Vol.2011-CLE-6 No.4
2011/12/1

Kumoi を用いたスケーラブルな Moodle 環境の構築

石井嘉明[†]　矢野恭平[†]　廣岡誠之[†]
杉木章義[††]　加藤和彦[††]

近年，教育機関や企業などで LMS の普及が進んでいる．LMS の依存度が高まるとシステム障害時の影響は大きくなり，特に教育機関では主事業に影響を受ける可能性がある．このため，LMS には事業継続性を向上させる高可用性をもったプラットフォームを構築することが求められる．そこで本研究では，高水準なシェル環境を提供しスクリプティング環境を提供するクラウドコンピューティング基盤 Kumoi を用いて Moodle に高可用性を実現する手法について述べる．

The Implementation of Scalable Moodle Environment using Kumoi

Yoshiaki Ishii[†], Kyohei Yano[†], Nobuyuki Hirooka[†],
Akiyoshi Sugiki[††] and Kazuhiko Kato[††]

In recent years, LMS is widely used in educational institutions and companies. Higher dependence on LMS may cause greater influence of disaster, especially on core business of educational institutions. So, higher availability platform for LMS is required to improve business continuity. In this research, we propose to realize high availability of Moodle using cloud computing platform 'Kumoi' which provides high level shell scripting environment.

1. はじめに

近年，教育機関や企業などで LMS（学習管理システム）の普及が進んでいる．文部科学省先導的大学改革推進委託事業「ICT 活用教育の推進に関する調査研究」の報告によると，2010 年における LMS の利用状況は大学（学部研究科）においては 40.2%，短期大学においては 24.5%，高等専門学校においては 73.2%となっている[1]．また，LMS を基盤とした e ラーニングによって遠隔教育を行っている機関も増えてきている[2].

このように LMS は多くの教育機関に普及し，主要な事業を支えるシステムとして利用されつつある．しかしながら，この依存度が高まるほどに，LMS のシステム障害は事業継続性に関わるリスクとなってくる．このため，事業継続性の確保として，LMS には高い稼働率を実現する高可用性が求められる．

従来，高可用性は障害発生時に予備機へ切り替えるコールドスタンバイ構成やサーバを 2 台以上用意し 1 台が使用不可となった場合に切り替えるホットスタンバイ構成，常に同様の機能を提供するサーバ 2 台以上で稼動させトラフィックの分散や負荷軽減などを実施する負荷分散構成などによって実現されていた[3].

これが近年，仮想化技術によって実現されるクラウドコンピューティングにより，仮想マシンの複製や起動，停止によって，HA(High-Availability)や Scalability が実現されるようになった．しかし，一般的にクラウドコンピューティング基盤は仮想化されたリソースのみをコントロールし，アプリケーションレイヤについてはコントロールしない．このため，自律的に HA や Scalability の機能が動作したとしても，アプリケーションの設定は手作業またはツール等を使用して行わなければならない．

本研究では，これらの問題に対処するため OSS（オープンソースソフトウェア）のクラウドコンピューティング基盤 Kumoi[4,5,6]を用いて，Moodle[7]の環境を構築した．Moodle は高等教育機関において最も高い割合で導入･利用されている LMS である[1]．Kumoi は高水準なシェル環境を提供し，物理・仮想リソースの言語オブジェクト化による統一的な操作を可能としたソフトウェアである．このシェル環境は対話環境およびスクリプティング環境を提供しており，抽象化されたクラウドコンピューティング環境を対話的に操作したり，スクリプトによって管理機能を追加したりすることを可能としている．これにより，Kumoi はリソースの効率的な管理を実現し，拡張可能なクラウドコンピューティング基盤となっている．Kumoi は 2010 年 11 月にバージョン 1.0 が公開されているが，現在，筑波大学が中心となり富士ソフト株式会社が協力し

[†] 富士ソフト株式会社
FUJISOFT INCORPORATED
[††] 筑波大学システム情報工学研究科コンピュータサイエンス専攻
Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba

てバージョン 2.0 の開発を進めている．この開発中のモジュールでは，リソースプールによる資源の抽象化，スケジューラによるスケール機構，監視機構，イベント処理，HA 機構，仮想ネットワーク対応などが追加されている．本研究では，これらの機能を活用し，HA やオートスケールの動作に合わせて，アプリケーション設定を自律的に変更する Moodle 自動管理スクリプトを開発することで，Moodle に対して高可用性を実現した．

