Web探索手順記述のためのスクリプト言語の設計および実装

Design and Implementation of a Script Language Describing Web Exploration Algorithms

高嶋　活輝 † 鈴木　貢 † 中山　泰一 †
Katsuki Takashima Mitsugu Suzuki Yasuichi Nakayama

† 電気通信大学　情報工学科
Department of Computer Science, The University of Electro-Communications

概 要

インターネットの一般化とWeblog等による情報発信の普及に伴い，Web上で情報が氾濫しWeb空間をクローリングした結果のデータベースと検索エンジンを以ってしても，必要な情報を素早く的確に得ることが困難になってきている．検索条件を緩めれば多くのページにヒットしてしまい，きつくすれば必要なページを逃がしてしまうが，多くの場合，ユーザは緩めの検索条件でヒットした膨大なページから先を探索しなければならない．そこで，ユーザがWebページを探索する際に，その手順を記述することにより，探索方法の共有と保存を可能とするスクリプト言語spiを提案する．本論文では，spiの機能や有用性について議論し，spiの設計と実装を述べ，実働例を紹介する．

1. 背景と目的

現在，我々が生活する上でWeb上から役立つ情報を得られることは少なくない．しかしインターネットの普及やWeblog等の情報発信の簡易化に伴い，Web上で情報が氾濫し自分のもっとも知りたい情報を得ることが困難になっている．また，その状況は今後も深刻化していくものと予想される．

Web上から自分の求める情報を得るためにはWeb検索エンジンが利用されるが，Google[1] 等従来のWeb検索エンジンではデータベースの更新周期が比較的長く，常に最新の情報にアクセスできるとは限らない．また，クローラの巡回周期や重点巡回領域が不明であり，クローラ巡回後のサイトの更新が検索結果と食い違う場合もある．検索エンジンのAPIによりデータベースへの自作プログラムによるアクセスも可能であるが，公開されている機能は限定的である．検索結果を提示する順序の決定法や，検索の仕組みなども非公開である．

Web上の情報量が増加する中で，精度の高い検索を行うためには検索エンジン自体の技術の向上だけではなく，検索をするユーザの検索エンジンに関する知識に基づく検索条件の与えた方や，その出力を元にしてWebを効率よく探索するための技術の向上が必要である．しかしながら，たとえそれを身につけることができたとしても，どのようにリンクをたどれば目的の情報にたどり着けるのかといった事項を，他人と共有したり保存する手段は，現状では自然言語による伝聞やメモに依っている．Web文書検索を上手に行える人々は，独自の探索や分析の基準をノウハウとして有しているが，ノウハウを他人と共有したりライブラリ化がされないのは，その手順をきちんとした文法と意味論を持った言語で記述する術が無いためであると考えられる．

現状では，検索エンジンに対してページの表やリストといった構造にまで立ち入った解析を指示することはできないが，これも問題である．例えば,「表のある項目の数値がある範囲の中に入っていれば，そのページを提示せよ.」といった指示を行うことができない．

一般にWeb検索では，検索条件を緩めることで多くのページが提示され，きつくすることで提示されるページは減るが必要な情報を持つページが提示されない場合が多い．従って，検索条件を緩めに設定して，提示された順番にページを探索・分析していくが，必要な情報が先頭に近いところにあるとは限らない．このような場合，提示されたページの全てを探索するのに，探索手順を与えることでそれを自動的に実施するシステムが必要である．

また，Webページの多くは情報量の多さや，情報管理の容易化，あるいはページランク [1] を用いた検索

[1] http://www.google.co.jp/

エンジンによる評価を得るために，個々の情報を小さくまとめてリンクによって繋ぎ，複数のページに分散させている．そのような構成を持つWebページでは，Web検索で得られた親ページからのリンクの個々について，リンクをたどり，内容を検討して，リンクを戻るという手順を繰り返し，必要な情報を持つページをリストアップすることがある．これは単純な作業の繰り返しであり機械化すべきである．

