μACP: A Formal Calculus for Expressive, Resource-Constrained Agent Communication

Arnab Mallick
Centre for Development of Advanced Computing
Hyderabad, India
arnabm@cdac.in

Indraveni Chebolu
Centre for Development of Advanced Computing
Hyderabad, India
indravenik@cdac.in

ABSTRACT

Agent communication remains a foundational problem in multi-agent systems protocols such as FIPA-ACL guarantee semantic richness but are intractable for constrained environments, while lightweight IoT protocols achieve efficiency at the expense of expressiveness. This paper presents μACP, a formal calculus for expressive agent communication under explicit resource bounds. We formalize the Resource-Constrained Agent Communication (RCAC) model, prove that a minimal four-verb basis *{PING, TELL, ASK, OBSERVE}* is sufficient to encode finite-state FIPA protocols, and establish tight information-theoretic bounds on message complexity. We further show that μACP can implement standard consensus under partial synchrony and crash faults, yielding a constructive coordination framework for edge-native agents. Formal verification in TLA$^+$ (model checking) and Coq (mechanized invariants) establishes safety and boundedness, and supports liveness under modeled assumptions. Large-scale system simulations confirm ACP achieves a median end-to-end message latency of 34 ms (95th percentile 104 ms) at scale, achieving competitive or lower latency than representative agent and IoT protocols under the evaluated workloads. The main contribution is a unified calculus that reconciles semantic expressiveness with provable efficiency, providing a rigorous foundation for the next generation of resource-constrained multi-agent systems.

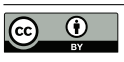
CCS CONCEPTS

• **Theory of computation** → *Distributed algorithms*; *Concurrent algorithms*; • **Networks** → **Peer-to-peer protocols**; **Network protocol design**; • **Software and its engineering** → Formal language definitions; • **Mathematics of computing** → Information theory.

KEYWORDS

Agent Communication Languages; Resource-Constrained Multi-Agent Systems; Formal Protocol Calculus

ACM Reference Format:
Arnab Mallick and Indraveni Chebolu. 2026. μACP: A Formal Calculus for Expressive, Resource-Constrained Agent Communication. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/PHRW6922



Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/PHRW6922

1 INTRODUCTION

Multi-agent systems (MAS) provide a powerful paradigm for complex distributed intelligence [47, 48]. Effective agent communication enables coordination, negotiation, and collaborative problem-solving [27, 43]. However, traditional agent communication languages (ACLs) like FIPA-ACL [19] and KQML [18] assume unconstrained environments, making them ill-suited for edge-native multi-agent systems.

The expansion of IoT devices, edge computing, and resource-constrained embedded systems demands protocols that operate under severe limitations: memory (<100KB), bandwidth (kilobits/second), processing power (microcontroller-class CPUs), and energy [8, 34, 38, 46]. Although lightweight protocols such as MQTT [5] and CoAP [42] address resource constraints, they lack the semantic richness required for sophisticated agent interactions [1].

This disconnect between semantic requirements and practical constraints creates a significant barrier for deploying intelligent agents in resource-constrained environments [9]. Current approaches either sacrifice expressiveness for efficiency (IoT protocols) or incur unacceptable overhead (traditional ACLs) [39]. Consequently, no formal theoretical framework bridges these domains, leaving a critical gap in multi-agent systems theory.

We present μACP (Micro Agent Communication Protocol), a formal calculus for agent communication under severe resource constraints. Our work establishes the theoretical foundation for edge-native multi-agent systems by addressing: *How can we design an agent communication protocol that maintains semantic richness while operating within strict resource constraints?*

Our contributions are fourfold:

(1) **Resource-Constrained Agent Communication (RCAC) Model**: A formal model explicitly accounting for resource constraints, enabling reasoning about semantic expressiveness vs. resource consumption trade-offs.
(2) **Minimal Verb Set Completeness**: Proof that four verbs {PING, TELL, ASK, OBSERVE} are complete for finite-state agent communication under resource constraints, establishing theoretical bounds on semantic primitives.
(3) **Semantic Compression Theory**: A formal theory providing rigorous bounds on message complexity, demonstrating μACP achieves optimal compression with amortized $O(1)$ per-message complexity.
(4) **Emergent Coordination Framework**: Theoretical conditions under which resource-constrained agents achieve emergent coordination, providing formal guarantees on system behavior despite individual limitations.

Our theoretical framework advances multi-agent system foundations enables edge-native deployments previously impossible due to

protocol limitations [21]. The calculus provides theoretical foundations for the deployment of intelligent agents on microcontrollers, sensor networks, and resource-constrained platforms, opening new research directions in edge AI and distributed intelligence [50].

Scope and assumptions. μACP targets finite-state, bounded-nesting agent protocols under crash/omission faults. These assumptions enable mechanized verification while covering widely used MAS interaction patterns, including contract-net, request/response, and publish/subscribe. Protocols requiring unbounded recursive conversations or Byzantine fault tolerance are outside the current scope and are discussed as future extensions.

2 BACKGROUND

Agent communication draws on speech act theory, resource-constrained computing, information theory, and formal verification [4, 22, 40].

Speech acts and agent communication. Speech act theory models utterances as actions [4, 40]. In multi-agent systems, communication is realized through performatives [12, 44]. A speech act is defined as

$$SA = (F, P, C),$$

where F is the illocutionary force, P the propositional content, and C the context [12]. This abstraction underlies ACLs such as FIPA-ACL and KQML [27]. Formally, for agents $\mathcal{A}$, messages $\mathcal{M}$, and time $\mathbb{T}$,

$$\text{Comm}(a_i, a_j, m, t) : \mathcal{A} \times \mathcal{A} \times \mathcal{M} \times \mathbb{T} \to \mathbb{B}$$

denotes successful communication [48].

Resource-constrained computation. Embedded and sensor systems operate under strict CPU, bandwidth, memory, and energy limits [34, 46]. A system is characterized by $(R, B, M, E) \in \mathbb{R}^+$. Protocol design minimizes weighted resource consumption

$$\min_{p \in \mathcal{P}} \{\alpha R_p + \beta B_p + \gamma M_p + \delta E_p\}$$

subject to functional requirements $F(p) \geq F_{\min}$ [9, 31].

Information theory and semantic compression. Information theory bounds communication via entropy [41]:

$$H(X) = -\sum_x p(x) \log_2 p(x).$$

For agent primitives $\mathcal{S} = \{s_i\}$, the semantic entropy is

$$H_S(\mathcal{S}) = -\sum_i p(s_i) \log_2 p(s_i)$$

[6]. Although compression is well understood [13, 51], the semantic minimality for agent communication remains largely unexplored.

Formal verification. Formal methods specify and verify protocol correctness [11, 23]. Safety, liveness, and fairness are expressed in temporal logic [33], and verified using tools such as SPIN and TLA+ [23, 28].

3 RELATED WORK

Agent communication languages. FIPA-ACL [19] and KQML [18] are the dominant ACLs. Their rich performative sets and BDI-grounded semantics [35] incur substantial message overhead, limiting suitability for constrained devices [27, 39]. Social semantics [43] and modern JSON-based protocols (MCP, A2A, ANP) [16] improve interoperability but lack formal semantics and resource guarantees.

