Vol.2014-CDS-10 No.10
2014/5/23

Web サービスとマルチデバイスのフレキシブルな連携方式の実現

渡部智樹[†1,†2]　高嶋洋一[†1]　杉村博[†2]　一色正男[†2]

家電やセンサなど，身の回りの任意のデバイスの状態を任意の Web サービスと関連づけて，その状態の管理や制御が可能な仕組みを提案する．これまでに，HEMS などの規定のデバイスに限定して制御するサービスや，コンテキストの変化に応じてサービスを提供する仕組みは存在していたが，ユーザが任意のデバイスの状態を関連づけてサービス利用時にデバイスと連携させる提案はなかった．本論文では，提案方式について説明した後，提案方式を実現するための設計指針をあげ，具体的なアーキテクチャとプロトタイプにより実現可能性を示し，さらに実際に利用可能なデバイスの状態数について実際の ECHONET Lite 家電を用いた調査および考察について述べる．

An Implementation of a Flexible Collaboration Method with A Web Service and Multiple Devices

TOMOKI WATANABE[†1,†2]　YOUICHI TAKASHIMA[†1]
HIROSHI SUGIMURA[†2]　MASAO ISSHIKI[†2]

We propose a collaboration method that is able to manage so as to relate a web service which the user is browsing and several states of appliances around a user. There are some methods that are focused on only limited devices as HEMS or are using context of devices. However, our proposal realizes that a user associates a web service to some state of some appliance and will be able to use the service in the same situation as at that time. After explaining the proposed method, design criteria for implementing our method and the architecture are described. A prototype system is created and shows the feasibility. And using the actual several appliances, we find out the number of devices and the states that can be applied.

1. はじめに

ECHONET Lite[1] や DLNA[2]の登場により，ネットワークに繋がる家電が増え，スマートフォンやタブレットなどから状態確認や操作実行ができるようになった．最近では Bluetooth や Zigbee などの省電力で通信可能なセンサも実現されている．さらには，WoT (Web of Things)[3]や IoT (Internet of Things)[4]と呼ばれる「モノのインターネット」が注目され，今後ますます身の回りの様々なモノがネットワークにつながり，それらを活用した多様なサービスが提供されるようになると考えられる．以降では，ネットワークにつながる家電やセンサ，モノを「デバイス」，これらの複数のデバイス群を「マルチデバイス環境」と称する．

このように，多様なサービスが増加すると，デバイスは様々なサービスや複数のユーザから利用されるため，あるユーザにとって期待していた設定や状態が変更されている事態が頻繁に発生すると予想される．例えば，父親が映画を視聴するときには，カーテンを閉め，シーリングライトを消し, 5.1ch のスピーカで出力するといったマルチデバイス環境を期待する．一方で，子どもがアニメを視聴する際は，カーテンを開け，シーリングライトを点灯し，音声吹き替えのモード設定を期待するかも知れない. このように，同じデバイスであっても使う人や使うサービスによって期待する状態は異なる．

そのため，サービスを利用したりコンテンツを選択したりする際に，利用するデバイスが期待している状態であるかを確認し，期待と異なる場合には自動的に期待した状態となるように制御する，といった対応が考えられる．しかし，サービス提供者側でこれらの設定することは難しい．なぜなら，各世帯により家庭内に配置されたデバイスの数や種類，またサービスと連携して利用したいデバイスやその状態が異なり，さらには同一世帯でもユーザによってこれらの関連付け方は異なると考えられるためである．

従来，状況変化に伴うサービス提示[5]や，サービス実行のために条件としてデバイス状態を利用する研究[6]などが行われていた．しかし，任意のサービスと任意のデバイスの状態をユーザが設定・登録し，利用するという研究はなされていなかった．

そこで我々は，各ユーザがサービス毎にマルチデバイス環境を設定し，サービス利用時に登録したデバイスの状態を確認し制御する方式を提案する．なお，対象とするサービスとしては，HTML5 の勧告により今後の普及や移行が見込まれる Web を使ったサービスに注目する．

提案方式を実現するにあたり，要件の抽出と実現可能性の検証，実際のマルチデバイス環境における有効性評価が

* †1 日本電信電話株式会社 NTT サービスエボリューション研究所
NTT Service Evolution Laboratories, NTT Corporation
†2 神奈川工科大学
Kanagawa Institute of Technology

