ユビキタス環境における家電制御ミドルウェア

The middle ware for controlling home appliances in ubiquitous environment.

芥川　友義 [1)]　伊部　浩章 [2)]　鈴木　寛 [3)]　服部　貴章 [4)]　山本　紘司 [5)]
Tomoyoshi AKUTAGAWA　Hiroaki IBE　Hiroshi SUZUKI　Takaaki HATTORI　Koji YAMAMOTO

1) 早稲田大学理工学部情報学科　中島研究室（〒169-8555 東京都新宿区大久保３－４－１
早稲田大学理工学部　５５－５０５　E-mail: Aku@dcl.info.waseda.ac.jp ）
2) 早稲田大学理工学部　中島研究室（E-mail: ibe@dcl.info.waseda.ac.jp ）
3) 早稲田大学理工学部　中島研究室（E-mail: suz@dcl.info.waseda.ac.jp ）
4) 早稲田大学理工学部　中島研究室（E-mail: hattori@dcl.info.waseda.ac.jp ）
5) 早稲田大学理工学部　中島研究室（E-mail: koji@dcl.info.waseda.ac.jp ）

ABSTRACT. Although ubiquitous environment is progressing, since the various remote control for controlling many household electric appliances spoiled a user's convenience, we unified apparatus control and built the system which changes the optimal control device dynamically according to a situation in the field of home computing.

１．背景

我々の生活はマイクロプロセッサの組み込まれた機器,そして世界中を網羅するネットワークの普及によって大きな変化を迎えようとしている.携帯電話の爆発的な普及は先端の情報機器が数年であたりまえのものになるという可能性を示した.このようないたるところにコンピュータが組み込まれるような環境をユビキタス環境という.このユビキタス環境は今後より一層進んでいくものとみられる.

このような環境ではいったいどういったことが考えられるか.その一例としてホームコンピューティングを挙げる.従来の家電製品は,ユーザは機器を制御するためリモコンを使用している.この場合,それぞれの機器は異なったリモコンを必要とし,またこれらのリモコンは多種多様な機能を提供するための沢山のボタンを持っている.しかし,ユビキタス環境ではさらに多くの機器にコンピュータが組み込まれ,その機能はさらに多様化することが考えられる.このような状況ではそれぞれの機器に対して異なった制御デバイスがあるというのはユーザの混乱を招き,非常に扱いづらいものとなってしまう.
このようにユビキタス環境においてユーザインタフェースは非常に重要である.

では前述の状況に対応するためには,どうすればいいのか.まずユーザインタフェースを一元化することが必要である.PDA や携帯電話などの携帯デバイスでこれらの組み込み機器を一元化して制御できれば複数のリモコンに迷わされる必要が無くなる.また共通したユーザインタフェースは,新しい機器に対しても今までと同じ使い勝手を提供することができる.しかし,ユーザインタフェースを一元化しただけでは,使い勝手がいいとはいえない.多種多様なデバイスが存在する中でユーザにとって最適なデバイスが何かということをわかる必要がある.例えば机に向かっている時は画面の見やすい PC を使って機器を制御することが最適であるかもしれないが,移動しながらでは PC を使うことはできないので携帯デバイスが最適なデバイスとなる.そして,動的にそのデバイスに切り替わるような仕組みがあればユーザは意識しないで最適なデバイスを使うことができる.

このようなことが実現できれば,将来に起こるであろう機器の増加や多機能化にも,ユーザは何の苦労もなくすんなりと新しい機器の恩恵を受けることができる.

２．目的

本提案のシステムにより,前項で述べたシナリオが実現される.例えば家電製品を制御するアプリケーションを本システム上で動作させると,ユーザは家電製品を携帯などの様々な入力デバイスで操作することができる.そして,状況認識により隣の部屋に移動すると別の入力装置から同じように家電を制御することができる.

では本システムを用いた場合の具体例を挙げる.
本システム上でテレビを見ることができるアプリケーションを走らせている場合を考える.部屋Ａでは,PC を使ってこのアプリケーションを制御し

そのディスプレイに出力している.通常の使用では,ユーザはテレビを見終えるまで PC の前を離れることはできない.しかし,本システム上でこのアプリケーションを起動した場合には,そのテレビアプリケーションはコンテキストアウェア情報を使うことができるアプリケーションになる.