Lightweight protocols. IoT protocols such as MQTT and CoAP minimize headers but provide no semantic reasoning layer [1, 5, 42]. More expressive models, including XC [3], SMrCaIT [10], and HpC [49] support coordination, timing, and security, but do not address semantic minimality or bounded-resource completeness.

Constrained agent systems. Lightweight BDI platforms still depend on heavyweight communication layers [32]. Existing calculi model timing, probability, or security [7, 10, 45, 49], but none identify a minimal set of expressive primitives with provable compression or coordination guarantees.

Session types and related logics. Session types ensure protocol conformance and progress [14, 24, 25], but abstract away execution costs. Concurrent-constraint calculi emphasize declarative synchronization rather than message semantics [36, 37]. Resource-bounded strategic logics extend temporal reasoning to limited budgets [2, 20], but do not yield executable communication protocols.

Recent advances. Recent work on LLM-based agents and large-scale interoperability has shifted away from classical ACLs toward practical standards [16, 17]. Formal verification of neural-symbolic multi-agent systems is also emerging [26], motivating a minimal, resource-aware, formally grounded communication model.

Research Gaps and Contributions. We identify four gaps: (1) **Semantic minimality**: no minimal, complete verb set under constraints. (2) **Resource-aware encoding**: no formal size/CPU/memory/energy bounds. (3) **Coordination with guarantees**: no reduction to consensus with safety/liveness proofs. (4) **Formal verification**: no machine-checked proofs of resource and coordination.

μACP addresses these by providing minimal primitives, resource-theoretic bounds, consensus reduction, and mechanized verification in Coq/TLA+.

4 THE μACP CALCULUS

We now present μACP, a formal calculus for resource-constrained agent communication.

Symbol	Meaning
$\mathcal{A}$	Set of agents
$\mathcal{R}_c = (M, B, C, E)$	Resource budget: memory, bandwidth, CPU, energy
$\mathcal{C}$	Set of communication channels
$\mathcal{P} = (\mathcal{V}, \mathcal{O}, \mathcal{Q})$	Protocol specification: verbs, options, QoS
$\mathcal{S}_i = (M_i, K_i, T_i, H_i)$	Local state of agent i
$\rho(a_i, v, o, p)$	Resource consumption of action (a_i, v, o, p)
$m = \langle h, v, o, p \rangle$	Message structure
$\tau(f)$	Translation from FIPA performative f to μACP
$T_{FIPA}(P)$	Observable traces of a FIPA protocol P
$T_{\mu ACP}(\tau(P))$	Observable traces of the μACP encoding of P

Table 1: Notation used in the paper

4.1 Formal Model Definition

We begin by defining the abstract space in which μACP operates.

Definition 1 (Resource-Constrained Agent Communication (RCAC) Space). An *RCAC space* is a tuple

$$\mathcal{R} = (\mathcal{A}, \mathcal{R}_c, C, \mathcal{P})$$

where:

- $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ is a finite set of agents,
- $\mathcal{R}_c = (M, B, C, E)$ is the resource budget, with memory $\leq M$, bandwidth $\leq B$, CPU cycles $\leq C$, and energy $\leq E$,
- $C = \{c_1, c_2, \ldots, c_m\}$ is the set of available communication channels,
- $\mathcal{P} = (\mathcal{V}, \mathcal{O}, \mathcal{Q})$ is the protocol specification, consisting of verbs, options, and QoS levels.

Definition 2 (Agent State). Each agent $a_i \in \mathcal{A}$ has a local state

$$\mathcal{S}_i = (M_i, K_i, T_i, H_i)$$

where M_i is the memory state, K_i is a knowledge base (set of ground literals), T_i is a timer configuration and H_i is a bounded history buffer. The global state is $\mathcal{S} = \prod_{i=1}^{n} \mathcal{S}_i$.

Definition 3 (Resource Consumption Function). An action of the form (a_i, v, o, p) induces consumption

$$\rho(a_i, v, o, p) = (\rho_m, \rho_b, \rho_c, \rho_e)$$

where each component denotes memory, bandwidth, CPU, and energy usage. The action is feasible iff $\rho(a_i, v, o, p) \leq \mathcal{R}_c$.

Corollary 1 (Concrete Resource Bounds). *For any agent a_i operating under message rate $\lambda_i \leq \lambda_{\max}$ over time horizon T, the cumulative resource consumption satisfies:*

$$\begin{aligned} \mathit{Bandwidth}_i(T) &\leq B \cdot T, \\ \mathit{Memory}_i(T) &\leq M, \\ \mathit{CPU}_i(T) &\leq C_p \cdot T, \\ \mathit{Energy}_i(T) &\leq E \cdot T \end{aligned}$$

Thus, all executions of μACP remain within linear bounds of system resources.

Remark. *The model is intentionally agnostic about the internal architecture of agents (BDI, reactive, learning, etc.). Only the observable communication and resource costs are relevant at this level of abstraction.*

4.2 Syntax and Semantics

Message Structure. Messages in μACP are defined by the following grammar:

$$m ::= \langle h, v, o, p \rangle$$

where

- $h = \text{header}(id, seq, q, flags)$ is a fixed-length header,
- $v \in \{\text{PING}, \text{TELL}, \text{ASK}, \text{OBSERVE}\}$ is a verb,
- o is a sequence of TLV (type–length–value) options,
- $p \in \{0, 1\}^*$ is the payload.

Definition 4 (Well-Formed Message). A message is well-formed iff (i) the header has size 64 bits, (ii) the verb field encodes one of four values in 2 bits, (iii) each option is consistent with its declared length, (iv) the total option size does not exceed 1024 bytes, (v) payload size $\leq 2^{16} - 1$ bytes.

Operational Semantics. We give semantics via a labeled transition system $(\mathcal{S}, \mathcal{L}, \rightarrow)$ where labels $\ell \in \mathcal{L}$ represent communication actions $(sender, receiver, verb, options, payload, channel)$.

Inference 4.1 (Send)**.**

$$\frac{m = \langle h, v, o, p \rangle \text{ well-formed} \quad \rho(a_i, v, o, p) \leq \mathcal{R}_c}{\sigma \xrightarrow{(a_i, a_j, v, o, p, c)} \sigma'}$$

where σ' updates a_i's history and resources.

Inference 4.2 (Receive)**.**

$$\frac{\text{channel}(c) = \text{stable} \quad \rho(a_j, v, o, p) \leq \mathcal{R}_c}{\sigma \xrightarrow{(a_i, a_j, v, o, p, c)} \sigma'}$$

where σ' updates a_j's knowledge base and history.

Resource Semantics. Instead of continuous ODEs, we use discrete invariants: every send or receive reduces the resource counters, and feasibility requires non-negativity.

