SOARS から SOARS CUI へのコンバータの作成
及び SOARS の機能強化
操作説明書

2021 年 3 月 10 日

(株)パイケーク

目次

1. はじめに 1-1
1.1. 動作環境 1-1
1.2. 起動方法 1-1
2. SOARS4 シミュレーションモデルの新規作成 2-1
3. Java プログラム出力 3-1
4. Java プログラム 4-3
4.1. Java プログラム構成 4-3
4.1.1. TMain.java 4-4
4.1.2. 変数及びクラス名 4-4
4.2. 各出力例 4-6
4.2.1. TStage.java 4-6
4.2.2. TAgentTypes.java 4-8
4.2.3. TSpotTypes.java 4-9
4.2.4. TMain.java 4-11
4.2.5. 初期ステージルール実行クラス 4-20
4.2.6. ロールクラス 4-21
4.2.7. メインステージルール実行クラス 4-23

図表目次

図 2-1 SOARS4 シミュレーションモデルの新規作成..... 2-1
図 3-1 Java プログラムのエクスポートメニュー..... 3-1
図 3-2 実験支援メニュー及びアイコン..... 3-1
図 3-3 実験支援編集画面..... 3-2
図 3-4 Java プログラムのエクスポートダイアログボックス..... 3-2
図 4-1 初期ステージ及びメインステージ定義ファイル出力例..... 4-7
図 4-2 エージェント定義ファイル出力例..... 4-8
図 4-3 スポット定義ファイル出力例..... 4-10
図 4-4 メインステージ定義例..... 4-11
図 4-5 スポット生成例..... 4-12
図 4-6 エージェント生成メソッド呼出例..... 4-13
図 4-7 エージェント生成メソッド例..... 4-13
図 4-8 スポット初期化メソッド呼出例..... 4-13
図 4-9 スポット初期化メソッド例..... 4-14
図 4-10 エージェント初期化メソッド呼出例..... 4-15
図 4-11 エージェント初期化メソッド例..... 4-15
図 4-12 初期ステージルール実行メソッド呼出例..... 4-17
図 4-13 エージェント初期ステージルール実行メソッド例..... 4-17
図 4-14 スポット初期ステージルール実行メソッド例..... 4-17
図 4-15 ログ出力開始例..... 4-18
図 4-16 各メインステージのルール実行(メインループ)例..... 4-19
図 4-17 ログ出力終了及びプログラム終了..... 4-19
図 4-18 初期ステージルール実行クラス例..... 4-20
図 4-19 ロールクラス例..... 4-22
図 4-20 ロールクラスの Getter /Setter メソッド例..... 4-22
図 4-21 メインステージのルール実行に際して呼び出されるメソッド doIt のオーバーライド例..... 4-24
図 4-22 メインステージのルール実行メソッド例..... 4-24
図 4-23 getMergedRole メソッドを使用したエージェントロール及びその変数へのアクセス..... 4-25
図 4-24 エージェントが現在存在しているスポットのスポットロール及び変数へアクセスする為のメソッド例..... 4-26
図 4-25 エージェントが現在存在しているスポットのスポットロール及び変数へのアクセス例..... 4-26

1. はじめに

この操作説明書には今回の開発で新たに追加された SOARS VisualShell4 から Java プログラムを出力する機能について記述されています。
尚、既存の機能については SOARS 操作説明書フォルダに同梱の各操作説明書を参照して下さい。

1.1. 動作環境

プログラムを実行する際には予め Java(バージョン 8 以上)がインストールされている必要があります。但し、出力される Java プログラムを実行する為には Java(バージョン 11)がインストールされている必要があります。

1.2. 起動方法

CD-ROM の「インストール用実行プログラム」ディレクトリに入っている「soars5.0.zip」を解凍し/soars/soars.jar を起動して下さい。

2. SOARS4 シミュレーションモデルの新規作成

SOARS Model Manager から新たに SOARS4 の新規シミュレーションモデルを作成できるようになりました。任意のフォルダ上でマウス右クリックするとコンテキストメニューが表示されるので、「新規シミュレーションモデル(SOARS4)」を選択して下さい。選択されているフォルダ内に新たな SOARS4 の新規シミュレーションモデルが作成されます。

