Abstract

Target length: ~300 words **Status:** Draft v0.1 **Format:** IEEE Systems Journal

---

ABSTRACT

Modern engineered systems—spanning aerospace, defense, healthcare, and energy domains—exhibit complexity that strains the capacity of systems engineering organizations. While individual AI assistants help with specific tasks, they face fundamental limitations when applied to systems engineering's inherently collaborative, multi-disciplinary nature. This paper presents a framework for applying agentic AI swarms to systems engineering: coordinated multi-agent systems in which specialized AI agents collaborate to support lifecycle activities.

We trace the evolution of AI capabilities from expert systems through large language models to agentic swarms, establishing the technological foundation for comprehensive systems engineering support. We present an architectural framework comprising agent architecture taxonomies, systems engineering-specific specialization patterns aligned with discipline roles, coordination mechanisms, and swarm topologies with trade-off analysis.

The core contribution systematically maps swarm capabilities to ISO/IEC/IEEE 15288 technical process areas, demonstrating applicability across stakeholder needs definition, requirements engineering, architecture definition, design, integration, verification, validation, and lifecycle support. We assess maturity across process areas, finding concentrated development in requirements and verification with substantial opportunities elsewhere.

We chart a phased transformation roadmap from near-term task augmentation through mid-term coordinated multi-agent support to long-term collaborative human-AI teams, identifying enabling technologies and organizational readiness factors. Domain examples across aerospace, defense, healthcare, and energy validate framework generality.

We categorize challenges as technical (scalability, reliability, domain knowledge), integration (tools, processes, data), human factors (trust, oversight, skills), and organizational (governance, certification, accountability). A prioritized research agenda addresses each challenge category with calls to action for practitioners, researchers, organizations, and standards bodies.

For systems engineering practitioners and researchers, this paper provides a framework for understanding how AI swarms may augment engineering capability while preserving human judgment and accountability essential to engineering practice.

Index Terms: Agentic AI, multi-agent systems, swarm intelligence, systems engineering, ISO 15288, human-AI collaboration, MBSE.

Word count: ~310 words **Index Terms:** 7

Revision Notes

- ☐ Verify word count meets IEEE requirements (typically 150-250 words)
- ☐ May need to condense
- ☐ Ensure index terms align with IEEE taxonomy

Section 1: Introduction

Target length: ~1,000 words **Status:** Draft v0.1

---

1. Introduction

1.1 The Capacity Challenge

Modern engineered systems exhibit unprecedented complexity. A contemporary satellite may contain millions of lines of software code, thousands of hardware components, and interfaces spanning dozens of organizations. Medical devices integrate mechanical, electrical, software, and biological subsystems under stringent regulatory constraints. Power grids interconnect generation, transmission, and distribution assets across continental scales while incorporating renewable sources, storage systems, and demand-side management.

Systems engineering provides the disciplined processes for managing this complexity—transforming stakeholder needs into verified, validated systems through structured lifecycle activities [1, 2]. Yet the systems engineering enterprise itself faces a capacity challenge. The breadth of expertise required, the volume of artifacts to be produced and reviewed, and the interdependencies to be managed strain the capabilities of engineering organizations operating under schedule and budget constraints. The gap between system complexity and engineering capacity continues to widen.

1.2 A Vision of the Future

Consider systems engineering practice a decade hence. Dr. Sarah Chen arrives at an aerospace engineering center to lead a preliminary design review for a next-generation satellite constellation. Her morning begins not with a stack of documents to review, but with a briefing from her AI swarm team.

"Good morning, Sarah," her lead systems agent summarizes. "Overnight, the requirements analysis swarm completed consistency checking across 2,847 requirements. We identified 12 potential conflicts between thermal management and power subsystem requirements—three are likely specification errors, nine represent legitimate trade-offs requiring your judgment. The architecture exploration swarm evaluated 340 configuration variants against mission effectiveness criteria. I've prepared a short-list of 5 candidates that dominate on at least three key performance parameters."

Sarah reviews the conflict summary. The AI has traced each conflict to its source requirements, identified affected interfaces, and proposed resolution options with predicted impacts. What would have taken her team weeks of manual analysis is available before her first meeting.

During the design review, discipline specialist agents participate alongside human engineers. When a question arises about antenna placement impacts on thermal dissipation, the thermal agent and RF agent engage in rapid dialogue, exploring the trade space while humans observe. Within minutes, they present a visualization of the Pareto frontier with recommended configurations. A human engineer adds a constraint the agents hadn't considered—heritage antenna mounting provisions—and the swarm instantly re-evaluates.

This scenario illustrates the potential of agentic AI swarms for systems engineering—coordinated multi-agent systems in which specialized AI agents collaborate to support engineering activities across the lifecycle. The vision is not AI replacement of engineers but AI amplification of engineering capability.

1.3 Research Questions

Realizing this vision requires addressing fundamental questions about how AI swarms should be architected, coordinated, and applied to systems engineering. This paper addresses three research questions:

RQ1: What architectural patterns define agentic AI swarms for systems engineering? We seek to characterize the structural elements—agent architectures, specialization patterns, coordination mechanisms, and topologies—that enable effective swarms for SE applications.

RQ2: How do agentic AI swarm capabilities map to systems engineering process areas? We aim to systematically map swarm capabilities to the technical processes defined in ISO/IEC/IEEE 15288 [2], demonstrating applicability across the SE lifecycle.

RQ3: What are the key challenges, enabling factors, and research directions for AI-augmented systems engineering? We identify obstacles to adoption, chart a transformation path from current practice to the envisioned future, and propose a prioritized research agenda.

1.4 Contributions

This paper makes the following contributions:

1. **Architectural framework.** We present taxonomies of agent architectures and coordination mechanisms, synthesizing multi-agent systems research with emerging LLM-based agent capabilities, and define agent specialization patterns aligned with SE disciplines (Section 4).
2. **Process mapping.** We provide systematic mapping of swarm capabilities to ISO 15288 technical process areas, demonstrating applicability across the SE lifecycle with maturity assessment (Section 5).
3. **Transformation roadmap.** We chart a phased transformation path from current practice to collaborative human-AI teams, identifying enabling technologies and organizational readiness factors (Section 6).
4. **Domain validation.** We illustrate framework applicability across aerospace, defense, healthcare, and energy domains, identifying cross-domain patterns that validate domain-agnostic generality (Section 7).
5. **Research agenda.** We categorize challenges, prioritize research directions, and issue calls to action for practitioners, researchers, organizations, and standards bodies (Section 8).

1.5 Paper Structure

The remainder of this paper is organized as follows. Section 2 provides background on the evolution of AI capabilities and systems engineering processes. Section 3 reviews related work in multi-agent systems, swarm intelligence, and AI for systems engineering. Section 4 presents our architectural framework. Section 5 maps swarm capabilities to SE process areas. Section 6 charts the transformation roadmap. Section 7 illustrates domain applications. Section 8 discusses challenges and research directions. Section 9 reflects on implications and limitations. Section 10 concludes.

Word count: ~820 words **References cited:** [1], [2]

Revision Notes

- ☐ Verify opening statistics against current sources
- ☐ Consider expanding scenario with additional lifecycle phase
- ☐ Ensure RQs align precisely with section content

Section 2: Background

Target length: ~1,500 words **Status:** Draft v0.1

---

2. Background

This section establishes the foundations underlying agentic AI swarms for systems engineering, tracing the evolution of AI capabilities and defining the systems engineering process framework to which swarm capabilities will be mapped.

