The Centauri Corporation

COPERS Integration

Integrating JIFX Experiments into COPERS

Leininger, John

2-24-2021

Integrating JIFX Experiments into COPERS

Table of Contents

Integrating JIFX Experiments into COPERS

Introduction

There are several ways that data has been integrated into COPERS as a part of JIFX experiments.

The primary way is to have a COPERS user enter data directly into the COPERS operational picture.

Machine to machine interfaces are available to integrate Cursor on Target data, Advisories / Notifications, and Video imagery into COPERS.

Integration Technology and Documents

This section describes technologies that have been used in conjunction with JIFX experiments. These tools are provided as examples for experimenters to use as they develop their tools to integrate into the JIFX environment.

Cursor on Target (CoT) Simulator

The CoT simulator provides a way to generate simulated CoT traffic that represent a specified number of UAS flying in circles around a given latitude and longitude.

This simulator has been extended from a simulator that I found on the internet several years ago. It is written in Python.

By default, the simulator is configured to produce CoT traffic and send it via UDP. The simulator can send traffic to TCP. It is recommended that the simulator produces UDP traffic because when it produces TCP traffic it makes and breaks a TCP socket connection for each message. The ./COP-UDP-TCP Bridge/UDP_Server.py script receives local UDP traffic and sends it to a CoT/TAK Server using a single long running TCP connection.

To run the CoT simulator:

- Unzip the JIFX-tools zip file.
- Navigate to the cotsim folder.
- Edit the ./cotsim/CoTSim-QTF.py file
    - Set the number of simulated UAS desired by changing the NUM_USERS variable (default 3)
    - Change the CENTER_LAT_DEGREES & CENTER_LON_DEGREES to set the desired center point of the simulated traffic.
    - Set the minimum and maximum radius of the simulated traffic by editing the CIRCLE_RADIUS_RANGE_METERS variable.
    - Set ATAK_IP to the desired TAK/COT server endpoint
    - Set ATAK_PORT to the desired TAK/COT server port
    - Set the ATAK_PROTO variable to UDP. TCP is also an option, but it is not recommended.
    - The ATOM_TYPE variable is used to set the “type” in the CoT message. This codes specifies the type of object represented by the message. CoT types can be found in the ./sample COT Data/CoTtypes.xml file
- Execute the CoTSim-QTF.py file.

UDP to TCP bridge for CoT

The UDP to TCP bridge is used to receive UDP packets (CoT messages) on a local network and forward them via a single long running TCP socket connection to a CoT / TAK server. The single long running TCP socket connection to the CoT / TAK Server is the key feature of this service. The CoT / TAK Server does not function well when a TCP socket is connected and closed for each message. This bridge allows multiple CoT messages to be sent over a single TCP connection. This bridge is designed to be used to allow the routing of CoT traffic through firewalls and the open internet. UDP traffic is often blocked by firewalls.

To run the UDP to TCP bridge for CoT:

- Unzip the JIFX-tools zip file.
- Navigate to the COT-UDP-TCP Bridge folder.
- Edit the ./ COT-UDP-TCP Bridge/UDP_Server.py file to set its configuration
  - UDP_HOST – localhost is normally used.
  - UDP_PORT – listening port for the local UDP traffic
  - TCP_HOST – The host name or IPv4 Address for CoT / TAK Server. For JIFX, coperstak.tdkc.com is often used when outside Camp Robert's JIFX network.
  - TCP_PORT – 8088 is the commonly used CoT / TAK Server port used to receive TCP socket connections
- Execute the UDP_Server.py file.

Log File to CoT example

At a JIFX several events ago, there was a experiment that was logging the position of their UAS to a file. They did not have the capability to publish to the CoT server. This script was created to watch and scrape the log file and publish the asset's location information as CoT messages to a CoT server. This is included for reference purposes.

This example is found at ./logScraper/logfile_to_cot.py

Sample CoT data

The 'sample COT data' folder contains to sample CoT messages that were created by the CoT Simulator. In this example, the there is a parent children relationship between the SwarmController (parent) and the Children (e.g. BravoOscar). This relationship allows for a set of sensors or vehicles to be viewed as belonging to the parent.

