290930 v1.1

Aptean Ltd

Contents

1 290930

aptean

DHL C-TMS

SMS Pre-advice

FUNCTIONAL SPECIFICATION - 10.7

12/10/2011 - 1.1
Reference: FS 290930 NW-8KENBD

2 Functional Overview

2.1 Client Requirement

Change Request Summary:

Project Rigel - SMS Pre-advice.

Change Request Details:

It is assumed that this RIO will be managed in conjunction with the Project Rigel System Requirements Document v1.0 or higher.

Create an output from C-TMS based on a pre-defined, modifiable template that can be sent to a service provider and delivered to recipients via SMS or e-mail. This will act as pre-advice of an upcoming delivery. Provision of the service provider is also in scope.

Benefits identified as a result of the change:

Defined as part of Project Rigel.

2.2 Solution

A new process will be developed to enable the recipient (i.e. deliver to/collect from location name) of the transport order to be informed via SMS, or an e-mail, of an impending delivery of Baxter orders.

An SMS will be sent by sending a text message in a pre-defined format and a recipient number(s) via a service box for forwarding to the recipient (i.e. deliver/collect from/to location).

Where e-mail notification is required an e-mail will be sent in a pre-defined format to the recipient via the message constructor process.

The format of the text messages and e-mails will conform to a pre-defined format as stored in C-TMS and selected for use for the method of transfer and the stage of its production. A new maintenance tab will be developed to store the pre-defined formats as the data will not be uploaded via an interface.

New medium maintenance and audit tab pages within the ?Message Maintenance? form will be developed to display the message log details sent to recipients via whichever method of transfer (EMAIL/SMS).

The sending of both message types will be made automatically at the setting of the trip status to ?ACCEPTED?.

A restriction of sending a SMS will be included to prevent the delivery of messages within a given time window.

2.3 Scope

This change will be applied to system version 10.7.0.

3 Set-up

3.1 Pre-requisites

The system parameters have been setup and the new tables created via the scripts in Appendix A.

3.2 Menu Structure

?Unchanged?

3.3 Data

The new tables have been created via the scripts in Appendix A.

3.4 Implementation Advice

The SMS and e-mail addresses will be maintained by the development for RIO 291211 NW-8KND7U.

Access to the new tab pages in the ?Messaging Maintenance? screen should be controlled by user groups.

4 Functional Description

4.1 System Parameters

4.1.1 ?SEND_SMS_EMAIL?

A new system parameter called ?SEND_SMS_EMAIL? will be created to determine whether a SMS or an e-mail will be sent to the recipient of the transport order.

The system parameter will permit a control record to be written to the ?MSG_EVENT? database table, by the database trigger described in section 3.5.1, when a trip is updated to status ?ACCEPTED?: if the system parameter is set to ?Y? then a record will be written to the database table.

The system parameter will be set at the cost-centre level for control by project (e.g. ?BA? for ?BAXTER?) and the description of this system parameter will be:

?Controls whether a SMS or an e-mail may be sent to the recipient of the order being delivered on the trip (Y/N)?

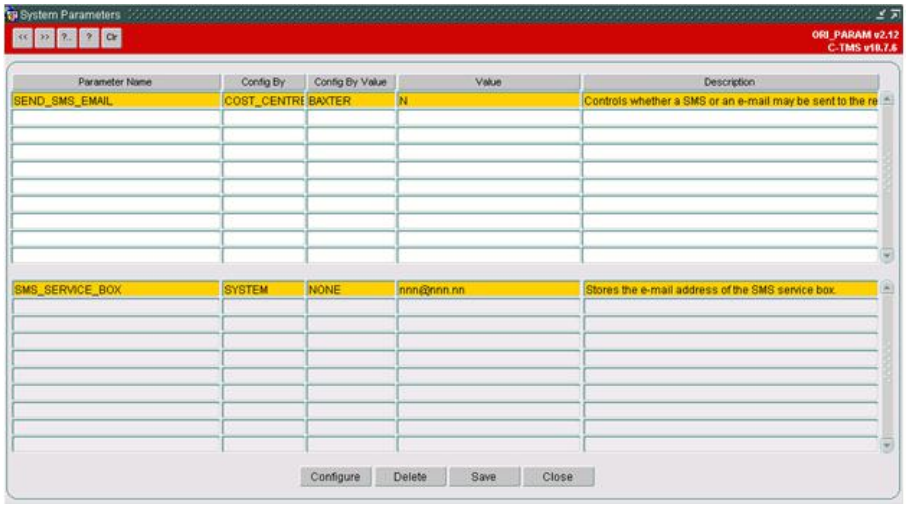
4.1.2 ?SMS_SERVICE_BOX?

A new system parameter called ?SMS_SERVICE_BOX? will be created to store the e-mail address of the service box.

The system parameter will be at the system level and the description of this system parameter will be:

?Stores the address of the SMS service box?

For example:



4.2 Message Event Type

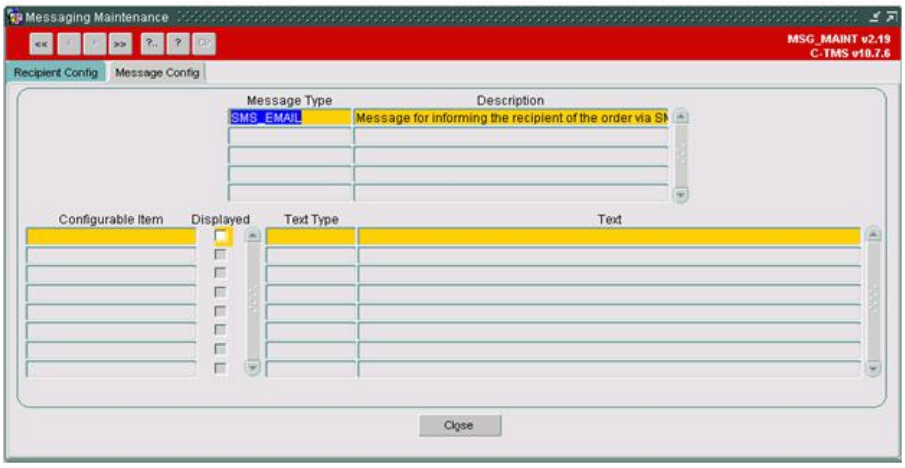
A new message type called ?SMS_EMAIL? will be created without any configurable items. (See Appendix A for the script)

A new message event type called ?SMS_EMAIL? will be created without any configurable items so that the control records may be written to the ?MSG_EVENT? database table. (See Appendix A for the script)

The description of the message type and the event type will be:

?Message for informing the recipient of the order via SMS or e-mail?

