# 1.0 Application parameters

The Program can be started with the following commands (the order does not matter):

**-listen** : starts the listening with startup

**-console** : the console takes some commands to control the program (see 1.2)

**-noView** : the program will be started without the main gui view

**-import:<filename>** : starts the program and runs the script with the given filename. The file needs to be in the saves/scripts folder of the project

# 1.1 Creating and Sending Messages

The Radio monitoring tool (RMT) allows specifying and sending messages to the network. The header of the messages files need to be converted by the MIG tool from TinyOS and the generated class files have to be in the mote folder of the RMT-project. The *message button* in the Measurement panel (tool icon) opens the **Message creation window**. The window shows the actual message, which is the last created or updated message. If no message is created yet, it shows a selection box, which displays all available message types (which class files are in the mote folder).

When a type is selected a new message of this type can be created by pressing the *new/load button*. The Window will then display all values that can be specified for this message type. In the case of C unions it is important that some fields are left empty. If a union variable can have two different types only one has to be set and the second needs to be left empty. After all values are set the name of the message can be defined in the field in the left bottom of the window. To finalize the message and make it the next message to send, the user can either press *done*, which closes the window or *create/update*.

All created messages (Set of Messages) can be selected from the box in the top right corner. The *new/load button* loads the message with its values, so that the user may change it or make it the next message to send (pressing create/update).

Next to the *message button* in the main RMT fram is the *send button* (play icon), which sends the actual message (the last created or updated) to the network.

# 1.2 Script commands

The program also allows to write bashes of commands (message creation, sending) and run them in a defined schedule. Under the menu *File/Script* the program opens the **Scripting Frame**. The frame shows the last created or loaded Script. Following commands can be written, with one command per line. Between each word or value has to be one space. Words (or symbols) are just typed out, values are in <> parenthesis, with the valuename and the valuetype (for example node addresses are integer values <address:int>)

wait <time:int>?

The wait command halts the script for a defined time. If wait is called without a time value then it is called with the default time, which is 500 (ms).

msg <? <messagename:string> <messageType:String> ( <address:int> | ALLKNWON ) <parameter\_i:type\_i>\*

This command creates a message, which can be used in the send command. If the optional parameter < is used the message will be added to the Set of Messages, so it can also be edited in the Message creation window and used in other scripts.

send <messagename:String> |

(<messageType:String> ( <address:int> | ALLKNWON ) <parameter\_i:type\_i>\* )

Send can be called with the name of a known message or the message to be sent can be specified in the command line. As a help for the user to create the messages with right values, the user can press F1 directly after the name of the message type. The Script Frame will add the names and data types of all values for this message type.

All messages which have been defined earlier in the script with the msg command and all messages from the Set of messages can be used.

Example:

(The class RssiMessage just contains one short field called rssi.

So the send command, which defines a message to send includes: RssiMsg <address:int> <rssi:short>)

send myRssiMsg

send RssiMsg 1 -70

import <scriptname:String>

This command includes another script into this script. The script has to be in the saves/scripts or saves/messages folder of the project. The script which is referred to will be executed, where the import command is called, so it will be as if instead of the import command all commands from the given script are executed.