DevOps for ReDi Services

ReD Shield Engineering

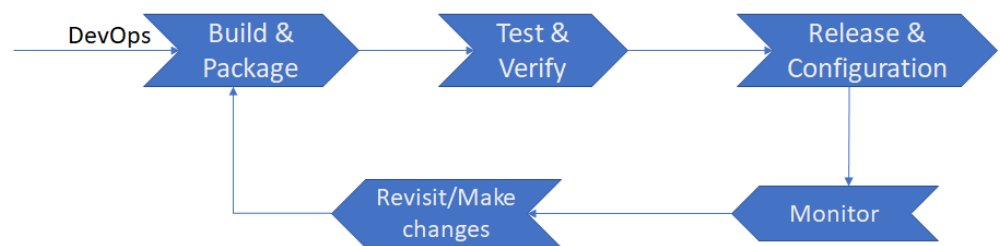Exported on 06/11/2020

Table of Contents

Purpose

This document describes all about DevOps works for ReDi Services.
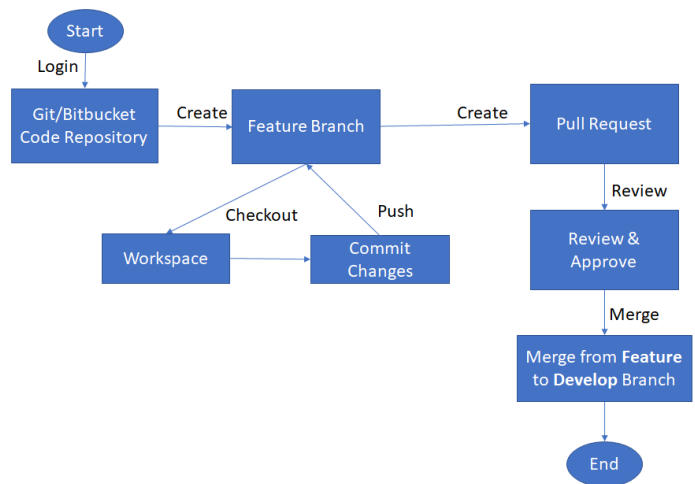
DevOps

DevOps is a set of practices that helps development & operations teams to deliver apllication/service(s) faster and reliable. By adopting a DevOps culture along with DevOps practices and tools, teams gain the ability to better respond to customer needs, increase confidence in the applications they build, and achieve business goals faster.

Devops enables Continuous integration and continuous delivery (CI/CD) of software application/service(s). Here are the DevOps tools used in ReDi Services:
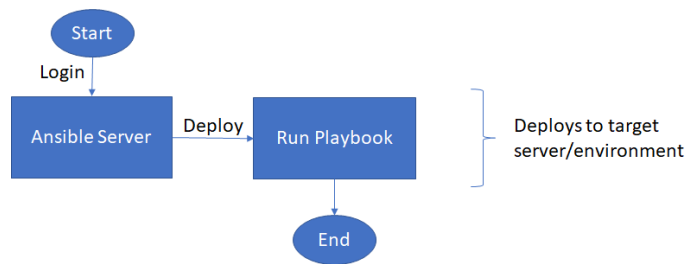
- GIT
- Jenkins
- Nexus
- Ansible

1 Developer Git Workflow:

```mermaid
flowchart TD
    Start((Start)) -->|Login| Repo[Git/Bitbucket Code Repository]
    Repo -->|Create| Feature[Feature Branch]
    Feature -->|Create| PR[Pull Request]
    Feature -->|Checkout| Workspace[Workspace]
    Workspace --> Commit[Commit Changes]
    Commit -->|Push| Feature
    PR -->|Review| Review[Review & Approve]
    Review -->|Merge| Merge[Merge from Feature to Develop Branch]
    Merge --> End((End))
```

2 Ansible Deployment Workflow:

GIT Repository Rationalization

Currently, we have a single GIT Repository for all ReDi Services[1]. As part of DevOps Best Practice, we have created an individual GIT Repository for each service (under https://bitbucket.am.tsacorp.com/projects/TIM) and migrated all source code from an existing repository.

