

# データ収集・蓄積・解析クラウド基盤 学認クラウドオンデマンド構築サービス (OCS) を支えるソフトウェア

国立情報学研究所 クラウド基盤研究開発センター

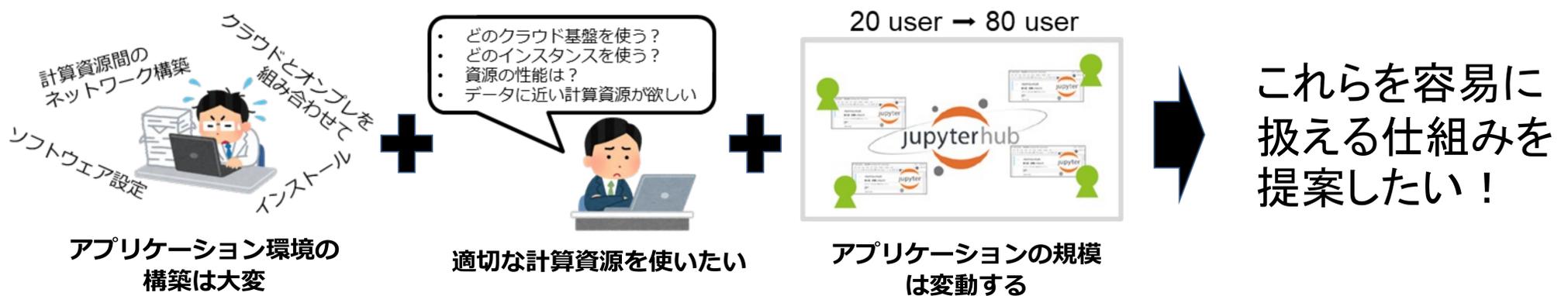
## どんな研究？

- 複数のクラウドやオンプレミスの計算資源を同一の方法で扱えるようにします
- 利用者からの操作を簡易にし、且つ、同一環境を容易に再現できるようにします
- アプリケーションの規模（ユーザ数、データ量など）の変化に応じた実行環境の再構築を容易にします

## 何がわかる？

- VCP (Virtual Cloud Provider)の仕組み
  - 複数の計算資源を同一方法で利用
  - ユーザからの操作方法
  - 公開テンプレートの概要
  - 応用例
    - MCJ-CloudHub
    - Open OnDemand

## 背景・目的



## 研究内容（方法・結果・結論）

## Virtual Cloud Provider (VCP)

### 方法:

- ◆ 利用するクラウドやオンプレシステムを相互にアクセス可能とするネットワーク設定を行い、VCコントローラから操作することで、すべての資源を同一方法で利用できる
- ◆ 可読性の高いJupyter Notebookテンプレートを用いてVCコントローラを操作することで誰でも操作可能

### 結果:

- ◆ アプリケーション環境の構築は大変
  - ・ 一度テンプレートを準備すれば、何度でも同じ環境を再現できる
- ◆ 適切な計算資源を使いたい
- ◆ アプリケーションの規模は変動する
  - ・ 適切な計算資源(オンプレ or クラウド)に環境の移動が出来、計算資源の増減も可能

### 公開テンプレート:

- ◆ LMS, MCJ-CloudHub\*, OpenHPC, Open OnDemand, etc

<https://github.com/nii-gakunin-cloud/ocs-templates>

\*: 2024年度中に公開予定

## JupyterNotebookの記述例

```
VCノードのspecを指定する
TLJH を利用するのに十分な性能・容量のノードspecを指定します。
固定割当てアドレスは、パブリック環境のNAT Proxyサーバに予め設定されているIPアドレスを使います。

In [ ]: # UnitGroup の作成
unit_group = vcp.create_ugroup(ugroup_name)

# VCノード spec
spec = vcp.get_spec(vc_provider, vcnode_flavor)

# spec オプション (ディスクサイズ 単位:GB)
spec.volume_size = volume_size

# spec オプション (固定割当てアドレス)
spec.ip_addresses = [fixed_ipaddress]

# ssh keyfiles
import os
ssh_public_key = os.path.expanduser('~/.ssh/id_rsa.pub')
spec.set_ssh_pubkey(ssh_public_key)

Unitの作成とVCノードの起動
Unitを作成します。Unitを作成すると同時にVCノード（ここでは Amazon EC2インスタンス）が起動します。処理が完了するまで1分半~2分程度かかります。

In [ ]: # Unitの作成 (同時にVCノードが作成される)
unit = unit_group.create_unit('tljh-node', spec)

疎通確認
まず、sshのknown_hostsの設定を行います。
その後、VCノードに対してuname -aを実行し、ubuntu x86_64 Linuxが起動していることを確認します。起動していない場合は、spec.imageに誤りがあります。本テンプレート下部にある「環境の削除」を実行し、spec.imageを修正、全てのセルをunfreezeしてから、最初から再実行してください。

In [ ]: # unit_group.find_ip_addresses() は unit_group内の各VCノードのIPアドレスのリストを返します
ip_addresses = unit_group.find_ip_addresses(node_state='RUNNING')[0] # 今後は1つのVCノードのみ起動しているので [0] で最初の要素を取り出す
print(ip_addresses)

# ssh 設定
!touch ~/.ssh/known_hosts
!ssh-keygen -A [ip_addresses] # ~/.ssh/known_hosts から古いホストキーを削除する
!ssh-keyscan -H [ip_addresses] >> ~/.ssh/known_hosts # ホストキーの登録

# システムの確認
!ssh [ip_addresses] uname -a

TLJH (The Littlest JupyterHub) 環境の構築
VCノード上に、本パブリック環境用に用意したThe Littlest JupyterHubのコンテナイメージを使用して環境を構築します。

TLJHコンテナイメージの取得
VCノード上にコンテナイメージを取得するために docker pull を実行します。
```