2.1 Evolution of AI Capabilities

Artificial intelligence capabilities relevant to systems engineering have evolved through distinct generations, each offering new possibilities for engineering support.

Expert systems (1970s-1980s) encoded human expertise as rule-based knowledge systems [3]. Early SE applications included configuration management assistants, fault diagnosis systems, and design rule checkers. These systems demonstrated AI value for structured engineering tasks but required extensive knowledge engineering and could not adapt beyond their encoded rules.

Machine learning (1990s-2010s) introduced statistical learning from data [4]. SE applications expanded to include defect prediction, cost estimation, and requirements classification. ML enabled pattern recognition beyond explicit programming but required substantial training data and remained narrow in capability.

Deep learning (2012-2020) achieved breakthroughs in perception and representation learning [5]. Foundation architectures including transformers [6] enabled processing of sequential data including natural language. SE applications emerged in requirements analysis, design pattern recognition, and automated code generation.

Large language models (2018-2023) demonstrated emergent capabilities in language understanding, reasoning, and generation [7, 8]. Models trained on massive text corpora exhibited in-context learning—adapting to new tasks from examples without parameter updates. SE applications proliferated: documentation generation, requirements analysis, code completion, and technical question answering.

Agentic AI (2023-2024) augmented LLMs with capabilities for autonomous action [9, 10]. Tool use enables LLMs to invoke external capabilities—calculators, search engines, APIs, code interpreters. Memory systems provide persistence beyond context windows. Planning approaches enable multi-step reasoning toward goals. Agentic AI can pursue objectives through extended sequences of reasoning and action.

Agentic swarms (2024-present) coordinate multiple agentic AI systems to address complex tasks [11, 12, 13]. Specialized agents collaborate, bringing diverse capabilities to problems exceeding individual agent capacity. Swarm architectures enable parallelism, perspective diversity, and collective intelligence. This generation offers the potential for comprehensive SE lifecycle support through human-AI collaboration.

2.2 From LLMs to Agentic Swarms

The transition from LLMs to agentic swarms involves progressive capability augmentation:

Tool use extends LLM capabilities beyond text generation. Agents can retrieve information from databases, execute code, query APIs, and interact with engineering tools. For SE, tool use enables agents to access requirements repositories, execute analyses, and update system models.

Memory systems provide persistent context:

- *Short-term memory* maintains recent interaction context
- *Long-term memory* accumulates knowledge across sessions
- *Episodic memory* records past interactions and outcomes

Memory enables agents to accumulate project knowledge, learn from experience, and maintain continuity across extended engineering activities.

Planning and reasoning approaches enable complex task decomposition:

- *Chain-of-thought* prompting elicits step-by-step reasoning [14]
- *ReAct* interleaves reasoning with actions [9]
- *Tree-of-thought* explores alternative reasoning paths [15]

These approaches enable agents to tackle multi-step engineering tasks requiring analysis, synthesis, and judgment.

Multi-agent coordination enables collective capability:

- *Role specialization* assigns distinct capabilities to different agents
- *Conversation* enables agents to discuss, debate, and refine
- *Shared memory* provides common context across agents
- *Orchestration* coordinates agent activities toward objectives

Multi-agent architectures enable swarms to address SE's inherently multi-disciplinary, collaborative nature.

2.3 Systems Engineering Process Framework

Systems engineering provides disciplined processes for developing complex systems across the lifecycle [1, 16]. ISO/IEC/IEEE 15288:2023 [2] defines the international standard framework, organizing lifecycle processes into technical processes, technical management processes, and organizational project-enabling processes.

The **technical processes** directly realize and support the system:

Process	ISO 15288 Ref	Description
Stakeholder Needs and Requirements Definition	6.4.1	Elicit and define stakeholder needs
System Requirements Definition	6.4.2	Transform needs into system requirements
Architecture Definition	6.4.3	Define system architecture
Design Definition	6.4.4	Detailed design of system elements
System Analysis	6.4.5	Trade studies, effectiveness analysis
Implementation	6.4.6	Realize system elements
Integration	6.4.7	Assemble system from elements
Verification	6.4.8	Confirm system meets requirements
Validation	6.4.9	Confirm system meets stakeholder needs
Transition	6.4.10	Establish in operational environment
Operation	6.4.11	Use system to deliver services
Maintenance	6.4.12	Sustain system capability
Disposal	6.4.13	End of life management

These processes apply across domains—aerospace, defense, healthcare, energy, transportation—making SE inherently domain-agnostic while requiring domain-specific knowledge for application.

The INCOSE Systems Engineering Handbook [1] and NASA Systems Engineering Handbook [16] provide guidance on applying these processes. Model-Based Systems Engineering (MBSE) establishes formal system models as the authoritative source of truth, creating structured representations that AI systems can process and analyze [17, 18].

2.4 The Case for Multi-Agent Approaches in SE

Several characteristics of systems engineering motivate multi-agent rather than single-agent AI approaches:

Multi-disciplinary nature. SE integrates contributions from diverse disciplines—mechanical, electrical, software, human factors, logistics. No single agent can embody deep expertise across all disciplines. Multi-agent architectures enable discipline specialization with cross-discipline coordination.

Collaborative practice. SE is inherently collaborative, involving teams of engineers, stakeholders, and organizations. Multi-agent systems mirror this collaborative structure, with agents representing different perspectives, roles, or organizational entities.

Artifact complexity. SE produces complex, interrelated artifacts—requirements, architectures, designs, test cases—with consistency and traceability requirements across thousands of elements. Agent swarms can achieve comprehensive coverage that individual analysis cannot.

Perspective diversity. Engineering quality benefits from multiple viewpoints examining artifacts. Agent swarms provide perspective diversity—different specializations, different analysis approaches—that surface issues single-point analysis misses.

Parallelism potential. Many SE activities can proceed concurrently. Agent swarms can exploit parallelism, analyzing multiple alternatives, checking multiple requirements, or evaluating multiple test cases simultaneously.

Lifecycle span. SE spans extended lifecycles from concept through disposal. Persistent agent systems can maintain continuity, accumulating knowledge and adapting to evolving system states across lifecycle phases.

These characteristics suggest that multi-agent architectures—agentic AI swarms—offer a natural fit for SE support, motivating the framework developed in subsequent sections.

Word count: ~1,100 words **Subsections:** 4 **Tables:** 1 **References cited:** [1]-[18]

Revision Notes

- [ ] Add figure showing AI evolution timeline
- [ ] Consider expanding MBSE connection
- [ ] Verify reference numbers align with final bibliography

Section 3: Related Work

Target length: ~1,000 words **Status:** Draft v0.1

---

3. Related Work

This section reviews foundational work in multi-agent systems, swarm intelligence, and AI applications in systems engineering, identifying the gap this paper addresses.

3.1 Multi-Agent Systems in Engineering

Multi-agent systems (MAS) research provides foundational concepts for coordinated AI systems. Wooldridge [19] defines agents as computer systems capable of autonomous action to meet design objectives, with properties including autonomy, social ability, reactivity, and proactivity. Weiss [20] and Ferber [21] established frameworks for multi-agent system design and coordination.

