```vb
Option Explicit

Const ReturnNoSolution As Integer = -32767
Const ReturnNotMovableDelta As Integer = -1000

Dim RecurLevel As Integer
Dim ListRecurCall As String



'------------------------------------------------------------- AU_NPM_MarginManageProtectedFlights
' modify the FDATime
' lprioModel must be assigned to GPrioModel_TimeMode_OnSchedule or GPrioModel_TimeMode_OnMargin
'  in this case we use the schedule or the margin to make the calculation

' lAllFl  is list off flights to find Margin solution (All except Pflights and ExplicitB flights)
' when finding margin solution, if default priority value is baseline, prio is put to lowest prio
' If  Time not after is greater then the Hotspot end, initialize to the end of the hotspot
' BE CAREFULL a B priority on margins flights exclude flight from management
Function AU_NPM_ManageMarginPrioFlights_Main(lprioModel As Integer, _
            lAll_AUFlights As CL_AllFlights, lMyFlightsIx As CL_AUFlightsIx, _
            ByRef lAllFl() As Integer, lAllFl_nb As Integer, _
            ByRef lAddFlight() As Integer, ByRef lMarginflight_Nb As Integer) As Integer

    Dim li As Integer
    Dim lj As Integer
    Dim lName As String
    Dim ltime As Date


    'Dim lAllFl_prio() As Integer ' containt prio off All flight for this AU (Ix in Allflight)

    'Dim lAllFlSortedOnBaseline() As Integer ' containt All flight to manage the margin sorted

    Dim lMarginFl() As Integer ' containt All flight twho have a  margin
    Dim lMarginSortedFl() As Integer ' containt All flight to manage the margin sorted
    Dim lMarginFl_nb As Integer ' nb of flight to manage the margin

    Dim lPrioOnlyFl() As Integer ' containt All flight twho have a  margin
    Dim lPrioOnlySortedFl() As Integer ' containt All flight to manage the margin sorted
    Dim lPrioOnlyFl_nb As Integer ' nb of flight to manage the margin



    Call EX_Mess(EX_MESS_Start, "AU NPRIO MARGINandPrio: " & lAll_AUFlights.AUName)
    'ListRecurCall = ""

    ' do nothing if only 1 flight
    If lAllFl_nb < 2 Then
        Exit Function
    End If


    Dim lMargeIx As Integer

    ' slot management
    Dim lMySlotsValue() As Date
    Dim lMySlotsValueSorted() As Date
    Dim lMySlotsUsed() As Integer

    ' get the flight with margin to manage
    ' get the list of flight with margins to manage


    lMarginFl_nb = 0
    lPrioOnlyFl_nb = 0

    ' get margin flights and prio only flights
    ' BE CAREFULL the management of explicit B on margin flight
    ' explicit B normally not part of the flights
    If lAllFl_nb > 0 Then
        ReDim lMarginFl(lAllFl_nb)
        ReDim lMySlotsValue(lAllFl_nb)
        ReDim lPrioOnlyFl(lAllFl_nb)
        ReDim lMySlotsUsed(lAllFl_nb)
        For li = 0 To lAllFl_nb - 1
            lMySlotsValue(li) = lAll_AUFlights.GetFDATime(lAllFl(li))
            lMySlotsUsed(li) = -1
            If (lAll_AUFlights.GetMarginNotAfterTimeIsInit(lAllFl(li)) = True) And _
                (lAll_AUFlights.GetPrio(lAllFl(li)) <> GPrioSuspended) Then
                lMarginFl(lMarginFl_nb) = lAllFl(li)
                lMarginFl_nb = lMarginFl_nb + 1
            Else
                lPrioOnlyFl(lPrioOnlyFl_nb) = lAllFl(li)
                lPrioOnlyFl_nb = lPrioOnlyFl_nb + 1
            End If
        Next li

        ' get my slots and manage the list of margin or prio flights
        ReDim lMySlotsValueSorted(lAllFl_nb)

        ' Sort my Time slots (by FDATime)
        Call AU_NPS_SortATimeTable(lMySlotsValue, lMySlotsValueSorted, lAllFl_nb)
        Erase lMySlotsValue

        If lMarginFl_nb < 1 Then
            Erase lMarginFl
        Else

            ' Sort my Margin flights by prio and Margins and Schedule
            ReDim lMarginSortedFl(lMarginFl_nb)
            Call AU_NPS_SortByPrioAndMarginTimeNotAfterAndBaselineTime(lAll_AUFlights, lMarginFl, lMarginSortedFl,
lMarginFl_nb)
            Erase lMarginFl

            ' loop on earch Margin flights
            ' put Margin flights on available slot

                Call EX_Log_Init

            For lMargeIx = 0 To lMarginFl_nb - 1

                ' for test
                Dim lCallsign As String
                Dim lMarginTime As Date
                Dim lMarginFlightIx As Integer

                lMarginFlightIx = lMarginSortedFl(lMargeIx)
                lCallsign = lAll_AUFlights.GetCallsignICAO(lMarginFlightIx)

                Dim lSlotAssigned As Integer

                If lMargeIx = 73 Then
                    lMargeIx = lMargeIx
                End If

                lMarginTime = lAll_AUFlights.GetMarginNotAfterTime(lMarginFlightIx)

                lSlotAssigned = AU_NPM_ManageMarginPrioFlights_ManageTimeSolution(lAll_AUFlights, _
                    lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, _
                    lMarginTime, lMarginFlightIx)



                ' Test if there is a slot
                If lSlotAssigned < 0 Then
                    ' no time solution
                    Call AU_NPM_MsgboxStop(" ERROR TO ASSIGN Margin flight to a slot : " & lCallsign, _
                            lAll_AUFlights, lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb)
                End If

            Next lMargeIx
        End If

        ' manage the other type of flights
        If lPrioOnlyFl_nb < 1 Then
            'Erase lMySlotsValue
            Erase lPrioOnlyFl
            'Erase lMySlotsUsed
        Else
            ' AU slot has been assigned to Margin flights
            ' now manage the other flights (prio only)
            ReDim lPrioOnlySortedFl(lPrioOnlyFl_nb)
            Call AU_NPS_SortByPrioAndSchedule(lAll_AUFlights, lPrioOnlyFl, lPrioOnlySortedFl, lPrioOnlyFl_nb)
            Erase lPrioOnlyFl

