SOFWERX

# Physiological Analysis Tool (PAT) Architecture

Version 1.4

Jim Ladd

Hansel Rios

Jason Young

May 20, 2021

SOFWERX

# Contents


Introduction ..... 3
Overview ..... 3
External Interfaces ..... 3
Subsystem Partitions ..... 4
Domain Analysis ..... 5
Architecture ..... 8
Hardware Architecture ..... 10
Software Architecture ..... 12
- Audio/Video Streaming ..... 13
- Messaging Service ..... 13
- IP Discovery ..... 14
- File Transfer ..... 15
- Clock Synchronization ..... 16
- Analysis ..... 17

Native to Canonical Sentiment ..... 18
Artifacts ..... 19
Who We Are ..... 20
References ..... 21
Appendix A – Android Screen Shots ..... 22
Appendix B – RTSP OPTIONS and DESCRIBE ..... 25
Appendix C – ZMQ Message Definitions ..... 28
Appendix D – On-body Enclosure ..... 30

## Introduction

The Physiological Analysis Tool (PAT) provides near-real time determination of physiological and overall sentiment responses of an audience while a speaker delivers a message. The analysis can be performed either from a distance with more than one person (crowd or group), or interpersonal encounters (one-on-one). The project incorporates the sentiment output of five algorithms from four different vendors and the physiological measurements from three providers. To accommodate this wide spectrum of technologies, software, and hardware, a heterogeneous platform was developed that integrates Android, Ubuntu, and Windows computing environments. The architecture of this system is presented in this document.

## Overview

The system has three major types of actors. The Operator role presents the message to one or more members of an Audience. The Operator will start the monitoring via a user interface provided by an on-body smart phone. The PAT will activate the built in camera and microphone along with any vendor specific sensors. The data feeds will be provided to a suite of software applications for analysis. The data from each sensor will also be persisted for post-session analysis on an off-body laptop or personal computer. The Operator may terminate the analysis via the user interface and the PAT will provide a synopsis or scorecard of the session. A diagram of the system boundary is shown below:

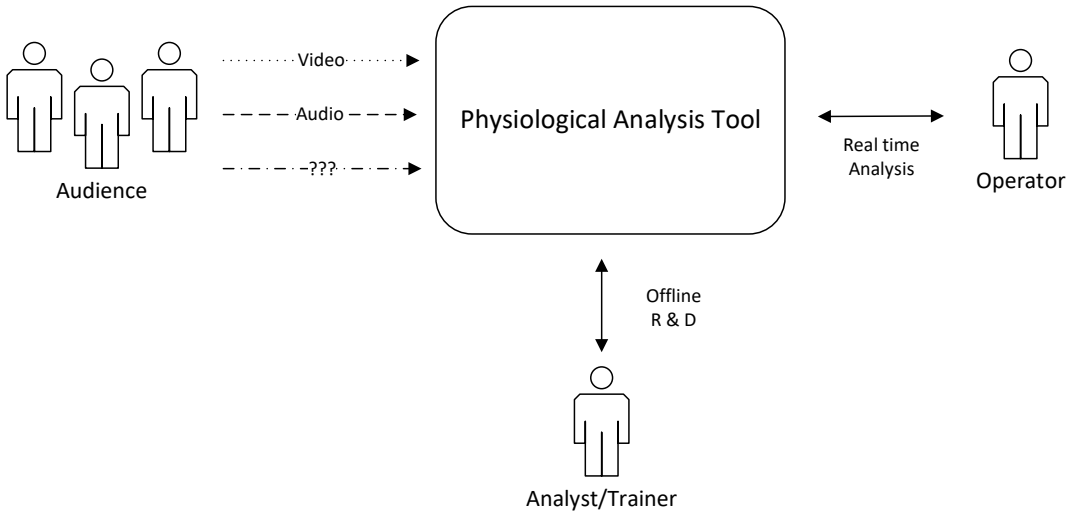


*Figure 1 - System boundary*

## External Interfaces

The external interfaces for the PAT consist of the three major actor roles; Operator, Audience, and Analyst/Trainer. There are no other interfaces to external systems or actors. All other interfaces are considered to be part of the solution.

The Operator role is the person monitoring the event while delivering the message or while another person is delivering the role. During the Physiological Analysis Tool Tech Sprint event held in December, 2019, the SMEs (Subject Matter Expert) guided the discovery activities toward a very simple user interface. This UI consists of the following features:

- ***Start Button*** – This action initiates the analysis and recording process.
- ***Status Indicator*** – This feature denotes that the PAT processing is active.
- ***Stop Button*** – This action terminates the processing.
- ***Summary*** – The overall measurement of sentiment of the audience after the conclusion of the event. In the screen mockups created during the Tech Sprint, the Summary section consisted of a single metric of percent positive.

The scenario begins with the Operator pressing the Start button to begin the session. The UI shows an "In Progress" on the display. The Operator delivers the message and then presses the Stop button. The PAT system summaries the different data feeds and internal processing and displays a value for the percent positive metric. The screen shots from the Tech Sprint are shown in Appendix A.

There are two user interface features for the Operator that may have been discussed during the Tech Sprint but were not captured in the event artifacts. The first is the ability for the Operator to enable/disable individual sensing systems to use for a given event. This mode, in effect, provides custom "sensing profiles" for different scenarios and environments. The second feature allows the Operator to review not just the summary scorecard but to drill into the underlying measurements or outputs from each sensing system.

The Audience role consists of one or more persons who are consuming the message and whose sentiment is of interest. The sensing technologies are non-contact in nature. These include video and audio as provided by commercially available smartphones and proprietary technologies by different vendors.

The Analyst/Trainer role focuses on post-event review and analysis, research and development, and training.

## Subsystem Partitions

The PAT system was partitioned into two major sub-systems. The "on-body" component provides a platform for near-real-time analysis. The architecture is designed to accommodate different analysis applications so that their collective monitoring produces a superior result than any singular technology.

The "off-body" component is responsible for post-session analysis, research, and training purposes. The data feeds from the on-body platform will be uploaded to the off-body system. The off-body system will probably consist of one or more commercial-off-the-shelf (COTS) applications. A diagram of the sub-system portioning is shown below:

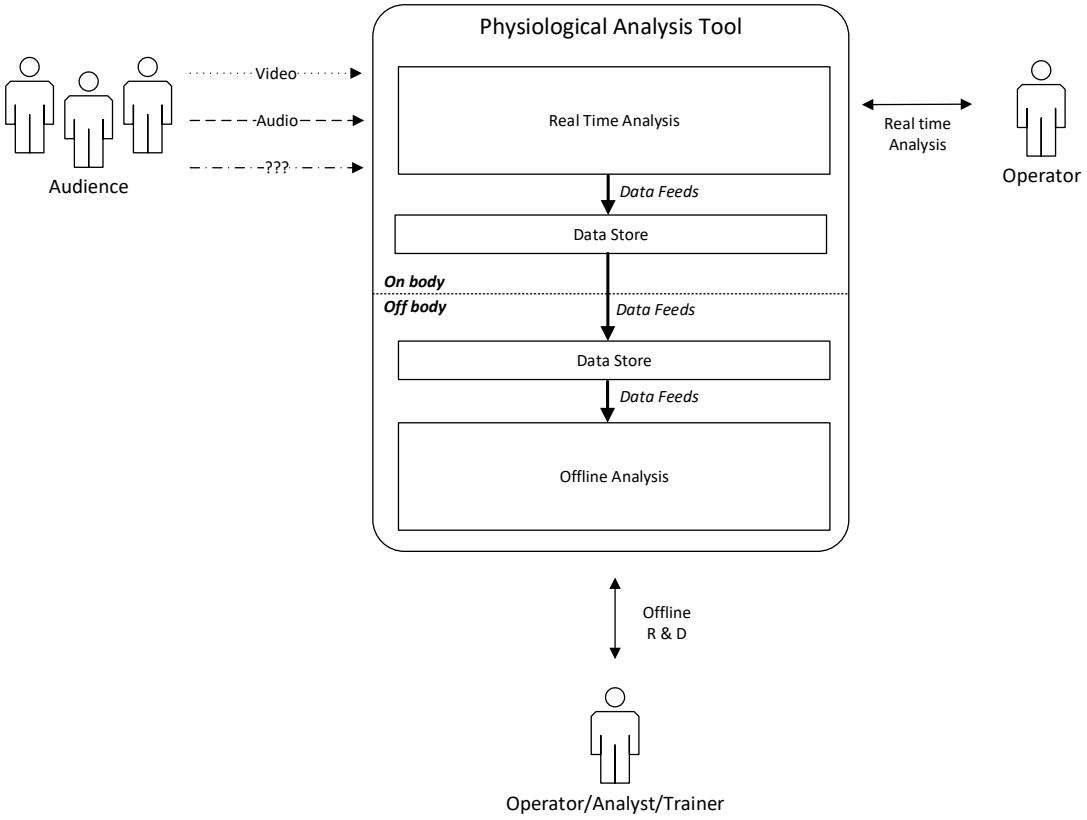


*Figure 2 - Sub-system partitions*

**NOTE:** The remainder of this document describes the on-body partition of the PAT system.

## Domain Analysis

Domain analysis consists of dividing the on-body system into distinct, independent subject matters [1]. Domains are organized in client- server relationships so that a domain acting as a client can rely on the server domain to provide needed services. The ovals in a domain diagram represent domains or clusters of similar objects and services. The arrows represent bridges between the domains, i.e., the client-server relationships. The domain diagram for the on-body platform is shown below.