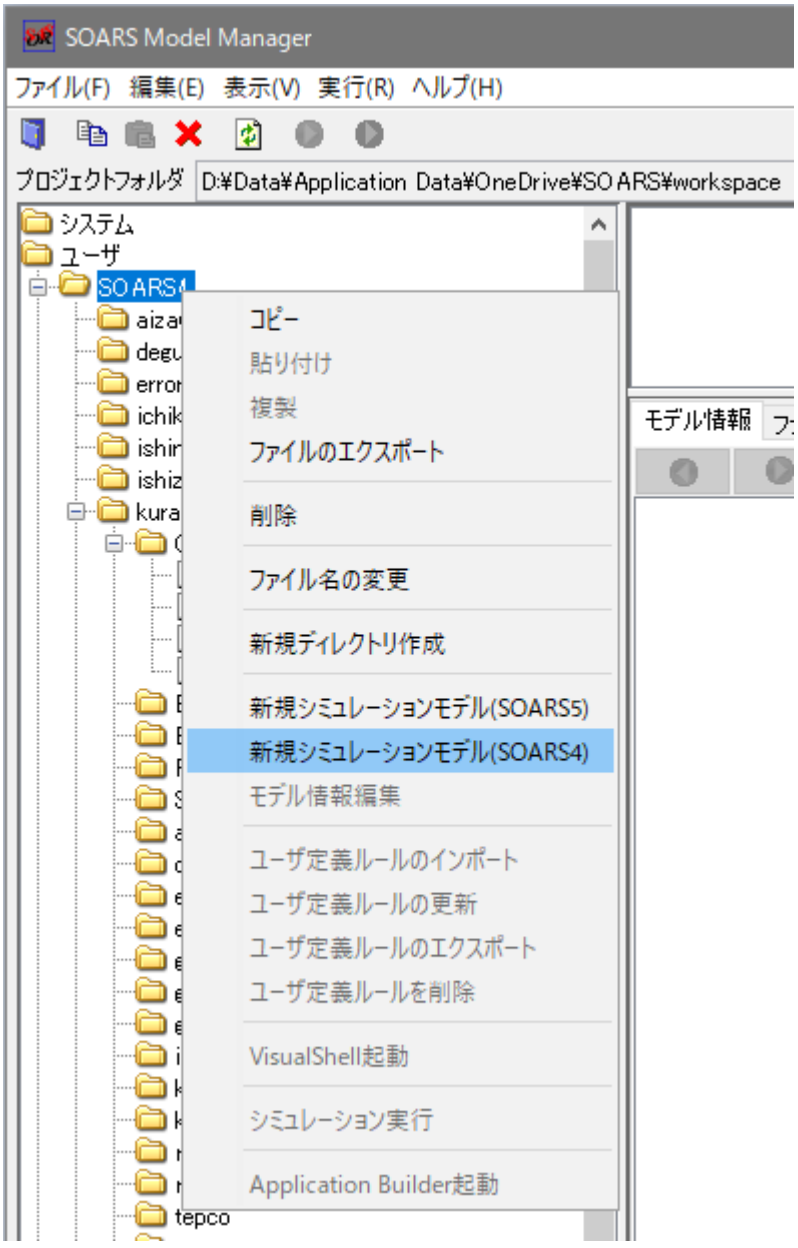


図 2-1 SOARS4 シミュレーションモデルの新規作成

3. Java プログラム出力

SOARS VisualShell4 のメニューから「ファイル(F)」→「Java プログラムのエクスポート」を選択して下さい。

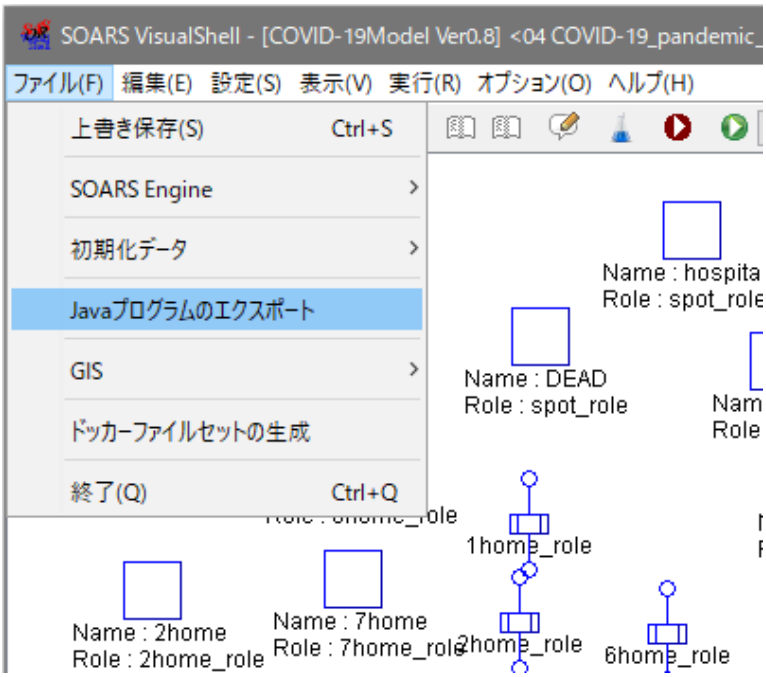


図 3-1 Java プログラムのエクスポートメニュー

実験支援を使用している場合は、SOARS VisualShell4 のメニューから「設定(S)」→「実験支援」を選択して下さい。またはルールバー上の実験支援アイコンを押して下さい。

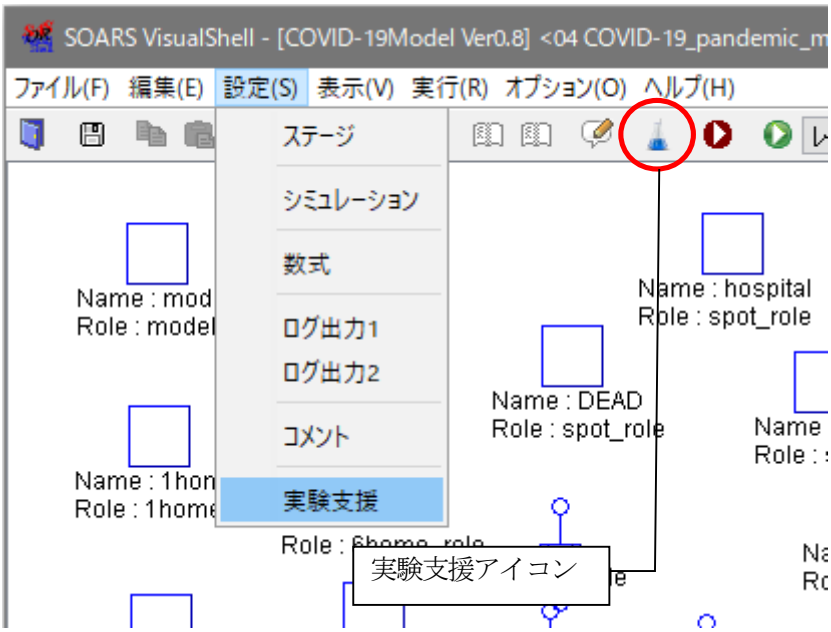


図 3-2 実験支援メニュー及びアイコン

実験支援編集画面が表示されるので、任意の実験を１つ選択し、行番号上でマウスを右クリックするとコンテキストメニューが表示されるので、「Java プログラムのエクスポート」を選択して下さい。

