



---

# LES OBJETS COMMUNICANTS

DÉMARCHE DE CONCEPTION D'UNE APPLICATION  
UTILISANT DES OBJETS CONNECTÉS

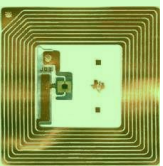
Jean-Paul Jamont, Lionel Médini, Michaël Mrissa



# PLAN DU COURS

---

1. Anatomie d'un objet communicant
2. Quelle démarche de conception ?
3. Analyse des besoins
4. Processus de développement
5. Validation

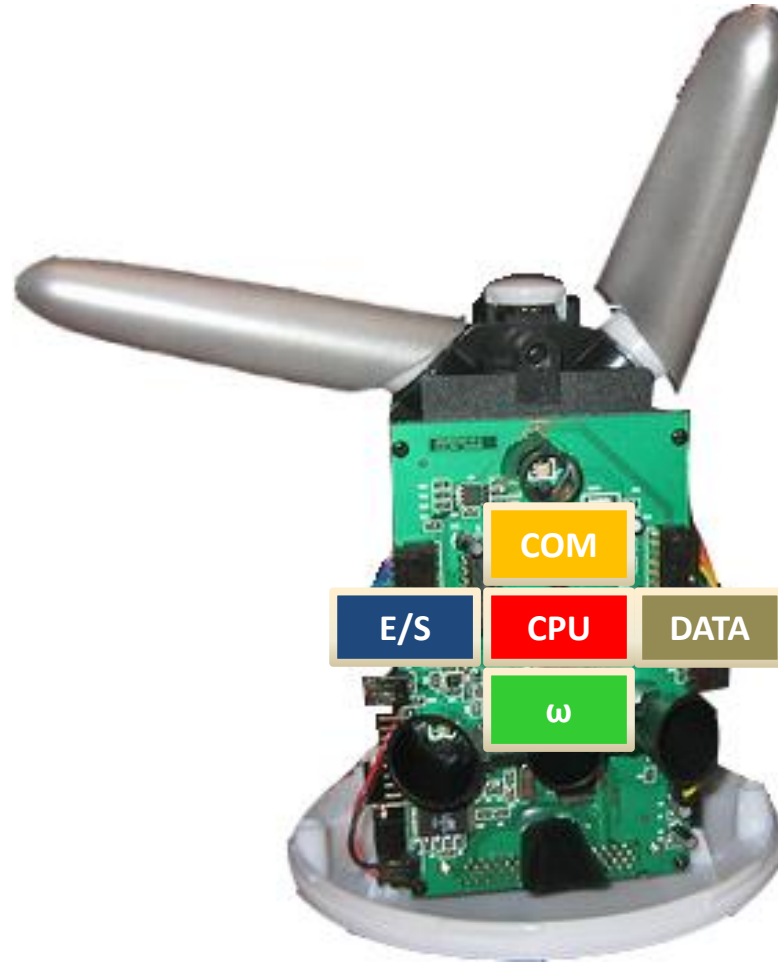






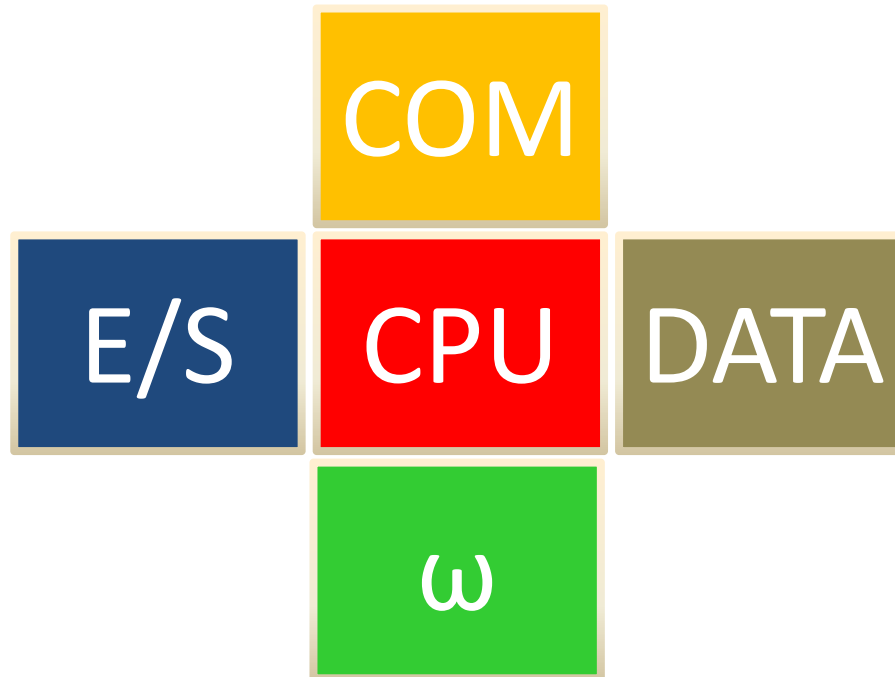
# 1. COMMENT FONCTIONNENT LES OBJETS COMMUNICANTS?

---



# 1. COMMENT FONCTIONNENT LES OBJETS COMMUNICANTS?

---



# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

---

CPU

1. PDA / Mini-PC / PC Industriels
2. Une carte de développement **existante**
3. From **scratch**





# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- PDA / Mini-PC / PC Industriels



Modèle: Samsung Galaxy S4GT-I9505

OS: Android 4.2.2 Jelly Bean

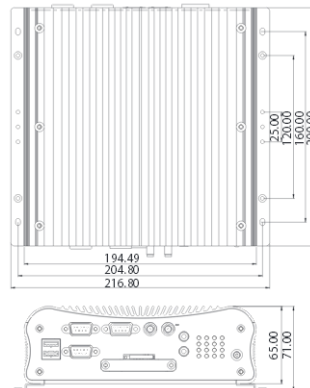
CPU,GPU: 4 cœurs Qualcomm Snapdragon 600 (ARMv7) à 1,9 GHz, Qualcomm Adreno 320

Mémoire vive: 2 Go de RAM LPDDR3

Stockage: 32 Go de mémoire flash intégrée

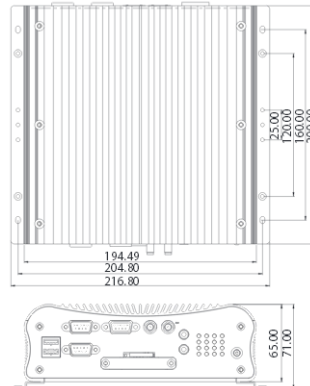
Connectique: HDMI, prise casque jack 3,5 mm, port micro USB 2.0 compatible MHL 2.0, WiFi 802.11 a/b/g/n/ac (HT80), NFC, Bluetooth 4.0 (LE), LED IR3

Capteurs: de proximité, de lumière, IR, thermomètre, hygromètre, baromètre, accéléromètre et gyroscope à 3 axes, magnétomètre



# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- PDA / Mini-PC / PC Industriels



Modèle: Android Mini PC MK802

OS: Android 4.2.2 Jelly Bean ou Ubuntu ou PicUntu

CPU,GPU: Cortex-A9 à 1.6 GHz, 400 MHz Mali GPU

Mémoire vive: 2 Go de RAM LPDDR3

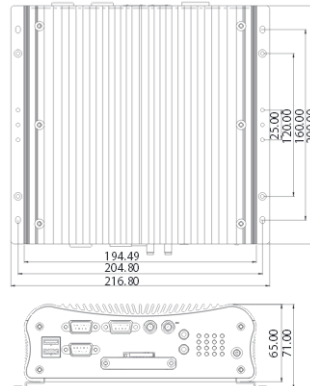
Stockage: 8 Go de mémoire flash intégrée

Connectique: HDMI, micro-USB 2.0, USB 2.0, microSD slot, alim. via micro-USB OTG, Wi-Fi 802.11 b/g/n, Bluetooth,

Capteurs:

# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- PDA / Mini-PC / PC Industriels



Modèle: PC industriel fanless NISE2200

OS: Linux, Windows ...

CPU,GPU: Intel® Atom™ Dual Core D2550 1.86GHz

Mémoire vive: 8Go DDR3 SODIMM

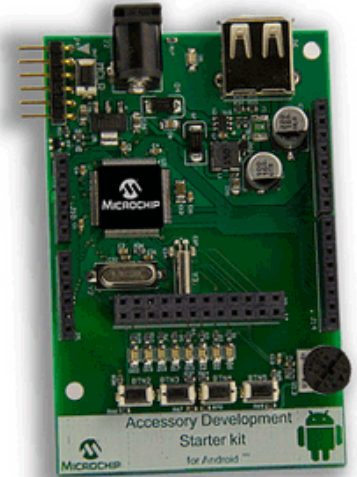
Stockage: Disque dur 2.5" SATA

Connectique: 6 ports USB2.0; 1 emplacement CFast; 1 emplacement carte SIM, 2 ports RS-232/422/485 isolés, 2 ports Ethernet Intel® 82574IT Gb, 1 port DB15 E/S , WiFi 802.11 a/b/g/n/ac or 3.5G (auto detected modules), Support 9-36V DV input, Audio Jack

Capteurs:

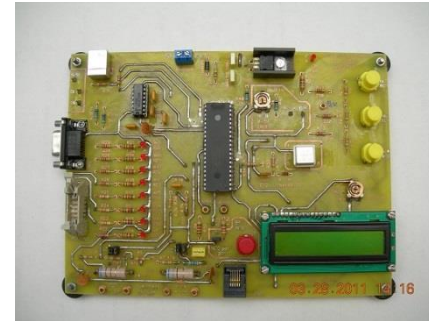
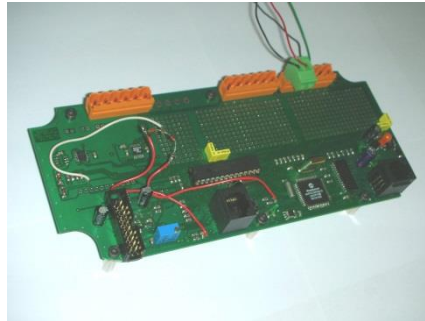
# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- Une carte de développement **existante**

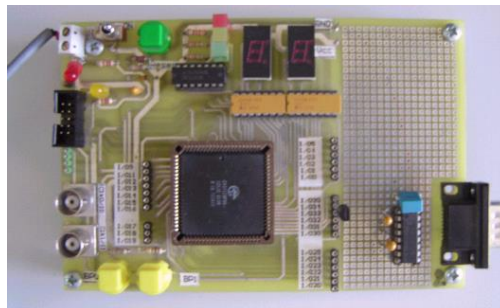


# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- From **scratch**
  - Architecture centrée **microcontrôleur**

