3H3-01

# ユーザ適応のためのフレームワーク“ A3 ”の提案と XSLTを用いた適応型ウェブサイトの構築

Framework for User Adaptation and Building a Adaptive Website using XSLT

官上大輔[*1] 河合由起子[*1] 田中克己[*1*2]
Daisuke Kanjo Yukiko Kawai Katsumi Tanaka

[*1]情報通信研究機構けいはんな情報通信融合研究センター
Keihannna Human Info–Communication Research Center, National Institute of Information and Communications Technology

[*2]京都大学大学院情報学研究科 社会情報学専攻
Dept. of Social Informatics, Graduate School of Informatics, Kyoto University

## 1. はじめに

ウェブの発展により，ユーザは容易に大量の情報を入手できるようになった．しかし，その情報の多くは，冗長性や表現の妥当性といった点で問題があり，必要な情報の発見や理解が困難なものになっている．そこで，ユーザの要求する情報を特定し，適切な表現を用いて提示するための手段として，ユーザ適応技術の利用が考えられる．

このような背景のもと，我々は，ユーザ適応のためのフレームワークである A3（Adaptation Anywhere & Anytime）を提案している．A3 では，ユーザやクライアントの側で集めた情報を適応的に選別する集約的なアプローチを採っておらず，A3 フレームワークによって，ユーザ適応に必要な機能を提供し，その機能を利用して各ウェブサイトやウェブアプリケーションが適応的にサービスを提供するという形態をとる．また，提供される機能を各システムが容易に利用できるようにするために，XSLT を用いた実装手段を提供している．

本稿では，A3 について概説した後，本フレームワークで集約的なアプローチを採らない理由について論じる．また，XSLT を用いた適応型システムの実装について，要件や方法を論じるとともに，作成したプロトタイプについても言及する．

## 2. A3 フレームワーク概要

A3 の構成を図 1 に示す．ユーザは任意の端末から，ウェブサイトやウェブアプリケーションなどの様々なウェブシステムを利用する．ウェブシステムは各ユーザに適した情報の選択や提示を行い，サービスを提供する．

A3 では，適応を行うために必要となるユーザの選好や知識をユーザオントロジーで表現する．ユーザオントロジーは，ウェブシステムの利用によってユーザが獲得したと想定される知識とその分類を示した分類木であり，システムとのインタラクションに基いて動的に構築される．また，構築されたユーザオントロジーは，システム間で共有される．オントロジーの動的構築と共有は，従来の適応システムが持っていた前処理による負荷の高さやシステム間の適応精度の差異といった問題を解消する．User Ontology Server（UOS）は，システムの利用が始まると，ユーザを特定し，そのユーザのユーザオントロジーを User Ontology Manager （UOM）に送る．利用を終えた

連絡先: 官上大輔，情報通信研究機構情報通信部門メディアインタラクショングループ，kanjo@nict.go.jp

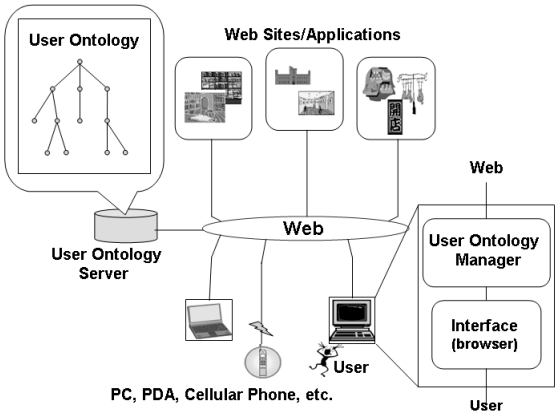


図 1: A3 の構成

際には，UOM からユーザオントロジーを受け取り，次に利用される時まで保管する．このようにして，各ユーザに対するユーザオントロジーの一意性を保証する．UOM は，ユーザに提示する情報（リソース）の選択とユーザオントロジーの構築を行う．A3 上のウェブシステムが作成する提示可能なリソースのリストに，ユーザオントロジーに基いてユーザの選好を反映した順序付けを行い，その順序に従って適当な数のリソースを選ぶことでリソース選択を実現する．また，システムは，あるリソースについてユーザが知識を得たと判断すると，そのリソースをユーザオントロジーに追加するように UOM に指示する．UOM はその指示に従ってリソースの追加を行い，ユーザオントロジーを構築する．したがって，A3 上に構築されるシステムは，リストの作成と，ユーザによる知識の獲得を入力などに基いて判断し，UOM にリソースの追加を指示する必要がある．一方，UOM はユーザオントロジーを直接的に操作し，システムからの指示を実行する．

