学習装置をイントラネットで共有する
プログラミング学習システムの開発

伊　藤　　敏　　井　上　祥　史

概　要

本研究では、先に開発をしたプログラミング入門学習者に学習意欲を維持しながら、本質的な点に集中できる学習装置をイントラネットで共有するシステムを開発した。学習装置のネットワーク共有にはサーバクライアント方式をとり、クライアントからプログラムを送付してサーバ側で実行をする方式とした。学習者が利用するプログラミング言語およびシステム構築に用いる言語として Excel 内の Visual Basic for Applications を採用した。

1．はじめに

高度情報化社会の登場により、社会活動に情報システムが深くかかわるようになってきている。そして広辞苑によれば、情報化社会とは「情報が物質やエネルギーと同等以上の資源とみなされ、その価値を中心にして機能•発達する社会」とされるように、情報の果たす役割が増大している。また情報技術の進歩は速い。新しい技術の導入によって、情報システムは大きく変化をする。すなわち、情報化社会は常に変化することを属性として持つ。

「情報教育＝コンピュータ操作教育」の時代は終わり、情報化社会の流動性に対応した情報化社会を生き抜くための、独創性・創造性を育成する教育観へ移行している［1］、［2］。すなわち、「問題を発見し、解決をする」 という能力の育成が要求されている。このような能力の育成に、プログラムを作成する教育は有効な方法の一つである［3］、［4］。また、情報を活かすためにはプログラムが必須であり、広く教育される必要性がある。

プログラムを作成する教育では課題に対して、問題点を発見し、解決の方法を探して、それに基づき変数とアルゴリズムを設計し、それらをプログラミング言語へコーディングする必要がある。一般にプログラム言語の習得には“Hello World”を表示するプログラム作りから始まり、そこから先は抽象的な「難しい」作業が続くことが多い。

筆者らはビット情報の入出力により、床を自由に移動可能な学習装置の開発を行った［5］。さらに、それを用いて学習装置としてロボット車両を動かすことから始める教材システムの提案を行った［5］、［6］。そこではロボット車両はコンピュータから 4 ビットの指令と時間管理指令を受けて任意の位置へ移動する。また、ロボット車両はコンピュータへ 2 ビットの床面明暗情報を送る。学習者はこれらの情報を基にビット制御のプログラ

ムによってロボット車両の動きを決定することができる［5］、［6］。さらに、筆者らはこの学習装置を動かすプログラム作成環境としてExcelとVisual Basic for Applicationsを用いたシステムの提案を行い、教育実践の結果、学習者のプログラム作成に対する自信や好悪感が好転する結果を得た［6］、［7］。これらは、学習者はロボット車両をどのように動作させるかイメージし、それをプログラムする。その結果がすぐにロボット車両の動きとして反映する。学習者は五感を使って自分が作成したプログラムを検証でき、意欲を維持しつつ学習を進めることが可能なためである。

本稿の目的は、先に設計・開発した学習装置であるロボット車両をイントラネットで共有する学習システムの設計・開発を行うことにある。そしてe-Learningでの利用を想定して、インターネット対応のセキュリティを実装することで、遠隔地からでも学習装置を利用したプログラミング学習が可能になることを意識している。

以下に2でシステムの設計の方針を、3でシステムの開発を、4でまとめを述べる。

2．ネットワーク対応学習システムの設計方針

前報で開発した学習装置の基本を継承する。前報［5］で開発した学習装置の基本は4ビットの指令と時間関数により平面状を自在に移動可能であり、学習装置前部のセンサにより床の明暗情報を2ビットでコンピュータへ伝える機能を持つ。概略を図1に再録する。

車輪制御に用いる4ビットのうち、上位2ビットが左車輪の制御、下位2ビットが右車輪の制御に当てる。センサ情報の2ビットは上位1ビットで左の明暗を下位1ビットで右の明暗情報を与える。ビット情報による車両の動きおよび床情報を表1に示す。

