

MCP(Model Context Protocol)

release date. 2026. 09. 11.

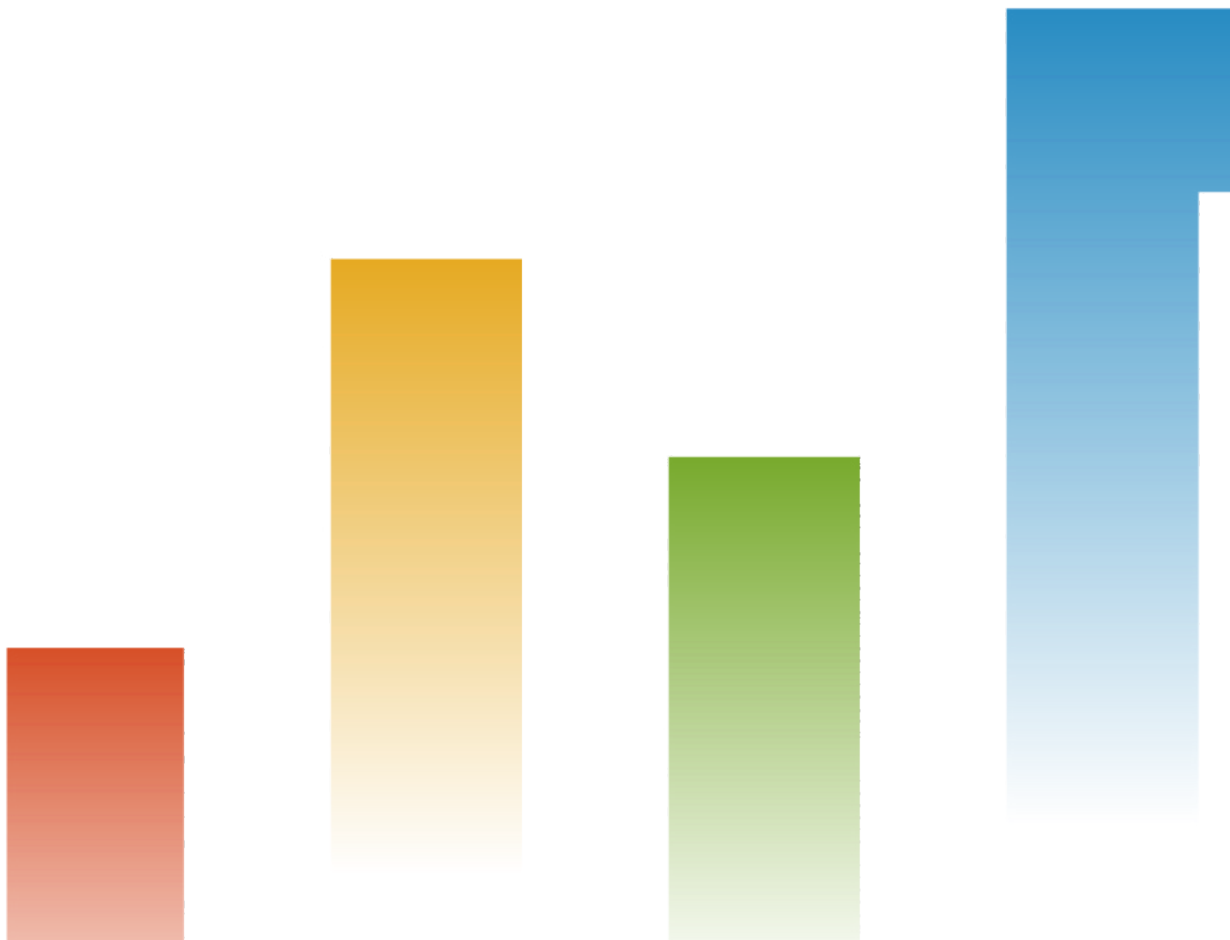


Table of Contents

• MCP	3
◦ はじめに	4
◦ 設定	8
▪ 環境設定	14
▪ アンインストール	17
◦ 使い方	19
▪ 活用例	23
◦ トラブルシューティング	28

