Vol.2020-SE-204 No.15
2020/3/3

サーバ状況確認履歴の多数決によるサーバ管理演習のための正誤判定支援システムの提案

西村　一輝[1]　井垣　宏[1]　尾花　将輝[1]

概要：サーバ管理演習とは，Web サービスの構築などに必要なサーバに関する基礎的な知識やノウハウを実習形式で行う演習である．我々は先行研究において，学生の進捗状況の把握や課題の誤り原因の特定を目的として，サーバ管理演習における 3 種類の履歴情報をログとして記録・可視化するログ収集可視化環境の構築を行った．本稿では，履歴情報の一つであるサーバ状況確認履歴を用いて演習課題ごとの正誤判定を支援する仕組みについて述べる．

1. はじめに

サーバ管理技術は，Web 技術などを利用した様々な分野に置いて必要不可欠な技術の一つである．近年では，クラウドコンピューティング [4] や IoT，AI 技術などの発展により，サーバ管理技術を有するエンジニアの需要がますます高まりつつある．大学などの教育機関においてもサーバ管理技術に関わる教育が数多く行われており，大学院生を対象とした教育プロジェクトである enpit プロジェクト [9] では，Amazon Web Service(AWS)[*1]の仮想サーバを利用した演習が行われている．他にも Hollingsworth ら [2] の Google Cloud Platform(GCP)[*2]を用いた Web アプリケーション開発に関する教育など，クラウドコンピューティング技術を用いてサーバ管理技術を学ぶ実践的な演習が様々な教育機関で多数開講されている．

一般に，サーバ管理技術に関連する演習（サーバ管理演習と呼ぶ）では，学生は SSH などを利用してサーバにアクセスし，Web サーバのインストール等の与えられた課題を達成するためのコマンドの実行やファイルの編集を行う．このようなサーバ管理演習では，学生は多くのコマンド実行や設定ファイルの編集等を行うため，演習課題に対して誤った操作を実行してしまったり，理解不足のために何をすればよいかわからなくなってしまったりすることで，演習を進められなくなってしまう学生も多い．不具合が発生したときは，学生自身で不具合の原因を修正するか，教員などのサポートを通して，不具合の原因に対する処理方法などを助言してもらい，演習を進めていく．一方で，サーバ管理演習は学生個人個人で課題に取り組む形式となっているため，誰がいつどのような不具合で行き詰まるかは各学生の PC 操作やサーバ関連の知識などによって大きく変わる可能性がある．結果として，教員はひとりひとり異なる学生の進捗状況を正確に把握することが困難になる．

また，学生が演習用サーバに不具合を発生させた場合，教員はサーバの不具合の原因をソフトのインストール状況や設定ファイルなどを確認する事で特定しなければならない．しかし，現在のサーバ状況は容易に確認できるが，過去に遡って学生がいつ，どこで，どのような操作をして不具合のあるサーバの状態になったのかを正確に確認することは困難な場合も多い．

そこで，我々は先行研究において [11][14][10][12] サーバの履歴情報をログとして記録し，可視化するログ収集・可視化環境の構築を行った．これにより Web ブラウザを利用して，学生が演習課題に行き詰まった際に，サーバの履歴情報と学生の操作履歴から誤った際のサーバの状況を確認することが可能となる．さらに本稿では，これまでの先行研究で構築したログ収集・可視化環境を利用し，学生が課題を正しく実行できているかの判定を支援する正誤判定支援システムについて詳述する．

2. サーバ管理演習とその課題

2.1 サーバ管理演習の流れ

サーバ管理演習とは，課題であるサーバソフトウェアのインストールや設定ファイルの作成・編集などを通して，サーバ構築・管理に必要な基礎的な技術を学ぶ演習である．

学生は教員が演習前に用意した複数台のサーバを利用し

[1] 大阪工業大学　情報科学研究科
Osaka Institute of Technology 1-79-1 Kitayama, Hirakata City, Osaka, 573-0196 Japan

[*1] https://aws.amazon.com/jp/

[*2] https://cloud.google.com/?hl=ja

て演習を実施していく．最近では，AWS や GCP といったパブリッククラウドを利用した仮想サーバが利用されることも多い．学生は演習開始とともに，図 1 のように SSH クライアントを用いて各学生の演習環境にログインし，演習用課題を順次解き進めていく．

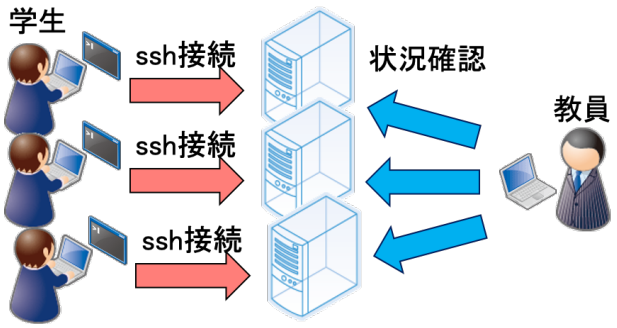


図 1　サーバ管理演習

学生が演習課題を正しく進められている場合は問題はない．しかし，こうした演習では，学生はサーバを構築・管理する為に，様々なディレクトリで多くのコマンド実行や Web サーバの設定ファイルの編集などを行う為，教員が想定しない操作を学生が行う事によりサーバに不具合が発生することも多い．こうしたサーバの不具合には様々なものが考えられる．例えば，ソフトのインストール漏れや誤ったコマンドの実行，設定ファイルにおける記述ミスなどである．学生はサーバが指示通りに動作しないことで異変に気づく．不具合が見つかった場合，学生はサーバの不具合の原因を特定し，サーバ設定の修正などを自身で行うか，教員や TA(Teaching Assistant) と呼ばれるサポートスタッフに質問を行い不具合の原因特定とその修正方法について助言をしてもらい，課題を実施していく．以上がサーバ管理演習の一般的な流れである．

