次世代ICカードの本命，Java Card™

（株）東芝 デジタルメディア機器社
酒井 高彦

ICカードとは ―

現在使用されているキャッシュカードあるいはクレジットカードは，プラスチックカードの上に埋め込んだ磁気ストライプに情報を記憶させている．これらの磁気ストライプカードと同じ大きさのカードにICチップを埋め込み，ICチップの中のメモリに情報を記憶させたカードがICカードである．埋め込まれるチップは，メモリのみで構成されるものからCPUを搭載したものまでさまざまである．CPUを搭載したものは，インテリジェンスを兼ね備えることからスマートカードと称されることもある．本稿では，CPU搭載型のICカードについて記載する．

ICカードを利用した実証実験は，1980年前半から日本国内でも始められていたが，限定された地域での実験にとどまり，広がることがなかった．磁気ストライプカードは，記録面が裸であり外部から直接アクセス可能である点から，CPUを介して記録媒体にアクセスするICカードに比べて安全面で劣る．また，記録容量の点でも劣る．これらの点が近年評価され，電子マネー実証実験等にはICカードが使われ始めて注目を集めている．国内で使用されているICカードの大半がCPU搭載型である．

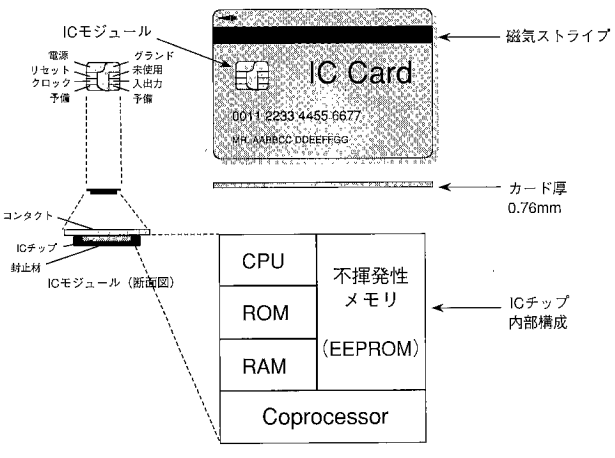

図-1 ICカードの構造

現在のほとんどのICカードは，1つのICチップのみが埋め込まれている．図-1は，ICカードの構造と高機能型のICカード用チップの構成を示している．チップ内部は主にCPUと3種のメモリおよび暗号処理専用回路のコプロセッサから構成されている．CPUは現在は8bitのものが主流であるが，16bitまたは32bitを搭載しているものもある．メモリは，作業領域であるRAMとCPU制御コードを保持するROMおよびデータの書換えが可能な不揮発性メモリの3種で構成されたものが一般的である．不揮発性メモリ技術は，現在はEEPROM技術が主流であるが，フラッシュメモリや強誘電体素子技術の利用も検討されている．コプロセッサは，公開鍵暗号処理を実施するものが一般的である．べき乗剰余型暗号の処理を高速に実施するための積和演算回路が現在の主流である．

Java Cardへの期待 ―

ICカードの上にJava動作環境を構築し，Java言語で書かれたアプリケーションをこの上で動作可能にしたICカードがJava Cardである．Sun Microsystems社がこの仕様を策定してホームページにて公開している．最新版は，本年3月にリリースされたJava Card 2.1である．図-2は東芝が試作したJava Cardサンプルの外観写真である．

既存のICカードは，ROMにアプリケーションをあらかじめ書き込んでいる．このため，一度ICカード用チップを作ってしまうとアプリケーションを変更することができない．市場からは，ICカードを利用した多彩なサービスの構築が要望され始めている．このためカード発行後でもカード上のアプリケーションの追加・削除が求められている．Java Cardは，その大きな特長の1つとしてこの機能を実現している．

現在のICカードの市場の中心である欧州では，ほとんどのカードがメモリ容量の制約のためにカードに1つのアプリケーションだけを搭載している．近年のHWの進歩

により複数のアプリケーションを1枚のカードに搭載することが可能になってきている．このため欧州市場を中心として安全なマルチアプリケーション環境をカード上で構築することが求められてきている．すなわち，カードに搭載された複数のアプリケーションがお互いに干渉し合わない環境の提供である．Java動作環境が実現しているセキュリティ機能をICカードシステムとして提供するJava Cardはこの要望に応えられると期待されている．