            ' assign prio flights be carrefull baseline flights must be assign first then flight with number
            Call AU_NPM_ManageMarginPrioFlights_AssignOtherFlights(lAll_AUFlights, _
                lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, _
                lPrioOnlySortedFl(), lPrioOnlyFl_nb)

            Erase lMarginSortedFl

        End If


        ' return the nb of margin flights
        lMarginflight_Nb = lMarginFl_nb
    End If

    ' at the en Pack the flights on slots (use the available AU slot
    ' needed if there is some Suspended flights
    If lMarginFl_nb > 0 Or lPrioOnlyFl_nb > 0 Then
        Call AU_NPM_ManageMarginPrioFlights_UseAvailableSlots(lAll_AUFlights, _
                lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb)

    End If

    ' assign the flights to the slots
    '------ Update the FDA time ------
    ' First assigne the FDA time on margins flights
    For li = 0 To lAllFl_nb - 1
        ltime = lMySlotsValueSorted(li)
        ' if the slot is used by a margin flight assign it
        If lMySlotsUsed(li) <> -1 Then
            ' assign the Margin flights
            lName = lAll_AUFlights.GetCallsignICAO(lMySlotsUsed(li))
            Call lAll_AUFlights.SetFDATime(lMySlotsUsed(li), lMySlotsValueSorted(li))
        End If
    Next li
```

```
191
192     ' ---------- Manage the Suspended flights
193     If lPrioOnlyFl_nb > 0 Then
194
195         ' Assign it the hotspot end time in fdaValue
196         Call AU_NPM_ManageMarginPrioFlights_AssignSuspendedFlights(lAll_AUFlights, _
197                 lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, _
198                 lPrioOnlySortedFl(), lPrioOnlyFl_nb)
199     End If
200
201     Erase lPrioOnlySortedFl
202     Erase lMySlotsValueSorted
203     Erase lMySlotsUsed
204
205
206     ' return nb of impacted flights to add on next list
207     AU_NPM_ManageMarginPrioFlights_Main = 0
208
209     Call EX_Mess(EX_MESS_End, "AU NPRIO MARGINandPrio: " & lAll_AUFlights.AUName)
210
211
212 End Function
213
214 '---------------------------------------- AU_NPM_ManageMarginPrioFlights_ManageTimeSolution
215 ' try and manage the solution before the target time
216 ' At this stage the target slot is already used by another flight
217 ' first look if there is an available slot before the target one to shift earlier the others
218 ' if yes :
219 '     try to shift the flights to make a hole at the target slot
220 '     otherwise give a slot before for this flight
221 ' if no slot return -1
222 Function AU_NPM_ManageMarginPrioFlights_ManageTimeSolution( _
223             lAll_AUFlights As CL_AllFlights, _
224             ByRef lMySlotsValueSorted() As Date, ByRef lMySlotsUsed() As Integer, lAllFl_nb As Integer, _
225             lTargetTime As Date, _
226             lFlightIx As Integer) As Integer
227
228     'Dim lSlotEarlierPossible As Integer
229
230     Dim lAvailableSlot_Earlier As Integer
231     Dim lAvailableSlot_Later As Integer
232
233     Dim lEarliestTime As Date
234     Dim lTargetSlot As Integer
235
236     Dim lReturn As Integer
237
238     'lAvailableSlot_Earlier = -1
239
240     RecurLevel = 0
241     'Call EX_Log_Init
242
243     ' earliest time of the margin flight
244     lEarliestTime = lAll_AUFlights.GetRefBlockTime(lFlightIx) _
245                                                 - GHspt_FlightEarlyDeparture_forDate
246
247
248
249     lTargetSlot = AU_NPMF_GetTargetSlots(lMySlotsValueSorted, lAllFl_nb, lTargetTime, lEarliestTime)
250
251     ' Test if there is a slot
252     If lTargetSlot < 0 Then
253         ' no time solution
254         Call AU_NPM_MsgboxStop(" ERROR TO DO SOMETHING Margin flight with no time solution : " &
lAll_AUFlights.GetCallsignICAO(lFlightIx), _
255                 lAll_AUFlights, lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb)
256
257                 Call EX_Log(RecurLevel, "  --> No slot solution for " & lFlightIx & " on targetTime " & lTargetTime)
258         lReturn = -1
259     Else
260         ' try manage solution by shifting flights earlier first
261         ' the slot could ber later then the one asked
262         lAvailableSlot_Earlier = AU_NPM_ManageMarginPrioFlights_ManageSolutionEarlier(lAll_AUFlights, _
263                 lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, _
264                 lTargetSlot, lFlightIx)
265
266         If lAvailableSlot_Earlier > -1 Then
267             ' Found a place in the slot list earlierassign it
268             lMySlotsUsed(lAvailableSlot_Earlier) = lFlightIx
269             'ListRecurCall = ListRecurCall & " S= " & lAvailableSlot_Earlier & "   |"
270             Call EX_Log(RecurLevel, "End FL: " & lFlightIx & " EARLIER Solution is slot: " & lAvailableSlot_Earlier)
271
272
273             lReturn = lAvailableSlot_Earlier
274         Else
275             ' No possible slot earlier
276             ' test and get if there is available slot later
277             lAvailableSlot_Later = AU_NPMF_GetLaterAvailableSlots(lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, _
278                                         lTargetSlot + 1, lEarliestTime)
279
280             If lAvailableSlot_Later > -1 Then
281                 ' put it at this place
282                 lMySlotsUsed(lAvailableSlot_Later) = lFlightIx
283                 Call EX_Log(RecurLevel, "End FL: " & lFlightIx & " LATER Solution is slot: " & lAvailableSlot_Later)
284                 lReturn = lAvailableSlot_Later
285             Else
```



```
286                 ' in this case no possibility to shift flight earlier
287                 ' and no slot available after
288                 ' otherwhise there is other possibilities .......
