社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

ホームネットワークシステムにおける デバイス状態ログマイニングのためのサービス指向プラットフォーム

瀬戸　英晴†　　柗本　真佑†　　中村　匡秀†

† 神戸大学　〒 657–8531 兵庫県神戸市灘区六甲台町 1–1
E-mail: †seto@ws.cs.kobe-u.ac.jp, ††{shinsuke,masa-n}@cs.kobe-u.ac.jp

あらまし　我々の研究グループでは実際の家電やセンサを用いたホームネットワークシステム（HNS）を開発している．HNS においては，各デバイスの稼動状態を記録しており，これらの「デバイス状態ログ」を活用することで，様々な付加価値サービスの実現が期待できる．本稿では，蓄積されたデバイス状態ログを検索し， HNS において発生した様々な状況を効率よく発見するためのプラットフォームの開発を目指す．また，ケーススタディとして，デバイス状態ログを活用した「エネルギー浪費行動自動検出サービス」を開発し，提案プラットフォームの有用性を示す．
キーワード　ホームネットワークシステム，デバイス状態ログ，データマイニング，プラットフォーム

Service Oriented Platform for Mining Device Status Log in HNS

Hideharu SETO†, Shinsuke MATSUMOTO†, and Masahide NAKAMURA†

† Kobe University　Rokkoudai-cho 1–1, Nada-ku, Kobe, Hyogo, 657–8531 Japan
E-mail: †seto@ws.cs.kobe-u.ac.jp, ††{shinsuke,masa-n}@cs.kobe-u.ac.jp

Abstract　We have been studying and developing the home network system (HNS), which provides value-added services by orchestrating home appliances, equipments and sensors via network. Our HNS periodically records status of every device within the system. The records, called *device status log*, can be used for various value-added services. In this paper, we implement the system that automatically search various status of home users, using the device status log. Also we demonstrate its feasibility through a case study of detecting energy-wasting behaviors.
Key words　home netwrok system, device status log, energy saving, data mining, platform

1. はじめに

近年，身の回りの様々な機器のデジタル化，ネットワーク化が進んでおり，それに伴ってデジタル機器の操作履歴や環境情報といった**ログ**が容易に取得できるようになりつつある．これらのログは，機器を利用した人やその場の環境を特徴付けるものであり，様々な活用例があると考えられる．

我々の研究グループでは，家電やセンサを宅内のネットワークに接続し，様々な付加価値サービスを実現する**ホームネットワークシステム**（以下 HNS）の研究・開発を行っている [1]. HNS では，各機器がネットワークに接続されているため，機器の稼働状態や，機器が計測した環境値を容易に取得できる．我々はこのような機器の状態や環境値などのログを，**デバイス状態ログ**と呼び，様々なサービスへの活用を検討している．

これらデバイス状態ログを長期間蓄積することで，様々な付加価値サービスを実現することが可能である．例えば，「テレビを長時間つけていた」「外が明るいのに照明を利用していた」といった機器使用状況の振り返りによるライフスタイルの把握や見直しのほか，消費電力量を振り返ることで省エネを促すといったことが考えられる．また，個人の嗜好や生活パターンをログから推定し，ユーザそれぞれの好みに応じた個人適応型サービスの実現といった利用方法も考えられる．

しかしながら，家庭には多種多様な家電やセンサがあり，一定時間間隔で蓄積されるデバイス状態ログはその容量が肥大化する傾向にある．そのような膨大なデバイス状態ログから必要なデータを抽出することは容易ではない．また，現状ではデバイス状態ログを活用するサービスそれぞれが，独立にログから必要な情報をマイニングしているため，サービス間でのマイニング結果の再利用はほとんど行われていない．

そこで本研究では，デバイス状態ログから必要なデータを効率よくマイニングするためのプラットフォームを提案する．提案プラットフォームはデバイス状態ログのマイニングを 2 段階に分けて実施する．まず，再利用性が高いデータをマイニングしておき（**第一段階マイニング**），次に，サービス開発者は用

