オープン環境による M2M/IoT システム構築の動向と取り組み事例

清 尾　克 彦[1]

１．はじめに

M2M（Machine- to- Machine）システムは，センサを含む機器同士または機器とクラウド上のサーバをネットワーク経由で接続し，センサからのデータを分析して様々なサービスを提供する[1)]．最近では IoT（Internet of Things，モノのインターネット）という言葉が M2M を包含する形で社会の変革を促す第 3 の IT 化の波[2)]として注目されている．また，モノづくりの分野でもインダストリ・インターネットやドイツによるインダストリー4.0 の取り組みがはじまり，第 4 次産業革命の到来と喧伝されている[3)]．まさに，オープンカルチャーに基づいたイノベーションが進みつつある．

この M2M/IoT システムは多くの技術の組み合わせで実現される複雑系システムであり，実現するためにはかなりの技術力と労力が必要となる．また，M2M/IoT システムはアイデア次第でその応用範囲は多岐にわたると考えられる．アイデアが本当に効果があるかどうかを確認するために，アイデアの創出からプロトタイプによる評価を短期間に繰り返して，実用化を目指すことが求められている．

このような課題を解決して，将来の発展が期待される M2M/IoT システムの実現を図るためには，オープン化及び標準化の成果を活用し，多くの技術から構成されるプロトタイプを容易に構築できるオープンなプラットフォームの整備が必要になると考えられる．

本稿では，M2M/IoT システムのプロトタイプを誰もができるだけ安価に効率よく開発できるように，オープン環境による M2M/IoT システム構築の動向と，M2M/IoT システムの仕組みを理解することを狙いとした教育用の M2M/IoT プロトタイプ構築のサイバー大学での取り組み事例を紹介する．M2M/IoT システムの概要や応用事例等については，サイバー大学紀要第 5 号「M2M（Machine to Machine）技術の動向と応用事例」[4)]を参照ください．なお，本稿は筆者が所属する NPO 法人 M2M 研究会[5)]や電気学会 M2M 技術調査委員会（第 1 次，第 2 次）での発表内容[5-7)]などを基にまとめたものである．

[1] サイバー大学 IT 総合学部・教授

２．M2M/IoT システムとオープン化の流れ

2.1. M2M/IoT システムとは

M2M システムは，機械や電気機器の間で，人間が介在することなく収集されたデータを送受信することによってサービスを提供するものである．図 1 に示すように，機器（デバイス）にはマイコン等のコンピュータが内蔵され，接続されたセンサからのデータが，ゲートウェイ経由ネットワークを介してサーバに送信される．センサのデータはサーバに蓄積され，クラウド上のアプリケーションにより，集積されたデータを可視化したり，分析をしてマネジメントの判断材料にしたり，分析結果から自動的に機器の制御をしたりするサービスが実現される．M2M の「M」は，Machine のかわりに Man や Management に読みかえて使われる場合もある．

IoT システムは，M2M を包含する概念として，従来のインターネットの世界にモノのインターネットを融合して，サービスを提供するものである．本稿では，以下，M2M/IoT システムとして説明をしていく．

このような M2M/IoT システムは，図 2 に示すようにセンサ/アクチュエータが接続された M2M デバイスや M2M ゲートウェイを取り扱う組込み技術，エリアネットワークとしてのワイヤレスセンサネットワークやアクセスネットワークとしての IP ネットワークなどを扱うネットワーク技術，M2M サーバにデータを蓄積し M2M アプリとしてサービスを提供するクラウドコンピューティング技術，マンマシンインタフェースとして人間との接点となるモバイルコンピューティング（スマートフォン）技術，安全・安心なシステムを提供するためのセキュリティ技術などを組み合わせて実現される．これらの技術を習得

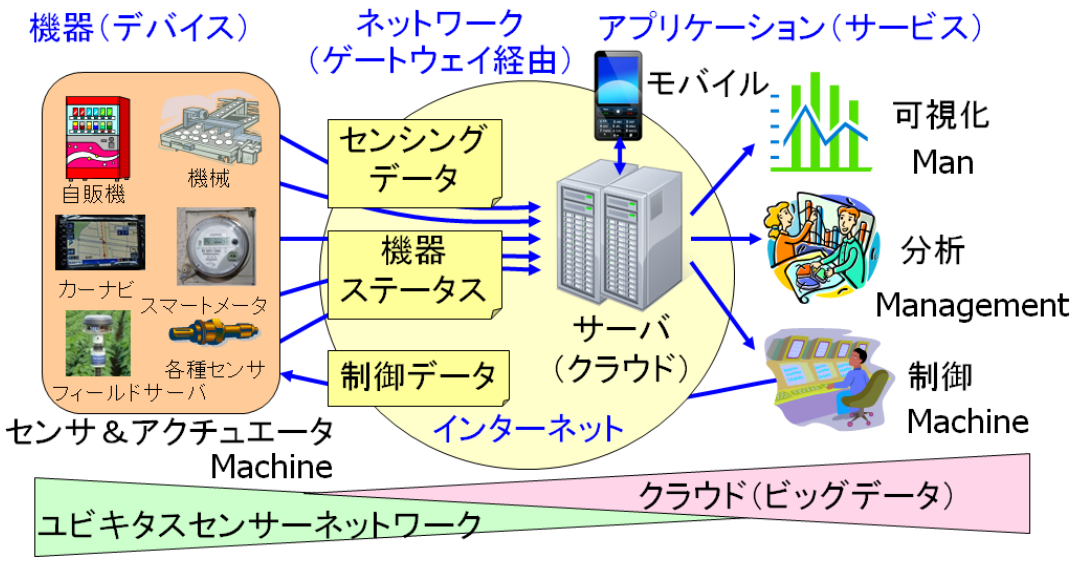


図 1　M2M/IoT システムのイメージ

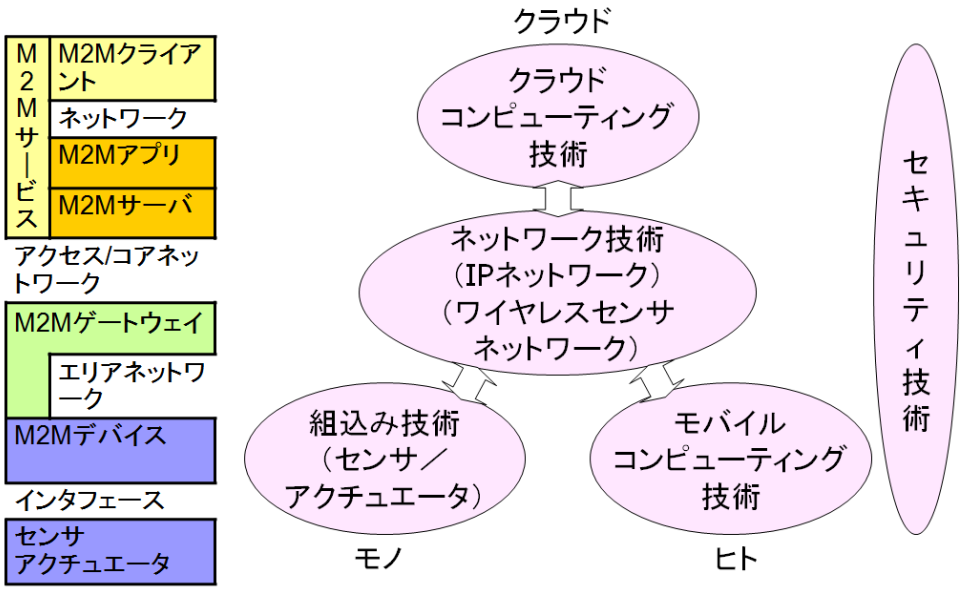


図 2　M2M/IoT システムを構成する主な技術

し，最適な組み合わせで応用分野に適した M2M/IoT システムを構築するには，多くの技術力と工数を必要とする．特に発展著しい分野で，これらの技術を一人（一社）ですべてを一から実現することは困難であり，急速な発展を支えているオープン化の成果と標準化の成果を活用することが必要である．

2.2. オープン化の流れ

ICT 分野ではオープン化により，急速に技術革新が進んでいる．オープン化の流れは，図 3 に示すように，無体物のソフトウェアから始まり，コンテンツのオープン化からさらに最近では国や自治体の持つ公共データのオープン化にまで拡がってきている．また，最近では安価なマイクロコンピュータやセンサなどが入手しやすくなり，プリント基板の製作も容易になったことから，有体物である基板などの回路図を公開するハードウェアのオープン化が進んできている．さらに，メカニカルな分野でも 3 次元データを使った安価な 3D プリンタなどのデジタルファブリケーションが普及しはじめたことから，ものづくりのオープン化も急速に進みつつある．このようなオープンな環境で，いろいろな分野の人たちがコラボレーションして創造的な設計を進めるオープンデザインも注目されている[8]．また，自社の技術と他社や大学などが持つ技術を組み合わせ，革新的なビジネスモデルや研究成果・製品開発に結び付けるオープンイノベーションの取り組みも行われつつある．

創造的な M2M/IoT システムを構築する観点からは，これらのオープン化や標準化の流れを有効に活用することが求められる．なお，以下の本稿では M2M/IoT システム構築に

