Vol.2016-ARC-219 No.35
Vol.2016-SLDM-175 No.35
Vol.2016-EMB-40 No.35
2016/3/25

kintoneとArduinoを用いた 低コスト在庫管理IoTシステムの開発

奥村 潤[1,a)]　佐野 隼輔[1]　浦本 竜[1]　久米 純矢[2]　舘 伸幸[3]　山崎 進[1,b)]

概要：倉庫を所有する企業では，在庫管理業務は非常に煩雑な作業である．そのため大企業では，倉庫の在庫をセンサーで自動検知できるような在庫管理システムを導入しているところが多い．一方で中小企業では，センサーが高額なためセンサーを持たない在庫管理システムを利用している場合が多い．その場合は人手で在庫を入力しており，十分に業務効率を改善できていない．
そこで，我々は在庫を自動検知できる低コストな在庫管理システムを構築した．これはクラウドサービスkintoneとArduinoを用いたIoTを組み合わせることで，低コストで最小限の機能を実現するものである．本研究では，アジャイル型の開発を行うことで，要求と成果物のギャップと，ハードウェア部品と工数のコストを削減した．

キーワード：在庫管理，IoT，Arduino，アジャイル開発，クラウドコンピューティング

Development of a Low Cost Inventory Management System Using kintone and Arduino

Jun Okumura[1,a)]　Shunsuke Sano[1]　Ryu Uramoto[1]　Junya Kume[2]　Nobuyuki Tachi[3]
Susumu Yamazaki[1,b)]

Abstract: Inventory management is a very complicated work in a company that has some distribution warehouses. Thus, many large companies have introduced inventory management systems that can be auto-sensing by sensors in the inventory. However, in particular, small- or medium-sized enterprises use the inventory management systems without having any sensors in many cases because the sensors are expensive. Then, they can't improve efficiency sufficiently they record their the inventories by hand.
Thus, we have built a low cost inventory management system that can be auto-sensing in the inventories. It has the minimum auto-sensing functions at low cost by IoT combining with using kintone, which is a cloud service, and Arduino.
In this research, we have reduced the gaps between requirements and artifacts, and the cost of hardware parts and efforts, by applying the agile software development method.

Keywords: inventory management，IoT，Arduino，agile software development，cloud computing

1. はじめに

物を販売する企業では，必ず倉庫を持ち，商品の在庫を用意している．そのため，在庫の管理業務が発生する．しかし，在庫管理は人が行うと非常に手間がかかり，効率が悪い．そこで，業務効率を改善するため，大企業では倉庫の在庫をセンサーで自動検知できるような，在庫管理シス

[1] 北九州市立大学
University of Kitakyushu, Kitakyushu, Fukuoka, Japan
[2] 株式会社 AISIC
AISIC,Inc., Onojo, Fukuoka, Japan
[3] 名古屋大学
Nagoya University, Nagoya, Aichi, Japan
[a)] w5mcb003@eng.kitakyu-u.ac.jp
[b)] zacky@kitakyu-u.ac.jp

Vol.2016-ARC-219 No.35
Vol.2016-SLDM-175 No.35
Vol.2016-EMB-40 No.35
2016/3/25

テムを導入しているところが多い．一方で，多くの中小企業では，センサーが高額なためセンサーを持たない在庫管理システムを利用している場合が多い．その場合，人手で在庫を入力しており，十分に業務効率を改善できない．中小企業は，大企業と比べると人手も少ないため，業務効率の改善は企業競争力を高めるために，非常に重要である．

そこで，我々は，自動で在庫検知できる，低コストの在庫管理システムを構築することにより，中小企業の業務効率を改善できると考えた．

本稿では，市場で簡単に入手可能な安価な WiFi 通信用 Arduino マイコンボード，ESP-WROOM-02 と，変更が容易で開発コストを抑えることが可能であるクラウドサービス，kintone を使用し，在庫管理 IoT システムを開発した事例について述べる．

