

Industrial Foundation Models: A Review of Architectures, Adaptation Strategies, and Deployment Challenges

Neelam Rajendra Sarode, Samruddhi Shantaram Shinde,

Mrs. Jaya Wagh, Mrs. Sonali Gosavi, Mrs. Diba Ansari

Department of Electrical Engineering

Guru Gobind Singh Polytechnic, Nashik

neelamsarode17@gmail.com, samruddhishinde972@gmail.com, jaya.wagh@ggsf.edu.in,

sonali.gosavi@ggsf.edu.in, diba.ansari@ggsf.edu.in

Abstract: *The rapid evolution of Foundation Models (FMs) has created new opportunities for intelligent industrial applications. However, deploying these models in real-world industrial environments remains challenging because of heterogeneous data sources, demanding operational reliability, diverse industrial processes, and computational constraints at the edge. This paper investigates recent developments in Industrial Foundation Models (IFMs) through a comparative examination of six complementary research studies. These studies address system-level IFM design through the IFMsys architecture and MetaIndux prototype, the application of foundation models in industrial manufacturing, the assessment of electromagnetic immunity in aero-engine control systems using digital models, security threat detection in Industrial Internet of Things (IIoT) networks using the LightGBM algorithm, the competencies and values that frame research software engineering, and model-based vibration damping for a cable-suspended robotic installation using input shaping. The reviewed approaches are evaluated with respect to their underlying objectives, methodological characteristics, validation strategies, reported performance, and implementation challenges. To support the comparative analysis, the paper incorporates architectural representations, lifecycle comparisons, quantitative visualizations derived from published findings, and structured comparison tables. The synthesis highlights six important considerations for advancing industrial intelligence: incorporating physical mechanisms and domain knowledge into model development, enabling resource-efficient model adaptation, integrating heterogeneous industrial data, validating system behaviour through simulation-based methods, strengthening cybersecurity, and adopting disciplined software-engineering practice. Based on these observations, a conceptual hybrid framework is developed to combine foundation models, industrial knowledge, digital representations, lightweight learning algorithms, model-based motion conditioning, and security-oriented components. The paper concludes by identifying research opportunities in trustworthy decision-making, computationally efficient deployment, model interpretability, and validation under practical operating conditions. These findings provide a foundation for further research toward dependable, scalable, and application-oriented IFMs for intelligent manufacturing and industrial automation.*

Keywords: Industrial foundation model, intelligent manufacturing, multimodal learning, parameter-efficient fine-tuning, digital twin, industrial IoT security, research software engineering, input shaping



I. INTRODUCTION

Foundation models, characterized by large-scale self-supervised pre training and adaptability to many downstream tasks, have achieved state-of-the-art results in language, vision, and speech [2]. Their promise for industry is considerable: predictive maintenance, quality inspection, process planning, and robot control all involve heterogeneous data and knowledge-intensive decisions that benefit from a shared, reusable model backbone.

Industrial deployment, however, differs sharply from consumer use. Industrial tasks require deterministic and trustworthy outputs, because prediction errors can exceed tolerance thresholds and create safety hazards; they involve specialized modalities such as CAD/CAE files, vibration, torque, and sensor time series; they span R&D, manufacturing, test, management, and maintenance scenarios; and they depend on long-term context across multiple processes [1]. Further barriers include training cost, inference latency on edge hardware, data scarcity and heterogeneity across sites, and security and ethical risks [2].

This paper reviews six works that address these issues from complementary angles. Ren et al. [1] propose a dedicated IFM architecture and prototype; Ayyat et al. [2] survey FMs and their industrial opportunities and challenges; Wang et al. [3] present a model-based, simulation-driven method for defining design requirements of safety-critical aero-engine control systems; Deng [4] develops a gradient-boosting intrusion detector for industrial control networks; Goth and Thiele [5] define the values and foundational competencies of the research software engineer; and Goubej et al. [6] refurbish a cable-suspended robotic installation using model-based input shaping for vibration damping (Table I). The first two are directly concerned with FMs, whereas [3]–[6] provide non-FM reference points—mechanism-based modeling, conventional machine learning, software-engineering practice, and model-based motion control—against which the claims and requirements for IFMs can be assessed.

The contributions of this review are: (i) a structured summary of the IFM architecture, adaptation methods, and application landscape, supported by redrawn diagrams; (ii) data plots constructed only from numbers reported in the reviewed papers; (iii) a comparative analysis of the six works, including a critical assessment of evidence strength; and (iv) a conceptual hybrid framework and a list of open directions. Section II gives background, Section III the IFM architecture, Section IV the adaptation and application landscape, Section V the four non-FM reference works, Section VI the comparison, Section VII the proposed framework, and Section VIII the open challenges.

TABLE I: SUMMARY OF THE REVIEWED WORKS

Work	Focus	Main contribution and evidence
Ren et al. [1], 2025	IFM system architecture	IFMsys (training, adaptation, application); agent-centric collaboration; MetaIndux prototype with qualitative case studies of six capabilities.
Ayyat et al. [2], 2025	Survey of FMs in manufacturing	FM fundamentals; 15 application areas; deployment challenges; seven solution strategies.
Wang et al. [3], 2024	Digital-modelbased requirement definition	Transfer-function method, $I(\text{HIRF}) = \text{TF} \times E(\text{HIRF})$; numerical case on a six-equipment engine control system.
Deng [4], 2024	IIoT intrusion detection	LightGBM-based detector; reports 96.2% accuracy, 95.6% recall, AUC 0.98 against five baselines.
Goth and Thiele [5], 2025	RSE competencies and values	Community-derived values and a three-pillar competency framework (software/technical, research, communication); twelve specializations; no quantitative evaluation.
Goubej et al. [6], 2026	Vibration damping, cable-suspended kinetic artwork	First-principles four-DoF model; dual-mode robust input shaper (convex QP); simulated 180° maneuver; exhibition deployment reported without measured data.