ⓒ 2014 Information Processing Society of Japan

必要である．本論文では，実際のマルチデバイス環境においてシステムの実現可能性を確認することを目標とし，実装面と性能面から検証したので報告する．

以下，関連技術との差分，課題の抽出，そして提案方式の考え方を説明した後，提案方式を実現するための要件をあげ，具体的なアーキテクチャとプロトタイプにより実現可能性を示し，利用可能なデバイスおよび状態数 a)について所要時間の観点から考察する．

2. 関連技術

サービスと複数のデバイスを関連付ける研究として，サービス提供者が予めコンテンツに組み込んで提供するアプローチと，ユーザが設定するアプローチがある．

前者には例えば HEMS(Home Energy Management System) [7]と呼ばれるシステムがある．これは家庭内にある家電やセンサをネットワークで接続し，主として消費電力をグラフなどにより可視化して PC で簡単に見えるようにしたシステムである. PC の画面から見たり操作したりできるデバイスは既定のものに限られており，またユーザが他のサービスにこれらの家電の情報を関連付けることはできない．

一方後者のアプローチでは，ユーザが身の回りにあるデバイスを選択することができる．例えば西垣らの研究[5]では，家庭内のデバイスを効率良く操作するために，デバイスの状態をコンテキストとして捉え，操作実行の条件となるルールを簡潔に記述する提案がなされている．また前田らの研究[6]では，連鎖する複数のコンテキストからサービス選択を制御する提案がなされている．

これらの研究はセンサやユーザ行動などの状態変化を起点としてデバイスを制御したりサービスを提供したりすることを目的としているが，我々の目的とするところは Web サービスの利用を起点としており，この点において異なる．すなわち，任意の Web サービスと任意のデバイスの状態をユーザが関連づけて動作させる仕組みはこれまで提案されていなかった．

3. 課題

市販のデバイスに付属された専用リモコンだけでなく，ソフトウェアによるリモコン，あるいは複数ユーザによる共同利用などのために，ユーザがサービスを利用しようとしたときに周囲のデバイスが期待していた状態でなかったり，サービスの途中で思いもよらない動作をしてしまったりする場合がある．以下に具体的な例を挙げる．

- 動画を見るときには部屋を明るくし，英語の音声で聞こうとしていたのに変更されてしまっている．
- 電子書籍での読書など集中したいとき，周りの音をできるだけ静かにしたいので，TV の電源を OFF（あるいは消音），洗濯機やエアコン，ロボット掃除機を静音モードにしなければならない．
- ネットバンキングやオンライン学習を行っているとき，洗濯機や調理家電などの運転が終わり，作業を中断しなければならなくなる．

これらの問題を解決するためには，それぞれのサービスにデバイスの自動制御や状態確認を行う仕組みを導入すれば実現できる．しかしながら，これらはサービス提供者側の問題でありユーザが望むあらゆるサービスに導入するのは難しい．また導入しようとしても各ユーザ（家庭）でどのようなデバイスをどのような状態に設定すべきかについてはユーザが決めるべきことであり，サービス提供者側で事前に設定できない． したがって，サービスとデバイスの状態をユーザの好みに合わせて関連付けられる仕組みが必要となる．

我々は，ネットワーク上のサービスとネットワークで制御可能なデバイスであれば，任意のサービス，任意のネットワークに適用することを目標とするが，本論文ではサービスは Web によるサービス（アプリ），デバイスは ECHONET Lite に対応するデバイスを対象とする．その理由としては，2014 年に HTML5 が W3C により勧告[8]されるのを受けて，今後 Web を使ったアプリ（Web アプリ）に移行していくと考えたためである. また ECHONET Lite は，2011 年に HEMS における公知な標準インターフェースとして推奨され, また 2013 年には国際標準化規格として承認されたことから，今後の普及が見込まれるためである．

以下，上記の問題を解決する方式について提案する．

4. 提案方式

4.1 概要

我々は, ユーザが利用している Web サービスとその時つながっているデバイスの状態を任意に選択し登録しておくことにより, 登録している Web サービスを利用する際には, 登録情報と現在の状態との照合を行い，デバイスを制御したり，あるいはサービスの利用を制限したりするといった仕組みを実現する．

4.2 設計指針

提案方式を実現するための設計指針を示す．

① Web ブラウザから LAN 上に接続する全てのデバイスの状態を確認できること
② ユーザが閲覧している Web サービスへの影響を最小限とすること
③ 提案方式の処理時間を最小限とすること