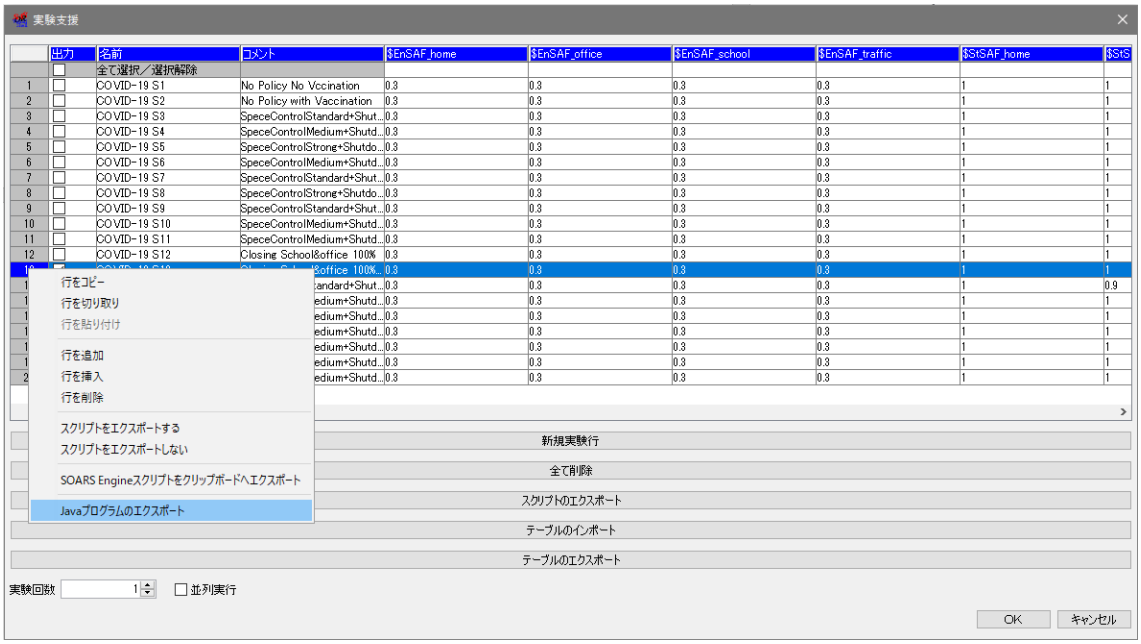


図 3-3 実験支援編集画面

Java プログラムのエクスポートダイアログボックスが表示されたら、参照ボタンを押して Java ソースフォルダを選択し、Java クラスパスを入力した後、インデントを指定して OK ボタンを押すと指定した場所に Java プログラム一式が出力されます。

Java クラスパスは、/ (スラッシュ) で区切って指定して下さい。先頭と最後尾に / (スラッシュ) は不要です。Java クラスパスには少なくとも１つのフォルダが必要になります。

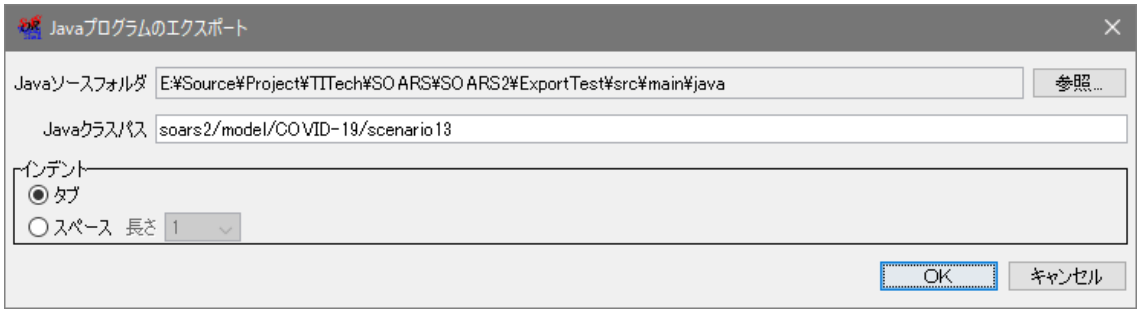
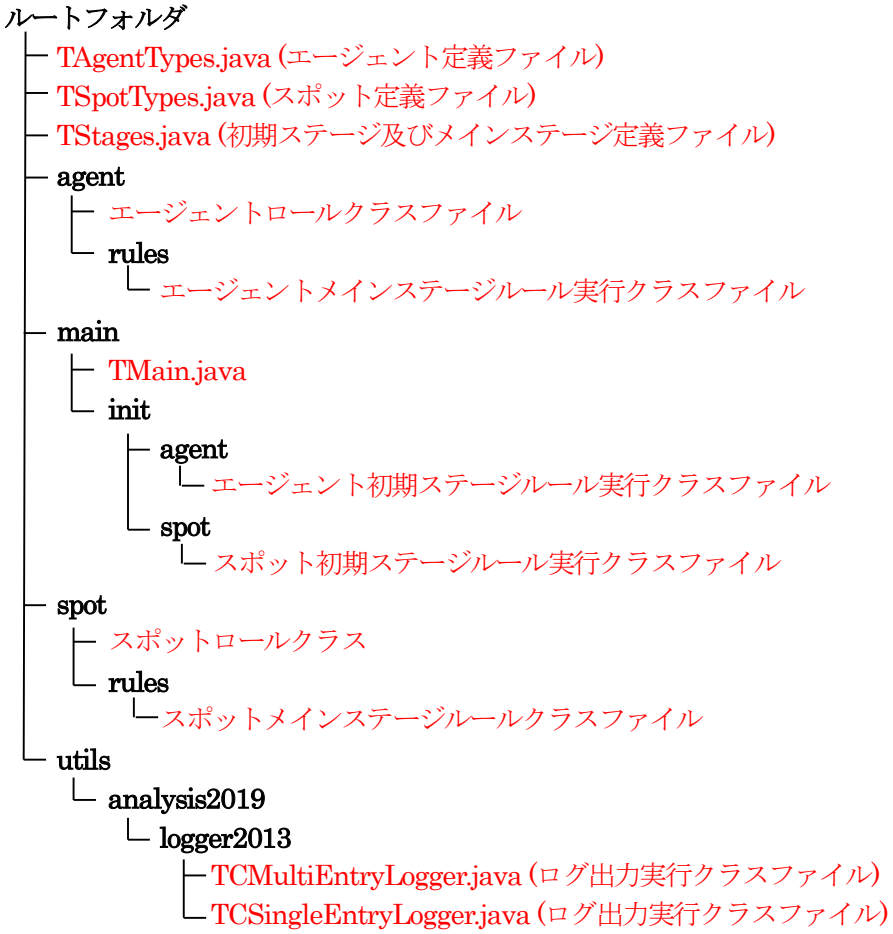


図 3-4 Java プログラムのエクスポートダイアログボックス

4. Java プログラム

4.1. Java プログラム構成

Java プログラムは、Java ソースフォルダと Java クラスパスで指定されるルートフォルダ直下に以下のフォルダ構成で出力されます。



4.1.1. TMain.java

TMain.java は main フォルダに出力される Java プログラムのエントリポイントであり、以下の順にプログラムを実行します。

・メインステージの定義
・スポット生成
・エージェント生成
・スポット初期化
・エージェント初期化
・初期ステージ実行
・ログ出力開始
・各メインステージのルール実行(メインループ)
・ログ出力終了及びプログラム実行終了

