基幹システム開発論

version 0.1

□□ □□

2012 年 03 月 23 日

目次

基幹システム開発特論

この資料について

※この資料は [GitHub Page] にあります．

※この資料に関する質問や意見，改善提案などは [GitHub Issues] にご記入ください．

目次

概要

授業の目標

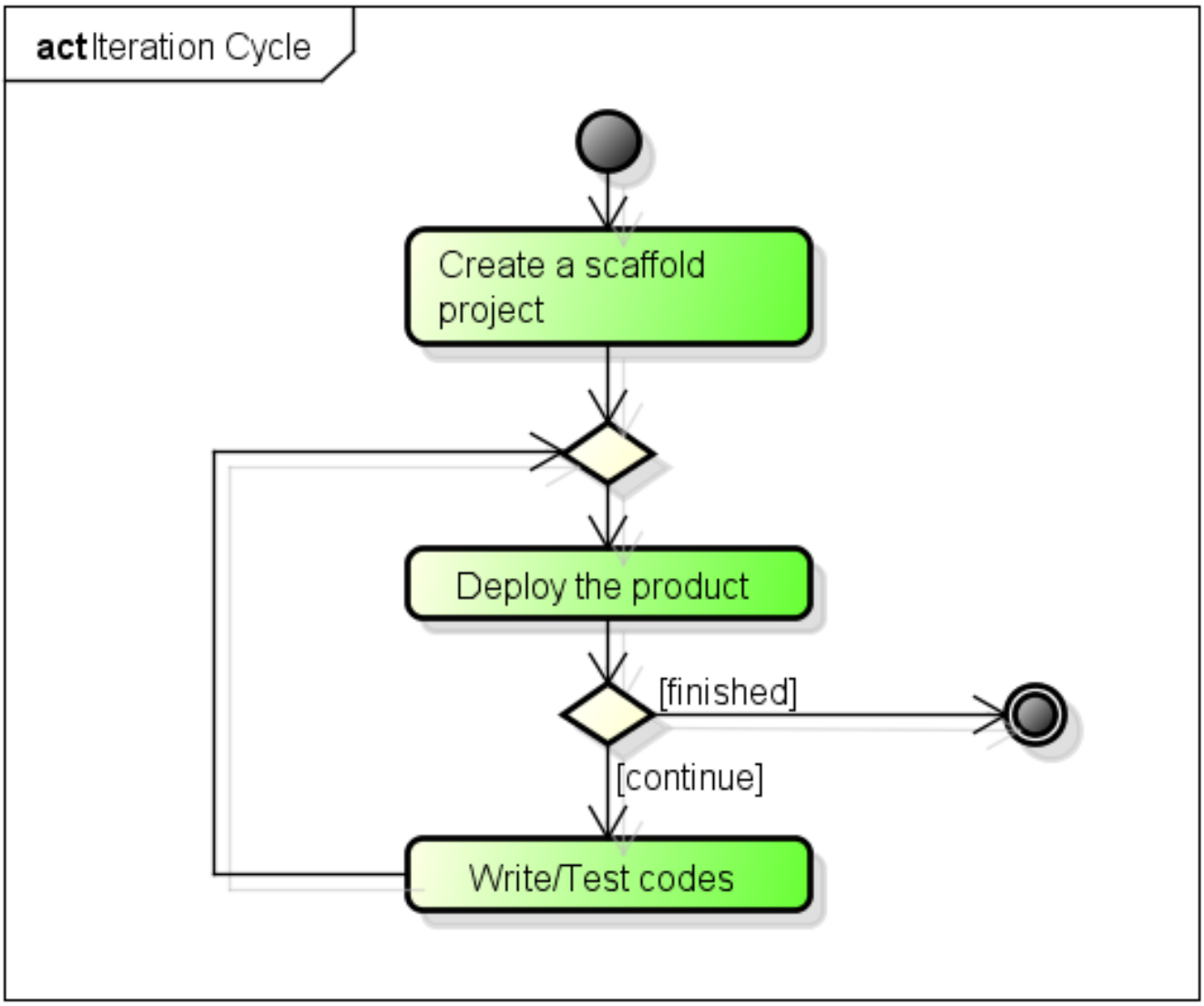
シラバスにある目標に対して，次の内容を追加します．

- 少人数のチームで，イテレーションサイクルを回すことができるようになる．

演習の進め方

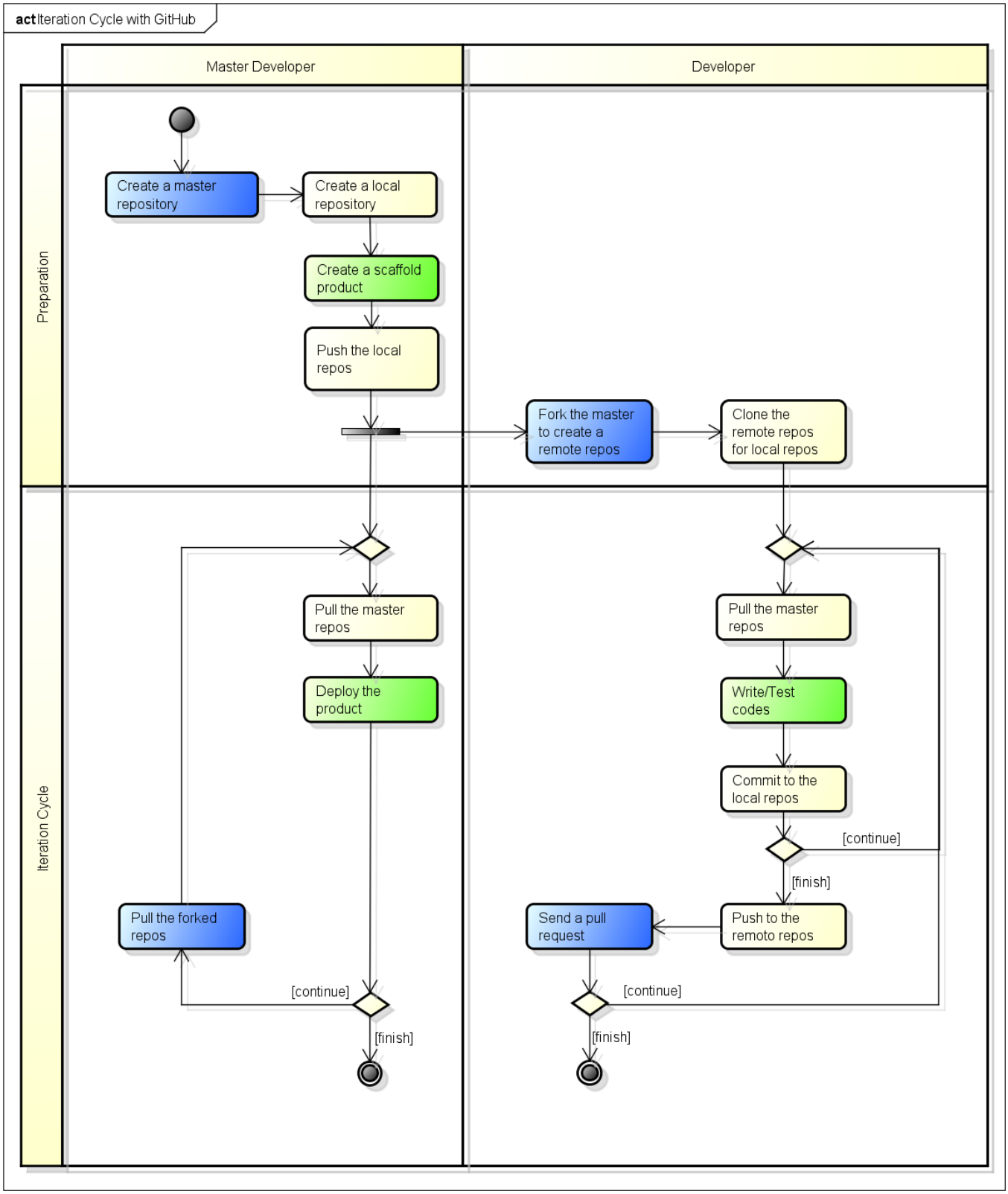
Stage 1: イテレーションを一人でまわす

はじめに，自分のマシンに開発環境を構築し，一人でイテレーションを回してみます．イテレーションサイクルは次の図に示す通りです．



