UC1160  (SCHED.28de)

Version Delivery Scheduling Information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Wave | Phase/Product Version | Bundle | Iteration | Notes |
|  |  |  |  |  |

Approvals

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Approved By | Signed | Date |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Author:

Document version:

Document Date:

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 2012-Jul-11 | 1.0a | Initial draft | Andrey Golovachev |
| 2012-Sep-14 | 1.0b | Revised description of DCS use scenarios after discussion with Glenn Rapson | Andrey Golovachev |
| 2013-Nov-21 | 1.0c | Added sections 7.1 and section 16.3 for capturing history and assisting to display history records of cabin configurations. | Sriram Ananthasivan |
| 2014-Jul-23 | 1.1 | Added section 16.4.2.2.3 to cabin configuration history story for exporting history records as CSV | Sriram Ananthasivan |
| 2016-Oct-03 | 1.2 | Explicitly stated that SITA Admin can manage airline-specific cabin configurations on behalf of subscribers.  Managing Global default cabin configs is now a separate Basic flow 5 (no changes to functionality – just becomes a separate screen on GUI). | Andrey Golovachev |
| 2016-Oct-13 | 1.3 | Added capturing of subscriber code in history. | Andrey Golovachev |

Contents

1. Brief Description 7

1.1 Background 7

1.2 References 8

2. Actors 8

3. General Preconditions 8

4. Basic Flow of Events 8

4.1 Basic Flow 1 – Create Airline-Specific Cabin Configuration 8

4.1.1 Specific Preconditions 8

4.1.2 Steps 8

4.1.3 Specific Post Conditions 9

4.2 Basic Flow 2 – Extract Airline-Specific Cabin Configuration 9

4.2.1 Specific Preconditions 9

4.2.2 Steps 9

4.2.3 Specific Post Conditions 9

4.3 Basic Flow 3 – Update Airline-Specific Cabin Configuration 9

4.3.1 Specific Preconditions 9

4.3.2 Steps 9

4.3.3 Specific Post Conditions 10

4.4 Basic Flow 4 – Delete Airline-Specific Cabin Configuration 10

4.4.1 Specific Preconditions 10

4.4.2 Steps 10

4.4.3 Specific Post Conditions 10

4.5 Basic Flow 5 – Create/Update Global Default Cabin Configuration 11

4.5.1 Specific Preconditions 11

4.5.2 Steps 11

4.5.3 Specific Post Conditions 11

5. Alternate Flows 11

6. Exception Flows 11

6.1 Exception Flow 1 – Validation Error 11

6.1.1 Specific Preconditions 11

6.1.2 Steps 12

6.1.3 Specific Post Conditions 12

6.2 Exception Flow 2 – No Records Found on Match 12

6.2.1 Specific Preconditions 12

6.2.2 Steps 12

6.2.3 Specific Post Conditions 12

6.3 Exception Flow 3 – Deletion Error 12

6.3.1 Specific Preconditions 12

6.3.2 Steps 12

6.3.3 Specific Post Conditions 13

7. Sub Flows 13

7.1 Sub Flow 1 – Create History Item 13

7.1.1 Specific Preconditions 13

7.1.2 Steps 13

7.1.3 Specific Post Conditions 13

8. General Post Conditions 13

9. Extension Points 13

10. Special Requirements 13

10.1 Navigation Requirements 13

10.2 Usability Requirements 13

11. Additional Information 14

12. Changes to Reviewed Use Cases 14

13. Future Use Case Considerations 14

14. Assumptions & Issues 14

15. Design Constraints 14

16. Use Case Elaboration 15

16.1 SCHED.28d Cabin Configurations CRUDE Service 15

16.1.1 Detail 15

16.1.2 Acceptance Tests 15

16.1.3 Non Functional Requirements 17

16.2 SCHED.28d1 Enable SITA Admin to Create Cabin Configurations on Behalf of Subscribers 18

16.2.1 Overview 18

16.2.2 Detail 18

16.2.3 Acceptance Tests 18

16.3 SCHED.28e Cabin Configurations GUI 19

16.3.1 Detail 19

16.3.2 Acceptance Tests 19

16.3.3 Non Functional Requirements 21

16.4 HIST.01.CBND: Cabin Configuration (Description) history 21

16.4.1 Scope 21

16.4.2 Details 21

16.4.3 Acceptance test 25

16.5 SCHED.28f Add DOW and ACV to SCT conditions 27

16.5.1 Summary 27

16.5.2 Detail 27

16.5.3 Acceptance Tests 30

# Brief Description

This Use Case describes the process that allows subscriber users and SITA Admin user to display, create and update Cabin Configurations.

## Background

Cabin Configurations is an extension of Schedule Control Table (SCT) functionality which allows definition of sets of cabins that will be used by a specific airline. A cabin can be defined with attributes such as code, description, hierarchy and colour.

SITA Administrator performs a one-off task of creating a global default cabin configuration which serves as a superset for selection of airline-specific carrier configurations. He can also manage airline-specific cabin configurations on behalf of subscribers.

