TRABAJO FINAL – RTOS 2 – CESE

GRUPO 1 v12-06-2019

TP1

INTEGRANTES:

Julian Bustamante Narvaez

Jacobo Salvador

Gustavo Paredes D.

Rafael Oliva

Estructura TP1 / Grupo 1 RTOS2

Estructura y diagrama conceptual

Paquetes de datos:

<1B> SOF	<1B> OP	<2B> T	<T BYTES> DATOS	<1B> EOF

Delimitación de paquete: 00 a 99

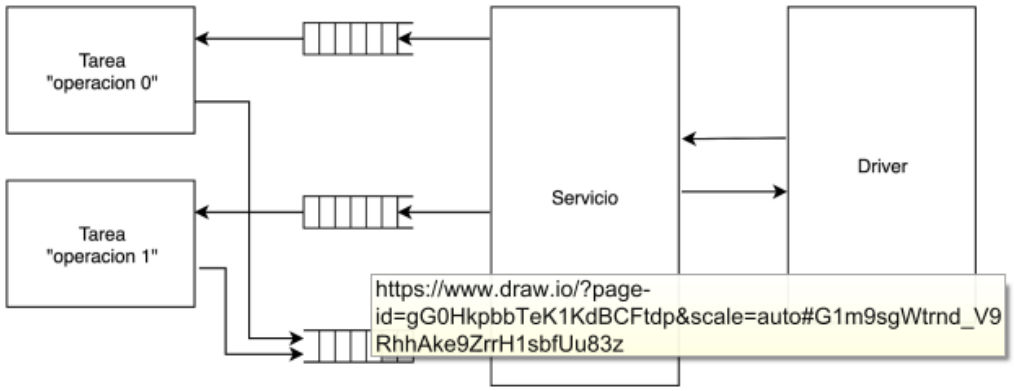
SOF: Carácter '{'
EOF: Carácter '}'.