289                 lReturn = -1
290                 Call EX_Log(RecurLevel, " --> ERR: No slot for " & lFlightIx)
291
292                 Call AU_NPM_MsgboxStop("ManageTimeSolution PB of NB of available slot not OK for : " & _
293                         lAll_AUFlights.GetCallsignICAO(lFlightIx) & " id= " & lFlightIx, _
294                         lAll_AUFlights, lMySlotsValueSorted, lMySlotsUsed, lAllFl_nb)
295             End If
296         End If
297     End If
298
299
300     AU_NPM_ManageMarginPrioFlights_ManageTimeSolution = lReturn
301
302 End Function
303
304
305
306 '---------------------------------------- AU_NPM_ManageMarginPrioFlights_ManageSolutionEarlier
307 ' Manage the solution at or before the target time or later if not found ealier
308 ' At this stage the target slot is already used by another flight
309 '       try to manage the solution by shifting the flight who occupy the slot earlier
310 '       because this flight has a higher priority because it's assigned before
311
312 'Input :
313 '     - the list of slot
314 '     - the slot used in this list by a previous assigned flight,
315 '     - the flight to be managed
316 '     - the started target slot
317
318 'Output:
319 '     - the available slot for this flight
320
321
322 ' first check if there is an available slot before the target one (needed to find a solution earlier)
323 ' if No available slot before : stop this function and return -1
324 ' if there is an empty slot before (minimum constraint to have a earlier solution)
325 '     loop from the current needed slot to the latest slot of the list
326 '                     (look also on later slot to return a slot later if no solution before)
327 '         Call "MoveFlightEarlier" the recursive function to try to shift the flights on earlier slot
328 '                 to make a hole to assign the flight on a slot
329 '         if a slot is found: return the slot found (end of loop)
330 '     End of the loop (at this stage the slot found could be later)
331
332 ' if no possible slot return -1
333 ' otherwise return the slot found
334
335
336 Function AU_NPM_ManageMarginPrioFlights_ManageSolutionEarlier( _
337             lAll_AUFlights As CL_AllFlights, _
338             ByRef lMySlotsValueSorted() As Date, ByRef lMySlotsUsed() As Integer, lAllFl_nb As Integer, _
339             lTargetSlot As Integer, _
340             lMarginFlightIx As Integer) As Integer
341
342     Dim lSlotEarlierPossible As Integer
343     Dim lAvailableSlot_Earlier As Integer
344     Dim lTargetMove As Integer
345
346     lAvailableSlot_Earlier = -1
347
348 ' to test
349 If lAll_AUFlights.GetCallsignICAO(lMarginFlightIx) = "AFR165Z" Then
350     lMarginFlightIx = lMarginFlightIx
351 End If
352
353
354     lSlotEarlierPossible = AU_NPMF_GetIxOfEarlierFlightCanMove(lAll_AUFlights, _
355                 lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, lTargetSlot, lMarginFlightIx)
356
357     ' some time say no but there is SO overwrite it
358     'lSlotEarlierPossible = 0
359
360     If lSlotEarlierPossible > -1 Then
361     ' move flight earlier is possible
362     'ListRecurCall = ListRecurCall & " | T:" & lMarginFlightIx & " " & lTargetSlot & " -> "
363
364
365         ' loop until solution found or end of slots
366         ' apply the recursive move flight earlier function from the target slot to the end of the slots
367         ' until a solution is found
368         For lTargetMove = lTargetSlot To lAllFl_nb - 1
369             'ListRecurCall = "FL: " & lMarginFlightIx & " Slot: " & lTargetMove & " -> "
370             Call EX_Log(RecurLevel, "Start FL: " & lMarginFlightIx & " Slot: " & lTargetMove & " -> ")
371
372             ' test and manage if there is a slot earlier
373             lAvailableSlot_Earlier = AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier(lAll_AUFlights, _
374                 lMySlotsValueSorted(), lMySlotsUsed(), lAllFl_nb, _
375                 lTargetMove, lMarginFlightIx)
376
377             'ListRecurCall = ListRecurCall & " Target= " & lTargetMove & " Out=" & lAvailableSlot_Earlier")
378
379             If lAvailableSlot_Earlier > -1 Then
380                 ' Found a place in the slot list earlierassign it
381                 'lMySlotsUsed(lAvailableSlot_Earlier) = lMarginFlightIx
```

```
382                 lTargetMove = lAllFl_nb
383                 'ListRecurCall = ListRecurCall & " S= " & lAvailableSlot_Earlier & "   |"
384                 Call EX_Log(RecurLevel, "En FL: " & lMarginFlightIx & " Slot: " & lAvailableSlot_Earlier)
385
386             Else
387                 'ListRecurCall = ListRecurCall & " ..Next.."
