Android 端末を用いた 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] 伊藤崇洋[†2,*1] 加藤悠一郎[†2,*2]
峰野博史[†1] 水野忠則[†3,*3]

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，Android 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，Android 端末を用いて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時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．

Development of Heterogeneous Network Devices Cooperating System with Android Devices

Go TANAKA,[†1] Takahiro ITO,[†2,*1] Yuichiro KATO,[†2,*2]
Hiroshi MINENO[†1] and Tadanori MIZUNO[†3,*3]

In this paper, we describe the system which controls various devices such as sensor devices and home appliances on the heterogenous network. This system can deal with a unified devices by making devices on the networks where communication protocol is different to interconnect and cooperate, and it is possible to monitor the sensing data and control the home appliances with Android devices. In addition, they all communicate through the Web, it is also possible to implement the device cooperating service such as setting a value for that sensor with Android devices, and when the sensor shows its value, it controls a specific appliance.

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．それに伴い，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ー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同士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，身の周りには複数のセンサが常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，通信プロトコルの違いからある用途のために設置されたセンサは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] 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，さらにそれら全てのデバイスを統一的に扱う

†1 静岡大学情報学部
Faculty of Infomatics, Shizuoka University
†2 静岡大学大学院情報学研究科
Graduate School of Infomatics, Shizuoka University
†3 静岡大学創造科学技術大学院
Graduate School of Science and Technology, Shizuoka University
*1 現在，サイバーエージェント
Presently with Cyber Agent, Inc.
*2 現在，住商情報システム
Presently with Sumisho Computer Systems 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

ことで，デバイスの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．

本論文では，全 5 章で構成される．次の第 2 章では，関連研究とその課題について述べる．第 3 章では，システムの提案を行う．第 4 章では，実装と評価について述べる．最後に第 5 章にて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 とその周辺機器を始めとし，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．UPnP はインターネットをベースとしたオープンな標準をベースとするプロトコルを定義しており，HTTP や SOAP などの技術を基本としている．つまり，下位通信プロトコルは IP 系であり，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ー・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挙げられる．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．

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．

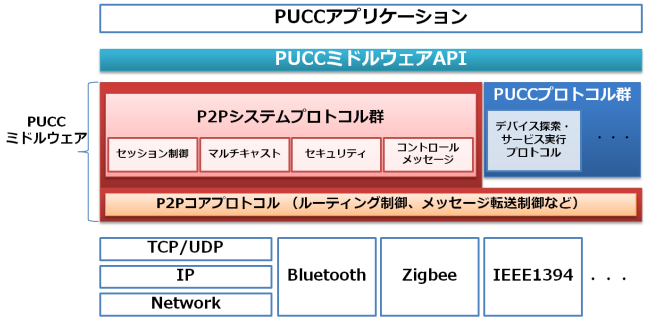

図 1 PUCC プロトコルスタック

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 を適応しており，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ことにおさまっており，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，あらゆる種類の全てのデバイスに対して一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応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，デバイスの相互接続性の課題を解決するために，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るようにする必要がある．さらに，デバイスの連携設定の際には，ユーザにとって冗長な処理やコストは可能な限り省き，効率的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，異種ネットワークに接続されたデバイスを下位通信プロトコルに左右されず，相互に接続するシステムの検討を行った．そして Web サーバを介して携帯電話端末からいつでもどこでもセンサや家電のデータを閲覧したり，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させるといったデバイス間の連携設定を可能とするシステムを提案する．また，本システムでは Web サーバを介すことでユーザにとって冗長な処理はサーバ側に任せるような仕組みを検討した．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

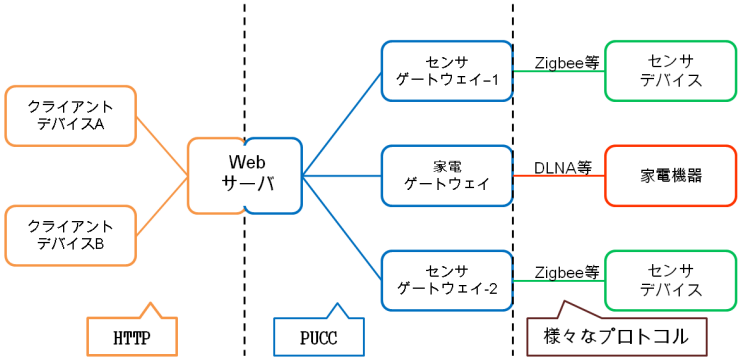

図 2 システム概要図

メタデータを管理することとなる．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，家電ゲートウェイ及び 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，単純にセンサデバイスに 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応シーンを考慮されて作られることが多く，それらは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のが照明の 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・サービス情報の参照や，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取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．

