Overview

• Fuser Overview
  – Swim sources
  – What is the Fuser
  – Fuser Flight Data
  – Fuser Flight Processing
  – Fuser in the Cloud
Many people have worked hard to make SWIM data available (*Thank you!*)
- Making the data available in a secure, stable platform was a major aviation engineering feat!

Pre-processing & merging SWIM flight data feeds can be difficult, expensive and error prone
- FAA decision support systems have valuable output data, but can provide inconsistent information on the same flight that is difficult for consumers to understand
- Without deep knowledge of the underlying 3T (TFMS, TBFM, TFDM - plus ERAM and STARS) systems, the consumption logic may not lead toward the benefit the community desires
- If everyone in the aviation industry creates their own SWIM flight data fusion process, many different organizations could come up with different definitions of the ‘truth’, degrading communication

The ATD-2 mission required swift progress in field (operational) demonstrations
- This led to a significant investment in logic that could address SWIM flight data pre-processing and mediation complexities. Much of this work is embodied in the ‘Fuser’ service.
- Additional analytical investment was made in post-processing, which evolved over time through an ATD-2 internal data governance process with a feedback loop into the Fuser for more data
Said another way ..

- SWIM contains lots of data
- The data has lots of value
- But consuming that data is challenging
- If only there were some way to fuse the data together...
Fuser Helps Solve SWIM Challenges

• Problem
  – With the great new FAA SWIM feeds, your organization feels they are drowning in data that they do not understand.
  – You need to make actionable, operational information out of this data!

• Solution
  – To accomplish this, you need a framework that can mediate between disparate sources of data, pulling in the right data, at the right time.
  – Need information on which data source is best to use for a specific need.
  – Access to the information in common well defined data model
Why Use Multiple Sources Of Data?

• To cover the entire flight duration and turn-around process
  – If you want the highest quality data available for the entire flight from gate to gate, this requires the use of multiple sources from SWIM
  – In general, the best data comes from the FAA system whose operational mission most closely matches your data need (e.g. if you want strategic constraints and planning info, then TFMS, tactical then TBFM or TFDM)

• Some information only exist, or is higher quality, in certain feeds
  – APREQ Release Times from TBFM
  – EDCT information from TFM Flight Data
  – Surface metering times from TFDM Terminal Publication

• Data redundancy/backup from secondary sources
  – Loss of any one feed still allows data from other feeds to provide value
What is the Fuser

• Fuser is a system composed of multiple components providing
  – Parsers for various data sources
  – Matching Services
  – Fusion Services
    • Transformation
    • Filtering
    • Updating
    • Mediation
  – Database Loading

• Common well defined schema
Fuser High Level Architecture

External Data Feeds
- TFMD ata
- SFDP S
- TB FM
- STDDS – 5MES (ASDE-X)

Data Collection
- SFDP S Connector
- TB FM Connector
- TFMD ata Connector
- Raw Data Repository
- STDDS Connector

Parsing and Matching
- SFDP S XML Parser
- TB FM XML Parser
- TFMD ata XML Parser
- STDDS XML Parser

Fusing
- Transformation Service
- Active Flight Repository
- Historical Flight Repository
- Sync Service
- Fusion Service
- Routing Service

COMMON MESSAGE BUS
- Application #1
- Application #2
- Application #3
Data Parsers
## Overview of Data Feeds

<table>
<thead>
<tr>
<th>Feed</th>
<th>Data Source</th>
<th>ATD-2 Use</th>
<th>Data</th>
</tr>
</thead>
</table>
| TfmData Flight           | TFMS        | • Schedule data  
                          |           | • CDM data  
                          |           | • Flight plans  
                          |           | • EDCTs  
                          |           | • Track data | Flight   |
| STDDS SMES               | ASDE-X/ASSC | • Surface track data  
                          |           |                          | Track    |
| TBFM MIS                 | TBFM        | • Release times  
                          |           | • EDCTs  
                          | Flight Plan | Flight Plan |
| TfmData Terminal         | TFMS        | • EOBTs  
                          |           | • Gates  
                          |           | • Tail Numbers  
                          |           | Flight       |
| TfmData Flow             | TFMS        | • Ground Stops  
                          |           | • MIT restrictions (planned)  
                          |           | TMI s       |
| SFDPS                    | ERAM        | • Support for NASA research  
                          |           | TBFM   
                          |           | Flight Plan   |
Overview of Data Feeds

The NAS systems generating flight data SWIM feeds are interconnected.
- Basic understanding the NAS architecture helps make sense of the SWIM data.

