1. ATMS/ETMS Software Requirements

Name:	Bob Sharick
Date/Version:	03/26~~0~~/2003
Feature Described:	Slot Credit Substitution
ETMS Version:	1.7~~6~~
Remarks:	
Reviewers/CSC:	Ken Howard, Tom Ciolino, Melia Stefanescu
Reviewers/Volpe:	Rick Oiesen
Reviewers/FAA:	

Revision History

Main Changes for Version 1.7

1. Added a requirement to allow the FAA to disable SCS for all airports and all current and future GDPs until they decide to reactivate it. This will be done by adding a parameter. Changing the parameter to disable/enable SCS will require a restart of EDCT.
2. Added a requirement to disallow an SCS request if the slot time of the SCS flight falls within the requested window.
3. Added a requirement to write both SCS and BRG flights to the EDCT_LOG files.
4. Added a requirement that when bridging times out for an airline, EDCT must send a notification to ADL_DD.

Main Changes for Version 1.6

1. Modified requirement 8.4. When a program is revised or compressed, bridging must be turned on for all airlines at that airport.

Main Changes for Version 1.5

1. Modified format of SCS BRIDGING OFF command.
2. Added a requirement for EDCT to send to ADL_DD a message when an airline turns bridging on or off.
3. Corrected the format of the GA and military bridging off display in the ADL file.
4. Added a requirement for EDCT to send to ADL_DD a message that bridging must be turned on for all airlines when a GDP is revised at an airport.
5. Added a requirement to specify the node name in the SCS ON/OFF commands submitted through TSD.

1. Overview

2. Functional Decomposition

EDCT Requirements

1. EDCT will accept SCS messages as part of Simplified Substitution (SS) packets.

   1.1. Only users authorized to send SS packets as defined in the airline definitions file can send SCS messages. All checks applied to the messages in an SS packet will be applied to the SCS messages.

   1.2. Multiple SCS messages can be submitted in an SS packet.

   1.3. FM and FX messages can be submitted with SCS messages in an SS packet.

   1.4. SCS messages must have the following fields:

      1.4.1. Message type – SCS

      1.4.2. Flight id – This is the flight the user would like to substitute into a new slot. It may be the same flight that owns the yielded slot.

      1.4.3. Departure airport

      1.4.4. Arrival airport

      1.4.5. IGTD

      1.4.6. Yielded slot

      1.4.7. Earliest acceptable slot time

      1.4.8. Latest acceptable slot time

2. EDCT will define the following parameters:

   2.1. Substitution window size – 20 minutes.

   2.2. Minimum notification time – 30 minutes.

   2.3. Minimum move up time – 10 minutes.

   2.4. Maximum move up time – 30 minutes.

   2.5. Bridge time-out – 30 minutes.

   2.6. Bridging off permanently for all general aviation flights.

   2.7. Bridging off permanently for all military flights.

   2.8. SCS processing enabled/disabled (SCS ON/OFF). The default is SCS ON.

3. FDB will define the following parameters:

   3.1. Taxi time – 15 minutes.

   3.2. SCS minimum notification time – 30 minutes.

4. The airline definitions file will contain the following information:

   4.1. Bridging off permanently for specific airlines.

4.1.1. The file will contain a new keyword, BRG_OFF_PERM, which will have as data the word YES. This is the only valid data for the keyword. Any airline that has this keyword will have bridging turned off at all airports at all times. Bridging can not be turned on for any such airline.

5. EDCT will process the SCS messages as follows:

5.1. EDCT will check each SCS message for the following errors and return the specified error message to the airline. If any errors are detected on any SCS messages then the entire SS packet must be rejected. This is in addition to the current checking of FM and FX messages.

5.1.1. Slot credit substitution must be enabled. Error message: SCS PROCESSING HAS BEEN DISABLED UNTIL FURTHER NOTICE.

~~5.1.1.~~5.1.2. **Slot credit substitution must be allowed at the arrival airport. Error message: SCS PROCESSING IS OFF AT ARRIVAL AIRPORT.**

~~5.1.2.~~5.1.3. **Substitution must be turned on. Error message: SUBS OFF.**

~~5.1.3.~~5.1.4. **The yielded slot must be owned by a flight that the user is authorized to sub for. Error message: NOT AUTHORIZED TO SUB FOR THIS FLIGHT.**

~~5.1.4.~~5.1.5. **The yielded slot must not be owned by a former pop-up. Error message: YIELDED SLOT OWNED BY FORMER POP-UP FLIGHT.**