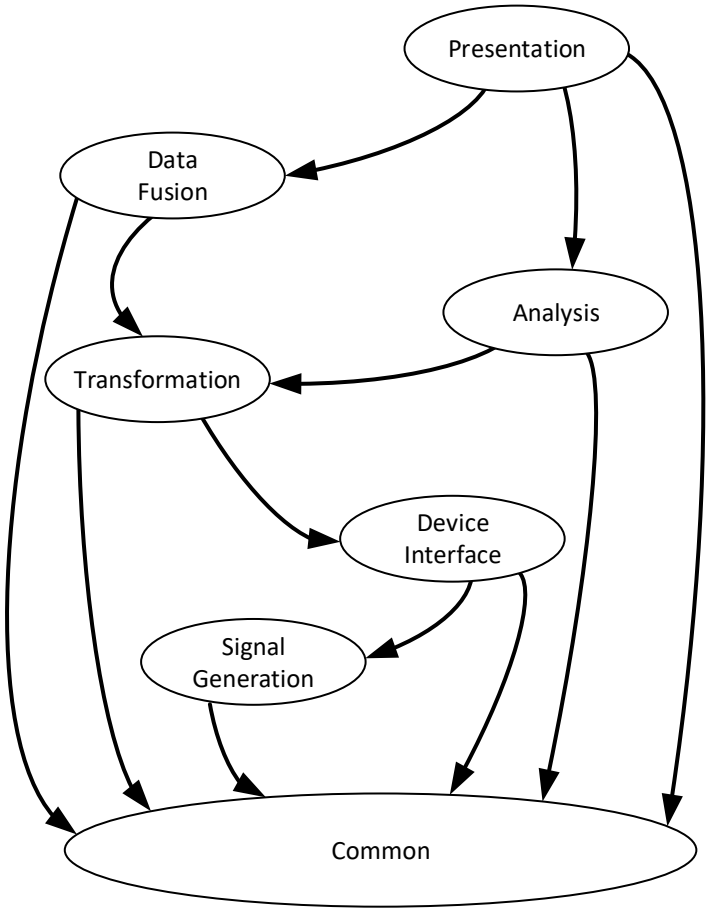


*Figure 3 - On-body domain diagram*

While domain diagrams are interesting, their primary purpose is to serve as a stepping stone to more detailed analysis. The domains for the on-body platform are further decomposed and presented in a different view shown below. This partitioning represents the "ideal" state. As the solution was developed, the "ideal" state evolved to accommodate the realities of the project.

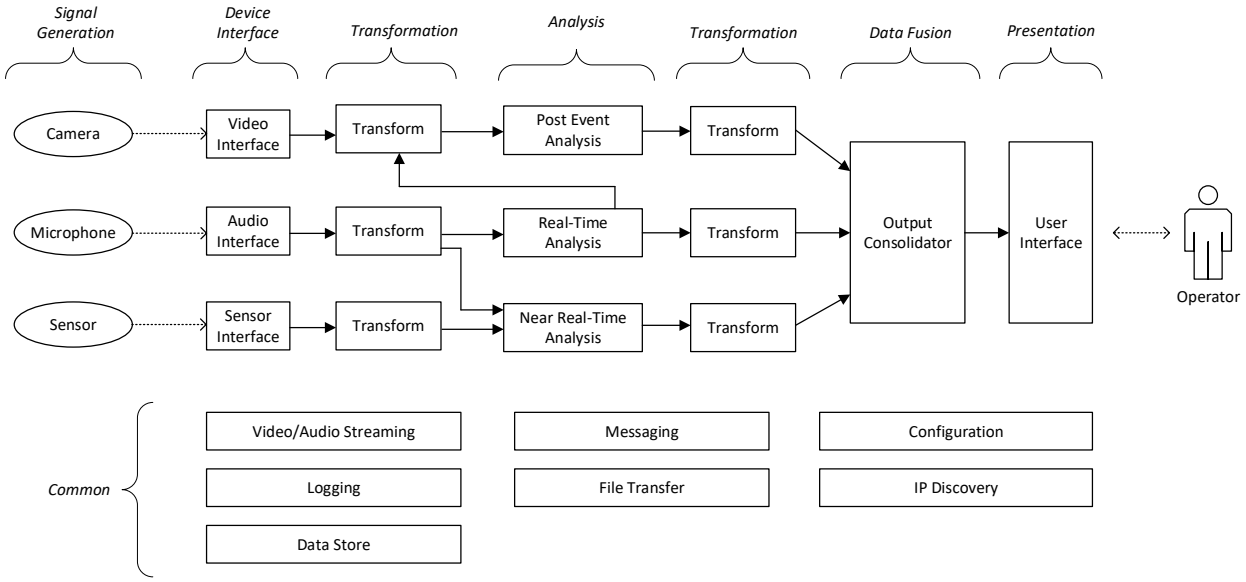


*Figure 4 - On-body domain breakdown*

The domains can be partitioned into key components or services as shown in the diagram presented above.

- ***Signal Generation*** – This domain consists of the built-in camera and microphone of the smartphone in addition to external sensors from the different vendors.

- ***Device Interface*** – The Device Interface includes the Android APIs for the camera and microphone. It also includes custom software written for the USB port to interact via vendor specific protocols.

- ***Transformation*** – The primary responsibility of a transformation domain is to change the form of something. In this architecture, there are two areas of transformation. In the pre-analysis step, the input data, audio, video, external sensor information is changed to be compatible with the downstream analysis software. In the post-analysis processing, the outputs of the different analyses are changed from their native forms to canonical states.

- ***Analysis*** – The Analysis domain interprets one or more input data streams into useful sentiment metrics. The type and range of metrics is dependent on the internal algorithms. For this project, there are three types of analysis, 1) real-time which processes the input stream(s) and generates output without significant delay, 2) near real-time samples the time-series streams over short periods of time and then generates output along the timeline, and 3) post-event analysis begins after the event has been completed and the entire data stream is available for processing.

- ***Data Fusion*** – This domain is responsible for consolidating the canonical data from the different analysis into a single representation. For the PAT project this domain was not developed due to time constraints.

- ***Presentation*** – The Presentation domain is responsible for interacting with the human operators. It allows the user to configure the PAT solution, initiate and terminate a session, and presents the final results.

- ***Common*** – The services that make up the Common domain are the most generic of the architecture and are utilized by the other and more specific domains.

- ***Common*** – *Video/Audio Streaming* – The A/V Streaming service allows multiple entities consume the data feeds. The consumers may be running concurrent in the same compute space and hosted across multiple, heterogeneous compute platforms.

- ***Common*** – *Messaging* – The Messaging service provides asynchronous and synchronous communication between software threads and processes.

- ***Common*** – *Configuration* – This service allows the user to customize the behavior of PAT solution. The user may enable and disable each of the major components and adjust the available dimensions of each component.

- ***Common*** – *Logging* – The Logging service provides insight into the operation of the PAT system by allowing a user to review the state of the internal processing.

- ***Common*** – *File Transfer* – The File Transfer component is responsible for moving files between unshared areas within the system.

- ***Common*** – *IP Discovery* – This service determines the dynamic connection points across the system. The other services within the system are required to only know how to access this service and no other. From IP Discovery service, the connection details to the other services can be gleaned.

- ***Common*** – *Data Store* – The Data Store service provides file storage on the smart phone and allows the user to quickly transfer files to an external computer.

## Architecture

The original architecture for the on-body platform was partitioned into two different computing devices; an Android device and a Single Board Computer (SBC). The Android device provides the user interface for the Operator as well as the data feeds from the built in camera and microphone. Vendor software that is Android compatible is executed on the Android device.

The Single Board Computer component originally provided an Ubuntu (a popular Linux variant) environment for the vendor software systems. The SBC also has I/O capabilities for interfacing with vendor specific sensors. Originally, vendors that have software that were not compatible with Linux and could not be hosted on the Android device would be candidates for using Docker container technology. As development progressed during this Proof of Concept phase, the need to expand the on-body platform emerged. To expand the platform capability and usefulness, a second Single Board Computer

was added to the platform. This SBC provides a Windows 10 environment for the vendor systems. The architecture currently consists of an Android device, a Single Board Computer running Ubuntu (SBC/U), and a Single Board Computer with Windows 10 (SBC/W). A diagram of the Android/SBC-U/SBC-W partitioning is shown below.

