1S-04

iHACシステムにおける異種プロトコル間の機器連携手法の提案

江崎 敬俊 †1　梅山 莉奈 †2　鈴木 秀和 †2
†1 名城大学理工学部　†2 名城大学大学院理工学研究科

1 はじめに

近年，ネットワークを通じて操作することができる情報家電が普及しており，ECHONET Lite や DLNA（Digital Living Network Alliance）など様々な規格がある．そのためユーザはこれらの規格や通信プロトコルの違いに応じて操作アプリを使い分ける必要がある．筆者らはプロトコルの違いを意識することなく直感的に機器を制御することができる iHAC（intuitive Home Appliance Control）システムを提案している [1].

本稿では，iHAC システムの利便性をさらに高めるため，異種プロトコル間の機器を連携する手法について提案する．

2 iHAC システム

iHAC システムは操作アプリケーションに実装される iHAC フレームワークがプロトコルの違いを吸収し，機器にリビングや寝室などの位置情報を付加することにより，ユーザが機器を直感的に操作することができる．iHAC フレームワークには機器の探索や操作に関わる API が定義されており，これらが各プロトコルの通信処理 API を呼び出すことにより，複数の規格に準拠した家電機器を統合的に制御することができる．しかし，DLNA 対応テレビで映画を視聴することに連動して，ECHONET Lite 対応照明の照度を下げるといった，規格の違う情報家電を連携，協調して制御することはできない．

3 提案システム

3.1 構成

本提案では異種プロトコル間の機器連携を行うために，iHAC フレームワークを拡張し，レシピ処理 API 及び各種データベース（DB）を実装する．レシピとは連携させる機器や動作内容などをまとめた情報である．図 1 に提案システムの構成を示す．iHAC システムのユーザインタフェース（UI）をユーザが操作することにより，レシピ処理 API を呼び出す．レシピ処理 API は UI からの命令に従い，レシピ登録と参照，既存の API の呼び出しなどを行う．また DB にはレシピ，カテゴリ，ユーザ情報を登録する．

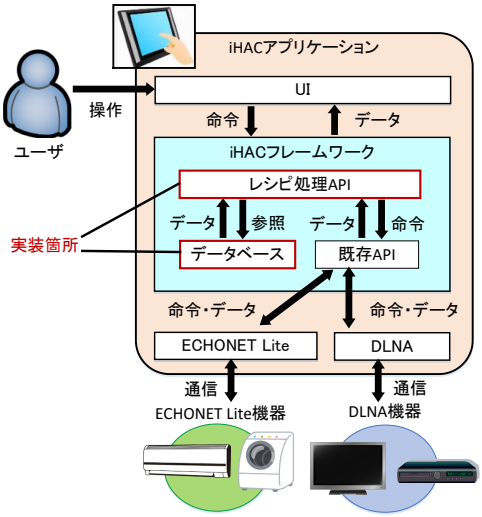

図 1　提案システムの構成

3.2 レシピ内容

表 1 にレシピ DB に登録するレシピの内容を示す．iHAC システムでは機器と機器の位置情報が関連付けられている．そのため，ユーザは位置情報から制御したい機器を絞り込むことができる．カテゴリとはレシピを機能により分類分けしたもので，「AV 機器操作」や「温度調節」などがある．選択したカテゴリによってレシピに設定できる機器を絞り込むことにより，機器が同一部屋内に多く登録されている場合でもユーザはレシピを設定しやすくなる．動作条件（トリガ）とは，レシピを実行するきっかけとなるものである．この項目には機器や

A Proposal of Device Cooperation Method between Different Protocols in iHAC System
Takatoshi Esaki[†1], Rina Umeyama[†2] and Hidekazu Suzuki[†2]
[†1] Faculty of Science and Technology, Meijo University
[†2] Graduate School of Science and Technology, Meijo University

