Vol.2013-MBL-65 No.23
Vol.2013-UBI-37 No.23
2013/3/14

小型無線センサノード用仮想マシン CILIX の適用事例

柳沢 豊[1]　須山 敬之[1]　寺田 努[2]　塚本 昌彦[2]

概要：CILIX は、小型無線センサノードで動作するアプリケーションを開発するための、.NET Framework の共通中間言語 (CIL) を実行できるプロセス仮想マシンである。C#, Visual Basic, C++/CLI, F# 等の .NET Framework で使用可能なプログラミング言語で記述されたプログラムを、8bit, RAM 4KB, ROM 32KB (Flash Memory 32KB) という極めて小さな計算リソースしかもたないデバイス上で実行することができる。本稿では、この仮想マシンの設計方針と実装の概略、およびこれをベースに用いて行った研究事例，実フィールドへの適用事例についてそれぞれ紹介する。

1. はじめに

センサネットワークは，広範囲の実世界で起きるさまざまな事象を，無線伝送技術と多数の小型センサを用いて取得すること目的としたシステムである [1]．近年では特に，ビッグデータ，M2M といったキーワードに注目されている中で，大量の実世界のデータを取得できる可能性のあるシステムとして期待が集まっている．センサネットワークによって取得した実世界のデータを解析する手法は，さまざまな分野のアプリケーションに適用できると考えられている [2]．

センサネットワークは，その用途に応じて求めらる機能，仕様が大きく変化するという特徴をもつ．たとえば，気候の観測なら，ハードウェアとして温度，湿度，雨量計，風速計といった複数のセンサを搭載し，一定時間間隔ごとにデータを取得して送信する機能が最低限必要である．一方で，傾斜地の土砂崩れを発見する場合は，振動を検知できる加速度計が必要であり，大きな振動を観測したときにデータを送信するという機能が求められる．

このような，運用現場でのさまざまな要求仕様に柔軟に答えるためには，使用するハードウェアをユーザが自由に選択でき，なおかつソフトウェアの開発が容易であることが必要である．この要件を満たす方法として，近年センサネットワークのアプリケーション開発を，プロセス仮想マシンを用いて行う方法が提案されてきた [3]．仮想マシンはハードウェアを抽象化することで，どのハードウェアにおいても同じ開発環境でアプリケーション開発を行うことを可能にする．また，既存のプログラミング言語に対応することで，ユーザがセンサデバイス独自の開発環境の使用法に習熟することなく，アプリケーションの開発を行うことが可能になる．

実際にこれまでに，センサノード向けのプロセス仮想マシンはいくつか提案されている [4], [5], [6]．これらの多くが Java か，既存言語に拡張を加えた独自の開発言語を使用する必要があった．これに対し，筆者らは開発言語が比較的自由に選択可能な，CIL 仮想マシンである CILIX を提供している．CILIX は，Microsoft .NET Framework で開発に用いることができる，C#, Visual Basic, C++/CLI, F# などの言語で記述したプログラムを，小型無線センサノード上で実行させることができる，プロセス仮想マシンである．ユーザは Windows 上の Visual Studio などの既存の開発環境を利用してアプリケーションを作成すれば，それをそのままセンサノード上で動かすことができる．

アプリケーションの開発を容易にし，なおかつ設置現場でのさまざまな要求に答えるため，CLI では規定されていない下記の機能を追加した．

- 無線通信による動的なプログラムの書き換え
- UDP 通信を用いた I/O の仮想化
- マルチスレッド制御
- センサノード間でのプログラムの遠隔呼び出し

本稿では，この CILIX の開発方針と特徴，および実装についての概略について述べる．さらに，実際に設置センサネットワークにおいて，これらの特徴を活かして専門家とインタラクティブに仕様の決定を行い，現場の要求に合致するセンサネットワークを構築できた事例について，研究例について 2 件，実用事例について 2 件，それぞれ紹介する．

[1] 日本電信電話株式会社 NTT コミュニケーション科学基礎研究所
[2] 神戸大学大学院工学研究科

表 1 小型無線デバイス向け OS，仮想マシンの比較（複数のマイコンやハードウェアに搭載されているものについては，最小のスペックのハードウェア，または最も一般的なハードウェアを載せている．）

名前	タイプ	開発言語	無線書き込み
TinyOS	OS	nesC，TinyScript (Java, C#)	○
Smart-Its	ライブラリ	C	○
PAVENET	OS	C	○ (IrDA)
Ubiquitous Chip	OS	ECA ルール	○
BTnodes	OS	C/C++	○ (Bluetooth)
Nano-RK	OS	C	×
SimpleRTJ	VM	Java	×
Darjeering	VM	Java	○
Sun Spot	VM	Java	○
Mate	VM	bytecode	○
ASVM	VM	TinyScript, motlle, and TinySQL	○
VM*	VM	Java	○
VAWS	VM	C	○ (IrDA)
NMF	VM+OS	C#，VB.NET，Managed C++ 等	○
Cilix	VM	C#，VB.NET，Managed C++ 等	○

