生成AIによるNetwork Automation

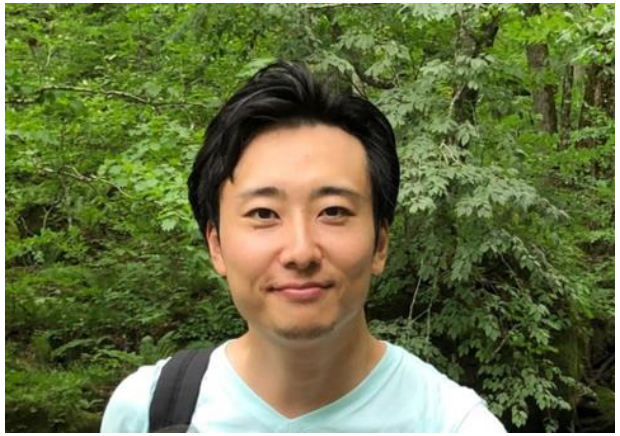
～LLMエージェントはネットワークオペレータになれるのか～

佐藤 亮介(株式会社NTTフィールドテクノ)
白井 嵩士(株式会社NTTフィールドテクノ)
宮川　優一(日本マイクロソフト株式会社)

メンバ紹介

・名前:佐藤 亮介
・現担当:NW運用/保守
・JANOG歴:登壇3回目
・趣味：ITガジェット・仮想通貨

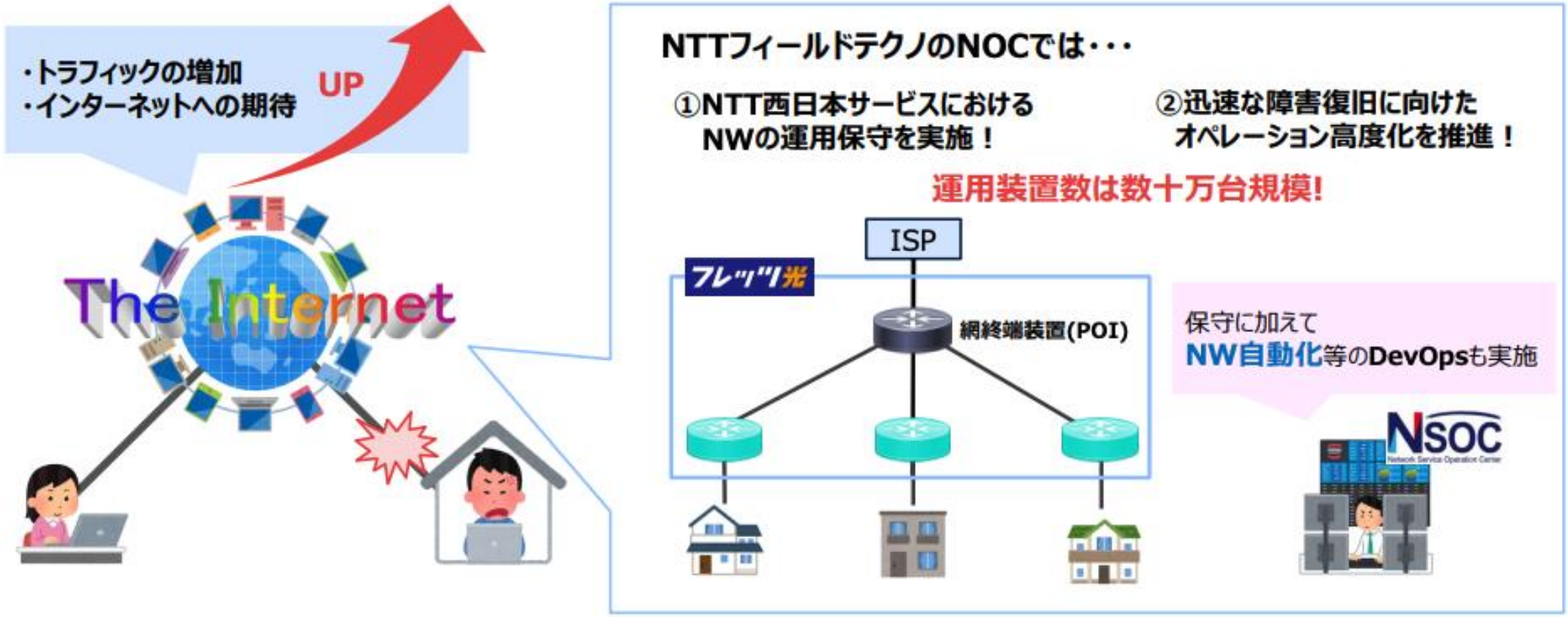
・名前:白井 嵩士
・現担当:NW運用/保守
・JANOG歴:登壇2回目
・趣味：局舎巡り、川魚のお世話



・名前　宮川優一
・現担当　AI/Analytics/DBの専門営業
・JANOG歴：今回初参加
・趣味：サウナ、料理

NTTフィールドテクノ業務紹介

- リモートワークの普及や大規模なネットワーク障害を通してNW保守の重要性が高まっており
- NTTフィールドテクノのNOCでは日々迅速な障害復旧に向けて努めています



Microsoft AI Co-Innovation Lab

(旧名称:AI & IoT Insider Labs)

Microsoft AI Co-Innovation Lab とは
お客様のビジネスに対する IT の適用可能性を
確認することができるラボ施設です

“地球上のすべての個人とすべての組織が、より多くのことを達成できるようにする”というマイクロソフトのミッションに基づき
お客様の製品・サービス・ビジネスのデジタル変革を支援するラボ施設を
グローバルに保有しております。

お客様のシステム・サービスに弊社の AI や IoT 製品をどのように適用させてビジネス化できるのかMicrosoft ラボエンジニアの支援の下
構築・開発・プロトタイプ作成・テストを行う機会と施設を提供致します。

通常長期にわたるテクノロジー調査やアーキテクチャ検討を
このラボのエンゲージメントにより**短期間で**進めることができます。
お客様環境や実データを用いてプロトタイプを作成することもできるため
商品化に向けた実用的な検証が可能です。

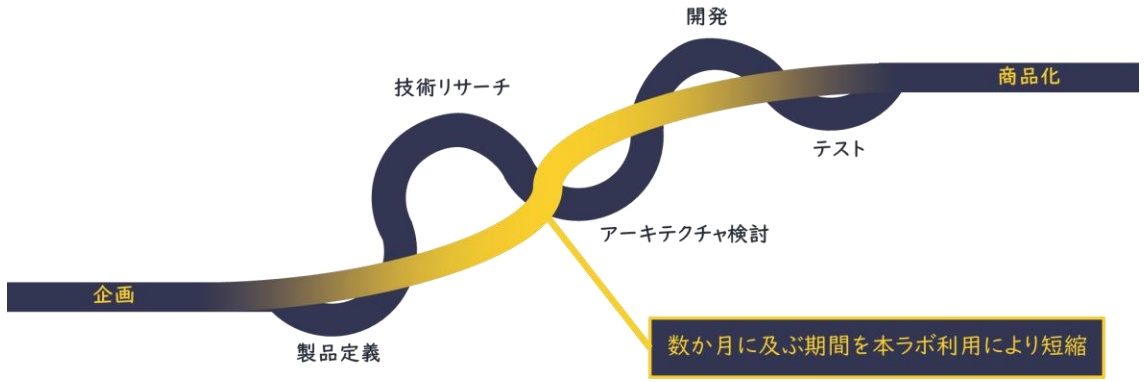
グローバルに	稼働エンジニア	エンゲージメント実施数
6 拠点*	20 名以上	800 回以上

