マルチメディア通信と分散処理 108－13
（２００２．６．７）

状況に適合するサービスをモバイル・ユーザに提供するシステム

中村 宏明　　百合山 まどか　　三科 昌司

日本アイ・ビー・エム(株)

モバイル・ユーザに対して嗜好・行動・場所などの状況に適合したサービスを提供することは、ユーザがある時点で属しているグループをダイナミックに計算し、属するグループに対応付けられているサービスを提供することで行える。このようなシステムを実現するための課題と解決方法を検討した。特に(1)マッチング・エンジンの高速化アルゴリズム、(2)マッチング動作の視覚化ツール、(3)ユーザ情報とサービス情報の編集ツールの設計と試作を行い、それらの効果を確認した。

Context-Aware Service Delivery to Mobile Users

Hiroaki Nakamura　Madoka Yuriyama　Shohji Mishina

IBM Japan, Ltd.

We can deliver context-aware services to mobile users by dynamically computing the groups of users and providing the users with the services associated with the groups. We analyzed the problems in realizing such mechanisms and studied the solutions. In particular, we developed (1) an algorithm for optimizing the performance of our matching engine, (2) a tool for visualizing the process of the matching engine, and (3) tools for editing user profiles and service descriptions, and verified the effectiveness of the algorithm and tools.

1. はじめに

モバイル・デバイスやワイヤレス技術の進歩と、さまざまな場所に設置される情報機器の普及によって、ネットワークを介して提供されるサービスに対する要求も変化してきている。ユーザは動き回りながら、いろいろなデバイスから自分に関連の高いサービスにアクセスできることを期待している。したがって、このようなモバイル・ユーザに向けてのサービスを提供するためには、嗜好・行動・場所などの個人的な状況をもとにしてユーザの関心を予測し、その時点でユーザに最も適するサービスを自動的に判断して提供できることが必要になる。

状況に適合したサービスの提供を実現するためには、次のような仕組みがあればよいと考えられる:

1. サービスが対象とするユーザのグループをあらかじめ指定しておく。
2. ユーザがどのグループに属するかを必要な時点で計算する。
3. 対応するサービスをユーザに提供する。

グループは嗜好・行動・場所などユーザの状況を基準として指定しておく。グループには重なりがあってもよいし、基準として場所と趣味などの複数の項目の組み合わせでもよい。ユーザの状況は変化し続けるので、グルーピングはダイナミックに行われる。またサービスを提供する側のグループの指定も時間とともに変化する。このような仕組みを使うことで、次のような利点が得られる:

