CS223 Introduction to Software Engineering
Assignment - Part B 1992/3

Lee Suker
Computer Science 2nd Year
usercode: leesuker

CS223B Introduction to Software Engineering. Individual Preliminary Pequirernents Analysis Task

(1) Introduction

The aim of this document is to briefly discuss the essential considerations that were taken into account during the development of the preliminary requirements. The following section discusses the steps taken in each version of the software, and highlights any major considerations.

(2) Developing the Specification

A. Version 1.1

The first logical Agent to consider was the Umpire. If the Umpire only Agent in the game then his life cycle model is very small. only order play to start. Hence the introduction of the ball_live. The Umpire can only call the ball live once, so the actions set ball live to true.
was the He can variable Umpires

B. Version 1.2

Introducing the bowler seemed the next best step, since introducing the Batsman or fielders Agents at this stage would make the integration of other Agents more difficult, and time consuming at a later date. The bowler Agent can simply deliver the ball. The style of ball is delivered beyond the scope of this preliminary stage. The ball can only be delivered when the Umpire Agent has ordered play to start. What happens to the ball once the ball has left the bowler needs to studied. Once delivered the ball can approach the wicket or veer wide. The wicket can only be hit if the ball is travelling towards it and, has not already been hit. The ball will continue to travel if it misses the wicket or goes wide.

C. Version 1.3

Describing the ball Agent in more detail after missing the wicket or going wide, requires a notion of speed. This is essentially difficult in when using a rule based system to develop a specification. Hence boundaries were imagined. Each time the ball travels into a new boundary a monitor recognises the distance traveled. The fourth boundary essentially simulates the boundary of the cricket ground. Definition for a SIX and a FOUR to be scored were introduced but in this version these definition were not used. Generally the ball just travels.

D. Version 1.4

The next Agent to enter the simulation is clearly the batsmen. Since the Batsman alone will interfere with the other existing Agents. Providing the ball has not veered wide the batsmen will receive the ball and attempt to hit it. The Agent only gets one opportunity at doing this. Once he has hit or missed, the ball must continue to travel. In this simple version the simulation stops when the Batsman has interacted with the ball

E. Version 1.5

The bugs from the old version were ironed out so that the ball continued to move after the Batsman interfered with ball. The Batsman can now be bowled, if the Batsman hits/slices the ball or misses the ball, then provided that he ball has not passed the wicket or travelled wide the ball can hit the wicket. If the ball misses the wicket it will travel off towards the boundary. If the wicket is not hit then the ball will travel
i.e.the ball will travel if the batsmen was the last Agent to

trouble the ball and the ball has passed the wicket without hitting it, or the bowler delivered a wide.

F. Version 1.6

The job of the Wicket Keeper is to intercept the ball if it travels directly behind the wicket. Once the ball has in effect passed the wicket through going wide or missing the wicket the Wicket Keeper can intercept the ball. But he can only intercept it if the ball is within a certain range, i.e directly behind the wicket. If the Agent intercepts the ball then Umpire will decide if it is a catch or not. Hence more properties of the ball Agent need to be defined, resulting in definition of the ball being in the air, and on the ground.

G. Version 1.7

A notion of the direction of the ball is introduced. The cricket field was basically split into four region. The On side, the Off side, directly in front of the wicket and directly behind. The Batsman actions now include hitting a Cover Drive, which directs to the Off side of the field, a On Drive towards the On side , a straight drive directly in front of the wicket, or the ball just clips the bat or misses, to travel directly behind the wicket. Fielder are introduced to cover these four positions and attempt to intercept the ball when the ball comes there way.

H. Version 1.8

The final Agent is introduced at this level. That is the second Batsman. The main problem here is to handle the actions of the batsmen running. There needs to be some state that represents where the Batsmen are, i.e. what wicket they are at, and also whether they are in the middle of a run or not. Since a Batsman making a run is not an atomic event, there needs to be states that show he is in the middle of a run. Definitions for each Batsman were established. Where_is_Batsman could be assigned "North_wicket" or "South_wicket", and Batsman_in_crease was a Boolean flag. Hence the Batsman actions were to start and complete a run. A monitor also needed to be set up to monitor the number of runs completed Then when the ball is no longer in play the score is printed. A further definition was required here since if runs were scored due to a boundary this would override the runs that the batsmen made. At this stage the batsman could not be run out.

