Diary Data Capture System for Time Use Built in Blaise and Maniplus

Fred Wensing

The Time Use Survey

The Time Use Survey will be conducted by Statistics New Zealand, commencing later in 2009.

For every household selected in the Time Use Survey, up to two adults are selected to participate in the survey. These persons are required to complete a hand-written diary of all the activities they do within a 48-hour period. Different days are allocated to different persons to ensure that all days of the week are covered.

As well as a primary activity, participants may record up to three additional activities which they may be performing at the same time. The diary also has provision to record where the activity is taking place, for whom and with whom the activity is done.

When the completed diary is collected, a supplementary Blaise questionnaire is administered by an interviewer to collect other socio-demographic information which will later be used in the analysis.

The paper diaries are returned to the office for data capture.

Sample questionnaire (left-hand page)

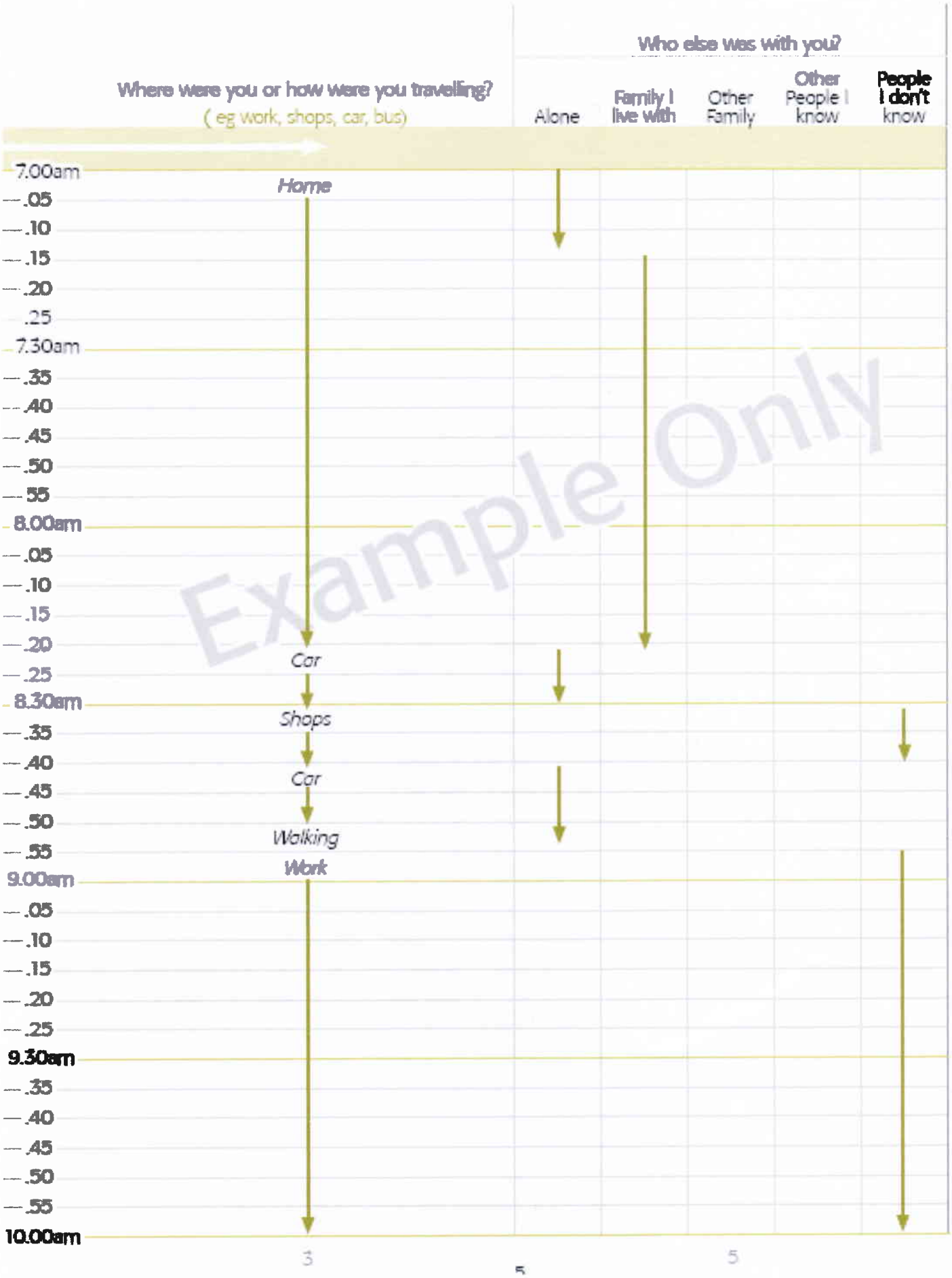
Office Use Only

HHLD Member		NonHHLD Member	
0-13	14+	0-13	14+

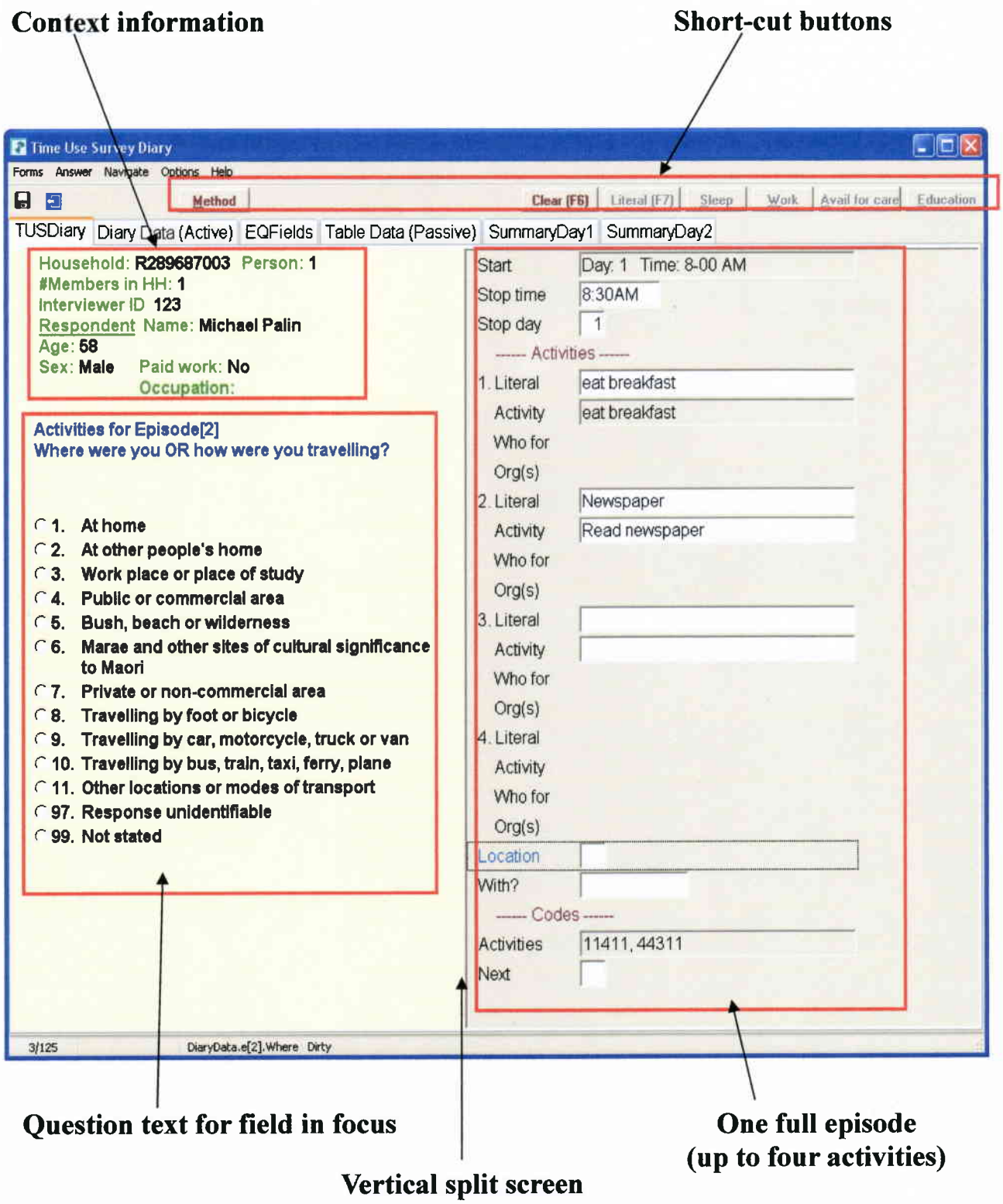
Day 1: 7am - 10am

	What were you doing?	What else were doing at the same time?
7.00am	Slept	
.05		
.10		
.15	Showered dressed	
.20		
.25	Got ready for work	Talked to partner
7.30am		
.35	Made childrens lunches	
.40		
.45	Made breakfast	
.50	Ate breakfast	
55		
8.00am	Internet banking	Checked email
.05		
.10	Tidied house	Talked to children
.15		
.20	Drove to shops	Listened to music
.25		
8.30am	Bought groceries for lunch	
.35		
.40	Drove to work	Listened to music
.45		
.50	Walked to work from carpark	
.55	Work	
9.00am		
.05		
.10		
.15		
.20	Work break morning tea	Drank coffee
.25	Work	
9.30am		
.35		
.40		
.45		
.50		
.55		
10.00am		
	4	2

Sample questionnaire (right-hand page)



Blaise data capture instrument
Form approach to capture



The Form approach shows the whole episode and all its elements

Blaise data capture instrument
Table approach to capture

Context information

Short-cut buttons

Question text for field in focus

Time Use Survey Diary

Forms Answer Navigate Options Help

Method Insert Delete Split Merge Clear (F6) Literal (F7) Sleep Work Avail for care Education

TUSDiary Diary Data (Passive) EQFields Table Data (Active) SummaryDay1 SummaryDay2

Household: R289687003 Person: 1 #Members in HH: 1
Interviewer ID 123
Respondent Name: Michael Palin
Age: 58
Sex: Male Paid work: No
Occupation:

Activities for Episode[1]
On what day did the episode start?