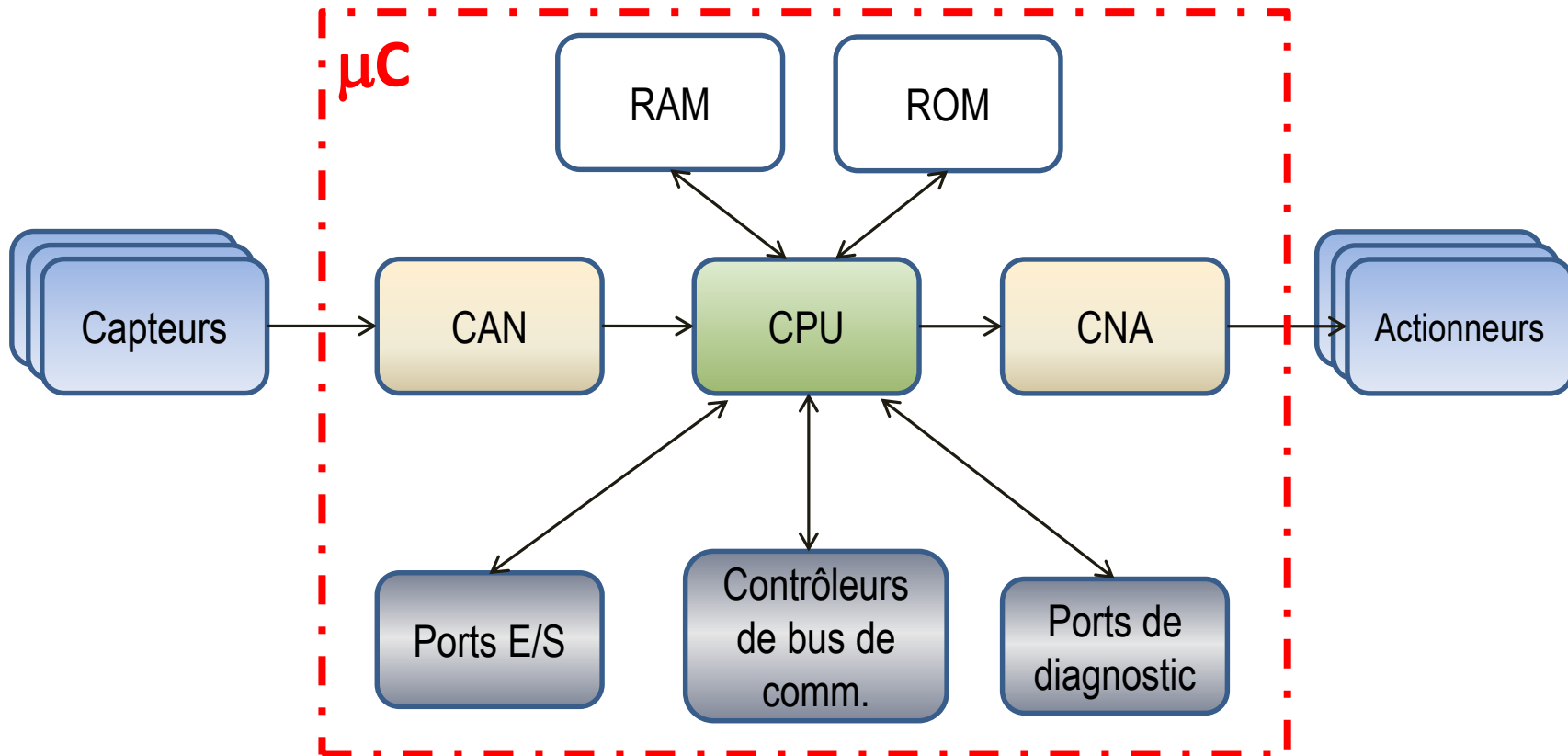


- Architecture centrée **FPGA/EPLD/ASIC?**



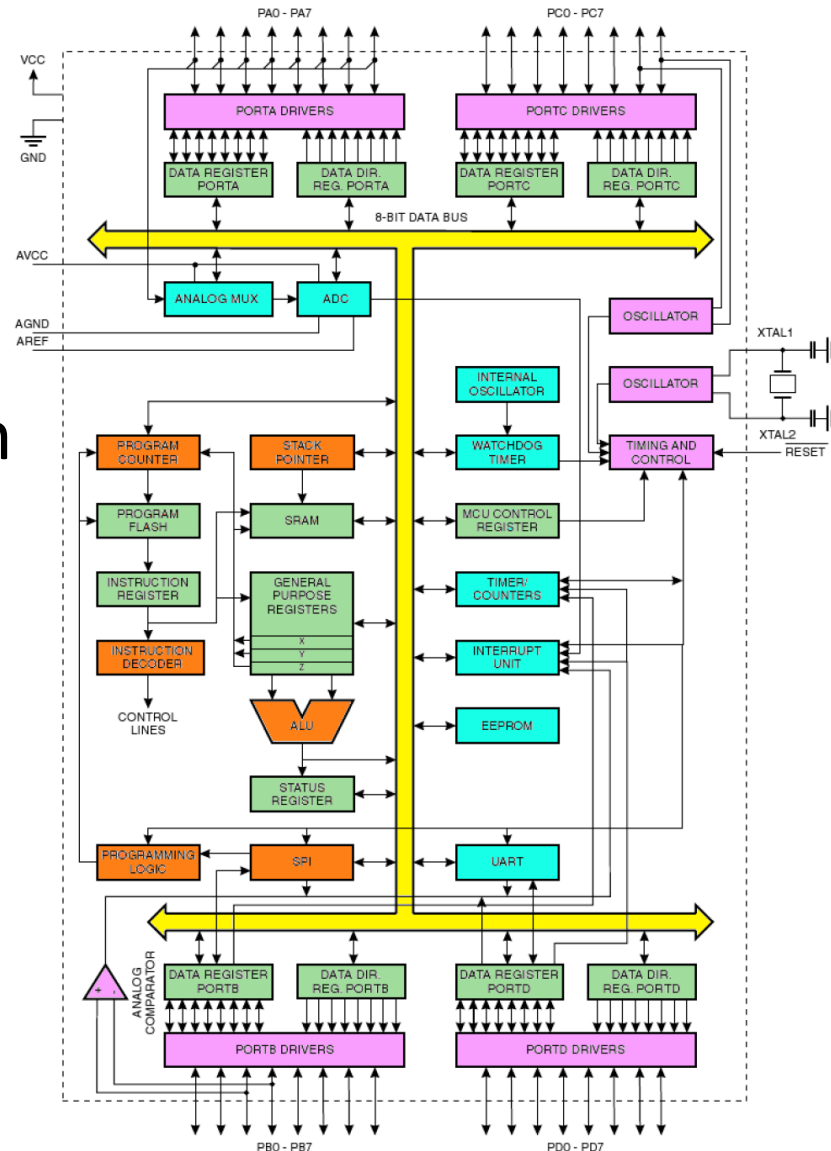
# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- Architecture d'objet communicant basée sur un microprocesseur ou un microcontrôleur



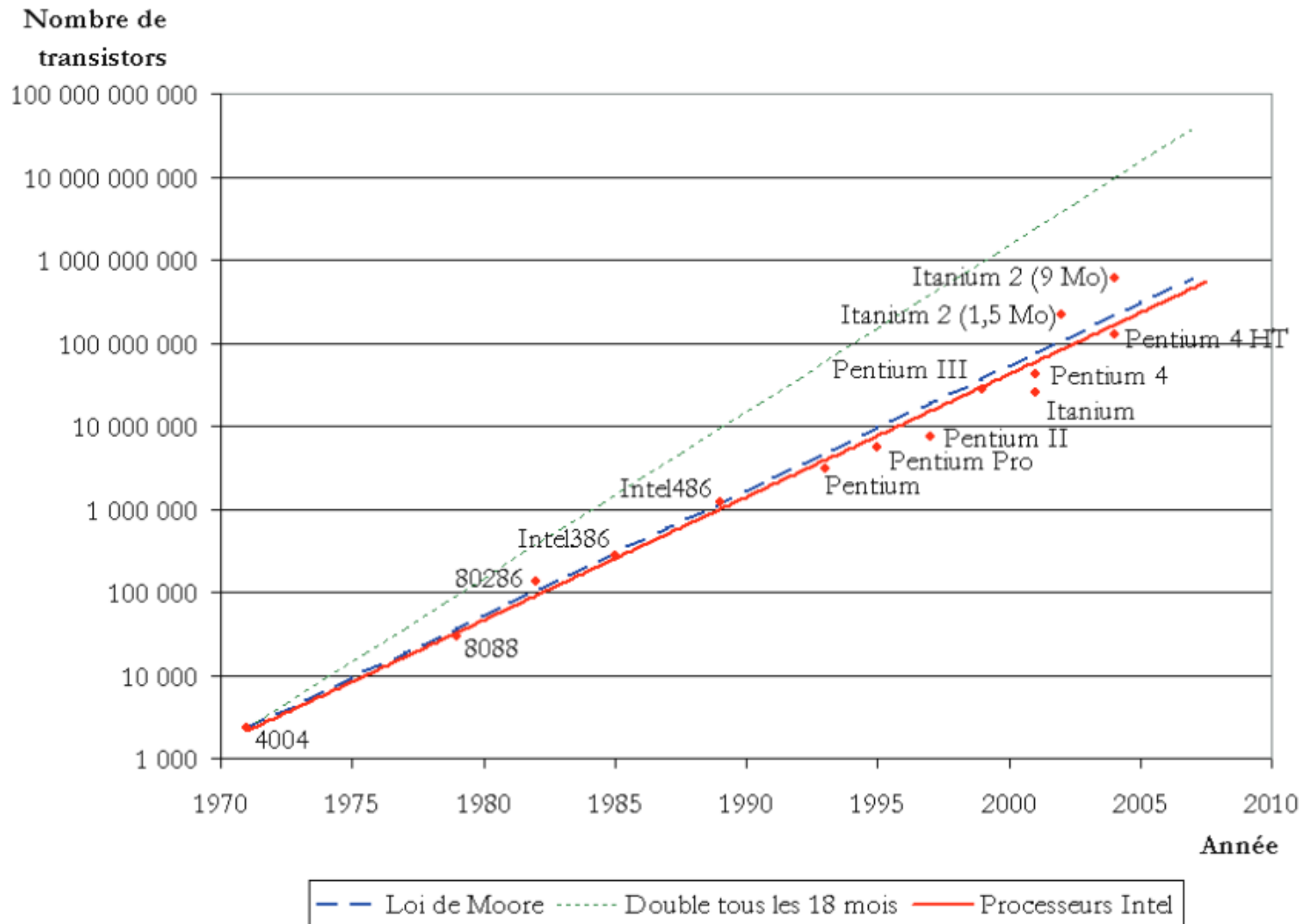
# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- Processeur
- Mémoire
- Périphériques
- Bus de communication
- Entrées/Sorties



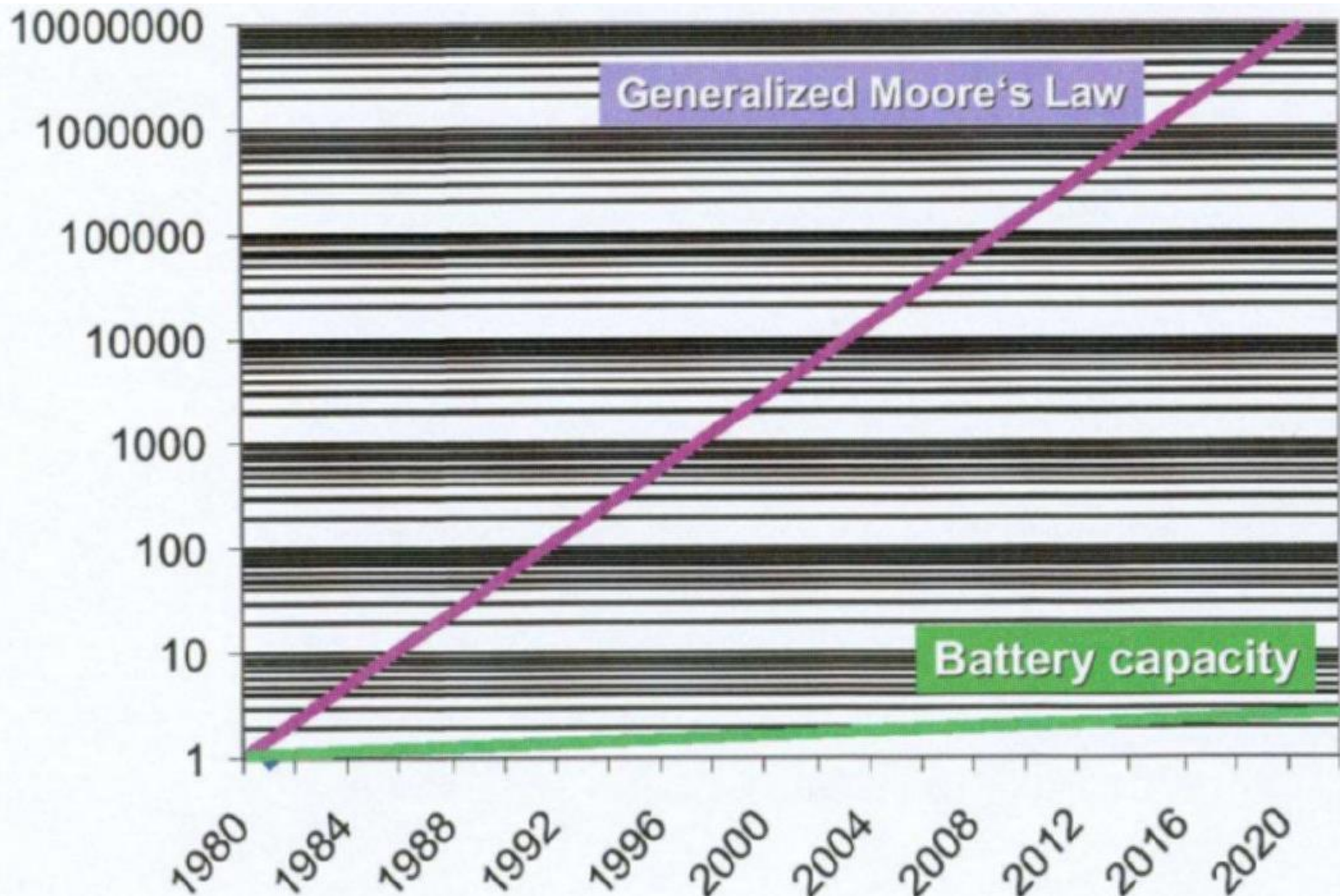
D'après Julien DeAntoni

# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?



