NGI UC for Activate Inventory

INV.29, INV.28

Version Delivery Scheduling Information

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

Approvals

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Approved By | Signed | Date |
| 1.0 | Tim Ruberg |  | 2012-Nov-21 |
| 1.0 | Bill Tanksley |  | 2012-Nov-21 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Author:

Document version:

Document Date:

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 2012-June-04 | 0.0 | Initial Draft | Bernie Leapaldt |
| 2012-June-22 | 0.0a | Remove Draft ICR references | Bernie Leapaldt |
| 2012-June-26 | 0.0a | Name change – was Activate & Undo Draft ICR's | Bernie Leapaldt |
| 2012-Oct-24 | 0.0b | Revised as per the Schedule Activation orchestration  Renamed from UC for Manage ICR Activation into UC for Activate Inventory. | Nguyen Van Binh |
| 2012-Nov-09 | 0.0c | Update to process inventory activation by flight index date. Added reference to BDD for Inventory Activation | Nguyen Van Binh |
| 2012-Nov-14 | 0.0d | Removed activation for OA flights and additional updates | Nguyen Van Binh |
| 2012-Nov-20 | 0.0e | Updated from 19Nov12 peer review | Nguyen Van Binh |
| 2012-Nov-21 | 1.0 | Approved Version | Bill Tanksley |
| 2012-Dec-20 | 2.0 | Major elaborations | Andrey Golovachev |
| 2013-Feb-04 | 2.1 | Changes in the Basic Flow 1 to account for new architecture described in CHANGE.03b story.  Added elaborations of INV.29c and INV.29d stories | Andrey Golovachev |
| 2013-Feb-21 | 2.2 | Added story description of INV.29e | Andrey Golovachev |
| 2013-Apr-25 | 2.3 | Modified INV.29d story description: do not protect ICRs which entirely consist of “unchanged” legs/segments. | Andrey Golovachev |
| 2013-Sep-12 | 3.0 | Added INV.28 (Basic flow 2) – Advance Inventory Detail Windows | Andrey Golovachev |
| 2013-Oct-09 | 3.1 | Added story description INV.29a-1: modified step 1 of the Basic flow 1. | Andrey Golovachev |
| 2014-Feb-21 | 3.2 | Minor changes to integrate Inventory Window Advancement with Nightly Inventory Maintenance | Andrey Golovachev |
| 2014-Mar-31 | 3.3 | Added INV.29a2 – Flight Dressing Rework & INV.29a3 – Flight Dressing for Groups & Waitlist | Krisna Pawan |
| 2014-May-27 | 3.4 | Added INV.29i - Retain Manual/RMS/Bid Price control indicators | Krisna Pawan |
| 2014-Oct-17 | 3.5 | Added INV.29j – Propagate schedule to grid and testing story | Adedapo Bayode |
| 2015-Feb-14 | 3.5 | Updated acceptance test for INV.29j | Krisna Pawan |
| 2015-Nov-20 | 3.6 | Added comments & test to INV.29a2 (sec.12.8.4) for scenario when RBDs are different (Defect-25970) | Krisna Pawan |
| 2016-Oct-06 | 3.7 | QC 30831: Added Step 12 to Basic Flow 1 and extended step 5 of Basic Flow 2 – now marking as Inactive BLs to be excluded due to duplicate POS | Andrey Golovachev |

Contents

1. Brief Description 7

1.1 Terminology 7

1.2 Story Mapping 9

1.3 Assumptions 9

1.3.1 Inventory Activation 9

1.3.2 Inventory Retention 10

1.3.3 Miscellaneous 10

1.4 References 10

2. Actors 11

3. General Preconditions 11

4. Basic Flow of Events 11

4.1 Basic Flow 1 – Activate Inventory for Planned Schedule Change 11

4.1.1 Specific Preconditions 11

4.1.2 Steps 12

4.1.3 Specific Post Conditions 12

4.2 Basic Flow 2 – Advance Inventory Detail Windows (INV.28) 13

4.2.1 Specific Preconditions 13

4.2.2 Steps 13

4.2.3 Specific Post Conditions 14

5. Alternate Flows 15

5.1 Alternate Flow 1 – ICR cannot be deleted 15

5.1.1 Specific Preconditions 15

5.1.2 Steps 15

5.1.3 Specific Post Conditions 15

5.2 Alternate Flow 2 – Unexpected new ICR 15

5.2.1 Specific Preconditions 15

5.2.2 Steps 15

5.2.3 Specific Post Conditions 15

6. Exception Flows 16

6.1 Exception Flow 1 – Inventory Activation ICR Error 16

6.1.1 Specific Preconditions 16

6.1.2 Steps 16

6.1.3 Specific Post Conditions 16

6.2 Exception Flow 2 – Inventory Activation Leg/Segment Error 16

6.2.1 Specific Preconditions 16

6.2.2 Steps 16

6.2.3 Specific Post Conditions 17

7. Sub Flows 17

8. General Post Conditions 17

9. Extension Points 17

10. Additional Information 17

11. Future Use Case Considerations 17

11.1 Documentation updates required 18

12. Use Case Elaboration 19

12.1 INV.29a - Activate inventory: Schedule Change interface and Retention 19

12.1.1 Overview 19

12.1.2 Implementation Details 19

12.1.3 Acceptance Tests 20

12.1.4 Non Functional Requirements 21

12.2 INV.29a-1 - Apply Inventory\_Detail\_Date Parameter during Schedule Change 22

12.2.1 Overview 22

12.2.2 Acceptance Tests 22

12.3 INV.29b - Activate inventory: apply extra MICT actions 25

12.3.1 Overview 25

12.3.2 Acceptance Tests 25

12.3.3 Non Functional Requirements 25

12.4 INV.29c - Propagate Inventory Activation Information 26

12.4.1 Overview 26

12.4.2 Implementation Details 26

12.4.3 Acceptance Tests 28

12.5 INV.29d - Apply Inventory Activation Information 29

12.5.1 Overview 29

12.5.2 Acceptance Tests 30

12.6 INV.29e - Notify Schedule Change on Inventory and Schedules Activation 31

12.6.1 Overview 31

12.6.2 Acceptance Tests 32

12.7 INV.28 - Advance Inventory Detail Windows 33

12.7.1 Overview 33

12.7.2 Implementation details 33

12.7.3 Acceptance Tests 34

12.8 INV.29a-2 - Flight Dressing – Rework 39

12.8.1 Overview 39

12.8.2 Rework Details 39

12.8.3 Additional examples for assigning segment inventory capacity 40

12.8.4 Acceptance Tests 41

12.9 INV.29a-3 - Flight Dressing - Groups and Waitlist 42

12.9.1 Overview 42

12.9.2 Details 42

12.9.3 Acceptance Tests 42

12.9.4 Non Functional Requirements 44

12.10 INV.29i - Retain Manual/ RMS/ Bid Price indicator controls 44

12.10.1 Overview 44

12.10.2 Details 44

12.10.3 Acceptance Tests 44

12.10.4 Non Functional Requirements 45

12.11 INV.29j – Process Host ASM in Gigaspaces 46

12.11.1 Summary 46

12.11.2 Overview 46

12.11.3 Scope 46

12.11.4 Acceptance Tests 47

# Brief Description

This UC describes the process on the Inventory side to:

* Create, Update or Cancel an ICR and populate inventory controls for its legs and segments as a result of Schedule Change Activation. This is covered by Basic Flow 1 which is invoked by UC for Schedule Change Orchestration [1]
* Create new active ICR for the flight dates moving into the Inventory Creation Window and delete ICR’s moving out of the Inventory Preservation Window as the time goes by. This is covered by Basic Flow 2 which is invoked by UC for Daily Maintenance [2].

## Terminology

This UC uses the following terms:

|  |  |
| --- | --- |
| **Term** | **Definition** |
| **Inventory Retention** | Inventory retention is transferring of some of the inventory controls (e.g. BAU, WAU, etc.) of the affected flight onto the alternate flight. The purpose of it is to preserve control levels that may have been historically established for a particular flight by RMS or manually by inventory analysts. When controls are retained they are prorated based on the inventory capacity of the affected and alternate flights.  Please note that Seat Sold Counts are NOT copied during retention. Instead they will be received from SnS once the Rebooking is over. |
| **Retention Source** | An existing affected leg or segment from which inventory controls are retained. This term was introduced to account for the fact that the same segment may be selected more than once as an alternative in the Schedule Change Plan, whereas inventory controls can be retained onto it from only one affected segment which is the retention source. |
| **Retention Target** | An alternative leg/segment onto which inventory controls are retained from the Retention Source. |
| **Inventory Preservation Window** | The time period within which the existing inventory records are preserved. It typically starts 7 days[[1]](#footnote-1) before the current date at the host’s home timezone[[2]](#footnote-2) (inclusive) and lasts into infinity. The reason for this window is that inventory for flown flights needs to be stored for some time in order to support Passenger Final Sales process with DCS.  Inventory preservation window is applied against the local departure date of the last leg of a flight. Every ICR having departure date of its last leg less than the lower boundary of the Inventory Preservation Window must be deleted. |
| **Inventory Creation Window** | The time period within which an inventory record must exist for every operational date of every active flight of a hosted carrier. It starts on today’s date at the host’s home time zone and typically lasts for 338 days[[3]](#footnote-3). The window is inclusive.  Some inventory records may be created on demand up to 1827 days[[4]](#footnote-4) into the future in order to support out-of-range sales. |

## Story Mapping

The use case breaks into the following stories:

|  |  |  |
| --- | --- | --- |
| Tag | Description | Flows |
| INV.29a | Activate inventory: Schedule Change interface and Retention | Every other part of UC for Activate Inventory, BDD for Activate inventory and UC for Dress Flight (Except for: Basic Flow 2 of Activate Inventory, steps 4-10 of UC for Dress flight) |
| INV.29a-1 | Apply Inventory\_Detail\_Date parameter during Schedule Change | Steps 1 and 2 of the Basic Flow 1, Exception Flow 1 |
| INV.29a-2 | Flight Dressing - Rework | Basic Flow 1 & UC for Dress Flight |
| INV.29a-3 | Flight Dressing – Groups & Waitlist | Same as in INV.29a-2 and includes Groups & Waitlist calculation |
| INV.29b | Activate inventory: Apply extra MICT actions | Steps 4-10 of UC for Dress flight (invoked at step 8 of UC for Activate Inventory) |
| INV.29c | Maintain Inventory Activation Information | None, as this is implementation of an auxiliary CRUDE webservice. |
| INV.29d | Apply Inventory Activation Information | Steps 2 and 13 of UC for Dress Flight |
| INV.29e | Inventory activation notification | Step 13 of the Basic Flow 1. |
| INV.28 | Advance Inventory Detail Windows | Basic Flow 2, Alternate Flows 1 and 2 |
| INV.29i | Retain Manual/RMS control indicators | Basic Flow 1 & UC for Dress Flight |
| INV.29j | Propogate Adhoc schedules to grid | All |

## Assumptions

The following assumptions and considerations have determined the internal interface between the Schedule Change and Inventory domains.

