RubyWorld Conference 2025

島根県立産業交流会館「くにびきメッセ」

2025.11.07 (金)

RubyでLLMアプリケーション開発を支える基礎技術

Ruby x LLM Ecosystem

伊藤 浩一 / ESM, Inc.

@koic

- OSSプログラマー
- RuboCopコアチームコミッター
- MCP steering groupメンバー [NEW]
- 株式会社永和システムマネジメント (ESM, Inc.)
  エンジニアリングマネージャー /
  ディスティングイッシュド・エンジニア

永和システムマネジメント アジャイル事業部

永和システムマネジメントアジャイル事業部は、Rubyとアジャイルに関連する技術力をさらに先鋭化させ、業界にとって必要不可欠な存在となるため、より専門性を高めた組織です。

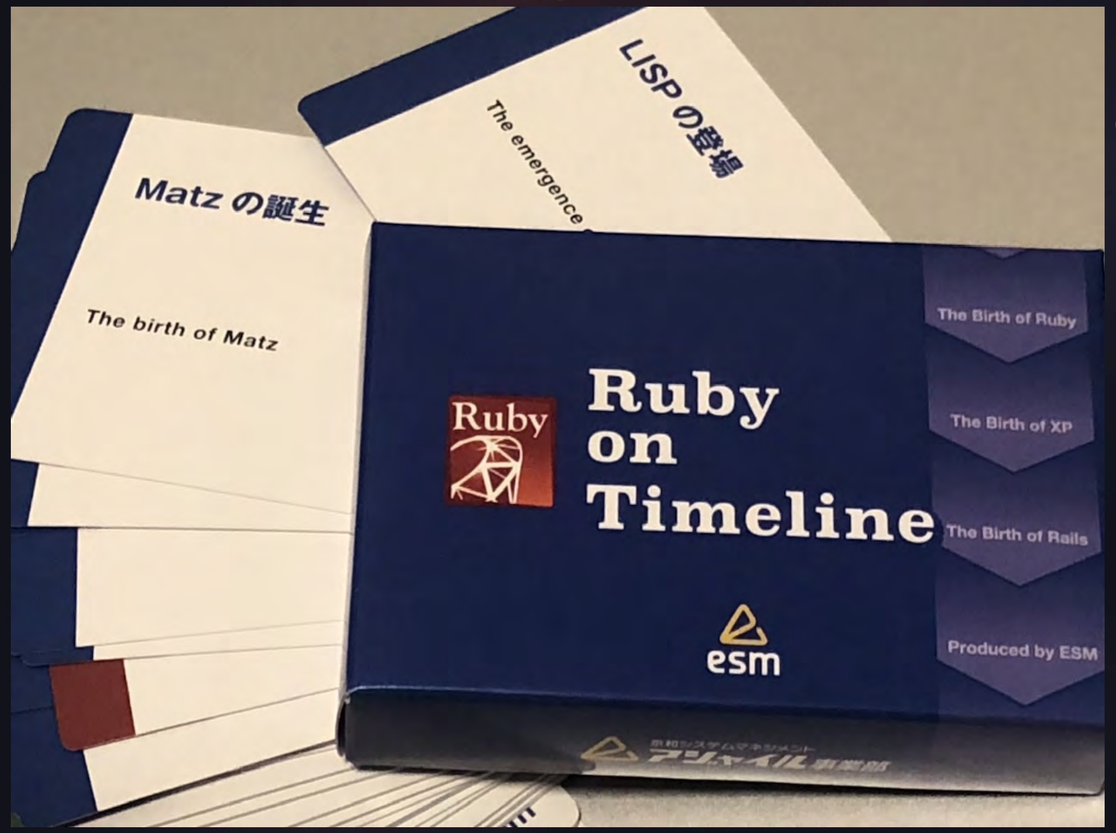
アジャイル事業部について

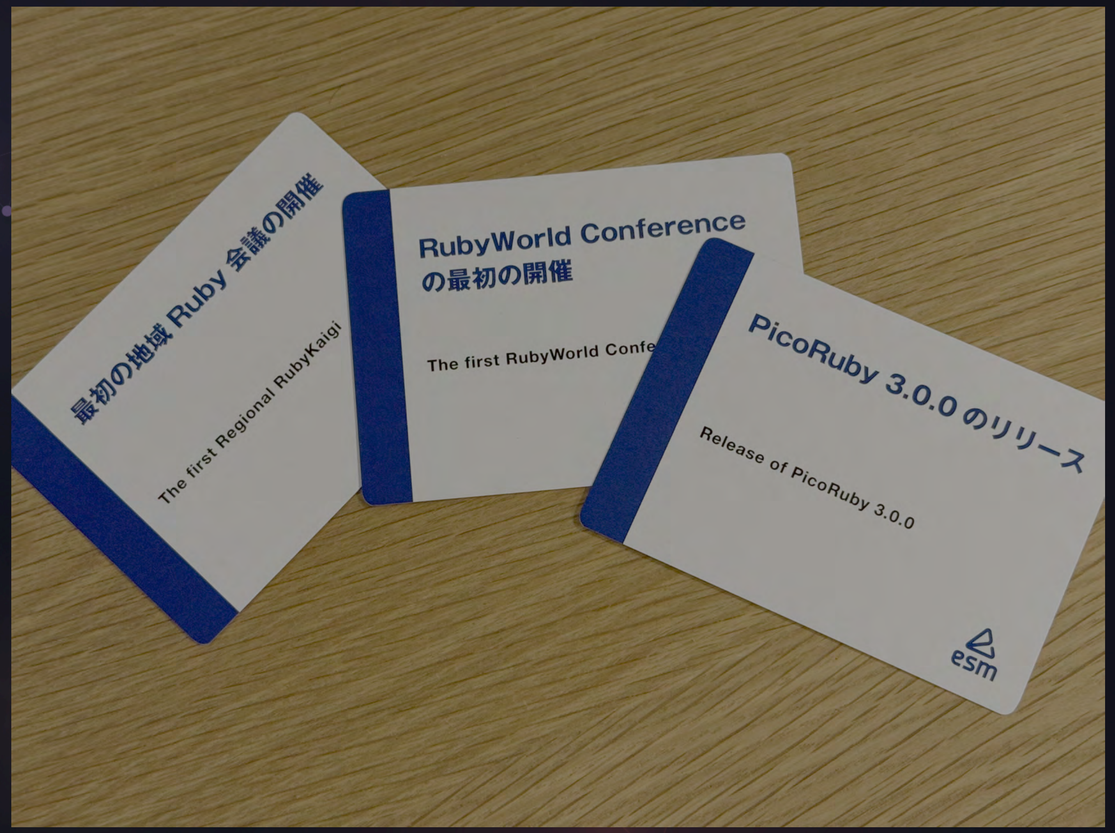
アジャイル事業部に入りたい

Rubyアジャイル受託開発

アジャイル事業部のRubyアジャイル受託開発は、アジャイル開発を10年以上続けて培った、決して手法や方法論ではまとめきれない、実践知や価値観、それを届ける人で構成されています。

