

移動体計算機環境における位置依存情報提供システムの設計と実現

寺 西 裕 一[†] 種 茂 文 之[†]
梅 本 佳 宏[†] 寺 中 勝 美^{††}

移動体計算機環境下でのアプリケーションとして、利用者の位置や時間等に応じて適切な内容の情報提供を行う位置依存情報サービスが考えられる。位置依存情報サービスでは、位置や時間等の利用者の状態に応じた情報や機能の提供、および、移動等の状態変化にともなうそれらの更新が要求される。本稿では両者を実現するための新たな枠組みとして、情報および機能を合わせたオブジェクトを位置依存情報サービスの管理・提供の単位とするモデルを提案する。本モデルに基づく情報提供システムでは、利用者から要求があると、オブジェクトを利用者の位置や時間等の状況に応じて各携帯端末上に転送し、再構成・実行することによって位置依存情報サービスを実現する。これにより、場所ごとに異なる案内機能を提供する等の、従来のシステムでは扱えなかった高度な位置依存情報サービスを構築することが可能となる。また、本モデルを移動体計算機環境下で実現するための新たなアーキテクチャについてもあわせて提案する。このアーキテクチャは、オブジェクトを再構成させる機構を固定ネットワーク側に利用者ごとに置くことにより、再構成にかかる通信回数を減らして実行効率を上げるとともに、携帯端末に必要となる処理の負荷を軽減させることができる。本稿ではさらに、提案手法の有効性を確認するために WWW 上に構築したプロトタイプシステムおよびその評価についても述べる。

Design and Implementation of Location-dependent Information System on Mobile Environment

YUUCHI TERANISHI,[†] FUMIYUKI TANEMO,[†] YOSHIHIRO UMEMOTO[†]
and KATSUMI TERANAKA^{††}

Location-dependent information service requires an ability to change both the information and functions provided to the mobile client according to user location and time of access. This paper describes a new framework for location-dependent information service, called the mobile object model. In mobile object model, a service consists of objects that contain both information and functions. These objects, called mobile objects, are managed as distributed objects. Mobile objects are recomposed to a composite object according to user status dynamically. This composite object is then executed to realize location-dependent information service. We also propose an architecture based on the model. The architecture reduces the load of network transmission and the amount of CPU power used by the mobile client. Finally, we present a prototype system developed on the WWW.

1. はじめに

近年の計算機の小型化や移動通信技術の発達にともない、利用者が任意の場所から計算機を通じてネットワーク上のリソースを利用できる移動体計算機環境が

実現可能となりつつある。この移動体計算機環境におけるアプリケーションの1つとして、位置依存情報サービスと呼ばれるものがある。

位置依存情報サービスの内容としては、たとえば次にあげるものが考えられる¹⁾。

● 最寄り検索

現在、利用者に最も近い対象物を探す。たとえば、病院を検索したのであれば、位置や営業時間を考慮して適切なものから順に検索結果を表示する。このとき、移動することで病院までの距離が変わったり、満員となってしまったりする場合には、

[†] NTT 情報通信研究所
NTT Information and Communication Systems Laboratories

^{††} NTT 研究開発推進部
NTT Research and Development Management Department

随時結果を更新する。

- ナビゲーション

人が携帯端末を持ち歩き、サービス端末として用いて実際の行動の支援に利用する。この場合、駅では電車の発車時刻等の駅案内を、空港では航空機の発着案内を行う等、場所や利用方法に応じて適切な内容・案内方法に切り替える。

- 地域依存型エンターテインメント

展示会やアミューズメントパークにおいて、その場にある展示物やアトラクションの設備を利用者の手元の携帯端末から操作する等、利用場所にあるシステムと連携したサービスを行う。

まず、最寄り検索を行うには、利用者の位置等の状況を検出し、状況に応じて検索結果を提示できることが要求される。さらに、通常の実検索とは異なり、利用者の位置等の状況が変化することに追従して、検索結果が逐次的に更新されることが要求される。この検索をここでは継続的検索と呼ぶ。

また、先に述べたナビゲーションや、地域依存型エンターテインメントといったサービスを実現するには、空港では飛行機の発着場所案内を行う、展示会場ではその場にある展示物との間で独自の通信を行う等のように、各携帯端末上のアプリケーション自身の機能を位置等の状況に応じて追加・変更可能とする必要がある。こうした利用者の状況に応じた機能の提供は、単に文字や画像といった静的な情報を表示するだけでは実現できない。この利用者の位置等の状況に合わせて提供されるアプリケーションの機能を、ここでは位置依存機能と呼ぶ。

位置依存情報サービスを扱った研究はこれまでもいくつか存在する。たとえば、DATAMAN^{4),5)}、Mobisaic¹⁰⁾等ではHTML文書を利用者の位置変化に応じて継続的に携帯端末に提供する方法が実現されている。また、AMDS³⁾では、変化する移動体計算機環境の状況に追従する情報をデータベースのビューとして定義し、継続的に情報を管理する方法が実現されている。すなわち、これらのシステムは、継続的検索を実現しているといえる。しかし、これらのシステムは画像や文字列といった静的な情報をサービスの単位として扱っており、先に述べたような位置依存機能は実現できていない。