388                 Call EX_Log(RecurLevel, "   FL: " & lMarginFlightIx & " Slot Not OK .. Next")
389             End If
390
391         Next lTargetMove
392
393
394
395         If lAvailableSlot_Earlier < 0 Then
396
397             ' LG2018-07 No now it's OK because use of all earlier slots
398             ' If a flight can't be on the earlier slot it's meen that ther is a slot later !!!
399             ' SO only return -1
400
401             Call EX_Log(RecurLevel, "En FL: " & lMarginFlightIx & " No Solution found earlier even with hole")
402 GoTo lNext1:
403             ' a solution can exist
404             ' here it's because an affected flight lock the list because it's after another one
405
406             If AU_NPM_ManageMarginPrioFlights_IsFlightScheduleCompatible(lAll_AUFlights, _
407                     lMySlotsValueSorted(lSlotEarlierPossible), lMarginFlightIx) = True Then
408                 ' this fligh is compatible
409                 lAvailableSlot_Earlier = lSlotEarlierPossible
410                 Call EX_Log(RecurLevel, "En FL: " & lMarginFlightIx & " Direct Slot: " & lAvailableSlot_Earlier)
411             Else
412                 ' no solution because current and earlier slot not compatible with schedule
413                 ' something wrong here
414                 Call EX_Log(RecurLevel, "En FL: " & lMarginFlightIx & " ERROR No slot ")
415
416                 Call AU_NPM_MsgboxStop("Margins PB possible slot before but not found solution !!! : " & _
417                         lAll_AUFlights.GetCallsignICAO(lMarginFlightIx) & " id= " & lMarginFlightIx & vbCr &
ListRecurCall, _
418                         lAll_AUFlights, lMySlotsValueSorted, lMySlotsUsed, lAllFl_nb)
419             End If
420
421 lNext1:
422
423         End If
424
425     Else
426         lAvailableSlot_Earlier = -1
427     End If
428
429
430     AU_NPM_ManageMarginPrioFlights_ManageSolutionEarlier = lAvailableSlot_Earlier
431
432 End Function
433
434
435
436
437
438 ' Recursive function to find and return a slot for a flight
439 ' If the slot is not free it make it available by shifting already assign flights earlier (the already assign flights
have higher priority)
440 ' this function make the shift of the flights only if all flights can be shifted (recursive test before shifting)
441 'Input :
442 '     - the list of slot
443 '     - the slot used in this list by a previous assigned flight,
444 '     - the flight to be managed
445 '     - the started target slot
446
447 'Output:
448 '     - return >-1 : the available slot position set to free for this flight
449 '     - return = -1  blocking point, no possible shift earlier
450 '     - return =  a negative value starting at -1000
451 '                         + the negative value of the slot corresponding to a Unmovable flight
452 '                          ex: -1051 : the slot 51 is occupied by a Unmovable flight
453
454
455
456 Function AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier(lAll_AUFlights As CL_AllFlights, _
457             ByRef lMySlotsValueSorted() As Date, ByRef lMySlotsUsed() As Integer, lSlots_nb As Integer, _
458             lTargetIx As Integer, lFlightIx_ToMove As Integer) As Integer
459
460     Dim li As Integer
461     Dim lReturn As Integer
462     Dim lTo As Integer
463     Dim lFrom As Integer
464     Dim lTryPreviousFlight As Boolean
465     Dim lEarliestTime As Date
466     Dim lPos As Integer
467     Dim lCallsign As String
468
469     ' to test
470     RecurLevel = RecurLevel + 1
471     lCallsign = lAll_AUFlights.GetCallsignICAO(lFlightIx_ToMove)
472
473     Call EX_Log(RecurLevel, "Mv FL: " & lFlightIx_ToMove & " to Slot: " & lTargetIx & "(used by " &
lMySlotsUsed(lTargetIx) & ")")
474
```



```
475 ' ca boucle ici !!!!!!!