以下順に説明する．

まず①について説明する．ユーザの端末がつながっている LAN 上の全ての ECHONET Lite デバイスを Web ブラウザで把握したり，その個々のデバイスの状態を確認したりするためには，ECHONET Lite のプロトコルを Web ブラウ

a) ECHONET Lite の家庭用エアコンでは，動作状態，運転モード設定，温度設定地，ON タイマ予約設定など，数十にわたる状態取得のコードが規定されている．

ザ上で扱える必要がある．そのためにはプラグインにより機能拡張する方法もあるが，利用できる Web ブラウザが限定されてしまう．そのため，芦村ら[9] や松村ら[10]の研究で行われているように，Web ブラウザとデバイスとの間を仲介する仕組みを選択する方法が現実的である．

次に②を説明する．ユーザは本来サービスを享受するために Web を閲覧しているため，その行為をできるだけ阻害しないようにすべきである．ユーザが閲覧している Web 画面が別な画面に移動してしまうと閲覧中のサービスを継続できなくなってしまう可能性があるため，現在閲覧中の Web ページを遷移することなく実現する必要がある．

③は，提案方式の処理に要する時間が長くなると，ユーザが不快感を抱き，提案方式だけでなく元のサービスさえも利用しなくなってしまう可能性があることに配慮したものである．Web のページ遷移において許容される待機時間については設計指標が提示されており，ネットワークが低速だった従来は 8 秒とされていたが，高速化された昨今では 3 秒とされている[11]．これに習って，提案方式でも 3 秒以内で処理を完了すべきとする．

以上のように，①②については実装面でのシステム要件，③についてはシステムの性能面での要件となる．以降，システム要件と性能要件との 2 つに分けてその実現性を説明する．

4.3 システム要件とアーキテクチャ

本節では前節で述べた設計指針①②に基づく方式のアーキテクチャを示す．まず前節の設計指針の検討からシステム構築要件として以下のように整理した．

① Web ブラウザとデバイスとを仲介する構成で実現
② 現在閲覧中の Web ページを遷移させないこと

これらを満たすアーキテクチャを図 1 に示す．

①を満たすため，図 1 のように家庭内 LAN 上にホームサーバを設置する．このホームサーバは，(1)Web との対話 I/F，(2)家庭内デバイス群の状態確認，(3)設定するデバイス状態の登録・管理と照合，の機能を備える．なお，各デバイスとの通信プロトコルは，プライベートな LAN 上の端末からアクセスされることが前提となっているため，ホームサーバは家庭内の LAN 上に配置される．

②を実現するために，既存研究にて実現している「タグレイヤ重畳方式[12]」を適用する．これは，任意の Web ページにユーザが関心を持つ情報を重畳して表示する技術である．このタグレイヤ重畳方式を用いて，デバイス状態の登録と照合の機能を実装する．タグレイヤ重畳技術について，ここでは概要のみ説明する．

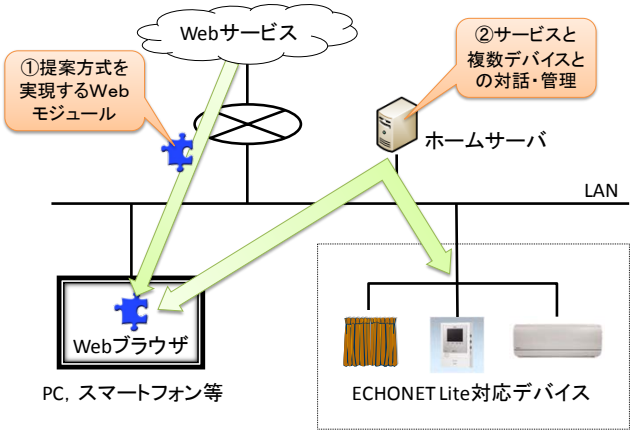

図 1 提案方式のアーキテクチャ
Figure 1 A proposal architecture.

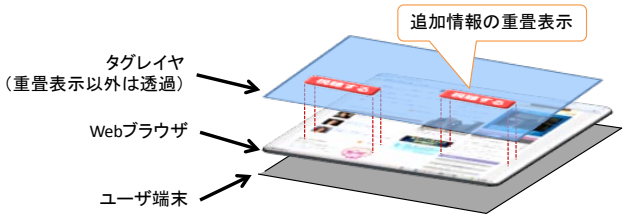

図 2 タグレイヤ重畳方式
Figure 2 A concept image of “Taglayer” overly method.

