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のための パーソナルリモコン開発フレームワーク

徳田　啓介†　稲田　卓也†　松本　真佑†　中村　匡秀†

† 神戸大学　〒 657–8501 神戸市灘区六甲台町 1–1
E-mail: †{tokuda,inada}@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，ネットワーク上から宅内家電を操作できるホームネットワークシステム (HNS) を構築しており，様々な種類の家電を単一のデバイスから操作可能で統合的なインタフェースを提供している．しかしながら，開発者側から提供される既成の操作インタフェースが全てのユーザにとって使いやすいものとは限らない．個人の嗜好の違いから利用機器の優先順位が異なることや，宅内環境の家電機器の違いなどがその原因である．本研究では，個人の利用スタイルに合わせてカスタマイズ可能な「パーソナルリモコン」の開発を容易化するアプリケーションフレームワークを提案する．提案手法を Android 端末用に実装し，実際の HNS を用いた評価実験を行った．
キーワード　HNS，パーソナルリモコン，ヒューマンインタフェース，アプリケーションフレームワーク，AndroidOS

Application Framework for Developing Personal Remote Controllers in Home Network System

Keisuke TOKUDA†, Takuya INADA†, Shinsuke MATSUMOTO†, and Masahide NAKAMURA†

† Kobe University　Rokkoudaityou 1–1,　Nada-ku,　Kobe,　Hyogo,　657–8501 Japan
E-mail: †{tokuda,inada}@ws.cs.kobe-u.ac.jp, ††{shinsuke,masa-n}@cs.kobe-u.ac.jp

Abstract　We have been studying integrated user interface which can operate heterogeneous home appliances by a single controller in the home network system (HNS). However, the ready-made interfaces developed by vendors are not always acceptable for all users. This is because individual preferences on appliances and operations vary, and also configurations of the HNS are different from house to house. In this paper, we present a practical application framework that allows users to develop their own *personal remote controller* for the HNS. We implement the proposed framework for the Android smart phones, and evaluate it by an experiment using a practical HNS.
Key words　HNS, personal remote controller, human interface, application framework, AndroidOS

1. はじめに

ネットワーク技術の発展に伴い，宅内に配置された家電機器をネットワークに接続し，計算機による制御を可能とすることで高い付加価値を提供する**ホームネットワークシステム (HNS)** [1] の研究・開発が進んでいる．家電機器をネットワークに接続することで，宅外からの機器状態モニタリングやセンサをトリガとした機器の自律制御，複数家電の連携制御などのサービスが実現されている．

我々はサービス指向アーキテクチャ(SOA) [2] の考えに基づいた HNS 環境「CS27-HNS」を構築しており，この環境下での様々なユーザインタフェースを開発している．例えば，画面上に表示された家電アイコンをタッチすることで家電を操作するタッチパネルインタフェースや，音声をトリガとして機器を操作する音声インタフェース [3] などが存在する．従来の一般的な家電の操作には家電機器一つにつき一つのリモコンなどの操作インタフェースが必要であった．一方，CS27-HNS 環境下では全ての家電機器を URI アクセスという標準的な API により操作可能であるため，機器の種類やベンダを問わずあらゆる家電を操作できる統合的なインタフェースを提供できるようになった．

しかしながら，上記の開発者側から提供される統合インタフェースは，全てのユーザの満足度を満たすものとは限らない．ユーザ個々の嗜好の違いから，よく利用する機器や機器操作の優先度が異なることが一つの原因である．また，HNS 環境では宅内に設置される家電の種類や個数があらかじめ決定されておらず，宅内環境の違いによって使いやすいインタフェースが異なる．より便利で使いやすい HNS インタフェースを提供す

るためには，このような個人の嗜好や環境に柔軟に適応したインタフェースを開発できる枠組みが必要である．

そこで本研究では，ユーザが自分の嗜好や環境に合わせて自らレイアウトを作成する個人適応型リモコン「パーソナルリモコン」を提案する．パーソナルリモコンの実現には以下の 2 つの要求を満たす必要がある．**要求 R1**：リモコン上の家電操作を制御する部分（HNS 操作制御部）とレイアウトが定義される部分（レイアウト定義部）を分離し，ユーザ自身がレイアウトを定義できる枠組みの開発．**要求 R2**：直感的かつ容易な操作でレイアウト定義を作成できる支援ツールの提供．