476 If lFlightIx_ToMove = 284 And lTargetIx = 57 And lMySlotsUsed(lTargetIx) = 272 Then
477     lTargetIx = lTargetIx
478 End If
479
480     ' try to make the target slot available by shifting it earlier
481     lTryPreviousFlight = True
482     lFrom = lTargetIx
483     lTo = lTargetIx - 1
484
485     ' initial condition
486     '     FromSlot = Target slot
487     '     ToSlot = Target slot - 1  slot to be use for the first shift in case the target one is used
488
489     ' loop on the flights until a solution is found or no possible solution
490     While lTryPreviousFlight = True
491         ' test if the flight to put on the target slot (FromSlot)is time compatible
492         '      (with its reference time - Airport early schedule duration)
493         If AU_NPM_ManageMarginPrioFlights_IsFlightScheduleCompatible(lAll_AUFlights, _
494                 lMySlotsValueSorted(lFrom), lFlightIx_ToMove) = False Then
495         'if the flight is not time compatible with the FromSlot
496             'stop the loop and return the index of the tested slot as negative value
497             '          add -1000 to be sur the value 0 is well managed
498
499                 lReturn = ReturnNotMovableDelta - lFrom
500                 lTryPreviousFlight = False
501                 Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Slot: " & lFrom & " To Early .. Return : " &
lReturn)
502         Else
503         'Else : the flight is compatible with the FromSlot
504             If lMySlotsUsed(lFrom) = -1 Then
505             'if the Fromslot is empty: no flight assigned on it
506                 'return this slot and stop the loop
507                 lReturn = lFrom
508                 lTryPreviousFlight = False
509                 Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Slot empty OK .. Return : " & lReturn)
510
511             Else
512             'Else : the slot is used
513                 'try to move the flight assigned in the used slot at an earlier position (to the ToSlot)
514                 'if the ToSlot is < 0 (it is not possible to move before because it was the first slot)
515                 '   in this case there is no solution:  and end the loop
516                 If lTo < 0 Then
517                     ' we are at the beginning of the slot list without finding a solution
518                     ' no possible solutions return -1
519                     lReturn = ReturnNoSolution
520                     lTryPreviousFlight = False
521                     Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Slot -- No SOLUTION -- At the end of
Slots list Return : " & lReturn)
522
523                 Else
524                 'Else: the slot is used
525                     'try to move the flight using the ToSlot to a earlier position by using this same recursive
function
526                     'call lPos =  MoveflightEarlier with in parameter: with the ToSlot target and with the flight in
the FromSlot position
527
528                     lPos = AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier(lAll_AUFlights, _
529                          lMySlotsValueSorted(), lMySlotsUsed(), lSlots_nb, _
530                          lTo, lMySlotsUsed(lFrom))
531
532                     'if the function return a positive value : (lPos > -1)
533
534                     If lPos > -1 Then
535                         'a compatible slot has been found
536                         'make the use of this slot effective (assign it)
537                             'put the flight use to call the recursive function on the Slot position returned by it
538                             'empty the slot used by it previously
539                             'end the loop and return the empty slot
540                         lMySlotsUsed(lPos) = lMySlotsUsed(lFrom)
541                         lMySlotsUsed(lFrom) = -1
542
543                         lReturn = lFrom 'return the slot put to free by the shift
544                         lTryPreviousFlight = False
545
546                         Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Recur OK return " & lReturn & _
547                                 " Flight move from: " & lFrom & " To: " & lPos)
548
549                     ElseIf lPos = ReturnNoSolution Then
550                     'Else if the function return a -1 value (no possible solution) : lPos = -1
551                         ' return no possible solution and close the loop
552                         ' no solution because flights impossible to move
553                         ' cannot create a hole
554                         lReturn = ReturnNoSolution
555                         lTryPreviousFlight = False
556                         Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Recur NO SOLUTION ")
557
558                     Else
559                     'Else : the function return a negative value < -1 a flight is blocked on its position lPos =
-1xxx
560                         'continue to loop with an earlier position
561
562                         'FromSlot = Slot position used to find solution on a flight blocked :
563                         '    Slot position -1 of the flight blocked
564                         '    (be careful : corresponding to the ToSlot position when the function is called)
565                         'ToSlot      = FromSlot - 1
```

```
566
567                         '                       test if not the earliest possible Slot to test
568                         '                        If FromSlot position is > -1 , continue the loop
569                         '                        otherwise stop the loop and end by a -1 solution (no solution)
570                         lFrom = -lPos + ReturnNotMovableDelta
571                         lTo = lFrom - 1
572
573                         If lFrom < 0 Then
