**The SAS CMS Project**

**Integration Functional Reference**

*CMS\_live\_feed CARMUSR*

*Document Type: Delivery Document*

*Date of Delivery: 20 March 2013*

*Document Status: Delivered FAT*

*Version: 1.18*

*Last Changed: 20 March 2013*

*Owner: Jeppesen*

Table of Contents

[1 Introduction 3](#_Toc346891142)

[1.1 This Document 3](#_Toc346891143)

[1.2 Change History 3](#_Toc346891144)

[2 Functional Reference Integration 5](#_Toc346891145)

[2.1 Request / Reply 5](#_Toc346891146)

[2.2 Live feeds 8](#_Toc346891147)

[2.3 Other subscriptions 11](#_Toc346891148)

[2.4 Publish 12](#_Toc346891149)

[2.5 Jeppesen Crew Portal 19](#_Toc346891150)

[2.6 Batch jobs 22](#_Toc346891151)

[3 Technical reference 24](#_Toc346891152)

[3.1 Technical summary 24](#_Toc346891153)

# Introduction

## This Document

This document contains the functional reference for the SAS CMS2 CARMUSR Integration. References are made to the Interface specifications on Project Place.

## Change History

|  |  |  |
| --- | --- | --- |
| Version | Release Date | Description |
| 1.0 | 23 November 2007 | The document contains a description of the integration and how it is configured. |
| 1.1 | 28 November 2007 | Updated info about mail handling for Meal Orders and Hotel bookings, and some general mail setup description, in setup chapter. |
| 1.2 | 03 October 2008 | Various updates |
| 1.3 | 07 October 2008 | Minor updates, CrewMeal Forecast. |
| 1.4 | 12 November 2008 | Added information about request/reply channel configuration |
| 1.5 | 2 February 2009 | Added flight filtering (CR176) |
| 1.6 | 24 February 2009 | Joined message flows (CR180).  Changed location of DIG configuration files. |
| 1.7 | 28 May 2009 | Added CMD\_INSTALL\_BIDS, CMD\_AG\_LOOP\_TIME\_MONITOR, CMD\_SERVER\_MONITOR, JOB\_CREWLANDINGS |
| 1.8 | 03 Jun 2009 | Corrected flt\_schedule and flt\_movements to flt\_planning and flt\_execution.  Corrected some errors in chapter 3. |
| 1.9 | 03 Jun 2009 | Updates according to template |
| 1.10 | 14 Jan 2010 | Updated rundates for Overtime / List12 / List9 / Instuctor allowance / Perdiem.  Added missing jobs. |
| 1.11 | 5 Feb 2010 | Recovered info about changed procedure for unknown crew (HR-sync).  Added info about X4 configuration implemented for CR411. |
| 1.12 | 14Jun2010 | Update to reflect changes to runtimes, and missing jobs. CMP accumulators has been split to its own job, with statistics gathering. |
| 1.13 | 15Sep2012 | Updates after migration from IB2 to IB5. Note that the non IB5 specific sections are still not migrated to CMS2 (e.g. desmond to sysmond). |
| 1.14 | 10Oct2012 | Updates for CMS2. |
| 1.15 | 18Oct2012 | Updated to reflect change regarding database truncate task scheduling. |
| 1.16 | 29Oct2012 | Revised nightly clean up and truncate job. |
| 1.17 | 25Jan2013 | Updated with Interbids vacation bidding changes. |
| 1.18 | 27Feb2013 | Updated section about interfaces |
| 1.19 | 20Mar2013 | General Review |

# Functional Reference Integration

## Request / Reply

R1 GetReportList

32.3.2 Get Crew Act List (R2 GetReport)

32.3.3 Get Flight Leg (R2 GetReport)

32.3.7 Get Simulator Acts (R2 GetReport)

32.3.9 Get Crew Accum Info (R2 GetReport)

32.13.2 Overtime Presentation (R2 GetReport)

32.13.3 Crew Slip (R2 GetReport)

32.14 Duty Calculation

32.15 Future Activities

32.17 Vacation and Balance (R2 GetReport)

32.21 Compensation Days presentation (R2 GetReport)

CrewBasicService

CrewFlightService

CrewListService

CrewRosterService

This interface, configured in DIG channel crewservices, reads CQFREQ, and from the request data, dispatches the right report. Reply is send back to the reply queue, stated in the request.

The crewservices channel uses a caching mechanism to improve overall performance and to guarantee the correct function of the checkin/checkout report. Out of multiple identical requests submitted close in time, only the first one is forwarded to the report server. The reply from the report server is returned by DIG and re-used for subsequent requests within the configurable cache timeout period. The cache timeout must be at least as long as the report server refresh interval (also configurable). The longest time (worst case) for a model change to be reflected by a request/reply report is equal to the cache timeout plus the refresh interval.

### Services

These services also updates information in the database.

32.1 Crew check-in/out + verification

Handles check in and check out messages from CrewServices.

46.1 Landings

Handles all landings messages from Hermes.