本研究では要求 R1 の実現を目指し，HNS 操作制御部とレイアウト定義部を分離するフレームワークの開発と，レイアウト定義のための言語「Personal Remocon Layout Language (PRLL)」の提案を行う．また，フレームワークの有効性と PRLL の作成コストを評価するための実験を行った．実験の結果，自身の好みに合わせて自由にレイアウトを作成できる点で既存の提供されるユーザインタフェースに比べ使いやすいといったコメントの他に，PRLL の表現能力の限界などを明らかにすることができた．

2. 準　　備

2.1 ホームネットワークシステム (HNS)

ホームネットワークシステム (HNS) とは，宅内の家電やセンサをネットワークに接続し，それぞれの機能を利用するための API を公開することにより，計算機による制御を可能とするシステムである．HNS を利用することで，宅外からの家電モニタリングや遠隔操作の他，複数家電の連携制御といった高い付加価値を生み出すことができる．

我々の研究グループは，実際の家電機器を用いた HNS(CS27-HNS と呼ぶ) を構築している．CS27-HNS では，様々な種類の家電をネットワーク越しに標準的な方法で操作するため，各家電の機能を Web サービス [4] として公開している．Web サービスとして公開された家電は，単純な URI へのアクセスにより操作することができる．例えば，TV の電源をつけるためには，`http://cs27-hns/TVservice/on` にアクセスし，また，TV のチャンネルを 8 に変更するには，`http://cs27-hns/TVservice/channel?param0=8` にアクセスするだけで家電を操作できる．

2.2 HNS における家電操作インタフェース

我々の研究グループでは，CS27-HNS 上に配置された複数の家電を単一の端末で操作可能な家電操作インタフェースを開発している．従来の家電操作は，個々の家電一つにつき一つの赤外線リモコン等の操作インタフェースを用意する必要があった．一方，CS27-HNS では全ての家電を URI 呼び出しという画一的な方法で操作可能なため，複数の家電を単一のユーザインタフェースから操作することができる．以下に CS27-HNS 上に配置された家電操作ユーザインタフェースの例を示す．

タッチパネルインタフェース: 液晶画面をタッチすることにより家電を操作するインタフェースである．空調，照明などのカテゴリを選択することで，そのカテゴリに属する家電を絞り込み操作することが可能である．例えば，空調カテゴリを選択すると，エアコン，空気清浄機，アロマなどの家電を操作できる．

携帯操作インタフェース: 携帯端末から家電を操作するインタフェースである．携帯端末上の Web ブラウザを利用することで，宅外からでも家電の操作や監視が可能である．

音声インタフェース: ユーザの音声によって家電を操作するインタフェースである．「扇風機 ON」と発音することで扇風機を実行できるほか，「寒い」「暗い」といったコンテキストを発話することで操作対象の家電をシステム側が推薦する機能を持つ．

2.3 研究の目的

上記の家電操作インタフェースは，すべてシステム開発者側から提供され，ユーザはそれらをそのまま利用する．しかしながら，各家庭に配置される家電が異なること，ユーザそれぞれの好みの家電やその使い方が違うことから，開発者側から提供された出来合いのユーザインタフェースが必ずしもユーザの満足を満たすとは限らない．より快適で使いやすい家電操作を実現するためには，ユーザ一人一人の嗜好や家庭の環境に柔軟に適応できる**パーソナルリモコン**を合わせて考えていく必要がある．

本研究におけるパーソナルリモコンとは，HNS の家電やサービスの操作インタフェースのうち，ユーザ自らがそのレイアウトとボタン操作時のアクションを自由にカスタマイズできるものを指す．HNS のためのパーソナルリモコンを実現するためには，以下の 2 つの要求を達成する必要がある．