	Day	Time	Stop time	Stop day	1. Literal	Activity	Code 1	Who for	Org(s)	2. Literal
[1]	1	4:00AM	8:00AM	1	Sleep	Sleep	11211			
[2]	1	8:00AM	8:30AM	1	eat breakfast	eat breakfast	11411			Newspaper
[3]	1	8:30AM								
[4]										
[5]										
[6]										
[7]										
[8]										
[9]										
[10]										
[11]										

1/1 DiaryTable.e[1].Start_Day Dirty

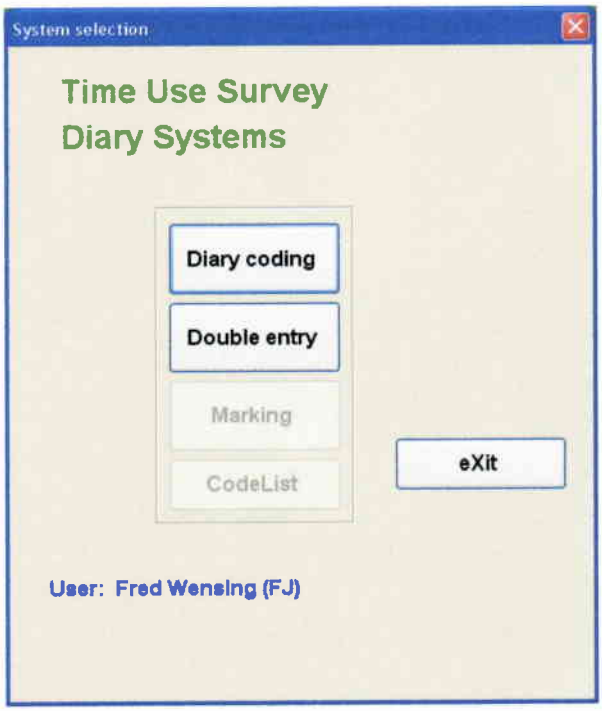
Horizontal split screen

Rows of episodes (up to four activities)

The Table approach shows multiple entries and supports insert, split, merge and deletion of entries

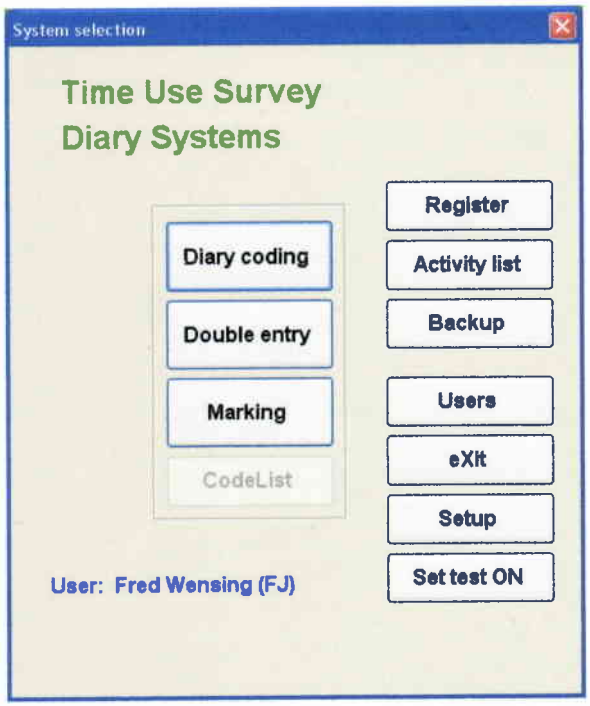
Entry interface

The main entry interface consists of a small dialog box containing buttons which are visible and/or activated based on the role of the known user.

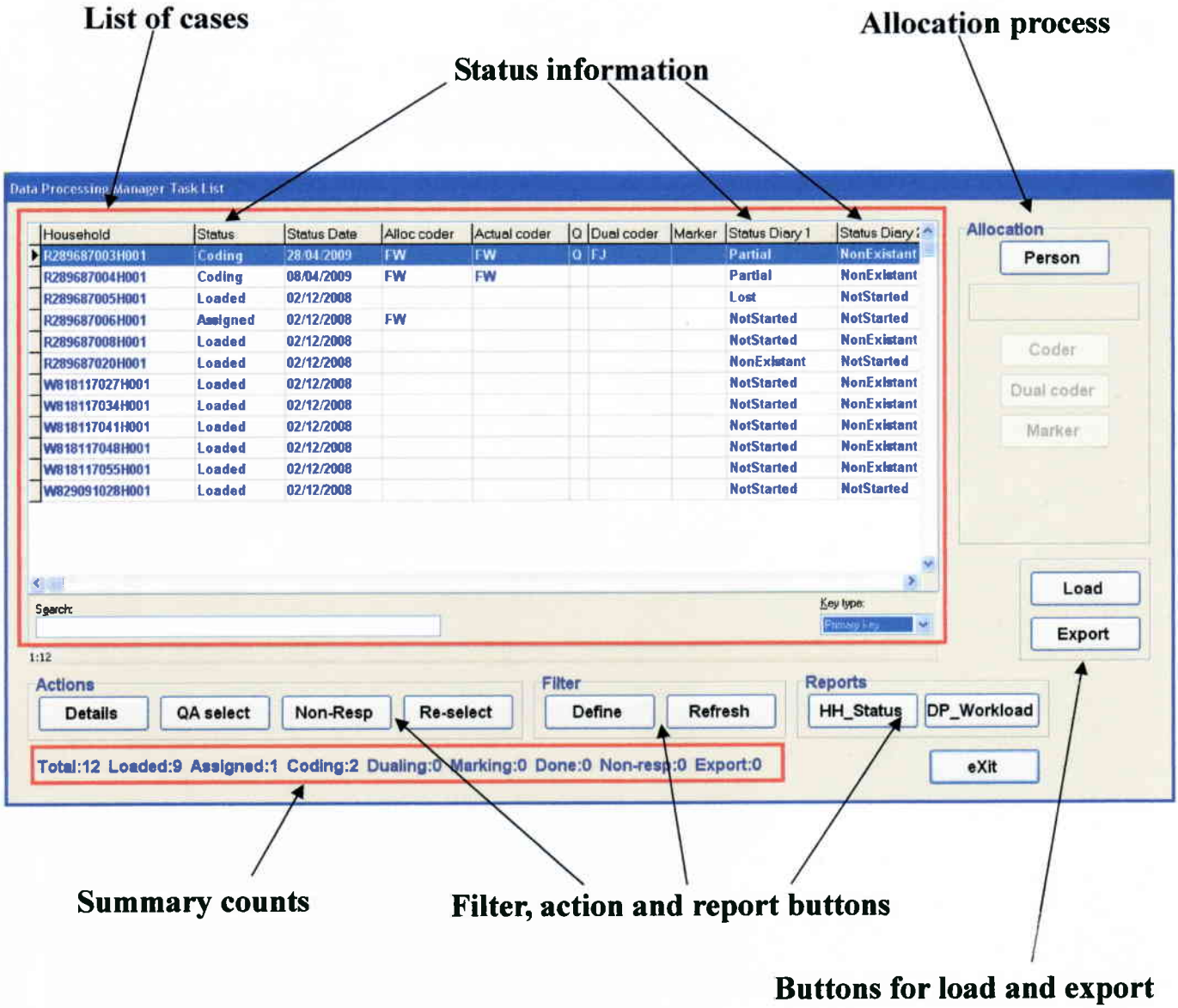


The main interface as seen by a data processor

The main interface as seen by a manager of the system

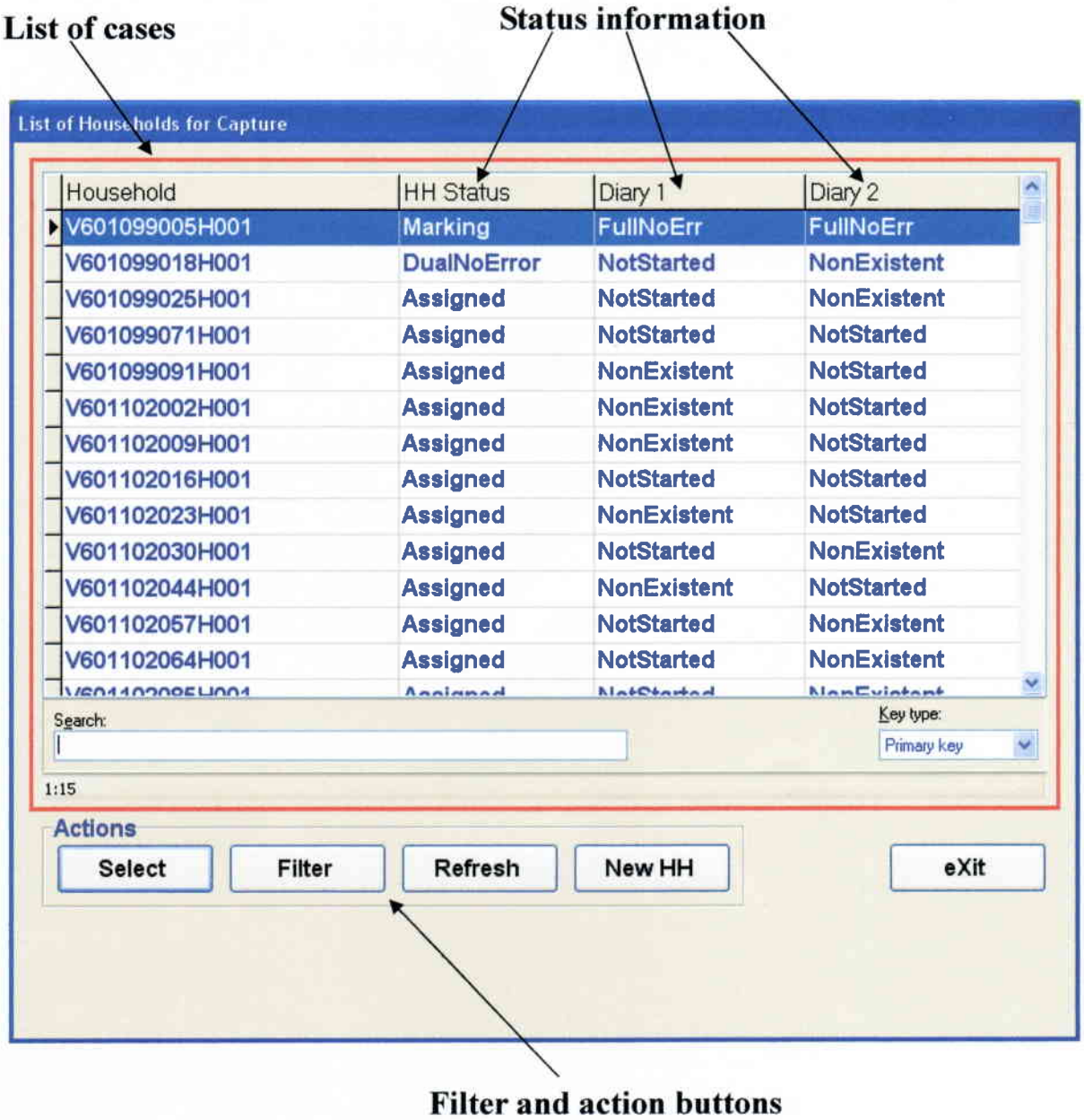


Data Processing Manager's Task list (Register)



