6R-06

環境情報に基づいて異種規格のIoTデバイス連携をサポートするiHAC Hubの提案

林 宏輔 [†1]　岡田 真実 [†2]　横地 リー紫音 [†1]　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

情報家電の通信プロトコルの違いを意識することなく機器を直感的に制御することができるiHAC（intuitive Home Appliance Control）システムが提案されている[1]. 文献[2]ではiHACシステムを応用して温度や湿度などの環境情報に基づいて機器連携を行うことが提案されているが，その実現方法は十分に考慮されていない.

本稿では，iHACシステムにMQTT（MQ Telemetry Transport）を用いて環境情報をリアルタイムに取得し，異種規格のIoTデバイス連携を実現する宅内用デバイスiHAC Hubを提案する.

2 iHACシステムの概要

iHACシステムは機器の登録や探索に関わるAPIが定義されたiHACフレームワークを導入し，このAPIが各通信処理部のAPIを呼び出すことで，プロトコルの違いを吸収している．また，ユーザがユーザインタフェース（UI）を操作し，温度や湿度などの環境情報の閾値，連携させる機器の動作内容などをまとめたレシピを作成し実行を選択すると，レシピ処理APIがレシピを実行し，環境情報に基づいた機器連携手法が提案されている.

しかし，現状では環境情報に基づいた機器連携を実現するためには，iHACシステムを導入した操作端末を常時起動させ，環境情報をセンシングしなければならない．また，iHACシステムはiOS向けの実装しかないため，実用性が十分に考慮されていない.

3 提案システム

3.1 構成

環境情報に基づいて機器連携を実現するためには，常に情報の変化を観測する必要がある．そこで，iHACフレームワークの機能を持つiHAC Hubを開発し，環境情報の収集機能を新たに実装する．図1に提案システムの概要を示す．iHAC Hubは，Raspberry Piなどのマイコンで実現することを想定し，これまでiOSのみサポートしていたiHACフレームワークをLinuxで動作するように再設計する．また，従来のiHACフレームワークの機能に加えてECAルール[3]に則り記述されたレシピの解析や環境情報の取得・登録処理などを行うECAルールエンジン（レシピエンジン），各種データベースを追加する．環境情報はIoTデバイスによりセンシングされ，MQTTを用いてクラウドに蓄積される．蓄積された環境情報をiHAC Hubが取得して処理を行うことにより，様々な通信規格の機器連携を行う.

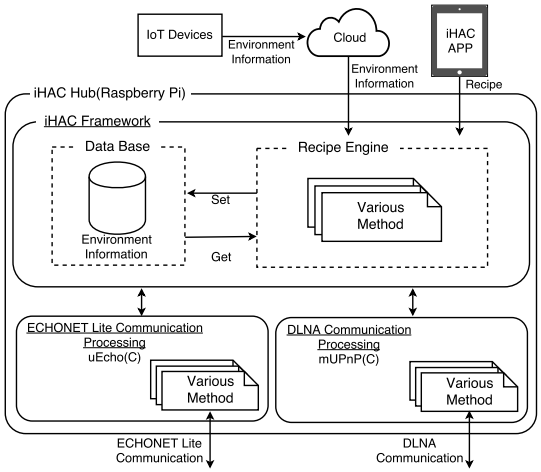

図1　提案システムの概要

3.2 動作

iHAC Hubによる機器連携の流れについて説明する．ユーザがiHACシステムを導入した操作端末からUIを操作し，レシピを作成・登録する．作成したレシピの実行を選択すると，iHACシステムはiHAC Hubにレシピ情報を送信する．レシピ情報を受信したiHAC Hubはレシピエンジンを起動しレシピを解析する．その後環境情報を取得するバックグラウンドプロセスを稼働させる．バックグラウンドプロセスによってiHAC Hubは Subscriberとしてクラウドからリアルタイムに環境情報の取得を開始する．取得した環境情報はレシピエンジンによってデータベースに登録され，レシピに登録された機器の動作条件を満たしているか比較を行う．iHAC Hubは動作条件を満たすまでクラウドから環境情報を取得し，レシピ実行の判断を繰り返し行う．動作条件を満

A Proposal of iHAC Hub Supporting Heterogeneous IoT Device Cooperation Based on Environment Information

Kosuke Hayashi[†1], Mami Okada[†2], Leeshion Yokochi [†2] and Hidekazu Suzuki[†2]
[†1] Faculty of Science and Technology, Meijo University
[†2] Graduate School of Science and Technology, Meijo University