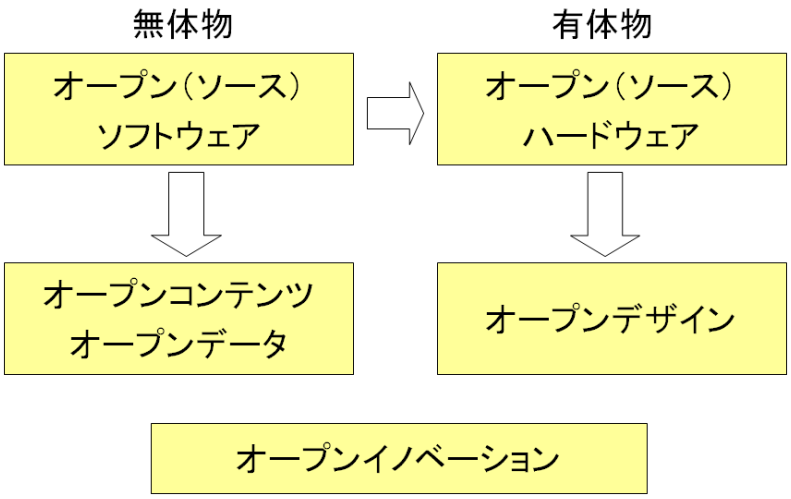


図 3　オープン化の流れ

必要なオープンソフトウェアとオープンハードウェアを中心に紹介していく．

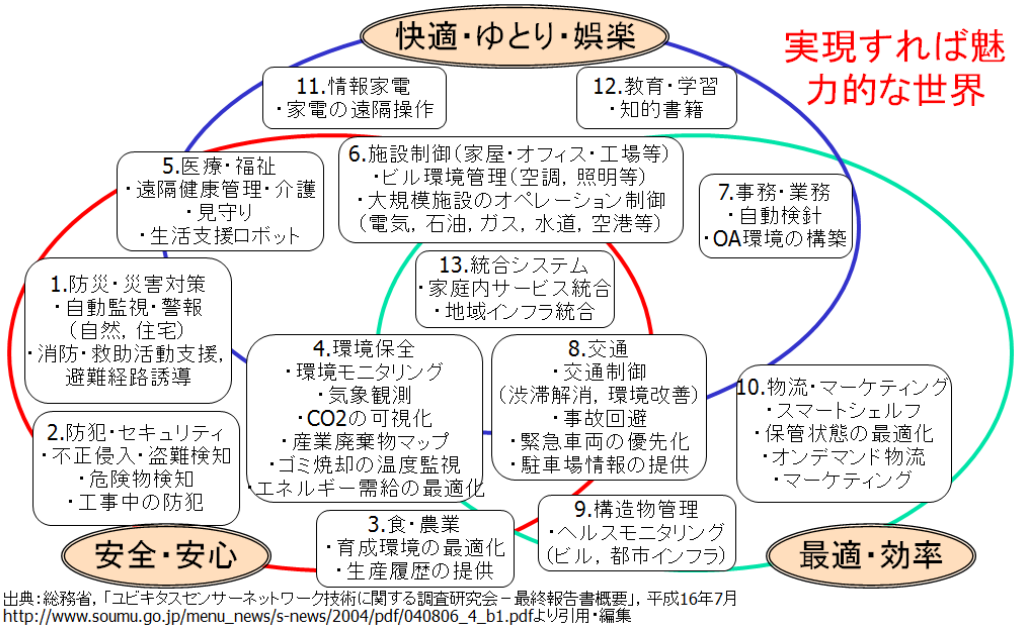
オープン化のメリットは，世界中の洗練されたソフトウェア/ハードウェア資産の利用，ソースコード/回路図の参照・改良が可能，使用料が安価で無償のソフトウェアの利用，ベンダロックインがない（特定ベンダに拘束されない），コミュニティが充実しインターネット上から多くの情報が入手可能等である．逆にデメリットとして，ソフトウェア/回路図自体が無保証，トラブルの解決は自己責任，日本語情報の不足，ある程度以上の技術力が必要，ライセンスの扱い（商用利用への影響）等があげられる．オープン化のメリットを最大限に生かし，デメリットを少なくするようにマニュアルの充実を行うなどのアプローチが求められる．

なお，以下の説明において，オープンという言葉は，ソースが公開されているものや，無償で使えるものなど，M2M/IoT システムを構築する時に容易に安価で入手可能なものを対象としている．

2.3. M2M/IoT システムの応用分野とプロトタイピング

M2M/IoT システムの応用分野は，図 4 に示すように安全・安心，最適・効率化，快適・ゆとり・娯楽の視点から 12 の分野へと多岐にわたる [9]．この図は，10 年ぐらい前に公開されたユビキタスセンサネットワークにおける応用分野であるが，M2M/IoT システムを構成する技術の革新により，まさに実用化が進みつつあるところである．また，モノづくりの分野でもインダストリ・インターネットやドイツによるインダストリー4.0 の取り組みがはじまり，大きく変化しようとしている．

アイデアがあれば，魅力的な世界を実現することが可能になりつつあるが，すぐに実用化し成功するとは限らない．本当に価値があり，効果を上げることができるかを，プロト



出典：総務省，「ユビキタスセンサーネットワーク技術に関する調査研究会－最終報告書概要」，平成16年7月
http://www.soumu.go.jp/menu_news/s-news/2004/pdf/040806_4_b1.pdfより引用・編集

図 4　ユビキタスセンサネットワークを利用したアプリケーション

タイプによりアイデアを評価しアイデアを洗練しながら，本格的に展開するかどうかを判断する必要がある．

魅力的な M2M/IoT システムを実現するためには，図 5 に示すように，プロトタイプを誰もができるだけ安価に効率よく構築できるオープンな M2M/IoT プラットフォームの提供が求められる．プロトタイピングにより，アイデアのブラッシュアップを繰り返すことにより，プロダクトとして実用化することが期待される．

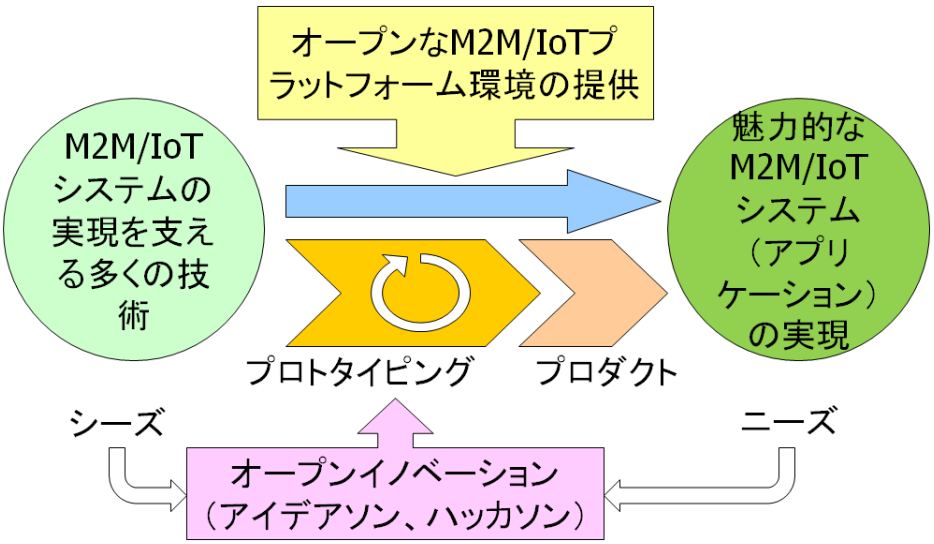


図 5　プロトタイピングによる魅力的な M2M システムの実現

図 6　M2M/IoT 関連のアイデアソン，ハッカソンの実施例

このようなオープンな環境を使って，図 6 に示すように，いろいろなところで M2M/IoT を対象にしたアイデアソンやハッカソンによるオープンイノベーションの試みが行われている．日本でも，自動車メーカによる公開された車からのセンサデータを活用してアイデアを競うハッカソンなどがいろいろ行われている．

3．オープン環境による M2M/IoT システムの構成

3.1．プラットフォーム

M2M/IoT システムを構成するプラットフォーム（ハードウェア，ソフトウェア）の全体イメージを図 7 に示す．M2M デバイス，M2M ゲートウェイ，M2M サービスに求められるハードウェア（コンピュータ機能）とソフトウェア（OS：オペレーティングシステム）の特性と，IP 化，低消費電力化，リアルタイム処理化の流れを示している．

IP 化の流れでは，リンク技術に依存しないアドレス体系・メッセージ配送技術・セキュリティ技術や多様な伝送メディア・リンク技術との相互接続性を実現可能で，豊富な実績がある IP 技術を低消費電力・低コストの M2M デバイスの領域まで展開するものである．IETF（Internet Engineering Task Force）で標準化された 6LoWPAN（ネットワーク層）と CoAP（アプリケーション層）が該当し，M2M デバイスでの実装が進んでいる．

低消費電力化の流れでは，すべての領域で求められるが，特に電力源のないところで使われる M2M デバイスについては最重要課題であり，軽量な OS，軽量なプロトコルが求められる．

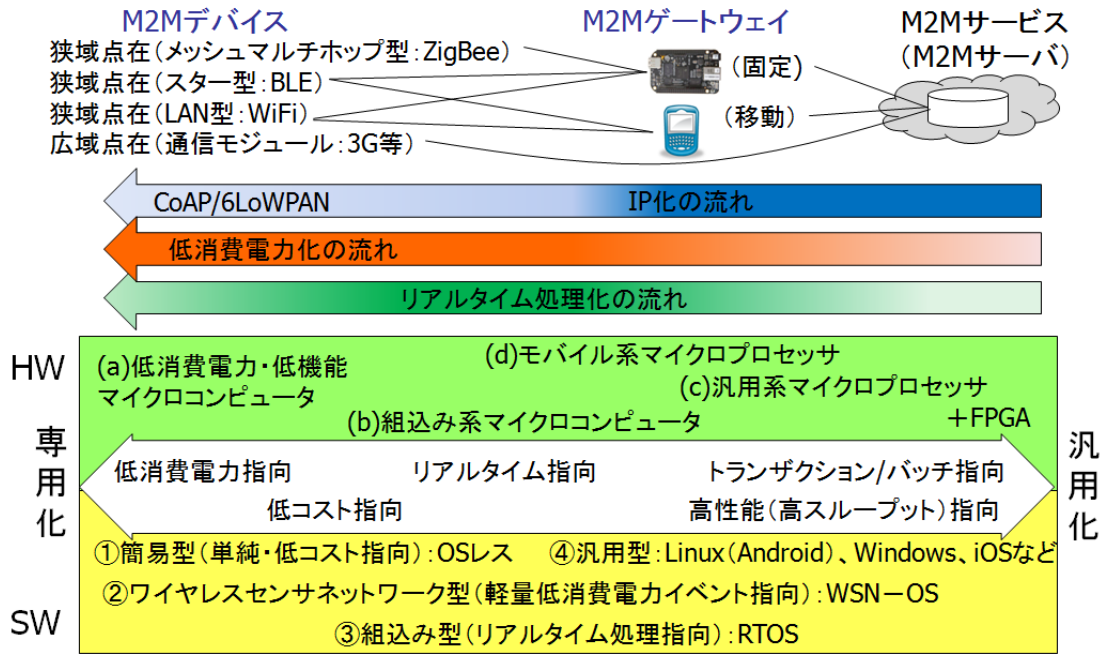


図 7　M2M/IoT システムのプラットフォーム

リアルタイム処理化の流れでは，M2M サーバでのストリーム処理やバッチ処理に対して,データ発生源に近いところで処理を行うエッジコンピューティングが注目されている．IoE（Internet of Everything）を提唱している米シスコ・システムズ社ではフォッグコンピューティングと呼んでいる．