Engineering applications of MAS span decades. Jennings [22] demonstrated agent-based approaches to business process management. Shen et al. [23] surveyed agent-based manufacturing systems for reconfigurable production. Leitão [24] reviewed agent-based distributed manufacturing control. These applications established that agent coordination could address complex engineering challenges.

However, classical MAS relied on explicit knowledge engineering and symbolic reasoning, limiting adaptability and requiring substantial development effort. The emergence of LLM-based agents fundamentally changes the capability profile, enabling natural language interaction, broad knowledge access, and flexible reasoning without extensive custom development.

3.2 Swarm Intelligence

Swarm intelligence draws inspiration from collective behavior in biological systems—ant colonies, bird flocks, fish schools—where simple individuals following local rules produce sophisticated collective behavior [25, 26].

Ant Colony Optimization (ACO) models pheromone-based coordination for combinatorial optimization [27]. Artificial ants deposit virtual pheromone on solution components, guiding subsequent search toward promising solutions. ACO has been applied to engineering routing, scheduling, and design optimization.

Particle Swarm Optimization (PSO) models flocking behavior for continuous optimization [28]. Particles move through search spaces influenced by individual and collective best-known positions. PSO has found application in engineering parameter optimization and control system design.

Stigmergic coordination enables indirect communication through environmental modification [29]. Agents leave traces influencing other agents' behavior without direct message exchange, enabling scalable coordination with minimal overhead.

Swarm intelligence contributes key insights: collective intelligence can exceed individual capability; simple local rules produce complex global behavior; decentralized coordination achieves robust, scalable solutions. However, traditional swarm approaches address optimization rather than the reasoning-intensive tasks characteristic of systems engineering.

3.3 AI Applications in Systems Engineering

AI applications to SE have evolved with AI capabilities:

Requirements engineering has received substantial attention. Mund et al. [30] applied ML to requirements classification. Ferrari et al. [31] developed NLP approaches for requirements quality analysis. Dalpiaz et al. [32] surveyed AI for requirements engineering, identifying applications in elicitation, analysis, specification, and validation. Recent work applies LLMs to requirements generation and consistency checking [33, 34].

Architecture and design applications include AI-assisted trade studies, design space exploration, and pattern recommendation. Metzger and Pohl [35] reviewed AI in software product line engineering. Malavolta et al. [36] surveyed ML in software architecture. These applications typically employ single models rather than coordinated agent systems.

Verification and validation applications span test generation, coverage analysis, and defect prediction. Feldt et al. [37] surveyed AI in software testing. Durelli et al. [38] reviewed ML for test case prioritization. Automated test generation has achieved commercial deployment in software contexts.

Model-Based Systems Engineering (MBSE) creates structured representations amenable to AI analysis. Madni and Sievers [17] discussed AI and MBSE integration. Huldt and Stenius [18] surveyed AI applications in systems engineering, noting growing interest but limited mature implementations.

3.4 LLM-Based Multi-Agent Systems

The emergence of LLM-based agents has spawned frameworks for multi-agent coordination:

MetaGPT [39] implements multi-agent software development with role-based specialization (product manager, architect, engineer). It demonstrates role-based patterns but focuses on software rather than broader SE.

AutoGen [40] supports conversational agents with customizable interaction patterns. It enables flexible multi-agent configurations but provides limited SE-specific guidance.

CrewAI [41] provides role-based agent "crews" with defined collaboration patterns. It emphasizes task decomposition and agent specialization but lacks SE domain grounding.

ChatDev [42] simulates software company dynamics with communicating agents. Like MetaGPT, it addresses software development specifically.

These frameworks demonstrate multi-agent LLM capabilities but focus predominantly on software development rather than domain-agnostic systems engineering. None provides systematic mapping to SE process standards or addresses the full lifecycle.

3.5 Research Gap

Despite substantial work in constituent areas, a gap exists at their intersection:

Existing Work	Coverage	Gap
Classical MAS	Engineering applications	Not LLM-based; limited flexibility
Swarm intelligence	Optimization	Not reasoning-intensive SE tasks
AI in SE	Individual process areas	Not multi-agent; not comprehensive
LLM multi-agent	Software development	Not domain-agnostic SE; not process-mapped

This paper addresses the gap by presenting a framework for LLM-based agentic swarms systematically mapped to domain-agnostic systems engineering processes per ISO 15288. We synthesize insights from MAS coordination, swarm intelligence, and emerging LLM agent capabilities, applying them to the full SE lifecycle across application domains.

Word count: ~820 words **Subsections:** 5 **Tables:** 1 **References cited:** [17]-[42]

Revision Notes

- ☐ Expand LLM multi-agent section with more recent frameworks
- ☐ Add additional SE application citations
- ☐ Consider adding comparison figure

Section 4: Architectural Framework

Target length: ~2,000 words **Status:** Draft v0.1

---

4. Architectural Framework

This section presents an architectural framework for agentic AI swarms in systems engineering, addressing agent architectures, SE-specific specialization patterns, coordination mechanisms, and swarm topologies.

4.1 Agent Architecture Taxonomy

Agent architectures can be classified along several dimensions relevant to SE applications:

Reactive architectures employ direct stimulus-response mappings without explicit world models [43]. Fast and robust, they suit monitoring and routine response but lack deliberative capability for complex reasoning.

Deliberative architectures maintain explicit world models and reason about goals and plans. The Belief-Desire-Intention (BDI) model [44, 45] represents agents with beliefs about the world, desires (goals) to achieve, and intentions (committed plans). BDI suits structured SE tasks with explicit objectives.

Hybrid architectures combine reactive and deliberative elements in layered designs [46]. Reactive layers handle routine situations; deliberative layers address complex cases. Hybrid approaches balance responsiveness with reasoning capability.

LLM-based architectures leverage pre-trained foundation models for flexible reasoning. Key patterns include:

- **ReAct** [9]: Interleaves reasoning traces with actions
- **Tool-augmented**: Extends LLM capability through tool invocation [10]
- **Memory-augmented**: Provides persistent context beyond context windows
- **Retrieval-augmented (RAG)**: Grounds responses in retrieved documents [47]

Multi-agent LLM architectures coordinate multiple LLM-based agents:

- **Role-based**: Agents assume specialized roles with distinct capabilities
- **Debate-based**: Agents argue positions, refining through discourse
- **Hierarchical**: Manager agents coordinate worker agents
- **Peer**: Agents collaborate as equals with complementary capabilities

For SE applications, we recommend **hybrid LLM-based architectures** combining:

- LLM foundation for flexible reasoning and natural language interaction
- Tool integration for accessing SE tools and data
- RAG for grounding in domain knowledge and standards
- Memory for accumulating project knowledge
- Multi-agent coordination for comprehensive coverage

4.2 Agent Specialization for SE Disciplines

Effective SE swarms require agent specialization aligned with engineering disciplines and roles. We define a taxonomy of agent types:

Core SE agents:

Agent Type	Role	Key Capabilities
Requirements Engineer	Requirements lifecycle	Elicitation, analysis, specification, traceability
Systems Architect	System structure	Architecture patterns, views, trade-offs
Integration Engineer	System assembly	Interface management, integration planning
V&V Engineer	Verification/validation	Test planning, execution, coverage analysis

Trade Study Analyst	Decision analysis	Multi-criteria analysis, sensitivity, optimization

Discipline specialist agents:

Agent Type	Domain	Key Capabilities
Mechanical Engineer	Structures, mechanisms	Structural analysis, kinematics, thermal
Electrical Engineer	Power, signals	Circuit analysis, EMI/EMC, power budgets
Software Engineer	Software elements	Code analysis, architecture, testing
Human Factors Engineer	Human-system interaction	Usability, workload, error analysis
Reliability Engineer	Dependability	FMEA, fault trees, availability analysis
Safety Engineer	Hazard management	Hazard analysis, safety cases

Coordination agents:

Agent Type	Role	Key Capabilities
Chief Engineer	Technical leadership	Direction, decision escalation, integration
IPT Lead	Team coordination	Work coordination, status tracking
Configuration Manager	Artifact management	Version control, baseline management
Quality Assurer	Process compliance	Standards adherence, audit support

Agent specialization is achieved through:

- **System prompts** defining role, responsibilities, and constraints
- **Tool access** providing role-appropriate capabilities
- **Knowledge bases** grounding agents in discipline-specific content
- **Interaction patterns** defining how agents engage with others

4.3 Coordination Mechanisms

Effective swarms require coordination mechanisms managing dependencies and achieving coherent collective behavior. We categorize mechanisms by approach:

Communication-based coordination:

Message passing provides direct agent-to-agent communication. Agents exchange information, requests, and responses through structured or natural language messages. Suitable for targeted information sharing but can create communication overhead.

Blackboard systems coordinate through shared data structures [48]. Agents post contributions to a shared space; others observe and respond. For SE, shared system models can serve as blackboards—agents contribute analyses while others incorporate findings.

Publish-subscribe enables event-driven coordination. Agents publish events (requirement changed, design updated); interested agents subscribe and respond. Supports loose coupling and change propagation.

Organization-based coordination:

Hierarchical structures arrange agents in authority relationships [49]. Higher-level agents decompose tasks, assign work, and integrate results. Mirrors SE organization (chief engineer → IPT leads → discipline engineers). Provides clear accountability but can create bottlenecks.

Market-based coordination uses economic mechanisms [50]. The Contract Net Protocol has agents announce tasks and solicit bids; work is awarded based on capability and availability. Enables dynamic allocation without central planning.

Team-based coordination groups agents pursuing shared goals [51]. Agents commit to collective objectives and coordinate execution. Natural fit for SE project teams.

Emergent coordination:

Stigmergy coordinates through environmental modification [29]. Agents leave traces (annotations, markers) influencing other agents without direct communication. Enables scalable coordination for exploration tasks.

Self-organization produces coordinated structures from local interactions. Agents following local rules generate global patterns. Useful for optimization but less predictable for precision requirements.

Recommended approach for SE: Hybrid coordination combining:

- Hierarchical structure for task decomposition and accountability
- Shared memory (system model as blackboard) for state coordination
- Message passing for targeted collaboration
- Event-driven updates for change propagation

4.4 Swarm Topologies

Swarm topology defines the communication and coordination structure among agents:

Hierarchical topology arranges agents in tree structures. A chief engineer agent coordinates IPT lead agents, who coordinate discipline specialist agents. Information flows up (status, issues) and down (direction, decisions). Pros: Clear accountability, manageable coordination. Cons: Potential bottlenecks, limited cross-branch communication.

Mesh topology connects agents peer-to-peer. Any agent can communicate with any other. Information flows freely based on relevance. Pros: Flexible, no bottlenecks. Cons: Coordination overhead scales with agent count, potential for inconsistency.

Hybrid topology combines hierarchical and mesh elements. Hierarchical structure for task management; mesh connections for discipline coordination. An architect agent and mechanical agent can communicate directly while both report through hierarchical channels.

Adaptive topology adjusts structure based on context. During architecture definition, mesh connections among discipline agents enable trade exploration. During integration, hierarchical control ensures coordinated assembly. Topology adapts to lifecycle phase and task characteristics.

Topology selection criteria:

Factor	Favors Hierarchical	Favors Mesh
Team size	Larger (>10 agents)	Smaller (<10 agents)
Task coupling	Loosely coupled	Tightly coupled
Decision authority	Centralized	Distributed
Communication overhead tolerance	Lower	Higher

Predictability requirement	Higher	Lower

4.5 Emergent Behavior Considerations

Multi-agent systems can exhibit emergent behaviors—collective outcomes not explicitly programmed into individual agents. Emergence presents both opportunities and risks for SE applications.

Beneficial emergence:

- Comprehensive coverage through parallel exploration
- Error detection through perspective diversity
- Creative solutions through agent interaction
- Collective intelligence exceeding individual capability

Problematic emergence:

- Reinforced errors through echo chambers
- Unexpected interactions producing invalid conclusions
- Runaway processes consuming resources without progress
- Inconsistent outputs from uncoordinated contributions

Managing emergence in SE contexts:

Monitoring mechanisms detect divergence from expected behavior. Metrics tracking agent activity, output consistency, and resource consumption identify anomalies.

Intervention capabilities allow human operators to pause, redirect, or terminate swarm activities. Clear escalation triggers define when human attention is required.

Bounds constrain swarm behavior within acceptable limits. Constitutional approaches encode inviolable constraints. Verification gates check outputs before acceptance.

Transparency enables understanding of swarm behavior. Explanation mechanisms trace conclusions to reasoning. Audit trails record agent contributions.

For safety-critical SE applications, emergence must be bounded and monitored. The framework should enable beneficial emergence (comprehensive analysis, diverse perspectives) while preventing harmful emergence (invalid conclusions, resource exhaustion).

Word count: ~1,350 words **Subsections:** 5 **Tables:** 5 **References cited:** [9], [10], [29], [43]-[51]

Revision Notes

- ☐ Add architecture taxonomy figure
- ☐ Add topology diagrams
- ☐ Expand coordination mechanism trade-offs
- ☐ Add specific examples of emergence in SE contexts

Section V: Mapping to Systems Engineering Process Areas

Target length: ~2,500 words **Status:** Draft v0.1 **Format:** IEEE Systems Journal

V. MAPPING TO SYSTEMS ENGINEERING PROCESS AREAS

This section presents the core contribution: systematic mapping of agentic AI swarm capabilities to ISO/IEC/IEEE 15288 technical processes. We demonstrate how swarm architectures can support each process area, identify applicable agent patterns, and assess current maturity.

A. Framework Overview

We map swarm capabilities to the thirteen technical processes defined in ISO/IEC/IEEE 15288:2023 [2]. For each process area, we identify:

1. Process objectives and key activities
2. Swarm support opportunities
3. Recommended agent configurations
4. Current maturity level

Maturity is assessed on a four-level scale: *Conceptual* (theoretical feasibility), *Research* (prototype demonstrations), *Pilot* (limited industrial trials), *Production* (operational deployment).

B. Stakeholder Needs and Requirements Definition

Process objectives: Define stakeholder needs and transform them into stakeholder requirements that can guide system definition.

Swarm support opportunities:

- *Multi-stakeholder elicitation:* Agents represent different stakeholder perspectives (operators, maintainers, regulators), ensuring comprehensive needs capture
- *Needs analysis:* Parallel analysis of stated needs to identify implicit requirements, conflicts, and gaps
- *ConOps development:* Agents collaborate to develop operational scenarios from multiple viewpoints
- *Validation facilitation:* Agents formulate questions that surface unstated assumptions

Agent configuration: Deploy stakeholder-representative agents (operator agent, maintainer agent, regulator agent) coordinated by a facilitator agent. Each representative agent embodies its stakeholder's perspective, priorities, and constraints. The facilitator manages elicitation sessions, identifies conflicts, and synthesizes findings.