*レドモンド, サンフランシスコ, ミュンヘン, 上海, ウルグアイ, 神戸
*2023/10/11時点

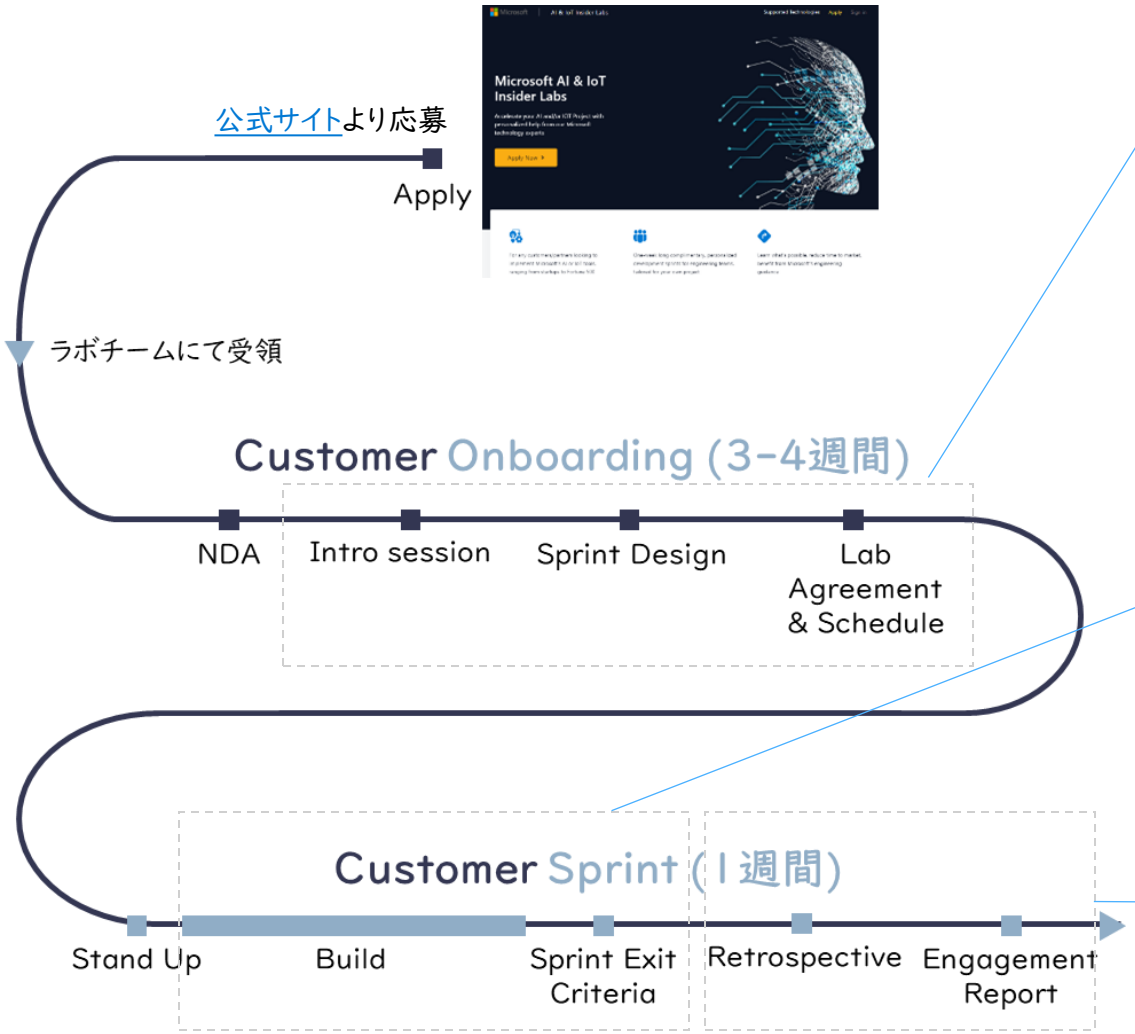
技術領域

- IoT によるデータ収集、AI による分析、分析結果の可視化
- OpenAI の実用化
- その他クラウド技術の適用

迅速なビジネス化を支援

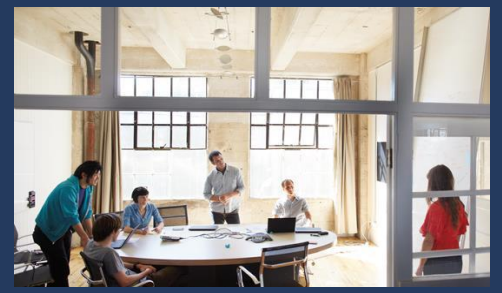


ラボエンゲージメント利用の流れ



事前ディスカッション(リモート)

- お客様ビジネスの全体感と
  今回のエンゲージメントの対象事項の
  背景情報の共有
- 1週間のSprint期間(マイクロソフト
  エンジニアとの共同開発期間)における
  対応範囲の定義
- アーキテクチャ検討、環境・データ・ツール
  等の事前用意の実施



1週間の共同開発の実施(ラボ施設内実施)

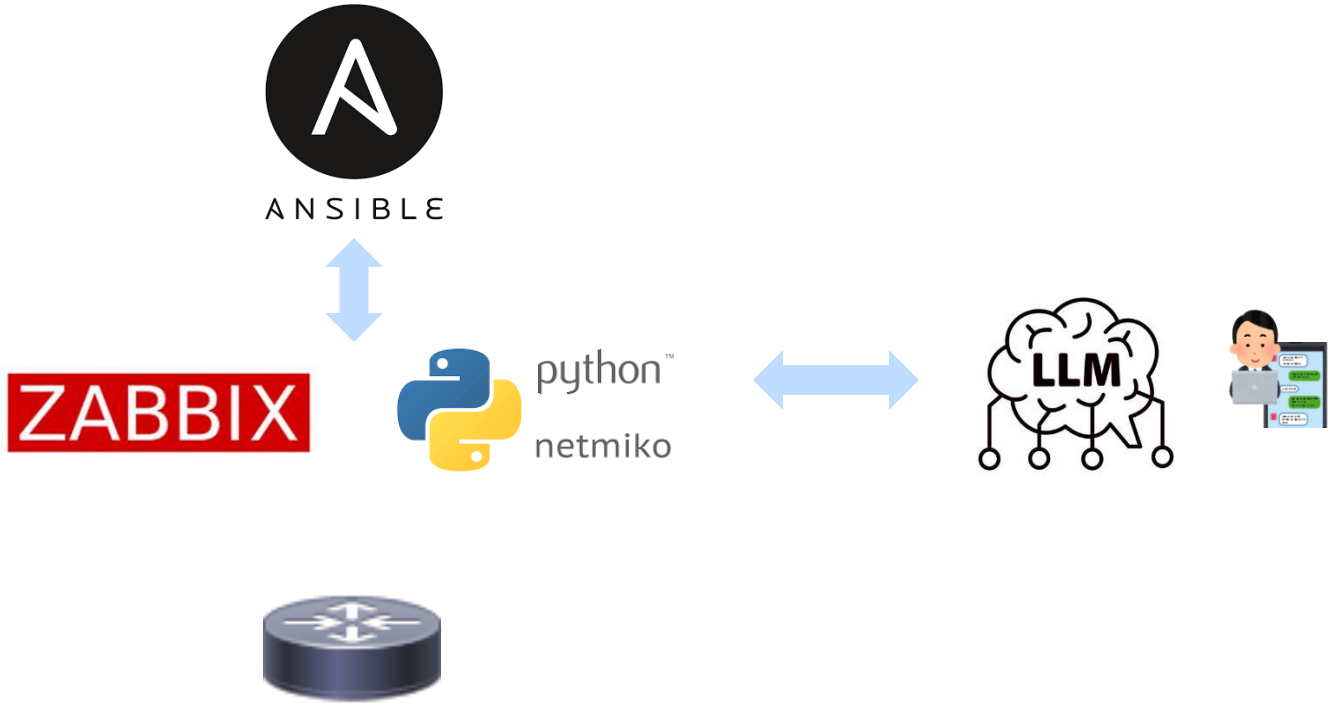
- 物物理的なラボスペースにてラボエンジニア支援の下、開発作業を実施

クロージング

- サービス・商品化に向けた今後の発展性を含めたレポーティングの実施
- パートナー企業様・スタートアップ企業様、マイクロソフト支援スキームのご紹介
- ご要望に応じて 事例化の実施、共同PRの実施

取り組みの背景・目的

- 大規模かつ成長し続けるNWを限りある人的リソースで保守するため，様々なケースのNW自動化に取り組んでいます．
- 生成AI技術に大きな期待をしており，現在の**Network Automationをより高度に拡張しオペレータの業務を全面的に任せられるレベル**にすることを目的として検討を進めています．



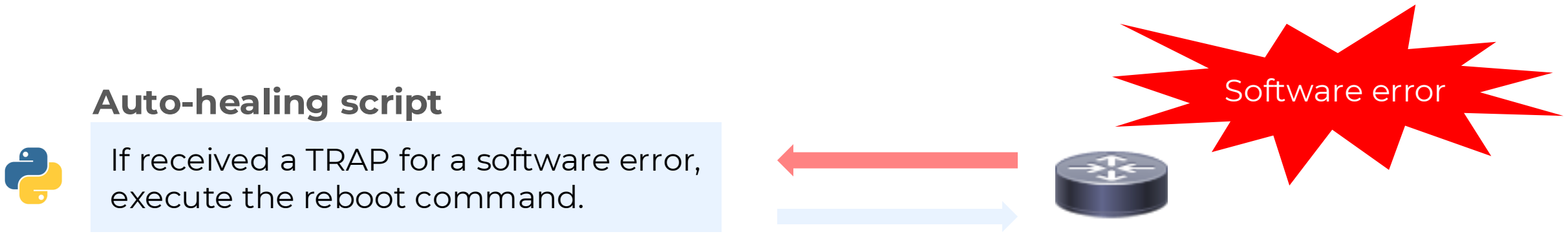
取り組みの背景・目的

- 前回JANOG54では**Local LLM**を中心としたアーキテクチャによりトラブルチケットに対する単純なQAや,ツールの実行など, **NWオペレーションの基本的な動作**を実現する例を紹介しました.

