情報処理学会論文誌　コンシューマ・デバイス & システム　Vol.2　No.1　10–19 (Mar. 2012)

コンシューマ・システム論文

携帯端末のWebブラウザを用いた異種ネットワークデバイス連携システムの開発

田中 剛[1,a)]　伊藤 崇洋[2,†1]　加藤 悠一郎[2,†2]　峰野 博史[1]　水野 忠則[3,†3]

受付日 2011年5月20日, 採録日 2011年12月2日

概要：本論文では，異なるネットワーク上に点在するセンサデバイスや家電機器などのあらゆるデバイスを統一的に制御するシステムについて述べる．本システムは，通信プロトコルの異なったネットワーク上のデバイスを相互接続し，連携させることでデバイスを統一的に扱うことができ，携帯端末を用いてセンサデータの閲覧や家電機器の操作が可能となる．また，それらはすべてWebを通して通信を行っており，携帯端末のブラウザからあるセンサに対して値を設定し，センサがその値を示したときに特定の家電機器を動作させるといったデバイス連携サービスの実現も可能となっている．

キーワード：デバイス連携，Webサーバ，PUCC，異種ネットワーク

Development of Heterogeneous Network Devices Integrating System with Web Browser of Mobile Device

Go Tanaka[1,a)]　Takahiro Ito[2,†1]　Yuichiro Kato[2,†2]　Hiroshi Mineno[1]
Tadanori Mizuno[3,†3]

Received: May 20, 2011, Accepted: December 2, 2011

Abstract: In this paper, we describe the system which controls various devices such as sensor devices and home appliances on the heterogeneous networks. This system can deal with a unified devices by making devices on the networks where communication protocols are different to interconnect and cooperate, and it is possible to monitor the sensing data and control the home appliances with mobile devices. Moreover, as they all communicate through a Web server, it is possible to achieve a integrative device service using the Web browsers of mobile devices. We implemented a sample service where a user set a value for a sensor with his/her Web browser, and when the sensor displayed its value, the Web server automatically controlled a specific appliance.

Keywords: Web server, PUCC, device integration, heterogeneous network

[1] 静岡大学情報学部
Faculty of Informatics, Shizuoka University, Hamamatsu, Shizuoka 432–8011, Japan
[2] 静岡大学大学院情報学研究科
Graduate School of Informatics, Shizuoka University, Hamamatsu, Shizuoka 432–8011, Japan
[3] 静岡大学創造科学技術大学院
Graduate School of Science and Technology, Shizuoka University, Hamamatsu, Shizuoka 432–8011, Japan
[†1] 現在，株式会社サイバーエージェント
Presently with Cyber Agent, Inc.
[†2] 現在，SCSK株式会社
Presently with SCSK Corporation
[†3] 現在，愛知工業大学情報科学部
Presently with Faculty of Information Science, Aichi Institute of Technology
[a)] go@minelab.jp

1. はじめに

近年，通信技術の発展により，Bluetooth，Zigbee，IrDAなど様々な通信規格が登場してきた．それにともない，通信機能搭載のデバイスも続々と販売され，様々な規格に準じた多くのデバイスがそれぞれネットワークを構築するようになった．そのため，異なるネットワークが同一環境に混在する状況が身近に感じられるようになった．しかし，情報家電などでは各メーカ独自の規格によってネットワークが構成されており，通信プロトコルの違いから他ネットワークのデバイスと相互接続できなくなっている．その中で，UPnP [1] やHAVi [2] のようなデバイス相互接続に関

する研究がされている．しかし，これらのミドルウェアどうしの相互接続性は確保しておらず，物理的に接続されていても機器連携を行うことはできない．

一方，相互接続性の課題はセンサネットワーク技術においても同様である．近年のセンサデバイスは様々な情報を得ることができ，あらゆる分野で注目を浴びている．今後，身の回りには複数のセンサがつねに存在することが予想される．しかし，一般的に個々のセンサデバイスがそれぞれネットワークを形成しており，省電力や通信性能などを考慮して独自の通信・制御プロトコルを使っている．そのため，ある用途のために設置されたセンサは通信プロトコルの違いからその用途以外での利用が難しくなっている．その中で，SENSORD [10] や GSN [3], [11] のような複数のセンサデバイスを統一的に管理するような研究がされている．しかし，これらはセンサデバイス，センサネットワークを対象としており，他デバイスとの連携や制御は意図していない．

このように，プロトコルの違いによる相互接続性には依然課題がある．情報家電やセンサデバイスそれぞれを相互接続できるようにし，さらにそれらすべてのデバイスを統一的に扱うことで，デバイスが持つサービス提供の範囲を超えて有効活用することが可能となると考える．そこで我々は，異種ネットワーク内のデバイスを相互接続し，Web を通して各種サービスの連携設定を行えるようなシステムの開発を行った．本システムでは，デバイス相互接続基盤を搭載した Web サーバを通してセンサや家電に統一的なアクセスが可能である．

本論文は，全 5 章で構成される．次の 2 章では，関連研究とその課題について述べる．3 章では，システムの提案を行う．4 章では，実装と評価について述べる．最後に 5 章で結論と今後の課題についてまとめる．

2. 関連研究

2.1 デバイス相互接続プロトコル

異種ネットワークに接続されたデバイスを相互に接続する技術として UPnP，HAVi，PUCC についてそれぞれ概要と課題を述べる．

UPnP（Universal Plug and Play）は，Plug and Play の機能をネットワーク上でも行えるようにした規格である．PC とその周辺機器をはじめとし，AV 機器などの家電製品と情報機器をネットワークを通じて接続し，連携して相互に機能を提供することが可能である．しかし，UPnP は HTTP や SOAP などの技術を基本としており，下位通信プロトコルは IP 系であるため，IP 系のデバイス対象という形で依存してしまっている．