Theorem 5 (Resource Boundedness). *If all actions are feasible and each agent's resources are finite at $t = 0$, then resources remain non-negative for all finite executions.*

Proof. By induction on the length of executions: each step subtracts bounded non-negative amounts from finite counters, hence values remain non-negative. □

4.3 Minimal Verb Set Theory

Verb Semantics. The core insight is that four verbs suffice. Their semantics are given in a Kripke-style model (W, R, V) where W is a set of worlds, R accessibility, V valuation:

$$[\![\text{PING}]\!] = \lambda a_i, a_j, w.\ \text{Reachable}(a_i, a_j, w) \tag{1}$$

$$[\![\text{TELL}]\!] = \lambda a_i, a_j, w, \phi.\ \text{Bel}(a_i, \phi, w) \wedge \text{Inform}(a_i, a_j, \phi, w) \tag{2}$$

$$[\![\text{ASK}]\!] = \lambda a_i, a_j, w, \phi.\ \text{Want}(a_i, \text{Bel}(a_j, \phi), w) \wedge \text{Query}(a_i, a_j, \phi, w) \tag{3}$$

$$[\![\text{OBSERVE}]\!] = \lambda a_i, a_j, w, \phi.\ \text{Happens}(\phi, w) \wedge \text{Notify}(a_i, a_j, \phi, w) \tag{4}$$

Completeness via Encoding. Let $\mathcal{F}$ be the set of FIPA-ACL performatives. Define a translation function $\tau : \mathcal{F} \rightarrow \mathcal{V} \times \mathcal{O}$.

Example 6.

$$[\![\text{PING}]\!] = \lambda a_i, a_j, w.\ \text{Reachable}(a_i, a_j, w) \tag{5}$$

$$[\![\text{TELL}]\!] = \lambda a_i, a_j, w, \phi.\ \text{Bel}(a_i, \phi, w) \wedge \text{Inform}(a_i, a_j, \phi, w) \tag{6}$$

$$[\![\text{ASK}]\!] = \lambda a_i, a_j, w, \phi.\ \text{Want}(a_i, \text{Bel}(a_j, \phi), w) \wedge \tag{7}$$

$$\text{Query}(a_i, a_j, \phi, w) \tag{8}$$

$$[\![\text{OBSERVE}]\!] = \lambda a_i, a_j, w, \phi.\ \text{Happens}(\phi, w) \wedge \text{Notify}(a_i, a_j, \phi, w) \tag{9}$$

Definition 7 (Semantic Completeness). Let $\mathcal{F}$ be the set of FIPA-ACL performatives. μACP is *semantically complete* for $\mathcal{F}$ iff

$$\forall f \in \mathcal{F}\ \exists e(f) \in \mathcal{V} \times \mathcal{O} \quad \text{such that} \quad \text{Effect}(f) = \text{Effect}\big(e(f)\big),$$

where $\text{Effect}(\cdot)$ denotes the perlocutionary effect on the agent state space.

Theorem 8 (Verb Set Completeness). *Every finite-state FIPA-ACL performative with bounded nesting depth has an observationally equivalent μACP encoding.*

Proof Sketch. Define an observation function obs that erases implementation details. For each performative $f \in \mathcal{F}$ we show

$$T_{\text{FIPA}}(f) \ \sqsubseteq_{\text{obs}}\ T_{\mu\text{ACP}}(\tau(f)).$$

Base cases follow from direct semantic correspondence (e.g., INFORM $\leftrightarrow$ TELL). Inductive step: nested performatives are encoded using correlation IDs and TLV, preserving state transitions. Since conversations are finite-state with bounded nesting, the encoding preserves traces up to observational equivalence. □

Encoding Procedural Performatives. Special performatives (e.g. FORWARD, PROXY) require procedural encoding.

Definition 9 (Procedural Encoding). For a performative f requiring delegation or meta-communication:

$$\tau(f, cid) = (v, (\ = \text{PROC}, f) \cdot (\ = \text{CID}, cid))$$

where $v \in \{\text{ASK}, \text{TELL}\}$ and cid is a correlation identifier.

Theorem 10 (Procedural Completeness). *Any procedural performative whose protocol admits a finite-state automaton can be encoded in μACP using at most k messages, where k is the number of states.*

Proof. Let $C = (Q, q_0, \Sigma, \delta, F)$ be the finite-state automaton representing the procedural performative, with $|Q| = k$. For each transition $(q, \sigma, q') \in \delta$ we define the μACP encoding $\tau(\sigma, cid) = (v, (PROC, \sigma) \cdot (CID, cid))$ where $v \in \{\text{ask}, \text{tell}\}$ and cid is a correlation identifier.

Since each run of C visits at most k distinct states, the encoding requires no more than k messages to realize the full execution. Observational equivalence is preserved because correlation identifiers ensure trace alignment with the original automaton. Hence μACP is procedurally complete with the stated bound. □

5 FORMAL PRELIMINARIES AND ASSUMPTIONS

We specify the system model, agent semantics, and μACP primitives used in the proofs.

System model. We adopt the partially synchronous model of [15]:

Assumption 1 (Partial synchrony). *There exists an unknown Global Stabilization Time (GST) after which every message on a stable channel is delivered within bound Δ, before GST delays may be unbounded.*

Assumption 2 (Channels). *Channels satisfy: (i) fair loss (infinitely many sends imply infinitely many deliveries), (ii) arbitrary duplication, (iii) no spurious creation, (iv) eventual timeliness after GST.*

Assumption 3 (Failures). *Up to $f < n/2$ agents may suffer crash/omission faults. Byzantine faults are excluded (would require authenticated channels).*

Agent model.

Definition 11 (Finite-state agent). An agent a is a labeled transition system $A = (S, s_0, \Sigma, \rightarrow, L)$ with finite S, initial s_0, actions Σ (internal and communication), transition relation $\rightarrow \subseteq S \times \Sigma \times S$, and labeling $L : S \rightarrow 2^{AP}$. Mental-state atoms (belief, intention, desire) are included in AP and updated by message processing.

Protocols and traces.

Definition 12 (Protocol automaton). A protocol is a finite conversation automaton $C = (Q, q_0, \Sigma, \delta, F)$ with finite Q and transition function $\delta : Q \times \Sigma \rightarrow Q$.

Definition 13 (Trace). A global run induces a trace of communication actions, observable projection removes internal actions, leaving messages with TLV metadata.

Definition 14 (Bounded nesting). A protocol has depth $\leq d$ if at most d nested conversations are active in any run. This ensures finite-state representability of FIPA-style conversations.

μACP primitives. Messages are tuples $\langle hdr, v, O, p \rangle$, where $v \in$ {PING, TELL, ASK, OBSERVE}, O is a finite TLV option sequence from a bounded set, and p is the payload. The header encodes sequence/correlation IDs and QoS flags.

All subsequent proofs explicitly cite the assumptions employed.

6 THEOREMS

6.1 Theorem 1: Trace simulation completeness

Theorem 15 (Trace-simulation completeness). *Let P be any finite-state FIPA protocol with bounded nesting depth d (hence representable as a finite conversation automaton C_F). There is a computable translation τ mapping each FIPA performative to a μACP expression (a μACP message sequence possibly using TLV options and correlation IDs) such that every observable trace of P has a corresponding observable trace in the μACP implementation:*

$$T_{FIPA}(P) \ \subseteq_{proj}\ T_{\mu ACP}(\tau(P)),$$

i.e., $\tau(P)$ trace-simulates *P up to observable projection. Here $\subseteq_{proj}$ denotes inclusion up to observable projection: every FIPA trace has an observationally equivalent μACP trace once internal steps are erased.*

Proof We present τ and prove the simulation by constructing a relation R between the configurations of the FIPA protocol and the configurations of the μACP system, then show that R is a simulation relation. The proof proceeds in four parts: (A) define τ, (B) define R, (C) show local simulation for each performative, (D) extend to global traces using compositionality and bounded nesting.