### Technical summary

#### Rave

References to modules used, some important or difficult rave variables can be described.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
| salary | crc/modules | Rave module which contains definitions used in salary reports. |
| %salary\_month\_start\_p% | crc/modules/salary | Start of salary month, used in DUTYOVERTIME report. |
| %salary\_month\_end\_p% | crc/modules/salary | Start of salary month, used in DUTYOVERTIME report. |
| report\_crewlists | crc/modules | Rave module which contains definitions used in reports, from lib/python/crewlists |
| %crew\_empno% | crc/modules/report\_crewlists | Employee number, used in various reports. |
| %released\_until% | crc/modules/report\_crewlists | High date indicating to which date the roster is published. Used in CrewRoster, (CrewServices) to limit when crew can see their roster. |
| %sort\_key\_1% | crc/modules/report\_crewlists | First sort criteria for CrewLists. |
| %crew\_seniority\_pp\_start% | crc/modules/report\_crewlists | Second sort criteria for CrewLists. |
| %is\_pre\_act% | crc/modules/report\_crewlists | Defines which activities to show in futureactivities report. |

#### Database Tables

References to tables used. (u) means that table is being updated by the interface.

**Note!** Technical details should be documented in the code.

| Name | Description |
| --- | --- |
| account\_entry | COMPDAYS, VACATION |
| account\_baseline | COMPDAYS, VACATION |
| crew\_employment | COMPDAYS, VACATION; CheckInOut |
| leave\_entitlement | VACATION |
| crew\_landing | crewlanding (u) |
| cio\_event | CheckInOut (u) |
| cio\_status | CheckInOut (u) |
| crew\_notification | CheckInOut |
| crew | CheckInOut |
| notification\_assignment | CheckInOut |
| notification\_message | CheckInOut |
| crew\_document | CheckInOut |
| flight\_message | CheckInOut |
| flight\_leg\_message | CheckInOut |

#### Scripts

References to scripts used.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
| elements.py | lib/python/crewlists | General module used by all reports which have replyBody as XML root element. |
| Xhandlers.py | lib/python/dig | Various utilities for DIG handlers. Dynamic ReportServer selection, AddressInjector etc. |

#### Reports

References to reports used.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
| rs\_getreportlist.py | lib/python/report\_sources/report\_server | R1 ReportServer interface |
| rs\_getreport.py | lib/python/report\_sources/report\_server | R2 ReportServer interface |
| rs\_crewbasic.py | lib/python/report\_sources/report\_server | CrewBasic ReportServer interface |
| rs\_crewflight.py | lib/python/report\_sources/report\_server | CrewFlight ReportServer interface |
| rs\_crewlist.py | lib/python/report\_sources/report\_server | CrewList ReportServer interface |
| rs\_crewroster.py | lib/python/report\_sources/report\_server | CrewRoster ReportServer interface |
| rs\_crewroster.py | lib/python/report\_sources/report\_server | CrewRoster ReportServer interface |
| rs\_crewlanding.py | lib/python/report\_sources/report\_server | Crew Landing ReportServer interface |
| rs\_dutycalculation.py | lib/python/report\_sources/report\_server | DutyCalculation ReportServer interface |
| rs\_crewbaggage.py | lib/python/report\_sources/report\_server | Crew Baggage Reconciliation |
| rs\_crew\_manifest.py | lib/python/report\_sources/report\_server | Crew Manifest |
| rs\_job\_crewlandings.py | lib/python/report\_sources/report\_server | Landings for A/C without ACARS. |
| rs\_loadsheet.py | lib/python/report\_sources/report\_server | Number of Crew to Load Sheet |
| rs\_servicesarchiver.py | lib/python/report\_sources/report\_server |  |
| getreportlist.py | lib/python/crewlists | R1 interface report. |
| getreportlistCMP.py | lib/python/crewlists | Special version of R1 interface report, used when only CMP is in production, and until CCT is in production. |
| getreport.py | lib/python/crewlists | R2 interface report. |
| crewbasic.py | lib/python/crewlists | CrewBasic interface report. |
| crewflight.py | lib/python/crewlists | CrewFlight interface report. |
| crewlist.py | lib/python/crewlists | CrewList interface report. |
| crewroster.py | lib/python/crewlists | CrewRoster interface report. |
| crewlanding.py | lib/python/crewlists | 46.3 Crew Landings from Hermes |
| dutycalculation.py | lib/python/crewlists | 32.14 Dutycalculation interface report. |
| futureactivities.py | lib/python/crewlists | 32.15.1 Future activities (PRE) interface report. |
| PILOTLOGCREW.py | lib/python/crewlists/subreports | 32.3.2 Pilot Log Crew Activity report (R2). |
| PILOTLOGFLIGHT.py | lib/python/crewlists/subreports | 32.3.3 Pilot Log Crew Flight report (R2). |
| PILOTLOGSIM.py | lib/python/crewlists/subreports | 32.3.7 Pilot Log Crew Simulator report (R2). |
| PILOTLOGACCUM.py | lib/python/crewlists/subreports | 32.3.9 Pilot Log Crew Accumulated report (R2). |
| CREWSLIP.py | lib/python/crewlists/subreports | 32.13.3 CrewSlip report (R2). (Published roster) |
| DUTYOVERTIME.py | lib/python/crewlists/subreports | 32.13 Duty Overtime report (R2). |
| VACATION.py | lib/python/crewlists/subreports | 32.17 Vacation balance report (R2). |
| COMPDAYS.py | lib/python/crewlists/subreports | 32.21 Compensation days report (R2). |
| BOUGHTDAYS.py | lib/python/crewlists/subreports |  |
| DUTYCALC.py | lib/python/crewlists/subreports |  |
| DUTYOVERTIME.py | lib/python/crewlists/subreports |  |

### Crew Services Archiver

The archiver interface services\_archiver stores responses from crew services requests to be able to display them later on. The period which will be archived is the previous month.

