273276 v1.0

Aptean Ltd

Contents

1 273276

aptean

DHL MTS

Aramco Badge Number Validation

FUNCTIONAL SPECIFICATION - 10.4.7

15/09/2010 - 1.0
Reference: FS 273276 SA-7Y59ZG

1.1 Client Requirement

Change Request Summary:

This RIO is for validation to be applied in MTS against Aramco badge numbers (POD/CMR No).

Change Request Details:

Currently in MTS, there is no validation to check if badge number is correct or not. We need to have validation in MTS to prevent us using Exel badge numbers or invalid badge numbers e.g. those containing Alpha characters.

Benefits identified as a result of the change:

Business requirement to decrease PODs rejection.

1.2 Solution

A new table called ?ADM_VALID_FORMATS? will be introduced to store invalid formats for the badge numbers.

It is expected that a number of different formats will be stored as alphanumeric characters up to a maximum of 20 characters in length. The new table will store 2 columns as VARCHAR2(20): ?AREA? (e.g. ?BADGE_NUMBER?) and ?INVALID_FORMAT?. This will enable the table to be used for different items during validation in different processes (VARCHAR2(20) is also the definition of the existing column ?BADGE_NUMBER? on the table ?HHT_INBOUND_DETAIL? so this will be consistent).

The badge numbers are currently validated as follows when the ?DETAIL? line of the HHT inbound message ?HHTtripmil? is uploaded for a trip stop of type ?DL?:

1. The length must not be more than 12 digits
2. The digits must be numeric with values between 0 and 9
3. If the length is 12 digits then it must start and end with ?00? and digits 3 to 9 will be stored
4. If the length is 8 digits then digits 1 to 7 will be stored
5. If the length is not 8 or 12 digits then all digits will be stored

The badge number provided will then be stored as the POD name against the scheduled order when it is updated to status ?DELIVERED?.

For example, ?00########00? would represent a valid format for point 3 above where ?0? represents a specific character value and ?#? a number. The following would represent invalid formats that would need to be setup for the points above:

INVALID_FORMAT (Numeric Only)
00#########0
00##########
0#########00
0##########0
0###########

The above validation will be changed to reference the invalid badge number formats stored on the new table should any exist, however, the badge numbers will still be extracted as described above. If no formats have been setup then any badge number format will be acceptable. Should wildcards be used i.e. ?00%? any id starting with ?00? will be invalid regardless of length.

N.B. It is presumed that a badge number with a character will be rejected should one be detected in any position of the badge number therefore only the numeric format will need to be setup and used to validate the badge number, if

characters may be accepted then each different combination of character and number will need to be stored on the new table for the validation to be performed.

For example, where ?^? represents a character and ?#? a number:

INVALID_FORMAT (Characters Only)
^
^^
^#
#^
^^^
^^#
^#^
^##
^#^
#^^
##^

The same validation may be applied to other related areas of MTS in the future for example where the POD name of the order is updated.

i.e., the ?POD/CMR? number may be entered in the ?Order Tracking?, ?Invoice Debrief?, ?Order Debrief? and ?Trip Debrief? screens and the validation described above will be included for when the POD details are applied in function OMS.APPLY_POD. In this function, the POD name of the order can be updated with the ?POD/CMR? number, the updated with the username or removed; if the user has access to the system function ?ORD_MODIFY_POD? and the ?POD/CMR? number is null then the POD name may be removed if it exists, otherwise if the user does not have access to the system function ?ORD_MODIFY_POD? and the ?POD/CMR? number is null then the username may be used to set the POD name, otherwise if the ?POD/CMR? number is not null then it may be used to set the POD name.

1.3 Scope

This change will be applied to system version 10.4.7 on SARTST and once approved SARPRD.

2 Set-up

2.1 Pre-requisites

The new table ?ADM_INVALID_FORMATS? must be created.

2.2 Menu Structure

Unchanged

2.3 Data

The invalid formats ought to be setup on the new table ?ADM_INVALID_FORMATS?.

3 Functional Description

3.1 Invalid Formats

It is expected that a number of difference formats will be stored as alphanumeric characters up to a maximum of 20 characters in length. The new table will store 2 columns as VARCHAR2(20): ?AREA? (e.g. ?BADGE_NUMBER?) and ?INVALID_FORMAT?. This will enable the table to be used for different items during validation in different processes (VARCHAR2(20) is also the definition of the existing column ?BADGE_NUMBER? on the table ?HHT_INBOUND_DETAIL? so this will be consistent).

The badge numbers are currently validated as follows when the ?DETAIL? line of the HHT inbound message ?HHTtripmil? is uploaded for a trip stop of type ?DL?:

1. The length must not be more than 12 digits
2. The digits must be numeric with values between 0 and 9
3. If the length is 12 digits then it must start and end with ?00? and digits 3 to 9 will be stored
4. If the length is 8 digits then digits 1 to 7 will be stored
5. If the length is not 8 or 12 digits then all digits will be stored

The badge number provided will then be stored as the POD name against the scheduled order when it is updated to status ?DELIVERED?.

