

# ROA

Resource Oriented Architecture  
リソース指向アーキテクチャー

日本アイ・ビー・エム株式会社  
クラウド・エバンジェリスト

米持 幸寿

[@pandrbox](https://twitter.com/pandrbox)



## 米持 幸寿

日本アイ・ビー・エム株式会社  
クラウド・エバンジェリスト

- 自然言語解析
- Big data
- クラウド・コンピューティング

これまで取り扱ってきた  
テクノロジー

Web2.0、リッチクライアント、  
SOA、J2EE/EJB、  
Webサービス、XML、Java

# リソース指向とはなにか

- 正統派の「RESTful」の思想
  - Webのようなサービス
    - スコープ情報はURIに
    - メソッド情報はHTTPメソッドに
- Roy Thomas Fielding 氏の論文に端を発する
  - <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>
- オライリー「RESTful Webサービス」で語られている
  - 標準ではない
  - ルールではない

# 簡単にいうと

- URI (URL) で「リソース」を示す (identity)
  - 「なにを」
  - アドレス可能
- HTTP で「メソッド」を示す
  - 「どうしたい」
  - ステートレス

# アドレス可能性

- リソースをURIで表現できる
  - <http://www.cloud.com/INSTANCE/123456>
  - <ftp://ftp.theses.com/doc?search=ROA>
- アドレス可能でないもの
  - 多くのAjaxアプリケーション
  - 多くのFLASHアプリケーション
  - キャッシュできない

# 制約された統一インターフェース

HEAD	特定リソースのメタ情報を取得
GET	特定リソース状態(コンテンツ)を取得
PUT	特定リソース状態の書き込み
DELETE	特定リソース状態を削除(除去)
POST	リソースを追加
OPTIONS	利用できるメソッド一覧の取得

CRUD(Create Read Update Delete)を実現する

# ステートレス

- リクエストを連続して呼び出す必要がない
  - 必要なパラメーターはすべて送信する
- 状態をサーバーが維持しない
  - 状態の管理はクライアント側の責任

SOAPのときも散々議論された

# 安全性 と べき等性

- 安全性
  - OPTIONS、HEAD、GET は、リソース状態に影響を与えない  
→ 平たく言うと「リードオンリー」
- べき等性
  - OPTIONS、HEAD、GET、PUT、DELETE は、1回以上呼び出しても同じ結果となる
- POSTは別格である

**注！** ↑・・・のように作らなくてはだめよ、ということ

# 中身はXMLか？

- 中身がなにであるか、は議論の中心ではない
  - HTML
  - 画像・動画・音声
  - XML
  - JSON
  - テキスト

# 例

# ROA な例: APP

## Atom Publishing Protocol

- The Atom Syndication Format:  
RSSに代わるフィードフォーマットとしてIETF  
で標準化
- APPは、ATOMを利用して記事を投稿  
(Publish)するプロトコルとしてIETFで標準化

