Real-Time Weather Tracking, Big Data, and Camel

Shane Kent, Ram Raju, and David Reiser
May 16, 2012

SERVING THE NATION AS A LEADER IN GLOBAL TRANSPORTATION INNOVATION SINCE 1970

RITA John A. Volpe National Transportation Systems Center
U.S. Department of Transportation

Overview of the FAA's SWIM Program

System Wide Information Management

http://www.swim.gov

Overview of the FAA's ITWS Program

Integrated Terminal Weather System

Our Role in SWIM & ITWS

- Conversion of a legacy system to a modern SOA-based system utilizing FuseSource.
- Real-time weather events distributed from sensor to external users in less than 1 second (average).
- Compressed data stream is approximately 1 Megabit per second, streaming constantly (approximately 9 Gigabytes per day).
- Data is processed and distributed by an ActiveMQ broker network.
- Very high throughput with ActiveMQ.
- All data is stored permanently.
- All data is accessible through HBase.

Integration Challenge # 1

- Persist streaming weather data from the incoming SWIM ITWS weather feed and store it in HBase Tables.
- Read the data from HBase Tables and display it on clients (for example Google Maps).
- Approach should be highly available, fault-tolerant, have low latency, and be scalable to meet fluctuating (seasonal) load.

Integration Solution # 1

Integration Solution # 1 (continued)

```
…
my $stomp_in;
eval {
local $SIG{ ALRM} = sub { die "timed out\n" };
alarm( 10);
$stomp_in = Net::Stomp->new( { hostname => $DATA_SRV_IP, port => '61613' } );
$stomp_in->connect( { login => '*********', passcode => '**********' } );
$stomp_in->subscribe({
                        destination => $DATA_TOPIC,
                                'ack' => 'client',
    'activemq.prefetchSize' => 1
 });
 };
…
my $frame;
my $x;
eval {
local $SIG{ ALRM} = sub { die "no data\n" };
alarm( 10);
$frame = $stomp_in->receive_frame;
$x = $frame->body;
alarm( 0);
};
…
$COLUMN_FAMILY = $prod_type;
$KEY = $gentime;
my $col_start = 'itws-' . $gentime . '-' . $exptime . '-' . $tracon . '-' . $airport . '-' . $prod_type . '-' . $rectime . '-' . $srv_time . '.xml';

my $mutation1 = Hbase::Mutation->new({ column => "$COLUMN_FAMILY:$col_start\.airport", value => $airport });
my $mutation2 = Hbase::Mutation->new({ column => "$COLUMN_FAMILY:$col_start\.exptime", value => $exptime });
my $mutation3 = Hbase::Mutation->new({ column => "$COLUMN_FAMILY:$col_start\.content", value => $x });

my $mutations = [ $mutation1, $mutation2, $mutation3 ];

my $date = qx/date/;
chomp($date);
print LOG $date . "> Adding data to HBase... ";
$client->mutateRow($TABLE, $KEY, $mutations);
if ($? == 0) {
   print LOG "Success.\n";
   print LOG "\$client->mutateRow ($TABLE, $KEY, $mutations)\n";
}
else { print LOG "ERROR!\n" }
…
```

Integration Challenge # 2

- Periodically (for example every 5 minutes) check current weather information (for example wind speed/direction) from an external weather feed (for example Yahoo Weather).
- Translate this data into compatible data formats.
- Store it in an HBase table so it can be accessed by clients.

