**CRoM Agile ETL and Metadata Process Analysis  
  
Version History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version #** | **Revision Date** | **Revised By** | **Change Descriptions** |
| V1.00 | 3/13/2017 | Yaija, Jyothi, Konasani, Avinash | First draft created. |
| V2.00 | 3/30/2017 | Puskas, Louis | Reformat Contents + Enrichments |
|  |  |  |  |

**Version History – Post Approval**

Changes approved through the change control process.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version #** | **CC#** | **Revision Date** | **Revised By** | **Change Descriptions** |
| v1.01 |  |  |  |  |
|  |  |  |  |  |

**Approvals *[Required]***

Documented approvals are **required** from all approvers designated below or within PMUWS’s Documents & Approvals.

Check if captured in PMUWS’ Documents & Approvals [delete the following table]

|  |  |  |
| --- | --- | --- |
| **Name** | **Project Role** | **Approval Documentation** |
|  | **Project Sponsor** *[Required]* | [Insert email approvals as embedded objects] |
|  |  |  |

**Reviewers**

The listed individuals had the opportunity to review this document and provide input prior to submission for approval.

|  |  |
| --- | --- |
| **Name** | **Project Role** |
| Shellie Kaiser | Technical Manager |
|  |  |

Table of Contents

[Introduction 3](#_Toc478651417)

[CRoM Agile Project Nomenclature 3](#_Toc478651418)

[Branch 3](#_Toc478651419)

[Main branches and Child branches 4](#_Toc478651420)

[Lanes 4](#_Toc478651421)

[CRoM Agile Code Release Lifecycle 5](#_Toc478651422)

[Development 5](#_Toc478651423)

[SIT 5](#_Toc478651424)

[UAT 5](#_Toc478651425)

[Advantages of the current process 6](#_Toc478651426)

[Limitations of the current process 6](#_Toc478651427)

[Noted solution suggestions for current limitations 6](#_Toc478651428)

[CRoM Current Migration and Retrofit Process 8](#_Toc478651429)

[CRoM and EMM Lane Specific Metadata Imports and Testing Approach 9](#_Toc478651430)

[Metadata import process for Lane3 9](#_Toc478651431)

[Advantages of the current process 10](#_Toc478651432)

[Limitations of the current process 10](#_Toc478651433)

[Noted solution suggestions for current limitations 10](#_Toc478651434)

[CRoM and EMM business term to physical metadata linkage 11](#_Toc478651435)

[Objective 11](#_Toc478651436)

[Proposed Process 11](#_Toc478651437)

[Metadata Stewards Sprint activities 11](#_Toc478651438)

[Current process 11](#_Toc478651439)

[Challenges with the proposed process 12](#_Toc478651440)

[Other notable challenges 12](#_Toc478651441)

[Noted solution suggestions for current limitations and proposed solutions 12](#_Toc478651442)

# 

# Introduction

Correspondent Reporting on MIDE, otherwise known as CroM, is a new data mart being built with a semantic layer present for reporting purposes. The new solution is set to replace correspondent reports built on CODA. The project leverages an agile software development methodology to release new ETL code written in the Ab Initio software development platform. Each sprint typically lasts for 3 weeks. Multiple Sprint releases in UAT are pooled together for a PROD release.

As of the time this document was written, they are the first and only team within EDA/MIDE to merge an agile based project approach with an emphasis on quality checks for the quantitative data produced as well as the qualitative. Change management for both types of data begins in the system integration testing (SIT) phase, which is also a first for ETL projects under the EDA/MIDE umbrella. New project management, ETL code development, and quality assurance testing approaches were used to achieve this.  
  
This document explains in detail many of those steps. Some of topics covered are: the agile ETL code migration process which uses EME lane concepts required to achieve the parallel development of code required for sprints, a new process of retrofits used to update parallel coding lanes with changes across lanes, and updates to our current ETL metadata import process managed by our enterprise management team (EMM).  
  
Also required to support the agile project approach, new approaches used to link business metadata to ETL code and physical metadata in lower and upper environments were derived by CroM with support from EMM.   
  
Throughout the document, we attempt to present the challenges of using the new approaches while highlighting their advantages. Possible solutions to overcome the challenges with new approaches are also discussed.

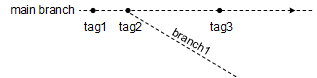
# CRoM Agile Project Nomenclature

## **Branch**

A branch is similar to a copy of a technical repository, with the advantage that unlike a real copy, a branch uses very little extra disk space. Changes made to a parallel development branch do not impact the original parent branch from which the parallel development branch was derived.

