ITS Sensor Transfer Spec

Michael Cotton

Kenneth Baker

Transfer Spec Mandate

- There is a broad set of missions for sensing networks
- There is a broad set of requirements for sensors
- Multiple sensor types will be needed at different locations and for different missions
- Ergo:

The transfer spec must be general enough to accommodate the most basic of sensors as well as the most sophisticated.

Classes of Sensors

- Network of stationary, single function sensors
- Network of mobile, single function sensors
  - Live stream
  - Store and forward
- Network of stationary, programmable sensors
  - Adaptive Antenna Control
  - Signal Identification Functions
  - Timing and Emitter Location Functions
  - etc.
- Network of mobile, programmable sensors
  - Live stream
  - Store and forward

Possible Sensor Network Functions

- Signal Detection
- Signal Identification
- Emitter location
  - Timing constraints
  - Antenna control
- Emitter tracking
  - Timing constraints
  - Antenna control
- Propagation data and control
- Environmental Data

Transfer Spec Functionality

- Permit Sensor Control
  - Two way communication
- Permit Precision Timing of events
- Permit real time streaming of data
- Enable store and forward connectivity
- Permit transfer of location information
- Permit various types of data
  - PSD
  - I/Q
  - Calibration Data
  - Environmental, etc.

Existing Standards

- VITA49
  - No two way communication and control
- IEEE 1900.6
  - Currently under review
  - Will likely need modification for our requirements
- 802.22.3

Current ITS Transfer Spec

- JSON based

ITS Message Format

- Messages in JavaScript Object Notation (JSON)
- Example LOC message:


```
{
   "version": "1.0.16",
   "messageType": "Loc",
   "sensorId": "101010101",
   "sensorKey": 846859034,
   "time": 987654321,
   "mobility": "Stationary",
   "environment": "Outdoor",
   "latitude": 40.0,
   "longitude": -105.26,
   "altitude": 1655,
   "timeZone": "America_Denver"
}
```

Required Data Fields

- version = Schema/data transfer version with the major.minor.revision syntax [string]
- messageType = Type of JSON message (“Sys” | ”Loc” | “Data” |"Capture-Event") [string of URL unreserved characters]
- sensorId = Unique identifier of sensor [string
- sensorKey = Authentication key given out by MSOD [integer]
- time = Time [seconds since Jan 1, 1970 UTC] long [integer]

Current Message Types

- Sys Messages
- Loc Messages
- Data Messages
- Capture-Event Messages

Sys Messages

- Sys (System) message lists the critical hardware components of the sensor along with relevant RF specifications

1. version = Schema/data transfer version with the major.minor.revision syntax [string]
2. type = Type of JSON message (”Sys”) [string]
3. sensorId = Unique identifier of sensor [string of URL unreserved characters]
4. sensorKey = Authentication key given out by MSOD [integer]
5. time = Time [seconds since Jan 1, 1970 UTC] long [integer]
6. antenna = data that describes the antenna (see Antenna object below)
7. preselector = data that describes RF hardware components in preselector (see Preselector object below)
8. cotsSensor = data that describes the COTS sensor (see COTSsensor object below)
9. calibration = data structure that describes the calibration measurement (optional, see Cal object below)

- If processed = “False”, then the data streams are:

10a. noiseSourceOnPowers(n) = Raw measured data vector [dBm ref to input of COTS sensor] when known source is on.

11a. noiseSourceOffPowers(n) = Raw measured data vector [dBm ref to input of COTS sensor] when known source is off.

- If processed = “True”, then the data streams are:

10b. noiseFigure(n) = Noise figure [dB] referenced to input of preselector.

11b. gain(n) = System gain [dB] referenced to input of preselector

Loc Messages

- The Loc message specifies the geolocation of the sensor.

1. Ver = Schema/data transfer version with the major.minor.revision syntax [string]
2. Type = Type of JSON message (“Loc”) [string]
3. SensorID = Unique identifier of sensor [string of URL unreserved characters]
4. SensorKey = Authentication key given out by MSOD [integer]
5. t = Time [seconds since Jan 1, 1970 UTC] [long integer]
6. Mobility = Mobility of sensor (“Stationary” | “Mobile”) [string]
7. Lat = angle [degrees N] from equatorial plane (0 – 360) [float]
8. Lon = angle [degrees E] from Greenwich median (-180 – 180) ([float]
9. Alt = height above sea level [float]
10. TimeZone = Local time zone identifier (“America/New_York”, “America/Chicago”, “America/Denver”, “America/Phoenix”, or “America/Los_Angeles”) [string]

Data Messages

- The Data message contains acquired data from measurements of the environment using an antenna.