タグレイヤ重畳技術の概念図を図 2 に示す．タブレットやスマートフォンなどの表示端末上で Web ブラウザを利用し，ユーザが閲覧中の Web ページに重畳して任意の情報を表示することができる．例えば，テレビ番組に関する話題の Web ページにはそのテレビ番組を録画予約する表示ウインドウを重畳して表示する．また，洗濯機の脱水運転が終了したら，引き続き乾燥運転を開始する操作ボタンと合わせて，天気が良ければ乾燥運転の電気代を表示して天日干しを促す省エネの工夫も取り入れ進められている[13]．

このタグレイヤ重畳方式は Web モジュール(JavaScript)で実装されており，元の Web ページにこのモジュールを追加することにより適用することができる．追加の方法としては，Web ブラウザの機能拡張とする方法もあるがスマートフォンやタブレットに適用するのは困難であるため，HTTP Proxy の機能を拡張することで実現する．

提案方式を利用するためには，閲覧中の Web ページにデバイス状態を登録する「登録機能」と，登録されている Web ページを表示する際に照合する「照合機能」の 2 つの機能が必要である．これら 2 つの機能を JavaScript で実装し，タグレイヤ重畳方式を使って閲覧中の Web ページに追加する．そして，Web ページの更新毎に登録されているか否かを判定し，登録されていれば照合を，そうでなければ登録のための画面を，タグレイヤを使って表示する．デバイス状態の登録と照合の流れについて，図 3 を用いてそれぞれ説明する．

ⓒ 2014 Information Processing Society of Japan

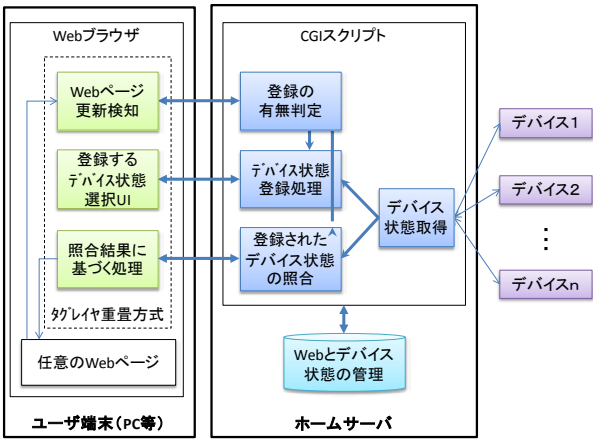

図 3　提案方式の機能構成

Figure 3　A functional structure for proposal method.

【照合の実行】

Web ブラウザ上で Web ページの更新が発生すると，その URL がホームサーバに登録されていればデバイス状態の照合の画面を，そうでなければ登録の画面を，Web ページ上に重畳して表示する．

【デバイス状態の登録】

各デバイスの現在の設定状態を取得し，登録画面を作成して CGI 応答により返却する．その中からユーザが選択したデバイスおよび設定状態の 1 つ以上の組み合わせと，その時の Web の URL をホームサーバに送信し，ホームサーバに登録する．

【デバイス状態の照合】

受け付けた Web の URL をキーに登録されているデバイス状態を取得し，それぞれのデバイスに対し状態を確認する．その結果が登録されている状態と同じであれば OK を，そうでなければ NG とし，全てが OK であるかを判定した結果を含めて CGI 応答として返却する．この結果を受信した追加モジュールは判定結果を元に，サービス利用の許可等の制御を行う．

4.4 性能要件

設計指針③で示したように，提案方式の実行処理時間を 3 秒以下に抑える．具体的には，前節のアーキテクチャで説明した Web ブラウザからの要求を受けたホームサーバが，各デバイスの状態を取得し，Web ブラウザに返却するまでの時間を 3 秒以内に抑えることとする．これを実現するためには，各デバイスに同時に問い合わせして並列処理で状態を受け付ける方法が考えられるが，例えば ECHONET Lite のように受け付けポートが固定されている場合は同時に受け付けられるデバイス数に制限がある．元々ECHONET Lite では短時間での多量アクセスを想定したプロトコル設計になっていないため，並列処理には不向きである．他のプロトコルも同様の問題があると考えられるため，順番にデバイスに問い合わせして状態を確認する方式を選択した．デバイス側での応答処理は各メーカ・機種に依存するため，実際に 3 秒以内にいくつのデバイスへの問い合わせが可能であるか明らかではない．そこで，実際の市販製品を用いて問い合わせまでに要する時間を計測し，提案方式で登録できるデバイス数の目安を調査する．なお，ECHONET Lite では 1 回の問い合わせで複数の状態を同時に取得することができる．