意された API を利用することで，第一段階マイニングによって整形されたデータから必要なものを取得する（**第二段階マイニング**）．ケーススタディとして，ホームネットワークにおけるエネルギー浪費行動自動検出サービスを開発し，提案プラットフォームの有用性を調べた．

2. 準　　備

2.1 ホームネットワークシステム

ホームネットワークシステムは，照明やテレビなど家庭内における様々な家電と，温度計や照度計などのセンサをネットワークに接続することで構築される．HNS 内における機器はユーザや外部エージェントがネットワーク越しに制御できるように，制御 API を備えている．我々の研究室で開発している CS27-HNS [1] は，サービス指向アーキテクチャ(SOA) の考えを取り入れ，すべての制御 API を，機種や実行環境に依存しない標準的な Web サービスとして公開している．

2.2 デバイス状態ログ

CS27-HNS では，各家電の状態を SOAP または REST 通信によって，`getStatus()` メソッドを利用して取得可能である．例えば，`http://cs27-hns/TVService/getStatus` にアクセスすると，テレビの現在状態が XML 形式で返される．状態は [power:ON, channel:6, volume:15] のように属性名とその値の組を並べた構造体で表現される．同様に，センサの計測した環境値は `getValue()` メソッドを利用することで取得できる．本論分では，家電の稼動状態，センサが計測した環境値など，ユーザの生活を性質付けるような情報を**デバイス状態**と呼び，さらに，デバイス状態をログとして蓄積したものを**デバイス状態ログ**と呼ぶ．

2.3 デバイス状態ログを活用したサービス

デバイス状態ログを活用したサービスの一例として，消費電力の見える化がある．これは消費電力ログをグラフ等で可視化しユーザに提示するものである．ユーザはエネルギー消費の実態を把握できるため，浪費行動の自発的な改善につながるとされる [2]．我々のグループでも HNS における電力消費振り返りサービス [3] を提案している．このサービスは，消費電力ログ，機器操作ログ，環境情報ログの 3 種類のログを重ね合わせることで，単に消費電力の推移を振り返るだけでなく，その電力消費に至った環境状態やその時の機器操作を客観的に観測できる．

ほかにも，デバイス状態ログを活用したサービスとして，機器の操作ログと外部環境情報ログを利用して，自動的に無駄な電力消費行動を特定し，ユーザに提示するサービス [4] や，過去の行動パターンから個人の好みや機器操作パターンを推定し，各ユーザの好みに応じた機器操作を推薦するといった個人適応型サービス [5] などが考えられる．

2.4 課　　題

デバイス状態ログを活用したサービスの実現に対する課題として，次の 2 点が挙げられる．1 点目はログの肥大化，2 点目はマイニング結果の再利用性の低さである．HNS 環境下では，テレビやエアコンといった家電から，温度センサ，照度センサなど，多種多様なデバイスがネットワークに接続されており，

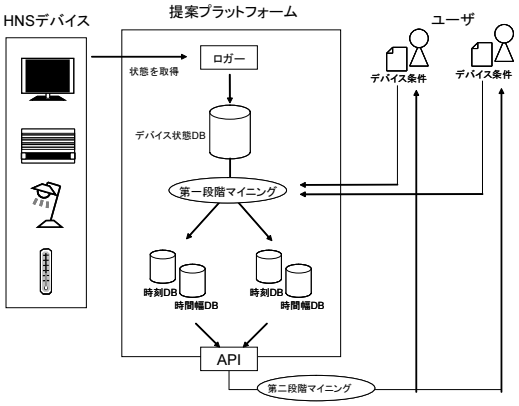

図 1　提案プラットフォームのアーキテクチャ

デバイス状態ログは膨大な容量となることが予想される．一度のマイニングで必要なデータを取得しようとすると，多くのリソースが必要となり，サービスの応答時間も長くなるため現実的ではない．また，従来のサービスにおいては，それぞれのサービスが個別に必要なデータをデバイス状態ログからマイニングしているために，サービス間でのマイニング結果の再利用はほとんど行われていない．

