**MAXDAT Contact Center Reference Guide**

**DRAFT 0.1**

Contents

[Version History 5](#_Toc382556831)

[Target audience 6](#_Toc382556832)

[Introduction 7](#_Toc382556833)

[Core Technologies Overview 8](#_Toc382556834)

[Installation Instructions 9](#_Toc382556835)

[Pentaho Data Integration (Kettle v4.2.1) 9](#_Toc382556836)

[Ant 9](#_Toc382556837)

[Ivy 10](#_Toc382556838)

[Subversion 10](#_Toc382556839)

[Microsoft SQL Server 10](#_Toc382556840)

[Oracle Database 10](#_Toc382556841)

[Development Environment Set-Up Instructions 11](#_Toc382556842)

[General 11](#_Toc382556843)

[Project-Specific Instructions 12](#_Toc382556844)

[Texas EB 12](#_Toc382556845)

[Hawaii HIX 12](#_Toc382556846)

[Illinois EB 13](#_Toc382556847)

[High-Level Architecture 14](#_Toc382556848)

[ETL Framework Architecture 14](#_Toc382556849)

[Scheduling Architecture 14](#_Toc382556850)

[Overview 14](#_Toc382556851)

[System Components 14](#_Toc382556852)

[General Deployment Process 17](#_Toc382556853)

[Data Output 18](#_Toc382556854)

[Data Administration 18](#_Toc382556855)

[How to add a new Contact Queue 18](#_Toc382556856)

[Project Structure 19](#_Toc382556857)

[Data Architecture 20](#_Toc382556858)

[Database Table Prefix Convention / Definitions 20](#_Toc382556859)

[Data Dictionary 20](#_Toc382556860)

[Data Diagrams 20](#_Toc382556861)

[Project-Specific Contact Center ETL Logic and Configuration 20](#_Toc382556862)

[Texas EB 20](#_Toc382556863)

[Environments 21](#_Toc382556864)

[Environment Variables 21](#_Toc382556865)

[Data Sources 21](#_Toc382556866)

[Root Kettle Jobs 21](#_Toc382556867)

[Shell Scripts 22](#_Toc382556868)

[Cron Schedules 22](#_Toc382556869)

[Logging 22](#_Toc382556870)

[File Upload Directories 23](#_Toc382556871)

[Deployment Process 23](#_Toc382556872)

[Testing Process 23](#_Toc382556873)

[Hawaii HIX 23](#_Toc382556874)

[Environments 23](#_Toc382556875)

[Environment Variables 24](#_Toc382556876)

[File Upload Directories 24](#_Toc382556877)

[Environment Variables 25](#_Toc382556878)

[Kettle Properties Definitions 25](#_Toc382556879)

[Data Sources 25](#_Toc382556880)

[Root Kettle Jobs 26](#_Toc382556881)

[Shell Scripts 26](#_Toc382556882)

[Cron Schedules 26](#_Toc382556883)

[Logging 27](#_Toc382556884)

[Data Load Process 27](#_Toc382556885)

[Deployment Process 28](#_Toc382556886)

[Testing Process 28](#_Toc382556887)

[Lessons Learned 28](#_Toc382556888)

[PDI/Kettle 28](#_Toc382556889)

[Dimension Lookup / Update Step Limitation 28](#_Toc382556890)

[JNDI Browse Limitation (v4.2.1) 28](#_Toc382556891)

[Oracle 29](#_Toc382556892)

[Microsoft SQL Server 29](#_Toc382556893)

[Technical Design Decisions 29](#_Toc382556894)

[Main / Implementation Structure 29](#_Toc382556895)

[Acronym List 29](#_Toc382556896)

[References 30](#_Toc382556897)

# Version History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Change Description** |
| 0.1 | 01/17/2014 | Cecil Beeland | Initial Outline; Scheduling Architecture Section; |
| 0.2 | 3/14/2014 | Austin Baker | Installing Caché Monitor for IL EB |
| 0.3 |  |  |  |
| 0.4 |  |  |  |
| 0.5 |  |  |  |

# Target audience

This reference guide is for developers and administrators of the Contact Center module of MAXDAT. Its intent is to provide developers and administrators with the guidance necessary to develop, build, test, deploy and manage MAXDAT Contact Center capabilities.

# Introduction

The MAXDat Contact Center project extracts data from the contact center infrastructure to support the Agent Efficiency and Production Planning MAXDat modules. The data sources for MAXDat Contact Center are the Automatic Call Distributor (ACD), Integrated Voice Response (IVR) system, Workforce Management (WFM) system and the ARENA staffing optimization models.

The purpose of the Agent Efficiency module is to support analysis and comparisons, both inter and intra project, on where agents paid time is being spent. This module provides the ability to measure key summary metrics including occupancy, utilization, absenteeism, and direct labor time per contact. For more information on the Agent Efficiency module, please see the Contact Center Agent Efficiency MAXDat Out-of-the-Box Presentation Objects [documentation](https://maxcs.maxinc.com/gm/document-1.9.1274926/Call%20Center%20Agent%20Efficiency%20OOTB.docx).

The Production Planning module enables the client and MAXIMUS management to monitor performance, anticipate call demand, establish realistic ongoing service targets based on past performance, and provide insight into areas for continuous process improvement. Production planning is an important aspect of contact center operations management; it involves tracking forecasted arrivals against actual arrivals, optimizing staffing needs, and managing service levels. For more information on the Production Planning module, please see the Call Center Production Planning Module [documentation](https://maxcs.maxinc.com/gm/document-1.9.1274778/20130716_Call%20Center%20Production%20Planning%20Module_v5.6.docx).

# Core Technologies Overview

The following core technologies were utilized for all of the components listed above. To see core technologies specific to a component, please skip to the component section.

* JRE (1.6.0\_45)
* [Pentaho Data Integration (v4.2.1)](http://www.pentaho.com/product/data-integration)
  + Pentaho data integration prepares and blends data to create a complete picture of your business that drives actionable insights. The complete data integration platform delivers accurate, “analytics ready” data to end users from any source.  With visual tools to eliminate coding and complexity, Pentaho puts Big Data and all data sources at the fingertips of business and IT users alike.
* [Apache Ant](http://ant.apache.org/)
  + Apache Ant is a Java library and command-line tool whose mission is to drive processes described in build files as targets and extension points dependent upon each other. The main known usage of Ant is the build of Java applications. Ant supplies a number of built-in tasks allowing to compile, assemble, test and run Java applications. Ant can also be used effectively to build non Java applications, for instance C or C++ applications. More generally, Ant can be used to pilot any type of process which can be described in terms of targets and tasks.
* [Apache Ivy](http://ant.apache.org/ivy/)
  + Apache Ivy is a popular dependency manager focusing on flexibility and simplicity.
* [Apache Ant-Contrib](http://ant-contrib.sourceforge.net/)
  + The Ant-Contrib project is a collection of useful tasks for Apache Ant.
* [Microsoft SQL Server](https://www.microsoft.com/en-us/sqlserver/default.aspx)
  + Microsoft SQL Server is a relational database management system developed by Microsoft.
* [Oracle Database Server](http://www.oracle.com/us/products/database/overview/index.html)
  + The Oracle Database is an object-relational database management system produced and marketed by Oracle Corporation.
* [Apache Subversion](http://subversion.apache.org/)
  + Subversion is an open source version control system. Subversion exists to be universally recognized and adopted as an open-source, centralized version control system characterized by its reliability as a safe haven for valuable data; the simplicity of its model and usage; and its ability to support the needs of a wide variety of users and projects, from individuals to large-scale enterprise operations.

