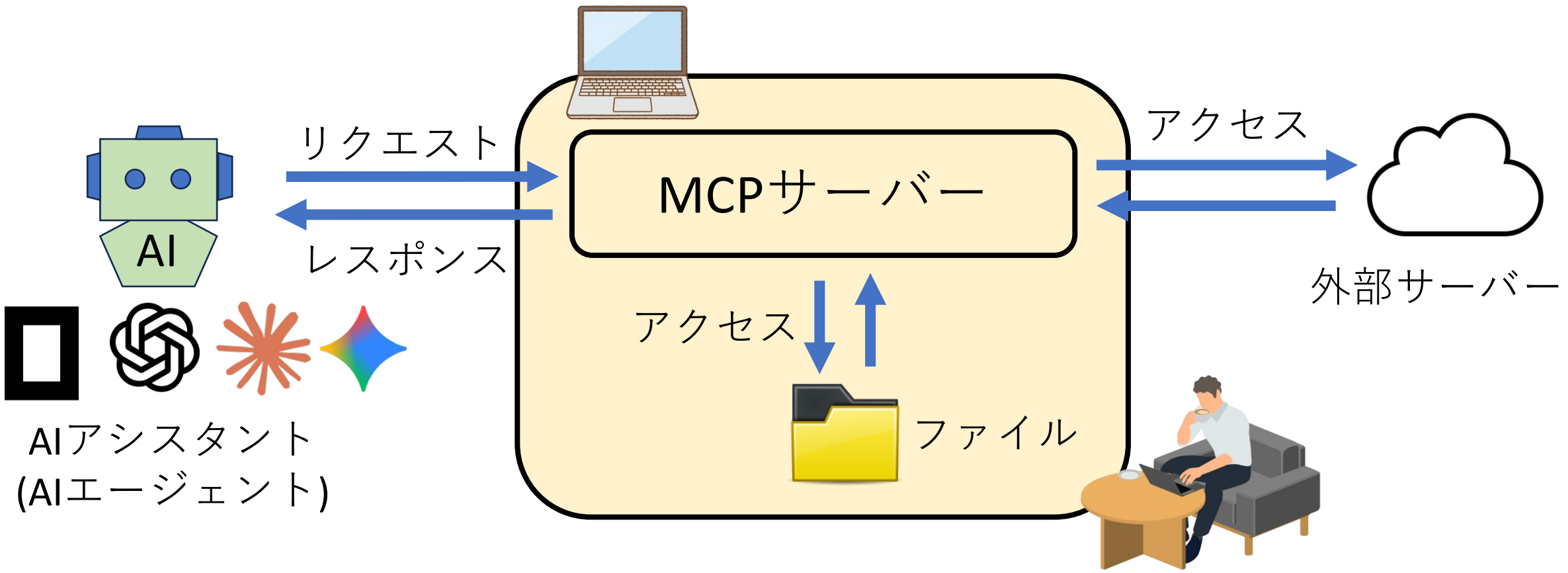
AIと「富岳」の連携ができるMCPサーバの紹介

高度情報科学技術研究機構(RIST)
HPC共用推進センター　産業利用推進部
須永 泰弘

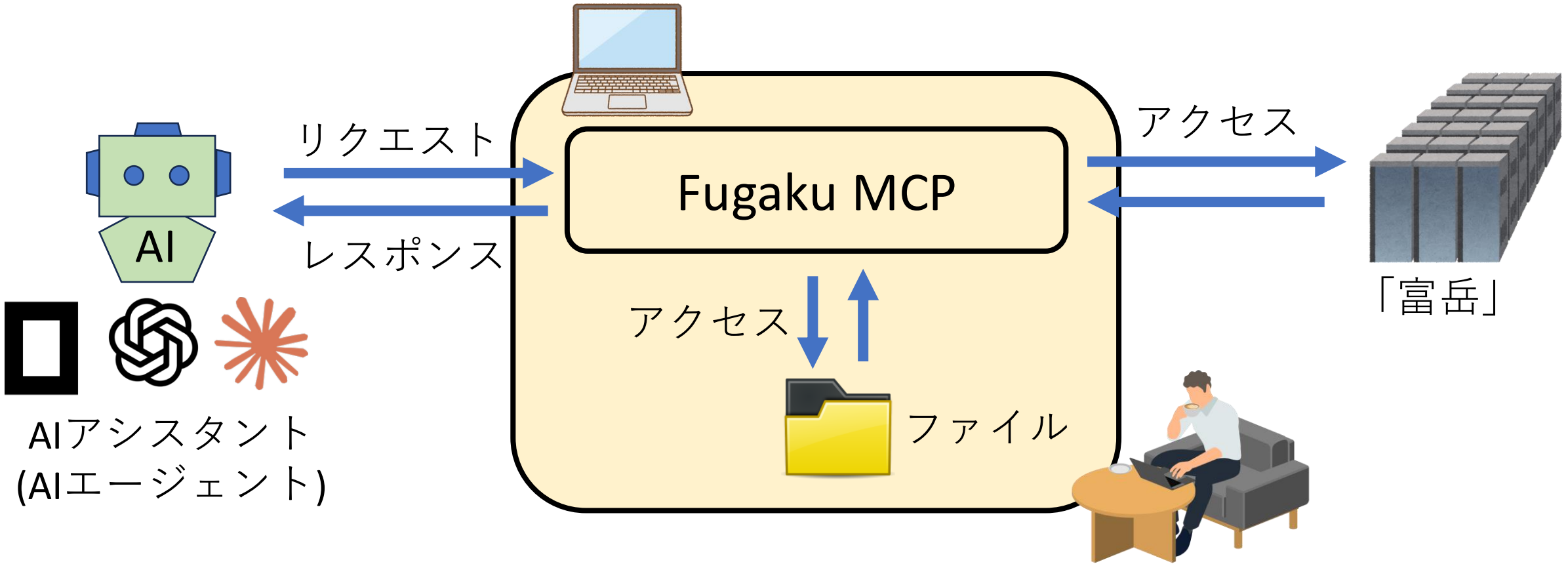
MCPサーバとは

Model Context Protocol (MCP)サーバーはAIが外部のツールやデータにアクセスするための「通訳」



富岳MCPサーバ

R-CCSが開発した「fugaku-mcp」により、アプリのことが分かっていなくても自然言語でAIエージェントに指示すれば「富岳」で実行して、結果を人に示してくれる「夢のような利用」も現実にできる。



「富岳」MCPサーバでできること

Fugaku-mcpクイックスタート に記載されている例

やりたいこと	話しかけ方の例
稼働状態	「富岳の状態を見て」
自分のジョブ一覧	「実行中のジョブある？」「最近終わったジョブを見せて」
ジョブ実行＋結果回収	「富岳で〇〇を実行して結果を見せて」
ファイルを送る/取る	「このファイルを富岳に置いて」「富岳の〇〇を取ってきて」

「富岳」の実行時のジョブスクリプトをイメージしながら
AIアシスタントに指示すれば、実行してくれる。

・リソースグループ(small? prior?)
・Spackにあるアプリケーションを使う？
・コンパイラは何を使う？　・Data領域で実行して！
・LLIO使う？　　・ログインノードで計算しない　　etc.

MCPを介した計算例(円周率の計算)

AIアシスタントへのプロンプト

