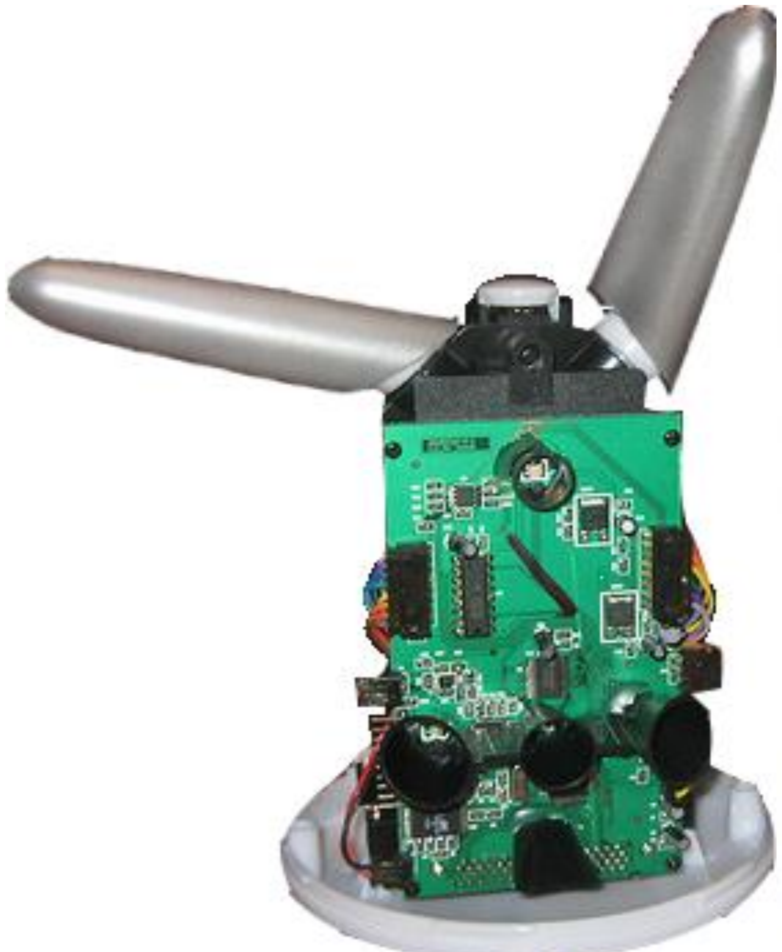
Les Objets Communicants

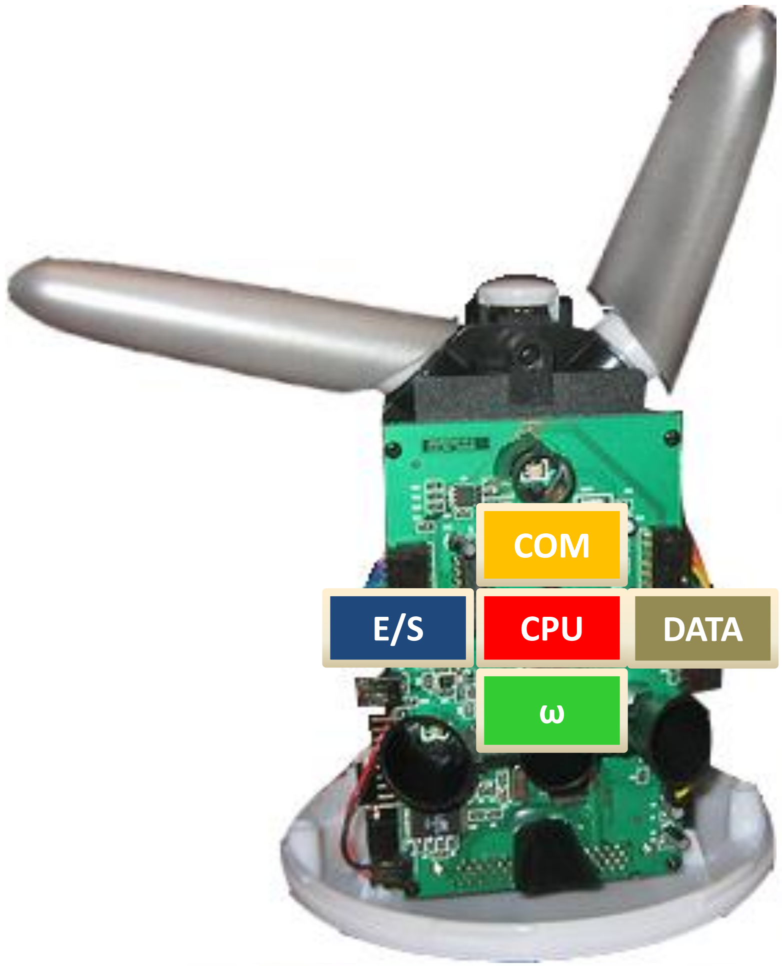
Jean-Paul Jamont, Lionel Médini, Michaël Mrissa

Comment fonctionnent les objets communicants?

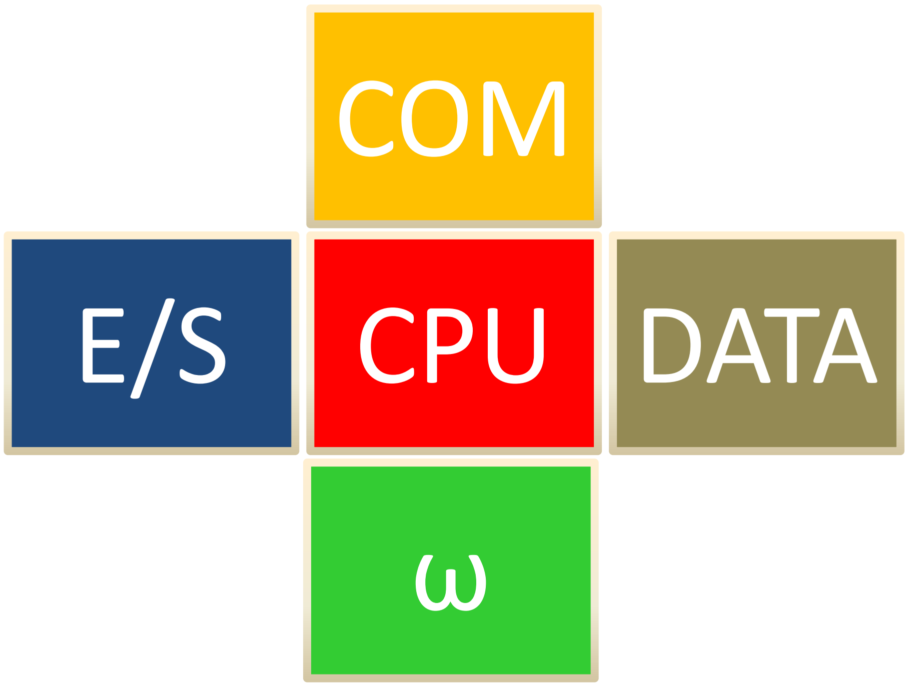
Comment fonctionnent les objets communicants?



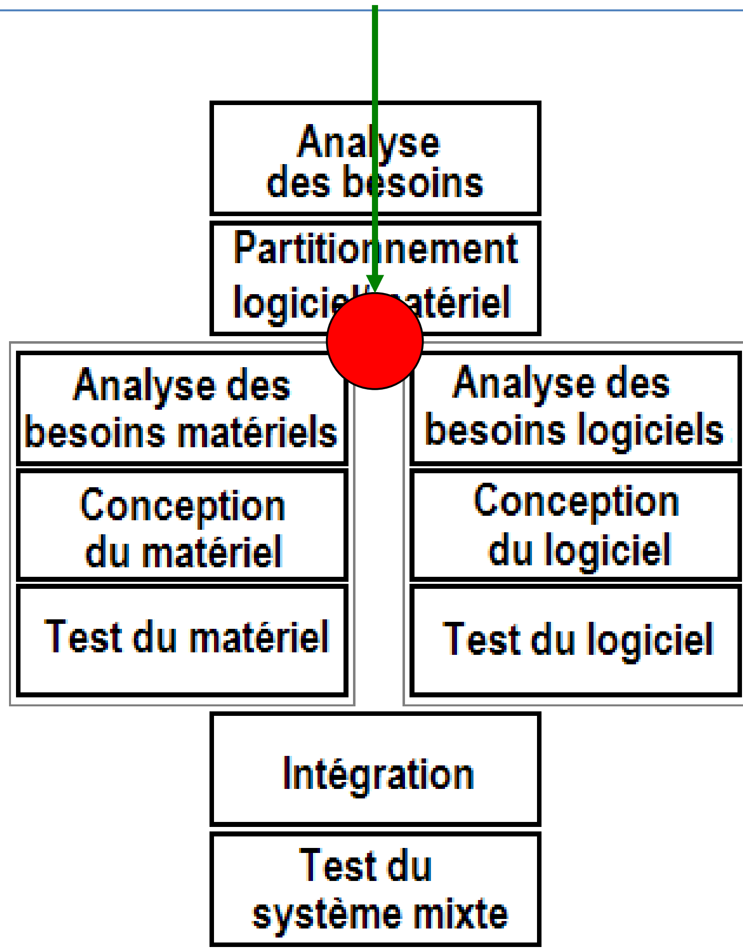
COMMENT FONCTIONNENT LES OBJETS COMMUNICANTS?



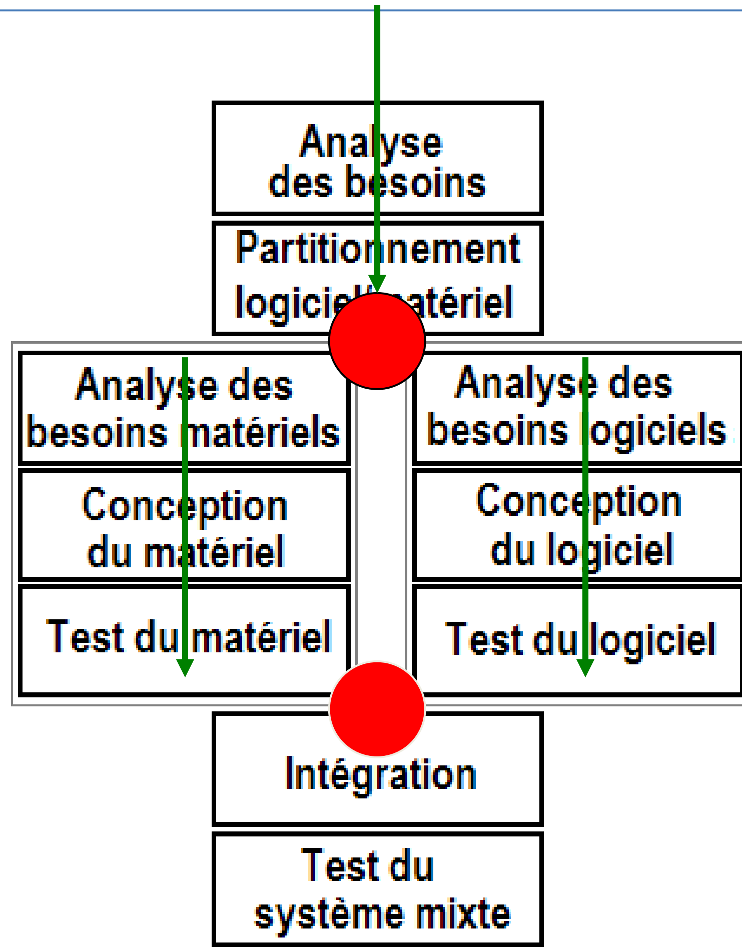
Comment créer ses propres objets communicants?



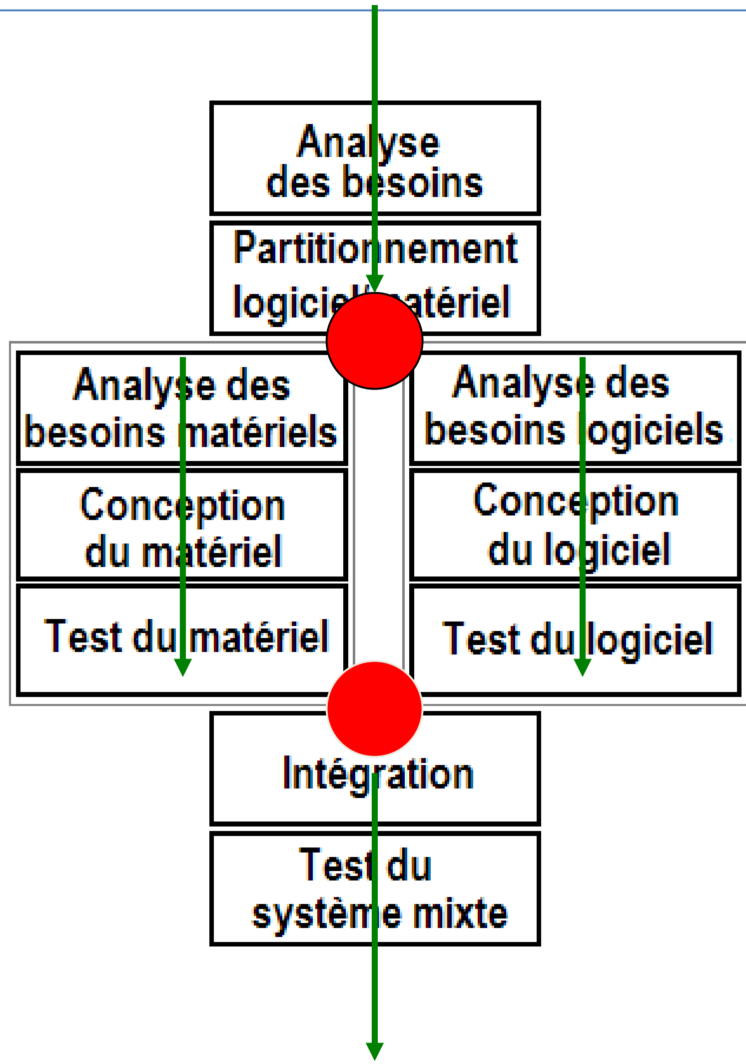
1. Quelle démarche?



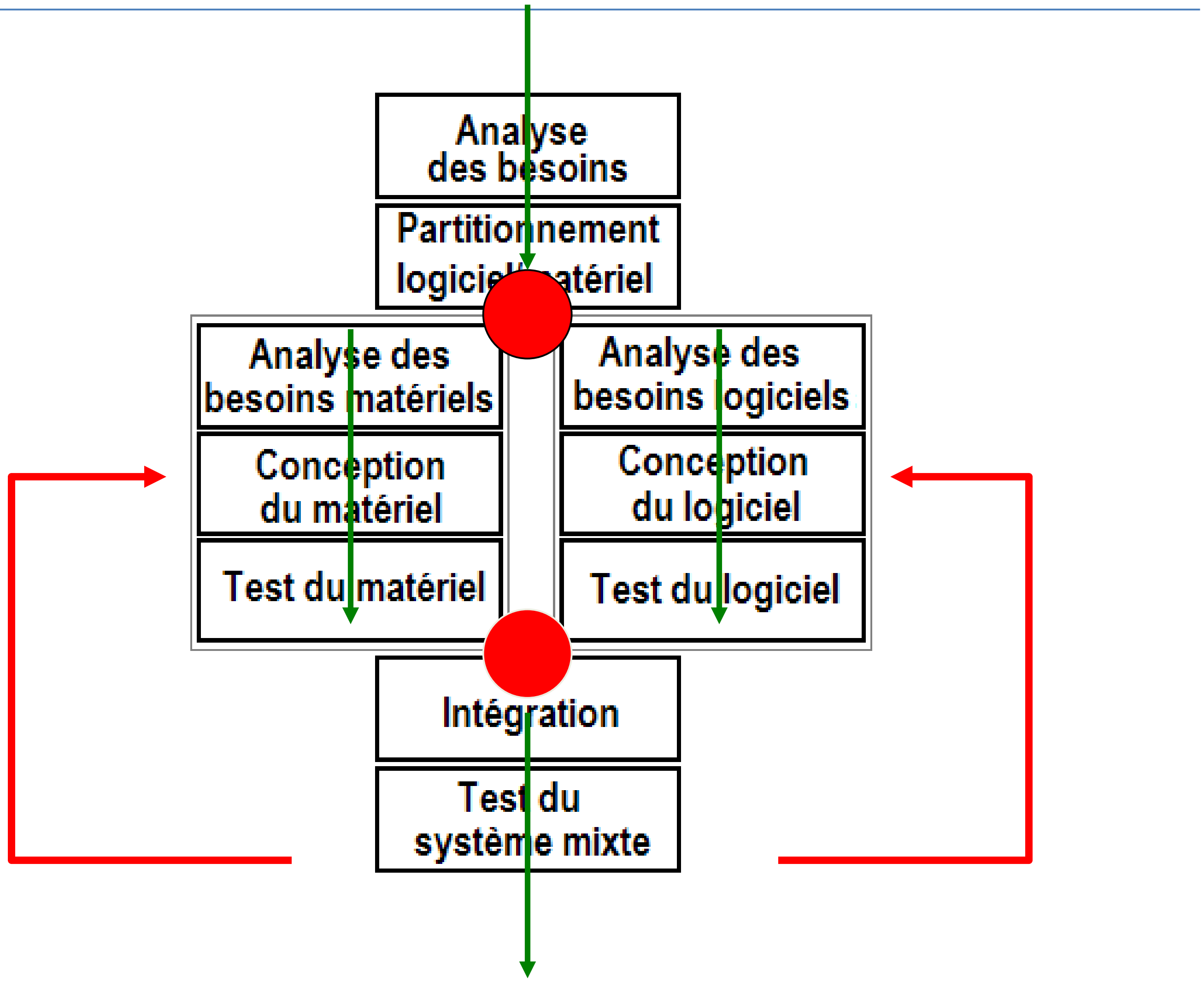
1. Quelle démarche?



