修士論文

題目

IoT **機器からの通知に基づいた機器監視サービスの開発**

学籍番号・氏名

15006・宮坂 虹槻

指導教員

横山 輝明

提出日

2017**年** 1**月** 28**日**

神 戸 情 報 大 学 院 大 学
情報技術研究科　　情報システム専攻

目 次

図 目 次

内容梗概

近年, 半導体技術の進歩により, コンピューターの小型化・低価格化が進んでいる。また, インターネット回線網の普及もあり、Internet of Things という概念が注目され、それによって収益を得る IoT サービスが登場してきた。Internet of Things(IoT) とは、様々な物がインターネットにつながり、相互に情報を交換し合うことで、様々な自動化を実現する概念である。

しかし、IoT サービスを開発・運用するには、開発コストの問題・セキュリティーの問題・稼働率の問題など様々な問題がある。

そこで、本研究では、IoT デバイスの死活監視問題に焦点を当て、IoT サービスとは独立した IoT デバイスの監視サービスを開発することにより、デバイスの故障検知に係る問題の解決を図ることにした。システムの構築に先立って、どのような機能が必要となるのか、実験し、デバイスの電源の状態 (電源が入っているのか・入っていないのか)・ネットワークの状態 (インターネットへ接続されているのかいないのか) が時系列に沿って整理されている事で、対処が決まる事が分かった。そこで、上記必要な機能を実装したシステムを提供し、協力者の理解を得て検証し評価を得た。

第1章　はじめに

近年、IoT が注目を集めている。IoT とは、コンピュータをさまざまなモノに取り付けることで、利便性の向上を図る概念である。近年の半導体技術の進歩により、コンピュータが安価・小型になったこと、インターネットへの通信が様々な場所で安価に行えるようになったことにより、注目が集まっている。

それらのモノが連携して提供するサービスは IoT サービスと呼ばれ、より生活に身近なサービスの登場が期待されている。IoT サービスは、IoT 機器とサーバーがインターネットを介して通信し合うことで、成り立っている。IoT 機器は、モノにコンピュータが取り付けられた物で、周囲の状況を検知、または、周囲へ働きかける機能を持つ。サーバーは、IoT 機器からの情報を蓄積・分析し、IoT 機器へ指示を送るか、ユーザーへ分析結果を表示する機能を持つこれら IoT 機器とサーバーが連携することで、IoT サービスは利便性をユーザーへ提供している。

IoT サービスを円滑に提供するには、IoT 機器とサーバーの連携を正常に維持しなければならない。そのため、IoT 機器の動作状態や通信状態の監視が重要となる。数も多く、さまざまなネットワークを介して接続される IoT 機器の監視は困難な問題である。IoT 機器が設置される様々なネットワークの構成を把握することは、IoT 機器が多量であることを考えると現実的ではない。多量の IoT 機器を個々に識別し、異常を検知することも難しい。そのため、設置されるネットワークに関係なく状態が監視できることが求められる。また、IoT 機器の状態を一覧して確認できることや、IoT 機器の過去の動作状態や通信状態を確認できることが必要である。

そこで、我々は、IoT 機器からの通知に基づいた機器監視サービスを提案する。IoT 機器が自身の過去の動作状態や通信状態を記録し、機器監視サービスがそれら記録を可視化する。この仕組みを用いることで、IoT 機器が設置されるネットワークに関係無く状態を監視することや、IoT 機器の過去の状態や通信状態を確認することを容易にする。本研究では、IoT 機器からの通知による機器の設置環境によらない機器の監視を行うことにより、IoT サービスの維持を容易にするシステムの開発に取り組む。

本論文では、IoT 機器の監視困難の問題を取り上げ、その問題解決のための監視サービスを開発し、効果を報告する。第 2 章では、IoT サービスの維持に関する背景と、IoT 機器の監視に関する問題を述べる。第 3 章では、第 2 章で述べた問題を分析し、IoT 機器監視サービスの機能要件について述べる。第 4 章では、IoT 機器監視サービスの実装の詳細について述べる。第 5 章では、実験により IoT 機器監視サービスがもたらす効果を検証し、考察を述べる。第 6 章では、本研究に関する評価について述べる。第 7 章では、本研究を通して得られた知見や今後の課題について述べる。

