MCP-Server-Vergleich: Unser RRL/D0-Prototyp und der offizielle ALLPLAN-2027-MCP-Server

Der wesentliche Unterschied liegt in der Ausrichtung: Unser MCP-Server war als spezialisierte Schnittstelle für kontrollierte Modellzugriffe und später die RRL/PG-Entwicklung gedacht. Der offizielle ALLPLAN-MCP-Server ist breiter angelegt und soll KI-Agenten sowie Entwicklungswerkzeuge mit ALLPLAN verbinden.

Dabei muss man zwischen unserem eigenen MCP-Prototyp, den bereits erfolgreich getesteten PythonPart-Funktionen und dem offiziellen Hersteller-MCP unterscheiden. Nicht jede Funktion, die wir mit PythonParts nachgewiesen haben, war bereits über unseren MCP-Server verfügbar.

1. Technischer Vergleich

Kriterium	Unser RRL/D0-MCP	Offizieller ALLPLAN-MCP 2027
Entwickler	Eigenentwicklung	ALLPLAN
Hauptziel	Kontrollierte Modellzugriffe, später RRL/PG	Allgemeine KI- und Entwicklungsintegration
Technischer Status	Experimenteller Prototyp, zuletzt deaktiviert	Herstellerseitig bereitgestellt
MCP-Anbindung	Eigene lokale Implementierung	Offizielle Server-Integration
KI-Agenten	Vorgesehen, noch keine produktive Nutzung	Agent Integration vorgesehen
Entwicklungsumgebungen	Eigene Anbindung erforderlich	IDE Integration vorgesehen
Native Bewehrung lesen	Im PythonPart-Test nachgewiesen, nicht als vollständiger MCP-Dienst	Konkrete MCP-Tools nicht vollständig öffentlich belegt
Native Bewehrung ändern	Einzelne Python-API-Tests erfolgreich	Über MCP nicht ausreichend dokumentiert
Eigene Modellattribute	Nicht implementiert	Konkreter MCP-Zugriff ungeklärt
Dauerhafte PG-Beziehungen	Nicht implementiert	Keine PG-spezifische Herstellerfunktion nachgewiesen

Kriterium	Unser RRL/D0-MCP	Offizieller ALLPLAN-MCP 2027
Transaktionen / Undo	Im nativen Test teilweise untersucht	MCP-seitiger Vertrag ungeklärt
Sicherheitsprüfungen	Eigene Projekt-, TB- und UUID-Prüfungen entwickelt	Herstellerinterne Absicherung nicht ausreichend öffentlich beschrieben
Dokumentation	Eigene Versuchsprotokolle und Quellcode	Offizielle Produkt- und API-Dokumentation
Herstellerunterstützung	Keine für unseren Server	Grundsätzlich Herstellerprodukt; konkrete Supporttiefe offen
Produktive Eignung	Nicht erreicht	Für unseren RRL/PG-Anwendungsfall nicht nachgewiesen

ALLPLAN führt in seiner aktuellen Produktübersicht ausdrücklich AI Feature Builder (MCP) und AI Workflow Builder (MCP). Die veröffentlichte Python-API enthält außerdem `AllplanMCPServerService` mit Start- und Stop-Funktionen für die Eingabeansicht. Daraus lässt sich allerdings kein vollständiger Katalog der verfügbaren MCP-Modelloperationen ableiten. allplan.com +1

2. Vergleich der KI-Integration

Hier zeigt die aktuelle ALLPLAN-Dokumentation einen interessanten Unterschied: Der Hersteller konzentriert sich beim Feature Builder ausdrücklich auf die KI-gestützte Entwicklung von PythonParts.