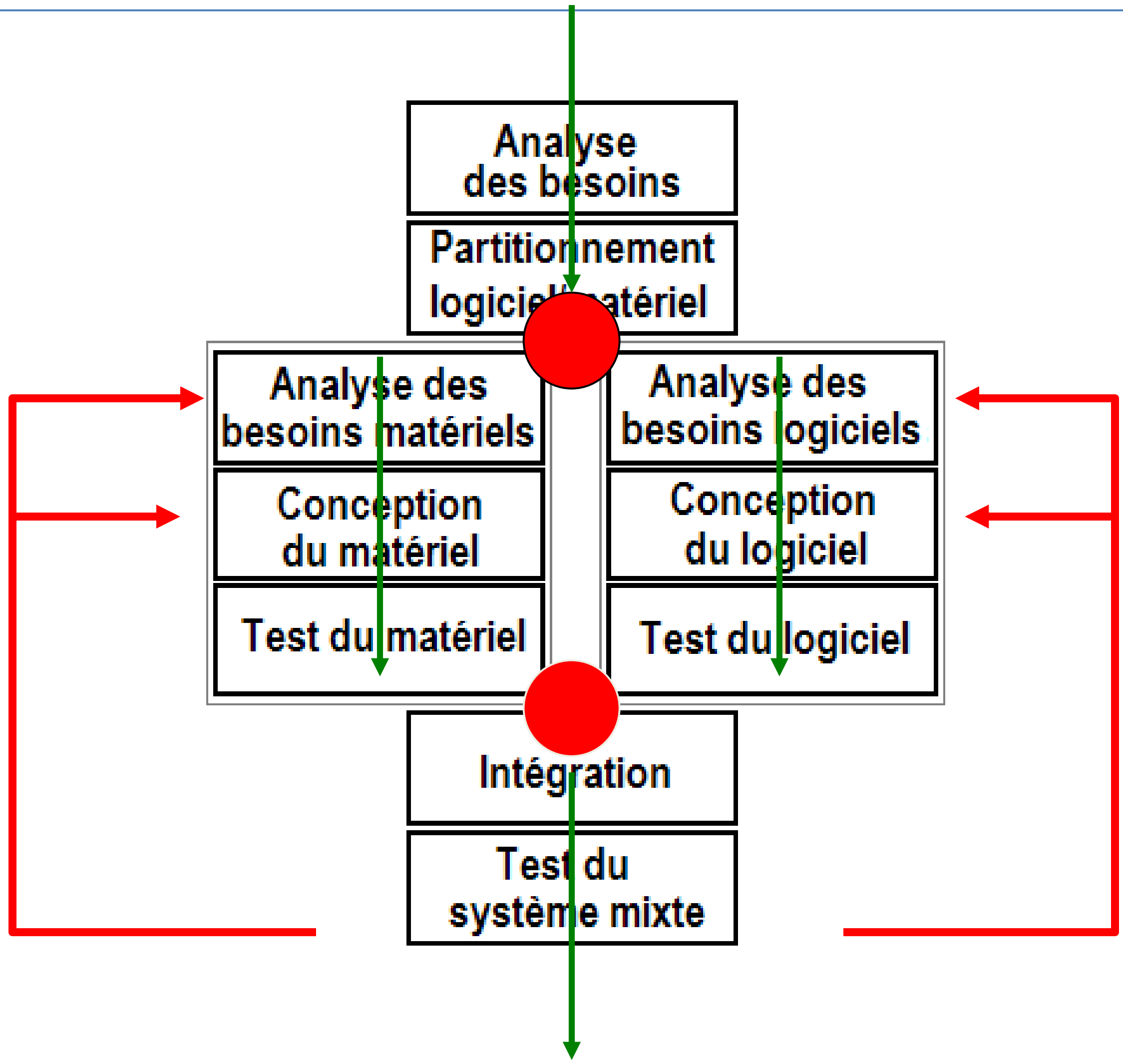
1. QUELLE DÉMARCHE?



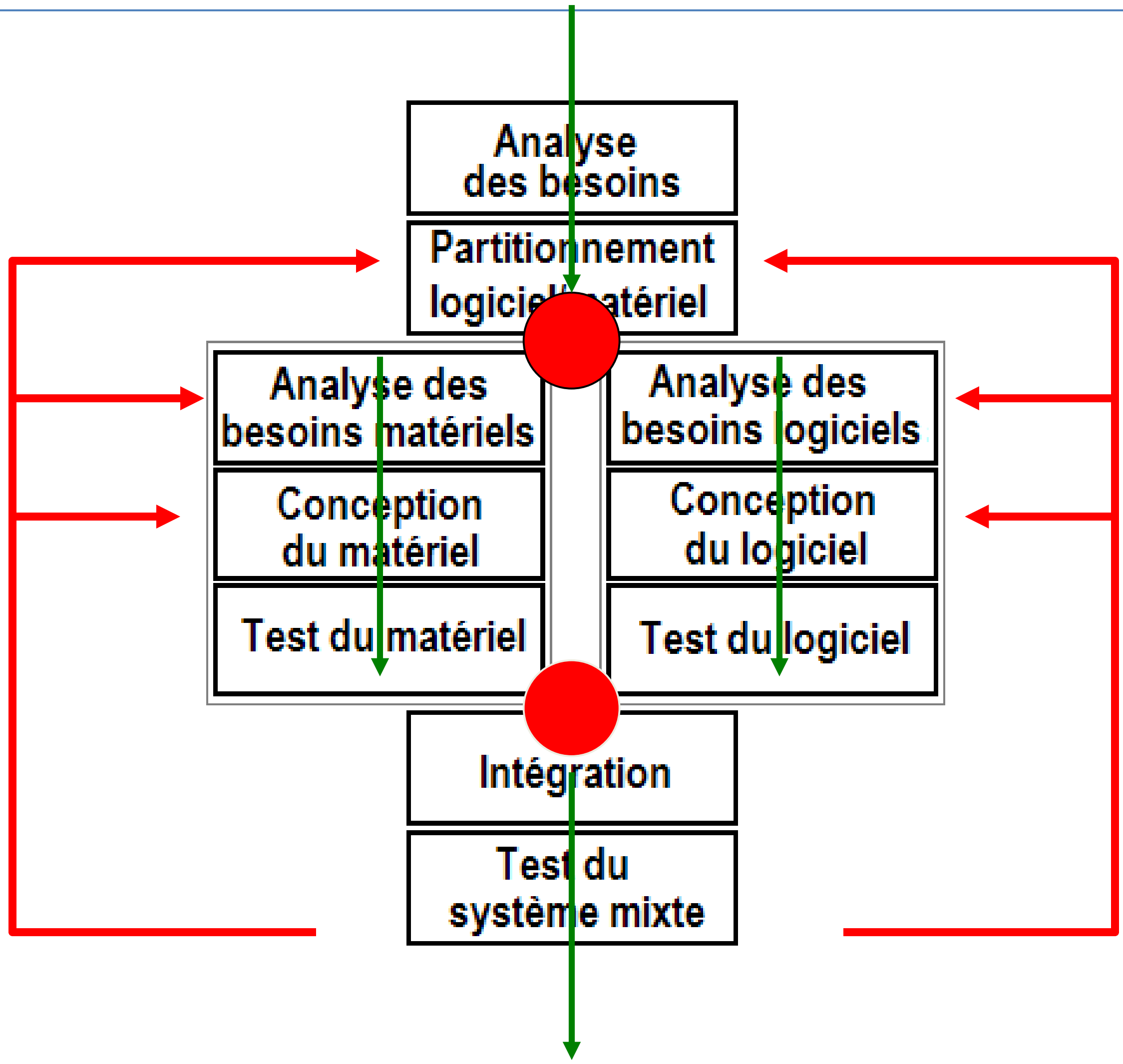
1. Quelle démarche?



1. Quelle démarche?

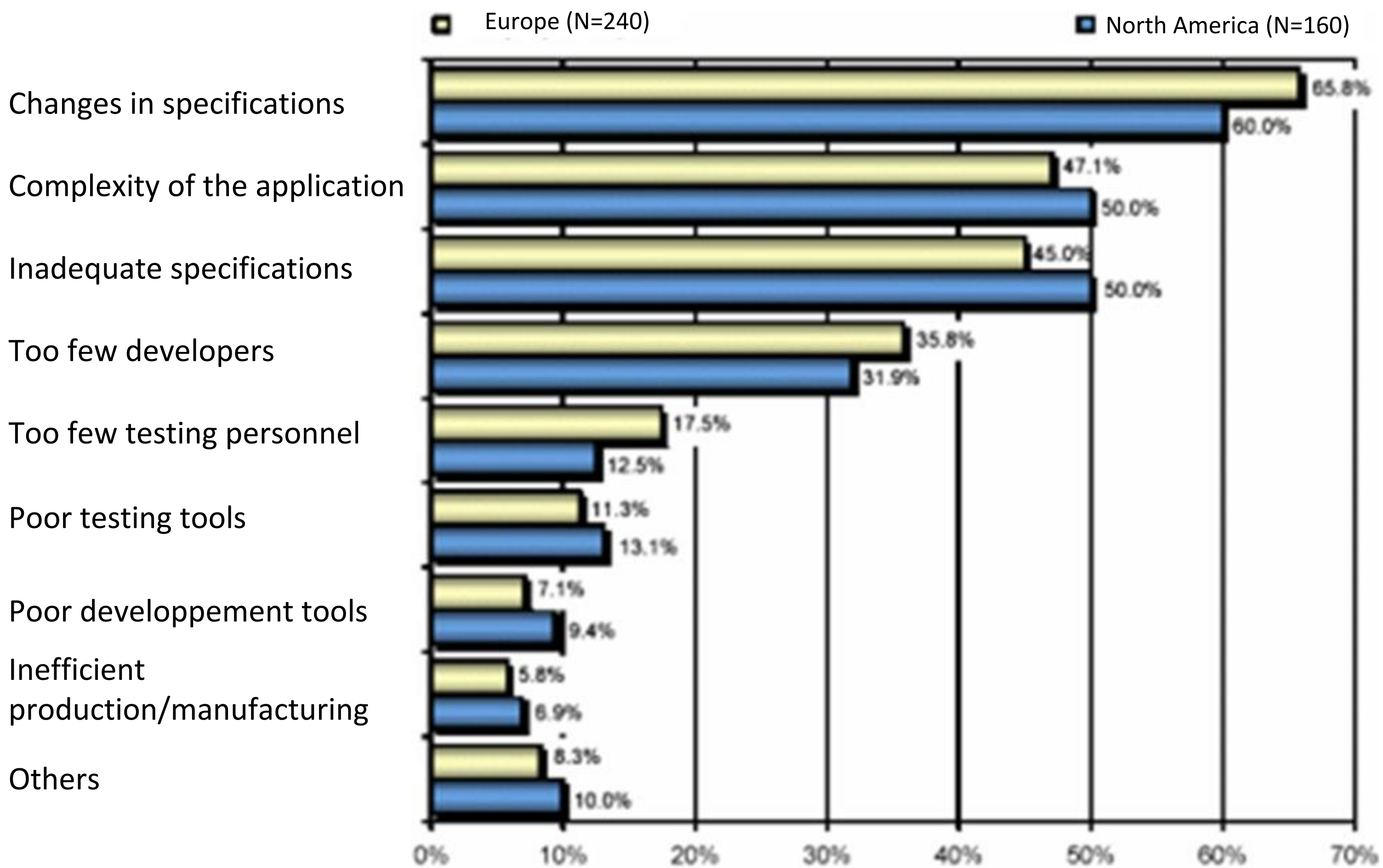


1. QUELLE DÉMARCHE?

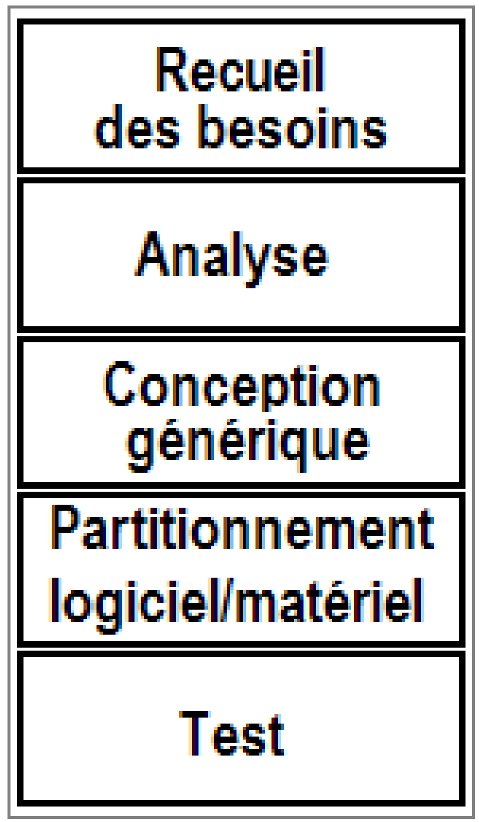


1. Quelle démarche?

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

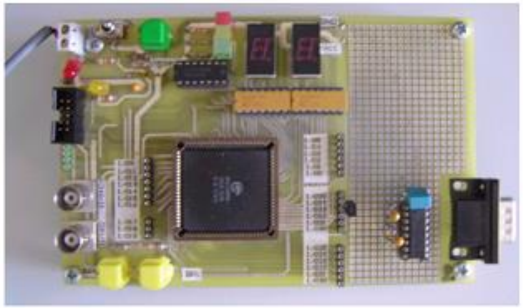


1. Quelle démarche ?



1. Quelle Démarche?

SOLUTION MATERIELLE
CARTE A BASE D'EPLD/FPGA



SOLUTION LOGICIELLE
CARTE A MICROPROSSEURMICROCONTROLLEUR

SYNTHESE MATERIELLE

```
-- COMPONANT Gestion role
-- Version 0.95a
-- WARNING: Default parameter use - Synchronous version -
-- State translation :
--    S1 by vector "000"
--    S2 by vector "001"
--    S3 by vector "010"
--    S4 by vector "011"
--    S5 by vector "100"

library iee;
use ieee.std_logic_1164.all;
use work.std_arith.all;

-- Componant description
entity GestionRole is port(
    reset : in str_logic;
    NVR : in std_logic_vector(7 downto 0);
    role : in out std_logic_vector(1 downto 0);
    -- Call to function "election"
    electionIn  : in std_logic_bit;
    electionOut : in std_logic_bit;
end GestionRole;

-- Componant architecture
architecture arch_GestionRole is
    signal pastState:std_logic_vector(2 downto 0);
    signal state:std_logic_vector(2 downto 0);
    signal futureState:std_logic_vector(2 downto 0);
begin
    exec:process(clk)
        begin
            if(clk'event and clk='1') then
                if(reset='1') then
                    state<="000";
                else
                    case state is
                        -- State S1
                        when "000" =>
                            if (NVR="00000001" then
                                futurState = "001";
```

SYNTHESE LOGICIELLE

```
String futurState = "";


public void exec(){
 ;//FIRST STATE ENTRY
          for(;;){
          switch(presentState){
                  case(S1):
                          if(NVR == 1)
                            futurState = "S2";

                          if(NVR == 2)
                            futurState = "S3";

                          //ACTION
                          if (presentState != futureState)
                            ;//EXIT
                          else if (presentState != pastState)
                            ;//ENTRY
                          else
                             role = auncun;//STATE
                         break;
                  case(S2):
                          if(NVR==1)
                          futurState = "D";
```

2. 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

2. QUELS SONT MES BESOINS?

Quels sont mes besoins / mes contraintes?