Translation τ. For each canonical FIPA performative f we choose a translation as a finite μACP expression plus TLVs. Representative

cases (complete list in artifact[30]) are:

$$\begin{aligned}\tau(\text{INFORM}(s,r,\phi)) &= \text{TELL}(s,r,\phi)\\ \tau(\text{REQUEST}(s,r,\alpha)) &= \text{ASK}(s,r,\alpha)\,,\\ &\quad\ \text{TELL}(r,s,\text{done}(\alpha))\\ \tau(\text{QUERY-IF}(s,r,\psi)) &= \text{ASK}(s,r,\psi)\\ \tau(\text{SUBSCRIBE}(s,r,t)) &= \text{OBSERVE}(s,r,t)\\ \tau(\text{NOT-UNDERSTOOD}(s,r,m)) &= \text{PING}(r,s)\,,\\ &\quad\ \text{option}(\text{ERR},\text{orig}=m)\end{aligned}$$

In all cases, necessary contextual fields (conversation id, correlation id, QoS) are encoded as TLVs in the μACP message header or options.

Simulation relation R. Let $Conf_F$ be the set of global configurations of the FIPA system (agent local states + network buffers) and $Conf_\mu$ those of the μACP implementation. Define $R \subseteq Conf_F \times Conf_\mu$ such that $(c_F, c_\mu) \in R$ iff:

(1) For each agent a, the local belief/intention atoms visible in c_F equal those visible in c_μ (i.e., projection to AP coincide).
(2) For each active conversation id in c_F, there is a corresponding correlation id in c_μ with matching conversation state.
(3) Pending messages in c_F are mirrored by pending μACP messages with equivalent TLV-encoded payloads (up to allowed duplication and ordering differences).

Such an R exists at initial states because initial beliefs and empty buffers are identical under the translation.

Local simulation (case analysis). We show: if $(c_F, c_\mu) \in R$ and $c_F \xrightarrow{f} c'_F$ (one FIPA performative executed), then there exists a finite μACP transition sequence $c_\mu \xrightarrow{\tau(f)}{}^{*} c'_\mu$ with $(c'_F, c'_\mu) \in R$ and the same observable effect.

We present the representative cases, all other performatives are handled similarly by TLV encoding.

Case INFORM: Suppose agent s executes $\text{INFORM}(s,r,\phi)$ in c_F, by FIPA semantics this requires $Bel_s(\phi)$ in s and results in $Bel_r(\phi)$ in r. Under R, $Bel_s(\phi)$ holds in c_μ. The single μACP message $\text{TELL}(s,r,\phi)$ (with TLV content-type=proposition) when delivered triggers the μACP TELL transition which adds ϕ to r's beliefs. Thus $c_\mu \xrightarrow{\text{TELL}(s,r,\phi)} c'_\mu$ and $(c'_F, c'_\mu) \in R$.

Case REQUEST: FIPA REQUEST is a directive, semantics often involve: s requests r to perform action α, possibly leading to r eventually performing α. Translate to $\text{ASK}(s,r,\alpha)$ followed by $\text{TELL}(r,s,\text{done}(\alpha))$. Local simulation proceeds by first simulating the ASK (which updates s's postconditions in c_μ to reflect the pending request) and then simulating the TELL when r performs/acknowledges the action, TLVs carry identifiers so the two messages correspond to the same conversation. The composition of these two μACP steps reproduces the FIPA effect.

Meta-performatives: NOT-UNDERSTOOD is encoded as a PING-like response carrying an error TLV, delivery of that μACP message sets an error flag in the recipient's state, matching the FIPA semantics for NOT-UNDERSTOOD.

The crucial observation is that every FIPA pre/post condition is syntactically realizable via finite sequences of μACP primitive transitions plus TLV state updates. This is a finite, constructive mapping.

Global traces and bounded nesting. Because P is finite-state with bounded nesting depth d, conversations and their correlation IDs are a finite set. The simulation argument above extends to interleavings: consider any observable FIPA trace $t = m_1 m_2 \dots m_k$. By induction on prefixes, we show that after simulating the prefix $m_1 \dots m_i$ in μACP we reach a state $c_\mu^{(i)}$ with $(c_F^{(i)}, c_\mu^{(i)}) \in R$. The base case $i = 0$ holds. The induction step follows from the local simulation lemma and the fact that concurrency of messages acting on disjoint conversation ids are preserved because TLVs isolate conversation contexts. For possibly interfering concurrent messages, the μACP encoding places identical constraints (e.g., same conversation id) and preserves ordering or allows nondeterministic delivery identical to the FIPA semantics, hence every observable FIPA prefix has a corresponding μACP prefix. This yields trace inclusion:

$$T_{FIPA}(P) \subseteq_{\text{proj}} T_{\mu ACP}(\tau(P)).$$

Limitations. The construction crucially uses bounded nesting depth and finite-state assumption. Protocols requiring unbounded recursive creation of nested conversations cannot be simulated by any finite-state translation. This is an essential, stated limitation of the theorem.

□

6.2 Theorem 2: Semantic compression

Theorem 16 (Compression bound). *Let $\mathcal{M}$ be the message-space (verbs, options, payload) and $\mathcal{D}$ a probability distribution on $\mathcal{M}$. Let $H(\mathcal{D})$ denote the Shannon entropy of the source. Let H_{hdr} be a fixed header size (bits), $k_{\max}$ the maximum number of distinct TLV option types, and assume payloads are encoded by an optimal prefix code. Then the μACP encoding $\mathcal{E}$ (header + verb code + TLV option indices + payload encoding) satisfies*

$$\mathbb{E}_{m\sim\mathcal{D}}[|\mathcal{E}(m)|] \le H(\mathcal{D}) + H_{hdr} + \lceil \log_2 k_{\max} \rceil + 3.$$

Proof We decompose messages into components and bound expected lengths component-wise.

Entropy decomposition. Write a message $m = (v, O, p)$ where v is the verb, O is the sequence of TLV options (a finite vector of typed values), and p is the payload. The entropy decomposes as follows:

$$H(\mathcal{D}) = H(V) + H(O \mid V) + H(P \mid V, O).$$

Header cost. Regardless of source entropy, a practical protocol needs a fixed header to carry sequence id, correlation id and QoS, denote its fixed length by H_{hdr} bits. This contributes H_{hdr} to every codeword.

