This document is subject to change. The latest version may be found at https://github.com/ashawnbandy/cecs491/tree/master/docs

10/29/2012

CSULB Marine Biology Department Software Project

*Functional Analysis (rev 1)*

Contents

[Class Specification 2](#_Toc339467823)

[Server 2](#_Toc339467824)

[Modules 2](#_Toc339467825)

[Events 2](#_Toc339467826)

[Client 3](#_Toc339467827)

[Class Diagrams 4](#_Toc339467828)

[Server 4](#_Toc339467829)

[Modules 5](#_Toc339467830)

[Application Events 5](#_Toc339467831)

[Client 6](#_Toc339467832)

[Entity Attributes 7](#_Toc339467833)

[Server 7](#_Toc339467834)

[Receiver 7](#_Toc339467835)

[Sequence Diagrams 8](#_Toc339467836)

[Receiver Life-Cycle 8](#_Toc339467837)

[Start/Stop Receiver Hardware 9](#_Toc339467838)

[Store Recorded Data 10](#_Toc339467839)

# Class Specification

## Server

* SerialPortManager – Boundary

This class monitors the system for changes in the serial port enumeration (new or removed serial ports). When a new serial port is discovered a Receiver is created. When the serial port is removed from the system enumeration or a Receiver consistently reports errors in decoding, the manager removes the device from the service.

* Receiver – Entity

Encompasses the functionality required to connect to and communicate with a VR2C receiver connected via serial port.

* Encoder – Control

The Encoder builds commands as requested by the Receiver (possibly through public write() methods), by looking up the requested command in the configuration file. The configuration file specifies the format and the encoder uses String.Format to build the string that is finally returned. Before returning the string, however, it is verified as conforming to the config's spec.

* Dispatcher – Control

The dispatcher maintains a thread-safe queue of events. When a new event arrives, the event is dispatched to any registered Modules. Any registered module may also enqueue a RealTimeEvent for distribution.

* Modules – Interface Definition

This class contains the specification all classes wishing to participate in the event system must implement.

## Modules

* Decoder – Control

Decodes raw messages sent from the VR2C hardware.

* ServerUI – Boundary

Provides the user interface to display the state of the service and modules and to allow the end-user to control over the VR2C hardware.

* Database – Boundary

Sends previously decoded messages from the receiver to external data stores (e.g. csulbsharklab.com MySQL database).

## Events

* RealTimeEvent -- Control

The interface from which all other event objects sent through the system dispatcher are derived.

* NewReceiver -- Control

Sent when a new Receiver has finished initialization and is ready to receive incoming messages from the VR2C hardware.

* UnparsedMessage -- Control

Messages sent from the VR2C receiver are sent out as a “raw” string in this object type.

* DelReceiver -- Control

Sent when a receiver has been taken off-line. This occurs when the receiver has experienced multiple errors in reading or writing or when the VR2C hardware has closed the serial port connection.

* ExcepReceiver -- Control

Usually accompanied by a system exception while communicating with the hardware. Can also be generated when malformed (unknown or otherwise unparsable) messages are received from the hardware.

* NoteReceiver -- Control

This event is used to relay general information about the system that may be of interest to a human operator. No action or response is required from any receiving module.

* ServerStartUp -- Control

Sent when the SerialPortManager begins to accept hardware serial port connections.

* ServerStop -- Control

Sent when the SerialPortManager is no longer accepting new hardware connection but has not removed operating Receiver objects.

* ServerStopped -- Control

Sent when the SerialPortManager has stopped all Receivers.

* RealTimeEventDecoded -- Control

When the text from the UnparsedMessage event is decoded, the results are sent through this event.

* DatabaseResponse – Control

Sent when an operation is attempted with an SQL database; contains the response from the database and the statement attempted (most often an INSERT).

## Client

## Front-end ([www.csulbsharklab.com](http://www.csulbsharklab.com) webpage)

## Sharktopus

* Query\_builder\_realtime – Control

This class builds SQL query as string which will be sent to database.

* Query //constants variables… – Control

Holds constant values that are used for building the query in Query\_builder\_realtime class.

* Mysqli\_Connection – Entity

Holds a connection with database.

# Class Diagrams

## Server



## Modules



## Application Events



## Client



# Entity Attributes

## Server

### Receiver

Attributes

* public string portName
* public int TTL
* private const int DEFULT\_TTL
* private const int COM\_READ\_TIMEOUT\_DEFAULT
* private const int COM\_READ\_TIMEOUT\_SPIRAL
* private const string VR2C\_COMMAND\_FOLDER
* private Encoder encoder
* private Dispatcher dispatcher
* private int firmwareVersion
* private TextReader textReader
* private int goState
* private int write\_wait
* private static char[] crlf

Methods

* public Receiver(SerialPort serialPort, String portName, Dispatcher dispatcher)
* public void init()
* private void \_write(string text)
* public void write(string command, object[] arguments)
* public void write(string command)
* public void shutdown()
* public async Task run()

## Front-end ([www.csulbsharklab](http://www.csulbsharklab) website)

\*\*\* Note: done in php (script language).

### Query\_builder\_realtime

Attributes

* none // they are all local variables within the functions.

Methods (addition/reused to/from previous website)

* public static function GenerateQuery();
* public static function GenerateCountQuery();
* protected static function BuildFromClause();
* protected static function BuildWhereClause();
* protected static function BuildFullQuery($select, $from, $where, $sort = ’’, $order = ‘’);

### Query // this class just has a list of constant variables

Attributes

* const Project
* const OuterQueryType
* const InnerQueryType
* const Fields
* const SortBy
* const SortOrder
* const Limit
* const Count
* const Query
* const Page
* const DateStart
* const DateEnd
* const TimeStart
* const TimeEnd
* const StationKey
* const Receiver
* const TransmitterRangeStart
* const TransmitterRangeEnd
* const Genus
* const Species
* const FrequencyCodespace
* const TransmitterID
* const TransmitterID
* const Weeks
* const Days
* const Hours
* const Minutes
* const Seconds

Methods

* None.

### Mysqli\_connection

Attributes

* None. // they are all local variables within the functions.

Methods

* public static function GetMysqliInstance();
* public static function Disconnect();

# Sequence Diagrams

## Receiver Life-Cycle



## Start/Stop Receiver Hardware



## Store Recorded Data



