非接触型 IC カード「FeliCa」の現状とプロジェクト演習の成果

西岡 茂樹
奈良産業大学情報学部

概 要

非接触型 IC カードの一つである SONY の FeliCa を「非接触型 IC タグ(RFID)」と「接触型 IC カード」という二つの流れの融合として捉え、その位置づけを明確にすると共に、FeliCa の技術要素を調査分析した後、FeliCa を利用するアプリケーション開発に取り組んだ。その成果の一例を報告する。

１．はじめに

我が国では 2001 年に JR 東日本が「Suica」の運用を開始したのを皮切りに、非接触型 IC カードと非接触型 IC タグは、社会に急速に普及し始めた。その中で、Suica に採用された SONY の非接触型 IC カード「FeliCa」は、日本の各地の交通カードにも次々に採用され、事実上の標準となった。

これらの動向を受けて、奈良産業大学情報学部において、2007 年度からプロジェクト演習のテーマの一つとして「FeliCa の調査研究とそのアプリケーションプログラム開発」を設定し、2012 年度までの 6 年間、学生と共に実践的なプログラミングに取り組んできた。ここでプロジェクト演習を一区切りとするにあたり、これまでの成果を取り纏めて報告しておきたい。

２．FeliCa の現在

２．１　FeliCa に至る道

FeliCa は、「非接触型 IC タグ」の流れと「接触型 IC カード」の流れの合流により誕生したカードと言える。

まず、非接触型 IC タグは、「無線 IC タグ」あるいは「RFID(Redio Frequency IDentification）タグ」とも呼ばれ、個体の識別のために利用される。「タグ」という名が示すように、いわば「荷札」としての利用である。微小の IC チップとアンテナから構成され、外部からの電波を受けてチップの内容を読み書きし、さらに発信元に対して応答することができる。

古くは第二次世界大戦中にイギリス空軍が戦闘機の識別のために用いたトランスポンダにその原型を見ることができるが、産業的には 1980 年頃から家畜にチップを埋め込んで識別・追跡を可能にしたり、1990 年頃には自動車のキーにチップを埋め込んで盗難防止を図るイモビライザが普及するなど、その利用は次第に拡大していった。

さらに、2000 年代に入ると、狂牛病(BSE)対策など、食の安全のためのトレーサビリティを確保するために積極的に活用されるようになる。また、世界最大の小売業であるウォ

ルマートや米国防総省が RFID の貼付による納品を要請すると、一気に注目が高まった。

一方、IC カードは、人間が所有する磁気カードの発展形として 1970 年頃にドイツ、日本、フランスで、ほぼ同時期に発明されたものである。やはり薄いカードの中に IC チップを埋め込み、金属接点により、電気的に外部と情報交換をする仕組みである。

やがて 1980 年代に入り、産業界での活用の試行が始まるが、当時はすでに銀行業界、クレジットカード業界、流通業界などにおいて、安価な磁気ストライプカードと安価な磁気カードリーダが大量に普及していたため、磁気に比べると高価な IC カードと IC カードリーダがそれを置き換えることは難しく、ブームは沈静化する。

しかし、1990 年代に入り、銀行業界やクレジットカード業界における偽造カードの被害の拡大が問題化し、磁気カードよりもセキュリティの強い IC カードに再度、注目が集まった。その結果、1990 年代中頃からクレジットカードが、また 2000 年代に入ってキャッシュカードが、磁気カードから IC カードへと切り替わっていく。

以上の RFID と IC カードの二つの流れが 2000 年頃に融合して発展を始めたものが非接触型 IC カードと言える。

最も早くから取り組んでいたのが、オランダの NXP Semiconductors 社であり、同社の「MIFARE」は、現在、世界で最も普及している非接触型 IC カードである。日本においては、たばこ購入のための「taspo」に採用されている。

また、アメリカのモトローラ社も注力しており、セキュリティが高いという特徴が評価され、日本では「住民基本台帳カード」「IC 運転免許証」「IC パスポート」の他、銀行の「IC キャッシュカード」にも多く採用されている。

2001 年には、通信距離が 100mm 以下の近接型非接触 IC カード技術の国際標準として ISO/IEC 14443 が制定され、NXP Semiconductors 社の規格は ISO/IEC 14443 Type A、またモトローラの規格は ISO/IEC 14443 Type B として採用されている。

一方、SONY は 1988 年に JR 東日本との共同で非接触型 IC カードによる乗車券の開発に着手するが、91 年、JR 東日本はプリペイド式の磁気カード「イオカード」の導入を決定し、開発プロジェクトは失速しかける。しかし、偶然、翌 92 年に香港の公共交通機関の共通交通カード(オクトパスカード)の入札案件が起こり、95 年には MYFARE を押さえて受注に成功する。

当時の SONY の非接触型 IC カードは電池を内蔵しており、通信にはマイクロ波を使っていたが、香港案件の受注に向けた仕様変更により、今日の電池がなく、短波を使う「FeliCa」の開発に成功した。そして、その後、JR 東日本が磁気カードシステムの更改の時期を迎えるのを機に「Suica」としてようやく採用されたのである。