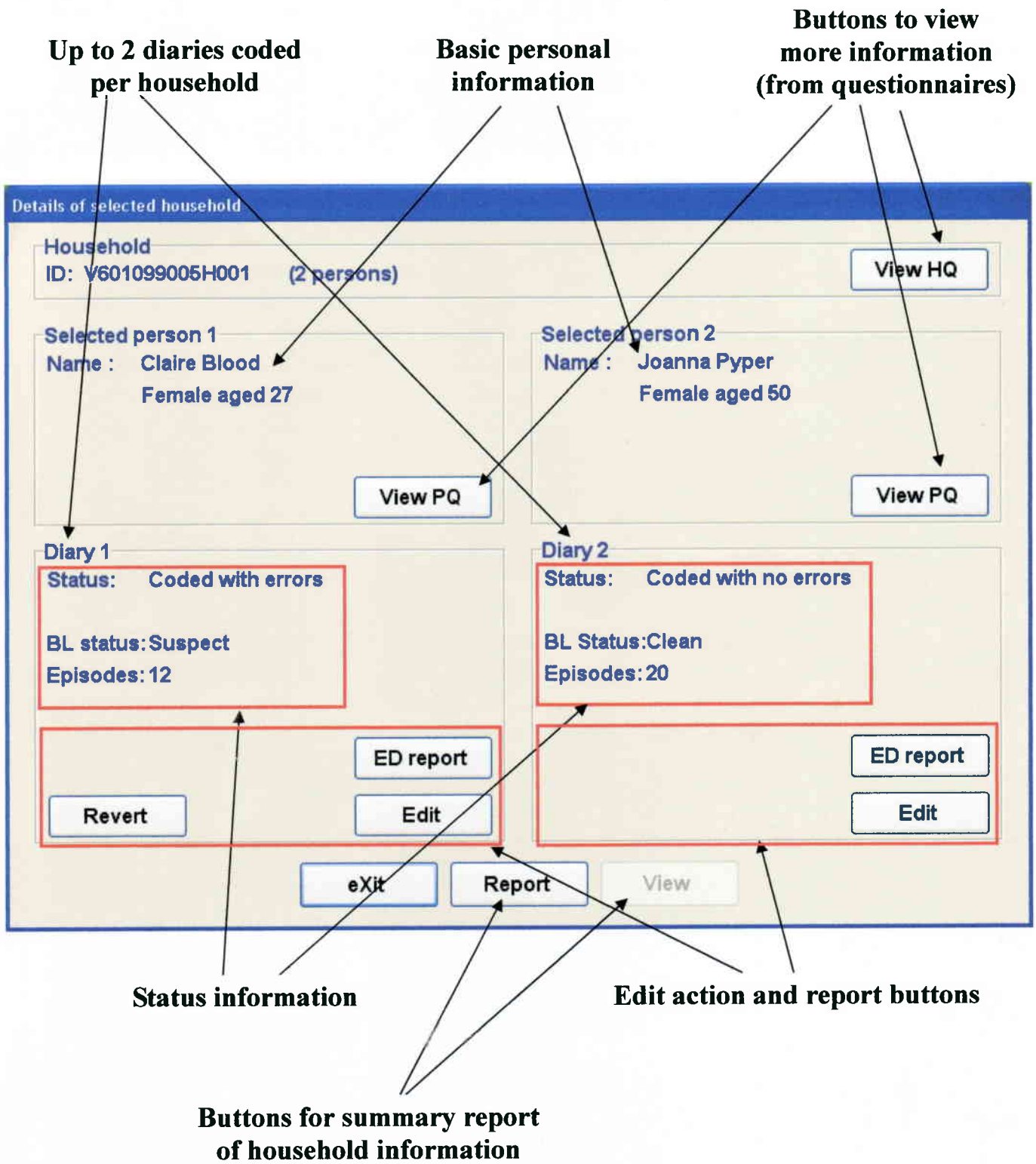
The Data Processing Manager's task list uses a central Register to keep track of diary coding progress. The interface can be used to load and export cases, assign cases to coders and produce reports

Data Processor's task list



The Data Processor's task list displays the cases that have been allocated to the coder. Status information is displayed and buttons are provided to initiate actions

Data Capture management interface



The Data Capture interface provides access to viewing, editing and reporting of progress with coding to the operator. Status information is displayed and buttons are provided to initiate actions and reports.

Data Capture reports

Summary household information to assist the operator with the family context of the respondents.

Report_R289687006H001.txt - Notepad

File Edit Format View Help

```
                    Time Use Survey - Summary of Household record : R289687006H001
28-04-2009                                                                                Page 1

Interviewer Number :          Number of Personal Questionnaries: 2          Number of Diaries: 2

| NAME              |SEX|AGE| RELATIONSHIP TO | RELATIONSHIP TO | ETHNICITY                |PERSONAL INCOME |
|                   |   |   | Fred            | Ted             |                          |                |
| Fred Jones        | M | 28|                 | other relative  |NZ                        |$50,001-$70,000
| Ted Jones         | M | 18| other relative  |                 |NZ                        |$100,001-$150,00
| John  Jones       | M | 28| other relative  | other relative  |NZ                        |$50,001-$70,000

RESPONDENT 1  FRED JONES                          DIARY DAYS: Friday to Sunday -- 10-10-2008 to 12-10-2008

INCOME TYPE:              self employment or business
OCCUPATION:
Paid work (Last 7 days):  No

UNPAID WORK NON HHLD MEM - 4 WEEKS:                        NAME OF ORGANISATIONS

RESPONDENT 2  TED JONES                           DIARY DAYS: Friday to Sunday -- 10-10-2008 to 12-10-2008

INCOME TYPE:              wages, salary or commission
OCCUPATION:               statistical analyst
Paid work (Last 7 days):  Yes

UNPAID WORK NON HHLD MEM - 4 WEEKS:                        NAME OF ORGANISATIONS

Care for child (under13)
Coach, teach or train                      Wellington Chess ass  Wellington Harriers
Provide transport                          Wellington Harriers
Shop
Gardening or repairs                       Wellington Bridge Cl
Fundraising or selling                     Tramping club
```

Outcome report of failure of edits applied to the diary entries.

Report_R289687006P001_Edits.txt - WordPad

File Edit View Insert Format Help

```
Report on edits for Diary : R289687006P001
-------------------------------------------------------------------------------

Edits with a status of : 3. Failed
#26    No eating or drinking activities recorded in 48 hours (Day 2).
#29    It is unlikely the respondent did not sleep in the last 24 hours (Day 2).
```

For Help, press F1 NUM

Dual coding

In order to maintain a good level of quality in the coding of activities, a proportion of households are selected for dual coding by another member of the coding team.

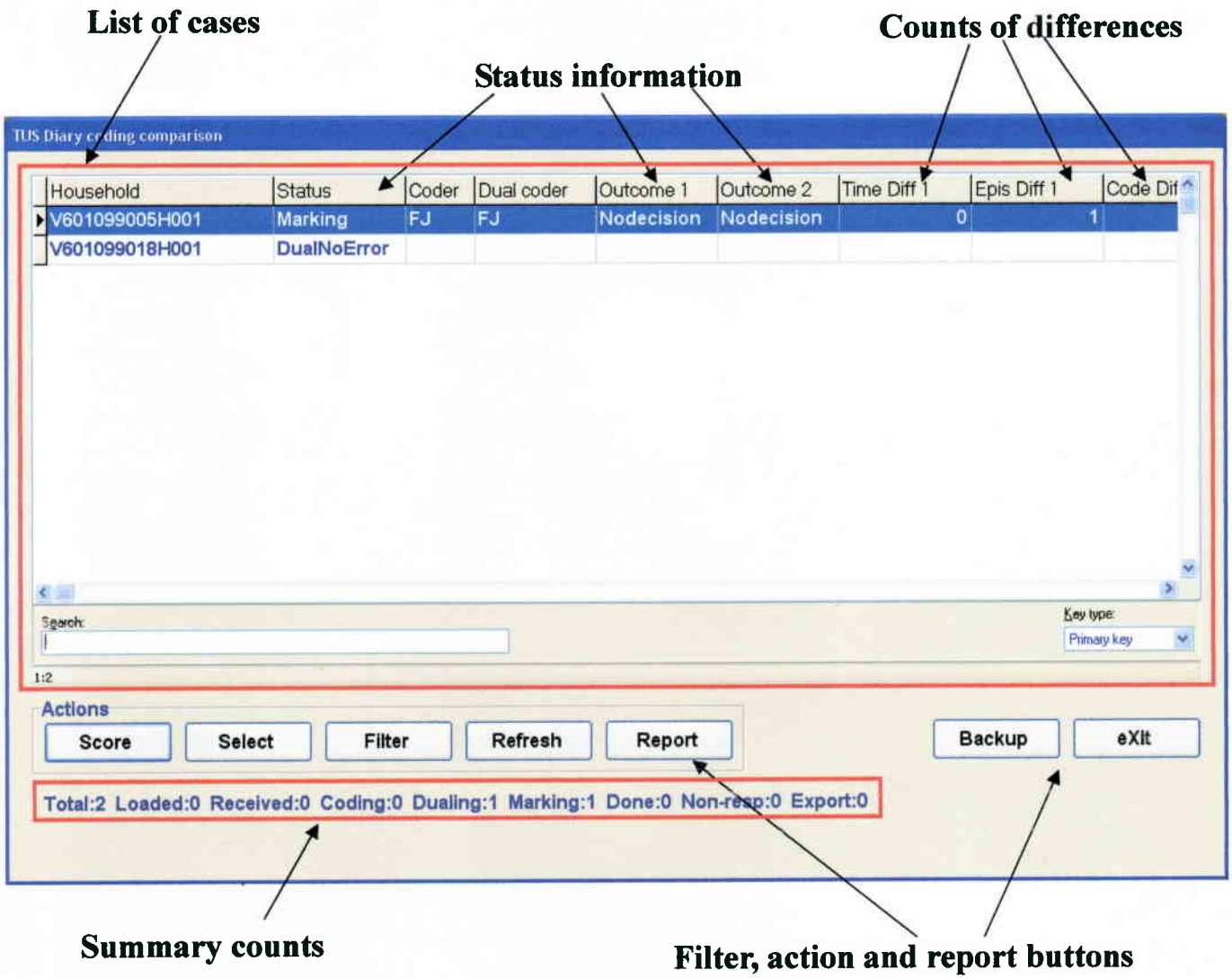
Once the diaries in a household have been coded and cleaned then the system uses a randomly generated number to decide whether that household is selected for dual coding or not. The proportion of households which are selected is based on a dual coding rate recorded against each user in the system.