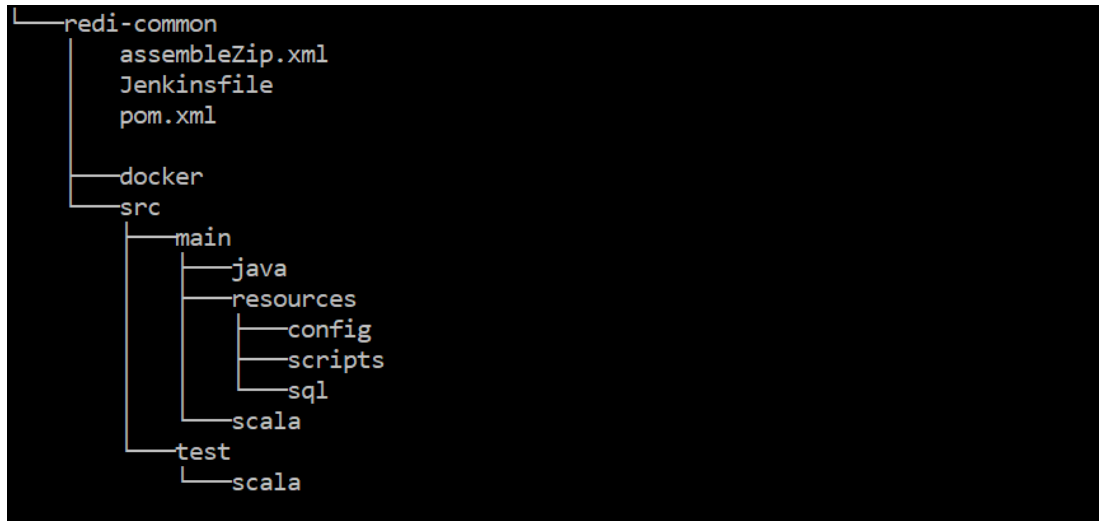| No.	Service Name	GIT Repository
1	redi-streaming	https://bitbucket.am.tsacorp.com/projects/TIM/repos/redi-stream-transformations/browse
2	redi-common	https://bitbucket.am.tsacorp.com/projects/TIM/repos/redi-common/browse
3	redi-csi	https://bitbucket.am.tsacorp.com/projects/TIM/repos/redi-csi/browse
4	redi-chargebacks	https://bitbucket.am.tsacorp.com/projects/TIM/repos/redi-chargebacks/browse
5	redi-tre-features	https://bitbucket.am.tsacorp.com/projects/TIM/repos/redi-tre-features/browse
6	redi-tre-rulehits	https://bitbucket.am.tsacorp.com/projects/TIM/repos/redi-tre-rulehits/browse
7	redi-tre-ruledetails	https://bitbucket.am.tsacorp.com/projects/TIM/repos/redi-tre-ruledetails/browse
8	redi-ibi-db	https://bitbucket.am.tsacorp.com/projects/TIM/repos/redi-ibi-db/browse

We have adopted below standards for the folder/file(s) structure of each service:

- All shell scripts should be placed in "src/main/resources/scripts"
- All properties including log4j (if any) should be placed in "src/main/resources/config"
- All Scala sources should be placed under "src/main/scala"

