可視化利用ガイド

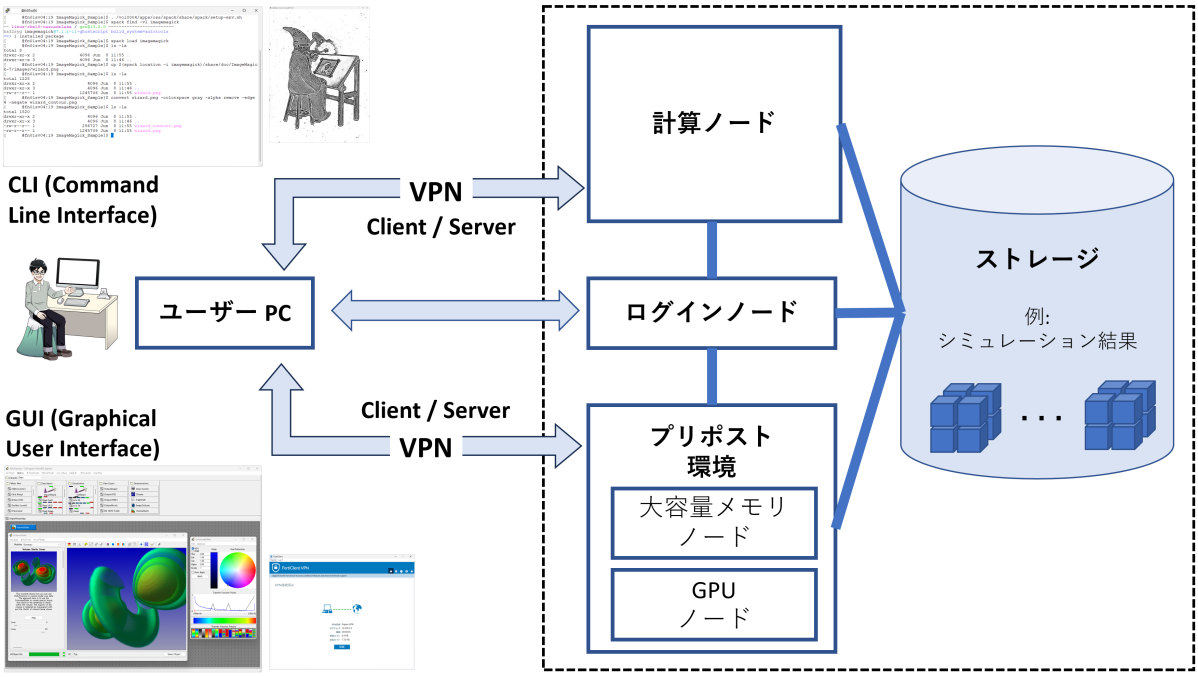
RIKEN R-CCS

2024 年 06 月 24 日

目次:

第1章　はじめに

- 本ドキュメントは「富岳」環境での可視化アプリケーションの主な利用方法（**CLI (Command Line Interface)**、**GUI (Graphical User Interface)**、**分散型（Client/Server）**）とその実行環境（**富岳 Open OnDemand**、**GNOME リモートデスクトップ**）について説明します。
- 可視化処理システムは **「プリポスト環境（大容量メモリノード、GPU ノード）」** と **「計算ノード」** を対象としています。
- **「プリポスト環境」** を用いた利用方法は **「ログインノード」** でも利用可能ですが、**ログインノードでの高負荷な処理の利用は極力避けるようお願いします**。ログインノード利用時の注意点については運用情報のお知らせ（https://www.fugaku.r-ccs.riken.jp/operation/20240611_01）もご確認ください。
- 可視化アプリケーションは汎用的なもの（ParaView、AVS/Express）から専用用途に特化したもの（Ncview、 GrADS）がインストールされており、ここでは主に起動方法について説明します。
- **ユーザーご自身でインストールしたソフトウェアを利用する際にはライセンス規定などに十分ご注意ください。**



注釈: 「可視化アプリケーション」の操作や実行時での問題はこのドキュメントの範囲外とさせて頂きます。必要に応じて開発元への問い合わせ等をお願いします。

1.1 表記について

可視化アプリケーションの利用において、ユーザー PC **(UserPC)**、ログインノード **(Login)**、プリポスト環境 **(PrePost)**、計算ノード **(Compute)** のどこでコマンド実行が行われるのかを明確にするため、本書では以下のプロンプト表記を使用します。

プロンプト	利用端末
[UserPC]$	利用者端末でのコマンド実行
[Login]$	ログインノードでのコマンド実行
[PrePost]$	プリポスト環境でのコマンド実行
[Compute]$	計算ノードでのコマンド実行

- 利用者 PC で複数の CLI コンソールを利用する場合は「UserPC_1]、「UserPC_2] と番号で識別します。
- プリポスト環境は「GPU ノード」、「大容量メモリノード」ともに共通です。

1.2 実行例について

実行例として富岳環境にインストールされている可視化・画像処理アプリケーションを利用しています。以下にインストール済みアプリケーションの一例を示しています。

- CLI (Command Line Interface)
  - **imagemagick**: ImageMagick (Image Processing Software)
  - **povray**: POV-Ray (Persistence of Vision Raytracer)
  - **grads**: GrADS (Grid Analysis and Display System)
  - **gmt**: GMT (The Generic Mapping Tools)
  - **gnuplot**: Gnuplot (Command-driven plotting program)
- GUI (Graphical User Interface)
  - **paraview**: ParaView (Data Analysis and Visualization Application)
  - **avs**: AVS/Express (Visual Programming Visualization Tool)
- Client/Server
  - **paraview**: ParaView (Data Analysis and Visualization Application)

Spack で提供されているアプリケーションは使用する前にロードする必要があります。以下はプリポスト環境での ImageMagick のロード例です。

```
[PrePost]$ . /vol0004/apps/oss/spack/share/spack/setup-env.sh

[PrePost]$ spack find -x
...
-- linux-rhel8-a64fx / fj@4.10.0 ---------------------------------
```

(次のページに続く)

(前のページからの続き)

```
povray@3.7.0.8       grads@2.2.3     paraview@5.11.2

-- linux-rhel8-cascadelake / gcc@13.2.0 -------------------------
gmt@6.2.0            ncview@2.1.9     gnuplot@5.4.3
imagemagick@7.1.1-11
...

[PrePost]$ spack load imagemagick@7.1.1-11%gcc@13.2.0
```

Spack で提供されていないアプリケーションや有効化されていないオプションが必要な場合はご自身の Spack 環境（プライベート・インスタンス）でのビルドを試みて利用することも可能です。また、最新の Spack 環境では提供されていない状況であっても、以下の ParaView を用いた例のように以前の Spack 環境で提供されていたアプリケーションであれば、そちらの利用が可能である場合があります。

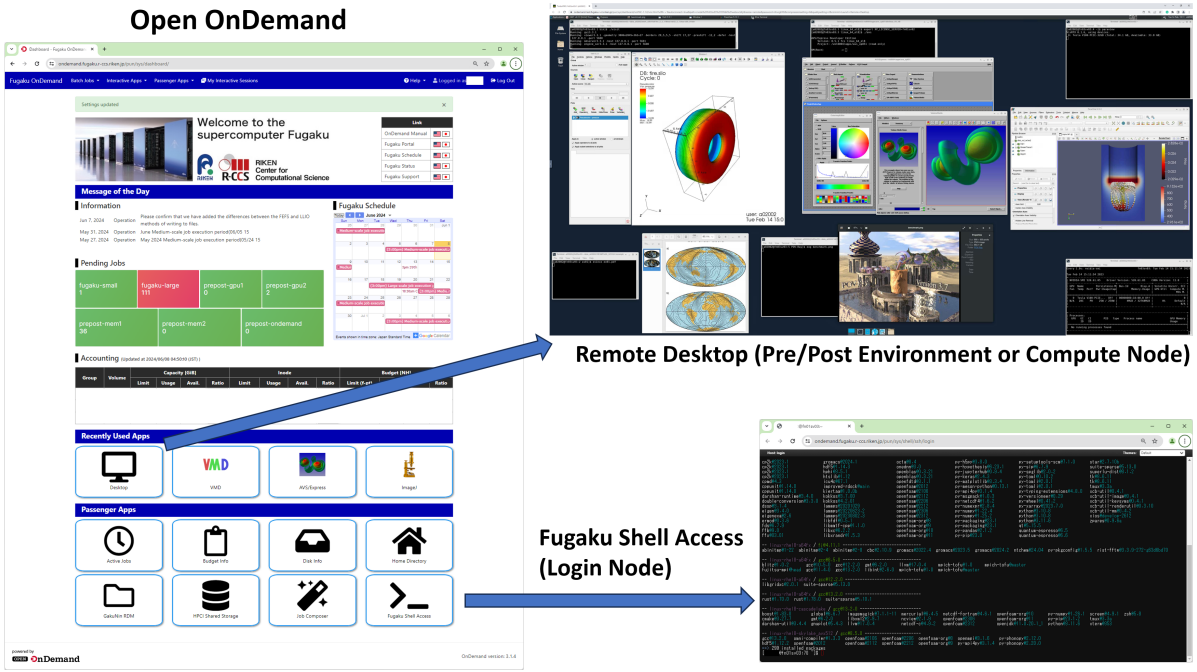
```
===========
Spack 0.21.0

[PrePost]$ . /vol0004/apps/oss/spack/share/spack/setup-env.sh

[PrePost]$ spack --version
 0.21.0

[PrePost]$ spack find paraview arch=linux-rhel8-cascadelake

==> No package matches the query: paraview arch=linux-rhel8-cascadelake



===========
Spack 0.19.0

[PrePost]$ . /vol0004/apps/oss/spack-v0.19/share/spack/setup-env.sh

[PrePost]$ spack --version
 0.19.0

[PrePost]$ spack find paraview arch=linux-rhel8-cascadelake
 -- linux-rhel8-cascadelake / gcc@12.2.0 -------------------------

 paraview@5.10.1
 ==> 1 installed package

[PrePost]$ spack load paraview@5.10.1%gcc@12.2.0
```

注釈: 上記の Spack 出力結果は単なる例であり、インストールによる提供開始やアップデートにより異なったバージョンが表示される可能性があります。また、異なる Spack 環境での利用時の不具合（例えば、

データベースの整合性エラー）が発生する場合は Spack キャッシュのクリアで解消される可能性があります。Spack の利用方法に関する詳細な説明は「富岳 Spack 利用ガイド」を参照してください。

Open OnDemand 環境では Spack で提供されているアプリケーション以外に Open OnDemand 側に直接インストールされているアプリケーションも多数あります。中には Spack で提供されていないアプリケーションもありますので、必要に応じて Open OnDemand 環境をご利用ください。以下にインストール済みアプリケーションの一例を示しています。

- 汎用的可視化アプリケーション
- **paraview**: ParaView (Data analysis and visualization application)
- **visit**: VisIt (Open source, interactive, scalable, visualization, animation and analysis tool)
- **avs**: AVS/Express (Visual programming visualization tool)
- 特定利用向け可視化アプリケーション
- **vmd**: VMD (Molecular visualization program)
- **ovito**: OVITO (Open visualization tools)
- **VESTA**: VESTA (3D visualization program for structural models)
- **pymol**: PyMol (Molecular visualization system)
- **xcrysden**: XCrysDen (Crystalline and molecular structure visualization program)



1.3 更新履歴

- **0.9.1 版　(2024 年 6 月 24 日)**
- 「1. はじめに」にユーザー自身でインストーしたソフトウェアのライセンスに関する注意を追記しました
- 「2.5. VNC（GNOME デスクトップ)」で Anaconda 利用時の不具合関連でライセンスに関する注意を追記しました
- **0.9 版　(2024 年 6 月 20 日)**
- 「4. Kombyne」を追加しました
- Spack アップデートによる情報更新
- Open OnDemand アップデートによる情報更新
- 図と説明のアップデートをしました
- **0.8 版　(2023 年 7 月 12 日)**
- 「Open OnDemand ベータ版 (Xfce)」を「Open OnDemand」へと変更
- Open OnDemand アップデートによる情報更新
- Spack アップデートによる情報更新
- **0.7 版　(2023 年 3 月 8 日)**
- **利用方法の順序を変更しました**
  - 「Open OnDemand ベータ版 (Xfce)」を最初「2.1」に移動
- 「 2.1.1.1. GUI アプリケーションの起動例」でプリポスト環境向け Open OnDemand にインストールした VisIt と AVS/Express を追加しました
- 「2.3. GUI (Graphical User Interface)」で GUI 版 ParaView の説明を削除しました
- Spack アップグレードに関する情報更新
- **0.6 版　(2022 年 11 月 10 日)**
- 「2.2. GUI (Graphical User Interface)」で「4. X サーバー (Windows OS)」を追加しました
- 「2.5 Open OnDemand ベータ版 (Xfce)」 を追加しました
- 「3.3 Open OnDemand ベータ版 (Xfce)」 を追加しました
- Spack 関連等の情報更新
- **0.5.1.1 版　(2022 年 5 月 31 日)**
- 「プリポスト環境 利用手引書 1.7 版」に合わせて実行例を修正しました
- **0.5.1 版　(2022 年 5 月 18 日)**
- 「2. 利用方法」と「3. AVS/Express」での インテラクティブ・ジョブ投入例にメモリ量と GPU 有無の設定を追加しました

- 「プリポスト環境 利用手引書」に合わせて表記（プリポスト環境、GPU ノード）を修正しました
- **0.5 版　(2022 年 4 月 12 日)**
- 「3.2 ローカル PC」に AVS/Express をローカル PC へのインストール向け情報を追加しました
- 実行例の更新（プリポスト環境に用意された新しいジョブキューを利用）
- **0.4 版　(2021 年 10 月 25 日)**
- 「2.1 CLI」の実行例に PvPython と PvBatch を追加しました
- 「2.3　分散型」でのプリポスト環境での実行方法の更新（「--x11」オプションの必要性）
- Spack アップデート等による情報更新
- **0.3 版　(2021 年 7 月 26 日)**
- **「SSH ポート・フォワーディング」を「富岳 VPN 接続」に変更しました**
  - 2.3 分散型（Client/Server)
  - 2.4 VNC（GNOME デスクトップ)
- 「3. AVS/Express」を追加しました
- **0.2 版　(2021 年 5 月 17 日)**
- ノードの表記を修正しました
- 「1.1 表記について」を追加しました
- 「1.2 実行例について」を追加しました
- 「1.3 更新履歴」を追加しました
- 「2.4 VNC(GNOME デスクトップ) を追加しました
- **初版　(2021 年 4 月 23 日)**

第2章　利用方法

2.1 Fugaku Open OnDemand

Web ブラウザ上で利用できる Fugaku Open OnDemand サービスでは **CLI (Command Line Interface)** 及びに **GUI (Graphical User Interface)** を用いた可視化アプリケーションの利用が可能です。Open OnDemand サービスの詳しい利用方法は「富岳 Open OnDemand ガイド」に記載されていますが、技術的詳細及びに Open OnDemand 上での可視化に関する追加資料としては以下の **第 12 回 A64FX 向けチューニング技術検討会 (Ease To Use Fugaku Open OnDemand Service)** 資料もご参照ください。

- Material: Fugaku Open OnDemand [PDF]
  - https://www.hpci-office.jp/documents/meeting_A64FX/231023/MTGA64FX_12-Fugaku_Open_OnDemand.pdf
- Material: Visualization HandsOn [PDF]
  - https://www.hpci-office.jp/documents/meeting_A64FX/231023/MTGA64FX_12-Visualization_HandsOn.pdf

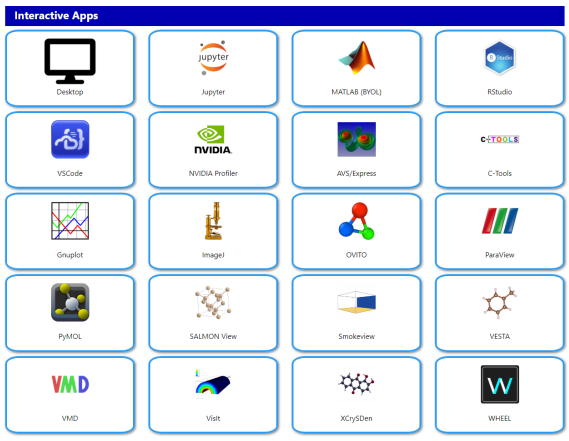
Open OnDemand サービスは専用 URL（https://ondemand.fugaku.r-ccs.riken.jp/） もしくは以下の様に富岳ウェブサイト（https://fugaku.r-ccs.riken.jp/） の **利用者支援** ウィンドウにある **Open OnDemand** のリンクからもアクセスが可能です。

“Default” (Recently Used Apps & Passenger Apps)

Help ➔ Profiles: Switch between “Default” and “All Apps”
“All apps” (“Default” + Batch Jobs + Interactive Apps)

注釈: Default 表示では Interactive Apps としてリストされている対話的に利用可能な可視化・解析向けアプリケーションが表示されませんので、その場合は All apps 表示に切換えて利用することを推奨します。

All apps モードでは **Batch Jobs** のグループに主に計算ノードでのバッチジョブ向けアプリケーションがリストされており、 **Interactive Apps** にのグループに主にプリポスト環境で対話的に利用可能な可視化・解析向けアプリケーションがリストされています。