HAVi（Home Audio Video Interoperability）は，家庭内においてネットワーク接続された異なるメーカ・機種の AV 機器を相互に接続するためのミドルウェア仕様である．主な特徴として，Plug and Play 機能，ネットワークの拡張性，リソースの管理などがあげられる．しかし，それらの相互接続される AV 機器は下位通信プロトコルとして IEEE1394 を利用することを意図して規格化されているため，IEEE1394 のネットワーク上のデバイス対象という形で依存してしまっている．また，HAVi は Java の技術を前提としたミドルウェア規格であり，サービスは Java オブジェクトとして表現されるため，家電で HAVi 規格を使うには家電に Java 実行環境も組み込む必要がある．

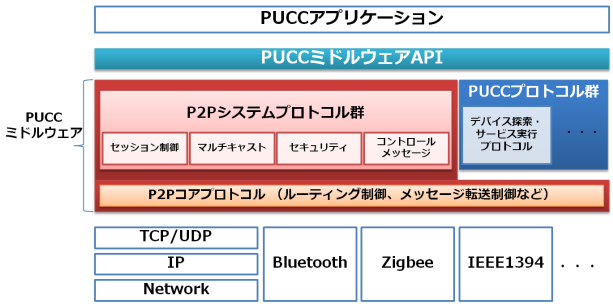

図 1　PUCC プロトコルスタック
Fig. 1　Protocol stack of PUCC.

PUCC（P2P Universal Computing Consortium）[4] では，Bluetooth や DLNA などの既存ネットワークの上に，オーバレイネットワークを形成することで，P2P ネットワークを利用して様々なネットワークに存在する機器を相互接続・運用可能にしており，そのために必要なプロトコルとデバイス情報を記載するメタデータの仕様定義を行っている．また，PUCC のプロトコルスタックは図 1 のようになっており，IP または，Bluetooth，Zigbee，IEEE1394 などの非 IP 系の下位通信プロトコルの上に，経路制御やメッセージ制御を行う PUCC プロトコルを実装している．

2.2 サービス連携システム

異種ネットワークに接続されたデバイスの持つサービスを連携させるシステムとして SENSORD，GSN についてそれぞれ概要と課題を述べる．

SENSORD（Sensor-Event-Driven Service Coordination Middleware）は，複数の異なるセンサデバイスの情報を統一的に管理し，連携機能を提供するミドルウェアである．低次元のセンサデータを高速に解析するために，センサデータをその時空間情報とともに共有メモリ上に保持する機能を持つ．しかし，SENSORD は異なるセンサデバイスを統一的に扱う，という点に集中して作りこまれたアーキテクチャであり，登録したイベントの検知と通知といった動作におさまってしまっている．

GSN（Global Sensor Networks）は，センサを抽象化し，異種センサネットワークの統合のためのプラットフォームを提案している．スケーラビリティを考慮して P2P を適用しており，Web 技術のように単純で力強く柔軟な設計方

針である．しかし，GSN は異種センサネットワークを統合し，様々な複数のセンサからデータを収集する，という点におさまっており，制御までは意図されていない．

2.3　まとめ

以上のように，UPnP と HAVi は家電機器を対象としており，あらゆる種類のすべてのデバイスに対して 1 つの規格で相互接続することが難しい．UPnP は IP ネットワーク上のデバイスを，HAVi では非 IP 系である IEEE1394 を使ったネットワーク上のデバイスを対象としているため，これらの技術を同一環境でそのまま適用することができない．また，SENSORD と GSN はセンサデバイスやセンサネットワークの統合を対象としているため，複数のセンサからデータを収集する点におさまっており，他のデバイス制御までを意図していない．しかし，PUCC ではオーバレイネットワークを構築することで IP 系や非 IP 系などの下位通信プロトコルの違いを気にすることなく様々なデバイスを相互接続可能である．それにより，センサデバイスと家電機器などの連携も可能となり，柔軟なサービスが可能となる．よって，我々は PUCC プロトコルを利用してデバイスを相互接続し，それらを連携させるサービス連携システムの提案を行う．

3.　異種ネットワークデバイス連携システム

本章では，提案する異種ネットワークデバイス連携システムの構成要素と要素技術について述べる．提案するシステムの概要図を**図 2** に示す．本システムは，複数の異種ネットワークが混在している環境での利用を想定している．そのような環境下で，ネットワークの違いを意識せず，異なるネットワークに接続されたデバイスを相互に接続できなければならない．また，それはいつでもどこでも身近な物を使って様々なデバイスの情報を閲覧できたり，相互接続されたデバイス間の連携設定を行えたりする必要がある．さらに，デバイスの連携設定の際には，ユーザがより直感的にシステムを扱うためにも，ユーザにとって冗長な処理やコストは可能な限り省き，効率的で簡単に連携を行える仕組みが必要である．

そこで，我々は PUCC プロトコルの上で，異種ネットワーク内のデバイスを下位通信プロトコルの違いを意識せず相互に接続し，それらデバイスを携帯端末の Web ブラウザから冗長な処理を行わず効率的に操作・連携できるようなシステムを提案する．常時 Web サーバを稼働させておくことで，定期的に PUCC プロトコルが提供する API を用いて異種ネットワークの状況をつねに把握し，ブラウザ機能を持つ携帯端末さえあればいつでもどこでもすべてのデバイスの最新の情報を閲覧したり，遠隔操作したりすることが可能である．さらに，センサに対してイベントを設定し，そのイベント発生時に特定の家電を自動的に動作