4.1.2. 変数及びクラス名

4.1.2.1. 変数名

4.1.2.1.1. ステージ名の変数名

TStages.java 内に定義されます。

・ステージ名を表す変数は全て大文字に変換されます。
・ステージ名が数字で始まる場合は先頭に”ST”がつきます。

4.1.2.1.2. エージェント名の変数名

TAgentTypes.java 内に定義されます。

・ステージ名を表す変数は全て大文字に変換されます。
・ステージ名が数字で始まる場合は先頭に”A”がつきます。

4.1.2.1.3. スポット名の変数名

TSpotTypes.java 内に定義されます。

・ステージ名を表す変数は全て大文字に変換されます。
・ステージ名が数字で始まる場合は先頭に”S”がつきます。

4.1.2.2. クラス名

4.1.2.2.1. 初期ステージルール実行クラス

エージェント初期ステージ実行クラスのファイルは main/init/agent フォルダに、スポット初期ステージ実行クラスのファイルは main/init/spot フォルダに出力されます。初期ステージ実行クラスは、初期ステージのルールを実行します。

・”T”＋先頭を大文字とした初期ステージ名と云うクラス名になります。

4.1.2.3. ロールクラス

エージェントロールクラスのファイルは agent フォルダに、スポットロールクラスのファイルは spot フォルダに出力されます。ロールクラスは、メインステージルール実行クラス(メインステージで実行されるルールの定義クラス)のインスタンス及び使用される変数を保持します。

・エージェントの無名ロールクラス名は"TAgentRole"になります。
・"T"＋先頭を大文字としたロール名＋"Role"と云うクラス名になります。
・"-"及び"_"は削除されてその次の文字が大文字になります。

4.1.2.4. メインステージルール実行クラス

エージェントルールクラスのファイルは agent/rules フォルダに、スポットロールクラスのファイルは spot/rules フォルダに出力されます。ルールクラスには、メインステージで実行されるルールのメソッドが定義されます。

・上記ロール名＋先頭を大文字としたステージ名＋１からのインデクス＋"Rule"と云うクラス名になります。

4.2. 各出力例

4.2.1. TStage.java

初期ステージ及びメインステージの定義ファイル例です。

```
public class TStages {

  //
  // 初期ステージ
  //

  /** 1. */
  public static final String INITADS = "initADS";

  /** 2. */
  public static final String INITSPOT = "initSpot";

  /** 3. */
  public static final String INITSPOT2 = "initSpot2";

  /** 4. */
  public static final String INITAGENT2 = "initAgent2";

  /** 5. */
  public static final String INITAGENT = "initAgent";

  /** 6. */
  public static final String INITVACCINE = "initVaccine";

  //
  // メインステージ
  //

  /** 1. */
  public static final String FEEDBACK = "feedback";

  /** 2. */
  public static final String FEEDBACK2 = "feedback2";

  /** 3. */
  public static final String CLEARAHL = "clearAHL";

  /** 4. */
  public static final String CALCAES = "calcAES";

```

```
  /** 5. */
  public static final String CALCAHL = "calcAHL";

  /** 6. */
  public static final String CALCDRSCL = "calcDRSCL";

  /** 7. */
  public static final String VACCINE0 = "vaccine0";

  /** 8. */
  public static final String VACCINE1 = "vaccine1";

  /** 9. */
  public static final String VACCINE2 = "vaccine2";

  /** 10. */
  public static final String CALCP = "calcP";

  /** 11. */
  public static final String UPDATEADS = "updateADS";

  /** 12. */
  public static final String CALCPATIENT_ONBED = "calcpatient_onbed";

  /** 13. */
  public static final String MOVE = "move";

  /** 14. */
  public static final String CHECK = "check";
}
```

図 4-1 初期ステージ及びメインステージ定義ファイル出力例

4.2.2. TAgentTypes.java

エージェント定義ファイル例です。

```
public class TAgentTypes {

  /**  */
  public static final String BABY = "baby";

  /**  */
  public static final String STUDENT = "student";

  /**  */
  public static final String YOUNG = "young";

  /**  */
  public static final String MIDDLE = "middle";

  /**  */
  public static final String SCHOOLCHILD = "schoolchild";

  /**  */
  public static final String OLD = "old";
}
```

図 4-2 エージェント定義ファイル出力例

4.2.3. TSpotTypes.java

スポット定義ファイル例です。

```
public class TSpotTypes {

  /**  */
  public static final String PRIMARYSCHOOL = "primaryschool";

  /**  */
  public static final String YARD = "yard";

  /**  */
  public static final String BIGOFFICE = "bigoffice";

  /**  */
  public static final String DEAD = "DEAD";

  /**  */
  public static final String HIGHSCHOOL = "highschool";

  /**  */
  public static final String SMALLOFFICE = "smalloffice";

  /**  */
  public static final String S3HOME = "3home";

  /**  */
  public static final String S4HOME = "4home";

  /**  */
  public static final String S8HOME = "8home";

  /**  */
  public static final String S7HOME = "7home";

  /**  */
  public static final String S5HOME = "5home";

  /**  */
  public static final String S1HOME = "1home";

  /**  */
  public static final String S2HOME = "2home";

  /**  */
```

```
  public static final String S10HOME = "10home";

  /**  */
  public static final String MIDDLEOFFICE = "middleoffice";

  /**  */
  public static final String S9HOME = "9home";

  /**  */
  public static final String S6HOME = "6home";

  /**  */
  public static final String TRAFFIC = "traffic";

  /**  */
  public static final String MODEL = "model";

  /**  */
  public static final String HOSPITAL = "hospital";
}
```

図 4-3 スポット定義ファイル出力例

4.2.4. TMain.java

4.2.4.1. メインステージの定義

メインステージを実行順に定義します。

```
List<String> mainStages = List.of( // メインステージ
  TStages.FEEDBACK,
  TStages.FEEDBACK2,
  TStages.CLEARAHL,
  TStages.CALCAES,
  TStages.CALCAHL,
  TStages.CALCDRSCL,
  TStages.VACCINE0,
  TStages.VACCINE1,
  TStages.VACCINE2,
  TStages.CALCP,
  TStages.UPDATEADS,
  TStages.CALCPATIENT_ONBED,
  TStages.MOVE,
  TStages.CHECK);
```