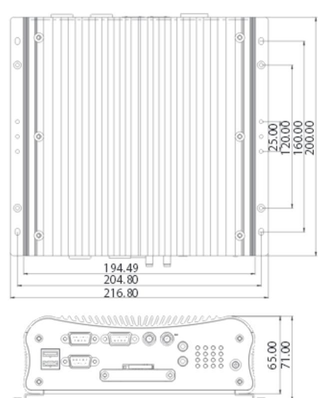
- Puissance de calcul
- Capacité de communication
- Capacité de stockage
- Consommation d’énergie
- Temps de réponse
- Tolérances aux pannes
- Durée de vie
- Testabilité/débogage
- Nombre d’unités produites

3. Quels choix pour la partie matérielle?

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

3. Quels choix pour la partie matérielle?

- PDA / Mini-PC / PC Industriels



<u>Modèle:</u> Samsung Galaxy S4GT-I9505

<u>OS:</u> Android 4.2.2 Jelly Bean

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

<u>Mémoire vive:</u> 2 Go de RAM LPDDR3

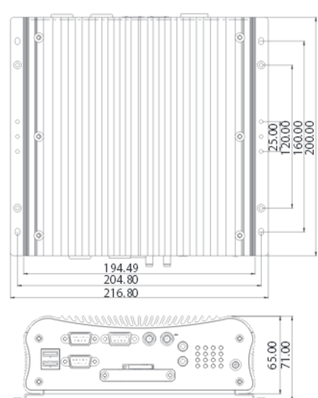
<u>Stockage:</u> 32 Go de mémoire flash intégrée

<u>Connectique</u>: 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

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

3. 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

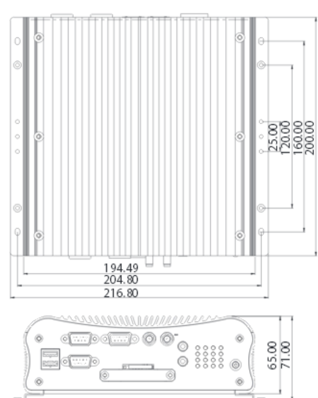
Stockage: 8 Go de mémoire flash intégrée

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:

3. 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

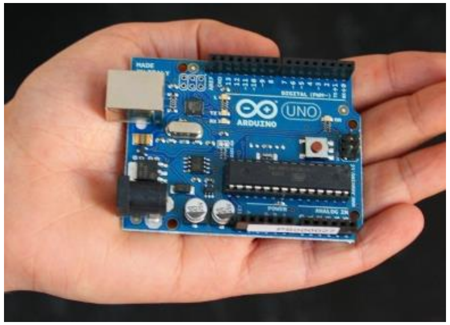
Stockage: Disque dur 2.5" SATA

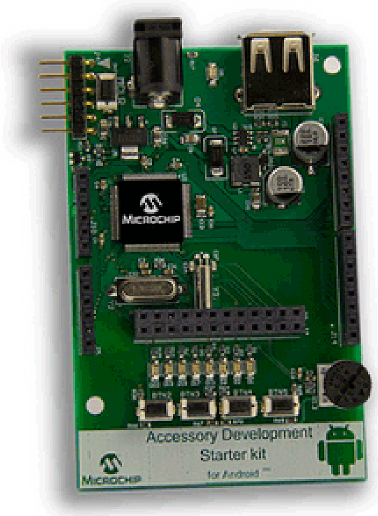
Connectique: 6 ports USB2.0; 1 emplacement CFast; 1 emplacement carte SIM, 2 ports RS-232/422/485 isolés, 2 ports Ethernet Intel® 82574IT Gb, 1 port DB15 E/S , WiFi 802.11 a/b/g/n/ac or 3.5G (auto detected modules), Support 9-36V DV input, Audio Jack

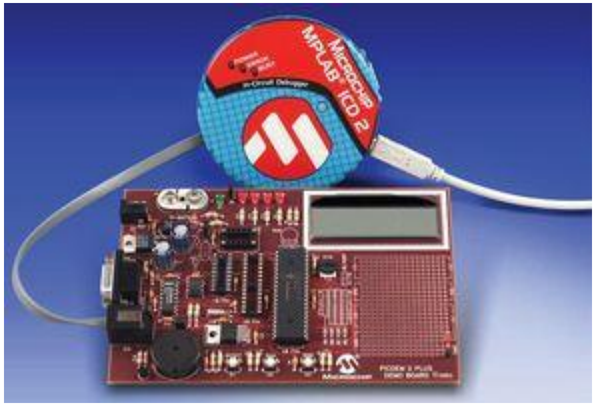
Capteurs:

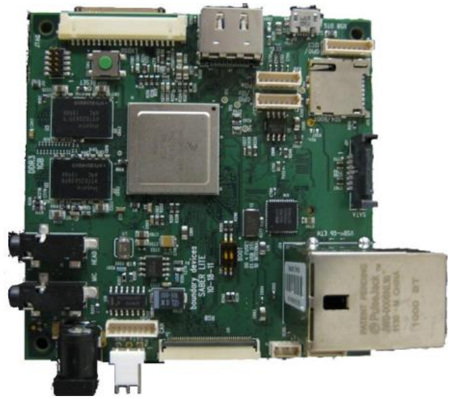
3. Quels choix pour la partie matérielle?

- Une carte de développement existante



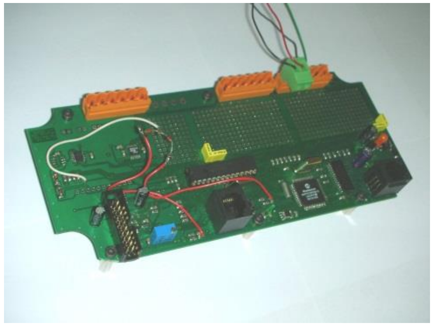


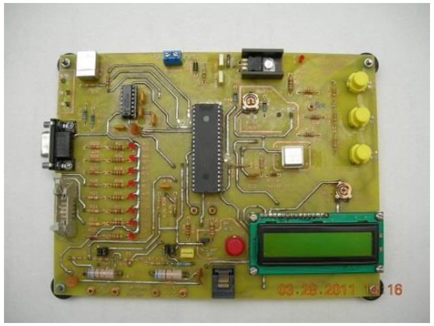




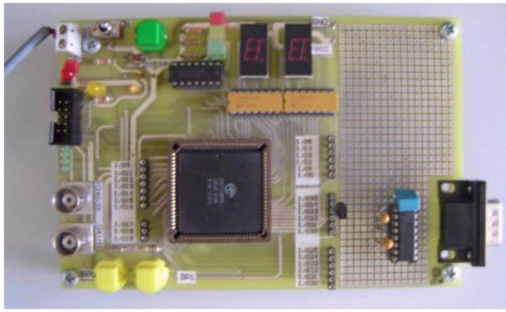
3. Quels choix pour la partie matérielle?

- From scratch
  - Architecture centrée microcontrôleur



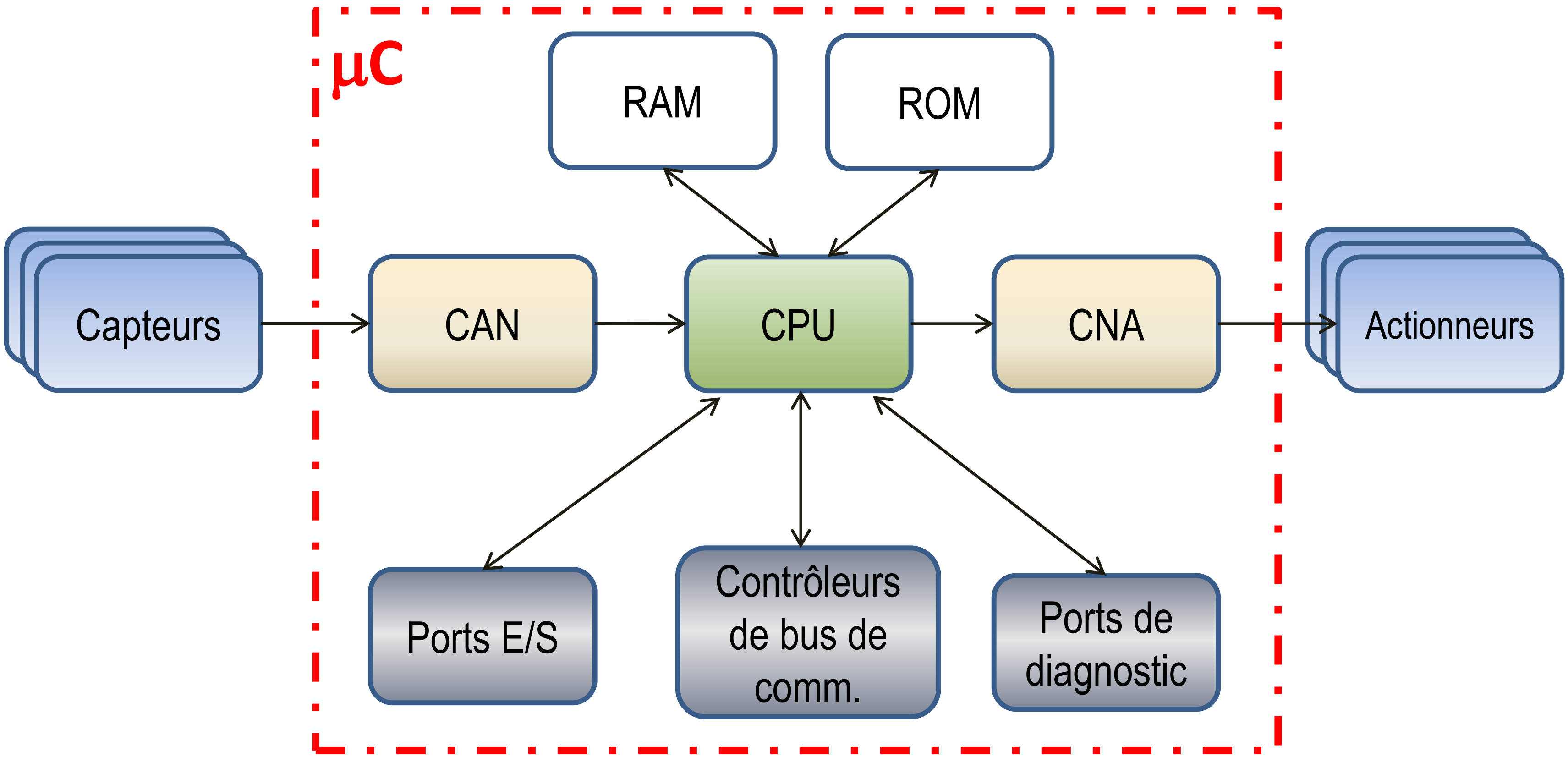


  - Architecture centrée FPGA/EPLD/ASIC?



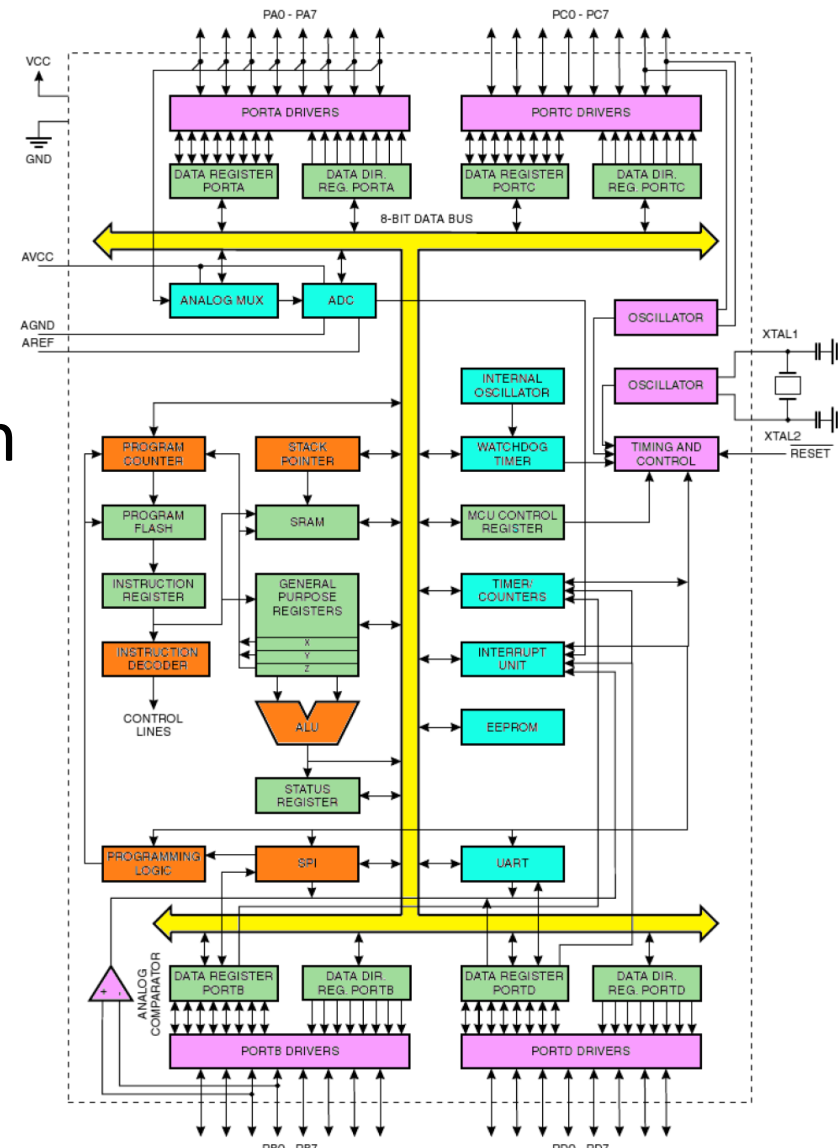
3. Quels choix pour la partie matérielle?