Integration Solution # 2

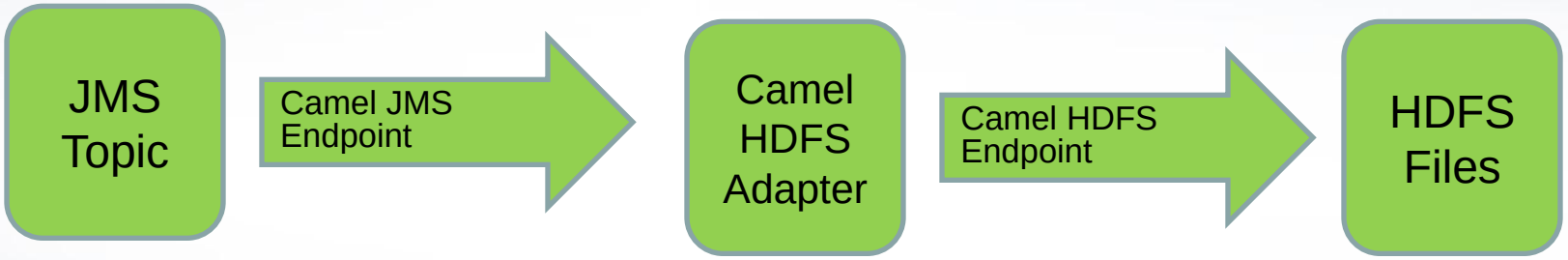
Integration Solution # 2 (continued)

```
public class MyRouteBuilder extends RouteBuilder {
      @Override
      public void configure() throws  Exception {
      from("timer:myTimerEvent?fixedRate=true&period=300000")
      .process(new WeatherProcessor())
      .to("ahc:get_wx")
      .convertBodyTo(String.class)
      .process(new HBaseWriter())
      .transform(simple("${in.headers.Last}"))
      .to("ahc:store_data");
      }
}

public class HBaseWriter implements Processor {

      public void process(Exchange exchange) throws Exception {
            SimpleDateFormat sdf = new SimpleDateFormat("yyyyMMddhhmmss");
            String wxTime = sdf.format(new Date());
            exchange.getIn().setHeader(Exchange.HTTP_URI, "http://localhost:8080/yahoowx/"+wxTime+"/currentwx:NYC");
            exchange.getIn().setHeader(Exchange.HTTP_METHOD, ("POST"));
            exchange.getIn().setHeader(Exchange.CONTENT_TYPE, ("application/octet-stream"));
            String currentWx = exchange.getIn().getBody(String.class).replaceAll("\"", "");
            exchange.getIn().setHeader("Last", currentWx);
      }
}


public class WeatherProcessor implements Processor {

      @Override
      public void process(Exchange exchange) throws Exception {
            exchange.getIn().setHeader(Exchange.HTTP_URI, "http://weather.yahooapis.com/forecastjson?jsoncallback=?&w=2459115");
      }
}
```

Integration Challenge # 3

- Read incoming compressed data from our JMS Topics.
- Unzip the XML data.
- Store the streaming data into the HDFS file system so we can access it for MapReduce (with Hadoop) or from HBase.
- Keep a log of incoming data.

Integration Solution # 3



Integration Solution # 3

```
public class ServerRoutes extends RouteBuilder {

@Override
    public void configure() throws Exception {

 from("jms:topic:FOO.ZIP.OUT")
        .unmarshal().zip()
        .to("hdfs://ip-address/output?splitStrategy=MESSAGES:1&replication=3")
        .to("log:camel");
 }
}
```

Summary

Camel Rocks!

- And so do other open source projects like ActiveMQ, ServiceMix, HBase, Hadoop …

Benefits of Using Opensource Software

- High quality software.
- When combined with topnotch support, creates an ideal programming environment.
- Enables a “Configure, don’t code” approach.