表 1　レシピ内容

項目	例
レシピ名	全消灯，映画視聴
部屋名	リビング，寝室
カテゴリ名	温度調節，AV 機器操作
動作条件	照明の動作状態を ON
動作内容	エアコンの運転モードを冷房

環境の状態を登録し，条件を満たしたときに動作内容（アクション）を実行する．トリガを設定するレシピ例としては「温度センサで計測した室温が 30 度になった時，エアコンを冷房運転する」などがあり，ユーザは機器を直接操作することなく複数の機器を自動的に動作させることができる．トリガを設定しないレシピ例としては「外出する際に部屋の照明を消灯し，Blu-ray レコーダで再生中の動画を停止する」などがあり，ユーザは複数の機器を一つの操作で動作させることができる．

3.3　iHAC フレームワークにおける処理手順

動作はレシピ登録とレシピ実行に分けられる．

レシピ登録はユーザが UI を操作し 3.2 節の内容を登録する動作である．ユーザの操作によって入力されたレシピ情報は，UI 部からレシピ処理 API に渡される．レシピ処理 API は，渡されたレシピ情報をレシピ DB へ登録する．

レシピ実行は自動的に実行されるものと，ユーザの操作で実行されるものがある．前者の場合，レシピ処理 API は，機器の状態を取得する既存 API を呼び出し，動作条件を満たしたことをトリガとしてレシピを実行する．後者の場合，ユーザが UI のレシピ名を選択することにより，レシピを実行する．レシピの実行処理はどちらのレシピ実行時も同様で，レシピ処理 API は，アクションとしてレシピ DB に登録されている機器と動作内容を取得する．その機器のプロトコルをレシピ処理 API が既存の機器 DB から取得し，各プロトコルに対応した通信処理部へ機器の動作命令を送ることで異種プロトコル間の機器連携を実現する．

4　実装と評価

4.1　実装

本システムは iPad で動作する iHAC アプリケーションの iHAC フレームワークに，レシピ処理 API，

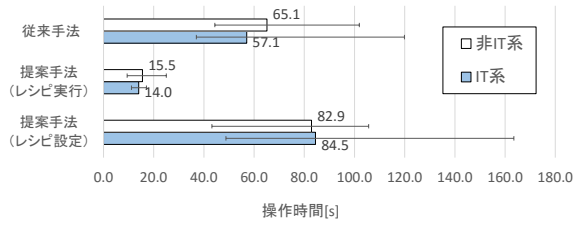

図 2　アプリケーションの平均操作時間

レシピの設定および設定したレシピを選択する UI を実装した．レシピ処理 API には，機器の状態変化を検知してレシピ DB のトリガと照合する機能，トリガ条件を満たしたとき，または UI でレシピが選択されたときにアクションを実行する機能を実装した．この API はレシピ DB から実行するレシピのアクションの内容を取得し，既存 API を呼び出してアクションを実行する．

4.2　評価

提案システムの有用性を評価するために，外出時を想定した機器操作実験（DLNA 準拠の Blu-ray レコーダで再生中の動画を停止し，2 つの ECHONET Lite 準拠の照明を消灯）を従来の iHAC と提案手法を用いて実施した．被験者（IT 系および非 IT 系の大学生各 10 名）に対して，全ての機器の操作を完了するまでの時間を計測した．

図 2 に両手法による機器操作時間および提案手法におけるレシピの設定時間を示す．提案手法は初回のみレシピを設定する必要があるが，操作時間を従来手法より 76 %削減できた．また既存手法は IT 系，非 IT 系被験者の差が大きかったが，提案手法はどちらの被験者もほぼ同じ操作時間であった．したがって，提案手法は IT スキルに関係なく操作できることが確認できた．

5　まとめ

本稿では iHAC システムにおけるに異種プロトコル間の機器連携手法を提案した．評価実験の結果，従来手法より大幅に操作時間を短縮しつつ，異種プロトコル間の機器連携を実現できることを確認した．

参考文献

[1] 梅山．他：情処学論（CDS）, Vol. 6, No. 1, pp. 84-93 (2016)