Jupyter Notebookを利用したMoodle運用

群馬大学 総合情報メディアセンター
浜元　信州

Moodleについて

- 多くの大学で利用されているLMS
- 学外公開されていることも多い

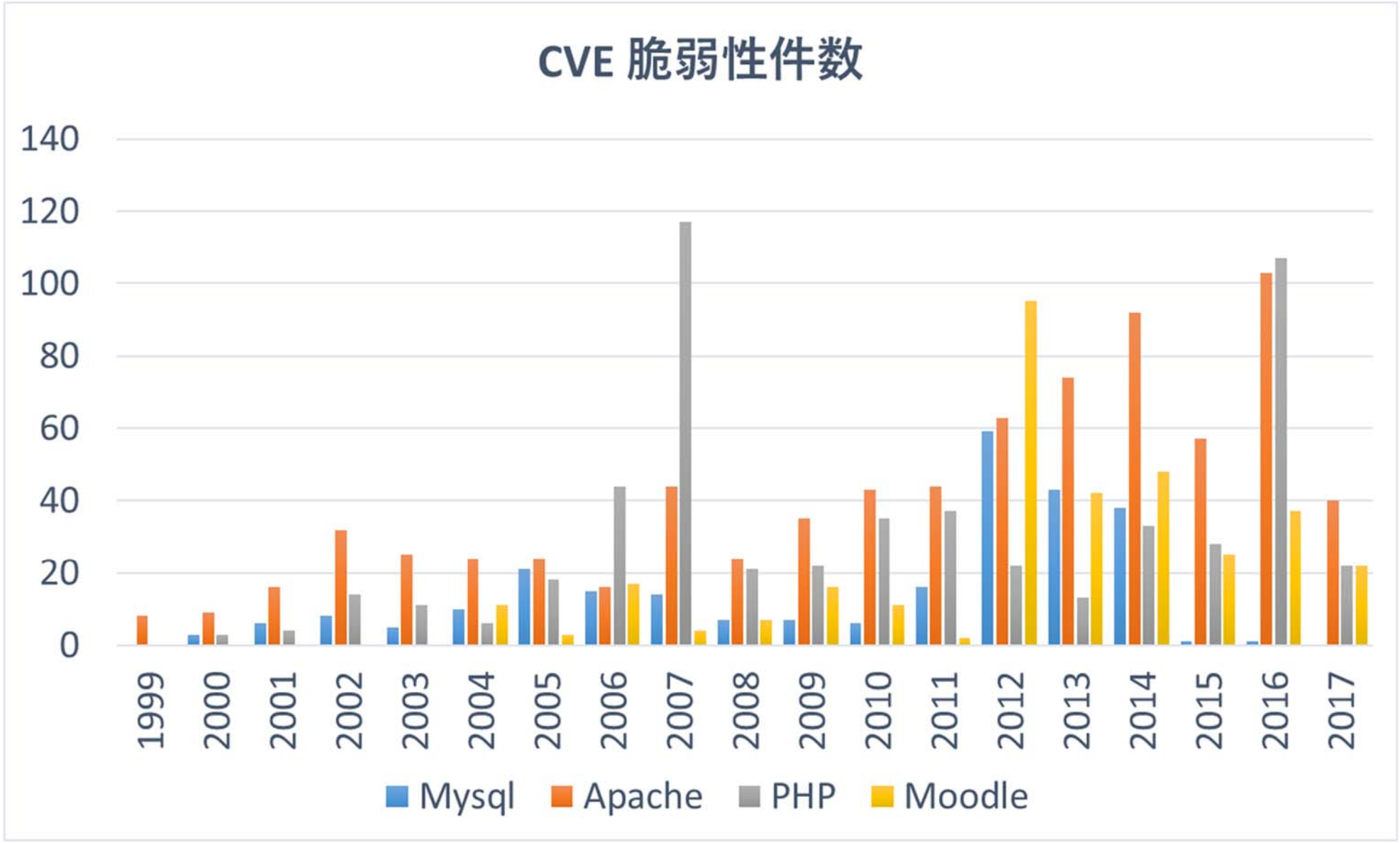
- 運用上の課題
  - コース作成
  - プラグインの導入可否
  - ハードウエアの老朽化
  - 脆弱性対応
    - ソフトウエア（LAMP環境）のアップデート
    - Moodle自身のアップデート

