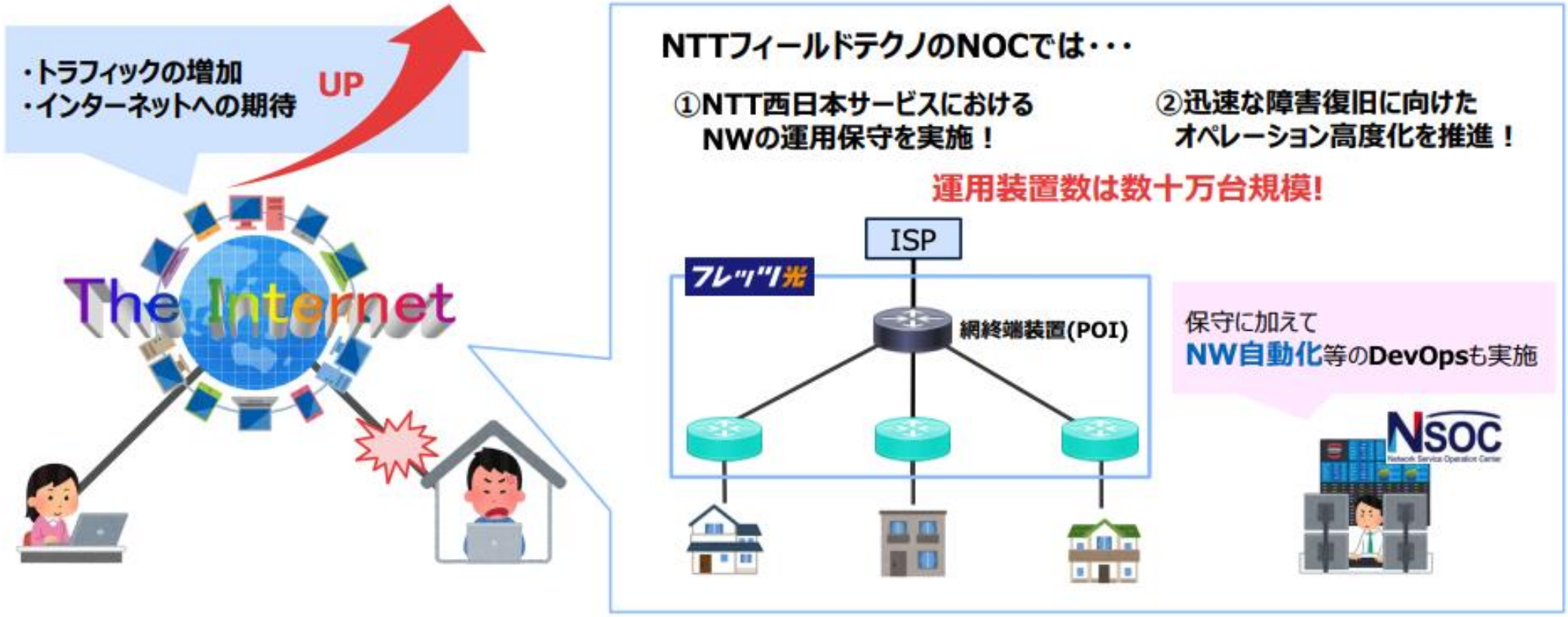
Local LLM×AIエージェントで挑むNOC DevOps
-商用AIOpsのリアル-

株式会社NTTフィールドテクノ　佐藤 亮介
重松 史哉
株式会社エヌ・ティ・ティ エムイー　荒井 新太郎
日本電気株式会社　関谷 郵政

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

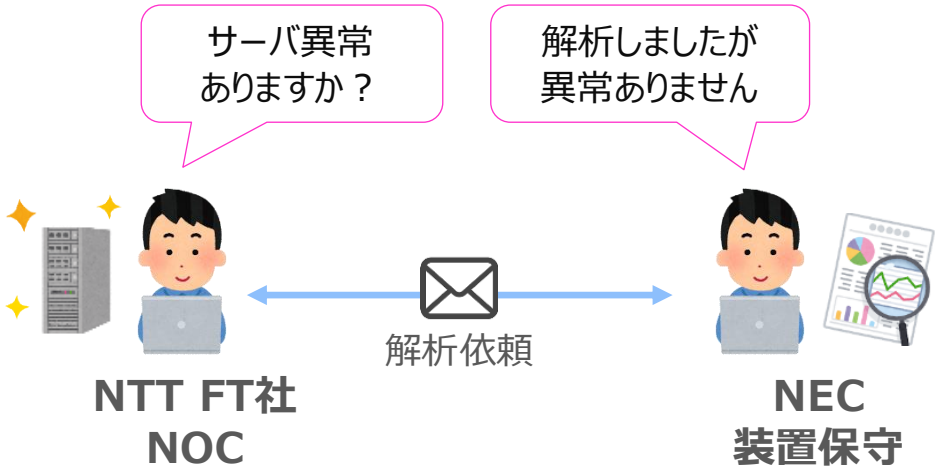


NTTエムイー業務紹介

- NTTエムイーではNTT東日本エリアの装置の保守運用を行っています。
- 人員減耗に加え、速やかな情報共有・設備復旧が求められることから自動化・効率化を推進しています。

