論　説

モデル進化型ビジネスゲームの構想

白　井　宏　明

1．ビジネスシミュレーションとビジネスゲーム

ビジネスシミュレーションは，企業が行う様々なビジネスの構造をモデル化し，それをもとにシミュレートすることで，問題の解決に役立つ解を求めるための手法である．

このようなシミュレーションは，モンテカルロシミュレーションやシステムダイナミックス等のコンピュータシミュレーションを用いることで可能となる．しかし企業経営の問題を対象とする場合，人間の意思決定ルールが明示化されないとプログラムを作成することができない．そのような場合に，プログラムにできない意思決定部分を人間に入力させることで，シミュレーションの実行を可能とするのが，ゲーミングシミュレーションである．ビジネスには競争相手がいるので，人間のプレーヤが参加するゲーミングシミュレーションが有効であり，これはビジネスゲームと呼ばれる．（白井 2010）

このビジネスゲームをビジネスモデルの研究に利用することが考えられる．人間が参加するゲーミング手法を用いて，複数のプレーヤが一定のルールのもとで敵対，競争，協調しながら課題を追求する仮想空間を提供することが求められる．失敗が許される環境の中で仮想の企業経営を行い，複数の人間による意思決定を繰り返すプロセスを通して未来世界での体験を積んでいき，これをもとに現実世界での合意を形成し新しいビジネスモデルを検証するという手法が期待される．（白井 2013）

ビジネスゲームを開発できる仕組みを実現するため，ビジネスゲームの開発と運用を支援するプラットフォームとして，筆者らはYBG（Yokohama Business Game）を構築した（白井 2005）．このシステムはもともとは筑波大学の社会人大学院（筆者も在籍）でプロトタイプが開発され，その後，横浜国立大学で改良を続けている．YBGの最大の特長は，大学教員自身がビジネスゲームを開発できるように，日本語が使える専用の簡易言語を実装したことである．このシステムは現在，全国120大学以上に提供されている．

2．言語的定性的ビジネスゲーム

これまで開発されてきたビジネスゲームは，数値分析を通じた合理的意思決定のスキル獲得を指向し，プレイヤーに戦略的意思決定よりは，オペレーショナルなレベルにおける意思決定

を求めるものが多かった．これを「定量的ビジネスゲーム」と呼ぶことにする．これに対して田名部らは，経営戦略策定やプロジェクトマネジメントなどの，将来に対する不確実性が高い活動において，関係者間での理解と解決策の共有や戦略策定における合意形成をもたらす方法として「言語的定性的ビジネスゲーム」を提案し，その実現例としてCIO育成ゲームを実装し，実験を通じてその有用性を評価した（田名部他 2014）．CIO育成ゲームは，情報システムの導入に対して，CIOの立場からプロジェクトチームがとるべきアクションを意思決定していき，与えられたミッションを円滑に達成するというものである．意思決定項目は，すべて言語で表現され，意思決定の結果も言語でフィードバックされる．一般的なビジネスゲームとは異なり，他チームとの競争は存在せず，単独チームにより，ゴールに到達するための正しい意思決定を模索するゴールシーク型のゲーム構造を有する．このため意思決定も，数値（販売価格など）ではなく，たとえば「業務の分析を行う」，「パッケージソフトを調査する」，「予算の増額を申請する」などの言語的定性的なアクションアイテムとなる．

このような構造のゲームの目的は，単独チーム内の複数のプレーヤによる集団意思決定を通じた，ゲームの主題に対する各プレーヤの持つ知識やノウハウの表出化と合意形成にある．すなわち，ゲーム実行を通して各プレーヤが討議し，ゲームの主題に対する各々の理解や現実のビジネス課題の解決への糸口を得ることを目指すものである．このため一連の意思決定を順序良く実行することで，ゲームのゴールに到達できるような意思決定の進展の連鎖をゲーム内に構築することが必要である．ただし，この連鎖は絶対的に正しいというような緻密なものである必要はなく，各プレーヤの議論を誘発するような程度の妥当性があれば十分である．