Moodleの脆弱性

http://www.cvedetails.com/vulnerability-list/vendor_id-2105/product_id-3590/Moodle-Moodle.html

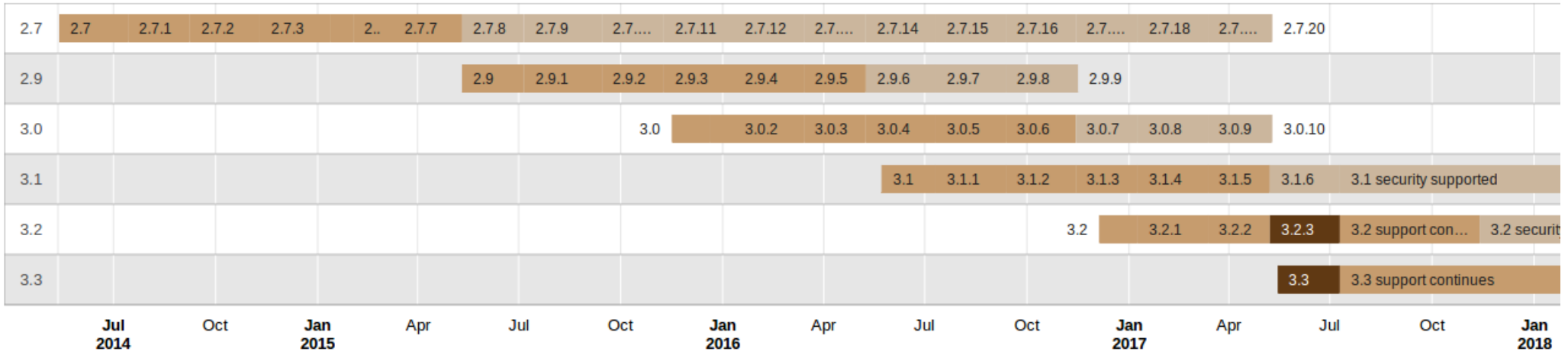
http://www.cvedetails.com/

脆弱性件数の推移



Moodleのリリーススケジュール

- Moodle2.6以降はリリーススケジュールが固定化
- マイナーバージョンは6カ月ごとにリリース
  （5月と11月第2週月曜）
- ほぼ毎月のペースでアップデート
- サポート期間は通常18カ月，LTSは36カ月
  - BugFix 12カ月＋Security Update 6カ月or24カ月



