PMAA（Personalized Media Access Architecture）：ユビキタス環境におけるメディアアクセスアーキテクチャ

岩本健嗣[†] 徳田英幸[†,††]

近年，研究領域，ビジネス領域を問わず，ユビキタスコンピューティング環境がさかんに取り上げられるようになった．ユビキタスコンピューティング環境では，部屋や公共の空間に様々な機器が埋め込まれ，これらをユーザが利用する．ユーザは，行く先々でこのような機器を利用することで，いつでもどこでも計算機環境を利用できるようになる．ユビキタスコンピューティング環境が今後発展すれば，様々な場所に公共的に機器が設置され，それを不特定のユーザが利用することが想定される．本稿では，ユーザが公共の空間を一時的に占有可能な計算機環境として，Personalized Public Spaceを提案する．Personalized Public Spaceにおいて，マルチメディアアプリケーションを実現するためには様々な課題が存在する．これらの課題を解決し，ユーザにメディアアクセスを提供するためのPMAA（Personalized Media Access Architecture）を提案し，設計・実装を行う．

PMAA: Media Access Architecture for Ubiquitous Computing

Takeshi Iwamoto[†] and Hideyuki Tokuda[†,††]

In recent years, Ubiquitous Computing is getting popular in not only research but buissness area. In Ubiquitous Computing, users can use many devices and sensors that are embedded in a room or space. If we assume the computing environment in which devices are embedded in the space such as room or public place, it is possible to realize media access environment which can be accessed anytime and anywhere with less restriction. This paper proposes the environment, named ersonalized Public Space, in which user can personalize devices in such a public space. The design and implementation of Personalized Media Access Framewor (PMAA) which realizes media access environment in Personalized Public Space is also described.

1. はじめに

近年，研究領域，ビジネス領域にかかわらず，ユビキタスコンピューティング環境[1),2)]に注目が集まりつつある．ユビキタスコンピューティング環境は一般的に，従来までのように単体の計算機に頼る計算機環境ではなく，ユーザの周辺に存在する複数の機器の協調によって，ユーザを支援する知的空間ととらえることができる．

ユビキタスコンピューティング環境では，部屋や公共の空間に様々な機器が埋め込まれ，これらをユーザが利用する．ユーザは，行く先々でこのような機器を利用することで，いつでもどこでも計算機環境を利用できるようになる．

ユビキタスコンピューティング環境が今後発展すれば，様々な場所に公共的に機器が設置され，それを不特定のユーザが利用することが想定される．このような環境を実現するためには，公共の機器を一時的に特定ユーザが占有し，あたかも自分専用の機器として利用できるような仕組みが必要となる．本研究では，公共の空間を個人の利用環境として一時的に占有可能な計算機環境として，Personalized Public Spaceを提案する．

Personalzed Public Spaceは公共空間のパーソナライゼーションに注目したユビキタスコンピューティング環境であり，以下の特徴を持つ．

- 公共的に提供された機器に対する一時的な個人占有と利用
- 複数機器の組合せなど，柔軟な機器利用方法の提供

† 慶應義塾大学政策・メディア研究科
Graduate School of Media and Governance, Keio University
†† 慶應義塾大学環境情報学部
Faculty of Environmental Information, Keio University

☆ 本研究の作業の継続性に関しては文部科学省科学技術振興調整費「人間支援のための分散リアルタイムネットワーク基盤技術の研究」の支援による．

Personalized Public Space を実現するために必要となる要素技術は非常に多い．本稿では，Personalized Public Space において，メディアアクセスをユーザに提供するための PMAA（Personalized Media Access Architecture）を提案する．PMAA は，Wapplet と呼ばれる分散アプリケーションを利用し，サービスの即興的な組合せや，環境変化に適応することができるアーキテクチャである．

本稿の構成は，まず 2 章で，パーソナライゼーションに注目したユビキタスコンピューティング環境として，Personalized Public Space を提案する．3 章で PMAA の概要を述べ，4 章でその設計を詳説する．5 章において，PMAA の実装について述べ，5 章で評価を行う．

2. Personalized Public Space

本章では，機器のパーソナライゼーションに注目したユビキタスコンピューティング環境である Personalized Public Space について述べる．Personalized Public Space は，周辺に存在する公共の機器が，ユーザの個人所有の機器のように利用可能な計算機環境である．はじめにパーソナライゼーションの概念について説明し，次に Personalized Public Space の概要を述べる．

