Independent Submission
Internet-Draft
Intended status: Informational
Expires: 5 December 2026

D. Soden
D. Walker
Beacon Spec

3 June 2026

A Well-Known URI Profile for Agent-Callable Commerce Endpoints
draft-soden-wellknown-mcp-commerce-00

Abstract

This document defines an informational profile for the Server Card document retrieved from the /.well-known/mcp.json Well-Known URI defined by the Model Context Protocol (MCP) project. The profile adds an optional commerce extension to the Server Card, carried in the Server Card's _meta object under the reverse-DNS key com.beaconspec/commerce, for commercial businesses operating MCP servers. The extension carries business identity, geographic location, industry classification, offering type, and advertised capability categories. The purpose of the profile is to give AI agents enough machine-readable information about a commercial MCP server to filter, select, and bootstrap an interaction without first opening an MCP session against every candidate server.

Note to Readers

Discussion of this draft takes place on the IETF DISPATCH mailing list (https://mailarchive.ietf.org/arch/browse/dispatch/ (https://mailarchive.ietf.org/arch/browse/dispatch/)). Source for this draft and an issue tracker can be found at https://github.com/beaconspec/mcp-commerce-profile (https://github.com/beaconspec/mcp-commerce-profile).

Status of This Memo

This Internet-Draft will expire on 5 December 2026.

Table of Contents

1. Introduction

The Model Context Protocol (MCP, see [MCP-SPEC]) provides a standardized mechanism by which Large Language Model (LLM) applications can invoke tools and retrieve context from external services. As of the 2025-11-25 specification release, MCP is hosted under the Linux Foundation (LF Projects, LLC) and governed by a Core Maintainer group.

A Specification Enhancement Proposal (SEP-1649) currently under review within the MCP project proposes a standardized discovery mechanism, the /.well-known/mcp.json endpoint, that allows an MCP client to retrieve a Server Card describing an MCP server's capabilities prior to establishing a session. This document does not duplicate or conflict with that work. It defines an optional commerce extension layered on top of the Server Card, addressed at the specific use case of discovering and transacting with commercial businesses through an AI agent.

1.1. Motivating Flow

The profile exists to enable interactions of the following form. An end user asks a digital assistant: "find me a pair of size 10 running shoes under $20 near me, and order a pair if you find something in stock." A well-implemented assistant decomposes this into three steps:

1. Query a marketplace or registry of MCP servers that index commercial businesses, filtering by geographic proximity to the user, by industry classification (apparel, footwear), and by advertised capability (the ability to take an order programmatically).

2. For each candidate result, open the candidate's MCP server and use the standard MCP capability-listing methods (such as tools/list) to obtain the precise inventory and order-placement tools.

3. Identify a small number of viable candidates, surface them to the end user for confirmation, and place the order via the chosen server's MCP tools.

Without a shared vocabulary for business-level metadata, step 1 either fragments across vendor-specific directories or forces the assistant to open MCP sessions against every candidate to inspect tool descriptions, both of which are expensive and slow. This profile defines the shared vocabulary.

1.2. Relationship to Other Work

This profile depends on the Server Card schema defined by SEP-1649 [MCP-SERVER-CARDS] as a baseline. Where SEP-1649 specifies how an MCP client connects to a server (transport, identity of the server, MCP-level authorization), this profile does not re-specify those mechanisms. It only adds business-level information about the entity operating the server.

This profile is not a competitor to Schema.org's LocalBusiness vocabulary ([SCHEMA-ORG-LB]). Schema.org describes businesses for human-facing web pages. This profile describes businesses for agent-facing MCP discovery. Publishers may publish both; an optional field of this profile carries a Schema.org type for cross-referencing.

This profile is not a replacement for, nor a competitor to, the llms.txt convention (https://llmstxt.org (https://llmstxt.org)). llms.txt is for passive LLM context; this profile is for active agent discovery and invocation. The two are complementary.

1.3. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.4. Terminology

Server Card The discovery document retrieved from /.well-known/mcp.json, as defined by the core MCP specification (referenced here as [MCP-SERVER-CARDS]). The same document may also be served at /.well-known/mcp/server-card.json, the location proposed by [MCP-SERVER-CARDS-2127].

