Grant Agreement No. 611373

FP7-ICT-2013-10

# D1.2.1. Initial communication transceiver units, protocols and software

Due date of deliverable: 11/2014
Actual submission date: 11/2014

Start date of project: 01 January 2014 Duration: 36 months

Organization name of lead contractor for this deliverable: UNEW

Revision: version 1

| Dissemination level | | |
|---|---|---|
| PU | Public | x |
| PP | Restricted to other programme participants (including the Commission Services) | |
| RE | Restricted to a group specified by the consortium (including the Commission Services) | |
| CO | Confidential, only for members of the consortium (including the Commission Services) | |

# Contents

# 1 Outline of the deliverable

This deliverable deals with the description of the communication transceiver units, protocols and software. Detailed definition of the protocols, communication sequences and synchronization procedures are reported.
The overall software architecture is designed, defining all the information and messages that the single agents (underwater segment, surface segment, diver) have to share in order to fulfill the project objectives.
Simulation tools are currently under development in order to implement delayed and loss-affected communications, thus validating the protocols and sequences to maintain the connectivity among the agents.

# 2 Design, construction and testing of transceiver units

UNEW has contributed to the communication related task with prototype low data rate (100bps) modems and navigation systems, initially delivered to Zagreb and integrated with surface and underwater segments. After successful initial trials, similar devices are going to be constructed for CNR and IST in the following months, to proceed with the integration with the related partner vehicles. The initial design of the 'Seatrac' modem/USBL device utilises a free flooded transducer ring, with a four element USBL array. The enclosure is machined from 316 grade stainless steel and depth rated to up to 3000m. Images of the prototypes are shown in Fig. 3.1a) and Fig. 3.1b), the X110 model shown on the left is a modem design offering bidirectional communications. The X150 model additionally offers navigation functionality, with the inclusion of the four USBL elements. The dimensions of the USBL, X150, model are shown in Fig. 3.1c), the small form factor of the unit is ideally aimed towards the deployment on the surface vehicle, the diver or 'buddy' vehicle.


*Fig. 3.1. X110 Modem Prototype, b) X150 USBL Prototype, c) X150 Modem Line Drawing*

Compared with systems previously used by UNIZG-FER on the surface segment and diver (Tritech MicronNav), the hardware required to compute an accurate USBL fix is fully integrated into the modem unit, reducing the need for any additional topside equipment or software and reducing space, weight and power. The new units also benefit from improved transmit circuitry and transducer design. The new hardware configuration supports higher output power (172dB compared with 167dB) and broader transmission bandwidth as shown in figure 3.2. The current signal design occupies 8 kHz bandwidth on a centre frequency of 28kHz but broader band transmission would be supported by the transducer in future to increase data rate and/or spread spectrum processing gain. This, together with higher data rate (100bps from 40bps), more efficient protocols and robust error checking lead to enhanced data throughput and higher position update rate than previously possible. The acoustic protocol developed for the new modems offers the capability to support varying payload lengths (up to 32 bytes) and the ability to request a USBL position fix on every packet (if required). This enables more flexibility in networking between the ASV, AUV and diver.

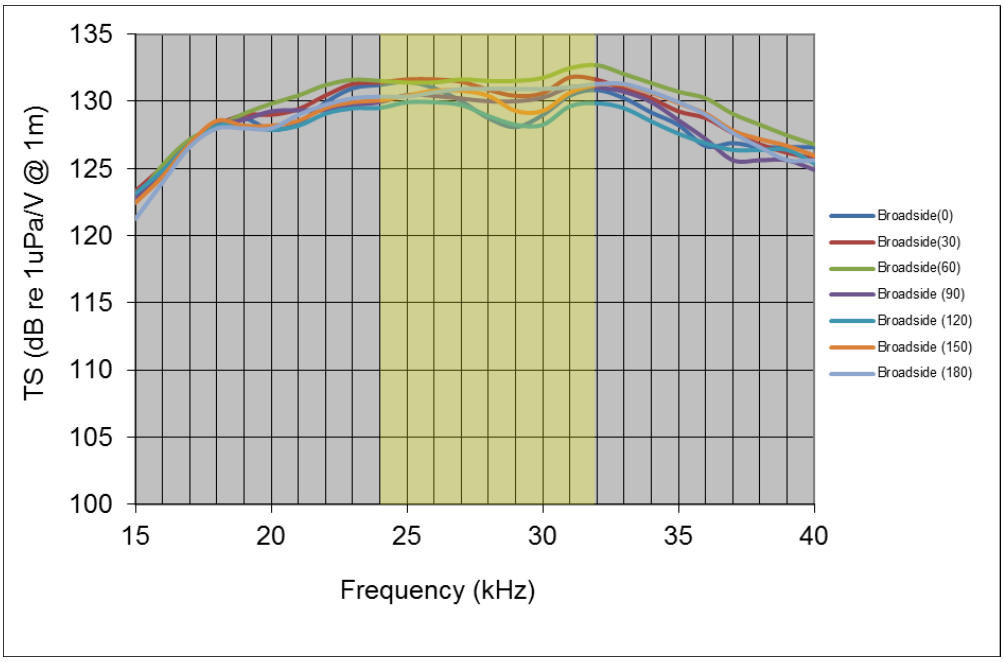


*Fig. 3.2 – transmit sensitivity of new Seatrac modem transducer*

Trials of the new modem are ongoing with experiments carried out by UNEW, in test tanks, lake Windermere and large seawater docks, and UNIZG-FER in the Adriatic. Communication reliability has proven very high to date, with very few dropped packets over ranges up to 1.5km. Ranging performance is also very pleasing, with repeatability of the order of 10-20cm at all ranges. Current work is focussing on quantifying and optimising accuracy and repeatability of USBL acoustic fixes (Azimuth and Elevation) and optimising the performance of the in-built AHRS system.