Maturity: Research. LLM-based requirements elicitation assistants exist; multi-agent stakeholder representation remains experimental.

C. System Requirements Definition

Process objectives: Transform stakeholder requirements into system requirements that specify what the system must do and its quality characteristics.

Swarm support opportunities:

- *Requirements derivation:* Systematic transformation of stakeholder requirements into system requirements with traceability
- *Completeness checking:* Comprehensive analysis against domain templates and precedent systems
- *Consistency checking:* Parallel verification of requirement compatibility across the requirement set
- *Allocation support:* Analysis of requirement allocation to system elements

Agent configuration: Requirements analyst agent performs derivation and specification. Discipline validator agents (thermal, structural, software) check technical feasibility within their domains. Consistency checker agent maintains cross-requirement coherence. Traceability agent maintains requirement relationships.

Maturity: Pilot. Requirements analysis tools incorporating AI are entering industrial use; multi-agent comprehensive checking remains limited.

D. Architecture Definition

Process objectives: Generate system architecture alternatives, select among alternatives, and develop architectural views addressing stakeholder concerns.

Swarm support opportunities:

- *Architecture pattern exploration:* Agents propose and evaluate alternative architecture patterns
- *View generation:* Specialized agents generate architecture views per ISO 42010 [52]
- *Interface identification:* Analysis of element interactions to define interfaces
- *Trade-off analysis:* Multi-criteria evaluation of architecture alternatives

Agent configuration: Architecture lead agent coordinates exploration. Viewpoint agents generate specific views (functional, physical, behavioral). Interface agent identifies and specifies interfaces. Trade analyst agent evaluates alternatives against criteria.

Maturity: Research. Architecture exploration tools exist; multi-agent coordinated architecture development is early-stage.

E. Design Definition and System Analysis

Process objectives: Provide sufficient detailed design to enable implementation; analyze system properties and behavior.

Swarm support opportunities:

- *Design space exploration:* Parallel evaluation of design alternatives across parameters
- *Trade study execution:* Multi-objective optimization with diverse evaluation perspectives
- *Sensitivity analysis:* Systematic exploration of design parameter impacts
- *Analysis integration:* Coordination of discipline-specific analyses

Agent configuration: Trade study coordinator agent manages studies. Objective-specific agents evaluate alternatives against different criteria (performance, cost, risk, schedule). Discipline analyst agents perform domain analyses. Integration agent synthesizes findings.

Maturity: Research. Design optimization tools exist; coordinated multi-agent trade studies are experimental.

F. Integration

Process objectives: Assemble system elements into a complete system that satisfies system requirements.

Swarm support opportunities:

- *Integration planning:* Analysis of dependencies to optimize build sequence
- *Interface verification:* Systematic checking of interface compatibility
- *Integration risk identification:* Analysis of potential integration issues
- *Progress tracking:* Monitoring integration status across elements

Agent configuration: Integration coordinator agent manages overall integration. Element-responsible agents track status of their elements. Interface verification agents check compatibility at integration points. Risk analyst agent identifies potential issues.

Maturity: Conceptual. Integration support tools exist; multi-agent integration coordination is theoretical.

G. Verification

Process objectives: Provide objective evidence that the system fulfills specified requirements.

Swarm support opportunities:

- *Test case generation:* Systematic generation covering requirements
- *Coverage analysis:* Assessment of verification completeness
- *Test procedure development:* Detailed procedure specification
- *Result analysis:* Interpretation of verification outcomes

Agent configuration: V&V lead agent coordinates verification. Requirements-to-test agents generate test cases for assigned requirements. Coverage analyst agent assesses completeness. Test execution agents (for automated tests) execute and report. Results analyst agent interprets outcomes.

Maturity: Pilot. AI-assisted test generation has commercial deployment; multi-agent comprehensive verification is emerging.

H. Validation

Process objectives: Provide objective evidence that the system satisfies stakeholder requirements and achieves intended use.

Swarm support opportunities:

- *Validation scenario development:* Generation of scenarios exercising system capabilities
- *Stakeholder criteria mapping:* Tracing validation activities to stakeholder needs
- *Operational context analysis:* Assessment of system behavior in operational contexts
- *Acceptance criteria verification:* Checking against stakeholder acceptance conditions

Agent configuration: Validation coordinator agent manages activities. Scenario generation agents develop validation scenarios. Stakeholder-perspective agents assess from different viewpoints. Operational analyst agents evaluate operational performance.

Maturity: Research. Validation scenario generation is emerging; comprehensive multi-agent validation is early-stage.

I. Transition, Operation, and Maintenance

Process objectives: Establish system capability in operational environment; operate system; sustain system capability.

Swarm support opportunities:

- *Transition planning:* Development of deployment procedures and training materials
- *Operational procedure generation:* Creation of operating procedures
- *Anomaly investigation:* Analysis of operational issues and failures
- *Sustainment analysis:* Assessment of maintenance needs and refresh opportunities

Agent configuration: Operations analyst agent addresses operational procedures. Maintenance engineer agent analyzes sustainment. Logistics agent addresses support requirements. Anomaly investigator agent analyzes issues.

Maturity: Research. Operations support exists in specific domains; comprehensive lifecycle support is experimental.

J. Maturity Assessment Summary

Table I summarizes maturity across process areas.

TABLE I: Process Area Maturity Assessment

Process Area	Maturity	Evidence Base
Stakeholder Needs Definition	Research	Prototype demonstrations
System Requirements Definition	Pilot	Limited industrial trials
Architecture Definition	Research	Academic prototypes
Design Definition / System Analysis	Research	Optimization tools
Integration	Conceptual	Theoretical frameworks
Verification	Pilot	Commercial tools emerging
Validation	Research	Early prototypes
Transition / Operations / Maintenance	Research	Domain-specific tools

Maturity concentrates in requirements and verification—processes with clearer task definitions and evaluation criteria. Architecture, design, and integration—requiring more judgment and context—remain less mature. Lifecycle processes (transition, operations, maintenance) have domain-specific applications but lack general SE frameworks.

K. Cross-Process Coordination

Effective SE requires coordination across process areas. Swarm architectures enable:

Concurrent execution: Requirements analysis, architecture exploration, and verification planning can proceed in parallel, with agents maintaining consistency through shared models.

Traceability maintenance: Dedicated agents maintain traceability links across process boundaries (requirements to architecture to design to verification).

Change propagation: When requirements change, architecture and verification agents receive notification and assess impacts in their domains.

Lifecycle continuity: Agent systems can maintain knowledge from development through operations, enabling feedback from operational experience to inform future development.

The swarm approach naturally supports the iterative, concurrent nature of modern SE practice, in contrast to sequential process models that constrain parallelism.

Word count: ~1,280 words **Subsections:** 11 **Tables:** 1 **References cited:** [2], [52]

Revision Notes

- ☐ Add process mapping figure (Vee diagram or similar)
- ☐ Expand maturity evidence with specific citations
- ☐ Consider adding example swarm configurations
- ☐ Verify alignment with ISO 15288:2023 terminology

Section VI: Transformation Roadmap

Target length: ~1,000 words **Status:** Draft v0.1 **Format:** IEEE Systems Journal

---

VI. TRANSFORMATION ROADMAP

The transition from current SE practice to the vision of collaborative human-AI teams requires a phased transformation. This section charts the path, identifying milestones and enabling factors.

