



Construction d'une WEBCAM

C-HTML-CSS-XML-JAVASCRIPT
Serveur WEB embarqué, caméra
Série

Contenu



1. Objectif, Communication
2. Choix les composants
3. Serveur WEB embarqué
4. Interface Camera série
5. Code de capture camera
6. Affichage, Page web, HTML, CSS, XML
7. Essais, première solution
8. Amélioration, utilisation du « cloud »



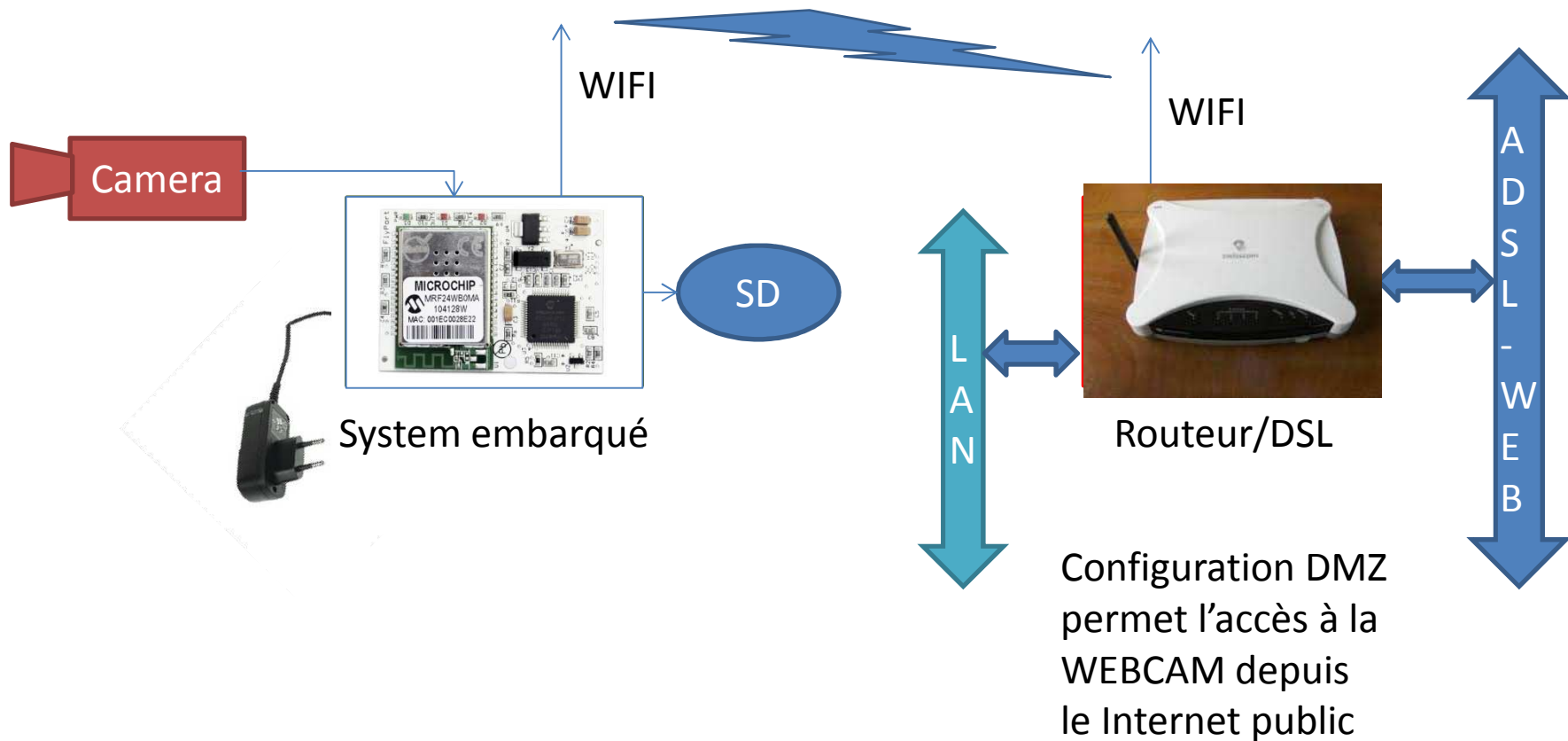
Objectif

- Camera de surveillance
- WEBCAM
- Connectée par wifi
- par routeur wifi local
- Si possible accessible depuis le web public
- Sécurité d'accès

Construction d'une WEBCAM Communication

Point d'observation

- Appartement/bureau



Choix des composants

Flyport Openpicus wifi



- Processeur Microchip 16 bits PIC24f256
- module WIFI Microchip
- Permet de charger plusieurs pages web simples sans trop de photos.
- Environnement de développement Openpicus
- Multiples E/S, UART , SPI, I2C, PWM à disposition