JR 東日本の「Suica」が 2001 年に稼働を開始すると、やはり FeliCa を利用して 2003 年には JR 西日本の「ICOCA」が、2006 年には JR 東海の「TOIKA」が、さらに 2007 年には首都圏の私鉄・地下鉄・バスで利用できる「PASMO」がスタートするなど、交通系において FeliCa の利用が急速に拡大し、事実上の標準となった。

さらに、SONY は FeliCa の電子マネーとしての活用に注力し、NTT ドコモなどと共同出資したビットワレット株式会社(現在は楽天 Edy 株式会社)を通じて電子マネー「Edy」を

2001 年にスタートさせ、2007 年にはセブン&アイ・ホールディングスの「nanaco」やイオンの「WAON」がスタートするなど、流通系の電子マネーとしても FeliCa のシェアは拡大している。

また、2004 年には FeliCa チップを内蔵した携帯電話「おサイフケータイ」が NTT ドコモから発売され、モバイルペイメントなどの道が大きく開けた。

国際標準という点では、2001 年には Type A と Type B に続く「Type C」として近接型非接触 IC カード標準に採用されることを目指していたが、これは実現しなかった。

しかしその後、NFC(Near Field Communication)という「近距離無線通信」の国際標準 ISO/IEC 18092 が制定されることになり、そこに FeliCa の規格が採用されている。ただ、そこには MYFARE の規格も共存しており、いわゆるエア・プロトコルとしての標準化であるという点が ISO/IEC14443 と異なる。

２．２　FeliCa カードの技術仕様と特徴

Felica カードの電波に関する主な仕様は、下表の通りである。先に述べたように、搬送波には 13.56MHz の短波が用いられている。通信速度は、理論的には 847kbps 以上が可能とされている。

表 1 Felica カードの通信に関する主な仕様

搬送波	13.56 MHz
変調方式	ASK 10%
符号化方式	Manchester
通信速度	212 kbps (Fc/64)
衝突検出/回避	タイムスロット

また、リーダ／ライタがカードを検出し、相互認証した後、データの読み書きが行われるが、それらに要する時間は、暗号処理を含めて約 0.1 秒以内で終了するという設計になっている。

次に FeliCa のファイルシステムは、「エリア」、「サービス」、「データブロック」という 3 階層で構成されている。エリアとはコンピュータで言うところのフォルダに相当するものであり、サービスは、データに対するアクセスの種類や権限を定義したものである。「データブロック」は、実際のデータが格納される領域であり、サービスを介してアクセスされる。エリアとサービスには「アクセスキー」が設定でき、セキュリティを確保している。

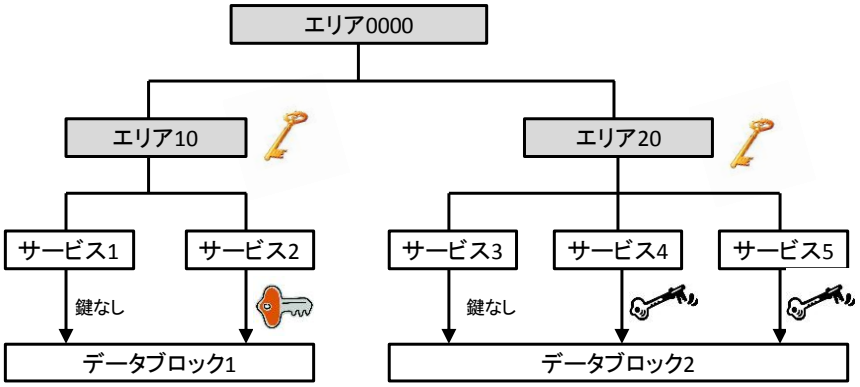

図１　FeliCa のファイルシステムの概念図(出典:http://www.sony.co.jp/Products/felica/)

３．プロジェクト演習における展開

３．１　経緯と準備

奈良産業大学情報学部では 2007 年度から「プロジェクト演習」を開始した。これは各授業で得た知識や技術を、プロジェクト演習の学習を通して、体系化・総合化させるとともに、主体的に学ぶ態度の育成を目指したものである。

そこでは、教員がすべて主導するのではなく、学生の自主性に基づく「サークル」的な運営とし、かつ「ゼミ」形式の学習形態をとる。また、共同作業を通して、コラボレーションの重要性、働くことの尊さや意義などを考えさせるとともに、職業観の基礎を養うことを目的としている。

そのテーマの一つとしてとりあげたのが「FeliCa の調査研究とそのアプリケーションプログラム開発」である。その理由は、先に見たように、2001 年に JR 東日本が FeliCa を使った「Suica」のサービスを開始したのを皮切りに、日本全国の交通機関が次々と FeliCa を採用し、2007 年当時では事実上の標準となっていたこと、そしてその流れは交通カードに留まることなく、電子マネーなどさまざまな分野へと広がりを見せつつあり、今後、社会の広範な領域で利用される基盤技術であると考えたからである。