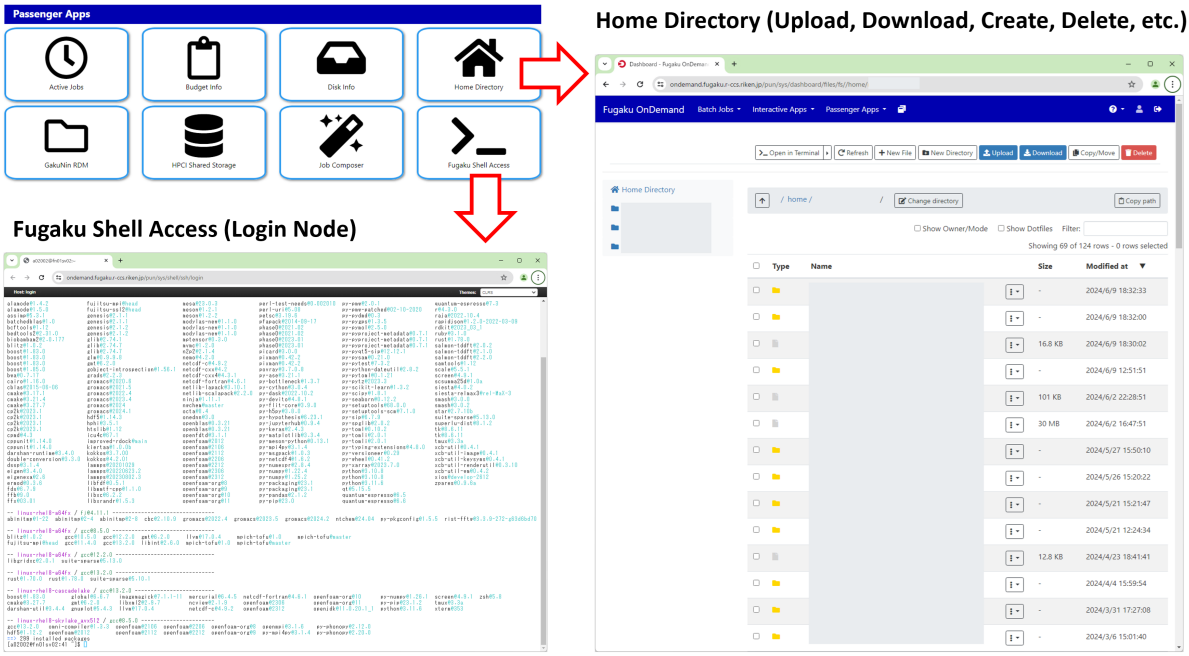
Batch Jobs	Interactive Apps
	

注釈: Open OnDemand の継続的なアップデートや設定変更により表示アイコンやインストールされているアプリケーションが上記例と異なる可能性があります。また、利用方法に関する詳細な説明は Open OnDemand サービスサイトの上部に表示されている "Link" リストの「OnDemand Manual」を参照してください。

2.1.1 Home Directory & Fugaku Shell Access

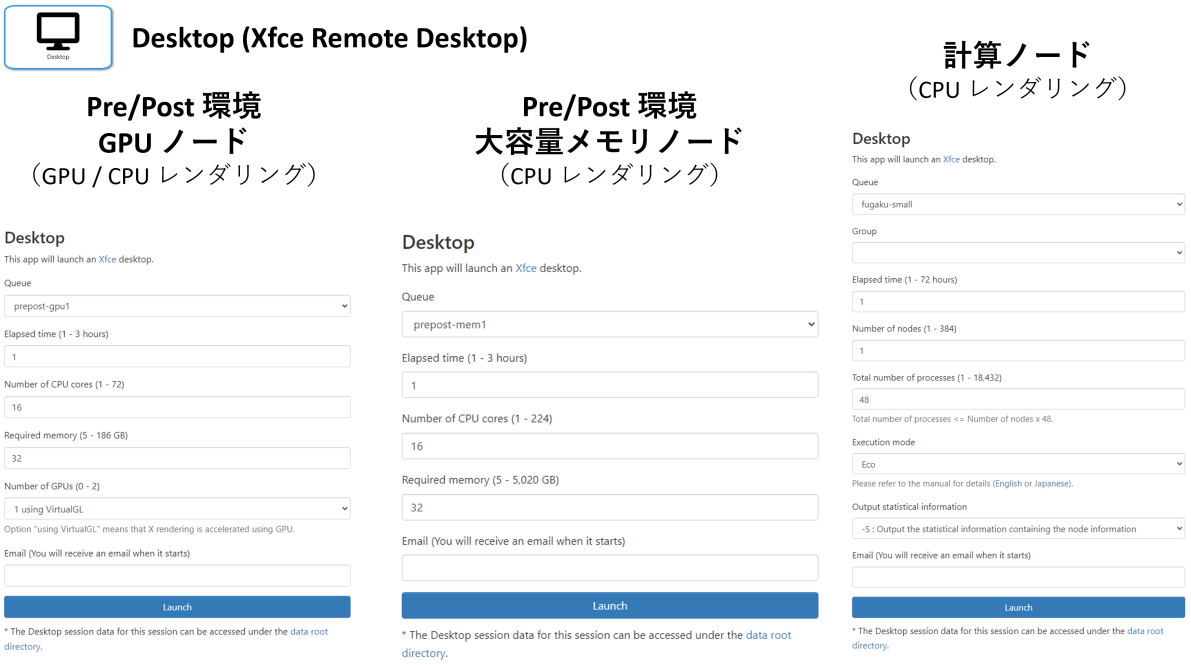
Passenger Apps にリストされている **Home Directory** より可視化向けファイルのアップロードや可視化結果のダウンロードが行えます。また、 **Fugaku Shell Access** より CLI ターミナル（ログインノード）が利用できます。注意点としては **ログインノードでの高負荷な処理の利用は極力避けるようお願いします**。その場合はプリポスト環境又は計算ノードへのバッチジョブ又はインテラクティブジョブでの利用をお願いします。

注釈: Passenger Apps の利用方法に関する詳細な説明は Open OnDemand サービスサイトの上部に表示されている "Link" リストの「OnDemand Manual」を参照してください。



2.1.2 Desktop (Xfce Remote Desktop)

プリポスト環境 または **計算ノード** で **Remote Desktop (Xfce Desktop)** を利用することができます。**Interactive Apps** の **Desktop** アイコンをクリックし、利用するジョブキュー、計算資源量や利用時間の設定を行い Remote Desktop セッションを起動します。



GPU ノードを利用するジョブキュー（prepost-gpu1, prepost-gpu2）で GPU の利用を選択した場合（Number of GPUs > 0）は GPU ベンダー（nVIDIA）の OpenGL を用いたハードウェア・レンダリング（GPU レン

ダリング）が実行されますので、レンダリング処理を重視する場合はこの利用を推奨します。GPU の利用を選択しなかった場合（Number of GPUs = 0）や大容量メモリノード又は計算ノード上では Mesa を用いたソフトウェア・レンダリング（CPU レンダリング）が実行されます。

Remote Desktop 環境で GPU レンダリングが有効になっているかは CLI ターミナルから以下の様に OpenGL ライブラリが GPU ベンダー（nVIDIA）のものに設定されているかを確認する方法があります。

```
GPU ノード（Number of GPUs > 0）
[PrePost]$ glxinfo | grep OpenGL
出力例：
OpenGL vendor string: NVIDIA Corporation
OpenGL renderer string: Tesla V100-PCIE-32GB/PCIe/SSE2
OpenGL core profile version string: 4.6.0 NVIDIA 535.129.03
...
```

以下の様に OpenGL ライブラリが Mesa に設定されていれば、CPU でレンダリング処理が行われます。

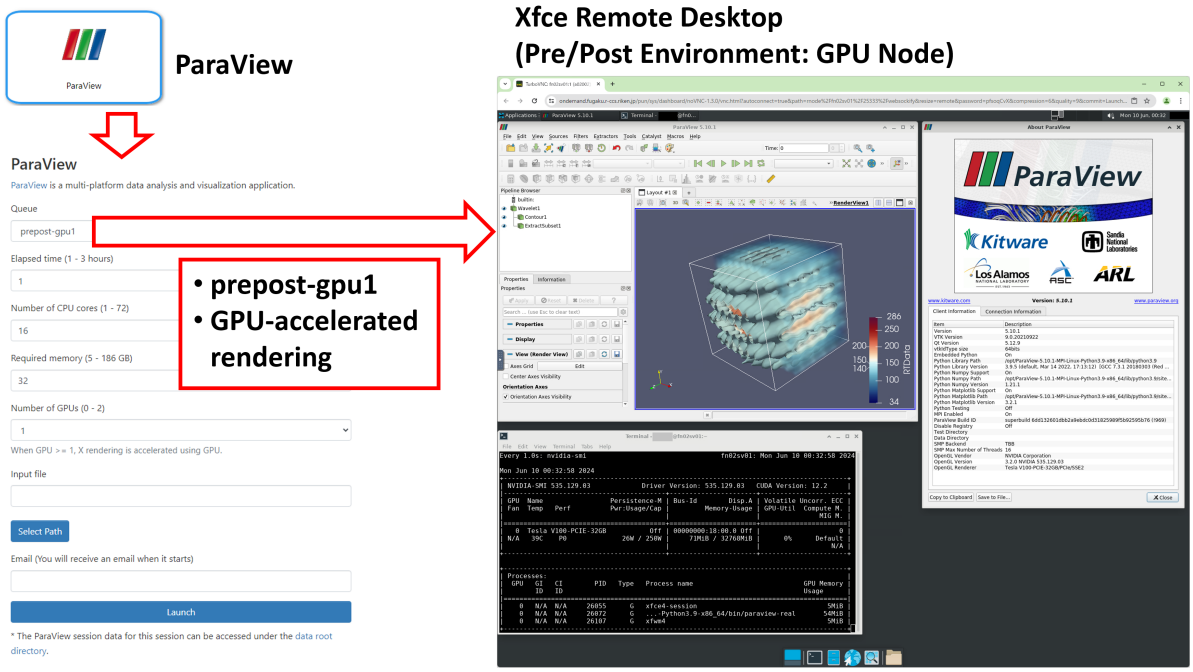
```
大容量メモリノード、GPU ノード（Number of GPUs = 0）
[PrePost]$ glxinfo | grep OpenGL
出力例：
OpenGL vendor string: Mesa/X.org
OpenGL renderer string: llvmpipe (LLVM 15.0.7, 256 bits)
OpenGL core profile version string: 4.5 (Core Profile) Mesa 22.3.0
...

計算ノード
[Compute]$ glxinfo | grep OpenGL
出力例：
OpenGL vendor string: Mesa
OpenGL renderer string: llvmpipe (LLVM 16.0.6, 128 bits)
OpenGL core profile version string: 4.5 (Core Profile) Mesa 23.1.4
```

注釈: 上記の出力結果は単なる例であり、アップデートや設定変更により異なるライブラリやバージョンが表示される可能性があります。

GPU レンダリングが有効になっている場合は、GPU 使用の確認が行える「nvidia-smi」ツールがあります（GPU を利用しているアプリケーションの確認も行えます）。例えば、以下の使用例では 1 秒毎に GPU 使用の情報更新が行われます。

```
[PrePost]$ watch -n 1 nvidia-smi
```

Interactive Apps にリストされているアプリケーションはアイコンをクリックすることで、設定画面より必要事項を選択及び記入後に起動できます。以下の例での ParaView をはプリポスト環境の GPU ノードで起動するための設定が行われ、最終的には Xfce リモートデスクトップ環境上に ParaView が起動されます。また、GPU ノードを利用するジョブキュー（prepost-gpu1, prepost-gpu2）で GPU の利用を選択しているため（Number of GPUs > 0）GPU を用いたハードウェア・レンダリングが有効となります。



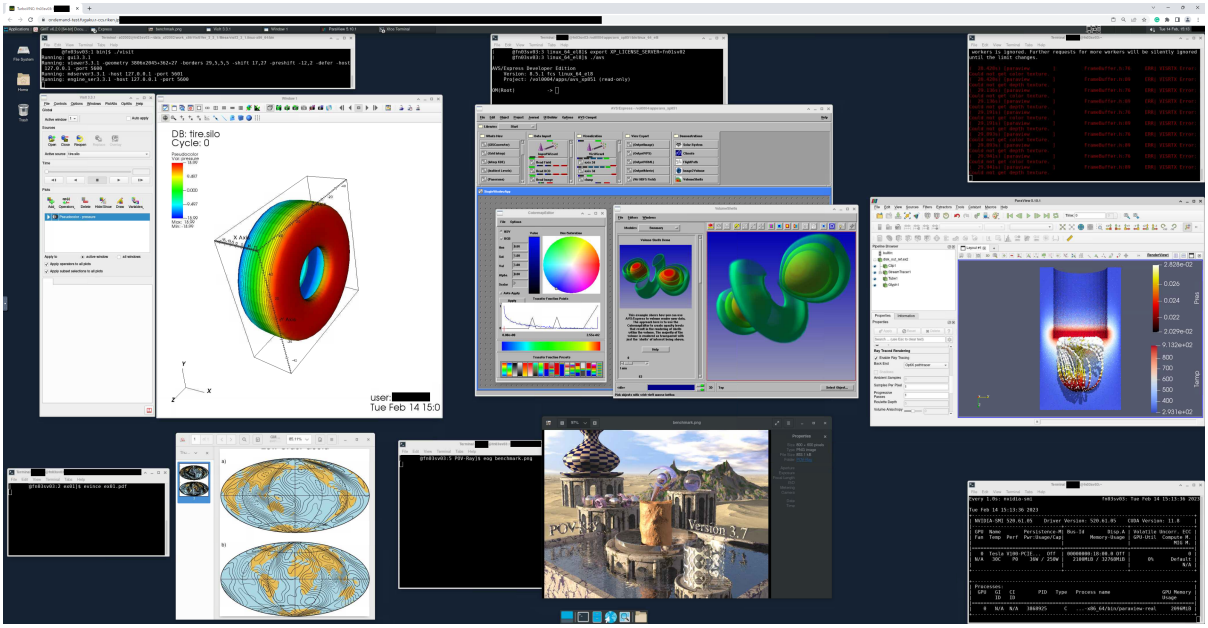
注釈: ParaView 以外にも「Interactive Apps」にリストされている多くのアプリケーションはリモートデスクトップ環境上での起動となるため、あらかじめ「Desktop」アイコンより Xfce リモートデスクトップを起動し、CLI ターミナルから可視化アプリケーションを起動して利用することも可能です。

プリポスト環境

以下は プリポスト環境の GPU ノードで起動した Xfce Remote Desktop セッションの利用例です。上段では Open OnDemand 上にインストールされている VisIt、AVS/Express、ParaView を実行しており、下段では evince、eog で GMT 及びに POV-Ray の可視化結果を表示しています。

Open OnDemand 環境にインストールされているアプリケーションは Spack を用いたロードをせずに利用可能です。以下は 一部のアプリケーション例であり、CLI ターミナルからは太字の部分を用いて起動できます。

- 汎用的可視化アプリケーション
- **paraview**: ParaView (Data analysis and visualization application)
- **visit**: VisIt (Open source, interactive, scalable, visualization, animation and analysis tool)
- **avs**: AVS/Express (Visual programming visualization tool)
- 特定利用向け可視化アプリケーション
- **vmd**: VMD (Molecular visualization program)



- **ovito**: OVITO (Open visualization tools)
- **VESTA**: VESTA (3D visualization program for structural models)
- **pymol**: PyMol (Molecular visualization system)
- **xcrysden**: XCrysDen (Crystalline and molecular structure visualization program)

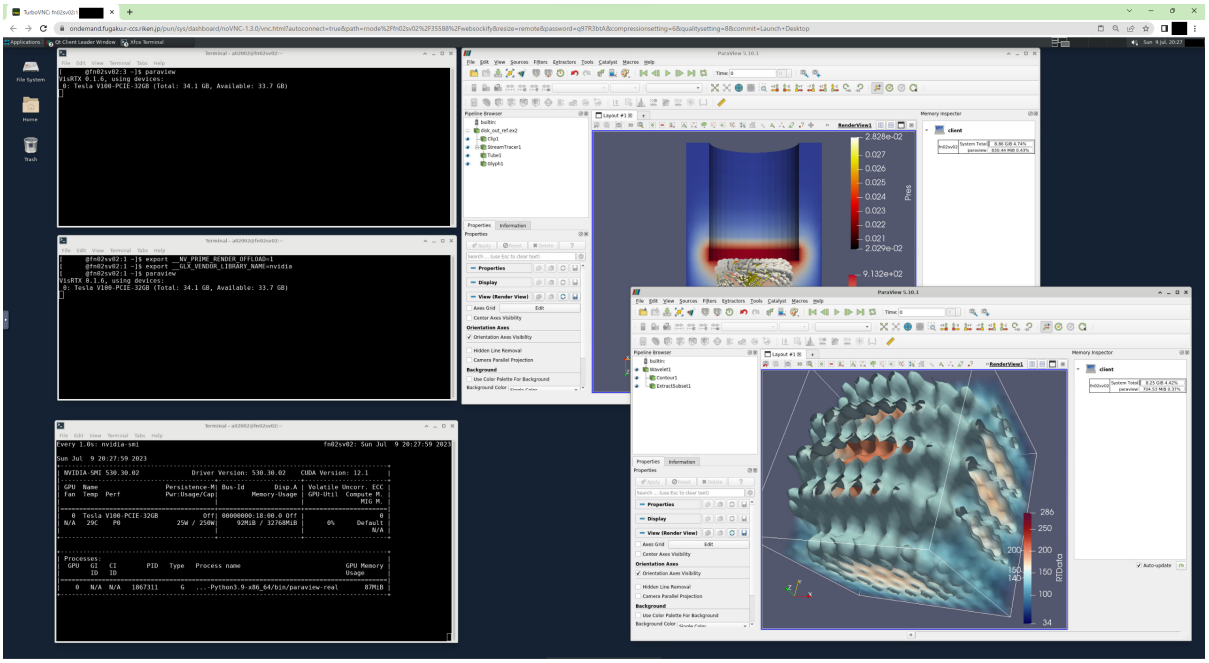
ア）ParaView

```
[PrePost]$ paraview
```

注釈: レイトレーシングを用いたレンダリング機能を利用するには「Ray Traced Rendering」設定画面で「Enable Ray Tracing」を選択する必要があります。その際に利用したいレイトレーシング・エンジン（CPU を利用する Intel OSPRay もしくは GPU を利用する NVIDIA OptiX）が選択できます。

