

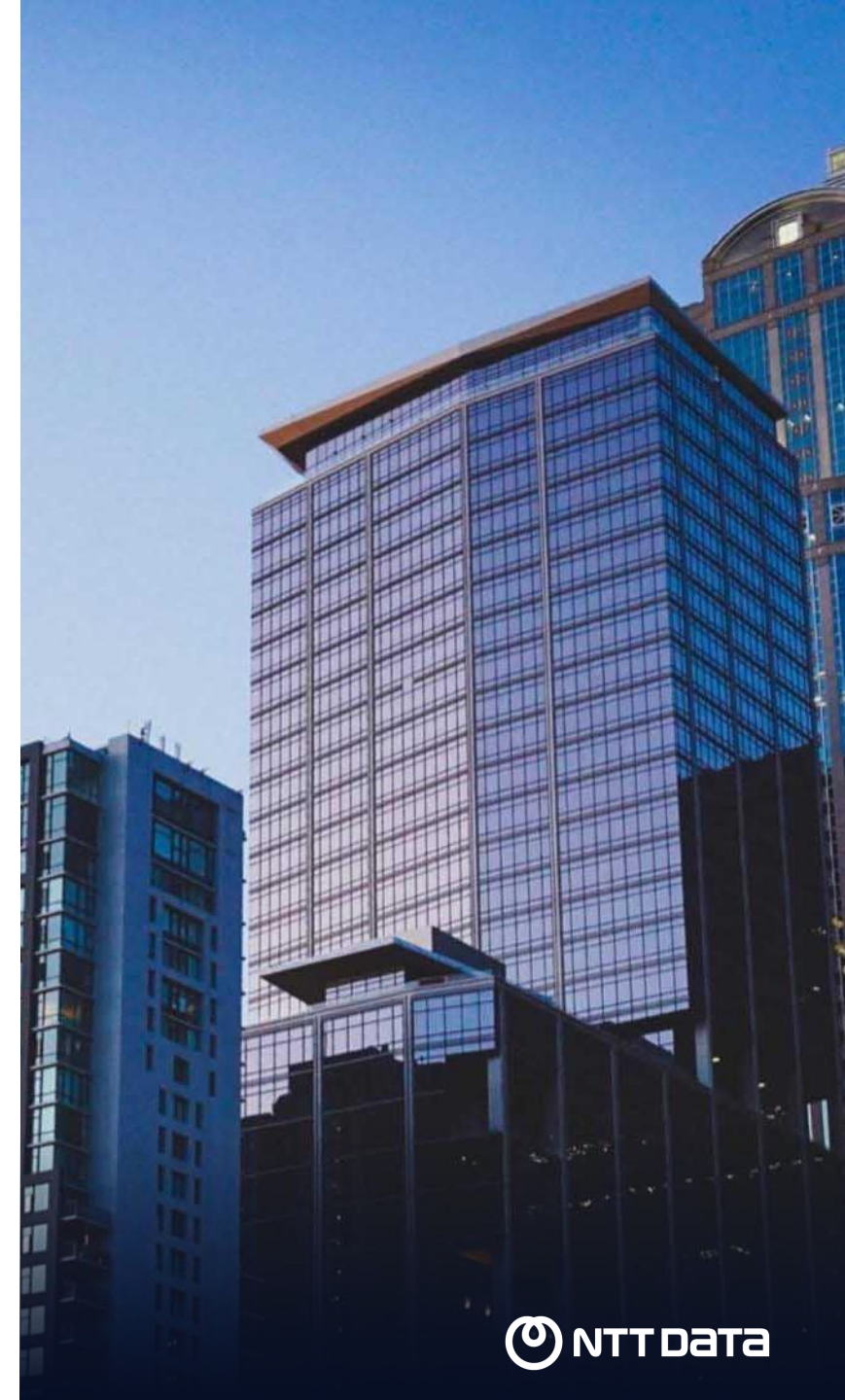
Hinemos AIエージェントのご紹介



Hinemos

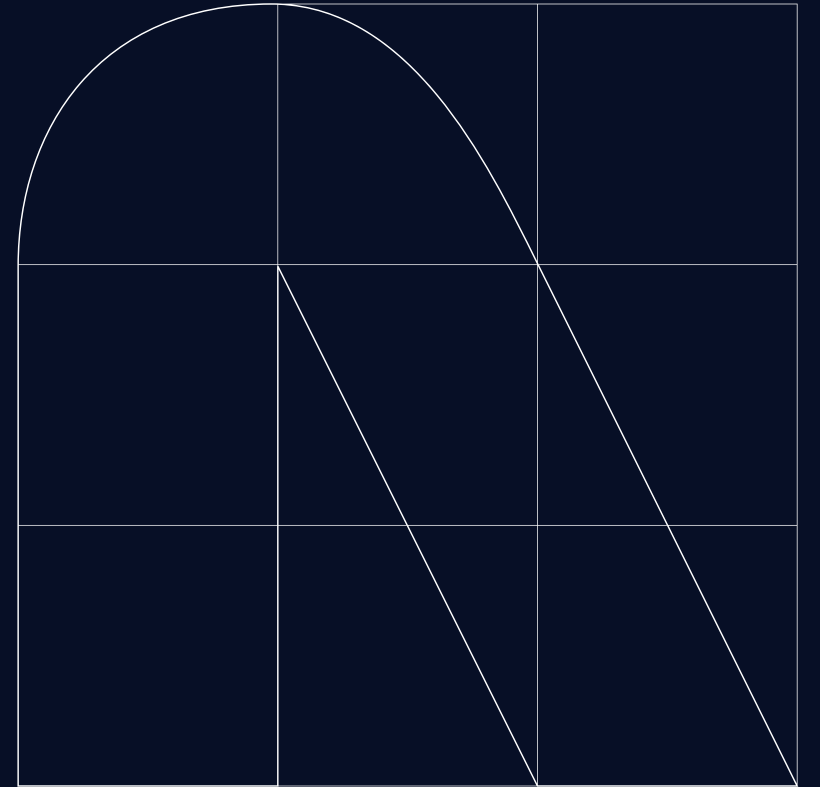
INDEX

1. Hinemos AIエージェントの概要
2. Hinemos AIエージェントのユースケースイメージ
3. 動作要件



01

Hinemos AIエージェントの概要



Hinemos AIエージェントとは

システム運用の確認から分析、調査、対処までを自動化をサポート

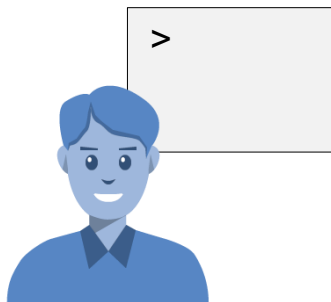