II. BACKGROUND

A. Characteristics of Foundation Models

Foundation models are large-scale models trained on broad data with self-supervision and then adapted to many downstream tasks; BERT, GPT-3, and CLIP are typical examples [2]. Two concepts capture their significance. *Emergence* refers to capabilities such as in-context learning that appear only after models pass a size and data threshold and are not explicitly trained; *homogenization* refers to the convergence of a few dominant architectures and training recipes across applications. Homogenization accelerates reuse and benchmarking but also propagates any flaw or bias of a widely used model to all of its derivatives [2].

The Transformer, with multi-head self-attention and positional encoding, is the dominant backbone. Its performance is described by empirical scaling laws, which relate loss L to the number of parameters N , dataset size D , and compute budget C [2]:

$$L(N, D, C) \propto N^{-\alpha} + D^{-\beta} + C^{-\gamma} \quad (1)$$

where α , β , and γ are empirically determined constants. Training cost is commonly estimated as a lower bound by $FLOPs = 6 \times N \times D$ (2), which explains why training FMs from scratch is out of reach for most manufacturers; GPT-3, for example, has 175 billion parameters and required tens of millions of dollars to train, according to [2].

B. Development Lifecycle

Fig. 1 contrasts the traditional machine learning workflow, in which each task has its own data selection, labeling, training, validation, and deployment, with the FM workflow, in which one expensive pretraining phase is shared and each task requires only adaptation, validation, and deployment [2]. The FM lifecycle consists of data curation, training, adaptation (few-shot learning or fine-tuning), evaluation and validation (including bias evaluation), and deployment with monitoring and version tracking.

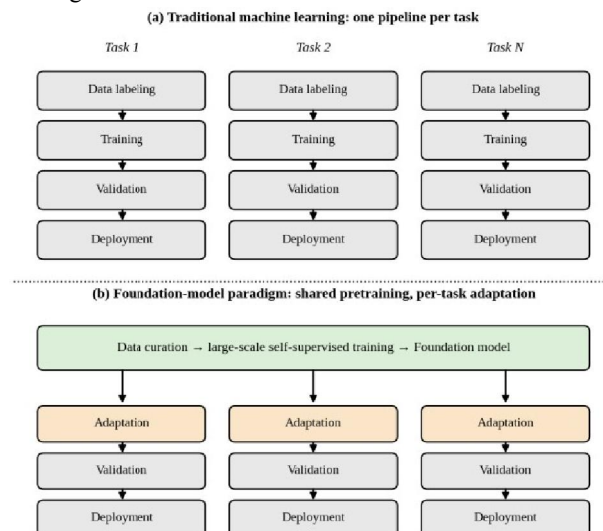


Fig. 1. Development cycles of traditional machine learning versus foundation models. Redrawn by the author from the description in [2].

C. Why Industry Needs Dedicated Models

Ren et al. [1] identify four challenges that general FMs face in industry. (1) *Untrustworthy output*: precision machining and defect inspection need deterministic results within tight tolerances, whereas FMs are probabilistic and may hallucinate. (2) *Cross-modal fusion*: vision, sound, vibration, temperature, and CAD data differ in sampling rate, format, and redundancy and are hard to align in space and time. (3) *Multiscenario inadaptation*: R&D, manufacturing, test, and maintenance have distinct requirements, and general models lack industrial domain knowledge. (4) *Contextual*



dissociation in multi process tasks: models with short-term memory cannot, for example, recall bolt positions across later assembly steps. These challenges motivate the architecture described next.

III. INDUSTRIAL FOUNDATION MODEL ARCHITECTURE

Ren et al. [1] define IFMs as large-scale neural network systems for the whole lifecycle of industrial products, comprising industrial base models, task-oriented models, and domain-specific models. Their system architecture, IFMsys, has three parts—model training, model adaptation, and model application—shown in Fig. 2.

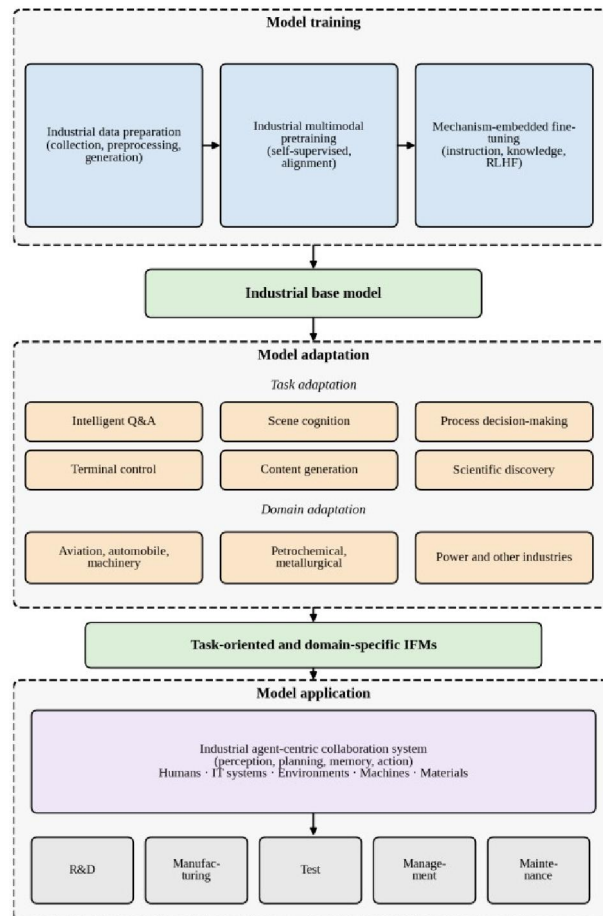


Fig. 2. Architecture of the industrial foundation model system (IFMsys). Redrawn by the author from the description in [1].

A. Model Training

Data preparation. Industrial data include CAX files, sensor time series, machine instructions, documents, images, video, and audio. Sensitive enterprise data can be handled with federated learning. Preprocessing comprises quality and sensitive-content filtering, de-duplication, text tokenization, image denoising, sensor normalization with sliding-window feature extraction, and CAD cleaning to remove superfluous geometry. Because industrial data are scarce, additional data are generated with simulation engines and digital twins (for example, aero-engine simulation) and with generative models such as diffusion-based time-series generators [1].

