

# 連携メタデータと中継プログラムによるサービス連携

牛嶋 悟†

(株)富士通研究所 Web テクノロジー研究部†

## 1. はじめに

日常の様々な業務が Web アプリケーションでサポートされるようになってきた。しかしながら、個々の Web アプリケーションは個別に動作するようになってきているため、それらの扱う情報がお互いに関連していても、ユーザ自身がそれを意識して Web アプリケーションを操作しなければならない。たとえば、ある Web アプリケーションで表示しているデータと関連するデータが別の Web アプリケーションに格納されている場合でも、ユーザは個別の Web アプリケーションにログインし、それぞれのページを自分で表示させなければならない。また、ある Web アプリケーションで入力したデータが別の Web アプリケーションの入力として要求されている場合でも、ユーザはそれぞれの Web アプリケーションに同じ情報を別々に入力しなければならない。

このため、これらを解決し、異なる Web アプリケーション上の関連する情報を自動的に表示したり、ある Web アプリケーションに入力したデータと同じ入力を要求する他の Web アプリケーションには自動的に入力を行う、といったサービス連携基盤のサポートが必要である(図1)。

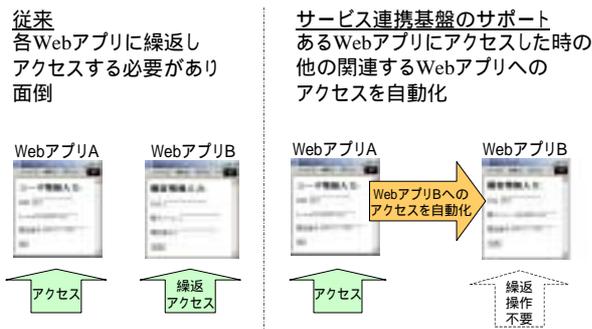


図1: サービス連携基盤の効果

## 2. フロント連携

Web アプリケーションは通常バックエンドにデータベースを持っている。そのため、Web アプリケーション間を連携させる方法の1つとして、データベース間で情報の受け渡しを行う方法がある。しかしながら、この場合 DB アクセスの部分のプログラムに手を入れなければならない。また、DB 連携で相手側の DB の参照はできたとしても、書き込みに関してはシステムの整合性がとれなくなる可能性が高く、データ変更は難しい。

これに対し、フロント側の Web のインタフェースを使い、ポータルサーバによって連携を実現するフロント連携の方法がある。この方法ではバックのサービスに手を入れずに、または Web 出力への比較的容易な変更で連携が実現可能である。その他、Web の画面をユーザに見せながら連携を実行できるため、不完全な連携でもユーザが確認して修正することができる。また、DB 連携では基本的に定期的なバッチ処理による連携しかできないが、フロント連携の場合はリアルタイムの連携が可能である。

フロント連携の実現には 2 つのアプローチの方法がある。1 つは、サービス同士の連携を実現し易いようにバックのサービスを作る方法で、もう 1 つは現在のサービスはそのままにして、ポータルで工夫することによって連携を実現する方法である。今回、前者のアプローチに基づき、異なる Web アプリケーション間のフロント連携の 1 つとして、画面上のオブジェクトの Drag&Drop による連携を実現した。

## 3. 連携メタデータと中継プログラムによる連携

今回開発したフロント連携のシステムは次のような特徴を持つ。

- (1) Web アプリケーションでプログラムを書かなくても連携が可能(ポータルにおいて連携プログラム管理)
- (2) Web アプリケーションの画面構成が変わってもプログラムを変更する必要がない

(1)を実現するために、Web アプリケーションの HTML のコメントに連携用のメタデータを記述し、ポータル上でそれを解析して連携のプログラム(Drag&Drop 実行スクリプト)を挿入する中継

Service coordination by meta-data and proxies  
 †Satoru Ushijima; Web Technology Lab.,  
 FUJITSU LABORATORIES LTD. 4-1-1, Kamikodanaka,  
 Nakahara-ku, Kawasaki 211-8588, Japan

プログラムを開発した。また(2)を実現するために、HTML 出力において、連携用の論理的なデータと対応する画面要素を分離して記述し、(1)の中継プログラムにおいてそれらのマッピングを行う方式を用いた。

今回開発したシステムの構成は図 2 の通りである。左側のブラウザは、中央のポータルサーバ Portalworks[1]を経由して右側の Web アプリケーションにアクセスする。

