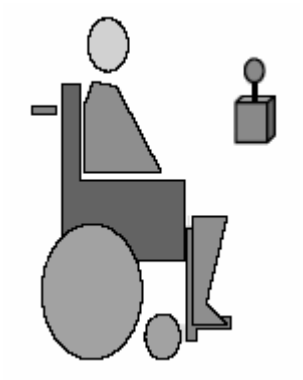
Cours M2

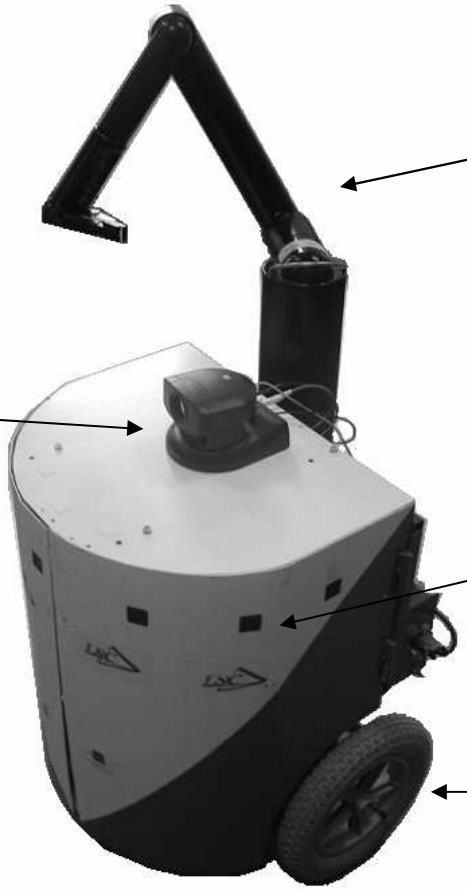
ARPH
Architecture

Sébastien DELARUE

10/12/2006

ARPH





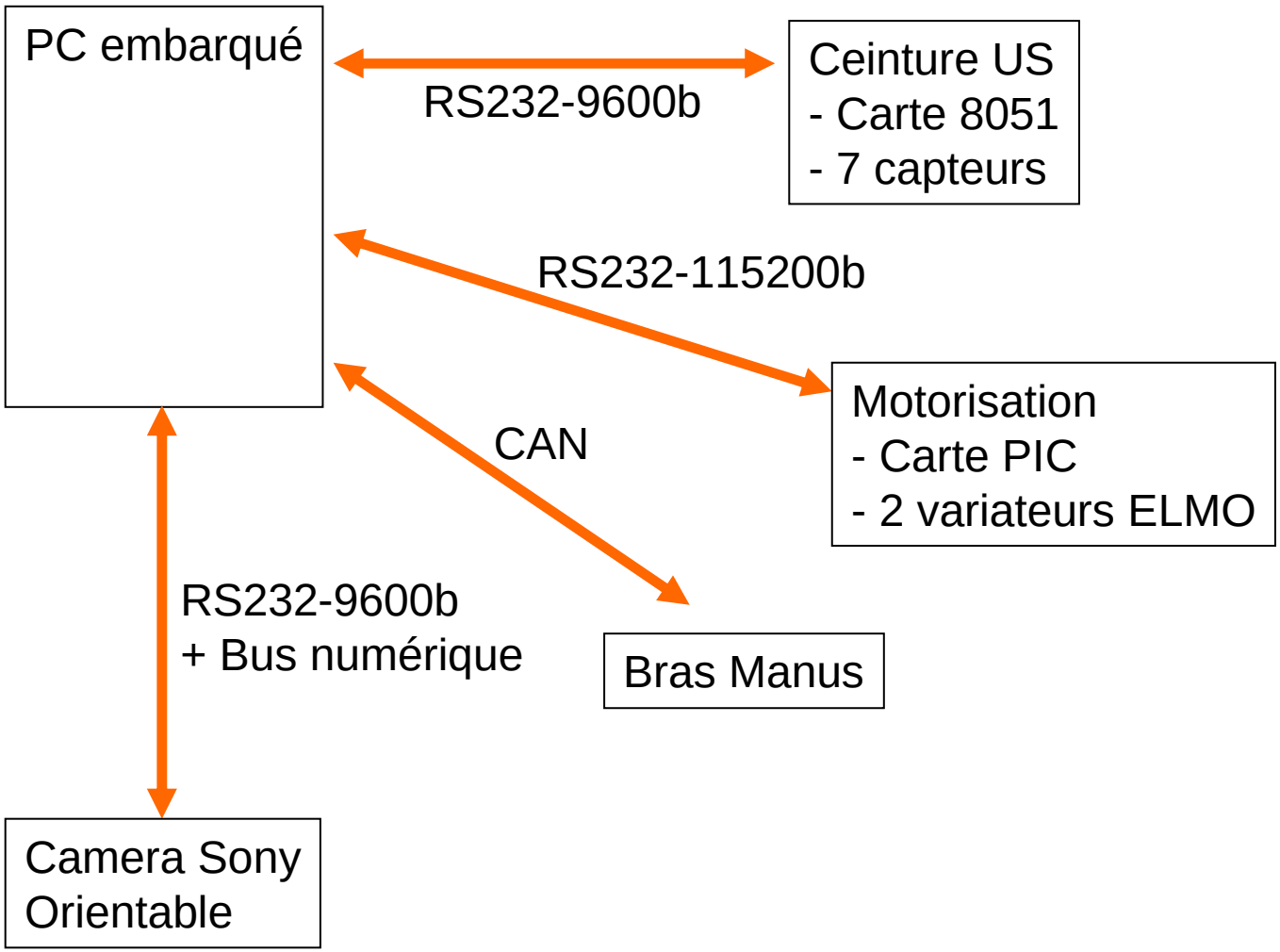
Bras Manus