TeleportX⁶⁾では、任意の機能を持つアプリケーションを、ユーザがいる位置に最も近い場所にある固定端末上に呼び出す方法が実現されている。したがって、TeleportXは位置依存機能を実現している。しかし、TeleportXは利用者が移動しながらアプリケーション

を利用することを前提としておらず、継続的検索を扱えない。

本稿では、継続的検索と位置依存機能の両者を持つ位置依存情報サービスを実現するための新たな枠組み、モバイルオブジェクトモデルを提案する。モバイルオブジェクトモデルでは、提供する情報と機能を統合的にオブジェクトとして管理する。利用者からサービスの要求があると、各オブジェクトを利用者の位置・時間等の状況に応じて各携帯端末上に転送し、再構成・実行することによって位置依存情報サービスを実現する。本稿ではさらに、提案モデルを移動体計算機環境で効率良く実現するためのアーキテクチャについてもあわせて提案する。

2章では、我々が提案するモバイルオブジェクトモデルについて述べる。3章では、提案モデルを移動体計算機環境で効率良く実現するために我々の提案するアーキテクチャについて述べる。4章では、本論文で提案する方式の有効性を確認するために構築したプロトタイプシステムの実現法、およびその評価について述べ、5章でまとめる。

2. モバイルオブジェクトモデル

本章では位置依存情報サービスを実現するために我々が提案するモバイルオブジェクトモデルについて述べる。まず、位置依存情報サービスを実現するためのモデルとしてこれまでに提唱されている、動的文書について説明する。モバイルオブジェクトは、基本的に動的文書の概念を拡張したものである。

2.1 動的文書

動的文書の概念は、文献5), 8), 10)等で提唱され、実現されている。

動的文書では、文書にアクセスすることがサービスにアクセスすることに対応している。1つのサービスは状況によって切り替わる複数の文書によって構成される。各文書は、スコープと呼ばれる位置、および、時間の制約条件とともに管理される。各文書の内容は登録されたスコープのもとで有効となる。

利用者が手元の携帯端末からサービスにアクセスすると、そのときの利用者の位置や時刻に基づいて各スコープが評価され、内容が有効となる文書が携帯端末に転送されて、表示される。また、利用者の移動等によって利用者の状況が変化すると、変化後の条件でスコープが再評価され、得られる文書が提示中の文書内容と切り替わる(図1)。

以上の仕組みにより、動的文書では、情報提供者が文書の内容に合わせてスコープを設定しておけば、利

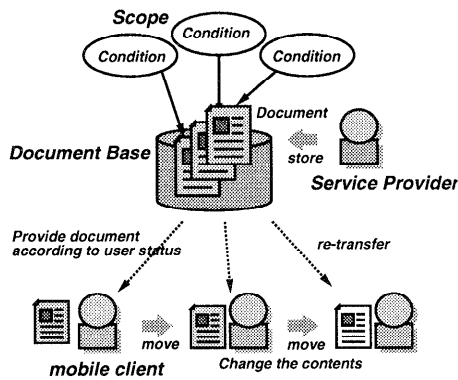


図 1 動的文書

Fig. 1 Dynamic document model.

ユーザーが検索して得た文書は、状況に合わせてつねに適切な文書に切り替わる。すなわち、動的文書は位置依存情報サービスに要求される継続的検索を実現しているといえる。

2.2 モバイルオブジェクトモデルの基本概念

動的文書では、サービスの管理提供の単位は静的な画像、文字等を含んだ文書である。しかし、静的な文書を単位としたモデルでは、各携帯端末上のアプリケーション自身の機能を状況に応じて追加・変更する位置依存機能までも扱うことはできない。

本稿では、動的文書モデルに基づき、かつ、情報および機能を合わせたオブジェクトをサービス提供の単位として位置依存情報サービスを行う、モバイルオブジェクトモデルを提案する。モバイルオブジェクトモデルでは、利用者の状況に応じた情報および機能を持つオブジェクトを各携帯端末上で動作させることによって、位置依存情報サービスを実現する。

提供者は動的文書と同様に、オブジェクトごとにその内容が有効となるスコープを登録しておく。利用者がサービスにアクセスすると、利用者の状態に基づいて各スコープの評価が行われ、当てはまるスコープを持つ複数のオブジェクトが携帯端末上に転送されて、再構成される。再構成されたオブジェクトは個々のオブジェクトが持つ機能が組み合わせられた固有の機能を持ったものとなる。このオブジェクトが携帯端末上で実行され、動作することによって、位置依存情報サービスが実現される。

利用者の移動等とともに利用場所や時間が変化すると、変化後の条件で各スコープが再度評価され、利用者端末上のオブジェクトも再構成されて、適切な内容のサービスに切り替わる（図 2）。

以上の仕組みにより、モバイルオブジェクトモデ

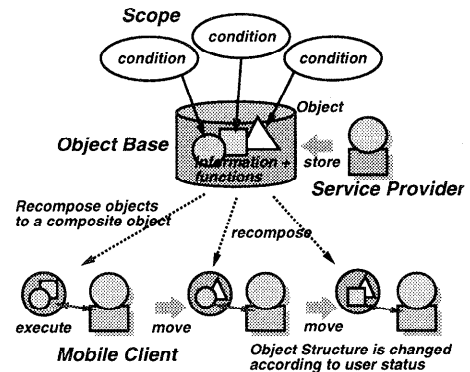


図 2 モバイルオブジェクトモデル

Fig. 2 Mobile object model.