2.2　既存のサーバ管理演習

こうした演習は大学などの様々な教育機関で開講されている．例えば、文部科学省の教育プロジェクトである大学院生を対象とした enpit プロジェクトのクラウドコンピューティング分野である Cloud Spiral では，演習を通して Web アプリケーションに関する基礎的な技術を学ぶ事ができる．この演習では，Amazon Web Service のようなパブリッククラウドと言われるクラウドサービスを利用することで，学生一人ひとりにそれぞれ仮想サーバが与えられ，演習が行われる [9]．同様に AWS を利用したマイクロサービスの開発・運用に関する教育についての研究が Ortiz[5] によって行われている．他にも，Hollingsworth ら [2] の Google Cloud Platform[*3]を用いた Web アプリケーション開発に関する教育や横山ら [6][7] のクラウドコンピューティング技術を用いた教育プラットフォームに関する研究のように，他の異なるクラウドプラットフォームを利用した研究や教育も多岐に渡る．また，開発とサーバ管理・運用の一体化を目指した DevOps[3] 教育を学ぶ研究 [1] も行われている．

鎌田ら [8] や中崎ら [15] は OpenStack を利用し，容易に演習を実施する環境が構築可能なサーバ管理演習のための支援システムの提案をしている．これらの仮想化技術を利用した支援環境を利用することで，サーバ演習を実施するための Linux 環境等を容易に導入することが可能となる．サーバ環境は Linux OS を対象としたものが多いが，授業によっては Windows Server や特殊な組み込み系の OS などを対象とすることもある．本研究では，仮想化技術を用いた Linux OS を対象としたサーバ管理演習を想定している．

2.3　サーバ管理演習の課題

サーバ管理演習では，通常のプログラミング演習と異なり，様々なディレクトリに移動してコマンド実行や設定ファイルの編集を行う．そのため演習を行う中で，タイピングスピードや CUI などの PC 操作に慣れているか，サーバに関するノウハウを演習以前から保有しているなどにより，演習を行うスピードが学生一人ひとりで大きく異なってしまう．また，教員は演習中に極端に進み具合が悪い学生がいないかなどを知る為に，全学生の進捗状況を把握する必要がある．そこで口頭により学生の進み具合を確認する場合もあるが，学生自身が正確に自分の進捗状況を把握出来るとは限らない．その為，教員は全ての学生の進捗状況を正確に把握することは困難である．よって，教員は課題の進捗状況を口頭で確認するだけでなく，学生毎のサーバがどのような状態にあるのかを定期的に確認する為の方法が必要となる．

また，演習中に学生がサーバの不具合を発生させた場合，教員は学生のサーバ状態を確認することで不具合の原因特定を行う．これは，ログファイルやインストールされたソフトウェア，設定ファイルの編集状況等を把握する事で確認可能であるが，学生がいつ，どこで，サーバが不具合を含む状態になってしまったのかを後から正確に把握することは困難である．また，学生がどのような原因で不具合を引き起こすのかを教員が正確に把握することは教育的に非常に重要な情報となりうる．

そこで，我々は先行研究においてサーバ管理演習における 3 種類の履歴情報をログとして記録するログ収集環境の構築 [11][14] を行った．これにより学生の履歴情報からサーバの状態を確認することが可能となる．また，これらのログを Web ブラウザを用いて容易に確認できるような可視化環境の構築 [10][12] を行った．これらの環境を教員がより容易に利用出来る事を目的として，履歴情報の一つであるサーバ状況確認履歴を利用した正誤判定支援システ

[*3] https://cloud.google.com/?hl=ja

ム [13] の仕組みを提案した．本稿ではこの正誤判定支援システムを実際に構築し，判定内容を可視化したサーバ状況確認履歴表示画面を示すとともに，この環境を利用した際のケーススタディについて詳述する．

3. 先行研究

3.1 サーバ管理演習における 3 種類の履歴情報

サーバ管理演習は，学生が課題で指定されたサーバ状態に合わせてコマンド実行やファイル編集等の操作を行う事で，目的を達成していく．例えば，Web サーバを構築し、接続するポート番号の設定ファイルを変更する課題であれば，yum などのパッケージ管理コマンドを用いて Web サーバである Apache のインストールを行い，service コマンドや systemctl コマンドなどを用いて Web サーバを起動させる．ブラウザなどを使用し，Web サーバのテストページに接続を行い，正しく表示されるのかを確認する．確認できれば課題で指示された状態にする為に，Apache の設定ファイルを編集し，Web サーバを再起動をさせる．課題で指示されたポート番号を指定して，Web サーバのテストページが正しく表示される事を確認できればサーバの構築が完了する．

そこで我々は学生が課題を通して行った一連の操作をログとして記録する．学生の行った操作を振り返ることができる様にする為に，コマンド実行履歴，ファイル編集履歴，サーバ状況確認履歴の異なる 3 種類のログの収集を行う．以降ではこの 3 種類のログについて詳述する．