3. 提 案 手 法

3.1 キーアイデア

前節で述べた課題を解決するためのアイデアとして，デバイス状態ログのマイニングを 2 つの段階に分けることを考える．まず第一段階マイニングでは，単純な比較演算のみで構成されるデバイス状態の定義に基づき，膨大なログの中から必要なログのみを抽出する．このデバイス状態の定義は各サービス開発者が作成するもので，XML で記述可能な言語によって定義される．第一段階マイニングにより抽出されたログは，生のログが蓄積されている DB とは異なる DB に蓄積されるため，サービス間での共有が可能となる．さらに第二段階マイニングでは，あらかじめ用意された API を呼び出すことで，第一段階マイニングで抽出したログから必要なログを抽出する．これによりサービス開発者は，SQL よりも直感的な API 操作でログを抽出することが可能となる．

3.2 目的とアプローチ

本研究の目的は，デバイス状態ログの蓄積方法を提案し，またデバイス状態ログから条件に合致する日時や期間を効率よく発見するためのプラットフォームの実現である．

図 1 に提案プラットフォームのアーキテクチャを示す．提案プラットフォームは大きく分けて，デバイス状態ログ蓄積部分，第一段階マイニング部分，第二段階マイニング部分に分けられる．提案プラットフォームの大まかな流れとしては，まず，ロガーが HNS 内のデバイスからデバイス状態を取得し，デバイス状態 DB に蓄積する．次に，ユーザが定義したデバイス条件に従い第一段階マイニングを行い，ユーザ固有の DB にマイニング結果を蓄積する．最後に，ユーザは指定の API を利用することで，ユーザ固有の DB から必要なデータを取得する．

```
<ns:getStatusResponse xmlns:ns="http://cs27-hns">
  <ns:return xmlns="http://status.cs27-hns/xsd"
     type="cs27.hns.appliance.status.TVStatus">
    <power>true</power>
    <channel>6</channel>
    <volume>8</volume>
    <input>1</input>
  </ns:return>
</ns:getStatusResponse>
```

図 2　テレビの getStatus レスポンス例

```
<ns:getValueResponse xmlns:ns="http://cs27-hns">
    <ns:return>25.778019999999998</ns:return>
</ns:getValueResponse>
```

図 3　室内温度センサの getValue レスポンス例

以降では，**A1: デバイス状態ログの蓄積方法**，及び第一段階マイニングで利用する **A2: デバイス条件を定義する言語，DCDL** について述べ，さらに 2 段階のマイニング **A3: 第一段階マイニング**，**A4: 第二段階マイニング**について説明する．

3.3　A1: デバイス状態ログの蓄積方法

2.2 で述べたように，デバイス状態は HNS に設置されている各デバイスの状態取得 API を用いて取得できる．我々の CS27-HNS では，家電の場合は `getStatus` インタフェース，センサの場合は `getValue` インタフェースを Web サービスで呼び出すことで取得できる．図 2 に CS27-HNS のあるテレビの状態を REST-Web サービスで呼び出した結果を示す．テレビの電源がオン (true) で，チャンネルが 6，音量が 8，入力が 1 であることがわかる．また，図 3 に温度センサが計測した環境値の取得例を示す．室内の温度が 25.8 ℃であることがわかる．

ログ取得プログラム (**ロガー**) は，`getStatus` や `getValue` インタフェースを用いて HNS 内のデバイスに定期的にアクセスし，デバイス状態を取得する．取得されたデバイス状態はデバイス状態ログとして後述のデータベースに蓄積される．デバイス状態の取得間隔については，間隔が短いほど正確な情報を得られるが，蓄積する情報量が膨大となる．現在のところ我々の CS27-HNS では 1 分間隔に設定している．

