C2ST
C2 Science and Technology

Research Article
https://doi.org/10.26599/C2ST.2026.9700003

C2 multi-agent factory: large-small model collaboration empowering a new command and control paradigm

Guohua Zhang[1], Chong Wang[2], Wei Dong[3,*], Qianchuan Zhao[2]

[1] Intelligent Science and Technology Academy of CASIC, Beijing 100041, China
[2] Department of Automation, Tsinghua University, Beijing 100084, China
[3] Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China

ABSTRACT

Command and control (C2) is an agile crisis management architecture for highly uncertain emergencies, originating from military operations but now widely deployed in public safety emergency response, traffic and airspace management, industrial energy systems, urban governance, and large-scale event coordination. Traditional C2 systems mainly rely on preset rules, static processes and manual decision-making, which expose obvious limitations in the face of modern environments with high dynamics, complex disturbances and incomplete information. Aiming at the significant problems of traditional command and control systems, such as structural rigidity, information overload, decision lag, and insufficient collaboration, based on the concepts of software factory and single-agent factory, combined with the latest progress of large-small model collaboration in the field of command and control, this paper proposes a new paradigm of large-small model collaboration empowering command and control—the concept of C2 multi-agent factory, which upgrades the single-agent factory that produces "individual intelligence" to a multi-agent factory that produces and runs "swarm intelligence", and is combined with the command and control system and its application scenarios. The overall system architecture of the C2 multi-agent factory and the large-small model collaboration mechanism based on the C2 multi-agent factory are given. Finally, an application example is given to further illustrate how the C2 multi-agent factory can use the large-small model collaboration technology to empower command and control. The C2 multi-agent factory proposed in this paper will find a new path for the construction of an evolvable intelligent command and control system, promote the substantive leap of artificial intelligence towards the C2 system.

KEYWORDS

command and control system; large-small model collaboration; single-agent factory; C2 multi-agent factory

1 Introduction

Command and control (C2) systems are agile, multi-domain decision architectures that orchestrate sensing, analysis, and action under high uncertainty; originating from military operations, they have now extended to a wide range of civilian arenas, including public safety emergency response, urban governance, large-scale event management, industrial energy operations, and transportation/airspace supervision[1]. Traditionally, these systems rely heavily on preset rules, static processes, and manual decision-making, which exposes obvious limitations in the face of modern environments characterized by high dynamics, complex disturbances, and incomplete information[2]. The specific manifestations of the above problems are as follows: 1) The structure is rigid. Traditional C2 systems often adopt the modular software stacking method, which belongs to the "software factory thinking". The function of the modular software is fixed and the reconstruction ability is weak. It is difficult to cope with the complex tasks of dynamic, uncertain and multi-domain integration; 2) Information overload. Multi-source heterogeneous sensor data grows exponentially, far exceeding the operators' processing capacity, and key signals are easily submerged in noise. 3) Decision-making lag. Due to the pace of events is accelerated in the complex environment, the traditional observe-orient-decide-act (OODA) cycle based on manual judgment is difficult to meet the second-level or even sub-second-level response requirements; 4) Insufficient collaboration. The traditional C2 system is characterized by scattered command and control elements, fragmented business, and lack of a unified agent scheduling and cluster collaboration mechanism.

A higher level of intelligence is the inevitable direction of the future development of the command-and-control system. Specifically, the core requirement of intelligent C2 system in the future is to realize the comprehensive qualitative change of easier reconstruction, more accurate perception, faster decision-making and deeper collaboration through "intelligent empowerment". Its final form is to build an agile, flexible, autonomous and evolutionary intelligent command and control ecosystem to navigate the highly complex and rapidly changing environment in the future[2-3].

In recent years, large language models (LLMs) based on Transformer architecture have shown outstanding emergent ability in

Received: June 26, 2026; Revised: July 7, 2026; Accepted: July 30, 2026

*Wei Dong, weidong@tsinghua.edu.cn

Citation: Zhang G H, Wang C, Dong W, et al. C2 multi-agent factory: large-small model collaboration empowering a new command and control paradigm[J]. *C2 Science and Technology*, 2026. https://doi.org/10.26599/C2ST.2026.9700003.

© Chinese Institute of Command and Control. Publishing services by Tsinghua University Press. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

natural language understanding, logical reasoning and generative tasks, which overcomes the deficiencies of traditional artificial intelligence methods such as strong data dependence, single task, weak transfer ability and lack of common sense, and provides a new technical path for C2 intelligence[4-5]. In theory, the large model can undertake high-order cognitive tasks such as situation summary generation, draft preparation of response plan, semantic translation of instructions, and significantly improve the level of human-machine cooperation in the command and control system. However, the direct deployment of large models to the core process of command and control faces significant obstacles[6-7]. Firstly, the large model has high inference delay and large computational resource consumption, and it is difficult to meet the requirements of high real-time command and control links. Secondly, the expression and utilization efficiency of domain-specific knowledge in the large model is low, and it is prone to the problem of factual hallucination or knowledge obsolescence. Finally, the large model lacks the ability to model the dynamic interaction of the physical world, and it is difficult to directly output executable precise control instructions.

At the same time, the field of command and control has accumulated many specialized small models over the years. These models are usually trained for specific tasks, such as target signal recognition, track association and prediction, risk level assessment, resource distribution optimization, etc. Small models have the advantages of fast response, low computational overhead, and high accuracy on specific tasks within their respective task ranges. Many of them have been deployed in existing command and control systems[3–7]. However, they are usually isolated from each other and operate independently. They lack a unified information representation and collaboration mechanism, and are difficult to cope with complex decision-making needs across tasks and levels.

A natural idea is to explore the collaborative work mode of large and small models, use the generalization and understanding ability of large models to guide the task execution of small models, and make up for the shortcomings of large models with the help of the high efficiency and professionalism of small models[8-9]. However, the current research on large-small model collaboration mostly stays at the level of lightweight combination or simple cascade, and lacks systematic framework design for the characteristics of the command and control field. It especially lacks the ability to dynamically organize, schedule and update multiple models to adapt to changing complex environment task requirements.