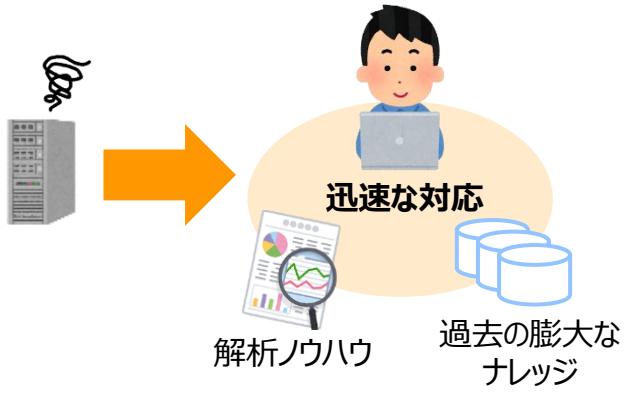
NEC業務紹介

- 装置ベンダとして、音声系サービス装置の保守/維持管理を対応しています。
- 人手不足、ノウハウ継承といった課題に対し、効率的な保守対応を目指しています。

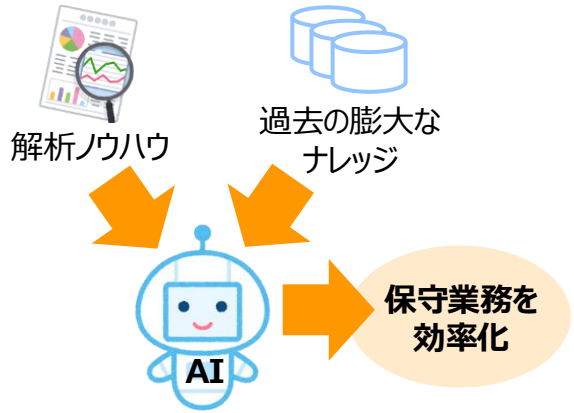


NECの音声サービス装置保守部門では・・・

①装置障害時の迅速な解析を実施



②効率的かつ適切にノウハウ活用した高度な保守対応を推進中



メンバ紹介

・名前:佐藤 亮介
・現担当:NW運用/保守
・JANOG歴:登壇4回目
・趣味：ITガジェット・仮想通貨

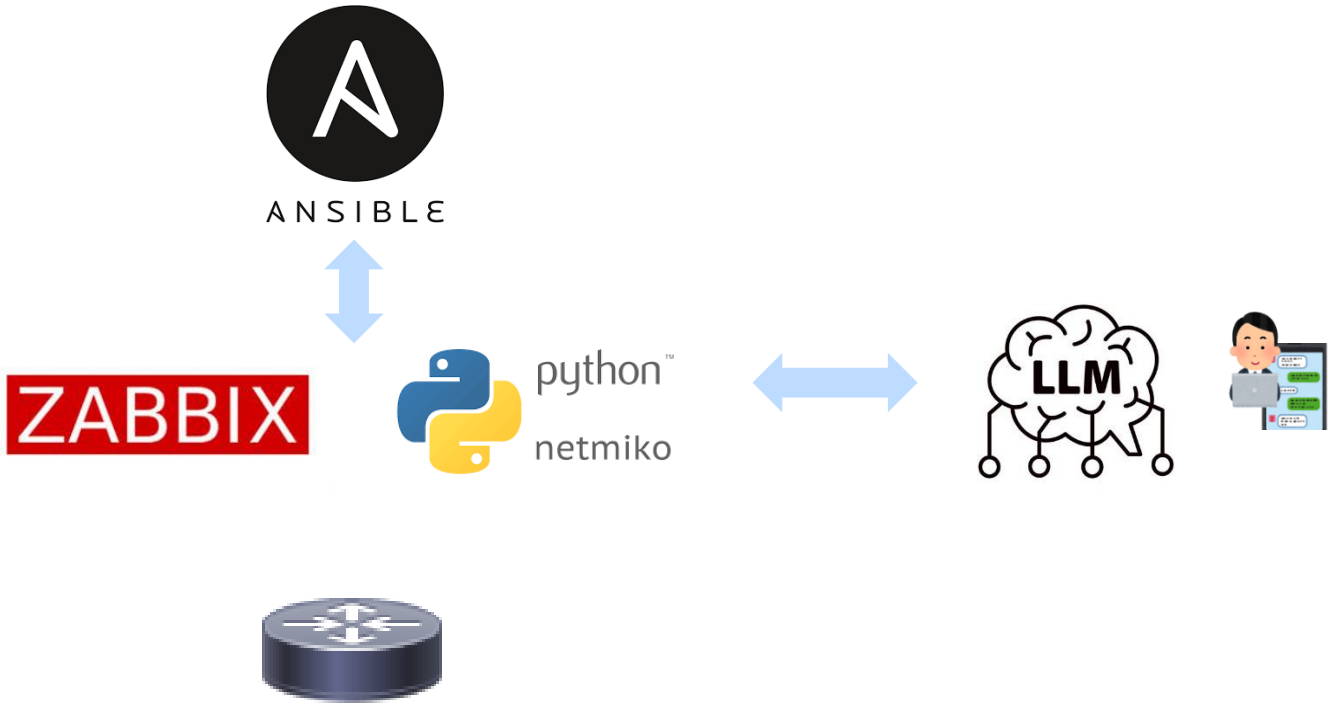
・名前:重松　史哉
・現担当:NW構築
・JANOG歴:初参加
・趣味：釣り・釣り具作り

・名前:荒井 新太郎
・現担当:NW構築/運用
・JANOG歴:登壇２回め
・趣味：局舎巡り、
　　ウォーキング

・名前　　：関谷 郵政
・現担当　：ソフト開発/保守
・JANOG歴：登壇２回め
・趣味　　：野球観戦

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．
- LLM等AI技術に大きな期待をしており，現在の**Network Automation(AIOps)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．

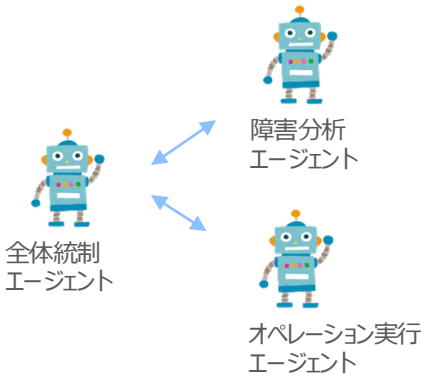


生成AI関連のトレンドと取り組みのターゲット

- マルチエージェント構成や様々なMCPによりAI適用ケースが拡大
  NW領域でのユースケースも多数提案あり
- Local LLM(gpt-oss等)の性能向上
- セキュリティ・運用コストの問題をLocal LLMで抑えながら、**商用環境でのAIOpsの実運用化**に取り組み中

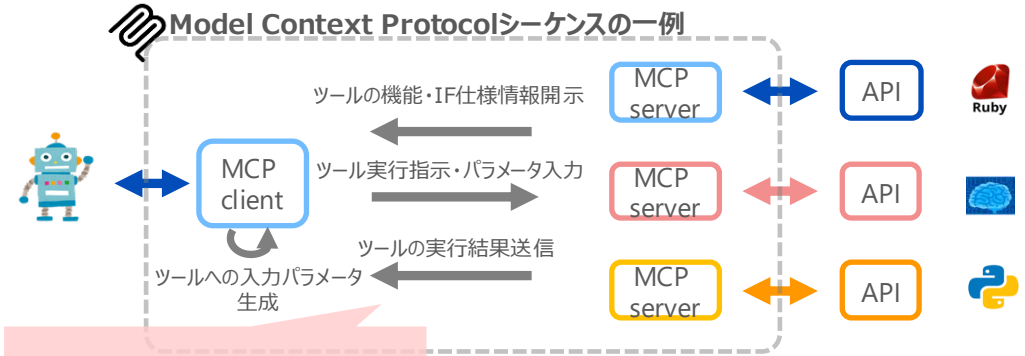
マルチエージェント

タスクに特化したAIエージェントを組み合わせより複雑なタスクを実現可能に



MCP(Model Context Protocol)

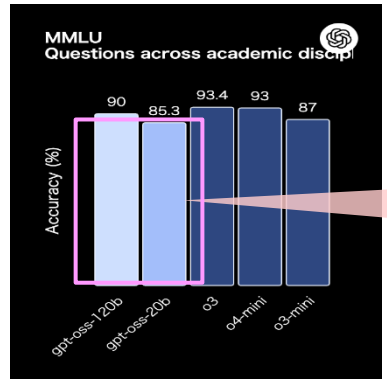
LLMと外部ツールの連携を標準化したプロトコル
AIエージェントの開発が効率化



個別実装が必要だった
煩雑なシーケンスが標準化

Local LLM

オンプレ環境で利用できるLLM 特に2025年8月に公開されたgpt-ossの性能が注目されている



クラウドLLM(o3, o4-mini, o3-mini)に迫る性能

gpt-ossモデルベンチマークの一部(※)

NW自動化への適用ケースが多数提案
（障害分析、syslog解析、config修正、アラーム分析 など）

セキュリティ・クラウドLLMの運用コストによる
商用運用のハードルが低減

(※)openai公式サイトより引用 https://openai.com/ja-JP/index/introducing-gpt-oss/?utm_source=chatgpt.com

（NEC事例）クラウドLLMとLocal LLMの性能・コスト比較結果

■ソースコードの脆弱性を確認するAIエージェントで精度,トークン数,TATを検証

■検証内容

・意図的に脆弱性を盛り込んだソースコード2ファイルに対し、AIにレビューを依頼する。
　ソースコード①：約200行。4件の脆弱性。
　ソースコード②：約70行。6件の脆弱性。
・脆弱性の検出精度と、回答生成に使ったコスト(課金対象のトークン数)、回答時間(TAT)を検証する