2. 既存の高可用性技術と問題点

継続性が求められる情報システムでは，高い稼働率を確保するために高可用性技術が導入されている．これらはシステムに何らかの障害が発生した場合に，システムの停止時間を極力短くする技術である．従来技術では，機器を冗長化する方法があり，コールドスタンバイ，ホットスタンバイ，フェイルオーバーなどが挙げられる．以下にこれらの技術の特徴とその問題点を述べる．

コールドスタンバイは，本番機と同様の構成をした予備機を用意しておき，障害発生時に予備機でシステムを継続させる方式である[8]．構成・設計・運用などが単純で確実な反面，障害発生時の起動時間や切り替え時間を含めたシステムの停止時間がかかり，また障害発生時に本番機で処理中の処理やデータは引き継げない場合が多いなどの問題がある．

ホットスタンバイは，予備機を用意するなど構成はコールドスタンバイと類似しているが，本番機と同様に予備機も常時起動させておき，同期させておく方式である．予備機が常時起動しているため，運用や管理面のコストが 2 倍以上になるという問題がある．

フェイルオーバーは，本番機と同じ構成を複数用意し，障害発生時には本番機のデータや処理をそのまま引継ぎ，処理を続行する方式である．最近では，サーバだけでなく，ルータやスイッチなどにも組み込まれている．フェイルオーバーは処理を自動的に引き継ぐ役割をしているが，データや処理の引継ぎ中はシステムが利用できない．また同じ構成を複数用意しなければならず，本番機の監視やデータの同期など運用・管理でのコストが高くなる問題がある．

このように既存の高可用性技術においては，ハードウェアを必要とするバックアップを用意することが前提となっており，コストが大幅にかかることが懸念される．また，本番機と予備機のデータの同期や移行作業においては，運用方法を事前に検討しておく必要があり，本番機に障害が発生した場合速やかに対応できなければ，システムを長時間停止させてしまう可能性がある．特にデータの引き継ぎやシステムの切り替えには，突発的なトラブルに備え専門的知識のある技術者が欠かせない．

システムの保守・運用は，企業では専門部門や技術者による対応が一般的である一方，教育現場においては，コストを極力抑えるために，専門知識を持たない教職員が対応している現場も少なくない．e ラーニングなどの ICT を活用した教育では，「システムの維持，管理で負担が増加」，「ICT に不慣れな教職員の対応の負担」といったデメリットが指摘されており[1]，教育現場においてはこのような負担を軽減することが望まれている．

3. 仮想マシンによる高可用性とその課題

近年では，ハードウェアが高性能化し，尚且つ安価に入手できるようになったこともあり，仮想化技術を用いたクラウドコンピューティングが普及してきている．米国標準技術局(NIST:National Institute of Standards and Technology)の定義によると，クラウドコンピューティングは，コンピューティング資源(ネットワーク・サーバー・ストレージ・アプリケーション・サービス)の共有プールへのオンデマンドなネットワークアクセスを可能にするモデルであり，管理の手間やサービスプロバイダーの仲介作業を最小にしつつ，これらのリソースをすみやかにプロビジョンしリリースできるものとされている[9].

クラウドコンピューティングによって提供されるサービスは，拡大傾向にある．そのため，仮想化技術を用いたクラウドコンピューティングにおいて，HA や Scalability などのサービスの継続性を確保する技術は欠かせないものとなっている．

従来技術では，ハードウェアによる依存度が大きかったが，仮想化することでこれまでよりも柔軟な冗長性構成が可能となり，大幅なコストダウンが見込めるようになった．仮想マシンによる高可用性技術として，HA と Scalability の 2 つについて述べる．

HA は，仮想マシンを監視し，障害発生時には自動で障害を検知して，復旧させる技術である．仮想マシンに障害が検出されると，仮想マシンの配置や再起動を自動的に開始する．これにより，仮想マシンの停止時間を短縮し，運用負荷の軽減を実現している．