第2章　IoTサービスの維持における問題

2.1　IoTサービスとは

IoTとは、Internet of Thingsの略で「モノのインターネット」とも呼ばれる概念である。IoTでは、様々な物がインターネットにつながり、相互に情報をやり取りすることで、多様な自動化を行う。IoTサービスとは、ユーザーに対しIoTによる利便性を提供するものである。

IoTサービスは、半導体技術の進歩によりコンピューターが小型且つ安価になったこと、通信ネットワークの整備が進み様々な場所から安価に通信が利用可能になったことで登場した。

例えば、次のような物がある。

- コーヒーポット
- 太陽光発電の監視

コーヒーポットの自動化とは、コーヒーポットまで移動し、コーヒーポットの状態を確認しに行く手間を省くためのサービスである。このサービスの実現の為に、コーヒーポットに対してコンピューターを取り付ける。これらコンピューターが、コーヒー豆の残量やコーヒーが入ったか否かの情報をサーバーとやり取りする。それにより、サーバーはコーヒー豆の自動発注や、コーヒーが入ったことユーザーに知らせる。

太陽光発電の監視とは、太陽光発電所の発電量や機器の異常を確認しに行くための手間を省くためのサービスである。このサービスの実現の為に、太陽光発電所の機器にコンピューターを取り付ける。これらコンピュータが発電量や機器の異常の情報をサーバーとやり取りする。それにより、サーバーは、発電量や機器の異常をユーザーに知らせる。

このように、IoTサービスは、IoT機器とサーバーが連携し、ユーザーに利便性を提供するものである。今後も数多くサービスが登場すると考えられている。

2.2　IoTサービスの構造とは

IoTサービスは、IoT機器とサーバーが連携し利便性を提供するものである。IoTサービスの構造として、多数のIoT機器とサーバーがインターネットを介し連携する事が挙げられる。

IoT機器は、様々な環境へ設置され、周囲の状況を検知することや、周囲へなんらかの働きを行う為に使用される。コーヒーポットの例では、コーヒーが出来上がったか否かを検知する。また、コーヒーポットに対し、コーヒーを入れるよう働きかけている。

この機器からの情報を収集し、処理しているのがサーバーである。サーバーは、IoT機器からの情報を蓄積・分析し、IoT機器やユーザーに対し何らかの働きかけを行う。コーヒーポットの例では、ユーザーがコーヒーを入れた回数からコーヒー豆の残量を分析し発注を行うよう働きかけている。

サーバーにて動作するプログラムが、IoT機器と通信することで、IoTサービスを構成する。この通信に利用されるのがインターネットである。様々な通信リンクを用いてIoT機器とサーバー上のプログラムが連携する。

このように、IoTサービスの構造は、IoT機器とサーバー上のプログラムがインターネットを介し通信し、連携することで成り立っている。

2.3 IoT サービス維持の問題

IoT サービスは、IoT 機器とサーバー上のプログラムがインターネットを介し通信し合うことで成り立っている。IoT サービスを維持するためには、これらの構造を維持する必要がある。そのため、IoT 機器が正常に動作しているのか、通信が途切れていないか、監視する事が重要である。ところが、IoT 機器が多量に存在することや、IoT 機器が接続するネットワークが多様であることから、その監視には技術的困難がある。また、個別の IoT サービスに組み込まれた監視システム等を別として、一般的な監視サービスも存在しない。

IoT サービスでは、多量の IoT 機器を使用する。そのため、個々の IoT 機器を識別し、適切に管理することが困難である。

また、家庭内や屋外等、様々な環境下に置かれる IoT 機器が接続されるネットワークを予め予期することは困難を極める。接続先のネットワークでもプライベートアドレスを付与されるなどの他、IoT 機器とサーバーとの通信に制限がかかる場合もある。そのため、遠隔から機器の状態を確認することが難しい。

このように、IoT サービスの維持において、IoT 機器の動作や通信状態を監視することは重要であるが、その監視には技術的な困難がある。IoT サービスの円滑な提供や今後の発展の為には、技術的な困難を越えた IoT 機器の監視の実現が重要である。

第3章　IoT機器からの通知に基づく機器監視サービスの提案

3.1　IoTサービスにおけるIoT機器の監視の重要性

IoT機器の監視は、IoTサービスを停止させないために重要である。

IoT機器が正常に動作していない場合、分析を正確なものとするため、分析時に該当のIoT機器を除外するか、全てのIoT機器が動作している時間帯を選び分析を行う必要がある。そのため、IoT機器を監視し、IoT機器が正常に動作していない時間を記録しておく必要がある。

