ユビキタスコンピューティングのためのルールに基づく入出力制御デバイス

早川, 敬介 ; 塚本, 昌彦 ; 寺田, 努 ; 義久, 智樹 ; 岸野, 泰恵 ; 柏谷, 篤 ; 坂根, 裕 ; 西尾, 章治郎

(Citation)
ヒューマンインタフェース学会論文誌,5(3):341-354

(Issue Date)
2003-08-26

(Resource Type)
journal article

(Version)
Version of Record

(URL)
https://hdl.handle.net/20.500.14094/90002137

Vol.5 No.3, 2003

ユビキタスコンピューティングのための ルールに基づく入出力制御デバイス

早川 敬介*1 塚本 昌彦*2 寺田 努*3 義久 智樹*2 岸野 泰恵*2
柏谷 篤*1 坂根 裕*4 西尾 章治郎*2

A Rule-based I/O Control Device for Ubiquitous Computing

Keisuke Hayakawa*1, Masahiko Tsukamoto*2, Tsutomu Terada*3, Tomoki Yoshihisa*2, Yasue Kishino*2, Atsushi Kashitani*1, Yutaka Sakane*4 and Shojiro Nishio*2

Abstract - This paper describes how ubiquitous computing environments are organized by interaction of a rule-based event-driven I/O control device. We propose a behavior description language that is based on a framework of the ECA(Event, Condition, Action) rules with simple I/O control functions. Moreover, we show our design and implementation of an I/O control device which can execute ECA rules of our proposing syntax.

Keywords : ubiquitous computing, ubiquitous computer, wearable computer, rule-based I/O control, ECA rule

1. はじめに

コンピュータは我々の日常生活へ着実に浸透してきている. 家電や携帯電話, インターネットに代表されるようにその中核を成すコンピュータの計算力は, すでに我々にとって必要不可欠な存在となっている. 近年ではMEMS (Micro Electro Mechanical Systems)技術の発展によってコンピュータを構成するマイクロチップやセンサ[1]を組み込んだ無線モジュールなどの部品の小型化が進んでおり, 今後はさらに多くの小型なコンピュータが我々の生活を満たしてゆくだろう[2][3]. このようにコンピュータが遍在化することで将来はコンピュータと生活空間が融合されて実世界上の情報とデジタル情報をシームレスに利用できる情報社会の到来[4][5][6]も現実味を帯びてきている. M・ワイザー(Mark Weiser)は, 相互に接続した無数のコンピュータが生活のあらゆる空間に埋め込まれてユーザはそれらの存在を意識することなくその計算パワーを“いつでも”“どこでも”利用できる未来のコンピューティング環境を「ユビキタスコンピューティング (ubiquitous computing)」と呼んだ[7][8].

すでに, ユビキタスという言葉は「遍在する」という意味によって多くの分野で幅広く使用されている. そのため, 現在ではユビキタスという言葉だけではM・ワイザーが提唱したユビキタスコンピューティングの本質を的確に捉えることが難しくなっている. そこで, 筆者らはユビキタスコンピューティングとして, 次のようなコンピュータの形態について考える.

(1) 「場所」コンピュータ
様々な場所に設置された据え置き型コンピュータ. あるいは, 床や壁などのあらゆる場所に埋め込まれたコンピュータ.

(2) 「もの」コンピュータ
あらゆるものに内蔵され, その「もの」と常に一緒に存在するコンピュータ.

(3) 「ひと」コンピュータ
人が装着可能なウェアラブルコンピュータ (Wearable Computer). あるいは体に埋め込むインプラントコンピュータ.

これらのコンピュータは, 各コンピュータが通信機能をもって有機的に結合して相互に情報交換をし, コンピュータやユーザが存在する場所や周囲の状況に応じて自律的に制御動作をおこなう. そして, 互いの協調動作によって自己組織化したシステムとして柔軟に機能する.

つまり, ユビキタスシステムでは, ユビキタスコンピュータがある「場所」に存在すればそこにコンピューティングのための“場”が構成され, 各コンピュータはその“場”に対して各種のコンピューティング機能を提供する. その「場所」に「もの」や「ひと」が集まれば, それらに付加されたユビキタスコンピュータが互いに情報を交換して新しい要求を生成する. そして, その要求を解決するために“場”に応じて各コンピュータの連携がとられることになる.

*1: NEC インターネットシステム研究所
*2: 大阪大学大学院情報科学研究科
*3: 大阪大学サイバーメディアセンター
*4: 大阪大学大学院工学研究科(現 静岡大学情報学部)
*1: Internet Systems Research Laboratories, NEC Corp.
*2: Graduate School of Information Science and Technology, Osaka University
*3: Cybermedia Center, Osaka University
*4: Graduate School of Engineering, Osaka University

実際，このような環境を実現するためには新しいコンピュータの枠組みが必要となるが，この意味でのユビキタスコンピュータを具体化するモデルは現時点では確立されていない．本稿ではユビキタスコンピュータに求められる用件を抽出し，できるだけ必要機能を絞り込んだうえでルールに基づく入出力制御という枠組みのユビキタスシステムを提案するとともに，その動作記述言語の設計と入出力制御をおこなうユビキタスコンピュータの実装について述べる．

2. ユビキタスコンピューティングの要件

従来のコンピュータはディスプレイと入力装置をもち，GUIやCUIによりユーザがシステムを操作して利用するのが一般的である．しかし，見えないコンピュータ[9]としてのユビキタスコンピュータは，物理的にみてもサイズやエネルギーの制約のため単体としては単機能であり，基本的にはPCが持つようなディスプレイやキーボード，マウスなどの入出力デバイスを持たないものと考えるべきである．そのため，ユーザは従来のコンピュータを操作するのとは全く違う動作原理を考える必要がある．その全く違う動作原理とは以下のようにまとめられる．

[ユビキタスコンピュータの動作原理]
ユビキタスコンピュータは，周囲の状況を察知して，他のユビキタスコンピュータと相互に情報交換をし，自律的に動作するものである

このようなユビキタスコンピュータによって統合されたシステム(ユビキタスシステム)は，ハードウエアおよびソフトウエアが小型でありながら次の要件を満たすコンピュータ制御アーキテクチャの実現が課題となる．

自律性： ユーザがいなくても自律的に動作する．
柔軟性： 状況に応じてさまざまな用途に適応する．
有機性： 複数コンピュータが有機的に連動する．

実は我々の生活環境をよく観察してみると，すでにそのような機能を有した組込み機器などが多数設置されていることに気付く．例えば，日が落ちて暗くなると自動的に点灯する街灯，日時によって動作モードを切り替えるエスカレータや信号機，単純なものはセンサで人の存在や周囲の異常状態を感知してドアを開閉したりブザーを鳴らすものまで様々である．このように家の中や街，工場などで広く利用されている通常のシーケンサ（プログラマブルコントローラ）では，ラダーと呼ばれる図的プログラミングが利用されるのが一般的である．しかし，このようなシーケンサは(1)形状が大型，(2)大容量電力の入出力制御，(3)動作中での制御変更が困難であるなどの点から，自律的かつ柔軟に入出力制御を行い有機的に連動することを想定するユビキタスコンピュータとしては，機能がオーバースペックであるか不十分であるといえる．

そこで，我々の生活環境に遍在する数多くのコンピュータが，さまざまな状況に応じて個々に相互作用をおこないユーザと共に全体の協調を生み出すことで，日常の些細な問題を解決するシンプルな仕組みが求められている．これらコンピュータ群が適切に相互接続されることで我々の日常空間が賢く振舞う仕組みとしてイベント駆動型のユビキタスコンピューティングについて次に解説する．

3. イベント駆動型のユビキタスコンピューティング

筆者らはシーケンサの機能をもとにこれらの点を補うためのユビキタスコンピュータの制御手法として，ルール形式によって簡潔に動作を記述する入出力制御を用いたイベント駆動型のユビキタスコンピューティングを提案している．入出力制御を用いたイベント駆動型のユビキタスコンピューティングとは，人工知能のような高度な推論を必要とせずに，ユーザと多種多様なユビキタスコンピュータが互いに環境内で協調しあうシンプルな動作原理に基づくコンピューティング環境である(図1)．

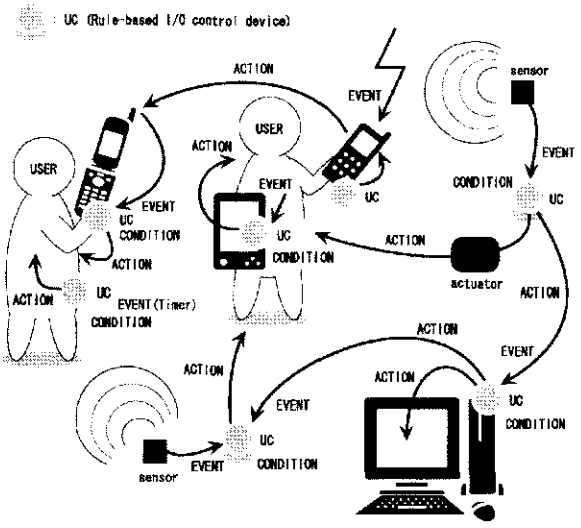


図1 イベント駆動型のユビキタスコンピューティングの概念図

一般に人間は現実世界の事象を因果的に捉えている．このような現実世界の物事の動作を理解する原理[10]を，ユビキタスコンピュータの動作原理に適用する．つまり，筆者らのユビキタスコンピューティングの基本原理では，生活空間に存在する全ての「もの」が簡単な入力と出力というシーケンス制御によって結ばれて協調動作するシステムを想定する．つまり，イベント駆動型のユビキタスコンピューティングでは，外部で発生した事象（入力）に応じて外部に対して行動（出力）を行う小型でシンプルなユビキタスコンピュータ（入出力制御デバイス）によってシステムを構成する．