9-Feb-13

Rolf Ziegler

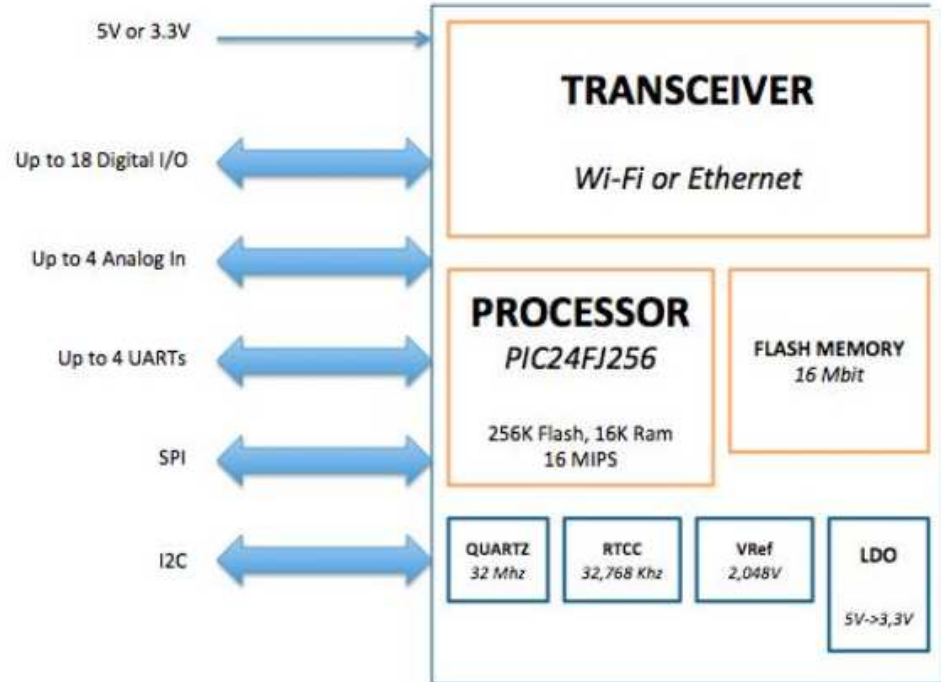
Camera 640x480 pixels
Interface UART (115200baud)
Paramètres programmable
Capteur de mouvements.
Compression JPEG intégrée