またIoT機器が正常に動作していない事が分かった場合、正確に分析できないため、サービスを提供することができなくなる。そのために、常にIoT機器を監視し、異常が有れば、復旧作業を行う必要がある。

このようにIoT機器の監視は、IoTサービスを維持する上で重要である。

3.2　IoT機器の監視とは

IoT機器の監視とは、動作の状態・通信の状態を確認することである。IoT機器の動作の状態・通信の状態を確認し、記録することは、IoTサービスを維持する上で重要である。

動作状態とは、少なくとも電源が入っているのかどうかで、通信状態とは、インターネットに接続され、サーバーと通信ができているのかどうかである。

このように、個々のIoT機器の動作状態、通信状態を確認し、記録することが、IoT機器の監視である。

3.3　IoT機器の監視に必要な機能とは

IoT機器の監視とは、個々のIoT機器の動作状態、通信状態を確認することである。そのため、次のような機能が必要となる。

- IoT機器の動作状態、通信状態を検知する機能
- IoT機器の動作状態、通信状態を記録する機能
- IoT機器の動作状態を一覧して表示する機能
- IoT機器の過去の状態を確認する機能

IoT機器の動作状態、通信状態を検知する機能とは、IoT機器の現在の状態を検知する機能である。

IoT機器の動作状態、通信状態を記録する機能とは、IoT機器の状態を時刻と共に記録する機能である。

IoT機器の動作状態を一覧して表示する機能とは、現在のIoT機器の動作状態を見ることができる機能である。あるIoT機器が動作しなくなった時に、動作していないことを明確に示す必要がある。

IoT機器の過去の状態を確認する機能とは、過去のIoT機器の動作状態と通信状態を確認することができる機能である。過去の動作状態や通信状態を整理し、いつ動作していて、いつ動作していなかったのか、各IoT機器ごとに示す必要がある。

このように、IoT機器の監視には、IoT機器の動作状態を一覧して表示する機能、IoT機器の過去の状態を確認する機能を持つ必要がある。

3.4 IoT 機器の監視に求められる要件とは

IoT 機器の監視には、上記機能が必要である他に、IoT 機器が設置されるネットワークにかかわらず監視ができることが求められる。何故ならば、IoT 機器が設置されるネットワーク環境は多様であるからである。

IoT 機器が設置されるネットワーク環境として、次のような環境が挙げられる。

- IoT 機器に対しプライベートアドレスが与えられる環境
- IoT 機器とサーバーの通信が制限される環境

多くの場合、IP アドレスの節約の観点から、IoT 機器に対し、プライベートアドレスのみ与えられる。この場合、インターネット側からは、IP アドレスを指定することが出来ないため、通信は常に IoT 機器から始まる必要がある。また、IoT 機器とサーバーとの通信がセキュリティの観点から制限される場合もある。HTTP 等一般的に広く使用される通信を除いてブロックされる事がある。

このように IoT 機器の監視には、IoT 機器から通信が始まることや、HTTP 等頻繁に利用される通信を利用しなければならないといった条件がある。

3.5 IoT 機器監視に求められる要件とは

これらを踏まえて、IoT 機器の監視には次のような機能要件が求められる。

- IoT 機器の動作状態を一覧して表示する機能
- IoT 機器の過去の状態を確認する機能

また、非機能要件として、次のような制約がある。

- プライベートアドレスが与えられたとしても動作する事
- インターネットの通信として一般に広く利用されている HTTP 等の通信を用いる事

第4章　機器監視サービスの実装

4.1　機器監視サービスの構成

第3章にて述べた要件に基づき、システムを構築した。システムは、エージェントプログラム、機器状態データベース、機器情報データベース、「かおりちゃん」というエージェントプログラム用インターフェースプログラム、Web アプリケーションサーバ、Web アプリケーションから成り立っている。エージェントプログラムと「かおりちゃん」、Web アプリケーションサーバと Web アプリケーションは、インターネットを介して通信しあう。

