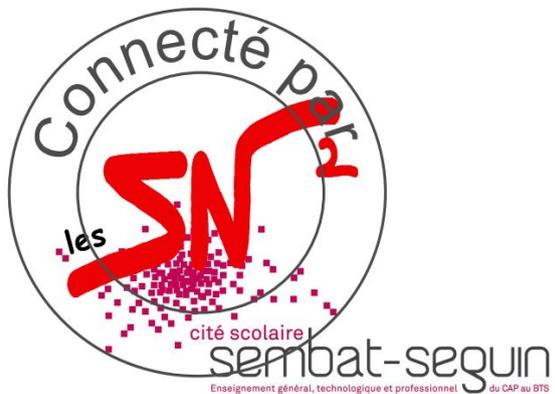


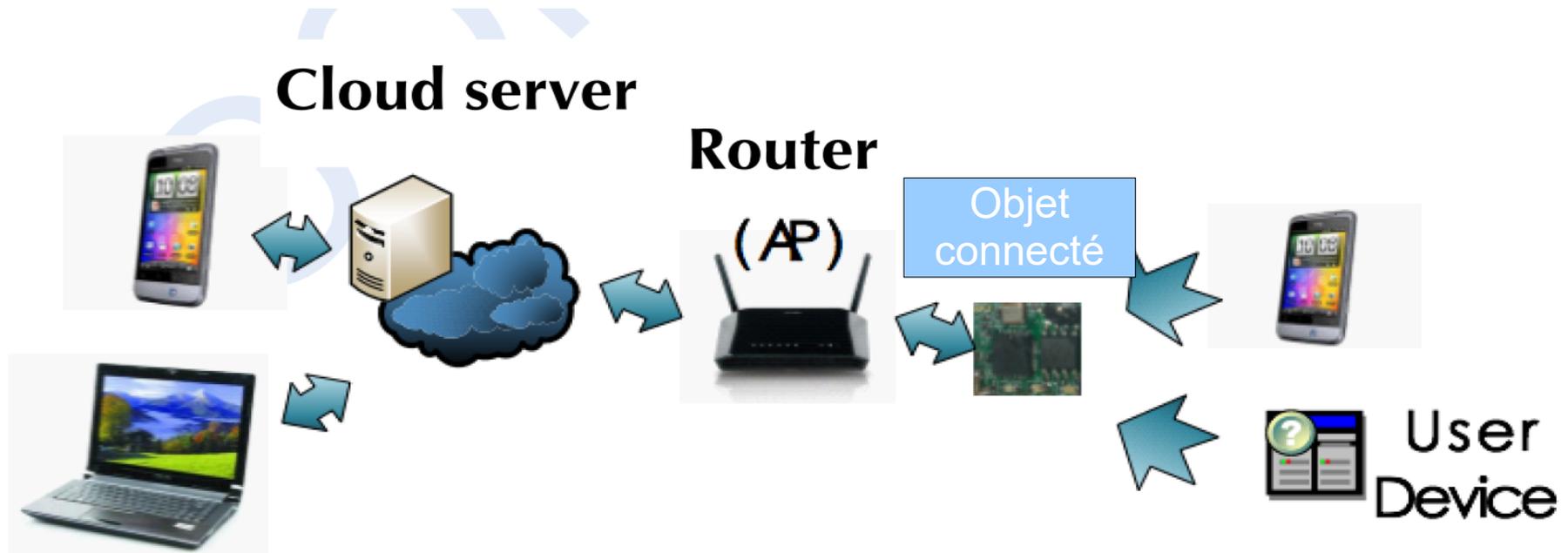
L'IoT

L'internet des objets

Internet of thing



Les objets connectés (IoT) ?



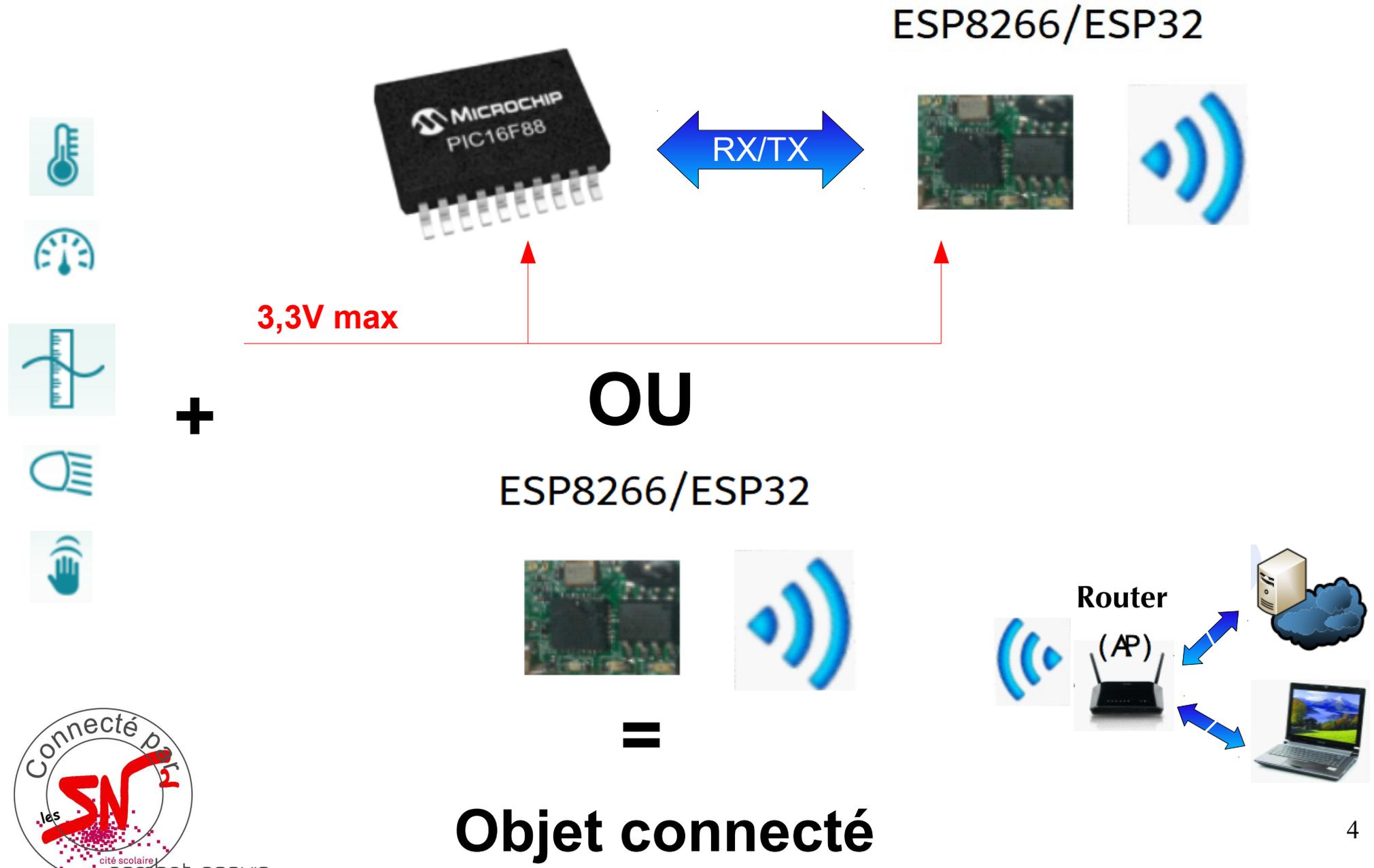
- Objet connecté : mode client et/ou serveur
- Cloud server : récolte, stocke, exploite les données
- Router(AP) : Access point (point d'accès wifi)

