生体情報とスマートミラーを活用した IoT機器連携システムの設計

池内 紀貴[1] 酒井 恵梨香[1] 鈴木 秀和[1]

概要：体温や心拍数，血圧といった生体情報のセンシングを行い生活をサポートするIoT機器の普及に伴い，取得した生体情報を活用して機器制御が可能なスマート家電が増加しつつある．しかし，生体情報を用いてスマート家電を動的に制御してしまうと，ユーザが所望しない機器制御が実行されてしまい，利便性が損なわれてしまう可能性がある．筆者らは，温度や湿度などの環境情報に基づいてスマート家電やIoTデバイス間の連携を実現するiHAC（intuitive Home Appliance Control） Hubを拡張し，生体情報に基づく機器連携機能とスマートミラーを活用した機器連携レコメンド機能を追加することを提案している．本稿では，提案システムを実現するための具体的使用について検討し，実装したプロトタイプシステムを用いた性能評価実験を行う．iHAC Hubがユーザを検知してからスマートミラー部のレコメンドに対するレスポンスをiHAC Hubが受信し，機器を制御するまでに要した時間を測定した結果，平均1,639[ms]で実現可能であることを確認した．

Designl of IoT Device Cooperation System Using Biological Information and Smart Mirror

NORIKI IKEUCHI[1] ERIKA SAKAI[1] HIDEKAZU SUZUKI[1]

1. はじめに

ネットワーク技術の高度化により，身の回りの様々な“モノ”がネットワークに接続されるIoT(Internet of Things)が急速に普及した．IoTの普及に伴い，スマートフォンやタブレット，スマートスピーカーで宅内のスマート家電をコントロールしたり，IoTノードがセンシングしたデータに応じて自動制御を行うデバイスが増加し，QOL(Quality of Life)を向上させる新しいサービスやシステムが登場している．その中でも，人々の健康意識の向上や高齢化により体温や心拍数，血圧といった生体情報を活用したヘルスケアサービスが注目されている．例えば人の体温をセンシングして，室温環境を自動で整えるエアコンや鏡に生体情報を映し出し日々の健康意識を向上させるスマートミラーなどがあり，家庭で使用される家電や家具を利用したサービスやシステムが増加すると考えられる．

しかし，スマートホームなどで使用されるデバイスにはそれぞれ独自のアプリケーションを有しており，アプリケーション同士の通信が必要であるが，通信に必要なプロトコルが複数存在するため異なるプロトコル同士では通信を行えない．通信プロトコルの代表的なものとしてECHONET Lite [1] やDLNA(Digital Living Network Alliance) [2] などがある．ECHONET Liteは, エアコンや洗濯機などの白物家電を中心に対応した通信プロトコルであり，DLNAは，テレビやBlu-rayレコーダなどのAV機器に対応している．以上のことから，現在の家庭環境ではこれらの通信プロトコルの家電が混在する環境となっている．そのため, ユーザは制御したい機器に合わせてアプリケーションや操作方法を変更しなければならない.

そこで梅山らは, 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システムであるiHAC（intuitive Home Appliance Control）システムを提案している [3]．iHACシステムは機器の登録や探索に関するAPIが定義されたiHACフレームワークを導入することで，通信プロトコルの違いを抽象化している．また，iHAC

[1] 名城大学大学院理工学研究科
Graduate School of Science and Technology, Meijo University

システムを応用して，温度や湿度などの環境情報に基づく異種規格の IoT デバイス連携を実現する iHAC Hub [4] を提案している．iHAC Hub は従来の iHAC フレームワークから UI 部を除き，iHAC フレームワークと各規格の通信処理部から構成されるデバイスである．iHAC フレームワーク部から MQTT（MQ Telemetry Transport）を用いて環境情報をリアルタイムに取得することで環境情報に基づいた機器連携を可能にしている．また，佐藤らが提案したレシピエンジンにより iHAC Hub がレシピに基づいた処理を行うことが可能になり，機器の状態に基づいた機器連携も可能になった．

しかし，生体情報の活用が期待される中で，iHAC Hub は環境情報と機器の状態に基づいた機器連携のみを実現しており，生体情報に基づいた機器連携の検討は十分にされていない．また，生体情報は環境情報のように “室温が 30 度を超えると部屋が暑い” というような一定の閾値がないため，人によって感じ方が異なる．そのため，従来の iHAC Hub のような完全自動の機器連携システムではユーザが所望しない制御が行われる可能性があるため，好ましくない．