このイベント駆動型のユビキタスコンピューティングでは様々な形態のシステムを想定することが可能だが，それゆえに想定したシステムを早期に実現して生活環境上での現実的な利便性や運用性に関連する技術的課題を検証することが必要かつ不可欠といえる．そのため，シス

テム設計の自由度が高く機能の拡張性が容易なプラットフォームの提供は，研究開発やユビキタスコンピューティング環境の普及を支援するうえで有用である．

現在，情報機器が相互に接続することでユーザの嗜好や状況に応じた利便性の高いサービスを実現するアプローチとして，センサーネットワークの研究開発が進められている．例えば，小型のセンサモジュール群を使用してユーザのプレゼンス情報に応じたビルの空調や照明を効率的に管理するシステムなどが提案されている．SmartDust[11]プロジェクトに代表されるセンサーネットワークに関連する研究開発の主目的は，個々の計算資源が乏しいセンサモジュール群を使用して広域な実空間上の情報を実時間で計測し収集することにある．そのため，一般に狭帯域リンクや省電力に最適化したルーティングプロトコルの開発に重点が置かれており，これらの研究に関連して低コストで小型のセンサモジュールの開発も進められている．小型無線センサモジュールMICA[12]では，マイクロチップやバッテリを実装するメイン基板が512バイト程度のRAM上で機能するTinyOS[13]と呼ばれるオペレーションシステムによって駆動し，音センサ，光センサ，温度センサを実装したサブ基盤を追加で装着することにより各アナログデータを収集することができる．TinyOSやPushpinComputing[15]のBerthaのように小型センサモジュールに最適なタスクスケジューリングを特徴とするオペレーションシステム上でも，例えばTinyOSで使用するC言語ライクなオブジェクト指向言語nesC[14]のようにイベント駆動システムの開発環境が提供されている．しかし，これらの技術はセンサモジュールの動作がマイクロプロセッサの実行プロセスに組み込まれるため，将来のシステム改良に伴って発生するセンサ機能の追加や削除が必要になる場合には動作記述の変更が容易ではない．このように，現時点で想定したユビキタスコンピューティング環境が時代とともに変化して要求されるサービスが多様化する場合には，イベント駆動システムが柔軟にその動作を変更できる制御アーキテクチャが必要となる．さらに，ユビキタスコンピューティングの形態（場所，もの，ひと）や駆動するセンサの違いに対応する電源供給の方法，センサやアクチュエータの拡張，有線や様々な無線方式（Bluetooth[16]，UWB[17]，ZigBee[18]，etc.）を将来に渡って自由に選択することが可能なハードウエアの実装形態が望ましい．これらの点を踏まえて，汎用的なマイクロチップを使った単一モジュールを，簡単なルールを用いて動作記述することによって様々な入出力制御に利用することを実現できれば，早期に試作システムを構築できるばかりでなく，個々に制御機器を設計開発する従来手法と比較してモジュールの再利用性や量産効果による開発コストの低減が期待できる．このような観点から，筆者らはイベント駆動型のユビキタスコンピューティングを実現するための制御アーキテクチャの設計と小型の入出力制御デバイスの試作をおこなった．

4. ECAルールによる動作記述言語の設計

イベント駆動型のユビキタスコンピュータは他のコンピュータとの連携が必要であり，また，動的な動作変更が必要となる．さらに，対象とするコンピュータが小型・軽量・非力であるため，動作記述言語もできるだけシンプルで処理が簡単なものが望ましい．そこで，因果関係を記述するための世界自体は次のようなECAルールによってモデル化する．

イベント(E)： 実世界で起こった事象
コンディション(C)： 実行するための特定の状態
アクション(A)： 外部への動作と内部状態の変更

これは，ユビキタスコンピュータにある事象(Event)が起こり，ある特定の状態(Condition)を満たしているとき，その行動(Action)を実行する，という動作原理に基づいている．ECAルールに基づいた動作記述方式は，古くからデータベースの分野において自律性を備えたデータベース(Active Database)の記述として使われてきたものである[19]．これはエキスパートシステムで人間の知識を表現するために利用されるif-then形式の記述に似ている．ここでは，if部に相当する部分を，イベントとコンディションという外部事象と内部状態に分割して段階的に処理することで，イベント駆動型のエンジンを実現している．このようなシステムは，ルールの集合として表現されるのでルールの部分的な追加・削除により自由に機能の変更や拡張ができるという柔軟性を備えている．また，ルールを送受信することで，自身が保有するルールを周囲の端末などに移し，場所に応じて必要なルールを受け取る位置・状況依存のサービスが実現できる[20][21]．よって，個々のコンピュータは幾つかのルールを組み合わせて実現した単純な機能しか持たないが，集合体としてシステムを構成することで利便性のあるサービスをユーザへ提供することを目的とする．イベント駆動型のユビキタスコンピューティングにおいてルール形式で記述すべき機能を以下に挙げる．

(1) 入出力信号の制御機能
任意の入出力ポートからの信号を制御する．
(2) 内部状態の管理機能
内部状態変数を保持/変更する．
(3) メッセージ機能
コンピュータ間で情報交換する．
(4) タイマ機能
時間によって動作タイミングを管理する．
(5) ルールの書換え機能
動作中に制御規則を追加/削除する．

一般に，ECAルールの言語仕様を設計するにあたっては，イベント，コンディション，アクションにそれぞれ何を記述できるようにするかをまず決定する必要がある．その後，それぞれの複数記述を許可するか，メタルールやルールのプライオリティの存在を認めるかといった点を決定する．想定するユビキタスコンピュータは図2に示すように，各種のセンサやボタンなどの入力状態を取得し

て処理を行い，さまざまなデバイスへの出力を行う機器である．内部状態を保持するためのレジスタをいくつか保持し，不揮発性の内部メモリにルールを記憶する．プロトタイプの実装ではまず有線通信を想定するため，センサやボタンを接続するポートとは別に2系統のシリアルポートを装備し，他のユビキタスコンピュータやPC，PDAなどとメッセージ交換を行うことができる．また，複数のシステムタイマを持ちルールで自由に設定し管理することができる．さらに，ユビキタスコンピュータをルールによって自由に省電力モードへ移行させて消費電力の節約がおこなえる．このようなユビキタスコンピュータの機能を実現するためのECAルール言語仕様の設計について各節で述べる．

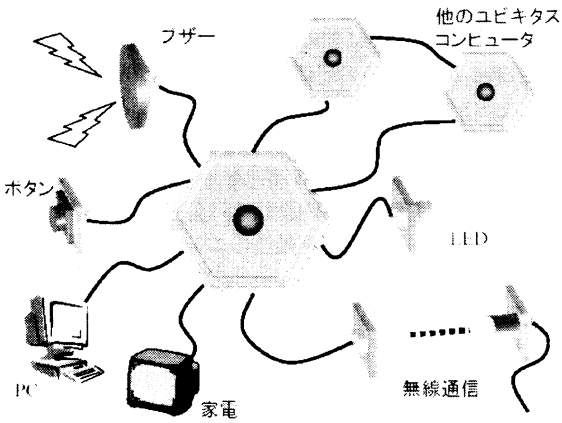


図2 想定するユビキタスコンピュータ

4.1 イベント(Event)

記述できるイベントを表1に示す．混成イベントを許可した場合，過去に起こったイベントの管理など処理が複雑になるため，今回のシステムでは単一のイベントを扱うルールしか記述できないようにした．また，入力ポートの信号に関してはそれぞれのポートがON/OFFの状態を保持しているためイベントとしては取り扱わずにコンディションとして扱う．そのため，入力ポートの状態のみでルールが発火するようにイベント部を持たないルールの記述を許可する．イベント部を持たないルールはコンディションが満たされる状態になればすぐに発火する．

よって，本稿のユビキタスコンピュータが取り扱うイベントはRECEIVEイベントとTIMERイベントの2種類となる．RECEIVEイベントは，デバイスのシリアル通信ポートからのメッセージ受信を検出して発火する．TIMERイベントは，あらかじめ指定した時間が経過したときに発火する．

表1 イベントの一覧

イベントの名称	内容
RECEIVE	メッセージの受信
TIMER	タイマの発火

4.2 コンディション(Condition)

記述できるコンディションを表2に示す．INPUTコンディションは，センサなどを経由した外部からの入力信号をまとめて評価する．INPUT_STATEコンディションはデバイスが保持する内部状態変数をまとめて評価する．それらの条件がすべて満たされたときのみアクションが実行される．