Les objets connectés : mode client

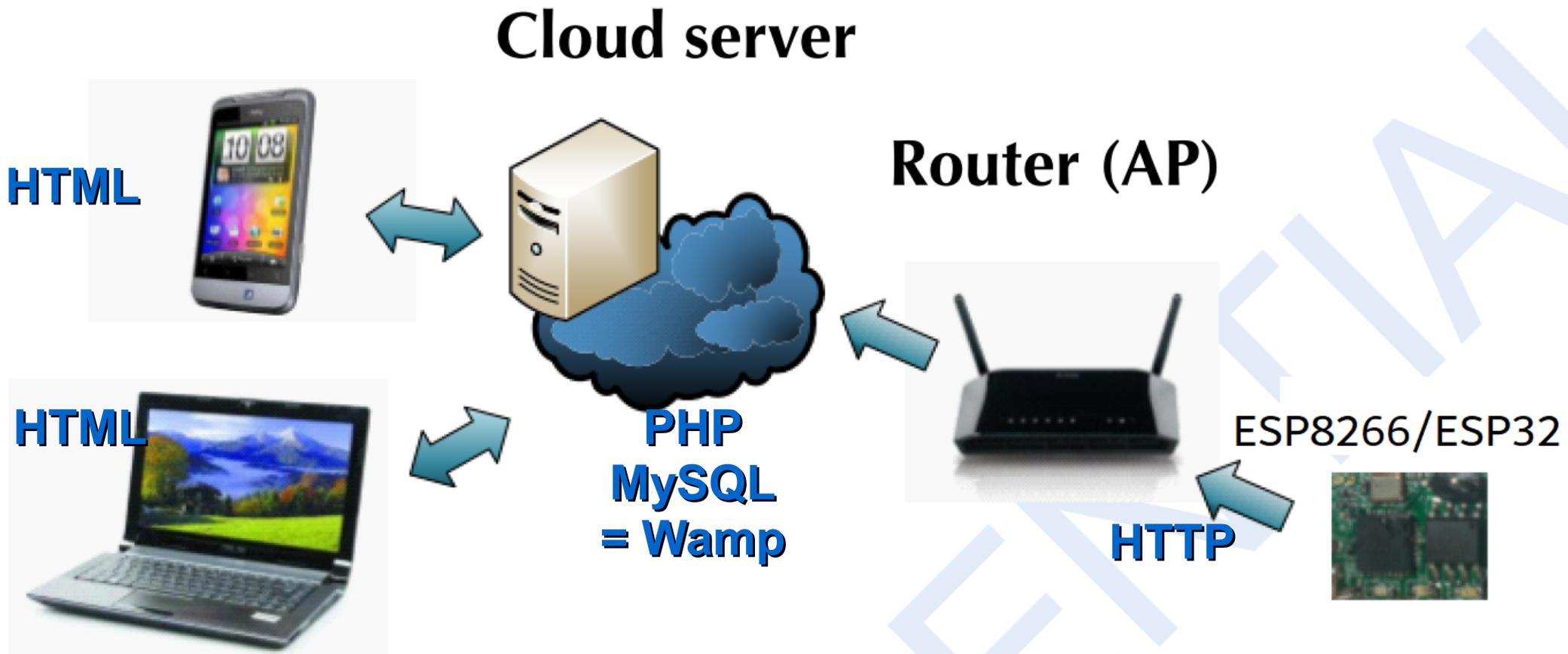
- L'IoT ?
- Les matériels
- Les échanges clients/serveurs
- Les requêtes HTTP
- Coté client :
 - Le module ESP8266/ESP32
 - ESP8266 : configuration
 - ESP8266 : mode AT / mode standalone
 - ESP32 : mode standalone
- Coté serveur :
 - Le serveur WAMP
 - La base de données : MySql
 - La programmation serveur avec Php
- L'IoT complet



Les matériels ...

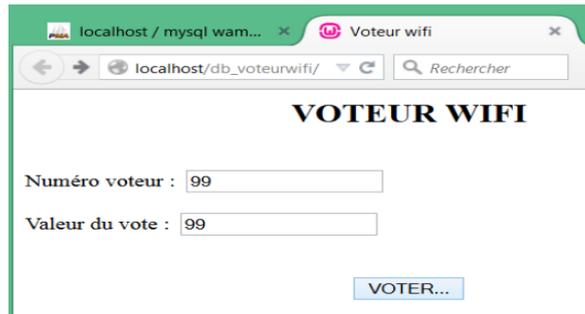


Les échanges client/serveur ...



- Objet vers Serveur : HTML+requêtes HTTP
- Serveur vers PC client : Php
- Serveur base de données : MySQL

Les requêtes HTTP...



