ヒューマンインタフェース　96−5
モバイルコンピューティングと　19−5
ワイヤレス通信
（2001．11．15）

ウェアラブルネットワークにおけるマルチメディアアプリケーションフレームワーク

岩本健嗣[†]　西尾信彦[†]　徳田英幸[†,††]

近年，一般のユーザが計算機を用いてマルチメディアアプリケーションを利用する機会は非常に多い．また，このようなアプリケーションをいつでもどこでも利用したいという要求も高まっている．しかし，既存のモバイルコンピューティングの手法では，1台の機器を持ち歩くことでこれに対応するため制限も大きい．一方で，様々な機器が部屋や公共の空間に埋め込まれるような計算機環境を想定した場合，これらの機器を組み合せることで，いつでもどこでも利用可能な，制限の少ないマルチメディア環境を実現することも可能となる．本稿では，このような環境を実現する手法について考察し，機器の組み合せを実現するためのウェアラブルネットワークを提案する．ウェアラブルネットワーク環境実現の第1段階としてウェアラブルコンピュータと周辺の機器との連携を支援するソフトウェアアーキテクチャである，Wapplet フレームワークを構築する．

Media Access Application Framework for Wearable Network Environment

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

In recent years, there are many of opportunities when users realize multimedia applications by using computers. In addition, there is also strong demand to utilize these multimedia applications anytime and anywhere. However, existing mobile computing approach has its limits, because in this approach the multimedia application is processed with only one computer along with the user. In contrast, 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 approach of wearable network which realize the collaboration of devices. The design and implementation of the Wapplet framework which realizes media access environment in the wearable network is also described.

1. はじめに

近年，計算機を用いてマルチメディアアプリケーションを利用する機会は多く，例えば，音楽の再生や，テレビ，DVD の閲覧といった様々なアプリケーションが利用されている．また，一方で家電機器や高性能なスピーカ，ディスプレイといった AV 機器が，ユーザの周辺に存在し，自由に利用可能になりつつある．これらの機器は，IEEE1394[1)]や Bluetooth[2)] などを用いたホームネットワーク環境で，AV データの送受信も可能となる．オフィスや居間などの環境に存在するこのような機器を利用することで，いつでもどこでも利用可能なマルチメディア環境をユーザに提供可能となる．

† 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

既存のモバイルコンピューティングも，いつでもどこでも利用可能な計算機環境の実現を目的としているが，主にラップトップ PC や PDA などを持ち歩くことで，場所や時間を選ばない計算機環境を提供することに主眼が置かれている．これに対し，本研究では部屋や公共の空間に様々な機器が埋め込まれ，ユーザからネットワークを介して利用可能な環境を想定する．この環境において，ユーザが行く先々で周辺の機器を組合せてその要求を実現することで，いつでもどこでも利用できる計算機環境の実現を目指す．

このような計算機環境を実現するために，様々な構築形態が考えられる．周辺の機器の組合せや利用環境の動的な構築は，パーベイシブコンピューティング[3)]やユビキタスコンピューティング[4)5)]といった分野で研究が行われている．本研究ではこれらの研究を踏まえ，いつでもどこでも利用可能なマルチメディア環境の実現を目指し，以下の課題に取り組む．

- 周辺の機器を適宜利用したマルチメディア環境の構築

- 移動などに伴なう状況変化への適応
- ユーザの設定や好みへの適応

上記の課題の実現のために，本稿では，既存の機器の組合せ手法に加え，ウェアラブルコンピュータに着目した解決手法を提案する．その第1段階として，ウェアラブルコンピュータと周辺の機器との連携を支援するソフトウェアアーキテクチャであるWappletフレームワークを提案し構築する．Wappletフレームワークは，周辺の機器を適宜利用してマルチメディア環境を提供するためのアプリケーションフレームワークである．本フレームワークを利用することで，周辺の機器を協調させ，また利用できる機器の変化に適応可能なアプリケーションを構築できる

本稿では，2章で本研究で目的とする，いつでもどこでも利用可能なマルチメディアアプリケーションを構築する手法について考察し，本研究で採るべき手法について述べる．その手法を踏まえ，3章ではWappletフレームワークの設計について説明し，4章でその実装につい述べる．5章で関連研究について述べ，6章で本稿の論旨をまとめる．