To this end, this paper proposes a new command and control swarm intelligence paradigm-C2 multi-agent factory. This paradigm is not a simple agent collaboration framework, but a productive system that can continuously produce, configure, combine and evolve large and small model groups. Among them, the meaning of "factory" is embodied in two aspects: one is the production of individual ability of the agent, that is, the factory can dynamically generate or call the corresponding large model instances and small model instances according to the task requirements; the second is the production of agent collaboration, that is, the core function of the factory is to solve the problem of how to achieve efficient collaborative work between large models and various small models through training-time collaboration and runtime collaboration, including key mechanisms such as task decomposition, information interaction, conflict resolution, and result fusion. The production of this collaborative relationship means that the "factory" produces "teams" rather than "individuals", which is the core difference between the "multi-agent factory" and the traditional "single-agent factory".

The specific contributions of this paper are summarized as follows:

1) Compared with the concept of "large-small model collaboration" in the general sense, the "multi-agent factory" proposed in this paper has two essential differences: it emphasizes the systematic organization and management of multi-agents; it emphasizes the evolvability of the system, that is, the factory itself can continuously optimize the production and collaboration strategies of the model through feedback learning.

2) Compared with the traditional single-agent factory, the "multi-agent factory" proposed in this paper realizes the "agent team" rather than the production, organization and management of "agent individual", and emphasizes the construction, training and support of the cooperative relationship and ability between multi-agents.

3) The paper constructs a new paradigm of intelligent command and control system with the core of large-small model collaboration and multi-agent factory, which enriches and improves the theoretical system of intelligent C2.

4) The paper provides a technical scheme that can be implemented in engineering for the real pain points of rigid structure, information overload, decision lag and insufficient coordination of the traditional command and control system, to realize the comprehensive improvement of the structure, perception, decision-making, coordination, adaptive and continuous evolution ability of the command and control system.

2 Related work

The evolution of the collaborative paradigm of large and small models has gone through three progressive stages: the large model tool learning stage has the preliminary ability to call and understand, but lacks closed-loop feedback[10-11]; the autonomous phase of single agent realizes the closed-loop link of independent tasks[12-13], but it is easy to produce conflict and resource preemption when multi-agents coexist[14]. Although the current multi-agent collaboration stage can handle multi-dimensional tasks in parallel, it faces the bottleneck of agent generation disorder, scheduling chaos, and shortcoming that system capability that cannot continue to evolve[15-16]. In order to break through the above limitations, this paper proposes the "C2 multi-agent factory" paradigm, in which the large model is responsible for global cognitive decision-making, the multi-agent factory undertakes the unified operation and maintenance of the whole life cycle of the intelligent command and control system, and the small model cluster is responsible for edge real-time perception and high-precision execution, so as to realize the essential leap from disorderly generation and isolated scheduling to on-demand production and evolvable system.

At the collaborative mechanism and theoretical level, the existing research has sorted out five types of collaborative paradigms, such as pipeline and routing[8] and has achieved results in general frameworks such as cloud-end collaboration[9], CoLLM-based dynamic model selection[17], LSAI embodied intelligent adaptive splitting[18], and AdaSwitch cloud collaborative learning[19].

In the application of command and control, the large model shows great potential and complementary effects in C2 organizational structure[4, 20], force recommendation system[4], hybrid agent and multi-round generative adversarial scheme (COA)[21-22, 23-24], hierarchical decomposition of behavior tree (BT) tasks[25], strategic sign warning[12], risk assessment[13], and multi-agent command and control process model[26-27]. However, the current application in the field of command and control is still dominated by dotlike exploration. Large models mostly invoke specialized small models in one direction, and a systematic two-way collaboration system has

not yet been formed.

On the whole, the collaboration of large and small models is in the critical period of transition from academic exploration to practical empowerment in the field of command and control. At present, it still faces three core constraints: first, the level of collaboration paradigm is low, and it is mainly based on one-way call of large models, lacking dynamic reorganization and multi-agent systematic collaboration ability; second, the management of dedicated small model is lacking, a large number of models running in isolation, and a unified representation and collaborative framework is lacking; third, the evolutionary closed-loop mechanism is missing, and the system is highly dependent on static configuration, and cannot continue to learn and optimize from historical tasks. These shortcomings restrict the application of complex multi-task scenarios and are the key bottlenecks that must be broken through to achieve long-term development of swarm intelligence.

3 C2 multi-agent factory

Based on the above research background and paradigm evolution analysis, this chapter further proposes and elaborates the new command and control paradigm of C2 multi-agent factory, focusing on concept definition, overall architecture, collaborative operation mechanism and key capabilities, aiming to build an intelligent command and control production system that can be engineered and sustainably evolved.

3.1 Concepts related to agent factory

To clearly define the unique connotation of the "C2 multi-agent factory" mentioned in this paper, it should be first distinguished from the two related but different concepts of "software factory" and "agent factory".

3.1.1 Software factory: static software module production

Software factory refers to an engineering paradigm that modularizes, standardizes, and automates the software development process. Its core features include: mass production and assembly of software functional units through mechanisms such as component libraries, templated code generation, and Continuous Integration/Continuous Deployment (CI/CD) pipelines[28]. However, the software factory is essentially rule-driven and unintelligent: this paradigm does not include the ability to perceive and adapt to the dynamic environment of the task. The software modules produced are fixed in function and predictable in behavior after deployment, and cannot adjust the collaboration relationship or optimize their own structure according to the change of complex environment situation. The "software factory thinking" adopted by traditional C2 system is one of the root causes of structural rigidity.