Origine Linksprite
Source Sparkfun/Adafruit

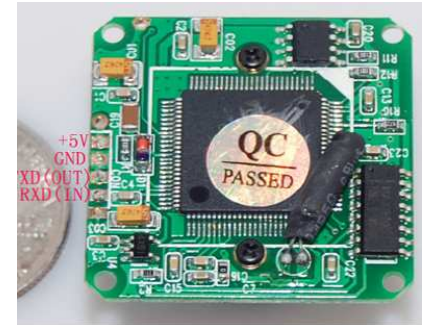
Serveur WEB embarqué

Flyport Openpicus

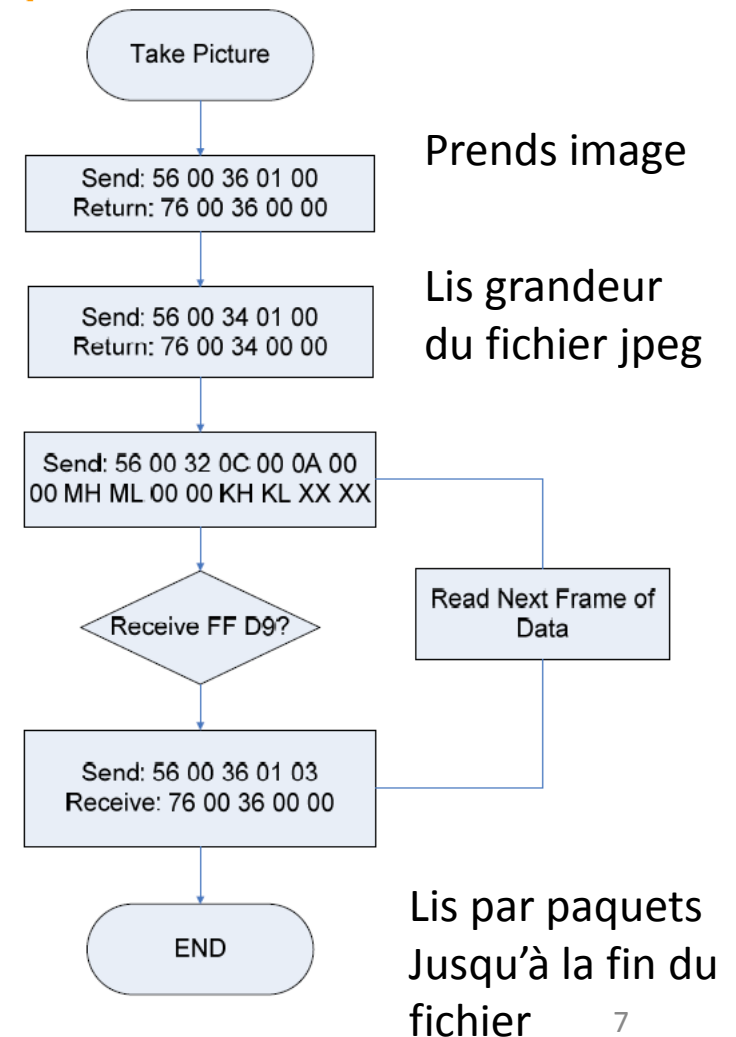


Camera série

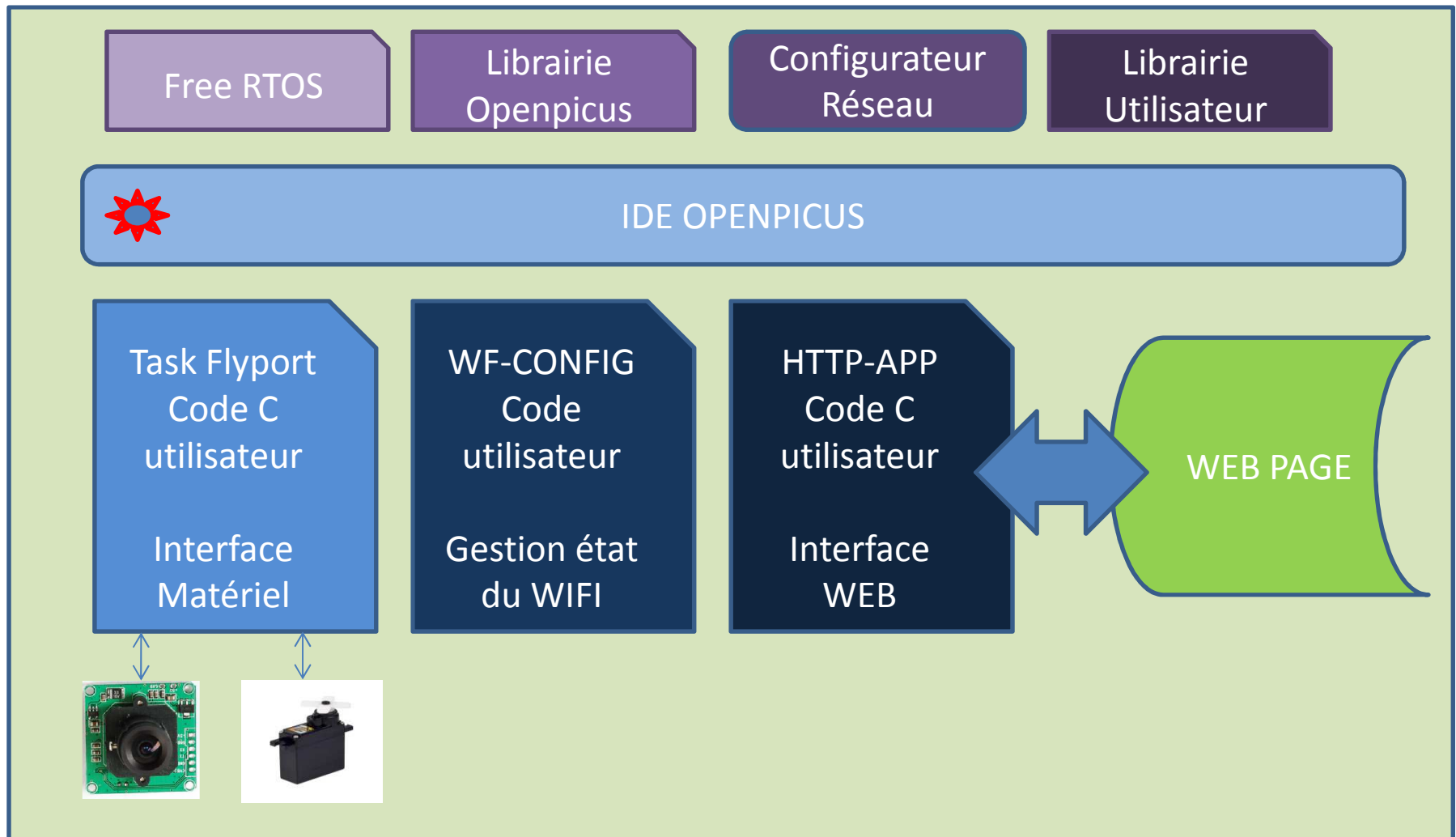
www.linksprite.com

- Commandée par ligne série
- Répond à des commandes
 - Vidéo on -> sortie comp.
 - Stop Vidéo
 - Copy image-> série
 - Copie par blocs de grandeur programmable
 - Information mouvements