Scalability は，負荷の高低に合わせてリソースプール内の動的リソースを拡大・縮小できる技術である．これをリソースの監視，自動制御と合わせることで，仮想マシンの負荷が高くなると台数を増やして負荷分散させ，負荷が低くなると仮想マシンを停止させてリソースを平準化するといったオートスケールを実現することができる．

このように，仮想化技術を用いた高可用性の実現は，従来の技術に比べ柔軟な冗長化構成が可能になったといえる．しかし，これらはリソースの高可用性に留まっており，サービス全体に対する高可用性は実現されていない．仮想化技術による HA やオートスケールを使うことで高可用性を実現することは可能だが，最終的にサービスを提供するアプリケーション，つまりは HA やオートスケール後に発生する各アプリケーションの設定などには，別途対応が必要となる．

しかしながら，仮想化は設備投資コストの低下が期待できるが，運用管理コストは高まることが想定される．これは，管理・保全の対象となる仮想化されたマシンが動的に増減することによって，運用管理が煩雑化するためである．

これらの問題に対応するには，仮想化に対応した統合運用管理ツールの導入が考えられる．しかし，統合運用管理ツールの導入に関しては，「費用対効果に見合わない」「一元管理されているため，耐障害性が問題」という点が懸念されている[10]．統合運用管理ツールの導入は，導入コストだけでなく，ツール自体の運用の習得や設定などにおいて，運用管理工数の増加を招く可能性がある．仮想化している物理サーバの台数がそれほど多くない場合は，多機能の統合運用管理ツールの導入コストは見合わないのである．

4. 構成

前述までの問題を解決するために，本研究ではシェル・スクリプティング環境を提供するクラウドコンピューティング基盤 Kumoi を用いて Moodle 環境を構築した．(図 1 参照)

4.1 物理マシン（pm）の構成

物理マシン層では，ホスト OS 上に Kumoi および Kumoi が対応する仮想マシンモニタである KVM を導入し，稼働させている．Kumoi は仮想マシンモニタを介し，仮想マシンをコントロールするクラウドコンピューティング基盤であるが，並列スケルトンを提供しており，各物理マシンの Kumoi によって，計算機を効率的に管理することが可能となっている．このため，どの物理マシンの Kumoi からでもスクリプトを実行することができる．

また，物理マシン間にてマスタイメージのファイル共有を実現するため，NFS（Network File System）を利用している．NFS では，以下 3 つのマスタイメージを物理マシン間で共有している．

(1) ロードバランサ用マスタイメージ

インストール済みソフトウェア：CentOS，LVS（Linux Virtual Server），keepalive

(2) Web サーバ用マスタイメージ

インストール済みソフトウェア：CentOS，Apache，rsync，lsync，Moodle

(3) DB 用マスタイメージ

インストール済みソフトウェア：CentOS，MySQL

4.2 仮想マシン（vm）の構成

(1) 負荷分散

仮想マシン層では，ロードバランサとして LVS DR を，Web サーバの死活監視のため keepalive を導入した．これによって，複数の Web サーバによる負荷分散によって可用性の向上を実現している．

(2) ファイル同期

Web サーバの仮想マシンでは，Web サーバ間の Moodle データディレクトリのリアルタイム同期を実現するため，lsync および rsync を利用した．rsync は，UNIX システムにおいて，差分符号化を使ってデータ転送量を最小化し，ファイルやディレクトリの同期を行う OSS である[11]．lsync は，ファイルやディレクトリに変更が生じた場合に rsync を実行する OSS である．これらを組み合わせることによって，リアルタイム同期を可能としている．

これにより，ファイルは同期され，ロードバランサがどの Web サーバに転送したとしても，ユーザには同一環境が提供することができる．また，ファイルの多重化によって冗長性が確保される．

(3) DB のレプリケーション

DB の仮想マシンでは，MySQL のレプリケーション機能を利用し，DB の複製によるマスタ/スレーブ構成としている．これによって，MySQL のマスタサーバに障害が生じた場合でも，スレーブサーバを自動的にマスタサーバに昇格させることで，MySQL クラスタを使用せずとも高可用性を実現することができる．