3.1.1 コマンド実行履歴

コマンド実行履歴はいつどのユーザがどういったコマンドをどの場所（ディレクトリ）で実行し，そのレスポンスが何であったかを記録するものである．具体的には以下の項目をコマンド実行履歴として収集する．

- id: 学生番号 (host 名)
- date: 実行した時間
- dir: 実行したディレクトリ
- cmd: 実行したコマンド
- res: 実行した結果

3.1.2 ファイル編集履歴

学生は演習の中でインストールしたサーバソフトウェアを課題の意図に合わせた動作させる為に，様々な設定ファイルを編集する必要がある．ファイル編集履歴は誰が，いつ，どのファイルをどのように変更したかを差分情報として記録するものである．例えば，“/etc/httpd/” などのサーバ管理演習において学生が修正する必要があるディレクトリを監視し，変更が行われるたびにファイル編集履歴として記録する．

3.1.3 サーバ状況確認履歴

想定しているサーバ管理演習において，教員は演習用サーバに対して何らかのコマンドを実行することにより，学生のサーバの状態を課題の段階毎に調べることは可能である．例えば，Apache などのサーバソフトウェアが正しくインストールできているかやサーバが正しく起動しているのか，演習用のサーバ外からサーバに対して，ブラウザまたは，curl コマンドを用いて，アクセスが可能か，さらにレスポンスが正しいのかなどである．その為，教員が学生のサーバ状況を確認する方法として 2 種類の方法がある．サーバ内にログインをして特定のソフトウェアがインストールされているかなどをコマンド実行などで確認可能なものと，サーバ内にはログインせずに Web アプリケーションなどのサービスが特定のポートでアクセス可能であるかを確認するものである．そこで，サーバ状況確認用のスクリプトやコマンドを実行する環境を用意し，どのようなスクリプトがいつ，どの学生の環境に対して実行され，結果がどうであったかをサーバ状況確認履歴として記録する．

- id: 学生番号 (host 名)
- date: 実行した時間
- host: 実行した host の IP アドレス
- script: 課題に対応したシェルスクリプト，またはコマンド
- res: 実行結果
- step: 何番目のサーバ状況確認コマンドか

サーバ状況確認履歴のログは，ログの収集時刻である Unix タイム， 学生を識別する為の ID，実行したスクリプト，またはコマンドとその実行結果がテキストファイルに空白区切りで記録されている．サーバ状況確認履歴のログファイルは，テキストファイルの内容を保存する項目に切り出して，データベースに記録を行う．ここで，確認後の結果が正しいかどうかの判定は教員のエフォート増大を考慮し，実施しない．教員はあくまで確認のためのコマンドを作成し，登録する．

以上の 3 種類のログを収集することで，学生がいつどのようなコマンドを実行し，ファイルを編集したかを後から正確に振り返って確認することが可能となる．また，サーバ状況確認履歴が記録されることで，学生がどの段階の演習まで正常に進めており，どの段階で不具合を発生させたのかを把握できる．また，複数のログを照らし合わせる事により，正常であることが確認されてから不具合が確認されるまでに学生がどのような操作を行ったのかを確認することが可能となる．これらのログを収集する [11][14] ログ収集環境の構築を行った．以降はこの環境について詳述する．

3.2 ログ収集環境

本稿では仮想化環境を用いる事を前提としている為，サーバ管理演習では演習用サーバのテンプレートファイルを用意し，テンプレートファイルにもとづいて学生ごとのサーバ環境を作成する．このテンプレートファイルに対して，

コマンド実行履歴，ファイル編集履歴の収集環境を導入し，学生の操作履歴を収集する．教員はこのテンプレートファイルを用いて学生人数分仮想サーバを複製することで演習用のサーバ環境を構築する．また，学生の進捗状況の把握を行う為にはサーバ状況確認履歴を収集する必要がある．その為，演習用サーバとは別にログ収集用サーバを教員が用意し，その中でログ収集を行う．図 2 にログ収集環境のアーキテクチャを示す．

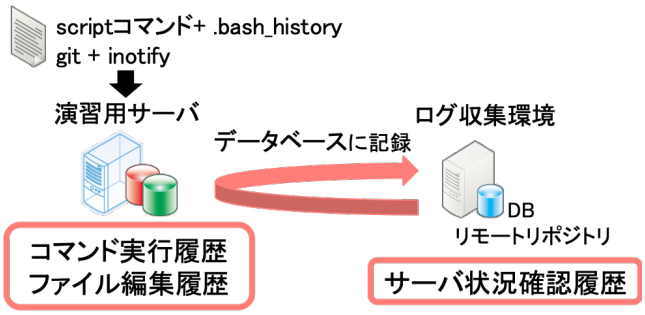


図 2　ログ収集環境のアーキテクチャ

図 2 の環境で演習が開始されると，学生はテンプレートファイルから作成された演習用サーバに SSH クライアントを利用してログインを行う．学生がログインすると script コマンド[*4]と history コマンド[*5]が起動する．script コマンドは，コマンドの実行結果を収集する為に広く利用されるコマンドである．このコマンドを使用すると，学生が演習用のサーバにログインを行ったタイミングからログアウトを行うタイミングまでに端末上で表示された画面の全ての情報をテキストファイルとして記録する事ができる．学生が演習の為の様々なコマンド実行を行うと，コマンドとコマンドの実行結果が，history.log と/var/log/script.log に保存される．