---

1 https://bitbucket.am.tsacorp.com/projects/TIM/repos/redi/browse

- All Java source should be placed under "src/main/java"
- All test code/scripts should be placed under "src/test"
- "docker" folder should be added with an empty pom for non-docker build
- Maven POM, Jenkinsfile and tar/zip assemble files are to be placed in root
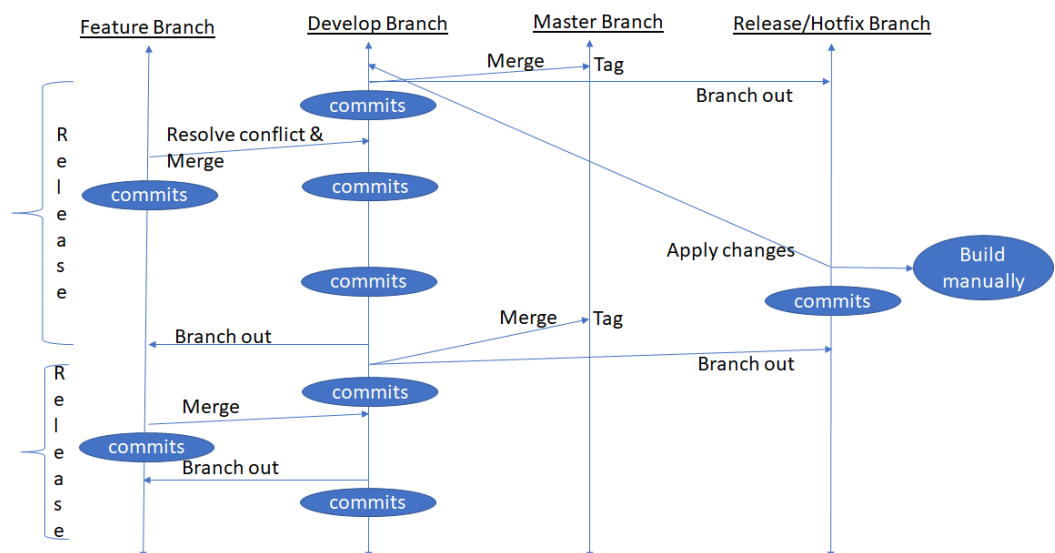
For example,

```
  └─redi-common
        assembleZip.xml
        Jenkinsfile
        pom.xml

    ├─docker
    ├─src
    │    ├─main
    │    │    ├─java
    │    │    ├─resources
    │    │    │    ├─config
    │    │    │    ├─scripts
    │    │    │    └─sql
    │    │    └─scala
    │    └─test
    │          └─scala
```

Note that there is NO SQL folder since all database objects/queries should go to "**redi-ibi-db**" repository.

Branchology

We are following below branching strategy:

- **Develop** to **Master** branch merge will be performed by Jenkins during the build from **Master** branch (No need of manual merge)
- All new PROD release artifact will be built from **Master** branch only
- SNAPSHOT build is applicable only for **Develop** branch
- SNAPSHOT artifact is used for lower environment such as Dev, CIT and UAT
- Any code changes to **Release/Hotfix** branch should be applied to Develop branch as well. So that next new release will have those changes

Jenkins Micro-Service Pipeline

We are using standard Micro-Service Pipeline framework (https://nrc2jenkins05vm.am.tsacorp.com/job/PRM/job/Titanium/job/Micro-Service_Pipelines/) created by Cobalt team.There are 3 types of Jenkins Pipelines:

No.	Pipeline Name	Pipeline URL	Purpose
1	Build-pipelines	https://nrc2jenkins05vm.am.tsacorp.com/job/PRM/job/Titanium/job/Micro-Service_Pipelines/job/Build-pipelines/	Used by Developers to build from **Develop** & **Feature** branches
2	Merge to Master (Code Lock)	https://nrc2jenkins05vm.am.tsacorp.com/job/PRM/job/Titanium/job/Micro-Service_Pipelines/job/Merge-Master/	Used by RM & PO to build from **Master** branch (merge from **Develop** to **Master** branch happens here)
3	Release(GA) Micro-Service for UAT and PROD	https://nrc2jenkins05vm.am.tsacorp.com/job/PRM/job/Titanium/job/Micro-Service_Pipelines/job/Release-UAT/	**NOT USED for ReDi**

Screen Shot:

We have an individual pipeline for each ReDi service under **Build-pipelines** & **Merge to Master (Code Lock).**

