太陽光発電とバッテリーを使ったIoTデバイスの動的なデータ送信よる省電力化

杉本 一彦[1,a)]　串田 高幸[1]

概要：外部電源に接続されていない IoT デバイスでは現状, 1 次電池を用いて駆動しているのが一般的である．しかし 1 次電池は使い切りの目的が前提となっており再充電が行えず電池残量がなくなった際は電池交換をしなければならず人的資源の確保やユーザへの負荷が大きくなってしまう．その問題を解決するために本研究では太陽光発電と 2 次電池を用いた IoT デバイスの電力管理手法を提案する．この手法では, 発電量や 2 次電池の残量, デバイスの消費電力から動作を継続的に持続させるためにスリープやネットワークへの通信を自動的に制御する．またその際に発生するセンサーデータの送信ができない場合や応答がない場合におけるユーザとの対応手法についても併せて言及する．

1. はじめに

昨今，インターネットに接続された多数のマシン (Internet of Things) は，日常生活をより便利にし，有益なサービスを提供するのに役立つ膨大な量のデータを生成している [1].

大半の IoT デバイスは大規模発電所から家庭や工場へ届く電線を伝い AC 電源のような外部から電力供給を受けることを前提に設計をされている．またウェアラブルデバイスのような小型のポータブルデバイスはデバイス上の再充電可能なバッテリーからの電力供給が期待でき，且つ月，年単位での連続した長時間の動作を想定していない．以上の場合では IoT デバイスにおけるバッテリーや消費電力はさほど大きな問題にはならない．

1.1 背景

太陽光発電は, 基本的にその発電量は時間や場所の周りの環境に依存をする．エネルギー源の性質, エネルギーがランダムな時間に任意の量で変化するため, 収集されたエネルギー量は予測ができない可能性が高い [2]．太陽光発電を利用する上で自らがどれほどの電力を消費しているのか，どのくらい発電しているのか，残りのバッテリー残量はどのくらいなのか, その変化に合わせて対応する必要がある．

具体的なユースケースとして, 図 1 のようなビニールハウス内の環境監視が例として挙げられる．

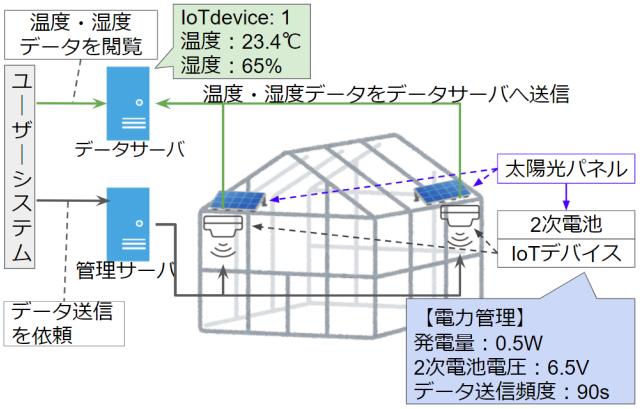

図 1　ユースケース図

これは太陽光発電と 2 次電池で駆動している IoT デバイスを用いてビニールハウス内の温度と湿度を測定しているケースを想定している．この場合, まず IoT デバイスからの温度と湿度データはユーザが送られてきたデータを閲覧できるためのデータを保管できるサーバ (データサーバと記載) へ送られる．温度と湿度データの送信間隔はユーザが設定を行う．例えば，気温や湿度は 1 時間であっても複雑に変化する．それは天気や気圧の変化，光照，外気温という外部からの影響によるものである．そのためビニールハウス内の温度を適切に管理するために 5 分に 1 回といったような短い間隔で「温度湿度データを送信する」,「定めた頻度で温度湿度データを送信する」といったリクエストというものを設定する．その設定に合わせて管理サーバは IoT デバイスへ送信間隔を設定する．また電力管理は IoT デバイスにて行うが, 太陽光発電からの発電量は予測をす