また，ファイル編集履歴の収集には inotify と lsyncd[*6]を用いる．学生が設定ファイルなどの編集を行うたびに，inotify 機能[*7]によって通知される．我々は通知のたびに変更されたファイルをディレクトリ構造ごと /git ディレクトリに同期する．さらに，/git ディレクトリを git[*8]と呼ばれる分散型版管理システムを利用してリポジトリにコミットすることで，編集履歴を git リポジトリ上に保存する．

ログ収集環境では，これら 2 つの履歴情報を定期的に収集すると共に，ログ収集環境からサーバ状況確認コマンドを演習用サーバに対して実行し，その結果と収集を行ったコマンド実行履歴をデータベースに保存する．また，ファイル編集履歴はリモートリポジトリを保存する gitbucket[*9]などのリモートリポジトリを利用してデータを保存する．

しかし，このままではログデータを直接参照することでしか，学生の状況を把握することが出来ない．そこで可視化環境の構築を行った．以降は可視化環境について詳述する．

3.3　3 種類のログの可視化環境

収集したログ情報を容易に確認できる事を目的とした可視化環境を Web システムとして構築した [10][12]．本稿では，この可視化環境のうち，正誤判定支援システムに利用するサーバ状況確認履歴表示画面について詳述する．

3.3.1　サーバ状況確認履歴表示画面

サーバ状況確認履歴は教員が学生の課題の進み具合について調べるコマンドやスクリプトを記録した履歴である．この履歴を可視化することにより，特に複数の学生の進捗状況の把握する際に有用であると考えられる．そこで，任意の時間におけるサーバ状況を容易に確認できるサーバ状況確認履歴表示画面を作成した．

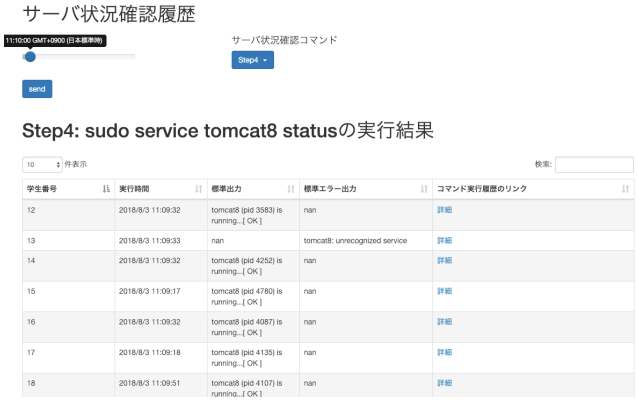


図 3　サーバ状況確認履歴表示画面

図 3 がサーバ状況確認履歴表示画面である．画面にはスライドバーとドロップダウンリストがあり，それぞれ時刻情報とサーバ状況確認コマンドが選択できる様になっている．

図 3 は実際に時刻として 11 時 10 分を，サーバ状況確認コマンドとして，Step4 の sudo service tomcat8 status を選択した時の画面を示している．この画面では，サーバ状況確認コマンドの実行結果である学生番号，実行時間，標準出力，標準エラー出力が表示されている．このテーブルは表示されている項目毎にソーティング機能がある為，標準出力やエラー出力が他と異なっている学生なども容易に確認することができる．実際に図 3 では，学生番号が 13 のサーバでまだ tomcat8 がインストールされていないことが確認できる．

*4　https://linuxjm.osdn.jp/html/util-linux/man1/script.1.html
*5　http://linux.sfttec.com/linux_command/history.html
*6　https://axkibe.github.io/lsyncd/
*7　https://linuxjm.osdn.jp/html/LDP_man-pages/man7/inotify.7.html
*8　https://git-scm.com/
*9　https://github.com/gitbucket/gitbucket

可視化環境であるサーバ状況確認履歴表示画面により，学生の進捗状況の把握が可能になった．本稿では説明を省略したコマンド実行履歴表示画面とファイル編集履歴表示画面と組み合わせることで，学生が課題に行き詰まった場合に，より詳細な不具合の原因特定が可能となる．しかし，今の可視化環境では学生が正しく課題を進められているかどうかの判断は教員自身が複数のログを確認するしかなく，教員のスキルに依存するところが大きい．そこで本研究では，サーバ状況確認履歴を用いて学生の進捗状況の正誤を教員が判定しやすい様にする正誤判定支援システムを提案する．

4. 正誤判定支援システム

サーバ状況確認履歴は学生 ID(SID とする), ログ収集時刻（Unixtime），サーバ状況確認コマンド，標準出力から構成されている[*10]．これらの履歴に，学生の進捗状況の正誤判定を支援する為の処理として，以下の 4 つのステップを適用する．表 1 に PHP をインストールする課題におけるサーバ状況確認履歴を示す．実行しているサーバ状況確認コマンドは，yum list installed |grep php である．この表を例に各ステップの内容について詳述する．

表 1 分析前のサーバ状況確認履歴

SID	標準出力	収集時刻
1	nan	1580201853
2	php.x86_64 … php-mbstring.x86…	1580201854
3	php.x86_64 … @updates	1580201855
4	php.x86_64 … php-mbstring.x86…	1580201855
5	php.x86_64 … php-mbstring.x86…	1580201856
6	nan	1580201857
7	nan	1580201858

