|  |
| --- |
| 統計等情報流通連携基盤システム  操作手順書  株式会社横須賀テレコムリサーチパーク |

目次

[1. はじめに 3](#_Toc387405469)

[1.1. 統計等情報流通連携基盤システムの概要 3](#_Toc387405470)

[1.2. 本書の構成 4](#_Toc387405471)

[2. 統計等情報流通連携基盤システムの構成 5](#_Toc387405472)

[2.1. Frontend Server 6](#_Toc387405473)

[2.2. Developer Site Server 6](#_Toc387405474)

[2.3. User Information Server 7](#_Toc387405475)

[2.4. Graph Database Server 7](#_Toc387405476)

[3. API機能の操作手順 8](#_Toc387405477)

[3.1. 必要なソフトウェアのインストール 8](#_Toc387405478)

[3.1.1. nginx 8](#_Toc387405479)

[3.1.2. Ruby 8](#_Toc387405480)

[3.2. API機能のインストール 10](#_Toc387405481)

[3.3. データの登録 10](#_Toc387405482)

[3.3.1. 統計情報のRDF化と登録方法 10](#_Toc387405483)

[3.3.2. 「政府データカタログサイト」メタデータの登録方法 11](#_Toc387405484)

[3.4. 操作方法 11](#_Toc387405485)

[4. 開発者サイト機能の操作手順 12](#_Toc387405486)

[4.1. 開発者サイトのインストール 12](#_Toc387405487)

[4.1.1. Ansibleを用いた自動インストール 12](#_Toc387405488)

[4.1.2. 開発者サイト機能のインストール 15](#_Toc387405489)

[4.1.3. （参考）手動での必要なソフトウェアのインストール 16](#_Toc387405490)

[4.2. 管理者のユーザ登録 19](#_Toc387405491)

[4.3. 開発者サイトの管理者用機能 22](#_Toc387405492)

[4.3.1. フォーラム管理者編集機能 22](#_Toc387405493)

[4.3.2. ニュース管理者編集機能 25](#_Toc387405494)

[4.3.3. Admin機能 29](#_Toc387405495)

# はじめに

本書は、「情報流通連携基盤の統計情報・データカタログへの適用実証」（以下「本実証と呼ぶ」）において構築した統計等情報流通基盤システム（以下「本システム」と呼ぶ）に関する操作手順について記す。

## 統計等情報流通連携基盤システムの概要

本システムは、情報流通連携基盤システム外部仕様書[[1]](#footnote-1)（以下「外部仕様書」と呼ぶ）が規定するSPARQL-Based Commandにより、アプリケーションからのRDFデータに対する検索・取得要求を受け付ける「API機能」と、開発者を支援するためのAPIドキュメント、サンプルプログラム、ライブラリ等を提供する「開発者サイト機能」からなるシステムである（図 1）。なおAPI機能には、次世代統計システム利用API[[2]](#footnote-2)やデータカタログAPI[[3]](#footnote-3)から取得した情報をRDF化した結果を保持する機能を含む。

本システムが提供するデータは以下の通りである。

* 統計センターの「次世代統計利用システム[[4]](#footnote-4)」が提供する統計情報のうち以下のもの
  + 次世代統計利用システムAPIから取得する統計表・関連情報
    - メッシュ統計、国勢調査小地域集計
    - 平成12年から現在までの以下のデータを収録
      * 国勢調査（500mメッシュ、1kmメッシュ、小地域）
    - 統計表は都道府県毎に分割されているため、およそ2,300件存在し、含まれるデータ数はおよそ1億1千万件である
  + HTMLやExcel等のデータから抽出する統計表・関連情報
    - 日本標準産業分類の一覧
      * 平成5年に改正されてから現在までの分類（計3回）
        + HTMLで公開されている範囲を対象としている
    - 日本標準職業分類の一覧
      * 平成9年に改正されてから現在までの分類（計2回）
        + HTMLで公開されている範囲を対象としている
* 内閣官房が提供する「政府オープンデータカタログサイト試行版[[5]](#footnote-5)」が提供するデータカタログのメタデータ

これらのデータの総サイズは46GB、 RDFトリプル総数は7.7億件であった。



図 1 統計等情報流通連携基盤システムの位置づけ

なお、「API機能」は、アプリケーションに対してデータの検索・取得機能を提供する機能であり、ユーザインタフェースを持たない。一方「開発者サイト機能」は、開発者に対してAPIドキュメント、サンプルプログラム、ライブラリ等を提供する機能であり、Webページとして構築した。

## 本書の構成

本書の構成は、以下の通りである。

* 第2章では、本システムの機能構成と各機能を提供するサーバのスペックを示す。
* 第3章では、API機能の操作手順について解説する。
* 第4章では、開発者サイト機能の操作手順について解説する。

# 統計等情報流通連携基盤システムの構成

本章では、統計等情報流通連携基盤システムの構成と、構成要素である各サーバのソフトウェア構成を示す。

本システムは、以下の4つのサーバから構成される。システムの構成図を図 2に示す。また、1分間に100件程度のリクエストを受け付け、それぞれのリクエストへの応答時間を1秒程度とするためには、それぞれのサーバのスペックを表 1のように設定することが望ましい。ただし、この値は本実証で扱ったデータと実証中に受け付けたリクエストから算出したものであり、応答時間は受付クエリの複雑度に依存することに注意が必要である。