Caméra orientable

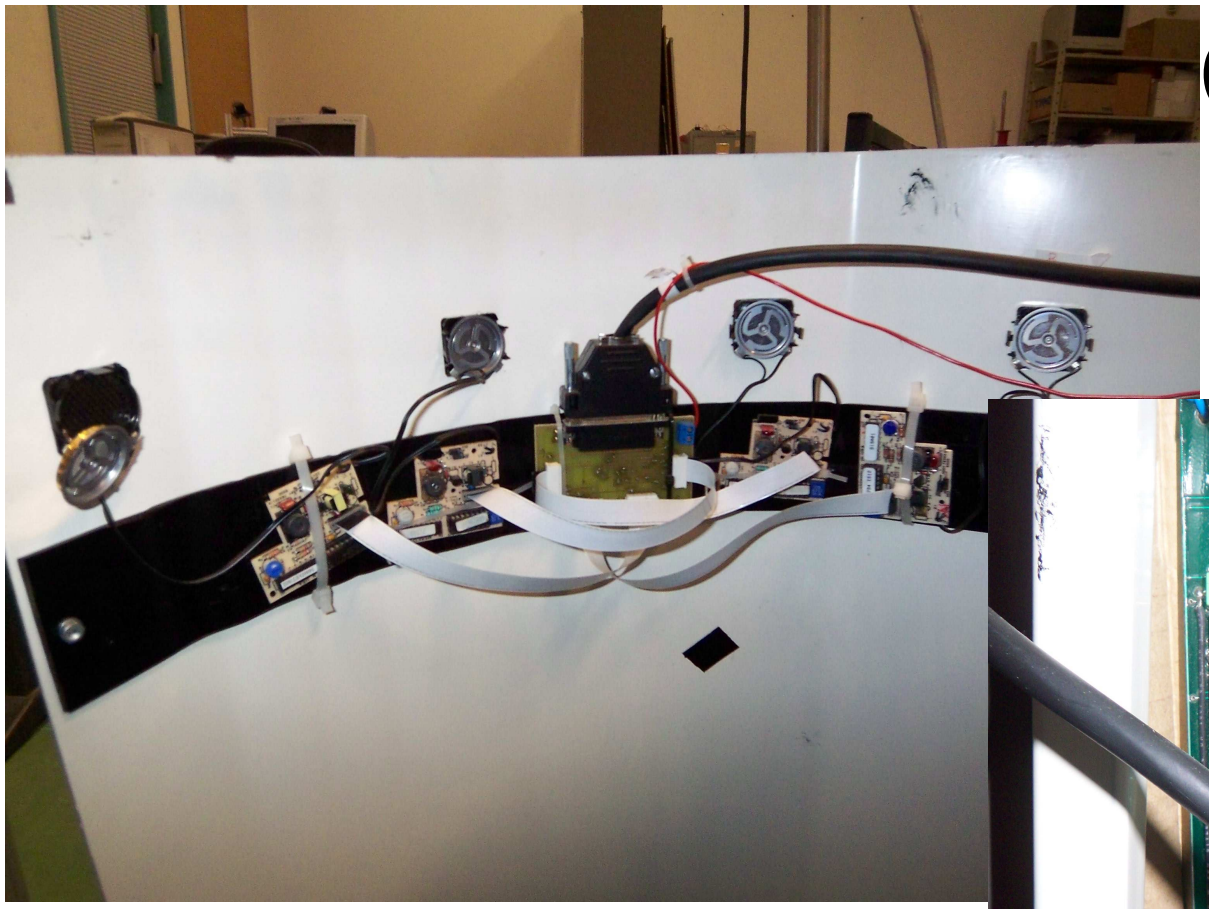
Capteurs US

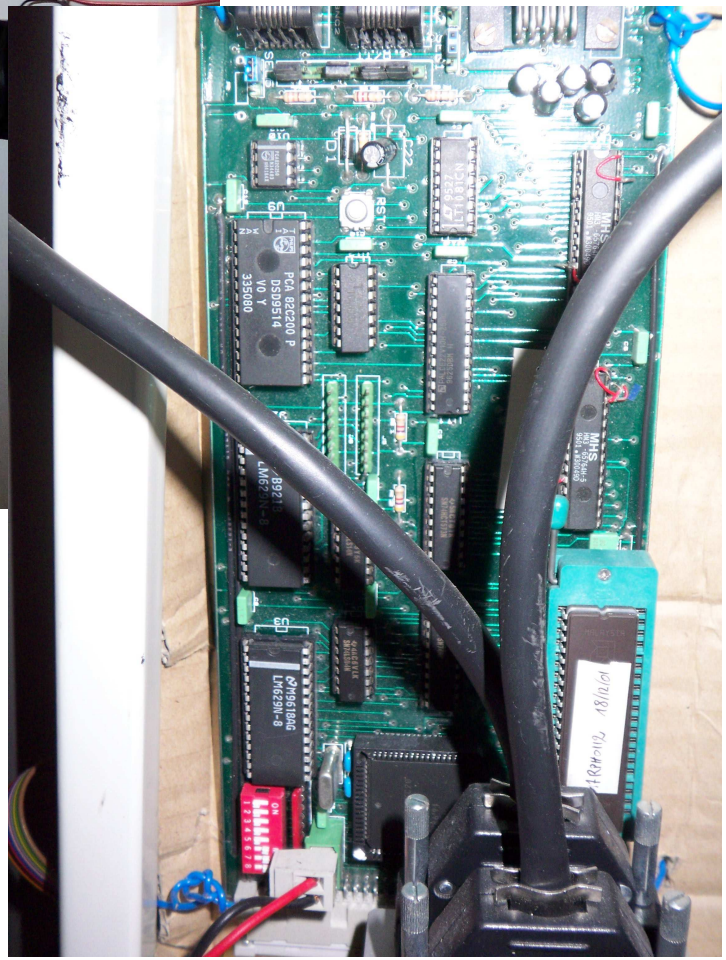
Roues motrices

Interconnexions



Capteur US





Camera

Sony

Liaison RS232 (Zoom, Pan, Tilt)