4.1 正誤判定処理の手順

4.1.1 Step1. Updatetime カラムの更新

表 1 のログ情報はサーバ状況確認コマンドを実行した結果が保存されており，サーバに何らかの変更が加えられると標準出力が変化する．SID3 の標準出力結果が表 1 に示された値に変更された時，表 2 に示す様に Updatetime が更新された．表 2 では，これまでに収集されているサーバ状況確認履歴の収集時刻と標準出力，Updatetime カラムの一部を示している．Updatetime カラムの初期値はログが収集された最初の時刻となっており，このログ情報の場合，一度も Updatetime カラムが更新されていない為，Updatetime の初期値は 1580199513 となる．演習が進むにつれログが収集されていき，標準出力の結果が一つ前と比較した時に nan から php.x86_64 … @updates に変化している．この時 Updatetime カラムの値を 1580199513 から現在の収集時刻 1580201855 に更新する．この操作を全ての学生のサーバ状況確認履歴についておいて行う．

表 2 学生 id3 のサーバ状況確認履歴

収集時刻	標準出力	Updatetime
1580201317	nan	1580199513
1580201499	nan	1580199513
1580201674	nan	1580199513
1580201855	php.x86_64 … @updates	1580201855

4.1.2 Step2. 標準出力のクラスタリング

演習を行う全学生を対象として一定の間隔ごと (通常数分程度）にサーバ状況確認履歴を収集している．サーバ状況確認履歴の標準出力を対象として，一度に収集された学生人数分の結果の文字列を，クラスタリング分析を行うことによりカテゴリー分けする．表 1 の標準出力の結果から，SID 1,6,7 と SID 2,4,5，SID 3 のそれぞれが同一のカテゴリであると判定される．文字列が完全一致していると同一のカテゴリに分類され，一部でも異なると別のカテゴリに分類される．

4.1.3 Step3. カテゴリ毎の Updatetime 平均値の算出と変更されていない履歴の特定

カテゴリ番号毎に Updatetime の平均値を算出する．表 3 の場合，カテゴリ 1 の平均値は 1580195371, カテゴリ 2 の平均値は 1580200118，カテゴリ 3 の平均値は 1580201855 となる．算出された平均値が小さいカテゴリほどサーバの状態が長時間変化していない学生が多いことを表す．そういったカテゴリが課題における正解とは考えにくい．そこで，Updatetime の平均値が最も小さいカテゴリに Status として No chage とする．表 3 の場合は，カテゴリ 1 が No Change となる．

4.1.4 Step4. 多数決による正誤判定支援処理

最後に残ったカテゴリ毎の要素数で多数決を行い，多いものを Correct，少ないものを Incorrect とし，Status カラムを更新する．表 3 の場合，カテゴリ 3 が Incorrect，カテゴリ 2 が Correct と推定される．これはサーバに対して正しい処理を行うことができる学生の方が，間違った処理を行う学生よりも多いという想定にもとづくものである．

表 3 分析後のサーバ状況確認履歴

SID	標準出力	Updatetime	Status
1	nan	1580195369	No Change
2	php.x86_64 … php-mbstring.x86…	1580199513	Correct
3	php.x86_64 … @updates	1580201855	Incorrect
4	php.x86_64 … php-mbstring.x86…	1580199341	Correct
5	php.x86_64 … php-mbstring.x86…	1580201500	Correct
6	nan	1580195372	No Change
7	nan	1580195372	No Change

[*10] 標準エラー出力も保存されているが，本稿では対象としない

4.2 正誤判定支援システムの可視化

上記で述べた 4 ステップの方法で学生ごとのサーバ状況確認履歴について正誤判定を行う．サーバ状況確認履歴では，正誤判定結果にもとづいて，各履歴を色分けして表示する．例えば，上記で述べたログ情報を元に可視化を行うと，図 5 の様になる．

学生ID	収集時刻	実収集時刻	サーバ状況確認コマンド	IPアドレス	標準出力	標準エラー出力	コマンド実行履歴
1	2020年01月28日 (火) 17時57分46秒	2020年01月28日 (火) 17時57分33秒	yum list installed | grep php	150.89.223.100	N/A	N/A	詳細
2	2020年01月28日 (火) 17時57分46秒	2020年01月28日 (火) 17時57分34秒	yum list installed | grep php	150.89.223.101	php.x86_64 5.4.16-46.1.el7_7 @updates php-cli.x86_64 5.4.16-46.1.el7_7 @updates php-common.x86_64	N/A	詳細
標準エラー出力: N/A 標準出力: php.x86_64 5.4.16-46.1.el7_7 @updates php-cli.x86_64 5.4.16-46.1.el7_7 @updates php-common.x86_64 5.4.16-46.1.el7_7 @updates php-mbstring.x86_64 5.4.16-46.1.el7_7 @updates							
3	2020年01月28日 (火) 17時57分46秒	2020年01月28日 (火) 17時57分35秒	yum list installed | grep php	150.89.223.102	php.x86_64 5.4.16-46.1.el7_7 @updates php-cli.x86_64 5.4.16-46.1.el7_7 @updates php-common.x86_64	N/A	詳細
標準エラー出力: N/A 標準出力: php.x86_64 5.4.16-46.1.el7_7 @updates php-cli.x86_64 5.4.16-46.1.el7_7 @updates php-common.x86_64 5.4.16-46.1.el7_7 @updates							
4	2020年01月28日 (火) 17時57分46秒	2020年01月28日 (火) 17時57分35秒	yum list installed | grep php	150.89.223.103	php.x86_64 5.4.16-46.1.el7_7 @updates php-cli.x86_64 5.4.16-46.1.el7_7 @updates php-common.x86_64	N/A	詳細
5	2020年01月28日 (火) 17時57分46秒	2020年01月28日 (火) 17時57分36秒	yum list installed | grep php	150.89.223.104	php.x86_64 5.4.16-46.1.el7_7 @updates php-cli.x86_64 5.4.16-46.1.el7_7 @updates php-common.x86_64	N/A	詳細
6	2020年01月28日 (火) 17時57分46秒	2020年01月28日 (火) 17時57分37秒	yum list installed | grep php	150.89.223.105	N/A	N/A	詳細
7	2020年01月28日 (火) 17時57分46秒	2020年01月28日 (火) 17時57分38秒	yum list installed | grep php	150.89.223.106	N/A	N/A	詳細

