スクリーン・スクレイピングを利用したアプリケーションの連携とデバッグ

西村紅美†　　塚本享治†

東京工科大学大学院バイオ・情報メディア研究科メディアサイエンス専攻†

1. はじめに

インターネット上のアプリケーションには、Web サービスや複数の Web ページから構成される Web アプリケーションがある。この Web アプリケーションは、入力フォームを用いて画面遷移を行うものが多い。これらを連携させるには、Web ページから情報を取得するためのスクリーン・スクレイピングと、下位レベルの通信を調べることが必要である。このことを念頭に置いたアプリケーションの連携とデバッグを行うツールを開発したので報告する。

2. アプリケーション連携で発生する問題点

たとえば、英語の Web ページを検索サイトで検索し、結果を翻訳サイトで日本語に翻訳するシステムと考える。このとき、以下の問題がある。

(1) Web ページである HTML から情報を抽出することが難しい。HTML は仕様に沿っていないものが多く、レイアウトのための情報も多いためである。

(2) アプリケーションごとにパラメータの設定が異なる上に、通信のたびに変更されるものがある。変更されるパラメータは、予め設定することができない。

(3) HTML の構造などが頻繁に変更される。変更されるたびに一度書いたプログラムを書き直さなければならない。

3. アプリケーション連携とデバッグの方法

2 節に述べた問題を解決するために、Web アプリケーションを対象に、アプリケーションの連携とデバッグを簡単に行えるツールを開発した。このツールは、パラメータの設定を自動で行い、Web ページである HTML から情報抽出を用いてあらたなアプリケーションにリクエストを行うといったアプリケーションの連携とデバッグを簡単に行うことを目指した。

3.1. スクリーン・スクレイピング

Web ページから情報を取得するためには、HTML から必要な情報のみ抽出する必要がある。しかし、HTML は仕様に沿っていないものが多く、レイアウトのための情報も多い。そこで、まず HTML を XML に変換することで仕様に曖昧な部分を解消する。次に、この XML に XSLT を用いて変換し、情報を整理する。

3.2. 入力フォームを含む Web ページのパラメータの設定

Web ページに入力フォームが含まれている場合、入力フォーム固有のパラメータを持つことが多い。検索サイトの例では、検索対象言語や表示件数、検索対象とするサイトまたはドメインなどである。

そこで、対象の Web ページをいったん取得し、スクリーン・スクレイピングを用いてパラメータを取得し、それを用いた。

3.3. デバッグ

アプリケーション側の変更があった際には、Web ページを取得する事ができない。そこで、デバッグを行うための支援機能を設けた。この機能は、連携対象のアプリケーションそれぞれに行うものである。

まず、取得した Web ページを表示する。次に、取得した Web ページを、XSLT を用いて変換・抽出し、その結果を表示する。表示内容が期待した内容と違う場合は、変換・抽出に使用した XSLT をエディタに表示する。(図 1)

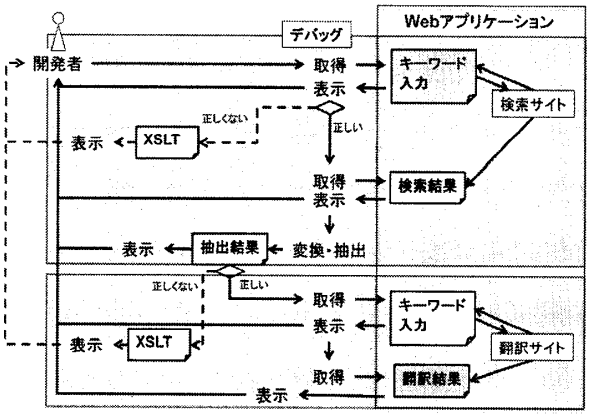

図 1 デバッグ時の画面遷移

3.4. アプリケーション連携

3.3 節を行った後、全体を通してアプリケーション連携を行う。この時、表示するのは最初のキーワード入力と最後の結果表示の Web ページ

Cooperation and debugging of applications using screen scraping

†Kumi Nishimura, Michiharu Tsukamoto
†Tokyo University of Technology
Graduate School Bionics, Computer and Media Science
Media Science Program

