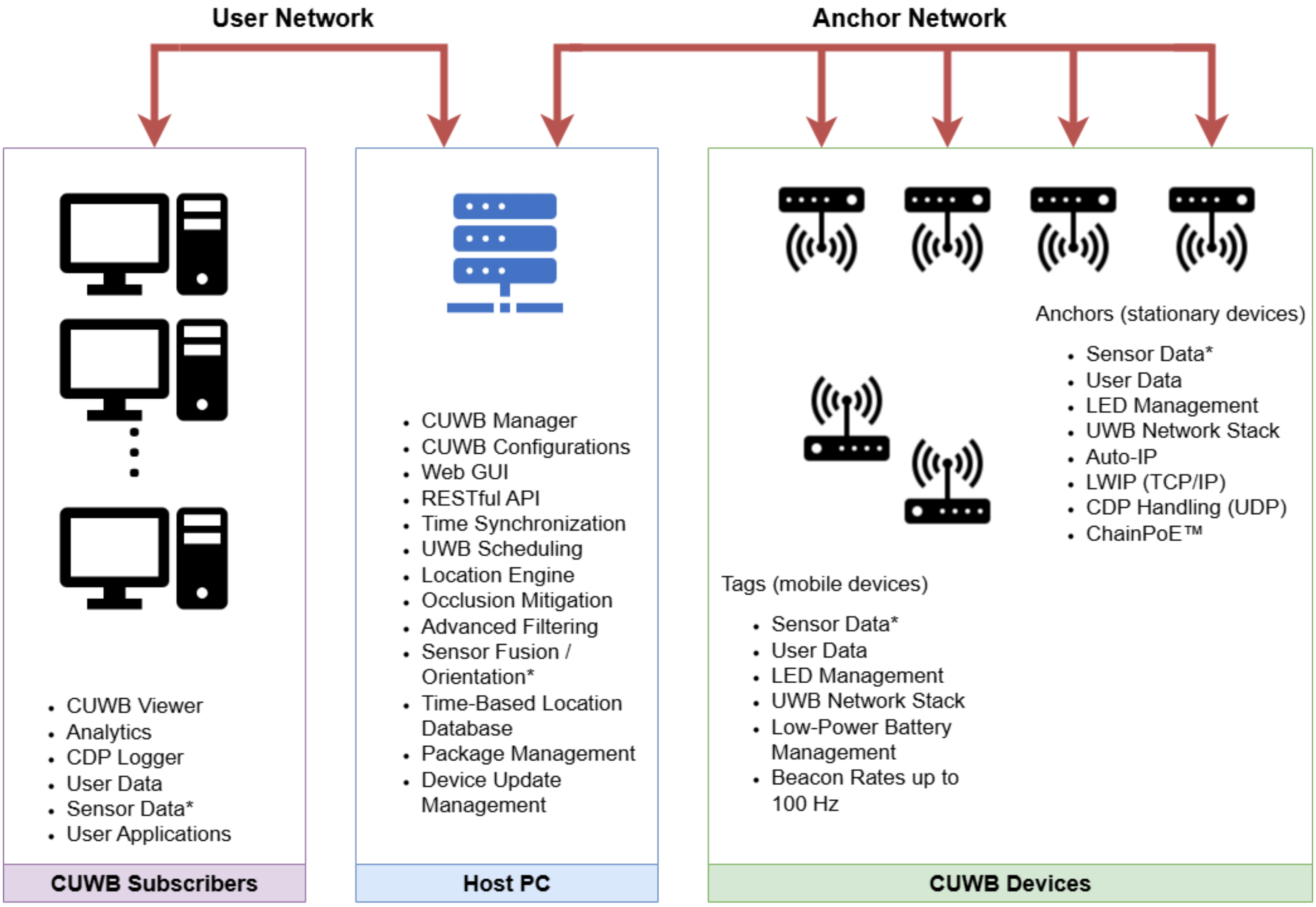
1 Architecture Overview

This document describes the CUWB Real-Time Location System (RTLS) architecture and associated components. It provides a high level overview of the system along with details on how user applications can integrate with the system and consume location data.

2 System Components Diagram



* Only Available on supported hardware

2.1 Host PC

The CUWB system utilizes a Host PC running the CUWB Management Package and other support applications. The Host PC typically runs 64-bit Ubuntu LTS, and is the heart of the CUWB system.