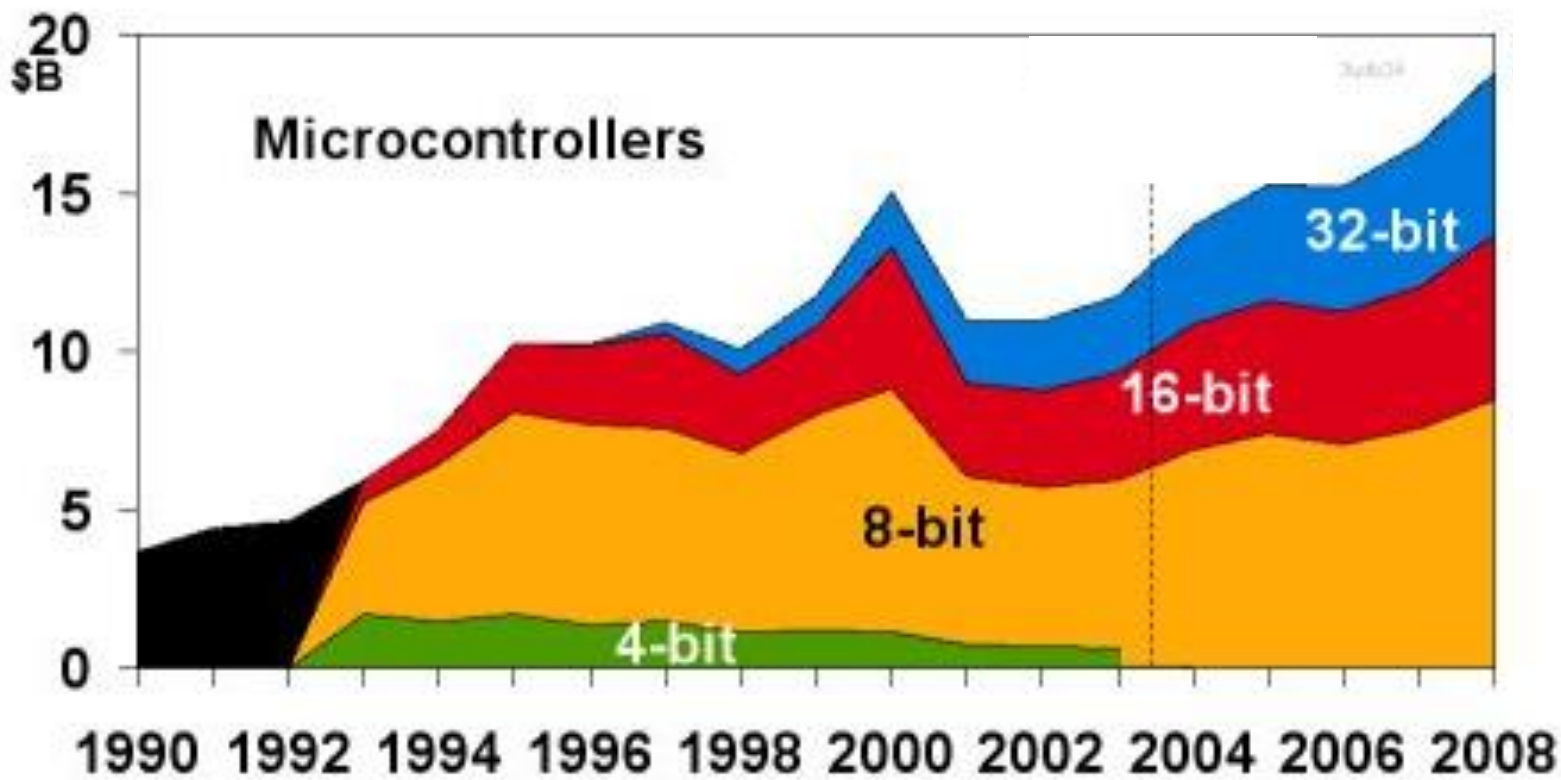


# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?



Source: Jan Raba

# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?



Source: Gartner Dataquest



# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?



Main - Mozilla Firefox

Fichier Édition Affichage Historique Marque-pages Outils ?

Main +

Page précédente Page suivante

www.microchip.com/maps/main.aspx

Actualiser Arrêter


microchip sele

Téléchargements Accueil Firebug

Les plus visités À la une Conférences acceptin... http://ade52-upmf.gr... Zimbra: Réception (18)

**MAPS** Microchip Advanced Part Selector **MICROCHIP**

### About Microchip Advanced Part Selector



**Microchip Advanced Part Selector**  
MAPS

**Copyright © Microchip Technology Incorporated**

Microchip Technology Inc.  
2355 West Chandler Blvd.  
Chandler, AZ 85224-1699  
Ph:480-792-7200 Fax:480-899-9210  
www.microchip.com

The information contained in this media regarding device specification or pricing and the like is intended through suggestion only and may be superseded by updates.

The content of the database is representative of various manufacturers' product specifications and contains the latest product specification information available to Microchip Technology Inc. at the time of MAPS last update. The product specifications in MAPS are subject to change without notice.

It is your responsibility to ensure that your product selection meets with your system specifications.

All trademarks mentioned herein are property of their respective companies.

**Where would you like to start?**

Analog Interface Memory Microcontroller Wireless

OK

Rechercher : jamont

Suivant Précédent Tout surligner Respecter la casse

# QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

Microcontroller - Mozilla Firefox

www.microchip.com/maps/microcontroller.aspx

Parameter Search

Match ALL (AND) Match ANY

Family: -All- 8-bit 16-bit 32-bit

Prefix: -All-

P.Memory(Kbytes): -All- -All-

P.Memory(KWords): -All- -All-

RAM(Bytes): -All- -All-

EEPROM(Bytes): -All- -All-

I/O Pins: -All- -All-

Max CPU Speed: -All- -All-

Int. OSC: -All- -All-

Comparator: -All- -All-

A/D Ch.: -All- -All-

A/D Bits: -All- -All-

D/A Ch.: -All- -All-

UART Ch.: -All- -All-

SPI: -All- -All-

I<sup>2</sup>C™: -All- -All-

CAN Ch.: -All- -All-

USB Ch.: -All- -All-

Input Capture: -All- -All-

PWM Ch.: -All- -All-

MtrCtrl PWM Ch.: -All- -All-

SMPS PWM Ch.: -All- -All-

Search Results: 819 MCHP Parts found

dsPIC33EP512MU814

dsPIC33EP512MU814 In Production

Specifications	Dev Tools	Technical Docs	Budgetary Pricing
P.Memory (Kbytes)			512 Flash
P.Memory (KWords)			170
Self-Write Flash			Yes
RAM (Bytes)			52K
EEPROM (Bytes)			0
DMA Ram			4096
Auxiliary/Boot Flash			24
I/O Pins			122
Max CPU Speed			140 MHz (70 MIPS)
Internal OSC			7.37 MHz, 32.768 kHz
Code Guard™ Security			Basic
System Mgmt Features			BDR, None, PDR, WDT, WUR, 15-DMA, nanoWatt [Low Sleep, Fast Wake, Pwr Modes]
Analog Peripherals			3-Comparators, Bandgap - No, OpAmp; 2A/D, 32x12-bit @ 500kps; 1D/A, 0x4-bit @ kpsps
Digital Comm. Peripherals			4-UART, 4-SPI, 2-I2C, AC97, I2S, PPS
Connectivity			1-Full Speed (Device, Host, OTG)-USB 2.0 OTG, 2-CAN, None, LIN, IrDA
Capture/Compare PWM Peripherals			16-Output Comp. & Std. PWM, 16-bitPWM, 16-Input Capture
Digital Timers			9x16-bit, 4x32-bit
Application Peripherals			14-MtrCtrl, PWM, 14-PS_PWM, 2-QEI, PMP, EBI-No
Debug/Development Features			JTAG-Boundary Scan, ICSP, ICDdebug - Yes
Package (Pins)			LQFP, TQFP (144)
Operating Voltage			(3V-3.6V)
Temperature Ranges			(-40 to +125)
Min PCB Area (mm2)			

Microchip DIRECT

Global Part Search

Selection Tools Home

Send Email

More Links

Select a part in the search results, press one of the buttons below to view

dsPIC33EP512MU814

Rechercher : jamont

Suivant Précédent Tout surligner Respecter la casse

# COMMENT MON OBJET ACCÈDE À INTERNET?

COM

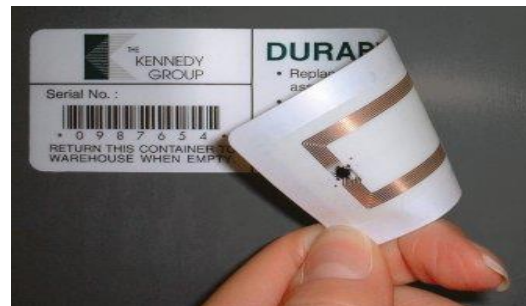
- Deux grands types d'objets physiques

- Objets communicants et/ou intelligents

- A base de micro-contrôleur(s) /FPGA équipés d'interface(s) de communication
    - Modèle de **comportement embarqué** sur l'objet

- Objets « chipless » tagués

- Objets sur lesquels on a apposé une étiquette RFID (ou autre)

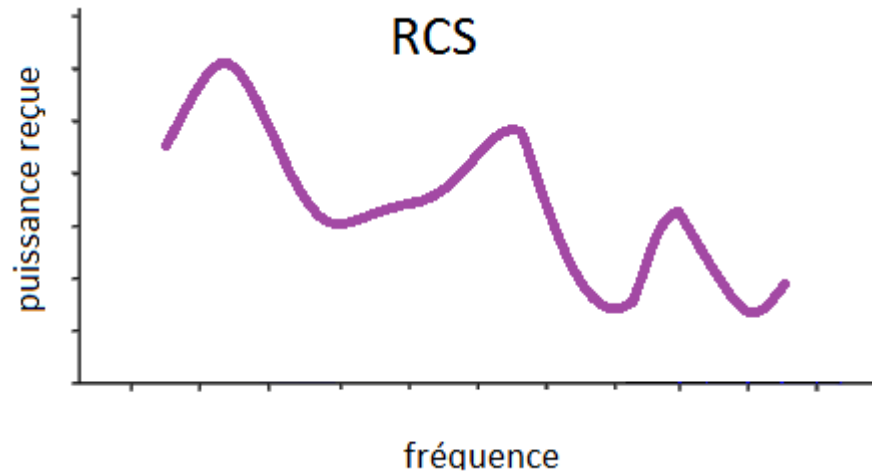


- Modèle de **COMPORTEMENT DÉPORTÉ** sur un serveur distant



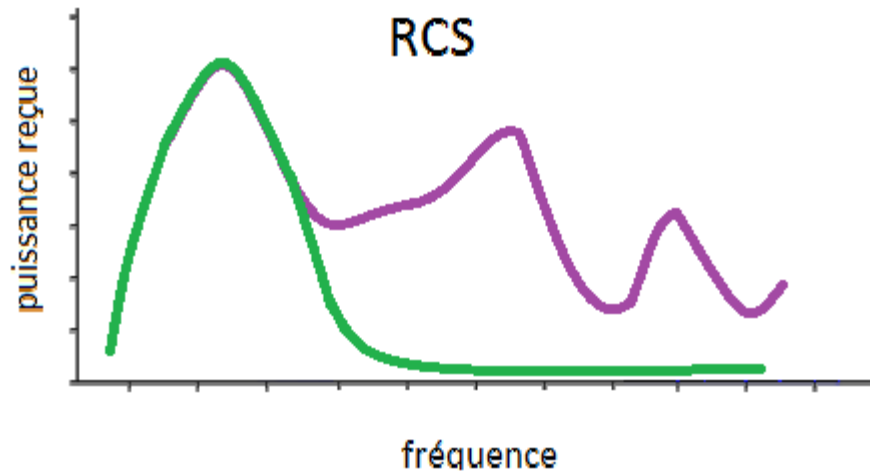
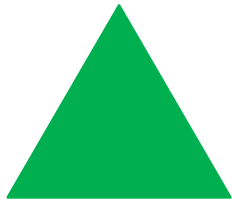