MCP(Model Context Protocol)

WhaTap MCPは、AIアシスタントが外部ツールを呼び出す標準プロトコルであるMCP(Model Context Protocol)を通じて、WhaTap APIと連携するサーバーです。ユーザーは自然言語1行で、サーバーCPU、APM TPS、K8s Podステータスなどのモニタリングデータを即座に照会できます。

- ・ ツール数: 10個（プロジェクト管理3、データ探索3、複合分析2、インストール・PromQL2）
- ・ 対応ドメイン: Server、APM、Kubernetes、Database、Log
- ・ 現在のバージョン: v1.2.1
- ・ オープンソース: github.com/whatap/whatap-open-mcp

❗ インストール済みのバージョンは `npx -y github:whatap/whatap-open-mcp --version` で確認できます。

主な特長

- ・ 複雑なダッシュボードを操作せず、自然言語での質問だけでモニタリングデータを照会できます。
- ・ サーバー、APM、Kubernetesなど、さまざまなWhaTapモニタリング製品のデータを統合して照会できます。
- ・ 既存のAIワークフローにモニタリングのコンテキストを自然に組み込みます。

対応AIクライアント

クライアント	提供元	設定方式
Claude Desktop	Anthropic	JSON設定ファイル
Claude Code	Anthropic	CLIコマンド1行
Codex CLI	OpenAI	CLIコマンドまたはTOML設定
Gemini CLI	Google	CLIコマンドまたはJSON設定

はじめに

AIアシスタント（Claude、Codex、Geminiなど）からWhaTapモニタリングデータを自然言語で照会できるMCPサーバーです。以下の手順でWhaTap MCPを開始してください。

ステップ1. 事前準備

ステップ2. 会員登録

ステップ3. APIトークン発行

ステップ4. AIクライアントの設定

事前準備

WhaTap MCPサーバーを使用する前に、以下の項目を準備してください。

Item	Details
Node.js	18.0.0以上（ <code>npm</code> 、 <code>npx</code> を含む）
Git	Windows環境でのみ必須
WhaTapアカウント	whatap.io の無料または有料アカウント
AIクライアント	Claude Desktop、Claude Code、Codex CLI、Gemini CLIのいずれか

Node.jsのインストール確認

ターミナルで以下のコマンドを実行してください。バージョンが低い場合やインストールされていない場合は、インストール方法を参照してください。

```
node --version # v18.x.x以上であればOK
npx --version # バージョンが表示されればOK
```

Node.jsのインストール

使用しているOSに合わせた方法でNode.jsをインストールしてください。

macOS

```
brew install node
```

❗ Homebrewがない場合は<https://brew.sh>でインストールするか、<https://nodejs.org>から直接ダウンロードしてください。

Windows

```
winget install OpenJS.NodeJS.LTS
```

❗ または<https://nodejs.org>からLTSバージョンのインストーラーをダウンロードしてください。インストール後は新しいターミナルを開く必要があります。

Ubuntu / Debian

```
curl -fsSL https://deb.nodesource.com/setup_22.x | sudo -E bash -  
sudo apt install -y nodejs
```

❗ `apt install nodejs` でインストールすると古いバージョン（v12～v16）がインストールされます。必ずNodeSourceを使用してください。

Fedora / RHEL / Amazon Linux

```
sudo dnf install -y nodejs npm
```

nvmでインストール（全OS対応）

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.3/install.sh | bash
source ~/.bashrc
nvm install --lts
```

Gitのインストール

WhaTap MCPサーバーはGitHubリポジトリを直接参照する際にGitを使用します。macOSとLinuxにはGitが標準で含まれているため、Windows環境でのみインストールしてください。

```
winget install --id Git.Git -e --source winget
```

❗ Windows Package Manager（winget）を使用しない場合は、<https://git-scm.com>からインストーラーをダウンロードしてください。インストール後は新しいターミナルを開く必要があります。

インストールが完了したら、ターミナルで以下のコマンドを実行してください。