2. 分散環境におけるマルチメディアアプリケーション

本章では，いつでもどこでも利用可能なマルチメディア環境を構築するために，ユーザ周辺の機器を効果的に組合せて利用する計算機環境の実現手法について考察する．次にこの手法を踏まえ，ウェアラブルコンピュータを主要な構成要素として捉えた実現手法として，ウェアラブルネットワーク環境を提案する．最後にウェアラブルネットワーク環境において，マルチメディア環境実現のために解決する課題を明かにする．

2.1 周辺の機器を利用する計算機環境

ネットワーク接続された周辺の機器を，適宜組合せて利用することでサービスを実現する計算機環境に関して，近年様々な研究が行われている．

本稿ではこれらの研究を次の2つの手法に分類する．

- **手法 1:分散制御による機器の協調**
- **手法 2:集中制御による機器の協調**

手法1では，遍在する機器が自律的に自らの機能やユーザの状況を把握し，それぞれが協調動作する．アプリケーションは各機器に分散して実行され，機器制御のための特別な機構を用意する必要はない．また，アプリケーションは各機器に分散しているため，各機器に特化した機能を利用できる．しかし，ユーザの個人的な設定の保持や，ユーザの状況の取得が困難で，パーソナライズされた利用環境を構築することは難しい．

手法2では，ユーザが身につけた小型端末などに制御機構を用意し，これによって周辺の機器を集中的に制御する．制御機構は周辺の機器やその制御方法を認識し，これを利用する．アプリケーションは制御機構上で動作し，各機器の制御インタフェースを取得して周辺の機器を利用する．この手法では，このような制御機構が，環境またはユーザの保持する機器に必要となるが，制御機構によってユーザの状況を取得できれば，それをアプリケーションに反映することも可能である．また個人的な設定を保持し，それをアプリケーションに適応することもできる．

2.2 本研究における手法の考察

本研究では，手法1，2の特徴を踏まえて，これらを組み合わせた手法を提案する．つまり，パーソナライゼーションを考慮して機器の制御については，手法2を用いて集中制御し，メディアデータの処理に関しては手法1を用いて分散制御する．つまり，本手法では制御データは制御機構によって集中的に管理し，マルチメディアデータは処理を行う各機器が処理を行う．

2.3 ウェアラブルネットワーク環境

本研究で採る手法では，周辺の機器を管理し制御する機構が必要となるが，これに要求される機能として以下の3つを想定する．

- ユーザと環境を共有し，監視することが可能
- 長時間の利用が可能
- ストレス無くユーザが保持することが可能

このような要件を満たす機器としてウェアラブルコンピュータが存在する．一般的に，ウェアラブルコンピュータは常時着用して利用することが想定されており，ユーザと常に環境を共有することが可能である．本研究では，ウェアラブルコンピュータを用いて周辺の機器を集中的に制御することで，これらの機器の組合せを実現する．このような機器の組合せを実現する環境としてウェアラブルネットワーク環境[6)7)]がある．

ウェアラブルネットワーク環境はユーザが保持している様々なウェアラブル機器と，遍在機器を組み合わせて利用する超機能分散型のコンピューティング環境である．その際，ウェアラブルコンピュータが周辺の利用可能な機器を把握して，アプリケーションを制御する．ユーザの移動などによって，利用可能な機器の集合が変化した場合は，その変化に対して代替的に利用可能な機器が存在すれば，アプリケーションを制御し，利用する機器を切り替える．また，ユーザの状況や作業状態が把握可能であれば，それらもアプリケーションに適用する．

つまりウェアラブルネットワーク環境は，既存のウェアラブルコンピュータのようにいわゆるPCの機能を超小型化した計算機を装着する計算機環境ではなく，ウェアラブルコンピュータを様々な環境の情報中枢として捉え，アプリケーションの実行のために，周辺の機器を積極的に利用する計算機環境である．