A. Near-Term (2025-2027): Task Augmentation

The transformation begins with AI augmentation of discrete engineering tasks. Engineers employ AI assistants for specific activities while maintaining direct oversight of all outputs.

Characteristic applications:

- Documentation assistance: AI generates initial drafts for human refinement
- Requirements analysis: AI identifies potential issues for human resolution
- Design exploration: AI-assisted parametric studies expand alternative evaluation
- Code generation: AI produces code from specifications, subject to human review

Human role: In the loop for all decisions. AI tools function as sophisticated assistants. Trust is limited and verified—human oversight catches AI errors.

Organizational adaptation: Modest. Existing processes accommodate AI tools. Training focuses on effective tool use. Governance treats AI outputs as engineer-produced artifacts subject to standard review.

Milestone: AI-generated artifacts enter production use in non-critical applications.

B. Mid-Term (2027-2030): Coordinated Multi-Agent Support

As AI capabilities mature and organizational experience accumulates, the transformation progresses to coordinated multi-agent support.

Characteristic applications:

- Cross-discipline analysis: Swarms of discipline-specialist agents collaborate on integrated analyses
- Continuous verification: Agent swarms maintain ongoing verification, alerting engineers to issues
- Trade study automation: Multi-objective optimization swarms explore trade spaces comprehensively
- Interface management: Interface agents monitor designs across boundaries, identifying conflicts

Human role: Oversight and exception handling. Engineers define objectives and constraints; swarms execute analyses; humans review results and intervene when needed. Trust expands as track records accumulate.

Organizational adaptation: More substantial. Processes evolve to accommodate continuous AI analysis alongside milestone reviews. New roles emerge: swarm configuration specialists, AI output auditors. Governance addresses agent authority and output attribution.

Milestone: AI swarms participate in certified system development programs.

C. Long-Term (2030-2035): Collaborative Human-AI Teams

The transformation culminates in collaborative human-AI teams where AI swarms function as team members with defined roles.

Characteristic applications:

- AI team members: Agent swarms occupy defined engineering roles, participating in reviews and discussions

- Continuous lifecycle support: Swarms maintain awareness throughout lifecycle
- Institutional memory: AI systems preserve and apply organizational knowledge
- Emergent capability: Human-AI collaboration produces insights neither achieves alone

Human role: Judgment, creativity, and accountability. Engineers set direction, make value-laden decisions, engage stakeholders, and bear responsibility. They conduct the orchestra rather than play every instrument.

Organizational adaptation: Profound. Organizations restructure around human-AI teams. Career paths and performance metrics reflect the new paradigm. Governance addresses AI participation in engineering decisions with clear accountability.

Milestone: Human-AI teams become standard practice in SE organizations.

D. Enabling Technologies and Organizational Readiness

Transformation pace depends on enabler maturity across multiple dimensions:

AI capability enablers:

- *Improved reasoning:* Multi-step reasoning, causal inference, planning
- *Domain grounding:* Reliable access to authoritative domain knowledge
- *Coordination at scale:* Effective coordination among many agents
- *Explainability:* Ability to explain reasoning and trace conclusions

Infrastructure enablers:

- *Digital thread:* Continuous, traceable information across lifecycle
- *Model-based representations:* MBSE maturity with formal system models
- *Secure execution environments:* Auditable AI operations
- *Computational resources:* Capacity for concurrent agent operation

Organizational enablers:

- *Workforce skills:* Engineers trained for AI collaboration
- *Process adaptation:* Workflows suited to human-AI teaming
- *Governance frameworks:* Clear structures for AI involvement
- *Trust calibration:* Appropriately calibrated reliance on AI

Table II summarizes enabler readiness assessment.

TABLE II: Enabler Readiness Assessment

Enabler Category	Current State	Required State	Gap
AI reasoning	Emerging	Robust	Significant
Domain grounding	Limited	Comprehensive	Moderate
Multi-agent coordination	Early	Production-ready	Significant
Digital thread	Partial adoption	Universal	Moderate
MBSE maturity	Growing	Widespread	Moderate
Workforce skills	Minimal	Ubiquitous	Significant
Governance frameworks	Ad hoc	Established	Significant

Progress is needed across all enablers. The transformation will advance as the slowest-moving enablers allow. Organizations advancing digital engineering maturity and MBSE adoption are positioning themselves for AI swarm integration.

Word count: ~720 words **Subsections:** 4 **Tables:** 1

Revision Notes

- ☐ Add transformation timeline figure
- ☐ Cite specific initiatives addressing enabler gaps
- ☐ Consider adding domain-specific transformation considerations

Section VII: Domain Applications

Target length: ~800 words **Status:** Draft v0.1 **Format:** IEEE Systems Journal

---

VII. DOMAIN APPLICATIONS

This section illustrates framework applicability across engineering domains, validating domain-agnostic generality while acknowledging domain-specific considerations.

A. Aerospace: Satellite System Development

Satellite development exemplifies SE complexity: multi-disciplinary integration, stringent verification requirements, and extended operational lifecycles.

Swarm application: A satellite development swarm might include agents specializing in orbital mechanics, power systems, thermal management, communications, and software. Architecture exploration swarms evaluate constellation configurations. Verification swarms coordinate testing across RF, environmental, and software domains.

Domain considerations: Space-qualified heritage components constrain design space. Radiation effects require specialized analysis. Long development cycles (5-10 years) benefit from persistent agent memory. Safety-critical missions demand bounded emergence and auditable contributions.

Illustrative scenario: During preliminary design, a swarm of discipline specialists evaluates 500+ architecture variants against mission requirements. The thermal agent identifies configurations exceeding allowable temperatures; the power agent flags insufficient solar array sizing; the communications agent assesses link margin. Human engineers review the Pareto-optimal subset rather than the full trade space.

B. Defense: Weapon System Acquisition

Defense acquisition involves complex stakeholder landscapes, regulatory compliance, and extended lifecycles spanning decades.

Swarm application: Acquisition swarms could coordinate requirements analysis across multiple stakeholders (warfighter, maintainer, acquisition authority). Compliance agents verify adherence to MIL-STD standards. Cost estimation agents support should-cost analysis. Sustainment agents assess lifecycle implications.

Domain considerations: Security classification constrains data access and tool deployment. DoD acquisition regulations impose specific process requirements. International Traffic in Arms Regulations (ITAR) affect information sharing. Adversarial considerations require analysis of system vulnerabilities.

Illustrative scenario: During source selection, analysis swarms evaluate proposals against technical requirements while compliance swarms verify regulatory adherence. Human evaluators focus judgment on discriminating factors rather than exhaustive compliance checking.

C. Healthcare: Medical Device Development

Medical device development operates under rigorous regulatory oversight with direct implications for patient safety.

Swarm application: Regulatory compliance swarms verify FDA requirements (21 CFR Part 820). Human factors swarms analyze use-related risks per IEC 62366. Software swarms address IEC 62304 for medical device software. Risk management swarms coordinate ISO 14971 analyses.

