Technische Hochschule
Brandenburg
University of
Applied Sciences

Exposé für:

MCP-Apps zur markenkonformen Einbindung redaktioneller Inhalte in KI-Chat-Interfaces: Konzeption und technische Erprobung am Beispiel des rbb24-Content-Bereichs

Bachelor-Arbeit

zur Erlangung des Grades Bachelor of Science in Medieninformatik
des Fachbereichs Wirtschaft an der
Technischen Hochschule Brandenburg

vorgelegt von:
Shirley Elane Zaldivar Martinez

Betreuer: Sebastian Kreideweiß, MSc.
Zweitgutachter: Martin Sanfilippo

Zaldivar Martinez, Shirley Elane Bachelor of Science Studiengang Medieninformatik	Abgabe: Mitte Januar 2027

Brandenburg/H., den 15. September 2026

Inhaltsverzeichnis

Abbildungsverzeichnis

Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.

Tabellenverzeichnis

Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.

Abkürzungsverzeichnis

CMS	Content-Management-Systeme
HS	Hochschule
THB	Technische Hochschule Brandenburg
API	Application Programming Interface
JSON	JavaScript Object Notation
MCP	Model Context Protocol
PoC	Proof of Concept
RPC	Remote Procedure Call
SDK	Software Development Kit
SEP	Specification Enhancement Proposal
UI	User Interface

1 Problemstellung

Der Zugang zu redaktionellen Inhalten verschiebt sich zunehmend von der klassischen, direkten Website-Nutzung hin zu einer Vermittlung über KI-Agenten und KI-gestützten Suchsysteme. Nach Angaben von SparkToro auf Basis von Similarweb-Clickstream-Daten endeten im Zeitraum Januar bis April 2026 68,01 % aller Google-Suchen in den USA ohne Klick auf ein externes Ergebnis. Dies ist ein Anstieg um 7,56 % gegenüber 2024. Durch diesen "Zero-Click"-Effekt zeichnet sich mit dem Aufkommen sogenannter Agentic Search ein noch grundlegenderer Wandel ab: KI-Agenten recherchieren, bewerten und nutzen Inhalte zunehmend eigenständig, ohne dass ein Mensch die ursprüngliche Quelle überhaupt noch besucht.

Für Anbieter journalistischer Inhalte, insbesondere für öffentlich-rechtliche Rundfunkanstalten wie dem rbb, entsteht daraus ein strukturelles Problem. Wenn redaktionelle Inhalte primär über KI-Agenten vermittelt werden, droht der ursprüngliche Anbieter auf eine passiv zitierte Datenquelle reduziert zu werden, ohne eigene Markenpräsenz, redaktionelle Einordnung oder Interaktionsmöglichkeit im eigentlichen Nutzungsmoment. Das Model Context Protocol (MCP), über das viele KI-Agenten externe Werkzeuge und Datenquellen anbinden, war ursprünglich auf rein textuelle bzw. strukturierte Antworten beschränkt und bot keine Möglichkeit, eigene, gestaltbare Oberflächen in die Interaktion mit dem Agenten einzubringen. Mit MCP-Apps, der offiziellen Erweiterung des MCP, existiert seit kurzem ein technischer Mechanismus, der dieses Problem grundsätzlich adressieren könnte. Bislang liegt jedoch, nach vorliegendem Kenntnisstand, keine dokumentierte Umsetzung dieses Mechanismus für den rbb vor.

2 Erkenntnisinteresse

Es besteht Interesse an der Erkenntnis, ob und wie sich MCP-Apps praktisch nutzen lassen, damit ein öffentlich-rechtlicher Anbieter wie rbb24 auch im agentenvermittelten Zugriff eine aktive, markenkonforme und interaktive Präsenz behält, statt rein textuell reduziert zu werden. Im Zentrum steht dabei weniger die technische Machbarkeit einer einzelnen Komponente, sondern vielmehr die Frage, wie sich ein vollständiger, spezifikationskonformer Anwendungsfall (von der Anforderungsanalyse über die Architektur bis zur lauffähigen Implementierung) unter realen Rahmenbedingungen (bestehende Backend-Systeme, Sicherheits- und Markenvorgaben) umsetzen lässt.