ルでは、サービスの提供者が、情報表示の機能を持つオブジェクトに適切なスコープを設定しておけば、移動等の状況の変化に合わせてつねに適切な情報を表示させることができる。これによって、動的文書と同様の継続的検索が実現できる。

また、再構成されたオブジェクトは利用者の状況に固有の機能を持つオブジェクトとなり、これが移動端末上で実行されることによって機能を含めた情報の入れ換えが端末上に起こる。すなわち、モバイルオブジェクトモデルは動的文書と同様に継続的検索を実現するとともに、位置依存機能を実現する枠組みをも提供している。

モバイルオブジェクトモデルに基づく位置依存情報サービスの利用者は、状況に合わせてつねに適切な機能と情報を利用可能となる。利用者ごとに動的にオブジェクトを再構成させることで、個人ごとに異なったナビゲーションや、行動支援を提供するといった高度なサービスにも対応できる。

2.3 モバイルオブジェクトモデルにおけるスコープ

モバイルオブジェクトモデルでは、単一の文書ではない、機能を持った複数のオブジェクトの再構成を扱うために、動的文書のスコープの概念を拡張している。以下ではこのスコープの拡張点について述べる。

2.3.1 オブジェクトの結合性

複数のオブジェクトを再構成させて実行するには、結合させるオブジェクト間で互いにデータのやり取り等を行うための一定の取り決めが存在することが必要となる。このためモバイルオブジェクトモデルでは、スコープに基づく検索において、オブジェクト間に結合性に関する制約（インタフェースの存在等）が満たされるかどうかについても検査する。提供者はスコープとして、動的文書と同様の位置、時間に関する制約

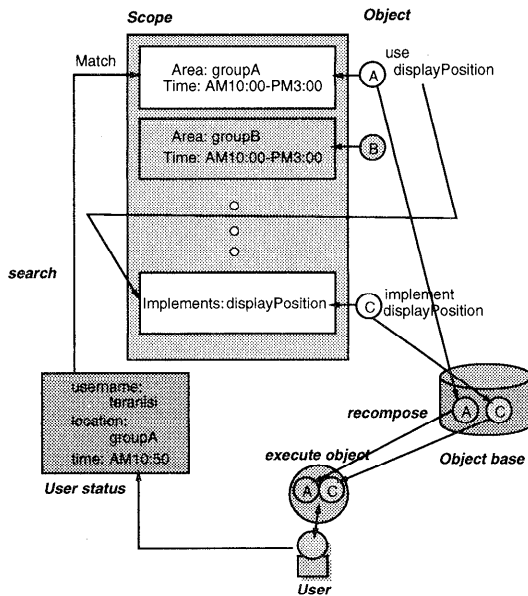


図3 スコープに基づくオブジェクトの再構成
Fig. 3 Recomposition according to the scope.

のほかに、この結合性に関する制約をオブジェクトごとに設定する必要がある。

図3はモバイルオブジェクトモデルにおいて、スコープに基づいて再構成が行われる一例を示している。図3中のオブジェクトAは実行中に、`displayPosition`（「位置表示」）というメッセージを他のオブジェクトに通知するオブジェクトである。Cは`displayPosition`のメッセージを受け取り、「位置表示」という動作を実装（Implement）しているオブジェクトである。オブジェクトAが動作するための必要条件は、オブジェクトCのようにメッセージ`displayPosition`を解釈し実行できるオブジェクトが存在することである。したがって、例では、Aのスコープが利用者状態に合致することが判明すると、`displayPosition`を実装しているオブジェクトを検索し、Cを得て再構成がなされている。

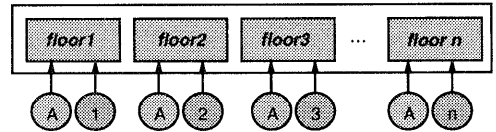
2.3.2 スコープの階層化

モバイルオブジェクトモデルによる位置依存情報サービスでは、複数の状態でオブジェクトを共有することが考えられる。

たとえば、あるデパートの情報案内において、売場ごとに商品紹介を行うオブジェクトを用意し、デパート全体で全体のイベント案内を行うオブジェクトを利用することが考えられる。

このとき、売場ごとにスコープを設けたとすると、同じデパート全体の案内オブジェクトをすべての売場

(a) Scope (No hierarchy)



(b) Hierarchical Scope

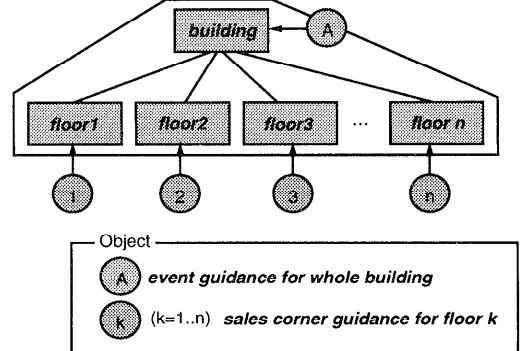


図4 スコープの階層化
Fig. 4 Hierarchical scope.