表 2 小型無線デバイス向け OS，仮想マシンの比較 2

名前	マイコン・ハードウェア	CPU	ROM（B）	RAM（B）
TinyOS	Atmega128L（MICA Mote）	8bit	128K+EEPROM 4K	4K
Smart-Its	PIC18F6720	16bit	128K + EEPROM 1K	4K
PAVENET	PIC18F452 (U-cube)	16bit	32K + EEPROM 256B	4K
Ubiquitous Chip	PIC16F873	14bit	4K + EEPROM 128B	192
BTnodes	Atmega128	8bit	128K + EEPROM 4K	4K
Nano-RK	Atmega128（Fire Fly）	8bit	128K + EEPROM 4K + SROM (64K+180K)	4K
SimpleRTJ	Atmega128，ARM，H8	8/16/32 bit	14-36K	0.2K 以上
Darjeering	Atmega128，MPS430	8/16 bit	128K	4-10K
Sun Spot	ARM920T	32bit	4M	512K
Mate	Atmega128L（Mote）	8bit	128K+EEPROM 4K	4K
ASVM	Atmega128L（Mote）	8bit	128K+EEPROM 4K	4K
VM*	Atmega128L（Mote）	8bit	128K+EEPROM 4K + 外部 FLASH 512K	4K
VAWS	PIC18F452 （U-cube）	16bit	32K+EEPROM 256B	4K
NMF	ARM7/9/Cortex M3, ADI BlackFin	32bit	256K	64K
Cilix	ATmega128, MSP430, 他	8/16/32bit	32K	4K

2. 関連研究

以下では，小型センサノード向けに開発された，まず既存のミドルウェアについて述べ，CILIX の位置づけについて説明する．なお，各研究の分類法については文献 [7] の分類を参考にした．

小型のセンサノード向けのミドルウェアとしては，TinyOS[8]，Smart-Its[9]，PAVENET[10]，Ubiquitous Chip[11]，BTnodes[12]，Nano-RK[13] といったオペレーティングシステム（OS）と，Darjeeling[6], SimpleRTJ[5]，Sun Spot[14]，Maté[4]，ASVM[3]，VM*[15]，VAWS[16]，といった仮想マシンの大きく二つに分類できる．.NET Micro Framework (NMF) [17] はこれらの双方の機能を提供する．表 1 に一覧を示す．本章では，主に開発にかかる労力という観点からこれらの研究を比較する．

センサネットワークシステムの開発に用いられる OS としては，TinyOS が最もよく知られている．TinyOS 上でプログラミングすることで，省電力なセンサネットワークシステムを構築できるが，プログラムの開発には，nesC あるいは TinyScript という専用のプログラミング言語を使用する必要がある．Smart-Its では，センサデバイスを扱うためのライブラリ群が提供され，開発者は PIC 専用の開発環

境上で C 言語でプログラミングをする．Ubiquitous Chip はイベント駆動型の ECA ルールでデバイスの動作を記述するため，ごく小規模のソフトウェアであれば簡単に開発できるが，複雑なシステムを開発するには，イベント駆動型の記述形式に慣れる必要がある．その他に Bluetooth で通信するセンサノードの BTnode，ハードウェアリアルタイム処理をサポートできる PAVENET，リアルタイムのセンサデータ処理のための Nano-RK などがあるが，いずれも C 言語などの限られた言語でしか開発が行えない．さらに Nano-RK のように，システムの動作を確実にするために，無線での書き込みが行えない例もある．

センサネットワーク向けの仮想マシンは，仮想マシンが実行する中間言語の命令を独自に規定したものと，Java のバイトコードのように既存の命令群をそのまま実行できるように設計したものの二つにさらに分類できる．Maté[4] は TinyOS 上に実装された仮想マシンであり，独自の 24 の命令をもち，1 命令でセンサデータを取得したり，データを無線送信できる．独自の命令を用いると，このように仮想マシン上のプログラムの大きさを小さくできるという利点がある．その他にも，命令を独自に拡張できる ASVM，リアルタイム処理向けの VAWS といった仮想マシンがある．しかし，独自の命令セットを用いると，コンパイラを専用に開発する必要があるため，多くの言語で記述できる環境を実現することは難しい．

後者の例としては Java のバイトコードを実行できる SimpleRTJ，VM*，Sun Spot,Darjeering がある．SimpleRTJ は，Java のバイトコードとそれを実行するためのコードを合わせてコンパイルしてハードウェアに書き込むことで，Java のバイトコードを小型のデバイス上で実行できるようにするプロジェクトである．Darjeering はプログラムの書き換えや，メモリ使用量削減のための工夫がなされ，SimpleRTJ よりも柔軟な開発が可能である．いずれも，これまでに 8bit，16bit，32bit の数種類のハードウェア上で動作する実績がある．しかし，JVM であることから，基本的として使用できる言語は Java のみである．

VM*は Java のバイトコードをそのまま実行可能な MOTE のハードウェア上に実装された仮想マシンである．バイトコード書き込み時にデバイス上でプログラムを実行するには不要な部分を削り，対応するモジュールを選択して仮想マシンに書き込む．プログラムは無線で書き換え可能であり，必要なモジュールを自動的に追加することも可能である．動的にモジュールを追加できる機構は，数多く用意された機能から必要なものを選択する際には有効であるが，小型軽量な仮想マシンを実現しようとする上ではオーバーヘッドが増える点も考慮する必要がある．Sun Spot も Java のバイトコードをそのまま実行できるが，センサノードで使用されるマイコンとしては高機能な ARM（32 ビット，ROM:4M バイト，RAM:512K バイト）を用いているため，デバイスの小型化，省電力化が難しくい．CILIX とは目指しているターゲットが異なる．