* Frontend Server（api.odstat.jp）
  + 外部のアプリケーションに対して、情報流通連携基盤システム外部仕様書が規定するAPI（Application Programming Interface）を提供する。
* Developer Site Server（developer.odstat.jp）
  + 開発者サイトをホスティングする。
* User Information Server
  + 開発者のユーザ情報を保持する。
* Graph Database Server
  + RDFデータを格納し、SPARQLで検索する機能を提供する。



図 2 システム構成図

表 1 サーバスペック一覧

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Frontend Server | Developer Site Server | User Information Server | Graph Database Server |
| CPU | Intel Xeon 1.7GHz相当 1コア | Intel Xeon 1.7GHz相当 1コア | Intel Xeon 1.7GHz相当 2コア | Intel Xeon 1.7GHz相当 8コア |
| Memory | 1.7GiB | 1.7GiB | 6.5GiB | 68GiB |
| Storage | 8GiB | 8GiB | 60GiB | 120GiB |

以下、それぞれのサーバのソフトウェア環境を示す。

## Frontend Server

Frontend Serverは、情報流通連携基盤システム外部仕様書が規定する、SPARQL-Based CommandのAPIを提供する。本実証では、表 2の環境でFrontend serverを構築した。

表 2 Frontend Server ソフトウェア環境

|  |  |
| --- | --- |
|  | ソフトウェア名・バージョン |
| Operating System | Amazon Linux AMI release 2013.09 |
| リバースプロキシ | nginx 1.4.3 |
| Ruby実行環境 | Ruby 2.1.0p0 |

## Developer Site Server

Developer Site Serverは、情報流通連携基盤の開発者サイトをホスティングする。本実証では、表 3の環境でDeveloper Site Serverを構築した。

表 3 Developer Site Server ソフトウェア環境

|  |  |
| --- | --- |
|  | ソフトウェア名・バージョン |
| Operating System | Amazon Linux AMI release 2013.09 |
| リバースプロキシ | nginx 1.4.3 |
| Database | PostgreSQL 9.3.1 |
| Ruby実行環境 | Ruby 2.0.0p247 |

## User Information Server

User Information Serverは、情報流通連携基盤の開発者サイトに登録されたユーザ情報を管理する。本実証では、表 4の環境でUser Information Serverを構築した。

表 4 User Information Server ソフトウェア環境

|  |  |
| --- | --- |
|  | ソフトウェア名・バージョン |
| Operating System | Amazon Linux AMI release 2013.09 |
| Database | MongoDB 2.4.9 |

## Graph Database Server

Graph Database Serverは、RDFで記述されたデータを格納し、情報流通連携基盤のSPARQL 1.1準拠APIで検索するための機能を提供する。本実証では、表 5の環境でGraph Database Serverを構築した。

なお、この環境でGraph Database Serverを構築する場合，Graph Database Systemのインデクス用に1トリプルあたり39バイト必要である。従って、ディスク容量は、格納するデータ本体に加えて、トリプル数×39バイト以上必要である。

表 5 Graph Database Server ソフトウェア環境

|  |  |
| --- | --- |
|  | ソフトウェア名・バージョン |
| Operating System | Amazon Linux AMI release 2013.09 |
| Graph Database System | Virtuoso Open-Source Edition 6.1.8 |

# API機能の操作手順

## 必要なソフトウェアのインストール

以下の手順で必要なソフトウェアをインストールできる。

### nginx

手順

1. ターミナルから以下のコマンドを入力する。  
     
   # rpm -ivh http://nginx.org/packages/centos/6/noarch/RPMS/  
   nginx-release-centos-6-0.el6.ngx.noarch.rpm
2. ターミナルから以下のコマンドを入力する。  
     
   # yum list nginx --disablerepo=\* --enablerepo=nginx
3. ターミナルから以下のコマンドを入力し、nginxをインストールする。  
     
   # yum install nginx
4. nginxのインストール後、以下のコマンドを入力し、バージョンを確認する。  
     
   # nginx -v  
   nginx version: nginx/1.4.5
5. ターミナルから以下のコマンドを入力し、nginxを起動する。  
     
   # /etc/init.d/nginx start

### Ruby

手順

1. ターミナルから以下のコマンドを入力し、コンパイルに必要なパッケージをインストールする。  
     
   # yum -y install gcc zlib-devel openssl-devel sqlite sqlite-devel
2. ターミナルから以下のコマンドを入力する。  
     
   # yum list | grep wget
3. ターミナルから以下のコマンドを入力し、wgetをインストールする。  
     
   # yum install wget
4. ターミナルから以下のコマンドを入力し、wgetが正しくインストールされているか確認する。  
     
   # which wget  
   /usr/bin/wget
5. ターミナルから以下のコマンドを入力する。  
     
   # cd /usr/local/src  
   # wget <ftp://ftp.ruby-lang.org/pub/ruby/2.0/ruby-2.0.0-p353.tar.gz>
6. ターミナルから以下のコマンドを入力し、ファイルを展開する。  
     
   # tar zxpf ruby-2.0.0-p353.tar.gz
7. ターミナルから以下のコマンドを入力する。  
     
   # cd ruby-2.0.0-p353  
   # ./configure
8. ターミナルから以下のコマンドを入力し、Rubyをメイクする。  
     
   # make
9. ターミナルから以下のコマンドを入力し、Rubyをインストールする。  
     
   # make install
10. ターミナルから以下のコマンドを入力し、バージョンを確認する。  
      
    # ruby –v  
    ruby 2.0.0p353 (2013-11-22 revision 43784) [x86\_64-linux]

## API機能のインストール

API機能をインストールする際には、インストール対象となるサーバのホームディレクトリに、API機能部のコードを固めたoddp\_odstat.tar.gzをコピーし、展開する。