[1] 東京工科大学コンピュータサイエンス学部
〒 192–0982 東京都八王子市片倉町 1404-1
[a)] C0117164

る可能性は低く 2 次電池の残量とデータを送信する頻度を考慮し IoT デバイスのスリープ時間や通信時間を変更する必要がある．

1.2 課題

ユースケース図では 2 つポイントがある．まず 1 つ目は電力管理である．太陽光発電からの発電量は不規則であり無視はできない．しかしデータの送信を行わなくてはならない．現状では IoT デバイスにユーザからの「データを送信する」というリクエストは管理サーバより指示が送られてくる, もしくは既にスリープモードからウェイクアップするスケジューリングがなされている．これは 2 次電池の残量や発電量を考慮されていない．ビニールハウスの監視には温度・湿度データの送信間隔は最低でも 10 分に 1 度測定を行い温度を管理する必要があるため消費電力は決して小さいものではない．そのため 2 次電池の残量が足りず, 発電量も少ない状態でウェイクアップやネットワークへの接続をしてしまうと思わぬ電池切れで IoT デバイスが停止をしてしまう問題がある．IoT デバイスが停止してしまうと再度の起動は人の手で行わなくてはいけないため不意の停止はサービスに支障をきたす要因となりえる．

2 つ目は, ユーザのリクエストの処理方法である．IoT デバイスがスリープ中の場合はサーバとの接続が取れておらず, サーバがユーザーよりリクエストを受け取ったとしても即時に IoT デバイスにリクエストを送信することができない．そのためリクエストが処理されずリジェクトやエラーをユーザに返してしまう可能性がある．これはユーザへの意図とは反している処理結果となってしまうため, ユーザ側のシステムでエラーやデータ処理ができないといった問題が発生してしまう危険性もある．

2 章では本研究に関連した先行研究について述べる．3 章では本研究の提案について述べる．4 章では, 提案した手法について実機を用いた実装について述べる．5 章では実装と評価について述べ，6 章ではこの論文で述べてきたことに対する議論を行う．7 章では今後の研究の課題と方向性について述べる．

2. 関連研究

太陽光発電を用いた IoT によるアプローチでは，公共の天気予報に基づく IoT デバイスでの太陽エネルギー予測がある [3]．リソースに制約のある IoT デバイスにてパフォーマンスを低下させることなく動作させるために、太陽光発電は IoT の多くのシナリオで重要であり，太陽エネルギーの予測は資源の効率的な管理と利用に必要としている．機械学習はすでに大規模発電所の太陽光発電を予測するために使用されているが，簡単に入手できる公共の気象データに基づいてさまざまな機械学習方法を使用する方法について調べている．

IoT デバイスのエネルギー問題に関するアプローチでは，エネルギーハーベスティングセンシングでバッテリーを再考する論文がある [4]．非充電式バッテリーを用いて IoT デバイスを最初は動作させていた．しかし合理的なサイズの一次電池で達成可能なセンサノードの寿命が短いため、動作を維持するためにノードのために頻繁に電池を交換する必要がある．その解決策としてエネルギーハーベスティングに研究者は目を付けた．バッテリー予測問題については、固定ラウンドロビンアクセス制御ポリシーを採用し、RL ベースのアルゴリズムを開発して、エネルギー源に関するモデル知識がなくても予測損失（エラー）を最小限に抑える [5].

IoT デバイスの省電力化について, 処理の増加による消費電力増加を抑えるため処理の分散手法を述べている [6]. 太陽光発電で稼働する場合において, 消費電力が増加した場合には発電量で対処ができないため負荷を分散させることが重要であり, 本研究と合わせて利用することが望ましい．

バッテリー依存を抑えるためにソーラーハーベスティング (太陽光発電) を用いて駆動させることを目的としている論文である [7]．太陽光発電を用いたものであるが, 恒久的な動作時間の延長ではなくあくまで動作時間の延長としており, 長期間動作としての電力管理は提案されていない．また発電量の変化には言及しているが IoT デバイスが停止した場合について考慮されていない．対して本研究ではエンドユーザに対しても焦点を当てている．