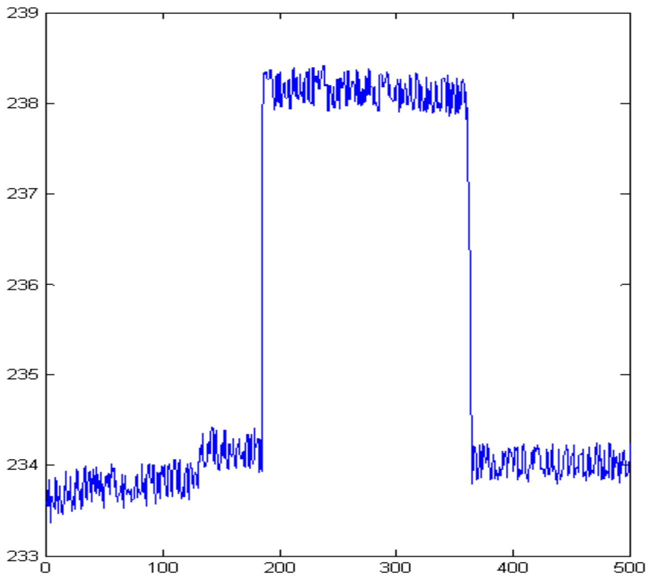



*Fig 3.3 - Bearing estimates showing repeatability in two different locations.*

Figure 3.3 shows the results of a tank test showing the repeatability of bearing estimates in two different positions with realistic received SNR, with sub 1 degree stability achieved. Figure 3.4 shows the Seatrac modem deployed on UNEW's LBV vehicle in lake Windermere and figure 3.5 shows the resulting output from the Seatrac navigation GUI as the vehicle is navigated in a loop around the Seatrac USBL deployed from a small dingy. Jitter on the trace is largely due to motion of the dingy/USBL but these results are very encouraging and suggest that new miniature USBL system is capable of delivering effective navigation data. On-going experiments with precisely controlled geometry, will quantify the absolute accuracy of the system and provide data for further enhancement of the position fix algorithms.

*Fig 3.4 – Seatrac modem deployed on Seabotix ROV*

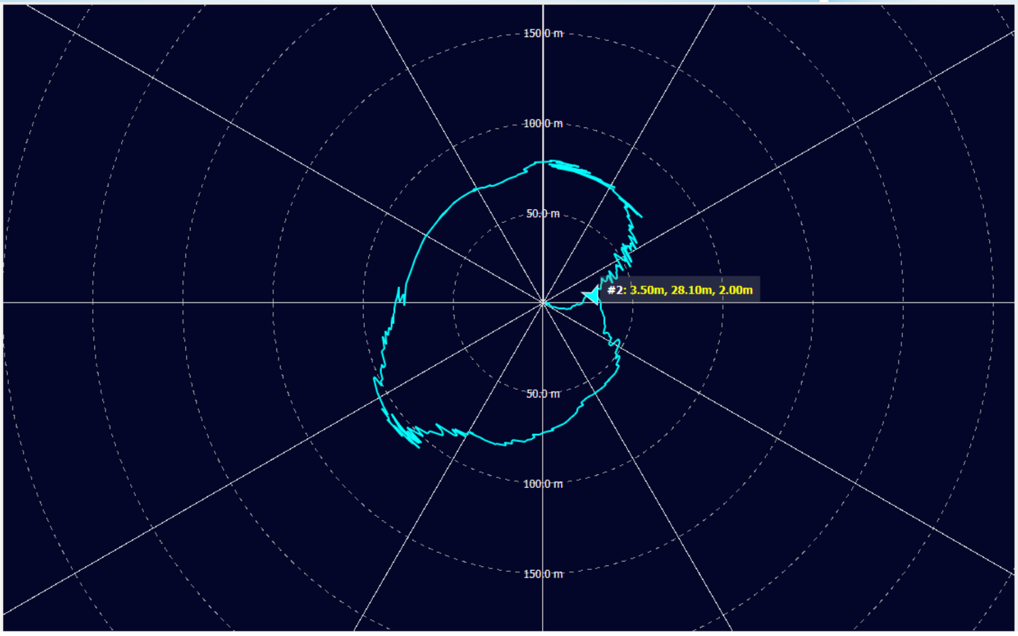



*Fig 3.5 – position trace as vehicle completes loop around USBL*

A serial command interface has been developed and fully documented to enable UNIZG-FER and other partners to operate the new modems and USBL from within a ROS architecture. This has enabled the successful integration of the system by UNIZG-FER for recent trials and demonstrations. Figure 3.6 shows positions estimated by the USBL equipped Seatrac unit as another unit is moved around the perimeter of a shallow, concrete lined water polo pool (50m x 25m). This is an extremely challenging environment for both acoustic communication and positioning due to the severe reverberation conditions but the system maintains communication and accurate positioning around the whole perimeter.


*Fig 3.6 – position overlay on google earth for perimeter of concrete lined pool 50m x 25m*

## 3 Development of higher data rate mode

In parallel, UNEW have also made good progress on the development of a higher data rate modulation scheme and receiver algorithm for deployment on the Seatrac modems (working towards D 1.2.2). This uses direct sequence spread spectrum modulation, in combination with an adaptive receiver, and powerful error correction codes to achieve data throughput of around 1.5kbps in very severe channel conditions. Figure 4.1 shows a block diagram of this scheme.

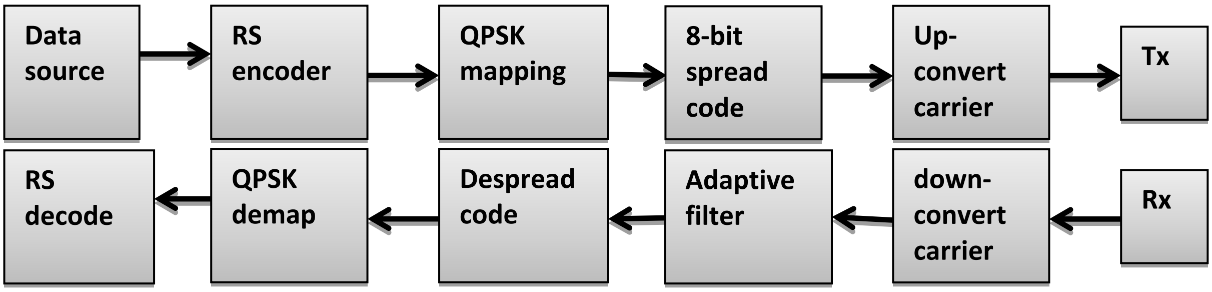
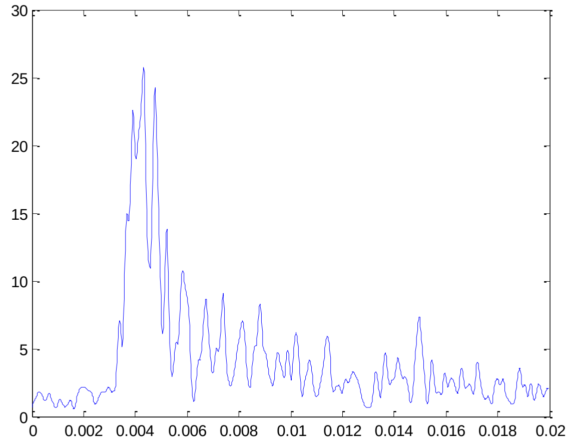