3.2. オープン環境の分類

M2M/IoT システムを実現するオープン環境（標準化されたものを含む）の分類を図 8 に示す．なお，ここで取り上げたソフトウェアやハードウェアは，本稿での調査の対象としたもので，オープンなものはその他にもいろいろ存在する．

オープンハードウェアについては，M2M デバイスと M2M ゲートウェイを対象にする．

オープンソフトウェアについては，M2M サーバ上の Web アプリケーション/Web サーバなどで使うソフトウェアや，M2M ゲートウェイ上のサーバやデバイス管理などで使うソフトウェアや，M2M デバイスでのワイレスセンサネットワーク OS を対象にする．

オープンクラウドサービスについては，無料で公開されている BaaS（Backend as a Service：PaaS の一部）を対象にする．また，BaaS と連携してワークフロー的なサービスを提供する Web アプリケーション（マッシュアップ）も対象とする．

ネットワーク（IP 系）については，アクセス/コアネットワークおよびエリアネットワークで使われる IP ネットワーク上の通信プロトコルを対象とする．

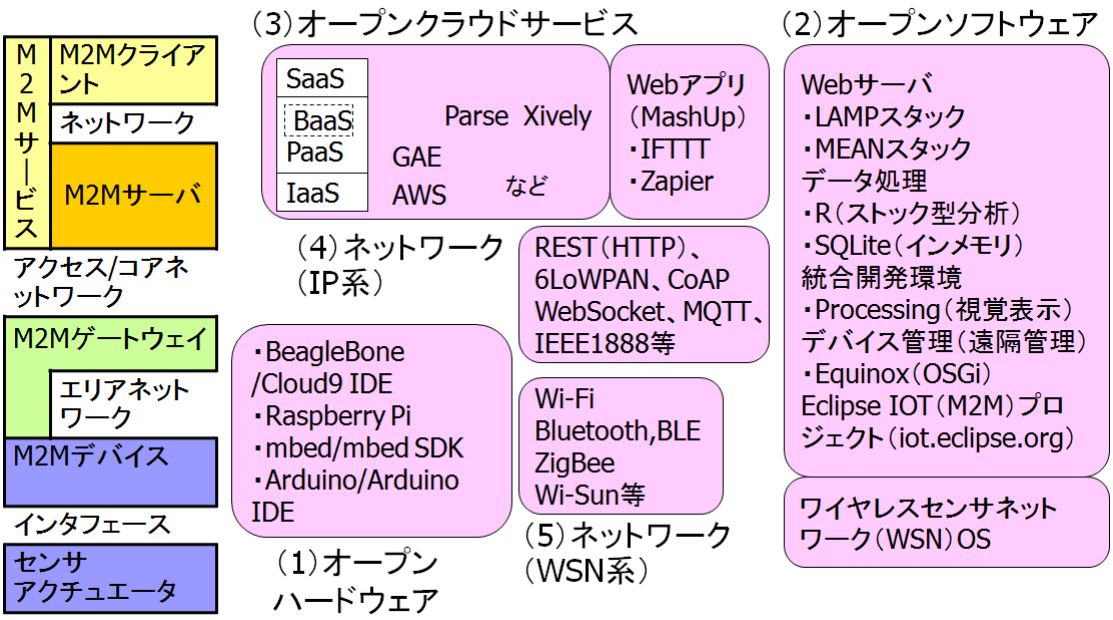


図 8　オープン環境の分類

ネットワーク（ワイヤレスセンサネットワーク系）については，エリアネットワークで使われるワイヤレスネットワークを対象にする．

3.3. オープンハードウェア

M2M デバイスとして Mote [10] が有名であるが，ここでは最近注目されている M2M デバイス/M2M ゲートウェイの安価で容易に入手可能なハードウェア（開発環境を含む）について紹介する．主なオープンハードウェアを図 9 に示す．

求められる低消費電力・低コスト性や性能・機能から目的とする M2M システムに最適なものを選択することになる．また，ここで紹介するものは，プロトタイプや実験機の作成を主眼にしており，製品として量産・出荷する場合には，信頼性の向上などを図る必要がある．

（1）Arduino（アルドゥイーノ）[11]：2005～，イタリア

フィジカルコンピューティングの元祖の 1 つで，ATmel 社の AVR マイコン（8 ビット系）を使用している．未経験者でも簡単に利用できるようにシンプルな構成（OS レス）で，PC 上の開発環境からプログラムをローディングするだけで使える．電子回路はシールドとして，プログラムはスケッチとして公開されている．シンプル故に性能や機能に制約がある．基板回路図が公開されており，多くの互換機（ルネサスエレクトロニクス社の GR-SAKURA，Intel 社の Galileo）が存在する．

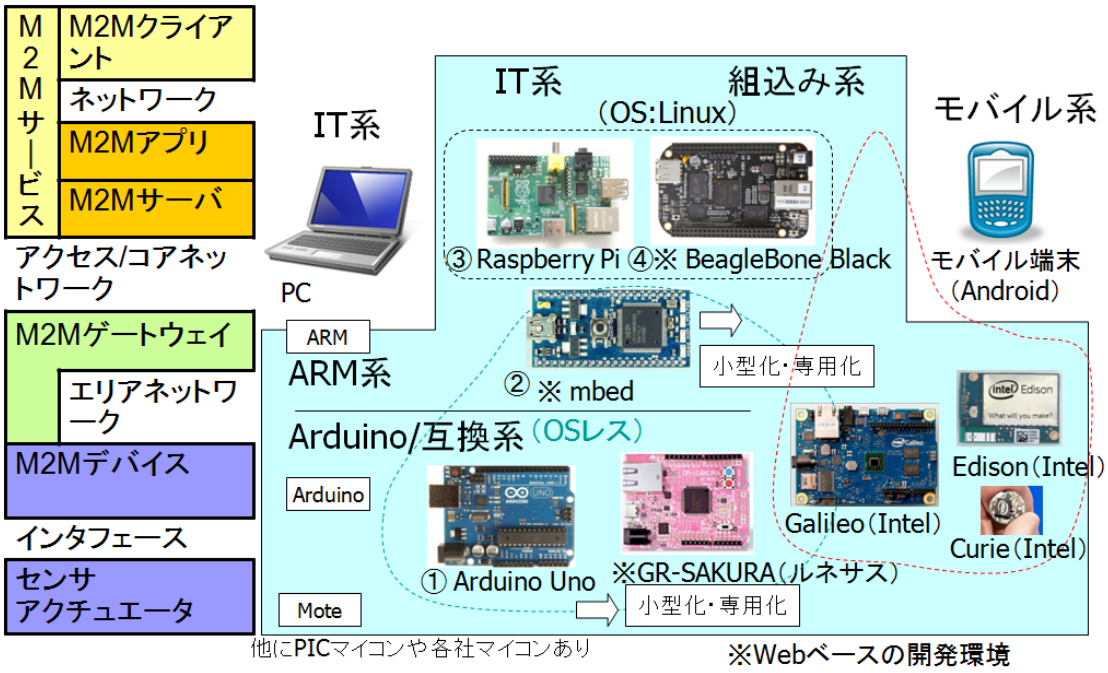


図 9　主なオープンハードウェア

（２）mbed[12]：2005～，ARM 社

ARM コアを搭載した NXP 社のマイクロコンピュータを使用している．すぐにプロトタイプを作れることを狙いシンプルな構成（OS レス）となっている．Web 上の開発環境からプログラムをダウンロードするだけですぐに使えることができる．基板回路図が公開されており，より小型化・専用化された互換機がつくられている．最近は，ARM 社が力を入れており，mbed OS を提供している．

（３）Raspberry Pi[13]：2012～，英国

ARM コアを搭載した Broadcom 社のマイクロコンピュータを使用している．IT 教育用に考案されたもので，コストパフォーマンスに優れた Linux ベースの小型 PC である．IT 教育用のため，他に比べて，I/O インタフェースが少ない．OS は組み込まれておらず，推奨の Rasbian を SD カードにダウンロードする必要がある．

（４）BeagleBone Black[14]：2008（Beagle Board）～，TI 社

ARM コア搭載の TI 社のマイクロコンピュータを使用している．組み込み用途向きで，I/O インタフェースが多く，この分野では最も高機能な Linux ベースの小型 PC である．プロトタイプからプロダクトへの対応が容易と言われている．OS（Angstrorm）が組み込まれておりすぐに使える．キーボードとマウスなしで起動できる．

（５）Galileo，Edison[15]：2013～，2014～，Intel 社

Galileo は，Quark SoC 搭載し，Arduino 互換機能のほかに，Linux を動作させることもできる．Edison は，ATOM SoC を搭載した SD カードサイズの Linux ベースの超小型 PC で，Arduino 互換カードも提供されている．

3.4. オープンソフトウェア

M2M/IoT システムで使用されるオープンソフトウェアはいろいろあるが，図 8 のオープンソフトウェアの中で以下の 4 つに絞って紹介する．

（１）Eclipse 財団 IoT プロジェクト（当初は M2M プロジェクト）[16]

オープンソースの Eclipse プロジェクトを運営する非営利団体の Eclipse 財団が，M2M/IoT システムをより簡単に開発できるように誰でもが使えるオープンな M2M/IoT 開発環境（プラットフォーム）を目指して開発を進めている．現在は IoT プロジェクトと名称を変更しているが，目標は同じである．M2M デバイス/M2M ゲートウェイにおけるデバイス管理に OMA-LWM2M（Lightweight M2M）を採用し，通信プロトコルとして CoAP や MQTT を採用し，Lua 言語による開発環境の提供を目指して開発が進められている．

（２）ワイヤレスセンサネットワーク（WSN）OS

大量のセンサを配置して低消費電力・低コスト性が求められるワイヤレスセンサネットワークのノードやゲートウェイにおいては，軽量な OS が求められる．単にセンサからのデータを送信するのみの場合は OS レスで十分であるが，デバイス管理機能，セキュリティ機能や IP ベースの低消費電力ネットワーク（6 LoWPAN や CoAP）プロトコルを処理する場合には OS が必要となる．実現方法として以下の 3 つの方法がある[17][18]．