Subscriber Administrator will be allowed to choose cabins he uses from the global default cabin configuration. A subscriber can create his own **view** of cabin configuration for any airline. Subscriber can specify a name for every cabin different to default name. For non-hosted airlines he can also define his view of RBD codes for every cabin.

**Usage of Cabin Configurations with DCS**

DCS processes will use the cabin codes and its associated RBD's only for look up purposes. There is no requirement in DCS to use the “Assign RBD” Schedule Control rules created in IAS.

Possible scenarios are listed below:

1. Cabin and RBD information is often missing for OA carriers in OAG SDS. Hence, during Flight Profile creation the Flights domain will access IAS Cabin Configurations in order to look up Cabin-RBD mapping for non-hosted OA carriers in case this information is missing in IAS Schedules.   
     
   Cabin Configurations will not be used for creation of Flight Profiles for hosted airlines because Cabin-RBD information for hosted airlines is always available in IAS Schedules.  
     
   As a rule, if Cabin-RBD information is present in Schedules for a specific host or OA flight, it should be used, as being more accurate, instead of generic information given by Cabin Configuration.
2. Cabin-RBD information for non-hosted carriers from Cabin Configurations will be used by DCS in case an individual name item on the PNL does not have a Cabin Code/RBD specified. I.e. if passenger to be booked in Economy, then HDCS will create CJ Record with identical Cabin/RBD (Cabin Code = Y and RBD = Y).
3. DCS will access Cabin Configurations for generic information about ALL cabin codes used by a certain **host or OA carrier** (instead of looking through all available Aircraft Configurations). An example scenario would be creating of Baggage Allowances.
4. Carrier-specific cabin “names” from Cabin Configurations will be printed on boarding passes.
5. Generic cabin colors will be used on seat maps for both hosted and OA airlines.
6. There is no requirement for DCS to use cabin hierarchy from Cabin Configurations for hosted carriers. The assumption is that cabin codes in IAS host Schedules are always ordered from upper to lower.  
   However there may be a requirement in future to validate new IAS Aircraft Configurations against Cabin Configurations to prohibit creation of ACs with more than one cabin of same hierarchy (for example cabins C and J would typically be mutually exclusive within the same flight).

## References

|  |  |  |
| --- | --- | --- |
| **Ref. ID** | **Reference** | **Document/ Reference Version** |
| [1] | NGI BDD for Manage Cabin Configurations |  |
| [2] | HIAS UC for Manage IAS History v2.0 | Section 4.1 |
| [3] | HIAS BDD for Manage IAS History v2.0 | Section 2.1 |

# Actors

**Primary Actor:** SITA Administrator, Subscriber Administrator (DCS user)

# General Preconditions

* None

# Basic Flow of Events

## Basic Flow 1 – Create Airline-Specific Cabin Configuration

### Specific Preconditions

* Actor is successfully logged into the system.
* Actor is either subscriber user or SITA Admin authorized to create airline-specific cabin configurations.

### Steps

1. Actor selects to create airline-specific .
2. System responds with data elements to create airline-specific  using  
   **BDD [1] section 2.1** – Create/Update airline-specific Cabin Configuration.
3. Actor provides information for data elements.
4. Actor submits information.
5. System validates information using **BDD [1] section 2.1.**
6. System creates the .
7. System notifies Actor of success.
8. Flow ends

### Specific Post Conditions

* System created the .
* System responded with a successful create message.

## Basic Flow 2 – Extract Airline-Specific

### Specific Preconditions

* Actor is successfully logged into the system.
* Actor is either subscriber user or SITA Admin authorized to extract airline-specific .

### Steps

1. Actor selects to extract airline-specific *.*
2. System responds with data elements for the extract criteria using **BDD [1] section 2.2** – Search Criteria for Airline-specific Cabin Configuration.
3. Actor provides search criteria.
4. Actor submits information.
5. System validates information using **BDD [1] section 2.2**.
6. System selects that matches the extract criteria.
7. System responds with data elements for using **BDD [1] section 2.1** – Create/Update airline-specific Cabin Configuration..
8. Flow ends.

### Specific Post Conditions

* System extracted a list of records.

## Basic Flow 3 – Update Airline-Specific

### Specific Preconditions

* Actor is successfully logged into the system.
* Actor is either subscriber user or SITA Admin authorized to update airline-specific .
* Actor completed Basic Flow 2 – Extract .

**Note**: the user must not be allowed to remove the last remaining Cabin from the cabin configuration. There must be at least one cabin specified.

### Steps

1. Actor updates information for airline-specific data elements.
2. Actor submits information.
3. System validates information using **BDD [1] section 2.1** – Create/Update airline-specific Cabin Configuration.
4. System updates .
5. System notifies Actor of success.
6. Flow ends

### Specific Post Conditions

* System updated for changed data elements.
* System responded with a successful update message.

## Basic Flow 4 – Delete Airline-Specific

Note: this flow is not used in GUI.

### Specific Preconditions

* Actor is successfully logged into the system.
* Actor is either subscriber user or SITA Admin authorized to delete airline-specific .