これらに対して CILIX は，開発者が好みの言語を用いて，複数のプラットフォームに対応した移植性の高いソフトウェアを開発できるという利点がある．さらに，.NET Micro Framework が使用できない 8bit/16bit CPU のデバイス上でも CIL プログラムを動作させることができる．そのため，CILIX を用いることで，開発者は従来より多くのデバイスをセンサノードとして選択できる．

3. CILIX の要求仕様

3.1 観測現場で求められる要件

センサネットワークは長年研究され，同時に多数の実証実験が行われてきたが，商用化にいたる例は米国での広域農地での気候観測など，その例は限られている．

センサネットワークは，個々のノードの測定精度が低く，なおかつ通信路の信頼性が低いために，精度が低いデータを少量しか送れないというデメリットをもつ．その代わり，廉価なデバイスを大量にばらまくことで，広範囲のデータを比較的廉価に取得できるという点が，メリットであると筆者らは考える．精度の低さは，ノードの精度を高めるよりも，数を増やすことで相互に補完して精度を高めるというアプローチをとることが想定されていた [1].

一般に，製品として販売されるセンサノードは，高精度，高信頼性，高機能性を担保するために，高額化する傾向にある．センサネットワークの用途を特化せず，汎用的なノードを作ろうとすると，高額かつ頑丈なものを作る必要性が生じる．こうした製品は，大規模かつ広範囲でのモニタリングを行うか，もしくは観測価値が非常に高い現場でのモニタリングを想定して開発が行われることが多い．そのため，小規模な地域での観測や，経済的余裕が少ない現場での観測といった目的においては，高い初期費用の用意ができず導入が難しいというのが現状である．仮に導入したとしても，導入コストが期待する効用に見合わず，継続的に運用することができないケースも多い．小規模な環境への導入を進めるためには，現場で必要とされる最小限の機能をもつ低価格のセンサノードを作成する必要がある．

しかし，センサネットワークで観測しようとする現場において，現場状況に精通した専門家であっても，どのようなデータを取れば十分であるかを事前に判断できないという問題がある．専門家は，観測した事象についてはきわめて詳細な知識を持っている一方で，その事象を観測するために必要なセンサの性能，個数，配置箇所，観測期間といった情報を持っていない．そもそも，そのような詳細なデータを取得した経験がないが故に，センサネットワークを用いて詳細に観測したいのである，故に，その情報を最初から保持していることを期待すること自体に無理がある．センサネットワークを設置する現場においては，センサネッ

トワークの開発者にも現場の専門家にも，観測前には要求仕様が分からないことは当然であるともいえる．

そのため，専門家が必要とするデータが十分収集できるセンサネットワークを構築するためには，現場でのインタラクティブなシステム開発が求められる．すなわち，データを取る実験を繰り返すと同時に，どのようなデータの取り方をすれば目的の事象が観測できるのかということを，専門家とシステム開発者とが議論を繰り返して進めるというプロセスが必要になる．

このプロセスの過程では，最終的なシステムの効用がはっきりしないまま，データ取得実験を繰り返す必要がある．その過程では，システムが廉価であること，設置位置の変更等が容易であること，システムの拡張性の高さ，開発の容易さといった要件が求められる．一方でこの段階では，高い精度や信頼性，処理の高速さといった要件は必要とされない．このプロセスが終了し，センサネットワークに求められる要求仕様が確定した段階で，改めて必要最小限の精度，信頼性，処理の高速性といった要件を満たす，最小コストで作成可能なシステムを設計すればよいと考えられる．

以上のことより，CILIX は廉価なセンサノード上で動作させて，現地でのデータ取得実験を繰り返し，センサネットワークに必要な仕様を確定させる用途に向いた，テストベッド用仮想マシンとして設計を行った．まず，開発者のアプリケーション作成コストを抑えるため，既存の開発環境を利用して開発できる中間言語に対応した．具体的には，.NET Framework で使用できる言語で記述されたプログラムを，そのまま実行可能な CIL 仮想マシンとして設計した．次に，さまざまな現場での要求仕様に答えられるように，センサネットワークのアプリケーションで使用される頻度が高い機能を，仮想マシンの組み込み機能として実装した．さらに，廉価なデバイスで実行可能なように，メモリ使用量を極力抑える設計とした．その一方で，処理の高速性や，.NET Framework クラスライブラリのもつ機能への対応という点については，大きく犠牲にしている．これらについては，テストベッドとして利用する上で，前述の他の要件を阻害しない範囲で，必要最小限の性能が得られるよう実装している．

3.2 互換性

仮想マシンを導入する最大のメリットは，ハードウェアを抽象化することで，ハードウェアのアーキテクチャの差異を気にすることなく，ソフトウェアの開発ができることである．そして，既存の仮想マシンを移植する場合においては，既存の開発環境やソフトウェアとの互換性が大きなメリットのひとつになる．

