BPM Developer’s Guide for

SVN

Devin Heimbuch

April 23, 2020

# Table of Contents

[Table of Contents 2](#_Toc38522456)

[1 Summary 3](#_Toc38522457)

[2 CORE 3](#_Toc38522458)

[3 Corp code 3](#_Toc38522459)

[4 Locks 3](#_Toc38522460)

[5 Versioning 3](#_Toc38522461)

[5.1 PS/SQL 3](#_Toc38522462)

[5.2 Unix scripts 3](#_Toc38522463)

[5.3 Kettle Code 4](#_Toc38522464)

[6 Folders 4](#_Toc38522465)

[6.1 BPM 4](#_Toc38522466)

[Create DB 4](#_Toc38522467)

[Patch 4](#_Toc38522468)

[6.2 Trunk 5](#_Toc38522469)

[Archive 5](#_Toc38522470)

[Kettle8 5](#_Toc38522471)

[7 Contact Center Code 5](#_Toc38522472)

# Summary

The purpose of using SVN is to have a the current copy and history of every script used in the production environment.

# CORE

This is code that is installed on all projects part of DB set up, BPM Queue, Admin packages etc

# Corp code

Both DB and ETL have Corp Code, this is code that was deployed to multiple projects without modifications. when looking for a file, your need to look in both corp and the project directory, if the file exists in project, it is to be used, if it does not, then the corp file is to be used. you should also look for the svn file url keyword on the Production server, it will tell you where that file came from in SVN. Before making changes, you should resolve that the file in SVN is indeed the correct file, correct version, etc

# Locks

Before making any changes to a file, you should lock it, if a lock already exists, you must first resolve the lock with the developer that currently has it. After Production deployment you must release the lock and write a log entry. The log entry should include the jira number and a brief description of the change. For KTR give the steps that was modified, for a package give the procedure or function change

# Versioning

## PS/SQL

Please open and look for “keyword” – you will find directions how to set up versioning on pl/sql:

svn://svn-staging.maximus.com/dev1d/maxdat/BPM/doc/ BPM\_Developers\_Guide.docx

## **Unix scripts**

Simply add this to the top of the script:

# ================================================================================

# Do not edit these four SVN\_\* variable values.  They are populated when you

#     commit code to SVN and used later to identify deployed code.

#   SVN\_FILE\_URL varchar2(200) := '$URL$';

#   SVN\_REVISION varchar2(20) := '$Revision$';

#   SVN\_REVISION\_DATE varchar2(60) := '$Date$';

#   SVN\_REVISION\_AUTHOR varchar2(20) := '$Author$';

# ================================================================================

## Kettle Code

Add this in Extended Description (Right click on job or transformation, select properties)

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

Do not edit these four SVN\_\* variable values.  They are populated when you commit code to SVN and used later to identify deployed code.

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

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

SVN\_FILE\_URL varchar2(200) := '$URL$';

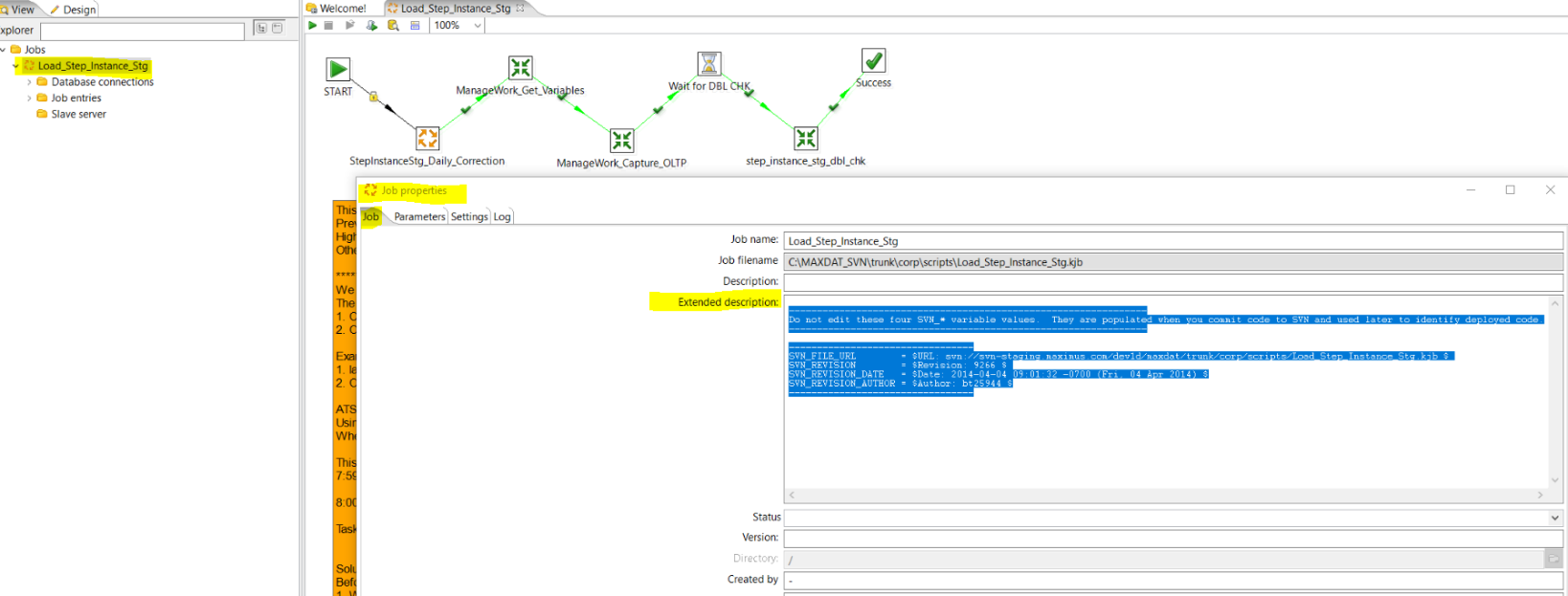
SVN\_REVISION varchar2(20) := '$Revision$';

SVN\_REVISION\_DATE varchar2(60) := '$Date$';

SVN\_REVISION\_AUTHOR varchar2(20) := '$Author$';

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

Example –



# Folders

## BPM

Create DB - This should be the code required to build an exact replica of production today. any patch created to add, change, remove db objects, must also be updated in these files

Patch - This is file required to update objects or data in the database, any objects changes made here must also be added to the creeatdb script. there are specific naming conventions required

Naming ex.

YYYYMMDD\_HHMM\_{Object\_Name}

20191018\_0858\_EMRS\_D\_PLAN.sql

## Trunk

Archive - Will hold all 4.2 that is no longer needed - This is going to be done soon

Kettle8 – All code for ETL server including Cron, kettle.properties, shared.xml, Unix Scripts, KJB, KTR

# Contact Center Code