次に，各デバイスから取得したデバイス状態を蓄積する**デバイス状態 DB** の実現方式を考える．デバイス状態をログとして記録する必須データ項目として，個々のログを区別するログ ID，ログの取得日，取得時刻が挙げられる．さらに，どのデバイスの状態かを記録する必要があり，HNS 内で定義される機器識別子を用いることにした．取得したデバイスの状態は，各デバイスごとにその解釈が異なるために，ここでは生データをそのまま保存する．また，個別のデバイスについての詳細，デバイス状態の解釈についてはそれぞれ別テーブルで定義した．

以上の議論を踏まえて，デバイス状態 DB のデータモデリングを行った．図 4 にその ER 図を示す．四角はデータエンティティ(テーブル) を表しており，エンティティ名，CRUD，キー，属性のスキーマが並ぶ．エンティティの下部にはインスタンスを併記している．エンティティ間の関連として，(＋ー…) は参照関係を表す．

図 4 において，例えば，DATA_TABLE の dataID が D001 であるレコードは，2011 年 9 月 1 日，15 時 20 分における tv01 の稼働状態を記録している．また，DEVICE_TABLE より，tv01 は機器クラス [6] が TV であり，場所はリビングにあることがわかる．さらに，tv01 の schemaID は S001 で，これは SCHEMA_TABLE を見ることで，生データのデータ型が XML であり，またスキーマの詳細については参照 URL http://cs27-hns/TV/Status.xsd を確認すればよいことがわかる．

3.4　A2: デバイス条件を定義する DCDL

3.3 で示したように，デバイス状態 DB は，ある時刻における機器の状態が生データとして逐次的に保存さている．生データを利用することで，時刻を指定したときにその時刻における機器の状態を知ることはできるが，逆に，状態を指定したときにその状態であった時刻や期間を知ることは難しい．サービス開発者にとっては，機器が特定の状態にあった時刻や期間を利用できたほうが，ユーザの嗜好や状況をより的確に指摘でき，サービスの開発が行いやすいと思われる．

そこで，本節では生データから特定の状態を抽出するための条件 (**デバイス条件**と呼ぶ) について述べる．ここで，デバイス条件は次のような条件式で定義する．

`[デバイス ID. 属性 比較演算子 属性値]`

デバイス ID は，3.3 で述べた機器識別子，属性は 2.2 で述べたその機器の状態を構成するプロパティ名，比較演算子は，等しい (`==`)，異なる (`!=`)，以上 (`>=`)，以下 (`<=`)，より大きい (`>`)，より小さい (`<`) のいずれかであり，属性値は属性が取りうる定数値である．例えば「テレビがついている」というデバイス条件は `[tv.power == true]` となり，また,「外の照度が 500lux 以上」というデバイス条件は `[outLightSensor.brightness >= 500]` と表現される．

デバイス条件を記述するにあたっては，XML 形式の記述言語，**DCDL (Device Condition Description Language)** を提案する．DCDL は，`<condition>`タグ内に 1 つのデバイス条件を記述する．`<condition>`内には，デバイス条件 ID`<conditionID>`，デバイス条件の説明`<description>`，デバイス ID`<deviceID>`，メソッド名`<method>`，属性`<property>`，比較演算子`<operator>`，属性値`<value>`が含まれている．なお，XML 中には不等号のような一部の記号を利用することができないため，比較演算子については表 1 参照に示す代替文字列を利用する．図 5 に「テレビがついている」というデバイス条件を DCDL で記述したものを示す．