There are two major components on the Host PC: the CUWB Manager and CUWB Configurations. A CUWB Configuration is a collection of settings that define CUWB RTLS device behavior and system outputs. A running CUWB Configuration is referred to as as a “CUWBNet,” and represents a collection of anchors and tags whose behavior and outputs are governed by the configuration. The CUWB Manager is a web based tool for managing and interacting with CUWB Configurations or CUWBNets. It can be used to launch (aka. start and stop) CUWBNets as well as provide status and other pertinent system information. CUWBNets also manage the underlying RTLS code, referred to as the CUWB Engine, that processes incoming timing data transforming it into output location.

The CUWB System may use one or more computers running as a Host PC. Systems that utilize more than one Host PC must be connected on the same Anchor LAN for intercommunication.

Please reference the CUWB Manager Manual for details on Host PC requirements and details on installation.

2.2 CUWB Subscribers

CUWB subscribers are applications that consume data from a CUWBNet instance. These applications range from Ciholas provided applications to custom integrations. Details on some of these applications can be found in Support Applications below.

Typical subscriber applications consume Ciholas Data Protocol (CDP) data that is received via a configured UDP stream, referred to as the “User Data Stream”. An example application that consumes CUWB output location data via CDP can be found here: Using CDP Application Note. More example applications of CUWB subscribers can be found on the Ciholas GitHub.

2.3 CUWB Devices

CUWB devices communicate over Ultra-Wideband (UWB) and are compatible with the CUWB RTLS system. Some CUWB devices also connect to a running CUWBNet over Ethernet utilizing CDP, or via a serial connection using Ciholas Serial Protocol (CSP) which is based on CDP.

There are two classes of CUWB devices: Anchors and Tags.

2.3.1 Anchors

To generate meaningful position output, RTLS applications require known physical locations within a region to compare timing data. Anchors are devices placed in a region of interest that act as a static reference point for the CUWB System.

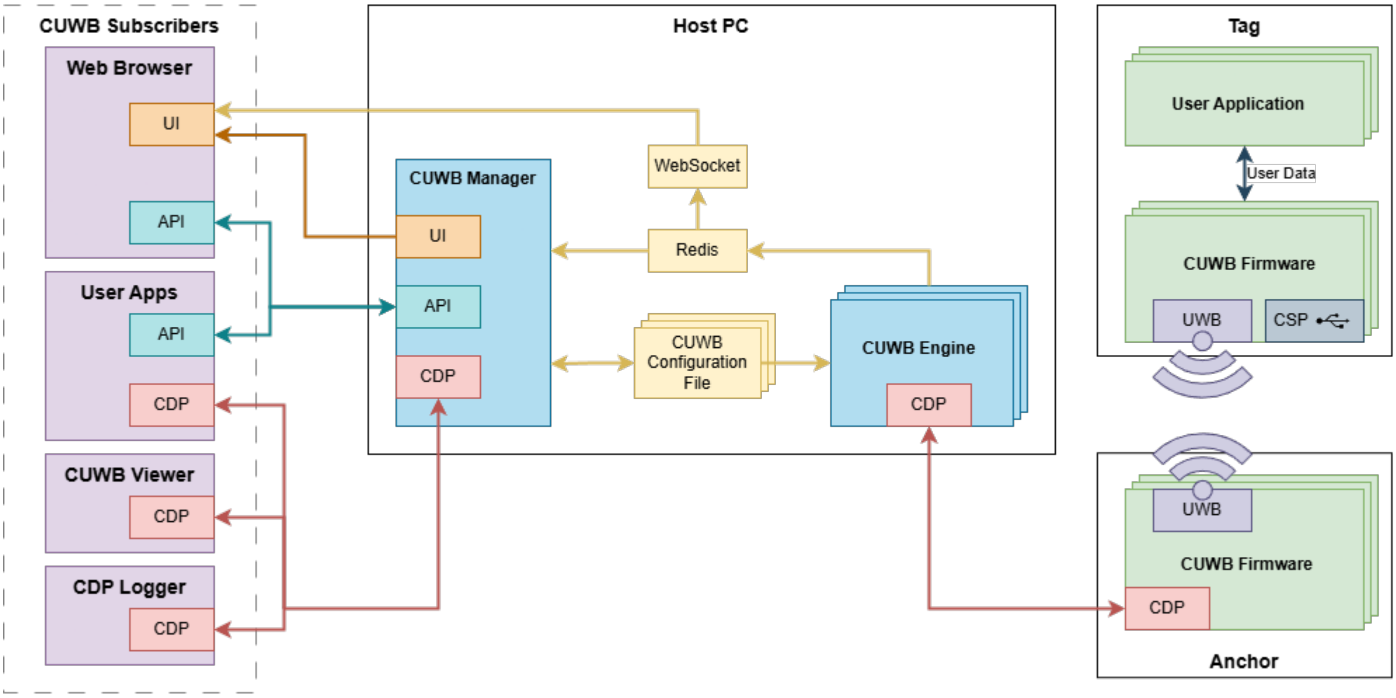
Anchors must have Ethernet connectivity to communicate with the CUWBNet for configuration, scheduling, commands and data transport.

