

スーパーコンピュータの利用方法

大阪大学 サイバーメディアセンター
大規模計算機システム担当

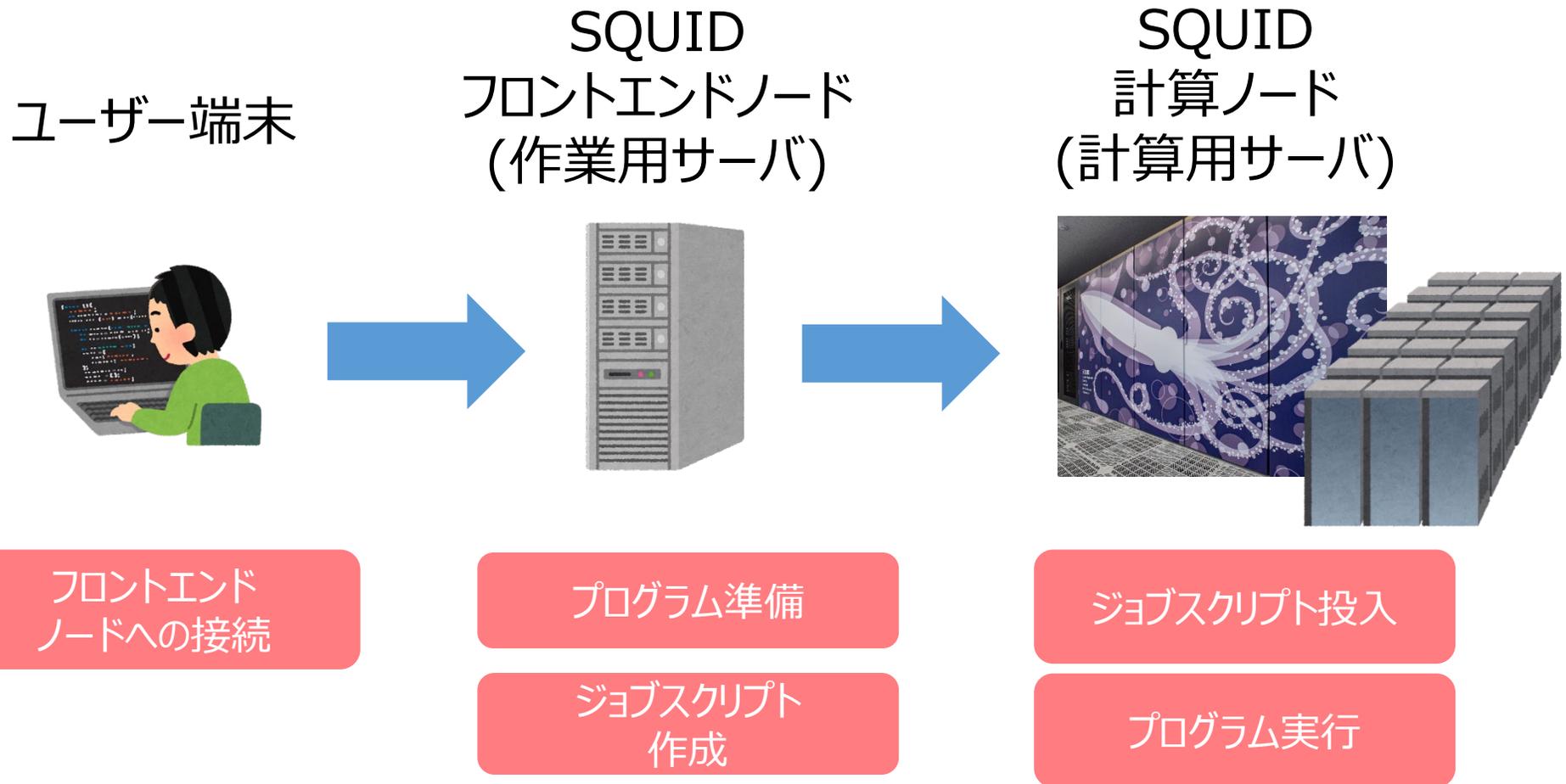
スーパーコンピュータ SQUID

- 3種類の計算ノードと21PBのストレージで構成される
- 総理論演算性能は16.591 PFLOPS
- 阪大だけでなく国内外の研究者に提供



	汎用CPUノード	GPUノード	ベクトルノード
コア数	76	76	VH:24 VE:80
演算性能	5.837 TFLOPS	161.837 TFLOPS	25.61 TFLOPS
メモリ	256 GB	512 GB	VH:128 GB VE:384 GB
ノード数 (サーバ数)	1520ノード	42ノード (8GPU / ノード)	36ノード (8VE / ノード)

スーパーコンピュータ利用の流れ



フロントエンドノードへの接続

SSH (Secure Shell) 接続

- ターミナル(Mac/Linux)やコマンドプロンプト(Win)を使用
- ターミナルソフトを使用 (TeraTerm, Putty等)