要求 R1：個人ごとのレイアウトと HNS 操作制御部を分離する仕組みの開発． HNS ユーザインタフェースは一般に，操作ボタンやメニューバー等のレイアウト定義部と，ボタン押下等のイベントを検知し，家電サービス API を呼び出す HNS 操作制御部から構成される．これまでの HNS 操作インタフェースはこのレイアウト定義部と HNS 操作制御部が分離されておらず，ユーザ自身がカスタマイズすることはできない．パーソナルリモコンを実現するためには，レイアウト定義部を HNS 操作制御部から分離し，ユーザが自由に操作レイアウトを定義できるような仕組みが必要となる．

要求 R2：個人ごとのレイアウトを容易に作成できる仕組みの開発． 要求 R1 におけるレイアウトの構成要素としては，ボタンの配置座標，色，押下時のアクション，画面の遷移など，様々な情報を定義する必要がある．幅広いエンドユーザを対象としたパーソナルリモコンを実現するためには，視覚的かつ容易に操作可能なレイアウト作成のための GUI 支援ツールが必須である．

本研究では，特に要求 R1 の実現を目的とし，レイアウト定義部と HNS 操作制御部をうまく分離できるようなアプリケーションフレームワークを提案する．

3. 提 案 手 法

3.1 アーキテクチャ

要求 R1 におけるレイアウト定義部と HNS 操作制御部を分離するためのフレームワークを考える．提案フレームワーク

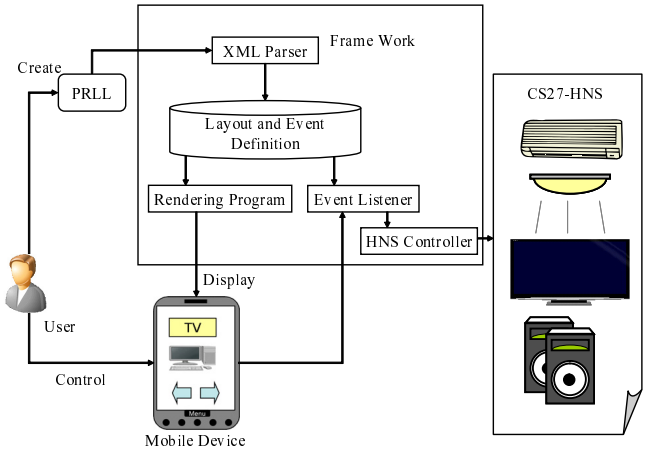


図 1　提案法のアーキテクチャ

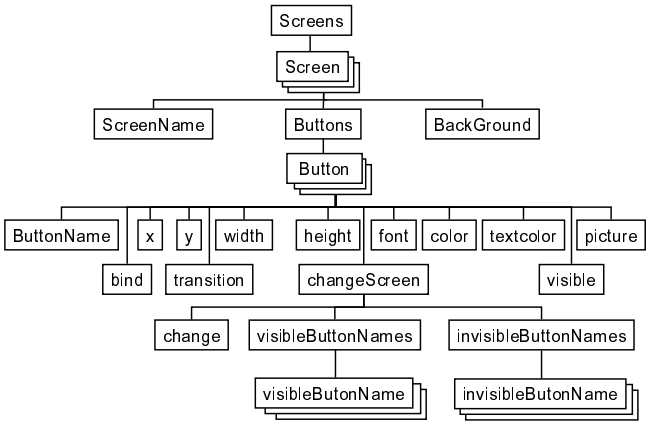


図 2　PRLL の構造

のアーキテクチャを図 1 に示す．まずレイアウト定義部に対して，レイアウト定義のための XML 形式の定義言語: Personal Remocon Layout Language (PRLL) を提案する．PRLL によって定義されたレイアウトをプログラム外部のファイルとして用意し，このレイアウトをリモコンプログラムが適宜読み，ユーザによるレイアウトのカスタマイズを実現する．

レイアウト定義ファイルは XML Parser を通じてリモコンプログラム内に読み込まれ，レンダリングプログラムによってユーザ端末にリモコン画面が表示される．またレイアウト定義ファイルにはレイアウト情報の他，ボタン押下時のアクションが併せて定義されている．HNS 操作制御部（図中の HNS Controller）はボタン押下時にイベントリスナーが検知したイベントを受け取り，定義されたボタン押下時アクションを実行する．

