

Mosaic AI Gateway でコーディングエージェントを配るための運用Tips

uma-chan

2026-01-27

1. はじめに

1.1. 自己紹介

馬渡 大樹 (Mawatari Daiki) / uma-chan

株式会社GENDA

IT戦略部 データチーム

データエンジニア / MLOpsエンジニア

Databricksを使った開発環境を改善するのが好き

1.2. 今日話すこと

Mosaic AI Gatewayを活用してAIコーディングエージェントを組織内に展開しやすくした構成を紹介

- データ基盤の費用に集約しながらAIコーディングエージェントを配る
- Databricksの認証設定だけでAIコーディングエージェントを渡す仕組み

2. AIコーディングエージェントの普及にまつわる課題

2.1. AIコーディングエージェントを配りたいけど

- 一時的に作業してくれる方にAIコーディング環境を用意したいがすぐに提供できない場合がある
- 技術部門以外のメンバーに使ってもらいたい場合があるが、技術部門と同等の環境を直ちに用意するのが難しい場合がある
 - 予算面・セキュリティ考慮・習熟度の差異などによるもの
- 従量課金での利用の場合は比較的ハードルが低いものの利用量の監視問題が発生する

このあたりの課題感を今回解決してみたい

3. 基礎事項の確認

3.1. Mosaic AI Gatewayとは

Mosaic AI Gateway は、組織内の生成AI モデルとエージェントの使用と管理を効率化するように設計されています。これは、ガバナンス、モニタリング、および本番運用の準備をモデルサービスエンドポイントにもたらす一元化されたサービスです。また、AI トラフィックを実行、保護、管理して、組織の AI 導入を民主化し、加速することもできます。