表 1　比較演算子の代替文字列

比較演算子	代替文字列
==	EQUAL
!=	NOT_EQUAL
>=	MORE_THAN_EQUAL
<=	LESS_THAN_EQUAL
>	MORE_THAN
<	LESS_THAN

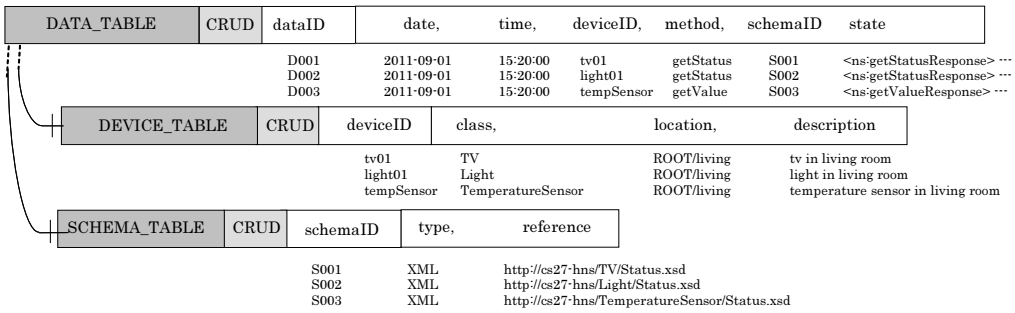

図 4　デバイス状態ログのデータモデル

```
<condition>
  <conditionID>1</conditionID>
  <description>テレビがついている</description>
  <deviceID>tv01</deviceID>
  <property>power</property>
  <operator>EQUAL</operator>
  <value>true</value>
</condition>
```

図 5　DCDL によるデバイス条件の記入例

3.5　A3: 第一段階マイニング

第一段階マイニングでは，DCDL により定義されたデバイス条件に基づき，生データからユーザが利用しやすいような別形式へのデータ変換を行い，後述のデータベースに蓄積する．

まず，第一段階マイニングによって変換されるデータ形式について述べる．r を任意のデバイス状態とし，t_r を r の時刻，f をデバイス条件とする．r が f を満たすとき，t_r を r の f に関する**デバイス条件成立時刻**と呼ぶ $r_i, r_{i+1}, ..., r_k$ が時間的に連続したデバイス状態ログで，$r_j (i \leqq j \leqq k)$ がすべてのデバイス条件 f を満たすとき，時刻の幅 $[t_{r_i}, t_{r_k}]$ を f に関する**デバイス条件成立条件幅**と呼ぶ．

次に，デバイス条件成立時刻，デバイス条件成立時間幅を蓄積するためのデータベースについて考える．なお，ふたつのデータは時刻と時間幅を示しているという点は異なっているが，それぞれが保持する情報はほぼ同一であり，データモデルもほぼ一致している．ここでは，デバイス条件成立時間幅を蓄積するための**デバイス条件成立時間幅 DB** のデータモデルを考える．デバイス条件成立時間幅を記録するためのデータ項目としては，個々の時間幅を区別する時間幅 ID，デバイス条件 ID，デバイス条件が成立した日付，さらにデバイス条件が成立した時間幅を示す成立開始時刻と成立終了時刻が必要である．

以上の議論に基づき，デバイス条件成立時間幅 DB のデータモデルを行った．図 6 にその ER 図を示す．具体例として，TIME_INTERVAL_TABLE の intervalID が I001 であるレコードを取り上げると，これは，2011 年の 9 月 1 日に，9 時 12 分から 15 時 33 分までデバイス条件 C001 が成立していたことを表している．また，C001 は CONDITION_TABLE を参照することで，tv01 の getStatus メソッドの戻り値において，電源が ON(true) であるという条件を表していることがわかる．

なお，詳細は省くが，デバイス条件成立時刻を蓄積するための**デバイス条件成立時刻 DB** は，デバイス条件成立時間幅 DB のデータモデルのうち，時間幅の終端を示すデバイス条件成立終了時刻を除いたものである．

提案プラットフォームを利用するにあたって，あらかじめユーザは，興味のあるデバイス条件を DCDL によって記述し登録しておく．次に，提案プラットフォームは登録されたデバイス条件をもとに，デバイス条件成立時刻，デバイス条件成立時間幅へのデータ変換を行い，ユーザ毎に固有のデータベースに蓄積していく（図 1 参照）．現在，CS27-HNS においては，1 時間に一度のタイミングで第一段階マイニングを行っている．

3.6　A4: 第二段階マイニング

第一段階マイニングによって蓄積されたデバイス条件成立時刻 DB，デバイス条件成立時間幅 DB から，ユーザは必要に応じてデータを取得する．ユーザが上記ふたつのデータベースから，必要な時刻データ，または時間幅データを抽出する過程を**第二段階マイニング**と呼び，提案プラットフォームは第二段階マイニングのための 4 つの基本 API を提供する