このメリットを享受できるようにするために，CILIX が既存のコンパイラで生成された CIL プログラム (exe ファイル) を，ほぼそのまま実行できるように設計した．ただし，計算資源の少ないデバイス上で CLR や Mono との完全な互換性をもつランタイムシステムを実装することはほとんど不可能である．そのため，センサノードで使用することがほぼないと考えられる機能を省いた，CLI のサブセットとして CILIX を設計した．これにより，省メモリ化と互換性の両立を図った．

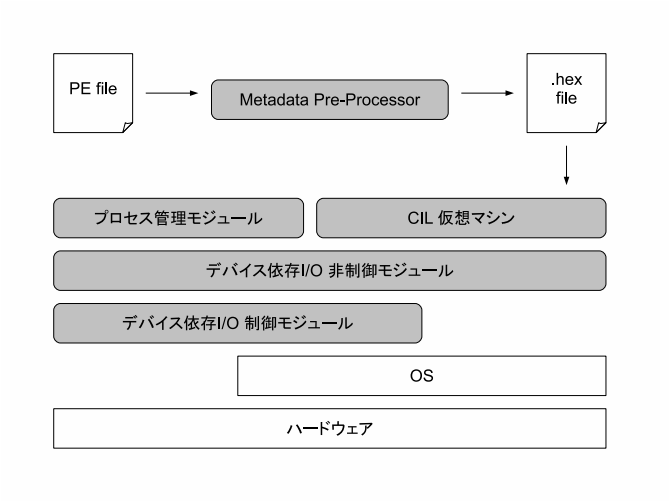


図 1 CILIX の構造

3.3 利便性

CIL 仮想マシンは，仮想化された CPU として設計されている．そのため，数値演算機能とメモリ間のデータ転送機能，プログラムの実行制御機能といった，一般の 8bit/16bit CPU がもつ機能と同等の基本的な機能しか，仕様上は規定されていない．仮想マシンの仕様にない機能の多くは，仮想マシン上に実装する CIL プログラムのライブラリとして実装することで，実現することができる．しかし，でバイズへの依存度が高いことから，CLI 上でのライブラリとしての実装が難しく，ランタイムシステムの機能の一部として実装されていることが望ましい機能がある．具体的には以下の四つの機能である．

(1) 実行する CIL プログラムを変更する機能（プログラムの動的な書き換え）
(2) I/O の基本的な制御機能（センサ，無線通信チップへのアクセス）
(3) マルチスレッド処理機能
(4) センサノード上のプログラムを他ノードから遠隔実行する機能

これらの機能は，実際にセンサノードを運用していく中で，必要とされた機能である．これらの機能は，既存の他のランタイムシステムでも採用されていることが多い．このため，CILIX をこれらの機能をランタイムシステム内に組み込みモジュールとしてもつ形で設計した．これらの機能のうち，マルチスレッド処理に関しては .NET Framework

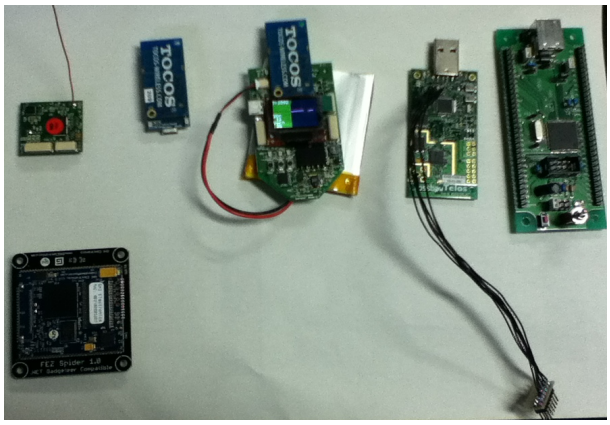


図 2 CILIX が動作するデバイス例

のライブラリと互換性のあるクラスの形で提供する．また，I/O アクセスについては I/O ポートを仮想的な UDP ポートにマッピングし，UDP 通信のクラスライブラリを利用する形のインタフェースとした．これによりユーザは，特殊なライブラリの追加を必要とせず，.NET Framework のクラスライブラリのみを用いて，センサノードのプログラムの作成を行うことができる．

3.4 省メモリ

8bit/16bit の CPU は, 搭載するメモリ (EEPROM/Flash Memory/RAM) の量が増えると，それがそのまま CPU の価格の上昇や実装面積の拡大につながる．つまり，仮想マシンが使用するメモリの量を抑えることで，コストが低く小さなデバイス上で仮想マシンを動すことができるようになる．

そこで筆者らは，仮想マシンの動作するメモリの搭載量の下限を，EEPROM (Flash Memory) 32KB, RAM 4KB とした．このスペックのデバイスは，2013 年の実売価格で $ 5 以下で調達が可能である．これを超えるの性能の 8bit/16bit の CPU は，32bit の CPU の価格との差異が少なくなる．そのような価格帯では，8bit/16bit の CPU を採用するメリットは薄い．また，2 章で挙げた他の仮想マシンや OS と比べても十分に小さいものであり，既存の多くのデバイスで動作させることが可能である．これらの点を考慮して，上記のメモリ量を動作要件の下限とした．