そこで，本論文では，上記の問題に対する1つの解決策として，Web上のリンクをたどる手段や，取得したHTML文書のリンクの関係や内容の分析，クエリの分析等の処理手順を計算機が実行可能な形式で記述することが可能なスクリプト言語spiと，それを中核とした探索システムを提案する．

提案手法においては，探索の起点となるURLの指定は複数の方法から選択し，与えることが可能である．そのようにして選択したURLをクローリングの出発点とし，リンクをたどりながら取得・分析し，最新の情報による探索結果を得ることができる．

spiは次のような特徴を有する．

- 人の手でブラウザを通して繰り返ししなければならないパターン化した動作や複雑な作業を自動化し，ユーザのブラウジングを支援する．
- 更新の日付といったページのソースだけではわからない情報にまで立ち入って，分析の対象とすることができる．通常のプログラム言語で同様の操作を記述すると，HTTP通信処理や文字列処理などのプロトコルやプログラムの専門的な知識を必要とする．
- Webページの表やリストといった構造も分析の対象とすることができる．通常のプログラム言語で同様のことを行うには，構文解析の知識を必要とする．
- 探索の深さ等を宣言的に記述できる．これにより，スクリプト記述を簡潔にできる．

2. 関連研究と現状のWeb検索の問題

2.1 半構造化文書の検索に関する関連研究

Web文書の多くを占めるのはHTML（Hyper Text Markup Language）と呼ばれる半構造化した文書である．半構造化文書には意味的な構造はなく，HTMLのタグは視覚的な構造を表している．Web文書はデータベースの形で格納されておらず，あらかじめ与えられている構造を持っていないためある程度自由な記述が可能であるがその分検索が難しい．また，タグは入れ子構造になっているが，タグが出現する深さのレベルは一定ではない．このように，XMLのような全構造化文書とは異なり，半構造化文書の取り扱いは非常に困難である．

半構造化文書を対象とした検索言語には以下のような例がある．

- Lorel

  Lorel[2]はHTML文書を代表する半構造化文書と対象とした検索言語である．ただし，操作の対象となる各Web文書が意味的構造を持つように統一される必要があり，そのための変換を行う機構が必要である．

- UnQL

  UnQL[3]は，ラベル付エッジグラフの操作言語として開発された言語であり，Web文書や半構造化データを対象としている．UnQLは，構造とそれを成すデータ，例えばWeb上のリンクされたHTML文書間の関係とリンクを示す値やHTMLファイルに対して，任意の深さや巡回構造への問い合わせを行うことができる．UnQLを用いた検索は次のようになる．

  ```
  select "John Doe's Projects".p

  where * > "John Doe" > * > "Project".p
        in http://www.research.att.com
  ```

  これは「www.research.att.com の，‘‘John Doe’’でラベル付けされたページから辿れ，そこから更に “Project” によりラベルを付けられたページから辿ることができる全てのページを，”John Doe’s Projects”という新しいHTMLページとして返せ.」という指令である．

2.2 既存のWeb検索の問題点

この節では，既存のWeb検索の問題点について議論する．

2.2.1 検索結果の即時性

検索エンジンでは，あらかじめクローラにより取得し作成したインデックスを参照して，実際にユーザが検索する際に検索結果として提示している．しかしながら，この方法ではクローラがあるWebサイトを取得した後でそのWebサイトに更新があった場合に，検索エンジンの提示するものと，実際にそのページをアクセスしたときに違う内容になっている場合がある．この種のしばしば出会う現象の1つは，Webページの不存在である．