### Inventory Activation

* Any affected leg/segment needs to be re-dressed except those where the changes to schedules were too minor to affect Inventory (for example, meal code change or e-ticket indicator).
* Once inventory controls of a cancelled leg/segment have been retained onto a new one, it should not be processed anymore even if there are more occurrences of it in the Schedule Change Plan. This is in order not to have the retained controls overwritten by redressing.
* FROM-TO pairs with “Flight number change” type of change (where the flight number is the only thing that changes) can be treated in exactly the same manner as a pair of cancelled affected segment and new alternate one.

### Inventory Retention

* Inventory Retention is only performed in the following scenarios (provided that the alternative leg/segment falls into retention window defined by Inventory Penalties):
  + Cancelled leg/segment (or a segment prohibited with Traffic Restriction A) onto a new leg/segment (whether it is a new leg/segment of an existing flight or a leg/segment of a new flight).
  + Changed leg/segment onto itself (for example in scenario of equipment change).
* The requirement is not to perform retention when there is a difference in “Operated by” fields of the affected and alternate segments. This will be enforced by setting Inventory Retention indicator in the Schedule Change Plan (CHANGE.01h story).
* A leg/segment can only be used once for Retention whether it is on the FROM side or on the TO side. This assumption and the ones above are enforced during generation of Inventory From-To when Inventory Retention Indicator is set (by CHANGE.01h story).
* Retaining controls of cancelled legs/segments onto new ones is performed while dressing the new ones and not when processing the ones to be cancelled. This is to enable future performance optimizations (as some steps of dressing the Retention Target cabins could be skipped), and to avoid having to create new segments in the same time with cancelling the existing ones.
* Retention cannot completely replace dressing as Raiding controls, Booking Limit Bucket controls and ISSR levels are never retained.
* Cancelled legs/segments are soft-deleted (and purged later) to allow inventory controls to be retained from them.

### Miscellaneous

* The requirement is to have segments with Traffic Restriction A dressed. Flagging them as restricted in order to exclude them from availability and sell responses may be a subject of a future story.

## References

|  |  |  |
| --- | --- | --- |
| **Ref. ID** | **Document/ Reference Version** | **Notes** |
|  | UC for Schedule Change Orchestration | The Use Case that invokes Activate Inventory for a Flight/Date instance |
|  | UC for Inventory Daily Maintenance | Use Case that invokes Activate Inventory when a date falls into the Open For Sale period. |
|  | UC for Dress Flights | Use Case that creates Inventory Control Record (ICR) and populates inventory controls in the ICR |
|  | BDD for Inventory Activation | Describes structure of input for and results of the inventory activation process |
|  | System BDD for Inventory Control Record (ICR) | Describes the structure of Inventory Control Records (ICR's) |
|  | UC for Inventory Adjustment | Describes the processes to calculate Cabin, Tree, Nested Bucket Seat Available and Effective Controls. It also records Inventory History. |
|  | UC for Assign Booking Limits | Describes the process to determine Booking Limit bucket Seat Available (SA's) for assigned Booking Limit Template(s) in a flight instance. |
|  | BDD for Inventory Maintenance | Describes processing statistics for inventory maintenance tasks |
|  | HIAS Business Parameters spreadsheet <https://glsvn01p.atlis.sita.aero/svn/voyager/Documentation/SIAM/branches/Cross%20Project%20Coordination/Parameters/HIAS%20Business%20Parameters.xlsx> | Describes SIAM parameters specific to HIAS |
|  | UC for Monitor HIAS | Describes a mechanism, which allows the HIAS GUI user to monitor long-running tasks such as Nightly Inventory Maintenance or Mass Re-Dress. |
|  | UC for Manage Inventory | Story INV.14j - Describes the changes made to ICR service to handle PLs (AUs) for nested buckets as per [6] |

# Actors

None (this use-case does not interact with any external systems or users).

# General Preconditions

* This UC has been initiated by an include action from one of the following:
  + UC for Schedules Change Orchestration
  + UC for Inventory Daily Maintenance Process

# Basic Flow of Events

## Basic Flow 1 – Activate Inventory for Planned Schedule Change

### Specific Preconditions

* This UC has been invoked by UC for Schedule Change Orchestration
* The Planned Schedule corresponding to the ICR being activated has been activated.

### Steps

1. System selects the next flight date (index date) among the affected segments of the current flight number taking into account schedule period start/end, frequency and days of week.  
   For new schedules, the dates to activate inventory for are additionally limited by Inventory Creation Window (see its definition in 1.1). This window does not limit inventory activation for schedule replacements or cancellations: the system should search for any existing ICRs beyond the window which could have been created by out-of-range sell.

It is suggested that inventory should be activated by flight dates so that all legs/segments belonging to the same ICR are activated together.  
Also, as a reminder, UC for Schedule Change Orchestration invokes activation of schedules and inventory for groups of intended DRAFT schedule periods with the same flight number

1. System attempts to locate the ICR being activated and performs the following checks based on the type of schedule changes of legs/segments belonging to the ICR:
   * For new schedules ICRs must not exist.
   * For replacements and cancellations, ICRs are expected to exist within the Inventory Creation Window.
2. If all legs/segments are new, the system initiates blank ICR using the basic details from schedule (airline, flight number, and index date) and continues at step 5 in order to add legs and segments to it.
3. If all legs/segments are to be cancelled, the system soft-deletes the ICR, records the cancellation of the flight in history using **Sub Flow 13 of UC [6]** and continues at step 12**.**
4. System selects the next leg or segment of the ICR.
5. System checks if the leg/segment actually does exist in the ICR being activated.
6. If the current leg/segment has changes in schedules too minor to affect inventory, the system continues at step 10.
7. If the current leg/segment is cancelled in schedules, the system soft-deletes the leg/segment from the ICR and continues at step 10.
8. If the current leg/segment is created or changed in schedules, the system invokes **UC for Dress Flight Basic Flow 1**.
9. If there are more legs or segments listed in the request the system rejoins step 5.
10. For Evo subscribers, system evaluates the number of SITA RES subclasses in accordance with section 10.1.2 of UC for Evo Schedule Change Orchestration, flagging the ICRs that violated the limit and marking the Booking Limits to exclude during subsequent publication.

For Evo subscribers, system detects any non-excluded Booking Limits which would result in duplicate POS Limits in accordance with the rules defined in section 2.1.8 of HIAS DEFD for Inventory Handoff, marking such BLs for subsequent exclusion during publication and flagging the ICR in case such BLs are present.

Rule: the following message is appended to the schedule change task details in case there are any BLs excluded due to duplicate POS: *'Duplicate POS have been assigned to the following flights: <list of flight numbers>'*

1. System sets the status of the current ICR to Protected. Note: this step is skipped if none of the ICR’s legs and segments has changes significant to affect inventory.
2. If there are more flight dates to process, the system rejoins step 1.
3. System returns the response to Schedule Change Orchestration indicating the intended DRAFT schedule periods which have been activated.
4. This flow ends and processing resumes in **UC for Schedule Change Orchestration**.

### Specific Post Conditions

* System has Created, Updated or Cancelled ICRs. This may include dressing and retaining inventory controls for new or changed segments.
* System has protected the ICR.
* System has returned the response to Schedule Change Orchestration.
* Inventory history reflects the ICR updates (future story).

## Basic Flow 2 – Advance Inventory Detail Windows (INV.28)

This flow describes the automated process of creating new active inventory records (ICRs) for the future flight dates moving into Inventory Creation Window as the time goes by. It also covers deletion of ICRs which have already moved outside of Inventory Preservation Window. Please see section 1.1 for terminology.

The diagram below illustrates the process of advancement of the inventory detail windows:

**XXXX(**\_\_\_\_\_\_\_\_\_\_{**TODAY**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**++++**}

* “TODAY” represents the current date in the host’s timezone, when this flow has been invoked
* \_Underscores\_ represent the existing Inventory Records which need to be preserved
* {Braces} represent the current **Inventory Creation Window**. It will be filled with ICRs once this flow has been completed
* **+**Green plus signs**+** represent new inventory records which need to be created. Typically that would be for one day only but can be more in case inventory window advancement task was not triggered on the previous night for some reason
* Parenthesis “(“ demarks the lower boundary of the current **Inventory Preservation Window**
* **X**Red crosses**X** represent the old inventory records that need to be deleted. Typically that would be for one day only

Advancement of inventory details window is one of the Nightly Inventory Maintenance activities executed automatically on the nightly basis.

### Specific Preconditions

* This flow has been invoked by the UC for Nightly Inventory Maintenance.

### Steps

1. System determines the current Inventory Preservation Window which is common for all flights of the carrier. The window starts on the **current date** **at the host’s timezone** (as defined by *Carrier\_Local\_Time\_Zone* SIAM parameter) **minus** the number of days specified in *Departed\_Flight\_Date* parameter **plus** *Offset[[5]](#footnote-5)* from the current task details (see **[8]** section 2.1) and lasts into infinity.
2. System determines the current Inventory Creation Window which starts on the current date at the host’s timezone **plus** *Offset*. The window ends on the current date **plus** the number of days specified in *Inventory\_Detail\_Date* SIAM parameter minus one (since today is included into the number) **plus** *Offset*.
3. System proceeds to the first or the next scheduled flight marketed by the current carrier.
4. System calculates a temporary list of the current flight’s index dates to be covered with inventory. This list contains local dates on which the flight operates (based on the scheduled period, frequency and days of week) falling into the current Inventory Creation Window. Only Active schedules (both Standard and Ad-Hoc) are taken into consideration.
5. For every date on the list without an existing ICR the system creates dressed ICR in Active status:

* Initiates blank ICR using the basic details from schedule (airline, flight number, and index date).
* Invokes **[3] Basic Flow 1** for every leg and every segment of the flight being dressed. Dressing must be based on Effective Schedules (Ad-Hoc Schedules) for those dates where they exist. Also, creation of new ICRs is atomic at the ICR level, i.e. if dressing of a leg or a segment fails then the entire ICR must not be created and is counted as failed in statistics.
* UC for Dressing will in turn invoke UC for Inventory Adjust which will calculate Effective Inventory Controls and Seats Available. The calculation will be based on zero Seat Sold Counts (as no seats have been sold yet). UC for Inventory Adjust will also trigger creation of a History record.