プラチナスポンサーノベルティ





RubyWorld Conference 2025特別拡張パックを1Fブースにて配布中

採用と働き方の多様化

・人生の状況にあわせた
ライフスタイルを尊重する

・関東圏、福井、滋賀、京都
和歌山、島根、宮崎からの
リモートワーク

Support OSS community

大規模言語モデル（LLM）研究部

- 変化の速いLLM周辺技術についての話題
- 予算のついた部活動
- 永和社内Slackの#llm-clubチャンネル
- 絶賛部員募集中
- 入社するだけで、もれなく入部できます

https://magazine.rubyist.net/articles/0042/0042-FollowUpKirishimaRuby.html

今日の話

RubyWorld Conference とは

RubyWorld Conferenceは、プログラミング言語「Ruby」の国内最大のビジネスカンファレンスです。

Rubyが、多様な現実世界にどのように適合し浸透していくのか、そのような普及過程と成長を考える機会となることを期待すると共に、Rubyのさらなる普及・発展とビジネス利用の促進を目指します。先進的な利用事例や最新の技術動向、開発者教育の状況などの情報を発信することで、「Rubyのエコシステム(生態系)」を知っていただくことができる場として開催します。

https://2025.rubyworld-conf.org/ja/about/

👉

13:55
–
14:10

15分講演-6

RubyでLLMアプリケーション開発を支える基礎技術

伊藤 浩一

株式会社永和システムマネジメント/エンジニアリングマネージャー

本資料提出後もChatGPT Apps SDK, Agent Skills, ChatGPT Atlasなど激流

目次

1. LLM: 大規模言語モデル
2. RAG: 検索拡張生成
3. MCP: モデルコンテキストプロトコル
4. Ruby x LLMの可能性

Ruby 3.4.7, Langchain.rb 0.19.5, MCP Ruby SDK 0.4.0

1

LLM

Large Language Model

LLMというゲームチェンジャー

- 機械学習といえばPythonと数学という世界から、LLMへの自然言語のプロンプティングで誰でもAIが実用可能になった
- テキスト生成のLLM（大規模言語モデル）と画像、音声、動画生成に使われるDiffusion Model（拡散モデル）が現代AIの花形

LLMはコンテキストに対し次トークンを自己回帰的に予測し続ける確率的生成モデル

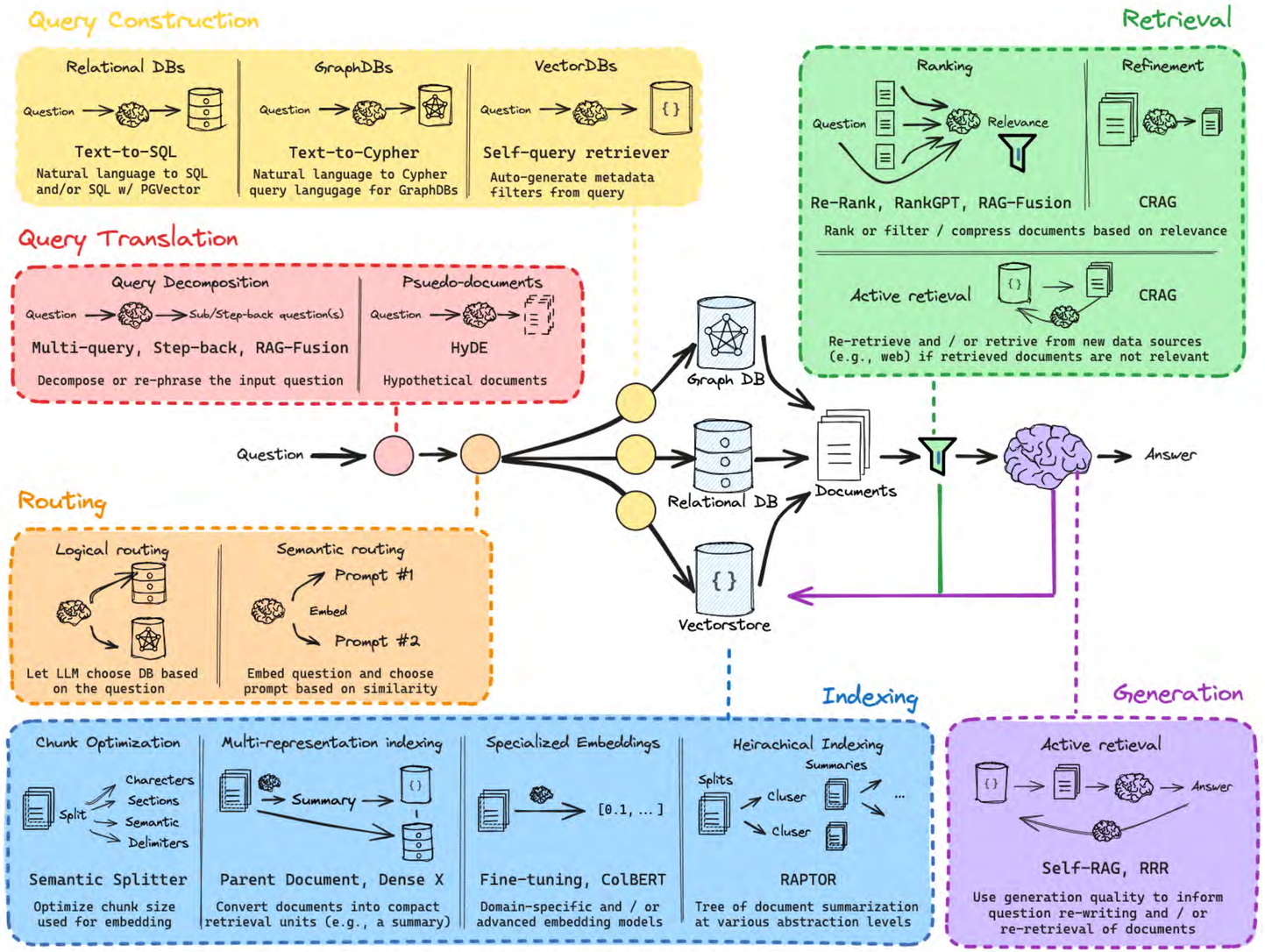
LLMとの様々な関わり(楽しみ)方

- 壁打ちのチャットやAI議事録等ビジネス活用
- Claude CodeやCursor等でAgentic/Vibe Coding
- アプリケーションへのLLM組み込み
- LLMへのAPI、RAG、MCP等ツール開発
- LLM自体やOllamaなどLLM実行基盤開発
- arXivの論文への評論や執筆する猛者も

Rubyの普遍性

- クライアントとサーバーでコード共有可能なUniversal JavaScriptの世界
- Pythonで機械学習を書くなら、Webまで含めてPythonという10年代のUSの潮流 (風聞)
- LLM周辺の実行環境をRubyで書ければ、Ruby/RailsのままLLMを組み込める (本題)

LLMアプリケーション開発



https://github.com/langchain-ai/rag-from-scratch

世の中ではLLMアプリ開発/研究が進んでおり、Python SDKに実装は限らない(Rubyの出番！)

