Heritage to DIP Migration –

Issues & Planning for CRADC/Exclusions

Latest version: Friday 26th July 2019

# Summary

The campaigns run by the Analytics team rely on a set of SAS programs “CRADC/Exclusions”[[1]](#footnote-1) that runs on the heritage platform. A lack of disk-space and inadequate processing power there means that the overnight processing often fails or extends well into the next business day[[2]](#footnote-2).

The main output is a “Single view of Customer” – a detailed snapshot of the current customer-base. Lack of resources described above means that it has not been possible to keep a history of these snapshots, thus preventing its use by other areas of the business eg for monitoring trends over time.

One option in the short-medium term (1-2years) is to migrate the CRADC processing to the DIP to take advantage of plentiful storage and processing power to keep the Analytics team running more smoothly while the future workings of that team on the DIP are worked out. Assuming that continuing to run the code on heritage is not an option[[3]](#footnote-3), migration of the code to the DIP is an gives the following benefits:

* It eliminates the problems with running on the heritage platform (estimate 4x faster)
* Offers a way to familiarise staff with new SAS tools and version control
* It lets us deploy version 1 of Tina’s Customer Analytic Record CAR, especially if we use the faster DIP processing to build up a history of snapshots.
* It would also support the work of the Debt & Returns delivery team

|  |  |  |
| --- | --- | --- |
| What 🡪 | Migrated CRADC | CRADC Re-engineered using Refined layer |
| Assurance | High | V. High |
| Frequency | High | High |
| Engineering | Medium | High |
| Delivery | Q3 2019 | 2021? |

But the migration is not trivial: there is a long list of issues that must be addressed before a successful migration to the DIP can occur. This document suggests a plan of attack.

# The Experiment

I took a recent version of first overnight CRADC job and ran it against a subset of data (Tax Agents and their clients) on the heritage platform, a standalone laptop and on SAS Studio as available in the Lab environment to learn the perils and pitfalls of migrating code off the heritage platform and onto the DIP.

|  |  |
| --- | --- |
| Environment | Processing |
| Heritage | 6mins |
| 2015 i5 laptop | 10mins |
| SAS Studio on DIP Lab Viya | 1.5mins |

So the new platform is considerably quicker than heritage when running in single threaded mode, presumably a result of more modern hardware. Even further processing gains should be possible with parallelisation.

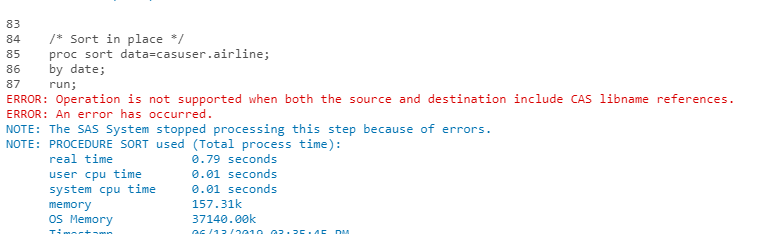
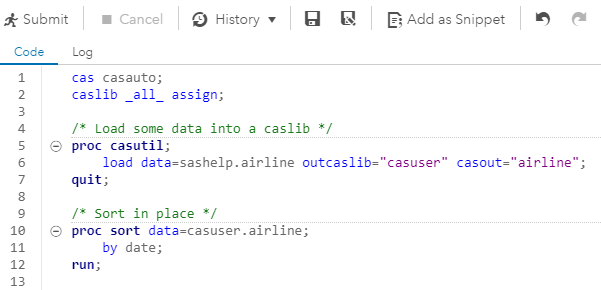
# Migration Issues

The CRADC code has been tended carefully over the years to capture complex business logic but an in-depth review shows that it lacks the structure required in a production environment[[4]](#footnote-4):

* Hardwired library names – preventing easy promotion from dev to test to prod
* Monolithic blocks of code with very few comments
* Inconsistent use of macros eg in-lined rather than use of autocall libraries
* No coding standards to speak of eg much coding is all UPPERCASE
* Bespoke SAS-based error-logging and notification macros have been used rather than using functionality of the underlying operating system or scheduling engine
* Reaching into personal schemas in EDW (read and write): campaign tables in 67AWCH, and struck-off companies in 17THHO, and temporary tables in the users home schema.
* Lack of awareness of sas data-management routines eg macro **%add\_sufix[[5]](#footnote-5)** re-writes a dataset to rename variables rather than just using proc datasets.
* The code runs against EDW/TDW and these tables are not the same as raw START tables - The TDW tables often have IRD\_NUMBER inserted as part of the processing and this is an intrinsic part of the logic in the CRADC/Exclusions code.
* Some processing is written as complex SQL pass-through queries eg FIRST Arrangements
* The heritage code is written against the SAS9 engine and does not run against a CASLIB engine- see below - so some reconfiguration of the Lab/Factory environment will be needed
* Weird mix of \_VIEW and \_VALL data – depending on the access available to the different staff writing the code. This makes it inconvenient for the platform team to take over the code developed by experts in the business unless the platform team have permissions to see everything.
* Mixed use of SYSDATE and Today() instead of macro variable Effective\_date.
* There also appears to be a security breach in the CRADC implementation – see section below