WEB Embarqué, logiciel



🌟 Environnement de développement WEB Embarqué logiciel

Projet

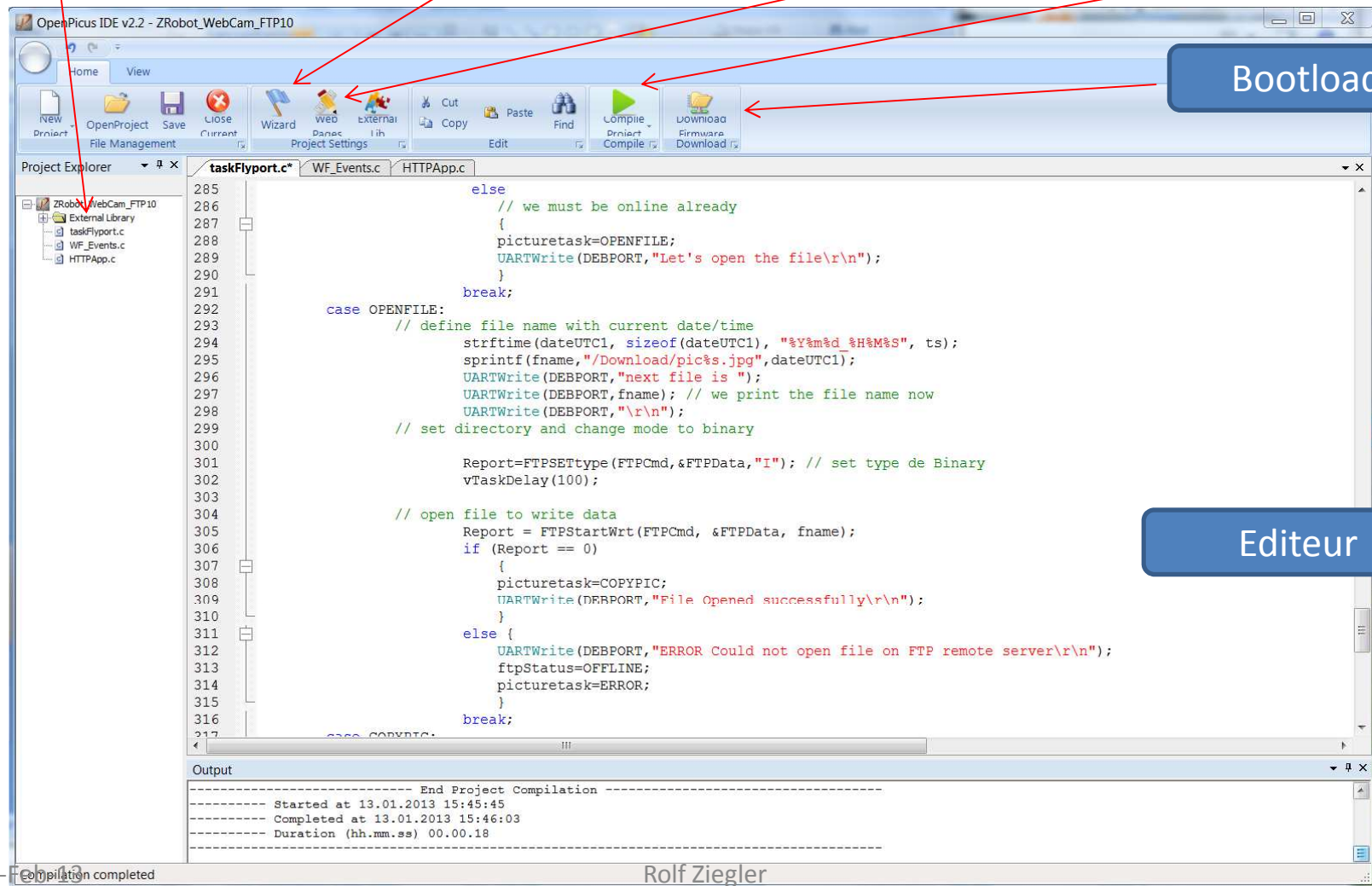
Configuration TCP

Charger HTML

Compilateur C

Bootloader

Editeur



CONFIGURATION

TCP-IP /WEB SERVEUR

TCP/IP Configuration

Services selection
"My application will need a..."

openPicus

Select the basic components you need for your application. You will be able to select more specific options on the following pages.

☒ Web Server	☐ Post Method	☐ ZeroConfig (BETA)
☒ DHCP Client	☐ DNS Server	☐ Dynamic DNS
☐ DHCP Server	☒ DNS Client	☒ TCP debug on UART1
☒ ICMP Client	☐ Announce service	☒ SNTP Client
☐ ICMP Server	☐ NetBIOS name service	☐ Remote reboot service

ZCam_SD22_v1108

< Back Next > Cancel

Wireless Configuration
"Let's fly!"

openPicus

Default SSID Name
MICROCLUB

Default Network Type
AdHoc

Default PS Poll
Disabled

Scan settings

Default Scan Type
Active Scan

Scan Channels

☒ ch.1	☒ ch.4	☒ ch.7	☒ ch.10
☒ ch.2	☒ ch.5	☒ ch.8	☒ ch.11
☒ ch.3	☒ ch.6	☒ ch.9	☐ ch.12

ZCam_SD22_v1108

< Back Next > Cancel

TCP Socket Configuration
"How many sockets do i need?"