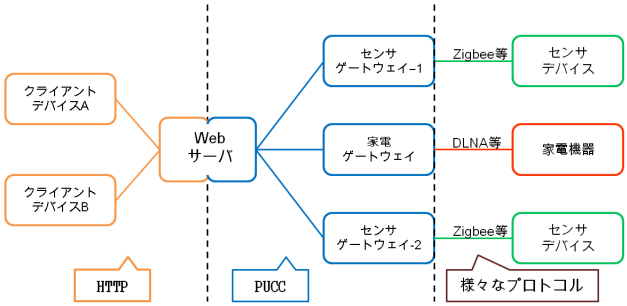

図 2　異種ネットワークデバイス連携システム概要図
Fig. 2　Overview of heterogeneous network devices integrating system.

させるといった異種ネットワークデバイス間の連携設定も可能となる．また，本システムでは Web サーバを介することで，他のデバイスとのコネクション確立やデバイスを探索するといった，ユーザにとっては冗長な処理はすべてサーバ側が定期的に行うような仕組みを検討した．この Web システムを使うことで，ユーザにとって身近で馴染みのある自分の携帯端末を利用できるため，より直感的で手軽にデバイスを連携させることが可能である．さらに，本システムにおける Web サーバを ASP サービスのような形でサービスを提供できるようにすることで，ユーザにとってコストの削減となるだけでなく，よりスケーラブルなシステムになると考える．

3.1　システム構成要素

本システムでは，センサゲートウェイによって様々なセンサデバイスからデータを収集し，家電ゲートウェイによって様々な家電に対してそれぞれが持つサービスを実行する．それぞれゲートウェイが管理するデバイスの情報はメタデータとして記述しているため，そのメタデータを管理することになる．また，クライアントデバイスによって，相互接続されたデバイスの情報の閲覧やデバイス間の連携設定を行うようにする．その際に，クライアントデバイスとセンサ・家電ゲートウェイとの通信を仲介するのが Web サーバである．センサゲートウェイ，家電ゲートウェイおよび Web サーバは PUCC のプラットフォームを実装しており，それらを PUCC ノードと呼ぶ．以下ではそれらの要素について述べる．

3.1.1　センサゲートウェイ

センサデバイスを PUCC のネットワークに参加させるには，センサデバイスに PUCC のプラットフォームを実装し，PUCC ノードとする必要がある．しかし，センサデバイスのような小型なデバイスは省電力やそれら適用シーンを考慮されて作られることが多く，用途に合った必要な機能しか持たないため処理能力が低い．PUCC のプラットフォームをそのようなデバイスの上で実装することは困難であり，電力消費も大きくなるため，センサゲートウェ

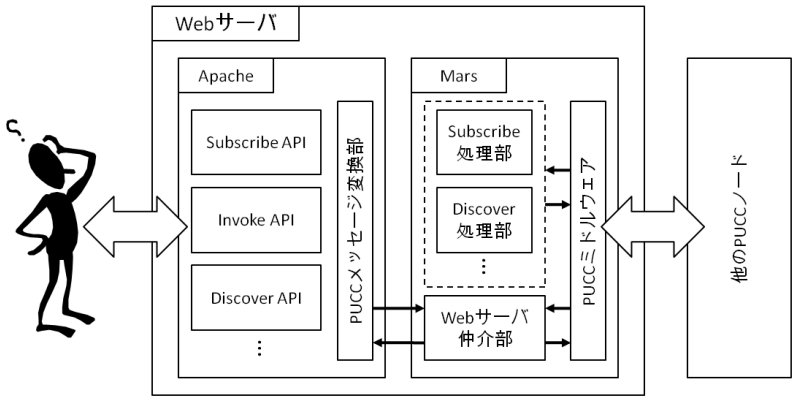

図 3　Web サーバアーキテクチャ
Fig. 3　Web server architecture.

イを利用し，その上で PUCC のプラットフォームを実装する．

センサゲートウェイは，様々なセンサデバイスが持つ状態変数などが記述されたメタデータを管理し，それらをオーバレイネットワーク上に提供する役割を持つ．また，様々なセンサデバイスから送られてくるセンサデータを監視し，Web サーバを通してユーザによって設定された連携ルールのイベント条件を満たしているかどうかの判定を行う．判定の結果イベント条件を満たしていた場合，イベントが発生したことを Web サーバへ通知する．以下では，センサゲートウェイの構成部について説明する．

センサデータ取得部

様々なセンサデバイスのセンサデータを取得するモジュールである．得られたセンサデータはイベント検知部に送られ，ユーザによって設定された条件を満たすかどうかの判定に使われる．

イベント検知部

実際にイベントの判定を行うモジュールである．センサデータ取得部から送られてくるデータが，イベント管理部に登録されているイベントの条件を満たすかどうかを判定する．センサの値が条件を満たした場合，イベントが発生したことを通知する．

イベント管理部

ユーザによって設定された判定すべきイベントを管理するモジュールである．Web サーバを通してクライアントデバイスからのイベント購読要求を受けた際に，イベント管理部でそのイベントに ID が振り分けられる．イベント発生時の通知にはこの ID 情報が含まれており，ユーザはどのイベントが発生したかを判断できる．

イベント解析部

クライアントデバイスからのイベント購読要求を受け，そのイベントを解析するモジュールである．PUCC ライブラリによって受け取られたイベント購読要求はまずイベント解析部に送られ，そこで解析を行い，得られたイベントはイベント管理部で管理される．