<table>
<thead>
<tr>
<th>NAS System</th>
<th>SWIM Feeds</th>
<th>Primary Inputs</th>
<th>Secondary Inputs</th>
<th>Deployments</th>
</tr>
</thead>
<tbody>
<tr>
<td>TFMS</td>
<td>TfmData Flight, TfmData Flow, TfmData Terminal</td>
<td>ERAM flight/track data, OAG schedule, Airline CDM messages, ATOP oceanic track</td>
<td>TBFM release times, STDDS surface times, OIS / NTML, International feeds, TFDM predictions</td>
<td>1</td>
</tr>
<tr>
<td>ERAM</td>
<td>SFDPS</td>
<td>ATC flight plan data, En route radar track</td>
<td>TFMS EDCTs, TFMS reroutes</td>
<td>1 per ARTCC (20 total)</td>
</tr>
<tr>
<td>TBFM</td>
<td>TBFM MIS</td>
<td>ERAM flight/track data (adjacent ARTCCs), TRACON flight/track data</td>
<td>TFMS international track, TFDM release time negotiation</td>
<td>1 per ARTCC (20 total)</td>
</tr>
<tr>
<td>ASDE-X</td>
<td>STDDS SMES</td>
<td>Surface radar track, ERAM flight data</td>
<td></td>
<td>1 per airport (38 total)</td>
</tr>
<tr>
<td>TFDM</td>
<td>TTP</td>
<td>All the above</td>
<td></td>
<td>1 per airport (future)</td>
</tr>
</tbody>
</table>
# Overview of Data Feeds

<table>
<thead>
<tr>
<th>Source</th>
<th>Flights Included</th>
<th>Earliest Flight Data</th>
<th>Latest Flight Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>TFMData Flight</td>
<td>All IFR</td>
<td>~24 hours before scheduled OAG operation</td>
<td>“Actual” arrival gate time published by Operators (up to 2 hours after taxi in)</td>
</tr>
<tr>
<td>TFMData Terminal</td>
<td>Only flights that publish CDM messages including Terminal Flight Data elements</td>
<td>~24 hours before scheduled operation</td>
<td>“Actual” arrival gate time published by Operators (up to 2 hours after taxi in)</td>
</tr>
<tr>
<td>SFDPS</td>
<td>All IFR and some VFR</td>
<td>Filed flight plan</td>
<td>Flight lands</td>
</tr>
<tr>
<td>TBFM MIS</td>
<td>Impacted by a TBFM arrival (TMA) or departure (EDC) system</td>
<td>Flight plan filed within TBFM adapted region</td>
<td>Track ends or leaves TBFM adapted region</td>
</tr>
<tr>
<td>STDDS SMES</td>
<td>Operating at or near ASDEX airport</td>
<td>First correlated flight track</td>
<td>Last correlated flight track</td>
</tr>
</tbody>
</table>
 Parsing Data Feeds

• One parser per feed
  – Transforms raw XML file into flattened object structure
    • Leveraging JAXB with jaxb2-basics to simply XML->Java Object conversion
    – In some cases, perform aggregation against messages from the same source

• Aggregation examples
  – STDDS SMES
    • Track data is sent as diffs
    • Have to aggregate with previous messages for same flight so that complete position update is processed by Fuser

  – TBFM-MIS
    • Need to be able to handle SYNC messages properly

<table>
<thead>
<tr>
<th></th>
<th>Previous</th>
<th>Current</th>
<th>To Fuser</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>12:00:01</td>
<td>12:00:02</td>
<td>12:00:02</td>
</tr>
<tr>
<td>Latitude</td>
<td>35.2156</td>
<td>-----</td>
<td>35.2156</td>
</tr>
<tr>
<td>Longitude</td>
<td>-80.9473</td>
<td>-80.9475</td>
<td>-80.9475</td>
</tr>
</tbody>
</table>
flight.getDepartureNas().getStandPositionAndTime().getStandTime().getActual().getTime()

flight.getActualOffBlockTime()
Flight Matching

External Data Feeds
- TFMDData
- SFDPSS
- TBFM
- STDDS – SMES (ASDE-X)

