \tag{3}$$

where D is the global digital design (the "source code" of the artefact), $\theta_i = \theta(\phi_i)$ represents the local transformation parameters derived from the node's physical characterisation $\phi_i = (\pi_i, \mu_i, \alpha_i)$ — toolchain configuration, material system, and process-class-specific settings — and r_i is the node-local design reference — the target trajectory that the firmware process controller K_i subsequently converts into control inputs $u_i[k]$. In practice, $\mathcal{I}$ subsumes the full CAD-to-fabrication backend: design file parsing, CAM strategy selection, slicer profile application, and G-code or equivalent instruction generation, all mediated through standardised open APIs (e.g., OctoPrint, Klipper's WebSocket interface, LinuxCNC's HAL). This backend is a product of the FOSS ecosystem and operates upstream of K_i; by the time r_i reaches the firmware controller, the toolchain work is complete. FOSS ensures that $\mathcal{I}$ is **invariant** across the network: any compliant node derives the same reference from the same design, providing the standardised protocols necessary for interoperability in the absence of a central supervisor.

 doi:10.20944/preprints202603.1360.v1

Although Figure 1 shows the FOSS governance layer connected to all nodes, this topology should not be read as a centralised star controller. The FOSS layer does not issue commands, allocate tasks, or maintain global state — it functions as a **protocol**: a shared set of interface standards, design rules, and quality criteria that each node voluntarily adopts. Connectivity in the figure reflects universal access to these open standards, not the existence of control channels. A centralised controller that becomes unavailable disables the entire network (Section 3.4); degradation of FOSS infrastructure, by contrast, reduces the interoperability index $\mathcal{C}(\Phi)$ without disabling nodes that have already incorporated the shared standards locally.

Network topology. The state of the FOSS ecosystem $\Phi[k]$ determines which pairs of nodes can exchange designs, calibration data, or process knowledge through shared open standards. This is encoded by the binary coupling matrix $A_c(\Phi) \in \{0,1\}^{N\times N}$, where $[A_c]_{ij} = 1$ if nodes i and j share an active FOSS-mediated link and $[A_c]_{ij} = 0$ otherwise [27,43]. When Φ is degraded — for example, by governance restrictions on software distribution or supply-chain disruptions that eliminate shared toolchains — A_c becomes sparser and the network fragments.

Interoperability index. To quantify the degree of network connectivity, we define

$$\mathcal{C}(\Phi) = \frac{1}{N(N-1)} \sum_{i\neq j} [A_c(\Phi)]_{ij}, \tag{4}$$

which counts the fraction of all possible ordered node pairs (i, j), $i \neq j$, that share a direct open-standard link [5,44]. For a symmetric coupling matrix this equals the edge density of the network graph. The index is bounded: $\mathcal{C} = 1$ means every node can directly exchange information with every other (fully connected open ecosystem); $\mathcal{C} \to 0$ signals progressive fragmentation into isolated clusters. Because governance disturbances d_{gov} and supply-chain disturbances d_{sc} both act on Φ — and hence directly on A_c — the index $\mathcal{C}(\Phi)$ serves as the primary robustness metric in the subsequent case studies.

Control-theoretic interpretation. The index $\mathcal{C}(\Phi)$ provides an intuitive count of active links, but its systems-engineering significance is made precise through the **graph Laplacian** $L(\Phi) = \mathrm{diag}(A_c(\Phi)\,\mathbf{1}) - A_c(\Phi)$, where $\mathrm{diag}(A_c(\Phi)\,\mathbf{1})$ is the degree matrix. The second-smallest eigenvalue of $L(\Phi)$, the **algebraic connectivity** (Fiedler value) $\lambda_2(L)$, provides the definitive connectivity criterion for undirected graphs [27]:

$$\lambda_2(L(\Phi)) > 0 \iff \text{network is connected.} \tag{5}$$

$\mathcal{C}(\Phi)$ is a density-based proxy: $\mathcal{C} = 0$ implies no edges and therefore $\lambda_2 = 0$, but $\mathcal{C} > 0$ does not guarantee connectivity — a graph can possess edges while remaining partitioned into disconnected components, yielding $\lambda_2 = 0$. The two quantities are therefore correlated but not logically equivalent; $\lambda_2(L)$ additionally quantifies the *margin* of connectivity — how many link removals the network can absorb before becoming disconnected — and is therefore a more conservative and informative indicator for robustness analysis.

The connection to standard control metrics is direct. In the consensus model for federated learning propagation, $\eta[k+1] = (I - \epsilon L(\Phi))\,\eta[k]$ (where $\eta_i \in \mathbb{R}^p$ are local ML model parameters), the convergence rate toward a common network estimate is governed by $(1-\epsilon\lambda_2)^k$: a large Fiedler value (dense, well-connected network) accelerates distributed adaptation, while $\lambda_2 \to 0$ slows it proportionally, and $\lambda_2 = 0$ halts consensus altogether [27,34]. Equivalently, viewing federated-learning parameter propagation as a linear network control problem with update input $B_c\,\Delta\eta[k]$, the controllability Gramian of the deviation dynamics is *rank-deficient* whenever $\lambda_2(L) = 0$: inter-component state differences lie in an unreachable subspace, meaning no federated update applied within one connected component can propagate information to any other. In practical terms, a fragmented FOSS ecosystem does not merely slow distributed learning — it renders the mechanism structurally inoperative for the disconnected subsets of nodes. In the sequel, $\mathcal{C}(\Phi)$ is retained as the primary metric for its direct interpretability, with $\lambda_2(L(\Phi))$ as its rigorous spectral counterpart.

 doi:10.20944/preprints202603.1360.v1

3.3. ML as Disturbance Estimator and Distributed Adaptive Compensator

In a centralised system, a supervisor would minimise a global error E. In the decentralised regime described here, we introduce a local Machine Learning (ML) agent at each node to act as a **disturbance estimator**. Rather than directly issuing control commands, ML_i observes the measured output $y_i^{\mathrm{m}}[k]$, the local historical dataset $\mathcal{X}_{\mathrm{hist}}$, and — in the federated setting — the incoming model update $\Delta\eta_i$ admitted by the Admission Controller Ω_i, to produce an estimate $\hat{w}_i[k]$ of the local process disturbance $w_i[k]$. The nominal software controller K_i then subtracts this estimate from its output as a feedforward correction, achieving practical disturbance compensation. This two-step mechanism deliberately separates *estimation* (the role of ML_i) from *compensation* (the role of the control law), keeping the ML block a statistically well-defined estimator while the closed-loop structure handles the corrective action.

In the digital twin interpretation of the node architecture, the nominal state-space model (A_i, B_i, C_i) constitutes the physics-informed machine twin. Its disturbance-free prediction evolves as

$$\hat{x}_i^{\mathrm{DT}}[k+1] = A_i\hat{x}_i^{\mathrm{DT}}[k] + B_i u_i[k], \qquad \hat{y}_i^{\mathrm{DT}}[k] = C_i\hat{x}_i^{\mathrm{DT}}[k], \tag{6}$$

and the *twin gap* signal

$$\delta_i[k] = y_i^{\mathrm{m}}[k] - \hat{y}_i^{\mathrm{DT}}[k] \tag{7}$$

measures the discrepancy between the twin's nominal prediction and the actual measured output. ML_i is therefore the cognitive output of the digital twin: it maps $(\delta_i[k],\ \mathcal{X}_{\mathrm{hist}};\ \eta_i)$ to a disturbance estimate $\hat{w}_i[k]$, learning to explain what the nominal physics model cannot account for [14,15,17]. In the federated setting, the incoming update $\Delta\eta_i$ (admitted by the Admission Controller Ω_i) adjusts the estimator's parameter vector as $\eta_i \leftarrow \eta_i + \Delta\eta_i$; rather than transferring raw peer observations, it transfers distilled calibration knowledge, functioning as a parameter-space extension of $\mathcal{X}_{\mathrm{hist}}$ that reduces the local data volume required for the estimator to converge. This benefit is strongest within homogeneous process-class clusters ($\pi_j = \pi_i$, similar $\phi_j \approx \phi_i$) and degrades as structural dissimilarity $\|\Delta A_i - \Delta A_j\|$ increases; for a newly onboarded node of a different hardware family, the local dataset $\mathcal{X}_{\mathrm{hist}}$ remains the primary driver of convergence until sufficient local data disambiguates which peers are genuinely informative. The resulting control law is:

$$u_i[k] = K_i(e_i[k]) - \hat{w}_i[k], \tag{8}$$

where $e_i[k] = r_i - y_i^{\mathrm{m}}[k]$ is the tracking error, $K_i(\cdot)$ is the firmware process controller (the real-time control law executing at the node level), and $\hat{w}_i[k] = \mathrm{ML}_i(\delta_i[k],\ \mathcal{X}_{\mathrm{hist}};\ \eta_i)$ is the disturbance estimate produced by a local estimator (e.g., a Gaussian Process [45] or neural network [46]) trained on historical process data $\mathcal{X}_{\mathrm{hist}}$ to predict and cancel w_i; here $\hat{w}_i[k]$ is treated as an input-equivalent correction, which achieves exact cancellation under the standard matched-disturbance assumption ($w_i[k] \in \mathrm{range}(B_i)$) [41]. When this assumption holds only approximately — as it will for disturbances with significant state-coupling (e.g., thermal drift acting jointly on A_i and B_i) — the compensation achieves partial rather than exact rejection, and the residual enters as a bounded bias in J_i; reducing this gap is one motivation for the Gaussian Process and neural-network estimator choices discussed below. Note that K_i is architecturally distinct from the upstream CAD / CAM / slicer toolchain: the latter is the FOSS backend that produces the design reference $r_i = \mathcal{I}(D, \theta_i)$ at the governance layer; K_i receives r_i and acts on it in real time at the node. The objective of the i-th node is to minimise the local cost function:

$$J_i = \mathbb{E}\left[\sum_{k=0}^{T}\left(\|y_i[k] - y_{\mathrm{ref},i}\|_Q^2 + \|u_i[k]\|_R^2\right)\right], \tag{9}$$

where $y_{\mathrm{ref},i}$ is the node-specific output quality specification (e.g., a dimensional tolerance or surface finish target) derived from the FOSS design, $Q \succeq 0$ and $R \succ 0$ are symmetric weighting matrices penalising output deviation and control effort respectively [41,47], and the cost is evaluated on the true

 doi:10.20944/preprints202603.1360.v1

(noise-free) process output $y_i[k] = C_i x_i[k]$; the measured output $y_i^{\mathrm{m}}[k]$ enters only through the control law (8).

Note that $y_{\mathrm{ref},i}$ and r_i serve distinct roles. The process reference $r_i = \mathcal{I}(D, \theta_i)$ is the time-varying trajectory generated by the FOSS toolchain that K_i tracks through the error signal $e_i[k]$; it enters the system through the control law (8). The quality specification $y_{\mathrm{ref},i}$, by contrast, is an outcome criterion: the value the fabricated output $y_i[k]$ must ultimately satisfy. A controller may track r_i exactly and still produce $y_i \neq y_{\mathrm{ref},i}$ if the process model is imperfect — closing that gap is precisely the role of the ML compensation term $\hat{w}_i[k]$.

The **global distributed control objective** across the entire network is:

$$\min_{u_1,\ldots,u_N} \; J = \sum_{i=1}^{N} J_i, \tag{10}$$

subject to the FOSS interoperability constraint (3), where the minimisation is taken over the control input sequences $\{u_i[k]\}_{k=0}^{T}$ [43], or equivalently over the controller policies K_i and ML parameters that determine them. While the "Design Rules" (FOSS) provide the structure, the ML layer provides the "Cognitive" adjustment to ensure that even across heterogeneous nodes — from a desktop Voron to a high-end industrial CNC — the output y_i remains within tolerance.

Each node reports to the FOSS network not its raw signals, but two distinct privacy-preserving outputs produced by the **Disclosure Function** $\mathcal{F}$:

$$\big(z_i[k],\, \Delta\eta_i\big) = \mathcal{F}(y_i^{\mathrm{m}}[k],\, \eta_i), \tag{11}$$

where $z_i[k]$ is the scalar quality telemetry (privacy-filtered from $y_i^{\mathrm{m}}[k]$; e.g. dimensional accuracy, surface quality), and $\Delta\eta_i$ is the compressed federated-learning update computed by $\mathcal{F}$ from the local ML model parameters η_i [12] — the distilled parameter increment encoding what the local disturbance estimator has learned, shareable with peer nodes of the same process class to enable *a priori* uncertainty reduction. The two channels are architecturally separate: $\mathcal{F}$ acts as a privacy filter on measurement data and as a compression gateway for federated model updates. Raw sensor streams and raw model weights never leave the node; only $z_i[k]$ and $\Delta\eta_i$ cross the node boundary, giving the pair $(z_i[k], \Delta\eta_i)$ the role of the node's sole output channel to the FOSS governance layer.

3.4. Systemic Robustness and Structural Resilience

The **systemic robustness** of the decentralised manufacturing network is defined as the system's ability to maintain aggregate performance despite external perturbations. Formally, let $d[k]$ denote a disturbance vector acting on the system (governance, technical, or supply chain). The robustness metric is defined as:

$$\mathcal{R}(d) = \frac{\|Y_{nom} - Y_d\|}{\|d\|}, \qquad \|d\| \neq 0, \tag{12}$$

where Y_{nom} is the nominal aggregate output, Y_d is the output under disturbance d, and $\|\cdot\|$ denotes the ℓ_2 signal norm over the horizon $k = 0, \ldots, T$. A robust system exhibits small $\mathcal{R}$ for large classes of disturbances; the worst-case quantity $\sup_{\|d\|\neq 0} \mathcal{R}(d)$ is the closed-loop ℓ_2-gain from disturbance to output deviation [47,48].

The FOSS-governed decentralised model exhibits **structural resilience**: if the admissible control set $\mathcal{U}_j$ becomes empty for node j (failure or regulatory lock-down), the remaining $N-1$ nodes maintain functionality because there is no single supervisory controller [43]. This is in direct contrast to the centralised case, where restricting the central controller — in the absence of fallback redundancy — constitutes a single point of failure for the entire network.

 doi:10.20944/preprints202603.1360.v1

$$\begin{aligned} \text{Centralised:} \quad & \mathcal{U}_{central} = \emptyset \implies J \to \infty; \\ \text{Decentralised:} \quad & \mathcal{U}_j = \emptyset \implies J_{op} \triangleq \sum_{i \neq j} J_i < \infty. \end{aligned} \tag{13}$$

Here J_{op} is the aggregate cost restricted to the $N-1$ operational nodes; each summand J_i ($i \neq j$) remains finite provided the corresponding node satisfies $\|A_i\| < 1$ and the FOSS governance layer remains intact. Throughout the case studies, $\|A_i\| < 1$ (for an induced matrix norm) is the standing stability assumption; it is a sufficient condition for asymptotic stability and simultaneously ensures convergence of the geometric series bounds used in Section 4.2. The cost of the failed node diverges but does not propagate to its peers.

4. Framework Application: Disturbance Analysis

To demonstrate the generality and analytical utility of the proposed framework, we apply it to three distinct disturbance classes, each entering the system at a different layer. For each case, we identify the disturbance vector, trace its propagation through the system model defined in Section 3, and characterise the role of FOSS architecture in disturbance rejection. The cases are drawn from real-world phenomena in the digital fabrication ecosystem.

4.1. Case A: Governance Disturbance — Regulatory Input Restriction

4.1.1. Disturbance Characterisation

WA House Bill 2321 (HB2321, introduced January 2026) [18] and similar legislative efforts aimed at restricting the digital distribution of files for 3D-printable components represent a **governance disturbance** $d_{gov}[k]$ acting on the system. This class of disturbance is evolving rapidly: whereas HB2321 targets the informational layer (file distribution), more recent proposals extend regulation to the hardware and firmware layers — NY A10005 (2026–27 Executive Budget) [20] and CA AB2047 (introduced February 2026) [19] mandate certified firearm-blocking firmware in all consumer 3D printers and impose criminal penalties for circumventing it. This escalation means that $d_{gov}[k]$ now acts not only on $\mathcal{U}_{legal}$ via file restrictions but also directly on the firmware bypass mechanisms analysed in Case C (Section 4.3), creating a coupled multi-layer disturbance. From a systems engineering perspective, such regulation acts as a **boundary condition** or **external perturbation** to the decentralised manufacturing network [5].

To characterise the depth at which governance disturbances penetrate the system hierarchy, we introduce a **regulatory intensity index** $\rho \in \{1, 2, 3, 4\}$:

- $\rho = 1$: distribution controls only — file-sharing restrictions targeting the informational layer (e.g., HB2321 [18]);
- $\rho = 2$: mandatory blocking middleware — software or firmware that must be integrated at the toolchain layer (e.g., NY A10005 [20], CA AB2047 [19]);
- $\rho = 3$: anti-circumvention and firmware-modification prohibition — criminal or civil penalties for replacing or disabling certified firmware;
- $\rho = 4$: hardware attestation and closed secure-boot chain — hardware-enforced certification that prevents installation of non-compliant firmware at the silicon level.

As ρ increases, $d_{gov}[k]$ targets successively harder-to-substitute layers. Current legislative proposals span $\rho = 1{-}3$.

Mathematically, $d_{gov}[k]$ acts by contracting the set of admissible control inputs:

$$u_i[k] \in \mathcal{U}_{legal}(d_{gov}[k]) \subset \mathcal{U}_i^{total}, \tag{14}$$

where $\mathcal{U}_i^{total}$ denotes the full admissible input set in the absence of external restriction; governance and vendor actions carve out strict subsets — $\mathcal{U}_{legal} \subseteq \mathcal{U}_i^{FOSS} = \mathcal{U}_i^{total}$ and $\mathcal{U}_i^{vendor} \subset \mathcal{U}_i^{total}$ respectively. Here $\mathcal{U}_{legal}(d_{gov}[k])$ shrinks as $\|d_{gov}[k]\|$ increases. When a centralised authority targets a specific design

 doi:10.20944/preprints202603.1360.v1

$D_{restricted}$, the regulation attempts to drive $\mathcal{U}_{legal}(D_{restricted}, d_{gov}[k]) \to \emptyset$, eliminating all compliant input trajectories for that design from $\mathcal{U}_i^{total}$.