3.2　Personal Remocon Layout Language (PRLL)

PRLL はパーソナルリモコンのレイアウトを定義するための XML 形式の言語である．リモコンのレイアウト情報としては，画面に表示するボタン，ボタンの形式や配置座標，背景画像，色等の見た目に関する情報の他，ボタン押下時アクションに関する情報等が含まれる．ボタン押下時アクションには，操作対象の家電サービス URI，もしくはリモコン画面の遷移や一部の変更といった情報が定義される．

PRLL は，上記で述べたようなレイアウト作成に必要となる情報を XML 形式ファイルで定義する．PRLL によって定義可能なレイアウト情報構成要素を図 2 に示し，その説明を以下に示す．

- **Screens**: 複数の Screen を持つ．
- **Screen**: 操作デバイス上に表示される 1 つの画面を指し，1 つの Screen は複数のボタンから構成される．例えば，音量やチャンネルなどの TV 操作に関する操作インタフェース画面が 1 つあった場合，それが 1 つの Screen となる．また，操作対象の家電一覧が表示された画面も 1 つの Screen となる．
- **ScreenName**: Screen の名前を指す．
- **BackGround**: Screen の背景色，もしくは背景画像を指す．
- **Buttons**: 複数の Button を持つ．
- **Button**: ユーザが操作するボタンを指す．1 つの Button は配置座標や大きさ，色などのレイアウト情報と，Button 押下時のアクション情報から構成される．
- **ButtonName**: Button の名前を示す．
- **x**: Button が配置される x 座標を絶対座標で示す．
- **y**: Button が配置される y 座標を絶対座標で示す．
- **width**: Button の横幅を示す．
- **height**: Button の縦幅を示す．
- **font**: Button の中にテキストを表示する場合のテキストのフォントを決める．
- **textcolor**: Button に表示されたテキストの色を決める．
- **picture**: Button に貼り付ける画像を指定する．
- **color**: Button の背景色を決める．
- **bind**: Button 押下時の家電操作イベントを表す．具体的には，家電サービスのアクセス URI が記述される．
- **transition** : Button 押下時の画面遷移イベントを表す．具体的には遷移先となる Screen 名が指定される．遷移を伴わない Button にはこの transition の指定は不要である．
- **visible**: Button の可視/不可視を指定する．2 つの Button を用意し，visible の値を次に示す changeScreen で切り替えることにより，Button の見た目を擬似的に変更することが可能である．
- **changeScreen**: Button 押下時に見た目が変更される全 Button の情報が記述される．例えば，TV の状態が変化する Button を押下した際に，チャンネルや音量の現状態が表示された Button の見た目を変更する場合は，機器状態取得 Button の changeScreen に変更対象となる各 Button を指定する．
- **change**: changeScreen タグを利用するか否かを true/false で指定する．
- **visibleButtonNames**: 複数の visibleButtonName を指す．
- **visibleButtonName**: changeScreen を行う際に可視になる Button の名前を指す．ここで指定された ButtonName と一致する Button が再描画される．
- **invisibleButtonNames**: 複数の invisibleButton-

Name を指す.

- **invisibleButtonName**: changeScreen を行う際に不可視になる Button の名前を指す．ここで指定された Button-Name と一致する Button がスクリーン上から削除される．

PRLL を用いて定義されたレイアウト情報は XML Parser によってリモコンプログラムに読み込まれる．本研究では，XML Parser としてイベント駆動型 API の 1 つである Simple API for XML(SAX) [5] を使用した．

3.3 レンダリングプログラム

レンダリングプログラムは，PRLL に記述されたレイアウト情報を解釈し操作端末上にそのインタフェースに表示する．レンダリングプログラムは以下の 4 つの項目に基づいてレイアウト画面を作成する．<> で囲まれたものが PRLL の要素名を表す.