Camera

-Vidéo
-Audio
-Pan Tilt

Video analogique

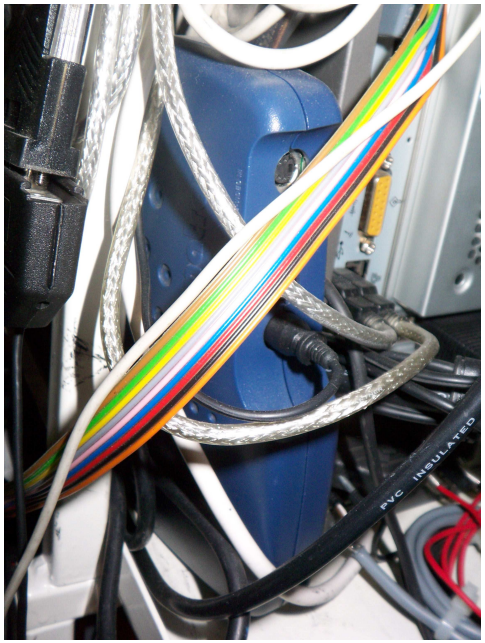
Audio analogique

Aquisition vidéo/audio

Bus numérique

Carte PCI traitement

- Compression MPEG
- 25im/s 388*316 pixels



Protocole Camera

- RS232- 8 bits, pas de parité, 1 bit stop, 9600 bauds
- RCA Audio + Vidéo

Commande

Pan-tiltDrive	Up	8x 01 06 01 VV WW 03 01 FF	VV: pan speed 01 to 18,
	Down	8x 01 06 01 VV WW 03 02 FF	WW: tilt speed 01 to 14
	Left	8x 01 06 01 VV WW 01 03 FF	YYYY: pan position: approx. FC90 to 0370
	Right	8x 01 06 01 VV WW 02 03 FF	(center 0000)
	UpLeft	8x 01 06 01 VV WW 01 01 FF	ZZZZ: tilt position: approx. FED4 to 012C
	UpRight	8x 01 06 01 VV WW 02 01 FF	(center 0000)
	DownLeft	8x 01 06 01 VV WW 01 02 FF	
	DownRight	8x 01 06 01 VV WW 02 02 FF	
	Stop	8x 01 06 01 VV WW 03 03 FF	
	Absolute position	8x 01 06 02 VV WW 0Y 0Y 0Y 0Y 0Z 0Z 0Z 0Z FF	Absolute Position Drive
	Relative position	8x 01 06 03 VV WW 0Y 0Y 0Y 0Y 0Z 0Z 0Z 0Z FF	Relative Position Drive. Set the relative coordina current position to the the target position.
	Home	8x 01 06 04 FF	
	Reset	8x 01 06 05 FF	Pan/Tilt Initialize command

Protocole Camera

Statut

CAM_ShutterPosInq	8x 09 04 4A FF	Y0 50 0Z 0Z 0Z 0Z FF	ZZZZ: position
CAM_IrisPosInq	8x 09 04 4B FF	Y0 50 0Z 0Z 0Z 0Z FF	ZZZZ: position
CAM_GainPosInq	8x 09 04 4C FF	Y0 50 0Z 0Z 0Z 0Z FF	ZZZZ: position
CAM_Backlight Mode Inq	8x 09 04 33 FF	Y0 50 02 FF	On
		Y0 50 03 FF	Off
CAM_MemoryInq	8x 09 04 3F FF	Y0 50 0Z FF	Z: 0 to 5
CAM_KeyLockInq	8x 09 04 17 FF	Y0 50 00 FF	Off
		Y0 50 02 FF	On
CAM_IDInq	8x 09 04 22 FF	Y0 50 0Z 0Z FF	ZZ: ID
VideoSystemInq	8x 09 06 23 FF	Y0 50 00 FF	NTSC
		Y0 50 01 FF	PAL
Wide_conLensInq	8x 09 07 26 FF	Y0 50 00 0Z FF	Z: lens No.
Pan-tiltModeInq	8x 09 06 10 FF	Y0 50 ZZ ZZ FF	ZZZZ: status
Pan-tiltMaxSpeedInq	8x 09 06 11 FF	Y0 50 WW ZZ FF	WW: pan, ZZ: tilt
Pan-tiltPosInq	8x 09 06 12 FF	Y0 50 0W 0W 0W 0W	WWWW: pan
		0Z 0Z 0Z 0Z FF	ZZZZ: tilt
DatascreenInq	8x 09 06 06 FF	Y0 50 02 FF	On
		Y0 50 03 FF	Off
AT/MD_ModeInq	8x 09 07 22 FF	Y0 50 00 FF	Normal mode
		Y0 50 01 FF	AT mode
		Y0 50 02 FF	MD mode
AT_ModeInq	8x 09 07 23 FF	Y0 50 ZZ ZZ FF	ZZ: status

Protocole Manus (1)

CAN

- Startup/initialisation mode (cbox0)
- Cartesian control (cbox1)
- Joint control/ Degree of freedom control (cbox4)
- Folding out (cbox5)
- Folding in (cbox6)

