

Are LLMs and the Model Context Protocol Sufficient for Automating Web-Based Information Processing?

Stephen Cranefield¹

¹*School of Computing, University of Otago, Dunedin, New Zealand*

Abstract

More than two decades ago, the concept of the Semantic Web was motivated by a vision of personal assistant agents that fulfil users' goals by locating, accessing and reasoning about information and services on the web. Until the recent advent of agents that can act on instructions generated by large language models (LLMs), progress towards this vision has been slow. Now, the natural language understanding abilities of LLMs and their emerging ability to generate and follow instructions suggest that LLM-powered agents may provide a path towards developing such general-purpose assistant agents.

This paper presents a case study of automating a multi-step web-based information seeking and filtering task using a large language model provided with tool access via the Model Context Protocol (MCP) and information about relevant web resources. We found this was possible, but only by providing the LLM with detailed information about how to use the resources. We discuss the reasons for this and how this requirement could be eliminated by providing discovery mechanisms and web page usage information intended for LLMs. We also propose the development of higher-level models of web resources in terms of information processing goals or tasks.

Keywords

Information processing, World Wide Web, Large Language Models (LLMs), Tools, Model Context Protocol (MCP)

1. Introduction

Since the birth of the World Wide Web in the 1990s, it has become an indispensable means for finding information, buying products and services, submitting service requests, communicating with our social networks and satisfying many other information processing and communication goals. Although the web was designed for people, it is no surprise that researchers and industry have long sought to develop software to extract information from the web [1] or to serve as a personal assistant to automate information-gathering from local and web-based resources [2]. As far back as 2001, a popular science magazine article introduced the concept of the Semantic Web by imagining the existence of personal assistant agents that fulfil users' goals by interacting with other assistants and web resources [3]. However, until the advent of pre-trained large language models (LLMs) and their connection with memory, planners, tools, etc., to form "LLM Agents" [4, 5, 6], little progress was made towards this vision.

Recently, researchers from the fields of Web Architecture and the Web of Things, the Semantic Web and Linked Data, and Autonomous Agents and Multi-Agent Systems have come together to study how agents can use and be part of the modern web, considered as a "homogeneous hypermedia fabric that interconnects everything—devices, physical objects, documents, or abstract concepts" [7, 8]. This motivates the development of *hypermedia multi-agent systems*: "systems of agents able to perceive, decide, and act through the Web to achieve goals".¹

On the other hand, a huge amount of activity is occurring in the IT industry to develop tools that enable LLM-powered agents to interact with each other and both local and online resources. Several specifications have been developed for how these interactions could work in a plug-and-play fashion

The Second International Workshop on Hypermedia Multi-Agent Systems (HyperAgents 2025), in conjunction with the 28th European Conference on Artificial Intelligence (ECAI 2025); October 26, 2025, Bologna, Italy

✉ stephen.cranefield@otago.ac.nz (S. Cranefield)

🌐 <https://www.otago.ac.nz/school-of-computing/our-people/stephen-cranefield> (S. Cranefield)

🆔 0000-0001-5638-1648 (S. Cranefield)

© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

¹<https://ecai2025.hyperagents.org/cfp>

[9]. In this paper we focus on the Model Context Protocol (MCP),² which is currently generating a large amount of activity, including a flood of blog posts about its architecture and use, and the development of various industry and community “MCP servers” that provide LLMs with one or more tools to perform computations or provide access to other software services. The MCP is based on a more traditional architecture than that envisaged for hypermedia multi-agent systems: it allows client-server communication (via JSON-RPC 2.0) between the application hosting the LLM and one or more MCP servers that provide tools.

In this paper, we evaluate how much effort is required (if at all possible) to prompt an LLM equipped with relevant MCP servers to automate a specific web-based information-seeking filtering task presented by Berners-Lee et al. [3]. We found that this was possible after iterated prompt development resulting in the provision of detailed usage information about the required web resources that was not available to the agent by examining the HTML of those web resources. This level of guidance is not realistic in practice—it would be more efficient to do the task yourself unless the task was frequently repeated. We present our observations about the causes of this problem and suggestions about what is missing to achieve general-purpose web information agents based on LLMs.