Commerce Profile (or "this profile") The set of additional fields defined by this document, placed inside the Server Card's _meta object under the key com.beaconspec/commerce.

Marketplace (or Registry) A service that indexes Server Cards from multiple commercial MCP servers and exposes a query interface to AI agents. A marketplace is consumed by AI clients; it is not part of the normative protocol defined here.

Profile Publisher The party operating an MCP server who authors and publishes a Server Card carrying a Commerce Profile in its _meta object.

Origin As defined in [RFC6454].

2. The Commerce Profile Extension

2.1. Location Within the Server Card

The Commerce Profile is carried within the Server Card's _meta object, the metadata extension mechanism defined by the MCP base specification [MCP-META]. The profile occupies a single member whose key is com.beaconspec/commerce.

This profile is independent of the path at which the Server Card is published. A Profile Publisher SHOULD serve the same Server Card document at both /.well-known/mcp.json (the canonical location) and /.well-known/mcp/server-card.json (the location proposed by [MCP-SERVER-CARDS-2127]), so that clients following either convention can discover it. Because the Commerce Profile travels inside the Server Card's _meta object, it is carried unchanged regardless of which path a client retrieves.

The key uses the reverse-DNS prefix form that MCP defines for _meta members. Per [MCP-META], only prefixes whose second label is modelcontextprotocol or mcp are reserved for the MCP project; com.beaconspec is controlled by Beacon Spec, so com.beaconspec/commerce is a conformant third-party key.

Implementations that do not understand the com.beaconspec/commerce member MUST ignore it, which is the default handling for unrecognized _meta members.

The value of the com.beaconspec/commerce member MUST be a JSON object. The object MUST conform to the schema defined in Section 3.

2.2. Profile Version

The version field within the commerce object carries the version of this Commerce Profile that the document conforms to. Versions follow Semantic Versioning 2.0.0 [SEMVER], formatted as the string "MAJOR.MINOR.PATCH". This document defines version "1.0.0".

Major-version increments indicate backward-incompatible schema changes. Minor-version increments indicate backward-compatible additions. Patch-version increments indicate editorial clarifications without schema change.

3. Fields

Unless otherwise stated, all fields in the commerce object are OPTIONAL. Fields explicitly marked "REQUIRED" below MUST be present in every commerce block.

3.1. version (REQUIRED)

A string in Semantic Versioning 2.0.0 format. MUST equal "1.0.0" for documents conforming to this profile.

3.2. lastUpdated (REQUIRED)

An ISO 8601 timestamp ([RFC3339]) indicating when the commerce block was last modified. Marketplace implementations SHOULD use this field to skip re-indexing unchanged entries.

3.3. businessName (REQUIRED)

The business name as it should be displayed to end users. This may differ from the legal entity name. UTF-8 string, recommended maximum length 200 characters.

3.4. businessDescription (REQUIRED)

A short paragraph describing the business in plain language, for presentation to end users by AI agents. UTF-8 string, recommended maximum length 1000 characters.

3.5. endpoint (REQUIRED)

An object identifying the single hosted endpoint an AI client connects to, and the protocol spoken there. It has two members:

type (REQUIRED) Either "mcp" (a Model Context Protocol server) or "api" (a plain HTTPS REST API described by an OpenAPI document). A commerce listing exposes exactly one hosted endpoint. The server self-describes its catalog on connection: for mcp, via the MCP initialize and tools/list exchange; for api, via its OpenAPI document.

url (REQUIRED) A fully-qualified HTTPS URL for the endpoint. url MUST begin with https://.

A commerce listing addresses a live, hosted endpoint, not a package to be installed. Where the SEP-1649 Server Card already specifies an MCP endpoint for the same server, the values SHOULD be consistent.

3.6. naics (REQUIRED)

A non-empty array of strings. Each string is a NAICS 2022 industry classification code (the North American Industry Classification System, jointly maintained by the United States Census Bureau, Statistics Canada, and INEGI Mexico; see [NAICS]).

Real businesses commonly span multiple codes; the array shape is intentional. Codes MUST be six-digit strings. Numeric encoding is not permitted, because leading zeros are significant.

NAICS is North American by origin but widely used as a key in customer-relationship-management, sales, and business-to-business systems globally. International classification systems (ISIC, NACE) are out of scope for this version of the profile.