**Note**: The decision whether to create an ICR is based on the flight index date falling into the Inventory Creation Window. This means that for some multi-leg flights the local departure dates of downstream legs may extend slightly beyond the Inventory Creation Window.

* For Evo subscribers, system evaluates the number of SITA RES subclasses in accordance with section 10.1 of UC for Inventory Re-Dress, flagging the ICRs that violated the limit and marking the Booking Limits to exclude during subsequent publication.
* For Evo subscribers, system detects any non-excluded Booking Limits which would result in duplicate POS Limits in accordance with the rules defined in section 2.1.8 of HIAS DEFD for Inventory Handoff, marking such BLs for subsequent exclusion during publication and flagging the ICR in case such BLs are present.

Rule: the following message is appended to the nightly maintenance task details in case there are any BLs excluded due to duplicate POS: *'Duplicate POS have been assigned to the following flights: <list of flight numbers>'*

1. System deletes any existing ICRs of the current flight where **departure date of the last leg** (index date adjusted by date variation of the last leg) is **earlier** than lowest boundary of the Inventory Preservation Window. **Sub Flow 13 of UC [6]** is used to record Inventory History.
2. If there are more flights to process for the current carrier the system continues at step 3.
3. System updates detailed and summary statistics of the parent Nightly Inventory Maintenance task in Monitor HIAS using **UC [10] Basic Flow 1 and BDD** **[8] sections 3.2 and 3.3.1**:
   * Flights/Legs/Segments created
   * Flights/Legs/Segments deleted
   * Flights/Legs/Segments failed (due to any dressing errors). If at least one leg or a segment has failed then all legs and segments of the flight are counted as failed.
4. This flow ends and processing resumes in the invoking use-case.

### Specific Post Conditions

* Every flight operation of the current carrier within the current Inventory Creation Window has a corresponding ICR
* For the current carrier there are no ICRs outside of the Inventory Preservation Window
* History reflects all creations and deletions of ICRs that occurred during this flow

# Alternate Flows

## Alternate Flow 1 – ICR cannot be deleted

### Specific Preconditions

* System is performing step 6 of Basic Flow 2.

### Steps

1. System detects that an ICR for a past flight outside of the Inventory Preservation Window is under Emergency Lock.
2. System increases the “Flights Blocked” statistics counter by 1.
3. System logs the fact that a flight under Emergency Lock could not be deleted.
4. System continues processing other flights at step 6 of Basic Flow 2.

### Specific Post Conditions

* System has skipped the ICR and has updated statistics indicating that the flight could not be deleted.

## Alternate Flow 2 – Unexpected new ICR

### Specific Preconditions

* System has performed step 5 of Basic Flow 2 for a certain ICR and has created the ICR.
* The created ICR has its index date earlier than the current date at the host’s timezone plus the number of days specified in the *Open\_For\_Sale\_Date* (subscriber-specific SIAM parameter, typically 331 day). This may indicate that inventory and schedules are significantly out of sync which may be due to errors in inventory activation during schedule change.

### Steps

1. System logs an error with Error Manager the fact that an ICR has been created within the Open for Sale window.
2. This flow ends and processing resumes in the invoking flow.

### Specific Post Conditions

* The ICR is created.
* The system has logged that an ICR has been created within the Open for Sale window.

# Exception Flows

## Exception Flow 1 – Inventory Activation ICR Error

### Specific Preconditions

System has performed step 2 of Basic Flow 1.

### Steps

1. System detects one or more of the following conditions:
   * An ICR already exists for a new schedule.
   * An ICR does not exist within the Inventory Creation Window for schedule replacements and cancellations.
2. If ICR for a new flight already exists, the system logs an error with Error Manager (including error code, flight designator, index date and the attempted action), abandons inventory activation and resumes processing in UC for Schedule Change Orchestration.
3. If ICR for the schedule being replaced does not exist within the Inventory Creation Window, the system logs a warning and continues at step 3 of Basic Flow 1 as if it was a new flight to be created.
4. If ICR for the schedule being cancelled does not exist within the Inventory Creation Window, the system logs a warning and continues at step 12 of Basic Flow 1.

### Specific Post Conditions

* System has recorded an error or warning in Error Manager.
* An ICR is created if it did not exist within the Inventory Creation Window for the schedule being replaced.

## Exception Flow 2 – Inventory Activation Leg/Segment Error

### Specific Preconditions

System has performed one of the following steps

* Steps 5-9 of Basic Flow 1.
* Any step of Basic flow 1 of the included UC for Dress Flight

### Steps

1. System detects one or more of the following conditions:
   * The leg/segment to be created already exists within the activated ICR  
     (Error code: *SegAlreadyExists*)
   * The leg/segment to be cancelled or changed does not exist in the activated ICR.  
     (Error code: *SegDoesNotExist*)
   * UC for Assign Nesting Template invoked by UC for Dress Flight returns an error.  
     (Error code: *NestingError*)
   * UC for Dress Flight returns an error of any other origin.  
     (Error code: *MiscError*)
2. System logs an error message with Error Manager (including error code, details of the activated ICR, leg/segment details and the requested actions).
3. System records the Segment Error Code to return it later to Schedule Change Orchestration and rejoins step 9 of Basic Flow 1.

### Specific Post Conditions

* System has recorded the error in Error Manager.
* Segment Error Code is recorded to return later it to Schedule Change Orchestration as a part of Inventory Activation response for the ICR.
* Leg/segment is not updated.
* Processing continues with the next leg/segment of the activated ICR.

# Sub Flows

None

# General Post Conditions

None

# Extension Points

This UC is an include UC for

* Schedule Change Orchestration.
* Daily Inventory Maintenance Process for Maintaining the "Open for Sale Period".

# Additional Information

None

# Future Use Case Considerations

* Future Use Cases shall consider scenarios where Seat Sold Count (SSC) can be retained for the ICR of an existing flight during the Activate Inventory process so as to minimize the volume of SSC Refresh during schedule change processing.
* The approach to processing of activation errors has yet to be defined. On one hand, a Schedule Change Plan can contain a lot of internal dependencies (like rebooking of passengers from a cancelled flight onto a changed one), and in theory even a single error may ruin it. On the other hand it is not possible to rollback Schedule Change due to massive amount of changes made.
* It may be necessary to rollback changes made on leg/seg level in case of an error.
* When deleting old inventory in Basic Flow 2 it may be necessary to make up closeout data for those flights which have not been closed yet (for example, by assuming that all bookings have boarded).
* It may be necessary to detect and log all occurrences of inventory records without corresponding schedules. Potentially such situations may occur, for example, when a schedule was cancelled but the corresponding inventory was not removed. They are not dangerous in the essence that inventory cannot be sold without schedules anyway but may indicate coding errors.  
  Currently it is not clear which of the inventory maintenance tasks should take responsibility of “repairing” inventory. Also, this is closely related to holistic handling of errors during schedule change which is still an open issue.
* If INV.28 is implemented prior to implementing inventory activation for ad-hoc schedules then a separate story will be required to **test** how ad-hoc schedules are accounted for during inventory window preservation.
* While dressing schedules during the Schedule Change, the system must only dress flights within the Inventory Creation Window. However, this rule would not allow creation of ad-hoc flights departing today from the airports having UTC offset significantly less than the carrier’s home.  
  For example, suppose the carrier is based in SYD and currently it is 21OCT in SYD. In HNL it may still be 20OCT and today’s departure from HNL may fall outside of the dressing window. Therefore, specifically for dressing ad-hoc schedules the window should be extended 1 day into the past (provided the ad-hoc flight’s time was not in the past at the departure station at the moment of schedule creation and dressing).

## Documentation updates required

* **NGI BDD for Inventory From – To Record**: To add the Change Indicator "Number of Stops" for Segment (Refer to BDD for Inventory From – To Record, Group: Change Indicators)
* **UC for Generate Inventory From – To Record**: To amend such that All Host Operated and Host Marketed – OA Operated flights have an Inventory From - To record created.

# Use Case Elaboration

## INV.29a - Activate inventory: Schedule Change interface and Retention

### Overview

This story brings together already developed pieces of the dressing process and adds application of the remaining MICT actions.

The scope of INV.29 includes:

* Invoking the existing Assign Nesting Template functionality on leg/segment basis and assigning default inventory controls based on the nesting template (existing functionality first developed during Stage 4 within INV.07).
* Applying extra MICT actions which may override the default controls: Inventory Capacity Adjustment, Cabin Controls, Tree Top Controls, Nested Bucket Controls, Booking Limit Controls. CRUDE for these actions was developed in INV.04F story.
* Performing Inventory Retention.
* Deleting the ICRs and ICR legs/segments to be cancelled.
* Protecting the affected and alternate ICRs in the beginning of Schedule Change Activation and unprotecting them once SnS Rebooking is over.

### Implementation Details

The following considerations should be taken into account during development:

* The call from Schedule Change to Inventory must NOT be performed over SOAP/HTTP due to performance considerations. Instead it should be made via internal interface of *FlightManager* service, which is described in **BDD [4]** and must be developed as part of this story.
* In order to minimise memory footprint ICR must not contain schedules data. Instead schedules data should be dynamically sourced from the existing schedules cache in grid.
* Inventory Activation must not start until the Planned Schedule for the activated ICR are propagated to grid. The suggested mechanism is that Schedule Change Orchestration sends the version of the schedules being activated to *FlightManager*, which checks if it matches the version of corresponding schedule in grid.   
  If there is a mismatch, respective error code will be returned to Schedule Change, which is expected to wait and retry.
* Temporary copy of the leg/segment created at step 2 of UC for Dress Flight must not be persisted and must be purged once dressing of this leg/segment is over.
* Additional MICT actions which need to be applied within this story are specific to Cabin, Tree, RBD and Booking Limit Template. In the same time, Cabin/Tree/RBD/BL Template fields are part of MICT Action Details of these actions, whilst MICT Extract (and MICT search from GUI) can only be performed by fields in Conditions.  
  However, no update of MICT schema is required because during dressing MICT data is queried via fast internal non-service call, which needs to be extended to add Cabin, Tree, RBD and BL Template as search criteria. There is no requirement for the external MICT webservice to support Extract by these field.

