

文書情報のための人文系 Web サービスの設計

白須裕之

東京大学大学院人文社会系研究科

次世代人文学開発センター

概要

人文学における様々なデジタル資料が Web で公開されているが、それらは個々に利用されるだけで、有機的に連携することが難しい。本稿は他の Web サービスとの連携を前提として、文書情報の Web サービスを RESTful に設計した。REST は *WS* - * 仕様のような大 Web サービスではなく、既存の技術のみで実現でき、様々な Web サービスをシンプルに連携できるような「設計の原則」を提唱するものである。そのために RESTful な設計に適合するような文書情報の抽象的なモデルについても述べる。

RESTful Web Services for Documents in Humanities

SHIRASU Hiroyuki

The Center for Evolving Humanities,

Graduate School of Humanities and Sociology, The University of Tokyo

Abstract

Various information resources in humanities published on the Internet. In order to recouple them, we may use a lot of programmable Web techniques. REST(Representational State Transfer) is a set of architectural principles by which we can design Web services that focus on a system's resources, and we define how Web standards, such as HTTP and URIs, are supposed to be used. In this paper, we propose that Web services of documents in humanities can be designed to be RESTful, and present a formal model of documents and their relationships on which their services are based.

1 はじめに

近年では様々な人文系の資料が Web 上で公開されて、個人でも身近に資料に接することができるような環境が整いつつあるが、それらを有機的に結びつけて使用することは、まだまだ難しい状況であると言えよう。文献 [2] では、既に構築されている大正新脩大蔵經テキストデータベースの相互運用の事例を述べて、人文学研究における人文系データベースの相互運用性の意義について述べている。

また、文献 [3] では、人文系情報サービスの質的な向上が進んでいない現状を、「人文系情報サービスの多くが互いに連携しておらず、孤立したデータが散在していて、データがデータを生み出すような環境が実現されていないから」とであると分析し、様々な解決策を提案している。本稿ではその一つの解決策として、文献 [3] が提案している、Web サービスの提供に関する「REST の勧め」の提唱に注目する。

従来、人文系データベースは Web サイトとして公開されているが、それを連携するためには、スクリーンスクレープの使用等により、非公式に Web サービス

スとして利用する必要があった。また、連携を考慮して、プログラムで利用することができるサイトでも、様々なプログラマブル Web の技術が使用されている。これは Web の再結合の技術として、Web を拡張した新しい技術を使用することを意味する。

これに対して、REST は設計の原則を設けることにより、既存の Web 技術のみを使用して、Web の再結合を実現しようというものである。文献 [3] は、人文系の情報サービスにおいても、この「REST による Web サービスの勧め」を提唱した訳である。本稿では、この提言を受けて、人文系の様々な Web サービスで利用できるように、文書情報のモデルを RESTful な Web サービスとして設計するとともに、汎用の文献テキストのために Web サービス・サーバーの構築を提唱する。

1.1 REST の原則について

REST という用語は幾つかの意味に用いられているので、本稿では曖昧さを排除するために、文献 [6] で示されたアーキテクチャスタイルの原則を充して

いるシステムを RESTful と呼ぶことにする。ここでは文献 [9] によるリソース志向アーキテクチャ ROA に従って、「REST の原則」について、必要な範囲で簡単に見ておこう。以下が「REST の原則」の概要である。

- アドレス可能性 — Web サービスとして提供するリソースは一意の識別子 (URI) を持つ。
- ステートレス性¹ — 通信によるリソースへのオペレーションは状態に依存しない。即ち、リソースを操作するための十分な情報が通信において渡される。
- 接続性 — リソースを辿るために、リソースはリソース間の接続を提供する。
- 統一インターフェイス — リソースへのオペレーションを統一的なインターフェイスで提供する (本稿では HTTP プロトコルを明示的に使用する)。