1. version = Schema/data transfer version with the major.minor.revision syntax [string]
2. messageType = Type of JSON message “Data” [string]
3. sensorId = Unique identifier of sensor [string of URL unreserved characters]
4. sensorKey = Authentication key for the sensor [string]
5. time = Time [seconds since Jan 1, 1970 UTC] [long integer] in the UTC time zone.
6. sysToDetect = System that measurement is designed to detect (“Radar–SPN43”| “LTE”| “None”) [string of URL unreserved characters]
7. sensitivity = Sensitivity of the data (“Low” | “Medium” | “High”) [string]
8. measurementType = Type of measurement (“Swept-frequency”| “FFT-power”) [string]
9. timeOfAcquisition = Time of 1st acquisition in a sequence [seconds since Jan 1, 1970 UTC] [long integer] in the UTC time zone.
10. acquisitionIndex = Index of current acquisition in a sequence [integer]
11. numOfMeasurements = Number of measurements per acquisition [integer]. Not relevant for streaming transfers (set to -1).
12. timeBetweenAcquisitions = Imposed time between acquisition starts [float]. This is the time between successive Data messages (not relevant for streaming transfers).
13. timeBetweenStreams = Time between spectrums when data is sent as a stream via a tcp socket (relevant for streaming transfers).
14. overloadFlag = Overload flag(s) (0 | 1) [integer]
15. detectedSysNosiePowers = Detected system noise power [dBm ref to output of isotropic antenna] [float]
16. comment [string]
17. processed = Indicator on processing of data ("True"|"False") [string]
18. dataType = Data type ("Binary–float32", "Binary–int16", "Binary–int8", "ASCII") [string]
19. byteOrder = Order of bytes for binary data ("Network" | "Big Endian" | "Little Endian" | "N/A") [string]
20. compression = Indicator on compression of data ("Zip" | "None") [string]
21. measurementParameters = Measurement parameters (elements listed in Objects section below)

If processed = “False”, then the data stream is

21a. rawMeasuredPowers(n, nM) = Raw measured data vector [dBm ref to input of COTS sensor]

If processed = “True”, then the data stream is

21b. measuredPowers(n, nM) = Measured power vector [dBm ref to output of isotropic antenna]

Capture-Event Messages

- The Capture-Event Message is used to POST an asynchronous event from the sensor to the server.

1. Ver = Schema/data transfer version with the major.minor.revision syntax [string]
2. Type = Type of JSON message “Capture-Event” [string]
3. SensorID = Unique identifier of sensor [string of URL unreserved characters]
4. SensorKey = Authentication key for the sensor [string]
5. t = Time [seconds since Jan 1, 1970 UTC] [long integer] in the UTC time zone.
6. Sys2Detect = System that measurement is designed to detect (“Radar–SPN43” | “LTE” | “None”) [string of URL unreserved characters]
7. Sensitivity = Sensitivity of the data (“Low” | “Medium” | “High”) [string]
8. mType = Type of measurement (“I_Q”) [string]
9. DataType = Data type ("Binary–float32", "Binary–int16", "Binary–int8") [string]
10. mPar = Measurement parameters (elements listed in Objects section below)
11. Decode = Detection results (elements listed in Objects section below)
12. sampleCount: Number of captured samples.