574                             ' From from next loop must be >= 0 otherwhise no solution, stop the loop
575                             lTryPreviousFlight = False
576                             lReturn = ReturnNoSolution
577
578                             Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Recur NO solution no more Slot to
check " & _
579                                 lFrom)
580                         Else
581                             lTryPreviousFlight = True
582
583                             Call EX_Log(RecurLevel, "  FL: " & lFlightIx_ToMove & " Recur Next target Dde: " & _
584                                     lTargetIx & " Check: " & lFrom)
585
586                         End If
587
588                     End If
589                 End If
590             End If
591         End If
592     Wend
593
594     AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier = lReturn
595     RecurLevel = RecurLevel - 1
596 End Function
597
598
599
600
601
602
603
604
605
606 ' Recursive function to find and return a slot for a flight
607 ' If the slot is not free it make it available by shifting already assign flights earlier (the already assign flights
have higher priority)
608 ' this function make the shift of the flights only if all flights can be shifted (recursive test before shifting)
609 'Input :
610 '     - the list of slot
611 '     - the slot used in this list by a previous assigned flight,
612 '     - the flight to be managed
613 '     - the started target slot
614
615 'Output:
616 '     - return >-1 : the available slot position set to free for this flight
617 '     - return = -1  blocking point, no possible shift earlier
618 '     - return =  a negative value starting at -1000
619 '                         + the negative value of the slot corresponding to a Unmovable flight
620 '                          ex: -1051 : the slot 51 is occupied by a Unmovable flight
621
622
623 Function AU_NPM_ManageMarginPrioFlights_MoveFlightEarlierOLD(lAll_AUFlights As CL_AllFlights, _
624             ByRef lMySlotsValueSorted() As Date, ByRef lMySlotsUsed() As Integer, lSlots_nb As Integer, _
625             lTargetIx As Integer, lFlightIx_ToMove As Integer) As Integer
626
627     Dim li As Integer
628     Dim lReturn As Integer
629     Dim lTo As Integer
630     Dim lFrom As Integer
631     Dim lTryPreviousFlight As Boolean
632     Dim lEarliestTime As Date
633     Dim lPos As Integer
634     Dim lCallsign As String
635
636     ' to test
637     RecurLevel = RecurLevel + 1
638     lCallsign = lAll_AUFlights.GetCallsignICAO(lFlightIx_ToMove)
639
640     Call EX_Log(RecurLevel, "Rec Start Mv FL: " & lFlightIx_ToMove & " to Slot: " & lTargetIx & "(used by " &
lMySlotsUsed(lTargetIx) & ")")
641
642
643     ' make it available by shifting earlier previous flights
644     lTryPreviousFlight = True
645     lFrom = lTargetIx
646     lTo = lTargetIx - 1
647
648
649     ' loop on the flights because some of them could not be moved earlier due to schedule
650     While lTryPreviousFlight = True
651         ' test if the target slot is compatible (with reference time - Airport early schedule duration)
652         If AU_NPM_ManageMarginPrioFlights_IsFlightScheduleCompatible(lAll_AUFlights, _
653                 lMySlotsValueSorted(lFrom), lFlightIx_ToMove) = False Then
654                 ' the flight is not compatible with the reference time of the target
655                 ' stop the loop on this flight and return the last tested flight (as negative value)
656                 ' add -1000 to be sur the value 0 is managed
657                 '  in recursive calling function, its the To target (in the current one it's the from or an earlier
one)
```



```
658                 lReturn = ReturnNotMovableDelta - lFrom
659                 lTryPreviousFlight = False
660                 Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " Slot: " & lFrom & " To Early .. Return :
" & lReturn)
661             Else
662                 If lMySlotsUsed(lFrom) = -1 Then
663                     'GOOD the slot is empty
664                     lReturn = lFrom
665                     lTryPreviousFlight = False
666                     Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " Slot empty OK .. Return : " & lReturn)
667
668                 Else
669                     ' the slot is used : try to move the used slot at an  earlier position
670                     If lTo < 0 Then
671                         ' no possible solutions
672                         ' we are at the beginning of the slot list without finding a solution
673                         lReturn = ReturnNoSolution
674                         lTryPreviousFlight = False
675                         Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " Slot -- No SOLUTION -- At the end of
Slots list Return : " & lReturn)
676
677                     Else
678                         ' slot used, try to move the flight using the slot to the lTo position
679                         ' use recursive call to move the used slot earlier
680
681                         lPos = AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier(lAll_AUFlights, _
682                              lMySlotsValueSorted(), lMySlotsUsed(), lSlots_nb, _
683                              lTo, lMySlotsUsed(lFrom))
684
685                         If lPos > -1 Then
686                             ' found a empty good slot
687                             ' make the swap slot
688                             lMySlotsUsed(lPos) = lMySlotsUsed(lFrom)
689                             lMySlotsUsed(lFrom) = -1
690
691                             lReturn = lFrom 'return the slot put to free by the shift
692                             lTryPreviousFlight = False
693
694                             Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " OK return " & lReturn & _
695                                     " Flight move from: " & lFrom & " To: " & lPos)