The dual coding rate can be adjusted by the Data Processing Manager in which case there is a process to reselect or deselect households that are to be dual coded.

The dual coding interface is identical in functionality to the main coding interface.

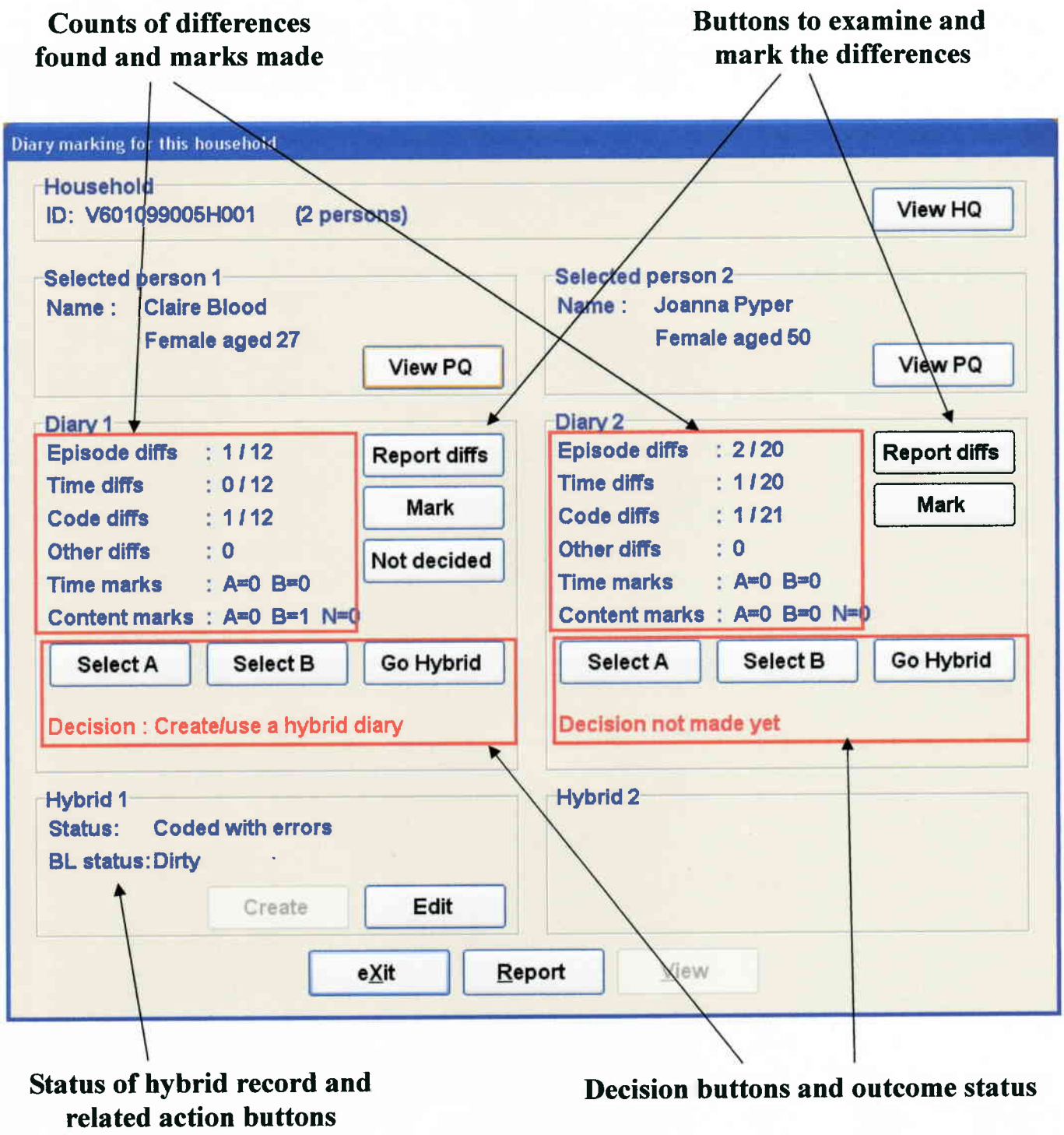
Once both coding and dual coding of a household has been completed then that is passed to a Quality Manager for identification of differences, scoring the differences and marking them.

Quality Manager's task list



The Quality Manager's task list shows cases that have been coded and dual coded and are ready for scoring and marking. Status information is displayed and buttons are provided to initiate actions and reports

Diary Marking management interface



The Diary Marking management interface shows the counts of differences and provides buttons to carry out marking and make decisions. Hybrid creation is also supported with buttons provided to initiate actions and reports.

Marking screen (default situation)

------ Time marking ------

Times
- (•) 0. Not marked yet
- ( ) 1. Primary coder
- ( ) 2. Dual coder

------ Content marking ------

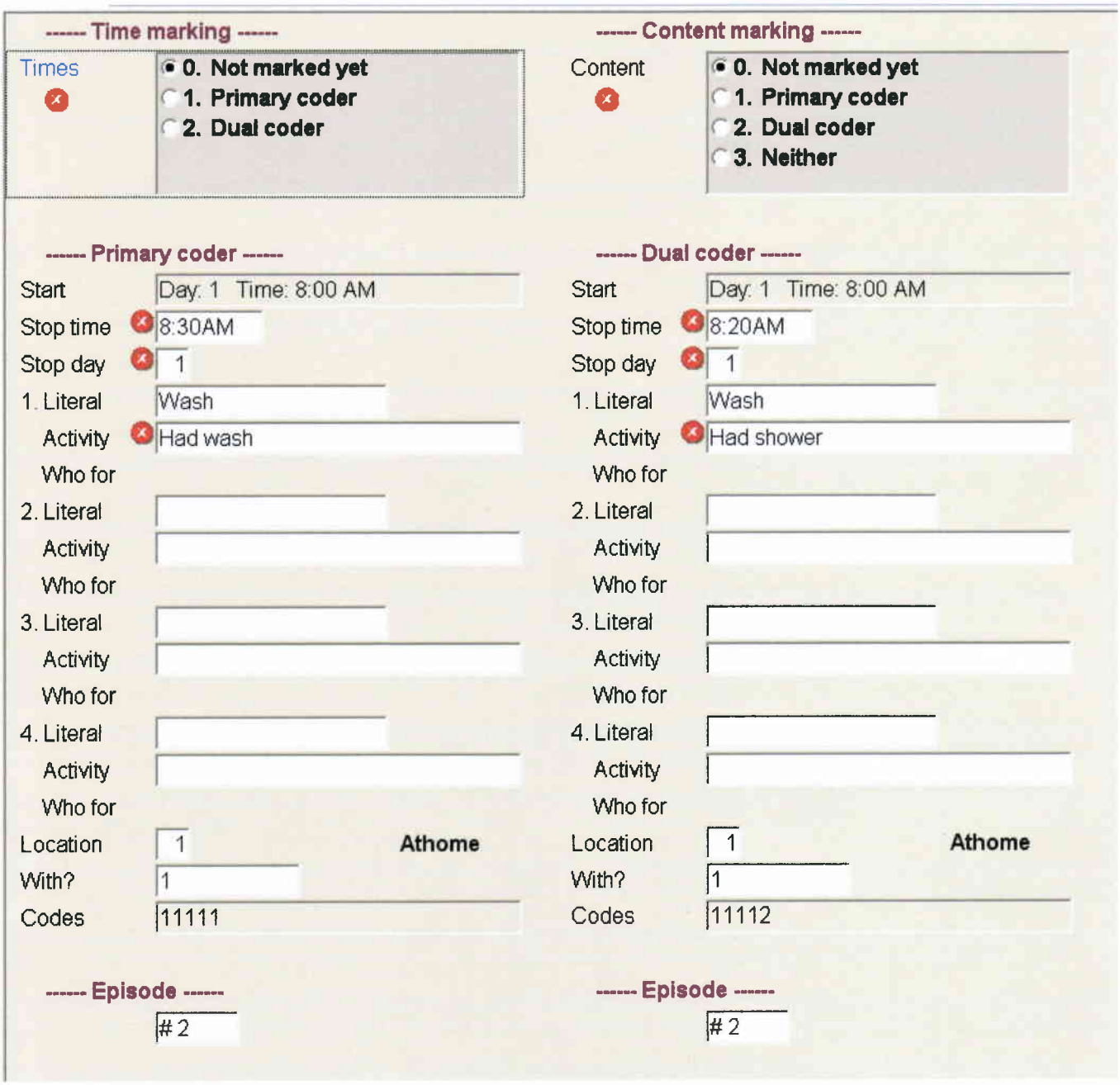
Content
- (•) 0. Not marked yet
- ( ) 1. Primary coder
- ( ) 2. Dual coder
- ( ) 3. Neither

------ Primary coder ------		------ Dual coder ------	
Start	Day: 0 Time: 4:00 AM	Start	Day: 0 Time: 4:00 AM
Stop time	8:00AM	Stop time	8:00AM
Stop day	1	Stop day	1
1. Literal	Sleep	1. Literal	Sleep
Activity	Sleep	Activity	Sleep
Who for		Who for	
2. Literal		2. Literal	
Activity		Activity	
Who for		Who for	
3. Literal		3. Literal	
Activity		Activity	
Who for		Who for	
4. Literal		4. Literal	
Activity		Activity	
Who for		Who for	
Location	1 **Athome**	Location	1 **Athome**
With?	1	With?	1
Codes	11211	Codes	11211