表2 コンディションの一覧

コンディションの名称	内容
INPUT	入力ポートのON/OFF比較
INPUT_STATE	内部状態変数の比較

4.3 アクション(Action)

記述できるアクションを表3に示す．OUTPUTアクションは，入出力制御デバイスの出力ポートを管理し，特定の出力ポートから信号を出力することで，出力ポートにつないだブザーやLED(Light Emitting Diode)，パラレルのデジタル信号などを制御する．

表3 アクションの一覧

アクションの名称	内容
OUTPUT	出力ポートのON/OFF設定
OUTPUT_STATE	内部状態変数の値設定
TIMER_SET	タイマの設定
SEND_MESSAGE	メッセージの送信
SEND_COMMAND	コマンドの送信
DEVICE_CONTROL	デバイス固有機能の制御

OUTPUT_STATEアクションは，内部状態変数の値を変更するためのアクションである．TIMER_SETアクションは内部タイマを設定するアクションで，タイマのインターバル時間および発火条件(1回のみ発火/繰り返し発火)を設定できる．SEND_MESSEGEアクションは，特定のIDをもつ3ビット長メッセージをシリアルポートを介して他のユビキタスコンピュータへ送信するアクションである．メッセージを受信したユビキタスコンピュータではRECEIVEイベントが発火するので，その発火したイベントに対する処理(アクション)を書いておけばユビキタスコンピュータ間で連携が可能になる．SEND_COMMANDアクションでは，他のユビキタスコンピュータに対して制御コマンドを送信することで，他のユビキタスコンピュータが保持するルールの追加や削除といったルールの書換え機能を実現する．制御コマンドとして取り扱える内容を表4に示す．制御コマンドはメッセージ送信とは異なり，受信したコンピュータ側でイベントが発生することなくシステムで処理するため，受信側で適応的に処理を変更することはできない．DEVICE_CONTROLアクションは，ユビキタスコ

ンピュータの機器制御を行うアクションとして，ユビキタスコンピュータの動作モニタとして主に利用するためにデバイスの中央部に設けた汎用LEDの発光制御(点灯/消灯/点滅)や，省電力モードへの移行と復帰，2ポートあるシリアルポートの排他的制御を行う．

表4 コマンドの一覧

コマンドの名称	内容
ADD_ECA	ECAルールの追加
DELETE_ECA	ECAルールの削除
REQUEST_ECA	ECAルールの送信要求

その他にも，複数のコンディション(マルチコンディション)をひとつのルールに記述できる機能や，複数のアクション(マルチアクション)をひとつのルールに記述して逐次実行できる機能を提供しており，柔軟に制御動作を記述できる．また，すべてのユビキタスコンピュータが必ずしも備える必要はないが，ユーザとの何らかのインタフェースを提供するためには，例えばダイヤルやキーなどの入力手段やディスプレイ機能のような出力手段が必要な場合もある．ここではこれらはセンサやアクチュエータの一種とみなして，入出力制御機能によって管理することを想定している．

4.4 ルールのコーディング

ルールはビット列に変換されてユビキタスコンピュータの内部メモリに格納される．ルールの基本構造は，後述するユビキタスコンピュータの実装にあわせて，イベント部およびコンディション部からなる実行条件記述部と，アクション部からなる実行動作記述部を各々1ブロック(2バイト)で記述する(図3)．

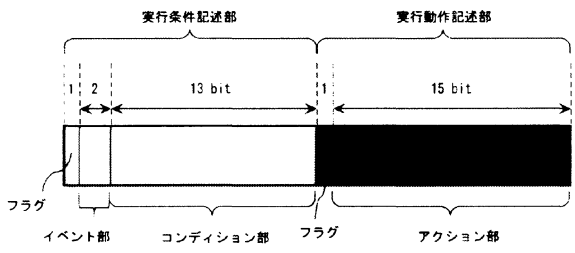


図3 ECAルールの基本構造

ECAルールのコーディングは図4に示す規則に沿って行う．マルチコンディションやマルチアクションを利用する場合は1ブロック単位でルール長が増加することになる．イベント部は実行条件記述部の2～3ビット目の2ビットで記述し，続く4～16ビット目までにコンディション部を記述する．ただし，その記述する内容は図に示すようにイベントの種類によって異なる．マルチコンディションフラグを“1”に設定した場合，次の2バイトにはアクション部ではなく2つ目のコンディションを記述する．実行動作記述部の先頭1ビットはマルチアクションを記述するかを決定する継続フラグとなっており，継続フラグが“1”である限り続けてアクション部を記述することで複数のアクションを順次実行させることができる．アクションは図4に示すようにそのアクション内容によってフォーマットが異なる．まず2ビット目でOUTPUT/OUTPUT_STATEアクションとそれ以外のアクションを区別し，OUTPUT/OUTPUT_STATEアクションの場合は残りの14ビットで出力を指定する．その際，オフセットフラグを用いてポート1～6と7～12を切り替えて12個の出力ポートを制御する．その他のアクションの場合はさらに先頭3ビットでアクションの種類を決定し，残り11ビットでそれぞれのアクション内容を記述する．

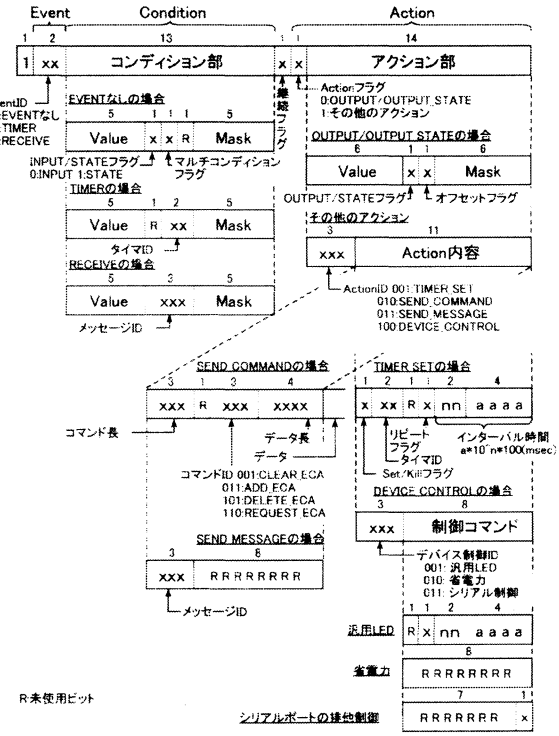


図4 ECAルールのフォーマット

例えば，このフォーマットに従って「入力ポートIN1～IN6のうちIN1がON，IN2がOFF，IN5がONのとき，入出力制御デバイスのタイマが5秒後に1回だけ発火する」というルールを記述すると図5に示すように「0x91 0x13 0x49 0x15」の4バイトにコーディングされる．

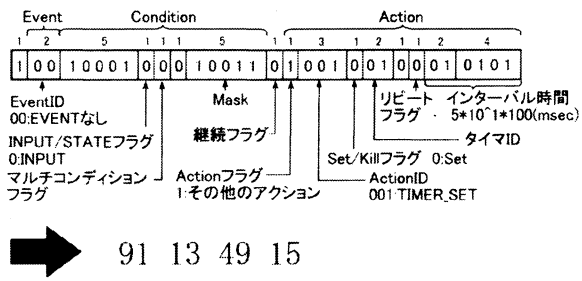


図5 ECAルールによる動作記述例

5. 入出力制御デバイスのプロトタイプ

筆者らは，数種類の試作を経て，イベント駆動型のユビキタスコンピューティングを実現する入出力制御デバイスを試作した．図6に示す入出力制御デバイスは前述したルールに基づく動作記述に従って，センサやアクチュエータの制御が可能である．このプロトタイプは，図7に示すように最大で入力信号5 ポート，出力信号12 ポートを制御できる．また，送受信機能として排他的に利用可能なシリアル通信2 ポートを用いて，他の入出力制御デバイスやPC などと連携して動作することもできる．

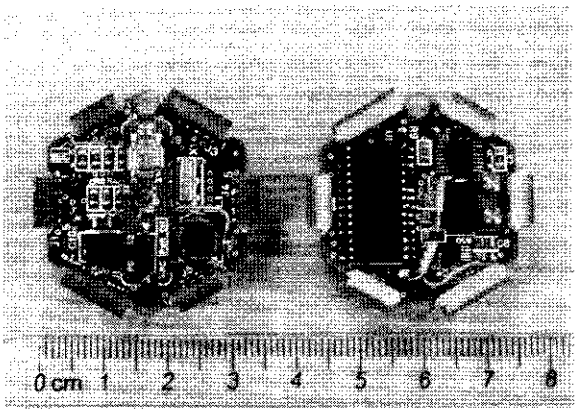


図6 入出力制御デバイスの回路基板(右：表側，左：裏側)