に用意し、それぞれに各売場のスコープを登録しなければならない（図4中の(a)）。

そこでこうした冗長なオブジェクトの定義を削減するために、モバイルオブジェクトモデルでは、スコープを階層的に定義可能とする。すなわち、複数の状態に共通のオブジェクトは、上位のスコープに対応づけし、上位のスコープに対応づけられたオブジェクトはすべて、下位のスコープでも有効になるものとする（図4中の(b)）。

これにより、複数の状態で利用するオブジェクトを何度も定義する必要がなくなり、オブジェクト定義の簡略化が可能となる。

3. モバイルオブジェクトモデルのアーキテクチャ

本章では、まず、2章で述べたモバイルオブジェクトモデルを実現するために必要な機能要素を提示する。次に、サービスを分散環境で行う場合に考慮すべき点について議論し、それらを踏まえたシステムの構成、および実行制御法を提案する。

3.1 モバイルオブジェクトの機能要素

図5は、モバイルオブジェクトモデルを実現するうえで必要となる機能要素を示している。

図5において、オブジェクト蓄積機構は、サービスを構成するオブジェクトを格納するためのものである。利用者状態監視機構は、GPS等の手段で検知される利用者の位置等の利用者状態を収集し、監視する。オブジェクト再構成機構では、モバイルオブジェク

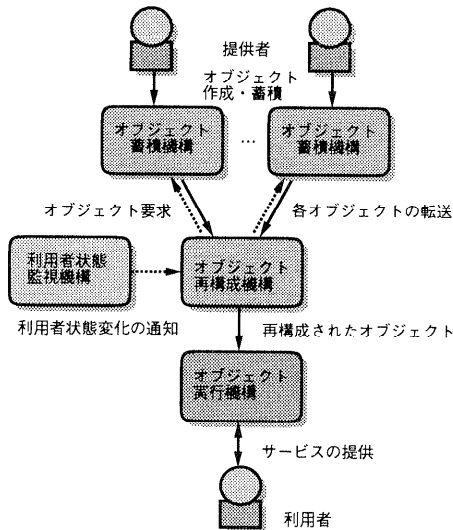


図5 モーバイルオブジェクト実現に必要な機能要素
Fig.5 Required modules.

トのスコープを管理し、利用者状態監視機構で得られる利用者の条件に基づいてオブジェクトを再構成させる。オブジェクト実行機構は、再構成されたオブジェクトを実行して利用者端末上に実際のサービスを提供する部分である。これらの機能要素を固定ネットワーク、携帯端末からなるアーキテクチャにおいてどこに配置するべきかが問題となる。

提供者がオブジェクトを容易に追加・更新するためには、オブジェクト蓄積機構は固定ネットワーク側にあることが望ましい。また、複数の提供者によってサービスが管理されることを考慮すると、この機構は分散して存在するのが自然である。したがって、ここではオブジェクト蓄積機構は固定ネットワーク側に分散して存在するものとする。また、利用者状態監視機構は、全利用者の位置等の状態を収集・管理するために、固定ネットワーク側にあるものとする。

移動体計算機環境では、利用者が持ち歩く携帯端末が固定端末ほどの処理能力を持つことは困難であり、端末の負荷をなるべく小さくすることが重要となる。また、位置依存情報サービスでは移動しながら無線通信を行う必要があるが、一般に無線による通信容量は有線による通信に比べ容量に制限があり、なるべく通信回数、転送量を削減できるようにシステムを構成する配慮が必要である。

3.2 機能要素の配置

オブジェクト実行機構が固定ネットワーク上にある場合、ユーザとシステムの間でインタラクションや、グラフィック表示等の入出力が生じるたびにネットワー

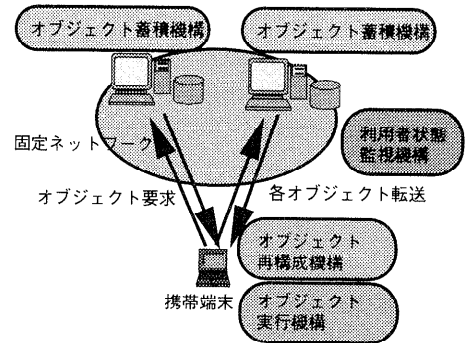


図6 端末上に再構成機構があるシステム構成
Fig.6 Object recomposer is located in the mobile client.

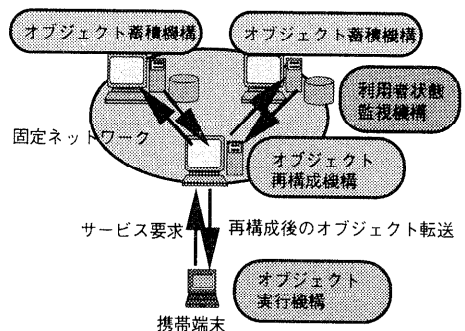


図7 再構成機構が固定ネットワーク上にあるシステム構成
Fig.7 Object recomposer is located in the fixed network.