# OBJETS CHIPLESS

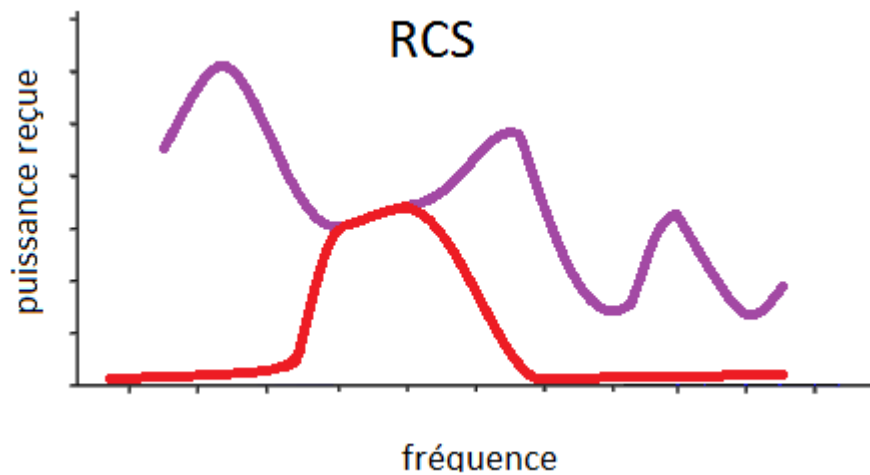




# OBJETS CHIPLESS



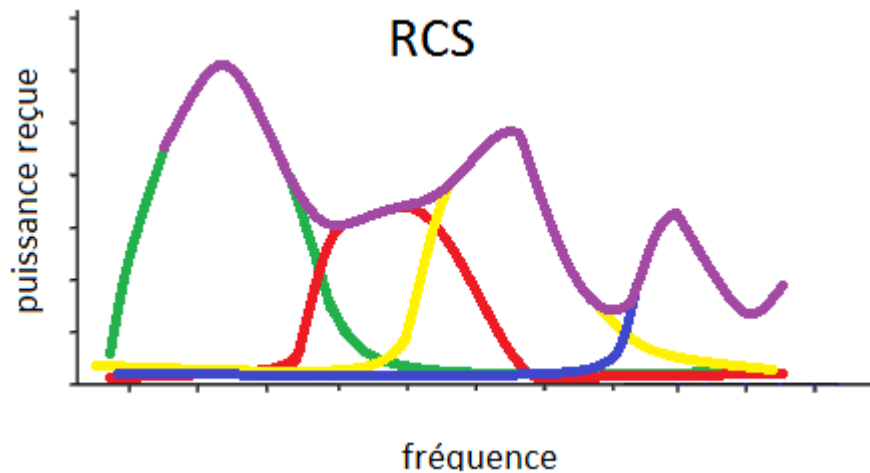
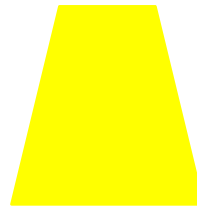
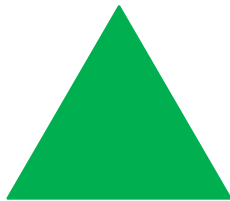
# OBJETS CHIPLESS







# OBJETS CHIPLESS

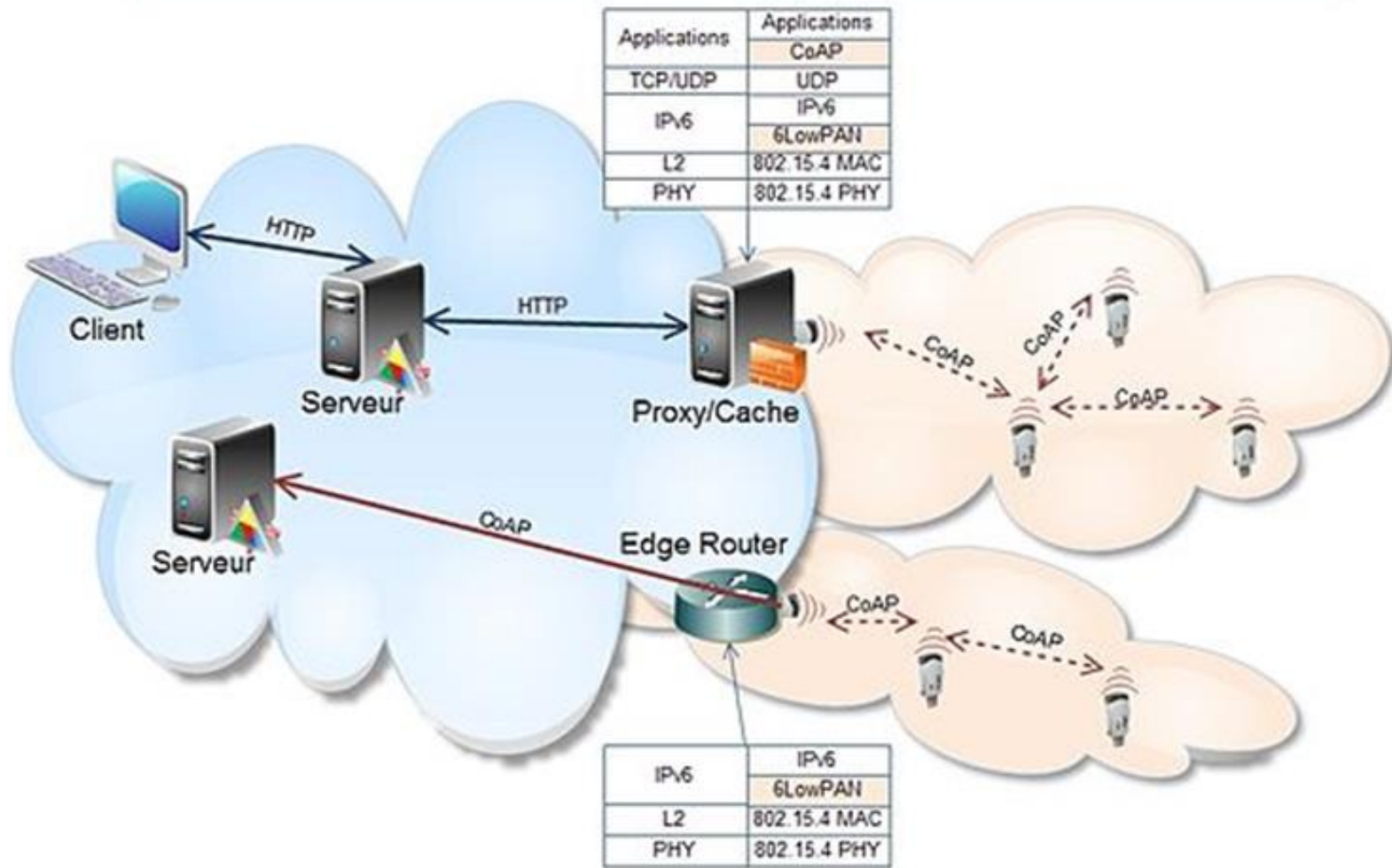


# MODÈLES DE COMMUNICATION

Comportement embarqué

Comportement déporté

## Architecture REST



# COMMENT VONT COMMUNIQUER LES OBJETS ENTRE EUX?

- Les réseaux sans fil sont des **bricks de base** de l'*IoT* et du *WoT*
  - ⇒ Il faut savoir identifier les critères pertinents qui vont conditionner le choix de ces bricks
  - ⇒ Il faut comprendre la caractérisation de ces bricks
- Il est indispensable de comprendre les couches basses d'un **système communicant** pour assurer une réelle **maîtrise d'ouvrage**
- Les couches basses impactent profondément la **qualité de service** d'un système communicant! En effet, il existe un **fort couplage** entre les technologies utilisées pour les couches basses et :
  - La consommation d'énergie,
  - Le déterminisme (ou non) temporel,
  - Le débit (couplé avec l'environnement du système)
  - La précision d'une radiolocalisation
  - Le taux de perte
  - Le déséquilibrage
  - La latence
  - La gigue
  - ...

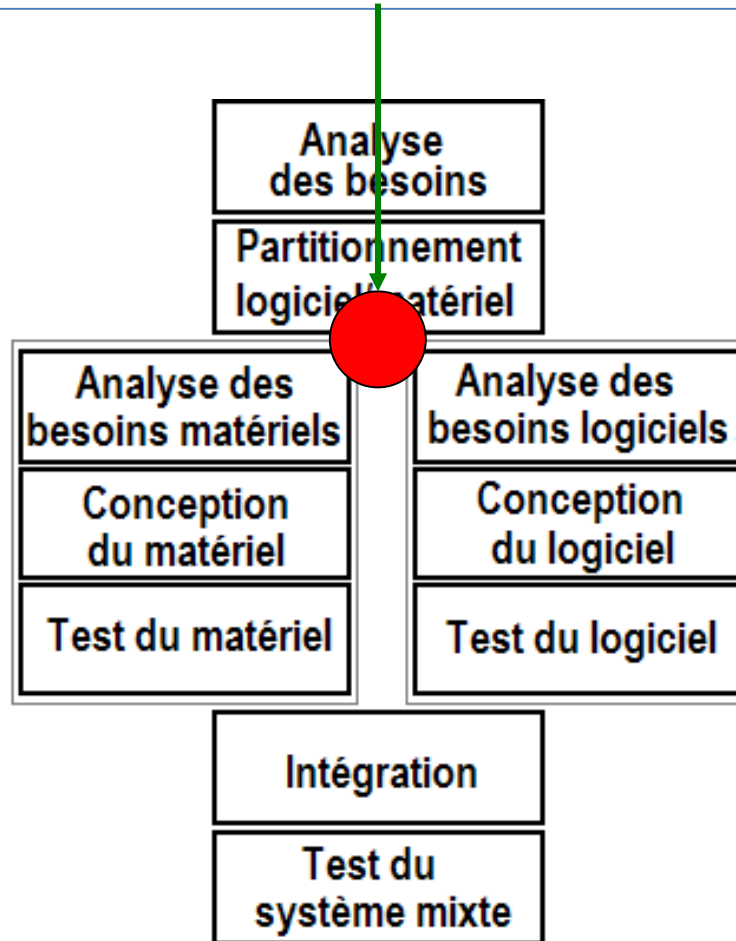
# COMMENT VONT COMMUNIQUER LES OBJETS ENTRE EUX?



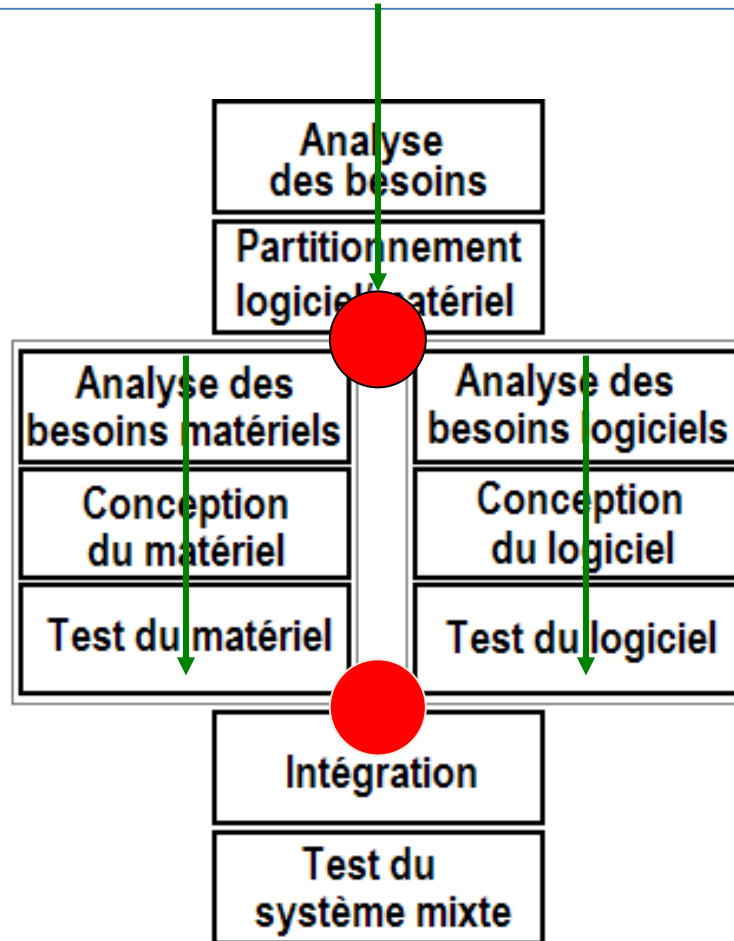
	<b>Zigbee</b>	<b>Bluetooth</b>	<b>Wi-Fi</b>
Besoins en mémoire	4-32 Kb	250 Kb	1 Mb
Autonomie avec pile	Années	Jours	Heures
Nombre de nœuds	65 000+	7	32
Vitesse de transfert	250 Kb/s	1 Mb/s	11-54-108-... Mb/s
Portée	10-100m	10-100 m	300 m



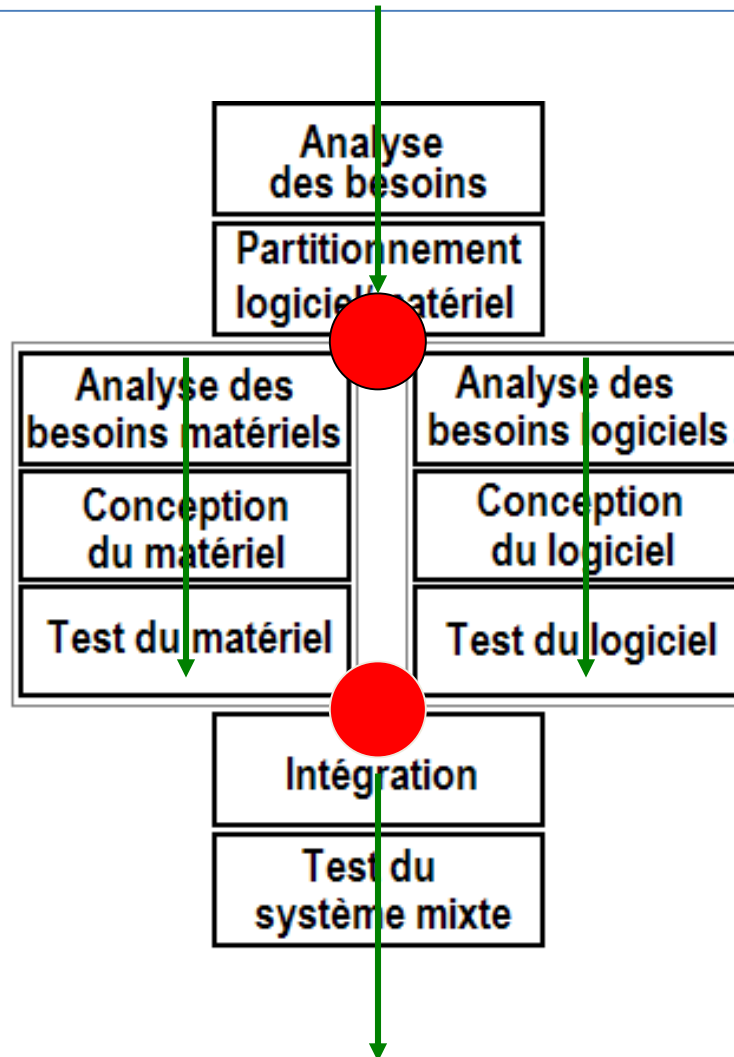
# 2. QUELLE DÉMARCHE?