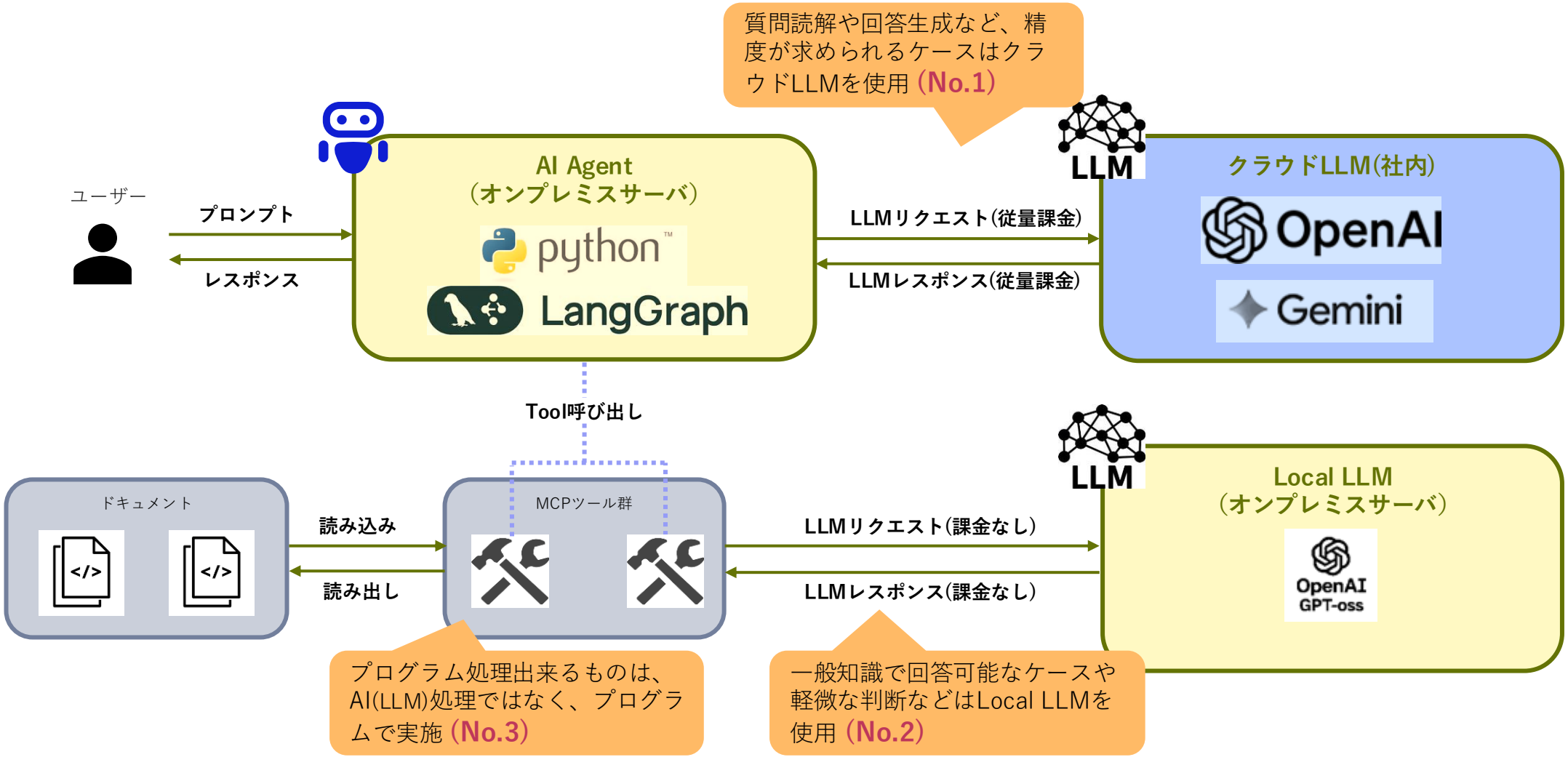
■検証結果 (2025/11時点)

No.	構成	精度	トークン数	TAT	備考
1	クラウドLLM (gpt-4.1)	①3/4検出 ②6/6検出	5416	65秒	・gpt,gemini,claude(社内版) ・**トークン数に比例して課金が発生** ・接続環境(API)準備が必要
2	ハイブリッド (gpt-4.1+gpt-oss)	①2/4検出 ②5/6検出	3564	68秒	・gpt-oss等のlocalLLMを使用 ・トークン数削減は可能だが、精度は少し落ちる ・**更なる精度向上により、単独環境での使用可能**(仮)
3	ハイブリッド＋ プログラム実装 (gpt-4.1+gpt-oss+python)	①4/4検出 ②6/6検出	511	18秒	・内容読解/最終見解　：**クラウドLLMモデル**で実施 ・軽微な判断/ソース解析：**LocalLLMモデル**で実施 ・AI処理不要な部分　：**python/MCP**でプログラム化

⇒No.3のように、処理の流れに適切にAI/プログラムを盛り込む事で効果あり

(NEC事例)クラウドLLMとLocal LLMのハイブリッド構成

■用途に応じてLLMを使い分ける

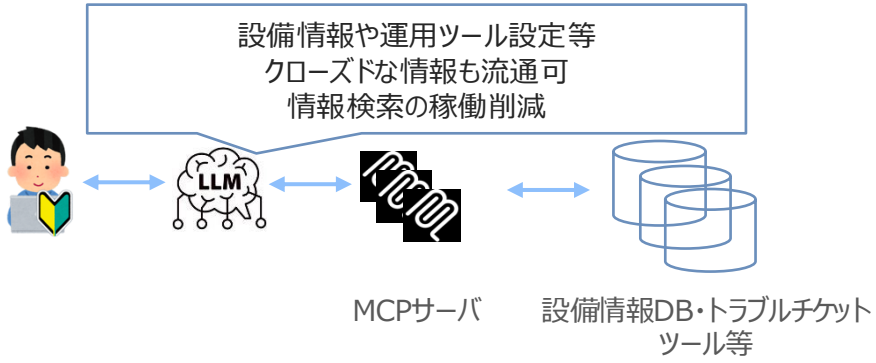


商用環境におけるAIOpsの現状と課題

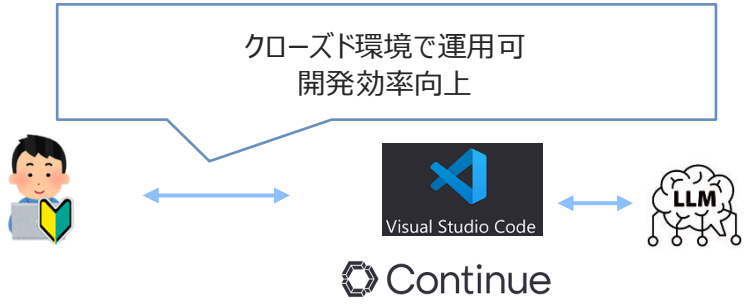
- 一般知識を基に回答可能な共通業務的タスクに関してはLocal LLMを用いてセキュリティ性を担保しつつ一定の運用上の効用が確認できている.
- NWトラブルの分析などNW運用のミッションクリティカルなタスクに関しては, LLMを含む機械学習手法が商用運用に寄与できているケースが少ない.