図 4 分析後のサーバ状況確認履歴表示画面

可視化は表 3 の Status 情報を元に Correct なら緑色，Incorrect なら黄色，無色であれば No Change に色分けされる．今回のログの場合，標準出力の詳細を確認すると演習中に利用する PHP で文字列を扱う為の mbstring というソフトがインストールされていない事が確認できる．このように，教員は演習を行う中で黄色と判定されている学生に注目することで，進捗状況の異なる学生を素早く見分ける事が出来る．

5. ケーススタディ

正誤判定支援システムがどのように動作するかを確認するため，Web サーバや各種アプリケーションをインストールするサーバ管理演習を作成した．具体的な内容としては，Web サーバである Apache のインストールと起動を行ったあと，Web アプリケーションサーバである tomcat をインストールし，gitbucket[*11] と pukiwiki[*12] と呼ばれる二つの Web アプリケーションをデプロイする課題を用意した．

gitbucket とは git のリポジトリをホスティングする Web アプリケーションである．今回の演習では，gitbucket を動作させる為に，Tomcat のインストールと Apache のリバースプロキシの設定をあわせて実施してもらった．pukiwiki は PHP で動作するオープンソースウィキソフトウェアである．サーバ環境に PHP を導入した後，Apache のドキュメントルートに配置し，pukiwiki の所有者の変更と Apache の再起動を行うことで動作させる事が可能となる．

*11 https://github.com/gitbucket/gitbucket
*12 https://pukiwiki.osdn.jp/?PukiWiki

学部 3 回生が 4 名，4 回生が 2 名の合計 6 名でこれらの課題を実施した．サーバ状況確認コマンドとして 17 個のコマンドを用意し，3 分に 1 回の間隔でログ収集を行った．サーバ状況確認履歴表示画面がどのようになったかについて述べる．

5.1 正常な動作をしたサーバ状況確認履歴表示画面

図 5 に示すとおり，サーバの状態は 3 種類に分かれている．緑で表示されているログは正誤判定支援処理により，Correct と判定された状態である．つまり，分析を行った結果，同様の結果になる学生が多くいた為，正解に近いデータカテゴリであると判定されている．また色が付いていない状態のログは No Change と判定されている為，結果が前の状態から変化していないデータカテゴリであると判定されている．そして，黄色で表示されているログは Incorrect カテゴリであると判定されている為, 状態が変化しているにも関わらず正解ではない状態の学生が含まれている可能性が高い．つまり，その演習課題について行き詰まっている可能性がある．その為，教員は黄色のログデータに特に注視することで課題に行き詰まった学生のサポートをスムーズに行える可能性がある．図 5 の標準出力の結果を良く見ると，緑色のカテゴリデータには tomcat-webapps.noarch というソフトウェアがインストールされているのに対して，黄色のカテゴリデータには tomcat-webapps.noarch というソフトウェアがインストールされた形跡が無い．その為，この学生は tomcat-webapps をインストールし忘れている事がログから確認できる．

5.1.1 正誤判定支援システムにおける課題

学内で行われたサーバ管理演習に対してサーバ状況確認履歴を収集し，正誤判定支援処理を行った．その中から分析するにあたっての問題点について詳述する．

標準出力によってクラスタリングを行った中から，変更がありかつカテゴリ毎の要素数が最も多いときに課題に対する正しい状態と推定している．その為，標準出力によるクラスタリング結果によっては正誤判定を正しく行えない．Web サーバを対象としたサーバ状況確認履歴の場合，標準出力にはマシンや実行インスタンスごとに異なるプロセス ID や時刻情報などの数値を含む文字列が表示されており，そのままでは正しく文字列比較ができない．このような値を含む文字列の比較を行うためには，数文字列の適切な正規化が必要となるが，事前にすべての文字列を対象として適切な正規化ルールを実装することは困難である．その為，あらゆる出力を想定した類似度判定の仕組みについての検討が必要である．現状のシステムでは，数値文字を全て特定の一文字に置換することで，数値が異なることによる文字列の違いを吸収する対策を行っている．