イ）VisIt

VisIt は ParaView と同様に VTK (Visualization Tool Kit) をベースにして開発された大規模データ向け可視化アプリケーションです。



```
[PrePost]$ visit
```

ウ) AVS/Express

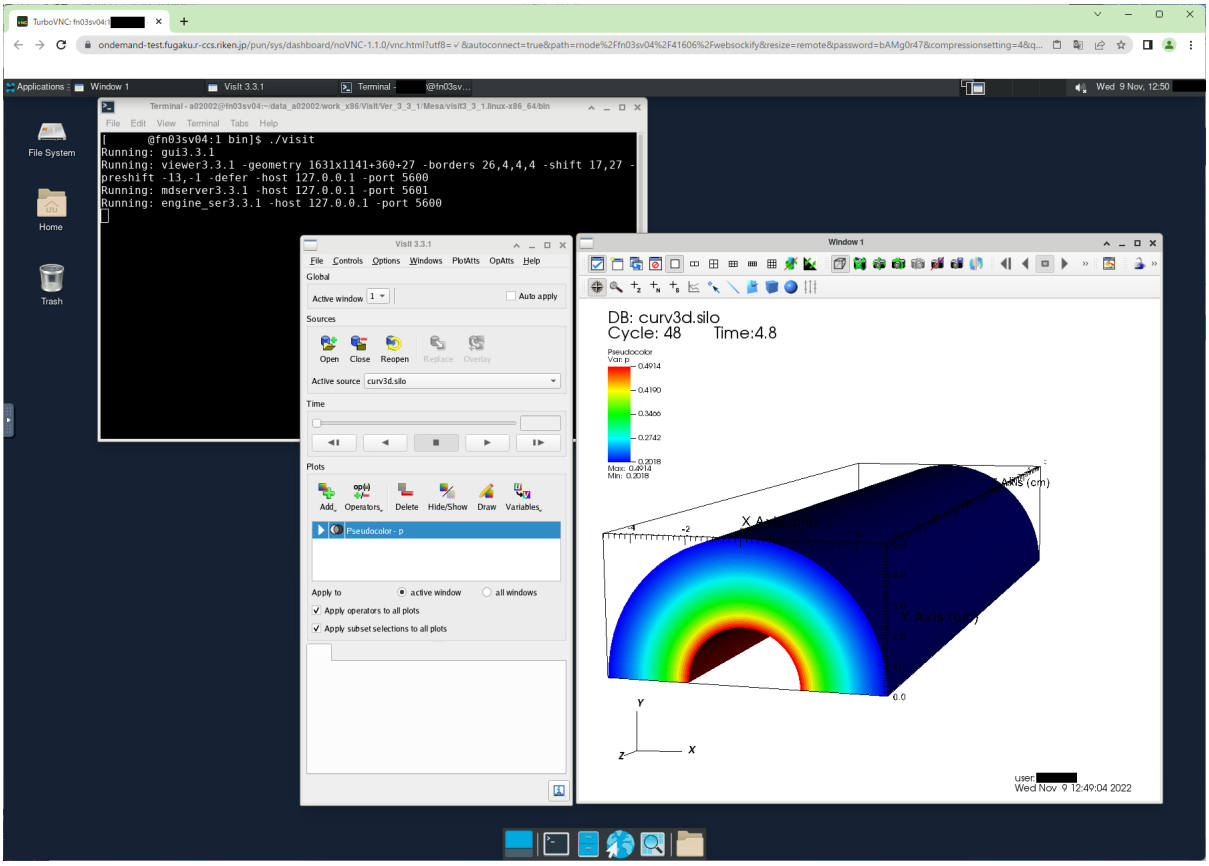
AVS/Express は ISV の商用汎用可的視化アプリケーションです。ライセンスサーバーはログインノード 2 (login2: fn01sv02) で動作しており、Open OnDemand 環境のみならずローカル PC にインストールして利用することも可能です。詳細は「AVS/Express」章を参照してください。

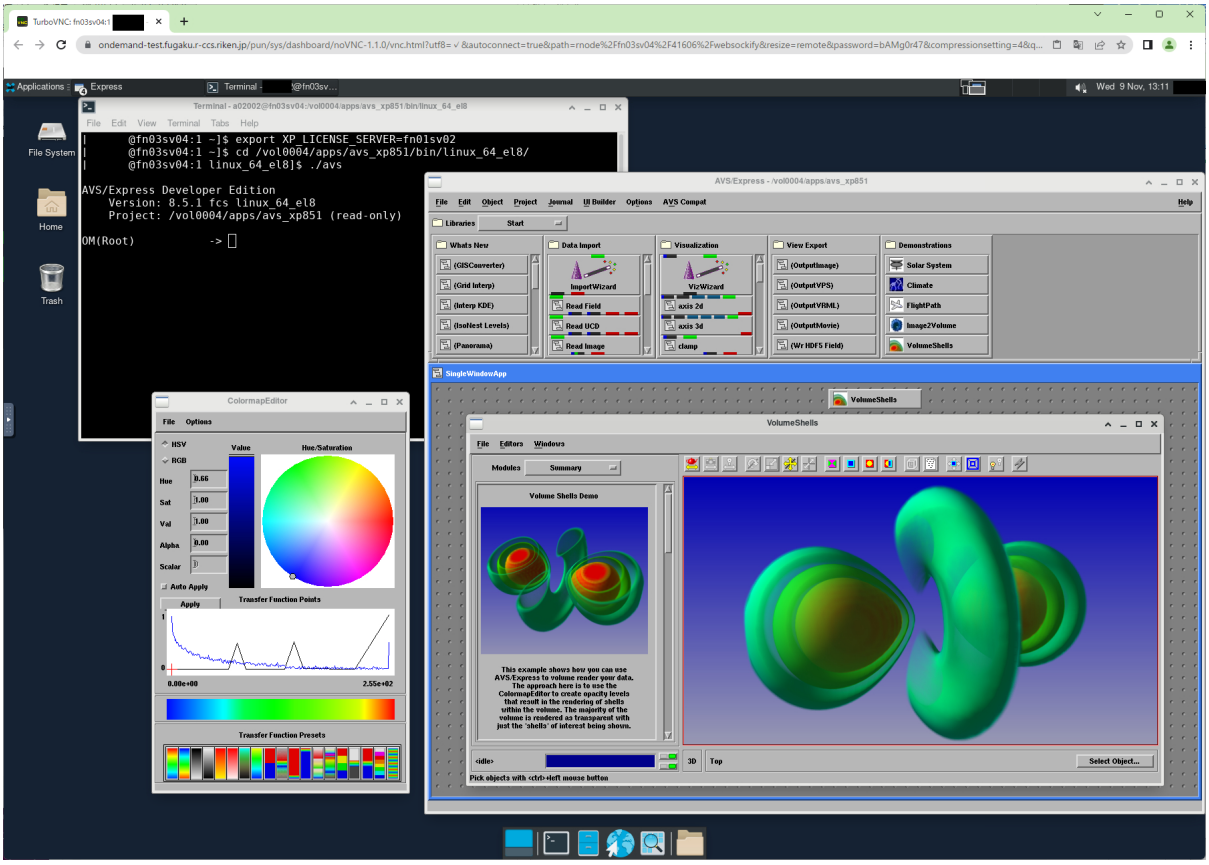
```
[PrePost]$ avs
```

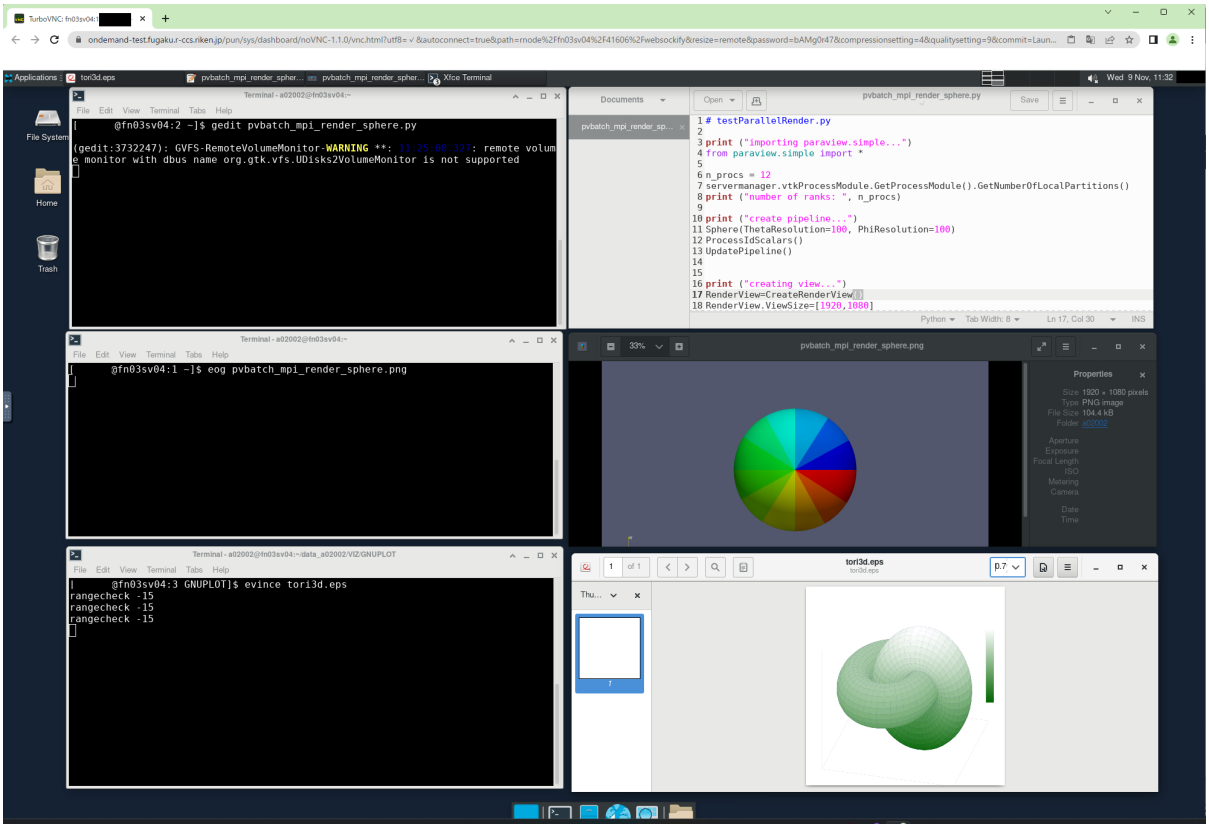
エ）デスクトップ環境アプリケーション

Xfce デスクトップ環境にアイコンで表示されているターミナル、ファイルブラウザーやウェブブラウザー以外にもコマンドラインから以下のような Xfce デスクトップ環境のアプリケーションも利用できます。

- テキストエディター： gedit
- 画像ビュワー： eog, display (ImageMagick)
- ドキュメントビュワー： evince
- グラフ描画ツール： gnuplot


Terminal - a02002@fn03sv04:~/data_a02002/work_x86/VisIt/Ver_3_3_1/Mesa/visit3_3_1.linux-x86_64/bin
@fn03sv04:1 bin]$ ./visit
Running: gui3.3.1
Running: viewer3.3.1 -geometry 1631x1141+360+27 -borders 26,4,4,4 -shift 17,27 -
preshift -13,-1 -defer -host 127.0.0.1 -port 5600
Running: mdserver3.3.1 -host 127.0.0.1 -port 5601
Running: engine_ser3.3.1 -host 127.0.0.1 -port 5600
VisIt 3.3.1
Window 1
DB: curv3d.silo
Cycle: 48 Time:4.8
Pseudocolor
Var: p
0.4914
0.4190
0.3466
0.2742
0.2018
Max: 0.4914
Min: 0.2018
X Axis (cm)
Y Axis (cm)
Z Axis (cm)
user:
Wed Nov 9 12:49:04 2022


Terminal - a02002@fn03sv04:/vol0004/apps/avs_xp851/bin/linux_64_el8
@fn03sv04:1 ~]$ export XP_LICENSE_SERVER=fn01sv02
@fn03sv04:1 ~]$ cd /vol0004/apps/avs_xp851/bin/linux_64_el8/
@fn03sv04:1 linux_64_el8]$ ./avs
AVS/Express Developer Edition
Version: 8.5.1 fcs linux_64_el8
Project: /vol0004/apps/avs_xp851 (read-only)
OM(Root) ->
AVS/Express - /vol0004/apps/avs_xp851
ColormapEditor
VolumeShells
Volume Shells Demo



注釈: Spack でロードされたアプリケーションの依存ライブラリによって上記のアプリケーションの起動に失敗する場合があります。その場合は新規の CLI ターミナルを開いて上記のアプリケーションの起動を試みてください。

オ）Spack で提供れているアプリケーション

CLI (Command Line Interface) や GUI (Graphical User Interface) 章の利用例で紹介している Spack（パブリック インスタンス）で提供されているアプリケーションが利用できます。

カ）Spack で提供されていないアプリケーション

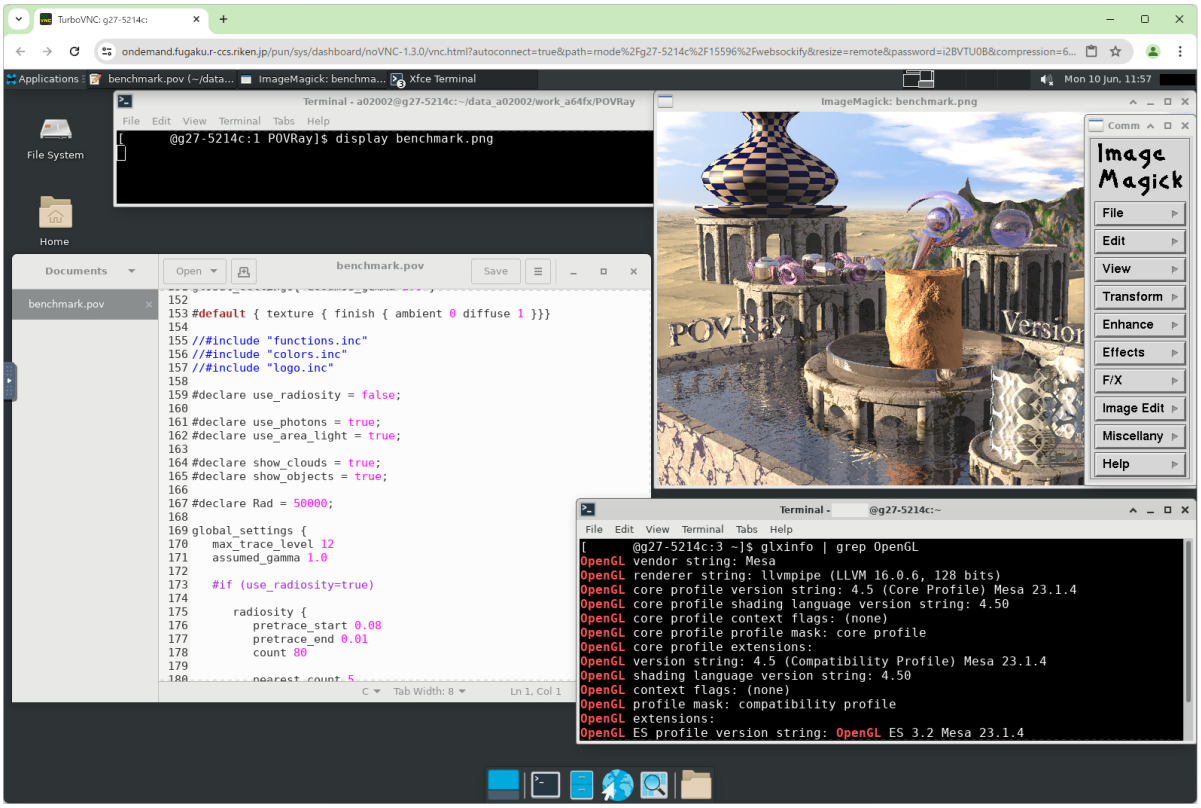
何らかの原因で Spack でビルドできなかったアプリケーションや Spack 向けビルドレシピが用意されていないパッケージ等をソースコードからビルドしたり、バイナリーファイルをダウンロードして利用することもできます。

計算ノード

ア）デスクトップ環境アプリケーション

Xfce デスクトップ環境にアイコンで表示されているターミナル、ファイルブラウザーやウェブブラウザー以外にもコマンドラインから以下のような Xfce デスクトップ環境のアプリケーションも利用できます。

- テキストエディター： gedit
- 画像ビュワー： display (ImageMagick)
- ドキュメントビュワー： evince
- グラフ描画ツール： gnuplot