(API1): checkCondition(date, time, conditionID)
(API2): getConditionDetails(conditionID)
(API3): getConditionsHoldAt(date, time)
(API4): getTimeInterval(date, conditionID)

checkCondition インタフェースは，日付，時刻，デバイス条件 ID をクエリにとり，指定されたデバイス条件 ID に対応するデバイス条件が，その時間に成立していたかの真偽を返す．

■ API： checkCondition()
■引数： date(年日付), time(時刻),
　　conditionID(デバイス条件 ID)
■戻り値： true/false

getDeviceConditionDetails インタフェースは，デバイス条件 ID をクエリにとり，そのデバイス条件 ID の詳細を返す

■ API： getDeviceConditionDetails()
■引数： conditionID(デバイス条件 ID)
■戻り値：デバイス条件 ID についての詳細をあらわす
　　DeviceConditionObject

DeviceConditionObject の詳細を表 2 に示す．DeviceConditionObject はフィールドとして，デバイス条件 ID，デバイ

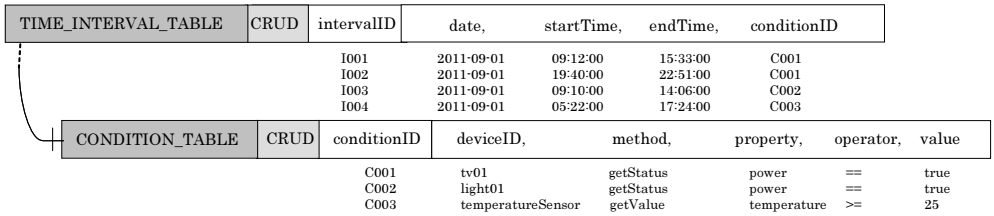

図 6　デバイス条件成立時間幅 DB のデータモデル

ス ID，属性，演算子，属性値を保持している．

表 2　DeviceConditionObject のフィールド一覧

フィールド	詳細	例
conditionID	デバイス条件 ID	C001
deviceID	デバイス ID	tv01
property	属性	power
operator	比較演算子	EQUAL
value	属性値	true

`getConditionsHoldAt` インタフェースは，日付と時刻をクエリにとり，指定された日付と時刻において，成立していたデバイス条件 ID の一覧を返す．

■ API： `getConditionsHoldAt()`
■引数： `date:`(年月日)`, time`(時刻)
■戻り値： デバイス条件 `ID` の配列

`getTimeInterval` インタフェースは日付，デバイス条件 ID をクエリにとり，クエリに一致したデバイス条件成立時間幅オブジェクトの一覧を返す．

■ `API: getTimeInterval()`
■引数： `date`(デバイス状態を取得したい年月日)，
`conditionID`(デバイス条件 `ID`)
■戻り値： デバイス条件成立時間幅オブジェクト
(`TimeIntervalObject`) の配列

TimeIntervalObject の詳細を表 3 に示す．TimeIntervalObject はフィールドとして，年月日 (date)，条件成立開始時刻 (startTime)，条件成立終了時刻 (endTime)，デバイス条件 ID(conditionID) を保持している．

表 3　TimeIntervalObject のフィールド一覧

フィールド	詳細	例
date	年日付	2011-09-01
startTime	条件成立開始時刻	9:12
endTime	条件成立終了時刻	15:33
conditionID	デバイス条件 ID	C001

3.7 実　　装

以上の議論に基づいて，提案プラットフォームの実装を行った．実装に利用した技術の諸元を下記に示す．

開発環境: Eclipse3.4.0
API 実装: Java
DBMS: MySQL 5.1
Web サーバ: Apache Tomcat 5.5
Web サービスエンジン: Apache Axis 2.1.3