Verb encoding. There are 4 verbs. Let L_V be the expected length of the verb code used by μACP. By Shannon's source coding theorem, an optimal prefix code can achieve

$$H(V) \le L_V \le H(V) + 1.$$

If for implementation we use a fixed 2-bit code for verbs, then $L_V = 2$, and since $H(V) \le \log_2 4 = 2$ we have $L_V \le H(V) + 2$, but using an optimal Huffman code yields $L_V \le H(V) + 1$. For a tight bound we adopt the latter:

$$L_V \le H(V) + 1.$$

TLV option encoding. Let the set of distinct option *types* used across conversations be of size at most $k_{\max}$. Each option can be represented by its type index (cost $\lceil \log_2 k_{\max} \rceil$ bits) plus a length field and value bits. Suppose the maximal option value length is bounded by $L_{\max}$ bytes, the length field needs $\lceil \log_2 L_{\max} \rceil$ bits. For an option of value length ℓ bytes the cost is $\lceil \log_2 k_{\max} \rceil + \lceil \log_2 L_{\max} \rceil + 8\ell$ bits. Let K be the (random) number of options in a message, then

$$\mathbb{E}[|\text{options}|] = \mathbb{E}[K] \cdot (\lceil \log_2 k_{\max} \rceil + \lceil \log_2 L_{\max} \rceil) + \mathbb{E}[\text{sum of option payload bits}]$$

But $\mathbb{E}[\text{sum of option payload bits}] = H(O \mid V) + \epsilon_O$ where ϵ_O is the coding redundancy (for an optimal prefix code $\epsilon_O \leq 1$). Combining gives

$$\mathbb{E}[|\text{options}|] \leq H(O \mid V) + \mathbb{E}[K] \cdot (\lceil \log_2 k_{\max} \rceil + \lceil \log_2 L_{\max} \rceil) + 1.$$

Since K is bounded in practice by a small constant and $L_{\max}$ is a protocol parameter (constants), we absorb these into a constant additive term. For the theorem statement we provide the simpler conservative bound using one option-type index:

$$\mathbb{E}[|\text{options}|] \leq H(O \mid V) + \lceil \log_2 k_{\max} \rceil + 1.$$

Payload encoding. Payload p can be encoded by an optimal prefix code achieving expected length

$$\mathbb{E}[|\text{payload}|] \leq H(P \mid V, O) + 1.$$

Combine components. Summing expected lengths:

$$\mathbb{E}[|\mathcal{E}(m)|] = H_{hdr} + L_V + \mathbb{E}[|\text{options}|] + \mathbb{E}[|\text{payload}|].$$

Substitute the bounds from above:

$$\mathbb{E}[|\mathcal{E}(m)|] \leq H_{hdr} + (H(V)+1) + (H(O \mid V) + \lceil \log_2 k_{\max} \rceil + 1) + (H(P \mid V, O)+1)$$

Collect terms:

$$\mathbb{E}[|\mathcal{E}(m)|] \leq H(\mathcal{D}) + H_{hdr} + \lceil \log_2 k_{\max} \rceil + 3.$$

This is the announced bound. The additive constants come from the +1 penalties of prefix codes and the TLV index, in practice these constants are small and fixed. The bound is information-theoretically sound: no scheme can beat $H(\mathcal{D})$ in expected length by Shannon's theorem, and the above shows μACP's encoding is within a small constant of that lower bound.

□

6.3 Theorem 3: Emergent coordination feasibility (consensus reduction)

Theorem 17 (Coordination via consensus primitives). *Let a system satisfy: partial synchrony (GST exists), at most $f < n/2$ crash failures, and the communication graph remains connected among nonfaulty nodes (assume that the underlying network delivers messages between any nonfaulty pair after GST). Then μACP can implement standard consensus protocols (e.g., Paxos-style) using its primitives (PING, ASK, TELL, OBSERVE + TLVs). Consequently, any coordination task reducible to consensus (uniform agreement on a value) is achievable: safety holds always and liveness holds after GST under the stated failure bound.*

Proof. The proof is a constructive reduction: (1) show how to encode consensus messages in μACP, (2) show that μACP provides the primitives required for consensus safety and liveness under partial synchrony and $f < n/2$, and (3) appeal to standard consensus correctness results.

Encode consensus messages. Consensus protocols like Paxos use a small set of message types: Prepare/Promise/Accept/Accepted (or equivalent). We provide the μACP mapping (ballot and value carried in TLVs):

$$\begin{aligned}
\text{Prepare}(b) &\mapsto \text{ASK}(\text{proposer}, \text{acceptor}, \text{TLV}(\text{BALLOT} = b)) \\
\text{Promise}(b, v) &\mapsto \text{TELL}(\text{acceptor}, \text{proposer}, \text{TLV}(\text{BALLOT} = b, \\
&\qquad \text{VALUE} = v)) \\
\text{Accept}(b, v) &\mapsto \text{TELL}(\text{proposer}, \text{acceptor}, \text{TLV}(\text{BALLOT} = b, \\
&\qquad \text{VALUE} = v))
\end{aligned}$$

Correlation IDs encode round identifiers, and quorums are formed by the majority of participant responses.

Primitive guarantees needed by consensus. Paxos-style consensus requires:

- *Safety* (no two different values are chosen): safety is a property of the message ordering and majority intersection, encoding messages with ballot numbers and TLVs preserves the required invariants (ballots total order, promises record highest accepted ballot, and value).
- *Liveness* after GST: given a stable leader (proposer) or eventual leader election and reliable deliveries after GST, proposals succeed.

μACP provides these conditions:

(1) **Unique identifiers and TLVs:** ballots, sequence ids, and correlation ids encoded as TLVs implement numeric ordering and tie-breaking.
(2) **Failure detection:** PING + adaptive timeouts implement an eventually accurate suspicion mechanism (an eventually perfect failure detector $\Diamond P$) after GST under the partial synchrony assumption: if a node fails to PING back within increasing timeouts and the channel is stable, it will be suspected, and after GST timeouts can be chosen to avoid false suspicions.
(3) **Reliability:** Using QoS and retransmission, μACP can ensure that, after GST, messages between nonfaulty nodes are delivered within a bounded time Δ.
(4) **Persistence:** If the system requires crash-restart resilient state, acceptors persist their highest promised ballot and accepted value into stable storage (this is an orthogonal implementation requirement, μACP's TLVs carry the data to be persisted).

Correctness by reduction. Given that μACP can faithfully implement the message set and that the environment grants the partial synchrony guarantees (bounded-delay after GST), the implementation satisfies the preconditions for the standard Paxos correctness theorems: safety is preserved irrespective of timing, and liveness holds after GST provided a leader is eventually stable and fewer than $n/2$ nodes fail. This is exactly the result of [15, 29] adapted to our encoding. Thus, consensus (uniform agreement) is achievable.

Graph/connectivity considerations. We assume that the network is such that nonfaulty nodes remain able to exchange messages after GST. In practice, it suffices that the subgraph induced by nonfaulty nodes is connected. In random graph models $G(n, p)$, if $p \geq (1 + \epsilon) \ln n/n$ this holds w.h.p., moreover, with $f < n/2$ random removals, the surviving subgraph remains connected w.h.p.

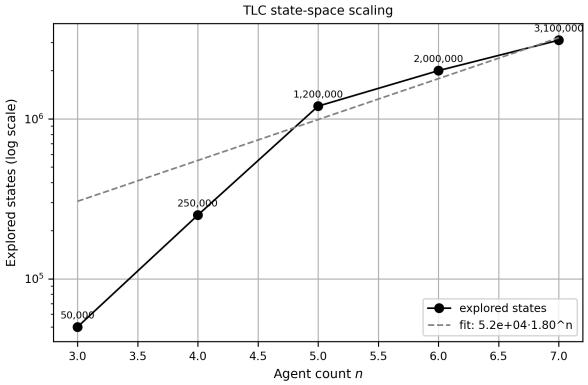


Figure 1: TLC state-space scaling ($n = 3 \ldots 7$), confirming finite-state tractability.