3. 提案

本研究の目的は太陽光発電の発電量に合わせて IoT デバイスの動作スケジュールを動的に決定する手法と, その動作スケジュールによりユーザからのリクエストに対する応答が返せない場合やスリープ中であった際のリクエストのユーザとの合意手法について提案する．この手法によりユーザがリクエストをした際, レスポンスの遅延や可否について合意を得た上で IoT デバイスの動作スケジューリングが可能となる．電力管理とユーザリクエスト処理のアーキテクチャを図 2 に示す．

電力管理は IoT デバイス内の Power management にて, ユーザーのリクエスト処理は Management server(管理サーバ) 内の User request process にて処理を実施する．

3.1 User request process

図 2 での Management server での User request process の具体的な説明をこの章で行う．

図 3 はユーザリクエストの処理方法について示している．ユーザからのリクエストというのは背景でも述べたようにある程度決まった頻度 (frequency) で送信を行うものである．例えば, 図 3 にも示しているような「Want to send

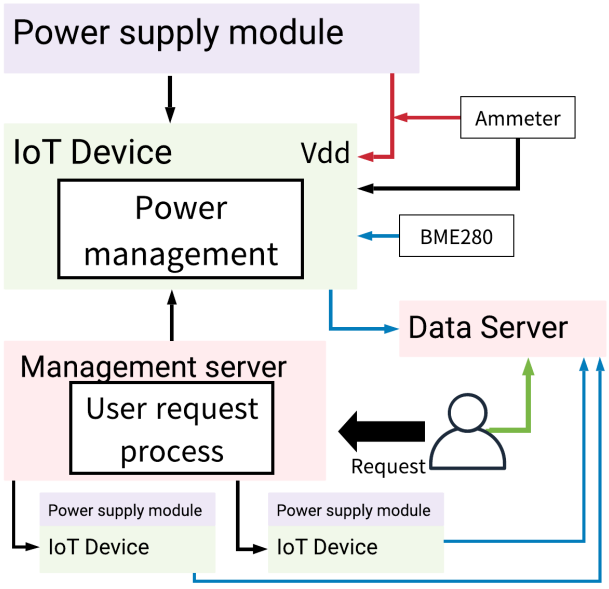

図 2　電力管理とユーザリクエスト処理のアーキテクチャ

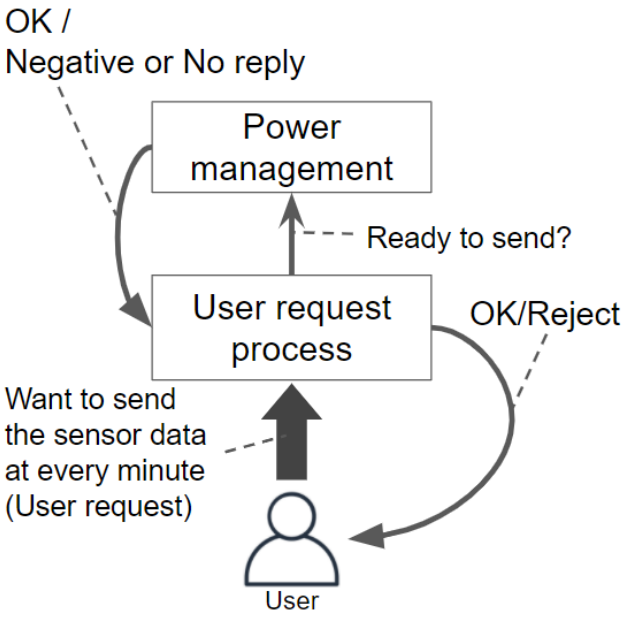

図 3　ユーザリクエスト処理

the sensor data at every minute」というものである．その処理や設定を行うのがこの User request process である．まずユーザからのリクエストを受け取ると，User request process から IoT デバイスへデータの送信指示をする．送信指示と合わせて，次の送信タイミングまでスリープをする時間の送信を行うことでスリープに入ったとしてもサーバ側が定めた時間にウェイクアップをすることができる．また，User request process は複数の Power management とのやり取りを行うため実際の構成としては図 4 のようになる．

IoT デバイスからの応答 (Response) は User request process から送信指示を受け取った際にデータ送信の可否を示しているものでいわゆる ACK のようなものである．また

