卒業論文 2002 年度 (平成 14 年度)

ユビキタスコンピューティング環境における
ユーザインタフェースフレームワーク

指導教員

慶應義塾大学環境情報学部

徳田 英幸

村井 純

楠本 博之

中村 修

南 政樹

慶應義塾大学 環境情報学部 環境情報学科

守分 滋

卒業論文要旨 2002 年度 (平成 14 年度)

ユビキタスコンピューティング環境における ユーザインタフェースフレームワーク

論文要旨

本論文では, ユビキタスコンピューティング環境において，ユーザの多様な状況に適したユーザインタフェースを利用可能とする機構を提案した．そしてユビキタスコンピューティング環境のユーザインタフェースフレームワークの設計と実装を行った．本システムはユビキタスコンピューティング環境において，ユーザが状況にあわせたユーザインタフェースでのサービス利用を実現した．

ユビキタスコンピューティング環境では，ユーザが持ち歩く携帯端末に加えて環境に遍在する多数の情報機器が利用可能である．よって，ユーザが利用するアプリケーションやデバイスであるサービスと，ユーザインタフェースを多対多の関係で捉える，協調型ユーザインタフェースモデルが有効である．ユーザインタフェースを自由に選択できることによって，携帯性を優先するために操作性を犠牲にした携帯端末端末のみに頼らずともよくなる．しかし，従来のユーザインタフェースフレームワークでは，サービスとユーザインタフェースは一対一または一対多で対応しているため，こういったユーザインタフェースの利用法を実現できない．

本論文では，ユビキタスコンピューティング環境におけるユーザインタフェースフレームワークである i-face を設計・実装した．ユーザが利用可能なユーザインタフェースと独立して操作できるように，サービスの操作方法をユーザインタフェース非依存に抽象化したサービス記述を定義し，これによってユーザインタフェースを構成するユーザインタフェース生成機構を作成した．また，ユーザが状況の変化に応じてユーザインタフェースを切り替えて作業を継続できるように，ユーザインタフェース状態の保存・復元のためのユーザインタフェース状態保存機構を実現した．そして，ユーザインタフェース利用をする際に周りのセンサで操作を補佐するためのコンテキスト情報利用機構を実装した．

i-face の性能を定量的に評価し，実用に耐えうることを示した．また，関連研究と比較し，ユビキタスコンピューティング環境のユーザインタフェースとしての有用性を示した．

キーワード :
ユビキタスコンピューティング環境, ユーザインタフェース, コンテキストアウェア, 作業の継続

慶應義塾大学大学 環境情報学部 環境情報学科

Abstract of Bachelor's Thesis
Academic Year 2002

User-Inteface Framework for Ubiquitous Computing Environments

Summary

This research allows users to select the best user-inteface for their situation in ubiquitous computing environments.

The ubiquitous computing environment is brought on by the development of faster computers and network technologies. In this environment, devices with computing capabilities are ubituitous. The user can utilize interface devices in the environment to avoid the usability-size tradeoff of mobile terminals. However, traditional computing environments assumed a one to one mapping between services and user-interfaces. Therefore, a new user-interface framework for ubiquitous computing environments is needed.

In this paper, we designed and implemented i-face, a user-interface framework for ubiquitous computing environments. We design a user-interface generator, and define a UI neutral service description to generate UIs from. We also realize task preservation between UI changes by designing a UI-state store. Third, context utilizers allow context aware UIs

We quantitatively evaluate the performance of i-face and show its that it is fit for actual use. We also compare i-face with other user-interface framworks, and show its utility.

Keywords :
Ubiquitous Computing, User Interface ,Context Aware, Task Continuation

Faculty of Environmental Information, Keio University

Shigeru Moriwake

目 次

図 目 次

表 目 次

第1章

序論

本章では本研究の背景と意義，および本論文の内容構成について述べる．

1.1 本研究の背景

次世代のコンピューティング環境は技術の進歩により，情報機器の遍在と，それを繋ぐ無線の普及と有線帯域の向上が特徴となると考えられる．

情報機器の遍在

近年，高機能な情報機器の小型化と低価格化が進んでいる．例えば iPAQ[22] は，従来のデスクトップコンピュータに匹敵する計算処理能力を持った PDA(Personal Digital Assistant) である．また，パソコンの数分の一程度の価格に過ぎないゲーム機が，軍事転用可能だとしていちじ輸出規制を受けるほど，処理能力が高くなっている [1]．また，携帯電話や PDA といった携帯機器の持ち運びが小型化によって容易になり，一般に普及している．日本の携帯電話の普及人数は，2006 年に 8400 万人程度になると予想されている [2]．そして，低価格化によって従来はコンピューティング機能を有していなかったものの中にも高度なコンピューティング機能が埋め込まれるようになった．例えばソニーの Cocoon チャンネルサーバ [12] は，映像を記録するという VCR の機能にとどまらず，ネットワーク接続性を含めた多くの拡張機能を実現するために Linux OS を組み込んでいる． このように，ユーザが持ち歩く携帯端末が普及し，環境に存在する情報機器の増加しており，情報機器の遍在が進んでいる．

無線の普及と有線帯域の向上

携帯する情報機器をネットワーク化できる無線が普及している．低電力消費で携帯機器に適した Bluetooth 技術を搭載した携帯電話が発売された [13]．アメリカのディズニーランドでは，Bluetooth よりも電力消費が高いが，より多くの帯域がある IEEE802.11b が全園に設置された [4]．多数の情報機器を繋ぐ無線技術の発達と共に，それらの送受信する情報を運ぶ有線ネットワークの帯域も向上している．従来の光ファイバーも，WDM(Wavelength Division Multiplex)[5] によって帯域を数倍から数百倍に増やせる．家庭でも，IEEE 1394 や USB 2.0 といった数百 Mbps の帯域がある技術が利用できる．遍在する情報機器を互いに接続するための，無線ネットワークの普及と有線ネットワークの高帯域化が進んでいる．

ユビキタスコンピューティング環境

これらの傾向から，次世代のコンピューティング環境として，ユビキタスコンピューティング環境 [3] が想定される．ここでは，ユーザが持ち運ぶ端末や様々なモノに埋め込まれたコンピュータが遍在し，それらがさまざまなネットワークを介して協調動作する．

ユビキタスコンピューティング環境は，多数の機器とそこで動作するアプリケーションにより，多数のサービスをユーザに提供する．これは，従来のコンピューティング環境

とは対称的である．これまでユーザインタフェースは，サービスとセットで提供されてきた．テレビのユーザインタフェースはリモコンであり，ライトのユーザインタフェースは壁のスイッチといったように，ユーザがサービスを操作する際のユーザインタフェースはサービスとの組み合わせに限定された．しかし，これでは無数のサービスが存在するユビキタスコンピューティング環境において，ユーザはそれぞれのサービスに組み込まれた無数のユーザインタフェースの利用を強いられてしまう．

1.2 本研究の意義

ユーザは状況に適した機器をインタフェースとして利用し，かつ継続的にサービスを操作できなければならない．最適な機器は，ユーザの位置，まわりの環境，そしてユーザの状態によって異なる．例えば周りにディスプレイがなければ GUI は表示できず，満員電車ではジェスチャは迷惑で，電話をしているときは音声ユーザインタフェースを利用すると相手方が混乱する．

本研究の目的は，ユーザの状況にあわせて，最適なユーザインタフェースを利用可能とし，その利用をセンサの情報で補助することである．こうして，移動などによってユーザが使用できる機器が変化したとき，継続して別の機器での操作を可能とする．また，ユーザの入力に加えて，センサが感知したユーザの状態に基づいて操作の対象や度合いを調整する．

1.3 本論文の構成

本論文は，本章を含め全 7 章から成る．次章では，本研究の基盤環境であるユビキタスコンピューティング環境に適したユーザインタフェースのモデルを論考する．続く第 3 章では，ユビキタスコンピューティング環境に適したユーザインタフェースの要件をまとめ，それに照らして既存のシステムの持つ問題点を指摘し考察する．また，第 4 章において，本システムの概要を説明すると共に，システムの設計について述べる．第 5 章では，その実装について詳述し，第 6 章でシステムの評価を行なう．第 7 章にて，本論文をまとめ，今後の課題について言及する．

第2章

ユーザインタフェースモデル

本章では，ユビキタスコンピューティング環境に適したユーザインタフェースモデルについて論考する．

2.1 用語定義

はじめに，本節では本論文で使用する用語を定義する．

サービス

サービスは，端末またはネットワーク上に存在するアプリケーションや機器である．端末上に存在するサービスの例としてはスケジューラやメーラのアプリケーションがある．ネットワーク上に存在するサービスには，WebCAM[14][15] やディスプレイなどがある．

ユーザは，サービスを利用することによって，映画の鑑賞やプレゼンテーションなどの作業を遂行する．

インタフェース機器

ユーザがサービスを利用する際に，サービス操作しフィードバックを受け取るために用いる機器である．GUI ならフィードバックのためにディスプレイを用意し，入力操作用にマウスやキーボードを備える．音声インタフェースならフィードバックのためにスピーカを用意し，入力操作のマイクを備える．ユーザが利用できる操作方法は，インタフェース機器の特徴によって決まる．

ユーザインタフェース

ユーザインタフェースはインタフェース機器と，その上で動くソフトウェアによって構成され，サービスを操作を実現するシステムである．リモコンボタンとテレビに組み込まれた動作規則や，マウス・キーボード・ディスプレイと Win32API によるメーラの GUI などが例である．ユーザインタフェースはサービスに必ず最低一つ存在する．

モダリティ

モダリティは GUI，音声やジェスチャなど似た性質を持つ機器とソフトウェアを用いるユーザインタフェースを性質ごとにまとめて表す概念である．一つのモダリティは複数のユーザインタフェース形式を含む．例えば，GUI のプログラミングツールキットには Java の AWT(Abstract Window Toolkit)[17]，GTK[18]，Qt[19] などがある．

表 2.1: **インタフェース機器，ユーザインタフェース，モダリティ．**用語定義で説明したみっつの言葉の関係は以下のようになる．

インタフェース機器	ディスプレイ マウス	マイク スピーカ	...
ユーザインタフェース	GUI による AV 機器制御	音声による エアコンの制御	...
モダリティ	GUI	音声	...