本論文は次の構成で説明する．第 2 章では本研究で使用する要素技術について説明する．続いて，第 3 章ではシステムの仕様，プロダクトの概要を示す．第 4 章では開発体制とそのプロセスについて説明し，第 5 章では開発で直面した課題とその解決について説明する．第 6 章では，開発の振り返りをし，第 7 章では，関連研究の紹介と比較をする．最後に第 8 章では，開発のまとめと将来課題を述べる．

2. 概念・要素技術

本システムでは，次のような概念・要素技術を用いる．

- IoT
- WiFi 通信用 Arduino マイコンボード ESP-WROOM-02
- kintone

本章では，これらについてそれぞれ述べる．

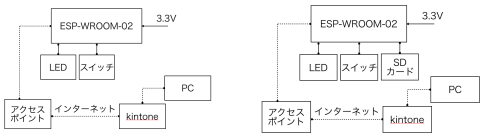
2.1 IoT

IoT とは Internet of Things（モノのインターネット）の略である．類似した名称で，Machine to Machine（M2M）というものがある．参考文献 [1] より，これは，人が介入せず，機械間コミュニーケーションを行うことを指す．この M2M という名称が，流行により IoT という名称に変遷した．

本研究では，ハードウェアに ESP-WROOM-02 を用いて，インターネットを経由し，クラウドサービスである kintone を使用した．以下，ESP-WROOM-02 と kintone について説明する．

2.2 WiFi 通信用 Arduino マイコンボード ESP-WROOM-02

ESP8266 [*1] を搭載した，技術基準適合証明済の WiFi 通信用モジュールである．ESP-WROOM-02 の WiFi 機能を使用するには Github にある ESP8266 core for Arduino をインストールする必要がある．また，参考文献 [2] によると，ESP-WROOM-02 には，SPI，UART，I2C，GPIO など様々なインターフェースが内蔵されている．さらに，TCP，UDP もサポートしている．



[1] 初期想定システム　　[2] 開発したシステム

図 1　システム構成概要図

Fig. 1　The system configuration

2.3 kintone

サイボウズが提供する PaaS 型クラウドサービスである．参考文献 [3] より，kintone は業務管理アプリの作成が容易で，マウス操作で直感的にフォームを作成，変更できる．また，Excel ファイルを読み込むだけで，データのインポートや入力フォームの生成ができる．

本研究では，ESP-WROOM-02 から送信されるデータを蓄積するデータベースとして利用する．

3. 仕様

本章では，開発するシステムの概要を示す．

3.1 システム構成

図 **1** に，システム構築を示す．研究・試作のため新たな顧客要求により，当初の計画 [1] から，最終的に [2] へと変遷した．詳しくは，4.2 節で述べる．

初期想定システムは，次のようになっていた．ESP-WROOM-02 には 3.3V の電源を供給する．また，GPIO インターフェースにはスイッチ，LED を接続し，任意のタイミングでスイッチを ON にすることで，通信を行うことができるようにする．通信の際はアクセスポイントを経由し，kintone に接続する．LED は，システムの稼働状況を示す．

一方，[2] に示す実際に開発したシステムでは，[1] の構成に加えて SD カードを追加し，ユーザーがオプションを指定することを可能にした．

3.2 プロダクト概要

本研究では，図 **2** のような使用状態を想定している．

計量器のように，台が重量によって上下する物に部品を乗せる．上下する部分と固定されている部分の間に，スイッチを挟み込むように設置する．在庫が減少した際にスイッチが ON になり，クラウドにデータを送信する．送信するデータはスイッチが ON になった時間とその部品名である．これらは，類似のシステムであれば代用が可能で