運用上の課題

- コース作成　→　今回は扱わない
- プラグインの導入可否　→　今回は扱わない

- ハードウエアの老朽化　→　IaaS型のクラウド利用

- 脆弱性対応
  - アップデートにより解決するか，授業に影響がないようにしたい
  - 長期間の停止を伴うアップデートは難しい

→　迅速にアップデートする手順の確立

目標

1. クラウド上でのMoodle立ち上げを（わかりやすい範囲で）自動化する。

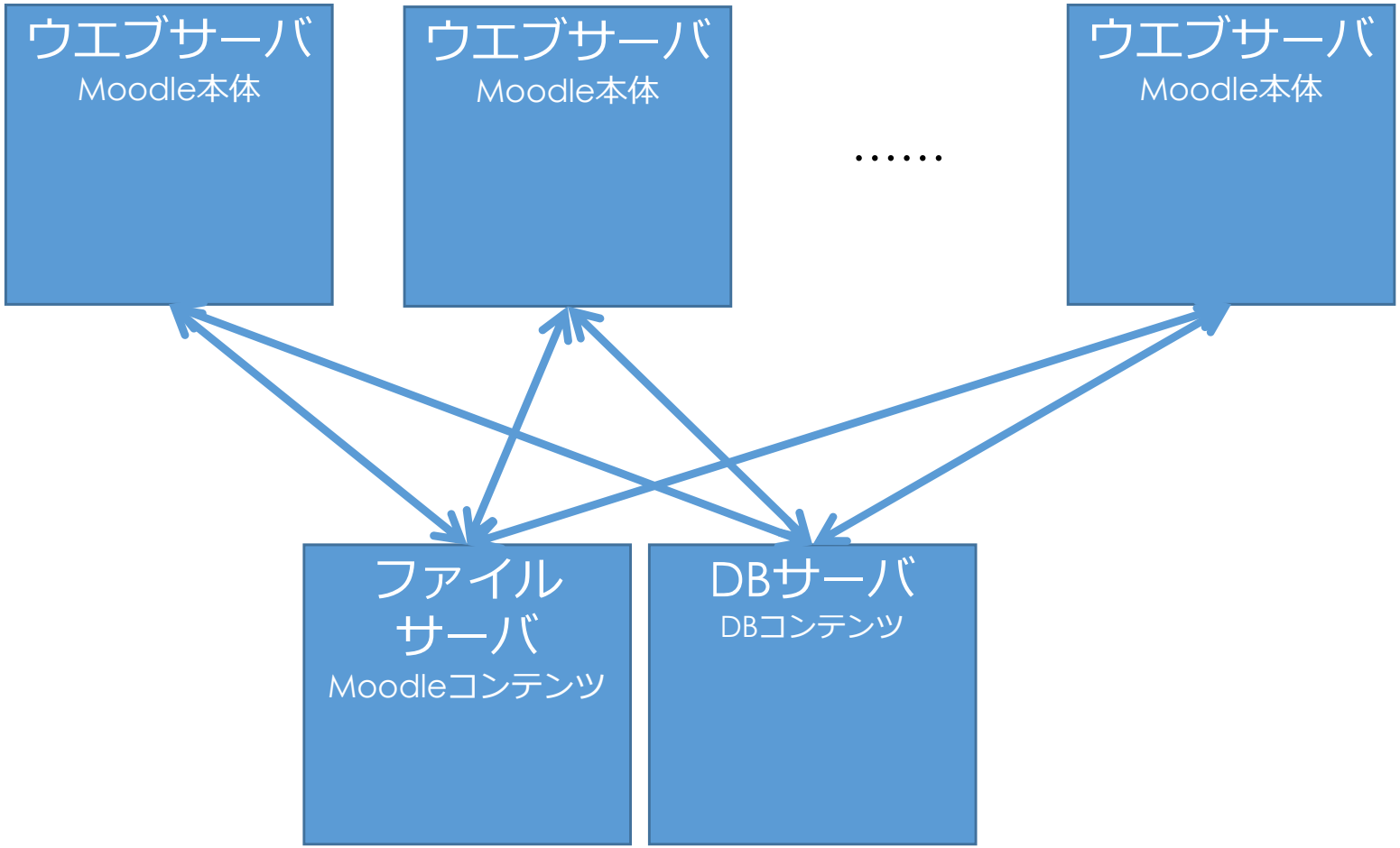
2. クラウド上でのMoodleのバージョンアップを（わかりやすい範囲で）自動化する。
    1. 本番環境と同内容の検証環境作成
    2. 検証環境のバージョンアップ
    3. 本番環境のバージョンアップ