次章では，本節で述べたアーキテクチャに沿って構築したプロトタイプを用いて実現性の検証を行うとともに，性能要件を満たすデバイスおよび状態の数について調査・考察する．

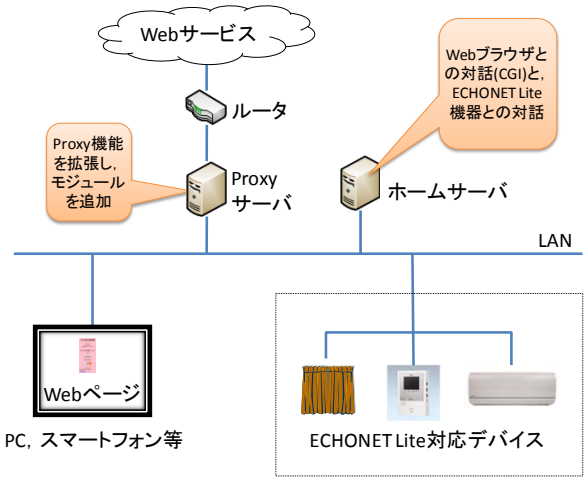

図 4　プロトタイプシステムの構成

Figure 4　Structure of the prototype system.

5. 評価

提案方式を実現するには，任意の Web サービスにユーザがデバイスの状態を関連付けて登録する画面が必要であるが，まずは提案手法の実現性検証を優先し，そのために必要となる適用できるデバイスの状態数について調査することを先行して進めるとした．

そこで本章では，現実的なハードウェア構成において，実現要件を満たすことが可能であることを確認し，適用できるデバイス数・状態について考察する．以降，検証に用いたプロトタイプについて説明し，このプロトタイプを用いた検証実験について述べる．

5.1 プロトタイプ

図 4 は，構築したプロトタイプシステムの構成図である．ホームサーバには，小型なシングルボードコンピュータである RaspberryPi TypeB[14]を利用した．RaspberryPi の電力は 700 mA (3.5 W)であり比較的省電力であるため，家庭用に配置されるサーバとして適切であると考えた．ホームサーバおよびユーザ端末となる PC の仕様は表 1 の通りである．また，表 2 にあげた実機の ECHONET Lite デバイスを LAN に接続し，状態の登録と照合の確認を行った（図 5）．

ⓒ 2014 Information Processing Society of Japan

表 1 プロトタイプシステムの仕様

Table 1 Specific of a prototype system.

ホームサーバ		
	ハードウェア	RaspberryPi TypeB
	OS	Linux rasberrypi 3.10.25+ #622 Raspbian “wheezy” 2012-09-18
	HTTP サーバ	Apache/2.2.22
	Web-CGI	Ruby1.9.3p194
ユーザ端末		
	ハードウェア	Panasonic 社製 Let’s Note CF-W8
	OS	Windows 7 Professional
	Web ブラウザ	Google Chrome 35.0.1916.27 beta-m

表 2 プロトタイプで使用する ECHONET Lite デバイス

Table 2 ECHONET Lite devices used in the prototype system.

デバイスタイプ	型番
冷蔵庫	GR-G51FXV
洗濯乾燥機	TW-Z96X1
エアコン	RAS-632NDR
コントローラ	BTR-4010AZ
エコキュート	HWH-FB372C
分電盤メータリング	HEM-EM31A
畜電池	ENG-B6630A1
シーリングライト	LEDH-LT1

図 5 プロトタイプで利用したデバイスとホームサーバ

Figure 5 Devices used in the prototype system.

図 6 プロトタイプの動作画面（照合中）

Figure 6 A screen of the prototype during inquiring approval.

図 7 プロトタイプの動作画面（結果表示の拡大）

Figure 7 A screen of the prototype after receiving a result from a home server.