共通業務におけるユースケース

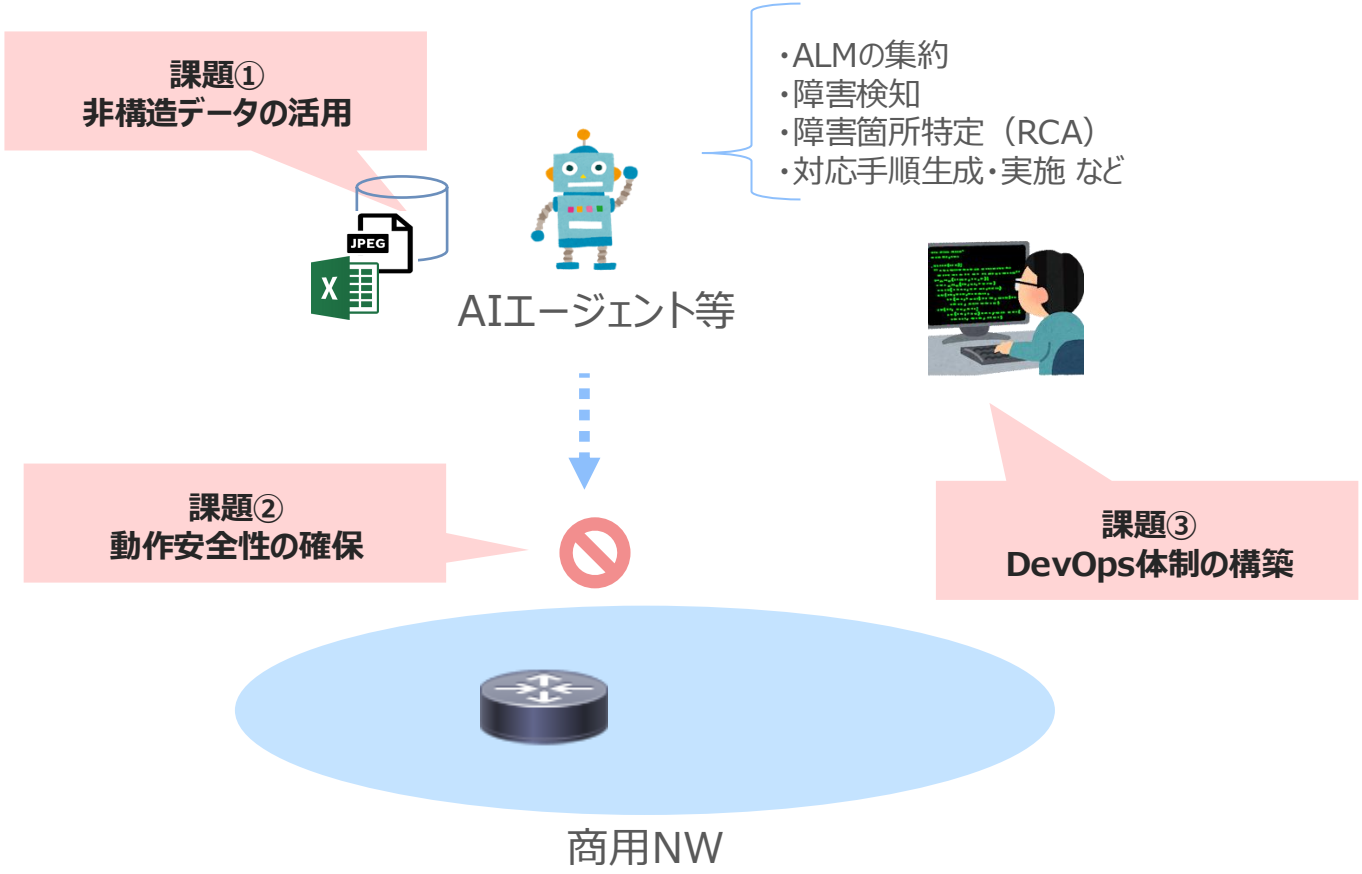
運用関連データ検索



運用ツールのコーディング支援

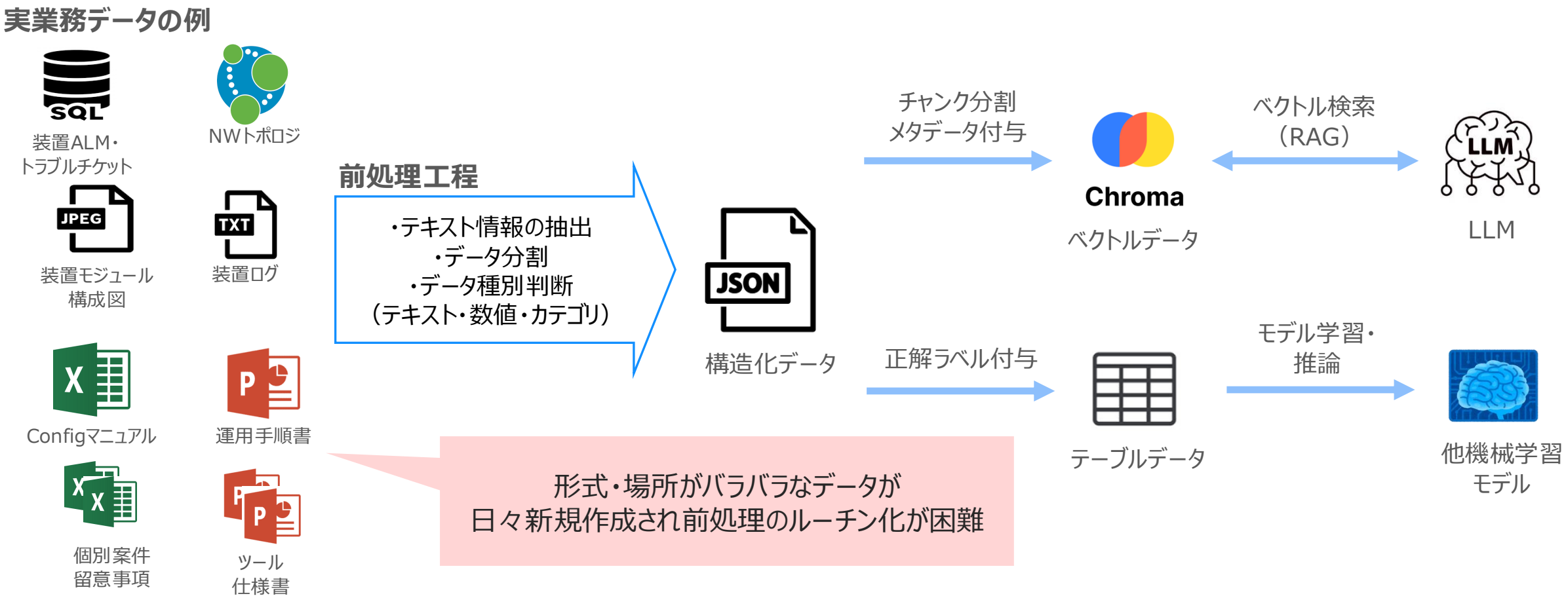


NW運用特有の業務におけるユースケース



課題①：NW運用業務における非構造データの活用

- AI判断・学習のベースとなる実業務データは一部を除き非構造状態で散在しており活用が困難
- 装置ログメッセージなどテキストのカテゴリ（どういった事象に分類されるのか）など前処理自体に機械学習の知見が必要ものはNOCオペレータのルーチン業務に組み込むことも難しい

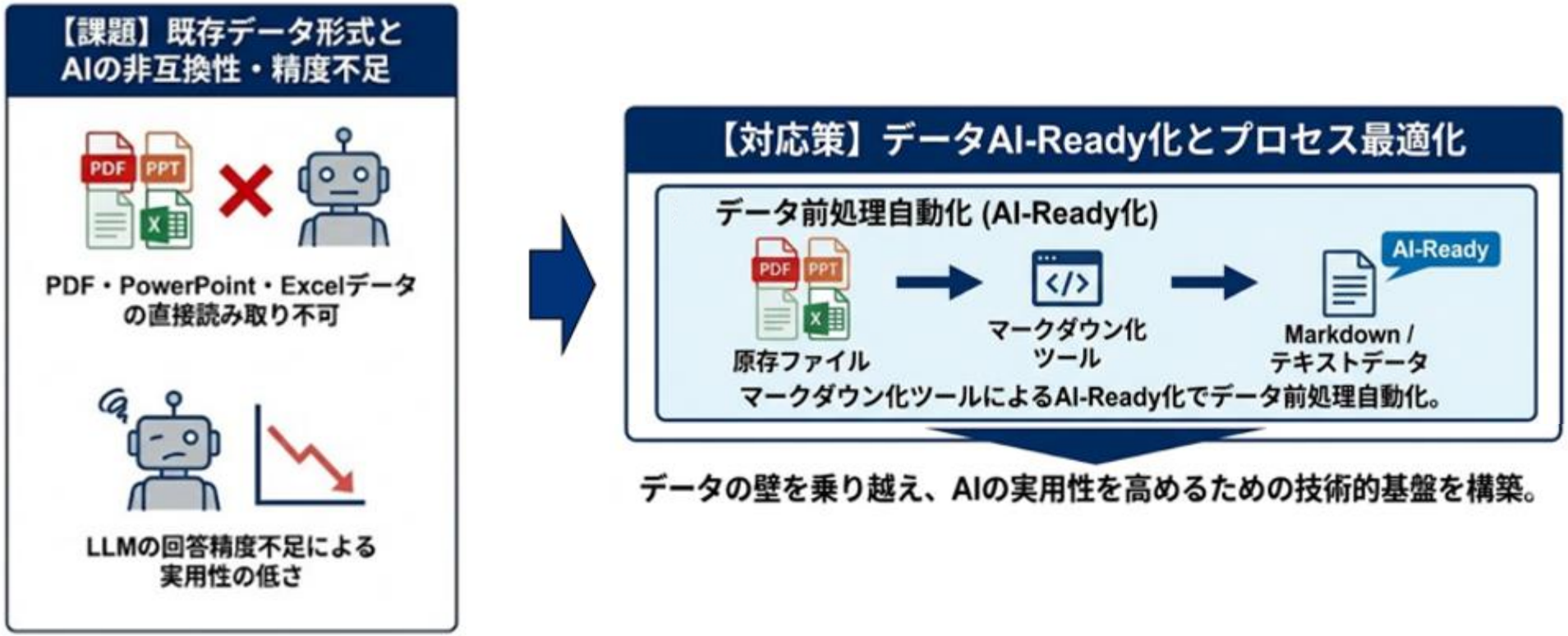


（NEC事例）非構造データのスムーズなAI活用

現状：保守運用ではマニュアル/ナレッジ等のOfficeファイル(PDF、pptx、Excel等)と、
マシン(Linux)上のログ/config等を突き合わせて、障害復旧や問い合わせ回答を実施。
課題：Linuxサーバ上でOfficeファイルの直接読み取りが出来ず、人手介在が残存

