IaaS 上の PaaS

渡辺 将太[†], 国島 康太[†], 安達 涼[†],
木部 真一郎[‡], 山際 基[†], 上原 稔[†]

クラウド時代には新たな技能が求められ，教育機関はそれを教育する必要ある．教育クラウドはクラウド時代の教育環境であり，学生に自由に使える複数の仮想マシンを提供する．我々は教育クラウドを UEC(Ubuntu Enterprise Cloud)に基づくプライベートクラウドとして構築する．我々の教育クラウドの最大の特徴は過飽和にある。過飽和とは，物理資源以上の論理資源を割り当てることである．過飽和クラウドは通常の 10 倍のインスタンスを実行することを許す．性能を犠牲にコストを大幅に低下する．本論文では，過飽和方式に基づく教育クラウドの利用率をさらに向上させる手法として PoI(PaaS on IaaS)方式を提案する．本論文では 2 つの PoI 方式を比較する．1 つは Single Tenant PaaS(ST-PaaS), もう 1 つは Multi Tenant PaaS(MT-PaaS)である．ST-PaaS はアプリケーションの導入が容易であり，MT-PaaS は集約度が高い．

PaaS on IaaS

Shota Watanabe[†], Kohta Kunishima[†], Ryo Adachi[†],
Shinichiro Kibe[‡], Motoi Yamagiwa[†], Minoru Uehara[†]

In cloud era, new cloud skill is required for IT specialists. Educational organizations such as University need to provide educational cloud for students. We are constructing private cloud based on UEC(Ubuntu Enterprise Cloud) as educational cloud. The most important feature of our cloud is super-saturation, which is defined as allocating much more logical resource than physical resource. Super-saturated cloud realizes to run more 10 times instances than conventional cloud. The performance decreases a little but the cost decreases drastically. In this paper, we propose PoI(PaaS on IaaS) as a method to further improve the util of the education cloud based on supersaturation. In this paper, we compare the two methods of PoI. One can install is Single Tenant PaaS (ST-PaaS), and another is a Multi Tenant PaaS (MT-PaaS). ST-PaaS is the application easying, MT-PaaS has a high intensity.

1. はじめに

近年，クラウドが急速に発達している．クラウドにより多くの技術革新がもたらされた．特に，クラウドへのサーバ集約により迅速なサービス立ち上げ，コストダウンが可能となり，加えて廉価なクライアントの普及によりサービスのすそ野が広がった．

NIST によるクラウド・コンピューティングの定義[1)]によると，クラウド・コンピューティングとは共用のコンピュータ資源（ネットワーク，サーバ，アプリケーション等）へ，ネットワークを経由して時と場所を選ばず簡単にアクセスすることを可能としたモデルである．このモデルは 5 つの基本的な特徴と 3 つのサービスモデル，4 つの実装モデルによって構成される．中でも 3 つのサービスモデルである IaaS(Infrastructure as a Service), PaaS(Platform as a Service), SaaS(Software as a Service)の分類は重要である．

クラウドは教育分野にも革新をもたらす．そもそもクラウド技術者の社会的ニーズは高く，そのような技術者の育成が教育に求められる．また，既存の教育環境では困難な教育が可能となる．例えば，管理者権限のない共用 PC では，学生は自由にアプリケーションのインストールができない．この問題はクラウドを利用することで解決できる．学生は管理者権限を持つサーバを複数台所有できる．ここで，クラウドが提供する仮想サーバをインスタンスと呼ぶ．すべてのインスタンスは強固なセキュリティで保護されているため，管理者権限で安全に利用することができる．教育用途のクラウドを教育クラウド[2)][3)][4)]と呼ぶ．

教育クラウドの課題はコストである．企業が提供しているクラウドサービスは，利用した分だけの料金を支払う．大人数の学生で集中した講義で利用すると無視できないコストとなる．また，クレジットによる料金支払いは教育機関に馴染まない．我々は教育クラウドのコストダウンに過飽和方式を採用する．過飽和方式とは与えられた資源を最大限利用する方式である．我々は，過飽和を物理資源以上の論理資源を割り当て，メモリ容量によりインスタンス数が決まる．このような過飽和クラウドは通常クラウド以上に資源の利用効率が高く，結果としてコストが削減される．過飽和クラウドの性能は必ずしも高くない．特に HPC には適さない．しかし，教育用途では要求される性能は高くない．よって，過飽和クラウドは教育クラウドに適する．

教育クラウドには，それぞれサービスモデルに応じた用途がある．IaaS の教育用途には個人で各種サーバを立ち上げるシステム管理演習などがある．PaaS の教育用途には Web アプリケーション開発，DB 演習，アプリケーション管理演習などがある．これらの演習は管理者権限が不要で