3.1.2 Agent factory: single agent operation and maintenance

Based on software factory, agent factory refers to the introduction of automatic construction, deployment, monitoring, operation and maintenance capabilities of agent instances. Its core features include: providing standardized training and development framework, runtime environment, condition monitoring, log collection and fault restart of agents[29]. However, the agent factory mainly focuses on the life cycle management of a single agent, focusing on solving two problems: "how to quickly generate a running agent" and "how to ensure the stable operation of the agent". Such factories do not naturally have the ability of collaborative training and collaborative scheduling among multi-agents, and do not contain mechanisms for optimizing and evolving the overall behavior of agent clusters. Therefore, the agent factory can be regarded as the infrastructure layer of the multi-agent factory, but it is not equivalent to the latter.

3.1.3 Multi-agent factory: agent mass production + swarm collaboration + system evolution

The multi-agent factory proposed in this paper is a paradigm upgrade that further integrates the two core capabilities of cluster collaboration and system evolution based on the agent factory. Its key features include the following three aspects:

1) Agent mass production: It can dynamically and batchly generate heterogeneous agent instances (including agents with large models as cognitive centers, agents with special small models as execution units, etc.) according to task requirements, rather than pre-static configuration.

2) Swarm collaboration: A unified collaborative training and scheduling framework is established to realize task decomposition, resource allocation, information interaction, conflict resolution and result fusion among agents, so that many agents can cooperate in an orderly manner like "factory assembly line", and swarm intelligence beyond individual agents emerges.

3) System evolution: By recording the execution data of the whole life cycle, the feedback learning mechanism is used to continuously optimize the generation strategy, scheduling algorithm, collaboration protocol and knowledge base of the agent cluster, so that the whole system can continuously improve the performance with the accumulation of task experience.

In short, the positioning of the multi-agent factory is a production and operation system for swarm intelligence. The factory not only "manufactures" the agent, but also "organizes" the training and operation of the agent cluster, so as to "manufacture" the "agent team" with good collaboration capabilities, and gives the "team" the vitality of self-evolution.

3.2 C2 multi-agent factory core definition

Based on the above concept analysis, this paper gives the following core definition of C2 multi-agent factory: C2 multi-agent factory is a production and operation paradigm for the new generation of command and control system in complex dynamic scenarios, which is driven by hierarchical collaboration of large and small models and multi-agent group. It has the capabilities of dynamic generation of agents, task grouping, collaborative scheduling, conflict resolution, full life cycle operation and maintenance, and closed-loop evolution.

The definition contains the following four key connotations:

1) For complex dynamic command and control scenarios: its application domain is a complex environment with high dynamics, complex disturbances and incomplete information, rather than a conventional scenario.

2) Hierarchical collaboration of large and small models is the core driving force: large models undertake global cognition and strategy generation, small models are responsible for real-time execution at the edge, and the two form an organic whole through factory scheduling.

3) Full-process multi-agent group collaborative management capability: covering the complete life cycle from agent generation to decommissioning, and on this basis, realizing multi-agent cluster generalization, grouping, scheduling and conflict resolution.

4) Closed-loop evolution ability: Enable the system to learn from historical tasks and continuously optimize its own structure and behavior.

Different from the "static function stacking" of the traditional

command and control system, the C2 multi-agent factory has realized the paradigm transition to "dynamic intelligent cluster growth". In the traditional system, the function module is no longer changed once it is deployed; the multi-agent factory can dynamically "grow" an agent cluster with collaboration capabilities that adapts to real-time task requirements, and automatically recover resources after the task ends. The on-demand, dynamic and adaptive ability is the core key to deal with complex environmental uncertainty.

3.3 The overall system architecture of C2 multi-agent factory

The system architecture of C2 multi-agent factory is divided into three layers from top to bottom: large model global decision layer, multi-agent factory core scheduling layer and small model edge execution layer. The three layers interact with data flow, control flow and feedback flow through standardized interfaces. Fig. 1 shows the overall architecture. Specifically, the top layer is the brain layer, that is, the global decision layer of the large model, which receives the global situation of the complex environment and the user's intention, and outputs the task decomposition scheme and the macro strategy. The middle layer is the hub layer, that is, the core scheduling layer of the multi-agent factory, which receives the tasks and strategies issued by the top layer and is responsible for the instantiation, grouping, scheduling, conflict resolution and evolutionary update of the agents. The bottom layer is the hand and foot layer, that is, the small model edge execution layer, which is deployed in each edge node or edge computing unit. It is responsible for real-time perception, accurate control and rapid feedback, and performs specific instructions issued by the middle layer.