ウェアラブルネットワークの概要を**図1**に示す．

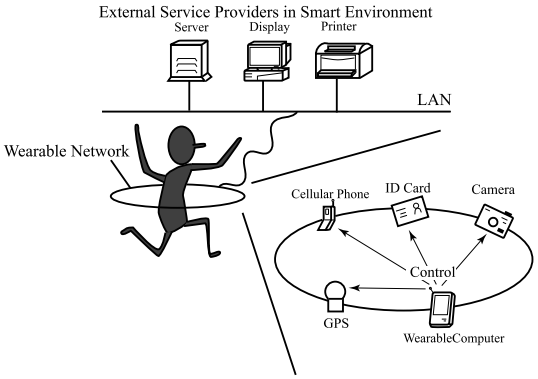

図1　ウェアラブルネットワーク
Fig. 1　Wearable Network

ウェアラブルネットワーク環境では，ユーザの持っている機器を近距離の無線ネットワークで接続し，ウェアラブルコンピュータから管理する．また，ユーザが所持する機器以外にも，環境に存在する利用可能な機器を外部のディレクトリサービスを用いて認識して積極的に利用する．

ウェアラブルネットワークに必要な機能を，次の3つにまとめることができる．

- 機器の協調
- 機器利用環境のパーソナライゼーション
- 状況適応

本研究では，これらの機能を備えるウェアラブルネットワーク環境において，マルチメディア環境を実現することを目的とする．

3. Wapplet フレームワーク

本章では，Wapplet フレームワークの概要と設計について述べる．Wapplet フレームワークは，ウェアラブルネットワーク環境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の枠組みである．最初にWapplet フレームワーク実現の課題について述べ，Wapplet フレームワークの構成と各機能について説明する．

3.1 Wapplet フレームワークの課題

本節では，Wapplet フレームワークで実現するべき機能をウェアラブルコンピュータ，アプリケーション，機器のそれぞれに分けて述べる．

3.1.1 ウェアラブルコンピュータ

機器の探索

ウェアラブルコンピュータは，常に周辺に存在する機器やそれらの機能を把握する必要がある．また，あらかじめアプリケーションが必要とする機器を認識する必要がある．

アプリケーションの制御

ウェアラブルコンピュータは機器探索機能によって把握した周辺の利用可能機器が変化した場合，それに応じてアプリケーションを実行したり，変化をアプリケーションに通知する必要がある．

3.1.2 アプリケーション

機器変化への適応

アプリケーションはウェアラブルコンピュータのアプリケーション制御機能と協調し，利用可能な機器の変化に応じて，その動作を適応させる必要がある．

本研究では，この適応内容を以下の2つに分類する．

- 機器の切り替え
- 動作変更

機器の切り替えは，アプリケーションが現在利用している機器そのものを変更する適応で，動作変更は機器の変更に応じてそのサービスの質を変更したりメディア変換などを行う適応である．また，これらの適応は同時に行われる可能性もある．

メディアデータの処理

各アプリケーションは，マルチメディアアプリケーションとして動作するために，利用する機器と協調する必要がある．その処理内容はメディアデータによって異なり，例えば，ディスプレイに動画を表示したり，スピーカから音声を出力するといった動作が考えらえる．これらの動作は，アプリケーションが各機器を操作することで実現される．また，上述の機器の切り替えが行われた後も，切り替え後の機器と連携して継続的にメディア処理を行える必要がある．

3.1.3 機　　器

アプリケーションへのインタフェース

アプリケーションによって処理，取得されたマルチメディアデータを表示したり出力するために，各機器はアプリケーションに対するインタフェースを提供しなくてはならない．しかし，機器の切り替えを前提にすると，このインタフェースは特

定の機器に依存することはできないため，各機器を適切に抽象化する必要がある．

機器登録

機器の探索を可能にするために，機器は各自の機能を登録する必要がある．一般的には，登録と検索はディレクトリサービスを用いることで実現可能である．しかし，代替可能な機器の発見を可能とするために，特定の機器名などで登録や検索することはできない．そのため，様々な機器を抽象化して登録する方式を検討する必要がある．

3.2 Wapplet フレームワークの構成

本節では，Wapplet フレームワークの構成について述べる．**図 2** に Wapplet フレームワークの概要を示す．前節で述べた機能とその実現の課題を踏まえて，Wapplet，サービスプロバイダ，ミッション機構の 3 つの構成要素に分割して設計した．以下で各構成要素について順に述べる．