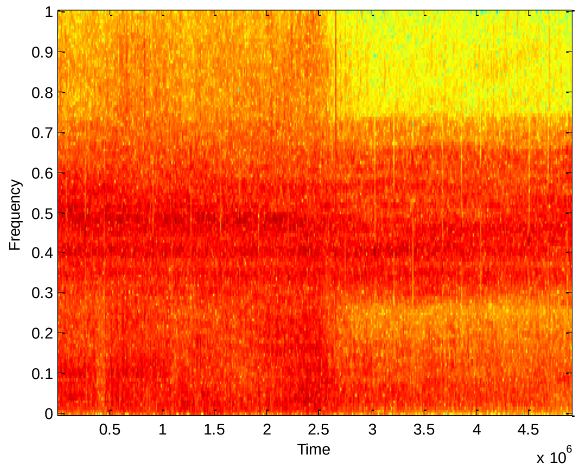

*Figure 4.1 – structure of higher bit rate (1.5kbps) spread spectrum transmission scheme*

Experiments have been carried out in the Tyne estuary over a distance of 1.5km to investigate the performance of this scheme. The nature of this channel leads to very severe multipath conditions and frequent high noise due to busy shipping traffic. Figure 4.2 (a) shows a typical example of a channel impulse response observed during this experiment, with numerous multipath arrivals spanning at least 15ms duration. Test data was transmitted in packets of 2048 bits and Figure 4.2(b) shows the spectrogram of the received signal over the period of 200 packet transmissions. This shows our signal between 8-16kHz and very low SNR during the first half of the transmission due to shipping traffic. Figure 4.2(c) shows the received SINR (signal to interference and noise ratio) during this period and finally figure 4.2(d) shows the bit errors observed in each of 200 packets, with an average bit error rate of 0.004 achieved before coding. After the RS decoder is applied all of these errors are corrected and a total of 300kbits is transferred without error in a period of just over 200 seconds.

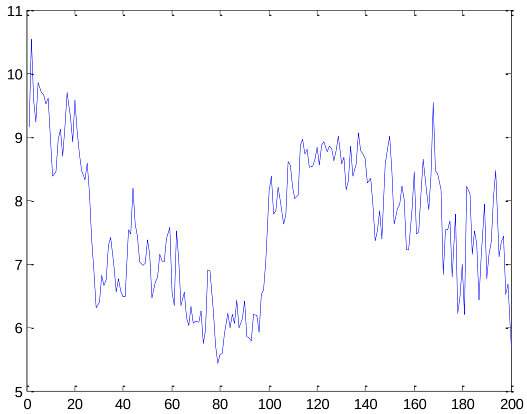
The performance of this scheme is very impressive, given the severe channel conditions, and the complexity of the receiver structure is relatively low compared to other possible schemes. Work during Q4 of the project will focus on implementing and testing this scheme on the Seatrac hardware platform in the 24-32kHz acoustic frequency band.



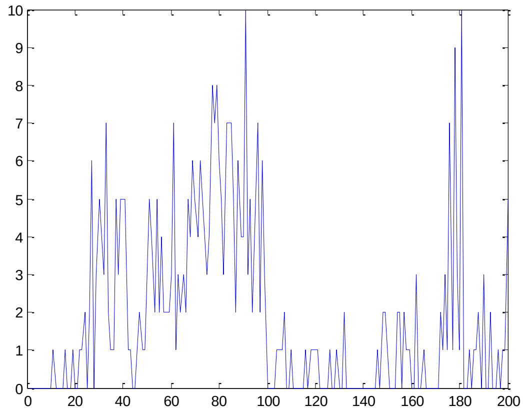

*(a) Typical channel impulse response*




*(b) Spectrogram showing noise and fading spectrum*



*(c) SINR observed over 200 data packets*



*(d) Bit errors observed in 200 data packets*

*Fig 4.2 – Results of Tyne estuary transmission experiments with 1.5kbits/s spread spectrum scheme*

# 4 Communication protocols

A first draft of the requirements for the communication protocol that will be used in the underwater segments is proposed by CNR.
Given the harsh environment, the scarce bandwidth and the high latency of acoustic links, a very careful use of communication resources is mandatory for the communication between the underwater segments. For these reason, the development of a new custom real-time protocol is being taken into consideration, as well as the use or adaptation of already existing low-level real-time protocols such as ORTE, Open Real-Time Ethernet, an open source implementation of Real-Time Publish-Subscribe (RTPS) communication protocol. The resulting protocol will need to implement and enforce a few requirements, such as real-time performance, priority and quality of service (QoS), platform independence and entropy maximization of transmitted data, while may also provide additional beneficial features such as error recovery of received data and further message compression.

## 4.1 Acoustic MAC layer specification

The initial Seatrac transceiver devices use UNEW's orthogonal linear frequency modulation (OLFM) spread spectrum signalling to achieve highly reliable communication at 100bps data rate. An efficient MAC layer protocol has been developed to make maximum use of this and to facilitate the CADDY messaging and positioning protocols described in later sections.

Acoustic messages always start with a synchronisation chirp (50ms linear frequency sweep) followed by a header field containing information about who sent the packet, who should receive it, how it should be responded to and the payload it contains. After the header field and optional data payload may be included that defines additional content used by higher layers in the protocol stack. Finally, after the header and payload (if included), a CRC16 checksum is used to ensure data integrity of the message.

Data values are always transmitted most-significant-bit first and, in header and checksum fields, most-significant-byte first (Big Endean). Optionally, certain message types require the packet being terminated with a 50ms USBL chirp to allow for positioning information to be computed. An acoustic message uses the packet structure shown in figure 5.1.

| | | | | | |
|---|---|---|---|---|---|
| **SYNC Chirp** | **Message Header** *(15 bits)* | **Payload Header** *(optional - 5 bits)* | **Payload Data** *(up to 31 bytes)* | **Message Checksum** *(16 bits)* | **USBL Chirp** *(optional)* |

*Fig 5.1 – Acoustic packet structure (10ms per data bit)*

| *Bits 14:11* | *10:7* | *6:4* | *3:0* |
|---|---|---|---|
| **DEST_ID** *(4 bits)* | **SRC_ID** *(4 bits)* | **MSG_TYPE** *(3 bits)* | **PAYLOAD_ID** *(4 bits)* |

*Fig 5.2 – Message header fields*

The message header is a 15-bit long field (figure 5.2) that contains the following fields, and is transmitted MSB first.