~~5.1.5.~~5.1.6. **The yielded slot must be in the future. Error message: YIELDED SLOT MUST BE IN THE FUTURE.**

~~5.1.6.~~5.1.7. **The earliest acceptable time must be greater than the yielded slot time. Error message: EARLIEST ACCEPTABLE TIME MUST BE LATER THAN TIME OF YIELDED SLOT.**

~~5.1.7.~~5.1.8. **The earliest acceptable time must be less than or equal to the latest acceptable time. Error message: EARLIEST ACCEPTABLE TIME MUST BE EARLIER THAN LATEST ACCEPTABLE TIME.**

~~5.1.8.~~5.1.9. **The time ranges of different SCS messages within a packet must not overlap. The time range of an SCS is defined as the time from the yielded slot to the latest acceptable time. Error message: SCS TIME RANGES OVERLAP.**

~~5.1.9.~~5.1.10. **The substitution window size for an SCS packet must be set to 0. If any flights within the packet violate the window size, then the packet must be rejected. Error message: 20 MINUTE WINDOW NOT ALLOWED WITH SCS.**

5.1.11. The slot owned by an SCS flight must not fall within the requested window. Error message: SLOT TIME OF SCS FLIGHT IS ALREADY WITHIN WINDOW.

5.2. EDCT will create a list of bridging candidates with the following criteria.

5.2.1. The flight is controlled; it has an arrival slot.

5.2.2. The flight is not a pop-up (control type of FA) and it is not ground stopped (control type of GS).

5.2.3. The flight does not have a status of cancelled, active, or completed.

5.2.4. The flight has a slot time later than the yielded slot and no later than the latest acceptable time.

5.2.5. The flight is not already delayed past its CTD: ETD <= CTD.

5.2.6. The flight's current ETD is later than its minimum notification time.

5.2.7.

5.2.8. The flight is not subbed by another message in the SS packet.

5.2.9. The flight belongs to a user that has bridging turned on.

5.2.9.1. Bridging may be turned on/off for a major airline, in which case bridging will be on/off for the major and all of its subcarriers as defined in the airline definitions file including those split by flight-id range.

5.2.9.2. It may also be turned on/off for an airline that is not a major, in which case only the flights for that airline would be affected.

5.3. The list of bridging candidates will have the following information.

5.3.1. The current slot time for the flight.

5.3.2. The name of the slot.

5.3.3. The earliest ETA that the flight can use, defined as:

5.3.3.1. If flight has an ERTA, set EETA to ERTA.

5.3.3.2. Else if flight has an LRTA, set EETA to LRTA.

5.3.3.3. Else if flight has an LGTA, set EETA to LGTA – taxitime (parameter set to 10 minutes).

5.3.3.4. Else if flight has an IGTA, set EETA to IGTA – taxitime.

5.3.3.5. Otherwise set EETA to ETA.

5.3.3.6. EETA = max (EETA, (current time + (ETA – ETD) + notify_time))

5.3.4. A flag for whether the flight is used for bridging.

5.4. The list of candidates will have the SCS flight as the first element. The elements in the list will be sorted in ascending order of slot time.

5.5. The goal of the SCS algorithm is to move the yielded slot to the earliest desirable slot time of the SCS request by iteratively swapping the yielded slot with other slots in the list of candidate flights. Flights that are moved up during this process are called bridge flights. The bridge flights are selected in order to achieve the most desirable move-up times, according to the precedence defined in the system requirements. See section 7 for details of the algorithm.

5.6. If no bridge can be found, return an error to the user. Reject the entire SS packet and send an error message to the requesting airline: SLOT CREDIT SUBSTITUTION CAN NOT BE FULFILLED. Do not update any databases.

6. If all SCS requests (and any and all SS requests) in a packet are satisfied, then EDCT must update its database and send transactions to FDB.

6.1. For the SCS flight:

6.1.1. Assign the new slot to the flight.

6.1.2. Set the CTA equal to the slot time.

6.1.3. Set the CTD equal to the CTA minus the ETE, where ETE equals the ETA minus the ETD.

6.1.4. Set the control type to SCS.

6.2. For a bridging flight:

6.2.1. Assign the new slot to the flight.

6.2.2. Set the CTA equal to the slot time.

6.2.3. Set the CTD equal to the CTA minus the ETE.

6.2.4. Set the control type to BRG.

6.3. EDCT must send transactions to FDB with this new information.

6.4. EDCT must write to the EDCT_LOG file an entry for each SCS and BRG flight. These entries should be in slot list format.