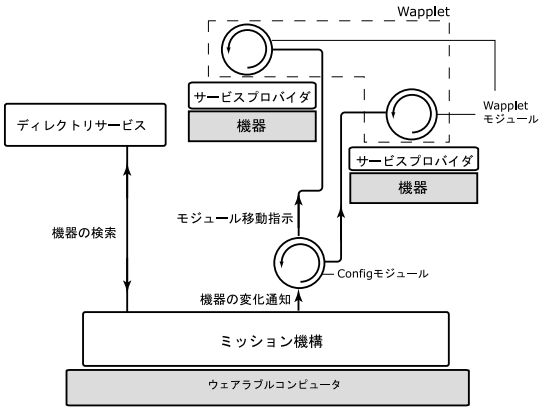

図 2 Wapplet フレームワーク
Fig. 2 Wapplet Framework

3.2.1 Wapplet

本研究では，アプリケーションが周辺の機器を利用するための手法として，分散制御の手法を採用することを 2.1節で述べた．そのため，各機器上では 1 つの分散アプリケーションの一部の機能が実行されているものとし，この機器上で動作する部分を Wapplet モジュールと呼ぶ．そして 1 つまたは複数の Wapplet モジュールを統合する 1 つのアプリケーションを Wapplet と呼ぶ．

アプリケーションレベルでの機器の切り替え機能を実現するために，Wapplet フレームワークでは，Wapplet モジュール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．メディアデータ処理機能は，Wapplet の API によって提供される．API では，後述する機器が提供するインタフェースを利用してメディアデータを処理する．API の詳細については 3.3.2項にて述べる．

3.2.2 サービスプロバイダ

各機器上で分散して実行されるアプリケーションの実行環境として，Wapplet モジュールを実行するためのミドルウェアを提供する必要がある．Wapplet フレームワークでは，こ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 モジュールに対して，機器の機能を利用するためのインタフェースや実行環境を提供する．サービスプロバイダのインタフェースについては，3.4.1項で述べる．

またサービスプロバイダは，ウェアラブルコンピュータによる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．

3.2.3 ミッション機構

ウェアラブルコンピュータでは機器の検索機能を実現する必要がある．機器の検索機能は，Wapplet が必要とするサービスプロバイダを検索し，Wapplet が利用するサービスプロバイダが変化した際には，各 Wapplet モジュールに移動の通知を行う．

Wapplet が必要とする機器を認識したり，各 Wapplet モジュールへの通知を行うために，Wapplet はウェアラブルコンピュータ上で Config モジュールを実行する．Config モジュールは Wapplet の各モジュールを制御したりミッション機構に Wapplet の実行に必要となる機器を登録する機能を持つ．Config モジュールは，他の Wapplet モジュールと異なり，ウェアラブルコンピュータ上で実行される．そのためウェアラブルコンピュータ部では，Config モジュールの実行環境を持つ必要がある．

3.3 Wapplet

本節では Wapplet の詳細について述べる．Wapplet は複数の機器を利用するアプリケーションであり，各機器上のサービスプロバイダで実行される Wapplet モジュールから構成される．また，Wapplet はミッション機構で実行される 1 つの Config モジュールを持つ．

3.3.1 Wapplet モジュール

Wapplet モジュールはサービスプロバイダ間を移動可能なモジュールであり移動先で 1 つのアプリケーションの一部としての機能を果たす．各モジュールはサービスプロバイダのインタフェースを操作して，機器の制御やメディアの表示を行う．

利用可能な機器が変化した場合，Wapplet モジュー

ルは次の 2 つの状況に対応する．

- 代替可能な機器が存在する場合
  Wapplet モジュールを代替可能な機器に移動し，作業を継続する．
- Wapplet の実行に充分な機器が発見できない場合
  メディア変換を用いて作業の質を変更して実行を継続するか，機器が利用できるようになるまで作業を一時停止する．

この判断は，後述するミッション機構と Config モジュールの連携によって行われる Wapplet フレームワークではこれらの判断や移動の操作は Wapplet プログラマから隠蔽されるため，機器の変化を意識することなくプログラムを記述できる．

3.3.2 Wapplet API

Wapplet プログラマが Wapplet を容易に作成するために，Wapplet フレームワークでは次の 2 つの API 群を提供する．これらの API は，メディアの表示やメディアデータの取り扱い，各モジュールの協調を実現する．これらの API の具体的な内容は，4章にて述べる．