2. Case Study

In a classic article from 2001, Berners-Lee et al. [3] introduce the concept of the Semantic Web in the context of two siblings (Lucy and Peter) seeking to jointly organise medical treatment for their mother. Lucy has just taken her mother to see a doctor, who says that a specialist appointment is needed and then a series of physical therapy sessions. Lucy uses an assistant agent on her “handheld Web browser” to set up the appointments. The agent contacts the doctor’s agent to obtain details of the required treatment and then prepares a shortlist of providers by looking up several online lists, and selecting those that are approved by the mother’s insurance plan, within a certain distance of her home, and highly rated on trusted rating services. It then obtains available appointment times from the provider’s web sites and cross references these with Lucy’s and Pete’s free times, as they will share the chauffeuring duties.

The article highlights how the Semantic Web would enable all parties’ agents to access information from web sites and communicate with each other using structured knowledge representations that make reference to online *ontologies* defining the concepts and relationship between them, as well as rules for reasoning about them. It also proposed that a service discovery framework would allow agents and web-based services to advertise their functionality in semantic terms so they can be discovered and made use of dynamically to solve problems such as the one above.

It has been nearly 25 years since that article was published, and while Semantic Web technologies are now used in niche areas, the vision of widespread Semantic Web enhancement of existing human-oriented web sites has not come to pass, arguably due to the high effort required to develop ontologies and annotate existing web sites with semantic information. Furthermore, the domain-independent planning competence required by the agents in the scenario has not yet become available, at least for consumer applications. However, the rapid advancement of large language models (LLMs) promises to solve both of these limitations: (i) their high level skills at understanding and generating natural language raises the question of whether they can mediate between agents and services that use different terminology, (ii) and their ability to decompose instructions into sequential task suggests they may eventually be able perform the type of task decomposition illustrated in the scenario (although we acknowledge that the general planning ability of LLMs has limitations [10, 11]). Furthermore, an explosion of frameworks and tools for LLM agents has made it possible for LLM agents to execute steps of a plan that require interaction with external resources.

We have therefore developed an implementation of a simplified version of the scenario above, using the Model Context Protocol (MCP), which Ehtesham et al. [9] position in Stage 1 of their roadmap

²<https://modelcontextprotocol.io/>

for interoperability between LLM-powered agents and external services and tools. The Agent Communication Protocol (ACP), Agent-to-Agent Protocol (A2A) and Agent Network Protocol (ANP) are positioned above MCP at increasingly higher levels. Our aim is to evaluate what is possible with the fast-emerging MCP. There are already a large number of commercially and community developed MCP servers available, and MCP is supported by a number of environments for running LLMs with tool access (“MCP Hosts”), such as Claude for Desktop and LM Studio. We identify some limitations of this approach and make suggestions about how these can be eliminated or mitigated.

We used LM Studio version 0.3.17, with the LLM qwen2.5-7b-instruct, one of three LLMs available for LM Studio that has “native” tool support. We installed the Fetch MCP server³, which fetches a web page and converts it to Markdown, the MCP Http Server⁴, which provides HTTP GET, POST, PUT and DELETE tools, and the Mapbox MCP Server⁵, which includes a forward geocoding tool and a matrix tool that “calculates travel times and distances between multiple points”.

Our simplified version of the scenario from Berners-Lee et al. [3] involves prompting an LLM to:

- Find all physiotherapists in Dunedin (New Zealand) and then either filter the list to keep only those that are listed by the nib insurance company as FirstChoice medical providers, or if none are, keep the full list.
- Ask the user for their address.
- Find the provider that is closest to the address
- Ask the user to phone the selected provider for the first three available appointment times, and then enter these in this chat.
- Check the user’s calendar to see at which of the appointment times the user is listed as free.