Multimodal pretraining. As shown in Fig. 3, the multimodal architecture has five components: modality encoders, an input projector that maps features into a shared space, a backbone network, an output projector, and modality



generators. Self-supervised objectives are adapted to industrial data, such as masked time-point prediction for time series and serialized topology/geometry encoding for CAD. Alignment is trained with objectives such as the InfoNCE contrastive loss, and [1] argues for industry-specific matching tasks such as CAX–text and sensor time series–text.

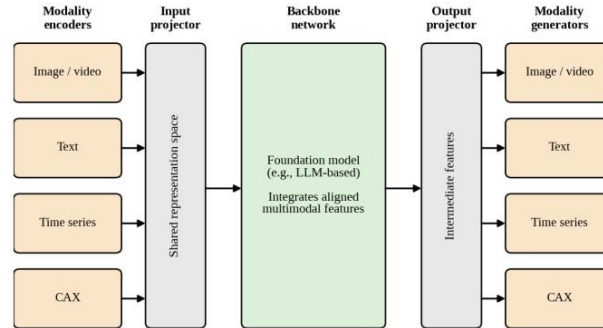


Fig. 3. Multimodal model architecture for IFM pretraining. Redrawn by the author from the description in [1].

Mechanism-embedded fine-tuning. This stage targets trustworthiness in three steps. First, multimodal instruction tuning teaches the model to follow industrial instructions with image–text, time series–text, or CAD–text samples. Second, industrial knowledge and physical mechanisms are embedded: knowledge is converted into knowledge-graph triples and embeddings and integrated through training objectives, model inputs, or knowledge-graph instruction tuning, while mechanism laws are added to the loss function to penalize physically inconsistent predictions, enforced by post-processing, or checked with simulation tools. Third, human preference alignment for industrial safety uses a reward model trained on annotated preferences that incorporate quality and efficiency standards, followed by reinforcement-learning fine-tuning with validation by simulation engines [1].

B. Model Adaptation

Task-oriented adaptation fine-tunes the base model on multitask instructions while freezing most layers to avoid overfitting on small datasets, and uses an input-layer adapter to tailor features to each task. Six task models are defined [1]: intelligent Q&A (trained on question–solution pairs from specifications and manuals), scene cognition (multimodal scene inputs to textual interpretation), process decision making (initial tuning on historical correct decisions followed by refinement with simulator feedback), terminal control (natural language to machine instructions such as Gcode, software code, or robot paths), content generation (natural language to CAX files, 3-D models, and trajectories), and scientific discovery (knowledge-graph-based exploration of mechanisms and materials). Domain-specific models for discrete industries (aviation, automobile, machinery) and process industries (petrochemical, metallurgical, power) are obtained with adapter fine-tuning and knowledge-graph fine-tuning, which retain task capabilities while learning domain knowledge.

C. Model Application

Because multi-process tasks require long-term memory and process association, [1] places the IFM at the core of an industrial agent with four functions: perception (collecting and preprocessing multimodal data), planning (decomposing tasks and adjusting schedules in real time), memory (correlating, compressing, and updating historical information), and action (outputting machine instructions, with embodied intelligence for hardware). The agent interacts with Humans, IT systems, Environments, Machines, and Materials: humans set objectives and reward–penalty rules; IT systems serve as an external knowledge base; the environment is sensed and modified; machines are controlled by generated instructions; and materials are monitored for quality and allocation. The six capabilities are mapped to five lifecycle stages—R&D, manufacturing, test, management, and maintenance—yielding a 6×5 application framework [1].



D. The MetaIndux Prototype

MetaIndux is the prototype built on IFMsys. Six illustrative studies are reported [1]: (1) root-cause analysis and repair procedures for a grounding disconnector failure in a 220-kV substation (Q&A); (2) interpretation of the function of a fixture-type assembly from its 3-D model (scene cognition); (3) a machining plan with G-code for a hole-board part (decision-making); (4) robot-arm code to recognize and grasp flanges (terminal control); (5) generation of aircraftengine sensor data for engines with fewer than 20 cycles of remaining useful life, using physical constraints such as the continuity equation and isentropic compression in the loss function (content generation); and (6) a copper–nitrogen doped carbon catalyst proposal for C₂H₄ production (scientific discovery). In each case, the output is compared with that of a general-purpose model; the comparison is qualitative.

IV. ADAPTATION AND APPLICATION LANDSCAPE

A. Adaptation Strategies

Ayyat et al. [2] observe that updating all parameters of a multi-billion-parameter model is often infeasible in industry because of compute, storage, and latency constraints, and review lighter alternatives. *Parameter-efficient fine-tuning* (PEFT) trains only small added modules, such as adapters, low-rank adaptation (LoRA) matrices, or prefix tokens; LoRA can reduce the trainable weights of a GPT-3-scale model from about 175 billion to about 37 million (Fig. 4a). *Prompt and instruction tuning* keeps weights fixed and steers behavior with crafted inputs; in the LLM4PLC case study, prompt engineering and LoRA adapters raised PLC code success rates from 47% to 72% (Fig. 4b). *Domain-adaptive pretraining* (DAPT) continues unsupervised training on domain text; an optoelectronics model required 80% fewer GPU-hours than training from scratch (Fig. 4c). *Federated fine-tuning* lets several sites adapt a model by exchanging only updates, preserving data privacy at the cost of slower convergence.