大規模言語モデル（LLM）
の活用

インターネット等の
一般的なナレッジの活用

Hinemosの蓄積した
運用イベントの活用

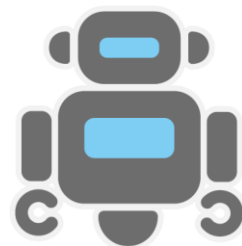
自動化エンジンとしての
Hinemosの活用

自然言語による問い合わせ



オペレータ・SE

AIとHinemosを連携



Hinemos AIエージェント

運用タスクの自動化をサポート・推進

確認

分析

調査

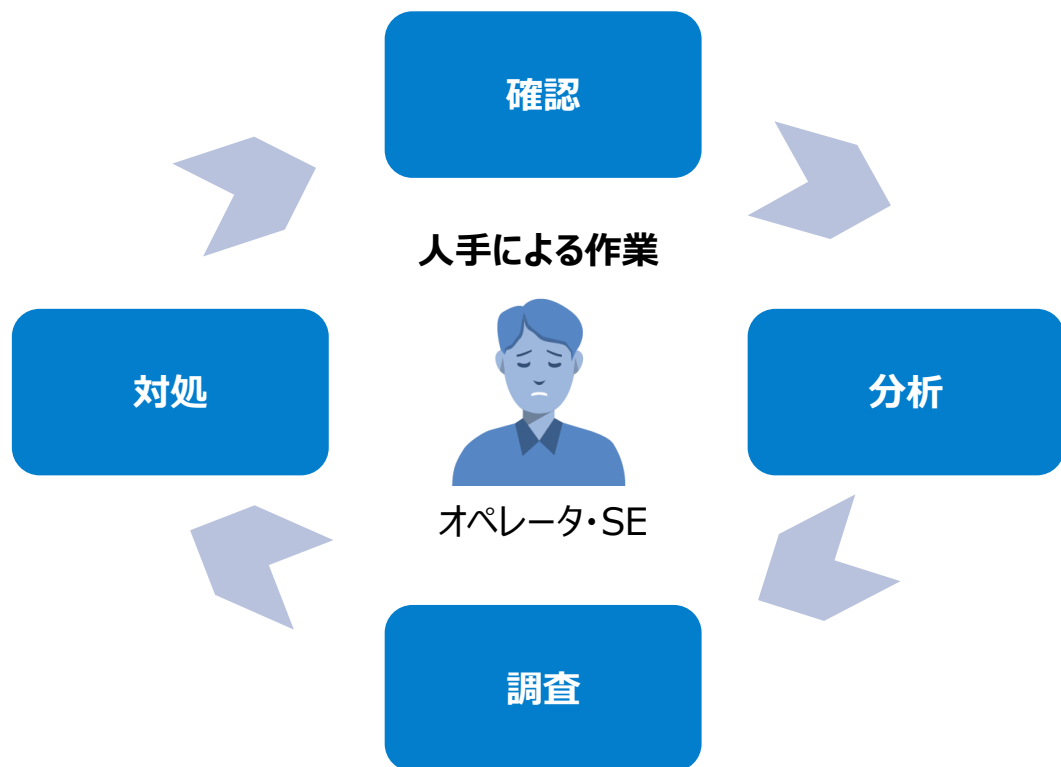
対処

自動化