- 対象とするユーザを的確に指定してサービスを提供することができる。
- ユーザはサービスに関する情報をあらかじめ知らなくても適切なサービスが選択される。

本稿では、このような機能を実現するために我々が提案しているシステムのアーキテクチャを紹介する。またシステムを実用的なものにするための課題と解決方法を議論する。

2. システムの概要

我々は状況に適合したサービスをモバイル・ユーザに提供するための図1に示すようなアーキテクチャを提案

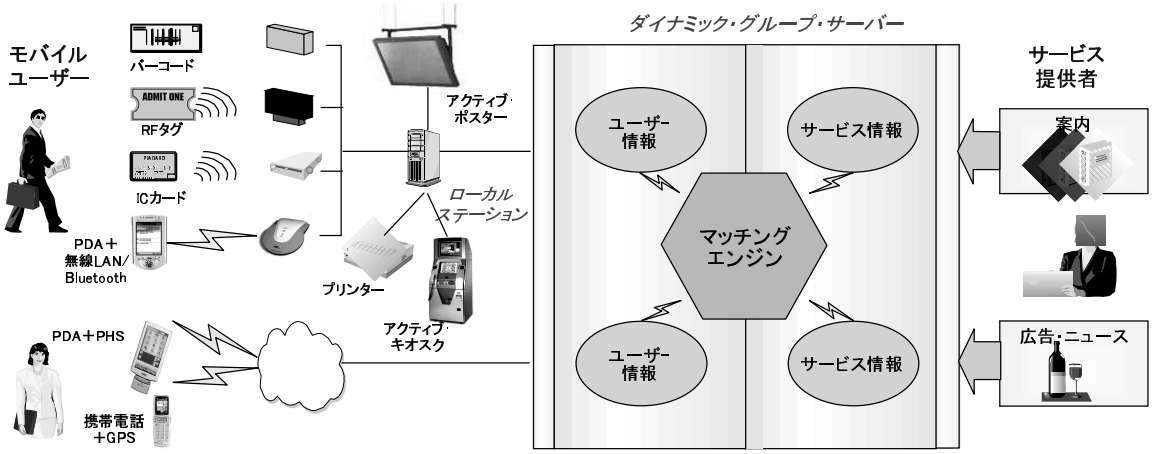

図1. 状況に適合するサービスをモバイル・ユーザに提供するシステム

している[7]。ダイナミック・グループ・サーバはグループの形成に必要な情報としてユーザ情報とサービス情報を管理している。ユーザ情報には嗜好などのようにあらかじめ登録しておくものもあるし、位置や行動などのようにシステムにアクセスする時点で獲得されるものもある。一方サービス情報のうちで状況に適合したサービス提供に必要な部分は、サービスを提供したいユーザのグループに対する指定である。これらの情報の管理には柔軟性と性能の要求を満たすためにマルチエージェント・システムのフレームワーク[5]を使う。

システムはさまざまな場所にある情報機器とともに設置される RF タグや IC カードの読取装置によってモバイル・ユーザを認識する。また PDA などからアクセスを受けることによってもユーザを認識する。このときにユーザの識別子に加えて場所・時間・行動などの情報が獲得される。サーバはユーザ情報とサービス情報のマッチングを行うことによって、そのユーザの属するグループを求めて、ユーザの状況に応じた最も適切なサービスを提供する。

図2はこのアーキテクチャに基づいて試作した、イベント会場内で案内サービスを行うシステムの画面表示例である。ユーザにはRFタグが埋め込まれたチケットが配られている。チケットを発行するときに得たアンケート結果と事前登録したセミナーに関するデータから、ユーザ情報を事前に登録しておく。サービスはシアターでの講演とブースでの展示物の案内である。個々の講演は興味と時間の組を指定することによって対象ユーザのグループを形成する。また個々の展示物は興味と場所の組を指定することによって対象ユーザのグループを形成する。あるユーザが会場内に設置された情報端末に RF タグを近づけると、時間・場所・興味にしたがって講演と展示物の案内が選択され、表示される。

図2. イベント会場内の案内サービスの画面表示例

3. 課題

このようなアーキテクチャを実用的なものにするためには、つぎのような課題を解決しなければならない。

(1)まず、マッチング・エンジンが高速に動作する必要がある。サービス数が増えると検査すべきグループ数が増える。しかしサービス数が増えてもレスポンス時間をなるべく低下させずにサービスを提供してほしい。
(2)つぎにマッチング動作をモニターできる必要がある。システムの外側からは、あるユーザとあるサービスが結び付けられたことは分かるが、どのようなグルーピングが行われた結果なのかわからない。意図するグルーピングが行われているかを調べる仕組みが必要である。
(3)さらに、ユーザ情報とサービス情報の追加や変更を容易に行うための仕組みが必要である。情報の項目のバリエーションは応用分野によって異なるので、項目の変化に柔軟に対応できることが重要になる。
4節、5節、6節でこれらの課題を解決する方法を議論する。