[*1] WiFi 通信用 SoC,32bitMCU

Vol.2016-ARC-219 No.35
Vol.2016-SLDM-175 No.35
Vol.2016-EMB-40 No.35
2016/3/25

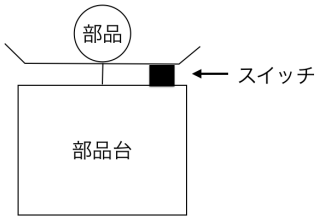
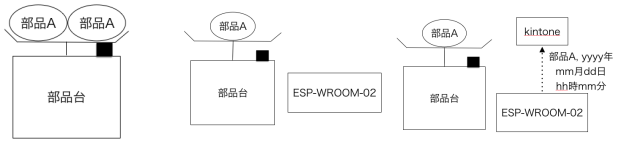


図 2　センサー設置状態例
Fig. 2　An example of a sensor installed statement



[1] 初期状態　[2] スイッチ ON　[3] データの送信

図 3　システムの流れ
Fig. 3　System flow

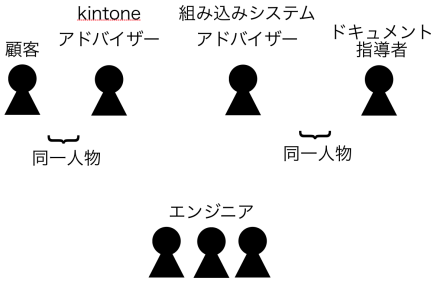


図 4　開発体制
Fig. 4　Development system in this project

ある．

システムの流れとしては，図 **3** のように動作する．在庫の部品が消費され，重量が軽くなると，スイッチが ON になる．スイッチが ON になると，ESP-WROOM-02 が kintone にデータを一度送信する．

4. 開発体制とそのプロセス

本章では，我々が行った開発体制とそのプロセスを説明する．

4.1 開発体制

本研究では，図 **4** のような体制で開発を行った．顧客，kintone アドバイザー，組込みシステム・アドバイザー，ドキュメント指導者，エンジニアが存在する．本研究では，顧客と kintone アドバイザーは同一人物であり，組込みシステム．アドバイザーとドキュメント指導者も同一人物で行った．また，エンジニアは 3 名で開発を行った．各エンジニアの開発内容については 4.2 節で説明する．

開発期間は，約 4 ヶ月で．基本 1 週間に 1 度顧客と組込みシステムアドバイザーそれぞれとのミーティングを行い，開発をした．

開発は，アジャイル型開発を参考に行い，1 週間を 1 イテレーションとして，1 週間での作業時間は約 4 時間とした．

表 1　開発前期の作業
Table 1　The work of 1H development.

	10 月	11 月
第 1 週	システムの提案	kintone 操作用 API 開発,
	—	試作機の回路図を作成
第 2 週	通信方式の提案	kintone 操作用 API 開発,
	—	ESP-WROOM-02 の配線
第 3 週	技術調査と試作	kintone 操作用 API 動作テスト
第 4 週	技術調査と試作	デバイスにスイッチを実装,
	—	データ送信機能動作テスト

4.2 プロセス

本章では，本研究で行った開発プロセスについて説明する．開発プロセスは，前期と後期で異なるため，分けて説明する．ここで，前期は 10 月第 1 週〜11 月第 4 週まで，後期は 12 月第 1 週〜1 月第 4 週を指す．

また，開発はエンジニア 3 名が行った．

4.2.1 前期

開発前期の特徴として，エンジニアが 3 名一緒に同じ作業を行っている．開発前期に行った作業を，表 **1** に示す．

開発が始まったばかりということもあり，試行錯誤していた．初めに我々は顧客の要求をドキュメント化した．その後，エンジニア全員で技術調査を行った．

エンジニア全員で同じ作業をしているため，作業進行が非常に遅かった．そこで，後期では作業を分担することで，作業進行を早めている．

表 1 の，kintone 操作用 API の開発は，元々 kintone に REST API が存在していたため，無駄な開発となっている．これは，エンジニアの調査不足に起因していると考えられる．