たした場合，通信処理部を呼び出し機器制御を行う．

4　実装と評価

4.1　実装

環境情報に基づいた機器連携をサポートするため iHAC Hub のプロトタイプを Raspberry Pi3 で実装した．iHAC Hub は C 言語で実装し，ECHONET Lite 機器を制御するライブラリとして uEcho[4] を，MQTT を使用するためのライブラリとして Mosquitto を使用した．また，環境情報を蓄積するクラウドは CloudMQTT[5] を使用した．

4.2　動作検証

実装した iHAC Hub のプロトタイプの動作検証として，ユーザが「リビングの温度が 30 ℃以上になったらエアコンを起動する」というレシピを作成した場合を想定し，環境情報に基づいた機器制御が可能か確認する．今回の動作検証では，自作した温度センサ，ECHONET Lite 対応エアコン，iHAC Hub を同一ネットワークに接続し検証を行う．温度センサは Raspberry Pi3 で実装し，センシングした温度をクラウドに送信するプログラムを作成した．ECHONET Lite 対応のエアコンは TOSHIBA の RAS-B806DRH-W を使用した．iHAC Hub がレシピエンジンによってレシピを解析できたものとし，クラウドからリビングの温度を取得するバックグラウンドプロセスを稼働し，取得した温度が 30 ℃以上であればエアコンを起動するという制御を行った．

動作検証の結果，温度センサによってセンシングされた温度が 30 ℃以上となった際，エアコンが起動したため，環境情報に基づいた機器連携が可能であることを確認した．

4.3　評価

4.3.1　評価方法

提案システムにおける処理のうち，IoT デバイスからトリガを満たした環境情報が送信されてから実際に機器を制御するまでにかかった時間が実用上問題ないか，ECHONET Lite で規定されているタイムアウト時間と比較し評価する．図 2 に測定環境のネットワーク構成を示す．温度センサ，エアコン，iHAC Hub を同一ネットワークに接続し，温度センサから動作条件を満たした温度がクラウドに送信された時刻から，iHAC Hub が環境情報を受信したのち対象機器に制御メッセージを送り，機器から動作完了の応答が返ってくるまでの時間を計測する．試行回数は 10 回でその平均を算出する．

4.4　結果

表 1 に測定結果の平均値を示す．レシピの動作条件を満たした環境情報が温度センサから送信されてからエアコンを起動するまでに 734.90[ms] の時間を要した．

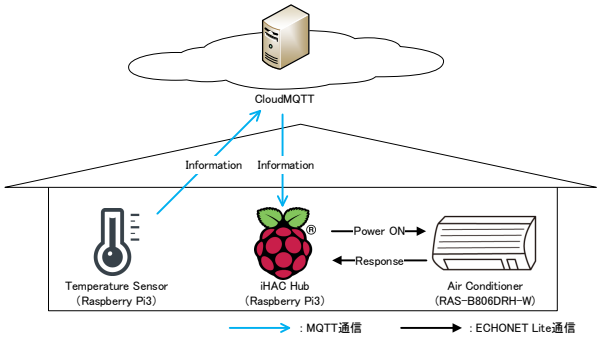

図 2　測定環境のネットワーク構成

表 1　測定結果

	提案システム
環境情報取得時間 [ms]	190.34
機器制御時間 [ms]	544.56
合計 [ms]	734.90

ECHONET Lite では機器が要求を受信してから応答を返信するまでのタイムアウト時間を 5000[ms] と規定されている [6]．提案システムの処理時間はタイムアウト時間の 14.68% であるため，提案システムの処理時間は実用上問題ないといえる．

5　まとめ

本稿では環境情報に基づいて異種規格の IoT デバイス連携をサポートする iHAC Hub を提案した．iHAC Hub のプロトタイプを作成して動作検証を行った結果，実用上問題ない時間で機器制御が可能であることがわかった．

謝辞

本研究の一部は JSPS 科研費 15K15987 の助成を受けたものである．

参考文献

[1] 梅山．他：情報処理学会論文誌　コンシューマ・デバイス＆システム, Vol. 6, No. 1, pp. 84–93, 2016.
[2] 江崎．他：第 79 回情報処理学会全国大会講演論文集, Vol. 2017, No. 3, pp. 65–66, 2017.
[3] Klaus R. Dittrich et al.: Rules in Database Systems, Vol. 985, pp. 3–20, 1995.
[4] GitHub-cybergarage/uecho：https://github.com/cybergarage/uecho
[5] CloudMQTT：https://www.cloudmqtt.com/
[6] 第 2 部 ECHONETLite 通信ミドルウェア仕様：ECHONET Consortium, 2015.