学習装置をネットワークで共有するには、ネットワーク越しに学習者と学習装置がビット情報のやり取りをできる必要がある。そして、スタンドアロンで実施した［7］場合のように学習者1名（または1グループ）に対し学習装置1台と言う関係ではなく、多数の学習者に対し、学習装置は1台と言う関係が想定される。その場合、学習装置はサーバに接続されることが必須となる。その際、二つの方式が考えられる。一つは、サーバはクライアントと学習装置のビット情報のやり取りのみをサポートする［8］。この場合は、ビット情報は一回一回サーバを経由してクライアントと学習装置の間を行きする。もう一つは、クライアントでビット情報の操作を含むプログラムを作成し、サーバへプログラム実行依頼を行い、プログラムの実行はサーバ側で行う方式である。本論文では後者の方式を採用する。

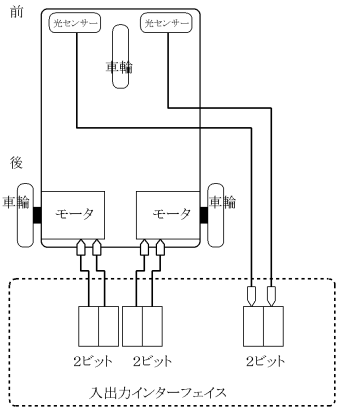

図１．　ビット操作で動くライントレーサ

表１．ロボット車両への４ビット指令と２ビットによる床情報

４ビット指令

00 00	停止	01 01	前進	10 10	後進
00 01	前進左折	01 00	前進右折	00 10	後進左折
10 00	後進右折	10 01	左旋回	01 10	右旋回

２ビットの床情報

00	暗 暗	01	暗 明	10	明 暗	11	明 明

この学習装置を用いたネットワーク対応のシステム開発方針をつぎの３点にまとめる。

１．サーバ・クライアント型とする

ネットワークからの利用は図２に示すサーバ・クライアント型とする。学習装置はサーバへ接続する。サーバはクライアントから送付されたソースプログラムを解析して、その要求に基づき学習装置をビット情報で操作する。学習者はクライアント上で学習装置を操作するプログラムを作成し、サーバへソースプログラムを送付することで操作依頼を出し、学習装置を遠隔操作する。

２．学習装置

先に開発した学習装置の機能を継承する。サーバであるコンピュータとUSB経由で直接接続し、４ビットの指令と指令持続時間により、任意の位置へ移動が可能であり、車両前方の左右に床の明暗情報である２ビット情報をコンピュータへ提供する。なお、学習装置はサーバにのみ接続される。

３．開発ソフトウェア

サーバとクライアントに用いるソフトウェアはExcelとVisual Basic for Applications（以下VBAと略す）を用いる。このソフトウェア選択はスタンドアロン環境での教

育実践実績［7］を継承することと、実際の教育現場で新しい言語環境を構築する抵抗感を減らす意味を持つ。

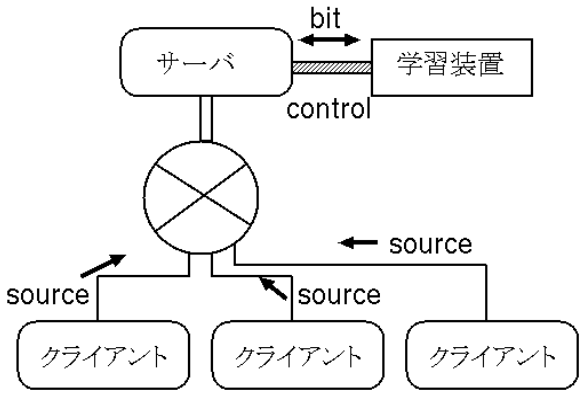

図2．サーバ－クライアントモデルの学習システム

3．システム開発

システムは学習装置、サーバ、クライアントの三つの部分から構成される。学習装置はUSB経由でサーバと接続される。サーバとクライアントはイントラネットを通じてつながる。一つのクライアントからシステムを利用する形態を図3に示す。学習者がクライアント経由でサーバの学習装置を操作し、結果が学習装置の動きとして学習者へ返される。ただし、ネットワーク経由であるがゆえに、クライアントへ提供する学習装置の動きは、別途Webカメラなどを用意する必要がある。

