

Let the AI ask the planning model directly — *on your terms.*

Large language models are good at reasoning over data and bad at getting it. Most "AI for finance" bridges that gap by pasting spreadsheets into a chat window, which loses the structure the numbers live in and every control that governed them. The Model Context Protocol is the open standard that lets an AI client query a system the way a governed user would — through named tools, with the caller's own permissions, audited. This article explains what MCP is, what EAConnect's embedded server exposes to Claude, Cursor and custom agents, and how the governance holds.

AUDIENCE	THE SERVER	DEFAULT EXPOSURE	READING TIME
Leaders of mid-size enterprises	In-process · JSON-RPC at /api/mcp	Off, everywhere	9 min read

Series Hours, Not Months

Evidence EAConnect MCP, External MCP Federation and API Keys user guides (v2.0, Jun 2026); MCP domain documentation, code-reviewed Jun 2026

Standard shipped vs specified stated explicitly

01 What MCP is, in one paragraph

A protocol, not a product. It standardises how an AI client discovers and calls the tools a system chooses to expose.

The Model Context Protocol (MCP) is an open specification, published in late 2024 and now supported by the major AI clients, for connecting language models to external systems. A system runs an *MCP server* that publishes a list of *tools* — each with a name, a description and a typed input schema — and the AI client calls them over a simple JSON-RPC exchange. The model never touches a database; it asks a tool, the server runs the tool under whatever rules it enforces, and the result comes back as data the model can reason over. The consequence for finance is that "AI access to the plan" stops meaning "export and paste" and starts meaning "the same read path a user has, with the same permissions and a log".

CLIENT

The AI host

Claude Desktop, Cursor, an in-product chat, a scheduled agent, or a script. It discovers tools, decides which to call, and reasons over what comes back.

SERVER

The system's tool surface

Publishes named tools with JSON schemas; validates input; runs each call as an authenticated caller; returns typed results; records the call.

TOOL

One governed capability

`planning.read_data`,
`query.aggregate`, `reports.run`.
Coarse enough to be auditable, specific enough that the model cannot wander.

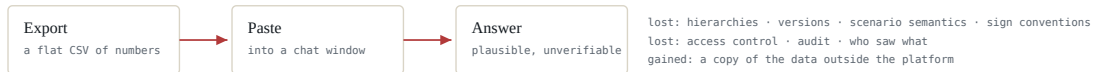
02 Why planning data needs a protocol, not a paste

A P&L is not a table. It is a set of measures at the intersection of dimensions with hierarchies, versions and access rules. A model that sees only a flat extract cannot reason about any of that.

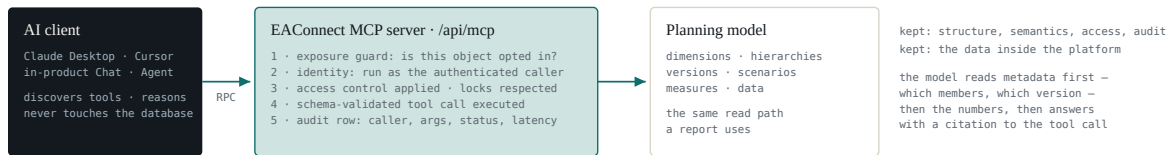
Ask a chat assistant "why is EMEA operating income below budget in Q2" and it needs to know which entity members roll into EMEA under the current hierarchy version, which

scenario is the budget, which version of the forecast is approved, which accounts constitute operating income and with what sign, and whether the asker is allowed to see the UK subsidiary at all. None of that is in a CSV. All of it is in the planning model's metadata — and MCP is how a model reads the metadata first, then the data, in the same governed way a report does.

THE PASTE PATH



THE MCP PATH



Every call is a row in the MCP audit log with caller kind (chat, external host, agent), argument summary, status and latency. Sensitive argument names are redacted.

Figure 1. Two paths. The paste path is fast and leaves the numbers, their meaning and their governance behind. The MCP path is a governed read, in place.

03 What EACConnect's embedded server exposes

The server runs in-process inside EACConnect Planning and publishes roughly thirty-two internal tools in named packs. Each pack is gated separately, and

nothing is exposed until an administrator turns it on.