INFO MANUS			**REPONSE DU PC**			**Description**	**Explication du message**
ID	Rtr	Taille					
0x350	0	8	Pas de réponse : lire le message			2 octets pour le statut	Warning, Error, foldstatus
						Orientation axes 1 à 3	6 octets: MSB et LSB des axes 1 à 3
0x360	0	8				Orientation axes 4 à 7	8 octets: MSB et LSB des axes 4 à 7
INFO MANUS			**REPONSE DU PC**			**Description**	**Explication du message**
ID	Rtr	Taille	ID	Rtr	Taille		
0x37F	1	0	0x370	0	0	Sélection de la cbox0	Libre / Initialisation
			0x371	0	8	Sélection de la cbox1	Déplacements sur les axes 0 à 7
			0x374	0	8	Sélection de la cbox4	Déplacements des articulations 0 à 7
			0x375	0	0	Sélection de la cbox5	Dépliement
			0x376	0	0	Sélection de la cbox6	Repliement

Protocole Manus (2)

Ordre de mouvement cartésien

DEPLACEMENTS				
Octets	**Paramètre**	**Unité**	**Minimum**	**Maximum**
1	(A0) Lift Unit	Up/off/down -1 0 1	-1	1
2	(A1) X	0.022 (mm)	0 (unités)	127 (unités)
3	(A2) Y			
4	(A3) Z			
5	(A4) Yaw	0.1 (degrés)	0 (unités)	10 (unités)
6	(A5) Pitch			
7	(A6) Roll			
8	(A7) Gripper	0.1 (mm)	0 (unités)	15 (unités)

Protocole Manus (3)

Trame Informations codeurs

ID	Octet	Ex. de valeur de l'octet	Valeur	cbox1	cbox4
0x350	**1**	1	Movement error	Statut	Statut
	2	3	Blocked DOF	Message	Message
	3	11	11*256 + 14 = 2830 (unités)	X	Axe1
	4	14			
	5	7	7*256 + 3 = 1795 (unités)	Y	Axe2
	6	3			
	7	5	5*256 + 6 = 1286 (unités)	Z	Axe3
	8	6			
0x360	**1**	5	5*256 + 2 = 1282 (unités)	Yaw	Axe4
	2	2			
	3	12	12*256 + 12 = 3084 (unités)	Pitch	Axe5
	4	12			
	5	5	5*256 + 0 = 1280 (unités)	Roll	Axe6
	6	0			
	7	0	0*256 + 6 = 6 (unités)	Gripper	Gripper
	8	6			

Protocole Manus (4)

Exemple d'échange entre le MANUS et le PC:

t=20ms Le MANUS envoie un message 0x350. Le PC le lit et l'interprète.
t=40ms Le MANUS envoie un message 0x360. Le PC le lit et l'interprète.
t=60ms Le MANUS envoie un message 0x37F pour que le PC lui donne la « control box » et les déplacements désirés.

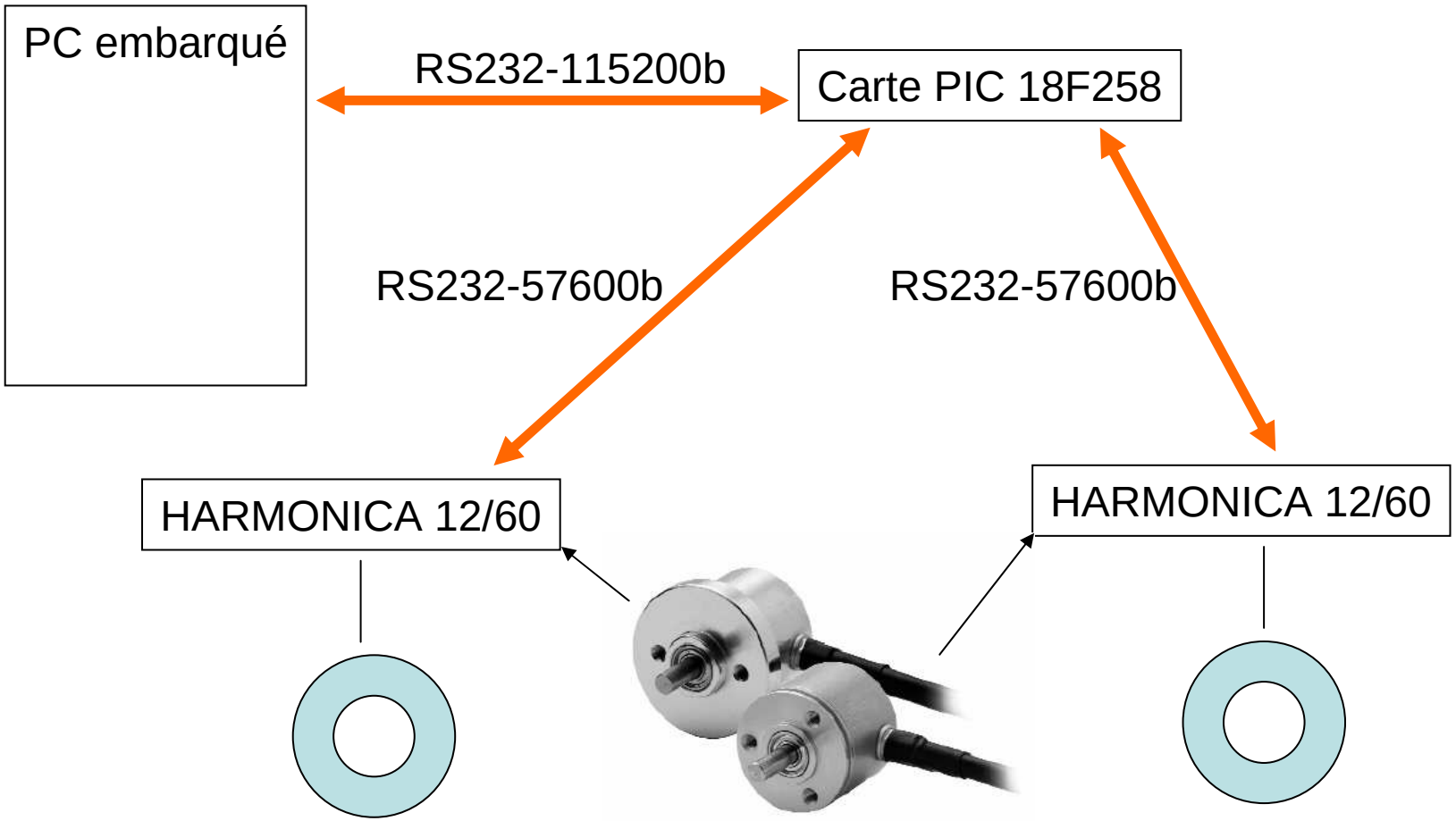
60<t<79ms Le PC répond avec 0x371, tous les octets à 0 : le MANUS se met en mode cartésien, et ne bouge pas.