2.2 組み込み型ユーザインタフェースモデル

ソフトウェアが存在する前は，ものの機能と形は不可分であった．遠くに矢を射たい場合は，弓につがえて弦を引き飛ばすしかなかった．組み込み型ユーザインタフェースモデルは，その関係をソフトウェアにも持ち込んだものである．組み込み型のユーザインタフェースは，サービスに付随して一対一で対応する．また，デスクトップコンピュータで利用される，ワードプロセッサやメディアプレーヤなどのアプリケーションは，それぞれ事前に作成された一つの GUI からしか利用できない．

組み込み型ユーザインタフェースモデルでは，サービスの数が増えるにつれ，ユーザインタフェースの数も同様に増加する．ネットワークでコンピュータが互いに繋がるとき，サービスの数は爆発的に増える．それに伴ってユーザインタフェースの数が増え，全ての利用方法を把握することは不可能となる．

2.3 中央集権型ユーザインタフェースモデル

中央集権型ユーザインタフェースモデルは，一つのユーザインタフェースが全てのサービスに対応し，一対多の関係をもつ．コンピュータのネットワーク化によるサービスの増加に対して，中央集権型ユーザインタフェースモデルでは，一つのユーザインタフェースで対応できる．例えば WWW(World Wide Web) を用いたユーザインタフェースはは，HTML というサービスの記述方式と HTTP という通信方式を守れば，Web ブラウザひとつで世界中のサービスにアクセスできる．また，PDA 上に家電の GUI を動的に合成して制御するというユニバーサルコントローラ [6] も中央集権型なユーザインタフェースの一例である．

中央集権型ユーザインタフェースモデルは，操作方法をあらかじめ想定したユーザインタフェースに限定するため，そのユーザインタフェースがなければサービスが利用できない．例えば WWW は，GUI を備えたデスクトップ型やノート型のパーソナルコンピュータがなければ利用できない．

2.4 協調型ユーザインタフェースモデル

協調型ユーザインタフェースモデルは，ひとつのユーザインタフェースでもひとつのサービスでもなく，ユーザインタフェースとサービスが多対多の関係にある．これは，インタフェースデバイスとして利用できる情報機器が遍在する，ユビキタスコンピューティング環境のためのユーザインタフェースモデルである．ユーザがいまいる環境から，目的のサービスに対するユーザインタフェースをユーザの環境に応じて動的に構成することで“いつでも・どこでも”コンピューティングが可能となる．

協調型ユーザインタフェースモデルは，インタフェースデバイスがどのように構成されるかによって，三つのタイプに分類できる．

2.4.1 n 権分立型ユーザインタフェースモデル

n 権分立型ユーザインタフェースモデルは，中央集権型ユーザインタフェースモデルが多数存在する状態である．ユーザは，複数存在する中央集権型のユーザインタフェースを，そのときの状況に応じてひとつ選択する．これにより，ユーザは常に最適のユーザインタフェースからサービスにアクセスできる．例えば，デスクトップコンピュータの前に座っているときは Web ブラウザを利用して，携帯端末を持って移動中はユニバーサルコントローラに移行する．

2.4.2 マルチモーダルユーザインタフェースモデル

マルチモーダルユーザインタフェースモデルは，中央集権型ユーザインタフェースの延長線上に存在する．ユーザは，ウェブブラウザや PDA といったソフトウェアではなく，より強く特定のインタフェースデバイス郡に拘束される．例としては，特定の会議室でのみ使える議事録システムなどである．その代わりに，音声や GUI を組み合わせた，利便性の高いユーザインタフェースが利用できる．

2.4.3 コンテキストアウェアユーザインタフェースモデル

コンテキストアウェアユーザインタフェースモデルは，n 権分立型とマルチモーダルの中間に存在するモデルである．n 権分立型モデルよりも多様なデバイスを利用してユーザインタフェースを構築するが，マルチモーダルモデルよりは環境に拘束されない．ユーザが直接利用するインタフェースデバイス以外に，それ単独ではユーザインタフェースを構成できないセンサを利用して，ユーザインタフェースの操作を補助する．

2.5　本章のまとめ

本章では，ユーザインタフェースとサービスの関係に基づいて，ユーザインタフェースモデルを三つに分類した．このうち，サービスとユーザインタフェースが多対多の関係となる協調型ユーザインタフェースモデルがユビキタスコンピューティング環境のユーザインタフェースとしてふさわしい．

次章では，協調型ユーザインタフェースモデルによるサービス利用のシナリオを示し，そこから要件を導き出す．

第3章

UIフレームワークの要件

本章では，前章で述べた協調型ユーザインタフェースモデルにおけるサービス利用のシナリオを示す．そして，そのシナリオを実現するために必要なフレームワークの要件を考える．

3.1　協調型ユーザインタフェースモデル

前章で述べたとおり，ユビキタスコンピューティング環境においては協調型ユーザインタフェースモデルが望ましい．

3.1.1　想定するシナリオ

協調型ユーザインタフェースモデルにおけるサービス利用のシナリオをふたつ述べる．

その 1

移動中のユーザはネットワーク上に存在するサービスに対して，携帯端末のユーザインタフェースからアクセスをする．その後，ユーザがオフィスに到着してより操作性の高いタッチパネルなどが利用できるようになると，ユーザはそこからサービスにアクセスする．

その 2

ユーザは街中を移動つつ，レストラン案内や交通情報の地図を利用する．それぞれの地図サービスを利用する際に，GPS の位置情報を用いて現在地が中心に地図が表示される．そのため，ユーザはまずすぐ近辺にある情報から優先的に調べることができる．

3.1.2　ユーザインタフェースフレームワークの要件

本節では，前節で示したシナリオをに実現するための，ユビキタスコンピューティング環境に適したユーザインタフェースの要件を述べる．

サービスのユーザインタフェース独立性

サービスのユーザインタフェース独立性とは，サービスの構築にあたり，サービスの機能とそれをアクセスするユーザインタフェースを分離することである．シナリオ 1 で，サービスのユーザインタフェースを変更したり，シナリオ 2 で特定の地図サービスに限らず位置情報のコンテキストをユーザインタフェースに適応するには，サービスのユーザインタフェース独立性が必要となる．

従来のテレビをサービスと考えるとリモコンというユーザインタフェース一対一で対応して，ユーザインタフェース独立性がない．そのため，例えばリモコンが見つからない場合，多くの操作ができなってしまう．

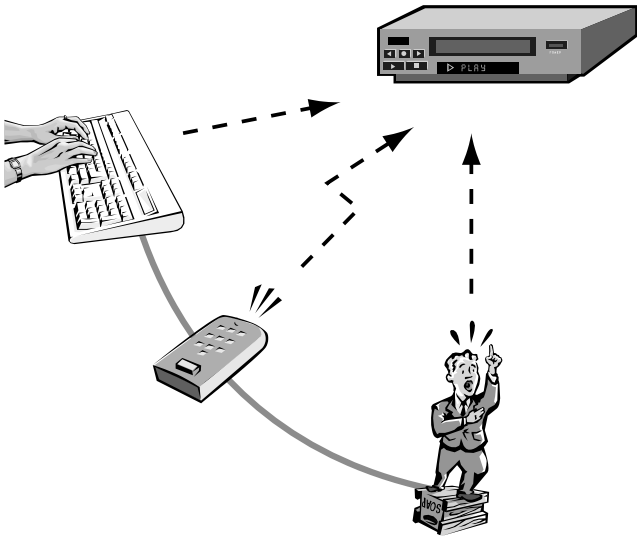

図 3.1: **サービスのユーザインタフェース独立性.**

操作の継続性

操作の継続性とは，ユーザがサービスのユーザインタフェースを切り替えても，以前利用していたユーザインタフェースで操作したところから操作を再開できることである．シナリオ 1 において，ユーザは屋外のインタフェースデバイスが少ない環境から，室内の豊富な環境に移動し，ユーザインタフェースがそれにともなって切り替わった．

ユーザは，状況が変わり，いままで使っていたユーザインタフェースが新しい状況に適さなければ，別のユーザインタフェースに切り替える．ユーザインタフェースを切り替える度に，初めから全ての操作を再度行う必要があれば，ユーザにとって負担となる．

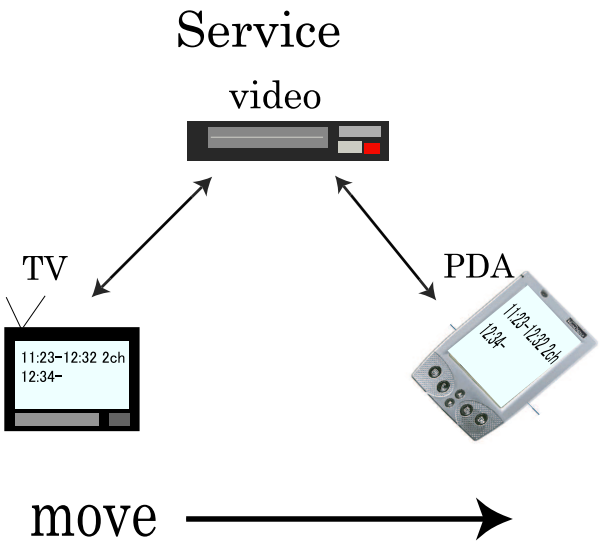

図 3.2: **操作の継続性.**

コンテキスト情報の利用

コンテキスト情報とは，ユーザが与えるコマンド以外に，まわりにあるセンサなどから取得できる情報である．シナリオ 2 では，GPS からの位置情報をコンテキスト情報として利用している．ユビキタスコンピューティング環境では，コンテキスト情報を利用したユーザインタフェースを利用することによって，ユーザの操作回数などが減り，利便性が高まる．

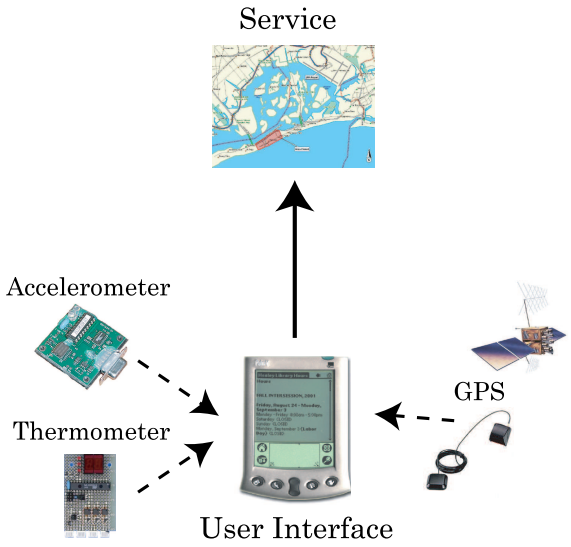

図 3.3: **コンテキスト情報の利用**.