These instructions, developed through a trial and error process, have already decomposed the user’s goal into sequential steps to reduce the demand for planning by the LLM. No tool was provided for calendar access in the last step, but without further instruction the LLM asked the user to provide their free times and then selected the best appointment time.

The following information about resources was also provided in the prompt:

- https://nzdirectory.co.nz/search?query=<BUSINESS_SEARCH_TERM>®ion=<REGION>&city=<CITY>. This is the search URL pattern for a directory of businesses in New Zealand. For <BUSINESS_SEARCH_TERM> you can use a sufficiently discriminating prefix of the business or service type. Given a <CITY> you MUST choose <REGION> to be whichever of the following New Zealand regions contains the city: Northland, Auckland, Waikato, Bay of Plenty, Gisborne, Hawke’s Bay, Taranaki, Manawatu-Whanganui, Wellington, Tasman, Nelson, Marlborough, West Coast, Canterbury, Otago and Southland.
Use your background knowledge to choose the correct region for the city. You must convert the region and city to lowercase when including them in the search URL. Performing a GET on the search URL returns an HTML page for human viewing, so use a tool that converts the result to a more concise format like Markdown.
- https://www.nib.co.nz/find-a-provider/api/recommendations?searchTerm=<SEARCH_TERM> is the search URL for the insurance company nib’s FirstChoice providers. The search term can be a sufficiently distinguishing prefix of the medical speciality of interest (in lowercase). The results are in JSON, so obtain the results using a tool that does not modify the format. Examine the JSON to find the details for the returned providers.

With this level of detail, the LLM was able to successfully reach the end of the simplified scenario. However, further prompt engineering and the introduction of a response verifier agent would be needed to create a more reliable solution.

³<https://github.com/modelcontextprotocol/servers/tree/main/src/fetch>

⁴<https://github.com/one-matrix/mcp-http>

⁵<https://github.com/mapbox/mcp-server>

The need for such a detailed prompt highlights the need for solutions to the following problems:

Tool documentation, selection and calling. During the iterative development and testing of the prompt above, we found that the Mapbox MCP Server’s matrix tool does not provide the MCP client with any information about the units of the distances it returns. This did not affect the task performance, which only required finding the *closest* nib FirstSelect physiotherapist. However, it highlights the need for a more expressive approach to documenting tools compared to MCP’s reliance on function names and a textual description. Furthermore, we tested a number of alternative tools for making HTTP requests and only those that LM Studio and qwen2.5-7b-instruct used successfully were made available to LM Studio for subsequent attempts. We also found it necessary to include hints about tool use in the prompt. Tool learning, which includes the problems of tool selection and calling, is an active research field [12, 13] and should alleviate these issues in the future.

Discoverability and informed selection of services. Ideally, there should be no need to provide web resources in the LLM prompt as these should be able to be discovered dynamically by the LLM agent from registries and/or catalogues. This could be achieved by extending the existing web indexing and search infrastructure, for example, by developing a web standard in collaboration with search engine providers that enables web search results to be enhanced with service descriptions. These could be either in natural language, under the assumption that LLM agents will be able interpret these, or use a semantic representation such as a Web of Things Thing Description [14]. It is an open question whether the latter approach is likely to reach widespread use in the mainstream web or whether a textual description will prove to be sufficient as the capabilities of LLMs improve.

Alternatively, a separate ecosystem of registries and catalogues could be developed. The original vision for web services included such infrastructure, which never came about (at least for public use). However, the promise of LLM-powered agents as universal clients of web resources may create sufficient demand for such infrastructure to be viable.