ユーザオントロジーは RDF[RDF, RDFS] と OWL[OWL] によって記述される．ユーザオントロジーの構築はリソースの追加によって行われるため，リソースの記述がシステムごとに異なると不都合が生じる．そこで，本研究では，各システムの持つリソースは，RDF と OWL で記述されているものとする．すなわち，セマンティックウェブ [Berners-Lee 01] の実現

された状態を仮定する．

ユーザ適応を行う際，大別して，クライアントやユーザの側で集めたリソースを選別・提示するアプローチと，アプリケーションやサイトなどシステムの側でリソースを選別・提示するアプローチが考えられる．A3 では，上で述べたように，提示するリソースのリストの作成やユーザオントロジー構築のタイミングの判断などをウェブシステムが行い，実際のユーザオントロジーの操作は UOM で行う，というように両者のアプローチが混在している．その理由は以下に因る．

- オントロジーの一貫性やユーザのプライバシーの保護

  オントロジーの構築やリソース選択を行うには，ユーザオントロジーに対する操作が必要となる．しかし，ユーザオントロジーに対する操作を複数の主体が任意に行うと，ユーザオントロジーの一貫性やユーザのプライバシーを損うものとなり得る．A3 では，ユーザオントロジーに変化が生じる操作や，ユーザの知識を特定できるような操作を全て UOM で行うことで，一貫性やプライバシーを保証する．

- コンテキストなどの考慮

  集約的なアプローチを採る場合，入力処理などをユーザ/クライアントの側で全て行う必要が生じるが，あらゆるコンテキストに収集した側だけで応じるのは現実的ではない．コンテキストなどの限定を，ウェブシステムがそれぞれに行うことで，異なったコンテキストへの対応を実現する．

- Deep/Hidden Web

  近年，予め作成したウェブページ等を提示するだけでなく，データベースから動的にウェブページを生成するようなサイトなどが存在するようになっている．これらのサイトが持つデータは，一般的な検索エンジンなどで発見できないことから，Hidden web[Florescu 98]，あるいは Deep web[Bergman 01] などと呼ばれている．これら Hidden/Deep web のデータを適応的に利用するには，そのデータを所有するサイト自身が適応的にサービスを提供する必要があり，集約的なアプローチでは実現できない．

- ウェブシステム作成への動機付け

  集約的なアプローチを採用すると，ウェブシステムの意図を反映した情報の提示が行われない．リソースのリストの作成などにシステム側の意図の反映を可能とすることで，A3 上にシステムを構築する動機を与えることを考える．また，リソースの記述など，適応のために必要なメタデータの付与・共有などの動機付けとなることも期待する．

## 3. XSLT を用いたシステムの実装

本研究では，A3 の提供するユーザ適応のための機能を容易に利用できるようにするために，XSLT[clark 99] を用いたウェブシステム（ウェブサイト）の実装手段を提案している．

XSLT を使って，提示可能な[*1]全てのリソースを含む XML ドキュメントを，ユーザに適したリソースのみを含む XML/XHTML ドキュメントに変換することでリソース選択を実現する．また，生成される XML/XHTML ドキュメントに，適切なリソースをユーザオントロジーに追加する機能を持った CGI を加えることで，ユーザからの入力があった場合にオントロジーの構築が実現されるようにする．

*1 「提示可能」とはシステム利用におけるユーザの絶対的な要求などを満たしていることを示すものとする．例えば，書籍を探しているユーザに対して，CD は提示“ 不可 ”となる．

### 3.1 リソースの選択

ユーザに対して提示可能な多数のリソースが存在する時，それらの間にユーザの選好に基く優劣をつけて，適当なリソースを選択することが望ましい場面がある．本章の冒頭で述べたように A3 では，XSLT を用いて，提示可能なリソースを記した XML ドキュメントを，XSLT スタイルシートに従って，適切なリソースのみを含む XML/XHTML ドキュメントへと変換する．変換によって特定のリソースのみを選別することで，リソースの選択を実現している．