I. Version 1.9

An extension of the fielders action was added here. They interfere with the life cycle of the batsmen in that they can run them out if the throw the ball at the wicket whilst the batsmen are running. The fielders action were simplified, in that if they threw the ball towards the wicket, they hit it. It was considered too complicate to simulate what would happen to the state of the model, if the ball missed the wicket. This would be a possible extension that may be undertaken by the team. The model of running a Batsman out was further simplified in this version by arbitrarily choosing which Batsman was out , no matter which wicket was actually hit.

J. Version 1.10

A suitable place to end the simulation of naive cricket was decided. Final definitions were introduced to decide which of the actual Batsman was run out. Further additions were made, to tidy up the prototype. Examples

include deciding when the ball is actually dead, and how the simulation ends.

(3) Final Preliminary Analysis

```
1   TRUE   = 1; FALSE = 0;
2   ball_live = FALSE;
3   ball_delivered=FALSE;
4   ball_received=FALSE;
5   ball_wide=FALSE;

6   six_runs is ball_flies && ball_crosses_boundary;
7   four_runs is ball_rolls && ball_crosses_boundary;
8   ball_rolls =FALSE;
9   ball_flies is ball_travels && !ball_rolls;
10  ball_travels is (last_ball_agent =='batsmen" ::(last_ball~agent =="bowler"
11                  && ball_wide)) && '.ballreturned_to_bowler;

12  ball_intercepted = FALSE;
13  ball_caught is ball_intercepted && ball_hit && !ball_rolls;
14  ball_crosses_boundary is ball_travels && ball~dist >-- 4;
15  balihit is coverdrive     straigbtdrive    ondrive    edge
16  cover_drive    FALSE;
17  straight_drive    FALSE;
18  on_drive    FALSE;
19  edge    FALSE;
20  ball_missed    FALSE;
21  batsmanl_in_crease    TRUE;
22  batsaan2_in_crease   TRUE;
23  both_in_crease is batsmanl_in_crease && batsman2_in_crease;
24  batl_out is ball_caught     batsman_bowled    batl_runout;
25  bat2_out is bat2_run_out;
26  batl_run_out is !batsmanl_in_crease && ((where_was_batsuanl  "North_wicket"
27       && ballthrown_to_Southwicket)     (wberewas_batsmanl     "Southwicket"
28       && ball_thrown_to_Nortbwicket));
29  bat2_run_out is !bats.an2_in_crease && ((wherewas_batsman2  "North_wicket"
30       && ball_thrown_to_South_wicket)    (where_was_batsnan2   "South_wicket"
31       && ball_thrown_to_North_wicket));
32  batsman_bowled   FALSE;
33  runs_scored is (six_runs) ? 6
34                 (( four_runs) ? 4
35                      ((bal l~intercepted &&
36                    (!batl~out      !bat2~out) ) ? runs
37                             0));

38  last_ball_agent is   (ball~delivered && !ball~hit && !ball_missed) ? "bowler"
39                    (ball~hit     balimissed) ? "batsmen"
40                       "fielder");
```

```
1       where_was_batsmanl is (runs    1    runs ----3) ? "Northwicket"
2                       "South~wicket";
3       wherewas~batsman2 is (runs ---- 1  runs ----3) ? "Southwicket"
4                     : "North_wicket";
5       ball_thrown_to_North~wicket - FALSE;
6       ball_thrown_to_South_wicket - FALSE;
7       ball_returned_to_bowler - FALSE;
8       simulation_over -- FALSE;
9       signal_Bl  FALSE;
10      signal_B2  FALSE;

11 action0 is ["Umpire says play", 0, !balllivel;
12 actionl is ["Umpire signals a SIX ", 1, six~runs && !simulation~over~;
13 action2 is ["Umpire signals a FOUR", 2, four~runs && !siaulation~overJ;
14 action3 is ["Umpire signals Batsmanl Out", 3, batl~out && !signal~Bl];
15 action4 is ["Umpire signals Batsman2 out", 4, bat2~out && !signal~B21;
16 proc umpire : action (
17 switch (action) (
18 case      0: ball~live   TRUE;
19 break;
20 case1: siaulation~over   TRUE; break;
21 break;
22 case2: simulationover - TRUE;
23 break;
24 case3: signal~Bl   TRUE; break;
25 break;
26 case4: signal~B2   TRUE;
27 break;
28 default: return;
29    }
30}

31      actionS is ("bowler delivers ball",
32      proc bowler : action (
33          switch (action) (
34               case 5: ball_delivered  TRUE;
35                   break;
36               default: return;
37 I
38      I

39 act ion6 is ["ball travels to wicket", 6, ball~delivered && !ball~received1;
40 action7 is ["ball travels wide", 7, ball~delivered && !ball~received];
41 actionS is ("ball on ground      ", 8, ball~travels && !ball_intercepted
42           && !batsman_bowled && !ball~crossesboundary);

S, ball~live &&

43      action9 is ["ball in air", 9, ball_travels && !ball~intercepted
44               && !batsman_bowled && !ball~rolls && !ball_crosses_boundary];

45      actionlO is ["batsmen is bowled", 10, ball_received && !batsman~bowled
46               && !ball~wide && (edge   ball~missed) && ball~dist    0);
47      proc ball : action (     switch (action) (
```

```
1  case 6: ball_received     TRUE;
2  break;
3  case 7: ball~wide   TRUE;
4  ball_received    TRUE;
5              break;
6  case8: ball~rolls    - TRUE;
7              break;
8  case 9: ball_rolls   FALSE;
9  break;
10 case10: batsman~bowled  TRUE;
11 break;
12 default: return;
13 I
14 I


15      action11("batsaan hit cover drive", 11, !ball~wide
16                    && last~ball~agent     "bowler" && ball_received];
17      action12["batsman bit straight drive                , 12, !ball_wide
18                    && last~ball~agent     "bowler" && balireceived);
19      action13["batsman hit on drive", 13, '.ball~wide
20                    && last~ball~agent     "bowler" && balireceived];
21       action14("batsman got a edge", 14, !ball~wide
22                    && last~ball~agent     "bowler" && ball~received];
23       action15f"batsaan misses ball", 15, !ball_wide
24                    && last~ball~agent     "bowler" && ball_received);
25      action16 is ["batsaen begin running", 16, !batl~out && '.bat2~out