4. ケーススタディ

4.1 エネルギー浪費行動自動検出サービス

提案プラットフォームの有用性を確認するために，**エネルギー浪費行動自動検出サービス**の実装を行った．エネルギー浪費行動とは,「つけっぱなし」に代表されるエネルギーの浪費につながるユーザ行動全般を指している．我々は先行研究 [4] において，デバイス状態ログからエネルギー浪費行動を検出する手法を提案している．しかしながら，先行研究においてはサービス自体がデバイス状態ログのマイニングを行っており，マイニング結果はエネルギー浪費行動を検出するためだけに利用されていた．提案プラットフォームであれば，必要なデータは第一段階マイニングにより事前に蓄積されており，サービスは第二段階マイニングによって必要なデータを取得することで，容易にエネルギー浪費行動を検出できると考えられる．

4.2 エネルギー浪費行動自動検出の流れ

先行研究において，我々はエネルギーの浪費行動を Type1: 長時間のつけっぱなし，Type2: ユーザ不在の機器使用，Type3: 効果が得られない環境下での機器使用，Type4: 競合する機器の同時使用の 4 種類に分類している．

ここでは，Type3 の例として “外が十分明るいのに天井照明がついていた” という浪費行動 (浪費行動 A と呼ぶ) について考える．提案プラットフォームの利用者は，浪費行動 A を検出するために以下の 4 ステップを踏む．

Step1: デバイス状態の推定
Step2: DCDL に従ってデバイス条件を記述
Step3: API を利用して必要なデータを取得
Step4: サービスロジックを記述

まず，Step1 として，ユーザは浪費行動 A を検出するために必要なデバイス状態を推定する．浪費行動 A を検出するために以下の 2 つのデバイス状態が必要である．

状態 c1: 照度センサーの値が 500lux 以上
状態 c2: 天井照明がついている

Step2 として，3.4 に従い，状態 c1，状態 c2 を抽出するためのデバイス条件を考える．デバイス条件はそれぞれ以下のように記述できる．

デバイス条件 dc1: `[outLightSensor.brightness >= 500]`
デバイス条件 dc2: `[light.power == true]`

さらに，これら 2 つのデバイス条件を DCDL に従って記述することで（図 7 参照），デバイス条件の登録を行う．

提案プラットフォームは記述された DCDL から定期的に第

```
<condition>
  <conditionID>2</conditionID>
  <description>照度センサーの値が 500lux 以上</description>
  <deviceID>outLightSensor</deviceID>
  <property>brightness</property>
  <operator>MORE_THAN_EQUAL</operator>
  <value>500</value>
</condition>
<condition>
  <conditionID>3</conditionID>
  <description>天井照明がついている</description>
  <deviceID>light</deviceID>
  <property>power</property>
  <operator>EQUAL</operator>
  <value>true</value>
</condition>
```

図 7　浪費行動 A を検出するために必要なデバイス条件の記述例

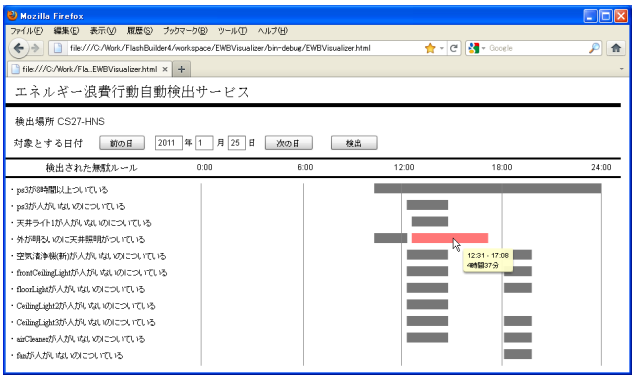

図 8　エネルギー浪費行動自動検出サービス

一段階マイニングを行い，ユーザ個別のデータベースに整形されたデータを蓄積していく．

次に，Step3 として，第二段階マイニングにおける `getTimeInterval` インタフェースの利用例を述べる．今，2011 年 9 月 1 日における浪費行動 A を検出しようとすると，デバイス条件 dc1 をマイニングするためのクエリ q_dc1 は

```
q_dc1 = {date=> "2011-09-01", deviceID=> "outLightSensor",
 property=> "brightness", operator=> ">=", value=> "500"}
```

