CalSAWS

California Statewide Automated Welfare System

Design Document

CA-207395 | DDID 1041, 1045, and 1046

Updates to State Forms Templates

CalSAWS	DOCUMENT APPROVAL HISTORY	
	Prepared By	Pramukh Karla
	Reviewed By	Amy Gill

DATE	DOCUMENT VERSION	REVISION DESCRIPTION	AUTHOR
09/01/2020	1.0	Initial Revision	Pramukh Karla
10/1/2020	2.0	Revised DDID 1046 in Section 4.1	Amy Gill

Table of Contents

1 OVERVIEW

1.1 Current Design

The 57 Migration Counties will be inheriting the State Forms that generate from LRS/CalSAWS Template Repository. There exists State Forms which include System, Los Angeles County, GAIN Services, Agency, or County Specific address references.

1.2 Requests

Update and/or remove any System, Los Angeles County, GAIN Services, Agency, or County Specific address references from State Forms which can then be used by all Counties.

1.3 Overview of Recommendations

1. Update and/or remove any System, Los Angeles County, Gain Services, Agency, or County Specific address references.
2. Update the forms to use updated NA Back 9 (NA_BACK9_FRAGMENT) template to populate the Hearing address and Legal Aid address dynamically.

1.4 Assumptions

1. Existing Forms trigger conditions will NOT be updated.
2. Each of the 57 Migration Counties will have one record for the Hearing address and one record for the Legal Aid address.

2 RECOMMENDATIONS

2.1 Update State Forms Templates

2.1.1 Overview

The State Forms templates currently in LRS/CalSAWS will be used by all counties. Update the System, Los Angeles County, GAIN Services, Agency, or County Specific address references on the State Forms templates. Update the forms to use updated NA Back 9 (NA_BACK9_FRAGMENT) template to populate the Hearing address and Legal Aid address based on what county the case is managed in.

2.1.2 Description of Change

1. Update the State Forms templates to change System, Los Angeles County, GAIN Services, Agency, or County Specific address references with the updates mentioned for the list of templates in the **Section 3.0 Supporting Document #1**.

2. Update the State Forms templates to use below mentioned NA Back 9 which will populate Header address and Legal Aid address dynamically. The Hearing address and Legal Aid address listed on the Correspondence List page will be used for the 57 Migration Counties. For Los Angeles County, it will continue to use the existing logic of retrieving the Hearing address and Legal Aid address.

   Fragment Name: NA_BACK9_FRAGMENT
   Fragment Id: 670

   Existing Language: English, Spanish, Armenian, Arabic, Cambodian, Chinese, Farsi, Filipino/Tagalog, Hmong, Lao, Korean, Russian, Vietnamese

   Refer to **Supporting Document #2** for threshold language translations for updated verbiage.

3 SUPPORTING DOCUMENTS

Number	Functional Area	Description	Attachment
1	Client Correspondence	List of templates that require header updates.	List of Templates.xlsx
2	Client Correspondence	Threshold language translation for updated verbiage.	Translations.pdf

4 REQUIREMENTS

4.1 Migration Requirements

DDID #	REQUIREMENT TEXT	Contractor Assumptions	How Requirement Met
1041	The CONTRACTOR shall have an allowance of hours included in the Migration estimate for State form anatomy and cosmetic updates.	**Original:** - 40 State Forms between C-IV and CalWIN will need to be migrated into CalSAWS. - Form Anatomy is comprised of the following components: - The Header, which contains; Address Lines, County of Title, Form Title and Pre-populated fields such as Work Name and Case Number. - The Body, which contains; Static Text, Tables, Checkboxes, pre-populated fields and/or test language translations and space for barcodes. - The Footer, which contains; Form Number, Version Number, Page Number and space for barcodes. - Cosmetic updates are slight modifications to the form which do not alter the message or generation/population logic of the form. For example, font size change, line thickness, spacing	Templates updated as specified in the requirement.

DDID #	REQUIREMENT TEXT	Contractor Assumptions	How Requirement Met
		between form components; Header, Body, Footer. - Refer to attachment for an inventory listing of the C-IV and LRS Forms as of July 2018. **Revised:** Cosmetic updates are slight modifications to the form which do not alter the message or generation/population logic of the form. For example, font size change, line thickness, spacing between form components; Header, Body, Footer.	
1045	The CONTRACTOR shall revise the State form names and numbers to match the corresponding form name and number provided by the State.	- Due to the removal of DDID #1043 from the CalSAWS SOR which accounted for the CONTRACTOR's research time to identify the State forms requiring updates, the CONSORTIUM will need to provide the list of State forms requiring form name and number updates by July 1, 2020 for the CONTRACTOR to meet design, build and system test milestones. - It is assumed no more than 340 State forms shall have the form names and numbers revised to match the corresponding form name and number provided by the State.	Templates updated as specified in the requirement.
1046	**Original:** The CONTRACTOR shall make the following changes to Forms: 1. Update the County Agency Name and Address on all Form Headers to	• Forms are developed using headers that are shared across multiple forms, therefore the testing approach will be conducted by validating a sample of the high volume forms used by each unique header to	Templates updated as specified in the requirement.

DDID #	REQUIREMENT TEXT	Contractor Assumptions	How Requirement Met
	display the County Name as a dynamic field to support Multi-County Use. 2. Remove the LA County logo from the following State Form headers: • ARC 1 - Statement of Facts Supporting Eligibility for Approved Relative Caregiver Funding Option Program • ARC 2 - Redetermination Statement of Facts Supporting Eligibility for ARC Funding Option Program • WTW 8 - Student Financial Aid Statement Welfare-To-Work Supportive Services • RFA 100 - Emergency Caregiver Funding Approval • RFA 100A - Emergency Caregiver Funding Discontinuance • NA 1261 - Notice of Action Fiscal Form **Revised:** The CONTRACTOR shall update the County Agency Name and Address on all forms to display the County Name as a dynamic field to support Multi-County Use.	confirm the requirement has been met.	