明言はしてないが、思想はAtom on ROA

<http://tools.ietf.org/html/rfc4287>

<http://www.ietf.org/rfc/rfc5023.txt>

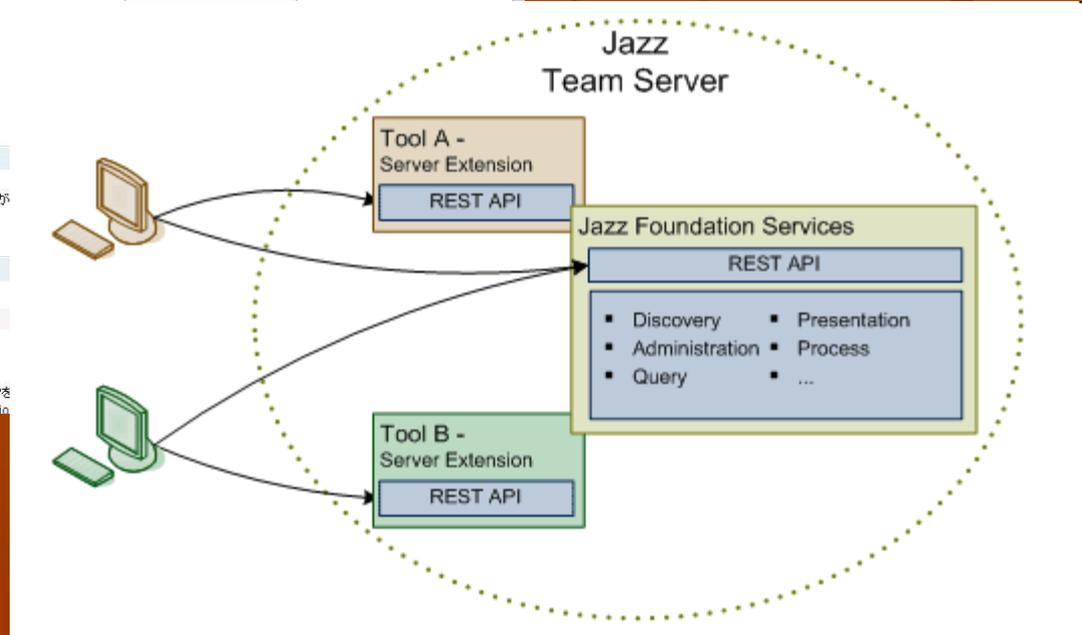
<http://abdera.apache.org/>

# WADL

## Web Application Description Language

- RESTfulサービスの記述言語
  - SOAPにとってのWSDLみたいなもの
  
- クライアント実装
  - Java™  
<http://wadl.dev.java.net>
  - Ruby  
<http://www.crummy.com/software/wadl.rb/>

# ROAな例: OSLC



Jazz / Rational Team Concert

<http://open-services.net/>

<https://jazz.net/wiki/bin/view/Main/ResourceOrientedWorkItemAPIv2>

# ROA な例: IBM SBCE

## IBM SmartCloud Enterprise

概説

コントロール・パネル

アカウント

サポート

### IBM SmartCloud Enterprise サポート・コミュニティー

このフォーラムを使用して、IBM SmartCloud Enterprise の使用中に発生した問題を報告したり、システムの現在の状況を確認したり、保守スケジュールを表示したりすることができます。このコミュニティー・エリアでは、よくある質問や、クラウドのエクスペリエンスを最大限に向上させるためのその他の有用なヒントも提供されています。

[サポート・コミュニティー・サイトにアクセスしてください](#)

### ビデオ・ライブラリー



お勧めのビデオ:

[デモ #1 - インスタンス作成 & 管理](#)

(YouTube)

サインインして、実行中のインスタンス (つまり VM) をイメージから作成する方法を説明します。

#### ビデオ・ライブラリー

[デモ #2 - SSH クライアントを使用したインスタンスへの接続](#)  
(YouTube)

[デモ #3 - IP、ストレージ、および鍵の操作](#) (YouTube)

### 資料ライブラリー



お勧め:

[REST API リファレンス](#)

Rest API リファレンス: REST API を使用してインスタンス、イメージ、ストレージ・ユニット、およびロケーションを管理する方法を説明します。

#### 資料ライブラリー

[コマンド・ライン・ツールのリファレンス](#)

[Java REST API クライアント](#)

[REST API 用の Javadoc](#)

[アカウント管理者用のウェルカム・キット](#)

### 検索



### アセット・カタログの表示

#### ログの表示

イメージおよびオフリングに関する詳細情報を検索します。

[アセット・カタログの表示](#)

### 関連リンク

[IBM の公式クラウド・コンピューティング・サイト](#)

