RoundTripTime

For debugging purposes, it might be interesting to determine the round trip time (RTT) within the system. We use RTT messages (https://git.rwth-aachen.de/CPM/Project/Lab/software/-/blob/master/cpm_lib/dds_idl/RoundTripTime.idl) to compute this time:

1. Send an RTT message to another participant
2. Wait for the participant to answer with another RTT message
3. Use the time diff to compute the round trip time in the system

Some more information on its usage can be found here.

Usage

The module that abstracts this functionality is called RTTTool and can be found here:

https://git.rwth-aachen.de/CPM/Project/Lab/software/-/blob/master/cpm_lib/include/cpm/RTTTool.hpp

You only need to consider three functions:

Instance()

This function creates and returns an RTTTool Singleton that you can use within your program. I uses the ParticipantSingleton internally, so you can only measure the RTT within that domain.

activate(...)

Call this function to allow it to **answer to RTT measurement requests**. You are only allowed to **either activate or measure** the RTT with the same program, but do not do both at the same time - it does not make sense to answer to RTT requests send by the same program.

You need to set an **ID** that identifies your program. It does not need to be uniquely identifiable - i.e. the vehicles in the Lab all identify as *vehicle*. But it should identify your program as what it is, so that the information makes sense in the measurement, and does not get mixed up with e.g. the information sent by the vehicles.

measure_rtt()

Call this function to **measure the RTT** in the network. After some computation time (waiting for answers etc.), it returns a map:

- Key: std::string, **ID** of the answers
- Value: **Lowest and highest RTT** measured with this ID (e.g. for multiple vehicles: Fastest and slowest RTT)

You can use this information e.g. to display it in the LCC, which has already been implemented for the vehicles and HLCs.