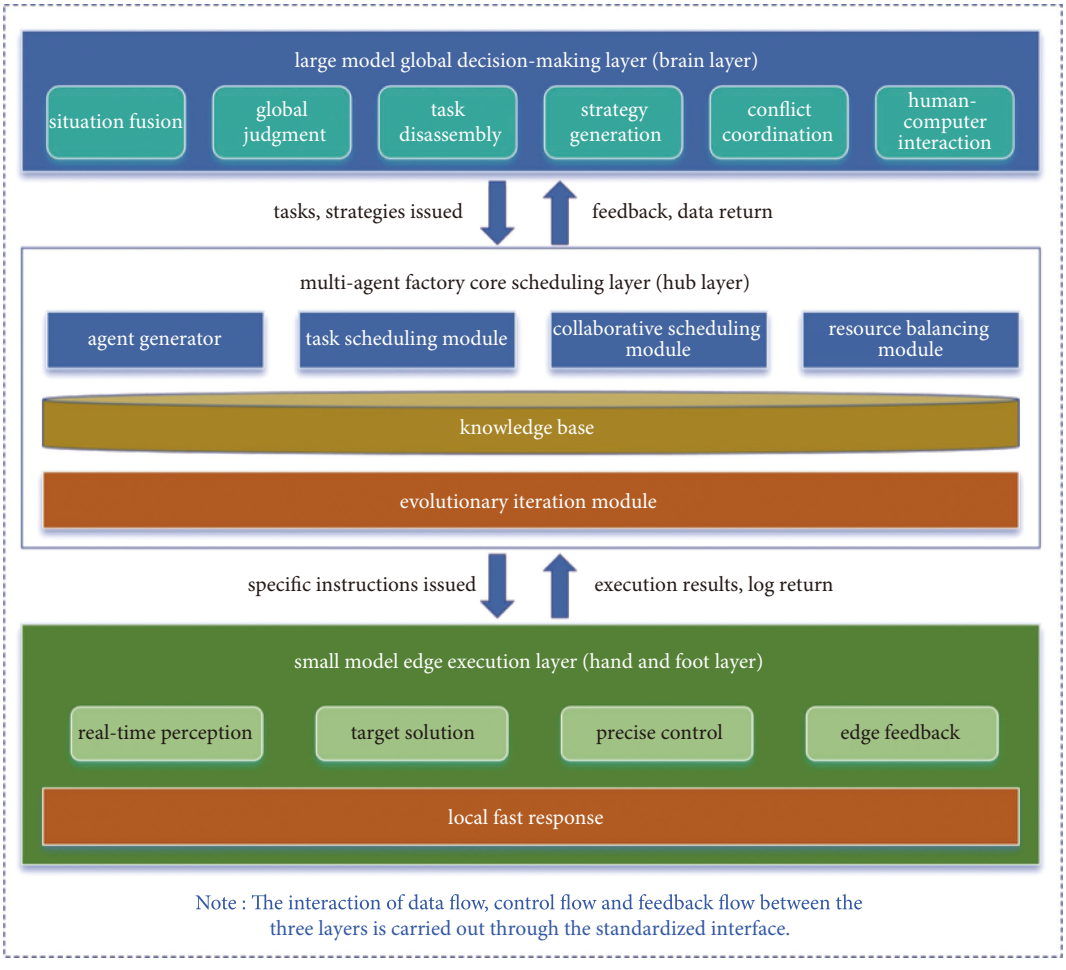


Figure 1 The system architecture of the C2 multi-agent factory, including the large-model global decision-making layer, multi-agent factory core scheduling layer, and small-model edge execution layer.

3.3.1 Global decision-making layer of large model

This layer is the cognitive center of the C2 multi-agent factory, which is usually deployed in the command center or the high computing power node in the cloud. It is not directly involved in millisecond-level real-time control, and its output provides task description and strategy guidance for the middle layer. The main functions of this layer include the following six aspects:

1) Situation fusion: semantic understanding and fusion of heterogeneous data from multi-source sensors to form a unified complex environment situation map.

2) Global judgment: based on the fusion situation, evaluate the capabilities, task environment and mission constraints, and carry out high-level risk analysis and task planning.

3) Task disassembly: the decision-maker's macro intention is decomposed into executable sub-task sequences.

4) Strategy generation: generate macro execution strategies for each subtask, including resource requirements, priorities, constraints, and collaboration protocol recommendations.

5) Conflict co-ordination: identify potential conflicts between different sub-tasks in terms of resources, timing, and space, and give a collaboration plan.

6) Human-computer interaction: provide a natural language interface to support decision-makers to intervene in tasks, adjust strategies or query status through dialogue, and give an interpretable decision-making basis.

3.3.2 Core scheduling layer of multi-agent factory

This layer is the central scheduling system of C2 multi-agent factory, which is responsible for transforming high-level tasks and strategies into specific agent collaborative behaviors. It consists of six core modules:

1) Agent generator: according to the task requirements, dynamically instantiate the required types of agents from the agent template library, support the batch generation of large model agents (cognitive), small model agents (executive) and composite agents, and carry out targeted training on the group collaboration ability of agents, and allocate unique identifiers, initial parameters and resource quotas for each agent.

2) Task scheduling module: mapping the sub-task sequence issued by the large model into the execution workflow of the agent, which is responsible for defining the dependencies between tasks, parallel/serial execution logic, synchronous/asynchronous communication mode, timeout and exception handling strategy.

3) Collaborative scheduling module: based on task scheduling, each agent is scheduled to perform corresponding actions, maintain the global agent state table (idle, executing, blocking, fault), and issue instructions according to priority and resource constraints. The module supports event-driven dynamic scheduling, which can respond to real-time changes in the complex environment and adjust the execution order.

4) Resource balancing module: continuously monitor the usage of each agent and underlying computing resources (CPU, memory, communication bandwidth, edge node load), dynamically adjust the deployment location and resource allocation of agents, and avoid local overload or resource idleness.

5) Knowledge base: store static knowledge (equipment parameters, task orders, response rules) and dynamic knowledge (historical task records, agent performance statistics, success/failure cases) related to the field of command and control; the knowledge base is not only used to assist task arranging and scheduling decisions, but also provides training data for evolutionary iteration modules.

6) Evolutionary iteration module: based on the execution logs and evaluation indicators accumulated in the knowledge base, the models and strategies of the agent generator, task scheduling module, and collaborative scheduling module are optimized regularly or trigger.

3.3.3 Edge execution layer of small model (hand and foot layer)

This layer consists of many dedicated small models and lightweight agents deployed on edge nodes (such as unmanned platforms, sensor front-ends). Its main functions include the following five aspects:

1) Real-time perception: low-latency processing of raw data from photoelectric, infrared and other sensors to achieve target detection, recognition and tracking.

2) Target solution: perform numerical computing tasks such as track filtering, association, prediction and risk value calculation.

3) Precise control: generate control commands for specific actuators, such as target assignment, motion control of unmanned platforms, etc.

4) Edge feedback: the execution results, local situation changes and abnormal events are sent back to the scheduling layer in real time in a compressed form to establish closed-loop feedback.

5) Local fast response: when the communication with the scheduling layer is interrupted or the delay is too long, it can make independent decisions according to the local cache strategy and preset rules to ensure the sustainability of the basic capability.

3.4 Collaborative operation mechanism of large-small models based on C2 multi-agent factory

In the C2 multi-agent factory, the collaboration of large and small models is not a simple one-way call, but a deep coupling through four complementary mechanisms. These mechanisms cover the division of capabilities, task flow, strategic distillation and data feedback, and integrate the core operational capabilities of the agent's standardized dynamic mass production, heterogeneous grouping collaborative operations, multi-task conflict resolution and load balancing, fault self-healing, and full-link closed-loop evolution, which together constitute a complete and sustainable optimization of the collaborative operation system.