Actor has completed one of the following steps:

* Step 7 for Basic Flow 2 – Extract .

### Steps

1. Actor selects to delete airline-specific .
2. System prompts Actor to confirm delete.
3. Actor confirms delete.
4. System deletes airline-specific .
5. System notifies Actor of success.
6. Flow ends

### Specific Post Conditions

* System deleted .
* System responded with a successful delete message.

## Basic Flow 5 – Create/Update Global Default Cabin Configuration

### Specific Preconditions

* Actor is successfully logged into the system.
* Actor is SITA Admin authorized to create/update the global default .

**Note**: the user must not be allowed to remove the last remaining Cabin from the cabin configuration. There must be at least one cabin specified.

### Steps

1. Actor selects to manage Cabin Configuration.
2. In case the Global Default Cabin Configuration does not exist, system responds with data elements to create it using **BDD [1] section 2.3** – Create/Update Global Default Cabin Configuration. In case the Global Default Cabin Configuration already exists, system responds with data elements to update it, using the same BDD.
3. Actor provides information for data elements.
4. Actor submits information.
5. System validates information using **BDD [1] section 2.3.**
6. System creates/updates the .
7. System notifies Actor of success.
8. Flow ends

### Specific Post Conditions

* System created/updated the Global Default for changed data elements.
* System responded with a successful update message.

# Alternate Flows

None

# Exception Flows

## Exception Flow 1 – Validation Error

### Specific Preconditions

System has performed one of the following steps:

* Step 5 for Basic Flow 1 – Create Airline-Specific .
* Step 3 for Basic Flow 3 – Update Airline-Specific .
* Step 5 for Basic Flow 5 – Create/Update Global Default Cabin Configuration.

### Steps

1. System detects one or more business rule violations.
2. System retains the information entered.
3. System responds with error messages indicating the data elements for which business rules are violated.
4. Flow ends.

### Specific Post Conditions

* System responded with appropriate error message.
* System remains unchanged for .

## Exception Flow 2 – No Records Found on Match

### Specific Preconditions

System has performed the following step:

* Step 6 for Basic Flow 2 – Extract .

### Steps

1. System detects no matches for
2. System responds with a message indicating not found.
3. Flow ends.

### Specific Post Conditions

* System responded with a message indicating not found.
* System remains unchanged.

## Exception Flow 3 – Deletion Error

### Specific Preconditions

System has performed the following step:

* Step 4 for Basic Flow 5 – Delete .

**Future sprint requirement**: prohibit cabin code deletion or update by SITA Administrator if it is in use in at least one carrier-specific configuration. Currently Global Default serves simply as a template for creation of carrier-specific cabin configurations, there is no dynamic link between them.

Also there is currently no requirement to maintain consistency between Cabin Configurations and cabin codes used in Schedules, this may be a subject of future requirements.

### Steps

1. System detects one or more rule violations while attempting delete activity.
2. System responds with error messages indicating unable to delete .
3. Flow ends.

### Specific Post Conditions

* System responded with error message.
* System remains unchanged for .

# Sub Flows

## Sub Flow 1 – Create History Item

This Sub-Flow is used to create a History Item each time data is changed (Create/Update/Delete) within the system by another flow.

### Specific Preconditions

* Data has been created, changed, or deleted

### Steps

1. System will create a History Item consisting of the original data prior to update and the data reflecting the update.
2. Control is returned to the invoking flow.

### Specific Post Conditions

* A History Item has been created which includes both the original information in the database and the information that has been changed or deleted.

# General Post Conditions

* None

# Extension Points

None

# Special Requirements

None

## Navigation Requirements

None

## Usability Requirements

None

# Additional Information

Example of default cabin configuration as set up by SITA Administrator:   
  
Hierarchy / Cabin Code(s) / Default Cabin Name / Cabin Colour  
1 / F, P / First Class / Red  
2 / J,C / Business Class / Blue  
3 / W / Premium Economy / Orange  
4 / Y,M / Economy / Green  
5 / TBD / TBD / Yellow

1. Subscriber Administrator  
   Separately, a Subscriber now needs to 'maintain' Cabin/RBD mapping per Handled Airline for Non-Res-linked airlines  
     
   They 'choose' a cabin from the SITA default list in one above, and the first RBD is always filled in as the Cabin class character and the default cabin name. They then need to manually enter the remaining RBDs per cabin. They may then choose to 'modify' the Cabin Name, if they wish.  
     
   Cabin / RBD / Name of Cabin  
   F / F,A / Diamond First Class  
   J / J,C,D / Pearl Business Class  
   Y / Y,B,T,R,V,U,S / Brass Economy Class  
     
   The above will be maintained at a Subscriber / Handled airline combination.

# Changes to Reviewed Use Cases

None

# Future Use Case Considerations

None

# Assumptions & Issues

None

# Design Constraints

None

# Use Case Elaboration

# SCHED.28d Cabin Configurations CRUDE Service

### Detail

Not all validation rules described in BDD are enforced by the schema. Both schema and BDD validation rules must be supported within the service implementation.