すべてのデータは、Unity Catalog の Delta テーブルに記録されます。

<https://docs.databricks.com/aws/ja/ai-gateway/>

要点

- ・ 従量課金のLLM API Endpointが提供されている
- ・ systemテーブルでコストのモニタリングができる

以降ではLLM API Endpointやsystemテーブルを個別で指すようにします

3.2. 対象ユーザーについて（構成上の制約）

適している



インターン・外部協力者



技術部門以外（データアナリスト等）



Databricks外の予算がない人



今すぐ使いたい人

適していない



ヘビーユーザー（大量利用）



リアルタイム遮断が必要な場合

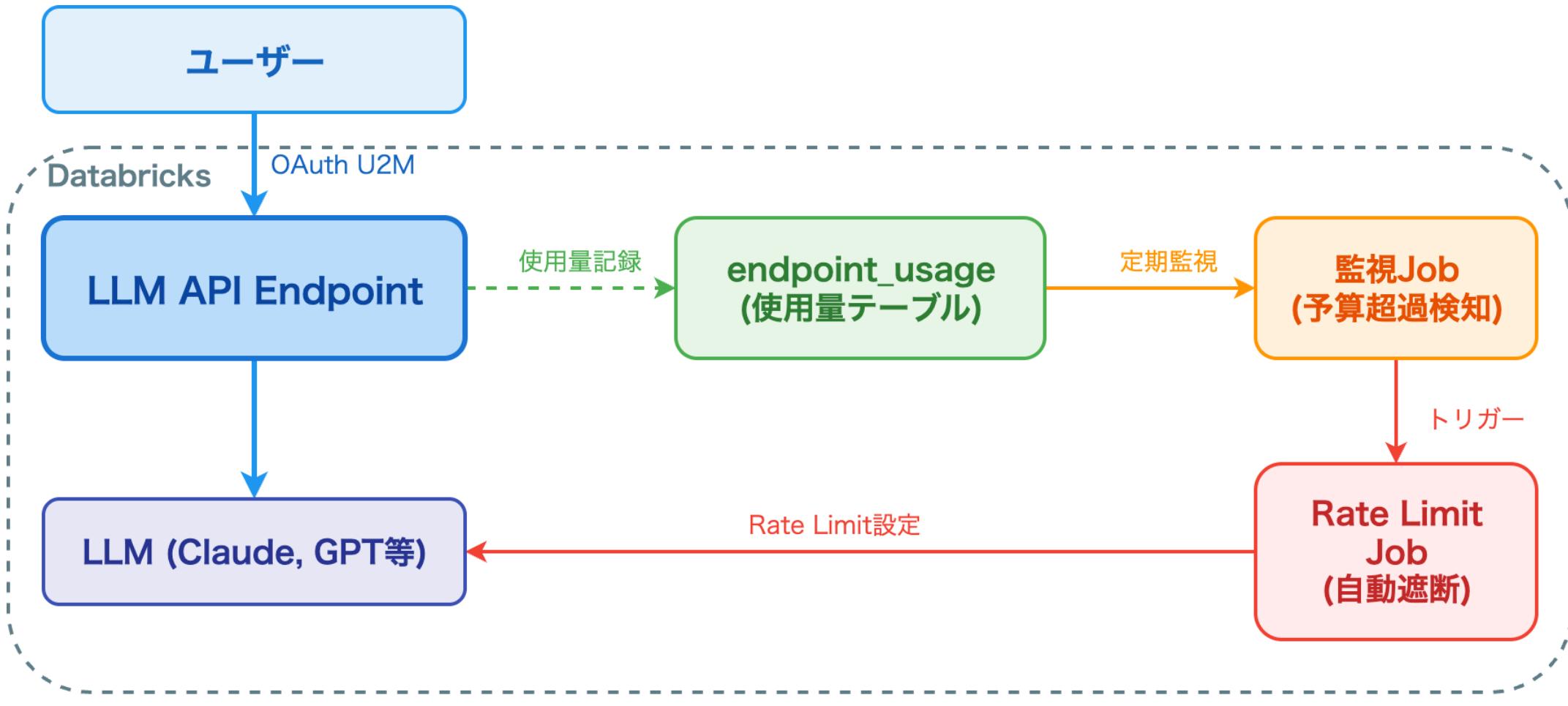
ポイント

指定した間隔で予算超過を検知
例えば15分間隔なら
ライトユーザーが使い切ることはほぼない

対象ユーザー

4. Databricks完結のアーキテクチャ紹介

4.1. Databricks側アーキテクチャの概要



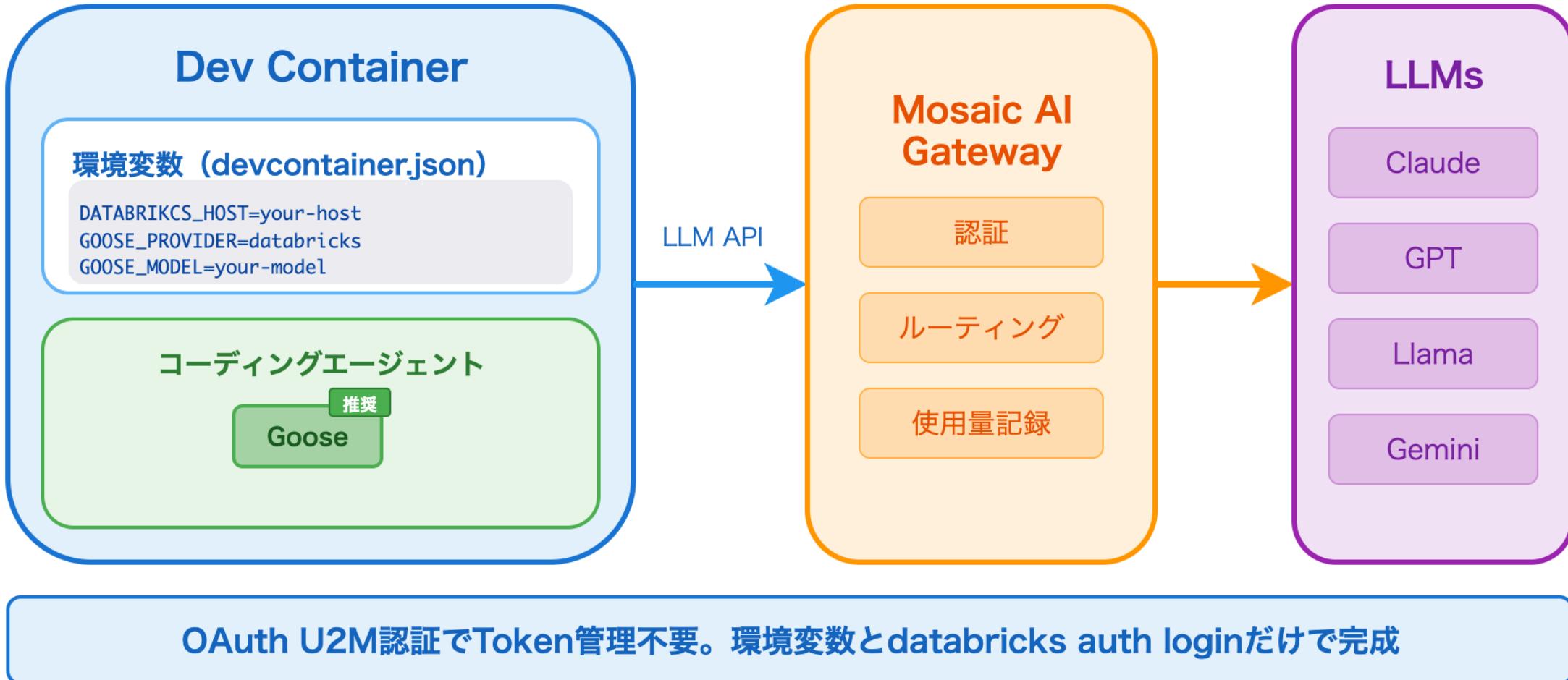
Databricks側アーキテクチャ

4.2. Databricks側アーキテクチャの特徴

ローカル環境以外がDatabricksで完結しているのが最大の特徴

- ローカルでの認証はOAuth U2M認証（ユーザーマシン間認証）だけで済むので難しくない
 - Personal Access Tokenによる認証の辛い面をOAuth U2Mが解決してくれる
- 監視ジョブはDatabricks Jobなので慣れた方法で設定・管理できる
 - 使用量に関するデータendpoint_usageテーブルを監視して柔軟にPythonでRate Limit APIによる遮断処理を記述できる
 - Databricks Jobとしてジョブが存在するので基盤管理者が見失いにくい
- 複数のLLM API Endpointが提供されていて簡単に利用切り替えができる
 - コストや性能に応じて手元で調整しやすい
- コストを対象ワークスペースに集約できる

4.3. ローカル側アーキテクチャの概要



ローカル側アーキテクチャ

4.4. Gooseとは

Block社が開発するAIコーディングエージェント