7. Bridge and Sub fulfillment messages.

7.1. When an SCS request is fulfilled, EDCT will send a reply back to the requesting airline as follows:

7.1.1. SCS REQUEST SUCCESSFUL

7.1.2. A slot list with each flight changed by the packet:

7.1.2.1. Aircraft ID

7.1.2.2. Assigned arrival slot

7.1.2.3. Departure airport

7.1.2.4. Controlled time of departure

7.1.2.5. Controlled time of arrival

7.1.2.6. Control type

7.1.2.7. Exempt flag

7.1.2.8. Cancel flag

7.1.2.9. Slot hold flag

7.1.2.10. Earliest runway time of arrival

7.1.2.11. Initial gate time of departure

7.2. For each flight used as a bridge, EDCT will send a message to the address identified in the airline definitions file with the following contents:

7.2.1. SCS BRIDGING UPDATE FOR AIRPORT

7.2.2. A slot list with all of the flight modified by the packet with the same information as in requirement 5.1.2.

8. Messages controlling SCS and Bridging

8.1. The FAA will have the ability to turn Slot Credit Substitution on or off at a specific airport or for all airports. These commands must be accepted by TSD, FSM, netmail, etc. and passed to EDCT. They will be sent to EDCT with the message type of NWA_ASD_TO_EDCT.

8.1.1. Authorization checking for this command must be node specific. The same authorization that applies to flight removal will apply to the sending of SCS messages. That is, nodes that are allowed to remove flights will be allowed to send SCS messages and nodes that are not allowed to remove flights will not be allowed to send SCS messages.

8.1.2. EDCT SCS OFF [AIRPORT/ALL]

8.1.2.1. EDCT must then send the following message to all connected airlines (ARINC and CDMNET): SCS HAS BEEN TURNED OFF FOR AIRPORT or SCS HAS BEEN TURNED OFF FOR ALL AIRPORTS.

8.1.2.2. If SCS is already off at the requested airport, then EDCT will return the following message to the requestor: SCS already off at [airport].

8.1.2.3. EDCT must send a message to all adl_dd programs for each controlled airport mentioned in the message.

8.1.2.3.1. Type – NWA_CDM_TO_DD_SCSOFF

8.1.2.3.2. Contents – airport name.

8.1.3. EDCT SCS ON [AIRPORT/ALL]

8.1.3.1. EDCT must then send the following message to all connected airlines (ARINC and CDMNET): SCS HAS BEEN TURN ON FOR AIRPORT or SCS HAS BEEN TURNED ON FOR ALL AIRPORTS.

8.1.3.2. If SCS is already on at the requested airport, then EDCT will return the following message to the requestor: SCS already on at [airport].

8.1.3.3. EDCT must send a message to all adl_dd programs for each controlled airport mentioned in the message.

8.1.3.3.1. Type - NWA_CDM_TO_DD_SCSON.

8.1.3.3.2. Contents – airport name.

8.1.4. When the FAA revises or compresses a GDP, EDCT will set bridging to ON.

8.1.4.1. EDCT must then send a message to all adl_dd programs for each airport mentioned in the message.

8.1.4.1.1. Type NWA_CDM_TO_DD_BRGON.

8.1.4.1.2. Contents – airport name airline name.

8.1.5. If bridging for an airline has been turned off by definition in the airline definitions file, then EDCT must ignore any subsequent messages that turn bridging on for that airline, such as through an EDCT SUB OFF.

8.1.6. After reading the parameter file for any bridging off data, EDCT must send a message to all adl_dd programs.

8.1.6.1. Type – NWA_CDM_TO_DD_BRGOFF_PERM.

8.1.6.2. Contents –GA or military. These must be listed together in one space delimited string.

8.2. Each airline that is authorized to send SS packets may turn bridging off for its flights and for the flights of any of its subcarriers, which may be split by flight-id ranges, as defined in the airline definitions file.

8.2.1. The format of the message will be EDCT BRIDGING ON AIRPORT and EDCT BRIDGING OFF AIRPORT.

8.2.2. If bridging has been turned off in the airline definitions file for the airline, then EDCT must reject any message from the airline to turn bringing on.