3.2　ユーザインタフェース状態の分類

サービスとの依存関係から，ユーザインタフェース切り替えの際に継続する状態は二つに分類できる．

- サービス状態依存なユーザインタフェース状態
- サービス状態非依存なユーザインタフェース状態

3.2.1　サービス状態依存なユーザインタフェース状態

一部のユーザインタフェースの状態はサービスの状態を反映して変化する．例えばビデオデッキをサービスとして，GUI からの操作を考える．ビデオデッキにビデオカセットが入っていなければ再生はできない．そのため，GUI の再生ボタンはグレーアウトして使えない状態にある．ユーザインタフェースの「再生ボタンは使えない」という状態は，サービスの「ビデオカセットが入っていない」という状態から導くことができる．

3.2.2 サービス状態非依存なユーザインタフェース状態

一部のユーザインタフェースの状態は，サービスの状態とはかかわりなく変化する．ビデオデッキの例で録画予約をする場合，チャンネルや時間を設定する際のテキストフィールドの中身がこれに当たる．全ての項目を決定し，送信するまでビデオデッキの状態は変化しない．

3.3 先行研究

本節では前節で述べた必要要件に関連する先行研究を紹介する．

3.3.1 Document Based Framework

カリフォルニア大学バークレー校の Todd Hodes による Document Based Framework は，サービスの操作法をメソッドとその引数として XML で記述して抽象化し，ユーザの利用時にユーザインタフェースを生成する．図**??**のように，PDA 上のクライアントが，XML を読み込み，GUI を生成する．

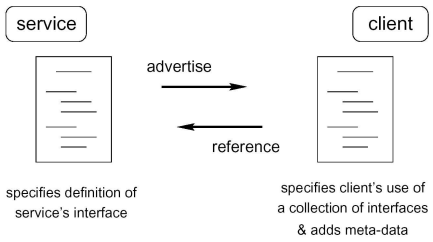

図 3.4: **Document Based Framework**.

Document Based Framework は，サービスを抽象化することでサービスのユーザインタフェース独立性を実現している．

3.3.2 HTTP Cookies

HTTP cookies[11] は HTTP サーバがクライアントに情報を保存し，後に取得する機構である．サーバはパスワード認証やオンラインショッピングのアイテムなどを，クライアントにクッキーとして送り，認証や決済の時にクライアントがクッキーを送りかえすことで，情報を復元する．

HTTP クッキーはクライアントの情報を保存し，復元することで，操作の継続性を実現している．しかし，HTTP という機構に限定されているので，サービスのユーザインタフェース独立性はない．

3.3.3 Voice as sound

五十嵐氏は，人間の声の言葉以外の要素を利用した，音声インタフェースのダイレクトマニピュレーションの研究を行っている [10]．従来の音声認識インタフェースは，会話による間接操作で，ユーザが支持をしてから，システムがそれを実行した．Voice as Sound では，声の高さや大きさといった要素にボタンやジョイスティックの役割を持たせる．

Voice as Sound は，従来の音声インタフェースを音声認識に以外の部分で拡張できる．つまり，コンテキスト情報の利用はできている．しかし，音声に限定しているためサービスのユーザインタフェース独立性はなく，また操作の継続性についてもとくに考慮していない．

3.3.4 EZWeb の簡易位置情報サービス

簡易位置情報サービスは，サービス側からの要求により，ユーザ同意の上で、携帯電話のの位置情報を送信する．これにより、ユーザの位置情報を元にしたサービスの提供が出来る．

サービスを提供する側で，HDML のなかに
`device:location?url=http://server/location.cgi` という URL を入れれば，ユーザがその URL を指定したときに，確認画面のあと位置情報が送信される．簡易位置情報サービスを利用した，宝探し等のゲーム [8] や，地図連動日記 [9] などのサービスが提供されている．

EZWeb の簡易位置情報サービスは，携帯電話に GPS を付属させ，サービスがその位置情報をアクセスことで，コンテキスト情報の利用が可能になっている．しかし，携帯電話に限定しているため，サービスのユーザインタフェース独立性はない．また，WAP では状態を保存するクッキーなどがないため，操作の継続性もない．

3.4 既存研究との比較

ここで，既存研究を 3.1.2 項で述べた用件と比較する．

表 3.1: ユーザインタフェースフレームワークの要件と関連研究の関係

	Document Based Framework	HTTP cookies	Voice as Sound	EZWeb
サービスの UI 独立性	△	×	×	×
操作の継続性	×	○	×	×
コンテキスト情報の利用	×	×	○	○

表 3.1 が示すように，ユビキタスコンピューティング環境におけるユーザインタフェー

スは既存の研究成果から構築できない．次章にて上で挙げた用件を満たすユビキタスコンピューティング環境におけるユーザインタフェースを設計する．

3.5 本章のまとめ

本章では，ユビキタスコンピューティング環境における，サービス利用のシナリオを提示した．そして，シナリオを実現するシステムの要件として，サービスのユーザインタフェース独立性，操作の継続性，そしてコンテキスト情報の利用を挙げた．次に，要件に関連したとして先行研究を Document Based Framework，HTTP Cookies，Voice as Sound，そして EZWeb を紹介した．

次章では要件と先行研究に基づき，ユビキタスコンピューティング環境のためのユーザインタフェースフレームワーク，i-face を設計する．

第4章

i-face設計

ユビキタスコンピューティング環境に適したユーザインタフェースフレームワークとして，**i-face(Intelligent interFACE)** フレームワークを設計する．

まず，**i-face** フレームワークの設計方針について触れ，概要，全体構成について述べる．その後，設計方針に基づいて**i-face** フレームワークの各機能を細説する．

4.1 設計方針

i-face の目的は，3.1.2 で示した用件を満たすユビキタスコンピューティング環境に適したユーザインタフェースフレームワークとして，協調型ユーザインタフェースモデルを実現することである．本節では，この目的を達成するための設計方針を 4 つ示す．それぞれ，1. サービスの抽象化，2. ユーザインタフェースの動的生成，3. ユーザインタフェース状態の保存・復元，4. センサの利用である．これらは 3.1.2 節で示した要件をもとにしている．表 4.1 に，双方の関係を示す

表 4.1: ユーザインタフェースフレームワークの要件と設計方針の関係

要　　件	方　　針
サービスのユーザインタフェース独立性	サービスの抽象化
サービスのユーザインタフェース独立性	ユーザインタフェースの動的生成
操作の継続性	ユーザインタフェース状態の保存・復元
コンテキスト情報の利用	センサの利用

4.1.1 サービスの抽象化

ユーザが利用するデバイスは状況に応じて変化するため，様々なユーザインタフェースから同じサービスを使えるように，特定のユーザインタフェースに依存しない形でサービスを抽象化する必要がある．

4.1.2 ユーザインタフェースの動的生成

ユーザインタフェースの動的生成とは，ユーザインタフェースをサービスの構築時に作成するのではなく，ユーザのサービス利用時にユーザインタフェースを動的に生成することである．これにより，サービス構築者はユーザがサービス利用時に使う可能性のあるユーザインタフェースの種類全てに予め対応しておく必要がなくなる．

4.1.3 ユーザインタフェース状態の保存・復元

ユーザがサービスを操作中にユーザインタフェースを切り替えた場合，操作の継続性を実現するためにはユーザインタフェースの状態を継続させる必要がある．ユーザインタフェース状態には二種類あり，それぞれの復元方法を**??** 節で示す．

4.1.4　センサの活用

センサの活用とは，ユーザのまわりにあるセンサを利用して状況を把握し，ユーザインタフェースの動作を補正することである．ユーザインタフェースにとって重要となるコンテキスト情報は，センサから得られる動的なものである．例えばGUIは，位置情報を利用して，表示するディスプレイを変える必要がある．

コンテキスト情報にはセンサから得られる情報以外にも，ユーザのスケジュールなど静的なものも含まれる．しかし，こういった情報はサービス側で対応するため，ユーザインタフェースは特に考慮する必要がない．

4.2　ユーザインタフェース状態の復元

本節では，3.2節に示したふたつのユーザインタフェース状態それぞれに対して復元方法を述べる．依存なユーザインタフェース状態は，サービス状態からの算出により復元する．サービス状態非依存なユーザインタフェース状態は，ユーザインタフェース状態を明示的に保存することにより復元する．

4.2.1　サービス状態からの算出による復元

ユーザインタフェースの「再生ボタンは使えない」という状態は，サービスの「ビデオカセットが入っていない」という状態から導くことができる．そのため，**サービス状態依存なユーザインタフェース状態**は，ユーザインタフェース の状態として明示的に保存せず，復元するときはサービスの状態から算出する．

4.2.2　ユーザインタフェース状態の明示的な保存による復元

サービス状態非依存なユーザインタフェース状態は，サービスの状態とは別に保存し，それを利用して復元する．また，複数のユーザが使えば，それぞれのユーザごとに状態が存在するため，**サービス状態非依存なユーザインタフェース状態**は，一つのサービスに複数存在しうる．例えば，二人が別々に録画を予約する場合，それぞれが入力している時刻やチャンネルは互いに異なる．そのため，**サービス状態非依存なユーザインタフェース状態**はユーザごとに保存する必要がある．

4.3　全体構成

4.1節で述べたように，ユビキタスコンピューティング環境のユーザインタフェースの要件は，サービスの抽象化，ユーザインタフェースの動的生成，ユーザインタフェース状態の保存・復元，そしてセンサの利用である．