図 4-4 メインステージ定義例

4.2.4.2. スポット生成

TSpotManager クラスの createSpot メソッド及び createSpots メソッドにより各スポットを生成します。

```
spotManager.createSpots(TSpotTypes.S10HOME, 31);
spotManager.createSpots(TSpotTypes.S1HOME, 2000);
spotManager.createSpots(TSpotTypes.S2HOME, 1500);
spotManager.createSpots(TSpotTypes.S3HOME, 500);
spotManager.createSpots(TSpotTypes.S4HOME, 250);
spotManager.createSpots(TSpotTypes.S5HOME, 200);
spotManager.createSpots(TSpotTypes.S6HOME, 83);
spotManager.createSpots(TSpotTypes.S7HOME, 42);
spotManager.createSpots(TSpotTypes.S8HOME, 25);
spotManager.createSpots(TSpotTypes.S9HOME, 22);
spotManager.createSpot(TSpotTypes.DEAD);
spotManager.createSpots(TSpotTypes.BIGOFFICE, 4);
spotManager.createSpots(TSpotTypes.HIGHSCHOOL, 3);
spotManager.createSpot(TSpotTypes.HOSPITAL);
spotManager.createSpots(TSpotTypes.MIDDLEOFFICE, 7);
spotManager.createSpot(TSpotTypes.MODEL);
spotManager.createSpots(TSpotTypes.PRIMARYSCHOOL, 3);
spotManager.createSpots(TSpotTypes.SMALLOFFICE, 7);
spotManager.createSpots(TSpotTypes.TRAFFIC, 40);
spotManager.createSpot(TSpotTypes.YARD);
```

図 4-5 スポット生成例

4.2.4.3. エージェント生成

TAgentManager クラスの createAgent メソッド及び createAgents メソッドにより各エージェントを生成します。

```
createAgent(agentManager, spotManager, TAgentTypes.BABY, 1000, "");
createAgent(agentManager, spotManager, TAgentTypes.MIDDLE, 2000, "");
createAgent(agentManager, spotManager, TAgentTypes.OLD, 1000, "");
createAgent(agentManager, spotManager, TAgentTypes.SCHOOLCHILD, 2000, "");
createAgent(agentManager, spotManager, TAgentTypes.STUDENT, 2000, "");
createAgent(agentManager, spotManager, TAgentTypes.YOUNG, 2000, "");
```

図 4-6 エージェント生成メソッド呼出例

```
private static void createAgent(TAgentManager agentManager,
  SpotManager spotManager, String agentName, int noOfAgents, String initialSpot) {
  ArrayList<TAgent> agents = agentManager.createAgents(agentName, noOfAgents);
  if (initialSpot.equals(""))
    return;
  for (TAgent agent:agents)
    agent.initializeCurrentSpot(initialSpot, spotManager.getSpotDB());
}
```

図 4-7 エージェント生成メソッド例

4.2.4.4. スポット初期化

各スポットの初期ロールクラスを生成し、ロールクラスの変数に初期値を設定します。

```
initializeSpot1(spotManager, agentManager, rand);
initializeSpot2(spotManager, agentManager, rand);
initializeSpot3(spotManager, agentManager, rand);
initializeSpot4(spotManager, agentManager, rand);
initializeSpot5(spotManager, agentManager, rand);
initializeSpot6(spotManager, agentManager, rand);
initializeSpot7(spotManager, agentManager, rand);
initializeSpot8(spotManager, agentManager, rand);
initializeSpot9(spotManager, agentManager, rand);
initializeSpot10(spotManager, agentManager, rand);
initializeSpot11(spotManager, agentManager, rand);
initializeSpot12(spotManager, agentManager, rand);
initializeSpot13(spotManager, agentManager, rand);
initializeSpot14(spotManager, agentManager, rand);
initializeSpot15(spotManager, agentManager, rand);
initializeSpot16(spotManager, agentManager, rand);
initializeSpot17(spotManager, agentManager, rand);
initializeSpot18(spotManager, agentManager, rand);
initializeSpot19(spotManager, agentManager, rand);
initializeSpot20(spotManager, agentManager, rand);
```

図 4-8 スポット初期化メソッド呼出例

```
private static void initializeSpot16(TSpotManager spotManager,
  TAgentManager agentManager, ICRandom rand) {
  TSpot spot = spotManager.getSpotDB().get(TSpotTypes.MODEL);
  ArrayList<Object> initia_linfected = new ArrayList<>();
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.YOUNG+"1"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.YOUNG+"2"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.YOUNG+"3"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.YOUNG+"4"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.YOUNG+"5"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.SCHOOLCHILD+"1"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.SCHOOLCHILD+"2"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.SCHOOLCHILD+"3"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.SCHOOLCHILD+"4"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.SCHOOLCHILD+"5"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.STUDENT+"1"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.STUDENT+"2"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.STUDENT+"3"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.STUDENT+"4"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.STUDENT+"5"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.MIDDLE+"1"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.MIDDLE+"2"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.MIDDLE+"3"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.MIDDLE+"4"));
  initia_linfected.add(agentManager.getAgentDB().get(TAgentTypes.MIDDLE+"5"));
  TModelRole role = new TModelRole(spot, rand,
    4000/4770, 0/100, 80/100, 700/770, 0.0, "no", "no", "all", 0.0, 0.0, 0.0, 0.0,
    0.0, 0.0, 0.0, 0, 0,  0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 10000, 0.0,
    0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 250, new ArrayList<>(),
    initia_linfected, new ArrayList<>(), new ArrayList<>(), new ArrayList<>(),
    new ArrayList<>(), new ArrayList<>(), new ArrayList<>(), new ArrayList<>());
  spot.setBaseRole(role);
}
```

図 4-9 スポット初期化メソッド例

4.2.4.5. エージェント初期化

各エージェントの初期ロールクラスを生成し、ロールクラスの変数に初期値を設定します。

```
initializeAgent1(agentManager, spotManager, rand);
initializeAgent2(agentManager, spotManager, rand);
initializeAgent3(agentManager, spotManager, rand);
initializeAgent4(agentManager, spotManager, rand);
initializeAgent5(agentManager, spotManager, rand);
initializeAgent6(agentManager, spotManager, rand);
```