そこで本稿では，iHAC Hub とスマートミラーを組み合わせた生体情報に基づく IoT 機器連携システムを提案する．本システムを実装することにより，既存の iHAC Hub に生体情報を対応させることが可能となる．また．生体情報に基づいた機器連携内容をスマートミラーにレコメンドを提示することで，ユーザの所望しない機器制御が行われないような配慮を実現することも可能となる．

以下，2 章で既存研究を示し，3 章で提案システムについて述べる．4 章で提案システムのプロトタイプ実装および評価を行い，5 章でまとめる．

2. 既存研究

2.1 生活の質の向上を目指した宅内行動・生体情報収集システムと QoL アウェア家電制御

文献 [5] では宅内における QoL（Quority of Life）を向上させることを目的とし，在宅時の行動と生体情報を収集するシステムにより，収集したデータの分析を行うことで行動や状態に合わせた家電の自動制御が検討されている．QoL を測る指標を家電の制御による短期的な QoL の変化に着目しており，例えば起床や帰宅前に部屋の温度が調整されている場合のトレスを指標にしている．

そこで，洗濯や睡眠といった在宅時の生活行動時のストレスを LF（Low Frequency）/HF（High Frequency）比から測定できるものとして宅内 QoL を定義している．QoL 調査実験から，宅内行動と宅内 QoL の関係性，及び家電操作による宅内 QoL 向上の可能性が報告されている．まず宅内行動と宅内 QoL の関係性では料理，食事，睡眠などの様々な項目を測定した結果，睡眠の LF/HF 比が他項目に比べ大きいという結果が出ている．しかし，これは睡眠時にレム睡眠による交感神経の活発化が影響しているため，LF/HF が大きくなったと考えられる．また，他の項目では被験者ごとに行動による違いや行動ごとの被験者による違いを明確に現れていない．

次に，エアコンによる温度制御により QoL を向上させることが可能であることを確かめるために，主観的アンケートと実際の生体情報のデータを比較した QoL 調査実験が行われた．その結果，ユーザごとに快適と感じる温度は異なり，また LF/HF 比とアンケート結果は必ずしも一致するわけではないという結果を得ている．

これらより，生体情報の実際のデータとユーザの主観的感覚は異なる場合があると考えられるため，生体情報に基づく自動制御ではユーザが所望しない制御が実行される可能性がある．

2.2 生体情報を利用した自販の機器連携サービス

生体情報を利用した市販の機器連携サービスとして Withings Sleep [6] がある．寝具の下に敷いておくことで，睡眠状態や心拍数, 呼吸の乱れなどのデータのトラッキングを行うことが可能なパッド型のスリーピングセンサである．また，トラッキングのみならず，IFTTT（IF This Then That）との連携が可能である．IFTTT とは異なるプラットフォーム間における連携を可能にする web サービスであり，このサービスにより朝起きると，部屋の電気をつけるといった自動制御が可能となる [7].

しかし，高齢化やストレスなどにより夜中に意とせずに起きてしまう場合，自動制御により電気をつけることでユーザを不快にさせてしまう可能性がある．

2.3 iHAC システム

2.3.1 概要

iHAC システムは，規格の違いを意識することなく機器を直感的に制御できるシステムである．**図 1** に iHAC システムの構成を示す．iHAC システムは，UI（User Interface）部，iHAC フレームワーク部，各規格の通信処理部から構成される．

UI 部はユーザに対して，機器の情報や操作メニューの表示とユーザからの操作を処理する．表示する機器リストは，iHAC フレームワーク部から取得し，ユーザが操作したい家電を選択することで，その家電の操作メニューやコンテンツなどを表示する．ユーザが UI を操作することで，iHAC フレームワークに機器連携を行うためのレシピの作成や登録，実行などのレシピ処理を要求する．

iHAC フレームワーク部は家電の通信規格の違いを吸収する部分であり，UI 部から要求に従って各規格の通信プロトコルの通信処理部に命令を行う．また規格ごとにデータベースをもち，機器名，機器を一意に識別する識別子，部屋名などの位置情報，エアコンや照明器具などの機器の