```
git --version #バージョンが表示されればOK
```

会員登録

WhaTapモニタリングサービスを使用するには、まず[会員登録](#)が必要です。詳細は[アカウント管理 > 会員登録](#)ドキュメントを参照してください。

APIトークン発行

WhaTap MCPサーバーがデータを照会する際に使用するAPIトークンを発行します。

1. [WhaTap Console](#)にログインします。
2. 右上のプロフィールアイコンをクリックします。

3. アカウント管理を選択します。
4. APIトークンの項目からトークンをコピーします。

ⓘ このトークンはアカウントレベルのトークンです。このアカウントに属するすべてのプロジェクトのデータを照会できます。データの変更や削除はできないため、安心して使用できます。ただし、トークンが外部に漏洩しないよう適切に管理してください。

設定

使用するAIクライアントに合わせた設定方法です。どのクライアントを選んでも同じMCPサーバーを使用し、設定形式のみ異なります。

!

すべてのクライアントでGitHubリポジトリを直接参照します: `npx -y github:whatap/whatap-open-mcp`

リポジトリの参照にGitを使用するため、Windows環境では事前にインストールしてください。詳細は[はじめに](#) > [Gitのインストール](#)ドキュメントを参照してください。

AI Client	Format	Config file	Command
Claude Desktop	JSON	<code>claude_desktop_config.json</code>	ファイルを直接編集
Claude Code	-	-	<code>claude mcp add whatap -e WHATAP_API_TOKEN=... -- npx -y github:whatap/whatap-open-mcp</code>
Codex CLI	TOML	<code>~/.codex/config.toml</code>	<code>codex mcp add whatap --env ... -- npx -y github:whatap/whatap-open-mcp</code>
Gemini CLI	JSON	<code>~/.gemini/settings.json</code>	<code>gemini mcp -s user add whatap -e ... -- npx -y github:whatap/whatap-open-mcp</code>

Claude Desktop

設定ファイルにWhaTap MCPサーバーの情報を追加します。

OS	Path
macOS	<code>~/Library/Application Support/Claude/claude_desktop_config.json</code>
Linux	<code>~/.config/Claude/claude_desktop_config.json</code>

OS	Path
Windows	%APPDATA%\Claude\claude_desktop_config.json

設定を追加する

設定ファイルを開き、以下の内容を追加します。

Claude Desktop 設定追加

```
{
  "mcpServers": {
    "whatap": {
      "command": "npx",
      "args": ["-y", "github:whatap/whatap-open-mcp"],
      "env": {
        "WHATAP_API_TOKEN": "ここにトークンを入力"
      }
    }
  }
}
```

ⓘ 他のMCPサーバーがすでに設定されている場合は、既存の "mcpServers" の中に "whatap": { ... } ブロックのみを追加してください。

登録を確認する

設定ファイルを保存してClaude Desktopを再起動すると、WhaTap MCPサーバーが接続されます。

⚠ Claude Desktopはウィンドウを閉じるのではなく、アプリを終了してから再起動しないと設定が反映されません。

Claude Code (CLI)

ターミナルコマンドでWhaTap MCPサーバーを登録します。

適用範囲を選択する

登録前に使用範囲を決定します。

Scope	Option	Description
全プロジェクト	<code>-s user</code>	アカウントのすべてのプロジェクトで使用
現在のプロジェクト	<code>-s project</code>	現在のプロジェクトのみ（ <code>.mcp.json</code> に保存）

コマンドで登録する

Claude Code (CLI) 登録

```
# 全プロジェクト (userスコープ)
claude mcp add whatap -s user \
-e WHATAP_API_TOKEN=ここにトークンを入力 \
-- npx -y github:whatap/whatap-open-mcp

# 現在のプロジェクト (projectスコープ)
claude mcp add whatap -s project \
-e WHATAP_API_TOKEN=ここにトークンを入力 \
-- npx -y github:whatap/whatap-open-mcp
```

登録を確認する

Claude Code (CLI) 登録確認