†東洋大学 総合情報学部
Faculty of Information Sciences and Arts, Toyo University
‡東洋大学大学院 工学研究科情報システム専攻
Department of Open Information Systems, Toyo University

あれば IaaS でなく PaaS でも可能である．また PaaS を利用することで，サーバを集約し，コストを削減できる．

Microsoft Window Azure（以下 Azure）[7)]，Google App Engine（以下 GAE）[8)]など初期の PaaS では，専用のプラットフォームが必要であった．しかし，最近では CloudFoundry[12)]のようにオープンソースの汎用プラットフォームが開発されてきた．IaaS 上に PaaS を実現することで，オープンソース PaaS は IaaS と共存し，両立する．その結果，IaaS だけでは不可能だったサーバ集約が可能となる．そこで，本論文では，IaaS 上に PaaS を構築することで高い集約性を持ったクラウドを実現する．このような方式を PoI(PaaS on IaaS)方式と名付ける．また，このようなクラウドが構築コストを削減できることを示す．

本文の構成は以下の通りである．関連研究として 2 節で PaaS について述べる．3 節で教育用クラウドについて詳細に述べる．4 節では PoI(PaaS on IaaS)方式を提案する．5 節で評価を行い，最後に結論を述べる．

2. 関連研究

ここでは, 主として PaaS(Platform as a Service)に関する関連研究を述べる．対象とする教育用クラウドについては次節で詳細に述べる．

PaaS とはアプリケーションソフトが稼働するためのハードウェアや OS などのプラットフォーム一式を，インターネット上のサービスとして遠隔から利用できるようにした形態のことである．

コンピュータの階層で例えるなら, IaaS はハードウェア，PaaS は基本ソフト，SaaS はアプリケーションに対応する．これらは階層を成している．しかし，現実のクラウドでは IaaS, PaaS, SaaS の関係は階層でないことが多い．事実，Azure, GAE には IaaS がない．これらが PaaS/SaaS に特化したクラウドである. 我々はこのような PaaS を第 1 世代(PaaS 1.0)と考える．

PaaS 1.0 の事例として Azure, GAE, Force.com を挙げる．

Microsoft Windows Azure[7)]はクラウドテクノロジーの集合体である「Windows Azure プラットフォーム」向けの開発，サービスホスティング，サービス管理環境として機能する，マイクロソフトのクラウドオペレーティングシステム（クラウド OS）である．メリットは設定や運用をすべてクラウド側が行う. デメリットは OS, ミドルウェアなどデフォルトの追加でインストールするソフトウェアに関して制限があり, 開発環境もサーバも Windows になってしまう．

Google App Engine[8)]は Google 社の提供するクラウドサービス．同社の運用するサーバ環境に自分の開発した Web アプリケーションを置き，実行，公開することができる．使用して，ドメイン所有権の証明や新しいドメイン名を購入することができる. メリットは Eclipse 上からデプロイができる為，簡単にサービスが展開できる．デメリットは使用できる言語が現在 Java と Python のみで, 選択幅は狭い．

Force.com[9)]は Salesforce CRM アプリケーションと同一のインフラ上で, ISV や企業内の IT 部門がビジネスアプリケーションを開発し，運用するための統合されたツールとアプリケーションサービスのセットであり，プラットフォーム上で 100000 以上のビジネスアプリケーションが運用されている．メリットは他の言語やプラットフォームのためのツールキットと統合できる．デメリットは日本語によるヘルプがまだまだ設備されていない．

PaaS 1.0 の多くは少数の開発言語しかサポートしていない．また，ビッグデータに特化した NoSQL タイプのデータベースしかサポートしていないことが多い．それに対して，近年，Heroku, Fluxflex など第 2 世代 PaaS(PaaS 2.0)が発展してきた．

Heroku[10)]は Ruby アプリケーションのクラウドプラットフォーム．ウェブサービスのホスティングサービスであり，基本機能を無料で提供し，拡張機能や特別な機能で課金するフリーミアム（無料版）が採用されている．メリットは多くの言語が使用可能やオープンな技術を使用して開発可能．デメリットは DB の容量が少ない場合，料金がやや割高（5MB まで無料だが、超えると$15 になる）．

Fluxflex[11)]は WordPress や Redmine をはじめとしたオプンソースソフトウェアを Fluxflex 上のプロジェクトにワンクリックでインストールし，利用することができる．AppGarage によって One Click Install が可能なアプリケーションを提供している．blog, CMS, フレームワークを中心に提供されており，無料でも十分使えるが比較的安い値段で有料プランを利用できる．Fluxflex へのサーバへのデプロイには git を用いるため，git の知識が必要となる．アプリケーションを 1 つのプロジェクトで管理をしているため，オプンソースソフトウェアやデプロイしたアプリケーションを複数利用するには有料プランに加入する必要がある．One Click Install とあるが，ものによっては複数回のクリックが必要であり厳密にはワンクリックによるインストールではない．