CalSAWS

California Statewide Automated Welfare System

Design Document

SCR CA-213978 – California Child Support Automated System (CSSAS) Outbound Interface - Performance Improvement

CalSAWS	DOCUMENT APPROVAL HISTORY	
	Prepared By	Chris Carandang
	Reviewed By	Balakumar Murthy, Karthikeyan Krishnamoorthy, Chris Larson, Edgars Reinholds

DATE	DOCUMENT VERSION	REVISION DESCRIPTION	AUTHOR
07/13/2020	1.0	Initial draft	CC

Table of Contents

1 OVERVIEW

This document outlines the changes to be implemented in the California Child Support Automated System (CSSAS) batch jobs to improve the execution time of the CSSAS Load Balancer, and Update Writer Thread jobs.

1.1 Current Design

Currently, the CSSAS Load Balancer, and Update Writer Thread jobs run nightly. The Load Balancer searches for Child Support cases, assigns thread number on each record, and loads data into a staging database table. The CSSAS Update Writer Thread job reads data from the staging table, processes and writes to an outbound transaction table used for transmitting data to Child Support.

1.2 Requests

1. Refactor the driving query of CSSAS Load Balancer job (PB00F1610) to improve the execution time of populating the staging table.
2. Modify the CSSAS architecture to utilize the Memcache framework.

1.3 Overview of Recommendations

The existing CSSAS batch jobs will be modified to utilize the Memcache framework for improving job execution time.

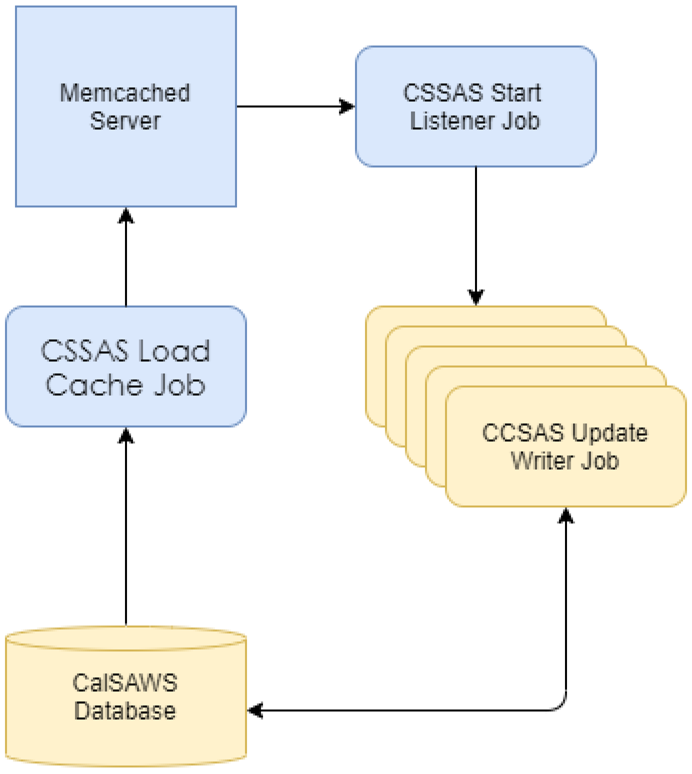


Fig 1.1 CCSAS Memcache Overview

1.4 Assumptions

1. Data volume and data format exactly match the current interface.
2. Existing CSSAS business logic will not be modified.

2 RECOMMENDATIONS

2.1 Modify the CSSAS Load Balancer Job (PB00F1610)

2.1.1 Overview

The CSSAS Load Balancer job runs nightly to search for new and update referrals child support cases. It loads these data into the staging table and assigns thread number on each row to be processed by the CSSAS Update Writer Thread job.

2.1.2 Description of Changes

Modify the driving query of the CSSAS Load Balancer job (PB00F1610) to insert data directly to the staging table without assigning thread number.

2.1.3 Execution Frequency

No change.

2.1.4 Key Scheduling Dependencies

No change.

2.1.5 Counties Impacted

All counties.

2.1.6 Data Volume/Performance

Current Max run time: ~35 mins for ~500k records.

Anticipated performance Improvement: 20%

2.1.7 Failure Procedure/Operational Instructions

The Batch and Tech Operation Support Teams will evaluate errors, diagnose the issue and work with the appropriate teams to the resolve the failure.

2.1.8 Test Validation

The CSSAS Load Balancer job with pre and post change will be executed in the same test environment. The number of records and contents populated in the staging table should match. Performance comparison between the runs will be published as part of validation.

2.2 Create new CSSAS Load Cache Job

2.2.1 Overview

The new batch job will load the CSSAS data from the staging table into Memcache. The Memcache will be used by CSSAS Thread jobs to pick up cases for more efficient processing.

2.2.2 Description of Changes

Create a new batch job to load CSSAS data from the staging table into Memcache. It will have its own port number, staging table record id to be used as key in Memcache.

2.2.3 Execution Frequency

Daily.

2.2.4 Key Scheduling Dependencies

The predecessor for this job is the CSSAS Load Balancer Job (PB00F1610).

The successor for this job is the CSSAS Start Listener Job.

2.2.5 Counties Impacted

All counties.

2.2.6 Data Volume/Performance

Not applicable.