- **DEST_ID** DEST_ID is a numeric identifier representing a beacon address (Beacon ID) within the current communications setup, that the packet is intended for.
  On reception, the receiving beacon will examine the DEST_ID value first, and reject any messages not intended for it (aborting subsequent reception and shutting down the receiver until the next SYNC chirp).
  The special case value of "0x0" is used to specify "broadcast to all" packets that will be decoded by all beacons.

- **SRC_ID** SRC_ID is a numeric identifier that specifies the address (or ID) of the beacon that is sending the packet.
  The special case value of "0xF" is reserved specifies the source beacon is considered "anonymous" and reserved for future use. (Possibly for protocols that automatically allocate a beacon ID address on the fly).

- **MSG_TYPE** The MSG_TYPE field specifies how the receiver should handle the message. Valid types are…
  - 0x0 – **OWAY**
    A One-Way message that requires no response.
  - 0x1 – **OWAYU**
    One-Way message that requires no response, and is sent with a USBL chirp appended.
  - 0x2 – **REQ**
    A request message that requires a response (RESP) message in reply.
  - 0x3 – **REQU**
    A request message that requires a response message (RESPU) to be sent with a USBL chirp appended (for positioning).
  - 0x4 – **RESP**
    A response message sent when prompted by the reception of a request (REQ) message.
  - 0x5 – **RESPU**
    A response message sent when prompted by the reception of a USBL request (REQU) message. The response has a USBL chirp appended to allow position of the responder to be computed.

- **PAYLOAD_ID** The PAYLOAD_ID field is a 4-bit value specifying the contents (protocol) of the messages payload. Value 0x0 is reserved to indicate the special "PING" case, where no Payload Header or Payload Data fields are sent (allowing the shortest possible message to be sent)

The Payload Header is sent in all cases EXCEPT where the PAYLOAD_ID field in the message header is "PING" (value '0'), in which case it is omitted.

The Payload Header contains one single field defined as:

- **PAYLOAD_LEN** A 5-bit value that specifies how many bytes are encoded in the payload field (that follows the header).

  Valid values are from 0 bytes to 31 bytes.

  NB: For best performance, the packet length should be kept as short as possible (between 16 and 20 bytes max).

The message payload is a sequence of bytes that are used by higher levels in the acoustic protocol stack to transfer data and implement functions. The payload can contain up to 31 bytes, but for best acoustic performance (both network speed and to reduce Doppler issues) the packet length should be kept as short as possible (around 16 to 20 bytes max). As with the Payload Header, Payload Data is sent in all cases EXCEPT where the PAYLOAD_ID field in the message header is "PING" (value '0'), in which case it is omitted.

To validate data integrity, all acoustic packets finish with a 16-bit checksum. The checksum is computed from all preceding data in the packet. The checksum uses the CRC-16[1] generator polynomial of $x^{16} + x^{15} + x^{2} + 1$ (with numeric msb-first polynomial representation of 0x8005 and reversed representation of 0xA001). The computed checksum is transmitted as a 16-bit sequence, MSB first.

### 4.2 CADDY protocol requirements

The underwater communication protocol is characterized by the following features:
- *Soft real-time performance*: high priority messages are delivered inside a well defined time frame (ideally on scheduled time) and low-priority messages, even in flooding conditions, shall not delay high-priority messages. The most efficient solution could be to program as much as possible of the protocol's details inside the network driver, to give it the full control of the process of frame generation and reception. This kind of solution, however, is costly and not flexible, so a trade-off might be necessary. The level of real-time performance of the underwater communication protocol should be a performance index of the whole software architecture.
- *Prioritization and QoS*: each message passed to the underwater communication stack must have its own priority tag (usually an integral) that assigns the message to a specific class of service (e.g. 1 = alarms, 2 = diver's and RECUV telemetry, 3 = diver's gestures, …, 10 = environmental data, …, 16 = multimedia content). Then, the priority queuing system generally proceeds to send the highest priority message available, or, in certain conditions and with the purpose of granting a certain amount of queuing fairness, may proceed to send a lower priority message. If any amount of queuing fairness is present, it's impact on soft real-time performance must be assessed.
- *Platform independent*: given the variety of platforms available today, this feature is becoming a true technical challenge. The underwater network protocol must be able to connect a wide range of devices, including SBCs (Single-Board Computers) and mobile devices (e.g. tablets or smart phones)

[1] Sometimes referred to as CRS-16-IBM or CRC-16-ANSI. Further details can be found at http://en.wikipedia.org/wiki/Cyclic_redundancy_check

and need to have real-time performance on all of them. This also leads to the requirement of a low memory footprint, especially if the protocol will be partially implemented as a network driver.

- *High entropy*: to maximize the efficiency of the transmission over an acoustic link, each sender must pass to the protocol stack only the minimum amount of symbols needed to represent the data to be transmitted. This means that the sender, who is aware of the meaning of the data that he is about to send, must maximize the entropy (meant as information content) of the message. For example, if a source of information periodically produces a 16-bit value representing the water temperature around the diver, the message will not be coded using 2 bytes. The dispatcher of the message, knowing that the temperature may range from (e.g.) -18 °C to 45 °C, will code the information in a 6-bit field, so that the underwater communication protocol can pack more messages together and minimize the amount of messages (and consequently packet headers) to be sent.

- *Error recovery*: to protect from acoustic noise in the environment (e.g. air bubbles), packets can be padded with checksums or even error recovery codes, such as Reed-Solomon, Low-density parity-check codes or Turbo codes, able to detect and correct multiple random-symbol errors. The use of one of these algorithms, presently used in DVB standard for the satellite transmission of digital television or in 3G and 4G mobile telephony standards, increases the size of the packet, but may prevent the retransmission of a corrupt packet where up to *n* bits are corrupt. The amount of memory and CPU required by the algorithm must be also evaluated before the use of error correcting codes inside the underwater communication protocol.

- *Compression*: to increase even more the entropy of the message that is about to be sent, the underwater communication protocol can make use of one of the many compression algorithms available today. A compression algorithm used as last step before sending the message may shorten low-entropy data such as text strings but will be probably ineffective against data whose entropy has already been maximized by the sender. The use of such protocols may be made subject to the type of message to be sent, and their computational and memory costs must be assessed.