|  |  |
| --- | --- |
| Request | Scheduled start time |
| DUTYOVERTIME | 10th every month at 01:59 |
| DUTYCALC | 10th every month at 01:29 |
| CREWROSTER | 10th every month at 02:29 |
| CREWLIST | 11th every month at 01:59 |
| PILOTLOGCREW | 12th every month at 01:59 |
| PILOTLOGFLIGHT | 12th every month at 03:59 |
| PILOTLOGSIM | 12th every month at 03:59 |

## Live feeds

### 36 – SSIM / ASM / SSM

36.1 Long Term Flight Schedule (SSIM)

36.2.1 Short Term Flight Schedule changes (SSM)

36.2.2 Ad Hoc Flight Schedule changes (ASM)

It handles changes to timetable, from TDB.

This is one interface, configured in DIG channels flt\_planning and flt\_execution to read MQ queues CQFTITA and CQFMVTD respectively. Both channels can handle SSIM/ASM/SSM as well as MVT/ROT messages transparently. This allows satisfying message ordering dependencies by routing dependent messages to the same queue.

### 37 – MVT / ROT

37.1 Aircraft Rotations, Initial Load

37.2 Flight Movements

37.3 Aircraft Rotation changes

37.4 Schooling and Training Flights

This includes all movements (flights taking off and landing) and aircraft rotations initial load and changes, from OPUS.

### DIGXML

16.2 Update slot time

32.2 Operational info

41.3 Passenger Prognosis, Short Term

This is configured as one channel, digxml, which reads MQ queue CQFDIGX, with several interfaces, which handles update of slot times from Flow, Operational info (flight messages) from SAS AAP pages, passenger prognosis changes from Level.

### Flight filter

The Flight filter function prevents irrelevant flight from coming into CMS. The incoming feed of ASM/SSM/SSIM messages may contain e.g. truck transports and flights of foreign carriers, some of which are not relevant to the SAS CMS system. Flights with service type code (STC) equal to ‘V’or ‘U’ are filtered out independently from any other filter condition. Flights to be included are those matching any of the following criteria:

1. If one of its legs already exist in the database (could have been created manually).
2. If carrier and flight number matches contents of database table flight\_filter. In this table one or more valid ranges of flight numbers can be configured for each carrier. Note that if filter settings are changed, dig channel flt\_planning needs to be restarted in order to activate the new settings.

Table flight\_filter (example):

CARRIER\_CODE FROM\_FLTNO TO\_FLTNO

--------------------------------------------------------------------------------------

QI 2000 2900

LH 100 999

1. If the flight leg has a cabin crew employer (CAE) or cockpit employer (CPE) that matches any of those specified in table employer\_filter. Note that if filter settings are changed, dig channel flt\_planning needs to be restarted in order to activate the new settings.

Table employer\_filter (example):

EMPLOYER

------------------------------------------------------------

SK

BUL

SKD

BU

SKN

SKS

### Technical summary

#### Rave

References to modules used, some important or difficult rave variables can be described.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
|  |  |  |

#### Database Tables

References to tables used. (u) means that table is being updated by the interface.

**Note!** Technical details should be documented in the code.

| Name | Description |
| --- | --- |
| flight\_leg | opusxmlparser (u) |
| flight\_leg\_delay | opusxmlparser (u) |
| crew\_landing | opusxmlparser (u) |
| aircraft | opusxmlparser (u) |
| aircraft\_flight\_duty | opusxmlparser (u) |
| rotation | opusxmlparser (u) |
| flight\_filter | Xhandlers |

#### Scripts

References to scripts used.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
| opusxmlparser.py | lib/python/dig | Opus MVT and ROT msg parser. |
| ssimp.py | current\_carmsys\_cct /lib/python/carmensystems/dig/messagehandlers | SSIM, SSM and ASM msg parser, main module |
| ssimparsers.py | current\_carmsys\_cct/lib/python/carmensystems/dig/support | SSIM, SSM and ASM msg parser, helper module |
| xhandlers.py  isValidFlight | lib/python/dig | isValidFlight is a FlightFilter to filter flight’s of interest. Used by OpusSsimDispatcher |
| dispatchers.py | lib/python/dig | OpusSsimDispatcher |

#### Reports

References to reports used.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
| aircrew.py | current\_carmsys\_cct/lib/python/carmensystems/dig/messagehandlers | CrewAttacher used to copy crew assignments to new flight legs, RTR or RRT messages. |

## Other subscriptions

* 1. **Passenger Actuals**

This interface, configured in DIG channel ldm, reads MQ queue CQF402 and updates flight\_leg\_pax.

**38 Meal Flight Leg Ownership**

This interface, configured in DIG channel flt\_ownership, reads a file from FIA, with information about flight leg ownership, and updates meal\_flight\_owner.

When a FIA file (FIA.\*) arrives to $CARMTMP/ftp/in, it is read by this interface. When processed, the file is moved to $CARMTMP/ftp/imported

**45.2 Currency, Exchange Rates**  
This interface, configured in DIG channel currency, reads currency files and updates exchange\_rate. When a currency file (CUR\*) arrives to $CARMTMP/ftp/in, it is read by this interface. When processed, the file is moved to $CARMTMP/ftp/imported

### HR synchronisation

* 1. Crew Info Synchronization. This interface, configured in DIG channel crewinfo, receives crew personal info and crew address info messages from HR on MQ queue CQF436 and updates tables crew, crew\_address, crew\_extra\_info, crew\_contact and crew\_relatives.