①イベントモデル型

イベントモデルではイベントに応じて run-to-completion のタスクを実行する．プリエンプションを前提としていないので省資源かつ低オーバヘッドで実現可能である．また，各タスクが不可分に実行されるので共有資源に対する排他制御が不要となり，安全性が高い．しかし，ユーザが一連の処理を細かい処理に分割しなければならないのでプログラムが書きづらいという問題が発生する．さらに，イベントモデルではタスクのプリエンプションをしないことを前提に設計されているのでハードリアルタイム処理のサポートができない．イベントモデル型のオペレーティングシステムとして代表的なものが TinyOS[19] で他に Contiki[20] などが挙げられる（共にオープンソース）．ARM 社が最近力を入れている mbed OS もこれに該当する．

②スレッドモデル型

スレッドモデルでは複数の独立した実行ストリームが実行され，それぞれのスレッドはプリエンプションされる．ユーザはあたかも CPU を占有しているかのようにプログラム

を記述できるのでプログラムが書きやすい．また，ハードリアルタイム処理をサポートすることができる．しかしながら，プリエンプション時のオーバヘッドの大きいことや必要とされる資源が多いこと，スレッド間の共有資源へのアクセス制御が必要となるために安全性が損なわれるなどの問題を持っている．スレッドモデルを用いたオペレーティングシステムの研究としては MANTIS OS や PAVENET OS などが挙げられる．

③仮想マシン型

ワイヤレスセンサネットワークに特化した命令セットを具備する仮想マシン上でモジュールを実行することでモジュール自体のサイズを小さくすることができ，モジュール転送に伴う負荷を軽減することが可能となる．さらに，仮想マシンを異なる種類の CPU に移植すればモジュールがそのまま使えるので高い移植性も実現できるが，仮想マシンとしてのオーバヘッドが存在する．仮想マシンの研究としては ASVM や VAWS などが挙げられる．以前の SUNSPOT や最近の IBM の Mote Runner が該当する．

（３）Web サーバ

オープンな環境で Web サーバを構築する場合，従来は LAMP スタック（Linux＋Apache＋MySQL＋PHP）などが使われてきた．M2M/IoT システムでは，1 回の通信におけるデータ量は少ないが，センサからの通信を同時に接続する数が増加傾向にある．従来のマルチタスクやマルチスレッドの Web サーバ（Apache）では，1 コネクションに 1 つのタスクやスレッドが生成され，同時接続数が増加するとメモリ増加，性能低下という問題が起きる（C10K 問題）．

これを解決するものとして Node.js[21] が注目され，非定型なデータを扱うことが容易な MEAN スタック（MongoDB＋Express＋AngularJS＋Node.js）が使われるようになってきた．Node.js は Chrome の JavaScript ランタイムを使った，高速かつスケーラブルなネットワークアプリケーションを容易に構築するためのプラットフォームである．シングルスレッドで，イベント駆動でキューを順次処理していく方式で，メモリ増加もなく，ノンブロッキング I/O で待たずに処理を続けることができ，高速に処理が可能である．

（４）統合開発環境の Processing[22]

Processing は Java をベースとしたオープンソースのプログラミング言語および統合開発環境である．Java を初心者に使いやすくした言語で，よく使うグラフィック，サウンド，シリアル通信，ネットワークなどに関して単純化することにより，簡単に扱うことができるようになっている．特に画像やアニメーションを用いた視聴的な表現を得意としている．Windows, Mac, Linux などの Java 仮想マシンを搭載可能なコンピュータで利用できる．また，作ったプログラムを，どのような OS でも動作する Java アプレットやプログラミング環境なしでも実行可能な形式に変換できる．

3.5. オープンクラウドサービス

クラウドサービスは図 10 に示すように，大きく IaaS（Infrastructure as a Service）と PaaS（Platform as a Service）と SaaS（Software as a Service）の 3 つに分類される．

M2M サービス（M2M サーバ，M2M アプリケーション）の実現方法として，IaaS や PaaS の上でオープンクラウド環境を利用して独自の M2M サーバを構築することも可能であるが，技術力と多くの開発コスト（工数）を必要とする．

最近では，PaaS 上に特定のアプリケーションサービスの運用に必要な汎用的なサーバ機能を提供する BaaS（Backend as a Service）が注目されるようになり，ある制約条件の中で無料で使用できるようになってきた．BaaS の一般的な構成を図 11 に示す．BaaS では M2M サーバとして共通的に必要となるユーザ管理/認証，データベース，データ表示，プッシュ通知などの機能を WeB API を使って容易に実現することができる．M2M アプリケーション開発者は，アプリケーション固有の開発に専念することができ，容易に開発・運営を行うことができる．

M2M 向けに特化した BaaS として，Xively（元 Pachube）[23] が有名である．特に，福島原発事故のときに即座に放射線データがアップされたことで注目された．センサからのデータを HTTP プロトコルで送信することでサーバにデータを蓄積し，グラフ表示をしたり，トリガ条件に基づいてプッシュ通知を行うことができる．なお，Xively と機能が似ている AT&T 社の M2X [24] を代替として使うことができる．Xively の特徴と表示例を図 12（a）に示す．

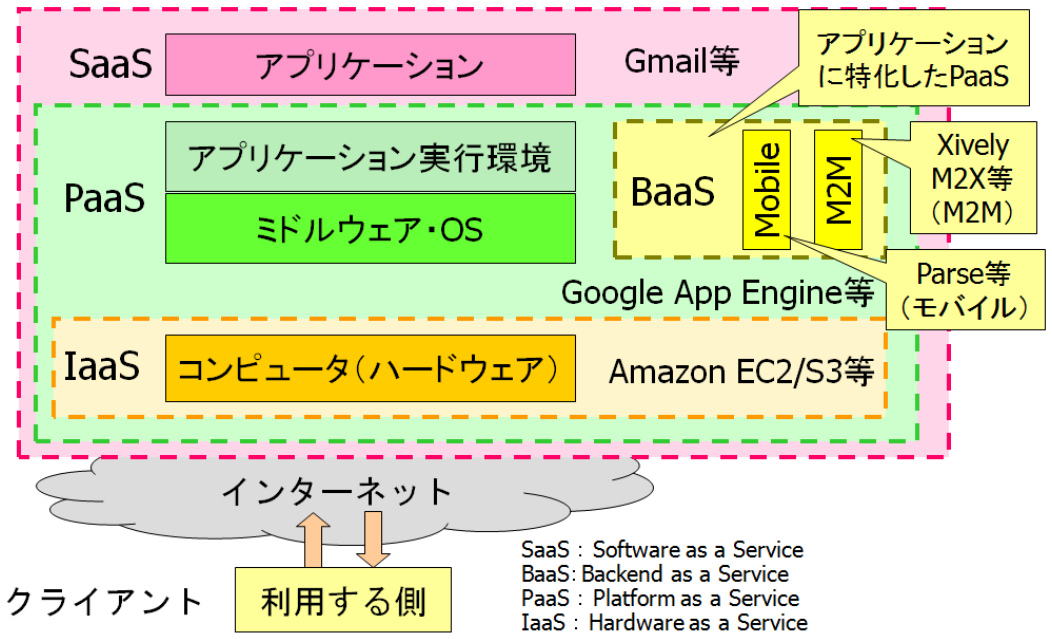


図 10　オープン M2M クラウドサービス

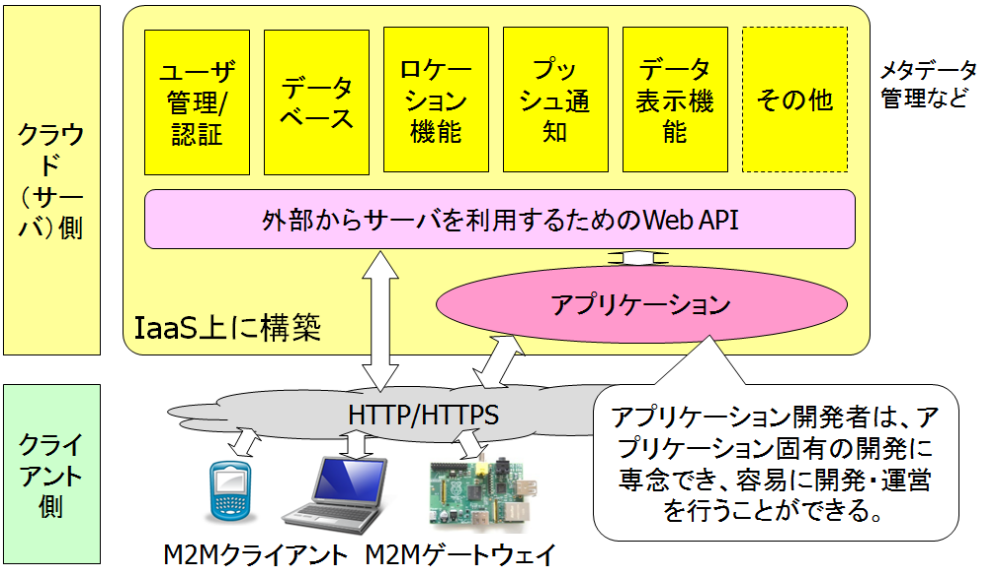


図 11　BaaS の一般的な構成

モバイルアプリ向けに汎用化された Parse[25] でも同じような機能を実現できるが，グラフ表示などは自分で作成する必要がある．同じクラウド上で Web アプリケーションを開発できる環境が提供されており便利である．Parse の特徴と表示例を図 12（b）に示す．

BaaS と連携してワークフロー的なサービスを提供する Web サービス（マッシュアップ）も便利な存在である．BaaS でのプッシュ通知機能と連動して，Gmail で異常を通知するなどいろいろな機能を容易に利用することができる．主な 2 つの例を以下に紹介する．