4. マッチング・エンジンの高速化

4.1. マッチングの表現

我々が提案するアーキテクチャでは、様々なユーザの状況を統一的に表現するために Extensible Markup Language (XML) を用いる。またサービスの対象となるユーザが属するグループを指定するために XML Path Language (XPath) [1] を用いる。あるユーザがあるグループに属するかどうかは、ユーザの状況を表現する XML 文書に対して、サービス提供者が指定したグループを示す XPath 式を評価することによって判断することができる。XPath は本来 XML 文書の部分を指し示すための言語であるが、文脈に応じて結果をブール値や数値として解釈することができる。値が真であればそのグループに属し、偽であればグループに属さない。数値を返すXPath式をメンバーシップ関数とみなすことでグループをファジー集合として扱うこともできる。

4.2. アルゴリズムの概略

このような枠組みではサービス数が増加すると評価しなければならない XPath 式の数も増加する。単純に XML 文書と XPath 式の組み合わせを1つづつ評価していくと、多数の XPath 式がある場合にレスポンス時間がX Path式の数に比例して悪化するという問題がある。

我々は XPath 式が多数ある場合でもレスポンス時間の増加が XPath 式の数に比例しないように抑えるアルゴリズムを開発し、この問題を解決した[6]。基本的なアイデアは以下のとおりである。XPath 式が複数ある場合、それらの XPath 式は対象となる XML 文書の構造や要素の値のバリエーションなどから制約を受けるので、複数の XPath 式の中に類似のものが含まれる。類似する XPath 式から共通の部分を取り出して評価し、それを含む XPath 式の間で結果を共有することで XPath 式を個別に評価するよりも高速に実行できる。XPath 式の数が増えるにしたがって共通部分が増えるので、我々のアルゴリズムは有効に働くようになる。評価結果を共用するものとして、次の言語要素を対象とした。

- ロケーション・パスの共通ステップ
- 述部の式
- 演算子や式の引数

4.3. 実験結果

次の3つの場合のパフォーマンスを測定し、本アルゴリズムの有効性を検証した。

1. 本アルゴリズムを使用しないで、XPath 式を個別に評価した場合
2. ロケーション・パスにおける共通ステップの評価だけを高速化した場合
3. 共通ステップの評価と述部の評価を高速化した場合

実験環境は、IBM ThinkPad T21 2647-8AJ(CPU:モバイル Pentium III-800 MHz、メモリ:384 MB、OS:Windows 2000)である。XML 文書の DTD には、CPExchange [3]を使用し、XML 文書は IBM の XML Generator [2]を用いて生成した。XML Generator は与えられた DTD とパラメーターからランダムに XML 文書を生成することができる。XPath 式も DTD と入力パラメーターからランダムに生成した。XPath 式(本アルゴリズムを用いて分解した個々のステップも含む)の評価には、Apache Xalan-Java 2 [4]に含まれる XPath プロセッサのパッケージ org.apache.xpath を使用した。

評価する XPath 式の数を増やしながら、評価時間を測定した。XPath 式の深さは一定の深さ5で、各 XPath 式は最後のステップに述部が存在した。図3が実験結果である。当初の予測通り、本アルゴリズムを使用しない手法では一番パフォーマンスが悪かった。ロケーション・パスにおける共通ステップの評価の共有のみの手法では、若干改良が見られるものの、XPath 式の数の

増加に比例して時間が増加した。一方、提案する全てのアルゴリズムを使用する手法は一番パフォーマンスが良く、XPath 式が増加しても直線的に評価時間が増えなかった。