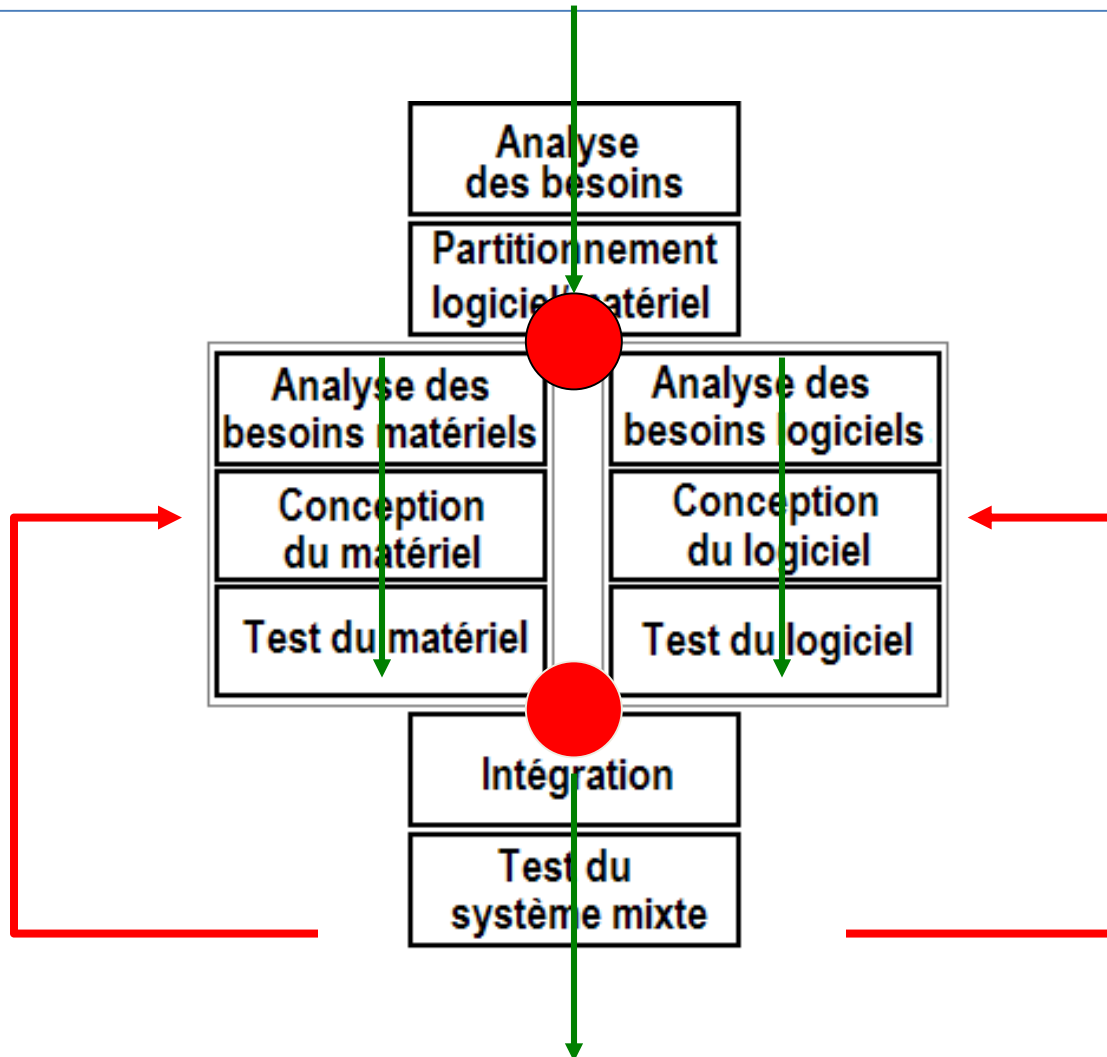
# 2. QUELLE DÉMARCHE?



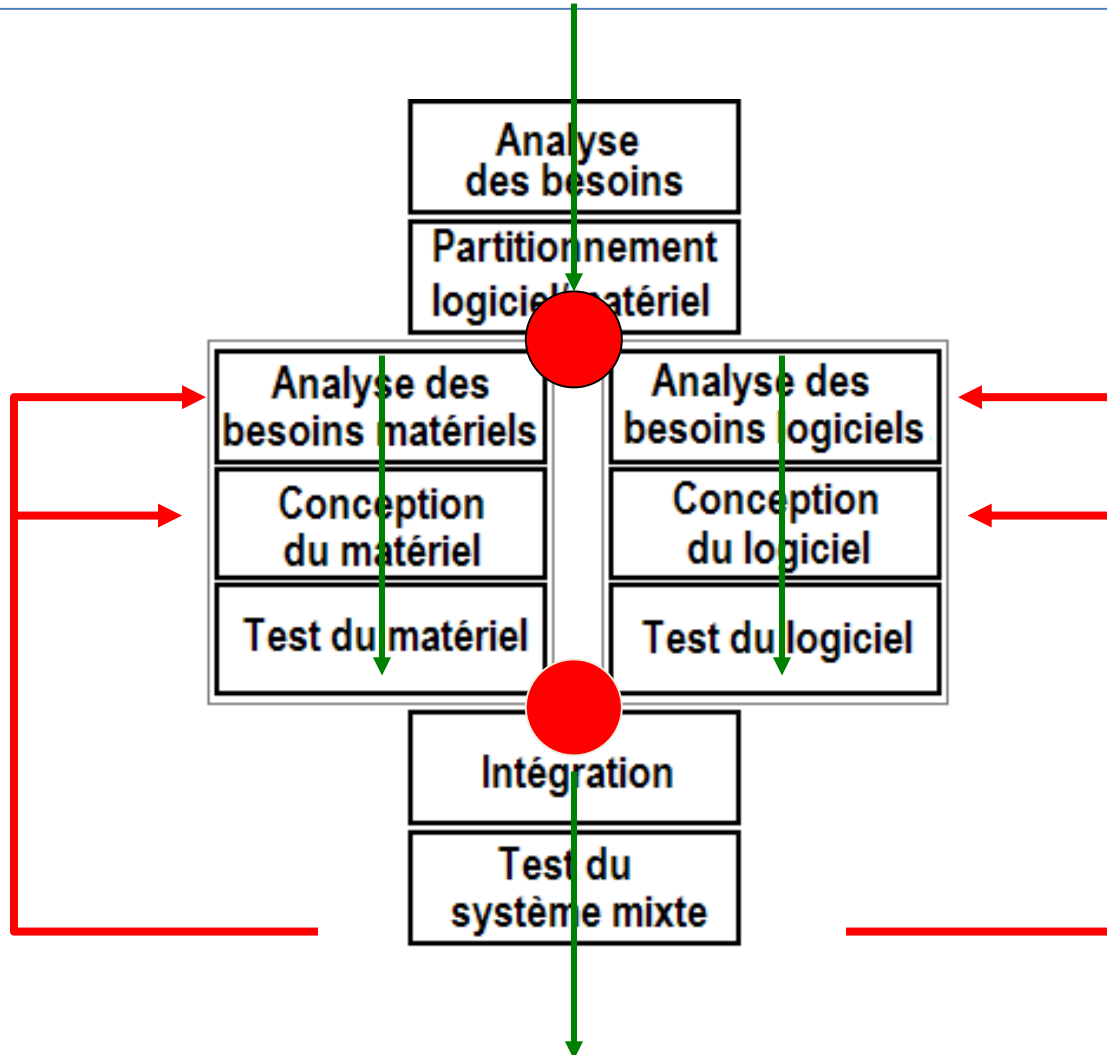
# 2. QUELLE DÉMARCHE?



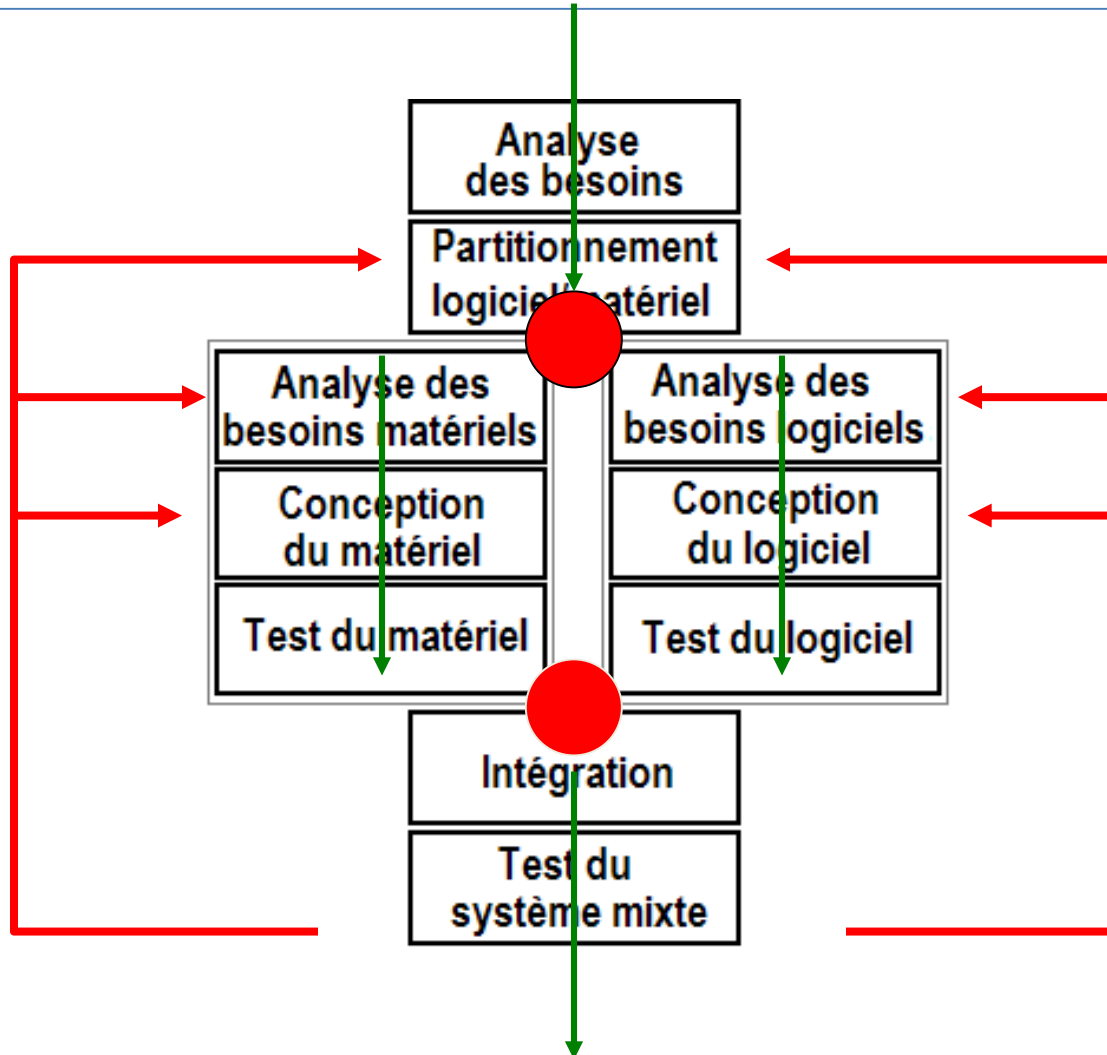
# 2. QUELLE DÉMARCHE?