C++・OpenMPでモンテカルロ法を使って円周率を推定して。
計算機は「富岳」の計算ノードを利用して。
１ノード48コアで計算した結果と、24コアずつ２ノードで計算した結果を比較して
試行回数は10回ずつ。計算時間と推定値を出力して
計算前に必要な情報があったら、計算前に相談して下さい。

AIアシスタントからの質問

・1試行あたりの乱数サンプル数（投点総数）はどれにしますか？　→ 10^9個
・「24コアずつ2ノード」の並列方式はどちらにしますか？
　→MPI×OpenMPハイブリッド (推奨)

mpiFCC -Kfast –Kopenmpでコンパイルできる C++のプログラムを生成（18 sec）

自動的に「富岳」の計算ノードに２個のジョブを投入

MCPを介した計算例(円周率の計算)

結果

構成	総コア	平均時間	平均時間(初回除く)
A: 1ノード48コア	48	0.242 s	0.234 s
B: 2ノード各24コア	48	0.239 s	0.234 s

AIアシスタントからの結果に対するコメント

・ 計算時間はほぼ同一（~0.235 s/試行）。総コア数がどちらも48なので当然で、単純な投点計算はノード間通信をほぼ行わないembarrassingly parallelのため、2ノード分割による差は出ません。
・推定値は全試行で 3.1415〜3.1417。10^9点での理論誤差（約1.3×10^{-5}）の範囲内で真値πに一致しています。
・構成Bは各MPIプロセス24スレッドで正しく2ノードに分散実行されました（nprocs=2, omp_threads=24を確認）。