TOOL PACK	WHAT IT DOES	GATE	STATUS
meta.*	Discover analytics workspaces: tables, columns, relationships, time intelligence	Workspace mcp_exposed	Shipped
query.*	Aggregate and filter analytics data; a constrained SQL variant for read queries	Workspace mcp_exposed	Shipped
reports.*	List and run saved reports programmatically	Per-report mcp_runnable	Shipped
planning.*	Read the planning model: list models and their details, measures, dimensions and dimension trees, hierarchy nodes, tables — and read data at an intersection	Per-model mcp_exposed + RBAC	Shipped — ten read tools
planning_variables.*, analytics_variables.*	List, parse and resolve planning and analytics variables — the central rates and member sets	Model gate for planning; open read for analytics	Specified (BUILD-SPEC-176)
workflow.*	Start and monitor workflow apps; mutations require a confirmation token	Per-app exposure flag	Shipped — eight tools
insights.*	Submit and resolve insight findings	Agent runtime only — hidden from chat and external hosts	Shipped

`external.{slug}.*`

Tools from registered
third-party MCP
servers, namespaced

Admin
registration;
trusted or
approval-per-call

Shipped

Tool counts and status are as documented in June 2026. The absence of a general write tool is deliberate: chat and external hosts read; the only writes are insight submissions by agents and confirmed workflow actions.

`planning.list_models`

→ the models this caller may see, with codes

`planning.get_dimension_tree`

→ e.g. Account: the versioned hierarchy, as a

`planning.get_hierarchy_nodes`

→ members under a node – "everything in Operat

`planning.read_data`

→ measures at an intersection: model, scenario

`reports.run`

→ a saved statement, executed with the caller

`workflow.start (tier C)`

→ begins a budget-submission app – after a cor

The design principle behind the list is that a tool should map to something a user can already do in the interface, so that exposing it to an AI client adds a caller, not a capability. The model reads what a user can read, through the same services, and nothing that was previously impossible becomes possible because a language model asked.

04 Governance: five controls, in order

The question a CFO should ask of any AI integration is not "what can it do" but "what stops it". Here is the sequence every MCP call passes through, as documented.

EVERY TOOL CALL, IN ORDER

1 · Exposure

mcp_exposed is false by default on every workspace, model, report and workflow app · each opted in separately by an admin

2 · Identity

runs as the authenticated user · an API key inherits exactly its owner's grants and never adds any · never elevated

3 · Access & locks

data access control by entity, cost centre, scenario applied to reads · slice locks respected · tools cannot bypass either

4 · Tier

A: read, automatic
B: read with caps
C: side-effect – needs a single-use, user-bound confirmation token

5 · Audit

append-only log:
caller kind · tool · args
summary · status · latency
sensitive fields redacted

Also: the chat orchestrator caps a conversation at roughly eight tool rounds, so a runaway loop cannot walk the whole model; agent-only write tools are invisible to chat and external hosts; external servers default to untrusted, so a person approves each call until an admin marks them trusted; stored credentials for external servers are encrypted at rest.

The order matters: exposure before identity, identity before data, and nothing side-effectful without a person's token. An AI client is the least-privileged user on the system.

Figure 2. The control sequence. A finance leader can verify each stage from the administration screens: which objects are exposed, which keys exist and whose they are, what the audit log recorded.

The consequence for access reviews

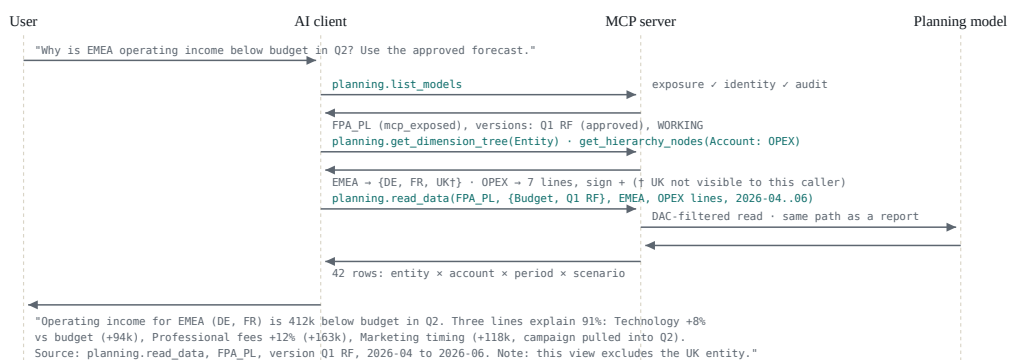
Because every MCP caller is a user — a person in chat, an API key owned by a person, an agent running as a person — the organisation's existing access model already covers AI access.

There is no separate "AI permissions" question to answer. If Priya cannot see the UK entity, neither can Claude when Priya asks, nor the agent Priya configured, nor a script using Priya's key.

05 A session, walked through

What actually happens when an FP&A lead connects Claude Desktop to the planning model and asks a variance question. Tool names are the documented ones; the question and answer are illustrative.