------ Episode ------	------ Episode ------
# 1	# 1

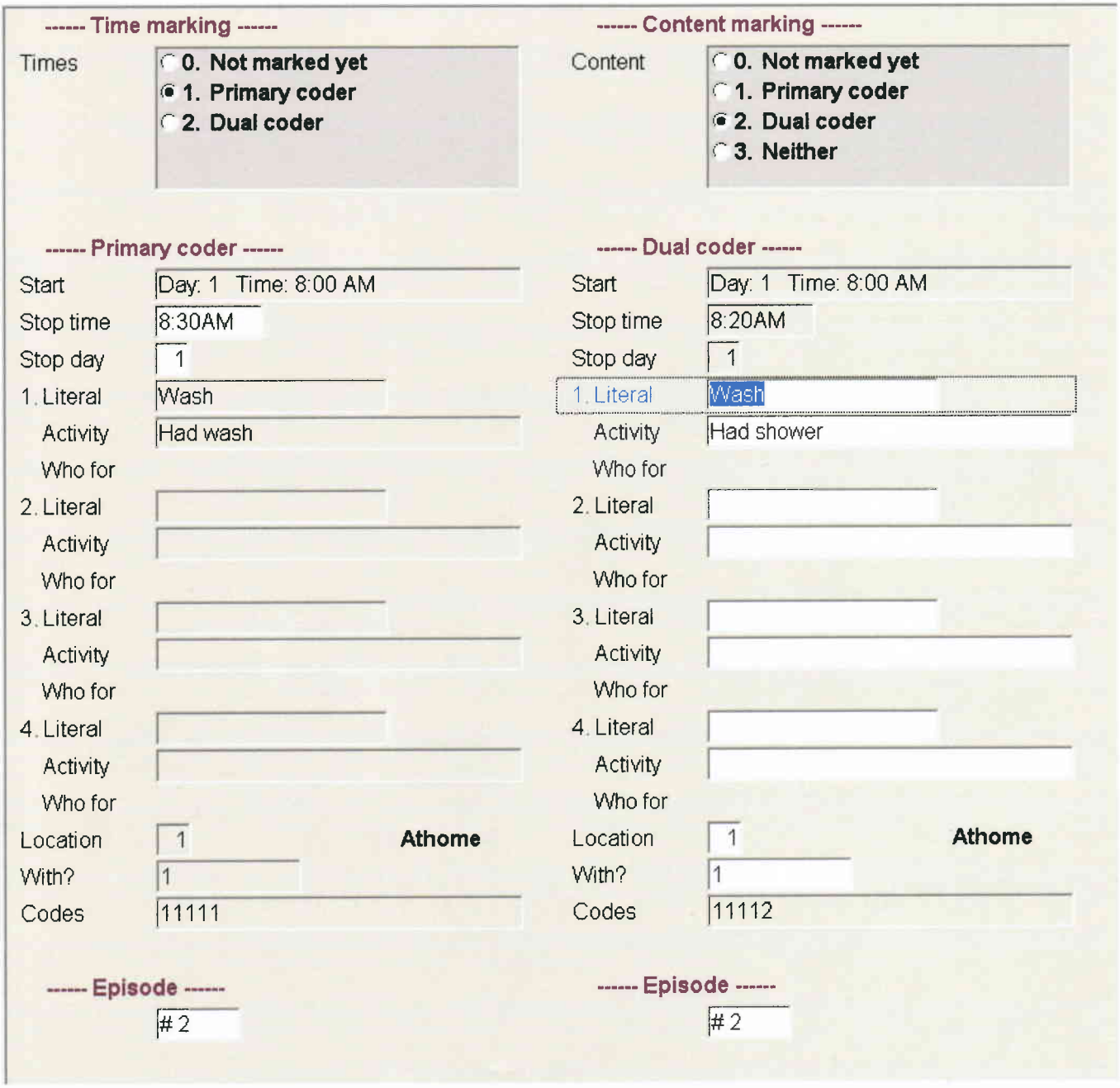
The Marking screens show matching entries for the primary and dual coder. Time and content marking occurs in the top section. The detail of entries cannot be changed (system is self-correcting). In the above screen shot the entries are matching therefore no marking is needed. The operator may proceed to the next episode or use the "Navigate – show errors" menu entry

Marking screen with differences identified



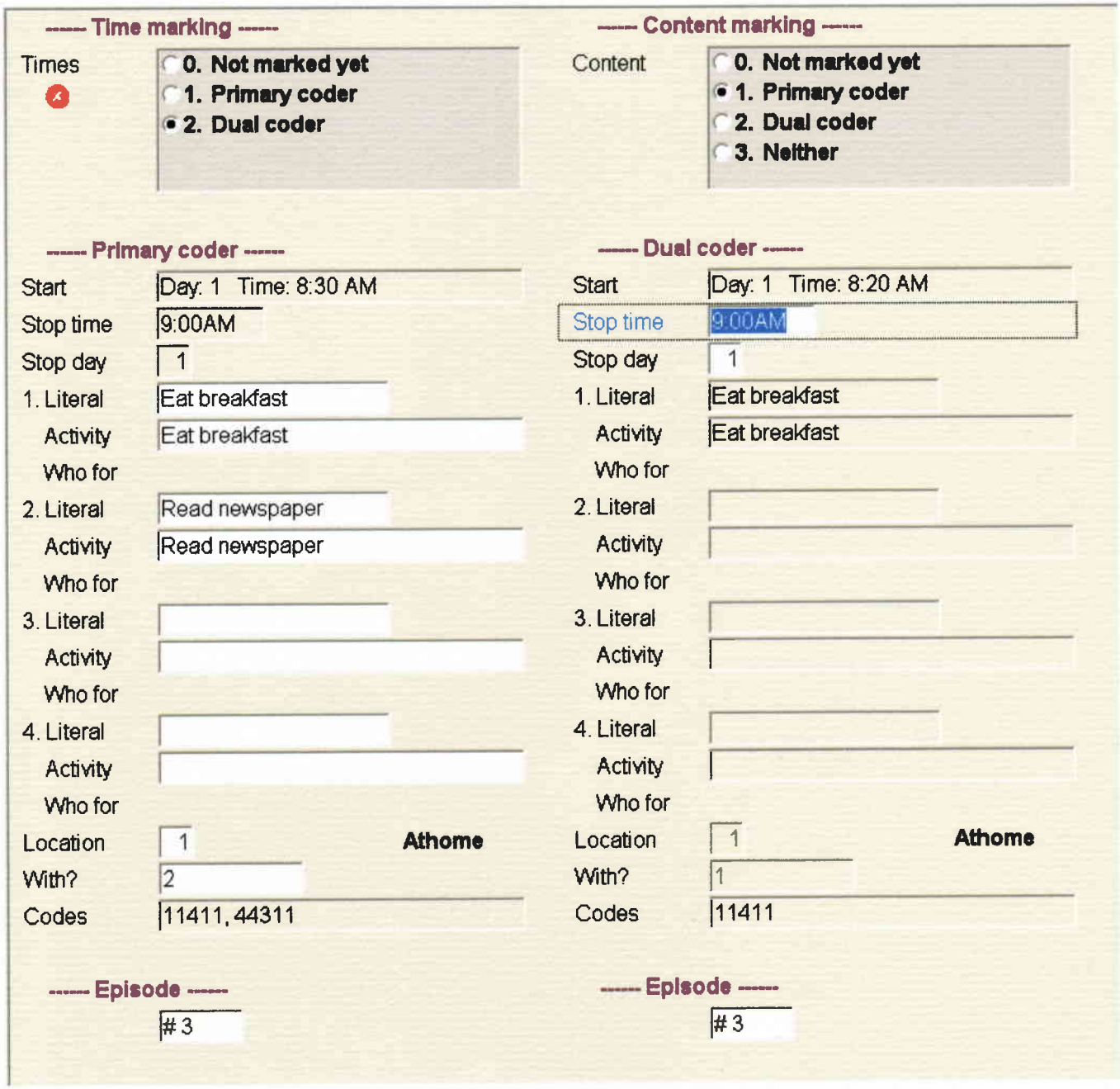
The Marking screen shows matching entries for the primary and dual coder. Where differences have been detected these are identified using red crosses. In the above screen shot there are discrepancies between the times and the content (see the Codes field). The marking fields at the top also show red crosses indicating that marking is required.

Marking screen with marking identified



The Marking screen also shows the outcome when marking has been done by "greying out" the non-selected entries. The screen shot above shows that the primary coder was selected for Times and the dual coder for Content. In this case the screen shows (in the white activated cells) which entries will be used if a hybrid is to be produced. The other entries are shown "greyed out" .
Notice that the red crosses have disappeared now that marking decisions have been made.