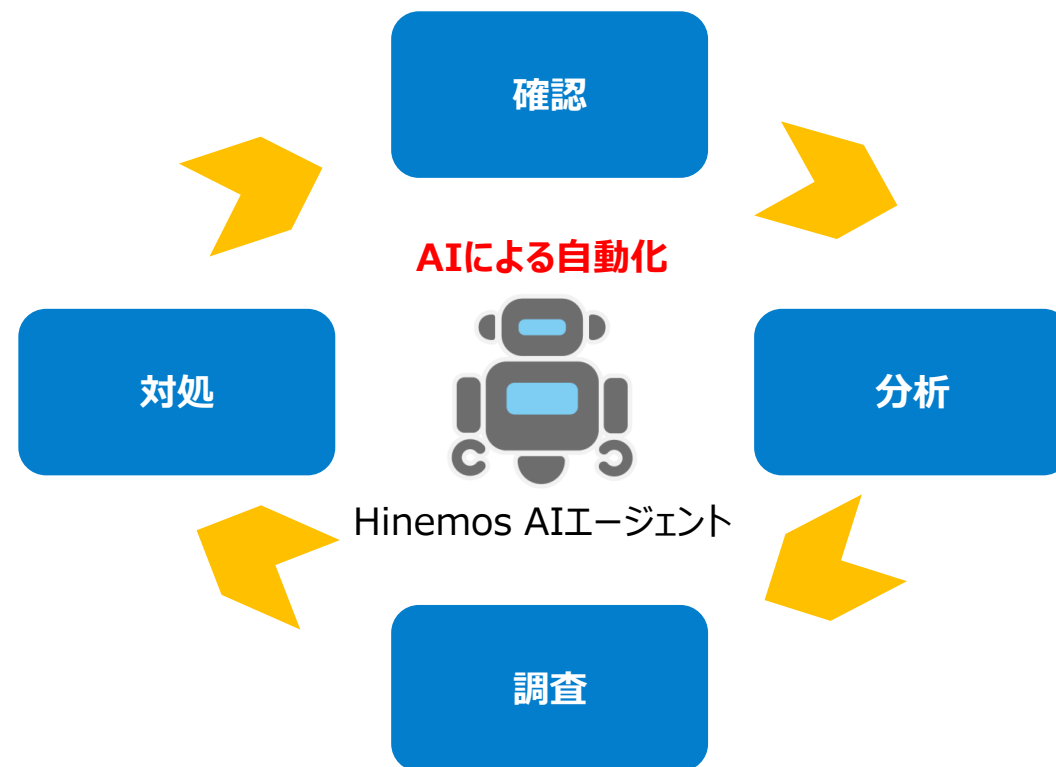
Hinemos AIエージェントで何ができる？

通常運用から障害イベント発生時の対処を、様々なナレッジを活用してより安全に実現

これまでの運用



Hinemosが提供する運用

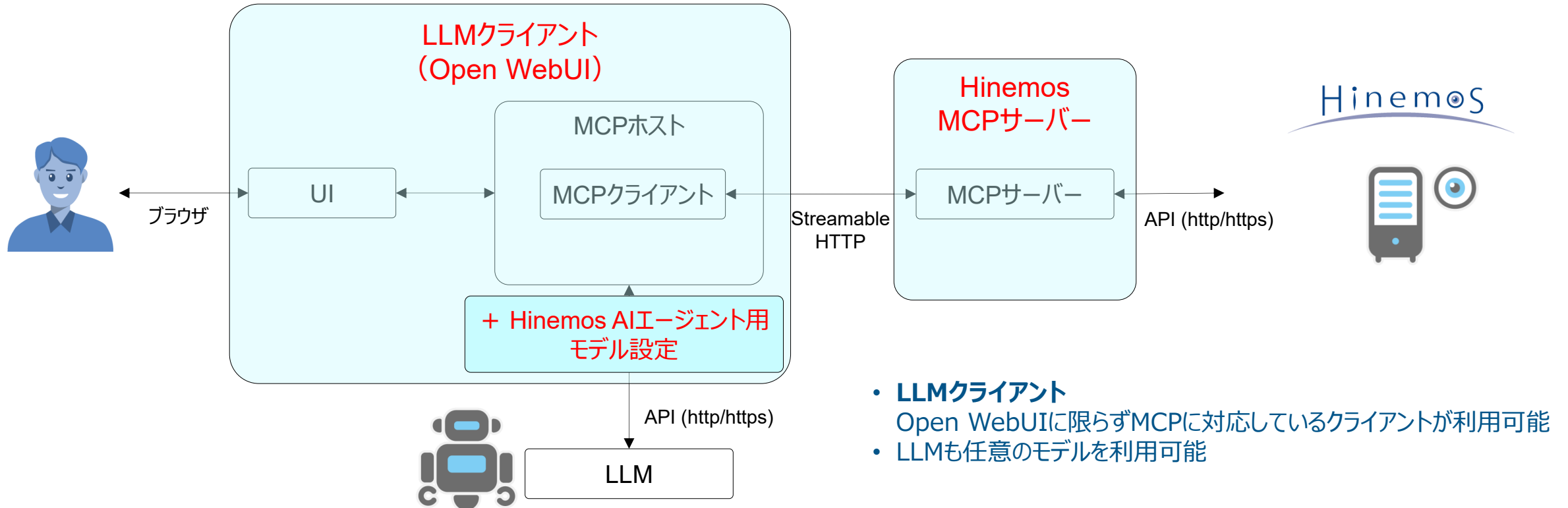


アーキテクチャ

標準プロトコルであるMCPを利用して生成AIとHinemosの連携を実現

提供物

- LLMクライアント（サンプルとしてOpen WebUIの導入手順、Hinemos AIエージェント用モデル定義を提供）
- Hinemos MCPサーバー（製品提供）