Campos:

OP (Operación):
0: Convertir los caracteres recibidos a mayúsculas. (CMD/RTA)
1: Convertir los caracteres recibidos a minúsculas. (CMD/RTA)
2: Reportar stack disponible (RTA)
3: Reportar heap disponible. (RTA)

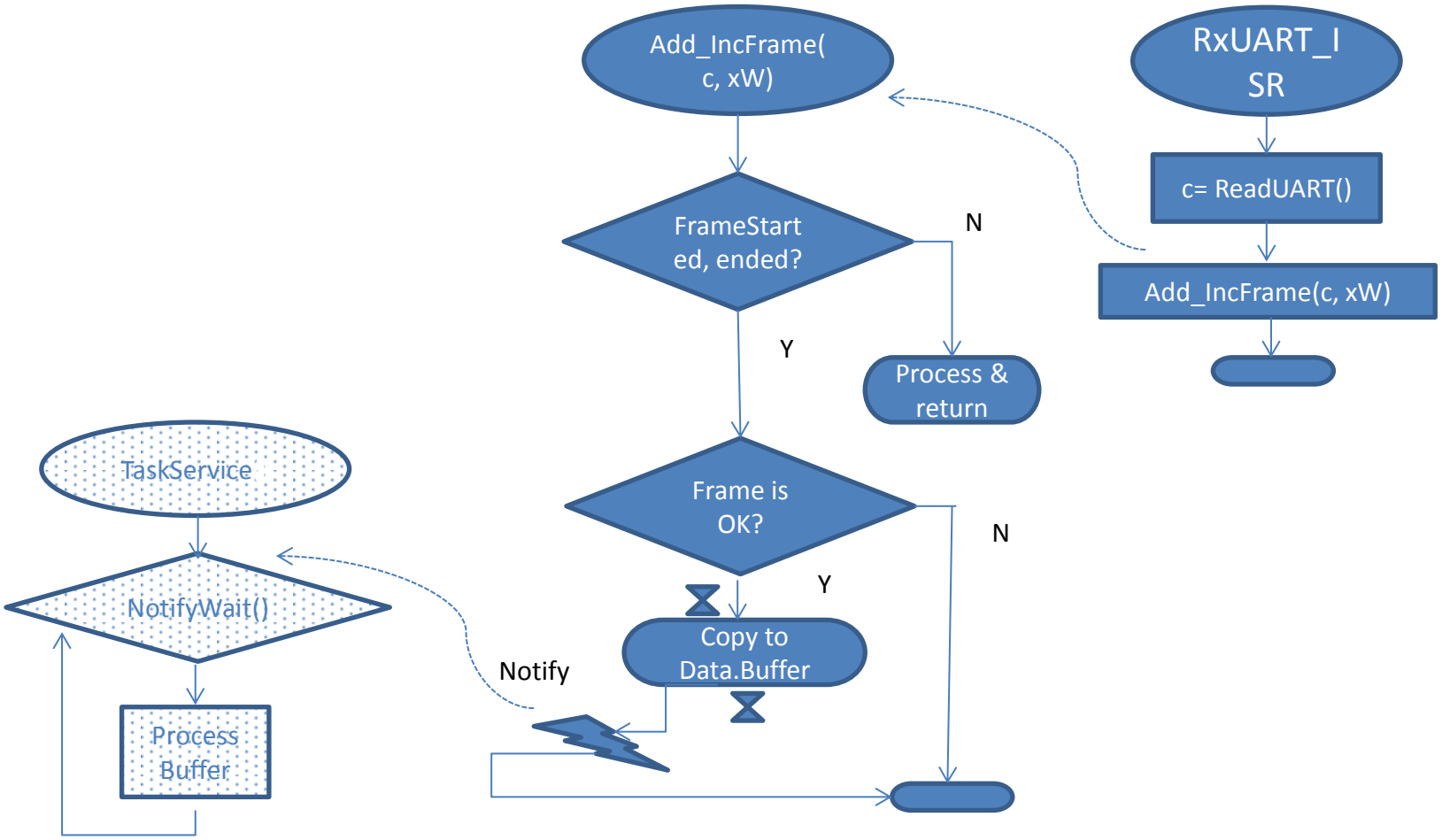
T (Tamaño): El tamaño del campo DATOS. "00" a "99" o sea máximo 99 bytes
DATOS: Texto a procesar. Deben ser caracteres ASCII legibles.