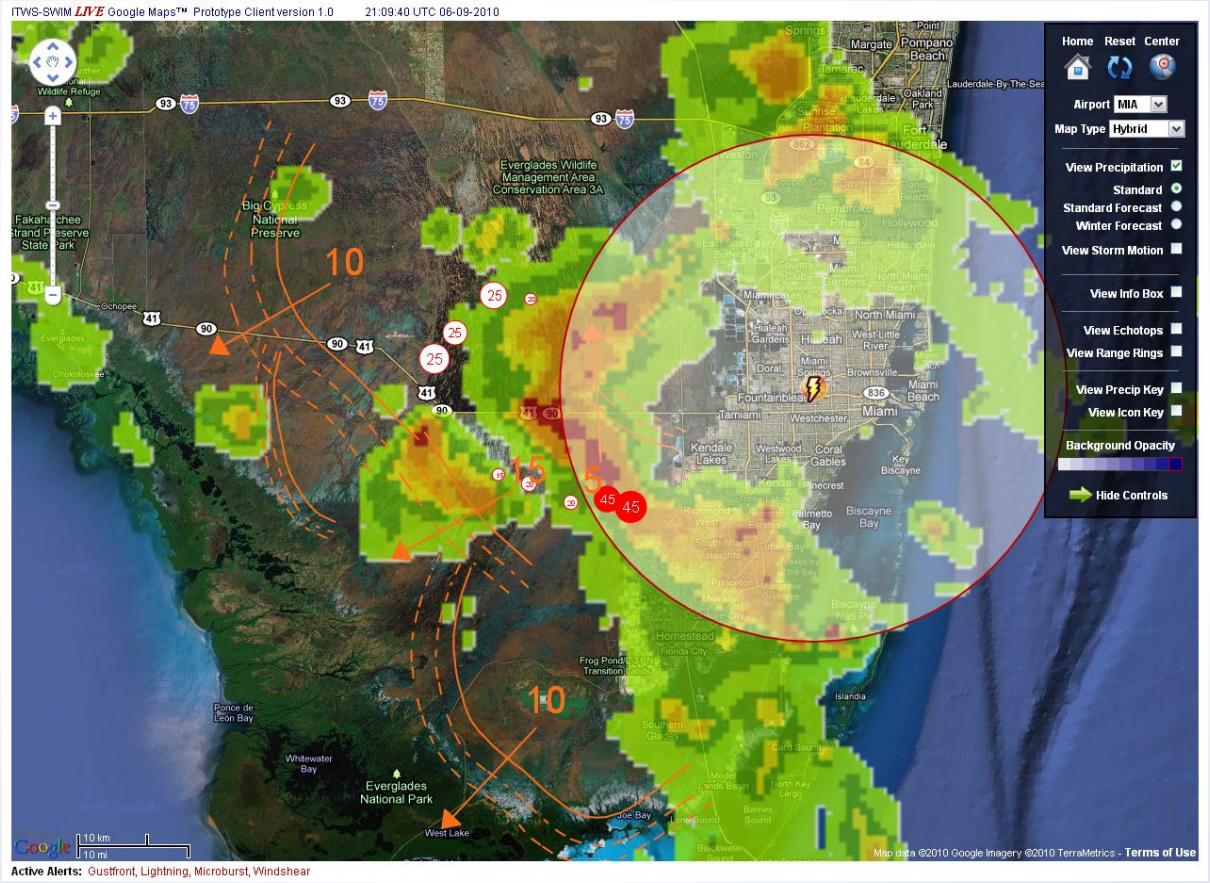
Open Source Software We're Using

- Apache Camel
- Apache ActiveMQ
- Apache Servicemix
- Apache CXF
- Apache Hadoop
- Apache HBase
- Apache ZooKeeper
- Apache Maven

Data Quality Assurance

How we ensured that the legacy data translation produced correct results …

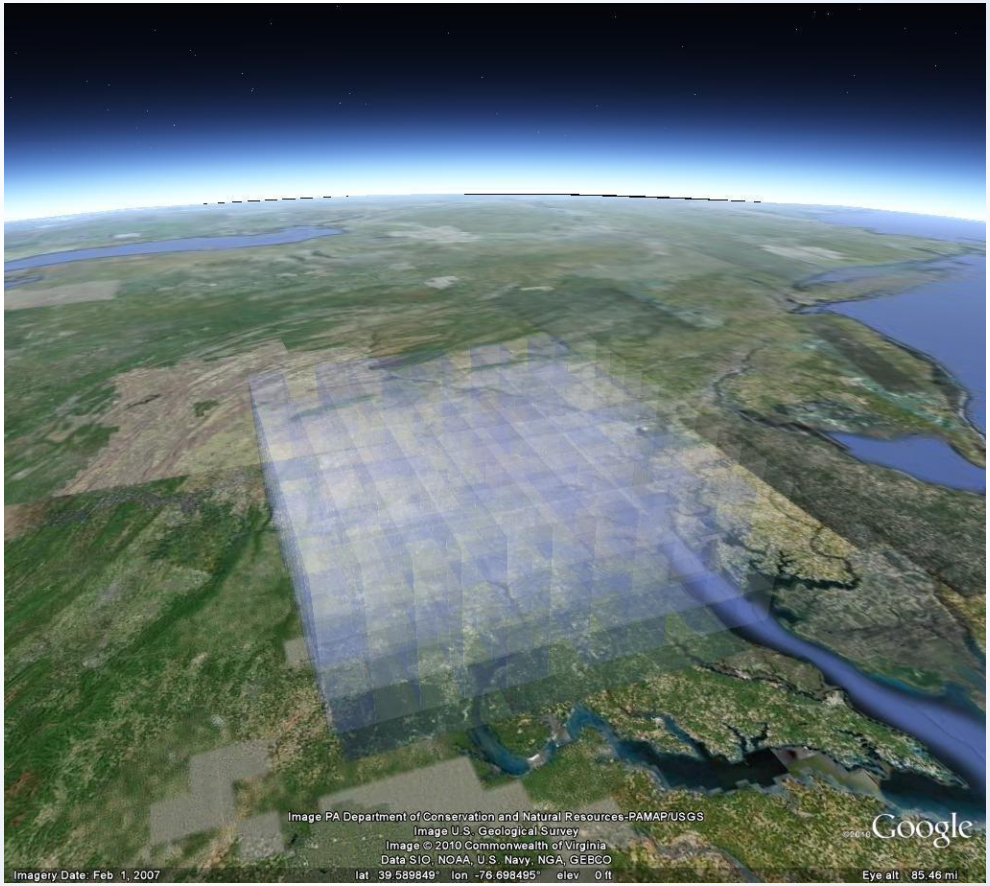
Google Maps Display of a Busy Weather Day at Miami