図１にCIO育成ゲームの意思決定進展図を示す．二重丸◎で示されるのは意思決定項目であり，丸印○は意思決定の結果として生じる事象を表している．また，実線の矢印は因果連鎖であり，点線の矢印は先行条件を表している．図２にCIO育成ゲームの意思決定入力画面を示す．意思決定項目は言語によるアクションアイテムがプルダウンメニューで示され，一度に２つの意思決定を行うことができる．

従来のような定量的ビジネスゲームは，数値データの入力によって進行し，状態を数式で表現できるような構造的な問題（企業のオペレーションなど）の最適解を探索するアプローチである．それに対し言語的定性的ビジネスゲームは，非構造的な問題である経営戦略策定を，創発的に促進するアプローチを目指すものである．これにより，複数のプレーヤが参加して，特定の経営課題について討議し，個々人の能力を組み合わせ，創発的な成果を生み出しながら，合意形成を目指すことで，実現可能性の高い企業戦略の立案を支援する効果が期待できる．

ビジネスゲームはシミュレーションの一種であるので，定量的ビジネスゲームおよび言語的定性的ビジネスゲームは，サイモンおよび飯島が整理しているように表１に示す意思決定技術に相当するものであると考えられる（サイモン 1979）（飯島 1993）．すなわち，定量的ビジネスゲームは構造的問題の現代的意思決定技術の（1）であるが，言語的定性的ビジネスゲームは非構造的問題の現代的意思決定技術の発見的問題開発法の新手法となりうる．