Java Card以前のカードの大半は，搭載するアプリケーションの開発がカードメーカに任されている．開発に当たっては，ICカード用チップに搭載しているCPUの機械語あるいは独自の開発言語を使用している．ICカード市場の拡大に伴い，カードメーカではすべての開発が請け負えず，開発者の不足が指摘され始めてきている．Java Cardは，Java言語でICカード用アプリケーションを開発できる．このため現在のJavaプログラマがそのまま開発者となり得るので，開発者不足が解消できる．また，開発言語の統一が図れるため，メーカ別の開発が不要となること，カードメーカ以外でもアプリケーション開発が可能になることなど，数々の利点が出てくる．

Java Cardの構造 ──

図-3は，ICカードのメモリの使い方について既存カードとJava Cardを比較したものである．両者共にRAM領域は，ICカードが動作中の作業領域として利用している．両者の主な違いは，ROMとEEPROMの使い方である．

既存のICカードは，カードと端末の通信・カード内のメモリ管理等の資源管理を実施するCOS（Card Operating System）とアプリケーションをROMに搭載している．EEPROM領域は，データ格納領域として利用されている．

Java Cardでは，ROMはCOSとJava実行環境の搭載に使用される．Javaは，オブジェクト指向言語のためアプリケーションの中でデータが一緒に扱われる．Java Cardの場合，データとその処理手続きが一緒になったカードに搭載されるアプリケーションをアプレットと称している．アプレットは，EEPROMに置かれるため，追加・削除が可能となっている．アプレットは，ROMに搭載されてJava実行環境内のVM（Virtual Machine）上で動作する．VMは，Java Card仕様で規定されているため，Java言語で記載されたICカード用アプリケーションは，Java実行環境が構築されていればいずれのカードでも動作可能である．

ICカードと端末およびICカードアプリケーション ──

ICカードの国際標準としてISO/IEC 7816で一般的な

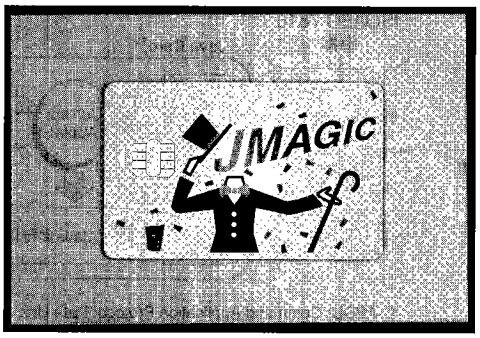

図-2　東芝Java Card™サンプルJMAGIC™の外観

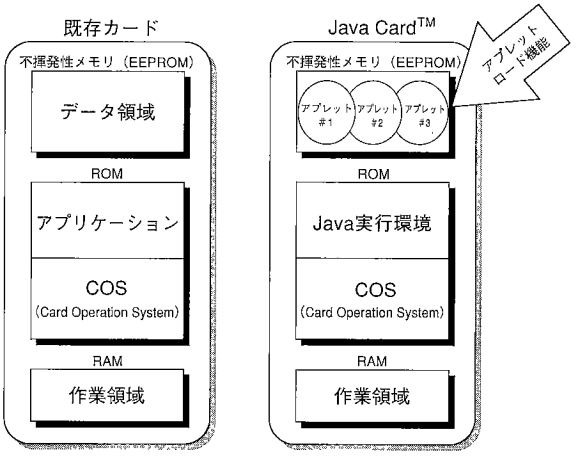

図-3　既存カードとJava Card™の違い

仕様が定められている．仕様は，ICカードの物理特性・電気特性から始まりアプリケーション層まで及んでいる．この仕様の中で端末とICカード間の通信についても規定されている．通信規定の最上位層は，現時点では，端末とICカードが交換するメッセージのフォーマットであり，APDU（Application Protocol Data Unit）と称されている．図-4は，端末とICカード間のメッセージ交換を示している．

基本的には，ICカードは端末の指示に従って動作する．C-APDU（Command-APDU）と称されるフォーマットに従った指示が端末からICカードに送られた後，ICカードはその指示内容を解釈・実行してその結果をR-APDU（Response-APDU）と称されるフォーマットに従って端末側に返している．このC-APDUとR-APDUのやりとりの繰り返しが，通常の端末とカード間の処理である．ICカードアプリケーションは，基本的には端末から送られてくるコマンド（指示）を解釈実行する一種のコマンド処理の集まりと捉えることができる．

現在のJava Card仕様の端末・カード間の通信はこのISO/IEC 7816のサポートのみが規定されている．また，既存カードの大半がこの規定を守っている．このためJava Card用に新たに端末側を改造する必要はない．既存のアプリケーションをJava言語で書き直してJava Card