Data Collection
- SFDPSS Connector
- TBFM Connector
- TFMDData Connector
- STDDS Connector
- Raw Data Repository

Parsing and Matching
- SFDPSS XML Parser
- TBFM XML Parser
- TFMDData XML Parser
- STDDS XML Parser
- GUIF Service

Fusing
- Transformation Service
- Active Flight Repository
- Sync Service
- Fusion Service

COMMON MESSAGE BUS
- Application #1
- Application #2
- Application #3

Source Format
Source Parser Format
Format
Flight Matching: Overview

• Goal
  – Assign a Global Unique Flight Identifier (GUFI) to every flight message.
    • Ex: AAL1428.DFW.MCI.190507.1504.0132.TFM

• Why?
  – GUFI links together flight data across all external data feed sources.
  – Crucial precursor to data fusion.

• What is a flight?
  – Flight matching is where the “flight” is defined.
  – ATD2 defines a “flight” as the full lifecycle: starting with the scheduled or planned operation and ending at the arrival gate.
**Flight Matching: ATD2 Approach**

- **ATD2 DFW GufiService**
  - Handles GUFI requests from over 10 different data sources.
  - Stores the flight data state of each GUFI.

- **External data feed processors**
  - Request GUFI from GufiService.
  - Messages to Fuser include GUFI.

- **GufiService performance (DFW Fuser):**
  - **Total:**
    - ~15M avg messages handled per day.
    - ~170 per second.
    - ~110K avg GUFI managed per day.
    - ~200 avg messages per GUFI.
  - **DFW:**
    - ~3.5M avg messages handled per day.
    - ~7,500 avg GUFI managed per day.
    - ~1,000 avg messages per GUFI.
Flight Matching: ATD2 Approach

• Basic approach to matching a flight
  – Find the best possible flight match, if one exists in the known data.
  – Otherwise, create a new GUFI, if sufficient data exists.

• Core flight matching data
  – Aircraft ID / Callsign (e.g. SWA568).
  – Origin / Destination airports (e.g. DFW -> LGA).
  – Flight time windows (e.g. 9:30 - 13:45).
  – Position (e.g. latitude / longitude / altitude).
  – External data feed system IDs (e.g. TFMDATA flightRef).
  – Aircraft attributes (e.g. registration number, Mode S transponder)
# Flight Matching: Example

<table>
<thead>
<tr>
<th>Time</th>
<th>Message</th>
<th>TFM FlightData</th>
<th>SFDPS</th>
<th>TBFM MIS</th>
<th>STDSS SMES</th>
<th>ATD2 GUFIs</th>
</tr>
</thead>
</table>
| Sunday 19:00 | **Scheduled flight plan**  
AAL045 DFW -> CLT  
Monday 19:00-->21:59 | X              |       |          |            | AAL045.DFW.CLT.021019.1900.0000 |
| Monday 17:00 | **Filed flight plan**  
AAL045 DFW -> CLT  
Monday 19:00-->21:59 | X              | X     | X        |            | AAL045.DFW.CLT.021019.1900.0000 |
| Monday 17:30 | **Amend flight plan route**  
AAL045 DFW -> CLT  
Monday 19:00-->21:59 | X              | X     | X        |            | AAL045.DFW.CLT.021019.1900.0000 |
| Monday 18:30 | **CDM times update**  
AAL045 DFW -> CLT  
Monday 20:05-->22:15 | X              |       |          |            | AAL045.DFW.CLT.021019.1900.0000 |
| Monday 19:02 | **Scheduled flight plan**  
AAL045 DFW -> CLT  
Tuesday 19:00-->21:59 | X              |       |          |            | AAL045.DFW.CLT.021119.1902.0000 |
| Monday 19:55 | **Surface surveillance**  
AAL045 ASEX=KDFW |                |       |          | X          | AAL045.DFW.CLT.021019.1900.0000 |
| Monday 19:59 | **EDCT**  
AAL045 EDCT=20:24 | X              |       | X        |            | AAL045.DFW.CLT.021019.1900.0000 |
| Monday 20:25 | **Flight Departure**  
AAL045 | X              | X     | X        |            | AAL045.DFW.CLT.021019.1900.0000 |
| Monday 20:27 | **Airborne surveillance**  
AAL045 | X              | X     | X        |            | AAL045.DFW.CLT.021019.1900.0000 |
Flight Matching: Is Simple, Right?