まず FeliCa 関連のハード／ソフトとして下記を調達した。

①ICS-D004/20J SDK for FeliCa Lite Ver.2.0

SONY から提供されるアプリケーションを開発する上で必要となるライブラリ群

②RS-S320　USB リーダ／ライタ

Windows パソコンと USB 接続して、FeliCa カードを読み書きする周辺機器

③RC-S860　サンプルカード

テスト用の FeliCa サンプルカード

一方、パソコン側のシステム環境としては、下記のものを利用した。

①OS Windows Vista Ultimate SP2

②Visual Studio / Visual Basic2008（当初は 2005 を利用）

システムの全体構成としては下図のようになる。

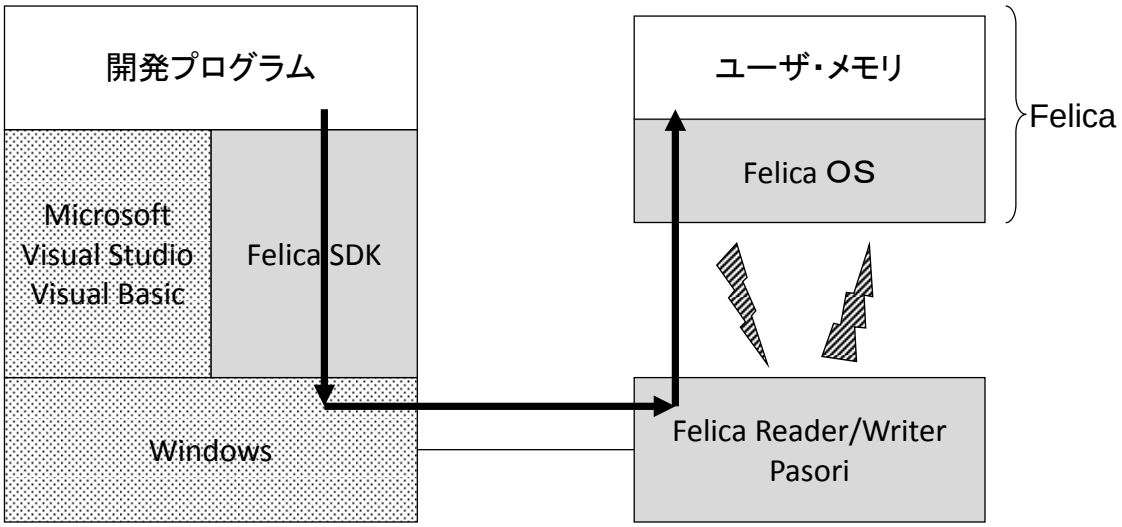

図 2　FeliCa 開発システムの全体構成

３．２　SDK for FeliCa Lite の機能とその特徴

開発用の SDK については、Lite と Standard の二種類が SONY から提供されていた。両者の機能の差を表 2 に示す。

表 2　SDK for FeliCa Lite と Standard の主な差異

分類	項目	FeliCa Lite	FeliCa Standard
ファイル構造	ファイルフォーマット	固定	自由に設定可
	エリア（階層構造）	なし	あり
	ユーザーブロック数	14 ブロック (224 バイト)	150 ブロック (2400 バイト)
セキュリティ	通信路の暗号化	なし	あり
	認証	片側認証	相互認証
	縮退鍵(アクセス鍵)	なし	あり
	アクセスタイプ	ランダムのみ	ランダム/サイクリック/パース
	認証コード生成機能(MAC)	あり	なし
	減算レジスタ	あり	なし
運用	システムコード	‘88b4h’固定	用途により異なる
	IDm 書き込みタイミング	チップ製造時(SONY)	カード製造時

主な違いを 3 点あげるならば、まず一点目として、Lite には、「エリア」が存在しない。社会で広く使用されている FeliCa では、メモリー内を複数のエリアに分け、それぞれに鍵を設定することにより、複数の事業者のアプリケーションを搭載することができるようになっているが、FeliCa Lite ではそれはサポートされない。

二点目として Lite では、通信路の暗号化やアクセス鍵などがサポートされず、セキュリティレベルが低い。そのためセキュリティ要件が厳しい社会アプリケーションには向かない。

三点目として、アクセスタイプがあげられる。FeliCa そのものには「ランダム」「サイクリック」「パース」の 3 種類が存在する。「ランダム」が通常のメモリーとしての読み書きのアクセスであるのに対して、「サイクリック」ではメモリーの一定サイズに対して FIFO の書き込みが可能であり、「パース」に対しては、「財布」という名前が示すとおり、減算、チャージなどの特別用途のアクセスが FeliCaOS を通して可能である。Lite では、このうち、最も一般的な「ランダム」のみが利用できる。

機能が制限された FeliCa Lite は Standard に比べて安価であることから、これらの制限を踏まえて、プロジェクト演習では FeliCa Lite を使用することとした。