openPicus

Currently Defined Sockets

- GENERIC_TCP_CLIENT
- GENERIC_TCP_SERVER
- HTTP_SERVER
- DEFAULT
- FTP_COMMAND
- FTP_DATA

Socket Type
HTTP_SERVER

TX Buffer
1536

Count
2

RX Buffer
512

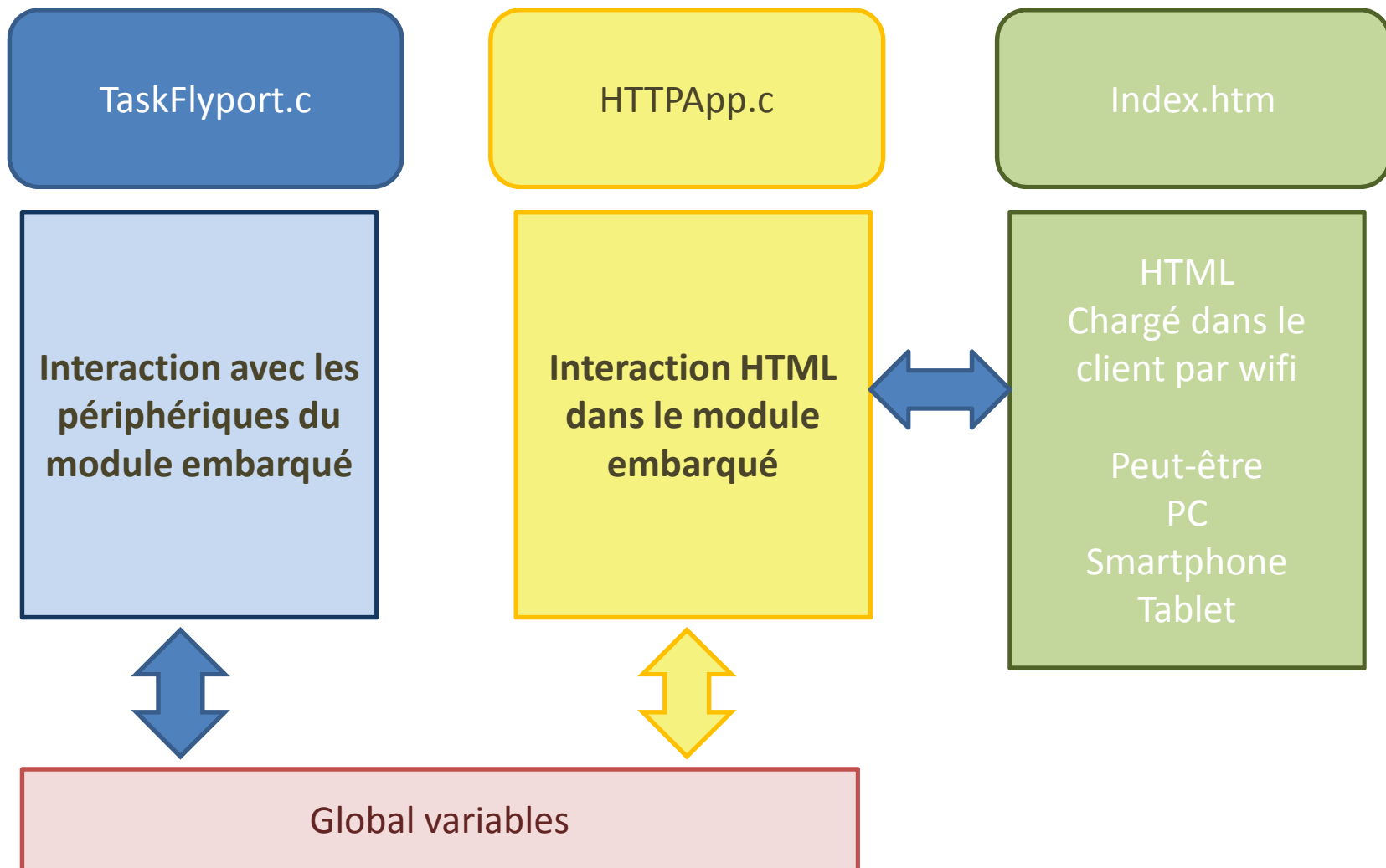
ADVANCED SETTING: if you don't know what you're doing... don't do it! Everything it's ok for a standard application.