となり，またデバイス条件 dc2 をマイニングするためのクエリ q_dc2 は以下のようになる．

```
q_dc2 = {date=> "2011-09-01", deviceID=> "light",
 property=> "power", operator=> "==", value=> "true"}
```

ここまでの一連の過程により，浪費行動 A を検出するためのデバイス条件成立時間幅が取得できた．後は，Step4 として，デバイス条件 dc1, デバイス条件 dc2 について，その条件が成立していた時間幅の共通集合を求めるようなサービスロジックを組むことで，浪費行動 A の検出が可能となる．

4.3　考　　察

図 8 に開発したエネルギー浪費行動自動検出サービスのサービス画面を示す．浪費行動を検出する日付を指定し，検出ボタンを押下することで，その日における浪費行動が自動で検出され可視化される．また，グラフにマウスオーバーすることで，浪費行動がなされていた時間帯についての詳細を知ることができる．また，浪費行動自動検出サービスを開発するために用いたデバイス条件や，そのデバイス条件によって導出されたデバイス条件成立時刻，デバイス条件成立時間幅は，他のサービスにも利用可能である点に注意したい．例えば，c2:“天井照明がついている”という状態は，“朝，天井照明がついたのはいつか”といった個人のライフスタイルを推定するサービスにも利用できると考えられる．

5.　おわりに

本論文では，デバイス状態ログから条件に合致する日時や期間を効率よく発見するためのプラットフォームを提案した．提案プラットフォームでは，DCDL によってデバイスの状態を記述し，マイニングを 2 段階にわけて実施することで，大容量となるログから効率的に必要なデータを取得できるよう設計を行った．また，エネルギー浪費行動自動検出サービスを実装することで提案プラットフォームの有用性を確認した．

今後の課題としては，複数の家庭におけるデバイス状態ログを集約できるようなデータベースやプラットフォームの提案を行いたいと考えている．現在，デバイス状態ログは，取得された家庭でのみ利用されることを想定しているが，複数宅のデバイス状態ログを集約することで，地域の生活やトレンドといったものを把握できると考えている．

謝辞　この研究の一部は，科学技術研究費（基盤研究 B 23300009，若手研究 B 21700077，研究活動スタート支援 22800042），および，ひょうご科学技術協会の助成を受けて行われている．

文　　献

[1] M.Nakamura, A.Tanaka, H.Igaki, H.Tamada, and K.Matsumoto. Constructing home network systems and integrated services using legacy home appliances and web services. *International Journal of Web Services Research*, Vol. 5, No. 1, pp. 82–98, 2008.

[2] C. Fitchett. Feedback on household electricity consumption a tool for saving energy. *Energy Efficiency*, Vol. 1, pp. 79–104, 2008.

[3] Hiroshi Igaki, Hideharu Seto, Masayuki Fukuda, and Masahide Nakamura. Mashing up multiple logs in home network system for promoting energy-saving behavior. In *Proc. of 8th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT2010)*, 2010.

[4] 北岡賢人, 瀬戸英晴, 柗本真佑, 中村匡秀. ホームネットワークシステムにおける機器状態ログからのエネルギー浪費行動の検出. 電子情報通信学会技術研究報告, 第 110 巻, pp. 037–042, 2011.

[5] 岡村雄敬, 中村匡秀, 柗本真佑. 嗜好アンケートに基づく個人適応型省エネ行動推薦手法の検討 ～家庭における空調サービスへの適用～. 電子情報通信学会 IN 研究会, 第 IN2011-63 巻, pp. 105–110, 2011.

[6] 江上公一, 井垣宏, 中村匡秀. ホームネットワークシステムにおけるサービス開発を容易化するネット家電標準データモデル. 電子情報通信学会 OIS 研究会, 第 75 巻, pp. 75–80, 2009.