### 4.3 Addressing of nodes

Given that the underwater communication protocol communicates through a shared medium (the water), it may not be necessary to address nodes for each packet sent. For example, high-priority real-time messages may be simply broadcast to all nodes, so that important or continuously updated information is sent to all listening nodes at once.

Conversely, low-priority data should be probably sent to specific recipients only, so some form of addressing of nodes will be necessary. Again, in order to maximize the entropy of the information, a few bits will be sufficient to address all nodes in the system. Particular attention will be given to unreachable nodes, namely nodes that, due to the distance or high noise levels can't be reached directly by the sender. A typical case can be the diver's device unable to directly transmit to the surface vehicle(s): in this case one node (the RECUV) must be used as relay to retransmit the packets between the unreachable nodes.

## 5 Navigation data size and quantization

Although quantization could be variable, the following quantization is initially investigated for navigation data transmission. The USBL has already some features to transmit navigation data to 0.1m and 0.1° precision between devices. Optionally, remote (interrogated) transponder depth with resolution 1m can be transmitted with the return USBL package.

| Data type | Size / bits | Range | Quantization |
|---|---|---|---|
| Speed | 4 | [0,1] m/s | ~0.06 m/s |
| Heading/Course | 10 | [0,360]° | ~0.35° |
| Depth | 7 | [0,64] m | 0.5 m |
| Position offset x/y | 10 | [-51.1,51.2] m | 0.1 m |
| Lat position init | 22 | [0,180]° | 1'' |
| Lon position init | 22 | [0,360]° | 1'' |

What was used before for diver tracking and communication (legacy):

- Geographical position is encoded as: $DDDM_1M_2m_1m_2m_3m_4$
- For Latitude +90° is added to have 0-180°
- For Longitude + 180° is added to have 0-360°
- Chat message are sent with a 6bit/character (a-z, 0-9, “.”, “,”, "?", "!", " ", ..., “\n”, “\0”)

| Data type | Size / bits | Range | Quantization |
|---|---|---|---|
| (Lat/Lon init) $DDDM_1M_2m_1m_2$ | 22 | [0° 0’ 00’,180° 59’ 59’’] | 1’’ |
| (Lat/Lon minimum) $m_3m_4$ | 7 | [0.00’’,0.99’’] | 0.01’’ |
| (Lat/Lon large) $M_2m_1m_2m_3m_4$ | 18 | [0’ 00.00’’, 9’ 59.99’’] | 0.01’’ |
| (Lat/Lon medium) $m_1m_2m_3m_4$ | 13 | [00.00’’, 59.99’’] m | 0.01’’ |
| (Lat/lon kml) $m_1m_2m_3$ | 10 | [00.0’’,59.99’’]° | 0.1’’ |
| Predefined messages | 5 | [0,31] | - |
| Checksum | 6 | [0,63] | - |

## 6 Interrogation scheme

UNIZG-FER proposes the following interrogation sequence that is going to be implemented in the overall CADDY architecture and that can be applied to each different platform composing the actual robotic formation.

Assumptions:

- PlaDyPos and Buddy have a USBL device
- diver has a Modem
- All devices can hear each other's transmission
- sonars are optional

The concept is shown in Figure 7.1.

**PlaDyPos -> Diver -> PlaDyPos (AT1+AT2/PT1+PT2)**

(AT1) PlaDyPos pings the diver and waits for a reply. The ping payload can be variable and can contain init data. The diver replays on the USBL ping and receives payload data.
(PT1) Buddy receives the possible payload data in the USBL ping and registers that the Pladypos has performed a ping.
(AT2) The diver modem replies with the measured heading and depth. Diver payload (if any) is sent with the navigation data. Pladypos receives the diver USBL measurement and additional information which can be used to update the diver kinematic estimate.
(PT2) Buddy receives the diver information by passive listening.

**Buddy -> PlaDyPos -> Buddy (AT3+AT4/PT3+PT4)**

(AT3) Buddy ping PlaDyPos and sent its kinematic information. PlaDyPos receives the data and updates the model
(PT3) The diver receives kinematic data from PlaDyPos by passive listening.
(AT4) PlaDyPos replies with the position and previously updated diver position (in AT1-AT2). Buddy receives the data and updates its position estimate.
(PT4) The diver gets a new update too and the PlaDyPos position for a potential estimation update.

**Buddy->Diver->Buddy (AT5+AT6/PT5+PT6)**

(AT5) Buddy interrogates the diver with a USBL ping and sends its updated position estimate. The diver receives the kinematic information of Buddy for the updated of a kinematic filter.
(PT5) PlaDyPos hears the Buddy position and updates the kinematic model estimate.
(AT6) Diver replies with the usual payload. Buddy receives a USBL fix and updates the diver position.
(PT6) PlaDyPos receives diver information by passive listening.

**PlaDyPos->Buddy->PlaDyPos (AT7+AT8/PT7+PT8)**

(AT7) PlaDyPos interrogates Buddy with a USBL ping sending a variable payload with possible navigation information. Buddy receives a potential data update from PlaDyPos

(PT7) The diver receives PlaDyPos message payload.
(AT8) Buddy answers with the updated position estimate of the diver and potentially its kinematic data.
(PT8) The diver hears its position for a new update of the position and possibly to update the Buddy kinematic model.