If new crew or crew who has changed base (which means crew get a new external empno), is received, and thereby not found in crew\_employment, these msg’s are put in a special table, called crew\_unknown. You then have to update crew employment table with the new external empno, and if it is a complete new crew, you also have to insert this new crew in the crew table. When done, you mark the crew ‘corrected’ in crew\_unknown. At the next Crew Info Synchronization run, if successfully imported, the crew will be removed from crew\_unknown.

### Technical summary

#### Rave

References to modules used, some important or difficult rave variables can be described.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
|  |  |  |

#### Database Tables

References to tables used. (u) means that table is being updated by the interface.

**Note!** Technical details should be documented in the code.

| Name | Description |
| --- | --- |
| flight\_leg\_pax | ldmparser (u) |
| exchange\_rate | currencyparser (u) |
| crew | crewinfoparser (personal info) |
| crew\_extra\_info | crewinfoparser (personal info) |
| crew\_contact | crewinfoparser (personal info and address info) |
| crew\_relatives | crewinfoparser (address info) |
| crew\_address | crewinfoparser (address info) |
| Meal\_flight\_owner | meal flight ownership |

#### Scripts

References to scripts used.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
| ldmparser.py | lib/python/dig | Load message parser |
| fiaparser.py | lib/python/dig | Meal Flight Owner information parser |
| currencyparser.py | lib/python/dig | Currency file parser |
| crewinfoparser.py | lib/python/dig | HR synchronization parser |

#### Reports

References to reports used.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
|  |  |  |

## Publish

These are configured to be run by the scheduler

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Interface name** | **DIG channel** | **Run date** | **Run time UTC** | **MQ queue / MAIL** | **Description** |
| 32.4 Meal Orders | meal | Daily | 03:00 | MAIL,  MQ | Generate meal orders |
| 32.4 Meal Orders, Forecast | meal | 20nd every month | 00:00 | MAIL | Generate meal forecast |
| Meal Orders, Update | meal | Every 10th minute | \*0:00 | MAIL, MQ | Generate meal updates |
| 33.4 Hotel Bookings | hotel | Daily | 00:00 | MAIL | Generate hotel bookings |
| 33.4 Hotel Bookings Update | hotel | Hourly | \*:30 | MAIL | Generate hotel bookings update |
| 33.4 Hotel Bookings, Forecast | hotel | 17 every month | 00:10 | MAIL | Generate hotel bookings forecast |
| 33.4 Hotel Bookings, Performed | hotel | 2nd every month | 00:05 | MAIL | Generate hotel bookings performed |
| 33.4 Transport Bookings | hotel | Daily | 00:05 | MAIL | Generate transport bookings |
| 33.4 Transport Bookings Update | hotel | Hourly | \*:35 | MAIL | Generate transport bookings update |
| 33.4 Transport Bookings, Performed | hotel | 2nd every month | 00:00 | MAIL | Generate transport bookings performed |
| 42 Passive Bookings | passive | Daily | 03:00 | CQT42 | Generate passive bookings report |
| X1 | x1 | Daily | 01:00 | CQTX1 | Replicate perkey and name |
| X3 | x3 | Daily | 02:00 | CQTX3 | Vacation trade snapshot |
| X4 Interbids | x4 | Daily | 01:00 | - | PAC and CAS data to Interbids |

32.4 Meal Orders / Forecasts, mails are send to suppliers, defined by email in mail\_supplier entity, and to customers, defined by email in mail\_customer entity.

32.4 Meal Orders, CARI information is put on MQ queues CQT324A and CQT324B.

33.4 Hotel Bookings, mail are send to hotels, defined by email in hotel entity, and to hotel customers (SAS regions), and defined by email in hotel\_customer.