3.4.1 Capability division collaboration: large model planning the overall situation, small model executing the local task, agent converting tasks

The mechanism clarifies the ability division of the three roles, and on this basis, integrates the standardized encapsulation and dynamic mass production capabilities of the agents (as shown in Fig.2). The large model is responsible for "seeking the overall situation", dealing with long context, complex reasoning, fuzzy intention understanding and creative generation; its output does not pursue millisecond response, but ensures the rationality and interpretability at the strategic level. The small model is responsible for "executing local", providing high-precision, low-latency deterministic calculations for specific perception, decision-making or control problems. Its output can directly drive the actuator. Agents act as the "transition layer" between large and small models: each agent encapsulates one or a set of models (large or small), and receives tasks and returns results through a standardized interface. In order to realize the efficient operation of the transition role, the system first defines a unified agent description specification, including capability declaration, interface protocol (input/output format, synchronous/asynchronous mode), life cycle state model (create → ready → run → pause → destroy) and metadata template. All agents need to follow this specification for encapsulation. On this basis, the scheduling layer can automatically expand and reduce the number of agent instances according to the peak-valley change of task load. For example, when faced with a large-scale target influx, the agent generator can quickly create dozens of target recognition agents for parallel processing; after the task is completed, the redundant instances are automatically recovered to release the resources. The scheduling layer does not directly operate the model, but interacts through agent instances, thereby shielding the heterogeneity of the underlying model and realizing the standardization and dynamic mass production of the agents.

3.4.2 Task flow collaboration: decision-disassembly-"agent generation"-scheduling-execution-"closed loop feedback"

The mechanism describes the complete information flow closed-loop from user instructions to task completion, and on this basis, the dynamic grouping, collaborative operation and task continuation under fault conditions of heterogeneous agents are realized.

As shown in Fig.3, the specific process includes six steps:

1) Decision-making: the global decision-making layer of the large model receives the instruction of the decision-maker or the task requirements from autonomous identification, and performs

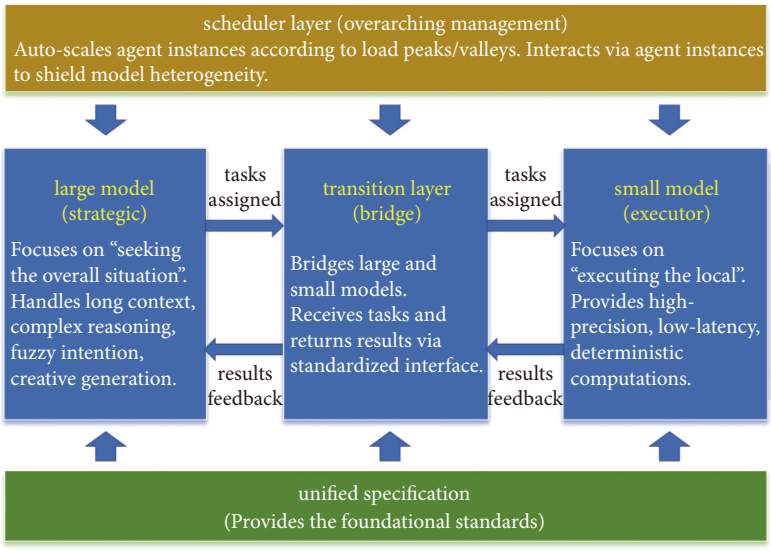


Figure 2 The capability division collaboration in the C2 multi-agent factory, in which the scheduler organizes and coordinates all the agents, the large model implements global reasoning, and the small models execute local tasks.

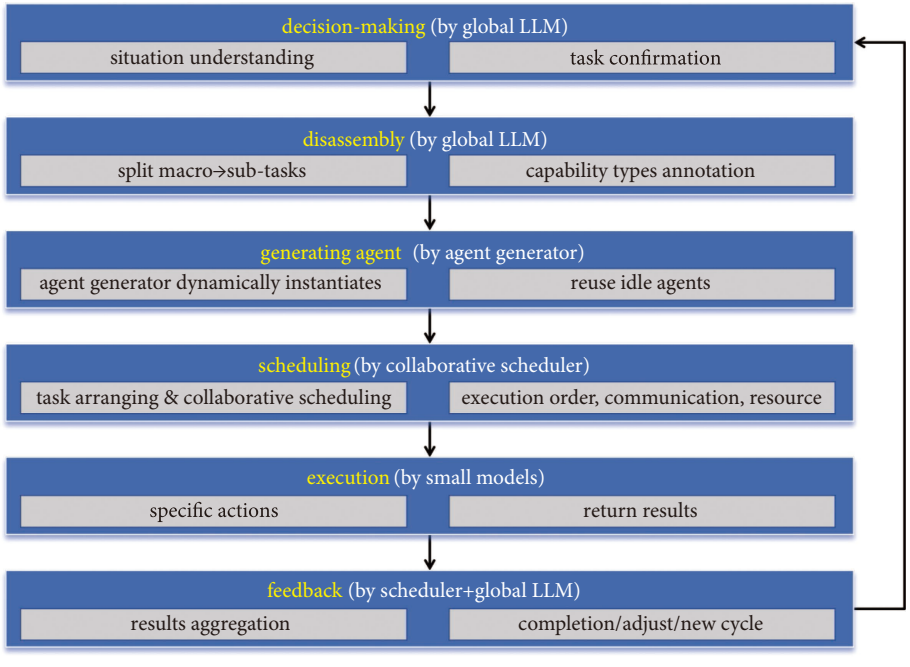


Figure 3 The task flow collaboration in the C2 multi-agent factory, from situation understanding and task disassembly to dynamic agent generation, collaborative scheduling, small-model execution, and feedback aggregation for closed-loop adjustment.

situation understanding and task confirmation.