For example:



4.3 Database Tables

Three new database tables will be required to store the formats of the messages and the audit records of the messages generated.

The format of the messages has been designed to allow other formats to be used for different messages.

4.3.1 ?MSG_SMS_VALUES? Table

Name	Type	Nullable	Default
MSG_EVENT_TYPE	VARCHAR2(12)	N	
MSG_VALUE	VARCHAR2(12)	N	
MSG_DESC	VARCHAR2(50)	N	
MSG_LEVEL	VARCHAR2(50)	N	
MSG_ITEM	VARCHAR2(50)	N	
MSG_ITEM_FORMAT	VARCHAR2(20)	Y	

An ascending normal index will be added to the table:

1. MSG_EVENT_TYPE
2. MSG_VALUE

Data example:

MSG_EVENT_TYPE	SMS_EMAIL
MSG_VALUE	COST_CENT
MSG_DESC	Trip Cost Centre
MSG_LEVEL	TRIP

MSG_ITEM	COST_CENTRE
MSG_ITEM_FORMAT	

Name	Comments
MSG_EVENT_TYPE	Contains the type of event.
MSG_VALUE	Contains the value for the message format to map the item required.
MSG_DESC	Contains the description of the message value.
MSG_LEVEL	Contains the database level (i.e. table) that will be substituted in the message for the message value.
MSG_ITEM	Contains the database item (i.e. column) that will be substituted in the message for the message value.
MSG_ITEM_FORMAT	Contains the specific format for the item.

The message values may be described below:

Value	Description	Level	Item	Format
COST_CENT	Trip Cost Centre	TRIP	COST_CENTRE	
DEL_DATE	Delivery Date	TRIP_STOP	ARRIVE	DD/MM/YY
START_ARR	Start Arrival Time	TRIP_STOP	ARRIVE	HH24:MI
END_ARR	End Arrival Time	TRIP_STOP	DEPART	HH24:MI
TRIP	Trip ID	TRIP	TRIP_ID	
OMS	OMS Reference	ORDER	OMS_REF	
CUST_REF	Customer Reference	ORDER	EXTERNAL_REF	
DEL_REF	Delivery Point Reference	ORDER	DEL_POINT_REF	
BOOK_REF	Booking Reference	ORDER	BOOKING_REF	

The list above may be extended as other items are required for inclusion in the messages.

The item format is expected to be used to define the date and time formats.

If the level is the same then the values may be interchanged as required because the database item will be obtained at the same level.

N.B. These message values and items for the message event type will be defined and maintained by OBS for the messages required. Therefore, they will not be maintained within C-TMS by users because the appropriate message database items need to be confirmed by OBS. The values and items may be displayed in the ?Medium Maintenance? tab page of the ?Messaging Maintenance? screen as described in section 3.4.1.

4.3.2 ?MSG_SMS_FORMAT? Table

Name	Type	Nullable	Default
MSG_EVENT_TYPE	VARCHAR2(12)	N	
MSG_MEDIUM	VARCHAR2(12)	N	
MSG_TRIGGER	VARCHAR2(20)	N	
MSG_SUBJECT	VARCHAR2(50)	Y	
MSG_FORMAT	VARCHAR2(1000)	N	
RESTRICTION_START	DATE	N	
RESTRICTION_END	DATE	N	

An ascending normal index will be added to the table:

1. MSG_EVENT_TYPE
2. MSG_MEDIUM
3. MSG_TRIGGER

Name	Comments
MSG_EVENT_TYPE	Contains the type of event.
MSG_MEDIUM	Contains the method by which the recipient is contacted, this may be ?SMS?, ?EMAIL? or ?BOTH? should an SMS and an e-mail be requested by the recipient of the order.
MSG_TRIGGER	Contains the trigger that creates the message, this may be ?TRIP_ACCEPTED? for automatic generation via trip processing.
MSG_SUBJECT	Contains the subject description (this is relevant for e-mails only).
MSG_FORMAT	Contains the format of the message.
RESTRICTION_START	Contains the start of the time window within which messages may not be sent to the recipient.
RESTRICTION_END	Contains the end of the time window within which messages may not be sent to the recipient.

Data example:

MSG_EVENT_TYPE	SMS_EMAIL
MSG_MEDIUM	SMS
MSG_TRIGGER	TRIP_ACCEPTED
MSG_SUBJECT	Pre-advice of Delivery
MSG_FORMAT	Your Baxter delivery is due to arrive on {2} between {3}-{4}
RESTRICTION_START	1800
RESTRICTION_END	0800

N.B. The message format setup should not exceed the maximum size of a SMS where the medium is ?SMS?. This is expected to be 147 characters.

The corresponding message value in the message format will be substituted with the value of the message item specified, e.g. <2> will be replaced by the delivery date of the trip stop as may be seen in the ?MESSAGE? in the data example of the message audit record in section 3.3.3.