For example, ?00########00? would represent a valid format for point 3 above where ?0? represents a specific character value and ?#? a number.

The above validation will be changed to reference the invalid badge number formats stored on the new table should any exist, however, the badge numbers will still be extracted as described above. If no formats have been setup then any badge number format will be acceptable. Note that should any wildcard validation be used, e.g. ?00%?, any badge number starting with ?00? will be invalid regardless of length. The invalid badge number formats will be setup using ?BADGE_NUMBER? as the ?AREA? on the new table; this will allow the new table to store invalid formats for different processes.

N.B. It is presumed that a badge number with a character will be rejected should one be detected in any position of the badge number therefore only the numeric format will need to be setup and used to validate the badge number.

If characters may be accepted then each different combination of character and number will need to be stored on the new table for the validation to be performed.

For example, where ?^? represents a character and ?#? a number:

INVALID_FORMAT (Characters Only)
^
^^
^#
^^^
^^#
^#^
^##
^#^

Each possible combination up to 20 characters in length would need to be setup.

3.2 Validation Process

The invalid formats setup on the new table will be assessed wherever the badge number may be uploaded or entered.

The ?AREA? on the new table will be accessed using ?BADGE_NUMBER? to ensure that the correct invalid formats are found for the validation.

A new function called ?CHECK_INVALID_FORMAT? will be created in package ?ADM?.

The function will receive parameters ?AREA? and ?VALUE? and return a boolean value of ?FALSE? if an invalid format is found or ?TRUE? if not; each parameter will be of type ?VARCHAR2?.

The ?AREA? will be the type of format to validate, e.g. ?BADGE_NUMBER?.

The ?VALUE? will be the string to validate, e.g. ?1234567890?.

If an invalid format for the badge number provided is found using pattern matching then an error message will be displayed in the calling screen or generated for the inbound EDI process:

?The badge number format is invalid?

If an invalid format is not found, or no values are setup, on the new table then the badge number received will be accepted and stored for the order in item ?SCH_ORD.POD_NAME? (after extracting the appropriate digits as described in section 3.1).

The following programs will be changed to use this new validation:

- **HHT inbound message**

The procedure ?INT_MSG.PROC_HHT_DL_DEPART? will be changed to validate the badge number received via ?ADM.CHECK_INVALID_FORMAT? instead of the existing validation (it will be assumed that the invalid formats are setup on the new table).

The parameters passed to the new function will be ?BADGE_NUMBER? and the item ?T_BADGE_NUMBER1?.

- **?Order Debrief? screen**

The entry of fields ?POD Ref? and ?POD/CMR?, for collection and delivery, will be changed to validate the badge number entered via ?ADM.CHECK_INVALID_FORMAT?.

The parameters passed to the new function will be ?BADGE_NUMBER? and the items ?POD_NAME_UPDATE?, ?CD_POD_NAME? or ?DD_POD_NAME?.

- **?Tracking? screen**

The entry of field ?POD/CMR No? will be changed to validate the badge number entered via ?ADM.CHECK_INVALID_FORMAT?.

The parameters passed to the new function will be ?BADGE_NUMBER? and the item ?POD_NAME?.

- **?Trip Debrief? screen**

The entry of field ?POD/CMR No? will be changed to validate the badge number entered via ?ADM.CHECK_INVALID_FORMAT?.

The parameters passed to the new function will be ?BADGE_NUMBER? and the item ?POD_NAME?.

- **?Debrief by Invoice? screen**

The entry of field ?POD/CMR No.? will be changed to validate the badge number entered via ?ADM.CHECK_INVALID_FORMAT?.

The parameters passed to the new function will be ?BADGE_NUMBER? and the item ?POD?.

N.B. At present, a badge number that contains a character will be rejected and the new function will not need to be called; if a character may be valid then further development will be required in the future as it is outside the scope of this development. (See section 3.1 for further information.)

It will still be possible to remove a badge number via the function ?OMS.APPLY_POD?.

Table Updates Required

A new table called ?ADM_INVALID_FORMATS? will be required as below:

Name	Type	Nullable	Default	Storage	Comments
AREA	VARCHAR2(20)	N			
INVALID_FORMAT	VARCHAR2(20)	Y			

The new table may be created using the following script:

References

Ref No	Document Title & ID	Version	Date
1	EST-273276 SA-7Y59ZG Aramco Badge Number Validation v4.0.doc	4.0	15/03/10

Document History

Version	Date	Status	Reason	Initials
0.1	23/08/10	Draft	Initial version	PDR
1.0	26/08/10	Issue	Reviewed and Issued	MJC

4 AUTHORISED BY

Matt Crisford	Development Manager
Peter Greer	TMSCC MTS Product Manager