3 Fragestellung

Die vorliegende Arbeit widmet sich primär folgender Fragestellung:

Wie lässt sich ein Proof of Concept einer MCP-App für den öffentlich-rechtlichen Rundfunk anhand ausgewählter redaktioneller Anwendungsfälle von rbb24 konzipieren und implementieren, welcher die architektonischen und sicherheitsbezogenen Vorgaben der MCP-Apps-Spezifikation (SEP-1865) einhält?

Ergänzend ergeben sich folgende Teilfragen:

- Welche funktionalen und nicht-funktionalen Anforderungen lassen sich aus ausgewählten redaktionellen Anwendungsfällen (Radio-Player, Verkehrskarte, Themen-Dashboard, Wetter) für eine solche MCP-App ableiten?
- Wie lässt sich eine solche Anwendung unter Einbindung bestehender Backend-Systeme (CMS, Streaming-Infrastruktur) architektonisch umsetzen?
- Inwiefern erfüllt der resultierende Prototyp die Vorgaben der MCP-Apps-Spezifikation, und wie lässt er sich hinsichtlich technischer Qualität (Wartbarkeit, Erweiterbarkeit) sowie Nutzungserfahrung bewerten?

4 Ziele und Hypothesen

Ziele:

- Erhebung und Systematisierung der funktionalen und nicht-funktionalen Anforderungen an eine MCP-App für ausgewählte redaktionelle Anwendungsfälle von rbb24.
- Entwurf einer Architektur, die diese Anforderungen unter Einhaltung der MCP-Apps-Spezifikation sowie bestehender Backend-Systeme umsetzt.
- Prototypische Implementierung eines Proof of Concept auf Basis des mcp-ui-SDK-Ökosystems.
- Evaluation des Prototyps hinsichtlich Spezifikationskonformität, technischer Qualität (Wartbarkeit, Erweiterbarkeit) und Nutzungserfahrung.

5 Theoriebezug

Die Arbeit stützt sich fachlich auf drei Bereiche:

1. die technische Spezifikation des Model Context Protocol und seiner Erweiterung MCP-Apps (SEP-1865), insbesondere das Ressourcenmodell (ui://-Ressourcen), welches bidirektionale JSON-RPC-Kommunikationsmodell sowie das iframe-basierte Sicherheitsmodell verwendet
2. das praktische mcp-ui-SDK-Ökosystem als Referenzimplementierung dieser Spezifikation sowie vergleichbare Ansätze (OpenAI Apps SDK, Google A2UI) zur Einordnung von MCP-Apps im breiteren Feld generativer Benutzeroberflächen ("GenUI")
3. die medienökonomische bzw. mediennutzungsbezogene Einordnung des Wandels von klassischer Websitenutzung hin zu agentenvermitteltem Content-Zugriff (Zero-Click-Search, Agentic Search) als gesellschaftlich-technischer Rahmen, der die praktische Relevanz der Arbeit begründet.

6 Forschungsstand

MCP-Apps wurde nach einem ersten Vorschlag im November 2025 am 26. Januar 2026 offiziell als Erweiterung der MCP-Spezifikation verfügbar. Zum Zeitpunkt 28.07.206 wurde final eine grundlegende überarbeitete Version der MCP-Kernspezifikation veröffentlicht, welche das zugrunde liegende Protokoll auf ein zustandsloses Modell umstellt und MCP-Apps vom experimentellen Status in den offiziellen Extensions Track überführt. Die Spezifikation befindet sich damit nachweislich in aktiver Weiterentwicklung, was bei der Wahl des konkreten Implementierungsstands berücksichtigt und im weiteren Verlauf der Arbeit transparent dokumentiert wird.