移動中はエアボードに出力デバイスを切り替える.それはロケーション情報やデバイス情報によりエアボードがユーザの移動に対応でき,携帯できるデバイスの中で画像を見るのに最も適していると判断したからであるからである.また移動先の部屋Ｂではその部屋にあるディスプレイに出力先を切り替える.ユーザの移動が終了し,その部屋にディスプレイが存在するならより綺麗に見れるディスプレイに切り替わるほうが良い.しかも,このシステムでは前述したようなアプリケーションと入出力の完全な分離をしているので,もとのテレビアプリケーションには変更は必要ない.

３．ミドルウェア本体

（１）システムの概要

我々のシステムは一言で言えば、家電アプリケーションサーバとそのユーザインタフェースを提供するミドルウェアである。

（２）システムの特徴

a)アプリケーションの入力と出力を完全に分離することが可能である.

既存のアプリケーションでは,入力デバイスや出力デバイスはアプリケーション本体と密接につながっている.ある PC 上のアプリケーションはその PC のキーボードやマウスを入力デバイスとし,その PC のディスプレイを出力デバイスとする.しかし,本システムではキーボードやマウスの動きを他のデバイスにマッピングすることにより入力デバイスを分離させ,ビットマップデータや音声データを制御することにより出力デバイスを分離させることを可能にしている.

b)既存のアプリケーションを変更せずに様々な先端デバイスからの制御を可能とする.

１で述べたように従来のアプリケーションでは入出力デバイスがアプリケーションと密接に結びついているので,新たなデバイスに対応させる場合にはアプリケーションの大幅な変更が強いられる.だが本システムでは,入出力デバイスがアプリケーションから分離しているのでこれから開発されるであろう新たなデバイスを入出力デバイスにすることを可能にする.

c)入出力機器はユーザの状況に応じて動的に変更可能.

本システムでは,ロケーション情報とデバイス情報を認識する機能を持つ.この情報によりユーザの状況を把握し,どのデバイスがユーザにとって最適かを判断する.そのデバイスに入出力を切り替えることを可能にする.

d)既存のアプリケーションをコンテキストアウェアなアプリケーションにすることができる.

コンテキストアウェアなアプリケーションを作成することは大きなコストがかかる.例えば,ロケーション情報を得るためのセンサを制御する仕組み,デバイスを発見するための仕組み,そのデバイス情報を保持するためのデータベース,ネットワークへの対応など様々な機能が必要になる.既存のアプリケーションをコンテキストアウェアなものにする場合にもこういった機能を追加する必要があり,プログラムの大幅な変更が強いられる.本システムでは,コンテキストアウェアに必要な機能をより下のレベルで提供することにより,アプリケーションに変更を加えることなく,コンテキストアウェアなアプリケーションにすることができる.

（３） システムの設計

本システム(Universal Interaction System, 以下 UnIt System)は以下のコンポーネントから成る(図 2 参照)

- UnItServer
- UnItProxy
- UnItInput/UnItOutput
- ContextManager
- UnItSpider

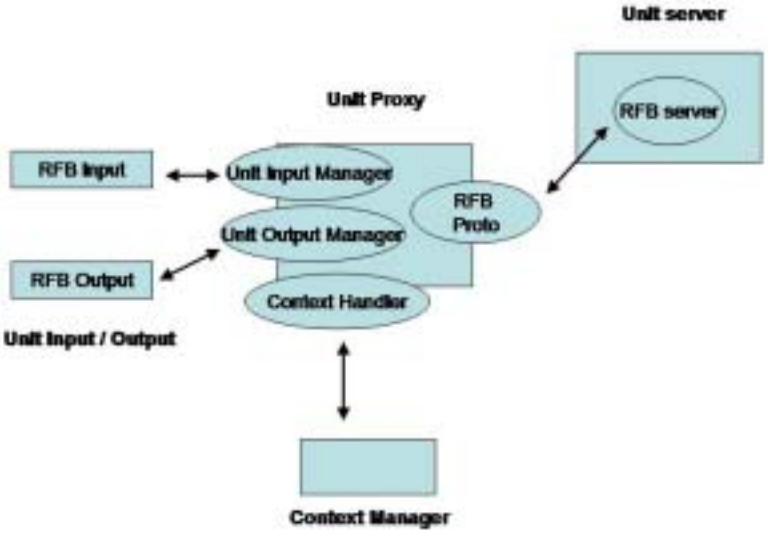

図 2 Universal Interaction (UnIt) System

本システムでは,アプリケーションは UnItServer 上で動作する.UnItServer は Windows,Unix,Macintosh 等多くの OS に対応しており,それぞれの OS 上で動作するアプリケーションをそのまま動かすことが出来る（本システムの為にスクラッチから作成したり,手直しをする必