学習装置はビット情報により動作を決める［5］、［6］、［7］。サーバ、クライアントを構築するソフトウェアはWindows上のExcel+VBAを用いる。ここでは学習装置の駆動にUSB-IOを動かすライブラリ［6］、［7］を、TCP/IPを使った通信にはMicrosoft Winsock Controlを用いる［9］。サーバはクライアントからのソースコード受け入れと、そのソースコードを基に学習装置を駆動する機能が必要である。ソースコード受け入れはTCP/IPの通信を利用し、学習装置の駆動はExcel+VBAで作成されたライブラリを利用する［6］、［7］。クライアントは学習者が学習装置を動かすプログラムを作成・編集する機能と、そのソースコードをサーバへ送付する機能が必要である。3.1でサーバについて、3.2でクライアントについて述べる。なお、本論文ではサーバとクライアントのTCP/IPプログラムをExcel+VBAで作成した。教材としての利用を意識してソースプログラムを巻末に提示する。

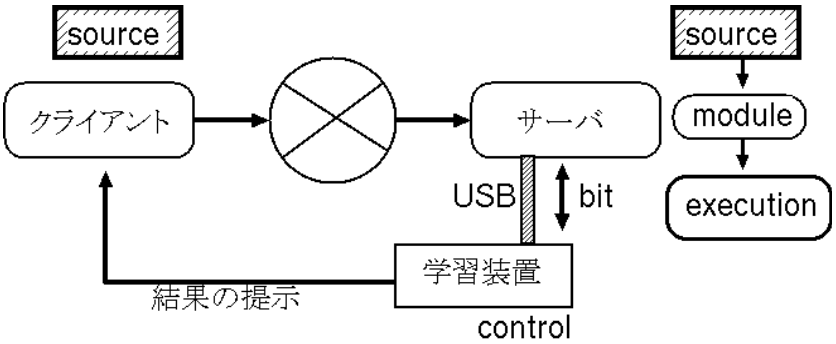

図３．　ビット操作で動くライントレーサ

３．１サーバの設計・開発

サーバはクライアントからの接続を待ち受けてソースコードを受け取る機能と学習装置を駆動する機能が必要である。ソースコード受け入れ機能と学習装置駆動機能を分けて述べる。

・待ち受けとソースコード受け取り

ある特定のポートでクライアントからの接続要求を待つのに Winsock を用いる［９］。接続要求があった場合、クライアントへ受け入れ可能のサインを送る。クライアントから送付されるデータを受信する。受信データを解析して、先頭にソースコード送付のサインである“Send”がある場合、それ以降をソースコードとして展開し、Excel+VBA のモジュールとして別のファイルへ保存をする。別ファイルへ保存する意味は、Excel+VBA では動的に生成されたプログラムを実行する機能を持たないため、一度主メモリから開放してハードディスクへ格納し、再度、主メモリへ呼び出す手順が必要となるためである。

“Send”以外の場合はクライアントへ処理ができない旨を通知して、引き続き接続要求を待つ。なお、今回は排他処理機能を実装しなかった。

・学習装置駆動機能

学習装置の駆動は、クライアントから送付されたソースコードを基にサーバで生成され保存された別ファイルを呼び出して実行することで行う。別ファイルには擬似的に階層化されたライブラリ［６］、［７］を含む。このライブラリはビット情報の入出力命令として、学習装置のポート０への出力 outP０，ポート１の情報入力 inP１を定義している。さらに、学習装置の９つの基本動作および時間管理関数をつぎの命令で定義している。前進：forward、後進：back、停止：stp、前進左折：rightplus、前進右折：leftplus、後進左折：rightminus、後進右折：leftminus、右旋回：twistright、左旋回：twistleft、時間関数：wait（数字）。

図４に実装されたサーバの画面を示す。中央部にクライアントから送られてきたデータを表示する機能を持つ。クライアントからのデータの受け取りから、ファイルの生成、学

習装置の実行まですべて自動で行う。そのため、サーバに必要なユーザインターフェイスはクライアントからのデータ受け取り状況を確認するデータ表示窓と終了操作ボタンのみである。実装したサーバのソースコードをリスト１に示す。

サーバには USB を経由して学習装置が取り付けられ、生成されたファイルからの指令で学習装置を動かす。サーバとクライアントが同じ部屋にあるイントラネット環境の場合、その様子を、学習者は観察することが可能である。