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



3. Quels Choix pour la Partie Matérielle?

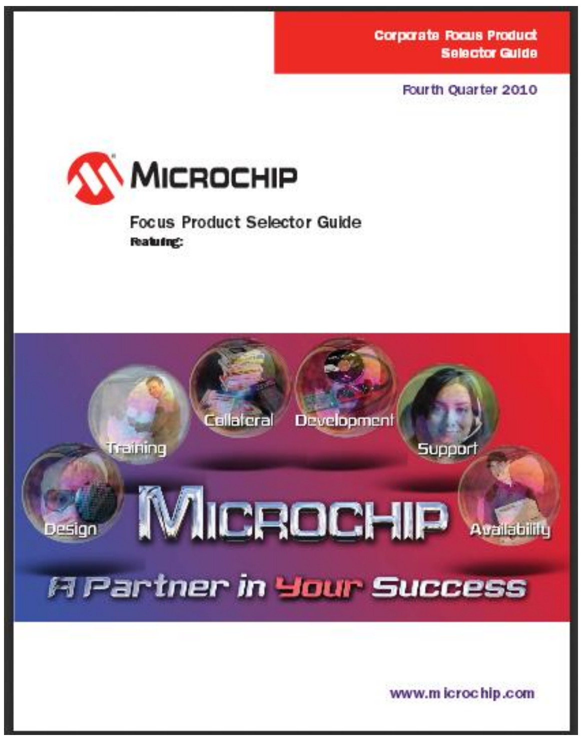
- Processeur
- Mémoire
- Périphériques
- Bus de communication
- Entrées/Sorties

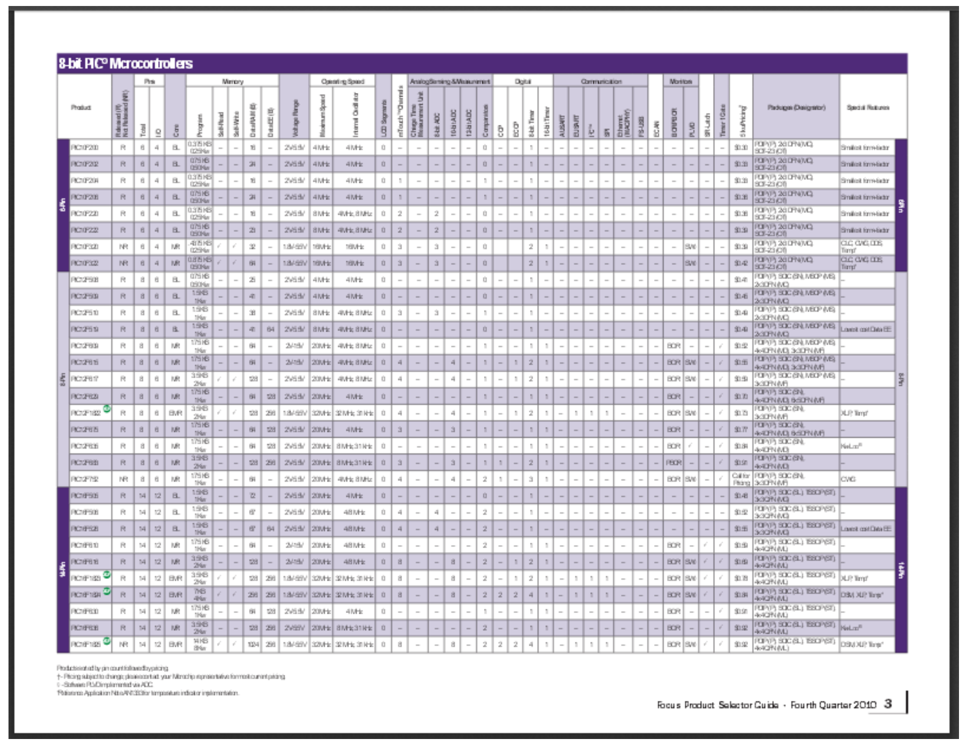


D'après Julien DeAntoni

3. Quels Choix pour la Partie Matérielle?

- Utiliser les « Product Selector Guide »



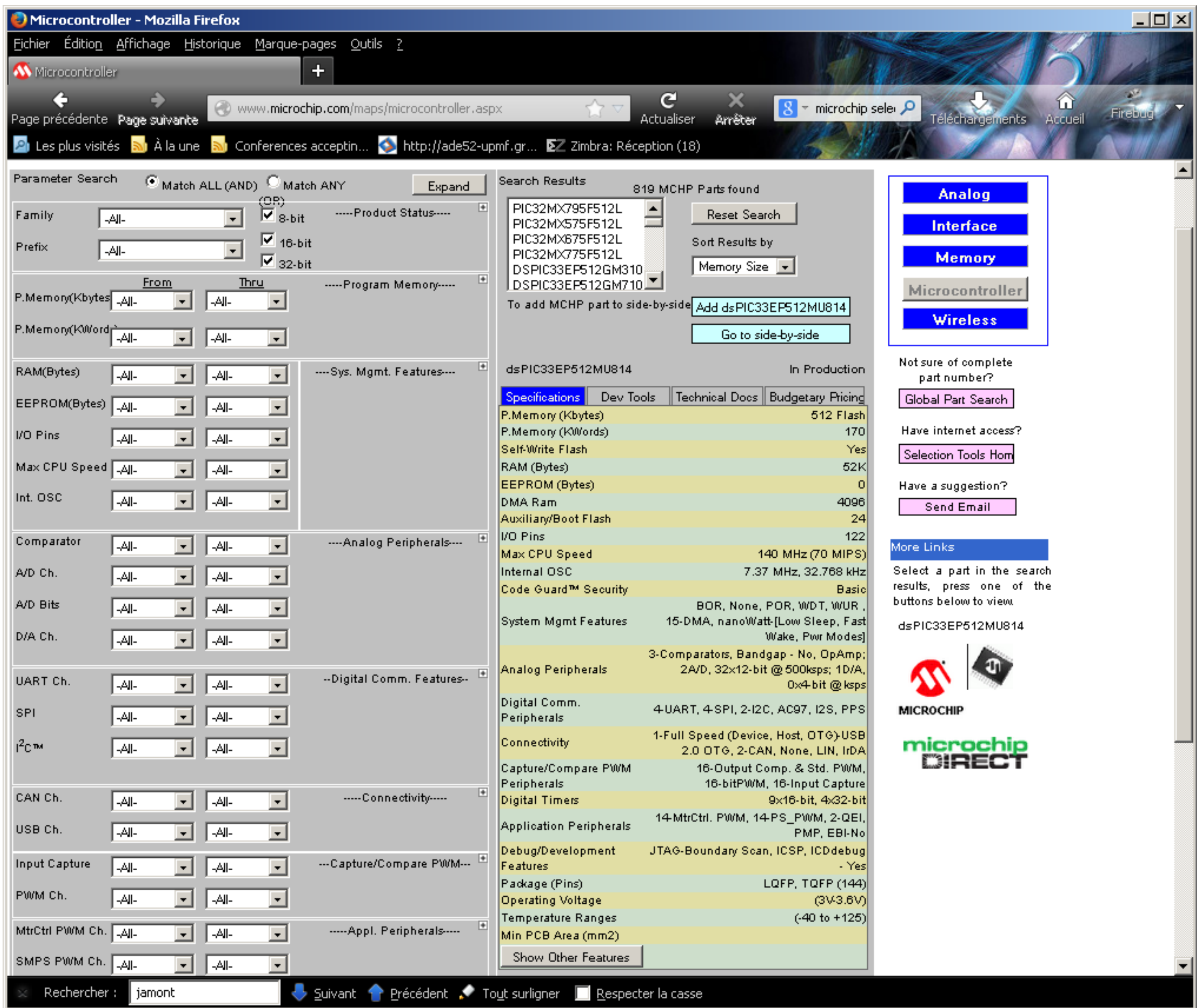

8-bit PIC® Microcontrollers

Focus Product Selector Guide • Fourth Quarter 2010 3

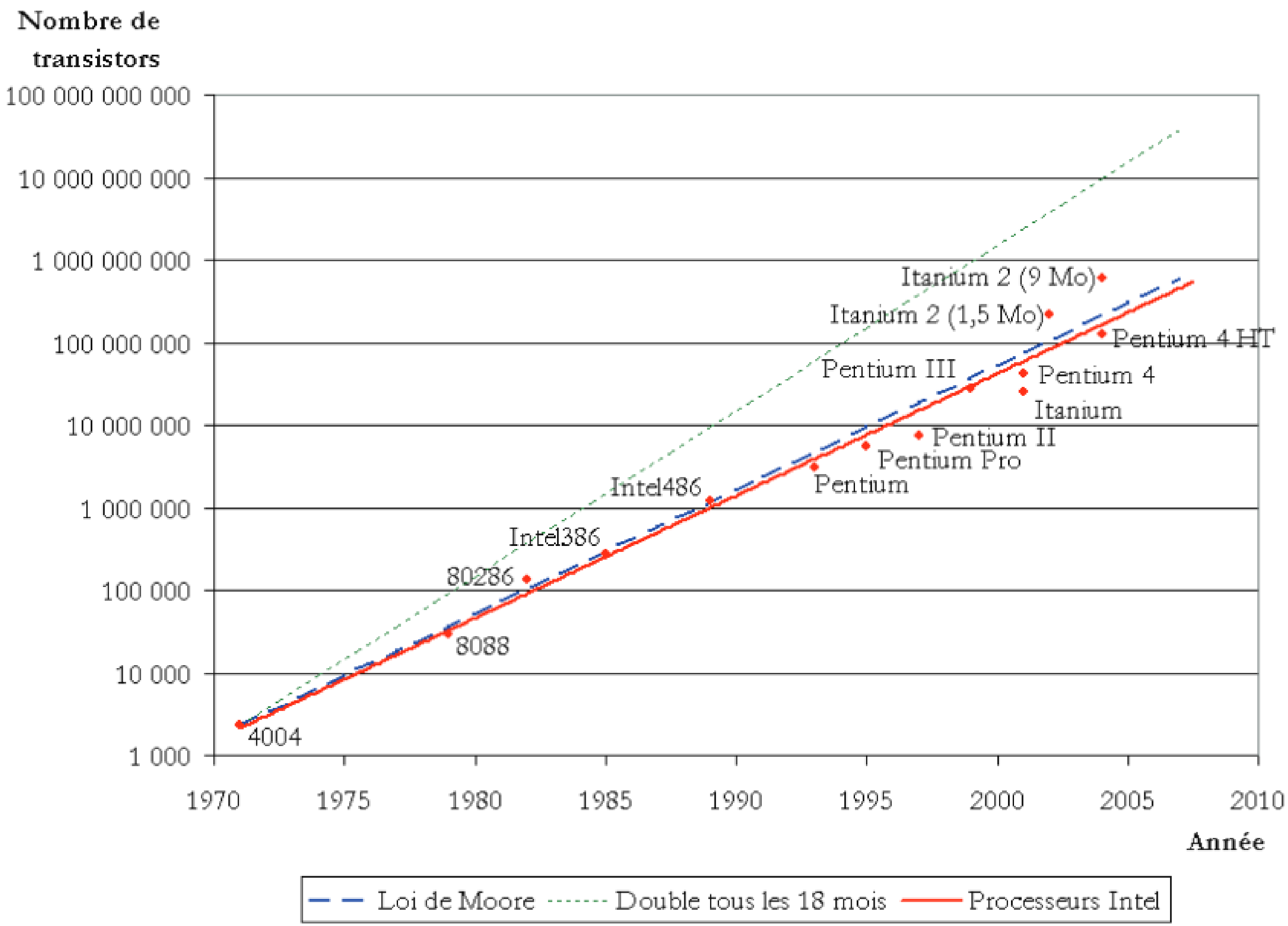
3. Quels Choix pour la Partie Matérielle?

- Utiliser les « Product Selector Guide »

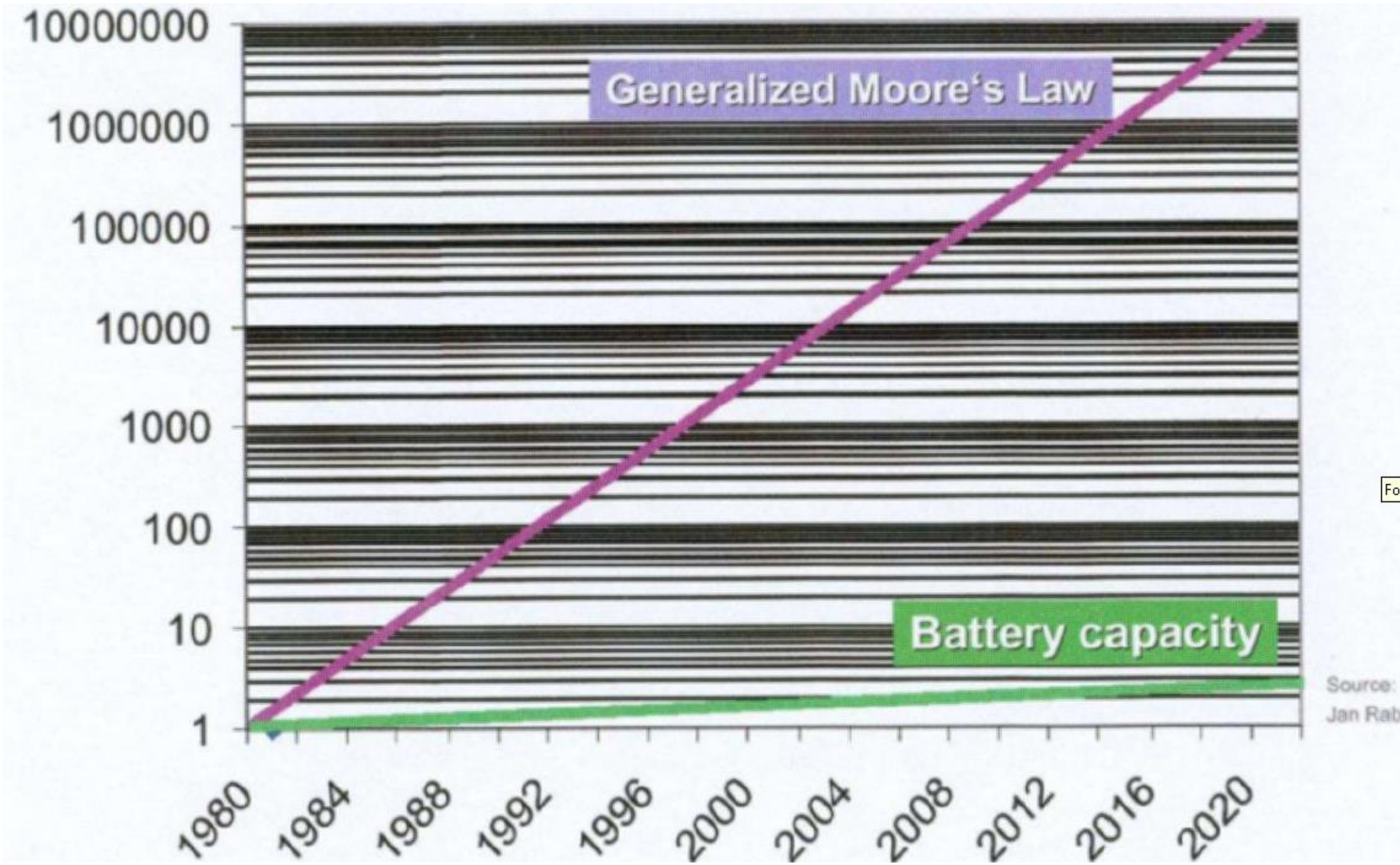
3. Quels Choix pour la Partie Matérielle?