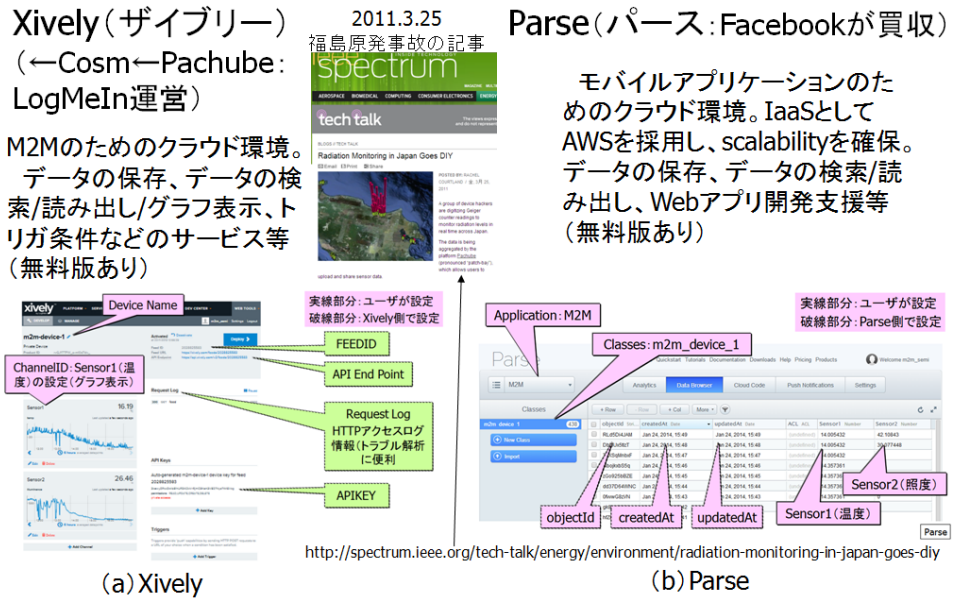


図 12　Xively と Parse の特徴と表示例

IFTTT（イフト）[26] は，「if this then that」というシンプルなコンセプトに基づく「レシピ」と呼ばれるプロフィールを使って，数ある Web サービス（Facebook，Evernote，Weather, Dropbox など）同士を連携することができる．レシピの「this」の部分は「Facebook で写真をタグ付けした時」といった「きっかけ」になり，「that」の部分は「テキストメッセージの送信」といった「行動」になる．Twitter，Foursquare，Flickr，Box といった 61 ものサービスに対応した「きっかけ」と「行動」を提供している．基本は無料で使える．

Zapier[27] も同様に，様々な API を連携してワークフロー的な処理を実行できる．例えば「Google スプレッドシートに書き込みを実施」したら「Gmail でメールを送る」という 1 つのワークフローを zap という単位で定義して，実行することができる．サポートしているサービスは 200 以上あり，基本は無料で使える．

3.6. ネットワーク（IP 系）

アクセス/コアネットワークでは IP ネットワークが使われている．従来の HTTP プロトコルを使った REST 方式が基本であるが，M2M では双方向通信やより軽量化が求められることから，いくつかのプロトコルが提案されている．REST を含めて，5 つの方式について以下に示す．

（1）REST（Representational State Transfer）：図 13 の（a）参照

HTTP を使って通信をおこなう方式で，「利用し易い」「確認し易い」ことが最大の特徴である．サーバから新しいデータを読み込むために定期的なポーリングを必要とする．これを改善するために考案されたロングポーリング方式は，クライアントからサーバへの HTTP 接続を，サーバが応答を送信するまで維持するもので，接続が多い場合にはオーバヘッドとなる．セキュリティを重視する場合は，HTTPS が使われる．

（2）低消費電力対応プロトコル（6LowPAN，CoAP）

IETF（インタネット技術標準化委員会）が HTTP の軽量化を目指して標準化をおこなった．6LoWPAN（ネットワーク層）[28] と CoAP（アプリケーション層）[29] に既存の UDP を組み合わせて，低 CPU 性能・低コスト・低消費電力可能なネットワークを実現することが可能である．

（3）WebSocket[30]：図 13 の（b）参照

最初に HTTP を使ってクライアントとサーバの間に持続的接続を確立し，そのあとはどちらの側からでも，いつでもデータの送信を開始することができる（双方向通信）．

（4）MQTT （Message Queue Telemetry Transport）[31]：図 13 の（c）参照

IBM が開発したものをオープン化したもので，ブローカー・ベースの軽量なパブリッ

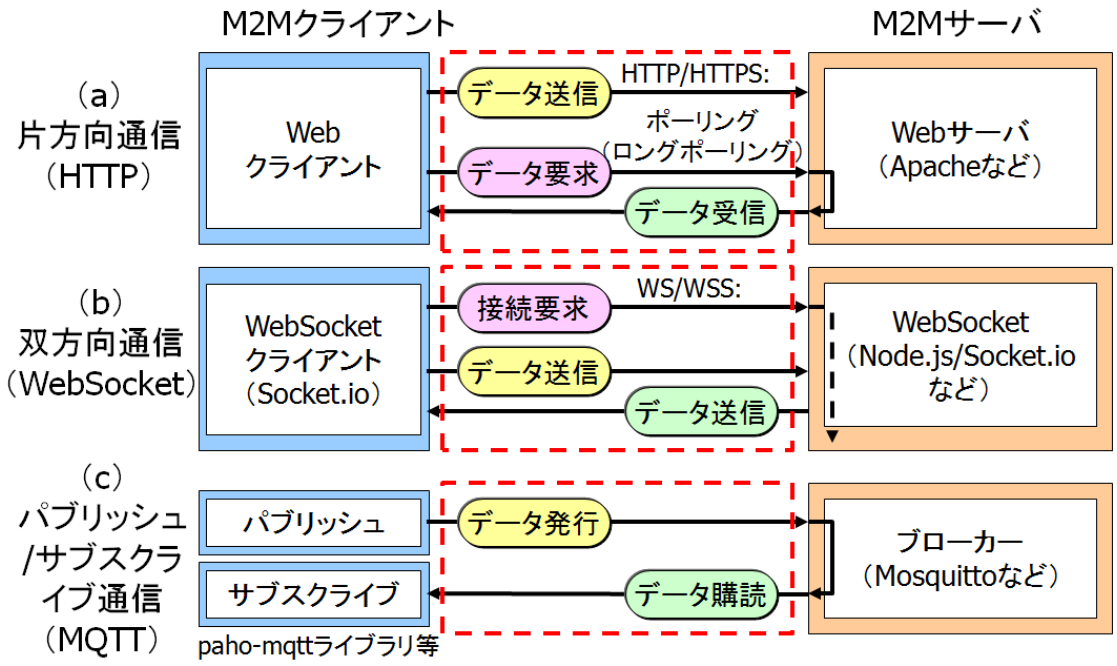


図 13　ネットワーク（IP 系）のプロトコル例

シュ/サブスクライブ型メッセージ・プロトコルである．MQTT はオープンで単純，軽量で，容易に実装できるように設計されている．

（５）IEEE1888（FIAP）[32)]

次世代 BEMS やスマートグリッド向けに開発され，2011 年に国際標準化されたオープンな通信規格である． HTTP と SOAP メッセージという XML によってメッセージ交換を行う．GW（ゲートウェイ），Storage（ストレージ），APP（アプリケーション）という 3 つのコンポーネントが，平等に WRITE，FETCH，TRAP と呼ぶ 3 種類の通信手順を使うことができる．

3.7. ネットワーク（ワイヤレスセンサネットワーク系）

M2M デバイスと M2M ゲートウェイ/M2M サービス間のワイヤレスセンサネットワークの構成を図 14 に示す．

（１）限られた領域（狭域）に配置された M2M デバイスと M2M ゲートウェイ間のワイヤレスネットワーク

（a）低消費電力で近接のセンサからデータを収集するスター型：Bluetooth Low Energy 等

（b）消費電力は大きいが高速で大量データを送信できる LAN 型：Wi-Fi 等

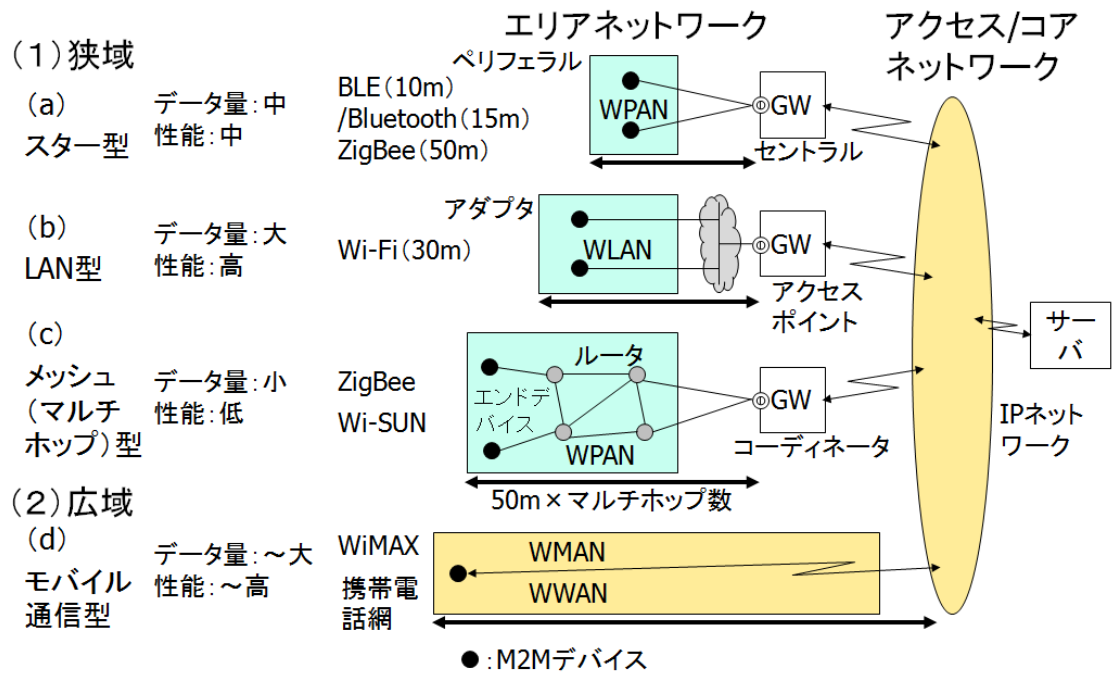


図 14　ワイヤレスセンサネットワーク系

（c）多数のセンサからデータを収集する低消費電力指向のマルチホップ・メッシュ型：ZigBee [33] 等．Wi-SUN [34] が日本の NICT（情報通信研究機構）を中心に標準化され，東京電力のスマートメータでの採用が決まり，今後大きく発展していくと考えられる．