次に SDK for FeliCa Lite とアプリケーションの関係は下図のようになる。

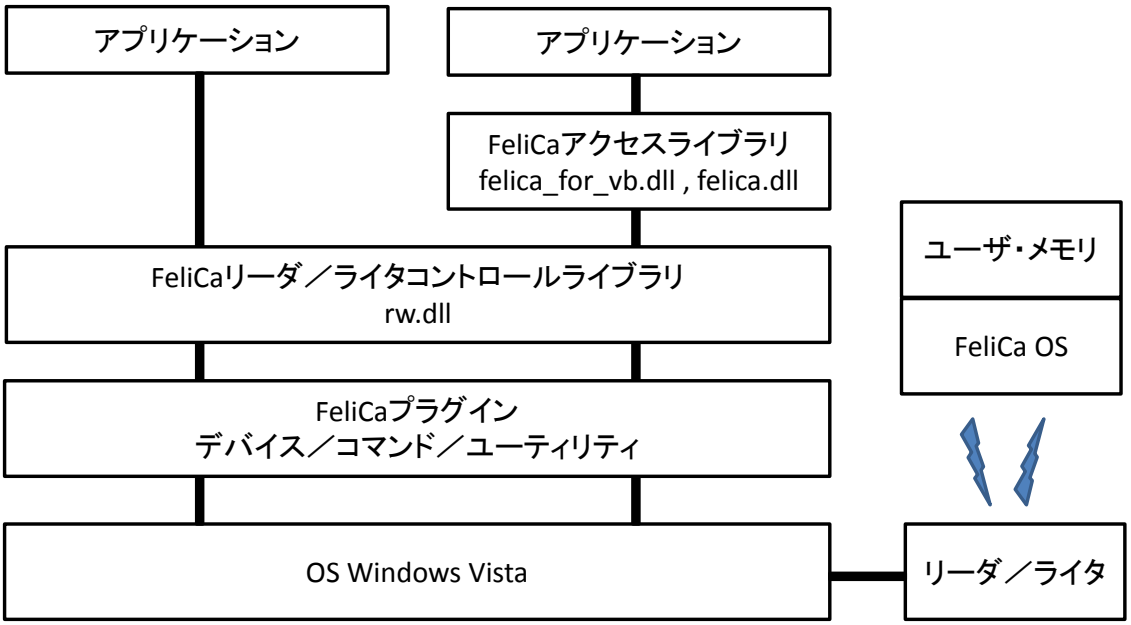

図 3　SDK for FeliCa Lite とアプリケーションの関係

SDK for FeliCa Lite は、①FeliCa アクセスライブラリ、②FeliCa リーダ／ライタコントロールライブラリ、③プラグインから構成されている。

①FeliCa アクセスライブラリ

Visual Basic 用の feliCa_for_vb.dll と C++および C#用の felica_dll の 2 種類が提供されている。リーダ／ライタのオープンやコントローラとの認証、FeliCa カードの認識や読み書きなど、リーダ／ライタや FeliCa カードへの基本的なアクセスに使用する上位のライブラリである。各種コマンドなどの API として提供されている。

これにより、下位ライブラリの FeliCa リーダ／ライタコントロールライブラリを意識することなく、ユーザはプログラミングが可能となる。

②FeliCa リーダ／ライタコントロールライブラリ

言語共通の rw.dll が提供されている。API は、リーダ／ライタのコマンドと 1 対 1 に対応している。

③プラグイン

よりハードウェアに近いレベルのデバイスマネージャーやコマンド、ユーティリティが提供されている。

３．３　FeliCa アクセスライブラリの利用と留意点

プロジェクト演習においては、FeliCa カードに対するデータの読み書きは、アプリケーションから FeliCa アクセスライブラリを呼び出す方式を採用した。FeliCa リーダ／ライタコントロールライブラリを使わなければ実現できないような要件はなかったからである。

FeliCa アクセスライブラリには豊富な関数が API として用意されているが、検討の結果、プロジェクト演習では、その中から、下記の 7 つの関数を利用することにした。