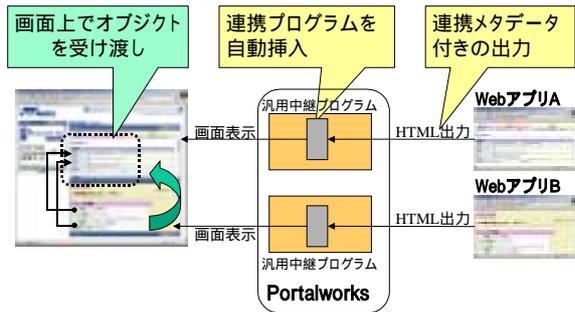


図2: 構成図

ここで Web アプリケーションは HTML の出力に図 3 のような形式で連携用のメタデータを挿入する。連携メタデータは 3 つの要素から構成され、それぞれの要素は HTML のコメント内に記述される。連携メタデータを構成する 3 つの要素は、HTML をクライアント側で表示した時の画面上での範囲を表す範囲開始と範囲終了、そしてその範囲内に表示されるものの属性情報である。これら 3 つの要素は ID によって関連付けられているため、属性情報は同一 HTML 内ならばどこに記述しても構わない。属性情報部分は、連携元では他のサービスに対して出力可能なデータと属性名、連携先では他のサービスから入力可能なデータの属性名を指定する。これによって、連携元から連携先へ属性名が一致するデータが受け渡され、連携元と連携先で属性の順番が違っていても適切に受け渡しが行われる。

このような連携メタデータを含んだ Web アプリケーションの出力がポータルに送信され、ポータル上の中継プログラムは、Web アプリケーションの出力を解析し、属性情報を取り出して一時保存した後、連携用の Drag&Drop 用のスクリプトを自動挿入してブラウザに送る。ブラウザの画面上でユーザが Drag&Drop を実行すると、Drop 側でブラウザ上のスクリプトから中継プログラムが呼び出され、Drag 側・Drop 側の中継プログラム間で複数データの受け渡しが行われ、それらのデータを使って Drop 側の画面が再描画

され、連携が完了する。

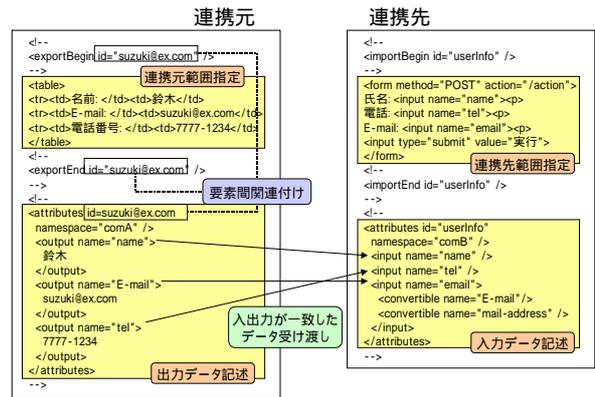


図3: 連携メタデータ

#### 4. まとめ

既存の Web アプリケーションの HTML 出力に連携メタデータを記述し、それをポータル上の中継プログラムで連携可能なページに加工して Web のインタフェースで連携を実現するフロント連携によって、Web アプリケーション間の連携を行なうシステムを実現した。

ここでは Web アプリケーションでの実現のため、HTML のコメントに連携メタデータを記述する方法をとったが、Remote Portlet[2]などの Web サービス部品でも連携メタデータを取り出すインタフェースを持たせることによって同様のことが実現可能である。また、今回既存の Web アプリケーションに簡単に導入するために連携メタデータを HTML 出力に埋め込んでいるが、これを Web アプリケーションの出力とは別に提供する方法もある。この場合、Web アプリケーション間の属性名空間の語彙の対応付けやサービス記述において、オントロジ記述言語 OWL[3]やそれに基づくサービス記述言語 OWL-S[4]といった Semantic Web の技術の活用が可能である。今後はこれらの技術を活用し、また Web アプリケーションや Remote Portlet など異なるタイプのサービスを共通に扱い、相互連携を可能にするシステムの検討を行っていく予定である。

#### 参考文献

- [1] Interstage Portalworks  
<http://interstage.fujitsu.com/jp/v6/pwks/>
- [2] Web Services for Remote Portlets  
<http://www.oasis-open.org/committees/wsrp/>
- [3] Web Ontology Language  
<http://www.w3.org/TR/owl-ref/>
- [4] OWL-based Web Service Ontology  
<http://www.daml.org/services/>