Given that multiple web sites may provide the same information, perhaps with different search mechanisms, levels of detail and information currency, there is also a need to support user comments, ratings and usage tips for specific sites

Navigation from a single entry point. Once a useful web site has been identified, a key principle of the web is that it should be possible to navigate the site by following hyperlinks that have sufficient context to understand their purpose. This is Fielding’s concept of “hypermedia as the engine of application state” [15], abbreviated as HATEOAS in REST API circles. While navigation through web sites has become natural for people browsing the web, and including links labelled with their relation type in REST API responses is considered best practice [16], understanding how to navigate through web sites is challenging for LLMs when accessing web pages designed for people to read. The number of tokens consumed when a web page in HTML is included in a prompt can overwhelm an LLM (especially when there are large amounts of JavaScript), hence some web access tools such as the Fetch MCP server convert web pages to Markdown. Furthermore, the allowed options for drop-down lists in the input fields for forms may be dynamically created by JavaScript. This may also be the case for the URL template used when the form is submitted.

For example, the NZ Directory search URL given in the prompt above cannot be readily inferred (at least by this author) by examining the search page at <https://nzdirectory.co.nz/>. It only becomes evident when the search results are returned. Furthermore, the accepted values for the region input element are not present in the HTML (including scripts). Only once a region has been entered, does the city input element appear, with its options populated using a POST request to the server. This dynamic modification of a web page cannot be predicted or observed by an LLM. Therefore, our prompt specified a standard list of New Zealand regions to choose from and stated that the selected region must be provided in the search URL’s query parameters in lowercase (otherwise no results are returned). Similarly, the nib “Find a Provider” web page has search input fields for the healthcare provider name

or speciality and for a suburb or region, and both are populated dynamically by GET requests to the server every time a character is typed in the field. However, it turns out the second of these inputs is not required as a search URL query parameter, so we did not prompt the LLM to use it.

This impediment to LLM agents' use of modern dynamic web sites could be addressed in several ways:

- Use an MCP server with a richer interface for an LLM to interact with and examine web sites, such as the browser-use MCP server⁶ [17]. This and other similar tools run a headless browser, which allows interaction and interrogation at the DOM level as well as screenshot capture. This approach would allow a suitably competent LLM to explore a web site by trial and error (and user-supplied heuristics) with the aim of satisfying the user's goal.
- Build separate web resources, parallel to the human-facing ones, that are optimised for use by LLM agents. This is envisaged by a proposal for a Markdown file, `llms.txt`, at the top level of a web site to provide LLMs accessing the site with "brief background information, guidance, and links to detailed markdown files" [18]. The links should be to LLM-friendly Markdown versions of existing web pages on the site.
- Alternatively, vastly expand the information that is available from web sites via web services. This and the option above seem unlikely to be attractive to organisations until there is widespread consumer use of LLM web agents, and in the meantime there will still be a need for agents that can use human-oriented sites.
- Publish web site documentation designed to help agents navigate their way through the site. This could include structured information such as site maps, stand-alone documents in natural language and machine-readable annotations embedded within web pages. A limiting factor of this approach is that it will take extra time and effort when developing and maintaining a web site, which may be difficult for organisations to justify.
- Develop web standards (in collaboration with the creators of web page development frameworks) for the use of server-push technology⁷ to provide navigational information to be to LLM agents as the content is dynamically updated by scripts. This would remove the requirement to maintain documentation for the LLM separately from the web site front end.

Reasoning at a goal or task level. The interface between an MCP tool and an LLM is specified at a low level, in terms of input schemas, a (hopefully meaningful) tool name and a "human-readable description". Any further information about when it would be useful to use each tool must be provided in a system or user LLM prompt. Beyond the scope of MCP, there is also a lack of description formats (other than natural language) and ontologies for the information processing *purpose* of web resources. Some steps in this direction have been made in the Web of Things Thing Description information model [14], which provides a format for describing (amongst other features) the affordances offered by a web-connected *thing* to inspect properties, perform actions and publish events. Hafiene et al. [19] discuss the dynamic use of information represented using the Hypermedia Multi-Agent Systems (hMAS) ontology to enable an agent to discover a description of *thing*'s affordance that will achieve one its goals.⁸

The abstraction of resources and tools in terms of goals or tasks seems an important means of facilitating high planning for the combined use of various tools and resources. Furthermore, reasoning at this level would allow the use of techniques like Hierarchical Task Network (HTN) planning [20] to make use of predefined decompositions of tasks into subtasks. This would allow successful recipes for combining multiple resources and tools to be published and considered for use by LLM agents equipped with a planner (or perhaps by the LLM itself). However, research is needed into goal or task representations suited to the context of using the web to find and create information (such as our case

⁶<https://medium.com/towards-agi/how-to-setup-and-use-browser-use-mcp-server-8d0725440f31>

⁷We note that MCP supports Server-Sent Events (SSE) for delivering notifications from tools to clients.