## **Main branches and Child branches**

In the Ab Initio EME technical repository used by the CroM project, the branch which most closely reflects their latest production release is called the main branch.   
  
A child branch is identical to its parent until new code edits are checked into the child branch. The following figure demonstrates how child branch1 is created from a snapshot of the main branch by using a snapshot of the main branch otherwise known as a tag. In this case branch1 was created from tag2.

  
  
A child branch has access to all tags that were created on its parent as well as to any tags created on it after it’s replication from the main branch.   
  
Since the CroM project is using the Ab Initio software suite, system commands like `air` (ab initio repository) are available to display information about branches and tags. For example in the figure above, running `air tag list` against branch1, the child branch, will return as list of tag1 and tag2 but not tag3. In the child branch tag3 was never created.

## **Lanes**

A concept of “Lanes” is used within the CroM team to quickly refer to different branches used during ETL code development. They currently have four lanes. In the Ab Initio source code repository (EME) the lanes are called: mtgetl1, mtget2, mtgetl3, mtgetl4. Each lane is used in each phase of their agile development lifecycle. The table below was added to help visualize this. Lane1 always carries their production ready code release. The remaining three lanes are used to code changes for the different subject areas shown in the table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Type of code | Dev | SIT | UAT | PROD |
| main | Lane 1 | lane 1 | Lane 1 | Lane 1 |
| services | Lane 2 | Lane2 | Lane2 |  |
| sourcing/Integrated | Lane 3 | Lane 3 | Lane 3 |  |
| other release | Lane 4 | lane 4 | lane 4 |  |

# CRoM Agile Code Release Lifecycle

## Development

Development of solutions to new stories and to bug fixes happens in the development phase of a Lane.

If the development phase of a Lane didn’t already exist, it would be created with a backwards copy of the production phase of Lane 1. It is assumed that through a process of change management code changes written and promoted alongside unrelated development lanes will eventually be backwards copied from production Lane 1 into the development phase of the lanes where the those code updates were not originally written.

In order to migrate their changes, developers create patch tags inside the Ab Initio EME for each object in the ETL code changed to meet the needs of their current sprint cycle. Next, the Ab Initio tags are recreated in third party software outside of Ab Initio called Anthil using a batch Id from the Anthil namespace. For assembly testing purposes, newly developed objects promoted into Anthil are checked out into an assembly testing sandbox owned by their lane’s process account. When all ETL artifacts are demonstrated to work together in full ETL executions using the process account, the Anthil packages are deemed ready for the SIT phase of that respective Lane.   
  
The following table is a pseudo example of a tracking exampled used to mitigate the retrofit process.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Project | Changed Artifact | Tag name | latest version | comments |
| project | a.mp | tag 1 | discard | discard a.mp because tag3 consists of latest code of a.mp |
| project | b.mp | tag 2 | b.mp | Tag2 have latest code of b.mp |
| project | a.mp | tag 3 | a.mp | Tag3 have latest code of a.mp |

## SIT

As discussed earlier in the explanation of Lanes, each Lane used by the CRoM team passes through the SIT phase of the Ab Initio ETL software suite. The SIT phase is unique in the fact that Ab Initio development tools are never pointed here. Each piece of code in SIT arrived via a code promotion process using Anthil. Further regression testing is completed in the SIT phase Ab Initio sandboxes.

The testing to some degree is more complete than what was done during the assembly testing phase of the code in the development phase. The SIT phase of each Lane is where impacts to the metadata which describes the Ab Initio code and its related artifacts like source databases and dataset lookups are validated for the first time using the Ab Initio Metadata Hub thin client.

## UAT

Similar to the SIT environment, the UAT environment is populated strictly via a code promotion process using Anthil. The current CRoM UAT environment is only being used to get user acceptance sign off for Lane 3 quantitative data output and the qualitative data which describes it.

## Advantages of the current process

Development on parallel branches is ongoing for multiple releases by multiple groups of development teams at the same time.

Assembling testing of all code pieces working together is done at each phase of each Lane as a regression test of the developer’s tags and the third party code migration software Anthil.

Hi priority production break fixes can be completed and tested quickly without interrupting new code development.

Backward compatibility tests can be performed for new releases in all phases of a Lane.

## Limitations of the current process

Retrofitting is a time consuming manual process which requires a lot of effective communication between the release leads and project managers.