### Acceptance Tests

Definition of done requires that all test scenarios are satisfied; furthermore it is expected that the development team will identify additional scenarios to add the test pack.  
Changes made to inventory should be demonstrated via ICR GUI.

| **Nr.** | **Test** | **Pre-conditions** | **Action** | **Post-condition** |
| --- | --- | --- | --- | --- |
|  | Dress New ICR | An ICR, which is the source of retained inventory controls, exists in grid.  MICT actions exists which would result in applying different nesting templates to different legs/segments of the new flight. | Inventory Activation interface of *FlightManager* is invoked with the details of a new multi-leg ICR, and retention source is specified for its segments. Protection of ICR is requested. | The new ICR is created in the grid, all legs and segments are dressed, and controls are retained onto some of its legs/segments in accordance with the request.  Need to demonstrate:   * Inventory retention is not performed between segments with different nesting templates. * Inventory retention is not performed for cabins where at least one RBD is not present in the corresponding Retention Source cabin. * Inventory retention is performed in all other cases when instructed by the request, and retained controls overwrite those assigned during dressing. * AUs of children in the nesting structure are capped with AUs of its parents. * ICR status is set to Protected as requested. * The *FlightManager* Inventory Activation interface returns the response in accordance with the BDD for Inventory Activation. |
|  | Redress the existing ICR | An ICR which will be redressed exists in grid. | Inventory Activation interface of *FlightManager* is invoked with the details of the existing multi-leg ICR. Some of the legs/segments are new, some are changed (with retention from themselves) and some are cancelled. Protection of ICR is requested. | The ICR is changed as requested: new segments have been added, changed segments have been redressed (with inventory retention when requested) and cancelled segments are deleted.  The ICR is protected. |
|  | Cancel the existing ICR | An ICR to be cancelled exists in the grid. | Inventory Activation interface of *FlightManager* is invoked with the details of the ICR to be cancelled. | The ICR is deleted from grid. |
|  | Unprotect ICR. | A protected ICR exists in grid. | Inventory Activation interface of *FlightManager* is invoked with the details of the ICR to be unprotected (ICR Action is Change, no segment details). | The status of the ICR becomes Active. |
|  | ICR level and leg/ segment level errors | Error-prone. | Inventory Activation interface of *FlightManager* is invoked with the request which causes errors on ICR level or on leg/segment level. (See Exception flows of this usecase.) | Corresponding error codes are returned in response. |

### Non Functional Requirements

Time for activating a single flight with inventory retention should be measured and reported at the demo.

## INV.29a-1 - Apply Inventory\_Detail\_Date Parameter during Schedule Change

### Overview

The purpose of this story is to apply *Inventory\_Detail\_Date* SIAM parameter during inventory activation as part of Schedule Change.

Summary of requirements:

* During inventory activation, inventory for new flights must only be created within the Inventory Creation Window (see its definition in 1.1) – typically between today and 338 days into the future[[6]](#footnote-6). Currently inventory is created for 331 days from the beginning of a schedule period which is incorrect.
* Inventory Creation Window does not apply to replacements and cancellations which should be applied to any ICRs existing beyond the window (they could have been created by an out-of-range sell).
* Modified handling of data integrity errors comparing to the original INV.29 story – see updated Exception Flow 1. In particular, if an ICR does not exist within the Inventory Creation Window for a flight to be replaced then the ICR is automatically created.

This story is expected to use real SIAM parameter values rather than fixed values. Therefore it should be implemented after ACRH.29 Synchronise subscriber options.

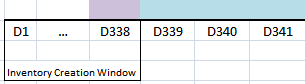
### Acceptance Tests

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

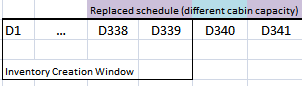
Similarly to INV.28 story, in all tests below **D1** denotes the current date at the carrier’s home time zone. For example, if the carrier is based in ATL and now in ATL is 20SEP then D2 is 21SEP.

The following diagrams illustrate the expected results of acceptance tests. **Purple** denotes dates with inventory and **teal** denotes scheduled flight dates without inventory. All specified date ranges are inclusive.

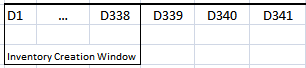
Expected results of test 1: inventory has only been created only within the Inventory Creation Window i.e. for D338.



Expected results of test 2 after having replaced the schedule, having extended the Inventory Creation Window and having performed an out-of-range sale for D341: D338 is re-dressed, D339 is created, D341 is re-dressed. All three inventory records reflect the new capacity.



Expected results of test 3 after having cancelled the schedule: inventory is not returned.



| **Nr.** | **Pre-conditions** | **Action** | **Post-condition** |
| --- | --- | --- | --- |
|  | The SIAM parameters are set to their default values, that is: Carrier\_Local\_ Time\_Zone=UTC  Open\_For\_Sale\_ Date=331  Inventory\_Detail\_ Date=338 | 1) Create a draft daily schedule extending beyond Inventory Creation Window: the period should start on D338 and end on D341.  2) Activate the schedule through Schedule Change. | * Inventory for the test flight exists on D338 * Inventory does not exist for D339 and D341 (ICR Extract request made by the flight number and departure dates range returns no results) |
|  | The previous test has been successfully executed. | 1) Set *Inventory\_Detail\_Date* parameter to 339[[7]](#footnote-7)  2) Perform an out-of-range sell for D341.  3) Create a draft replace schedule with the same details as the one from the previous test but having a different cabin capacity.  4) Activate the schedule through Schedule Change.  5) Restore the value of *Inventory\_Detail\_Date* parameter to 338. | * Inventory records exist for D338 (created by the previous test), D339 (automatically created during activation of the draft replace schedule) and D341 (created by out-of-range sell). * All three inventory records reflect the updated cabin capacity in schedules. * A warning is logged (in file, no need to log this with ErrorsManager) indicating that inventory did not exist for a flight to be replaced (for D339). * Inventory does not exist for D340. |
|  | The previous test has been successfully executed. | 1) Create a draft cancel schedule with the same flight number for the entire period between D338 and D341.  2) Activate the schedule through Schedule Change. | * The ICR Extract request does not return inventory records for the test flight between D338 and D341 (inventory records may continue to exist in grid in soft-deleted state). |

## INV.29b - Activate inventory: apply extra MICT actions

### Overview

This story covers application of extra MICT actions during dressing of a flight: Inventory Capacity Adjustment, Cabin Controls, Tree Top Controls, Nested Bucket Controls, Booking Limit Controls.

### Acceptance Tests

Definition of done requires that all test scenarios are satisfied; furthermore it is expected that the development team will identify additional scenarios to add the test pack.  
Changes made to inventory should be demonstrated via ICR GUI.

| **Nr.** | **Test** | **Pre-conditions** | **Action** | **Post-condition** |
| --- | --- | --- | --- | --- |
|  | Dress New ICR | Schedules cache in grid contains the details of the new multi-leg flight to be dressed.  MICT actions exists in grid which would result in applying different nesting templates to different legs/segments of the new flight.  In addition, ALL types of MICT actions (**more than one of the same type**) mentioned in UC for Dress Flight and matching the details of the new flights exist.  Some of the existing MICT actions are not applicable to the flight being dressed. | ICR dressing is invoked. | The new ICR is created in the grid, all legs and segments are dressed in accordance with the request.  Need to demonstrate:   * All types of MICT actions are applied. * From multiple existing MICT actions only the most specific ones are selected and applied. * Controls not affected by MICT actions remain at default levels defined from the Nesting Template (for example, Tree Top Controls MICT action should be demonstrated only on some tree tops) * AUs of children in the nesting structure are capped with AUs of its parents. * AUs of BL buckets are capped with AUs of the respective cabin. |

### Non Functional Requirements

Time for dressing a single flight with inventory retention and all MICT actions should be measured and reported at the demo.

## INV.29c - Propagate Inventory Activation Information

### Overview

The scope of this story is to create operations on *FlightInventoryService* enabling propagation of auxiliary information from Schedule Change to the Inventory Domain. This information is used when activating inventory during schedule change.

**Propagate**

*PropagateInventoryActivationInfo* operation will be used by *ScheduleChangeService* as a way of propagating Inventory Activation Information to the grid. This information relates to Inventory domain and cannot be sent to the grid along with the activated schedules. Currently it comprises:

* + 1. Inventory Retention Candidates, i.e. the legs/segments which will receive retained inventory controls (“retention targets”) and optionally the corresponding cancelled segments which need their controls to be retained (“retention source”).  
       If retention source is blank, then it means that this is a changed flight, and controls will be retained from old version of the same flight. (From is the same is To in Schedule Change Plan.)  
       Otherwise, if both retention source and target are populated, it means that inventory controls are retained from a cancelled flight onto a new one.
    2. Unchanged legs/segments, i.e. those where the changes to schedules were not significant enough to affect inventory (for example meal code change). Therefore the corresponding inventory segments do not need to be re-dressed. Since it is expected that the number of such segments can be significant, skipping them during dressing can significantly speed up inventory activation.

Both items above already exist in the Schedule Change Plan. CHANGE.03g story will enable Schedule Change to invoke *PropagateInventoryActivationInfo* for the current Schedule Change.

Note that this operation is nothing more than a mechanism of getting the auxiliary data from the Schedule Change Plan into the grid. It is not exposed to any external consumers and provides no immediate benefit to the end user, therefore there is no use case or BDD for this CRD functionality. Please refer to schema annotation for specific field details. Also note that the service is not supposed to perform any business validation of the information provided by Schedule Change.

Story INV.29e will enable the Inventory Activation process introduced by INV.29a to actually use this Inventory Activation Information: perform inventory retention and exclude unchanged segments from re-dressing.

**Delete**

Once inventory activation has been completed, this auxiliary information is no longer required and will be deleted by Schedule Change invoking the *DeleteInventoryActivationInfo* operation. This invocation is in scope of CHANE.03g.  
ID of the Schedule Change Plan will be used as a key when issuing the delete request.

### Implementation Details

**Partitioning and Persisting**

The data should be pushed into the grid using Map Reduce so that it can be used by Inventory Activation. It should be partitioned by the flight number so that each partition **only** has the information relevant to activation of its own flights. (Retention target’s flight number should be used if both source and target are specified, as in such case retention is actually performed when a new flight is created and not when the Retention Source is cancelled.)

Inventory Activation Information stored in the grid should be mirrored in the database to enable resuming of inventory activation after the system shutdown.

**“Compressing” the Data**

Both Retention Candidates and Unchanged Segments can be specific to single dates, however in most cases they will be the same for the entire period. Therefore the service supports sending the same information for a range of departure dates, and this should be used to optimize overall performance and minimise memory footprint.