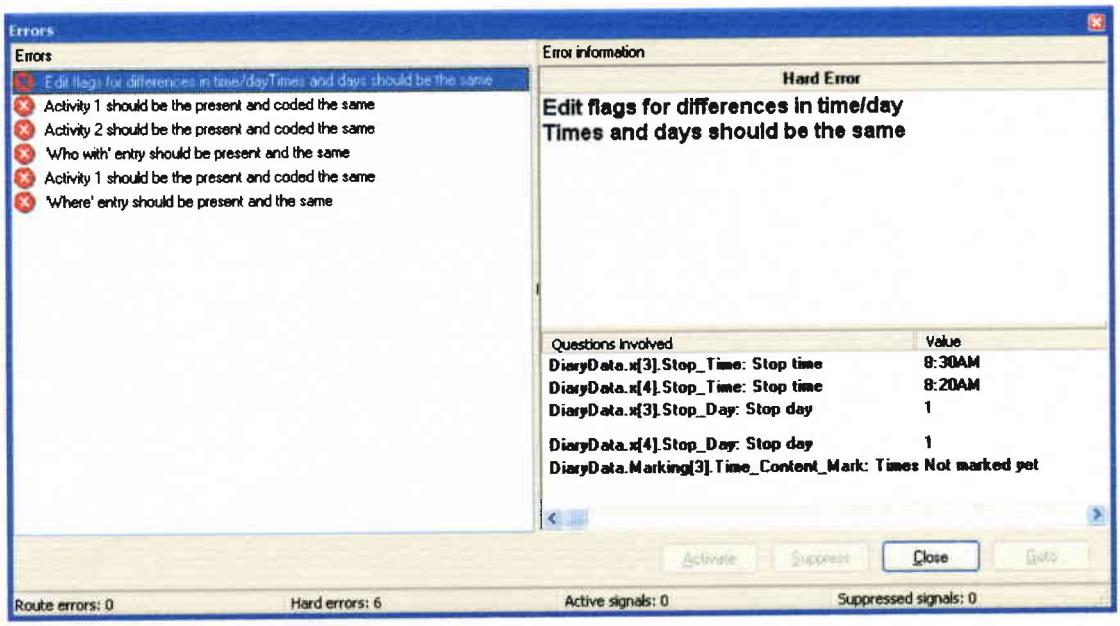
Marking screen with marking errors



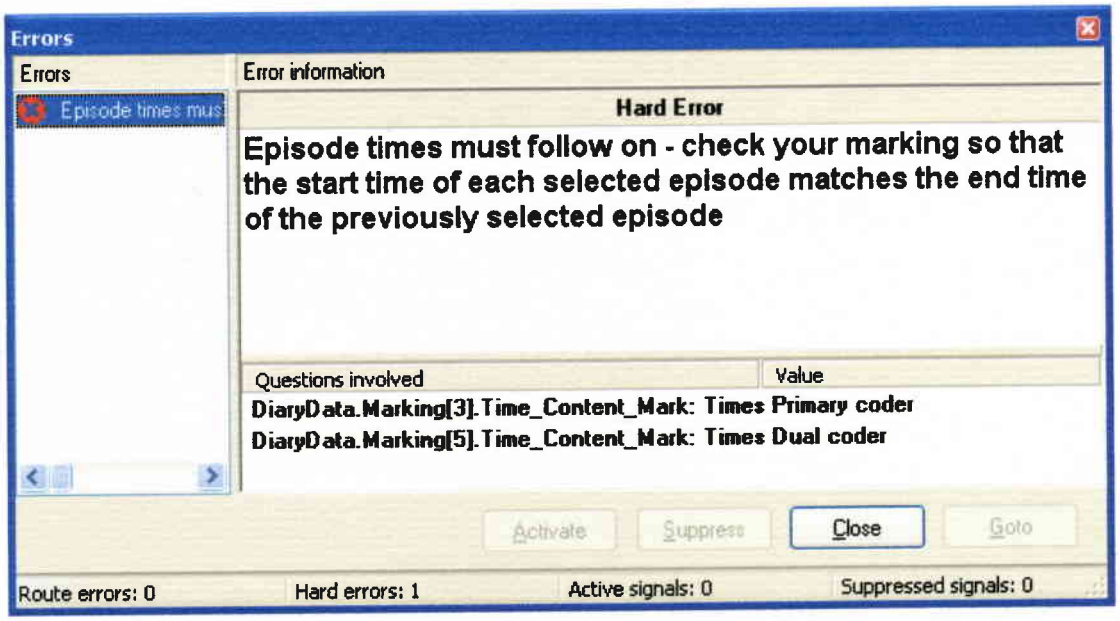
The Marking screen also shows when marking selections may produce errors in the continuity of times. The above screen shot shows a red cross against the time marking entry to indicate a problem with this marking selection.

Navigation and error messages

The operator can locate the differences and problems using the "Navigate / Show all errors" menu item. The screen shot below shows the list of problems and provides buttons to "go to" the entries involved.



To see the text of any indicated problem the operator double clicks on the red cross. The screen shot below shows the message produced by the system when the times are not continuous.



Reporting the differences and marking

Report of the counts of differences, the marking and outcome for one or more coders.

TUS_Score_Mark_List_2009_05_14.TXT - WordPad

File Edit View Insert Format Help

```
Time Use Survey Scoring and Marking Report : 14-05-2009
Discrepancy rates by selected coders

Diary                        ---- Counts --- ------ Differences ----------- -- Time Marks -- --- Content Marks -----
ID              Coder Dual   Epis   Codes   Times   Codes   Epis    Other  CoderA  CoderB  CoderA  CoderB  Neither  Outcome
-------------------------------------------------------------------------------------------------------------------------
V601099005P0FX    FJ           0       0       2       1       1       0       0       0       0       0       0    No decision
V601099005P0FX    FJ           0       0       1       0       1       0       0       0       0       0       0    Hybrid diary
-------------------------------------------------------------------------------------------------------------------------
                               0       0       3       1       2       0       0       0       0       0       0

                        Summary for Coder : FX
                        ----------------------------------
                        Total episodes coded    :       0
                        Total episodes accepted :       0
                        Total episodes rejected :       0
                        Rejection rate          :     N/A
                        ----------------------------------
```

For Help, press F1 NUM

Report showing the detail of differences found in one episode. Differences are marked with an asterisk (*)

Report_Diffs_V601099005P002.txt - WordPad

File Edit View Insert Format Help

```
                          Time Use Survey - Summary of Diary differences for V601099005P002
11-05-2009
Person #2 Female aged 27
-------------------------------------------------------------------------------
Summary
  Coder A (FX) : Episodes=12 Codes=12
  Coder B (FX) : Episodes=12 Codes=13
  Time differences     : 0
  Code differences     : 1
  Episode differences  : 1
  Other differences    : 0
-------------------------------------------------------------------------------
Detail for episode #13
  Start time  : 7:00 AM                          7:00 AM
  Start day   : 2                                2
  Stop time   : 8:00 AM                          8:00 AM
  Stop day    : 2                                2
              :
  Literal 1   : READ NOVEL                       READ NOVEL
  Activity 1  : READ NOVEL                       READ NOVEL
  Code 1      : 99666                            99666
  Who for 1   :
              :
  Literal 2   :                                  Drink tea
* Activity 2  :                                  Drink tea
* Code 2      :                                  99666
  Who for 2   :
              :
  Literal 3   :
  Activity 3  :
  Code 3      :
  Who for 3   :
              :
  Literal 4   :
  Activity 4  :
  Code 4      :
  Who for 4   :
              :
  Location    : 1                                1
  Who with    : 1                                1
-------------------------------------------------------------------------------
```

For Help, press F1 NUM

Technical challenge: Local deployment

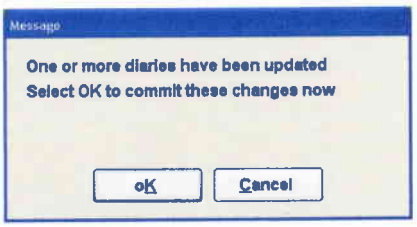
The most efficient way to operate Blaise data capture, when there is a complex instrument and a fair amount of external look-up steps, is to place the Blaise data capture file and the external look-up files local to the operator.

The two issues that need to be handled as a result are:

- Ensuring that the activity code list is always up-to-date
- Transferring the diary data to and from the central data store

The consistency of the activity code list is managed by recording the version number of the current activity code list in the set-up settings for the system. The version number of the activity code list in the local environment is then checked whenever the system is started. If the version is not up-to-date then the main data entry access buttons are de-activated until the latest version is installed.

The Diary data capture system has a central data store for each process (diary coding, double entry and marking) which is placed on the network. Once a household has been selected then the system copies any existing data to the local disk drive before enabling the operator to change it. At the end of involvement with that household the system prompts the operator to commit the changes or go back and revert to the original version:

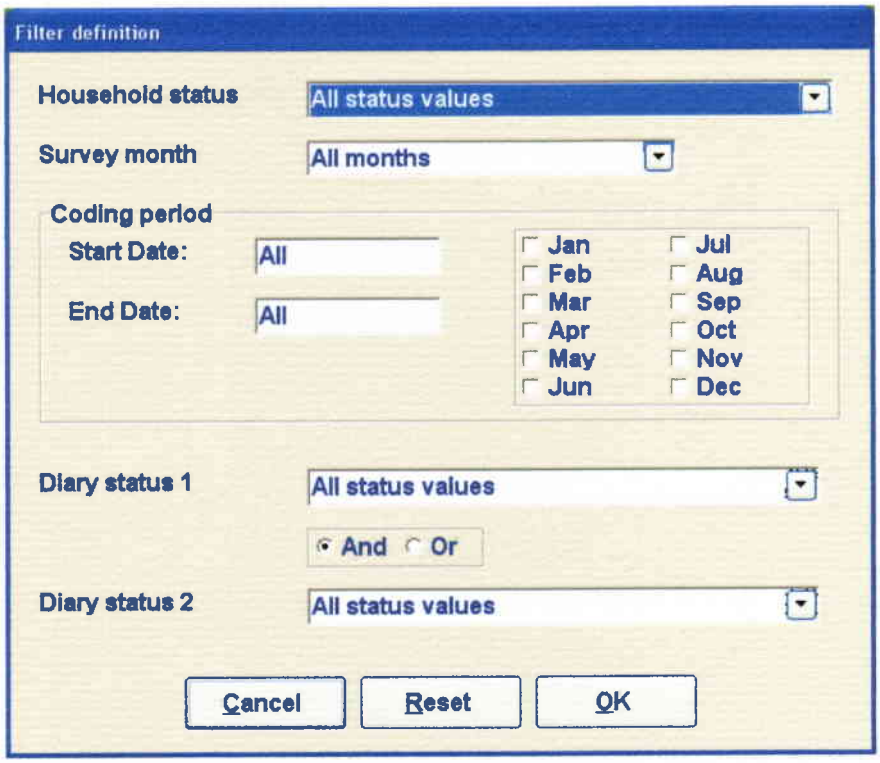


Technical challenge: Filtered lists

Once the survey gets underway, the list of households is expected to become large so a filter technique has been added to enable the system to target households that have been assigned to a particular user, have reached a particular status or from a particular month.

The filter works by applying a series of conditions to the list of entries in the main register. Those entries which match the filter conditions are then copied to a temporary file before displaying them on the screen.

Using a filter frees up the register for multiple shared access. The register data only gets updated with status information when the operator closes a diary coding, double entry or marking process.

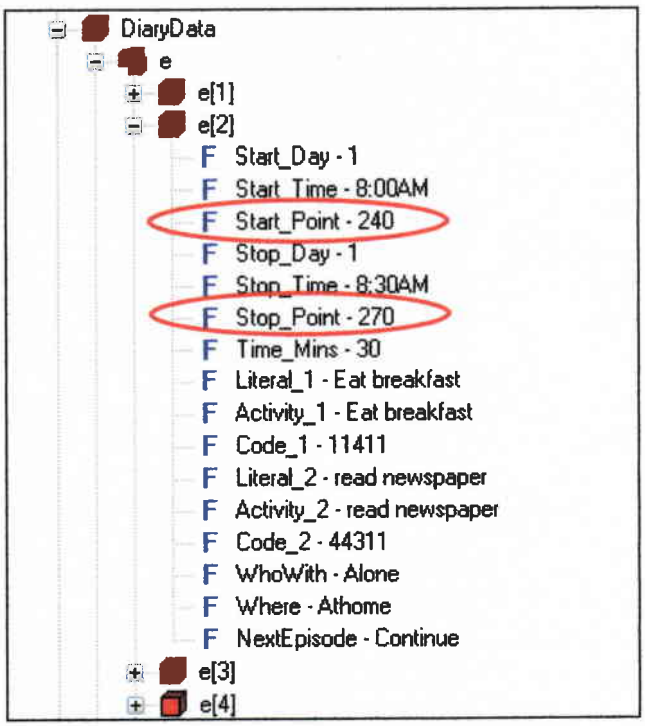


Technical challenge: Tracking time over 48 hours

The Time Use diaries record episodes and activities over a 48 hour period commencing at 4AM on the first day and finishing at 4AM on the third day.