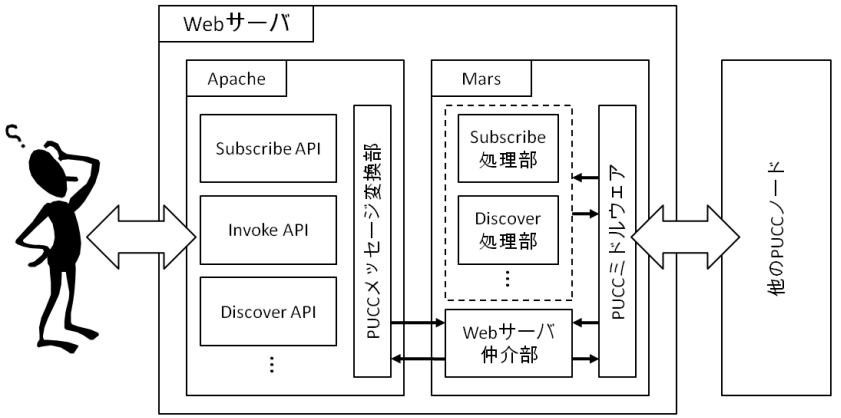

図 3 Web サーバアーキテクチャ

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 サーバの構成は以下の通りである．

- 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 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 を基にユーザは Subscribe や Invoke，Discover などのような操作を行うので，その内容は Web サーバへ送信される際にそれぞれ対応したモジュールへメッセージが渡される．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．また，ユーザが要求を行うための GUI は，受け取ったデバイスのメタデータによって動的に生成されるため，デバイスの追加や削除なども容易に対応可能である．

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 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 ノードへと通信を行うモジュール．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．

3.2 Mars

我々が開発した Mars とは，様々なネットワークをオーバレイネットワークを構築して接続し，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．

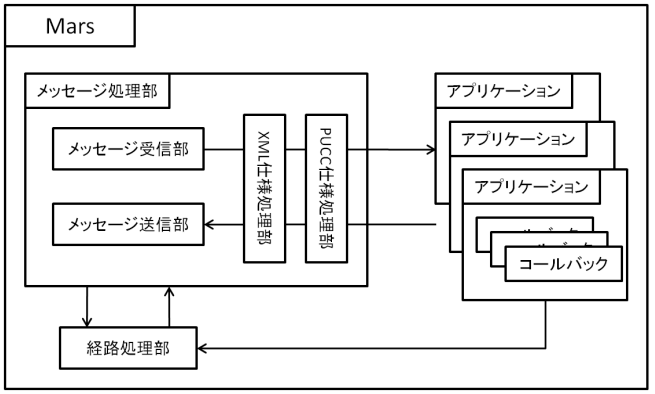

図 4 Mars アーキテクチャ

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

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．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，またそれ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

デバイスの名称や位置，製造番号や製造元などのデバイス自体に関する静的な情報を記述する．

StateVariableList 要素

電源や動作モード，取り得る値などのデバイスの状態変数をリスト上に記述する．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．

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がリスト上に記述する．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．

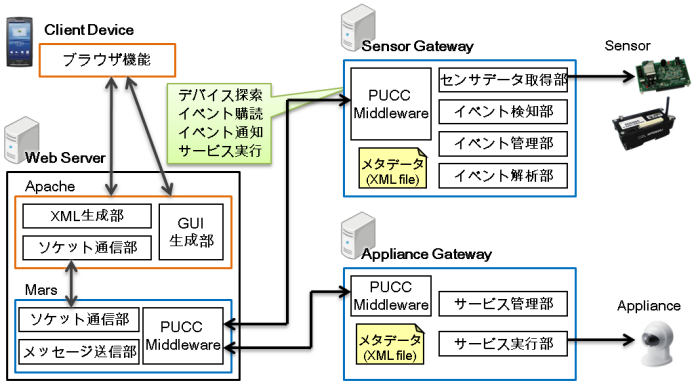

図 5 実装システム

4. 実　　装

本章では，第 3 章の設計を基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 5 に示す．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．

センサゲートウェイと家電ゲートウェイには，PUCC プラットフォームを基に我々が実装したデバイス連携基盤ミドルウェア Mars とサービス用ソフトウェアを組み込んだ．Web サーバは Proxy として働くため，Web サーバソフトウェア Apache と Mars を組み込んだ．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，かつ高性能な Android 端末を用いた．また，これらはすべて第 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・家電機器連携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，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．

表 1　実装構成機器

	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 (1.00GHz)	Athlon(tm) 64*2 Dual Core Processor 4400+ (1.00GHz)	Core i5 2500K (3.30GHz)	Core i5 2500K (3.30GHz)
OS	Android 2.3.4	Ubuntu 10.04	Ubuntu 10.04	Ubuntu 10.04
メモリ	512MB	1GB	8GB	8GB