Praktisch umgesetzt wird der Standard unter anderem durch das mcp-ui-Ökosystem, welches dem Standard historisch vorausging und dessen Entwicklung mitgeprägt hat. Zudem wird der Standard durch Apps SDK von OpenAI erweitert, welcher MCP-Apps als Basis übernimmt und um Zusatzfunktionen ergänzt. Mit Googles A2UI existiert ein alternativer, deklarativer und plattformübergreifend nativer Ansatz für KI-generierte Benutzeroberflächen. Wissenschaftliche Arbeiten zum Model Context Protocol befassen sich bislang vor allem mit dessen Sicherheitsmodell und genereller Architektur, jedoch nicht mit der noch sehr jungen MCP-Apps-Erweiterung im Detail. Zum Rückgang klassischer, klickbasierter Websitenutzung liegen aktuelle Branchendaten vor (SparkToro/Similarweb 2026), jedoch keine peer-reviewte Forschung zu den betrachteten Zeiträumen. Eine dokumentierte Anwendung von MCP-Apps im Kontext des öffentlich-rechtlichen Rundfunks konnte im Rahmen der bisherigen Recherche nicht identifiziert werden, was die praktische Relevanz und den explorativen Beitrag der vorliegenden Arbeit begründet.

7 Methode

Die Arbeit folgt einem gestalterisch-konstruktiven, Artefakt basierten Vorgehen (Proof-of-Concept-Entwicklung) in vier Phasen:

- Recherche & Anforderungsanalyse (Wochen 1–3): Aufarbeitung der MCP-Apps-Spezifikation und des mcp-ui-Ökosystems, Erhebung der Anforderungen an die rbb24-Anwendungsfälle.
- Architekturentwurf (Wochen 4–6): Entwurf der Systemkomponenten und Schnittstellen (CMS, Streaming) sowie des Sicherheitskonzepts.
- Implementierung (Wochen 7–9): Prototypische Umsetzung auf Basis der MCP-Apps-Spezifikation und des mcp-ui-SDK.
- Evaluation (Wochen 10-12): Prüfung der Spezifikationskonformität, technische Bewertung sowie qualitative Einschätzung der Nutzungserfahrung, z. B. mittels leitfadengestützter Tests mit ausgewählten Nutzern.

8 Material

Zum Einsatz kommt folgendes Material:

- MCP-Apps-Spezifikation SEP-1865 sowie die zugehörige Spec-Datei apps.mdx (Stand 26.01.2026), ergänzt um den Release Candidate der MCP-Kernspezifikation vom 28.07.2026.
- Das mcp-ui-SDK-Ökosystem (Server- und Client-Pakete, TypeScript).
- Interne Materialien des rbb: der Pitch "MCP-Apps für ARD & RBB" sowie bestehende Backend-Systeme (Content-Management-System, Streaming-Infrastruktur) und vorhandene UI-Bausteine/Corporate-Design-Vorgaben.
- Technischer Stack der Implementierung: TypeScript, React, Node.js, Tailwind CSS, ggf. GraphQL (abhängig von den tatsächlich genutzten rbb-Schnittstellen).
- Vergleichsmaterial zu verwandten Ansätzen: OpenAI Apps SDK und Google A2UI.