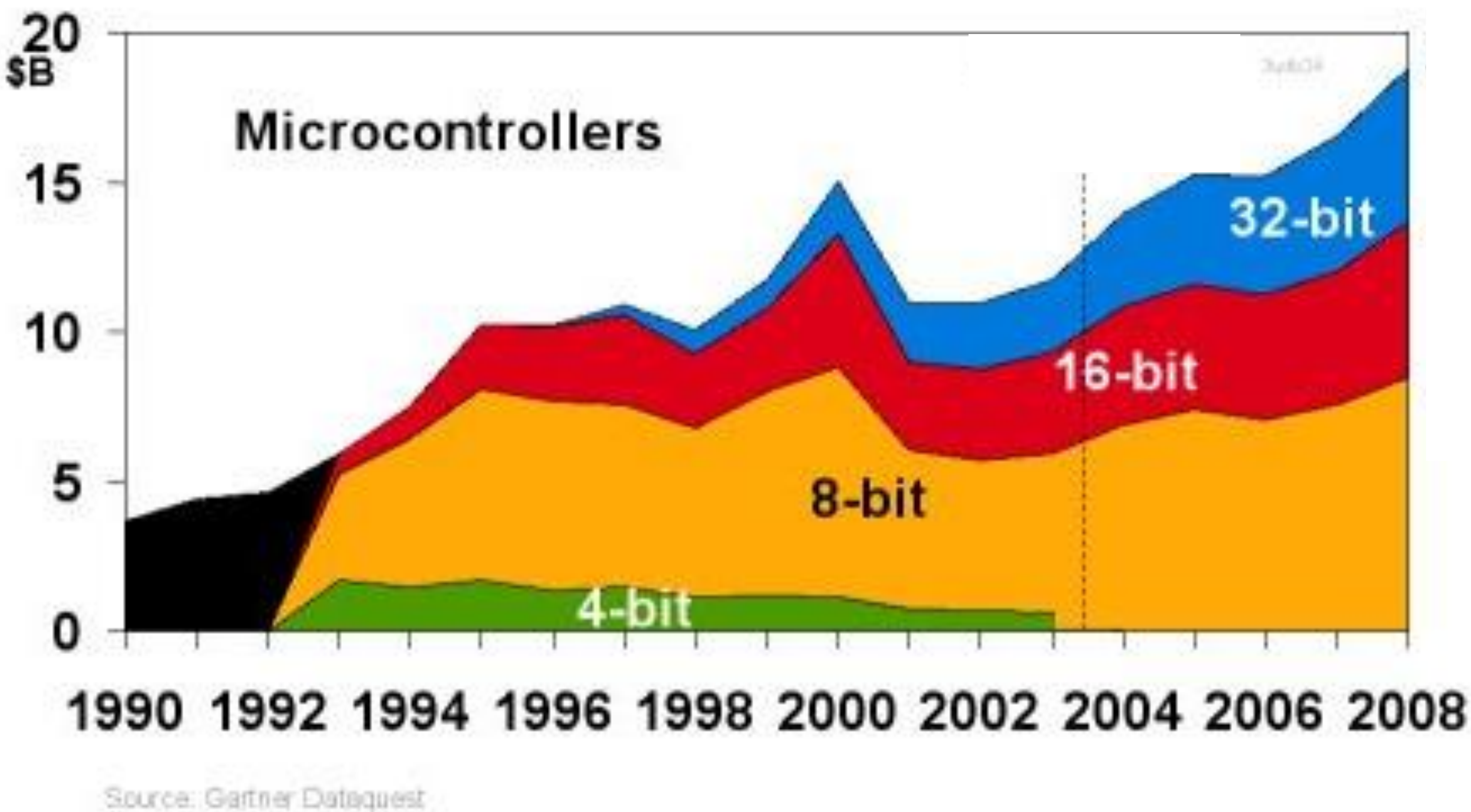
3. Quels choix pour la partie matérielle?



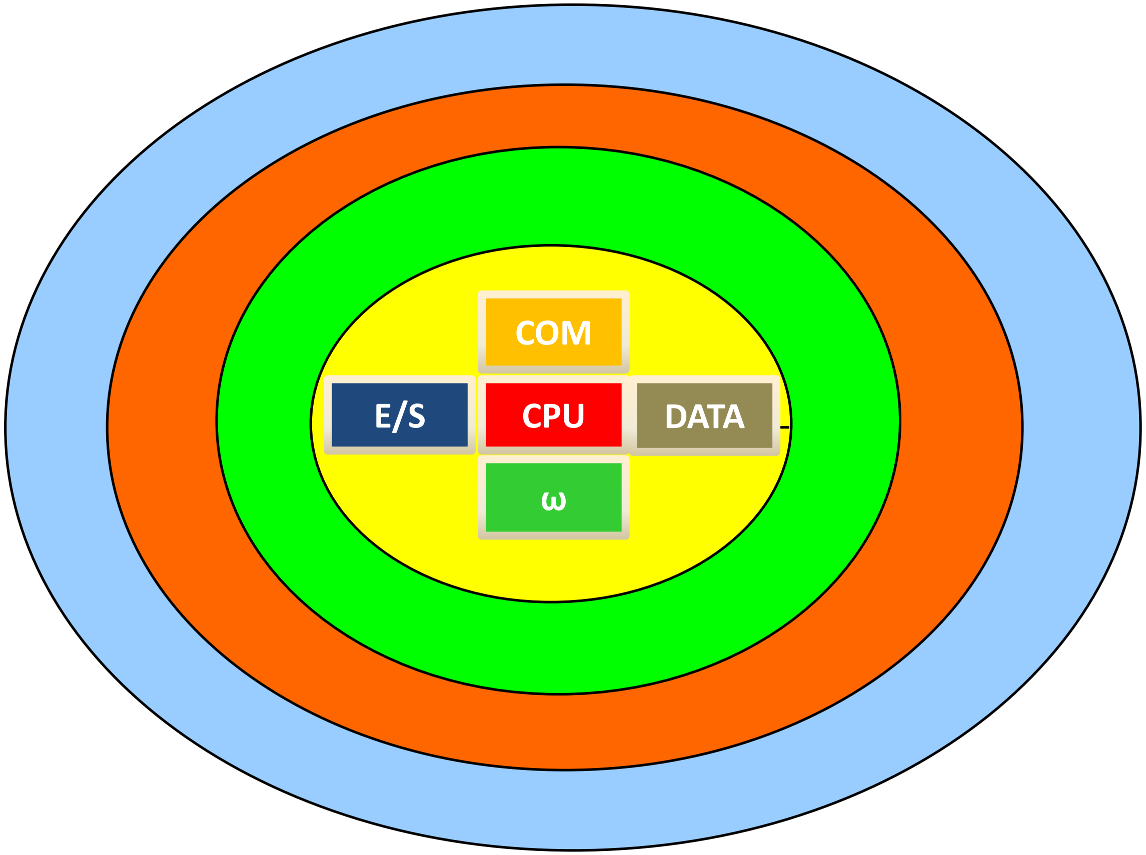
3. Quels choix pour la partie matérielle?



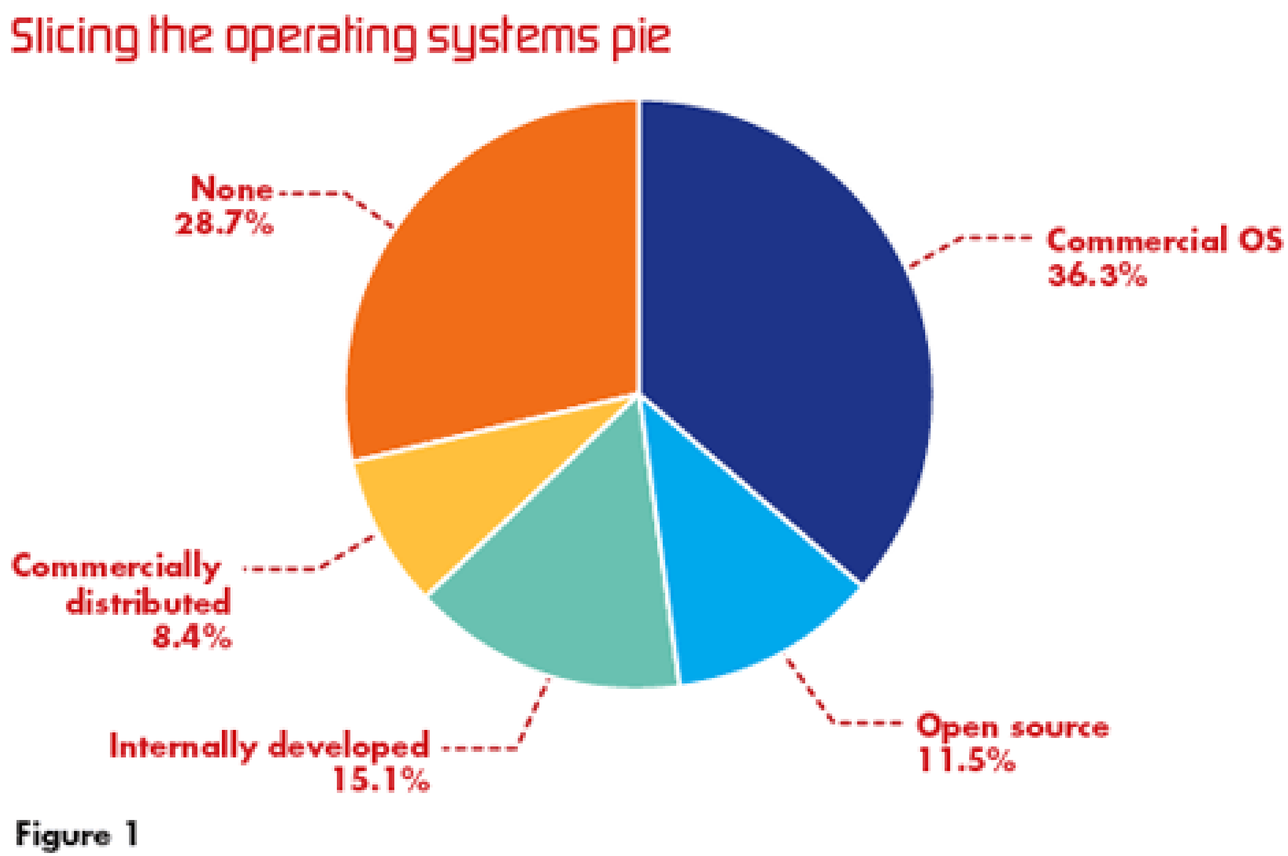
3. Quels choix pour la partie matérielle?



3. VAIS-JE UTILISER UN OS?



3. Vais-je utiliser un OS?



[Turley]

3. VAIS-JE UTILISER UN OS?

Quel type d'OS?

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

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?

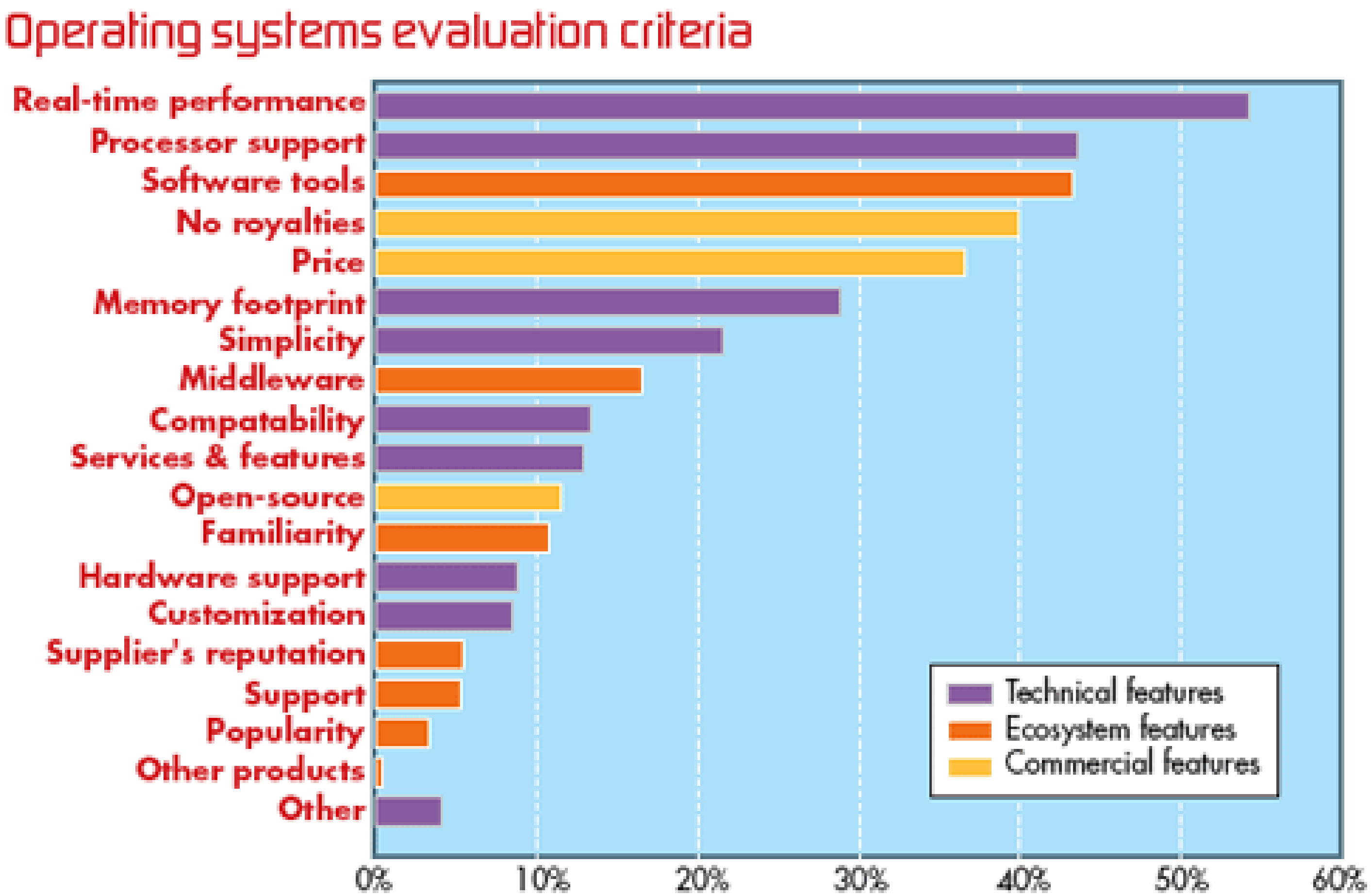


Figure 5

[Turley]

3. VAIS-JE UTILISER UN OS?

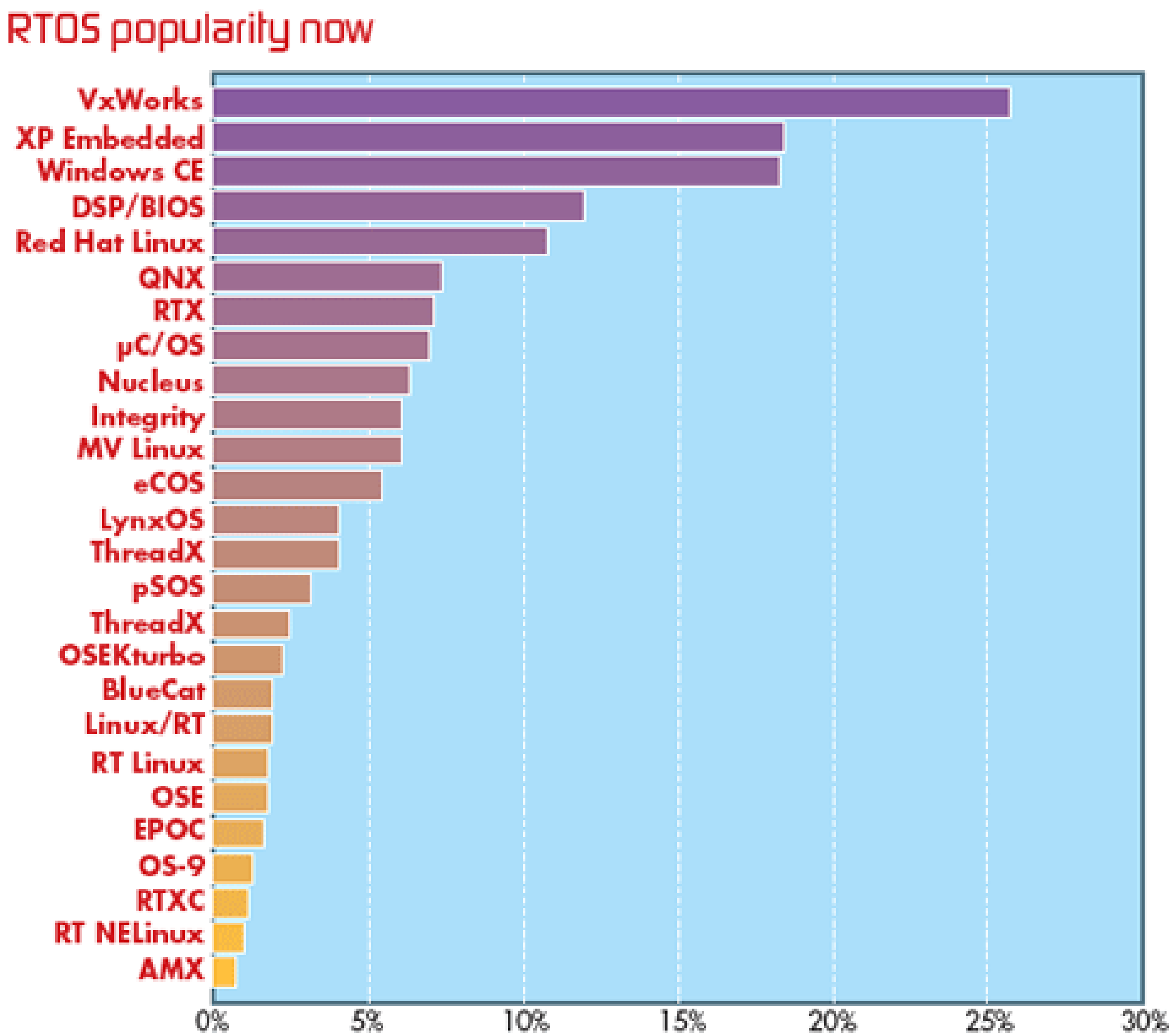


Figure 6

[Turley]

3. VAIS-JE UTILISER UN OS?

Pour créer&initaliser&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**}**)**

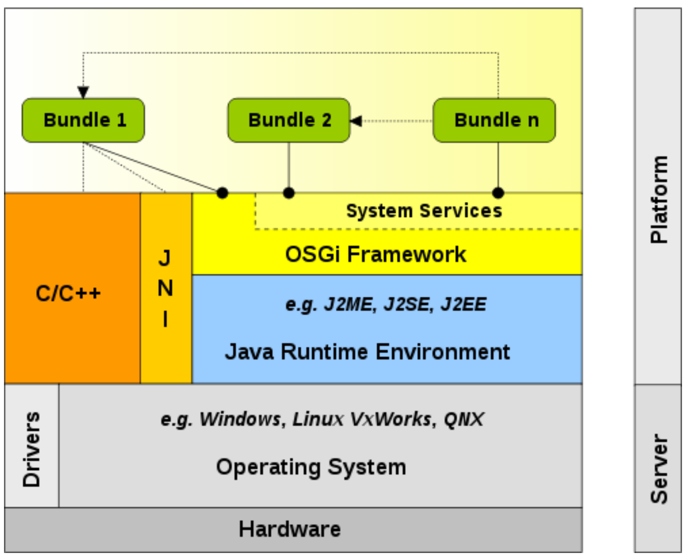
=> Après appel du taskSpawn, une image de la tâche est créée (Process Control Block , pile, programme)

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 OS?