# 2. QUELLE DÉMARCHE?

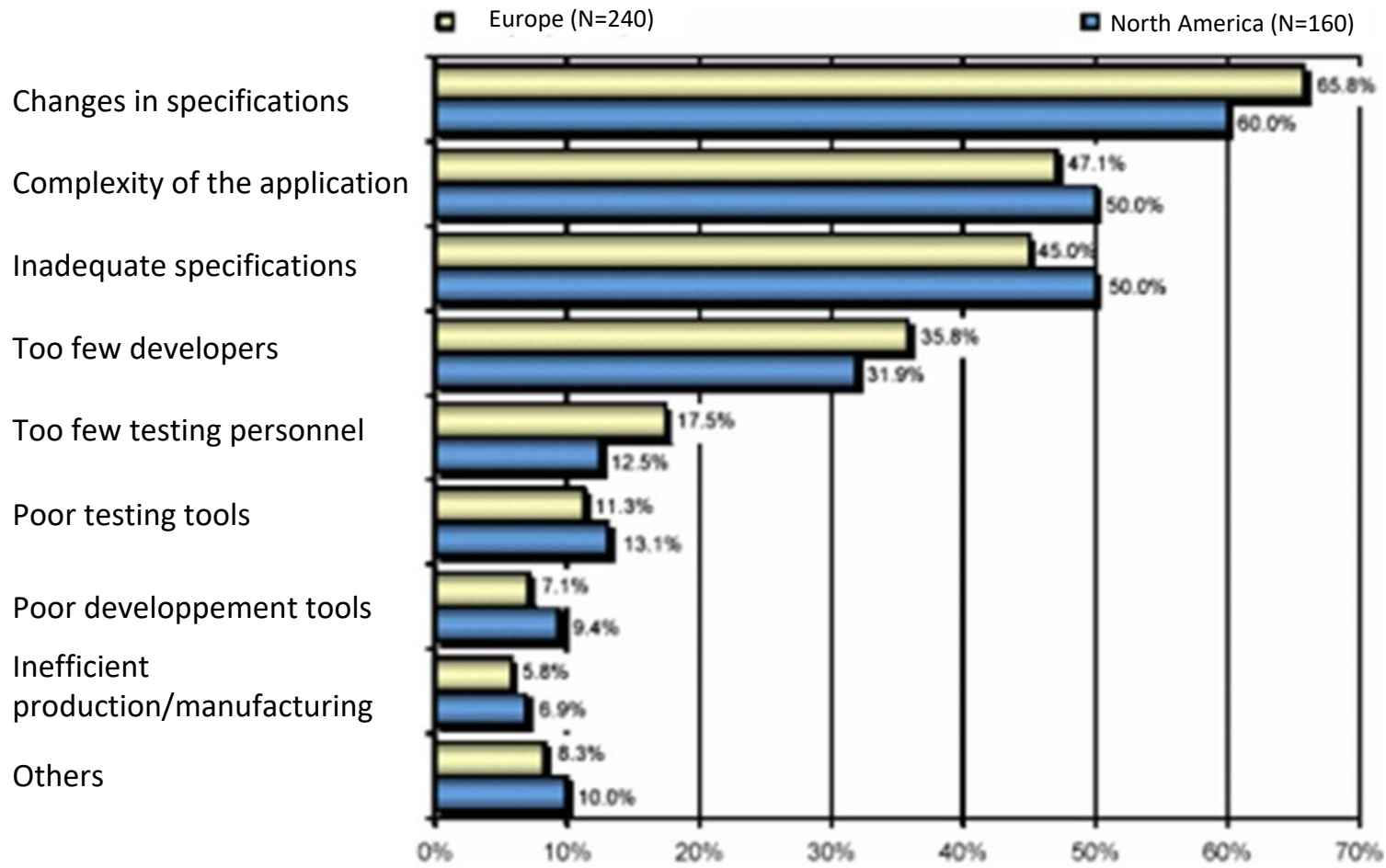


# 2. QUELLE DÉMARCHE?



## 2. QUELLE DÉMARCHE?

71.5% of traditional embedded system designs were not within 30% of pre-design performance expectations. [1]



# 2. QUELLE DÉMARCHE?

**Recueil  
des besoins**

**Analyse**

**Conception  
générique**

**Partitionnement  
logiciel/matériel**

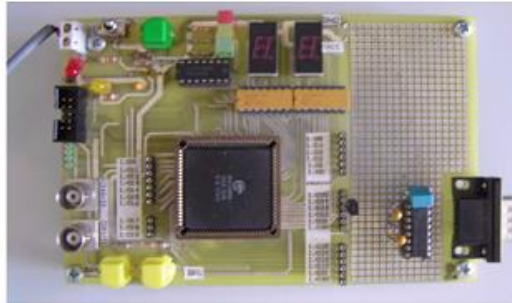
**Test**





# 2. QUELLE DÉMARCHE?

**SOLUTION MATERIELLE  
CARTE A BASE D'EPLD/FPGA**



**SOLUTION LOGICIELLE  
CARTE A MICROPROSSEUR/MICROCONTROLEUR**



**SYNTHESE MATERIELLE**

```
1 -- COMPONENT Gestion role
2 -- Version 0.95a
3 -- WARNING: Default parameter use = Synchronous version =
4 -- State translation :
5 -- S1 by vector "000"
6 -- S2 by vector "001"
7 -- S3 by vector "010"
8 -- S4 by vector "011"
9 -- S5 by vector "100"
10
11 library ieee;
12 use ieee.std_logic_1164.all;
13 use work.std_arith.all;
14
15 -- Component description
16 entity GestionRole is port(
17     reset : in std_logic;
18     NVR : in std_logic_vector(7 downto 0);
19     role : in out std_logic_vector(1 downto 0);
20     -- Call to function "election"
21     electionIn : in std_logic_bit;
22     electionOut : in std_logic_bit;
23 end GestionRole;
24
25 -- Component architecture
26 architecture arch_GestionRole is
27     signal pastState:std_logic_vector(2 downto 0);
28     signal state:std_logic_vector(2 downto 0);
29     signal futureState:std_logic_vector(2 downto 0);
30 begin
31     exec:process(clk)
32     begin
33         if(clk'event and clk='1') then
34             if(reset='1') then
35                 state<="000";
36             else
37                 case state is
38                     -- State S1
39                     when "000" =>
40                         if (NVR="00000001" then
41                             futurState = "001";
```

**SYNTHESE LOGICIELLE**

```
6 String futurState = "";
7
8
9 public void exec(){
10 //FIRST STATE ENTRY
11     for ( : ){
12         switch(presentState){
13             case(S1):
14                 if(NVR == 1)
15                     futurState = "S2";
16
17                 if(NVR == 2)
18                     futurState = "S3";
19
20 //ACTION
21         if (presentState != futureState)
22             //EXIT
23         else if (presentState != pastState)
24             //ENTRY
25         else
26             role = awncun;//STATE
27             break;
28         case(S2):
29             if(NVR==1)
30                 futurState = "D";
```

# 3. QUELS SONT MES BESOINS?

---

Pour quel type d'application l'objet que je dois construire va me servir?

1. Calcul généraliste
2. Contrôle de systèmes
3. Traitement du signal
4. Réseaux et communications



# 3. QUELS SONT MES BESOINS?

---

Quels sont mes **besoins** / mes **contraintes**?

- Puissance de **calcul**
- Capacité de **communication**
- Capacité de **stockage**
- Consommation d'**énergie**
- Temps de réponse
- Tolérances aux **pannes**
- Durée de vie
- Testabilité/débogage
- Nombre d'unités produites





# 3. VAIS-JE UTILISER UN OS?

Slicing the operating systems pie

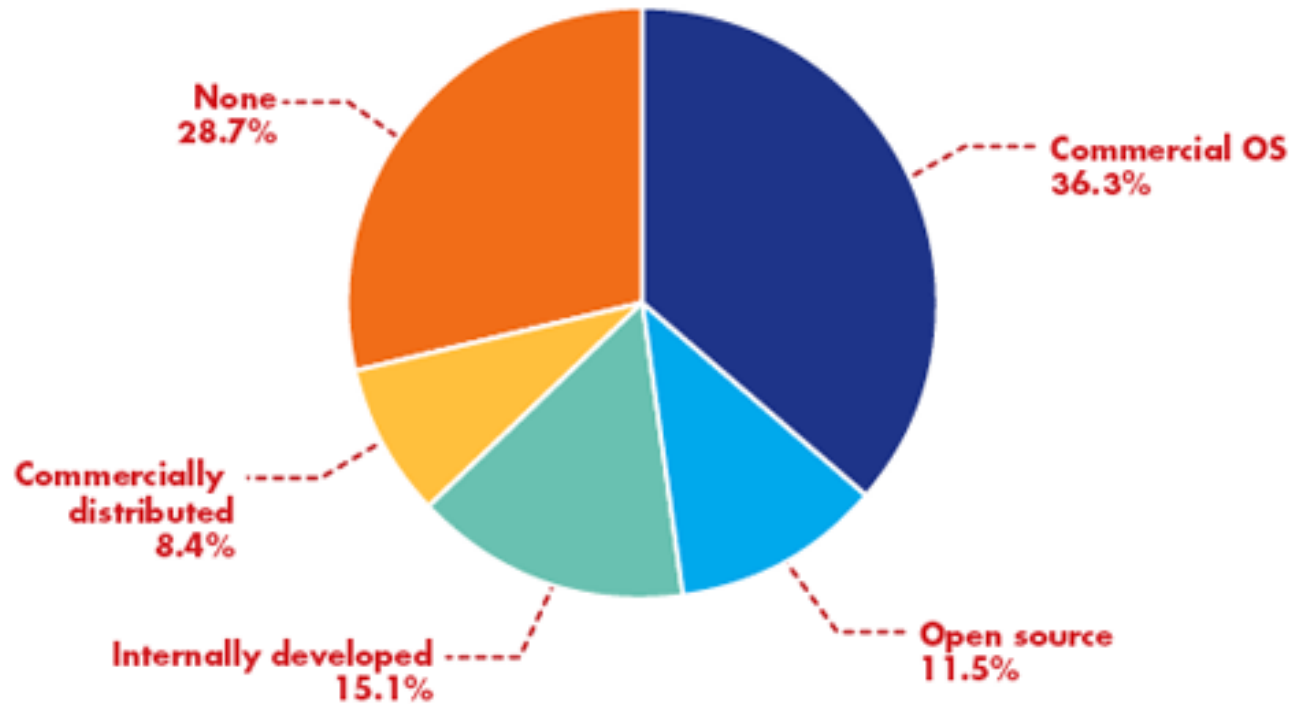


Figure 1

# 3. VAIS-JE UTILISER UN OS?

Quel type d'OS?

- **GPOS** (Normal General Purpose Operating System)  
=> **Interruptible**
- **RTOS** (Real Time Operating System)  
=> **Prédictible**

Critères	Temps partagé	Temps réel
But	Maximiser la capacité de traitement (débit) & utilisation des ressources	Etre prévisible (garantir les temps de réponse)
Temps de réponse	Bon en moyenne	Bon dans le pire des cas / moyenne non importante
Comportement à la charge	Confortable à l'utilisateur	Stabilité et respect des contraintes de temps