9 Gliederungsentwurf

1 Einleitung
- 1.1 Motivation und Problemstellung
- 1.2 Zielsetzung und Forschungsfrage
- 1.3 Methodisches Vorgehen

2 Grundlagen
- 2.1 MCP - Model Context Protocol
- 2.2 Die MCP-Apps-Spezifikation
  - 2.2.1 ui://-Ressourcen
  - 2.2.2 Resource Discovery / Verknüpfung von Tools mit UI
  - 2.2.3 Kommunikationsmodell (bidirektionales JSON-RPC)
  - 2.2.4 Sicherheitsmodell und iframe-Sandboxing
- 2.3 Das mcp-ui-Ökosystem
  - 2.3.1 Entstehung und Verhältnis zur MCP-Apps-Spezifikation
  - 2.3.2 Serverseitige SDKs
  - 2.3.3 Clientseitige SDKs
- 2.4 KI-Agenten als neuer Zugangskanal zu redaktionellen Inhalten
- 2.5 Abgrenzung zu verwandten Ansätzen

3 Anforderungsanalyse
- 3.1 Ausgangssituation bei rbb24
- 3.2 Anwendungsfälle
  - 3.2.1 Radio-Player
  - 3.2.2 Verkehrskarte
  - 3.2.3 Themen-Dashboard
  - 3.2.4 Wetter
- 3.3 Funktionale Anforderungen
- 3.4 Nicht-funktionale Anforderungen

4 Konzept und Architektur
- 4.1 Systemüberblick
- 4.2 Architektur des MCP-Servers
- 4.3 Anbindung bestehender Backend-Systeme (CMS, Streaming)
- 4.4 Gestaltung der UI-Ressourcen im Sandboxing-Modell
- 4.5 Umsetzung von Markendesign / Corporate Identity
- 4.6 Sicherheitskonzept
- 4.7 Zusammenfassung des Architekturentwurfs

5 Implementierung
- 5.1 Technischer Rahmen und eingesetzte Werkzeuge

10 Vorläufiges Literaturverzeichnis

Add UI to your MCP server – Plugins. (o. J.). OpenAI Developers. Abgerufen 3. September 2026, von https://developers.openai.com/plugins/build/chatgpt-ui

Ashley (OpenAI), A. P. (Maintainer), Olivier Chafik (Maintainer), Ido Salomon (MCP-UI), Liad Yosef (MCP-UI), Nick Cooper (Core Maintainer), Sean Strong (Anthropic), Jerome Swannack (Anthropic), Alexi Christakis (OpenAI), Bryan. (2025, November 21). *MCP Apps: Extending servers with interactive user interfaces*. Model Context Protocol Blog. https://blog.modelcontextprotocol.io/posts/2025-11-21-mcp-apps/

Barenholtz, A. L. (2026, Juni 10). *Zero-Click Marketing: What the 2026 Data Means*. Similarweb. https://aisearch.similarweb.com/blog/zero-click-marketing/

Fishkin, R. (2026, Juni 9). In 2026, Less than One Third of Google Searches Still Send a Click. *SparkToro*. https://sparktoro.com/blog/in-2026-less-than-one-third-of-google-searches-still-send-a-click/

Hasan, M. M., Li, H., Fallahzadeh, E., Rajbahadur, G. K., Adams, B., & Hassan, A. E. (2026). *Model Context Protocol (MCP) at First Glance: Studying the Security and Maintainability of MCP Servers* (arXiv:2506.13538). arXiv. https://doi.org/10.48550/arXiv.2506.13538

Hou, X., Zhao, Y., Wang, S., & Wang, H. (2025). *Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions* (arXiv:2503.23278). arXiv. https://doi.org/10.48550/arXiv.2503.23278

HTML Standard. (o. J.). Abgerufen 3. September 2026, von https://html.spec.whatwg.org/multipage/browsers.html#sandboxing

HTML5 Security—OWASP Cheat Sheet Series. (o. J.). Abgerufen 3. September 2026, von https://cheatsheetseries.owasp.org/cheatsheets/HTML5_Security_Cheat_Sheet.html

Introducing the Model Context Protocol. (2024, November 25). https://www.anthropic.com/news/model-context-protocol

JSON-RPC 2.0 Specification. (o. J.). Abgerufen 3. September 2026, von https://www.jsonrpc.org/specification