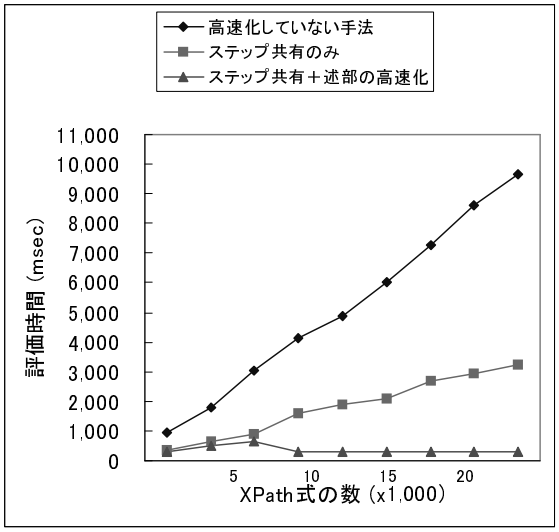

図3. 実験結果

5. マッチング動作の視覚化ツール

マッチング・エンジンは、あるユーザがあるグループに対して属するかどうかの検査を、多数のグループを対象として行う。また各グループのユーザに対する条件は多様である。そのためマッチングの動作は非常に複雑になる。本研究では、複雑なマッチングの動作をモニターすることを容易にするために、マッチング動作をリアルタイムで視覚表現するためのツールを開発した。

このツールはマッチング・エンジンのもつデータと状態をグラフで表現する。マッチング・エンジンのデータや状態が変化するときに発生するイベントを捕まえ、それによってノードやアークの追加・削除や色の変更を行うことによって、マッチング動作を視覚的に表現する。

データ構造は XPath 式のロケーション・パスを分解した結果のステップ、述部の高速化のためのハッシュテーブルや二分探索木の項目、演算子や式、XPath 式自身を要素として持つ。それぞれの要素をノードとして表し、要素間をアークで結ぶ（ステップとそれに続く次レベルのステップ間など）ことにより、データ構造をノードとアークを用いて視覚化することが可能になる。XPath 式が追加・削除される場合、ステップなどがデータ構造に追加・削除されるので、それを視覚化されたデータ構造に対してもノードやアークの追加・削除として同期をとって反映させる。前節のアルゴリズムを用いたマッチング・エンジンを実行すると、データ構造の各要素に対する評価値がステップ・述部に対応するハッシュテーブルや二分探索木の項目、演算子や式・XPath 式自身の順に得られる。こうして順々に得られる評価値を即座に視覚化されたデータ構造に対して反映させることにより、マッチング・エンジンの動作状況をリアルタイムに視覚的に表現する。

50個の XPath 式をマッチング・エンジンに登録して、視覚化されたデータ構造を表示しているウインドウを図4に示す。ある XML 文書が与えられると、視覚化ツールはマッチング・エンジンの処理の進行にしたがってノードとアークの色の種類と濃淡を変更することによって、最終的な結果が得られるまでのプロセスを示す。このツールを実際に使うことによって、複雑なマッチングの動作のモニターに有効であることを確認した。