これら PaaS 2.0 は PaaS 1.0 に比べて多様な開発言語をサポートし，RDBMS もサポートする．言い換えれば，既存 Web アプリケーションの開発方法をそのまま継承できる利点を持つ．また，PaaS 2.0 の多くは AWS など IaaS 上で実現されている．明確に階層化されている．しかし，上記の PaaS 2.0 のミドルウェアはオープンではない．

最後に，オープンソース PaaS として CloudFoundry[12)]をあげる．CloudFoundry とは VMware が提供する PaaS であり，ホスティングサイトである CloudFoundry.com，ローカル環境下限定で動き仮想マシンとして配布されている MicroCloudFoundry, そしてオープンソースとして公開され

ている VCAP(VMware Cloud Application Platform)がある．

CloudFoundry のコア部分(CF Kernel)のコンポーネントは以下の 5 つである．

- CloudController : 制御コントローラー．アプリのデプロイ等 CloudFoundry のコンポーネントへの状態変更命令は CloudController を経由して行う．
- Router : URL ロードバランサー．外部からの HTTP 通信は Router を経由する．アプリをデプロイする際の命令なども Router を経由する．
- DEA : Droplet Execution Agent．ユーザーアプリを動かす．全てのユーザのアプリの実行を司っており，動いているアプリの状態監視も行なっている．
- HealthManager : 監視機構．DEA 上のアプリの状態を監視している．
- Messaging System : コンポーネント間のメッセージ処理. NATS という独自のミドルウェアを使っている．
- CloudFoundry への接続や状態の確認，アプリケーションのデプロイ, ユーザの追加等をするためのクライアントソフトとして VMC がある．CloudFoundry.com や MicroCloudFoundry, VCAP に対する使い方は同じである．

CloudFoundry のようなオープンソース PaaS を用いることで，IaaS 上に PaaS を構築できる．

3. 教育クラウド

ここでは，我々が開発中の教育クラウド[2)][3)][4)]について述べる．教育クラウドの目的は高度ICT技術者の育成に必要な技能を習得する教育環境の構築である．

教育クラウドで教育可能な高度ICT技術には，基本的なシステム管理技術から大規模サーバ運用技術まで含まれる．教育クラウドでは，ユーザは管理者権限を持つサーバを複数台所有できる．既存のPC教室では，共用PCを管理者権限で操作することは許されない．そのため授業に必要なソフトも管理者の許可なくインストールすることはできない．これは教育環境の管理コストを増大させる一因でもある．仮想マシンを用いることで一部制約は緩和できる. しかし，仮想マシンを管理者権限で用いることは無防備なネットワークに不正なアクセスを行うことと等しい．それに対してクラウドでは，すべてのインスタンスは強固なセキュリティで保護されているため，管理者権限でインスタンスを起動しても問題ない．これはパブリックネットワークにBYOD(Bring Your Own Device)を接続することと等しい．

教育クラウドは教育用プライベートクラウドである．一般にパブリッククラウドの方がコストは小さいが，利用率が高くなるとプライベートクラウドの方が有利になる．教育クラウドの利用率を高めることがコスト削減のポイントとなる．

過飽和クラウドは，与えられた資源を最大限利用するクラウドである．我々は，過飽和を物理資源以上の論理資源を割り当てることと定める．例えば，CPU資源はコア数として計量される．あるノードの仮想マシンのコア数の合計がそのノードの物理的なコア数より大きいとき，そのノードは過飽和状態にあるという．我々のクラウドでは，単独インスタンスのCPU性能はAWSマイクロインスタンスの24倍であり，過飽和状態でも十分な性能が得られる[14]. 加えて，教育用途では性能の低下は大きな問題ではない. 過飽和クラウドのインスタンス数はメモリ容量で決定される．メモリを過飽和状態で利用すると，swapにより性能が大きく低下する. そこで, メモリの過飽和は推奨できない. 一方，CPUの過飽和は著しい性能低下を起こさないため容認できる．ディスク資源は余剰傾向にあり，ほとんど飽和しない. よって, メモリ容量がインスタンス数を決定する．

我々の教育クラウドでは，ノードは100GBのメモリを持ち，インスタンスは0.5GBのメモリを消費する．それゆえ理論上はノードあたり約200インスタンスを稼働できる．