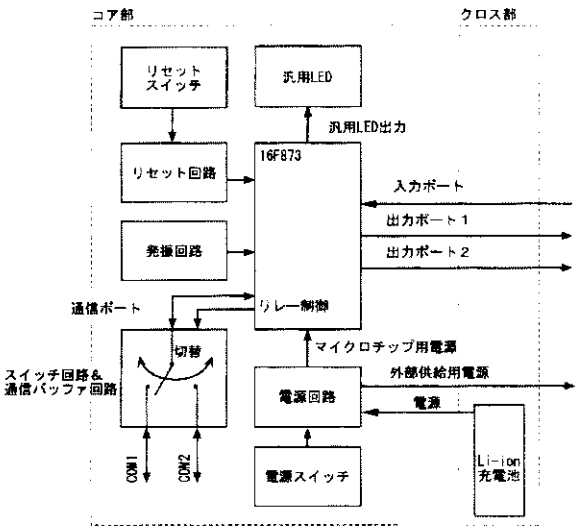


図7 入出力制御デバイスの回路構成

5.1 システム設計

今回試作した入出力制御デバイスは，Microchip Technology 社製のマイクロプロセッサ(PIC16F873)を使って実装している．このマイクロプロセッサは，入出力ピンを持つプログラミング可能なRISC型プロセッサで，安価であるが不揮発性メモリやタイマ，省電力モードなどの機能を有し，命令数も必要最低限で容易に制御プログラミングが可能であるため，近年様々な分野で使用されている．ただし，使用するマイクロプロセッサが機器制御に利用する汎用的な小型チップのため，プログラムメモリやファイルレジスタ，ルールの保持領域などにはかなりの制限がある．そのため，実装上の工夫が必要となり，イベント駆動型のユビキタスコンピューティングのアーキテクチャを検証する対象としてこのマイクロプロセッサを選択した．例えば，試作したデバイスはマイクロプロセッサの内部にあるEEP-ROM(128 バイト)に保持した全てのルールを逐次処理することで動作する．バイナリで記述したルールを効率的に逐次処理するために，ルールのフォーマットに従ってルールは1ブロック (2バイト) 単位で処理される(図9)．図8に示すようにルール処理部はルールベースから実行条件記述部のみを読み込む．イベントおよびコンディションが満たされている場合には，実行動作記述部を読み込み，アクションを実行する．満たされない場合には，続けて次のECAルールの実行条件部を読み出す．n個のECAルールを全て参照すると再び最初のECAルールから処理をおこなう．このようなルール処理によって，イベントが発火しないルールに対しては無駄なメモリアクセスを行わないなどの小型デバイスの特性を考慮したルール処理エンジンを実装している．このルール処理エンジンは軽量であるため，PIC16F873 上であってもルールの実行動作をストレスなくおこなうことができている．

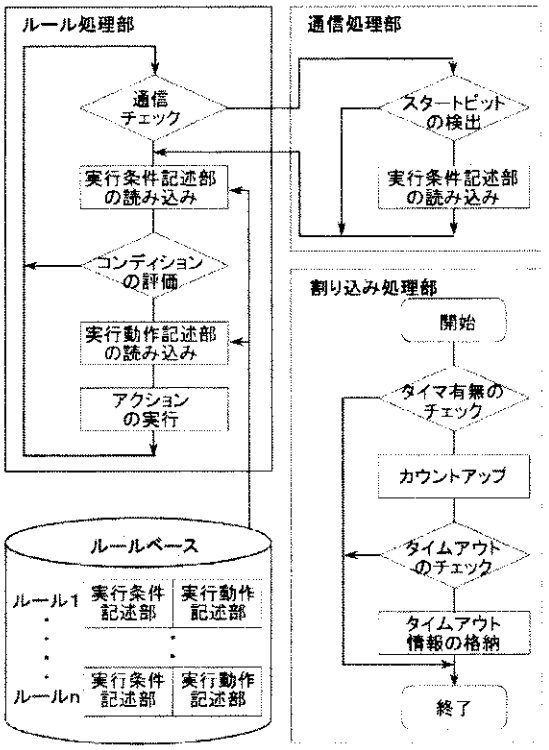


図8 入出力制御デバイスのルール処理ブロック図

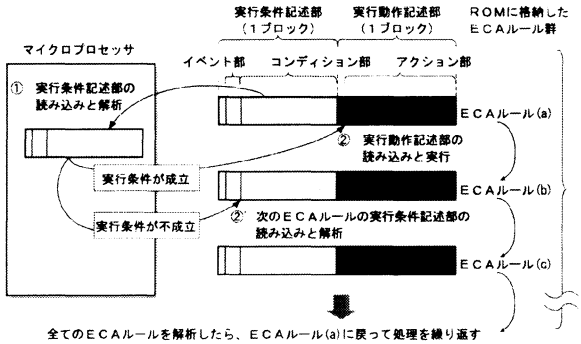


図9 ECAルールの処理フロー

5.1.1 入出力制御および内部状態変数の管理機能

ECAルールを用いてデバイスの入出力制御を行うには，コンディションにON/OFFの入力状態を指定し，アクションに出力状態を指定する．さらに，マスクを用いることで，ある端子の入力状態のみ評価したり，ある端子の出力状態は変更せず，他の端子の出力状態をONにするといった制御を可能にする(図10)．INPUT_DATAを現在の入力状態，INPUT VALUE をコンディションで指定された入力状態とする．それぞれの変数のi（= 1,・・・,m）ビット目は，m 個ある入力端子のi 番目の入力状態とする．INPUT_MASK を入力端子のマスク情報とし，出力端子に関しても同じ様に定義する．例えば，INPUT_MASK=00011の場合，INPUT_DATA=01010，INPUT_VALUE=01110とすれば，これらはマスク後にINPUT_DATA=INPUT_VALUE=000010となり，コンディションが成立する．もし，INPUT_DATA=01000，INPUT_VALUE=01110とすると，これらはマスク後にINPUT_DATA=000000，INPUT_VALUE=000010となり，コンディションは成立しない．すなわち，

(INPUT_DATA XOR INPUT_VALUE) AND INPUT_MASK

が0になると，コンディションは満たされ，出力端子を

(OUTPUT_DATA AND OUTPUT_MASK) OR
(OUTPUT_VALUE AND OUTPUT_MASK)

で与えられる状態に変更する．同様に，入出力フラグを設定することで内部状態変数も管理することができる．

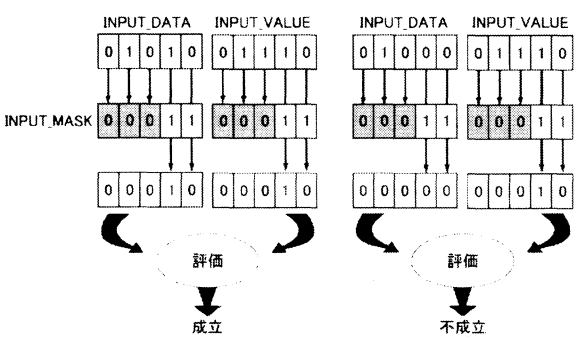


図10 ECAルールに基づく入出力制御の例

5.1.2 メッセージの送受信機能

メッセージはシリアル通信で送受信するため，スタートビット検出に失敗すると通信エラーが発生する．そのため，図8左に示す様に，1個のルールをチェックする度にスタートビットを検出するための通信チェックを行う．スタートビットが検出されると，メッセージをメッセージ専用レジスタに格納する．受信したメッセージは，RECEIVEイベントでコンディションの評価を行った後にレジスタから削除する．この様に，すべてのルールをチェックする時間は数ミリ秒あり，メッセージを受信してからRECEIVEイベントが評価されるまでの遅延が生じるが，提案するイベント駆動型のユビキタスコンピューティングは一般に非同期システム[22]となるため想定するシステムの多く場合では問題にならない遅延時間と考えられる．

5.1.3 タイマの設定機能

使用するマイクロプロセッサは，あらかじめ定められた一定間隔の割り込み処理しか行えないため，任意の間隔のタイマを実現することは困難である．そこで，タイマの間隔を割り込み処理間隔の整数倍に固定し，カウンタを用いて任意の間隔のタイマを実現する．割り込み処理でカウンタの値を1ずつ増やし，あらかじめ設定した値になると，タイムアウトしたこと示す情報をレジスタに格納する．例えば，20ミリ秒間隔で割り込み処理が行われる場合に100ミリ秒間隔のタイマを発生させるには，カウンタが5になるとタイムアウトとみなすように設定すればよい．複数のカウンタを用いることで，間隔の異なる複数のタイマを同時に利用できる．タイムアウトしたタイマはTIMERイベントで評価する．

5.1.4 コマンドの送受信機能

ルールの消去や追加はシリアル通信でコマンドを送受信して行う．コマンドには，ルールを追加するADD_ECA，格納されているルールを削除するDELETE_ECA，アドレスを指定してメモリの内容の送信を要求するREQUEST_ECAなどがある．さらに，コマンドを発生させるルールも記述できるため，ルールを用いて接続している他のデバイスに対してルールを追加するなどの操作ができる．

5.2 利用形態

試作した入出力制御デバイスは，図11に示すようにコア部（白色）とクロス部（黒色）の二つのデバイスで構成される．外部から電源供給を受けられない実行環境では本体であるコア部をリチウムイオン充電池を備えたクロス部に装着することで独立したモジュールとして利用できる．クロス部はセンサや他の入出力制御デバイスとの接続にも利用する．入出力制御デバイスは，その周囲に他の入出力制御デバイスとの通信や多数のセンサやアクチュエータを接続することが容易な六角柱状の亀甲型形状でデザインした．