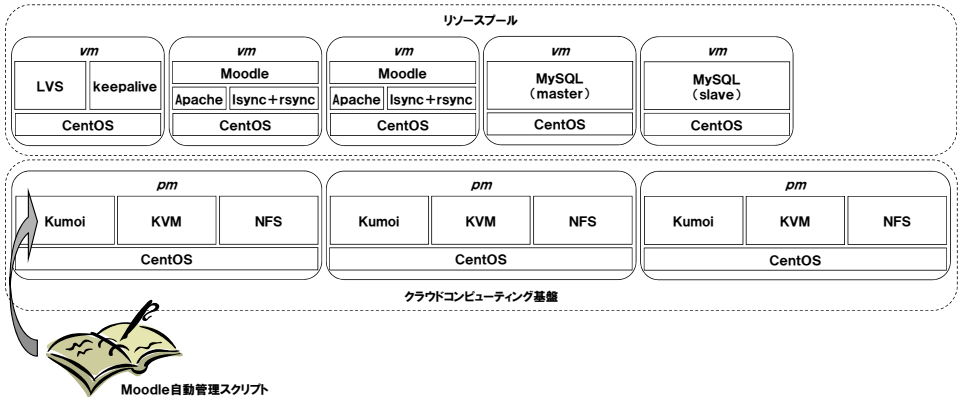

図 1　Kumoi を用いた Moodle 環境の構成図

5. Kumoi スクリプトの設計

本稿では，Kumoi の基本機能を説明した後，本研究で開発した Moodle 自動管理スクリプトの HA，オートスケールおよびこれらに連動したアプリケーション設定を行

う Moodle 自動管理拡張機能について示す.

5.1 Kumoi の基本機能

(1) 資源の言語オブジェクト化

Kumoi では，物理マシンや仮想マシン，ネットワークなど，クラウドコンピューティングにおける様々な資源を言語オブジェクト化し，同一インターフェースを提供している．実資源と言語オブジェクトの間でマッピングを行っており，これにより各資源をオブジェクトとして直感的に扱うことができる[12].

(2) 高水準なスクリプティング環境

Kumoi のシェル環境は，プログラミング言語 Scala の追加ライブラリの形式で実装されており, Scala が備える型推論つきの静的型付けやリスト操作をはじめとする高水準な記述を利用することができるように設計されている．これにより，従来のシェル環境だけでなく，スクリプティング環境や対話環境が提供されており，効率的なクラウドコンピューティングの管理を実現している[12].

(3) 並列分散フレームワーク

サービス基盤の多くは中央管理方式であるのに対し，Kumoi は分散管理方式で構成されている．このため，Kumoi は言語オブジェクトを操作するための並列スケルトンを提供している．また，物理マシン間の Kumoi は互いに Gossip プロトコルによって通信し，動的にメンバシップ管理しており，正常に起動している物理マシンのみによってシステムが構築されるため耐障害性に優れている[12].

5.2 Moodle 自動管理スクリプトの設計

Moodle 自動管理スクリプトでは, Kumoi の管理機能を利用してリソースプールを作成し，このリソースプールに対して所属する物理マシンを追加することで動的なリソースの割り当てを可能としている．これらの仮想マシンに HA やオートスケールの対象として設定することで，リソースの高信頼性を実現している．また，仮想マシン上のアプリケーションに対する制御を Moodle 自動管理スクリプトから実現することにより，HA やオートスケールによって実現されるリソースの高可用性に適応する自動アプリケーション設定を実現させることが可能となる．これにより，自律型のスケーラブルな Moodle 環境を実現した.

Moodle 自動管理スクリプトでは，ロードバランサ用仮想マシンおよび DB 用仮想マシンを対象として HA を，Web サーバ用仮想マシンに対してはオートスケールを実現するように設計した. 本スクリプトを実行することで, ロードバランサ用仮想マシン, Web サーバ用仮想マシン, DB 用仮想マシンがマスタイメージから作成・起動される. これに合わせて，Kumoi の HA 機能によって HA プロセスがロードバランサ用仮想マシンおよび DB 用仮想マシン毎に生成され，オートスケール機能からは Elasticity プロセスが Web サーバ用仮想マシンに対して生成される.