26                && !ball_crosses~boundary && both~in~crease && ball~travels];
27      action17 is ["batsuanl ends run", 17, !batl~out && !batsmanl_in_crease];
28      action18 is ["batsuan2 ends run", 18, !bat2out    && !batsaan2~in~crease];
29     proc batsmen :  action
30          switch (action) I
31               case  11: cover~drive   TRUE;
32                     break;
33               case  12: straight~drive    TRUE;
34                     break;
35               case  13: on_drive - TRUE;
36                     break;
37               case  14: edge  TRUE;
38                     break;
39               case  15: ball~missed   TRUE;
40                     break;
41               case  16: batsuanl~in_crease    FALSE;
42                     batsuan2~in~crease    FALSE;
43                     break;
44               case  17: batsmanl~in_crease    TRUE;
45                     break;
46               case  18: batsman2~in_crease - TRUE;
47                     break;
48               default: return;
49
50
51      action19 is ["Wicket Keeper Intercepts" ,19 , ball~travels && edge       &&
ball~dist                                                                        11;
52      action20 is ["Offside fielder Intercepts          ", 20, ball~travels
```

```
1       && !ballintercepted && ball_dist > 1 && ball~dist < 4 && cover_drive);
2       action21 is ["Onside fielder Intercepts", 21, ball~travels
3         && !ball~intercepted && ball_dist > 1 && ball_dist ( 4 &&
4       action22 is ["Bowler Intercepts Ball                    ", 22, ball_travels
5                 && ball~dist     1 && !ball_intercepted && straight~drive);
6       action23 is ["Fielder returns ball to bowler", 23, ball~intercepted
7                           && !ball_returned_to~bowler && both_in_crease];
8       action24 is ["Fielder throws to North Wicket     ", 24,
9            ((where_was_batsuanl     "Southwicket" && !batsmanl_in_crease)
10           (where~was~batsman2     "South_wicket" && '.bats.an2~in~crease))
11                              && !ball_thrown_to_North_wicket
12                                      && ball_intercepted];
13      action2S is ["Fielder throws to South Wicket", 25,
14           ((wherewasbatsmanl     "Northwicket" && 1.batsmanl_in_crease)
15           (where_was_batsaan2     "North_wicket" && !batsman2_in_crease))
16                              && !ball~thrown_to_South_wicket
17                                       && ball intercepted];
18      proc fielders : action (
19          switch (action) (
20              case 19: ball intercepted   TRUE;
21                    break;
22              case 20: ball~intercepted  TRUE;
23                    break;
24              case 21: ball_intercepted  TRUE;
25                    break;
26              case 22: ball~intercepted  TRUE;
27                   break;
28              case 23: ball~returned~to~bowler   TRUE;
29                    break;
30              case 24: ball~thrown_to_North_wicket   TRUE;
31                   break;
32              case 2S: ball~thrown~to~South~wicket   TRUE;
33                    break;
34             default: return;
35}
36}
37 proc  init~ball~dist : ball~live (
38 ball_dist - 0;
39     }

40  proc inc~ball_dist : ball_rolls (
41  if (ball_travels---- TRUE) ball_dist++;

42     }


43  procmonitor_distance : ball_dist (
44  if (ball~dist >0) (
45  write("ball~dist   "); writeln(ball~dist);
46     }
47}
48      proc init~batsmen_run : ball~live (   runs   0;
49 }
```

```
1       proc count_runs : both~in~crease
2           if (both_in_crease ---- TRUE && ball~travels ---- TRUE) runs++;
3 }

4 proc   monitor_scores    runs~scored {
5 if (runs~scored !--0) (
6 write(" Runs Scored    ") ; writeln(runs~scored)
7           )
8)

9 proc monitor~batsmen          runs (
10 write("Batmansl is at the "); writeln(wherewas_batsmanl);
11 write("Batmans2 is at the "); writeln(wherewasbatsman2);
12)

13 proc print    enabled {
14 auto i;
15 writelno; for (il; i<--enabled#; i++) writeln(enabled[i]); writelno; I
16  )

17 proc monitor~ball~agent  last~ball~agent (
18 write("last ball agent   "); writeln(last~ball~agent);)

19 proc monitor_actions    action (
20 writeln("\n\t" I, record(action+1) II "\n");
21          )

22 func record (
23 para actnum;
24 return actions [actnum] [1);
25

26 actions is (actionO, actioni, action2, action3, action4, actionS, action6,
27 action7, action8, action9, actionlo, actionil, actionl2,
28 actionl3, actionl4, actioniS, actionl6, actionl7, actionl8,
29 actionl9, action20, action21, action22, action23, action24,
30 action2S];

31      enabled is (umpire~enabled, bowler_enabled, ball~enabled, batsmen_enabled
32                 ,fielders~enabled)

33      umpire~enabled is (actiono, actioni, action2, action3, action4];
34      bowler~enabled is (actionS];
35      ball~enabled is (action6, action7, action8, action9, actionlOl;
36      batsmen~enabled is (actionll, actionl2, actionl3, actionl4, actionlS, actionl6,
37                    action17, actionl81;
38      fielders_enabled is (actionl9, action20, action21, action22, action23,
39                    action24. action2S];

40      proc getready
41      (    autocalc  0;
42           ball~live FALSE;
43           ball~delivered   FALSE;
44           ball~received  FALSE;
```

```
ball~wide -- FALSE;
          batsman_bowled - FALSE;
          ball_rolls    FALSE;
          cover~drive    FALSE;
          straight~drive    FALSE;
          on~drive    FALSE;
          edge    FALSE;
          ball~missed   FALSE;
          ball_intercepted    FALSE;
          batsmanl~in~crease    TRUE;
          batsman2~in~crease   TRUE;
          ball_thrown~toSouth~wicket   FALSE;
          ball_thrown~to~North_wicket  FALSE;
          ball~returned~to~bowler    FALSE;
          simulation~over    FALSE;
          signal_El    FALSE;
          signal_B2    FALSE;
          autocalc    1;