展開後、展開してできたディレクトリに移動して以下の手順を実行することでアプリケーションを起動できる。

[手順1]以下のコマンドを実行して必要なライブラリのインストールを行う。

# bundle

[手順2]以下のコマンドを実行してサーバを起動する。

# bundle exec puma --config config/puma.rb -d -e production

## データの登録

### 統計情報のRDF化と登録方法

次世代統計システム利用APIから入手した統計情報を、以下の手順でRDF化し、本システムにキャッシュする。

1. 対象とする統計表を、次世代統計利用システムからダウンロードする。
2. 取得した統計表に含まれる統計データを、統計表を表すデータセットや軸に対応したボキャブラリを利用してRDF化し、Turtle形式で出力する。  
   本手順の詳細は「付録1 実装詳細仕様書」の3.1.3.2節を参照のこと。
3. 前工程において出力されたTurtle形式のファイルを、本システムのGraph Database Serverに登録する。

### 「政府データカタログサイト」メタデータの登録方法

政府データカタログサイト試行版のシステムであるCKANに登録されているメタデータを、以下の手順で本システムにキャッシュする。

1. 以下のURLにアクセスし、CKANに登録されているデータの識別子一覧を取得する。  
   http://www.data.go.jp/data/api/action/package\_list
2. 得られた識別子をキーとして、登録されているデータのメタデータをRDF形式で取得し、Turtle形式に変換する。
3. 得られたTurtleファイルを、本システムのGraph Database Serverに登録する。

## 操作方法

　API機能は、ユーザインタフェースを持たず、操作画面はない。

# 開発者サイト機能の操作手順

## 開発者サイトのインストール

### Ansibleを用いた自動インストール

手順

1. ansible-playbookコマンドで必要なパッケージをインストールする。

$ ansible-playbook -k -i production site.yml -u root

注意:インストール処理の状況は、開発者サイトのサーバのログを参照することで、処理内容が確認できる。

/var/log/messages

注意:サーバにインストールするソフトに関しては、設定されているバージョンのソフトがすでに存在しない場合はダウンロードできず、エラーで処理が停止する場合がある。現在公開されているソフトになるように設定ファイルを編集し、再度インストール処理を行う。

例

ansible/roles/ruby\_centost/vars/main.yml

変更前

ruby\_ver: "2.0.0-p247"

変更後

ruby\_ver: "2.0.0-p353"

例:インストール時の表示

# ansible-playbook -k -i production site.yml -u root

SSH password:

PLAY [centos] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

GATHERING FACTS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [192.168.159.155]

TASK: [nginx\_centos | nginx 1/4 | Set up the RPM for CentOS] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [nginx\_centos | nginx 2/4 | List nginx] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [nginx\_centos | nginx 3/4 | Install nginx] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [nginx\_centos | nginx 4/4 | Start nginx service] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

PLAY [centos] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK: [ruby\_centos | Ruby 1/13 | Install gcc] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [ruby\_centos | Ruby 2/13 | Install zlib-devel] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [ruby\_centos | Ruby 3/13 | Install openssl-devel] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [ruby\_centos | Ruby 4/13 | Install sqlite] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [192.168.159.155]

TASK: [ruby\_centos | Ruby 5/13 | Install sqlite-devel] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [ruby\_centos | Ruby 6/13 | Install wget] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [ruby\_centos | Ruby 7/13 | Remove download file] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [ruby\_centos | Ruby 8/13 | Remove ruby-2.0.0-p353 directory] \*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [ruby\_centos | Ruby 9/13 | Download ruby 2.0.0-p353 archive] \*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [ruby\_centos | Ruby 10/13 | Extract ruby archive] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [ruby\_centos | Ruby 11/13 | Configure ruby] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [ruby\_centos | Ruby 12/13 | Make ruby] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [ruby\_centos | Ruby 13/13 | Install ruby] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

PLAY [centos] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK: [rails\_centos | Ruby on Rails 1/2 | GEM Update] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [rails\_centos | Ruby on Rails 2/2 | Install rails] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

PLAY [centos] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK: [postgresql\_centos | PostgreSQL 1/9 | Set up the RPM for CentOS] \*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [postgresql\_centos | PostgreSQL 2/9 | Install postgresql93] \*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [postgresql\_centos | PostgreSQL 3/9 | Install postgresql93-server] \*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [postgresql\_centos | PostgreSQL 4/9 | Install postgresql93-devel] \*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [postgresql\_centos | PostgreSQL 5/9 | Install postgresql93-libs] \*\*\*\*\*\*\*\*

ok: [192.168.159.155]

TASK: [postgresql\_centos | PostgreSQL 6/9 | Install postgresql93-contrib] \*\*\*\*\*

changed: [192.168.159.155]

TASK: [postgresql\_centos | PostgreSQL 7/9 | Initialization of a database] \*\*\*\*\*

changed: [192.168.159.155]

TASK: [postgresql\_centos | PostgreSQL 8/9 | Command chkconfig] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

TASK: [postgresql\_centos | PostgreSQL 9/9 | Start PostgreSQL] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.159.155]

