From Stone and Thread to Shared Intelligence

QR code
Fixed W0-007 release

Part III — The Collaboration Model: People, Tools, Resources and Evidence

A Credit-Card-Sized Starting Point for Local Making and Global Participation

Benny (Yan) Liao — Founder, HSIMC
W0-007 · P3-C English Master · 11 November 2026

Human Authority · Local Capability · Global Participation
No One Left Behind in the GenAI Physical Era

Part III at a glance

Part I defined the opportunity: GenAI can produce plans quickly, but a physical outcome still requires materials, machines, people, time, authority and evidence.

Part II showed that participation can begin with what people already know and already have. A mineral offcut, a sewing machine, an embroidery machine, a printer, copper foil, a camera, a jig or a local workshop can become part of an intelligent production network when a function, its conditions and its evidence are made explicit.

Part III asks the system question: **how can many people, models, tools, resources and organizations work together without turning access into authority, context into control, or a successful request into an unverified claim of completion?**

The answer proposed here has three parts.

First, **people hold authority**. A model can propose, compare and call permitted tools, but the right to read a document, reserve a machine, start a motor, disclose a customer record, accept a commercial obligation or release a payment comes from an accountable person or organization acting within an agreed scope. HSIMC therefore treats authority as a first-class business fact rather than an assumption hidden inside a chat session.

Second, **tools and resources need shared contracts**. MCP tools expose callable operations. MCP resources expose data and context. HSIMC Resources are real-world business objects such as people, places, machines, materials, inventories, work slots and services. These concepts can be connected, but they should not be collapsed into one another. A machine may be represented by an HSIMC Resource, described through MCP resources and controlled through one or more MCP tools. The relationship must be explicit.

Third, **stateless protocol requests do not remove real-world state or responsibility**. MCP 2026-07-28 removes protocol-level sessions and makes each request self-describing. Applications can still maintain state through explicit handles. In a physical workflow, persistent state is unavoidable: a machine can be busy, a pallet can be in transit, a bag can be in custody, a motor can still be moving, an invoice can remain unpaid. HSIMC therefore adopts a simple principle:

> **Stateless request; persistent real-world responsibility.**

The purpose of this Part is not to invent a new replacement for MCP, ERP, MES, WMS, IAM or contract law. It is to define a collaboration layer that lets those systems and real-world participants meet around one verifiable Work.

Chapter 7 — People Hold Authority

7.1 M-Person, organizations and M-Teams

GenAI changes who can plan. It should not silently change who is allowed to decide.

A person may use an AI system to draft a design, inspect a photograph, compare suppliers or prepare an operation request. A team may use several models and tools to work in parallel. An organization may expose selected capabilities through an API or MCP server. None of those facts, by itself, answers a more basic question: **who has the right to authorize the next real-world action?**

HSIMC uses the term **M-Person** as a participation concept: a real person who can work with models, tools and teams while retaining authority over the rights that belong to that person or the organization they legitimately represent. An **M-Team** can include people, models, software services and specialized agents, but it is not itself a magical legal principal. It acts through defined authority relationships.

This matters because the same person can occupy different roles in different Works.

A factory owner may be a buyer in one Work and a capability provider in another. A designer may contribute a drawing but have no permission to start a laser. A hotel may control the room and custody process but not the airline baggage system. An engineer may be permitted to query a T736 reference board but not deploy a new image to a delivered unit. A founder may be able to authorize one company directly but need another director or co-holder for a different company.

The architecture therefore separates:

- **identity** — who or what is being referred to;
- **role** — what function the participant has in this Work;
- **authority** — what decisions or operations the participant is permitted to make;
- **scope** — which Resource, time, location and operation the authority covers;
- **evidence** — what proves the authority was granted and what happened under it.

A single global profile can be useful for discovery, but authority should be evaluated at the point where a decision matters.

Identity is continuity, not permission

HSIMC preserves existing Runtime and Resource identities wherever possible. A card, QR code, NFC tag or short URL can help a person find the same object again. This continuity is valuable. It does not make the identifier an authorization token.

A printed QR on a machine might open its public capability page. That does not mean anyone who scans it may move the machine. An NFC tag on a luggage item might identify the service record. It does not mean anyone with a phone may change the delivery destination. A Card ID can connect a physical object to a digital profile. It does not replace the identity of the actual T736 device, its operator or its owner.