2.1 パーソナライゼーション

本稿で想定するユビキタスコンピューティング環境では，不特定多数のユーザが，空間に埋めこまれた様々な機器を利用する．このような計算機環境では，機器のパーソナライゼーションが重要となる．パーソナライゼーションとは，ユーザの周辺に分散した公共の機器が，ユーザの要求や嗜好を認識し，それに合わせた動作を行うことである．特に将来，公共サービスとして駅や公園，繁華街などの街頭に多様な機器が埋め込まれることが予想される．これらの機器はあらかじめユーザの目的を想定したり，機器上に特定ユーザのための設定を保持したりしておくことは難しい．そのため，特定ユーザを対象としないユビキタスコンピューティング環境を実現するためには機器のパーソナライゼーションが必要不可欠となる．

本研究では機器のパーソナライゼーションに着目した計算機環境として Personalized Public Space を提案する．Personalized Public Space は，特定個人のために，自律的に自分自身を再設定，再構成可能な知的空間である．Personalized Public Space は以下のような機能を持つ．

- 公共環境の個人占有

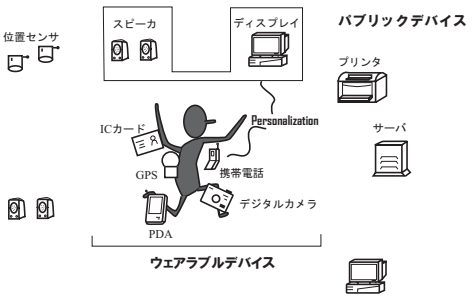

図 1 Personalized Public Space の概要
Fig. 1 Personalized Public Space.

機器やセンサが公共的に提供されている空間において，これらをユーザが利用したいときに一時的に占有し，あたかも自らの所有物であるかのように利用することが可能である．利用後はまたこれらの機器を解放し，他のユーザが利用することも可能である．

- 柔軟な機器利用方法の提供
公共の空間において，ユーザが利用できる機器は場合によって異なる．同じ空間であっても，他のユーザによって機器が占有されることが考えられるため，あらかじめ利用できる機器を想定することはできない．そのため，ユーザの要求に対して，機器を組み合わせたり代替デバイスを提供したりして，柔軟にサービスを提供する必要がある．

2.2 Personalized Public Space の概要

Personalized Public Space の概要を**図 1** に示す．

ユーザが保持している個人所有の機器をウェアラブルデバイスと呼ぶ．ウェアラブルデバイスには，PDA や携帯電話，携帯音楽プレーヤなどが含まれる．

公共空間に存在する機器をパブリックデバイスと呼ぶ．これらの機器は，不特定のユーザに対してその機器のハードウェアによる機能を提供する．

ウェアラブルデバイス，パブリックデバイスの集合によって Personalized Public Place は形成される．

2.2.1 ウェアラブルデバイス

ウェアラブルデバイスとは，ユーザが保持する個人所有の機器を指す．これらの機器は，ユーザによって持ち運ばれ，常時利用可能と考えることができる．ウェアラブルデバイスは携帯性の点からハードウェアは貧弱であることが多いが，無線ネットワークを利用可能な機器と想定しており，ウェアラブルデバイスどうしや，パブリックデイバスとの協調動作を可能とする．たとえば，ユーザにとって必要となる機能がパブリックデバイスだけでは提供できない際に，補完的に利用したり，また，ユーザインタフェースの提供に用いた

りする．

2.2.2 パブリックデバイス

オフィスや街頭などの公共の空間に設置され，不特定多数のユーザから利用されることを前提とした機器である．そのためウェアラブルデバイスと異なり，以下の制限を受ける．

- 特定の利用方法の提供ができない
  機器の利用方法はユーザによって異なる．そのため，利用ごとに異なるアプリケーションによって利用される可能性がある．
- 特定ユーザの嗜好の反映が困難
  不特定多数のユーザに利用されることを前提としているため，特にスケーラビリティやセキュリティの点から，個人の設定や作業の途中経過を機器に保存するのは現実的ではない．

3. PMAA：メディアアクセス提供のためのアーキテクチャ

本章では，Personalized Public Space で，ユーザにメディアアクセスを提供するためのアーキテクチャとして，PMAA（Personalized Media Access Architecture）を提案する．PMAA は，パブリックデバイスを用いたマルチメディアデータの閲覧，操作といったメディアアクセスを提供する．本章では，特に PMAA の目的と，実現のための要件について述べる．

3.1 PMAA の目的

PMAA は，ユーザに Personalized Public Space でメディアアクセスを提供する．定義として，何らかのメディアデータの閲覧や操作を行うアプリケーションが利用可能なことを，メディアアクセス可能とする．具体的なアプリケーションとして，ストリーミングのプレーヤや，テレビ電話，メーラなどが考えられる．