4.3.3 ?MSG_SMS_AUDIT? Table

Name	Type	Nullable Default
MSG_EVENT_TYPE	VARCHAR2(12)	N
MSG_MEDIUM	VARCHAR2(12)	N
MSG_TRIGGER	VARCHAR2(20)	N
TRIP_ID	VARCHAR2(12)	N
OMS_REF	VARCHAR2(12)	N
MSG_SUBJECT	VARCHAR2(50)	Y
MESSAGE	VARCHAR2(1000)	N
MSG_STATUS	VARCHAR2(20)	N
RECIPIENT_ADDRESS	VARCHAR2(100)	N
CREATED_DATE	DATE	N
CREATED_BY	VARCHAR2(40)	N
UPDATED_DATE	DATE	Y
UPDATED_BY	VARCHAR2(40)	Y

Name	Comments
MSG_EVENT_TYPE	Contains the type of event.
MSG_MEDIUM	Contains the method by which the recipient was contacted.
MSG_TRIGGER	Contains the trigger that created the message.
TRIP_ID	Contains the trip on which the order was assigned.
OMS_REF	Contains the order
MSG_SUBJECT	Contains the subject description of the e-mail sent.

MESSAGE	Contains the message of the SMS and/or e-mail sent.
MSG_STATUS	Contains the status of the message generated (see below).
RECIPIENT_ADDRESS	Contains the SMS number or the e-mail address used.
CREATED_DATE	Contains the date when the message was sent.
CREATED_BY	Contains the user who generated the message.
UPDATED_DATE	Contains the date when the message was reprocessed.
UPDATED_BY	Contains the user who updated the message.

New ascending normal indexes will be added to the table:

Index 1:

1. MSG_EVENT_TYPE
2. MSG_MEDIUM
3. TRIP_ID
4. OMS_REF

Index 2:

1. MSG_EVENT_TYPE
2. MSG_MEDIUM
3. OMS_REF

Index 3:

1. MSG_EVENT_TYPE
2. MSG_MEDIUM
3. CREATED_DATE

Data example:

MSG_EVENT_TYPE	SMS_EMAIL
MSG_MEDIUM	SMS
MSG_TRIGGER	TRIP_ACCEPTED
TRIP_ID	PAR-12345678
OMS_REF	123456
MSG_SUBJECT	Pre-advice of Delivery
MESSAGE	Your Baxter delivery is due to arrive on 04/07/11 between 12:08-16:08
MSG_STATUS	REPROCESSED
RECIPIENT_ADDRESS	07777 123456
CREATED_DATE	05/09/2011 16:34:56
CREATED_BY	SUPERUSER1
UPDATED_DATE	06/09/2011 08:01:23
UPDATED_BY	SUPERUSER2

If the message medium was ?BOTH? then separate audit records will be written for the ?SMS? and the ?EMAIL? provided that sufficient recipient address details were present on the order.

If multiple messages were required by the customer of the transport order because it had multiple phone numbers set, separate audit records will be written for the ?SMS?.

?CREATED_DATE, ?CREATED_BY?, ?UPDATED_DATE? and ?UPDATED_BY? will be updated by the database trigger ?TRG_MSG_SMS_AUDIT_IU? as described in section 3.5.2.

The message status is described below:

Status	Description
PROCESSED	The message was sent successfully to the recipient.
FAILED	The message was sent unsuccessfully to the recipient.
REPROCESSED	The message was re-sent to the recipient.

The message audit record provides visibility to the users in an enquiry screen about whether the messages have been sent successfully to the recipient.

4.4 ?Messaging Maintenance? Screen (?MSG_MAINT? Form)

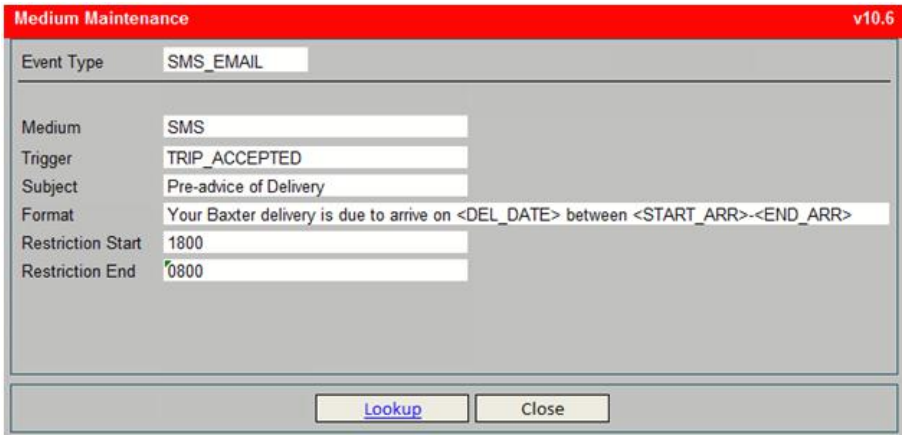
Access to the new tab pages may be controlled by user groups.

These tab pages are designed for maintenance and enquiry purposes, plus there will be the ability to re-process failed messages from the audit records.

4.4.1 Medium Maintenance

A new tab page called ?Medium Maintenance? will be added to the ?Messaging Maintenance? screen to store the valid formats of the SMS and e-mails at the trigger points. (See Appendix A for the script to control access to the new tab page)

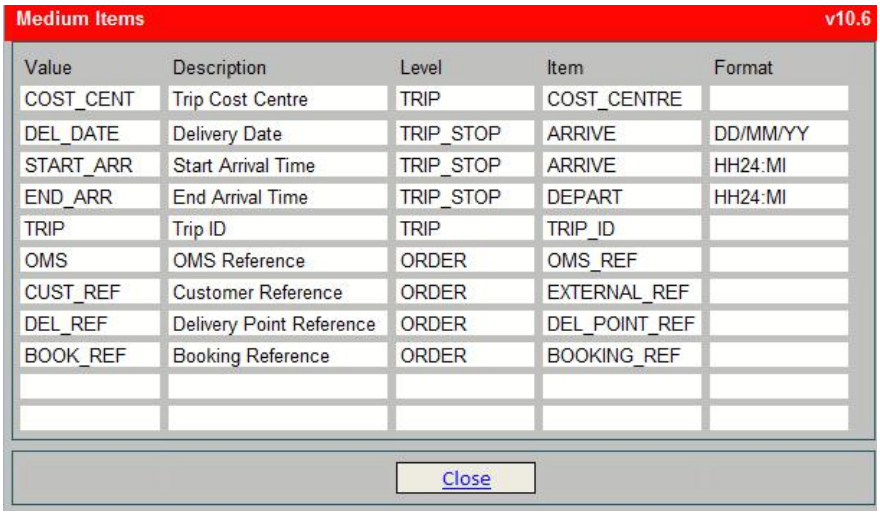
For example:



- Clicking 'Close' will close the screen and return the user to the C-TMS menus.
- Clicking 'Lookup' will call a Medium Items popup screen that will display the items available for inclusion in the message.