The distinction is simple:

> **Identity answers “what is this?” Authority answers “who may do what with it, now?”**

This is especially important in a global system where the same public Resource may be discovered by many models and users.

M-Teams should make responsibility clearer, not blur it

A strong M-Team can divide work across specialties:

- one model interprets a customer request;

- another checks a manufacturing constraint;
- a human operator confirms a material condition;
- a vision tool measures alignment;
- a supplier accepts a machining task;
- a local controller executes deterministic motion;
- an evidence service records the result.

The advantage is parallel capability. The risk is diffusion of responsibility.

For every Work, HSIMC therefore asks for named responsibility points. At minimum:

1. **Requester** — who wants the outcome;
2. **Resource authority** — who can commit the relevant machine, material, space, data or service;
3. **Executor** — who or what performs the action;
4. **Acceptor** — who decides the result meets the agreed condition;
5. **Economic principal** — who pays or receives value when relevant;
6. **Evidence custodian** — where the authoritative result record is retained.

One person may hold several of these roles. A model may assist one or more roles. The record should still distinguish them.

7.2 Document access, device control and contribution rights

A recurring mistake in connected systems is to treat all access as one permission.

It is not.

Reading a PDF, editing a design file, querying a sensor, reserving a machine and commanding physical motion are different actions with different risks. They should have different permissions even when they appear in the same user interface.

A useful permission ladder is:

Permission class	Example	Typical consequence
Discover	See that a capability exists	low consequence; may still expose commercial information
Read		information disclosure

Permission class	Example	Typical consequence
	Read a datasheet, drawing or current observation	
Contribute	Upload a model, design, sample result or process note	changes shared knowledge
Reserve	Hold a machine slot, inventory quantity or service window	affects availability to others
Configure	Change parameters, mappings or profiles	may alter later execution
Execute	Start a machine, write a tag, move a cart	physical or transactional side effect
Stop / recover	Stop motion, cancel or recover a process	safety and continuity impact
Accept / release	Accept output, release custody or authorize payment	commercial / governance consequence
Administer	Change identities, permissions or policy	system-wide consequence

This ladder is not a universal standard. It is a practical design tool. A real implementation can map these categories to existing IAM, OAuth scopes, operating procedures or local control roles.

Document authorization is not device authorization

The T736 handoff illustrates the distinction. An engineer may be authorized to receive a versioned hardware profile in OneDrive or another workspace. That delivery can be proven by file identity, version, hash and readback. It does not imply the engineer may control the reference board.

Similarly, a remote client can be given permission to run a simulator without being granted a physical target. The H0-001 handoff explicitly preserves this separation: document destinations and hardware control are different permissions, and a simulated execution target must not be silently promoted to a physical one.

This separation improves collaboration because information can move earlier than control.

A software engineer can begin writing against a common contract. A hardware engineer can design the interface. A local technician can prepare a fixture. The physical actuator can remain unbound until the authority and safety case are ready.

Contribution rights matter as much as access rights

A global capability network also needs to answer a different question: **what happens to the contribution itself?**

If a small workshop develops a better jig, a student writes an adapter, an engineer creates a calibration file, or a supplier provides a tested antenna geometry, the system should not assume unrestricted reuse.

A contribution record can include:

- contributor identity or organization;
- Work and Resource to which the contribution relates;
- version and hash;
- permitted use — evaluation, one project, one customer, internal use, resale, open reuse or another agreed scope;
- attribution requirement, if any;
- payment or revenue arrangement, if any;
- confidentiality classification;
- expiry or withdrawal conditions when applicable;
- evidence showing what was actually used in a delivered result.

This is not a substitute for a legal agreement. It is a way to make the operational facts legible enough that legal and commercial agreements can be executed with less ambiguity.

Least authority, explicit step-up

The safest default is to grant the smallest useful authority and expand it when the Work requires more.

A user may start with public discovery. Reading a private drawing requires authenticated access. Reserving a machine requires a stronger business relationship. Starting motion may require a named operator, location check, current safety state and an execution token or equivalent local authorization.