Please note that Retrieve flow is not described here because it will not be frequently used. It should be developed for to comply with Reference Architecture requirements. For input/output parameters of the Retrieve operation of the webservice please refer to wsdl.

### Acceptance Tests

This story should be demonstrated by sending CRUDE requests to the service and viewing the response via soapUI. Note that tests 1-13 should be performed in context of specific Subscriber User performing CRUDE operations for respective subscriber.

| **Nr.** | **Test** | **Pre-condition** | **Action** | **Post-condition** |
| --- | --- | --- | --- | --- |
|  | Create a record | None | Send Create request with valid input parameters | The system adds new record and responds with success |
|  | Create invalid record or a record with mandatory fields missing | None | Send Create request with invalid input parameter which violates validation rules or with missing mandatory input parameter. Repeat for each parameter. | The system responds with InvalidRequestFault |
|  | Update a record | Record with specific ID exists | Send Update request with specific ID and valid input parameters | The system updates the record and responds with success |
|  | Update a record with invalid details or skip mandatory fields | Record with specific ID exists | Send Update request with specific ID and invalid input parameter, which violates validation rules, or omit a mandatory parameter. Repeat for each parameter. | The system responds with InvalidRequestFault |
|  | Update missing record | Record with specific ID does not exist | Send Update request with specific ID and valid input parameters | The system responds with InvalidRequestFault |
|  | Delete a record | Record with specific ID exists | Send Delete request for a specific record | The system deletes the record and responds with success |
|  | Delete missing record | Record with specific ID does not exist | Send Delete request for a specific record | The system responds with InvalidRequestFault |
|  | Extract specific records | Several records exist | Send extract request with specific request parameters | The system responds with all matching records |
|  | Extract records: no matches | Several records exist | Send extract request with specific request parameters which do not match any of the existing records | The system responds with a blank response |
|  | Cabin code is not unique | None | During create/update the user submits a cabin configuration which contains duplicate cabin codes (when creating a new record or updating an existing one). | The system responds with InvalidRequestFault |
|  | More than one Cabin Configuration | A Cabin Configuration already exists for a certain subscriber/airline | The user attempts to create more than one Cabin Configuration for a combination of subscriber/airline | The system responds with InvalidRequestFault |
|  | Airline-specific cabins missing in global default | None | During create/update the user submits a cabin configuration which contains cabins missing in the global default cabin configuration | The system responds with InvalidRequestFault |
|  | Global Default by SITA Admin only | None | A user other than SITA Administrator attempts to create/update/delete Global Default Cabin Configuration | The system responds with ServiceAuthorisationFault |
|  | Primary RBD missing | None | During create/update the user submits a cabin configuration which contains cabins with RBD specified, but no RBD equal to cabin code among them | The system responds with InvalidRequestFault |
|  | Subscriber attempts to CRUDE records of another subscriber | See tests 1-14 | A Subscriber User attempts to perform any RUDE operation described in tests 1-14 on record(s) of another subscriber; or attempts to Create a record for another subscriber | The system responds with ServiceAuthorisationFault |
|  | Global Default can’t be deleted | A global default cabin configuration exists | A SITA admin attempts to delete global default | The system responds with InvalidRequestFault |
|  | More than one global default | A global default cabin configuration exists | A SITA admin attempts to create another one | The system responds with InvalidRequestFault |

### Non Functional Requirements

The expected maximum reasonable number of Cabin Configurations is 5000 (assuming multiple subscriber-specific views per each airline).

The response time must be within 0.5 seconds (and preferably faster) for each type of CRUD transaction.

# SCHED.28d1 Enable SITA Admin to Create Cabin Configurations on Behalf of Subscribers

### Overview

As per this use case, a Cabin Configuration can be either Default (effectively a template created by SITA Admin) or it can describe cabins of a particular hosted or OA carrier.

Every subscriber can create a Cabin Configuration describing cabins of the subscriber airline itself or any other airline. For example, Portway ground handler subscriber could create Cabin Configuration for S7 handled airline. In the same time, it is quite possible for another ground handler subscriber, for example, Swissport to create his own Cabin Configurations of S7.

However, currently SITA Admin is not able to create Cabin Configurations on behalf of subscriber. This story enables him to do that.

### Detail

GUI Changes

### Acceptance Tests

This story should be demonstrated by sending CRUDE requests to the service and viewing the response via soapUI. Note that tests 1-13 should be performed in context of specific Subscriber User performing CRUDE operations for respective subscriber.

| **Nr.** | **Test** | **Pre-condition** | **Action** | **Post-condition** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# SCHED.28e Cabin Configurations GUI