ここで簡単に統一インターフェイスとして、HTTP プロトコルをどのような使用するかを見ておこう。HTTP GET でリソースの表現を取得する。新しい URI への HTTP PUT、または既存の URI への HTTP POST で新しいリソースの作成を行なう。既存の URI への HTTP PUT で既存リソースを変更し、HTTP DELETE で既存リソースの削除を行なう。

特に、あるリソースに関連して存在するリソースを従属リソースと言い、その依存先のリソースを親と言う。従属リソースは親に対する POST によって作成する。このような POST を特に POST(a) と言うことがある ((a) はアペンド)。従属リソースの概念は Web サービスの設計において重要な役割を果たす。

1.2 何故 RESTful なのか？

次節以降では、人文系の様々なサービスとの連係が容易であるように、文書情報の Web サービスを RESTful であるように設計していく。ここでは何故 RESTful に設計するのかという、その意義について述べる。

文書情報を Web サービスとして提供する場合、その情報をプログラムで使用するには、必ずプロトコル (或いは API)、即ちそのサービスを使用するための約束が必要である。このために現在では様々な技術が提

供されているが、WS-* 仕様のような大 Web サービスでは、Web 標準を拡張した新たな技術が必要である。人文系の資料を公開する場合は、そのような大規模な技術を使用するよりも、既存の技術のみで実現できるのが望ましい。REST は設計の原則を示すのみで、技術としては既存の Web のみ (と若干の XML 技術) で充分である。人文系の Web サービス全てが RESTful に提供されれば、統一的なインターフェイスにより、その再結合は非常に容易になる所以である。

2 文書情報の形式的な表現

文書の構造には、XML 文書等のマークアップ言語による木構造や、TEI で使用されている XPointer によって実現されたリンク構造等、様々なものがあり、それを統一的にモデリングする必要がある。また、後で述べるように文書間の情報、例えば、校訂情報や対訳情報等が記述しやすいように、文書のモデルを設計する必要がある。本節では、文献 [10] で提出された「構造化文書の並行アライメント」に基づいた抽象的な文書モデルについて述べる。

2.1 構造化文書の表現

階層構造を持つ文書構造の表現モデルとしては色々なものが考えられるが、ここでは一つの文書は階層レベルを持った木構造であると仮定する²。

文書を構成している基本オブジェクトの列を U とする。基本オブジェクトとは対象とするテキスト断片であり、例えば、単語やパラグラフ等である。列 U から構成される n 階層のテキストは、 $U = \langle U^i \rangle_{i=1}^n$ である。但し、階層のレベル $i (1 \leq i \leq n)$ のテキストは、以下の性質を持つような列 $U^i = [u_1^i, \dots, u_{n_i}^i]$ とする。

$$1. U^0 = U,$$

$$2. 0 < i < n \text{ のとき } U^{i+1} \text{ は } U^i \text{ の分割である。}$$

即ち、

$$\bigcup_{u \in U^{i+1}} u = U^i \text{ かつ } \bigcap_{u \in U^{i+1}} u = \emptyset,$$

$$3. |U^n| = 1.$$

¹ここではクライアント側とサーバー側で維持される状態を区別する。各々アプリケーション状態、リソース状態と呼ぶが、ここで言うステートレスという用語は、リソース状態に関連する。詳しくは文献 [9] を参照。

²この抽象モデルの適用範囲については節 2.4 で議論する。

XML 文書のようなマークアップされた文書は、この文書木に対するラベル集合への関数 $L^i (i \leq n)$ を使用して表現できる。

$$L^i : U^i \rightarrow I : u \mapsto L(u).$$

但し、 I はラベルの集合である。例えば、 $I = \{\text{div}, \text{p}, \dots\}$ のような html の要素名の集合の場合、 $L(u)$ は u にマークアップされたタグを示す。また、階層が意味論的な要素を表現している場合、 L^i は i のみによって決まる恒等関数である。