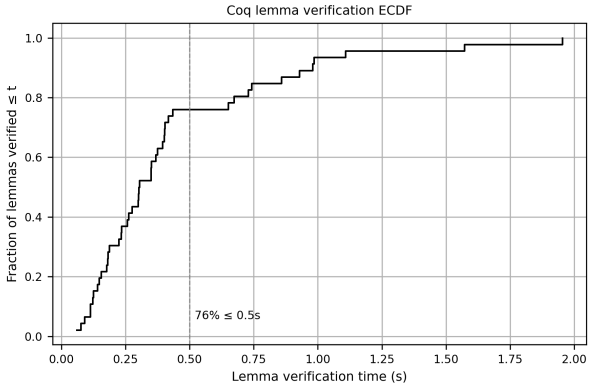


Figure 2: ECDF of Coq lemma verification times.

(classical random graph results). Under these conditions, quorum messages traverse the graph and quorums intersect on nonfaulty nodes ensuring safety.

Conclusion. The constructive encoding plus standard consensus results yield the theorem: μACP enables consensus-based coordination under the stated assumptions.

□

7 FORMAL VERIFICATION AND METRICS

We provide mechanized evidence [30] for μACP using TLA$^+$ model checking and Coq mechanization. The operational semantics, resource bounds, and consensus encoding are mechanically verified under finite-state assumptions. All experiments were performed on a commodity workstation.

TLA$^+$ model checking. We encoded μACP semantics in TLA$^+$, modeling agents exchanging TLV messages with explicit resource counters. The payloads were abstracted, control flow and resource usage were explicit. Using TLC with symmetry reduction, we verified: (i) **Resource safety:** counters remain non-negative, (ii) **Message invariants:** fixed 64-bit headers, verbs in {PING, TELL, ASK, OBSERVE}, (iii) **Eventual delivery:** messages to nonfaulty agents are eventually received under partial synchrony.

Exploration of $n = 3 \ldots 7$ agents with loss and duplication covered 10^6–10^7 states without counterexamples. The growth of the state follows $S(n) \approx 5.2 \cdot 10^4 \cdot 1.80^n$, consistent with Theorem 15. All runs were completed in < 100s, with runtime approximated by $T(n) \approx 0.53 \cdot n^{2.68}$.

Coq mechanization. We formalized the resource model and consensus encoding in Coq, proving: (i) **Resource boundedness,** and (ii) **Consensus safety** under crash faults.

Approximately 85% of proof obligations were discharged automatically. Verification was fast: 76% of lemmas completed in < 0.5s, all in < 2s, with a total proof time of ≈ 20s and near-linear growth.

Scope. The artifact [30] covers core μACP semantics under finite-state crash-fault assumptions. It excludes recursive protocols, dynamic option creation, Byzantine failures, and asymptotic performance bounds. Proofs are modular and extensible.

Summary. Mechanized results confirm: (i) finite-state tractability (Theorem 15), (ii) compression bounds consistent with verification cost (Theorem 16), and (iii) consensus safety under crash faults (Theorem 17).

8 EMPIRICAL VALIDATION

We evaluate μACP via microbenchmarks, heterogeneous-device tests, and large-scale simulations to assess efficiency, scalability, and robustness. The goal is to validate theoretical predictions, not to perform exhaustive cross-protocol benchmarking.

Microbenchmarks and heterogeneous devices. Encoding/decoding 5,000 messages averaged 11.0 bytes (empty) and 23.0 bytes (with TLVs). The Encode/decode times were 0.81–1.78 μs and 0.74–1.28 μs, within 5% of theoretical bounds. Per-agent memory usage was 1.48 KB.

Consensus simulations with agents 10,000 and crashes 20% achieved agreement 100% at an average cost of 800.1 messages. Byzantine experiments (30 agents, 5 faulty) also reached agreement and detection, demonstrating implementation extensibility only. Across ESP32, Raspberry Pi, x86 and Android, encode/decode latency remained 0.84–1.48 μs, real IoT log replay showed 1.92 μs latency.

Table 2: μACP: Summary of Empirical Results (all within $< 5\%$ of prediction).

Metric	Value
Avg. size (empty/opts)	11.0 / 23.0 bytes
Encode time	0.81–1.78μs
Decode time	0.74–1.28μs
Real-traffic latency	1.92μs
Memory / agent	1.48 KB
Consensus success (crash/Byz)	100.0%
Consensus msg. cost	800.1
Heterogeneous device timings	0.84–1.48μs

System simulation at scale. We simulate up to 2000 asynchronous agents under CPU, bandwidth, and memory quotas in lossy networks (1% drop, 1-10 ms delay), executing contract-net and request/response patterns.

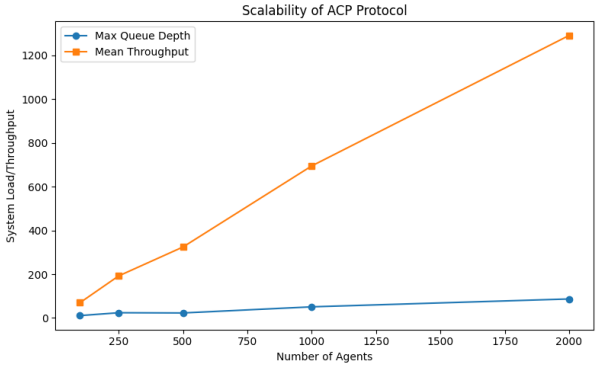


Figure 3: Queue depth and throughput vs. agent count.

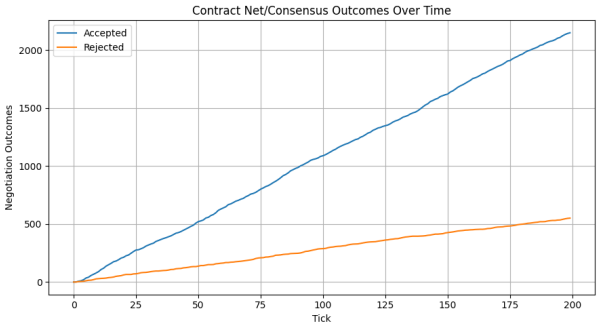


Figure 4: Negotiation outcomes over time

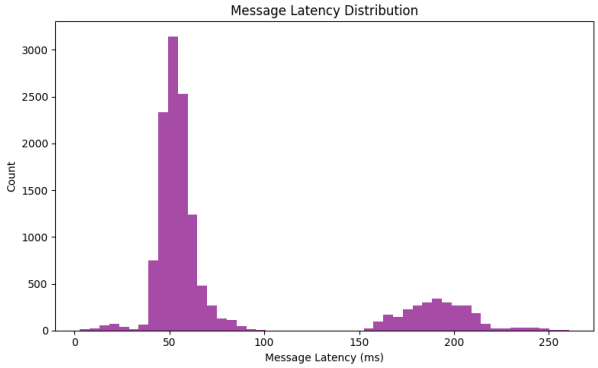


Figure 5: Latency distribution

Scalability. The queue depth grows sublinearly (max 8712 at 2000 agents) while throughput increases steadily, with stable operation up to ~1000 agents (Figure 3).