### Acceptance Tests

Definition of done requires that all test scenarios are satisfied; furthermore it is expected that the development team will identify additional scenarios to add the test pack.  
This story should be demonstrated via SOAP UI.

| **Nr.** | **Test** | **Pre-conditions** | **Action** | **Post-condition** |
| --- | --- | --- | --- | --- |
|  | Create | None specific | *CreateInventoryActivationInfo* operation is invoked with some details of retention candidates and unchanged segments | Inventory Activation Info is pushed into the grid and confirmation is returned to the requestor.  It is suggested that propagation of the data into the grid is demonstrated via Gigaspaces logs. |
|  | Delete | Test A1 has been successfully completed | *DeleteInventoryActivationInfo* operation is invoked with the ID of an existing document | Inventory Activation Info for the schedule change plan specified in the request is deleted from the grid. |

## INV.29d - Apply Inventory Activation Information

### Overview

The scope of this story is to use the auxiliary information during activation of inventory in grid. This information is propagated to grid by Schedule Change using *PropagateInventoryActivationInfo* operation of the *FlightInventoryService* (story INV.29c). It will be deleted by Schedule Change invoking *DeleteInventoryActivationInfo* operation once inventory activation has been completed.

#### Identifying Legs/Segments to Perform Inventory Retention

Steps 2 and 13 of UC for Dress Flight have been changed to reflect the different way of triggering Inventory Retention process. Whenever the system dresses a leg/segment, it should check whether the leg/segment being dressed is listed among the Retention Candidates within the Inventory Activation Information, which was sent to grid by *ScheduleChangeService* in the beginning of Schedule Change execution (stories INV.29c and CHANGE.03g).

Inventory retention should be performed if:

* The leg/segment being dressed matches Retention Target in Inventory Activation Info and there is no Retention Source. It means that the retention source is the same as the retention source, i.e. controls should be retained onto the leg/segment itself (by using a temporary copy created at step 2 of UCR for Dress Flight).  
  OR
* The leg/segment being dressed matches Retention Target and there is Retention Source specified. It means that the leg/segment being dressed is a new leg/segment which should receive its inventory controls from the specified Retention Source.

Once retention source and retention target leg/segment have been identified, retaining of inventory controls is performed as described in UC for Dress Flight (assuming it was developed as part of INV.29a story).

Note: specific operational departure date of the source of retention controls should be calculated by applying *tns:RetentionSource/tns:RetentionSourceDateOffset* to the operational departure date of the leg/segment being dressed. This is to support retention of inventory controls across different dates.

For example, let’s suppose there is a flight MH1 departing at 11:30PM, and it has been cancelled during schedule change. Its controls are then retained onto a new flight MH2 (retention target), which departs at 00:30AM. If Retention Target and Retention Source are specified as date ranges, there should be a way of indicating that for every Retention Target its Retention Source departs on the previous day. *RetentionSourceDateOffset* allows to specify this:

* If it is zero, then retention source departs on the same day as retention target.
* If it is 1, then retention source departs the day after retention target.
* If it is -1, then retention source departs the day before retention target.

#### Soft-Deletion of Cancelled Legs/Segments

There is an implicit requirement that to-be-cancelled legs/segments which are the source of retention controls must not be deleted until the inventory controls are retained from them onto corresponding new flights. INV.29a addressed this issue by assuming that individual legs/segments are activated in certain order (i.e. new legs/segments are dressed first); however this will not work with the current solution.

Rather than physically deleting legs/segments to-be-cancelled, they should be marked in the grid by a special flag on leg/segment level. Soft-deleted legs/segments must be completely discarded from all uses as if they did not exist. (For example, they should not be used in availability calculation or show up in ICR GUI.) However, such legs/segments will still be accessible by inventory retention process.

Purging of soft-deleted legs/segments will be triggered by unprotect inventory call from Schedule Change to the grid performed for individual flight dates as soon as re-accommodation for them has been completed (in scope of INV.14i story).

#### Skipping Unchanged Legs/Segments when Dressing

Step 7 has been added to the current UC to enable skipping the dressing of a leg/segment if the corresponding changes to schedules were not significant enough to affect inventory. If the leg/segment being activated is listed among the Unchanged Segments within the Inventory Activation Information, which was sent to grid by *ScheduleChangeService* in the beginning of Schedule Change execution (stories INV.29c and CHANGE.03g), then this leg/segment should not be dressed.

In addition, if all legs and segments of the ICR are listed as “unchanged” then the ICR is not protected at step 11 of the current use case.

### Acceptance Tests

Definition of done requires that all test scenarios are satisfied; furthermore it is expected that the development team will identify additional scenarios to add the test pack. Acceptance tests for this story are based on those from CHANGE.03b story. It is implied that the test data should be re-used. **The story should be demonstrated via Schedules GUI, FROM-TO GUI, ICR GUI and Availability Simulation GUI.**

| **Nr.** | **Pre-conditions** | **Action** | **Post-condition** |
| --- | --- | --- | --- |
|  | Draft and Active Schedules exist in the database as specified in **Scenario 1 of CHANGE.03b story.** | The user selects to activate draft schedules via Schedule Activation GUI and starts execution of the resulting Schedule Change Plan via From-To GUI. | 1) Inventory controls from the cancelled CGK-KUL leg/segment of MH1 flight between May and August are retained onto the corresponding leg/segment of MH2 flight.  2) Inventory controls from the cancelled KUL-BLR leg/segment of MH1 flight in May are retained onto the corresponding leg/segment of MH3 flight.  3) Inventory legs and segments KUL-BLR between June and August are not re-dressed (assuming there were no significant changes to schedules).  4) Cancelled legs/segments do not appear in availability requests (should be demonstrated via Availability Simulation GUI) and in ICR GUI. |
|  | Draft and Active Schedules exist in the database as specified in **Scenario 2 of CHANGE.03b story.** | The user selects to activate draft schedules via Schedule Activation GUI and starts execution of the resulting Schedule Change Plan via From-To GUI. | 1) Inventory controls of the changed KUL-LED leg/segment flying in July are retained onto new version of this leg/segment. Need to demonstrate that the controls are prorated in accordance with the changed capacity.  2) Inventory legs and segments KUL-SYD flying in May, June, August and September are not re-dressed. |
|  | Draft and Active Schedules exist in the database as specified in **Scenario 3 of CHANGE.03b story.** | The user selects to activate draft schedules via Schedule Activation GUI and starts execution of the resulting Schedule Change Plan via From-To GUI. | Inventory legs and segments KUL-LED flying in May, June, August and September are not re-dressed. |

## INV.29e - Notify Schedule Change on Inventory and Schedules Activation

### Overview

The scope of this story is to notify Schedule Change service once the inventory records corresponding to a set of intended DRAFT schedule periods have been activated in grid. This activity is represented with step 6 in section 14.2.2.2 of **UC for Schedule Change Orchestration**.

#### Background

When the system begins executing a Schedule Change Plan, the *ScheduleChangeService* invokes *ActivateHostSchedules* operation of the *FlightSchedulesService* with a set of intended schedule changes (i.e. DRAFT schedule periods) **with the same flight number** to be activated. (Described in CHANGE.03b story.)

Then the Schedules Manager:   
  
1) Applies these changes to the existing ACTIVE schedules and defragments the result

2) Persists the resulting new schedules in the Weblogic’s database

3) Calculates the difference (“delta”) between the new and old schedules and propagates it to the grid via *ScheduleChangeAgent*. (All of the above items are in scope of SCHED.03j story.)  
  
Prior to applying the “deltas” to the active schedules in grid, *FlightManager* calculates leg/segment level difference between the two (i.e. identifies new, cancelled and existing legs/segments, which is covered by CHANGE.03b story). Once the schedules in grid have been updated, this difference is used as an input to Inventory Activation process, which was covered by INV.29 story.

#### Scope

Once the *FlightManager* finishes activation of all inventory records matching the leg/segment difference (identified from the current set of intended DRAFT schedule periods with the same flight number), it should invoke *ScheduleChangeActivationStatus* operationof the *ScheduleChangeService.* This operation should have been already implemented as part of CHANGE.03h story and allows Schedule Change to acknowledge the inventory activation and issue activation request for the next set of intended DRAFT schedule periods.

The inputs to this operation are (see XML schema for types and multiplicities):

* ID of the schedule change plan for which the inventory has been activated.
* IDs of the intended DRAFT schedule periods, which triggered the inventory activation. Along with the ID of the Schedule Change Plan, they are used throughout the whole process as correlation IDs and allow the Schedule Change Process to orchestrate activation.

Please note that in the future story INV.29h the inventory activation notification will be extended with error statistics. In the current implementation the requirement is only to log all errors with Error Manager. Even if there have been errors during activation, all the intended DRAFT periods with the current flight number should be included in *ScheduleChangeActivationStatus* request.

### Acceptance Tests

Definition of done requires that all test scenarios are satisfied; furthermore it is expected that the development team will identify additional scenarios to add the test pack. Acceptance tests for this story are based on those from CHANGE.03b story. It is implied that the test data should be re-used. The story should be demonstrated via Schedules GUI, FROM-TO GUI and SOAP UI.

| **Nr.** | **Pre-conditions** | **Action** | **Post-condition** |
| --- | --- | --- | --- |
|  | Draft and Active Schedules exist in the database as specified in **Scenario 1 of CHANGE.03b story.** | The user selects to activate draft schedules via Schedule Activation GUI and starts execution of the resulting Schedule Change Plan via From-To GUI. | Inventory and schedules for MH1, MH2 and MH3 flights have been activated.  *ScheduleChangeActivationStatus* operation of the *ScheduleChange* service has been invoked **three times**: one time for each of the three flight numbers.  All **five** intended draft schedule periods defined in the test have been marked as Activated by setting the following Boolean element to true: tns:ShowScheduleChangePlan/tns:ScheduleChangePlan/schd:IntendedScheduleChanges/schd:IntendedScheduleChange/schd:ActivatedInd  (This should be demonstrated via SOAP UI) |

## INV.28 - Advance Inventory Detail Windows

### Overview

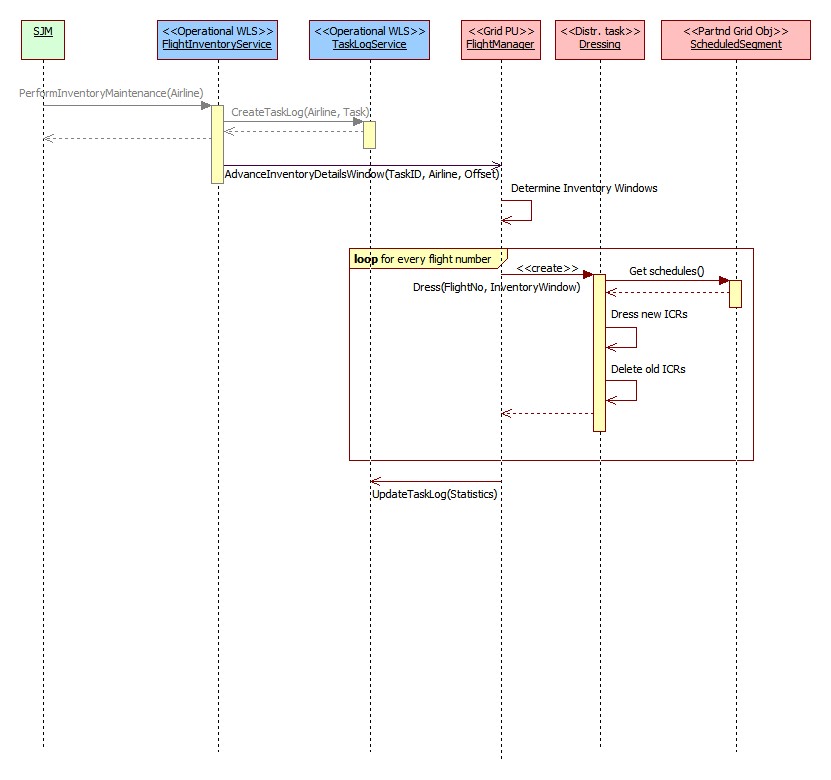
This story covers Basic Flow 2 of the use case: creation of ICRs for the oncoming flight dates and deletion of inventory records for the flown flights.