2.2 関連情報の表現

ここでは一つの階層レベルに注目して、テキスト断片間の関係を表現する方法について述べる。各テキストの対応関係は、二つの文書間に含まれるテキスト断片の間に、各々の関係を定めることによって定義できる。 S, T を二つのテキストとする。

即ち、

$$\begin{aligned} S &= [s_1, s_2, \dots, s_u], \\ T &= [t_1, t_2, \dots, t_v]. \end{aligned}$$

このとき、二つのテキストの関係 $\text{Rel}(S, T)$ は $\text{Rel}(S, T) = [(\sigma_1, \tau_1), (\sigma_2, \tau_2), \dots, (\sigma_r, \tau_r)]$ で表現できる。但し、 σ_j, τ_j は各々 S, T の部分列であり、 $\{\sigma_j\}_{j=1}^r, \{\tau_j\}_{j=1}^r$ は各々 S, T の分割である。即ち、

$$\bigcup_{j=1}^r \sigma_j = S \text{ かつ } \bigcup_{j=1}^r \tau_j = T$$

が成り立つ。関係の構成に参与しないテキスト断片も存在することから、 σ_j, τ_j は各々空列であっても良い。図 1 はテキスト断片 s_i が二つのテキスト断片 t_j, t_k に対応している例である。この事実は $([s_i], [t_j, t_k]) \in \text{Rel}(S, T)$ で表現できる。

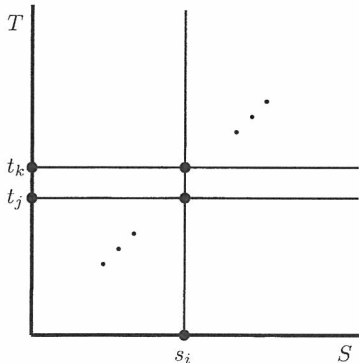


図 1. 二つのテキストの関連空間の例

2.3 多階層のテキストの関連情報

S, T を上で述べた階層構造を持つ二つの文書とし、各々 $S = \langle S^i \rangle_{i=1}^n, T = \langle T^i \rangle_{i=1}^n$ とする。多階層の関連情報は、各階層 i に関係 $\text{Rel}^i(S^i, T^i)$ を定義することで表現できる。但し、階層間に以下のような整合性を仮定する。

1. $\text{Rel}^n(S^n, T^n) = [(\sigma^n, \tau^n)]$,
2. 全ての $i < n$ に対して、 $(\sigma_j^i, \tau_j^i) \in \text{Rel}^i(S^i, T^i)$ ならば、 $(\sigma_k^{i+1}, \tau_k^{i+1}) \in \text{Rel}^{i+1}(S^{i+1}, T^{i+1})$ である $k (< n_{i+1})$ が存在して、

$$\sigma_j^i \subseteq \bigcup_{s \in \sigma_k^{i+1}} s \text{ かつ } \tau_j^i \subseteq \bigcup_{t \in \tau_k^{i+1}} t$$

が成り立つ。

最初の条件は階層のトップレベルで必ず関連があることを、第二の条件は階層 i で対応するテキスト断片は、そのすぐ上の階層 $i+1$ でも対応していて、対応するテキストどうしは分割されることがないことを示す。

2.4 文書モデルの適用範囲

文献 [8] では、文書モデルを OHCP (Ordered Hierarchy of Content Objects) のテーゼとして捉えることを提唱している。本稿で提出した抽象モデルが、文書モデルとして、どの程度の適用範囲を持つかをここで述べる。まず使用される概念を説明する。内容オブジェクト content object とは、意味に基づいて、或いは伝達の目的で使用するために、テキストを自然な単位に分割した文書の最小単位である。これは本稿の基本オブジェクトと同等のものと考えることができる。また、順序階層 ordered hierarchy とは、階層が他方をネストとする構造で作られ、それらの順序が線形関係であることを意味する。即ち、順序階層の作る構造は木構造である。文献 [8] では、OHCP のテーゼを順に提示して、様々な観点から文書モデルとしての適用範囲を検討する。