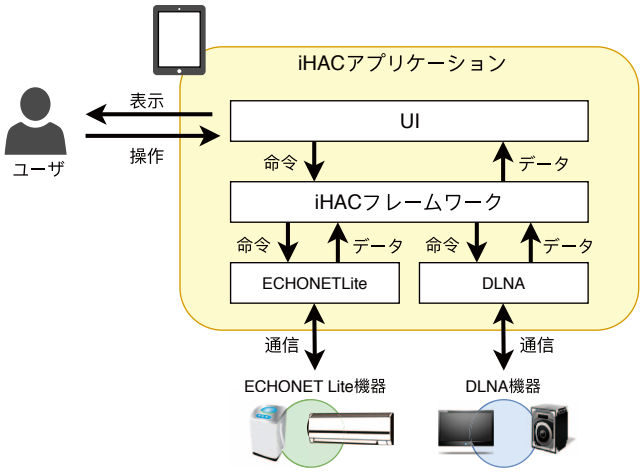

図 1　iHAC システムの構成

表 1　レシピ内容の例

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，家事
共有設定	YES，NO
動作条件	温度計の温度が 30 ℃以上
動作内容	エアコンの電源を ON

種類を示すデバイスタイプを保持している．

各規格の通信処理部は，iHAC フレームワーク部からの命令に従って，機器の探索や命令を行う．各規格の通信処理部は独立して実装することが可能で，ECHONET Lite や DLNA 以外の通信プロトコルに対応したものを追加実装することで，iHAC システムがサポートできる対象機器を拡張することが可能である．

2.3.2　レシピを用いた機器連携手法

iHAC システムを応用し，レシピを用いた機器連携手法が提案されている [8]．レシピとは**表 1** のような連携させる機器や動作内容などをまとめたものであり，レシピ名，部屋名，カテゴリ名，機器の動作条件，動作内容などを UI から設定する．ユーザによって設定されたレシピ情報は UI 部から iHAC フレームワーク部へと渡され，レシピ処理 API によって機器の制御や機器の状態を取得する API の呼び出しやデータベースへレシピの登録や参照を行う．ユーザが温度や湿度などの環境情報に関するレシピを作成することで環境情報に基づいた機器連携が可能となる．

しかし，ユーザが「リビングの温度が 30 ℃以上になったらエアコンの電源を ON にする」というレシピを作成した場合，現状では iHAC システムを導入した操作端末を常時起動させ，温度センサから送信される情報をセンシングし続けなければならない．また，iHAC システムは iOS 向けの実装しかなく，iPhone や iPad などの操作端末を常時起動するという用途は向いておらず，実現方法について十分に検討されていない．

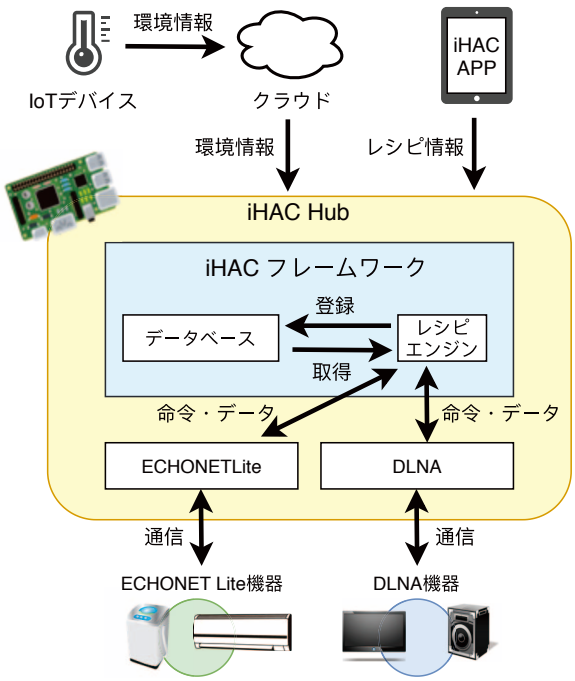

図 2　iHAC Hub の構成

2.4　iHAC Hub

2.4.1　概要

iHAC Hub は 2.3.2 節の課題を解決するために，Raspberry Pi などのマイコンを用いて環境情報や機器状態に基づいた異種規格の IoT デバイス連携を実現する宅内用デバイスである．**図 2** に iHAC Hub の構成を示す．iHAC システムで使用されていた UI 部，iHAC フレームワーク部，各規格の通信処理部の内，UI 部を除いた iHAC フレームワーク部，各規格の通信処理部で構成される．また，従来の iHAC フレームワークの機能に加え，レシピの解析を行うレシピエンジン [9] が追加されており，レシピエンジンはレシピの解析とレシピ内容に応じた処理の決定，環境情報や機器状態の取得，確認を行うなどの各種 API から構成される．レシピに使用される環境情報は 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で，環境情報に基づいた様々な通信規格の機器連携を行う．