```

(4)Conclusion and Further Refinements

The life cycle of the model can be further extended by focusing more on the types of ball that the bowler would deliver, and the type of strokes a bowler would be play. How these would interact with the fielding positions, a more in depth analysis could be made about what happens to the ball once the fielder has thrown the ball at the wicket and missed.

Through experience developing this model it has been noticed that when extending the actions of Agents, you have to be careful how they interact with other Agents, and it is often necessary to change when and how other Agents operate.

C5223 Introduction to Software Engineering Part B 1992/3

All the work contained or documented in this report is my own work. Any help I have received is acknowledged below.
Signed: -

ACKNOWLEDGEMENTS

STATEMENT OF QUALITY FOR INDIVIDUAL EDEN SPECIFICATION OF CRICKET SIMULATIONS

PROTOTYPE

Program design is prototyped using EDEN ~s No

Prototype was developed using SCCS <No

No

EDEN prototype interprets without errors

REQUIREMENTS

Are agent orientated design techniques used to Q>~e~ No develope prototype.

Were any resources, other than those stated Yes used in the design of prototype.

Other resources are indicated on reverse of this sheet.

Level of Design goes beyond that of urchin cricket

Prototype include the following features.

Minimum of 3 fielders

A wicket keeper

Two Batsmen

A bowler

A method of wicket taking

Defects

The product has no known defects.

Known defects are documented on the reverse of this sheet.

Yes No

LeThNo

L/Ye;~'No

GLS$)NO

(Le~\No

GYes

No

GYes~ No

No

Yes No