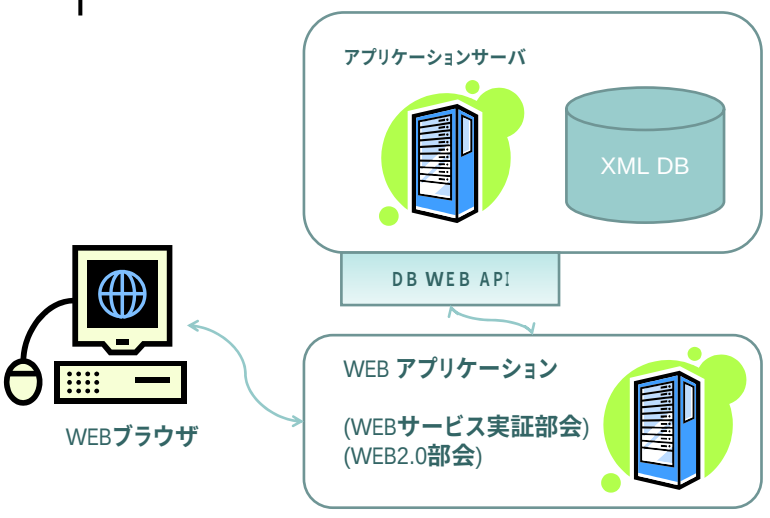
フューチャーサイト／文書管理 2.0
データベース WEB API

XMLDB部会
山口洋一（サイバーテック）

構成図



今回のAPI

■文書リスト - 取得
■文書検索 – タグ
▲文書検索 – 全文
■タグリスト – 取得 (全体)
■汎用検索 – クエリ実行

■文書 - 取得
▲文書 - 登録
▲文書 - 更新
▲文書 - 削除

■タグリスト - 取得 (文書)

■タグ - 取得
■タグ - 登録
■タグ - 更新
■タグ - 削除

■基本機能
▲オプション機能

用語

文書ID {document-id}
文書を一意に識別可能な文字列。
文書IDは文書登録時にデータベース内部で採番されます。
文書IDの形式は実装によって異なることがあります。

タグ（フォークソノミー タグ）
原則として名前と値で一組になるものを「タグ」と呼称しますが、
値のみで名前を持たないタグも許容します。
例）「拍手：喝采」（「拍手」が名前、「喝采」が値）
「注目」「あとで読む」（値のみ）
同一文書に同一内容のタグを複数登録することも可能です。

タグID（フォークソノミー タグID）
データベース全体からタグを一意に識別可能な文字列。
タグIDはタグ登録時にデータベース内部で採番されます。
タグIDの形式は実装によって異なることがあります。

補足①

1.タグIDの扱い
タグIDはデータベース全体で一意になる識別子としているため、本来は
取得・更新・削除時に文書IDを指定する必要はありませんが、登録時には
文書IDが必要なこと、論理的にはタグは特定の文書に紐付いていることから
取得・更新・削除時にも文書IDを指定する形式に統一しました。

2.文字エンコーディング
UTF-8を使用します。
これはサーバ・クライアント間のリクエスト／レスポンスおよびそれに
含まれるXML文書に適用されます。

3.REST APIの代替方式
HTTPのメソッドとしてPUTやDELETEを送るのが困難なクライアントを想定して
GETのパラメータでmethod="PUT"と指定するなど代替方式を検討していました
が今回共通のAPI仕様としては盛り込まないことにさせていただきました。
もちろんそれぞれの実装でこれらをサポートすることに制限はありません。

補足②

4.アプリケーションルート
本仕様書中で URI を表記する際に /archive/{document-id} のように表記
していますが実装によって /{application-root}/archive/{document-id} の
ようになる可能性があります。
従ってクライアント側の実装ではサーバのホスト名、ポート名に加え、
アプリケーションルートもハードコーディングしないことをおすすめします。

5.レスポンスとして返却されるXMLの形式等
サンプルで示しているようなインデントされた形式とは限りません。
またタグリスト、メタデータリストに含まれる要素の順序は不定です。

■文書 – 取得

```
GET /archive/{document-id} HTTP/1.1
```

指定した文書(XML形式)を取得します。

例）/archive/1543
　　/archive/550e8400-e29b-41d4-a716-446655440000

```
=== response（正常）===
200 OK

=== response（該当文書なし）===
404 Not Found
```