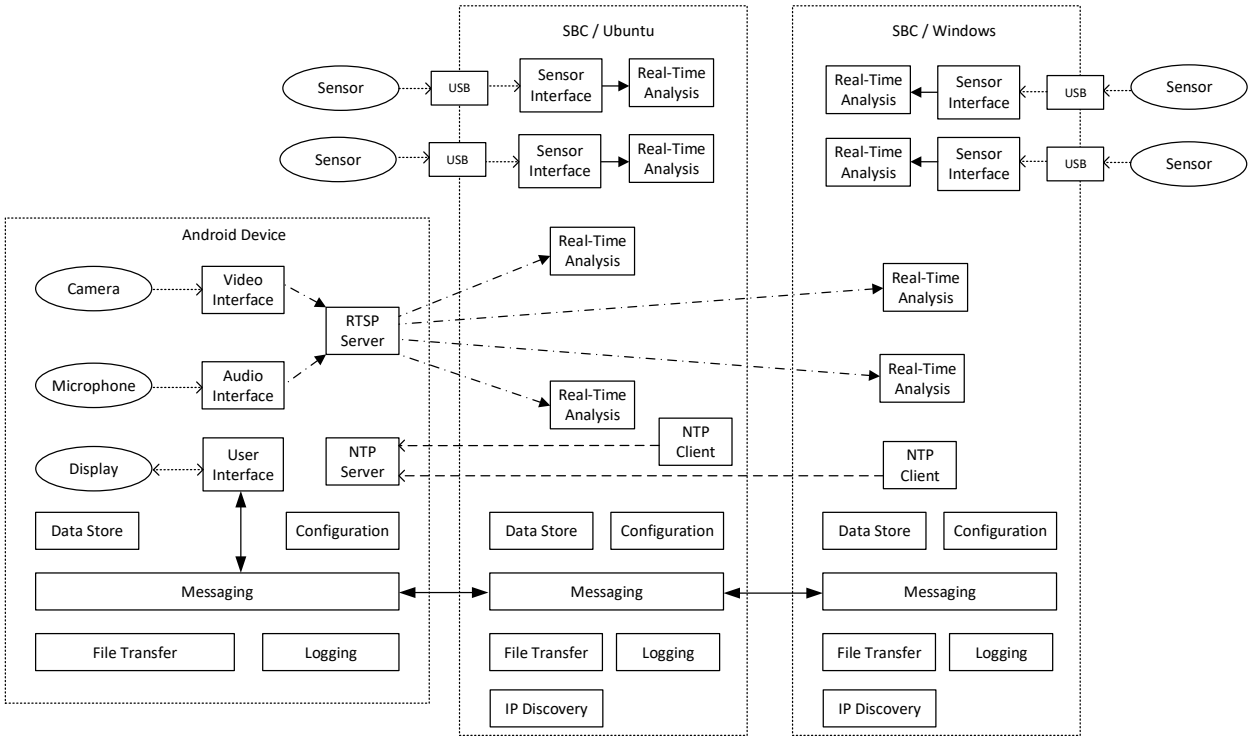


*Figure 5 - Architectural partitioning*

The key points for this diagram include the following:

- The data feeds for the Android camera and microphone will be available on the Android, the SBC/U, and the SBC/W subsystems via the A/V streaming service.
- The data feeds for the Android camera and microphone will be persisted on the Android device.
- The data feeds generated on an SBC will be persisted on that SBC. The files that are created there will be uploaded to the Android device via the File Transfer service for export to the off-body system.
- The Messaging service will be using the Zero MQ framework. This service provides inter-service communications.
- The Android device will host the Real Time Streaming Protocol (RSTP) server. Multiple RTSP clients may be hosted on Android, SBC/U, and SBC/W
- The Android will host the Network Time Protocol (NTP) server. A NTP client will be hosted on the SBC/U and a NTP client will be hosted on the SBC/W.

- The IP Discovery service is hosted on the SBC/U platform and will determine the IP of the Android device which is dynamically assigned on power up. The other services will request that IP from the IP Discovery service. Once known, the other services may communicate with each other.

Also, it should be noted that the output of the real-time analysis services can be 1) produced throughout during the session on a time-based structure or an event-based format or 2) at the termination of the session. The architecture, especially the Messaging service, will be able to detect and publish the outputs regardless of the timing.

## Hardware Architecture

The on-body subsystem consists of three components that are networked together. The Android device is the Samsung Note20 Ultra smartphone [2]. This device provides the user interface along as host for Android-based software.

The second component is the BioDigitalPC 12, a single board computer that has been used in at least one other SOCOM project [3]. This device runs the Ubuntu Server 20.04 operating system. The purpose of this component is to host software that is not Android or Windows compatible and/or requires access to external sensors. The BioDigitalPC uses the M7 interface board that provides I/O connectors [4].

The third component is another BioDigitalPC 12 SBC that is running the Windows 10 operating system. This computer executes software that is only compatible with Windows.

The two SBCs each feature Google's M.2 Accelerator with Dual Edge TPU device. It connects to the BioDigitalPC computer via a connector on the M7 interface board. [5] This system accelerates TensorFlow Lite models in a power efficient fashion. Each of the two TPU processors execute 4 trillion operations per second (TOPS).

The two SBCs are stacked in a custom enclosure designed and constructed by The Foundry @ SOFWERX. This item is the only custom piece of the computing platform. A silent 9 dB fan is housed in the enclosure and provides air flow to the electronics. An image of the enclosure with the electronics is shown in Appendix D.

One of the architectural guidelines on this project is to avoid wireless signals during deployment. Another guideline is to minimize hardware, especially for the on-body platform. With these considerations, the three components of the on-body platform are networked via hard wired connections in different and unique ways. The only external port on the Samsung device is a USB-C female connector. The Android to SBC/U connection is a USB cable from the Samsung phone to the USB 3.0 USB port on the SBC/U device. This connection provides an IP over USB connection between the two devices via the USB tethering feature of the phone.

The SBC/U is connected to the Single Board Computer/Windows (SBC/W) via an Ethernet cross over cable. This type of connection was selected in order to maximize the number of USB ports available for the vendor solutions. This connection provides an IP connection between the two devices.

A diagram with the connectors of the three devices is shown below:

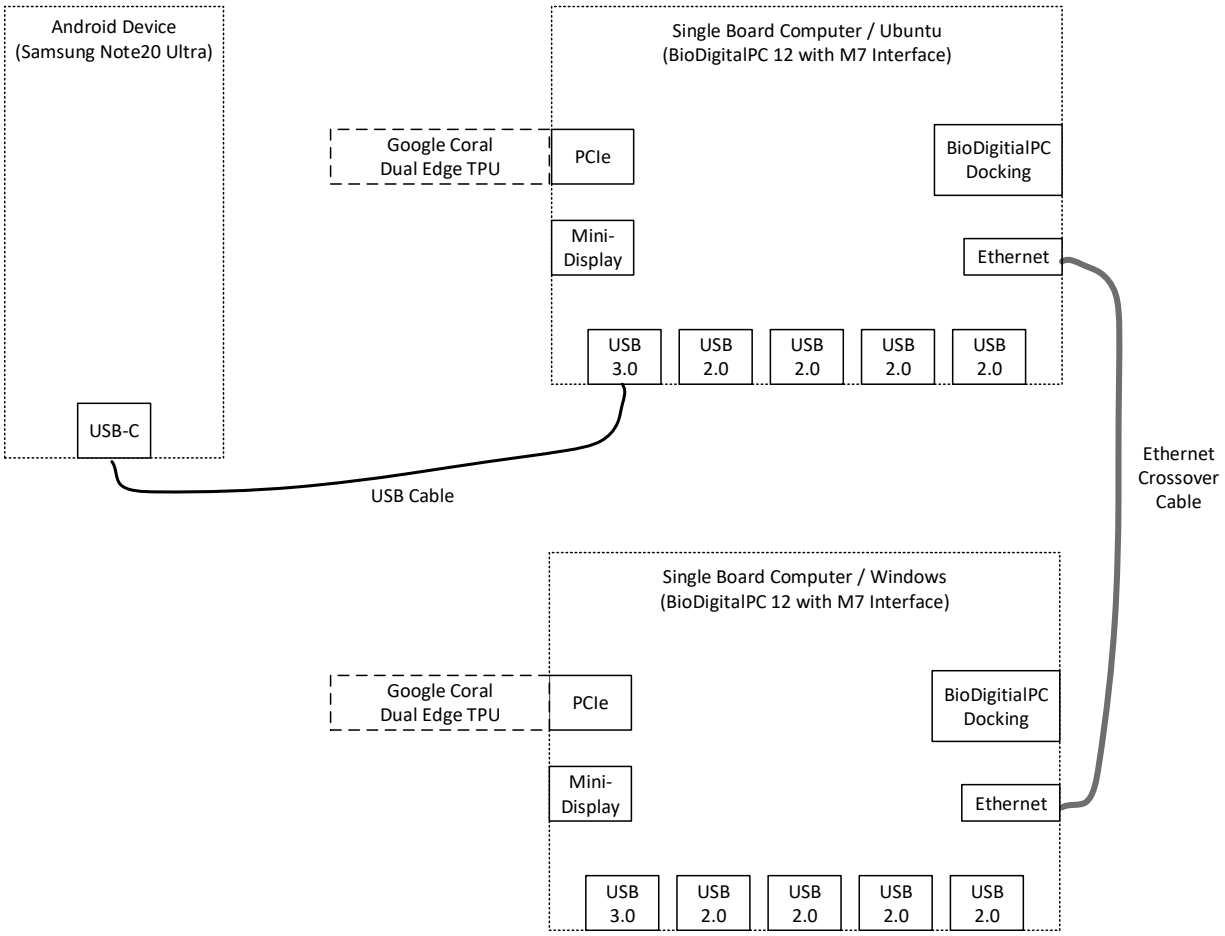


*Figure 6 – On-body Connector Layout*

**NOTE:** The on-body platform will NOT have access to external systems or resources during the event. Connection to the internet is not possible and shall not be assumed by the real-time analysis software or any of its dependencies, libraries, or packages.

