Regional SNS
クラウド環境への導入ガイド

2012 年 2 月 29 日

株式会社ネットワーク応用通信研究所

目次

1. はじめに

本文書では、クラウド環境（MOGOK）に RegionalSNS を導入する手順を説明します。

2. 導入環境及び前提条件

導入先のクラウド環境は株式会社インターネットイニシアティブ（IIJ）社が開発を進める PaaS 環境「MOGOK」（http://mogok.jp/）を対象とします。

導入に使用する作業環境は、MOGOK サービスが提供する Windows 用クライアントパッケージを導入した Windows PC（32bit 環境の Windows Xp または Windows 7）か、Linux 環境を前提とします。

また、MOGOK サービスへのアカウントは事前に準備済みであるものと前提します。
MOGOK サービスの利用申し込みは、http://mogok.jp/ をご参照ください。

3. 導入準備

作業用 PC において、以下のいずれかの準備が整っているものと前提します。

3.1. Windows 環境の場合

MOGOK のポータルサイト（https://portal.mogok.jp/）にログインし、「ダウンロード」画面から「mogok_client_packages_setup.exe」をダウンロードし、インストールします。
インストール先は任意ですが、本文書では「c:¥mogok」に環境がインストールされたものとして説明します。
以降の説明において、コマンドラインの操作は、「c:¥mogok¥msysgit¥msys.bat」から起動したシェル環境を利用するものとします。

3.2. Linux 環境の場合

Linux 環境では、以下の各ソフトウェアをインストールしてください。

- Ruby 1.9.2 以降
  - Mogok コマンドの実行環境として
  - ruby インタプリタを環境変数 PATH に通してあることを前提する
- Git
  - Mogok コマンドが依存するバージョン管理ソフトウェア

3.3. MOGOK 用 Ruby Gems パッケージの導入／更新

前述のダウンロード画面から、「mogok-x.x.x.gem」（x.x.x 部分はバージョン番号）の Gem パッケージをダウンロードし、Ruby 実行環境にインストールします。
※ バージョンは最新版を使用してください。

$ gem install mogok-x.x.x.gem

3.4. mogok 環境へのログイン

任意の mogok コマンドを実行し、MOGOK サービスへログインを行います。

例）
$ mogok key
login: ○○○
password: △△△△△

3.5. 鍵ペアの作成・公開鍵の登録

MOGOK サービスで利用する公開鍵／秘密鍵のペアを作成します。
以下のコマンドを実行し、HOME/.ssh/id_rsa、id_rsa.pub のペアを作成します。

$ ssh-keygen –t rsa

生成後、以下のコマンドで公開鍵を MOGOK サービスに登録します。

$ mogok key add ~/.ssh/id_rsa.pub

登録した公開鍵の一覧は「mogok key」コマンドで確認可能です。

4. ソースコードの準備

次に、デプロイする RegionalSNS のソースコードを作業用 PC に配置します。
ソースコードは、ホームディレクトリの「regionalsns」ディレクトリに配置されたものと前提します。
ソースコードの入手は、RegionalSNS の Subversion リポジトリから「svn export」コマンドで抽出するなどして用意します。
RegionalSNS の Subversion リポジトリは、以下の URL になります。

```
http://regionalsns.googlecode.com/svn/trunk/
```

5. デプロイ

準備したソースコードを用いて、以下の手順に沿って MOGOK へデプロイし、稼働状態まで作業します。

5.1. Git ローカルリポジトリの作成

ホームディレクトリの regionalsns ディレクトリで、以下の一連のコマンドを実行します。

```
$ git init
$ git add .
$ git commit –m “first commit”
```

上記の作業で、ローカル環境に git リポジトリが生成されます。

5.2. MOGOK アプリケーションの登録

次に、MOGOK サービス上に RegionalSNS 用のアプリケーション情報を登録します。
以下のコマンドを実行します。

```
$ mogok create regionalsns
```

エラーメッセージ無く、「Done」のログが出れば正常です。
また、MOGOK のポータルサイトにログインし、「アプリケーション」画面を参照しても生成結果は確認できます。

なお、アプリケーション名（regionalsns の部分）は、他のユーザーが先に登録した場合は重複登録はできません。
名称は任意ですので、その場合はアプリケーション名を変更してください。

mogok コマンドで確認する場合は、以下のコマンドを実行します。

$ mogok list

5.3. ソースコードのアップロード

次に、regionalsns ディレクトリで以下のコマンドを実行し、ソースコードを MOGOK サービス側の Git リポジトリに push（アップロード）します。