- 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 cadricie (Diagnostic/Monitoring and Framework Layering).

4. Comment vont communiquer les objets entre eux?

- Les réseaux sans fil sont des briques de base du *Web Intelligence* et de l'*Internet of Things*
  ⇒ Il faut savoir identifier les critères pertinents qui vont conditionner le choix de ces briques
  ⇒ Il faut comprendre la caractérisation de ces briques

- 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 déséquencement
  - La latence
  - La gigue
  - …

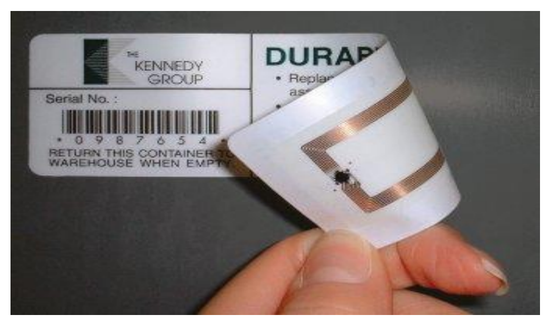
4. COMMENT VONT COMMUNIQUER LES OBJETS ENTRE EUX?

---SYNTHESE---

	Zigbee	Bluetooth	Wi-Fi
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

5. Comment mon objet accède à Internet?

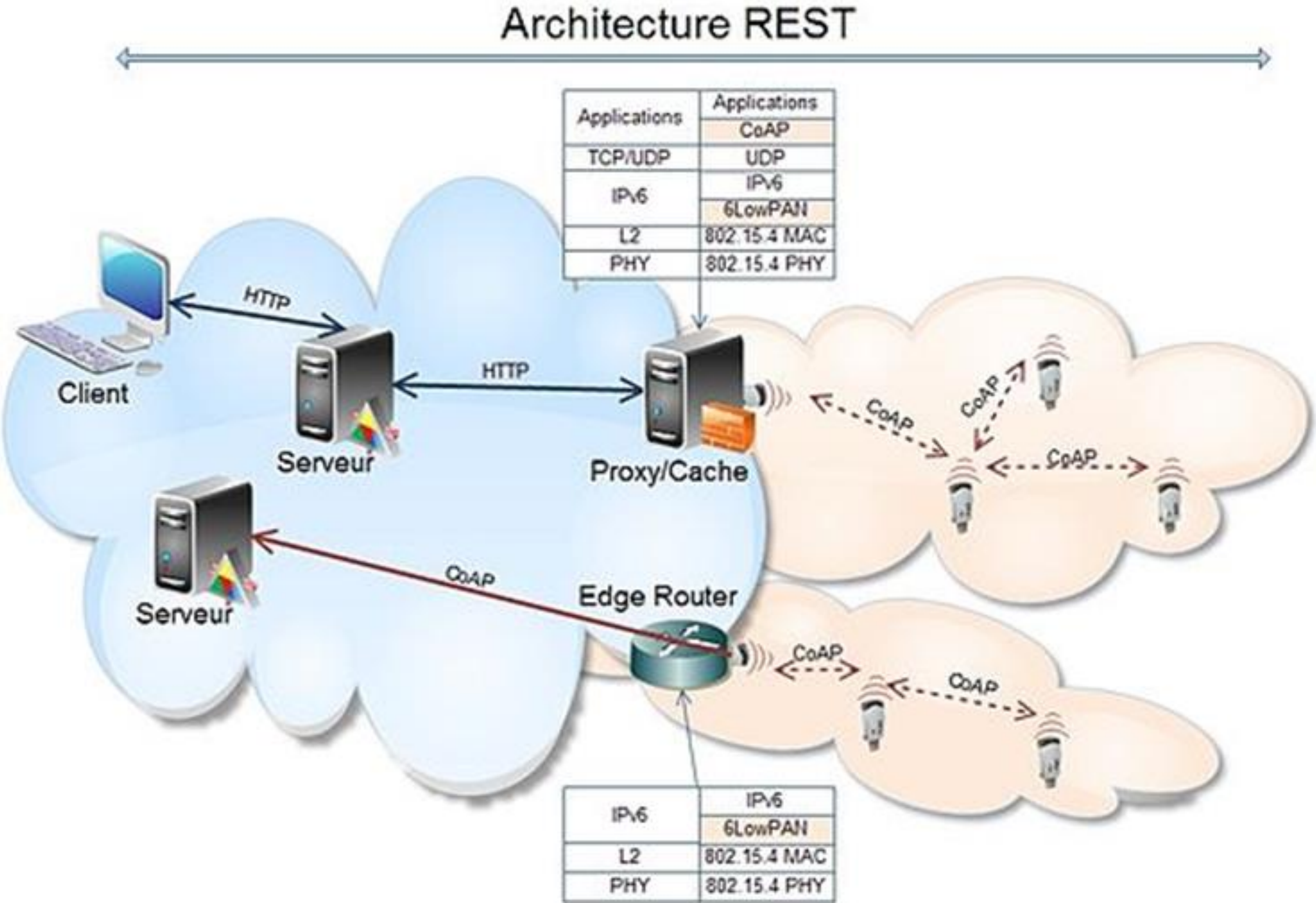
- Deux grands types d'objets physiques
  - Objets communicants et/ou intelligents
    - A base de micro-contrôleur(s) /FPGA équipés d'interface(s) de communication
    - Modèle de COMPORTEMENT EMBARQUÉ sur l'objet
  - Objets « chipless » tagués
    - Objets sur lesquels on a apposé une étiquette RFID (ou autre)



    - Modèle de COMPORTEMENT DÉPORTÉ sur un serveur distant

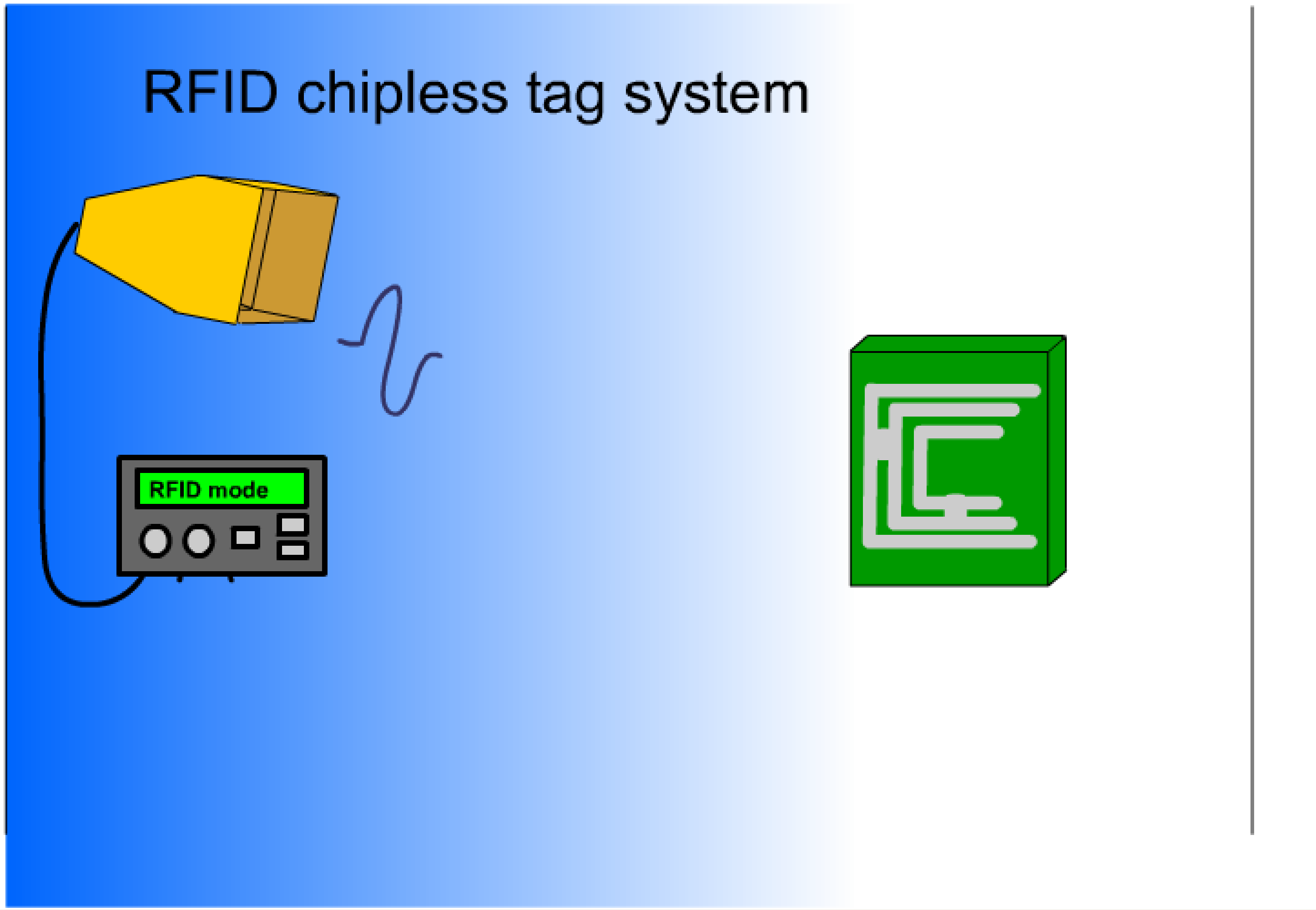
5. Comment mon objet accède à Internet?