対応

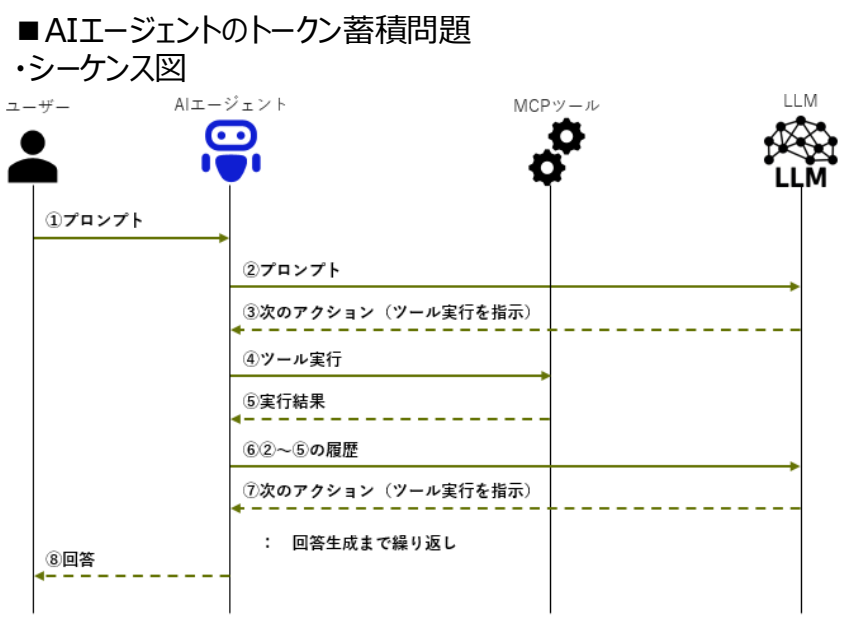
- Officeファイルを、Linux上で読み取り可能な形式(md/txt形式)に変換
- マークダウン化ツールによって、既存ノウハウ(Office)を実運用環境(Linuxサーバ)上で活用可能
- AI/AgentがLinuxサーバ上でマニュアルを基に自律的な解析が可能



（NEC事例）参考：markdown変換ツール(実際の変換例)

・マークダウン化ツール自体はAIコーディングにより実現。
・AIが扱えるデータ増を、個人任せではなく本ツールを共有する事で、AI活用を促進/活性化
・変換はファイル単位で可能。複数ファイル一括変換も可能。

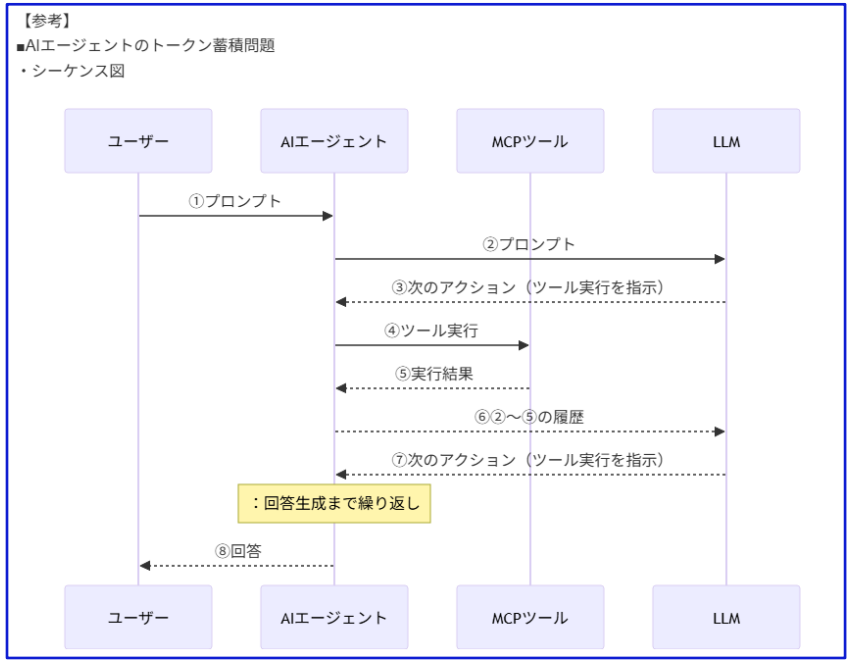
【PowerPoint】



【Markdown】

```
【参考】
■AIエージェントのトークン蓄積問題
・シーケンス図
```mermaid
sequenceDiagram
    participant user as ユーザー
    participant agent as AIエージェント
    participant mcp as MCPツール
    participant llm as LLM
    user->>agent: ①プロンプト
    agent->>llm: ②プロンプト
    llm-->>agent: ③次のアクション（ツール実行を指示）
    agent->>mcp: ④ツール実行
    mcp-->>agent: ⑤実行結果
    agent-->>llm: ⑥②～⑤の履歴
    llm-->>agent: ⑦次のアクション（ツール実行を指示）
    Note over agent: ：回答生成まで繰り返し
    agent-->>user: ⑧回答
```
```

【Markdownのプレビュー】



(ME事例) 故障対応業務における情報検索

- 受付部門からの故障情報(PDF)をもとに様々な文書・ナレッジを組み合わせて対応
- PDF記載の内容を読み取り、情報検索(RAG)結果を一元的に表示する仕組みを作成

ビル：新宿
装置種別：ROUTER
ホスト名：R1234
故障PKG：CONT
故障位置：SLOT1-1
…

●故障情報(PDF)

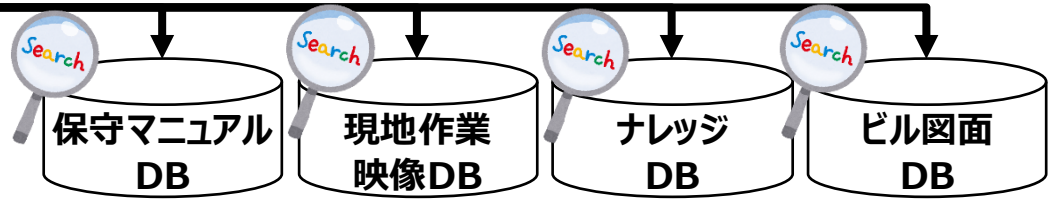
「故障情報」から対応に
必要な情報を教えて！

Dify

Q. PDFには何が書いてある？

・ビル名は？
・装置種別は？
・故障PKGは？
・故障位置は？ …etc.

ナレッジ情報の横断的な検索（RAG）



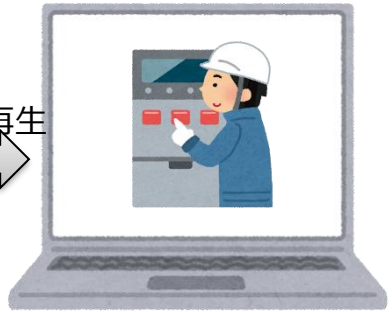
A. 各種DBから検索したデータを一元的に取りまとめて表示

■故障情報
故障発生ビル：新宿
装置種別：ROUTER

■過去ナレッジの検索結果
【1件目】
タイトル：ROUTER CONT盤故障について
・故障情報記載のSLOT1-1はXXという点に注意

■該当保守マニュアルへのリンク
1-1.保守マニュアル
1-2.カード交換マニュアル
【別紙1】正常性確認について

■現地対応映像へのリンク クリックすると動画再生
・ROUTER CONT盤交換



(ME事例) 故障対応業務における情報検索

- 業務遂行のためのデータが点在しており整理されてないケースはある？
  - MEでは様々な場所に資料が点在
    - ファイルサーバの奥深くに埋もれていてベテランに聞かないと分からないケースも

- 1つの業務分野にどれだけの資料数がある？
  - MEのケースだとマニュアル35種類＋ナレッジ160件ほど
  - 今回対応した約200資料で全体の1/4程度をカバー

- 開発組織からの資料（故障対応手順書・リアクションマニュアルなど）のフォーマットがバラバラで個別に対応していることはある？
  - MEでは開発組織と業務組織（NOC）が別組織