# Installation Instructions

The following provide instructions to install all the tools for all required features for all components. To only install features required for a specific component, please skip to the component’s section.

## Pentaho Data Integration (Kettle v4.2.1)

* + 1. Download Kettle v4.2.1, available from SourceForge:
  + <http://sourceforge.net/projects/pentaho/files/Data%20Integration/4.2.1-stable/>
  + Change the following line in the ‘Kitchen.bat’ script to facilitate JNDI:
    - OPT=”$PENTAHO\_DI\_JAVA\_OPTIONS … %KETTLE\_LOG\_SIZE\_LIMIT%” ***to***

OPT=”$PENTAHO\_DI\_JAVA\_OPTIONS … %KETTLE\_LOG\_SIZE\_LIMIT%”

“-DKETTLE\_JNDI\_ROOT=%KETTLE\_DIR%/simple-jndi”

* + 1. Ensure that Java JDK is installed
    2. Ensure that SVN is installed
  + Grab the repository from svn://rcmxapp1d.maximus.com/maxdat/Contact Center
    1. Set the following environment variables:
  + KETTLE\_DIR=*<path to Kettle installation>*
  + JAVA\_HOME=*<path to java>*
  + PATH=*<path to Kettle installation>*;*<path to JRE>*
    1. Copy the contents of the ‘kettle.properties’ file found at <*repo-root*>/Contact Center/trunk/kettle to that of the ‘kettle.properties’ file located in your home directory under the ‘.kettle’ folder
  + C:\Users\*<name>*\.kettle
    1. Copy the contents of the ‘jdbc.properties’ file found at <*repo-root*>/Contact Center/trunk/kettle to that of the ‘jdbc.properties’ file located in your pentaho data integration directory under the ‘simple-jndi’ folder
  + C:\*<Kettle dir>*\data-integration\simple-jndi

## Ant

1. Install Ant

* Apache Ant v1.9.0
  + Apache Ant utilized to create the build script that packages the different deployment bundles (e.g. Texas, Hawaii, MOTS) and runs the test suite
  + Download from the archives page: <http://archive.apache.org/dist/ant/binaries>
  + Install using the following instructions: <http://ant.apache.org/manual/install.html>
* ant-contrib v0.6
  + Collection of tasks used in the build script (e.g. the <for> task)
  + Download/Install using the following instructions: <http://ant-contrib.sourceforge.net>

1. Set the following environment variables to facilitate build script:

* ANT\_HOME=*<path to Ant install>*
* PATH=:*<path to Ant install>*/bin

## Ivy

* Install Ivy v2.3.0
  + Ivy utilized to resolve dependencies around database connections and creating a common build file for PDI’s Kitchen script (.bat | .sh)
  + The setup of the Ivy files was cloned from ‘testkitchen’ – a testing framework for PDI[1]
    - Download here: <https://code.google.com/p/testkitchen/downloads/list>
    - The download contains installation instructions and sample files
  + Download from the archives page: [http://archive.apache.org/dist/ant/ivy](http://archive.apache.org/dist/ant/ivy/)
  + Install using the following instructions: <http://ant.apache.org/ivy/history/2.2.0/install.html>

## Subversion

1. Navigate to <http://tortoisesvn.net/downloads.html>

* Download and Install TortoiseSVN

2. Navigate to the directory that you wish the repository to reside in.

* Right-Click in the directory and select ‘SVN Checkout…’
* For URL of repository, type: svn://rcmxapp1d.maximus.com/maxdat/Contact Center
* Checkout Depth: Fully recursive
* Revision: HEAD Revision

## Microsoft SQL Server

1. Download and install SQL Server Express
   * <http://www.microsoft.com/en-us/sqlserver/editions/2012-editions/express.aspx>
2. Download and install SQL Server Management Studio 2012 (non-Express)
   * <http://download.microsoft.com/download/8/D/D/8DD7BDBA-CEF7-4D8E-8C16-D9F69527F909/ENU/x64/SQLManagementStudio_x64_ENU.exe>
3. After installation
   * Open MSSQLServer Configuration Manager
   * Navigate to SQL Server Network Configuration  Protocols for SQLEXPRESS
   * Ensure that TCP/IP Connections are enabled
   * Right-click on TCP-IP and select Properties
   * Select the IP Addresses tab; Scroll down to the very bottom of the properties. *(These steps required for JNDI connection to succeed)*
     1. Remove the value from the TCP Dynamic Ports field of the ‘IP All’ section.
     2. Add the value **1433** to the TCP Port field of the ‘IP All’ section.
   * Restart the **SQL Server (SQLEXPRESS)** Service

## Oracle Database

1. Navigate to

# Development Environment Set-Up Instructions

## General

* Install Pentaho Data Integration (v.4.2.1)
* Install Oracle Database Server
* Create database
  + Create tablespaces
    - …\trunk\kettle\MAXDAT\main\scripts\create\_tablespaces.sql
  + Create an oracle user for the database, [state code]maxdat (e.g., himaxdat)
    - …\trunk\kettle\MAXDAT\main\scripts\create\_users.sql
    - If you create a new user, make sure that you update jdbc.properties
  + Create staging area as the maxdat user
    - …\trunk\kettle\MAXDAT\main\scripts\create\_staging\_database.sql
  + Create dimensional area as the maxdat user
    - …\trunk\kettle\MAXDAT\main\scripts\create\_dimensional\_database.sql
* Add jdbc.properties to PDI install location
  + Copy file from project directory
    - …\trunk\kettle\jdbc.properties
  + To pentaho install directory
    - …/data-integration/simple-jndi
  + Review to make sure that any necessary updates for Oracle PE are applied.
* Add kettle properties to your kettle.properties file
  + Copy …\trunk\kettle\kettle.properties to $HOME\.kettle
  + Modify kettle. properties as necessary for your environment.
    - E.g., cc.project.main.directory=…\\trunk\\kettle\\MAXDAT\\main
  + Ensure that the appropriate project properties are uncommented. Each project will have the following properties available to set:

|  |  |
| --- | --- |
| **Property** | **Description** |
| JNDI\_MAXDAT\_SOURCE | The JNDI name for the MAXDAT contact center database |
| JNDI\_WFM\_SOURCE | The JNDI name for the Workforce Management data source |
| JNDI\_ACD\_SOURCE | The JNDI name for the Automatic Call Distribution data source |
| cc.project.implementation.directory | The path to the project-specific ETL scripts |
| cc.project.main.directory | The path to the contact center ETL scripts |
| cc.project.log.directory | The path to the contact center ETL logs |
| cc.project.test.directory | The path to the contact center ETL test suite |
| cc.project.loadTimeZoneAm | A Boolean (‘true’ or ‘false’ value) property to determine whether or not to load the timezone data corresponding to the cc.timezone property |
| cc.project.forecast.directory | The path to the location where forecast files will be uploaded for processing into the MAXDAT database |
| cc.project.name | The name of the project (e.g. ‘TX EB’, ‘HI HIX’) |
| cc.program.name | The name of the program (e.g. ‘EB’, ‘HIX’) |
| cc.geomaster | The name of the geography location (e.g. ‘Texas’, ‘Hawaii’) |
| cc.timezone | The name of the timezone (e.g. ‘America/Chicago’, ‘America/Hawaii’) |
| cc.mots.transport | The method of transport for files to the Segment database. A value of ‘ftp’ will cause the ETL to transfer the files to the application server described by the cc.mots.ftp.\* properties, while leaving it blank will cause the files to be placed in the ‘Outbound’ folder under the directory set by the cc.mots.file.directory property for the MoveIt program to transfer the files |
| cc.mots.ftp.name | The IP or host name of the MOTS application server |
| cc.mots.ftp.port | The port number for the MOTS application server |
| cc.mots.ftp.user | The username for the MOTS application server |
| cc.mots.ftp.pass | The password for the MOTS application server |
| cc.mots.file.directory | The path on the local machine where files from the project will be created before being sent to the MOTS application server for processing in the Segment database |
| cc.mots.file.destination | The path to the remote location of the ‘Inbound’ folder on the MOTS application server for processing project files into the Segment database |