クを介した通信が必要となってしまう。したがって、ここではオブジェクト実行機構が固定ネットワーク上にあることは前提とせず、オブジェクト実行機構は携帯端末上に置くこととする。

まず、携帯端末上にオブジェクト再構成機構とオブジェクト実行機構の両方を置くことが考えられる。すなわち、分散管理されたオブジェクトを単に携帯端末上に集めて実行する形態である(図6)。この場合、アーキテクチャは通常のサーバクライアント型の単純なものとなる。しかし、通信回数は集めるオブジェクトの個数と同じだけ必要となってしまうため、多くのオブジェクトを利用するアプリケーションでは不利となる。また、携帯端末側に再構成しながら実行するという複雑な仕組みが必要となり、実行時に携帯端末に大きなCPUパワーが必要となる。

これらの問題を解決すべく我々が提案するアーキテクチャを図7に示す。図6のアーキテクチャとの機能要素の配置の違いは、固定ネットワーク側に仲介役としてオブジェクトの再構成機構が置かれている点である。このシステム構成では、固定ネットワーク側で分散管理されたオブジェクトが一度集められ、再構成されてから無線通信を介して転送される。この場合、

利用するオブジェクトの数にかかわらず、1回の再構成における通信回数は再構成後のオブジェクトの転送1回ですむ。また、携帯端末側では、転送されたオブジェクトの実行のみを制御すればよく、実行時の負荷は低くてすむ。

3.3 オブジェクト転送プロトコル

位置依存情報サービスにおいては、状況の変化に応じて新たに情報や機能を加えるばかりでなく、一部の情報や機能を継続して利用することが考えられる。モバイルオブジェクトモデルでは、スコープを検索して得られる複数のオブジェクトを利用者端末上で結合させるため、現在利用中のオブジェクトを再利用することで機能の継続が可能である。

また、アミューズメント施設のようなクローズドな環境内での情報案内等では、同じ場所に戻ることに等により、何度も同じオブジェクトが使われることが考えられる。この場合、オブジェクトがキャッシュデータとして保持されていれば、通信して再びオブジェクトを得ることなく、再利用することができる。

通信回数、通信量を削減するためには、このように実行を継続するオブジェクトや、キャッシュデータとして保持されるオブジェクトを再利用できる転送・実行制御の仕組みが端末側に必要になる。

この端末側でのオブジェクト再利用を実現し、かつ前節で示した後者のアーキテクチャに適合する実行制御を行うために、我々は差分オブジェクト転送プロトコル(DOTP: Differential Object Transfer Protocol)を提案している。

DOTPでは、オブジェクト再構成機構側で、つねにオブジェクト実行機構中のオブジェクト構成、およびキャッシュ状況を保持しておく。各オブジェクトには、一意に決められた識別子があらかじめ割り当てられているものとする。

状態変化にともなう再構成時には、端末側にオブジェクトがすでに存在している場合には、オブジェクトの実体ではなく、オブジェクト識別子を用いて再構成を行う。すなわち、すでに端末上にあるオブジェクトについてはオブジェクト識別子のみが転送され、新規のオブジェクトのみが実体として転送される。

携帯端末上では、識別子で指定されたオブジェクトと、転送されてきた新規のオブジェクトとを組み合わせ、実行を継続する(図8)。この方法で利用者端末上にオブジェクトを転送、実行制御を行うことによって、無線による通信量を減らすとともに、オブジェクト再構成段階で同じオブジェクトが継続利用される場合にサービスが停止することを回避できる。

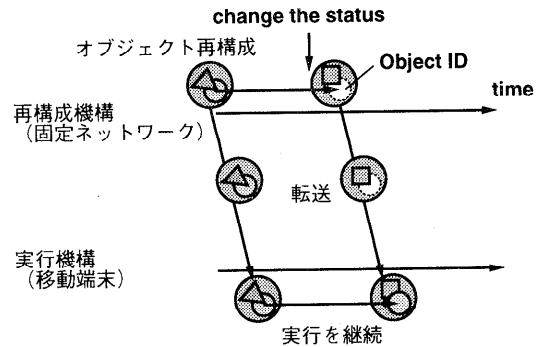


図8 差分オブジェクト転送プロトコル
Fig.8 Differential object transfer protocol.

4. システムの実現

本章では3章で示したアーキテクチャおよび実行制御方式に基づいて構築したプロトタイプシステムについて述べる。プロトタイプではWWWシステム上でJava⁹⁾言語を開発言語としてモバイルオブジェクトモデルを実現している。

4.1 MODS: モバイルオブジェクト記述言語

本システムでは、モバイルオブジェクトモデルに基づくアプリケーションを定義するための記述言語MODSを定義している。

MODSの記述は、大きく分けてスコープ文とオブジェクト文からなる。

スコープ文では、利用者のいる現在位置、他の利用者との位置関係、利用者の行動状態等を利用して制約条件を記述する。

オブジェクト文によって、モバイルオブジェクトを定義する。モバイルオブジェクトが持つ機能は、Javaのクラスとして定義する。オブジェクト文では、このJavaクラス、スコープ、および利用する情報等を記述して、モバイルオブジェクト自身を規定する。