要は全く無い）.
アプリケーションは,Xwindow のような既存のユーザインタフェースシステムにより GUI を実装する.ユーザインタフェースシステムによって生成されたビットマップイメージは,ユーザの位置に応じて適切な出力デバイスに出力される.
一方,キャプチャされたインプットイベント（マウス,キーボードイベント）は,UnItServer 上で動作しているアプリケーションに送られる.
音声も同様に UnItSoundServer 上でアプリケーションがはいた音声はデータはネットワークを経由して適切なデバイスに出力される.また,マイクによりキャプチャされた音声は UnItSoundServer でリダイレクトされ様々な音声デバイスに出力される.これに関しては第何章の hogehoge で後述する。
UnItProxy は本システムで最も重要なコンポーネントであり,各コンポーネントと通信し,I/O デバイスの管理,切り替え,イベントや画像の変換などを行なう.
ContextManager は I/O デバイスの状態を把握し,デバイスの切り替えを UnItProxy に指示する.無線 RFID センサ（spider）を使うことでユーザの位置情報を検出し,適切な場所のデバイスを判断し Proxy に指示する.
次節で,これらのコンポーネントの詳細を説明する.

本システムは,ユーザのデスクトップ環境（マウス,キーボードから入力してディスプレイに GUI を出力する）を,ネットワークを通じて拡張する（ネットワーク越しの Device を用いて入出力できる）ことができる.
UnItProxy は Java で実装される.また,UnItInput/Output, ContextManager も Java で実装される.そのため,Java の実行環境が存在する様々な OS,デバイス上で動作させる事ができる.

（4）UnItProxy コンポーネント
UnItProxy は UnIt システムの中心部に位置し,各コンポーネント間のゲートウェイとして,様々な機能を持つ.UnItProxy は図１で示されるモジュールを含んでいる.
RFB Manager モジュールは RFB Input デバイスからの入力イベントを RFB プロトコルに変換して UnIt Server に送る.また,UnItServer から送られてきたビットマップイメージを Output デバイスに応じたビットマップイメージに変換し,送信する.
SoundProxy は同様に UnItSoundServer と Sound I/O との仲立ちを行い,アプリケーションへの入出力を担う.
I/O Manager は UnIt I/O,のプラグアンドプレイ接続や切断を感知し,**DeviceDatabase に登録する.また, ContextManager からのメッセージを待ち受け,**指示された Device を RFB Manager や Sound Proxy に通知する.
DeviceDatabase は接続されているデバイス情報を保存する.

（5）UnItServer コンポーネント
UnItServer はアプリケーションの動作を担当する.いくつかの仮想画面を持ち,それぞれの仮想画面内で複数のアプリケーションを動作させる.RFB Server と Sound Server の二つのモジュールより構成される.RFB Server は RFB プロトコルを実装し,UnItProxy との間でビットマップイメージ,インプットイベントを交換しあう.

（6）UnItInput / Output コンポーネント
このコンポーネントとしては RFB Input/Output, Sound Input/Output がある.RFB Input/Output コンポーネントは各デバイス上で動作し,**アプリケーションへの入出力を行なう.RFB Input はユーザの入力命令をキャッチして,RFB プロトコルにより Proxy に命令を伝達する.RFB Output は Proxy より送られてきた GUI イメージをデバイス上に出力する.このイメージは,RFB Output の持つデバイス情報により,デバイスごとに適切な形に変換されて Proxy より送られてくる.**

（7）ContextManager コンポーネント
ContextManager は UnItSpider と通信することでユーザやデバイスの位置情報を取得し,最も適切なデバイスへの切り替えを UnItProxy/SoundProxy に指示する.ContextManager は内部に入出力デバイスの情報データベースを持ち,これにより最も適切なデバイスを判断する.UnItProxy/SoundProxy との通信は ContextManager プロトコルにより行われ,入出力デバイスの情報や切り替えの指示をやりとりすることができる.