Cloud server



ESP8266



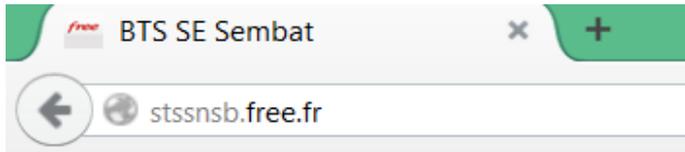
3,3V max

Échange client vers serveur = requête HTTP

- GET : données dans l'URL
- POST : données dans le datagramme (utilisé par php)

IoT : Les requêtes HTTP...

- GET



Appel d'une page HTML depuis un client (navigateur...)

```

HTTP 372 GET / HTTP/1.1
HTTP 683 HTTP/1.1 200 OK (text/html)
HTTP 418 GET /menuhaut.htm HTTP/1.1
HTTP 420 GET /indexmain.htm HTTP/1.1
HTTP 413 HTTP/1.1 200 OK (text/html)
HTTP 900 HTTP/1.1 200 OK (text/html)
HTTP 404 GET /logosembat.gif HTTP/1.1
HTTP 403 GET /logosembat.gif HTTP/1.1

Bytes captured (2976 bits)
c8:d2:61), Dst: f4:ca:e5:56:72:03 (f4:ca:e5:
(192.168.1.39), Dst: 212.27.63.104 (212.27.
(49279), Dst Port: http (80), Seq: 1, Ack:

50 18 ?h...P$Y ..+.a.P.
54 50 ...h..GE T / HTTP
73 73 /1.1..Ho st: stss
73 65 nsb.free .fr..Use
6c 61 r-Agent: Mozilla
4e 54 /5.0 (wi ndows NT
76 3a 6.3; wo w64; rv:
31 30 39.0) Ge cko/2010
39 2e 0101 Fir efox/39.
74 2f 0..Accep t: text/
6f 6e html,app lication
6c 69 /xhtml+x ml,appli
2e 39 cation/x ml;q=0.9
63 65 ,*/*;q=0 .8..Acce
72 2c pt-Langu age: fr,
Fr-FR;q= 0.8,en-U
5;q=0.5, en;q=0.3
..Accept -Encodin
g: gzip, deflate
..DNT: 1 ..Connec
tion: ke ep-alive
.....
    
```

- POST

Réponse d'un formulaire rempli par un client vers un serveur

On retrouve le nom du bouton radio et sa valeur : radioLEDROUGE=1

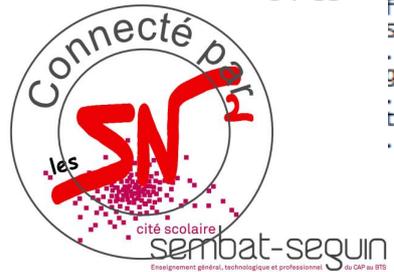
```

8.0.39 TCP 58 http > 53295 [SYN, ACK] Seq=0 Ack=1 win=5840 Len=0 MSS=1460
8.0.13 TCP 54 53295 > http [ACK] Seq=1 Ack=1 win=16616 Len=0
8.0.13 HTTP 477 POST / HTTP/1.1 (application/x-www-form-urlencoded)
8.0.39 TCP 54 http > 53295 [ACK] Seq=1 Ack=424 win=5417 Len=0
c SSDP 208 M-SEARCH * HTTP/1.1
c SSDP 208 M-SEARCH * HTTP/1.1

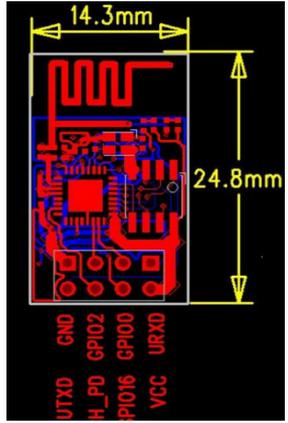
M-SEARCH: * HTTP/1.1
Host: 192.168.0.13
Man: http://schemas.xmlsoap.org/wsman/
Content-Type: application/xml;q=0.9,*/*;q=0.8
Content-Length: 15...

HTTP/1.1 200 OK (text/html)
Date: Wed, 11 Jun 2014 12:51:00 GMT
Server: Apache/2.4.18 (Ubuntu)
Content-Type: text/html; charset=UTF-8
Content-Length: 15...

<input type="radio" value="radioLEDROUGE=1" />
    
```



module ESP8266 : présentation...



Différents modules :

- Antenne intégrée : mod 01, 12, 12E
- Antenne externe : mod 02,05,07...

module ESP8266 : configuration...

The image shows two windows from a Windows environment. The left window is 'AppStack ESP8266 Config V. 1.0.0.0'. It has a top bar with 'ESP8266 Config' and a Facebook icon. Below that, it shows 'Port: COM1' and 'Baudrate: 9600'. The main area is divided into 'Config' and 'About' tabs. Under 'Config', there are sections for 'Mode' (AP, STA, AP+STA), 'Mux' (Single, Multiple), 'AP' (SSID: ESP_9CFADB, Password, Channel: 1, Encryption: Open), 'Server (TCP)', and 'Client (TCP,UDP)'. A table of detected WiFi networks is visible at the bottom right of the config window.

Name	Encryption	Signal
FreeWifi_secure	Open	-48
orange	Open	-92
NUMERICABLE-MAISON	WPA_WPA2_PSK	-57
SFR Wifi FON	Open	-63
SFR Wifi Mobile	Open	-62
NEUF_5D1C	WPA2_PSK	-63
freebox_soukfamille	WPA_PSK	-52
FreeWifi_secure	Open	-54
FreeWifi	Open	-52

The right window is 'Serial Monitor'. It shows a series of AT commands and their responses. The commands include setting the AP mode, setting the SSID to 'BLUSSONWIFI', and connecting to the AP. The responses show the module is busy, then reports the connection status and IP address (192.168.4.1).

```
+CWLAP: (2, "NEUF_5D1C", -63, "e0:a1:d7:71:5d:20", 11)
+CWLAP: (1, "Freebox_soukFamille", -52, "00:24:d4:e7:8e:10", 13)
+CWLAP: (0, "FreeWifi_secure", -54, "00:24:d4:e7:8e:12", 13)
+CWLAP: (0, "FreeWifi", -52, "00:24:d4:e7:8e:11", 13)

OK
busy p...
AT+CWLAP="BLUSSONWIFI", [redacted]

OK
AT+CIFSR
192.168.0.13 IP si connecté en STA sur "BLUSSONWIFI"

OK
AT+CWMODE=3

OK
AT+CWLAP? +CWLAP:"BLUSSONWIFI"

connecté en AP/STA : (access point/station hote)
connexion STA sur "BLUSSONWIFI"
connexion AP (access point) : passerelle 192.168.4.1
192.168.4.1
192.168.0.13

OK
AT+CIPSERVER=1,
ERROR
AT+CWLIF

OK

AT+CWLIF
adresse IP d'un tph connecté
en Wifi sur le AP

OK
AT+CWLIF 192.168.4.100,dc:f1:10:a7:07:c7

OK
```