なお、クライアントがサーバへ接続するために必要な IP アドレスやドメイン名は教授者が学習者へ伝える。

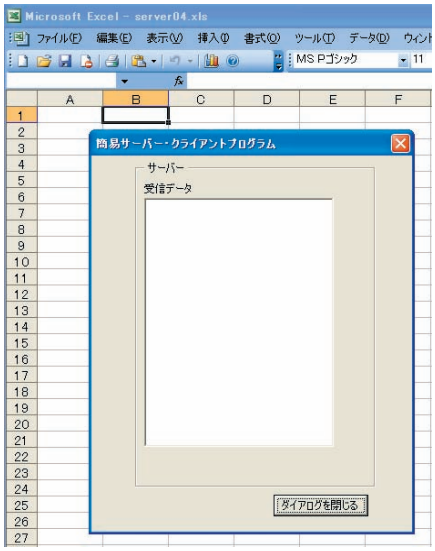

図４．サーバの待ち受け画面

３．２クライアントの設計・開発

クライアントが必要な機能は、学習者がプログラムを作成・編集する機能、サーバへ作成したプログラムを送付し、実行依頼する機能である。

本システムを用いた学習者によるプログラミングの流れをつぎに示す。

１．クライアントでプログラム作成・編集
２．サーバ名または IP アドレスを入力し接続
３．サーバからのメッセージ確認（接続確認）
４．ソースコードの送付と実行依頼
５．実行結果を確認　サーバが同一の部屋にある場合、その場で確認可能
　　別室などの場合は学習装置を実時間で撮影・放映する Web カメラが必要
６．必要に応じて１へ戻る

ここで、教授者は学習者へサーバへの接続情報（サーバ名または IP アドレス）を伝え、

学習装置の動きを確認する。

上記クライアント機能を満たすユーザインターフェイスを図5に示す。フォーム画面の上から順にプログラム入力・編集画面、サーバへの接続情報入力と接続ボタン、サーバからの応答情報表示画面、サーバへソースコード転送と実行画面依頼ボタン、終了ボタンからなる。編集画面でプログラムを作成する。Visual Basic Editor が持つような入力支援機能は実装しなかった。学習者はプログラム入力・編集画面より学習装置の9つの基本動作である前進、後進、停止、前進左折、前進右折、後進左折、後進右折、左旋回、右旋回、および時間管理関数の知識と順次処理、繰り返し処理、条件分岐を使ってプログラムを作成する。

図5.　クライアントの画面

3.3 試験運用

作成したサーバとクライアントを用いて試験運用を行った。その際用いた基本オペレーティングシステムは Windows XP Professional である。サーバ及びクライアントともに Excel+VBA で作成をし、Windowsx XP のサーバ機能は利用していない。サーバの待ち受けポートとして1000番を利用した。Windows XP でポート1000番を待ち受け可能なように設定する。

イントラネット環境で、サーバはクライアントからの接続要求に応答し、クライアントはサーバからの接続許可後、ソースコードを送付し実行依頼をする。サーバはソースコードを解析して、標準モジュールの一つとして別名で保存を行い、直ちにその別名ファイルを呼び出して、実行する。その結果、サーバに接続された学習装置がソースコードに基づ

く動きをする。サーバからクライアントへは学習装置の動き以外の応答はしない。

試験運用の結果、サーバの動きは若干時間の遅れはあるものの、クライアントからの要求どおりの動作を行った。

4．まとめ

イントラネット環境でサーバに学習装置を取り付けて、クライアントから学習装置を動かすプログラムを送付することで、サーバ側で学習装置を駆動することができた。学習装置の製作には多くの人手がかかる。イントラネット利用により、１台の学習装置でも複数の学習者が学習装置を利用することが可能になった。

なお、学習装置の駆動にはビット情報を送付するだけで可能である。したがって駆動するためのソースコードを送付する必要はない。しかし本研究ではソースコード送付方式を選択した。その理由は、学習装置を駆動することが目的ではなく駆動する一連の流れを記述するプログラム作成の学習が目的であるためである。

現状ではクライアントから送付したプログラムでエラーが発生した場合の処理は実装をしていない。サーバで実行中に発生したエラー情報をクライアント側へ転送するなどの機能、およびエラー処理やエラーからの回復などの機能を実装する必要がある。