▲文書 – 登録

```
POST /archive HTTP/1.1
Content-Type: application/xml

{document-xml}
```

文書(XML形式)を登録します。
文書中に文書IDが含まれていれば、それが文書IDとして採用されることがあります（実装による）。通常は自動採番されます。

```
=== response ===
201 Created
Location: http://host/archive/{document-id}
```

(200 OK が代用されることもあります)

エラーの場合はエラーに応じた適切なステータスコードが返されます。
正常終了およびエラー時ともにエンティティボディの内容は実装によります。

▲文書 – 更新

```
PUT /archive/{document-id} HTTP/1.1
Content-Type: application/xml

{document-xml}
```

文書(XML形式)を更新（存在しない場合は登録）します。
URI中の文書IDが有効であり、文書中の文書IDがこれと一致しない場合は
サーバ側で上書きされるか、エラーが発生します（実装による）。

```
=== response（正常）===
200 OK
```

エラーの場合はエラーに応じた適切なステータスコードが返されます。
正常終了およびエラー時ともにエンティティボディの内容は実装によります。

▲文書 – 削除

```
DELETE /archive/{document-id} HTTP/1.1
```

指定した文書(XML形式)を削除します。

```
=== response（正常）===
200 OK
```

エラーの場合はエラーに応じた適切なステータスコードが返されます。
正常終了およびエラー時ともにエンティティボディの内容は実装によります。

■タグリスト – 取得 (文書)

GET /archive/{document-id}/tags HTTP/1.1

指定した文書のタグリストを取得します。

参照：付録A タグリスト形式

=== response（正常）===
200 OK

エラーの場合はエラーに応じた適切なステータスコードが返されます。
エラー時のエンティティボディの内容は実装によります。

■タグ – 取得

GET /archive/{document-id}/tags/{tag-id} HTTP/1.1

指定の文書から指定のタグを取得します。
結果のタグリストに含まれるタグは1つです。

※このAPIは主に整合性の確保を目的としています。実用性は低いかも知れません。

=== response（正常）===
200 OK

(付録 A タグリスト形式 を参照してください)

=== response（該当タグなし）===
404 Not Found

エラーの場合はエラーに応じた適切なステータスコードが返されます。
エラー時のエンティティボディの内容は実装によります。

■タグ – 登録

```
POST /archive/{document-id}/tags HTTP/1.1
Content-Type: application/x-www-form-urlencoded

{tag-name}={tag-value}
```

指定の文書にタグを追加します。
{tag-name}が"|"(%7C)から始まるタグは登録できません。

```
=== response ===
201 Created
Location: http://host/archive/{document-id}/tags/{tag-id}
```

(200 OK が代用されることもあります)

エラーの場合はエラーに応じた適切なステータスコードが返されます。
正常終了およびエラー時ともにエンティティボディの内容は実装によります。

■タグ – 更新

```
PUT /archive/{document-id}/tags/{tag-id} HTTP/1.1
Content-Type: application/x-www-form-url-encoded

{tag-name}={tag-value}
```

指定の文書のタグを上書き更新します。

```
=== response (正常) ===
200 OK
```

エラーの場合はエラーに応じた適切なステータスコードが返されます。
正常終了およびエラー時ともにエンティティボディの内容は実装によります。

■タグ – 削除

DELETE /archive/{document-id}/tags/{tag-id} HTTP/1.1

指定の文書の指定のタグを削除します。

=== response（正常）===
200 OK

エラーの場合はエラーに応じた適切なステータスコードが返されます。
正常終了およびエラー時ともにエンティティボディの内容は実装によります。

■文書リスト – 取得

GET /archive HTTP/1.1

データベースに格納されている文書のメタデータのリストを取得します。

参照：付録B メタデータリスト形式

=== response（正常）===
200 OK

エラーの場合はエラーに応じた適切なステータスコードが返されます。
エラー時のエンティティボディの内容は実装によります。

※今回は登録される文書数が少ないことを理由に全件取得の方法のみを
提供しています。本来はn件ずつ結果を取得する方法が必要です。AtomPub
またはOpenSearchで使用されている方法を採用することが考えられます。

RFC 5023 The Atom Publishing Protocol, 10 Listing Collections
OpenSearch 1.1 Draft 3, 7 OpenSearch response elements