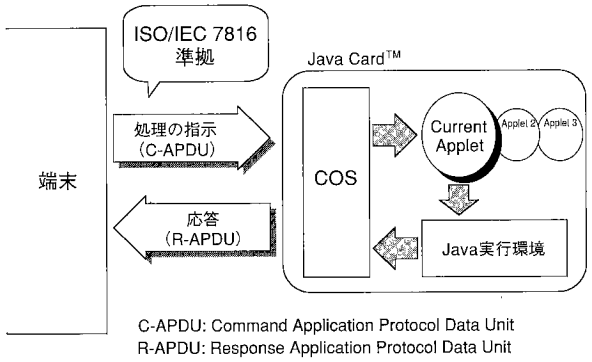

図-4　端末とICカード

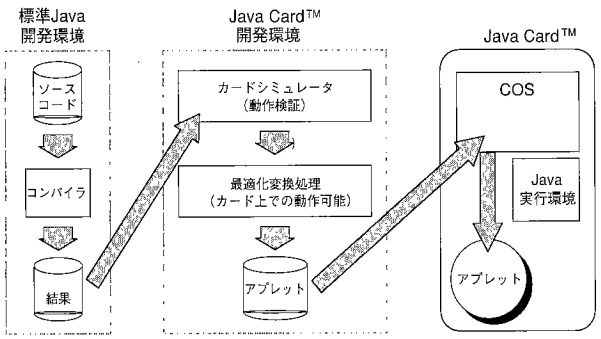

図-5　アプリケーション開発からJava Card™への搭載まで

に搭載すれば，既存の端末でそのまま動作することが可能である．

ICカード用チップ仕様 ―

Java動作環境をICカードに構築する上での最大の問題点は，端末に比べHW資源があまりに貧弱なことである．限定されたHW資源上にJava動作環境を構築するためにJava仕様をICカード向けに限定したJava Card仕様が規定されているということができる．ICカード用チップのHW資源の代表例を以下に示す．

HW	資源
CPU	8bit（～32bit）
RAM	512bit（～4KB程度）
ROM	20KB（～32KB程度）
EEPROM	8KB（～64KB程度）

括弧内は，最新のHW拡張の例であるが，いまだ一般的ではない．パーソナルコンピュータが標準的に32bit CPU，32MB RAM，ICカードのROM，EEPROMに相当するプログラム・データ保存領域として1GB HDDを備えていることに比べると格段の違いがある．

図-1にICカード用チップの外部インタフェースを記述している．ICカード用チップは，外部からクロックと電源が供給されている．ICカードの場合外部クロック入力は，通常1～5MHzである．最近のパーソナルコンピュータのCPUのように500MHz程度の動作周波数とは比べものにならない．電源は，最近のものは3～5V動作である．外部から供給されるものをそのまま使用しているため，電源断に対する耐力は皆無であり，電源断と同時に動作が停止する．したがって，内部データの更新に当たってはこうしたHW特性をよく理解した上での工夫が要求される．

Java Cardシステム ―

図-5にJava Card用アプリケーションの開発からICカードへの搭載までの流れの概略を示している．前述した通り，ICカードのHW資源はいまだ貧弱であり通常のJavaをそのまま動作させることはできない．したがって，Javaコンパイラの出力をJava Card用に最適化してその結果をICカードに搭載している．Java Cardでは，ICカードのJava Card開発環境（Off-Cardと称される）とJava Card（Off-Cardに対してOn-Cardと称される）を組み合わせたJava Cardシステム全体でICカード上でのJava動作環境を実現している．

Javaは逐次実行型の言語であり，実行時に数多くの検査を実施している．この動作検証用のコード量は膨大なため，現状のICカード用チップのメモリには搭載することができない．このためICカード側は，Java Card開発環境側での検証を前提として検査機能の搭載を必要最小限に抑えている．Java Card開発環境側では，シミュレータを用意し，アプリケーション・デバッグ時に動作検証を実施するようにしている．カードに搭載されるアプリケーションが必要な動作検証を実施してきたことを証明するためJava Card開発環境からJava Cardへアプリケーションを送付する際に署名の添付などを実施する機能が準備されている．

Java Cardでは，標準Javaに対してICカードで動作するための制約を設定している．型・クラスなどの言語仕様上の制限，バイトコード・スタックサイズなどの動作環境（仮想マシン仕様）の使用制限，APIの制限とICカード用の拡張など制約事項は多岐に渡る．これらのほとんどもJava Card開発環境で検査される．

Java Card上のアプリケーションの動作 ―

図-4に示したようにICカードの動作は，通常すべて端

末側からの指示に基づいて実施される．Java Cardの場合，端末側からの指示は大きく2つに分けられる．ICカード上のアプリケーションが処理するものとこれ以外の処理で主にJava動作環境が処理するものである．後者の例は，アプリケーションのダウンロードやアプリケーションの選択である．