図 4-10 エージェント初期化メソッド呼出例

```
private static void initializeAgent1(TAgentManager agentManager,
  TSpotManager spotManager, ICRandom rand) {
  for (int i = 1; i <= agentManager.getNoOfAgents(TAgentTypes.BABY); ++i) {
    TAgent agent = agentManager.getAgentDB().get(TAgentTypes.BABY+String.valueOf(i));
    TAgentRole agentRole = new TAgentRole(agent, rand,
      80/100, 0.8, 0.8, 0.2, "0", "baby", 0.0, 1, 0.0, 0.0, 0.0, 0.3, 0.9, 0.0, 0.8, 0.0,
      1, 0.008, new TTime("0/0:00"), new ArrayList<>());
    TBabyMoveRole babyMoveRole = new TBabyMoveRole(agent, rand);
    babyMoveRole.mergeRole(agentRole); // 統合
    agent.setBaseRole(babyMoveRole); // 基本役割の設定
  }
}
```

図 4-11 エージェント初期化メソッド例

4.2.4.6. 初期ステージ実行

初期ステージのルールを定義順に実行します。

```
doTInitADS1(TAgentTypes.BABY, agentManager, spotManager, rand);
doTInitADS1(TAgentTypes.STUDENT, agentManager, spotManager, rand);
doTInitADS1(TAgentTypes.YOUNG, agentManager, spotManager, rand);
doTInitADS1(TAgentTypes.MIDDLE, agentManager, spotManager, rand);
doTInitADS1(TAgentTypes.SCHOOLCHILD, agentManager, spotManager, rand);
doTInitADS1(TAgentTypes.OLD, agentManager, spotManager, rand);

doTInitSpot1(TSpotTypes.PRIMARYSCHOOL, spotManager, rand);
doTInitSpot2(TSpotTypes.YARD, spotManager, rand);
doTInitSpot3(TSpotTypes.BIGOFFICE, spotManager, rand);
doTInitSpot2(TSpotTypes.DEAD, spotManager, rand);
doTInitSpot4(TSpotTypes.HIGHSCHOOL, spotManager, rand);
doTInitSpot5(TSpotTypes.SMALLOFFICE, spotManager, rand);
doTInitSpot6(TSpotTypes.S3HOME, spotManager, rand);
doTInitSpot7(TSpotTypes.S4HOME, spotManager, rand);
doTInitSpot8(TSpotTypes.S8HOME, spotManager, rand);
doTInitSpot9(TSpotTypes.S7HOME, spotManager, rand);
doTInitSpot10(TSpotTypes.S5HOME, spotManager, rand);
doTInitSpot11(TSpotTypes.S1HOME, spotManager, rand);
doTInitSpot12(TSpotTypes.S2HOME, spotManager, rand);
doTInitSpot13(TSpotTypes.S10HOME, spotManager, rand);
doTInitSpot14(TSpotTypes.MIDDLEOFFICE, spotManager, rand);
doTInitSpot15(TSpotTypes.S9HOME, spotManager, rand);
doTInitSpot16(TSpotTypes.S6HOME, spotManager, rand);
doTInitSpot17(TSpotTypes.TRAFFIC, spotManager, rand);
doTInitSpot2(TSpotTypes.HOSPITAL, spotManager, rand);

doTInitSpot21(TSpotTypes.MODEL, spotManager, rand);

doTInitAgent21(TAgentTypes.STUDENT, agentManager, spotManager, rand);
doTInitAgent22(TAgentTypes.YOUNG, agentManager, spotManager, rand);
doTInitAgent23(TAgentTypes.MIDDLE, agentManager, spotManager, rand);
doTInitAgent24(TAgentTypes.SCHOOLCHILD, agentManager, spotManager, rand);

doTInitAgent1(TAgentTypes.BABY, agentManager, spotManager, rand);
doTInitAgent1(TAgentTypes.STUDENT, agentManager, spotManager, rand);
doTInitAgent1(TAgentTypes.YOUNG, agentManager, spotManager, rand);
doTInitAgent1(TAgentTypes.MIDDLE, agentManager, spotManager, rand);
doTInitAgent1(TAgentTypes.SCHOOLCHILD, agentManager, spotManager, rand);
doTInitAgent1(TAgentTypes.OLD, agentManager, spotManager, rand);

doTInitVaccine1(TAgentTypes.BABY, agentManager, spotManager, rand);
doTInitVaccine2(TAgentTypes.STUDENT, agentManager, spotManager, rand);
```

```
doTInitVaccine3(TAgentTypes.YOUNG, agentManager, spotManager, rand);
doTInitVaccine4(TAgentTypes.MIDDLE, agentManager, spotManager, rand);
doTInitVaccine5(TAgentTypes.SCHOOLCHILD, agentManager, spotManager, rand);
doTInitVaccine6(TAgentTypes.OLD, agentManager, spotManager, rand);
```

図 4-12 初期ステージルール実行メソッド呼出例

```
private static void doTInitAgent22(String agentName,
  TAgentManager agentManager, TSpotManager spotManager, ICRandom rand) {
  for (int i = 1; i <= agentManager.getNoOfAgents(agentName); ++i) {
    TAgent agent = agentManager.getAgentDB().get(agentName+String.valueOf(i));
    TSpot spot = spotManager.getSpotDB().get(agent.getCurrentSpotName());
    if (TInitAgent2.init5(agent, spot, (TYoungMoveRole)agent.getBaseRole(),
      agentManager, spotManager, rand))
      continue;
    if (TInitAgent2.init6(agent, spot, (TYoungMoveRole)agent.getBaseRole(),
      agentManager, spotManager, rand))
      continue;
    TInitAgent2.init7(agent, spot, (TYoungMoveRole)agent.getBaseRole(),
      agentManager, spotManager, rand);
  }
}
```

図 4-13 エージェント初期ステージルール実行メソッド例

```
private static void doTInitSpot1(String spotName, TSpotManager spotManager,
  ICRandom rand) {
  for (int i = 1; i <= spotManager.getNoOfSpots(spotName); ++i) {
    TSpot spot = spotManager.getSpotDB().get(spotName+String.valueOf(i));
    TInitSpot.init6(spot, (TBigofficeRole)spot.getBaseRole(), spotManager, rand);
  }
}
```

図 4-14 スポット初期ステージルール実行メソッド例

4.2.4.7. ログ出力開始

SOARS VisualShell4 モデルでログ出力するよう指定された変数のログ出力を開始します。但し、ログ出力出来る変数のタイプはキーワード、確率変数及び数値変数のみとなります。

```
TAgentLogger agentLocationLog = new TAgentLogger(logDir + File.separator
  + "agentLocation.csv",TAgentLogger.CURRENT_SPOT_NAME_KEY, agentManager);
String[] activeKeys = { "time", "InfectionLevel11", "dead1", "infected1", "naive1",
  "patientFlowOver11", "patientFlowOver21", "patient_severe_critical1", "vaccinated1"};

TCMultiEntryLogger logger = new TCMultiEntryLogger(logDir + File.separator
  + "log", activeKeys); // ロガーを作成

logger.beginLog(); // ロギング開始
```