- **LLMクライアント**
Open WebUIに限らずMCPに対応しているクライアントが利用可能
- LLMも任意のモデルを利用可能

提供物詳細

Hinemos AIエージェントの提供物は以下の通り

パッケージ名	提供種別	説明
Hinemos MCPサーバ	製品	HinemosとLLMの連携を実現するプログラム。 任意のLLMクライアント（MCP対応）と連携可能。
LLMクライアント（サンプル）	サンプル	LLMクライアントのサンプルとして、Open WebUIの導入手順、Open WebUI向けのHinemos AIエージェントモデル定義を提供。 本サンプルを利用することで、環境にLLMクライアントが無くても、Hinemos AIエージェントを手軽に開始可能。

注1) 連携先のLLMはお客様にて用意いただく必要があります。

サポート範囲

Hinemos AIエージェントのサポート範囲は以下の通り

対象	サポート対象	説明
Hinemos MCPサーバ	○	「想定ユースケース」、「取得可能なHinemos情報」の範囲内において、MCPサーバとして正しく動作することをサポート <small>注1)</small>
LLMクライアント (サンプル)	×	サンプル提供物であり、サポート対象外
生成AIの回答	×	生成AIがHinemos MCPサーバを利用して生成する回答の正しさ、品質はサポート対象外

注1) Hinemosから正しい情報を取得すること、その情報を正しく生成AIにインプットすることをサポートします。生成AIの回答はサポート対象外です。

想定ユースケース

Hinemos AIエージェントが想定するユースケースは以下の通り

カテゴリ	目的	想定ユースケース
確認	異常の発生有無を知る	重要度「危険」イベント有無を確認 異常終了しているジョブ有無を確認 長時間実行しているジョブを検知
確認	ジョブの実行予定を知る	実行予定のジョブの把握 特定のジョブの実行予定を確認
分析	発生した事象をサマライズする	イベント発生状況をカテゴリ別に集計 ジョブ実行状況をカテゴリ別に集計
調査	障害の事象を知る	危険イベントの内容・事象を把握 異常終了ジョブの内容・事象を把握 障害ノードに対する他の監視結果から原因を推測
調査	障害の発生個所を特定する	イベント傾向から怪しいノードを特定 ジョブ異常と監視履歴から障害ノード特定 障害ノードと同スコープの影響範囲を確認
対処	特定した発生個所を調査する	特定したノードの疎通確認・手動調査
対処	一次的な対処を実施する	障害ノードの監視を一時的に無効化