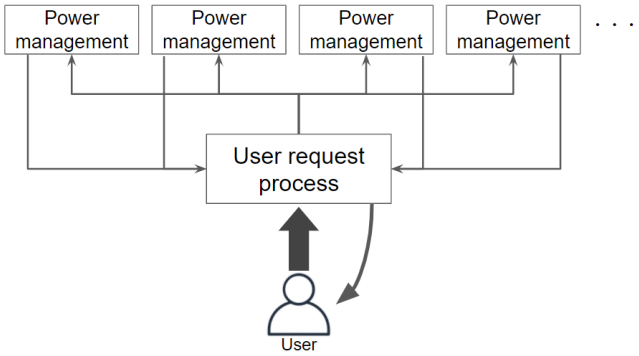

図 4　複数の Power management

同時に IoT デバイスのエネルギーリソースデータ (発電量，2 次電池の残量) の送信も行い，サーバ側でもデバイスの状態を把握する．そしてデータ送信が OK であれば IoT デバイスからセンサーから得られたデータをユーザが閲覧できるデータサーバへ送信する．User request process はユーザに対して，データサーバにセンサーデータの送信が行えたならば送信完了の通知を送信する．図 5 は送信が行えた場合の処理のやり取りを示す．今回はセンサーデータを 1 分毎に送信するというものをユーザのリクエストとして説明を進めていく．

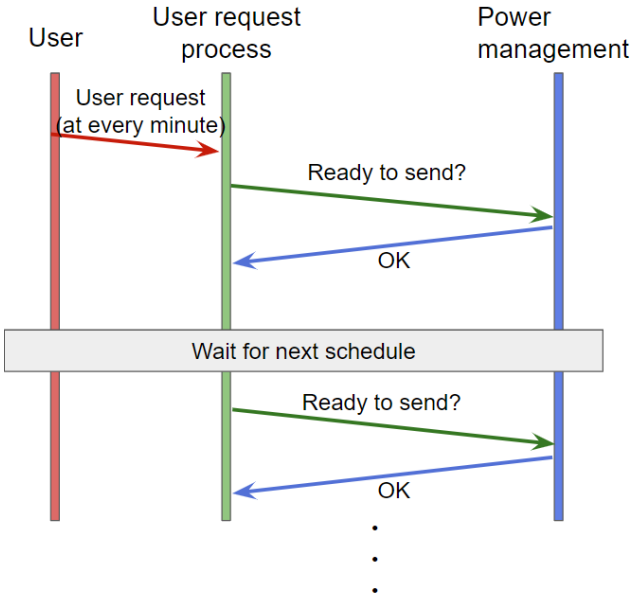

図 5　送信可能な場合の処理

図 6 では IoT デバイスからバッテリー残量が足りず送信ができない場合 (Reject) や応答がそもそもないケースを示しており，なおかつそのデータ送信不可についてユーザがデバイスからの応答があるまで待機しているのを選択した際の処理方法を表している．ユーザが待機をした場合，User request process から IoT デバイスへ 1 秒に 1 度の間隔で再送やリブートの実行コマンドを送信する．1 秒に 1 度としているのは，IoT デバイスから応答がないというのは電池の残量が足りず起動電圧に達していない場合である

ため, いずれ発電により再度起動するため短い頻度にて再送する．IoT デバイスから送信 OK が返ってくるまで実行を行う．

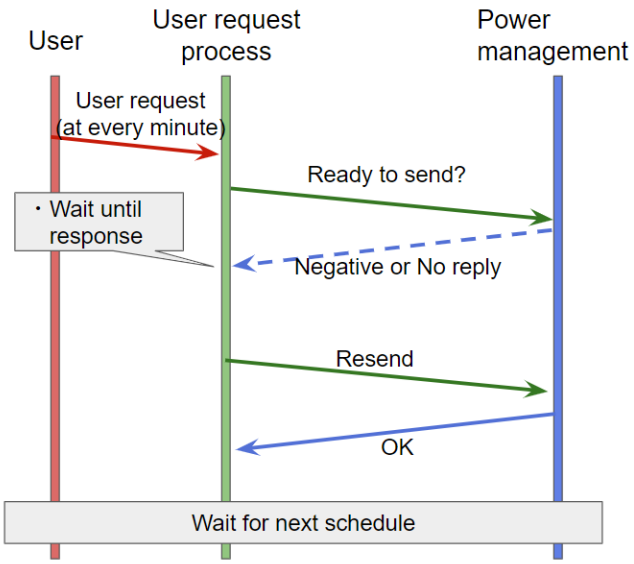

図 6　送信不可 (ユーザが待機する場合)