Advancement of Inventory Details Window is one of activities performed during Nightly Inventory Maintenance hence INV.28 can only be implemented after INV.69.

This story is expected to use real SIAM parameter values rather than fixed values. Therefore it should be implemented after ACRH.29 Synchronise subscriber options.

### Implementation details

* Advancement of Inventory Details Window is performed in grid by the *FlightManager* service, re-using the existing code for Inventory Dressing – see the sequence diagram below (based on the sequence diagram of INV.69)



* When determining which flight dates to dress *FlightManager* works off the local schedules cache in the grid rather than querying *FlightSchedulesManager* in Weblogic. It is assumed that the schedule change process will ensure that host schedules in grid are in sync with Weblogic
* There is a possibility for nightly re-dressing process to run past midnight at carrier’s timezone. For consistency, any calculations of inventory windows must use the actual task start date **at the carrier’s local timezone** instead of the current date
* The Inventory Creation Window most likely will be limited to about 350 days. It is assumed that the time required for *FlightManager* to check if ICR exists against every scheduled flight operation within this window is insignificant (<5ms per flight is acceptable). If this is not the case then alternative solutions should be discussed with CFT
* ICRs for flights outside of the Inventory Preservation Window are physically deleted from the grid and the database (i.e. there is no need to put into PendingDeletion state)
* Unlike re-dressing, creation of an ICR is atomic. If an error occurs while dressing a single leg or segment of an ICR then the entire ICR is not persisted and the error is logged
* In order to limit impact on performance the system should perform inventory window advancement iterating by flight numbers. Multiple partitions can be performing the task in parallel. The control point is *FlightManager* which should distribute the task between partitions requesting to process no more than N flight numbers at a time, where N is a configurable value and can be 5 by default
* Like in Nightly re-dressing (INV.69), *FlightManager* puts processing statistics to *TaskDetails* element of the respective Task Log item by invoking the *UpdateTaskLog* operation.  
  The XML type to use for detailed statistics is *NightlyInventoryMaintenanceStatisticsType* defined in the Inventory Maintenance schema.  
  Summary statistics are also updated – see **section 3.3.1 of BDD [8]**

#### Summary of logging requirements

The service must at least log the following events (events with Error severity must be additionally logged in the *ErrorsManager* service):

| **Severity** | **Situation** |
| --- | --- |
| Info | A flight under emergency lock could not be deleted (alternate flow 1). |
| Error | An ICR has been created within the Open for Sale window during inventory window preservation. |
| Info | An Inventory window preservation task has been created specifying a non-zero “Inventory Details Window Offset” (should be used for test purposes only). |
| Error | Dressing error of an ICRs leg or segment. |

### Acceptance Tests

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

In all tests below, **D1** denotes the current date at the carrier’s home timezone. For example, if the carrier is based in ATL and now in ATL is 20SEP then D2 is 21SEP.

SOAP UI tests ideally should not depend on the specific values of SIAM parameters (namely, UTC carrier timezone, 7 and 338 days of the inventory window). The current values of SIAM parameters may be fetched using *GetSubscriberParameters* operation of the *SubscriberProfileManager* service.

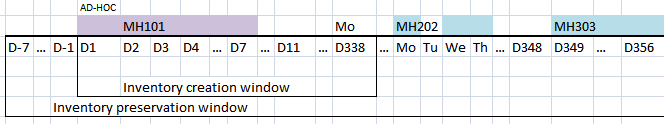
These tests focus on verification of the following requirements:

* Deletion of old inventory is based on departure date of the last leg. This is why the first schedule period of the test flight MH101 has two legs and the second leg has Date Variation +1.
* Flights outside of the current inventory window are not dressed by inventory window advancement process. In order to check that, the second flight MH303 is put entirely outside of the inventory window.
* Creation of new inventory is not only restricted by the current Inventory Creation Window but also follows the schedule pattern (taking into account days of week and frequency). This is why the third flight MH202 only operates on certain days of the week.
* Statistics are updated for both sunny-day scenarios as well as for failures. This is why the third flight is set up so that on a certain date it cannot be dressed with the existing nesting templates.
* If dressing of a single leg/segment fails then the entire ICR is not created.
* Inventory is created and/or deleted for ad-hoc schedules as well as for standard schedules.

The tests assume that there are no other MH flights, which could alter the processing statistics namely those where Period Start adjusted by date variation of the last leg is less than D3 or Period End is greater than D338. In addition to that, a check must be built in to ensure that the individual tests are not separated by midnight at HNL, otherwise the processing statistics would not match the expected results.

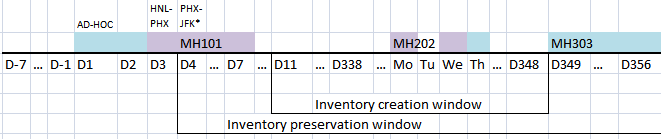
The following diagrams illustrate the expected results of acceptance tests. Purple denotes dates with inventory and teal denotes scheduled flight dates without inventory. All specified date ranges are inclusive.

Expected results of test 1: inventory for MH101 has been created as a result of Schedule Change. Inventory for MH202 and MH303 was not created because both flights are outside of the current Inventory Creation window.



Expected results of test 2: inventory windows have been offset by 10 days simulating advancement. This causes inventory for D1 and D2 to be deleted. Inventory for D3 remains intact as its last leg has date variation of +1 and hence is still within the Inventory Preservation Window.

Inventory for MH202 has been created in accordance with the flight’s operating days of week. Inventory was not created for Thursday due to dressing error.



| **Nr.** | **Pre-conditions** | **Action** | **Post-condition** |
| --- | --- | --- | --- |
|  | The SIAM parameters are set to their default values, that is: Carrier\_Local\_ Time\_Zone=UTC  Departed\_Flight\_ Date=7  Inventory\_Detail\_ Date=338 | 1) Create a new schedule period for a flight MH101, which operates daily, starts on D2 and ends on D7. The stations and timings must be as follows: HNL2215 PHX0710/1[[8]](#footnote-8) PHX1300/1 JFK2045/1  2) Create a new schedule period for a single-legged flight MH202 departing HNL, which operates on Mondays, Wednesdays and Thursdays, starts on the first Monday after D338 and ends **four** days later (on Thursday).  The **second leg** of this schedule period has an RBD, which is not present in the default nesting template, for example “E”.  3) Create a nesting template containing all RBDs of MH202 (including “E”) and create an MICT rule assigning this template specific to MH202 flight and departure dates between the first Monday after D338 and the following **Wednesday**. With such set-up, we achieve that the second leg of the last operation of the flight (Thursday) cannot be dressed once the inventory details window is advanced.  4) Create a new schedule period for a single-legged flight MH303 departing HNL**,** which operatesdaily,startson D349 and ends on D356.  5) Activate the schedule through Schedule Change and check the created inventory using the ICR Extract service.  6) Create and activate a new ad-hoc flight for MH101 operating on the D1 (i.e. today). The timing must not be in the past; hence, it can be different from the timing of the standard schedule (e.g. current time at HNL+10 minutes). | This expected result assumes that *Inventory\_Detail\_Date* parameter is applied during Schedule Activation (subject of INV.29a-1 story)  1) Inventory for MH101 exists on D1 through D7 (7 ICRs must be returned by Extract ICR in response to a request for MH101 and this date range).  2) Inventory for MH202 and MH303 does not exist on D339 through D356. |
|  | The previous test has executed successfully. | 1) Create “Advance inventory windows” inventory maintenance task for MH carrier setting the *Offset* to 10 (this sets the current window between D4 and D348).  2) Wait until the task’s status changes to “Completed” then check the inventory and task’s statistics. | 1) Inventory for MH101 does not exist on D1 and D2.  2) Inventory for MH101 exists on D3 through D7 (this is because the lower boundary of the window is applied against the departure of the flight’s last leg, which is +1 day for MH101).  3) Inventory for MH303 does not exist between D349 and D356.  4) **Two** inventory records for MH202 exist in between D339 and D348, one of them is Monday and the other is Wednesday.  5) The task’s statistics are as follows: Flights created: 2 Flights deleted: 2 Flights failed: 1 Legs created: 4 Legs deleted: 4 Legs failed: 2 Segments created: 6 Segments deleted: 6 Segments failed: 3 The task’s processing start and finish times are returned. |
|  | The previous test has executed successfully. | 1) Modify the “Assign nesting template” MICT rule created in test A1 to extend its departure date range by 1 day into the future. This change should fix the dressing error for Thursday’s operation of MH202.  2) Create “Advance inventory window” task with the same settings as in the previous test.  3) Wait until the task’s status changes to “Completed” then check the inventory and the new task’s statistics. | 1) **Three** inventory records for MH202 exist between D339 and D348.  2) The task’s statistics are as follows: Flights created: 1 Legs created: 2 Segments created: 3 |

## INV.29a-2 - Flight Dressing – Rework

### Overview

This rework task to flight dressing includes;

* Applying the correct logic when assigning nesting template to segments during dressing of segment ICRs.
* Validating the manner by which segment cabin inventory capacities are determined.
* Inventory Retention methodology will need to be modified to cater to different subscriber parameter, viz., **Inventory\_Bucket\_Control\_Type** (AU or PLs)

### Rework Details

The framework for dressing an ICR is already covered under INV.29a task. However prior to taking up this rework task, it is assumed that the changes to *FlightInventoryControlsManager* interface as defined in **[11]** is already delivered.

The following modifications need to be addressed as part of this story:

* While assigning nesting templates to segments (Step 3 of Basic Flow 1**, [3]**), the cabins of the nesting template must exactly match cabins of the segments longest leg by total kilometres

Eg: If there are three legs for a flight, where leg 1 is F/Y, leg 2 is C/Y & leg 3 is C/Y. Suppose Leg 1 is the longest leg but the sum of leg 2 and leg 3 miles are more, then the segment will have C/Y assigned (whereas, in current implement Leg 1 cabin F/Y is getting applied to the segment)

Even though this is a part of INV.07c implementation, the above validation must be addressed within the scope of this task.

