Evaluating the boundary and emissions for Agentic AI use cases

Discussion Draft for Review and Consensus

This document is intended to drive discussions and develop shared understanding across GSF members. It is not a final approved position.

1. Introduction

Agentic AI refers to systems composed of autonomous or semi-autonomous agents that operate independently or collaboratively to achieve tasks. These agents typically coordinate and reason over a sequence of actions, invoking models, tools, and services based on goals, context, and memory.

Unlike conventional single-model applications, Agentic AI systems introduce additional architectural complexity by involving:

- **Multiple agents**, often specialized for tasks such as planning, retrieval, summarization, or interaction.
- **Calls to multiple LLMs**, which may be hosted internally or accessed via external APIs (e.g., OpenAI, Anthropic, Google Cloud).
- **Tool use**, including internal APIs, SaaS platforms, or open web services.
- **Orchestration layers**, such as a central agent planner or emerging infrastructure standards like **Anthropic's Model Context Protocol (MCP)**, which facilitate context sharing and routing across agents.

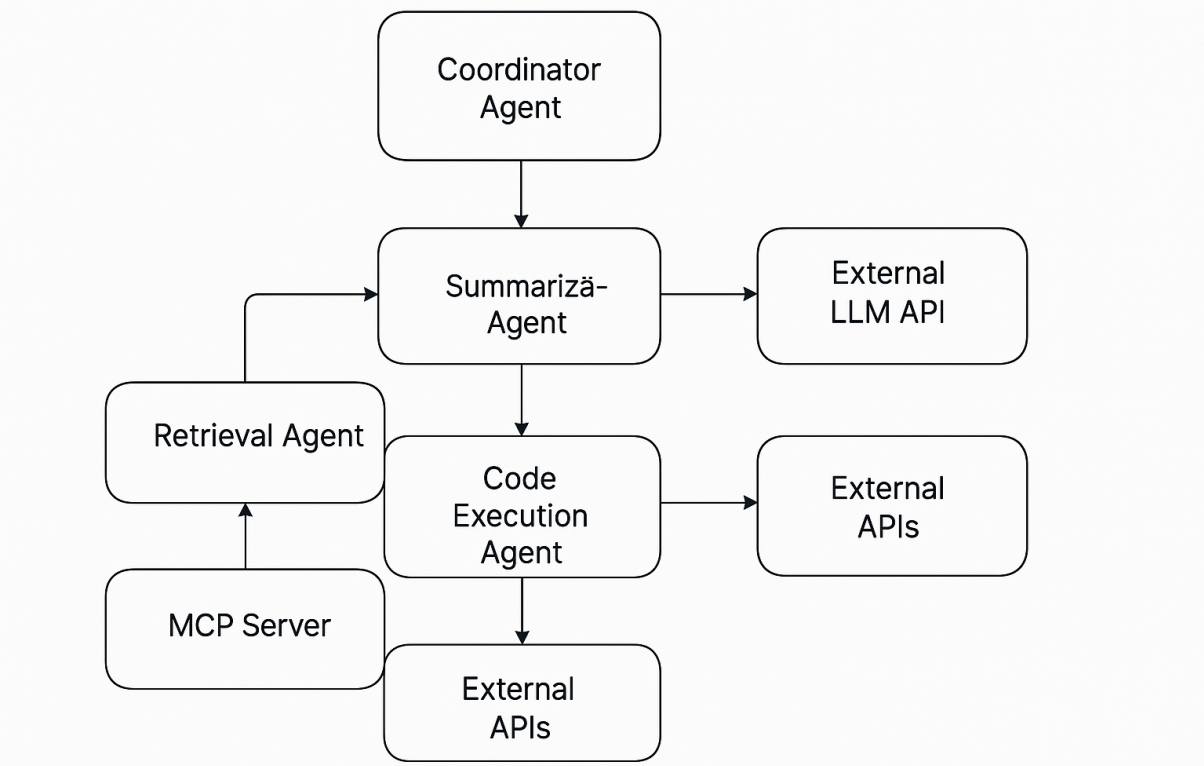
These systems execute across **distributed infrastructure** and span both **internal enterprise components** and **external third-party services**. The emissions resulting from such workflows originate from multiple sources, some of which are directly controlled by the deploying organization, while others are not