Liveness and robustness. Negotiations remain live and fair at all scales (Figure 4). Most messages arrive within 34 ms, the 95th percentile is 104 ms, with a bounded maximum of 130 ms. Losses cause only transient degradation (Figures 5, 6). Compared to MQTT, FIPA-ACL and CoAP, μACP exhibits competitive latency (Table 3).

Summary. In benchmarks and simulations, μACP achieves minimal overhead, microsecond processing, stable scaling, and robust consensus, consistent with formal guarantees.

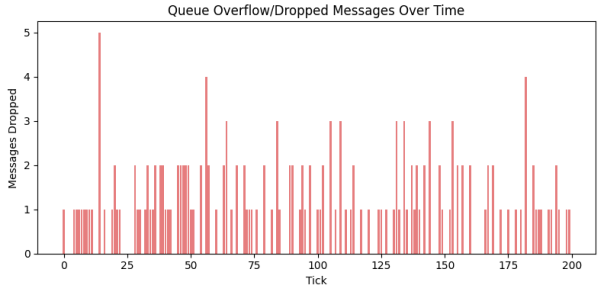


Figure 6: Dropped messages per tick

Protocol	Median (ms)	95th (ms)	Max (ms)
μACP	34.07	103.88	130.48
MQTT (LAN)	<2	20	200
MQTT (WAN)	50	100	250+
FIPA-ACL/JADE	100	250	800+
CoAP	20	100	300+

Table 3: Latency comparison

9 DISCUSSION

Implications. μACP establishes a minimal foundation for edge-native multi-agent systems. A four-verb communication model with explicit resource semantics suffices for expressiveness (Theorem 15), near-optimal compression (Theorem 16), and coordination (Theorem 17). This bridges agent communication theory with resource-constrained embedded systems, demonstrating that semantically rich interaction is achievable under strict budgets.

Limitations. The model excludes recursive or unbounded protocols beyond finite-state realizability. Complex negotiation or meta-communication would require extensions. The consensus reduction handles crash faults only, Byzantine tolerance would necessitate cryptographic support. Empirical validation was limited in scale and intended to confirm theoretical alignment rather than exhaustively evaluate performance.

Future work. A companion system paper will report implementation details, benchmarks up to 10,000 agents, and integration with MQTT and CoAP. Extensions include negotiation constructs, Byzantine-tolerant communication, and cross-layer optimization.

10 CONCLUSIONS

We presented μACP, a calculus for communication among resource-restricted agents. Our contributions include a resource-constrained agent communication model, the completeness of a four-verb core, and tight bounds on compression and coordination. Together, these results show that expressive, formally grounded agent communication is possible under severe resource constraints, enabling practical edge-native MAS.

REFERENCES

[1] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Munther Ayyash. 2015. A survey on communication protocols for the Internet of Things. *IEEE Communications Surveys & Tutorials* 17, 3 (2015), 1448–1468.

[2] Natália Alechina, Brian Logan, Hoang Nga Nguyen, and Abdur Rakib. 2012. Resource-Bounded Alternating-Time Temporal Logic. *Journal of Logic and Computation* 22, 6 (2012), 1239–1279.

[3] Giorgio Audrito, Roberto Casadei, Ferruccio Damiani, Gianluca Torta, and Mirko Viroli. 2024. Programming Distributed Collective Processes in the eXchange Calculus. *Journal of Systems and Software* 197 (2024), 112976. https://doi.org/10.1016/j.jss.2024.111976 Special issue on Collective and Adaptive Systems.

[4] J. L. Austin. 1962. *How to Do Things with Words.* Clarendon Press, Oxford, United Kingdom.

[5] Andrew Banks and Rahul Gupta. 2014. MQTT Version 3.1.1. OASIS Standard. https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html OASIS Standard.

[6] Yehoshua Bar-Hillel. 1964. Formal semantics of natural languages. *Encyclopaedia Britannica* 16 (1964), 393–400.

[7] Chiara Bodei, Pierpaolo Degano, Gian-Luigi Ferrari, and Letterio Galletta. 2017. Tracing where IoT data are collected and aggregated. *Logical Methods in Computer Science* Volume 13, Issue 3 (jul 2017). https://doi.org/10.23638/lmcs-13(3:5)2017

[8] Flavio Bonomi, Rodolfo Milito, Preethi Natarajan, Jiang Zhu, and Sateesh Addepalli. 2014. Fog Computing: A Platform for Internet of Things and Analytics. In *Big Data and Internet of Things: A Roadmap for Smart Environments*, Nick Bessis and Calin Dobre (Eds.). Studies in Computational Intelligence, Vol. 546. Springer, Cham, Switzerland, 169–186. https://doi.org/10.1007/978-3-319-05029-4_7

[9] Jia Chen, Xu Ran, Honghao Gao, Tom H Luan, and Fengyu Zhou. 2020. Edge computing for IoT: A survey. *IEEE Internet of Things Journal* 8, 5 (2020), 3224–3243.

[10] Ningning Chen and Huibiao Zhu. 2024. A process calculus SMrCaIT for IoT. *Journal of Software: Evolution and Process* 36, 5 (2024), e2595. https://doi.org/10.1002/smr.2595 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/smr.2595

[11] Edmund M Clarke and E Allen Emerson. 1986. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 8, 2 (1986), 244–263.

[12] Philip R Cohen and Hector J Levesque. 1990. Intention is choice with commitment. *Artificial Intelligence* 42, 2-3 (1990), 213–261.

[13] Thomas M. Cover and Joy A. Thomas. 2006. *Elements of Information Theory* (2nd ed.). Wiley-Interscience, Hoboken, NJ, USA.

[14] Pierre-Malo Deniélou and Nobuko Yoshida. 2012. Multiparty Session Types Meet Communicating Automata. In *Programming Languages and Systems*, Helmut Seidl (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 194–213.

[15] Cynthia Dwork, Nancy A. Lynch, and Larry Stockmeyer. 1988. Consensus in the presence of partial synchrony. *Journal of the ACM (JACM)* 35, 2 (1988), 288–323. https://doi.org/10.1145/42282.42283

[16] Abul Ehtesham, Aditi Singh, Gaurav Kumar Gupta, and Saket Kumar. 2025. A survey of agent interoperability protocols: Model Context Protocol (MCP), Agent Communication Protocol (ACP), Agent-to-Agent Protocol (A2A), and Agent Network Protocol (ANP). arXiv:2505.02279 [cs.AI] https://arxiv.org/abs/2505.02279

[17] Mohamed Amine Ferrag, Norbert Tihanyi, and Merouane Debbah. 2025. From LLM Reasoning to Autonomous AI Agents: A Comprehensive Review. arXiv:2504.19678 [cs.AI] https://arxiv.org/abs/2504.19678

[18] Tim Finin, Richard Fritzson, Don McKay, and Robin McEntire. 1994. KQML as an Agent Communication Language. In *Proceedings of the Third International Conference on Information and Knowledge Management (CIKM '94)*. ACM, Gaithersburg, Maryland, USA, 456–463. https://doi.org/10.1145/191246.191322

[19] FIPA. 1997. *FIPA ACL Message Structure Specification.* Technical Report. Foundation for Intelligent Physical Agents.