2.4.2　レシピ仕様

iHAC Hub で使用するレシピの記述方法は，ECA ルールに則った ECA タグを用いて JSON 形式で記述する．ECA ルールとは，アクティブデータベースにおいて自動的に実行する処理を定義するために用いられ，ルール実行のトリガとなる “Event”，ルールが実行された際に確認される条件 “Condition”，条件を満たした際に実行される処理 “Action” から構成され，それぞれに対応したタグを用いて

```
{ "recipeType":2,
  "event": [
   { "number": 1, "fetch": "MQTT", "provider": "m13.cloudmqtt.com", "mqttTopic": "living/temperature" },
   { "number": 2, "fetch": "MQTT", "provider": "m13.cloudmqtt.com", "mqttTopic": "living/humidity" }
  ],
  "condition": [
   "$and": [
    { "number": 1, "operator": "$gte", "value": 30.0 },
    { "number": 2, "operator": "$gte", "value": 32.8 }
   ]
  ],
  "action": [
   { "controlDeviceId": 1, "controlDeviceStatus": "Power", "controlDeviceStatusValue": "On" },
   { "controlDeviceId": 2, "controlDeviceStatus": "lampColor" }
  ]
}
```

図 3　ECA ルールに基づいたレシピ例

ルールを表現する．例えば，「リビングで熱中症の危険性のある温湿度になった場合，エアコンを起動し，照明の色を変更する」というレシピは図 3 のように表現され，複数の動作条件や複数台の機器制御を行うといったルールを表現することが可能となる．

2.4.3　環境情報に基づいた機器連携手法

iHAC Hub ではレシピを使用して環境情報に基づいた機器連携を行う．まず，連携させる機器の動作や取得する環境情報の情報，動作条件となる閾値などの情報はレシピとして予め作成しておく．作成されたレシピは iHAC Hub へ送信され，iHAC Hub 側でレシピ内容を解析する．解析終了後，クラウドから環境情報を取得を開始する．取得した値が動作条件を満たした場合，各通信規格の通信処理部を呼び出し，機器制御を行う．

しかし，iHAC Hub は環境情報や機器状態に基づいた機器連携は検討されているが，心拍や血圧などの生体情報に基づいた機器連携は十分に検討されていない．

3.　提案システム

3.1　概要

生体情報に基づく機器連携を行う場合，機器連携を行う直前にユーザ自身が所望している制御かどうか確認する必要がある．ユーザの確認末，所望する制御には機器連携を実行し，所望しない制御には機器連携を中止することが望ましく，これによりユーザが所望しない機器制御が行われないように配慮することが可能となる．

そこで，提案システムではスマートミラーの機能と iHAC Hub を組み合わせることで生体情報に基づく機器連携を実現する．スマートミラーに機器連携が行われる前に制御内容のレコメンドを表示させ，“はい”．“いいえ” などの簡単なレスポンスにより機器連携を実行するか否か決定する．iHAC Hub にはスマートミラーとの通信を行うため，レコメンドやレスポンスなどの送受信を行う機能を追加する．また，ウェアラブルデバイスなどにより取得された生体情報を収集するクラウドを利用することで，生体情報をトリガとする機器連携を実現する．

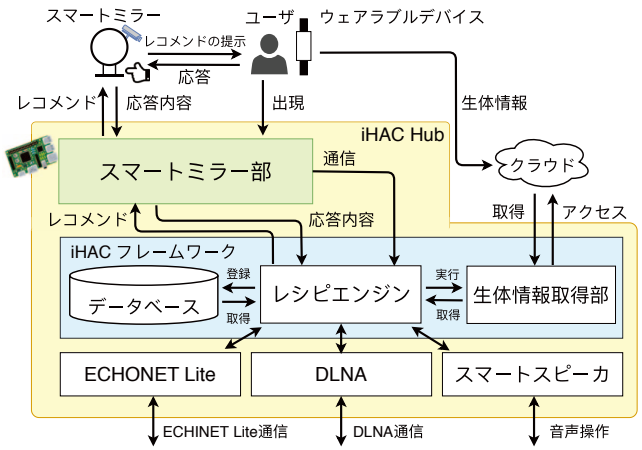

図 4　提案システムの構成

3.2　システム構成

図 4 に提案手法の概要を示す．従来の iHAC Hub で使用されていた iHAC フレームワークと通信処理部に加え，スマートミラーの機能を有したスマートミラー部によって構成される．提案システムは従来の iHAC Hub と同様に Raspberry Pi などのマイコンを想定してあるため，ディスプレイと接続することでスマートミラーを実現する．