(FT事例)configの整合性確認

- 原本がExcelの運用ドキュメントとconfigを照合

デモ実施予定

課題の想定要因

- 様々なスキル背景のオペレータがおり、入退場も頻繁に行われるため運用を統一できない
  - 様々な人が自分流のexcel管理表やパワポ構成図をつくってしまう

- 雑多なデータの**前処理自体を取り扱ったプラクティス・研究が少ない**
  例えばRAG運用プロセスのうち、Embedding / Retrieval / Re-ranking は様々な手法が研究されているが、Ingestion / Preprocessing を取り扱ったものはあまりない
  - 泥臭すぎて研究的評価が難しい、多くの研究や導入に向けたPoCでは適切に前処理されたデータがあることが前提になっている

今後の解決の方向性

- 現場のプラクティスとして、AI活用の前処理を意識したドキュメント作成のルールを整備していくことは前提
  - 列タイトルをしっかり付ける、セル結合を用いないなど

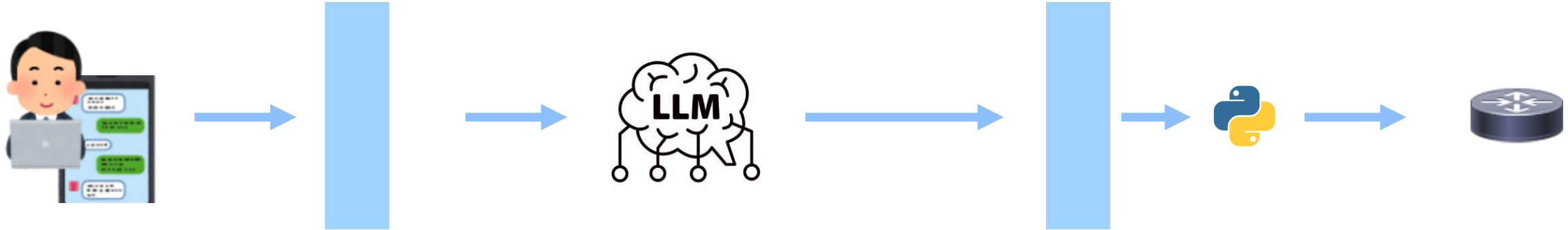
- **前処理自体をAIタスクとして捉え、LLM等を活用する**
  - ネットワーク運用を前提としたLLMによる運用ドキュメント前処理プロセスの自動構成手法
  - ネットワーク運用文書における前処理判断を含むLLM活用手法 など

（FT事例）前処理（ALMのカテゴライズ）をLLMでやってみる

デモ実施予定

課題②：AI出力の安全性の確保

- 特に大規模NW運用業務はミスが大きな社会的影響に発展することがあるため
  ハルシネーションなど**AI出力の不確実性から導入に至らない**ケースが多い
- 市中では入出力前後のプロセスで想定外の出力に対する対処を行うアプローチが試行されている



危険な入力・情報不足の入力を排除（filtering）

対象設備・時間帯など、判断に必要な前提条件が不足している入力を差し戻す など

市中実装例
- NeMo Guardrails（NVIDIA）

不適切な出力の抑制(moderation)

ログや設計書に根拠がない断定を除外、確定情報・暫定情報・不足情報を分離した出力フォーマットを強制 など

市中実装例
- LLM-as-a-Judge
- LlamaGuard（Meta）

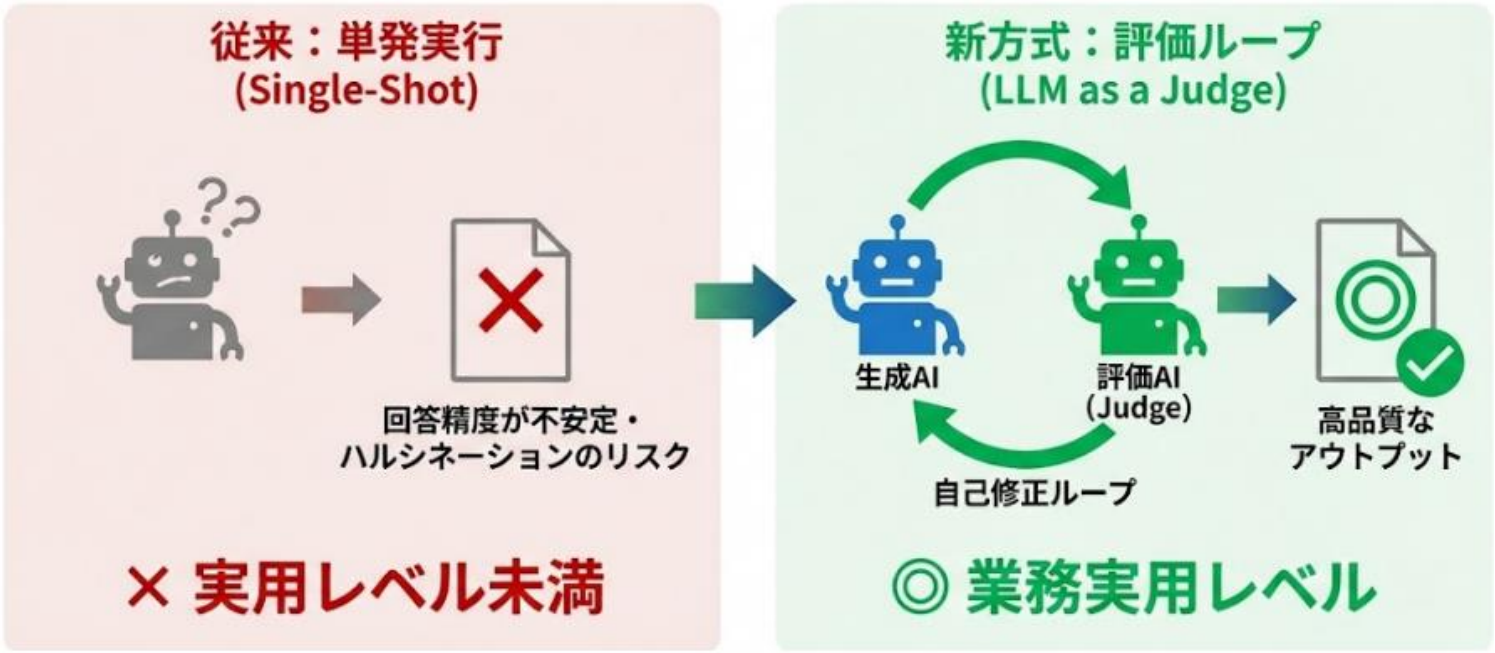
危険な動作実行の中止(gating)

Show系コマンド以外は排除、テスト環境でのdry runの強制 など

市中実装例
- langchain(human-in-the-loop/interrupt)

(NEC事例)LLM as a judge/フロー型APLによる精度向上(1/2)

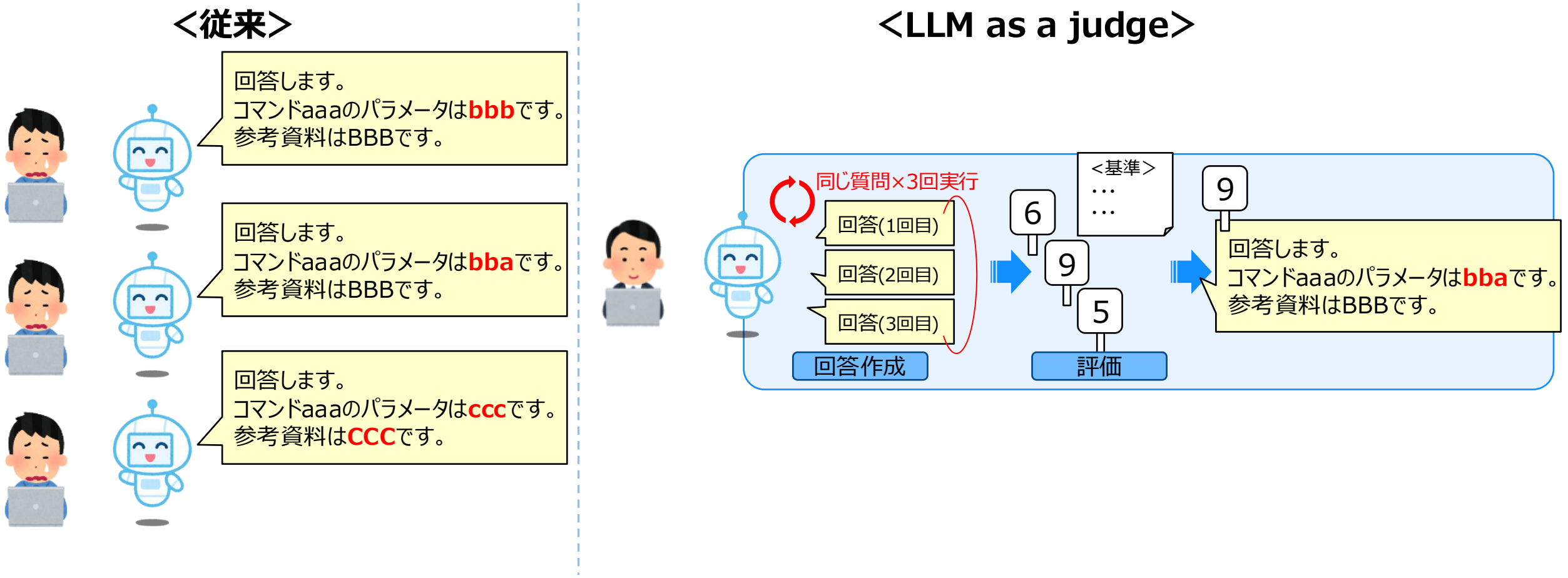
・AIが作った生成物(回答)を、別のAIが評価する仕組み。
・生成AIの回答は「毎回微妙に違う」為、そのばらつきも低減したい。
※従来時点でも、システムプロンプトは十分な品質を担保すると前提。