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

192.168.159.155 : ok=29 changed=26 unreachable=0 failed=0

### 開発者サイト機能のインストール

開発者サイト本体につきましては、開発者サイトのサーバのホームディレクトリに、system.tar.gzをコピーし展開する。

展開後、展開してできたディレクトリに移動して以下の手順を実行することでアプリケーションを起動できる。

[手順1]以下のコマンドを実行して必要なライブラリのインストールを行う。

bundle

[手順2]以下のコマンドを実行してDatabaseを初期化する。

RAILS\_ENV=production rake db:migrate

[手順3]以下のコマンドを実行してサーバの起動を行う。

bundle exec puma --config config/puma.rb -d -e production

### （参考）手動での必要なソフトウェアのインストール

Ansibleを用いた自動インストールではなく、手動で必要なソフトウェアのインストールを行う場合は、以下の手順で可能である。

#### nginx

手順

1. ターミナルから以下のコマンドを入力する。  
     
   # rpm -ivh http://nginx.org/packages/centos/6/noarch/RPMS/  
   nginx-release-centos-6-0.el6.ngx.noarch.rpm
2. ターミナルから以下のコマンドを入力する。  
     
   # yum list nginx --disablerepo=\* --enablerepo=nginx
3. ターミナルから以下のコマンドを入力し、nginxをインストールする。  
     
   # yum install nginx
4. nginxのインストール後、以下のコマンドを入力し、バージョンを確認する。  
     
   # nginx -v  
   nginx version: nginx/1.4.5
5. ターミナルから以下のコマンドを入力し、nginxを起動する。  
     
   # /etc/init.d/nginx start

#### Ruby

手順

1. ターミナルから以下のコマンドを入力し、コンパイルに必要なパッケージをインストールする。  
     
   # yum -y install gcc zlib-devel openssl-devel sqlite sqlite-devel
2. ターミナルから以下のコマンドを入力する。  
     
   # yum list | grep wget
3. ターミナルから以下のコマンドを入力し、wgetをインストールする。  
     
   # yum install wget
4. ターミナルから以下のコマンドを入力し、wgetが正しくインストールされているか確認する。  
     
   # which wget  
   /usr/bin/wget
5. ターミナルから以下のコマンドを入力する。  
     
   # cd /usr/local/src  
   # wget <ftp://ftp.ruby-lang.org/pub/ruby/2.0/ruby-2.0.0-p353.tar.gz>
6. ターミナルから以下のコマンドを入力し、ファイルを展開する。  
     
   # tar zxpf ruby-2.0.0-p353.tar.gz
7. ターミナルから以下のコマンドを入力する。  
     
   # cd ruby-2.0.0-p353  
   # ./configure
8. ターミナルから以下のコマンドを入力し、Rubyをメイクする。  
     
   # make
9. ターミナルから以下のコマンドを入力し、Rubyをインストールする。  
     
   # make install
10. ターミナルから以下のコマンドを入力し、バージョンを確認する。  
      
    # ruby –v  
    ruby 2.0.0p353 (2013-11-22 revision 43784) [x86\_64-linux]

#### Ruby on Rails

手順

1. ターミナルから以下のコマンドを入力する。  
     
   # gem update --system
2. ターミナルから以下のコマンドを入力し、Ruby on Rails 3.2.15をインストールする。  
     
   # gem install rails -f -V –v 3.2.15
3. ターミナルから以下のコマンドを入力し、バージョンを確認する。  
     
   # rails –v  
   Rails 3.2.15

#### PostgreSQL

手順

1. ターミナルから以下のコマンドを入力する。  
     
   # rpm -ivh http://yum.postgresql.org/9.3/redhat/rhel-6-x86\_64/  
   pgdg-centos93-9.3-1.noarch.rpm
2. ターミナルから以下のコマンドを入力し、PostgreSQLをインストールする。  
     
   # yum -y install postgresql93 postgresql93-server  
    postgresql93-devel postgresql93-libs postgresql93-contrib
3. ターミナルから以下のコマンドを入力し、PostgreSQLのデータベースを初期化する。  
     
   # service postgresql-9.3 initdb
4. ターミナルから以下のコマンドを入力する。  
     
   # chkconfig postgresql-9.3 on
5. ターミナルから以下のコマンドを入力し、PostgreSQLを起動する。  
     
   # service postgresql-9.3 start

## 管理者のユーザ登録

以下の方法で、管理者アカウントを作成する。

[手順1] 管理者アカウントにしたいメールアドレスを使って通常のユーザ登録を行い、メールアドレス認証まですませて審査中の状態にする。

例

メールアドレス:foo@bar.com

[手順2] 開発者サイトのデータベース developer\_development を対象に psql コマンドを起動する。

例

＄ psql developer\_development

psql (8.4.18, サーバ 9.3.1)

注意： psql バージョン 8.4, サーババージョン 9.3.