ESP8266 2 modes Wifi :

- AP : access point : le module fournit un accès Wifi autour de lui (192.168.4.1)
- STA : (Station) le module se connecte à un réseau existant (xxx.xxx.xxx.xxx)

ESP8266 : les fonctionnements...

- Mode AT (module + microcontrôleur) :

ESP8266EX AT Instruction Set

Espressif Systems

3. Instruction Listing

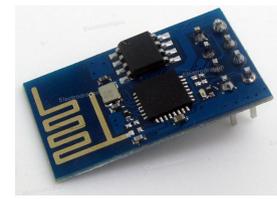
Command	Description
Basic	
AT	Test AT startup
AT+RST	Restart module
AT+GMR	View version info
Wi-Fi	
AT+CWMODE	Select Wi-Fi application mode
AT+CWJAP	Join Ap
AT+CWLAP	Lists available AP
AT+CWQAP	Disconnect from AP
AT+CWSAP	Set parameters under AP mode
AT+ CWLIF	Check IP of connected device
TCP/IP	
AT+CIPSTATUS	Get connection status
AT+CIPSTART	Establish TCP connection or register UDP ports
AT+CIPSEND	Send Data
AT+CIPCLOSE	Close TCP or UDP connection
AT+CIFSR	Get local IP address
AT+CIPMUX	Start multiple connections
AT+CIPSERVER	Configure as server
AT+CIPMODE	Set module transfer mode
AT+CIPSTO	Set server timeout
Data Rx	
+IPD	Receive network data

ESP8266EX AT Instruction Set

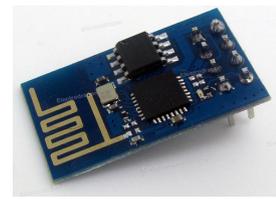
Espressif Systems

4. Basic AT Instruction Set

Command	Description	Response	Reference
AT	Test AT startup	OK	
AT+RST	Restart module	OK	



- Mode Standalone (module sans microcontrôleur) :



ESP32 : le fonctionnement...

- Mode Standalone (module sans microcontrôleur) :
Evolution de l'ESP8266 avec :
- Plus de GPIO
- Bluetooth BLE et standard
- Wifi

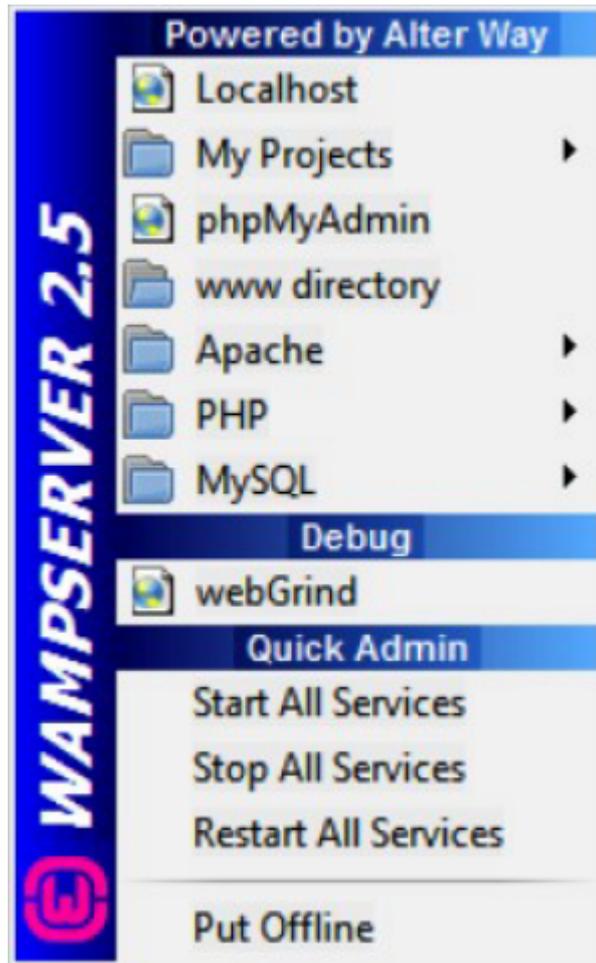


OU

+



Coté serveur : Wamp ...



Rôles :

- Créer un serveur Web : Apache
- Gérer des bases de données : MySQL
- Programmer des interactions Web2.0 avec PHP (langage de programmation)
- Travailler en local avant publication



Coté serveur : MySQL...



Rôles :

- Créer
- Gérer
- Organiser

The screenshot shows the phpMyAdmin interface for a MySQL server. The table 'vote' is selected, and its structure is displayed. The interface includes a sidebar with a database tree (annotated with '1'), a top navigation bar with various actions (2), and a main table structure view with columns: 'id' (3), 'heure' (5), 'num_voteur' (7), 'val_vote' (8), and 'commentaire' (9). The 'id' column is highlighted in blue. The 'heure' column has a default value of 'CURRENT_TIME' (6). The storage engine is set to 'InnoDB' (7). The 'commentaire' column is of type 'TEXT' (9). A 'Sauvegarder' button is visible at the bottom right (10).

des bases de données facilement



Coté serveur : programmation



Rôles :