4.1.2. Propagation Through the System

The disturbance $d_{gov}[k]$ enters at the **FOSS governance layer** (Figure 1), contracting $\mathcal{U}_{legal}$ and propagating downward to restrict the admissible input set at every node. In a **centralised system** — meaning one in which a single platform or authority mediates all node–design interactions, controls distribution, and enforces updates (e.g., a proprietary cloud-based CAD/CAM workflow) — the supervisor acts as a single point of enforcement, reducing $\mathcal{U}_{legal}(D_{restricted})$ to the empty set for targeted designs. No compliant input trajectory exists, so the tracking error grows unbounded and the local cost diverges:

$$\mathcal{U}_{legal}(D_{restricted}) = \emptyset \implies J_i(D_{restricted}) \to \infty \quad \forall i. \tag{15}$$

The enforcement is total because a single platform mediates all node–design interactions; restricting that component simultaneously restricts every node.

4.1.3. FOSS-Enabled Disturbance Rejection

In the **decentralised FOSS-governed model**, governance is not a "gatekeeper" but a "protocol." The case of HB2321 demonstrates the **decoupling of the informational layer from the physical node**. The FOSS ecosystem provides disturbance rejection through three mechanisms:

1. **Redundancy:** Because FOSS toolchains (slicers, firmware, CAD) are distributed and peer-to-peer, the "Design Rules" Φ remain accessible even if specific distribution channels are throttled. The interoperability index satisfies $\mathcal{C}(\Phi_{d_{gov}}) > 0$ as long as any subset of open channels persists, where $\Phi_{d_{gov}}$ denotes the governance-disturbed ecosystem state.
2. **Observability Gap:** For a regulator, the state x_i of a private, decentralised node is hidden, making enforcement of $\mathcal{U}_{legal}$ computationally and logistically expensive — the distributed enforcement problem. Formally, by analogy with the standard observability construction, the regulatory reach matrix

$$\mathcal{O}_{reg}^{reach} = \begin{bmatrix} C_{reg} \\ C_{reg}A_c \\ \vdots \\ C_{reg}A_c^{N-1} \end{bmatrix} \tag{16}$$

   has $\text{rank}(\mathcal{O}_{reg}^{reach}) \ll N$ in the decentralised case [27], as no single measurement matrix C_{reg} can observe all node states simultaneously. Here A_c is the network coupling matrix governing information flow over the topology (not the physical plant dynamics matrix), so $\mathcal{O}_{reg}^{reach}$ quantifies the nodes reachable within $N-1$ hops of the regulator's direct observation points — a graph-theoretic reachability measure, not LTI-system observability. This mismatch can be reframed as a "big observability, small controllability" condition: visibility can improve through interface-level reporting, but actionable intervention remains local unless component contracts expose explicit assume–guarantee boundaries for diagnosis and coordination [3]. It should be noted that this argument captures *information asymmetry* and per-node enforcement overhead — not formal legal enforceability; a regulator retains extra-network instruments (physical inspection, subpoena, forensic analysis) that operate outside the information-flow topology entirely. The practical force of the observability gap is therefore a cost-of-enforcement argument rather than a formal impossibility result.
3. **ML-Enabled Functional Substitution:** If HB2321 constrains the distribution of certain "functional" files, an ML-capable node could in principle derive a functionally equivalent variant D' of a restricted design D — one that achieves a physically similar state x_i while satisfying $u_i[k] \in \mathcal{U}_{legal}$. This **functional equivalence transformation** represents an analytical possibility under

 doi:10.20944/preprints202603.1360.v1

the assumption that such a variant exists and is learnable from local process data; whether it is achievable in practice depends on the specificity of the regulatory restriction and the expressive capacity of the local model.

4.1.4. Robustness Assessment

Under FOSS architecture, the interoperability index degrades gracefully with respect to governance pressure:

$$\frac{\partial \mathcal{C}}{\partial \|d_{gov}\|} \approx 0 \quad \text{for} \quad \|d_{gov}\| < d^*_{gov}(\rho), \tag{17}$$

where $d^*_{gov}(\rho)$ is the critical disturbance magnitude below which the FOSS ecosystem's redundancy absorbs the perturbation without reducing network connectivity, and is a **decreasing function of** ρ: $d^*_{gov}(1) \geq d^*_{gov}(2) \geq d^*_{gov}(3) \geq d^*_{gov}(4)$.

Derivation of the dead-zone (illustrative model). To see why $\partial \mathcal{C}/\partial \|d_{gov}\| \approx 0$ holds for small $\|d_{gov}\|$, consider a symmetric N-node network in which each directed link (i, j) is maintained by $R(\rho) \geq 2$ independent FOSS distribution channels (e.g., distinct hosting platforms, peer-to-peer mirrors, or protocol implementations). The governance disturbance blocks each channel independently with probability $p(\|d_{gov}\|) \in [0, 1]$, where $p(0) = 0$. A link (i, j) fails only when all R channels are simultaneously blocked, which occurs with probability p^R. Taking the expectation over independent link failures:

$$\mathbb{E}[\mathcal{C}(\Phi_{d_{gov}})] = 1 - p(\|d_{gov}\|)^{R(\rho)}, \qquad \frac{\partial \mathbb{E}[\mathcal{C}(\Phi_{d_{gov}})]}{\partial \|d_{gov}\|} = -R(\rho)\, p^{R(\rho)-1}\, p'(\|d_{gov}\|). \tag{18}$$

Since $p(0) = 0$, the factor p^{R-1} vanishes at $\|d_{gov}\| = 0$ whenever $R \geq 2$, confirming the dead-zone: the first-order effect of small governance pressure on network connectivity is zero. The threshold $d^*_{gov}(\rho)$ is well-approximated by the value where this gradient first becomes significant, which decreases with $R(\rho)$. Because higher regulatory intensity ρ eliminates distribution channels (reducing $R(\rho)$), $d^*_{gov}(\rho)$ is indeed a decreasing function of ρ. At $\rho = 3{-}4$, hardware-certification mandates constrain every node at the point of manufacture, driving $R(\rho) \to 1$ and $d^*_{gov}(\rho) \to 0$. Quantifying $R(\rho)$ and $p(\|d_{gov}\|)$ empirically for specific regulatory regimes is identified as a direction for future work [8,44].

4.2. Case B: Technical Disturbance — Process Variability Across Heterogeneous Nodes

4.2.1. Disturbance Characterisation

The second disturbance class addresses a fundamental challenge of decentralised manufacturing: identical digital designs produce non-identical physical outputs across different fabrication nodes. This arises because the node heterogeneity parameters ϕ_i and the stochastic process noise $w_i[k]$ vary across the network.

Consider a network of N geographically distributed fabrication nodes of mixed process types, collectively producing a parametric mechanical assembly (e.g., a structural bracket whose flat base plate is laser-cut from sheet metal, whose mounting bores are finish-machined on a CNC lathe, and whose complex mounting geometry is FDM-printed). The technical disturbance $d_{tech}[k]$ manifests as inter-node variance in the system matrices:

$$A_i(\phi_i) = A_{nom}(\pi_i) + \Delta A_i, \qquad B_i(\phi_i) = B_{nom}(\pi_i) + \Delta B_i, \tag{19}$$

where $A_{nom}(\pi_i)$, $B_{nom}(\pi_i)$ represent the nominal dynamics for process class π_i, and ΔA_i, ΔB_i are node-specific additive perturbations [48] arising from process-class-dependent sources: extruder geometry and stepper resolution for FDM nodes [9,10]; spindle runout and fixturing compliance for CNC nodes [35,36]; beam focus and assist-gas pressure for laser nodes [38,39]; and common factors such as ambient temperature, material batch properties, and mechanical wear — including in metal-forming processes [4].

 doi:10.20944/preprints202603.1360.v1

4.2.2. Propagation Through the System

The disturbance enters at the **node level** (Equation 1), directly affecting the state evolution. The deviation has two components: accumulated distortion of past control inputs caused by actuation mismatch ΔB_i, and drift in the state trajectory caused by dynamic mismatch ΔA_i. Without compensation, the output deviation at each node is decomposed into two distinct contributions — input-driven accumulation from actuation mismatch ΔB_i and initial-condition drift from dynamic mismatch ΔA_i — using the discrete variation-of-parameters formula [40]:

$$\Delta y_i[k] = C_i \sum_{\tau=0}^{k-1} A_i^{k-1-\tau}\, \Delta B_i\, u_i[\tau] \;+\; C_i\, (\Delta A_i)^k\, x_i[0]. \tag{20}$$

For a stable node ($\|A_i\| < 1$), the geometric series in the first term is bounded and the second term decays geometrically [41]. Applying submultiplicativity of the matrix norm and bounding the input by $\bar{u} = \max_\tau \|u_i[\tau]\|$ yields:

$$\|\Delta y_i[k]\| \;\le\; \underbrace{\frac{\|C_i\|\,\|\Delta B_i\|\,\bar{u}}{1-\|A_i\|}}_{\text{input-driven accumulation}} \;+\; \underbrace{\|C_i\|\,\|\Delta A_i\|^k\,\|x_i[0]\|}_{\text{initial-condition decay}}. \tag{21}$$

The bound (21) reveals the scaling behaviour more clearly than the exact expression. The first term represents a **steady-state plateau**: the input-driven deviation accumulates over time but saturates to a finite value determined by the ratio $\|\Delta B_i\|/(1-\|A_i\|)$. The second term represents **transient drift** from the initial condition, which decays geometrically since the node is stable. In a centralised factory, $\|\Delta A_i\| \approx 0$ and $\|\Delta B_i\| \approx 0$ because machines are calibrated to tight tolerances under controlled conditions. In the decentralised network, both perturbations can be substantial, and without a corrective mechanism, the aggregate output $Y = [y_1, \ldots, y_N]^T$ exhibits high variance across nodes.