謝　辞

本研究は、文部科学省科学研究費、基盤研究Ｃ研究代表者　伊藤敏　課題番号17500657、および基盤研究Ｃ研究代表者　井上祥史　課題番号18500696より補助を受けた。

参　考　文　献

［１］教育システム情報学会情報教育特別委員会、「情報教育推進のための提言書」、(2003)

［２］大学における一般情報処理教育の在り方に関する調査研究委員会、「大学における一般情報処理教育の在り方に関する調査研究（文部科学省委託調査研究）平成13年度報告書」、(2002)

［３］原田悦子、「文科系大学・学部における情報教育―その目的と問題―」、情報処理、第41巻、No. ３、 pp227-233（2000）

［４］家本修、「プログラミング教育」、教育工学事典、 pp459（2000）

［５］伊藤敏、井上祥史、「体験型プログラミング学習装置の設計・開発」、 Review of

Economics and Information Studies、Vol. 8 No. 1・2 pp159-170、(2007)
［6］伊藤敏，井上祥史、『体験型プログラミング』、現代図書、2008年、pp 1 -141
［7］伊藤敏、井上祥史、「見て、聞いて、触って操作する教材の開発」、教育システム情報学会論文誌、Vol. 25、No.1、pp. 75-80、(2008)
［8］伊藤敏、「電磁リレーの基礎と応用　ネットワークから制御するリレー」、新電気、vol. 62, pp101-106 (2008)
［9］志村康人、『VBA による TCP/IP プログラミング入門』、CQ 出版、2005年

リスト1. サーバのソースコード

```
Private Sub CommandButton1_Click ()
    'サーバー
    '----「ダイアログを閉じる」ボタン
    Winsock2.Close  'サーバーを閉じる
        End
    End Sub
Private Sub Frame1_Click ()
End Sub
Private Sub TextBox1_Change ()
End Sub

Private Sub UserForm_Initialize ()
    'サーバー
    '---- 初期化処理
    'Winsock2 サーバー
    Winsock2.Close
    'サーバーの初期化処理
    Winsock2.LocalPort = 1000
    '待ち受けのポート番号
    Winsock2.Listen
   End Sub

Private Sub Winsock2_ConnectionRequest
(ByVal requestID As Long)
    'サーバー
    '--- クライアントからの接続要求受け付け処理
    TextBox1.Text = TextBox1.Text &"接続
されました"& vbCrLf '自分のテキストボック
スへ表示
    If Winsock2.State<>sckClosed Then
        Winsock2.Close
    End If
    Winsock2.Accept requestID
    Winsock2.SendData "接続されました"&
vbCrLf 'クライアントへメッセージ送付
End Sub

Private Sub Winsock2_DataArrival (ByVal
bytesTotal As Long)
    'サーバー
    '--- クライアントからのデータ受信処理
    Dim com (2) As String 'コマンド文字列
    Dim work_data_name () As String '
作業用文字列
    Dim pro_name As String ' Procedure
name
    Winsock2.GetData buff, vbString
    TextBox1.Text = TextBox1.Text & buff
 & vbCrLf
    TextBox1.SetFocus
    'コマンドを取り出す
    pic_len = Len (buff)
    com (1) = Mid$ (buff, 1, 4)
    com (2) = Mid$ (buff, 7, pic_len - 6)
    '## ソースコードのプロシージャ名取得
    work_data_name=Split(com(2),vbCrLf)
    pro_name=Replace(work_data_name(0),
"sub","")
    pro_name=Replace (pro_name, "", "")
    pro_name=Replace (pro_name, " () ", "")
    pro_name=Replace (pro_name, " (", "")
    pro_name=Replace (pro_name, ") ", "")

    myBUFF = com (2)  ' source text
```

```
'################################
'  now constract  2006.05.02
    'コマンドを処理する
    Select Case com（1）
        Case "Send"          ' Client から Se
nd メッセージを受け取ったら実行
' st-trace-tmp.xls ファイルへ動的にプロシー
ジャを追加

' treatbuff は Module で定義
        Call treatbuff
        Winsock2.SendData "Surely received
 and Make file " & pro_name & vbCrLf '
Client へ message
        ActiveWorkbook.RefreshAll
        Reset
        Sleep 1000 '動作チェックのため1秒待つ
        On Error Resume Next
        Winsock2.SendData "Now running!" '
 Client へ message
  '## 実行する Excel ファイルとプロシージャ
名の指定
  '##
      Application.Run ThisWorkbook.Path &
 "¥st-trace-tmp.xls!" & pro_name
      Reset
          Case Else
              Winsock2.SendData "処理できな
いコマンドです " & com（1） & vbCrLf
    End Select
    Exit Sub
'################################
    'エラー処理
dir_err:
    Select Case err
        Case 53
            Winsock2.SendData "エラー " &
 err & "ファイルが見つからない" & vbCrLf
        Case 76
            Winsock2.SendData "エラー " &
 err & "パスが見つからない" & vbCrLf
        Case Else
            Winsock2.SendData "エラー " &
 err & vbCrLf
    End Select
    On Error GoTo 0
End Sub
Public myBUFF As String
'#################
'##    動的に Module と Procedure を追加する
Public Sub treatbuff（）
Dim myStr As String
Dim myWorkbook As Workbook
myStr = myBUFF 'com（2） souce code ;
ネット経由で取得したソースコード
'## 以下の Filename は加工するファイル指
定のパスとファイル名は環境に応じて変更して
ください
Workbooks.Open
Filename:="D:¥home¥usbio¥st-trace-tmp.xls
" 'ThisWorkbook.Path & "¥st-trace-tmp.xls"
Set myWorkbook = ActiveWorkbook ' This
Workbook
'##  上記 Filenmae で指定したファイルへ動
的に Module を追加し，プロシージャを加える
With
myWorkbook.VBProject.VBComponents.Add
（vbext_ct_StdModule）
.CodeModule.AddFromString myStr
End With
ActiveWorkbook.Save  '保存
ActiveWorkbook.Close '終了
Set myWorkbook = Nothing 'オブジェクトの開放
'With
ThisWorkbook.Application.VBE.ActiveVBPro
ject.VBComponents.Add(ComponentType:=v
bext_ct_StdModule)
' .CodeModule.AddFromString myStr
' .Name = myModule_name
'End With
End Sub
```