表 2　連携対象機器の持つ状態変数とサービスの定義

	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

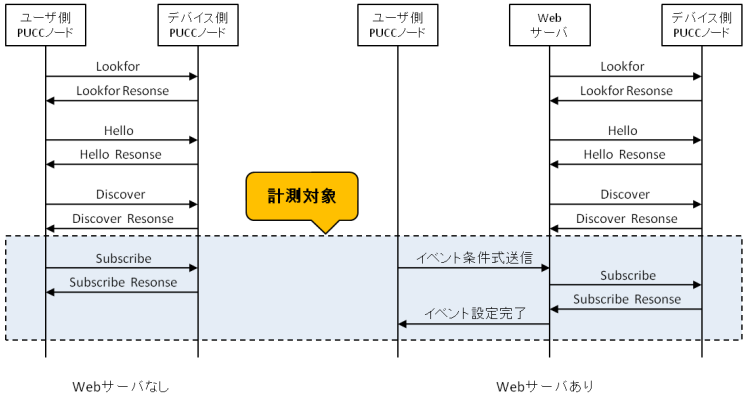

図 6　イベント設定までの処理シーケンス

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監視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上で必要な各種機能をそれぞれに実装した．

4.3　評　　価

提案する異種ネットワークデバイス連携システムと，従来の Web サーバを用いないシステムとの比較評価を行った．比較項目は，二つのシステム間で通信時にどの程度のオーバーヘッドが発生するのかを比較評価により検証した．図 6 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 6 に示した枠内を処理時間の計測対象とする．評価実験は表 1 の環境のもと，提案システムと従来システムの 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 7 に Web サーバありの場合となしの場合とでそれぞれ Subscribe メッセージを送信した時の結果を示す．また，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 8 に示す．

図 7 より，Subscribe 処理に Web サーバなしの方は約 150msec かかり，Web サーバありの方は約 230msec 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 80msec 処理時間がかかることが確認できた．これは，Web サーバを介す分，ユーザからのリクエストメッセージ及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 8 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 400msec，コネクションを確立する Hello 処理に約 150msec，Discover 処理に約 300msec，Subscribe 処理に約 150msec かかり，すべて合わせると約 1000msec 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 230msec 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 6 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が分かる．つまり，これらの前処理はすべて前もって Webs 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 処理だけ行えば良いためである．

以上の結果から，一つ一つのメッセージ処理に関しては，Web サーバを介す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 Web サーバを常に起動したままにできるため，定期的に Discover などの前処理を行っておくことでオーバレイネットワーク上のデバイスの変化を把握することができる．ユーザがイベントを設定したいという場合，従来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，提案システムではそれらの処理をあらかじめサーバ側で行うことが可能となるため，ユーザは Lookfor から Discover までは意識することなく，イベント設定から操作を行うことが可能なので，処理時間を大幅に削減できる．これにより，ユーザにとって冗長な処理を省き，起動時コストの削減が可能となる．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．これらの利点は，この Web サーバを介すことによって生じた通信時のオーバーヘッドを十分に補えると考える．

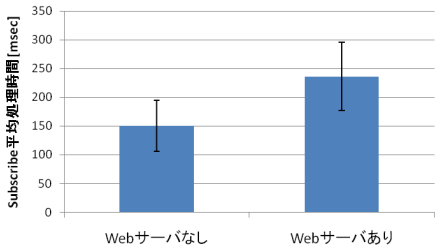

図 7　Subscribe メッセージ平均処理時間

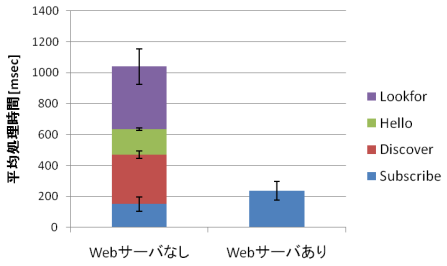

図 8　ユーザによるイベント設定までの処理時間

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，実装及び評価を行った．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時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及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，それに対する評価を行った．Web サーバを介すことで約 80msec 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そのオーバーヘッドは十分に補えるものだと考えられる．

今後の課題としては，扱えるデバイスの数を増やして複合イベントの実装及び評価，それに伴うサービス実行の衝突回避のための排他制御などが挙げられる．また，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 http://upnp.org/
2) HAVi http://www.havi.org/
3) GSN http://sourceforge.net/apps/trac/gsn/
4) PUCC 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 75 巻， pp.45-50(1999.12).
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(2010.3).
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(2008.11).
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(2007.2).
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).
11) Karl Aberer，Manfred Hauswirth，Ali Salehi ：Global Sensor Networks，Technical report LSIR-REPORT-2006-001