* Initialize database tables via initialize job
  + ...\trunk\kettle\MAXDAT\main\jobs\initialization\initialize\_Contact\_Center.kjb

## Project-Specific Instructions

### Texas EB

* Install Microsoft SQL Server
* Create stub Cisco database in SQL Server
  + …\trunk\kettle\source-systems\cisco\scripts\CREATE\_DATABASE\_EBHDSDB.sql
  + Load stub Cisco database with sample data
    - …\kettle\source-systems\cisco\load\_EB\_Cisco.kjb
* Create stub Blue Pumpkin database in SQL Server
  + …\trunk\kettle\source-systems\blue-pumpkin\scripts\CREATE\_DATABASE\_BPMAINDB.sql
  + Load stub Blue Pumpkin database with sample data
    - …\kettle\source-systems\blue-pumpkin\load\_Blue\_Pumpkin.kjb
* Run the contact center smoke test
  + …\trunk\kettle\MAXDAT\test\jobs\contact\_center\_smoke\_test.kjb

### Hawaii HIX

* Run test of load\_Contact\_Center job
  + Put 1 days’ worth of data in the cc.project.newFiles.directory folder as defined in kettle.properties from Echopass test file zip.
    - …\Contact Center\HIHIX\Echopass\MAXDAT\_Echopass\_20131122.zip
      * MAXHIHIX\_CC\_S\_ACD\_AGENT\_ACTIVITY\_111313050006.csv
      * MAXHIHIX\_CC\_S\_ACD\_INTERVAL\_111313050005.csv
      * MAXHIHIX\_CC\_S\_AGENT\_111313050002.csv
      * MAXHIHIX\_CC\_S\_CALL\_DETAIL\_111313050007.csv
      * MAXHIHIX\_CC\_S\_CONTACT\_QUEUE\_111313050001.csv
      * MAXHIHIX\_CC\_S\_IVR\_INTERVAL\_111313050012.csv
      * MAXHIHIX\_CC\_S\_IVR\_STEP\_111313050012.csv
  + ...\trunk\kettle\MAXDAT\implementation\HIHIX\jobs\load\_Contact\_Center.kjb
* Load November data
  + Load all of the November files (this should take @ 30 minutes to load)
    - …\Contact Center\HIHIX\Echopass\MAXDAT\_Echopass\_November.zip
  + ...\trunk\kettle\MAXDAT\implementation\HIHIX\jobs\initialization\load\_Contact\_Center.kjb
* Generate MAXDAT Daily Metric File Set for delivery to MOTS
  + Run flatten\_project\_facts.kjb to generate the daily file set that is consumed by MOTS.
    - ...\trunk\kettle\MAXDAT\implementation\HIHIX\jobs\dimensional\flatten\_project\_facts.kjb
    - Parameters
      * startDate = 2013/11/01
      * endDate = 2013/11/01
      * projectName = HIHIX
  + Check for file set in the following folder
    - ${cc.mots.file.directory}/Archive

### Illinois EB

* Register, download, and install ‘Caché 2013.1.13’:
  + <http://download.intersystems.com/download/register.csp>
  + Run ‘CacheClient\_x64.exe’ wizard
  + TRYCACHE will be the default instance for this install
* A cube icon appears in the system tray after install
  + right-click on the cube and go to Remote System Access > Management Portal > TRYCACHE:
    - Management Portal – the DBA web app (similar to Oracle’s)
* Create a new namespace in the Management Portal:
  + Click System Administration > Configuration > System Configuration > Namespaces
  + Click Create New Namespace
    - Call it CCMS\_STAT
  + Click Create New Database
    - Call it CCMS\_STAT
    - Give it a directory path
  + Leave all the defaults present for the database details and resources
  + Click Finish
  + Click Save
* Add the ‘cachejdbc.jar’ to Kettle:
  + Located in ‘C:\InterSystems\Cache\lib’ by default
  + Add it to ‘$PENTAHO\_HOME/data-integration/libext/JDBC’ folder
* Download and install SQuirreL SQL Client
  + http://squirrel-sql.sourceforge.net/#installation
  + Add cachejdbc.jar to the SQuirreL lib folder
    - [squirrel-install-directory]\lib
* Create alias for local instance of Cache
  + From Menu, select Aliases, select New Alias
    - Name: LOCALCACHE
    - Driver: Intersystems Cache
    - URL: jdbc:Cache://127.0.0.1:1972/CCMS\_STAT
    - User Name:
    - Password:
  + You will see the newly created namespace under the Namespaces folder
* Create the database tables within the CCMS\_STAT namespace
  + Run the SQL script located in ‘[repoRoot]\kettle\source-systems\avaya\scripts\

CREATE\_TABLES.sql’

* Load sample data from Avaya source database into local Avaya stub database
  + Run job ‘[repoRoot]\kettle\source-systems\avaya\load\_Local\_Avaya\_Data.kjb’

# High-Level Architecture

## ETL Framework Architecture

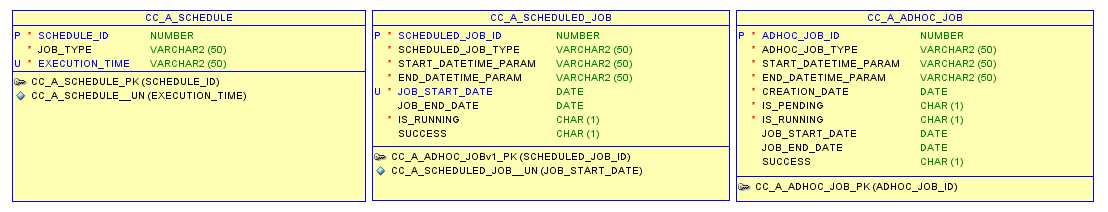
## Scheduling Architecture

### Overview

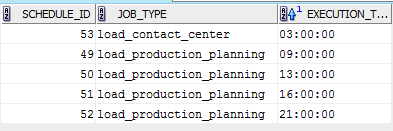
During the development of the ETL for the TXEB MAXDAT Contact Center implementation, the requirement was defined that there should be a data-driven scheduling architecture that would also allow one-off adhoc executions as needed. As such, a scheduling architecture was designed that leverages a set of scheduling-related administrative database tables, which are referenced by Kettle scripts that manage the high-level execution of the ETL process. These Kettle scripts are called on a regular basis (Ex: Hourly) by shell scripts. The design allows for two modes of execution: one which loads all components of the Contact Center ETL, and another that only loads tables relevant to the Production Planning component of Contact Center.