To make it easier to manage these time events, and check the sequencing of episodes, the diary instrument has been set up to convert each time point (consisting of a clock time and day number) into an integer with values from 0 to 2880. These codes then represent the 2880 minutes in a 48 hour period. Each start and end time is duly converted.

Using this simple conversion makes it much easier to check times against each other (to ensure episodes are consecutive) and to calculate time differences or sums.



Technical challenge: Providing alternative data entry methods

In order to synchronise the two methods the instrument uses two arrays, one for each method, containing all the episode fields. The first array is used in the ordinary flow of the questionnaire and the second is used in a table. Both arrays are initiated using the KEEP instruction then a single field called EntryMethod (with two possible values) is used to control which one is the active method and which becomes the passive method.

Depending on which method is active, the Rules in the diary instrument copies all the values of the corresponding active array elements to the other (passive) array.

```
DiaryData.KEEP(CodingMethod,Days_to_do,Minutes_to_do)
DiaryTable.KEEP(CodingMethod,Days_to_do,Minutes_to_do)
{Capture diary episodes using sequential pages}
IF EntryMethod=UseForm THEN
    DiaryData(CodingMethod,Days_to_do,Minutes_to_do)
    {Populate the table with data from the episode entries}
    FOR i:=1 TO 120 DO
        IF DiaryData.e[i].Start_Day<>EMPTY THEN
            DiaryTable.e[i].Start_Day := DiaryData.e[i].Start_Day
            DiaryTable.e[i].Start_Time := DiaryData.e[i].Start_Time
            DiaryTable.e[i].Start_Point := DiaryData.e[i].Start_Point
            ... (etc) ...
            DiaryTable.e[i].WhoWith := DiaryData.e[i].WhoWith
            DiaryTable.e[i].Where := DiaryData.e[i].Where
            DiaryTable.Max_Point := MAX(0,DiaryTable.e[i].Stop_point)
            DiaryTable.e[i].Time_Mins := DiaryData.e[i].Time_Mins
        ENDIF
    ENDDO
ELSE
    DiaryTable(CodingMethod,Days_to_do,Minutes_to_do)
    {Populate the episode array with data from the table}
    FOR i:=1 TO 120 DO
        IF DiaryTable.e[i].Start_Day<>EMPTY THEN
            DiaryData.e[i].Start_Day   := DiaryTable.e[i].Start_Day
            DiaryData.e[i].Start_Time  := DiaryTable.e[i].Start_Time
            DiaryData.e[i].Start_Point := DiaryTable.e[i].Start_Point
            ... (etc) ...
            DiaryData.e[i].WhoWith     := DiaryTable.e[i].WhoWith
            DiaryData.e[i].Where       := DiaryTable.e[i].Where
            DiaryData.Max_Point := MAX(0,DiaryData.e[i].Stop_point)
            DiaryData.e[i].Time_Mins := DiaryTable.e[i].Time_Mins
            IF DiaryTable.e[i].Stop_Point < DiaryTable.Max_Point THEN
                DiaryData.e[i].NextEpisode := Continue
            ENDIF
```

Technical challenge: Table operations

The special block commands of INSERT and DELETE, along with other assignments, are used to insert or delete rows in the episode table and move the remaining data around to achieve the desired outcome (being either insert, delete, split or merge).

These special operations are controlled by setting a Flag to allow the corresponding rules to be executed once. The Flag is reset to the inactive value at the end of that section of the rules to stop the actions from being repeated.

```
{Rules to insert, delete, split or merge entries}
xConfirm.KEEP
xRowNo.KEEP
xAction.KEEP
IF xConfirm=1 THEN
   IF xAction='INSERT' THEN
      aRowno1:=xRowno+1 {this adjustment needed to ensure action at the right line}
      DiaryTable.e.INSERT(aRowno1)
      {The following row gets over the problem of having all rows disappear off route
      when inserting on 1st row Set stop time to start time + 1 minute}
      DiaryTable.e[xRowno].Stop_time := DiaryTable.e[xRowno].Start_time  + (0,1,0)
   ELSEIF xAction='DELETE' THEN
      aRowno1:=xRowno+1 {this adjustment needed to ensure action at the right line}
      DiaryTable.e.DELETE(aRowno1)
   ELSEIF xAction='MERGE' THEN
      aRowno1:=xRowno+1 {this adjustment needed to ensure action at the right line}
      aRowno2:=xRowno
      DiaryTable.e[aRowno2].Stop_time :=DiaryTable.e[aRowno1].Stop_time
      DiaryTable.e[aRowno2].Literal_2 :=DiaryTable.e[aRowno1].Literal_1
      DiaryTable.e[aRowno2].Activity_2:=DiaryTable.e[aRowno1].Activity_1
      ... etc ...
      DiaryTable.e[aRowno2].WhoFor_4  :=DiaryTable.e[aRowno1].WhoFor_3
      aRowno1:=xRowno+2 {this adjustment needed to ensure action at the right line}
      DiaryTable.e.DELETE(aRowno1)
   ELSEIF xAction='SPLIT' THEN
      aRowno1:=xRowno+2 {this adjustment needed to ensure action at the right line}
      aRowno2:=xRowno+1
      aRowno3:=xRowno
      DiaryTable.e.INSERT(aRowno1)
      DiaryTable.e[aRowno2]:=DiaryTable.e[aRowno3] {copies from line above}
      DiaryTable.e[aRowno2].Literal_1 :=DiaryTable.e[aRowno2].Literal_2
      DiaryTable.e[aRowno2].Activity_1:=DiaryTable.e[aRowno2].Activity_2
      ... etc ...
      DiaryTable.e[aRowno2].Literal_3 :=DiaryTable.e[aRowno2].Literal_4
      aDiffmins:=(ABSTime(DiaryTable.e[aRowno2].Stop_time)-ABSTime(DiaryTable.e[aRowno2].Start_time))/1000/60
      IF aDiffMins >1 THEN
         aDiffMins := INT(aDiffMins/2)
         DiaryTable.e[aRowno3].Stop_time:=DiaryTable.e[aRowno3].Start_time+(0,aDiffMins,0)
      ENDIF
   ENDIF
   xConfirm := 0
ENDIF {of xConfirm=1}
```

Technical challenge: Re-aligning episode times

When comparing the dual coding of activity episodes it is not a simple matter of comparing the elements of two arrays. That is because one array or the other may contain more or less entries. This can happen where one operator considers that two or more episodes have taken place for a particular time period, and another operator considers that only one episode has taken place for the same time period. There are many other possibilities for time discrepancies to arise.

So, before the detail of two sets of activity episodes can be compared, they need to be processed and re-aligned based on start times and end times.

A procedure was developed in Maniplus that compares the start times and end times of the two arrays of episode information and makes corresponding re-alignments to all the entries when problems are detected. The re-alignment process involves inserting dummy episodes that have the same start and end times (effectively an episode with zero elapsed time) and contain no activities. The episodes effectively push the subsequent mismatched episode up by one. The whole process is then repeated across all the elements of the array until all entries match on start time or end time or both.

(See the code extracts on the pages which follow).

Re-alignment code (part 1)

```
{Procedure to adjust the pointers for transfer of data}
PROCEDURE PointAdjust
PARAMETERS
   IMPORT Pointer_to_fix : STRING
   IMPORT From_point : INTEGER
INSTRUCTIONS
   IF Pointer_to_fix='A' THEN
      FOR x:=From_point TO 120 DO
         IF PointerA[x]>0 THEN
            PointerA[x]:=PointerA[x]-1
         ELSEIF PointerA[x]=0 THEN
            IF x=From_Point THEN
               PointerA[x]:=PointerA[x-1]
            ELSE
               PointerA[x]:=PointerA[x-1]+1
            ENDIF
            EXITFOR
         ENDIF
      ENDDO
   ELSEIF Pointer_to_fix='B' THEN
      FOR x:=From_point TO 120 DO
         IF PointerB[x]>0 THEN
            PointerB[x]:=PointerB[x]-1
         ELSEIF PointerB[x]=0 THEN
            IF x=From_Point THEN
               PointerB[x]:=PointerB[x-1]
            ELSE
               PointerB[x]:=PointerB[x-1]+1
            ENDIF
            EXITFOR
         ENDIF
      ENDDO
   ENDIF
ENDPROCEDURE

{Procedure to check entries until difference in times is encountered}
{When differences are found adjust one or other by repeating}
PROCEDURE CheckTimes
PARAMETERS
   IMPORT Start_from : INTEGER
   EXPORT Up_to : INTEGER
INSTRUCTIONS
   FOR i:=Start_From TO 120 DO
      {Test if aligned entries exist}
      IF (Diary.DiaryData.e[PointerA[i]].stop_point>0) AND (DiaryQA.DiaryData.e[PointerB[i]].stop_point>0) THEN
         Up_to := i
         {1. Test if start points are the same}
         IF Diary.DiaryData.e[PointerA[i]].start_point = DiaryQA.DiaryData.e[PointerB[i]].start_point THEN
            {Test if end points are the same}
            IF Diary.DiaryData.e[PointerA[i]].stop_point = DiaryQA.DiaryData.e[PointerB[i]].stop_point THEN
               {no adjustment needed}
               Adjust:=0
            ELSE
               {adjustment needed}
               {test if end A is greater than end B}
               IF Diary.DiaryData.e[PointerA[i]].stop_point > DiaryQA.DiaryData.e[PointerB[i]].stop_point THEN
                  {test if end A is greater than or equal to end B+1}
                  IF Diary.DiaryData.e[PointerA[i]].stop_point >= DiaryQA.DiaryData.e[PointerB[i]+1].stop_point THEN
                     Adjust:=i+1
                     PointAdjust('A',Adjust)
                     EXITFOR
                  ENDIF
               {test if end A is less than end B}
               ELSEIF Diary.DiaryData.e[PointerA[i]].stop_point < DiaryQA.DiaryData.e[PointerB[i]].stop_point THEN
                  {test if end B is greater than or equal to end A+1}
                  IF DiaryQA.DiaryData.e[PointerB[i]].stop_point >= Diary.DiaryData.e[PointerA[i]+1].stop_point THEN
                     Adjust:=i+1
                     PointAdjust('B',Adjust)
                     EXITFOR
                  ENDIF
               ENDIF
            ENDIF
```

