Adaptive Fusion Tracker (AFT) Frequently Asked Questions (FAQ):

**Operating Environment**

**Architecture and Design**

**Sensors and Interfaces**

### Operating Environment:

1. **What operating system is the tracker operating under?**
   
   AFT currently supports a variety of Linux OS distributions (such as RHEL, CentOS, and Fedora) in addition to other Real Time Operating Systems (RTOS) like VxWorks.

2. **Can AFT run under other Operating Systems?**
   
   Yes, the AFT development environment is Rhapsody UML Tool which provides built-in OS abstraction called Rhapsody Framework. Rhapsody Framework supports many OS platforms so it’s a straightforward procedure to migrate AFT to another OS. The AFT architecture also includes a global layer that realizes machine abstraction (called Platform Adaption) by providing common base modules for OS functionality, like locks and threads; this approach decouples algorithms from the business logic of the platform.

3. **Does the AFT require a 32-bit or 64-bit operating system?**
   
   AFT can be built to run in either 32 or 64 bit platforms.

4. **What is the execution speed of the AFT?**
   
   AFT is designed and built for the real-time surveillance environment. Actual performance is dependent on several factors including configuration of the host computer, sensor types (namely timeliness of detection reports and fidelity of sensor data), and overall system track load. To provide an example based on a recent deployment, AFT operates on a virtual machine (VM) hosted on a Dell Blade server running RHEL 6. Under this environment, the customer regularly observes a track load in the hundreds of tracks; however, there is nothing in the software that bounds the track picture to a max value. AFT is fully capable of running on multi-processor/multi-core machines (as processing power requirements may dictate) which is transparent to the application.

### Architecture and Design:

5. **How is the AFT designed/maintained and what language is AFT developed in?**
AFT is developed with the IBM Rational Rhapsody in C++ UML Tool which directly generates and compiles C++ code from the model driven design. Since source code is generated by the model, design and implementation do not diverge. AFT was designed entirely by Leidos engineers at our St. Petersburg, FL division. AFT does not incorporate any third party proprietary software and utilizes open source products (like GNU) whenever possible, and Leidos maintains full rights to the AFT software. Comprehensive configuration control and management are performed at the St. Petersburg facility.

6. Does AFT support real-time communications and to what extent is the software separable?
AFT has integrated OMG’s Pub/Sub protocols via COTS middleware, such as RTI DDS and PrismTech OpenSplice DDS. AFT is entirely separable from the middleware and can be built with or without the dependency based on customer requirements. Further, the AFT interface package is fully configurable at runtime to select which communication layer to instantiate from a specific DDS implementation, web service, TCP/IP or UDP.

7. What documentation can be provided:
   a. Does a user manual exist?
      Yes, a contractor formatted User/Installation manual does exist.
   b. Does a system model design document exist?
      Since AFT has been developed in the IBM Rational Rhapsody UML, the tool provides a reporter feature that generates documentation directly from the model to capture system design and source with text and diagrams.
   c. Does a design guide exist?
      Yes, the design is in UML, the standard modeling language endorsed by the Object Management Group. A Rhapsody wiki page exists (see link below) for tutorials, samples, and demos will walk you through the process of designing embedded systems with exercises in creating UML diagrams, generating code, and building the executable image.

       https://www.ibm.com/developerworks/wikis/display/Rhapsody/Tutorials,+samples+and+demos
   d. Does interface description documentation exist?
      Yes, IDDs for various external interfaces (including sensors, clients, displays, etc.) do exist and in many cases a complete XML schema can be made available.

8. Can the AFT be run in a hands-off, no-operator mode?
   Yes. AFT is designed to run with no operator in the loop, and supports the automatic reporting of track data to a variety of external visualization tools. For hands-on usage, we provide a web-server which allows the user to visualize the complete track picture, start/stop data recording, playback recorded data, generate/maintain geospatial zones, and view amplifying track information, all from the users’ web browser.

9. What is sensor playback?
The AFT includes a playback feature which allows the user to inject either recorded sensor data or canned target scenarios into the AFT without the need to evaluate/test the system with full-up live sensor set. This is useful for new sensor integration as well as performance and forensic analysis.

**Sensors and Interfaces:**

10. What types of Sensors, data types and sensor update rates does AFT Support?
    AFT supports sensors in the air, maritime and ground domains. See table for additional information.

<table>
<thead>
<tr>
<th>Sensor Type</th>
<th>Domain</th>
<th>Format</th>
<th>Notional Update Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFF</td>
<td>Air</td>
<td>CD-2</td>
<td>Variable: seconds</td>
</tr>
<tr>
<td>TPS-70</td>
<td>Air</td>
<td>TPS-70</td>
<td>Variable: seconds</td>
</tr>
<tr>
<td>LSTAR</td>
<td>Air</td>
<td>ASTERIX</td>
<td>Variable: seconds</td>
</tr>
<tr>
<td>TPS-59</td>
<td>Air</td>
<td>Native Binary</td>
<td>Variable: seconds</td>
</tr>
<tr>
<td>ADS-B</td>
<td>Air</td>
<td>XML</td>
<td>Variable: seconds</td>
</tr>
<tr>
<td>SR Hawk</td>
<td>Ground</td>
<td>XML</td>
<td>Variable: seconds</td>
</tr>
<tr>
<td>Spotter Hawk</td>
<td>Ground &amp; Maritime</td>
<td>JASON</td>
<td>Variable: seconds</td>
</tr>
<tr>
<td>Koden</td>
<td>Maritime</td>
<td>NMEA 0183</td>
<td>Variable: seconds</td>
</tr>
<tr>
<td>SRR</td>
<td>Maritime</td>
<td>COMRIC</td>
<td>Variable: seconds</td>
</tr>
<tr>
<td>AIS</td>
<td>Maritime</td>
<td>NMEA, XML</td>
<td>Variable: seconds - minutes</td>
</tr>
<tr>
<td>Direction Finding Antenna</td>
<td>RF Transmitters</td>
<td>Native Binary</td>
<td>Variable: seconds</td>
</tr>
</tbody>
</table>

11. How is the asynchronous timing of different sensors handled?
    AFT’s positional filter was designed to smooth noise of asynchronous positional reports from multiple sensors. The accuracy of the time stamp is important, and AFT utilizes a Network Time Protocol (NTP) solution to achieve system synchronicity.

12. Is AFT capable of correcting for sensors biases?
    Yes, AFT has a Track Bias function that actively estimates bias correction between contributing sensors. An operator entered bias correction is also possible by which the user seeds or overrides the running bias correction.

13. Is there any prioritization of sensor handling provided?
    The most accurate sensors are weighted the highest in the track state.

14. Is there ability to differentiate between “search”, “track”, and “classification” sensors?
    Yes, the sensor registration file captures the various characteristics of each sensor model.

15. Is there any ability to steer a video camera?
    We have demonstrated this capability with the Web User Interface. For instance, the AFT system has successfully cued cameras based on the reports from contributing AIS receivers and RF radars.
16. Is a geo-based user interface required for operation?
   No, neither a graphical user interface nor maps are required for AFT operation; however, alert zones are typically configured at Google Earth, and then imported/handled by AFT. In short, while the AFT supports a geospatially enabled GUI it is not required for operation.

17. What types of data is available from AFT and what formats are supported?
   AFT is configurable to report any combination of the following: new tracks, track updates, track pairs, track drops, and alerts; there is a data recording function that captures raw sensor interfaces and internal track processes. This data may take the form of XML, KML, DDS, or native data formats (in the case of some sensors). AFT also generates KML files for the AFT web user interface, client tools, and Google Earth for situational awareness and forensic analysis.