2. Scenario: Multi-Agent Research Assistant with Code Execution

An enterprise develops a **Multi-Agent Research Assistant** to support internal research and literature analysis workflows. The system consists of several specialized agents, tools, and orchestration components working together:

- **Coordinator Agent**: Plans the task sequence, assigns subtasks to specialized agents, and handles retries and feedback.
- **Retrieval Agent**: Queries internal repositories and external research databases (e.g., PubMed, arXiv) to gather relevant content.
- **Summarization Agent**:

    - Calls an **external LLM API** (e.g., Claude by Anthropic) to summarize external content.
    - For internal document summarization, a **self-hosted LLM** is deployed on the enterprise's GPU-based infrastructure.
- **Code Execution Agent**:
  Executes Python-based data processing routines using an **MCP-compliant code execution server** (e.g *mcp-run-python*).
  The server can be:
    - **Self-hosted** within the enterprise, or
    - **Accessed as a managed external service**, depending on data sensitivity and resource availability.
- **MCP Coordination Server**:
  Used to manage **context routing, memory sharing, and model-agent communication** across the workflow.
  The MCP server may also be:
    - **Hosted internally** by the enterprise as part of its orchestration layer, or
    - **Hosted externally** by a third-party vendor as a managed coordination platform.
- **External APIs**:
  The system also integrates with third-party APIs for semantic enrichment, content classification, and similarity scoring.



Illustrative Example: Multi-Agent Research Assistant with Code Execution

This example showcases the architecture of a real-world Agentic AI system, combining:

- **Local and remote model execution**

- **Tool invocation through agents**
- **MCP-enabled memory and routing infrastructure**
- **Hybrid deployment models** for LLMs, tools, and orchestration servers

Such a system spans multiple boundary types, with agents operating across internal compute, external APIs, and shared coordination layers. Whether components are **self-hosted or externally managed** directly influences their inclusion in SCI calculations — a topic explored in the following sections.

2.1 Key Question for Consideration

This leads to a key question in the context of SCI standardization for Agentic AI systems:

How should emissions be allocated across distributed agentic workflows, where agents invoke models, tools, and shared coordination layers that span both enterprise-managed and third-party infrastructure?

While prior discussions around LLM API usage established that:

- **Inference emissions from external LLM APIs** are **in-scope**, and
- **Training emissions** are **not included by default** (unless disclosed and allocated)

Agentic AI introduces additional layers of complexity, such as:

- **Tool execution** initiated by agents, including third-party APIs or SaaS platforms that may perform significant computation outside enterprise control.
- **Middleware orchestration layers**, such as **MCP servers**, which may handle prompt routing, memory management, or context sharing — and could be either **self-hosted** or **externally managed**.

This introduces a broader boundary consideration for SCI:

- Should emissions be scoped only to components under **enterprise control** (e.g., self-hosted LLMs, internal APIs, agent code), or
- Should a **portion of emissions** from **external tool execution or coordination infrastructure** be included when used as part of a broader system workflow?

This document builds on existing SCI guidance, presents both treatment paths for Agentic AI systems, and outlines associated challenges and recommendations to support consensus-building within the working group.

3. Boundary Classification and Component Breakdown

Agentic AI systems span a range of infrastructure layers — including agents, models, tools, orchestration servers, and memory components — many of which may be internally or externally hosted. For Software Carbon Intensity (SCI) accounting, it is important to define which components fall **within the boundary** of the consuming system (in-scope) and which are **outside** that boundary (out-of-scope).

This classification follows SCI's core principles:

- **Control**: Whether the emissions can be directly influenced by the consuming entity.
- **Observability**: Whether usage and energy data are available or measurable.
- **Attribution**: Whether emissions can be credibly attributed to the system or agent invoking the service.