MCP 2026-07-28 strengthens protocol authorization and routing, including issuer validation and credential isolation. Those mechanisms help answer whether a client is authorized to call a protected protocol endpoint. HSIMC still needs the application-level decision about whether the requested real-world action is appropriate for that Resource, time and Work.

The protocol can carry the request. The business layer defines the authority.

7.3 Responsibility, payment, redress and withdrawal

Authority is incomplete without accountability.

A collaborative system should record not only who was allowed to act, but also what happens when work is late, incomplete, unsafe, disputed or no longer wanted.

Acceptance is a separate act

An operation can finish without being accepted.

A laser can complete a marking cycle and still produce the wrong serial number. A sewing machine can complete a conductive path that fails resistance testing. A delivery can arrive at a hotel but before the guest is ready. A model can generate a valid file that uses the wrong design revision.

Therefore a useful Work lifecycle separates:

requested → accepted for work → executing → execution complete → evidence available → accepted / rejected / partially accepted

This allows the system to distinguish “the machine stopped” from “the customer got the agreed result.”

Payment should follow a defined economic event

A global network becomes more inclusive only if contribution can translate into understandable economic value.

Possible payment triggers include:

- reservation accepted;
- material supplied;
- sample delivered;
- test passed;
- service completed;
- customer accepted;
- usage measured;
- revenue received.

The correct trigger depends on the Work. It should be agreed before execution where possible.

The evidence record should also distinguish cash value from non-cash contribution. A workshop may donate a sample for a pilot. A student may contribute work under an educational arrangement. A supplier may loan a development board. A company may sponsor testing. Those contributions still have value and should be recorded without pretending they are revenue.

Redress and correction

“No One Left Behind” requires more than access. Participants need a way to challenge an incorrect record or harmful outcome.

Examples include:

- a maker says the system attributed a sample to the wrong contributor;
- an operator says a task was assigned outside the agreed time window;
- a customer disputes an acceptance result;
- a person wants a private data link revoked;
- a device owner reports that a capability profile is outdated;
- a supplier says a model was reused outside the agreed scope.

The system should preserve the original evidence while allowing a correction, dispute or superseding decision to be added. Silent rewriting of historical evidence is usually worse than a visible correction trail.

Withdrawal does not reverse physics

A person can withdraw permission for future use. That does not always reverse an action already completed.

A file may already have been sent to an authorized recipient. Material may already have been cut. A package may already be in transit. A motor may already have moved a cart. A payment may already have settled.

Withdrawal therefore needs temporal scope:

- stop new requests;
- stop future reuse;
- cancel reservations not yet executed;
- request stop or recovery for actions in progress;
- preserve evidence of what already occurred;
- apply return, deletion, recall or compensation rules where those are actually possible and agreed.

This is another reason persistent real-world state matters even in a stateless protocol architecture.

Chapter 8 — Tools, Resources and Shared Contracts

8.1 Tools for actions; resources for context

MCP provides a useful vocabulary for connecting models to external systems, but the vocabulary must be used precisely.

In MCP, **tools** expose callable operations. A tool has a name, description and input schema, and may return structured or unstructured output. The server can list available tools and the client can call a selected tool.

In MCP, **resources** expose data and context, identified through URIs. They can represent text or binary content and can be listed or read by clients according to the server's capabilities and authorization model.

HSIMC uses **Resource** in a different, broader business sense: a real or organizational object that can participate in Work. Examples include:

- a person or team;
- a room, factory, hotel, farm or airport zone;
- a T736 device;
- a sewing machine or laser marker;
- a material batch;
- an inventory position;
- a pallet, wheelchair or luggage cart;
- a UWB anchor;
- a delivery service;
- a time slot;
- a verified engineering capability.

These concepts connect naturally, but they are not synonyms.

One real Resource can have many digital surfaces

Consider a UV laser marker in a local workshop.

The **HSIMC Resource** is the actual machine instance with an owner, location, configuration, maintenance state and authority relationship.

MCP **resources** might expose:

- operating manual;
- supported material profile;
- current calibration record;
- current job queue;
- latest camera alignment image;
- approved logo or serial-number specification.

MCP **tools** might expose:

- `get_status`;
- `prepare_job`;
- `locate_marks`;
- `submit_marking_job`;
- `get_operation`;
- `request_stop`.