PMAA の要素機能は多岐にわたるが，本稿では以下の要素機能を実現することを目的とする．

- 作業環境の即興的構築
- 作業環境への個人設定の反映
- 移動などの環境変化への対応

Personalized Public Space では，ユーザが用いることのできるパブリックデバイスは，場所や状況によって変化する．よって，ユーザの作業環境は，利用可能なパブリックデバイスを即興的に用いて構築しなくてはならない．また，即興的に構築された作業環境のパーソナライゼーションのために，ユーザ個々の設定を反映する必要がある．さらに，PMAA は，移動などによって利用できる機器が変化した場合にも対応し，ユーザに対して作業環境を継続的に提供することが求められる．

3.2 PMAA 実現のための課題

PMAA を実現するために，以下のような課題を解決する必要がある．

- パブリックデバイスとアプリケーションの分割
- パブリックデバイスの切替え
- 作業の継続性

3.2.1 パブリックデバイスとアプリケーションの分割

Personalized Public Space では，作業環境は即興的に構築されるため，パブリックデバイスの利用目的はユーザやその目的によって異なる．そのため，パブリックデバイスはあらかじめ特定の機能を実現するアプリケーションを提供することはできない．たとえば，ディスプレイをビデオ再生として利用するか，GUI の画面として利用するかはユーザの目的に委ねられるため，ビデオ再生アプリケーションや GUI クライアントなどは，パブリックデバイスにあらかじめ用意しておくことはできない．

3.2.2 パブリックデバイスの切替え

ユーザが利用できるパブリックデバイスは，他のユーザの占有状況や，サービスの起動などにともなって変化する．アプリケーションは，この変化に適応し，利用するパブリックデバイスを切り替える方が望ましい場合がある．たとえば，ユーザの選択や嗜好によって，より処理に適したパブリックデバイスを選択するといった適応が考えられる．

パブリックデバイスを切り替える場合，Personalized Public Space では，異なるパブリックデバイスを用いて，同じメディアデータの閲覧や操作を行うことができなくてはならない．たとえば，動画のストリーミング再生をディスプレイで行っていた場合に，PDA のディスプレイしか利用できない環境に変化しても，再生を維持できなくてはならない．

3.2.3 作業の継続性

Personalized Public Space は，デスクトップ環境のような静的な計算機環境に比べて，場当たり的に利用される計算機環境である．そのため，ユーザの移動や状況に応じて作業が突然中断されたり，再開されたりすることがある．PMAA は，頻繁に起こる作業の中断と再開に対応するために，作業状態のスナップショットの取得と，そのスナップショットからの作業の再開を行う必要がある．

4. PMAA の設計

本章では，前章で述べた PMAA 実現の要件をふま

えて設計手法について考察し，それに従った具体的な設計について述べる．

前章で PMAA を実現するための以下の 3 つ課題を述べた．

- パブリックデバイスとアプリケーションの分割
- パブリックデバイスの切替え
- 作業の継続性

PMAA を設計するにあたり，各課題に対応して，次の 3 つの解決手法を採用する．

- Wapplet によるメディア・制御トラフィック分割
- WSIMA による機器の抽象化
- 使い捨てアプリケーション

以下の各節で，それぞれの手法について述べる．

4.1 Wapplet によるメディア・制御トラフィック分割

パブリックデバイスとアプリケーションの機能を分割するためには，パブリックデバイスが機能提供のためのインタフェースを提供し，ユーザの用いるアプリケーションが，そのインタフェースを通じてパブリックデバイスの機能を利用する手法が考えられる．この際，アプリケーションを実行する実行環境について，以下の 2 つの手法が考えられる．

- 1 つの PDA やウェアラブルコンピュータなどの情報端末上で実行
- 複数のパブリックデバイス上にアプリケーションを分散して実行

1 つの情報端末上でアプリケーションを実行する場合，アプリケーションとパブリックデバイス間のトラフィックは，アプリケーションを中心に完全な集中型となる（**図 2** (b)）．それに対し，複数のパブリックデバイス上にアプリケーションを分散する場合，そのトラフィックは各アプリケーション間で分散して行われる（図 2 (a)）．

メディアアクセスを提供することを考えた場合，アプリケーションが取り扱うトラフィックにはメディアデータが含まれる．そのため，トラフィックを集中させた場合，パフォーマンスやスケーラビリティの面で問題が生じる．それに対し，アプリケーションを分散する場合，トラフィックの分散は可能になるが，パーソナライゼーションを行うための集中制御が困難になる．

そのため，本研究では，Wapplet と呼ばれる分散アプリケーションを用いて，メディア・制御トラフィック分割を行う[3]．Wapplet は，利用する各パブリックデバイスごとに，アプリケーションを分散することで複数の機器の組合せを実現する．各アプリケーションは処理に応じてマルチメディアデータの交換を行い，

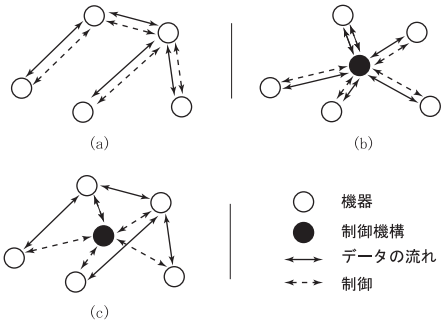

図 2 トラフィックの流れ
Fig. 2 Traffic flow.