### System Components

#### Relevant Database Tables



* CC\_A\_SCHEDULE
  + Represents a daily schedule of executions by Job Type. Currently limited in granularity to Hourly.
  + Example Schedule:



* CC\_A\_SCHEDULED\_JOB
  + Serves as an audit table to manage potential concurrency issues and provide a history of scheduled executions.
* CC\_A\_ADHOC\_JOB
  + Referenced by the ETL to identify any pending ad-hoc executions.
  + The table also serves as an audit table to manage potential concurrency issues and provide a history of scheduled executions.

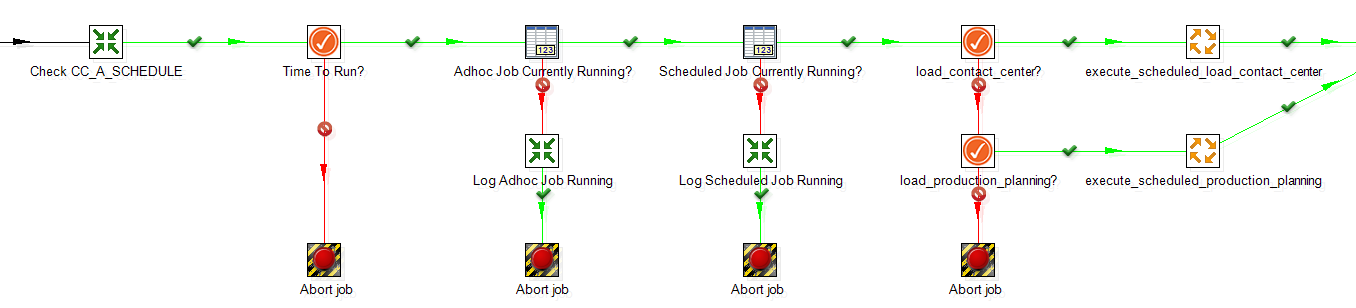
#### High-Level ETL Architecture

There are two top-level jobs that execute the TXEB Contact Center ETL, one that manages scheduled executions and one that manages ad-hoc executions. Each of these is called regularly by its own shell script.

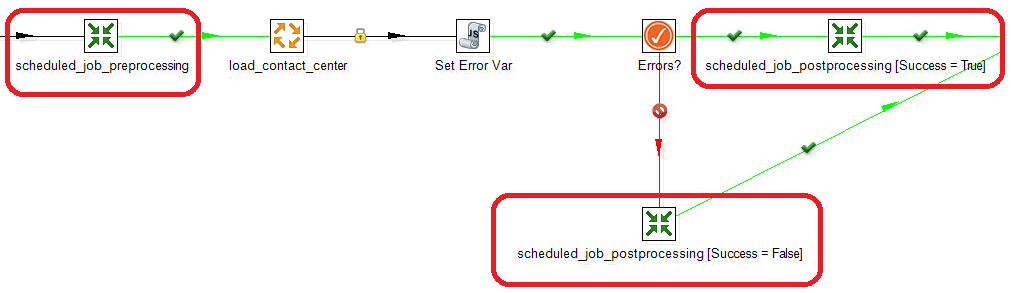
##### Manage All Scheduled Jobs

This job is called hourly by a shell script and manages the scheduled execution of both the full Contact Center load as well as the Production Planning load.

* General flow is illustrated below:



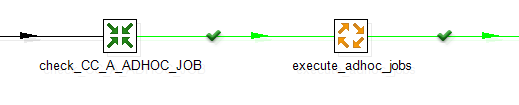
* There is additional scheduling and audit related processing before and after execution of load\_contact\_center or load\_production\_planning, including the creation and maintenance of a new record in the CC\_A\_SCHEDULED\_JOB table to represent the current execution. Illustrated below by the execute\_scheduled\_load\_contact\_center job:



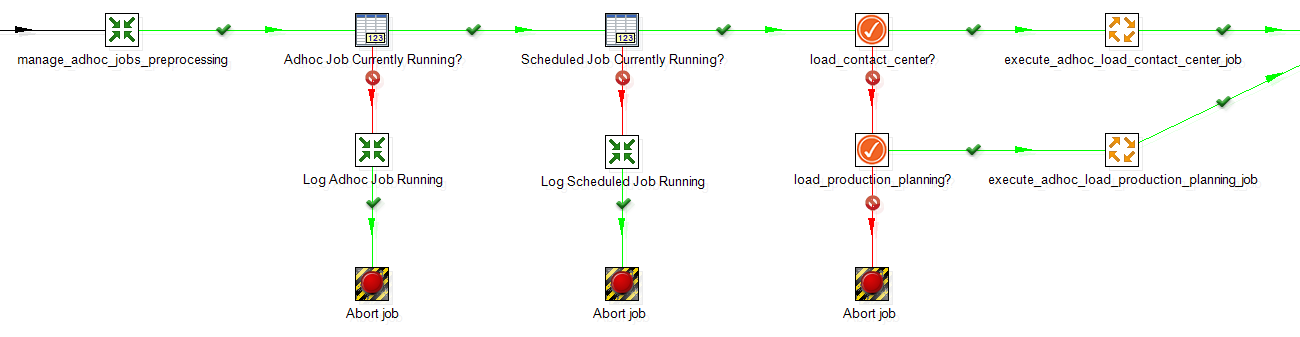
##### Manage All Ad-hoc Jobs

This job is called regularly by a shell script and checks for / manages ad-hoc executions of both the full Contact Center load as well as the Production Planning load.

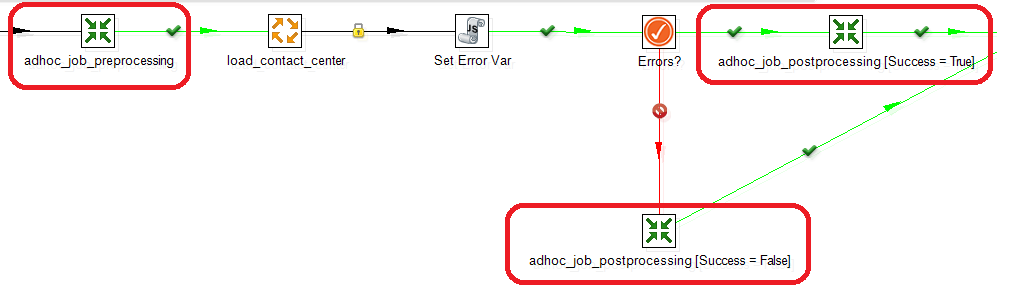
* The ETL first checks CC\_A\_ADHOC\_JOB table for any pending executions, then runs execute\_adhoc\_jobs for each record found:



* General flow of execute\_adhoc\_jobs is similar to manage\_all\_scheduled\_jobs, illustrated below:



* There is additional scheduling and audit related processing before and after execution of load\_contact\_center or load\_production\_planning, including the maintenance of the relevant record in the CC\_A\_ADHOC\_JOB table that represents the current execution. Illustrated below by the execute\_adhoc\_load\_contact\_center job:



##### Cron Jobs

* manage\_adhoc\_contact\_center\_jobs.sh
  + Executes manage\_all\_adhoc\_jobs.kjb
  + Scheduled to run regularly (Ex: Every 5 mins)
* manage\_scheduled\_contact\_center\_jobs.sh
  + Executes manage\_all\_scheduled\_jobs.kjb
  + Scheduled to run hourly (15 mins past each hour)