Component Boundary Mapping

Component	Hosting Option	SCI Boundary Treatment
Coordinator Agent	Internal	✅ **In-scope** — Emissions from orchestration, planning, and task routing within enterprise control.
Retrieval Agent	Internal	✅ **In-scope** — If querying internal systems or calling APIs hosted by the enterprise.
Summarization Agent (External LLM)	API-based external model	✅ **Inference: In-scope** (measurable token usage) ❌ **Training: Out-of-scope by default** unless disclosed.
Summarization Agent (Self-hosted LLM)	Internal infrastructure	✅ **In-scope** — Includes inference emissions and, optionally, training emissions if performed internally.
Code Execution Agent (Self-hosted)	Internal	✅ **In-scope** — Emissions from script execution on MCP-compatible infrastructure controlled by the enterprise.
Code Execution Agent (Managed Service)	External	❌ **Out-of-scope** — Excluded unless usage data and emissions are disclosed.
MCP Coordination Server (Self-hosted)	Internal	✅ **In-scope** — Coordination, memory management, and routing emissions are included.
MCP Coordination Server (External)	Third-party provider	❌ **Out-of-scope** — Excluded unless transparent reporting or SLAs include emissions metrics.
Memory or Context Store	Internal or external	Depends — Internal vector stores are in-scope; managed services are out-of-scope unless emissions data is shared.
External APIs / Tools	External	❌ **Out-of-scope** — Third-party tool calls are excluded by default. May be included in extended boundary if usage and emissions data are disclosed.

Boundary Complexity in Multi-Function Components

Some components in Agentic AI workflows serve **multiple functions** or may be deployed in **hybrid configurations**, making boundary classification less straightforward. Below are examples and how to approach them:

1. MCP Servers (Model Context Protocol)

MCP servers can serve **more than one purpose**:

- **Coordination & Routing**: Handling prompt orchestration, agent messaging, and context passing.
- **Memory Management**: Hosting shared context, previous responses, or vector embeddings.
- **Execution Support**: In some implementations, MCP servers may even manage lightweight code execution or facilitate tool plugins.

Boundary guidance:

- If **self-hosted**, all emissions related to coordination, memory, or lightweight execution are **in-scope**.
- If **externally managed**, emissions are **out-of-scope by default**, unless the service provides transparent emissions data or is contractually included.

2. Cross-Organizational Agent Collaboration

In some advanced deployments, an agent in one enterprise may trigger another agent or tool in a **different organization's system** (e.g., a public Agent-as-a-Service endpoint).

Boundary guidance:

- Emissions should be attributed **only within the deploying entity's boundary**, unless there is **joint measurement** or **shared visibility** into the downstream agent's infrastructure and execution footprint.

3. Shared or Hybrid Memory Stores

Memory modules (e.g., for storing conversation history, embeddings, or knowledge graphs) can be:

- **Embedded** in the orchestration platform (e.g., Redis, Pinecone running on-premises)
- **Externally hosted** as part of a managed AI stack

Boundary guidance:

- If the memory store is **within the infrastructure of the consuming entity**, emissions are **in-scope**.
- If **outsourced to a third-party service** (e.g., vector DB-as-a-Service), emissions are **out-of-scope** by default unless the provider offers emissions tracking.

General Principle

For any component performing **multi-functional or multi-tenant roles**, the SCI boundary should be evaluated based on:

- **Who operates it**

- **Who controls the data flow**
- **Whether usage and emissions data are observable**

4. Challenges in SCI Accounting for Agentic AI Systems

Agentic AI introduces a dynamic execution environment with multiple agents, shared memory, orchestration layers, and third-party services. Unlike traditional AI workloads, where control and infrastructure boundaries are more clearly defined, agentic workflows span diverse components with varying levels of visibility, attribution, and control.

This section outlines key challenges in applying Software Carbon Intensity (SCI) principles to such systems.

4.1 Distributed Execution and Responsibility

In Agentic AI systems:

- Tasks are delegated across multiple agents, often asynchronously.
- Each agent may call models, tools, or services — internally or externally hosted.
- Execution responsibility is **distributed**, making it difficult to define **where emissions originate** and **who is accountable**.

Without centralized orchestration visibility, tracing emissions back to the initiating application can be difficult, especially when services operate across different ownership domains.

4.2 Tool and Service Diversity

Agents may invoke:

- External APIs for search, summarization, or classification
- SaaS platforms with opaque backends
- Self-hosted or cloud-based execution environments

Each tool may have different:

- Compute intensities
- Energy profiles
- Reporting practices (or lack thereof)

SCI accounting becomes inconsistent when some services provide emissions data while others don’t — particularly when agents rely on dynamic tool selection at runtime.