- Créer une interactivité entre client/serveur (web2.0)
- Recevoir et organiser les données de l'objet connecté
- Réaliser des calculs,
- Générer des pages HTML

```
<?php

//script affichant les données de la base db_mesures en mode texte et en mode graphique en utilisant canvas de html5

$db = mysqli_connect("localhost","root","","db_voteurwifi") or die("Erreur : ouverture de la base " . mysqli_error($link));

//écriture des données dans la base Attention rajouter une verification du format des données...
//recuperation et mise en variable des data provenant de POST
$num_voteur = strip_tags(trim($_POST['num_voteur'])); //recupere la mesure1 du formulaire
$val_vote = strip_tags(trim($_POST['val_vote'])); //recupere la mesure2 du formulaire
// $heure = strip_tags(trim(date("Y-m-d H:i:s",strtotime("now")))); //recupere date et heure du systeme voir si necessaire du fait de at

//insertion des variables ds la table de la db
$sql = "INSERT INTO table_vote (num_voteur, val_vote) VALUES ('".$_POST['num_voteur']."','".$val_vote."')";
$result = mysqli_query($db,$sql);

//consultation:
$query = "SELECT * FROM table_vote" or die("Erreur : consultation.." . mysqli_error($db));

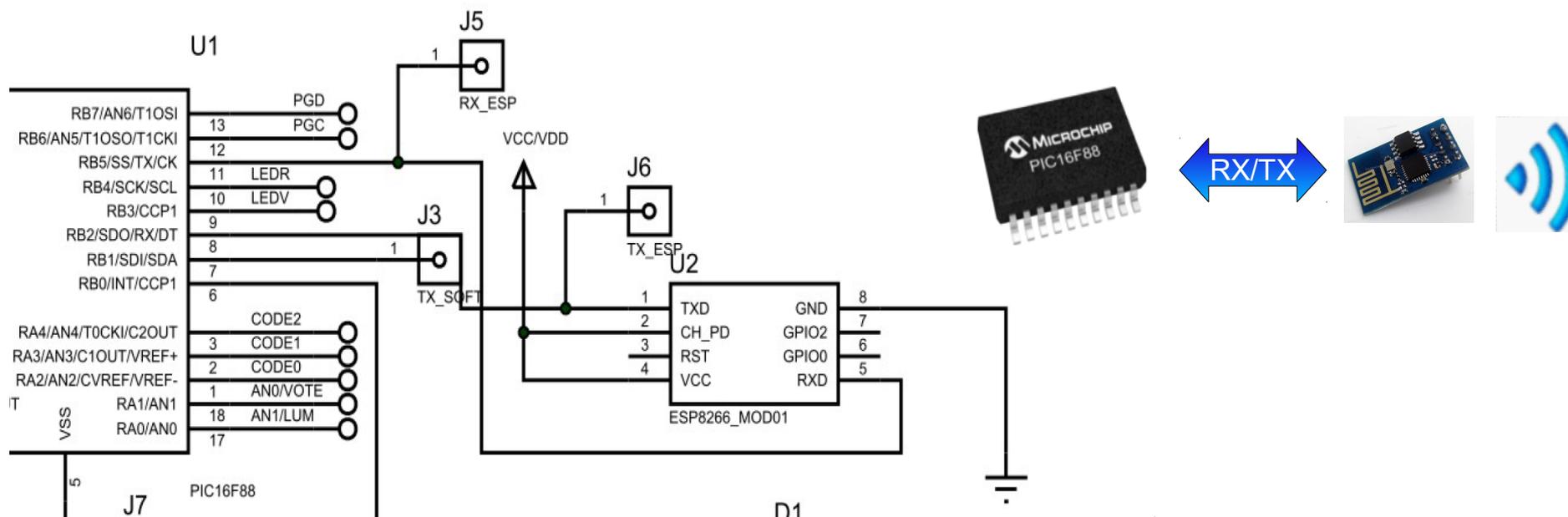
//execute the query.
$result = $db->query($query);

//display information:
echo '<body style="color: rgb(0, 0, 0); background-color: rgb(255, 204, 153);" alink="#000099" link="#000099" vlink="#990099">';
echo '<style>H1 {color:red;text-align:center} H3 {color : green;text-align:center;font-size: xx-large;} H4{font-size:xx-small}</style>';

"<H1>VOTEUR WIFI : Visualisation des votes </H1>";

ce=0;//indice de tableau
($row = mysqli_fetch_array($result)) {
```

IoT : le voteur wifi ...



```

D1
#include <rs232.h> (baud=9600, parity=N, xmit=PIN_B0, rcv=PIN_B1, bits=8, errors, STREAM=DEBUG)

#define CODE0 PIN_A2
#define CODE1 PIN_A3
#define CODE2 PIN_A4
#define CODE3 PIN_A6
#define CODE4 PIN_A7

#define LEDVERTE PIN_B3
#define LEDROUGE PIN_B4

#define VOTE_PIN PIN_A1
#define LUM_PIN PIN_A0

// #define ESP8266_RST PIN_B1 // reset du ESP !! attention ne pas utiliser B1 pour rs232 DEBUG

#define ADRESSEWEB "192.168.73.1" // adresse du serveur web (PC sur lequel WAMP est en marche)

// variables globale
int8 NUM_VOTEUR = 12;
int8 VAL_VOTE = 4; // 0 < VAL_VOTE < 10
boolean ReponseRequeteOK = false;
    
```

IoT : le voteur wifi – code .C ...

```
void EnvoiRequetePOST(int8 mesure1,mesure2){
//Envoi la requete POST au module par commande AT : la requete contient les données mesure1 et mesure2 (ici mesure1 = NUM_VOTEUR mesure2 = VAL_VOTE
//ok testé

    printf("AT+CIPSTART=\"TCP\", \"192.168.73.1\", 80\r\n");
    testOK();
    //test LINKED
    while(getc() != 'L' ){};
    while(getc() != 'i' ){};
    while(getc() != 'n' ){};
    while(getc() != 'k' ){};
    while(getc() != 'e' ){};
    while(getc() != 'd' ){};

    delay_ms(1000);

    printf("AT+CIPSEND=186\r\n");//=xxx avec xxx le nombre d'octet à envoyer
    printf("POST /db_voteurwifi/db_voteurwifi.php HTTP/1.1\r\n");//50octets-2\=48
//    printf("Host: 192.168.0.10\r\n");//22-2\=20
    printf("Host: 192.168.73.1\r\n");//22-2\=20
//    printf("Connection: keep-alive\r\n");//26-2\=24 (210 si mis)
    printf("Content-Type: application/x-www-form-urlencoded\r\n");//51-2\ = 49
    printf("Content-Length: 27\r\n\r\n");//26-4\ = 22 //27=taille de la ligne suivante
    printf("num_voteur=%3u&val_vote=%3u",mesure1,mesure2);//27 donc total = 159

    printf("AT+CIPCLOSE\r\n");//fermeture connection pas de test car retour varié !
    delay_ms(1000);//
    printf("AT+CWQAP\r\n");testOK();//désactive la connexion

    ReponseRequeteOK=true;
} //fin EnvoiRequetePOST()
```