ウェブシステムの作成者は，提示可能なリソースを記した XML ドキュメントと XSLT スタイルシートを作成する必要がある．XML ドキュメントは，予め作成したものでも，システムの利用時に動的に作成されるものでも構わない．スタイルシートによって，情報提示の際の見た目などを決定できる．次に，選択の対象となるリソースを特定する必要がある．リソースは，XML ドキュメントではノードで表される．XSLT スタイルシートで，処理を行うノードを特定するために用いられる要素は，`xsl:apply-template` 要素と `xsl:for-each` 要素である．A3 では `a3:sort` 要素を，これらの要素の子として XSLT スタイルシートに記述することで，その `xsl:apply-template` 要素，あるいは `xsl:for-each` 要素によって特定されるノードについて選択の処理を行うことを示す．`a3:sort` 要素は，選択されるリソースの最大数を示す属性 `maxN` を持つ．スタイルシートの該当する箇所の例を以下に示す．

```
<xsl:apply-templates>
    <a3:sort maxN="10" />
</xsl:apply-templates>
```

UOM は，XML ドキュメントと XSLT スタイルシートを受け取り，通常の XSLT で定義される変換処理を行うとともに，スタイルシートに上の記述がある場合には，対象となっているリソースから適切なリソースを選択する．リソースの選択は，各リソースに重み付けを行い，その重みに従って上位から `manN` 個のリソースを選択することで行われる．選択されたリソースにはテンプレートが適用され，スタイルシートに記される定義に従って変換される．

### 3.2 ユーザオントロジーの構築

A3 では，ユーザの知識をユーザオントロジーで表現している．ユーザにあるリソースが提示された場合や，ユーザがあるリソースに関する情報を入力した場合，ユーザはそのリソースに関する知識を持っていると考えられる．そのような時に，ユーザが当該のリソースに関する知識を持っていることを示すため，そのリソースをユーザオントロジーへと追加する．A3 では，XSLT を用いて，変換によって生成される XHTML ドキュメントに，ユーザからの入力があった際にユーザオントロジーへリソースの追加を行う機能を加えることで，入力があった時にリソースの追加が実現されるようにする．

一般的なウェブブラウザやウェブページをインタフェースとして用いると考えた場合，入力を扱う要素として `input` 要素や `form` 要素などがある．XSLT を用いる場合，これらの要素が変換後の XML/XHTML ドキュメントに生成されるように，XSLT スタイルシートを作成する必要がある．システムの作成者はまず，これらの要素を用いた入力処理の中から，ユーザの知識獲得を示す入力を判断する必要がある．例えば，あるリソースによって表される対象を購入した，というような場合は，そのリソースに関する知識を獲得することが期待されるので，そのリソースをユーザオントロジーに追加する必要がある．一方，検索エンジンへのキーワードなどは，それが特定のリソー

スを示すものであっても，ユーザがそのリソースについて知識を持っている尤度は低く，リソースの追加は妥当ではない．システムの作成者は，リソースの追加の是非に関する上記のような判断を行う必要がある．A3 では，a3:add_resource 要素によって，子となっている form 要素への入力がリソースの追加を意味するものであることを示す．この要素の resource 属性は，追加されるリソースの uri を示す．

次に，知識獲得を意味する入力が行われた時に追加されるリソースの定義と，ユーザからの入力があった際にリソース追加の指示を UOM に送るようにする必要がある．また，たとえば，form 要素の action 属性で示される CGI のように，その入力によって本来行われるべき機能を実行できるようにする必要もある．A3 では，a3:resource-template 要素によって，追加されるリソースを定義する．resource 属性は，追加されるリソースの uri を示す．リソースの持つ各属性の属性値には，form 要素の子要素である input 要素への入力値などがなる．さらに，UOM にリソース追加の指示を送る機能と，form 要素の action 属性で示される CGI を実行する機能の 2 つの機能を持つ CGI を新たに作成し，XML ドキュメントを変換する際に，もとの CGI と置き換える．入力が行われた際には，新しい CGI が実行され，それにともない，もとの CGI が実行と，リソース追加の指示の伝達が行われる．XSLT スタイルシートの該当箇所の例を以下に示す．

```
<a3:add_resource resource="ex:example">
  <form ... action="cgi-bin/example.cgi">
    <div>attr A:<input .. name="ATTR_A"></div>
          ...
    <input type="submit" ...>
          ...
  <\form>
</a3:add_resource>
<a3:resource-template resource="ex:example">
  <ATTR_A>_ATTR_A</ATTR_A>
      ...
</a3:resource-template>
```

name 属性が ATTR_A の input 要素への入力値を attr_a とすると，追加されるリソースを RDF で記述した場合，以下のようになる．

```
<rdf:description about="ex:example">
  <ATTR_A>attr_a</ATTR_A>
      ...
</rdf:description>
```