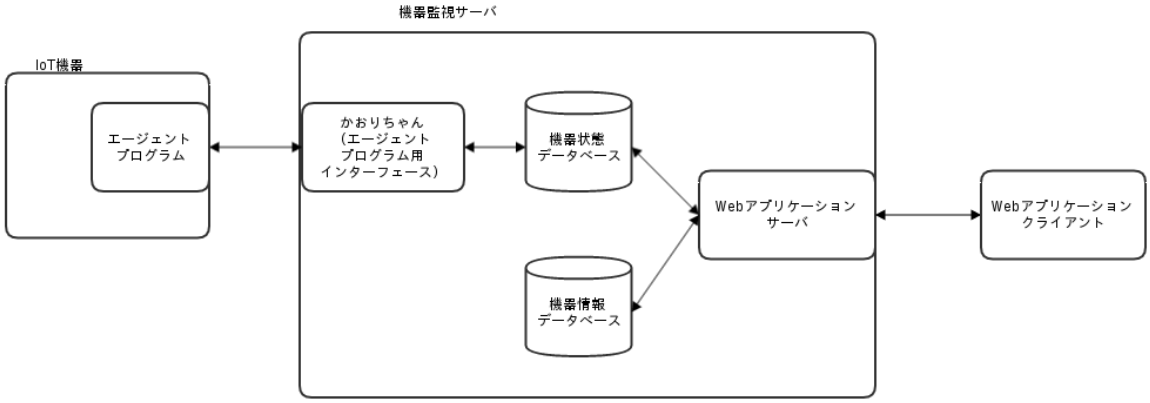

図 4.1: システムのブロック図

エージェントプログラムは、IoT 機器上で動作するプログラムである。定期的に自身の状態を状態蓄積システムへ送信する役割を果たす。定期的かつ自発的に状態を送信することで、ネットワーク環境によらない機器の監視を可能にした。

「かおりちゃん」とは、エージェントプログラム用インターフェースである。各 IoT 機器上で動くエージェントプログラムから送られてきた状態を、時刻と共に機器状態データベースへ書き込む役割を果たすプログラムで、機器監視サーバー上で動作する。

機器状態データベースとは、機器の状態を時系列に沿って蓄積するデータベースである。機器監視サーバー上で動作する。

機器情報データベースとは、機器 ID や、機器名、ユーザー名を記録するデータベースである。機器監視サーバー上で動作する。

Web アプリケーションとは、ユーザーからの入力を受けユーザーへ表示する他必要な情報を問い合わせる、ブラウザ上で動作するプログラムである。

Web アプリケーションサーバーは、Web ページや Web アプリケーション自体を配信する。Web アプリケーションからの要求に答え、現在の機器の状態や、機器名等を返答する。

これら各要素が連携することで、機器の監視を容易なものとした。

4.1.1 エージェントプログラム

エージェントプログラムとは、IoT 機器上にインストールされるプログラムである。エージェントプログラムの役割は、定期的に送信失敗回数を「かおりちゃん」へ報告することである。送信失敗回数とは、ネットワークの不具合等により、機器監視サーバーへ送信されなかった報告の数である。自発的に状態を報告するため、IoT 機器にプライベートアドレスが付与されていても、状態を検知することができる。また、HTTP を用いるため、間のネットワークにてブロックされることがない。

4.1.2 「かおりちゃん」

かおりちゃんとは、エージェントプログラム用インターフェースで、サーバー上で動くプログラムである。かおりちゃんの役割は、エージェントプログラムから送信されたメッセージを受け取り、現在の時刻と正常である旨を機器状態データベースへ書き込む。また、エージェントプログラムから送られた送信失敗回数から、IoT 機器がインターネットから切断された時刻を逆算し、機器状態データベースへ書き込む事も行う。

4.1.3 機器状態データベース

機器状態データベースとは、サーバ上で動作するデータベースである。各 IoT 機器の状態を時刻とともに記録する。機器状態監視システムの中心にあるデータベースである。

4.1.4 機器情報データベース

機器情報データベースとは、サーバー上で動作するデータベースである。各 IoT 機器の機器 ID、機器名、機器詳細情報、ユーザーのメールアドレスとパスワードを記録する。

4.1.5 Web アプリケーションサーバ

Web アプリケーションサーバとは、サーバ上で動作するプログラムである。Web アプリケーションや Web ページの配信、Web アプリケーションからの要求の処理などを行う。必要に応じて、機器状態データベースと機器情報データベースへアクセスを行う。次に、Web アプリケーションとのインターフェースを挙げ、それぞれについて説明する。