接続先

squidhpc.hpc.cmc.osaka-u.ac.jp

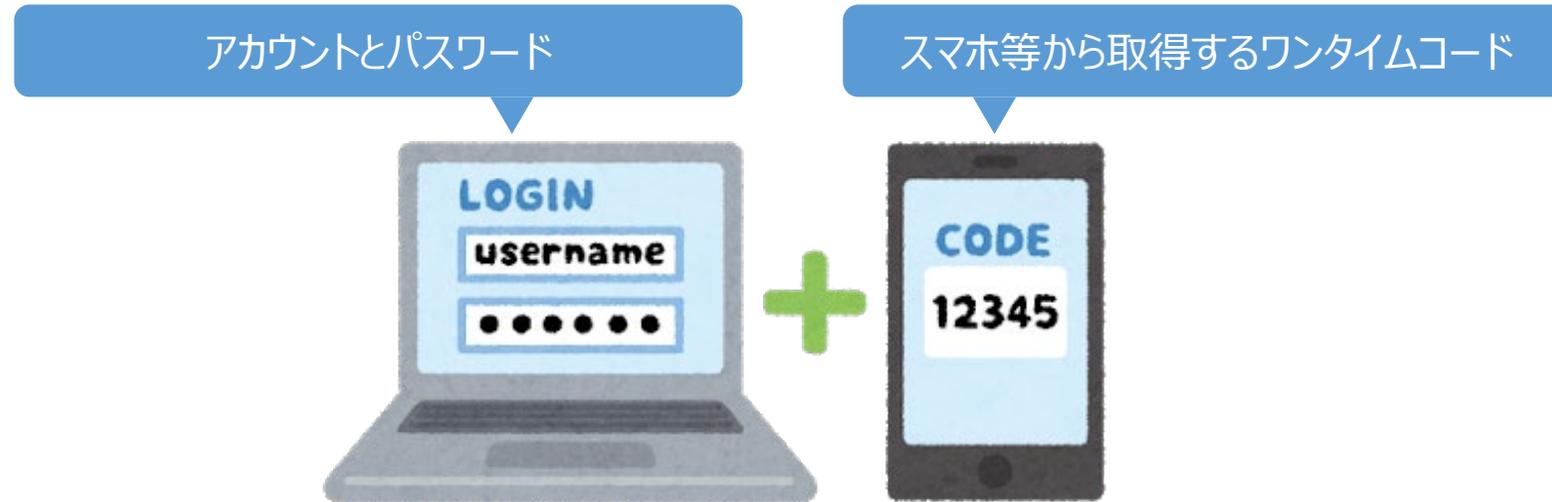
接続コマンド例

```
ssh 利用者番号@squidhpc.hpc.cmc.osaka-u.ac.jp
```

学内/外、国内/外どこからでも接続可能

フロントエンドノードへの接続

SQUIDは多要素認証でのログインとなります
多要素認証用の端末が必要です



※公開鍵認証には対応していません

多要素認証用の端末

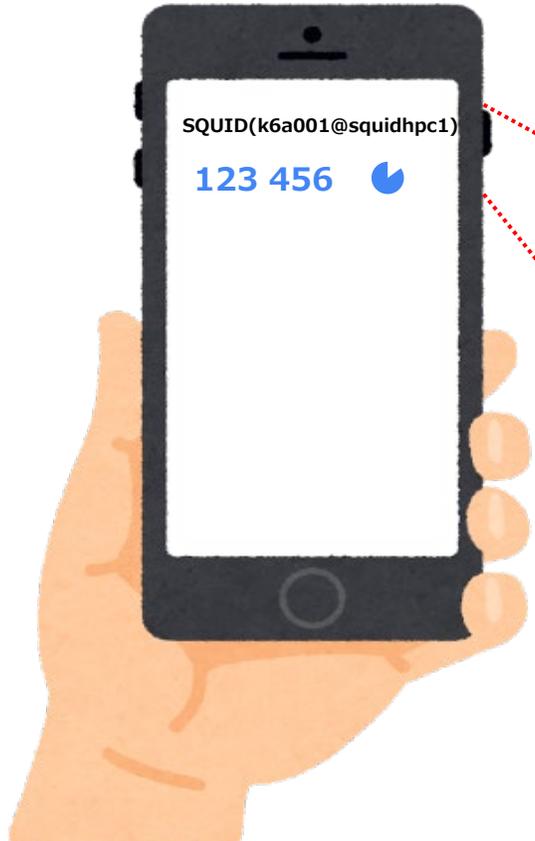
ご自身のスマートフォンやパソコンを多要素認証用の端末としてお使いください
以下いずれかのアプリケーションをインストールしてください

OS	アプリケーション	配布元
Android	Google Authenticator	Google Play Store
iOS	Microsoft Authenticator	Apple App Store
Windows	WinAuth	Github
macOS	Step Two	Apple App Store



フロントエンドノードへの接続

SQUIDに初めてログインするとQRコードが表示されます。
QRコードをアプリで読み込むことで多要素認証の登録が完了します



Initialize google-authenticator

Warning: pasting the following URL into your browser exposes the OTP secret to Google:

https://www.google.com/chart?chs=200*200&chld=M|0&cht=qr&otpauth://totp/user1@squidhpc.hpc.cmc.osaka-u.ac.jp%3Fsecret%3DDXXXXXXXXXCLI%26issuer%3Dsquidhpc.hpc.cmc.osaka-u.ac.jp