図 7 は, ケースとして図 5 と変わらないがユーザは応答があるまで待機ではなく, データの再送を取りやめて元々予定されていた次のスケジュールを待つという事をしている．つまり時間通りにデータが送れなかった場合は”データの送信が行われなかった”として処理されるという意味になる．

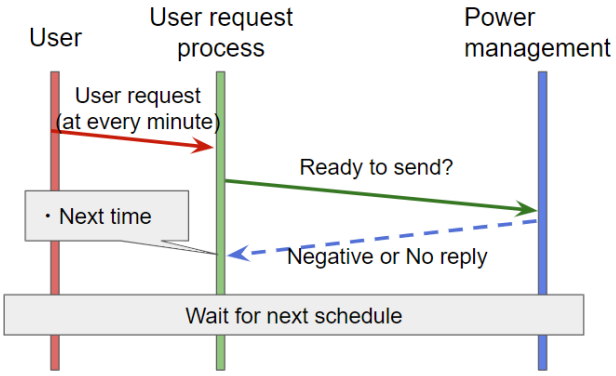

図 7　送信不可 (ユーザが再送を取り止める場合)

この No Reply や Negative という応答に対するユーザの対応というのは, その都度確認を取るのではなく, ユーザと User request process にて事前に「送信不可の場合にどのようにするか」を定めているものである．そのため Wait until response であれば IoT デバイスから応答が来るまで待機しておき, Next time であればその送信は見送りをし次のスケジュールまで待つといったようになる．またこのユーザとの対応方法について上記以外にも考えられるように柔軟性を持たせている．データ送信は行わず, センサーデータの取得のみ行うことも可能でありユーザのシステムにより影響が出ないようにするために合意を測る必要性がある．

先述した様に, User request process ではデバイスの管理を行っていると共に, ユーザと IoT デバイスとの橋渡しのような役割を持っている．これらのケースはユーザと IoT デバイスの関係が 1 対 1 のみではなく, 1 対多, 多対 1, 多対多の時でも同様な処理を行う．

3.2　Power management

図 7 は Power management への入力を表した図となっている．Power management では発電量と 2 次電池の残量を入力として受け取る．

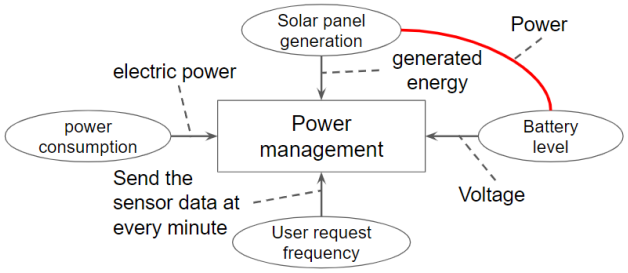

図 8　電力管理図

電力管理として太陽光パネルからの発電量を監視し, それに合わせた対応をすることが重要なポイントとなる．まず太陽光パネルで生成された電力は一度 2 次電池へ貯められ IoT デバイスへと送られる．監視する電力は太陽光パネルでの発電量と 2 次電池から供給される電源の電力量である．電力の監視は, IoT デバイスがアクティブでなければならずまた電力を測定するためも電力は消費されるため, 常時監視することは著しく消費電力を増加させてしまうため望ましくない．