{"ref":"http://*******.html",
"108100000ef0010130006202b000b300":"OK",
"108100000ef001026b006202d3009000":"OK",
"108100000ef0010290006202800008f00":"OK",
"result":"1"}

図 8 ホームサーバから受信した結果データ

Figure 8 A received result data from the home server.

5.2 動作検証

図 6 はタグレイヤ重畳方式により追加されたモジュールにより，一般のWebページに提案方式のダイアログが重畳され，ダイアログ中の照合実行ボタンの押下後，ホームサーバに問い合わせている画面である．このとき，ホームサーバでは今回用いたECHONET Liteデバイスに状態を問い合わせ [b]，登録された状態と同一であるかを照合判定している．図 7 はホームサーバで判定した結果を表示した画面である．またこの表示の元になる，ホームサーバから受信した状態確認の結果を図 8 に示す．

以下，この結果の内容について説明する．まずフォーマットは汎用的に利用されている JSON 形式とした．"ref"は現在表示中の Web ページの URL である．それに続く，3 つの"108100･･･"の項目は 3 つのデバイスに問い合わせた内容で，ECHONET Lite の電文をそのまま表している（青色部の"02"が同時に問い合わせる状態数，黄色・緑色部が状態数分の結果．詳しくは ECHONET Lite 仕様書[15]を参照）．この実行結果では，各デバイスに対し 2 つの状態を同時に問合せしている．そしてそれぞれの項目ごとに登録していた状態と一致するか否かの照合結果が"OK"/"NG"として表されている．これらが全て"OK"であれば"result"は"1"，そうでなければ"0"なる．この実行結果では，全て"OK"で"result"は"1"となっており，その結果を JavaScript により判断して，図 7 のように照合された旨のメッセージを表示している． ここでは確認用にこれらの表示をしているが，実際のサービスでは JavaScript で内部的に処理され，動画の視聴など元のサービスを実行する．また，1 つでも"NG"が

b) 具体的には，エアコン（運転モード設定，温度設定値），エコキュート（温度設定値，ON タイマ予約設定），シーリングライト（動作状態，節電動作設定）の計 3 デバイス 6 状態を問い合わせしている．

ⓒ 2014 Information Processing Society of Japan

あった場合は"result"が"0"となり,「照合されませんでした」というメッセージを表示し，画面の遷移を制限する．プロトタイプの実行例では動画再生のボタン（動画領域中央）の上に照合のダイアログを重畳表示しており，照合 OK であればダイアログ消去，NG であればそのまま表示することにより再生の制御を行った. 以上の一連の動作を確認し，提案方式による実現性を確認した.

5.3 性能評価

本節では，性能要件である応答時間 3 秒以内を満たすデバイスおよび状態について，実際の家電を用いて現実的に適用可能な数を考察する．前項のプロトタイプで実際の ECHONET Lite デバイスを扱った場合の処理時間を測定し，利用可能な条件について考察する.

5.3.1 計測方法

設計指針で述べた 3 秒はユーザが待機する時間のことであり, 本来は Web ブラウザでの操作開始から表示されるまでの時間を計測すべきである．しかし，Web ブラウザとホームサーバ間のネットワーク遅延や Web ブラウザ内での処理時間等の影響を排除するため，ここではホームサーバと家電との間でやりとりされる時間を計測することとした. 具体的には，ホームサーバで受け付ける CGI モジュールをホームサーバ上で直接実行し，その時間を linux シェルコマンド”time”により計測し，5 回の平均値をとる.

また，ECHONET Lite デバイスから取得するデバイス数と状態数は，それぞれのデバイスから取得可能な状態を一度の問い合わせで同時にすべて取得する.

5.3.2 結果

今回用意した ECHONET Lite デバイスのそれぞれの応答時間を表 3 と図 9 に示す. 冷蔵庫, エアコン, 洗濯乾燥機, エコキュートについては，平均応答時間が設計指針で定めた 3 秒を超えている（表 4）. また，問い合わせの状態数と応答時間には相関性はなく，デバイス毎に異なっていることが分かる（図 9）.

表 3　ECHONET Lite デバイスの応答時間
Table 3　Response time of ECHONET Lite devices.