42 Passive Bookings – Update reports are not scheduled but triggered by Studio save operation.

X1, X3 and X4 – The last generated report is available at $CARMTMP/ftp/out.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Interface name** | **DIG channel** | **Trigger condition** | **Check condition** | **MQ queue / FTP** | **Description** |
| Crew Manifest AE PDF | manifest\_pdf | Flights 5 days ahead with destination/departure Country=’AE’ | Every day at 4 PM UTC | FILE | Generate Crew manifest for AE |
| 33.8 Crew Manifest US | manifest\_dep | 60 min to departure, and destination Country=’US’ | every 2 min. | FILE | Generate Crew manifest for USA |
| 33.9 Crew Manifest CN | manifest\_dep | 30,60 mins to departure, and destination/departure Country=’CN’ | every 2 min. | FILE | Generate Crew manifest for China |
| 33.9 Crew Manifest CN | manifest\_arr | 120 mins to arrival, and destination Country=’CN’ | every 2 min. | FILE | Generate Crew manifest for China |
| 33.9 Crew Manifest CN PDF | manifest\_pdf | Flights 5 days ahead with destination/departure Country=’CN’ | Every day at 7 AM UTC. | FILE | Generate Crew manifest for China |
| Crew Manifest GB | manifest\_sched | 60 minutes before scheduled departure/destination country=’GB’ | Every 2 min | FILE | Generate Crew manifest for GB |
| Crew Manifest GB | manifest\_actual | 30 minutes after actual departure/destination country=’GB’ | Every 2 min | FILE | Generate Crew manifest for GB |
| Crew Manifest IN | manifest\_dep | 30 min to departure and destination=’IN’ | Every 2 min. | FILE | Generate Crew manifest for IN |
| CR12 Crew Manifest JP | manifest\_dep | 30 min to departure, and destination Country=’JP’ | every 2 min. | FILE | Generate Crew manifest for JP |
| CR12 Crew Manifest JP PDF | manifest\_pdf | lights 5 days ahead with destination/departure country=’JP’ | Every day at 4 PM UTC | FILE | Generate Crew manifest for JP |
| CR26 Crew Manifest TH | manifest\_dep | 30 min to departure, and departure or destination Country=’TH’ | Every 2 min. | FILE | Generate Crew manifest for TH |
| CR26 Crew Manifest TH PDF | manifest\_pdf | Flights 5 days ahead with destination/departure country=’TH’ | Every day at 4 PM UTC | FILE | Generate Crew manifest for TH |
| 40.1 Crew Baggage Reconciliation | Baggage | 6 hours before departure | every 2 min. | CQT401 | Inform PCI about crew schedule on today date. |
| 41.1 Number of Crew to Load Sheet | loadsheet | 60,22 min to departure | every 2 min. | CQT411 | Inform PAH about actual crew figures. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Interface name** | **DIG channel** | **Run date** | Run time UTC | FTP/ File | Description |
| 43.2 Ambi List | salary | 4th every month | 04:00 | FTP | Generates logical reports with prefix: **AMBI\_DK\_** |
| 43.5.1.1 Vacation days performed (DK) | salary | 6th every month | 03:00 | FTP | Generates logical reports with prefix: **VACATION\_P\_DK\_** |
| 43.5.1.1.2 Vacation days remaining (DK) | salary | 6th every month | 03:00 | FTP | Generates logical reports with prefix: **VACATION\_R\_DK\_** |
| 43.5.2 Vacation days/year account (DK) | salary | 2nd of June every year | 03:00 | FTP | Generates logical reports with prefix: **VACATIONYF\_DK\_** |
| 43.5.2 Vacation days/year account (NO,FD) | salary | 2nd of June every year | 03:00 | FTP | Generates logical reports with prefix: **VACATIONYF\_NO\_** |
| 43.5.2 Vacation days/year account (NO,CC) | salary | 2nd of January every year | 03:00 | FTP | Generates logical reports with prefix: **VACATIONYC\_NO\_** |
| 43.5.2 Vacation days/year account (SE) | salary | 2nd of June every year | 03:00 | FTP | Generates logical reports with prefix: **VACATION\_Y\_SE\_** |
| 44.1 Per Diem (DK) | salary | 6th every month | 02:30 | FTP | Generates logical reports with prefix: **PERDIEM\_DK\_** |
| 44.1 Per Diem (NO) | salary | 6th every month | 02:30 | FTP | Generates logical reports with prefix: **PERDIEM\_NO\_** |
| 44.1 Per Diem (SE) | salary | 6th every month | 02:30 | FTP | Generates logical reports with prefix: **PERDIEM\_SE\_** |
| 44.1 Per Diem (CN) | salary | 6th every month | 02:30 | FTP | Generates logical reports with prefix: **PERDIEM\_CN\_** |
| 44.1 Per Diem (JP) | salary | 6th every month | 02:30 | FTP | Generates logical reports with prefix: **PERDIEM\_JP\_** |
| CR213 Per Diem TAX (NO) | salary | 6th every month | 02:30 | FTP | Generates logical reports with prefix: **PERDIEMTAX\_NO\_** |
| 44.2 Instructor Allowance (DK) | salary | 2nd every month | 02:30 | FTP | Generates logical reports with prefix: **SUPERVIS\_DK\_** |
| 44.2 Instructor Allowance (NO) | salary | 2nd every month | 02:30 | FTP | Generates logical reports with prefix: **SUPERVIS\_NO\_** |
| 44.2 Instructor Allowance (SE) | salary | 2nd every month | 02:30 | FTP | Generates logical reports with prefix: **SUPERVIS\_SE\_** |
| 44.3 Overtime and Allowances (DK) | salary | 2nd every month | 01:45 | FTP | Generates logical reports with prefix: **OVERTIME\_DK\_** |
| 44.3 Overtime and Allowances (NO) | salary | 2nd every month | 01:55 | FTP | Generates logical reports with prefix: **OVERTIME\_NO\_** |
| 44.3 Overtime and Allowances (SE) | salary | 2nd every month | 02:05 | FTP | Generates logical reports with prefix: **OVERTIME\_SE\_** |
| 44.3 Overtime Temp Crew (DK) | salary | 2nd every month | 02:15 | FTP | Generates logical reports with prefix: **TEMP\_CREW\_DK\_** |
| 44.3 Overtime Temp Crew (NO) | salary | 2nd every month | 02:25 | FTP | Generates logical reports with prefix: **TEMP\_CREW\_NO\_** |
| 44.3 Overtime Temp Crew (SE) | salary | 2nd every month | 02:35 | FTP | Generates logical reports with prefix: **TEMP\_CREW\_SE\_** |
| 44.5 Bought Days and Compensation Days (DK) | salary | 5th every month | 03:00 | FTP | Generates logical reports with prefix: **COMPDAYS\_DK\_** |
| 44.5 Bought Days and Compensation Days (NO) | salary | 5th every month | 03:00 | FTP | Generates logical reports with prefix: **COMPDAYS\_NO\_** |
| 44.5 Bought Days and Compensation Days (SE) | salary | 5th every month | 03:00 | FTP | Generates logical reports with prefix: **COMPDAYS\_SE\_** |
| 44.6 Vacation Lists STO | salary | 2nd every month | 04:30 | FTP | Generates logical reports with prefix: **VAC\_LISTS\_STO** |
| 44.6.2 Vacation Lists OSL | salary | 2nd every month | 04:30 | FTP | Generates logical reports with prefix: **VAC\_LISTS\_OSL** |
| 32.12 Office List | crewreports | 16th every month | 05:00 | FTP | Generates office list report |
| 33.1 List 12 | crewreports | 2nd every month | 01:30 | FILE | Generates List 12. Scheduled and addressed separately per station (CPH AH, CPH CA rest, CPH FD,OSL,STO,SVG,TRD,NRT,BJS,SHA) |
| 33.2 List 9 | crewreports | 2nd every month | 01:25 | FILE | Generates List 9. |
| 33.9 Master Crew List, Initial Load | crewreports |  |  | - | Generates Master crew list, initial load.  Only run manually. |
| 33.9 Master Crew List, Incremental | crewreports | Daily | 01:00 | FILE | Generates Master crew list incremental updates |
| Noncore statistics | noncore\_statistics | Daily | 01:30 | FILE | Generates statistics file at $CARMDATA/STATISTICS |
| CrewLandings | crewlandings | Daily | 02:30 | - | Updates missing landings. |
| SubQ Violation | subq\_violation | Daily | 10:00 |  | Updated table rule\_violation\_log with SubQ rule violations |