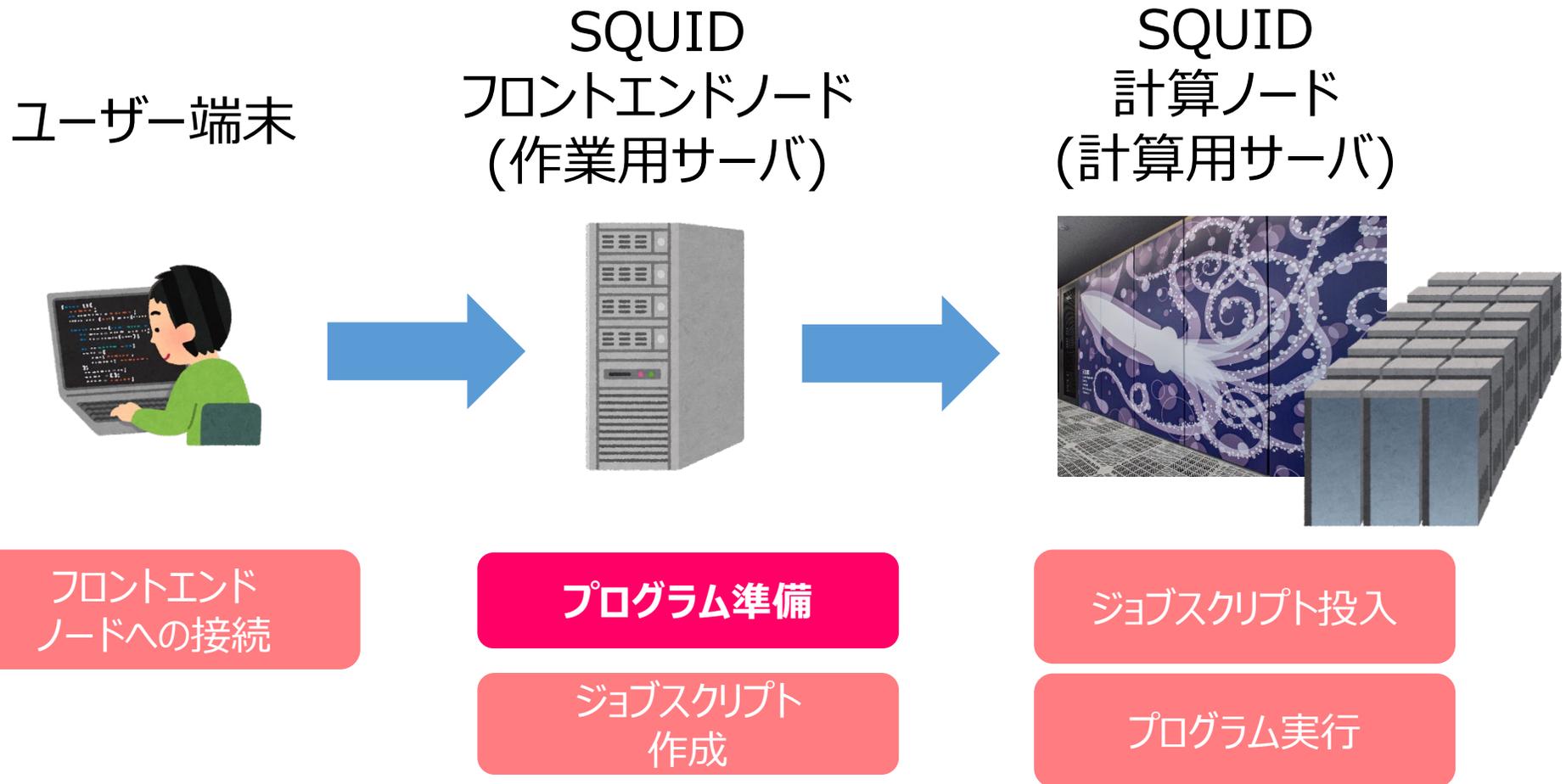


[-1]を入力して
登録完了

Your new secret key is: XXXXXXXXXXXX

Enter code from app (-1 to skip): -1
Code confirmation skipped
Your emergency scratch codes are:

スーパーコンピュータ利用の流れ



プログラムの準備

スパコンを利用するために
プログラムやアプリケーションを準備する必要があります

当センターの計算機で使用可能な主なプログラム言語
Fortran言語、C言語、C++言語、Python、R、Julia

当センターの計算機で使用可能な主なアプリケーション
**OpenFOAM、LAMMPS、Gaussian、GROMACS
PyTorch、QuantumESPRESSO、etc...**

必要なアプリケーションをご自身でインストールすることも可能です！

利用環境の設定

利用するプログラムやアプリケーションに応じて環境の設定が必要

Environment modulesというツールを使用

	Intelコンパイラ	NVIDIA HPC SDK	ベクトルコンパイラ	GNUコンパイラ
モジュール	BaseCPU	BaseGPU	BaseVEC	BaseGCC

などなど…

コマンド例

```
module load BaseCPU  
→Intelコンパイラのコマンド「ifort」が使用可能になる
```

```
module load BaseApp  
module load gromacs/2021.2  
→アプリケーション GROMACSが使用可能になる
```