2.2.7 Failure Procedure/Operational Instructions

The Batch and Tech Operation Support Teams will evaluate errors, diagnose the issue and work with the appropriate teams to the resolve the failure.

2.3 Create new CSSAS Start Listener Job

2.3.1 Overview

The new batch job will open socket connection and start monitoring for CSSAS Update Writer Thread jobs.

2.3.2 Description of Changes

Create a new start listener job.

2.3.3 Execution Frequency

Daily.

2.3.4 Key Scheduling Dependencies

The predecessor for this job is the new CSSAS Load Cache Job.

The successor for this job is the CSSAS Update Writer Thread Job (PB00F1611- PB00F1730).

2.3.5 Counties Impacted

All counties.

2.3.6 Data Volume/Performance

Not applicable.

2.3.7 Failure Procedure/Operational Instructions

The Batch and Tech Operation Support Teams will evaluate errors, diagnose the issue and work with the appropriate teams to the resolve the failure.

2.4 Create new CSSAS Stop Listener Jobs

2.4.1 Overview

The new batch jobs will stop the CSSAS batch listener job.

2.4.2 Description of Changes

Create a new force stop listener job. It is an on-demand job that will forcefully stop the listener job.

Create a new graceful stop listener job that will be a scheduled batch job to stop the listener job.

2.4.3 Execution Frequency

Daily.

2.4.4 Key Scheduling Dependencies

Only the graceful stop listener job will be scheduled.

The predecessor for this job is the CSSAS Update Writer Thread Job (PB00F1611- PB00F1730).

The successor for this job is the CSSAS Interface Writer Job (POXXF1601).

2.4.5 Counties Impacted

All counties.

2.4.6 Data Volume/Performance

Not applicable.

2.4.7 Failure Procedure/Operational Instructions

The Batch and Tech Operation Support Teams will evaluate errors, diagnose the issue and work with the appropriate teams to the resolve the failure.

2.5 Modify the CSSAS Update Writer Thread Job (PB00F1611- PB00F1730)

2.5.1 Overview

The CSSAS Update Writer Thread job runs nightly. It picks up cases from the staging table to process the referral changes and stores data into the Child Support outbound transaction table for the final job to create an outbound file.

2.5.2 Description of Changes

Modify the CSSAS Update Writer Thread job to pick up cases that resides in Memcache.

2.5.3 Execution Frequency

No change.

2.5.4 Key Scheduling Dependencies

The predecessor for this job is the new CSSAS Start Listener Job.

The successor for this job is the new CSSAS Stop Listener Job.

2.5.5 Counties Impacted

All counties.

2.5.6 Data Volume/Performance

Current Max run time: ~45 mins per thread processing ~40k records.

Currently CalSAWS has 120 threads.

Anticipated performance Improvement: 20%

With this new approach the jobs/threads will be scalable to improve the performance on larger caseloads.

2.5.7 Failure Procedure/Operational Instructions

The Batch and Tech Operation Support Teams will evaluate errors, diagnose the issue and work with the appropriate teams to the resolve the failure.

2.5.8 Test Validation

The CSSAS Load Balancer job with pre and post change will be executed in the same test environment. The number of records and contents populated in the staging table should match. Performance comparison between the runs will be published as part of validation.

3 SUPPORTING DOCUMENTS

Number	Functional Area	Description	Attachment

4 REQUIREMENTS

4.1 Project Requirements

REQ #	REQUIREMENT TEXT	How Requirement Met

4.2 Migration Requirements

DDID #	REQUIREMENT TEXT	Contractor Assumptions	How Requirement Met

CalSAWS

California Statewide Automated Welfare System

Design Document

SCR CA-213980 – Convert to Stream Architecture – MEDS AP19 Transaction

CalSAWS	DOCUMENT APPROVAL HISTORY	
	Prepared By	Sivagami Nachiyappan
	Reviewed By	Balakumar Murthy, Chris Larson, Karthikeyan Krishnamoorthy

DATE	DOCUMENT VERSION	REVISION DESCRIPTION	AUTHOR
6/12/2020	1.0	Initial draft	Sivagami Nachiyappan

Table of Contents

1 OVERVIEW

The AP19 transaction is used to verify the citizenship status and identity of an individual for a given person from MEDS. The AP19 transaction will be rearchitected to leverage the "Stream Processing Architecture."

1.1 Current Design

Currently, the AP19 MEDS job runs nightly. The data changes are made through the CalSAWS online application during the day. However, the same data changes are captured, processed into AP19 transaction records, and sent to MEDS as part of an outbound file during the nightly batch process.

1.2 Requests

Refactor AP19 transaction to leverage "Stream Processing Architecture" to run during business hours. This will eliminate the AP19 transaction processing batch job and reduces the number of batch jobs running during the batch window to accommodate all 58-counties in the CalSAWS system.

1.3 Overview of Recommendations

The Citizenship verification (AP19) transaction will be rearchitected to leverage the "Stream Processing Architecture".

1.4 Assumptions

1. The new streaming application will have no functional differences from the existing AP19 batch job
2. Data volume and data format should exactly match the current interface
3. Existing AP19 batch job will not be modified
4. As per current implementation, AP19 transactions will still be sent to MEDS in the nightly outbound file

2 RECOMMENDATIONS

2.1 MEDS Transaction – AP19

2.1.1 Overview

The Citizenship verification (AP19) Transaction Background

Daily MEDS transactions are generated during a nightly batch process. The Citizenship verification (AP19) transaction, which is generated to report the citizenship Status verification and identity of an individual for a given person from MEDS.

Current Architecture

Currently, there exist nightly batch jobs that are used to verify the citizenship status and identity of an individual for a given person from MEDS. It is captured in intermediate tables in the CalSAWS.