このうちサービスの抽象化とユーザインタフェースの切替の考慮はユーザインタフェー

スの種類に非依存であり，ユーザインタフェースの動的生成とコンテキスト情報の利用は依存している．

4.3.1　ハードウェア構成

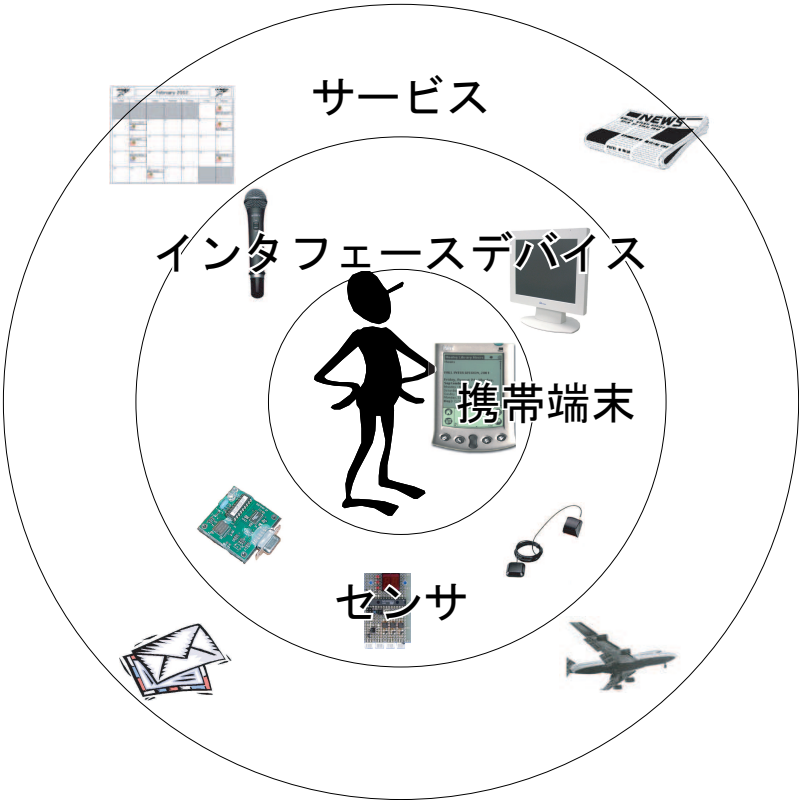

図 4.1: **ハードウェア構成**

i-face のハードウェアは図 4.1 の通りである，ユーザが持つ携帯端末，周辺に存在するインタフェースデバイス及びセンサ，そしてネットワーク越しにアクセスできるサービスで構成される．携帯端末は，ユーザが環境から環境へ移動する際にユーザインタフェース状態を保持する．インタフェースデバイスは，サービスを操作する際に，ユーザインタフェースを構成する要素となる．センサはユーザの状況を把握し，ユーザインタフェースを補助する．

4.3.2　ソフトウェア構成

i-face のソフトウェアは，ユーザが持つ携帯端末上で実行されるユーザインタフェース状態保存機構，インタフェースデバイスを構成要素としてユーザインタフェースを生成するユーザインタフェース生成機構，センサからの情報によりユーザインタフェースか

らのコマンドを補正するコンテキスト情報利用機構からなる．また，サービスを抽象化したサービス記述と，ユーザインタフェース状態保存機構に保存されるサービス状態非依存なユーザインタフェースを表したユーザインタフェース状態記述が定義されている．

4.4 ユーザインタフェース切り替えの基本動作

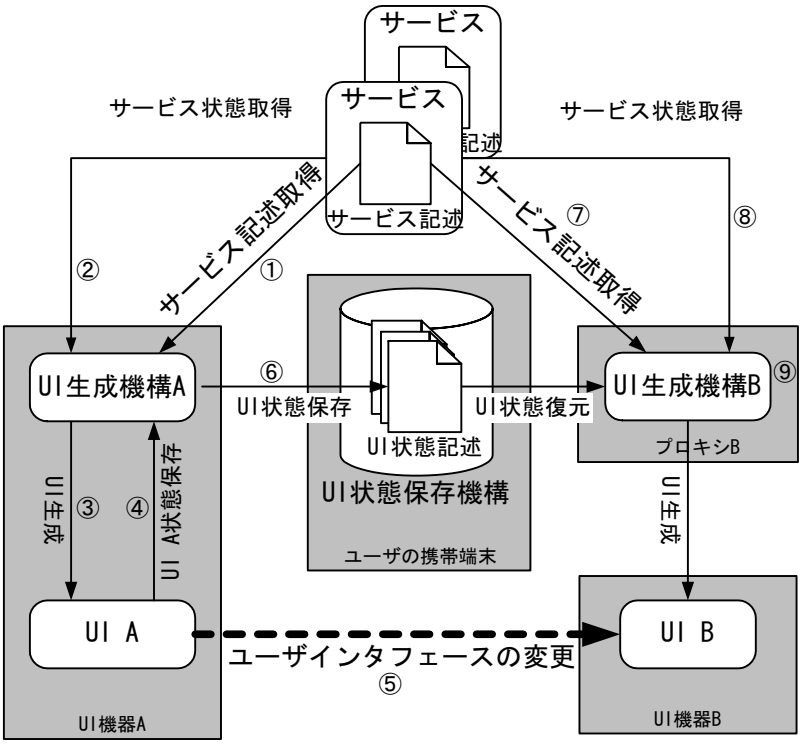

図 4.2: **ユーザインタフェース切替の基本動作**

図 4.2 に，ユーザインタフェース切り替えの基本動作を示す．

ユーザインタフェースの切り替えは，ユーザの状況が変化することによって利用できるインタフェースデバイスが変化して起こる．例えば，移動してディスプレイがない環境となった場合，音声インタフェースなど他のユーザインタフェースに切り替える必要がある．

ユーザはサービスを操作し始める際，最適なモダリティのユーザインタフェースを選択する．まず，選択された種類のユーザインタフェース生成機構は，ユーザのユーザインタフェース状態保存機構に以前のユーザインタフェース状態が保存されていないか問い合わせる．初期状態では保存されていないので，サービスの状態のみを反映してユーザインタフェースを生成しユーザに提供する．ユーザは提供されたユーザインタフェースから，サービスを操作する．

ユーザインタフェースを切り替えるときは，まず利用中のユーザインタフェースの操作を中断する．中断されたユーザインタフェースの状態のうち，サービス状態非依存な

部分をユーザインタフェース生成機構が取得し，ユーザインタフェース状態記述に変換する．さらにユーザインタフェース状態保存機構へ送られ，保存される．

次に新たなユーザインタフェースからサービスの操作を再開する場合，新しく利用するユーザインタフェースの生成機構は，ユーザインタフェース状態保存機構に保存されているユーザインタフェース状態記述を，生成したユーザインタフェースに反映する．最後に状態を復元されたユーザインタフェースが，ユーザに提示される．

4.5　ユーザインタフェース生成機構

ユーザインタフェース生成機構はHTMLやVoiceXMLといったユーザインタフェースの種類ごとに存在する．ユーザインタフェース生成部とユーザインタフェース状態部の二つの部分から構成されている．ユーザインタフェース生成部は，サービスのユーザインタフェースを，ユーザが利用開始するとき動的に生成する．ユーザインタフェース状態部は生成したユーザインタフェースにユーザインタフェース 状態を復元する．また，ユーザが操作を中断し，他のユーザインタフェースに移行するときは，サービス状態非依存なユーザインタフェース状態をユーザインタフェース状態保存機構に保存する．

4.5.1　ユーザインタフェース利用開始時の処理

ユーザインタフェース生成機構がユーザインタフェース利用開始時に行う処理の流れを図4.3に示す．

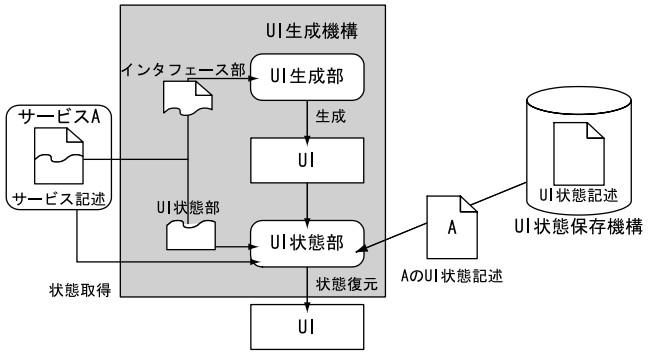

図 4.3: **ユーザインタフェース生成機構　利用開始時の処理**

ユーザインタフェース生成処理

サービス記述のインタフェース部からユーザインタフェース を生成する．例えばGUIなら各ボタンがサービス記述のインタフェース部のメソッドに対応し，フィールドやスライドバーがその引数に対応する．生成されたユーザインタフェースのボタンを押すとサービスにメソッドが発行される．

複数のサービスのインタフェース部から一つのユーザインタフェースを生成する場合もある．例えばテレビ電話で，こちらと通話先のカメラ両方を同時に操作したい場合は，二つのカメラサービスのインタフェース部を統合してユーザインタフェースを生成する．

ユーザインタフェース状態復元処理

生成したユーザインタフェースに，ユーザインタフェース状態を復元する．まず，サービス状態をサービスに問い合わせる．次に，サービス記述のユーザインタフェース状態部の条件に照らし合わせて，サービス状態依存なユーザインタフェース状態を復元する．そして，ユーザインタフェース状態保存機構に，このサービスのユーザインタフェース状態記述がないか問い合わせる．あれば取得しサービス状態非依存なユーザインタフェース状態を反映する．

4.5.2　ユーザインタフェース利用終了時の処理

ユーザインタフェース生成機構が，ユーザインタフェース利用終了時に行う処理を図 4.4 に示す．

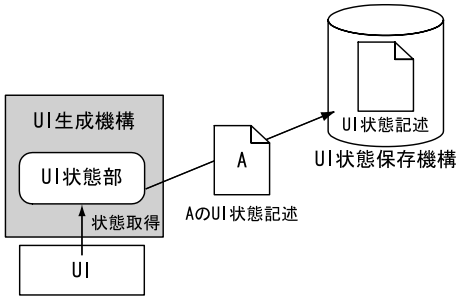

図 4.4: **ユーザインタフェース生成機構　利用終了時の処理**

ユーザインタフェース状態保存処理

ユーザが操作を中断したときは，ユーザインタフェースからサービス状態非依存なユーザインタフェース状態を取得する．送られてきた状態を**ユーザインタフェース状態記述**に変換し，ユーザが持っている端末上で動作している**ユーザインタフェース状態保存機構**に送る．