Re-alignment code (part 2)

```
      {2. Test if start point A is less than start B}
      ELSEIF Diary.DiaryData.e[PointerA[i]].start_point < DiaryQA.DiaryData.e[PointerB[i]].start_point THEN
         {Test if end points are the same}
         IF Diary.DiaryData.e[PointerA[i]].stop_point = DiaryQA.DiaryData.e[PointerB[i]].stop_point THEN
            {no adjustment needed}
            Adjust:=0
         ELSE
            {adjustment needed}
            {test if end A is greater than end B}
            IF Diary.DiaryData.e[PointerA[i]].stop_point > DiaryQA.DiaryData.e[PointerB[i]].stop_point THEN
               {test if end A is greater than or equal to end B+1}
               IF Diary.DiaryData.e[PointerA[i]].stop_point >= DiaryQA.DiaryData.e[PointerB[i]+1].stop_point THEN
                  Adjust:=i+1
                  PointAdjust('A',Adjust)
                  EXITFOR
               ENDIF
            {test if end A is less than end B}
            ELSEIF Diary.DiaryData.e[PointerA[i]].stop_point < DiaryQA.DiaryData.e[PointerB[i]].stop_point THEN
               {test if end B is greater than or equal to end A+1}
               IF DiaryQA.DiaryData.e[PointerB[i]].stop_point >= Diary.DiaryData.e[PointerA[i]+1].stop_point THEN
                  Adjust:=i+1
                  PointAdjust('B',Adjust)
                  EXITFOR
               ENDIF
            ENDIF
         ENDIF
         {3. Test if start point B is less than start A}
         ELSEIF DiaryQA.DiaryData.e[PointerB[i]].start_point < Diary.DiaryData.e[PointerA[i]].start_point THEN
            {Test if end points are the same}
            IF Diary.DiaryData.e[PointerA[i]].stop_point = DiaryQA.DiaryData.e[PointerB[i]].stop_point THEN
               {no adjustment needed}
               Adjust:=0
            ELSE
               {adjustment needed}
               {test if end B is greater than end A}
               IF DiaryQA.DiaryData.e[PointerB[i]].stop_point > Diary.DiaryData.e[PointerA[i]].stop_point THEN
                  {test if end B is greater than or equal to end A+1}
                  IF DiaryQA.DiaryData.e[PointerB[i]].stop_point >= Diary.DiaryData.e[PointerA[i]+1].stop_point THEN
                     Adjust:=i+1
                     PointAdjust('B',Adjust)
                     EXITFOR
                  ENDIF
               {test if end B is less than end A}
               ELSEIF DiaryQA.DiaryData.e[PointerB[i]].stop_point < Diary.DiaryData.e[PointerA[i]].stop_point THEN
                  {test if end A is greater than or equal to end B+1}
                  IF Diary.DiaryData.e[PointerA[i]].stop_point >= DiaryQA.DiaryData.e[PointerB[i]+1].stop_point THEN
                     Adjust:=i+1
                     PointAdjust('A',Adjust)
                     EXITFOR
                  ENDIF
               ENDIF
            ENDIF
         ENDIF
      ELSE
         Up_to := 121
         EXITFOR
      ENDIF
   ENDDO
ENDPROCEDURE
```

Technical challenge: Scoring and marking instrument

With the two sets of coded episodes aligned on times, it is then important to enable the discovering and recording of differences and then enable the marking process to resolve the differences without actually changing the recorded values.

Detection of differences is achieved by using edit checks between the matched pairs of episodes. For these edit checks to work, the involved fields are set up using the ASK method rather than the SHOW method. The edit results are piped into a series of fields defined using the EDITTYPE attribute.

Absence of a mark is part of the edit logic

Piping of edit result

```
{Apply checks when there is no marking done}
IF Marking[i].Time_Content_Mark=Nomark THEN
    {Apply comparison edits for activity 1}
    CHECK
        CodeDiff1[i] := x[i].Code_1 = x[i-1].Code_1
        INVOLVING (x[i].Activity_1,x[i-1].Activity_1)
        "Activity 1 should be the present and coded the same"
```

For convenience these edit flags are set up in arrays so that they can be counted and also recorded permanently for reporting purposes. This is done by copying edit results once only to a series of backup array elements.

```
{copy edit results to backup entries once only}
IF CopyFlag=1 THEN
    For i:=1 to 120 DO
        j:=i*2-1
        k:=j+1
        TimeDiff[j]     := TimeDiff[k]
        CodeDiff1[j]    := CodeDiff1[k]
        CodeDiff2[j]    := CodeDiff2[k]
        CodeDiff3[j]    := CodeDiff3[k]
        CodeDiff4[j]    := CodeDiff4[k]
        WhoForDiff1[j]  := WhoForDiff1[k]
        WhoForDiff2[j]  := WhoForDiff2[k]
        WhoForDiff3[j]  := WhoForDiff3[k]
        WhoForDiff4[j]  := WhoForDiff4[k]
        WhereDiff[j]    := WhereDiff[k]
        WhoWithDiff[j]  := WhoWithDiff[k]
    ENDDO
    CopyFlag:=0
ENDIF
```

Technical challenge: Scoring and marking instrument (cont'd)

The recorded values (now displayed using the ASK method) are preserved by transferring them at the start of an edit session to a temporary array which is used to refresh the actual values if and when the operator tries to change them.

```
{populate the temporary array y so that we can restore any changed values to permanent array x}
{this happens only once when the record is first opened - the y array is discarded later}
IF aYFlag=EMPTY THEN
   FOR i:=1 TO 240 DO
       y[i]:=x[i]
   ENDDO
   aYFlag:=1
ENDIF
```

ASK and SHOW parameters passed into the call to the block

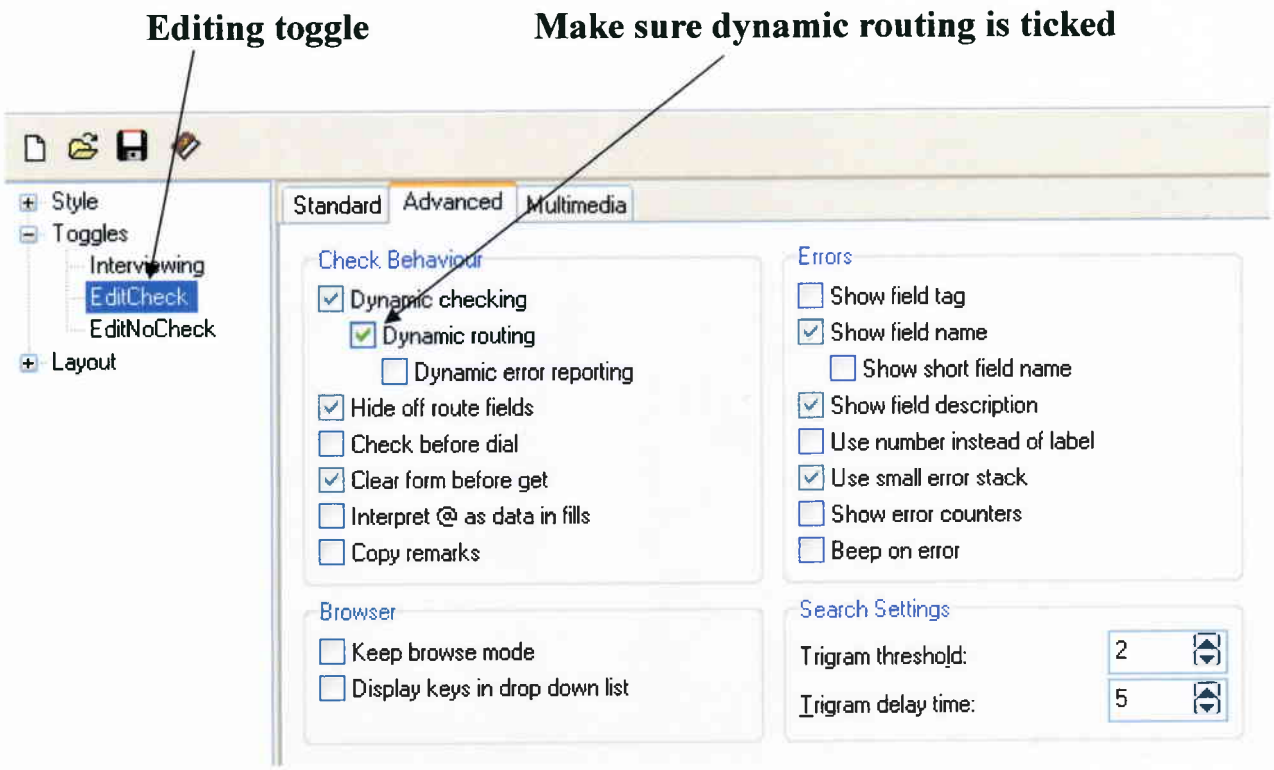
```
{ASK the original episode line (odd value index) then show the dual episode line (even value index)}
IF I MOD 2=1 THEN {Odd value index}
    {determine SHOW/ASK based on marking that has been done}
    IF Marking[i].Time_Content_Mark=NoMark OR Marking[i].Time_Content_Mark=CoderA THEN
        aAskTime:=Yes
    ELSE
        aAskTime:=No
    ENDIF
    IF Marking[i+1].Time_Content_Mark=NoMark OR Marking[i+1].Time_Content_Mark=CoderA THEN
        aAskContent:=Yes
    ELSE
        aAskContent:=No
    ENDIF
    {set the default marking value if empty}
    IF Marking[i].Time_Content_Mark=EMPTY THEN
        Marking[i].Time_Content_Mark:=NoMark
    ENDIF
    {display the time marking field}
    Marking[i](i)
    {display the episode data}
    x[i](i, aDay, aStart_time,  aStart_point, aNext, aActionParm, i, aAskTime, aAskContent)
    {restore the original values from the temporary copy - to overcome any changes made}
    x[i]:=y[i]
```

Data is restored from temporary backup

The edit checks are cleared through the marking entries made by the operator by ensuring that the mark field is part of the logic of the edit check.

Technical challenge: Scoring and marking instrument (cont'd)

The edit checks are made visible (as red marks) by using the Editing layout from the Mode Library. Ability to follow the routing during editing is enabled by changing the dynamic routing setting for the second behaviour toggle in the Mode Library (as shown).



The Editing layout and the second toggle are invoked via line command parameters when the Edit session is initiated within the Maniplus Marking interface.

Invoking the editing layout and behaviour toggle 2

```
aResult := Scoring.EDIT('/X /K'+aNotesKey[pDiaryNum]
      +' @TUSDiary_System.ini /MDiaryDiffs.bmf /P2 /T2')
```