- OHCP1: テキストは内容オブジェクトの順序階層 (OHCP) である。
- OHCP2: パースペクティブは OHCP を決定する。

- *OHCP3*: パースペクティブは *OHCP* に分割できる。

パースペクティブ perspective とは、様々な文書の文書タイプを与えるもので、韻律的構造、言語的構造等を与えるものを指す。即ち、*OHCP2* はパースペクティブを *OHCP* によって与えている。二つの内容オブジェクトがオーバーラップしている場合、各々は異なるパースペクティブに属す。

パースペクティブの分割にはサブパースペクティブの概念を導入しなければならない。詳細は文献 [8] を参照してもらうことにして、ここではパースペクティブの順序が定義できて、下位のパースペクティブを指すとする。パースペクティブの分割とは、二つの内容オブジェクトがオーバーラップしている場合、各々がそのパースペクティブの異なるサブパースペクティブに属すことである。

本稿の抽象モデルでは、字義通りには *OHCP1* までの文書しか扱えない。しかし、抽象モデルの木構造はパースペクティブを与えると解釈できるため、基本オブジェクトを細かく設定して、共通のリソースとすることで、一つの文書に複数のパースペクティブを与えることができる。即ち、本稿の抽象モデルで *OHCP2* までの文書を適用範囲とすることができる³。*OHCP3* 以上の構造を持つ文書については、別に考察する必要がある。

3 RESTful な文書情報サービス

本節では RESTful な文書情報サービスの設計について述べる。但し、サービスが実際に送受信する具体的な表現フォーマットまでは議論しない。以下で示すように、構造化文書とその間の関連情報は RESTful なリソースとして、インターフェイス **Resource** を充すクラスとして定義する。

まず文書情報のデータセットを明かにし、リソースを定義しよう。前節の文書の形式的モデルに対して、RESTful なリソースを定義する。リソースのアドレッシング (文書情報と URI との関係) は、文書構造とそれに付随する関連情報を動的に構成できるように設計する⁴。次に HTTP プロトコルを利用したオペレーションを定義する。

³ *OHCP2* の文書に対するマークアップについては、例えば、文献 [1] を参照してほしい。

⁴ 例えば、対訳並行テキストに使用する場合、一方の文書構造を参照して、もう一方の文書構造を決定できるような仕組みが要求される。

3.1 構造化文書

識別子 *id* を持つ文献に対して、基本オブジェクトの列 *U* から構成される *n* 階層のテキストを $U = \langle U^i \rangle_{i=1}^n$ とする。このとき各リソースに対応する表現は以下のようなになる。

- *doc/* — 文献情報
- *doc/id* — 識別子 *id* を持つ文献
- *doc/id/i* — 階層 *i* のテキスト
- *doc/id/i/j* — 階層 *i* のテキストの *j* 番目のテキスト断片 u_j^i

3.2 関連情報

二つの構造化文書を $S = \langle S^i \rangle_{i=1}^n, T = \langle T^i \rangle_{i=1}^n$ とする。これらの構造化文書を含む多階層の関連情報は、各階層 *i* において関係 $\text{Rel}^i(S^i, T^i)$ として定義する。このとき各リソースに対応する表現は以下のようなになる。

- *rel/* — 関連情報
- *rel/id* — 識別子 *id* を持つ関連
- *rel/id/i* — 階層 *i* の関連
- *rel/id/i/k* — 階層 *i* の関連の *k* 番目の関連要素 δ_k^i

3.3 リソースタイプとオペレーション

文書情報のリソースが提供するオペレーションを定義するために、リソースを二つのリソースタイプに分類する。**Collection**、**Data** の二つである。階層の最下位のリソースは **Data** であり、それ以外は **Collection** である。これらはインターフェイス **Resource** とともに、ある種の **Composite** パターンを構成している。このリソースタイプに従って、節 1.1 で述べた統一インターフェイスに則って、オペレーションを定義すると以下のようなになる。