4.4.2 Medium Items

A new popup screen called by the ?Lookup? button in the ?Medium Maintenance? screen will be created to display the ?Medium Items? that may be included in the message format:

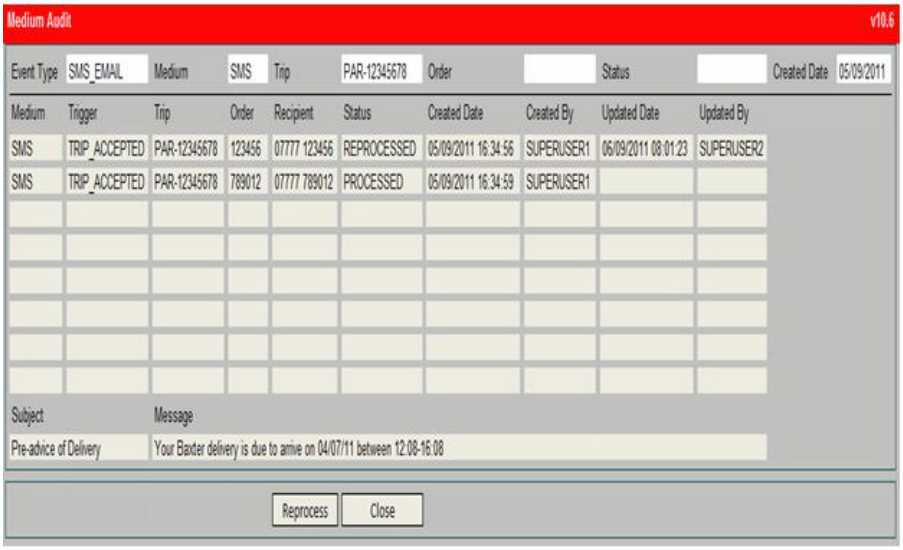


- Clicking 'Close' will close the screen and return the user to the ?Medium Maintenance? screen.

4.4.3 Medium Audit

A new tab page called ?Medium Audit? will be added to the ?Messaging Maintenance? screen to display the messages generated and to re-submit a failed or processed message event for a trip via a button. (See Appendix A for the script)

For example:



- Clicking 'Close' will close the screen and return the user to the C-TMS menus.
- Clicking 'Reprocess' will re-send the 'FAILED' record.

The user will have the option, in a popup screen, to reprocess the trip or the order for the row highlighted provided that the record has a status of ?FAILED?.

Records may be found via the search buttons at the top of the screen: ?Event Type?, ?Medium?, ?Trip?, ?Order?, ?Status? and ?Created Date?.

A vertical scrollbar will be present for the audit data block (i.e. for data in the columns ?Medium? to ?Reprocess?).

Each record in the audit block will display the subject and message in a separate data block at the bottom of the screen for better visibility.

N.B. The audit records may not be edited in this screen.

N.B. A database job may be created to delete the old audit records in the future.

A ?FAILED? message can be reprocessed via the ?Reprocess? button at the bottom of the screen. A new message event for the trip and order will be written as a record to database table ?MSG_EVENT? with the following data:

EVENT_ID	SEQ_MSG_EVENT.NEXTVAL
EVENT_TYPE	?SMS_EMAIL?
EVENT_REF	MSG_SMS_AUDIT.MSG_TRIGGER + MSG_SMS_AUDIT.TRIP_ID + MSG_SMS_AUDIT.OMS_REF
STATUS	?NEW?

4.5 Triggers

4.5.1 ?TIU_TRIP_STATUS?

The existing ?TIU_TRIP_STATUS? trigger on database table ?SCH_TRIP? will be changed to write a record to database table ?MSG_EVENT? with the following data when the status of the trip is updated to ?ACCEPTED?:

EVENT_ID	SEQ_MSG_EVENT.NEXTVAL
EVENT_TYPE	?SMS_EMAIL?
EVENT_REF	?TRIP_ACCEPTED? + SCH_TRIP.TRIP_ID
STATUS	?NEW?

The record will be written if the system parameter ?SEND_SMS_EMAIL? is set to ?Y? for the cost centre of the trip (e.g. ?BA? for ?BAXTER?).

The existing sequence ?SEQ_MSG_EVENT? will be used.

See Appendix A for the script to update this trigger.

4.5.2 ?TRG_MSG_SMS_AUDIT_IU?

A new database trigger will be written for the insert and update actions on new database table ?MSG_SMS_AUDIT?.

Action	Item	Value
INSERTING	CREATED_DATE	SYSDATE
INSERTING	CREATED_BY	USER
UPDATING	UPDATED_DATE	SYSDATE
UPDATING	UPDATED_BY	USER

See Appendix A for the script to create this trigger.

4.6 SMS/E-mail Messaging

If a database job does not exist then a new database job will be created to process the message events generated by the trigger described in section 3.5.1 or the buttons in the ?Message Maintenance? screens.

If only a trip ID is provided in the event reference then all orders on the trip will be processed; however, if a trip ID and an OMS reference is provided (e.g. via reprocessing) in the event reference then only that order on the trip will be processed.