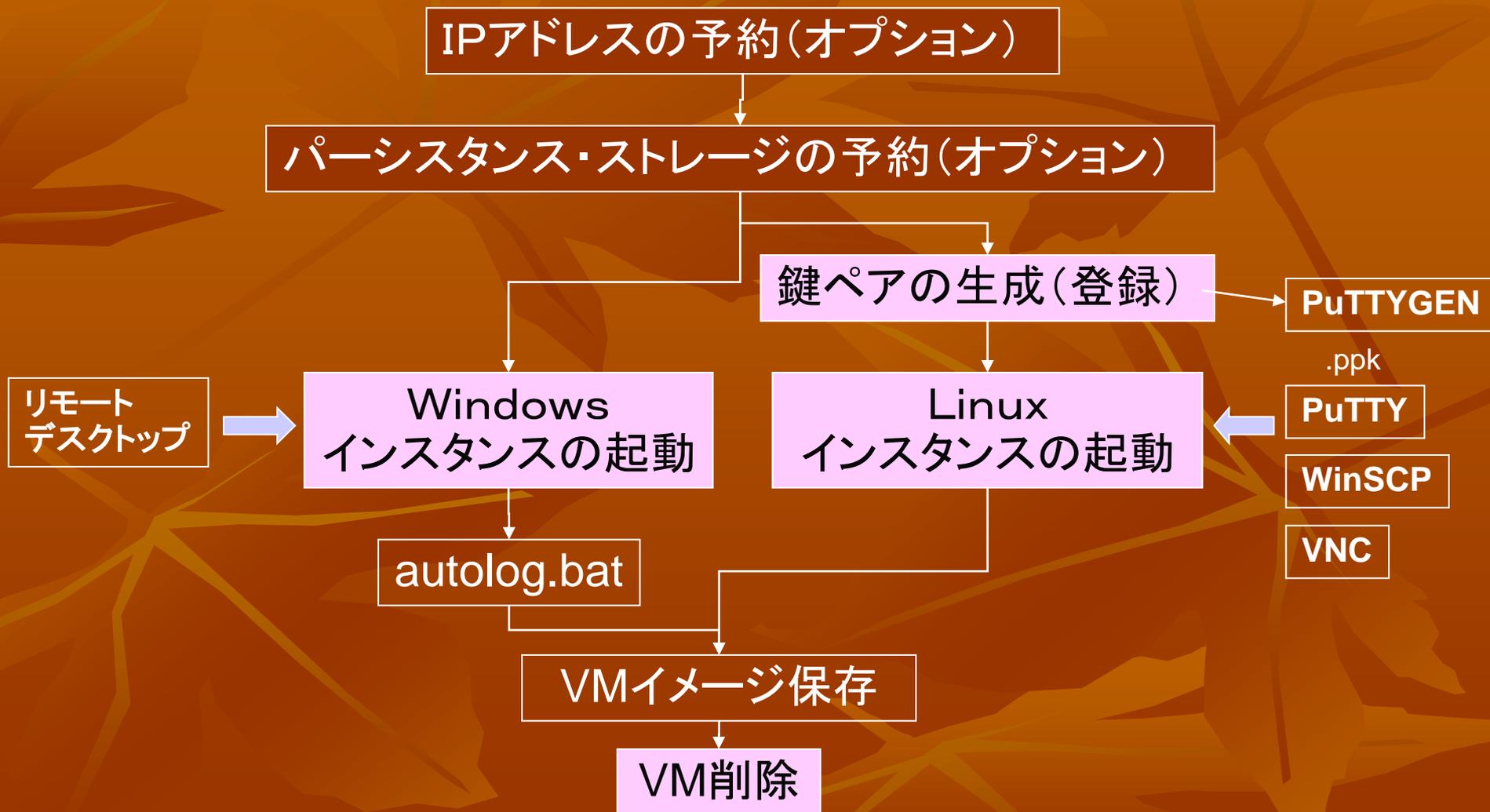
[IBM SmartCloud Enterprise 情報](#)

[サポート・コミュニティー](#)

[クラウド・コンピューティングが初めての方](#)

[ユーザーズ・ガイド](#)