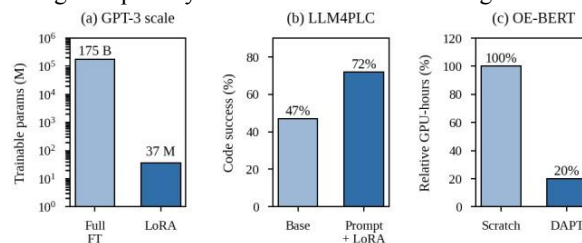


Fig. 4. Efficiency and accuracy effects of adaptation strategies as reported in [2]: (a) trainable parameters, full fine-tuning versus LoRA (log scale); (b) PLC code success rate before and after prompt engineering with LoRA; (c) relative GPU-hours of domain-adaptive pretraining versus training from scratch. Values are those quoted in [2]; the baseline in (c) is normalized to 100%.

B. Application Landscape

Table 4 of [2] maps application areas to data modalities; Fig. 5 reproduces this mapping for the 13 areas it lists (the text of [2] discusses 15, adding predictive maintenance and supply chain optimization). Language is the most common modality (10 of 13 areas), followed by vision and audio (five each), robotics and simulation (three each), and sensing (two). Quality control, worker behavioral analysis, and industrial safety combine four modalities, which supports the argument in [1] that multimodal alignment is central. The low count for sensing in Fig. 5 should be read with care: [2] also describes sensor-based predictive maintenance, in which FMs combine vibration, acoustic, thermographic, and log data, and cites reports of up to 50% less unplanned downtime and 10–40% lower maintenance expense, although these figures come from a secondary consulting source.



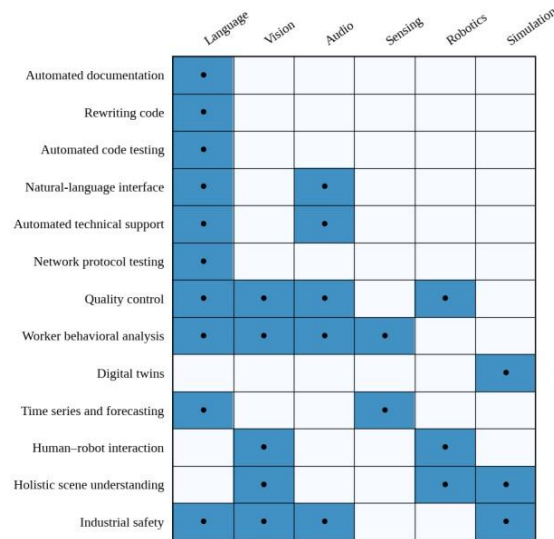


Fig. 5. Mapping of industrial FM application areas to data modalities. Replotted by the author from Table 4 of [2].

C. Deployment Challenges and Solution Strategies

The challenges reviewed in [2] span the model lifecycle: (1) data divergence and heterogeneity across sites, where small differences in sensors, lighting, or protocols cause distribution shift; (2) training cost; (3) inference latency and cost, addressed with quantization (e.g., GPTQ), pruning, early-exit, distillation, and conditional computation; (4) data scarcity for rare failure modes; (5) model and dataset bias; (6) lack of standardization and difficulty of integration with legacy systems; (7) security threats such as prompt injection and membership inference; and (8) ethical issues including fairness, transparency, privacy, and workforce impact. The seven solution strategies proposed in [2] are modular finetuning, conditional computation, multi-layered runtime guardrails, simulation-aligned development with digital twins, AI co-pilots, explainability and traceability, and governance with ROI justification.

V. NON-FM REFERENCE APPROACHES

A. Digital-Model-Based Requirement Definition

Wang et al. [3] address the definition of HIRF immunity requirements for a commercial aircraft engine control system. Document-based requirements from industry manuals ignore the configuration of the system and the limited shielding of composite nacelles, whereas physical tests need a manufactured engine and nacelle. The proposed procedure (Fig. 6) builds digital models of topology, signal transmission, electric loading, and cable harness, places a cable probe at a measurement point 5 cm from the controller, and illuminates the model with unit-magnitude plane waves to obtain the transfer function

$$TF = I_{\text{unit}}(s) / E_{\text{unit}}(s) \quad (3)$$

The induced current under the certification environment is then

$$I_{\text{HIRF}} = TF \times E_{\text{HIRF}} \quad (4)$$

and the requirement for the validation test must satisfy

$$I_{\text{req}} > I_{\text{HIRF}} \quad (5)$$



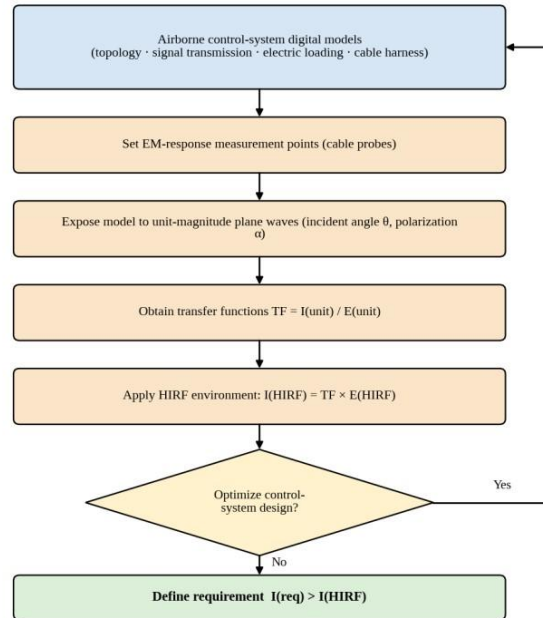


Fig. 6. Digital-model-based procedure for defining HIRF immunity requirements. Redrawn by the author from [3]. In the case study, six pieces of equipment (a controller, an aircraft interface unit, and four sensors) are connected by a multi-branch shielded harness. Plane waves with incidence angles of -90° , 0° , 90° , and 180° and polarizations of 0° and 90° are used with 30 frequency points from 0.01 to 400 MHz, generated by $f_{n+1} = f_n \times 10^{1/k}$ (6) and responses are computed with a multiconductor transmission line algorithm. Fig. 7 plots the induced current from the tabulated results of [3], together with the document-based and proposed requirements. The induced current exceeds the 200 mA document-based level at 15.8 MHz in both polarizations (about 354 mA and 403 mA), because the aircraft interface is exposed outside the nacelle. The proposed requirement raises the level to 500 mA at 10–30 MHz and envelops all induced responses; elsewhere it coincides with the document-based level. The plotted lines between tabulated frequencies are straight-line interpolations by the author.