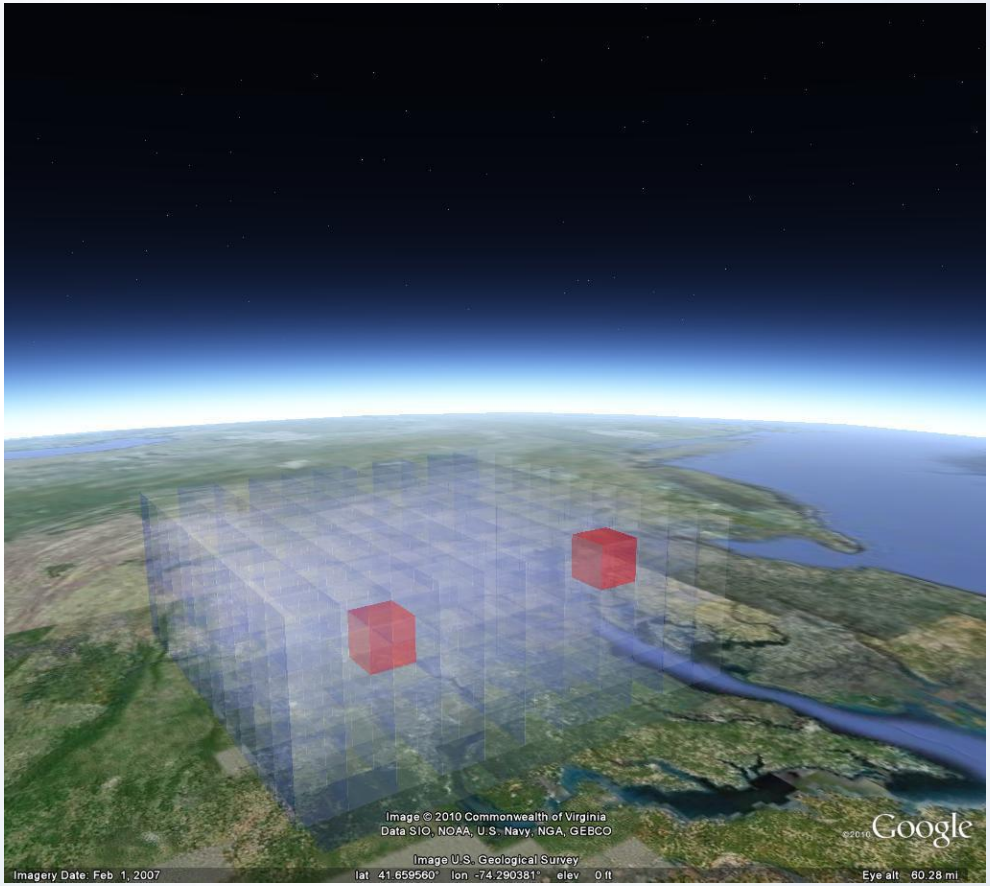
View From the Tower at JFK International Airport