2) Disassembly: the large model disassembles the macro task into sub-task sequences and annotates the required capability types (such as visual recognition, risk assessment, resource distribution) for each sub-task.

3) Generating Agent: the agent generator of the core scheduling layer dynamically instantiates the corresponding agent according to the capability type (if there are already idle agents, they are reused).

4) Scheduling: the task scheduling module and the collaborative scheduling module determine the execution order, communication mode and resource allocation of the agent, and issue instructions.

5) Execution: the small model edge execution layer (or other types of agents) performs specific actions and returns the results to the scheduling layer.

6) Feedback: after the execution results are aggregated by the scheduling layer, they are fed back to the large model decision-making layer to determine whether the task is completed, whether the strategy needs to be adjusted, or whether a new round of cycle is initiated.

In this closed-loop, the scheduling layer supports the dynamic combination of multiple heterogeneous agents (such as large model cognitive agents, small model perception agents, and control agents) into a logical group, giving group identification and collaboration protocols. The grouping can be preset according to the task type, and can also be generated in real time by the large model. Agents in the group communicate through publish/subscribe, request/response, or shared blackboard modes. The scheduling layer is responsible for maintaining communication

topology, message routing, and synchronization barriers to ensure timing consistency and data consistency of collaborative operations. In addition, the system has a built-in health monitoring probe, which regularly detects the heartbeat of agent instances and underlying nodes. Once a fault is detected, the agent is immediately marked as abnormal and triggers the recovery process. For the agent of critical task, the system maintains its state checkpoint; when the original agent fails, the agent generator quickly creates a new instance of the same type, and restores the state from the nearest checkpoint to realize agent replacement and task continuation. The replaced agent seamlessly continues the original task, and the upper caller does not need to perceive the fault process; for unrecoverable faults, the scheduling layer rearranges the tasks and allocates them to other healthy agents to ensure the overall completeness of the tasks. The closed-loop supports nesting and concurrency, and can realize the interleaving operation of multiple task flows.

3.4.3 Strategy distillation collaboration: large model sends the optimal strategy and rules to small models

Although the large model does not directly participate in real-time control, it can generate better local strategies from a global perspective. The policy distillation mechanism realizes the downward migration of the above knowledge, and integrates multi-task conflict resolution and load balancing capabilities in this process. Specifically, the large model extracts decision rules, parameter configuration or behavior patterns for specific scenarios by analyzing historical task data or simulation deduction. This knowledge is encapsulated as a "policy package", which is sent to the relevant small model agent through the scheduling layer. The small model agent uses the policy package to adjust its own model parameters, rule base or decision threshold locally, to perform better in subsequent execution. For example, the large model judges the interference mode that may occur according to the overall complex environment situation, generates the corresponding filtering parameter adjustment strategy, and sends it to the small model of signal recognition, so that it can maintain high recognition accuracy in the interference environment. In the process of policy release and execution, when multiple concurrent tasks compete for the same agent instance, the same edge node or the same communication resource, the cooperative scheduling module of the scheduling layer performs conflict resolution according to the preset priority strategy (such as task critical level, deadline, resource preemption mode), and supports deadlock detection and cancellation mechanism. At the same time, the resource balancing module continuously monitors the resource utilization rate of each computing node. By dynamically migrating agent instances, adjusting task allocation weights or starting new computing nodes, the overall load of the system is maintained within a reasonable threshold to ensure that the policy distillation collaboration can still operate stably in a high load and conflict environment.

3.4.4 Data feedback collaboration: edge executing data feedback, driving the continuous evolution of the system

This mechanism is the data basis for C2 multi-agent factory to realize closed-loop evolution, and it is also the core embodiment of the closed-loop evolution ability of the whole link. The edge execution layer records fine-grained logs such as input features, intermediate results, output actions, execution time, resource consumption, and abnormal events during the execution of tasks. After desensitization and aggregation, these logs are stored and preprocessed through the knowledge base of the scheduling layer. The evolutionary iteration module periodically reads the accumulated data in the knowledge base, and uses offline or online learning algorithms to optimize the model and strategy of each module in the scheduling layer. Specifically, the system automatically collects key performance indicators from the task initiation to the end of the execution, including task completion rate, average response delay, resource efficiency, decision accuracy, abnormal rate, and process log. The evolution iteration module supports two modes: offline mode, that is, training the optimization strategy on the simulation environment or historical data, and deploying in batches after full verification; online mode, which uses online learning algorithms (such as contextual multi-armed bandit, policy gradient) to adjust scheduling parameters in real time, realizes the gradual improvement of system performance. After each evolutionary deployment, the system automatically compares the key indicators before and after evolution; if performance degrades or an exception occurs, it automatically rolls back to the previous stable version and triggers an alert for manual analysis. Through the circulation of the above mechanism, the C2 multi-agent factory can continuously improve the rationality of task disassembly, agent scheduling efficiency, resource utilization and decision-making quality with the increase of task times, and truly realize the evolutionary goal of "more use, smarter".

4 Application examples

In order to further clarify how the C2 multi-agent factory uses large-small model collaboration to empower command and control, this section presents a typical urban traffic signal control scenario, showing the closed-loop logic of the factory's 'on-demand generation → collaborative operation → feedback evolution'. This case fully reflects the essential advantages of the factory relative to the traditional paradigm: dynamic response, continuous evolution, decentralized coordination, and complementarity between the large inference model and the small execution model.

In the urban central road network composed of dozens of signalized intersections, there are significant differences in traffic flow in time and space. Traditional signal control systems usually rely on fixed-time plans or rule-based adaptive strategies to optimize each intersection in isolation. These methods perform well in medium and predictable traffic conditions. However, in the morning and evening rush hours, when traffic demand surges, or accidents such as vehicle failures or bad weather lead to local congestion, the limitations of isolated optimization become obvious: a single congested intersection can propagate backward through the network, triggering a chain reaction and reducing the mobility of the entire road network. Under such conditions, the operator is forced to manually adjust the signal timing plan, which is a time-consuming, passive process and heavily dependent on the operator's experience.