プログラムの準備：まとめ

スパコンを利用するために、プログラムやアプリケーションを準備する必要があります

① 開発したC言語やFORTRAN言語のプログラムをお持ちの方

→スパコンにプログラムを持ってきて、コンパイルしましょう

② Pythonで機械学習をしている方

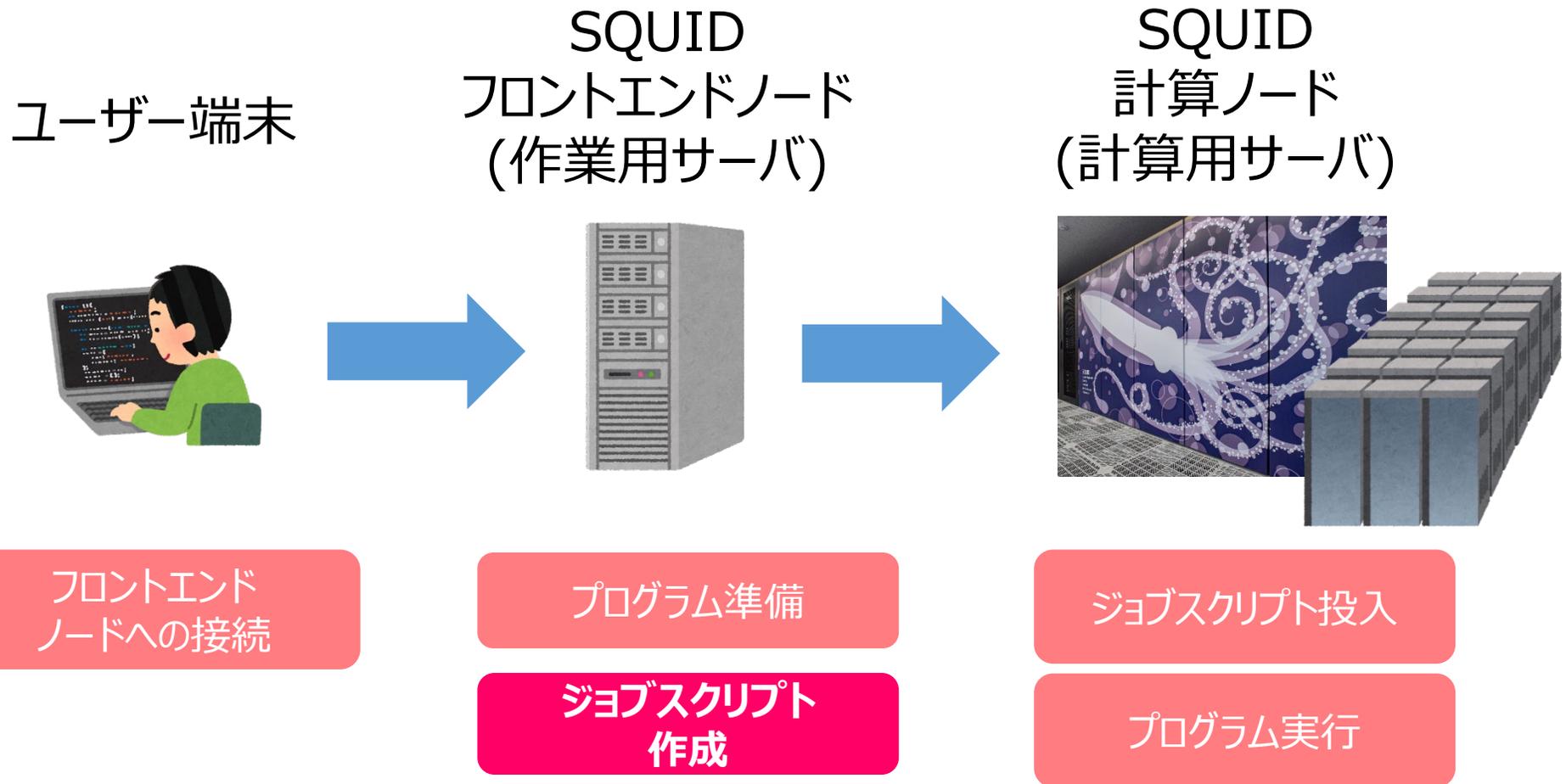
→スパコンで機械学習のフレームワーク等、Pythonパッケージを準備しましょう

③ オープンソースのアプリケーションで計算されている方

→スパコンに入力ファイル等必要なデータを持ってきましょう

→スパコンにアプリケーションをインストールしましょう

スーパーコンピュータ利用の流れ



コンピュータの利用方法

インタラクティブ利用

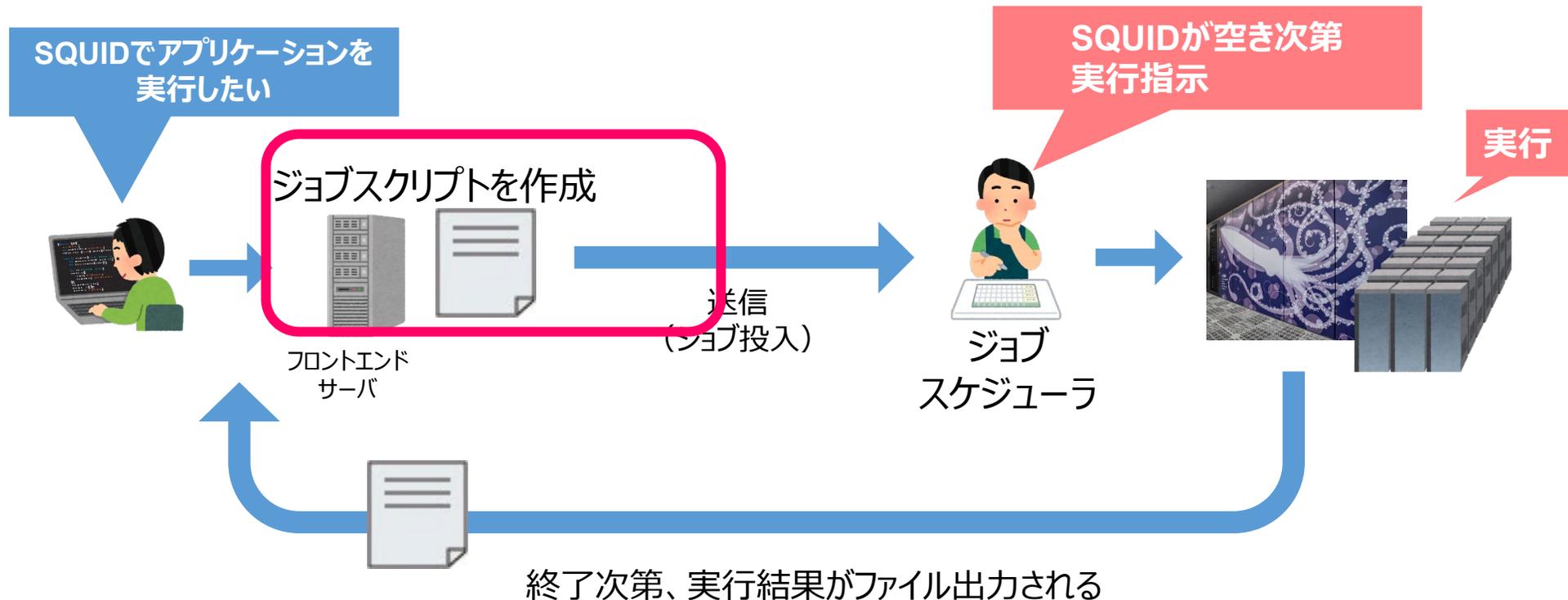
コマンド等を通してコンピュータに直接命令し、リアルタイムで処理を実行
操作として手軽