パーソナライゼーションのための制御トラフィックは 1 カ所に集中して行う（図 2 (c)）．この方式によって，トラフィック負荷の集中を避け，また集中的な制御を実現できる．

4.2 WSIMA による機器の抽象化

作業環境の変化への適応を実現するために，本研究では，同じメディアを表現可能な複数の機器を抽象化し，単一のインタフェースを提供する手法を用いる．このインタフェースを WSIMA（Wapplet Serivice Interface for Multimedia Applications）と呼ぶ．WSIMA を，パブリックデバイスの種類によってグループ分けし，各グループごとにその機能を利用するためのインタフェースを定める．アプリケーションはグループごとのインタフェースを用いることで，同じグループに属する複数の機器を切り替えて利用することが可能になる．

しかし，機器の利用インタフェースは抽象化されているため，特定の機器に特化した機能などを利用することは困難になる．本研究では，マルチメディアデータを取り扱う（再生，表示など）ための，メディアアクセスを提供することに重点を置き，機器の操作や特定の機能の利用は GUI によって提供する．

WSIMA では，機器のグループ分けの方法として，mime[4] を用いる．実際には mime のトップレベルメディアタイプを利用し，機器を分類した．また，各メディアタイプに対して，入力と出力に分けた．本研究における機器の種類を**表 1** にまとめる．

各メディアタイプに分類される機器はアプリケーションに対して同じインタフェースを提供し，これにより利用方法が共通化され，機器の切替えが容易になる．

4.3 使い捨てアプリケーション

作業の継続性を実現するために，作業状態の保存とその復元を行う必要がある．PMAA では，アプリケーションを，パブリックデバイス上で動作するエンティ

表 1 機器の種類
Table 1 Appliance classes.

機器の種類		含まれる機器例
メディアの種類	入出力	
video:out	出力	ディスプレイ
video:in	入力	ビデオカメラ
image:out	出力	ディスプレイ，プリンタ
image:in	入力	デジタルカメラ
sound:out	出力	スピーカ，ヘッドホン
sound:in	入力	マイク，録音機器
text:out	出力	ディスプレイ，プリンタ
text:in	入力	キーボード，スキャナ

ティとそれらの制御部分に分割し，パブリックデバイス上で動作するエンティティに関しては，使い捨てとする．制御部分で，アプリケーションの作業状態を保持しておき，この作業状態から，作業の再開時にアプリケーションの再生成を行う．

このようなアプリケーションの再生成による対応は，いったん作業が中断され，新たな環境において再開されるような環境変化へ対応するために行われる．たとえば，ユーザの移動などにともなって，通信の切断をともなうような環境変化が考えられる．作業の再開時は，アプリケーションの実体が，作業内容から再生成されて，新しく利用するパブリックデバイスに分散しなおして実行される．

4.4 PMAA の構成

PMAA は以下の 3 つのコンポーネントから成る．

- 分散アプリケーション（Wapplet）
- パブリックデバイス上で機器の機能を提供するミドルウェア（サービスプロバイダ）
- アプリケーションを制御するための制御機構（ミッション機構）

Wapplet は，PMAA におけるアプリケーションであり，複数のパブリックデバイスに分散して実行される Wapplet ワーカと，その制御部分である Wapplet ハートからなる．サービスプロバイダは，Wapplet の実行環境と，パブリックデバイスの機能を Wapplet に提供するための WSIMA を提供する．ミッション機構は，Wapplet の制御機構であり，ユーザの個人情報を管理し，パーソナライゼーションを集中的に行うミドルウェアである．ミッション機構によって，分散して Wapplet を集中的に制御する．

図 3 に PMAA の概要を示す．

4.5 Wapplet

本アーキテクチャでは，アプリケーションはパブリックデバイス上のミドルウェアに分散して実行される．このミドルウェア上で動作するエンティティを Wapplet ワーカと呼ぶ．また，複数の Wapplet ワーカを集中制御するエンティティを Wapplet ハートと呼ぶ．1 つまたは複数の Wapplet ワーカと 1 つの Wapplet ハートからなるアプリケーション全体を，Wapplet と呼ぶ．