```
claude mcp list
```

Codex CLI (OpenAI)

コマンドまたは設定ファイルでWhaTap MCPサーバーを登録します。

登録する

以下の2つの方法から選択できます。

- [コマンドで登録する](#) 設定ファイルを直接編集して登録する

Codex CLI (OpenAI) コマンド登録

```
codex mcp add whatap \  
--env WHATAP_API_TOKEN=ここにトークンを入力 \  
-- npx -y github:whatap/whatap-open-mcp
```

~/codex/config.toml

```
[mcp_servers.whatap]  
command = "npx"  
args = ["-y", "github:whatap/whatap-open-mcp"]
```

```
[mcp_servers.whatap.env]  
WHATAP_API_TOKEN = "ここにトークンを入力"
```

❗ CodexはJSONではなくTOML形式を使用します。

登録を確認する

Codex CLI (OpenAI) 登録確認

```
codex mcp list
```

Gemini CLI (Google)

コマンドまたは設定ファイルでWhaTap MCPサーバーを登録します。

適用範囲を選択する

登録前に使用範囲を決定します。

Scope	File location	Purpose
ユーザー（全体）	<code>~/.gemini/settings.json</code>	個人設定、全プロジェクト
プロジェクト	<code>.gemini/settings.json</code> （プロジェクトルート）	チーム共有用

登録する

以下の2つの方法から選択できます。

- コマンドで登録する 設定ファイルを直接編集して登録する

Gemini CLI (Google) 登録

```
gemini mcp -s user add whatap \  
-e WHATAP_API_TOKEN=ここにトークンを入力 \  
npx -- -y github:whatap/whatap-open-mcp
```

~/.gemini/settings.json

```
{  
  "mcpServers": {  
    "whatap": {  
      "command": "npx",  
      "args": ["-y", "github:whatap/whatap-open-mcp"],  
      "env": {  
        "WHATAP_API_TOKEN": "ここにトークンを入力"  
      }  
    }  
  }  
}
```

```
}  
}  
}
```

登録を確認する

Gemini CLI (Google) 登録確認

```
gemini mcp list
```

接続を確認する

1. 設定完了後、AIクライアントを開きます。
2. 以下のように入力します。

```
WhaTapのプロジェクト一覧を見せて
```

3. プロジェクトコード（pcode）、名前、プラットフォーム、製品タイプを含む一覧が表示されれば接続は正常です。表示されない場合はトラブルシューティングドキュメントを参照してください。

環境設定

WhaTap MCPサーバーは WHATAP_API_TOKEN 1つだけ設定すれば、SaaS環境ですぐに使用できます。On-premises環境では、WHATAP_API_URL を追加で設定する必要があります。

環境変数

変数	必須	デフォルト値	説明
WHATAP_API_TOKEN	O	-	WhaTapアカウントAPIトークン
WHATAP_API_URL	X	https://api.whatap.io	APIエンドポイント（SaaS環境では変更不要）

APIエンドポイント

WhaTap SaaS環境では、すべてのリージョン（ソウル、東京、シンガポール、フランクフルトなど）が同じAPI URL（https://api.whatap.io）を使用します。リージョン別のルーティングはサーバー側で自動的に処理されるため、別途設定は不要です。

以下をご参照の上、WHATAP_API_URL を変更してください。

環境	URL	備考
SaaS（デフォルト）	https://api.whatap.io	変更不要
On-premises	お客様の内部URL	WHATAP_API_URL の設定が必要

On-premises設定例（MCPサーバーのIPが192.168.1.100の場合）

Claude Desktop

Claude Desktop

```
{
  "mcpServers": {
    "whatap": {
      "command": "npx",
      "args": ["-y", "github:whatap/whatap-open-mcp"],
      "env": {
        "WHATAP_API_TOKEN": "ここにトークンを入力",
        "WHATAP_API_URL": "http://192.168.1.100"
      }
    }
  }
}
```

Claude Code

Claude Code

```
claude mcp add whatap \
-e WHATAP_API_TOKEN=ここにトークンを入力 \
-e WHATAP_API_URL=http://192.168.1.100 \
-- npx -y github:whatap/whatap-open-mcp
```

Codex CLI

Codex CLI