### Detail

* Where possible, the GUI client must perform all field validation checks as described in the BDD, in addition to server-side validation.
* The Cabin Configuration functionality should be accessible via **Administration/Cabin descriptions** menu item.
* If the currently logged in subscriber previously set up cabin configurations for multiple airlines, then he is able to switch between these carrier-specific cabin configurations using the “Airline code” dropdown (which works as a navigation control). During this switch, the GUI client should query the extract operation of the CRUDE webservice with a combination of subscriber code and airline code.  
  By default, the cabin configuration of the first airline in the list should be displayed.
* The subscriber can click “Create new cabin description” button in order to create his view of a cabin configuration for any airline. However he cannot create more than one cabin configuration per airline.
* If the subscriber has not set up any cabin configurations yet, then he should be presented with a blank “Create new cabin configuration” screen for the first airline in the list.
* When SITA Administrator impersonates as a SITA Administrator, he should only be able to create/modify Global Default cabin configurations. In order to be able to modify a carrier-specific cabin configuration created by a certain subscriber, he must impersonate as this subscriber during log in.
* The requirement to persist Created/Updated username and timestamp was missed in the original story for Cabin Configurations CRUDE webservice. It needs to be addressed in the current story to enable display of respective fields.
* tns:ExtractCabinConfiguration/tns:SearchCriteria/tns:Airline has been made optional to support populating the Airline Code dropdown on the Update GUI
* Due to change of requirements, the cabin Description field is marked as mandatory on GUI, this should be enforced by the client only. To avoid rework this field should remain optional in the server side for now, the schema will be updated in future iterations.

### Acceptance Tests

| **Nr.** | **Test** | **Pre-condition** | **Action** | **Post-condition** |
| --- | --- | --- | --- | --- |
|  | SITA Admin creates default Cabin Configuration | SITA Administrator has logged in | The user selects to manage Cabin Configurations | The user is presented with the Update screen showing one blank line. He is able to add more lines and thus create the Global Default cabin configuration.  Need to demonstrate:   * Cabins are ordered by hierarchy then alphabetically (after saving) * Cabin code must be unique within Cabin Config * Mandatory fields are checked |
|  | Subscriber Administrator creates cabin configuration for a hosted airline | Global Default cabin configurations exist, but no carrier-specific for the current subscriber. Subscriber Administrator has logged in | The user selects to manage Cabin Configurations | The user is presented with a blank table and “Create” button that takes him to Create screen. There he is able to add more lines and thus create carrier-specific cabin configuration.  Need to demonstrate:   * Switching between multiple airlines via dropdown * Description, hierarchy and colour are fetched from default configuration * Description is editable and does not affect Global Defaults * RBDs are editable and mandatory for non-hosted airlines only. * The first RBD is automatically populated by the system, same letter as the cabin code. It is mandatory in the list of RBDs. |
|  | Subscriber Administrator creates cabin configuration for a non-hosted airline | Carrier-specific cabin configurations exist. | The user selects to manage Cabin Configurations | The user is presented with the Update screen for the first airline in the list. He is able to modify cabin configurations.  Need to demonstrate:   * Switching between multiple airlines via dropdown * Using “Create new cabin description” button to create a new cabin configuration. * “Last updated” timestamp is modified upon update. |
|  | Cabin configuration already exists for that airline | Carrier-specific cabin configurations exist. | The user attempts to create a cabin configuration for an airline which already has a cabin configuration for this subscriber. | An error message is displayed. |

### Non Functional Requirements

Subscribers will use GUI for Cabin Configurations infrequently; for SITA Administrator setting up the defaults will most likely be one-off task. Therefore, there are no specific performance requirements for this screen.

However, as per supplementary specification, in 99.5% of cases the screen response time should not exceed 2000ms.

# HIST.01.CBND: Cabin Configuration (Description) history

### Scope

The scope of this story is

* Record the history of cabin configuration upon its creation, update or deletion in the system.
* Prepare a transformation script to transform history records of cabin configuration for the GUI to render the history record (XML).
* Store the transformation script to transformation data store.
* Demonstrate search and viewing of cabin configuration history records in conjunction with Manage IAS History use case.

### Details

#### Capturing History records for cabin configuration

Following are the activities in the life cycle of cabin configuration where history record needs to be captured

1. A new cabin configuration is created.
2. A cabin configuration is updated.
3. A cabin configuration is deleted.

The history record comprises of 2 significant parts

* History index details
* The cabin configuration represented as XML
  + - * + In case of create, the document with the content supplied during its creation.
        + In case of update, the document after the update. In other words, the document with the new content.
        + In case of delete, the document header (having the document identifier) without the content.

History index details are used to specify attributes of the cabin configuration explicitly to make the history record searchable in the history data store. Please refer to HIAS BDD for Manage IAS History (Reference [3]) section 2.1 Create History Record for the complete (generic) set of data elements required for recording history.

For cabin configuration history record, the following data elements must have specific values as detailed in the table below

| **S No** | **Data element name** | **Value** | **Remarks** |
| --- | --- | --- | --- |
| 1 | Airline | Hosted airline code or 'XS' | Host Subscriber's airline code.  Use 'XS' if global default indicator is set. |
| 2 | EventDateTime | Date/Time | The date and time when an activity such as create, update or delete was performed. |
| 3 | UserID | User's Id | The Id of the logged in user |
| 4 | Process | "GUI" | The value without quotes. |
| 5 | Type | "CABIN" | The value without quotes. |
| 6 | Item | "CABIN" | Value without quotes. |
| 7 | Action | "Create" or "Update" or  "Delete" | Any single value (without quotes) from the set that describes the activity. |
| 8 | Record | Cabin Configuration XML | The XML representing the cabin configuration that was created updated or deleted. |
| 9 | RecordVersion | The version of the Cabin Configuration XML | The primary (root) namespace URL of Cabin Configuration XML. |

The story will need to invoke Create History record (flow) in Manage IAS History use case to record the cabin configuration history record.