（8） UnItSpider コンポーネント
UnItSpider は、無線 RFID センサ“ spider ”とシリアル経由で通信する。各 UnItSpider は、範囲内にあるタグの ID をそれぞれ取得する。ユーザ（ユーザのタグ）をセンサの範囲内に検出すると、現在範囲内にある全てのタグ情報を ContextManager に送信する。

４．アプリケーション

（1）UPnP による家電制御用システムの概要
本システムは UPnP を用いた家電制御デモ用のアプリケーションで、家電を制御するコントロールポイントのプログラムと、家電の具体例として X10 で制御できるライトのプログラムとＴＶを赤外線で制御するためのリモコンのプログラムから

構成されている。このシステムにより、ライトとテレビの２つの家電をコントロールポイントのＰＣから制御することができる。

（２）Jini による家電制御用システムの概要
クロッサム 2+へのコマンドインタフェースを Jini の Lookup Service に登録し、それを利用して TV の制御を行なう。Jini サービスを監視するアプリケーションは Jini Lookup Service を監視し、利用可能なサービスを表示させる。

５．開発成果の特徴

（１）特徴

a)レガシーサポート
これまでに様々なソフトウェア（プログラム）が開発されてきた.しかし,それらのほとんどが各アーキテクチャに依存しており,新しいアーキテクチャが出現すると新しく作り直さなければならないといったことがおこる.当然,これからも新しいデバイス,OS などがどんどん開発されていくだろう.このことは新しい製品を開発する上で余計な時間も費用も費やすことになる.上記のように本システムはあらゆるアーキテクチャ上のあらゆる OS 上で動作し,既存のアプリケーションを変更せずに使用することができる.このことで既存の膨大なソフトウェア資源を有効に活用できる

b)コンテキストアウェアネス
本システムは各ユーザの位置情報やデバイスの位置情報,デバイスの詳細情報などを保持している.既存のコンテキストを意識しないで開発されたソフトウェアに本システムを適用することでコンテキストアウェアなソフトウエアとして使用できる.

c)ユーザインタフェースとアプリケーションの分離
アプリケーションとユーザインタフェースを完全に切り分けることができる.またそのことで新しいユーザインタフェースの出現に柔軟に対応することができる.

d)シンクライアント設計
これからのユビキタス環境を考えると,PDA や携帯電話を代表とするマシンパワーの貧弱な先端デバイスでシステムを動作させる必要がある.これを満たすために本システムはシンクライアント設計になっている.デバイスに対するリソース要求が少ないため,簡易なデバイス上での使用も容易である.

e)マルチメディアサポート
本システムはリモートマシン上の画像や音声といったマルチメディアデータを扱うことができる.

f)プラグアンドプレイ
デバイスをネットワークに接続するとユーザは設定することなく,自動的にそのデバイスを使用可能状態にできる.ユーザビリティを向上させる.

g)動的なデバイスの切り替え
入力デバイスと出力デバイスを個別にユーザの好みに応じたもの,またはシステムが適していると判断したものに動的に切り替えて使用することができる.

（２）成果の製品化の可能性,市場性
期待される効果として以下のようなものがあげられる.

a)ユビキタス環境の普及の促進
　これからのユビキタス環境ではあらゆる機器に対してコンピュータが組み込まれて、制御が多様化,複雑化することができる.このような状況は前述したとおり必ずしも利用者に対して便利とは言えず、ユビキタス環境の普及を阻む大きな原因になりかねないない.本提案ではその障害を取り除きユビキタス環境の普及を促進することができる。

b)新たなユーザインタフェースに柔軟に対応できる
　本提案はアプリケーションとユーザインタフェースを切り分けるという新しい概念を取り入れている。これは今までのアプリケーションとユーザインタフェースが密接に結びついているというコンピュータの概念に斬新な一石を投じることになるはずである。新しいユーザインタフェースが開発されたときに、有用なアプリケーションがないためにそのユーザインタフェースの普及の妨げになる場合がある。本提案を用いて従来のアプリケーションに新たなユーザインタフェースを柔軟に適応することで、そのユーザインタフェースの普及に貢献する。このようにして新しいマーケットを開拓することによって多大な経済効果を生む可能性は十分ある。

c)既存のソフトウェアに対して新たな利用法を提案することができる
　本システムでは、アプリケーションに変更を加えずに新たな機能と新たなユーザインタフェースを追加することができる。それにより、既存のアプリケーションの新しい使い方を提案することができる

d)開発者の負担が減り,コストの削減につながる
　また新しい製品が次々と開発される中,ソフトウェアの開発がボトルネックになって製品の開発が遅れたり,コストが高くなったりしているという現

状がある.本システムではユーザインタフェースに関する機能やコンテキストアウェアの機能を下のレベルで提供することによりソフトウェアプログラマは開発の期間を短縮できコストの削減にもつながる。

e)家電制御用途で用いる場合,ほかのミドルウェア（UPnP，Jini，Havi など ）との相性もよい

このように我々のシステムは利用者とベンダーの両方からのニーズに適していて、なおかつ、将来のユビキタス環境を考えたシステムである。このことから製品化の可能性,市場性は十分にあると考える.