①initialize_library 関数

FeliCa アクセスライブラリの初期化を行うものである。

タイムアウト値やリトライカウント値など、ライブラリの内部で保有している設定情報はすべて初期化される。

②open_reader_writer_auto 関数

対象デバイスを走査してコミュニケーションポートを open し、さらにリーダ／ライタを自動認識して相互認証を行う。

③polling_and_get_card_information 関数

リーダ／ライタを経由して FeliCa カードを捕捉しにいき、成功したらカードの IDm と PMm を読み取ってきて FeliCa ライブラリ内部に記憶する。

カードの IDm は製造 ID、PMm は製造パラメータと呼ばれ、IC チップ製造時あるいはカード製造時に割り振られるユニークな 8 バイトの ID である。

IDm と PMm の構成を図 4 に示す。

アプリケーションとして重要なのは IDm であり、これにより製造者コード、カード識別番号を取得して利用することができる。

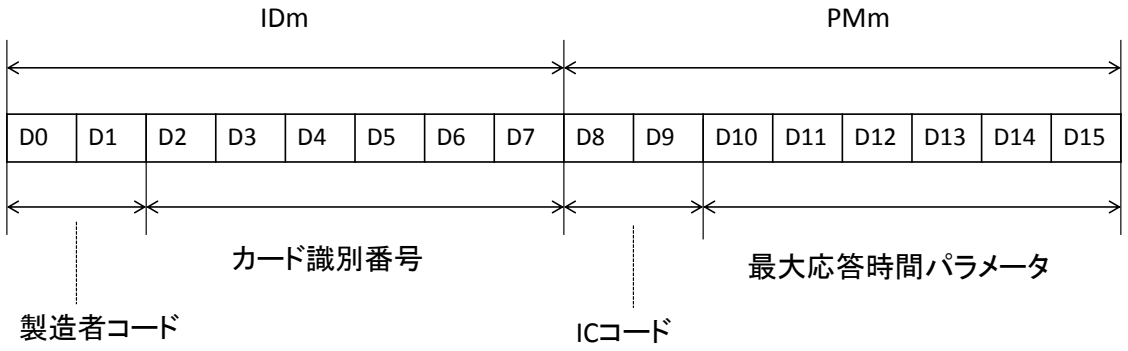

図 4　カードの IDm と PMm の構造(出典: FeliCa カード ユーザーズマニュアル)

以上の 3 つの関数の影響範囲を図 5 に示す。

これらの 3 つの関数は、必ずこの順に発行しなければならない。そしてこれらがすべて正常に完了した後に初めて、SDK for FeliCa Lite では、鍵のないサービスへのデータの読み書きが可能となる。

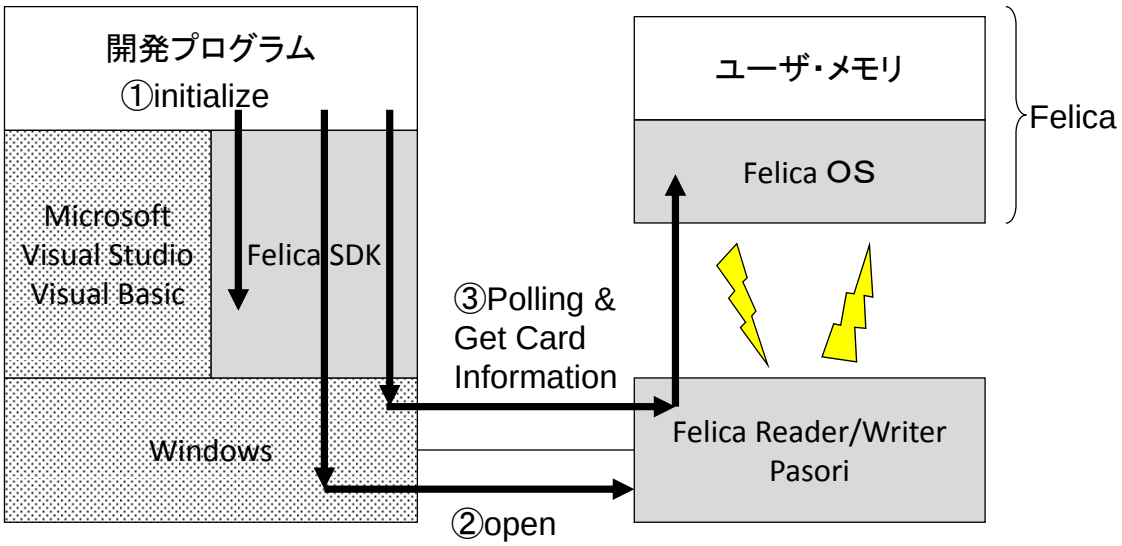

図 5　「initialize」、「open」、「polling and get card informaton」の影響範囲

④write_block_without_encryption 関数

鍵のないデータを FeliCa カードに書き込む。

FeliCa カードのメモリーアクセスは、16 バイトからなる「ブロック」単位となっているため、この関数もブロック単位のデータの書き込みとして使用する。

⑤read_block_without_encryption 関数

鍵のないデータを FeliCa カードから読み込む。

この関数もまた、ブロック単位のデータの読み込みとして使用する。

⑥close_reader_writer

リーダ／ライタをクローズする。終了処理に使用する。

⑦dispose_library

FeliCa ライブラリを解放する。同じく終了処理に使用する。

さて、これらの関数の戻り値はすべて bool 型であり、成功したら true、失敗したら false が返る。また、入力パラメータや出力結果の格納領域の確保はアプリケーション側の責任となっている。

例として、read_block_without_encryption 関数の API を下記に示す。

```
bool read_block_without_encryption (
    const input_structure_read_block_without_encryption input_read_block_without_encryption ,
    output_structure_read_block_without_encryption  output_read_block_without_encryption
)
```

表 3　read_block_without_encryption 関数のパラメータ

	パラメータ	内容
読み込みに必要なデータ input_read_block_without_encryption	card_idm	カードの IDm　8 バイト
	number_of_services	サービス数 n　1≦ n ≦ 16
	service_code_list	サービスコードリスト 2n バイトまたは 4n バイト
	number_of_blocks	ブロック数 m　1≦ m ≦ 255
	block_list	ブロックリスト（2m～3m バイト）
読み込み結果およびデータ output_read_block_without_encryption	status_flag_1	ステータスフラグ 1（1 バイト） 0x00:OK　その他:NG
	status_flag_2	ステータスフラグ 2（1 バイト） 0x00:OK　その他:NG
	result_number_of_blocks	ブロック数 m（1 バイト）
	block_data	ブロックデータの格納領域 （16m バイト）