消費電力は①通信を行っていない状態，②通信を行っている状態，③スリープの状態の 3 つに分類して求めだす．

得られた計算結果から消費電力が把握できるようになると, そのデバイスの残りの動作可能時間の計算が可能になる．ただしあくまでもこれは将来予測ということではなく, 現在のデータに基づいて算出するため時間は逐次変化し, 直線的なグラフとして示される．そのため現在の時間より先の事象になればなるほど正確性は欠けてしまう．よって, 6 時間先の未来を対象として, その間にデバイスが動作を停止してしまう可能性を考慮することとする．最長 6 時間としているのは, 万が一 6 時間で動作が停止してしまう可能性が高い場合となった際, 太陽光パネルからの発電量が急激に増大するわけではないためある程度長いスパンで対処を行う必要がある．そのため 6 時間前から対処動作を行うことで生存時間の延長や, 電力管理が余裕を持ち行えるようになる．

4. 実装と評価

4.1 実装

この章では具体的な実装方法について述べる．まず図 9 にソフトウェア構成図を示す．

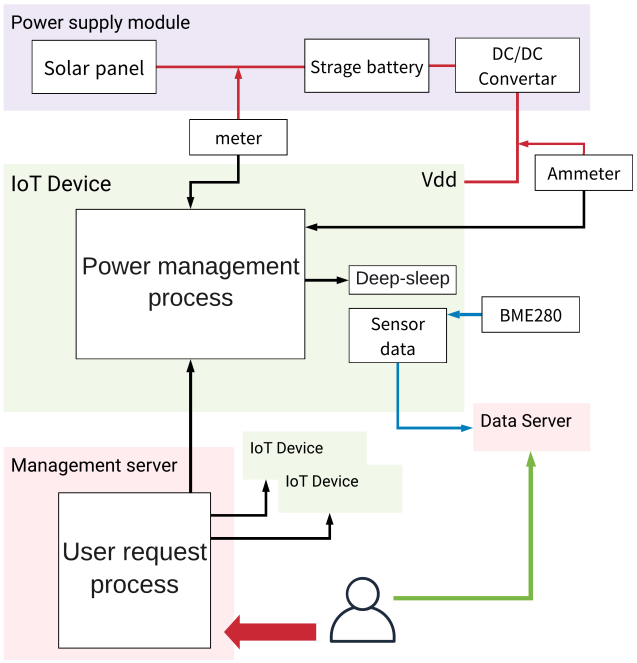

図 9　ソフトウェア構成図

このソフトウェア構成図は図 2 を拡張したものであり，より詳細に示されている．Power supply module では，赤い線が回路 (導線) を示しており太陽光パネル (solar panel) から発電された電力は 2 次電池 (Storage battery) へ一度貯められ，DC/DC コンバータにて電源電圧を IoT デバイスの規格電圧へと変圧し Vdd へ接続されるようになっている．この回路には電流計が太陽光パネルの発電量と 2 次電池からの電源電圧を測るために設置されている．今回，IoT デバイスとして用いるのは ESP32 というマイクロコントローラであり，電源電圧は 3.3V であるため DC/DC コンバータの出力はこの数値に合わせる．また DC/DC コンバータは 2 次電池の出力電力を入力とするが，2 次電池はバッテリー残量により出力電力が変化する．電池が満タンの際は定格電圧で動作を続けるが，残量が低下していくと共に次第に出力電力も下がっていく性質があるため今回は電池残量がある程度減ったとしても動作電圧に余裕を持たせるべく 2 次電池には定格電圧 5.0V のものを利用することとする．

2 次電池には電気二重層キャパシタを利用する．電気二重層キャパシタを利用する利点として，充放電を繰り返しても劣化しにくいという点である [8]．種類にもよるが他の 2 次電池では 200 回〜1000 回ほど利用すると容量は半分程度になるがそのようなことは発生しない．よって 2 次電池のメンテナンスという面においても長期間メンテナンスフリーで動作可能であり，長期間の動作に大きく貢献できる．

今回実装で用いる IoT デバイス，ESP32 には能力の高いスリープ機能が搭載されている．それが Deep-sleep である．Deep-sleep は CPU モジュールや発振器，ペリフェラルの一部といったマイクロコントローラの大部分への電力供給を止め，タイマーや外部割込みを動作されるためだけに動作してる低電力で稼働するモードである．公式のデータシートによると消費電流は 5 μ A とされている．本研究での太陽光発電を用いて駆動させるためにはこの機能は安定した動作と長期間の運用のために重要な役割を担うこととなる．基本的に CPU での処理やネットワークとの通信を行っている状態でない場合はデバイスをスリープにさせ，必要な際にスリープからウェイクアップをして処理を行う．またスリープに入る前には予めスリープ時間を定めておく必要がある．この処理に関してはサーバからデバイスに対して時間を与えるようにすることにする．これにより，サーバとデバイスとでスリープ時間を事前に共有することが可能であり次のウェイクアップタイミングをサーバ側でも把握することができる．