3.1.2　家電ゲートウェイ

家電機器においてもセンサデバイスと同様に，家庭用防犯カメラなどの小型なデバイスへの PUCC プラットフォームの実装は困難であり，様々な家電機器を扱うことも考慮して家電ゲートウェイを PUCC ノードとして利用する．

家電ゲートウェイは各家電機器のメタデータを管理しており，それらをオーバレイネットワーク上に提供する役割を持つ．また，イベント検知で検知する対象が照明の ON/OFF などの家電機器の状態であったり，サービス実行の際にそれら家電機器に対してアクチュエータとしてサービス実行の処理を行う．

3.1.3　クライアントデバイス

センサや家電のデバイス・サービス情報の閲覧や，センサイベントと家電のアクションの関係を設定するための端末である．クライアントデバイスに Android などの高性能な端末を用いれば，その上に PUCC プラットフォームを実装して PUCC ノードとすることも可能だが，Web サーバを介して通信がやりとりできるためクライアントデバイスにはブラウザ機能さえあれば任意の場所からデバイス探索や家電の操作，連携設定などが可能となる．

3.1.4　Web サーバ

デバイスの連携設定を行うクライアントデバイスと，センサ・家電ゲートウェイとの通信を仲介するのが Web サーバである．Web サーバのアーキテクチャを**図 3** に示す．Web サーバを介して処理を行うことでユーザはいつでもどこでもすぐにデバイスのメタデータの参照やデバイスの連携設定を行うことができる．クライアントデバイスが Web サーバにアクセスし，実際にメッセージをゲートウェイに送信する際には，Web サーバが PUCC のミドルウェアを利用してメッセージ処理を行う．クライアントデバイスとは HTTP で，ゲートウェイなどの他の PUCC ノードとは PUCC プロトコルで通信を行う．Web サーバは大きく分

けて以下の 2 つから構成されている．

- Apache 2.2.14
- Mars（PUCC Middleware）

Web サーバには Apache を利用する．Apache は自宅サーバなどでも用いられるなど幅広く利用されている Web サーバソフトウェアであり，複数の OS をサポートしているため，今回はサーバ処理向きの Ubuntu（Linux）上で動作させる．また，Web サーバには Apache と並行して Mars のプログラムも動作させる．ここでの Mars は，Apache 内の PHP プログラムがクライアントデバイスより受け取ったリクエストメッセージをオーバレイネットワーク上に流すために必要なリレーを行うためのプログラムである．Apache の PHP プログラムが生成した XML 形式のメッセージを Mars へ渡し，Mars がそのメッセージを HTTP on PUCC の仕様にしてオーバレイネットワークへと流し，またそのレスポンスメッセージを PHP プログラムへと返す役割を担う．Web サーバとの UI は，デバイス・サービスのメタデータをもとに，PHP プログラムが必要な情報を取り出して GUI を動的に生成する．以下では，Web サーバの構成部について述べる．

(1) Apache

各種 API（Subscribe API, Invoke API, Discover API など）

クライアントデバイスによる要求によってそれぞれの処理を行うモジュール．生成された GUI をもとにユーザは Subscribe や Invoke，Discover などのような操作を行うので，その内容が Web サーバへ送信された際にそれぞれ対応したモジュールでメッセージを受け取る．メッセージを受け取ったモジュールはそれぞれ内容に合った XML を生成し，PUCC メッセージ変換部へ渡す．

PUCC メッセージ変換部

生成された XML を受け取り，それを HTTP 形式にして Web サーバ内の Mars が持つ Web サーバ仲介部へ渡すモジュール．Mars からのレスポンスも受け取って返す役割も持つ．

(2) Mars

Web サーバ仲介部

Apache の PUCC メッセージ変換部から送られてくる XML を処理するモジュール．Apache から送られてきたメッセージを判断し，適切なものだと判断したらそれを PUCC ミドルウェアへ渡す．また，他の PUCC ノードからのレスポンスを Apache のプログラムへ返す役割も持つ．

PUCC ミドルウェア

Web サーバ仲介部から受け取った XML ファイルを実際に他の PUCC ノードへ送信するモジュール．また，送信した後のレスポンスを受け取って再び Web サーバ仲介部へ返す役割も持つ．この PUCC ミドルウェアを利用してデータを送受信する場合は，ミドルウェア上の Subscribe 処理部や Discover 処理部などそれぞれ対応したモジュールでリクエスト・レスポンスなどの処理を行う．

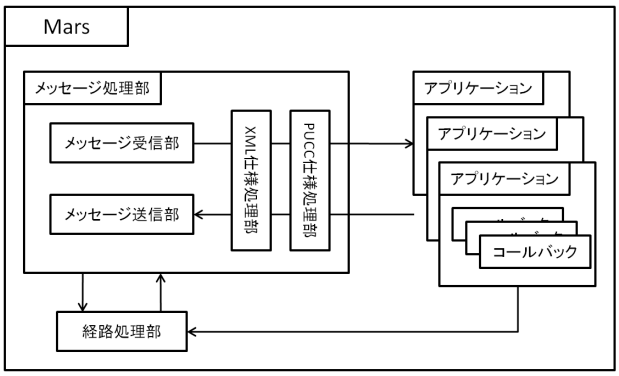

図 4　Mars アーキテクチャ
Fig. 4　Mars architecture.