4.2.2 後期

開発後期では，エンジニア 3 名の基本作業をそれぞれ次のように振り分けた．

- プログラマ
  システムのプログラムを行う
- プロダクトマネージャー
  顧客との連絡を主に行う
- ドキュメント管理者
  要求書，仕様書などの作成を行う

それぞれ各エンジニアが得意としている作業を受け持った．これにより，我々の開発速度は上がった．

また，開発後期では，実際にプロダクトを作成し，開発物をベースにミーティングを行った．開発後期に行った作業については，表 **2** に示す．

Vol.2016-ARC-219 No.35
Vol.2016-SLDM-175 No.35
Vol.2016-EMB-40 No.35
2016/3/25

表 2　開発後期の作業
Table 2　The work of 2H development.

	12 月	1 月
第 1 週	スイッチ ON 時に時間データを送信する機能を実装	データ送信失敗時に再送する機能実装 —
第 2 週	SD カードからアクセスポイントを読み込み接続する機能を実装	データ送信失敗時に再送する機能実装 —
第 3 週	通信不良時のデータ書き出し機能を実装, 稼働状態に合わせた LED 動作を実装 —	WiFi 接続動作の改善, 仕様書第 2 版の作成, LED の動作を変更
第 4 週	割り込み処理のデバッグ, 仕様書第 1 版の作成	仕様書最終版の作成 —

5. 直面した課題とその解決

本研究で直面した課題は，以下である．

- ESP-WROOM-02 の配線問題
- WDT [*2] の問題
- デバッグの問題
- SPI 配線の問題
- SD カード入出力制御問題
- 割り込み処理の問題

本章では，この直面した課題の解決方法について説明する．

5.1 ESP-WROOM-02 の配線問題

これは，使用する部品の適切な配線が不明だったために起きた問題である．

この問題で直面した課題は，まず情報量の少なさである．本研究で我々が使用した ESP-WROOM-02 は MICROTECHNICA 製であった．しかし，インターネットにある情報では，他社製が多く，PIN の位置が違うため，配線方法が異なっていた．加えて，インターネットの情報は不正確な情報も多い．

そこで我々は参考文献 [2] の ESP-WROOM-02 のデータシートを参照し，この情報をもとにインターネットに公開されている画像を参考に配線を行うことで，この課題を解決した．

5.2 WDT の問題

これは，マイコンボードに対する知識不足によって起きた問題である．この問題は，Arduino 言語 [*3] を扱っていた際に発生した．

我々は，この問題について，インターネットを使い情報を集めた．その結果，Arduino 言語で，while ループを使った時間待ちを行う際に，delay() を全く呼ばない場合に発生することが分かった．Arduino の仕様により，delay() が全くない場合，WDT が更新できず，プログラムがパニックになる．そして，プログラムが異常終了するため，今回の問題は発生していた．

そこで，我々は，while ループを使った時間待ち箇所には delay() を呼び，1ms 時間待ちすることで対処した．しかし，この対処法は後述する 5.6 節の割り込み処理と競合するため，注意する必要がある．本研究では，この対処による支障がない動作にしている．

5.3 デバッグの問題

我々は，デバッグをシリアルモニターを使って行なっていた．

シリアルモニターは，シリアル通信を行なって数値や文字を表示することが可能である．

また，プログラムをする際に標準出力機能を使ったデバッグはよく行われる．そのため，我々も，シリアルモニターに文字の出力を行いデバッグ作業をしていた．

しかし，Arduino においては，標準出力を行うにはシリアル通信が必要である．そのため，Arduino などの組込みシステムでは，LED などの素子を用いたデバッグが望ましい．