イ）Spack で用意されているアプリケーション

CLI (Command Line Interface) や GUI (Graphical User Interface) 章の利用例で紹介している Spack (パブリック インスタンス）で提供されているアプリケーションが利用できます。

注釈: 計算ノードでは CPU レンダリングが行われますので、高負荷なレンダリング処理を行う場合はプリポスト環境（特に GPU ノード）の利用を推奨します。

2.2 CLI (Command Line Interface)

CLI コンソール上で可視化アプリケーションを利用する手法です。GUI インターフェイスを経由せず、主にファイル入出力で可視化処理を行う手法になります。

2.2.1 プリポスト環境

プリポスト環境でのインテラクティブ・ジョブ上で CLI 可視化アプリケーション（例として **imagemagick**）を利用する手順を示します。以下の手順は省略コマンドを利用しています。プリポスト環境の利用方法に関する詳細な説明は「プリポスト環境 利用手引書」を参照してください。

1. 富岳（ログインノード）へログイン

```
[UserPC]$ ssh user@login.fugaku.r-ccs.riken.jp
```

注釈: VNC GNOME Remote Desktop 内の CLI ターミナル、または Fugaku Shell Access (Open OnDemand) での CLI ターミナルからも利用可能です。ただ、後者の場合は X サーバーを必要とする可視化処理には利用できません。

---

2. プリポスト環境でインテラクティブ・ジョブをスタート

（GPU ノードでの実行例）

```
[Login]$ srun -p gpu1 -n 4 --mem 16000 --time=0:15:00 --pty bash -i
```

注釈: 必要に応じてジョブキュー (-p)、CPU 数 (-n)、メモリ量 (--mem)、GPU の有無 (--gpus=)、経過時間 (--time) を指定してください。プリポスト環境の利用方法に関する詳細な説明は「プリポスト環境 利用手引書」を参照してください。また、プリポスト環境での Open Ondemand Xfce Remote Desktop 内の CLI ターミナルを利用する場合は上記の 1. と 2. はスキップ可能です。

---

3. 実行例

- ImageMagick

```
[PrePost]$ . /vol0004/apps/oss/spack/share/spack/setup-env.sh

===========
[PrePost]$ spack find imagemagick

 -- linux-rhel8-cascadelake / gcc@13.2.0 -------------------------
 imagemagick@7.1.1-11
 ==> 1 installed package

[PrePost]$ spack load imagemagick@7.1.1-11%gcc@13.2.0
```

(次のページに続く)

(前のページからの続き)

```
==========
[PrePost]$ cp $(spack location -i imagemagick)/share/doc/ImageMagick-7/images/wizard.
↪png .

[PrePost]$ convert wizard.png -colorspace gray -alpha remove -edge 4 -negate wizard_
↪contour.png
```

入力ファイル: wizard.png	出力ファイル: wizard_contour.png
	

注釈: 上記の実行例では ImageMagick サンプル画像フォルダー「ImageMagick フォルダー/share/doc/ImageMagick-7/images/」のサンプル画像「wizard.png」を利用しています。Spack で提供されているパッケージのアップデート等によりフォルダーが上記と異なる可能性があります。Spack の利用方法に関する詳細な説明は「富岳 Spack 利用ガイド」を参照してください。

- ParaView PvBatch

```
==========
Spack 0.21.0

[PrePost]$ . /vol0004/apps/oss/spack/share/spack/setup-env.sh

[PrePost]$ spack --version
```

(次のページに続く)

(前のページからの続き)

```
0.21.0

[PrePost]$ spack find paraview arch=linux-rhel8-cascadelake

==> No package matches the query: paraview arch=linux-rhel8-cascadelake

===========
Spack 0.19.0

[PrePost]$ . /vol0004/apps/oss/spack-v0.19/share/spack/setup-env.sh

[PrePost]$ spack --version
0.19.0

[PrePost]$ spack find paraview arch=linux-rhel8-cascadelake

-- linux-rhel8-cascadelake / gcc@12.2.0 --------------------------
paraview@5.10.1
==> 1 installed package

[PrePost]$ spack load paraview@5.10.1%gcc@12.2.0

[PrePost]$ mpirun -n 4 pvbatch filename_of_parallel_pvbatch_script.py
```

- ParaView PvPython

```
[PrePost]$ mpirun -n 1 pvpython filename_of_pvpython_script.py
```

以下の様に提供されているパッケージが複数存在する場合、バージョン、コンパイラ、アーキテクチャーやハッシュ値等を用いた選択を行う必要があります。Spack の利用方法に関する詳細な説明は「富岳 Spack 利用ガイド」を参照してください。

```
===========
Spack 0.19.0

[PrePost]$ . /vol0004/apps/oss/spack-v0.19/share/spack/setup-env.sh

[PrePost]$ spack --version
0.19.0

[PrePost]$ spack load paraview

==> Error: paraview matches multiple packages.
Matching packages:
  oh5qwqo paraview@5.10.1%fj@4.8.1 arch=linux-rhel8-a64fx
```

(次のページに続く)

(前のページからの続き)

```
  uzxnwov paraview@5.10.1%fj@4.8.1 arch=linux-rhel8-a64fx
  i66ryps paraview@5.10.1%fj@4.8.1 arch=linux-rhel8-a64fx
  6temuv4 paraview@5.10.1%gcc@12.2.0 arch=linux-rhel8-cascadelake
Use a more specific spec (e.g., prepend '/' to the hash).
```

アーキテクチャを用いる例

```
[PrePost]$ spack load paraview arch=linux-rhel8-cascadelake
```

ハッシュ値を用いる例

```
[PrePost]$ spack load /6temuv4
```

2.2.2 計算ノード

計算ノード上でのインテラクティブ・ジョブ（会話型ジョブ）として可視化アプリケーション（例として **povray**）を利用する手順を示します。以下の手順は省略コマンドを利用しています。計算ノードの利用方法に関する詳細な説明は「利用手引書」を参照してください。

1. 富岳（ログインノード）へログイン

```
[UserPC]$ ssh user@login.fugaku.r-ccs.riken.jp
```

注釈: VNC GNOME Remote Desktop 内の CLI ターミナル、または Fugaku Shell Access (Open OnDemand) での CLI ターミナルからも利用可能です。ただ、後者の場合は X サーバーを必要とする可視化処理には利用できません。

---

2. 計算ノード上でインテラクティブ・ジョブをスタート

```
[Login]$ pjsub -x PJM_LLIO_GFSCACHE=/vol0004 --interact -L "eco_state=2" -L "node=1" -
↪-mpi "max-proc-per-node=4" -L "rscgrp=int" -L "elapse=0:15:00" --sparam "wait-
↪time=600"
```

注釈: 必要に応じてジョブで使用するデータ領域の volume（上記例：-x PJM_LLIO_GFSCACHE=/vol0004）や省エネルギー実行モード（上記例：-L "eco_state=2"）等の設定を行ってください。詳細な説明は「富岳ウェブサイト」や「富岳 利用手引書」を参照してください。計算ノードでの Open Ondemand Xfce Remote Desktop 内の CLI ターミナルを利用する場合は上記の 1. と 2. はスキップ可能です。

---

3. 実行例

- POV-Ray

```
[Compute]$ . /vol0004/apps/oss/spack/share/spack/setup-env.sh

==========
[Compute] spack find povray

-- linux-rhel8-a64fx / fj@4.10.0 ---------------------------------
povray@3.7.0.8
==> 1 installed packages


[Compute]$ spack load povray@3.7.0.8%fj@4.10.0

==========
[Compute]$ cp $(spack location -i povray@3.7.0.8)/share/povray-3.7/scenes/advanced/
↪benchmark/benchmark.pov .

[Compute]$ povray benchmark.pov
```

出力ファイル： benchmark.png

注釈: 実行例では POV-Ray サンプルシーンフォルダー「POV-Ray フォルダー/share/povray-

3.7/scenes/advanced/benchmark/」のサンプルシーン「benchmark.pov」を利用しています。

- ParaView PvBatch

提供されているパッケージが複数存在する場合、バージョン、コンパイラ、アーキテクチャー、ビルドオプションやハッシュ値等を用いた選択を行う必要があります。

```
[Compute]$ . /vol0004/apps/oss/spack/share/spack/setup-env.sh

===========
[Compute]$ spack find paraview

-- linux-rhel8-a64fx / fj@4.10.0 ---------------------------------
paraview@5.11.2  paraview@5.11.2  paraview@5.11.2
==> 3 installed packages

ハッシュ値やビルドオプションを検索
[Compute]$ spack find -vl paraview arch=linux-rhel8-a64fx

-- linux-rhel8-a64fx / fj@4.10.0 ---------------------------------
sknjmpn paraview@5.11.2~adios2~advanced_debug~catalyst~cuda+development_files~
 ↪examples~eyedomelighting~fortran~hdf5~ipo+kits~libcatalyst+mpi~nvindex+opengl2~
 ↪openpmd~osmesa~pagosa+python~qt~raytracing~rocm+shared~tbb~visitbridge build_
 ↪edition=canonical build_system=cmake build_type=Release generator=ninja␣
 ↪patches=02253c7,acb3805 use_vtkm=default
vx26636 paraview@5.11.2~adios2~advanced_debug~catalyst~cuda+development_files~
 ↪examples~eyedomelighting~fortran~hdf5~ipo+kits~libcatalyst+mpi~nvindex+opengl2~
 ↪openpmd+osmesa~pagosa+python~qt~raytracing~rocm~shared~tbb~visitbridge build_
 ↪edition=canonical build_system=cmake build_type=Release generator=ninja␣
 ↪patches=02253c7,acb3805 use_vtkm=default
tfrzwlo paraview@5.11.2~adios2~advanced_debug~catalyst~cuda+development_files~
 ↪examples~eyedomelighting~fortran~hdf5~ipo+kits~libcatalyst+mpi~
 ↪nvindex+opengl2+osmesa~pagosa~python~qt~raytracing~rocm~shared~tbb~visitbridge␣
 ↪build_edition=canonical build_system=cmake build_type=Release generator=ninja␣
 ↪patches=02253c7,acb3805 use_vtkm=default
==> 3 installed packages

===========
ビルドオプションとアーキテクチャを指定したロード例
[Compute]$ spack load paraview +python +osmesa arch=linux-rhel8-a64fx

ハッシュ値を用いたロード例
[Compute]$ spack load /vx26636

===========
[Compute]$ mpirun -n 4 pvbatch filename_of_parallel_pvbatch_script.py
```

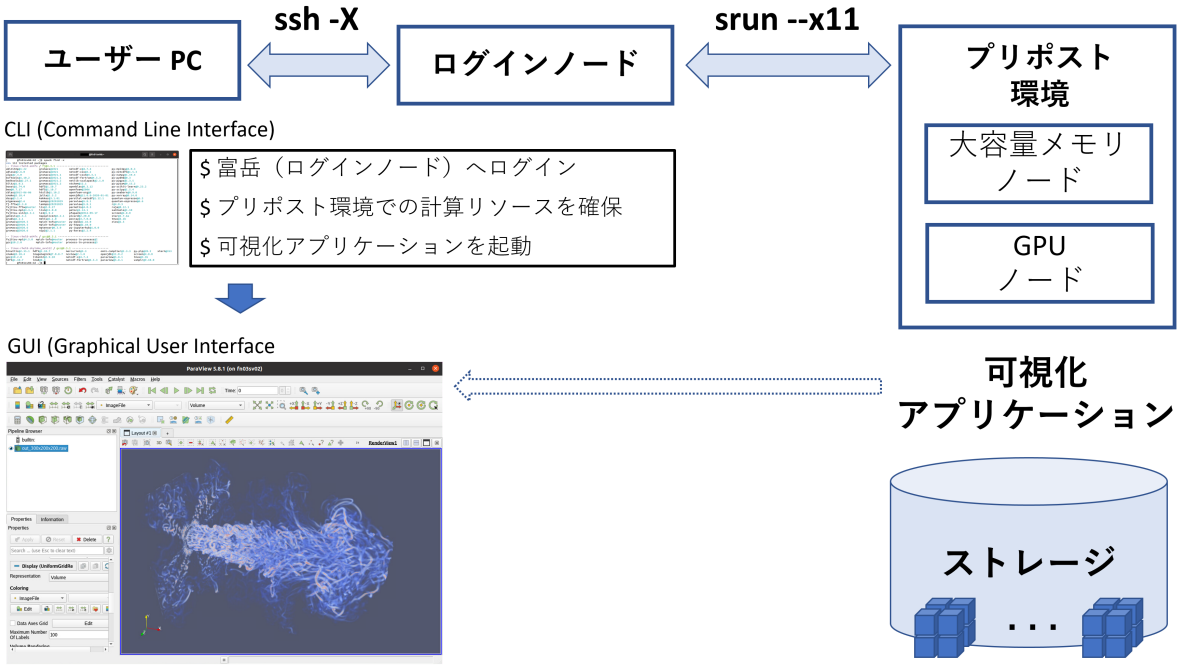
注釈: Spack の利用方法に関する詳細な説明は「富岳 Spack 利用ガイド」を参照してください。

2.3 GUI (Graphical User Interface)

GUI 機能を有する可視化アプリケーションを起動して対話的に利用する手法です。VNC（GNOME デスクトップ） や Open OnDemand（ Xfce デスクトップ）環境ではそのまま GUI 表示が可能ですが、PuTTY 等の CLI ターミナルから利用する場合は GUI 部分を（「X11 フォワーディング」機能を用いて）ユーザー PC で表示して利用することになります。その際に Windows OS では「VcXsrv」の様な X サーバーを起動していないと GUI 画面の表示が失敗しますので、事前に X サーバーソフトウェアのインストールと起動が必要になります。

注釈: GUI を用いた可視化アプリケーションの利用は Remote Desktop 上からの利用を推奨します。VNC（GNOME デスクトップ） や Open OnDemand の Xfce Remote Desktop 環境での GUI 実行は該当する章を参照してください。

2.3.1 プリポスト環境



プリポスト環境で起動する GUI 付き可視化アプリケーションの GUI をユーザー PC 側に送信し、操作を行う手順を示します。以下の手順は省略コマンドを利用しています。プリポスト環境の利用方法に関する詳細な説明は「プリポスト環境 利用手引書」を参照してください。

1.「-X」オプションを用いて富岳（ログイン・ノード）へログイン

```
[UserPC]$ ssh -X user@login.fugaku.r-ccs.riken.jp
```

2. 「--x11」オプションを用いてプリポスト環境でインテラクティブ・ジョブをスタート

注釈: PuTTY 利用の際は「Enable X11 forwarding」を有効にしてください。PuTTY を用いたログイン方法は「スーパーコンピュータ「富岳」 スタートアップガイド」を参照してください。

（GPU ノードでの実行例）

```
[Login]$ srun --x11 -p gpu1 -n 8 --mem 16000 --time=0:15:00 --pty bash -i
```

注釈: 必要に応じてジョブキュー (-p)、CPU 数 (-n)、メモリ量 (--mem)、GPU の有無 (--gpus=)、経過時間 (--time) を指定してください。プリポスト環境の利用方法に関する詳細な説明は「プリポスト環境 利用手引書」を参照してください。

---

3. GUI アプリケーションの起動

- ParaView

  GUI 版（Qt ライブラリ版）の ParaView は富岳 Open OnDemand 環境での利用を推奨しており、Spack では CLI 向け pvserver (Client/Server) と pvbatch (バッチジョブ）のみを提供しています。

- ImageMagick

```
[PrePost]$ . /vol0004/apps/oss/spack/share/spack/setup-env.sh

===========
[PrePost]$ spack find imagemagick

 -- linux-rhel8-cascadelake / gcc@13.2.0 --------------------------
 imagemagick@7.1.1-11
 ==> 1 installed package

[PrePost]$ spack load imagemagick@7.1.1-11%gcc@13.2.0

===========
[PrePost]$ cp $(spack location -i imagemagick)/share/doc/ImageMagick-7/images/wizard.
↪png .

[PrePost]$ display wizard.png
```

Panel	Display Window
	

- X ターミナル

```
[PrePost]$ . /vol0004/apps/oss/spack/share/spack/setup-env.sh

===========
[PrePost]$ spack find xterm

-- linux-rhel8-cascadelake / gcc@13.2.0 --------------------------
xterm@353
==> 1 installed package
```

(次のページに続く)

(前のページからの続き)

```
[PrePost]$ spack load xterm@353%gcc@13.2.0

===========
[PrePost]$ xterm
```