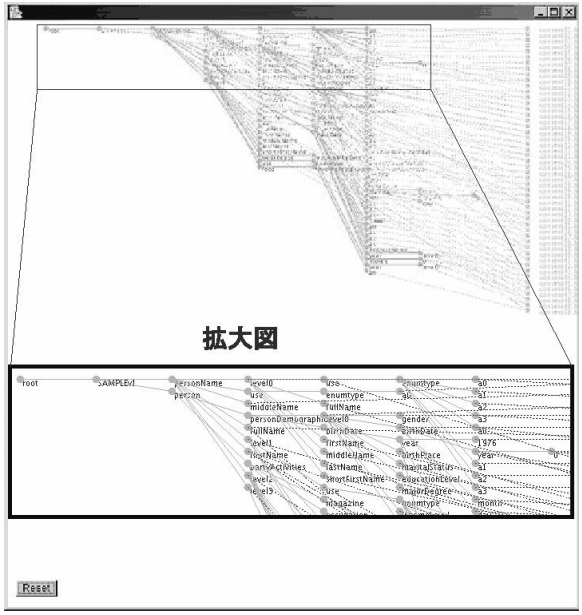

図5. マッチング動作の視覚化ツールの表示例

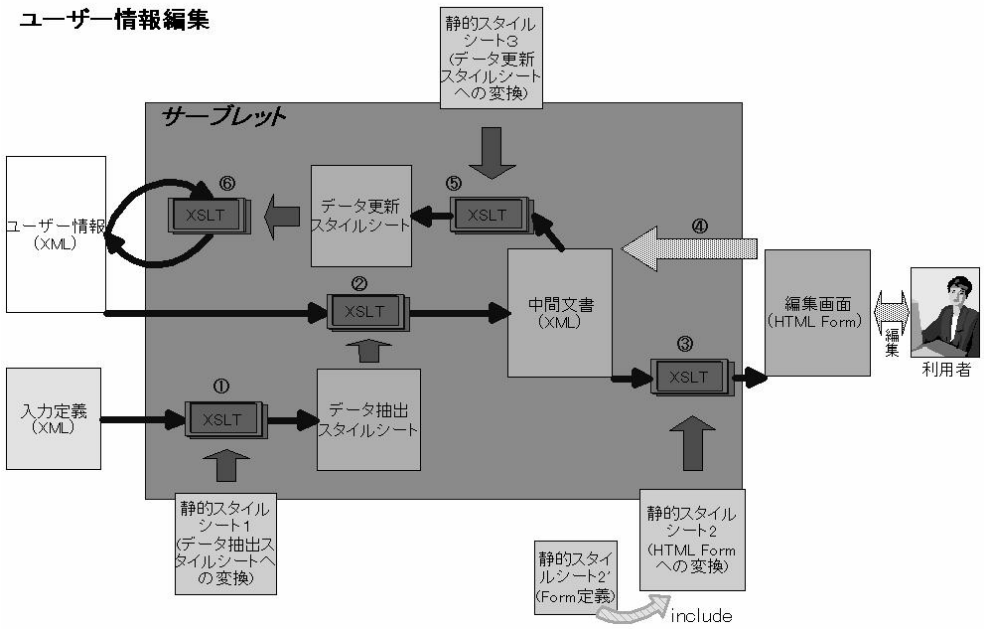

図6. ユーザ情報編集ツールの構造

6. データ編集ツール

ユーザ情報のうちでスタティックな部分（たとえば嗜好情報）と、サービス提供者が指定するグループの情報は、個々のユーザやサービス提供者などの一般利用者がシステムに与える性質のものである。したがってこれらのデータは XML や XPath を直接入力するのでなく、利用者が使い慣れたインタフェースで、また特別な用意を必要としない環境で編集できることが望まれる。さらに応用分野によってデータ項目の種類や取り得る値の範囲が異なるため、このツールはデータ項目の多様性に柔軟に対処できることが望まれる。

以上の要求を満たすため、Web ブラウザを使ってデータの追加・変更・削除が行えるように、サーブレットのアプリケーションとしてデータの編集ツールを開発した。また編集の対象とする箇所(XML の要素)、その要素の取り得る値、編集画面上での表示の仕方等をダイナミックに変化させられるよう、図６のような仕組みを構築した。基本的なアイデアは応用分野によって変化する部分をすべて外部入力定義ファイルで記述し、これを基にしてユーザ情報を記述する XML 文書から特定の部分を取り出したり、ユーザの編集画面の HTML ファイルを生成したりすることである。処理の流れは次の通りである：

① 入力定義(XML で記述)から、データ抽出スタイルシートを得る。
② このスタイルシートを使って、ユーザ情報から指定された値を抽出し、中間文書(XML)を得る。
③ 中間文書には入力定義から得たフォームの指定などが埋め込まれているので、これを使って編集画面(HTML)を生成する。
④ ユーザが画面に入力した値がサーバに返り、これを使って中間文書を更新する。
⑤ 更新された中間文書を使って、データ更新スタイルシートを得る。
⑥ このスタイルシートをユーザ情報に対して起用することで、ユーザ情報を更新する。

図７と図８に本ツールを使ったユーザ情報編集の実行例を示す。入力定義ファイルを変更することで、編集項目やフォームの入力方法などをダイナミックに変化させることができる。入力定義ファイルをデバイスの種類などにしたがって複数個用意しておくことで、データ編集の方法自体も状況に対応させて変えることができるようになる。