図9にMODSによる記述例を示す。

scope { } の記述部分がスコープ文である(3行~26行)。ここで、利用者の位置や時間等による制約条件を記述してスコープを定義する。現在、MODSのスコープ文で利用可能としている変数の一覧を表1に示す。MODSでは、2章で述べたとおり、スコープは階層的に定義可能である。スコープ文の**is:**フィールドで指定しているのが、階層で上位にあるスコープである。記述例でのスコープの階層関係を図10に示す。

処理系では、利用者状態の変化があると、スコープ階層を下位から順に評価していき、条件にあうスコープを持つオブジェクトすべてを再構成することになる。

object { } の記述部分がオブジェクト文である(27

```

1: # amusement.mods
2: # statements after '#' are comments.
3: scope world {
4:   condition:
5:     [mylocation() == "world"];
6: }
7: scope entrance {
8:   is: world;
9:   condition:
10:    [mylocation() == "entrance"];
11: }
12: scope entrance-expert{
13:   is: entrance;
14:   condition:
15:    [mystatus("level") != "novice"];
16: }
17: scope western {
18:   is: world;
19:   condition:
20:    [mylocation() == "western"];
21: }
22: scope fujiyama-coaster {
23:   is: western;
24:   condition:
25:    [distance("fujiyama") <= 20];
26: }
27: object worldNavigator {
28: # uses interface "displayNavigation";
29: #       and       "setTarget";
30:   scope:      world;
31:   codebase:   "http://server/objs";
32:   code:       "mods.Navigator";
33: }
34: object entranceNavigator {
35:   scope:      entrance;
36:   codebase:   "http://wildboy/objs";
37:   code:       "mods.EasyNavigation";
38:   implements: "displayNavigation";
39:   implements: "setTarget";
40:   parameters: {
41:     mapimg="http://server/exp/enter.gif";
42:     mapname=entrance;
43:     archive="http://server/enter.jar";
44:   }
45: }

```

図9 MODS記述の例

Fig. 9 An example of MODS description.

～45行). **object** に続く文字列がオブジェクト識別子となっている。オブジェクト識別子は自由に決められるが、すでに他のオブジェクトで使用されているものは利用できないようにすることで、識別子の一意性を保証している。各オブジェクトには、スコープ文で定義されたスコープが活性化条件として登録されている。たとえば、27～33行で定義されているオブジェクトでは、スコープ **"world"** が活性化条件として登録されている。すなわち、スコープ階層 **"world"** から下の階層ではすべてこのオブジェクトが活性化されること

表1 スコープで利用できる変数一覧

Table 1 Parameters available for scope definition.

名前	引数	帰値
mylocation	なし	ユーザの現在位置
currenttime	なし	現在時刻
otherlocation	ユーザ名	他人の場所
distance	目標物名	ユーザからの距離
mystatus	状態名	ユーザの状態
otherstatus	ユーザ名, 状態名	他人の状態

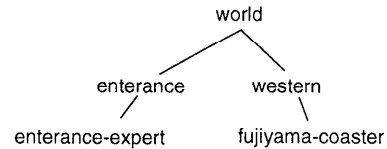


図10 記述例でのスコープの階層

Fig.10 Scope hierarchy in the example.

を意味している。

オブジェクト文の **codebase:**, **code:** フィールドでは、活性化すべき Java クラスの分散環境上での場所、およびクラス名を指定する。 **parameter:** フィールドでは、指定した Java クラスが利用する画像情報等を引数として定義する。たとえば、41 行目では **mods.EasyNavigation** クラスが利用する画像が指定されている。

implements 文は、そのオブジェクトが実装しているインタフェースを規定している。これが2章で示したこのオブジェクトの結合性に関するスコープとなる。

本システムでは、我々が Java 上で開発した **MobileObject** というクラスに、移動、再構成、実行といったモバイルオブジェクトとしての基本機能を持たせている。MODS のオブジェクト文で指定する Java クラスは、この **MobileObject** クラスのサブクラスである必要がある。

基本的に **MobileObject** が用意しているいくつかのメソッドをオーバライドしたクラスを規定することによって、モバイルオブジェクトの機能を定義できるようになっている。表2は **MobileObject** が用意しているメソッドの一部を示している。

4.2 Javaによるモバイルオブジェクト処理系の実現

実現したシステムのモジュール構成を図11に示す。携帯端末側では、オブジェクトの実行機構として通常のWWWブラウザのJava実行機構を利用している。システムの動作のために特に特殊な設定等は必要なく、Netscape等のJavaが動作する環境さえあれば実行可能である。

本システムでは、分散オブジェクトの枠組みとして

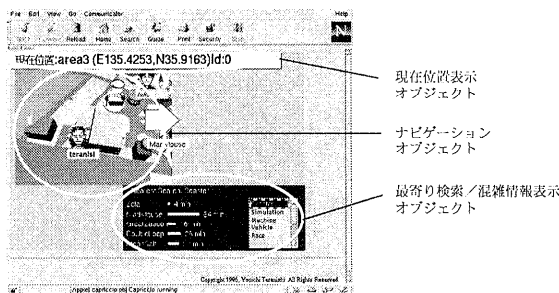


図 14 ナビゲーション

Fig. 14 Navigation.