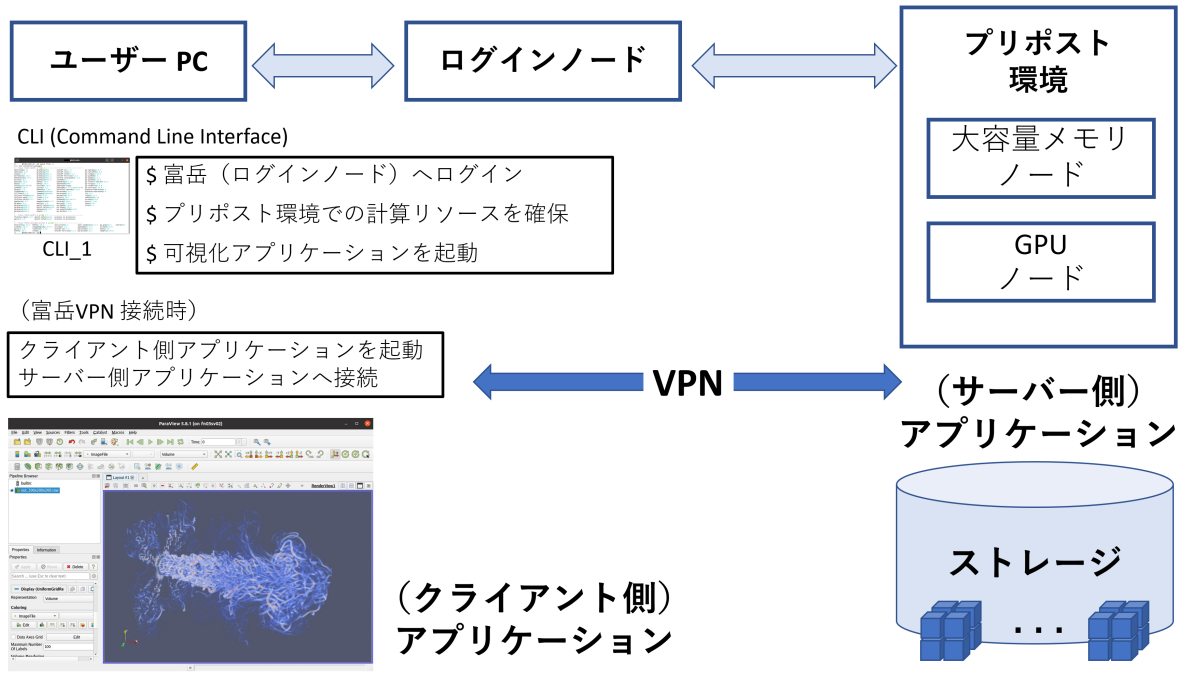
2.4 分散型（Client/Server）

ソケット通信を用いた（Client/Server）分散型可視化機能を有する可視化アプリケーションで（富岳 VPN 接続機能を用いて）プリポスト環境または計算ノードで動作する *Server* とユーザー側 PC で動作する *Client* を接続して利用する手法です。

- **ParaView** を利用した例では:
  - Server： **ParaView Server** (**pvserver**)
  - Client： **ParaView GUI Client** (**paraview**) は Server と同バージョンのものをユーザー側で用意する必要があります。
    - https://www.paraview.org/download/
  - ポート番号: **11111（デフォルト）**

注釈: 本章での表示例に利用しているデータ（CFD シミュレーション結果）は九州大学 小野謙二教授（旧 R-CCS 可視化技術研究チーム）からの提供によるものです。

2.4.1 プリポスト環境



プリポスト環境で Server 側アプリケーションを起動し、富岳 VPN 接続機能を用いてユーザー PC 側で動作する Client 側アプリケーションから接続して利用する手順を示します。以下の手順は省略コマンドを利用しています。プリポスト環境の利用方法に関する詳細な説明は「プリポスト環境 利用手引書」を参照してください。

1. 富岳（ログインノード）へログイン

```
[UserPC_1]$ ssh user@login.fugaku.r-ccs.riken.jp
```

2. プリポスト環境でインテラクティブ・ジョブをスタート

（GPU ノードでの実行例）

```
[Login]$ srun -p gpu1 -n 4 --mem 32000 --time=0:15:00 --pty bash -i
```

注釈: 必要に応じてジョブキュー (-p)、CPU 数 (-n)、メモリ量 (--mem)、GPU の有無 (--gpus=)、経過時間 (--time) を指定してください。プリポスト環境の利用方法に関する詳細な説明は「プリポスト環境 利用手引書」を参照してください。

---

3. Server 側アプリケーションの起動

（Server 側 ParaView「pvserver」を 4 並列での起動例）

```
============
Spack 0.21.0

[PrePost]$ . /vol0004/apps/oss/spack/share/spack/setup-env.sh
```

(次のページに続く)

(前のページからの続き)

```
[PrePost]$ spack --version
0.21.0

[PrePost]$ spack find paraview arch=linux-rhel8-cascadelake

==> No package matches the query: paraview arch=linux-rhel8-cascadelake

===========
Spack 0.19.0

[PrePost]$ . /vol0004/apps/oss/spack-v0.19/share/spack/setup-env.sh

[PrePost]$ spack --version
0.19.0

[PrePost]$ spack find paraview arch=linux-rhel8-cascadelake

-- linux-rhel8-cascadelake / gcc@12.2.0 -------------------------
paraview@5.10.1
==> 1 installed package

[PrePost]$ spack load paraview@5.10.1%gcc@12.2.0

===========
[PrePost]$ mpirun -n 4 pvserver

実行例：
Waiting for client...
Connection URL: cs://fn02sv02:11111
Accepting connection(s): fn02sv02:11111
```

注釈: 提供されているパッケージが複数の場合、バージョン、コンパイラ、アーキテクチャーやハッシュ値等を用いた選択を行う必要があります。Spack の利用方法に関する詳細な説明は「富岳 Spack 利用ガイド」を参照してください。

4. ユーザー PC 側アプリケーションの起動（VPN 接続時）

ユーザー端末と富岳間で VPN 接続が確立していることを前提としています。また、「3.」の「Connection URL: cs://」で表示されるホストネームに対応する VPN ホストネームを利用します。VPN 接続の利用方法については「富岳 VPN 接続設定手順」を参照してください。

Hostname	VPN Hostname
fn02sv01	pps01.pp.fugaku.r-ccs.riken.jp
fn02sv02	pps02.pp.fugaku.r-ccs.riken.jp
fn02sv03	pps03.pp.fugaku.r-ccs.riken.jp
fn02sv04	pps04.pp.fugaku.r-ccs.riken.jp
fn03sv01	pps05.pp.fugaku.r-ccs.riken.jp
fn03sv02	pps06.pp.fugaku.r-ccs.riken.jp
fn03sv03	pps07.pp.fugaku.r-ccs.riken.jp
fn03sv04	pps08.pp.fugaku.r-ccs.riken.jp
fn02sv05	ppm01.pp.fugaku.r-ccs.riken.jp
fn03sv05	ppm02.pp.fugaku.r-ccs.riken.jp
fn06sv01	ppm03.pp.fugaku.r-ccs.riken.jp

（Windows OS 等で ParaView Server 側情報をオプショとして渡す ParaView Client GUI の起動例）

「3.」でのホストネームは「**fn02sv02**」ですので、VPN ホストネームは「**pps02.pp.fugaku.r-ccs.riken.jp**」となります。

```
[UserPC_2]$ ./paraview --server-url=cs://pps02.pp.fugaku.r-ccs.riken.jp:11111
```

注釈: 追加の CLI コンソール「UserPC_2」を用いた起動例です。

（接続が成功すれば ParaView Server 側で「**Client connected.**」が表示されます）

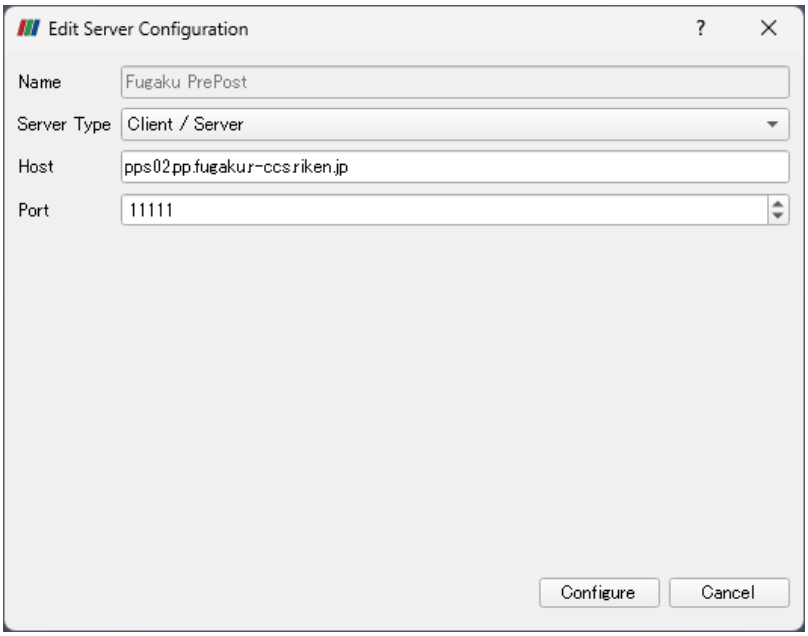
```
Waiting for client...
Connection URL: cs://fn02sv02:11111
Accepting connection(s): fn02sv02:11111
Client connected.
```

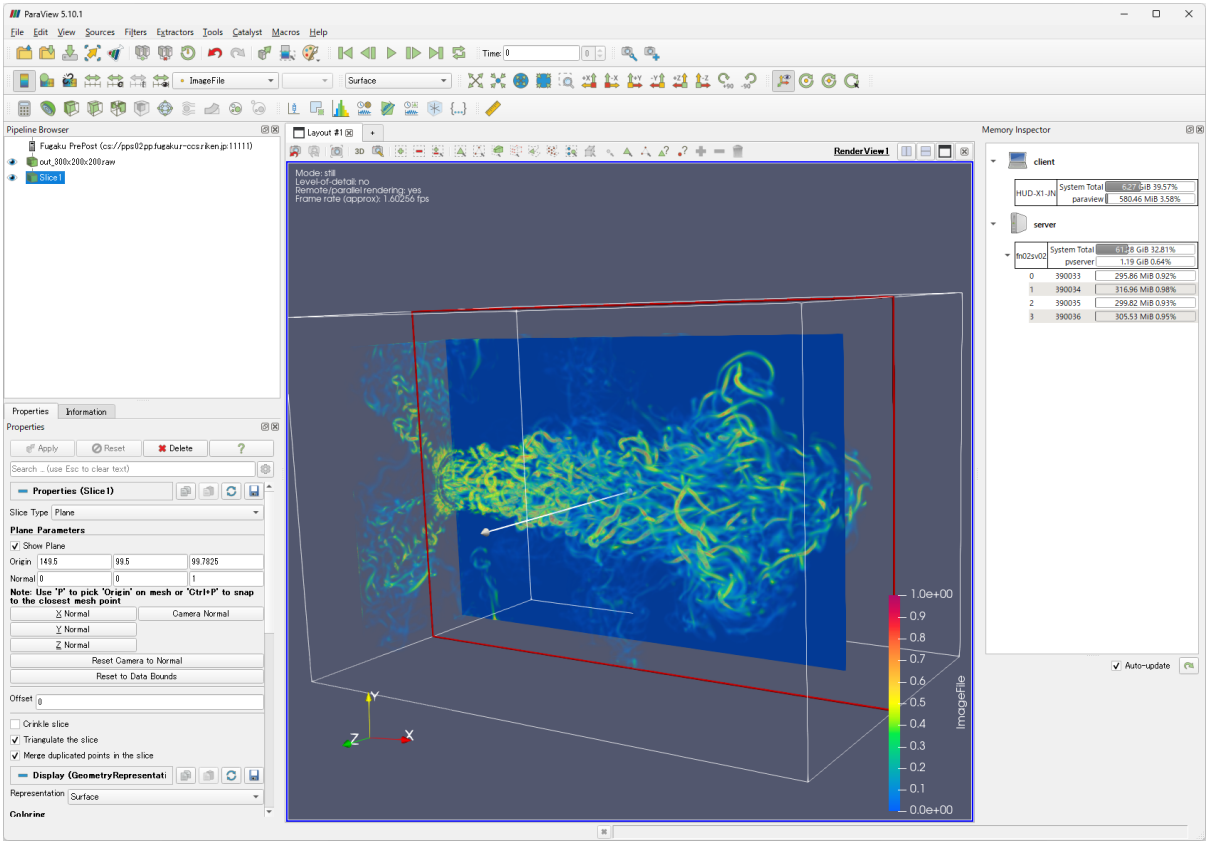
ParaView の GUI 上でサーバー情報の設定と接続を行う方法を示します。

タブの「File→Connect」または GUI「Connect」ボタンから「Choose Server Configuration」画面を開きます。「Add Server」で新規接続情報を作成します。好みの接続名を記入し、他の設定情報はデフォルトのを利用。

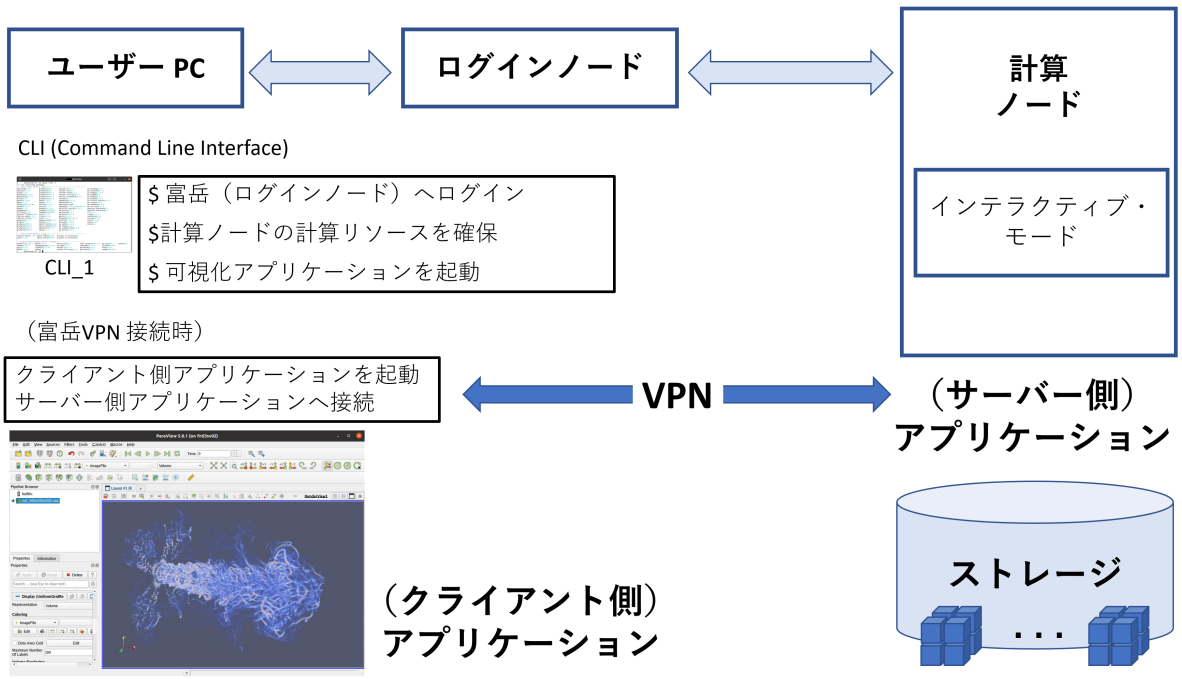
- Name: Fugaku
- Server Type: Client / Server
- Host: VPN ホストネーム（上記の例では *pps02.pp.fugaku.r-ccs.riken.jp*）
- Port 11111

Windows OS 上での ParaView Client から ４並列で動作している Server 側に接続して利用する表示例です。


Edit Server Configuration
Name
Fugaku PrePost
Server Type
Client / Server
Host
pps02pp.fugaku.r-ccs.riken.jp
Port
11111
Configure
Cancel


ParaView 5.10.1
Pipeline Browser
Properties
Information
Memory Inspector
client
server
RenderView1
Mode: still
Level-of-detail: no
Remote/parallel rendering: yes
Frame rate (approx): 1.60256 fps
1.0e+00
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0e+00
ImageFile

2.4.2 計算ノード



計算ノード上で Server 側アプリケーションを起動し（例として **paraview**）、富岳への VPN 接続機能を用いてユーザー PC 側で動作する Client 側アプリケーションから接続して利用する手順を示します。以下の手順は省略コマンドを利用しています。計算ノードの利用方法に関する詳細な説明は「利用手引書」を参照してください。

1. 富岳（ログイン・ノード）へログイン

```
[UserPC_1]$ ssh user@login.fugaku.r-ccs.riken.jp
```

2. Server 側アプリケーションを起動するためのリソースを確保

（計算ノードの４ノードをインテラクティブ・モードでの確保例）

```
[Login]$ pjsub -x PJM_LLIO_GFSCACHE=/vol0004 --interact -L "eco_state=2" -L "node=4" -
↪L "rscgrp=int" -L "elapse=0:15:00" --sparam "wait-time=600"
```

注釈: 必要に応じてジョブで使用するデータ領域の volume（上記例： -x PJM_LLIO_GFSCACHE=/vol0004）や省エネルギー実行モード（上記例： -L "eco_state=2"）等の設定を行ってください。また、ParaView サーバーを１ノードにつき１プロセス以上で起動した場合、大幅に時間がかかることが確認されているため、並列数分のノードを要求することを推奨します（上記例： -L "node=4"）。ジョブ実行に関する詳細な説明は「富岳 利用手引書」を参照してください。

---

3. Server 側アプリケーションの起動

Server 側 ParaView「paraview」を４並列での起動例。

```
[Compute]$ . /vol0004/apps/oss/spack/share/spack/setup-env.sh

===========
[Compute]$ spack find paraview

-- linux-rhel8-a64fx / fj@4.10.0 ---------------------------------
paraview@5.11.2  paraview@5.11.2  paraview@5.11.2
==> 3 installed packages

[Compute]$ spack load paraview +python +osmesa arch=linux-rhel8-a64fx

===========
[Compute]$ mpirun -n 4 pvserver

===========
 サーバー側の準備が整ったかを調べるには stdout ファイルを確認してください（ 別 CLI ターミナルより）。
[Compute]$ cat output.[ジョブ ID]/0/[実行 ID]/stdout.1.0

Waiting for client...
Connection URL: cs://[計算ノード ID]:11111
Accepting connection(s): [計算ノード ID]:11111
```