LLMアプリ例 (Gumroad)

```ruby
def ask_ai(prompt)
  OpenAI::Client.new.chat(
    parameters: {
      messages: [{ role: "user", content: prompt }],
      model: "gpt-4o-mini",
      temperature: 0.0,
      max_tokens: 10
    }
  )
end
```

gumroad / app / models / product_refund_policy.rb

```ruby
def determine_max_refund_period_in_days
  return 0 if title.match?(/no refunds|final sale/i)

  begin
    response = ask_ai(max_refund_period_in_days_prompt)
    days = Integer(response.dig("choices", 0, "message", "content")) rescue response.dig("choices", 0, "message", "content")

    # only values from ALLOWED_REFUND_PERIODS_IN_DAYS or default to 30
    ProductRefundPolicy::ALLOWED_REFUND_PERIODS_IN_DAYS.key?(days)


    logger.debug("Unknown refund period for policy #{id}: #{days}")
    ProductRefundPolicy::DEFAULT_REFUND_PERIOD_IN_DAYS

  e
    logger.debug("Error determining max refund period for policy #{id}: #{e.message}")
    ProductRefundPolicy::DEFAULT_REFUND_PERIOD_IN_DAYS


def max_refund_period_in_days_prompt
  prompt = <<~PROMPT
    You are an expert content reviewer that responds in numbers only.
    Your role is to determine the maximum number of days allowed for a refund policy based on the refund policy title.
    If the refund policy or fine print has words like "no refunds", "refunds not allowed", "no returns", "returns not allowed

    The allowed number of days are 0 (no refunds allowed), 7, 14, 30, or 183 (6 months). Use the number that most closely mat

    Example 1: If the title is "30-day money back guarantee", return 30.
    Example 2: If from the fine print it clearly states that there are no refunds, return 0.
    Example 3: If the analysis determines that it is a 3-day refund policy, return 3.
    Example 4: If the analysis determines that it is a 2-month refund policy, return 30.
    Example 5: If the analysis determines that it is a 1-year refund policy, return 183.
```

Thanks Ginza.rb

https://github.com/antiwork/gumroad/blob/production-f020aa7a68c1/2025-07-05-21-18-17/app/models/product_refund_policy.rb

Ruby OpenAI gem

- OpenAI APIをRubyから使うgem
- アプリ側からLLMを呼び出す基本

```ruby
OpenAI::Client.new.chat(
  parameters: {
    messages: [{role: "user", content: prompt}],
    model: "gpt-4o-mini",
    temperature: 0.0,
    max_tokens: 10
  }
) # gem 'ruby-openai'
```

LLMへのプロンプティング文字列

https://github.com/antiwork/gumroad/blob/production-f020aa7a68c1/2025-07-05-21-18-17/app/models/product_refund_policy.rb#L120-L129

LLMアプリの考え方と用例

- ルールベースでプログラミングできる決定的なものから溢れた、人が運用でカバーしていた非決定的な領域をまずLLMでカバーする
- LLMを使ったチャットアプリケーション (例: QAシステム)
- ユーザー入力を元にLLMで要約や提案を生成

2

RAG

Retrieval-Augmented Generation

LLMの特性とその対処

- LLMの学習データはトレーニングしリリースした時点に限られるため、学習以後の最新情報や非公開情報の外部知識を持っていない
- 対応としてLLMをチューニングする手法と、外部知識を検索してプロンプトに埋め込む手法がある

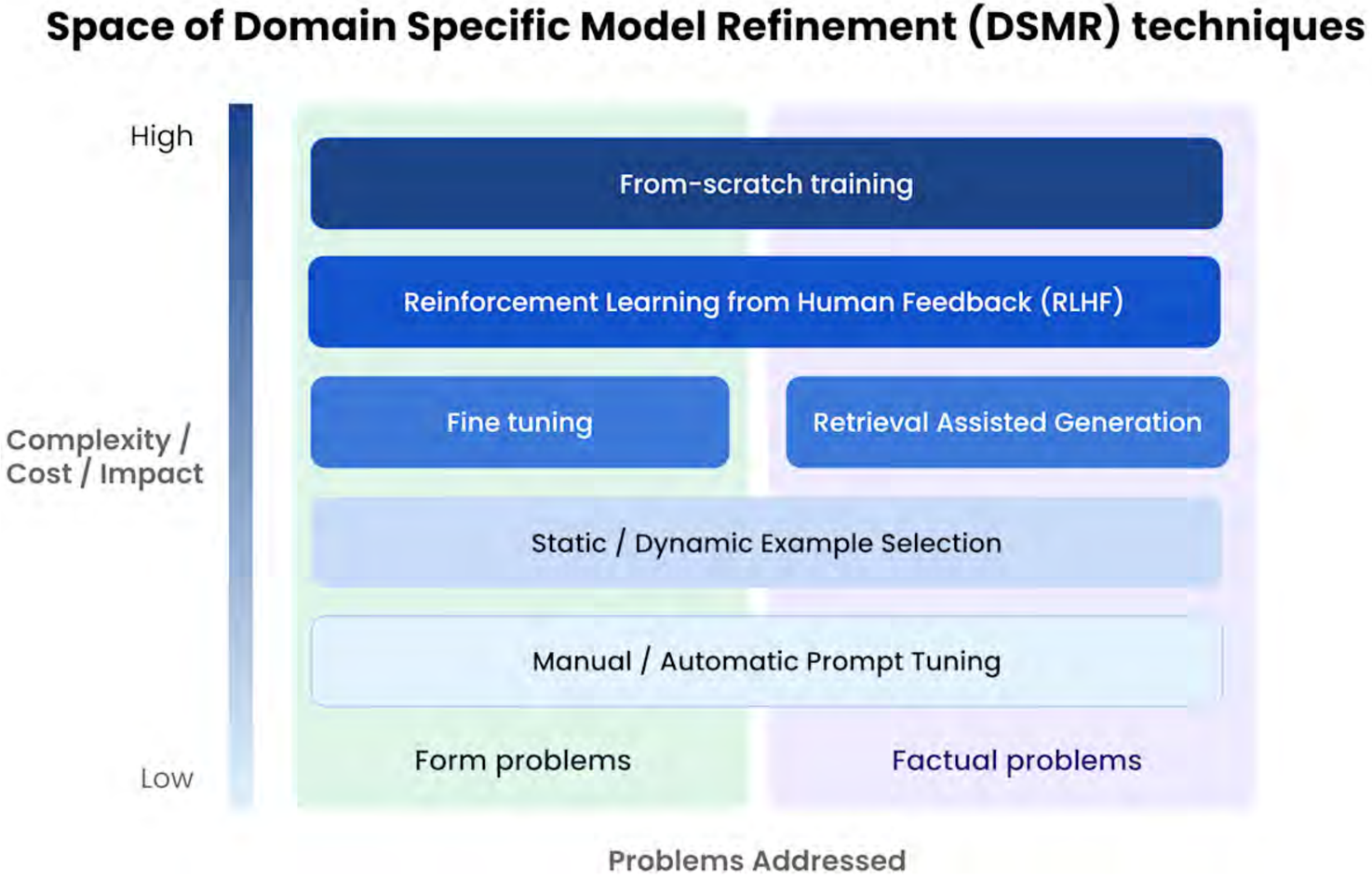
LLMそのものの訓練や調整

Training Fine-turning

- PythonではPyTorchやTensorFlowが、LLMのトレーニングに使われる (らしい)
- 例えばPyTorchのRubyバインディングにTorch.rbがあるが、LLMのトレーニングはPythonを使っておくのがわかりやすいかも？