## General Deployment Process

* Build deployment package using Ant from the MAXDAT directory
  + ant package -Dproject=[project] -Dversion=[version]
    - projects
      * HIHIX
      * TXEB
  + Find the deployment bundles in the MAXDAT/build directory
  + Rename per the MAXIMUS deployment standard
    - kettle bundle
      * AS\_PATCH\_CONTACT\_CENTER\_YYYYMMDD\_[DEV NAME]\_v[VERSION].zip
    - db bundle
      * DB\_PATCH\_CONTACT\_CENTER\_YYYYMMDD\_[DEV NAME]\_v[VERSION].zip
  + upload deployment bundles to SVN
    - e.g. svn://rcmxapp1d.maximus.com/maxdat/Contact Center/HIHIX
* Deploy to local M21 test environment
  + Draft instructions for ISG and follow/edit them as you test the deployment
* Deploy to MAXIMUS DEV environment
  + Finalize instructions
  + Run test file
  + Run deployment test scripts
  + Communicate to team that the deployment to DEV is complete
    - Ann/Pam/Steven/Liz
  + Wait for Go/No Go from testers to deploy to UAT
* Create Release ticket
  + Link any jira tickets that are resolved by the release to the Release ticket
* Create deployment ticket for UAT deployment as sub task of the Release ticket
  + Link issues fixed by the deployment to the ticket
  + Revise instructions as necessary if issues encountered
  + Run deployment test scripts
  + Communicate to team that the deployment to UAT is complete
    - Ann/Pam/Steven/Liz
  + Wait for Go/No Go from testers to close the ticket and deploy to PRD
* Create deployment ticket for PRD deployment as sub task of the Release ticket
  + Link issues fixed by the deployment to the ticket
  + Confirm the deployment window
  + Assign ticket to ISG
  + Run monitoring job to ensure that source = staging = dimensional
  + Run deployment test scripts
  + Communicate to team that the deployment to PRD is complete
    - Ann/Pam/Steven/Liz
  + Wait for Go/No Go from testers to close the ticket

## Data Output

Projects participating in the daily load of the MOTS Contact Center will deliver actuals and forecast data from the project’s data mart. The data comes in an archive file (.zip) that contains a set of comma separated values (CSV) files which mirror the structure of the project’s dimensional tables:

* [ProjectName]\_[YYYYMMDD]\_[HHMI].zip
  + [YYYYMMDD]\_[ProjectName]\_CC\_F\_ACTUALS\_IVR\_INTERVAL.csv
  + [YYYYMMDD]\_[ProjectName]\_CC\_F\_ACTUALS\_QUEUE\_INTERVAL.csv
  + [YYYYMMDD]\_[ProjectName]\_CC\_F\_AGENT\_ACTIVITY\_BY\_DATE.csv
  + [YYYYMMDD]\_[ProjectName]\_CC\_F\_AGENT\_BY\_DATE.csv
  + [YYYYMMDD]\_[ProjectName]\_CC\_F\_FORECAST\_INTERVAL.csv
  + [YYYYMMDD]\_[ProjectName]\_CC\_F\_IVR\_SELF\_SERVICE\_USAGE.csv

Documentation of these files can be found in the following directory:

documentation/mots/MOTS File Dictionary.xlsx

## Data Administration

### How to add a new Contact Queue

1. Identify the following attributes of the queue.
   1. Queue name
      1. This should match exactly the name that is provisioned in the source system.
   2. Queue number
      1. For HI, this is manually assigned by the developer using the max number + 1 as queue number is provided in the Echopass data extract.
   3. Service percent
   4. Service seconds
   5. Unit of work name
   6. Queue type
2. Determine if the unit of work already exists.
   1. If unit of work does not already exist, add new record to the following tables
      1. CC\_C\_UNIT\_OF\_WORK
      2. CC\_D\_UNIT\_OF\_WORK
3. Add new record for the contact queue to the following tables using the attributes identified above. Any remaining attributes should be defaulted to match the other records in the tables.
   1. CC\_C\_CONTACT\_QUEUE
   2. CC\_S\_CONTACT\_QUEUE

# Project Structure

**svn://rcmxapp1d.maximus.com/maxdat/Contact Center**

* branches – unused
* HIHIX – HI deployment artifacts and instructions
* tags – unused
* trunk (directories/files of note below)
  + documentation
    - design
    - install – contains installation instructions for setting up a developer environment (will likely be unnecessary for a pre-existing dev environment)
    - spot – information from spot consulting including Cisco documentation
      * UCCE\_BK\_D832C827\_00\_database-schema-guide.pdf
  + kettle – primary project folder with ETL code and DB scripts
    - MAXDAT
      * build.xml – Ant build file used for packaging project for deployment
      * conf – contains project specific configuration for build process
      * implementation – contains project specific ETL code and DB scripts
        + HIHIX

jobs - Kettle .kjb files

load\_Contact\_Center.kjb – entry point for daily data load

* + - * + TXEB
      * main – shared ETL code and DB scripts.
        + bin – shell scripts for use in job execution
        + conf – build configuration files
        + jobs – Kettle .kjb files

execute\_scheduled\_load\_contact\_center.kjb – entry point for nightly job

execute\_scheduled\_load\_production\_planning.kjb – entry point for intraday ACD data load

load\_Contact\_Center\_Forecast.kjb – entry point for loading forecast data

dimensional – jobs for loading the dimensional area

initialization – jobs for initializing a deployment

sample-data – jobs for loading sample test data

staging – jobs for loading the staging area

utility – jobs for utility functions

* + - * + scripts – DB scripts

analysis – queries for analysis and testing purposes

insert – initialization scripts for reference data

versions – patch scripts

* + - * + transforms – Kettle .ktr files
        + simple-jndi – folder for version control of jdbc.properties
        + test – unit tests of Kettle transforms
    - MOTS – corporate data mart ETL code
    - source-systems - contains kettle jobs/transforms/data that allow us to populate a clone of the source database(s) for development purposes
  + models – datamodeler files

# Data Architecture

## Database Table Prefix Convention / Definitions

|  |  |
| --- | --- |
| Table Prefix | Purpose |
| CC\_A\_\* | Administrative tables. As currently implemented, tables with a CC\_A prefix facilitate job schedules. (Administrative tables are not presently in use in the HI implementation) |
| CC\_C\_\* | Configuration tables. The primary purpose for configuration tables is to enrich source data with MAXIMUS business context data. E.g, which unit of work a queue is associated, agent activity categories, MAXIMUS project attributes, etc. Also included in the configuration tables are filter tables for use in data extraction from source tables without having to modify ETL code. Instead, records can be added or removed from this table to modify the data filter logic. E.g., which ACD skill groups or queues to include in a data extraction. (Filter tables are not presently in use in the HI implementation) |
| CC\_L\_\* | Logging tables. The purpose for logging tables is to capture audit information about the system. This includes error logging, transformation logging, and database patch logging (i.e., which db patch scripts have been executed). |
| CC\_S\_\* | Staging tables. The staging area is the initial location for data extracted from the source system. CC\_S\_TMP\_\* tables are temporary tables that are used to quickly extract data from source systems without applying transformation or business logic on them to mitigate impact to the source systems. (Temporary tables are not presently in use in the HI implementation) |
| CC\_D\_\* | Dimension tables. Dimension tables contain the attributes associated with the facts to facilitate grouping, filtering and labeling of data. |
| CC\_F\_\* | Fact tables. Fact tables contain the Contact Center metrics. |

## Data Dictionary

## Data Diagrams

# Project-Specific Contact Center ETL Logic and Configuration

## Texas EB

The TXEB MAXDAT Contact Center ETL process extracts data from the Cisco Automatic Call Distributor (ACD) Historical Data Server (HDS) and the Blue Pumpkin Workforce Management (WFM) Database for the purpose of collecting data for Agent Efficiency and Production Planning reporting. This process is broken into two separate jobs, one nightly job that extracts Agent Efficiency and Production Planning data from both the ACD and WFM and an intraday job that currently runs four times per day and only extracts Production Planning data from the ACD.