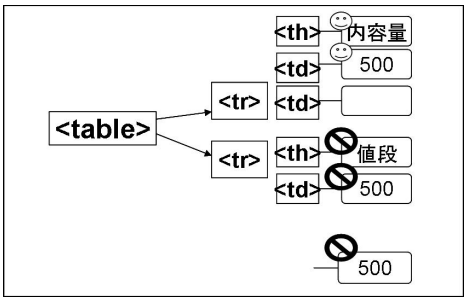

図 1: HTML 文書のテーブル構造

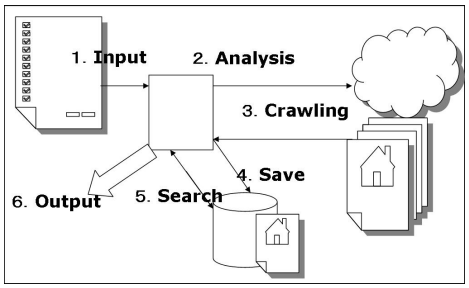

図 2: 本システムの設計

常に最新の更新情報を得るためには現在 RSS(RDF Site Summary)[6] などがある．しかし例えば RSS ではユーザの所有するアプリケーションやブラウザに組み込まれた機能に予め RSS 情報を登録する必要があり未知のページには適用できない．

提案方式では探索を行う際に実際に Web のリンクをたどるのでその時点での最新の情報での探索が可能である．

2.2.2 提示する検索結果の質的向上

検索エンジンを利用する上で検索結果の向上のためには，検索エンジン自体の能力の向上，あるいは，検索エンジンを利用するユーザが検索エンジンの特性を理解し検索エンジンに与える指示を工夫することが考えられる．検索エンジンの改良に関しては，リンクに基づくランキング手法や文脈に基づくランキング手法 [4]，検索結果の可視化に関する改良などが行われてきた．

これらは提示する結果の順位付けを改善するための手法であるが，次の節で議論する検索指示の方法の非柔軟性とも関連して，本研究で目指すようなユーザの意思を反映した柔軟な探索を行うことはできない．

2.2.3 検索指示の限界

例えば，最も細かい指定が可能であるとされている Google 検索のユーザインタフェースでは，キーワードやファイルタイプや検索ドメイン等について，and 条件/or 条件や包含/除外指定により検索条件を設定すること，および，3ヶ月単位の荒い更新日付の設定や，文書における照合範囲の指定を行うことができ，その内容を記録することで，検索方法のそのレベルでの共有や保存は可能である．

しかし，図 1 に示す表や，リストといった構造を加味した検索指定ができない．例えば，表の項目 A の値が数値 a であると言う条件で検索を行いたいとする．これに対して「A a」といった検索条件を与えると項目 A の値が数値 b であったとしても HTML 文書の他の部分に a という数値が含まれていたならヒットページとして提示されてしまう．

また，ユーザの指定したキーワードの組み合わせの一部を特別に扱い，検索アルゴリズムに調整を加えるといった工夫が行われているようであるが，これは本当にユーザが必要とする情報を除外する可能性がある．

さらに，指定したページを起点とした探索の機能は用意されていない．

確かに，検索エンジンでは例えば Google SOAP Search API[2] 等の API が提供されており，それを利用して例えば更新時刻などに関して細かい検索を指定することも可能であるが，Java 等のプログラム言語の知識を有する者でなければ扱いが難しい．また Google のデータベースを直接扱うことができるわけではなく，動作が遅い（実用的ではない）ため研究目的には多く利用されるものの一般性は低い．

3. 提案方式

図 2 に本システムの設計を示す．まず，ユーザの記述したスクリプトを読み込みそれを解析．スクリプトに基づき Web 上をクローリングし Web ページを収集，収集した Web ページ中をスクリプトに基づき探索し，結果を出力する．

本システムにおけるクローリングとは，起点となる Web ページよりリンクをたどりながら Web ページを取得して行き，取得した HTML 文書を木構造として保存することである．