トレーニングは膨大な計算量と電力が必要で、本気ならデータセンターの入手から！？

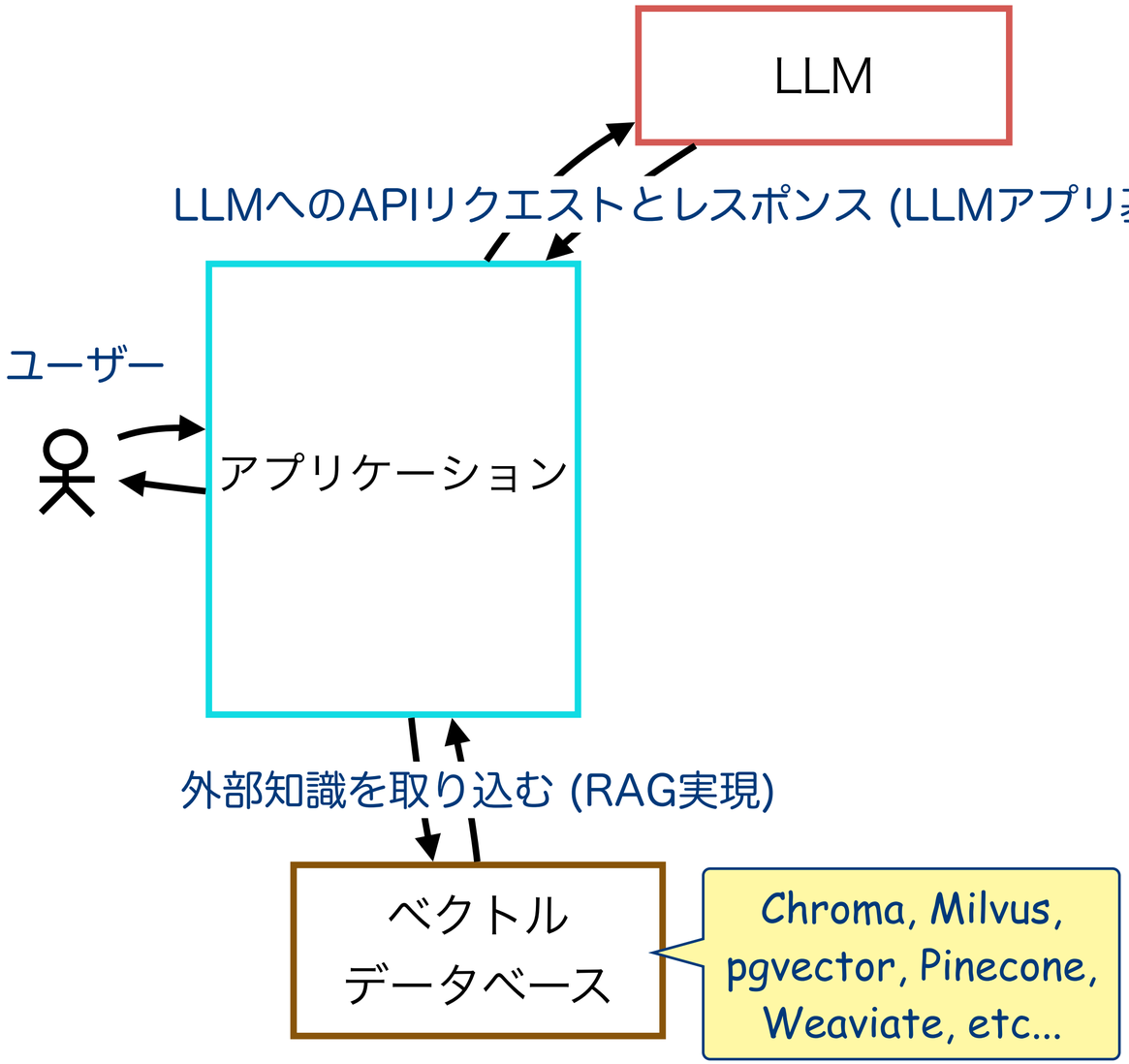
Fine-turning vs RAG



- Fine-turningは言葉遣いや出力という表現 (Form) への調整に使う
- 学習済み知識への事実 (Factual) の調整はRAGを使う

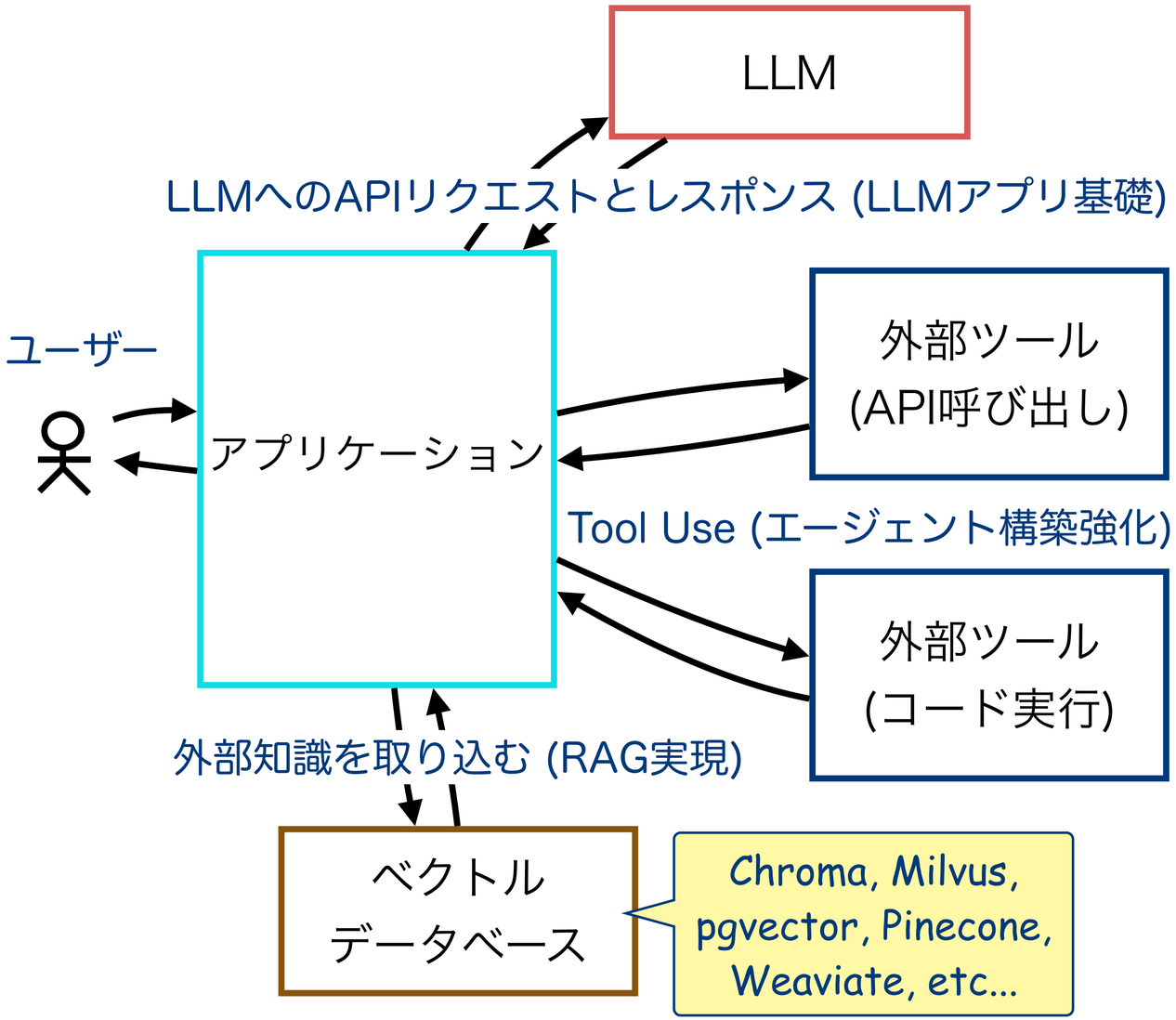
https://www.anyscale.com/blog/fine-tuning-is-for-form-not-facts