- 機器制御 API
  サービスプロバイダは，3.4.1項で述べる機器の種類に応じたインタフェースを提供している．作成者はそのモジュールが利用する機器の種類に応じたインタフェース利用することで機器の制御が可能である．
- Wapplet モジュール間協調 API
  Wapplet モジュール間で制御情報のような簡単なメッセージをやりとりする方法を提供する．その際各モジュールの位置は Config モジュールによって管理されており，移動や位置を意識することなく通信することが可能である．

3.4 サービスプロバイダ

サービスプロバイダは，Wapplet モジュールを実行するために以下の機能を持つ．

- Wapplet モジュールのランタイム
- Wapplet モジュールがサービスプロバイダの機能を利用するためのインタフェース

Wapplet モジュールはサービスプロバイダ間を移動し，そのインタフェースを通じてサービスプロバイダの機能を利用する．

各サービスプロバイダの インタフェースは機器の種類毎に規定されており，Wapplet 開発者は特定のサービスプロバイダに依存せず，この機器の種類に依存したモジュールを構築すれば良い．

3.4.1 サービスプロバイダのインタフェース

Wapplet モジュールがサービスプロバイダ間を移動しても，移動前のサービスプロバイダと同様に利用できる仕組みが必要となる．そのため Wapplet フレームワークでは機器の操作を幾つかのカテゴリに分け，それぞれについて抽象化したインタフェースを用意する．種類が同一の機器間を移動する場合は，Wapplet モジュールは特別な変更無しに動作することができる．

Wapplet フレームワークにおける機器の種類を**表 1** にまとめる．

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
image:out	出力	ディスプレイ，プリンタ
sound:out	出力	スピーカ，ヘッドホン
text:out	出力	ディスプレイ，プリンタ

機器の種類はメディアの種類と入力，出力によって表わされる．メディアの種類は mime-type[8] のトップレベルメディアタイプを利用している．

また，同機種に属するサービスプロバイダは同じインタフェースを提供しているため Wapplet モジュールはこれらを代替的に利用することができる．例えば，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インタフェースで操作可能とし，代替的に利用できる．

この方法では，アプリケーション作成者は利用機種に応じたインタフェースを操作することで，マルチメディアアプリケーションを作成する．

3.5 ミッション機構

ミッション機構はサービスプロバイダの検索機能と，Config モジュールの実行環境を提供する．

Config モジュールは，Wapplet につき 1 つ Wapplet プログラマが作成し，その Wapplet のモジュール構成や実行に必要な機器の種類を記述する．ミッション機構は，Config モジュールから Wapplet の情報を取得し，機器の利用可否状況が変化し，通知の必要な場合，Config モジュールに通知を行う．Config モジュールはそれらの通知から各 Wapplet モジュールを制御する．

3.5.1 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．ディレクトリサービスの管理する範囲は実際に利用する組織に委ねられるが，ウェアラブルネットワーク環境では，部屋やフロアといったある区切られた空間毎に存在することを想定している．

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その機器の種類を登録し，ミッション機構はこの種類を基に検索要求を行う．これにより，サービスプロバイダはそのサービスプロバイダが取り扱うメディアデータの種類によって抽象化されるため，ミッション機構は具体的なサービスプロバイダの機器名や機器の種類とは非依存にサービスプロバイダを検索できる．

3.5.2 Config モジュール

Config モジュールは Wapplet の制御を行うモジュールであり，ミッション機構上で実行される．Wapplet プログラマは，Config モジュール用に用意された API を用いて，Wapplet のモジュール構成や実行に必要な機器の情報を含む Config モジュールを作成する．

Config モジュールは各 Wapplet モジュールと通信し，これらの位置の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作業状態，機器の利用可否状況についての通知を受ける．

- Wapplet モジュールの管理
  Wapplet モジュールが相互に通信を行い協調作業を行うため，Config モジュールは各 Wapplet モジュールの位置を把握する．
- ミッション機構との協調
  Config モジュールはミッション機構と協調動作し現在のサービスプロバイダの利用可否状況を把握する．これらが変化した際は，各 Wapplet モジュールに移動の指示を与える．この協調のために，Config モジュールとミッション機構間のインタフェースが用意される．

4. 実　　装

本章では，Wapplet フレームワークの実装として，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

4.1 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，Config モジュールとの通信機能を実装し，メディア処理や他の Wapplet モジュールとの協調のために API を用意した．これらの API はクラスライブラリとして提供した．