また，Moodle 自動管理スクリプトからは Kumoi を介してロードバランサ監視プロセス・MySQL 監視プロセス・Moodle 監視プロセスの３つのプロセスが Moodle 自動管理用拡張機能として生成される．ロードバランサ監視プロセスは，HA プロセスからのイベントを検知し，ロードバランサ用仮想マシン上のアプリケーションに対してチューニングを行うシステムを実現する．また，MySQL 監視プロセスは HA プロセスからのイベントに加えて，Moodle 監視プロセスからのイベントを検知し，DB 用仮想マシン上のアプリケーションに対してチューニングを行うシステムを実現する．Moodle 監視プロセスは，Elasticity プロセスからのイベントに加えて MySQL 監視プロセスからのイベントを検知し，Web サーバ用仮想マシン上のアプリケーションに対してチューニングを行うシステムを実現する．（図 2 参照）

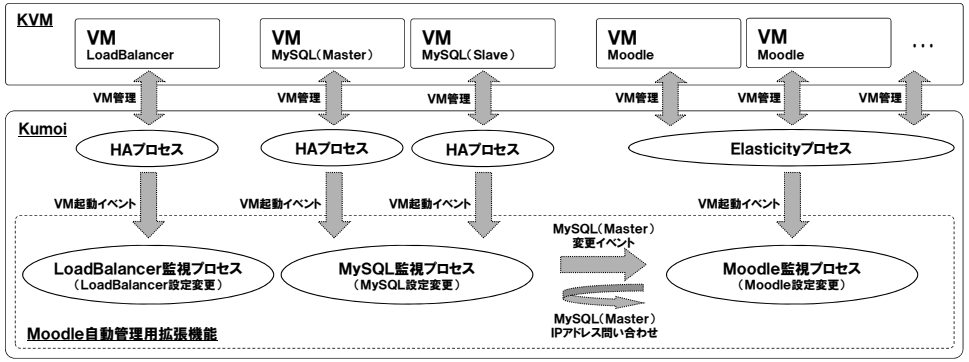

図 2　Moodle 自動管理スクリプトの処理

5.2.1 HA

HA プロセス, ロードバランサ監視プロセスおよび MySQL 監視プロセスの処理を示す.

HA プロセスの処理

(1) 仮想マシンの死活監視を行う.
(2) 仮想マシン障害を検知すると，自動的に障害が発生した仮想マシンのマスタイメージをコピーオンライトにて複製する.
(3) 複製したイメージから仮想マシンを起動させる．（図 3 参照）
(4) ロードバランサ監視プロセスまたは MySQL 監視プロセスに，VM 起動イベントを送信する.

ロードバランサ監視プロセスの処理

(1) HA プロセスからのイベント受信待ち.
(2) 仮想マシン起動イベントを受信した場合，以下の処理を行う.

① プロセスが保持する固定情報から keepalive の設定を変更する．
② keepalive を起動させる．

MySQL 監視プロセスの処理

(1) MySQL マスタサーバ用仮想マシンの死活監視および HA プロセスからのイベント受信待ちを行う．
(2) 仮想マシン障害を検知した場合，以下の処理を行う．
① MySQL スレーブサーバの設定を変更し，マスタサーバへと昇格させる．
② Moodle 監視プロセスに，MySQL マスタサーバ変更イベントを送信する．
(3) 仮想マシン起動イベントを受信した場合，以下の処理を行う．
① 起動した MySQL をスレーブサーバとして，MySQL マスタサーバの IP アドレス等を設定し，レプリケーション構成とする．

