HubPP ModemsManager API

1. Data structures
2. Functionality API
3. Logs
4. ModemsManager Test
5. Data structures

Modem’s statistics

typedef struct {

uint64\_t rx\_bytes;

uint64\_t tx\_bytes;

uint64\_t rx\_pkts;

uint64\_t tx\_pkts;

uint64\_t tx\_ts\_pkts;

uint64\_t rx\_ts\_pkts;

uint64\_t tx\_acks;

uint64\_t rx\_acks;

uint64\_t tx\_noacks;

uint64\_t rx\_noacks;

uint64\_t tx\_retransmits;

} modem\_stat\_t;

Each modem should contain statistics information, this information gathered during modem’s operations RX/TX/ACKs

The external ModemsManager API is kept in **modemmgr\_api.h** header file

There is additional header file - **modem\_common.h**, which is in use between internal ModemsManager module’s parts

1. Functionality API
   * int init\_modemmgr(const char\* modems\_cfg\_file); - Initialized ModemsManager, the configuration file should be supplied

Note: Configuration file currently not supported, the NULL should be delivered as a parameter

* + int start\_modemmgr(void); - Starts the ModemsManager
  + int stop\_modemmgr(void); - Stops the ModemsManager
  + int get\_modem\_stat(modem\_stat\_t\*\* stat, int modem\_ind); - Get Modem’s Statistic structure, currently only modem\_ind = 0 is supported
  + struct circ\_queue\_t\* get\_modem\_rxq(int modem\_ind); - Retrieves the RX queue of specified modem
  + struct circ\_queue\_t\* get\_modem\_txq(int modem\_ind); - Retrieves TX queue of specified modem
  + struct circ\_queue\_t\* get\_modem\_ackq(int modem\_ind); - Retrieves ACKs queue of specified modem

1. Logs

HubPP ModemsManager module uses syslog for writing logs

There is USER\_LOG facility is used

HUBPP Log supports 4 log levels: LOG\_INFO, LOG\_ERR, LOG\_DEBUG and LOG\_WARNING

For each of levels, there are functions (macros) - msgInfo, msgErr, msgDbg, msgWarn

On Debian 10 system, rsyslog – is default system logger. The /var/log/user.log is create and HubPP messages redirected to this file. There is probably an ability to configure rsyslog or any other logger to keep application’ specific log in standalone file. But this is out of scope of this document

Here is an example of HubPP log:

**May 6 14:49:08 orion hubpp-modemsmgr: [LOG\_INFO] [deqeue\_elem:98] queue (modem0-txq) is empty!**

What we can see from follow log:

Date: May 6 14:49:08

Host: orion

App: hubpp-modemsmgr

Log Level: [LOG\_INFO]

Function: deqeue\_elem

Line: 98

Message: queue (modem0-txq) is empty!

1. ModemsManager Test

ModemsManager tests simulates the work of ULC modem, tries to send TS data packages, to received TS data packages and to send ACKs as a result on received Network message

For current test, ModemsManager defines two UDP ports, one for receiving data from ULC Modem and other one is to send data to ULC Modem. It can’t be one port due to we’re testing it on same PC, there only one bind socket for specified port can be created

ModemsManager test runs continuously till the application will not receive SIGHUP signal from shell

The received TS packets from “ULC Modem” is written to /tmp/hub/bin.

The file is overwritten on each new Network Data packet