A shared **Work contract** connects the business request to these interfaces:

> **Mark this versioned identifier on this batch, using this machine profile, within this time window, under this authority, and return this evidence.**

The machine is not the tool. The manual is not the machine. The URI is not the authority. The Work is not the transport session.

Keeping these layers separate is what allows the system to scale.

8.2 MCP resources and HSIMC Resource objects

A global collaboration network needs both **context discovery** and **real-world identity**.

MCP resources are well suited to context discovery. They give a model or client a structured way to obtain the documents or data needed for a task. HSIMC Resource objects are used to maintain continuity across physical and organizational changes.

A Resource object should survive interface changes

A machine may first be controlled manually. Later, a USB adapter is added. Later still, a T736 node provides an MCP interface. The real machine should not receive a new identity every time its software interface changes.

Likewise, a pallet can acquire a QR label, then an NFC tag, then an RFID tag. The tag technology can change while the Resource remains the same physical object or business object.

A useful Resource profile can include:

- stable Resource ID;
- human-readable name;
- organization or owner relationship;
- location or location reference;
- type and model where applicable;
- interfaces currently available;
- capability profiles;
- current evidence level;
- authority bindings;
- last verified observation time;
- versioned references to manuals, drawings or test reports;
- known limitations.

The profile should distinguish **possible capability** from **observed capability**.

A supplier datasheet may state that a module supports a bus. That is documentary evidence. A reference board may show that the bus is wired a certain way. That is another evidence layer. A real device test demonstrating communication with a specified peripheral is stronger evidence for that instance.

The HSIMC T736 handoff already uses this discipline. C0-005 is a document hardware baseline. C0-007 is a verified-source simulator. C0-008 records a remote simulation run. None of those records is silently rewritten as a current physical-board capability.

Fresh observation versus historical receipt

A collaboration system also needs to distinguish two questions:

1. **What happened in operation X?**
2. **What is true now?**

The answer to the first should be stable. The answer to the second may require a new observation.

For example, an operation receipt may correctly say that a tag was read at 09:00. Replaying the same receipt at 14:00 does not prove the tag is still present. A machine may have reported `idle` before another team reserved it. A UWB coordinate can become stale in seconds.

HSIMC therefore distinguishes:

- **operation receipt** — immutable or append-only evidence about a specific operation;
- **observation** — a time-stamped measurement or status assertion;
- **effective state** — application logic derived from observations, reservations, accepted Works and known transitions;
- **freshness requirement** — how recent the observation must be for the next action.

This distinction prevents idempotency from turning into stale-state reuse.

8.3 Function contracts across hardware and software

A capability becomes globally usable when another team can understand its contract without depending on the original conversation.

A shared function contract should answer at least:

- **name** — what function is requested;
- **purpose** — what useful result it produces;
- **inputs** — parameters, files, Resource IDs and units;
- **outputs** — result fields, units and evidence references;
- **preconditions** — physical, electrical, authority and freshness conditions;
- **side effects** — what may change in the real world;
- **exclusive resources** — what cannot be used concurrently;
- **timeouts** — when the caller should stop waiting or change strategy;
- **idempotency** — how retries avoid duplicate side effects;
- **errors** — machine-readable failure reasons;
- **recovery** — what to do after uncertain or partial execution;
- **version** — contract, adapter, firmware, model and process versions;
- **acceptance** — how the business result is verified.

A contract can exist before the final hardware

This is one of the most important benefits of the HSIMC cloud-development route.

Suppose a team is developing a sample-alignment function. Before the T736 camera path is ready, the contract can already specify:

- image input format;
- marker definition;
- output coordinates;
- confidence or quality field;
- calibration reference;
- error cases;
- version and evidence output.

A cloud simulator can operate on fixed images. A remote engineer can implement the algorithm. A local maker can print the marker and build a jig. A hardware engineer can design the camera adapter. When the physical board becomes available, the real camera replaces the synthetic input behind the same business-facing function, subject to new device validation.

The contract does not eliminate hardware differences. It gives those differences a place to be recorded.

Effects need explicit identity

For operations with side effects, a retry should not create a second physical action simply because a network reply was lost.