The result of incorrect retrofitting into the lower phases of a development lane could equate to serious code overwrites causing an exponential amount of code conflicts in that phase of the lane.

Each code retrofit application across lanes requires a test in the lane it was added too which results in large test cycle scope. If code retrofits are not properly tested in the lower phase of a lane, it may increase the amount of test cases will be added in the upper env phases of a lane.

At the moment, no online collaboration software is used to historically document the required retrofits across lanes to maintain succinctness. Teams are dependent on emails which cannot be tracked effectively.

Creating a full tag of an Ab Initio EME project is very difficult if not possible. Meaning you are always in a constant state of managing various patch tags without a true snapshot of your entire Ab Initio EME project at any given point in time.

## Noted solution suggestions for current limitations

Leverage code freeze times in your production release schedule to consolidate your patch tags into one fully complete Ab Initio EME project and then create a single tag for all objects that are involved for the particular code promotion.  
  
If no online collaboration tool like Jira is available to track fixes for different efforts, considering using a traditional thick client software solution like excel to track and collaborate. Store the excel document on a SharePoint available to the entire team. For reach Lane create a separate tab in the excel document.   
  
Per branch tab in this excel record the details of the code base used to create the branch, the names of the patch tags create and the objects contained within those Ab Initio EME tags. Make sure for the tags to include metadata about when the tags were created, by whom, and for what problem they were originally intended to address.   
  
Most importantly, keep track of when each patch tag has been retrofit applied to the development lanes which do not already have those code updates. Use this knowledge of which retrofit fixes have been applied and have not applied to create a difference report, or outstanding required patch applications report.

When advancing patch tags to high level phases for each lane, leverage your branch diff reports. If they were updated before retrofitting the changes and after retrofitting the changes they could be used in a code promotion approval and audit process.   
QA team should take few critical functionalities of previous release branch from which retrofit changes are taken/code base is taken and include few test cases to validate the functionality.

Document the entire process of hand shake between the teams to handle and explain and document the importance of code retrofit document to all developers and educate all the teams on the process of handling the retrofits .This avoids major issue of source code control.

# CRoM Current Migration and Retrofit Process



# CRoM and EMM Lane Specific Metadata Imports and Testing Approach

To facilitate the testing of Metadata changes resulting from ETL code changes released in parallel lanes, the following prototype process was implemented in the SIT EMH (Enterprise Metadata Hub) by our EMM (Enterprise Metadata Management) team.   
  
The end goal eventually will be to support lower environment testing of metadata changes across all development lanes. This would mean retrofit changes implemented across all lanes for ETL code completeness could have their metadata impact tested in the SIT EMH before the ETL code was promoted to UAT.  
  
Currently metadata changes resulting from planned ETL code changes in Lane 3, and those implicit metadata changes resulting from retrofit changes from Lanes 1, 2, and 4 being applied to Lane 3 can be tested in the EMH.

## Metadata import process for Lane3

Prerequisite validation is completed by the EMM team. They check if source feeds known to support Lane3 metadata have completed successfully in the SIT EMH. An example of a source feed known to support Lane 3 would be the feed importing EIWS Teradata assets.  
  
ETL developers notify the EMM team during regularly scheduled touch base meetings that their changes have been migrated to the SIT phase of the Ab Initio EME (source repository). When notified, the EMM team executes metadata hub import jobs which extract Lane 3 code artifacts from the SIT phase of the Ab Initio EME into the SIT phase of the Ab Initio Metadata Hub.   
  
There are approximately 30 import feeds which can affect assets related to CRoM sprint changes. For a complete refresh of Lane 3 metadata prior to an upcoming release (or CRoM sprint), these feeds should be run in a predetermined order.

There is an AutoSys Job specifically created to support running the Lane 3 metadata imports. The AutoSys job will run through a list of Lane 3 import feeds one-at-a-time, and it is specially configured so when one import feed completes, the changeset will be automatically approved and committed to the SIT metadata hub. This cycle will continue unattended until all import feeds in the list complete.   
  
EMM can review changesets from the run to determine if any unusual errors are seen. If they encounter abnormalities during the import like missing objects those issues are extracted into an excel report which is sent to the developers to address. An attempt to resolve those issues can be achieved by the developers promoting their fixes to the SIT phase of the AB Initio EME.

After code correction(s) are in place, it is possible to run the Lane 3 AutoSys job with an abbreviated list of import feeds so that only those objects that have potentially been changed by the correction will be re-imported. This cycle can continue until a satisfactory result is obtained.