Domain considerations: Regulatory approval requires comprehensive documentation and traceability. Patient safety demands conservative emergence management. Clinical validation requires human-centered assessment. Post-market surveillance creates ongoing monitoring requirements.

Illustrative scenario: Risk management swarms maintain hazard analyses as design evolves, automatically assessing impacts of design changes on identified hazards. Human safety engineers review flagged changes rather than re-analyzing the complete hazard set.

D. Energy: Power Grid Modernization

Grid modernization integrates legacy infrastructure with new technologies across continental scales.

Swarm application: Integration swarms analyze interoperability across legacy and modern components. Resilience swarms assess vulnerability to disruption. Renewable integration swarms evaluate variable generation impacts. Cybersecurity swarms analyze attack surfaces.

Domain considerations: Grid scale creates massive state spaces. Real-time operation constrains analysis latency. Regulatory frameworks vary across jurisdictions. Physical infrastructure creates long replacement cycles.

Illustrative scenario: Planning swarms evaluate grid modernization alternatives, assessing reliability impacts, renewable hosting capacity, and cybersecurity implications. Human planners review recommendations with supporting analyses rather than conducting analyses from scratch.

E. Cross-Domain Patterns

Despite domain differences, common patterns emerge:

Regulatory compliance support. Every domain operates under regulatory frameworks. Swarms can maintain compliance checking against relevant standards, freeing human engineers to focus on technical content.

Multi-stakeholder coordination. Complex systems serve multiple stakeholders with potentially conflicting needs. Swarms representing different perspectives can surface conflicts early.

Trade study amplification. Domains involve trade-offs among competing objectives. Swarms can explore trade spaces more comprehensively than manual analysis permits.

Verification comprehensiveness. All domains require verification. Swarms can achieve coverage that resource constraints preclude manually.

Lifecycle continuity. Long-lived systems benefit from persistent knowledge. Agent systems can maintain continuity across personnel changes and lifecycle phases.

Table III summarizes domain-specific considerations.

TABLE III: Domain-Specific Considerations

Domain	Key Regulation	Primary Constraint	Swarm Benefit
Aerospace	NASA/ESA standards	Heritage/radiation	Architecture exploration
Defense	MIL-STDs, acquisition regs	Classification/ITAR	Compliance verification
Healthcare	FDA 21 CFR, IEC 62366	Patient safety	Risk management
Energy	NERC/FERC standards	Scale/real-time	Integration analysis

The framework's domain-agnostic core—agent architectures, coordination mechanisms, process mapping—applies across domains, while domain-specific knowledge bases, regulatory agents, and constraint sets provide specialization.

Word count: ~750 words **Subsections:** 5 **Tables:** 1

Revision Notes

- ☐ Add specific regulatory citations
- ☐ Consider expanding one domain as detailed case study
- ☐ Verify technical accuracy of domain examples

Section VIII: Challenges and Research Agenda

Target length: ~1,500 words **Status:** Draft v0.1 **Format:** IEEE Systems Journal

---

VIII. CHALLENGES AND RESEARCH AGENDA

Despite significant potential, substantial challenges impede practical adoption of agentic AI swarms for systems engineering. This section categorizes challenges and proposes prioritized research directions.

A. Technical Challenges

Scalability. Coordination overhead grows with agent count—potentially faster than capability gains. Communication volume, conflict frequency, and convergence time may scale poorly. Research is needed on coordination mechanisms that scale efficiently, theoretical models predicting overhead, and practical limits of swarm size.

Reliability. LLM-based agents exhibit unpredictable failures including hallucinations, reasoning errors, and instruction drift. Multi-agent systems can amplify failures through error propagation. Research priorities include failure mode characterization, architectural patterns improving robustness, and reliability metrics for SE applications.

Domain knowledge integration. Effective SE support requires deep domain knowledge that current LLMs may represent incorrectly. Research directions include retrieval approaches for authoritative sources, constraint encoding methods, hybrid architectures combining LLM reasoning with physics-based analysis, and domain-specific evaluation benchmarks.

Consistency maintenance. Multi-agent artifact generation must maintain consistency. Concurrent modifications can introduce conflicts. Research needs include consistency models for SE artifacts, conflict detection and resolution mechanisms, and coordination protocols ensuring coherent outputs.

B. Integration Challenges

Tool integration. Connecting agent systems with SE tools requires substantial effort. Proprietary APIs, data formats, and access models complicate integration. Research should address integration reference architectures, abstraction layers, and standards for agent-tool interaction.

Process integration. Inserting AI agents into established processes raises workflow questions. Research needs include process adaptation patterns, workflow models for human-AI collaboration, and guidelines for hybrid traditional/AI-augmented processes.

Data integration. Agents require access to engineering data distributed across systems with varied formats. Research directions include engineering data accessibility frameworks and data quality requirements for agent consumption.

C. Human Factors Challenges

Trust calibration. Engineers must develop appropriate trust—neither over-reliance nor excessive skepticism. Research priorities include trust development models, calibration mechanisms, and approaches for maintaining trust as capabilities evolve.

Oversight effectiveness. Human oversight becomes challenging as agent activity increases. Research needs include oversight models for multi-agent systems, attention management approaches, and interfaces supporting efficient review.

Skill evolution. Working with agent systems requires new skills. Research should address required competency models, training approaches, and curricula for engineering education.

D. Organizational Challenges

Governance frameworks. Organizations need frameworks governing AI involvement. Research priorities include governance model development, accountability frameworks, and organizational structures supporting appropriate AI involvement.

Certification implications. In regulated industries, AI involvement raises certification questions. Research needs include certification framework development, evidence requirements, and regulatory pathway exploration.

Economic justification. Adoption requires demonstrated value. Research should develop value frameworks, benefit measurement approaches, and return-on-investment models.

E. Prioritized Research Directions

Based on challenge analysis, we prioritize research directions by time horizon:

Near-term (1-3 years):

1. *Benchmark development:* Standardized task suites for SE process areas enabling cross-study comparison
2. *Reliability characterization:* Failure mode taxonomies, detection methods, robustness patterns
3. *Domain grounding:* Retrieval approaches for SE knowledge, constraint encoding, hybrid architectures
4. *Tool integration patterns:* Reference architectures for common SE tools

Mid-term (3-7 years): 5. *Coordination at scale:* Theoretical models, hierarchical architectures, predictable emergence 6. *Human-AI teaming:* Performance models, interface design, training approaches 7. *Evaluation methodology:* Coordination quality metrics, longitudinal methods, team effectiveness measures 8. *Governance frameworks:* Accountability models, audit mechanisms, certification approaches

Long-term (7+ years): 9. *Collective engineering intelligence:* Capabilities exceeding either humans or AI alone 10. *Self-improving systems:* Learning from deployment, safe self-modification 11. *Automation boundaries:* Principled task allocation, evolving boundaries

Table IV summarizes challenge prioritization.

