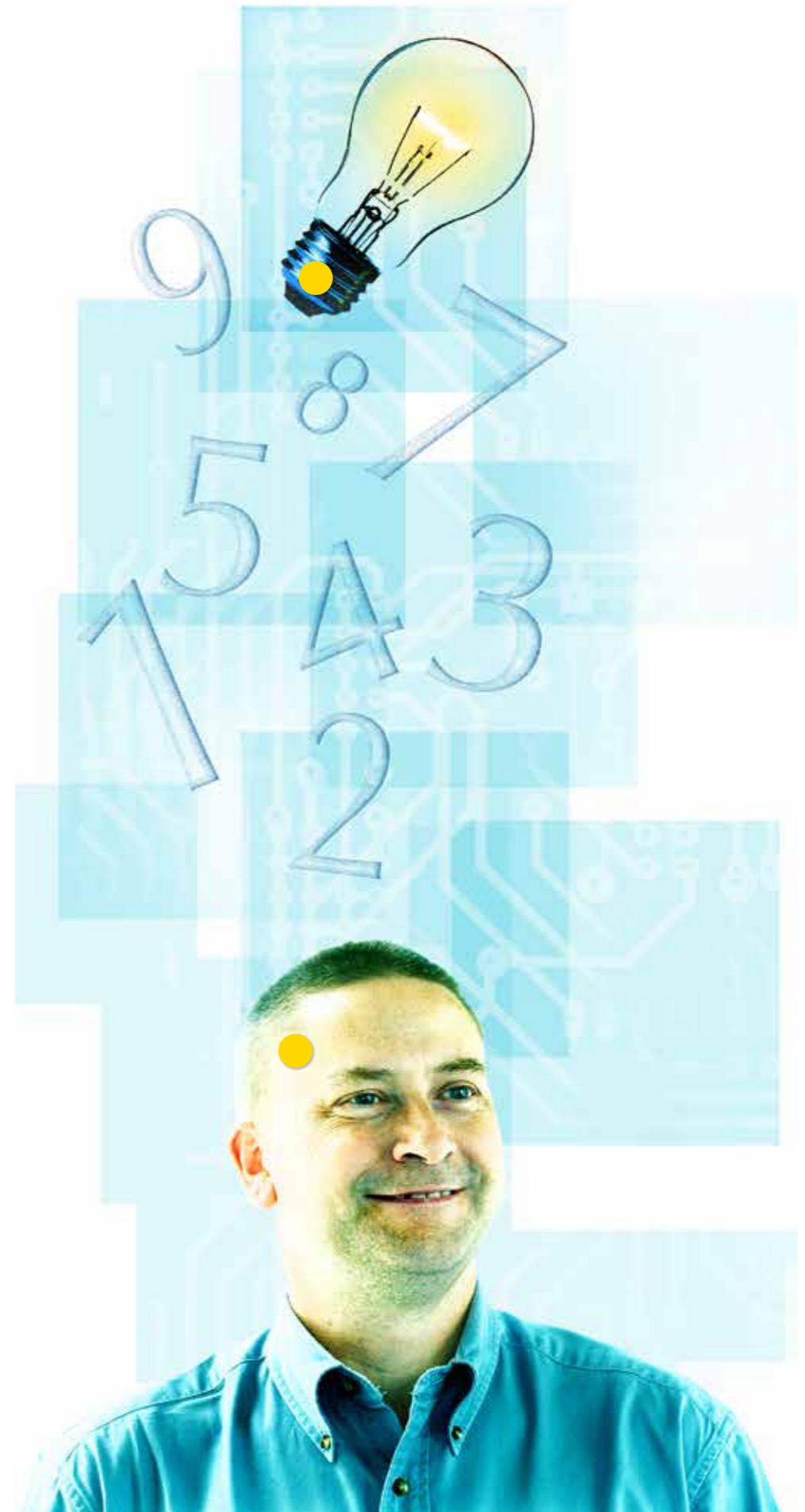


(組み込み)