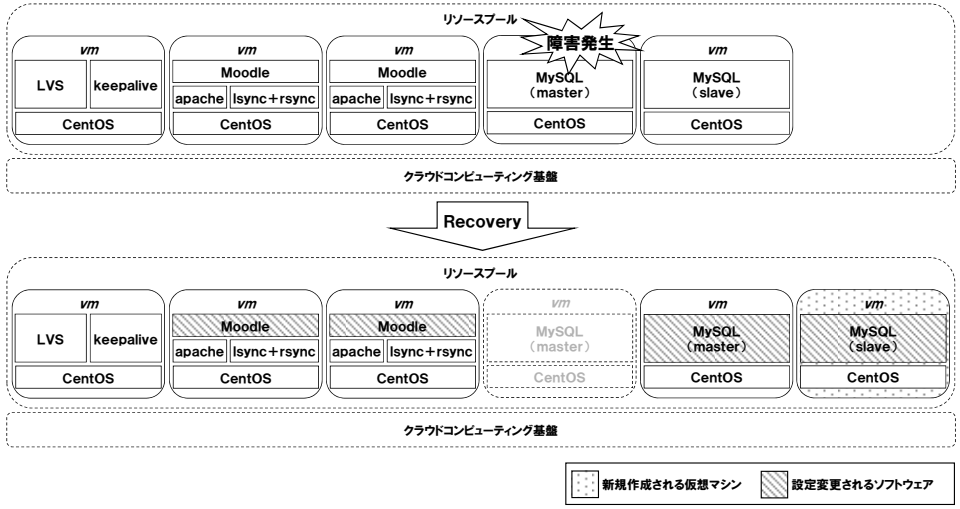

図 3　HA の動作

5.2.2 オートスケール

Elasticity プロセス，Moodle 監視プロセスの処理を示す．

Elasticity プロセスの処理

(1) 仮想マシンの CPU 使用率監視を行う．
(2) CPU 使用率が最大閾値を上回った場合，以下の処理を行う．
① 自動的に仮想マシンのマスタイメージをコピーオンライトにて複製する．
② 複製したイメージから仮想マシンを起動させる．（図 4 参照）
(3) CPU 使用率が最小閾値を下回った場合，以下の処理を行う．
① Web サーバ用仮想マシンを一台停止させる．

Moodle 監視プロセスの処理

(1) MySQL 監視プロセスおよび Elasticity プロセスからのイベント受信待ち．
(2) MySQL マスタサーバ変更イベントを受信した場合，以下の処理を行う．
① MySQL 監視プロセスから，変更後の MySQL マスタサーバの IP アドレスを取得する。
② Moodle の参照先 DB の設定を変更後のマスタサーバの IP アドレスに変更する．
(3) 仮想マシン起動イベントを受信した場合，以下の処理を行う．
① Moodle データディレクトリが同期されるように，すべての Web サーバ用仮想マシンにおいて lsync の同期先 IP アドレスを設定し，lsync を再起動させる．
② Moodle の wwwroot にロードバランサの IP アドレスによる URI に設定を変更する．
③ LVS DR 向けの設定として，CentOS の ARP を設定後，ループバックインターフェースにロードバランサの IP アドレスを設定する．
(4) 仮想マシン停止イベントを受信した場合，以下の処理を行う．
① すべての Web サーバ用仮想マシンにおいて lsync の同期先設定から停止した仮想マシンの IP アドレスを削除し，lsync を再起動させる．

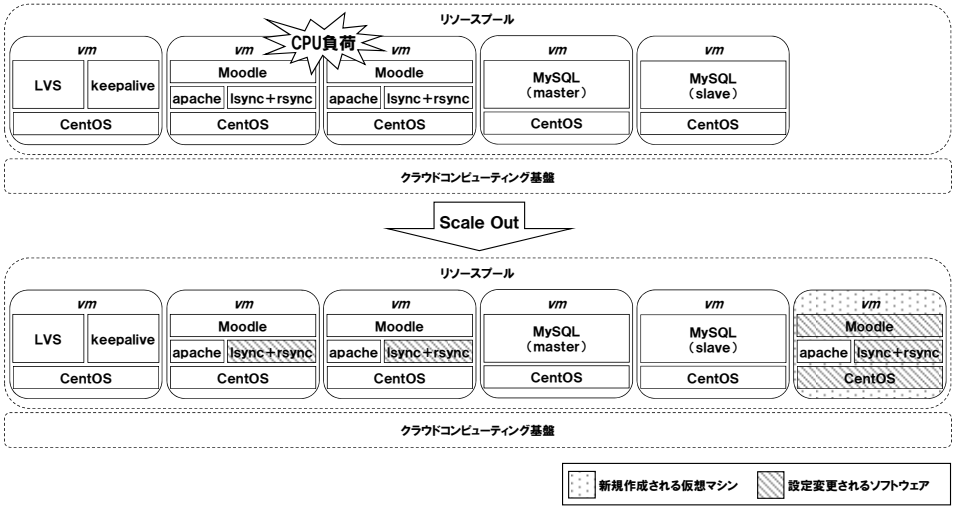

図 4　オートスケールの動作

6. 評価

Moodle 自動管理スクリプトの開発によって実現させたスケーラブルな Moodle 環境を評価するため, 仮想マシンの障害時および CPU リソース不足時における復旧やスケールアウトまでの時間を測定した．測定した時間は以下の 4 パターンである．