４．サンプルアプリケーションの開発と評価

４．１　SDK のインストールと環境設定

SONY から提供されるライブラリをパソコン内のハードディスクにコピーした後、コントロールパネルから、FeliCa リーダ／ライタコントロールライブラリである rw.dll を環境変数として path に追加する。

例　Path C:¥FeliCa_Library¥FeliCa_Library¥bin

さらに、FeliCa プラグインのホームディレクトリを設定する。レジストリに登録する方

法もあるが、演習段階ということで、コントロールパネルからシステム環境変数 FELICA_PLUGINS として設定することとした。

例　FELICA_PLUGINS C:¥Felica_Library¥bin¥plugins

最後に、USB のカードリーダ/ライタのドライバをハードディスクにコピーした後、リーダ／ライタを接続し、ドライバをインストールした。

４．２　Visual Basic アプリケーションの開発環境の整備

Visual Basic のアプリケーションの構造としては、メイン・アプリケーション、API を呼ぶためのパラメータの構造体の定義、felica_vb.dll へのリンクの 3 つの部分に分けて構成した。

■メイン・アプリケーション

```
Imports System.Runtime.InteropServices
Public Class Form1
    グローバル変数の定義
    ・
    Private Sub initializeCmd_Click(ByVal ・・・
            ローカル変数の定義
            ・
            ・
            ’initialize_library 関数の呼び出し
            If (FeliCaDllWrapperBasic.InitializeLibrary() = 0) Then
                ErrorRoutine()
                Exit Sub
            End If
    End Sub

    Private Sub・・・
    ・
End Class
```

■API を呼ぶためのパラメータの構造体の定義

```
Imports System.Runtime.InteropServices
<StructLayout(LayoutKind.Sequential)> Public Structure StructureCardInformation
    Dim lngCardIdm As Integer
    Dim lngCardPmm As Integer
End Structure
・
```

■felica_vb.dll へのリンク

```
Imports System.Runtime.InteropServices
Public Class FeliCaDllWrapperBasic
    Public Declare Function InitializeLibrary Lib "felica_for_vb.dll"
        Alias "initialize_library" () As Byte
    Public Declare Function OpenReaderWriterAuto Lib "felica_for_vb.dll"
        Alias "open_reader_writer_auto" () As Byte
    Public Declare Function PollingAndGetCardInformation Lib "felica_for_vb.dll"
        Alias "polling_and_get_card_information" (ByRef ・・・) As Byte
    Public Declare Function WriteBlockWithoutEncryption Lib "felica_for_vb.dll"
        Alias "write_block_without_encryption" (ByRef・・・) As Byte
    Public Declare Function ReadBlockWithoutEncryption Lib "felica_for_vb.dll"
        Alias "read_block_without_encryption" (ByRef・・・) As Byte
    Public Declare Function CloseReaderWriter Lib "felica_for_vb.dll"
        Alias "close_reader_writer" () As Byte
    Public Declare Function DisposeLibrary Lib "felica_for_vb.dll"
        Alias "dispose_library" () As Byte
End Class
```

また、FeliCa カードの読み書きを行う「read_block_without_encryption」と「write_block_without_encryption」を実行するには、パラメータの設定など、事前と事後の作業が煩雑なため､1 ブロックの読み書きのためのサブルーチン ReadCard と WriteCard を作成し、学生の演習においてはそれぞれを下記のように呼び出してプログラムしてもらうことにした。

```
Dim ReadData(15) As Byte              ’ 読み込むデータを格納する領域
Dim NumberOfBlocks(0) As Byte         ’ API のためのパラメータ領域
ReadCard(ReadData, NumberOfBlocks)    ’ 読み込みのためのサブ ReadCard を呼ぶ

Dim WriteData(15) As Byte             ’ 書き込むデータを格納する領域
Dim NumberOfBlocks(0) As Byte         ’ API のためのパラメータ領域
・・・書き込みたいデータを WriteData(15)に格納する・・・
WriteCard(WriteData, NumberOfBlocks)  ’ 書き込みのためのサブ WriteCard を呼ぶ
```

このサブルーチンにより、プログラミング経験が少ない学生でも、比較的容易に FeliCa とのデータの読み書きができるようになった。

４．３　開発したアプリケーション

例として、「キャンパスカードシステム」と「交通カードプログラム」を紹介する。

４．３．１　キャンパスカードシステム

FeliCa をキャンパスカードとして使用するアプリケーションである。

[initialize open ボタン]をクリックすると、ライブラリが初期化され、リーダ／ライタが open されて使用可能となる。

[カード待ち受け開始ボタン]をクリックすると、カードを捕捉するための電波がリーダ／ライタから飛び始める。この処理は、カードがその時に存在しないとエラーで返ってくるので、Visual Basic のタイマーコントロールにより、一定間隔でリトライを続ける。