4.6　ユーザインタフェース状態保存機構

サービス状態非依存なユーザインタフェース状態を記述した，ユーザインタフェース状態記述を保存する．サービス状態非依存なユーザインタフェース状態は，ユーザごと

に異なり，環境を移動しても存続する．そのため，携帯電話などユーザが持ち運ぶ携帯端末上で実行される．

ユーザがユーザインタフェースを切り替えるときに，ユーザインタフェース生成機構のユーザインタフェース状態部か送られてくるユーザインタフェース状態記述を保存する．このとき，ユーザインタフェース状態記述はサービスごとに分けて保存される．ユーザインタフェース状態記述は，新しいユーザインタフェース生成機構に送るまで保持する．

4.7 コンテキスト情報利用機構

コンテキスト情報利用機構は，ユーザインタフェース生成機構とセットになり，コンテキスト情報を利用してより自然にサービスが利用できるようにする．

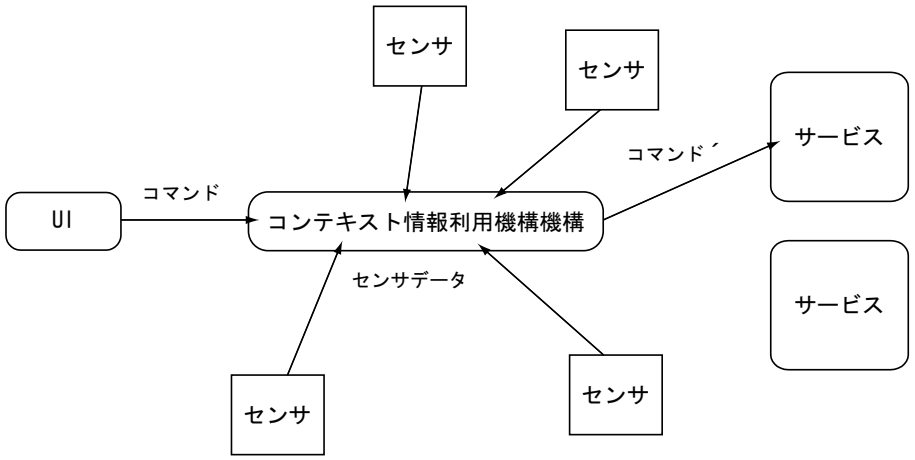

図 4.5: **コンテキスト情報利用基本動作**

基本動作を図4.5に示す．ユーザインタフェース生成機構が生成したユーザインタフェースをユーザが利用し，ボタンを押すなどしてコマンドを発行すると，その引数やあて先のサービスをコンテキストに応じて変更する．例えば，ユーザが音声インタフェースによって「ライトオン」とコマンドを出しても，部屋のどこにいるかによって点灯するライトを変更する．

4.8 サービス記述

サービスを抽象化するために，サービスの操作方法とユーザインタフェース状態復元のための情報を，ユーザインタフェースに依存しないドキュメントに記述する．

サービスの操作法は 3.3.1 項の Document-based Framework と同じく，メソッドとその引数で記述する．これによって，特定のユーザインタフェースに依存せず，ユーザインタフェースを構築するプログラムからサービスが操作できる．例えば「メソッド名：再生，引数：なし」のように記述される．これをインタフェース部と呼ぶ．

ユーザインタフェース切替時のサービス状態依存なユーザインタフェース状態の算出・復元のために，サービスの状態とそれによって決まるサービス状態依存なユーザインタフェースの状態を列挙する．まず，サービスの状態を判定するために，ある特定の状態を表す「変数名：値」ペアを条件として並べる．次に，判定されたサービス状態にから算出されるユーザインタフェース状態を並べる．これを，それぞれのサービス状態について繰り返す．これをユーザインタフェース状態部と呼ぶ．

サービス記述は，サービス構築者によって作成される．サービス側が保持し，サービス利用時にユーザインタフェース生成機構がネットワークを通じて取得する．

4.9 ユーザインタフェース状態記述

ユーザインタフェース状態の保存・復元のために，サービス状態非依存なユーザインタフェース状態を記述する．ユーザインタフェース状態は「メソッド名:引数名: 値」のように，サービス記述のインタフェース部にあるメソッドの引数の値として記述される．たとえば，Web ブラウザサービスを利用しているときに，ユーザインタフェースを切り替えると,「Go / URL：`http://www.google.co.jp/`」といった記述が保存される．

4.10 本章のまとめ

本章では，前章で示したユビキタスコンピューティング環境のためのユーザインタフェースフレームワークの要件に基づき，i-face の設計を行った．i-face は，ユーザインタフェース生成機構，ユーザインタフェース状態保存機構，そしてコンテキスト情報利用機構から構成される．また，サービス抽象化のドキュメントしてサービス記述を，サービス状態非依存なユーザインタフェース状態抽象化としてユーザインタフェース状態記述をそれぞれ定義した．

次章では，本章で行った設計に基づき，ユビキタスコンピューティング環境のユーザインタフェースフレームワークである i-face を実装する．また，音声インタフェースのコンテキスト情報利用機構である Earshot も実装する．

第5章

実装

本章では，**i-face**の実装について述べる．また，**i-face**フレームワークのなかの，コンテキスト情報利用機構のひとつ，**Earshot**の実装も説明する．

5.1 i-face

i-faceは，ユーザインタフェースの切り替えと作業の継続を考慮したユーザインタフェースフレームワークである．ユーザインタフェースを切り替えたときの作業の継続を可能にすることで，切り替え後に再設定する手間を省いた．また，ユーザインタフェースが自由に切り替えられることで，ユーザは状況に合ったモダリティによるユーザインタフェースでサービスを利用できる．

5.1.1 実装環境

i-faceのプロトタイプをJava言語で実装した．実装環境を表5.1に示す．

表5.1: i-face実装環境

項目	携帯端末	UI A	UI B		サービス
ハードウェア	Compaq iPAQ 3630		AMD K6-III	DV カメラ	
OS	Familiar Linux v0.5[23]		Vine Linux 2.1.5		
ソフトウェア	—		Apache Tomcat[24]		dvcont
実装言語	Java				

5.1.2 サービス

本節ではサービスの実装をDVカメラを例として用いた．

サービスサーバー

DVカメラサービスは，TCP/IPのサーバーとして実装した．ポートを指定して起動すると，そのポートに送られたコマンドに基づいてDVカメラを操作する．DVカメラとサーバーが動くPCは，IEEE1394で接続されている．

サービス記述

サービス記述は，汎用性を高めるためにXML(eXtensible Markup Language)[7]で実装した．XMLを扱うライブラリが多数の言語提供されているため，ポータビリティがある．また，XSL(eXtensible Stylesheet Language)を使うことで，XMLをベースとしたインタフェース(HTMLやVoXMLなど)への変換が容易に出来る．

4.8項で示した項目を元に，サービス記述のDTDを図5.1の通り定義した．

- ルートエレメントは`<service>`タグであり識別子`name`を属性として持っている．

```
<!ELEMENT service (interface, uistates)>
<!ELEMENT interface (label, (method+))>
<!ELEMENT label (#PCDATA)>
<!ELEMENT method (param*)>
<!ATTLIST method name ID #REQUIRED>
<!ELEMENT param EMPTY>
<!ATTLIST param lextype (int|real|boolean|enum|string) 'string'>
<!ELEMENT uistates (state+)>
<!ELEMENT state (condition*, enable*, disable*)>
<!ELEMENT condition (#PCDATA)>
<!ATTLIST condition key CDATA #REQUIRED>
<!ELEMENT enable (#PCDATA)>
<!ELEMENT disable (#PCDATA)>
```

図 5.1: サービス記述の DTD

- <service>タグはインタフェース部の<interface>タグとユーザインタフェース状態部<uistate>タグで構成されている．

- インタフェース部を示す<interface>タグには，サービスの名前を示す<label>タグが一つと，メソッドを記述した<method>タグが列挙されている．

- <method>タグは，それぞれユニークな name 属性をもっている．

- <method>タグの中には，それぞれの引数を示す<param>タグが，その引数の種類をあらわす lextype 属性を持ってリストされている．

- ユーザインタフェース状態部を示す<uistates>タグには，それぞれのサービスの状態を示す<state>タグを複数含んでいる．

- <state>タグには，その状態を示す条件を記述した<condition>タグと，そのサービス状態がサービス依存のユーザインタフェース状態にどのような変化をもたらすかを示す<enable>タグと<disable>タグで構成されている．

- <enable> タグは，その状態で使えるインタフェース部のメソッドを記述し，逆に<disable>タグには，その状態で使えないメソッドを記述する．

図 5.2 に，DV カメラサービスのサービス記述を例として示す．

インタフェース部にはサービスの名前が「DV Camera」であることと，四つのメソッドが記述してある．それぞれ，再生，停止，一時停止，そしてパラメータの値で速度が変わる再生である．

```
<?xml version="1.0"?>
<service name="dvcamera">
  <interface>
    <label>DV Camera</label>
    <method name="play"/>
    <method name="stop"/>
    <method name="pause"/>
    <method name="trickplay">
      <param name="speed"/>
    </method>
  </interface>
  <uistates>
    <state>
      <condition>
        <key>screen</key>
<value>Winding</value>
      </condition>
      <condition>
        <key>command</key>
<value>stopped</value>
      </condition>
      <enable>play</enable>
      <disable>stop</disable>
      <disable>pause</disable>
    </state>
    <state>
      <condition>
        <key>screen</key>
<value>Playing</value>
      </condition>
      <condition>
        <key>command</key>
<value></value>
      </condition>
      <enable>stop</enable>
      <enable>pause</enable>
      <disable>play</disable>
    </state>
    <state>
      <condition>
        <key>screen</key>
<value>Playing</value>
      </condition>
      <condition>
        <key>command</key>
<value>Paused</value>
      </condition>
      <enable>play</enable>
      <enable>stop</enable>
      <disable>pause</disable>
    </state>
  </uistates>
</service>
```