3.7. schemaOrgType (OPTIONAL)

A string equal to a Schema.org LocalBusiness subtype identifier (e.g., "Plumber", "HVACBusiness", "Restaurant"). Provided as a complement to NAICS for AI agents that consume Schema.org JSON-LD natively. See [SCHEMA-ORG-LB].

3.8. offeringType (REQUIRED)

A controlled-vocabulary string identifying what the business fundamentally offers. MUST be one of:

"product" The business sells physical or digital goods.

"service" The business performs work for a customer (including Software-as-a-Service).

"content" The business produces or hosts informational content (news, reference, documentation, community).

"mixed" The business combines two or more of the above as primary offerings.

3.9. locality (REQUIRED)

A controlled-vocabulary string identifying whether the business serves a defined geographic region or operates online without location dependence. MUST be one of:

"local" Service or fulfillment is geographically bounded (e.g., a plumber, a restaurant). The geo field MUST be present.

"online-only" The business operates without geographic constraint (e.g., a global SaaS). The geo field MAY be omitted.

"hybrid" Both modes apply (e.g., a manufacturer that ships globally but also operates a flagship retail location). The geo field MUST be present.

3.10. geo (REQUIRED when locality is "local" or "hybrid")

An object with at minimum the following fields:

country (REQUIRED, string) ISO 3166-1 alpha-2 country code (e.g., "US", "DE", "JP").

city (REQUIRED, string) City name in UTF-8.

region (OPTIONAL, string) State, province, or other first-level subdivision; ISO 3166-2 RECOMMENDED.

postalCode (OPTIONAL, string) Postal or ZIP code as published locally.

3.11. capabilityTags (REQUIRED, possibly empty)

An array of capability identifiers advertising what the endpoint can do. These are not drawn from a fixed vocabulary. They are the function names discovered from the endpoint at ingestion time: the MCP tool names returned by tools/list for an mcp endpoint, or the operation identifiers in the OpenAPI document for an api endpoint. The full tool or operation definitions live in the Server Card itself; capabilityTags is the flat, filterable list a marketplace indexes.

The array MAY be empty (for example, before an endpoint has been probed). Entries MUST be unique. Account registration and authentication functions carry no special meaning here; registration and authentication are handled by the endpoint itself and are out of scope for this profile.

3.12. Other Optional Fields

contact Object with optional email and phone string members. For human-escalation paths when an AI agent cannot resolve a transaction.

currency ISO 4217 currency code (e.g., "USD"). Indicates the

primary currency in which prices are quoted. Independent of geo.country (a US-based site may quote in "USD" while serving customers worldwide).

languages Array of ISO 639-1 (or 639-3 where 639-1 does not exist) language codes that the MCP server's user-facing surfaces understand. Defaults to whatever the site's primary language is.

privacyPolicyUrl, termsOfServiceUrl HTTPS URLs to the publisher's privacy policy and terms of service. AI agents acting on user behalf SHOULD surface these to end users before transacting.

displayName Branding name to be used for display when it differs from businessName.

logoUrl HTTPS URL to a square logo image suitable for AI client display. PNG or SVG RECOMMENDED.

4. Marketplace Consumption

This profile defines a publication format. It does not define a federation protocol between marketplaces and AI clients. Integration of marketplaces with end-user AI clients (such as chat-based assistants, IDE-integrated coding agents, or operating system-level digital assistants) is currently client-specific and is outside the scope of this document.

Marketplaces MAY populate their indexes by any means consistent with the policies of the sites they index. Two common patterns are recognized:

1. *Submission.* The Profile Publisher submits the URL of their /.well-known/mcp.json to the marketplace via a marketplace-specific submission form or API.

2. *Crawl.* The marketplace fetches /.well-known/mcp.json from a known set of origins on a schedule.

Marketplaces that accept submissions SHOULD verify that the submitter controls the origin from which the Server Card is served before listing the entry. This document defines two compatible challenge methods. The marketplace issues a one-time token and the submitter satisfies the challenge by either method (the submitter chooses based on their available control surfaces):

HTTP file challenge The submitter publishes the token at the path

/.well-known/mcp/marketplaceValidation.txt on the origin. The marketplace fetches this URL and matches the body against the issued token. Mirrors the HTTP-01 challenge type of [RFC8555].