4.3 External Coordination and Memory Infrastructure

With the rise of protocols like **Anthropic’s MCP**, new infrastructure components are introduced that handle:

- Prompt routing

- Agent-to-agent communication
- Shared memory management
- Lightweight task execution

These may be hosted internally or externally, but emissions from these coordination layers are **non-trivial** and can affect the SCI outcome significantly.

The challenge lies in determining:

- Whether such infrastructure is within the system boundary
- Whether emissions from shared use can be isolated and allocated

4.4 Lack of Uniform Observability

Unlike traditional software systems where telemetry and energy usage can be integrated at the infrastructure layer, agentic workflows face several observability gaps:

- Token-level usage across multiple LLMs
- Tool execution time and compute consumption
- Cross-agent handoffs and retries
- Latency and idle emissions of orchestrators

Without standardized instrumentation or visibility across components, emissions estimation must rely on approximations, increasing the margin of uncertainty.

4.5 Versioning, Routing, and Dynamic Behaviour

Agents may:

- Route tasks to different LLMs or tools based on performance, cost, or availability.
- Fall back to alternate providers or internal models.
- Dynamically evolve their plans based on retrieved context or feedback.

This behaviour introduces challenges in attributing emissions to:

- Specific model versions
- Specific agents
- Specific workflow paths

Traditional SCI models assume more stable and linear execution flows, which may not hold true in agent-driven systems.

5. Opportunities for Standardization

While Agentic AI systems introduce significant complexity for Software Carbon Intensity (SCI) accounting, they also offer a unique opportunity to define the next generation of sustainable software practices. Addressing the gaps outlined in Section 4 will require collaboration across model providers, tool developers, infrastructure operators, and standards bodies.

This section outlines key areas where standardization can enhance SCI applicability and reliability in Agentic AI use cases.

5.1 Model and Tool Emissions Disclosure

Standardized reporting of emissions associated with:

- **LLM training and inference**
- **Tool execution workloads**
- **Hosted orchestration and MCP infrastructure**

...would enable more accurate attribution of emissions in workflows that depend on multiple third-party services.

Disclosures should include:

- Energy usage (kWh)
- Carbon intensity (gCO_2e/kWh)
- Token- or API-level usage metrics
- Model version identifiers

Such transparency would allow downstream consumers to optionally include upstream emissions in an extended boundary.

5.2 Shared Infrastructure Tagging and Attribution

For components like **MCP servers** and **vector memory stores** used by multiple agents or applications:

- Emissions can be **tagged at the request or session level**
- Allocation methods could be standardized based on request volume, data size, or execution time

This supports fair sharing of emissions across tenants while preserving performance and architectural flexibility.

5.3 Agent Telemetry Standards

Developing a telemetry framework for Agentic AI systems that includes:

- **Per-agent energy tracking**
- **Tool invocation logs with timestamps and usage**
- **LLM call routing details with token counts and latency**

...would enable reproducible and verifiable SCI calculations, especially for workflows involving retries, fallback mechanisms, or dynamic planning.

5.4 Extended Boundary Labels in SCI Reporting

Introduce a standardized label or marker in SCI reports to distinguish:

- **Default SCI boundary**: Inference and orchestration under direct control
- **Extended SCI boundary**: Includes optionally reported upstream emissions (e.g., training, third-party tool execution)

This would preserve comparability across applications while rewarding transparent ecosystems.

5.5 Alignment with Emerging Protocols (e.g., MCP)

As coordination standards like **Anthropic's Model Context Protocol (MCP)** gain adoption:

- Emission-aware extensions could be proposed to enable usage tagging, power modeling, or carbon-aware routing
- Middleware could integrate carbon budget constraints into agent decision-making

This opens pathways for not only reporting emissions, but **actively optimizing them during agent execution**.

These standardization efforts will enable the SCI framework to scale with the growing complexity of intelligent agent-based systems while promoting transparency, fairness, and actionability.

6. Recommendation Framework

Based on the characteristics of Agentic AI systems and the SCI boundary principles of control, observability, and attribution, the following dual-path recommendation is proposed:

6.1 Default SCI Boundary for Agentic AI

For most implementations, SCI calculations should include **only emissions from components under the control of the consuming entity**. This keeps the SCI practical, measurable, and aligned with the intent of actionable sustainability reporting.