(1) 画面背景の決定: <BackGround> によってその Screen 画面の背景を決定する．色の指定は 16 進表記による単色の指定，もしくは画像ファイルの指定のいずれかである．

(2) ボタン配置の決定:

Screen 画面上での Button の配置を <x>，<y> によって決定する．指定する座標空間は，Screen 画面の左上を原点 (x, y)=(0，0) とし右, 下方向を正とする．

(3) ボタンデザインの決定: 各 Button の見た目を決定する．<width>，<height> により指定されたサイズで四角形のテキストボタンが配置される．Button の背景色は <color> で決定される．しかし，<picture> により画像ファイルが指定されている場合は，背景に画像が用いられる．Button 上の文字に関しては，<font> からテキストのサイズを決定し，<textcolor> からテキストの配色を決定する．

(4) ボタンの初期可視性: <visible> により指定された値によって，そのボタンが初めから画面に表示されているかどうかを判断する．

3.4 HNS 操作制御部

HNS 操作制御部は，イベントリスナーから検知されたボタン押下時のイベントをトリガとして，PRLL に記述された操作を実行する．ボタン押下時に実行する操作を以下に示す.

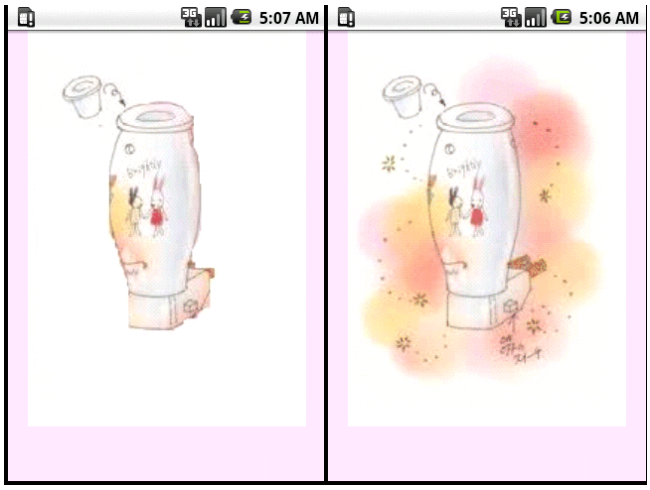
家電操作: <bind> で指定された URI にアクセスし家電サービスを実行する．ただし，<bind> に記述できる家電操作は 1 種類のみとする．

画面遷移: 画面遷移は，現在表示されている Screen 画面から別の Screen 画面に遷移する場合と，現在の Screen 画面の内容を変更する場合の 2 通りが存在する．Screen 画面全体の遷移は <transition> で指定される．複数機器の名前が指定された Screen 画面から，個々の機器の操作を行う Screen 画面に遷移する場合などは，この画面遷移を利用する．また，Screen 画面の一部を変更する場合の変更内容は <changeScreen> に格納されている．<change> の内容が true ならば画面変更を開始し，<invisibleButtonNames> の中にある名前のボタンを不可視とし，<visibleButtonNames> の中にある名前のボタンを新しく画面に表示する．

上記の家電操作と画面遷移は同時に実行することが可能である．例えば，TV の電源 ON ボタンを押すと TV の電源が ON 状態になり，同時に Screen 画面上の TV 電源 ON アイコンが表示されるといった，機器の状態を Screen 上に反映させることが可能である．ただし，家電の状態を実際に取得しているわけではないので，リモコンに表示されている家電の状態と実際の状態がずれが発生する可能性はある．

```
<Screen>
    <ScreenName>Aroma</ScreenName>
    <BackGround>#FFE1FF</BackGround>
    <Buttons>
        <Button>
            <ButtonName>Aroma_off</ButtonName>
            <x>0</x>                    <y>0</y>
            <width>320</width>          <height>400</height>
            <picture>aroma_off.png</picture>
            <bind>http://cs27-hns/AromaService/off</bind>
            <transition>false</transition>
            <changeScreen>
                <change>true</change>
                <visibleButtonNames>
                    <visibleButtonName>Aroma_on</visibleButtonName>
                </visibleButtonNames>
                <invisibleButtonNames>
                    <invisibleButtonName>Aroma_off</invisibleButtonName>
                </invisibleButtonNames>
            </changeScreen>
            <visible>false</visible>
        </Button>
        <Button>
            <ButtonName>Aroma_on</ButtonName>
            <x>0</x>                    <y>0</y>
            <width>320</width>          <height>400</height>
            <picture>aroma_on.png</picture>
            <bind>http://cs27-hns/AromaService/on</bind>
            <transition>false</transition>
            <changeScreen>
                <change>true</change>
                <visibleButtonNames>
                    <visibleButtonName>Aroma_off</visibleButtonName>
                </visibleButtonNames>
                <invisibleButtonNames>
                    <invisibleButtonName>Aroma_on</invisibleButtonName>
                </invisibleButtonNames>
            </changeScreen>
            <visible>true</visible>
        </Button>
    </Buttons>
</Screen>
```