デバイスのコア部は，マイクロプロセッサ，シリアル通信用バッファ，電源安定化回路を基本構成とするデバイスの本体である．直径が約34mm の六角柱の形状をしたコ

ア部の中央には，デバイスの駆動状態を示す汎用LEDがあり，その両脇にデバイスの電源スイッチおよびリセット・スイッチが設けられている．コア部六角柱の各側面には電源ポート1つ，入力ポート1つ(IN1～6)，出力ポート2つ(OUT1a～6a，1b～6b)を設ける．また，六角柱の2つの側面にはシリアル通信ポート（COM1, COM2）を設けている（図12）．デバイスのクロス部は，外部のセンサやアクチュエータなどの入出力デバイスとデバイスのコア部との接続を仲介する変換デバイスである．試作した入出力制御デバイスは，独立で動作する場合には通常はコア部をクロス部中央にあるスロットに挿入して使用する．

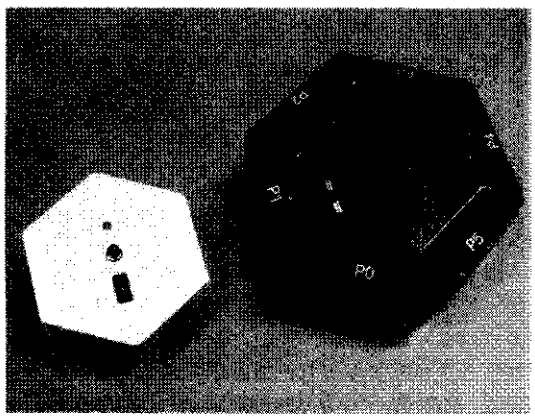


図11 コア部(右)とクロス部(左)

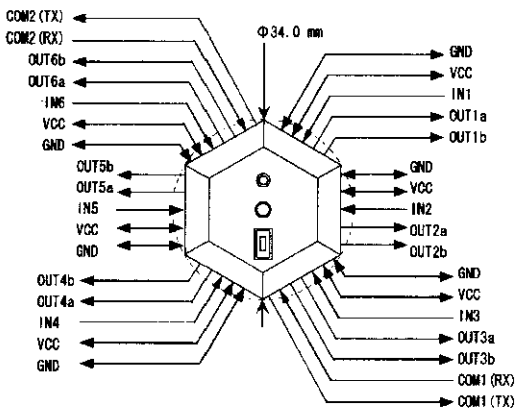


図12 入出力制御デバイスの入出力ポート

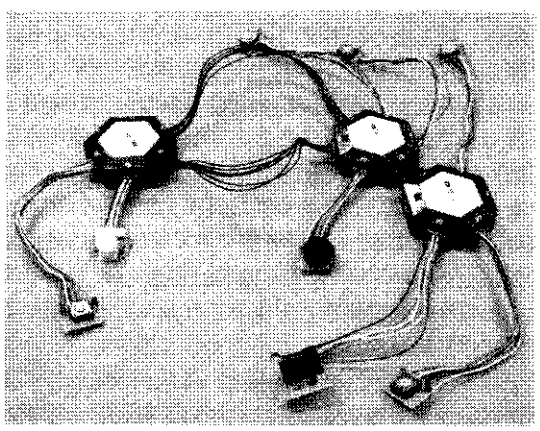


図13 クロス部とケーブルを利用した接続例

コア部の側面に配したコネクタがクロス部のスロット側面に設けたスプリング式コネクタと接触することで，コア部とクロス部が連結する．直径が約59mm の六角柱の形状をしたクロス部は，Li-ion充電池を内蔵しており，クロス部六角柱の各側面には充電用ポートと外部に接続するセンサなどへの電源供給ポートを兼用するポートを配する．コア部と同様に，クロス部六角柱の各側面には電源ポート1つと出力ポート2つ，六角柱の5つの側面には入力ポートが各1つ，六角柱の2つの側面にはシリアル通信ポートが各2つ設けられている．クロス部の側面は入出力デバイスを挿抜することを目的とするコネクタ形状になっており，デバイス用の幾つかのソケット器具を用いることでデバイス間を自由に連結した有線通信が可能である（図13）．また，クロス部の底面に無線モジュールなど積み重ねることで，クロス部の内部でシリアルポートを経由した入出力制御デバイスの無線化など拡張が容易な構造となっている（図14）．

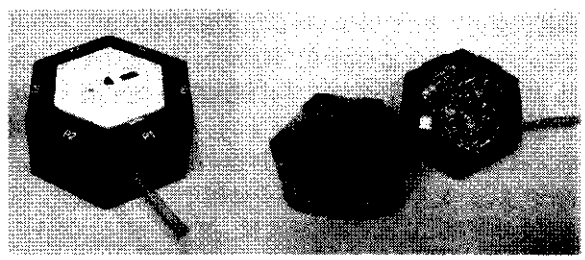


図14 クロス部を無線化したハードウエアの拡張例

6. 入出力制御デバイスの開発環境

ECAルールのフォーマットを解説したとおり，今回実装する入出力制御デバイスの動作記述言語は計算資源の乏しい汎用のマイクロプロセッサで実行するためビット列で表現されている．しかし誰もが効率的にシステムを構築するためには直感的にルールの記述内容を把握してルールの作成[23]やデバッグするツールが必要である．そこで，PC上で簡単にECAルールを作成して入出力制御デバイスに書き込むことができるエディタとルールライタを開発環境として準備した．また，PCと入出力制御デバイスの連携を容易に実現するためにPC上で実行可能な入出力制御デバイスのエミュレータソフトも開発した．これらのソフトについて各節で述べる．

6.1 ECAルールのエディタとルールライタ

入出力制御デバイスに格納するECAルールの作成を容易にしてアプリケーションの開発を支援するための環境として，ECAルールのエディタと入出力制御デバイスへルールを書き込むライタソフトを開発した．ECAルールのエディタは，ユーザがデバイス上で動作させたいルールを入力していくことで，前述のフォーマットに従ってバイナリの形式で記述されたルール群をファイルへ出力するアプリケーションである．ECAルールはイベント，コンディション，アクションの3つを一組にして動作を記述する言語であり，図15に示す入力画面では，イベント記述

部でメッセージの受信，タイマの発火といった条件を，コンディション記述部で入力状態と内部状態を，アクション記述部で出力やタイマの設定，メッセージの送信などの動作をマウス操作で指定する．製作したルールの動作記述内容は例えばルールモニター部に次のような文章として表示されるので，デバッグ作業が簡単に行える．

「ID が T1 のタイマが発火したとき，ST1 と ST3 が OFF を満たせば，ST2 と ST3 を ON に ST1 を OFF にする．」

次に，エディタで作成したECAルールをライタソフトを使ってPCから入出力制御デバイスへシリアル通信でコマンドを送信することでルールの消去や追加などの書き込みが行える．図16にECAルールライタの操作画面を示す．ルールを書き込むときには，エディタで出力されたファイルを読み込み，ルールを分析して個々のルールに分ける．また，このプログラムはデバイスに格納されたルールの読み取り機能をもっている．全てのアドレスについてREQUEST_ECA を繰り返すことでルールを読み取り，読み取ったルールを分析して，別のデバイスに書き込める．このように，ECAルールのライタソフトによってPCとシリアル通信で接続したデバイスのルールを自由に制御できる．