スマートミラー部ではスマートミラーの前にユーザが姿を現すとスマートミラーに設置しているカメラがユーザを検知し，カメラからユーザの検知情報を受信する．また，iHAC フレームワークから機器連携の直前に制御内容の情報を受信し，スマートミラーの UI 部へレコメンドを提示する．ユーザは提示されたレコメンドにタッチや音声操作などによりレスポンスをし，レスポンス結果を iHAC フレー

```
{ "recipeType": 3,
  "event": [
   { "number": 1, "fetch": "WebAPI",
     "provider": "api.fitbit.com", "biologicalName": "activities/heart", "userId": "2" }
  ],
  "condition": [
    { "number": 1, "operator": "$gte", "value": 85.0 }
  ],
  "action": [
   { "controlDeviceId": 3, "controlDeviceStatus": "Play",
     "controlMethod": "DLNA", "recommendControl": "Yes" }
  ]
}
```

図 5 生体情報をトリガとするレシピ例

ムワークへ送信する．

iHAC フレームワークはデータベース，レシピエンジンに加え，クラウドなどから生体情報を取得する生体情報取得部から構成される．データベースは環境情報用テーブルと生体情報用テーブルが存在し，IoT デバイスやウェアラブルデバイスから取得した情報を蓄積する．レシピエンジンには機器連携に使用されるレシピの解析を行うと共に，環境情報や機器状態の取得，生体情報取得部を呼び出す機能を有する．

通信処理部は ECHONET Lite 通信処理部，DLNA 通信処理部，スマートスピーカ通信処理部の 3 つの通信処理部が存在する．ECHONET Lite 通信処理部は ECHONET Lite 規格の家電との通信を行う．DLNA 通信処理部は DLNA 規格の家電との通信を行う．スマートスピーカ通信処理部はスマートスピーカに向けて家電制御のトリガとなる音声の再生を行うことにより，人を介さない自動制御が提案されており [10]，その手法を用いることでスマートスピーカーに対応した家電の制御も可能となる．

3.3 レシピの追加フォーマット

機器連携を実現するレシピ仕様に対して，生体情報に対応したタグを新たに追加することにより，生体情報をトリガとした機器連携を実現する．**図 5** に「心拍数が自律神経の乱れと考えられる値を超えた場合，スピーカーから好きな音楽を流す」というレシピ例を示す．レシピの心拍数が自律神経の乱れと考えられる値は文献 [11] を参考とし，85bpm とする．生体情報を機器連携のトリガーとするために，“recipeType” として “3” を定義する[*1]．

event 部に記載するタグとして，新たに “biologicalName” タグ，“userId” タグを追加する．“biologicalName” タグには取得する生体情報名を，“userId” タグにはユーザの識別番号を記載する．

condition 部は既存のタグにより表現可能であるため，特に追加は行わない．

action 部に “recommendControl” タグを新規に追加する．“recommendControl” はレシピが動作条件を満たした場合，レシピの制御内容を実行するか否かのレコメンドを行うかを判断する．このタグによりレシピの設定のみで生体情報に限らず，環境情報や機器状態のレコメンドも可能である．

3.4 システム動作

図 6 に提案システムの機器連携の流れを示す．ユーザが生体情報に基づく機器連携を行うレシピを生成し，iHAC Hub にへ送信されているものとする．また，ユーザはウェアラブルデバイスを装着しており，計測された生体情報はクラウドに蓄積されているものとする．

まず, iHAC Hub が上記レシピを受信するとレシピ解析を行い，レシピタイプが生体情報である場合（“recipeType” が 3 の場合），“provider” タグで指定されたクラウドから “fetch” タグで指定された手法を用いて生体情報を定期的に取得し，iHAC Framework のデータベースに蓄積する．iHAC Framework はレシピの動作条件を満たすまで，生体情報の取得とチェックを継続する．レシピの動作条件を満たすと，スマートミラーからユーザ情報が送信されるまで待機する．

iHAC Hub が待機している間にユーザがスマートミラーの前に姿を現すと，スマートミラーに設置されているカメラが顔認証を行い，ユーザを特定する．スマートミラーがユーザを特定すると，iHAC Hub にユーザの情報（“userId”）を送信する．

iHAC Hub がスマートミラーからユーザの情報を受信すると，当該ユーザのリアルタイムまたは蓄積された生体情報をスマートミラーに表示する．さらに，特定したユーザのレシピが動作条件が満たされていれば，iHAC Framework はスマートミラー に機器制御に関するレコメンドを送信する．スマートミラーは受信したレコメンド内容を表示する．ユーザは表示されたレコメンドに対して，タッチ操作