DNS TXT challenge The submitter publishes the token as a DNS TXT record at a conventional name beneath the origin's apex (e.g., _mcp-marketplace-validation.example.com). The marketplace resolves the record and matches it against the issued token. Mirrors the DNS-01 challenge type of [RFC8555].

Marketplaces SHOULD re-fetch the Server Card on a schedule consistent with the lastUpdated field, and SHOULD reject or de-list entries whose ownership challenge can no longer be satisfied.

5. Security Considerations

Documents conforming to this profile are public by definition; they describe a business's intent to be discovered by AI agents. Profile Publishers MUST NOT place secrets, end-user data, or session credentials in the commerce block.

AI agents that consume Commerce Profile data MUST treat it as self-asserted by the publisher and not, in itself, a statement of identity verification. Trust establishment is out-of-band and not specified by this document. The ownership-of-origin challenges defined in Section 4 establish that the submitter controlled the origin at submission time; they do not establish anything about the legal identity, regulatory status, or solvency of the business behind the origin.

AI agents SHOULD surface privacyPolicyUrl and termsOfServiceUrl (when present) to end users before initiating an irreversible transaction (for example, a payment) on the user's behalf.

6. IANA Considerations

This document requests no new IANA registrations on its own authority. The path /.well-known/mcp.json is registered under the procedure described by SEP-1649; this profile defines a _meta member inside the document served at that path.

Editor's note for IANA reviewer: should circumstances require a sub-suffix registration for the artifact at /.well-known/mcp/marketplaceValidation.txt defined in Section 4, the authors are prepared to file a separate registration request consistent with [RFC8615].

7. Examples

A reference example Server Card with a populated commerce block is published in the source repository for this draft at https://github.com/beaconspec/mcp-commerce-profile/blob/main/examples/.well-known/mcp.json (https://github.com/beaconspec/mcp-commerce-profile/blob/main/examples/.well-known/mcp.json). A normative JSON Schema is published at https://github.com/beaconspec/mcp-commerce-profile/blob/main/schema/commerce-profile-v1.0.0.json (https://github.com/beaconspec/mcp-commerce-profile/blob/main/schema/commerce-profile-v1.0.0.json).

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002, <https://www.rfc-editor.org/info/rfc3339>.

[RFC6454] Barth, A., "The Web Origin Concept", RFC 6454, December 2011, <https://www.rfc-editor.org/info/rfc6454>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

[RFC8555] Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", RFC 8555, March 2019, <https://www.rfc-editor.org/info/rfc8555>.

[SEMVER] Preston-Werner, T., "Semantic Versioning 2.0.0", 2013, <https://semver.org/spec/v2.0.0.html>.

[NAICS] U.S. Census Bureau / Statistics Canada / INEGI, "North American Industry Classification System, 2022 Edition", 2022, <https://www.census.gov/naics>.

8.2. Informative References

[MCP-SERVER-CARDS]
Model Context Protocol Project, "SEP-1649: MCP Server Cards: HTTP Server Discovery via .well-known", 2025, <https://github.com/modelcontextprotocol/modelcontextprotocol/issues/1649>.

[MCP-SERVER-CARDS-2127]
Model Context Protocol Project, "SEP-2127: MCP Server Cards (HTTP Server Discovery via .well-known), successor to SEP-1649", 2026, <https://github.com/modelcontextprotocol/modelcontextprotocol/pull/2127>.

[MCP-SPEC] Model Context Protocol Project, "Model Context Protocol Specification", 2025, <https://modelcontextprotocol.io>.

[MCP-META] Model Context Protocol Project, "Model Context Protocol, Base Protocol: General fields (the _meta property)", November 2025, <https://modelcontextprotocol.io/specification/2025-11-25/basic>.

[SCHEMA-ORG-LB]
Schema.org Community Group (W3C), "Schema.org: LocalBusiness", <https://schema.org/LocalBusiness>.

Acknowledgements

The authors thank the authors of SEP-1649 for the core Server Card specification on which this profile depends.

Authors' Addresses

David Soden
Beacon Spec
Email: david@beaconspec.com

Duane Walker
Beacon Spec
Email: duane@beaconspec.com