4. CILIX の実装

CLI の仕様は ECMA-335 [18] により規定されている．CILIX は ECMA-335 に準拠し，CIL 中間言語を実行することができる．ただし, Visual Studio や Mono コンパイラの生成する exe ファイルには，CIL 中間言語や Metadata table のすべてが含まれるわけではない．筆者らは，従前のコンパイラが出力する exe ファイルを精査し，コンパイラが出力しないオペレーションコードや Metadata table および Signature については実装を行っていない．これにより，CILIX の実行バイナリサイズを 32KB 程度のサイズにまで抑えることができたている．以下，CILIX の構造の概要と，CILIX を使ったアプリケーション開発方法について，その概略を説明する．

4.1 CILIX の構造

CILIX のランタイムシステムは，次の四つのモジュールで構成される．

- 仮想マシンモジュール：CIL プログラムをメモリから読み出して実行するモジュール
- プロセス管理モジュール：仮想マシンの起動や停止を行う．また CLI の仕様外の機能として，メモリ上のプログラムの書き換えを行う機能も含む．
- デバイス非依存 I/O 制御モジュール：I/O 制御に必要な機能のうち，デバイスに依存しない機能を提供するモジュール．主に文字列変換処理を行う．
- デバイス依存 I/O 制御モジュール：I/O 制御に必要な機能のうち，デバイス毎に書き換える必要のある機能を含むモジュール．

図 1 に CILIX の構造を示す．CILIX は，これらのモジュールと，Metadata Pre-Processor (MPP) とよぶプログラムで構成される．MPP は実行ファイル (PE ファイル) に前処理を行い，プログラムデータを圧縮することで，使用するプログラムメモリの領域を削減できる．MPP は PE ファイル (.exe) から CILIX の実行ファイル (.hex) へ変換を行う．

4.2 CILIX によるアプリケーション開発

CILIX を用いたアプリケーション開発は，Visual Studio (Express Edition を含む) および Mono のコンパイを用いて行うことができる．現在動作を確認している開発言語は，C#, Visual Basic, C++/CLI, J++, および F# である．Visual Studio のバージョンは，2005, 2008, 2010, 2012 でそれぞれ動作を確認している[*1]．Mono では C# のみ動作を確認している．

対応している MPU (デバイス) としては，ATMega128L (8bit)，MSP430 (16bit)，TWE001 (32bit)，ARM7 (32) がある．図 2 に示すデバイスは，上の段の左から順に，MSP430，TWE001(小型版)，TWE001(有機 EL 版)，TelosB(MSP430), ATMega128L である．左下は .NET Micro Framework が動作する .NET Gadgeteer FEZ Spider Mainboard (ARM7) を，サイズの比較のために掲載している..

CILIX は，それぞれのデバイスで動作可能なバイナリファイルとして提供される．EEPROM と Flash Memory とを持つデバイスでは，CILIX を EEPROM に書き込ん

[*1] Express Edition を含む

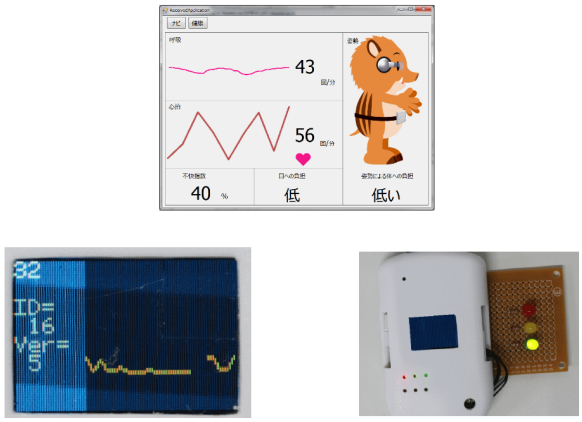


図 3 ヘルスケアシステム実験への適用例

でおき，Flash 側に C# 等で記述したプログラムを MPP で変換した後の .hex ファイルを書き込む．この MPP は Windows の実行ファイルとして提供している．MPS430 と TWE001 に関しては，データの圧縮と Flash への書き込みを同時に行うプログラムを用意している．

なお，Flash に書き込む方法以外に，SD カードの先頭領域に書き込んでおき，SD カードからプログラムを起動させる方法と，無線経由でプログラムを書き込む方法を提供している．現在，MPS430 と TWE001 で作成したデバイスについては，無線経由でプログラムを書き換えるための Windows 側の転送用プログラムを用意している．

5. 適用事例

CILIX と無線小型センサノードを利用して行った 2 件の研究事例と，2 件の実フィールドでの実験事例について紹介する．

5.1 ウェアラブルコンピューティング環境への適用

寺田ら [19]，および武田ら [20] は，ウェアラブルコンピューティング環境下において，ウェアラブルコンピュータとデータの送受信を行う機器間の，データフローのディペンダビリティを担保する技術について研究を行った．