６．今後の課題・展望

（１） サポートする入力デバイスの増加

現状では入力デバイスとして PC（マウス・キーボード）、PDA、ゲームコントローラのサポートのみである。これからのユビキタス環境では、新しいデバイスの台頭が考えられるので、それらもサポートできるようにしていきたい。

・サポートするセンサーの増加
現状では Spider のみを用いてコンテキストアウェアネスを実現しているが、今後は使えるセンサーデバイスの数を増やし、それらのデバイスから得られるコンテキスト情報を有効に活用して、よりユーザの嗜好を反映することができるシステムを実現していきたい.

７．実施計画書内容との相違点

（１）UnIt Server，UnIt I/O 間の入出力データフロー(開発者：伊部　担当分)
実施計画書には図２のように UnIt Proxy というユーザインタフェース部分(UnIt I/O)とアプリケーションサーバ(UnIt Server)部分の仲介をし、それらを管理するコンポネントを作る事を提案した。

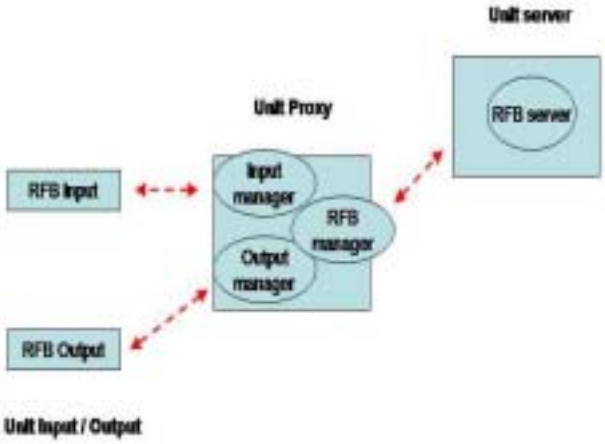

図２：UnIt Proxy コンポネントを介した入出力データ制御

開発段階において、仲介をおく事によるオーバヘッドを考えるようになり、Proxy コンポネントをなくして直接 Server と I/O を通信させることでパフォーマンス向上をはかった。

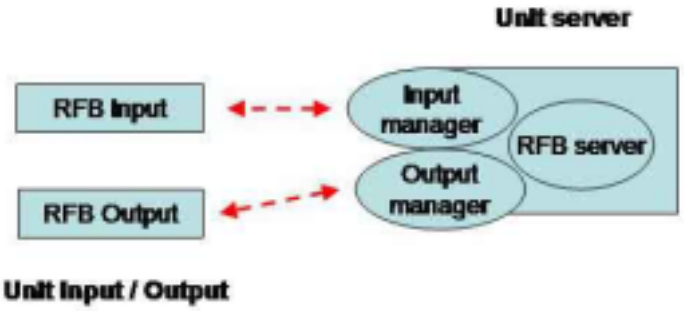

図３：UnIt Proxy コンポーネントを用いない入出力データ制御

しかし、ContextManager コンポネントとの通信も加わり、プラットホーム依存性、
ネットワークのスケーラビリティも考えると機能ごとにコンポネント化することの重要性のほうが強いという結論に到達し結局、もとの図１の設計に戻る事になった。

７、開発分担

（１）芥川は ContextManager と UnItSpider の開発、各コンポーネントにおけるデバイスのデータ管理およびプラグアンドプレイ部分の開発を担当した。
（２） 伊部は主に proxy component の RFBProto モジュール開発、UnItInputManager モジュール開発、UnItOutputManager モジュール開発を担当した。また input component の UnItInputProto モジュール、output component の UnItOutputProto モジュール開発を担当した。
（３） 鈴木は家電制御デモ用のシステムとして、UPnP のコントロールポイントと UPnP 制御の X10 ライト及びＴＶリモコンを実装した。
（４） 服部は Jini を利用したアプリケーションの開発、Jini サービスを監視するアプリケーションの開発を担当した。
（５） 山本は UnItSoundSystem の開発を担当した。

８、その他（特記しておくべき事項　等）

・Spider API は、早稲田大学中島研究室で管理されています。
http://www.dcl.info.waseda.ac.jp

５．参加企業及び機関
なし
６．参考文献

なし