Consider the illustrative example of two FDM nodes assigned the same print. The numerical values below are **order-of-magnitude estimates chosen for illustration**, not fitted to specific hardware; they are consistent with the inter-node variability ranges reported in the FDM process literature [9,10]: home node h is a desktop printer with worn extruder gears and ambient thermal variation ($\|A_h\| \approx 0.3$, $\|\Delta B_h\| \approx 0.15$); professional node p is an enclosed unit operating within a climate-controlled enclosure ($\|A_p\| \approx 0.02$, $\|\Delta B_p\| \approx 0.01$). The values quoted are perturbation norms in a state-space normalised around the operating equilibrium, under the assumption $\|A_{nom}(\pi_i)\| \ll \|\Delta A_i\|$, so that the total norm is dominated by the perturbation term. For comparable output and input norms ($\|C_h\| \approx \|C_p\|$, $\bar{u}_h \approx \bar{u}_p$), the steady-state bound ratio is:

$$\frac{\|\Delta y_h\|_{\max}}{\|\Delta y_p\|_{\max}} \;\approx\; \frac{\|\Delta B_h\|/(1-\|A_h\|)}{\|\Delta B_p\|/(1-\|A_p\|)} \;=\; \frac{0.15/0.7}{0.01/0.98} \;\approx\; 21. \tag{22}$$

The home node h's worst-case output deviation is approximately **21 times larger** than the professional node p's. The dominant contributor is the actuation mismatch $\|\Delta B_i\|$ (worn mechanical components distorting the input mapping), amplified by the factor $1/(1-\|A_i\|)$ which penalises nodes whose dynamics are closer to the stability margin. After 100 deposited layers, this translates to several hundred microns of dimensional error at home node h while professional node p remains within tolerance—precisely the heterogeneity that the ML compensation layer in (8) is designed to close.

4.2.3. FOSS-Enabled Disturbance Rejection

The FOSS ecosystem provides a **distributed feedback mechanism** that functions as a decentralised calibration infrastructure:

1. **Community-Shared Calibration Profiles:** Open-source slicer software (e.g., PrusaSlicer, Cura, SuperSlicer) maintains community-contributed machine profiles that encode empirically vali-

 doi:10.20944/preprints202603.1360.v1

dated parameter sets for specific hardware configurations. These profiles act as a **feedforward compensation**:

$$u_i^{ff}[k] = \mathcal{P}(\phi_i), \tag{23}$$

where $\mathcal{P}$ is the profile lookup function that maps node heterogeneity ϕ_i to pre-compensated input parameters. This reduces $\|\Delta B_i\, u_i\|$ without requiring explicit system identification at each node.

2. **Open Process Data Sharing:** Under FOSS governance, nodes can share process telemetry (temperatures, flow rates, dimensional measurements) through open APIs and standardised data formats. This effectively increases the **network observability**:

$$\operatorname{rank}(\mathcal{O}_{network}) = \operatorname{rank}\begin{bmatrix} C_1 \\ C_2 A_c \\ \vdots \\ C_N A_c^{N-1} \end{bmatrix} > \operatorname{rank}(\mathcal{O}_{single}), \tag{24}$$

where $\mathcal{O}_{single} = C_i$ is the observability of an isolated node and A_c is the network coupling matrix (information propagates one hop per step), enabling distributed state estimation across the network [27].

3. **ML-Augmented Process Compensation:** The local ML disturbance estimator (Equation 8) learns the mapping $\mathrm{ML}_i : (\delta_i[k], \mathcal{X}_{\text{hist}};\, \eta_i) \mapsto \hat{w}_i[k]$, predicting the process disturbance from the twin gap (7) so the control law can cancel it. Crucially, under FOSS governance, training data can be aggregated across nodes using **federated learning** [12], enabling each node to benefit from the collective experience of the network without centralising sensitive process data.

4.2.4. Robustness Assessment

With the FOSS-enabled compensation mechanisms active, the effective disturbance seen by each node is reduced:

$$w_i^{eff}[k] = w_i[k] - \mathrm{ML}_i(\delta_i[k], \mathcal{X}_{\text{hist}};\, \eta_i), \tag{25}$$

and the aggregate cost function satisfies $J_{\text{FOSS+ML}} < J_{\text{isolated}} < J_{\text{cent-fail}}$, where J_{isolated} corresponds to nodes operating without FOSS-mediated data sharing and $J_{\text{cent-fail}}$ represents the centralised case under single-point-of-failure conditions.

4.3. Case C: Supply Chain Disturbance — Vendor Lock-In and Platform Discontinuation

4.3.1. Disturbance Characterisation

The third disturbance class addresses supply chain disruptions arising from proprietary vendor actions: firmware lock-down, component discontinuation, or deliberate incompatibility introduced to capture market share. In the framework, a supply chain disturbance $d_{sc}[k]$ acts by **removing nodes from the network** or **restricting their control input space**.

Consider a scenario in which a proprietary 3D printer manufacturer issues a firmware update that restricts the use of third-party filaments by implementing a material authentication protocol. For all nodes i using this hardware, the admissible input set is suddenly constrained:

$$u_i[k] \in \mathcal{U}_i^{vendor}(d_{sc}[k]) \subset \mathcal{U}_i^{total}, \qquad \text{for} \quad i \in \mathcal{S}_{affected}, \tag{26}$$

where $\mathcal{S}_{affected} \subset \{1, \ldots, N\}$ is the set of nodes running the proprietary platform and $\mathcal{U}_i^{vendor}(d_{sc}[k])$ shrinks as $\|d_{sc}[k]\|$ increases, restricting inputs to vendor-approved materials and parameters only [1].

More severely, if a vendor discontinues a hardware platform entirely, the affected nodes are removed from the network:

$$N_{eff}[k] = N - |\mathcal{S}_{discontinued}[k]|, \tag{27}$$

 doi:10.20944/preprints202603.1360.v1

reducing the effective network size and, consequently, the aggregate production capacity and resilience.

4.3.2. Propagation Through the System

The supply chain disturbance propagates through **two pathways simultaneously**:

1. **Direct node impact:** Affected nodes experience a constrained or eliminated control input, degrading their local cost J_i or setting it to infinity (node failure).
2. **Network topology degradation:** The coupling matrix A_c is modified as affected nodes lose the ability to participate in open data exchange:

$$[A_c(\Phi, d_{sc})]_{ij} = 0 \quad \text{for all} \quad i \in \mathcal{S}_{affected} \text{ or } j \in \mathcal{S}_{affected}, \tag{28}$$

reducing the interoperability index $\mathcal{C}(\Phi_{d_{sc}}) < \mathcal{C}(\Phi)$, where $\Phi_{d_{sc}}$ denotes the supply-chain-degraded ecosystem state.

4.3.3. FOSS-Enabled Disturbance Rejection

The FOSS ecosystem provides resilience against supply chain disturbances through **component substitutability** and **informational persistence**:

1. **Open-Source Firmware as a Bypass Mechanism:** Community-developed firmware projects (e.g., Marlin, Klipper, RepRapFirmware) enable users to replace proprietary firmware on affected hardware nodes. This effectively restores the full control input space:

$$\mathcal{U}_i^{FOSS} = \mathcal{U}_i^{total} \supset \mathcal{U}_i^{vendor}, \tag{29}$$

re-integrating the node into the open network. The disturbance is fully rejected at the firmware level. However, recent governance proposals (NY A10005 [20], CA AB2047 [19]) that mandate certified firmware and criminalise its modification would directly constrain this bypass mechanism, effectively coupling supply-chain and governance disturbances. In such a regime, $\mathcal{U}_i^{FOSS}$ becomes legally inaccessible without regulatory compliance, reducing the effective rejection capacity. This interaction is discussed further in Case A (Section 4.1) and identified as a future modelling direction.
2. **Open Hardware Design and Node Substitution:** When a proprietary platform is discontinued, open-source hardware designs (e.g., Voron, Prusa, RatRig) enable the construction of functionally equivalent replacement nodes. Because the FOSS interface constraint $\mathcal{I}$ (Equation 3) is **hardware-agnostic**, a new open-source node j can substitute for a discontinued proprietary node i:

$$\mathcal{I}(D, \theta_j) \approx \mathcal{I}(D, \theta_i) \quad \Longrightarrow \quad y_j[k] \approx y_i[k], \tag{30}$$

provided the ML compensation layer has adapted to the new node parameters ϕ_j — an adaptation that requires local process data and therefore implies a finite ramp-up period rather than instantaneous substitution.
3. **Informational Persistence of Design Rules:** Unlike proprietary ecosystems where design files, slicer profiles, and toolchain knowledge are lost upon vendor exit, FOSS ensures **informational persistence**:

$$\left| \frac{\partial \Phi}{\partial \| d_{sc} \|} \right| \approx 0, \tag{31}$$

the state of the open-source ecosystem is approximately invariant to individual vendor actions — under conditions of healthy community redundancy (multiple repository mirrors, active maintainer base, distributed version control) — because the design rules are not owned, hosted, or controlled by any single commercial entity [8,23].

 doi:10.20944/preprints202603.1360.v1

4.3.4. Robustness Assessment

The supply chain disturbance is absorbed at two levels. At the **node level**, open-source firmware restores $\mathcal{U}_i^{total}$ and open hardware enables node substitution, restoring $N_{eff} \to N$ after a local-learning ramp-up period during which J_j may temporarily exceed J_i while ML_j accumulates sufficient process data to match the compensation accuracy of the replaced node. At the **network level**, the FOSS ecosystem ensures Φ is approximately invariant to vendor discontinuation (under community-redundancy conditions), preserving the coupling matrix A_c and the interoperability index $\mathcal{C}$.