変換時に，”example.cgi” は新しく作成される CGI に置換される．また，a3:add_resource 要素と a3:resource-template 要素は HTML 要素としての機能を持たないために削除される．作成される CGI を”new_example.cgi” とすると，上記のスタイルシートで記された箇所は以下のように変換される．

```
<form ... action="cgi-bin/new_example.cgi">
  <div>attr A:<input .. name="ATTR_A"></div>
        ...
  <input type="submit" ...>
        ...
<\form>
```

この箇所がウェブページなどに表層化され，form 要素への入力が行われると，”new_example.cgi” が実行される．それによって，”example.cgi” の実行と，UOM にリソース（ex:example）の追加の指示が送られる．UOM はこの指示に従い，(1) 既存のカテゴリへのカテゴライズ，(2) 新規カテゴリの作成，(3) カテゴリ移動によるオントロジーの再構築の 3 つの操作を行い，オントロジーを更新する．この 3 つの操作によって，リソース間に共通する属性の発見や，ユーザが関心を持ちやすいリソースがカテゴライズされるカテゴリがより上位となるようなオントロジーの構造化などが行われる．

### 3.3 構築例

ユーザ適応の過程は，ユーザの選好や知識に関する情報を獲得する過程と，その情報を用いて適応を実現する過程に大別でき，後者の過程はさらに，ユーザに適したリソースを選択する過程と，そのリソースを分かりやすく提示する過程に分けられる．A3 では現在，リソース選択の機能を提供しており，システムの構築対象としては，リコメンデーションシステムやポータルサイトのような，多数のリソースの中からユーザにとって適切なリソースを選択し，提示することが重要となるようなシステムが適する．そこで本研究では，プロトタイプシステムとして，書籍の管理と推薦を行う適応システムを作成した．

- 書籍の管理

  ユーザの所有する書籍について管理し，検索やソートなどの機能を提供する．これらの機能を実現するためには書籍に関する情報が必要となるので，ユーザには，書籍を入手した際にその書籍のデータの入力を要求する．書籍のデータの入力が行われた場合，そのユーザが書籍に関する知識を得たと判断できるので，その書籍をユーザオントロジーに追加する．書籍データの入力をウェブページから行い，データをデータベースに追加するための CGI を” book.cgi” とした時の，XSLT スタイルシートの例を以下に示す．

```
<a3:add_resource resource="ex:book">
  <form ... action="cgi-bin/book.cgi">
    <div>題名：<input .. name="TITLE"></div>
    <div>作者：<input .. name="AUTHOR"></div>
          ...
    <input type="submit" ...>
          ...
  <\form>
</a3:add_resource>
<a3:resource-template resource="ex:book">
   <TITLE>_TITLE</TITLE>
   <AUTHOR>_AUTHOR</AUTHOR>
    ...
</a3:resource-template>
```

- 書籍の推薦

  任意の書籍が，変換前の XML ドキュメントで以下のように記述されるとする．

```
<BOOK>
  <AUTHOR>星新一</AUTHOR>
  <TITLE>エヌ氏の遊園地</TITLE>
      ...
</BOOK>
```