■文書検索 – メタデータ＆タグ①

GET /search/tags/{name1},{value1};{value2}; ... HTTP/1.1

指定のメタデータまたはタグが付与されている文書を検索し、結果をリスト形式で返します。
複数タグの指定(AND検索)は";"(%3B)で区切ります。
検索項目名と値は","(%2C)で区切ります。区切りなしの文字列は値のみの指定とみなされます。

参照：付録B メタデータリスト形式

=== response（正常）===
200 OK

エラーの場合はエラーに応じた適切なステータスコードが返されます。
エラー時のエンティティボディの内容は実装によります。

■文書検索 – メタデータ＆タグ②

メタデータ項目を指定する場合は項目およびその階層を"|"(%7C)で区切ってください。
例）|Title,about+xml+db　　(メタデータ Title が "about xml db")
　　|Creator|Name,John+Doe　(メータデータ Creator/Name が "John Doe")

"|"で始まらない項目はタグ項目とみなされます。

例）http://host/tags/speaker,John+Doe;applause,loud
　　http://host/tags/remarkable;|Creator|Name,Jane+Doe

▲文書検索 – 全文

GET /search/full-text?q={keywords} HTTP/1.1

データベース内の文書を全文検索し、ヒットした文書のメタデータの
リストを取得します。ヒットしない場合は空のリストが返ります。

検索キーワードを複数指定(AND検索)する場合は半角スペースで区切ります。

参照：付録B メタデータリスト形式

例) http://host/search/full-text?q=xmldb+rdb

=== response ===
200 OK

エラーの場合はエラーに応じた適切なステータスコードが返されます。
エラー時のエンティティボディの内容は実装によります。

■タグリスト – 取得 (全体)

GET /archive-tags HTTP/1.1

データベースに格納されているタグのリストを取得します。

参照：付録A タグリスト形式

=== response (正常) ===
200 OK

エラーの場合はエラーに応じた適切なステータスコードが返されます。
エラー時のエンティティボディの内容は実装によります。

※今回は登録される文書数が少ないことを理由に全件取得の方法のみを
提供しています。本来はn件ずつ結果を取得する方法が必要です。AtomPub
またはOpenSearchで使用されている方法を採用することが考えられます。

RFC 5023 The Atom Publishing Protocol, 10 Listing Collections
OpenSearch 1.1 Draft 3, 7 OpenSearch response elements

■汎用検索 – クエリ実行

```
GET /search/xquery?q={xquery} HTTP/1.1

POST /search/xquery HTTP/1.1
Content-Type: application/x-www-form-urlencoded

q={xquery}
```

XQueryを使用してデータベースを検索します。
結果形式は実装に依存します。

付録A - タグリスト形式

```
<tags xmlns="archive.xmlconsortium.org/seminar/documents">
  <tag id="xxx" name="xxx">xxxxx</tag>     …①
  <tag id="xxx">xxxxx</tag>                …②
    ...
</tags>
```

①…名前と値を組で持つタグ
②…値のみを持つタグ

例)

```
<tags>
  <tag id="cd1fd8f4-50f5-49bb-94af-9242223822fe" name="拍手">喝采</tag>
  <tag id="bec923f1-dfd1-444b-b6fd-deb0ad369e23">あとで読む</tag>
</tags>
```

付録B – メタデータリスト形式

```
<metadata-list
     xmlns:dc="http://purl.org/dc/elements/1.1/"
     xmlns:dcterms="http://purl.org/dc/terms/"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns="http://archive.xmlconsortium.org/seminar/documents">
  <metadata pages="nn"><!-- ページ数 -->
    <dc:identifier><!-- 文書ID --></dc:identifier>
    <dc:title><!-- タイトル --></dc:title>
    <dc:coverage><!-- 発表場所等 --></dc:coverage>
    <dc:date><!-- 日付 (YYYY-MM-DD) --></dc:date>
    <dc:creator>
      <name><!-- 作者／発表者氏名 --></name>
      <company><!-- 作者／発表者所属会社名 --></company>
    </dc:creator>
    <dc:contributor><!-- 部会名 --></dc:contributor>
    <dc:relation>
      <dcterms:isFormatOf xsi:type="dcterms:URI">
        <!-- PDF等文書本体のURI-->
      </dcterms:isFormatOf>
    </dc:relation>
  </metadata>
   ...
</metadata-list>
```