The Citizenship verification (AP19) transaction job queries these intermediate tables to verify the citizenship and identity of an individual and generate AP19 transactions. These transactions are staged in the MEDS transaction table for the final job to create an outbound file with all the transactions.

2.1.2 Description of Changes

1. A connector will be set up to gather the Vital statistics inserts, updates, and deletes information through the application.
2. Below is the list of source tables from which connector will be gathering and sending corresponding IDs (System generated primary key for the tables) to the source topics for downstream processing. No PII data will be stored in the source topics.
    i. VITAL_STAT_REQ
    ii. VITAL_STAT
    iii. VITAL_STAT_BIRTH_CERT
    iv. VITAL_STAT_DOC
3. A streaming application will be monitoring the source topics, process the data, and sends the IDs of cases that need to be sent to MEDS to a sink topic.
4. A consumer application will process the data from the sink topic, generate AP19 transactions using the existing logic, and stage them in the MEDS transaction table for the final job to create an outbound file. In alignment with existing implementation, no duplicate records will be sent to MEDS.

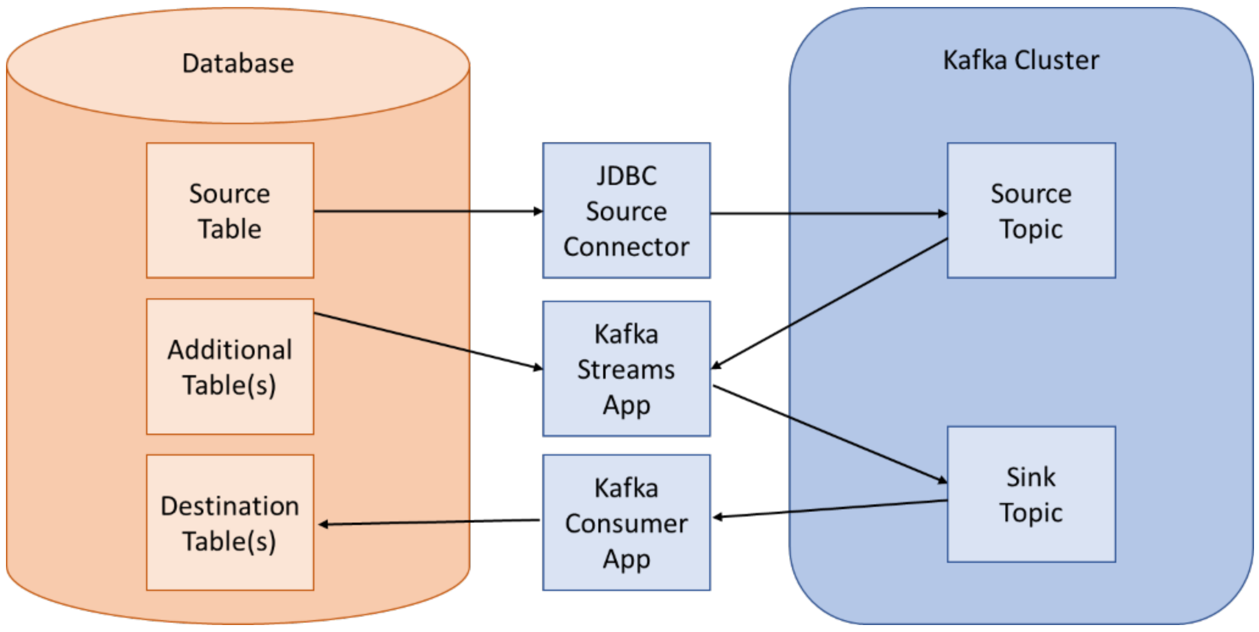


Figure 1. Streams Processing Architecture for AP19 Transaction

2.1.3 Execution Frequency

Source Connector and Streams Application will be running 24X7.

Consumer Applications will be scheduled to run every hour.

2.1.4 Key Scheduling Dependencies

Consumer Application will be set as the predecessor to the MEDS outbound writer job.

2.1.5 Counties Impacted

All counties.

2.1.6 Data Volume/Performance

N/A

2.1.7 Failure Procedure/Operational Instructions

The Batch\Tech Operation Support Team will evaluate errors, diagnose the issue and work with the appropriate teams to resolve the failure.

2.1.8 Production Validation

After production deployment, both new AP19 streaming applications and existing AP19 jobs will be running parallel, after comparing the results existing job will be decommissioned.

Below is the production validation and comparison flow:

1. A separate MEDS file will be generated from the AP19 streaming application and will be made available in S3 to compare records with a nightly AP19 job.
2. Generated MEDS file from AP19 streaming application will never be sent to meds till validation is complete.
3. Once the validation is complete, nightly AP19 batch jobs will be decommissioned gracefully and AP19 streaming application will start sending transactions to MEDS through the nightly file.
4. In addition to file comparison, a new batch job will be scheduled to send an email with processed record counts of streaming and nightly AP19 job for comparison.

2.1.9 Enable Streaming Architecture/Operational Instructions

After validation, the nightly AP19 job will be turned off and the AP19 streaming application will be configured to send a transaction to MEDS through the nightly file. With client approval, these actions will be taken through BPCRs\BSCRs.