# Smart Business Cloud: 操作の概要



# REST API 一覽

- Image Management
  - GET /offerings/image
  - GET /offerings/image/\$IMAGE\_ID
  - GET /offerings/image/\$IMAGE\_ID/agreement
  - DELETE /offerings/image/\$IMAGE\_ID
- Instance Management
  - GET /instances
  - GET /instances/\$INSTANCE\_ID
  - POST /instances
  - DELETE /instances/\$INSTANCE\_ID
  - PUT /instances/\$INSTANCE\_ID
    - PUT expiration Time
    - PUT state
  - GET /requests/\$REQUEST\_ID
- Storage Management
  - GET /offerings/storage
  - GET /storage
  - GET /storage/\$STORAGE\_ID
  - POST /storage
    - POST create storage
    - POST clone storage
  - DELETE /storage/\$STORAGE\_ID
- IP Address Management
  - GET /offerings/address
  - GET /addresses
  - POST /addresses
  - DELETE /addresses/\$ADDRESS\_ID
  - GET /offering/vlan
- Key Management
  - GET /keys
  - GET /keys/\$KEY\_NAME
  - POST /keys
    - POST name
    - POST name & publicKey
  - PUT /\$KEY\_NAME
    - PUT public key
    - PUT default
  - DELETE /keys/\$KEY\_NAME
- Location Management
  - GET /locations
  - GET /locations/\$LOCATION\_ID

# ROAな例: Zero Resource Manager

- Project Zero に搭載されているRESTfulサービス公開のためのフレームワーク
- appbuilderで簡単に構築可能
  - データモデルの定義
  - ドライバーの記述

	Name	Label	Required	Default Value		
abi	title		<input type="checkbox"/>		端	×
abi	author		<input type="checkbox"/>		端	×
abi	publisher		<input type="checkbox"/>		端	×
abi	category		<input type="checkbox"/>		端	×

# ROAな例: JAX-RS (JSR-311)

- Javaアノテーションを利用して、RESTfulサービスを実装するためのフレームワーク
- JavaEE™ 6に搭載 (JAX-RS 1.1)
  - Oracle GlassFish™ Open Source Edition 3.x
  - TmaxSoft の JEUS™ 7
  - IBM WebSphere™ Application Server V8.0 など
- @GETといったアノテーションでRESTを実装可能

# その他書籍の中で紹介されているもの

- Ruby on Rails
- Restlet
- Django

# 参考文献

- Resource Oriented Architecture and REST
  - European Commission  
Joint Research Centre  
Institute for Environment and Sustainability
  - [http://inspire.jrc.ec.europa.eu/reports/ImplementingRules/network/Resource\\_orientated\\_architecture\\_and\\_REST.pdf](http://inspire.jrc.ec.europa.eu/reports/ImplementingRules/network/Resource_orientated_architecture_and_REST.pdf)
- RESTとROA
  - 株式会社リコー ソフトウェア研究開発本部 山本陽平
  - [http://blogs.ricollab.jp/webtech/2008/02/rest\\_and\\_roa/](http://blogs.ricollab.jp/webtech/2008/02/rest_and_roa/)

# オライリー社 RESTful 2冊



O'REILLY\*  
オライリー・ジャパン

Leonard Richardson 著  
Sam Ruby  
山本 陽平 監訳  
株式会社クイープ 訳

O'REILLY\*  
オライリー・ジャパン

Bill Burke 著  
erbor 監訳  
菅野 真二 訳

**ROAに従わなければいけないか**  
**ROAは常に優れているか**

# ありがとうございました

- 本資料の多くは、参考文献をもとにまとめたものです。
- 記述されている内容はスピーカーの個人的な意見を含んでおり、必ずしも所属する企業・団体を代表するものではないかもしれません。
- 内容は絶対的かつ確定的なものではありません。状況による例外もありますし、時間によっても変化するでしょう。間違い等への指摘はSNS等で受け付けますが、攻撃は好みません。

IBM、WebSphere は、世界の多くの国で登録されたInternational Business Machines Corporationの商標。  
現時点での IBM の商標リストについては、[www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml)をご覧ください。  
JavaおよびすべてのJava関連の商標およびロゴは Sun Microsystems, Inc.の米国およびその他の国における商標。  
他の会社名、製品名およびサービス名等はそれぞれ各社の商標。