* The default Inventory Capacity assigned to segments (Step 4 of Basic Flow 1**, [3]**), prior to applying MICT rule 'Inventory Capacity Adjustment', should be the smallest capacity for the same cabin type across (i.e. common cabins) all legs.

Eg: if leg AB is F10Y200 and BC is F20Y100, then Seg AC should take smaller common cabin cap ie F10Y100. Some more scenarios are provided in section 12.8.3:

* Inventory Retention needs to support subscriber parameter '**Inventory\_Bucket\_Control\_Type'**, which defines how the nested bucket controls are used to calculate effected bucket authorization. Sub Flow 1 of **[3],** is updated to cater to the changed definition.
  + During retention, when an RBD is missing in the leg/seg of Retention Target flight the system **should not** add its protection limits (PL) to parent bucket RBD (This may involve removal of existing logic already implemented).
  + Likewise when there are more RBDs in leg/seg of retention target flight than its retention source (vice versa case), the system must simply retain the PLs that were assigned to them by nesting template.
  + The hierarchy of bucket is based on its serial number. Higher the numeric value, lower the level of the bucket. For parallel/dual/hybrid nesting structure, in addition to numeric value the alphabetic order will apply to determine level of buckets, Depending on the hierarchy/level of the buckets, system must adjust excess seats accordingly (when capacity is reduced). Same when capacity increases.
    - eg: Y01.A03 is lower than Y01.C02. But Y01.C03 will be lower than Y01.B03 and Y01.A03. Similarly Y01.B03 is lower than Y01.A03.

Note: Each tree has its own hierarchy. When there are multiple treetops, then seat adjustments are carried out within each tree independent to the others.

* Capping child controls by its corresponding parent controls is not required and in case this is implemented it must be amended**.** Allcappingis addressed as a part of UC for Full Inventory Adjustment, which is invoked during dressing by UC for Dress Flight.
* Booking Limit buckets should be dressed using most specific 'Inventory Capacity Adjustment' MICT rule, else by Inventory Capacity from schedules.

### Additional examples for assigning segment inventory capacity

Few examples of how cabins are assigned to segments (based on nesting template) and the subsequent cabin inventory capacity that needs to be allocated to the segment (with multi-legs) is provided below.

The leg distance (total kilometers) influences what cabins a segment will get assigned and in the next step the smallest capacity (of the common cabins) will then get assigned to the segment.

| **No** | **Leg** | **Leg Cabin Config/Cap.** | **Distance** | **Segment** | **Seg. Cabin/capacity assigned** |
| --- | --- | --- | --- | --- | --- |
| 1 | Leg A-B | F 10 Y 100 | 100 | Seg A-B | F 10 Y 100 |
| Leg B-C | **F** 10 **Y** 200 | 200 | Seg B-C | F 10 Y 200 |
|  |  |  | **Seg A-C** | **F 10 Y 100** |
| 2 | Leg A-B | F 10 Y 100 | 100 | Seg A-B | F 10 Y 100 |
| Leg B-C | **F** 20 **Y** 75 | 200 | Seg B-C | F 20 Y 75 |
|  |  |  | **Seg A-C** | **F 10 Y 75** |
| 3 | Leg A-B | **F** 10 **C**30 **Y** 100 | 200 | Seg A-B | F 10 C30 Y 100 |
| Leg B-C | C 20 Y 200 | 100 | Seg B-C | C 20 Y 200 |
|  |  |  | **Seg A-C** | **F 10 C20 Y 100** |
| 4 | Leg A-B | **F** 10 **Y** 100 | 200 | Seg A-B | F 10 Y 100 |
| Leg B-C | C 20 Y 200 | 100 | Seg B-C | C 20 Y 200 |
|  |  |  | **Seg A-C** | **F 10 Y 100** |
| 5 | Leg A-B | F 10 Y 100 | 100 | Seg A-B | F 10 Y 100 |
| Leg B-C | **C** 20 **Y** 200 | 200 | Seg B-C | C 20 Y 200 |
|  |  |  | **Seg A-C** | **C 20 Y 100** |
| 6 | Leg A-B | **F** 10 **Y** 100 | 200 | Seg A-B | F 10 Y 100 |
| Leg B-C | **C** 20 **Y** 200 | 150 | Seg B-C | C 20 Y 200 |
| Leg C-D | **C** 20 **Y** 200 | 150 | **Seg A-C** | **F 10 Y 100** |
|  |  |  | **Seg A-D** | **C 20 Y 100** |
| 7 | Leg A-B | Y 100 | 150 | Seg A-B | Y 100 |
| Leg B-C | **C** 20 **Y** 150 | 250 | Seg B-C | C 20 Y 150 |
| Leg C-D | **F** 20 **Y** 200 | 350 | Seg C-D | F 20 Y 200 |
|  |  |  | **Seg A-D** | **F 20 Y 100** |
|  |  |  | **Seg A-C** | **C 20 Y 100** |
|  |  |  | **Seg B-D** | **F 20 Y 150** |

### Acceptance Tests

Definition of done requires that all test scenarios are satisfied; furthermore it is expected that the development team will identify additional scenarios to add the test pack.  
Changes made to inventory should be demonstrated via ICR GUI.

| **Nr.** | **Test** | **Pre-conditions** | **Action** | **Post-condition** |
| --- | --- | --- | --- | --- |
|  | Dress New ICR | Schedule as shown in sl.no.7 of section 12.8.3 exists in the system in Draft status  Subscriber parameter for 'Inventory\_Bucket\_Control\_Type' is set to 'Authorization' or 'Protection limits'  MICT actions exists which would result in applying different nesting templates to different legs/segments of the new flight. | Activate schedule | The new ICR is created in the grid, all legs and segments are dressed.  Need to demonstrate:   * Segments are assigned correct nesting template as per new logic * Cabin capacities allocated to segments are as per smallest common cabin capacity of its legs. * Depending on Subscriber parameter the PLs and effective AUs for nested buckets, are correctly populated |
|  | Dress New ICR with Retention | An ICR, which is the source of retained inventory controls, exists in grid.  Subscriber parameter for 'Inventory\_Bucket\_Control\_Type' is set.  MICT actions exists which would result in applying different nesting templates to different legs/segments of the new flight. | Inventory Activation interface of *FlightManager* is invoked with the details of a new multi-leg ICR, and retention source is specified for its segments. Protection of ICR is requested. | The new ICR is created in the grid, all legs and segments are dressed, and controls are retained onto some of its legs/segments in accordance with the request.  Need to demonstrate:   * Inventory retention for missing RBDs is as per new logic. * Inventory retention is performed in all other cases when instructed by the request, and retained controls overwrite those assigned during dressing. * Subscriber parameter influences how nested bucket controls are retained (AUs or PLs) * ~~AUs of children in the nesting structure are capped with AUs of its parents.~~ |
| 1. ` | Redress the existing ICR | An ICR which will be redressed exists in grid. | Inventory Activation interface of *FlightManager* is invoked with the details of the existing multi-leg ICR. Some of the legs/segments are new, some are changed (with retention from themselves) and some are cancelled. Protection of ICR is requested. | The ICR is changed as requested: new segments have been added, changed segments have been redressed including booking controls (with inventory retention when requested) and cancelled segments are deleted.  The ICR is protected. |
|  | Retention when RBDs between Source & Target flights don’t match | An ICR exists in the grid. 1. Draft cancel for the above flight exits.  2. Another Draft New exists which is expected to retain inventory controls from the active ICR ( ie same Nesting ID applies to both) | Activate both the schedule.  Test scenarios where RBDs in target flight is less than source flight & vice versa  Test with AU & PL subscriber parameter | System retains inventory controls from source to target flight though some RBDs are not same  Note: When target flight has more RBDs, then controls (PLs) from the nesting template are retained for such additional RBDs. |

## INV.29a-3 - Flight Dressing - Groups and Waitlist

### Overview

This story covers dressing leg / segment ICRs for Group & Waitlist. All MICT actions during dressing of a flight, including Inventory Retention for groups & waitlist need to be delivered in this scope.

### Details

This story should leverage the development work for dressing & retention, which is already delivered for booking controls (BAUs).

* All Steps of Basic flow 1, **[3]** are applicable to groups and waitlist.
* Groups and waitlist controls for legs and segments get created when they are dressed with default inventory controls (during step 5 of basic flow 1, **[3]**, which further invokes UC for full inventory adjustment)
* Inventory retention as defined in Sub Flow 1 of **[3],** needs to be implemented for groups and waitlist controls also.
* Raiding is not within the scope of this story.

### Acceptance Tests

Definition of done requires that all test scenarios are satisfied; furthermore it is expected that the development team will identify additional scenarios to add the test pack.  
Changes made to inventory should be demonstrated via ICR GUI.

| **Nr.** | **Test** | **Pre-conditions** | **Action** | **Post-condition** |
| --- | --- | --- | --- | --- |
|  | Dress New ICR for Groups & Waitlist | Schedule in draft status exists.  Subscriber parameter for 'Inventory\_Bucket\_Control\_Type' is set to 'Authorization' or 'Protection limits'  MICT actions exists which would result in applying different nesting templates to different legs/segments of the new flight. | Activate schedule | The new ICR is created in the grid, all legs and segments are dressed.  Need to demonstrate:   * Group controls are populated * Waitlist controls are populated * Waitlist indicators are correctly set * Depending on Subscriber parameter the PLs and effective AUs for group & waitlist, are correctly populated |
|  | Dress New ICR with Retention | An ICR, which is the source of retained inventory controls, exists in grid.  Subscriber parameter for 'Inventory\_Bucket\_Control\_Type' is set.  MICT actions exists which would result in applying different nesting templates to different legs/segments of the new flight. | Inventory Activation interface of *FlightManager* is invoked with the details of a new multi-leg ICR, and retention source is specified for its segments. Protection of ICR is requested. | The new ICR is created in the grid, all legs and segments are dressed, and controls are retained onto some of its legs/segments in accordance with the request.  Need to demonstrate:   * Inventory Retention is carried out for Groups & Waitlist controls * Subscriber parameter influences how nested bucket controls are retained (AUs or PLs) * ~~AUs of children in the nesting structure are capped with AUs of its parents.~~ * The *FlightManager* Inventory Activation interface returns the response in accordance with the BDD for Inventory Activation. |
|  | Redress the existing ICR | An ICR which will be redressed exists in grid. | Inventory Activation interface of *FlightManager* is invoked with the details of the existing multi-leg ICR. Some of the legs/segments are new, some are changed (with retention from themselves) and some are cancelled. Protection of ICR is requested. | The ICR is changed as requested: new segments have been added, changed segments have been redressed including groups & waitlist controls (with inventory retention when requested) and cancelled segments are deleted.  The ICR is protected. |

### Non Functional Requirements

Time for activating a single flight with inventory retention should be measured and reported at the demo.