No.	Service Name	Build-pipelines	Merge to Master (Code Lock)
1	redi-streaming	https:// nrc2jenkins05vm.am.tsacorp.com/job/ PRM/job/Titanium/job/Micro-Service_Pipelines/job/Build-pipelines/ job/ReDiStreaming_Pipeline/	https:// nrc2jenkins05vm.am.tsacorp.com/job/ PRM/job/Titanium/job/Micro-Service_Pipelines/job/Merge-Master/ job/ReDiStreaming_Release_Prepare/
2	redi-common	https:// nrc2jenkins05vm.am.tsacorp.com/job/ PRM/job/Titanium/job/Micro-Service_Pipelines/job/Build-pipelines/ job/ReDiCommon_Pipeline/	https:// nrc2jenkins05vm.am.tsacorp.com/job/ PRM/job/Titanium/job/Micro-Service_Pipelines/job/Merge-Master/ job/ReDiCommon_Release_Prepare/
3	redi-csi	https:// nrc2jenkins05vm.am.tsacorp.com/job/ PRM/job/Titanium/job/Micro-Service_Pipelines/job/Build-pipelines/ job/ReDiCSIService_Pipeline/	https:// nrc2jenkins05vm.am.tsacorp.com/job/ PRM/job/Titanium/job/Micro-Service_Pipelines/job/Merge-Master/ job/ReDiCSIService_Release_Prepare/
4	redi-chargebacks	https:// nrc2jenkins05vm.am.tsacorp.com/job/ PRM/job/Titanium/job/Micro-Service_Pipelines/job/Build-pipelines/ job/ReDiChargebacksService_Pipeline/	https:// nrc2jenkins05vm.am.tsacorp.com/job/ PRM/job/Titanium/job/Micro-Service_Pipelines/job/Merge-Master/ job/ ReDiChargebacksService_Release_Prepare/

No.	Service Name	Build-pipelines	Merge to Master (Code Lock)
5	redi-tre-features	https://nrc2jenkins05vm.am.tsacorp.com/job/PRM/job/Titanium/job/Micro-Service_Pipelines/job/Build-pipelines/job/ReDiTREFeaturesService_Pipeline/	https://nrc2jenkins05vm.am.tsacorp.com/job/PRM/job/Titanium/job/Micro-Service_Pipelines/job/Merge-Master/job/ReDiTREFeaturesService_Release_Prepare/
6	redi-tre-rulehits	https://nrc2jenkins05vm.am.tsacorp.com/job/PRM/job/Titanium/job/Micro-Service_Pipelines/job/Build-pipelines/job/ReDiTRERuleHitsService_Pipeline/	https://nrc2jenkins05vm.am.tsacorp.com/job/PRM/job/Titanium/job/Micro-Service_Pipelines/job/Merge-Master/job/ReDiTRERuleHitsService_Release_Prepare/
7	redi-tre-ruledetails	https://nrc2jenkins05vm.am.tsacorp.com/job/PRM/job/Titanium/job/Micro-Service_Pipelines/job/Build-pipelines/job/ReDiTRERuleDetailsService_Pipeline/	https://nrc2jenkins05vm.am.tsacorp.com/job/PRM/job/Titanium/job/Micro-Service_Pipelines/job/Merge-Master/job/ReDiTRERuleDetailsService_Release_Prepare/
8	redi-ibi-db	Not ready yet	Not ready yet

Screen Shots:

Sample Jenkins Output:
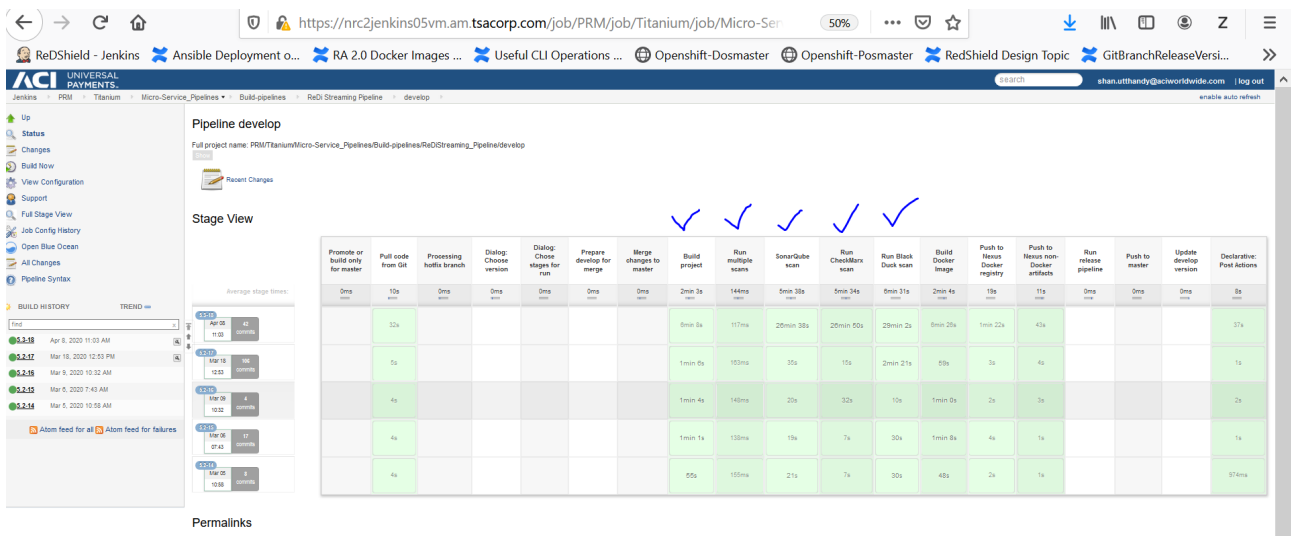
Ansible Deployment