"help" でヘルプを表示する.

developer\_development=>

[手順3] 以下のコマンドでメールアドレスを検索し、検索結果に表示される「id」の値を記録しておきます。

developer\_development=> SELECT id, email, forem\_admin, approved FROM users WHERE email=’foo@bar.com’;

┌─┬──────┬──────┬────┐

│id│ email │forem\_admin │approved│

├─┼──────┼──────┼────┤

│84│foo@bar.com │f │f │

└─┴──────┴──────┴────┘

(1 行)

上記の例では、idの「84」の値。

[手順4] 以下のコマンドで forem\_admin カラムと approved カラム を true にする。コマンド末尾の id 値は前項で記録した id 値を使います。

例

developer\_development=> UPDATE users SET forem\_admin=true, approved=true WHERE id=84;

UPDATE 1

developer\_development=>

[手順5] 以下のコマンドで forem\_admin カラムと approved カラム が true になったことを確認する。

例

developer\_development=> SELECT id, email, forem\_admin, approved FROM users WHERE email=’foo@bar.com’;

┌─┬──────┬──────┬────┐

│id│ email │forem\_admin │approved│

├─┼──────┼──────┼────┤

│84│foo@bar.com │t │t │

└─┴──────┴──────┴────┘

(1 行)

developer\_development=>

[手順6] 以下のコマンドで rokes\_users テーブルにデータを追加する。末尾の「84」は、idの値に置き換えます。

例

developer\_development=> INSERT INTO roles\_users VALUES (84,1);

INSERT 0 1

developer\_development=> INSERT INTO roles\_users VALUES (84,2);

INSERT 0 1

developer\_development=>

[手順7] 以下のコマンドで rokes\_users テーブル末尾にデータが追加されたことを確認する。

例

developer\_development=> SELECT \* FROM roles\_users;

┌────┬────┐

│user\_id │role\_id │

├────┼────┤

│ 2│ 1│

│～～～ 省略 ～～～│

│ 44│ 3│

│ 84│ 1│ ← 追加されている

│ 84│ 2│ ← 追加されている

└────┴────┘

(15 行)

developer\_development=>

[手順8] ￥q コマンドで終了する。

例

developer\_development=> ￥q

＄

[手順9] 開発者サイトにアクセスする。

[手順10] 「ログイン」をクリックする。

[手順11] foo@bar.com でログインできますことを確認する。

[手順12] 正常にログインできますと、メインメニューに「Admin」が追加されます。

## 開発者サイトの管理者用機能

開発者サイトに管理者アカウントでログインすると、画面右上の「Menu」タブに現れる項目が以下の3つ増加する（図 3）。

* フォーラム管理者編集
* ニュース管理者編集
* Admin



図 3 管理者用メニュー

このそれぞれの項目が提供する機能を説明する。

### フォーラム管理者編集機能

フォーラムとは、開発者サイトに登録した開発者同士、または開発者とサイト管理者が意見交換を行うための掲示板です。

開発者サイトのMenuタブの下「フォーラム管理者画面」リンクから、本機能を提供するフォーラム管理者画面に遷移する。この画面から、以下の2つの機能が利用できます。

* フォーラムの管理
* フォーラムカテゴリの管理

管理者画面には「グループの管理」というメニューも表示されますが、この機能は利用していません。

#### フォーラムの管理

　「フォーラムの管理」機能を利用すると、フォーラムの名称や説明文の編集と、投稿されたユーザの質問の公開可否の設定ができます（図 4）。

　画面の「編集」ボタンを押すと、該当するフォーラムの名称や説明文を編集できます。また、フォーラム欄にあるフォーラム名（図 4では「フォーラム」欄の下にある「ご要望」「ご質問」リンク）をクリックすると、選択したフォーラムに投稿された利用者のメッセージが表示されます。それらのメッセージの公開可否を指定できます。