[*1] “1” は機器状態をトリガーとする場合．“2” は環境情報をトリガーとする場合．

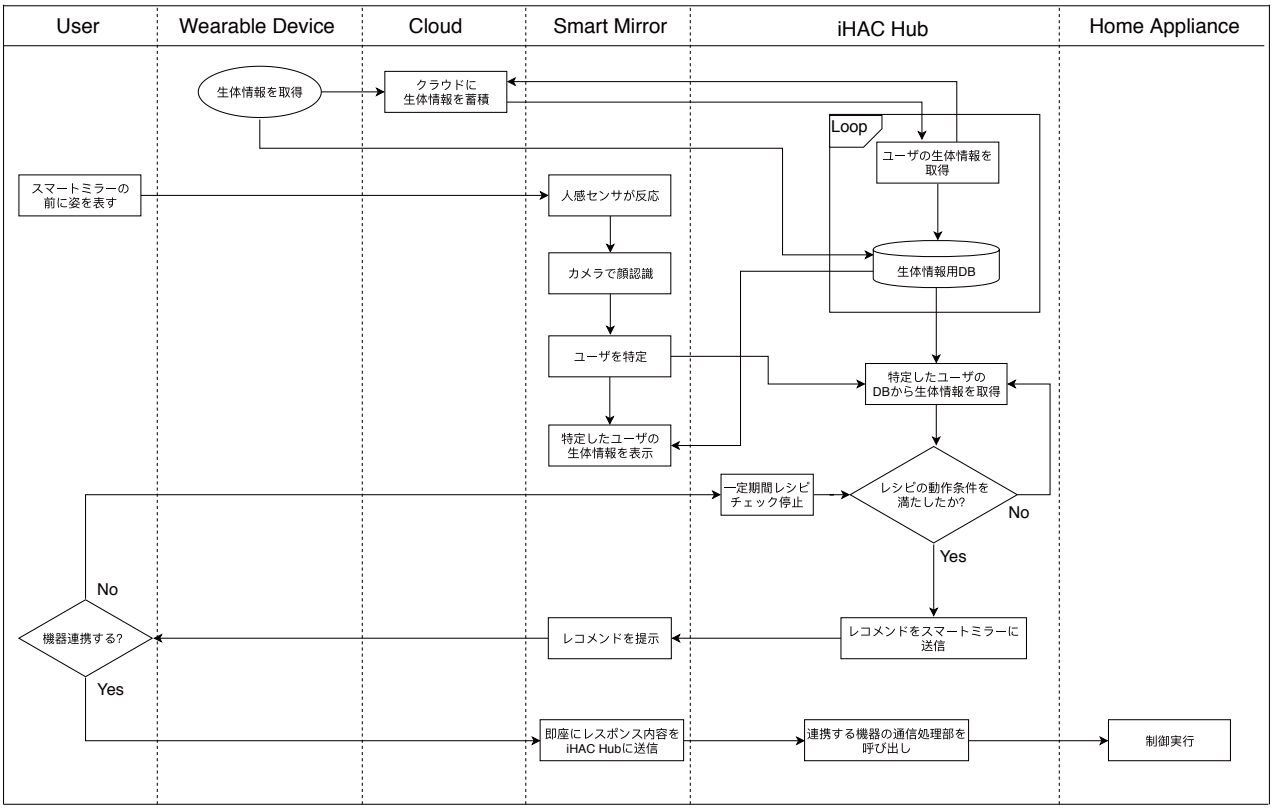

図 6　提案システムのフローチャート

または音声命令で例えば “Yes” を応答すると，動作条件を満たしていたレシピで定義された機器連携制御を行うために，通信処理部に命令を行う．なお，ユーザが提示された機器制御に関するレコメンドに対して “No” を応答すると，レシピで定義された機器連携制御は行わず，一定時間レシピチェックを停止する．

