`NetRadio` plannification and first draft

- Commons functionalities
  - Errors handling
  - Messages related
- Planning
  - Tasks
  - Sprints

Commons functionalities

It's mainly about the commons functions used in both broadcaster and manager.

Errors handling

In the c file:

```c
void print_err(int net_errid) {
    fprintf(stderr, "net: %s", net_errmsgs[net_errid - 1]);
}
```

In the header associated:

```c
/**
 * Assign to each error an "index" associated to the message
 * in the array `errors_msg`.
 *
 */
enum net_errids {
    ACCEPT_FAILURE = 1, // We have to start the index at 1 because net_ok = 0
    ...
};

const int net_ok = 0;
const char* net_errmsgs[] = {"ACCEPT error !", ...};

void print_err(int net_errid);
```

Messages related

```
complete_with_hashes (message: String, len_message: Nat): String
(* complete_with_hashes "RADIO" -> "RADIO###" *)
(* len_message is needed when the message is a mess (140 chars) or an id (8 chars). *)

addr_to_str (addr: Address): String
(* addr_to_str "127.0.0.1" -> "127.000.000.001" *)
```

```
to_str (num: Nat): String
(* to_str 120 -> "0120" *)

is_legal_type (msg: String): Nat
(* is_legal_type "REPO" -> ERR *)

is_legal_format (msg: String): Nat
(* is_legal_format "DIFF 1 1 Hello" -> OK *)
(* Tests if the format is legal depending the type of the message *)

is_legal_msg_receiver (msg: String): Nat
(* (i.e, the receiver is a manager) is_legal_msg_receiver "DIFF 1 1 Hello" -> ERR *)
(* Tests if the format is legal depending the actual receiver of the message.*)

split_on_char (msg: String): List[String]
(* split_on_char "DIFF 1 1 Hello" -> ["DIFF", "1", "1", "Hello"] *)

create_msg (content: String): Message
(* create_msg "DIFF 1 1 Hello" -> {owner_id: 1, num_mess = 1, content: "Hello"} *)

send_msg (fd: Nat, msg: Message): Nat
(* It has to test if the message can be send (depends on the type and the sender) *)
```

Planning

Expected delivering of the project: *03/05/2021* (subject to change).

Tasks

Netlib	Multicaster	Manager	Client	Multicaster & Manager
Implement error handling.	Implement multicast routine. (get list of msg and send them)	Implement getting the multicaster list, the registration of new managers.	Implement a simple client that can receive messages.	Implement commons functions.
Handle message reception and associate to their corre-sponding functionnality.	Implement new messages registration, get last messages sent.	Implement multicaster registration routine.	Implement the posibility to interact with a Broad-caster/Manager and use their functionnalities.	

List of the possible sprints:

- Sprint 1 - **Appropriation of the subject** (11/03 - 18/03) :
  - Planification and commons functions first draft.
  - Diagram and specifications.
  - Setup main project structure.
- Sprint 2 (18/03 - 25/03)
- Sprint 3 (25/03 - 01/04)
- Sprint 4 (01/04 - 08/04)
- Sprint 5 (08/04 - 15/04)
- Sprint 6 (15/04 - 22/04)
- Sprint 7 (22/04 - 29/04)
- Sprint 7.5 (29/04 - 03/05):
  - Final rush with various bugs to fix.
  - Project delivered.