If the order is for a delivery to the destination location (i.e. ?To Location?), or if the order is for a collection from the source location (i.e. ?From Location?), then the message may be sent to the recipient at that location.

The message will be constructed based in the specified format for the SMS or the e-mail required and interfaced to the service box for relay to the recipient?s SMS and/or e-mail address provided.

The recipient address details for the orders will be obtained from the new ?SCH_ORD_INFORMATION? database table:

Title	Column	Example
Message Medium	MSG_MEDIUM	B
SMS Address 1	SMS_ADDRESS_1	07777 123456
SMS Address 2	SMS_ADDRESS_2	07777 789012
SMS Address 3	SMS_ADDRESS_3	
SMS Address 4	SMS_ADDRESS_4	
E-mail Address	EMAIL_ADDRESS	John.Allen@mail.uk

If the ?Message Medium? is ?S? then only a SMS will be sent; if it is ?E? then only an e-mail will be sent; if it is ?B? then both a SMS and an e-mail will be sent; if it is ?N? then no message will be sent.

In this example an SMS will be sent to ?07777 123456? and ?07777 789012? and an e-mail will be sent to ?John.Allen@mail.uk?.

N.B. If a SMS or an e-mail address is not provided then a message will not be sent for that transport order, the processing of the message for the trip will proceed and the status will be updated to ?PROCESSED? upon completion (after having been updated to ?PROCESSING? upon the commencement of the process).

If an exception occurred during processing then the status of the message event will be updated to ?FAILED? as per the standard at present.

If the message medium is ?SMS? then the service box will be accessed and the message generated will be sent for processing via an e-mail, the subject of this e-mail will contain the SMS address, delimited by semi-colons and without spaces, of the transport order for forwarding the message via the service box.

For example:

Subject	Content
07777123456;07777789012	Your Baxter delivery is due to arrive on 04/07/11 between 12:08-16:08

See section 3.1.3 of the functional specification for RIO 291211 NW-8KND7U for further details about the maintenance of transport orders within C-TMS

N.B. Up to 4 SMS addresses (i.e. phone numbers) may be stored on the transport order uploaded from WISE WMS, therefore, the recipient could receive up to 4 SMS to different phone numbers.

If the message medium is ?EMAIL? then an e-mail will be constructed and sent to the e-mail address of the order and the subject of this e-mail will contain the subject of the message format (i.e. ?MSG_SMS_FORMAT.MSG_SUBJECT?) for the trigger point.

4.7 Service Box

A service box called ?outboxsms?, provided and supported by ?Felltech?, will be used to send the messages as a SMS to the recipient of the order.

The service box will have an e-mail address (stored in system parameter ?SMS_SERVICE_BOX?, e.g. ?outboxsms@obs-logistics.com?) for the interface of messages; it will receive and redirect the message to the SMS sent as a number in the subject of the e-mail sent to the service box. There will also be the option to receive a response made as an e-mail for the message by the recipient; this can be performed by the service box because it will store the original e-mail address of the provider of the message.

Table Updates Required

New database tables will be created via the following scripts to store the standard message formats and time window restrictions:

-- Create table create table MSG_SMS_FORMAT (MSG_EVENT_TYPE VARCHAR2(12), MTS_MEDIUM VARCHAR2(12), MSG_TRIGGER VARCHAR2(20), MSG_SUBJECT VARCHAR2(50) nullable, MSG_FORMAT VARCHAR2(1000), RESTRICTION_START DATE, RESTRICTION_END DATE); / -- Grant/Revoke object privileges grant select, insert, update, delete, alter on MSG_SMS_FORMAT to MTS_USER; grant select on MSG_SMS_FORMAT to MTS_USER_READ_ONLY; create public synonym MSG_SMS_FORMAT for MSG_SMS_FORMAT; -- Add a normal index MSG_SMS_FORMAT_IDX1 create index MSG_SMS_FORMAT_IDX1 on MSG_SMS_FORMAT (MSG_EVENT_TYPE, MSG_MEDIUM, MSG_TRIGGER); / -- Create table create table MSG_SMS_VALUES (MSG_EVENT_TYPE VARCHAR2(12), MSG_VALUE VARCHAR2(12), MSG_DESC VARCHAR2(50), MSG_LEVEL VARCHAR2(50), MSG_ITEM VARCHAR2(50), MSG_ITEM_FORMAT VARCHAR2(20) nullable); / -- Grant/Revoke object privileges grant select, insert, update, delete, alter on MSG_SMS_VALUES to MTS_USER; grant select on MSG_SMS_VALUES to MTS_USER_READ_ONLY; create public synonym MSG_SMS_VALUES for MSG_SMS_VALUES; -- Add a normal index MSG_SMS_VALUES_IDX1 create index MSG_SMS_VALUES_IDX1 on MSG_SMS_VALUES (MSG_EVENT_TYPE, MSG_VALUE); CREATE TABLE SCH_ORD_INFORMATION (ID NUMBER NOT NULL,

```
OMS_REF           VARCHAR2(12) NOT NULL
);
```

ALTER table SCH_ORD_INFORMATION

```
 add constraint PK_SCH_ORD_INFORMATION primary key (ID);
```

create index IDX_SOI_OMS on SCH_ORD_INFORMATION (OMS_REF);

grant select,alter on SCH_ORD_INFORMATION to MTS_USER; grant select on SCH_ORD_INFORMATION to MTS_USER_READ_ONLY;

create public synonym SCH_ORD_INFORMATION FOR SCH_ORD_INFORMATION; -- Create sequence create sequence SEQ_SCH_ORD_INFORMATION minvalue 1 maxvalue 999999999999999999999999 start with 1 increment by 1 cache 20;

grant select,alter on SEQ_SCH_ORD_INFORMATION to MTS_USER; grant select on SEQ_SCH_ORD_INFORMATION to MTS_USER_READ_ONLY;

create public synonym SEQ_SCH_ORD_INFORMATION FOR SEQ_SCH_ORD_INFORMATION;

