package com.twitter.tweetypie.config

import com.twitter.finagle.thrift.ClientId

import com.twitter.servo.util.FutureArrow

import com.twitter.tweetypie.\_

import com.twitter.tweetypie.service.{ClientIdSettingTweetServiceProxy, TweetServiceProxy}

/\*\*

\* This class builds deciderable ThriftTweetService and FutureArrows that respect the

\* simulateDeferredrpcCallbacks decider. When simulateDeferredrpcCallbacks=true, invocations will

\* be performed synchronously by the root ThriftTweetService.

\*/

class ServiceInvocationBuilder(

val service: ThriftTweetService,

simulateDeferredrpcCallbacks: Boolean) {

def withClientId(clientId: ClientId): ServiceInvocationBuilder =

new ServiceInvocationBuilder(

new ClientIdSettingTweetServiceProxy(clientId, service),

simulateDeferredrpcCallbacks

)

def asyncVia(asyncService: ThriftTweetService): ServiceInvocationBuilder =

new ServiceInvocationBuilder(

new TweetServiceProxy {

override def underlying: ThriftTweetService =

if (simulateDeferredrpcCallbacks) service else asyncService

},

simulateDeferredrpcCallbacks

)

def method[A, B](op: ThriftTweetService => A => Future[B]): FutureArrow[A, B] =

FutureArrow(op(service))

}