ログイン API

Web アプリケーションから、メールアドレスとパスワードを受け取り、 機器情報データベースのユーザーテーブルと照合する。照合した結果、ユーザー名とパスワードが合致したユーザーが存在すれば、HTTP クッキーにセッションキーをセットし、機器状態一覧ページへのリダイレクトを返す。合致したユーザーが存在しなかった場合、エラーメッセージを返す。

4.1.6 ログアウト API

Web アプリケーションに、HTTP クッキーから該当のセッションキーを削除するよう要求する。

機器情報・機器状態取得 API

ログインチェックを行った後、機器情報データベースと機器状態データベースより、全 IoT 機器の機器情報と機器状態を返す。

機器 ID 生成 API

ログインチェックを行った後、機器情報データベースに存在しない、ランダムな機器 ID を返す。

機器 ID 重複チェック API

ログインチェックを行った後。Web アプリケーションから機器 ID を受け取る。機器情報データベースに該当の機器 ID が存在するかしないかを返す。

機器作成 API

ログインチェックを行った後、Web アプリケーションから、機器 ID、機器名、機器の詳細と、機器の作成なのか編集なのかの指示を受け取る。機器の作成であった場合、機器 ID に重複が無いことを確認したうえで、機器情報データベースに該当のエントリを作成・機器状態データベースにメジャーメントを作成する。作成されたか、されなかったかを返す。機器の編集であった場合、機器情報データベースから、受け取った機器 ID を持つものを探しだし、編集する。編集できたか否かを返す。

機器削除 API

ログインチェックを行った後、Web アプリケーションから、機器 ID を受け取る。機器情報データベースから、受け取った機器 ID を持つものを削除する。その後、機器状態データベースから、メジャーメントを削除する。削除されたか否かを返す。

過去の機器状態取得 API

ログインチェックを行った後、機器状態データベースから、全ての IoT 機器の過去の状態をまとめ、返す。

4.1.7 Web アプリケーション

Web アプリケーションとは、ユーザーのブラウザ上で動作するアプリケーションである。ユーザーへグラフィカルインターフェースを提供する。必要に応じて、Web アプリケーションサーバへ必要な情報を要求する。Web アプリケーションは、3 つの Web ページから成り立っている。以下に各ページとそれぞれの役割を述べる。

- ログインページ
  正当なユーザーであることを確認するために、メールアドレスとパスワードの入力を求める。メールアドレスとパスワードは、Web アプリケーションサーバーへ送信される。
- 機器状態一覧ページ
  機器状態を一覧して表示する他、機器の作成や、機器情報の編集、機器の削除等を行う事ができる。現在の機器の状態を取得するため、定期的に Web アプリケーションサーバと通信する。また、機器の作成や機器情報の編集、削除の為、Web アプリケーションサーバと通信する。
- 過去の機器状態一覧ページ
  過去の機器状態を時刻と共に整理し、一覧表示するページである。現在の機器の状態を取得するため、定期的に Web アプリケーションサーバと通信をする。

4.2 エージェントプログラムの実装

エージェントプログラムの役割は、送信失敗回数を定期的に IoT 機器監視サーバーに送信することにある。約 1 分おきに、送信失敗回数を、「かおりちゃん」へ送信する。どのような Linux 環境でも動作することを考え、Shell スクリプトにて実装した。

4.3 「かおりちゃん」の実装

「かおりちゃん」の役割は、エージェントプログラムから送信失敗回数を受け取り、時刻と正常である旨を機器状態データベースへ書き込むこと、また、エージェントプログラムから送られた送信失敗回数から、インターネットより切断された時刻を逆算し、機器状態データベースへ書き込む事である。Falcon と呼ばれる API の作成に特化したフレームワークを使用し、Python にて実装した。また、機器状態データベースへの書き込みには、InfluxDBClient というライブラリを使用した。

4.4 機器情報データベース

機器情報データベースの役割は、機器の名前、機器の詳細説明、機器 ID、ログイン用ユーザー ID とパスワードを記録し保持する事である。機器情報データベースには、SQLitei3 を用いた。機器情報データベースには、次のようなテーブルが用意されている。