Ansible is an open-source scripting language for software provisioning, configuration management and application-deployment. Ansible provides reliability, consistency, and scalability for automating configurations of databases, storage, networks, firewalls.

Ansible is popular due to:

- **Agentless** – Which means there is no kind of software or any agent managing the node like other solution such as puppet and chef
- **Python** – Built on top of python, which is fast and one of the robust programming languages in today's world
- **SSH** – Very simple passwordless network authentication protocol which is secure. So, your responsibility is to copy this key to the client
- **Push architecture** – Push the necessary configurations to them, clients. All you have to do is, write down those configurations (playbook) and push them all at once to the nodes. You see how powerful it can be to push the changes to thousands of servers in minutes
- **Setup** – easy installation and configuration

Ansible Architecture:
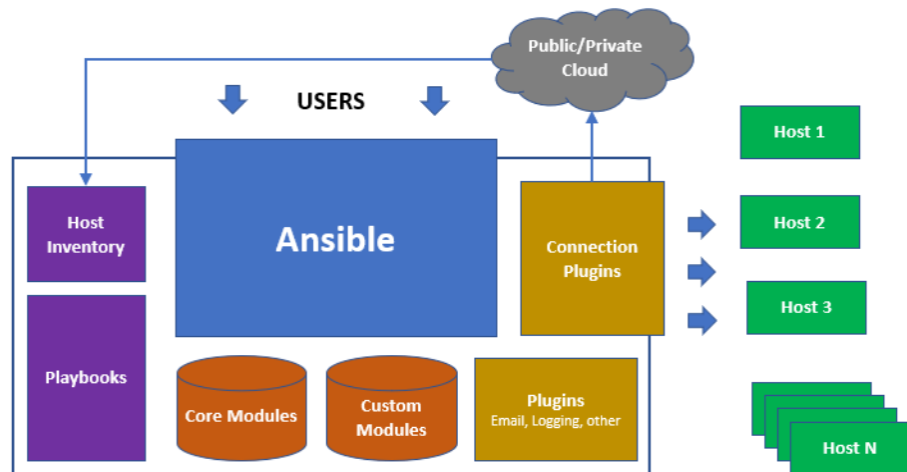
ReDi Deployment Format/Structure:

5.2 Release

```
    apps
        ReDi
            config
            lib
            v5.2
                    redi-chargebacks
                    redi-csi
                    redi-streaming
                    redi-tre-rulehits
```

5.3 Release

```
    apps
        ReDi
            config
            lib
            v5.3
                    redi-chargebacks
                    redi-csi
                    redi-streaming
```

5.3 Hotfix

```
└──apps
    └──ReDi
        ├──config
        ├──lib
        └──v5.3.1
            ├──redi-chargebacks
            ├──redi-csi
            └──redi-streaming
```

Running Playbook:

ansible-playbook playbook/install-spark-services.yml -i environment/spark-cit -t redi_streaming

We would like to perform 2 types of ReDi deployments:

- **Incremental** for minor release
- **Full** for major release

But right now, we are doing **Full** deployment and not cleaning up all unwanted releases.