3.2　Mars

我々が開発した Mars とは，オーバレイネットワークを構築して様々なネットワークを接続し，デバイス間の通信を行うためのミドルウェアである．オーバレイネットワーク上の通信，デバイス・サービスの情報の記述などは PUCC の技術仕様に準拠して実装した．Mars のアーキテクチャを**図 4** に示す．PUCC ミドルウェアであるこの Mars をセンサゲートウェイ，家電ゲートウェイ，Web サーバそれぞれに実装して PUCC ノードとすることで，PUCC プロトコルを用いていつでもオーバレイネットワークに参加することができ，オーバレイネットワーク上で様々なデバイスと相互接続することが可能となる．オーバレイネットワークに参加している限り，クライアントデバイスは IP や Zigbee などの下位リンクに左右されず，どこに移動してもサービスが利用可能である．以下では，Mars の主な構成部について述べる．

メッセージ処理部

外部との通信機能を提供する．メッセージ受信部は受信を扱い，受信データから PUCC データの抽出を行ってアプリケーションに渡し，処理結果をレスポンスとして送信する．メッセージ送信部は送信を扱い，アプリケーションから渡された PUCC データを送信する．

経路処理部

ルーティングテーブルを保持し，メッセージを送信する際の候補デバイスを選択するために利用される．

3.2.1　Mars が提供する機能

Mars は，ネットワーク上のデバイスを制御するための機能として以下の 4 つの機能を備えている．これらの機能は API を通じてミドルウェア上のアプリケーションが利

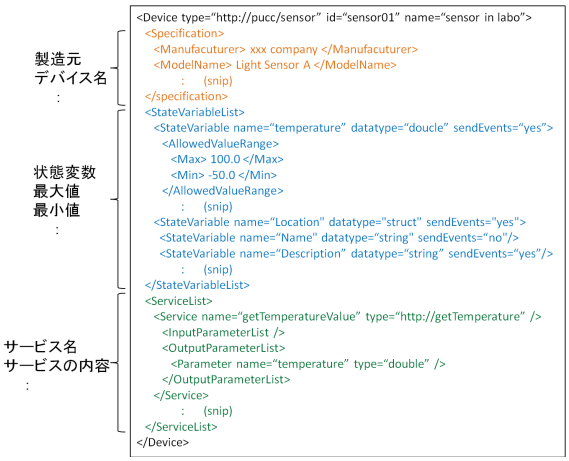

図 5　PUCC デバイスメタデータの例
Fig. 5　Example of PUCC device metadata.

用可能である．

デバイス・サービス探索機能

Discover メソッドにより，探索条件を指定し，条件を満たしたデバイス・サービスを探索する．この機能により，デバイスの情報，デバイスが持つサービスなどを記述したメタデータを取得できる．

イベント購読機能

Subscribe メソッドにより，取得したメタデータをもとに PUCC ノードに対してイベントの購読を行う．また，Unsubscribe メソッドにより，購読したイベントのキャンセルも可能である．

イベント通知機能

Notify メソッドにより，イベントの通知を行う．購読元への通知は msec 単位での間隔を指定できるほか，設定したイベント発生時に通知させることも可能である．

サービス実行機能

Invoke メソッドにより，取得したメタデータをもとに PUCC ノードに対してサービスの実行を行う．実行したいサービスをメッセージ内に記述し送信することで，このメッセージを受け取ったデバイスのサービスを実行することができる．

3.2.2　メタデータ

例として温度センサ搭載のセンサデバイスのメタデータを図 **5** に示す．メタデータは，デバイスの名称，種類，提供するサービスなどのデバイスに関する各種情報を記述した XML ファイルである．提供するサービスが複数ある場合でも，その数だけタグを用意し，それらをリストタグの中に記述することですべての情報を把握することができる．デバイス・サービスの情報が記述されたメタデータを持つことで，自分とは異なるネットワークに接続されたデバイスの情報を知ることができ，またそのメタデータを利用することで他のネットワーク内の様々なデバイスのサービスを実行したりすることが可能となる．以下では，主な要素について説明する．

Specification 要素

デバイスの名称や製造番号，製造元などのデバイス自体に関する静的な情報をそれぞれタグで挟んで記述する．ほかにも，デバイスの情報が記載された URL やデバイスの UUID などの詳細も記述することができる．

StateVariableList 要素

電源や動作モード，とりうる値などのデバイスの状態変数をリストタグの中に記述する．センサデバイスでは，取得可能なセンサの種類を状態変数として扱う．温度センサを搭載したセンサデバイスの場合，図 5 に示すように temperature という名前の状態変数を持つ．また，この状態変数は double 型で −50〜100 の範囲の値をとりうることになる．さらに，要素内に位置情報を記述することもできるため，デバイスの移動も把握することが可能である．

ServiceList 要素

イベント検知などのデバイスが提供可能なサービスの内容をリストタグの中に記述する．DVD デッキなどの場合，提供可能なサービスとして録画，再生，早送りなどを記述する．温度センサ搭載のセンサデバイスが現在の温度を取得するというサービスを持っていた場合，図 5 のように記述される．この例では，getTemperature というサービスで現在の温度情報を取得できる．

4.　実装

本章では，3 章の設計をもとに実装したプロトタイプシステムの概要・評価について述べる．実装システムの全体図を図 **6** に示す．今回我々は，PUCC ミドルウェアである Mars を開発し，センサゲートウェイ，家電ゲートウェイ，Web サーバそれぞれに Mars を実装した．センサゲートウェイにはセンサデータ取得，イベント検知，イベント管理，イベント解析のモジュールを実装し，センサデバイスのメタデータを持たせた．家電ゲートウェイにはサービス管理，サービス実行のモジュールを実装し，家電機器のメタデータを持たせた．Web サーバの Apache 上には GUI 生成，XML 生成（各種 API 処理），ソケット通信（PUCC メッセージ変換）のモジュールを実装した．Web サーバの Mars 上にはソケット通信，メッセージ処理（Web サーバ仲介）のモジュールを実装した．本システムでは連携デバイスとして，センサデバイスと家電機器を対象とする．

PUCC のコンソーシアムでは PUCC プラットフォームのプロトコルやメタデータなどの仕様書しか提供していないため，今回はそれらの仕様書をもとに我々が一から，デバイス相互接続基盤ミドルウェアである Mars を開発した．Mars は Java 言語で開発し，総ステップ数は 6,297 である．

表 1　利用デバイスの仕様
Table 1　Specication of used devices.