## Advantages of the current process

Graphical lineages for current production code and also for lane3 codes can be validated in parallel.  
  
Generally, once an import feeds has been established and run in the SIT EMH, EMM seldom encounters problems with the feed.

EMH feeds related to CRoM will run cleaner in upper phase releases of the Ab Initio Metadata Hub. This will be the result of the feed import abnormalities being addressed in the SIT EMH.

UAT and Production instances of the MDH will have less lineage issues when displaying graphical lineage because of the cleaner running feeds.   
  
Business metadata to physical metadata linkage spreadsheets can be test run in SIT before they are executed in later phases of the EMH.

## Limitations of the current process

Ab Initio developers are required to have training in a specialized skill set of understanding the metadata hub import feed errors which were sent to the by the EMM team.   
  
As well as receiving training for understanding the changeset errors from the metadata hub imports, the developers would need to be trained on how to fix their code to eliminate those errors.   
  
The current EMH infrastructure is not robust enough to handle the metadata validation of all CRoM lanes of development. At the moment there is a SLA and document system in place for the support of Lane 3 only.

## Noted solution suggestions for current limitations

CRoM sponsors and managers should arrange knowledge transfer sessions targeted to their ETL development team to increase their knowledge on metadata hub lineage issues and known solutions.

# CRoM and EMM business term to physical metadata linkage

## Objective

Incorporate business term linkage activities in lower environments as part of CROM Sprint cycles when new business term to physical metadata linkage relationships are introduced by their business sponsors.

## Proposed Process

* Create business terms in the production EMH using import sheets or the MDH Portal.
* Backport production EMH business terms into the SIT EMH.
* Import physical metadata into the SIT EMH (Lane 3) from SIT MIDE/CROM Databases.
* Create a SIT topology linkeage spreadsheet in excel. Send to EMM to import via EMH feed into SIT EMH.
* Backport production EMH business terms into the UAT EMH.
* Import physical metadata into the UAT EMH (Lane 3) from SIT MIDE/CROM Databases.
* Reuse working SIT linkeage spreadsheet in excel. Change physical values to match UAT topology. Send to EMM to import via EMH feed into UAT EMH.
* Reuse working UAT linkeage spreadsheet in excel. Change physical values to match PROD topology. Send to EMM to import via EMH feed into PROD EMH.

## Metadata Stewards Sprint activities

Metadata Stewards are responsible for Business term linkage in CROM project.

Following tasks are performed by the Metadata stewards during the Sprint:

* Work with project team on daily basis to review CROM business elements.
* Help determine Name and Definition so that they meet Data Program Standards.
* Help with the creation of Business Metadata stories.
* Implement Business Metadata terms into Ab Initio production.
* Obtain approvals from the owning Data Steward. (Mostly, it's Matt Barrett)
* Resolve any errors resulting from testing

## Current process

* Create business terms in the production EMH using import sheets or the MDH Portal.
* Create a PROD topology linkeage spreadsheet in excel. Send to EMM to import via EMH feed into PROD EMH.

## Challenges with the proposed process

1. All the environments technical and business metadata is not in sync.
2. Linkage spreadsheets developed in SIT need to have the physical metadata elements updated to match the topology of the upper environments before being used again in upper env EMH imports.
3. Since no exclusive mapping documents are available. Metadata stewards have to rely on stories for the sprint.
4. Business terms and linkage work backlog needs to be brought current in PROD before copying to the lower environment.
5. The business has requested business terms created and linked for all the data elements sourced from each SOR’s, used in the target CROM views. This volume of this request has been understated and is quite large.

## Other notable challenges

1. No Mapping documents available per sprint basis. Data stewards have to mostly rely on the information shared in stories and their self-analysis.
2. CROM Business term Linkage is complete till R2.16 release. Quite a bit of backlog exists to bring it to the current.
3. Business term linkage is only done for Semantic views.
4. Project work is currently being placed on hold for Safety & Soundness.
5. Maintaining CROM Master Spreadsheet. It is used for multiple purposes.  However, it does not have all the information it needs to track for all activities.

## Noted solution suggestions for current limitations and proposed solutions

1. Business term & linkage stories should be part of sprint.
2. Master data worksheet maintained by sprint teams can be leveraged for business term linkage or further enrichments
3. Linkage spreadsheet needs to be maintained and used for importing into Metadata Hub across all the environments.