696
697                         ElseIf lPos = ReturnNoSolution Then
698                             ' no solution because flights impossible to move
699                             ' cannot create a hole
700                             ' Normaly a later solution is possible
701                             lReturn = ReturnNoSolution
702                             lTryPreviousFlight = False
703                             Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " NO SOLUTION ")
704
705                         Else
706                             ' a negative value is returned if the position is not movable
707                             ' continu to loop with an earlier position
708
709                             'from is initiated and used for next test of ealier slot in this loop
710                             lFrom = -lPos + ReturnNotMovableDelta
711                             lTo = lFrom - 1
712
713                             If lFrom < 0 Then
714                                 ' From from next loop must be >= 0 therwhiqse no solution, stop the loop
715                                 lTryPreviousFlight = False
716                                 lReturn = ReturnNoSolution
717
718                                 Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " NO solution no more Slot to
check " & _
719                                     lFrom)
720                             Else
721                                 lTryPreviousFlight = True
722
723                                 Call EX_Log(RecurLevel, "Rec - FL: " & lFlightIx_ToMove & " Initial: " & lTargetIx & _
724                                         " reloop Next target: " & lFrom)
725
726                             End If
727
728                         End If
729                     End If
730                 End If
731             End If
732     Wend
733
734     Call EX_Log(RecurLevel, "Rec EndLevel : " & lFlightIx_ToMove & " return : " & lReturn)
735
736     AU_NPM_ManageMarginPrioFlights_MoveFlightEarlier = lReturn
737
738     RecurLevel = RecurLevel - 1
739 End Function
740
741
742 '------------------------------------------------------------ AU_NPM_ManageMarginPrioFlights_AssignPrioOnlyFlight
743 ' Assign priority only flights in the remaining slot
744 ' list containt prio only + baseline + suspended
745
746 Sub AU_NPM_ManageMarginPrioFlights_AssignOtherFlights(lAll_AUFlights As CL_AllFlights, _
747             ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer, _
748             ByRef lPrioFlightSortedFl() As Integer, lPrioFlight_nb As Integer)
749
750     'don't manage the suspended flight here
```

```vb

    Dim lFl As Integer
    Dim lFlIx As Integer

    Dim lBaselineFlights() As Integer
    Dim lBaselineFlightsNb As Integer

    Dim lPrioOnlyFlights() As Integer
    Dim lPrioOnlyFlightsNb As Integer

    'Dim lSuspendedFlights() As Integer
    'Dim lSuspendedFlightsNb As Integer


    ReDim lBaselineFlights(lPrioFlight_nb)
    lBaselineFlightsNb = 0

    ReDim lPrioOnlyFlights(lPrioFlight_nb)
    lPrioOnlyFlightsNb = 0

    'ReDim lSuspendedFlights(lPrioFlight_nb)
    'lSuspendedFlightsNb = 0

    For lFl = 0 To lPrioFlight_nb - 1
        lFlIx = lPrioFlightSortedFl(lFl)
        If lAll_AUFlights.GetPrio(lFlIx) = GPrioBaseline Then
            lBaselineFlights(lBaselineFlightsNb) = lFlIx
            lBaselineFlightsNb = lBaselineFlightsNb + 1
        ElseIf lAll_AUFlights.GetPrio(lFlIx) = GPrioSuspended Then
            'lSuspendedFlights(lSuspendedFlightsNb) = lFlIx
            'lSuspendedFlightsNb = lSuspendedFlightsNb + 1
        Else
            lPrioOnlyFlights(lPrioOnlyFlightsNb) = lFlIx
            lPrioOnlyFlightsNb = lPrioOnlyFlightsNb + 1
        End If
    Next lFl


    ' ---------- manage the baseline flights on schedule
    If lBaselineFlightsNb > 0 Then
        Call AU_NPM_ManageMarginPrioFlights_AssignBaselineFlights(lAll_AUFlights, _
                lSlotTime(), lSlotList(), lSlot_nb, _
                lBaselineFlights(), lBaselineFlightsNb)
    End If
    Erase lBaselineFlights


    ' ---------- Manage the prio flights
    If lPrioOnlyFlightsNb > 0 Then
        Call AU_NPM_ManageMarginPrioFlights_AssignPrioFlights(lAll_AUFlights, _
            lSlotTime(), lSlotList(), lSlot_nb, _
            lPrioOnlyFlights(), lPrioOnlyFlightsNb)
    End If
    Erase lPrioOnlyFlights


    '' ---------- Manage the Suspended flights
    'If lSuspendedFlightsNb > 0 Then
    '    Call AU_NPM_ManageMarginPrioFlights_AssignSuspendedFlights(lAll_AUFlights, _
    '        lSlotTime(), lSlotList(), lSlot_nb, _
    '        lSuspendedFlights(), lSuspendedFlightsNb)
    'End If
    'Erase lSuspendedFlights


End Sub


'------------------------------------------------------------- AU_NPM_ManageMarginPrioFlights_AssignBaselineFlights
' Assign default Baseline flights in the remaining slot
' PB what we do if no possible slot for baseline ??????

Sub AU_NPM_ManageMarginPrioFlights_AssignBaselineFlights(lAll_AUFlights As CL_AllFlights, _
            ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer, _
            ByRef lFlightSorted() As Integer, lFlight_nb As Integer)

    Dim lFl As Integer
    Dim lSlotAssigned As Integer
    Dim lEarliestTime As Date
    Dim lBaselineTime As Date
    Dim lFlAssigned As Integer
    Dim lFlIx As Integer

    ' ---------- manage the baseline flights on schedule
    lFlAssigned = 0
    ' loop on baseline flights
    For lFl = 0 To lFlight_nb - 1
        lFlIx = lFlightSorted(lFl)
        ' idem part then for Margins
        ' find a slot corresponding to the Margin value to put the flight
        lBaselineTime = lAll_AUFlights.GetBaselineTime(lFlIx)

        lSlotAssigned = AU_NPM_ManageMarginPrioFlights_ManageTimeSolution(lAll_AUFlights, _
                    lSlotTime(), lSlotList(), lSlot_nb, _
                    lBaselineTime, lFlIx)