The C2 multi-agent factory starts the dynamic response and online evolution process when the traffic management system detects an abnormal situation.

Step 1: Global perception and anomaly detection. The large model integrates real-time data from ring detectors, radar sensors, connected vehicle trajectories and intersection cameras, as well as external information such as weather forecasts and scheduled nearby events. Through semantic understanding and causal reasoning, it is determined that a large concert is about to end in the southern part of the region, and eastbound traffic on the main artery is rapidly approaching saturation. Based on this fusion perception, the large model decomposes the overall control target into three mutually coordinated sub-tasks: 1) Trunk green wave optimiza-

tion to promote eastbound export; 2) Adjacent inlet boundary metering adjustment for inflow; 3) Dynamically adjust the surrounding diversion to realize the redistribution of traffic between alternative corridors.

Step 2: Dynamic agent generation and task allocation. The agent generator dynamically instantiates three types of agents according to subtask requirements. For the arterial green wave task, the signal control small model based on the reinforcement learning algorithm is re-used and initialized with the latest traffic state parameters. For the boundary measurement task, it is based on the same basic model, but has a specific speed limit constraint, instantiating a separate agent. For the perimeter offloading task, it initiates a cognitive agent that invokes a dedicated regional traffic prediction model. Each intersection signal controller operates as an independent agent, and the regional-level global scheduling agent coordinates collaborative efforts.

Step 3: Coordinated scheduling execution. The scheduling layer organizes the generated agents into a logical "congestion mitigation task group" and defines their collaboration protocols. For the trunk green wave agent, the scheduling layer provides real-time data feed from the upstream and downstream adjacent intersections to ensure that the agent is not optimized in isolation, but is optimized by continuously exchanging the number of vehicles and the queue length with the adjacent intersection agents. This dynamic coordination together forms a smooth green wave traffic. When local congestion begins to form, the boundary measurement agent is activated at the upstream entrance, and an adaptive rate limit is imposed to prevent queue overflow from entering the main line. At the same time, the induced cognitive agent calculates the alternative path recommendation and propagates it in the network. This distributed coordination ensures that local congestion is absorbed locally without spreading to the whole network.

Step 4: Online fault recovery and evolutionary adaptation. During the implementation process, the network connection with the agent responsible for the secondary corridor failed. The health probe in the scheduling layer immediately detects the loss of heartbeat, marks the agent as abnormal, and triggers recovery. The agent generator instantiates an alternative agent, restores its state from the nearest checkpoint, and seamlessly restores the operation without destroying the overall control loop. At the same time, the execution layer records fine-grained logs – including detection count, phase duration, queue length, and travel time estimation – and feeds them back to the knowledge base. The evolutionary iteration module periodically mines these accumulated data to refine the underlying optimization model. The green wave model is retrained offline by using historical congestion events, and the boundary measurement threshold is fine-tuned online by using the contextual multi-armed bandit algorithm, so that the system can gradually improve the response capability to repeated traffic patterns.

This example shows the online evolution and adaptive coordination ability of C2 multi-agent factory. The conventional "development → test → deployment" long period of signal timing optimization is compressed to the order of minutes. Crucially, it embodies a deep collaboration between large and small models: large models provide a semantic understanding and strategy decomposition of the overall traffic signal control task, while small models ensure real-time, low-latency control of each intersection. Distributed coordination prevents congestion propagation, and continuous feedback loops allow the system to evolve with changes in traffic patterns.

5 Conclusions

Aiming at the bottlenecks of traditional command and control system in intelligent transformation, such as structural rigidity, information overload, decision lag and lack of collaboration, this paper proposes a large-small model collaboration paradigm of "multi-agent factory", which is oriented to the field of command and control. It aims to build a systematic system that can continuously produce, configure and evolve agent clusters, and realize the whole process of multi-model and data fusion to enable command and control tasks. Different from the direct deployment of general large models or the cascade of simple models, the core of multi-agent factory is to realize high-level understanding and task decomposition of complex environment by using the general cognition and reasoning ability of large model. At the same time, with the help of the deployed dedicated small models (such as target recognition, track prediction, resource distribution, etc.), efficient and low-latency accurate calculation and execution are realized, and the targeted training and runtime support of multi-agent collaboration are realized. Thus, the generalization ability, special advantages, collaboration and iterative evolution are incorporated into a unified intelligent command and control framework.

In recent years, technologies such as model lightweighting, federated learning, knowledge distillation, and edge intelligence have gradually matured, laying the underlying foundation for the collaborative learning of large and small models between command and control nodes and knowledge transfer under data protection. With the large-scale deployment of unmanned platforms and intelligent sensors, the demand for "agility, flexibility, autonomy and evolution" of the command and control system is increasingly urgent. The multi-agent factory will open a new path for the construction of an evolvable intelligent command and control system, and promote the deep empowerment of artificial intelligence for collaborative and autonomous command and control.

Acknowledgement

This work has been guided by Jincai Huang, Research Fellow of College of Systems Engineering, National University of Defense Technology.

Data availability

No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Author contribution statement

Guohua Zhang: Conceptualization, Investigation, Writing – original draft. Chong Wang: Investigation, Writing – original draft. Wei Dong: Conceptualization, Supervision, Writing – review & editing, Project administration. Qianchuan Zhao: Conceptualization, Supervision, Writing – review & editing. All authors contributed to the literature review and approved the final manuscript.

Use of AI statement

None.

References

[1] Drews P, Fiedrich F. Performance in command and control: results from a scoping review[J]. *European Journal for Security Research*, 2024, 9(1): 57-92.

[2] Almuntasheri S A, Alrumaih T N I, Alablani I A L, et al. A survey on battlefield management systems: architecture, challenges, and future research directions[J]. *Alexandria Engineering Journal*, 2026, 147: 93-116.

[3] Xu C R, Yang C X, Zhang R H. Development of intelligent command and control system[J]. *Ship Electronic Engineering*, 2023, 43(8): 6-9 (in Chinese).