	クライアントデバイス	Web サーバ	センサゲートウェイ	家電ゲートウェイ
製品名 (マザーボード)	Google Nexus One	acer Aspire L5100	自作 PC (ASUS P8P67 REV 3.0)	自作 PC (ASUS P8P67 REV 3.0)
CPU	Qualcomm Snapdragon QSD8250 (1.00 GHz)	Athlon(tm) 64*2 Dual Core Processor 4400+ (1.00 GHz)	Core i5 2,500 K (3.30 GHz)	Core i5 2,500 K (3.30 GHz)
OS	Android 2.3.4	Ubuntu 10.04	Ubuntu 10.04	Ubuntu 10.04
メモリ	512 MB	1 GB	8 GB	8 GB

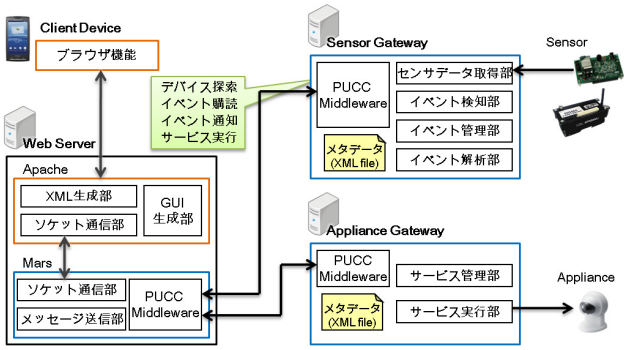

図 6　実装システムの全体像
Fig. 6　Overview of implemented system.

この Mars をセンサゲートウェイと家電ゲートウェイのそれぞれに実装し，それぞれ独自のアプリケーション開発も行った．センサゲートウェイでは，Mars 上で動くアプリケーションとしてセンサデータの取得やイベント検知などを行うモジュールを Java 言語で開発した．それらの総ステップ数は 1,187 である．家電ゲートウェイでは，Mars 上で動くアプリケーションとしてサービス管理やサービス実行を行うモジュールを Java 言語で開発した．それらの総ステップ数は 1,471 である．Web サーバにも Mars を組み込んだが，Web サーバは Proxy として働くため Mars だけでなく Web サーバソフトウェアとして Apache 2.2.14 も組み込んだ．Web サーバでは，Mars 上で動くアプリケーションとして Web サーバとの仲介を行うモジュールを Java 言語で開発した．それらの総ステップ数は 562 である．さらに，Apache 上には，クライアントデバイスがブラウザを通してアクセスしてきた際に表示する設定画面や Subscribe，Invoke，Discover などの処理を行うモジュールをそれぞれ PHP 言語で開発した．それらの総ステップ数は 665 である．クライアントデバイスには，HTTP で Web サーバにアクセスできるブラウザ機能を持ち，かつ通信性能に優れた Android 端末を用いた．また，これらの構成要素はすべて 3 章で述べた各種機能を実装している．

4.1　利用デバイス

センサデバイスと家電機器を連携させるシステムの開発において，実際に利用したデバイスを**表 1** に示す．また，

表 2　連携対象機器の持つ状態変数とサービスの定義
Table 2　State variables and service definitions of target devices.

	センサデバイス	家電機器
製品名	Renesas Solutions 製の センサボード	Sony 製の SNC-P5 ネットワークカメラ
状態変数	温度，照度，モーション	-
サービス	温度値取得，照度値取得 モーション値取得	ズーム，パン，チルト 画像取得

連携対象となるセンサデバイスと家電機器の持つ状態変数とサービスを**表 2** に定義した．センサデバイスは温度，照度，モーションを状態変数として持ち，これらの値を取得することができる．ネットワークカメラはズーム，パン（水平移動），チルト（垂直移動），画像取得をサービスとして持つ．

4.2　動作シナリオ

今回開発したプロトタイプシステムは，センサデバイス，センサゲートウェイ，家電機器，家電ゲートウェイ，Web サーバ，クライアントデバイスから構成されている．システムの動作シナリオとして，実際に以下の連携ルールを設定してデバイス間の連携を実現する．

- イベント：センサデバイスの照度が一定以下
- サービス：ネットワークカメラをズーム

センサデバイスのイベント検知をセンサゲートウェイが行い，イベントが発生した際には家電ゲートウェイがサービス実行の処理を行う．このデバイス連携を行ううえで必要な各種機能をそれぞれに実装した．

4.3　評価

3 章で述べたように，本システムを利用することでユーザは異種ネットワークデバイスの連携設定をシステム起動後すぐに行うことができる．デバイス連携に必要なコネクション確立やデバイス探索などの前処理を事前にサーバ側で行うことで，ユーザのより直感的で効率的な操作を可能としているが，Web サーバを介することで発生するオーバヘッドが前処理にかかる時間よりも大きくなってしまってはならない．それらをふまえたうえで今回は，以下の 2