        If lSlotAssigned < 0 Then
            Call AU_NPM_MsgboxStop("Baseline flight PB of NB of available slot not OK for : " & _
                    lAll_AUFlights.GetCallsignICAO(lFlIx) & " id= " & lFlIx, _
                    lAll_AUFlights, lSlotTime, lSlotList, lSlot_nb)
        End If
    Next lFl

End Sub



'------------------------------------------------------------- AU_NPM_ManageMarginPrioFlights_AssignPrioOnlyFlight
' Assign priority only flights in the remaining slot

Sub AU_NPM_ManageMarginPrioFlights_AssignPrioFlights(lAll_AUFlights As CL_AllFlights, _
            ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer, _
            ByRef lFlightSorted() As Integer, lFlight_nb As Integer)

    Dim lFl As Integer
    Dim lFlIx As Integer
    Dim lSlotIx As Integer

    Dim lTargetTime As Date
    Dim lSlotAssigned As Integer

    Dim lFlightHaveSolution As Boolean

    ' ---------- Manage the prio flights

    For lFl = 0 To lFlight_nb - 1
        lFlIx = lFlightSorted(lFl)

        ' Manage Prio Flights try to find a free slot compatible with the schedule
        lFlightHaveSolution = False
        For lSlotIx = 0 To lSlot_nb - 1
            If lSlotList(lSlotIx) = -1 Then
                If AU_NPM_ManageMarginPrioFlights_IsFlightScheduleCompatible(lAll_AUFlights, _
                        lSlotTime(lSlotIx), lFlIx) Then
                    ' the schedule is compatible with the slot time
                    ' lTime = lMySlotsValueSorted(lSlotIx)
                    lSlotList(lSlotIx) = lFlIx

                    'Call lAll_AUFlights.SetFDATime(lFlIx, lSlotTime(lSlotIx))
                    lSlotIx = lSlot_nb ' stop the loop
                    lFlightHaveSolution = True
                End If
            End If
        Next lSlotIx


        ' test if no solution found because slot too early
        If lFlightHaveSolution = False Then
            ' no slot available due to schedule time until the end of the slot list


            lTargetTime = lAll_AUFlights.GetHotspotEndTime

            lSlotAssigned = AU_NPM_ManageMarginPrioFlights_ManageTimeSolution(lAll_AUFlights, _
                        lSlotTime(), lSlotList(), lSlot_nb, _
                        lTargetTime, lFlIx)

            If lSlotAssigned < 0 Then
                Call AU_NPM_MsgboxStop("Prio flights PB of NB of available slot not OK for : " & _
                        lAll_AUFlights.GetCallsignICAO(lFlIx) & " id= " & lFlIx, _
                        lAll_AUFlights, lSlotTime, lSlotList, lSlot_nb)
            End If

        End If
    Next lFl
End Sub

Sub AU_NPM_ManageMarginPrioFlights_AssignSuspendedFlights(lAll_AUFlights As CL_AllFlights, _
            ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer, _
            ByRef lPrioFlightSortedFl() As Integer, lPrioFlight_nb As Integer)

    Dim lFl As Integer
    Dim lFlIx As Integer
    Dim ltime As Date

    ' get the suspended flights
    For lFl = 0 To lPrioFlight_nb - 1
        lFlIx = lPrioFlightSortedFl(lFl)
        If lFlIx > -1 Then ' normally never
            If lAll_AUFlights.GetPrio(lFlIx) = GPrioSuspended Then
                ltime = lAll_AUFlights.GetHotspotEndTime - G_OneSec_AsDate
                Call lAll_AUFlights.SetFDATime(lFlIx, ltime)
            End If
        End If
    Next lFl
End Sub



'------------------------------------------------------------- AU_NPM_ManageMarginPrioFlights_AssignPrioOnlyFlight
' Assign priority only flights in the remaining slot
```

```vb

Sub AU_NPMOLD_ManageMarginPrioFlights_AssignSuspendedFlights(lAll_AUFlights As CL_AllFlights, _
            ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer, _
            ByRef lFlightSorted() As Integer, lFlight_nb As Integer)

    Dim lFl As Integer
    Dim lFlIx As Integer
    Dim ltime As Date

    ' ---------- manage the baseline flights on schedule
    ' ---------- manage the suspended flights
    For lFl = 0 To lFlight_nb - 1
        lFlIx = lFlightSorted(lFl)
        If lFlIx <> -1 Then
            ' suspended flights at the end of the hotspot
            ' Dont use a slot in the middle
            ' the slot will be use when conpacting at the end

            ltime = lAll_AUFlights.GetHotspotEndTime - G_OneSec_AsDate

            Call lAll_AUFlights.SetFDATime(lFlIx, ltime)
        End If
    Next lFl

End Sub


'------------------------------------------------------ AU_NPM_ManageMarginPrioFlights_UseAvailableSlots
' Assign priority only flights in the remaining slot

Sub AU_NPM_ManageMarginPrioFlights_UseAvailableSlots(lAll_AUFlights As CL_AllFlights, _
            ByRef lSlotTime() As Date, ByRef lSlotList() As Integer, lSlot_nb As Integer)

    Dim lFl As Integer
    Dim lFlIx As Integer

    Dim lFlChg As Integer
    Dim lFlChgIx As Integer

    Dim lEarliestTime As Date

    ' compact the list
    For lFl = 0 To lSlot_nb - 1
        lFlIx = lSlotList(lFl)
        If lFlIx = -1 Then
            ' there is a hole
            ' find a flight to put here
            For lFlChg = lFl + 1 To lSlot_nb - 1
                lFlChgIx = lSlotList(lFlChg)
                If lFlChgIx > -1 Then
                    ' there is a flight here
                    If AU_NPM_ManageMarginPrioFlights_IsFlightScheduleCompatible(lAll_AUFlights, _
                        lSlotTime(lFl), lFlChgIx) Then

                    'lEarliestTime = lAll_AUFlights.GetBaselineTime(lFlChgIx)
                    'If lSlotTime(lFl) >= lEarliestTime Then
                        ' use this flight to fill the hole
                        'Call lAll_AUFlights.SetFDATime(lFlChgIx, lSlotTime(lFl))
                        lSlotList(lFl) = lFlChgIx
                        lSlotList(lFlChg) = -1 ' free the slot
                        lFlChg = lSlot_nb ' stop the loop
                    End If
                End If
            Next lFlChg
        End If
    Next lFl

End Sub


```