組み込み型 アプリケーション でJAVAを使う 10の理由

なぜJavaがInternet of Things
(モノのインターネット)に
最適な言語なのか
SIMON RITTER

照明用電球から自動車にいたるまでのあらゆるものが内蔵「インテリジェンス」とともにネットワーク化される、Internet of Things(モノのインターネット)という話は何年も前からありました。ついに今、ムーアの法則と規模の経済性が結び付き、Internet of Things(モノのインターネット)が現実のものとなりつつあります。たとえばPhilipsは先頃、インターネット対応型の電球をAppleストアから発売すると発表しました。このようなネットワーク化されたプログラム可能なデバイスが本当に使えるようになるには、データを集め、処理し、送信するアプリケーションが必要です。従来、組み込み型システムのコードはアセンブラかC言語で書かれてきましたが、プログラム可能なネット対応型デバイスの爆発的増加に対応するにはもっと優れた言語が必要です。それがJavaなのです。組み込み型デバイスにとってJavaが適切な言語である理由は何でしょうか。ここでその理由のトップ10を紹介します(今回参考とさせていただいたブログ記事の執筆者Roger Brinkley氏に感謝します)。



画像:NICHOLAS PAVKOVIC、写真:BOB ADLER



1 車輪はすでにある。再発明し続けるのはやめよう

Java SE 7には、コレクションから並行処理、ネットワーキングまで、すべてを網羅する約4,000の標準クラス・ライブラリがあります。標準APIの膨大なコレクションがあるということは、組み込み型アプリケーションのために標準的機能を書き直す時間が大幅に削減されるということです。Oracle Java ME Embeddedなど比較的フットプリントが小さな部類のJavaプラットフォームは、デフォルトでは標準クラスを多く備えていませんが、使用できる機能は豊富です。Oracle Java ME Embedded APIの多くが組み込み型システムのニーズに的を絞っています。たとえばデバイスAPIは、シリアル通信(SPI: Serial Peripheral Interface)やI2C(Inter-Integrated Circuit)などの低レベル・プロトコルを使用するための標準化された方法を備えています。JavaのAPIパワーはプラットフォームの一部として利用できるクラスに留まりません。Javaは幅

ワン・フォー・オール

Javaでのコードの移植は大きな問題ではありません。アプリケーションで使用するAPIに一切変更がない限り、それは単に既存のクラスやJARファイルの再デプロイメントに過ぎません。

広く使われている言語であるため、たとえばセンサーや、シリアル・ポート通信、ZigBee APIなど、考えられるほとんどすべてのことに対応できるライブラリがすでに開発されています。こうしたライブラリの多くはオープン・ソース化されていて、新しい組み込み型アプリケーションに容易に組み込むことができます。

2 「セグメンテーション違反」(メモリアクセスエラー) を避ける

Javaの強みの1つであり、Javaがプログラミング言語として広く使われている理由の1つに、アプリケーション・コードが堅牢であることが挙げられます。CやC++などの言語では、メモリを参照するのに明示的なポインタを使用します。Javaでは、すべてのオブジェクト参照は、アプリケーション・コードで操作することが不可能な暗黙的ポインタとなります。そのため、誤ったポインタ計算によるバッファのオーバーランやメモリ・アクセス違反などの潜在的問題を回避できます。こうした状況では、アプリケーションが突然停止することもあります。組み込み型システムでは、エラー・メッセージが表示されるスクリーンなどのデバイスがないため、こうした種類のエラーの追跡はかなり難しい場合があります。

3 1つのプラットフォームがすべてを執行してくれる

デスクトップ・マシンやサーバーとは異なり、組み込み型システムは、アーキテクチャ、リソース、およびOS機能に関してはかなりばらつきがあります。組み込み型システムのアップデートは、プロセッサやOSをそれ以前の製品リリースからの大幅な変更を行うよい機会でもあります。アセンブラやC言語で書かれたアプリケーションを新しいプラットフォームに移すのは時間やコストのかかるプロセスであり、エラーが発生しやすくなります。そもそもJavaは「Write once, run anywhere」(一度書けばどこでも実行できる)プログラミング言語です。Javaでのコードの移植は大きな問題

ではありません。アプリケーションで使用するAPIに一切変更がない限り、それは単に既存のクラスやJavaアーカイブ(JAR)ファイルの再デプロイメントに過ぎません。Javaの新しい(つまり、より高性能な)バージョンに移行する場合、必要なのは単なる再コンパイルだけです。