3 SUPPORTING DOCUMENTS

Number	Functional Area	Description	Attachment

4 REQUIREMENTS

4.1 Project Requirements

REQ #	REQUIREMENT TEXT	How Requirement Met

4.2 Migration Requirements

DDID #	REQUIREMENT TEXT	Contractor Assumptions	How Requirement Met

CalSAWS

California Statewide Automated Welfare System

Design Document

CA-215722

Eligibility Performance Refactor Phase II

CalSAWS	DOCUMENT APPROVAL HISTORY	
	Prepared By	Henry Thamas, Cassie Mestayer
	Reviewed By	Richard Weeks, Gopal Vedula, Prakash Thota

DATE	DOCUMENT VERSION	REVISION DESCRIPTION	AUTHOR
7/16/2020	1.0	Initial Version	Henry Thamas, Cassie Mestayer

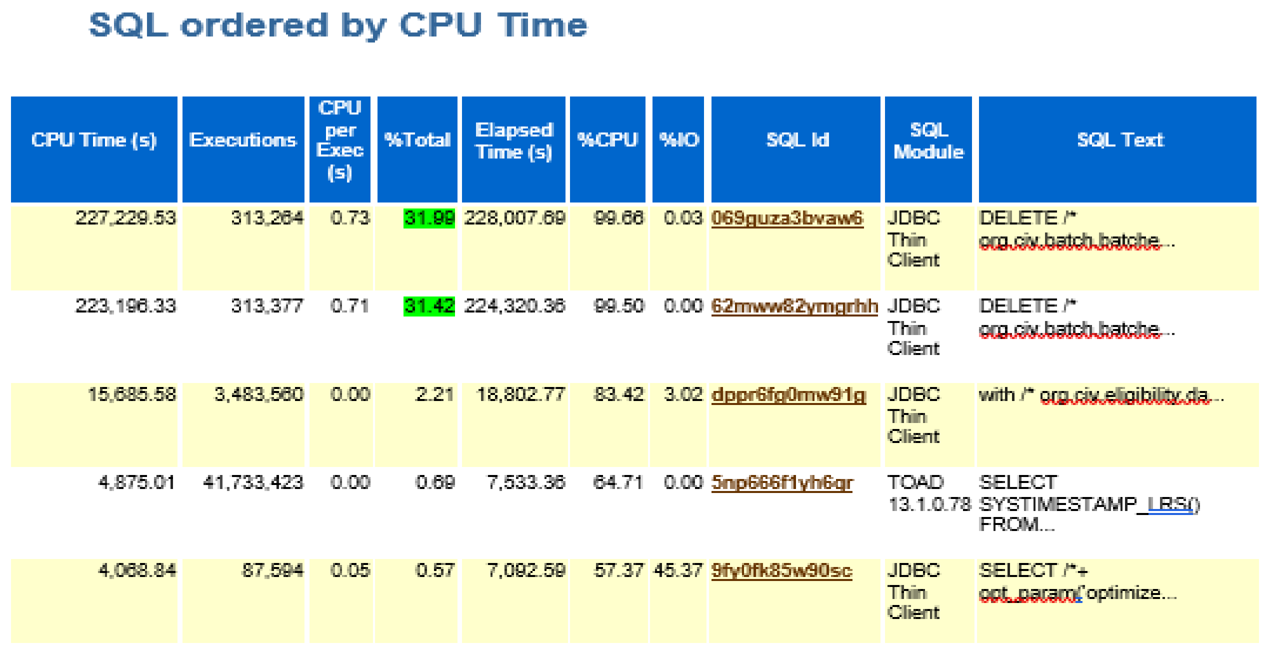
Table of Contents

1 OVERVIEW

1.1 Current Design

Prior to a Batch EDBC (PB00E501) running, cases/programs are inserted into the database tables called SYS_TRANSACT and TRANS_PERS. These tables contain data about how the batch will process these records, like if it's Batch EDBC, which benefit month will be run or if all the programs on the case will be processed too. When the Batch EDBC job starts, cases/programs are picked up from the SYS_TRANSACT table with the Batch EDBC type code (CT626_BE) and processed through the EDBC. Once the case/program has run through the Batch EDBC, there are a few post activities that happen which include removing the case/program from the SYS_TRANSACT and TRANS_PERS table.

Currently the Batch EDBC process takes 13.25 hours to complete approximately 2 million cases/2.5 million programs (total of active cases) for LA County. Batch Processing is taking 40% of the time in overall. This design is focused to reduce the Batch EDBC processing time as part of phase II to reduce the overall time taken to process an EDBC. The bulk of the Batch EDBC time comes from Batch Post Activities, 4 hours and 58 mins for the 2 million cases. These activities include Batch Eligibility logging and removing processed triggers. Specifically, the delete SQL statements for TRANS_PERS and SYS_TRANSACT have major impact on overall performance with about 30% of the Batch Post Activities time taken for each statement. Below image shows the two delete statements taking the longest time out of the top 5 Batch Post Activities calls to the database.



SQL ordered by CPU Time

CPU Time (s)	Executions	CPU per Exec (s)	%Total	Elapsed Time (s)	%CPU	%IO	SQL Id	SQL Module	SQL Text
227,229.53	313,264	0.73	31.99	228,007.69	99.66	0.03	069guza3bvaw6	JDBC Thin Client	DELETE /* org.civ.batch.batche...
223,196.33	313,377	0.71	31.42	224,320.36	99.50	0.00	62mww82ymgrhh	JDBC Thin Client	DELETE /* org.civ.batch.batche...
15,685.58	3,483,560	0.00	2.21	18,802.77	83.42	3.02	dppr6fg0mw91g	JDBC Thin Client	with /* org.civ.eligibility.da...
4,875.01	41,733,423	0.00	0.69	7,533.36	64.71	0.00	5np666f1yh6gr	TOAD 13.1.0.78	SELECT SYSTIMESTAMP_LRS() FROM...
4,068.84	87,594	0.05	0.57	7,092.59	57.37	45.37	9fy0fk85w90sc	JDBC Thin Client	SELECT /*+ opt_param('optimize...