Planes Taxiing at JFK

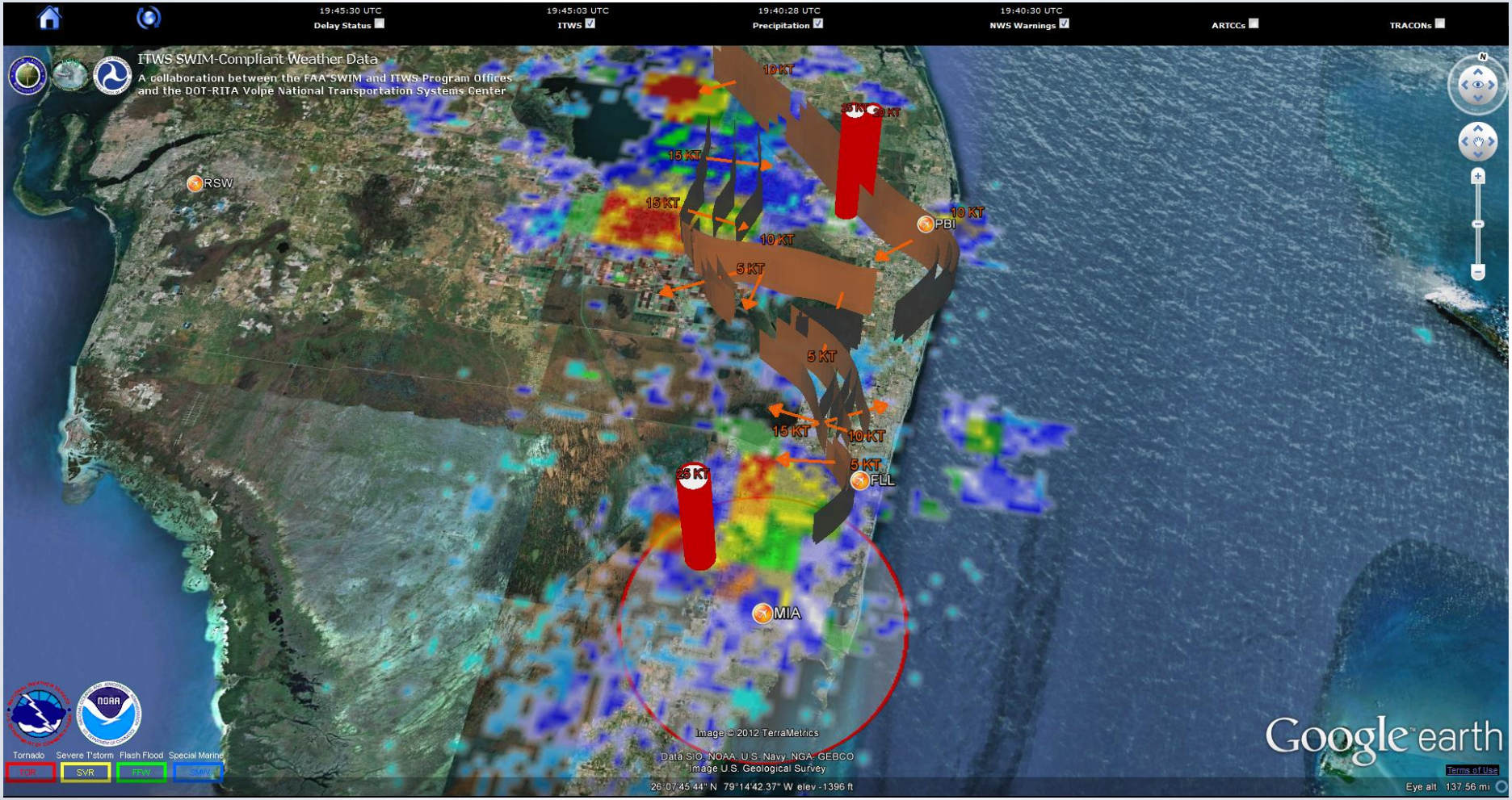
Future Concepts



Future Concepts (continued)



Google Earth API Demo



Questions?

Thank you.