## Problems with running heritage code in DIP environment

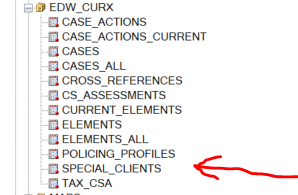
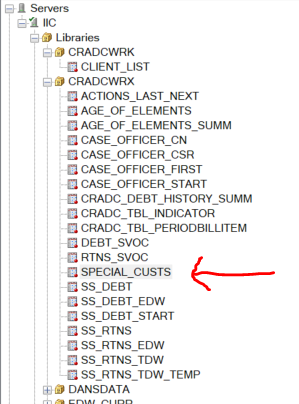
The assumption behind the current configuration of the SAS Studio Lab environment is that users will be using SAS’ Cloud Analytics Services (CAS) to take advantage of parallel processing and the large amount of HDFS storage available on the DIP. Unfortunately running heritage code against a CAS engine will not be possible without some rework because HDFS datasets cannot be updated in place. Eg one cannot sort in place a dataset referred to in a CASLIB.



Running heritage code against a traditional SAS9 engine makes more sense – but no large, persistent storage area has been set up in the current environment, and user home directories at present are very small. According to Matt Knight from SAS, this can be easily remedied by SAS Global Hosting team but requires some analysis of requirements beforehand.

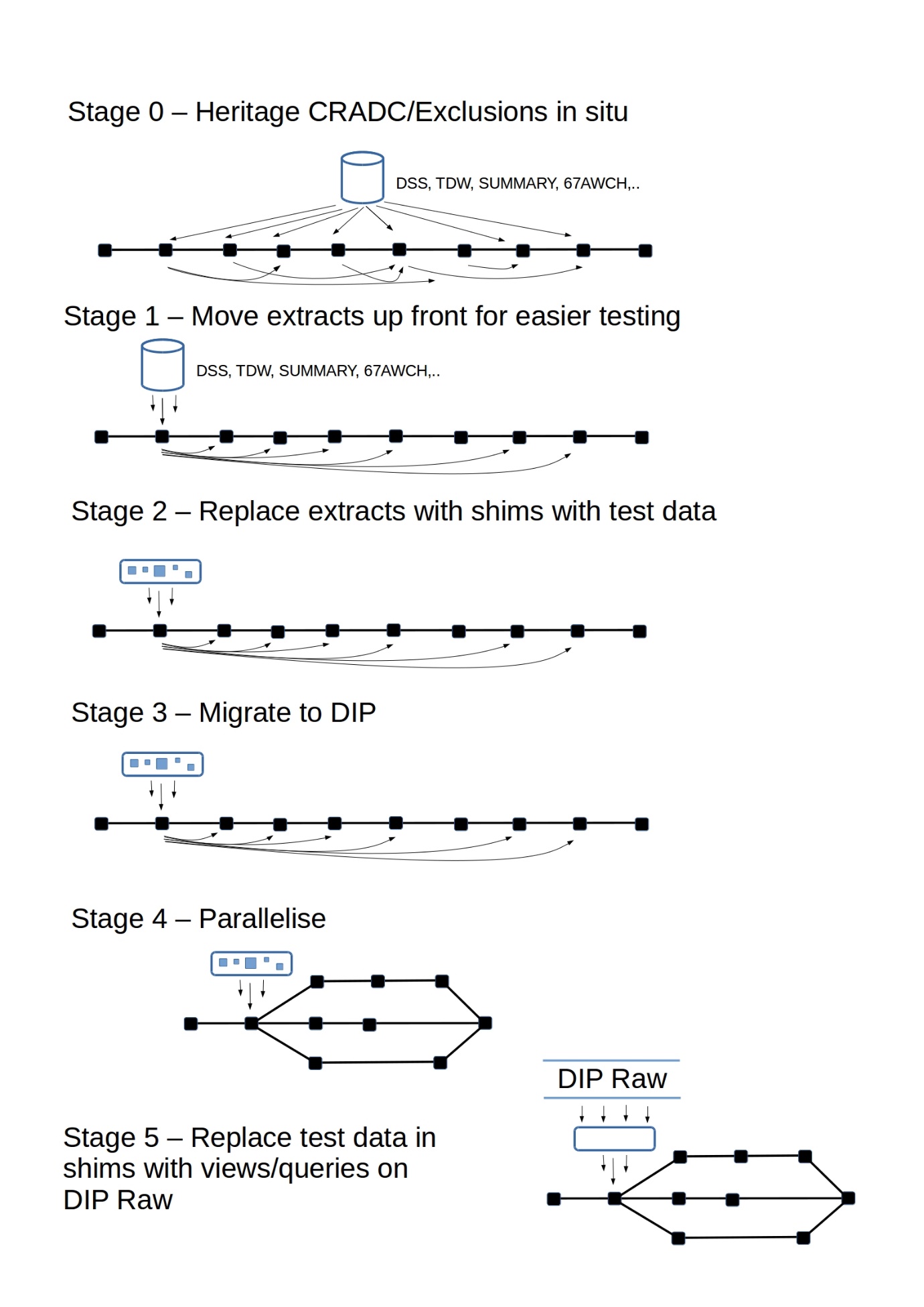
## Possible Security Breach in CRADC?