- 検討環境の拡張（GPTモデル、Azure環境など）とともにMicrosoft様に技術的な支援をいただきながら、より**実際の監視オペレーションに近いユースケースの実現**に向けて検討を進めてきました. 今回は**NWトラブルの切り分け**をLLMエージェントにて実施する例とともに技術的なアップデートを紹介させていただきます

現在のNW自動化ユースケース

- Ansibleなどのオーケストレータやプログラム言語で静的にスクリプト化できるケースに限定されている
  - ルータのソフトウェアエラーに対する自動リブート実施（Auto healing）
  - ログ収集などの定期タスク
- FT NOC（NSOC）では実装可能なケースの大部分がすでに実装済で，コスト削減効果が頭打ち



生成AIで拡張したい自動化ユースケース

- NWトラブルの切り分けなどはオペレータの状況に応じた判断が必要で静的なスクリプト化が難しい
- 判断部分をLLMエージェントで再現したい

NWトラブルの切り分け手順の例

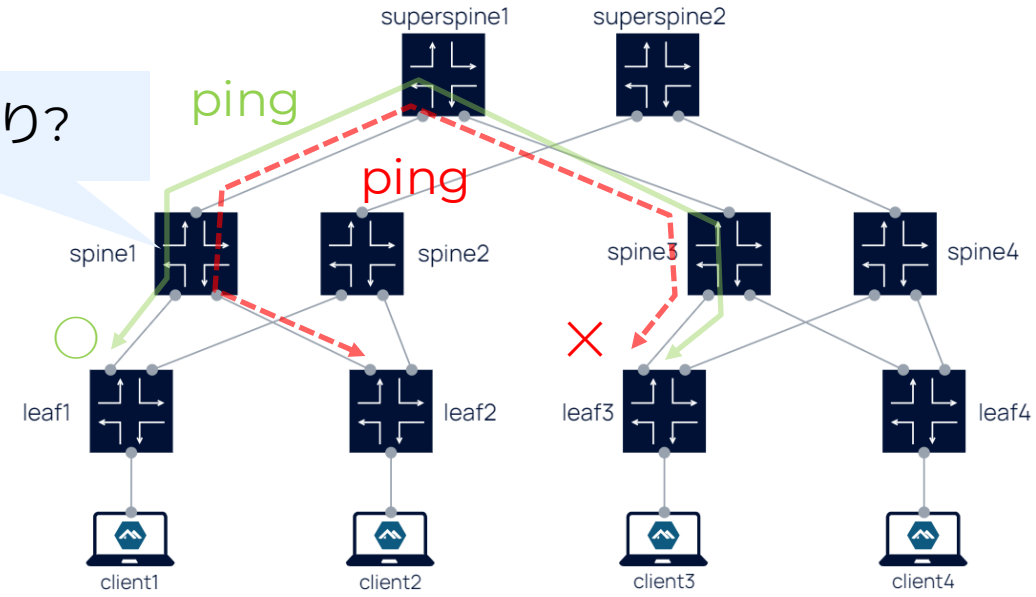
一部ユーザから通信不可申告あり！

まずはアラームの確認
→ 特になし,,,

Pingなどで疎通確認
→ パケットロスを確認

ping経路上の装置の設定確認
→ 怪しい設定を発見!

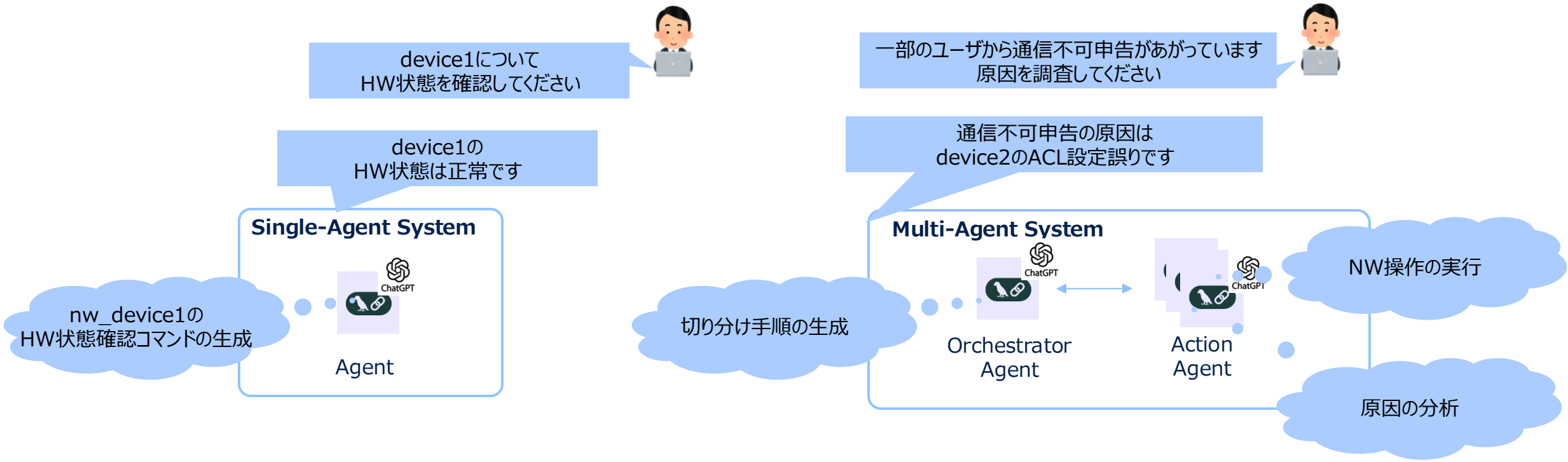
config誤り?



生成AIのNetwork Automationへの活用に関する課題とアプローチ

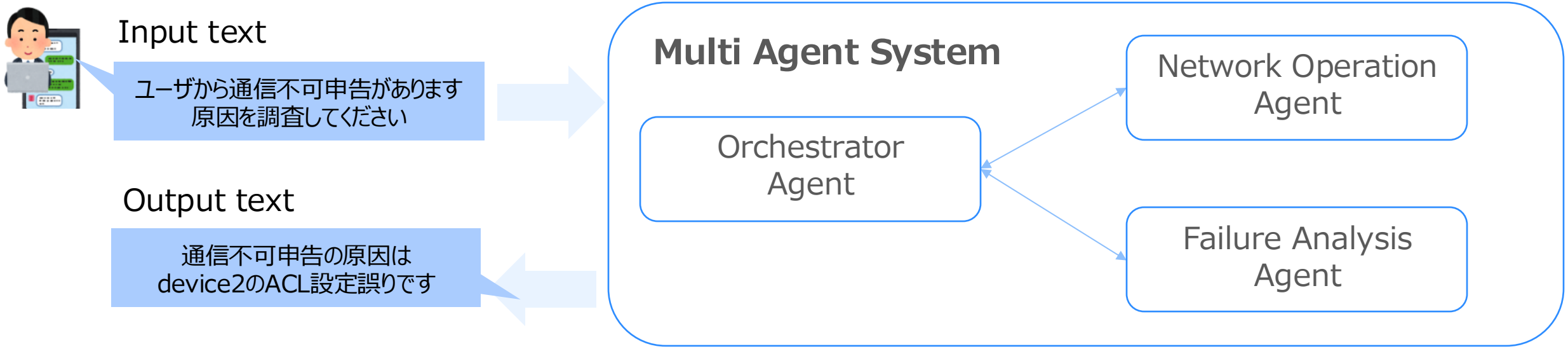
【課題】マルチエージェントの構成

- 単体のLLMエージェントのチューニングで単一のタスクの精度はあげられるが
  発展的なユースケースは複数のタスクが必要になる
- タスクに特化したエージェントを組み合わせたマルチエージェントの構成検討が必要