2.3.2 Tags

Tags are mobile devices placed in a region of interest that are trackable. Anchors hear UWB transmissions from tags and report receive timing to the CUWB Engine where the information is used to compute the location of each tag. Tags can be placed on objects or personnel to provide real-time location data.

3 System Architecture Diagram

The following diagram illustrates the logical connections between the Anchor Network, Host PC, and CUWB Subscribers. The connections lines show various application end-points and communication between hardware and software within the CUWB System.



3.1 CUWB Manager

The CUWB Manager is a process that runs on the Host PC. The process runs at system startup, and is used to create, configure, and perform actions on CUWB Configurations. It hosts the CUWB Manager frontend to HTML clients. See the CUWB Manager Manual for additional instructions on use.

3.1.1 API Backend

The backend is a python-based web server that exposes a RESTful HTTP API. The documentation for the API is available under API Control.

The backend reads and writes to the CUWB Configuration File and monitors CDP Streams for status information. The backend utilizes redis for interprocess communication with the CUWB Engine.

3.1.2 UI Frontend

The frontend is a web application served by the CUWB Manager which provides end users with a convenient interface to configure, monitor, and control CUWB Configurations and CUWBNets.The frontend is accessed by a browser. It utilizes the RESTful API to communicate with the backend, providing users with up-to-date information regarding CUWB Configurations and CUWBNets. Additionally, the frontend opens a WebSocket connection to receive real-time updates from the Host PC. For a walkthrough of the CUWB Manager UI, see the CUWB Manager GUI Walkthrough.

3.1.3 Websocket

The CUWB Manager launches a separate ‘WebSocket’ process at startup. The WebSocket process is used to send events, such as device and network status, to frontend instances.

3.2 CUWB Configurations & CUWBNets

A CUWB Configuration is a user configured set of UWB devices and settings that define a CUWBNet. A CUWBNet is the running, or operating, collection of anchors and tags whose settings are defined by a CUWB Configuration. The Host PC can have multiple CUWBNets running at the same time.

3.2.1 CUWB Configuration File

CUWB Configuration Files are databases with the .cuwb file extension stored on the Host PC. The configuration files are read and monitored by the CUWB Manager and allow for dynamic changes even when the associated CUWBNet is running.

3.3 CUWB Engine

The CUWB Engine is a process that is instantiated by the CUWB Manager using the settings defined by a CUWB Configuration. In the frontend User Interface (UI) users ‘start’ a CUWB Configuration resulting in a CUWB Engine (active CUWBNet instance) that is based on that configuration. The CUWB Engine is responsible for managing Anchors and Tags: scheduling UWB transmission, synchronizing time, calculating locations, and much more.

3.3.1 Configuration Management

When a CUWB Engine is launched, it reads the CUWB Configuration file and schedules device transmission over UWB accordingly. It monitors the CUWB Configuration file for any changes, making adjustments to network and device parameters as indicated by the configuration file.

3.3.2 Scheduler

The CUWB Engine implements a scheduler that is responsible for managing the UWB transmission and receptions for all CUWB devices in a CUWB Configuration. The schedule is based on parameters provided by the user using the CUWB Manager front-end web interface. These parameters define various network behaviors such as Tag beacon rates, Anchor synchronization rate, smoothing, etc.

3.3.3 Time Synchronization Management

The CUWB Engine tracks the internal drift and offset of the Anchor clocks. The CUWB Engine generates a common network time that is utilized by each device to schedule transmissions and receptions. Network time is used for precision time stamping of events and is used by the CUWB Engine for Tag localization.

3.3.4 Localization

The CUWB Engine is responsible for collecting timestamps for all UWB transmissions and receptions defined by a CUWB Configuration. UWB transmit and receive timestamps are converted to network time and are used by the CUWB Engine for localization. Depending on the network mode (MultiTime™ or MultiRange™), the CUWB Engine calculates ranges or time differences between devices and implements an algorithm to determine XYZ location of devices based on that data.