The key structural difference from the centralised case is summarised as:

$$\begin{aligned} \text{Proprietary:} \quad & \mathcal{U}_i \xrightarrow{d_{sc}} \mathcal{U}_i^{vendor},\ \Phi \xrightarrow{d_{sc}} \Phi^{vendor}; \\ \text{FOSS:} \quad & \mathcal{U}_i \xrightarrow{d_{sc}} \mathcal{U}_i^{total},\ \Phi \xrightarrow{d_{sc}} \Phi. \end{aligned} \tag{32}$$

where $\xrightarrow{d_{sc}}$ denotes the transition of each quantity under the supply chain disturbance d_{sc}; the approximate FOSS invariance $\Phi \xrightarrow{d_{sc}} \Phi$ — holding under conditions of community redundancy — is the formal expression of informational persistence [8,23].

4.4. Cross-Case Comparison

Table 2 summarises how each disturbance class propagates through the system and the corresponding FOSS rejection mechanism, providing a unified view of the framework's analytical power.

Table 2. Disturbance propagation and FOSS rejection mechanisms across the three case studies.

	Case A: Governance	**Case B: Technical**	**Case C: Supply Chain**
Disturbance	Restriction on $\mathcal{U}$ via regulation	Variance in A_i, B_i across nodes	Node removal or $\mathcal{U}$ restriction by vendor
Entry point	FOSS governance layer (Φ)	Node dynamics (x_i)	Hardware/firmware layer
Propagation	$\mathcal{C}(\Phi) \to 0$ (fragmentation)	$\mathrm{Var}(Y) \to \infty$ (quality divergence)	$N_{eff} \to 0$, $\mathcal{C} \to 0$
FOSS rejection	Redundancy, observability gap	Calibration profiles, federated ML	Open firmware, hardware substitution
Key metric	$\partial\mathcal{C}/\partial\|d_{gov}\| \approx 0$ (low ρ)	$J_{FOSS+ML} < J_{isolated}$	$\|\partial\Phi/\partial\|d_{sc}\|\| \approx 0$

To ground these mechanisms in practice: in **Case A**, when a regional regulation restricts a proprietary slicer's feature set, a FOSS-governed network can fork or substitute the toolchain at short notice — the regulatory disturbance never reaches the production output. In **Case B**, a newly onboarded printer with a misaligned heated bed would, in isolation, produce systematic dimensional errors; within the framework, a federated calibration profile detects and compensates the offset locally, keeping network-wide variance within tolerance. In **Case C**, if a filament vendor discontinues a specialty material or a firmware vendor bricks a controller via a forced update, open-source alternatives (open hardware hotends, community firmware) restore full capability without dependence on the original supplier. In each case, the FOSS layer acts as a structural buffer that decouples internal network dynamics from external disturbances.

Table 3 provides a broader structural comparison between centralised and FOSS-governed decentralised architectures.

 doi:10.20944/preprints202603.1360.v1

Table 3. Structural comparison of centralised and decentralised (FOSS + ML) governance architectures.

Dimension	Centralised Governance	Decentralised (FOSS + ML)
Control Logic	Top-down / Hierarchical	Bottom-up / Emergent
Failure Mode	Fragile (Single point of capture)	Resilient (Redundant nodes)
Enforcement	Direct (API blocking)	Indirect (Difficult due to local autonomy)
Adaptability	Rigid	High (ML-driven local compensation)
Interoperability	Vendor-dependent	Standards-based (FOSS Design Rules)
Observability	Global (centralised data)	Local (distributed, privacy-preserving)
Supply Chain Lock-in	High (proprietary dependencies)	Low (open hardware, firmware, software)

5. Discussion

5.1. Synthesis Across Disturbance Classes

The three case studies presented in Section 4 demonstrate that the proposed framework provides a unified analytical lens for disturbances that are typically studied in isolation across different disciplinary communities. Governance restrictions (Case A) are usually analysed through policy and legal scholarship; process variability (Case B) through manufacturing engineering and quality control; and vendor lock-in (Case C) through supply chain management and industrial economics. By formalising all three as perturbations to a common state-space model, the framework reveals a shared structural mechanism: FOSS-governed decentralisation provides robustness not through optimality but through **redundancy, substitutability, and informational persistence**.

A key finding is that the disturbance rejection mechanisms are **complementary rather than independent**. The FOSS governance layer (Φ) simultaneously enables the calibration sharing that compensates for technical variability (Case B) and the firmware alternatives that counter vendor lock-in (Case C). This suggests that investments in open-source infrastructure yield compounding resilience benefits across disturbance classes — a systemic property that is invisible to single-domain analyses.

The three disturbance classes differ in the layer at which they enter the system. Governance disturbances act at the FOSS governance layer, contracting $\mathcal{U}_{legal}$ and modifying the structure of Φ; technical disturbances act at the node-dynamics layer, perturbing A_i and B_i; and supply-chain disturbances act at both layers simultaneously, removing nodes from the coupling matrix $A_c(\Phi)$ and restricting individual input sets. Despite this structural diversity, the framework reveals a unified rejection mechanism: the same FOSS governance layer that provides redundancy against governance restrictions ($\mathcal{C}(\Phi) > 0$ for $\|d_{gov}\| < d^*_{gov}$, Equation (17)) also enables the calibration sharing that compensates for technical variability, and the open-standard interfaces that enable node substitution under supply-chain disruption. This multi-level role of the single architectural layer Φ is the framework's central structural finding.

The mathematical coherence of this picture rests on the interoperability index $\mathcal{C}(\Phi)$ and the aggregate cost $J = \sum_i J_i$ serving as common metrics across all three cases. Complementarity of the rejection mechanisms implies that a network well-governed under FOSS is simultaneously resilient to governance interventions (Case A), calibrated across heterogeneous nodes via federated learning (Case B), and substitutable under vendor exit (Case C). This emergent joint robustness — not achievable by optimising a single disturbance class in isolation — provides strong motivation for treating open-source governance as a **structural layer** in manufacturing CPS design rather than a secondary implementation concern. It also implies that the three cases are not independent: any erosion of Φ that weakens the Case A redundancy argument simultaneously reduces the Case B calibration-sharing capacity and the Case C substitutability, predicting a correlated degradation across all three robust-

 doi:10.20944/preprints202603.1360.v1

ness metrics that a single-domain analysis would fail to capture. Formally characterising this joint robustness — deriving a single metric that accounts for the coupled disturbance vector $(d_{gov}, d_{tech}, d_{sc})$ simultaneously rather than per-case — is identified as the primary open analytical direction arising from this work.

The cases can also **couple causally**: a governance disturbance at high regulatory intensity ($\rho \geq 2$, Section 4.1) may induce a secondary technical disturbance by mandating blocking middleware that alters slicer or firmware behaviour, effectively modifying the nominal input map B_i and increasing $\|\Delta B_i\|$ for affected nodes (Case B pathway). The same regulatory pressure may simultaneously restrict the supply landscape to compliance-certified vendors, concentrating $\mathcal{S}_{affected}$ (Equation (26)) around non-compliant hardware and triggering a supply-chain disturbance (Case C pathway). Conversely, a supply-chain disruption that eliminates open-source firmware from the market amplifies governance pressure by removing the bypass channel that sustains $\mathcal{U}_i^{FOSS} = \mathcal{U}_i^{total}$ (Equation (29)). These causal couplings imply that the effective system robustness under simultaneous multi-class disturbances is lower than any single-case analysis would predict, and that the ρ-dependent threshold $d_{gov}^*(\rho)$ (Equation (17)) should be treated as a joint quantity over the full disturbance vector rather than a per-case constant.