Two major factors drove the design of the network topology. First, no network router or switch was to be used. Second, the Android OS on the Samsung was NOT to be modified for privileged access (i.e., rooting). After significant online research and exhaustive hands-on trial and error, a near-optimal solution using two networks was configured. The default network is the one the Android creates when the USB tethering feature is enabled. A static IP of 192.168.42.129 is assigned to the Android device and a DHCP IP is assigned to the USB on the SBC/U computer. The second network is created between the Ethernet port (192.168.2.100) on the SBC/U and the Ethernet port on the SBC/W (192.168.2.150). These IPs are static and the 192.168.2.100 is configured as the software gateway for the 192.168.2.150. A diagram of this configuration is shown below.

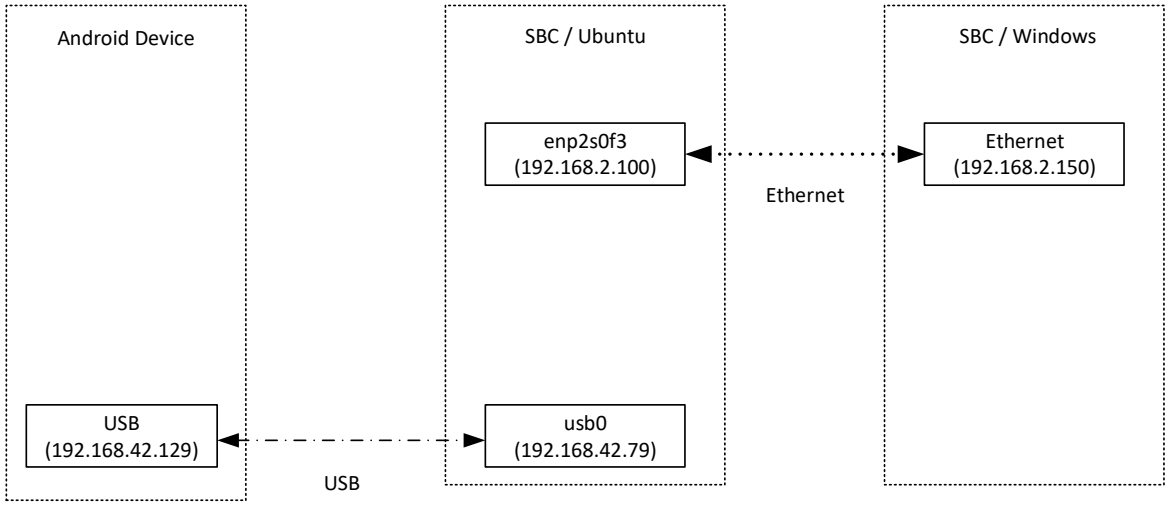


*Figure 7- IP Assignment*

The major constraint in this effort is the software gateway on the Android device. It is configured not to allow forwarding past the *usb0* interface (192.168.42.79). The rules of the software gateway cannot be altered without rooting the phone. However, IP forwarding can be configured from the *enp2s0f3* interface (192.168.2.100) to the *usb0* interface. The end result is that software on the SBC/U and SBC/W can access the Android device (192.168.42.129) while the Android device is not permitted access beyond the *usb0* interface. The IP visibility is shown in the following table.

| | ***TARGET IP*** | | | | |
|---|---|---|---|---|---|
| ***SOURCE IP*** | | **192.168.42.129** | **192.168.42.79** | **192.168.2.100** | **192.168.2.150** |
| | **192.168.42.129** | YES | YES | NO | NO |
| | **192.168.42.79** | YES | YES | NO | NO |
| | **192.168.2.100** | YES | YES | YES | YES |
| | **192.168.2.150** | YES | YES | YES | YES |

*Table 1 - IP Visibility*

Although the connection topology is asymmetrical, it does provide an IP layer between the different environments. This configuration allows messaging, streaming, and other IP-based services to encapsulate the heterogeneous environments from the vendor software solutions. While the SBC/U is the center of the "physical" model, the Android device is the hub of the "logical" model. Since the SBC/U and the SBC/W devices have access to the Android device, software-based servers/services are hosted in the Android environment.

## Software Architecture

This section describes the different services that comprise the software architecture of the on-body subsystem.

## Audio/Video Streaming

The Real Time Streaming Protocol (RTSP) is a very common and proven communications protocol used to control streaming media servers [6]. While there are several options for a RTSP server available, the *pedroSG94* open source project was selected for this project [7]. This server allows multiple, concurrent clients to receive the video and audio streams and can be embedded into the PAT Android application. The consumers of the streaming data can be on the Android device, the Ubuntu SBC, and the Windows SBC. A diagram of the streaming service is shown below:

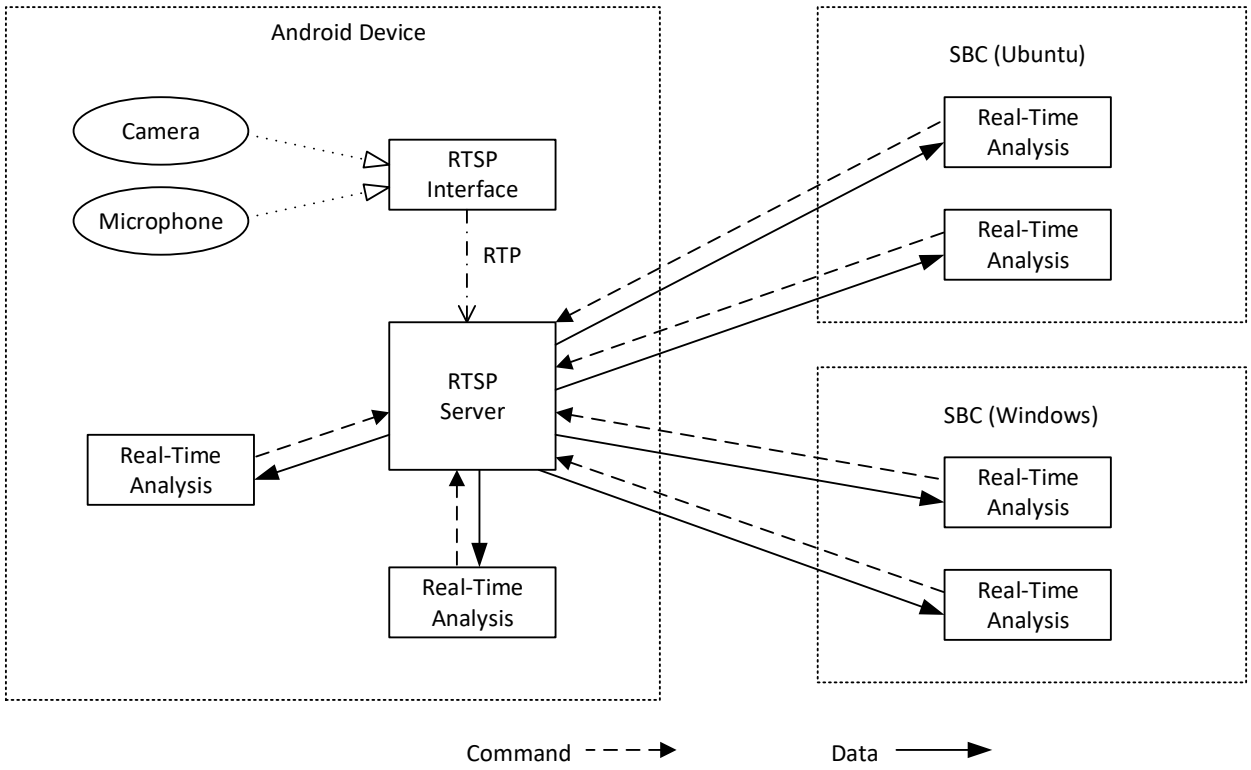


*Figure 8 - RTSP streaming service*

Details of the RTSP Options and Describe commands are shown in Appendix B.

## Messaging Service

ZeroMQ is a high-performance asynchronous messaging library, aimed at use in distributed or concurrent applications. It provides a message queue, but unlike message-oriented middleware, a ZeroMQ system can run without a dedicated message broker [8]. The layout of the messaging architecture is shown in the following diagram.