TABLE IV: Challenge Prioritization Matrix

Challenge	Severity	Tractability	Priority
Reliability	High	Moderate	Critical
Domain knowledge	High	Moderate	Critical
Trust calibration	High	Moderate	Critical
Governance	High	Low	High
Scalability	Moderate	Moderate	High
Tool integration	Moderate	High	High
Certification	High	Low	High
Oversight	Moderate	Moderate	Medium
Skills evolution	Moderate	High	Medium

F. Call to Action by Stakeholder

Realizing the research agenda requires action across the SE community:

For practitioners:

- Develop AI collaboration skills through training and experimentation
- Provide feedback on AI tool capabilities and limitations
- Participate in pilot programs validating swarm approaches
- Document and share lessons learned

For researchers:

- Address prioritized challenges through rigorous investigation
- Develop benchmarks enabling cumulative progress
- Pursue cross-disciplinary collaboration (AI, SE, human factors)
- Engage practitioners in validation

For organizations:

- Invest in digital infrastructure (digital thread, MBSE) as AI foundation
- Evolve processes and governance for human-AI collaboration
- Build organizational AI literacy
- Conduct controlled pilots

For standards bodies:

- Develop guidance for AI involvement in engineering
- Address certification implications
- Enable interoperability through standards
- Facilitate community learning through shared resources

Word count: ~880 words **Subsections:** 6 **Tables:** 1

Revision Notes

- ☐ Add specific research questions for each direction
- ☐ Cite existing work addressing challenges
- ☐ Consider adding funding landscape discussion

Section IX: Discussion

Target length: ~600 words **Status:** Draft v0.1 **Format:** IEEE Systems Journal

---

IX. DISCUSSION

A. Implications for SE Practice

The framework presented suggests several implications for systems engineering practice:

Augmentation, not replacement. Agentic AI swarms are positioned to augment human systems engineers, not replace them. The framework identifies tasks where AI swarms add value (parallel analysis, comprehensive coverage, artifact generation) while recognizing activities requiring human judgment (stakeholder engagement, creative insight, accountability). Organizations should approach AI swarms as force multipliers rather than substitutes for engineering expertise.

Process adaptation. Effective use of AI swarms may require adaptations to traditional SE processes. Sequential review gates designed for human-paced work may not suit rapid AI-generated analysis. Organizations may need new review approaches, quality criteria, and approval workflows suited to human-AI collaborative outputs.

Skill evolution. Systems engineers working with AI swarms need new competencies: formulating effective prompts, evaluating AI outputs, managing swarm configurations, and intervening appropriately. Engineering education and professional development must evolve to prepare practitioners for collaborative practice.

B. Implications for SE Research

This work identifies several research opportunities:

Empirical validation. The framework is conceptual; empirical studies are needed to validate effectiveness claims. Controlled experiments comparing swarm-assisted versus traditional approaches, field studies in operational environments, and longitudinal tracking would strengthen the evidence base.

Formalization. More rigorous formal treatment—using established formalisms from multi-agent systems, coordination theory, and systems engineering ontologies—would enable precise reasoning about swarm properties and support formal verification of swarm behavior.

Tool development. Practical adoption requires tools supporting swarm configuration, execution, monitoring, and analysis. Research into tool architectures, user interfaces, and integration with existing SE tool chains would accelerate adoption.

C. Relationship to Digital Engineering and MBSE

Agentic AI swarms emerge as digital engineering transformation advances across SE organizations. Model-Based Systems Engineering (MBSE) establishes authoritative system models as the primary communication means among disciplines [17, 18]. AI swarms could participate in this model-centric ecosystem:

- Swarms consume system models, analyze properties, and generate derived artifacts
- Shared system models serve as coordination substrate for swarm state
- Model-based traceability supports governance and audit requirements

Organizations advancing MBSE maturity are positioning themselves for AI swarm adoption. Conversely, AI capabilities create new value from MBSE investments. The synergy suggests coordinated advancement of digital engineering and AI adoption strategies.

The Department of Defense Digital Engineering Strategy [53], NASA's digital transformation initiatives, and industry digital engineering programs create infrastructure that AI swarms can leverage. Organizations should view these initiatives as complementary—digital foundations enabling AI augmentation.

D. Limitations

This work has several limitations:

1. **Conceptual framework.** The mapping and analysis are conceptual rather than empirically validated. Actual effectiveness requires demonstration through controlled studies and field deployment.

2. **Technology maturity.** Agentic AI swarms are early-stage technology. Capabilities and limitations evolve rapidly; assessments may require revision as technology matures.

3. **Domain generalization.** While intended as domain-agnostic, examples and analysis reflect experience primarily in aerospace and defense. Applicability to other domains requires validation.

4. **Single perspective.** This paper presents one architectural approach. Alternative frameworks may prove more effective for specific contexts.

5. **Limited empirical grounding.** The maturity assessments are based on literature review rather than systematic empirical survey.

These limitations should inform interpretation and suggest directions for follow-on research.

Word count: ~560 words **Subsections:** 4

Revision Notes

- [ ] Add specific MBSE tool references
- [ ] Consider expanding digital engineering discussion
- [ ] May trim if total length exceeds target

Section X: Conclusion

Target length: ~400 words **Status:** Draft v0.1 **Format:** IEEE Systems Journal

---

X. CONCLUSION

This paper has presented a framework for applying agentic AI swarms to systems engineering—a multi-agent architectural approach that addresses the limitations of single-agent AI systems when applied to complex, multi-disciplinary engineering challenges.

Summary of Contributions

We traced the evolution of AI capabilities through six tiers—from expert systems through machine learning, deep learning, and large language models to agentic AI and agentic swarms—establishing the technological foundation for comprehensive SE support. We presented an architectural framework comprising agent architecture taxonomies, SE-specific specialization patterns, coordination mechanisms, and swarm topologies.

The core contribution systematically mapped swarm capabilities to ISO/IEC/IEEE 15288 technical process areas, demonstrating applicability across stakeholder needs definition, requirements engineering, architecture definition, design, integration, verification, validation, and lifecycle support. Maturity assessment revealed concentrated development in requirements and verification processes, with substantial opportunities in other lifecycle phases.

We charted a transformation roadmap from near-term task augmentation through mid-term coordinated support to long-term collaborative human-AI teams, identifying enabling technologies and organizational readiness factors. Domain examples in aerospace, defense, healthcare, and energy validated framework generality while acknowledging domain-specific considerations.

We categorized challenges—technical, integration, human factors, and organizational—and proposed a prioritized research agenda with calls to action for practitioners, researchers, organizations, and standards bodies.

Key Takeaways

For systems engineering practitioners, agentic AI swarms represent augmentation rather than replacement—a means to amplify engineering capability while preserving essential human judgment and accountability. The transformation requires skill evolution but promises substantial capacity enhancement.

For systems engineering researchers, the paper identifies substantial challenges requiring investigation: coordination overhead, domain knowledge integration, emergent behavior management, evaluation metrics, human-swarm interaction, and governance frameworks.

For the broader SE community, the framework provides common vocabulary and structure for advancing AI-augmented practice.

Call to Action

As AI capabilities advance, the systems engineering community has an opportunity—and responsibility—to shape how these technologies are applied to engineering practice. Reactive response to AI advancement risks outcomes poorly suited to engineering needs. Proactive engagement can ensure that AI swarms serve engineering values: rigor, safety, accountability, and human welfare.

We call for empirical research validating framework claims, standards development establishing governance frameworks, education evolution preparing engineers for collaborative practice, and community building enabling shared learning.

This paper provides one framework for human-AI integration in systems engineering. We invite the community to build upon, critique, and extend this work toward a future where AI amplifies engineering capability in service of humanity's complex challenges.

Word count: ~420 words

Revision Notes

- ☐ Ensure conclusion aligns with final paper content
- ☐ Consider strengthening final statement
- ☐ Verify all contributions mentioned are present in paper