## INV.29i - Retain Manual/ RMS/ Bid Price indicator controls

### Overview

This story covers retaining Manual, RMS and Bid Price indicator controls as part of schedule change retention process.

### Details

While retaining inventory controls during schedule change process from source flight to target flight, it is necessary to retain the following indicators also

* Manual control indicator (*ManualControlInd*)
* Revenue (RMS) controls indicator (*RevenueControl*)
* Bid price indicator (*BidPriceInd*)

A new step has been added to address this gap in the Inventory retention Sub Flow 1 of **[3]**.

### Acceptance Tests

Definition of done requires that all test scenarios are satisfied; furthermore it is expected that the development team will identify additional scenarios to add the test pack.  
Changes made to inventory should be demonstrated via ICR GUI.

| **Nr.** | **Test** | **Pre-conditions** | **Action** | **Post-condition** |
| --- | --- | --- | --- | --- |
|  | Dress New ICR with Retention  (Can reuse test A2 from INV.29a2) | An ICR, which is the source of retained inventory controls, exists in grid.  Subscriber parameter for 'Inventory\_Bucket\_Control\_Type' is set.  MICT actions exists which would result in applying different nesting templates to different legs/segments of the new flight. | Inventory Activation interface of *FlightManager* is invoked with the details of a new multi-leg ICR, and retention source is specified for its segments. Protection of ICR is requested. | The new ICR is created in the grid, all legs and segments are dressed, and controls are retained onto some of its legs/segments in accordance with the request.  Need to demonstrate:   * Manual control indicator, Revenue (RMS) control indicator & Bid price indicator are retained |
|  | Redress the existing ICR  (Can reuse test A3 from INV.29a2) | An ICR which will be redressed exists in grid. | Inventory Activation interface of *FlightManager* is invoked with the details of the existing multi-leg ICR. Some of the legs/segments are new, some are changed (with retention from themselves) and some are cancelled. Protection of ICR is requested. | The ICR is changed as requested: new segments have been added, changed segments have been redressed including groups & waitlist controls (with inventory retention when requested) and cancelled segments are deleted.  Manual control indicator, Revenue (RMS) control indicator & Bid price indicator are retained  The ICR is protected. |

### Non Functional Requirements

Time for activating a single flight with inventory retention should be measured and reported at the demo.

## INV.29j – Process Host ASM in Gigaspaces

### Summary

When an airline deviates from its standard schedule for a single operation of an existing flight (Flight identifier date) such as a change in equipment, aircraft configuration, routing, timing or other data or cancellation of a flight are transmitted using the Ad-Hoc Message (ASM).

This task addresses certain improvements to flight schedules in Gigaspaces (In Memory Data Grid) to enhance its suitability to handle and process (compute flight availability) for these Ad-hoc schedule messages

### Overview

* The following ASMs will be handled for this task
  + ASM / NEW – Insertion of new flight information, defined by a
    - flight identifier that has previously not existed

Or

* + - Flight identifier that had been cancelled
  + ASM/CNL – Cancels (i.e. declares as not operating) an existing flight identifier
  + ASM/RPL –Replacement of existing information for a flight identifier. Can be
    - Change equipment information (EQT)
    - Change Aircraft configuration version –ACV (CON)
    - Change of flight identifier (FLT)
    - Change of routing information (RRT)
    - Change of time information (TIM)
    - Change of information expressed by DEI (ADM)
    - Any change that involves all of the above or any combination of the above (RPL)
* When a flight schedule is cancelled using ASM/CNL the Inventory status – Suspended is set
* When a flight schedule needs to be reinstated using ASM/RIN then status is reverted from Suspended to Active
* ASM/NEW can also be a flight identifier that has previously been cancelled (Suspended) and ASM/NEW means that 'Suspended' is changed to 'Active' and flight is re-dressed.

### Scope

The scope for this story is to test a set of acceptance test scenarios. No fresh development is expected to be part of this story.

### Acceptance Tests

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

**Sample Test Data**

|  |  |
| --- | --- |
| Flight designator | BA100 |
| Flight identifier Date (FID) | 01 NOV 2014 |
| Flight identifier (Flight designator + FID) | BA100 01NOV14 |
| Service type | G |
| Aircraft type | 767 |
| ACV | J20W40Y60 |
| PRBD | JDCIZG WSHKP YBRLUMEQVXNO |
| Departure station | LHR |
| Aircraft STD | 1300 |
| Arrival station | JFK |
| Aircraft STA | 2000 |

**All tests are to be performed through the Availability Screen in the GUI**

| **No.** | **Test** | **Pre-Condition** | **Action** | **Post-Condition** |
| --- | --- | --- | --- | --- |
| 1 | ASM/NEW  Flight identifier that does not exist | Effective ad-hoc schedule for Flight identifier (BA100 01NOV)  does not exist in Gigaspaces ( In-memory Data Grid) | Request availability for flight identifier (BA100 01 NOV 2014) | No Flight availability is returned for flight identifier (BA100 01 NOV 2014) |
| 2 | ASM/NEW  Flight identifier that does not exist before | Effective ad-hoc schedule for Flight identifier (BA100 01NOV)  exists in the Grid | New flight identifier (BA100 01 NOV 2014)  Request availability for flight identifier (BA100 01 NOV 2014) | Flight availability is returned only for flight identifier (BA100 on 01 NOV 2014) |
| 3 | ASM/CNL  To cancel an existing flight identifier | Post condition as per test 2 | Cancel flight identifier (BA100 on 01 NOV 2014)  Request availability for flight identifier (BA100 on 01 NOV 2014) | No Flight availability is returned for flight identifier (BA100 on 01 NOV 2014).  (Status is 'Suspended') |
| 4 | ASM/RIN  Reinstate a cancelled flight identifier | Post condition as per test 3 | Reinstate flight identifier (BA100 on 01 NOV 2014  Request flight availability for flight identifier (BA100 on 01 NOV 2014) | Flight availability is returned only for flight identifier (BA100 on 01 NOV 2014) - the reinstated flight number  (Status is reset to Active) |
| 5 | ASM/NEW (RTNS)  Flight identifier that has been cancelled | Post condition as per test 3 | New flight identifier (BA100 on 01 NOV 2014) by changing the  Request flight availability for flight identifier (BA100 on 01 NOV 2014) | Flight availability is returned only for flight identifier (BA100 on 01 NOV 2014) - the new flight number |
| 6 | ASM/EQT  Change equipment information  For existing Flight identifier | Post condition as per test 4 | Change Equipment information (Aircraft type) to B777 – a bigger aircraft with a much bigger  Inventory capacity  (**J20W50Y100**)  Request normal availability for Flight identifier (BA100 on 01 NOV 2014)  Request full availability for Flight identifier (BA100 on 01 NOV 2014) | Flight availability (normal and full) for flight identifier (BA100 on 01 NOV 2014) based on the new inventory capacity (J20W50Y100) |
| 7 | ASM/CON  Change ACV information  For existing Flight identifier | Post condition as per test 4 | Change the ACV information for flight identifier (BA100 on 01NOV 2014) so that it becomes **J20W30Y70**  Request normal availability for Flight identifier (BA100 on 01 NOV 2014)  Request full availability for Flight identifier (BA100 on 01 NOV 2014) | Flight availability (normal and full) returned for flight identifier( BA100 on 01 NOV 2014) based on the new Aircraft configuration **J20W30Y70** |
| 8 | ASM/FLT  Change flight identifier  For existing Flight identifier | Post condition as per test 4 | Change Flight designator from BA100 to BA200  Request flight specific normal availability for Flight identifier (BA200 on 01 NOV 2014) | Flight availability is returned for BA200 (new flight designator) on 01 NOV 2014 |
| 9 | ASM/TIM  Change time information  For existing Flight identifier | Post condition as per test 4 | Change Timing information for Flight identifier (BA100 on 01 NOV 2014) from 1300 to 1600  Request normal availability for Flight identifier (BA100 on 01 NOV 2014) | Flight availability is returned for flight identifier (BA100 on 01 NOV 2014) showing the new time of 1600 |
| 10 | ASM/ADM  Change of Existing Information Expressed by the Use of Data Element Identifier  Only  For existing Flight identifier | Post condition as per test 4 | For flight identifier (BA100 on 01 NOV 2014) change only those data elements which are specified by the use of Data Element Identifier for example  Request normal availability for Flight identifier (BA100 on 01 NOV 2014) | Flight availability is returned for flight identifier (BA100 on 01 NOV 2014) |
| 11 | ASM/RPL  Replaces all or any combination of information for a flight identifier  For existing Flight identifier | Post condition as per test 4 | For flight identifier (BA100 on 01 NOV 2014) change the following  Flight identifier to BA300  Aircraft type to A320 –bigger  Inventory capacity  (**J10W70Y400**)  Time from 1300 to 1400  Request normal availability for Flight identifier (BA300 on 01 NOV 2014) | Flight availability is returned for flight identifier (BA300 on 01 NOV 2014) showing the new time of 1400 and based on the new Aircraft configuration **J10W70Y400**) |

1. The number is defined in *Departed\_Flight\_Date* SIAM parameter. Refer to [9] for more up-to-date information on SIAM parameters. [↑](#footnote-ref-1)
2. The host carrier’s timezone is defined in *Carrier\_Local\_Time\_Zone* SIAM parameter [↑](#footnote-ref-2)
3. The number is defined in *Inventory\_Detail\_Date* SIAM parameter. [↑](#footnote-ref-3)
4. The number is defined in *Maximum\_Sell\_Date* SIAM parameter. [↑](#footnote-ref-4)
5. This value allows shifting the inventory windows and is used for test purposes only. [↑](#footnote-ref-5)
6. It is assumed that inventory for new flights outside of the window and used for re-accommodation will be created upon re-booking request from Sales & Services (whether it will be reconcile or a set of sell request). [↑](#footnote-ref-6)
7. It is assumed that SOAP UI can use *PutSubscriberParameters* operation of the *SubscriberProfileManager* service to temporarily change parameter values. [↑](#footnote-ref-7)
8. Arrival on the next day is denoted as “/1” in SSM and “01” in SDS. Departure and arrival on the next day is denoted as “11” in SDS. See IATA SSIM definition of Date Variation DEI. [↑](#footnote-ref-8)