4.1.1 基 本 機 能

本実装では，モジュールの移動機能として 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．また，Config モジュールとの通信は TCP/IP による通信を利用した．Wapplet モジュールが，現在利用しているサービスプロバイダから他のサービスプロバイダへ移動する必要がある場合，まず Config モジュールから移動要求が送信される．Wapplet モジュールは，この要求を処理するために，現在利用中のサービスプロバイダに対して，移動要求を行う．Wapplet モジュールのシリアライズと RMI による送信処理は，実際にはサービスプロバイダによって実行される．

4.1.2 機器制御 API

遍在する機器の制御用 API としては，機器の種類毎に Java 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[9] 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．

4.1.3 モジュール間連携動作 API

Wapplet モジュール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．メッセージの送信側の Wapplet モジュールは，受信側の Wapplet モジュールの名前を指定し，任意の文字列を送ることができる．モジュールの名前空間は Wapplet で閉じていれば良く，プログラマが任意に付けることができる．Wapplet モジュール間のメッセージ通信には，ミッション機構上で実行される Config モジュールが介在する．Config モジュールは，Wapplet モジュール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，メッセージは Config モジュールを通じて受信側に送られる．メッセージは受信側のメッセージキューに溜められ，受信側モジュールが任意のタイミングで読み出せる．

4.2 サービスプロバイダ

サービスプロバイダとして，Wapplet の移動を提供する基本機能とメディア処理機能，機器登録機能を実装した．

4.2.1 基 本 機 能

サービスプロバイダの基本機能として，Wapplet モジュールの移動を行う機構を用意した．サービスプロバイダはシリアライズされた Wapplet モジュールを，RMI を介して送受信し，Wapplet モジュールを受け取った場合は，スレッドを生成して再実行する．

4.2.2 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メディアの再生を行う機能を提供した．

メディア再生は JMF2.1[10](Java Media Framework) によって提供されている `javax.media` パッケージを用いた．このパッケージを利用することで，音声の再生やビデオの表示などの操作を行うことが可能である．また，RTP やファイルによるメディアデータの取得も JMF を用いて実装した．

4.2.3 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[11] 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

サービスプロバイダ名と IP アドレスは Wapplet モジュールを移動する際に必要となる．

4.3 ミッション機構

ミッション機構は，ウェアラルコンピュータ上で動作し，ディレクトリサービスを用いた機器の検索と，Config モジュールの実行環境を提供する．ミッション機構の実装は，ディレクトリサーバの検索部分の実装には C++を用い，Config モジュールの実行環境部分の実装には Java 言語を用いた．

4.3.1 基 本 機 能

ミッション機構は LDAP サーバと通信し，Wapplet の実行に必要なモジュールを検索する．検索の属性は表 1 のメディアの種類を用いる．サービスプロバイダの実装で述べたように，検索の結果サービスプロバイダ名と IP アドレスを取得する．この情報を Config モジュールに通知することで，Config モジュールは Wapplet モジュールの移動を指示する．

4.3.2 Config モジュール

Wapplet プログラマが Config モジュールを作成するために，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とるためのイベント駆動型の API を提供する．また Wapplet モジュールに対する移動の通知を行うための API も用意した．これらの通知の受け取りや Wapplet モジュールへのメッセージの送信は，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. 関 連 研 究

本研究の関連研究として，機器の組合せによって計算機環境を提供する研究との比較を行う．

Hive[12] 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み合せを実現している．複数のエージェント同士が，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み合わせを実現しており，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．Wapplet フレームワークでは利用できるサービスプロバイダが変化した際にミッション機構でこれを検知し，Wapplet モジュールを移動して適応することが可能である．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，このような集中的な管理を行う機構が考慮されていない．そのため，アプリケーションがユーザの移動や環境の変化に適応して，機器を切り替える動作を実現することは難しい．

Ninja Project[13][14] や Danse[15] 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み合わせを実現する．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同士の組合せも，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．例えば，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 Wapplet フレームワークではメディアアクセスを実現することが目的であるため，単純な機器の制御では充分ではない．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，複雑な処理が可能になる．

Jini[16] 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 フレームワークでは，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．