Figure 1.1 – Top 5 database calls during the Post Batch Activities

After testing and research, a solution was proposed to move these activities outside of Batch EDBC.

1.2 Requests

1. Reduce EDBC Time and Batch Processing Time by moving the mentioned batch post activities outside the Batch job. Please see the Appendix for a review of the batch post activities statistics.

1.3 Overview of Recommendations

1. Modify Batch EDBC to update the SYS_TRANSACT and TRANS_PERS table records instead of deleting.
2. Delete the SYS_TRANSACT and TRANS_PERS table records outside of the batch EDBC transactions.

1.4 Assumptions

1. This will not change why records(cases/programs) are removed from the batch SYS_TRANSACT and TRANS_PERS tables.
2. This SCR is not changing or impacting how the Batch EDBC processes a case or program. The EDBC results will not be impacted.

2 RECOMMENDATIONS

2.1 Batch – Create a new Batch job to delete records processed by the Batch EDBC

2.1.1 Overview

The EDBC Batch Job will no longer delete processed records and will instead UPDATE records as Processed ("BEP"). The deletion of these record will be handled by a new batch job and will be performed after the EDBC Batch is completed. Any record already processed will not be picked up again when the EDBC Batch is restarted.

In the event of All Program mode for Batch EDBC, the program that was inserted into SYS_TRANSACT would be updated to "BEP" under the same conditions as it would be deleted prior to this SCR. This SCR will not change the conditions in which a program/case is removed from SYS_TRANSACT.

The DCFS EDBC batch shares the common logic with the regular EDBC batch to delete transactions from the SYS_TRANSACT and TRANS_PERS table. This change will also be applicable to the DCFS EDBC batch.

2.1.2 Description of Change

a. Modify the Batch EDBC (PB00E501) logic to not delete records from the SYS_TRANSACT or TRAN_PERS tables when finished processing each program/case. These records will now be updated to indicate that they have been processed by the Batch EDBC.

   i. Update the type code on SYS_ TRANSACT to 'Batch Eligibility Processed' (CT626_BEP) for processed records.

   Note: TRANS_PERS records are linked to a single SYS_TRANSACT record and do not need to be updated/deleted here.

b. Create a new Batch job which deletes records already processed by the Batch EDBC (type code = 'BEP') from the SYS_ TRANSACT and TRANS_PERS tables.

   Note: When Batch EDBC is restarted, the prior records will have been marked to already processed ('BEP') and will not be picked in the new run.

2.1.3 Execution Frequency

This batch job will be scheduled to run daily for all CalSAWS business days, excluding Sundays and Holidays.

Note: The first job will be delayed by a week and then scheduled every day there after.

2.1.4 Key Scheduling Dependencies

This should process after the DCFS EDBC run.

2.1.5 Counties Impacted

All CalSAWS Counties

2.1.6 Data Volume/Performance

N/A

2.1.7 Failure Procedure/Operational Instructions

The Batch Operations Support Team will evaluate errors, diagnose the issue and work with the appropriate teams to the resolve the failure.

2.2 Modify the SYS_TRANSACT History Trigger

2.2.1 Overview

When records are inserted/updated/deleted from SYS_TRANSACT, a trigger will create records in the SYS_TRANSACT_HST table. With the modification of how cases/programs are being removed from the SYS_TRANSACT table during Batch EDBC, there will be a lot of unnecessary records inserted into SYS_TRANSACT_HST table when an update or delete is made.

2.2.2 Description of Change

Modify the SYS_TRANSACT_HST_TRG to not insert into SYS_TRANSACT_HST after a row is updated or deleted.

Note: The Trigger will still insert in the history table when a row is inserted into the SYS_TRANSACT table.

3 TESTING PLAN

3.1 Overview

1. **Key data point comparison regression test:**
   Regression testing of entire case load will be performed using the process that is currently in place (app-dev regression test process) for every major CalSAWS release. This testing will be performed at both the Assembly Test phase, and the System Test phase and will be run for the whole case load against all programs.

   This involves the following steps:
   a. BRG1 environment that will have latest production build **without** Batch Eligibility improvements.
   b. BRG2 environment that will have latest production build **with** Batch Eligibility improvements.
   c. Both environments will have the latest production data snapshot.
   d. EDBC batch is run in both environments.
   e. Statistics (time taken to complete Batch EDBC Post Activities) will be gathered and analyzed. Goal is to make sure the time taken for batch EDBC with the modified is less than the time taken without the modifications.
   f. SYS_TRANSACT Results (records updated instead of deleted) will be compared to confirm there is no impact because of making these changes.
   g. A sample case list will be provided from the cases processed through the regression test.
2. **Performance Testing:**
   a. To be performed during the System Test phase in a performance test environment.
   b. Performance testing is performed on the modified code base on the same caseload that is used for any major CalSAWS release.

c. Purpose of this is to make sure the frequency and time duration of function calls is as expected after the changes during the Batch EDBC performance testing
d. The new batch job to delete the processed records will also be measured to determine its performance.

3.2 Assumptions

1. Once conversion data becomes available the feasibility of additional testing at a higher data volume can be reassessed.

2. There is no negative impact in the system with respect to notices generated, benefits issued, interfaces, and reports generated in CalSAWS.

4 APPENDIX

Where is the Maximum Time taken in Batch and EDBC Processing?

Below table explains the time taken for each category.