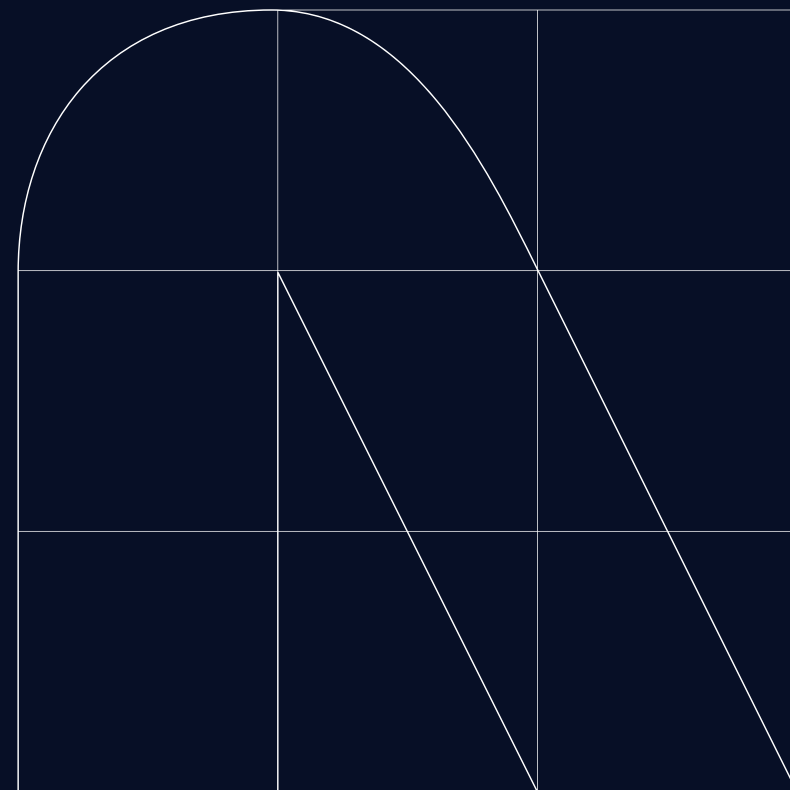
取得可能なHinemos情報

Hinemos AIエージェントが取得可能なHinemos情報は以下の通り

カテゴリ	取得可能情報	利用イメージ
結果の参照	<ul style="list-style-type: none">・ イベント一覧取得・ ステータス一覧取得・ ジョブ履歴一覧取得・ ジョブスケジュール予定取得	<p>Q. 昨日発生した重要度危険のイベントは？</p> <p>A. 昨日は10件のイベントが発生しています。代表的なものをお伝えします。 監視項目ID: MON_01、ファシリティID: NODE_01、メッセージ: ...</p>
対象の設定の参照	<ul style="list-style-type: none">・ 監視設定取得・ ジョブ設定取得	<p>Q. MON_01ってどういう監視内容？</p> <p>A. 「MON_01」はPINGコマンドの応答時間を閾値によって監視しています。 設定内容の詳細をお伝えします。スコープ: REGION_01、...</p>
関連する設定の参照	<ul style="list-style-type: none">・ ノード設定取得・ スコープ設定取得	<p>Q. NODE_01って何のサーバー？</p> <p>A. 「NODE_01」は説明によるとアプリケーションサーバーのようです。 ノードの詳細をお伝えします。IPアドレス: 127.0.0.1、...</p>
設定一覧の参照	<ul style="list-style-type: none">・ 監視設定一覧取得・ ジョブツリー取得	<p>Q. 現在動いている監視の一覧を教えてください</p> <p>A. 現在は20件の監視が有効になっています。一覧をお伝えします。 監視項目ID: MON_01、プラグインID: MON_PING_N、...</p> <p>Q. JOB_01ってどういう構成？</p> <p>A. 「JOB_01」はジョブユニット「UNIT_01」に含まれます。 ジョブツリーの構成をお伝えします。...</p>
設定変更を伴わない実行・有効無効切り替え	<ul style="list-style-type: none">・ 監視有効無効切り替え・ ノードへのPING実行	<p>Q. 監視のMON_01を停止して</p> <p>A. 監視設定「MON_01」を無効化しました。</p> <p>Q. NODE_01にPINGを送ってみて</p> <p>A. 「NODE_01」にPINGを送った結果をお知らせします。...</p>

02

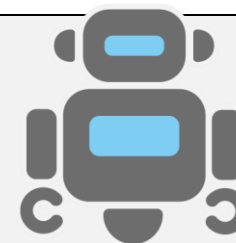
Hinemos AIエージェントの ユースケースイメージ



ユースケース 確認 異常の発生有無を知る

今のシステムの状態はどう？何かクリティカルなアラートは出ている？

今のシステムの状態はどう？何かクリティカルなアラートは出ている？



現在、Hinemos上ではクリティカル(危険)なステータスが1件出ています。

- 監視項目ID: `Ping03`
- プラグイン: PING監視 (`MON_PNG_N`)
- 対象ノード(ファシリティID): `Node03`
- 重要度: 危険 (CRITICAL)
- 最終更新日時: 2026-03-02 11:36:41.009
- メッセージ:
`Packets: Sent = 1, Received = 0, Lost = 1 (100% loss)`

意味としては、`Node03` に対するPING疎通が取れておらず、100%パケットロスしている状態です。

****運用上の次のアクション (提案) :****

1. `Node03` へのネットワーク疎通確認
 - 他サーバから `ping Node03` / ホスト名 or IP で確認
 - 必要に応じて traceroute などで経路調査
2. `Node03` 自体の状態確認
 - 物理/仮想マシンの電源・ハイパーバイザ側コンソール確認
 - OSがハングしていないか、シャットダウンしていないか
3. 想定された停止かどうかの確認
 - メンテナンスや計画停止中であれば、監視一時停止やメンテナンスカレンダの適用を検討

必要であれば、このノードの詳細情報や、同ノードに関する他のイベント履歴 (WARNINGなど) も取得して状況を整理しますが、確認しますか？

ユースケース 分析 発生した事象をサマライズする

最近、どのスコープでアラートが多い傾向にある？

最近、どのスコープでアラートが多い傾向にある？

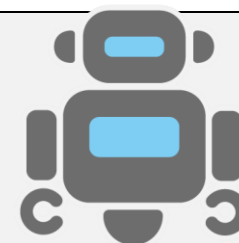


スコープごとのアラート傾向を出すには、「イベント履歴（監視履歴[イベント]）」を集計する形になります。

まず、どの範囲で確認したいか教えてください。

1. 期間
2. 対象の重要度
3. 対象スコープ
 - 無指定なら全スコープを対象

この3点を教えていただければ、その条件でイベント履歴を取得し、スコープごとのアラート件数を集計して、「どのスコープでアラートが多いか」を一覧でお出します。



条件は任せます

直近7日間のイベント履歴（危険／不明／警告）を全件検索して集計した結果、次の通りです。

- アラートが多いスコープ（ファシリティID）
 - ****Node03****：少なくとも ****100件以上****（危険）
 - 監視項目ID: `Ping03`
 - メッセージ: `Packets: Sent = 1, Received = 0, Lost = 1 (100% loss)` が連続発生