ウェアラブルコンピュータは，屋外や危険地での作業に利用されることもあり，PC やそれに接続されている機器の故障する確率が高い．作業中に PC や機器が故障すると，作業が中断されたり，場合によっては情報不足による事故が発生する可能性がある．これを防ぐため，PC に故障が発生した時には，一時的に入出力デバイス間でデータを直接送受信できるようにし，データの表示が停止しない仕組みを提案した [19]．入出力デバイス間での直接のデータ送受信を可能とする方法ために，表示デバイスや入力デバイスに無線通信デバイスを搭載し，そのデバイス上で動作するファームウェアを使って，データフローの制御を行えるようにした．このファームウェアを動作させるミドルウェアとして CILIX を用いた．

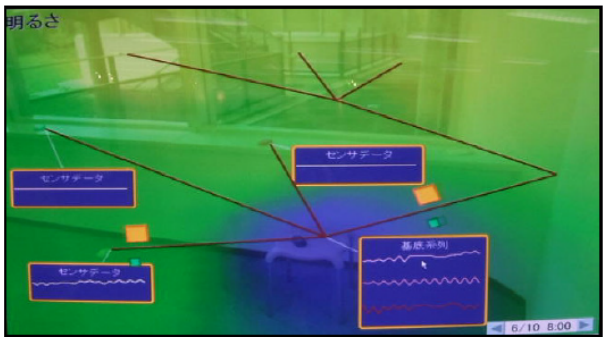
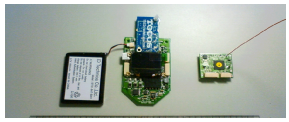


図 4 データ圧縮実験への適用例

CPU	TWE-001 (32bit RISC)
RAM	128KB
ROM	128KB (flash memory)
無線通信	IEEE802.15.4/Zigbee
搭載センサ/デバイス	温度センサ (誤差 ±0.5℃) 湿度センサ (誤差 ±5%) 照度センサ 三軸加速度センサ 64 x 64 有機EL (オプション)
外部I/O	シリアルコネクタ x2



外寸	400 x 900 x 5 mm (基板)
重量	32g (電池込み)
電池容量	830mAh (リチウムイオン)
通信距離	30-40m (見通し，実測)
通信速度	250Kbps

図 5 使用したセンサノード例 (TWE001 版)

CILIX を利用することで，アーキテクチャの異なる各デバイス用のファームウェアを，Visual Studio を利用して統一的に開発することが可能となった．また，ウェアラブル PC 上で動作する OS の多くは Windows であることから，PC 上で動作するアプリケーションの開発と同じ開発環境でファームウェアの開発を行えることも，開発の容易さを高める要因となっている，さらに，CILIX の無線通信によるプログラム書き換え機能を利用することで，ファームウェアを必要に応じて遠隔でアップデートすることも可能である．この機能を用いることで，実験結果をもとにファームウェアを頻繁にアップデートするというプロセスを効率的に進めることができた．

図 3 は，構築したシステムをヘルスケアシステムに適用した例を示したものである．ランニング中の人間の健康状態をモニタリングして，図の上部のような表示内容を HMD に出力するアプリケーションである．仮にランニング中に PC が停止しても，心拍センサや呼吸センサなどから表示デバイスに直接データが無線経由で送信されるため，PC の再起動中でも表示が途切れることはない．また HMD が故障した場合は，すぐに予備の小型有機 EL つきデバイスにデータが転送され，そのデバイス上でデータを確認することができる．システムの復旧や交換後は，再び HMD などのデバイスにデータを転送することで，表示を復旧させることができる．

5.2 データ圧縮実験への適用

CILIX を用い，センサデータの圧縮手法に関する研究の評価実験を行った事例について紹介する．圧縮手法として

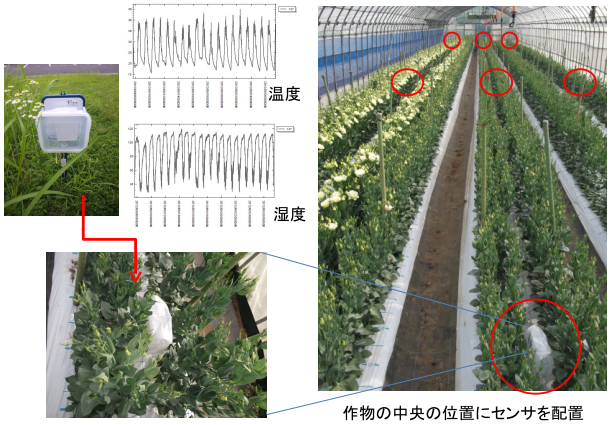


図 6 トルコギキョウ栽培環境のモニタリング

は，岸野らが提案する特異値分解と階層型ネットワーク構造を利用した非可逆データ圧縮方式 [21] と，柳沢らが提案する LDPC を用いた可逆データ圧縮方式がある [22]．センサデバイスとしては，MSP430 および TWE001 の二種類のデバイスを用いた．TWE001 デバイスの仕様と概観について図 5 に示す．

階層型ネットワークを利用したデータ圧縮方式では，データを中継するノードに集まったデータに類似する特徴を特異値文化によって抽出し，特徴を強く示す基底データのみをサーバに送る方法で圧縮を実現する．LDPC を用いる方法では，各ノードが観測するデータの相関性を利用し，単独のノードで観測したデータを圧縮するよりも，高い圧縮率で可逆圧縮する技術である．これらの技術を実データで試験するために，CILIX を搭載したセンサデバイスを用いた．図 4 は，室内外に配置したセンサノードで温度を計測し，マルチホップ型センサネットワークを構築して，データの収集を行っているデモ実験の様子を示したものである．