やがてカードがリーダ／ライタの上に来ると、②の待ち受け処理が正常終了し、カードの IDm と PMm が返ってくる。ここで妥当な IDm と PMm かをチェックすることもできる。

その後はアプリケーションの処理となる。

利用できるサービスとしては、次の 3 種類とした。

・学生証

　カードには学籍番号だけ保持し、氏名や写真は PC 側にもっている。

・バス回数券

　入力した金額に応じたバス回数券がチャージされ、乗車ごとに回数が減る

・電子マネー

　入力した金額に応じてチャージと減算が行われる。SDK の Standard を利用すればパースアクセスで処理可能であるが、今回はアプリケーションで処理した。

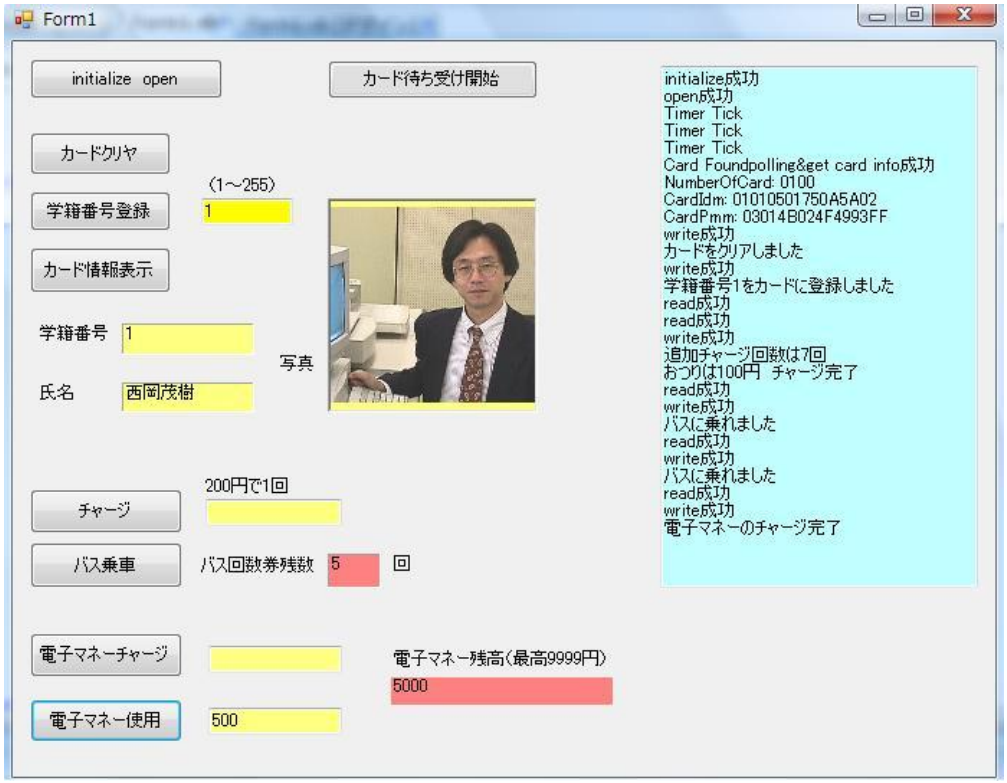

図 6　キャンパスカードシステム

４．３．２　交通機関の改札システム

FeliCa を交通カードとして使用するアプリケーションであり、乗車駅プログラムと降車駅プログラムの１セットで構成される。

[乗車駅改札スタートボタン]および[降車駅改札スタートボタン]をクリックすると、ライブラリが初期化、リーダ／ライタの open に続き、polling によりカードを捕捉するための電波がリーダ／ライタから飛び始める。タイマーコントロールによりカードが来るのを待ち続け、カードを捕捉したら乗車駅プログラムではカードに乗車駅コードを書き込み、降車駅プログラムでは降車駅コードを書き込んだ後、運賃を計算し、カードの残高から引き落とす。