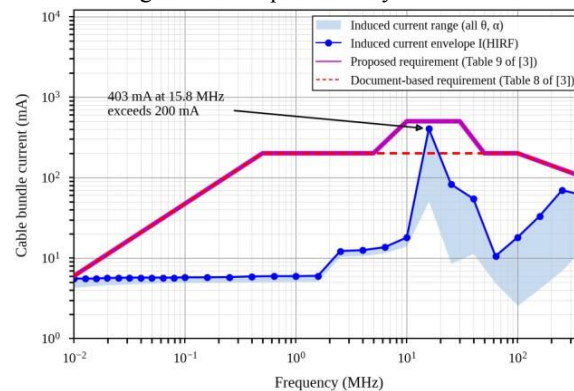


Fig. 7. Induced cable bundle current in the certification environment and the two requirement definitions. Plotted by the author from Tables 6–9 of [3]; envelope is the maximum over four incidence angles and two polarizations.

B. LightGBM-Based IIoT Intrusion Detection

Deng [4] describes the layered structure of industrial control networks (field, monitoring, and management layers) and argues that detection systems must be both accurate and fast. The detector uses LightGBM, a gradient-boosting method



with leaf-wise tree growth under a depth limit, gradient-based one-side sampling (GOSS), and exclusive feature bundling (EFB). Inputs from KDD CUP 99, NSLKDD, and UNSW-NB15 are cleaned and standardized with the Z-score,

$y_j = (x_j - w_j) / s_{ij}$ (7) after which category encoding is applied and features with importance above 0.1 are retained. Using an 80/20 stratified split, LightGBM is compared with decision tree, random forest, XGBoost, k-NN, and SVM. As summarized in Fig. 8, it attains the highest AUC (0.98 versus 0.80–0.93) and reported accuracy, precision, recall, and F1 of 96.2%, 97.4%, 95.65%, and 96.51%. The reported precision, recall, and F1 are mutually consistent, since $F1 = 2PR/(P + R) \approx 0.965$. The cost is a detection time of 1.01 s versus 0.23 s for the decision tree.

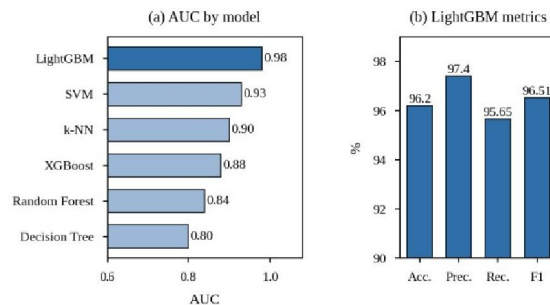


Fig. 8. Reported results of the LightGBM intrusion detector: (a) AUC of six models; (b) accuracy, precision, recall, and F1 of LightGBM (vertical axis starts at 90%). Plotted by the author from the values reported in [4].

C. Research Software Engineering Competencies

Goth and Thiele [5] condense community workshop discussions (deRSE23, un-deRSE23, and deRSE24 in Germany) into a domain-independent set of values and foundational competencies for research software engineers (RSEs), who occupy the spectrum between researcher and software engineer. The competencies are grouped into three pillars (Fig. 9): software/technical skills (classical software-engineering skills, adapting to the software lifecycle, documented reusable code, distributable software, software repositories, and behavior analysis), research skills (conducting research, understanding the research cycle, software reuse, publication and citation, and domain repositories), and communication skills (teamwork, teaching, project management, and interaction with users and stakeholders). Twelve specializations are listed, among them an ML-RSE, a data-focused RSE, an HPC-RSE, and a legal RSE.

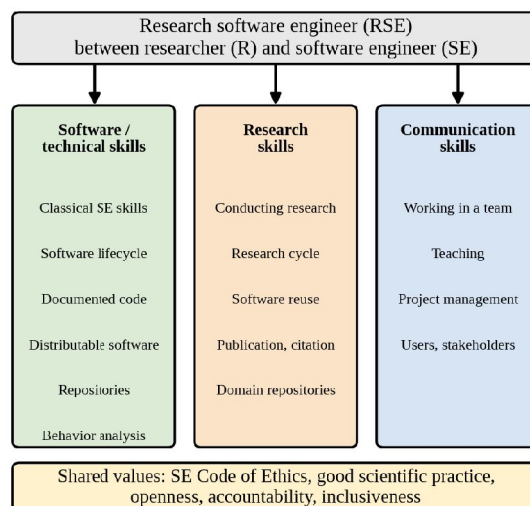


Fig. 9. Competency framework of the research software engineer. Redrawn by the author from the description in [5].



The values are loosely based on the Software Engineering Code of Ethics and on good scientific practice: objectivity and honest conclusions, openness and accountability, adherence to best practices, professional integrity, respect and inclusiveness, and passing these values on to new practitioners. The article also notes that the outputs of computer simulations, particularly those of computationally irreducible systems such as artificial neural networks, are “epistemically opaque,” and that RSEs carry responsibility for assessing the adequacy of a model for its purpose and for communicating the limits of its interpretability. It further discusses data integrity, privacy, the FAIR principles for research software, and the environmental cost of large language models, for which it points to the GREENER principles. The article is a condensed community position paper and contains no quantitative evaluation. Its connection to IFMs is the review’s own reading: building, adapting, and deploying an IFM involves software artifacts, such as data pipelines, adapters, simulators, and benchmarks, to which the practices of testing, documentation, versioning, and reuse described in [5] apply.