JSON Object Definitions: Antenna

1. Antenna = antennas parameters with elements
2. Model = Make/model (“AAC SPBODA-1080_NFi” | “Alpha AW3232”) [string]
3. fLow = Low frequency [Hz] of operational range [float]
4. fHigh = High frequency [Hz] of operational range [float]
5. g = Antenna gain [dBi] [float]
6. bwH = Horizontal 3-dB beamwidth [degrees] [float]
7. bwV = Vertical 3-dB beamwidth [degrees] [float]
8. AZ = direction of main beam in azimuthal plane [degrees from N] [float]
9. EL = direction of main beam in elevation plane [degrees from horizontal] [float]
10. Pol = Polarization (“VL” | “HL” | “LHC” | “RHC”, “Slant”) [string]
11. XSD = Cross-polarization discrimination [dB] [float]
12. VSWR = Voltage standing wave ratio [float]
13. lCable = Cable loss (dB) for cable connecting antenna and preselector [float]

JSON Object Definitions: Preselector

1. fLowPassBPF = Low frequency [Hz] of filter 1-dB passband [float]
2. fHighPassBPF= High frequency [Hz] of filter 1-dB passband [float]
3. fLowStopBPF = Low frequency [Hz] of filter 60-dB stopband [float]
4. fHighStopBPF = High frequency [Hz] of filter 60-dB stopband [float]
5. fnLNA = Noise figure [dB] of LNA [float]
6. gLNA = Gain [dB] of LNA [float]
7. pMaxLNA = Max power [dBm] at output of LNA, e.g., 1-dB compression point [float]
8. enrND = Excess noise ratio of noise [dB] diode for y-factor calibration

JSON Object Definitions: COTSsensor

1. Model = Make and model ("Agilent N6841A"| "Agilent E4440A"| "CRFS RFeye"| "NI USRP N210"| "ThinkRF WSA5000-108"| "Spectrum Hound BB60C") [string]
2. fLow = LowMinimum frequency [Hz] of operational range [float]
3. fHigh = HighMaximum frequency [Hz] of operational range [float]
4. fn = Noise figure [dB] of COTS sensor in contrast to overall system [float]
5. pMax = Maximum power [dBm at input] of COTS sensor [float]

JSON Object Definitions: Cal

1. CalsPerHour = Number of cals per hour [float]
2. Temp = Measured temperature inside preselctor [F] [float]
3. mType: Type of measurement (“Swept-frequency”, “FFT-power”) [string]
4. nM = Number of measurements per calibration [integer]
5. Processed = Indicator on processing of data ("True"| "False") [string]
6. DataType = Data type ("Binary–float32"| "Binary–int16"| "Binary–int8"| "ASCII") [string]
7. ByteOrder = Order of bytes for binary data ("Network", "Big Endian", "Little Endian", "N/A") [string]
8. Compression = Compression of data ("Zip"| "None") [string]
9. mPar = Measurement parameters (elements listed in Objects section below)

JSON Object Definitions: mPar

1. fStart = Start frequency [Hz] of sweep <Required for swept-freq> [float]
2. fStop = Stop frequency [Hz] of sweep <Required for swept-freq> [float]
3. n = Number of frequencies in sweep <Required for swept-freq> [float]
4. td = Dwell time [s] at each frequency in a sweep <Required for swept-freq> [float]
5. Det = Detector: ("RMS"| "Positive" | "Peak" | "Average") <Required for swept-freq> [string]
6. RBW = Resolution bandwidth [Hz] <Required for swept-freq> [float]
7. VBW = Video bandwidth [Hz] <Required for swept-freq> [float]
8. Atten = COTS sensor attenuation [dB] <Required for swept-freq> [float]
9. SampleRate = Sampling rate [Samples/second] <Required for I/Q capture>
10. fc = Center frequency [Hz] \Required for I/Q capture>

LTE Decode

- Note: <System2Detect,fStart,fStop> determine the MSOD band for which we are capturing I/Q data. fc and CaptureEvent.sampFreq determine the bandwidth of the I/Q samples. In the case of a swept frequency sensor, there could be several capture events corresponding to a single scan.
- Decode = Decoded LTE information
- algorithm = Algorithm used for detection ("coherent"|"matched-filter"|"cyclostationary")
- The following additional fields are relevant to the "coherent" scheme for LTE detection:
  - CellID = Cell identification number [integer]
  - SectorID = Sector identification [integer]
  - linktype = ("uplink" | "downlink")

Transfer Mechanism

Secure socket transport

- **Socket Setup**
  - The sensor is a pure client.
  - The client initiates the connection to the server.
- **HTTPS post**
  - Sensors may also intermittently connect and POST data by connecting to the server
- **Database (MSOD) Ingest Process**
  - Not part of the transfer spec