**Protocol Analyzer Requirement**

1. **Purpose**

Protocol Analyzer is aimed to help FC software developers test and debug implementations of DDD protocol.

1. **Functionality**
   1. **Test Message Format**
      1. **Common Message Format**

* All Messages should confirm following format:

MSG\_ID, CMD\_NAME, ARG\_1, ARG\_2 .. ARG \_N<CR>

* All Messages should be in ASCII or GB18030 encoded.
* All Parameters in each single Message should be separated by Comma (“,”).
* Each Message should contain at least 2 fields, <MSG\_ID> and <CMD\_NAME>.
* <MSG\_ID> should be an Integer and confirms to following range.

FC --> DDD: [0, 99]

DDD --> FC: [100,199]

* <MSG\_ID> should be same between “Send” & “Response” Messages, except following cases.

FC --> DDD: <MSG\_ID> is 0 if FC doesn’t expect response from DDD.

DDD --> FC: <MSG\_ID> is 100 if DDD doesn’t expect response from FC.

* <ARG\_1> ~ <ARG\_N> should conform format defined by different <CMD\_NAME>.
* <CMD\_NAME> should be in following list (Case Sensitive).

GET\_CONFIG

GET\_STATUS

UPDATE\_STATUS

REPORT\_EVENT

REPORT\_UNIT\_LOCATION

REPORT\_RESULT

REPORT\_KPI

* + 1. **GET\_CONFIG Message Format**
* FC should NOT send this message to DDD.
* Format

GET\_CONFIG,DEV\_TYPE,DEV\_OS\_VER,DEV\_APP\_SW,PROTO\_VER,FC\_ID,DATE,TIME<CR>

* <DEV\_TYPE> should be a string.
* <DEV\_OS\_VER> should be a string.
* <DEV\_APP\_SW> should be a string.
* <PROTO\_VER> should be a string.
* <FC\_ID> should be a string, which should be unique on the line.
* <DATE> should conform “MM/DD/yyyy”, e.g. “02/24/2017”.
* <TIME> should conform “HH:mm:ss” and 24-hour format, e.g. “17:24:23”.
  + 1. **GET\_STATUS Message Format**
* FC need to respond using UPDATE\_STATUS
* FC should not send this message to DDD.
  + 1. **UPDATE\_STAUS Message Format**
* DDD should not send this message to FC.
* This message must be a response to a <GET\_STATUS> message
* This message need to send to DDD periodically (async)
* Format

UPDATE\_STATUS, FC\_STATE,IO\_BIT\_MAP,ERROR\_BIT\_MAP<CR>

* <FC\_STATE> is a string, which must be in following list (Case Sensitive).

AUTOMATION,

ALARM,

IDLE,

MANUAL

* <IO\_BIT\_MAP> must be an integer.
* <ERROR\_BIT\_MAP> must be an integer.
  + 1. **REPORT\_EVENT Message Format**
* Format

REPORT\_EVENT,UUT\_ID,UUT\_LOCATION,EVT\_ID,EVT\_TYPE,EVT\_MSG<CR>

* <UUT\_ID> must be a string. <UUT\_ID> could be empty but UI should pop up warning for this.
* <UUT\_LOCATION> must be a string, which must be in User-Defined location list (Case Sensitive). <UUT\_LOCATION> could be empty, but UI should pop up warning for this.
* <EVT\_ID> must be an integer, which must be in User-Defined event list.
* <EVT\_TYPE> must be an integer, which mush be in following list (Case Sensitive).

USER

ALARM

WARNING

INFO

* <EVT\_MSG> must be a string.
  + 1. **REPORT\_UNIT\_LOCATION Message Format**
* Format

REPORT\_UNIT\_LOCATION,UUT\_ID,UUT\_LOCATION<CR>

* <UUT\_ID> must be a string.
* <UUT\_LOCATION> must be a string, which must be in User-Defined location list.
  + 1. **REPORT\_RESULT Message Format**
* Format

REPORT\_RESULT,UUT\_ID,UNIT\_RESULT,START\_TIME,END\_TIME,UUT\_LOCATION,GRADE\_CODE,MESSAGE

* <UUT\_ID> must be a string.
* <UUT\_RESULT> must be a string, which must be in following list (Case Sensitive).

PASS

FAIL

* <START\_TIME> must be a string of time.
* <END\_TIME> must be a string of time.
* <UUT\_LOCATION> must be a string, which must be in User-Defined location list.
* <GRADE\_CODE> must be an integer. <GRADE\_CODE> could be empty if <UUT\_RESULT> is PASS. Otherwise, UI should pop up warning for empty string.
* <MESSAGE> must be a string. <MESSAGE> could be empty if <UUT\_RESULT> is PASS. Otherwise, UI should pop up warning for empty string.
  + 1. **REPORT\_KPI Message Format**
* Format

REPORT\_KPI,UUT\_ID,KPI\_NAME,UNITS,MIN,MAX,VALUE,…<CR>

* <UUT\_ID> must be a string.
* Each KPI should contain <KPI\_NAME>, <KPI\_UNITS>, <MIN>,<MAX>,<VALUE>
* <KPI\_NAME> should be a string.
* <KPI\_UNITS> should be a string.
* <MIN> must be a double if not empty.
* <MAX> must be a double if not empty.
* <VALUE> must be a double.
  1. **Test Message Sequence**
     1. **<MSG\_ID> in request message and response message**
* <MSG\_ID> should be match in both request message and response message.
* If FC --> DDD: 20, SOME\_CMD, then response must be : 20, SOME\_CMD.
  + 1. **<REPORT\_UNIT\_LOCATION> message and its function**
* <UUT\_LOCATION> should follow same order of User-Defined location list.
* The order MUST NOT be random.
* The User-Defined location list should conform:

The first location should have “Load” function.

At lease 1 location should have “Operation” function.

At lease 1 location should have “Unload” function.

“Unload” function must be after “Load” function.

* Each unit should go through all the locations. Should not skip any one.
  + 1. **<UUT\_ID> in all messages**
* <UUT\_ID> must be same in a message loop.
* Each individual UUT must appear from first location.
* Each individual UUT can disappear when and only when it comes to last location.
  + 1. **<UUT\_LOCATION> vs. UUT’s actually location**
* <UUT\_LOCATION> should match with UUT actual location.
* <REPORT\_EVENT>, <REPORT\_RESULT>, <REPORT\_KPI> also have <UUT\_LOCATION>. When those messages come, they have to match UUT actual location. If not, UI should pop up Alarm.
  + 1. **<REPORT\_RESULT> message timing**
* <REPORT\_RESULT> should come when and only when the UUT is about to leave location with “Load” function. Before <REPORT\_UNIT\_LOCATION> reports to other location, <REPORT\_RESULT> message must be sent.
* Before unit comes to unload location, at least 1 <REPORT\_RESULT> message must be sent.
  1. **UI Display**
* UI should display all messages received and sent.
* UI should attach analyze result after each message. The Result could be Alarm with red color and Warn with yellow color.
* UI should allow user to select TCP mode such as client or server.
  1. **TCP Communication**
* TCP should support both Client and Server mode.