Required fields for a CoT Message

Field	Purpose
/event/@version	The CoT schema version to validate against. This document is only valid for '2.0'
/event/@uid	Differentiates new messages from updates to old ones
/event/@type	Describes what this message is
/event/@how	Describes how /event/point was derived
/event/@time	The time (in UTC) when this message was sent from the originating machine
/event/@start	The time (in UTC) this message becomes valid

/event/@stale	The time (in UTC) this message should no longer be considered valid
/event/point/@lat, /event/point/@lon	The position in WGS84 degrees of the event
/event/point/@ce	Radius of the bounding cylinder in meters for this object plus 1 sigma of lat/lon measurement error (this better not be 0)
/event/point/@hae	The altitude of the object, in meters above the WGS84 ellipsoid
/event/point/@le	Height of the bounding cylinder in meters for this object plus 1 sigma of hae measurement error (again, better not be 0)

Other useful CoT message fields

Field	Purpose
/event/detail/contact/@callsign	A friendly name to display on a user interface for this object
/event/detail/remarks	Human readable text associated with this object
/event/detail/link	A link to another object or an external resource @relation – type of link between the objects @type – type of the linked object @uid – uid of the linked object
/event/detail/track	Describes the object's velocity

SwarmController-sample.xml

```
<event
    how="m-s"
    stale="2021-02-24T16:08:29Z"
    start="2021-02-24T16:07:29Z"
    time="2021-02-24T16:07:29Z"
    type="a-f-G-U-U-M-R-X"
    uid="Swarm Controller"
    version="2.0">
    <detail>
        <track course="90.00" speed="20.5"/>
        <status battery="99.96"/>
        <medical content="71" id="Swarm Controller" name="HR"/>
        <medical content="37.26" id="Swarm Controller" name="CT"/>
        <_flow-tags_ TAK-Server-42d2348f="2021-02-24T16:07:29Z"/>
    </detail>
    <point
        ce="10"
        hae="1524"
        lat="39.650435646197685"
        le="10"
        lon="-83.84428820142662"/>
</event>
```

BravoOscar-sample.xml

```
<event
    how="m-s"
    stale="2021-02-24T16:08:29Z"
    start="2021-02-24T16:07:29Z"
    time="2021-02-24T16:07:29Z"
    type="a-f-G-U-C-V-U-R"
    uid="Bravo Oscar"
    version="2.0">
    <detail>
        <track course="338.29" speed="20.5"/>
        <link relation="p-p" type="a-f-G-U-U-M-R-X" uid="Swarm Controller"/>
        <status battery="99.96"/>
        <medical content="58" id="Bravo Oscar" name="HR"/>
        <medical content="36.87" id="Bravo Oscar" name="CT"/>
        <_flow-tags_ TAK-Server-42d2348f="2021-02-24T16:07:29Z"/>
    </detail>
    <point
        ce="10"
        hae="1524"
        lat="39.651209733268274"
        le="10"
        lon="-83.84308707149279"/>
</event>
```

Other included Files

Included in the package are several documents located in the ‘documents’ folder of the zip file.

The CoT Best Practices - Application to UAS Domain - 2011 1130 (PUBLIC RELEASE).pdf and CoT Developer's Guide - 2005 0831 (PUBLIC RELEASE).pdf documents are publicly released documents that were sourced from a TAK Server release. They are useful as reference material.

The CotTypes.xml file is useful when trying to find the ‘type’ to use to represent an asset in a CoT message. It is a useful cross reference between the types and human readable text.

Advisories integration

COPERS hosts an https endpoint that can receive alert, advisory, and notification data to be displayed in common operating picture. A Talend API Tester – Free Edition project is included that provides an example of how to POST the advisory to COPERS. The project file is

./Advisories/COPERS_Advisories.json. For the basic authentication credentials to use when posting the message, please contact john.leininger@centauricorp.com

Method: POST

URL: https://www.tdkc.com/MOLES_Server/input/notification?sourceID=JIFX

Request Headers:

Content-Type: application/json
Authorization: Basic ********************

Sample advisory message body

Remove the comments that start with ## to make this a 'usable' json.

```
{
    "id": "test15",      ## Unique Message ID
    "source": "JIFX",    ## Identifier used by COPERS to route the message
    "sourceType": "Demo", ## High level message category (keep it short)
    "notificationType": "Evacuation",  ## Type of notification
    "notificationTypeIcon": "exclamation-circle", ## name of icon used
    "importance": "High",       ## "Low", "Normal", or "High"
    "icon": "exclamation-circle",     ## name of icon used
    "time": "2021-02-24T14:43:00.000Z",     ## begin time for the notification
    "endTime": "2021-02-25T17:52:00.000Z",  ## time the notification expires
    "fromName": "John",                     ## who sent the message
    "fromAddress": "john.leininger@centauricorp.com",    ## email of sender
    ## Geo Location of the message / notification
    "latitude": 39.83398189973468,
    "longitude": -84.07201766967775,
    "radiusKM": 0.01,
    "content": "TEST15 -
- This is a test Notification generated by an external input",  ## message being
sent
    "properties": {
        "Name": "Value"
    }
}
```

Video Integration

There are multiple ways to integrate video into COPERS. If the video is hosted at a location directly reachable by the COPERS server, then the URL to the video can be provided to the COPERS event administrator and it can be configured into the operational picture. There are several video formats:

- RTMP video streams.
- Twitch, Facebook, YouTube (live & videos).
- URLs that host changing .png or .jpg files.