FP&A LEAD · CLAUDE DESKTOP · EACONNECT MCP · PLANNING MODEL



Four tool calls, four audit rows. The answer cites the version it used and states the scope limit the access rule imposed — the model was told what it could not see.

Figure 3. Metadata first, then data, then the answer. The hierarchy read is what lets the model use "EMEA" and "operating expense" correctly; the version read is what stops it mixing a working draft with the approved forecast.

Two details in the sequence are the whole argument for doing this through a protocol. The model discovered that the caller cannot see the UK entity and said so in the answer, rather than silently reporting a partial number as a total. And the answer carries the tool, model, version and period range it was built from, which is what makes it checkable in the same screens a controller would use.

06 Federation: other systems' tools, alongside yours

MCP is symmetric. Just as EAConnect exposes a server to external clients, it can register external servers and offer their tools inside its own chat and agents.

An administrator registers a third-party MCP server — a ticketing system, a source-code host, an internal tool — by URL and credential. EAConnect fetches its tool list, caches it, and presents the tools namespaced as `external.{slug}.{tool}` next to the internal ones. The chat orchestrator merges the two catalogues each turn. Servers default to *untrusted*, which means a person approves each call until an administrator marks the server trusted; an optional allowlist limits which of a server's tools are federated at all; credentials are encrypted at rest.

What it is for

Correlating a plan variance with the tickets or releases behind it; letting an agent read an approved external source alongside the planning data; reaching enterprise systems that publish their own MCP endpoints.

What it is not for

Replacing analytics workspaces as the home of core FP&A data, or bypassing internal access control — external tools run with the server's stored credential, not the user's, so what they return is only as governed as that credential.

A note on the iPaaS

An earlier dedicated chat adapter for the EAConnect integration platform was retired; chat now uses internal tools and generic external federation. Integration flows are run through processes and workflow tools, not through a chat-specific bridge.

07 Three ways a finance team uses it

The same server, three callers, three governance profiles — which is why the

audit log records which kind of caller made each call.

CALLER	HOW IT CONNECTS	TYPICAL USE	GOVERNANCE NOTE
In-product chat	Built in; the orchestrator holds the session	A planner asks a question of the model or a saved report during a review	Read-only; workflow actions need a confirmation token; roughly eight tool rounds per turn
External AI client	Claude Desktop, Cursor or a custom agent over Streamable HTTP at <code>/api/mcp</code> , authenticated with a personal API key	An analyst works in their preferred assistant with live access to the plan; a script prepares a narrative from the approved version	The key inherits its owner's permissions and can never add any; shown once at creation; revocable; default 90-day expiry
Scheduled agent	Runs inside the platform as a named user on a cron schedule	Standing surveillance: variance, headcount, readiness — the subject of the agents article	The only caller allowed the <code>insights.* write tools</code> ; team-scoped datasources
The external-client route is the one leaders most often underestimate: it lets the organisation's analysts use whichever AI assistant they already work in, against governed planning data, without a copy of that data leaving the platform.			

08 A practitioner's view for leaders

Opinions from people who have connected AI clients to financial systems and

reviewed the results. Not product claims.

Insist on a protocol, not an export

The first question for any AI-in-finance proposal: does the model read the system, or a copy of it? A copy has no permissions, no versions and no audit trail, and it is out of date the moment it is made.

Expose narrowly, then widen

Default-off is the right default. Start with one model and one report exposed, for one team, and read the audit log for a month before opening more. What the log shows people actually ask is a better guide to exposure than any policy document.

Every AI caller is a user

Chat as the person, keys owned by a person, agents running as a person. If a vendor's AI integration uses a service account with broad access, your access reviews no longer cover it — and neither does your audit.

Reads are cheap; writes need a person

Letting a model read the approved forecast is low-risk and high-value. Letting it change a number is neither. The confirmation-token pattern — the model proposes, a person confirms, the token is single-use — is the right boundary for anything with a side effect.

Demand citations in the answer

A good answer names the model, version and period it read. A model that can cite its tool calls can be checked; one that cannot is an opinion with confident formatting.

Treat federation as an integration, not a chat feature

Registering an external server is granting a credential. Review it as you would any integration: what it can reach, who trusted it, and whether the tool allowlist is as narrow as the use case.

EACConnect Planning · Research library · Hours, Not Months

Evidence: EACConnect MCP, External MCP Federation and API Keys user guides (v2.0, Jun 2026); MCP domain documentation code-reviewed Jun 2026. Tool packs and counts as documented; variable tools marked specified. The session in section 05 is illustrative; tool names are documented.

2,042 words · 9 min read