図 10 は Management server と IoT デバイスのシステム構成を示した図である．

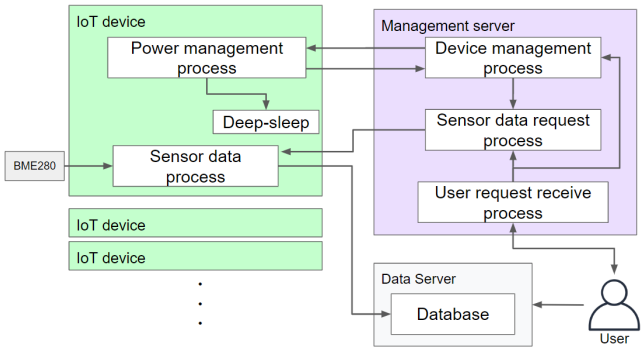

図 10　システム構成図

図 10 に示されているそれぞれのシステムの役割の説明は以下に示す．

- User request process
  ユーザからのリクエストの受け取りを行う．ユーザが設定するためのウェブページもここに設定をする．
- Sensor data request process
  User request process からユーザのリクエストに基づいて，データの送信を定めた期間で行うようにする．しかしデータ送信は Device management process が IoT デバイスに対してデータ送信が行えるかの確認を取ったうえで動作するものとする．
- Device management process
  User request process からユーザのリクエストに基づいて，IoT デバイスとの電力に関するデータのやり取りやデータ送信ができるかの確認を行う．IoT デバイスの Power management process へデータの送信ができ

るかの確認を行い, OK であれば Sensor data request process へデータを送信する旨の指示を出す.

- Power management process
  図 2 における Power management の処理を行うシステムである．システムの内容は先述した通り, 電力管理を行うことであり太陽光発電や消費電力, ユーザのリクエストに応じて IoT デバイスの動作やスリープの処理を行うのが主な役割である.
- Sensor data process
  これはサーバ側の Sensor data request process からのセンサーデータの送信依頼に応じてセンサーからデータを取得して Data Server へ送る役割を持っている．今回は BME280 という温度/湿度/気圧のデータを取得できるセンサーからデータを取得するものとする．(ビニールハウスでのデータ取得を考えているため)
- Database(Data Server)
  ここにはユーザが閲覧することを目的としているデータベースのことを指している．MySQL にて構築する.

4.2 評価

評価項目を以下に示す.

(1) 既存の IoT デバイスとの比較
既存の 1 次電池や 2 次電池で動作するものや, 関連研究で述べた IoT デバイスとの動作時間や稼働率を比較する．また, 関連研究にて述べた論文のシステムとの比較を行うことで本研究の優位性を確かに示すものとする.

(2) IoT デバイスの非稼働率
2 次電池の残量が減り, IoT デバイスの動作が停止していた時間の割合を示している．この数値が低ければ低いほど電力管理ができているという指標になる.

(3) ユーザのリクエストの達成率
ユーザと予め定めた, センサーデータを送信するというリクエストに対して, どの割合で達成できているのかを評価する．この数値が 100 へ近いほどユーザの要求は満たしている状態になる．またこのユーザーリクエストの達成率と IoT デバイスの非稼働率に相関があるか否かについても評価を行う.

5. 議論

本研究では, 太陽光発電と 2 次電池で動作する IoT デバイスにおける電力管理手法と電力リソースが足りずセンサーデータの送信が行えない場合やデバイスが停止している場合のユーザへの対処手法について提案した．しかし, 課題もまだいくつか残されている.