注釈: Spack で提供されているパッケージのアップデート等によりバージョンが上記と異なる可能性があります。Spack の利用方法に関する詳細な説明は「富岳 Spack 利用ガイド」を参照してください。

4. ユーザー側アプリケーションの起動（VPN 接続時）

ユーザー端末と富岳間で VPN 接続が確立していることを前提としています。また、「3.」の「Connection URL: cs://」で表示されるホストネームに「.cn.fugaku.r-ccs.riken.jp」を追加した VPN ホストネームを利用します。VPN 接続の利用方法については「富岳 VPN 接続設定手順」を参照してください。

Hostname	VPN Hostname
example	example.cn.fugaku.r-ccs.riken.jp

（Server 側情報をオプショとして渡す ParaView Client GUI の起動例）

例えば、ホストネームが「**a25-5012c**」ですので VPN ホストネームは「**a25-5012c.cn.fugaku.r-ccs.riken.jp**」を利用します。

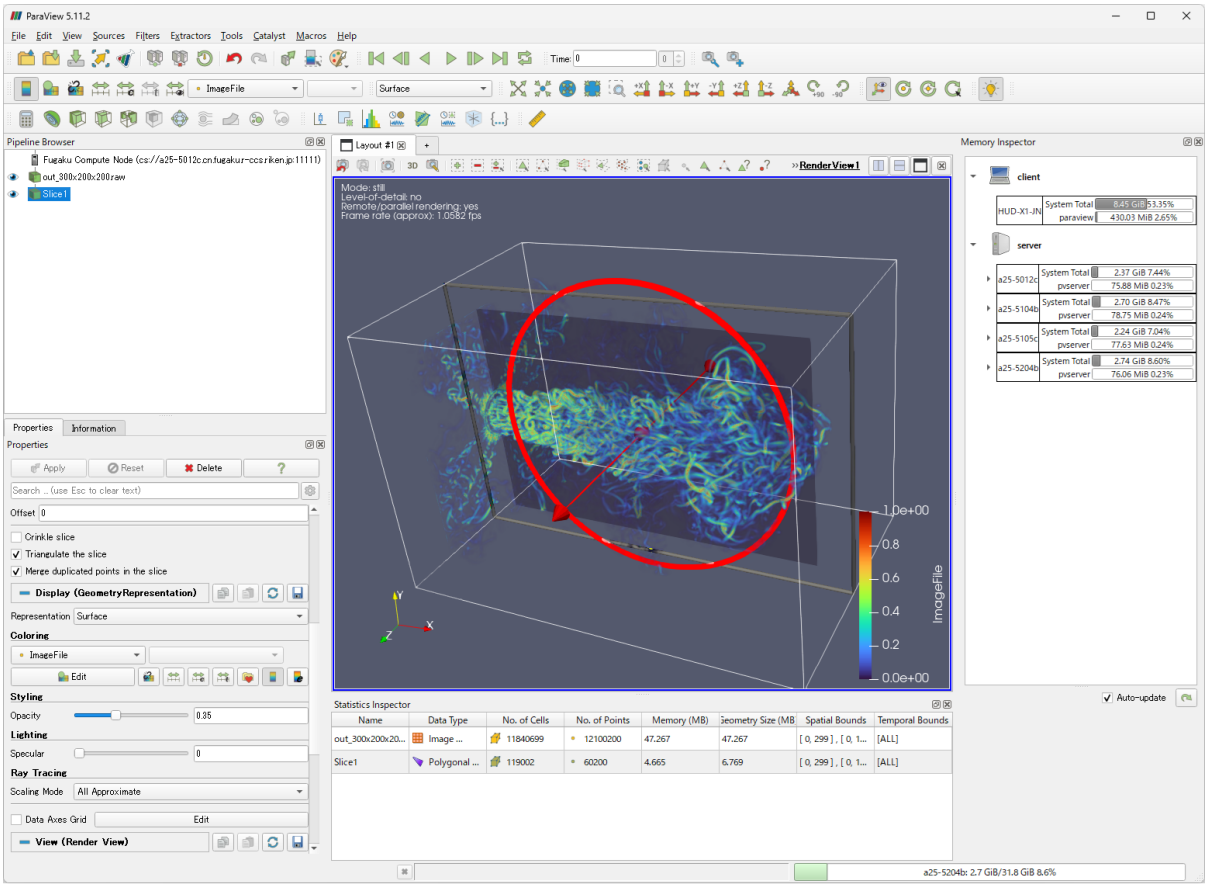
```
[UserPC_2]$ ./paraview --server-url=cs://a25-5012c.cn.fugaku.r-ccs.riken.jp:11111
```

注釈: 追加の CLI コンソール「UserPC_2」を用いた起動例です。

（接続が成功すれば Server 側で「**Client connected.**」が表示されます）

```
[Compute]$ cat output.[ジョブ ID]/0/[実行 ID]/stdout.1.0

 Waiting for client...
 Connection URL: cs://a25-5012c:11111
 Accepting connection(s): a25-5012c:11111
 Client connected.
```

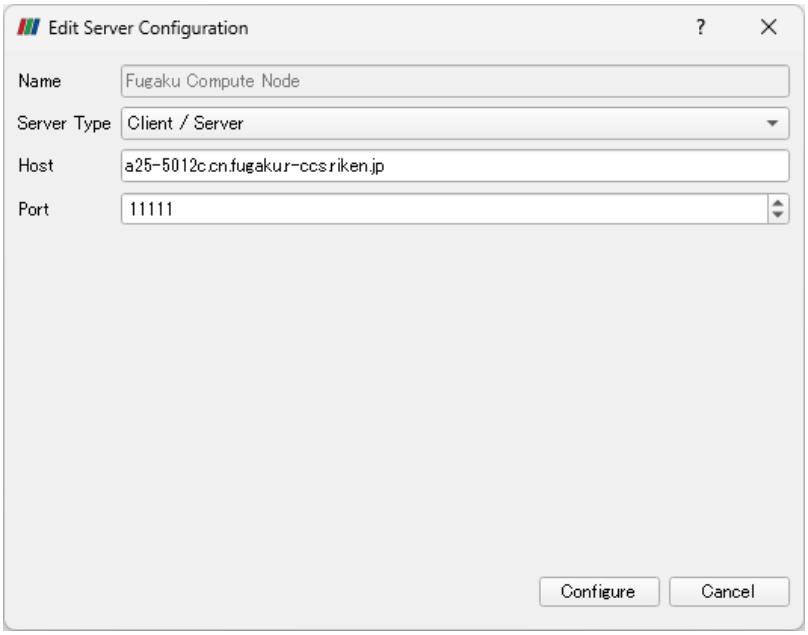
Windows OS 上での ParaView Client から ４並列で動作している Server 側に接続して利用する表示例です。



ParaView の GUI 上でサーバー情報の設定と接続を行う方法を示します。

タブの「File→Connect」または GUI「Connect」ボタンから「Choose Server Configuration」画面を開きます。「Add Server」で新規接続情報を作成します。好みの接続名を記入し、他の設定情報はデフォルトのを利用。

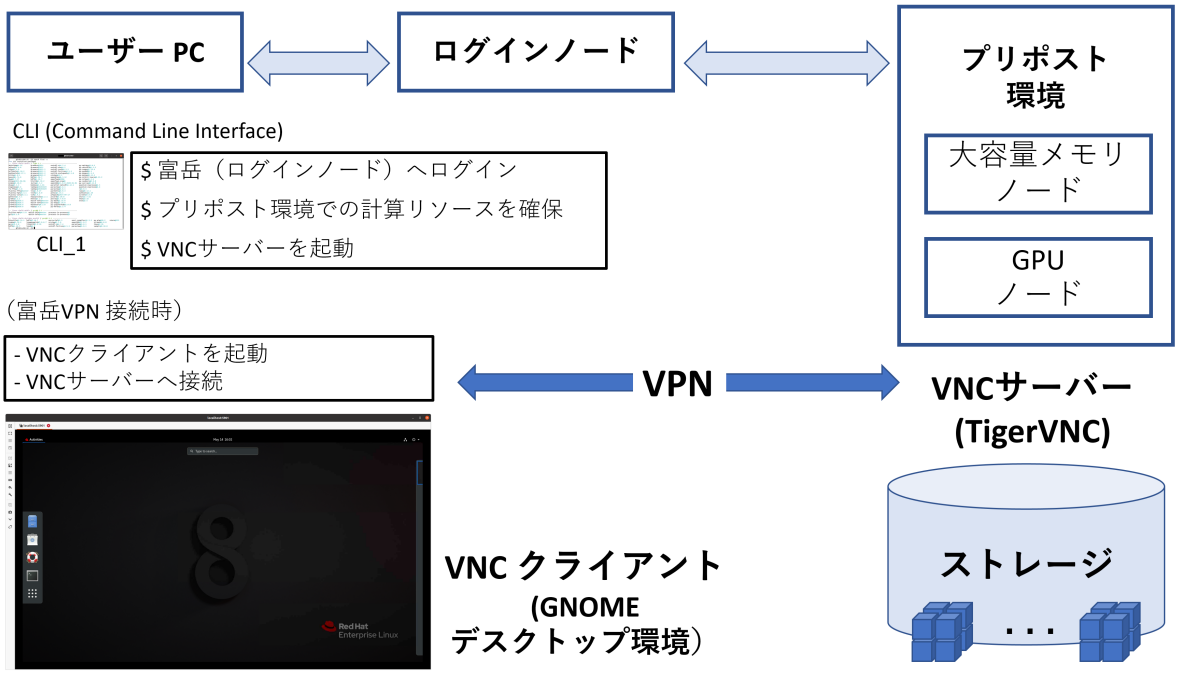
- Name: Fugaku Compute Node
- Server Type: Client / Server
- Host: VPN ホストネーム（上記の例では *a25-5012c.cn.fugaku.r-ccs.riken.jp*）
- Port 11111



2.5 VNC（GNOME デスクトップ）

VNC（Virtual Network Computing）を用いたリモートデスクトップ（GNOME デスクトップ）を利用する手法です。リモートデスクトップは Open OnDemand 環境でより簡単な設定で利用できますので、そちらを推奨します。

2.5.1 プリポスト環境



- VNC サーバー
  - TigerVNC

- VNC クライアント
  - 例：Remmina (Ubuntu 20)
- ポート番号: **5901（デフォルト）**

注釈: ユーザー PC 側の VNC クライアントとして Ubuntu の Remmina を利用した実行例を紹介していますが、他 OS（Windows や MacOS）の VNC クライアントからでも利用可能です。Anaconda をインストールしている環境では VNC 接続に不具合が発生する可能性が確認されています。なお、Anaconda などご自身でインストールされるソフトウェアについては、ライセンス規定などに十分ご注意ください（理化学研究所での Anaconda 利用は有償ライセンスが必要になります）。

プリポスト環境で VNC サーバを起動し、富岳 VPN 接続機能を用いてユーザー PC 側の VNC クライアントからプリポスト環境の GNOME デスクトップを利用する手順を示します。以下の手順は省略コマンドを利用しています。プリポスト環境の利用方法に関する詳細な説明は「プリポスト環境 利用手引書」を参照してください。

1. 富岳（ログイン・ノード）へログイン

```
[UserPC_1]$ ssh user@login.fugaku.r-ccs.riken.jp
```

2. プリポスト環境でインテラクティブ・ジョブをスタート

（GPU ノードでの実行例）

```
[Login]$ srun -p gpu1 -n 16 --mem 64000 --gpus=1 --time=0:15:00 --pty bash -i
```

注釈: 必要に応じてジョブキュー (-p)、CPU 数 (-n)、メモリ量 (--mem)、GPU の有無 (--gpus=)、経過時間 (--time) を指定してください。プリポスト環境の利用方法に関する詳細な説明は「プリポスト環境 利用手引書」を参照してください。

---

3. 初期設定（VNC サーバーを初めて起動する場合のみ必要です）

（初めての起動の際には VNC 接続パスワードやデスクトップ起動の設定などの前準備が必要となります）

- ア）VNC 接続パスワードの設定 (vncserver を起動します)

```
[PrePost]$ vncserver

You will require a password to access your desktops.

Password:
Verify:
Would you like to enter a view-only password (y/n)? n
A view-only password is not used

New 'fn02sv03:1 (user)' desktop is fn02sv03:1
```

(次のページに続く)

(前のページからの続き)

```
Creating default startup script /home/group/user/.vnc/xstartup
Creating default config /home/group/user/.vnc/config
Starting applications specified in /home/group/user/.vnc/xstartup
Log file is /home/group/user/.vnc/fn02sv03:1.log
```

- イ）vncserver を一旦終了します

```
[PrePost]$ vncserver -kill :1
Killing Xvnc process ID プロセス ID
```

- ウ）デスクトップ起動の設定

ア）で作成された「xstartup」の最後尾に GNOME デスクトップ起動コマンド（gnome-session &）を追加します。

```
#!/bin/sh

unset SESSION_MANAGER
unset DBUS_SESSION_BUS_ADDRESS
/etc/X11/xinit/xinitrc
# Assume either Gnome will be started by default when installed
# We want to kill the session automatically in this case when user logs out. In case␣
↪you modify
# /etc/X11/xinit/Xclients or ~/.Xclients yourself to achieve a different result, then␣
↪you should
# be responsible to modify below code to avoid that your session will be␣
↪automatically killed
if [ -e /usr/bin/gnome-session ]; then
    vncserver -kill $DISPLAY
fi
gnome-session &
```

4. VNC サーバーの起動

```
[PrePost]$ vncserver

New 'fn02sv03:1 (user)' desktop is fn02sv03:1

Starting applications specified in /home/group/user/.vnc/xstartup
Log file is /home/group/user/.vnc/fn02sv03:1.log
```

解像度（スクリーンサイズ）なども引数として実行時に指定できます。（例：[PrePost]$ vncserver -geometry 1920x1080）

```
[PrePost]$ vncserver -help
```

(次のページに続く)

(前のページからの続き)

```
usage: vncserver [:<number>] [-name <desktop-name>] [-depth <depth>]
                 [-geometry <width>x<height>]
                 [-pixelformat rgbNNN|bgrNNN]
                 [-fp <font-path>]
                 [-cc <visual>]
                 [-fg]
                 [-autokill]
                 [-noxstartup]
                 [-xstartup <file>]
                 <Xvnc-options>...
```

VNC サーバーが正しく終了しなかったりした場合にロックファイルが残っているとワーニング・メッセージが表示され、連番的に次の VNC サーバーが起動されます。

```
[PrePost]$ vncserver

Warning: fn02sv02:1 is taken because of /tmp/.X1-lock
Remove this file if there is no X server fn02sv03:1

New 'fn02sv03:2 (user)' desktop is fn02sv03:2

Starting applications specified in /home/group/user/.vnc/xstartup
Log file is /home/group/user/.vnc/fn02sv03:2.log
```

ロックファイルを削除することで「:1」から VNC サーバーが起動されるようになります。他ユーザーの VNC セッションで作成されたファイルが混合している可能性もあるため、事前にファイル所有者の確認を推奨する。

```
[PrePost]$ rm /tmp/.X?-lock
[PrePost]$ rm /tmp/.X11-unix/X?
```

5. VNC クライアントからの接続（VPN 接続時）

ユーザー端末と富岳間で VPN 接続が確立していることを前提としています。また、「４.」の「desktop is **ホストネーム：X ディスプレイ番号**」に表示されるホストネームに対応する **VPN ホストネーム** と X ディスプレイ番号に対応する **ポート番号** を利用します。VPN 接続の利用方法については「富岳 VPN 接続設定手順」を参照してください。

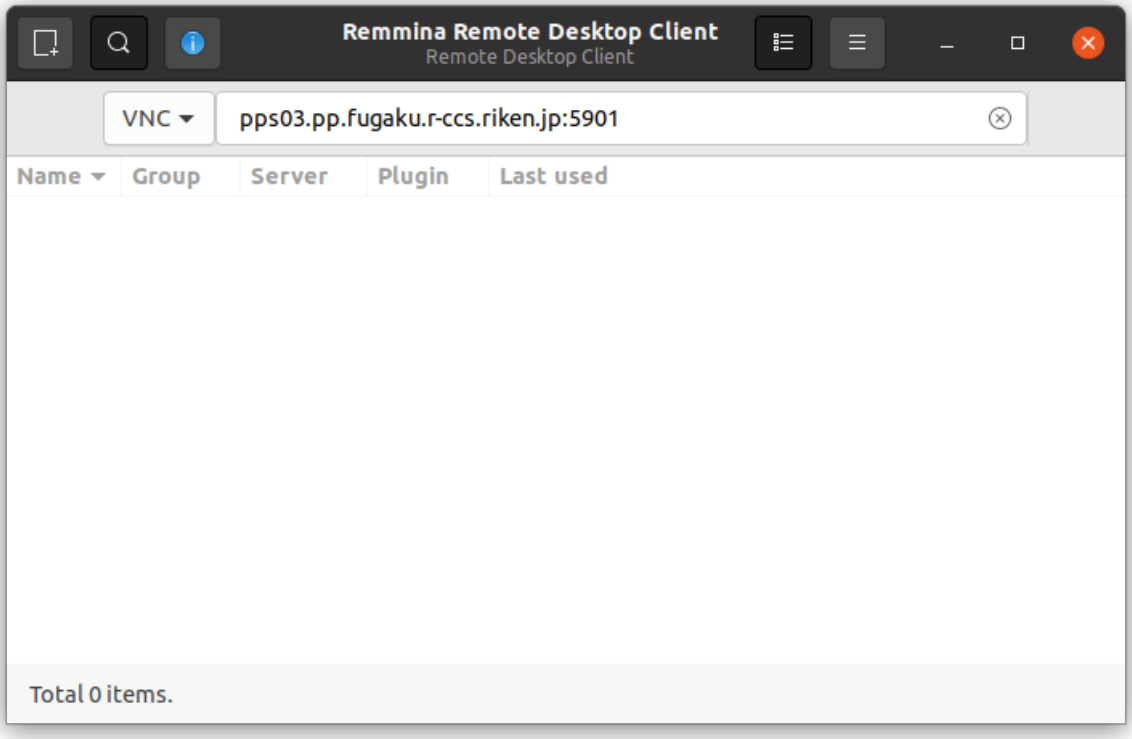
Hostname	VPN Hostname
fn02sv01	pps01.pp.fugaku.r-ccs.riken.jp
fn02sv02	pps02.pp.fugaku.r-ccs.riken.jp
fn02sv03	pps03.pp.fugaku.r-ccs.riken.jp
fn02sv04	pps04.pp.fugaku.r-ccs.riken.jp
fn03sv01	pps05.pp.fugaku.r-ccs.riken.jp
fn03sv02	pps06.pp.fugaku.r-ccs.riken.jp
fn03sv03	pps07.pp.fugaku.r-ccs.riken.jp
fn03sv04	pps08.pp.fugaku.r-ccs.riken.jp
fn02sv05	ppm01.pp.fugaku.r-ccs.riken.jp
fn03sv05	ppm02.pp.fugaku.r-ccs.riken.jp
fn06sv01	ppm03.pp.fugaku.r-ccs.riken.jp

X Display #	ポート番号
:1	5901
:2	5902
:3	5903
...	...