※ハルシネーションがゼロになるわけではないです。

(NEC事例)LLM as a judge/フロー型APLによる精度向上(2/2)

・NW保守/運用においては、微妙な差分ですらも判断に影響を及ぼす可能性あり、出力の不確実性/不安定性は限りなくゼロとしたい。
⇒LLM as a judgeにより、回答の安定性向上を確認



(NEC事例)参考プロンプト：LLM as a judge

- プロンプト例
- ユーザ質問＋参照ドキュメント(検索結果)＋(前段の)AIの回答を基に、評価基準で判定

<プロンプト>
あなたは、AIチャットボットの回答を厳格に審査するスペシャリストです。
[入力情報]に基づき、AIの回答が適切であるかを評価してください。

入力情報
- ユーザーの質問: {user_question}
- 参照ドキュメント(検索結果): {retrieved_documents}
- AIの回答: {ai_answer}

評価基準
1. 直接的な回答
   - 質問に対して、結論から端的に答えているか？
   - 質問の意図を正しく理解し、的外れな回答になっていないか？

2. 内容の充足性
   - 質問に対する答えとして、情報が不足していないか？
   - 追加の質問をしなくても済むよう、必要な補足情報が含まれているか？

3. 事実への忠実性
   -「参照ドキュメント」に記載されている情報のみに基づいているか？
   - ドキュメントにない情報を勝手に捏造していないか？

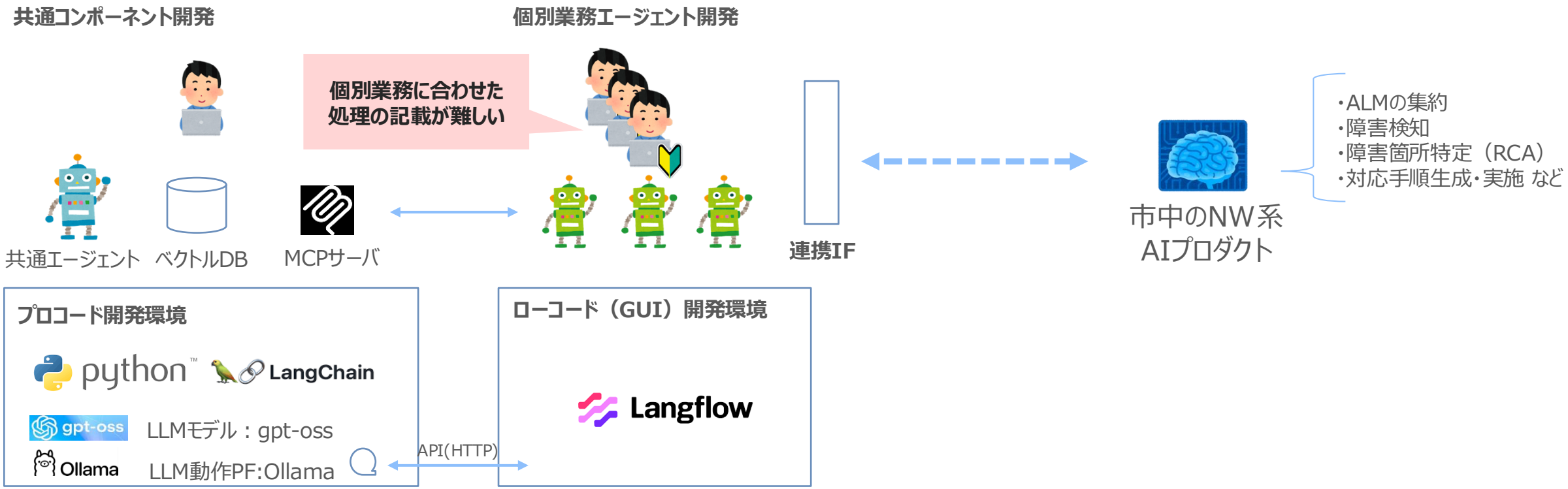
(FT事例) ルータ操作に対する誤り抑止

デモ実施予定

課題③：エージェントDevOps体制の構築

- LLMエージェントのプロコード(スクラッチ)開発を行える者は人数が限られ, ローコード開発ツールを用いても RAG等AI関連の知見を多く要求されるため**業務に耐えられる規模のエージェントを作成することが難しい**

NOCにおけるエージェントDevOps環境の例

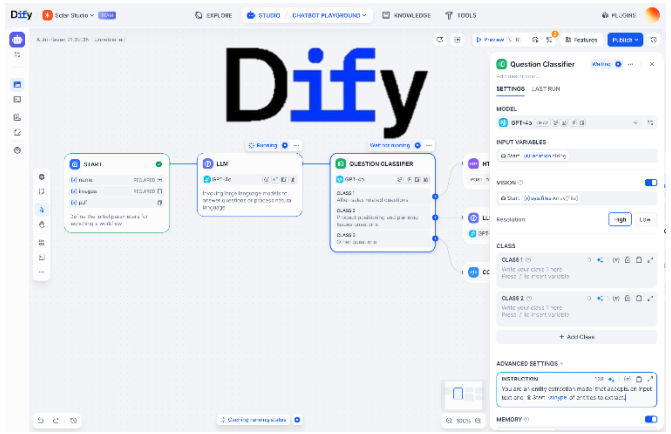


(ME事例)ローコードツールを用いたDevOps体制

- ローコード開発ツール「Dify」を用いてAIアプリのワークフローを実務担当者レベルで開発
- 担当内のスキル濃淡に応じて作成分担し、対応範囲を拡大中

ビル：新宿
装置種別：ROUTER
ホスト名：R1234
故障PKG：CONT
故障位置：SLOT1-1
…

●故障情報(PDF)



●Difyワークフロー

A. 各種DBから検索したデータを一元的に取りまとめて表示

■故障情報
故障発生ビル：新宿
装置種別：ROUTER

■過去ナレッジの検索結果
【1件目】
タイトル：ROUTER CONT盤故障について
・故障情報記載のSLOT1-1はXXという点に注意

■該当保守マニュアルへのリンク
1-1.保守マニュアル
1-2.カード交換マニュアル
【別紙1】正常性確認について

■現地対応映像へのリンク
・ROUTER CONT盤交換

プログラム有スキル者・リーダー

- ナレッジDBなどの整備
- LLMが苦手な領域をPython処理するための機能作成
- 作業者のフィードバックを踏まえた改良

…etc.