5.2. Implications for Manufacturing Systems Design

The framework suggests that decentralised manufacturing systems should be designed as *architectures for graceful degradation* rather than architectures optimised for the nominal case. In centralised settings, performance is often maximised by tight integration and global observability; in heterogeneous open networks, resilience depends more strongly on modular interfaces, substitutable components, and the ability of nodes to continue operating under partial information. From this perspective, FOSS design rules are not merely an implementation preference but a structural requirement for maintaining interoperability when nodes differ in process class, hardware maturity, and governance context. Formally, any erosion of open interface standards degrades the coupling matrix $A_c(\Phi)$ and drives the interoperability index $\mathcal{C}(\Phi) \to 0$ (Equation (4)), potentially fragmenting the feasible task-allocation space even when local controllers remain functional.

For system architects and manufacturing engineers, a practical implication is to treat interoperability as a first-class design variable that can be specified, measured, and improved. Design choices that preserve open interfaces across CAD/CAM, slicer, firmware, and telemetry layers reduce coupling fragility and expand the feasible response set under disturbance. This shifts evaluation from local process capability alone toward network-level properties:

- *substitutability* — the degree to which one node can be replaced or bypassed without loss of task-allocation feasibility;
- *recovery time* — the latency before task allocation resumes after node loss or governance disruption;
- *continuity* — the ability to maintain throughput under software or supply-chain disturbance.

A related design dimension is the **degree of node autonomy** in accepting incoming FOSS updates. The architecture permits a continuous spectrum: at one extreme, a fully autonomous AI agent embedded in the Cognitive Gateway monitors FOSS repositories for firmware patches, calibration updates, and revised slicer profiles and applies them automatically — the manufacturing analogue of a background package manager. At the other extreme, a manual operator reviews and approves each incoming change before it is applied to the local node. Between these poles, semi-autonomous configurations allow operators to whitelist categories of updates (e.g., safety-critical firmware patches applied automatically, while changes to process parameters require explicit approval). As a worked example, Klipper-based nodes — such as a Voron or Prusa running Klipper on a Raspberry Pi host — illustrate this spectrum directly: the firmware controller K_i and the Cognitive Gateway are collocated on the same host computer, sharing both the compute substrate and the configuration API; the operator who edits `printer.cfg` and restarts the Klipper service is enacting the manual pole of Ω_i, while an AI agent with read–write access to the same configuration files and the ability to trigger a firmware

 doi:10.20944/preprints202603.1360.v1

reload represents the semi-autonomous pole. This operator-facing policy layer — formally a node-local acceptance function Ω_i governing which signals from the FOSS governance layer cross the node boundary — is symmetric with the Disclosure Function $\mathcal{F}$ already in the architecture: $\mathcal{F}$ governs what leaves the node, and Ω_i governs what enters it. Designing Ω_i as a configurable policy rather than a fixed behaviour preserves node sovereignty while still enabling the network-level benefits of distributed update propagation.

For passive users, the architecture is transparent by design: calibration improvements accumulate and distribute automatically, and output quality improves without visible intervention. Under normal operating conditions, the functional experience is the same whether a node runs proprietary or open-source firmware. The difference surfaces under disturbance and, for experienced operators, in the ability to inspect, modify, and contribute to the shared calibration infrastructure — a capacity the architecture preserves but does not require.

For policy and governance design, the analysis indicates that governance mechanisms that improve transparency, auditability, and shared quality criteria at interface boundaries can increase safety and accountability without requiring full centralisation of control. Contract-based interface specifications (assume–guarantee style) can further operationalise this by making fault attribution and cross-node diagnosis compositional, even when intervention authority remains distributed [3]. Conversely, interventions targeting single platforms or distribution channels may have limited systemic effect in open, multi-node ecosystems unless they also alter interface-level coordination: formally, such interventions selectively remove entries from $A_c(\Phi)$ without reducing the density of open-standard connections, leaving $\mathcal{C}(\Phi)$ largely intact.

A complementary pathway for policymakers is **policy-compatible openness**: governance regimes at low-to-moderate regulatory intensity ($\rho = 1-2$) may be accommodated without compromising FOSS structural properties if open implementations incorporate auditable, certified detection modules as optional, interface-compliant firmware components. Such a design preserves substitutability and informational persistence ($\|\partial\Phi/\partial\|d_{sc}\|\| \approx 0$, Equation (31)) while satisfying the regulatory constraint at the toolchain layer. Cryptographic attestation schemes that remain compatible with open interchange formats (G-code, STL, 3MF) would leave the coupling matrix $A_c(\Phi)$ intact. The framework provides a formal lens for evaluating this design space: any policy intervention that preserves $\mathcal{C}(\Phi) > \mathcal{C}_{min}$, where $\mathcal{C}_{min}$ is the connectivity threshold above the network percolation point [44], leaves the task-allocation and federated-learning capacities of the FOSS layer functionally intact. This suggests a negotiation space between regulators and open-source communities that is quantifiable in terms of the interoperability index rather than purely qualitative.

For the ML/AI community, the implication is that distributed learning in manufacturing should be framed as a robustness mechanism as much as an optimisation method. Federated and privacy-preserving learning pipelines — exemplified here by the $z_i[k]$ reporting channel (11), which forwards quality summaries and compressed model updates while retaining raw process data at the node — allow local controllers to reduce uncertainty while preserving node autonomy. This is essential in decentralised deployments where data centralisation is infeasible or undesirable. In this sense, the proposed CPS–FOSS–ML architecture provides a transferable template for other distributed industrial systems — such as energy microgrids, logistics networks, and distributed sensing infrastructures — in which technical heterogeneity and governance fragility are jointly constrained [49–51].

5.3. Limitations

Several limitations of the present work should be acknowledged. First, the framework is **qualitative-analytical**: while the mathematical formulation provides precise definitions and structural relationships, the case studies do not include quantitative simulation or empirical validation. The robustness metrics ($\mathcal{R}$, $\mathcal{C}$, $\partial\mathcal{C}/\partial d$) are defined formally but not computed numerically for specific network configurations. Second, the ML compensation layer is treated as a **conceptual component** of the framework; no specific learning algorithms are implemented or benchmarked. Third, the analysis assumes that FOSS ecosystems are inherently resilient, but real-world open-source projects

 doi:10.20944/preprints202603.1360.v1

face sustainability challenges (maintainer burnout, funding gaps, governance disputes) that could reduce Φ endogenously rather than through external disturbance.

Fourth, the framework assumes that the nominal digital twin model (A_i, B_i, C_i) constitutes a valid approximation throughout the node's operational life. Progressive wear, calibration drift, and hardware replacement cause the twin gap $\delta_i[k]$ to grow; if the residual exceeds the ML estimator's compensation capacity or becomes structurally biased, the disturbance rejection mechanism degrades. The federated learning pipeline partially mitigates this by propagating systematic corrections across nodes of the same machine class, and the $z_i[k]$ reporting channel provides the monitoring infrastructure to detect anomalous drift [51], but periodic DT re-identification remains a prerequisite for long-lived deployments.

Fifth, the framework treats the process class π_i of each node as fixed — which fabrication modality (additive, subtractive, laser, forming) handles a given task is assumed to be determined prior to network operation. Dynamic task-to-modality allocation, optimising assignment across heterogeneous process classes in response to disturbances, cost, or availability, is outside the current scope and remains an open direction.

5.4. Future Work

The directions below arise directly from the paper's analytical scope. Establishing the formal foundations — node-level dynamics, FOSS coupling, ML compensation architecture, and robustness metrics — while leaving simulation, implementation, and empirical testing for subsequent work is a deliberate choice: combining rigorous framework development with full empirical validation in a single paper would require sacrificing depth in one or the other. The directions are grouped accordingly: the first two concern testing and grounding the current framework; the remaining three extend it into new territory that the framework makes tractable but does not itself address.

Validation

1. **Simulation and Empirical Validation:** Agent-based or Monte Carlo simulation of the N-node network under parameterised disturbance scenarios would enable numerical computation of the robustness metrics ($\mathcal{R}$, $\mathcal{C}$, $\partial\mathcal{C}/\partial d$) and validate the qualitative predictions of the case studies. A controlled empirical experiment — comparing output quality distributions ($\mathrm{Var}(Y)$) across a network of heterogeneous FDM printers with and without FOSS-mediated calibration sharing — would provide direct physical evidence for the disturbance rejection mechanism formalised in Case B.
2. **ML and Federated Learning Implementation:** Deploying federated learning across a testbed of open-source fabrication nodes would test whether the ML disturbance estimator ML_i and its compensation scheme achieve the predicted cost reduction $J_{FOSS+ML} < J_{isolated}$ in practice, and reveal the conditions under which the Admission Controller Ω_i policy spectrum most affects network-level calibration quality.