4 雑用は仮想マシンに任せる

Java Virtual Machine(JVM)は、ネイティブにコンパイルされたコードでは得られない多くのメリットを組み込み型アプリケーション開発者に提供します。

メモリ管理はJVMがすべて自動的に実行します。メモリは、`malloc`などのライブラリ関数に対して明示的コールを使用するのではなく、オブジェクトをインスタンス化することによって割り当てられます。開発者はオブジェクト参照を追跡する必要も、メモリの割当を明示的に解除する必要もありません。そうした処理はガベージ・コレクタがすべて実行してくれます。ガベージ・コレクタは、再起動を必要としない、メモリに制約のある環境でアプリケーションが長時間動作しなければならない組み込み型システムに、多大な影響を及ぼす可能性があるメモリ・リークの可能性を大幅に低下させることができます。

アプリケーションに関する並行処理サポートもJVMが対応します。Javaには最初から、異なる実行スレッドを作成し、同期化する機能が組み込まれています。(最近では少なくなってきましたが)JVMがデプロイされる組み込み型プラットフォームがマルチスレッドをダイレクトにサポートしていない場合でも、グリーン・スレッドのコンセプトによって機能をエミュ

ネイティブは避けよう
ネイティブ命令ではなく仮想マシンを使用することで、パフォーマンスが損なわれるという誤った考えを持つ人たちがいます。

レートできます。

ネイティブ命令ではなく仮想マシン(VM)を使用することで、パフォーマンスが損なわれるという誤った考えを持つ人たちがいます。約20年に及ぶJVM開発史の中で、これらの問題はもはや重大なものではなくなりました。特定の状況下では、アプリケーション動作中にしか得られない的確な情報を活用すれば、ネイティブにコンパイルされたコードよりもパフォーマンスを向上させることさえ可能です。

5 組み込み型プラットフォーム選択の自由

組み込み型システムは一般に、車内エンターテインメントの提供方法にしても、産業プロセスの一部におけるpH値のモニタリング方法にしても、特定の問題を解決するために設計されています。そのため、システム・ハードウェアは開発対象となるソリューションに合わせて作られるため、設計者には、新しいデスクトップ・システムやラップトップ・システムを設計するよりもずっと多くの選択肢があります。

FreescaleやBroadcomなど、多くの組み込み型チップ・メーカーでは、ARMなどの企業が保有するアーキテクチャを使って、プロセッサやチップ上のシステム(SoC)を構築します。そのため、プロセッサやSoCの標準について、一部に違いが生じますが、多くの場合は、浮動小数点処理やバス・アーキテクチャといった点に関してはほとんど違いがありません。Java

はこうした相違を排除することで、開発を大幅に簡易化します。

Intel x86やMIPS、さらには(組み込み型デバイスでもっとも広く使われているOSである)Linuxで動作するARM v5、v6、v7の各アーキテクチャに関しては、Oracle Java Embedded Clientのリファレンス実装が用意されています。Intel x86とIBM Power e500v2システムおよびe600システム、またLinuxで動作するARM v5、v6、v7の各アーキテクチャには、Oracle Java SE Embeddedがあります。そのため、組み込み型システムの設計者は使用するプラットフォームに関して幅広い選択肢を持つことができます。

6 1995年以來の技術の進化

Javaはもともと、組み込み型デバイスであったStar7 PDA用アプリケーションを作成するために、1990年代初めに開発されたものです。Javaが汎用コンピューティング・プラットフォームとして初めて登場した1995年当時、平均的なパソコンのスペックは、RAMが8MBで、プロセッサの動作周波数は200MHz以下でした。現在でも、フルJava SEランタイムがWindows XPで動作するのに最低限必要とするRAMは64MBに過ぎません。今、典型的ローエンド組み込み型システムでも、上記以上のスペックを備えています。

Javaはそもそものスタート段階から、リソースに制約のある環境での使用を目的に開発されたため、今日の組み込み型システムのニーズに最適なものとなっています。Oracle Java ME Embeddedが必要とするROMはわずか350KB、RAMは130KBに過ぎ