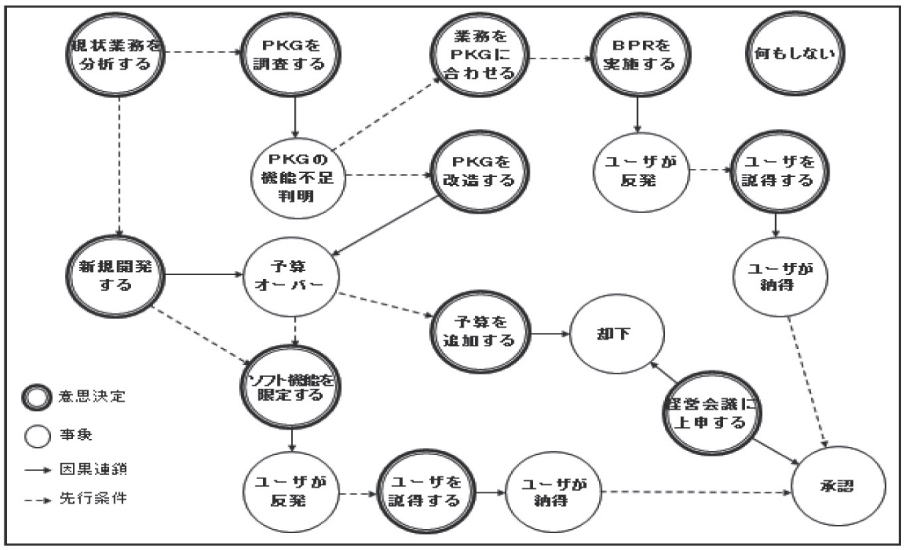

図１　CIO育成ゲームの意思決定進展図

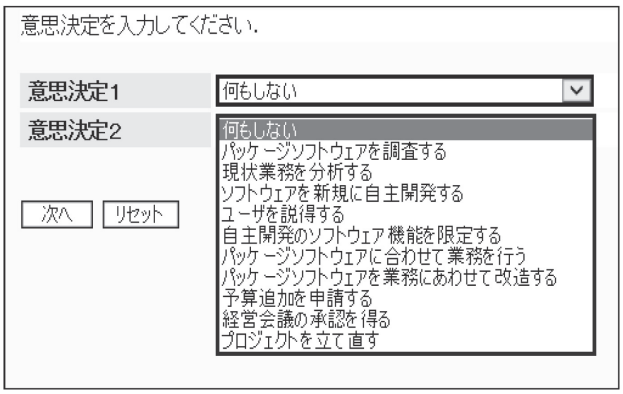

図２　CIO育成ゲームの意思決定入力画面

表１　問題の種類と意思決定技術

問題の種類	意思決定の種類	意思決定技術	
		伝統的	現代的
構造的	プログラム化しうるもの	(1)習慣 (2)事務上の慣例 (3)組織構造	(1)オペレーションズ・リサーチ 数学解析，モデル， コンピュータ・シミュレーション (2)電子計算機によるデータ処理
非構造的	プログラム化しえないもの	(1)判断，直感，創造力 (2)目の子算 (3)経営者の選抜と訓練	発見的問題解決法 (a)人間という意思決定者への訓練 (b)発見的なコンピュータ・プログラムの作成

（注）サイモン（1979）および飯島淳一（1993）をもとに著者作成.

３．モデル進化型ビジネスゲーム

適切なモデルを使ってゲーミングを行うことで，ゲーム参加者間の議論を誘発し，衆知を集めて合意を形成し，さらに参加者やゲーム開発者も気づいていなかった新しい概念を創発することが可能になると考えている．ＣＩＯ育成ゲームでは言語的データの選択によって進行するゲーミングシミュレーションの実装可能性は確認できたものの，操作する言語的データはあらかじめ設定されているものであった．そのためプレーヤによる創発性の発揮には限界がある．これを改善するためには，各プレーヤが自由に意思決定項目を文字として入力し，それをゲームにフィードバックしていくような仕組みが必要となる．つまりゲーム実行中に，プレーヤの気づきに応じて，ゲームが変化していくことになる．このようなビジネスゲームを「モデル進化型ビジネスゲーム」と呼ぶことにする．このモデル進化型ビジネスゲームの実現可能性を実証するため，汎用的なコンピュータプログラム言語を用いて，プロトタイプを試作した．

ここでは簡単なサンプルモデルとして，図３に示すように，意思決定のアクションに，「調査する」,「分析する」,「評価する」があり，それぞれを選択するとモデル内の状態が遷移し，メッセージとして，「調査完了しました」,「分析完了しました」,「評価完了しました」が出力されるものを想定する．また，状態が遷移するための先行条件として，たとえば「分析する」というアクションを選択した場合，その前に「調査する」というアクションが実行され，「調査完了しました」というメッセージが出力されていることが必要になるものとする．

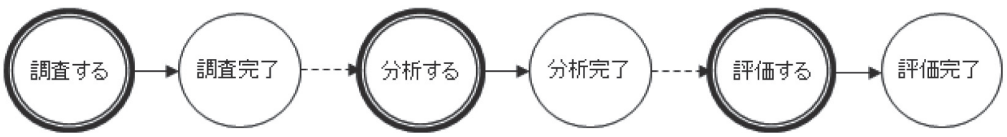

図３　サンプルモデルの構造

モデル進化型ビジネスゲームの構造を図４に示す．ゲーム編集モードでは，ゲームモデルの編集が行える．たとえばアクションとして「評価する」,メッセージとして「評価完了しました」などを追加すると，それぞれがアクションテーブルとメッセージテーブルに蓄積される．また，リンク設定画面から，あるアクションと，それに対応するメッセージのリンクを定義することができる．たとえばACT01「調査する」とMSG01「調査完了しました」をリンクする．さらに先行条件設定画面では，あるアクションが実行可能になるための先行条件であるメッセージを設定する．図３では，ACT02「分析する」の先行条件として，MSG01「調査完了しました」を設定している．この先行条件を満足しない場合には，あるアクションを選択しても無効となる．

ゲーム実行モードでは，人間プレーヤがアクション入力画面からアクションを選択し，それに対するメッセージ出力を見て，さらに次のアクションを選択することを繰り返して，ゲームの目的を達成していく．