また，標準出力のクラスタリング分析を行った結果，図 6 のような結果になるログデータがあった．図 6 は systemctl

学生ID	収集時刻	実収集時刻	サーバ状況確認コマンド	IPアドレス	標準出力	標準エラー出力	コマンド実行履歴
1	2020年01月28日 (火) 16時57分44秒	2020年01月28日 (火) 16時57分17秒	yum list installed | grep tomcat	150.89.223.100	N/A	N/A	詳細
2	2020年01月28日 (火) 16時57分44秒	2020年01月28日 (火) 16時57分18秒	yum list installed | grep tomcat	150.89.223.101	tomcat.noarch 7.0.76-9.el7_6 @base tomcat-el-2.2-api.noarch 7.0.76-9.el7_6 @base tomcat-jsp-2.2-api.noarch	N/A	詳細
3	2020年01月28日 (火) 16時57分44秒	2020年01月28日 (火) 16時57分18秒	yum list installed | grep tomcat	150.89.223.102	N/A	N/A	詳細
4	2020年01月28日 (火) 16時57分44秒	2020年01月28日 (火) 16時57分19秒	yum list installed | grep tomcat	150.89.223.103	tomcat.noarch 7.0.76-9.el7_6 @base tomcat-el-2.2-api.noarch 7.0.76-9.el7_6 @base tomcat-jsp-2.2-api.noarch	N/A	詳細
5	2020年01月28日 (火) 16時57分44秒	2020年01月28日 (火) 16時57分20秒	yum list installed | grep tomcat	150.89.223.104	tomcat.noarch 7.0.76-9.el7_6 @base tomcat-el-2.2-api.noarch 7.0.76-9.el7_6 @base	N/A	詳細

標準エラー出力: N/A

標準出力: tomcat.noarch 7.0.76-9.el7_6 @base tomcat-el-2.2-api.noarch 7.0.76-9.el7_6 @base tomcat-jsp-2.2-api.noarch 7.0.76-9.el7_6 @base tomcat-lib.noarch 7.0.76-9.el7_6 @base tomcat-servlet-3.0-api.noarch 7.0.76-9.el7_6 @base

標準エラー出力: N/A

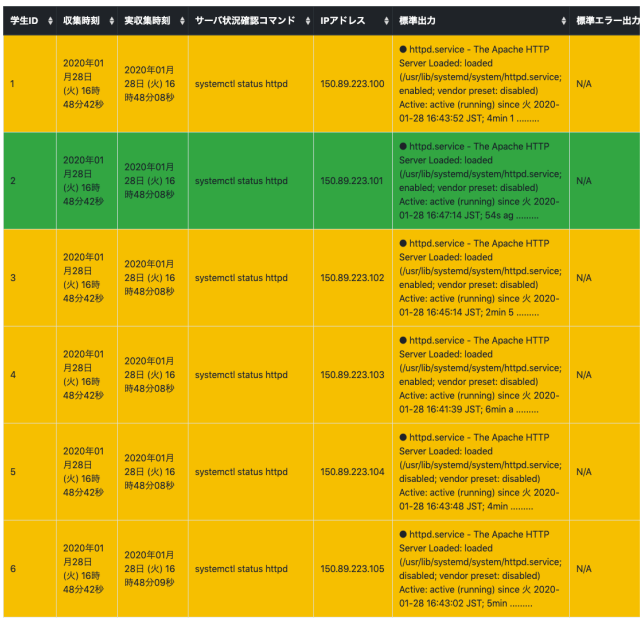
標準出力: tomcat.noarch 7.0.76-9.el7_6 @base tomcat-el-2.2-api.noarch 7.0.76-9.el7_6 @base tomcat-jsp-2.2-api.noarch 7.0.76-9.el7_6 @base tomcat-lib.noarch 7.0.76-9.el7_6 @base tomcat-servlet-3.0-api.noarch 7.0.76-9.el7_6 @base tomcat-webapps.noarch 7.0.76-9.el7_6 @base

図 5　正常な場合のサーバ状況確認履歴表示画面

コマンドを用いて，各学生の演習用サーバの Apache が起動しているか確認するコマンドが実行されており，ログの内容をよく見ると緑色のカテゴリと黄色のカテゴリの内容はどちらとも Apache が active(running) と表示されており，学生の Apache は正常に起動出来ているにも関わらず，クラスタリングが正しく行えていない事がわかる．このような結果になってしまう原因としては，ログの詳細情報を確認した所，systemctl などでソフトの status を確認すると，Apache が動作しているかの情報だけでなく，どのプロセスが何時に立ち上がっているなどの詳細なログ情報が付加された形で結果を表示する為，ログ情報の文字列自体の長さが各学生毎に異なってしまいクラスタリングが正しく行われないという事がわかった．現状では，文字列の数が同一で文字列中に含まれている数値が異なるような標準出力の結果には対応しているが，文字列自体の長さが異なる状態での正確なクラスリングは行えていない．よって，ソフトウェアの動作を確認する為のサーバ状況確認コマンドなどには，grep などを用いて特定の文字列が含まれる結果のみを抽出したログ情報の収集が必要になる．

6. 考察

今回のサーバ状態確認履歴を用いた正誤判定支援システムの結果から，ソフトウェアが正しくインストールされているのかを確認するといった標準出力の内容が限られているケースにおいては正常に正誤判定が行えることが確認できた．一方で，systemctl status httpd といったコマンドの出力のような本来評価したい出力以外に詳細なログデータなどが出力されることにより，文字列の長さが異なってしまう事で正確なクラスリングが行えないという問題点が見つかった．