バッチ利用

コンピュータにまとめて処理を命令し実行
処理の命令が終われば、ログアウトしてもOK

バッチ利用

処理を「ジョブスクリプト」に記述し送信 (→ジョブ)
ジョブスクリプトに基づき計算機が処理を実行



ジョブスクリプト

```
#!/bin/bash  
  
#PBS -q SQUID  
#PBS -l elapstim_req=1:00:00  
  
module load BaseCPU  
cd $PBS_O_WORKDIR  
./a.out
```

SQUIDのリソースや環境設定
実行したい処理を記載したシェルスクリプト

ジョブスクリプト

```
#!/bin/bash
```

```
#PBS -q SQUID  
#PBS -l elapstim_req=1:00:00
```

使用する
リソースや環境

```
module load BaseCPU  
cd $PBS_O_WORKDIR  
./a.out
```

NQSオプション(#PBS~)でリソースや環境の設定を行う

オプション	説明
#PBS -q	ジョブクラスを指定し、計算に使用する優先度等を指定する
#PBS -l	使用する資源値
	elapstim_req : ジョブの経過時間
	memsz_job : 1ノードあたりのメモリ量
	cpunum_job : 1ノード当たりのCPU数
#PBS -v	環境変数の指定(setenvではなくこちらを使うことを推奨する)
#PBS -T	MPI 実行時に指定(IntelMPIの場合、#PBS -T intmpi と指定)
#PBS -b	使用するノード数

必須!

ジョブスクリプト

```
#!/bin/bash
```

```
#PBS -q SQUID
```

```
#PBS -l elapstim_req=1:00:00
```

使用する
リソースや環境

```
module load BaseCPU  
cd $PBS_O_WORKDIR  
./a.out
```

ジョブクラス	利用可能経過時間	利用可能コア数	同時利用可能ノード数	備考
SQUID	120時間	38,912Core (76Core × 512ノード)	512ノード	
SQUID-H	120時間	38,912Core (76Core × 512ノード)	512ノード	高優先度
SQUID-S	120時間	38Core (76Core × 0.5ノード)	0.5ノード	ノード共有

ジョブスクリプト

```
#!/bin/bash
```

```
#PBS -q SQUID
```

```
#PBS -l elapstim_req=1:00:00
```

```
#PBS --group=G12345
```

```
module load BaseCPU
```

```
cd $PBS_O_WORKDIR
```

```
./a.out
```

SQUIDで
実行する処理

ファイルやディレクトリの実行・操作を記述(シェルスクリプト)

- 利用するプログラムやアプリケーションに応じて環境設定が必要
→module loadを実施してください
- **\$PBS_O_WORKDIR** : ジョブ投入時のディレクトリが設定される

ジョブスクリプト

ジョブクラスの指定

```
#!/bin/bash
```

リソースの指定

```
#PBS -q SQUID
```

```
#PBS -l elapstim_req=1:00:00
```

環境設定

```
module load BaseCPU
```

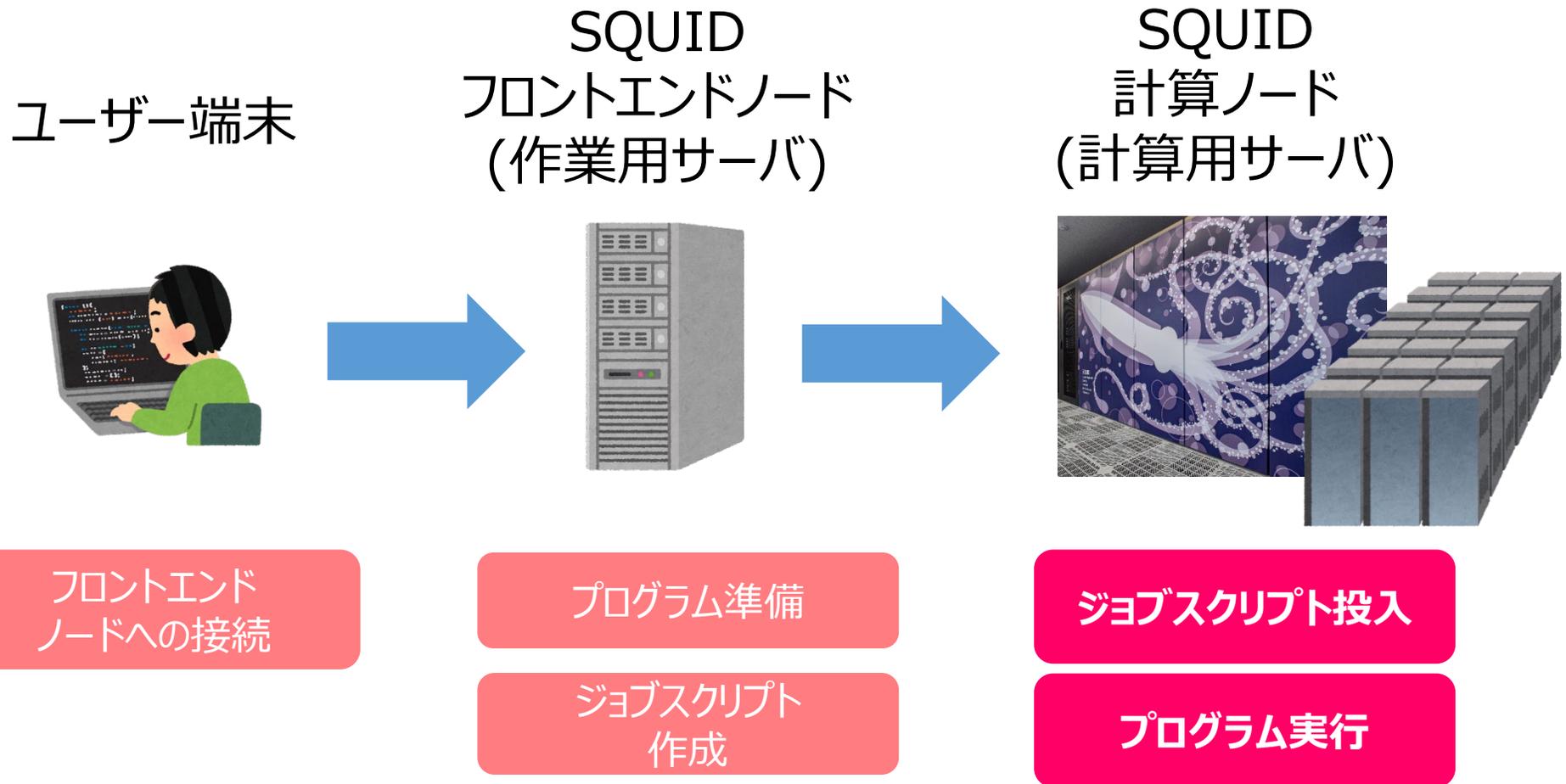
ジョブ投入時のディレクトリへ移動

```
cd $PBS_O_WORKDIR
```