t=80ms Le MANUS envoie un message 0x350. Le PC le lit et l'interprète.
t=100ms Le MANUS envoie un message 0x360. Le PC le lit et l'interprète.
t=120ms Le MANUS envoie un message 0x37F pour que le PC lui donne la « control box » et les déplacements désirés.

120<t<139ms Le PC répond avec 0x371, le 2ème octet à 1: le MANUS reste en mode cartésien, et se déplace le long de l'axe X, à vitesse 1.

…….

Motorisation



Codeurs incrémentaux KUBLER

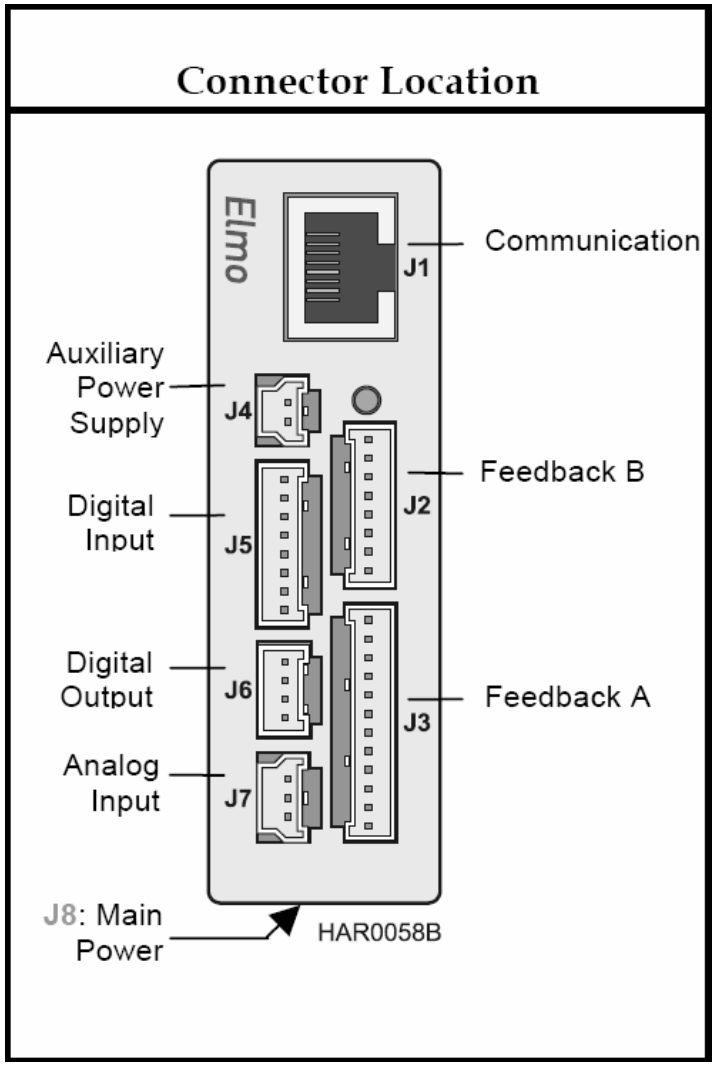
Motorisation

Matériel

Variateur Harmonica (1)

-12A 60V
-Mémoire interne
-Communication par CAN ou RS 232
-Capture d’événement
-Courbe d’accélération
-Protection surpuissance
-Contrôle en position/vitesse/accélération
-….

Variateur Harmonica (2)