A new database table will be created via the following scripts to store the audit records of the messages generated and sent via SMS or e-mail:

-- Create table create table MSG_SMS_AUDIT (MSG_EVENT_TYPE VARCHAR2(12), MSG_MEDIUM VARCHAR2(12), MSG_TRIGGER VARCHAR2(20), TRIP_ID VARCHAR2(12), OMS_REF VARCHAR2(12), MSG_SUBJECT VARCHAR2(50) nullable, MESSAGE VARCHAR2(1000), MSG_STATUS VARCHAR2(20), RECIPIENT_ADDRESS VARCHAR2(100), CREATED_DATE DATE, CREATED_BY VARCHAR2(40), UPDATED_DATE DATE nullable, UPDATED_BY VARCHAR2(40) nullable); /

-- Grant/Revoke object privileges grant select, insert, update, delete, alter on MSG_SMS_AUDIT to MTS_USER; grant select on MSG_SMS_AUDIT to MTS_USER_READ_ONLY; create public synonym MSG_SMS_AUDIT for MSG_SMS_AUDIT; -- Add a normal index MSG_SMS_AUDIT_IDX1 create index MSG_SMS_AUDIT_IDX1 on MSG_SMS_AUDIT (MSG_EVENT_TYPE, MSG_MEDIUM, TRIP_ID, OMS_REF); -- Add a normal index MSG_SMS_AUDIT_IDX2 create index MSG_SMS_AUDIT_IDX2 on MSG_SMS_AUDIT (MSG_EVENT_TYPE, MSG_MEDIUM, OMS_REF); -- Add a normal index MSG_SMS_AUDIT_IDX3 create index MSG_SMS_AUDIT_IDX3 on MSG_SMS_AUDIT (MSG_EVENT_TYPE, MSG_MEDIUM, CREATED_DATE); /

A new database trigger will be created via the following script to generate and send the messages:

CREATE OR REPLACE TRIGGER TRG_MSG_SMS_AUDIT_IU BEFORE INSERT OR UPDATE ON MSG_SMS_AUDIT REFERENCING NEW AS NEW OLD AS OLD FOR EACH ROW DECLARE BEGIN

```
IF INSERTING THEN
   :new.created_date := SYSDATE;
   :new.created_by := USER;
END IF;
IF UPDATING THEN
   :new.updated_date := SYSDATE;
   :new.update_by := USER;
END IF;
```

END TRG_MSG_SMS_AUDIT_IU; /

A new database trigger will be created via the following script for insert and update actions on the new database table:

CREATE OR REPLACE TRIGGER tiu_trip_status

-- Based on Revision: 5.11

BEFORE INSERT OR UPDATE ON sch_trip REFERENCING NEW AS NEW OLD AS OLD FOR EACH ROW DECLARE

```
v_type       msg_event.event_type%TYPE;
v_type_l     msg_event.event_type%TYPE;
v_mess_count NUMBER;

CURSOR c_message_sent(c_type MSG_EVENT.event_type%TYPE,
                      c_carrier RES_CARRIER.carrier_id%TYPE) IS
SELECT count(*)
FROM v_msg_details a, msg_electronic_address b
WHERE EVENT_REF = :new.sched_name || '-' || :new.trip_id
AND EVENT_TYPE = c_type
AND a.address_id = b.address_id
AND b.recipient_id = c_carrier
AND a.status = 'SENT';

CURSOR c_message_sent_no_carr(c_type MSG_EVENT.event_type%TYPE) IS
SELECT count(*)
FROM msg_event
WHERE EVENT_REF = :new.sched_name || '-' || :new.trip_id
AND EVENT_TYPE = c_type
AND status = 'PROCESSED';

CURSOR c_Lanes IS
SELECT DISTINCT(so.Template_Id) Template_Id
FROM   SCH_Trip_Stop        sts,
       SCH_Haulage_Activity sha,
       SCH_Ord              so,
       LBO_Lane             lbo
WHERE  sts.Sched_Name    = :new.Sched_Name
AND    sts.Trip_Id       = :new.Trip_Id
AND    sha.Stop_Id       = sts.Stop_Id
AND    sha.Oms_Ref       = so.Oms_Ref
AND    so.Template_Id    = lbo.Lane_Id;

CURSOR c_carrier IS
SELECT hub_location
FROM res_carrier
WHERE carrier_id = :new.carrier_id;
```

```
t_hub      VARCHAR2(12);
t_send_sub VARCHAR2(1);

CURSOR c_check_payments IS
SELECT COUNT(1)
FROM acc_payment
WHERE event_ref  = (:new.sched_name || '-' || :new.trip_id)
AND debit_acc  = :new.cost_centre
AND credit_acc = :new.carrier_id;

t_count_payments NUMBER;

CURSOR c_check_desp_bay IS
SELECT 'X'
FROM   geo_location gl, wcs_location wl
WHERE  :new.owning_depot = gl.location_id
AND    gl.rf_xdock = 'Y'
AND    :new.bay_number = wl.location_id
AND    :new.owning_depot = wl.depot
AND    wl.location_type = 'D';

CURSOR c_msg_medium (cp_carrier IN VARCHAR2, cp_type IN VARCHAR2, cp_id IN VARCHAR2) IS
SELECT 1
FROM MSG_TYPE_RECIPIENT_REQ
WHERE RECIPIENT_ID = cp_carrier
AND MSG_TYPE = cp_type AND RECIPIENT_TYPE_ID = cp_id
AND (msg_medium = 'ZETAFAX' OR msg_medium = 'EMAIL');

t_one NUMBER(1);
v_check_desp_bay VARCHAR2(1);
v_check_message_active VARCHAR2(1);
v_ret                  BOOLEAN;
v_errmsg               VARCHAR2(2000);
```