デバイスタイプ	状態数 (Max)	平均応答時間(sec)	1 状態あたりの応答時間 (sec/状態)
コントローラ	2	1.34	0.67
冷蔵庫	13	3.51	0.27
エアコン	19	3.27	0.17
シーリングライト	19	2.91	0.15
洗濯乾燥機	21	5.13	0.24
分電盤	24	2.12	0.09
蓄電池	24	2.84	0.12
エコキュート	30	3.13	0.10

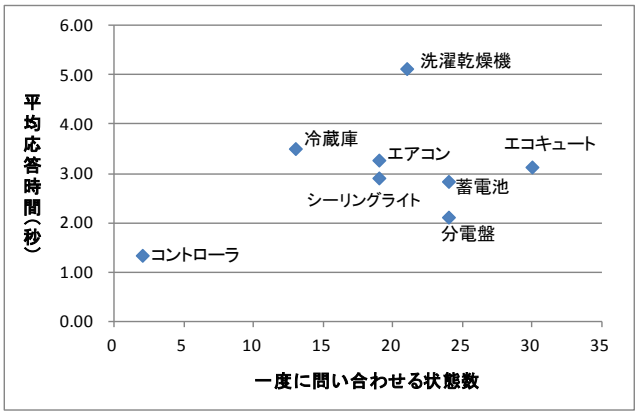

図 9　問い合わせ状態数と平均応答時間
Figure 9　Relations between requests for status and average of response time.

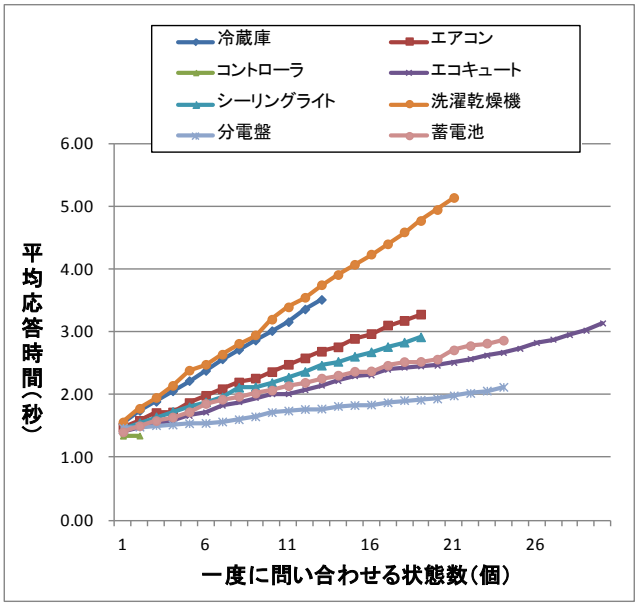

図 10　問い合わせる状態数と応答時間
Figure 10　Average of response time per a request for status.

5.4 考察

前項の結果の通り，一度に多くの状態数を問い合わせる場合，設計指針とした応答時間 3 秒を超えてしまうデバイスがあった．そのため，3 秒を超えないようにデバイスの状態数を選択できる可能性を調査する.

状態数と応答時間の関係を探るため，1 回に問い合わせる状態数を 1 つから順に 1 つずつ増やし，その時の応答時間を計測する追加調査を行った．図 10 はその結果である. デバイス毎に見ればほぼ比例の関係が見られる．一方で，デバイスの種類によって比例の傾きが異なっている．そこで最小二乗法によりデバイス毎の傾きと状態数 1 つの場合の応答時間を算出した（表 4 の a）．この値を元にすれば，デバイスの状態数を選択することで，概ねの応答時間を予測することが可能と考えた. 表 4 の b は, 冷蔵庫から 1 つ，コントローラから 2 つの状態を選択したときの例であるが,

単純に応答時間の総和を取ると 2.94 秒となり，この組み合わせであれば指針である 3 秒以内に収めることができる．

以上のことから，機種毎に応答速度は異なるが，3 つの状態数の組み合わせで 3 秒以内に収められることが分かった. さらに, 機種毎に応答速度を計測しておくことにより，ユーザが選択した照合に利用する状態の組み合わせに要する応答時間を予測し，状態選択の目安とすることが可能である．

なお各デバイスの応答時間の和について，時間の短縮と正確な予測を行うためには，表 4 の 1 状態取得の応答時間の処理内容を解析する必要がある．この時間には，今回使用した Ruby プログラムの処理と，各デバイスの処理時間が含まれている．前者はプログラムの最適化により時間短縮が見込める．一方後者のデバイス側はメーカや機種に依存するため時間短縮は難しい．