Funktion	Unser MCP-Prototyp	ALLPLAN Feature Builder / MCP
Programmcode durch KI erzeugen	Über externe KI vorgesehen	Offizieller Bestandteil des Workflows
PythonParts-Dokumentation abrufen	Über eigene Recherche	Integrierter Dokumentationszugang
API-Aufrufe auf Gültigkeit prüfen	Eigene Prüfroutinen erforderlich	Prüfung gegen installierte ALLPLAN-Version vorgesehen
Planungsagent	Nicht eigenständig implementiert	PythonParts Planner
Programmieragent	Extern über ChatGPT Work	PythonParts Coder

Funktion	Unser MCP-Prototyp	ALLPLAN Feature Builder / MCP
Direktes Lesen bestehender Bewehrung	Ziel unseres RRL-Ansatzes	Für Feature Builder nicht als allgemeine Modellfunktion belegt
PG-Beziehungen verwalten	Unser spezialisiertes Entwicklungsziel	Nicht als Herstellerfunktion dokumentiert

Der Feature Builder unterstützt laut Hersteller unter anderem Claude Desktop und Visual Studio Code mit GitHub Copilot. Die erzeugten PythonParts werden gegen die lokal installierte ALLPLAN-Version geprüft. Allplan Hilfe

Das ist eine konkrete Verbesserung gegenüber unserer bisherigen Entwicklungsweise: Viele Fehler durch nicht vorhandene oder falsch verwendete API-Funktionen könnten früher erkannt werden.

Aber auch diese Prüfung erklärt nicht automatisch undokumentiertes Verhalten bei Objektidentitäten, Kopiervorgängen oder der Persistenz komplexer Beziehungen.

3. Welche Lösung wäre für unsere Aufgaben besser?

Aufgabe	Besser geeigneter Ansatz
Neue PythonParts per KI entwickeln	ALLPLAN Feature Builder
Existierende Modellobjekte automatisiert untersuchen	Eigener MCP, sofern passende Modellwerkzeuge implementiert sind
Eigene, streng begrenzte Prüfoperationen	Eigener MCP
Herstellerseitig geprüfte PythonPart-Codegenerierung	ALLPLAN
Native Bewehrung und PG-Beziehungen verändern	Noch keine Lösung ausreichend nachgewiesen
Komplexe Planprüfungen mit KI	Kombination aus Modellzugriff und spezialisierten Prüfwerkzeugen
Entwicklungsaufwand und Wartung reduzieren	Herstellerlösung, soweit sie benötigte Funktionen tatsächlich anbietet

4. Was bedeutet das für unser Projekt?

Es gibt einen grundsätzlichen Unterschied zwischen zwei MCP-Anwendungen:

MCP als Entwicklungsassistent: Die KI erzeugt oder überprüft Python-Code, den ALLPLAN später ausführt. Das ist die dokumentierte Stärke des Feature Builders.

MCP als Modellsteuerung: Die KI kann vorhandene Bauteile und Bewehrungen direkt abfragen, verändern und Arbeitsabläufe koordinieren. Das war ein wesentlicher Teil unserer eigenen Zielsetzung.

Für den zweiten Ansatz ist entscheidend, welche konkreten Modellwerkzeuge der offizielle Server tatsächlich anbietet. Eine öffentlich belastbare vollständige Werkzeugliste konnte ich nicht verifizieren.

Deshalb würde ich unseren MCP-Server nicht als technisch unterlegen einstufen. Er war stärker auf unsere speziellen Aufgaben ausgerichtet, aber noch nicht produktionsreif. Der Hersteller-MCP ist besser in die offizielle Entwicklungsumgebung eingebunden, seine Fähigkeiten zur direkten Modellbearbeitung sind jedoch noch nicht ausreichend transparent.

Mein Ergebnis: Der offizielle ALLPLAN-MCP könnte einen großen Teil unserer selbst entwickelten Infrastruktur überflüssig machen. Er beseitigt aber nicht zwangsläufig die Probleme, an denen wir beim RRL/PG-Projekt gescheitert sind. Dafür müsste er insbesondere persistente Beziehungen zwischen bestehenden nativen BarPlacements und deren Verhalten beim Kopieren nachweisbar unterstützen.

Quellen: ALLPLAN Feature Builder – Herstellerdokumentation · ALLPLAN MCP Server Service – API