#### Preparing the transformation script to format cabin configuration history record

This aspect of the story is to prepare the transformation agent script that is capable of transforming the cabin configuration XML (history) into a form suitable for displaying on the GUI.

The transformation agent needs to translate the cabin configuration XML into

1. Full Annotated (Labelled) Form
2. Semi-Annotated Form

#### Full Annotated (Labelled) Form

In the full annotated form, the cabin configuration XML will be formatted as text along with annotations (labels) alongside the values that indicates what a particular value stands for. Example of full annotated form is as below

XS  
CAB:P DESC:First class premium HIER:1 COLOR:Red RBD:P  
CAB:F DESC:FirstClass HIER:2 COLOR:Red RBD:F A I R  
CAB:J DESC:BusinessClassPremium HIER:3 COLOR:Blue RBD:J C I Z O  
CAB:C DESC:BusinessClass HIER:4 COLOR:Blue RBD:M N K L  
CAB:Y DESC:EconomyCoach HIER:5 COLOR:Green 5 RBD:Y M B Q

**EBNF (template):**

|  |  |
| --- | --- |
| Cabin Configuration History format | <header>(new line)<cabin configurations> |
| <header> | <Airline or default>"/"<subscriber> |
| <cabin configurations> | [**CAB:**<cabin code>(space)**DESC:**<cabin description>(space) **HIER:**<hierarchy>(space)**COLOR:**<color>(space)**RBD:**<RBDs>] [1 to 15 lines] |
| <RBDs> | <RBD>[1 to 26 letters separated by (space)] |

#### Semi Annotated (Compact) Form

The semi-annotated form is an optimised variant of the full annotated form. It is expected that a user who regularly views history records will be familiar with values in cabin configuration information such that the user may not need the support of annotations (or labels) to describe the values. With the absence of annotations (or labels), a history record can be displayed in much lesser space as compared to the full annotated form and thus display more history records.  
Example of semi annotated form is as below

XS  
P/First class premium/1/Red/P  
F/FirstClass/2/Red/F A I R  
J/BusinessClassPremium/3/Blue/J C I Z O  
C/BusinessClass/4/Blue/M N K L  
Y/EconomyCoach/5/Green 5/Y M B Q

**EBNF (template):**

|  |  |
| --- | --- |
| Cabin Configuration History format | <header>(new line)<cabin configurations> |
| <header> | <Airline or default>"/"<subscriber> |
| <cabin configurations> | [<cabin code>"/" <cabin description>"/" <hierarchy>"/" <color>"/" <RBDs>] [1 to 15 lines] |
| <RBDs> | <RBD>[1 to 26 letters separated by (space)] |

#### Export Form

#### The export form is used when the user chooses to save the cabin configuration history records that were fetched in the user's computer. In the process of exporting, the history records are transformed into a format described below using a transformation agent such as XSLT.

Example of export form (XSLT output) is as below

AirlineOrDefault,Cabin,Description,Hierarchy,Color,RBDs,Cabin,Description,Hierarchy,Color,RBDs,Cabin,Description,Hierarchy,Color,RBDs  
XS,P,First class premimum,1,Red,P,F,First class,2,Red,F A I R,J,Business Class Premium,3,Blue, J C I Z O

**EBNF (template):**

|  |  |
| --- | --- |
| Cabin Configuration History export format | <Cabin Configuration headers>(new line)<Cabin Configuration record> |
| <Cabin Configuration headers> | AirlineOrDefault,Subscriber <Cabin description headers> |
| <Cabin description headers> | [,Cabin,Description,Hierarchy,Color,RBDs] [ Repeat as many times as the number of cabins] |
| <Cabin Configuration record> | <Airline or default>","<subscriber>","<Cabins description records> |
| <Cabins description records> | [","<cabin code>","<cabin description>","<hierarchy>","<color>","<RBDs>] [1 to 5 records, one for each cabin capacity appended; forming a single line] |
| <RBDs> | <RBD>[1 to 26 letters separated by (space)] |

#### EBNF (Template) fields to XML Schema Mapping

#### To prepare the transformation scripts to output cabin configuration information in both forms, the following table provides the XPath mapping for the EBNF (template) fields. This mapping is valid for Full Annotated Form and Semi-Annotated Form.

|  |  |
| --- | --- |
| EBNF (Template) Field | XPath //element(\*,CabinConfiguration) |
| <Airline or default> | /AirlineOrDefault/Airline **(or)** 'XS' (if global default indicator is set) |
| <subscriber> | //SubscriberReference |
| <cabin code> | /Cabin/CabinCode |
| <cabin description> | /Cabin/Description |
| <hierarchy> | /Cabin/Hierarchy |
| <color> | /Cabin/Color |
| <RBDs> | /Cabin/CabinRBDs/BookingClass |

#### Store the transformation script to transformation data store

#### This aspect of the story is to ensure that the transformation script(s) that was prepared is stored in the transformation data store. In the absence of this, the GUI will not be able to display the cabin configuration history record due to the lack of transformation agent (script). The transformation agent (script) is included in the response to extract history records.