EDBC Processing Categories	**Batch Run 04/07/2020 without changes**	**Batch Run 04/08/2020 with Phase 1 changes**	**Comments**
Build	1 hr 31 min 55 sec	1 hr 19 min 19 sec	
Run (Build + Run)	2 hr 45 min 17 sec	2 hr 33 min 7 sec	(This included time measured in Build)
Authorize	2 hr 53 min 16 sec	2 hr 47 min 39 sec	
NOA	2 hr 22 min 16 sec	2 hr 9 min 18 sec	
EDBC Total Time	8 hr 1 min 49 sec	7 hr 30 min	
Batch Processing Categories			
Journal	0 hr 1 min 18 sec	0 hr 1 min 18 sec	
Post Activities	5 hr 8 min 9 sec	4 hr 58 min 56 sec	
Job Controller	0 hr 0 min 40 sec	0 hr 0 min 42 sec	
Others	0 hr 30 min 00 sec	0 hr 25 min 00 sec	
Total	5 hr 10 min 8 sec	5 Hours	
Grand Total	13 hr 45 Min	12 Hr 55 Min	

What Shall we do?

As Eligibility is making changes in Phases to reduce the overall time taken. For example: moving the authorization EDBC level changes to Run EDBC, preview NOA logic in the batch to avoid duplicate loading of build outs etc. Changes should be done in the **Batch Post Activities** to reduce the overall time.

What is happening in Batch Post Activities?

1. **Batch Eligibility Log:** Inserts data into log tables whether batch has processed or not
2. **Remove Processed Triggers:** Delete triggers from log tables like SYS_TRANSACT, TRAN_PERS.

Major Impacted SQL Queries: Delete SQL statements for TRANS_PERS and SYS_TRANSACT have major impact on overall performance and the stats have been reviewed.

CalSAWS

California Statewide Automated Welfare System

Design Document

CIV Alfresco Document Migration

CalSAWS	DOCUMENT APPROVAL HISTORY	
	Prepared By	Jonathan Goldsmith
	Reviewed By	

DATE	DOCUMENT VERSION	REVISION DESCRIPTION	AUTHOR
5/26/2020	1.0	Initial Revision	

Table of Contents

1 OVERVIEW

1.1 Current Design

Both CIV Online and Batch systems currently use Alfresco to store and retrieve documents (.pdf, .xml, etc.). Alfresco consist of a web UI, FTP interface and an Alfresco Database. It is currently used by online and batch applications through CIV Interfaces, which uses the CIV architecture File Management Interface. The architecture File Management Interface uses open source API librabry CMIS to manage files in the Alfresco System.

1.2 Requests

The CalSAWS system uses AWS S3 for document storage. In order for the CalSAWS system to interact with previous CIV documents, the CIV Alfresco documents will need to be migrated over to the CalSAWS AWS S3.

1.3 Overview of Recommendations

Create new Batch jobs that will copy CIV Alfresco documents over to CalSAWS S3 and record the new S3 document number into the database.

1.4 Assumptions

2 RECOMMENDATIONS

2.1 Add CMIS client to interact with Alfresco server

Update Architecture File Management code to add CMIS API client and new methods for interacting with Alfresco server. This will allow new Batch code to copy over C-IV Alfresco documents to CalSAWS S3.

2.2 Set up C-IV Alfresco instance to be used by Migration Batch Jobs

Set up a C-IV Alfresco instance with a copy of the Production Alfresco database data. The Alfresco instance will have access to a replicated mount drive of the Production files. This will allow the Migration Batch jobs to pull copies of the Production files without causing performance issues to the Production Alfresco server.

2.3 Migration job setup on AWS

The following describes the setup used for the document migration jobs:

1. AWS EC2 instances configured with role based AWS authentication. On ec2 /etc/profile.d directory, create file with Java environment variables and export SPRING_PROFILES_ACTIVE="ec2"
2. Deployment of the File Service code on ec2
3. Deployment of CalSAWS code with new file migration interface
4. Deployment of CalSAWS code with migration batch jobs
5. Batch job server and BicSuite scheduler setup
6. Schedule of required jobs in BicSuite

2.4 Migrate current documents from C-IV Alfresco to AWS S3

1. A DCR or similar process will populate temp tables based on existing CIV production data.
2. Migration jobs will take entries from temp tables and convert them updating the entry in the temp table with the new FMS id. Each table and column combination would have a job set with:
    a. Loader
    b. Listener
    c. Stop Work
    d. Stop Listener
    e. 1+ worker threads
    f. Multiple job sets can potentially write to the same temp table if multiple columns are being migrated on it. However, this is acceptable as the run time for those jobs is expected to be weeks vs hours for migration cutover.

Batch Job Set 1 (Migration/Copy):

Will migrate documents from AWS Alfresco to AWS S3

Separate temp tables will be updated with file id mappings

Temp Table List:

GENERATE_DOC_TEMP.ALF_FMS_NUM(FMS columnName)

The table will have the following common attributes:

id - this matching the original table on C-IV.

created_on

updated_on

created_by

updated_by

For each FMS column to migrate, the following will be required:

${columnName} - original FMS ID value from CIV

${columnName}_new - migrated FMS ID to retrieve the file in S3

${columnName}_md5 - MD5 of original file

${columnName}_md5_new - MD5 of migrated file

${columnName}_stat - status of the column on this record

Note: 'new' and 'stat' were chosen to avoid going over the 30 identifier limit for some columns to migrate like print_file where using more standard names like enclosure_alf_fms_num_stat_code or enclosure_alf_fms_num_conversion would go over the limit.