Stage 2: イテレーションをチームでまわす

次に，サーバを構築し，チームでイテレーションを回してみます．ここでは，GitHubを使った共同開発をします．その概要を次に示します．



なお，チーム編成は，Stage 1を終わった順にアサインしていきます．

Stage 3: 基幹システムを開発する

最後に，冗長性や可用性，拡張性などを備えたシステムをチームで開発します．

- DBサーバのクラスタリング
- Webアプリケーションサーバのクラスタリング
- ロードバランサの導入

・仮想化

授業の日程

1. 12/08
2. 12/15
3. 12/22
4. 01/12
5. 01/19
6. 01/26
7. 02/02
8. 02/09

スケジュール

第1回の授業は，個人が作業するためのWorkstation環境を構築し，Stage 1を開始します．Stage 1が終わったら，担当者の確認を受けてください．

第2回の授業から，Stage 1が終わった人順にチームを編成し，Stage 2に移ります．グループの中でイテレーションサイクルを回してみましょう．作成するアプリケーションについては，担当者が指示をします．

第3回の授業以降，Stage 3で何をするかをグループで相談し，実際の作業に移ります．なお，Stage 2と3の間で，グループを再編成する可能性があります．

第4回以降の計画は，各グループで立案してください．

第8回では，グループで行った作業内容について報告してもらいます．

利用する技術

利用する技術についての方針

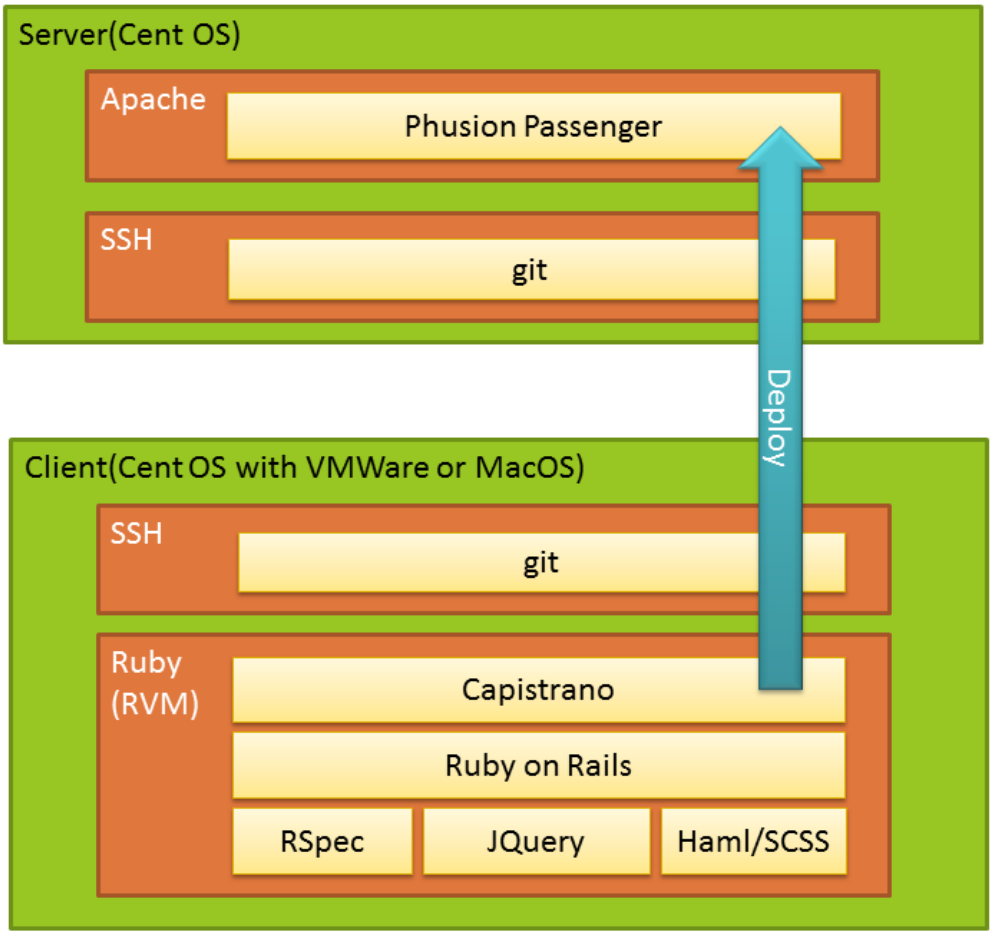
Stage 1,2では，利用する技術を指定します．ここでは，Ruby on Railsを利用します．全員が同じ環境で，作業をすることになります．