CRADC working directories contains a dataset “special\_custs” from START readable by everyone.



Same for the EDW equivalent “special\_clients”

Proposed Approach



Migration Plan  
  
Goal: Moving CRADC/Exclusions processing to DIP Lab/Heritage environment in order to

* Feed the new Debt&Returns dashboard c/o Megan Sinclair
* Help the analytics team on their migration to the DIP, esp by reducing processing time
* Perhaps also bring Data-science into the fold with CAR as "feature-factory" for predictive models. (We need "as-at" functionality for this).

In the first instance we will run against EDW/TDW tables and then introduce a ‘shim’ so that START raw tables are made to look like TDW so this system can be decommissioned. Or wrap EDW/TDW ingest in as-at-date parameterised macros that can be worked on in parallel. “Making the new world look like the old one”.

### Outline of PI 3

Sprint 3.1 – Preparation – Analysis & Experiment Mon 24th June – Fri 5th July

Sprint 3.2 – Migrate to heritage Dev, with test data. List of tables. Mon 8th July – Fri 19th July

Sprint 3.3 – Final refactoring on heritage and final list of tables. Mon 22nd July – Fri 2nd Aug

Sprint 3.4 – Migrate to DIP. Data Integration Studio, scheduling Mon 5th Aug – Fri 16th Aug

Sprint 3.5 – Contingency Mon 19th Aug – Fri 30th Aug

Sprint 3.6 – Wrap-up, Prezzos, EDC Mon 2nd Sept – Fri 13th Sept

### Sprint 3.1 – Preparation – Analysis & Experiment

|  |  |
| --- | --- |
| Status | Task |
| Done | Talk to Richard, Ryan, Nisha etc about what we are doing. No surprises. Tester? |
| Done | Pick a fork-date away from changes and major development. Emphasise the need for frequent backups after fork so changes can be merged back later |
|  | Collate all docs for EDC |
| Done | Analysis of structure |
| Done | Preliminary list of tables needed |
| Done | Request Training for team: DIS, Viya programming etc |
| Done | Data from prod for testing. |

### Sprint 3.2 – Migrate to development area on heritage

|  |  |
| --- | --- |
| Status | Task |
| Done | Snapshot heritage EG project(s) |
| Done | Extract code out of EG project, minimal changes. Insert into new EG project |
| Done | Create data for testing |
| Done | First list of tables to platform team |
| Done | Essential refactoring: Autoexec with Environments. Comments Strip out logging, bespoke error trapping, separating autocall macros from programs etc. |
| Done | Start conversation: SAS9 Environment in Lab. Work being done by Platform team. |
| Done | Run#0 for timing on test data |
| Done | Proc compare testing |

### Sprint 3.3 – Final refactoring on heritage and final list of tables

|  |  |
| --- | --- |
| Status | Task |
| Done | Revised list of tables to platform team |
|  | Shim layer? |
| Done | Merge recent changes from heritage prod |
| Done | Refactor for general cohort subset |
|  | Refactor for compliance measures??? |
| Done | Run #1 for timing on test data. |
| Done | Proc compare testing |
|  | Code into version control |
|  | Training for team: version control |

### Sprint 3.4 – Migrate to DIP. Data Integration Studio & Scheduling

|  |  |
| --- | --- |
| Status | Task |
|  | Transfer code to Jumphost and/or clone from Repository? |
|  | Transfer test data to DIP |
|  | Set up environment |
|  | Get it running in SAS Studio, with minimal changes |
|  | Careful proc compare with heritage output. |
|  | Run #1 First Lab run for timing etc |
|  | Proc compare testing. |
|  | Run in DIS |
|  | Documentation |