学生ID	収集時刻	実収集時刻	サーバ状況確認コマンド	IPアドレス	標準出力	標準エラー出力
1	2020年01月28日 (火) 16時48分42秒	2020年01月28日 (火) 16時48分08秒	systemctl status httpd	150.89.223.100	● httpd.service - The Apache HTTP Server Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled) Active: active (running) since 火 2020-01-28 16:43:52 JST; 4min 1	N/A
2	2020年01月28日 (火) 16時48分42秒	2020年01月28日 (火) 16時48分08秒	systemctl status httpd	150.89.223.101	● httpd.service - The Apache HTTP Server Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled) Active: active (running) since 火 2020-01-28 16:47:14 JST; 54s ag	N/A
3	2020年01月28日 (火) 16時48分42秒	2020年01月28日 (火) 16時48分08秒	systemctl status httpd	150.89.223.102	● httpd.service - The Apache HTTP Server Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled) Active: active (running) since 火 2020-01-28 16:45:14 JST; 2min 5	N/A
4	2020年01月28日 (火) 16時48分42秒	2020年01月28日 (火) 16時48分08秒	systemctl status httpd	150.89.223.103	● httpd.service - The Apache HTTP Server Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled) Active: active (running) since 火 2020-01-28 16:41:39 JST; 6min a	N/A
5	2020年01月28日 (火) 16時48分42秒	2020年01月28日 (火) 16時48分08秒	systemctl status httpd	150.89.223.104	● httpd.service - The Apache HTTP Server Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled) Active: active (running) since 火 2020-01-28 16:43:48 JST; 4min	N/A
6	2020年01月28日 (火) 16時48分42秒	2020年01月28日 (火) 16時48分09秒	systemctl status httpd	150.89.223.105	● httpd.service - The Apache HTTP Server Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled) Active: active (running) since 火 2020-01-28 16:43:02 JST; 5min	N/A

図 6　異常な動作をしたサーバ状況確認履歴表示画面

また，プロセス ID や時刻情報の数値を対象として文字列の正規化をすることは可能であるが，事前に正規化対象の文字列をすべて洗い出すことは現実的ではない．そのため，grep などで標準出力をさらにフィルタして，余計な情報が入り込まない様なコマンドを考えてログ収集を行うか，クラスタリングを工夫して，実行ごとに異なる文字列は無視するといったアルゴリズムが考えられないか今後検討していきたい．

7. おわりに

本稿ではサーバ状態確認履歴を用いた正誤判定支援システムの提案を行い，それらの環境を実際にサーバ管理演習におけるログ情報を用いて可視化を行い，ケーススタディとして示した．今回の可視化環境により，正誤判定支援の精度を高める為には，学生の進捗状況に関わる必要最低限の情報のみが抽出できるようなサーバ状況確認コマンドの選択が重要になってくることが分かった．

謝辞　本研究の一部は JSPS 科研費 17K00500 の助成を受けた．

参考文献

[1] Christensen, H. B.: Teaching DevOps and cloud computing using a cognitive apprenticeship and story-telling approach, *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education*, pp. 174–179 (2016).

[2] Hollingsworth, J. and Powell, D. J.: Teaching web programming using the Google Cloud, *Proceedings of the 48th Annual Southeast Regional Conference*, pp. 1–5 (2010).
[3] Loukides, M.: *What is DevOps?*, " O'Reilly Media, Inc." (2012).
[4] Mell, P., Grance, T. et al.: The NIST definition of cloud computing (2011).
[5] Ortiz, A.: Architecting Serverless Microservices on the Cloud with AWS, *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pp. 1240–1240 (2019).
[6] Yokoyama, S., Yoshioka, N. and Shida, T.: Cloud in a cloud for cloud education, *2012 4th International Workshop on Principles of Engineering Service-Oriented Systems (PESOS)*, IEEE, pp. 63–64 (2012).
[7] Yokoyama, S., Yoshioka, N. and Shida, T.: Edubase Cloud: Cloud platform for cloud education, *2012 First International Workshop on Software Engineering Education Based on Real-World Experiences (EduRex)*, IEEE, pp. 17–20 (2012).
[8] 鎌田元樹，桝田秀夫
：OpenStack を利用したサーバ設定演習システムの提案，研究報告インターネットと運用技術 (IOT)，Vol. 2013, No. 3, pp. 1–6 (2013).
[9] 佐伯幸郎，まつ本真佑，井垣　宏，福安直樹，水谷泰治，中村匡秀
：ソフトウェア開発 PBL における AWS in Education 助成プログラムの活用，日本ソフトウェア科学会大会論文集， No. 32, p. 5 (2015).
[10] 西村 一輝，井垣宏
：サーバ管理演習におけるサーバ状況確認履歴可視化手法の検討，信学技報，Vol. 118, No. 293 (2018).
[11] 西村 一輝，井垣宏
：サーバ管理演習のためのユーザー行動履歴収集システムの検討，信学技報，Vol. 117, No. 381, pp. 155–160 (2018).
[12] 西村 一輝，井垣宏
：サーバ管理演習における誤り原因特定のためのコマンド実行履歴とファイル編集履歴の可視化手法の検討，研究報告コンピュータと教育（CE），Vol. 2019-CE-148, No. 5, pp. 1–7 (2019).
[13] 西村 一輝，井垣宏
：サーバ管理演習によるサーバ状況確認履歴を用いた自動正誤推定システムの検討，ソフトウェアエンジニアリングシンポジウム 2019 (2019).
[14] 西村 一輝，井垣 宏，尾花将輝
：進捗状況の把握と振り返り支援を目的としたサーバ管理演習のためのユーザ行動履歴収集システムの検討，ソフトウェアエンジニアリングシンポジウム 2018 論文集，Vol. 2018, pp. 253–254 (2018).
[15] 中崎満晶，越智　徹，中西通雄
：OpenStack を用いた Web サーバ設定演習環境の構築，信学技報，Vol. 115, No. 482, pp. 49–54 (2016).