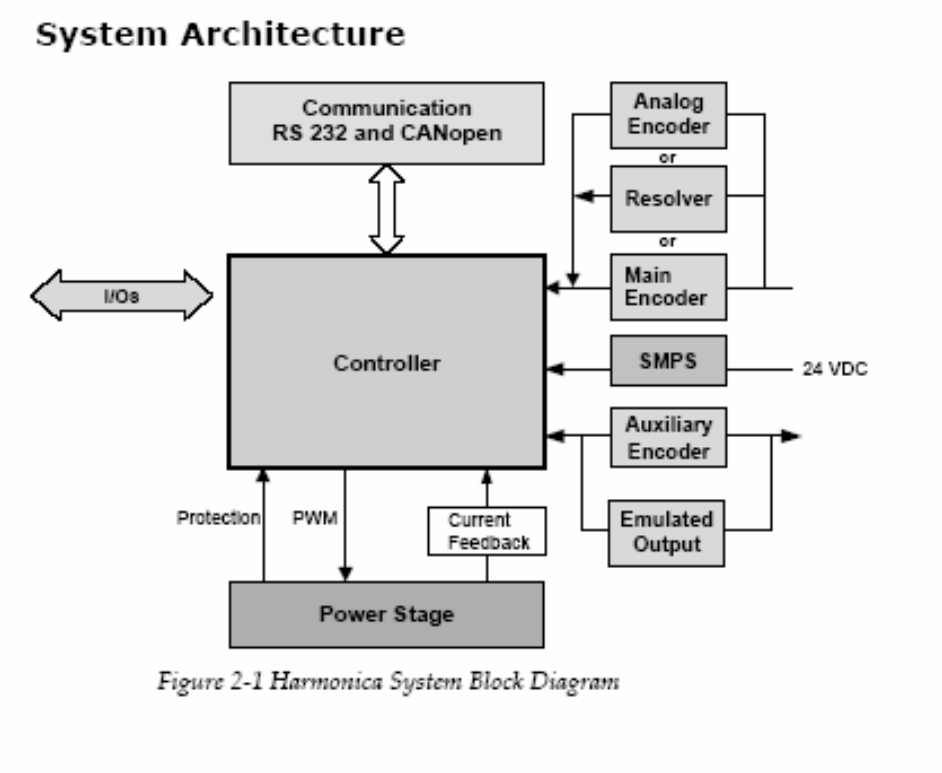
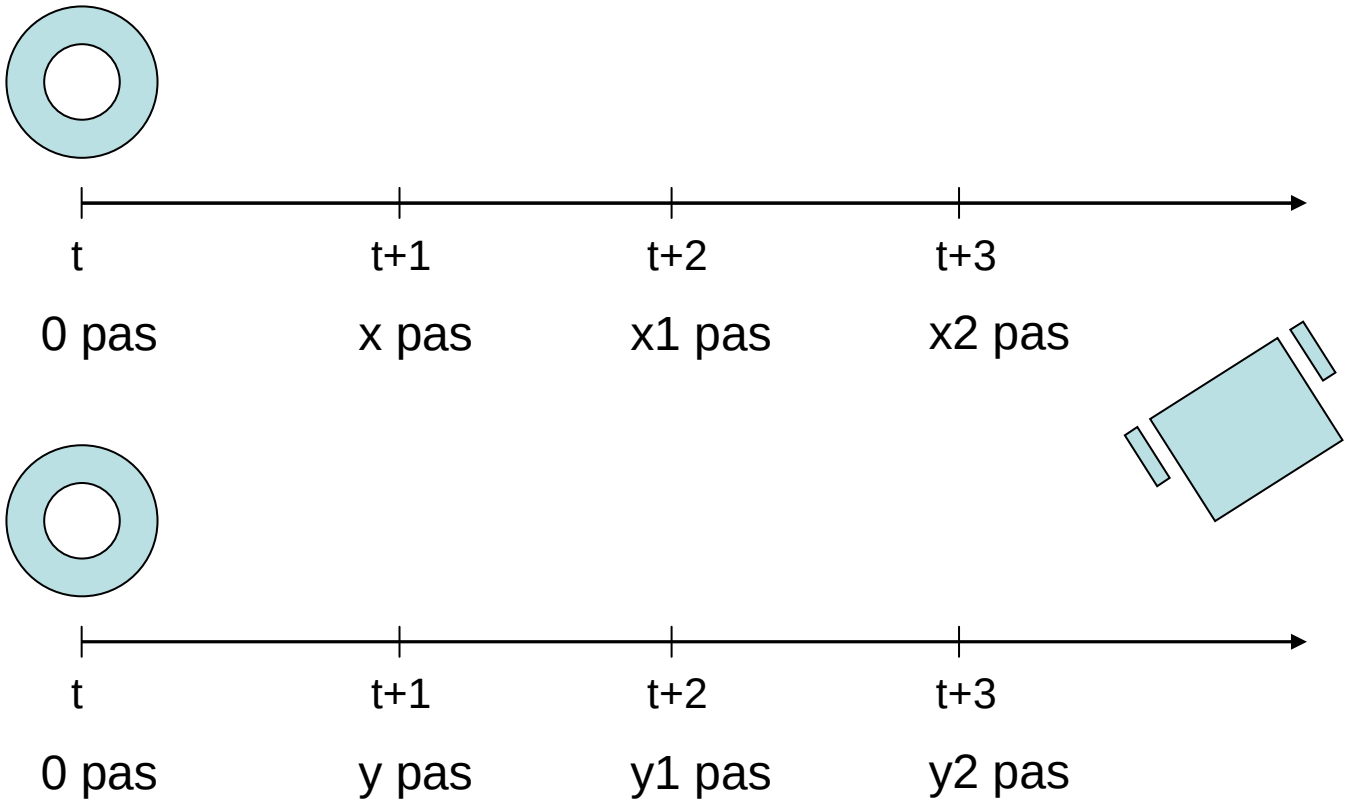


