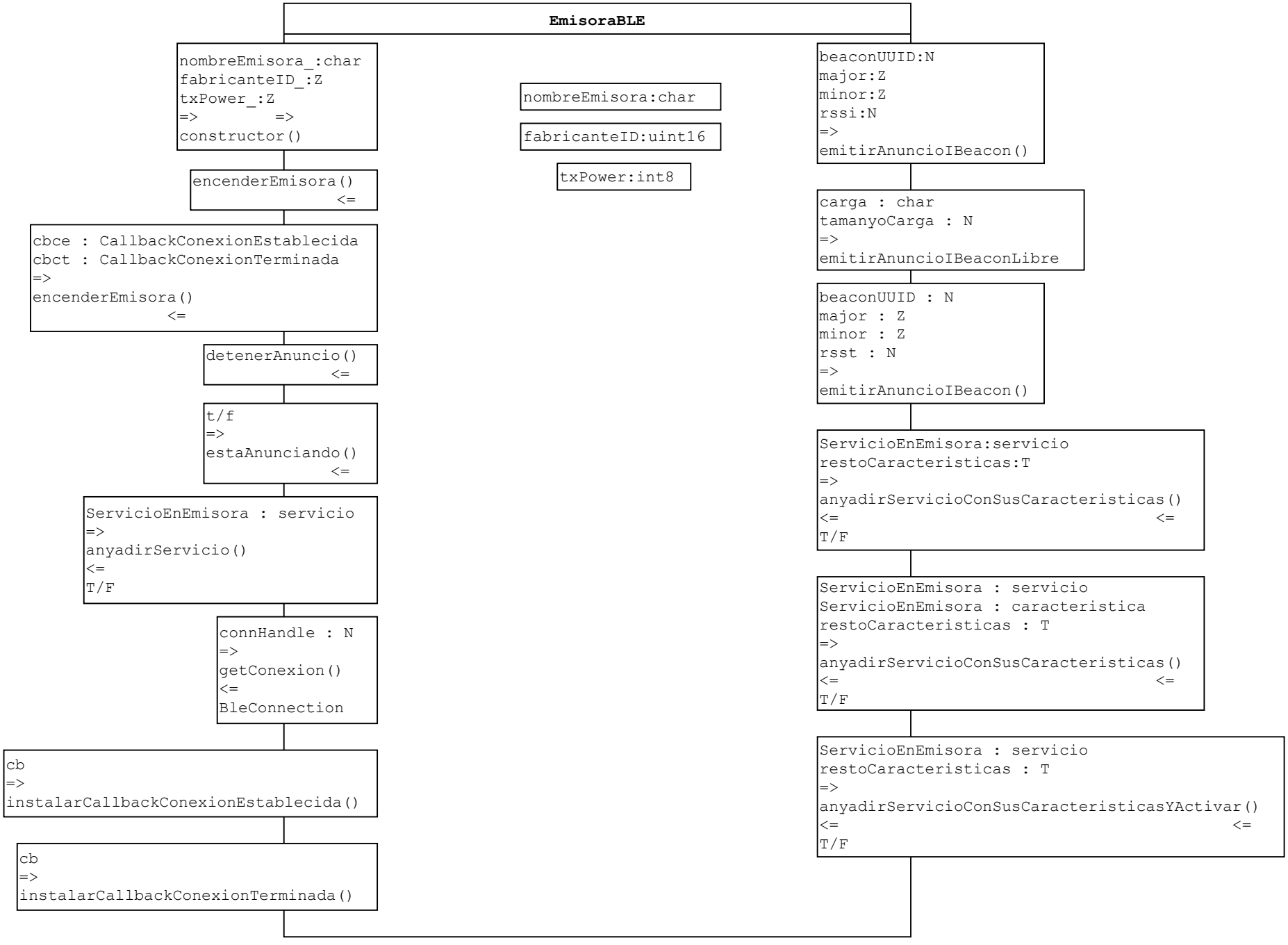
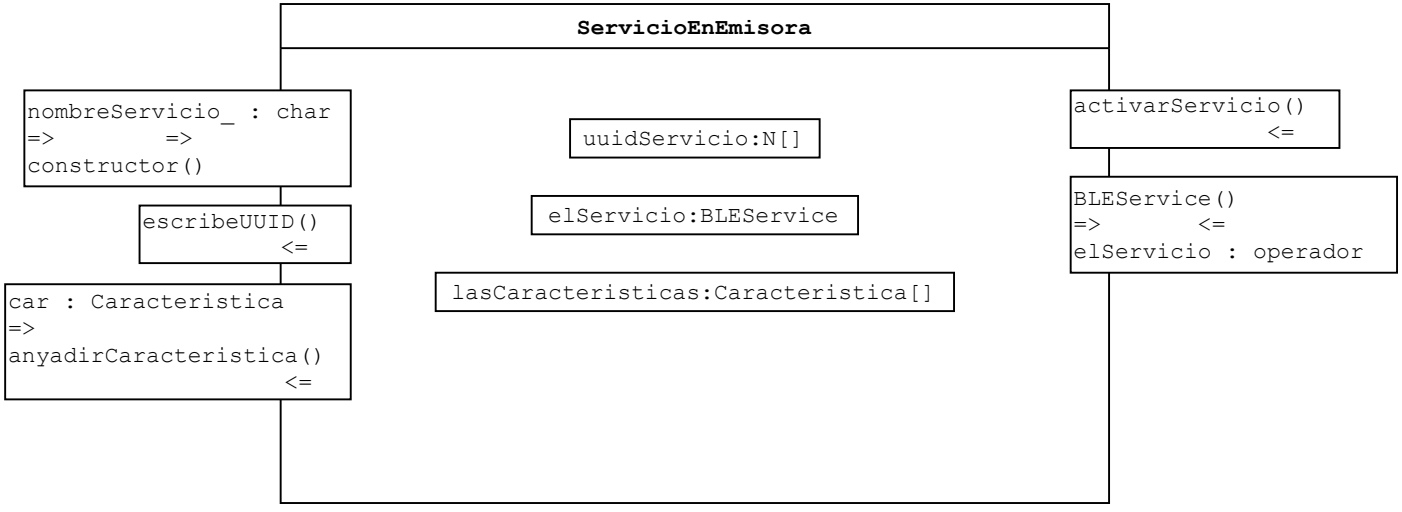
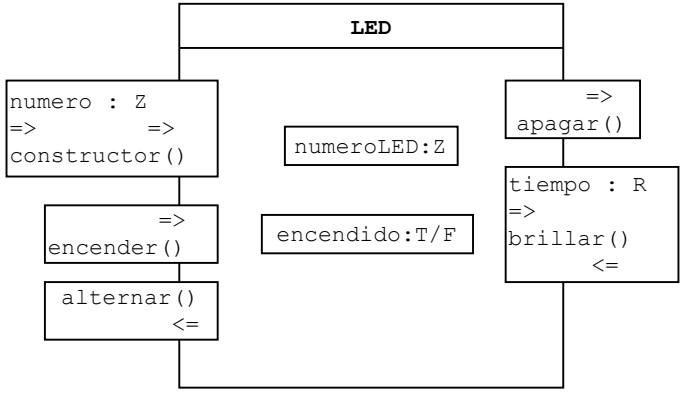