3. クラウド上でのMoodleのバックアップを（わかりやすい範囲で）自動化する。

想定する構成(1)
小規模サービス用：今回想定する構成

ウエブ，DB，コンテンツが全て１台で完結

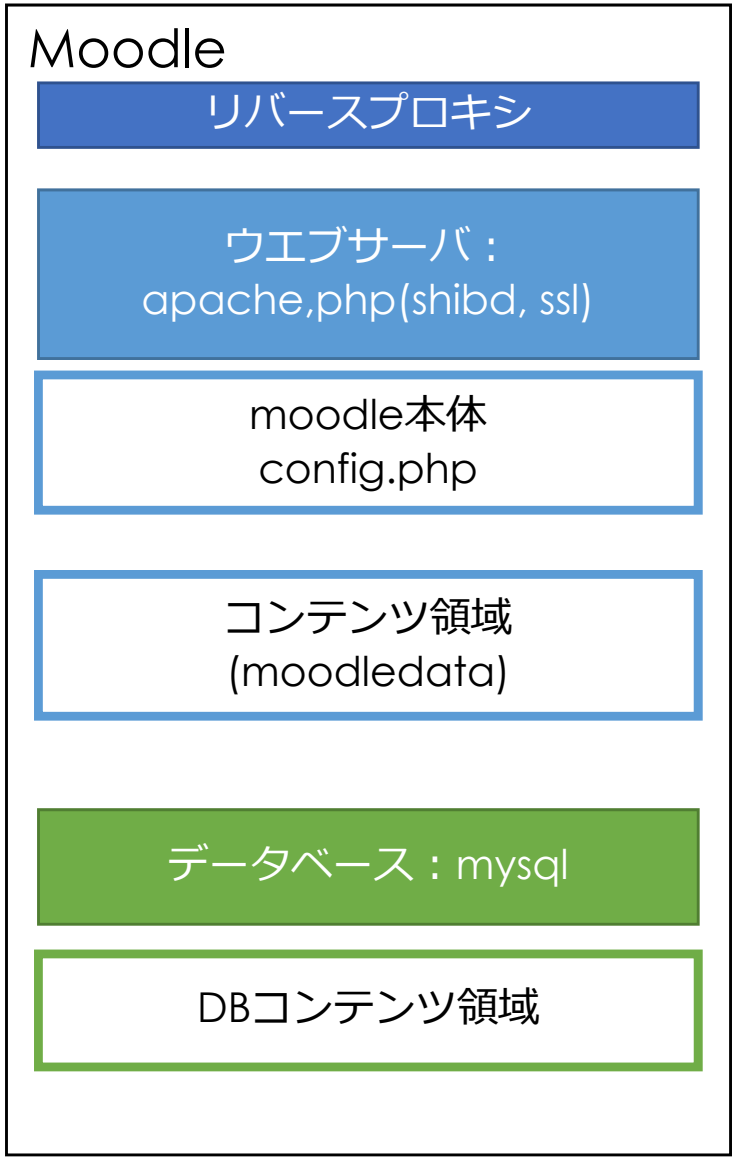
想定する構成(2)
大規模サービス用



＋検証環境，バックアップ

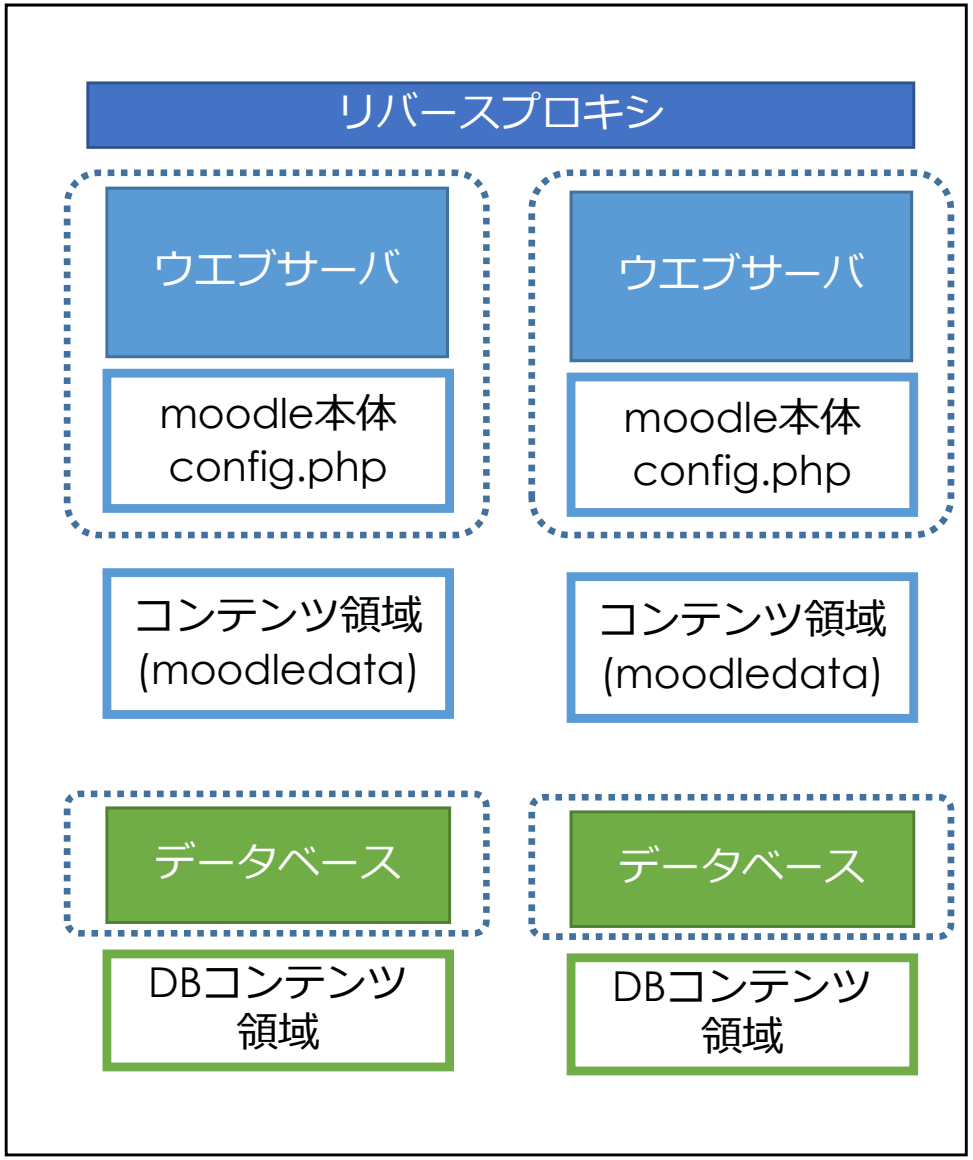
小規模構成でのアップデートの流れ

1. 検証環境の作成
    1. データコピー
        1. Moodleコンテンツ（moodledata）
        2. DBコンテンツ
    2. サーバアップデート
        1. ウエブサーバ
        2. DBサーバ
    3. moodleアップデート
        1. moodle本体ダウンロード
            1. DBアップデート・設定アップデート
        2. プラグインアップデート
            1. DBアップデート・設定アップデート
2. 本番環境の作成
    1. 同じことを繰り返す
3. 切り戻し



小規模構成でのアップデートの流れ

1. 検証環境の作成
    1. データコピー
        →copy on writeによる高速化
    2. サーバアップデート
        →Dockerコンテナを採用
        →イメージを作成し再利用する
    3. moodleアップデート
        1. moodle本体ダウンロード
            1. DBアップデート・設定アップデート
        2. プラグインアップデート
            1. DBアップデート・設定アップデート
2. 本番環境の作成
    1. 同じことを繰り返す
3. 切り戻し
    →リバースプロキシで切り戻す
    （Blue-Green Deplyment）



実演とまとめ

- 小規模構成を想定して，Jupyter Notebookを利用したMoodleのアップデート手順を作成した。
- ダウンタイムが極力短くなるよう，copy on writeによる高速化，Dockerによるイメージ化を行った。
- Dockerコンテナの利用により，現実に近い事前検証環境を同一サーバ内準備することができた。
- アップデートと切り戻しを確実に行うため，リバースプロキシによる切り替えを行う構成とした。