Diagrama conceptual



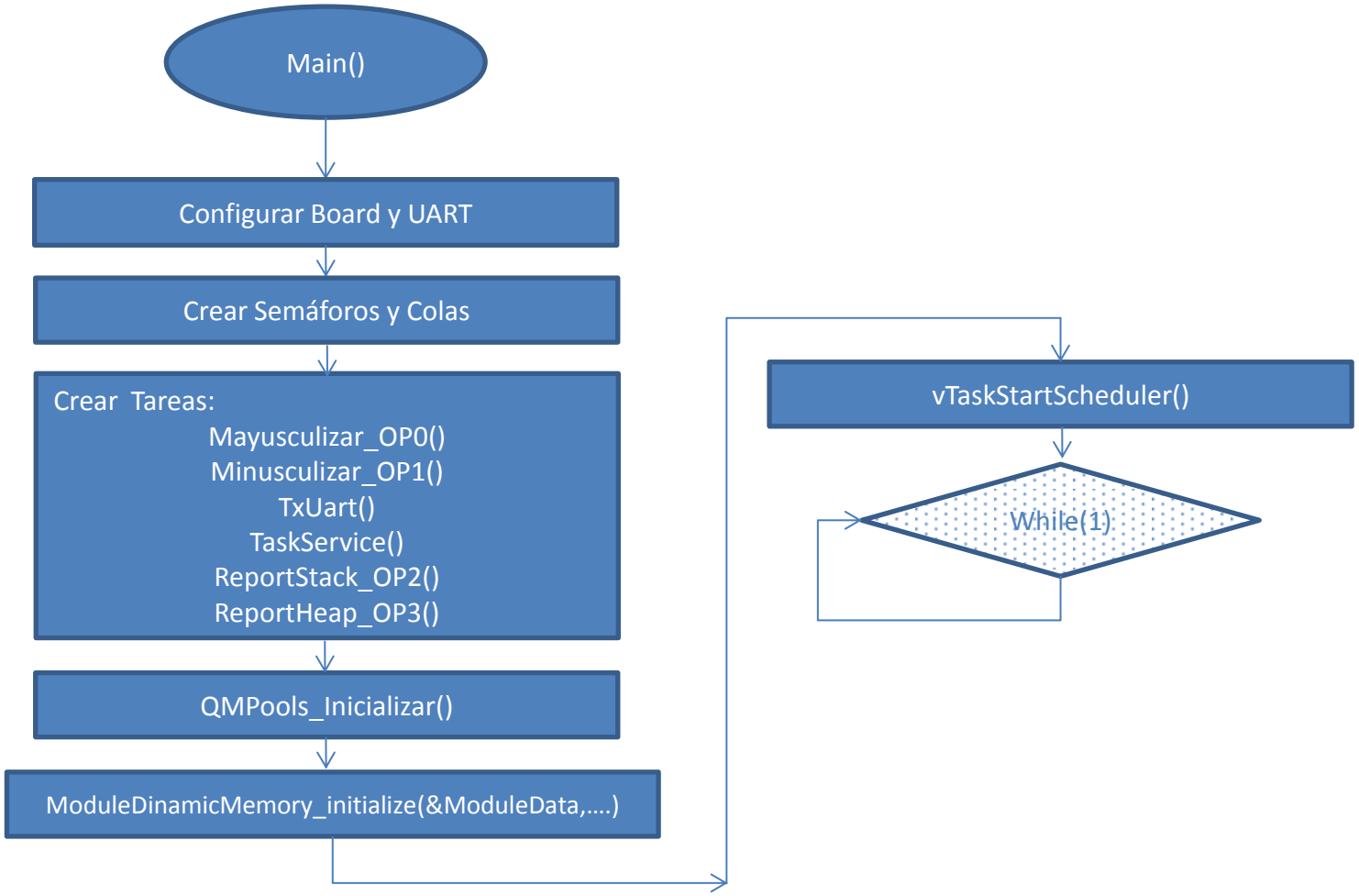
Estructura actual TP1 / Grupo 1 RTOS2

1.a) Ingreso caracteres dde UART v4.6.19



Estructura actual TP1 Main() / Grupo 1 RTOS2

1.b) TP1- Función main()



Estructura actual Main() / Grupo 1 RTOS2

1.b.1) Función main()

```
16
17 int main(void){
18
19     boardConfig();
20
21     QmPoolOrMalloc = eUseMalloc ;//eUseQMPool;
22
23     /*=======Config Uart================================*/
24     uartConfig(UART_USB, 115200);
25     /*Callback interrupt*/
26     uartCallbackSet(UART_USB, UART_RECEIVE, CallbackRx, NULL);
27     /*Habilito todas las interrupciones de UART_USB*/
28     uartInterrupt(UART_USB, true);
29
30     semaphoreCreateAll();
31     QueueCreateAll();
32     TaskCreateAll();
33     QMPools_inicializar();
34
35
36     /*Inicializar Driver memoria dinamica*/
37     ModuleDinamicMemory_initialize(&ModuleData,50,xQueueGenericSend,xQueueGenericSendFromISR
38
39     /* Iniciar scheduler*/
40     vTaskStartScheduler();
41
42     while( TRUE ) {
43     }
44
45     return 0;
46 }
47
```