図 4-15 ログ出力開始例

4.2.4.8. 各メインステージのルール実行(メインループ)

各メインステージのルールを実行し、指定されたログの出力を行います。

```
for (TTime t = new TTime("0/00:00"); t.isLessThan("300/01:00"); t.add("0:30")) {
  for (String stage : mainStages) {
    // 時刻 t, ステージ stage に登録されたルールを実行する
    ruleAggregator.executeStage(t, stage, spotManager, agentManager, globalSharedVariableSet);
  }
  if (t.isEqualTo("0:00")) {
    System.out.println(t); // 現在時刻を画面に表示する
  }
  agentLocationLog.output(t); // 現在時刻の各エージェントの位置をログに出力する
  logger.beginEntry(true); // ログのエントリ開始
  logger.putData("time", t.toString()); // 時間をログに出力
  logger.putData("InfectionLevel11",
    ((TModelRole)spotManager.getSpotDB().get(TSpotTypes.MODEL).getRole()).getInfectionLevel1());
  logger.putData("dead1",
    ((TModelRole)spotManager.getSpotDB().get(TSpotTypes.MODEL).getRole()).getdead());
  logger.putData("infected1",
    ((TModelRole)spotManager.getSpotDB().get(TSpotTypes.MODEL).getRole()).getinfected());
  logger.putData("naive1",
    ((TModelRole)spotManager.getSpotDB().get(TSpotTypes.MODEL).getRole()).getnaive());
  logger.putData("patientFlowOver11",
    ((TModelRole)spotManager.getSpotDB().get(TSpotTypes.MODEL).getRole()).getpatientFlowOver1());
  logger.putData("patientFlowOver21",
    ((TModelRole)spotManager.getSpotDB().get(TSpotTypes.MODEL).getRole()).getpatientFlowOver2());
  logger.putData("patient_severe_critical1",
    ((TModelRole)spotManager.getSpotDB().get(TSpotTypes.MODEL).getRole()).getpatient_severe_critical());
  logger.putData("vaccinated1",
    ((TModelRole)spotManager.getSpotDB().get(TSpotTypes.MODEL).getRole()).getvaccinated());
  logger.endEntry(); // ログのエントリ終了
}
```

図 4-16 各メインステージのルール実行(メインループ)例

4.2.4.9. ログ出力終了及びプログラム実行終了

```
agentLocationLog.close();
logger.endLog();
long elapsedTime = System.currentTimeMillis() - startTime;
System.out.println(elapsedTime + " [msec]");
```

図 4-17 ログ出力終了及びプログラム終了

4.2.5. 初期ステージルール実行クラス

初期ステージルール実行クラスのメソッドは全てスタティックメソッドとなっています。SOARS では、ルール１行に複数の if 文を定義するこが可能であり、また、ルールの先頭は if 文とは限りません。更に、ルールの途中で if 文が偽となった場合は次のルールを実行する必要があるので、ルール１行を１メソッドとしています。

```
public static boolean init1(TAgent agent, TSpot spot, TAgentRole ownerRole, TAgentManager agentManager,
  TSpotManager spotManager, ICRandom rand) {
  TModelRole modelRole = (TModelRole) spotManager.getSpotDB().get(TSpotTypes.MODEL).getBaseRole();
  // condition : collection : <model>containsAgent initia_linfected
  if (modelRole.getinitia_linfected().contains(agent)) {
    // command : keyword : set ADS=1
    ownerRole.setADS("1");
    // command : time : setTime timer=timer ; setTime timer+=0/0:00
    ownerRole.settimer(new TTime("0/0:00").add("0/0:00"));
    // command : substitution : <model>askEquip infected=infected+1
    modelRole.setinfected(modelRole.getinfected()+1);
    // command : substitution : <model>askEquip patientpluusdead=patientpluusdead+1
    modelRole.setpatientpluusdead(modelRole.getpatientpluusdead()+1);
    // command : substitution : <model>askEquip InfectionLevel1=InfectionLevel1+1
    modelRole.setInfectionLevel1(modelRole.getInfectionLevel1()+1);
    return true;
  }
  return false;
}
```

図 4-18 初期ステージルール実行クラス例

4.2.6. ロールクラス

ロールクラスは、変数を保持し、そのコンストラクタでメインステージのルールを実行するクラスのインスタンスを生成します。また、各変数の Getter / Setter メソッドを定義することにより、メインステージルール実行クラスがこれらの変数を使用することが出来ます。

メインステージで実行するルールが全く同じロールは１つのロールクラスに集約されます。

```
public TAgentRole(TAgent ownerAgent, ICRandom rand,
  double p12, double p23, double p3s4m, double p4cd, String ADS, String age,
  double ACL, double ACPF, double AES, double AHL, double AgentAF, double EPF,
  double EnAAF, double P, double PC, double SHLAP, double StAAF, double a,
  TTime timer, ArrayList<Object> home) {
  super(ROLE_NAME, ownerAgent, rand); // 親クラスのコンストラクタを呼び出す
  fp12=p12;
  fp23=p23;
  fp3s4m=p3s4m;
  fp4cd=p4cd;
  fADS=ADS;
  fage=age;
  fACL=ACL;
  fACPF=ACPF;
  fAES=AES;
  fAHL=AHL;
  fAgentAF=AgentAF;
  fEPF=EPF;
  fEnAAF=EnAAF;
  fP=P;
  fPC=PC;
  fSHLAP=SHLAP;
  fStAAF=StAAF;
  fa=a;
  ftimer=timer;
  fhome=home;
  // TStages.CALCAES クラス
  new TAgentRoleCalcAES1Rule(this).setTimeAndStage(new TTime("0:00"),
    new TTime("23:30"), new TTime("0:30"), TStages.CALCAES);
  // TStages.CALCAHL クラス
  new TAgentRoleCalcAHL1Rule(this).setTimeAndStage(new TTime("0:00"),
    new TTime("23:30"), new TTime("0:30"), TStages.CALCAHL);
  // TStages.VACCINE1 クラス
  new TAgentRoleVaccine11Rule(this).setTimeAndStage(true, new TTime("8:00"),
    TStages.VACCINE1);
  // TStages.VACCINE2 クラス
  new TAgentRoleVaccine21Rule(this).setTimeAndStage(true, new TTime("8:00"),
    TStages.VACCINE2);