A practical pattern is to give the intended effect a stable operation identity and bind that identity to the effect definition.

For example:

- operation ID: `op-123`;
- target Resource: `laser-KT-01`;
- function: `submit_marking_job`;
- design hash: `abc...`;
- batch: `B-42`;
- quantity: `20`;
- authority context: current authorized caller and scope.

If the same operation ID is retried with the same intended effect, the server can return the existing receipt or continue the same work. If the operation ID is reused with a different effect, the server should reject the conflict.

The operation identity should not depend on a short-lived access token. Authorization is evaluated on each request; effect identity remains stable across token refresh or client reconnect.

Shared contracts are not universal permissions

Publishing a contract means others can understand how a capability works. It does not mean everyone may call it.

A public contract can coexist with private execution. This is useful for ecosystem development: engineers can build compatible clients and simulators while real devices remain protected by local and organizational authority.

This is another way to let collaboration begin before control is granted.

Chapter 9 — Stateless Requests, Accountable Outcomes

9.1 28 July 2026: context carried by each request

The 28 July 2026 MCP specification is a useful milestone because it changes the protocol's collaboration model in a way that maps well to global engineering.

The release moves MCP to a **stateless protocol core**. The previous `initialize` / `initialized` handshake and protocol-level session ID are removed for the modern protocol. Each request carries the protocol version, client identity and relevant client capabilities. A client can use `server/discover` when it wants server capabilities in advance, but the protocol no longer depends on a long-lived hidden session for ordinary request meaning.

The practical consequence is significant:

> **A request can be understood from explicit information carried with the request, rather than from an invisible conversational history attached to one connection.**

This makes ordinary web-style routing, scaling and recovery easier. It also aligns with the HSIMC objective that another engineer, device or network can continue work when the required identities, versions, authority and evidence are explicit.

Stateless protocol does not mean stateless application

The MCP maintainers make this distinction directly. If an application needs state across calls, it can mint an explicit handle and have the client pass that handle on later requests.

For a physical system, that approach is not optional. Many facts persist:

- Work ID;
- Resource reservation;
- device instance;
- operation ID;
- task handle;
- material batch;
- custody record;
- calibration version;
- payment state;
- acceptance result;
- recovery state.

The improvement is that the state is explicit and addressable rather than being hidden inside a transport session.

HSIMC summarizes the design principle as:

> **Stateless request; persistent real-world responsibility.**

Header-based routing can support governance

MCP 2026-07-28 also adds protocol headers such as `Mcp-Method` and, where relevant, `Mcp-Name`. Gateways can use this information for routing, metering and policy enforcement without having to infer the operation from a long-lived session.

For HSIMC, this suggests a useful separation:

- protocol gateway decides whether a request is structurally valid and authenticated;
- application authorization decides whether the principal may request this function on this Resource;
- local runtime checks current physical preconditions;
- deterministic controller executes the allowed physical action;
- evidence layer records the outcome.

No single layer needs to pretend it can decide everything.

9.2 Fresh observations, operation receipts and retries

Stateless systems must be especially disciplined about retries because a client can lose a response while the physical action still occurs.

Imagine a laser job:

1. client submits the operation;
2. server accepts it;
3. laser performs the mark;
4. network connection fails before the success response reaches the client.

If the client simply submits a new operation, the part may be marked twice.

The correct recovery behavior is:

> **When the result is uncertain, query the original operation before creating a new effect.**

This principle applies to manufacturing, logistics and payments.

Suggested operation states

A practical physical-operation lifecycle can include:

- `prepared` — inputs validated, no physical effect yet;
- `accepted` — authority and resources accepted;
- `running` — physical or transactional execution has started;
- `input_required` — execution needs explicit additional input or confirmation;
- `cancel_requested` — a stop/cancel request has been received;
- `cancel_confirmed` — the responsible executor confirms the action has stopped or will not proceed;
- `completed` — execution finished;
- `failed` — execution ended unsuccessfully with known state;
- `result_unknown` — the system cannot yet prove the final physical state;

- `accepted_result` — business acceptor has accepted the delivered result;
- `rejected_result` — result was inspected and rejected.

These are HSIMC application concepts, not a claim that MCP defines this exact state machine.