If the video stream is not reachable by the COPERS server, then a video relay can be used to proxy the video to the COPERS server. The Video Relay can be run on a network that can reach out to the COPERS server (e.g. a VPN), but where video is not available from outside the network. The video relay is a Java application.

To start the Video Relay, execute the .\VideoRelay\startVideoRelayGUI.bat file.

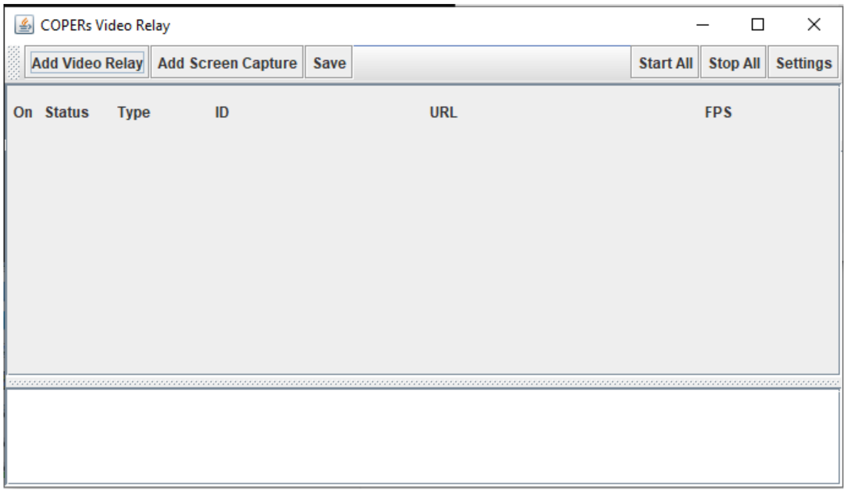
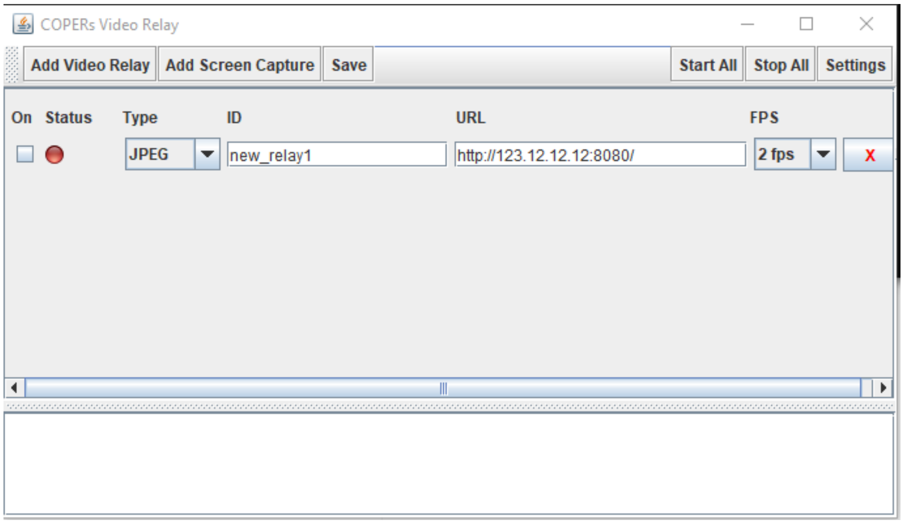


Figure 1- Initial Video Relay GUI

Within the Video relay, you can relay a specified video feed to COPERS and you can start a screen capture of part of your display and send that to COPERS. The screen capture feature will send updates to COPERS as the display in the captured region of the screen changes.