4.4.1 機器情報テーブル

機器 ID、ユーザー ID をキーとして、機器の名前、機器の詳細説明を記録し保持するテーブルである。

4.4.2 ユーザーテーブル

ユーザー ID をキーとして、ユーザー名とパスワードを記録し保持するテーブルである。

4.5 機器状態データベース

機器状態データベースの役割は、機器の状態を時刻と共に記録・保持することである。機器状態データベースには、Influxdb を用いた。機器 ID をメトリクス名（テーブル名）とし、時刻をキーとして、機器の状態を記録している。

4.6 Web サーバーアプリケーション

Web サーバーアプリケーションの役割は、与えられた HTTP リクエストを元に、Web ページや、各種情報を返却することにある。Flask と呼ばれる Web アプリケーションフレームワークを用いた。Python を使用している。Web サーバーアプリケーションの設計と Web アプリケーションの設計から、下記の様に実装した。先頭に付いている GET や POST は HTTP メソッド、/login 等は URL を示している。

GET /login
　　ログインページを返す。

POST /login
メールアドレスとパスワードを受け取る。クッキーからセッションキーを探し、存在すれば既にログインしているとして、/dashboard へのリダイレクトメッセージを返す。データベースにメールアドレスとパスワードの組が存在するか確認し、存在した場合、セッションキーを返す。存在しなかった場合、ログインエラーページを返す。

GET /logout
セッションキーを受け取り、該当のセッションを削除する。

GET /dashboard
ログインチェックをし、機器状態一覧ページを返す。

GET /devicelog
ログインチェックをし、過去の機器状態一覧ページを返す。

GET /api/device/all
ログインチェックをし、現在の状態、最後にメッセージを受け取った日時、機器 ID、機器名、機器詳細のデータを、JSON 形式にまとめたものを返す。

GET /api/deviceID
ログインチェックをし、ランダムにデバイス ID を生成し、JSON 形式にまとめたものを返す。

POST /api/deviceID
ログインチェックをし、受け取ったデバイス ID が既に存在するか確認し、その結果を JSON 形式にまとめ、返す。

POST /api/device/{DeviceID}
ログインチェックをし、デバイス ID と受け取った JSON データから、デバイスの新規作成・編集をし、結果を JSON 形式にまとめ、返す。

DELETE /api/device/{DeviceID}
ログインチェックをし、該当のデバイス ID を持つデバイスをデータベースから削除する。

4.7 Web アプリケーション

Web アプリケーションの役割は、ユーザーインターフェースを提供することである。Bootstrap,JQuery というライブラリを用いて作成した。HTML,CSS,Javascript で書かれている。定期的に Web アプリケーションサーバから状態を取得し、HTML,CSS を用いて表示する。

4.8 エージェントプログラムと「かおりちゃん」間の通信の実装

エージェントプログラムと「かおりちゃん」は、インターネットを介して、HTTP というプロトコルを用いて通信する。エージェントプログラムと「かおりちゃん」間の通信は次の様な形になる。

1. エージェントプログラムが「かおりちゃん」に対し、送信失敗回数を報告する。
2. 「かおりちゃん」は、時刻と正常である旨を機器状態データベースへ送信する。
3. 機器状態データベースより、正常に書き込んだというメッセージが帰ってくる。
4. 「かおりちゃん」は、必要に応じて送信失敗回数からインターネットから切断された時刻を逆算し、その時刻と切断されていた旨を機器状態データベースへ送信する。
5. 機器状態データベースより、正常に書き込んだというメッセージが帰ってくる。