[2] http://code.google.com/apis/soapsearch/

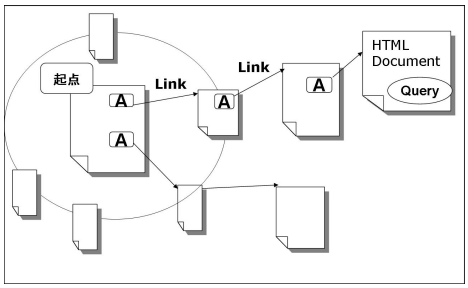

図 3: 本システムの探索範囲

3.1 人手による探索の考察

Web 検索のユーザが，提示された大量の情報のページの中から，求める情報を得る手順は例えば以下のようになるであろう．

1. 提示された各ページの要約をヒントに，それらしいページをアクセスする．
2. 文字列検索を用いたり，表などの構造に注目する場合には目視でページを一覧し，特定のキーワードを探し出したり，注目する構造を探す．
3. もしそのキーワードや構造の周辺や内部に必要な情報が記載されているなら，一連の探索は終了する．
4. 必要な情報を得られず，注目するキーワードからリンクが生えている場合は，そのリンク先をアクセスして，同じようにしてページを一覧する (2 へ)．
5. 現在のページに必要な情報も無く，進むべきリンクも無ければ，ブラウザの戻りボタンを用いて 1 つ前のページに戻る．

例えば,「キャッシュ」や「イメージ」のようなキーワードが張り付いたリンクのように，進むべきリンクからは，除外すべきキーワードを考える場合もある．

注目しているページを指している親ページに必要な情報がある場合もある．その場合には,「RETURN」や「戻る」，あるいは「サイトマップ」のような文字列が張り付いたリンクの先に行き，上記のような探索を行う．また親に戻るための手段として，提示されたページの URL の文字列を右側から左側に向かって，次の“/”に当たるまで削り，その URL でアクセスことを，有効なページが表示されるまで繰り返すこともある．

大抵のブラウザは，一度あるいは最近アクセスしたページへのリンクのボタンは，色を変える等によりユーザに知らせるようになっているが，ユーザはこれを参考にして「どうどう巡り」を避けている．

3.2 spi に要求される機能

以上よりスクリプト言語 spi を設計した．spi は以下の特徴を持つ．

1. コンパイルの手間がかからない

   spi は，多くのスクリプト言語と同様に，コンパイルの手間をかけずプログラムを実行できるインタプリタとして実装する．spi を実行し，探索結果を得た後でもその結果を踏まえてスクリプトをすぐさま変更し再び実行することが可能である．

2. HTML 文書の取得や処理に特化している

   spi は，探索が必要となった際に速やかに実行することができるようにする．実際に HTML 文書を取得し処理を行おうとすると，通信の処理，HTML 文書中のテキストデータの処理等を記述することに多くの手間を要することとなる．spi を用いることにより，このような処理を記述する手間が省ける．

3. 最新の情報を得ることができる

   上記の通り，従来の検索エンジンではクローラを事前に動作させる．spi では，探索時に実際にリンクをたどりクローリングを行うためその時点での最新情報を得ることができる．

4. 探索の範囲や深さを決めることができる

   spi で探索の対象とするのは図 3 のように探索の起点となるページから数リンクの内にたどることの可能なページに限定する．実際にクローラを動作させ Web ページを取得する範囲は，spi 内で指定できるものとする．