RAGのアプローチ



- LLMそのものをチューニングするのでなく、検索した外部知識をプロンプトに埋め込む
- 検索は主にベクトルDB
- RAGは手法であり標準化された規格ではない

RAGとツール呼び出し



- LLM知識にない情報が必要ならベクトルデータベース問い合せ結果を加味してLLMへプロンプティング (いわゆる広義のRAG)
- LLMのレスポンスに応じ、外部ツール呼び出しの仕組みを入れる (RAGと別軸)

RAG実現のフレームワーク

- RAGは手法を指すものであり、プラットフォームを跨いだ共通の規約はない
- PythonならLangChainやSemantic KernelなどRAG実現のためのフレームワークがある
- Ruby製RAGフレームワークLangchain.rbがRubyConf Taiwan 2024で紹介されている

Langchain.rb (Andrei Bondarev作)

- RAGやエージェント構築を実現するための、LLMへのAPIクライアント、ベクトルデータベース統合、ツール呼び出しなどを梱包
- それぞれのAPIを抽象化したプラガブルな構成になっている
- 本家LangChainと比べて、まだいろいろ不足

LLMを透過的に扱うAPIを持つ

- Langchain::LLM::Anthropicや
  Langchain::LLM::OpenAIなど
  LLMに応じたクライアントのラッパーを持つ

```
llm = Langchain::LLM::Anthropic.new(
  api_key: ENV['ANTHROPIC_API_KEY']
) # Requires the `anthropic` gem as an LLM client
```

- Langchain::LLM::Baseを継承し、llm#chatや
  llm#summarizeなど透過的な抽象APIを持つ

Langchain::Vectorsearch

- LLMが未知の外部知識を提供するため、
  ベクトルDBから意味的に近い情報を検索

```
require 'langchain'
require 'chroma-db'
chroma = Langchain::Vectorsearch::Chroma.new(
  url: ::Chroma.connect_host,
  index_name: 'ruby40_collection',
  llm: Langchain::LLM::Ollama.new)
response =
  chroma.ask(question: 'Ruby 4.0のRuby::Boxの作者は？')

response.chat_completion # => 'tagomorisさんです。'
```

Langchain::Tool

- LLMが外部機能として呼び出し判断できる、
  いくつかの組み込みツールをサポート

```
# Fetch weather from https://home.openweathermap.org
weather = Langchain::Tool::Weather.new(
  api_key: ENV['OPEN_WEATHER_API_KEY']
)

# Use `eqn` gem
calculator = LangChain::Tool::Calculator.new

# Use `safe_ruby` gem
interpreter = Langchain::Tool::RubyCodeInterpreter.new
```

Langchain::Assistant

- LLM、ツールなどを組み合わせてユーザーのプロンプティングに応答する中核のクラス

```
require 'langchain'
llm = Langchain::LLM::Ollama.new
assistant = Langchain::Assistant.new(
  llm: llm,
  tools: [Langchain::Tool::Weather.new(api_key: api_key)]
)
assistant.add_message(content: '明日の天気は？')

messages = assistant.run!
messages.each { |message| p message.to_hash[:content] }
```

RAGは冗長な抽象化説も浮上？

- RAGでは抽象化したAPIが用意されるが、LLMを変えることが通常ないのであれば、特定LLMのAPIに直接依存した方が容易？
- 例えば、Langchain::LLM::OpenAIはruby-openai gemへのラッパーに過ぎない

フレームワークでLLMアプリ像を知り、必要な個別ライブラリを使うとシンプル？

Langchain.rbによるRAGまとめ

- Langchain.rbは、LLMを囲むRubyのライブラリやミドルウェアを知るきっかけになる
- PythonのLangChainにある機能をRubyのLangchain.rbに移植できそう
- Langchain.rbへのコントリビューションのポイントとしてTool拡充、LCEL（LangChain Expression Language）サポート等

3

MCP

Model Context Protocol

modelcontextprotocol.io

MCPとは？

- Model Context Protocolの略称
- Anthropic社から策定公開されたプロトコル仕様であり、プラットフォーム非依存
- 2025-11-07現在、2025-06-18が最新版でドラフト版として次期仕様を策定中
  - Specification Enhancement Proposal
- SEPとしてGitHubイシューで仕様提案される

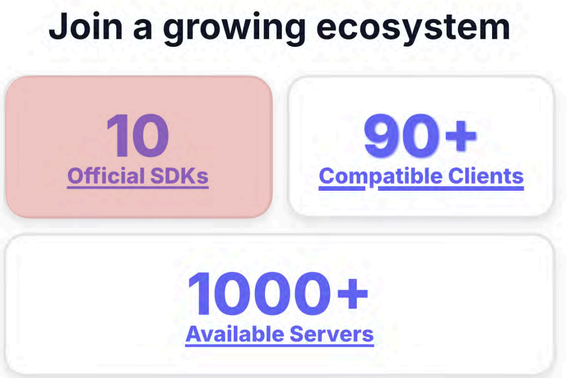
RubyにおけるMCP SDK gem

- Fast MCP (yjacquin/fast-mcp)
- MCP-RB (funwarioisii/mcp-rb)
- MCP on Ruby (rubyonai/mcp_on_ruby)
- MCP Ruby SDK (modelcontextprotocol/ruby-sdk)