これらの手法によるデータの圧縮率は，観測するデータの特性によって大きく変化することが予想された．そのため，性能を十分に評価するためには，室内外のさまざまな環境下でデータを長時間にわたって取得する実験を行う必要があった．このように環境を変えて実験を繰り返すケースにおいて，CILIX によるプログラムの開発の容易さや，プログラム書き換えコストが低いことが，実験コストの低減に役立った．

5.3 施設園芸での栽培環境モニタリング事例

個人農家において，ビニールハウス内の栽培環境が均一であるかを調査する目的で，ハウス内の環境モニタリング実験を，長野県佐久穂町との共同実験として行った．栽培品目は，切花として商業価値が高いトルコギキョウである．トルコギキョウは，昼間の日射量，日光の強さ，温度の日較差などによって，作柄に大きな差異が生じる品目であり，農家の工夫次第で収益が大きく増減する作物のひとつである．

本実験では，ハウス内の温度，湿度，照度の環境が均一であるかを調べることを目的に実験を行った．ハウスは 45m × 6m の面積をもち，作物は 5 列に並べて栽培されている．ハウスの長辺が南北方向に向いており，一日の日照量がハウスの東側，西側で異なっていることから，温度分布が各地点で異なるのではないか，ということは農家の経験的に知られていた．

この実験では，当初センサの配置個数や頻度，必要な電池量等のパラメータが分からず，10 個のセンサをハウス内の南北の辺に各 3 つと，中央に 3 つ，それぞれ地上 10cm の位置に配置してデータの観測を行った．その後，データの内容を見ながら農家と配置法について検討を行い，4 回にわたって配置変更やプログラムのアップデートを繰りかえした．その結果，20 個のセンサを使い，地上 10cm と 1.5m の位置に配置して温度と照度の計測を行うことで，日射量や温度の差異を取得できることがわかった．最終的に，農家が予想していた日射量の差が，ハウス内で 1 日の間に最大 3 時間もの差があることが分かった．栽培品目によって必要とされる日射量が異なることから，来年度の栽培時には今回の観測結果をもとに，品目の配置を変更することを予定している．本実験に関しては，次回の栽培時に配置変更によって作物の作柄の改善が生じたかを検証する予定である．

図 6 は，農家のビニールハウス内に CILIX を搭載し，温度，湿度，照度のセンサを接続した TWE001 デバイスを，防水型ケースに入れて設置した様子を示している．

5.4 作業環境の温湿度調査の事例

節電対策が行われた企業のオフィス環境が，作業に適した温湿度から大きく逸脱していないかを調べるための，実フィールド実験を行った事例について紹介する．本実験は，次の三つの目的で行われた．最初の実験は，冬季の節電による室内の温度低下により，室内の一部で想定温度 (19 度) を下回っている地点がないかを確認することを目的とした．次の実験は，夏季の節電によって室内の温度に大きく偏りが生じているときの，サーキュレータ導入効果の有無を調査する目的で行われた．最後の実験は，通年における実験室の温度調節を最適化するための，空調の設定方法や作業設備の設置位置，扉の開閉の調整方法の検討を行うことを目的に行った．

本実験でも，当初センサを設置する位置や測定頻度などの条件が分からなかったため，20 平方メートル程度の部屋の中に 50 個のセンサを密に配置し，10 秒間隔でデータを取得する形で実験を開始した．センサは部屋の外側の廊下や，窓の外（屋外）にも配置した．同じ地点であっても高さによる温度の変化がある可能性を考え，床上 5cm, 1m, 2m といったように複数の高さに配置した．この状態で実験を開始し，観測されるデータを見ながら配置の変更や，

頻度の変更（プログラムの書き換えによる頻度変更）を行い，最適化を図った．最終的にはセンサ 10 個を要所にのみ配置し，1 分間隔でデータを取得しておき，それらのセンサで観測される温度が 19 度以下とならないように空調の温度設定をすることで，室内の温度が 19 度以下になることを防ぐことができるようになった．他の実験でもほぼ同様の手順で，作業環境測定の専門家の意見を聞きながらシステムの要件の決定を行った．

この過程で，センサの配置変更を頻繁に行い，50 個のセンサノードのプログラムの書き換えを行った．しかし，電池駆動の小型センサを用いたため配置変更はきわめて容易であった．また，プログラムの書き換えについても無線経由での書き換え機能により，手間を大きく削減できた．最終的に配置位置や測定頻度が求められた後に，センサデバイスを大型化し，電池も大容量なものに変更することで，コストと必要な性能がつりあうセンサノードを設計することができた．

6. おわりに

本稿では，センサノード用の仮想マシンである CILIX の特徴と，他研究，実フィールドへの適用事例について紹介した．現在，本稿で紹介した事例以外に，他大学との共同研究 4 件，企業との共同プロジェクト 2 件（海外企業を含む）を推進中である．これらの適用事例に関しては，2013 年 6 月開催予定の NTT コミュニケーション科学基礎研究所オープンハウスにて発表する予定である．