5. 起点 URL を指定可能である

   起点 URL は，

   - 直接入力
   - ブラウザで現在表示中の URL
   - 検索エンジンの検索結果

   のうちより選択し指定可能とする．

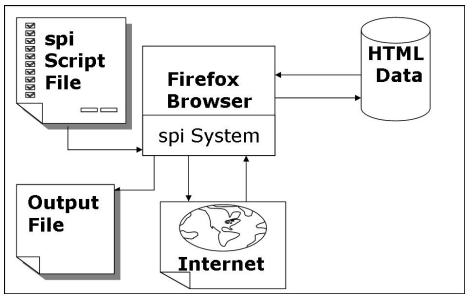

図 4: 構成

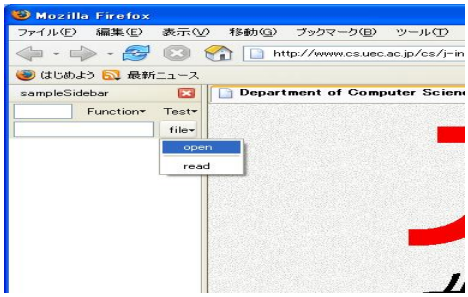

図 5: 本システムの GUI

6. HTML 文書の構造を理解できる

   従来の検索エンジンにおいて，検索の対象となるのは HTML 文書中に存在するタグ内部のテキストのみである．spi では，HTML 文書を取得する際にタグの構造をパースし，探索の対象を指定したタグに限定したりリンクをたどることができる．クエリには文字列のほか正規表現，数値および画像を指定できる．このようにユーザのブラウジングの際の行動を記述することができる．

7. フィルタリングを行える

   従来の検索エンジンのクローリングでは，全てのリンクを探索するので，必要な情報とは関係のない情報が多く含まれることが多い．spi では，検索結果に対して，HTTP リクエストヘッダに記述された更新日時や，URL 文字列でフィルタをかけることによって必要のない情報を取らず，結果のより細かい絞り込みを行ない，質のよい検索ができる．

8. XML 形式による探索結果の出力

   XML 形式の出力は，ライブラリによって GUI によるわかりやすい出力が可能である．

4. 実装

本システムは Mozilla Firefox の拡張機能として Javascript で実装を行なった．構成を図 4 に示す．

ユーザはブラウジングを行っている際に本システムを実行することで，ファイル入力ダイアログよりスクリプトファイルを読み込み，スクリプトを解釈し，実行する．スクリプトの実行の際に，本システムは起点となるページからリンクをたどり Web ページを収集し木構造として保存する．同時に Web ページからクエリにマッチするページを探索し探索結果として保存，結果を出力する．

まとめると以下のようになる．

- スクリプトファイルの読み込み
- スクリプトの解読
- HTTP リクエスト，クローリング
- 探索実行
- 探索結果リターン

以下に各機能の実装について述べる．

4.1 本システムの探索動作

Mozilla Firefox 上でブラウジングを行っている際に探索したい Web サイトを発見した場合に，本拡張機能を実行しユーザの記述したスクリプトファイルを読み込むことでスクリプトを実行する．本拡張を起動すると，Mozilla Firefox のサイドバー部分に本拡張の GUI（図 5）が表示される．

スクリプトファイルは XML[5] 形式での記述を行うこととした．XML 形式はここ数年で急速に成長しており，近年では XML 形式のデータ流通量は年々増加している．特に，XML とプログラム言語の変換に関する研究なども頻繁に行われている．そのため spi の拡張案として，将来的に通常の言語風の文法として内部で XML 形式に変換し処理することを考えている．XML 形式で記述されたスクリプトの例を図 7 に示す．

本システムにおいては，上記のようにユーザの記述したプログラムを読み込んだ際に Web 上をクローリングしサイトの情報を収集する．まず，指定したページ

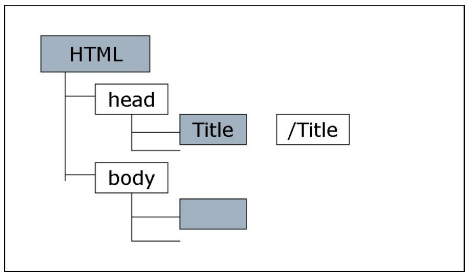

図 6: HTML の木構造

```
<program name="sample plogram v3">
  <search query="中山研究室" ignorecase="true">
    <search query="member" tag="a" ignorecase="true">
     <filter date="Sun, 1 Apr 2006 00:00:00 GMT"
       ignoreTag="script,style">
         <search query="高 (.*)?輝" />
         <search query="Katsuki(.*)?MA" />
     </filter>
    </search>
  </search>
</program>
```