取得できたデータでは、****ほぼ全てのアラートが Node03（Ping監視）に集中している****状態です。

つまり、最近アラートが多いスコープは ****「Node03」**** になります。

推測としては、Node03 へのネットワーク疎通が継続的に失敗している、もしくは Ping監視の閾値／リトライ設定によって短間隔で危険イベントが蓄積している可能性があります。

必要であれば：

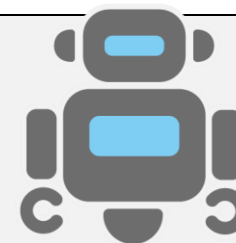
- Node03 への疎通確認（`repositoryPing` 相当、もしくは実環境で ping 実行）
- `Ping03` 監視の設定（監視間隔・リトライ回数・閾値）の見直し

を一緒に確認していきましょう。

ユースケース 調査 障害の発生個所を特定する

エラーイベントが多発しているホストを特定して

エラーイベントが多発しているホストを特定して



取得できた直近のイベント履歴（2026-02-25 16:35:20.026 ~ 2026-03-02 15:44:50.013）の中で、

- 重要度「危険」のイベント件数: 33件
- その全てが `facilityId: Node03`（監視項目ID: `Ping03`）からのイベント
 - メッセージ: `Packets: Sent = 1, Received = 0, Lost = 1 (100% loss)`
 - = Ping監視で疎通不能（100%ロス）が継続して発生

このため、「エラーイベントが多発しているホスト」は ****Node03**** と特定できます。

もし次の対応をするなら、

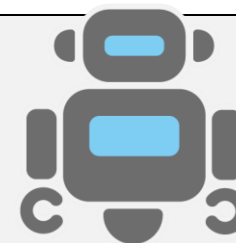
- Node03 への Ping を手動で実行（Hinemosから `Node03` に対して repositoryPing 実行）
- ネットワーク/ホスト側の障害（電源断、NIC障害、ルータ設定など）を確認

といった切り分けを進めるのがよいです。

ユースケース 調査 障害の事象を知る (1/4)

『Ping01』のエラー、具体的に何が原因か教えて

『Ping01』のエラー、具体的に何が原因か教えて



1. 事実として分かっていること

先ほどのイベントは、いずれも以下の内容です。

- プラグインID: `MON_PNG_N` (PING監視)
- 監視項目ID: `Ping03`
- ファシリティID: `Node03`
- メッセージ:
`Packets: Sent = 1, Received = 0, Lost = 1 (100% loss)`

Hinemos の PING監視(MON_PNG_N)は、監視対象ノードのIPアドレスに対してOSレベルの ping コマンドを実行し、その成否で死活を判定します。
このメッセージが出ている =

> Hinemosマネージャから Node03 に送った ICMP Echo (ping) に ****一切応答が返ってこなかった****

というところまでは、確定した事実です。

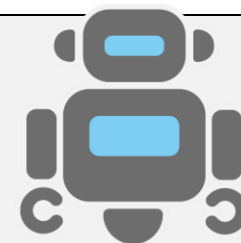
また、この「100% loss」のイベントが数分間連続しているので、

> 一時的な1パケット落ちではなく、****その間ずっと疎通不能状態が続いていた****

ことも事実と言えます。

ユースケース 調査 障害の事象を知る (2/4)

『Ping01』のエラー、具体的に何が原因か教えて



2. しかし、「具体的な1つの原因」はイベントだけでは特定不可

Hinemosのイベント（PING監視の結果）は「疎通できなかった」という結果だけを示しており、

- 電源断なのか
- OSハングなのか
- NIC故障なのか
- FW/セキュリティ製品によるICMP遮断なのか
- 経路上のスイッチ/ルータ障害なのか
- IPアドレス設定ミスなのか

といった「物理/論理的な原因」までは持っていません。

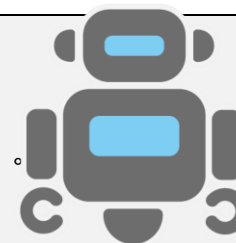
したがって、**イベント情報だけから「これが唯一の原因です」と断定することはできません。**
分かるのは「Node03に対して ping が通らない状態が続いていた」という事実までです。

ユースケース 調査 障害の事象を知る (3/4)

『Ping01』のエラー、具体的に何が原因か教えて

3. イベント内容から見て、現実的な原因候補

「Ping03 だけが NG」「Node01 / Node02 の Ping は OK」という状況から、以下のような可能性が高いと考えられます（推測です）。



1. **Node03 自体の障害**

- 電源断
- OS停止 / ハング
- 再起動中
- NICダウン など

2. **Node03 周辺のネットワーク障害**

- Node03 が接続されているスイッチポートのダウン
- VLAN/経路設定の変更ミスで、Hinemosマネージャからの経路だけ切れた
- ケーブル抜け・断線

3. **ICMP を遮断する設定・機器の影響**

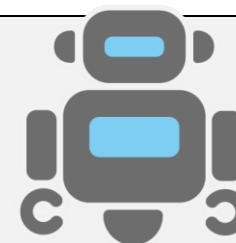
- Node03 のOSファイアウォール設定で ICMP (echo request/reply) が拒否されるように変わった
- 間に入っているFW/セキュリティ製品で、Node03宛の ICMP がブロック、またはレート制限されている