a.outを実行する

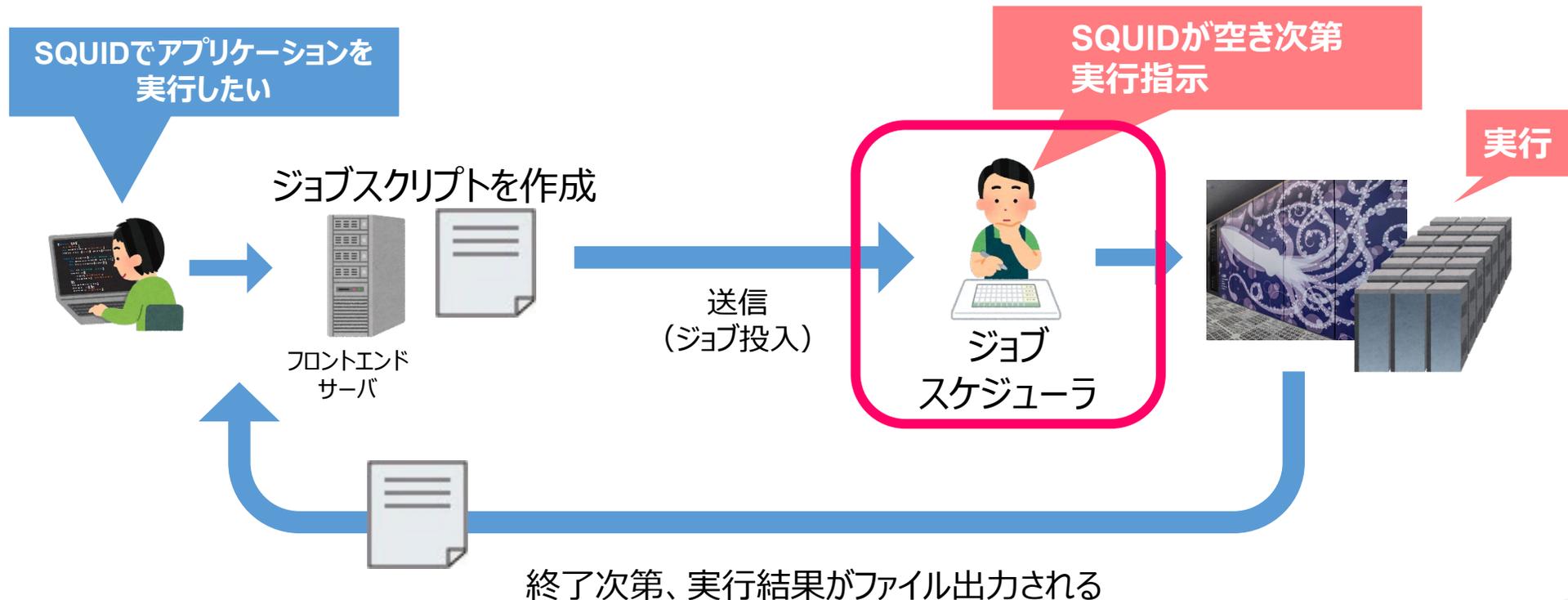
```
./a.out
```

スーパーコンピュータ利用の流れ



バッチ利用

処理を「ジョブスクリプト」に記述し送信 (→ジョブ)
ジョブスクリプトに基づき計算機が処理を実行



ジョブスケジューラとは

あらかじめ管理者によって設定された割り当てポリシーに従い、ジョブを計算ノードに割り当てるソフトウェア



主な役割

計算機システム各ノードのストレージ容量、メモリ容量、性能、使用率を定期的に監視、管理
ユーザより実行したいジョブ要求を受信し、適切なノードを選定
ジョブ実行に伴う入出力データのファイル転送