謝辞 本研究の一部は，科学研究費補助金基盤 (A)(20240009) の支援によるものである．ここに記して謝意を表す．

参考文献

[1] Warneke, B., Last, M., Liebowitz, B. and Pister, K.: Smart Dust: communicating with a cubic-millimeter computer, *Computer*, Vol. 34, No. 1, pp. 44 –51 (2001).
[2] Kuorilehto, M., Hännikäinen, M. and Hämäläinen, T. D.: A survey of application distribution in wireless sensor networks, *EURASIP J. Wirel. Commun. Netw.*, Vol. 2005, pp. 774–788 (2005).
[3] Levis, P., Gay, D. and Culler, D.: Active sensor networks, *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation - Volume 2*, NSDI'05, Berkeley, CA, USA, USENIX Association, pp. 343–356 (2005).
[4] Levis, P. and Culler, D.: Maté: a tiny virtual machine for sensor networks, *SIGPLAN Not.*, Vol. 37, pp. 85–95 (2002).
[5] RTJ-Computing: The Simple Real Time JAVA, http://www.rtjcom.com/.
[6] Brouwers, N., Langendoen, K. and Corke, P.: Darjeeling, a feature-rich VM for the resource poor, *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, New York, NY, USA, ACM, pp. 169–182 (2009).
[7] 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，技術報告 2008003，森川研究室 (2008).
[8] Hill, J., Szewczyk, R., Woo, A., Hollar, S., Culler, D. and Pister, K.: System architecture directions for networked sensors, *SIGPLAN Not.*, Vol. 35, pp. 93–104 (2000).
[9] Beigl, M. and Gellersen, H.: Smart-Its: An Embedded Platform for Smart Objects, *In Proc. Smart Objects Conference (SOC 2003)*, pp. 15–17 (2003).
[10] Saruwatari, S., Kashima, T., Minami, M., Morikawa, H. and Aoyama, T.: PAVENET: A Hardware and Software Framework for Wireless Sensor Networks, *Transaction of the Society of Instrument and Control Engineers*, Vol. E-S-1, No. 1, pp. 74–84 (2005).
[11] Terada, T., Tsukamoto, M., Hayakawa, K., Yoshihisa, T., Kishino, Y., Kashitani, A. and Nishio, S.: Ubiquitous Chip: A Rule-Based I/O Control Device for Ubiquitous Computing, *Pervasive* (Ferscha, A. and Mattern, F., eds.), Lecture Notes in Computer Science, Vol. 3001, Springer Berlin / Heidelberg, pp. 238–253 (2004).
[12] BTnode-Project: BTnodes - A Distributed Environment for Prototyping Ad Hoc Networks, http://www.btnode.ethz.ch/.
[13] Eswaran, A., Rowe, A. and Rajkumar, R.: Nano-RK: An Energy-Aware Resource-Centric RTOS for Sensor Networks, *Real-Time Systems Symposium, IEEE International*, Vol. 0, pp. 256–265 (2005).
[14] Oracle: Sun SPOT, http://jp.sun.com/products/software/-sunspot/.
[15] Koshy, J. and Pandey, R.: VMSTAR: synthesizing scalable runtime environments for sensor networks, *Proceedings of the 3rd international conference on Embedded networked sensor systems*, SenSys '05, New York, NY, USA, ACM, pp. 243–254 (2005).
[16] 鈴木誠，猿渡俊介， 南正輝，森川博之：無線センサノードのためのハードリアルタイム保証が可能な仮想マシン，電子情報通信学会論文誌 B， Vol. 92, No. 1, pp. 130–139 (2009).
[17] Villar, N., Scott, J., Hodges, S., Hammil, K. and Miller, C.: .NET Gadgeteer: A Platform for Custom Devices, *Pervasive*, pp. 216–233 (2012).
[18] ECMA: Standard ECMA-335: Common Language Infrastructure (CLI), http://www.ecma-international.org/publications/standards/Ecma-335.htm.
[19] 寺田 努，柳沢 豊，塚本昌彦，武田誠二，岸野泰恵，須山敬之：装着デバイス間の直接通信によるウェアラブルコンピューティングの信頼性確保手法について，情報処理学会第 32 回ユビキタスコンピューティングシステム研究会, 2011-UBI-32(8) (2011).
[20] 武田誠二，岸野泰恵，柳沢 豊，須山敬之，寺田 努，塚本昌彦：ウェアラブルコンピューティングのディペンダビリティを確保する情報変換機構をもつ装着型入出力デバイスの設計と実装，情報処理学会第 32 回ユビキタスコンピューティングシステム研究会, 2011-UBI-32(9) (2011).
[21] 岸野泰恵，櫻井保志，前川卓也，須山敬之：階層的センサネットワークのための特異値分解を用いたデータ圧縮手法，情報処理学会第 31 回ユビキタスコンピューティングシステム研究会, 2011-UBI-31(2) (2011).
[22] 柳沢 豊，岸野泰恵，前川卓也，須山敬之：相関のあるデータを観測する無線センサネットワークのためのデータ集約手法，情報処理学会第 32 回ユビキタスコンピューティングシステム研究会, 2011-UBI-32(4) (2011).