### Other

|  |  |
| --- | --- |
| ~~Status~~ | ~~Task~~ |
|  | ~~Merge any changes from heritage~~ |
|  | ~~Backfilling – First for weekly~~ |
|  | ~~Backfilling – Monthly for comparison with previous years~~ |
|  | ~~Think about running against START tables in Raw ie moving from EDW/TDW.~~ |
|  | ~~Run #3 etc Lab run for timing etc~~ |
|  |  |
|  | ~~EDC Enterprise Data Catalog – Metadata? Documentation? Data Lineage?~~ |
|  | ~~Connect to Dashboard~~ |
|  | ~~Feedback from the business Tracy B etc~~ |
|  |  |
|  | ~~Refactor for As-at date logic and Backfill (partitioned hive HDFS or SAS?)~~ |
|  | ~~Refactor for Parallel processing: Debt, Returns, Cases should be easy~~ |

# Test Results

Timing on test data – 485,365 obs in initial client-list and 73,992 obs in CRADC\_SVOC

|  |  |  |  |
| --- | --- | --- | --- |
| Run | Time start 000\_cradc\_excl\_autoexec.sas | Time end  013\_cradc\_test\_compare.sas | Elapsed |
| Run#1 – Heritage Dev | 12:16:09 | 12:32:11 | 16:02 |
| Run#2 – Dip LAB |  |  |  |
| Run#3 – DIP Lab Parallel |  |  |  |

# Appendix - Migration Strategies

There are several approaches to migrating software from heritage/legacy systems to cloud[[6]](#footnote-6)

* Rehost “Lift & Shift” NOT POSSIBLE WITH INSTALLED PLATFORM[[7]](#footnote-7)
* Replatform “Lift, Tinker & Shift” VIABLE OPTION FOR SHORT/MEDIUM TERM[[8]](#footnote-8)
* Refactor/Rearchitect NEEDS A LOT OF ANALYSIS
* Rebuild from scratch: FOUNDATIONS IN PROGRESS WITH KYC/RLT
* Replace/Repurchase: POSSIBLE FOR CAMPAIGNS COMPONENT
* Retain “Do nothing for now” NOT A LONG-TERM OPTION[[9]](#footnote-9)
* Retire SOME APPS CAN BE RETIRED WITH NO LOSS

At present the “Rebuild from scratch” is the only approach receiving attention, but it is a time-consuming, expensive and relatively high-risk approach, especially given the underlying data-model is changing too and the “silo-ed” way it is being developed

KYC/RLT  
SM  
BAs  
SMEs

R3A  
SM  
BAs  
SMEs  
DS

Property  
SM  
BAs  
SMEs  
DS?

D&R  
SM  
BAs  
SMEs  
DS

GST  
SM  
BAs  
SMEs  
DSs

Platform  
Data Engineers

Is there an alternative? Yes, one that is more focussed on value-delivery. In Agile-speak we need to institute “Chapters” across the delivery teams, especially data engineers and data scientists whose job is to get things built properly & in a reusable way and to transfer knowledge into the delivery teams. This way offers a much higher chance of sustainable delivery.

R3A  
SM  
BAs  
SMEs  
DS  
DEs

Property  
SM  
BAs  
SMEs  
DS  
DEs

D&R  
SM  
BAs  
SMEs  
DS  
DEs

GST  
SM  
BAs  
SMEs  
DSs  
DEs

Chapter of Data Engineers   
& Data Scientists

1. Child support, Returns and Debt Collection – Single View of Customer – Campaign Exclusions [↑](#footnote-ref-1)
2. 6 hours most days, 12 hours on 13th June 2019 – Richard Holley, email to Analytics team. 21 hours on Thursday 4th July. [↑](#footnote-ref-2)
3. Is this true? Could we not just kill off some other processes? [↑](#footnote-ref-3)
4. Eg by comparison with the way things are done at Ministry of Social Development, or the Ministry of Business, Innovation and Employment. [↑](#footnote-ref-4)
5. Yes, this is the spelling [↑](#footnote-ref-5)
6. <https://aws.amazon.com/blogs/enterprise-strategy/6-strategies-for-migrating-applications-to-the-cloud/> [↑](#footnote-ref-6)
7. Heritage is a mix of processing using SAS 9 and Oracle RDBMS, whereas the new platform as delivered is SAS Viya with Hive/Imapala on HDFS. [↑](#footnote-ref-7)
8. Extend DIP to include SAS 9 area so legacy SAS code will run. Rewrite ORACLE SQL to run in SAS9 or Hive/Impala [↑](#footnote-ref-8)
9. IR wants to turns off EDW/TDW to save on Oracle licences and exit Unisys premises, and in any case the heritage platform is on its last legs. [↑](#footnote-ref-9)