D'après J.Boukhobza , Systèmes d'exploitation embarqués

# 3. VAIS-JE UTILISER UN OS?

## Operating systems evaluation criteria

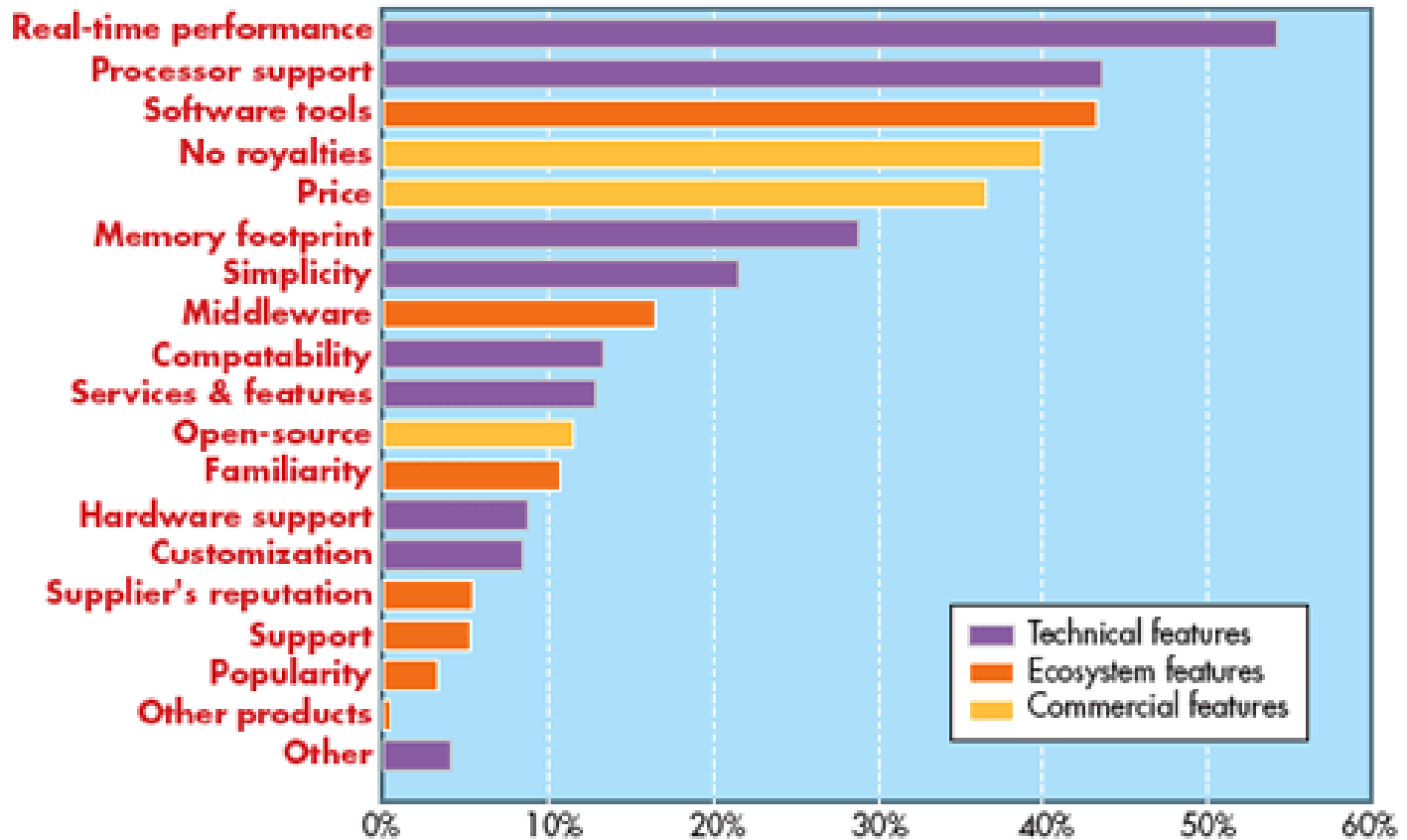


Figure 5

# 3. VAIS-JE UTILISER UN OS?

RTOS popularity now

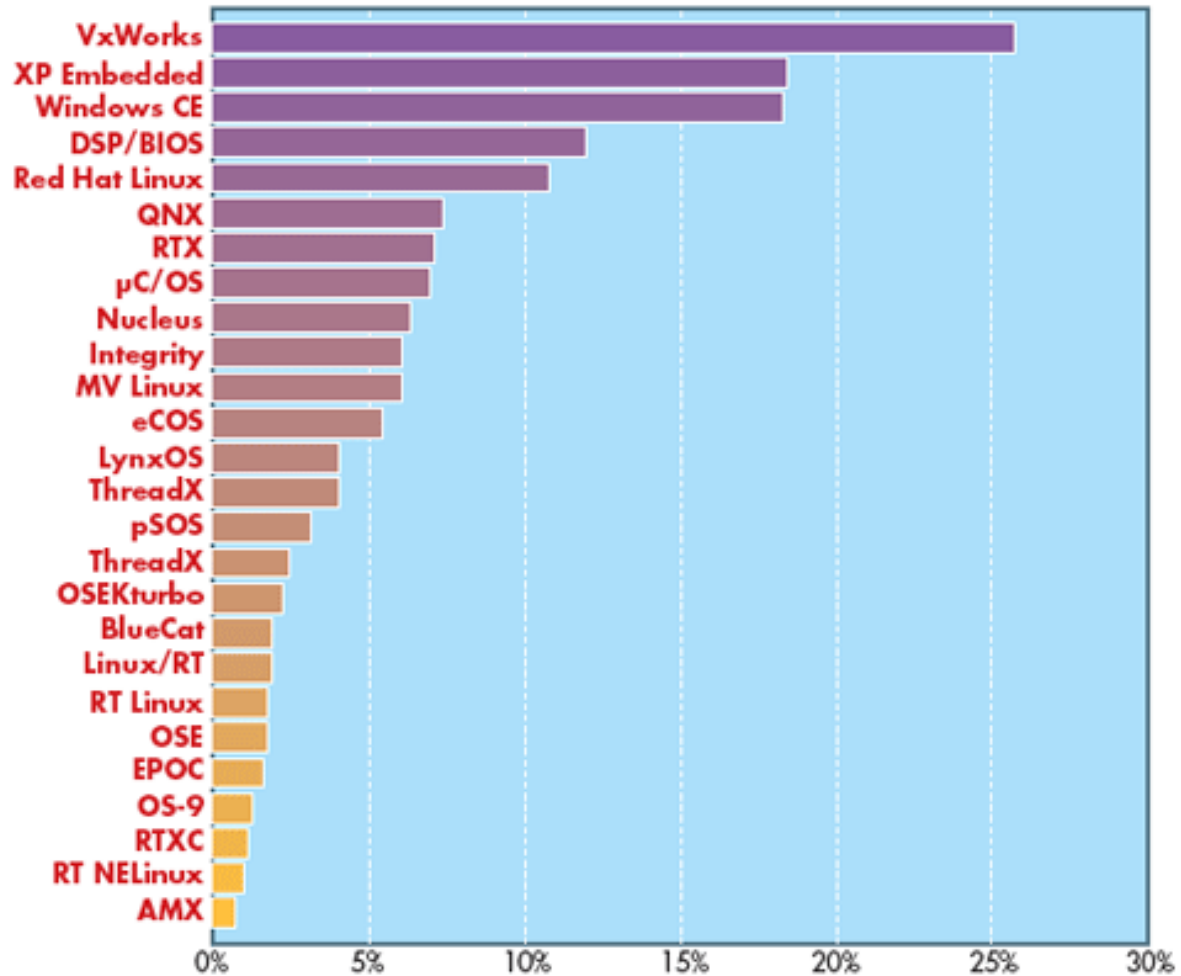


Figure 6



# 3. VAIS-JE UTILISER UN OS?

## Créer, initialiser et activer une tâche avec VxWorks

```
int taskSpawn(  
  {Task Name},  
  {Task Priority 0-255, related to scheduling},  
  {Task Options - VX_FP_TASK, execute with floating point  
  coprocessor  
  VX_PRIVATE_ENV, execute task with private environment  
  VX_UNBREAKABLE, disable breakpoints for task  
  VX_NO_STACK_FILL, do not fill task stack with 0xEE}  
  {Stack Size}  
  {Task address of entry point of program in memory-initial PC  
  value}  
  {Up to 10 arguments for task program entry routine})
```

=> Après appel du taskSpawn, une image de la tâche est créée  
(Process Control Block , pile, programme)

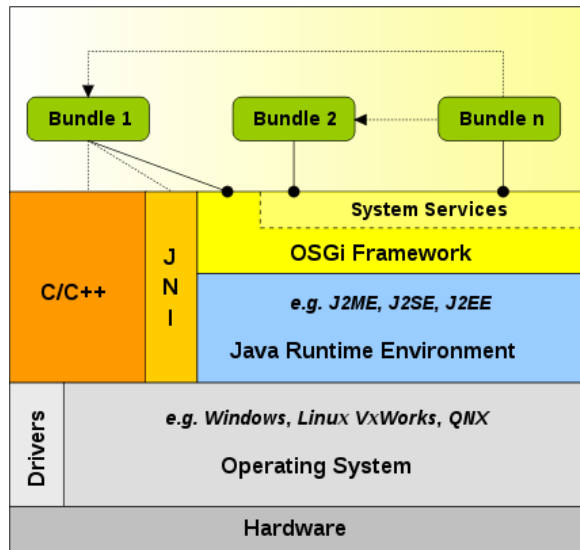
# 3. VAIS-JE UTILISER UN OS?

---

```
// Tâche du parent qui active le l'horloge logicielle
void parentTask(void) {
...
if sampleSoftware Clock NOT running {
newSWClkId = taskSpawn ("sampleSoftwareClock", 255,
VX_NO_STACK_FILL, 3000, (FUNCPTR) minuteClock, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0);
...
}
// Tâche executée par le programme fils
void minuteClock (void) {
integer seconds;
while (softwareClock is RUNNING) {
seconds = 0;
while (seconds < 60)
seconds = seconds+1;
}
.....
}
```

# 3. VAIS-JE UTILISER UN FRAMEWORK ?

- Un **framework** au-dessus de l'OS?



Bundle = applications et/ou composants déployés

Services fournis:

- **journalisation**,
- gestion des **configurations**,
- le service **HTTP** (exec. servlets),
- l'analyse syntaxique **XML**, l'accès aux dispositifs (Device Access),
- l'administration de **paquetage** (Package Admin),
- l'administration des **permissions** (Permission Admin),
- le niveau de **démarrage** (Start Level),
- la gestion des **utilisateurs** (User Admin),
- le connecteur d'ES (IO Connector; IO = Input Output = Entrées Sorties),
- la gestion des **connexions** (Wire Admin),
- Jini, l'exportateur UPnP (UPnP Exporter),
- le **pistage applicatif** (Application Tracking),
- les paquets signés (Signed Bundles),
- les services **déclaratifs** (Declarative Services),
- la **gestion de l'énergie** (Power Management),
- la gestion des **dispositifs** (Device Management),
- les politiques de **sécurité** (Security Policies),
- **diagnostic/contrôle** et organisation en couches du cadriciel (Diagnostic/Monitoring and Framework Layering).

# 4. JE DÉVELOPPE MON APPLICATION

- Choisir la chaîne de développement

The screenshot displays the MPLAB X IDE v1.10 interface. The main window shows the source code for a C program named 'Lab1.c'. The code defines a 'delay' function and includes an interrupt service routine for a switch press. The disassembly view shows the corresponding assembly instructions. The debugger console indicates that the target is halted.

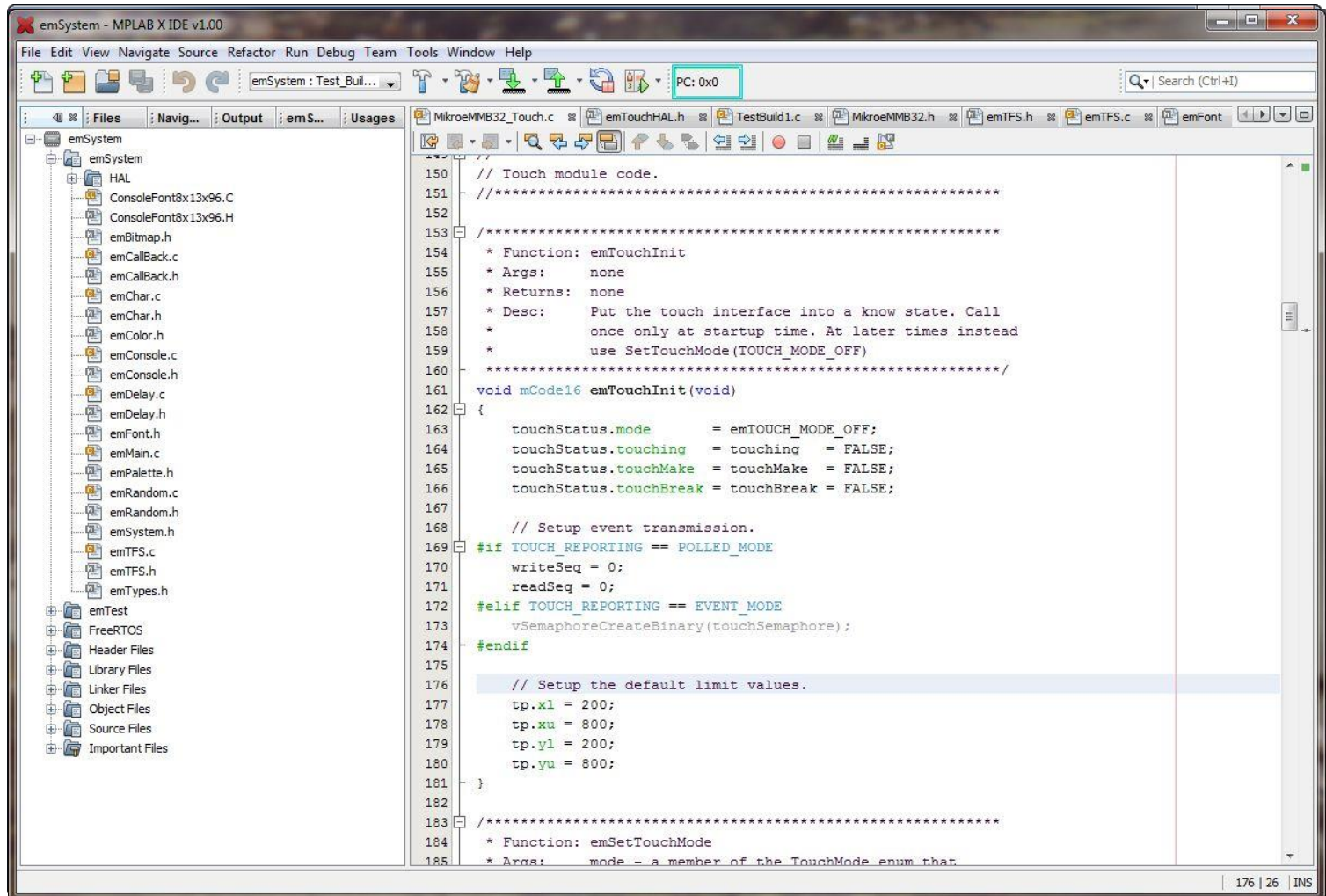
```
1 !void delay(void)
2 !{
3 0x410: LNK #0x2
4 ! unsigned timedelay = count;
5 0x412: MOV 0x856, W0
6 0x414: MOV W0, [W14]
7 ! while (timedelay)
8 0x416: BRA 0x49A
9 0x49A: CPO [W14]
10 0x49C: BRA NZ, 0x418
11 ! {
12 !     timedelay--;
13 0x418: DEC [W14], [W14]
14 !     if (SwitchPressed(&PORTD, 13))
15 0x41A: MOV #0xD, W1
16 0x41C: MOV #0x2D4, W0
17 0x41E: RCALL SwitchPressed
18 0x420: CPO W0
19 0x422: BRA Z, 0x458
20 ! {
21 !     LATLED ^= 0x01;
22 0x424: MOV LATA, W0
23 0x426: BTG W0, #0
24 0x428: MOV W0, LATA
25 !     count++;
26 0x42A: MOV 0x856, W0
```

The debugger console shows the following output:

```
Running
Target Halted
```

# 4. JE DÉVELOPPE MON APPLICATION

- Choisir la chaine de développement





# 4. JE DÉVELOPPE MON APPLICATION

- Etudier les *datasheets* du micro-contrôleur

## 7.3 Reading the Data EEPROM Memory

To read a data memory location, the user must write the address to the EEADRH:EEADR register pair, clear the EEPGD control bit (EECON1<7>), clear the CFGS

control bit (EECON1<6>) and then set the RD control bit (EECON1<0>). The data is available for the very next instruction cycle; therefore, the EEDATA register can be read by the next instruction. EEDATA will hold this value until another read operation or until it is written to by the user (during a write operation).

