<?xml version="1.0" encoding="UTF-8"?>

<!-- Ingesters process tweet create events from TweetyPie and write them to a queue for Earlybird

to index. -->

<pipeline>

<property

propName="validator"

className="org.apache.commons.pipeline.validation.SimplePipelineValidator"/>

<listener

className="org.apache.commons.pipeline.listener.ObjectProcessedEventCounter"/>

<driverFactory

className="org.apache.commons.pipeline.driver.DedicatedThreadStageDriverFactory"

id="kafka">

<property

propName="queueFactory"

className="org.apache.commons.pipeline.util.BlockingQueueFactory$ArrayBlockingQueueFactory"

capacity="1000"

fair="false"/>

</driverFactory>

<!-- Read tweets from the thrift kafka queue. The reader loops forever. -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.kafka.KafkaRawRecordConsumerStage"

kafkaClusterPath=""

kafkaClientId=""

kafkaTopicName=""

kafkaConsumerGroupId=""

maxPollRecords="1"

pollTimeoutMs="1000"

partitioned="false"

deciderKey=""

driverFactoryId="kafka"/>

<!-- Deserialize the bytes into TweetData -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.TweetEventDeserializerStage"

driverFactoryId="kafka"/>

<!-- Filter to only have the safetytype for this cluster -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.FilterEventsBySafetyTypeStage"

tweetCreateLatencyLogThresholdMillis="5000"

safetyType="PROTECTED"

driverFactoryId="kafka"/>

<!-- Parse to TwitterMessage -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.ThriftTweetParserStage"

tweetDeleteEventBranchNames="kafka\_update\_events\_delete"

driverFactoryId="kafka"/>

<branch>

<pipeline key="kafka\_update\_events\_delete">

<property

propName="validator"

className="org.apache.commons.pipeline.validation.SimplePipelineValidator"/>

<listener

className="org.apache.commons.pipeline.listener.ObjectProcessedEventCounter"/>

<driverFactory

className="org.apache.commons.pipeline.driver.DedicatedThreadStageDriverFactory"

id="kafka\_update\_events\_delete">

<!-- we are willing to queue more deletes than other stages,

to make sure we don't slow down the incoming tweets -->

<property

propName="queueFactory"

className="org.apache.commons.pipeline.util.BlockingQueueFactory$ArrayBlockingQueueFactory"

capacity="1000"

fair="false"/>

</driverFactory>

<stage

className="com.twitter.search.ingester.pipeline.twitter.kafka.DeleteUpdateEventsKafkaProducerStage"

kafkaClusterPath=""

kafkaClientId=""

kafkaTopicName=""

driverFactoryId="kafka\_update\_events\_delete"/>

</pipeline>

</branch>

<!-- filters out messages that are not formatted correctly -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.FilterTwitterMessageStage"

driverFactoryId="kafka"/>

<!-- retrieves space ids from space urls if the tweet has space urls -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.RetrieveSpaceIdsStage"

driverFactoryId="kafka"/>

<!-- looks up user reputation scores for each message -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.LookupUserPropertiesBatchedStage"

driverFactoryId="kafka"/>

<!-- extract text features of the message -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.TextFeatureExtractionWorkersStage"

driverFactoryId="kafka"/>

<!-- compute text quality score of the message -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.TextQualityEvaluationWorkerStage"

driverFactoryId="kafka"/>

<!-- Extract lat/lon pairs from the text, and geocode them -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.SingleTweetExtractAndGeocodeLatLonStage"

driverFactoryId="kafka"/>

<!-- adds coded locations -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.PopulateCodedLocationsBatchedStage"

driverFactoryId="kafka"/>

<!-- Parse the TwitterMessages into ThriftStatuses -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.ConvertMessageToThriftStage"

thriftVersionedEventsBranchName="kafka\_base\_tweets"

driverFactoryId="kafka"/>

<!-- Branch for tweets -->

<branch>

<pipeline key="kafka\_base\_tweets">

<property

propName="validator"

className="org.apache.commons.pipeline.validation.SimplePipelineValidator"/>

<listener

className="org.apache.commons.pipeline.listener.ObjectProcessedEventCounter"/>

<driverFactory

className="org.apache.commons.pipeline.driver.DedicatedThreadStageDriverFactory"

id="kafka\_base\_tweets">

<property

propName="queueFactory"

className="org.apache.commons.pipeline.util.BlockingQueueFactory$ArrayBlockingQueueFactory"

capacity="1000"

fair="false"/>

</driverFactory>

<stage

className="com.twitter.search.ingester.pipeline.twitter.kafka.TweetThriftVersionedEventsKafkaProducerStage"

kafkaClusterPath=""

kafkaClientId="search\_ingester\_indexing\_events"

kafkaTopicName="search\_ingester\_indexing\_events\_protected\_prod"

driverFactoryId="kafka\_base\_tweets"/>

</pipeline>

</branch>

<!-- Resolve compressed URL via Pink -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.ResolveCompressedUrlsBatchedStage"

pinkClientId="INGESTER"

batchedStageBatchSize="10"

tweetMaxAgeToResolve="10000"

driverFactoryId="kafka"/>

<!-- Retrieve card information -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.RetrieveCardBatchedStage"

tweetypieClientId="ingester.prod"

filterProtected="false"

internalBatchSize="50"

driverFactoryId="kafka"/>

<!-- Retrieve named entities -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.RetrieveNamedEntitiesSingleTweetStage"

driverFactoryId="kafka"/>

<!-- retrieves space admins and title for a tweet if the tweet has space urls -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.RetrieveSpaceAdminsAndTitleStage"

driverFactoryId="kafka"/>

<!-- extract text features of the message -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.TextUrlsFeatureExtractionStage"

driverFactoryId="kafka"/>

<!-- Compute the tweet signature -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.ComputeTweetSignatureStage"

driverFactoryId="kafka"/>

<!-- Parse the TwitterMessages into ThriftStatuses -->

<stage

className="com.twitter.search.ingester.pipeline.twitter.ConvertDelayedMessageToThriftStage"

driverFactoryId="kafka"/>

<stage

className="com.twitter.search.ingester.pipeline.twitter.kafka.TweetThriftVersionedEventsKafkaProducerStage"

kafkaClusterPath=""

stageName="UpdateEvents"

kafkaClientId=""

kafkaTopicName=""

driverFactoryId="kafka"/>

</pipeline>