Pending work

No.	Name	JIRA Link
1	Automation to get the SSO working	☑ MERF-14991[2] - Automation to get the SSO working RESOLVED
2	Ansible script changes (Miscellaneous)	🔴 MERF-22377[3] - Ansible script changes (Miscellaneous) CLOSED
3	Test and Validate Ansible deployment for all ReDI Services	🔴 MERF-23391[4] - Test and Validate Ansible deployment for all ReDI Services CLOSED

References

Please use below wiki page for migrating any Micro-Services to Standard Micro-Services Pipeline:

---

2 https://jira.aciworldwide.com/browse/MERF-14991?src=confmacro
3 https://jira.aciworldwide.com/browse/MERF-22377?src=confmacro
4 https://jira.aciworldwide.com/browse/MERF-23391?src=confmacro

- Migration steps[5]
- Build Pipeline[6]

- JIRA/GIT tickets
    - https://gojira.am.tsacorp.com/projects/SDPDT/issue/SDPDT-6147
    - https://gojira.am.tsacorp.com/projects/SDPDT/issue/SDPDT-3538[7]
    - https://gojira.am.tsacorp.com/projects/SDPDT/issue/SDPDT-4930[8]
    - https://gojira.am.tsacorp.com/projects/SDPDT/issue/SDPDT-6418
- ReDi service/job listing

No.	Jar Name	Shell Script Name	Class Name
1	ReDi_common-5.2-SNAPSHOT.jar	BEDDataProcess.sh CBReasonCodeProcess.sh ClientMasterDataProcess.sh CurrencyRatesDataProcess.sh CurrencyRatesHistoricalDataProcess.sh CurrencyRatesHistoricalDataProcessfile.sh ReDiJobScheduler.sh SDSExcludeValueProcess.sh	com.aciworldwide.ra.redi.common.actions.BedDataProcess com.aciworldwide.ra.redi.common.actions.CBReasonCodeProcess com.aciworldwide.ra.redi.common.actions.ClientMasterDataProcess com.aciworldwide.ra.redi.common.actions.CurrencyRatesDataProcess com.aciworldwide.ra.redi.common.actions.CurrencyRatesHistoricalDataProcess com.aciworldwide.ra.redi.common.actions.CurrencyRatesHistoricalDataProcessFile com.aciworldwide.ra.redi.common.actions.ReDiJobScheduler com.aciworldwide.ra.redi.common.actions.SDSExcludeValueProcess
2	ReDi_Chargebacks-5.2-SNAPSHOT.jar	AutomatedCBDataProcess.sh BedCBDataProcess.sh	com.aciworldwide.ra.redi.chargebacks.actions.AutomatedCBDataProcess com.aciworldwide.ra.redi.chargebacks.actions.BedCBDataProcess
3	ReDi_CSI_Dataflow-5.2-SNAPSHOT.jar	CSIDataProcess.sh	com.aciworldwide.ra.redi.csi.actions.CSIDataProcess
4	TRE_Features-5.2-SNAPSHOT.jar	TREFeaturesKafkaConsumer.sh	com.aciworldwide.ra.redi.tre.features.actions.TREFeaturesKafkaConsumer

---

5 https://wiki.aciworldwide.com/display/ReD/Migration+steps
6 https://wiki.aciworldwide.com/display/ReD/Risk+Analytics+Build+Pipeline
7 https://gojira.am.tsacorp.com/projects/SDPDT/queues/issue/SDPDT-3538
8 https://gojira.am.tsacorp.com/projects/SDPDT/queues/issue/SDPDT-3538