Maintainer), D. S. P. (Lead M., Den Delimarsky (Lead. (2026, Mai 21). *The 2026-07-28 MCP Specification Release Candidate*. Model Context Protocol Blog. https://blog.modelcontextprotocol.io/posts/2026-07-28-release-candidate/

Maintainers, M. C. (2026, Januar 26). *MCP Apps—Bringing UI Capabilities To MCP Clients*. Model Context Protocol Blog. https://blog.modelcontextprotocol.io/posts/2026-01-26-mcp-apps/

MCP-UI. (2026). *MCP-UI-Org/mcp-ui* [TypeScript]. https://github.com/MCP-UI-Org/mcp-ui (Ursprünglich erschienen 2025)

MCP-UI | Interactive UI for MCP. (2026a, März 9). MCP-UI. https://mcpui.dev/

MCP-UI | Interactive UI for MCP. (2026b, März 9). MCP-UI. https://mcpui.dev/

MCP-UI | Interactive UI for MCP. (2026c, März 9). MCP-UI. https://mcpui.dev/

MCP-UI | Interactive UI for MCP. (2026d, März 9). MCP-UI. https://mcpui.dev/

MCP-UI | Interactive UI for MCP. (2026e, März 9). MCP-UI. https://mcpui.dev/

modelcontextprotocol. (o. J.). *GitHub - modelcontextprotocol/ext-apps: Official repo for spec & SDK of MCP Apps protocol - standard for UIs embedded AI chatbots, served by MCP servers*. GitHub. Abgerufen 3. September 2026, von https://github.com/modelcontextprotocol/ext-apps

Reference – Plugins. (o. J.). OpenAI Developers. Abgerufen 3. September 2026, von https://developers.openai.com/plugins/reference

SEP-1865: MCP Apps - Interactive User Interfaces for MCP. (2025, November 21). Model Context Protocol. https://modelcontextprotocol.io/seps/1865-mcp-apps-interactive-user-interfaces-for-mcp

The Accelerating GenUI Ecosystem. (2026, März 27). https://www.telusdigital.com/insights/data-and-ai/article/accelerating-genui-ecosystem-mcp-apps-openai-apps-sdk-and-google-a2ui

What is the Model Context Protocol (MCP)? (2026, Juli 28). Model Context Protocol. https://modelcontextprotocol.io/docs/2026-07-28/getting-started/intro

11 Grober Zeitplan

Folgender Zeitplan besteht als Vorschlag auf Basis der eigenen Vorbereitungsplanung. Wichtige Meilensteine sind fett hervorgehoben.

18.09.2026	Erstellung und Bereitstellung Exposé an Betreuer
01.10.2026	Überarbeitung und Abnahme Exposé
05.10.2026	Abgabe der Anmeldung der BSc im Prüfungsamt
15.10.2026	**Beginn der Bearbeitungszeit (12 Wochen)**
28.10.2026	Grundlagenteil (Kapitel 2) bei 80%
11.11.2026	Kapitel 3 (Konzeption) bei 80%
18.11.2026	Kapitel 3 (Konzeption) bei 100%
25.11.2026	Beginn Implementierung
09.12.2026	**Implementierung zu 100%**
16.12.2026	Formulierung Kapitel 5 (Implementierung)
02.01.2027	Formulierung Kapitel 6 (Evaluation)
06.01.2027	**Kapitel Grundlagen, Konzeption, Implementierung und Evaluation auf 100% – Ende der Bearbeitungszeit**
09.01.2027	Korrekturlesen durch Dritte
11.01.2027	Einarbeitung Korrekturen
12.01.2027	Druckauftrag an Druckerei
14.01.2027	Empfang Druckversion von Druckerei
15.01.2027	**Abgabe Digital- und Printversion im Prüfungsamt**
Ende Jan. 2027	Anfertigung von Präsentationsunterlagen
Feb./März 2027	Phase Gutachtenerstellung der Betreuer
März 2027	Kolloquium BSc-Thesis