- Air Taxi
- Pilots Using Wrong Beacon Codes
- Callsign Amendments
- Multiple Flight Plans Filed
- Cancelations and Diversions
- Long Flight Delays

Spend 90% effort solving the last 10% of the problem.

The Easy-er Flight Data
Flight Matching: Regression Testing

• Regression Testing = Key to matching logic improvement
  – Definition: A suite of tests that are required to pass when software changes are made.
  – Matching logic improvement is heuristic-based, adaptive process.
    • There is no 100% solution. Flight data is always scheming to defeat you.
  – Goal is intelligent whack-a-mole – fix one issue without causing another.

• Purpose
  – Emulate full matching process: replaying input messages from mock sub-components through the internal GufiService logic.
  – Reliable debugging of operational use cases.
  – Each test is simple format for analysts/testers to describe a matching problem.
  – Stockpile of regression tests to run against future development.

Image Credit: TPapi
• Standard naming convention used with most data elements in the Fuser Flight
• Naming convention is based on a flattened version of the Flight Object Data Dictionary (FODD) and FIXM Schema.
• Named so that related elements are close together alphabetically
• Self-documenting for the most part
• Each name consists of three parts
  – Information category
  – Resource Type
  – Source Type
• Naming convention: category_resourceType_sourceType
• Examples:
  – arrival_runway_actual_time
  – departure_stand_airline (departure gate from an airline source)
In order to transform the data into a common flight definition we first had to do a mapping of all the data.

Challenges
- Most feeds are not well documented at the field level
- Field names vary a great deal between feeds
- IATA vs ICAO issues
- Some data is deeply nested
- Precision issues
- Data found in multiple messages in some feeds (TFMS)

Overcoming the challenges
- Start with whatever documentation we can find
- Rely on experience from team members
- Consult subject matter experts
- Analysis
- Testing
### Data Mapping Example

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>TFM Flight Transfer Mappings</th>
<th>TMA</th>
<th>TFM TFDM Data Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>acid</td>
<td>acid</td>
<td>tmaType.air.flt.aid</td>
<td>acid</td>
</tr>
<tr>
<td>aircraftAddress</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>aircraftEngineClass</td>
<td>physicalClass</td>
<td>tmaType.air.flt.Eng</td>
<td></td>
</tr>
<tr>
<td>aircraftEquipmentQualifier</td>
<td>acEqpSuffix</td>
<td>tmaType.air.flt.Typ equipment qualifier parsed from TMA element</td>
<td></td>
</tr>
<tr>
<td>aircraftRegistration</td>
<td></td>
<td></td>
<td>acftRegistrationNumber</td>
</tr>
<tr>
<td>aircraftType</td>
<td>aircraftType</td>
<td>tmaType.air.flt.Typ aircraft type parsed from TMA element</td>
<td></td>
</tr>
<tr>
<td>altitudeAssigned</td>
<td>altitudeAssigned</td>
<td>tmaType.air.flt.Ara (if tmaType.air.flt.Fps != proposed)</td>
<td></td>
</tr>
<tr>
<td>altitudeFiled</td>
<td>altitudeFiled</td>
<td></td>
<td></td>
</tr>
<tr>
<td>altitudeRequested</td>
<td>altitudeRequested</td>
<td>tmaType.air.flt.Ara (if tmaType.air.flt.Fps == proposed)</td>
<td></td>
</tr>
<tr>
<td>arrivalAerodrome.iataName</td>
<td>arrAirport</td>
<td></td>
<td>Derived from arrArpt/arrArpticao</td>
</tr>
<tr>
<td>arrivalAerodrome.icaoName</td>
<td></td>
<td>tmaType.air.flt.Apt</td>
<td>Derived from arrArpt/arrArpticao</td>
</tr>
</tbody>
</table>