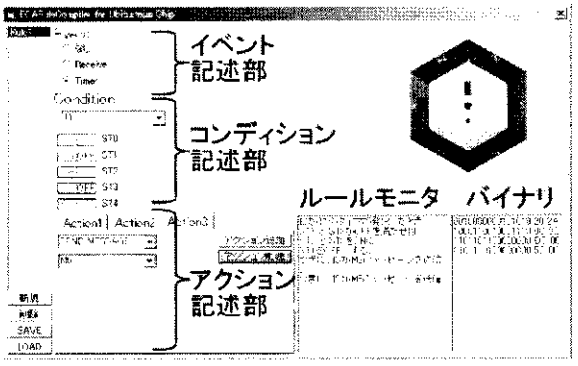


図15 ECAルールのエディタ画面

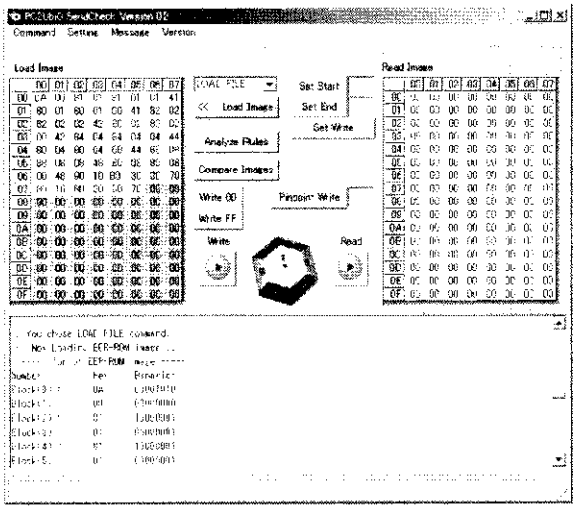


図16 ECAルールライタの操作画面

6.2 PC環境での入出力制御デバイスのエミュレーション

デバイスはシリアル通信を用いて，メッセージやコマンドを送受信できることはすでに述べた．この機能を用いて，入出力制御デバイスに接続したセンサから得られる情報をPCで利用したり，入出力制御デバイスに格納されているルールをPCで制御するなど，PCと入出力制御デバイスを連携させることで実空間と仮想空間をシームレスに繋ぐ多様なサービスが提供できる．そこで，PCと入出力制御デバイスを統合利用するために，PC上に入出力制御デバイスのエミュレータを実装した．図17にエミュレータの画面を示す．エミュレータも，入出力制御デバイスで用いているものと同じフォーマットのルールで動作を記述できる．エミュレータはPC上で動作するため，ルールを記憶するメモリの大きさに制限がない．この特徴を利用し，入出力制御デバイスと同じように動作させだけでなく，他の入出力制御デバイスのためのルールをメモリに格納し，接続してきた入出力制御デバイスへルールを配信するといった利用法も可能である．また，実機の入出力制御デバイスがセンサやアクチュエータを介して実世界に作用するように，入出力制御デバイスのエミュレータはセンサやアクチュエータに準じる簡単なソフトウエアを介してPC上のアプリケーションが発生させるイベントを捕らえたり，アプリケーション動作を制御するなど仮想世界に作用することができる(図18)．

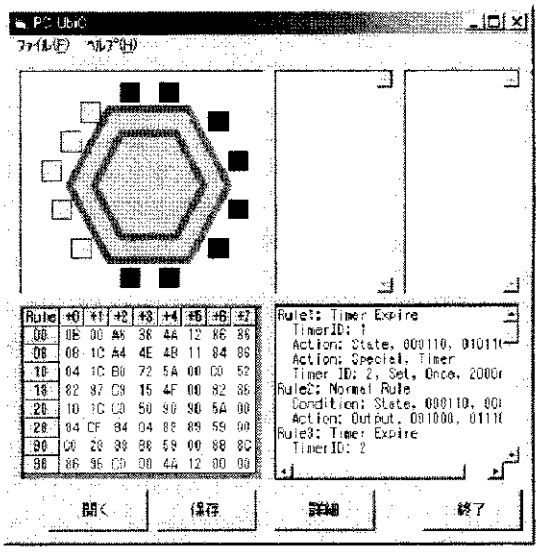


図17 入出力制御デバイスのエミュレータ画面

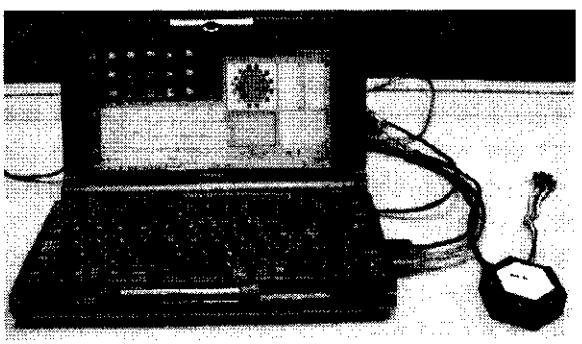


図18 入出力制御デバイスとPCの接続

7. 入出力制御デバイスの適用例

従来，ユビキタスシステムを構築してその検証するにあたって各々のシステムアーキテクチャに合わせて個々に制御機器を設計する手法が一般的であった．本章では，自由度の高いユビキタスシステムの設計と評価作業を早期に達成するために，単一の制御アーキテクチャの入出力制御デバイスを利用したイベント駆動型のユビキタスコンピューティングの二つの具体的例をついて述べる．

7.1 適用例その1

あらゆる場所に埋め込まれた「場所」コンピュータやものに埋め込まれた「もの」コンピュータを入出力制御デバイスで実現するシステムとして，例えば，代表的なユビキタスコンピューティング環境例であるIT化された住宅やビルの管理システムのように，ユーザのプレゼンス情報（在室，不在，会議中など）に合わせて，照明やエアコンなどの家電製品が連動する生活環境を構築することを考える．

図19のようにドアに埋め込まれた入出力制御デバイスAは入力ポートに接続したドア開閉検出センサ(防犯用の磁気スイッチ)から入力される状態をコンディションとして捉えて，出力ポートに接続した無線送信モジュール(1ビットデジタル信号の送信）を制御するアクションを実行する．無線受信モジュール(1ビットデジタル信号の受信)を接続した他の入出力制御デバイスBは，受信信号の状態をコンディションとして捉えて部屋の照明や空調システムの各スイッチを制御するアクションを実行する(図20)．もし，部屋の照明などを管理するホームサーバが存在するならば，入出力制御デバイスBはルール処理のエミュレータを実行するホームサーバに各スイッチを制御するアクションを実行して，その情報をホームサーバはイベントとして捉えてコンディションに応じてアクションを実行すればよい．このように，「ユーザが部屋に入った」という状況に応じて生活空間が賢く振舞う仕組みを，入出力制御デバイスのルール処理によって比較的簡単に構築することができる(表5)．また，ユーザが部屋に入った状況をより高い精度で検出するために入出力制御デバイスAに焦電センサ(人体の赤外線を感知)を追加したり，入出力制御デバイスCが伝言メモを録音済みでなおかつ入出力制御デバイスBがスイッチを入れるアクションを実行したときにだけ伝言メモを再生するアクションを実行して，伝言メモが録音されていないときに不要な再生動作を行わせないというような動作の追加も実現できる．このように，イベント駆動型のユビキタスコンピューティングでは発生するイベントのコンディションを相互利用することで“場”が形成される．もちろん，これらの基本的な動作内容は，部屋という“場”を想定してあらかじめ考えて作成した基本的なルール群を準備する必要はあるが，入出力制御デバイスが特徴とする自律性や柔軟性，有機性によって，ユーザの嗜好や部屋の環境変化にあわせて後からルールを追加変更して“場”の振る舞いを自由にカスタマイズすることができる．

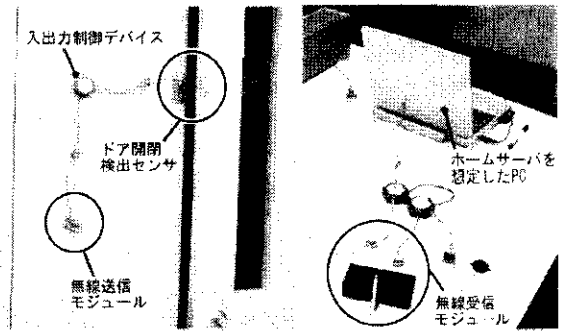


図19 「場所」コンピュータとしての入出力制御デバイス

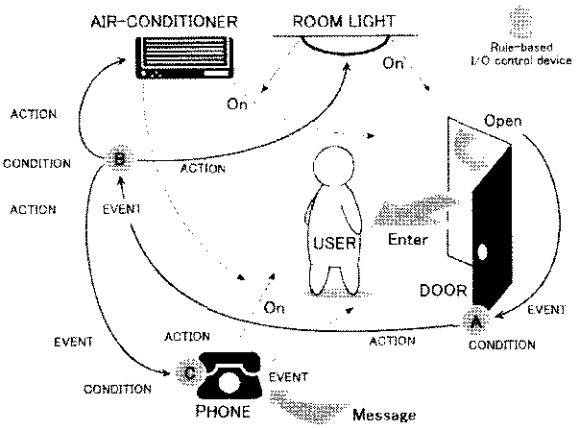


図20 入出力制御デバイスが形成する“場”の例1

表5 適用例1のルール記述例

入出力制御デバイスA		入出力制御デバイスB	
ドアが開いたら，無線送信モジュールをONにする		無線受信モジュールがONならば，ホームサーバにメッセージを送信して伝言メモを再生する	
IN1: ドア開閉センサ OUT3: 無線送信モジュール		IN4: 無線受信モジュール OUT5: 入出力制御デバイスC COM1: ホームサーバ	
ルール1	ルール2	ルール3	ルール4
Event: なし	Event: なし	Event: なし	Event: なし
Condition: IN1= ON	Condition: IN1= OFF	Condition: IN4= ON	Condition: IN4= OFF
Action: OUT3= ON	Action: OUT3= OFF	Action: OUT5= ON, COM1:MessageID= 1 を送信	Action: OUT5= OFF, COM1:MessageID= 2 を送信

入出力制御デバイスC	
制御信号を受信したら，伝言メモを再生する	
IN1: 入出力制御デバイスB OUT1: 伝言メモの再生 OUT2: 伝言メモの停止	
ルール5	ルール6
Event: なし	Event: なし
Condition: IN1= ON	Condition: IN1= OFF
Action: OUT1= ON OUT2= OFF	Action: OUT1= OFF OUT2= ON

7.2 適用例その2

人が装着可能な「ひと」コンピュータや壁に埋め込まれた「場所」コンピュータを入出力制御デバイスで実現するシステムとして，ユーザの位置情報などを管理したり場所に関連づけられたデジタル情報を提示するようなロケーションシステム[4][24]を構築することを考える．