図 7: 読み込ませるスクリプトファイルの例

を起点としクローリングを開始する．起点としたページに含まれるリンクからたどり着けるページ（数リンク内）を収集，タグで指定したクエリを含むページの情報を得る．該当するページが見つかった場合には，そのページを起点として再びクローリングを行う．その際の探索範囲も上記と同様である．その動作は，各タグとそれに与えるパラメタの値によって決定される．

クローリングした Web サイトは図 6 のように木構造で保存される．また，木構造の他にも Web ページの URL，タイトル，日付情報などが同時に保存される．本拡張を終了した際および，Mozilla Firefox を終了した際にはその情報が破棄される．そのため，再び探索する際には再度クローリングを行う．

4.2 本システムの機能の紹介

各タグとパラメタの詳細は以下の通りである．

- <program>: プログラム開始タグ
  - call
    ファイルパスを指定しほかのファイルのスクリプトを実行する
- <search>: 下記パラメタを与え探索を実行する．
  - query
    分析する文字列または正規表現を表す．正規表現の文法は Javascript の正規表現オブジェクトに依存する．
  - tag
    探索する範囲を指定したタグ中に限定する．
  - ignorecase
    大文字小文字を区別するか否かのオプション．true にすることで大文字と小文字の区別なしに分析が可能．

さらに，図 7 のように入れ子にすることで一つ目の <search> で見つかったページを次の <search> の起点とすることができる．例えば，図 7 においては一つ目の <search> タグにおける query である「中山研究室」が存在したページを起点とし二つ目の <search> タグにおける query を分析することになる．また，一つ目の query が存在したページが複数あった場合には，複数ページで同様に二つ目の <search> を行うこととなる．さらに，同階層に複数の <search> タグを記述した場合，同じ URL を探索の起点とする．

- <filter>: <search> を入れ子にすることにより，このタグ以降に探索する範囲にフィルタをかけることができる．
  - date
    HTTP ヘッダに記述されている Last - Modified（最終更新時刻）を取得し date 以降の更新がないページは取得しない．
  - ignoreTag
    探索する際にこのタグ内を探索しない．「,」で区切ることで複数のタグを指定することが可能．図 7 における例では <script> タグ，<style> タグ内を無視する．

5. 実験

図 7 のスクリプトを電気通信大学情報工学科ホームページを起点とし本システムで実行した結果の動作手順を以下に記す．

まず一つ目の <search> タグにより，上記ホームページ上よりリンクでたどることのできるページに存在する，「中山研究室」と書かれたページがヒットす

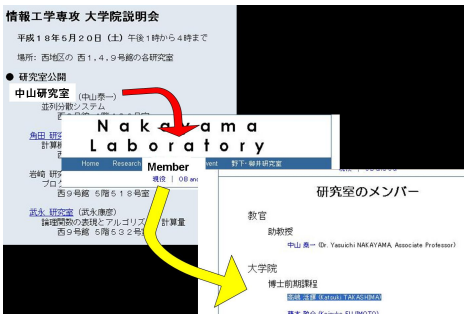

図 8: 探索の手順

る.「中山研究室」を含むページは複数存在し，その各ページを二つ目の起点とし二つ目の <search> タグによる探索が行われる．

二つ目のタグによる探索では上記二つ目の起点よりたどることが可能なページに含まれる，a タグで囲まれた「member」を分析する．二つ目の起点となるページには中山研究室ホームページへのリンクが存在している．また，中山研究室ホームページには a タグで囲まれた Member の記述が存在する．そのため，二つ目の <search> タグでは「member」の含まれる上記中山研究室ホームページ他のページがヒットし，次の探索の起点となる．