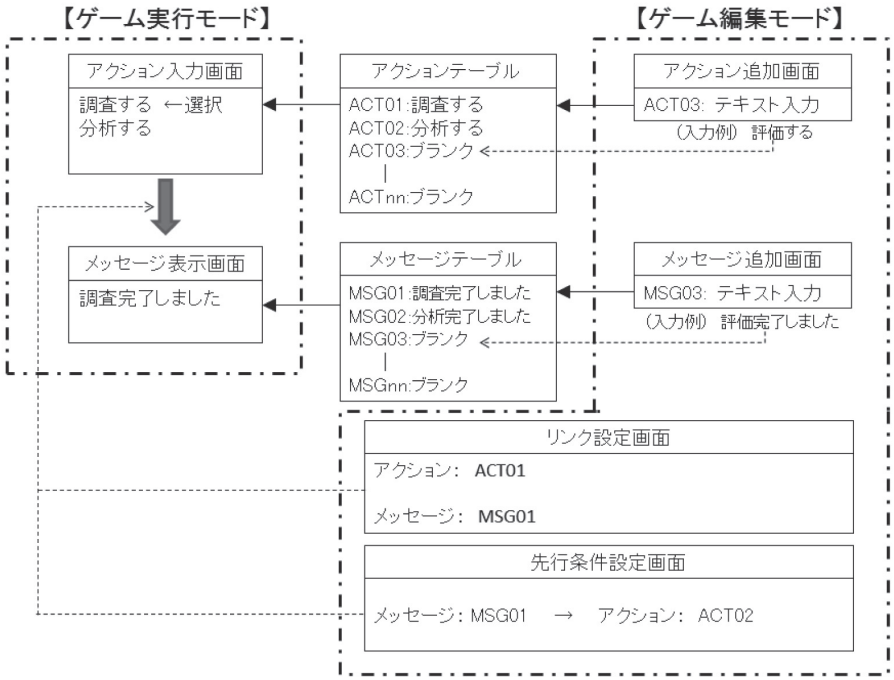

図４　モデル進化型ビジネスゲームの構造

４．プロトタイプの試作評価

モデル進化型ビジネスゲームのプロトタイプは，Active Basic（山本2004）を用いて開発した．このプロトタイプでは，初めに，ゲーム実行モードとゲーム編集モードを選択する．図５に示すように，最初はゲームのモデルにはアクションは何も定義されてないため，まず編集モードを選択して，アクションを入力していく．図６ではアクション「調査する」と「分析する」を入力している．

```
モデル進化型ビジネスゲーム プロトタイプ

 ゲームを実行する場合は   1
 ゲームを編集する場合は   2

? 1

##### アクションアイテム #####


##############################

アクションを選択してください．（はじめに戻る場合は99）
?
```

図５　ゲームの初期状態

```
 アクションを入力してください.
? 調査する
   1 調査する

 アクションを新規設定する場合は              11
 メッセージを新規設定する場合は              12
 アクションとメッセージをリンクする場合は    13
 アクションの先行条件を設定する場合は        14

 ゲーム実行にもどる場合は                    99

? 11

 アクションを入力してください.
? 分析する
   1 調査する
   2 分析する
```

図６　アクションの入力

次に,編集モードでメッセージを入力する．図７ではメッセージ「調査完了しました」と「分析完了しました」を入力している．さらに，アクションとメッセージのリンクを設定する．図８では，アクション「調査する」とメッセージ「調査完了しました」をリンクしている．

```
? 12

 メッセージを入力してください.
? 調査完了しました
調査完了しました
   1 調査完了しました

 アクションを新規設定する場合は              11
 メッセージを新規設定する場合は              12
 アクションとメッセージをリンクする場合は    13
 アクションの先行条件を設定する場合は        14

 ゲーム実行にもどる場合は                    99

? 12

 メッセージを入力してください.
? 分析完了しました
分析完了しました
   1 調査完了しました
   2 分析完了しました
```

図７　メッセージの入力

```
 ? 13


 アクション;
    1 調査する
    2 分析する
    3 評価する

 メッセージ;
    1 調査完了しました
    2 分析完了しました
    3 評価完了しました

 アクションとメッセージの番号 x,y を入力してください.
 ? 1,1

 調査する と 調査完了しました をリンクします.

 調査する 調査完了しました
```