以上の処理により，スマートミラーを通じてユーザに対して即座にレコメンドを提供でき，かつユーザの意思に基づいて機器連携制御を容易に実現することができる．

4. 実装・評価

4.1 iHAC Hub の拡張

iHAC Hub で生体情報に基づいた機器連携を行うため，スマートミラーの機能を追加した iHAC Hub のプロトタイプを Raspberry Pi 3 Model B+で実装した．

図 7 にプロトタイプのモジュール構成を示す．スマートミラーの機能はオープンソースである MagicMirror^2 [12] を使用し，ユーザの情報を受信する機能と UNIX ドメインソケットを利用してレシピエンジンとの通信を行う機能の実装を行った．ユーザの検出及び特定はエッジデバイス上で AI による画像解析を行い，解析結果を webhook により別のプラットフォームに送信することが可能な Actcast [13] を用いた．スマートミラー部では Actcast が特定したユーザの情報を受信するために，node.js の Web フレームワークである Express [14] を使用して，webhook の受信を行った．また，UNIX ドメインソケットによる通信を可能にするライブラリである node-ipc [15] を使用した．

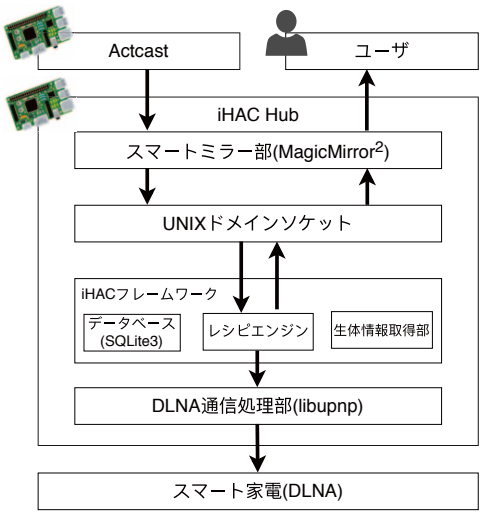

図 7　プロトタイプのモジュール構成

レシピエンジンには生体情報をトリガとするレシピに対応させるため，レシピフォーマットで追加したタグを処理する機能も実装し，生体情報を取得するプロセスの稼働，取得した生体情報をデータベースにから取得する機能，レシピの動作条件を満たしたか判断する機能を実装した．また，レシピの動作条件を満たすと，スマートミラーへレコメンド内容の送信／レコメンドに対するレスポンス結果の受信を行う機能も実装した．

生体情報を取得するプロセスではFitbit APIを利用してFitbitサーバから生体情報の取得し，データベースへ生体情報の登録を行う．データベースはSQLite3を使用して，生体情報テーブルおよびユーザ名テーブルを定義した．

生体情報はWebAPI [16] を利用してFitbitクラウドから取得する．DLNA機器を制御する通信処理部では，DLNA機器を制御するためにlibupnp [17] というライブラリにより，UPnPプロトコルを使用した．通信処理部は，レシピエンジンからIPアドレスやDLNA機器のプロパティなどを受け取り，同一ネットワーク上のDLNA規格に対応した機器同士を通信させることで，機器制御を行う．

4.2 評価

実装したプロトタイプシステムを用いて，「心拍数が自律神経の乱れと考えられる値を超えた場合，スピーカーから好きな音楽を流す」というレシピを実行した場合を例に，iHAC Hubの処理時間等を計測することにより，実用上問題ない応答時間および制御時間を達成できているかを評価する．　**図 8** に計測で使用したネットワーク環境を示す．生体情報と機器連携の動作検証を行うために，心拍数を計測できるフィットネストラッカーとしてFitbit Charge 3を筆者が装着した．また，制御対象となるDLNA機器として，Windows PCにIntel提供のDMSソフトウェアであるIntel Tools for UPnP Technologiesを稼働させてメディアサーバを構築した．また，DMR（Digital Media Renderer）にはSONY社製ワイヤレススピーカーSRS-X77を利用した．

本稿では，次の3種類の処理時間を計測することにより，提案システムの性能を評価する．

- **レシピチェック時間**：提案システムが生体情報を取得してから，レシピチェックが行われるまで
- **レコメンド表示時間**：iHAC HubがActcastから検知されたユーザ情報を受信してから，スマートミラー部にレコメンドを送信して，UI部に提示するまで
- **機器制御時間**：ユーザがレコメンドに対してレスポンスを行い，その結果に応じた機器制御が行われるまで