ません(単位がメガバイトではなくキロバイトである点に注意)。標準のフル構成でもROMが1.5MB、RAMが700KBです。Oracle Java SE Embeddedの方はもっとパワーのあるシステム用ですが、それでもROMは39MB、RAMは32MBで済みます。遅いクロック速度でもうまく動作するJavaプラットフォームは、バッテリー駆動デバイスの充電間隔を広げたい場合にも有効であることを意味します。リソースが限られている場合でも、Javaは仕事を遂行できるのです。

7 「道具をくれ、そうすれば仕事を片付けてみせよう」*

コード開発にはツールが必要です。ツールが優れたものであればあるほど、仕事を簡単に終わらせ、短時間で完成させることができます。

今日、製品の成否にとって市場に投入するまでの時間は、まさに機能と同じように非常に重要な意味を持ちます。組み込み型システム用にアセンブラやCのコードを書くためのツールは、Makeなど取扱いが難しいツールによって制御されるテキスト・エディタやコマンドライン・ツール・チェーンを使用する傾向にあります。こうしたツールは役に立ちますが、開発者の生産性は大きく制限されます。Java開発者であれば、市販だけでなく、無料オープン・ソースによる数多くの統合開発環境(IDE)から選ぶことができます。NetBeansやEclipseといったツールは、コード作成作業を非常に簡単にし、エラーの発生を抑えます。キーボード入力中の自動コード補完や構文チェック、統合ドキュメンテーションといった機能はすべて、作業を迅速化し、コードの堅牢性

Java Embedded: 開発を始めよう

Java Magazineテクノロジー編集者のTori Wieldtが、Oracle Java ME Embeddedの最新情報についてOracle Technical Business DevelopmentのKevin Smithにインタビューしました。

Java Magazine: パートナーにとって、Java MEの組み込み分野における新展開はありましたか。

Smith: JavaOne San Francisco 2012で、Oracle Java ME Embeddedの発表を行いました。Java ME Embeddedは、マシンツーマシン(M2M)市場およびInternet of Things(モノのインターネット)のニーズに対応する新しいJavaプラットフォームであることから、大きな注目を集めました。この件に関しては、CinterionおよびQualcommとの密接な協力の下、作業を進めてきました。Oracle Java ME Embeddedには、JSR 228、Information Module Profile—Next Generationが採用されています。Java MEチームは長年にわたり、PC、ワイヤレス・ハンドセット、タブレット、ラップトップによって、人をインターネットに接続するために活動してきました。また、Oracle Java ME Embeddedは、モノをインターネットにつなげるプラットフォームとして機能しています。どのアナリスト・レポートを読んだのかにもよりますが、2020年までには、300億~600億のモノがインターネット接続されるようになります。今、既存のJavaスキルを使って、こうした多数のデバイスのプログラミングが可能で

Java Magazine: Qualcomm Orionボードとはどのようなものでしょうか。

Smith: Orionボードとは、QualcommのM2MおよびInternet of Thingsを対象とした開発者ボードです。2012年初めに、Qualcommが新しいM2Mビジネス・ユニットを発足させて以降、私たちはOracle Java ME EmbeddedでOrionをサポートするため、Qualcommと共同で作業を進めてきました。

Java Magazine: OrionがJava開発者にとって重要な理由は何でしょうか。

Smith: Orionではチップ上のシステム(SoC)がドーター・カードを介してボードに取り付けられている点が、大半の開発者ボ

ードと異なります。Qualcommは、機能やモデムが異なる多彩なSoCコンビネーションを用意しており、開発者は1つのボードで数多くの異なるデバイス・タイプに対応することが可能になります。マザーボードには、WiFi a/b/g/n、SDカード・スロット、4個のDB9コネクタ、5個のLED、JTAG、USIM、GPIO、SPI、I2Cが装着されています。バッテリーを搭載してボードを動作させることもできます。さらには、温度センサーや光センサー、GPS、3D加速度計も備えているため、開発者はこのボードで、コンテナ管理、ビル・オートメーション、自動車や、組み込み型デバイスの新世代製品に関するソリューションのプロトタイプを作成できます。NetBeans IDEにはJava ME Software Development Kitが統合されているため、ボード機能のエミュレーションも可能です。その結果、開発者はOEM(相手先ブランド名製造メーカー)のプロトタイプを待つことなく、ただちにプラットフォームを利用できるため、製品を市場に投入するまでの時間が短縮されると考えられます。驚くべきことです。さらに開発者にとっての良いニュースがあります。このようなプロトタイプ・ボードの価格は通常、ソリューション1件につき約5,000ドルであるのに対し、Orionボードの場合は約500ドルで、スケーラビリティのあるプラットフォームを入手できます。