そこで，我々は，シリアルモニターが不正確であることを認識した上で，LED 表示を併用してデバッグ作業を行うことで対処した．

5.4 SPI 配線の問題

SPI とは，シリアル・ペリフェラル・インタフェースのことである．本研究では，SD カードモジュールの接続に SPI を使用した．

我々はインターネットを使用し，情報を集めることにした．しかし，ESP-WROOM-02 に SPI を使用した配線例が非常に少なかった．そのため，配線しても正常に動作しない，不具合を起こすなど試行錯誤が続いた．

[*2] Watch Dog Timer の頭文字．コンピュータが正常か監視するためのタイマーである．

[*3] Arduino で扱うプログラミング言語．

Vol.2016-ARC-219 No.35
Vol.2016-SLDM-175 No.35
Vol.2016-EMB-40 No.35
2016/3/25

この問題は，参考文献 [2] のデータシートを参照することにより，SPI 配線する際の使用する PIN を参照することで解決した．

5.5 SD カード入出力制御問題

本研究では，顧客要求により，SD カードを実装している．

SD カード内のファイルは，Arduino のライブラリを使うことで読み書きが可能である．本研究では，ユーザーがプロダクトのオプションを指定できるようにしているが，このオプションの読込み動作で問題が発生した．

この問題は，2 つの原因があった．1 つは，読込み動作の際に，ポインタ参照ミスを起こし，プログラムが異常終了を起こしていた．これは，SD カードのファイル参照は引数の型に厳密さが要求されることに起因しているようで，定数型，ポインタ型を調整することで我々はこの問題を解決した．

2 つ目は，ファイルの読込み確認をしていた際に，読込み文字数が想定より少なくなっていた．これについては，プログラムに問題はなく，結果としては 5.3 節で述べたデバッグの問題であった．

我々はこの問題の原因調査のため，手戻りを発生させてしまった．手戻りについては第 6 章で述べる．

5.6 割り込み処理の問題

我々は特定の処理の実装のため，Ticker と呼ばれるコールバック関数を使用した．これにより，指定した時間毎に関数の割り込みが可能となる．本研究では，システムの稼働状態を示すため，LED の点灯方式に割り込み処理を使用している．

しかし，割り込み処理は delay 処理と競合するため，本研究では LED の点灯タイミングに多少のずれが生じている．また，ここでも，バグや設計の変更により手戻りが発生している．

この問題を解決するには，delay 処理を Ticker で代用する必要がある．だが，この対処法は 5.2 節に示した WDT の問題がある．そのため，本研究では顧客承認の上，許容することとした．

6. 開発の振り返り

本研究では，180 人 H という短期間でシステム開発をした．180 人 H は約 1 人月に相当し，小さな開発コストで実現できたと言える．また，開発に使用する機器を汎用ボードにすることで，費用を抑えることができた．

しかし，本研究の開発では，以下の 2 つの手戻りが発生している．1 つは，SD カードモジュールの実装の際に，プログラムの設計を多少変更する必要があったことに加え，その変更の際にバグが発生したことに起因している．このバグを修正するために，プログラムの構造を変える作業が必要になった．

もう 1 つは，割り込み処理に起因する手戻りである．これは，delay を使った制御する部分の設計を変更する必要があったためである．設計変更により，LED が想定外の動作をすることがあったため，LED の点灯方式の見直しを行う必要があった．

本研究では，上記の手戻りが発生したが，どちらも 4 人 H 以内で解消している．手戻りが小さく，かつ，少なく開発できた要因としては，参考文献 [4], [5] にある，機能を少しずつ作り上げていくという，アジャイル型の開発による成果だと考えられる．

また，次回の開発の際には，割り込み処理や SD カードなどのモジュール増設を考慮して着手することで，手戻りを無くすことができる．そのため，工数はさらに削減でき，より低コストでの開発が可能であると考えられる．

7. 関連事例：Mats

Mats とは，米国ラスベガスにて開催された CES2016 で展示された，ドイツの smarter 社のスマート家電である [6]．なお，smarter 社ホームページ [7] では，論文投稿時点では公表されていなかった．