(1) ロードバランサ用仮想マシン復旧時間
ロードバランサ用仮想マシンの障害発生から，ロードバランサ用仮想マシンが復旧し，各アプリケーション設定が完了するまでの時間

(2) DB 用仮想マシン復旧時間
DB 用仮想マシンの障害発生から，DB スレーブサーバ用仮想マシンが復旧し，各アプリケーション設定が完了するまでの時間

(3) DB スレーブサーバからマスタサーバへの昇格時間
DB マスタサーバとなっている DB 用仮想マシンの障害発生から，スレーブサーバをマスタサーバに昇格させるための各アプリケーション設定が完了するまでの時間

(4) Web サーバ用仮想マシンのオートスケール時間
Web サーバ用仮想マシンの CPU 使用率が最大閾値を超えてから，Web サーバ用仮想マシンがスケールアウトし，各アプリケーション設定が完了するまでの時間

測定は，CPU に Intel(R) Xeon(R) CPU E5502 @ 1.87GHz を使用し，メモリは 4GByte の物理マシン上の環境において実施した．仮想マシン復旧時間・オートスケール完了時間ともに，アプリケーションレイヤへの設定が完了し，サービスが提供可能になった時点とし，10 回実施した測定結果の平均値を算出した．測定結果を表 1 に示す．

この結果，DB のマスタに障害が発生した場合，スレーブがマスタに昇格し，DB が復旧するまでの所要時間は 1 分以下と十分な結果が得られた．しかし，仮想マシン復旧時間はオートスケール完了時間の倍以上となる結果となった．これについては，HA と Scalability の監視時間の違いと，アプリケーションレイヤへの設定項目数の違いによるものと考えられる．また，全体的にスレーブからマスタへの昇格時間以外は所要時間が長い印象を受ける．これについては評価環境に使用した物理マシンの性能が十分でないため，仮想マシンの起動に時間を要していることが原因の一端であると考えられる．

表 1　測定結果

	監視周期	復旧時間
ロードバランサ用仮想マシン復旧時間	100 秒	4 分 14 秒
DB 用仮想マシン復旧時間	100 秒	4 分 01 秒
DB スレーブからマスタへの昇格時間	60 秒	0 分 56 秒
Web サーバ用仮想マシンオートスケール時間	60 秒	1 分 47 秒

7. 関連研究

LMS などのスケーラビリティに関する研究や高可用性の実現に向けた構築は様々行われている．王ら[13]は，Moodle に対してスケーラビリティを実現するため，OSS のロードバランサや共有ストレージを用いてスケールアウトする Moodle サイト負荷分散システムを提案し，その有効性を検証している．また，玉城ら[14]は，分散 Key-Value ストアデータベース Cassandra を用い，スケーラビリティのある CMS を設計し，100 台の PC クラスタによってその有効性を検証している．

LMS などの高可用性を実現するために, 仮想マシンモニタを活用したシステム構築, 技術開発に関する研究も活発におこなわれている．桝田ら[15]は，システム障害の発生が与える影響に対して, 低運用コストで可用性を向上させるように工夫した Moodle システムの設計･実装をしている. 仮想マシンを利用して, 物理サーバ 2 台構成で RAID による冗長化させ，各物理サーバには仮想マシンに Web サーバ，DB サーバを稼働させて lsync でデータの同期をおこなっている．従来技術を組み合わせることで，低コストかつ可用性の高いシステムを実現している．また，田村ら[16]は，障害時でもアプリケーションや OS に依存せずにサービスを継続させるため，仮想マシン間の同期による対故障クラスタリング技術 Kemari を開発した．Kemari は，運用系の仮想マシンと待機系の仮想マシンとの同期の契機に，外部の状態遷移を起こすイベントを利用した高可用化技術である．これにより，継続性をもった耐障害性を実現している．