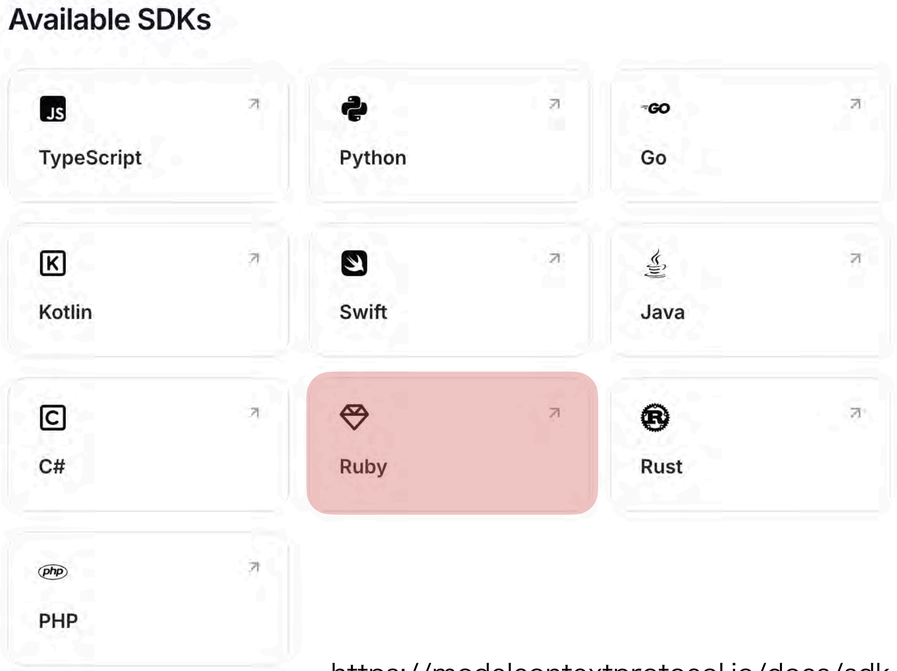
modelcontextprotocolのURLが圧倒的だったので、そこにコミットしている

MCP Official SDKs

- Python SDKとTypeScript SDKが先行
- 2025年11月現在は10個の公式SDKが存在



https://modelcontextprotocol.io/about



https://modelcontextprotocol.io/docs/sdk

MCP Ruby SDK

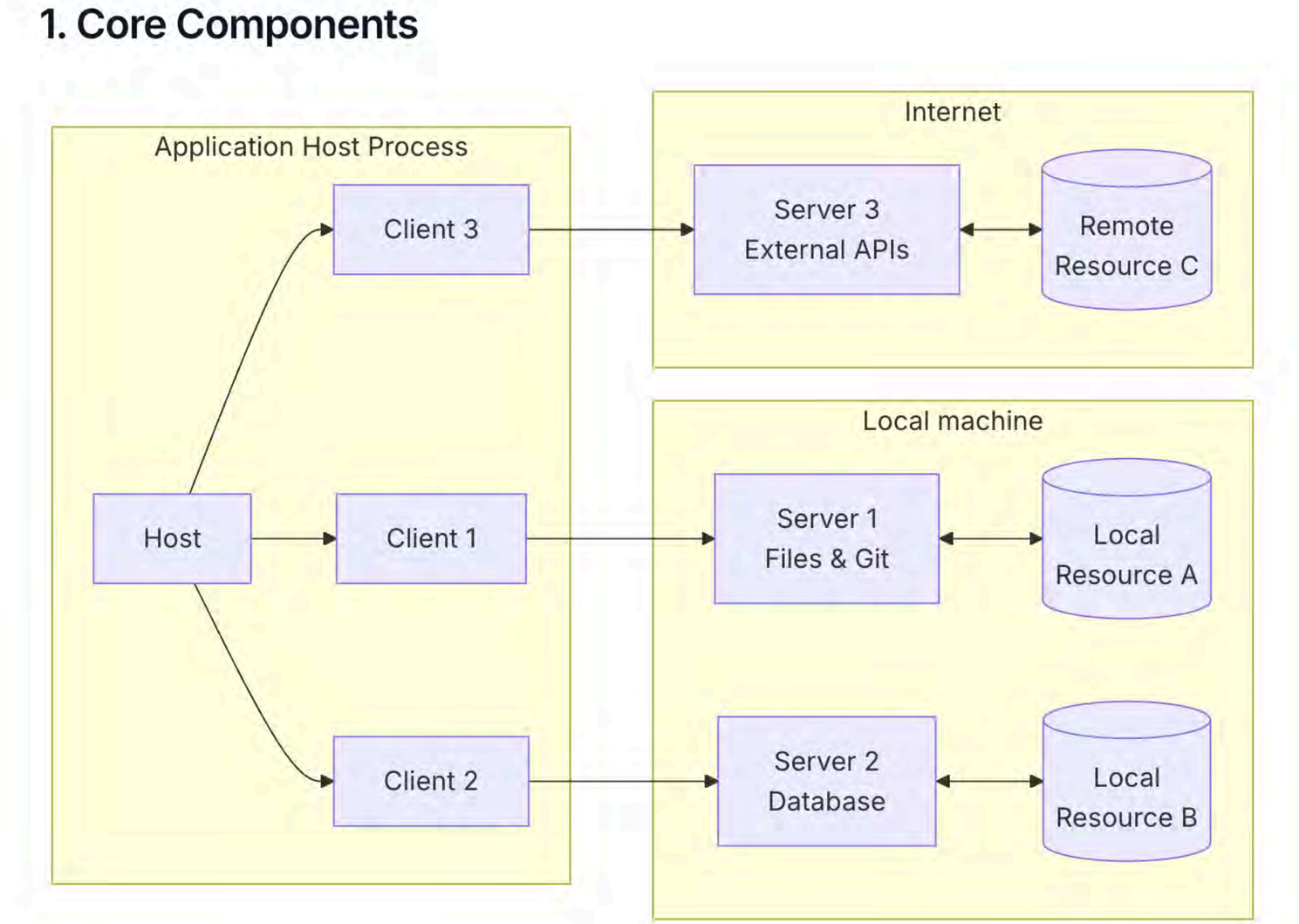
- Shopifyが社内で作っていたgemを公式として提案して公開が始まったっぽい
- MCPのガバナンス (SEP-001) が整うまでの間は、Shopifyでメンテナンスされていた
- 現在、Ateş Göral, Topher Bullock, 私の3人がメンテナーとして活動している

The Official Ruby SDK

- 公式SDKはMCPの仕様へ準拠追随することが求められている (ベストエフォート)
- Ruby版のMCP SDK gemでサーバーSDKに加え、クライアントSDK提供は現在唯一
- MCP Authentication (OAuth 2) 実装中