NWトラブル切り分けのためのマルチエージェントの構成例

- 以下の3エージェントにより構成
  - オペレーションの全体統制（Orchestrator）
  - NW装置への操作実行（Network Operation）
  - トラブル原因の解析（Failure Analysis）



Orchestrator Agentの詳細

- ReAct(Reasoning and Action)プロンプティングをベースに，他のAgentへの指示と，結果を受けた次の指示を推論する．（オペレータの状況に応じた判断を模擬）

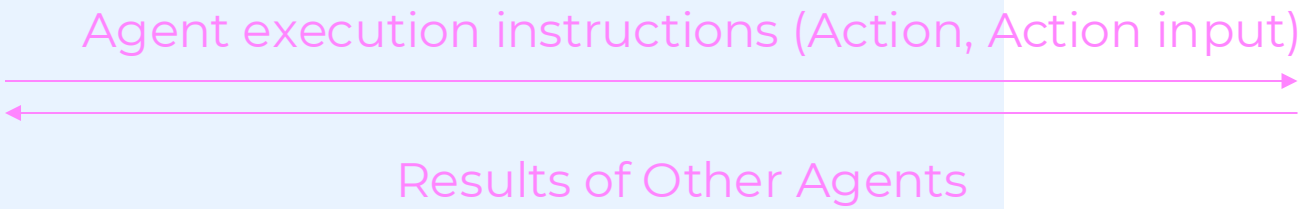
プロンプト概要

あなたは優れたネットワークオペレーターです.以下の質問に可能な限り最善の回答をしてください。あなたは次のエージェントにアクセスできます：
{Agents}

以下の形式を使用してください：
Question: 回答すべき入力質問
Thought: 何をすべきか
Action: 実行するアクション（[{tool_names}]のいずれか）
Action Input: アクションへの入力
Observation: アクションの結果
Final Answer: 入力された元の質問への最終的な答え

network topology
{nw_topology}

Begin !
Question: {question}
{results}



Other Agents

Network Operation Agentの詳細

- Orchestratorからの指示からNW装置への必要な操作を推論し
  対応するデータフォーマットで出力する

プロンプト概要

あなたは優れたネットワークエンジニアであり、ネットワーク障害分析の最中です。

instruction
• NW装置への内容を決定し、コンテキスト情報から対応するJSON-RPCインターフェースのパスパラメーターを出力してください。

context
{context}

output format
output must be in list of JSON format as below
{{"hostname":"the name of the host determined", "path":"path parameter of the JSON-RPC"}}

Failure Analyze Agentの詳細

- 他のエージェントが集めた情報をもとに障害原因を分析する.

プロンプト概要

あなたは優れたネットワークエンジニアであり、ネットワーク障害分析の最中です。

instruction
障害原因を分析し説明してください

network topology
{nw_topology}

information on network status
{network_status}

analyze policy
- エラーなどの明らかな原因がないか
- 疎通不可能な区間がないか
…

Results of Other Agents

Other Agents

LLMによるNWオペレーション所作の正確性

- LLMはNW運用の知識（特にベンダ特有の実装）に乏しいため，出力の細部に誤りが含まれることがある

Network Operation Agentのプロンプト概要

あなたは優れたネットワークエンジニアであり、ネットワーク障害分析の最中です。

instruction
• NW装置への内容を決定し、コンテキスト情報から対応するJSON-RPCインターフェースのパスパラメーターを出力してください。

context
clab-clos02-leaf2のインターフェース状態を確認したい

output format
output must be in list of JSON format as below
{{"hostname":"the name of the host determined", "path":"path parameter of the JSON-RPC"}}

{
"hostname":"clab-clos02-leaf2",
“path”:“/interfaces/status”（間違い）
}

In-context learningによるLLM出力の正確性の向上

- 知識をプロンプトに付加することで出力の正確性を向上させる（In-context learning）

Network Operation Agentのプロンプト概要

あなたは優れたネットワークエンジニアであり、ネットワーク障害分析の最中です。

instruction
• NW装置への内容を決定し、コンテキスト情報から対応するJSON-RPCインターフェースのパスパラメーターを出力してください。

context
{context}

hint
- The OS of the host is Nokia SR Linux

Path information
- path: /network-instance[name=*]/protocols/bgp
 - description: BGP status
- path: /interface[name=*]/subinterface[index=*]/acl/input/ipv4-filter
 - description: the content about ipv4 ACL(Access Control List)
- path: /acl/ipv4-filter[name=*]/entry[sequence-id=*]
 - description: the configuration of ACL that have been activated
- path: '/interface[name=*]/oper-state'
 - description: interface status
…


```
{
  "hostname":"clab-clos02-leaf2",
  “path”:“'/interface[name=*]/oper-state”
}
```

(参考)Orchestrator Agentに対するIn-context learning

- ReActプロンプトに対しても, 理想的な思考ステップの例をインプット(few shot)することで精度向上可能

あなたは優れたネットワークオペレーターです.以下の質問に可能な限り最善の回答をしてください。あなたは次のエージェントにアクセスできます：
{Agents}

以下の形式を使用してください：
Question: 回答すべき入力質問。
Thought: 何をすべきかを常に考えるべきです。
Action: 実行するアクション（[{tool_names}]のいずれか）。
Action Input: アクションへの入力。
Observation: アクションの結果（観察結果のステップで出力を終了しないでください）。
Final Answer: 入力された元の質問への最終的な答え。

examples
Thought: ...
Action: ...
...

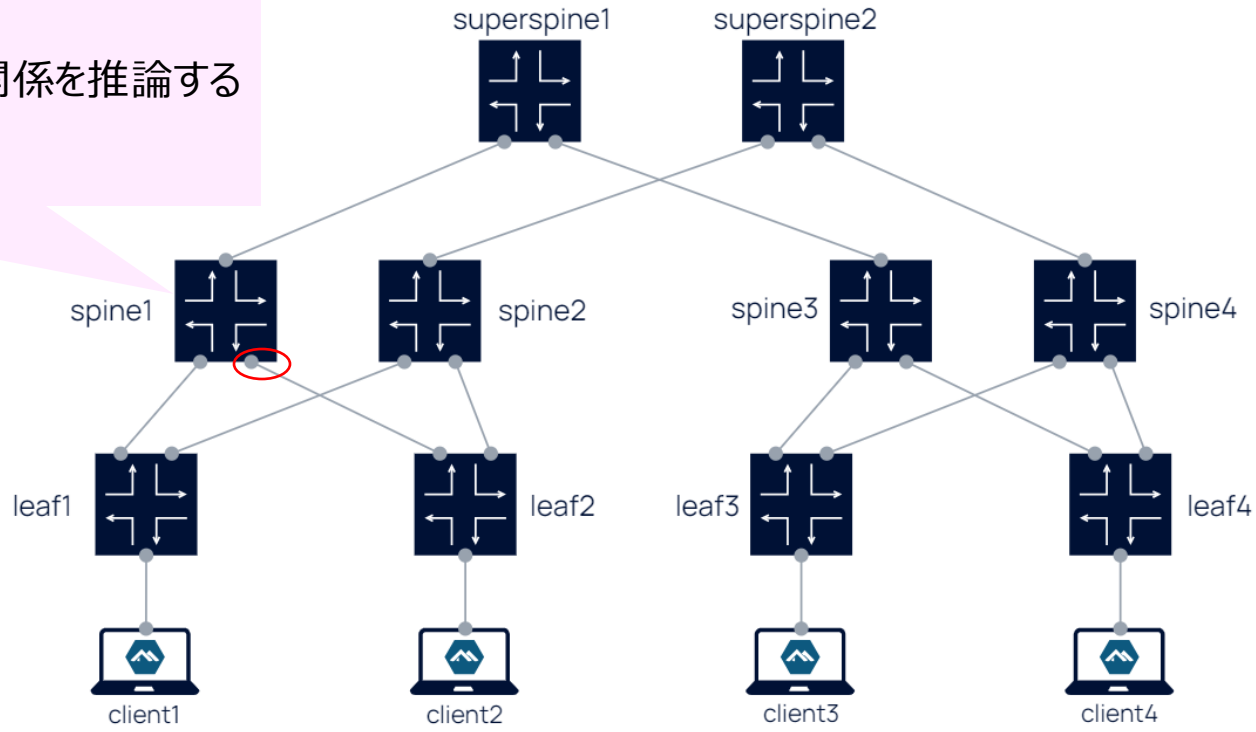
Begin！
Question: {question}
{results}

LLMエージェントを用いたNWトラブル切り分けのデモ

NWトラブル切り分けユースケースの評価

- Containerlabで構成したNW上でNWトラブルを模擬
- 「一部のclientから通信不可申告を受けている」という初期情報のみからLLMに分析を行わせる