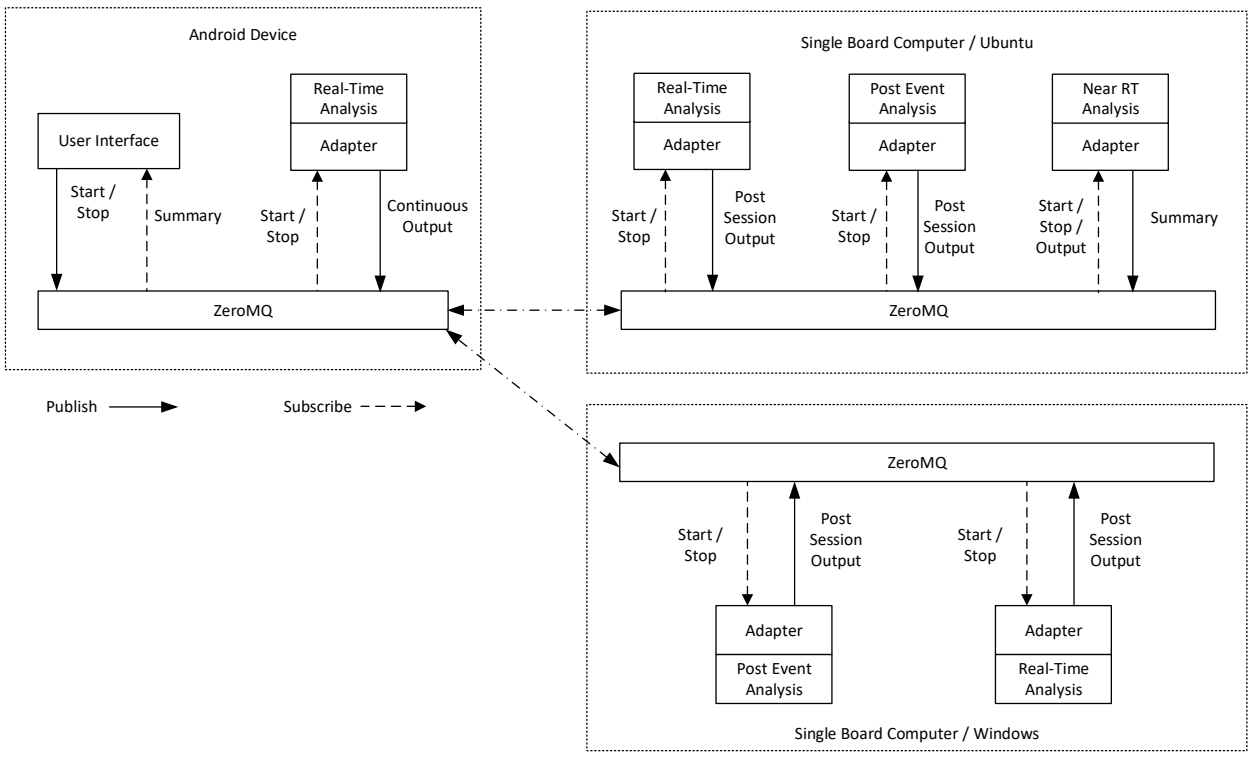


*Figure 9- On-body Messaging Service*

Examples of the ZMQ messages are presented in Appendix C.

## IP Discovery

This service is responsible for determining the IP to use for communicating with the Android device. The IP value is assigned by the Android device and cannot be changed. The IP is presented to the Ubuntu SBC when the USB-C cable is connected.

The IP Discovery service consists of two components; a server that can determine the Android USB IP and client code that sends a ZMQ message to the server requesting that IP value. Once the IP is retrieved, the client then subscribes to events from the Android PAT application. The event sequence for this interaction is shown in the diagram below:

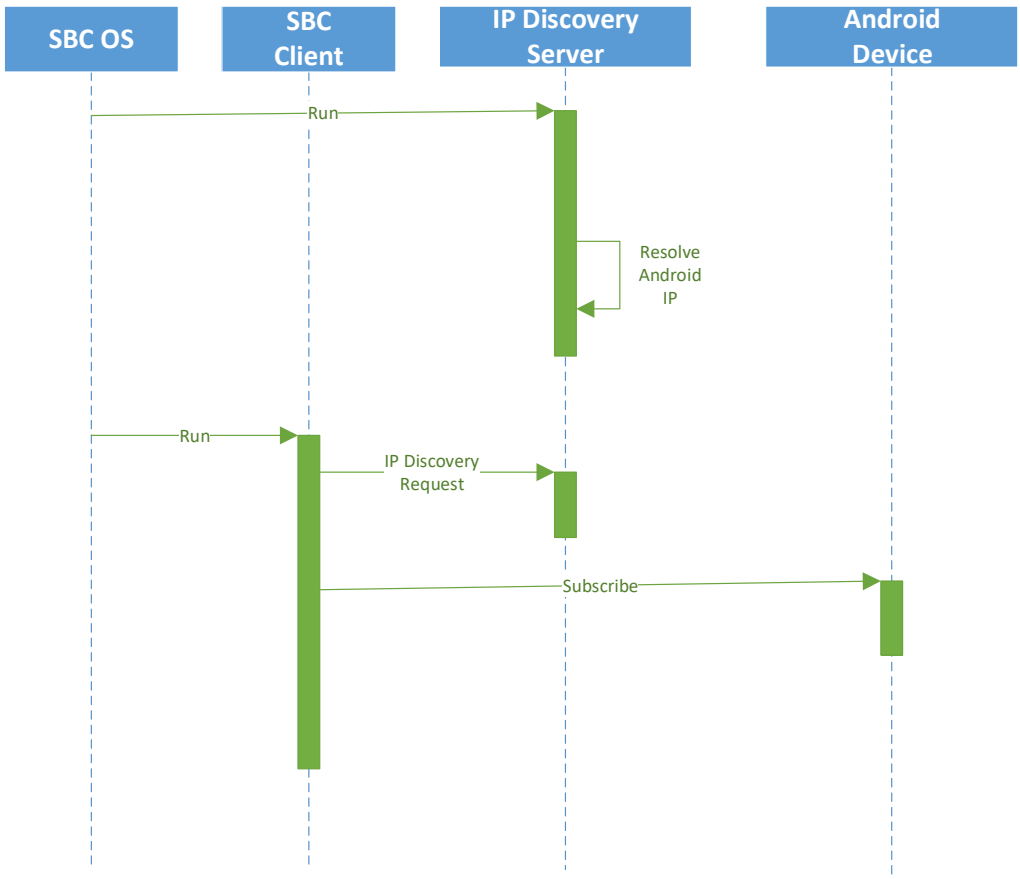


*Figure 10 - Event sequence for IP discovery*

## File Transfer

The File Transfer service encapsulates the details of moving a file from the Android device to the Single Board Computers and from the SBCs to the Android device. This service is based on the *File Transfer Model 2* pattern provided by ZQM [9]. A diagram of the event sequence is shown below:

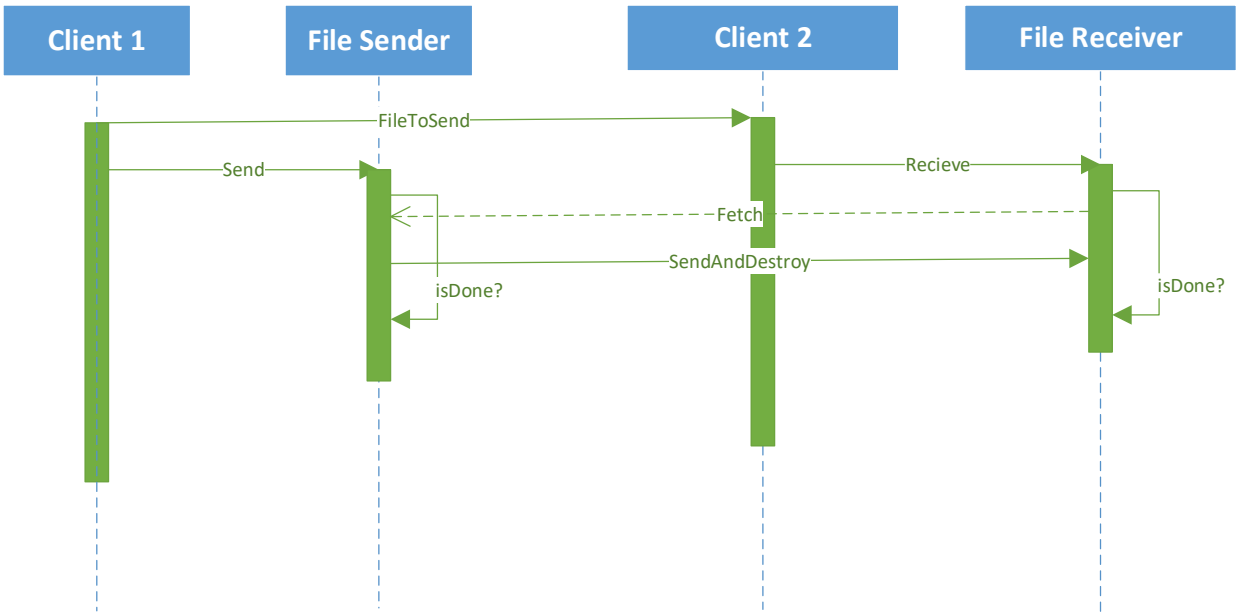


*Figure 11 - Event sequence for file transfer*

## Clock Synchronization

The objective of the clock synchronization service for the on-body platform is to ensure the timestamps among the three computing devices are consistent. Since the Operator may not be in an environment with access to network-based timeservers, accuracy of the timestamps to an external, Internet-based time service is less critical. Due to the PAT network topology, a Network Time Protocol (NTP) server is hosted on the Android device while the two SBCs each host a NTP client. This approach is shown in the diagram below:

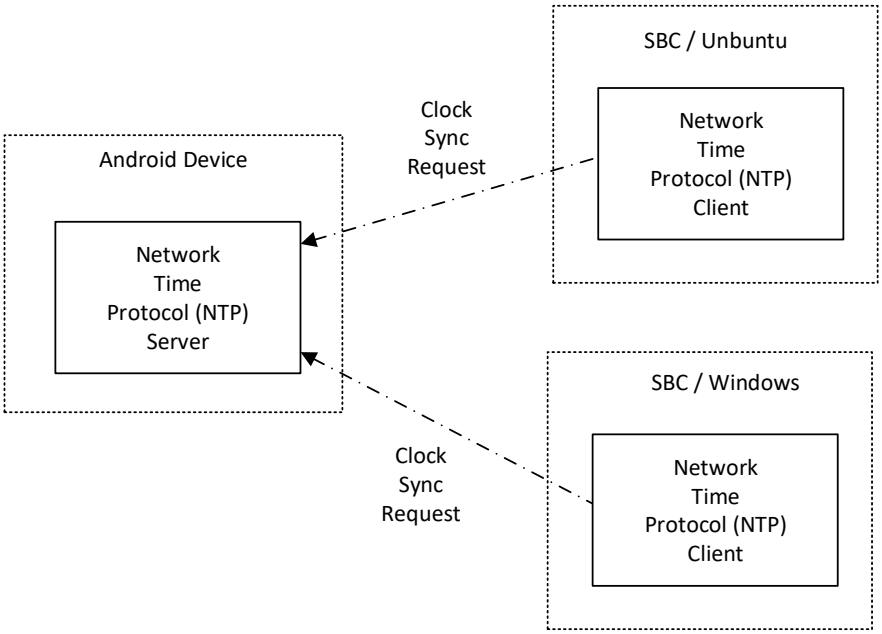


*Figure 12 - Clock Synchronization*

## Analysis

The Analysis service covers two major areas of interest. The first is sentiment analysis which attempts to measure the state of emotion of the observed entity. The second area is physiological analysis that contributes to the sentiment processing.