リスト2．クライアント

```
Private Sub CommandButton2_Click（）
    'クライアント
    '---- 「ソースの送信・実行」ボタン
```

```
    'TextBox1.Text = "send" & vbCrLf &
TextBox1.Text

   Winsock1.SendData "Send" & vbCrLf &
TextBox1.Text
  "MsgBox "send" & vbCrLf & TextBox1.Text
   TextBox3.Text = "" 'サーバからの応答画
面を消す

End Sub

Private Sub TextBox1_Change ()

End Sub

Private Sub UserForm_Initialize ()
    'クライアント共通
    '---- 初期化処理
    'Winsock1 クライアント
    Winsock1.Close
        'クライアントの初期化処理
    Winsock1.RemotePort = 1000

    'TextBox4.Text = "localhost" 'サーバー名
    TextBox2.Text = "172.20.1.11" 'サーバー
IP アドレス

End Sub

Private Sub CommandButton1_Click ()
    'クライアント
    '----「サーバーに接続」ボタン

    Winsock1.RemoteHost = TextBox2.Text
    Winsock1.Connect     'サーバーへの接続


'     Call UserForm_Initialize
End Sub

Private Sub CommandButton4_Click ()
    'クライアント共通
    '----「ダイアログを閉じる」ボタン
    Winsock1.Close  'クライアントを閉じる

    End
End Sub

Private Sub TextBox2_Change ()
 ' サーバのアドレス
End Sub

Private Sub TextBox3_Change ()
  'サーバから送られてきた応答の表示
End Sub

Private Sub Winsock1_DataArrival(ByVal
bytesTotal As Long)

    'クライアント
    '--- サーバーからのデータ受信処理

    Winsock1.GetData buff, vbString

    TextBox3.Text = TextBox3.Text & buff
 & vbCrLf

    TextBox3.SetFocus

End Sub

Private Sub Winsock1_Error (ByVal Number
As Integer, Description As String, ByVal
Scode As Long, ByVal Source As String,
ByVal HelpFile As String, ByVal HelpContext
As Long, CancelDisplay As Boolean)

End Sub
```