（2）広域に配置された M2M デバイスと M2M サーバ間のワイヤレスネットワーク

（d）IP をベースとしたモバイル通信型の WiMAX や携帯電話通信モジュール（3G, LTE, PHS）が使用される．

4．オープン環境による M2M/IoT システムの構築事例

3 章のオープン化の状況を踏まえて，だれもが安価な費用で容易に M2M/IoT システムを構築できるように，図 15 に示すオープン環境による M2M/IoT システムの構築を進めている．M2M ゲートウェイを固定した場合における入門編（図 15 の a）と応用編（図 15 の b），M2M ゲートウェイを移動可能にした場合の応用編（図 15 の c）の 3 通りに分類し，入門編について具体的な事例を紹介する．なお，入門編の事例についてはサイバー大学 IT 総合学部のゼミナール（e ラーニング）で 2014 年秋学期から適用しているものである．

4.1. M2M/IoT システムの構成

（1）M2M ゲートウェイを固定した場合の例

M2M デバイスには低消費電力指向のマイクロコントローラが搭載され，センサやアクチュエータを接続し，ZigBee や Wi-Fi などのエリアネットワークにより固定された M2M ゲートウェイとの間でセンサデータやアクチュエータ制御データが送受信される．M2M ゲートウェイには比較的高性能なマイクロコントローラが搭載され，必要に応じてストリーム処理などのアプリケーションが実行されたり，センサデータなどがデータベースに蓄積されたりする．IP ネットワークを介して M2M サーバへセンサデータが送信され，ストリーム処理や蓄積した後にバッチ処理が行われる．これらの処理の結果として分析結果をわかりやすくビジブルに表示したり，アクチュエータを制御するデータを M2M ゲートウェイ経由 M2M デバイスに発信したりする．なお，限られた領域での小規模なシステムでは，M2M ゲートウェイで大部分の機能を処理することが可能であり，運用費用の安価な M2M クラウドを補助的に使用する場合も多い．

a：入門編

M2M/IoT システムの基本原理・基本機能を理解し，短期間に安価に容易にプロトタイプを構築して体感できることを狙いとした入門用のシステムである．詳細は 4.2 項で紹介する．M2M ゲートウェイに Arduino を，M2M ゲートウェイに PC を，M2M サーバにオープン M2M クラウドの Xively（または M2X）使うことにより，身近な環境で容易に M2M/IoT システムを体感できることを狙いとしている．

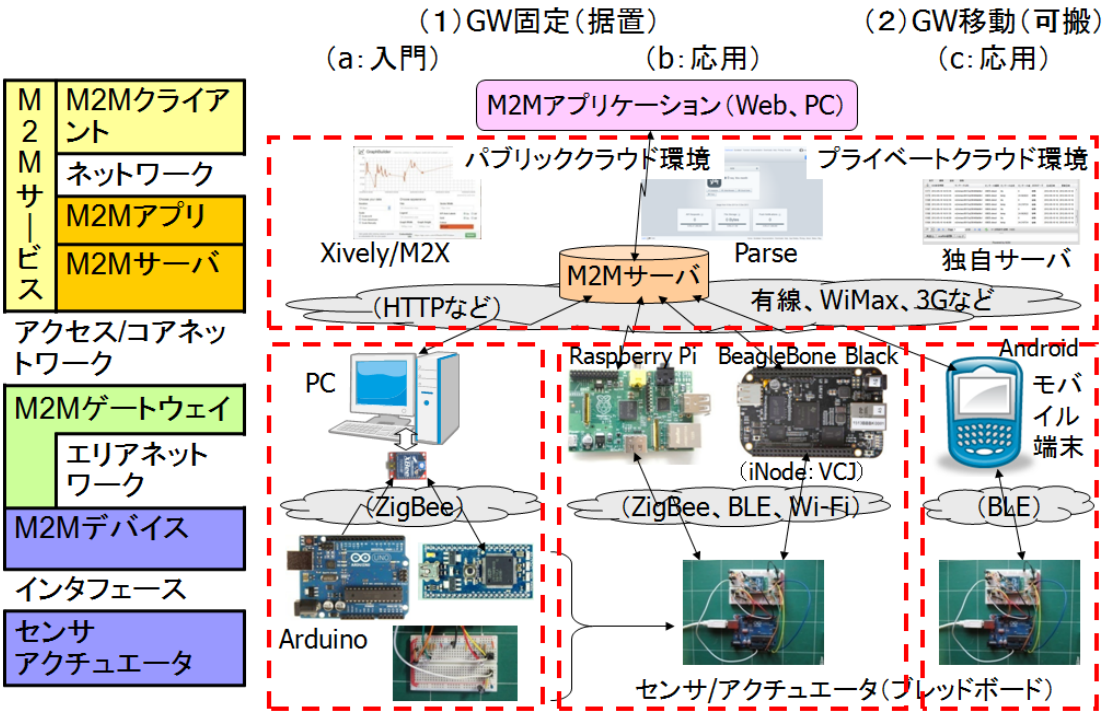


図 15　オープン環境による M2M/IoT システムの構成例

b：応用編

本格的に実用レベルで使用されることを前提にしている．入門編における M2M ゲートウェイの PC の代わりに RaspberryPi や BeagleBone Black などの専用ボードに置き換えたり，オープン M2M クラウドの Xively の代わりに IaaS 環境の上に独自の M2M サーバを構築することにより，より実用的なシステムに拡張したものである． また，エリアネットワークとして ZigBee 以外に高速送信可能な Wi-Fi や，低消費電力の BLE を使った応用も考えられる．

（2）M2M ゲートウェイを移動可能とした場合の例（c：応用編）

最近普及の著しいスマートフォンやタブレット端末などのモバイル機器は，Wi-Fi や携帯通信網で容易に M2M サーバと接続できる．また，低消費電力の BLE（Bluetooth Low Eneargy）があらかじめ搭載されており，BLE を搭載した M2M デバイスとも容易に接続できる．このようなモバイル機器を M2M ゲートウェイとして使用することにより，近辺に配置された BLE 搭載の M2M デバイスからのセンサデータを，M2M サーバへ送信することが容易に実現できる．モバイル機器には高性能なマイクロコンピュータと表示ディスプレイが搭載されているので，M2M ゲートウェイでリアルタイム処理をしたり，人間とのインタフェースとして M2M サーバに蓄積されたデータをビジブルに表示したりすることができる．

4.2. M2M/IoT システム入門編

教育用途の M2M/IoT システム入門編は，M2M/IoT に精通していない人でも，M2M/IoT システムを構成している主な技術を理解しながら，オープン化・標準化の成果を活用して安価に短期間でプロトタイプを構築し，実際に M2M/IoT システムを体感することで，新たな M2M/IoT システムの応用例を創造することを狙いとしている[7]．

M2M デバイス，M2M ゲートウェイ，及び M2M サービスの 3 つのドメイン構成で，ネットワークとしてエリアネットワークと IP ネットワーク（アクセス/コアネットワーク）を利用し，M2M デバイスに接続されたセンサからのデータを M2M ゲートウェイを経由して M2M サーバまでアップし，蓄積されたデータを時系列でグラフ表示するアプリケーションと，ストリーム処理で最新のセンサデータに対するトリガ条件の判定に基づいて M2M に接続されたアクチュエータを制御するアプリケーションを提供する．

M2M/IoT システム入門編は，はじめてこの分野を学ぶ方々を対象にステップバイステップで個々の M2M/IoT 技術を理解しながら，プロトタイプを構築できるように，図 16 に示す四部構成の演習カリキュラムで構成されている．

第一部の「M2M デバイスを動かす」では，M2M デバイスに各種センサを接続し，一定の周期でセンサの値を読み込み，測定値を計算する．

第二部の「M2M ゲートウェイにつなぐ」では，エリアネットワークを介して，M2M デ

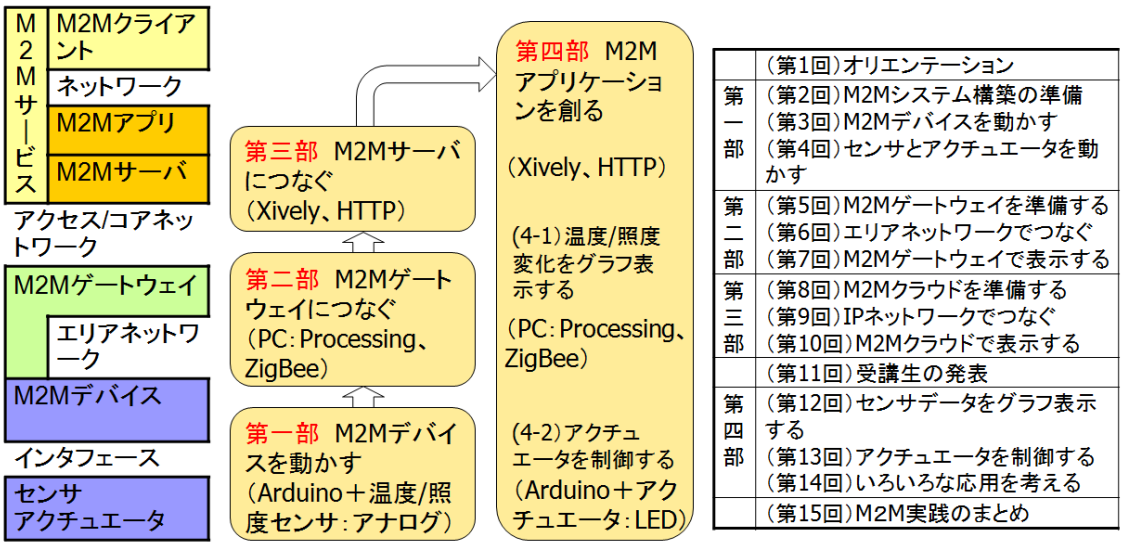


図 16　M2M/IoT システム入門編の演習カリキュラム

バイスから M2M ゲートウェイにセンサからの測定値を送信する．M2M ゲートウェイでは，センサからの測定値を一定の周期で読み込み，表示画面上に測定値を表示する．

第三部の「M2M サーバにつなぐ」では，アクセス／コアネットワークを介して M2M サーバが提供する API を使用して，M2M ゲートウェイからオープンクラウドの M2M サーバにセンサの測定値を格納する．M2M サーバが提供するグラフ表示機能で測定値のグラフ表示をおこなう．

第四部の「M2M アプリケーションを創る」では，2 つのアプリケーションを作成する．1 つ目は．M2M サーバが提供する API を使用して，M2M サーバから指定した期間のセンサの測定値を読み出して時系列にグラフ表示を行うアプリケーションである．2 つ目は，センサからの測定値があらかじめ設定されたトリガ条件を満たした時に，M2M デバイスのアクチュエータを制御するアプリケーションである．