No.	Jar Name	Shell Script Name	Class Name
5	TRE_Rule_Details-5.2-SNAPSHOT.jar	DveTswIngestionAndTransAction.sh	com.aciworldwide.ra.redi.tre.ruledetails.actions.DveTswIngestionAndTransAction
		TRERuleDetailIngestionProcess.sh	com.aciworldwide.ra.redi.tre.ruledetails.actions.TRERuleDetailIngestionProcess
		VelocityLimitsMasterAction.sh	com.aciworldwide.ra.redi.tre.ruledetails.actions.VelocityLimitsMasterAction
6	TRE_Rule_Hits-5.2-SNAPSHOT.jar	TRERuleHitsProcess.sh	com.aciworldwide.ra.redi.tre.ruleHits.actions.TRERuleHitsProcess
7	RS_Trans_Dataflow-5.2-SNAPSHOT.jar	historical_failover.sh	com.aciworldwide.ra.redi.rstransflow.actions.HistoricalTransFlowFailedAction
		HistoricalIngestProcess.sh	com.aciworldwide.ra.redi.rstransflow.actions.HistoricalRSExecIngestProcess
		incremental_failover.sh	com.aciworldwide.ra.redi.rstransflow.actions.TransFlowFailedAction
		TransFlowProcessAction.sh	com.aciworldwide.ra.redi.rstransflow.actions.TransFlowProcessAction

Need to discuss about below scripts:

No.	Script Name	Purpose
1	batch-job-monitoring-AutomatedCBDataProcess.sh	
	batch-job-monitoring-BedCBDataProcess.sh	
	batch-job-monitoring-BedDataProcess.sh	
	batch-job-monitoring-CBReasonCodeProcess.sh	
	batch-job-monitoring-ClientMaster.sh	
	batch-job-monitoring-CSIDataProcess.sh	
	batch-job-monitoring-CurrencyRatesDataProcess.sh	
	batch-job-monitoring-CurrencyRatesHistoricalDataProcess.sh	
	batch-job-monitoring-DveTswIngestionAndTransAction.sh	
	batch-job-monitoring-redi-reprocess-transactions.sh	
	batch-job-monitoring-SDSExcludeValueProcess.sh	
	batch-job-monitoring-VelocityLimitsMasterAction.sh	

No.	Script Name	Purpose
2	restart_historical_failover.sh restart_incremental_failover.sh restart_ingestion_monitor.sh restart_TransFlowProcessAction_monitor.bash restart_transformation_clientgroup1_monitor.sh restart_transformation_clientgroup2_monitor.sh restart_transformation_monitor.sh restart-HistoricalIngestProcess-monitor.sh restart-ReDiJobScheduler-monitor.sh restart-redi-streaming-monitor.sh restart-start-fallout-streaming-monitor.sh restart-TREFeaturesIngestion-monitor.sh restart-TRERuleDetailIngestionProcess-monitor.sh restart-TRERuleHitsProcess-monitor.sh RestartAutomatedDP.sh RestartBedCBDataProcess.sh RestartBedDataProcess.sh RestartCBProcess.sh RestartClientMasterProcess.sh RestartCSIDataProcess.sh RestartcurrrateDP.sh RestartDveTswIngestionAndTransAction.sh RestartVelocityLimitsMasterAction.sh	
3	RSExecIngestProcess.sh TransFlowProcessAction_clientgroup1.sh TransFlowProcessAction_clientgroup2.sh	
4	relfraud_update_client.sh relfraud_update_subclient.sh	

No.	Script Name	Purpose
5	job-restart-monitoring.sh monitoring_Transformation_beeline.sh	
6	alert_failover.sh generic.sh	

- Ansible for DevOps[9]
- ansible-playbook-essentials[10]
- jenkins-the-definitive-guide[11]
- groovy_tutorial[12]

9 https://wiki.aciworldwide.com/display/AUTOD/ACI+DevSecOps+Community+of+Practice+Home?
preview=%2F220957650%2F317008132%2FAnsible+for+DevOps.pdf
10 https://wiki.aciworldwide.com/display/AUTOD/ACI+DevSecOps+Community+of+Practice+Home?
preview=%2F220957650%2F317008134%2Fansible-playbook-essentials.pdf
11 https://wiki.aciworldwide.com/display/SCM/SCM+-+Educational+Items?preview=%2F310970753%2F317017622%2Fjenkins-
the-definitive-guide.pdf
12 https://wiki.aciworldwide.com/display/AUTOD/ACI+DevSecOps+Community+of+Practice+Home?
preview=%2F220957650%2F317008170%2Fgroovy_tutorial.pdf