図８　アクションとメッセージのリンク

この状態でゲーム実行モードに移ると，図９のように，設定されたアクションが表示され，たとえば番号１のアクション「調査する」を選択すると，モデル編集モードでリンクしたメッセージ「調査完了しました」が表示され，モデル内の状態が遷移したことを示している．

```
##### アクションアイテム #####
   1 調査する
   2 分析する
   3 評価する

################################
アクションを選択してください．（はじめに戻る場合は99）
? 1
調査完了しました
```

図９　３つのアクションが設定されたゲーム

続いてモデル編集モードで先行条件の設定を行う，図10に示すように，メッセージ「調査完了しました」を，アクション「分析する」の前提条件に設定すると，ゲーム実行モードでは，図11に示すように，番号２のアクション「分析する」をいきなり選択しても，先行条件となるメッセージ「調査完了しました」が成立していないのでアクションは無効となり，メッセージ「そのアクションは実行できません」と表示される．

以上のようにして図４の構造を実装することができた．

```
? 14


アクション；
   1 調査する
   2 分析する
   3 評価する

メッセージ；
   1 調査完了しました
   2 分析完了しました
   3 評価完了しました

アクションとその先行条件となるメッセージの番号 x,y を入力してください．
? 2,1

調査完了しました を 分析する の先行条件とします．

調査完了しました 分析する
```

図10　先行条件の設定

```
##### アクションアイテム #####
   1 調査する
   2 分析する
   3 評価する

##############################
アクションを選択してください．（はじめに戻る場合は99）
? 2
そのアクションは実行できません．
```

図11　先行条件を満足していない場合のメッセージ

５．今後の課題

これまで述べたように，汎用的なコンピュータプログラム言語Active Basicを用いてプロトタイプを試作したことにより，図４のような構造のモデル進化型ビジネスゲームが実現可能であるという見通しを得ることができた．これをさらに発展させて，従来のYBG言語により実現すれば，多くのYBGユーザが様々なゲームを開発できるようになり，現実のビジネスの諸問題の解決の一助となると期待される．

ビジネスゲームの教育への利用効果については確立してきているが，今後はさらに，研究や実際の問題解決のためのシミュレーションツールとしての活用が増えると思われる．これによって必ず成功する戦略戦術が発見できるとは限らないが，リスクを減らして成功確率を高めることは可能である．企業経営における諸問題の分析や改善，新たなビジネスモデルの開発などに対する「実験経営学」的な新しいアプローチを指向して，ビジネスシミュレーションのパラダイム革新を目指したい．

＜付記＞　本研究は科学技術融合振興財団の助成を受けている．

参　考　文　献

サイモン（稲葉・倉井訳）（1979），「意思決定の科学」，産業能率大学出版部
飯島淳一（1993），「意思決定支援システムとエキスパートシステム」，日科技連
白井宏明（2005），「ビジネスゲームのプラットフォーム」，経営システム，Vol.15，NO.4，pp.245-248
白井宏明（2010）「ビジネスゲームによる体験型教育」，内野明編『ビジネスインテリジェンスを育む教育』，白桃書房，第４章，73-98.
白井宏明（2013）「YBGを用いたイノベーション戦略の設計手法」，『日本シミュレーション＆ゲーミング学会全国大会論文報告集』，2013年秋号.
田名部元成，佐藤亮，白井宏明（2014），「言語的定性的ビジネスゲームとそのダイナミック・ケイパビリティ戦略論への展開」，横浜経営研究，Vol.35，No.2，pp.95-114
山本大祐（2004），「Active Basicオフィシャルユーザーズガイド」，毎日コミュニケーションズ