教育クラウドはUEC(Ubuntu Enterprise Cloud)に基づき以下の要素で構成される（図1参照）．

- CLC(Cloud Controller): クラウド全体を制御する．複数のCC(Cluster Controller)を配下に持つ．
- Walrus: インスタンスイメージ等を保管するストレージサービスである．CLCに所属する．
- CC(Cluster Controller): 一群のNC(Node Controller)を束ねて管理する. 大規模なクラウドは1000台以上のNCを持つことがあるが，このようなクラウドは単一ネットワークで構成することは不適切である. サブネットをクラスタとして構成するためにCCが役立つ．
- SC(Storage Controller): EBS(Elastic Block Store)を管理するストレージサービスである．EBSは仮想的な外付けディスクであり, 手軽にインスタンスのディスク容量を増大できる．
- NC(Node Controller): VMM(Virtual Machine Monitor)を持ち, 複数の仮想マシンのインスタンスを実行する主体となる．
- インスタンス: 仮想マシンで実行される実態である. インスタンスのスペックはイメージにより異なる．典型的な演習は最小構成で可能であり，そのスペックは0.5GBメモリ，10GBディスクである．
- FE(Front End): クラウド利用の利便性を図るために導入された共用クライアントである. クラウドを手軽に利用するためメニュー形式の操作法を提供したり，インスタンスにアクセスするSSH鍵を管理し

たりする．

- Proxy: 教育クラウドはセキュリティのために学外との直接アクセスを禁止する．そこで，学外へのアクセスを許可するProxyや学内（特定インスタンス）へのアクセスを許可するReverse Proxyを提供する．
- DNS: インスタンスに外部から参照可能な名前を与える．

我々の教育クラウドは以下のような特徴を持つ．まず，教育クラウドは過飽和クラウドである．これにより教育クラウドの課題であるコストを削減でき，安価に教育クラウドを構築できる． 100インスタンス以上を稼働させるためにNCとCCを1:1で対応させ，プライベートネットワークを構成する．プライベートネットワークは外部からの攻撃を防ぐのに有用であり，教育クラウドのセキュリティを高める．プライベートネットワークから外部へのアクセスにはProxyを用いる．Proxyはキャッシュを兼ね，トラフィックを低減する．一般的に，授業では同じデータにアクセスすることが多い．キャッシュの効果は高い．一方で，サービスを公開する上で障害ともなるため，公開可能なサービスに対してReverse Proxyを用いて外部からのアクセス手段を提供する．クラウドにアクセスするためのツールはポータブルであり，USBメモリ等でどこでも利用できる．

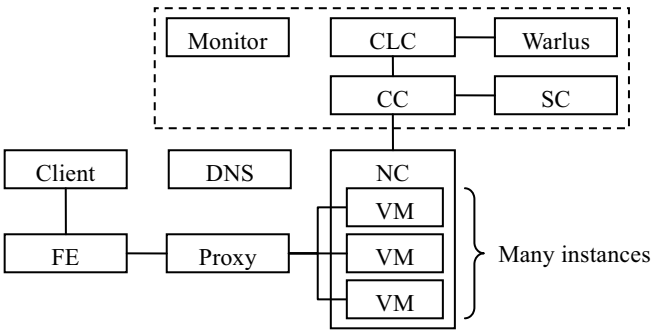

図1. 教育クラウドのシステム概要

我々の教育クラウドは以下のようなサービスを提供する．基本的に教育クラウドはIaaS(Infrastructure as a Service)である．利用者の学生は初心者である．よってCLI(Command Line Interface)よりGUI(Graphical User Interface)の方が親しみやすい．そこで，DaaS(Desktop as a Service)[5)]を提供する．DaaSではSSHトンネルにより安全な通信路を確保する．また，トンネル接続の手順を簡略化するツールも提供している．また，IaaS上にPaaS(Platform as a Service)の層を構築している．PaaSでは代表的なWebアプリケーションをワンクリックでインストールできる．さらに，PaaS上でソフトウェア開発に特化したサービスを集約したDEVaaS(Development as a Service)[6)]を構築した．DEVaaSは必要な時すぐにでも共同開発が始められるように典型的な開発パッケージをサポートしている．

4. IaaS 上の PaaS

4.1 基本設計

我々は教育クラウドに PaaS を構築する．その際，別途 PaaS 用のクラウドを構築するのではなく，現在の教育用 IaaS 上に PaaS を構築する．こうすることでクラウドの構築コストを削減でき，PaaS を弾力的に運用できる．このような方式を PoI(PaaS on IaaS)方式と名付ける．

本論文では PoI として 2 つの方式を提案する．

Single-tenant PaaS (ST-PaaS): 下位層である IaaS のインスタンスを一つの組織が専有し、PaaS を提供する．

Multi-tenant PaaS (MT-PaaS): 下位層である IaaS のインスタンスを複数の組織で共有し、PaaS を提供する．

いずれの場合も PaaS 層では各組織が専有しているように見える．PoI の概念は CloudFoundry と等しい．実際，MT-PaaS では CloudFoundry を採用する．しかし，我々の PoI は単にプラットフォームを提供するだけではなく，迅速に SaaS を配備するサービスを含む. このようなサービスこそ PaaS 2.0 では望まれている．