Extensions

3. **Adaptive DT Re-identification:** The architecture naturally supports a three-timescale adaptation structure: fast closed-loop control, medium-timescale ML learning of η_i, and slow re-identification of the nominal model (A_i, B_i, C_i) itself. Systematic growth in the twin gap $\delta_i[k]$, observable via the $z_i[k]$ quality reports at the FOSS governance layer, could trigger automatic re-identification without centralised supervision — closing the long-timescale adaptation loop and extending the framework's validity to long-lived heterogeneous deployments [51].
4. **Network Dynamics:** Extending the framework to model time-varying network topologies — nodes joining and leaving, forks in FOSS projects, ecosystem mergers — would capture the evolutionary dynamics of real open-source manufacturing communities and test the stability of the interoperability index $\mathcal{C}(\Phi)$ under structural change.

 doi:10.20944/preprints202603.1360.v1

5. **Governance Intensity and Security:** Recent legislative proposals (NY A10005 [20], CA AB2047 [19]) target not only file distribution but also hardware certification and firmware modification, directly engaging the open-firmware bypass mechanism formalised in Case C. Extending the framework to model coupled governance disturbances that simultaneously constrain $\mathcal{U}_{legal}$ at the software layer and mandate hardware-level enforcement would assess the revised critical threshold d^*_{gov} and inform policy design. As governance pressure escalates toward hardware attestation, the privacy-preserving reporting channel $z_i[k]$ (11) and the federated update pipeline $\Delta\eta_i$ also acquire an information-security dimension, requiring formal analysis of the disclosure function $\mathcal{F}$ to quantify the tradeoff between network coordination utility and node privacy under adversarial conditions.

6. Conclusions

This paper presented a systems-theoretic framework for analysing decentralised manufacturing networks as distributed cyber-physical systems governed by open-source design rules. Each fabrication node is modelled as a stochastic state-space system coupled through a free and open-source software (FOSS) governance layer and equipped with a node-local Cognitive Gateway — hosting a digital twin nominal model, an ML disturbance estimator, and an Admission Controller that governs inbound FOSS updates across a configurable spectrum from manual operator review to autonomous AI-driven operation. Uncertainty reduction operates at two complementary levels: the FOSS governance layer informs process selection, collapsing the bulk of the uncertainty envelope before local compensation is needed, while the Cognitive Gateway compensates residual process variability through local learning and receives calibration improvements propagated via federated learning from structurally similar nodes. Three case studies — governance restriction (exemplified by WA HB2321 [18] and the emerging regulatory wave including NY A10005 [20] and CA AB2047 [19]), inter-node process variability, and vendor lock-in — demonstrated the framework's ability to provide a unified analytical treatment of disturbances that cross disciplinary boundaries.

The central finding is that FOSS-governed decentralisation provides structural robustness through mechanisms fundamentally different from those available in centralised architectures: redundancy of distribution channels, an inherent observability gap that limits external enforcement, component substitutability enabled by open standards, and informational persistence that decouples the knowledge base from any single commercial entity. These mechanisms are complementary and causally coupled — any erosion of the FOSS governance layer simultaneously weakens all three disturbance rejection pathways, a systemic property that single-domain analyses cannot capture. Their strength is nonetheless **regime-dependent**: most effective against governance disturbances operating at the informational layer ($\rho = 1$, file-distribution controls), and diminishing as regulatory intensity escalates toward hardware attestation and anti-circumvention mandates ($\rho = 3{-}4$). Within the regimes represented by current legislative proposals, these properties are not merely desirable features but structural consequences of the distributed architecture — a distinction the proposed framework makes precise through its mathematical formulation.

The framework provides a foundation for two lines of subsequent investigation. The first concerns validation: agent-based simulation and controlled empirical experiments would enable numerical computation of the robustness metrics and direct testing of the disturbance rejection mechanisms predicted here, while deployment of federated learning across a heterogeneous fabrication testbed would ground the ML compensation layer in practice. The second concerns extension: adaptive digital twin re-identification, time-varying network topology, and the coupled governance and information-security challenges posed by hardware-level regulatory enforcement each build on the formal foundations established here but lie outside the scope of the present analytical treatment.

Author Contributions: Conceptualization, B.D., J.R. and M.J.; methodology, B.D.; software, B.D.; validation, B.D., J.R., M.J. and M.Č.; formal analysis, M.J.; investigation, B.D.; resources, B.D. and J.R.; data curation, B.D. and M.Č.; writing — original draft preparation, B.D.; writing — review and editing, B.D., J.R., M.J. and M.Č.; visualization,

 doi:10.20944/preprints202603.1360.v1

B.D.; supervision, M.J.; project administration, B.D.; funding acquisition, B.D., J.R., M.Č. and M.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No new data were created or analyzed in this study. A companion curated repository of FOSS manufacturing tools is archived at https://doi.org/10.5281/zenodo.18976364 [33].

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
API	Application Programming Interface
CAD	Computer-Aided Design
CAM	Computer-Aided Manufacturing
CG	Cognitive Gateway
CNC	Computer Numerical Control
CPS	Cyber-Physical Systems
CPPS	Cyber-Physical Production Systems
DM	Decentralised Manufacturing
DT	Digital Twin
FDM	Fused Deposition Modelling
FL	Federated Learning
FOSS	Free and Open Source Software
MBSE	Model-Based Systems Engineering
ML	Machine Learning
SME(s)	Small and Medium-Sized Enterprise(s)

References

1. Verboeket, V.; Krikke, H. The disruptive impact of additive manufacturing on supply chains: A literature study, conceptual framework and research agenda. *Computers in industry (Print)* **2019**. https://doi.org/10.1016/J.COMPIND.2019.07.003.
2. Rayna, T.; West, J.R. Where digital meets physical innovation: Reverse salients and the unrealized dreams of 3D printing. *The Journal of product innovation management* **2023**. https://doi.org/10.1111/JPIM.12681.
3. Graebener, J.B.; Incer, I.; Murray, R.M. A Compositional Approach to Diagnosing Faults in Cyber-Physical Systems. In Proceedings of the Runtime Verification; Könighofer, B.; Torfah, H., Eds., Cham, 2026; pp. 438–456.
4. Allwood, J.M.; Duncan, S.; Cao, J.; Groche, P.; Hirt, G.; Kinsey, B.L.; Kuboki, T.; Liewald, M.; Sterzing, A.; Tekkaya, A.E. Closed-loop control of product properties in metal forming. *Cirp Annals-manufacturing Technology* **2016**. https://doi.org/10.1016/J.CIRP.2016.06.002.
5. Pan, S.; Trentesaux, D.; McFarlane, D.; Montreuil, B.; Ballot, E.; Huang, G.Q. Digital interoperability in logistics and supply chain management: state-of-the-art and research avenues towards Physical Internet. *Computers in Industry* **2021**, *128*, 103435. https://doi.org/https://doi.org/10.1016/j.compind.2021.103435.
6. Baldwin, C.Y.; Clark, K.B. *Design Rules, Volume 1: The Power of Modularity*; The MIT Press, 2000. https://doi.org/10.7551/mitpress/2366.001.0001.
7. Baldwin, C.Y. *Design Rules, Volume 2: How Technology Shapes Organizations*; The MIT Press, 2024. https://doi.org/10.7551/mitpress/7374.001.0001.
8. Benkler, Y. *The Wealth of Networks: How Social Production Transforms Markets and Freedom*; Yale University Press, 2006.
9. Meng, L.; McWilliams, B.; Jarosinski, W.; Park, H.; Jung, Y.; Lee, J.H.; Zhang, J. Machine Learning in Additive Manufacturing: A Review. *JOM* **2020**. https://doi.org/10.1007/S11837-020-04155-Y.
10. Omairi, A.; Ismail, Z.H. Towards Machine Learning for Error Compensation in Additive Manufacturing. *Applied Sciences* **2021**. https://doi.org/10.3390/APP11052375.