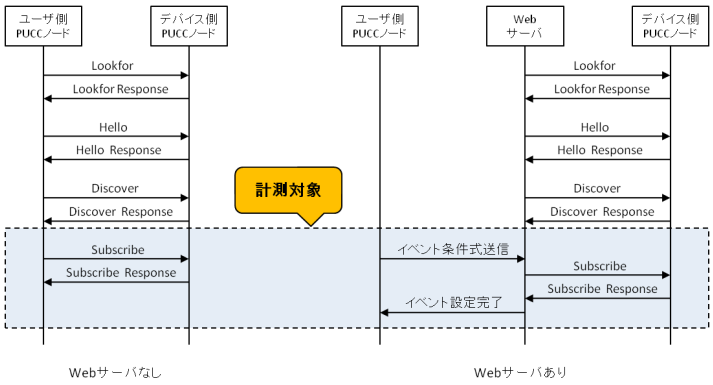

図 7　イベント設定までの処理シーケンス
Fig. 7　Sequence diagram of event setting.

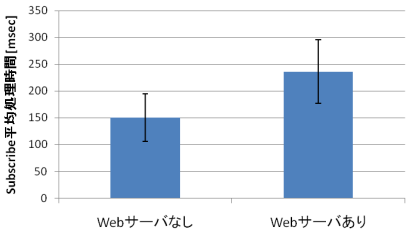

図 8　Subscribe メッセージ受信時の平均処理時間
Fig. 8　Average processing time of subscribe message.

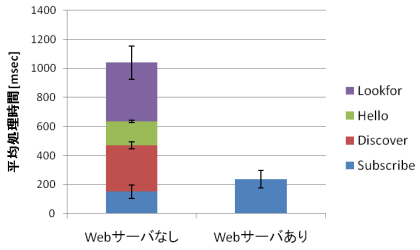

図 9　ユーザによるイベント設定までの処理時間
Fig. 9　Processing time until user completes an event setting.

点を検証し，異種ネットワークデバイス連携システムにおける Web サーバの有用性を示すことを目的にシステム評価を行った．今回我々が提案する Web サーバを用いた異種ネットワークデバイス連携システムと，以前我々が開発した Web サーバを用いずにクライアントデバイス自体（Android 端末）を PUCC ノードとしたシステムの 2 つを評価対象とし，デバイス連携に必要な通信にかかる処理時間にどの程度違いが現れるかの比較評価を行った．

1 つ目は，Web サーバを用いたシステムと Web サーバを用いないシステムで通信時にどの程度のオーバヘッドが発生するかを検証した．**図 7** にユーザがイベントを設定するまでの処理シーケンスを示す．図 7 に示した枠内を処理時間の計測対象とする．Web サーバありのシステムと Web サーバなしのシステムの Subscribe メッセージ送信からレスポンスが返るまでの処理時間を計測し，それらの値を比較した．**図 8** に Web サーバありの場合となしの場合で，それぞれ Subscribe メッセージを送信したときの結果を示す．2 つ目は，Web サーバありの場合となしの場合で，ユーザがシステム起動からイベント設定を完了するまでにかかる処理時間を計測し，Web サーバを介することでどの程度冗長を省いて効率的にイベント設定が行えるかを検証した．その結果を**図 9** に示す．これら 2 つの評価実験は表 1 の環境のもと行った．

図 8 より，Subscribe 処理に Web サーバなしの方は約 150 msec かかり，Web サーバありの方は約 230 msec かかっていることが分かる．Web サーバありの方は，Web サーバなしの方よりも約 80 msec 余分に処理時間がかかることが確認できた．これは，Web サーバを介する分，ユーザからのリクエストメッセージおよびレスポンスメッセージが必ず Web サーバに渡されてから送られるためである．また，図 9 より，ユーザがイベント設定するまでにかかる時間は，Web サーバなしの方は，ID などの対応付けをする Lookfor 処理に約 400 msec，コネクションを確立する Hello 処理に約 150 msec，Discover 処理に約 300 msec，Subscribe 処理に約 150 msec かかり，すべて合わせると約 1,000 msec かかることが分かる．一方，Web サーバありの方は Subscribe 処理にかかる約 230 msec だけと，Web サーバなしの場合の約 5 分の 1 程度しかかからないことが確認できる．これは，図 7 で示した処理シーケンスより，Web サーバありの場合は，本来必要な Lookfor から Discover までの処理はすべて Web サーバ側が行っていることから説明できる．つまり，これらの前処理はすべて事前に Web サーバが行っているため，ユーザのイベント設定時には Subscribe 処理だけ行えばよいためである．

以上の結果から，1 つ 1 つのメッセージ処理に関しては，Web サーバを介する分提案システムの方がわずかに処理時間がかかってしまうことが分かった．しかし，Web サーバを介した場合はサーバが定期的に Discover などの前処理を行っておくことができる．ユーザがイベントを設定したいという場合，Web サーバを用いないシステムだと起動時に毎回 Lookfor から Discover までの前処理を行わなければならない．しかし，Web サーバを用いたシステムでは

ユーザはそれらの前処理は意識することなく，オーバレイネットワーク上のデバイスの状況をすべて知った状態ですぐにイベント設定から操作を行うことが可能である．Web サーバを介することで，通信時には約 80 msec 余分に処理時間がかかってしまうが，イベント設定までに必要な前処理時間の約 850 msec を削減することができる．これにより，ユーザにとって冗長な処理を省き，起動時コストが大幅に削減されるため，通信の仕組みを知らなくても，より直感的にシステムを扱うことが可能となる．また，クライアントデバイスにミドルウェアなどの大きなプログラムを組み込む必要がなく，既存のブラウザ機能のみで通信が可能となるため，クライアントデバイスへの負担が削減できる．これらの利点から，異種ネットワークデバイス連携システムにおける Web サーバは有用的であると考える．