6. まとめと今後の課題

任意の Web サービスに複数のデバイスの状態をユーザが関連づけ，そのサービスとデバイスの状態の同一性を担保して，サービスを利用する仕組みを提案した．実現にあたり，その時ユーザが利用している Web サービスの継続性を維持するため，Web ブラウザにおいてデバイス状態を確認できること，画面遷移しないこと，応答時間を 3 秒以内に抑えることを設計指針とした．それに従って，アーキテクチャを示し，プロトタイプにより実現性を確認した．さらに，応答時間を満たすために選択可能なデバイスおよびその状態数について実機を用いて計測および評価を行い，応答時間 3 秒以内という性能要件において，状態数が延べ 3 つまで取れる組み合わせが存在することを確認した．

今後は，応答時間の大半を占めているプログラムの最適化を図り，今回省略した登録のためのユーザ対話画面も含めたユーザビリティ評価を進める．

表 4　状態数と応答速度の算出
Table 4　A response time of ECHONET Lite devices.

	a)最小二乗法適用		b)応答速度算出(例)	
	傾き	1状態取得の応答時間	状態数	応答時間(秒)
冷蔵庫	0.153	1.600	1	1.60
エアコン	0.100	1.472	0	0.00
コントローラ	0.002	1.339	2	1.34
エコキュート	0.055	1.459	0	0.00
シーリングライト	0.080	1.477	0	0.00
洗濯乾燥機	0.176	1.597	0	0.00
分電盤	0.028	1.437	0	0.00
各デバイスの応答時間の和				2.94

参考文献

[1] エコーネットコンソーシアム:ECHONET CONSORTIUM, http://www.echonet.gr.jp/index.htm
[2] Consumer Home, http://jp.dlna.org/
[3] S. Duquennoy, G. Grimaud, and J.-J. Vandewalle: The Web of Things: Interconnecting Devices with High Usability and Performance, ICESS '09. International Conference on, 2009, pp323-330 (2009)
[4] Atzori, Luigi, Antonio Iera, and Giacomo Morabito: The internet of things: A survey, Computer Networks, pp.2787-2805 (2010)
[5] 西垣弘二，安本慶一，柴田直樹，伊藤実：コンテキストに基づいた情報家電の連携を実現するためのフレームワークおよびルールベース言語の提案 (フレームワーク). 情報処理学会研究報告 UBI, Vol.112, pp21-27 (2004)
[6] 前田潤，福田健一，木下和彦，村上孝三: 情報の連想的結合によるコンテクストアウェアサービス制御方式，電子情報通信学会論文誌 B 通信(1), pp22-34 (2008)
[7] HEMS(ECHONETLite)認証センター: HEMS(ECHONETLite)認証センター, http://sh-center.org/
[8] W3C HTML Working Group，http://www.w3.org/html/wg/
[9] 芦村和幸，小松健作，一色正男：Web と機器を透過的につなぐ Multimodal Interaction フレームワークの実装，コンシューマ・デバイス & システム Vo. 2, No. 2, 情報処理学会論文誌論文誌トランザクション, pp19-28 (2012)
[10] 松村剛志，安川健太ほか：Web とホームデバイスをつなぐプラットフォームによる新しいユーザ体験，電子情報通信学会技術研究報告情報ネットワーク 110, pp181-186 (2011)
[11] CROSS-STYLEーウェブデザイン講座「８秒ルールは今は昔！？進化した「３秒絶対ルール」とは？」, http://www.cross-style.com/webdesign/006.html
[12] 渡部智樹，青木良輔，小林透，小林稔，一色正男：Web 閲覧と連動したアンビエントな家電操作方式の提案，情報処理学会論文誌コンシューマ・デバイス＆システム(CDS),Vol.2, No.2,pp.73-80 (2012)
[13] Tomoki Watanabe, Rika Mochizuki, Toru Kobayashi, Masao Isshiki: HACCS: Home Appliance Control Concierge System: Extending Functions on Web Service, COMPSAC 2013, pp 208-213 (2013)
[14] Model B | Raspberry Pi, http://www.raspberrypi.org/product/model-b/
[15] APPENDIX ECHONET 機器オブジェクト詳細規定 Release E：エコーネットコンソーシアム (2014)

ⓒ 2014 Information Processing Society of Japan