```
[mcp_servers.whatap]
command = "npx"
args = ["-y", "github:whatap/whatap-open-mcp"]

[mcp_servers.whatap.env]
WHATAP_API_TOKEN = "ここにトークンを入力"
```

```
WHATAP_API_URL = "http://192.168.1.100"
```

Gemini CLI

Gemini CLI

```
gemini mcp -s user add whatap \
-e WHATAP_API_TOKEN=ここにトークンを入力 \
-e WHATAP_API_URL=http://192.168.1.100 \
npx -- -y github:whatap/whatap-open-mcp
```

WhaTapサービスリージョン

リージョンはWhaTap Consoleでプロジェクト作成時に選択します。リージョンが異なってもAPI URLは同じです。

リージョン	場所	クラウド
ソウル	韓国	AWS、Azure、Kakaoクラウド
東京	日本	AWS、Azure
シンガポール	シンガポール	AWS
ジャカルタ	インドネシア	AWS
ムンバイ	インド	AWS
カリフォルニア	米国西部	AWS
バージニア	米国東部	AWS
フランクフルト	ドイツ	AWS

アンインストール

WhaTap MCPサーバーを使用しなくなった場合や設定を初期化したい場合は、AIクライアントからMCPサーバーを削除してください。

削除方法

Claude、Codex、GeminiなどのAIクライアントからWhaTap MCPサーバーを削除する方法です。

Claude Desktop

1. `claude_desktop_config.json` から `"whatap"` ブロックを削除します。
2. Claude Desktopを再起動します。

Claude Code

```
claude mcp remove whatap
```

Codex CLI

`~/.codex/config.toml` から `[mcp_servers.whatap]` セクションを削除します。

Gemini CLI

```
gemini mcp remove whatap
```

または `~/.gemini/settings.json` から `"whatap"` ブロックを削除します。

キャッシュの削除（任意）

npxキャッシュを削除すると、次回実行時に最新バージョンが取得されます。

```
npx clear-npx-cache
```

使い方

WhaTap MCPサーバーに接続した後、AIアシスタントでモニタリングデータを照会する方法を説明します。

動作の仕組み

AIアシスタントに自然言語で質問すると、以下の処理が自動的に行われます。

- プロジェクト一覧の取得 — プロジェクトコード（pcode）を確認します。
- データカタログの探索 — 照会できるデータの種類を確認します。
- クエリの実行 — 必要なデータを照会します。

✔ 「プロジェクト一覧を見せて」から始めて、まずプロジェクトコードを確認してください。そのコードを使用すると、より正確な結果が得られます。

提供ツール一覧

MCPサーバーはAIアシスタントに合計10個のツールを提供します。ユーザーが直接ツールを呼び出す必要はなく、自然言語で質問するとAIが適切なツールを自動的に選択します。

プロジェクト管理ツール

ツール	機能
<code>whatap_list_projects</code>	全モニタリングプロジェクト一覧の照会（pcode、名前、プラットフォーム）
<code>whatap_project_info</code>	特定プロジェクトの詳細情報の照会
<code>whatap_list_agents</code>	プロジェクト内のエージェント（サーバー/インスタンス）一覧の照会

データ探索ツール

ツール	機能
<code>whatap_data_availability</code>	900件以上のMXQLクエリを20+カテゴリで探索、リアルタイムデータ検知、OpenMetricsメトリクス一覧、保存済みPromQLクエリ表示
<code>whatap_describe_query</code>	MXQLパスまたはOpenMetricsメトリクスの詳細照会（パラメーター、出力フィールド、メトリクスタイプ、推奨PromQL）
<code>whatap_query_data</code>	MXQL、PromQL、保存済みクエリの実行（単位変換、しきい値、分析ガイド含む）

複合分析ツール

ツール	機能
<code>whatap_apm_anomaly</code>	APM異常検知 — TPS、レスポンスタイム、エラー、Active TXの4つのクエリを同時実行して統計分析
<code>whatap_service_topology</code>	サービス間の接続関係とネットワークボトルネックの検知（NPMエージェントが必要）

インストール・PromQLツール

ツール	機能
<code>whatap_install_agent</code>	29プラットフォームのエージェントインストールコマンドを生成（接続情報自動付与、プラットフォーム自動検知）
<code>whatap_create_promql</code>	PromQLクエリの作成、リアルタイム検証、保存（再利用可能）

whatap_install_agentの対応プラットフォーム

カテゴリ	プラットフォーム
インフラOS	Debian/Ubuntu、Amazon Linux、RHEL/CentOS、SUSE、FreeBSD、Windowsなど9種
APM	Java、Node.js、Python、PHP、.NET、Go
データベース	PostgreSQL、Oracle、MySQL、Redis、MongoDB、DB2など12種
コンテナ	Kubernetes (Helm)
サーバーアプリ	Kafka、NGINX、Apacheなど7種

ツールの使用フロー

データ照会ワークフロー（MXQL）