ジョブスケジューラとは

当センターでは**バックフィル型**を採用

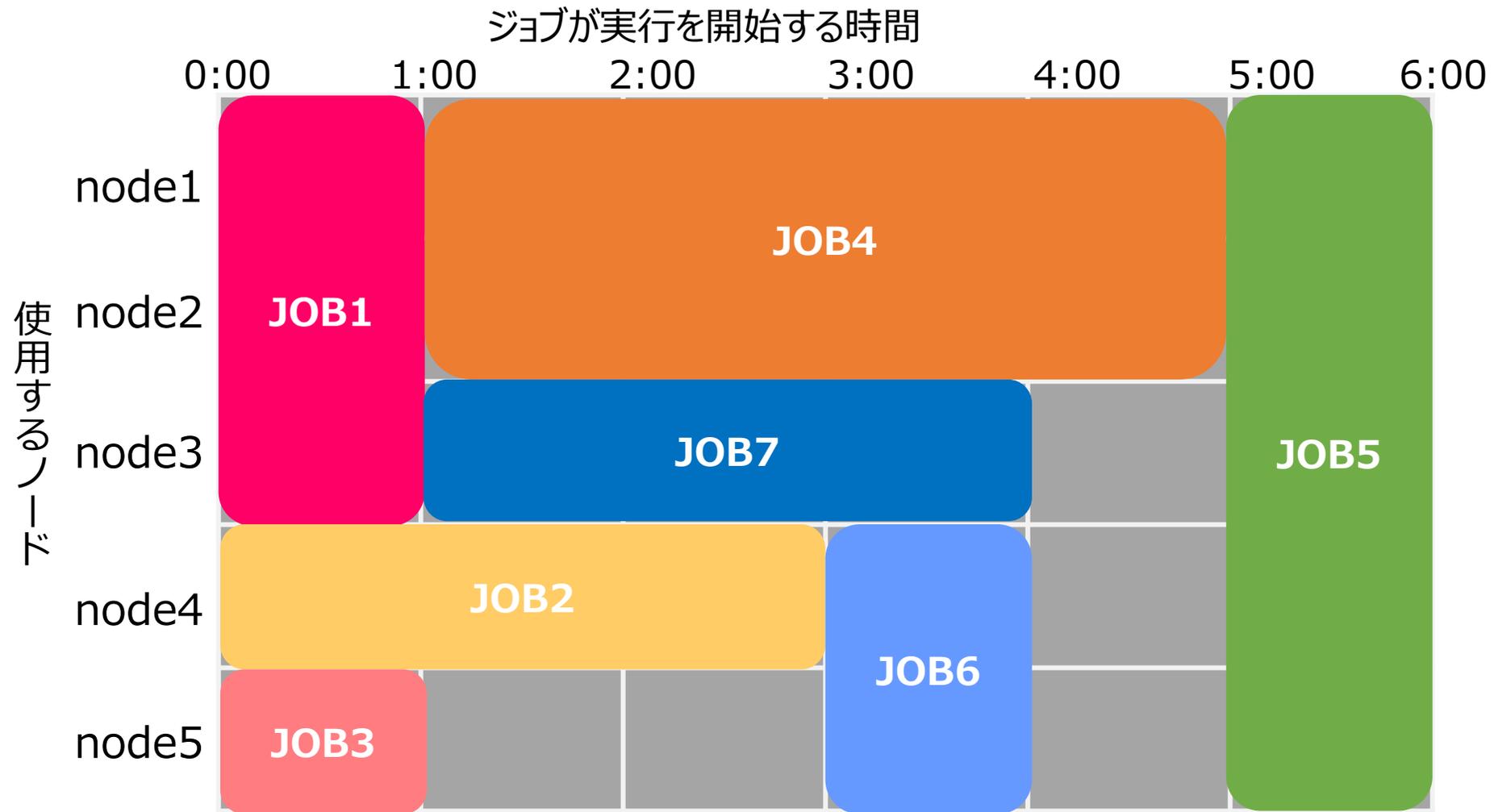
特徴

ジョブの実行開始時間のマップを作成する

マップに載れば、実行開始時間が保障される

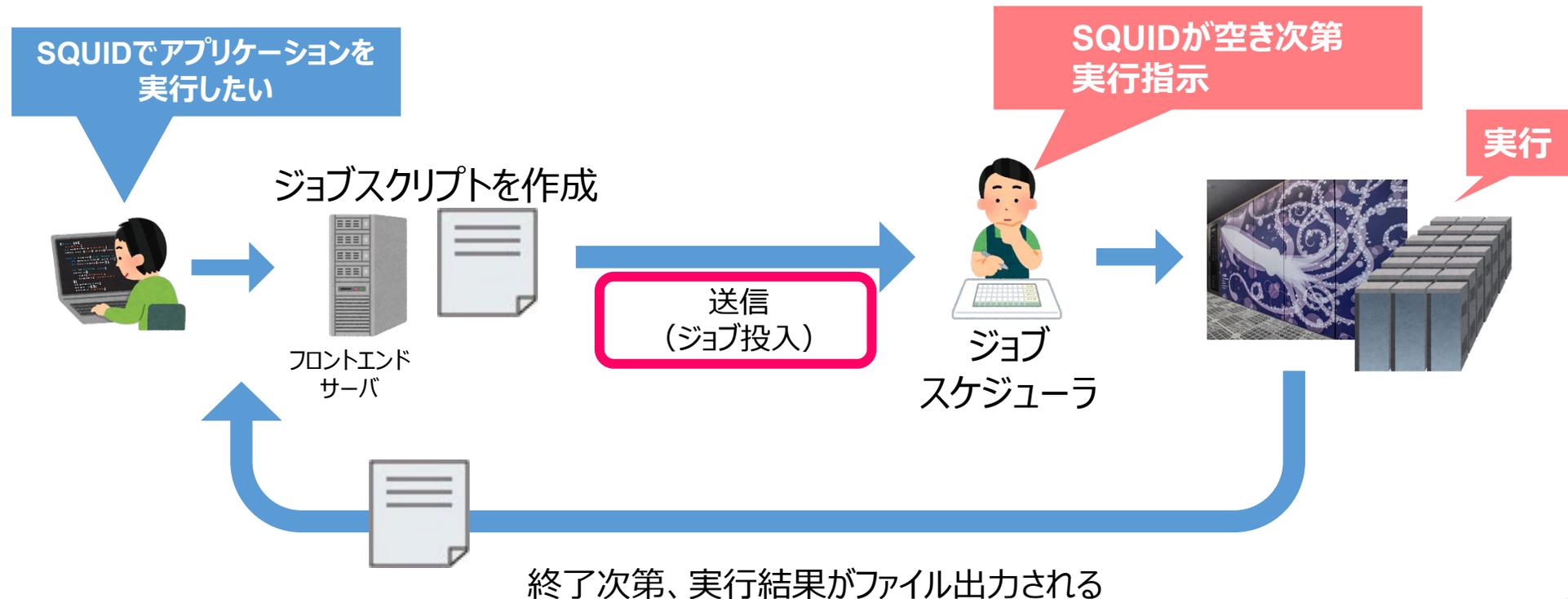
実行中は指定したリソースを占有して割り当てる

ジョブスケジューラのイメージ



バッチ利用

処理を「ジョブスクリプト」に記述
スクリプトに基づき計算機が処理を実行



ジョブの操作方法

ジョブの投入コマンド

```
$ qsub [ジョブスクリプトファイル]
```

投入に成功すると

“Request [リクエストID] submitted to queue: ジョブクラス名”