8.2.3. EDCT must record the time a bridging OFF message was received and if a bridging ON message is not received within the bridging time-out limit, then EDCT must turn bridging ON for that airline at that airport. EDCT must also send a message to the airline to inform it that bridging has been turned on: SCS BRIDGING OFF AT AIRPORT HAS TIMED OUT. BRIDGING IS NOW ON. EDCT must reset the time to the time-out limit if another bridging OFF message is received from the airline. EDCT should check for airlines that exceed the bridging time-out limit every 5 minutes.

8.2.4. EDCT must send a message to ADL_DD for each receipt of a bridging on/off request and it must send a message to ADL_DD when bridging times out for an airline.

8.2.4.1. NWA_CDM_TO_DD_BRGON – the data portion of this message will contain the airport name and the airline name for which bridging is now ON.

8.2.4.2. NWA_CDM_TO_DD_BRGOFF – the data portion of this message will contain the airport name and the airline name for which bridging is now OFF.

8.3. When a program is purged at an airport, slot credit substitution will revert to ON at that airport, as will bridging.

8.4. When a program is revised or compressed at an airport, slot credit substitution will revert to ON at that airport, as will bridging.

8.4.1. EDCT must send to ADL_DD a message of type NWA_CDM_TO_DD_BRGON_ALL. The data portion of this message will contain the airport name for which bridging is now ON.

ADL requirements

ADL_DD must display the current state of slot credit substitution and bridging in the adl file for each controlled airport.

9. ADL_DD must store the state of SCS for each airport. It will receive messages from EDCT that will toggle the state between ON and OFF. The initial value will be ON. ADL_DD will return an acknowledgment for each message to EDCT.

9.1. NWA_CDM_TO_DD_SCSON – the data portion of this message will contain the airport name at which SCS is now ON.

9.2. NWA_CDM_TO_DD_SCSOFF – the data portion of this message will contain the airport name at which SCS is now OFF.

9.3. When generating an ADL for an airport, ADL_DD must include the current state of SCS in the START_SUB_FLAG data block: SCS ON/OFF.

10. ADL_DD must store a list of airlines for which bridging is OFF at each controlled airport. It will receive messages from EDCT that will toggle the state between ON and OFF. The initial value will be ON so that the list will be empty. ADL_DD will return an acknowledgment for each message to EDCT.

10.1. NWA_CDM_TO_DD_BRGON – the data portion of this message will contain the airport name and the airline name for which bridging is now ON. ADL_DD must search the list of airlines at this airport for which bridging was OFF for this airline and remove it from the list.

10.2. NWA_CDM_TO_DD_BRGOFF – the data portion of this message will contain the airport name and the airline name for which bridging is now OFF. ADL_DD must add this airline to the list of airlines at this airport for which bridging is OFF. When generating an ADL file, ADL_DD must list all of the airlines on the list in the START_SUB_BLOCK on a separate line: BRIDGING OFF airline.

10.3. NWA_CDM_TO_DD_BRGOFF_PERM – the data portion of this message will contain an ”GA”, or “military”. ADL_DD must display whether bridging is off for all General Aviation flights or for all military flights.

10.3.1. If the contents is “GA”, then ADL_DD will display BRIDGING OFF GA in the START_SUB_BLOCK of all monitored airports.

10.3.2. If the contents is “military”, then ADL_DD will display BRIDGING OFF MILITARY in the START_SUB_BLOCK of all monitored airports.

10.3.3. If ADL_DD receives this message when no airports are monitored, it must retain these settings so that they can be applied later.

10.4. An airline may have bridging turned off permanently. When ADL_DD reads the airline definitions file, if it finds the keyword BRG_OFF_PERM with the data value of YES, then it must add this airline to the list of airlines for which bridging is off and write BRIDGING OFF airline in the START_SUB_BLOCK of all monitored airports.

11. If ADL_DD receives an NWA_CDM_TO_DD_PUR message, it must set SCS to ON for that airport and it must remove all airlines from the bridging off list for that airport. Any permanently OFF airlines must stay OFF.

12. If ADL_DD receives an NWA_CDM_TO_DD_BRGON_ALL message, it must set SCS to ON for that airport and it must remove all airlines from the bridging off list for that airport. Any permanently OFF airlines must stay OFF.

13. ADL_DD must write the contents of the bridging off airline list into the aar_gdp data file for each airport. On startup, the program will read these files and recreate the list of airlines for which bridging is off at each airport.

14. ADL_DD must write the contents of the permanently bridging off data into a new file named permanent_bridging_off in the data directory. On startup, the program must read this file to recreate the list of airlines for which bridging is permanently off. This may also include GA or military.