作成するプログラムは簡単なものですので，Rubyになれているかどうかの違いはあまり気にならないはずです．目標はあくまでもイテレーションサイクルを回してみることです．

Stage 3まで進んだら，チーム内でお好きな技術を使って開発してみてください．

なお，Stage 2とStage 3との間で，チームを組み替える可能性があります．

Ruby on Railsを用いた開発



- Phusion Passenger - RailsアプリケーションをApacheに接続して動作させるためのモジュール
- Capistrano - SSH/gitを使って，RailsアプリケーションをPassengerに対してデプロイするためのツール

ワークステーションを準備する

ワークステーションとは

- プログラマが開発に使用するコンピュータ
- ソースコードのコーディングを行う
- サースコードの単体テスト（Unit Test）を行う
- ローカルのテスト用サーバを使ってテストを行う

全体構成

- OS
    - Windows（またはMacOS）
- 仮想化
    - Windowsの場合， VMWare Player を用いて CentOS 6.0 をインストールする
- IDE(統合開発環境)
    - GNU Emacs
- Ruby on Rails

- Ruby 1.9.3
- Rails 3.1
- Capistrano
- Passenger

仮想OSのインストール

VMware Playerのインストール

- VMware PlayerのDownloadページ に行き，アカウントを作成して，ダウンロードして，インストールする

CentOS 6.0のインストール

- CentOS 6.0 DVDのISOイメージをダウンロードする
  - 32ビット版
    - http://ftp.riken.jp/Linux/centos/6.0/isos/i386/CentOS-6.0-i386-bin-DVD.iso
  - 64ビット版
    - http://ftp.riken.jp/Linux/centos/6.0/isos/x86_64/CentOS-6.0-x86_64-bin-DVD1.iso
    - http://ftp.riken.jp/Linux/centos/6.0/isos/x86_64/CentOS-6.0-x86_64-bin-DVD2.iso
- VMware Playerを起動して，ISOイメージからインストールを行う.

アカウントの作成と設定

- 自分のルーザアカウントでログインする
- rootアカウントに切り替える

```
$ su -
```

- OSをアップグレードする

```
# yum upgrade
```

- 自分のユーザアカウント(例：username）をwheelグループに追加する

```
# usermod -G wheel username
# groups username    # <- check
```

- wheelグループに対して，sudoでのコマンド実行を許可する

```
# visudo
```

変更する箇所:

```
## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)    ALL
```

- exitする

```
# exit
# whoami    # <- check
```

- 設定をが上手くいったか確認する

```
$ cat /etc/shadow    # <- check this will fail
$ sudo cat /etc/shadow  # <- check this will success
```

日本語環境のインストール

- ユーザアカウントでログインする
- 日本語パッケージをインストールする

```
$ sudo yum -y groupinstall "Japanese Support"
```

参考: http://www.server-world.info/query?os=CentOS_6&p=japanese

SSHのための公開鍵の作成

SSH接続で使う公開鍵/非公開鍵のペアを作成します．演習で利用する重要なファイルですので，USBメモリなどにバックアップをとっておくことをおすすめします．

- ユーザアカウントでログインする
- 公開鍵を作成する（演習用なので，パスフレーズは入れなくてもかまわない）

```
$ ssh-keygen
$ ls ~/.ssh   # <- check
```

- 公開鍵をauthorized_keysに登録する

```
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
$ chmod 600 ~/.ssh/authorized_keys
```

- 公開鍵で（＝パスワード無しで）ログインできるか確かめる

```
$ ssh localhost   # <- you don't need to type your password
$ exit
```

Git

- gitのインストール

```
$ sudo yum -y install git
```

GitHub

- GitHub にアカウントを作成する
  - アカウント名とメールアドレスは大学のものと同じにしてください
  - 登録ができたら，管理画面でSSHの公開鍵を登録してください [1] .

railsユーザアカウントを作成する

Railsのアプリケーションをデプロイする場所として，railsアカウントを作成してそのhomeを利用する．このhomeは，6770の設定をし，railsグループに所属するユーザからの参照・更新を許可する．