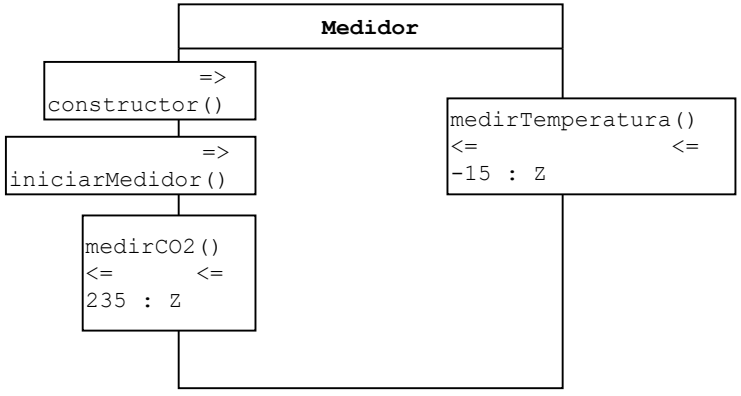
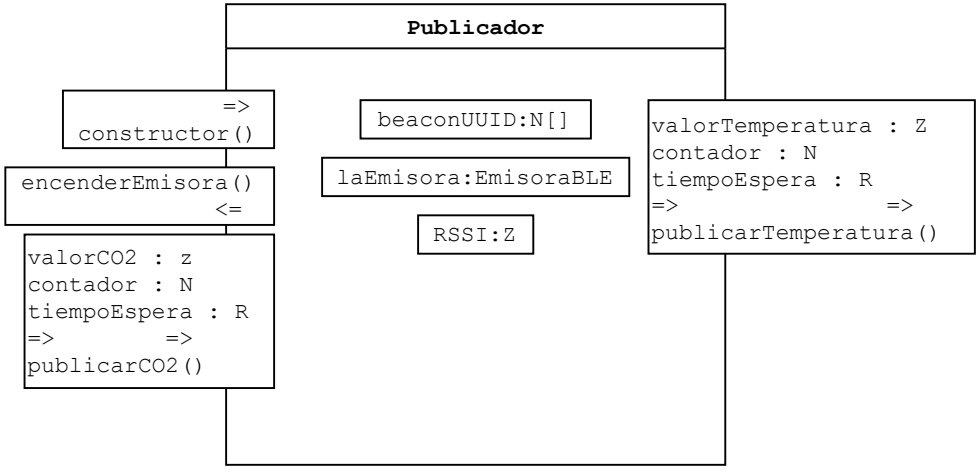
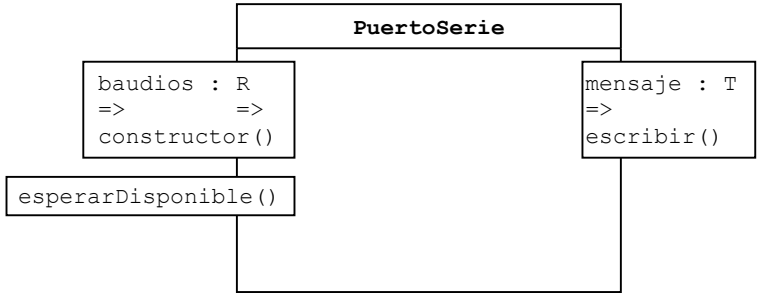