のみである。デバッグ時に表示した途中結果は表示しない。（図 2）

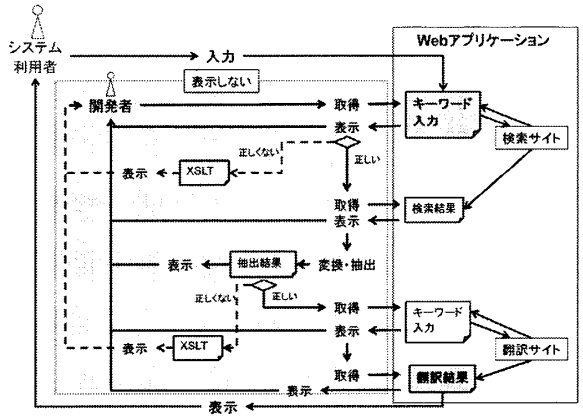

図 2 アプリケーション連携時の画面遷移

このように、デバッグとアプリケーション連携をひとつのツールで行う。

3.5. ツールの開発

3.3 節と 3.4 節で述べた動作を行うツールを開発した。このツールは、以下の 4 点の機能から成り立つ。

(1) アプリケーションへのアクセス

アプリケーションへのアクセスのインターフェースを、HTTP Client を用いて作成した。

(2) Web ページおよび抽出結果と XSLT の表示

Web ページおよび抽出結果の表示には Web ブラウザを、XSLT の表示には vi を利用した。

(3) Web ページの変換・抽出

HTTP Client を用いて取得した Web ページを変換・抽出するために、JTidy を用いた。

(4) パラメータの生成

3.2 節で述べたように、入力フォームを含む Web ページから、XSLT を用いて変換・抽出し、パラメータを生成する。パラメータの情報は、HTML の<form>要素及び子要素に記述されていることに着目し、変換・抽出を行った。

4. 検証

実際にアプリケーションの連携を行うために実行環境を構築した。行った事は以下の 2 点である。

(1) アプリケーションの構築

JSF と既存のマッシュアップツール[1]を用いて、Web アプリケーションを構築した。構築したアプリケーションは、入力フォームから入力したテキストを単純に表示するものと、Web サービスを利用するものである。利用した Web サービスは、Google、Excite 翻訳、NIKKEI NET である。

(2) Web ページを変換抽出するための XSLT の作成

構築したアプリケーションの Web ページに対応する XSLT を作成した。

そして、アプリケーションの連携とデバッグの有効性を検証した。連携の順番は、「Google、Excite 翻訳」と「テキストを単純に表示するもの、NIKKEI NET または Google、Excite 翻訳」の 2 パターンを用意した。（図 3）

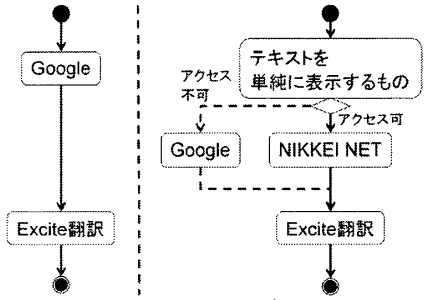

図 3 アプリケーション連携のパターン

(1) デバッグ

連携対象とするアプリケーションそれぞれに対して、デバッグ機能が有効かどうか実験した。まず、実行環境に手を加えずにツールを実行した。次に、一部の Web ページの HTML 構造を変更し、ツールを実行した。

(2) アプリケーション連携

(1)を行った後、ツールを実行し、アプリケーションの連携が行われているか実験した。

(1)では、対象の Web ページが正しく取得されている場合、該当する Web ページと抽出結果を表示した。一部の Web ページの HTML 構造を変更した場合、該当する Web ページと XSLT を表示した。これにより、デバッグを行うことができた。

(2)では、(1)で表示されていた途中結果は表示されず、最初の入力フォームを含む Web ページと連携結果の Web ページを表示した。これにより、アプリケーション連携が正しく行われていることを確認できた。

5. おわりに

以上のように、アプリケーションの連携とデバッグを簡単に行う事ができた。これにより、頻繁に変更が行われるアプリケーションに対して、直接プログラムを書く場合に比べて柔軟に対応できる。

今後は、デバッグを行う際の画面を今のエディタ表示ではなく GUI にするなど、より簡単にデバッグを行えるものにしていきたい。また、より複雑なアプリケーションの連携に対応できるよう改良する必要がある。

参考文献

[1]山本, 小山, 杉田, 塚本, マッシュアップツールの開発とそれを用いた東京都防災マップの構築, 第 69 回全国大会論文, 2007