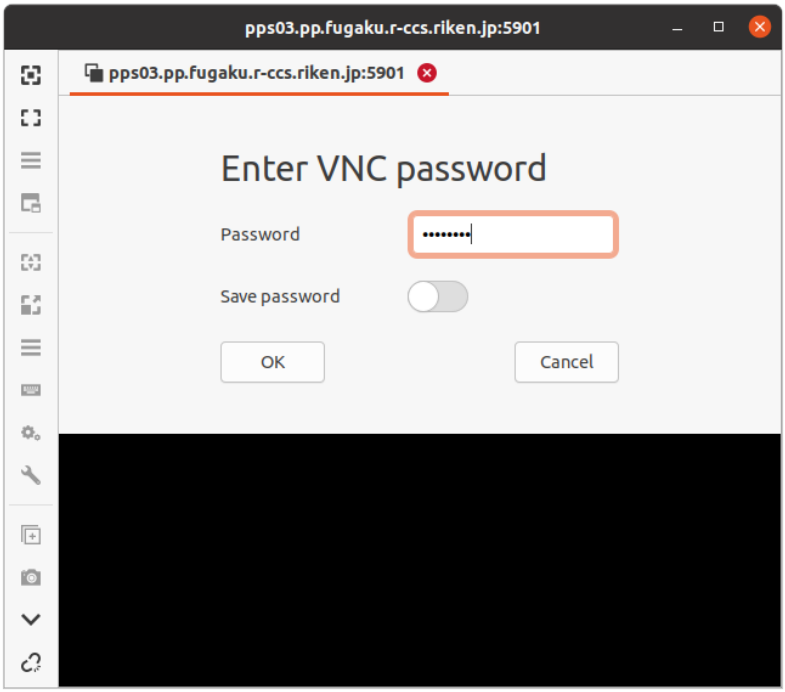
VNC クライアントより「**VPN ホストネーム:ポート番号**」へ接続します。

「4.」の例では「**fn02sv03:1**」であるため「**pps03.pp.fugaku.r-ccs.riken.jp:5901**」を利用します。

例：Ubuntu の Remmina を利用



接続に成功すると VNC 接続のパスワード画面が開きます。



パスワード認証が終了すると GNOME デスクトップ画面が開きます。（GNOME デスクトップを初めて起動した場合の画面）

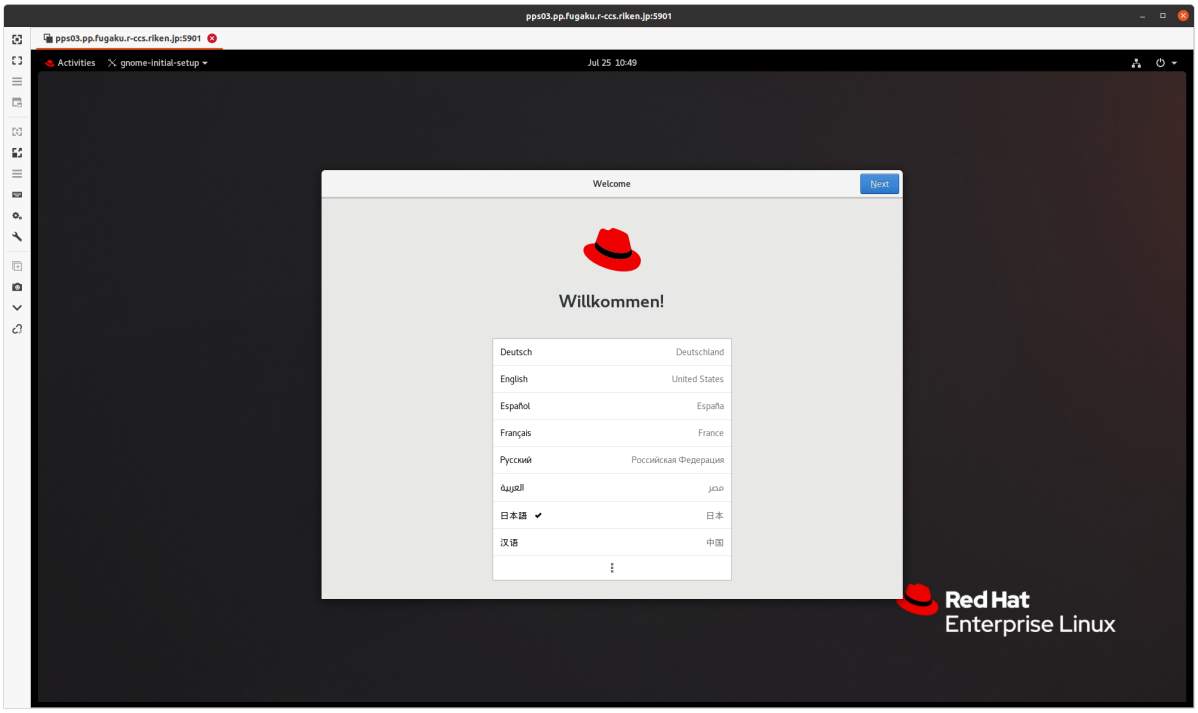
6. VNC サーバーの終了

起動中の VNC サーバーを調べるには「-list」引数を vncserver に渡します。

```
[PrePost]$ vncserver -list

TigerVNC server sessions:

X DISPLAY #  PROCESS ID
:1           プロセス ID_1
:2           プロセス ID_2
```

特定の VNC サーバーを終了するには「X DISPLAY #」番号を「vncserver -kill」に引数として渡します。

```
[PrePost]$ vncserver -kill :1
Killing Xvnc process ID プロセス ID_1
[PrePost]$ vncserver -kill :2
Killing Xvnc process ID プロセス ID_2
```


pps03.pp.fugaku.r-ccs.riken.jp:5901
pps03.pp.fugaku.r-ccs.riken.jp:5901
Activities
gnome-initial-setup
Jul 25 10:49
Welcome
Next
Willkommen!
Deutsch
Deutschland
English
United States
Español
España
Français
France
Русский
Российская Федерация
日本語
日本
汉语
中国
Red Hat
Enterprise Linux

第3章　AVS/Express

ISV 汎用可視化アプリケーションである「AVS/Express」をプリポスト環境またはユーザー PC で起動するための手順を記載しています。富岳側でのインストール先（Install Dir.）、実行ファイルの場所（Application Dir.）、ユーザー PC へのインストール向けダウンロードファイルの場所（Download Dir.）、マニュアル等のヘルプ情報の場所（Help Dir.）とライセンス・サーバー情報は以下の通りです。

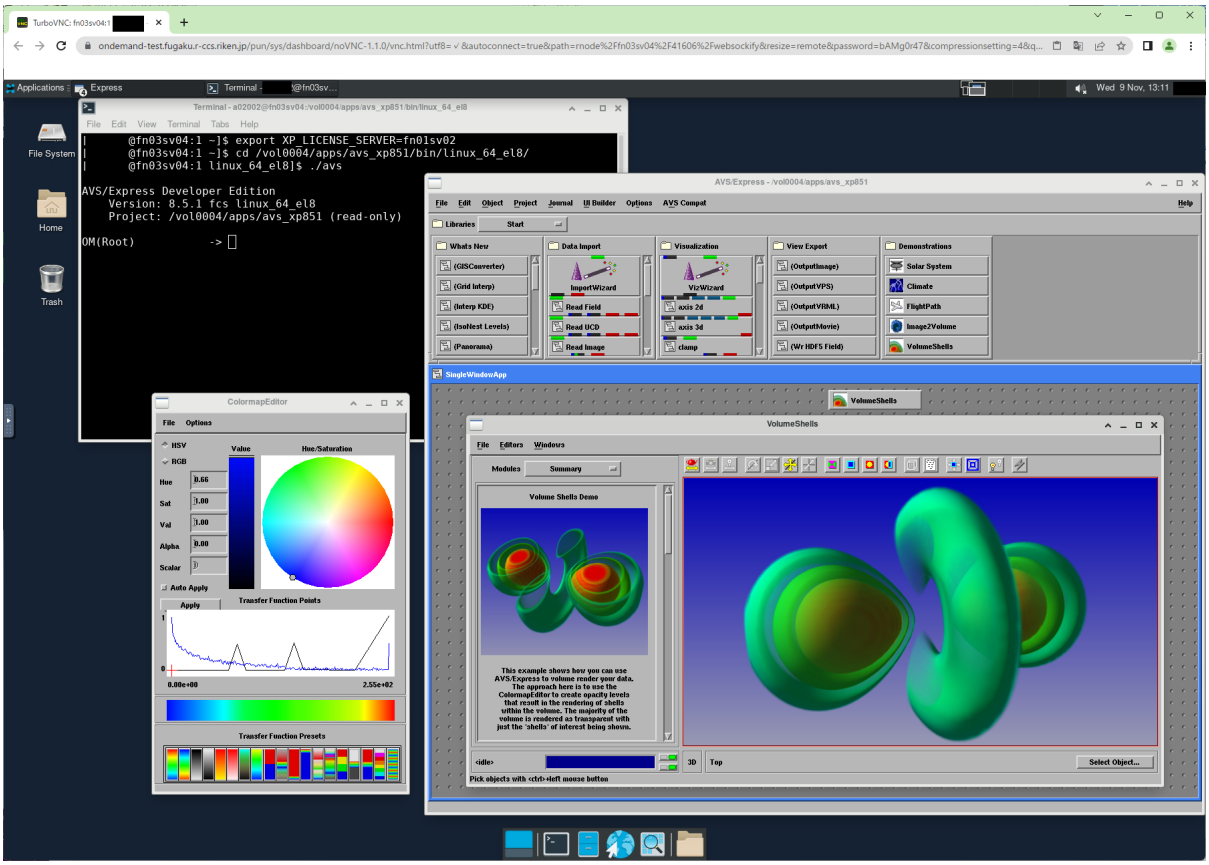
AVS/Express	Developer 8.5.1
Install Dir.	/vol0004/apps/avs_xp851
Application Dir.	/vol0004/apps/avs_xp851/bin/linux_64/el8
Download Dir.	/vol0004/apps/avs_xp851/download
Help Dir.	/vol0004/apps/avs_xp851/runtime/help
License Server	fn01sv02 (login2.fugaku.r-ccs.riken.jp)

3.1 Open OnDemand

Open OnDemand のリモートデスクトップ（Xfce デスクトップ環境）上で AVS/Express の起動例を以下に示しています。

```
[PrePost]$ avs
```

注釈: Open Ondemand の利用方法に関する詳細な説明は「富岳 Open OnDemand ガイド」を参照してください。また、「Fugaku Ondemand」ポータルにもにマニュアルへのリンクがございます。



3.2 ローカル PC（富岳 VPN 接続時）

1． インストール

章の始めに記載されているダウンロードフォルダ（Download Dir.）から OS に合ったファイルをローカル環境にコピーして、AVS/Express のインストールを行ってください。プリポスト環境では「RHEL8」がインストールされています。

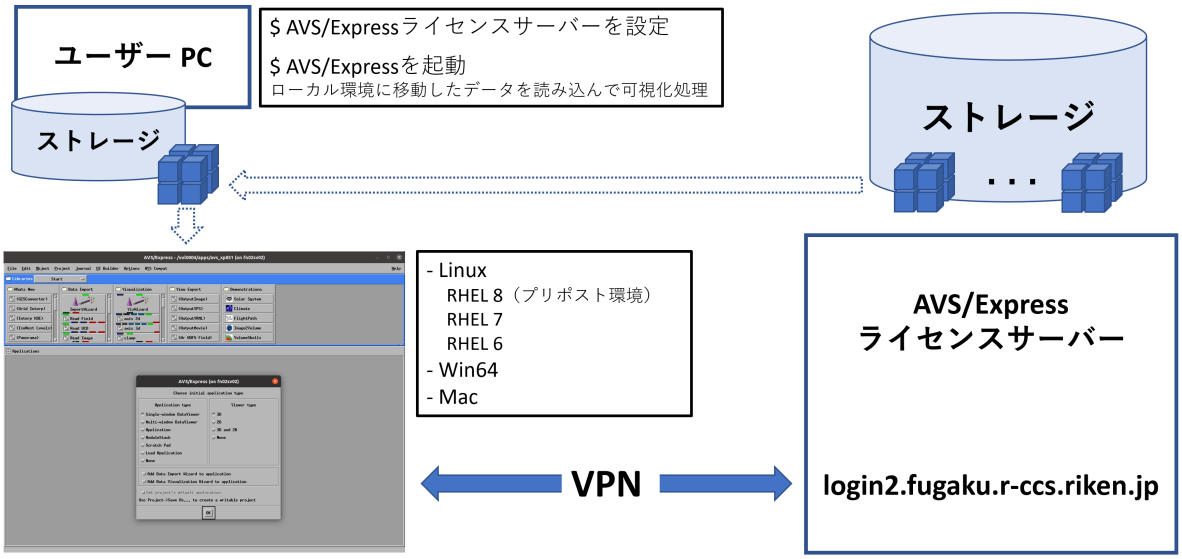
OS	Folder	File Package
RHEL8	Linux/linux_64_el8	express85.tgz
RHEL7	Linux/linux_64_el7	express85.tgz
RHEL6	Linux/linux_64_el6	express85.tgz
Windows	Win64	Win64.zip
OSX	Mac	darwin_64.zip

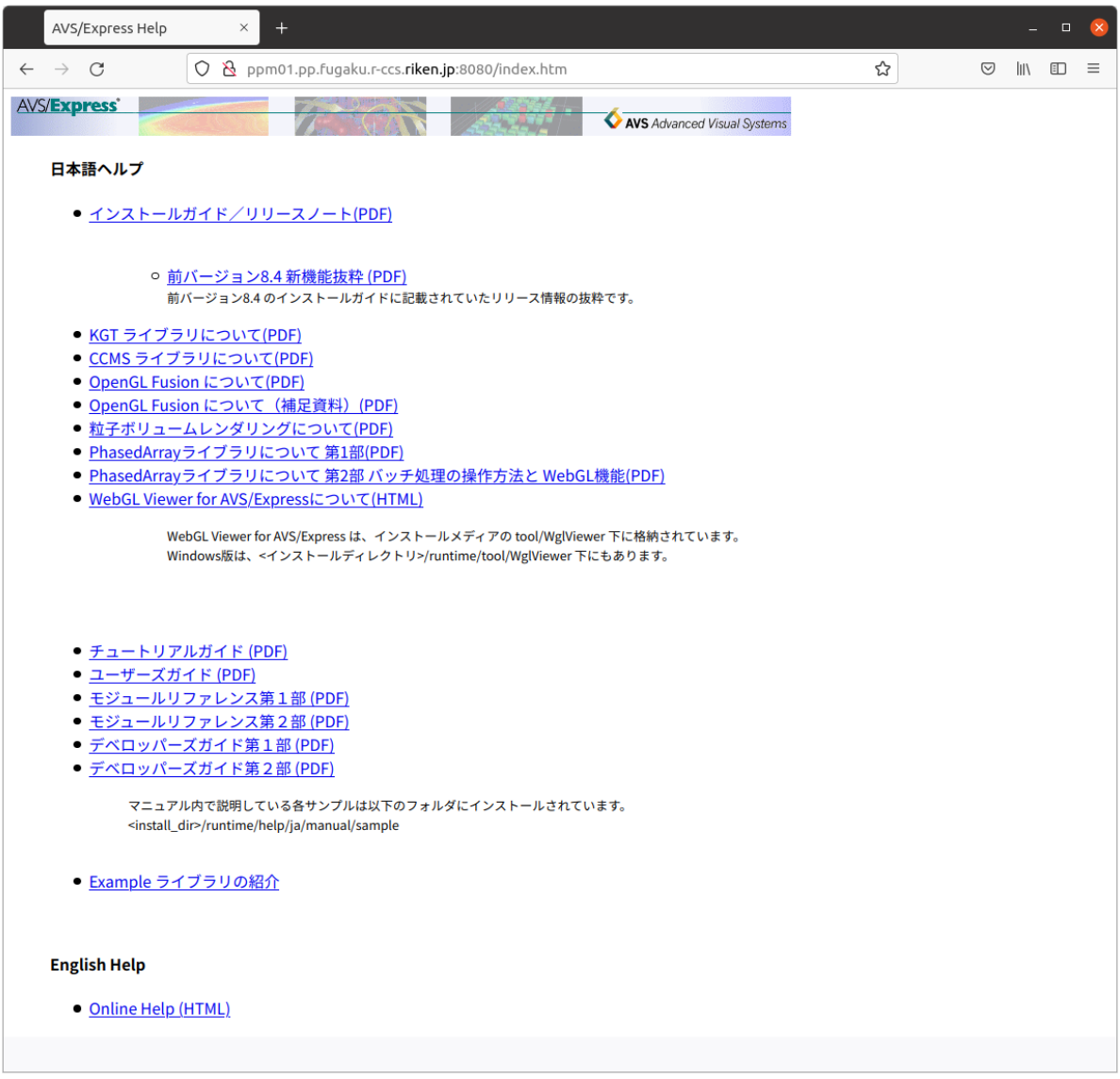
AVS/Express の「インストールガイド」は Linux 向けパッケージ（express85.tgz）解凍後の「runtime/help」フォルダ、又は章の始めに記載されているヘルプ情報フォルダ（Help Dir.）にありますので、Web ブラウザで「index.htm」を開いてください。「チュートリアルガイド」や「ユーザーズガイド」もそこからアクセス可能です。