### Environments

|  |  |  |
| --- | --- | --- |
|  | User | Database |
| DEV | [your maximus login] | TXMAXDAT =  (DESCRIPTION =  (ADDRESS = (PROTOCOL = TCP)(HOST = 10.11.135.2)(PORT=1545))  (CONNECT\_DATA = (SERVER=DEDICATED)  (SERVICE\_NAME = TXMAXDAT)  (INSTANCE\_NAME=TXMAXDAT)  )  ) |
| PROD | [your maximus login] | TXMAXDATPROD =  (DESCRIPTION =  (ADDRESS = (PROTOCOL = TCP)(HOST = ldohstxeb001.oracleoutsourcing.com)(PORT=14010))  (CONNECT\_DATA =(SERVER = DEDICATED)  (SID = PTXE4T) ) ) |

### Environment Variables

The table below describes the file paths where the Contact Center scripts are installed and where the Contact Center logs are written. These file paths are codified in Unix environment variables in the .bash\_profile. The table below contains the PRD file path. In other environments, the “-prd” should be replaced with “-$env\_code”. E.g., /u01/maximus/maxdat-dev/TXEB/ETL/scripts.

|  |  |
| --- | --- |
| **Env Variable** | **Physical path** |
| $MAXDAT\_ETL\_PATH | /u01/maximus/maxdat-prd/TXEB/ETL/scripts |
| $MAXDAT\_ETL\_LOGS | /u01/maximus/maxdat-prd/TXEB/ETL/logs |

A Contact Center specific directory, ContactCenter will be found within the directories defined above. All Contact Center files are contained within the ContactCenter directory.

### Data Sources

* Blue Pumpkin (BP)
* Workforce Management (WFM)

### Root Kettle Jobs

* manage\_all\_adhoc\_jobs.kjb
  + Designed to run regularly (Ex: Every 5 mins)
* manage\_all\_scheduled\_jobs.kjb
  + Scheduled to run hourly (15 mins past each hour)
* load\_Contact\_Center\_Forecast.kjb
  + Designed to run regularly to check for forecast files in the directory set by the cc.project.forecast.directory kettle property
* initialize\_Contact\_Center.kjb
  + Designed to be run once during the installation to load default and unknown values to support the Kimball Type II Slowly Changing Dimension (SCD)

### Shell Scripts

* manage\_adhoc\_contact\_center\_jobs.sh
* manage\_scheduled\_contact\_center\_jobs.sh
* run\_initialize\_contact\_center.sh
  + initialize the contact center database with default and unknown values to support the Kimball Type II SCD
* run\_load\_contact\_center.sh
  + load both the staging and dimensional tables on a scheduled basis
* run\_load\_contact\_center\_forecast.sh
  + check for and load the forecast files into the contact center database
* run\_flatten\_contact\_center.sh
  + flatten the fact and associated dimensional table data to files that are zipped and prepared for the MOTS application server
* run\_smoke\_test.sh
  + availability check of tables in the contact center database

### Cron Schedules

* Manage Ad-hoc Contact Center Loads
  + Would execute *manage\_adhoc\_contact\_center\_jobs.sh.* Not currently scheduled to run.
* Manage Scheduled Contact Center Loads
  + Executes *manage\_scheduled\_contact\_center\_jobs.sh*. Currently scheduled to run hourly at 15 mins past each hour.
* TXEB Schedule (per the CC\_A\_SCHEDULE table):

|  |  |
| --- | --- |
| **Job Type** | **Execution Time** |
| load\_production\_planning | 09:00:00 |
| load\_production\_planning | 13:00:00 |
| load\_production\_planning | 16:00:00 |
| load\_production\_planning | 21:00:00 |
| load\_contact\_center | 03:00:00 |

### Logging

Logging consists of the following items:

* CC\_L\_ERROR Table
  + Each of the Kettle jobs contains steps for logging errors to the CC\_L\_ERROR table in the event that a job fails. This transform will log a date and time in addition to the name of the job/transform that failed.
* CC\_L\_TRANSFORMATION Table
  + This is the default logging table outlined by PDI for logging transformation activity. The following table describes the fields made available in the database:

|  |  |
| --- | --- |
| **Field name** | **Field description** |
| ID\_BATCH | The batch ID. It's a unique number, increased by one for each run of a transformation. |
| CHANNEL\_ID | The logging channel ID (GUID), can be matched to the logging lineage information |
| TRANSNAME | The name of the transformation |
| STATUS | The status of the transformation : start, end, stopped |
| LINES\_READ | The number of lines read by the specified step. |
| LINES\_WRITTEN | The number of lines written by the specified step. |
| LINES\_UPDATED | The number of update statements executed by the specified step. |
| LINES\_INPUT | The number of lines read from disk or the network by the specified step. This is input from files, databases, etc. |
| LINES\_OUTPUT | The number of lines written to disk or the network by the specified step. This is input to files, databases, etc. |
| LINES\_REJECTED | The number of lines rejected with error handling by the specified step. |
| ERRORS | The number of errors that occurred. |
| STARTDATE | The start of the date range for incremental (CDC) data processing. It's the 'end of date range' of the last time this transformation ran correctly. |
| ENDDATE | The end of the date range for incremental (CDC) data processing. |
| LOGDATE | The update time of this log record. If the transformation has status 'end' it's the end of the transformation. |
| DEPDATE | The dependency date: the maximum date calculated by the dependency rules in the transformation settings. |
| REPLAYDATE | The replay date is synonym for the start time of the transformation. |
| LOG\_FIELD | The field that will contain the complete text log of the run. Usually this is a CLOB or (long) TEXT type of field. |

* Application server log files
  + The application server contains a location specified by the MAXDAT\_ETL\_LOGS environment variable for logging the results of each of the shell scripts that calls the root Kettle jobs.

### File Upload Directories

#### Forecast Files

The following folders must exist under the directory defined by cc.project.forecast.directory:

|  |  |
| --- | --- |
| **Directory** | **Description** |
| /Inbound | Location for files to be processed into the contact center database |
| /Processing | Location for files currently being processed into the contact center database |
| /Completed | Location for files that have completed processing |
| /Error | Location for files that have failed processing |