- **GET** — **Collection** ではその下位リソースのリストを返し、**Data** ではその詳細情報を返す。
- **PUT** — **Collection** では未使用、**Data** ではその詳細情報をアップデートする。

- POST — Collection でも Data でも下位リソースを追加する。
- DELETE — /doc、/rel では未使用。それ以外ではそのリソースを削除する。

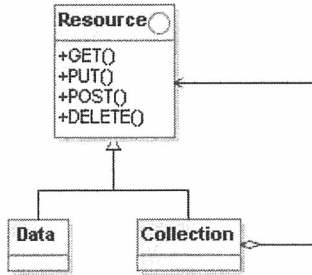


図 2. リソース

但し、実際にサービスを運用するには、各リソースがもつ詳細情報とそれに含まれる項目と内容の定義が必要である。また、サービスが実際に送受信する具体的な表現フォーマット、実際には XML 文書設計する必要がある。

4 文書サービスの利用

本稿で設計した文書情報サービスを利用して、文書を登録する例と、その他の応用について述べる。

4.1 利用のシナリオ

まずは簡単な文書を登録する例について、そのシナリオを簡単に描いてみよう。登録したい文書が章、節、行からなるものとする。行を基本オブジェクトとしよう。まず、文書名、著者等の文書情報を含む XML 文書をパラメータ⁵として、doc/に HTTP PUT メソッドを発行して、新しい文書サービスを作成する。文書構造を決める木構造の節、及び文書の基本オブジェクトにあたるリソースは、従属リソースとして、その親に HTTP POST メソッドを発行して作成する。以上のように、利用のシナリオは、節 1.1 で述べた統一インターフェイスの自然な使用を拡張したものになるのは容易に想像できるであろう。

⁵リソースとして新規の文書を登録する場合、詳細情報を最初から指定して登録する場合と、必須情報のみで登録して、HTTP POST メソッドで詳細情報を追加する場合、の二種類を用意するのが自然である。

4.2 汎用 Web サーバー

以上述べた利用のシナリオを汎用化して、文書のための汎用 Web サーバーを設計することができるであろう。そのためには文書の種類、文書構造等に依存しない設計が必要である。また、リソースのオーソライズ情報、セキュリティ、エラー、同期処理等についての考察も必要である。

4.3 テクストアライメント

本稿で設計した文書情報サービスとしては、二つの対訳テキストの間の対訳関係を格納すること、及び対訳関係を推定するアルゴリズムを扱うのが容易なように設計されている。

今、一つのテキスト S とその対訳テキスト T が与えられたと仮定する。節 2.2 で述べた記号を使用すると、 T の文書断片が S のある文書断片の翻訳になっているときに、 S, T を並行テキスト、 $\text{Rel}(S, T)$ をその対訳関係と呼ぶ。このような対訳関係 $\text{Rel}(S, T)$ を推定するアルゴリズムとして、一般に対象となる文書の性質によって様々なものが提唱されている。例えば、文献 [11] を参照。文レベルの対応については、文献 [4][5] において提唱された、文に含まれる文字数等の内部情報を使用するアルゴリズムが出発点であり、その後、訳語情報を考慮したアルゴリズム等が提唱されてきている。このような対訳関係を推定するアルゴリズムは、(多言語) 対訳コーパスを構築する上で非常に有益である。

文献 [10] で述べられた文書モデルは、対訳関係を表現するためのモデルであり、本稿で述べた文書情報 Web サービスは、このモデルを基にして設計されているので、テキストアライメントの推定アルゴリズムのために使用するの、自然なことである。