**Full Fuser Mappings Available**

[https://aviationsystems.arc.nasa.gov/atd2-industry-days/fuser/Fuser-Database-Input-Mapping-Table_85328219.html](https://aviationsystems.arc.nasa.gov/atd2-industry-days/fuser/Fuser-Database-Input-Mapping-Table_85328219.html)
Fuser Flight Processing

Fuser Sources
Airline Sources
STBO

Fuser Database

FAA Sources

Transform to MatmFlight

MatmFlight Update

Apply Filters to Incoming Message

Find Existing Flight or Create New One

Run Pre-Updaters on Incoming Message

Identify changed fields in the incoming message

Apply Mediation Rules to Changes

Update Fuser Flight

Run Post-Updaters on Outgoing Updates

Archive Updated Flight in DB

Publish Flight Update to STBO
Filtering

- Filtering is used to filter out an entire update before the data is applied to the fused flight
  - Eliminate unnecessary processing and/or updates that reduce data quality.
  - Filtering is based on defined rules:

<table>
<thead>
<tr>
<th>Filter Name</th>
<th>Description</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttributeFilter</td>
<td>See Industry Day Documentation for more details Link.</td>
<td>In certain cases, we know that a source has bad data in certain fields, we don't trust the values, or we feel we will have better data from another source.</td>
</tr>
<tr>
<td>FlightPositionFilter</td>
<td>Filter out the position update if that latitude, longitude, or timestamp is null.</td>
<td>The systems needs to have all three to have a valid position. At times we are getting bad data with one of those three fields missing causing problems downstream.</td>
</tr>
<tr>
<td>GufiFilter</td>
<td>Filter out any messages that have not been assigned a GUFI (Global Unique Flight Identifier).</td>
<td>These are messages we were unable to match typically due to a lack of information.</td>
</tr>
<tr>
<td>LocationFilter</td>
<td>Filter out any messages that are not arriving at CLT, departing from CLT, or a Surveillance target at CLT.</td>
<td>In this case, the Fuser was for an STBO system operating for Charlotte Douglas International Airport (CLT) and Therefore only needed data relevant to CLT. Implemented to keep performance under control by not processing data not relevant to CLT.</td>
</tr>
</tbody>
</table>
Data Synchronization/Mediation

• Fuser mediation processing
  – Input data correction
    • Data source has known errors or anomalies
  – Value-added data computations
    • Create new data elements not available in the input data source
  – Input data source priority rules
    • Used to define a precedence/authority between sources providing data for the same data elements.
    • Implemented when necessary
      – Default behavior allows any data source to modify a value
    • Filter out data modifications from one data source, if data modifications already exist from a higher priority source.
      – E.g. TFMDData messages are more favorable in setting the Fuser schema “routeText” data element than TBFM MIS.

• More information
  – https://aviationsystems.arc.nasa.gov/atd2-industry-days/fuser/Data-Mediation-Overview_85328193.html
Fuser Metadata

- In support of the mediation rules, Fuser tracks metadata of each Fuser schema data element:
  - Last modification input data source and message type.
  - Last modification timestamp.

Fuser Schema

```xml
<xs:element name="altitudeAssigned" type="xs:double" minOccurs="0"/>
<xs:element name="altitudeRequested" type="xs:double" minOccurs="0"/>
<xs:element name="altitudeFiled" type="xs:double" minOccurs="0"/>
<xs:element name="speedFiled" type="xs:double" minOccurs="0"/>
<xs:element name="arrivalAerodrome" type="mc:aerodrome" minOccurs="0"/>
<xs:element name="arrivalFixActual" type="xs:string" minOccurs="0"/>
<xs:element name="arrivalFixUser" type="xs:string" minOccurs="0"/>
<xs:element name="arrivalFixSourceData" type="xs:string" minOccurs="0"/>
<xs:element name="arrivalFixPositionDerived" type="xs:string" minOccurs="0"/>

<xs:complexType name="aerodrome">
  <xs:sequence>
    <xs:element name="iataName" type="xs:string" minOccurs="0"/>
    <xs:element name="icaoName" type="xs:string" minOccurs="0"/>
    <xs:element name="faaLid" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

Fuser MetaData

```xml
<xs:complexType name="metaData">
  <xs:sequence>
    <element name="source" type="xs:string" minOccurs="0"/>
    <element name="timestamp" type="xs:dateTime" minOccurs="0"/>
    <element name="systemType" type="xs:string" minOccurs="0"/>
    <element name="fieldName" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
Fuser Airport Mediation Use Case

• Airport Conversion and Correction by Source
  – Mapping over 27k ICAO / IATA / FAA LID.
  – Based upon input data source, set ICAO, IATA, and FAA LID values for all messages.