### Deployment Process

### Testing Process

## Hawaii HIX

### Environments

|  |  |  |  |
| --- | --- | --- | --- |
|  | User | App Server | Database |
| DEV | etladmin | 10.200.90.78  MAXDAT\_ETL\_PATH=/u01/maximus/maxdat-dev/HCCHIX/ETL/scripts  MAXDAT\_ETL\_LOGS=/u01/maximus/maxdat-dev/HCCHIX/ETL/logs | HIHXMXDD =  (DESCRIPTION =  (ADDRESS = (PROTOCOL = TCP)(HOST = rcmxdb09d.maximus.com)(PORT = 1547))  (CONNECT\_DATA =  (SERVER = DEDICATED)  (SERVICE\_NAME = hihxmxdd.maximus.com)  )  ) |
| UAT | [your maximus login] | rchxap13ua.maximus.com  MAXDAT\_ETL\_PATH=/u01/maximus/maxdat-uat/HCCHIX/ETL/scripts  MAXDAT\_ETL\_LOGS=/u01/maximus/maxdat-uat/HCCHIX/ETL/logs | HIHXMXDU =  (DESCRIPTION =  (ADDRESS = (PROTOCOL = TCP)(HOST = kil-scan-01.maximus.com)(PORT = 1547))  (CONNECT\_DATA =  (SERVER = DEDICATED)  (SERVICE\_NAME = hihxmxdu.maximus.com)  )  ) |
| INT | [your maximus login] | rchxap13ua.maximus.com  MAXDAT\_ETL\_PATH=/u01/maximus/maxdat-int/HCCHIX/ETL/scripts  MAXDAT\_ETL\_LOGS=/u01/maximus/maxdat-int/HCCHIX/ETL/logs | HIHXMXDI =  (DESCRIPTION =  (ADDRESS = (PROTOCOL = TCP)(HOST = kil-scan-01.maximus.com)(PORT = 1547))  (CONNECT\_DATA =  (SERVER = DEDICATED)  (SERVICE\_NAME = hihxmxdi.maximus.com)  )  ) |
| PROD | [your maximus login] | 10.1.229.162  MAXDAT\_ETL\_PATH=/u01/maximus/maxdat-prd/HCCHIX/ETL/scripts  MAXDAT\_ETL\_LOGS=/u01/maximus/maxdat-prd/HCCHIX/ETL/logs | HIHXMXDP =  (DESCRIPTION =  (ADDRESS = (PROTOCOL = TCP)(HOST = rsmxdb07.maximus.com)(PORT = 1547))  (CONNECT\_DATA =  (SERVER = DEDICATED)  (SERVICE\_NAME = hihxmxdp.maximus.com)  )  ) |

### Environment Variables

The table below describes the file paths where the Contact Center scripts are installed and where the Contact Center logs are written. These file paths are codified in Unix environment variables in the .bash\_profile. The table below contains the PRD file path. In other environments, the “-prd” should be replaced with “-$env\_code”. E.g., /u01/maximus/maxdat-dev/HCCHIX/ETL/scripts.

|  |  |
| --- | --- |
| **Env Variable** | **Physical path** |
| $MAXDAT\_ETL\_PATH | /u01/maximus/maxdat-prd/HCCHIX/ETL/scripts |
| $MAXDAT\_ETL\_LOGS | /u01/maximus/maxdat-prd/HCCHIX/ETL/logs |

A Contact Center specific directory, ContactCenter will be found within the directories defined above. All Contact Center files are contained within the ContactCenter directory.

### File Upload Directories

#### Echopass Files

The table below describes the file paths relevant to the Echopass data file ETL process. Files are delivered to the Inbound directory. From there, they are sorted into their respective context, ACD, IVR or WFM and then moved into the Processing directories while being processed. Once processed, they are moved into the Inbound\_archive directory. If an error occurs during processing, the files are moved into the Error directory.

|  |  |
| --- | --- |
| **Virtual path** | **Physical path** |
| New | /u01/maximus/maxdat-prd/HCCHIX/Echopass/Inbound |
| Processing | /u01/maximus/maxdat-prd/HCCHIX/Echopass/Processing |
| Completed | /u01/maximus/maxdat-prd/HCCHIX/Echopass/Inbound\_archive |
| Error | /u01/maximus/maxdat-prd/HCCHIX/Echopass/Error |

#### Forecast Files

The table below describes the file paths relevant to the Forecast data file ETL process. Files are delivered to the Inbound directory. From there, they are moved into the Processing directory while being processed. Once processed, they are moved into the Completed directory. If an error occurs during processing, the files are moved into the Error directory.

|  |  |
| --- | --- |
| **Virtual path** | **Physical path** |
| New | /u01/maximus/maxdat-prd/HCCHIX/forecasts/Inbound |
| Processing | /u01/maximus/maxdat-prd/HCCHIX/forecasts/Processing |
| Completed | /u01/maximus/maxdat-prd/HCCHIX/forecasts/Completed |
| Error | /u01/maximus/maxdat-prd/HCCHIX/forecasts/Error |

### Environment Variables

|  |  |
| --- | --- |
| **Variable Name** | **Description** |
| MAXDAT\_ETL\_PATH | Path to the Kettle scripts |
| MAXDAT\_ETL\_LOGS | Path to the Kettle logs |

### Kettle Properties Definitions

The following table describes the Kettle properties specific to the Hawaii implementation:

|  |  |
| --- | --- |
| **Property Name** | **Description** |
| cc.project.completedFiles.directory | The location for Echopass that have completed processing |
| cc.project.processingFiles.directory | The location for Echopass files currently in processing |
| cc.project.newFiles.directory | The location for Echopass files awaiting processing |
| cc.project.errorFiles.directory | The location for Echopass files that have failed processing |
| cc.project.FTP.user | ? |
| cc.project.FTP.Port | ? |
| cc.project.FTP.pass | ? |
| cc.project.FTP.IP | ? |
| cc.project.config.id | ? |
| cc.project.HIHIX.projectName | ? |

### Data Sources

#### Contact Center Actuals

The Production Planning Actuals report for the HCCHIX Project Contact Center relies on a data load from Echopass. The data comes in a set of comma separated values (CSV) files which mirror the structure of the staging tables.

* MAXHIHIX\_CC\_S\_ACD\_AGENT\_ACTIVITY\_[MMDDYYHHMMSS].csv
* MAXHIHIX\_CC\_S\_ACD\_INTERVAL\_[MMDDYYHHMMSS].csv
* MAXHIHIX\_CC\_S\_AGENT\_[MMDDYYHHMMSS].csv
* MAXHIHIX\_CC\_S\_CALL\_DETAIL\_[MMDDYYHHMMSS].csv
* MAXHIHIX\_CC\_S\_CONTACT\_QUEUE\_[MMDDYYHHMMSS].csv
* MAXHIHIX\_CC\_S\_IVR\_INTERVAL\_[MMDDYYHHMMSS].csv
* MAXHIHIX\_CC\_S\_IVR\_STEP\_[MMDDYYHHMMSS].csv

Documentation of these files can be found in the Echopass directory in the following files.

* MaxDat Data Dictionary.xlsx
* Maximus Data Dictionary V1.1.docx

#### Contact Center Forecasts

The Contact Center Production Planning Forecast relies on a data load from the Contact Center Arena model. This data source consists of a set of flat files.

* [YYYYMMDD]\_MAXDat\_Production\_Plan\_Parameters.csv
* [YYYYMMDD]\_MAXDat\_Agent\_Usage\_Report.csv
* MAXDat\_service\_metrics\_final.csv
* [YYYYMMDD]\_MAXDat\_Volume\_Data.csv

The Production Plan Parameters file contains the header information describing the production plan and production plan horizon that the file set is associated with. The Agent\_Usage, Service\_Metrics and Volume files contain the details of the forecast and are joined together via the production plan name, horizon start and end dates, interval start and end dates and the unit of work.