8. おわりに

本研究では，開発中のクラウドコンピューティング基盤 Kumoi バージョン 2.0 を用いたスケーラブルな Moodle 環境を構築した．リソースプール内の仮想マシンは，Kumoi の HA 機能および Scalability 機能によって障害および CPU 負荷に対応できる．また，この際に発生する仮想マシンの起動・停止イベントから仮想マシン上の Moodle を動作されるために構成されたアプリケーションの設定を行う Moodle 自動管理スクリプトを開発した．これにより，仮想化されたマシンの障害および CPU 負荷に対しては自動的に復旧またはスケールする Moodle 環境を構築することができた．

このように，本環境は仮想化されたマシンの負荷および障害に自動対応できるものとして構築した．このため，物理マシン障害やアプリケーション障害においては対応できていない部分が多い．アプリケーション障害においては，ロードバランサの死活監視機能によって，アプリケーションのある一定の障害に対応できている．また，Kumoi は一部の物理マシンに障害が発生しても動作し続ける機能を備えているが，本研究で開発した Moodle 管理スクリプトによって生成される各監視プロセスは物理マシン障害に対応していない．今後はこれらプロセスの高可用性を実現し，システム全体に及ぶ，より包括的な高可用性の実現を目指す．

謝辞 本研究は，総務省 SCOPE「ディペンダブルな自律連合型クラウドコンピューティング基盤の研究開発」の支援を受けている．

参考文献

1) 放送大学，平成 21 年度・22 年度先導的大学改革推進委託事業「ICT 活用教育の推進に関する調査」委託業務成果報告書 (2011).
2) 独立行政法人メディア教育開発センター, 2008 年度 e ラーニング等の ICT を活用した教育に関する調査報告書 (2009).
3) Wikipedia:高可用性, http://ja.wikipedia.org/wiki/高可用性
4) Kumoi, http://code.google.com/p/kumoi/
5) Akiyoshi Sugiki, Kazuhiko Kato, Yoshiaki Ishii, Hiroki Taniguchi, Nobuyuki Hirooka, “Kumoi: A High-Level Scripting Environment for Collective Virtual Machines”, IEEE 16th International Conference on Parallel and Distributed Systems, Shanghai, China, pp.322-329 (2010).
6) 相川拓也，杉木章義，石井嘉明，谷口寛季，廣岡誠之，加藤和彦, “高水準な記述によるクラウド研究開発環境の構築”，第 21 回コンピュータシステム・シンポジウム（ComSys2009）ポスター発表，情報処理学会 (2009).
7) Moodle, http://moodle.org/
8) 日経コミュニケーション, “ネットワーク大事典”，日経 BP 社 (2004).
9) NIST, “NIST Definition of Cloud Computing v15”, http://www.nist.gov/itl/cloud/upload/cloud-def-v15.pdf (2010).
10) TechTarget ジャパン, “サーバ仮想化運用管理に関するアンケート調査リポート”, http://techtarget.itmedia.co.jp/tt/news/1108/19/news06.html (2011).
11) rsync, http://rsync.samba.org/
12) 杉木章義，加藤和彦, ”国産クラウド基盤ソフトウェア Kumoi 開発の現状と今後について”, 第 22 回コンピュータシステム・シンポジウム（ComSys2010）ポスター発表，情報処理学会 (2010).
13) 王躍,小柏香穂理,刈谷丈治,小河原加久,”OSS に基づいた Moodle サイトのスケーラビリティに関する報告，情報処理学会研究報告，インターネットと運用技術, Vol.2011-IOT-14, No.2, pp.1-5 (2011).
14) 玉城将士，谷成雄，河野真, ”Cassandra を使ったスケーラビリティのある CMS の設計”，情報処理学会研究報告，システムソフトウェアとオペレーティング・システム, Vol.2011-OS-117, No.23, pp.1-6 (2011).
15) 桝田秀夫,村田和義,渋谷雄,”低コストな高可用性と学務システム連携を考慮した Moodle システムの構築”，情報処理学会研究報告，インターネットと運用技術, Vol.2008, No.37, pp.65-69 (2008).
16) 田村芳明，柳澤佳里，佐藤孝治，盛合敏, ”Kemari: 仮想マシン間の同期による耐故障クラスタリング”，情報処理学会論文誌．コンピューティングシステム, Vol.3, No.1, pp.13-24, 2010-03-16