図 5.2: サービス記述例：DV カメラ

ユーザインタフェース状態部には，サービス依存なユーザインタフェース状態が三つ記述してある．最初の状態は，DV カメラが停止している場合で，ここでは再生しかコマンドが使えない．二つ目は再生中の場合で，再生は使えず，停止と一時停止が利用できる．三つ目が一時停止の状態で，一時停止以外のコマンドが利用できる．パラメータの値で速度が変わる再生は，三つの場合のいずれでも利用できる．

5.1.3 ユーザインタフェース生成機構

ユーザインタフェース生成機構を HTML と Java AWT の二つのユーザインタフェースのふたつ実装した．ユーザはサービスを利用する際，サービスの IP アドレスとポート番号，そしてユーザインタフェース状態保存機構の IP アドレスとポート番号を指定する．

サービス記述を解析するライブラリとして Xerces[25] を利用した．

HTML

Vine Linux 2.1.5 上で動く Apache httpd / Tomcat サーバー上に，JAVA Servlet として実装した．URL を指定すれば通常のウェブブラウザからアクセスできる．

初期段階では，サービスの IP アドレスとポート番号及びユーザインタフェース状態保存機構の IP アドレスとポート番号をしてする四つのフィールドと，HTML フォームの SUBMIT ボタンが一つ表示される．

フィールドに値を入力して SUBMIT ボタンを押すと，サービスからサービス記述を取得する．

サービス記述の XML より HTML に変換するために，XSL(eXtensible Stylesheet Language)[26] を利用した．XSL は XML ドキュメントのタグを他のものに置換するための言語である．これによって，サービス記述の各`<method>`タグが`<form>`タグに，各`<param>`が`<input type="text">`のテキストフィールドに変換される．

XSL によって HTML を生成した後，ユーザインタフェース状態保存機構からユーザインタフェース状態記述を取得する．ユーザインタフェース状態記述の内容に基づいて，`<param>`タグの `value` 属性に，値を代入する．

図 5.3 に DV カメラサービスのユーザインタフェースを生成したものを示す．

図 5.3: **HTML ユーザインタフェース生成機構**

Java AWT

Java Abstract Window Toolkit を利用した，GUI の生成機構を Sun JDK1.3 で実装した．

初期段階では，HTML の生成機構と同じくサービスの IP アドレスとポート番号及びユーザインタフェース状態保存機構の IP アドレスとポート番号をしてする四つのフィールドと，LOAD ボタンが一つ表示される．

値を入力して LOAD ボタンを押すと，サービス記述を取得する．

Java AWT の生成機構では，method タグごとに MethodButton クラスを生成し，初期の Frame に追加していく．MethodButton クラスは，パラメータを示す TextField クラスを格納する ArrayList を持った Button クラスのサブクラスである．MethodButton クラスの定義を図 5.4 に示す．

```
public class MethodButton extends Button{
    ArrayList params;
    int eventid = 0;

    public MethodButton(String name){
        super(name);
        params = new ArrayList();
        setActionCommand(name);
        setName(name);
    }

        ⋮
}
```

図 5.4: MethodButton クラス

MethodButton を追加したあとは，<param>タグごとに TextField クラスを生成し，Frame に追加するとともに上で生成した MethodButton クラスの ArrayList に追加する．

図 5.5 に DV カメラサービスのユーザインタフェースを生成したものを示す．

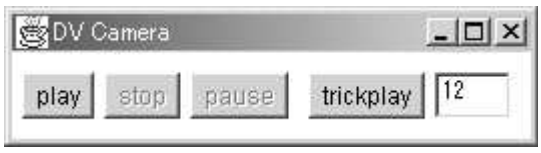

図 5.5: **Java AWT ユーザインタフェース生成機構**

5.1.4 ユーザインタフェース状態保存機構

ユーザインタフェース状態保存機構が動作する携帯端末として，Compaq 社が出している iPAQ シリーズを利用した．ユーザインタフェース状態保存機構は，TCP/IP のサーバーとして実装した．ユーザがユーザインタフェースを切り替えるとき，ユーザインタフェース生成機構はユーザインタフェース状態保存機構に接続し，「SAVE <サービス名>」のコマンドを発行した後，メソッド名，パラメータ名，そしてパラメータの値からなるユーザインタフェース状態を複数保存する．ユーザインタフェース状態の例を図 5.6 に示す．別のユーザインタフェース生成機構が，サービス記述を取得するときは，TCP/IP

で接続し,「LOAD <サービス命>」コマンドを発行すると，保存サービス状態記述が返される．

```
trickplay/speed:7
```

図 5.6: ユーザインタフェース状態記述の例

5.2 Earshot

Earshot は，音声のモダリティにおいてのコンテキスト情報利用機構である．センサからの情報を用いて，ユーザが操作するサービスを特定する．Earshot は，各サービスがユーザのコマンドに「聞き耳を立てている」というモデルでサービス特定を行っている．例えば小さい声で喋ると近くのサービスにしか聞こえず，大きな声で喋ると遠くのサービスまでコマンドが及ぶ．Earshot は「声が届くところ」という意味である．

5.2.1 実装環境

Earshot のプロトタイプを C 言語で実装した．Earshot の実装環境を図 5.2 に示す．実装はユビキタスコンピューティング環境の実験環境である，SSLab(Smart Space Laboratory)[21] で行った．

表 5.2: Earshot 実装環境

項目	UI	コンテキスト情報利用機構	サービス
ハードウェア	東芝音声ボード	PentiumIII 800MHz	DuonosPC，スポットライト
OS	—	Gentoo Linux 1.4rc1	FreeBSD 3.4-RELEASE
実装言語	C		

5.2.2 サービス

サービスは，SSLab に設置されているスポットライト (図 5.7) を使用した．これらのスポットライトは，電源制御により PC からコントロールできる．制御用 PC として duonos が常設され，TCP/IP サーバが動いている．サーバに TCP/IP で接続し，ASCII テキストを送ることで制御できる．

図 5.7: **SSLab のスポットライト**

5.2.3　ユーザインタフェース:音声インタフェース

ユーザインタフェースは，ユーザインタフェース生成機構を使用せず，東芝の音声認識ボードを利用した音声インタフェースを作成した．音声認識ボードは RS-232C シリアルケーブルで PC に繋がる．音声認識ボードには，ひらがなで単語を登録でき，その単語が認識されるとシリアルポートから単語の ID が読み取れる．

5.2.4　コンテキスト情報利用機構

Earshot はセンサからの情報とサービスの位置に基づいて，操作するサービスを絞り込む．例えば，ユーザが北の方角をむいてしゃべったときは，そちらのサービスに限定し，また小さい声でしゃべったときは近くのサービスに限定する．

図 5.8 に動作の原理を示す．まず，部屋に存在する全てのサービスから，コンテキスト情報によって段階的に絞っていく．最終的に残ったサービスに対して，ユーザが喋ったコマンドを発行する．

5.3　本章のまとめ

本章では，ユビキタスコンピューティング環境のユーザインタフェースフレームワークである i-face を実装した．また，音声インタフェースのコンテキスト情報利用機構である Earshot の動作を解説し，実装した．

次章では，本章の実装を利用して性能評価，及び関連研究との比較を通して議論を行う．

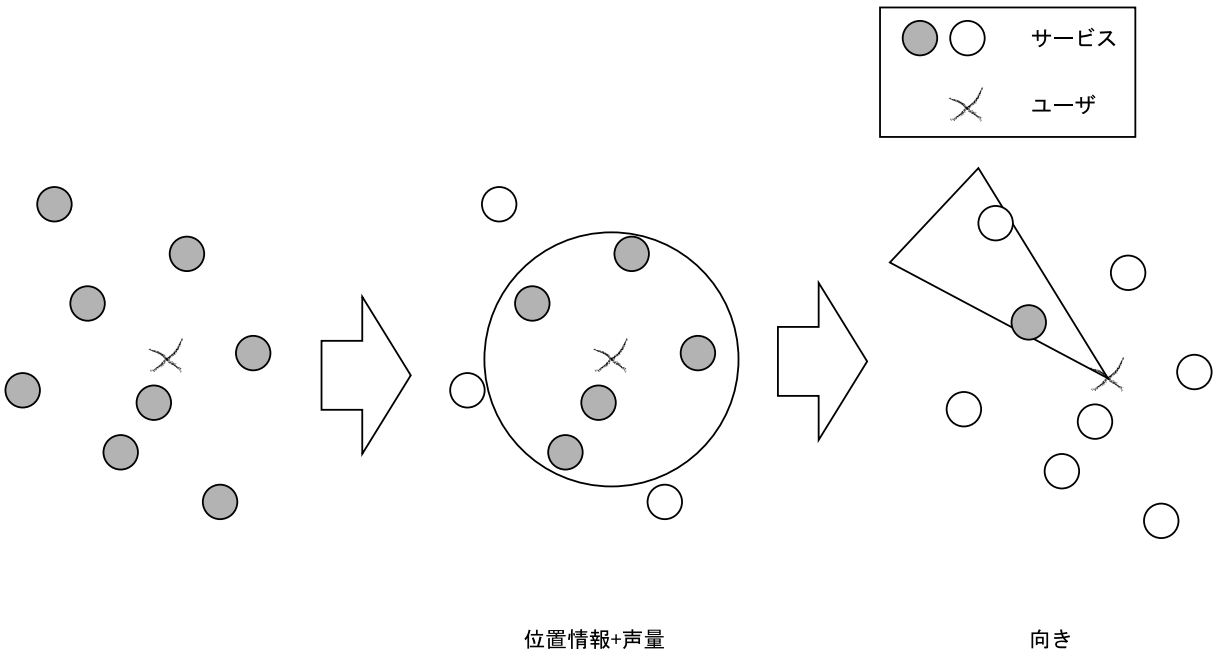

図 5.8: **Earshot: サービスの絞込み**

第6章

評価

本章では **i-face** の性能評価，そして関連研究との比較を通しての議論を行う

6.1 基本要件

本節では 3.1.2 項で述べたユビキタスコンピューティング環境におけるユーザインタフェースの要件を，実装したシステムがどの程度満たしているか検証する．

サービスのユーザインタフェース独立性

ユーザの移動などによって変化する状況に対して，柔軟にサービスに対してのユーザインタフェースを選択できることが目的であった．本論文では，ユーザインタフェースを利用時に動的に生成することにより，選択した UI の種類のユーザインタフェースを利用可能にした．また，サービスを抽象化して言語で記述することにより，特定にユーザインタフェースの実装に依存しないようにした．よってサービスのユーザインタフェース独立性は実現できたといえる．