#### Please note that while storing the transformation agent script to the data store, the version(s) supported by the transformation agent is also stored.

### Acceptance test

#### Definition of done requires that all test scenarios are satisfied; furthermore it is expected that the development team will identify additional scenarios to add to the test pack.

| **Sr.** | **Scenario** | **Action** | **Post Condition** |
| --- | --- | --- | --- |
| 1 | Create a cabin configuration resulting in creation of a history record for the cabin configuration. | Subscriber navigates to cabin description screen and creates a cabin configuration for a carrier.  Verify the created history record using Manage IAS History GUI. | The system stores the new cabin description (configuration) and creates a history record for the cabin configuration.  The created history record should be searchable and viewed on Manage IAS History GUI. |
| 2 | Update an existing cabin configuration resulting in the creation of a history record for the change to cabin configuration. | Subscriber navigates to cabin description screen, retrieves an existing cabin configuration, and updates the configuration.  Execute this test case for update of all possible aspects of the cabin configuration.  Verify the created history record using Manage IAS History GUI. | The system updates the cabin description (configuration) and creates a history record for the cabin configuration.  The history record should be created every time the cabin configuration is updated.  The created history record should be searchable and viewed on Manage IAS History GUI. |
| 3 | Delete an existing cabin configuration resulting in the creation of a history record of the cabin configuration. | Subscriber navigates to cabin description screen, retrieves an existing cabin configuration and deletes it.  Verify the created history record using Manage IAS History GUI. | The system deletes the cabin description (configuration) and creates a history record for the deletion of cabin configuration.  The created history record should be searchable and viewed on Manage IAS History GUI. |
| 4 | Using Manage IAS History GUI, search for history records and view the history records created in the above scenarios. | Subscriber uses Manage IAS History GUI and submits a search for history records for type as 'Cabin Description ' (CABIN). | Manage IAS History GUI results screen should display the at least 3 history records of cabin configurations from the previous test case scenarios.  The history detail should show well-formatted cabin configuration content in full-annotated form. The user should be able to switch between full-annotated and semi-annotated forms instantly. The GUI should not perform any service calls pertaining to shifting from one annotated form to another. |
| 5 | Export cabin configuration history records as CSV (file). | Subscriber uses Manage IAS History GUI and submits a search for history records for type as 'Cabin Configuration' (CABIN).  One or more history records of cabin configurations are displayed.  User clicks on 'Export as CSV' button or menu item. | The system prompts the user to save the CSV file.  The CSV file containing all the fetched history records (across all pages) is saved to the user's chosen location.  The CVS file displays correctly in Excel, in particular the columns succeeding the values with embedded commas and double quotes appear in the correct columns. |

# SCHED.28f Add DOW and ACV to SCT conditions

### Summary

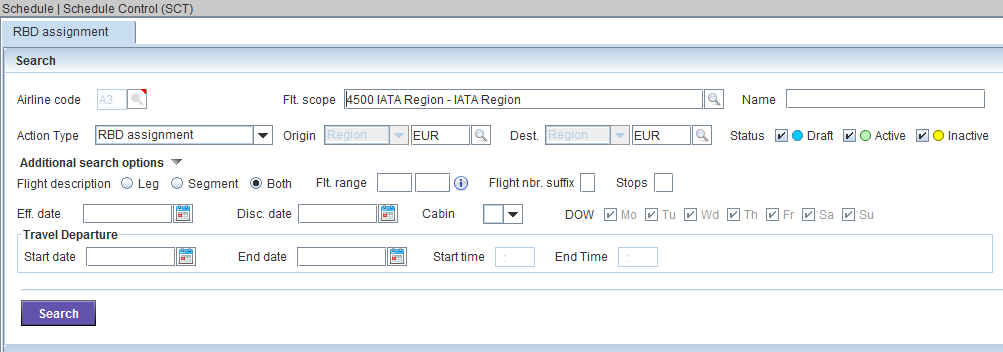
This task addresses certain improvements to the SCT rules GUI to enhance its suitability for users in terms of presentation and interaction. The scope includes the following tasks

1. Change the graphics, layout and text for the following SCT – Schedule Control Screens
   * NEW (conditions section)
   * Search (Additional search options section)
2. Add DOW and ACV fields to the following SCT – Schedule Control Screens
   * NEW (conditions section)
   * Search (Additional search options section)
3. Add an additional element called **AircraftConfiguration of Type AircraftConfigurationIdentifierType** to the XML schema (ScheduleControlsRuleManager)