Cancellation is not physical stop

A digital `cancel` request can be acknowledged while the machine is still decelerating, while a courier is still holding the parcel, or after an irreversible step has already happened.

Therefore the system should distinguish:

- **request to cancel**;
- **executor's acknowledgement**;
- **confirmed physical stop or non-execution**;
- **recovery or disposition of partially changed material**.

For safety-related motion, local stop behavior must remain deterministic and independent of a remote model's continued connectivity.

Freshness belongs in the contract

A stateless request can be self-describing and still use stale information.

Therefore the request should identify freshness expectations where they matter. Examples:

- inventory observation no older than 5 minutes;
- UWB location no older than 500 milliseconds;
- machine safety state read immediately before execution;
- guest delivery preference valid until a stated time;
- calibration valid for a specific fixture and software version.

If the available observation is older than the threshold, the correct outcome may be `refresh_required`, not a guessed result.

Multi Round-Trip Requests and explicit confirmation

MCP 2026-07-28 introduces Multi Round-Trip Requests for cases where a tool needs additional input during a stateless interaction. A server can return an input-required result, and the client can resubmit the original call with the requested responses.

This is conceptually useful for high-consequence actions:

- confirm the quoted cost before a paid manufacturing run;

- ask for a missing dimension before cutting;
- request approval before releasing a shipment;
- confirm a changed hotel delivery time;
- request a local operator acknowledgement before activating a physical reference board.

MRTR is a protocol mechanism. Whether a particular real-world action requires confirmation remains an application and governance decision.

9.3 Persistent responsibility across changing clients and roles

The strongest reason to adopt explicit handles, Resources, Work IDs and evidence is not technical elegance. It is continuity.

Global collaboration means the original client may disappear.

The engineer who designed the part may go offline. The phone used to scan a tag may change. The model session may end. A different service provider may take custody. The hardware may be shipped from Hong Kong to another country. A future engineer may need to reproduce a result months later.

A responsible system should still answer:

- What Work was requested?
- Which version was used?
- Which Resource actually performed it?
- Who had authority at the time?
- What observations were fresh enough to support the decision?
- What physical effect occurred?
- What evidence proves it?
- Who accepted the result?
- What remains uncertain?

That is what makes a collaboration portable across teams.

The Work object as the continuity anchor

HSIMC proposes treating **Work** as the business continuity anchor. A Work is not merely a task record. It binds:

- desired result;
- requester;
- relevant Resources;
- authority and acceptance roles;
- time and place;
- capability / tool contracts;
- input versions;
- reservations;
- operations;
- evidence;
- economic terms;
- acceptance outcome.

A Work can outlive a chat session. It can reference an MCP task when a tool chooses to run asynchronously. It can also exist without using MCP Tasks at all.

This distinction is important because MCP 2026-07-28 moves Tasks into an extension. Protocol support should be detected and used when both sides actually support it. The business Work should not disappear simply because a particular client or server does not implement that extension.

From global planning to local execution

A useful architecture for physical work is:

Global / cloud layer

- discover capabilities;
- compare plans;
- assemble contracts;
- check business authorization;
- schedule and route Work;
- maintain versioned evidence references.

Local runtime layer

- bind to the actual device;
- read current physical state;

- enforce local interlocks and exclusive resources;
- translate common contracts to device-specific drivers;
- keep operation records through network interruption;
- expose result and recovery state.

Deterministic control layer

- execute timing-sensitive motor, laser, actuator or safety logic;
- stop safely on defined faults;
- avoid depending on remote language-model latency for hard real-time safety.

This layered model lets GenAI contribute strongly without asking it to become a safety PLC.

Evidence is the bridge between trust and scale

Global participation fails if every new partner must trust every other partner personally.

Evidence reduces that burden.

A maker can show that a sample was produced under a known process. A remote engineer can reproduce the analysis. A machine can return an operation receipt. A customer can accept or reject against a stated criterion. A second location can repeat the procedure and compare results.

The evidence can include:

- hashes and version IDs;
- photographs and video;
- sensor or measurement files;
- calibration references;
- machine logs;
- material batch records;
- read/write verification for NFC or RFID;
- signed or authenticated acceptance;
- timestamps and location references;
- financial receipts where appropriate.