This project featured five sentiment measurements by four different providers. The approaches include the following:

- **Florida Institute of Technology (FIT)** - This software outputs one of four emotion states based on the WAV file of the recorded session.
- **Intelligent Automation Inc (IAI) Video** – This software outputs one of six emotion states based on video streaming input.
- **Intelligent Automation Inc (IAI) Audio** - This software outputs one of six emotion states based on audio streaming input.
- **smileML** – This software outputs one of ten emotion states based on the video data of the MP4 file of the recorded session.
- **SOFWERX** – This approach converted the WAV file to text and then applied a text sentiment algorithm to output a percent positive and percent negative sentiment.

This project also included physiological measurements from three different vendors:

- **Caaresys** – This vendor's approach uses proprietary hardware and software to determine respiration rate and heart rate.

- **Vivonics** – This approached incorporates off-the-shelf hardware with custom algorithms to provide respiration rate and heart rate.

- **Zansors** – This approach was based on their commercial product to measure respiration rate.

The deployment of the different components of the Analysis service is shown in the diagram below. This view also shows the flexibility of the heterogeneous computing platform.

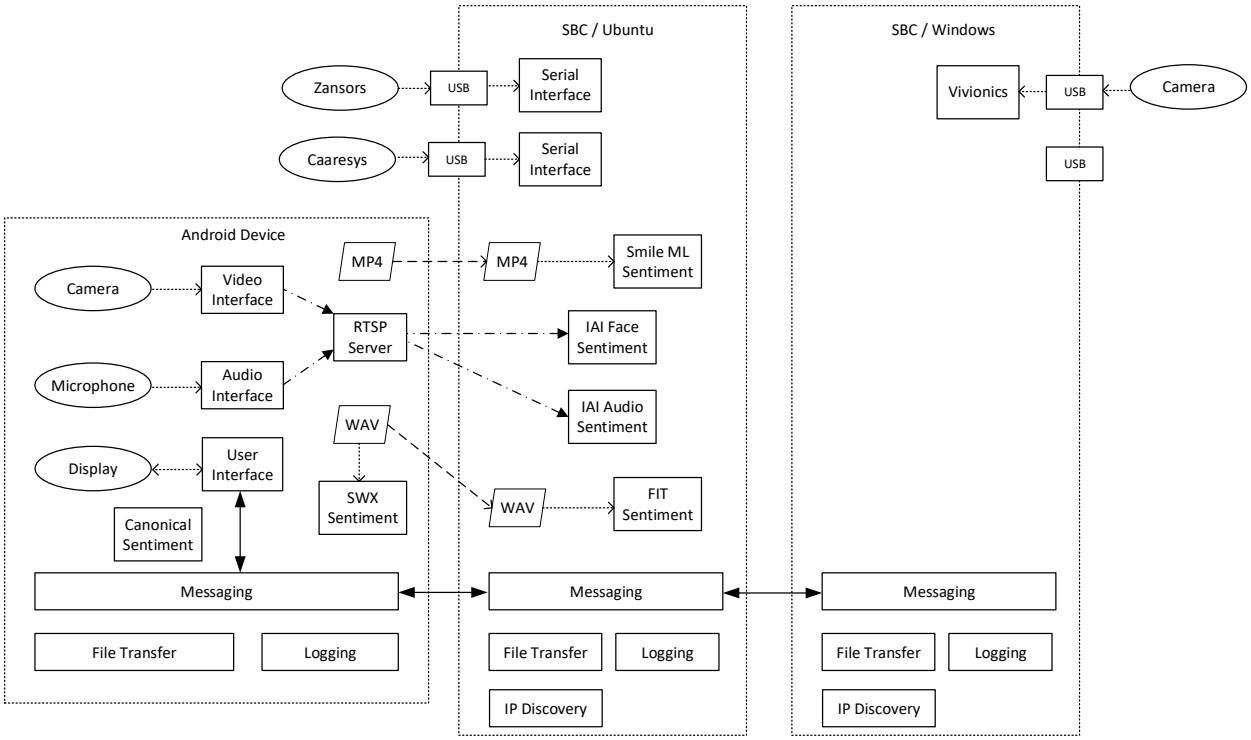


*Figure 13 - Deployment architecture*

## Native to Canonical Sentiment

This service is responsible for converting the native outputs of each of the five sentiment analysis approaches into a canonical representation. The canonical values include "Very Positive", "Positive", "Neutral", "Negative", and "Very Negative". The mapping from the native values from each approach to these canonical values is shown below:

| CANONICAL | NATIVE | | | | |
|---|---|---|---|---|---|
| | *IAI - Face* | *IAI - Audio* | *FIT* | *Smile ML* | *SOFWERX* |
| *Very Positive* | | | | | Positive > 80 |
| | | | | | |
| *Positive* | HAPPY | HAPPY | HAPPY | HAPPINESS | Positive > 60 |
| | | | | | |
| *Neutral* | NEUTRAL | NEUTRAL | NEUTRAL | NEUTRAL | Positive <= 60 |
| | | | | SURPRISE | Negative <= 60 |
| | | | | NF | |
| | | | | UNKNOWN | |
| | | | | | |
| *Negative* | SAD | SAD | SAD | SADNESS | Negative > 60 |
| | FEAR | FEAR | | FEAR | |
| | | | | | |
| *Very Negative* | ANGER | ANGER | ANGRY | ANGER | Negative > 80 |
| | DISGUST | DISGUST | | DISGUST | |
| | | | | CONTEMPT | |

*Table 2 - Native to Canonical Mapping*

## Artifacts

After the recording session is stopped by the user and all of the processing is completed, there are a set of files regarding the session that is available on the Android device. The user may access these files by connecting the device to a personal computer via the USB-C port. The files available from an actual session are described below. The names of the files include the timestamp of the start of the session or a unique session id.

- **a_2021-05-05T15-06-44Z-audio-fit.json** – The native sentiment measurements by the Florida Institute of Technology (FIT) software.
- **a_2021-05-05T15-06-44Z-speech.json** – The results of the speech-to-text conversion by the SOFWERX software.

- **a_2021-05-05T15-06-44Z-audio.json** – The results of the text sentiment analysis by the SOFWERX software.

- **a_2021-05-05T15-06-44Z-audio.pcm** – This is the native audio file generated by the Android device.

- **a_2021-05-05T15-06-44Z-audio.wav** – The .pcm file is converted to the WAV format so it can be processed by the FIT and SOFWERX software.

- **a_2021-05-05T15-06-44Z-sentiment.json** – This file contains the canonical sentiment results from the five algorithms.

- **audio_embeddings_237c9856-7e50-4254-bffa-99eae5c8223e.jsonl –** This file has the results of the audio-based sentiments from IAI.

- **face_embeddings_237c9856-7e50-4254-bffa-99eae5c8223e.jsonl** - This file has the results of the audio-based sentiments from IAI.

- **av_2021-05-05T15-06-45Z-video-smileml.json** – This file contains the sentiment output of the smileML algorithms.

- **av_2021-05-05T15-06-45Z-video.**mp4 – This is the native video file generated by the Android device.

- **zansor-23-050521-111057.txt** – This file contains the respiratory rate data recorded by the Zansors device.

## Who We Are

SOFWERX is a non-profit entity that accelerates evolution of the Warfighter through technology discovery, engagement, development, and transition. SOFWERX was created under a Partnership Intermediary Agreement, established in September of 2015, between DEFENSEWERX and the United States Special Operations Command (USSOCOM).

Jim Ladd is a Senior Software Architect and IT Manager at SOFWERX where he leads the software engineering, web development, and system administration teams. Jim has been developing software solutions for over 35 years. Before joining SOFWERX in 2019, Jim was CEO and Principal Consultant at Wazee Group, a niche consulting company, for 20 years. His LinkedIn profile link is https://www.linkedin.com/in/jim-ladd/

Hansel Rios is a software developer at SOFWERX where he focuses on custom software solutions. Before joining SOFWERX, Hansel was a Software Engineer at Whitridge Associates, TRU Simulation & Training, and Textron Aviation. Hansel co-owns a patent related to alert and alarm systems for vehicle occupants. His LinkedIn profile link is https://www.linkedin.com/in/hansel-rios-b9a53b69/

Janson Young is currently a Software Manager and Senior Software Engineer with WinTec Arrowmaker. While at SOFWERX, Jason developed several critical projects including the Android Low Latency Camera Platform (ALLCaP) and the Physiological Analysis Tool (PAT). His LinkedIn profile link is https://www.linkedin.com/in/jason-young-7144064a/

## References


[1] S. S. &. S. J. Mellor, "The Shlaer-Mellor Method," Project Technology, Inc., 1996.