（1）システム構成とデータの流れ

図 17 に入門編の全体のシステム構成とセンサからのデータの流れを示す．

M2M デバイスにはフィジカルコンピューティングの分野で草分けとして最も良く使われているオープンハードウェアの Arduino [11]を利用している．M2M ゲートウェイとしては 24 時間稼働する Linix マシンが想定されるが，初心者でもシステム構築やデバッグを容易にするために，入門編では PC と容易にビジュアルな表示やイベント処理を行える Processing [22] を利用している．M2M サーバには，制約のある範囲内で無償で使える Xively（元 Pachube）[23] を利用している．また，センサはアナログセンサの温度センサと照度センサの 2 種類に，アクチュエータはとしてデジタル出力（on/off）と PWM 出力に対応した LED に絞り込んでいる．

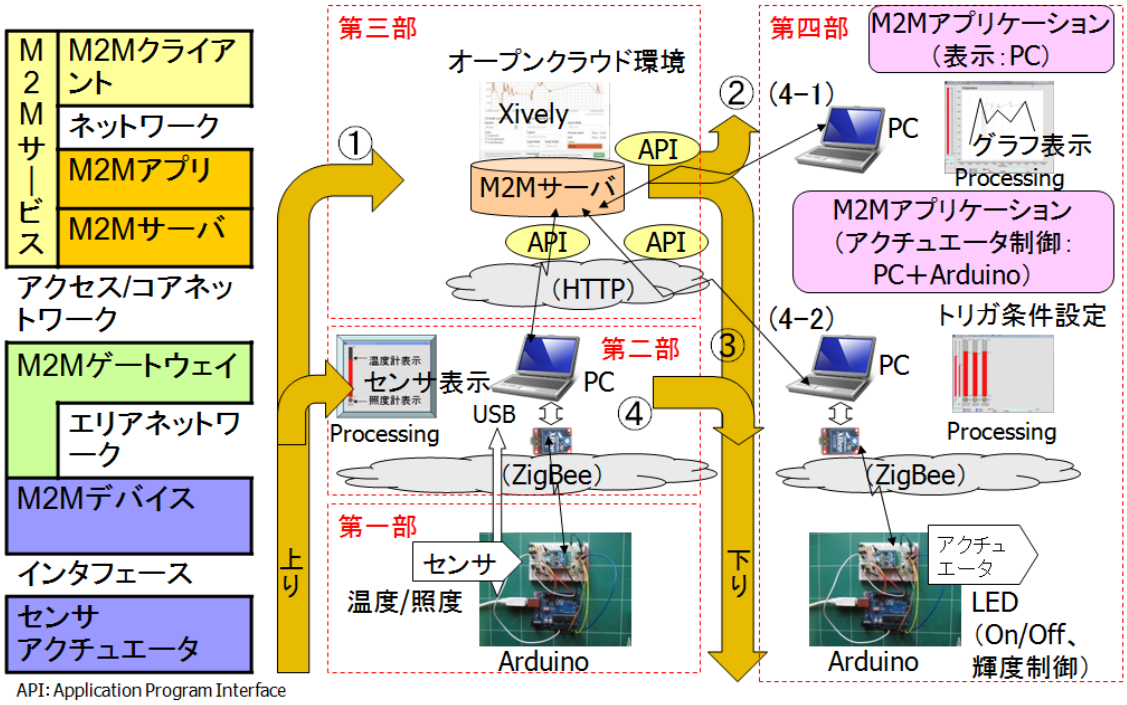


図 17　入門編のシステム構成とデータの流れ

エリアネットワークには低消費電力でメッシュネットワークを構成可能な ZigBee [33] を，アクセス/コアネットワークでは IP ネットワーク（HTTP/HTTPS）を利用している．

図 17 の①のパスでは，M2M デバイスに接続されたセンサ（温度，照度）のデータがアナログデータであるので，Arduino で AD 変換によりデジタルデータに変換し，さらに測定値に換算して，M2M ゲートウェイ経由 M2M サーバである Xively に蓄積する．②のパスでは，M2M サーバの Xively に蓄積されたセンサデータを読み出し，M2M クライアントである PC でグラフ表示する．③のパスでは，M2M サーバの Xively から読みだした最新のセンサデータに対して，M2M ゲートウェイで指定したトリガ条件に基づいて判定を行い，M2M デバイスに接続されたアクチュエータを制御する．例えば，蓄積された最新の温度が指定した値より大きい時に，警告として LED を点灯することができる．④のパスでは，M2M デバイスからのセンサデータに対して，M2M ゲートウェイで指定したトリガ条件に基づいてリアルタイムで判定を行い，M2M デバイスに接続されたアクチュエータを制御する．

（2）アプリケーションの概要

今回の M2M システム入門編で提供するアプリケーションは，センサデータをいろいろな形で表示（見える化）する機能と，時々刻々収集されてくるセンサデータをあらかじめ設定したトリガ条件を満足するか判定してアクチュエータを制御する機能である．

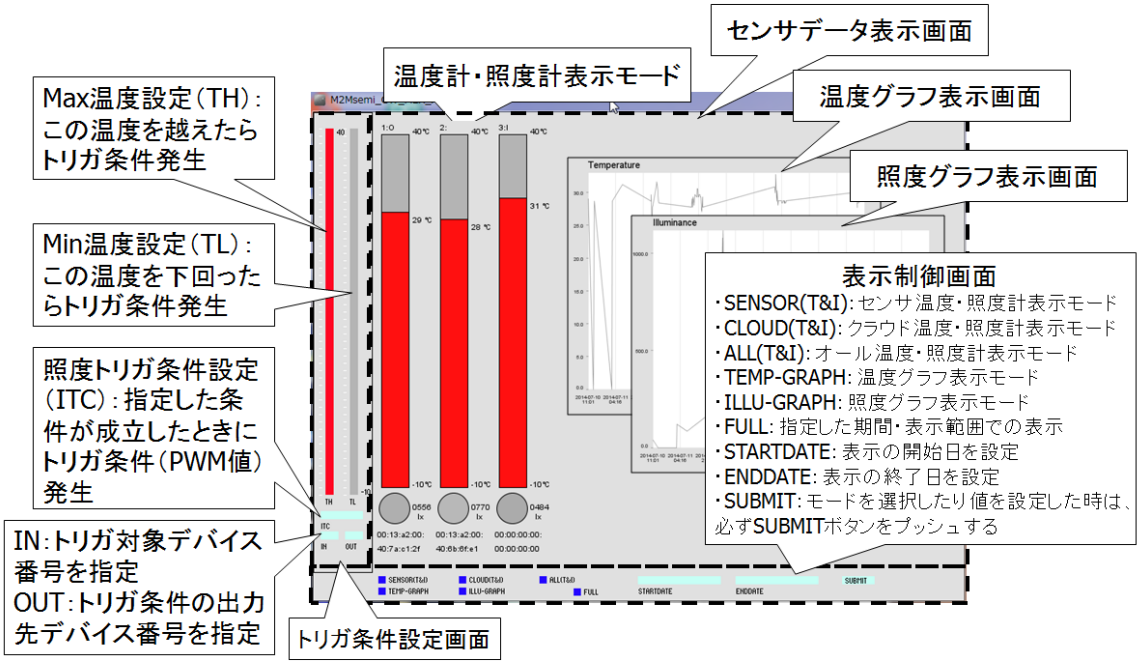


図 18　表示機能とトリガ条件設定機能

これらの機能は図 18 に示す M2M ゲートウェイの制御画面上で選択し，パラメータを設定することができる．

表示制御画面で表示したいセンサや表示モードを指定することができる．温度計・照度計表示モードでは，M2M デバイスからのセンサデータおよび M2M サーバからのセンサデータをビジブルに表示する．温度グラフ表示モードや照度グラフ表示モードでは，M2M サーバに蓄積されている指定した期間のセンサデータを時系列的にグラフ表示する．

トリガ条件設定画面では，どの M2M デバイスまたは M2M サーバからのセンサデータを対象にするかを選択（IN）し，トリガ条件が成立した時にどの M2M デバイスのアクチュエータを制御するかを選択（OUT）することができる．

温度に関するトリガ条件設定では，上限の温度と下限の温度を設定することができる．温度センサの値が，上限の温度を越えた場合または下限の温度を下回った場合に，デジタル出力によりアクチュエータである Arduino 内蔵 LED を警告として点灯する．

照度に関するトリガ条件設定では，照度に関するいろいろなトリガ条件を設定することができる．トリガ条件の判定に基づいて，アナログ（PWM）出力に基づいてアクチュエータであるブレッドボード上の外部 LED の輝度を制御する．例えば上限値を超えたり下限値を下回った場合にフルで点灯したり，照度データに応じた輝度で表示したりすることができる．

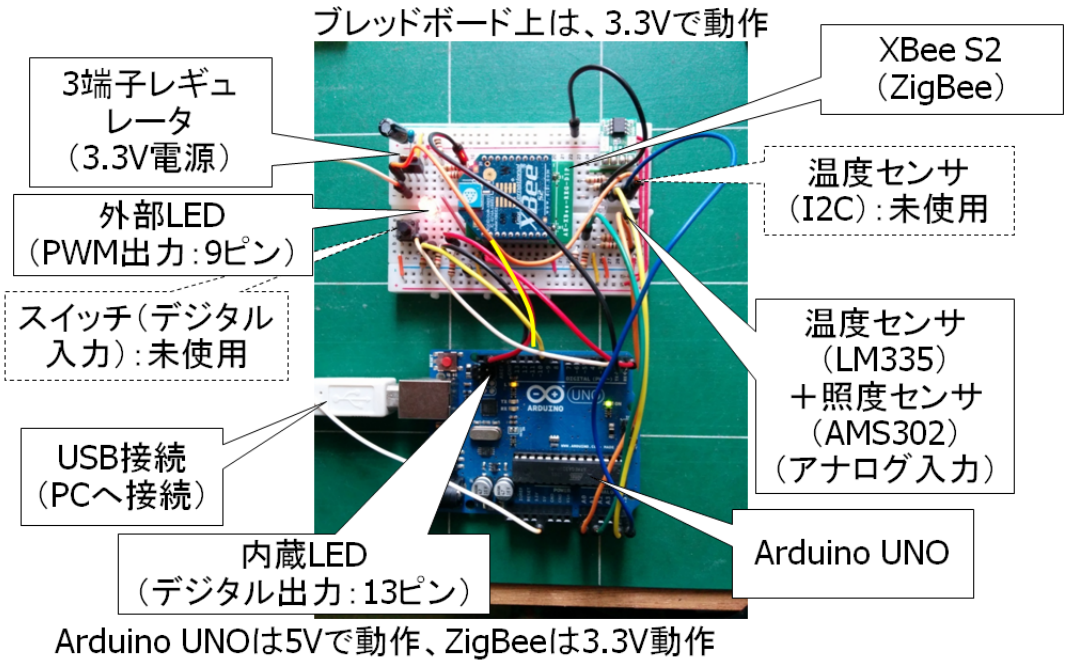


図 19　M2M デバイスの実装例

（３）実装例

M2M システム入門編の M2M デバイスの実装例を図 19 に示す．

M2M デバイスは，Arduino UNO 基板と，温度センサ，照度センサ，アクチュエータとしての外部 LED, ZigBee ネットワークを司る XBee モジュールなどを搭載したブレッドボードから構成されている．部品間や基板間の接続は，ジュンパー線により半田ゴテを使わずに行うことができる．