計算の待ち時間を含めて11分7秒で完了
→　利用者は待ってるだけ！

「富岳」MCPサーバーで使えるAIアシスタント

AIアシスタント(エージェント)：人のように判断し実行まで行うシステム

- 商用のLLMを利用したい。 → Claude code, Codex
- 無料でLLMを利用したい。→OpenCode
- ローカルLLMを利用したい。 →OpenCode

Fugaku-mcp

https://github.com/RIKEN-RCCS/fugaku-mcp

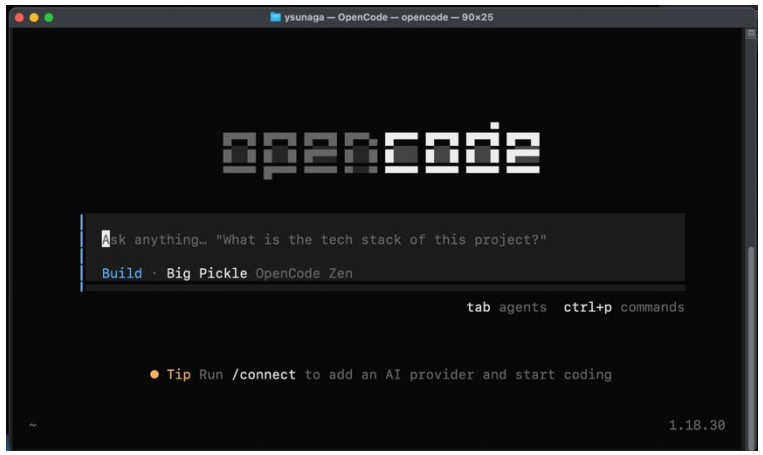
クイックスタート ― 富岳をAIアシスタントから使う

https://github.com/RIKEN-RCCS/fugaku-mcp/blob/main/QUICKSTART.md

OpenCode

オープンソースのAIコーディングエージェント

https://opencode.ai/ja/download



TUI(ターミナル)

デスクトップアプリ
(Win, Mac, Linux)

TUI

Windows/Linx	curl -fsSL https://opencode.ai/install | bash
Mac	brew install anomalyco/tap/opencode

注：OpenCodeのPathが設定が反映されない場合は端末の再起動か、source ~/.bashrcを実行

「富岳」MCPサーバの導入

クイックスタート ― 富岳をAIアシスタントから使う

https://github.com/RIKEN-RCCS/fugaku-mcp/blob/main/QUICKSTART.md

①準備

- 「富岳」証明書(.p12)
- Python (3.10以上)
- OpenSSL
- WSL2(Windowsのみ)

②MCPサーバーのダウンロードとインストール

```
git clone https://github.com/RIKEN-RCCS/fugaku-mcp.git
```

```
python3 -m venv .venv
```

```
.venv/bin/pip install "mcp[cli]~=2.0"
```

③証明書の登録(pemファイルの作成)

```
./setup_user.sh /mnt/c/Users/<Windowsのユーザー名>/Downloads/<あなたの証明書>.p12
```

Cloude Code用の設定ファイルが表示されるのでEditor等にコピーして保存することを推奨

OpenCodeへのMCPサーバーの登録

MCPサーバーの設定

注：OpenCodeを一度起動してから設定

~/.config/opencode/opencode.jsonc

```
{
  "$schema": "https://opencode.ai/config.json",
  "mcp": {
    "fugaku": {
      "type": "local",
      "command": [
        "/path/to/fugaku-mcp/.venv/bin/python",
        "/path/to/fugaku-mcp/fugaku_mcp.py"
      ],
      "enabled": true,
      "environment": { "FUGAKU_CERT": "/path/to/あなたの証明書.pem" }
    }
  }
}
```

← このカンマを忘れがち

LLM(大規模言語モデル)の選択

契約しているならClaudeやGPTなどの商用アプリを使うのが最適解

無償でとりあえず利用したい。
トークンを減らしたい。

Big Pickleを選択
(注：学習に利用される)
GLM-4.6相当

Big PickleはC++の生成は速いが、Fortranは時間がかかる印象

期間限定で他のLLMも利用可能なので試してみてください。

まとめ

- MCPサーバーによる「富岳」とAIアシスタントとの連携が可能となった。
- GitHubにあるR-CCSのクイックスタートガイドで簡単に導入できる。

- 有償のAIエージェントを契約していない方は、まず無償のOpenCodeを試してください。

- もう少し詳しいOpenCodeの導入手順書の公開を検討中