BEGIN

```
IF     :new.trip_status = 'TENDERED'
  AND (:old.trip_status != :new.trip_status
   OR  NVL(:old.carrier_id, 'k' ) != :new.carrier_id) THEN

   OPEN c_message_sent('TRIP_ASGND',:new.carrier_id);
   FETCH c_message_sent into v_mess_count;
   CLOSE c_message_sent;

     IF nvl(v_mess_count,0) >= 1 THEN
        IF NVL(:new.send_amended_email,'Y') = 'Y' THEN
           v_type := 'TRIP_ASGND_A';
        ELSE
           adm.write_msg('MSG','INFO','Send Amended email flag not set for Sched - ' || :new.sched_name |
           ', Trip - ' || :new.trip_id || ' so no Trip Confirmation Amended email has been sent.');
        END IF;
     ELSE
        v_type := 'TRIP_ASGND';
     END IF;
ELSIF  :new.trip_status = 'ACCEPTED'
   AND :old.trip_status IN ('PLANNED', 'TENDERED') THEN

   OPEN c_message_sent_no_carr('TRIP_ACPTD');
   FETCH c_message_sent_no_carr into v_mess_count;
   CLOSE c_message_sent_no_carr;

   IF nvl(v_mess_count,0) >= 1 THEN
      IF NVL(:new.send_amended_email,'Y') = 'Y' THEN
         v_type   := 'TRIP_ACPTD_A';
         v_type_l := 'TRIP_L_ACP_A';
      ELSE
         adm.write_msg('MSG','INFO','Send Amended email flag not set for Sched - ' || :new.sched_name ||
                       ', Trip - ' || :new.trip_id || ' so no Trip Acceptance Amended email has been sent
      END IF;
   ELSE
      v_type   := 'TRIP_ACPTD';
      v_type_l := 'TRIP_L_ACP';
   END IF;

   IF     nvl(ADM.get_param_vchar('EFX_SEND_SUB'),'N') = 'Y'
      AND nvl(:new.efx_send_flag,'N') = 'N'
```


```
            AND nvl(:new.carrier_id,'EFX') <> 'EFX' THEN

            t_send_sub := 'Y';

            OPEN  c_carrier;
            FETCH c_carrier INTO t_hub;
            CLOSE c_carrier;
            IF nvl(t_hub,'+') = nvl(:new.owning_depot,'+') THEN
               t_send_sub := 'N';
            END IF;

            OPEN  c_check_payments;
            FETCH c_check_payments INTO t_count_payments;
            CLOSE c_check_payments;
            IF t_count_payments = 0 THEN
               t_send_sub := 'N';
            END IF;

            IF t_send_sub = 'Y' THEN
               v_ret := INT_MSG.WRITE_TRIP_DTL_MSG(:new.sched_name, :new.trip_id, v_errmsg, 'EFX_TRIP_DTL');
               :new.efx_status  := 'REQNEW';
               :new.efx_message := 'Sub-Contracted Trip has been sent to EFX';
            END IF;
         END IF;

         OPEN  c_msg_medium (:new.carrier_id,'TRIP_ACCEPT', 'CARRIER');
         FETCH c_msg_medium INTO t_one;

         IF c_msg_medium%FOUND THEN
            INSERT INTO msg_event ( event_id
                                  , event_type
                                  , event_ref
                                  , status
                                  , subject
                                  , message )
                           VALUES ( SEQ_MSG_EVENT.nextval
                                  , 'TRIP_ACCEPT'
                                  , :new.trip_id
                                  , 'NEW'
                                  , ADM.Get_Param_VChar('TA_EMAIL_SUBJECT')
                                  , ADM.Get_Param_VChar('TA_EMAIL_TEXT'));

         END IF;
         CLOSE c_msg_medium;

         -- Generate a SMS/email for Baxter orders
         IF nvl(ADM.get_param_vchar('SEND_SMS_EMAIL', :NEW.TRIP_ID),'N') = 'Y' THEN
           INSERT INTO MSG_EVENT (  EVENT_ID,
```

EVENT_TYPE, EVENT_REF, STATUS)

```
                                  VALUES (  SEQ_MSG_EVENT.NEXTVAL,
```

'SMS_EMAIL', :NEW.TRIP_ID, 'NEW');

```
            END IF;
         ELSIF  :new.trip_status = 'CONFIRMED'
            AND :old.trip_status <> 'CONFIRMED'
            AND nvl(ADM.get_param_vchar('EFX_SEND_SUB'),'N') = 'Y'
            AND nvl(:new.carrier_id,'EFX') <> 'EFX'
            AND nvl(adm.get_param_vchar('EFX_FORMAT'),'CSV') = 'XML' THEN

            t_send_sub := 'Y';

            OPEN  c_carrier;
            FETCH c_carrier INTO t_hub;
            CLOSE c_carrier;
            IF nvl(t_hub,'+') = nvl(:new.owning_depot,'+') THEN
               t_send_sub := 'N';
            END IF;

            OPEN  c_check_payments;
            FETCH c_check_payments INTO t_count_payments;
            CLOSE c_check_payments;
            IF t_count_payments = 0 THEN
               t_send_sub := 'N';
            END IF;
```


```
            IF t_send_sub = 'Y' THEN
               v_ret := INT_MSG.WRITE_TRIP_DTL_MSG(:new.sched_name, :new.trip_id, v_errmsg, 'EFX_TRIP_DTL');
               v_ret := INT_MSG.WRITE_TRIP_DTL_MSG(:new.sched_name, :new.trip_id, v_errmsg, 'EFX_XML_DEB');
            END IF;
         ELSIF  :new.trip_status = 'EN-ROUTE'
            AND :old.trip_status != :new.trip_status
            AND NVL(ADM.get_param_vchar('WCS_ENABLED'),'N') = 'Y' THEN

            OPEN  c_check_desp_bay;
            FETCH c_check_desp_bay INTO v_check_desp_bay;
            IF c_check_desp_bay%FOUND THEN

               dp_wcs_if.pr_check_message_active ('403', 'QMC', v_check_message_active);

               IF v_check_message_active = 'Y' THEN
                  INSERT INTO WCS_OUT_CONTROL (MESSAGE_TYPE,
                                               TRIP_ID,
                                               DEPOT)
                  VALUES                      ('403',
                                               :new.trip_id,
                                               :new.owning_depot);
               END IF;
            END IF;
            CLOSE c_check_desp_bay;

            dp_wcs_if.pr_check_message_active ('401', 'QMC', v_check_message_active);

            IF v_check_message_active = 'Y' THEN
               INSERT INTO WCS_OUT_CONTROL (MESSAGE_TYPE,
                                            TRIP_ID)
               VALUES                      ('401',
                                            :new.trip_id);
            END IF;

            dp_wcs_if.pr_check_message_active ('432', 'QMC', v_check_message_active);

            IF v_check_message_active = 'Y' THEN
               INSERT INTO WCS_OUT_CONTROL (MESSAGE_TYPE,
                                            TRIP_ID)
               VALUES                      ('432',
                                            :new.trip_id);
            END IF;
         END IF;

         IF v_type IS NOT NULL THEN
            INSERT INTO msg_event ( event_id
                                  , event_type
                                  , event_ref
                                  , status )
                           VALUES ( SEQ_MSG_EVENT.nextval
                                  , v_type
                                  , :new.sched_name||'-'||:new.trip_id
                                  , 'NEW' );

            IF v_type IN ('TRIP_ACPTD','TRIP_ACPTD_A') THEN
               FOR v_Lanes IN c_Lanes LOOP
                   INSERT INTO Msg_Event ( event_id
                                         , event_type
                                         , event_ref
                                         , status )
                                  VALUES ( SEQ_MSG_EVENT.nextval
                                         , v_type_l
                                         , :new.sched_name||'-'||:new.trip_id||'/'||v_Lanes.Template_Id
                                         , 'NEW' );
               END LOOP;
            END IF;
         END IF;
```