Figure 2-1 Harmonica System Block Diagram

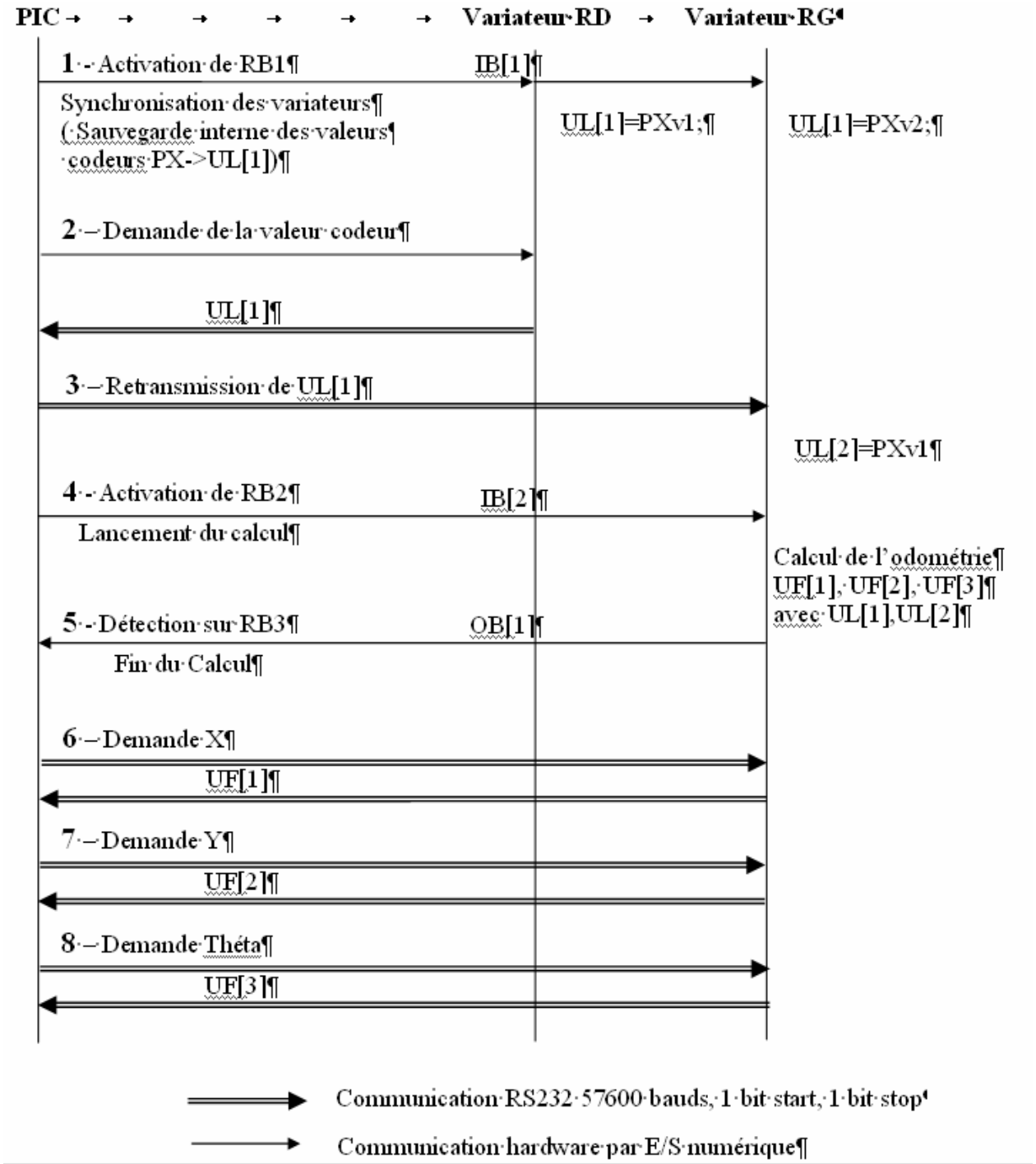
Calcul de l’odométrie

- À partir de 2 codeurs incrémentaux (12000 pas/tour)



- Réglage du pas en fct vitesse et des codeurs
- Dépendant du diamètre des roues, imprécision, local

Variateur Harmonica (4)



Variateur Harmonica (5)

```
// input 1
// Pour la synchronisation de l'odométrie
// => sauvegarde de PX
##AUTO_I1
                UI[SAVE_PX]=PX;
return

// input 2
// Arret d'urgence, on arrete les moteurs
##AUTO_I2
JV=0; BG; MO=0;
return

// input 3
// Test : marche avant
##AUTO_I3
MO=1; JV=200; BG;
return

// input 4
// Do Calcul
##AUTO_I4
OB[1]=0; Compute(); OB[1]=1;
return
```

```
##start
##AUTOEXEC

UF[NB_PAS_CODEUR]=4096.0;
UF[DIAMETRE_ROUEG]=0.30;
UF[DIAMETRE_ROUED]=0.30;
UF[DEMI_ENTRAXE]=0.30;
run();

function Compute()
                float deltatheta;
                float drg,drd,dr2;
                float dpasg,dpasd;
                drg=UI[SAVE_PX]-UI[OLD_PX1];
                drd=UI[VALUE_PX2]-UI[OLD_PX2];
                dpasg=(UF[DIAMETRE_ROUEG]*3.1457)/UF[NB_PAS_
                dpasd=(UF[DIAMETRE_ROUED]*3.1457)/UF[NB_PAS_
                drg=drg*dpasg;
                drd=drd*dpasd;
                dr2=(drg+drd)/2.0;
                deltatheta=(drg-drd) / (2.0*UF[DEMI_ENTRAXE]);
                UF[ORIENT]=UF[ORIENT]+deltatheta;
                UF[XCART]=UF[XCART]+dr2*sin(UF[ORIENT]);
                UF[YCART]=UF[YCART]+dr2*cos(UF[ORIENT]);
                UI[OLD_PX1]=UI[SAVE_PX];
                UI[OLD_PX2]=UI[VALUE_PX2];
return
```