推薦の対象となる書籍のデータを上記のように表し，XML ドキュメントに記述している時に，その中から推薦する書

籍のタイトルをウェブページのかたちでユーザに提示する場合，XSLT スタイルシートの例は以下のようになる．

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
    xmlns:xsl=
    "http://wwww3org/1999/XSL/Transform">
  <xsl:output method="html">
      ...
    <xsl:template match="/BOOKS">
      <HTML>
        ...
        <xsl:apply-templates>
          <a3:sort maxN="10" />
        </xsl:apply-templates>
        ...
      </HTML>
    </xsl:template>
    <xsl:template match="BOOK">
      <xsl:value-of select="TITLE"/>
    </xsl:template match="BOOK">
</xsl:stylesheet>
```

このスタイルシートにより，XML ドキュメントに書かれた書籍の中から，ユーザにとって関心を持ちやすい書籍のタイトルの一覧を提示する XHTML が生成される．

このシステムの作成のために行った作業は，必要な書籍データの収集を除くと上記の XML ドキュメントと XSLT スタイルシートの作成だけである．ウェブシステムの作成者の観点からは，リソース選択やオントロジー構築のアルゴリズムの考案や実装を行うことなく適応的なシステムが作成でき，容易に適応的なシステムの構築が可能になっていると考えられる．

### 3.4 考察

上で記しているように，選択対象となるリソースのリストの作成は，ウェブシステムの構築者，あるいはシステム自身に委ねられる．そこで，対象となるリソースのデータを変更することで，様々なコンテキストへの対応が可能となる．書籍の例の場合，書い忘れのチェックが目的であれば，当日発刊された書籍や前回のシステム利用後に発刊された書籍を対象とすれば良い．また，書店での購入を考える場合であれば，実際に書店に在庫のある書籍を対象とすれば良い．ユーザの位置情報などを加えることで，入店時にその書店の在庫から推薦するなどの応用も考えられる．

一方で，作成される XML ドキュメントに意味的に不適切なリソースが混入していても，A3（UOM）はそれを考慮しない．そのリソースの評価が高い場合，そのリソースが選択される．ウェブシステム，あるいはシステムの作成者は，意味的に整合するリソースを XML ドキュメントに記述する必要がある．意味的な整合性については，セマンティックウェブなどが発展することで解決される可能性もある．

また，XSLT スタイルシートを変更することでコンテキストに対応することが可能である．例えば，ユーザの利用している端末が携帯端末の場合，選択するリソースの数や提示する情報を減らしたスタイルシートを用いるといったことが考えられる．スタイルシート自身を一つのリソースとして捉えると，それ自身を選択対象として動的に変更可能となり，個々のユーザや端末の違いなどのコンテキストに応じて，適応的利用するといった可能性がある．

上記の例では，`a3:resource-template` 要素を用いて，ユーザオントロジーに追加されるリソースの持つ属性などを定義している．これはウェブシステムの作成者の作業となっているが，これを RDF スキーマや OWL で書かれたクラス定義などを外部から引用して利用することが考えられる．また，それに伴ない，各属性値を決定するための入力自身を，変換時に動的に生成するような枠組を考えれば，より容易にシステムを構築することが可能になると思われる．

## 4. おわりに

本稿では，ユーザ適応のためのフレームワークである A3 について述べた．また，A3 が，適応を行うにあたり集約的なアプローチを採らない理由について考察を行った．さらに，XSLT を利用した A3 上へのウェブシステムの構築についても述べた．

ここでは適応を実現するためのフレームワークと，その上にシステムを実装することに焦点を置き，適応精度などについて言及していない．また，選択したリソースについて，何故そのリソースを選択したのか，といった理由の提示について考慮していない．適応を行う過程は，リソースを選択する過程と，選択したリソースを提示する過程に分かれると述べたが，現状では，後者の過程について考慮していないため，理由の提示といった処理が行われていない．また，学習や解説といったタスクについては提示の過程が重要であり，この過程について考慮する必要がある．今後，これらの課題について検討し，フレームワークに機能を追加していく必要がある．また，複数のプロトタイプシステムを作成し，オントロジーの構築や共有，上記の課題について検証し，改良を加えていく必要がある．

## 参考文献


[Bergman 01] Bergman,M.K., The Deep Web: Surfacing Hidden Value, Journal of Electronic Publishing from the Univ. of Michigan Press, 2001

[Berners-Lee 01] Berners-Lee,T.,Hendler,J.,Lassile,O., The Semantic Web,Scientific American(2001)

[clark 99] Clark,J., XSL Transformations(XSLT) Version1.0, http://www.w3.org/TR/xslt, 1999

[Florescu 98] Florescu,D.,Levy,A.Y.,Mendelzon,A.O., Database Techiniques for the world wide web, A survey.SIGMOD Record,27(3),pp59-74,1998

[Kay 02] Kay,J.,Kummerfeld,B.,Lauder,P., Personis: A Server for User Models, In Proc. of Adaptive Hypermedia and Adaptive Web-Based System AH2002,pp203-212(2002)

[RDF] Beckett,D.,RDF/SML Syntax Specfication, http://www.w3c.org/TR/rdf-syntax-grammar/

[RDFS] Brickley,D.,Guha,R.V., RDF Vocabulay Description Language 1.0:RDF Schema, http://www.w3.org/TR/rdf-schema/

[OWL] Dean,M., Connolly,D., Harmelen,F., Hendler,J., Horrocks,I., McGuinness,D.,Schneider,P.,Stein,L., OWL Web Ontology Language 1.0 Reference,http://w3.org/TR/owl-ref/