END TIU_TRIP_STATUS; /

User access to the new ?Messaging Maintenance? tab pages will be created via the following scripts:

INSERT INTO ADM_FORM_TAB (FORM_NAME,TAB_NAME,DESCRIPTION) VALUES
('MSG_MAINT','SMS_MAINT','Maintenance');

INSERT INTO ADM_FORM_TAB (FORM_NAME,TAB_NAME,DESCRIPTION) VALUES ('MSG_MAINT','SMS_ENQ','Audit'); /

INSERT INTO ADM_GROUP_FORM_TAB (FORM_NAME,TAB_NAME,GROUP_NAME) VALUES ('MSG_MAINT','SMS_MAINT','ADMIN');

INSERT INTO ADM_GROUP_FORM_TAB (FORM_NAME,TAB_NAME,GROUP_NAME) VALUES ('MSG_MAINT','SMS_ENQ','ADMIN'); /

Two new system parameters will be created via the following script to control the SMS messaging:

INSERT INTO ADM_SYSTEM_PARAM (PARAM_NAME,VALUE,DATA_TYPE,MAX_LENGTH,DISPLAYED,USER_MODIFIABLE,DESCRIPTION,CREATED_BY,CREA
VALUES ('SEND_SMS_EMAIL','N','S',1,'Y','Y','Controls whether a SMS or an e-mail may be sent to the recipient of the order being delivered on the trip (Y/N).','OBS',SYSDATE,'OBS',SYSDATE,'COST_CENTRE','BAXTER');

INSERT INTO ADM_SYSTEM_PARAM (PARAM_NAME,VALUE,DATA_TYPE,MAX_LENGTH,DISPLAYED,USER_MODIFIABLE,DESCRIPTION,CREATED_BY,CREA
VALUES ('SMS_SERVICE_BOX','outboxsms@obs-logistics.com','S',100,'Y','N','Stores the e-mail address of the SMS service box.','OBS',SYSDATE,'OBS',SYSDATE,'SYSTEM','NONE'); /

A new record will be written to the ?MSG_TYPE? database table via the following script:

INSERT INTO MSG_TYPE (MSG_TYPE,MSG_TYPE_DESC,CONCAT) VALUES ('SMS_EMAIL','Message for informing the recipient of the order via SMS or e-mail','N'); /

A new record will be written to the ?MSG_EVENT_TYPE? database table via the following script:

INSERT INTO MSG_EVENT_TYPE (EVENT_TYPE,EVENT_TYPE_DESC) VALUES ('SMS_EMAIL','Message for informing the recipient of the order via SMS or e-mail'); /

Modules to be changed

Module Name	Module Type	Notes
MSG_MAINT.fmb	Form	New tab pages.
MSG_PROCESSING.sql	Package	New procedure.
TIU_TRIP_STATUS.sql	Trigger	New trigger event.
TRG_MSG_SMS_AUDIT_IU.sql	Trigger	New trigger for audit trail.

References

Ref No	Document Title & ID	Version	Date
1	EST-290930 NW-8KENBD SMS Pre-advice v1.0.doc	1.0	26/08/11

Glossary

Term or Acronym	Meaning

C-TMS	Calidus TMS
SMS	Short Message Service
CSV	Comma Separated Values
XML	Extensible Markup Language

Document History

Version	Date	Status	Reason	Initials
0.1	07/09/11	Draft	Initial version	PDR
0.2	20/09/11	Draft	Updated RECIPIENT_ADDRESS to VARCHAR2(100) for e-mail addresses and added up to 4 SMS addresses per transport order.	PDR
0.3	22/09/11	Draft	Review of the initial draft - returned to PDR with comments and questions	PJH
0.4	22/09/11	Draft	Updated	PDR
0.4	23/09/11	Draft	Review of v0.4	PJH
1.0	23/09/11	Issue	Issued to client	PJH
1.1	12/10/11	Draft	Updated after development review.	PDR

5 AUTHORISED BY

Matt Crisford	Development Manager
Peter Greer	TMSCC MTS Product Manager