[4] Sun L J, Zhou Y, Wei D J, et al. Issues of large language models for C2 organization[J]. *Journal of Command and Control*, 2024, 10(6): 669-676 (in Chinese).

[5] Jiao P B, Gong Z X, Luo Z H, et al. Large language models for military force recommendation: applications and considerations[J]. *Journal of Command and Control*, 2025, 11(2): 137-145 (in Chinese).

[6] Zhang M W, Kuang M C, Shi H, et al. Command-agent: reconstructing warfare simulation and command decision-making using large language models[J]. *Defence Technology*, 2026, 56: 294-313.

[7] Zhao L, Ju H B, Zhang P Y, et al. Capability boundaries of foundation models and their applications in command and control[J]. *Journal of Command and Control*, 2024, 10(6): 653-660 (in Chinese).

[8] Chen Y, Zhao J H, Han H H. A survey on collaborative mechanisms between large and small language models[PP/OL]. arXiv (2025-05-12)[2026-06-10]. https://doi.org/10.48550/arXiv.2505.07460.

[9] Wang Y W, Shen T, Zhang S Y, et al. Advances in edge-cloud collaboration and evolution for large-small models[J]. *Journal of Image and Graphics*, 2024, 29(6): 1510-1534 (in Chinese).

[10] Xu W K, Huang C R, Gao S, et al. LLM-based agents for tool learning: a survey[J]. *Data Science and Engineering*, 2025, 10(4): 533-563.

[11] Liao N Y, Tian Y, Li Y S, et al. Tool learning with large language models: methods, functions, and mechanisms[J]. *Computer Engineering*, 2025, 51(12): 1-17 (in Chinese).

[12] Zeng Z F, Cheng Q, Mu J W, et al. Strategic early-warning based on experts knowledge and large language models[J]. *Journal of Command and Control*, 2025, 11(2): 158-171 (in Chinese).

[13] Wang Z D, He C C, Gao X G, et al. Causal threat assessment model supervised by LLMs[J]. *Journal of Command and Control*, 2025, 11(2): 191-201 (in Chinese).

[14] Sun C N, Huang S J, Pompili D. LLM-based multi-agent decision-making: challenges and future directions[J]. *IEEE Robotics and Automation Letters*, 2025, 10(6): 5681-5688.

[15] Liu Q D, Wu Y L, Zhang S L, et al. Autonomous cooperation of USV swarms via large-small language model collabo-ration[J/OL]. *Chinese Journal of Ship Research*, 1-13(2025-12-19)[2026-06-15]. https://doi.org/10.19693/j.issn.1673-3185.04552 (in Chinese).

[16] Sun Z F, Zhou M Z, Liao L T, et al. LLM based C2 agent and decision-making in dynamic domains simulation environment[J]. *Journal of Command and Control*, 2025, 11(4): 445-458 (in Chinese).

[17] Wang H T, Ren L, Zhao T, et al. CoLLM: industrial large-small model collaboration with fuzzy decision-making agent and self-reflection[J]. *IEEE Transactions on Fuzzy Systems*, 2026, 34(4): 1120-1130.

[18] Zhou L Y, Leng S P, Liang T H, et al. LSAI: a large small AI model codesign framework for agentic robot scenarios[PP/OL]. arXiv (2026-03-23) [2026-06-10]. https://doi.org/10.48550/arXiv.2603.21726.

[19] Sun H, Wu J Y, Cai H Y, et al. AdaSwitch: adaptive switching between small and large agents for effective cloud-local collaborative learning[C]//Proceedings of 2024 Conference on Empirical Methods in Natural Language Processing. Miami: Association for Computational Linguistics, 2024: 8052-8062.

[20] Gu X Q, Luo J R, Zhou Y Z, et al. Survey on large language agent technologies for intelligent game theoretic decision-making[J]. *Journal of System Simulation*, 2025, 37(5): 1142-1157 (in Chinese).

[21] Tran K T, Dao D, Nguyen M D, et al. Multi-agent collaboration mechanisms: a survey of LLMs[PP/OL]. arXiv (2025-01-10) [2026-06-10]. https://doi.org/10.48550/arXiv.2501.06322.

[22] Luo J Y, Zhang W Z, Yuan Y, et al. Large language model agent: a survey on methodology, applications and challenges[PP/OL]. arXiv (2025-03-27) [2026-06-10]. https://doi.org/10.48550/arXiv.2503.21460.

[23] Pang N, Xiao W D, Zhao X, et al. Course of action based on mixture-of-agents[J]. *Journal of Command and Control*, 2025, 11(2): 181-190 (in Chinese).

[24] Yan Y S, Zhou Y, Wang C, et al. COA generation based on pre-trained large language models[J]. *Computer Science*, 2025, 52(1): 80-86 (in Chinese).

[25] Li Q L, Huang X Q, Zhu X Q, et al. BT-D2: a hierarchical combat task decomposition framework with dynamic constraint injection in behavior tree[C]//Proceedings of 2025 IEEE International Conference on Systems, Man, and Cybernetics (SMC). Vienna: IEEE, 2025: 2709-2714.

[26] Sun Y, Zheng Y, Huang H Y, et al. Multi-loop nested LLM-based multi-agent command and control processes[J]. *Journal of Command and Control*, 2024, 10(6): 732-739 (in Chinese).

[27] Zhao R H, Zeng W X, Tang J Y, et al. Automatic generation of few-shot knowledge graph QA questions via large and small model collaboration[J]. *Journal of Command and Control*, 2025, 11(2): 172-180 (in Chinese).

[28] Greenfield J. Software factories: assembling applications with patterns, models, frameworks, and tools[C]//Proceedings of the 3rd International Conference on Software Product Lines. Boston: Springer, 2004: 304.

[29] Li S Y, Tong Y X, Yang Q, et al. Factory for agent (FAgent): collaborative learning of large and SmallLanguage models[J]. *Computing Magazine of the CCF*, 2025, 1(5): 83-90.