Total TCP socket memory (max. 8100)
7792

ZCam_SD22_v1108

< Back Next > Cancel

Multi-tâches, interaction



Code, fonctions

TaskFlyport.c

1.Initiation
périphériques
-camera
-pwm servo
-carte SD

2.Activer web
3.Boucle attente
- Transfer données
camera

HTTPApp.c

**-Code interaction
Boutons (get)**

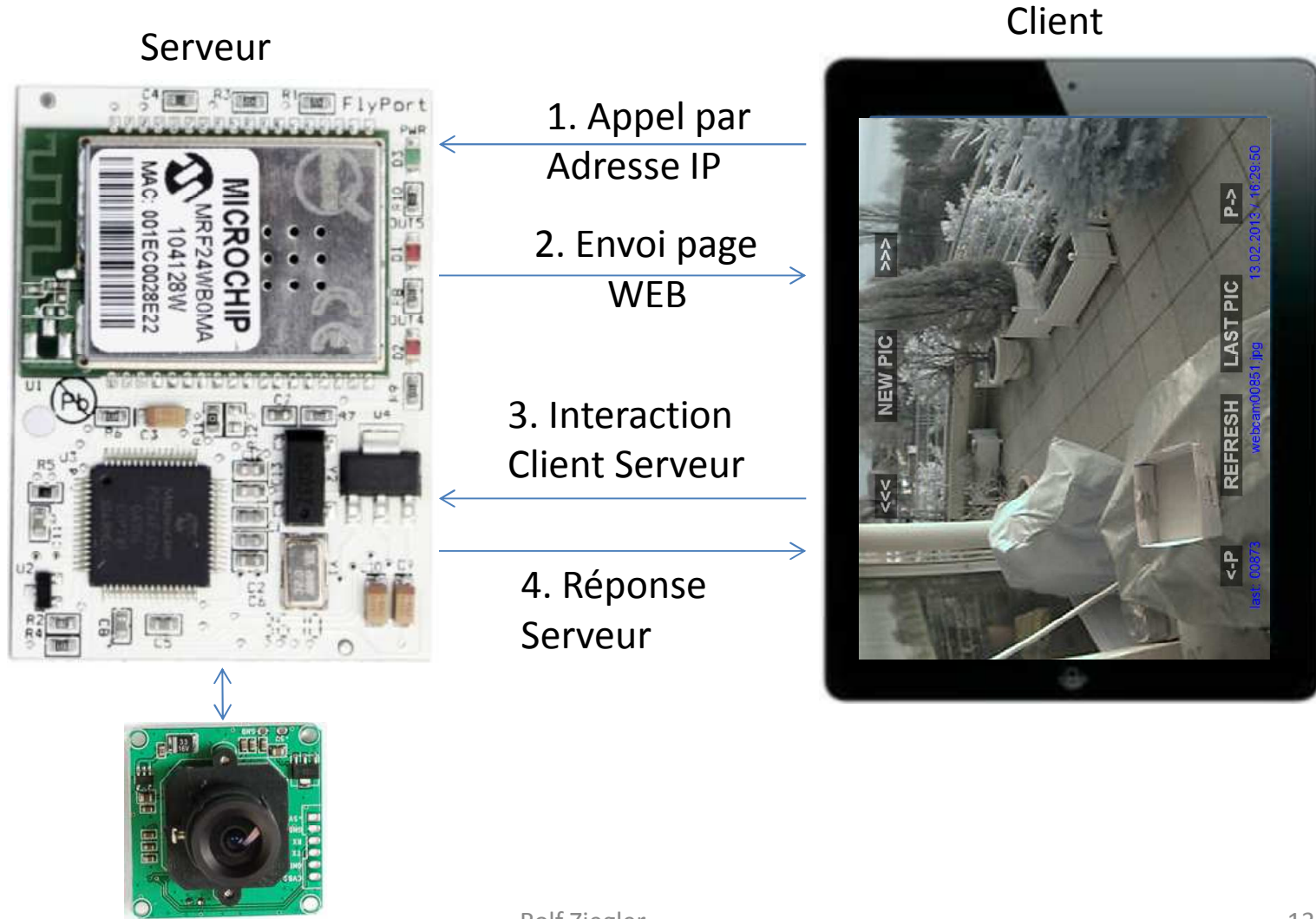
**-Changement info
page web(print)**

**-Lecture SD, envoi
vers page WEB**

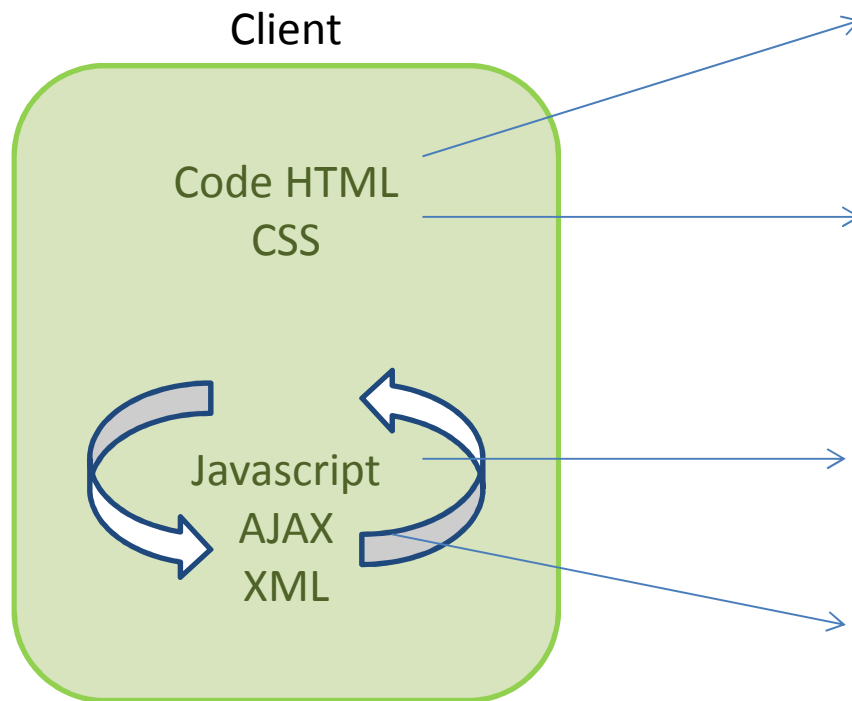
Index.htm

-Initiation page
web
-Détection iPhone,
lexplorer
-Affichage
graphique
-Code Javascript

Communication Client-Server



Contenu HTML,...