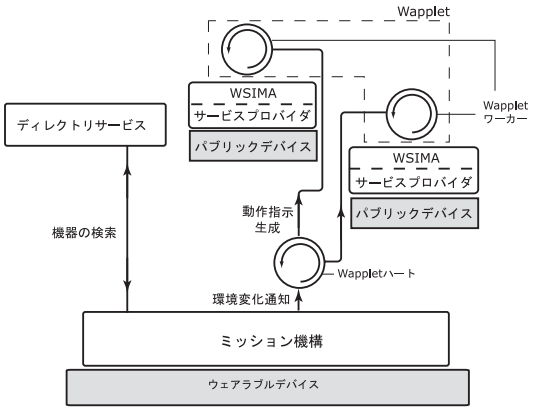

図 3 PMAA の概要
Fig. 3 PMAA.

4.5.1 メディアアクセスの提供

各 Wapplet ワーカは，ユーザにメディアアクセスを提供するために，必要なパブリックデバイス上に分散して動作する．サービスプロバイダ上に用意された WSIMA を通じて，パブリックデバイスの機能を利用し，メディアデータの表示などを行う．また，各 Wapplet ワーカは Wapplet アプリケーションを構成するその他の Wapplet ワーカと，アプリケーションの処理内容に応じた協調動作を行う．

4.5.2 パブリックデバイスの切替え

アプリケーションレベルでの機器の切替え機能を実現するために，PMAA では，Wapplet ワーカを新たに利用する機器に移動することによって実現する．具体的には，同じ WSIMA を提供する他のサービスプロバイダに移動して同じ WSIMA を通じて作業を継続する．

4.5.3 Wapplet ワーカの制御

Wapplet ハートは Wapplet ワーカの制御を行うエンティティであり，ミッション機構上で実行される．

Wapplet ハートは Wapplet ワーカと異なり Wapplet に 1 つだけ存在し，以下の内容を含む．

- Wapplet ワーカの構成
  Wapplet のワーカ構成（各ワーカの名前，必要とするパブリックデバイスの種類）を持つ
- 各 Wapplet ワーカの作業状態
  各 Wapplet ワーカの作業状態を管理し，後述する Wapplet ワーカの再生成に備える

- Wapplet ワーカの制御機能
  環境変化に適応して，Wapplet ワーカの移動の指示と，再生成を行う．そのため，現在の Wapplet ワーカの位置や作業状態を把握する．

Wapplet ハートは各 Wapplet ワーカと通信し，Wapplet ワーカの作業状態管理や，移動の指示を行う．また，ミッション機構と協調しユーザの状況や，機器の利用可否状況についての通知を受ける．

4.5.4 Wapplet ワーカの再生成

たとえば，ユーザの移動などによって，不測の環境変化が起こった場合や，パブリックデバイスに障害が起こった場合，Wapplet ワーカは，Wapplet ハートから切断されてしまう．作業の継続性を実現するために，切断後，作業が再開できるようなパブリックデバイスが利用可能になった際，あらためて Wapplet ワーカを再生成する．

Wapplet ハートは，作業の継続に必要なスナップショットを Wapplet ワーカから取得しておき，このスナップショットから Wapplet ワーカの再生成を行う．

4.6 サービスプロバイダ

PMAA で，各機器上で Wapplet ワーカを実行するためのミドルウェアをサービスプロバイダと呼ぶ．サービスプロバイダは，Wapplet ワーカが機器の機能を利用するためのインタフェースとして，WSIMA を提供する．

また，サービスプロバイダは，ミッション機構からの検索を可能にするために，機器の機能を登録する必要がある．登録は，機器の機能を提供するインタフェースをディレクトリサービスに登録することで行う．登録の際に，デバイス固有の特徴や性能を表す情報を同時に登録する．この情報は，複数の候補から実際に利用するサービスプロバイダを，ユーザが選択する際に利用される．

4.6.1 サービスプロバイダの切替え

Wapplet ワーカは，サービスプロバイダを切り替えるために，サービスプロバイダ間を移動する．そのとき，切替え後のサービスプロバイダも利用できる仕組みが必要となる．そのため PMAA では，機器の操作をいくつかのグループに分け，それぞれについてインタフェースを用意する（4.2 節参照）．

このインタフェースを WSIMA と呼び，同じグループに属するサービスプロバイダは同じ WSIMA を提供するため，Wapplet ワーカはこれらを代替的に利用することができる．たとえば，スピーカとヘッドホンは音声（sound）の出力を提供する機器として同じ機器の種類に分類され同じ WSIMA を通じて操作可能とし，代替的に利用できる．