D. Model-Based Vibration Damping by Input Shaping

Goubej et al. [6] describe the refurbishment of the interactive installation *Standards and Double Standards*, in which belts suspended from the ceiling by cables rotate to follow visitors tracked by an overhead camera. The original system limited rotational velocity and acceleration because of torsional and pendulum-like oscillations of the flexibly suspended belts. In the refurbished system (Fig. 10), a Mac Mini runs a YOLOv8-based person detector and tracker, an ESP32 microcontroller receives angular-offset commands over a serial link, and a stepper driver actuates a geared rotary stage with positional updates at 100 Hz; the reported hardware cost is USD 210 per belt unit.

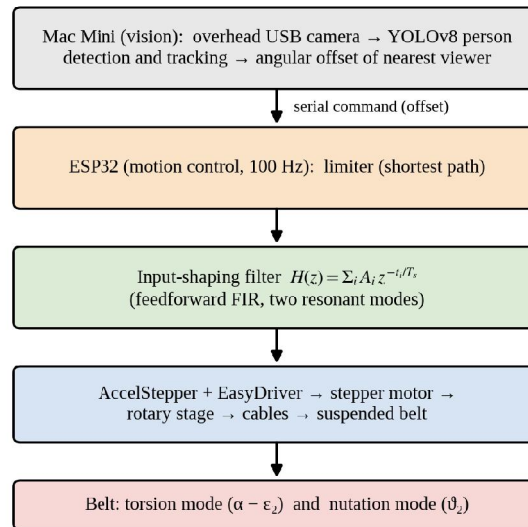


Fig. 10. Motion-control pipeline of the refurbished flying-belt installation. Redrawn by the author from the description in [6].

The authors derive a first-principles multibody model with four degrees of freedom, assuming rigid cables loaded in tension, a rigid belt with a metal buckle, and an ideal revolute actuator. Linearization about the hanging equilibrium yields two dominant oscillatory modes, torsion (angle ϵ_2) and nutation (angle ϑ_2), with natural frequencies of 2.58 and 3.55 rad/s for the simulated parameters. For vibration damping, they use a feedforward lumped-delay input shaper, which in digital form is the finite-impulse-response filter

$$H(z) = \sum A_i z^{-(i-1)}, i = 1, \dots, n \quad (8)$$

where A_i are the amplitudes of pulses spaced by the sampling period T_s (pulse times $t_i = (i-1)T_s$). The amplitudes are found by solving a convex quadratic program that enforces zero-vibration conditions for both modes, robustness to uncertain modal parameters, and a monotonic step response while minimizing the H_2 norm of the shaper; a smoothing



factor s_f trades settling time against smoothness. Only the resonant frequencies, measured experimentally or taken from the model, are needed, and the filter is executed in real time as a weighted sum of current and past inputs. In a point-to-point simulation of a 180° rotation, four motion-planning strategies were compared (Fig. 11): constant velocity, a jerk-limited polynomial profile, a time-optimal input shaper (T-Opt IS), and an H_2 -optimal shaper-smoother with $s_f = 0.15$. The H_2 -optimal shaper reduced the peak-to-peak torsion error from 57.2 (constant velocity) and 5.07 (polynomial) to 0.0941 and the belt settling time from more than 60 s to 5.4 s. The nutation error is not reduced to the same degree (0.869 versus 0.688 peak-to-peak for the polynomial profile), which the authors attribute to centrifugal forces acting on the asymmetrically suspended load.

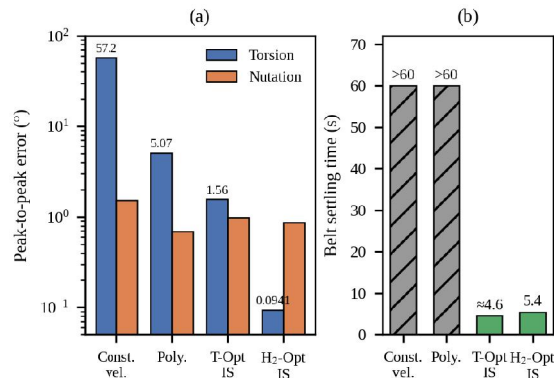


Fig. 11. Point-to-point simulation results of four motion-planning strategies: (a) peak-to-peak torsion and nutation errors (log scale; degrees as in Fig. 8 of [6]); (b) belt settling time (hatched bars are reported as >60 s). Plotted by the author from Table 3 of [6]; IS, input shaper.

The experimental section of [6] reports a gallery presentation (about 350 visitors at the opening reception) and a demonstration video, not measured performance data; the quantitative comparison therefore rests on the simulation. For IFMs, [6] is a reference for the actuator end of the chain. The terminal-control capability of [1] lets a model emit machine instructions and robot paths, whereas [6] shows that a trajectory that is kinematically correct can still excite resonances of the physical system, and that a model-based feedforward layer needing only the resonant frequencies can condition the commanded motion. This transfer is the review's interpretation; [6] does not involve an FM.

VI. COMPARATIVE ANALYSIS

A. Problem Framing and Coverage

Both FM papers argue that generic FMs do not transfer directly to industry, but they emphasize different aspects. Ren et al. [1] stress trustworthiness, cross-modal alignment, and long-horizon process context, and respond with a purpose-built architecture. Ayyat et al. [2] adopt a wider lens that includes cost, data availability, bias, integration, and security, and respond with a catalogue of techniques. [2] cites [1] when calling for domain-specific industrial benchmarks. Table II summarizes which criteria each work addresses, according to the author's reading. Works [5] and [6] are not FM studies; they are included as non-FM reference points for software-engineering practice and for model-based motion control, respectively.