HTML:

Contenu statique de la page web

Texte, images, boutons,...

Formatage par défaut

CSS: Macro de formatage, grandeur du

Texte, largeur des lignes, position des

Objets par type d'appareil

(explorer, iphone,...)

JS: Programmation de la page web

Animation, Changement de couleurs

Envoi d'informations vers le serveur

AJAX: Asynchrone Java-Script and XML

Transfert de données

Rem: le code étant téléchargé vers le client

La performance et compatibilité est donc

dépendante de ce dernier (PC, Smartphone, tablette).

Objectif, Ecran sur client



2. Boutons permettant d'envoyer des commandes au server

3. Affichage info
de l'image

Code HTML, initialisation CSS

```
<head>
<title>www.Z-Control.ch Z-WEBCAM </title>

<!--meta name="viewport" content="width=device-width; initial-scale=1.0; maximum-scale=1.0;"-->
<!--meta name="viewport" content="width=device-width/" -->
<meta name="viewport" content="width=325px,user-scalable=yes"/>
<link rel="apple-touch-icon" href="../images/zrobot.png"/>
<link rel="apple-touch-startup-image" href="../images/Startup_picture.png" />

<meta name="apple-mobile-web-app-capable" content="yes" />
<meta name="apple-mobile-web-app-status-bar-style" content="black-translucent" />

<link rel="stylesheet" type="text/css" href="../style/iphone.css" media="only screen and (max-width: 480px)" />
<link rel="stylesheet" type="text/css" href="../style/desktop.css" media="screen and (min-width: 481px)" />
</head>
```

```
<style type="text/css">
.btn {
    position: absolute;
    border-color: #ffffff;

    font-family: Arial, Helvetica, sans-serif;
    font-size: 8pt;
    font-weight: bold;
    color: #ffffff;
    text-align: center;

    z-index: 100;
    filter: progid:DXImageTransform.Microsoft.Alpha(opacity=50);
    -moz-opacity: 0.5;
    opacity: 0.5;
}
```

Chargement dynamique
Du fichier de formatage

Code HTML & Ajax



```
<!-- WEBCAM POSITIONNEMENT DE L'IMAGE -->
<div id="display">
  
```

```
<!-- SERVO + UPDATE MENU -->
<div class="btn" id="dirbtn" style="left: 10px; top: 210px; width: 90px; height: 30px" onclick="newAJAXCommand('buttons.cgi?btn=1');
newAJAXCommand('status.xml', updateStatus, false);"><span class="btnText">&nbsp;<<<&nbsp;<<<</span></div>
<div class="btn" id="dirbtn" style="left: 220px; top: 210px; width: 90px; height: 30px" onclick="newAJAXCommand('buttons.cgi?btn=2');
newAJAXCommand('status.xml', updateStatus, false);"><span class="btnText">&nbsp;<>>>&nbsp;<>>></span></div>
<div class="btn" id="cambtn" style="left: 120px; top: 210px; width: 90px; height: 30px" onclick="newAJAXCommand('status.xml', updateStatus, false);">
<span class="btnText">&nbsp;<UPDATE&nbsp;</span></div>
```

La commande AJAX va envoyer un message HTML par TCP-IP vers
Le serveur embarqué indiquant la page web et l'identifiant (btn1 ou btn2 ou UPDATE)

Exemples de code JavaScript

```
</div-->
<small><span id="cnt" class="btnx" small style="position: absolute; left: 328px;
top: 225px; color:#FFFF00 ;">0</span></small>
<small><span id="date" class="btnx" small style="position: absolute; left: 216px;
top: 230px; color:#0000FF ;font-size:8px;">07/10/2012 00:00:00</span></small>

<script type="text/javascript">
// Parses the xmlResponse from status.xml and updates the status box
function updateStatus(xmlData) {
    var mainstat = document.getElementById('display').style.display;
    var loadstat = document.getElementById('loading').style.display;
    var imagecnt = 0;

    // Check if a timeout occurred
    if(!xmlData)
    {
        mainstat = 'none';
        loadstat = 'inline';
        return;
    }
    document.getElementById('date').innerHTML = getXMLValue(xmlData, 'val'+0);
    document.getElementById('cnt').innerHTML = getXMLValue(xmlData, 'val'+1);
}
setInterval("newAJAXCommand('status.xml', updateStatus, false)",500);
</script>
```

Objectif: rafraichir toutes les 500ms les noms de fichiers en bas d'écran
l'information se trouve dans le fichier status.xml qui est échangé
avec le système embarqué.