$ git push mogok master

5.4. MOGOK サービス上での配置実行

以下のコマンドを実行し、MOGOK サービス上にアプリケーションサーバを立ち上げ、RegionalSNS を稼動状態に遷移させます。

$ mogok deploy

全てのデプロイプロセスが正常終了すると、「Done」のログが表示されます。
※ 失敗した場合は「Deploy Error」のログが出力されます。

5.5. DB のマイグレーション並びに初期データ投入

次に、以下のコマンドを連続して実行し、データベースの準備を行います。

$ mogok rake db:migrate
$ mogok rake db:seed

最後に、以下のコマンドでアプリケーションサーバを再起動します。

$ mogok restart

6. 動作確認

ここまでの作業が完了した後、以下のコマンドを実行し、結果を確認します。

$ mogok info

AppName: regionalsns
RubyVersion: 1.9.2
Status: on
DeployStatus: Done
MaintenanceStatus: off
AppInstancesCount: 1
Repository: git@git.mogok.jp:regionalsns.git
URL: http://regionalsns.ruby.iijgio.com/

ここで示される URL にブラウザでアクセスし、以下の初期管理者アカウントでログインをします。

ユーザー名： quentin
パスワード： monkey

ログイン後、速やかに画面右上の「設定」リンクから「パスワード変更」を選択し、管理者アカウントのパスワードを変更してください。

7. MOGOK β版での制約条件

MOGOKβ版では、以下の制約があります。
2012 年 2 月現在、クローズドβ版の状況にあるため、RegionalSNS の一部機能が利用できません。
利用できない（機能を OFF にして導入する）機能は以下の通りです。

- メール通知機能
  - MOGOK アプリケーションから外部の MTA（メールサーバ）に接続することがβ版ではできないため。

8. RegionalSNS 以外のアプリケーションをデプロイする際の注意点

MOGOK 環境（β版）にアプリケーションをデプロイする際に、アプリケーション側で注意すべきポイントについて説明します。

8.1. 一般的な注意事項

MOGOK の PaaS 環境においてアプリケーションをデプロイする際は、以下に注意ください。

8.1.1. 外部の依存ソフトウェアの有無とバージョン

Ruby のライブラリの中には、OS にインストールされている特定のソフトウェアの存在を前提としているものがあります（RMagick における ImageMagick など）。

このようなライブラリをアプリケーションで使用している場合、PaaS のアプリケーション実行環境に当該ソフトウェアが入っているのか、事前に PaaS 側のドキュメント等で確認する必要があります。

また、この際にはソフトウェアのバージョンなどにも注意すべきです（特定のバージョン以降でなければ動作しないライブラリもあり得ます）。

もし、当該 PaaS 側で任意の OS 側ソフトウェアを追加することは一般的に難しい場合が多いため、アプリケーション側でそれを回避する修正が必要になる場合もあります。

8.1.2. アプリケーション実行環境のファイルシステム利用

MOGOK に代表される PaaS のアプリケーション実行環境では、デプロイする都度、アプリケーション実行環境のソースコードが一新される場合が多く見られます。

これは、例えば Rails アプリケーションで、ユーザーからアップロードされたファイルを、RAILS_ROOT 配下の任意のディレクトリに保存するような構造のアプリケーションの場合、デプロイした瞬間、蓄積していたファイルが失われることを意味します。

対策は大きく以下の 2 つのパターンに分かれると思われます。

- アプリケーション実行環境が提供する永続化可能なパスに保存するよう設定変更する
- クラウドサービスが提供する永続化ストレージサービス（IIJ GIO の FV/S や、Amazon S3 など）を利用し、外部ストレージに永続化が必要なデータを保存する

いずれの場合も、アプリケーション側のソースコードでの対応が必要になります。

8.1.3. サーバ構成への配慮

MOGOK を始め、多くの PaaS 環境は多数のリクエストを効率的に処理するため、独自のサーバ構成を採っています。
多くの場合、ユーザーからのリクエストを受け付けるフロント Web サーバと、アプリケーションを実際に公開するアプリケーションサーバは、論理的に切り離された別のサーバとして構成されています。

このような環境において、例えば Web アプリケーションからユーザーに、ファイルシステム内に配置されたファイルをダウンロードさせるようなコードでは、注意が必要です。
Rails アプリケーションの場合、このような際は send_file メソッドを使って目的のファイルをダウンロードさせることが多いですが、実際にユーザーの HTTP リクエストに応答するフロントサーバが別マシンになっている場合、send_file では正しくファイルが送信できません（送信対象となるファイルをフロントサーバが取得することが出来ないため）。
このような場合には、一度目的のファイルの内容をメモリに読み込み、send_data メソッドを使ってデータとして送信を行うなどの対処が必要になります。

サーバ構成によって、アプリケーション側のソースコードに変更が必要になることは有り得ることです。
そして、PaaS 環境は手軽にアプリケーションを公開できる反面、アプリケーション実行環境の構成については、カスタマイズの余地が少ない場合が多く、この点についてアプリケーション開発者側でも十分に認識しておく必要があります。