TABLE II: COVERAGE OF KEY CRITERIA (AUTHOR'S ASSESSMENT)

Criterion	[1]	[2]	[3]	[4]	[5]	[6]
Uses FM / LLM	Yes	Yes	No	No	No	No
Multimodal data	Yes	Yes	Part.	No	No	No
Mechanism / physics grounding	Yes	Part.	Yes	No	No	Yes



Parameter-efficient adaptation	Yes	Yes	No	No	No	No
Security/ethics addressed	Part.	Yes	No	Yes	Yes	No
Industrial case study	Yes	Part.	Yes	Yes	No	Part.
Quantitative evaluation	Part.	Part.	Yes	Yes	No	Yes

B. Architecture, Training, and Adaptation

Both works build on Transformer backbones and self-supervised pre-training. Ren et al. specify industrial encoders and projectors for CAD, time-series, text, and image data (Fig. 3), whereas Ayyat et al. discuss modality-specific extensions (vision Transformers, Perceiver IO, efficient longsequence models) at the lifecycle level (Fig. 1). On adaptation they agree that one model cannot serve all scenarios: [1] uses layer freezing, adapters, and knowledge-graph fine-tuning, and [2] generalizes these into PEFT, prompt tuning, DAPT, and federated fine-tuning (Fig. 4). The principal open design question is therefore how industrial modalities are tokenized, aligned, and combined, not the backbone alone; [1] itself lists a novel industrial-oriented backbone as a future need.

C. Trustworthiness Through Mechanism and Simulation

A recurring theme is grounding outputs in physical mechanisms. Ren et al. add mechanism equations to the loss, apply post-processing consistent with industrial rules, and validate outputs in simulators. Ayyat et al. propose simulation-aligned development and runtime guardrails. Wang et al. [3] illustrate the deterministic end of this spectrum: a physics-based model yields a transfer function and an auditable requirement (Figs. 6 and 7), in contrast to the probabilistic outputs of FMs. A hybrid pattern—FMs propose and interpret, mechanism models verify—is therefore attractive and is elaborated in Section VII. Goubej et al. [6] apply the same logic at the actuator level: a first-principles model identifies the dominant resonant modes, and a feedforward input shaper designed from them conditions the commanded motion without direct oscillation measurement (Figs. 10 and 11).

D. Data Modalities and Engineering Effort

Deng [4] works on tabular network-traffic features and relies on explicit feature selection and encoding, which contrasts with the representation learning expected from FMs and shows the data-engineering effort that unified pretraining aims to reduce. Wang et al. [3] use structured system models rather than learned representations. Both are efficient and transparent within their narrow scope, but neither generalizes across tasks the way an IFM is intended to. In [6], the only learned component is a generic, off-the-shelf YOLOv8 detector for perception, whereas motion control relies on an explicit physical model and a designed filter. Goth and Thiele [5] place engineering effort elsewhere: testing, documentation, reproducibility, and software publication are treated as core competencies, an effort that the data, model, and simulator pipelines of an IFM system also require.

E. Security, Ethics, and Governance

Ayyat et al. [2] devote considerable attention to prompt injection, membership inference, bias, privacy, transparency, workforce impact, and regulation. Ren et al. [1] address safety mainly through preference alignment and simulation checks. Deng [4] addresses a complementary problem: protecting the industrial control network on which IFM-driven agents will operate. Because the agent-centric system in [1] lets models issue machine instructions to robots, AGVs, and machine tools, a compromise could have physical consequences, so FM-level safeguards and network-level detection should be considered together. Goth and Thiele [5] supply a values-based complement: the Software Engineering Code of Ethics, openness and accountability, care for data integrity and personal data, and awareness of the environmental footprint of large language models, with the GREENER principles as a pointer. These are professional norms rather than technical safeguards, and [5] does not evaluate them quantitatively.

F. Strength of Evidence and Limitations

The MetaIndux case studies in [1] are qualitative and compare with a general-purpose model (e.g., GPT-4o in the time-series example) without quantitative metrics in the text reviewed here, so claims of superior trustworthiness remain to be validated on standardized benchmarks. The survey in [2] is broad, and some covered applications (e.g., network



protocol testing, code testing) lie at the edge of manufacturing; several quantitative figures it quotes come from secondary sources. The study in [3] is numerically validated on a simplified system with assumed engine and nacelle models; it does not report comparisons with physical measurements. The study in [4] provides consistent comparative results. Still, its title and method emphasize fusion with a multi-layer perceptron. In contrast, the experiments mainly compare LightGBM with other classifiers, do not break down results by dataset, and a few figures differ slightly between the text and the conclusion (accuracy margin 5.6% versus 5.7%; precision 97.4% versus 97.3%). The article in [5] is a condensed community position paper derived from workshop discussions and contains no quantitative evaluation; its authors describe the competency list as a first step. The study in [6] reports its quantitative comparison in numerical simulation with a model whose parameters are chosen to match an existing physical setup; its experimental section describes a gallery presentation and a demonstration video rather than measured results, and the shaper does not improve the nutation error relative to the polynomial profile (Fig. 11a). Neither [5] nor [6] involves an FM, so their relevance to IFMs is by analogy and reflects the review's interpretation.

PROPOSED CONCEPTUAL FRAMEWORK

Synthesizing the reviewed works, we propose the hybrid framework in Fig. 12. It is a conceptual proposal of this review and has not been evaluated experimentally. The IFM agent of [1], adapted with parameter-efficient methods [2], receives multimodal industrial data and knowledge and proposes actions or analyses. A mechanism and simulation verifier, in the spirit of the model-based requirement checking in [3] and the simulation-aligned development in [2], evaluates each proposal against physical models or a digital twin and returns feedback. Approved outputs pass through runtime guardrails and, where required, human approval before commands reach machines. Approved motion commands are conditioned at the actuator interface by a model-based feedforward filter, such as the input shaper of [6], so that generated trajectories do not excite resonances of the physical system. An intrusion detection layer, as in [4], monitors the network and the machine interface so that a compromised input or command channel can be detected and the guardrails tightened. Across all components, the engineering practices described in [5]—testing, documentation, version control, FAIR publication, and ethical conduct—apply to the software artifacts that make up the framework.