[2] "Samsung Note20 Ultra Specifications," [Online]. Available: https://www.samsung.com/us/smartphones/galaxy-note20-5g/specs/.

[3] B. 1. Specifications. [Online]. Available: https://addc.net/products/biodigitalpc-12/.

[4] "R8K 3DE Specifications," [Online]. Available: https://addc.net/products/r8k-3de/.

[5] Google, "M.2 Accerlator with Dual Edge TPU datasheet," [Online]. Available: https://coral.ai/docs/m2-dual-edgetpu/datasheet/.

[6] H. Schulzrinne, "RTSP prime," Columbia University, 1996.

[7] "pedroSG94/RTSP-Server," [Online]. Available: https://github.com/pedroSG94/RTSP-Server.

[8] "ZeroMQ," [Online]. Available: https://zeromq.org/.

[9] ZMQ, "Chapter 7 - Advanced Architecture using ZeroMQ," [Online]. Available: https://zguide.zeromq.org/docs/chapter7/. [Accessed 01 04 2021].

## Appendix A – Android Screen Shots

This section presents the screen shots for the Operator role that were generated during the Physiological Analysis Tool Tech Sprint event held in December, 2019.

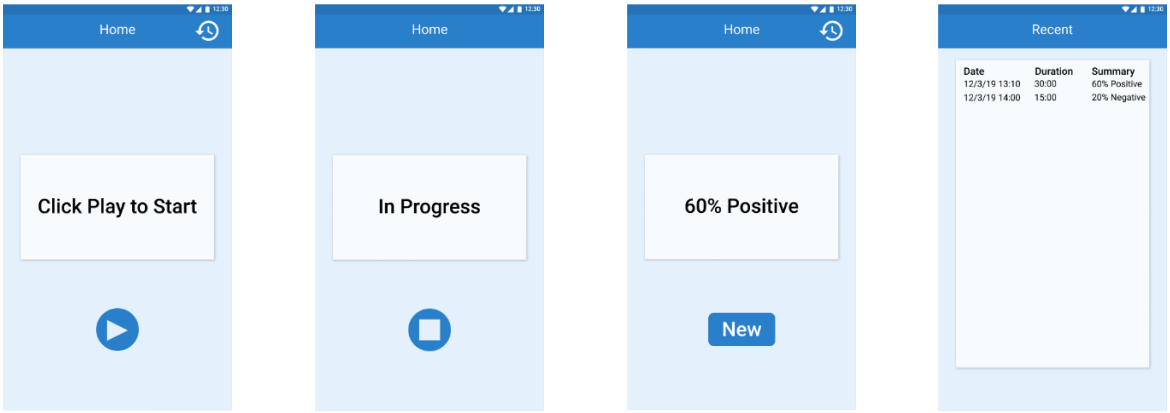


*Figure 14 - Conceptual User Interface*

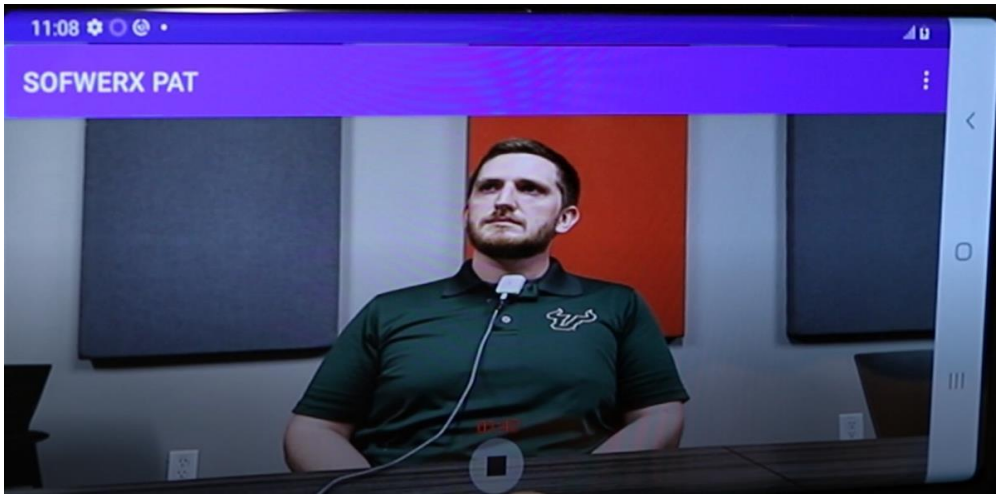


*Figure 15 - User interface with video preview*

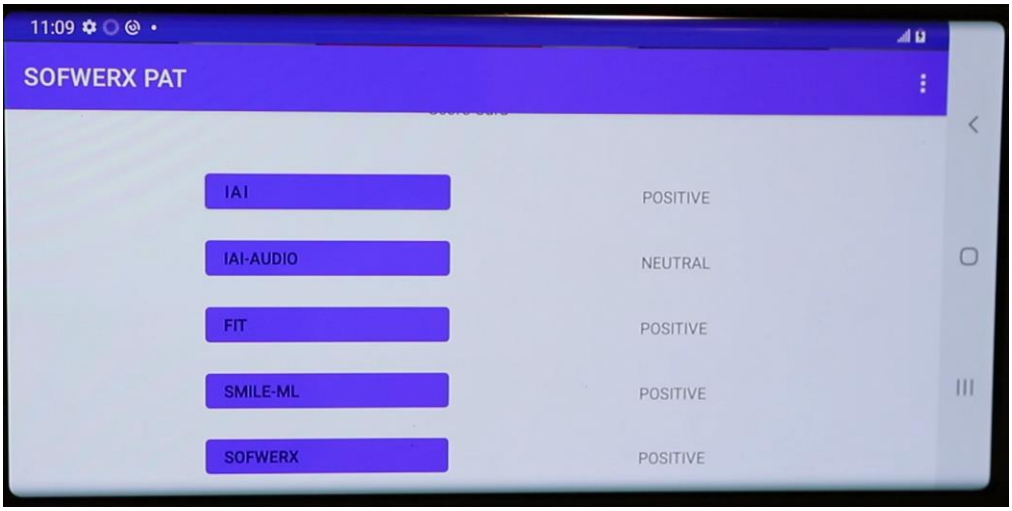


*Figure 16 - Display of canonical sentiments*

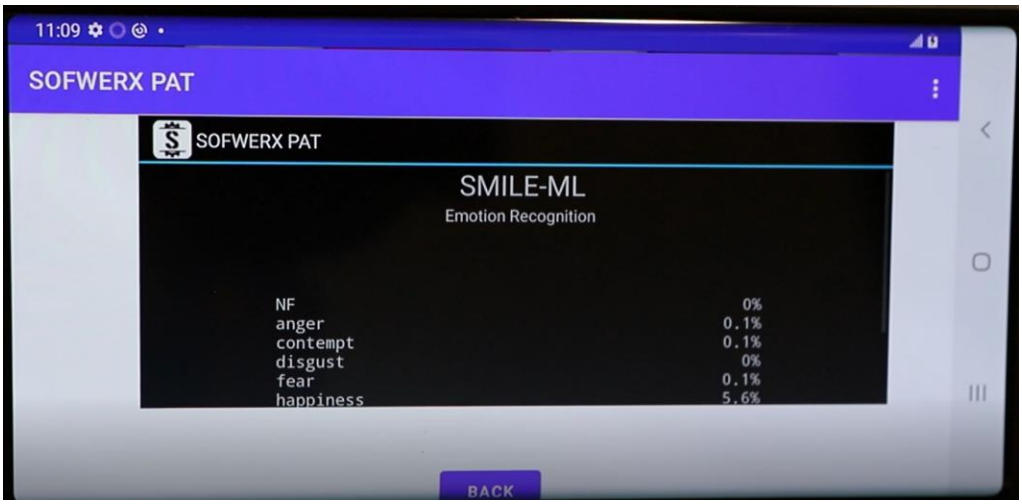


*Figure 17 - Display of smileML native values*

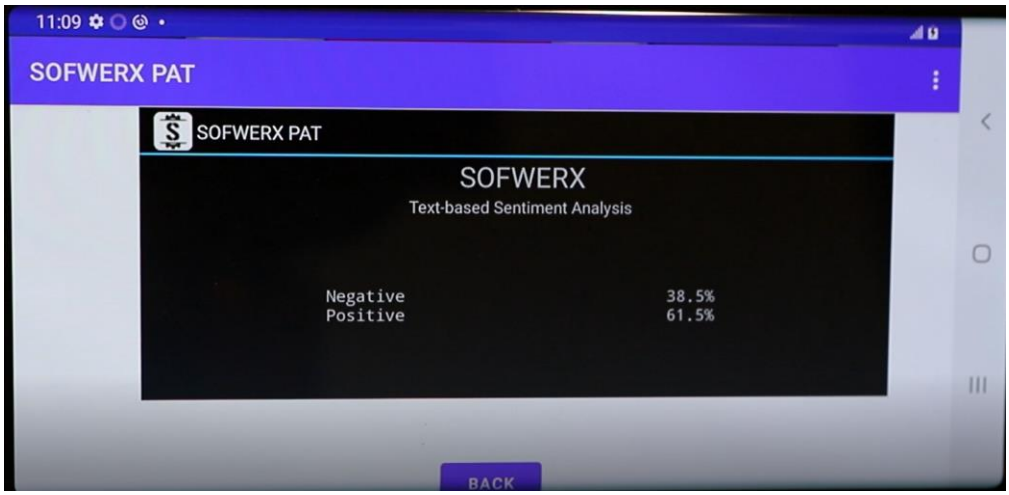


*Figure 18 - Display of SOFWERX native values*

## Appendix B – RTSP OPTIONS and DESCRIBE

This appendix shows the responses of the RTSP OPTIONS and DESCRIBE commands from the Samsung Note20 device.

```
Created new TCP socket 3 for connection
Connecting to 192.168.42.129, port 8554 on socket 3...
...remote connection opened
Sending request: OPTIONS rtsp://192.168.42.129:8554 RTSP/1.0
CSeq: 2
User-Agent: openRTSP (LIVE555 Streaming Media v2020.01.19)


Received 98 new bytes of response data.
Received a complete OPTIONS response:
RTSP/1.0 200 OK
Server: pedroSG94 Server
Cseq: 2
Public: DESCRIBE,SETUP,TEARDOWN,PLAY,PAUSE


Sending request: DESCRIBE rtsp://192.168.42.129:8554 RTSP/1.0
CSeq: 3
User-Agent: openRTSP (LIVE555 Streaming Media v2020.01.19)
Accept: application/sdp


Received 583 new bytes of response data.
Received a complete DESCRIBE response:
RTSP/1.0 200 OK
Server: pedroSG94 Server
Cseq: 3
Content-Length: 448
Content-Base: 0.0.0.0:8554/
Content-Type: application/sdp

v=0
o=- 0 0 IN IP4 0.0.0.0
s=Unnamed
i=N/A
c=IN IP4 192.168.42.35
t=0 0
a=recvonly
m=audio 0 RTP/AVP 96
a=rtpmap:96 MPEG4-GENERIC/48000/2
a=fmtp:96 streamtype=5; profile-level-id=15; mode=AAC-hbr; config=1190;
SizeLength=13; IndexLength=3; IndexDeltaLength=3;
a=control:trackID=0
m=video 0 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 packetization-mode=1;sprop-parameter-
sets=Z0KAFtoC0PaUgoEBA2hQmoA=,aM4G8g==;
a=control:trackID=1
```

```
Opened URL "rtsp://192.168.42.129:8554", returning a SDP description:
v=0
o=- 0 0 IN IP4 0.0.0.0
s=Unnamed
i=N/A
c=IN IP4 192.168.42.35
t=0 0
a=recvonly
m=audio 0 RTP/AVP 96
a=rtpmap:96 MPEG4-GENERIC/48000/2
a=fmtp:96 streamtype=5; profile-level-id=15; mode=AAC-hbr; config=1190;
SizeLength=13; IndexLength=3; IndexDeltaLength=3;
a=control:trackID=0
m=video 0 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 packetization-mode=1;sprop-parameter-
sets=Z0KAFtoC0PaUgoEBA2hQmoA=,aM4G8g==;
a=control:trackID=1


Created receiver for "audio/MPEG4-GENERIC" subsession (client ports 36034-
36035)
Created receiver for "video/H264" subsession (client ports 52804-52805)
Sending request: SETUP 0.0.0.0:8554/trackID=0 RTSP/1.0
CSeq: 4
User-Agent: openRTSP (LIVE555 Streaming Media v2020.01.19)
Transport: RTP/AVP;unicast;client_port=36034-36035


Received 238 new bytes of response data.
Received a complete SETUP response:
RTSP/1.0 200 OK
Server: pedroSG94 Server
Cseq: 4
Content-Length: 0
Transport: RTP/AVP/UDP;unicast;destination=192.168.42.35;client_port=8000-
8001;server_port=39000-35968;mode=play
Session: 1185d20035702ca
Cache-Control: no-cache


Setup "audio/MPEG4-GENERIC" subsession (client ports 36034-36035)