### Technical summary

#### Rave

References to modules used, some important or difficult rave variables can be described.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
|  |  |  |

#### Database Tables

References to tables used. (u) means that table is being updated by the interface.

**Note!** Technical details should be documented in the code.

| Name | Description |
| --- | --- |
| meal\_order | Meal (u) |
| meal\_order\_line | Meal (u) |
| meal\_customer | Meal |
| meal\_supplier | Meal |
| meal\_order\_update | Meal (u) |
| meal\_order\_update\_line | Meal (u) |

#### Scripts

References to scripts used.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
| Meal.py | lib/python/meal | CrewMeal generation |
| HotelBookingRun.py | lib/python/hotel\_transport | Hotel bookings |
| TransportBookingRun.py | lib/python/hotel\_transport | Transport bookings |
| PassiveBookingRun.py | lib/python/passive | Passive bookings |
| X12.py | lib/python/replication | X12 interface |
| X3.py | lib/python/replication | X3 interface |
| X4.py | lib/python/replication | X4 interface |
| crew\_manifest.py | lib/python/carmusr/paxlst | Crew Manifest |
| baggage.py | lib/python/crewlists | Crew Baggage |
| loadsheet.py | lib/python/crewlists | Crew to loadsheet |

#### Forms

References to forms used.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
| MealOrderFormHandler.py | lib/python/meal | CrewMeal form handler. |

#### Reports

References to reports used.

**Note!** Technical details should be documented in the code.

|  |  |  |
| --- | --- | --- |
| Name | Location | Description |
|  |  |  |

#### X4 Configuration

Optional configuration for the X4 interface - added in CR411 - introduces possibility to control the behaviour of the X4 crew data export:

* Control per qualification type if all future qualifications should be exported (ALL) or just current ones (CURRENT). Default is to export also future qualifications.
* Possibility to block export of individual crew and/or crew belonging to certain groups.

Example:

:

<dig\_reports>

<x4>

<qualification>

<class>crewQualMainRank</class>

<export>CURRENT</export>

</qualification>

<qualification>

<class>crewQualLocStation</class>

<export>CURRENT</export>

</qualification>

<block>

<group>V - CC - Denmark</group>

<crew>34112</crew>

<crew>85677</crew>

</block>

</x4>

:

Note that dig\_reports/x4 is the default parent path in the common configuration, under which the various <qualification> and <block> elements are placed. The parent path may be changed by configuring ‘configPath’ in the x4 DIG channel. Configuration of dig\_reports are located in $CARMUSR/etc/dig/settings

Valid qualification classes are: *crewQualGroupType, crewQualMainRank, crewQualLocBase, crewQualLocStation* and *crewQualLocCivicStation*.

For valid group names please refer to X4 specification “InterBids back office use case”.

## Jeppesen Crew Portal

### Architecture overview

Please be aware of the naming similarity between the “SAS Crew Portal” and the “Jeppesen Crew Portal Bid”.The “SAS Crew Portal” is the main application for SAS crew web services whereas “Jeppesen Crew Portal” is the product name for the bidding application delivered by Jeppesen. In order to avoid confusion, the Jeppesen Crew Portal Bid application will be referred to as **JCPB** in this document.

The diagram below gives an overview of how data flows from end users (Crew and Planner) to the data stores (CMS and JCPB database and Carmdata) and also how JCPB and the rest of CMS keep synchronized through scheduled cron jobs.

CrewPortal_deployment_data_flow.wmf

### SAS Portal

When crew login to the SAS Portal a LTPA cookie is created. This cookie contains a WA\_UID parameter with the CMS employment number of the crew member. This value is used by JCPB to authenticate the user.

### Load balancer

The load balancer is configured using sticky sessions meaning that a user session will either be allocated to cp\_main\_node or cp\_app\_node.