ST-PaaS は 4.2, MT-PaaS は 4.3, Proxy は 4.4 で述べる．

4.2 ST-PaaS

ST-PaaS は下位層の IaaS によって提供されるインスタンスを特定ユーザが専有する．すなわち，IaaS によって実現された LAMP サーバである．図 2 に ST-PaaS の構成図を示す．ST-PaaS では，ユーザごとに異なるインスタンスを起動する．アプリケーションが同じでもユーザが異なるとインスタンスも異なる．それゆえ，インスタンス数はユーザ数より減らすことができない．一方，ユーザがインスタンスを専有できるため，既存アプリケーションの移植が容易である．移植に関してほとんど制約がない．

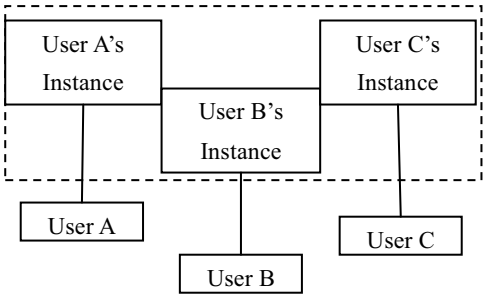

図 2. ST-PaaS の構成

ST-PaaS と単なる LAMP サーバの違いは Web アプリケーションを容易にインストールできる点にある．我々は ST-PaaS にブラウザから Web アプリケーションをインストールできるサービスを実装する．このようなサービスを Instant Install と名付ける．コマンドラインからインストー

ルする必要がなく，手軽に Web アプリケーションを利用できる．

ST-PaaS がサポートするプログラミング言語は PHP である．Web サーバは Apache2, DBMS は MySQL を採用している．

Web アプリケーションの種類として代表的なオプンソースソフトウェアである Pukiwiki, WordPress 等を提供する．現状ではインストールできる Web アプリケーションが少ないため数を増やしていく。Web アプリケーションのインストールを競う競技にインストールマニアックス（サイトを参照する）がある．インストールマニアックスでは，Windows Server/Azure プラットフォームにインストールする Web アプリケーションの数を競う大会である．前回大会と前々回大会ではクラウドがテーマとなり Windows Azure を用いて行われた．このような大会が催される背景には Web アプリケーションのインストールが必ずしも容易でないことがある．ST-PaaS では Instant Install により使いたいときすぐにかつ容易にインストールできる．

Fluxflex では，我々の Instant Install と同様に，手軽に Web アプリケーションのインストールができるしくみとして One Click Install がある．One Click Install では，GitHub からリポジトリをインポートすることによって Web アプリケーションをデプロイする．このためインストール時の詳細なカスタマイズは難しい．

対して我々の Instant Install では予め用意したディレクトリ内にあるインストール用スクリプトをブラウザ越しで起動させることによって Web アプリケーションをデプロイする．このスクリプトによってパッケージをダウンロード，展開・移動・編集を必要に応じて行う．

ST-PaaS における Instant Install は Fluxflex の One Click Install に比べて最初からカスタマイズが可能である．Fluxflex はリポジトリのインポート後，ユーザの名前やパスワード，DB の名前，DB のユーザおよびパスワードなどを既定値で生成する．それに対し我々の Instant Install では，インストール前に，上記のような情報をフォームから入力することができる．これを元にダウンロードしたパッケージ内の設定ファイルを編集することによってカスタマイズを実現している．One Click Install は厳密にはワンクリックでのインストールではない．逆に，Instant Install では標準設定を用いると 1 回のクリックでインストールするモードを持つ．

同様に手軽にインストールできるものとして BitNami[13]がある．BitNami は Web アプリケーションのインストールスクリプトを配布している．このようなスクリプトを用いることで ST-PaaS へのインストールはもっと簡単になる．ただし，BitNami のスクリプトは既定値が多く，Instant Install ほどの自由度は少ない．自由度と手軽さはトレードオフの関係にある．いずれのニーズも無視できない．

4.3 MT-PaaS

MT-PaaS は下位層の IaaS で起動されたインスタンスを複数のユーザで共有する．図 3 に MT-PaaS におけるユーザとインスタンスの関係を示す．MT-PaaS では，起動するインスタンス数を ST-PaaS より減らすことができる．これによりクラウドの資源利用率が向上し，コストを削減できる．

MT-PaaS は CloudFoundry によって実現されている．それぞれのユーザの領域はサンドボックスとなっており，それぞれのプログラムが干渉することはない．逆に開発ならびにデプロイ方法が通常と異なり，スキルを必要とする．具体的には，各クライアントにインストールされた VMC を用いて CloudFoundry にユーザやアプリを追加する．