Estructura de datos TP1 / Grupo 1 RTOS2

1.b.2) Versión TP1 v4.6.19

task_operaciones.c | task_operaciones.c | Task1.c | Task1.h

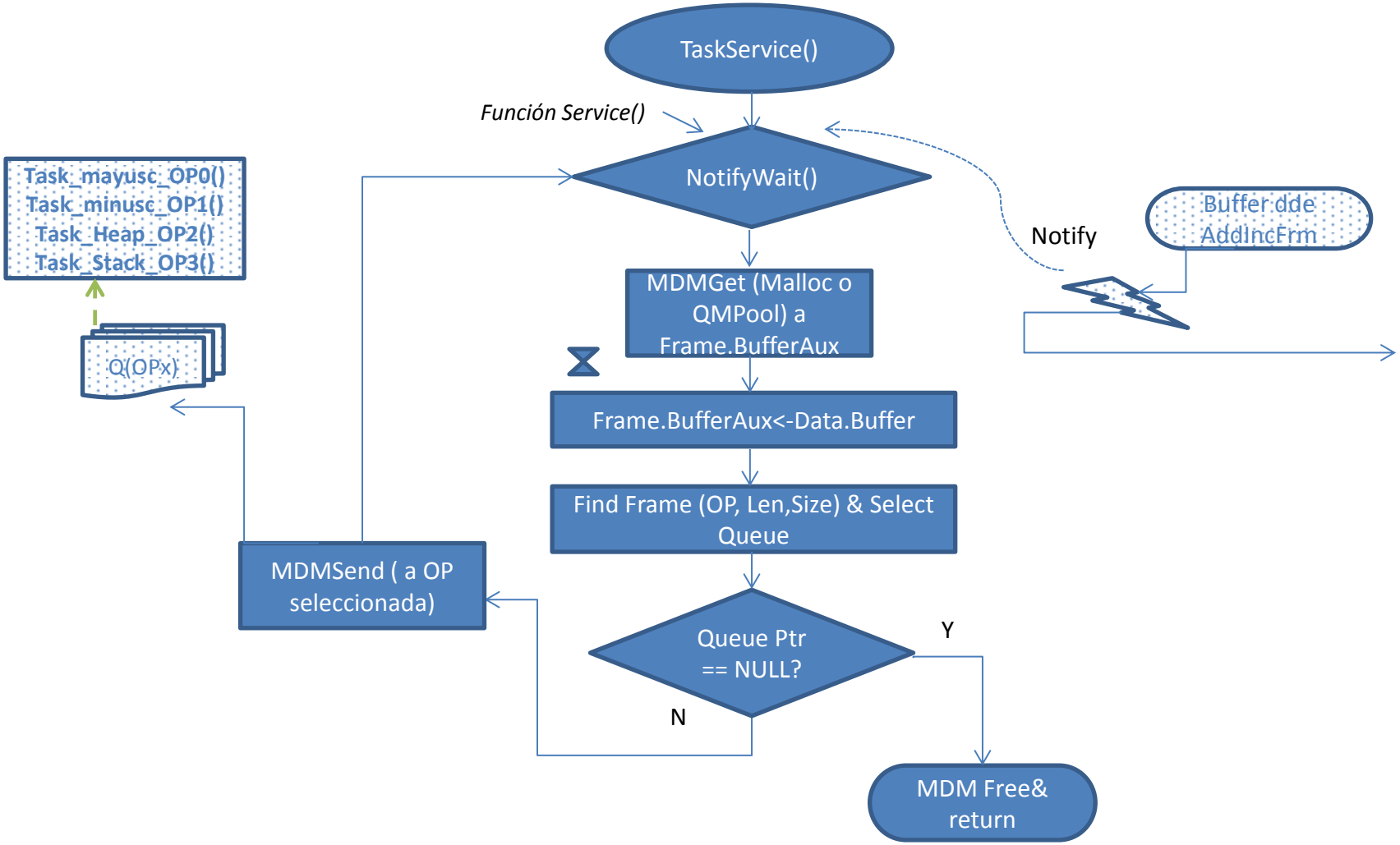
```
35 /** ======= Datos para llenar buffer local=====================================*/
36 typedef struct {
37     char Buffer[106];
38     uint8_t Ready;
39     uint8_t Index;
40     uint8_t StartFrame;
41 }DataFrame_t;
42 extern volatile DataFrame_t Data;
43
44 /** =======Operaciones a usar=================================================*/
45 typedef enum{
46     OP0 = 0,    /* Convertir los caracteres recibidos a mayúsculas. (CMD/RTA)*/
47     OP1,        /*Convertir los caracteres recibidos a minúsculas. (CMD/RTA)*/
48     OP2,        /*Reportar stack disponible (RTA)*/
49     OP3         /*Reportar heap disponible. (RTA)*/
50 }Enum_Op_t;
51
52 /** =======Parametros de la trama de  llegada ================================*/
53 typedef struct {
54     char _SOF;
55     Enum_Op_t Operation;
56     uint8_t T[2];
57     char* Datos;
58     char* BufferAux;
59     char _EOF;
60 }Frame_parameters_t;
61
62 extern volatile Frame_parameters_t Frame_parameters;
```