運賃計算のためのデータはコンピュータ側に保持している。

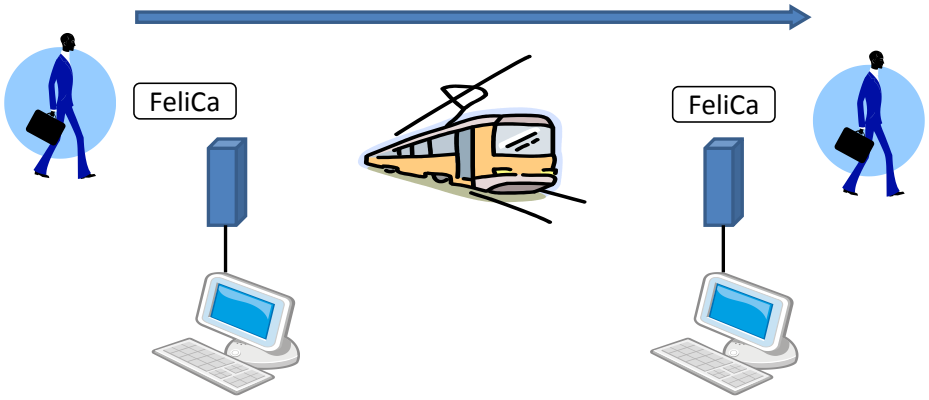

図７　システムの全体構成

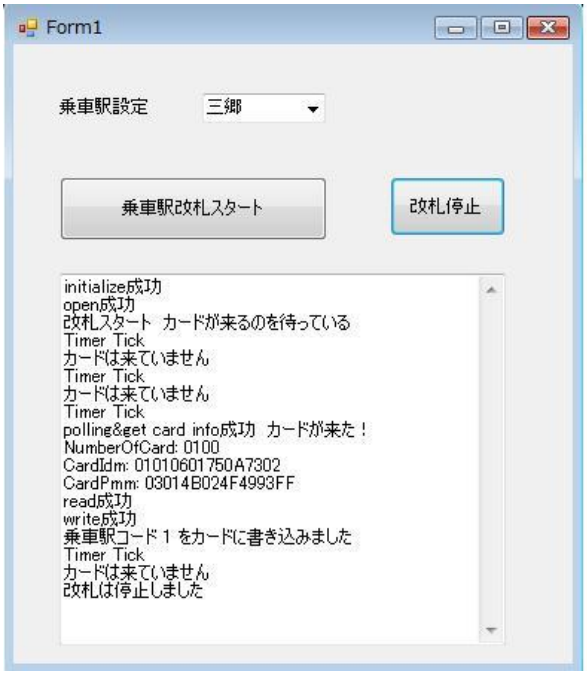

図８　乗車駅プログラム

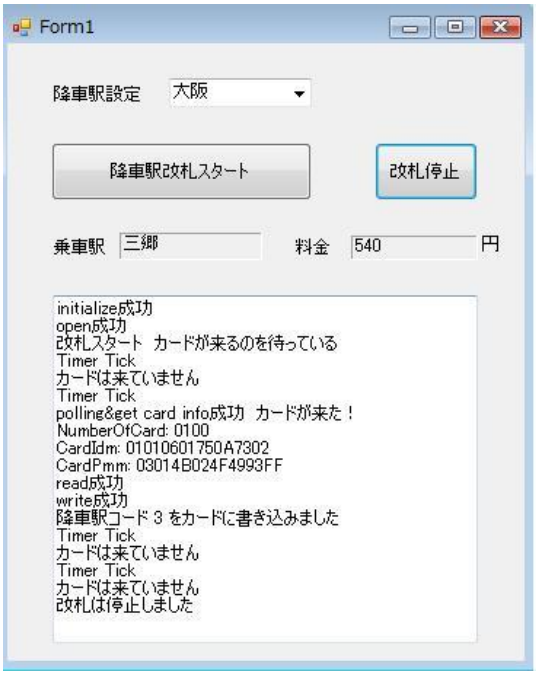

図９　降車駅プログラム

５.おわりに

非接触 IC カード FeliCa について、関連する技術開発の歴史を踏まえて、その現状を明らかにし、さらに SDK を利用して実際に開発したシステムの成果の一部を発表した。

無線を使ったアプリケーションということで、さまざまな試行錯誤があったが、一応、SDK for FeliCa Lite を使用したシステム開発の方法論と枠組みは確立できたと考えている。

また、今回は予算の都合から SDK for FeliCa Lite を使用したため、本来、FeliCa が持つ 3 つのアクセスタイプ「ランダム」、「サイクリック」、「パース」のうち、最も一般的な「ランダム」を使ったシステム開発しかできなかった。今後は、環境を整えた上で「サイクリック」「パース」についても開発・実験を実施していきたいと考える。

FeliCa への取り組みを開始してから 6 年が経過したが、その間、FeliCa 技術も進化し、その応用領域も拡大し続けている。また、近年では ISO/IEC 18092 として制定された NFC としての活用も注目されており、今後は、さらに広がりを見せると考えられる。引き続き、NFC Forum の動きなどを注視しながら、非接触 IC カード、RFID について調査・研究を進めていきたい。

参考文献

1) 立石泰則,「フェリカの真実」, 草思社、2010 年 11 月

2) RFID テクノロジ編集室,「無線 IC タグのすべて」,日経 BP,2004 年 4 月

3) RFID テクノロジ編集室,「無線 IC タグ導入ガイド」,日経 BP,2004 年 11 月

4)「情報化白書 2012 年」

5)「流通とシステム」,No.149/2012,財団法人流通システム開発センター

6) FeliCa カード ユーザーズマニュアル

7) FeliCa カーに関するソフトウェア開発テクニカルノートド

8) FeliCa Lite ユーザーズマニュアル

9) FeliCa Lite に関するソフトウェア開発テクニカルノート

10) SDK for FeliCa ユーザーズマニュアル機能概要編

11) SDK for FeliCa ユーザーズマニュアルアクセスライブラリ編

12) http://www.sony.co.jp/Products/felica/　FeliCa ホームページ

13) http://www.nfc-forum.org/home/　NFC Forum