組み込み分野におけるパートナーおよびJava開発者にとっての最新情報を説明するOracle Technical Business DevelopmentのKevin Smith

を高める上で有用です。リッチ・クライアント・プラットフォーム(RCP)として、これらのツールは拡張が容易であり、アプリケーション開発における個別のニーズに対応することが可能です。デスクトップ・ベースのIDEで使用できるターゲット・ハードウェア用エミュレータの構築によって、開発時間を大幅に短縮できます。

8

開発とデプロイメントを別のマシンで

多くの場合、組み込み型システムは本質的に、デスクトップ・システムやラップトップ・システムとは似ていません。ヘッドレスと呼ばれることがしばしばありますが、それはディスプレイがないことを意味しています。このディスプレイがないことが、コードのコンパイルとデプロイを別々のマシンで行うという点において、ソフトウェア開発にさらなる問題と複雑さをもたらします。

Javaでは仮想マシンを使用することから、どこでコードをコンパイルするかは問題になりません。つまり、複雑なクロスコンパイル用ツール・チェーンをセットアップする必要がありません。好みのJava IDEを使って、使い慣れたデスクトップでコードを書き、それをターゲット・デバイスのクラス・ファイルかJARファイルにコピーするだけです。開発者の仕事を容易にするため、大半のIDEが、ネットワーク接続によるリモート・デバッグ・コンセプトをサポートしています。コードが開発者の意図しない動作をした場合は、IDEのデバッグ機能を使用することで、何が問題かを確認できます。こうする方が、print文を使うより、はるかに短時間ですみます。

* WINSTON CHURCHILL, BBC RADIO BROADCAST, FEBRUARY 9, 1941

9 900万人の開発者が間違えることはあり得ない

調査結果にもよりますが、Javaでのプログラミング方法を知っている開発者は世界中で900万人に達します。ほとんどすべての大学で、オブジェクト指向プログラミングの基礎用教育言語にJavaを使っていますが、それは同時に、Java開発者の数の増加が見込まれることを意味します。

Javaコードの開発を行うと、膨大な数の頼れるプログラマの知識を利用することができるのです。JVMとクラス・ライブラリによって組み込み型システム・コード開発の複雑さが大幅に軽減されたことから、この分野で豊富な経験を持つ人材を探し回る必要がなくなりました。特定の分野に特有の知識が必要な状況であっても、そうした状況に対応できるJavaの知識を持った開発者が数多く存在します。

10 デバイスからデータ・センターへ

組み込み型システムが真の価値を持つためには、他の組み込み型システムとだけでなく、データの集積、分析、検索が行われている場所ともネットワークで結ばれる必要があります。データ・センターはまさにそうした場所であり、多様で多数のソースからの大量データを記録し、処理するとともに、有用な情報を抽出するデータ・マイニングも行っています。エンタープライズ・アプリケーション開発用でもっとも人気が高いプラットフォームは何でしょうか。Javaです。組み込み型デバイスで、ビッグ・データ処理用にデータ・センターで、システム全体の表示/制御用デスクトップでJavaを

使用することは、完全なソリューションをより簡潔に、迅速に、低コストで構築できることを意味します。

お分かりだと思いますが、Javaは組み込み型システム用コード開発で多数の利点をもたらします。低コスト、幅広いプラットフォーム選択肢、優れた技能を持つ既存の多くの開発者による、Internet of Thingsの構築を躊躇する理由はありません。

</article>

豊富な人材
JVMとクラス・ライブラリ
によって組み込み型システム・
コード開発の複雑さが大幅
に軽減されたことから、豊富
な経験を持つ人材を探し
回る必要がなくなりました。

Simon Ritter はオラクルの Java テクノロジー・エバンジェリストです。Java technology の開発部門およびコンサルティング部門で働いた経験があります。現在はコア Java プラットフォームとクライアント・アプリケーション向けの Java を中心に手がけています。