⁸The scenario also involves the agent joining a regulated agent organisation as part of the scenario, but this is beyond the scope of our discussion.

study of finding and selecting medical practitioners and making appointments). An HTN planning approach to information processing on a personal computer was proposed by the author [21]. This requires a data model of the problem domain so that information flows into, out of, and between tasks involving different resources can be modelled. This cannot be provided for general-purpose agents that may be asked to perform any web information processing task. However, textual descriptions may suffice for LLM-powered agents.

3. Conclusion

We have presented a case study of automating a multi-step information seeking and filtering task using a large language model, tool access via MCP, and information about relevant web resources that require providing search inputs. Our aim was to evaluate how close this current technology could get to the capabilities envisaged for Semantic Web agents by Berners-Lee et al. [3] in 2001, under the premise that LLMs' language understanding abilities could remove the need for semantic mark-up on web sites. We were able to implement our case study, but detailed guidance about resource use was required in the LLM prompt. Our conclusion is that to facilitate the use of the web by LLM-powered agents, further advances are needed in the areas of resource registries and catalogues and web site usage information intended for LLMs. Furthermore we believe that higher level abstractions of web resources are needed that model the information-processing goals or tasks they can be used to achieve and how these can be composed to solve complex goals.

References

- [1] O. Etzioni, M. Banko, S. Soderland, D. S. Weld, Open information extraction from the web, *Communications of the ACM* 51 (2008) 68–74. doi:10.1145/1409360.1409378.
- [2] P. Cohen, A. Cheyer, E. Horvitz, R. El Kaliouby, S. Whittaker, On the future of personal assistants, in: *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, Association for Computing Machinery, 2016, p. 1032–1037. doi:10.1145/2851581.2886425.
- [3] T. Berners-Lee, J. Hendler, O. Lassila, The semantic web, *Scientific American* 284 (2001) 34–43. URL: <https://static.scientificamerican.com/sciam/cache/file/394EDA92-D03F-4110-B5AA4465CE486800.pdf#page=25>.
- [4] T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N. V. Chawla, O. Wiest, X. Zhang, Large language model based multi-agents: a survey of progress and challenges, in: *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, 2024. doi:10.24963/ijcai.2024/890.
- [5] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, et al., A survey on large language model based autonomous agents, *Frontiers of Computer Science* 18 (2024) 1–26.
- [6] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, M. Zhang, J. Wang, S. Jin, E. Zhou, R. Zheng, X. Fan, X. Wang, L. Xiong, Y. Zhou, W. Wang, C. Jiang, Y. Zou, X. Liu, Z. Yin, S. Dou, R. Weng, W. Qin, Y. Zheng, X. Qiu, X. Huang, Q. Zhang, T. Gui, The rise and potential of large language model based agents: a survey, *Science China Information Sciences* 68 (2025). doi:10.1007/s11432-024-4222-0.
- [7] O. Boissier, A. Ciortea, A. Harth, A. Ricci, Autonomous Agents on the Web (Dagstuhl Seminar 21072), *Dagstuhl Reports* 11 (2021) 24–100. doi:10.4230/DagRep.11.1.24.
- [8] O. Boissier, A. Ciortea, A. Harth, A. Ricci, D. Vachtsevanou, Agents on the Web (Dagstuhl Seminar 23081), *Dagstuhl Reports* 13 (2023) 71–162. doi:10.4230/DagRep.13.2.71.
- [9] A. Ehtesham, A. Singh, G. K. Gupta, S. Kumar, A survey of agent interoperability protocols: Model context protocol (MCP), agent communication protocol (ACP), agent-to-agent protocol (A2A), and agent network protocol (ANP), 2025. arXiv:2505.02279.