IoT : le voteur wifi – code .php ...

```
<?php
//script affichant les données de la base db_mesures en mode texte et en mode graphique en utilisant canvas de html5
$db = mysqli_connect("localhost","root","","db_voteurwifi") or die("Erreur : ouverture de la base ")
//recuperation et mise en variable des data provenant de POST
$num_voteur = strip_tags(trim($_POST['num_voteur'])); //recupere la mesure1 du formulaire
$val_vote = strip_tags(trim($_POST['val_vote'])); //recupere la mesure2 du formulaire
//insertion des variables ds la table de la db
$sql = "INSERT INTO table_vote (num_voteur,val_vote) VALUES ('$num_voteur','$val_vote')";
$result = mysqli_query($db,$sql);
//consultation:
$query = "SELECT * FROM table_vote" or die("Erreur : consultation.." . mysqli_error($db));
//execute the query.
$result = $db->query($query);
//display information:
echo "<body style='color: rgb(0, 0, 0); background-color: rgb(255, 204, 153);' aLink='#000099' link=";
echo "<style>H1 {color:red;text-align:center} H3 {color : green;text-align:center;font-size: xx-large";
echo "<H1>VOTEUR WIFI : Visualisation des votes </H1>";
$indice=0; //indice de tableau
while($row = mysqli_fetch_array($result)) {
    echo "<h4>N= ".$row["id"]. " Date = ".$row["heure"]." Numero du voteur = ".$row["num_voteur"] . "
    $tab_num_voteur[$indice] = $row["num_voteur"]; //sauve les mesures dans un tableau tab_mesure1
    $tab_val_vote[$indice] = $row["val_vote"]; //sauve les mesures dans un tableau tab_mesure2
    $indice++; //
    $nbMesure=$indice; //sauve le nombre de mesure dans la variable nbMesure
}
//calcul de la moyenne à partir des val_vote et du nombre de votant obtenu à partir du nombre de cha
$moyennevote = 0;
for($li=0;$li<$nbMesure;$li++){
    $moyennevote = $moyennevote+$tab_val_vote[$li];
} //fin for
$moyennevote = $moyennevote/$nbMesure;
echo "<br><h2>La note moyenne est de </h2><h3>'. $moyennevote.' </h3> avec '$nbMesure.' votants<br>";
echo "<form method='post' action='db_voteurwifi_vider.php' name='form1'><br>";
```

**VOTEUR WIFI :
Visualisation des votes**

N= 82 Date = 2015-09-22 18:55:30 Numero du voteur = 33 Valeur du vote = 0
N= 83 Date = 2015-09-22 18:57:35 Numero du voteur = 12 Valeur du vote = 13
N= 84 Date = 2015-09-22 18:57:49 Numero du voteur = 33 Valeur du vote = 0
N= 85 Date = 2015-09-22 18:58:17 Numero du voteur = 12 Valeur du vote = 13
N= 86 Date = 2015-09-22 18:58:40 Numero du voteur = 33 Valeur du vote = 0

La note moyenne est de

5.2

avec 5 votants

Vider la base...



IoT : le voteur wifi – la base ...

localhost/phpmyadmin/#PMAURL-3:sql.php?db=db_voteurwifi&table=table_vote&server=1&target=&token=38299: Rechercher

Serveur: mysql wampserver » Base de données: db_voteurwifi » Table: table_vote

Afficher Structure SQL Rechercher Insérer Exporter Importer Privilèges

Trier sur l'index: Aucune

+ Options

				id	heure	num_voteur	val_vote	commentaire
<input type="checkbox"/>				108	2015-09-24 09:56:07	33	0	
<input type="checkbox"/>				109	2015-09-24 09:56:29	6	6	
<input type="checkbox"/>				110	2015-09-24 09:56:34	33	0	
<input type="checkbox"/>				111	2015-09-24 09:56:54	6	1	
<input type="checkbox"/>				112	2015-09-24 09:56:57	33	0	
<input type="checkbox"/>				113	2015-09-24 09:57:15	6	15	
<input type="checkbox"/>				114	2015-09-24 09:57:18	33	0	
<input type="checkbox"/>				115	2015-09-24 09:57:54	6	12	
<input type="checkbox"/>				116	2015-09-24 09:57:57	33	0	

Tout cocher Pour la sélection : Modifier Effacer Exporter

Nombre de lignes : 25

Opérations sur les résultats de la requête

Version imprimable Version imprimable (avec textes complets) Exporter Afficher le graphique Créer une vue