つまりMCPの仕様のアップデートに応じて、SDKも対応することが期待される

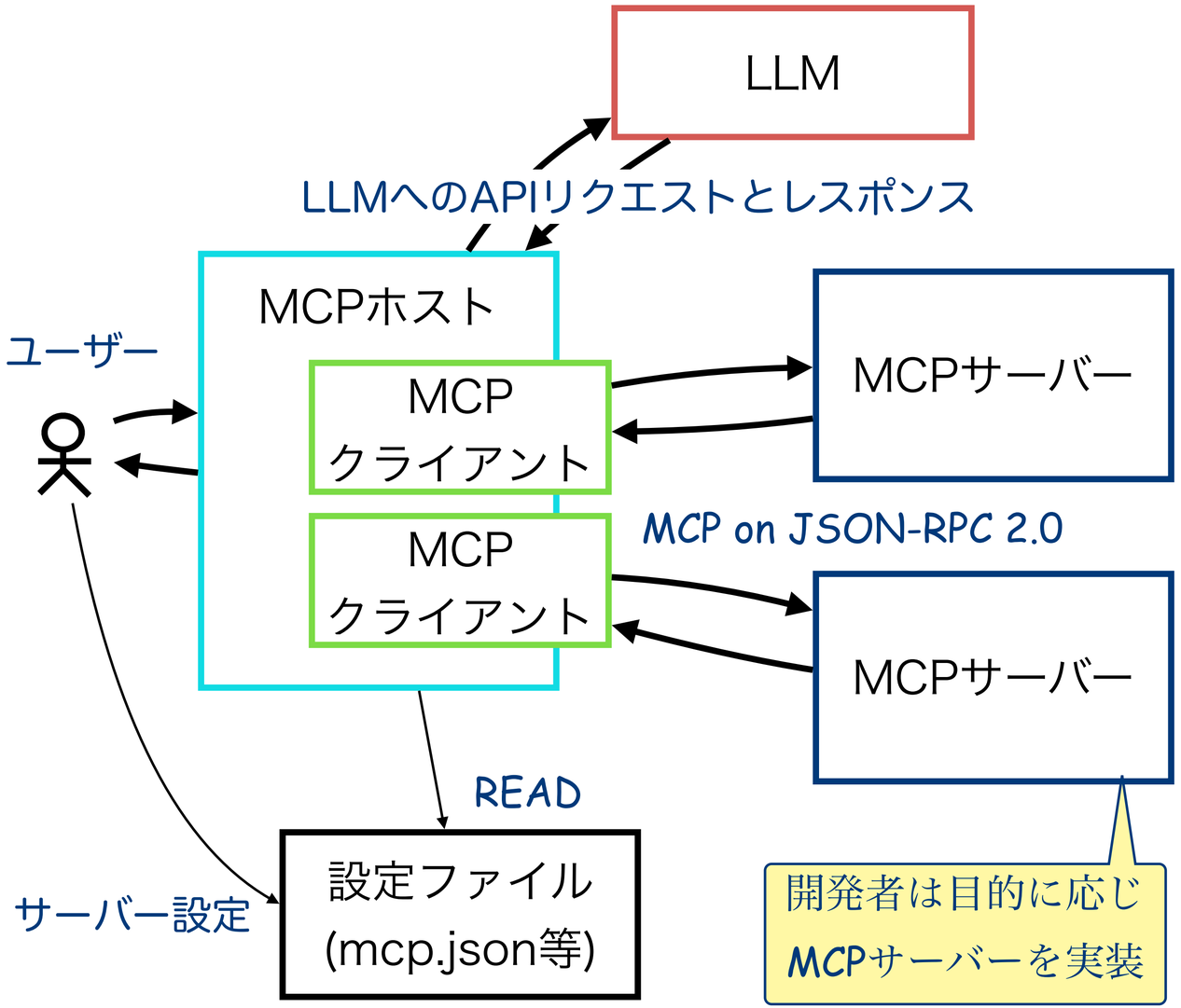
MCPのアーキテクチャ



https://modelcontextprotocol.io/specification/2025-06-18/architecture

- Language Server Protocol (LSP) に影響を受けたアーキテクチャ
- MCPはホスト、クライアント、サーバーで成る
- クライアント、サーバーはJSON-RPC 2.0基盤のMCPで通信する

MCPホスト中心のオーケストレーション



- MCPホストは、ユーザーとLLMとMCPクライアントの架け橋
- MCPホストは設定に基づいてMCPサーバーをLLMに伝える
- MCPホストはLLMがMCPサーバー呼び出しをするよう返したらMCPサーバー呼び出しをする
- LLMが直接MCPサーバーの呼び出しをするわけではない

Native御三家

- MCPサーバー提供のツール、リソース、プロンプトが主要機能
- 特にツールは汎用的な機能でクライアントの多くがサポート

Example Clients

Copy page

A list of applications that support MCP integrations

This page provides an overview of applications that support the Model Context Protocol (MCP). Each client may support different MCP features, allowing for varying levels of integration with MCP servers.

Feature support matrix

Client	Resources	Prompts	Tools	Discovery	Sampling	Roots	Elicitation
5ire	✗	✗	✓	✗	✗	✗	?
AgentAI	✗	✗	✓	?	✗	✗	?
AgenticFlow	✓	✓	✓	✓	✗	✗	?
AIQL TUUI	✓	✓	✓	✓	✓	✗	?
Amazon Q CLI	✗	✓	✓	?	✗	✗	?
Amazon Q IDE	✗	✗	✓	✗	✗	✗	?
Amp	✓	✗	✓	✗	✓	✗	?
Apify MCP Tester	✗	✗	✓	✓	✗	✗	?
Augment Code	✗	✗	✓				
BeeAI Framework	✗	✗	✓				
BoltAI	✗	✗	✓				
Call Chirp	✗	✓	✓				
ChatGPT	✗	✗	✓				
ChatWise	✗	✗	✓				
Claude.ai	✓	✓	✓				
Claude Code	✓	✓	✓				
Claude Desktop App	✓	✓	✓				
Chorus	✗	✗	✓				

Join a growing ecosystem

10 Official SDKs

90+ Compatible Clients

1000+ Available Servers

https://modelcontextprotocol.io/about

https://modelcontextprotocol.io/clients

クライアントサーバー間の通信

- stdioとStreamable HTTPをサポート
- stdio ... ローカルのMCPサーバーとして接続
- Streamable HTTP ... 必要に応じてSSE (Sever-Sent Events) が使われる。サーバー公開する場合はOAuth 2サポートを要確認

現状Streamable HTTPはエントリポイントをRack/Railsで自前実装する形式

RubyでMCPを実装するとは？

- MCP仕様に沿った実装がされたMCPクライアント、MCPサーバー間は相互運用可能
- プロトコル仕様の動くソフトウェアとして、何らかの言語処理系の実装が必要
- Rubyで実装すれば実装と運用のメンテナンスをRubyでできる

例

```ruby
require 'mcp'

server = MCP::Server.new(name: 'example')
server.define_tool(
  name: 'echo',
  description: 'Echoes back its arguments',
  input_schema: {
    properties: {
      message: {type: 'string'}
    }, required: ['message']
  }
) do |message:|
  MCP::Tool::Response.new([{
    type: 'text', text: "Echo for #{message}"
  }])
end
MCP::Server::Transports::StdioTransport.new(server).open
```

Langchain.rb vs MCPサーバー

- RAGは手法でLangchain.rbはLLMアプリの独自APIの提供フレームワークとして再利用
- MCPはプロトコル仕様でありMCPサーバーは規約準拠MCPクライアント横断で再利用
- いずれもLLM結果の評価は必要だが、MCPは小さく再利用しやすいデザイン