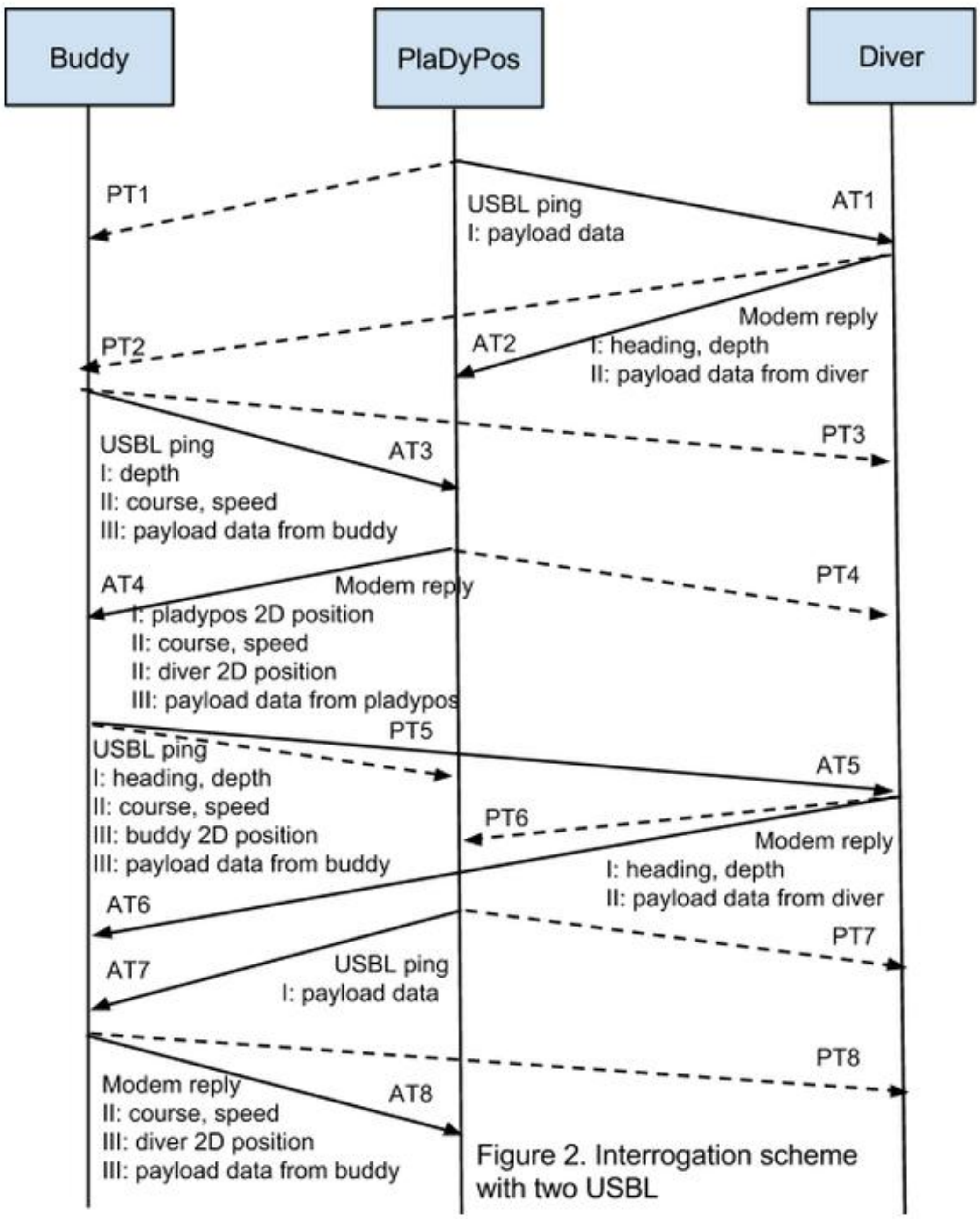



*Fig.7.1. 3-agents communication sequence*

## 6.1 Messages and message sizes

Assumptions:

- Minimum turn around time without data = 1.3s
- Minimum turn around for two modem comms = 0.3s
- Additional data bytes add 80 ms/byte to the transmission length
- Range quantization is 0.1-0.5m (depending on message type) - 2bit flag
- Depth quantization is 0.5 - 1m (depending on message type)

Diver position updates should have better quantization and occur more frequently than Buddy position updates. Buddy is assumed to have a DVL and can therefore operate longer without position updates. While on surface data between agents is transmitted wirelessly. The data encoding can be done depending on the operating distance of the diver and buddy and changes in position. If a platform is performing DP and the error in positioning is less than 0.5m the platform position does not need to be resend. Operation with medium horizontal distances (+/- 51.2m at 0.1m) with respect to buddy and Pladypos.

PlaDyPos pings the Diver (no navigation, with potential payload)
Diver transmits modem reply on interrogation:

| Data type | Size(bits) | Range | Quantization |
|---|---|---|---|
| Message ID | 4 | 0-16 | 16 levels |
| Heading | 10 | [0,360] | @0.35° |
| Depth | 7 | [0,64] | 0.5m |

Buddy pings PlaDyPos

| Data type | Size(bits) | Range | Quantization |
|---|---|---|---|
| Message ID | 4 | 0-16 | 16 levels |
| Depth | 7 | [0,64] | 0.5m |
| Buddy course | 10 | [0,360] | @0.35° |
| Buddy speed | 4 | [0,1] | @0.06 m/s |
| Diver 2D offset to Buddy/INIT* | 20 | [-51.2,51.2] m | @0.1m |

* Note that this position is sent when the diver can be seen in the sonar (it can be reduced to 16 bits with [-12.8, 12.8] m relative to buddy since the sonar is 15m range)

PlaDyPos replies

| Data type | Size(bits) | Range | Quantization |
| --- | --- | --- | --- |
| Message ID | 4 | 0-16 | 16 levels |
| Quantization (optional if adaptive quant) | 2 | 0-3 | 4 levels |
| PlaDyPos 2D offset to INIT | 20 | [-51.2,51.2] m | @0.1m |
| PlaDyPos course | 10 | [0,360] | @0.35° |
| PlaDyPos speed | 4 | [0,1] | @0.06 m/s |
| Diver 2D offset to PlaDyPos/INIT | 20 | [-51.2,51.2] m | @0.1m |

Buddy interrogates Diver

| Data type | Size(bits) | Range | Quantization |
| --- | --- | --- | --- |
| Message ID | 4 | 0-16 | 16 levels |
| Depth | 7 | [0,64] | 0.5m |
| PlaDyPos 2D offset to INIT | 20 | [-51.2,51.2] m | @0.1m |
| Buddy course | 10 | [0,360] | @0.35° |
| Buddy speed | 4 | [0,1] | @0.06 m/s |
| Buddy 2D position relative to INIT | 20 | [-51.2,51.2] m | @0.1m |
| Diver 2D offset to INIT | 20 | [-51.2,51.2] m | @0.1m |

Diver replies:

| Data type | Size(bits) | Range | Quantization |
| --- | --- | --- | --- |
| Message ID | 4 | 0-16 | 16 levels |
| Heading | 10 | [0,360] | @0.35° |
| Depth | 7 | [0,64] | 0.5m |

PlaDyPos pings only
Buddy replies with diver position

| Data type | Size(bits) | Range | Quantization |
|---|---|---|---|
| Message ID | 4 | 0-16 | 16 levels |
| Diver 2D offset to INIT | 20 | [-51.2,51.2] m | @0.1m |

| Message | Size(bit) | Transmit Time (s) |
|---|---|---|
| PlaDyPos -> Diver | 0 | 1.3 (USBL) |
| | | 1.28 |
| Diver->PlaDyPos | 21 | 0.24 (Data) |
| | | |
| Buddy-> PlaDyPos | 25 (45) | 1.54 (1.78) (USBL+Data) |
| PlaDyPos -> Buddy | 60 | 0.6 (Data) |
| Buddy -> Diver | 45 (65) | 1.78 (1.94) (USBL+Data) |
| Diver -> Buddy | 21 | 0.24 (Data) |
| PlaDyPos -> Buddy | 0 | 1.3 (USBL) |
| Buddy -> PlaDyPos | 24 | 0.24 (Data) |
| | | 7.24 (7.64)* |

* The green times are when the Buddy observes the diver in sonar or camera and can send a updated position.