Mats は WiFi 通信を利用し，スマートフォンに醤油，ケチャップなどの調味料の残量を記録する IoT である．Mats には，重量センサーやネットワーク通信用機器を内蔵しており，本研究の思想と非常に類似している．また，参考文献 [6] より，Mats の価格は 150 ドル未満を想定している．

しかし，本研究で開発したプロダクトのハードウェアは約 3,000 円で構成可能であり，これに人件費などを加算しても Mats より低コストであると考えられる．

8. まとめ及び将来課題

8.1 まとめ

本研究では，中小企業に向けた低コスト在庫管理システムの需要を見出した．そこで，我々は，低コストでシステムを構築するため，WiFi 通信用 Arduino マイコン ESP-WROOM-02 と PaaS 型クラウドサービス kintone を利用した．

本研究で作成したプロダクトは，スイッチが ON になった際，kintone にデータの送信を行うものである．

また，プロダクトにかかった開発期間は約 4 ヶ月で，3 名のエンジニアと 2 名のアドバイザーで行い，開発はアジャイル型の開発を参考に行った．

アジャイル型の開発を行うことにより，手戻りがあっても最小限の追加工数で抑えることができた．また，ハードウェア部品に汎用マイコンボードを使用することで，ハードウェア部品のコストを抑えることができた．

その結果，成果物として，低コスト在庫管理 IoT システムのプロダクトを開発することができた．

Vol.2016-ARC-219 No.35
Vol.2016-SLDM-175 No.35
Vol.2016-EMB-40 No.35
2016/3/25

8.2 将来課題

今回のプロダクトはシステムの基礎的部のみの開発であるので，様々な課題が残っている．

まず，本研究で使用した通信用モジュールが WiFi モジュールであるため，倉庫には無線通信環境がある事が前提になっている．これは，通信用モジュールを 3G 回線を使用するものに変え，通信環境のコストを下げることが可能と考えられる．

次に，本研究では，消費電力については考慮していない．そのため，実運用する場合には省電力化する必要がある．これに関しては，スリープモード搭載のモジュールを使用することである程度は解決可能であると考えられる．

他には，在庫状況を検知するセンサーも一考する余地がある．本研究では，センサーをスイッチとしたが，重量センサーを使う事で，在庫の保管形式の影響を抑えることができる．また，使用するモジュールによっては，GPIO インターフェースの数が限られているため，搭載可能なセンサーの数が決まっている．

さらに，本研究で開発したプログラムはまだモジュール化できていない．プログラムの再利用を可能にすることで，今後の開発をよりスムーズに進めることができるため，モジュール化は必要である．

今後はこれらの課題を解決していきたい．

参考文献

[1] 中村 太一：モヤモヤ解消！タイプ別クラウド・サーバ全集，Interface，pp.68-75，CQ 出版社 (2016-01).
[2] 秋月電子通商：ESP-WROOM-02 データシート，入手先 ⟨http://akizukidenshi.com/download/ds/espressifsystems/0C-ESP8266__WROOM_WiFi_Module_Datasheet__EN_v0.4.pdf⟩（参照 2016-02-19）
[3] サイボウズ：kintone，入手先 ⟨https://kintone.cybozu.com/jp/⟩（参照 2016-02-19）
[4] Jonathan Rasmusson：アジャイルサムライ–達人開発者への道 初版，オーム社 (2011).
[5] 平鍋健児，野中郁次郎：アジャイル開発とスクラム 初版，翔泳社 (2013).
[6] Engadet：iPhone で醤油やケチャップの残量がわかる。スーパーでの買い物中に便利なスマート家電「Mats」：CES 2016，入手先 ⟨http://japanese.engadget.com/2016/01/05/iphone-mats-ces-2016/⟩（参照 2016-02-19）
[7] smarter：ホームページ，入手先 ⟨http://smarter.am/⟩（参照 2016-02-23）