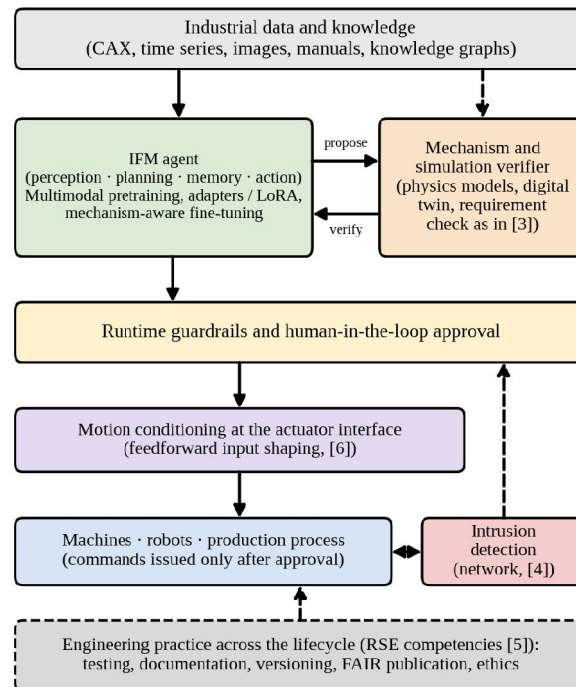


Fig. 12. Proposed hybrid framework combining an IFM agent, a mechanism and simulation verifier, runtime guardrails, actuator-level motion conditioning, and network-level intrusion detection, supported by software-engineering practice. Conceptual figure synthesized by the author from [1]–[6]; dashed arrows indicate monitoring and data-quality links.

OPEN CHALLENGES AND FUTURE DIRECTIONS

Drawing on the reviewed works, we identify the following directions.

Trustworthy outputs. Combining learned models with mechanism embedding, simulation validation, and runtime guardrails to meet industrial tolerances and reduce hallucination [1], [2].

Novel backbones and world models. Simple Transformer backbones are considered insufficient for multimodal, cross-process industrial demands; an industry-oriented backbone and a world model to guide training and deployment are proposed in [1].

Unified multimodal representation. A satisfactory unified representation for CAX, time-series, and other industrial modalities remains unresolved [1].

Industrial benchmarks. Existing benchmarks emphasize language and vision; benchmarks capturing sensor data, CAD, multi-step workflows, fault tolerance, and legacy integration are needed [2].

Adaptive knowledge and mechanism recall. As knowledge and mechanism libraries grow, retrieval errors, knowledge conflicts, and incorrect mechanism invocation arise; adaptive recall for each scenario is needed [1].

Efficient adaptation and real-time inference. PEFT, distillation, quantization, and conditional computation must be tailored to edge hardware, latency limits, and heterogeneous computing resources without unacceptable accuracy loss [1], [2].

Interpretability and safety. Decision traceability, safety-aware architectures, and human-centric control are required in regulated, safety-critical settings [1], [2], [3]. Responsibility for assessing and communicating the limits of interpretability of opaque computational models is likewise emphasized for RSEs [5].

Collaboration with specialized models and conventional ML. The strong results of well-engineered classical methods [4], physics-based models [3], and model-based motion control [6] suggest that IFMs should orchestrate, rather than replace, specialized components [1].

Security of IFM-driven systems. Joint treatment of model-level attacks and network-level intrusions is needed as agents gain control over physical equipment [2], [4].

Engineering practice and sustainability. Treating IFM tooling as research-grade software—tested, documented, versioned, and published under FAIR principles [5]—and accounting for the cost of large-model training [2], [5] remain open needs.

Actuator-level safety of generated commands. The combination of model-based feedforward conditioning, as in [6], with the machine instructions generated by IFMs [1] should be evaluated with measured rather than simulated performance.

IX. CONCLUSION

This review examined six works around the theme of industrial foundation models. Ren et al. contribute a comprehensive IFM architecture and prototype; Ayyat et al. contribute a broad survey of opportunities, adaptation techniques, and challenges; Wang et al. demonstrate mechanism-grounded requirement definition that parallels the mechanism-embedding aims of IFMs; Deng shows the performance and engineering profile of a conventional ML detector for IIoT security; Goth and Thiele define the values and competencies that frame research software engineering practice; and Goubej et al. show how a first-principles model and a feedforward input shaper suppress vibration in a physical installation.

Taken together, they indicate that IFMs are most credible when combined with domain mechanisms, parameter-efficient adaptation, simulation-based verification, model-based motion conditioning, security safeguards, and



disciplined software-engineering practice, and that rigorous quantitative evaluation on industrial benchmarks is the most pressing gap.

REFERENCES

1. L. Ren, H. Wang, J. Dong, Z. Jia, S. Li, Y. Wang, Y. Laili, D. Huang,
2. L. Zhang, and B. Li, "Industrial foundation model," *IEEE Trans. Cybern.*, vol. 55, no. 5, pp. 2286–2301, May 2025, doi: 10.1109/TCYB.2025.3527632.
3. M. Ayyat, M. Osman, and T. Nadeem, "Opportunities and challenges of foundation models in industrial manufacturing," *IEEE Access*, vol. 13, pp. 138745–138775, 2025, doi: 10.1109/ACCESS.2025.3589777.
4. B. Wang, S. Zhang, and Z. Zhang, "A digital-model-based approach for defining HIRF immunity design requirements of commercial aircraft engine control systems," *IEEE Access*, vol. 12, pp. 110942–110954, 2024, doi: 10.1109/ACCESS.2024.3435132.
5. Y. Deng, "Design of industrial IoT intrusion security detection system based on LightGBM feature algorithm and multi-layer perception network," *J. Cyber Secur. Mobility*, vol. 13, no. 2, pp. 327–348, 2024, doi: 10.13052/jcsm2245-1439.1327.
6. F. Goth and J. P. Thiele, "Foundational competencies and specializations of a research software engineer," *Comput. Sci. Eng.*, pp. 27–34, Apr.–Jun. 2025, doi: 10.1109/MCSE.2025.3552156.
7. M. Goubej, L. Clarke, M. Hrabáčka, and D. Tolar, "Vibration damping in underactuated cable-suspended artwork: Flying belt motion control," *IEEE Robot. Autom. Mag.*, pp. 116–126, Sep. 2026, doi: 10.1109/MRA.2026.3687282