5. おわりに

本論文では，異なるネットワーク上のセンサデバイスや家電機器などを PUCC によって提案されている異種デバイス相互接続技術を用いて，Web サーバを通して統一的に制御するシステムの提案，実装および評価を行った．ユーザはブラウザ機能を有した携帯端末さえあれば，いつでも Web サーバにアクセスすることができ，端末からセンサデバイスの情報を取得して閲覧したり，家電機器の遠隔操作を行ったりすることが可能である．また，あるセンサデバイスに対してイベントを設定し，イベントが発生したときに特定の家電機器を自動的に動作させるといった，デバイス連携サービスも可能である．評価では，Web サーバを介することで通信時にどの程度のオーバヘッドが発生するかの検証およびユーザがイベントを設定するまでにかかる処理時間を検証し，提案システムにおける Web サーバの有用性を示した．Web サーバを介することで約 80 msec のオーバヘッドはあったものの，起動時コストの削減やクライアントデバイスへの負担削減を考慮すると Web サーバの効果は大きなものだと考えられる．

今後の課題としては，デバイス相互接続基盤に対する評価や，扱えるデバイスの数を増やして複合イベントの実装，それにともなうサービス実行の衝突回避のための排他制御などがあげられる．また，デバイスの数が膨大に増えた際には，ユーザ自身が多くのデバイスのリストの中から探して選択しなければならないという手間が発生することが考えられるため，その対策としてデバイスの位置情報や過去の使用履歴などでフィルタリングできるような仕組みも必要である．さらに，プロトタイプでは実装していない，ASP サービスとして提供可能なシステムとすることについても検討する予定である．

参考文献

[1] UPnP, available from ⟨http://upnp.org/⟩.
[2] HAVi, available from ⟨http://www.havi.org/⟩.
[3] GSN, available from ⟨http://sourceforge.net/apps/trac/gsn/⟩.
[4] PUCC, available from ⟨http://www.pucc.jp/⟩.
[5] 小田　正：AV 機器相互運用のための通信ミドルウェア技術（HAVi），Sharp Technical Journal, No.7, シャープ技報，Vol.75, pp.45–50 (1999).
[6] 樋口正生，盛岡隆一郎，稲垣勝利，戸崎明宏：家庭内 AV ネットワーク技術「HAVi」の概要，情報処理，PIONEER R&D, Vol.11, No.2.
[7] 加藤悠一郎，峰野博史，角野宏光，石川憲洋，水野忠則：携帯電話を用いた異種ネットワークデバイス連携システムの開発，情報処理学会研究報告（ユビキタスコンピューティングシステム），2010-UBI-25, Vol.2010, No.22, pp.1–6 (2010).
[8] 高山洋史，小板隆浩，佐藤健哉：機器連携におけるネットワークミドルウェア統合システムの提案（ネットワーク），情報処理学会研究報告，2008-EMB-10, Vol.2008, pp.59–65 (2008).
[9] 井上貴仁，服部篤人，田中宏一，河口信夫，西尾信彦：適応的変換機構を用いた異種サービス連携の実現，情報処理学会研究報告，2007-MBL-40, Vol.2007, pp.75–82 (2007).
[10] 幸島明男，池田　剛，井上　豊，車谷浩一：センサイベント指向のサービス連携ミドルウェア：(SENSORD)，情報処理学会研究報告，2006-UBI-12, Vol.2006, pp.37–44 (2006).
[11] Aberer, K., Hauswirth, M. and Salehi, A.: Global Sensor Networks, Technical report LSIR-REPORT-2006-001.

田中 剛　（学生会員）

1989 年生．2012 年静岡大学情報学部情報科学科卒業予定．同年静岡大学大学院情報学研究科修士課程進学予定．コンシューマシステム，ユビキタス Web インタフェース，対面コミュニケーション支援に関する研究に従事．

伊藤 崇洋

1986 年生．2011 年静岡大学大学院情報学研究科修士課程修了．同年株式会社サイバーエージェント入社．大学では，異種ネットワークデバイス連携 Web システム，Web アプリケーションに関する研究に従事．

加藤 悠一郎

1987 年生．2011 年静岡大学大学院情報学研究科修士課程修了．同年住商情報システム株式会社（現，SCSK 株式会社）入社．大学では，ユビキタスコンピューティング，異種ネットワークデバイス連携システムに関する研究に従事．

峰野 博史　（正会員）

1974 年生．1999 年静岡大学大学院理工学研究科修士課程修了．同年日本電信電話（株）入社．NTT サービスインテグレーション基盤研究所を経て，2002 年 10 月より静岡大学情報学部助手．2006 年九州大学大学院システム情報科学府博士（工学）．2011 年 4 月より，静岡大学情報学部准教授．ヘテロジニアスネットワークコンバージェンスに関する研究に従事．電子情報通信学会，IEEE，ACM 各会員．

水野 忠則　（フェロー）

1945 年生．1969 年名古屋工業大学経営工学科卒業．同年三菱電機（株）入社．1993 年静岡大学工学部情報知識工学科教授．1996 年同情報学部情報科学科教授．2006 年同創造科学技術大学院教授．2011 年より愛知工業大学教授．工学博士．情報ネットワーク，モバイ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関する研究に従事．著訳書としては『コンピュータネットワーク』（日経 BP），『モダンオペレーティングシステム』（ピアソン・エデュケーション）等がある．電子情報通信学会，IEEE，ACM，Informatics Society 各会員．情報処理学会フェロー．