計測回数は100回であり，各処理時間の平均を算出する．

表 2 に計測結果を示す．提案システムが生体情報を取得し，機器制御が行われるまでに平均で1,639[ms]の時間を要した．文献 [9] によると，従来システムの平均機器制御時間が1,302[ms]であることから，提案システムの処理時間は従来システムに比べてわずかな増加であることが確認できた．なお，レコメンドを表示してから機器制御が行われるまでの間には，ユーザがレコメンド内容を認識，判断して応答を返すまでの所作に要する時間が発生するが，これらの時間は秒単位を要すると考えられる．したがって，提案システムにおける処理時間は実用上問題ないといえる．

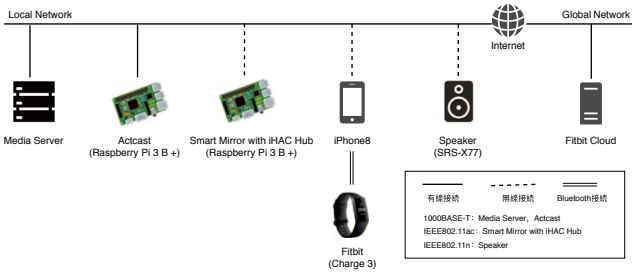

図 8　計測で使用したネットワーク環境

表 2　計測結果

	処理時間 [ms]
レシピチェック時間	1,003
レコメンド表示時間	123
機器制御時間	513
合計	1,639

5. まとめ

本稿では，生体情報とスマートミラーを活用したiHAC Hubによる IoT機器連携システムを提案した．提案システムでは，IoT機器連携の直前にスマートミラーに制御内容のレコメンドを行い，ユーザのレスポンスに応じて機器連携を行うことが可能である．これによりiHAC Hubような完全自動機器連携システムで蓋然的であるユーザの意に介さない制御が行われない事を可能にしている．

提案システムのプロトタイプを実装して実環境において動作確認および性能評価実験を行った結果，実用上問題のない処理時間でスマートミラーへのレコメンド，またレスポンスに応じてスマート家電と連携して制御できることを確認した．

参考文献

[1] : ECHONET Lite. `https://echonet.jp/`.
[2] : DLNA. `http://www.dlna.org/`.
[3] 梅山莉奈，増田剛志，鈴木秀和：規格の違いを意識しない直感的家電制御システムの提案，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. 6, No. 1, pp. 84–93 (2016).
[4] 林宏輔，鈴木秀和：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，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. 10, No. 1, pp. 40–49 (2020).
[5] 佐々木渉，大西晃正，三崎慎也，諏訪博彦，藤本まなと，水本旭洋，荒川豊，木村亜紀，三木智子，安本慶一：生活の質の向上を目指した宅内行動・生体情報収集システムとQoLアウェア家電制御の検討，*SIG-SAI*, Vol. 34, No. 1, pp. 1–8 (2019).
[6] : Withings Sleep. `https://www.withings.com/jp/ja/sleep-analyzer`.
[7] : IFTTT. `https://ifttt.com/`.
[8] 江崎敬俊，梅山莉奈，鈴木秀和：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，第79回全国大会講演論文集，Vol. 2017, No. 1, pp. 65–66 (2017).

[9] 佐藤里奈子，鈴木秀和：iHAC Hub における IoT デバイス連携のレシピ解析を行う ECA ルールエンジンの実装，第 81 回情報処理学会全国大会講演論文集，Vol. 2019, No. 1, pp. 171–172 (2019).
[10] Hayashi, K. and Suzuki, H.: M2M Device Cooperation Method Using iHAC Hub and Smart Speaker, *2020 IEEE International Conference on Consumer Electronics (ICCE)*, pp. 1–3 (2020).
[11] : 日本人間ドック学会:判定区分 (2018 年 4 月 1 日 改 定)，https://www.ningen-dock.jp/wp/wp-content/uploads/2013/09/e128ee593f3401bb50e5cb4dfc701389-2.pdf.
[12] : MagicMirror. https://github.com/MichMich/MagicMirror.
[13] : Actcast, https://actcast.io.
[14] : Express, https://expressjs.com/ja/.
[15] : node-ipc, https://www.npmjs.com/package/node-ipc.
[16] : Web API - Fitbit SDK, `https://dev.fitbit.com/build/reference/web-api/`.
[17] Johan Pascal, S. B.: libupnp. `https://github.com/BelledonneCommunications/libupnp`.