Conclusion

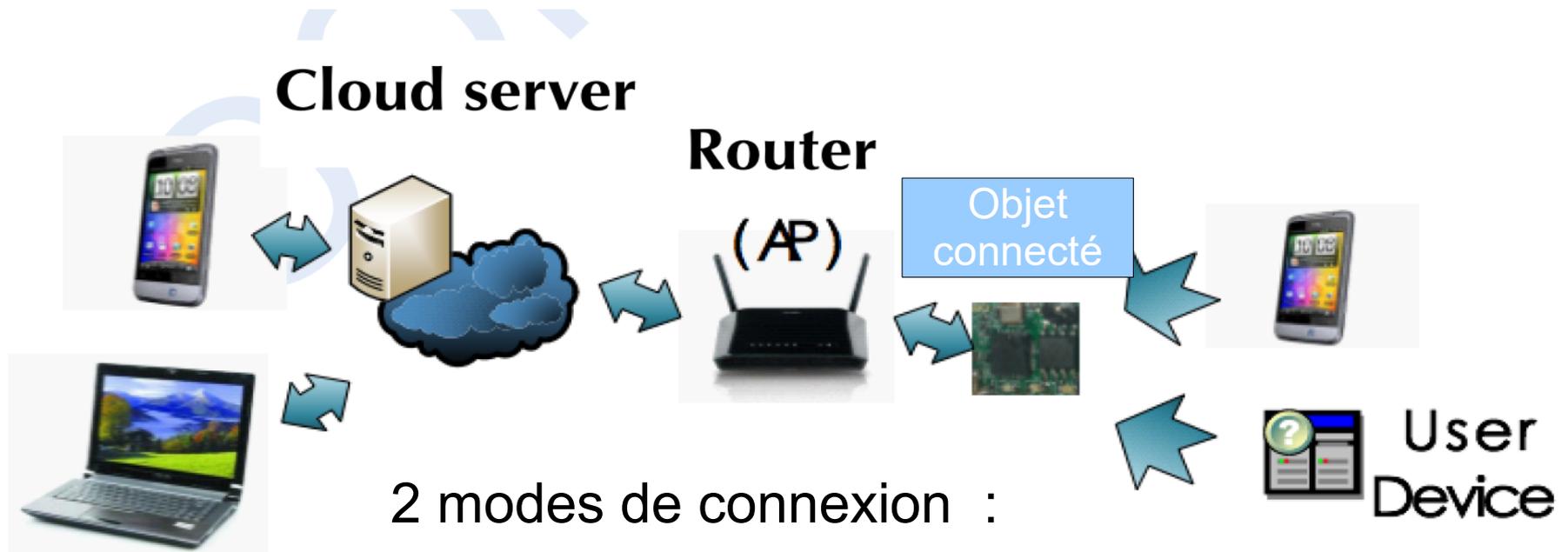
IoT : en mode client

Les outils nécessaires au mode client :

- Wamp : serveur web et base de donnée
- Connectify : création d'un hotspot sur pc serveur Wamp
- MySQL, PHP, HTML
- Wireshark : sniffer



IoT : client ou serveur ?...



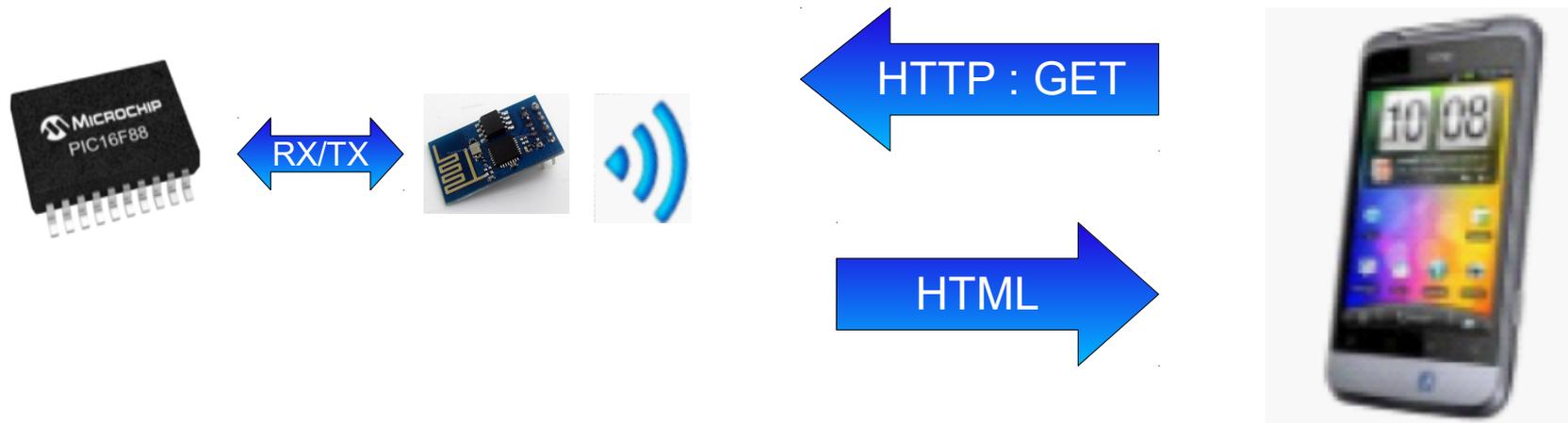
Client :

- Requête (HTTP : GET,POST) du client VERS le serveur

Serveur :

- Réponse du serveur par envoi d'un page HTML, d'une image, d'un fichier...

IoT : serveur ...



2 modes de connexion :

Client :

- Requête (HTTP : GET,POST) du client VERS le serveur

Serveur :

- Réponse du serveur par envoi d'une page HTML, d'une image, d'un fichier...

IoT : serveur (requête GET)...

```
do{
// fputs("ATE0");//annule l'echo des commandes AT
// fprintf(DEBUG,"ATE0 \r\n");
// delay_ms(1000);
fputs("AT");//test la communication avec le module ESP8266
testOK();//attend la confirmation OK du module

delay_ms(1000);
fputs("AT+CIPMUX=1"); testOK(); delay_ms(1000); //mode multiplexage m
Efface_Ecran();printf(Affiche_caractere,"CIPMUX=OK"); delay_ms(1000);

valCAN0 = acquerirCAN(0);

fputs("AT+CIPSERVER=1,80"); testOK(); delay_ms(1000);//mode SERVER sur port 80 (HTTP)
Efface_Ecran();printf(Affiche_caractere,"Attente reqHTTP");
Lcd_Place_Curseur(2, 1); printf(Affiche_caractere,"CAN0 = %u",valCAN0);
fprintf(DEBUG,"REQ HTTP? \r\n"); delay_ms(1000);

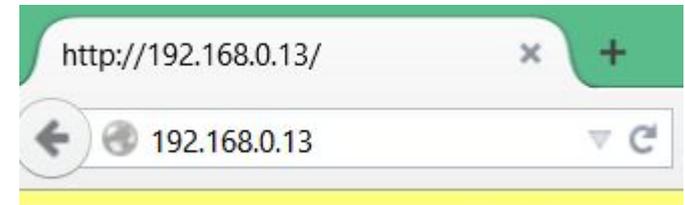
while(getc()!='G') {};//attente du G de GET provenant d'une requete

Efface_Ecran();printf(Affiche_caractere,"Cfç canaldata"); delay_ms(1000);
channel = 0;
printf("AT+CIPSEND=%u,%lu\r\n",channel,nbocetet);//indique le canal et la taille des données envoyées

// Efface_Ecran();printf(Affiche_caractere,"Attente >");//risque de gener la synchro
while (getc()!='>'){};//attente >
CreerPageHtml();//envoi page web
Efface_Ecran();printf(Affiche_caractere,"Envoi PageWeb"); delay_ms(1000);
puts("AT+CIPCLOSE=0"); delay_ms(1000); //fermeture de la connection

//printf(Affiche_caractere,"%s",buffer);
printf(Affiche_caractere,"Fin");
}while(1);
```

ESP8266



Requête GET :

Un client (PC, Tph) veut accéder à la page Web de l'objet connecté

(Serveur en STA : 192.168.0.13).