- [10] K. Stechly, K. Valmeekam, S. Kambhampati, Chain of thoughtlessness? an analysis of CoT in planning, 2024. URL: <http://arxiv.org/abs/2405.04776>, arXiv:2405.04776 [cs].
- [11] S. Kambhampati, K. Valmeekam, L. Guan, K. Stechly, M. Verma, S. Bhambri, L. Saldyt, A. Murthy, LLMs can't plan, but can help planning in LLM-Modulo Frameworks, 2024. URL: <http://arxiv.org/abs/2402.01817>, arXiv:2402.01817 [cs].
- [12] Y. Qin, S. Hu, Y. Lin, W. Chen, N. Ding, G. Cui, Z. Zeng, X. Zhou, Y. Huang, C. Xiao, C. Han, Y. R. Fung, Y. Su, H. Wang, C. Qian, R. Tian, K. Zhu, S. Liang, X. Shen, B. Xu, Z. Zhang, Y. Ye, B. Li, Z. Tang, J. Yi, Y. Zhu, Z. Dai, L. Yan, X. Cong, Y. Lu, W. Zhao, Y. Huang, J. Yan, X. Han, X. Sun, D. Li, J. Phang, C. Yang, T. Wu, H. Ji, G. Li, Z. Liu, M. Sun, Tool learning with foundation models, *ACM Computing Surveys* 57 (2024). doi:10.1145/3704435.
- [13] C. Qu, S. Dai, X. Wei, H. Cai, S. Wang, D. Yin, J. Xu, J.-r. Wen, Tool learning with large language models: a survey, *Frontiers of Computer Science* 19 (2025) 198343. doi:10.1007/s11704-024-40678-2.
- [14] World Wide Web Consortium, Web of Things (WoT) Thing Description 1.1, W3C Recommendation, <https://www.w3.org/TR/wot-thing-description11/>, 2023.
- [15] R. T. Fielding, Architectural Styles and the Design of Network-based Software Architectures, PhD dissertation, University of California, Irvine, 2000. URL: <https://ics.uci.edu/~fielding/pubs/dissertation/top.htm>.
- [16] M. Fowler, Richardson Maturity Model, <https://martinfowler.com/articles/richardsonMaturityModel.html>, 2010. Accessed: 2025-07-01.
- [17] E. Milošević, How to setup and use Browser Use MCP Server, Towards AGI (Medium), 2025. URL: <https://medium.com/towards-agi/how-to-setup-and-use-browser-use-mcp-server-8d0725440f31>.
- [18] J. Howard, The /llms.txt file, <https://github.com/AnswerDotAI/llms-txt>, 2024.
- [19] N. Hafiene, L. G. Nardin, O. Boissier, Knowledge level support for programming agents to interact in regulated online forums, in: S. Cranefield, L. G. Nardin, N. Lloyd (Eds.), *Coordination, Organizations, Institutions, Norms, and Ethics for Governance of Multi-Agent Systems XVII*, Springer Nature Switzerland, Cham, 2025, pp. 100–112.
- [20] I. Georgievski, M. Aiello, HTN planning: Overview, comparison, and beyond, *Artificial Intelligence* 222 (2015) 124–156. doi:10.1016/j.artint.2015.02.002.
- [21] S. Cranefield, E. Moreale, B. McKinlay, M. Purvis, Automating the interoperation of information processing tools, in: *32nd Annual Hawaii International Conference on System Sciences*, IEEE Computer Society, 1999. doi:10.1109/HICSS.1999.773089.

Declaration on Generative AI

While the use of a generative AI tool was the subject of this paper, the author has not employed any generative AI tools to write the paper.