4.7 ミッション機構

ミッション機構は，ユーザが持つウェアラブルデバイス上で実行されるミドルウェアで，Wapplet ハートを実行する機能と，サービスプロバイダの検索機能を提供する．

ミッション機構は，Wapplet ハートに対して，環境で利用可能なパブリックデバイスが変化したことを通知する．

4.7.1 サービスプロバイダの検索

ミッション機構は現在利用可能なサービスプロバイダを認識するために，ディレクトリサービスを用いたサービス検索を行う．ディレクトリサービスはある特定の範囲内のサービスプロバイダの管理を行い，ミッション機構の問合せに対して適当なサービスプロバイダのリストを通知する．

各サービスプロバイダはディレクトリサービスに対して提供する WSIMA の種類を登録し，ミッション機構はこの種類を基に検索要求を行う．

5. 実　　装

本章では，PMAA の実装として，Wapplet，サービスプロバイダ，およびミッション機構の実装について述べる．

実装は Java 言語（JDK1.3）を用いて行い，メディアデータ処理には JMF2.1 [5]を用いた．

5.1 Wapplet

Wapplet の基本機能として，機器を切り替えるための移動機能と，Wapplet ハートとの通信機能を実装した．また，WMISA を利用したメディアアクセスの提供と，Wapplet ワーカ間の協調動作のための API を用意した．

5.1.1 基本機能

Wapplet ワーカの移動機能として Java 言語によるオブジェクトシリアライゼーションと RMI（Remote Method Invocation）で実現している．

5.1.2 メディアアクセス API

メディアアクセスを提供するための，API としては，WSIMA のカテゴリごとのインタフェースを実装した．また，取り扱うメディアデータとして，RTP [6]を用いたストリーミングメディアとファイルを用いたストレージメディアの利用を実現した．

以下に，音声出力機器を利用する Wapplet ワーカのサンプルコードの一部を示す．

```
 1
 2  setRTPAddress("224.2.253.125","49150","1");
 3  //RTP アドレスの設定
 4
 5  AudioOut = createAudioOutPlayer(); //音声再生用オブジェクトの取得
 6                                     //音声再生機器のインタフェースを実
装している
 7   --- 省略 ---
 8  //イベントハンドラ GUI などからのイベントを受けとる
 9  public void MessageHandler(WappletEvent ev){
10     if(ev.getContents().equals("PLAY")){
11         AudioOut().play(); //音声再生用インタフェースの利用 (再生)
12     } else if(ev.getContents().equals("STOP")){
13         AudioOut().stop(); //音声再生用インタフェースの利用 (停止)
14     }
15 }
```

5.1.3 Wapplet ワーカ協調 API

Wapplet ワーカ間で協調動作を行うための API であり，メッセージ通信機能を提供する．メッセージの送信 Wapplet ワーカは，受信側の Wapplet ワーカの名前を指定し，任意の文字列を送ることができる．Wapplet ワーカの名前空間は Wapplet で閉じていればよく，プログラマが任意に付けることができる．Wapplet ワーカ間のメッセージ通信には，Wapplet ハートが介在するため，実際に Wapplet ワーカの現在の位置をプログラマが意識する必要はない．Wapplet ハートは，Wapplet ワーカの名前と現在実行されているサービスプロバイダを把握しており，送信 Wapplet ワーカは，Wapplet ハートに問合せを行って送信先を特定する．

本 API で用意したメソッドを以下に示す．`WappletMessage` は送信側，受信側の Wapplet ワーカ名とメッセージ本体を含む．

```
public void sendMessage(WappletMessage msg, String wapplet_name);
//メッセージ送信用メソッド
WappletMessage ReceiveMessage();
//メッセージ受信用メソッド．
```

5.1.4 Wapplet ハート

Wapplet プログラマが Wapplet ハートを作成するために，Wapplet が利用する機器の種類を登録するための API と，登録した機器の種類が変化した際のミッション機構からの通知を受け取るためのイベント駆動型の API を構築した．

また，ミッション機構からのサービス検索結果を受け取り，Wapplet ワーカに対する移動の通知を行うための機能を実装した．これらの通知の受け取りや Wapplet ワーカへの移動通知は，クラスライブラリの内部で処理されるため，Wapplet プログラマが実際に利用する必要はない．

5.2 サービスプロバイダ

サービスプロバイダの機能として，メディア処理機能，機器登録機能を実装した．

5.2.1 基本機能

サービスプロバイダの基本機能として，Wapplet ワーカを受け付け，実行する機能を実現した．サービスプロバイダはシリアライズされた Wapplet ワーカを，RMI を介して送受信し，スレッドを生成して実行する．

5.2.2 メディア処理

メディア処理を行う機能として，RTP とファイルによって提供されるメディアデータを処理する機能と，Wapplet モジュールからの操作によって操作を行う機能を提供した．