### EXAMPLE 7-1: DATA EEPROM READ

```
MOVLW DATA_EE_ADDRH ;  
MOVWF EEADRH ; Upper bits of Data Memory Address to read  
MOVLW DATA_EE_ADDR ;  
MOVWF EEADR ; Lower bits of Data Memory Address to read  
BCF EECON1, EEPGD ; Point to DATA memory  
BCF EECON1, CFGS ; Access EEPROM  
BSF EECON1, RD ; EEPROM Read  
MOVF EEDATA, W ; W = EEDATA
```

## 7.4 Writing to the Data EEPROM Memory

To write an EEPROM data location, the address must first be written to the EEADRH:EEADR register pair and the data written to the EEDATA register. Then the sequence in Example 7-2 must be followed to initiate the write cycle.

The write will not initiate if the above sequence is not exactly followed (write 55h to EECON2, write AAh to EECON2, then set WR bit) for each byte. It is strongly recommended that interrupts be disabled during this code segment.

Additionally, the WREN bit in EECON1 must be set to enable writes. This mechanism prevents accidental writes to data EEPROM due to unexpected code

execution (i.e., runaway programs). The WREN bit should be kept clear at all times except when updating the EEPROM. The WREN bit is not cleared by hardware.

After a write sequence has been initiated, EECON1, EEADRH, EEADR and EEDATA cannot be modified. The WR bit will be inhibited from being set unless the WREN bit is set. Both WR and WREN cannot be set with the same instruction.

At the completion of the write cycle, the WR bit is cleared in hardware and the EEPROM Write Complete Interrupt Flag bit (EEIF) is set. The user may either enable this interrupt or poll this bit. EEIF must be cleared by software.

### EXAMPLE 7-2: DATA EEPROM WRITE

```
MOVLW DATA_EE_ADDRH ;  
MOVWF EEADRH ; Upper bits of Data Memory Address to write  
MOVLW DATA_EE_ADDR ;  
MOVWF EEADR ; Lower bits of Data Memory Address to write  
MOVLW DATA_EE_DATA ;  
MOVWF EEDATA ; Data Memory Value to write  
BCF EECON1, EEPGD ; Point to DATA memory  
BCF EECON1, CFGS ; Access EEPROM  
BSF EECON1, WREN ; Enable writes  
  
BCF INTCON, GIE ; Disable Interrupts  
MOVLW 0x55 ;  
MOVWF EECON2 ; Write 55h  
MOVLW 0xAA ;  
MOVWF EECON2 ; Write AAh  
BSF EECON1, WR ; Set WR bit to begin write  
BSF INTCON, GIE ; Enable Interrupts  
  
; User code execution  
BCF EECON1, WREN ; Disable writes on write complete (EEIF set)
```

# 5. JE VALIDE MON SYSTÈME

---Validation---

Risques acceptables / Satisfaction des utilisateurs

Sûreté de fonctionnement	Satisfaction des utilisateurs
Fiabilité	QoS
Disponibilité	Performances
Maintenabilité	
Sécurité confidentialité	
Sécurité innocuité	
...	

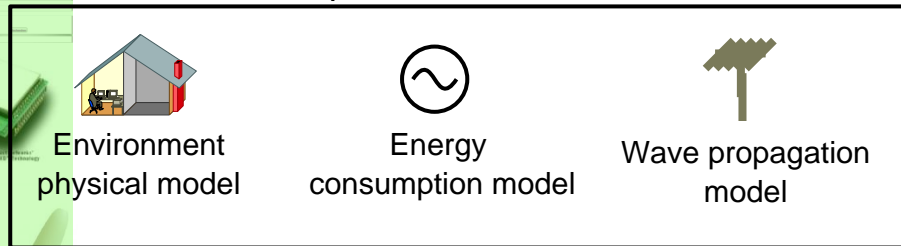


# 5. JE VALIDE MON SYSTÈME

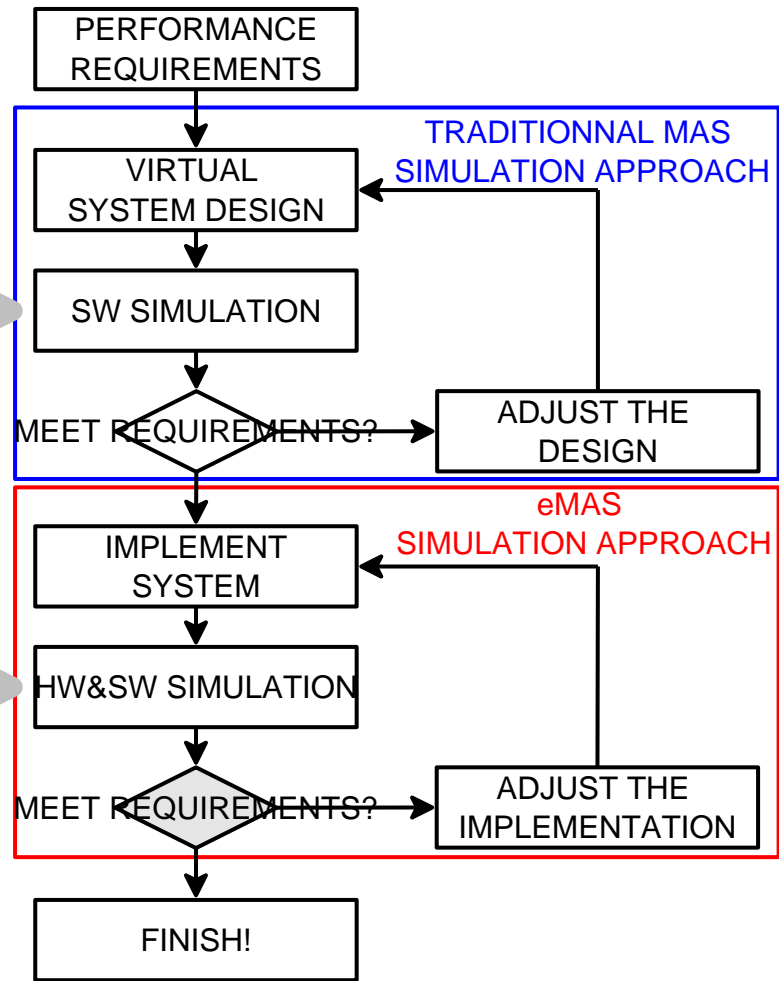
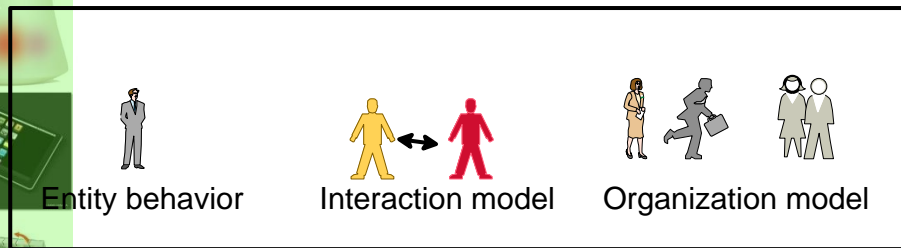
---Des outils spécifiques---

## MODELS INVOLVED IN THE SYSTEM SIMULATION

### Environnement description models

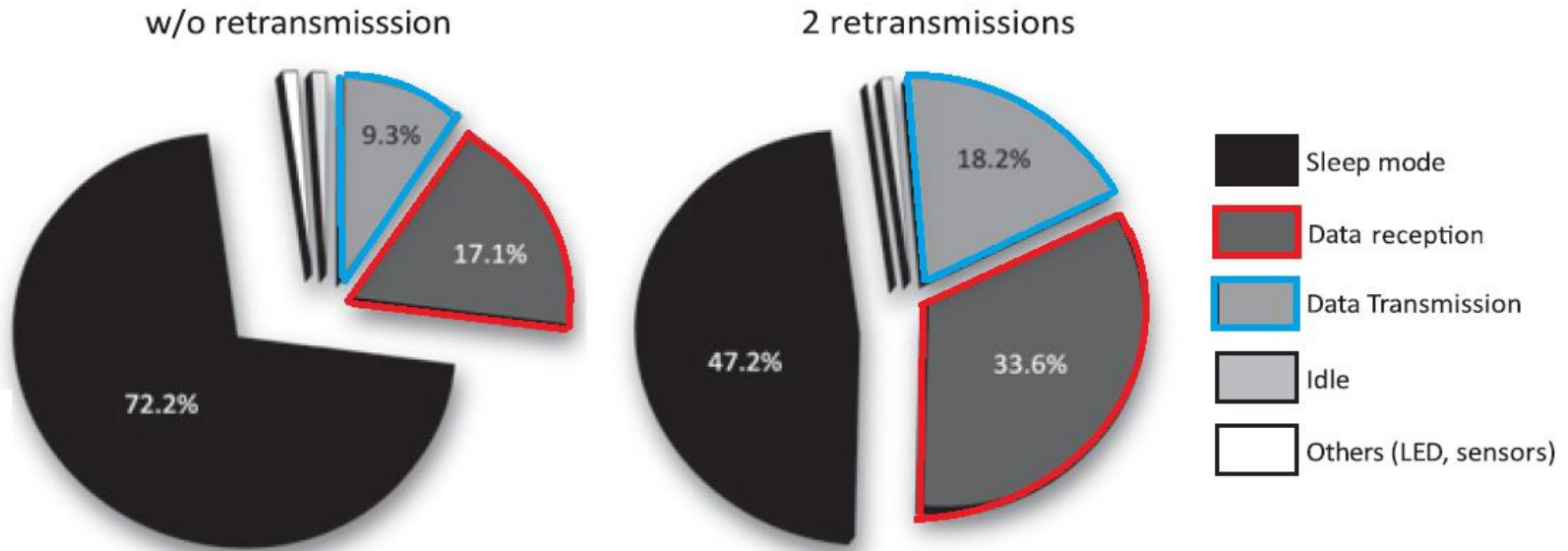


### Entities involved models



# 5. JE VALIDE MON SYSTÈME

## ---Performances énergétiques---



**Fig. 6.** Energy consumption repartition for different scenari.

[Fourty12]



# BIBLIOGRAPHIE

---

- Culture mobile – Le Blog <http://blog.culturemobile.net>
- Jalil Boukhobza, Systèmes d'exploitation pour l'embarqué (cours)
- Julien DeAntoneti, Introduction à la programmation micro-contrôleur (cours)
- Nicolas Fourty, Adrien van den Bossche, Thierry Val: An advanced study of energy consumption in an IEEE 802.15.4 based network: Everything but the truth on 802.15.4 node lifetime. Computer Communications 35(14): 1759-1767 (2012)
- Jean-Paul Jamont, Michel Ocelllo, Eduardo Mendes, Decentralized intelligent real world embedded systems : a tool to tune design and deployment, Advances in Intelligent and Soft-Computing, Springer, ISSN: 1867-5662, 2013.
- Jean-Paul Jamont, Les réseaux sans fils (cours)
- Emmanuel Jez, Développement d'une colonie de robots mobiles « Kurasu ».
- Stéphane Lavirotte , Objets Communicants et Terminaux Mobiles (cours)
- Etienne Perret, RFID sans puce, de l'identification au capteur .
- Olivier Rance, Construction de la signature électromagnétique d'un tag RFID sans puce à partir d'un assemblage de résonateurs.
- Jim Turley , Source: <http://www.eetimes.com/discussion/other/4025674/Operating-systems-on-the-rise>