TP1 – trama usada por ISR

TP1 – trama del frame q llega de ISR

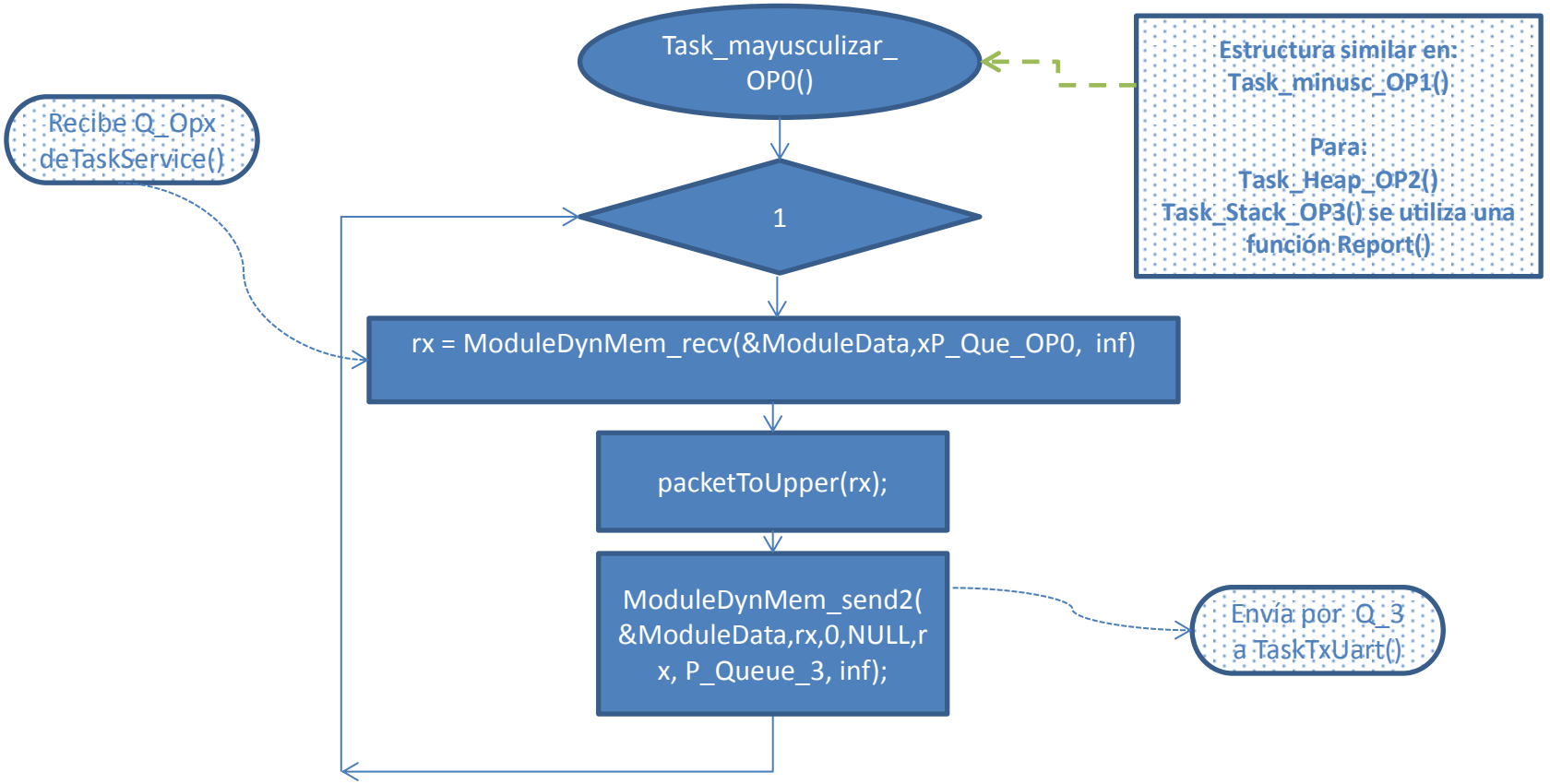
Estructura actual TP1 / Grupo 1 RTOS2

2) Proceso en TaskService v4.6.19



Estructura actual TP1 / Grupo 1 RTOS2

3) Formato de Tasks Mayusculizar, Minusc, Stack y Heap



Estructura actual TP1 / Grupo 1 RTOS2

4) Salida de Tasks Mayusculizar, Minusc, Stack y Heap