輪番者・作業者

- LLMへの指示プロンプトの作成
- 出力内容が業務運用に適しているかの評価
- 実オペレーションを通したフィードバック

…etc

（ME事例）ローコードツールを用いたDevOps体制

- AIアプリの作成はトップダウン/ボトムアップどちらが多い？
  - MEの場合は現場組織によるボトムアップになりがち

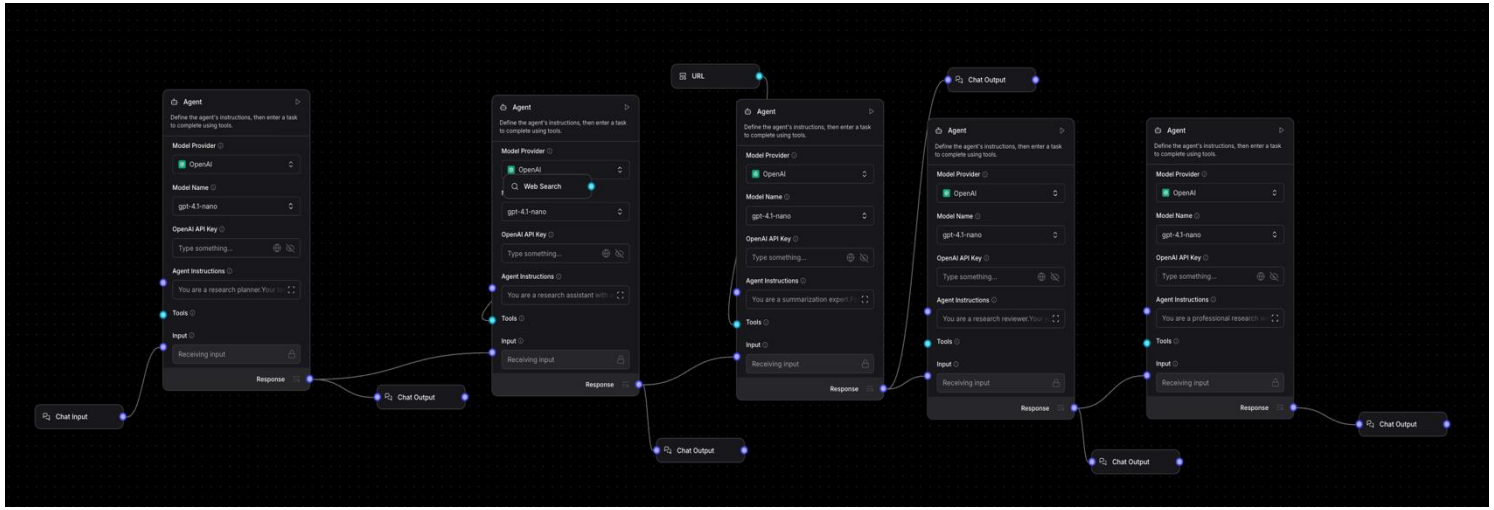
- どれだけコスト・時間をかけられているか、一部のメンバに稼働が寄っていないか？
  - MEは残念ながら寄っちゃってます
    - 実作業者・リーダーで8時間×3人ほどでデータ整備
    - ワークフローは改良を繰り返しながら1ヶ月ほど

- 組織間をつなぐ故障票（トラブルチケット）の記入形式の未統一も問題でした

(FT事例)NOC版 Deep Researchの実装

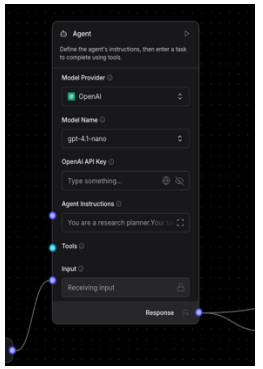
- NOCの多様なデータの調査には、RAGや少数ノードのマルチエージェントを超えて網羅的な調査が行えることで知られているDeep Researchアーキテクチャを活用したい
- ローコードのみでの実装はスキルやGUI機能・可読性の点で難があるため、プロコード実装したものを抽象化して活用

Deep Research実装の例（最小構成）



クエリ分解　情報取得　要約　レビュー　レポート

抽象化



カスタムコンポーネント

(FT事例)NOC版 Deep Researchの実装

デモ実施予定

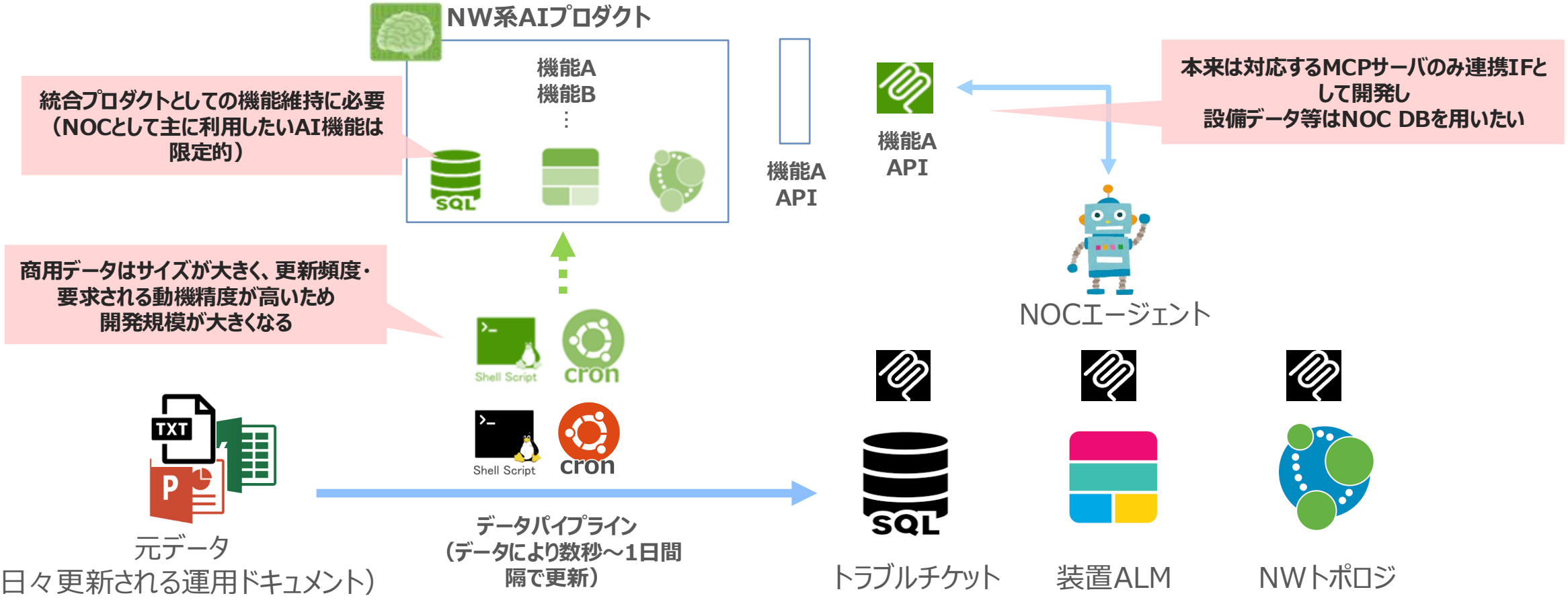
まとめ

- 実際のネットワーク運用保守業務における Local LLM を用いた AI エージェント環境の運用事例を基に, AIOps の現状の到達度及び課題に関して紹介させていただいた
- 現場のプラクティスや開発・実装、基礎研究など複合的なアプローチが必要と考えておりますので、ぜひ議論させていただきたい.

以下参考（議論材料）スライド

想定される要因（連携IF開発について）

- 市中のNW系AIはOpS機能統合プロダクトとしての実装で, 設備データやALM情報をそれぞれの独自規格で内包する形が多い
- このためNOCのDB群に加えて**AIプロダクトにも常時データを流通させるパイプラインの拡張**が必要となる.
- 本来はSSoT(Single Source of Truth)の構成で一箇所の共通データベースから機能実行の都度データをreadする形が望ましい



解決の方向性

- ネットワーク領域特有の判断パターンに関するエージェントのデザインパターンの拡充
  - ローコード開発者はより抽象的なコンポーネントを利用できる

- 統合プロダクトの必要な機能を**Ephemeral Tool（状態を持たず関数的に呼び出せるツール）として運用**できるMCPサーバ設計パターンの検討
  - 機能実行ごとにコンテナ立ち上げ、初期設定、データ投入、タスクの実行、現状回復（コンテナ消去）まで一連の手順を実施
  - 常時データ同期不要としてパイプライン拡張の負荷を避けられる
  - Ansible等でインスタントにサービス立ち上げ・廃棄を行うプラクティスは市中に多数存在するが個別のサービス・機能ごとに個別実装が必要
    MCPシーケンスの中でEphemeral Tool的な振る舞いの手順を判断できれば大幅な開発コスト減になる