Comportement embarqué | Comportement déporté



5. Comment mon objet accède à Internet?

Comportement embarqué | Comportement déporté

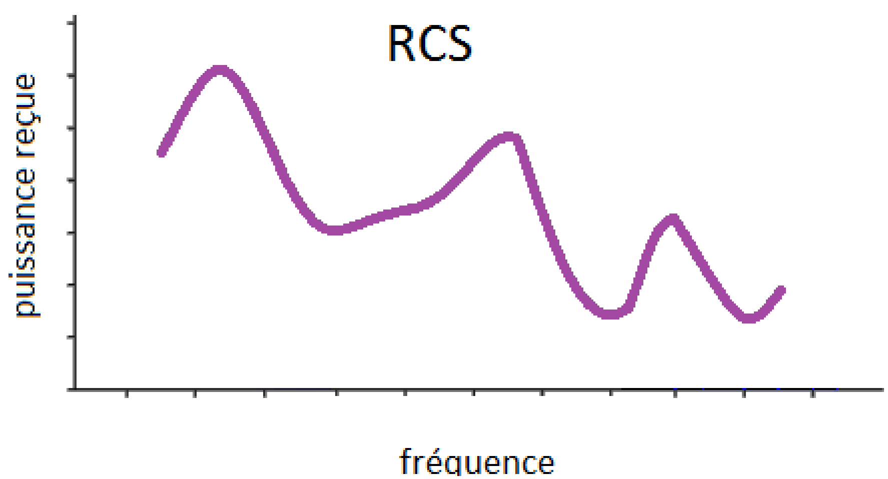


E. Perret

5. COMMENT MON OBJET ACCÈDE À INTERNET?

Comportement embarqué

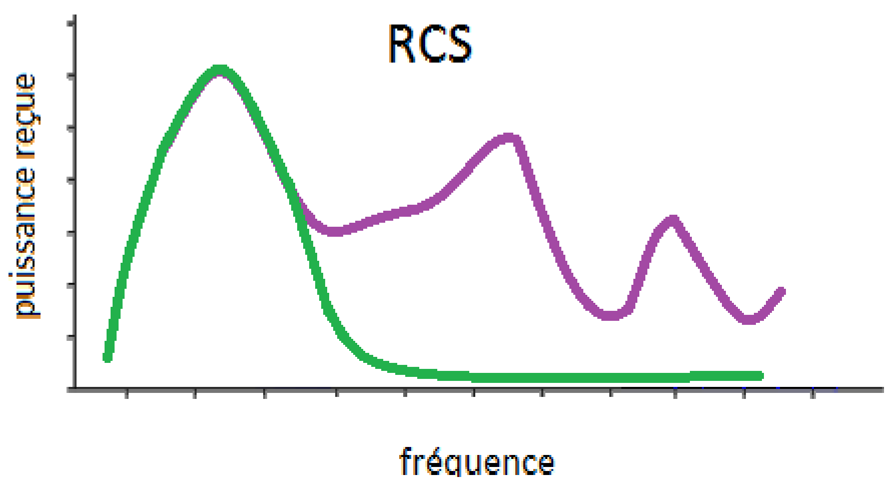
Comportement déporté



O. Rance

5. Comment mon objet accède à Internet?

Comportement embarqué | Comportement déporté

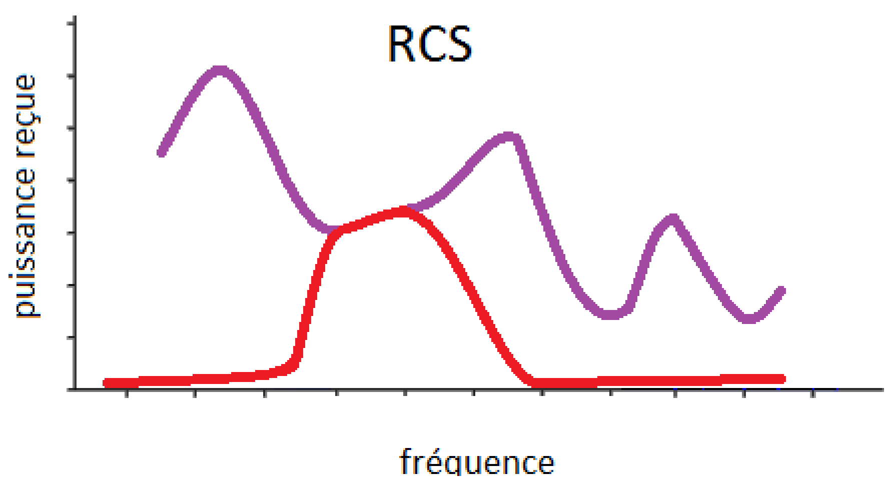


O. Rance

5. Comment mon objet accède à Internet?

Comportement embarqué

Comportement déporté

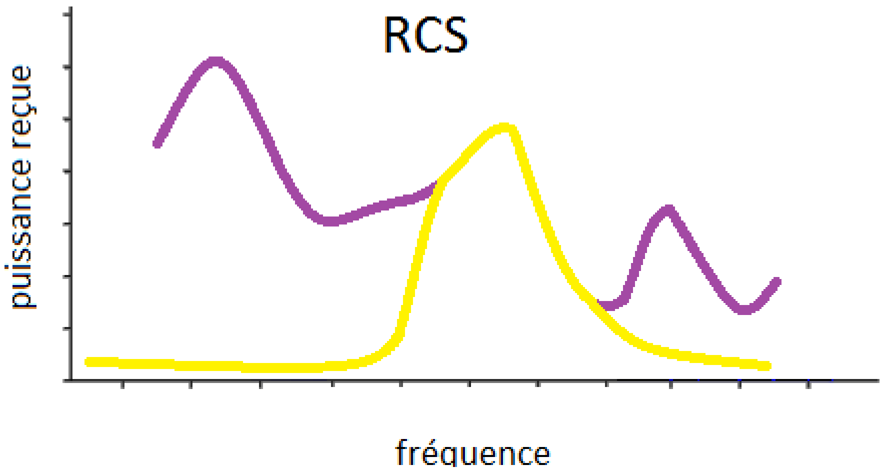


O. Rance

5. Comment mon objet accède à Internet?

Comportement embarqué

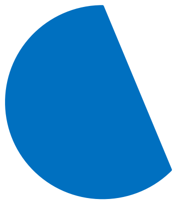
Comportement déporté

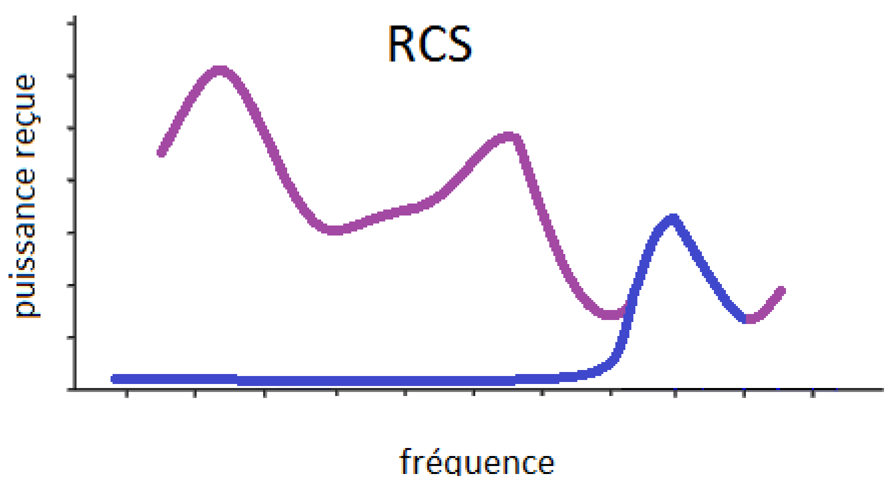


O. Rance

5. Comment mon objet accède à Internet?

Comportement embarqué | **Comportement déporté**



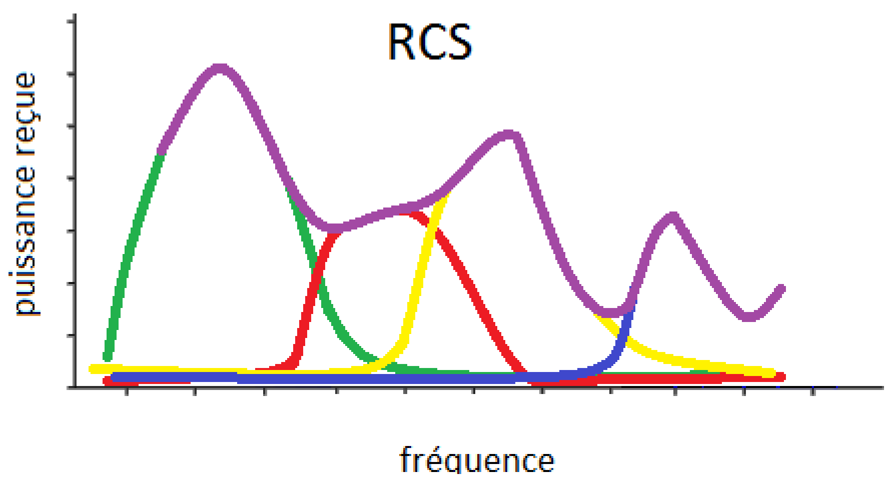


O. Rance

5. Comment mon objet accède à Internet?

Comportement embarqué

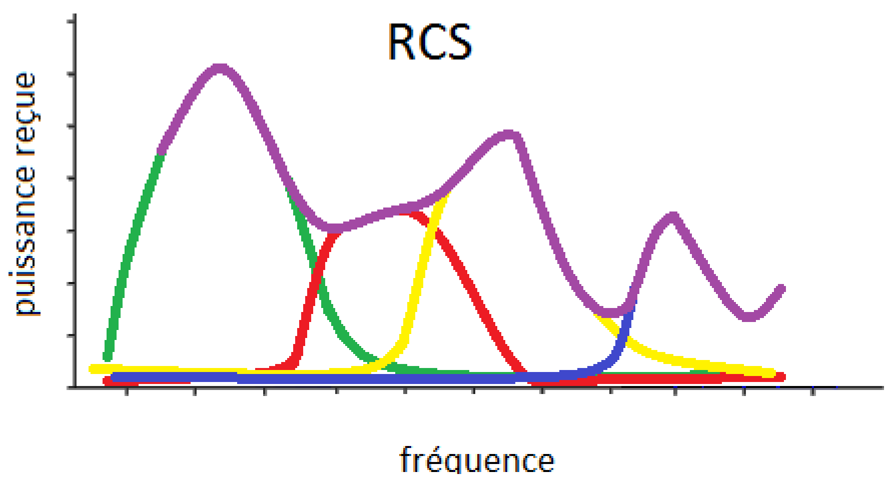
Comportement déporté



O. Rance

5. Comment mon objet accède à Internet?

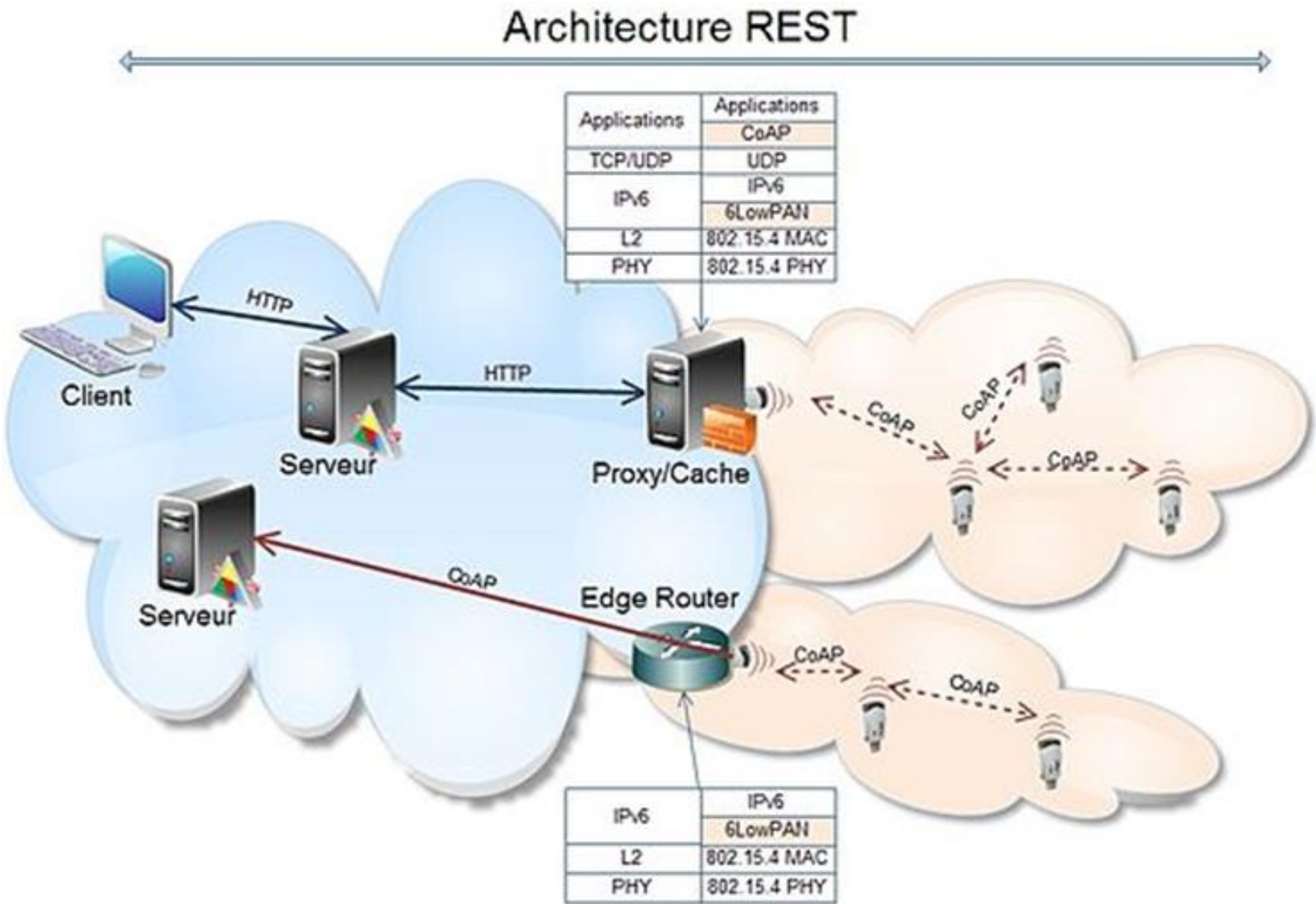
Comportement embarqué | Comportement déporté



O. Rance

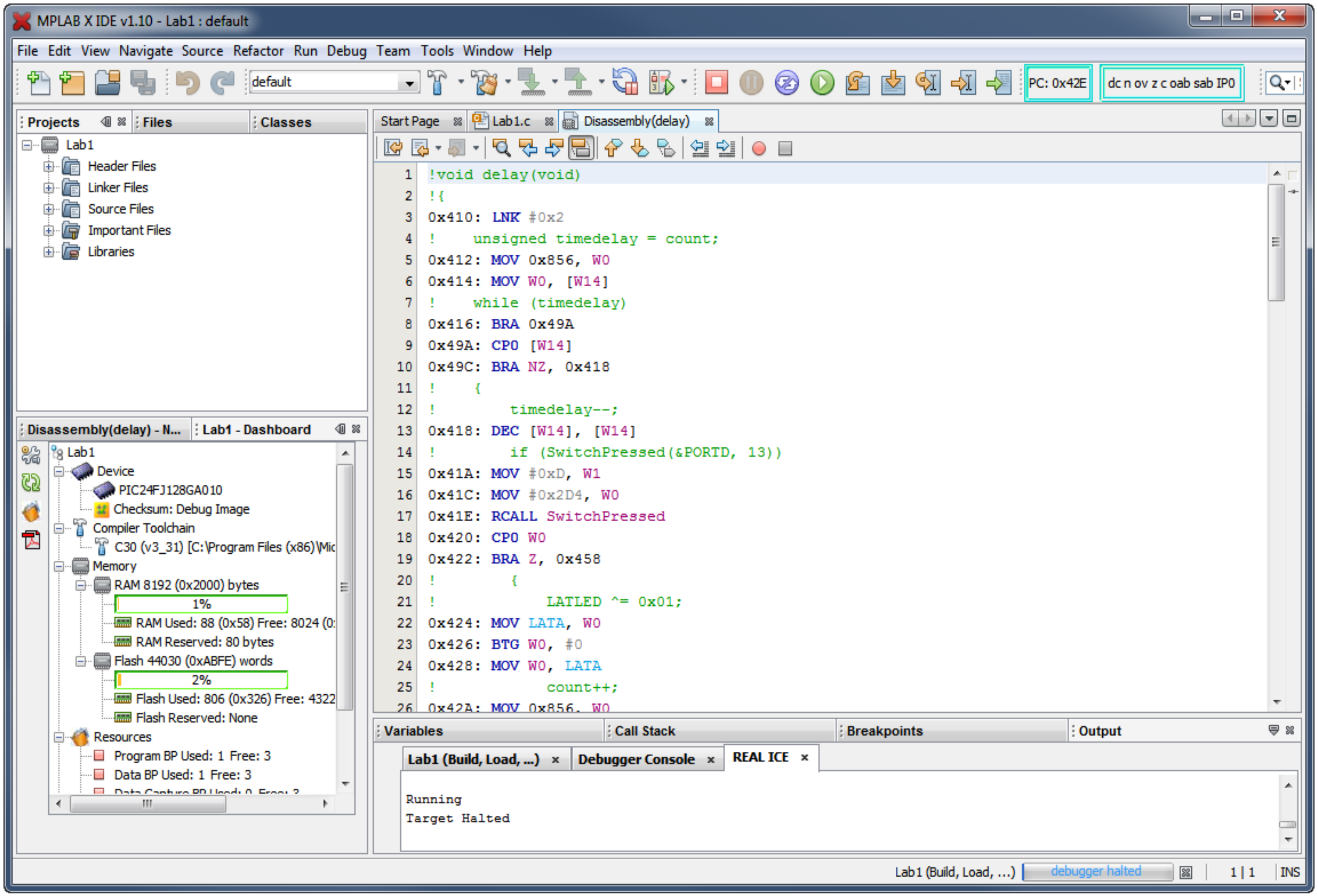
5. Comment mon objet accède à Internet?