• Special Cases and Data Anomalies
  – TBFM MIS “airport” can be a waypoint.
    • Airborne IFR filed flight plans.
  – TFM Schedule flight plan OAG errors.
    • E.g. Doha OTBD airport closed in 2014 (replaced by OTHH), but is published in TFMS Scheduled flight plan messages.
Fuser Airport Mediation Use Case

- **Fuser Airport Mediation Rule**
  - **Problem:** When flights amend the arrival airport, some TFM messages continue to use the old arrival airport.
    - Without mediation, the arrival airport toggles between the current and old values.
  - **Solution:** Fuser mediation to ignore data modifications by TFM sources that may use the incorrect arrival airport.

Fuser Mediation Rule

```xml
<bean id="fuser-rules.AerodromeRule" class="com.mosaictm.fuser.rules.MultiplePriorityMediationRule">
  <property name="active" value="${fuser.rules.AerodromeRule.enable}" />
  <property name="priority" value="${fuser.rules.AerodromeRule.priority}" />
  <property name="name" value="${fuser.rules.AerodromeRule.name}" />
  <property name="idLookup" ref="fuser.core.NaNIdLookup" />
  <property name="sourcePriorityListFromString">
    <list>
      <value>TFM_FLIGHT_PLAN_INFORMATION,TFM_FLIGHT_PLAN_AMENDMENT_INFORMATION,TMA,SFDPS,FUSER</value>
      <value>TFM_FLIGHT_SCHEDULE_ACTIVATE,TFM_FLIGHT_CREATE,TFM_FLIGHT_MODIFY,TFM_TFDM</value>
    </list>
  </property>
  <property name="includes">
    <list>
      <value>arrivalAerodrome.iataName</value>
      <value>arrivalAerodrome.icaoName</value>
      <value>arrivalAerodrome.faaLid</value>
      <value>departureAerodrome.iataName</value>
      <value>departureAerodrome.icaoName</value>
      <value>departureAerodrome.faaLid</value>
    </list>
  </property>
</bean>
```
### Fuser Airport Mediation Use Case

- **Arrival Airport Mediation Messaging Example.**

```xml
<property name="sourcePriorityListFromString">
  <list>
    <value>TFM_FLIGHT_PLAN_INFORMATION,TFM_FLIGHT_PLAN_AMENDMENT_INFORMATION,TMA,SFDPS,FUSER</value>
    <value>TFM_FLIGHT_SCHEDULE_ACTIVATE,TFM_FLIGHT_CREATE,TFM_FLIGHT_MODIFY,TFM_TFDM</value>
  </list>
</property>
```

<table>
<thead>
<tr>
<th>Time</th>
<th>Source</th>
<th>Arrival Airport</th>
<th>Fuser</th>
</tr>
</thead>
<tbody>
<tr>
<td>1500</td>
<td>TFMData Schedule</td>
<td>OTBD (error)</td>
<td>DOH OTHH OTHH 1500 (TFMData Schedule)</td>
</tr>
<tr>
<td>0900</td>
<td>Airline Source</td>
<td>DOH</td>
<td>DOH OTHH OTHH 1500 (TFMData Schedule)</td>
</tr>
<tr>
<td>1100</td>
<td>TFMData Filed Flight Plan</td>
<td>OTHH</td>
<td>DOH OTHH OTHH 1100 (TFMData Filed Flight Plan)</td>
</tr>
<tr>
<td>1130</td>
<td>TFMData Amend Flight Plan</td>
<td>OMDB</td>
<td>DXB OMDB OMDB 1130 (TFMData Amend Flight Plan)</td>
</tr>
<tr>
<td>1150</td>
<td>TFMData Flight Modify</td>
<td>OTHH</td>
<td>DXB OMDB OMDB 1130 (TFMData Amend Flight Plan)</td>
</tr>
</tbody>
</table>
Flight Matching