〔しらい　ひろあき　横浜国立大学大学院国際社会科学研究院教授〕
〔2016年２月１日受理〕

付録　モデル進化型ビジネスゲームのソースコード（Active Basic）

```
#N88BASIC
Line (0,0)-(1100,1000),7,B : paint (1,1),7 ,7 :color 0,7
'キー入力用
Dim a as Long
'For Next用カウンタ
Dim i As Long
'データ設定用
Dim outmsg[100] As String : Dim outmsgcnt As Long : Dim premsgcnt As Long
Dim condnmb[10] As Long : Dim actcnt As Long : Dim act[10] As String
Dim newact As String : Dim Linkact[10] As Long : Dim msgcnt As Long
Dim msg[10] As String : Dim newmsg As String : Dim Linkmsg[10] As Long
Dim Linkcnt As Long : Dim premsg[10] As String : Dim aftact[10] As String
'データ保存用
Dim Dataoutmsg[100] As String : Dim Dataoutmsgcnt As Long
Dim Datapremsgcnt As Long : Dim Datacondnmb[10] As Long
Dim Dataactcnt As Long : Dim Dataact[10] As String
Dim Datanewact As String : Dim DataLinkact[10] As Long
Dim Datamsgcnt As Long : Dim Datamsg[10] As String
Dim Datanewmsg As String : Dim DataLinkmsg[10] As Long
Dim DataLinkcnt As Long : Dim Datapremsg[10] As String
Dim Dataaftact[10] As String
‘カウンタ
actcnt=1 : msgcnt=1 : Linkcnt=1 : outmsgcnt=1 : premsgcnt=1
‘
Print:Print "モデル進化型ビジネスゲーム プロトタイプ":Print
10 Print
Print " ゲームを実行する場合は　1 "
Print " ゲームを編集する場合は　2 "
Print
Input a
‘
If a=1 Then
        Goto 100
Endif
If a=2 Then
        Goto 200
Endif
Goto 10
```

```
‘
100 Print
Print "##### アクションアイテム #####"
Print
For i=1 to 10
        If act[i] <> "" Then
          Print "  "; i ; act[i]:Print
        Endif
Next
Print
Print "##############################"
print:print "アクションを選択してください.（はじめに戻る場合は99）"
input a
'
If a=99 Then
        Goto 10
Endif
‘
110 If a=1 Then
        For i=1 to 10
                If act[1]=aftact[i] Then
                        For a=1 to 100
                          If outmsg(a)=premsg(i) Then
                              condnmb[1]=condnmb[1]-1
                              If condnmb[1]=0 Then   '複数の先行条件をチェック
                                Goto 111
                              Endif
                          Endif
                        Next
                        Goto 300
        Endif
    Next
111 For i=1 to 10
        IF act[Linkact[i]]=act[1] Then
          outmsg[outmsgcnt]=msg[Linkmsg[i]]
          Print outmsg[outmsgcnt]:outmsgcnt=outmsgcnt+1
        Endif
Next
Endif
‘
```

```
120 If a=2 Then
   For i=1 to 10
                If act[2]=aftact[i] Then
                        For a=1 to 100
                            If outmsg(a)=premsg(i) Then
                              condnmb[2]=condnmb[2]-1
                             If condnmb[2]=0 Then
                              Goto 121
                             Endif
                            Endif
                        Next
                        Goto 300
                Endif
   Next
121 For i=1 to 10
        IF act[Linkact[i]]=act[2] Then
                outmsg[outmsgcnt]=msg[Linkmsg[i]]
        Print outmsg[outmsgcnt]:outmsgcnt=outmsgcnt+1
        Endif
   Next
Endif
‘
130 If a=3 Then
        For i=1 to 10
                If act[3]=aftact[i] Then
                        For a=1 to 100
                              If outmsg(a)=premsg(i) Then
                                condnmb[3]=condnmb[3]-1
                             If condnmb[3]=0 Then
                              Goto 131
                             Endif
                            Endif
                        Next
                        Goto 300
     Endif
  Next
131 For i=1 to 10
        IF act[Linkact[i]]=act[3] Then