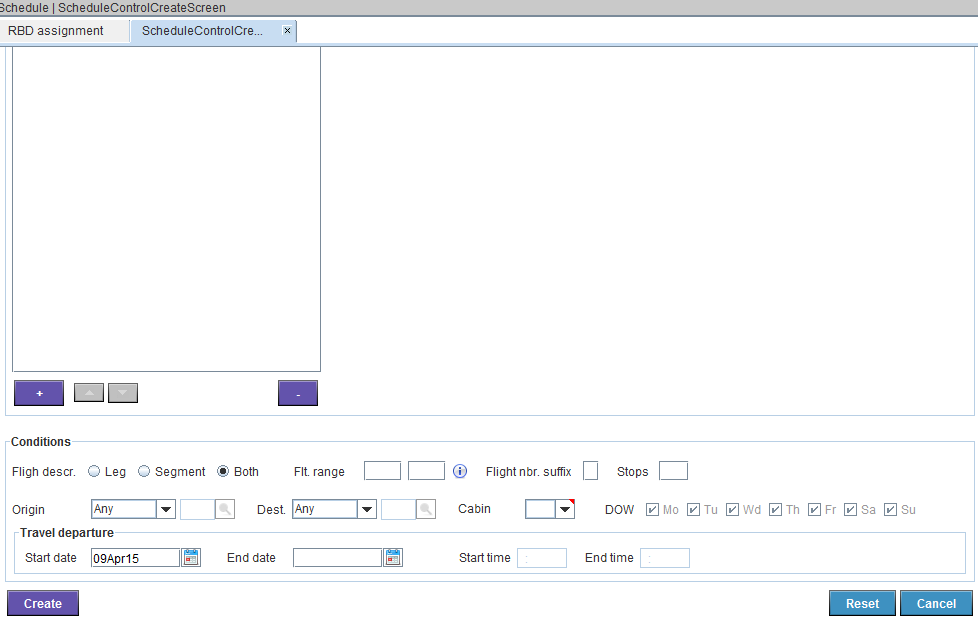
### Detail

**Current GUI Presentation**

**Search Screen (Additional search options section)**



**Create NEW SCT Screen (Conditions section)**

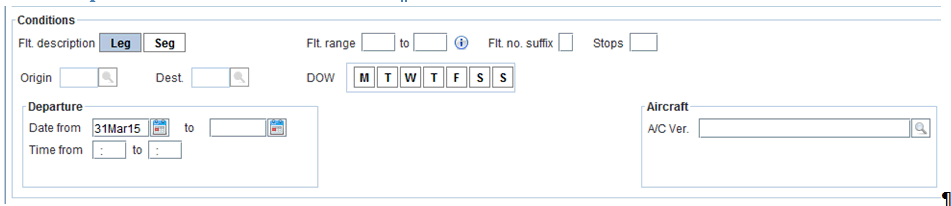


**Expected GUI Presentation**

Expected Layout, graphics, text for the following SCT screens is shown below.

* **Search Screen (Additional search options section)**
* **Create NEW SCT Screen (Conditions section)**

This story involves adding DOW and ACV as conditions on the schedule controls rules.  There are also some items in the GUI clean up list for the schedule conditions panel as well. Both affect the schedule condition panel.  If these two tasks were combined we would expect the schedule condition panel and additional search options to look something like this



It's has the same look and feel as the inventory rule conditions but does not include the arrival, mkt by, and op by conditions.

**SERVICE**

**AircraftConfigurationIdentifierType** is made up of the following elements as shown below

|  |
| --- |
|  |
|  |

**SERVICE CALLS FOR SEARCH, CREATE NEW AND UPDATE SCT RULES**

* 1. **Service**: interface-ScheduleControlsRuleManager
     1. **Interface**: ExtractScheduleControlRule
     2. **Operation**: ExtractScheduleControlRule (**search**)
     3. **Input**: ExtractScheduleControlRule
     4. **Element**: /SearchCriteria/SearchConditions/**AircraftConfiguration**

1. **Interface**: ManageScheduleControlRule
2. **Operation**:CreateScheduleControlRule (**Create new**)
3. **Input**: CreateScheduleControlRule
4. **Element**: /ScheduleControlRule/Conditions/**AircraftConfiguration**
5. **Interface**: ManageScheduleControlRule
6. **Operation**:UpdateScheduleControlRule (**update**)
7. **Input**: UpdateScheduleControlRule
8. **Element**: /ScheduleControlRule/Conditions/**AircraftConfiguration**

### Acceptance Tests

| **Sr.** | **Test** | **Pre-condition** | **Action** | **Post-condition** |
| --- | --- | --- | --- | --- |
| **1** | SCT GUI changes for Full HIAS subscribers | User logged into the system | User navigates to various SCT control screens  SEARCH  CREATE NEW SCT  UPDATE | Presentation of the following screens (including menu items, controls, text fields etc)  Search Screen (Additional search options section)  Create NEW SCT Screen (Conditions section)  As described in **DETAIL Expected GUI Presentation (section 16.5.2)** |