ユーザーが通信に利用するプロトコルの通信のみ遮断する**フィルタ設定（ACL）**を適用．該当IFを通過する一部通信が不可能になる
→明示的なTRAPやエラーログが発生しないため
遮断区間を特定し，該当箇所のACL設定を調べ障害との関係を推論するなどの複雑な分析が必要



LLMエージェントのデモ実施予定

評価結果とさらなるユースケース拡張のための課題

- マルチエージェントとIn-context learningを中心としたアプローチにより、実際に起こり得るNWトラブルを臨機応変に判断し解決するLLMエージェントを実装できた.

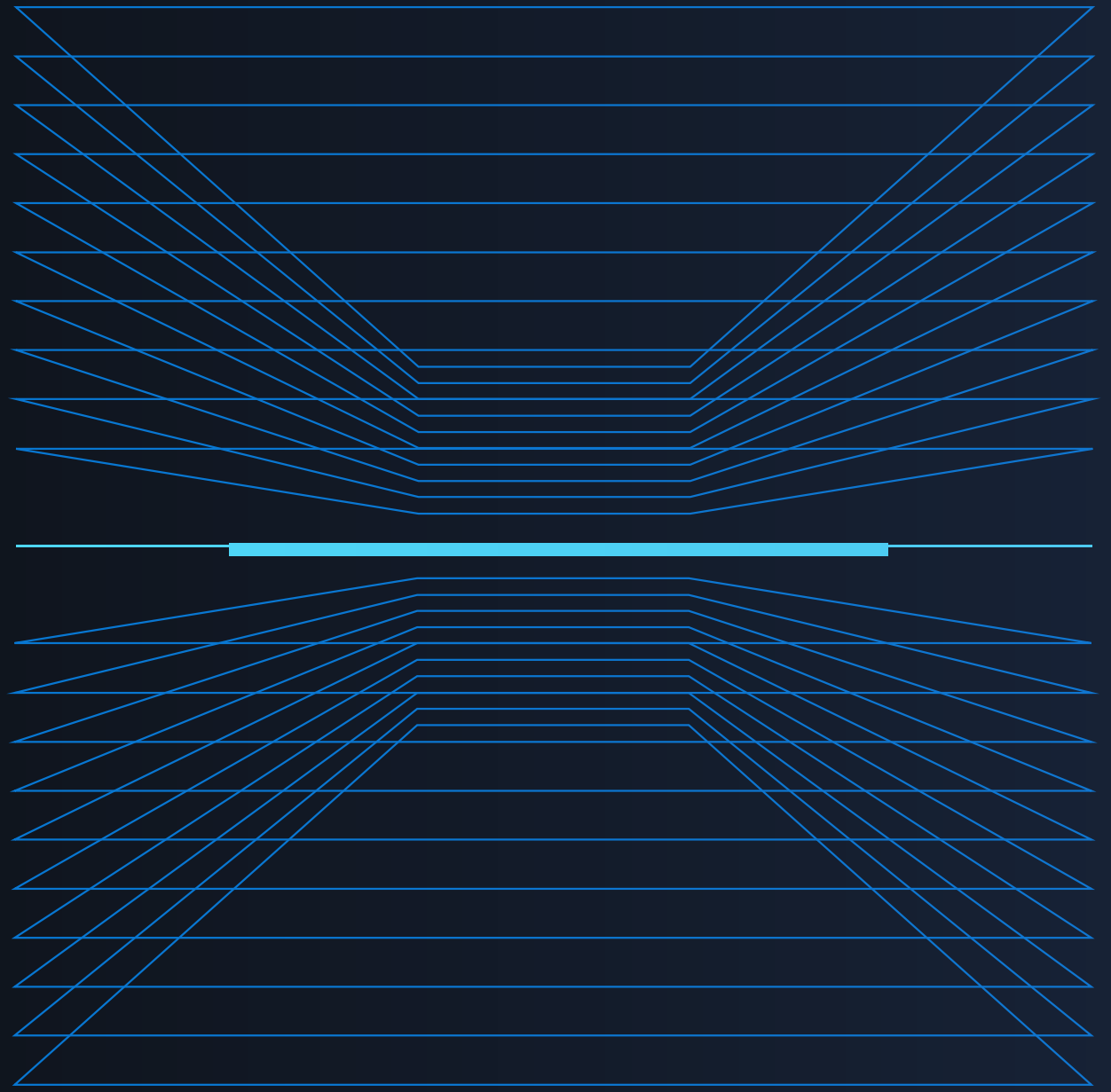
- 対応できるNWトラブルのケースやNWの構成規模を拡張していきたいが In-context learningでプロンプトに付加できるテキスト量は限られているため **状況に応じて情報を検索し、動的にプロンプトに付加する仕組み（検索拡張生成：RAG）のブラッシュアップが必要**
- FTの実装した単純なベクトル検索のRAGでは検索精度が低かったため、Microsoft様の支援を受けAzure AI search技術方式の活用を検討中