```

whatap_list_projects → projectCodeの確認
↓
whatap_data_availability(projectCode) → 使用可能なMXQLパスの確認
↓
whatap_describe_query(path) → パラメーター、フィールド、MXQL原文の確認
↓
whatap_query_data(projectCode, path) → データ照会

```

PromQLワークフロー（Kubernetes/OpenMetrics）

```

whatap_data_availability(projectCode) → OpenMetricsメトリクス一覧の確認
↓
whatap_create_promql(projectCode, name, query) → PromQLの検証・保存
↓
whatap_query_data(projectCode, savedQuery="name") → 保存済みクエリの再利用

```

エージェントインストールワークフロー

```

whatap_list_projects → projectCode + プラットフォームの確認

```

↓
 whatap_install_agent(projectCode) → 接続情報 + インストールコマンドの出力
 ↓
 whatap_list_agents(projectCode) → エージェント登録の確認

🟢 このフローはAIが自動的に進めます。ユーザーは照会したい内容を自然言語で質問するだけです。

主要MXQLパスリファレンス

whatap_data_availability を呼び出さずにすぐ使用できる主要パスの一覧です。

ドメイン	パス	説明
Server	v2/sys/server_base	CPU、メモリ、基本指標
Server	v2/sys/server_disk	ディスク使用量
Server	v2/sys/server_network	ネットワークI/O
APM	v2/app/tps_pcode	TPS（プロジェクト全体）
APM	v2/app/resp_time_pcode	レスポンスタイム（プロジェクト全体）
APM	v2/app/tx_error_pcode	エラー数（プロジェクト全体）
K8s	v2/container/kube_pod	Podステータス
K8s	v2/container/kube_node	ノードステータス
K8s	v2/container/kube_event	クラスターイベント
DB	v2/db/instance_active_session	アクティブセッション

活用例

自然言語での質問例と実際の会話フローを通じて、WhaTap MCPサーバーの活用方法を確認してください。

よく使う質問

状況	質問例
はじめて使うとき	「プロジェクト一覧を見せて」
サーバー状態確認	「直近5分のサーバーCPU・メモリ状態を要約して」
APM異常確認	「異常のあるエージェントを見つけて」
障害対応	「直近5分のエラー状況を要約して」
データ探索	「このプロジェクトで照会できるデータは何がある？」
K8s点検	「Podの状態を確認して」
サービス関係	「サービストポロジを見せて」
OpenMetrics探索	「プロジェクト33194で利用できるOpenMetricsメトリクスを見せて」
PromQL生成	「Pod別CPU使用率のPromQLクエリを作って」
保存済みクエリ実行	「保存済みの'CPU by Pod'クエリを実行して」
エージェントインストール	「プロジェクト12345にモニタリングエージェントをインストールする方法を教えて」

自然言語での質問例

モニタリング作業に使用する自然言語の質問例です。用途に合った質問を選択または参考にして作成してください。

プロジェクトの探索

WhaTapのプロジェクト一覧を見せて
プロジェクト12345の詳細情報を教えて
プロジェクト12345にはどんなエージェントがある？

サーバーモニタリング

直近1時間のサーバーCPU使用率を見せて
メモリ使用量が80%を超えているサーバーはある？
ディスク使用量を確認して
CPU使用率の高い順にサーバーTop 5を見せて

APM（アプリケーションパフォーマンス）

直近5分のTPS推移を見せて
平均レスポンスタイムはどのくらい？
トランザクションエラーは発生している？
現在進行中のトランザクション状態を確認して

Kubernetes

Podの状態を確認して
ノード別のCPU・メモリ使用量を見せて
最近のクラスターイベントでWarningはある？

データベース

DBインスタンス一覧を見せて
現在のアクティブセッションを確認して