### JBoss cluster

The JBoss cluster consists of two machines, cp\_main\_node and cp\_app\_node, running one JBoss process each.

cp\_app\_node (JCPB application node) handles user sessions allocated to this node by the load balancer.

cp\_main\_node (JCPB main node) handles, in addition to allocated user sessions, the execution of scheduled JCPB cron jobs for importing and exporting data from/to the rest of CMS.

For each node, the JCPB is installed in the “default” JBoss server which has the base directory /opt/java/jboss/server/default. This directory contains three important sub directories:

* deploy. This directory contains the JCPB binaries which are compiled from the JCPB Carmsys and the Carmusr source code.
* conf. This directory contains JCPB configuration files generated by the cmsshell command “crewportal buildsystem”
* log. This directory contains log files written by JBoss and JCPB.

The log directories contain files with a daily file name rotation scheme. To avoid that these disks run out of available disk space these directories should be monitored and have a regular clean up of old log files.

The JBoss processes are started individually on each machine with:

/etc/init.d/jboss start

To stop JBoss, first find the JBoss java process id with:

ps -fHu jboss

And then kill it with

kill PID

where PID is the Java process id.

#### JBoss status

To verify that JBoss is up and running, go to **Error! Hyperlink reference not valid.** and the JCPB login form will be presented if Jboss is running.

If the login form is not presented, logon to the JBoss host and run ps –fHu jboss and check for the jboss java process.

### Backend servers

The JCPB communicates with several CMS backed processes. Where these are running can be found out by issuing the “cmsshell smd services” command. In addition to the existing sysmond nodes main\_node and publish\_node, the JCPB has introduced cp\_main\_node running crewportalproxy.

#### portal\_latest

Called by the JCPB for the service calls get\_rosters, get\_available\_days\_off and create\_request. Since the portal\_latest is used by other CMS functions, JCPB calls are proxied through the crewportalproxy command. This both allows the load on main\_node running portal\_latest to be controlled and also it allows for a panic shutdown of all JCPB generated traffic to portal\_latest.

Log files for the report workers of portal\_latest will end up in:

1. /var/carmtmp/logfiles/ if the directory exist
2. $CARMUSR/current\_carmptmp\_cct/logfiles/ otherwise

The log directory contains files with a daily file name rotation scheme. To avoid that the disk run out of available disk space the directory should be monitored and have a regular clean up of old log files.

#### portal\_publish

Called by the JCPB for the service call get\_rosters. As with the portal\_latest, portal\_publish is also used for other CMS functions and therefore calls from JCPB are proxied through the crewportalproxy command.

Since portal\_publish is used for crew checkin, it is even more critical than portal\_latest. Because of this the get\_roster service call can easily be re-routed to portal\_latest instead. How to do this is described in the trouble shooting guide in SystemAdministratorManual. The result of re-routing the get\_roster call will be that crew see a best guess of what their published roster looks like instead of the real published one.

Log files for the report workers of portal\_publish will end up in:

1. /var/carmtmp/logfiles/ if the directory exist
2. $CARMUSR/current\_carmptmp\_cct/ otherwise

The log directory contains files with a daily file name rotation scheme. To avoid that the disk run out of available disk space the directory should be monitored and have a regular clean up of old log files.

#### portal\_manpower\_c and portal\_manpower\_f

Called by JCPB during vacation bidding. These are only used by JCPB but can generate load on the sysmond nodes. Due to this they are also proxied through crewportalproxy.

Log files for the report workers of portal\_manpower\_c and portal\_manpower\_f will end up in:

1. $CARMUSR/current\_carmptmp\_cmp/logfiles/

The log directory contains files with a daily file name rotation scheme. To avoid that the disk run out of available disk space the directory should be monitored and have a regular clean up of old log files.

#### crewportalproxy

The crewportalproxy process limits the amount of load JCPB can generate on the sysmond nodes by limiting the amount of simultaneous xmlrpc service calls that can be active at any time. By configuring crewportalproxy to allow less service calls than the report servers have workers it can be guaranteed that the report servers have workers available for higher prioritized CMS related service calls.

The amount of simultaneous crewportalproxy service calls is configured in $CARMUSR/etc/programs/crewportalproxy.xml

crewportalproxy is started with sysmond on cp\_main\_node. Not having it running on main\_node or publish\_node allows it to be shutdown “brutally” (in worst case by shutting down cp\_main\_node) without affecting the rest of CMS.

### Jeppesen Crew Portal batch jobs

JCPB tasks are defined in the file $CARMUSR/lib/webapp/conf/interbids.properties using the property called interbids.scheduler.fileNames.

Each task corresponds to a xml file located in $CARMUSR/lib/webapp/conf. This xml file contains both defines the task and cron job scheduling.

To apply changes in these files the JCPB system needs to be rebuilt (“crewportal buildsystem” in cmsshell).

#### import\_job

This task will import crew data and periods from the XML data source $CARMDATA/crewportal/datasource/import.

For details about XML datasources and the import document, see the Jeppesen Crew Portal Bid Developer Guide.

The result of the job will be appended to the log $JBOSS\_HOME/server/default/logs/interbids.log

#### trip\_cache\_job

This task will refresh the trip cache in JCPB by reading them from portal\_latest.

This task can generate quite heavy load on both portal\_latest and JBoss meaning that it is preferrebaly scheduled once a night during low load.