Microsoft

RAG 概要

Azure AI Search の
Retriever(情報検索)システムへの活用

1/23/2025



RAG の基本

Reasoning + Knowledge

生成 AI と レトリーバーシステム それぞれの役割

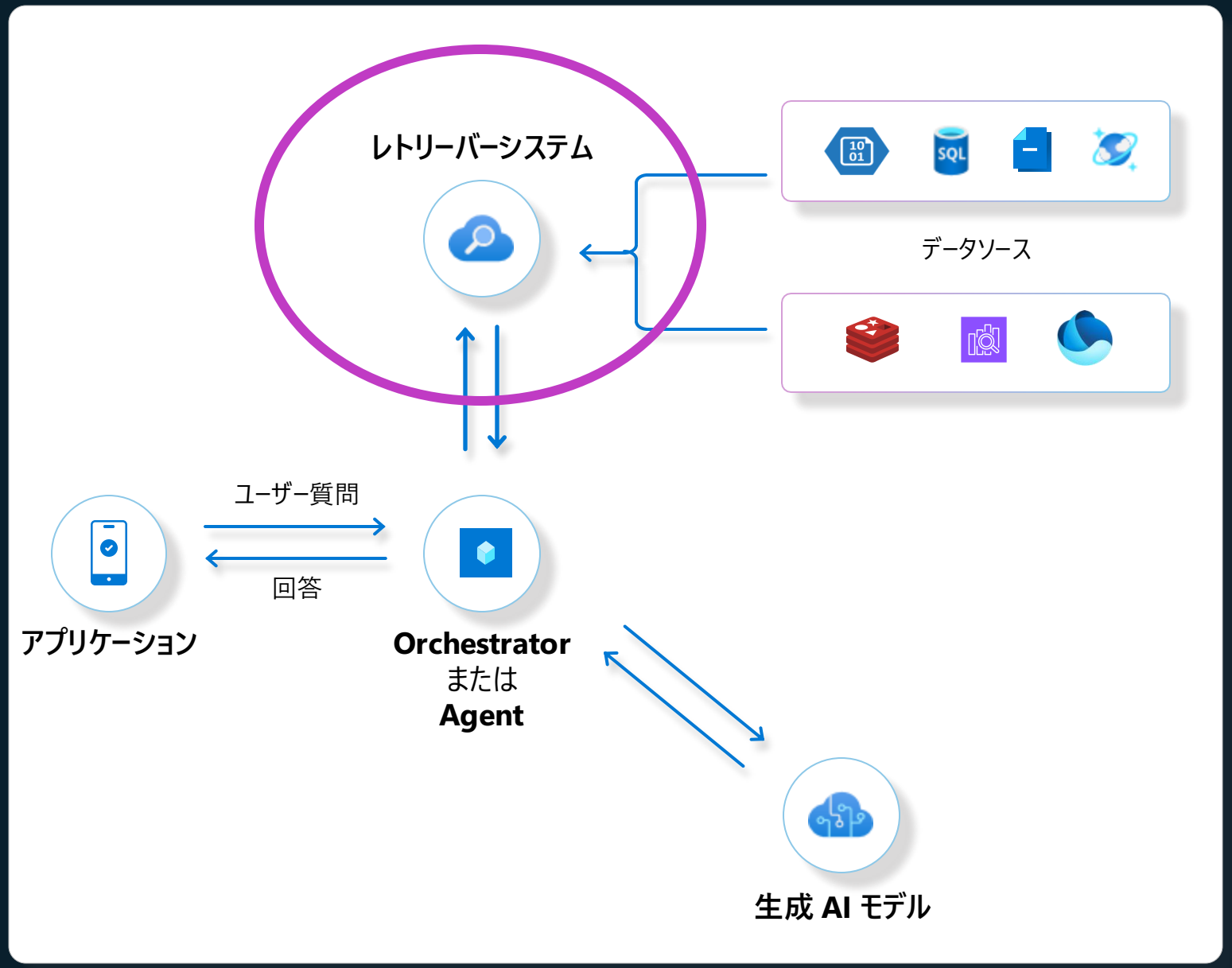
Reasoning

ファンデーションモデルを利用
必要な情報、コンテキスト をつかって回答内容を整理、ツールやエージェントの実行を提案、
質問のフォローアップ

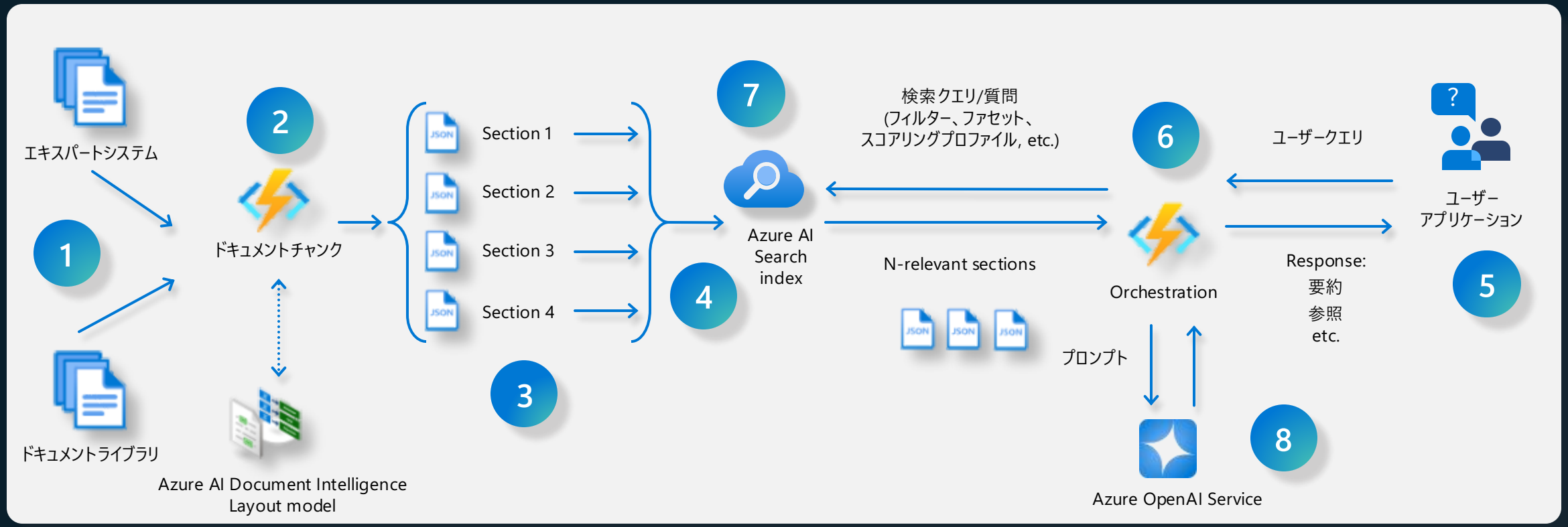
Knowledge

レトリーバーシステムの機能
特定のコンテキストに最も関連性の高い情報を見つける
モデルに期待される機能に合わせて知識を抽出する
データの鮮度、アクセス制御を確保する

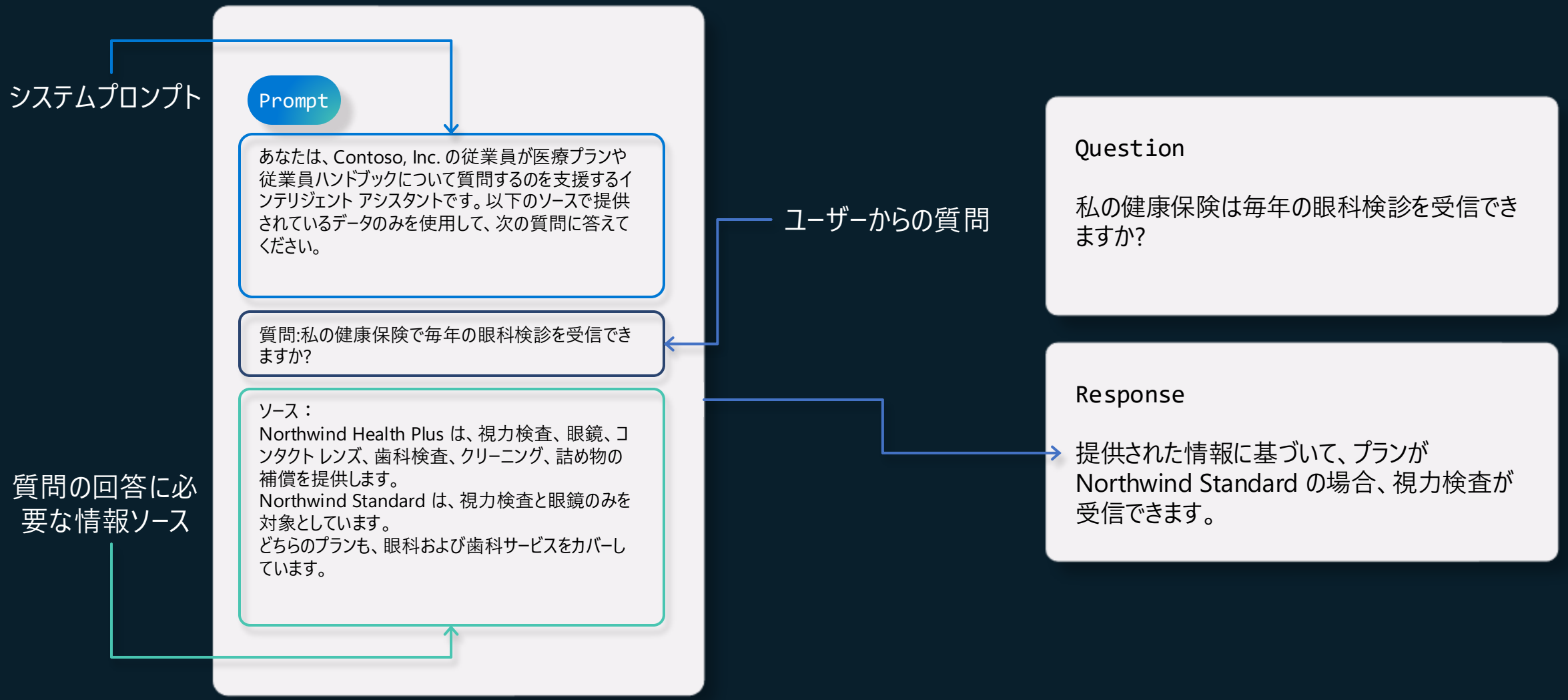
Retrieval-Augmented Generation



RAG の全体像


エキスパートシステム
1
ドキュメントライブラリ
2
ドキュメントチャンク
Azure AI Document Intelligence
Layout model
Section 1
Section 2
Section 3
Section 4
3
4
7
Azure AI
Search
index
検索クエリ/質問
(フィルター、ファセット、
スコアリングプロファイル, etc.)
N-relevant sections
6
Orchestration
プロンプト
Azure OpenAI Service
8
ユーザークエリ
Response:
要約
参照
etc.
ユーザー
アプリケーション
5