図21のように手首に装着した入出力制御デバイスA(「ひと」コンピュータ)の通信ポートには赤外線通信モジュールが接続されており，壁などに埋め込まれた別の入出力制御デバイスB(「場所」コンピュータ)が送出するメッセージやコマンドをイベントとして外界から受信する．受信したメッセージの内容に応じて入出力制御デバイスAが振動子を駆動することで，イベントの発生をユーザが振動で知覚することができる．また，ユーザが装着した別の入出力制御デバイスやPCへ出力信号やメッセージを送信するアクションを実行することで，ユーザの周囲に“場”が拡張される．

例えば，図22のように足首に装着した入出力制御デバイスCは，靴底に埋め込んだ複数のスイッチのコンディションを捉えてユーザの歩行情報の一部を取得する．手首の入出力制御デバイスAと足の入出力制御デバイスCから得られるユーザのセンシング情報は，さらにルール処理のエミュレータを実行するウェアラブルPCがイベントとして処理してアクションを実行する(表6)．よって，ユーザが立ち止まったときに外部からメッセージを受信すれば，腕の振動と共に頭部搭載ディスプレイの画面にはそのメッセージIDに対応するインターネット情報を表示するようなサービスがルールの記述だけで容易に構築できる．振動ではなく汎用LEDの発光に切り替えたければ，そのルールを新たに追加して解決できる．サービスをうけるタイミングにバリエーションを持たせたければ，腰のウェアラブルコンピュータや壁の入出力制御デバイスがコマンドを発行してルールの追加や交換だけで対応できる。

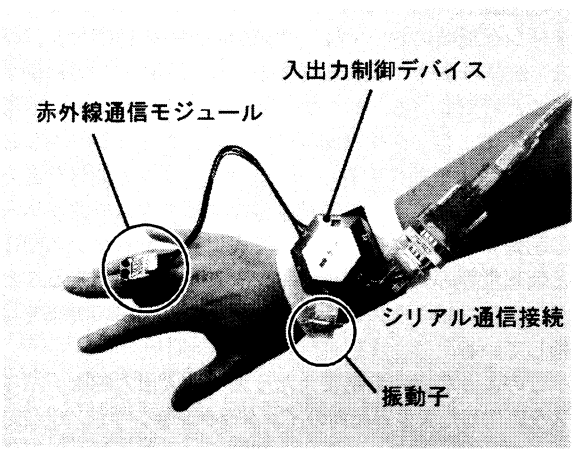


図21 手首に装着した入出力制御デバイス

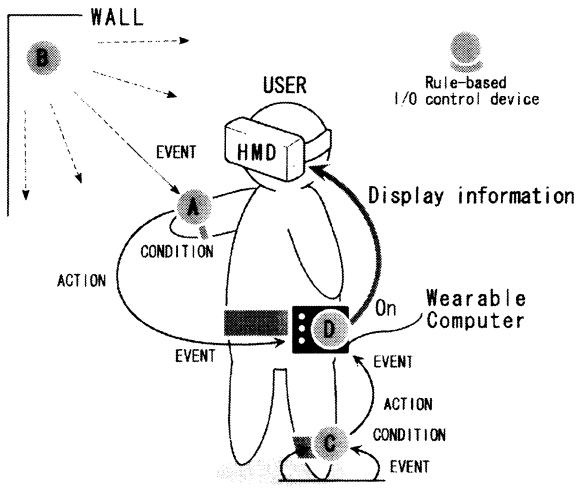


図22 入出力制御デバイスが形成する“場”の例2

表6 適用例2のルール記述例

入出力制御デバイスA		入出力制御デバイスB
赤外線通信モジュールがメッセージを受信したら，ウェアラブルコンピュータにメッセージを送信する		メッセージを送信しつづける
COM1: 赤外線通信モジュール COM2: ウェアラブルコンピュータ		COM1: 赤外線通信モジュール
ルール1	ルール2	ルール3
Event: RECEIVE	Event: TIMER	Event: なし
Condition: MessageID= 1	Condition: TimerID= 1	Condition: なし
Action: COM2にMessageID= 4を送信 5秒のTimerID= 1を設定	Action: 通信ポートをCOM1に再設定	Action: COM1にMessageID= 1を送信

入出力制御デバイスC			
靴底のスイッチの状態をウェアラブルコンピュータへ送信する			
IN1: スイッチ1 IN2: スイッチ2 COM1: ウェアラブルコンピュータ			
ルール4	ルール5	ルール6	ルール7
Event: なし	Event: なし	Event: なし	Event: なし
Condition: IN1= ON IN2= ON	Condition: IN1= ON IN2= OFF	Condition: IN1= OFF IN2= ON	Condition: IN1= OFF IN2= OFF
Action: COM1 にMessageID= 1 を送信	Action: COM1 にMessageID= 2 を送信	Action: COM1 にMessageID= 3 を送信	Action: COM1 にMessageID= 4 を送信

8. まとめと今後の課題

本稿では，イベント駆動型のユビキタスコンピューティングの基本原理に基づいて，ルールに基づいて入出力制御するユビキタスコンピュータの動作記述手法について解説し，試作した入出力制御デバイスと二つの適用例について述べた．そして，一般に入手可能な部品で構成した低コストの小型の入出力制御デバイスが，単一の制御アーキテクチャに基づいてルールの追加や変更によって，センサやアクチュエータを使って想定する様々なユビキタスコンピューティング環境を柔軟に構築する可能性を示した．

近年，強力な計算能力をもつコンピュータを身近に利用できる環境が整いはじめている．しかし，高度に情報化が進むにつれて便利になったように思える我々の日常生活を改めて振り返ると，意外にも生活に密着した些細な問題の多くがいまだに解決されていないことに気付く．例えば，「無くしたものが見つからない」，「賞味期限に気づかない」というようなごく身近なところでの簡単なコンピューティングが実現されていない．つまり，本稿で述べた入出力制御デバイスは，このような身近なコンピューティング環境を見直して問題を解決する方法を探るためのデバイスである．

この入出力制御デバイスは，7章で述べた適用例1のように従来のホームサーバのような中央管理システムにも適用することができるが，むしろユビキタスコンピューティング環境で提唱される“あらゆる場所や物に偏在する無数のコンピュータによるシステム”の考え方を実現する手法として，各情報機器の制御信号が伝播することで制御ノードが自律的に機能するようなシステムに適用することに重点を置いて開発を進めた．なぜなら，シーケンス制御が主となるイベント駆動型のユビキタスコンピューティングが大規模になれば，従来型の中央管理システムのように多数のセンサの値を収集して解析を行い制御指令を通達する手法では通信遅延などによって実世界と仮想世界とのインタラクション性が低下することが懸念されるため，入出力制御デバイスなどの各制御ノードがある程度の自律性を持って動作することで反射作用的に情報サービスを提供する仕組みが重要だと考えるからである．

現在，筆者らはイベント駆動型のユビキタスコンピューティングで想定されるアプリケーションを入出力制御デバイスで構築しながら検証作業をおこなっている．実際に検証の過程において幾つかの想定するアプリケーションでは，要求する制御内容を実現するためには本稿で述べた動作記述方法だけでは不十分であったり冗長な制御ルールになるなどの問題がみつかっている．これらを改善すべく仕様の拡張や変更と同時に新技術の検討を進めている．例えば，入出力制御デバイスが集合して情報サービスを提供する“場”では，アプリケーションが異なれば各メッセージが持つ意味が当然異なることが予想されるため，本稿で述べる3ビット(8種類)のメッセージ機能だけで世界中の“場”を一元管理することは不可能である．そのため，大規模なユビキタスコンピューティング環境を支援するためには，本稿で述べた制御アーキテクチャ以外にも“場”で利用するメッセージをアプリケーション毎に意味付けするための新たな仕組み（仮にサービスプロファイルと呼ぶ）が必要だと考えている．サービスプロファイルを定義するためには，様々なユビキタスコンピューティング環境を構築しながらアプリケーションに応じた基本的なルールセットを抽出しつつ十分にシステムを洞察することが求められるため，今後の大きな課題である．

また，本稿ではシーケンス機能に重点を置いたため，センサーネットワークみられるデータ収集機能については述べていない．しかし，ユビキタスコンピューティング環境の構築において必須であるユーザのプレゼンス情報やコンテキスト管理にはデータベースによるデータ収集機能は欠くことができない．そこで，筆者らは特にアドホックな無線通信ネットワークへの適用を十分に考慮しながら，シーケンス制御の即応性を加味したイベント駆動型のユビキタスコンピューティングに最適なルーティング方法をシミュレーションによって現在検証中である．