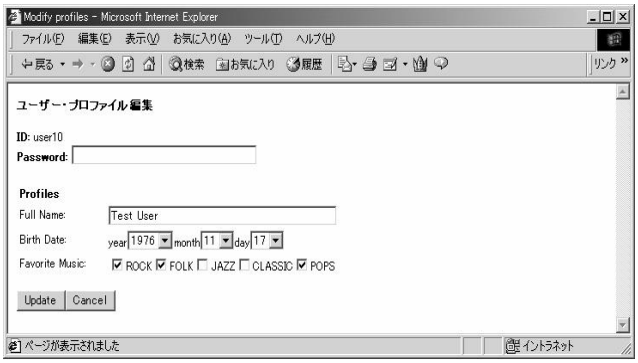

図７．ユーザ情報編集の実行例１

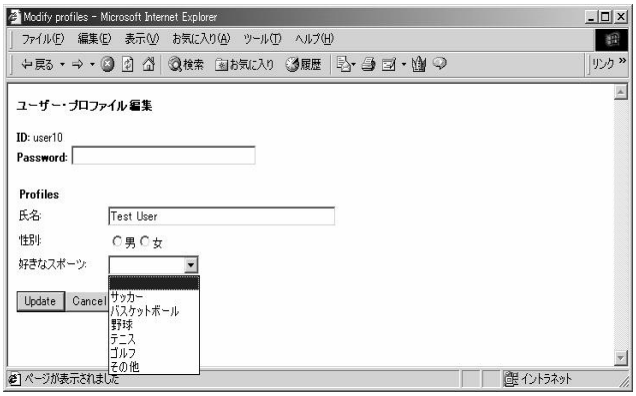

図８．ユーザ情報編集の実行例２

7. おわりに

本稿では、モバイル・ユーザに対して嗜好・行動・場所などの状況に適合したサービスを提供するために我々が開発しているシステムを紹介し、またこのシステムを実用的なものにするための性能に関する問題と、使いやすさに関する問題を解決する方法について述べた。

ダイナミック・グルーピングのためのアーキテクチャを実用的なものにするためには、本稿では扱わなかった項目についても検討が必要である。ユーザ情報のうちで静的なものは編集ツールを使って入力することを前提としていたが、たとえば行動履歴からユーザ情報を抽出するように、自動的にユーザ情報を構成する仕組みも重要である。また位置と嗜好など異なる種類のユーザ情報を同時に扱うときに、それらに重みをつけるのを現在は人手によって行っている。重みをつけるのを自動的にサポートする仕組みも必要になるであろう。

謝辞

本研究を進めるにあたって、森本典繁氏、宮澤達夫氏、本田良司氏、鈴木和洋氏、金來氏、中村大賀氏との議論が非常に有益でした。またシステムの設計と試作を共同で行いました。感謝致します。

参考文献

[1] J. Clark and S. DeRose, “XML Path Language (XPath): Version 1.0”, 1999. http://www.w3.org/TR/xpath.html

[2] L. Diaz and D. Lovell, “XML Generator”, 1999. http://www.alphaworks.ibm.com/tech/xmlgenerator

[3] IDEAlliance, “CPExchange Specification Version 1.0”, 2000. http://www.cpexchange.org.

[4] The Apache XML Project, “Xalan-Java Version 2.0”, 2001. http://xml.apache.org/xalan-j/index.html

[5] G. Yamamoto, H. Tai, "Agent Server Technology for Next Generation of Web Applications", 4th International Conference on Computational Intelligence and Multimedia Applications, IEEE Computer Society Press, 2001.

[6] M. Yuriyama, H. Nakamura, "`Efficient Algorithm for Evaluating Multiple XPath Expressions", IBM Research Report, RT0445, 2002.

[7] 日本アイ・ビー・エム(株), "モバイル・リソース・マネージメント",2002. http://www.trl.ibm.com/projects/mrm/index.htm