RAG の内部 固有のナレッジをプロンプトに挿入



LLM にドメイン知識を与える

プロンプトエンジニアリング

In-context learning

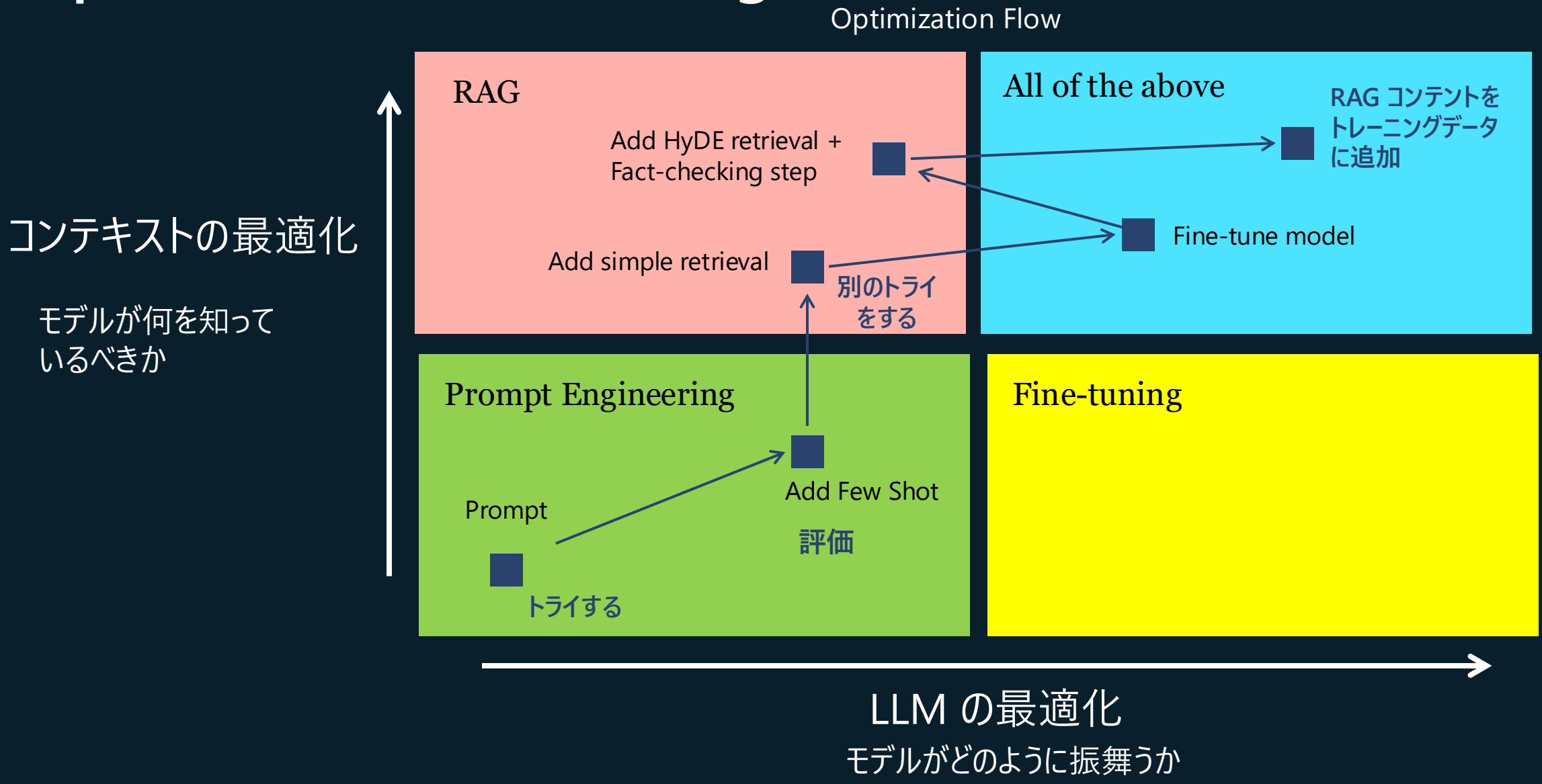
ファインチューニング

Learn new skills

RAG

Learn new facts

OpenAI モデルと Fine tuning



Source: [A Survey of Techniques for Maximizing LLM Performance: OpenAI DevDay 2023]

検索ストラテジー

Keyword search

- **テキストキーワードの一致**
- Q&Aシステムにおける「語彙のギャップ」が生じる

Vector search

- **概念的な類似性や意味の一致**
- 完全一致(商品 ID やコードなど)でのパフォーマンスが弱い

Hybrid search

- **ベクトルとキーワードの両方の長所**
- さまざまなシナリオでより正確な応答を実現

Search re-ranking

- **取得したすべてのドキュメントを関連性によってスコアリングしランク付け**
- 検索結果を再評価

Azure AI Search のクエリー

検索モード

キーワード検索

- 従来の全文検索方法
- コンテンツは用語に分割
- BM25 によるスコアリング

ベクトル検索

- テキストはベクトル表現に変換されます
- Azure OpenAI 埋め込みモデル 等を使用

ハイブリッド検索

- キーワードとベクターの長所を組み合せる
- Fusionステップでは、Reciprocal Rank Fusion(RRF)を使用して、両方の方法から最良の結果を選択します

セマンティック ランカー

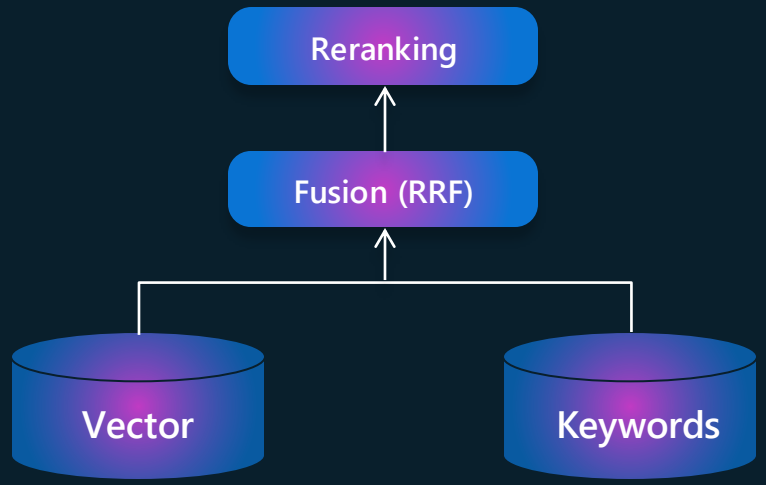
セマンティック検索

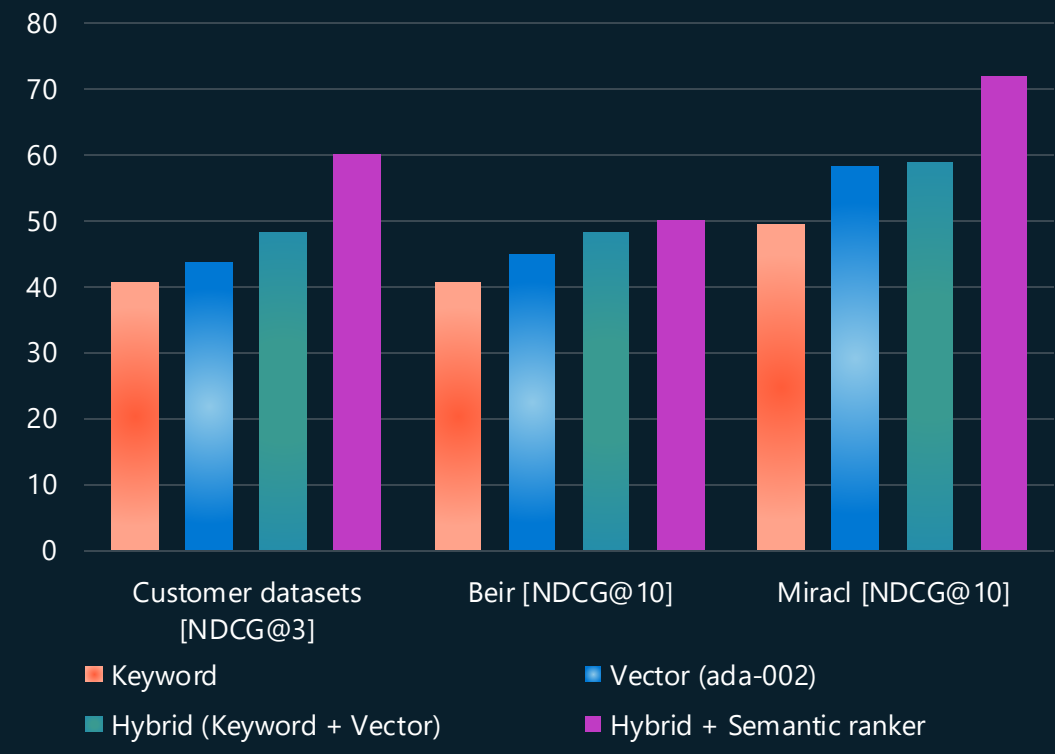
- クロスアテンション機能を備えたトランスフォーマー モデルを使用して、クエリとドキュメントのテキストを同時に処理する Bing テクノロジ
- 最も重要な結果を優先
- 関連性スコアにより 低品質の結果を除外
- Score Range: 0 (irrelevant) to 4 (highly relevant)