CloudFoundry がデフォルトでサポートしている言語は Java1.6, Ruby1.9.2, Node.js 0.4.5, Erlang R14B02 である．また，Python や PHP も追加可能である．

現状では，MT-PaaS に Instant Install は提供していない．MT-PaaS では同じ CloudFoundry(PaaS)上では同名のアプリケーションをデプロイすることはできない．名前の権利は早い者順である．異なるユーザが同名の Web アプリケーションをデプロイしたい場合は多々あると思われるので，混乱を避けるために username-app のように一定のルールの元に名前をつける等の工夫が必要である．これが解決することによって MT-PaaS での Instant Install の提供が可能となる．

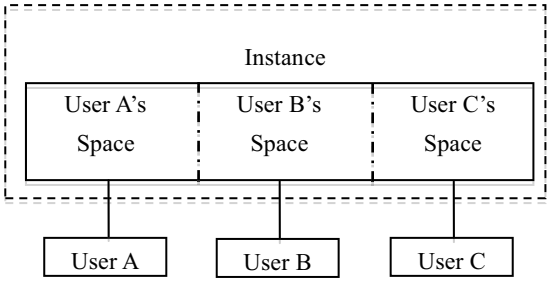

図 3. MT-PaaS の構成

4.4 Proxy

セキュリティの観点から PaaS を無原則に公開することは望ましくない。特に教育クラウドにおいては学生を保護することは重要な課題である．我々の教育クラウドでは，NC を FW(Fire Wall)内側へ配置し，Proxy/Reverse Proxy を介して外部と接続する．

Proxy サーバはクラウドのインスタンスが外部のインターネットにアクセスするために必要となる．クライアントが Web サーバにアクセスする動作を中継する．同時に Proxy サーバはキャッシュサーバを兼ね，外部へのトラフィックを削減する．キャッシュサーバは，ブラウザからリクエストによって得られたコンテンツを HDD のキャッシュ領域に保持し，同じ URL のリクエストが生じたとき，本来の Web サーバにコンテンツを取りに行かず，キャッシュ

内容をブラウザに渡し，ネットワークのトラフィックやサーバの負荷分散も図る．

Proxy サーバには Squid を採用した．Proxy サーバの PC スペックとして Ubuntu server 11.04, CPU AMD Sempron(tm) Processor3400+, Memory 2G, OS 用 HDD 160GB, キャッシュ用 HDD 1TB である．オブジェクトの最大サイズを 5GB にし，DVD 等もキャッシュできるようにした．これは Linux 等オープンソース OS のダウンロードを想定したためである．また，キャッシュ容量として 1TB をすべて使用した．

外部から内部のクラウドインスタンスにアクセスするには Reverse Proxy が必要となる．Reverse Proxy により，外部から内部へのサーバに代わって受け付ける機能やインターネット上に露出するサーバ数を少なくし，公開するホストの数を抑え，1 つのホスト名から様々なサービスに接続でき，ポータル性が向上する．今回 Reverse Proxy として Apache2 を採用した．

Reverse Proxy の方式には仮想パス方式，ポート方式，仮想ホスト方式がある．仮想パス方式では Web アプリケーションのパスを動的に変更する必要がある．ポート方式はインスタンス内で複数の Web サーバを起動する際に用いられる．我々は仮想ホスト方式を採用した．ただし，DNS と IP アドレスの対応は固定とし，既定の名前を用いた．

静的な仮想ホストでは非公開インスタンスまで公開してしまう．セキュリティグループでアクセスを制御すると学内からのアクセスもできなくなる．また，Apache2 は動的仮想ホストをサポートしていない．そこで，動的に仮想ホストを設定する仕組みを自作した．

特別な Proxy アドレスを FE からアクセスすることで仮想ホストの設定を自動生成する．また，定期的に設定変更の有無を調べ，変更されていれば Apache2 の設定を再読み込みする．ただし，現状ではユーザ認証を行っていないため，他人のインスタンスも公開できてしまう問題がある．これはユーザ認証を行い，所有者のインスタンスに対してのみ仮想ホストの設定を許すようにすればよい．今後の課題の一つである．

セキュリティ強化として，セキュリティが保護されたサイトを作るために，Stunnel を用いて https の接続を実現した．これは WWW の個人情報の送信や電子決済など，セキュリティが重要な通信に広く使われている．セキュリティ保護の強度は Web サーバやブラウザで用いられる SSL の実装の正確性や，使用する暗号アルゴリズムに依存する．多くのブラウザでは鍵マークに似た小さなアイコンか鍵を表示し，デジタル認証局による SSL 暗号化，https によって，秘密情報が保護される安全なサイトを表示できるようにした．