When Buddy only gets the diver position with the USBL it makes no sense to send the diver kinematic model estimate in between the pings since the diver computer can do that himself.

# 7 Software & communication architecture

The CADDY software architecture, which is based on the ROS system, is mainly composed by the following five interconnected elements:

- diver;
- AUV;
- USV;
- remote control and supervision station;
- environment.

The ***diver*** node can actively interact with the architecture, in particular dialoguing with the AUV (companion/buddy autonomous vehicle - RECUV) in two ways: sending commands through the underwater tablet or performing gestures in front of the AUV camera.
Within the CADDY architecture, the diver node can be simulated through a simple kinematic model (commanded by the user or programmed to execute specific tasks). The node produces all the navigation state variables (namely absolute position and velocity, body-frame velocity). Being the diver equipped with an acoustic modem, the communication channel is used to send **commands** (generated through the tablet) and **status** data structure which is filled with biological measurements (if available) for instance obtained by heart-beat reader, breath sensor, body temperature, etc.

The ***AUV*** is one of the two completely autonomous agents in the framework. Two of the main sensors equipping the AUV are: *i*) the video-camera which is essential to interact on one side with the diver, allowing the gesture recognition, and on the other side with the environment in order to track or avoid collisions with objects in the operative area; *ii*) the USBL system which is used to detect the relative position of the USV and (theoretically) also of the diver (it will strongly depend on the mounting configuration of the devices, the quality of the acoustic communication/tracking depends on the relative position of the pingers/beacons). Through the acoustic channel the AUV transmits the current diver position (if available) and her/his known status, also the body-frame velocity of the AUV itself (the velocity information can be used from the USV to ease the task of cooperative motion or tracking, the AUV position is already known by USV thanks to the USBL measurement and thus it has not to be dispatched) and the operating mode; this latter data is generated by the AUV's mission control system in function of the received commands/gestures.

The ***USV*** is the other autonomous agent of the architecture and its framework-related characteristics are similar to the AUV's ones. It senses the position of the AUV and the diver via the USBL measurements. The USV is directly connected through a wi-fi link to the remote control station, thus the USV can be also exploited as a communication relay to forward information or commands to the underwater segment. The information sent through the acoustic channel are the position of the diver (obtained by the USBL) and the diver status (obtained by information sent by the tablet), the USV also sends it body-frame velocity and operating mode.
The USV and the AUV can dispatch additional information (to be defined) in order coordinate the evolution of the planned mission or to adapt to unexpected situations.

The ***remote control station*** is a supervision point that is used to monitor, first of all, the status of the diver and to track the state of advancement of the mission and/or desired tasks for the robotic platforms. Information can be forwarded to the diver (in turn appearing on the tablet) and commands can be sent to the platforms in order to force a specific operating mode or to update targets, references, etc.

The ***environment*** is a node that has to provide some essential information for the consistency of the framework, as for instance the GPS coordinates of the local origin in such a way that all the nodes can refer their positions to the same inertial frame. A shared clock or time data structure is also needed to synchronize all the nodes and all the variables of the architecture.
Within a simulative framework, the environment will also contain additional nodes and/or data structures in order to represent objects in the area, sea-bottom morphology, physical characteristics involving for instance the quality of the acoustic communication and so forth.

The proposed overall CADDY software architecture is reported in Fig. 8.1.