図 3　Aroma 操作インタフェースに関する PRLL の記述例



初期画面　　アロマポットのONボタン押下後の画面

図 4　図 3 に示す PRLL によって作成されたレイアウト

3.5 PRLL によるレイアウトの例

PRLL を用いて定義したレイアウトの一例を図 3 に示し，レンダリングプログラムにより描画された結果を図 4 に示す．こ

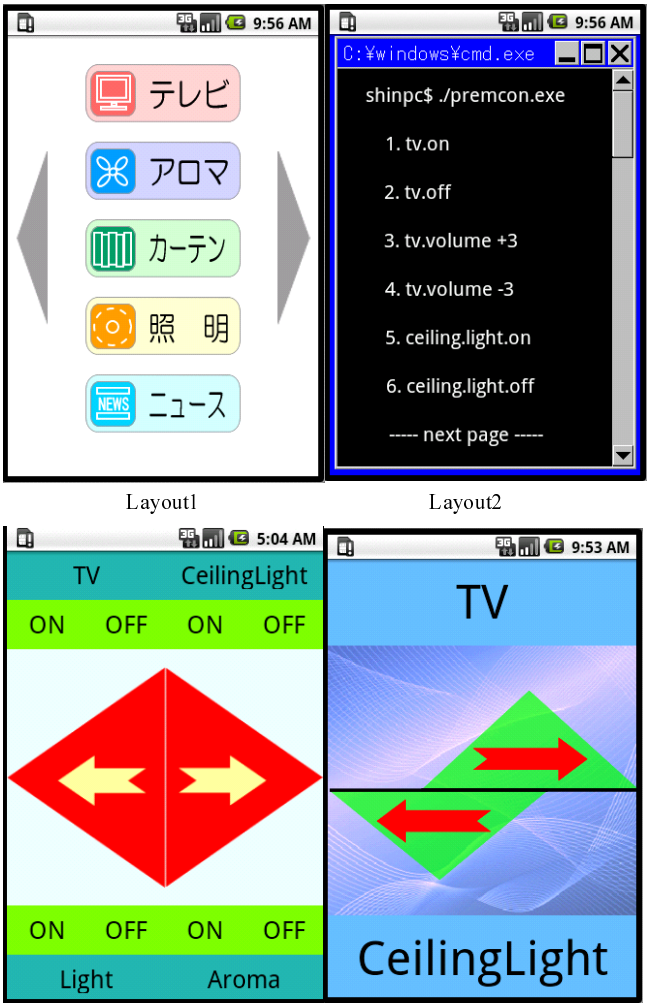


図 5　作成したレイアウト 4 種類

の例ではアロマポットの操作インタフェースを題材としている．初期状態ではアロマポットの電源は OFF 状態であり，画面中にはアロマポット OFF 状態の画像ボタンのみが表示されている．このボタンを押下することで，アロマポット ON のための家電操作 URI が呼び出され，同時に changeScreen で指定された画面遷移が行われる．具体的には，OFF 状態のボタンを不可視にし，ON 状態のボタンを可視にすることで，アロマポットの現在の状態をリモコン上に表示している．

3.6　実　　装

実装に利用した技術を以下に示す

- 開発言語: Java
- 対象 OS: Android 1.6 [6]
- 操作デバイス: Android Dev Phone 1

なお，提案フレームワークに基づいたパーソナルリモコンの開発には 100 人日を要した．また，パーソナルリモコンのクラス数は 12 個で，プログラム行数は 2,267 行である．

4.　評　　価

4.1　パーソナルリモコン作成実験

パーソナルリモコンの有用性と PRLL の記述コストを確かめるために，著者を含む 4 人の被験者に好みのリモコンを作成してもらった．4 つのパーソナルリモコンの Screen 画面を図 5 に，その特徴を以下に示す．