EmisoraBLE
nombreEmisora_:char
fabricanteID_:Z
txPower_:Z
=> =>
constructor()
encenderEmisora()
<=
cbce : CallbackConexionEstablecida
cbct : CallbackConexionTerminada
=>
encenderEmisora()
<=
detenerAnuncio()
<=
t/f
=>
estaAnunciando()
<=
ServicioEnEmisora : servicio
=>
anyadirServicio()
<=
T/F
connHandle : N
=>
getConexion()
<=
BleConnection
cb
=>
instalarCallbackConexionEstablecida()
cb
=>
instalarCallbackConexionTerminada()
nombreEmisora:char
fabricanteID:uint16
txPower:int8
beaconUUID:N
major:Z
minor:Z
rssi:N
=>
emitirAnuncioIBeacon()
carga : char
tamanyoCarga : N
=>
emitirAnuncioIBeaconLibre
beaconUUID : N
major : Z
minor : Z
rsst : N
=>
emitirAnuncioIBeacon()
ServicioEnEmisora:servicio
restoCaracteristicas:T
=>
anyadirServicioConSusCaracteristicas()
<= <=
T/F
ServicioEnEmisora : servicio
ServicioEnEmisora : caracteristica
restoCaracteristicas : T
=>
anyadirServicioConSusCaracteristicas()
<= <=
T/F
ServicioEnEmisora : servicio
restoCaracteristicas : T
=>
anyadirServicioConSusCaracteristicasYActivar()
<= <=
T/F


ServicioEnEmisora
nombreServicio_ : char
=> =>
constructor()
escribeUUID()
<=
car : Caracteristica
=>
anyadirCaracteristica()
<=
uuidServicio:N[]
elServicio:BLEService
lasCaracteristicas:Caracteristica[]
activarServicio()
<=
BLEService()
=> <=
elServicio : operador


LED
numero : Z
=> =>
constructor()
=>
encender()
alternar()
<=
numeroLED:Z
encendido:T/F
=>
apagar()
tiempo : R
=>
brillar()
<=


Medidor
=>
constructor()
=>
iniciarMedidor()
medirCO2()
<=
<=
235 : Z
medirTemperatura()
<=
<=
-15 : Z


Publicador
=>
constructor()
encenderEmisora()
<=
valorCO2 : z
contador : N
tiempoEspera : R
=> =>
publicarCO2()
beaconUUID:N[]
laEmisora:EmisoraBLE
RSSI:Z
valorTemperatura : Z
contador : N
tiempoEspera : R
=> =>
publicarTemperatura()


PuertoSerie
baudios : R
=> =>
constructor()
mensaje : T
=>
escribir()
esperarDisponible()