5.2.3 機器登録機能

機器情報の管理はディレクトリサービスを利用している．本実装ではディレクトリサービスとして LDAP [7] を用いた．LDAP サーバとしては openLDAP2.0.11 を用いた．サービスプロバイダは LDAP サーバと通信し，以下の項目を属性として登録する．

- サービスプロバイダ名
- 機器の種類
- IP アドレス

5.3 ミッション機構

ミッション機構としては，ディレクトリサービスを用いた機器の検索機能と，Wapplet ハートの実行環境を実装した．

5.3.1 基本機能

ミッション機構は LDAP サーバと通信し，Wapplet の実行に必要なサービスプロバイダを検索する．検索の属性は表 1 のメディアの種類を用いる．検索の結果は Wapplet ハートに通知される．

Wapplet が利用する機器の種類を登録するためのメソッドを以下に示す．

```
public void RegistServce(MediaType media, String mod_name);
//mod_name の名前を持つモジュールが必要とする機器の種類 media を登録する．
```

6. 評　　価

本章では，PMAA の評価として，サービスプロバイダの切替えにともなうコストの測定と，関連研究との比較を行う．

6.1 機器の切替えコスト

PMAA では，2 種類のサービスプロバイダの切替え方式を提供している．1 つは，Wapplet ワーカの移動による切替えであり，同一の空間内で，サービスプロバイダを切り替える際に用いられる（方式 A）．2 つ目は，Wapplet ワーカの実行状態を保存しておき，それを用いて Wapplet ワーカーを新しいサービスプロ

表 2 実験環境
Table 2 Measurement environment.

PC	CPU	OS
Service A	Pentium III 800 Mhz	Windows 2000
Service B	Pentium III 800 Mhz	Windows 2000
Mission	Pentium III 1 Ghz	Windows 2000

表 3 実験結果
Table 3 Measurement.

1 Kbyte		5 Kbyte		10 Kbyte	
方式 A	方式 B	方式 A	方式 B	方式 A	方式 B
23	79	30	89	39	97

単位：10 msec

バイダに再生成する方法である（方式 B）．これらの 2 方式について，それぞれのコストを測定した．

2 つの PC（Service A，B）を用意し，これらのホスト間で Wapplet ワーカの移動と再生成を行った．作業状態として，1 Kbyte，5 Kbyte，10 Kbyte のダミーデータを Wapplet ワーカに持たせて実験を行った．また，別の PC（Mission）を用意し，ミッション機構を動作させ，Wapplet ハートを利用して，移動・再生成の指示，作業状態の取得を行った．各 PC の条件を**表 2** に示す．各 PC は，IEEE802.11 準拠の無線 LAN を用い，ad-hoc モードで接続した．また，実験の結果を**表 3** に示す

この実験から，Wapplet の再生成が切替えにくらべて，よりコストがかかることが分かった．ただし，この実験では作業状態の取得時間が含まれており，これを適宜事前に行うことができれば，そのコストは低く抑えることができる．また，どちらの方式かによらず，作業状態として保存するべき情報の大きさも問題となる．

6.2 関 連 研 究

本研究の関連研究として，ユビキタスコンピューティング環境において，機器の組合せや，アプリケーション環境を提供する他の研究との比較を行う．

Hive[8]は移動エージェントの枠組みを利用して様々な機器の組合せを実現している．複数のエージェントどうしが，メッセージ交換によってお互いに協調して動作することで機器の組合せを実現しており，またシャドウと呼ばれる実際のハードウェアを抽象化し制御するためのインタフェースが用意されている．本研究では，Wapplet ハートによって，各 Wapplet ワーカを制御することで，環境を集中的に制御することを可能にしている．Hive では完全に分散したアプリケーションの構築を目的としており，Personalized Public Space のようなユビキタス環境を実現する場合，環境を集中制御する機能が考慮されていない．

Ninja Project[9),10)]や Danse[11)]では．PDA などの小型のクライアントがサービスプロバイダの情報を静的なドキュメントとして取得し，これを用いて機能の組合せを実現する．この場合，サービスプロバイダに対しては簡単なコマンドや決められた操作要求を送信することで，サービスプロバイダの制御を実現している．この方式では，サービスプロバイダを利用するためのインタフェースの統一化が容易であり，またサービスプロバイダどうしの組合せも，ドキュメントの記述を適宜参照しサービスプロバイダを検索することで行う．たとえば，電灯のような家電製品の電源投入や，リモコン操作のような作業には非常に有用である．しかし操作自体が，簡単なコマンドの送信によって実現されるため，複雑な動作や組合せ，特にアプリケーションに依存するようなデータの交換や操作が困難になる．それに対し本研究では，メディアアクセスを提供することが目的であるため，単純な機器の制御だけでは十分ではない．各機器にサービスプロバイダを配置する方式は，各機器が一定の計算能力を有することを前提とする反面，アプリケーションが任意にその組合せや利用方法を定義することが可能であるため，複雑な作業を実行することが可能になる．