IoT : serveur (requête GET réponse)...

```

void CreerStyleHtml () {
  /*creation du style de la page
  printf("<STYLE>");//7char
  printf("H1 {color:#804040; font-weight: bold; font-size: 24px; font-family: serif; font-style: normal; text-align: center; }");//19char
  printf("H2 {color:#000000;background:#FFFFFF; font-family: Comic Sans MS; font-size: 16px; font-style: normal; text-align: left; }");//35char
  printf("</STYLE>");//8char
  //256 char
} //fin CreerStyleHtml

void CreerPageHtml () {
  printf("<HTML><HEAD></HEAD>");//19 char
  CreerStyleHtml (); //256char
  printf("<BODY bgcolor = '#FFFFFF7F'>");//26char
  printf("<H1>SERVEUR WEB</H1>");//20 char
  printf("<H2>CAN1 = %03u </H2>", valCAN0); //20char
  printf("</BODY></HTML>");//14char
  printf("\r\n\r\n");
} //fin CreerPageHtml
  
```

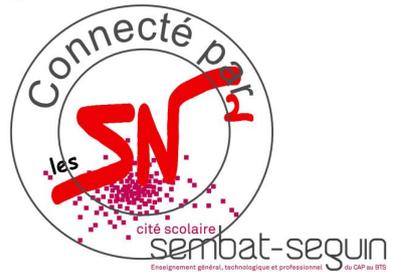


Requête GET réponse :

L'objet connecté (serveur) renvoie la page HTML

```

while (getc() != '>') {} //attente >
CreerPageHtml (); //envoi page web
Efface_Ecran (); printf(Affiche_caractere, "Envoi PageWeb"); delay_ms(1000);
puts("AT+CIPCLOSE=0"); delay_ms(1000); //fermeture de la connection
  
```

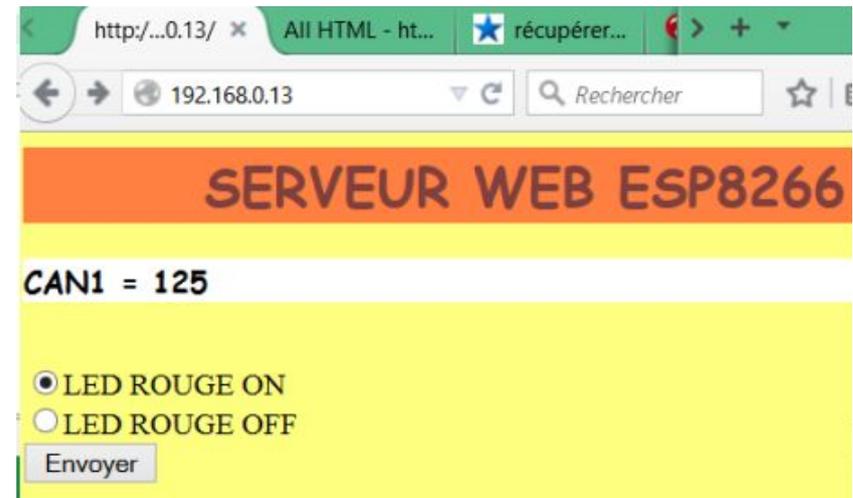


IoT : serveur (requête POST)...

```
void CreerFormHtml(){
//creation du formulaire HTML
//si LEDON alors
//alors cocher la case de la LEDON et mettre la valeur à 1
//sinon ne pas cocher la case de la LEDOFF et mettre la valeur à 0
printf("<form method=\"post\" name=\"form1\"><br>");//37char le \" permet d'écrire \" en html sans fermer le printf !

if (LEDROUGEON)
{
printf("<input name=\"RDIOLR\" value=\"1\" \");//31char
printf("type=\"radio\" checked>LED ROUGE ON<br>");//37char
printf("<input name=\"RDIOLR\" value=\"0\" \");//31char
printf("type=\"radio\">LED ROUGE OFF<br>");//30char = 129total
}
//fin if LEDROUGEON
else
{
printf("<input name=\"RDIOLR\" value=\"1\" \");//31char
printf("type=\"radio\">LED ROUGE ON<br>");//29char
printf("<input name=\"RDIOLR\" value=\"0\" \");//31char
printf("type=\"radio\" checked>LED ROUGE OFF<br>");//38char = 129total
}
//fin else
//ajout bouton submit
printf("<input type=\"submit\">");//21char
printf("</form>");//7char

}
//fin CreerFormHtml //37 + 129+21+7 = 194
```



Requête POST :

L'objet connecté (serveur) reçoit une requête POST lors de l'interaction sur un formulaire (bouton radio des LEDs)

IoT : serveur (requête POST réponse)

```
//attente d'une réponse de la page web par un POST
//detecter la variable retour attendu "RDIOLR=1" (detecter RDI puis enregistrer dans un tableau afin de récupérer le =0 ou =1
Efface_Ecran();printf(Affiche_caractere,"Attente POST..");
while(getc()!='R'){};
while(getc()!='D'){};
while(getc()!='I'){};
while(getc()!='O'){};
while(getc()!='L'){};
while(getc()!='R'){};
while(getc()!='=' ){};
if(getc()=='1') //ok testé
{
    output_high(LEDROUGE);
    LEDROUGEON=true;
}//if
else //sinon RDIOLR=0 alors eteindra ledrouge et memoriser l'info
{
    output_low(LEDROUGE);
    LEDROUGEON=false;
}
}//else
```



Requête POST réponse :

- L'objet connecté (serveur) attend la requête POST du client ("Envoyer") et décode la valeur des variables (ici RDIOLR)

Merci...