 doi:10.20944/preprints202603.1360.v1

11. Rosa, A.D.L.; Armani, A.; Golmohamadi, M. Defect Detection and Closed-loop Feedback Using Machine Learning for Fused Filament Fabrication. *Procedia CIRP* **2024**. https://doi.org/10.1016/J.PROCIR.2024.08.247.
12. Zhang, C.; Xie, Y.; Bai, H.; Yu, B.; Li, W.; Gao, Y. A survey on federated learning. *Knowledge-Based Systems* **2021**, *216*, 106775. https://doi.org/https://doi.org/10.1016/j.knosys.2021.106775.
13. Brusoni, S.; Henkel, J.; Jacobides, M.G.; Karim, S.; MacCormack, A.; Puranam, P.; Schilling, M. The power of modularity today: 20 years of "Design Rules". *Industrial and Corporate Change* **2023**, *32*, 1–10, [https://academic.oup.com/icc/article-pdf/32/1/1/49001156/dtac054.pdf]. https://doi.org/10.1093/icc/dtac054.
14. Grieves, M.; Vickers, J. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*; Kahlen, F.J.; Flumerfelt, S.; Alves, A., Eds.; Springer International Publishing: Cham, 2017; pp. 85–113. https://doi.org/10.1007/978-3-319-38756-7_4.
15. Tao, F.; Qi, Q.; Wang, L.; Nee, A.Y.C. Digital Twins and Cyber–Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison. *Engineering* **2019**. https://doi.org/10.1016/J.ENG.2019.01.014.
16. Villegas, L.F.; Macchi, M.; Polenghi, A. Digital twins in manufacturing: A unified conceptual framework. *Annual Reviews in Control* **2025**, *60*, 101031. https://doi.org/https://doi.org/10.1016/j.arcontrol.2025.101031.
17. Kennedy, M.C.; O'Hagan, A. Bayesian Calibration of Computer Models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **2001**, *63*, 425–464. https://doi.org/10.1111/1467-9868.00294.
18. Washington State Legislature. HB 2321 — Preventing Unlawful Manufacturing of Firearms by Requiring Blocking Technologies on Additive and Subtractive Manufacturing Equipment. House Bill 2321, 2025–2026 Regular Session, 2026. Introduced 12 January 2026; primary sponsor O. Salahuddin.
19. California State Assembly. AB-2047 — Firearms: 3-Dimensional Printing Blocking Technology. California Assembly Bill 2047, 2025–2026 Regular Session, 2026. Introduced 17 February 2026; mandates DOJ-certified firearm-blocking hardware/firmware for all consumer 3D printers sold in California; knowingly circumventing blocking technology is a misdemeanour; civil penalty up to $25,000 per offence; effective March 1, 2029.
20. New York State Assembly. S.9005/A.10005 — 2026–27 Executive Budget: Public Protection. New York State Executive Budget Bill, 2026–2027 Session, 2026. Requires firearm-blocking firmware in all 3D printers sold in New York; criminalises circumvention of blocking technology; effective April 1, 2026.
21. Madni, A.M.; Madni, C.C.; Lucero, S.D. Leveraging Digital Twin Technology in Model-Based Systems Engineering. *Systems* **2019**, *7*. https://doi.org/10.3390/systems7010007.
22. Tao, F.; Zhang, H.; Liu, A.; Nee, A.Y.C. Digital Twin in Industry: State-of-the-Art. *IEEE Transactions on Industrial Informatics* **2019**. https://doi.org/10.1109/TII.2018.2873186.
23. Kostakis, V.; Niaros, V.; Dafermos, G.; Bauwens, M. Design Global, Manufacture Local: Exploring the Contours of an Emerging Productive Model. *Futures* **2015**, *73*, 126–135. https://doi.org/10.1016/j.futures.2015.09.004.
24. Bonvoisin, J.; Molloy, J.; Haeuer, M.; Wenzel, T. Mapping the Types of Modularity in Open-Source Hardware. *Design Science* **2021**, *7*, e17. https://doi.org/10.1017/dsj.2021.11.
25. Mariscal-Melgar, J.C.; Omer, M.; Moritz, M.; Hijma, P.; Redlich, T.; Wulfsberg, J.P. Distributed Manufacturing: A High-Level Node-Based Concept for Open Source Hardware Production. In Proceedings of the Proceedings of CPSL 2022, 2022. https://doi.org/10.15488/12171.
26. Rajkumar, R.; Lee, I.; Sha, L.; Stankovic, J.A. Cyber-physical systems: The next computing revolution. *Design Automation Conference* **2010**. https://doi.org/10.1145/1837274.1837461.
27. Olfati-Saber, R.; Fax, J.A.; Murray, R.M. Consensus and Cooperation in Networked Multi-Agent Systems. *Proceedings of the IEEE* **2007**, *95*, 215–233. https://doi.org/10.1109/JPROC.2006.887293.
28. Monostori, L.; Kádár, B.; Bauernhansl, T.; Kondoh, S.; Kumara, S.; Reinhart, G.; Sauer, O.; Schuh, G.; Sihn, W.; Ueda, K. Cyber-physical Systems in Manufacturing. *CIRP Annals* **2016**, *65*, 621–641. https://doi.org/10.1016/j.cirp.2016.06.005.
29. Leitão, P.; Colombo, A.W.; Karnouskos, S. Industrial Automation Based on Cyber-Physical Systems Technologies: Prototype Implementations and Challenges. *Computers in Industry* **2016**, *81*, 11–25. https://doi.org/10.1016/j.compind.2015.08.004.
30. Dogančić, B.; Jokić, M.; Alujević, N.; Wolf, H. Structure Preserving Uncertainty Modelling and Robustness Analysis for Spatially Distributed Dissipative Dynamical Systems. *Mathematics* **2022**, *10*. https://doi.org/10.3390/math10122125.

31. Akkaya, I.; Derler, P.; Emoto, S.; Lee, E.A. Systems Engineering for Industrial Cyber–Physical Systems Using Aspects. *Proceedings of the IEEE* **2016**, *104*, 997–1012. https://doi.org/10.1109/JPROC.2015.2512265.
32. Sajadieh, S.M.M.; Noh, S.D. From Simulation to Autonomy: Reviews of the Integration of Artificial Intelligence and Digital Twins. *International Journal of Precision Engineering and Manufacturing-Green Technology* **2025**, *12*, 1597–1628. https://doi.org/10.1007/s40684-025-00750-z.
33. Dogančić, B. Awesome FOSS for Decentralised Manufacturing, 2026. Companion material to: Dogančić et al., Systems, 2026, https://doi.org/10.5281/zenodo.18976364.
34. Carrascosa, C.; Pico, A.; Matagne, M.M.; Rebollo, M.; Rincon, J. Asynchronous consensus for multi-agent systems and its application to Federated Learning. *Engineering Applications of Artificial Intelligence* **2024**, *135*, 108840. https://doi.org/10.1016/j.engappai.2024.108840.
35. Mu, S.; Yu, C.; Lin, K.; Lu, C.; Wang, X.; Wang, T.; Fu, G. A Review of Machine Learning-Based Thermal Error Modeling Methods for CNC Machine Tools. *Machines* **2025**, *13*, 153. https://doi.org/10.3390/machines13020153.
36. Wang, Y.; Cao, Y.; Qu, X.; Wang, M.; Wang, Y.; Zhang, C. A Review of the Application of Machine Learning Techniques in Thermal Error Compensation for CNC Machine Tools. *Measurement* **2025**, *243*, 116341. https://doi.org/10.1016/j.measurement.2024.116341.
37. Cui, C.; Zan, T.; Ma, S.; Sun, T.; Lu, W.; Gao, X. Thermal Image-Driven Thermal Error Modeling and Compensation in CNC Machine Tools Based on Deep Attentional Residual Network. *The International Journal of Advanced Manufacturing Technology* **2024**, *134*, 3153–3169. https://doi.org/10.1007/s00170-024-14280-6.
38. Wei, J.; He, W.; Lin, C.; Zhang, J.; Chen, X.; Xiao, J.; Xu, J. Optimizing Process Parameters of In-Situ Laser Assisted Cutting of Glass–Ceramic by Applying Hybrid Machine Learning Models. *Advanced Engineering Informatics* **2024**, *62*, 102590. https://doi.org/10.1016/j.aei.2024.102590.
39. Cao, C.; Zhao, Y.; Song, Z.; Dai, D.; Liu, Q.; Zhang, X.; Meng, J.; Gao, Y.; Zhang, H.; Liu, G. Prediction and Optimization of Surface Roughness for Laser-Assisted Machining SiC Ceramics Based on Improved Support Vector Regression. *Micromachines* **2022**, *13*, 1448. https://doi.org/10.3390/mi13091448.
40. Kailath, T. *Linear Systems*; Prentice-Hall Information and System Sciences Series, Prentice-Hall: Englewood Cliffs, NJ, 1980.
41. Sontag, E.D. *Mathematical Control Theory*; Vol. 6, *Texts in Applied Mathematics*, Springer: New York, NY, 1998. https://doi.org/10.1007/978-1-4612-0577-7.
42. Anderson, B.D.O. *Optimal Filtering*, dover ed ed.; Dover Books on Engineering, Dover Publications: Mineola, N.Y, 2005.
43. Bullo, F.; Cortés, J.; Martinez, S. *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*; Princeton University Press, 2009.
44. Newman, M.E.J. *Networks: An Introduction*; Oxford University Press: Oxford, 2010. https://doi.org/10.1093/acprof:oso/9780199206650.001.0001.
45. Rasmussen, C.E.; Williams, C.K.I. *Gaussian Processes for Machine Learning*, 3. print ed.; Adaptive Computation and Machine Learning, MIT Press: Cambridge, Mass., 2008.
46. Hornik, K.; Stinchcombe, M.; White, H. Multilayer Feedforward Networks Are Universal Approximators. *Neural Networks* **1989**, *2*, 359–366. https://doi.org/10.1016/0893-6080(89)90020-8.
47. Zhou, K.; Doyle, J.C.; Glover, K.; Doyle, J.C. *Robust and Optimal Control*; Prentice Hall: Upper Saddle River, NJ, 1996.
48. Scherer, C.W. Theory of Robust Control. Lecture notes, Delft University of Technology.
49. Moore, E.; Imteaj, A.; Hossain, M.Z.; Rezapour, S.; Amini, M.H. Blockchain-Empowered Cyber-Secure Federated Learning for Trustworthy Edge Computing, 2024, [arXiv:cs/2412.20674]. https://doi.org/10.48550/arXiv.2412.20674.
50. Li, Y.; He, S.; Li, Y.; Shi, Y.; Zeng, Z. Federated Multi-Agent Deep Reinforcement Learning Approach via Physics-Informed Reward for Multi-Microgrid Energy Management. *IEEE Transactions on Neural Networks and Learning Systems* **2024**, *35*, 5902–5914, [arXiv:eess/2301.00641]. https://doi.org/10.1109/TNNLS.2022.3232630.
51. Belay, M.A.; Rasheed, A.; Rossi, P.S. Digital Twin-Driven Communication-Efficient Federated Anomaly Detection for Industrial IoT, 2026, [arXiv:cs/2601.01701]. https://doi.org/10.48550/arXiv.2601.01701.