4. **設定ミス**

- Hinemos の Node03 のIPアドレスが誤っている
- Node03 側のIP変更があったが、Hinemos側未更新

ユースケース 調査 障害の事象を知る (4/4)

『Ping01』のエラー、具体的に何が原因か教えて



4. 具体的な切り分け手順 (次にやると良いこと)

もしまだ原因調査中であれば、以下の順で確認するをお勧めします。

1. **現在の疎通確認**

- Hinemosマネージャが動いているサーバから、Node03 のIPアドレスへ手動で `ping` 。
- 可能なら `ssh` / `RDP` / 管理ツール (iLO/iDRAC等) でログインできるか確認。

2. **Hinemosからの疎通確認**

- Hinemosクライアントから Node03 (ファシリティID: `Node03`) に対して「ノードに対するPING実行」を試す。
- 同じく 100% loss になるか、今は復旧しているかで、**一時的障害か継続中か**が分かります。

3. **Node03側の状態チェック**

- 電源状態 (コンソールやリモート管理ツール)
- OSログ (システムログ、ネットワークログ)
- NICリンク状態 (link up/down)

4. **ネットワーク機器/FWのログ・設定確認**

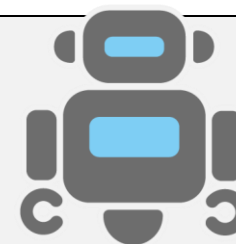
- Node03 の接続スイッチのポート状態
- 間にあるFWで、該当時間帯に ICMP のブロックログやエラーがないか
- 経路変更やACL変更の履歴

5. 必要なら手順を一緒に詰められます

ユースケース 対処 一次的な対処を実施する (1/4)

不安定なノードはある？ある場合、メンテしたいから監視を全て止めて

不安定なノードはある？ある場合、メンテしたいから監視を全て止めて



不安定なノードはあります。

1. 不安定なノード (監視ステータス)

- DB01
 - 監視項目ID: `PingDB` (PING監視 / MON_PNG_N)
 - 重要度: 危険
 - メッセージ: `Packets: Sent = 1, Received = 0, Lost = 1 (100% loss)`
 - 最終変更日時: 2026-03-02 17:52:32.008
- Node03
 - 監視項目ID: `Ping03` (PING監視 / MON_PNG_N)
 - 重要度: 危険
 - メッセージ: `Packets: Sent = 1, Received = 0, Lost = 1 (100% loss)`
 - 最終変更日時: 2026-03-02 17:53:11.010

※上記は監視履歴[ステータス]から、「警告以上 (CRITICAL/UNKNOWN/WARNING)」で絞って取得した結果です。

ユースケース 対処 一次的な対処を実施する (2/4)

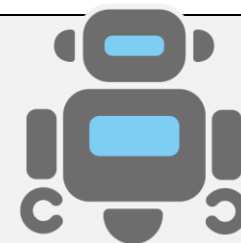
不安定なノードはある？ある場合、メンテしたいから監視を全て止めて

2. 停止対象になりそうな監視設定

該当ノードに紐づく監視設定を確認すると、以下のようになっています (抜粋) :

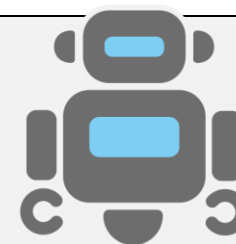
- DB01 向けの個別監視
 - `PingDB`
 - monitorId: `PingDB`
 - monitorTypeId: `MON_PNG_N` (PING 監視 / 数値監視)
 - facilityId: `DB01`
 - 監視有効: `true`
- Node03 向けの個別監視
 - `Ping03`
 - monitorId: `Ping03`
 - monitorTypeId: `MON_PNG_N` (PING 監視 / 数値監視)
 - facilityId: `Node03`
 - 監視有効: `true`

同じ一覧の中に `ANDROID_*` や `WINDIRECTOR_*` など、`facilityId = REGISTERED` の監視も多数ありますが、これは「登録ノードすべて」に共通で効くテンプレートです。これらが無効化すると**他のノードの監視まで止まる**ため、メンテ対象ノードだけ止めたい場合は外した方が安全です。



ユースケース 対処 一次的な対処を実施する (3/4)

不安定なノードはある？ある場合、メンテしたいから監視を全て止めて



3. 監視停止の方針について確認させてください

実行系（監視有効フラグ変更）は影響が大きいので、次のどちらにするか教えてください。

A. メンテ対象ノードのPING監視だけ止める（推奨・影響小）

- 無効化する監視:
 - `PingDB` (facilityId: DB01)
 - `Ping03` (facilityId: Node03)
- 影響: DB01 / Node03 の死活 (PING) 監視だけが止まります。他ノードの監視や共通監視には影響しません。

B. メンテ中は、DB01 / Node03 に関する監視を全部止めたい

- `PingDB` / `Ping03` に加えて、「REGISTERED 向けの共通監視」も止めるかどうかを決める必要があります。
- 共通監視を止める場合、**全登録ノードに対する監視が止まる可能性**があり、システム全体に影響します。