External Data Feeds
- TFMDData
- SFDPS
- TBFM
- STDDS – SMES (ASDE-X)

Data Collection
- SFDPS Connector
- TBFM Connector
- STDDS Connector

Raw Data Repository

Parsing and Matching
- SFDPS XML Parser
- TBFM XML Parser
- TFMDData XML Parser
- STDDS XML Parser

GUI Service

Fusing
- Transformation Service
- Active Flight Repository
- Historical Flight Repository

Sync Service
- Fusion Service
- Routing Service

COMMON MESSAGE BUS
- Application #1
- Application #2
- Application #3
Where does the Fuser Data Go

- **Redis (Recovery)**
- **ActiveMQ**
- **Postgres Database**
- **STBO**
- **Fuser Clients**
  - Fuser Database Loader
  - Fuser STBO Bridge
  - Fuser Test Tool
Additional Benefits

• Flight Management
  – Create, update, delete/remove

• Built in recovery solution using Redis

• Reconstitution available for Fuser clients

• Built in performance monitoring

• Pluggable
  – Pluggable architecture for new data feeds
Technology Stack

Fuser Architecture

Redis
Postgresql
Drop Wizard

Messaging

Activemq
Apache CXF

Core Fuser

Java
Apache Camel
Spring Framework
• Fuser in the cloud is the fastest and easiest way to get access to the fuser data (currently also happens to be the only way)
• Fuser in the Cloud allows you to test the water before going all in
• Fuser gets you closer to where you want to be faster
Fuser In The Cloud Benefits

Pros
- Reduced Development Time
- Reduce Operational Cost
- Enhancements
- Lower Maintenance

Cons
- Lack Full Control
- More difficult to customize for specific needs
Fuser In The Cloud Architecture

- Running on Amazon Web Services (AWS)
- EC2 instances with Elastic Load Balancers ELBs sitting in front
  - Users connect to a dedicated ELB
- Activemq is the broker
Fuser In Cloud Pilot Program

• Pilot program still in the early stages
  – Learning how to deal with challenges as we go
  – Most of the challenges have been related to data sharing, security, and process

• Pilot/beta participants
  – Delta Airlines
  – United Airlines
  – American Airlines
  – Southwest Airlines
Connecting to the Fuser

• Develop a fuser client consumer
  – Connect to Fuser Activemq Topic/Queue
  – Consume data
  – Parse Fuser XML data
  – Build your own internal flight repository to manage the data

OR

• Use the Fuser Client API
  – The Fuser Client API is a utility for establishing a connection and interfacing with the Fuser
  – All data consumed from the Fuser will be automatically converted from XML back to a Fuser Flight structure
• Quickly develop Fuser driven applications
• Handles connecting and managing the connection
• Parses the XML
• Creates an in memory repository
• Support for event listeners
Ways to Use the Fuser

• The Fuser is the whole processing backbone of ATD-2

It doesn’t have to be used as your infrastructure backbone

• Use it a single source SWIM feed
  – Schema is easier to work with compared to SWIM schemas
  – Only have to learn how to consume one format. One stop shopping!
  – Flexible enough to filter down to just sources you care about
Challenges

- Sharing CDM data requires data sharing agreements to be in place
- Sharing the Fuser Client API may require a data sharing agreement
- Still working on long term hosting solution beyond ATD-2
- Security and process requirements on both the producer and consumer side

*** NASA ATD-2 team is working with the FAA and Industry Partners to solve these challenges ***
Future

• Fuser in the Cloud
• Build Data Services On Top and Make Them Available

Let’s Get Here!
• Fuser Overview
• Component Descriptions
• Fuser Data Dictionary
  – Complete list of fields found in the Fuser
  – Field description
    • Most fields were meant to be self documenting
• SWIM data to Fuser Mappings
  – Likely more valuable than the data dictionary itself
• References to other SWIM related documentation
• https://aviationsystems.arc.nasa.gov/atd2-industry-days/fuser/ATD-2-Industry-Day-Documentation-Outline_81565170.html
Questions
Image Credits

• “Whack A Mole Fever,” Author: TPapi, https://www.flickr.com/photos/tpapi/2765541278/, Creative Commons License CC BY-NC-SA 2.0, https://creativecommons.org/licenses/by-nc-sa/2.0/