Comportement embarqué | **Comportement déporté**



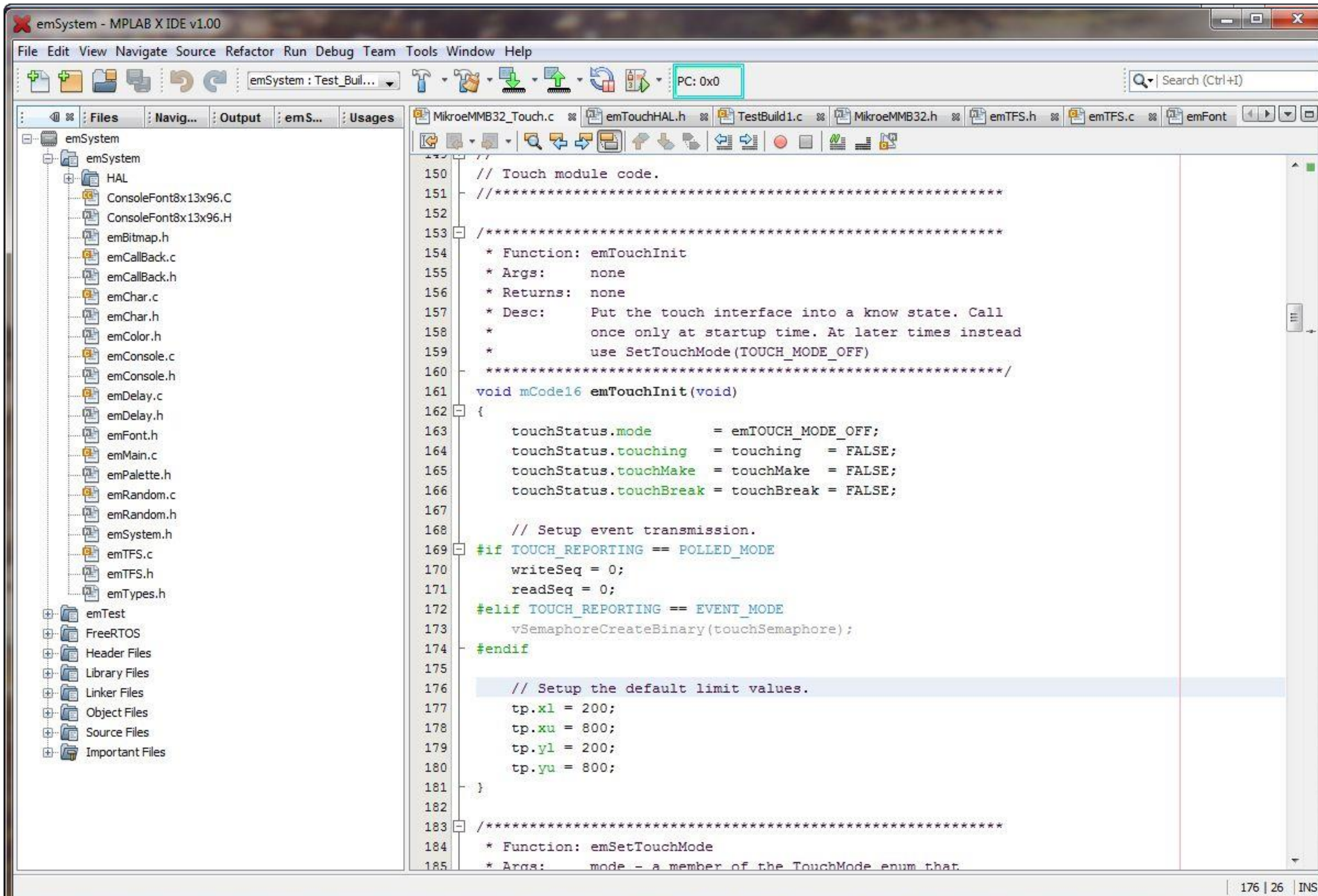
6. JE DÉVELOPPE MON APPLICATION

- Choisir la chaine de développement



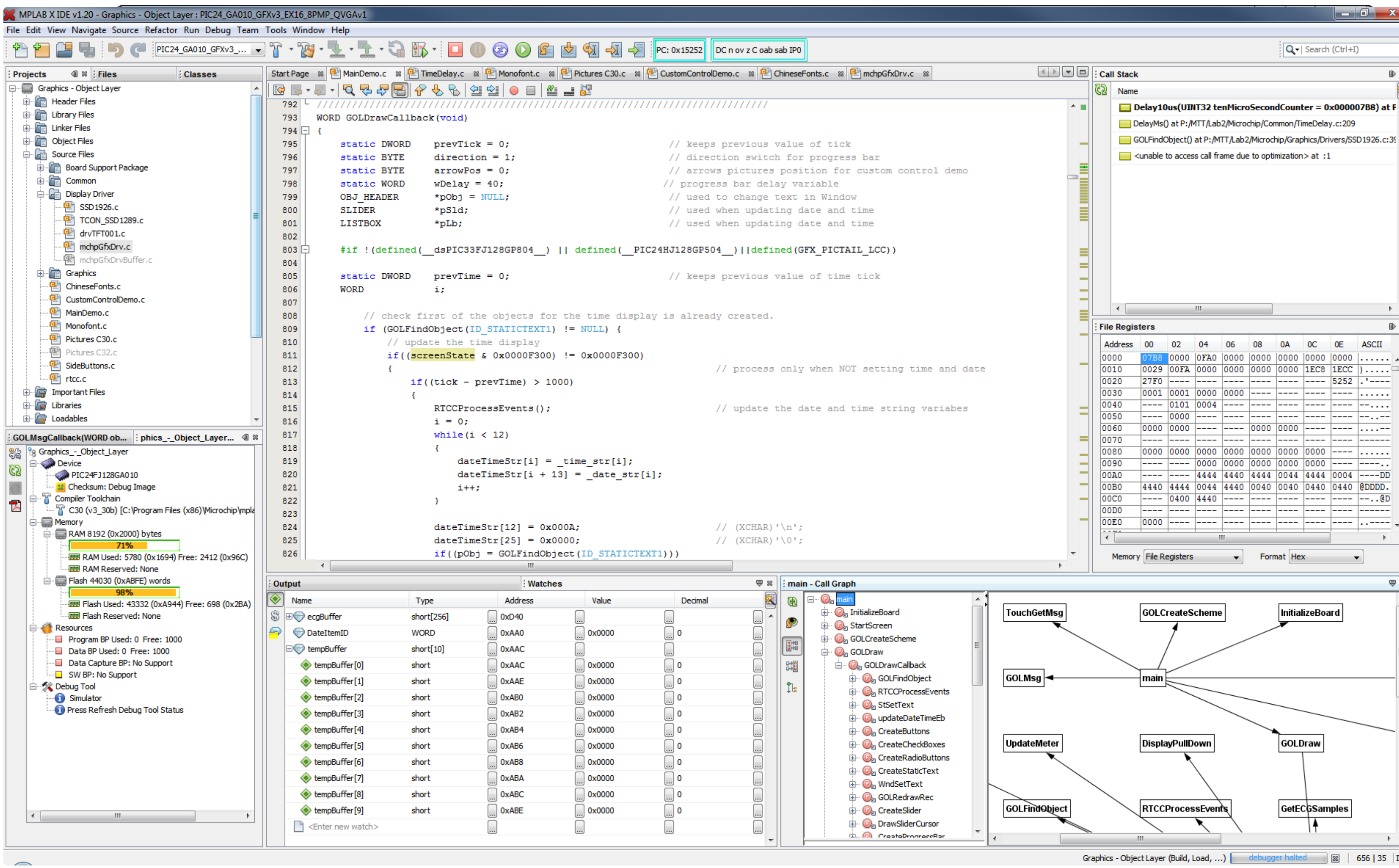
6. JE DÉVELOPPE MON APPLICATION

- Choisir la chaine de développement



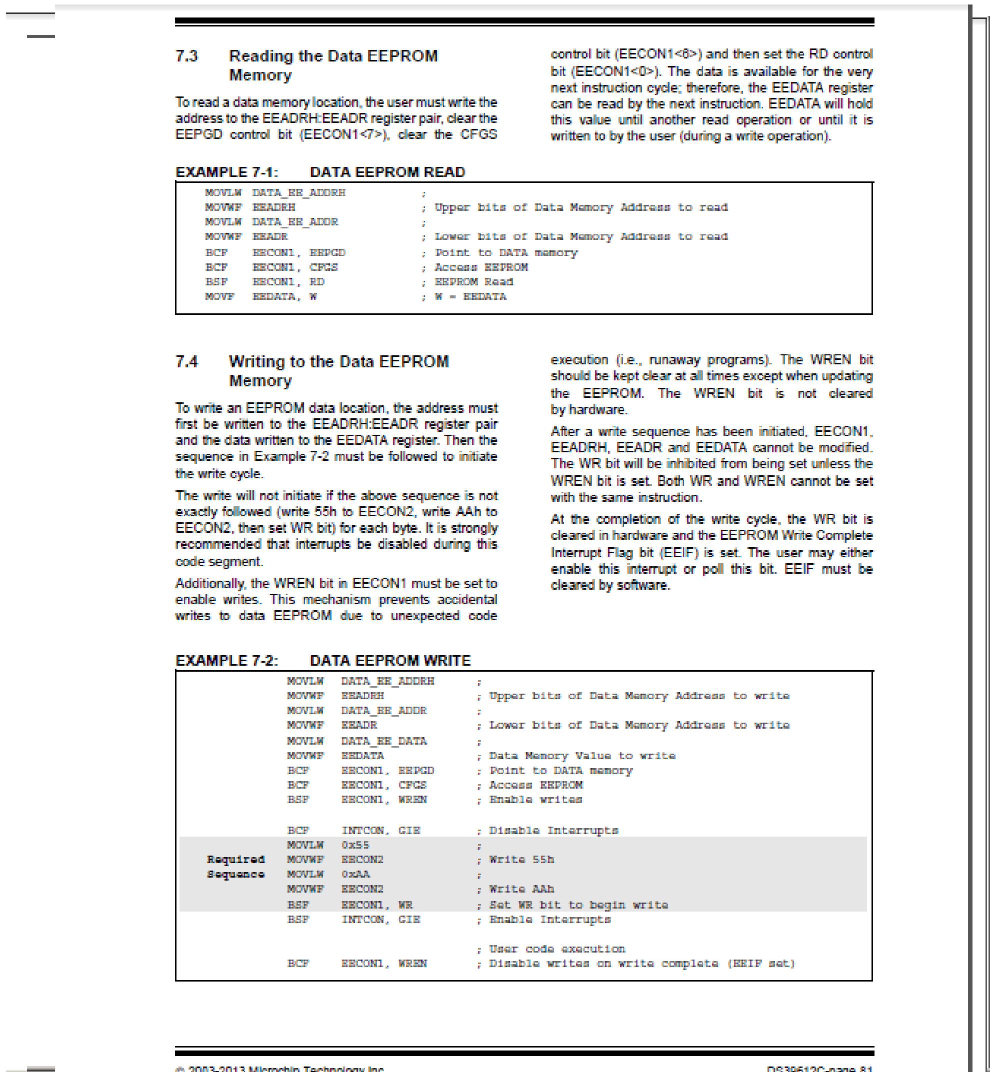
6. Je développe mon application

- Choisir la chaine de développement



6. 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            ;
Required    MOVWF  EECON2          ; Write 55h
Sequence    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)
```

7. JE VALIDE MON SYSTÈME

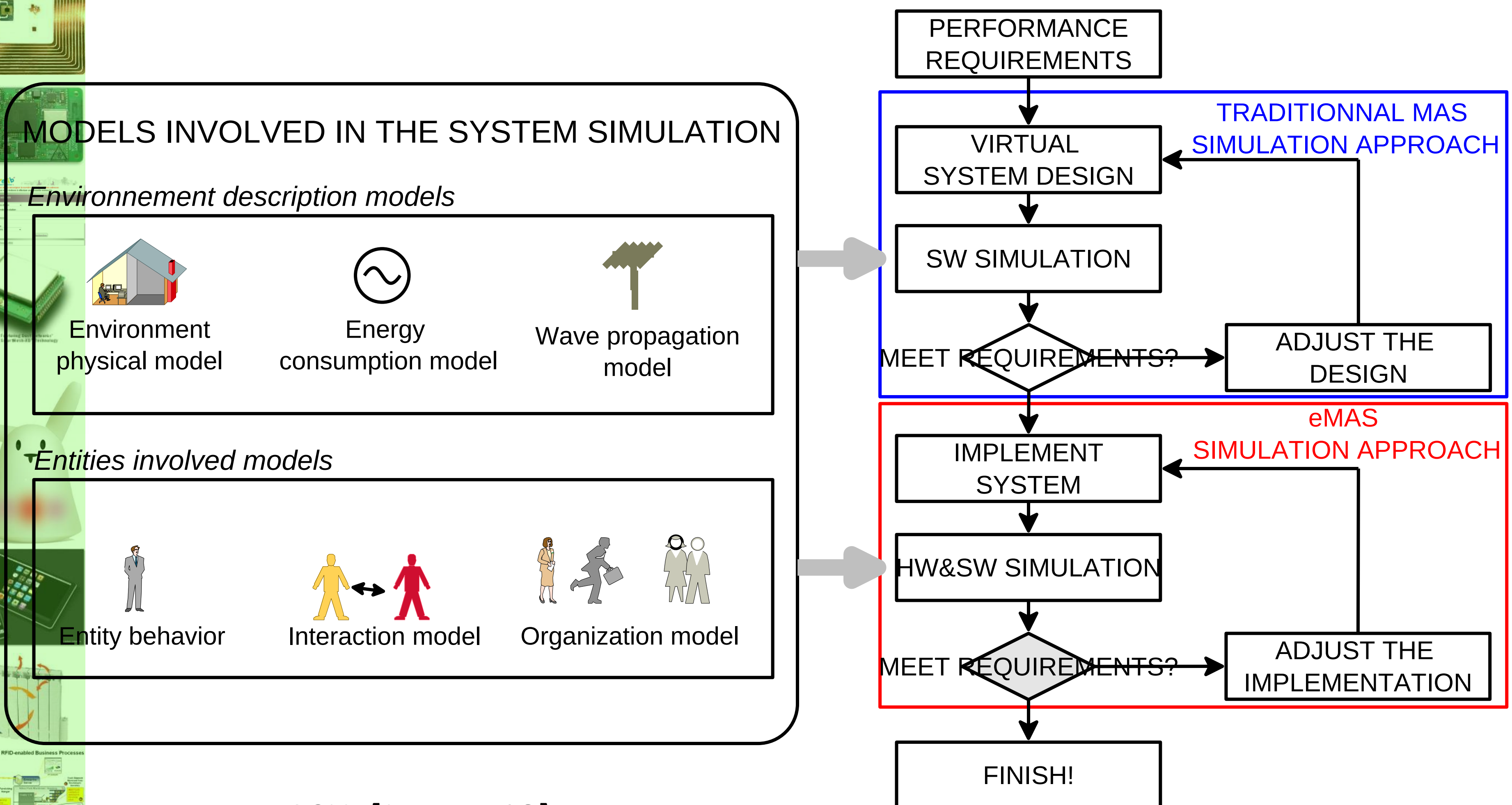
---Validation---

Risques acceptables / Satisfaction des utilisateurs

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

7. JE VALIDE MON SYSTÈME

---Des outils spécifiques--



MASH [Jamont13]

7. Je valide mon système

---Performances énergétiques---

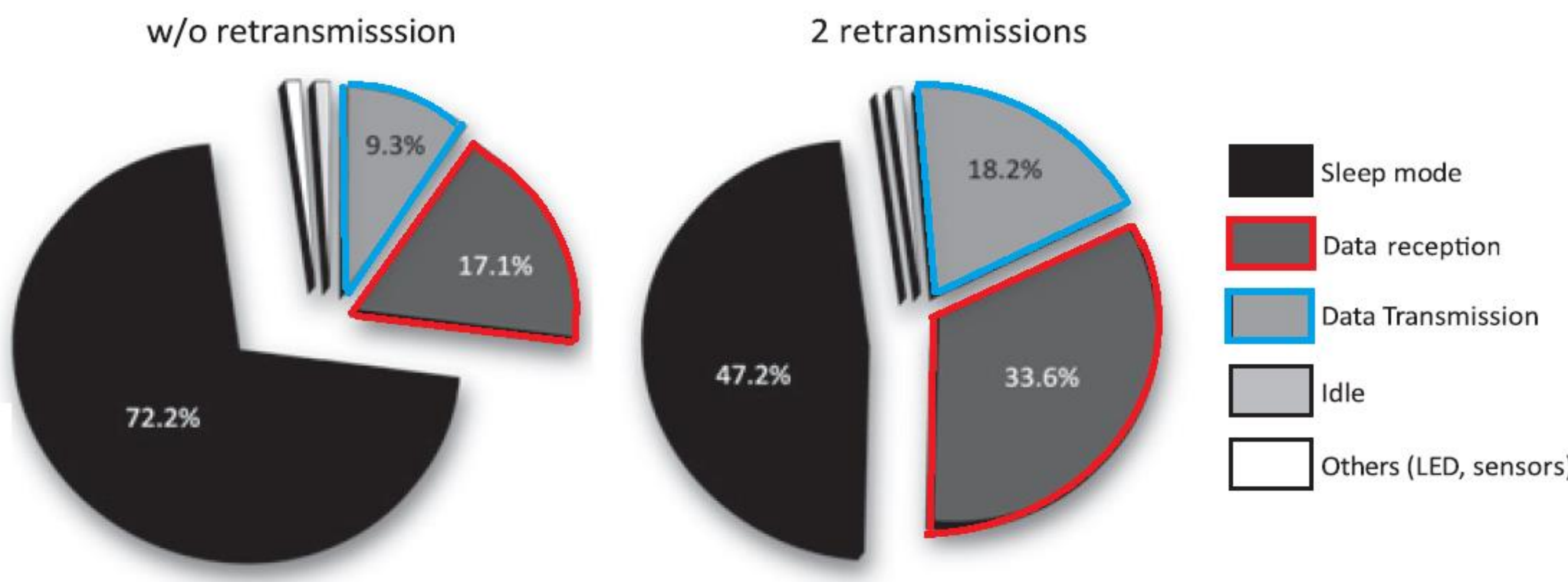


Fig. 6. Energy consumption repartition for different scenari.

[Fourty12]

Merci pour votre écoute

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 Occello, 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