FDB requirements

There are two new control types: SCS and BRG.

15. FDB must store the new control types of SCS and BRG.

15.1. When FDB receives a transaction from EDCT with a control type of SCS or BRG, it should treat them the same as GDP control.

15.2. FDB must display the new control types in fdb dumps, in replies to list requests, and in any other output.

16. FDB must calculate the EETA value.

16.1. FDB must use the algorithm defined in 3.3.3 to calculate the EETA for a flight.

16.2. FDB must send the EETA to EDCT in every transaction. The record passed to EDCT must be expanded to include the EETA.

16.3. If the value of EETA changes, then FDB must send the new value to EDCT. FDB should use a new transaction type, EETA_UPDATE, for this transaction.

3. External Systems

TSD Requirements

17. TSD must add the following items to the EDCT Commands menu:

17.1. EDCT SCS ON – TSD must then display a dialogue box to prompt for an airport name or the word ALL. A directory name including the node name must be attached to the end of the request (eg., //cdmk04/reports). EDCT must know the node name in order to perform authorization checking on the request.

17.2. EDCT SCS OFF – TSD must then display a dialogue box to prompt for an airport name or the word ALL. A directory name including the node name must be attached to the end of the request (eg., //cdmk04/reports). EDCT must know the node name in order to perform authorization checking on the request.

17.3. If the user sends the command, then TSD must send the contents of the prompt to EDCT with the message type of NWA_ASD_TO_EDCT.

4. Databases

None.

5. Data Interfaces

None.

6. User Interfaces

None.

7. Implementation Issues/Suggestions

Bridging Algorithm

Given:

A list of flights where the first flight is the flight in the slot being yielded (call this the SCS flight) and the rest of the flights are the candidates for creating the bridge.

Define:

nFlights = number of flights in the candidate list, including the SCS flight
flight = flight data structure, composed of fields:
    EETA = earliest possible ETA for this flight
    slot_time = time of the current slot owned by this flight
desired_move = configuration parameter defining the optimal time to move a bridging flight (30 minutes)
min_move = configuration parameter defining the minimum time to move a flight
T1 = earliest acceptable slot time

Note: The list of candidate flights only includes flights whose slot times are earlier than the latest acceptable time. This prevents the algorithm from moving the SCS flight past the latest acceptable time.

Logic:

```
cur_flight = 1  ;Index of the SCS flight – the flight we want to move down.

;Iteratively look for bridges until either the yielded slot is moved to the desired time range or we can't
;create a bridge.
Do
    best_flight = 0        ;index of the best bridging candidate

    ;Find the next bridging flight. Loop from the current location of the SCS flight to the end of
    ;the candidate list, if necessary.
    For f = cur_flight + 1 TO nFlights

        ;If we have moved past desired_move, and we already have a bridging flight that exceedsthe
        ;min_move, that is the best we can do, so let's bail out of the loop.
        If best_flight <> 0 AND
        flight(f).slot_time – flight(cur_flight).slot_time > desired move AND
        flight(best_flight).slot_time – flight(cur_flight).slot_time >= min_move then
            exit the for loop
        ;Otherwise we keep looking. See if this is a flight we can even use as a bridge
        Else if flight(f).EETA <=  flight (cur_flight).slot_time then,
            ;If this is the first possibility, just save it.
            If best_flight = 0 then
                best_flight = f

            ;If this flight is min_move from T1, it must be better than what we already have
            Else if (T1 – flight(f).slot_time) > min_move then
                best_flight = f

            ;If this flight is past T1, we are done. Save it as the bridge and get out of the loop.
            Else if (flight(f).slot_time >= T1 then
                best_flight = f
                exit for
            End if
        End if
    ;If we haven’t exited the loop, there might still be a better choice so keep looking.
    loop for next flight

    ;Now we either fell out the loop without finding a bridge, or we have the best candidate.

    If best_flight = 0, then
        exit the do loop      ;we have failed to find a flight, so the SCS cannot be performed.
    Else,
        swap the best flight and the current flight in the candidate list
        set cur_flight = best flight
    End if

;If we haven’t gotten to T1 yet, go look for another bridging flight.
Continue do loop until flight(cur_flight).slot_time >= T1

;Again, we have either failed and fallen out of the do loop, or we are done.
If best_flight = 0
    SCS has failed. Send back the error message.

Else
    SCS has succeeded. Commit changes and send out responses.
End if
```

If no best_flight has been found, then SCS has failed and an error message must be returned to the airline.