Evidence does not eliminate the need for trust, standards or legal responsibility. It makes trust more transferable.

No One Left Behind requires visible authority and visible value

A global system can be technically open and still exclude people economically or socially.

If only large companies can describe capabilities, small makers remain invisible. If the model can use a person's contribution but the contributor cannot see how it was used, participation becomes extractive. If an operator is expected to absorb every failure but has no authority to reject unsafe work, the system is not meaningfully inclusive.

Part III therefore ends with a governance test for every Work:

1. **Can the participant understand the requested result?**
2. **Can the participant say yes, no or propose conditions?**
3. **Is the authority scope explicit?**
4. **Can the contribution be attributed and evidenced?**
5. **Can the participant see the acceptance outcome?**
6. **Can economic value or non-cash contribution be recorded accurately?**
7. **Can a dispute or correction be added without erasing history?**
8. **Can another authorized team reproduce the result from the records?**

If the answer is consistently yes, Stateless MCP becomes more than a scalable transport pattern. It becomes part of an accountable collaboration architecture in which models can move quickly while people retain authority over real-world consequences.

Part III conclusion — From connectivity to accountable collaboration

The physical GenAI era does not need another promise that “everything will be connected.” Connection is already abundant.

The harder problem is coordination with responsibility.

Part III proposes a simple operating model:

> **People hold authority. Tools expose actions. Resources provide context. HSIMC Resources preserve real-world identity. Shared contracts define functions. Stateless requests carry explicit context. Persistent Work and Evidence preserve responsibility.**

This model allows a small maker, a global engineer, a cloud model, a T736 node, a machine controller, a logistics provider and a customer to participate in the same Work without pretending they have the same role or authority.

It also prepares the next Part of this white paper.

Part IV moves from collaboration architecture into the engineering path: how teams develop together before hardware ships, how the T736 reference route supports cloud-to-physical iteration, and how one function can be reproduced by many makers and devices.

Evidence and source notes for Part III

S1 — Model Context Protocol, 2026-07-28 release. The official MCP release describes the stateless protocol core, removal of the `initialize` / `initialized` exchange and protocol-level session ID, self-describing requests, `server/discover`, header-based routing, cacheable list results, authorization hardening, MRTR, Tasks as an extension and deprecations. It explicitly states that applications can remain stateful through explicit handles.
https://blog.modelcontextprotocol.io/posts/2026-07-28/

S2 — MCP 2026-07-28 release candidate / protocol evolution. The release-candidate documentation describes the stateless redesign, Tasks lifecycle changes, authorization hardening and the principle that application state can be carried by explicit handles rather than transport sessions.
https://blog.modelcontextprotocol.io/posts/2026-07-28-release-candidate/

S3 — MCP SDK migration guidance for 2026-07-28. Current SDK documentation describes per-request modern headers, protocol-version handling, resource subscriptions and authorization requirements for the 2026-07-28 era. It is implementation guidance and does not by itself prove compatibility of any HSIMC client or server.
https://ts.sdk.modelcontextprotocol.io/v2/migration/support-2026-07-28

S4 — HSIMC Card Phone & Cloud Development Handoff H0-001, 11 November 2026. The project handoff preserves T736 as the shortest current route, cloud simulation → remote physical reference board → freeze → ship → local binding, Resource / Authority / Evidence continuity, separate document and hardware-control permissions, and the evidence boundary for C0-005 / C0-007 / C0-008. H0-001 did not re-probe the physical T736.

S5 — HSIMC Stone & Thread White Paper, Part I W0-005 and Part II W0-006. These earlier Parts define the physical-outcome gap, credit-card-sized starting point, Leave No One Behind participation principle, material/function evidence discipline and the transition from craft to reproducible capability.

Evidence discipline

This Part distinguishes:

- **MCP protocol facts** from official MCP sources;
- **HSIMC architectural proposals** such as M-Person, Work, the permission ladder and proposed physical-operation states;
- **existing HSIMC project evidence** from H0-001 and the previous white-paper Parts;
- **future implementation requirements** that still need client, server, hardware and real-work validation.

No statement in this Part should be read as proof that a particular external MCP client already supports HSIMC physical control, that a real T736 board is currently reachable, or that any machine is authorized for remote execution.