5 おわりに

本稿では人文系の様々なサービスとの連係を考慮して、文書情報サービスを RESTful に設計した。そのために文書情報モデルの形式化が必要であった。ここで定義した文書情報サービスを利用して、文献の登録、関連情報の構築等を動的に行なうことができる。因に大蔵経テキストデータベースのプロジェクトでは、この方式を利用して、英訳等の多言語対訳コーパス、索引、テキストの提供を予定している。また、こ

これらの知見を活かし、様々な文献のテキストを登録できる汎用の文書情報 Web サービス・サーバーの構築を目指す。

汎用の文書 Web サーバーを設計するにあたっては、本稿で述べた木構造を前提とする抽象モデルではなく、文書構造を関係のリソースで表現する方法も検討すべきである。これには以下のような利点が考えられる。

1. 文書構造が木構造であることを仮定せず、一般の構造が表現できる。
2. 基本オブジェクトを従属リソースとしないで使用できる。

対訳のようなテキストアライメントには、本稿で述べた方式が有利であるので、或いは両方の方式を併用するのも良いかもしれない。文書モデルの理論的な考察としては、文献 [8] 以降、様々なものが検討されているが、その文書モデルに対応した RESTful な設計については今後の検討課題とする。

本稿では文書に対する Web サービスについてののみ、その設計を述べた。これを機に今後設計される人文系の Web サービスが RESTful に行なわれるならば、人文系の様々なサービスの関係、再結合が容易になるであろう。人文系 Web サービスを RESTful に設計される方は、例えば、文献 [9] を参照されたい。

謝辞 文献 [3] の発表の際に、人文系の Web サービスの在り方について議論してくれた守岡知彦さんに感謝します。下田正弘先生には次世代人文学開発センターでお世話になっています。永崎研宣さんには共同研究の際の多くの議論を通して、様々なご教示を頂きました。大蔵経プロジェクトでは清水元広さんにお世話になっています。概要論文に対して査読頂いたお二人の方からは有益なコメントを頂戴しました。これら多くの助言、助力を賜った多くの方に感謝いたします。最後に妻留美と娘に感謝します。

参考文献

- [1] 白須裕之: 中国古典文献のための電子テキストの概念モデル, 情報処理学会研究報告, 2008-CH-77, 2008.
- [2] 永崎研宣, 下田正弘: 「人文系データベース」における相互運用性をめぐる諸問題, 情報処理学会 人文科学とコンピュータシンポジウム「じんもんこん:-)2008」, 2008.
- [3] 守岡知彦: データを生み出すデータのために, 情報処理学会 人文科学とコンピュータシンポジウム「じんもんこん:-)2008」, 2008.
- [4] P.F. Brown, J.C. Lai, R.L. Mercer: Aligning Sentences in Parallel Corpora, Proc. 29th Annual Meeting of the Association for Computational Linguistics, 1991.
- [5] K.W. Church, W.A. Gale: Concordances for Parallel Text, Proc. 29th Annual Meeting of the Association for Computational Linguistics, 1991.
- [6] R.T. Fielding: Architectural Styles and the Design of Network-based Software Architectures, Dissertation, Information and Computer Science, University of California, Irvine, 2000.
- [7] N. Ide, S. Hockey, eds.: Research in Humanities Computing, Selected papers from the ALLC/ACH Conference, Christ Church, Oxford, April 1992, Oxford University Press, 1996.
- [8] A. Renear, E. Mylonas, D. Durand: Refining our Notion of What Text Really Is: The Problem of Overlapping Hierarchies, in [7], 1996.
- [9] L. Richardson, S Ruby: RESTful Web Services — *Web services for the real world*, O'Reilly, 2007. (邦訳 山本陽平監訳, (株) クイープ訳: RESTful Web サービス, オライリー・ジャパン, 2007.)
- [10] L. Romary, P. Bonhomme: Parallel alignment of structured documents, in [11], 2000.
- [11] J. Véronis, ed.: Parallel Text Processing — *Alignment and Use of Translation Corpora*, Kluwer Academic Publisher, 2000.