Jini[12)]では，アプリケーションが必要な機器を検索し，検索の結果として機器を利用するためのインタフェースを取得することができる．Wapplet アーキテクチャでは，機器のインタフェースは機器の種類によって抽象化されている．そのため，Jini に比べて機器独自の機能を利用できない可能性があるが，同じアプリケーションによって機器を代替利用する機構を持つ．

7. お わ り に

本稿では，ユビキタスコンピューティング環境においてパーソナライゼーションに注目した計算機環境として Personalized Public Space を提案した．特に Personalized Public Space において，ユーザに対してメディアアクセスを提供するためのアーキテクチャである，PMAA（Personalized Media Access Architecture）の設計・実装について述べた．

Personalized Public Space において，メディアアクセスを実現するためには，いくつかの課題が存在することを明らかにし，PMAA の設計方針として，課題を解決するための解決手法を考察した．

PMAA を用いることで，ユビキタスコンピューティング環境において，ユーザは即興的に機器を組み合わせてメディアアクセスを実現することが可能である．

また，PMAA は頻繁に起こりうる環境変化に対して，利用する機器の切替えや，作業状態の保存によって適応することができる．

しかし，本アーキテクチャは，Personalized Public Space を実現するための第 1 歩であり，様々な課題が存在する．特にパーソナライゼーションのための，認証機構に対する取組みが必要となる．PMAA は認証機構とは切り離されているため，認証が行われたことを前提にアプリケーションを動作させるが，今後，なんらかの認証機構との協調を行う必要がある．また，ミッション機構に，個人の状況や，嗜好を認識させ，これに Wapplet の動作を適応させることで，より高度なパーソナライゼーションを行うことが可能となる．

参 考 文 献

1) Weiser, M.: Some Computer Science Issues in Ubiquitous Computing, *Comm. ACM*, Vol.36, No.7, pp.74–83 (1993).
2) Weiser, M.: The Computer for the Twenty-First Century, *Scientific American*, Vol.265, No.3, pp.94–104 (1991).
3) Iwamoto, T., Nishio, N. and Tokuda, H.: Wapplet: A Media Access Framework for Wearable Applications, *Proc. 16th International Conference on Information Networking* (*ICOIN-16*), Cheju Island, Korea (2002).
4) Borenstein, N. and Freed, N.: *MIME* (*Multipurpose Internet Mail Extensions*), RFC 1341 (1992).
5) Sun Microsystems: *Java Media Framework API*. http://java.sun.com/products/java-media/jmf/
6) Schulzrinne, H., Casner, S., Frederick, R. and Jacobson, V.: *RTP: A Transport Protocol for Real-Time Applications*, RFC 1998 (1996).
7) Yeong, W., et al.: Lightweight Directory Access Protocol (1995). Internet Request For Comments RFC 1777.
8) Minar, N., Gray, M., Roup, O., Krikorian, R. and Maes, P.: Hive: Distributed Agents for Networking Things, *Proc. ASA/MA'99, the 1st International Symposium on Agent Systems and Applications and 3rd International Symposium on Mobile Agents* (1999).
9) Chandrasekaran, S., Madden, S. and Ionescu, M.: Ninja Paths: An Architecture for Composing Services over Wide Area Networks.
10) Hodes, T.D., Katz, R.H., et al.: Composable Ad-hoc Mobile Service for Universal Interaction, *Proc. 3rd Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pp.1–12 (1997).
11) 板生知子，松尾真人：適応型ネットワーキングサービス環境 DANSE, 電子情報通信学会論文誌, Vol.J82-B, No.5, pp.730–739 (1999).
12) Sun Microsystems: *Jini Architecture Specification* (2000).

(平成 14 年 7 月 8 日受付)
(平成 14 年 12 月 3 日採録)

岩本 健嗣

1998 年慶應義塾大学環境情報学部卒業．2000 年同大学院政策・メディア研究科修士課程修了．現在，同大学院政策・メディア研究科後期博士課程に在学中．ウェアラブル・ユビキタスコンピューティング等の研究に従事．

徳田 英幸（正会員）

慶應義塾大学より工学修士．カナダ，ウォータールー大学より Ph.D.（Computer Science）．現在，慶應義塾大学常任理事，同大学環境情報学部教授．分散リアルタイムシステム，マルチメディアシステム，通信プロトコル，超並列・超分散システム，モバイルシステム等の研究に従事．IEEE，ACM，日本ソフトウェア科学会各会員．