5. 評価

ここでは PoI 方式の比較評価を行う．今回の評価で用いた教育クラウドの仕様を表 1 に示す．ここで，NC およびインスタンスの OS は Ubuntu Server 11.10 である．

5.1 ST-PaaS の評価

はじめに，Instant Install の効果を評価する．手動による通常のインストール，Instant Install，Fluxflex での Oneclick install による Web アプリケーションのインストールにかかる時間を比較する．今回は講義で CMS である WordPress を用いたと仮定して検証する．結果を表 2 に示す．Fluxflex での Oneclick install では 1 分 2 秒かかった．Instant Install は手動に比べ約 2.5 倍の速さでインストールすることできた．fluxflex の Instant Install とはあまり速さが変わらないことがわかる．手動でのインストールはいうほど時間がかからないというわけではないが，複数の Web アプリケーションをインストールする際は効率がよく時間の短縮ができる．

表 1. クラウドの仕様

	#cores	Memory[GB]	Disk[GB]
NC	8	96	1000
instance	1	0.5	6

表 2. WordPress のインストール時間

	Instant install	Manual install	One click install
Install Time[s]	56	141	62

次に，対応アプリケーションを代表的な PaaS 2.0 である Fluxflex と比較する．結果を表 3 に示す．○は Instant Install 対応済み，△は現在対応中，×は未対応となっている．対応アプリの数では Fluxflex の方が多いが，カスタマイズの自由度は ST-PaaS の方が高い．

表 3. サポートされた Web アプリケーション

Application	ST-PaaS	Fluxflex
WordPress	○	○
phpMyAdmin	△	○
Moodle	△	○
Redmine	△	○
Jenkins	△	×

5.2 MT-PaaS の評価

CloudFoundry のユーザやアプリの管理は VMC で行われる．VMC で作成したユーザ単位でアプリを管理している．互いのアプリはサンドボックスとなっており干渉しない．

次に，MT-PaaS とその他の PaaS を比較する．ST-/MT-PaaS

は共に過飽和方式に基づく教育用クラウドであるため，そのコストは非常に小さい．GAE も Fluxflex も一定の資源制約の下なら無料で利用できる．ST-/MT-PaaS は共に資源の制約は小さい．表 4 に結果をまとめる．

表 4. MT-PaaS とその他 PaaS の機能比較

	MT-PaaS	ST-PaaS	GAE	Fluxflex
コスト	○	○	△	△
資源	○	○	△	△
言語	△	△	×	○

MT-PaaS の有効性を示すために，あるサービスを提供するために必要なインスタンス数をその他の方式と比較する．比較する対象として ST-PaaS と Multi Tenant SaaS(MT-SaaS)をあげる．ここで，MT-SaaS とはマルチテナントに対応した SaaS である．MT-PaaS はシングルテナント SaaS を PaaS 層でマルチテナント化する．これによりマルチテナント未対応の多くの Web アプリケーションを効率よく運用できる．ここで，テナント数を n としたとき各方式で必要となるインストール数(#installs)とインスタンス数(#instances)を表 5 に示す．MT-PaaS では 1 台のインスタンスに同一アプリをテナント数分インストールする必要がある．

表 5. MT-PaaS と他方式の比較

	ST-PaaS	MT-PaaS	MT-SaaS
#install	1	n	1
#instances	n	1	1

5.3 Proxy の評価

ここでは Proxy によるキャッシュサーバについての評価を行う．今回の評価では，Ubuntu Server 12.04 の CD イメージ（サイズ約 717.3MB）をダウンロードするのに要した時間をキャッシュの有無で比較した．ここで表 6 に表す．結果としてキャッシュを構築すると約 4 倍にも向上した．キャッシュ機能により，一アクセスが高速になることが分かる．

表 6. キャッシュの性能評価

Cache	Time[s]	Speed[M/s]
First	208	3.4
Second	53	13.5

次に，Reverse Proxy の速度評価について行う．今回の評価ではインスタンスを閲覧する速度と，Reverse Proxy を通してインスタンスを閲覧する速度を比較した．ツールは Apache ベンチマークを用いて分析した．クライアントは Ubuntu Server 11.10 を使用した．まず，リクエスト回数を 100 にし，同時アクセス数を 10 で調査した．次に，リクエスト数を 100 にし，同時アクセス数を 100 で調査した．ここで表 7 に表す. 結果として Reverse Proxy を構築すると約 10%低速した．さらに，Reverse Proxy により，リクエスト数が増えると速度が低速することも考えられる．