Situated Computing Framework[17]は，PDAから環境に存在する様々な出力機器を利用して，ユーザの状況に適応したマルチメディアアプリケーションを提供することを目的としている．また，充分な機器が存在しない場合のメディア変換なども考慮されている．しかし，機器上で必要となるソフトウェアやアプリケーションとのインタフェースなどに関して，Wappletのような枠組みが規定されていない．

6. おわりに

本論文では，いつでもどこでも利用可能なメディアアクセス環境の実現について述べた．このような環境を実現するために，機器の組合せによるアプリケーションの構成機能が不可欠であり，我々は具体的な実験環境としてウェアラブルネットワーク環境の構築に着手した．そして，その実現の第1段階として，Wappletフレームワークを提案し構築した．Wappletフレームワークは，周辺にある機器にアプリケーションのモジュールを分散させ，各モジュールをウェアラブルコンピュータから集中的に制御するアプリケーションフレームワークである．このフレームワークを用いて，ユーザ周辺の機器を用いたメディア処理を実現し，また，利用できる機器の変化に対応した．特に様々な機器を，機器の種類によって抽象化することで，同じアプリケーションモジュールで代替的に利用する枠組みを提供することができた．

本フレームワークは，ウェアラブルネットワークにおいて，利用できる機器や環境変化に適応可能なアプリケーションを実現するための第1歩であり，今後，状況認識やユーザ設定を適応対象に取り入れることで，ユーザにより高い利便性を提供可能なアプリケーションを構築する仕組みを検討している．これにより，ユビキタスやパーベイシブといった計算機環境を構築するための一つ手法を確立できる．

参考文献

1) of Electrical, I. and Engineers, E.: *IEEE Standard for a High Performance Serial Bus mendment 1*.
2) SIG, B.: The Official Bluetooth Website. http://www.bluetooth.com.
3) Banavar, G., Beck, J., Gluzberg, E., Munson, J., Sussman, J. B. and Zukowski, D.: Challenges: an application model for pervasive computing, *Proceedings of The Sixth Annual International Conference on Mobile Computing and Networking (MobiCom2000)*, Boston, Massachusetts (2000).
4) Weiser, M.: Some Computer Science Issues in Ubiquitous Computing, *Communications of the ACM*, pp. 74–83 (1993).
5) Weiser, M.: The Computer for the Twenty-First Century, *Scientific American*, pp. 94–104 (1991).
6) Nobuhiko NISHIO, Takeshi IWAMOTO, T. N. and TOKUDA, H.: Wearable Network: Architecture and an Implementation, *The 2nd International Workshop on Networked Appliances*, New Brunswick (2000).
7) 岩本健嗣, 西尾信彦, 徳田英幸: An Adaptive Application Framework in Wearable Network, 情報処理学会 マルチメディア，分散，協調とモバイル (DICOMO 2000) シンポジウム論文集 (2000).
8) Borenstein, N. and Freed, N.: *MIME (Multipurpose Internet Mail Extensions)* (1992). RFC 1341.
9) Schulzrinne, H., Casner, S., Frederick, R. and Jacobson, V.: *RTP: A Transport Protocol for Real-Time Applications* (1996). RFC 1998.
10) Sun Microsystems, I.: *Java Media Framework API*. (http://java.sun.com/products/java-media/jmf/).
11) Yeong, W. et al.: Lightweight Directory Access Protocol (1995). Internet Request For Comments RFC 1777.
12) Minar, N., Gray, M., Roup, O., Krikorian, R. and Maes, P.: Hive: Distributed Agents for Networking Things, *Proceedings of ASA/MA'99, the First International Symposium on Agent Systems and Applications and Third International Symposium on Mobile Agents* (1999).
13) Chandrasekaran, S., Madden, S. and Ionescu, M.: Ninja Paths: An Architecture for Composing Services over Wide Area Networks.
14) Hodes, T. D., Katz, R. H. et al.: Composable Ad-hoc Mobile Service for Universal Interaction, *proceedings The Third Anual ACM/IEEE Internatioanl Conference on Mobile Computing and Networking*, pp. 1–12 (1997).
15) 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, 東京, pp. 730–739 (1999).
16) Sun Microsystems, I.: *Jini Architecture Specification* (2000).
17) Pham, T.-L., Sshneider, G. and Goose, S.: A Situated Computing Framework for Mobile and Ubiquitous Multimedia Access using Small Screen and Composite Devices, *the ACM International Conference on Multimedia* (2000).