さらに，イベント駆動型のユビキタスコンピューティングにおけるルールの追加や削除をより効率的に行うため，ファイルシステムに似たルール管理方法の実装を進めている．ただし，本研究では，独自のI/O制御やファイル管理を行うルール処理エンジンをオペレーションシステム(OS)とは呼ばず，ミドルウエアの位置付けとして取り扱っている．これは，ユビキタスシステムを構成する機器が本論文で述べた汎用のマイクロプロセッサのようなデバイスだけでなく，例えばパーソナルコンピュータ，携帯情報端末，携帯電話，家電のような組込み機器など，動作環境(プラットフォーム)が異なる多種多様な機器によって構成すると考えるのが一般的だからである．すでに，各プラットフォームにおいて最適なOSは多数存在しており，厳密なタスク管理やセキュリティー機能などを実行レベルで保証する動作環境[25]も整備されつつある．これら多くの基盤技術によって形成するユビキタスコンピューティング環境においては，本研究のルール処理エンジンは，既存のOS上で協調動作するモジュールとして提供すること考えている．そこで，図17に示したWindowsOS用のPCエミュレータやWindowsCE用のPDAエミュレータなど異なるOS上で動作するルール処理エンジンの開発を進めており，これによって多種多様なユビキタス機器が単一の制御アーキテクチャによって実世界のシーケンス動作と仮想世界のデジタル情報をシームレスに利用することができるユビキタスコンピューティング環境の実現を目指している．

このような課題を踏まえたうえで，入出力制御デバイスを応用した実世界指向のインタラクション手法[26][27]の研究など，試作した入出力制御デバイスの新たな可能性も探りながら「ユビキタスコンピュータが如何にしてユーザにとって快適な“場”を形成するのか」を今後も追求してゆく．

参考文献

[1] P.Saffo: Sensors:The Next Wave of Infotech Innovation; Ten-Year Forcast, Institute for the Future (IFTF), pp.115-122, (1997).

[2] J.Hill, R.Szewczyk, A.Woo, S.Hollar, D.Culler and K.Pister: System architecture directions for networked sensors; Architectural Support for Programming Languages and Operating Systems (ASPLOS2000), pp.93-104, (2000).

[3] K.Sakamura: TRON: Total Architecture; In Proceedings of the Architecture Workshop in Japan' 84, pp.41-50, (1984).

[4] R.Want, A.Hopper, V.Falcao and J. Gibbons: The Active Badge Location System; ACM Transactions on Information Systems, Volume10, Number 1, pp91-102, (1992).

[5] A.Ward: Sensor-Driven Computing; doctoral dissertation (University of Cambridge), (1998).

[6] L.Holmquist, F.Mattern, B.Schiele et al: Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts; Ubicomp 2001, pp116-122, (2001).

[9] D.Norman: The Invisible Computer; MIT Press, (1998).

[7] M.Weiser: The Computer for the Twenty-first Century; Scientific American, Vol.265, No.3, pp.94-104, (1991).

[8] P.Wellner, E.Machay, R.Gold, M.Weiser, et al: Computer-Augmented Environments: Back To The Real World; Communications of the ACM (CACM),Vol.36, No.7, pp.24-97, (1993). (「電脳強化環境」;パーソナルメディア)

[10] R.Arkin: Behavior-Based Robotics; MIT Press, (1998).

[11] J.Kahn, R.Katz and K.Pister: Mobile Networking for Smart Dust; In Proceedings of ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom99), pp.271-278, (1999).

[12] MICA http://www.xbow.com/Products/Wireless_Sensor_Networks.htm

[13] TinyOS http://webs.cs.berkeley.edu/tos/

[14] nesC http://nescc.sourceforge.net/nesc-ref.pdf

[15] J.Lifton, D.Seetharam, M.Broxton and J.Paradiso: Pushpin Computing System Overview:A Platform for Distributed, Embedded, Ubiquitous Sensor Networks: In Proceedings of the Pervasive Computing, pp.139-151,(2002).

[16] Bluetooth https://www.bluetooth.org/

[17] UWB(Ultra Wide Band) http://www.uwb.org/

[18] ZigBee http://www.zigbee.org/

[19] J.Widom and S.Ceri: Active Database Systems: Triggers and Rules for Advanced Database Processing; Morgan Kaufmann Publishers, (1996).

[20] A.Schmidt, K.Adoo, A.Takaluoma, U.Tuomela, et al: Advanced Interaction in Context; In Proceedings of the First International Symposium on Handheld and Ubiquitous Computing (HUC99), pp.89-101, (1999).

[21] D.Ipina: An ECA Rule-Matching Service for Simpler Development of Reactive Applications; In Supplement to the Proceedings of Middleware 2001, IEEE Distributed Systems Online, (2001).

[22] I.Sutherland and J.Ebergen: COMPUTERS WITHOUT CLOCKS; Scientific American, Vol.287, No.2, pp.62-69, (2002).

[23] M.Resnick and S.Ocko: LEGO/Logo: Learning Through and About Design; http://llk.media.mit.edu/papers/1991/ll.html

[24] M.Addlesee, R.Curwen, S.Hodges, J.Newman, P.Steggles, A.Ward and A.Hopper: Implementing a Sentient Computing System; IEEE Computer Magazine, pp.50-56, (2001).

[25] 別冊トロンウェア "T-Engine"; パーソナルメディア, (2003)

[26] M.Gorbet, M.Orth and H.Ishii: Triangles: Tangible Interface for Manipulation and Exploration of Digital Information Topography; In Proceedings of the ACM CHI'98, pp.49-56, (1998).

[27] K.Fishkin, K.Partridge and S.Chatterjee: Wireless User Interface Components for Personal Area Network; IEEE Pervasive Computing Magazine, pp.49-55,(2002).

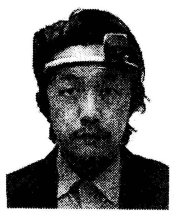
(2003年1月31日受付, 6月2日再受付)

著者紹介

早川 敬介 (正会員)

平成8年筑波大学大学院理工学研究科修士課程終了. 同年, 日本電気株式会社に入社, 現在に至る. 主に, 携帯情報端末に関するハードウエアとヒューマンインタフェースの研究に従事. ヒューマンインタフェース学会, 日本バーチャルリアリティ学会, 情報処理学会会員.

塚本 昌彦

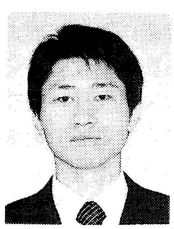


昭和62年京都大学工学部数理工学科卒業. 平成元年同大学院工学研究科博士前期課程修了. 同年, シャープ(株)入社. 平成7年大阪大学大学院工学研究科情報システム工学専攻講師を経て, 平成8年同専攻助教授, 平成14年より同大学院情報科学研究科マルチメディア工学専攻助教授となり, 現在に至る. 工学博士. ウェアラブルコンピューティング・ユビキタスコンピューティングに興味をもつ. ACM, IEEEなど8学会の会員.

寺田 努

平9年大阪大学工学部情報システム工学科卒業. 平11同大大学院工学研究科修士課程了. 平成12年同大学院退学. 同年より大阪大学サイバーメディアセンター助手. 平成14年より大阪大学大学院情報科学研究科マルチメディア工学専攻助手を併任. 現在に至る. アクティブデータベース, モバイルコンピューティング, データ放送の研究に従事.

義久 智樹

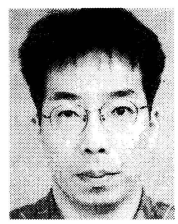


昭和54年生. 平成14年大阪大学工学部電子情報エネルギー工学科卒業. 同年, 同大学院情報科学研究科マルチメディア工学専攻在籍. ユビキタスコンピューティングおよび衛星デジタル放送に興味をもつ.

岸野 泰恵

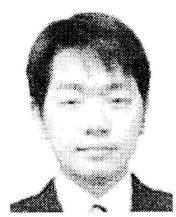
平成14年大阪大学工学部電子情報エネルギー工学科. 現在, 同大学院情報科学研究科マルチメディア工学専攻在籍. バーチャルリアリティ, ヒューマンインタフェースに興味をもつ.

柏谷 篤



平成元年大阪大学大学院理学研究科物理専攻修士課程修了. 同年, 日本電気株式会社に入社, 現在インターネットシステム研究所ユビキタスシステムテクノロジーグループに所属. 画像入出力機器, 携帯端末の研究に従事. 電子情報通信学会会員.

坂根 裕



平成10年大阪大学工学部情報システム工学科卒業. 平成12年同大学院工学研究科修士課程修了. 平成14年より静岡大学情報学部助手となる. ウェアラブルコンピューティングとユビキタスコンピューティングの研究に従事. 情報処理学会, 電子情報通信処理学会, 日本バーチャルリアリティ学会会員.

西尾 章治郎

昭和50年京都大学工学部数理工学科卒業. 昭和55年同大学工学研究科博士課程修了. 工学博士. 京都大学工学部助手, 大阪大学基礎工学部および情報処理教育センター助教授, 同大学院工学研究科情報システム工学専攻教授を経て, 平成14年より同大学院情報科学研究科マルチメディア工学専攻教授となり, 現在に至る. 平成12年より大阪大学サイバーセンター長を併任. この間, カナダ・ウォータールー大学, ビクトリア大学客員. データベース, 知識ベース, 分散システムの研究に従事. 現在, ACM Trans. on Internet Technology, Data & Knowledge Engineering, Data Mining and Knowledge Discoveryなどの論文誌編集委員. 情報処理学会フェロー含めACM, IEEEなど8学会の会員.