3.3.5 CDP Processing

The CUWB Engine's primary Real-Time form of communication is CDP (Ciholas Data Protocol) over UDP/IP. The CUWB Engine has three main streams of CDP traffic and three user optional streams :

Stream Name	Definition
Anchor Stream	Information sent from Anchor Network devices to the CUWB Engine.
User Stream	Output data for User Applications to track position, configuration, status, and sensor information.
Config Stream	Data exchange utilized to bring new devices online.
Command Stream	Data exchange utilized by the CUWB Engine to send multicast commands to UWB devices.
Debug Stream	[*Optional Stream*] Information sent from the Anchor Array containing debug information and statistics.
Data to Device Stream	[*Optional Stream*] CDP commands packets from external applications bound for CUWB tags and anchors.

3.4 CUWB Firmware

Each device in a CUWB Configuration utilizes CUWB Firmware. The firmware has limited default behaviors, along with configurable applications that help the devices fulfill their roles within a CUWB Configuration.

The CUWB Engine provides a list of commands to each device upon joining a CUWB Configuration. These commands determine how the device will interact, when it will receive and transmit, what sensors it will use, and other behaviors like Wake-on-Shake.

3.4.1 UWB Communications Search

All devices, even Ethernet connected devices, listen for UWB traffic periodically when they are not configured to be in a CUWBNet. Once a device is able to detect and lock into a CUWBNet, a UWB packet is sent to join the device to the CUWBNet. The CUWB Engine gives each device their commands, allowing the device to behave according to the CUWB Configuration.

3.4.2 Command Processing / Applications

Each CUWB device handles commands from the CUWB Engine. The firmware has separate applications to handle each command. The applications can do any number of tasks, such as setup sensors, drive LEDs, or schedule events in the UWB stack to cause transmissions and receptions to occur.

3.4.3 Ethernet Devices

CUWB devices with Ethernet connectivity can serve as Anchors. After devices establish an Ethernet link, they send discovery CDP packets on the Config Stream. The CUWB Engine utilizes these discovery packets to detect new devices and send them commands.

3.5 Support Applications

3.5.1 User Applications

Customers often run their own applications to log, monitor, and control CUWBNets. These applications have access to position data, sensor data (if available), and user data.

CDP processing from the User Stream is the most common connection. Some applications utilize the API to dynamically change configuration and device settings. Additionally, some applications send CDP to a CUWBNet using the Data to Device Stream.

3.5.2 CUWB Viewer

CUWB Viewer provides a user interface displaying the locations of CUWBNet devices, such as anchors and tags in real time. The CUWB Viewer also presents basic system information, such as device status, statistics, sensor plotting, and more.

CUWB Viewer also gives users the ability to import objects and create 3D environments replicating the physical space of a CUWBNet. Additional information can be found in the CUWB Viewer Manual.

The CUWB Viewer listens to the User Stream.

3.5.3 CDP Logger / Player

The CDP Logger records data from the CDP Streams. The player can playback the logged data in a time synchronized fashion. The logger and player are useful in debugging and development. Additional information can be found in the CDP Logger Manual.

4 System Interface

4.1 Data Hooks

The Ciholas Data Protocol (CDP) provides a method of communication between devices and services. CDP data is transmitted over ethernet as User Datagram Protocol (UDP) packets. Ciholas Real-Time Location Systems (RTLS) emit position data using the CDP format as an accessible way for users to gain access to the data and integrate their software. CDP packets are transmitted through CDP Streams. CDP Streams are identified by the Ethernet interface, IP address, and Port through which the packet is sent. Streams are allowed to be both multicast and unicast. Different CDP Streams are defined to distinguish different categories or classes of data.

The CUWB RTLS supports using Ciholas Data Protocol (CDP) over a User Datagram Protocol (UDP) connection. The CDP packet definitions are available under Ciholas Data Protocol.

4.2 Control Hooks

The CUWB RTLS has an RESTful API available for use. This API configures all of the resource units required to facilitate the creation, configuration, and deletion of CUWB Networks within the CUWB RTLS platform. Users have the option of creating their own control mechanisms utilizing this API. The expected response formats are also provided for success, failure, and error scenarios.

The CUWB RTLS API is available under CUWB Manager - API Control

4.3 User Data

A CUWBNet can be configured to allow Tags to send additional data (User Data) along with beacons. The Tags are configured to send this data over time, using fragmentation, to reduce the air traffic required to support this additional data. CUWB Engine collects and reassembles the fragmented data. Upon full reassembly the CUWB Engine emits a CDP Data Item on the User Stream with the contents of the additional data.

4.4 SNMP Support

Ciholas is working to provide SNMP support for Enterprise level customers. Keep an eye out for additional SNMP announcements.