異常検知

直近5分で異常のあるエージェントを見つけて
高感度で1時間の異常検知を実行して

サービストポロジ

サービス間の接続関係を見せて
ネットワークのボトルネックはある？

OpenMetrics / PromQL

プロジェクト33194で利用できるOpenMetricsメトリクスを見せて
Pod別CPU使用率のPromQLクエリを作って
保存済みの'CPU by Pod'クエリを実行して

エージェントインストール

プロジェクト12345にモニタリングエージェントをインストールする方法を教えて
Java APMエージェントをインストールして
Kubernetesエージェントの設定方法を見せて
PostgreSQL DBモニタリングをインストールして
DebianサーバーにInfraエージェントをインストールして

複合質問

朝のヘルスチェックをして。
全プロジェクトのサーバー状態を確認して、
APMプロジェクトに異常がないか分析して。

会話例

以下の会話は例であり、実際の応答はプロジェクトのモニタリングデータによって異なります。

例1. 朝のサーバー点検

ユーザー: 朝の点検をして。サーバーのCPUとメモリの状態を要約して。

AI: 3台のサーバーを確認しました。

- web-01: CPU 45%, Memory 62% (正常)
- web-02: CPU 87%, Memory 71% (CPU注意 — 推奨70%超過)
- db-01: CPU 23%, Memory 55% (正常)

web-02サーバーのCPUが87%と高い状態です。プロセスを確認しますか？

例2. APM異常検知

ユーザー: Java APMプロジェクトに異常がないか確認して。

AI: 5つのエージェントを分析しました。

異常検知 (1件) :

- api-server-03: レスポンスタイム急増 (平均120ms → 最大2.3秒、+4.2σ)

残りの4エージェントは正常範囲内です。

api-server-03のトランザクション詳細を確認しますか？

例3. 障害対応

ユーザー: さっきエラーアラートが来た。直近5分のエラー状況を要約して。

AI: 直近5分のトランザクションエラー状況です。

| エージェント | TPS | エラー数 | エラー率 |

---	--- ---	---	
api-01	45.2 23	10.2%	
api-02	42.8 2	0.9%	
api-03	44.1 0	0%	

api-01のエラー率が10.2%と異常な水準です。

レスポンスタイムの推移とActive TXも合わせて確認しますか？

例4. 経営層向けレポートデータ

ユーザー: 今週の全サービスのレスポンスタイムとエラー率の推移を要約して。

AI: 直近7日間のサービス状況です。

- 平均レスポンスタイム: 142ms (前週比 -8%)
- 平均エラー率: 0.3% (前週比 変動なし)
- TPS平均: 1,240件/秒 (前週比 +12%)

全体的に良好な状態です。TPS増加にもかかわらず

レスポンスタイムが改善されており、パフォーマンスが向上していると判断されます。

トラブルシューティング

MCPサーバーへの接続や使用中に問題が発生した場合は、以下の項目を確認してください。

"WHATAP_API_TOKEN environment variable is required"

Q APIトークンが設定されていないか、MCPサーバーに渡されていません。

A 各クライアントの設定ファイルで以下の項目を確認してください。

- Claude Desktop: "env" ブロックに "WHATAP_API_TOKEN" キーがあるか確認（"WHATAP_TOKEN" ではありません）
- Codex CLI: TOMLファイルに [mcp_servers.whatap.env] セクションがあるか確認
- Gemini CLI: settings.jsonに "env" ブロックがあるか確認
- Claude Code: `claude mcp add` コマンドに `-env` フラグが含まれているか確認

"No API token found for project XXXXX"

Q アカウントトークンは有効ですが、該当プロジェクトコードが存在しないかアクセス権限がありません。

A AIアシスタントに「プロジェクト一覧を見せて」と入力し、正確なプロジェクトコードを確認します。

"npx: command not found"

Q Node.jsがインストールされていないか、PATHに登録されていません。

A Node.js 18以上をインストールしてください。