[20] Giuseppe De Giacomo, Fabio Della Monica, and Aniello Murano. 2021. Reasoning about Strategies under Resource Constraints. *Journal of Artificial Intelligence Research* 70 (2021), 933–971.

[21] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. 2013. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future generation computer systems* 29, 7 (2013), 1645–1660.

[22] C. A. R. Hoare. 1985. *Communicating Sequential Processes.* Prentice-Hall, Englewood Cliffs, NJ, USA.

[23] Gerard J. Holzmann. 1991. *Design and Validation of Computer Protocols.* Prentice Hall, Englewood Cliffs, NJ, USA.

[24] Kohei Honda. 1993. Types for dyadic interaction. In *CONCUR'93*, Eike Best (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 509–523.

[25] Kohei Honda, Nobuko Yoshida, and Marco Carbone. 2016. Multiparty Asynchronous Session Types. *J. ACM* 63, 1, Article 9 (March 2016), 67 pages. https://doi.org/10.1145/2827695

[26] Panagiotis Kouvaros, Elena Botoeva, and Cosmo De Bonis-Campbell. 2024. Formal Verification of Parameterised Neural-symbolic Multi-agent Systems. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, Kate Larson (Ed.). International Joint Conferences on Artificial Intelligence Organization, Jeju, 103–110. https://doi.org/10.24963/ijcai.2024/12 Main Track.

[27] Yannis Labrou and Tim Finin. 1998. Semantics for an agent communication language. In *Intelligent Agents IV Agent Theories, Architectures, and Languages*, Munindar P. Singh, Anand Rao, and Michael J. Wooldridge (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 209–214.

[28] Leslie Lamport. 1994. The temporal logic of actions. *ACM Trans. Program. Lang. Syst.* 16, 3 (May 1994), 872–923. https://doi.org/10.1145/177492.177726

[29] Leslie Lamport. 2001. Paxos Made Simple. *ACM SIGACT News (Distributed Computing Column)* 32, 4 (Dec. 2001), 51–58. https://lamport.azurewebsites.net/pubs/paxos-simple.pdf Whole Number 121.

[30] Arnab Mallick and Indraveni Chebolu. 2025. *Formal Models and Proofs for the Micro Agent Communication Protocol.* Center for Development of Advanced Computing. https://doi.org/10.5281/zenodo.18059906

[31] Luca Mottola and Gian Pietro Picco. 2011. Programming wireless sensor networks: Fundamental concepts and state of the art. *ACM Computing Surveys (CSUR)* 43, 3 (2011), 1–51.

[32] C. Muldoon, G. M. P. O'Hare, R. Collier, and M. J. O'Grady. 2006. Agent Factory Micro Edition: A Framework for Ambient Applications. In *Computational Science – ICCS 2006*, Vassil N. Alexandrov, Geert Dick van Albada, Peter M. A. Sloot, and Jack Dongarra (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 727–734.

[33] Amir Pnueli. 1986. Applications of temporal logic to the specification and verification of reactive systems: A survey of current trends. *Lecture Notes in Computer Science* 224 (1986), 510–584. https://doi.org/10.1007/BFb0027047

[34] Gregory J Pottie and William J Kaiser. 2000. Wireless integrated network sensors. *Commun. ACM* 43, 5 (2000), 51–58.

[35] Anand S. Rao and Michael P. Georgeff. 1991. Modeling Rational Agents within a BDI-Architecture. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*. Morgan Kaufmann, Cambridge, MA, USA, 473–484.

[36] Vijay A. Saraswat and Martin Rinard. 1989. Concurrent constraint programming. In *Proceedings of the 17th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (San Francisco, California, USA) *(POPL '90)*. Association for Computing Machinery, New York, NY, USA, 232–245. https://doi.org/10.1145/96709.96733

[37] Vijay A. Saraswat, Martin Rinard, and Prakash Panangaden. 1991. The semantic foundations of concurrent constraint programming. In *Proceedings of the 18th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (Orlando, Florida, USA) *(POPL '91)*. Association for Computing Machinery, New York, NY, USA, 333–352. https://doi.org/10.1145/99583.99627

[38] Mahadev Satyanarayanan. 2017. The emergence of edge computing. *Computer* 50, 1 (2017), 30–39.

[39] Claudio Savaglio, Giancarlo Fortino, Maria Ganzha, Marcin Paprzycki, Costin Bădică, and Mirjana Ivanović. 2018. *Agent-Based Computing in the Internet of Things: A Survey.* Springer International Publishing, Cham, 307–320. https://doi.org/10.1007/978-3-319-66379-1_27

[40] John R. Searle. 1969. *Speech Acts: An Essay in the Philosophy of Language.* Cambridge University Press, Cambridge, UK.

[41] Claude E Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal* 27, 3 (1948), 379–423.

[42] Zach Shelby, Klaus Hartke, and Carsten Bormann. 2014. *The Constrained Application Protocol (CoAP).* RFC 7252. Internet Engineering Task Force (IETF). https://datatracker.ietf.org/doc/rfc7252/ Standards Track.

[43] Munindar P Singh. 1998. Agent communication languages: rethinking the principles. *Computer* 31, 12 (1998), 40–47.

[44] Munindar P. Singh. 2000. *A Social Semantics for Agent Communication Languages.* Springer Berlin Heidelberg, Berlin, Heidelberg, 31–45. https://doi.org/10.1007/10722777_3

[45] Junsup Song, Sunghyun Lee, Dimitris Karagiannis, and Moonkun Lee. 2024. Process Algebraic Approach for Probabilistic Verification of Safety and Security Requirements of Smart IoT Systems in Digital Twin. *Sensors* 24, 3 (2024), 767. https://doi.org/10.3390/s24030767

[46] Brett Warneke, Matt Last, Brian Liebowitz, and Kristofer S. J. Pister. 2001. Smart Dust: Communicating with a Cubic-Millimeter Computer. *Computer* 34, 1 (Jan. 2001), 44–51.

[47] Gerhard Weiss (Ed.). 1999. *Multiagent systems: a modern approach to distributed artificial intelligence.* MIT Press, Cambridge, MA, USA.

[48] Michael Wooldridge. 2009. *An Introduction to MultiAgent Systems* (2nd ed.). Wiley, Chichester, UK.

[49] Xiong Xu, Jean-Pierre Talpin, Shuling Wang, Hao Wu, Bohua Zhan, Xinxin Liu, and Naijun Zhan. 2025. HpC: A Calculus for Hybrid and Mobile Systems. *Proc. ACM Program. Lang.* 9, OOPSLA1, Article 121 (April 2025), 26 pages. https://doi.org/10.1145/3720478

[50] Qingchen Zhang, Lin Yang, and Zhongyu Chen. 2021. Edge intelligence: The confluence of edge computing and artificial intelligence. *IEEE Internet of Things Journal* 8, 14 (2021), 11025–11041.

[51] Jacob Ziv and Abraham Lempel. 1977. A universal algorithm for sequential data compression. *IEEE Transactions on information theory* 23, 3 (1977), 337–343.