Résumé mise a jour image par XML

2. Le serveur embarqué répond avec une mise a jour des valeurs ou d'une image



« status.xml »

```
GET http://192.168.1.122/status.xml 200 OK 94ms
En-têtes Réponse XML
<response>
<led0>0</led0>
<led1>0</led1>
<btn0>dn</btn0>
<btn1>up</btn1>
<btn2>dn</btn2>
<btn3>dn</btn3>
<btn4>dn</btn4>
<btn5>dn</btn5>
<btn6>dn</btn6>
<var0>last: 00183</var0>
<var1>08.03.2013 / 14:46:02
</var1>
<var2>webcam00182.jpg</var2>
</response>
```

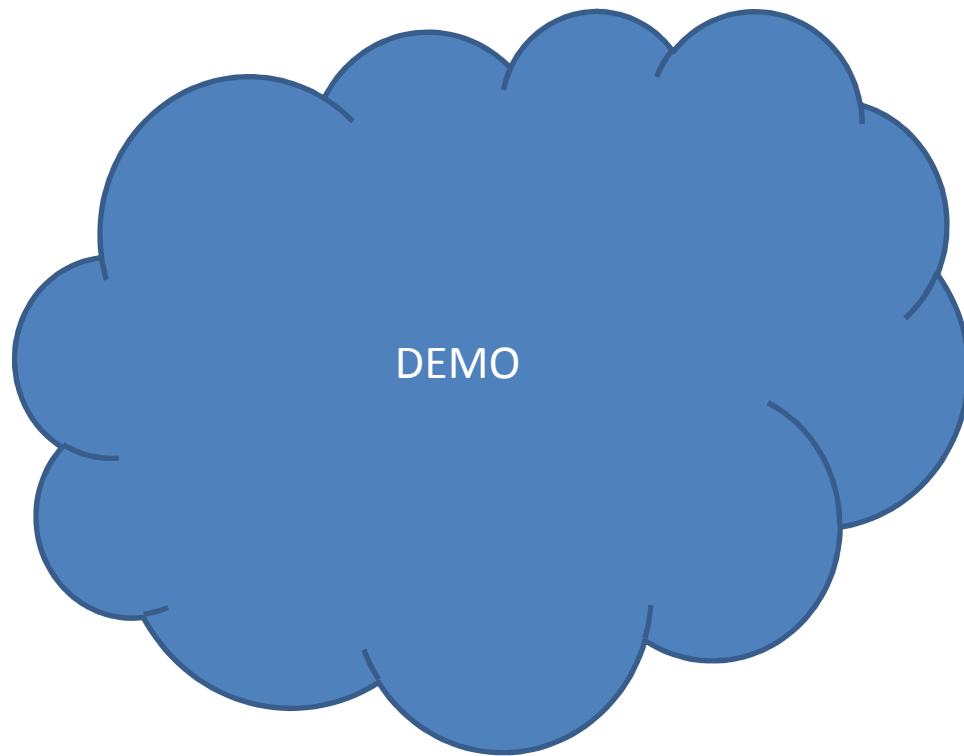
« cam.xml »

```
<?xml version="1.0"?>
<response>
  <cam0>~cam~</cam0>
</response>
```



1. HTML+ JavaScript
Appelle toutes les x millisecondes
un rafraichissement du fichier
XML





Problèmes transfert d'image

XML ne permet pas d'échanger des informations binaires

Il fallait donc trouver une solution pour envoyer les données binaires sans trop d'effort



La camera crée un fichier JPEG
de 10-50kb

JPEG=fichier binaire

Solution: On va donc encoder le
fichier binaire en base 64
Ce format utilise 6 bits donc ascii
3x8 bits (24) seront donc envoyés en
4 caractères ascii (4x6bits=24bits)

9-Feb-13

Rolf Ziegler

Côté client (page web)

On a de la chance, les exploreurs web
Sont capable de décompresser du base64
suffisait plus que trouver le code
Java-Script qui effectuerait l'opération.

22



Origine

- ✓ Camera de surveillance
- ✓ WEBCAM
- ✓ Connectée par wifi
- ✓ par routeur wifi local
- ✓ Si possible accessible depuis le web public
- ✓ Sécurité d'accès

Résultats



Ajouts

- ✓ Stockage sur carte SD
- ✓ Date+Heure Internet (temps réel SNTP)
- ✓ Mouvement par servo-moteur
- Version FTP stockage des photos dans le « cloud »

Installation sur ma terrasse



9-Feb-13

7-Control

Rolf Ziegler

24

Examples



9-Feb-13

Rolf Ziegler

25

Références

Explications et simulation HTML en ligne

<http://www.w3schools.com/html/default.asp>

Web-Serveur embarqué www.openpicus.com

Exemple, code source http://wiki.openpicus.com/index.php?title=Community_Projects

Camera, matériel: www.adafruit.com