                outmsg[outmsgcnt]=msg[Linkmsg[i]]
        Print outmsg[outmsgcnt]:outmsgcnt=outmsgcnt+1
```

```
        Endif
Next
Endif
'
140 If a=4 Then
    For i=1 to 10
            If act[4]=aftact[i] Then
                For a=1 to 100
                    If outmsg(a)=premsg(i) Then
                      condnmb[4]=condnmb[4]-1
                     If condnmb[4]=0 Then
                        Goto 141
                     Endif
                    Endif
                Next
                Goto 300
            Endif
    Next
141 For i=1 to 10
        IF act[Linkact[i]]=act[4] Then
         outmsg[outmsgcnt]=msg[Linkmsg[i]]
         Print outmsg[outmsgcnt]:outmsgcnt=outmsgcnt+1
        Endif
Next
Endif
'
150 If a=5 Then
        For i=1 to 10
            If act[5]=aftact[i] Then
                For a=1 to 100
                    If outmsg(a)=premsg(i) Then
                      condnmb[5]=condnmb[5]-1
                     If condnmb[5]=0 Then
                        Goto 151
                     Endif
                    Endif
                Next
                Goto 300
        Endif
  Next
```

```
151 For i=1 to 10
        IF act[Linkact[i]]=act[5] Then
                outmsg[outmsgcnt]=msg[Linkmsg[i]]
        Print outmsg[outmsgcnt]:outmsgcnt=outmsgcnt+1
        endif
Next
Endif
goto 100
‘
200 Print
Print " アクションを新規設定する場合は　　11 "
Print " メッセージを新規設定する場合は　　12 "
Print " アクションとメッセージをリンクする場合は　13 "
Print " アクションの先行条件を設定する場合は　14 "
Print
Print " ゲーム実行にもどる場合は　　99 "
Print
Input a
Print
‘
If a=99 Then
        Goto 10
Endif
'アクションを新規設定
If a=11 Then
  Print " アクションを入力してください．"
  input newact
  act[actcnt]=newact : actcnt=actcnt+1
  For i=1 to 10
        If act[i] <> "" Then
          Print "  "; i ; act[i]
        Endif
  Next
Endif
'メッセージを新規設定
If a=12 Then
    Print " メッセージを入力してください．"
  input newmsg
  msg[msgcnt]=newmsg
  Print msg[msgcnt]
```

```
  msgcnt=msgcnt+1
    For i=1 to 10
          If msg[i] <> "" Then
             Print "  "; i ; msg[i]
          Endif
      Next
Endif
'アクションとメッセージをリンク
If a=13 Then
    Print
    Print "アクション；"  'アクションの一覧表示
    For i=1 to 10
          If act[i] <> "" Then
            Print "  "; i ; act[i]
          Endif
    Next
'
     Print
     Print "メッセージ；"   'メッセージの一覧表示
     For i=1 to 10
          If msg[i] <> "" Then
             Print "  "; i ; msg[i]
          Endif
     Next
    Print
'
    Dim X As Long : Dim Y As Long
    Print "アクションとメッセージの番号 x,y を入力してください．"
    input X,Y
    Print:Print act(X);"と";msg(Y);"をリンクします．"
    Linkact[Linkcnt]=X:Linkmsg[Linkcnt]=Y:Linkcnt=Linkcnt+1
    For i=1 to 10
     If act[Linkact[i]] <> "" Then
       Print:Print act[Linkact[i]];msg[Linkmsg[i]]
     Endif
    next
Endif
'先行条件の設定
If a=14 Then
          Print
```

```
    Print "アクション；"  'アクションの一覧表示
    For i=1 to 10
          If act[i] <> "" Then
            Print "  "; i ; act[i]
          Endif
    Next
         Print
  Print "メッセージ；"  'メッセージの一覧表示
  For i=1 to 10
           If msg[i] <> "" Then
             Print "  "; i ; msg[i]
           Endif
  Next
  Print : Print "アクションとその先行条件となるメッセージの番号 x,y を入力してください．  "
 input X,Y
 Print:Print msg(Y);"を";act(X);"の先行条件とします．  "
 premsg[premsgcnt]=msg(Y):aftact[premsgcnt]=act(X)
 condnmb[X]=condnmb[X]+1   '先行条件の数をカウント
 premsgcnt=premsgcnt+1
 For i=1 to 10
   If aftact[i] <> "" Then
     Print:Print premsg[i];aftact[i]
   Endif
 next
Endif
Goto 200
‘
300 print:print "そのアクションは実行できません．  ":Print:Goto 100
```