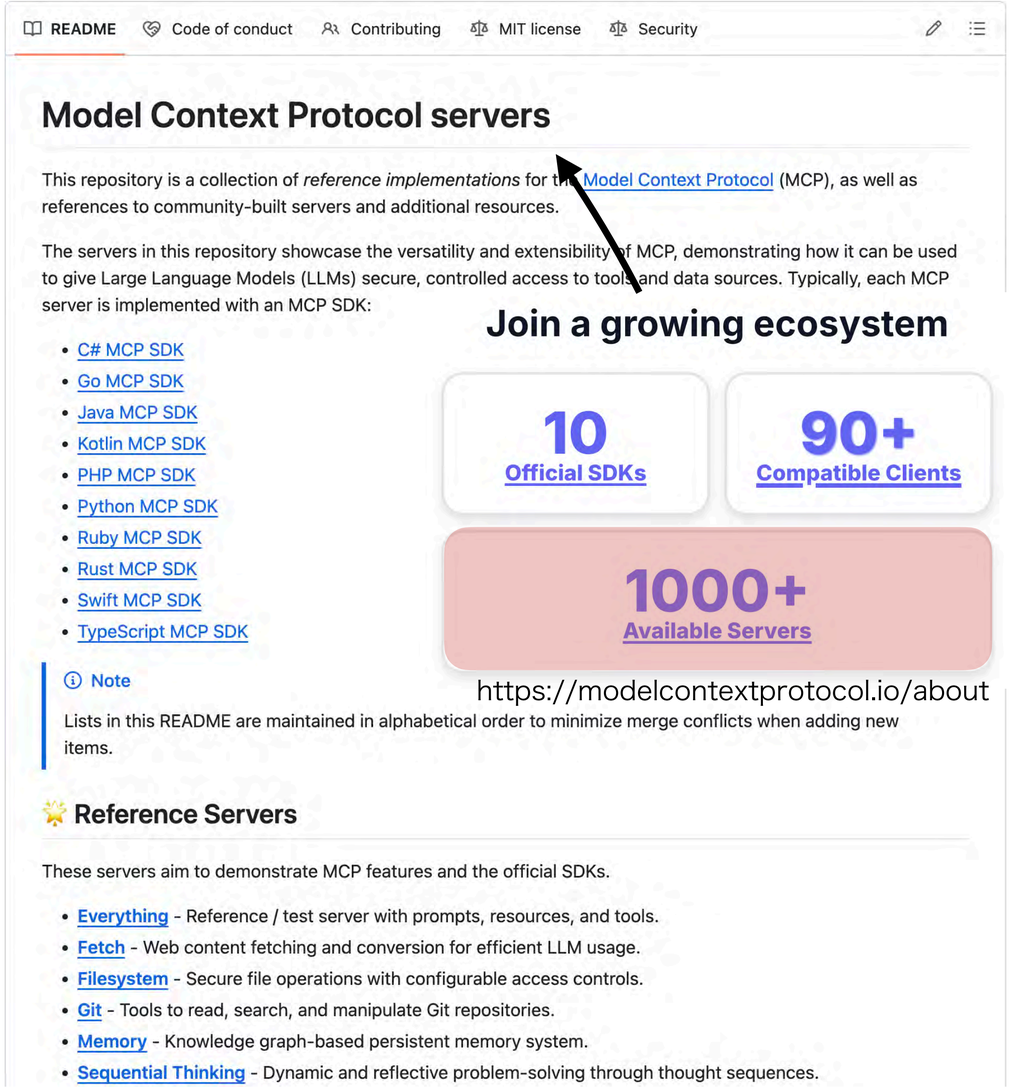
4

Ruby x LLMの可能性

RubyのRAGとMCPは後追い

- 実際のところ、Pythonが先行している
- RAGに関しては手法に過ぎないため、「ぼくが考えた最強のRAGフレームワーク」Ruby版が登場する余地はあるかもしれない
- MCPは仕様準拠のSDK実装のため仕様理解と不足しているRuby SDK開発を進めていく

MCPサーバー実装は言語間レース？



README | Code of conduct | Contributing | MIT license | Security

Model Context Protocol servers

This repository is a collection of *reference implementations* for the Model Context Protocol (MCP), as well as references to community-built servers and additional resources.

The servers in this repository showcase the versatility and extensibility of MCP, demonstrating how it can be used to give Large Language Models (LLMs) secure, controlled access to tools and data sources. Typically, each MCP server is implemented with an MCP SDK:

- C# MCP SDK
- Go MCP SDK
- Java MCP SDK
- Kotlin MCP SDK
- PHP MCP SDK
- Python MCP SDK
- Ruby MCP SDK
- Rust MCP SDK
- Swift MCP SDK
- TypeScript MCP SDK

Join a growing ecosystem

10 Official SDKs

90+ Compatible Clients

1000+ Available Servers

https://modelcontextprotocol.io/about

> **Note**
> Lists in this README are maintained in alphabetical order to minimize merge conflicts when adding new items.

Reference Servers

These servers aim to demonstrate MCP features and the official SDKs.

- Everything - Reference / test server with prompts, resources, and tools.
- Fetch - Web content fetching and conversion for efficient LLM usage.
- Filesystem - Secure file operations with configurable access controls.
- Git - Tools to read, search, and manipulate Git repositories.
- Memory - Knowledge graph-based persistent memory system.
- Sequential Thinking - Dynamic and reflective problem-solving through thought sequences.

https://github.com/modelcontextprotocol/servers

- Rubyエコシステムの広がりには Ruby処理系インストールが重要
- MCPサーバーは各言語で実装可
- Ruby製MCPサーバーが広がるということは、実行環境にRuby処理系が入る機会が増え、Rubyの多様性の広がりに繋がる！？

MCP Ruby SDKのアプリ事例求む

- Ruby界隈でのMCPの利用事例として、Claude CodeやCursorなどのツールをMCPホストとした話が割り合いとして多い？
- RailsアプリをMCPホストとしたLLMアプリからMCPサーバーを使った事例は寡聞

先日OpenAIからChatGPT Apps SDK基盤がMCPという発表もあり激流の最中

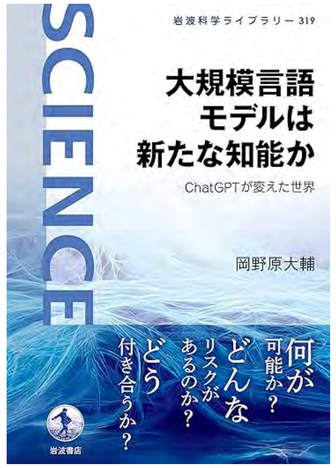
https://openai.com/index/introducing-apps-in-chatgpt

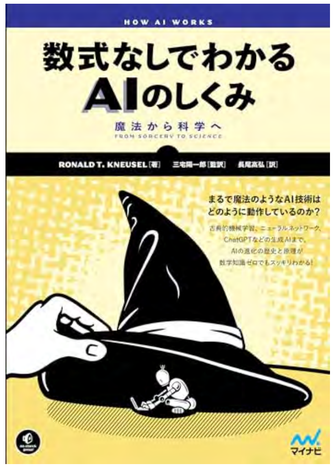
まとめ

- Ruby on RailsによるアプリケーションへのLLM活用の組み込みはRubyでもできるよ
- LLM周辺にはOSSで解決できることが沢山あり、パイオニアになるチャンス！？
- こんなLLM活用はできないかな？と思ったらPython先輩から学んでみるのもひとつ

LLMそのものを知る

- LLMはテキスト生成エンジンというツール
- LLMの仕組みを知ることはLLMを使った
  エンジニアリングスキルの向上に必須





Ruby x AI