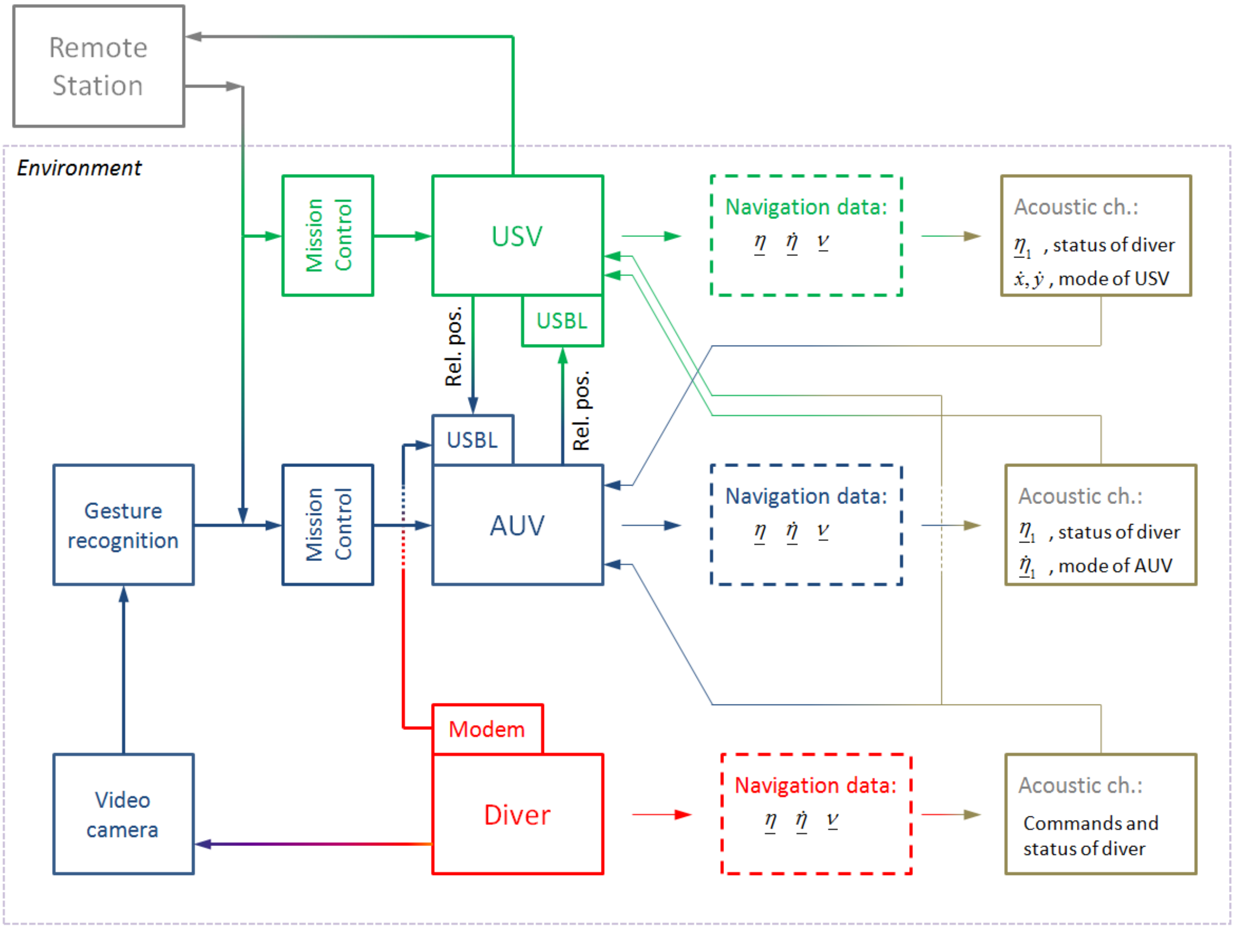



*Fig.8.1. Overall software architecture*

The diagram reported in Fig. 8.2 represents the basic set of software modules to be instantiated and tailored for each particular scenario. The “CADDY Vehicle Wrapper” block represents a generic vehicle to be used for the CADDY project wrapped with the necessary interfacing software such that it complies to the standard interface adopted for CADDY.

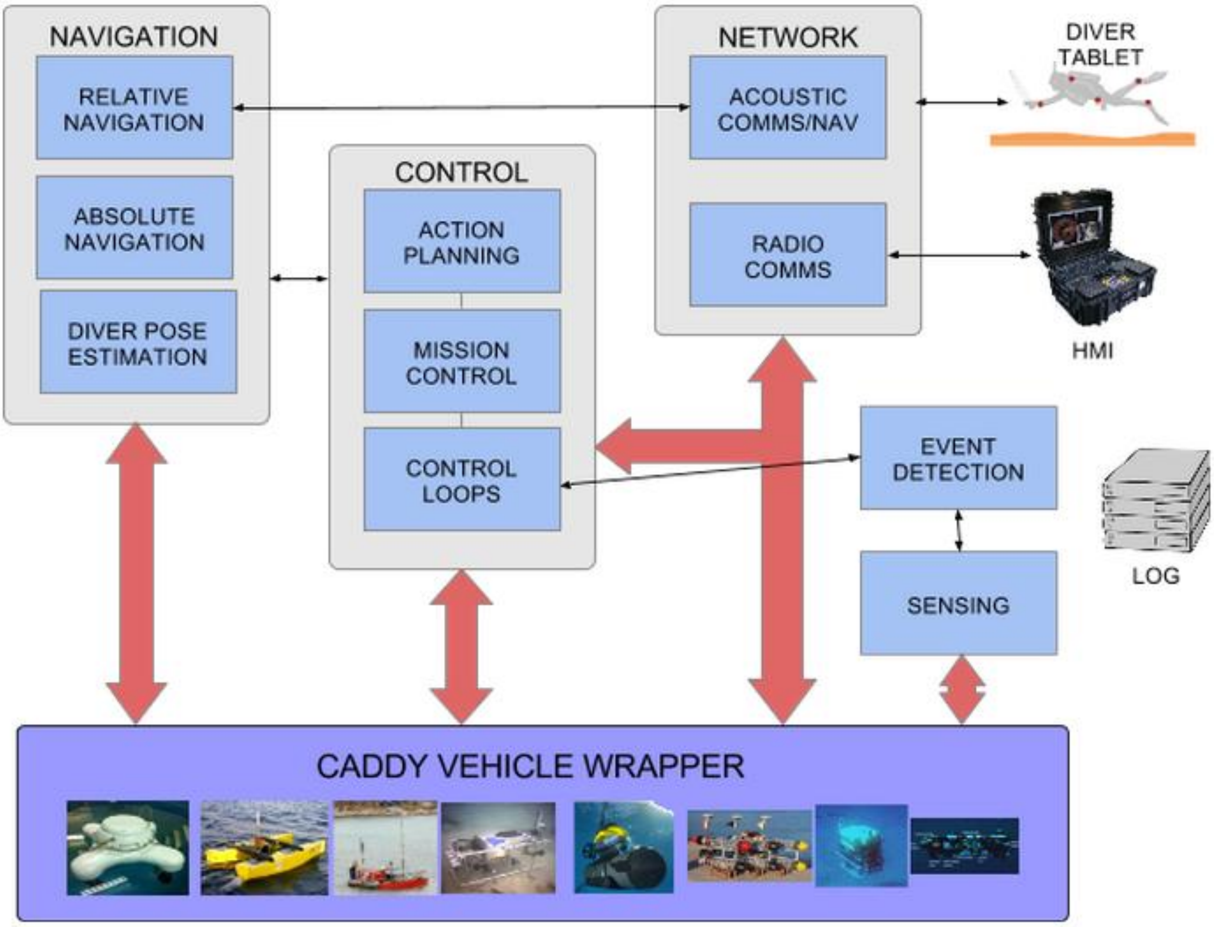



*Fig.8.2. Basic software module scheme*

# 8 Conclusions

This first round of definition and implementation of software and hardware components has been carried out with success and within the expected delivery time. Underwater transceiver units, documentation and the software implementing a first version of the underwater communication protocol have been delivered to UNIZG and tested, demonstrating reliable communication and positioning performance. Similar units are being delivered to the other partners. The first versions of the acoustic modems still have limited bandwidth but their reliability will enable the upper communication layers to establish a basic but working underwater acoustic network.

The next three months until D1.2.2 delivery due date will be devoted to improve both the hardware and the software of the transceivers to deliver much higher data throughput, via the new DSSS modulation scheme, and enhanced positioning accuracy. In the case of the upper layers of the underwater protocol,  the effort will be focused in particular on adaptive quantization schemes able to increase the entropy of the quantities that are being sent through the acoustic link. In turn, this optimization should improve the available bandwidth for the delivery of asynchronous events, ranging from high priority commands and alarms originated by the diver, through to low priority chat messages flowing bi-directionally between the operation centre and the diver.