複数のLLM APIに対応していてDatabricksをネイティブサポートしている（OAuth U2M認証OK）

<https://block.github.io/goose/docs/getting-started/providers/>

Gooseでできること

- コマンド実行（つまりjupyter-databricks-kernelでノートブック実行ができる）
- ノートブック読み取り・編集

ノートブックの編集や実行を任せられるので利用体験はなかなか悪くない

4.5. jupyter-databricks-kernelとは

- <https://github.com/i9wa4/jupyter-databricks-kernel>
 - 拙作。完全リモート実行を可能にするJupyter向けカーネル
- Databricks Notebook 開発環境を AI-Ready にする (紹介記事)
 - 今回メリットが活かせてGooseとうまく接続できた

4.6. ローカル側アーキテクチャの特徴

- Gooseが意外と有用
 - 個人的に普段からClaude CodeやCodex CLIを使い倒しているのでその目線では物足りない部分は正直ありますがノートブックにまつわる操作を一任できるので優秀
- OAuth U2M認証で簡単に認証できることやjupyter-databricks-kernelでノートブック実行はローカル完結できることから提供する開発環境をDev Container化しやすい
 - Dev ContainerをセットアップするタイミングでOAuth U2M認証させるスクリプトを書いておけばほぼ自動で開発環境構築が終わってしまう
- AIコーディングエージェント付きのDatabricksへのクエリ実行環境として配ることもできる
 - MCPサーバー利用もしくはAgent Skills利用で対話的にクエリを作成しやすくなる

5. まとめ

5.1. Mosaic AI Gatewayを活用してAIコーディングエージェントを配ろう

- ローカルでノートブックを書きたい人やクエリを対話的に実行してみたい方を念頭に置いて一連の構成を紹介した
- 組織の課題解決にうまくハマると思った場合
 - Dev Container化を真面目に頑張って利用手順書を整備することで理論上は誰でも使える環境として提供できる（実際はDev Containerもちょっと難しさはあります）

5.2. Thank you

Thank you!