どちらの方針にしますか？

例:

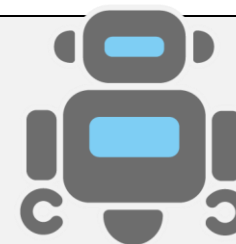
- 「AでOK。今すぐ `PingDB` と `Ping03` の監視を止めて」
- 「Bで、全ノードの監視が止まっても構わないので全部止めて」

方針と「実行してよい」という意思表示をいただければ、対象と影響内容を明示した上で、`monitorId` を指定して監視無効化（`monitorsettingsetStatusMonitor`）を実行します。

ユースケース 対処 一次的な対処を実施する (4/4)

不安定なノードはある？ある場合、メンテしたいから監視を全て止めて

AでOK



次の2つの監視を「無効化」しました (monitorFlg = false) :

1. DB01 向け
 - 監視項目ID: `PingDB`
 - 種別: 数値監視 (PING監視 / MON_PNG_N)
 - facilityId: `DB01` / scope: `DB01`
 - 更新日時: 2026-03-02 17:56:10.929 (updateUser: hinemos)
2. Node03 向け
 - 監視項目ID: `Ping03`
 - 種別: 数値監視 (PING監視 / MON_PNG_N)
 - facilityId: `Node03` / scope: `Node03`
 - 更新日時: 2026-03-02 17:56:10.940 (updateUser: hinemos)

【事実】

- 上記2件の監視設定は、Hinemos上で「監視有効 = false」に変更されています。
- これ以降、DB01 / Node03 については PING 監視によるアラートは上がりません。

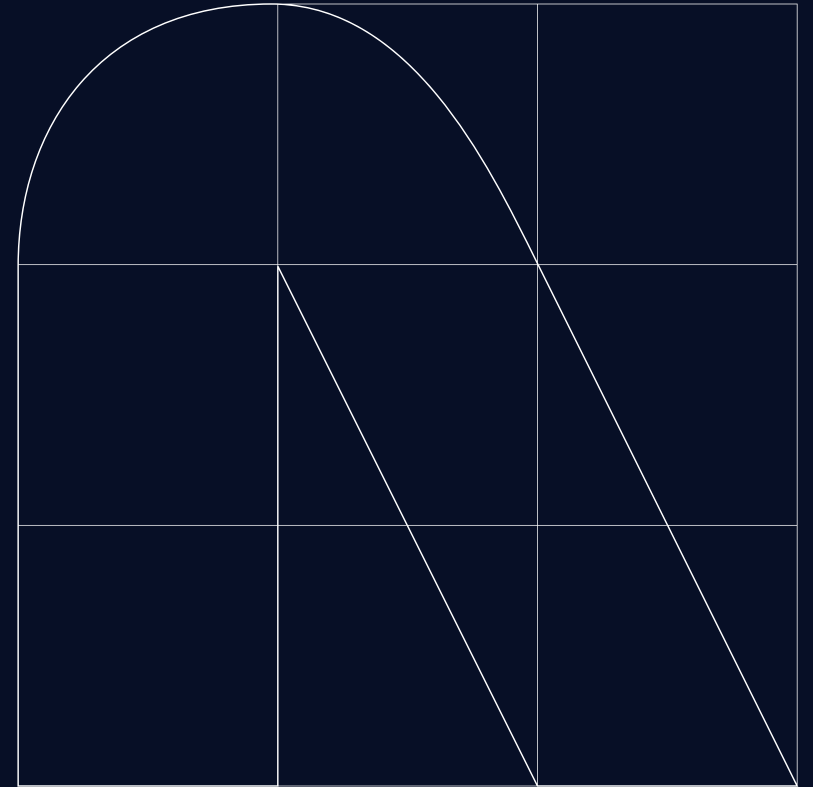
【推測される影響】

- メンテ中に DB01 / Node03 が停止・疎通不可になっても、PING監視のイベントは発生しないため、意図しないアラートは抑止できていると考えられます。
- 他ノードの監視設定 (`REGISTERED` 向け共通監視など) は変更していないため、他ノードへの影響はありません。

メンテが終わったら、同じ2つの監視 (`PingDB`、`Ping03`) を再度有効化する必要があります。
終わったタイミングで「監視を元に戻して」などと指示してもらえれば、再度有効化します。

03

動作要件



動作要件

Hinemos AIエージェントの各コンポーネントの対応環境は以下の通り

Hinemos MCPサーバ	対応OS	RHEL 8, 9, 10 Oracle Linux 8,9,10 Amazon Linux 2023 Windows Server 2019, 2022, 2025
	Hinemosバージョン	Hinemos ver.7.2
	動作検証済みLLMモデル	Azure Open AI : GPT-4o, GPT-5.1-Global, GPT-5.2-Global Amazon Bedrock : Claude Opus 4.6
LLMクライアント (サンプル)	動作検証済みOpen WebUIバージョン	Open WebUI v0.7.2
	動作検証済みOS	RHEL 10 Oracle Linux 10 Windows 11, Windows Server 2019, 2022, 2025