4.3. M2M/IoT システム応用編への展開

入門編では，はじめてこの分野を学ぶ人のために，必要となる技術をステップバイステップで，容易に理解しながら，システムを構築していけるように構成している．本格的にいろいろな分野で応用しようとする場合には，十分対応できないところがある．この入門編により M2M/IoT システムを理解することにより，図 20 に示すように，応用分野に応じて以下の部分をそれぞれ拡張して，新しい M2M/IoT システムを構築することが可能になる．

・M2M デバイス：応用目的に応じて，各種のセンサや各種アクチュエータを接続する．入門編ではセンサとのインタフェースにアナログ I/F のものを使ったが，アナログでのわずらわしさを避けるためにデジタル I/F（1-Wire，I2C，SPL）のセンサの利用も有効である．

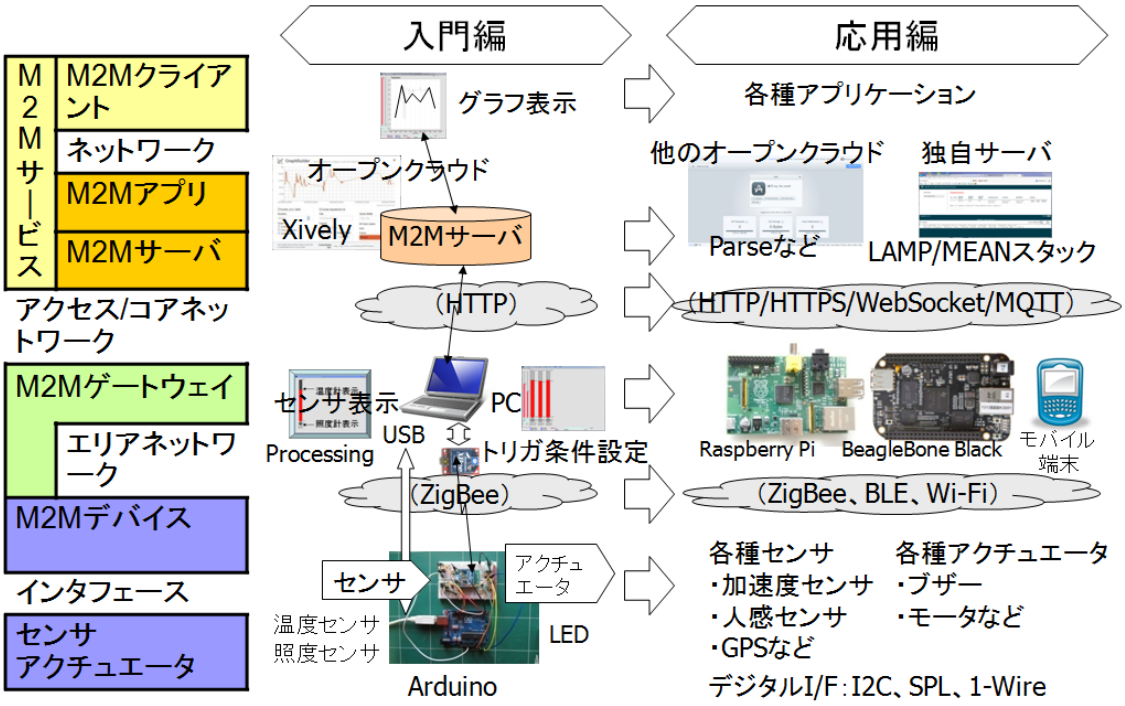


図 20　応用編への展開

・M2M ゲートウェイ：PC の代わりに Raspberry Pi/BeagleBone Black やモバイル機器（Android 端末など）を使用する．入門編で作成した Windows 上の Processing のプログラムは，Linux 上でも容易に動作させることができる．また，Processing には Android モードが設けられているので，Android への移植も可能と思われる．

・M2M サーバ：無料で使えるオープン M2M サーバは Xively 以外にも，Parse や AT&T 社の M2X など各種あり，目的に応じて使うことができる．また，独自のサーバとして，LAMP スタックや MEAN スタックを活用して M2M サーバを構築することもできる．この場合，PC の Windows 上に仮想マシン環境（例えば VirtualBox など）を立ち上げてゲスト OS である Linux 上で独自 M2M サーバを立ち上げて使用することも可能である [35]．

・エリアネットワーク：ZigBee 以外に，応用目的に応じて，Bluetooth，BLE，Wi-Fi を使用することが可能である．

・アクセス/コアネットワーク（プロトコル）：HTTP は基本的にクライアント・サーバ形式の片方向通信型であるが，応用目的に応じて，双方向通信型の WebSocket や，パブリッシュ/サブスクライブ型の MQTT を使用することが可能である．

５．まとめ

M2M/IoT システムを普及させるには，効率よく実現できるための M2M/IoT システム構築環境の整備・普及と技術者教育の充実が必要である．オープン化と標準化の成果を活用して，オープン環境による M2M/IoT システムをより充実させることにより，今後とも M2M/IoT の発展に寄与していく所存である．

なお，M2M/IoT システムの入門編については，サイバー大学 IT 総合学部のゼミナールの学生に使っていただき，改善を図ることができ，ご協力に感謝いたします．

参考文献

1) 辻秀一，澤本潤，清尾克彦，北上眞二：「M2M（Machine-to-Machine）技術の動向」，電気学会論文誌 C，vol. 133，No. 3，pp. 520-531(2013)
2) マイケル E. ポータ，ジェームズ E. ヘプルマン：「IoT 時代の競争戦略」，Harvard Business Review　2015 年 4 月号，ダイヤモンド社(2015)
3) 日経 BP ムック：「わるわかりインダストリー4.0　第 4 次産業革命」，日経 BP 社(2015)
4) 清尾克彦：「M2M（Machine to Machine）技術の動向と応用事例」，サイバー大学紀要第 5 号，pp. 1-22，(2013)
5) 清尾克彦，渡辺透，山本森樹，北上真二，三浦恵太：「M2M クラウドプロトタイプ開発の概要と取り組み事例」，平成 25 年度電気学会 C 部門北見大会. pp. 239-240(2012)
6) 清尾克彦:「オープン環境による M2M システム構築の動向」，平成 26 年電気学会 C 部門島根大会，pp. 505-508(2014)
7) 清尾克彦，辻秀一，三井浩康，中西美一：「スマート社会を支える M2M システム技術の最新動向-M2M システム構築技術-」，平成 27 年電気学会全国大会，Vol. 3，3-S14(11)-(14)(2015)
8) 川本大功：「オープンデザインにおけるインセンティブ設計としてのコモンズライセンスの提案~Fab Commons~！」，2011 年度 新保史生研究会 卒業プロジェクト 研究成果「Fab Commons」，http://halu9.com/file/fabcommonsv1.pdf
9) 総務省：「ユビキタスセンサーネットワーク技術に関する調査研究会－最終報告書概要」，平成 16 年 7 月
10) 無線センサネットワーク MOTE：http://www.xbow.jp/01products/index.html
11) Arduino：http://www.arduino.cc/
12) mbed：https://mbed.org/
13) Raspberry Pi：http://www.raspberrypi.org/
14) BeagleBone Black：http://beagleboard.org/BLACK
15) Edison：http://www.intel.co.jp/content/www/jp/ja/do-it-yourself/edison.html
16) Eclipse IoT プロジェクト：http://iot.eclipse.org/

17) 猿渡俊介，鈴木誠，大原壮太郎，南正輝，森川博之：「無線センサネットワークにおけるオペレーティングシステムの研究動向」，東京大学先端科学技術研究センター森川研究室技術研究報告書，No. 2008003(2009)

18) Muhammad Omer Farooq and Thomas Kunz：「Operating System for Wireless Sensor Networks:A Survey」，Sunsors 2011, 11, 5900-5930, http://www.mdpi.com/1424-8220/11/6/5900

19) Philip Levis：「Experiences from a Decade of TinyOS Development」，Proceeding OSDI'12 Proceedings of the 10th USENIX conference on Operating Systems Design and Implementation Pages 207-220, https://www.usenix.org/system/files/conference/osdi12/osdi12-final-183.pdf

20) Contiki：http://www.contiki-os.org/

21) Node.js：https://nodejs.org/

22) Processing：https://processing.org/

23) Xively：https://xively.com/

24) M2X：https://m2x.att.com/

25) Parse：https://parse.com/

26) IFTTT：https://ifttt.com/

27) Zapier：https://zapier.com/

28) Z. Shelby, S. Chakrabarti, and Cisco Systems：「Neighbor Discovery Optimization for Low Power and Lossy Networks (6LoWPAN) draft-ietf-6lowpan-nd-18」， IETF 6LoWPAN Working Group (2011)

29) Z. Shelby, K. Hartke, C. Bormann, and B. Frank：「Constrained Application Protocol (CoAP) draft-ietf-core-coap-08」，IETF CoRE Working Group (2011)

30) RFC6455−The WebSocket Protocol 日本語訳：http://www.hcn.zaq.ne.jp/___/WEB/RFC6455-ja.html

31) MQTT：http://mqtt.org/

32) IEEE1888(FIAP)：http://www.gutp.jp/fiap/

33) ZigBee Alliance：http://www.zigbee.org/

34) Wi-SUN Alliance：http://www.wi-sun.org/

35) 清尾克彦：「仮想実験環境による組込みシステム教育演習の取り組み」，サイバー大学 e ラーニング研究第 3 号，pp. 1-14(2014)