検索方式による性能の違い

データ量が多いほど 検索精度が重要になる

- ハイブリッド (ベクトル + キーワード) L1 ステージにより再現率が向上
- L2 ステージ での セマンティック リランキング により ランキング精度が向上
- クエリ パイプラインの豊富な制御
  - (マルチベクトル の重み、キーワード 検索のカットオフ、ベクトルメトリック のしきい値など)





Retrieval comparison using Azure AI Search in various retrieval modes on customer and academic benchmarks
Source: Outperforming vector search with hybrid + reranking

https://learn.microsoft.com/ja-jp/azure/search/hybrid-search-ranking

ベクトル検索の補足資料

ベクトル検索ストラテジー

ANN search

- 大規模での高速ベクトル検索
- HNSWを採用、優れた性能再現率プロファイルを持つグラフ方式
- インデックスパラメータのきめ細かな制御

```
r = search_client.search(
        None,
        top=5,
        vector_queries=[VectorizableTextQuery(
            text=query,
            k_nearest_neighbors=5,
            fields="embedding")])
```

Exhaustive KNN search

- パフォーマンスが分かっている定型クエリー
- リコールベースラインの作成に便利
- 選択性の高いフィルターを使用するケース

```
r = search_client.search(
        None,
        top=5,
        vector_queries=[VectorizableTextQuery(
            text=query,
            k_nearest_neighbors=5,
            fields="embedding",
            exhaustive=True)])
```

[参考] RAG を支える技術 – ベクトル近傍検索

・大量データからの近似ベクトル検索手法

課題

データ量が多くなると
計算量が増える

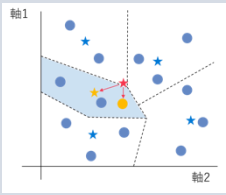
完全

BruteForce

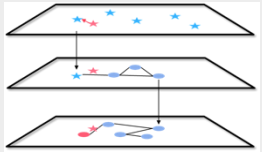
全探索!!
最高精度
検索速度最低

近似:精度を犠牲に高速化

IVFFlat (反転リスト/インデックス検索)
データをクラスタに分けて、クラスタの重心を検索→そのクラスタのデータを全検索する
比較的高精度
低メモリ利用
検索速度はそこそこ



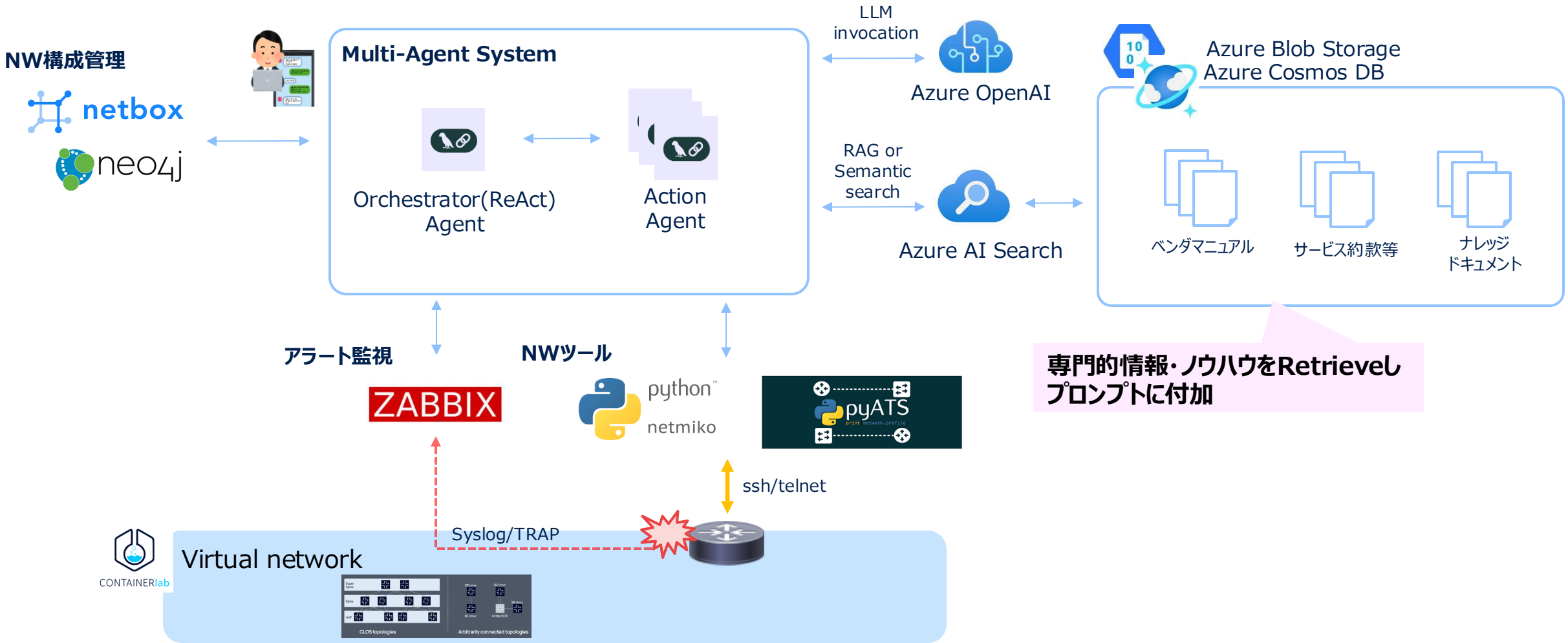
HNSW(階層化ナビ可能な小世界)
データをグラフ(つながりのあるデータ)に分割し、階層化して辿れるようにする
検索速度高速化
メモリ利用量増大
検索精度そこそこ低下



※その他ベンダー独自の近似手法も随時開発されている

（参考）Microsoft AI Co-Innovation Lab PoC概要

- 一部のユースケースを対象としAzureコンポーネントを活用したマルチエージェントシステムを構築する. 特にNWに関わる多種大量の専門的情報をAzure AI Search技術で動的に検索・活用することにより実運用に近い自律的判断を伴うNetwork Automationの一事例を実現する



議論ポイント

- 生成AIをネットワークの業務にどのように活用していますか？
  活用したいユースケースはありますか？
- 生成ＡＩ活用に関して, どのような技術的・運用的ハードルを感じていますか？ など

（参考）その他のLLMエージェントの推論傾向から判明した課題

- Actionの粒度が粗く非効率
  - 結論自体は正解するケースが多いが, NW全台のステータス調査や, すべてのノード組み合わせのpingなど網羅的(虱潰し)な調査Actionを出力することがあり, 実行時間が長くなりがち（３～5分）
  - 商用NWの規模では大きな課題になると思われる

- 否定形の指示（禁止事項）を守らない
  - 同じオペレーションを多重に行う, clientへの直接操作などをプロンプトで禁止しても実行してしまうことがある
  - NOCの保守要件は素直に表現すると否定形の物が多く, これらの要件をプロンプトに反映する際の課題となる.

- LLM自体のネットワーク技術リテラシが貧弱
  - 使っていないインターフェースのdownを障害と推定する, トポロジ上あり得ない経路の通信を推定しようとすることがある
  - RAG等で推論時にプロンプトに挿入できる知識に限度があるため, fine-tuningなどでNW知識に特化したモデルが別途必要？