　また、「フォーラムの作成」ボタンを押すと、新しいフォーラムを作成できます（図 5）。



図 4 フォーラム管理画面

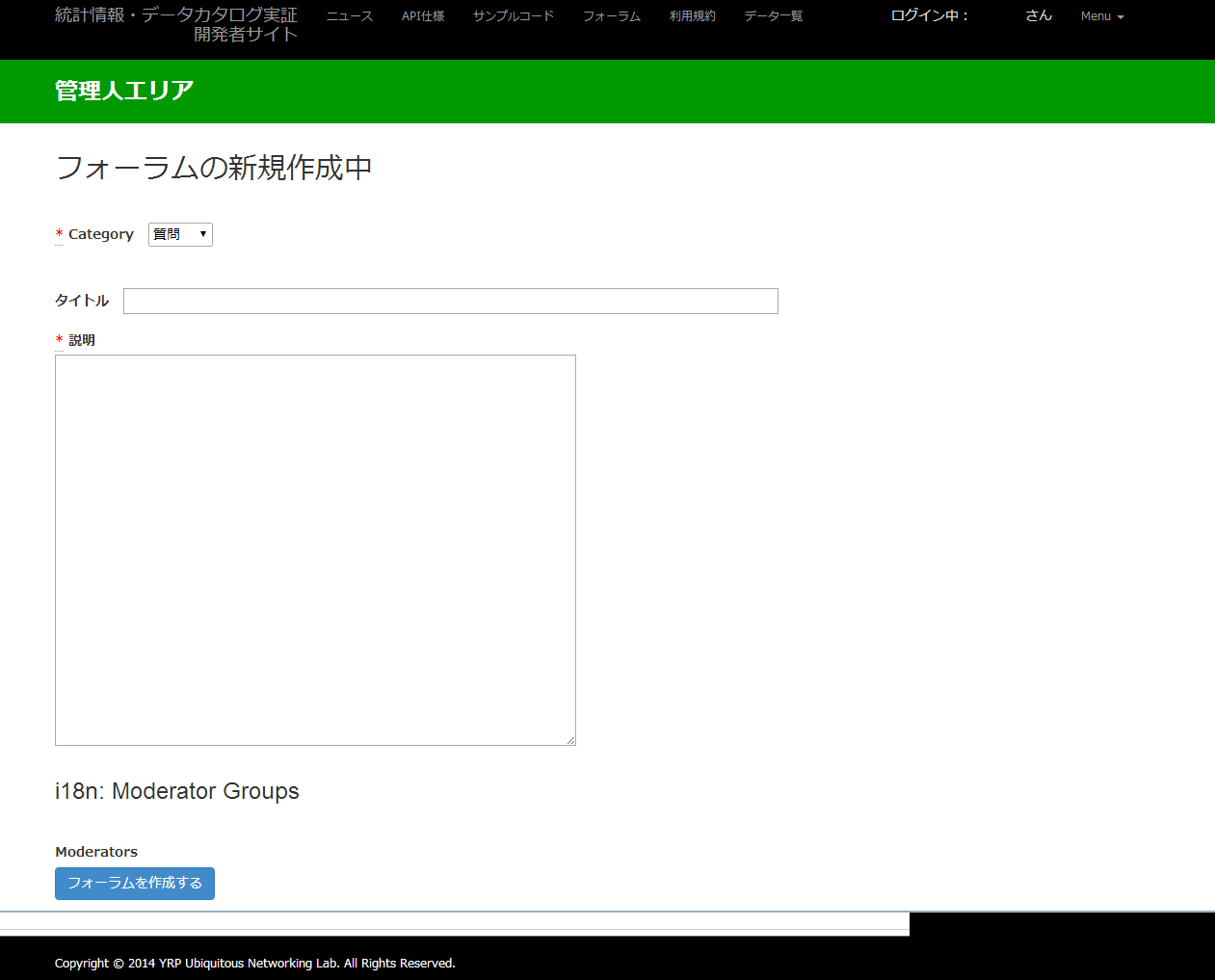


図 5 新規フォーラム作成画面

#### フォーラムカテゴリの管理機能

　「フォーラムカテゴリの管理機能」を利用すると、フォーラムのカテゴリを作成・編集・削除できます（図 6）。カテゴリにデータセットを結びつけることもできます。



図 6 フォーラムカテゴリ編集画面

### ニュース管理者編集機能

ニュース管理者編集機能は、ニュースほか開発者サイトの管理者がサイトに登録した開発者に対して連絡するページを編集する機能です。

開発者サイトのMenuタブの下「ニュース管理者画面」リンクから、本機能を提供するニュース管理者画面に遷移する（図 10）。



図 7 ニュース管理者画面

#### スタティックページの編集

この画面の「ページ」タブをクリックすると、API仕様書やデータ一覧などのスタティックページを一覧する画面に遷移する（図 8）。「新規ページを追加」ボタンを押すと、そのページの内容を編集する画面に遷移する（図 9）。タイトル・本文等必要事項を入力し、「保存」ボタンを押すと、作成したページが保存されます。

既存のページにある「編集」ボタンを押すと、図 9と同様のページが表示されます。タイトル・本文等必要事項を編集し、「保存」ボタンを押すと、編集したページが保存されます。