操作の継続性

ユーザが異なるユーザインタフェースを利用する場合であっても，ユーザの作業状態を継続することが目的であった．本論文では，異なるユーザインタフェース間でもユーザインタフェース状態の保存・復元を可能にした．よって操作の継続性は実現できたといえる．

コンテキスト情報の利用

ユーザインタフェースの利用を，センサからのコンテキスト情報を用いて補強するのが目的であった．本論文では，音声インタフェースにおいて声の大きさを利用して操作するサービスの範囲を調整した．一つのセンサのデータから限定的にコンテキスト情報の利用は実現できた．今後の課題として位置情報など声量以外のコンテキスト情報の利用や，違うモダリティでのコンテキスト情報の利用が挙げられる．

6.2 ユーザインタフェース生成機構の評価

本節では，i-face のユーザインタフェース生成機構の性能の測定を実施する．これにより，実用性に耐えるか検証する．また，各処理の所有時間を計測し，改善するべき部分を洗い出す．

6.2.1 測定環境

測定は図 6.1 で示す環境で行った．ユーザインタフェース生成機構は，ユーザの携帯端末各マシンの仕様を表 6.1 に示す．

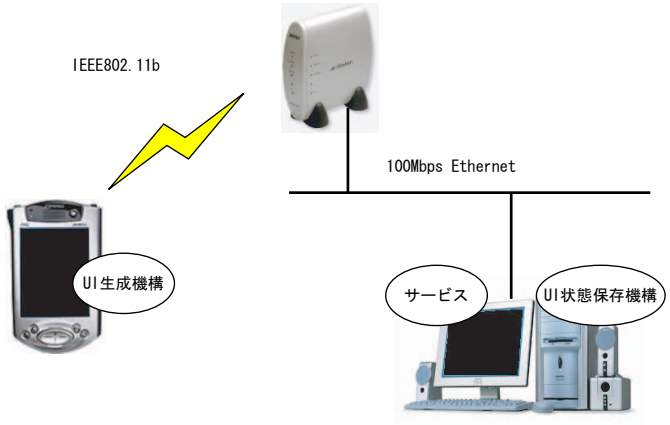

図 6.1: **測定環境**

表 6.1: 測定環境におけるマシンの仕様

	UI 生成機構	サービス 及び UI 状態保存機構
CPU	Intel StrongARM-1110 206MHz	Intel Pentium 4 1800MHz
メモリ	32MB	1.2GB
ネットワーク	IEEE 802.11b 11Mbps	100Mbps Ethernet
OS	Familiar Linux v0.5.3	Microsoft Windows XP

測定方法

Java AWT の UI 生成機構の各処理にかかる時間を測定した．サービスのメソッドの数を 10 と 20 の場合それぞれについて，200 回測定した値を平均した．

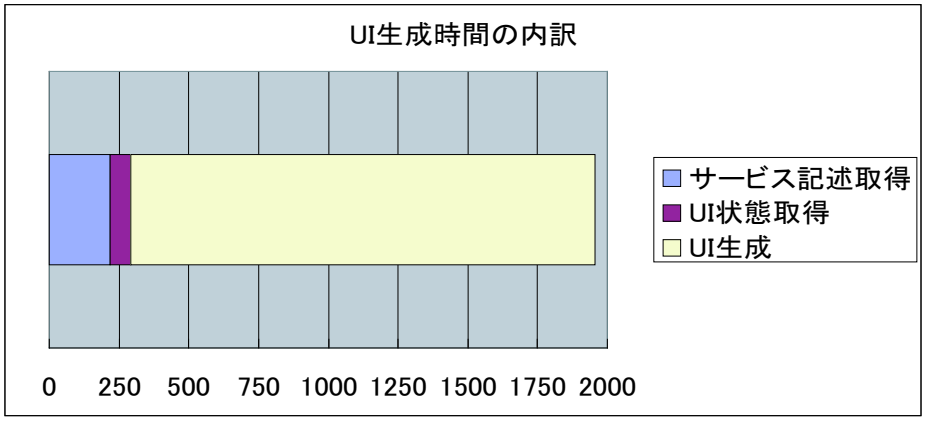

図 6.2: **UI 生成時間の内訳 メソッド数 10**

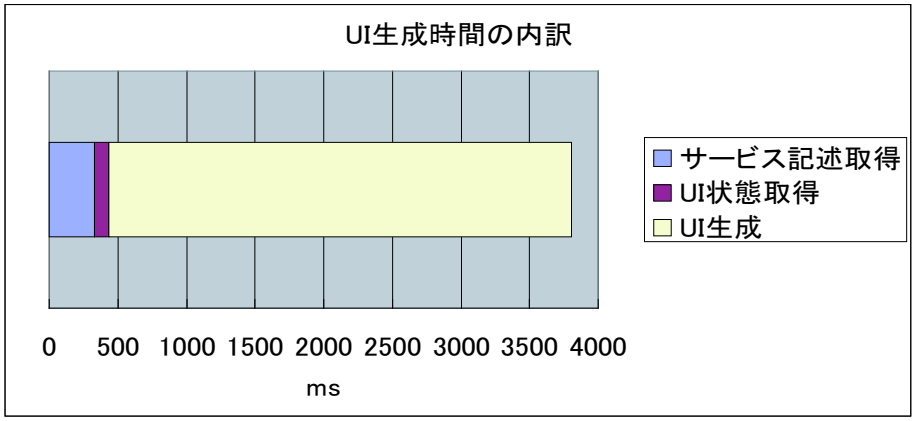

図 6.3: **UI 生成時間の内訳 メソッド数 20**

測定結果

測定結果を図 6.2 と図 6.3 に示す．処理時間の合計は，メソッド数に比例して増加している．このうち，ユーザインタフェース生成時間が 90%の時間を占めている．これは，オブジェクトの生成に時間がかかっているためである．事前にオブジェクトを生成しておくことによって，処理時間の大幅な短縮ができる．

6.3 関連研究との比較

本節では，3.1.2 節の要件に基づき他のユーザインタフェースフレームワークと本論文を比較する．

6.3.1 VNC: Virtual Network Computing

VNC[27] は画面に表示される画像と，キーボード及びマウス操作をネットワーク越しに転送することで GUI を備えた計算機を遠隔から操作可能にするものである．VNC は GUI のデスクトップ環境を前提に開発された．しかし，マイクロコントローラに乗せられる uVNC という [28] バージョンも存在しており，ユビキタスコンピューティング環境で広く利用できる．図 6.4 は VNC をつかってデスクトップコンピュータを遠隔操作しているものである．

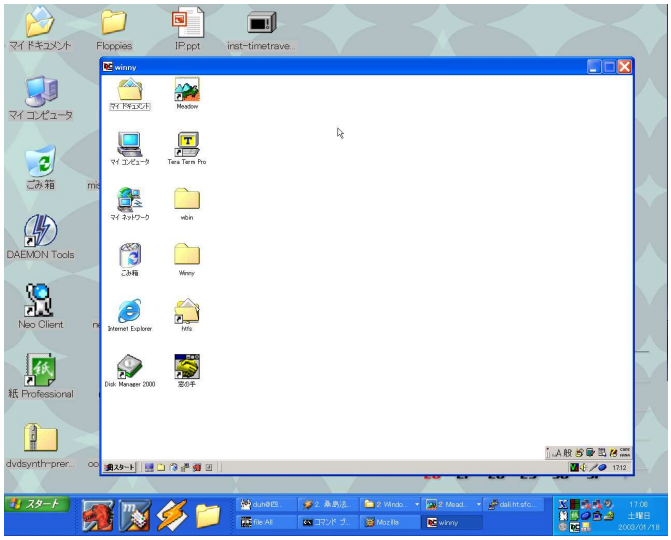

図 6.4: **VNC: Virtual Network Computing**

サービスのユーザインタフェース独立性

VNC は，サービスのインターフェースデバイスに縛られずに，ネットワークを通して遠隔操作を可能にする．uVNC ならば，デスクトップコンピュータでなくても遠隔から操作できる．しかし，システムが GUI を前提に構築されているため，音声やジェスチャなど他のモダリティは利用できない．

操作の継続性

VNC では，全てのステートはサービス側が保持している．クライアントはサービスから送られてきた GUI の映像を表示し，キーボードとマウスのイベントを送るという単純な動作をしている．そのため，操作の継続性は同じサービスを使い続ける限り保証される．

コンテキスト情報の利用

VNC ではコンテキスト情報は考慮されておらず，クライアントとサービスともにユーザが明示的に指定する．

6.3.2 Pebbles: PDAs for Entry of Both Bytes and Locations from External Sources

カーネギーメロン大学で研究されている Pebbles[29] は，ユニバーサルコントローラというアプローチの代表的なものである．普及している PDA や，これから出てくるプログラマブルな携帯を利用して，家庭やオフィスのサービスを全てコントロールしようというコンセプトである．図 6.5 は，プレゼンテーションサービスを iPAQ からコントロールするためのインタフェースである．

図 6.5: **Pebbles**.

サービスのユーザインタフェース独立性

Pebbles は PDA をインタフェースデバイスとして利用することを前提としている．PDA から様々なサービスを利用できるが，逆に，状況にあわせて UI を使い分けることはできない．そのため，サービスのユーザインタフェース独立性は低い．

操作の継続性

PDA という一つのインタフェースを想定しているため，ユーザインタフェースを切り替えた際の操作の継続性は考えられていない．

コンテキスト情報の利用

Pebbles ではコンテキスト情報の利用は考慮されていない．

6.3.3 ICrafter

ICrafter[30] はスタンフォード大学で研究されているユビキタスコンピューティング環境のサービスフレームワークである．ICrafter は，サービス利用のために UI の選択，生成，適応のためのインフラストラクチャを提供する．図 6.6 は，ICrafter による二つの部屋のライトサービスの GUI である．

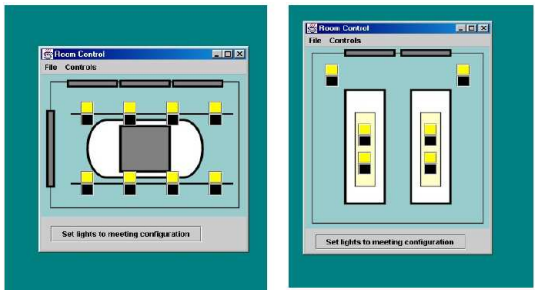

図 6.6: **ICrafter**.