最後のタグでの探索では，filter タグにより最終更新日（2006 年 4 月）で絞込みを行なった後，中山研究室メンバーページ内に記述された「高嶋　活輝」および「Katsuki TAKASHIMA」がヒットする．中山研究室メンバーページへは中山研究室ホームページからのリンクが張られている．この様子を図 8 に示す．なお，同階層に存在する 2 つの <search> タグは，同じ URL を探索の起点とする．

spi による探索結果は，XML 形式（図 9），および，拡張機能の GUI（図 10）で出力できる．

6. まとめ

本論文では Web 探索の手順を記述し自動化するためのスクリプト言語 spi を中核とした Web 探索手法の提案を行なった．現在まで Mozilla Firefox 上で動作する拡張機能として，ユーザの記述したスクリプトファイルの読み込みおよび実行を行う部分を実装した．そして，spi の機能として，起点となるホームページからの正規表現による文字列の検索，HTML 構造解

```
<title> title</title>
<url>
 http://www.cs.uec.ac.jp/cs/j-index.html
</url>
<summary> sum</summary>
<found query=nakayama depth=1>
  <title> no title</title>
  <url>
   http://www.cs.uec.ac.jp/cs/..006-05.html
  </url>
  <summary> nakayama</summary>
  <found query=member depth=2>
    <title> Nakayama Laboratory</title>
    <url>
     http://chess.cs.uec.ac.jp/nakayama-lab/
    </url>
    <summary> Member</summary>
    <found query=高 (.*)?輝 depth=3>
      <title> People in Our Laboratory</title>
      <url>
       http://chess.cs.uec.ac.jp/..m/index.html
      </url>
      <summary> 高嶋 活輝</summary>
    </found>
  </found>
  <found query=member depth=2>
    <title> KAKUDA Human-Interface Lab. </title>
    <url> http://ltm.cs.uec.ac.jp/</url>
    <summary> member</summary>
  </found>
  <found query=member depth=2>
    <title>
    電気通信大学 阿部研究室 -Abe Laboratory-
    </title>
    <url> http://almond.cs.uec.ac.jp/</url>
    <summary> member</summary>
    <found query=query=高 (.*)?輝 depth=3>
      <title>
      電気通信大学 阿部研究室 -Abe Laboratory-
      </title>
      <url>
       http://almond.cs.uec.ac.jp/papers/papers.html
      </url>
      <summary> 高範，阿部公輝</summary>
    </found>
  </found>
</found>
```

図 9: 探索結果の例（XML 形式）

図 10: 図 9 の GUI 表示

析による絞込み，最終更新日やタグによるフィルタリングを実装した．spi では，Web 文書をツリー形式で保存することで探索範囲となるタグの指定をしたクエリの探索が可能である．また，フィルタリングによる探索の絞込み方式は，既存言語による記述と比べ，簡易かつ理解し易い．

今後はさらに，実用化に向け，文法や関数・変数の定義，結果の表示，保存，制御構文などのスクリプトの機能拡張に向けて，それらの仕様を煮詰めていく．そして，機能拡張を容易にする機能としてユーザ定義のタグの追加を簡単に書けるような仕組みや，スクリプトファイルの編集機能を加えたい．スクリプトファイルの編集機能や，また，文法を XML ベースではなく通常のプログラミング言語のような記述形式としたい．

参考文献

[1] Sergey Brin, Lawrence Page, “The Anatomy of a Large-Scale Hypertextual Web Search Engine” ,Proc. the Seventh International Conference on World Wide Web 7, pp.107–117(1998).

[2] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. Wiener, “The Lorel Query Language for Semistructured Data”, International Journal on Digital Libraries Vol.1,No.1, pp.68–88(1997).

[3] Peter Buneman, Susan Davidson, Gerd Hillebrand and Dan Suciu, “A Query Language and Optimization Techniques for Unstructured Data”, Proc. ACM SIGMOD , pp.505–516(1996).

[4] Taher H. Haveliwala, “Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search”, IEEE Transactions on Knowledge and Data Engineering, Vol.15,No.4,pp.784–796(2003).

[5] T. Bray, J. Paoli, C. M. Sperberg-McQueen, “Extensible Markup Language (XML) 1.0”, `http://www.w3.org/TR/REC-xml/`

[6] RDF Site Summary(RSS)1.0, `http://web.resource.org/rss/1.0/`