The result of the job will be appended to the log $JBOSS\_HOME/server/default/logs/interbids.log

#### current\_period\_export\_job

This task exports bids to $CARMDATA/crewportal/datasource/bids/new/bids.etab. This file is later used by the cmd\_install\_bids job. I.e. current\_period\_export\_job should be scheduled before cmd\_install\_bids in order for cmd\_install\_bids to use the latest data.

The result of the job will be appended to the log $JBOSS\_HOME/server/default/logs/interbids.log

## Batch jobs

These are configured to be run as Sysmond tasks in the file $CARMUSR/etc/sysmond/tasks.xml. The scheduling of the tasks is configured in the $CARMUSR/etc/sysmond/crontab.xml

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Run date** | **Run time UTC** | **Description** |
| **CMD\_RS\_CLEANUP** | Every night | 03:00 | Cleanup of old report files |
| **CMD\_DIG\_MONITOR** | Hourly | \*:30 | Not migrated yet: Monitor DIG channel errorr |
| **CMD\_CREWUSERFILTER\_UPDATE** | Hourly | \*:00 and \*.30 | Update of crew user filter table |
| **CMD\_ACCUMULATE\_CLEANUP\_AND\_STATS** | Every night | 00:02 | Various accumulators incl. CMP accumulators |
| **CMD\_CREW\_PORTAL\_EXPORT** | Every night | 04.00 | Produces a JCPB “import” xml document and fixed pattern files. |
| **CMD\_COMPDAYS\_RESET** | 1st every month | 03.00 | Reset BOUGHT and SOLD accounts |
| **CMD\_COMPDAYS\_F31** | 17th every month | 02:00 | Convert F3 to F31 for FC LH |
| **CMD\_COMPDAYS\_F3S** | 17th Nov. each year. | 02:10 | Convert F3 to F3S for FC |
| **CMD\_COMPDAYS\_F7SGAIN** | 17th Nov. each year. | 02:00 | Add yearly F7S.  Let all remaining F7S go to payment (FC)  Check exact time to run this job, it has to be after Interface 44.5 has been completed |
| **CMD\_COMPDAYS\_F7SRESET\_OSL** | 4th Jan each year | 04:00 | Resetting comp days for OSL. |
| **CMD\_COMPDAYS\_F7SRESET\_SVG** | 4th Jan each year | 04:05 | Resetting comp days for SVG. |
| **CMD\_COMPDAYS\_F7SRESET\_TRD** | 4th Jan each year | 04:10 | Resetting comp days for TRD. |
| **CMD\_GATHER\_SCHEMA\_STATS** | From Mon to Sat. | 02:00 | Running database statistics |
| **CMD\_CLEANUP** | 1st and 15th every month | 01:00 | Cleanup various tables |
| **CMD\_INSTALL\_BIDS** | 16th – 6th | 05:00 | Install JCPB current PBS-bids |
| **SAS\_AG\_LOOP\_TIME\_MONITOR** | Hourly | \*:00 | Not migrated yet: Creates xml-log of alert generator looptimes |
| **SAS\_SERVER\_MONITOR** | Every 5th minute. |  | Not migrated yet: SAS Server Monitor - Creates HTML/URL pointers in carmtmp to google charts that shows the servers cpu load and memory usage |
| **CMD\_CREATE\_REVISION** | 16th every month. | 02:00 | Create CMP revision after Roster release. To enable CMP to open old periods. |
| **CMD\_UPDATEPUBREV** | Every night. | 04:00 | Updateing pub revision to speed up database. |
| **CMD\_CHECK\_FDP\_EXTENSION\_FLAG** | 16th every month. | 01:30 | Recalculates FDP extension flags due to a different definition in Tracking than Planning. |
| **CMD\_CREWLOG** | Every day | 15:30 | Summary of block hours, landings, etc with values stored in accumulator tables. |
| **CMD\_ALERT\_RESTART** | Every day | 01:30 | Restarts the alert generator. |
| **CMD\_EXT\_PUBLISH\_SERVER\_RESTART** | Every day | 02:40 | Restarts the external publish server. |
| **CMD\_ARPC\_RESTART** | 16th every month. | 01:00 |  |

# Technical reference

## Technical summary

All Integration processes is run on the Jeppesen Infrastructure cluster h1cms07a for PROD and on h1cms97a for TEST.

All configuration is found in $CARMUSR/etc. There are a few target systems: PROD, PROD\_TEST, a few starting with GOT and dynamic. The file local.xml in the “etc” directory specifies which system to use and if no local.xml is found, the dynamic configuration will be used. (this is usally used when developing and testing individually). The local.xml is usually a link to the target system e.g. to PROD\_TEST.xml in the test environment.

The directory “hosts” contains configuration where different programs shall be located in the target system e.g. the file PROD.xml states that the SAS\_RS\_WORKER\_LATEST shall be executed on h1cms07a.

The Sysmond configuration is found in the “sysmond” directory. It contains configuration for each target system e.g. PROD and PROD\_TEST. The batch jobs configuration is found in the “tasks.xml” and the scheduling of these tasks in “crontab.xml”.

All program configuration such as dig and reportservers configuration is found in the programs directory. This is common for all target systems.

Dig configuration is found in the “dig” directory. The channels are located in the “dig/channels” directory and the schedule definitions are found in the digscheduler.xml file. The main configuration file is found in “programs/dig.xml” Dig settings as mail and MQ configuration is located in “dig/settings”.

Database configuration is found in the “db” directory. There are configuration files for each target system.