### Root Kettle Jobs

* load\_Contact\_Center.kjb
  + Designed to run regularly to check for inbound Echopass files to load into the contact center database
* initialize\_Contact\_Center.kjb
  + Designed to run once during the installation to load the database with default and unknown values in support of the Kimball Type II Slowly Changing Dimension (SCD)

### Shell Scripts

* run\_initialize\_contact\_center.sh
  + initialize the contact center database with default and unknown values to support the Kimball Type II SCD
* run\_load\_contact\_center\_forecast.sh
  + check for and load the forecast files into the contact center database
* run\_flatten\_contact\_center.sh
  + flatten the fact and associated dimensional table data to files that are zipped and prepared for the MOTS application server
* run\_smoke\_test.sh
  + Availability check of tables in the contact center database

### Cron Schedules

#### Contact Center Actuals

Timeline of events for daily file import:

* 7:00 ET - Echopass begins the file transfer to the HCCHIX FTP server (approx. time lapse 4 minutes)
* 7:10 ET - MoveIt picks the files up from the FTP server and puts them in the Inbound directory on the HI PROD app server (approx. time lapse 5 minutes)
* 7:30 ET – CRON job runs the ETL scripts every hour on the half hour (approx. time lapse 1 hour)

### Logging

Logging consists of the following items:

* CC\_L\_TRANSFORMATION Table
  + This is the default logging table outlined by PDI for logging transformation activity. The following table describes the fields made available in the database:

|  |  |
| --- | --- |
| **Field name** | **Field description** |
| ID\_BATCH | The batch ID. It's a unique number, increased by one for each run of a transformation. |
| CHANNEL\_ID | The logging channel ID (GUID), can be matched to the logging lineage information |
| TRANSNAME | The name of the transformation |
| STATUS | The status of the transformation : start, end, stopped |
| LINES\_READ | The number of lines read by the specified step. |
| LINES\_WRITTEN | The number of lines written by the specified step. |
| LINES\_UPDATED | The number of update statements executed by the specified step. |
| LINES\_INPUT | The number of lines read from disk or the network by the specified step. This is input from files, databases, etc. |
| LINES\_OUTPUT | The number of lines written to disk or the network by the specified step. This is input to files, databases, etc. |
| LINES\_REJECTED | The number of lines rejected with error handling by the specified step. |
| ERRORS | The number of errors that occurred. |
| STARTDATE | The start of the date range for incremental (CDC) data processing. It's the 'end of date range' of the last time this transformation ran correctly. |
| ENDDATE | The end of the date range for incremental (CDC) data processing. |
| LOGDATE | The update time of this log record. If the transformation has status 'end' it's the end of the transformation. |
| DEPDATE | The dependency date: the maximum date calculated by the dependency rules in the transformation settings. |
| REPLAYDATE | The replay date is synonym for the start time of the transformation. |
| LOG\_FIELD | The field that will contain the complete text log of the run. Usually this is a CLOB or (long) TEXT type of field. |

* Application server log files
  + The application server contains a location specified by the MAXDAT\_ETL\_LOGS environment variable for logging the results of each of the shell scripts that calls the root Kettle jobs.

### Data Load Process

#### Contact Center Actuals

Echopass uploads the files to an FTP server (sftp:\\Transfer.us2.echopass.com) and the MoveIt team picks them up and moves them to the HIHIX Production server (10.1.229.162). The Production server has a CRON job running run\_load\_contact\_center.sh on an hourly basis. As of release 0.1.5, the CRON job runs hourly on the half hour. The load\_contact\_center kettle script picks the files up from the New directory on the application server, moves them to the Processing directory for processing, and finally to the Completed directory upon successful completion of the scripts. If an error occurs, the erroneous files are moved to the Error directory.

#### Contact Center Forecasts

There are two mechanisms for uploading a forecast file set to the MAXDAT application server. The first is to sftp or scp the files into the forecast inbound directory located at the path below where [env] represents the relevant environment shorthand (dev/uat/int/prd).

* /u01/maximus/maxdat-[env]/HCCHIX/forecasts/Inbound

In the production environment, the forecast files can be uploaded via the MAXIMUS Xchange Portal.

* <https://xchange.maximus.com/>

Files uploaded to the [/](https://xchange.maximus.com/human.aspx?r=310263694&Arg12=folders)[Health\_Services/](https://xchange.maximus.com/human.aspx?r=310263694&Arg12=filelist&Arg06=652816666)[HI HIX/](https://xchange.maximus.com/human.aspx?r=310263694&Arg12=filelist&Arg06=935060343)[Forecasts/](https://xchange.maximus.com/human.aspx?r=310263694&Arg12=filelist&Arg06=947853206)https://xchange.maximus.com/images/null.gifdirectory will be delivered to the forecast inbound directory and picked up for processing.

### Deployment Process

### Testing Process

# Lessons Learned

## PDI/Kettle

### Dimension Lookup / Update Step Limitation

* This step only manages Type-2 SCD Insert/Updates effectively into the future. It will create invalid duplicates if you try to run this step for a record with an effective date that is prior to the most recent history record already in existence in the table (For example, if there was a day where your ETL did not load, and you wanted to backfill data for that day, you could run into this issue).
* Matt Caster does not consider this to be a defect, so it is unlikely to be improved upon. Per Matt Caster’s response to [PDI-3912](http://jira.pentaho.com/browse/PDI-3912):

*“This is one of these cases where there are 2 possibilities:*

*1) The data is handed to the "Dimension Lookup/Update" step out-of-order. Certainly the way the test is set up this is the case.*

*The changed date goes from May 17th to May 16th 2010.*

*Out-of-order processing of data is actually not supported in a slowly changing dimension. That's more accurately handled by a Data Vault or something of that nature.*

*I would recommend adding a sort somewhere to make sure the data arrives in the right order in the "Dimension Lookup/Update" step.*

*2) The changed data time on the source system is incorrect. I've seen this happen personally where data was being updated on several systems at the same time and the clocks were slightly different. Another possibility is that the changed date is not always updated by the software of the source system (also seen that). If the changed date is unreliable on the source system, please consider not using it as a provider for the date ranges. Rather simply add it as an attribute.*

*Whatever the case, it set me on the wrong track for a second, but I don't think this is a bug.”*

*“Records are delivered out of order, this is not supported.”*

### JNDI Browse Limitation (v4.2.1)

When leveraging JNDI connections in PDI/Spoon v4.2.1, you will see the following error when trying to “Browse” the target table of an Insert / Update step:

Unable to retrieve database information because of an error   
        at org.eclipse.jface.operation.ModalContext$ModalContextThread.run (ModalContext.java:113)   
        at org.pentaho.di.ui.core.database.dialog.GetDatabaseInfoProgressDialog$1.run (GetDatabaseInfoProgressDialog.java:72)   
        at org.pentaho.di.core.database.DatabaseMetaInformation.getData (DatabaseMetaInformation.java:371)   
        at org.pentaho.di.core.database.Database.getTablenames (Database.java:3868)   
        at org.pentaho.di.core.database.Database.getTablenames (Database.java:3873)   
        at org.pentaho.di.core.database.Database.getTableMap (Database.java:3900)

The error is fairly benign. While you are unable to browse, you can manually enter the table name.  You will then be able to select columns from the drop-down list after the table name is entered.

## Oracle

## Microsoft SQL Server

# Technical Design Decisions

## Main / Implementation Structure

# Acronym List

|  |  |
| --- | --- |
| ERD | Entity-Relationship Diagram |
| PDI | Pentaho Data Integration (a.k.a. Kettle) |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# 

# References

[1] Kinsley, D. “testkitchen – Testing Framework for Pentaho PDI/Kettle.” June 2010. <https://code.google.com/p/testkitchen/>.