Layout1: 初期画面に操作対象の家電一覧が表示されており，各機器のボタンを押すことで機器ごとの操作リモコン画面に遷移する．PRLL 記述総行数 は 642 行．

Layout2: Windows のコマンドラインを模したレイアウトである．Layout1 のような機器ごとのカテゴライズはなく，機器の操作が初期画面上に記述されている．PRLL 記述総行数は 145 行．

Layout3: 優先度が高い ON/OFF ボタンを初期画面に配置したレイアウトである．ON/OFF 以外の機器操作を行う場合は，機器名のボタンを押すことで当該機器の操作画面に遷移し機器操作を行う．PRLL 記述総行数は 1472 行．

Layout4: 携帯端末などのディスプレイの小ささを補うために，他のレイアウトに比べボタンのテキストを大きめに表示している．PRLL 記述総行数は 1423 行．

作成実験の結果，個人にとって使いやすいと感じるレイアウトや機器操作の優先順位はそれぞれ大きく異なっており，個人適応型のパーソナルリモコンが利便性の向上に繋がると考えることができる．

4.2　ユーザから得られた意見

実際にパーソナルリモコンを作ってもらった結果，被験者から以下のようなコメントが得られた．

好評価な点

- 「自身の使用頻度が高い操作」をまとめて，かつ思った通りに配置できる点は非常に便利である．
- 既存のレイアウトと比べてもレスポンスに優劣がなく，また携帯端末から操作できるのは便利である．
- 完成したものを触ると，自分で作ったものが動いたということが嬉しい．
- 携帯の待ち受けのように，飽きたらまた変えられるというのが良い．

好意的な意見として，自分の使用頻度の高い操作を携帯端末から手軽に操作できるのが便利という意見が挙げられた．これはユーザ個人個人によって，使用頻度の高い家電やその操作が異なることが大きな要因である．また，自身の作成したレイアウトに愛着が持てたという意見のほか，レイアウトに不満が出てきた場合や，飽きがきた場合にその都度新しいレイアウトを作成できるというのは非常に有効であるなどの意見が得られた．これらの意見より，パーソナルリモコン有効性は確認できたといえる．

改善が必要であると感じた点

一方で，以下のような改善が必要であるというコメントも得られた．

- ユーザが XML ファイル形式をテキストエディタでそのまま記述するというのは非常に煩雑な作業である．
- 座標指定，サイズ指定などの視覚的な情報が非直感的であり，分かりにくい．
- 家電機器の現在状態が分からない．

- レイアウトの指定項目が足りず，自由にレイアウトを組めない部分がある．

多くの被験者から得られた批判的なコメントとして，PRLL を用いた XML ファイルの記述が困難であるという点が挙げられる．Layout3 や 4 のような画面遷移を伴うレイアウトの場合，1,000 行を超える XML を直接記述する必要があり，ユーザへの負担が大きい．また著者自身が作成した Layout4 の PRLL ファイル作成には 3 時間を費やす結果となった．同様に PRLL 作成の難しさに関する批判として，PRLL 記述内容の中心を占める視覚的な情報をテキストで記述することに対する不満が得られた．これらの問題を解決し，実用的なパーソナルリモコンを実現するためには 2.3 節で述べた要求 R2 の達成も不可欠であるといえる．

他にも PRLL の要素不足に関する批判として，家電状態の取得に関するコメントとレイアウトの自由度に関するコメントが得られた．HNS では家電機器の現状態を取得する API が機器ごとに用意されており，これを利用することで宅外からの家電モニタリングが可能となる．このような HNS の利点を活かすために，機器状態を取得しレイアウト上に表示できるような PRLL の拡張が必要であると考えられる．またレイアウトの自由度に関しては，ボタンの形状や余白といった見た目に関する情報の指定ができない点の他に，チェックボックスやスライドバーといったボタン以外の操作方法が欲しいというコメントが得られた．より柔軟で自由度の高いレイアウト作成を実現するためには，上記の点を踏まえた PRLL の拡張が必須である．