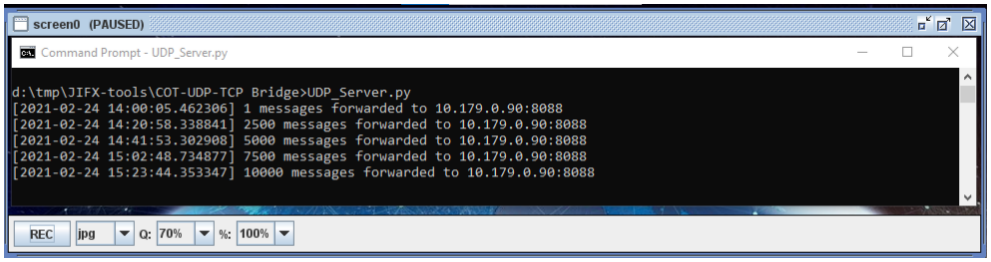
To add a new video relay instance, click on the ‘Add Video Relay’ button as depicted in Figure 1- Initial Video Relay GUI. This will create a new entry in the list that you can edit. Figure 2- Add New Video Relay shows the default values for a new Video Relay instance. Change the URL to be the URL of the video that will be relayed. Typically, we try to use a low frame rate (FPS) to keep the band width requirements small. The “ID” field is the name you give to the relay. It must be lower case and not contain spaces. This ID value must be given to the COPERS Event Administrator to finish the configuration in COPERS. To start the video relay instance, click on the ‘On” checkbox for the instance. When the Status icon is yellow, the instance is running but not sending data. When the status icon is green, it is connected and sending video data to COPERS. Click the ‘Save’ button to save your current configuration.



On	Status	Type	ID	URL	FPS
□	●	JPEG	new_relay1	http://123.12.12.12:8080/	2 fps

Figure 2- Add New Video Relay

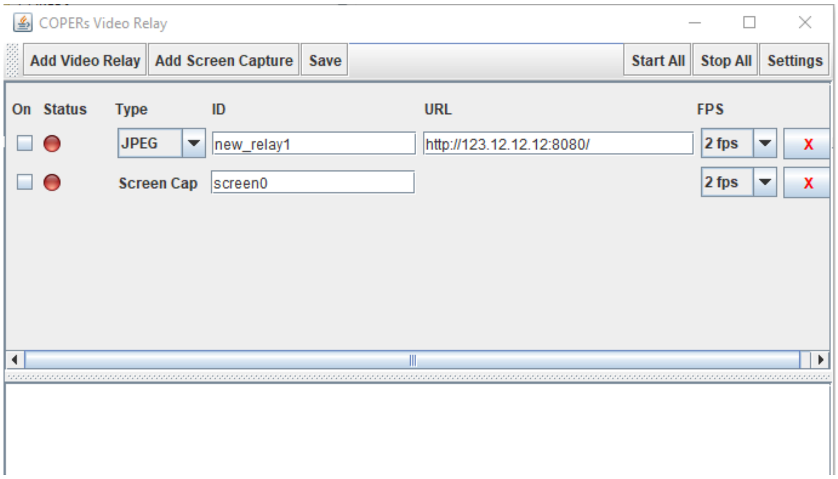
To start capturing and broadcasting a region of your screen, click the 'Add Screen Capture' button. It opens a window on the screen, Figure 3 - Screen Capture window, that has a transparent center. This window can be dragged around and resized. What is shown inside this window is what will sent to the COPERS server. Figure 4- Screen Capture config shows the configuration instance for the screen capture. Again, the ID specified for the screen capture instance must be given to the COPERS Event Administrator to complete the integration of the video into COPERS.



```
d:\tmp\JIFX-tools\COT-UDP-TCP Bridge>UDP_Server.py
[2021-02-24 14:00:05.462306] 1 messages forwarded to 10.179.0.90:8088
[2021-02-24 14:20:58.338841] 2500 messages forwarded to 10.179.0.90:8088
[2021-02-24 14:41:53.302908] 5000 messages forwarded to 10.179.0.90:8088
[2021-02-24 15:02:48.734877] 7500 messages forwarded to 10.179.0.90:8088
[2021-02-24 15:23:44.353347] 10000 messages forwarded to 10.179.0.90:8088
```

jpg Q: 70% %: 100%

Figure 3 - Screen Capture window



On	Status	Type	ID	URL	FPS
□	●	JPEG	new_relay1	http://123.12.12.12:8080/	2 fps
□	●	Screen Cap	screen0		2 fps

Figure 4- Screen Capture config

COPERS Points of Contact

Mr Raymund Garcia, AFRL/RYW, COPERS Program Manager

raymund.garcia.1@us.af.mil

Mr Preston Hoeve, COPERS Principle Investigator

Preston.Hoeve@centauricorp.com

Mr John Leininger, COPERS Senior Technical Staff

John.leininger@centauricorp.com