（Firefox を Web ブラウザーとして利用）

2． 環境変数（AVS/Express ライセンス・サーバー）の設定

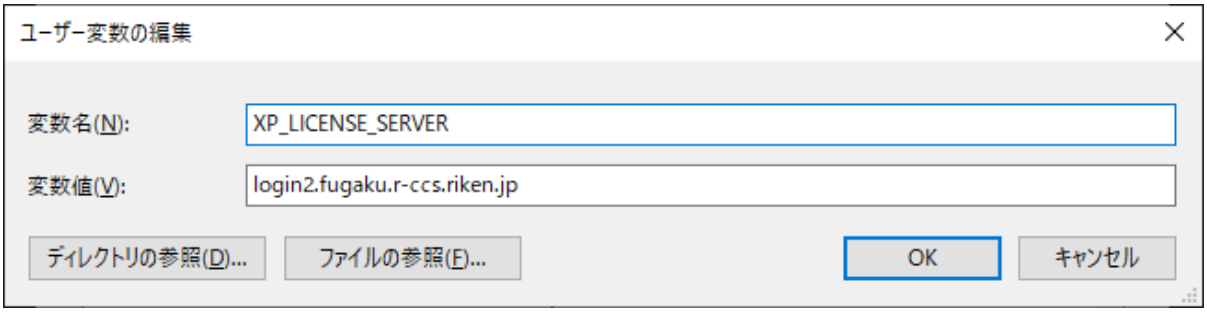
ア）Linux


ユーザー PC
ストレージ
$ AVS/Expressライセンスサーバーを設定
$ AVS/Expressを起動
ローカル環境に移動したデータを読み込んで可視化処理
ストレージ
- Linux
RHEL 8（プリポスト環境）
RHEL 7
RHEL 6
- Win64
- Mac
VPN
AVS/Express
ライセンスサーバー
login2.fugaku.r-ccs.riken.jp


AVS/Express Help
ppm01.pp.fugaku.r-ccs.riken.jp:8080/index.htm
AVS/Express
AVS Advanced Visual Systems
日本語ヘルプ
インストールガイド／リリースノート(PDF)
前バージョン8.4 新機能抜粋 (PDF)
前バージョン8.4 のインストールガイドに記載されていたリリース情報の抜粋です。
KGT ライブラリについて(PDF)
CCMS ライブラリについて(PDF)
OpenGL Fusion について(PDF)
OpenGL Fusion について（補足資料）(PDF)
粒子ボリュームレンダリングについて(PDF)
PhasedArrayライブラリについて 第1部(PDF)
PhasedArrayライブラリについて 第2部 バッチ処理の操作方法と WebGL機能(PDF)
WebGL Viewer for AVS/Expressについて(HTML)
WebGL Viewer for AVS/Express は、インストールメディアの tool/WglViewer 下に格納されています。
Windows版は、<インストールディレクトリ>/runtime/tool/WglViewer 下にもあります。
チュートリアルガイド (PDF)
ユーザーズガイド (PDF)
モジュールリファレンス第１部 (PDF)
モジュールリファレンス第２部 (PDF)
デベロッパーズガイド第１部 (PDF)
デベロッパーズガイド第２部 (PDF)
マニュアル内で説明している各サンプルは以下のフォルダにインストールされています。
<install_dir>/runtime/help/ja/manual/sample
Example ライブラリの紹介
English Help
Online Help (HTML)

```
[UserPC]$ export XP_LICENSE_SERVER=login2.fugaku.r-ccs.riken.jp
```

イ）Windows



3． 富岳 VPN 接続

ライセンスサーバーへの接続にはユーザー端末と富岳間で VPN 接続が確立していることが必要です。VPN 接続の利用方法については「富岳 VPN 接続設定手順」を参照してください。

Hostname	VPN Hostname
fn01sv02	login2.fugaku.r-ccs.riken.jp

4． AVS/Express の起動

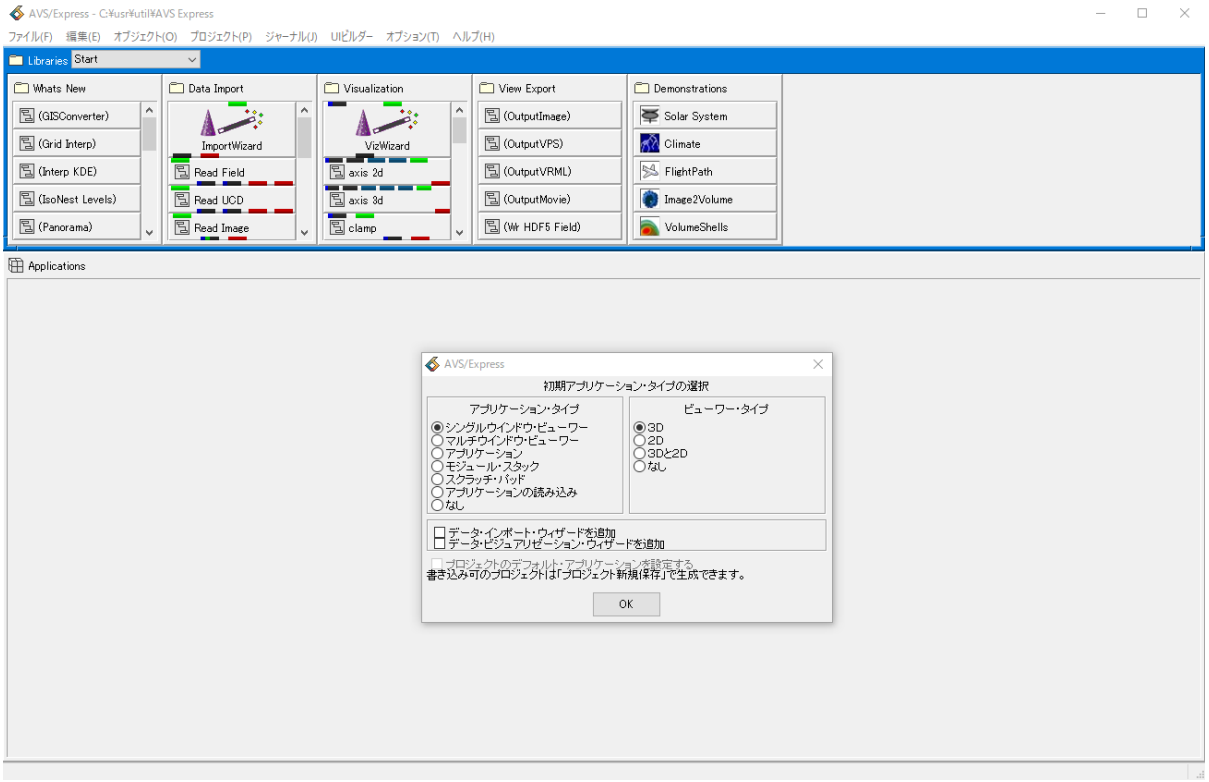
ア）Linux

(RHEL8 での実行例)

```
[UserPC]$ cd bin/linux_64_el8/
[UserPC]$ ./express.static -mavs
```

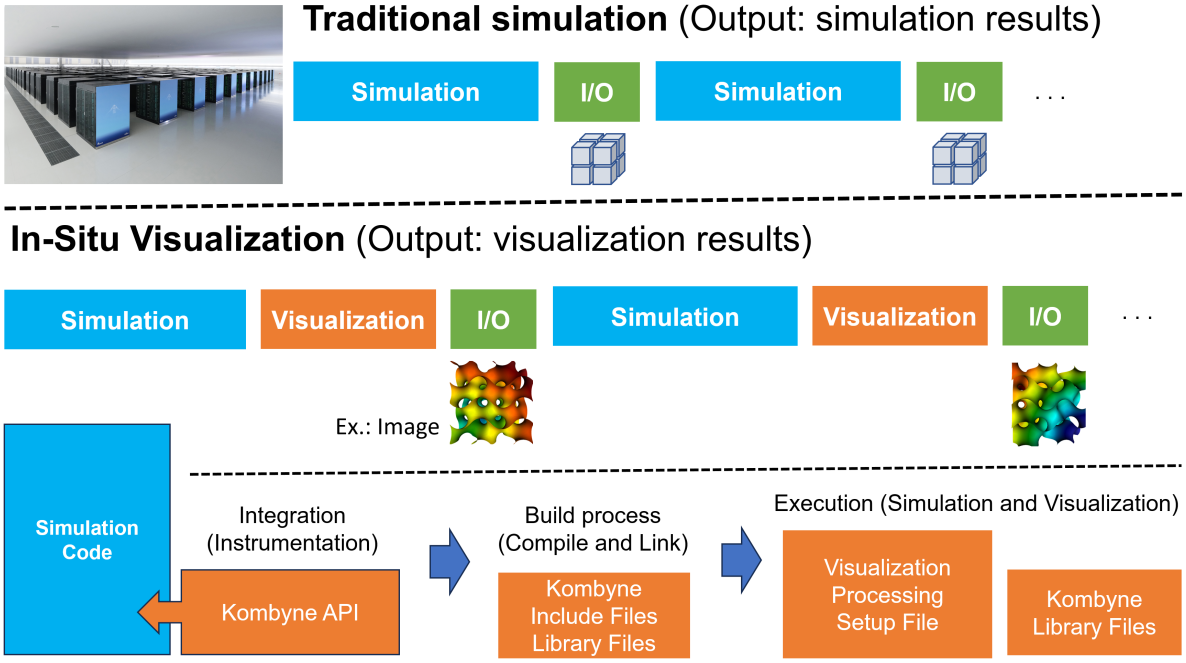
イ）Windows

Windows10 では AVS/Express のアイコン（内容は「インストール先￥bin￥pc16_64￥express.exe -mavs」）をクリックすることで起動できます。


AVS/Express - C:¥usr¥util¥AVS Express
ファイル(F) 編集(E) オブジェクト(O) プロジェクト(P) ジャーナル(J) UIビルダー オプション(T) ヘルプ(H)
Libraries Start
Whats New
(GISConverter)
(Grid Interp)
(Interp KDE)
(IsoNest Levels)
(Panorama)
Data Import
ImportWizard
Read Field
Read UCD
Read Image
Visualization
VizWizard
axis 2d
axis 3d
clamp
View Export
(OutputImage)
(OutputVPS)
(OutputVRML)
(OutputMovie)
(Wr HDF5 Field)
Demonstrations
Solar System
Climate
FlightPath
Image2Volume
VolumeShells
Applications
AVS/Express
初期アプリケーション・タイプの選択
アプリケーション・タイプ
シングルウインドウ・ビューワー
マルチウインドウ・ビューワー
アプリケーション
モジュール・スタック
スクラッチ・パッド
アプリケーションの読み込み
なし
ビューワー・タイプ
3D
2D
3Dと2D
なし
データ・インポート・ウィザードを追加
データ・ビジュアリゼーション・ウィザードを追加
プロジェクトのデフォルト・アプリケーションを設定する
書き込み可のプロジェクトは「プロジェクト新規保存」で生成できます。
OK

第4章　Kombyne

Kombyne はシミュレーションと同時に可視化処理を行う「In-Situ 可視化」向けの ISV 商用ツールで、試験的に導入されています。従来のシミュレーションで事後解析向けにストレージに出力されるシミュレーション結果に可視化処理を施し、その結果を出力する手法になります。その為、I/O コストを大幅に削減できると期待されている手法であり、計算科学ロードマップ 2023 (https://cs-forum.github.io/roadmap-2023/) でも言及されています。In-Situ 可視化ツールは Kombyne 以外にも多数の OSS（例：Catalyst, Libsim, KVS）も存在しますが、Kombyne では（富岳サポートサイト経由で）開発元からの技術的サポートが期待できます。



注釈: 「Visualization」処理では上図のように画像（PNG フォーマット）を出力するレンダリング処理以外、等値面を構成するポリゴンデータ等の中間データ抽出処理も含まれます。

In-Situ 可視化ではシミュレーションと連動して可視化処理を行うため、上記の図で示されているようにシミュレーション・コードに Kombyne が提供する API の組込作業が必要となります。現時点で Kombyne API が対応しているプログラミング言語は以下の通りです。

- C/C++
- FORTRAN
- Julia
- Python

インストール先（Install Dir.）は以下の通りです。サブディレクトリに実装及び実行に必要なライブラリ (lib64) とインクルード・ファイル (include)、またサンプル例 (examples) とマニュアル (docs) が置かれています。

Kombyne	Version 1.5
Install Dir.	/vol0004/apps/opt/kombyne
Library Dir.	/lib64
Include Dir.	/include
Examples	/examples
Manual	/docs/Kombyne.pdf

利用する際には以下の様に環境変数 (PATH と LD_LIBRARY_PATH) を事前に設定しておくと便利です。

```
[Compute]$ export PATH=/vol0004/apps/opt/kombyne/bin:$PATH
[Compute]$ export LD_LIBRARY_PATH=/vol0004/apps/opt/kombyne/lib64:$LD_LIBRARY_PATH
```

Kombyne API を組込んだ C/C++ 又は FORTRAN コードのビルド向けコンパイラ・フラグとリンカー・フラグは **/bin/kombyne-config** で得られます。

```
[Compute]$ kombyne-config --c-flags
[Compute]$ kombyne-config --c-libs

[Compute]$ kombyne-config --cxx-flags
[Compute]$ kombyne-config --cxx-libs

[Compute]$ kombyne-config --fortran-flags
[Compute]$ kombyne-config --fortran-libs
```

以下は FORTAN コード向けの出力結果です。

```
[Compute]$ kombyne-config --fortran-flags
-DMPICH_SKIP_MPICXX -DOMPI_SKIP_MPICXX -D_MPICC_H -I/vol0004/apps/opt/kombyne/
↪thirdparty/ZeroMQ/include -I/vol0004/apps/opt/kombyne/include/kombyne -I/vol0004/
↪apps/opt/kombyne/thirdparty/Conduit/include -I/vol0004/apps/opt/kombyne/thirdparty/
↪Conduit/include/conduit
```

（次のページに続く）

(前のページからの続き)

```
[Compute]$ kombyne-config --fortran-libs
-Wl,-rpath,/vol0004/apps/opt/kombyne/lib64:/vol0004/apps/opt/kombyne/thirdparty/
↪ZeroMQ/lib64:/vol0004/apps/opt/kombyne/thirdparty/Conduit/lib:/vol0004/apps/opt/
↪kombyne/thirdparty/ADIOS2/lib64 /vol0004/apps/opt/kombyne/lib64/libkombyne.so /
↪vol0004/apps/opt/kombyne/thirdparty/ZeroMQ/lib64/libzmq.so.5.2.2 /vol0004/apps/opt/
↪kombyne/thirdparty/Conduit/lib/libconduit_relay.so -ldl -lrt /vol0004/apps/opt/
↪kombyne/thirdparty/Conduit/lib/libconduit_blueprint.so /vol0004/apps/opt/kombyne/
↪thirdparty/Conduit/lib/libconduit.so -Wl,-rpath-link,/vol0004/apps/opt/kombyne/
↪thirdparty/ADIOS2/lib64
```

以下に 可視化向けデータを生成する「bin/producer」とレンダリング処理を定義した「examples/pipeline/render.yaml」 を用いた In-Situ 可視化の実行例を示しています。

```
[Compute]$ cp /vol0004/apps/opt/kombyne/bin/producer .
[Compute]$ cp /vol0004/apps/opt/kombyne/examples/pipelines/render.yaml .

4 並列（MPI プロセス）
ドメイン分割（４分割：２ｘ２ｘ１）
タイムステップ：４
[Compute]$ mpirun -n 4 ./producer -domains 2,2,1 -nsteps 4 -pipeline ./render.yaml

各タイムステップ毎に「render.yaml」で定義された可視化処理結果（PNG 画像ファイル）を出力
[Compute]$ ls *.png
render000000.png  render000001.png  render000002.png  render000003.png
```

出力結果：render000000.png