図 8 スタティックページ一覧画面

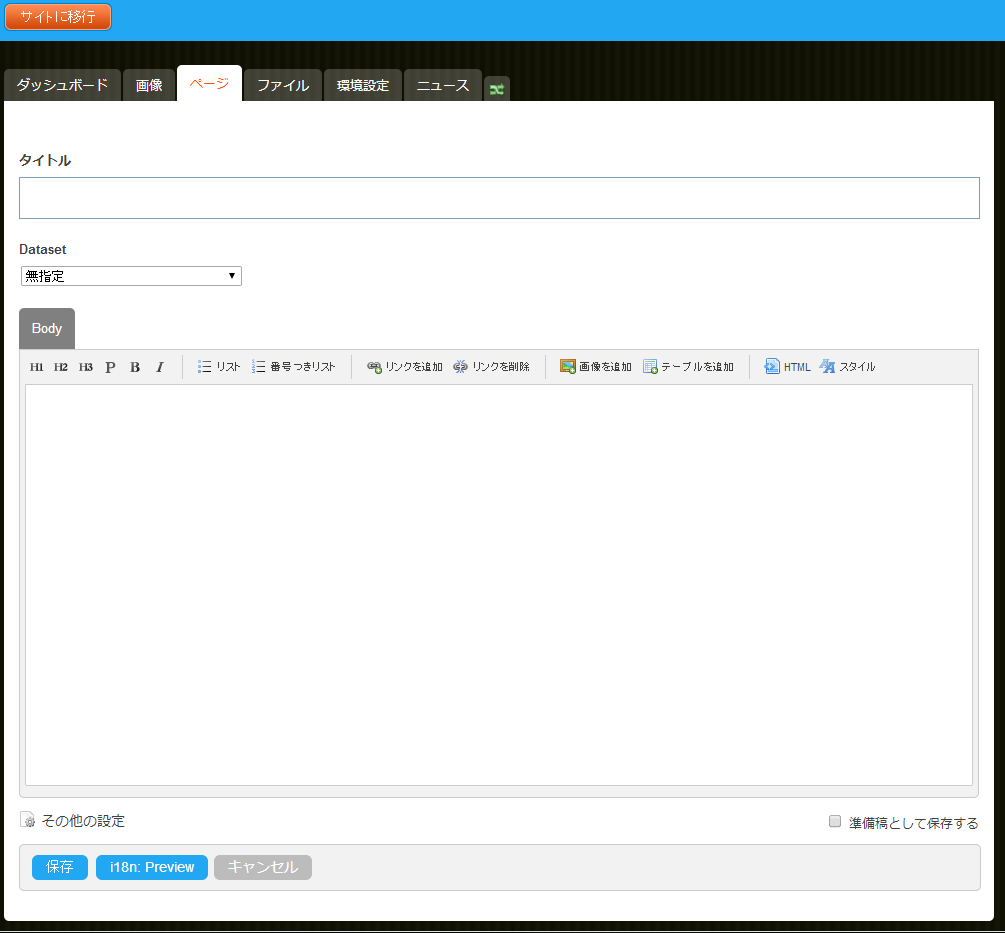


図 9 スタティックページの追加・編集画面

#### ニュースの編集

ニュース管理画面にある「ニュース」タブをクリックすると、ニュースの管理ページに遷移する（図 10）。この画面の「新規ニュースを追加」ボタンを押すと、ニュースの編集画面が表示されます（図 11）。題名・本文等必要な情報を入力し、画面下側の「保存」ボタンを押すと、このニュース項目が開発者に公開されます。



図 10 ニュース一覧画面



図 11 ニュース編集画面

### Admin機能

　「Admin」機能は、開発者サイトに登録したユーザの管理機能を提供する（図 12）。

　Approved欄がfalseになっているユーザは、管理者により承認されていません。

　各行には「詳細」「編集」「削除」という3種類のリンクがあります。これらについて、以下で説明する。

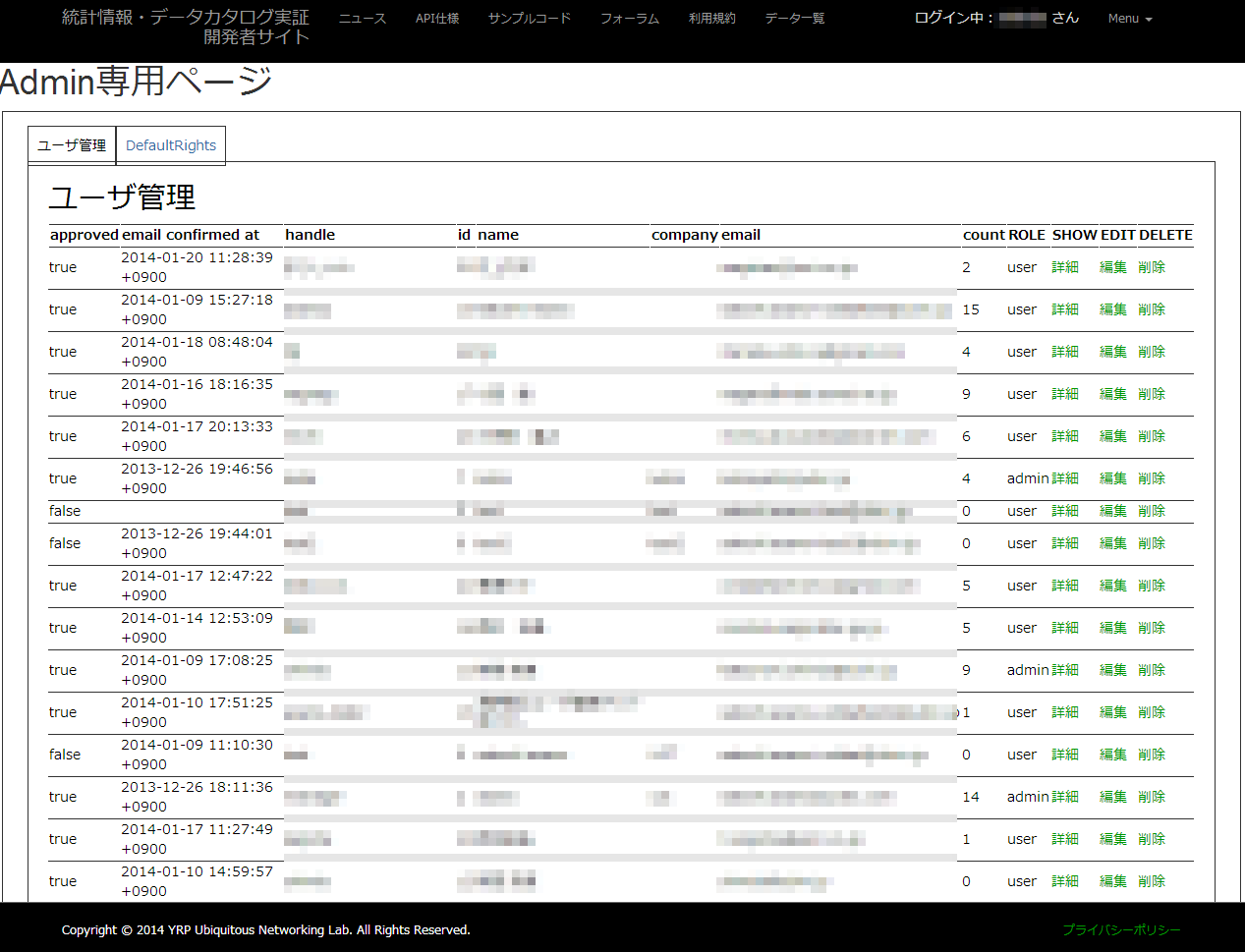


図 12 ユーザ管理ページ

#### ユーザ情報表示画面

　「詳細」リンクをたどると、そのユーザの情報を閲覧できます（図 13）。このページにある「編集」リンクをたどると、次の「ユーザ情報編集画面」に遷移する。



図 13 ユーザ情報表示画面

#### ユーザ情報編集画面

　Admin機能ページの「編集」リンクをたどるか、ユーザ情報閲覧画面の「編集」リンクをたどると、ユーザ情報を編集できます（図 14）。

　このページにある「Approved」をチェックすると、そのユーザは開発者サイトにログインできますようになります。また、このユーザに管理者資格（admin）を与えるか一般ユーザ権限（user）を与えるかも選択できます。