4.3 関連研究

実世界のデバイスを利用した，直感的で使いやすいインタフェースを実現するための研究が数多く行われている．木村の研究[7] や，安村の研究[8] などは，ユーザが普段使用するハサミや雑巾などの道具を操作デバイスとして用いる手法が提案されている．これらの研究では，ユーザの身近にあり使用方法を理解している道具を利用することで，直感的な操作インタフェースを実現している．また，香川らの研究[9] では，専用の端末でユーザが撮影した家電機器の画像を，ウィンドウシステムの通常アイコンのように操作できるインタフェースを提案している．さらに，上田[10] は対話型ロボットを利用した音声インタフェースを提案している．これらの研究はすべて，利便性の高い家電操作インタフェースの研究であるが，本研究におけるパーソナルリモコンはユーザ自身が自らの嗜好や環境に合わせて操作インタフェースを作成できるという点で異なる．

アプリケーション開発フレームワークという点での関連研究として Struts[11] が挙げられる．Struts は，MVC (Model, View, Controller) モデルに基づいた Web アプリケーション開発のためのフレームワークである．Struts は見た目に関するレイアウト部分と，画面遷移やボタン押下時アクションなどの操作部分を分離するという点で本研究との類似点が存在する．しかしながら，Struts は Web アプリケーションの開発者を支援するという目的であり，利用者は Java プログラミングや JSP などのプログラミング技術が必須となる．これはアクションの定義やレイアウトを高い自由度で定義できる利点を持つが，必要となる前提知識が膨大になるという問題も抱える．一方で，本研究の支援対象はエンドユーザであり，専門的な知識を持たないユーザでも自身の好みのレイアウトを作成できる環境の提供が目的である．

5. おわりに

本研究ではユーザが自分の好みに合わせてレイアウトを作成できるパーソナルリモコン開発フレームワークを提案し，レイアウト作成のための定義言語 PRLL の提案を行った．評価実験として，4 人の被験者に好みのレイアウトを作成してもらった．実験の結果，各ユーザの使いやすいレイアウトが異なっていることから，パーソナルリモコンの有効性が確認できた．

今後の課題として，まず要求 R2 の達成が挙げられる．被験者からも XML 記述の難しさが批判点として数多く挙げられており，GUI 支援ツールの開発が不可欠といえる．また 4.2 節で述べたような家電機器のモニタリング及び，レイアウト作成のさらなる自由度の確保のために，PRLL の改良・拡張も残された課題の一つである．また，本稿でパーソナルリモコンの利用環境として用いた AndroidOS には様々なセンサが埋め込まれており，これらのセンサを利用した操作の提供も開発予定である．例えば，上方向に傾けることで音量を上げるといったジェスチャによる家電操作を組み込むことで，より直感的で便利なパーソナルリモコンを実現できると考える．

謝辞 この研究の一部は，科学技術研究費 (若手研究 B21700077，研究活動スタート支援 22800042) 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] M.P. Papazoglou and D. Georgakopoulos, “Service-oriented computing,” 2003.
[3] 松原典行，江上公一，井垣　宏，“暗黙的なユーザ要求を抽出・推定するホームネットワークのための対話型音声インターフェース,” 電子情報通信学会技術研究報告，vol.109，no.450，pp.61–66，2010.
[4] “W3C web service activity”. http://www.w3.org/2002/ws/.
[5] “Simple API for XML”. http://www.sax.org/.
[6] “Android”. http://www.android.com/.
[7] 木村朝子，“道具と実世界指向インタフェース,” ヒューマンインタフェース学会誌，vol.12，no.2，pp.105–110，2010.
[8] 安村通晃，“日常生活のインタラクションデザイン ―実世界インタフェースの展開と今後,” 情報処理，vol.51，no.7，pp.803–811，2010.
[9] 香川景一郎，檀野隆一ら，“ネットワークを利用した携帯型情報家電マルチリモコン「オプトナビ」システムの基本実証,” 映像情報メディア学会誌，vol.60，no.6，pp.897–908，2006.
[10] 上田博唯，“スマートハウスと温かいインタフェース：Nict ユビキタスホームと京都産業大学ヨ home,” ヒューマンインタフェース学会誌，vol.12，no.1，pp.19–24，2010.
[11] “Struts”. http://www.struts.org/.