Note: column name pattern is the same as above for the below temp tables

GENERATE_DOC.ALF_XML_FMS_NUM

GENERATE_RPT_TEMP.ALF_FMS_IDENTIF

ENCLOSURE_TEMP.ALF_FMS_NUM

PRINT_FILE_TEMP.ALF_FMS_NUM

PRINT_FILE_TEMP. ENCLOSURE_ALF_FMS_NUM

ICT_IMG_TEMP.FMS_NUM

VLP_MNL_VERIF_REQ_TEMP.FMS_DOC_NUM_IDENTIF

Jobs

For each table/column combination, Job set example:

1. Temp table populator (e.g.: PGDAFNPREPARE) - copies the data from C-IV data source to temp table (on separate data source)
2. Copy Load Work (e.g.: CGDAFNLOAD) – load all work for the copy listener to use
3. Copy Listener (e.g.: CGDAFNLISTEN) – distributes work to copy worker jobs
4. Copy Stop Work (e.g.: CGDAFNSTOPWORK) – stops the copy work
5. Copy Stop Listener (e.g.: CGDAFNSTOPLISTEN) – stops the copy listener
6. Copy Workers (e.g.: CGDAFNWORK001-PGDAFNWORK999) – copy worker jobs

Job Status

The main statuses are:

PC (pending copy – Tech Architecture team owned process) - Populating job data from C-IV table will set the record to this status once the ID is available.

PV (pending verify - Tech Architecture team owned process) - Copy job will set the status to this once the file is copied.

There are additional intermediate or diagnosis status codes like Copy Error, which would prompt manual intervention or analysis.

2.5 Validate documents is AWS S3

Batch Job Set 2 (Verification):

There will be a batch job that will verify documents are the same in AWS S3 as C-IV Alfresco.

This job will utilize and update the same temp tables as above to verify documents stored in AWS S3.

Validation job set will use a md5 hash to validate documents and will update status in temp tables from PV (Pending Verify) to CM (Complete).

Jobs

Verify Load Work (e.g.: VGDAFNLOAD) – load all work for the verify listener to use
Verify Listener (e.g.: VGDAFNLISTEN) – distributes work to verify worker jobs
Verify Stop Work (e.g.: VGDAFNSTOPWORK) – stops the verify work
Verify Stop Listener (e.g.: VGDAFNSTOPLISTEN) – stops the verify listener
Verify Workers (e.g.: VGDAFNWORK001-PGDAFNWORK999) – verify worker jobs

Job Status

CM (complete – DBA team owned process) - Verify job will set the status to this once the MD5 checksums are verified. Records with all columns on this status are ready to go back to the production CalSAWS Database under the new column holding the FMS ID.

2.6 Cutover to AWS S3

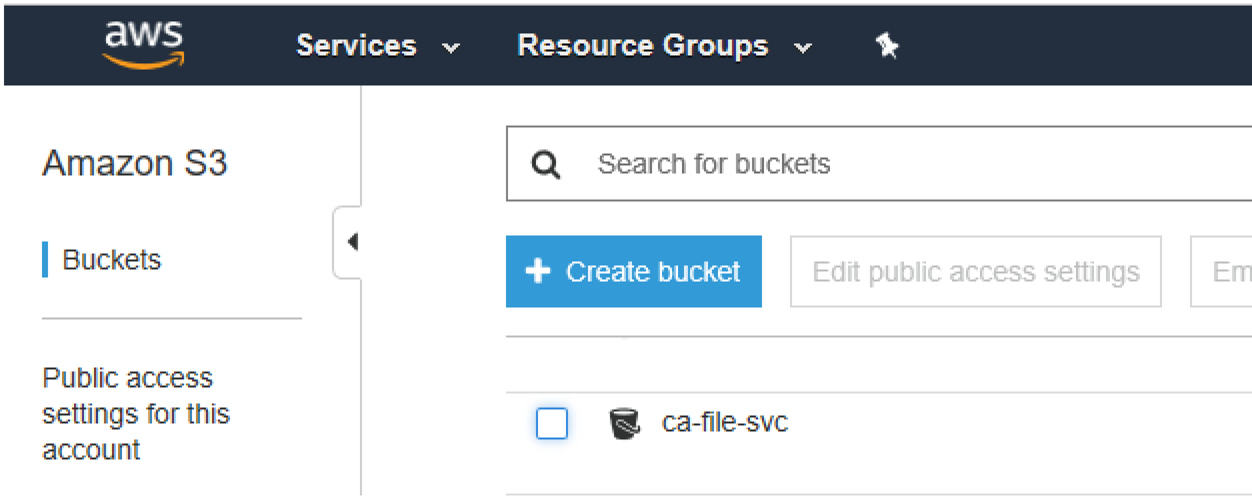
Batch Job 3 or DCR:

Steps as follows to merge the IDs from Temp tables(e.g. GENERATE_DOC_TEMP) back into the CalSAWS database tables(e.g. GENERATE_DOC, etc.):

1. DBA team will import temp table into CalSAWS DB
2. DBA team will disable updated_on triggers on target tables
3. DBA team will update by ID from ${tableName}_temp. ${columnName}_new into ${tableName}.${columnName}_new for the columns migrated for each record.
4. DBA team will rename old column to some other name (e.g. alf_fms_num -> alf_fms_num_old), rename new column to old column name (e.g. alf_fms_new -> alf_fms_num) [Report Name] Mockup

2.7 AWS S3 UI Document Access

Current users with Alfresco FTP access will have access to the AWS S3 Console Teams will use the AWS S3 console to search, view, download, and add documents to AWS S3.



AWS S3 console is used to manage files in the cloud by production operations teams

2.8 Document Migration Monitoring

A URL/Report will be created to query the TEMP table's for stats on the migration. This will return back the number of transferred documents for each TEMP table.

3 APPENDIX

[Include any supplementary items that my not fit in the Description section. Examples could include flow charts, lengthy code tables, etc.…]