Sending request: SETUP 0.0.0.0:8554/trackID=1 RTSP/1.0
CSeq: 5
User-Agent: openRTSP (LIVE555 Streaming Media v2020.01.19)
Transport: RTP/AVP;unicast;client_port=52804-52805
Session: 1185d20035702ca


Received 238 new bytes of response data.
Received a complete SETUP response:
```

```
RTSP/1.0 200 OK
Server: pedroSG94 Server
Cseq: 5
Content-Length: 0
Transport: RTP/AVP/UDP;unicast;destination=192.168.42.35;client_port=8000-
8001;server_port=39000-35968;mode=play
Session: 1185d20035702ca
Cache-Control: no-cache


Setup "video/H264" subsession (client ports 52804-52805)
Created output file: "audio-MPEG4-GENERIC-1"
Created output file: "video-H264-2"
Sending request: PLAY 0.0.0.0:8554/ RTSP/1.0
CSeq: 6
User-Agent: openRTSP (LIVE555 Streaming Media v2020.01.19)
Session: 1185d20035702ca
Range: npt=0.000-


Received 135 new bytes of response data.
Received a complete PLAY response:
RTSP/1.0 200 OK
Server: pedroSG94 Server
Cseq: 6
Content-Length: 0
RTP-Info: url=rtsp://0.0.0.0:8554/
Session: 1185d20035702ca
```

# Appendix C – ZMQ Message Definitions

This appendix describes the ZMQ messages for the "Start Session" and "End Session" events. The messages are JSON-based that are published on the "PAT_EVENT" topic.

The JSON data for a sample Start Session message is shown below. The *feedUrl* attribute is the URL for the streaming feed. While the current software design does not feature concurrent sessions, the *sessionUid* attribute is a unique identifier that correlates a specific Start Session message with the corresponding End Session message.

```
{ "patEvent": {
    "sessionContext": {
      "feedUrl":"rtsp://192.168.42.129:8554/ATAK/PATVideo"},
      "sessionUid": "2455dfd5-055c-4545-bf0a-c022ce6b8abc",
      "time": "2021-02-25T18:05:15Z",
      "eventType": "Start"
    }
}
```

The JSON data for a sample Start Session message is shown below.

```
{ "patEvent": {
    "sessionUid": "2455dfd5-055c-4545-bf0a-c022ce6b8abc",
    "time": "2021-02-25T19:35:17Z",
    "eventType": "Stop"
    }
}
```

*// Published to the services that the recording session has started*
*Published Topic: PAT_EVENT*
*Published Data:*
*{"patEvent": {*
*"sessionContext": {"feedUrl": "rtsp://192.168.1.246:8554/ATAK/PATVideo"},*
*"sessionUid": "2455dfd5-055c-4545-bf0a-c022ce6b8abc",*
*"time": "2021-02-25T18:05:15Z",*
*"eventType": "Start"*
*}}*

*// Published to the services that the recording session is completed*
*Published Topic: PAT_EVENT*
*Published Data:*
*{"patEvent": {*
*"sessionUid": "2455dfd5-055c-4545-bf0a-c022ce6b8abc",*
*"time": "2021-02-25T18:05:17Z",*
*"eventType": "Stop"*
*}}*

*// Published to the Time Sync client*
*Published Topic: PAT_EVENT*
*Published Data:*

{"patEvent": {
"sessionUid": "2455dfd5-055c-4545-bf0a-c022ce6b8abc",
"time": "2021-02-25T18:05:17Z",
"eventType": "TimeSync"
}}

*// Publish from a service to denote that it's processing is finished*
*Published Topic: PAT_STATUS*
*Published Data:*
{"patEvent": {
"eventType": "Processing Done"
"time": "2021-02-25T18:05:17Z",
"subsystem": "IAI"
"fileName": "face_embeddings.jsonl"
}}

*// Publish to a specific subscriber to send the requested file*
*Published Topic: PAT_EVENT*
*Published Data:*
{"patEvent": {
"eventType": "FileRequest"
"time": "2021-02-25T18:05:17Z",
"subsystem": "IAI"
"fileName": "face_embeddings.jsonl"
}}

*// Publish for the FileTransfer subscriber to begin receiving a file*
*Published Topic: PAT_EVENT*
*Published Data:*
{"patEvent": {
"eventType": "FileReadyToSend"
"time": "2021-02-25T18:05:17Z",
"fileName": "av_2021-04-05T195236Z-video.mp4"
}}

*// Publish to all subscribers that a file is ready for processing*
*Published Topic: PAT_EVENT*
*Published Data:*
{"patEvent": {
"sessionUid": "2455dfd5-055c-4545-bf0a-c022ce6b8abc",
"time": "2021-02-25T18:05:17Z",
"eventType": "FileAvailable"
"fileName": "av_2021-04-05T195236Z-video.mp4"
}}

## Appendix D – On-body Enclosure

This section describes the custom enclosure for the on-body system. An image of the dual SBC architecture is shown below:

*Figure 19 - Image of custom enclosure*