### コマンドだけにすると...

```
unit_group = vcp.create_ugroup(ugroup_name)
spec = vcp.get_spec(vc_provider, vcnode_flavor)
spec.volume_size = volume_size
spec.ip_addresses = [fixed_ipaddress]
import os
ssh_public_key = os.path.expanduser('~/.ssh/id_rsa.pub')
spec.set_ssh_pubkey(ssh_public_key)
```

- 図表を組み合わせた説明を挿入できる
- Pythonなどのプログラムを組み込むことができ、ここから実行できる
- 実行結果を残すこともできる

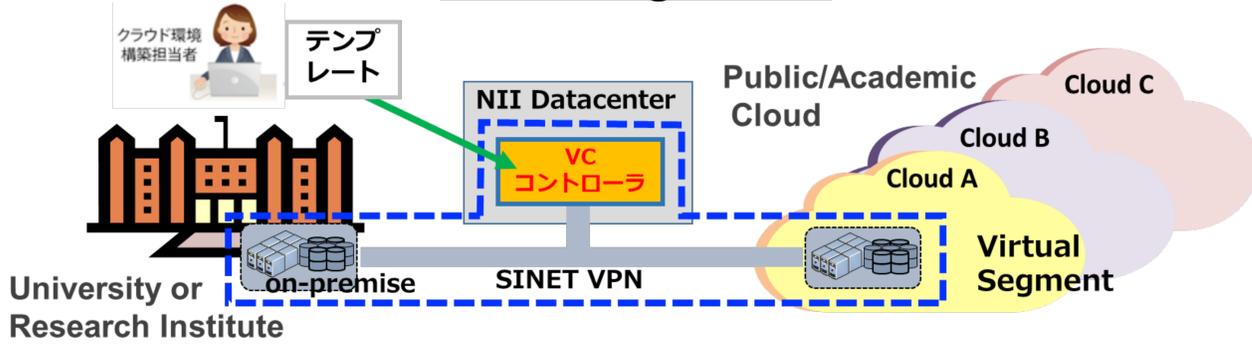


連絡先: 国立情報学研究所 クラウド基盤研究開発センター 大江和一

URL: <https://ccrd.nii.ac.jp/> Email: [cld-ocs-office@nii.ac.jp](mailto:cld-ocs-office@nii.ac.jp)

# データ収集・蓄積・解析クラウド基盤 学認クラウドオンデマンド構築サービス (OCS) を支えるソフトウェア

## VCP configuration



## 利用方法:

- ◆ 国立情報学研究所 学認クラウドオンデマンド構築サービス (OCS) からサービスを提供

<https://cloud.gakunin.jp/ocs/>

## 応用例 (MCJ-CloudHub)

### 目的:

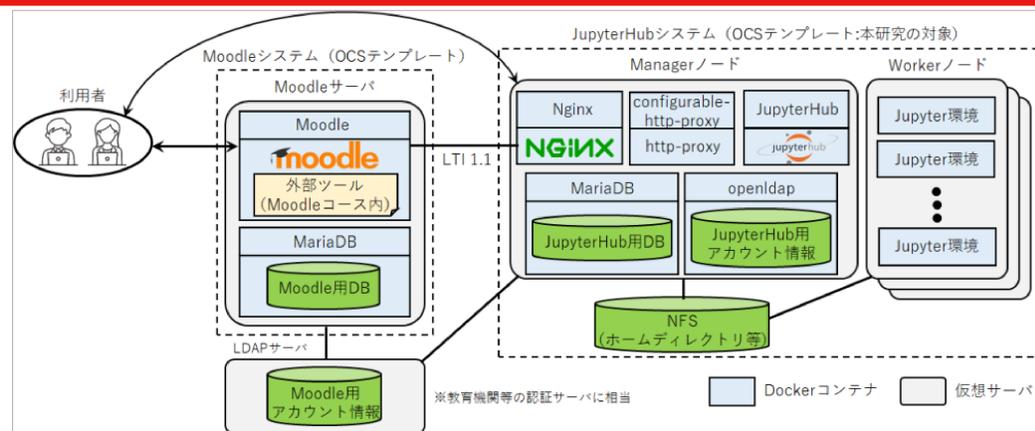
- ◆ オンプレミスやクラウドなどのコンピュータシステム環境上でオンライン講義演習システムを構築・運用できる技術の確立。
- ◆ 情報システム部門が手薄な大学においても運用可能とする

### アプローチ:

- ◆ 山口大学の講義演習システムを汎用化した上でOCSから構築可能なアプリケーションテンプレートを開発(山口大学との共同研究)

### 進捗状況と今後の予定:

- ◆ 昨年度: OCSテンプレートが完成し、山口大学で試運用を実施
- ◆ 今年度: 有効性を高める機能開発と試運用。他大学への普及に向けた説明会開催(利用体験を含む)。



### MCJ-CloudHubの概要

(「複数科目で共同・同時利用可能なWeb型プログラミング教育支援システムのアプリケーションテンプレート開発」より引用)

## 応用例 (Open OnDemand)

### 目的:

- ◆ Open OnDemand環境を容易に構築・運用可能な手段を提供することでHPCクラスタ利用の裾野を広げる

### アプローチ:

- ◆ 開発済OpenHPCテンプレートを前提
- ◆ Open OnDemand環境の構築が可能なアプリケーションテンプレートを新規開発

### 今後の予定:

- ◆ OCSから構築したOpen OnDemand環境と外部スパコン(富岳、ABCI、他)のOpen OnDemandとの連携機能開発