サービスのユーザインタフェース独立性

ICrafter は既成のユーザインタフェースと，サービスの記述から動的に生成されるユーザインタフェースを組み合わせることでサービスのユーザインタフェース独立性を実現している．

操作の継続性

ICrafter にはユーザインタフェースの状態を保存し，別のインタフェースに復元する機構はないため，操作の継続性は考慮されていない．

コンテキスト情報の利用

ICrafter はある部屋に存在するサービスの種類など，静的なコンテキストは考慮している．しかし，センサを用いてユーザの動的なコンテキストを把握し，利用する仕組みはない．

6.3.4 比較結果

比較結果を表 6.2 に示す．i-face は，サービスのユーザインタフェース独立性，操作の継続性，そしてコンテキスト情報の利用の要件それぞれについて，関連研究よりも優れていることを示した．

表 6.2: ユーザインタフェースフレームワークの要件と関連研究の関係

	VNC	Pebbles	ICrafter	i-face
サービスの UI 独立性	△	△	○	○
操作の継続性	○	×	×	○
コンテキスト情報の利用	×	×	△	△

6.4 本章のまとめ

本章では，i-face を基本要件に照らし合わせて検証し，関連研究との比較を行った．また，ユーザインタフェース生成機構の性能を測定し，実用に耐えることを示した．

第7章

おわりに

本論文の最後に今後の課題を述べ，その後本研究をまとめる．

7.1 今後の課題

本節では今後の課題を，他モダリティのユーザインタフェース生成機構の作成，既存サービスの利用，そして他システムとの協調のみっつ述べる．

7.1.1 他モダリティのユーザインタフェース生成機構の作成

本論文のプロトタイプでは，Java AWT と HTML の二つのユーザインタフェース生成機構を実装した．しかし，どちらもモダリティが同じ GUI であるため，操作の継続性の検証が不十分である．音声やジェスチャなどのユーザインタフェース生成機構を実装し，今回実装したユーザインタフェース生成機構との間でのユーザインタフェース状態が復元できるか評価する必要がある．

7.1.2 既存サービスの利用

本論文のプロトタイプでは，いずれもサービスを i-face のフレームワークに合わせて新しく実装した．しかし，他の環境でも利用することを考えると，この全体は望ましくない．既存のサービスに対してもユーザインタフェースを生成し，操作できるようにラッパーなどを定義する必要がある．

7.1.3 他システムとの協調

i-face は，ユーザインタフェースのフレームワークである．そのため，サービスはつねに単独で扱うことを前提とした．しかし，例えばカメラサービスとプリンタサービスを結び付けて，カメラで撮った写真を出すといったように，複数のサービスの協調も考える必要がある．Wapplet フレームワークや VNA(Virtual Network Applicance) アーキテクチャによって合成したサービスに対しては，ユーザインタフェースの合成を行う必要がある．

7.2 まとめ

本論文では，初めにユビキタスコンピューティング環境は，情報機器の遍在と，それらを互いに繋ぐネットワークの存在が特徴となると述べた．そして，そこでのユーザインタフェースのモデルに，既存の組み込み型や中央集権型ではなく，新しい協調型モデルを提案した．協調型ユーザインタフェースモデルでは，サービスとユーザインタフェースの関係が多対多であり，そのときユーザがいる環境のインタフェースデバイスからサービスを操作できる．このモデルを実現するためのフレームワークとして，i-face を設計した．

i-face は，サービスをサービス記述で抽象化し，ユーザインタフェースを動的に生成

することで，サービスのユーザインタフェース独立性を実現する．また，ユーザインタフェース状態を二つに分類し，それぞれについて復元方法を実装し，ユーザがユーザインタフェースを切り替えた際にも操作の継続性を実現する．そして，センサ情報を利用して，ユーザが発行するコマンドを調整することで，コンテキスト情報を活用したユーザインタフェースの動作を実現する．

そして，i-face を実装しそのユーザインタフェース生成機構について性能評価を行った．ユーザインタフェースの生成時間は，サービスの複雑さに応じて線形に増加し，実行時間は実用の範囲内に収まることを示した．また，他のユーザインタフェースフレームワークとも比較し，協調型ユーザインタフェースモデルを実現するシステムとして優れていることを示した．

本研究によって，ユビキタスコンピューティング環境における協調型ユーザインタフェースモデルが実現できた．ユーザは状況に合わせて最適なユーザインタフェースを利用できるようになった．

謝辞

本研究を進めるにあたり，御指導を頂きました，慶應義塾大学環境情報学部教授徳田英幸博士に深く感謝致します.

また，貴重な御助言を頂きました慶應義塾大学政策・メディア研究科の西尾信彦博士のご指導に感謝します．新天地でも持ち前のパワーで研究を推し進めて下さい.

病める時も健やかなる時もともに歩み，一方的に支えて下さり，+激しく叱咤激励+ を頂きました慶應義塾大学政策・メディア研究科の権藤俊一氏および永田智大氏に感謝の意を表します.

村瀬正名氏，榊原寛氏，米澤拓郎氏，米山遼太氏，小泉健吾氏をはじめとする慶應義塾大学徳田研究室 move!研究グループの諸氏，また徳田・村井・楠本・中村研究室の方々には，研究会の活動を通じて多くの御意見，御助言を頂きましたこと拝謝します.

最後に，研究の日々を共に過ごした，鈴木源太氏，門田昌哉氏，青木俊氏，高橋元氏，志和木愛子氏，村上朝一氏，峰松美佳氏，中山裕佳子氏，田丸修平氏，滝澤允氏，高橋ひとみ氏，青木基衣氏その他多くの友人に深謝し，謝辞と致します.

2003 年 1 月 22 日

守分 滋

参考文献

[1] プレイステーション 2、初回販売台数は 98 万台
http://www.watch.impress.co.jp/pc/docs/article/20000307/ps2.htm

[2] 2006 年国内携帯電話市場は 8369 万人、7.8 兆円に・ガートナージャパン予測
http://www.computerworld.jp/contents/free/200205/20020515gartner.html

[3] Weiser, M.: The Computer for the 21st Century, *Scientific American*, Vol. September, pp. 94-100(1991).

[4] Mickey Mouse goes wireless
http://www.cnn.com/2001/TECH/ptech/11/27/wireless.mickey.idg/index.html

[5] Wavelength Division Multiplexing (WDM)
http://www.lightreading.com/document.asp?doc_id=3109

[6] Nichols H. and Myers B.A.:Studying The Use of Handhelds To Control Everyday Appliances. Submitted for publication, citeseer.nj.nec.com/443862.html(2000)

[7] W3C, Extensible Markup Language (XML),
http://www.w3.org/XML/

[8] GPS/簡易位置情報サービスを使った EZweb 向け宝探しゲーム
http://k-tai.impress.co.jp/cda/article/news_toppage/0,,12223,00.html

[9] ライコスジャパン、「EZweb」GPS ケータイ対応の地図連動型日記の課金サービス『Lycos あしあと日記』を本格始動！
http://www.lycos.co.jp/help/info/release.html?press=185

[10] Takeo Igarashi, John F. Hughes. ”Voice as Sound: Using Non-verbal Voice Input for Interactive Control ” 14th Annual Symposium on User Interface Software and Technology, ACM UIST’01, Orlando, Florida, November 11-14, 2001, pp.155-156.

[11] PERSISTENT CLIENT STATE HTTP COOKIES
http://wp.netscape.com/newsref/std/cookie_spec.html

[12] <CoCoon> チャンネルサーバー CSV-E77
http://www.sony.jp/products/Consumer/cocoon/CSV-E77/index.html

[13] C413S
http://www.au.kddi.com/phone/cdmaone/c413/c413.html

[14] 製品：Video Blaster WebCam Pro：製品紹介
http://japan.creative.com/products/pc_cameras/webcampro/

[15] アルファ・オメガソフト ／ WebCam Pro
http://www.alphaomega.co.jp/webcam/pro/index.html

[16] VoiceXML Forum
http://www.voicexml.org/

[17] Java 2 Platform SE v1.3.1: Package java.awt
http://java.sun.com/j2se/1.3/docs/api/java/awt/package-summary.html

[18] GTK+ - The GIMP Toolkit
http://www.gtk.org/

[19] Trolltech - Qt - Overview
http://www.trolltech.com/products/qt/

[20] Palm Products - Handhelds
http://www.palm.com/products/handhelds/

[21] 慶應義塾大学徳田研究室 SSLab(Smart Space Laboratory) プロジェクト (2002).
http://www.ht.sfc.keio.ac.jp/SSLab/

[22] Handheld Productivity: iPAQ 3765 PocketPC
http://athome.compaq.com/showroom/static/ipaq/3765.asp

[23] the Familiar Project
http://familiar.handhelds.org/

[24] The Apache Jakarta Project
http://jakarta.apache.org/tomcat/

[25] Xerces2 Java Parser Readme
http://xml.apache.org/xerces2-j/index.html

[26] The Extensible Stylesheet Language
http://www.w3.org/Style/XSL/

[27] Tristan Richardson, Quentin Stafford-Fraser, Kenneth R. Wood, and Andy Hopper. Virtual Network Computing. *IEEE Internet Computing*, Vol. 2, pp. 33-38, 1998.

[28] uVNC - A VNC Server for 8-bit Microcontrollers
http://www.sics.se/~adam/uvnc/

[29] Brad A Myers., Using Handhelds and PCs Together. *COMMUNICATIONS OF THE ACM*. November 2001/Vol. 44, No. 11.

[30] Ponnekanti S.R., Fox A., Hanrahan P., and Winograd T.: ICrafter:A Service Framework for Ubiquitous Computing Environments, *UbiComp 2001*(2001).