```

```
  // TStages.CALCP クラス
  new TAgentRoleCalcP1Rule(this).setTimeAndStage(new TTime("0:00"),
    new TTime("23:30"), new TTime("0:30"), TStages.CALCP);
  // TStages.UPDATEADS クラス
  new TAgentRoleUpdateADS1Rule(this).setTimeAndStage(new TTime("0:00"),
    new TTime("23:30"), new TTime("0:30"), TStages.UPDATEADS);
  // TStages.CALCPATIENT_ONBED クラス
  new TAgentRoleCalcpatientOnbed1Rule(this).setTimeAndStage(new TTime("0:00"),
    new TTime("23:30"), new TTime("0:30"), TStages.CALCPATIENT_ONBED);
  // TStages.CHECK クラス
  new TAgentRoleCheck1Rule(this).setTimeAndStage(new TTime("0:00"),
    new TTime("23:30"), new TTime("0:30"), TStages.CHECK);
}
```

図 4-19 ロールクラス例

```
//
// Getter and Setter
//

public double getp12() {
  return this.fp12;
}

public void setp12(double fp12) {
  this.fp12=fp12;
}
```

図 4-20 ロールクラスの Getter /Setter メソッド例

4.2.7. メインステージルール実行クラス

メインステージルール実行クラスは、メインステージのルール実行に際して呼び出されるメソッド doIt をオーバーライドしています。

メインステージルール実行クラスは以下の規則に基づいて作成します。

・１つのメインステージ毎に１つのメインステージルール実行クラスを作成します。
・但し、特定時刻に実行されるルールはこれを１つのメインステージルール実行クラスとします。
・また、２つ以上のロール内のメインステージに共通のルールが存在する場合は１つのメインステージルール実行クラスに集約します。

SOARS では、ルール１行に複数の if 文を定義するこが可能であり、また、ルールの先頭は if 文とは限りません。更に、ルールの途中で if 文が偽となった場合は次のルールを実行する必要があるので、ルール１行を１メソッドとしています。

```
@Override
public boolean doIt(TTime currentTime, String currentStage, HashMap<String, TSpot> spotSet,
  HashMap<String, TAgent> agentSet, HashMap<String, Object> globalSharedVariables) {
  if (!meetSpotCondition())
    return false;
  //
  if (method1(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  //
  else if (method2(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  //
  else if (method3(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  //
  else if (method4(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  //
  else if (method5(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  //
  else if (method6(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  //
  else if (method7(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  //
  else if (method8(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  //
  else if (method9(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
```

```
  //
  else if (method10(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  //
  else if (method11(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  //
  else if (method12(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  //
  else if (method13(currentTime, currentStage, spotSet, agentSet, globalSharedVariables))
    return true;
  return false;
}
```

図 4-21 メインステージのルール実行に際して呼び出されるメソッド doIt のオーバーライド例

```
private boolean method1(TTime currentTime, String currentStage, HashMap<String, TSpot> spotSet,
  HashMap<String, TAgent> agentSet, HashMap<String, Object> globalSharedVariables) {
  TAgentRole ownerRole = (TAgentRole) getOwnerRole();
  // condition : keyword : is ADS=0
  if (ownerRole.getADS().equals("0")) {
    // command : substitution : askEquip AES=0
    ownerRole.setAES(0);
      return true;
  }
  return false;
}
```

図 4-22 メインステージのルール実行メソッド例

尚、エージェントのロールが無名ロールクラス(TAgentRole)ではない場合に無名ロールクラスの変数を使用する場合は次のように getMergedRole メソッドを使用してエージェントロール及びその変数へアクセスします。

```
private boolean method1(TTime currentTime, String currentStage,
  HashMap<String, TSpot> spotSet,  HashMap<String, TAgent> agentSet,
  HashMap<String, Object> globalSharedVariables) {
  TAgentRole ownerRole = (TAgentRole) getOwnerRole().getMergedRole("AgentRole");
  // condition : spot : <hospital>isSpot
  if (getAgent().getCurrentSpotName().equals(TSpotTypes.HOSPITAL)
    // && condition : keyword : is ADS=0i
    && ownerRole.getADS().equals("0i")) {
    // command : list : moveToFirst home
    getAgent().moveTo(((TSpot)(ownerRole.gethome().get(0))).getName(), spotSet);
    return true;
  }
  return false;
}
```

図 4-23 getMergedRole メソッドを使用したエージェントロール及びその変数へのアクセス

また、SOARS VisualShell4 では、Java プログラム実行時にエージェントが現在存在しているスポットを特定することは不可能なので、使用している変数から現在存在しているスポットを推測し、キャストを使用することによりスポットロール及びその変数へアクセスします。

```
//
// Getter and Setter
//

/**
 *
 * return
 */
private ArrayList<Object> getschool() {
  if (TStudentMoveRole.class.isInstance( getOwnerRole()))
    return ((TStudentMoveRole)getOwnerRole()).getschool();
  else if (TSchoolchildMoveRole.class.isInstance( getOwnerRole()))
    return ((TSchoolchildMoveRole)getOwnerRole()).getschool();
  else
    return null;
}

/**
 * @ param fschool
 */
private void setschool(ArrayList<Object> fschool) {
  if (TStudentMoveRole.class.isInstance( getOwnerRole()))
    ((TStudentMoveRole)getOwnerRole()).setschool(fschool);
  else if (TSchoolchildMoveRole.class.isInstance( getOwnerRole()))
    ((TSchoolchildMoveRole)getOwnerRole()).setschool(fschool);
}
```

図 4-24 エージェントが現在存在しているスポットのスポットロール及び変数へアクセスする為のメソッド例

```
private boolean method1(TTime currentTime, String currentStage,
  HashMap<String, TSpot> spotSet, HashMap<String, TAgent> agentSet,
  HashMap<String, Object> globalSharedVariables) {
  // command : list : moveToFirst school
  getAgent().moveTo(((TSpot)(getschool().get(0))).getName(), spotSet);
  return true;
}
```

図 4-25 エージェントが現在存在しているスポットのスポットロール及び変数へのアクセス例