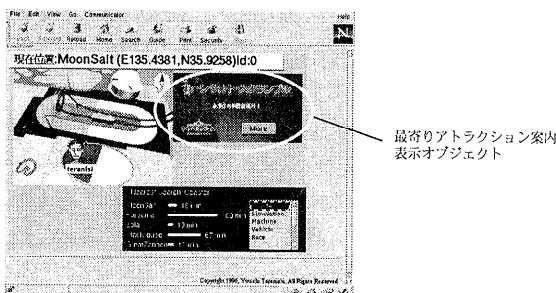


図 15 最寄りアトラクション検索

Fig. 15 Nearest Attraction Search.

テーマランド案内サービスを実行している様子を示している。

図 14 では、テーマランド内のアトラクションへの道案内を行うナビゲーションオブジェクト、アトラクションの最寄り検索/混雑状況表示オブジェクト、および、ユーザの現在の座標を表示する現在位置表示オブジェクトの3つがスコープの検索の結果組み合わせられて実行されている状態を示している。図中の混雑状況表示のオブジェクトは、混雑状況を管理するサーバとの間で独自のプロトコルで通信を行っている。この通信機能はテーマランド内でのみ移動端末上に実現される位置依存機能である。

図 15 は、継続的検索が実現されている例である。図は、最寄りアトラクションの検索により、ハイパーテキストによる案内情報が表示されている状態である。ユーザが別のアトラクションのそばに近付くと、オブジェクトが入れ替わり、案内表示が対応した内容に切り替わる。

図 16 は、図 15 のハイパーリンクを利用者がたどり、アトラクションの詳細情報が表示された様子を示している。

今回試作したテーマランド案内アプリケーションでは、1つのテーマランドの案内を行うための57個のオブジェクトを用意した。記述量は、Java オブジェク

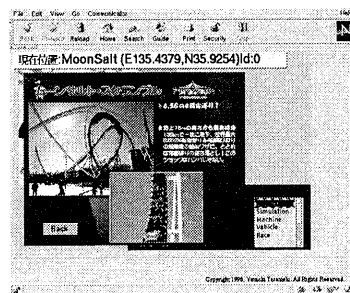


図 16 案内詳細情報の提示

Fig. 16 Displaying detailed guidance information.

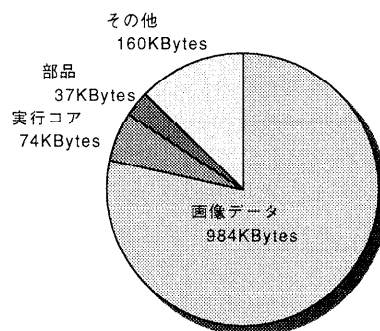


図 17 利用目的別転送バイト数/割合 (1時間)

Fig. 17 Transmitted data per hour.

トとしての記述が 900 行程度、MODS の記述が 640 行程度となった。

案内の実行においては、スコープの評価、検索、再構成、転送、実行の切替えという一連の処理が前記のシステム構成上で実行されるが、オブジェクト切替えはキャッシュ上にデータがある状態であれば 1 秒もかからずに行われ、3.2 節で示した機能要素の配置によれば携帯型パソコンの上でも実用上問題ない程度の実行速度が得られることが分かった。

図 17 は、本アプリケーションを 1 時間実行したときに、通信で占められたバイト数の利用目的別の割合を示している。

図 17 中の「実行コア」は主に Applet として Java の実行環境がサーバから HTTP プロトコルで端末側に送るデータである。すなわちこれらのほとんどのデータはアプリケーション起動時に送られる。図の「部品」はサービスを実現しているモバイルオブジェクトのクラスである。「その他」に含まれているのは、定期的にサーバから通知するようにした混雑情報等のアプリケーションが独自に用いたデータである。

図に示されているとおり、通信のほとんどを占めているのは画像データ (GIF イメージ) であり、再構成において転送される『部品』にあたるオブジェクト

の転送量は比較的少なくなっている。これは差分オブジェクト転送方式を用いていることにより重複するオブジェクトが転送されないためである。

画像データは1時間の間にサーバから30個のデータが送られた。この中には最大で170 KBytesの画像データが含まれている。これをストレスなく(2秒以内に)転送して切り替えるには、少なくとも1 Mbps程度の通信量が必要となる。

5. ま と め

本稿では、移動体計算機環境でアプリケーションを実現するための新たな方法として、情報と機能をモバイルオブジェクトとして管理し、利用者の位置等に応じて利用者端末上に再構成させる手法を提案した。また、この手法を実現するためのシステム構成法として、再構成部を固定ネットワーク側に置くことで高速に通信可能とする方法を提案した。さらに、提案方法を実装したJavaによるプロトタイプシステムの実現法についても示した。プロトタイプシステムは現在も改良を進めているところである。これまでに、モバイルオブジェクトモデルによる位置依存情報サービスの一連の動作が確認できている。

本システムを構築し、実際にアプリケーションを試作して利用してみた結果、以下の3つの問題点が明らかになった。

第1に、新たなモバイルオブジェクトをサービスに加える場合に、MODS記述とJavaのクラス記述を熟読しなければ、オブジェクト開発者以外にはどのようなオブジェクトが存在するかが分からない。また、どんなオブジェクトが存在するか分かっていても、それらとどの程度連携/再利用ができるかを理解するのは非常に困難である。