- railsユーザアカウントの設定

```
$ sudo /usr/sbin/useradd rails
$ sudo chmod 6770 /home/rails/
```

- railsグループに，自分のユーザアカウントとapacheアカウントを追加する

```
$ sudo /usr/sbin/vigr
```

Ruby on Rails

- Ruby 1.8.7のインストール（開発用パッケージも含む）

```
$ sudo yum -y install ruby ruby-devel
$ ruby -v  # <- check
```

・Gem 1.8.7のインストール

```
$ sudo yum -y install rubygems
$ gem -v  # <- check
```

・Rails 3.1 のインストール

```
$ sudo gem installl rails
$ rails -v  # <- check
```

・Capistrano のインストール

```
$ sudo gem install capistrano
```

・Phusion Passenger のインストール

```
$ sudo gem install passenger
```

・Passengerをapacheと接続するためのモジュールをコンパイルする

```
$ sudo yum -y install gcc-c++ curl-devel openssl-devel zlib-devel httpd-devel apr-devel apr-util-devel
$ sudo passenger-install-apache2-module
```

以下の内容を/etc/httpd/conf.d/rails.confに記述する

```
LoadModule passenger_module /usr/lib/ruby/gems/1.8/gems/passenger-3.0.11/ext/apache2/mod_passenger.so
PassengerRoot /usr/lib/ruby/gems/1.8/gems/passenger-3.0.11
PassengerRuby /usr/bin/ruby

<VirtualHost *:80>
  ServerName localhost
  DocumentRoot /home/rails/myapp/current/public
  <Directory /home/rails/myapp/current/public>
    AllowOverride all
    Options -MultiViews
  </Directory>
</VirtualHost>
```

・httpdを再起動する

```
$ sudo service httpd restart
```

追記(12/8)

gitの設定方法

次のコマンドで，gitの設定をしてください．

```
$ git config --global user.name "Your Name"
$ git config --global user.email "username@domain.example"
```

gitの利用で困ったら？

次のコマンドで，状況が確認できます．

```
$ git status
```

次のコマンドで，ログが確認できます．

```
$ git log
```

なお，gitのエラーメッセージは英語ですが，わりと丁寧に記述されているので，しっかりと読むようにしましょう．

RVM を使って，コンパイルします．

追記（12/10）

Macにrubyをインストールする

・GUIツール

http://unfiniti.com/software/mac/jewelrybox

・コマンドラインからインストールする場合

```
$ curl -O -s https://raw.github.com/wayneeseguin/rvm/master/binscripts/rvm-installer
$ sudo . rvm-installer
```

Rubyをインストールします

```
$ rvm install 1.8.7
$ ruby -v # <- check
$ rvm use 1.8.7 --default
```

追記（12/11）

EmacsでGit

Magitがよいようです．

http://philjackson.github.com/magit/

追記（12/15）

CentOSインストール時の注意

ないこと（とりあえず，間違えなく入力できる文字でパスワードを指定しておき，インストール後にログインできることが確認できてか
entOSのインストール時に，SELinuxは選択しないこと．SELinuxを有効にしてしまうと，httpdを起動するときにWarningがでてしまい

課題

SELinux時のエラーをとりあえず回避する方法については， 選択しないこと．SELinuxを有効にしてしまうと，ht
tpdを起動するときにWarningがでてしまいます．

※ SELinux時のエラーをとりあえず回避する方法については， http://kennyqi.com/archives/43.html を参照

ToDo

課題

コラボレーティブソフトウェアデベロップメント (CSD)

課題

図の修正：deployはDeveloper Aが行う

1 http://help.github.com/win-set-up-git/ の4節参照

(元のエントリ は、 /home/yc/git/enterprise_system_development/source/iteration_team.rst の 51 行目です)

課題

SELinux時のエラーをとりあえず回避する方法については， 選択しないこと．SELinuxを有効にしてしまうと，httpdを起動するときにWarningがでてしまいます．

(元のエントリ は、 /home/yc/git/enterprise_system_development/source/workstation.rst の 329 行目です)

課題

コラボレーティブソフトウェアデベロップメント (CSD)

(元のエントリ は、 /home/yc/git/enterprise_system_development/source/index.rst の 35 行目です)