と表示され、ジョブごとにリクエストIDという通し番号が付与される

リクエストIDの例：123456.sqd

ジョブのキャンセルコマンド

```
$ qdel [リクエストID]
```

キャンセルに成功すると

“Request [リクエストID] was deleted”と表示される

投入済みジョブの確認方法

ジョブの状態確認コマンド

\$ **qstat**

RequestID	ReqName	UserName	Queue	STT	Memory	CPU	Elapse
12345.sqd	nqs-test	k6a001	SQUID	RUN	8.72G	830.66	208

ジョブの状態

待ち状態では「QUE」
実行が始まると「RUN」となる。

実行時間

CPU : 実際にジョブが消費した時間
複数CPU指定の場合は、全CPUを累積表示
Elapse : ジョブが実行されてからの経過時間

ジョブのスケジューリング状況確認コマンド

\$ **sstat**

RequestID	ReqName	UserName	Queue	Pri	STT	PlannedStartTime
12345.sqd	nqs-test	k6a001	SQUID	-1.5684/ -1.5684	ASG	2023-12-07 13:30:23

状態監視

実行時刻が決まると「ASG」表示になる。

混雑具合や優先度により、「実行時間の決定」までの待ち時間が異なるが、一旦実行時間が決定されるとその時刻にジョブ実行が始まる。

実行開始時刻

システムメンテナンスやトラブル時は
再スケジュールされることをご了承ください。

実行結果の確認方法

実行結果や実行エラーは指定しない限り

実行結果： **ジョブスクリプト名.oリクエストID**

実行エラー： **ジョブスクリプト名.eリクエストID**

というファイル名で自動出力される

catやlessコマンドでファイルの内容を出力し確認

```
$ cat jobscript.nqs.o123456
```

意図通りの結果が表示されていれば計算は成功！

プログラムの実行：まとめ

スパコンでプログラムやアプリケーションを実行する際は「**バッチ利用**」

- ① **ジョブスクリプトを作成する**
→右のようなファイルを作成
- ② **ジョブスクリプトをSQUIDに送信する**
→qsubコマンドを使用
- ③ **定期的にジョブの状態を確認する**
→qstatやsstatコマンドを使用
- ④ **実行終了したら結果を確認する**
→サーバ上に保存されたファイルを開いて確認

```
#!/bin/bash
#PBS -q SQUID
#PBS -l elapstim_req=1:00:00

module load BaseCPU
cd $PBS_O_WORKDIR
./a.out
```

さらなるスパコン利用に向けて

利用の参考になるWebページ

サイバーメディアセンター 大規模計算機システム Webページ
<http://www.hpc.cmc.osaka-u.ac.jp>

利用方法

<http://www.hpc.cmc.osaka-u.ac.jp/system/manual/>

FAQ

<http://www.hpc.cmc.osaka-u.ac.jp/faq/>

問い合わせフォーム

http://www.hpc.cmc.osaka-u.ac.jp/support/contact/auto_form/

研究成果

<http://www.hpc.cmc.osaka-u.ac.jp/researchlist/>

さらなるスパコン利用に向けて

本日以降の講習会（全てオンライン）

開催日	講習会名	概要
6/3	スパコンに通じる並列プログラミングの基礎	並列プログラミングの手法や考え方の基礎
6/6	初めてのスパコン	スーパーコンピュータの基礎的な知識と、その使い方 初心者向け【お試しかアカウント付き】
6/12	OpenMP入門	OpenMPによる一般的な並列プログラミングの基礎とその利用方法
6/17	並列プログラミング入門(OpenMP/MPI)	並列プログラミングの基礎と利用方法【お試しかアカウント付き】
6/18	スーパーコンピュータ バッチシステム入門 / 応用	計算機利用（バッチ利用）の概要【お試しかアカウント付き】
6/24	SX-Aurora Tsubasa 高速化技法の基礎	性能測定や基礎的なチューニング手法の説明【お試しかアカウント付き】
6/26	ONION活用講習会	ONIONを使ったデータの保管・共有方法の説明
6/28	コンテナ入門	Singularityを使ったコンテナの利用方法【お試しかアカウント付き】

まずは試用制度をお試ください

3 カ月間 **無料**で以下の資源をご提供



SQUID

75 SQUIDポイント + ストレージ 5TB

汎用CPU ノード
294 ノード時間

GPU ノード
48 ノード時間

ベクトルノード
78 ノード時間

- アプリケーション等 計算環境や技術サポートは有償利用と同等に使用可能
- 有償利用へアカウントの移行も可能

スパコン利用方法のまとめ

- ご自身で開発したプログラム、オープンソースのアプリケーション等、柔軟に使用可能
- スパコンは「**バッチ利用**」
 - たくさんの方が同時に、計算規模に応じてスパコンを切り出して使う
 - ジョブスクリプトを使って、スパコンに計算を指示
- スパコンを使ってみたい方は**試用制度**や**各種講習会**へ！
- 疑問があれば system@cmc.osaka-u.ac.jp まで！