Java Cardは複数のアプリケーション搭載は実現しているが，実行されるアプリケーションは常に1つである．すなわち，常に1つのアプリケーションのみが実行可能状態にあり，他のアプリケーションを動作させるには切り換えが必要となる．このアプリケーションの切り換えが，前述のアプリケーションの選択である．端末側からアプリケーションが実行すべき指示が送られてくるとそのまま選択されているアプリケーションに送られ，そこで解釈・実行される．

Java Cardの安全性 ──

Java Cardの安全性で特に問題となるのは次の2点といえる．Java Card開発環境とJava Cardの組合せで実現しているシステムにおいて，2つのブロックの間でアプリケーションが送信する際に外部から手が加えられる可能性とJava Cardに複数搭載されたアプリケーションが互いに不当に干渉し合う可能性である．

前者については，「Java Cardシステム」の章でもすでに記載したが，Java Card開発環境において署名を付与することでJava Cardに送られるアプリケーションの正当性を確認できるようにしている．また，送られるアプリケーションの盗用と改ざんを防ぐために送られるアプリケーションを暗号化できるようにしている．このためJava Card側に署名確認と復号化の機能が用意されている．

前章に記載したようにJava Cardでは複数のアプリケーションを搭載してこれを切り換えながら使用している．したがって，複数のアプリケーションが同じ動作する際の相互干渉は存在しない．主に動作中のアプリケーションからの不当なデータアクセスの排除が求められる．Java言語はデータアクセスに際してポインタを使用せずに名前参照方式を採用している．このためポインタの計算誤りによる不正アクセスは言語仕様上不可能である．また，第三者のサブルーチンを使用する際には，その実体をそのまま使用せずに別に新たな実体を作成して使用しているため他人のデータを直接アクセスすることができない．

その他のJava Cardの特長 ──

「ICカード用チップ仕様」の章で記載したようにICカードは外部からの電源・クロックの供給が切断されると直ちに動作が停止する．たとえば，動作中のICカードを端末から引き抜くと同様な事態となる．アプリケーションのダウンロードの時とアプリケーションが書き込み動作中にこのような事態が発生する際に対処する機能が用意されている．前者に対しては，カードの起動時に搭載途中のアプリケーションの有無を確かめ，存在する場合にはそのデータをすべて削除することで対応している．後者については，いわゆるコミットバッファの用意で対処している．コミットバッファはアプリケーションの指定により使用される．一度コミットバッファを介する書き込みにより前述の事態が発生しても書き込み以前あるいは書き込み以後の正しい状態が保証される．こちらもカード起動時にコミットバッファにデータが存在する際には，自動的にそのデータを書き込むべき領域に反映する．ただし，「ICカード用チップ仕様」の章に記載したように全体として大きなメモリ領域がないため，コミットバッファ用の領域もまだかなり小さい．

前章で記載したアプリケーション相互不干渉とは相反するが，データ共有機能が用意されている．

Java Card 2.1および今後の展望 ──

Java Card 2.1仕様は，1999年3月にリリースされた．図-5のJava Card開発環境が生成するJava Cardに搭載するアプリケーションのファイルフォーマットの規定がJava Card 2.1と以前の版との大きな違いである．以前の仕様に基づくカードでは，Java Card開発メーカ各社がそれぞれ独自のフォーマットを規定したため互換性がなかった．互換性は，Javaソースコードレベルで実現されていた．Java Card 2.1では，Java Cardに搭載するアプリケーションのファイルフォーマットまで互換性レベルを進めている．

ICカード用チップの進化，Java Card利用拡大による市場からの要望拡大，従来にないプログラム可能なカード用の新しい運用規程の模索など発展途上のJava Cardに関する検討項目はいまだ数多くあり，今後の課題である．

参考文献
1) Java Card 2.0 Applicartion Programming Interfaces, Revision 1.0 Final, (C) 1997 Sun Microsystems, Inc. (Oct. 1997).
2) Java Card 2.0 Language Subset and Virtual Machine Specification, Revision 1.0 Final, (C) 1997 Sun Microsystems, Inc. (Oct. 1997).
3) Java Card 2.0 Programming Concepts, Revision 1.0 Final, (C) 1997 Sun Microsystems, Inc. (Oct. 1997).
4) Java Card 2.1 Virtual Machine Specification, Revision 1.0 Final, (C) 1999 Sun Microsystems, Inc. (Mar. 1999).
5) Java Card 2.1 Application Programming Interface, Revision 1.0 Final, (C) 1999 Sun Microsystems, Inc. (Mar. 1999).
6) Java Card 2.1 Runtime Environment 2.1 Specification, Revision 1.0 Final, (C) 1999 Sun Microsystems, Inc. (Mar. 1999).

(平成11年7月19日受付)