まず電力管理についてである．現状, 太陽光パネルの面積や 2 次電池の容量や種類を考慮していない．そのために太陽光パネルの面積による発電量の増減率や効率の変化に応じた電力管理ができていないという課題がある．太陽光パネルの面積を考慮することで, 設置する太陽光パネルの面積が最も効率よくなるように設定することが可能であり, より小さい IoT デバイスの作成に貢献できると考える．また 2 次電池の容量や種類については, 例えば電気二重層キャパシタとリチウムイオン電池とでは充放電によるバッテリー劣化や充電時間, 放電量が全く異なる．しかし, これを考慮していないためシステムは電池残量からの電圧の数値のみでしか判断することができないため長期的な予測やバッテリーの特徴に応じた対処をすることができない．今後は上記のパラメータもシステムの設定の中に入れることでさらに効率を上げた電力管理を検討する.

次に, ユーザとの合意手法についてである．IoT デバイスからエラーや応答がない場合, 予めユーザと合意した設定に基づいて対処方法を定めているが, 管理する IoT デバイスの数が数百台や数千台となった場合, 1 台 1 台に対する通知をユーザへ行った場合, ユーザは膨大な通知を受け取ることとなりむしろシステムの効率を低下させることに繋がりかねない．また IoT デバイスにセンサーが複数台設置されている場合も同様である．原則としてユーザが初期パラメータを設定した後がシステムが自動的に動作することが望ましいため今後はその観点からも検討を行っていく.

6. おわりに

本研究では, 外部電源から電力供給が受けられない IoT デバイスにおける電力問題の解決のため太陽光発電と 2 次電池を用いた電力管理手法を提案した．またその手法に伴うセンサーデータの送信ができない場合におけるユーザへの対処手法についても併せて提案した．しかし実際にユーザにシステムを使ってもらわなければ分からないユーザとのやり取り手法についてや電力管理はまだ検証が不十分であり, 裏付けが必要な点が複数存在する．今後の課題として実際にシステムを稼働させ, 得られた数値からさらに効率の良い手法を検証できないか検討していく.

参考文献

[1] Chi, Q., Yan, H., Zhang, C., Pang, Z. and Xu, L. D.: A Reconfigurable Smart Sensor Interface for Industrial WSN in IoT Environment, *IEEE Transactions on Industrial Informatics*, Vol. 10, No. 2, pp. 1417–1425 (online), DOI: 10.1109/TII.2014.2306798 (2014).

[2] Chu, M., Li, H., Liao, X. and Cui, S.: Reinforcement Learning-Based Multiaccess Control and Battery Prediction With Energy Harvesting in IoT Systems, *IEEE Internet of Things Journal*, Vol. 6, No. 2, pp. 2009–2020 (2019).

[3] Kraemer, F. A., Ammar, D., Braten, A. E., Tamkittikhun, N. and Palma, D.: Solar Energy Prediction for Constrained IoT Nodes Based on Public Weather Forecasts, *Proceedings of the Seventh International Conference on the Internet of Things*, New York, NY, USA, Association for Computing Machinery, (online), DOI:

10.1145/3131542.3131544 (2017).
[4] Jackson, N., Adkins, J. and Dutta, P.: Reconsidering Batteries in Energy Harvesting Sensing, (online), available from ⟨https://doi.org/10.1145/3279755.3279757⟩ (2018).
[5] Chu, M., Li, H., Liao, X. and Cui, S.: Reinforcement Learning-Based Multiaccess Control and Battery Prediction With Energy Harvesting in IoT Systems, *IEEE Internet of Things Journal*, Vol. 6, No. 2, pp. 2009–2020 (2019).
[6] Kul, B. and Şen, M.: Energy saving IoT-based advanced load limiter, *2017 XXVI International Scientific Conference Electronics (ET)*, pp. 1–5 (2017).
[7] Gummeson, J.: Energy Harvesting is Charging Up, Vol. 22, No. 4 (online), DOI: 10.1145/3325867.3325877 (2019).
[8] 岡村廸夫：電気二重層キャパシタを用いた蓄電装置，電気学会誌， Vol. 120, No. 10, pp. 610–613（オンライン），DOI: 10.1541/ieejjournal.120.610 (2000).