✅ *Included:*

- Emissions from self-hosted agents, internal orchestration logic, and internal tools
- Inference emissions from external LLMs (if usage is measurable and tokens/API calls are known)

❌ *Excluded:*

- Training emissions for externally hosted LLMs (unless disclosed)

- Emissions from external tools, managed MCP servers, and other third-party services where observability is limited

This boundary is suitable for **enterprise-grade implementations** where data and infrastructure are partially managed and measured internally.

6.2 Extended SCI Boundary (Optional Inclusion)

Where data is available, or where model/tool providers offer standardized disclosures, consumers may optionally include upstream emissions to reflect **full system impact**. This is especially applicable when:

- MCP or tool infrastructure is co-managed or usage is contractually tied to emissions reporting
- Providers disclose emissions from training or execution workloads in usable formats
- The system is part of a sustainability audit, ESG reporting, or regulated disclosure context

To ensure clarity, any extended boundary SCI score should include an annotation indicating:

“Includes disclosed upstream emissions for models, tools, or shared coordination infrastructure”

6.3 Summary Table

Component Type	Default SCI Treatment	Extended Boundary Option
Self-hosted Agents & Tools	✅ In-scope	✅ In-scope
External LLMs (API-based)	✅ Inference only	🟡 Training: Only if disclosed *and* standardized amortization is available
Self-hosted LLMs	✅ Inference and training	✅ In-scope
Tool APIs (External)	❌ Out-of-scope	✅ If usage/emissions disclosed
MCP Servers (Self-hosted)	✅ In-scope	✅ In-scope
MCP Servers (Managed)	❌ Out-of-scope	🟡 Only if provider shares usage data *and* supports shared emission allocation
Shared Memory (Internal)	✅ In-scope	✅ In-scope
Shared Memory (External)	❌ Out-of-scope	🟡 Only if usage, total load, and emissions data are disclosed and allocatable

🟡 “Shared Memory (External)” in Extended Boundary

Externally hosted memory components, such as **managed vector databases** or **contextual memory services**, are often:

- Shared across customers
- Operated as black-box services
- Billed on storage or query volume, with no exposure to actual compute or energy usage

For these to be included in the SCI boundary:

- Providers must disclose **total emissions for the service** (`E_total`)
- Provide the **total operational usage** (`N_total`)
- Allow consuming systems to determine their **usage share** (`N_entity`)
- Apply a fair **allocation method** (e.g., emissions per vector query, per session, or GB-month)

Without these, emissions from external memory components remain **out-of-scope by default**, and **conditionally includable** if appropriate disclosures are available.

6.4 Guiding Principle for Agentic Systems

SCI for Agentic AI should begin with what can be measured and influenced directly.
As transparency improves, boundary extensions can follow — with clear labeling — to support deeper accountability and system-wide sustainability awareness.

7. Conclusion

Agentic AI systems represent a new class of intelligent software — dynamic, distributed, and increasingly reliant on orchestration across models, tools, and memory. While they offer powerful capabilities, they also challenge traditional boundaries of emissions accounting.

This document has presented a structured approach to defining SCI boundaries for Agentic AI workflows, grounded in the principles of **control**, **observability**, and **attribution**. It outlines:

- How different components (e.g., self-hosted LLMs, external APIs, MCP servers) should be treated within SCI boundaries
- The challenges of distributed execution, opaque tooling, and shared infrastructure
- Opportunities for standardization in telemetry, emissions disclosure, and agent coordination
- A dual-boundary recommendation framework to balance practicality with future extensibility

As Agentic AI systems continue to evolve, the SCI standard has an opportunity to lead the way in responsible compute accountability. Ongoing collaboration between model providers,

tool developers, AI platform vendors, and sustainability working groups will be essential to ensure that emissions data becomes more accessible, standardized, and actionable.

This document is intended to drive further discussion and alignment within the SCI and broader sustainability community. It does not represent a finalized standard, but provides a foundation for shared understanding and next-step consensus.

Revision History

Version	Date	Author	Description
v0.1	9 Apr 2025	Navveen Balani	Initial draft for discussion