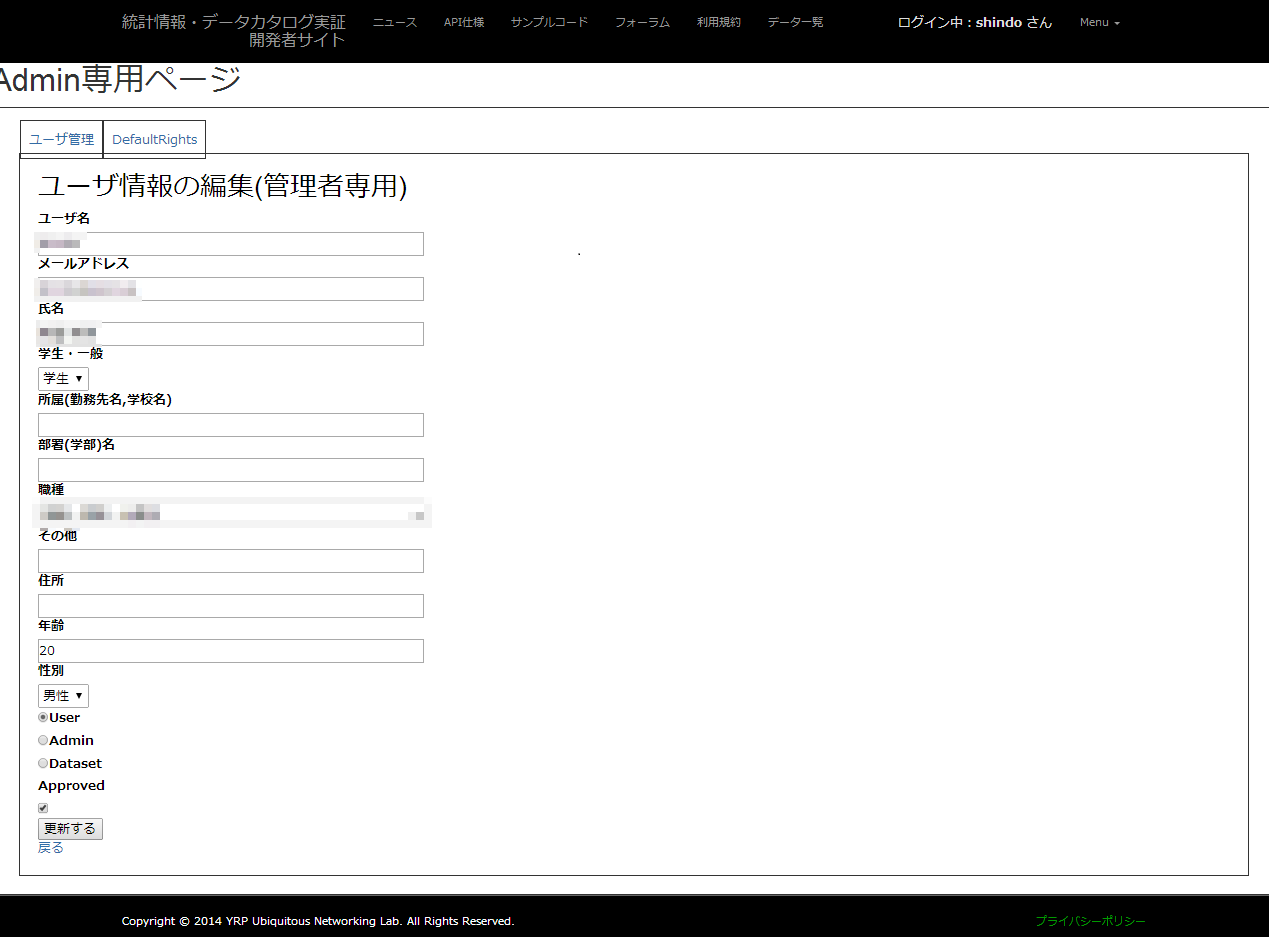


図 14 ユーザ情報編集画面

#### ユーザの削除

　Admin機能ページの「削除」リンクをたどると、当該ユーザ情報を削除できます。

　「削除」を選択すると、図 15のような確認画面が表示される。「OK」ボタンを押すと、当該ユーザを削除できます。

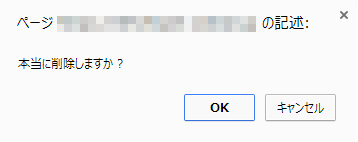


図 15 ユーザ削除確認画面

1. http://www.opendata.gr.jp/cfc/。ただし、SPARQL-Based Commandの内部で、平成25年度版仕様にあるSecurity Management Commandを利用している。 [↑](#footnote-ref-1)
2. http://statdb.nstac.go.jp/system-info/api/ [↑](#footnote-ref-2)
3. http://docs.ckan.org/en/latest/api/ [↑](#footnote-ref-3)
4. http://statdb.nstac.go.jp/ [↑](#footnote-ref-4)
5. http://data.go.jp/ [↑](#footnote-ref-5)