第2に、試作したシステムではスコープに登録されているものが必ず利用者に提示されるが、この方法は必ずしも有効ではない。たとえば、同じ人が案内アプリケーションを利用しているときに、同じ場所を通る、もしくは同じ人に出会う等すると、すでに分かっている情報であってもスコープに登録されているために提示されてしまう。これによって他の知りたい情報が隠蔽されてしまう場合があった。

第3に、前章で示したとおり、利用するオブジェクトはすべてサーバ側に存在するため、画像データ等を用いるデータ量が大きいオブジェクトの実行への切替えに、時間がかかってしまうことがある。このため、適切な位置で切り替わらず、多少遅れてから必要なサービスが出現することがあった。

今後は以下の項目について重点的に検討を進めていく予定である。

- モバイルオブジェクトの開発環境、およびオブジェクト再利用方法の検討
- 利用者履歴を獲得し、それに応じてサービスを変化させる等の高度なサービスへの展開
- 位置依存プリフェッチ/キャッシング²⁾等の移動体データベース技術の利用によるデータアクセスの効率化の検討

また、MODS およびその処理系の汎用性、他サービスへの適用性の検証や、複数の位置依存情報サービスを有機的に結合させた複雑なサービスの実現法等の検討も行い、より高度なサービス基盤へと発展させていきたいと考えている。

謝辞 本研究を進めるにあたり、様々なご意見をいただいた4大学研究プロジェクトの皆様にご感謝いたします。特に、塚本昌彦大阪大助教授、西尾章治郎大阪大教授、石田亨京都大教授、田中克己神戸大教授、西田豊明奈良先端大教授には、熱心にご討論いただき、貴重なご助言をいただきました。

参 考 文 献

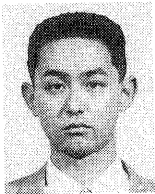
- 1) 寺西裕一, 種茂文之, 梅本佳宏, 寺中勝美: 移動体計算機環境におけるオブジェクトの動的再構成, マルチメディア通信と分散処理ワークショップ論文集, pp.173-178 (1996).
- 2) 佐藤健哉, 左近 透: カーナビのためのモバイルコンピューティング技術, マルチメディア通信と分散処理ワークショップ論文集, pp.159-164 (1996).
- 3) 木村 章, 仲 秋朗, 劉 渤海, 塚本昌彦, 西尾章治郎: アクティブデータベースシステムを用いた移動体ビューの設計および実装, 第6回データ工学ワークショップ (1996).
- 4) Acharya, A., Badrinath, B.R., Imielinski, T. and Navas, J.C.: A WWW-based Location-Dependent Information Service for Mobile Clients, *Fourth International WWW Conference*, Vol.8, No.1 (1994).
- 5) Acharya, A., Imielinski, T. and Badrinath, B.R.: DATAMAN project: Towards a Mosaic-like Location-Dependent Information Service for Mobile Clients, *Rutgers University Department of Computer Science* (1994).
- 6) Bennett, F., Richardson, T. and Harter, A.: Teleporting - Making Applications Mobile, *IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz (1994).
- 7) Hirano, S.: HORB - The Magic Carpet for Network Computing, [http:// ring.etl.go.jp/](http://ring.etl.go.jp/)

openlab/horb/.

- 8) Kaashoek, F., Pickney, T. and Tauber, J.: Dynamic documents: Mobile wireless access to the WWW, *Workshop on Mobile Computing Systems and Applications* (1994).
- 9) Sun Microsystems: Java Language Environment, <http://www.sun.co.jp/smi.jp/whitepapers/JavaPS/index.html>.
- 10) Voelker, G.M. and Bershad, B.N.: *MOBI-SAIC: AN INFORMATION SYSTEM FOR A MOBILE WIRELESS COMPUTING ENVIRONMENT*, Kluwer Academic Publishers, Boston (1996).

(平成 9 年 5 月 12 日受付)

(平成 10 年 2 月 2 日採録)



寺西 裕一 (正会員)

平成 5 年大阪大学基礎工学部情報工業科卒業。平成 7 年同大学院基礎工学研究科修士課程修了。同年日本電信電話(株)情報通信研究所入所。主にデータベース、情報システム

の研究開発に従事。



種茂 文之 (正会員)

平成 3 年名古屋大学工学部情報工学科卒業。平成 5 年同大学院工学研究科修士課程修了。同年日本電信電話(株)情報通信網研究所入所。主にオブジェクト指向データベース、

ハイパーメディアの研究開発に従事。



梅本 佳宏 (正会員)

昭和 61 年大阪大学基礎工学部情報工学科卒業。昭和 63 年同大学院基礎工学研究科修士課程修了。同年日本電信電話(株)情報通信処理研究所入所。主にデータベース管理

システムの研究開発に従事。



寺中 勝美 (正会員)

昭和 49 年京都大学工学部理数学科卒業。昭和 51 年同大学院工学研究科修士課程修了。同年日本電信電話公社入社。主にデータベース管理システムの研究開発に従事。現在、

NTT 研究開発推進部総括部門長。IEEE-CS, ACM, 電子情報通信学会各会員。