```
node --version # v18以上を確認  
npx --version # バージョンが表示されるか確認
```

"spawn git ENOENT"または"An unknown git error occurred"

Q Gitがインストールされていないか、実行パスがPATHに登録されていません。

A WhaTap MCPサーバーはGitHubリポジトリを直接参照するため、Gitが必要です。主にWindows環境で発生します。

```
winget install --id Git.Git -e --source winget
```

インストール後は新しいターミナルを開き、以下のコマンドを実行してください。

```
git --version # バージョンが表示されるか確認
```

詳細は[はじめに > Gitのインストール](#)ドキュメントを参照してください。

MCPサーバーが起動しないか、タイムアウトが発生する

Q MCPサーバーが起動しないか、タイムアウトが発生します。

A サーバーを直接実行してエラーメッセージを確認してください。

```
WHATAP_API_TOKEN=ここにトークンを入力 npx -y github:whatap/whatap-open-mcp
```

サーバーが正常に起動すると入力待ち状態になります。 **Ctrl+C** で終了してください。エラーメッセージが表示された場合は、トークンの値またはNode.jsのバージョンを確認してください。

Claude Desktopでツールが表示されない

Q Claude DesktopでMCPツールが表示されません。

A 以下の項目を順番に確認してください。

1. JSONファイルが有効か確認します（カンマの欠落、括弧の不一致など）。
2. Claude Desktopを完全に終了してから再起動します（ウィンドウを閉じるのではなく、アプリを終了）。
3. ログを確認します: `~/Library/Logs/Claude/mcp*.log`（macOS）

Codex CLIでツールが表示されない

Q Codex CLIでMCPツールが表示されません。

A 以下の項目を順番に確認してください。

1. `codex mcp list` で登録状況を確認します。
2. `~/.codex/config.toml` が正しいTOML形式であるか確認します（JSON文法は使用不可）。
3. 削除後に再登録します: `codex mcp add whatap ...`

Gemini CLIでツールが表示されない

Q Gemini CLIでMCPツールが表示されません。

A 以下の項目を順番に確認してください。

1. `gemini mcp list` で登録状況を確認します。
2. `~/.gemini/settings.json` が正しいJSON形式であるか確認します。
3. 適用範囲を確認します: `-scope user`（全体）または `-scope project`（プロジェクト）

"WhaTap API error (401)" または "(403)"

Q APIトークンが無効か期限切れです。

A WhaTap Consoleで**アカウント管理** > **APIトークン**に移動し、新しいトークンを発行します。

レスポンスが遅い

Q レスポンスが遅いです。

A 照会時間の範囲を短くしてください。`"5m"`（5分）なら数秒、`"1d"`（1日）なら数十秒かかる場合があります。連続リクエスト時にWhaTap APIのレート制限に引っかかる場合があります。しばらく待ってから再試行してください。

データが照会されない

Q データが照会されません。

A 以下の原因と確認方法を参照してください。

原因	確認方法
プロジェクトタイプの不一致（例：ServerプロジェクトでAPMクエリを実行）	<code>whatap_data_availability(projectCode)</code> でアクティブカテゴリを確認 → 「このプロジェクトで照会できるデータを見せて」
エージェント未インストールまたは非アクティブ	<code>whatap_list_agents(projectCode)</code> でエージェントの状態を確認 → 「エージェント一覧を見せて」
時間範囲が狭すぎる	<code>timeRange="1h"</code> など範囲を広げて再試行