表 7. ReverseProxy の性能評価

#clients	ReverseProxy[KB/s]	NonReverseProxy[KB/s]
10	294	321
100	304	347

6. まとめ

本論文では，IaaS 上に PaaS(PoI)を構築した．PoI では，Azure や GAE のような専用のプラットフォームを使用せず，汎用のプライベートクラウドの構築ができる．IaaS と PaaS とでプラットフォームを共通化することにより，資源の集約，コスト削減が期待できる．

我々の研究室では過飽和クラウド方式による IaaS，すなわち過飽和 IaaS が実現されている．今回，過飽和 IaaS 上で ST-PaaS を実現したことにより PaaS でも高い利用率を達成した．また，ST-PaaS 環境は必要に応じて弾力的に利用可能であり，講義開始時に短時間で環境を構築できる．これにより授業の手間も削減できる．

また, CloudFoundry により MT-PaaS を構築した. MT-PaaS は 1 つのインスタンスに複数人でアクセスするため，ST-PaaS に比べ資源利用率が高い．よって，一層のコストダウンが期待できる．

PoI の利点として移植性がある．インスタンスを保存することにより、同じ環境を構築することができる。OS のインストールが不要となる。インスタンスは簡単に削除することができ、間違えてしまってもすぐに演習の再開ができる。

我々の独自モデルである MT-PaaS は従来の PaaS よりも利用率の高い過飽和クラウドである。Instant Install による授業支援，Proxy/Reverse Proxy によるアクセス管理などが MT-PaaS をサポートする．これが我々の提案する PoI を用いた新しい教育クラウドのモデルである．

今後の課題は、ST- PaaS における Instant Install のサポートの強化と MT-PaaS の安全な運用である．

現在の Instant Install ではサポートしている言語と Web アプリケーションの種類が少ない状態である．今後は様々な言語に対応した Web アプリケーションを Instant Install できるようにする．これによって ST- PaaS の環境が他言語に対応するため，将来的には自分の好きな言語で記述しアプリケーションをデプロイすることができる．またインストールできる Web アプリケーションのバージョン管理をできるようにする．

インスタンス上に CloudFoundry のインストールし MT-PaaS の環境(PoI)を構築できたが，現在の段階では FW 内部の CloudFoundry にターゲットが通っていない．これはセキュリティの問題でもあり，今後は安全な運用を実現することを目指す．

参考文献

1) P. Mell, T. Grance: "The NIST Definition of Cloud Computing", NIST Special Publication 800-145(draft), (2011)
2) Shinichiro Kibe, Minoru Uehara: "Proposal for a Cloud-based Educational Environment", In Proc. of 3rd International Workshop on Information Technology for Innovative Services(ITIS2011) in conjunction with the 14th International Conference on Network-Based Information Systems(NBiS2011), pp.523-528, (2011.9.7-9,Tirana,Albania)
3) Shinchiro Kibe, Minoru Uehara, Motoi Yamagiwa: “Evaluation of Bottlenecks in an Educational Cloud Environment“, In Proc. of the 13th International Symposium on Multimedia Network Systems and Applications (MNSA2011) in conjunction with the 3rd IEEE International Conference on Intelligent Networking and Collaborative Systems (INCoS2011), pp.520-525, (2011.11.30-12.2, Fukuoka, Japan)
4) Shinichiro Kibe, Motoi Yamagiwa, Minoru Uehara: “Proposal for Improving Throughput in Supersaturated Cloud“, In Proc. of 4th International Workshop on Information Technology for Innovative Services(ITIS-2012-03) in conjunction with 2012 26th IEEE International Conference on Advanced Information Networking and Applications (AINA2012), pp.981-986, (Fukuoka, Japan, 2012.3.26-29)
5) Shinichiro Kibe, Motoi Yamagiwa, Minoru Uehara: “The Evaluations of Desktop as a Service in an Educational Cloud”, ITIS-2012-09, (TBA)
6) Katsuyoshi Matsumoto, Shinichiro Kibe, Minoru Uehara, Hideki Mori: “Design of Development as a Service in Cloud”, WReCS-2012, (TBA)
7) Microsoft: “Windows Azure”, http://www.windowsazure.com/
8) Google: “Google App Engine”, http://developers.google.com/appengine/
9) Salesforce: “Force.com”, http://www.force.com/
10) Heroku: “Cloud Application Platform”, http://www.heroku.com/
11) Fluxflex: “One Click Installable Web Applications”, http://doc.fluxflex.com/oneclickinstall
12) CloudFoundry: “Welcome to CloudFoundry”, http://www.cloudfoundry.com/
13) BitNami: “Open Source. Simplified”, http://bitnami.org/
14) 木部 真一郎, 上原 稔, 山際 基: "過飽和クラウドにおけるスケールアップ方式の評価", マルチメディア、分散、協調とモーバイル(DICOMO 2012)シンポジウム論文集, 2D-3, pp.412-417, (2012.7.4-6, 山代温泉)