6. 「かおりちゃん」は、エージェントプログラムに対し、正常に受け付けたというメッセージを返す。

エージェントプログラムと「かおりちゃん」の間の通信は、約 1 分おきに繰り返される。HTTP 上でやり取り

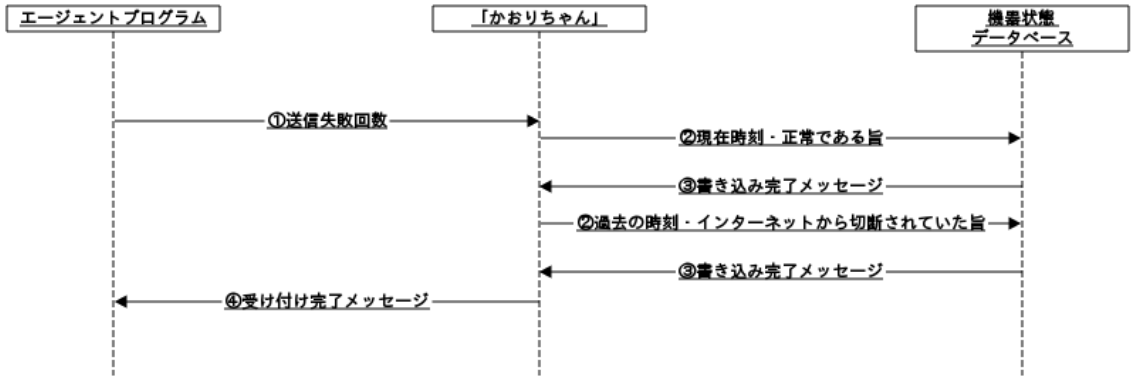

図 4.2: エージェントプログラムと「かおりちゃん」の間のメッセージシーケンス図

するデータの形式としては、Javascript Object Notation(JSON) という形式を用いる。

4.9 Web アプリケーションサーバと Web アプリケーション間の通信の実装

Web アプリケーションサーバと Web アプリケーションは、インターネットを介し、HTTP というプロトコルを用いて通信する。HTTP 上でやり取りするデータの形式としては、Javascript Object Notation(JSON) という形式を用いる。

第5章　機器監視サービスのユーザーテストと考察

作成したシステムの使用感・使い勝手を検証するために、以下のような実験を行った。

5.1　実験目的

RaspberryPi の監視をすることを通して、ユーザーインターフェースが適切か、要件に漏れがないかを検証する。

5.2　実験方法

1. RaspberryPi を 2 個、機器監視サービスに登録する。
2. RaspberryPi の電源を抜き差しし、その様子を機器監視サービスで観測する。
3. RaspberryPi に刺さっている LAN ケーブルを抜き差しし、その様子を機器監視サービスで観測する。

5.3　結果

5.4　考察

第6章　おわりに

本研究の背景には、IoT サービスにおける機器監視の煩わしさが挙げられる。具体的には、

1. IoT サービスに組み込む形で機器監視を行った場合、IoT サービス毎に監視部分を開発するので、コストが高い
2. 設置箇所が遠く、分散して設置されるため、定期的に確認しに行くことは現実的ではない

が挙げられる。

そこで、機器監視をサービスから独立させ、IoT 機器監視サービスとして提供すれば良いのではないかと考え、開発を行った。しかし、既存手法を調査する中で、IoT 機器は NAPT 環境下に置かれる事があることがわかり、IoT 機器から定期的に通知を送ることで解決を図った。
プロトタイプの開発の結果、下記のような知見を得ることが出来た。

- NAPT 環境下にある機器の状態を監視するために、機器から状態を定期的に通知する手法は有効であることが分かった。
- また、既存の機器監視サービスの多くは、機器の IP アドレスをサーバー側で管理しなければならないが、本手法を用いることで、IP アドレスに関係なく監視できることが分かった。

また、今後の課題としては、次のような事があることが分かった。

- ユーザビリティーの向上
  現状では、機器 ID やサーバー IP アドレスを IoT 機器に打ち込まなければならず、また、自動起動の設定も行わなければならない。
  シェリスクリプトを実行すれば自動起動の設定まで行われる等、いっそうの簡略化を行う必要がある。
- 機器の識別問題の解決
  現状では、どの機器が何だったのかまでは管理できていない。
  QR コードを出力し、機器に貼り付けるなどによって、簡略化できると思われる。
- アラート機能の実装
  アラートメールの送信などもできれば、常に監視していなくて良くなるのではないかと思われる。

これらの課題を解決することで、IoT サービスの開発の手間を省けるのではないかと考える。

第7章　謝辞

参考文献

[1] 平成 27 年版 情報通信白書 (総務省) 「IoT の実現に向けたアプローチと我が国 ICT 産業の方向性」より http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h27/html/nc254150.html

[2] 6 割が「IoT は流行語」–エスキュービズム調査 (ZDBet Japan) http://japan.zdnet.com/article/35093272/

[3] 先進テクノロジのハイプ・サイクル２０１１年　Gartner　https://www.gartner.co.jp/press/html/pr20110907-01.html