Motorisation

Protocole PC-PIC

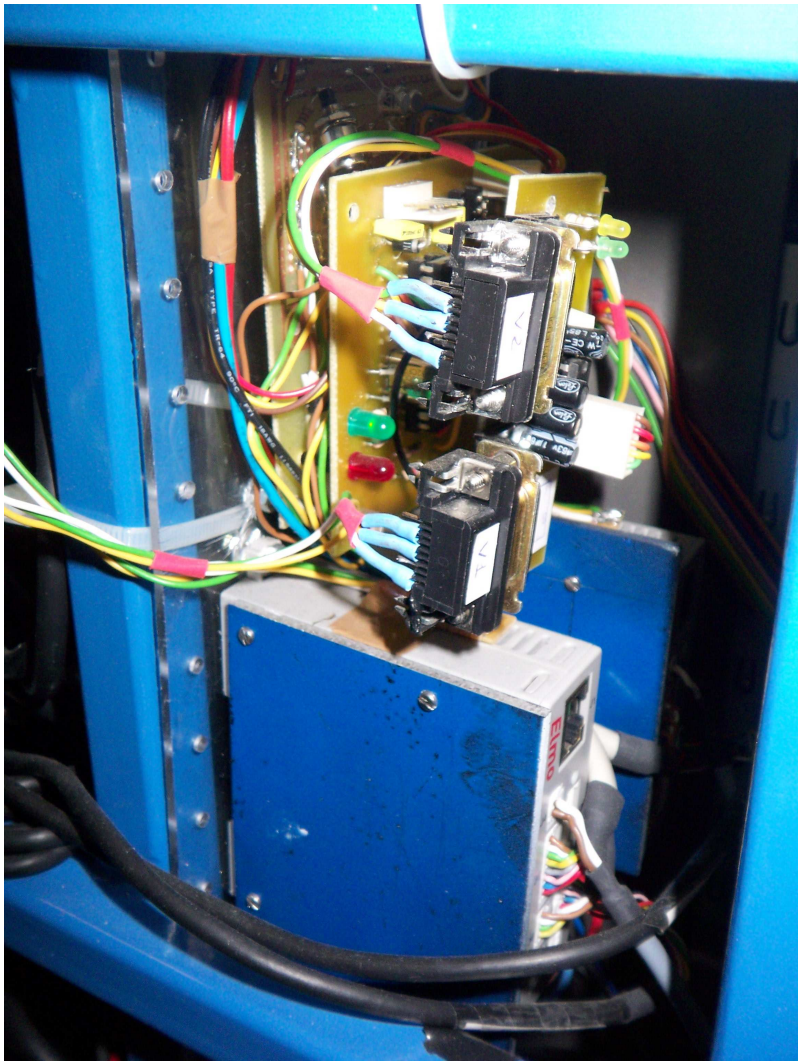
Commande¤	Entete·¤	ltrame¤	ldata¤	data¤	contenu¤	Equivalent· Harmo¤
SetVitesses¤	0x50·0x05¤	7¤	5¤	0xhh·0xll¶ 0xhh·0xll¶ 0xss¤	V·variateur·1¶ V·variateur·2¶ Sens·(b0·et· b1)¤	JV=hhll°;¶ JV=hhll°;¶ ¤
Allumage· moteurs¤	0x51·0x00¤	2¤	-¤	¤	¤	MO=1°;¤
Go·moteurs¤	0x52·0x00¤	2¤	-¤	¤	¤	BG°;¤
Arret·moteurs¤	0x53·0x00¤	2¤	-¤	¤	¤	MO=0°;¤
GetX¤	0x54·0x00¤	2¤	-¤	¤	¤	UF[1]°;¤
GetY¤	0x55·0x00¤	2¤	-¤	¤	¤	UF[2]°;¤
GetThéta¤	0x56·0x00¤	2¤	-¤	¤	¤	UF[3]°;¤
Stop·moteurs¤	0x57·0x00¤	2¤	-¤	¤	¤	ST°;¤
Lecture·RX·V2¤	0x58·0x00¤	2¤	-¤	¤	¤	¤
Reset·Variables·¤	0x59·0x00¤	2¤	-¤	¤	¤	¤
Acces·variables¤	0x60·0xaa¤	2+aa¤	aa¤	NumV·+· chaine¤	¤	¤

¶

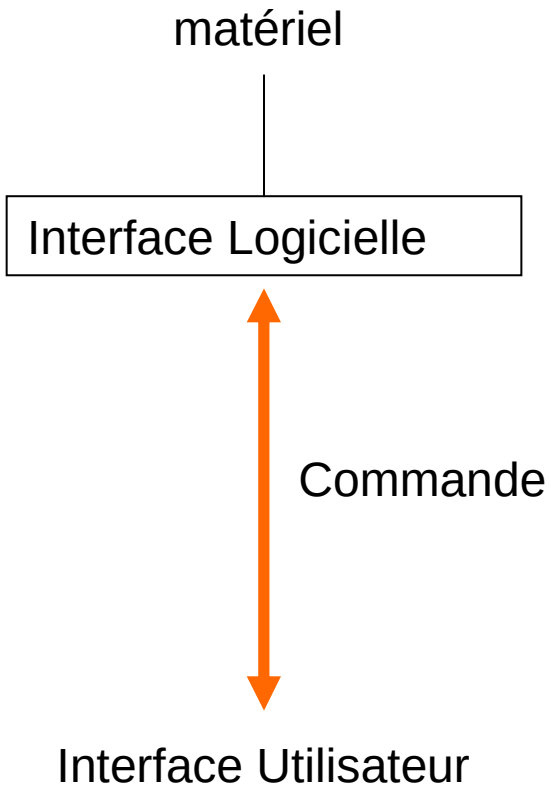
Commande¤	ID¤	Réponse¤
SetVitesses¤	SV¤	0x50·0x00¤
Allumage· moteurs¤	OnM¤	0x51·0x00¤
Go·moteurs¤	GM¤	0x52·0x00¤
Arret·moteurs¤	OfM¤	0x53·0x00¤
GetX¤	GX¤	0x54·taille·Flottant·ascii·terminé·par·un·“°;”···ex°:·«°12.300°;°»¤
GetY¤	GY¤	0x55·taille·Flottant·ascii·terminé·par·un·“°;”···ex°:·«°12.300°;°»¤
GetThéta¤	GT¤	0x56·taille·Flottant·ascii·terminé·par·un·“°;”···ex°:·«°12.300°;°»¤
Stop·moteurs¤	SM¤	0x57·0x00¤

Motorisation

Implantation

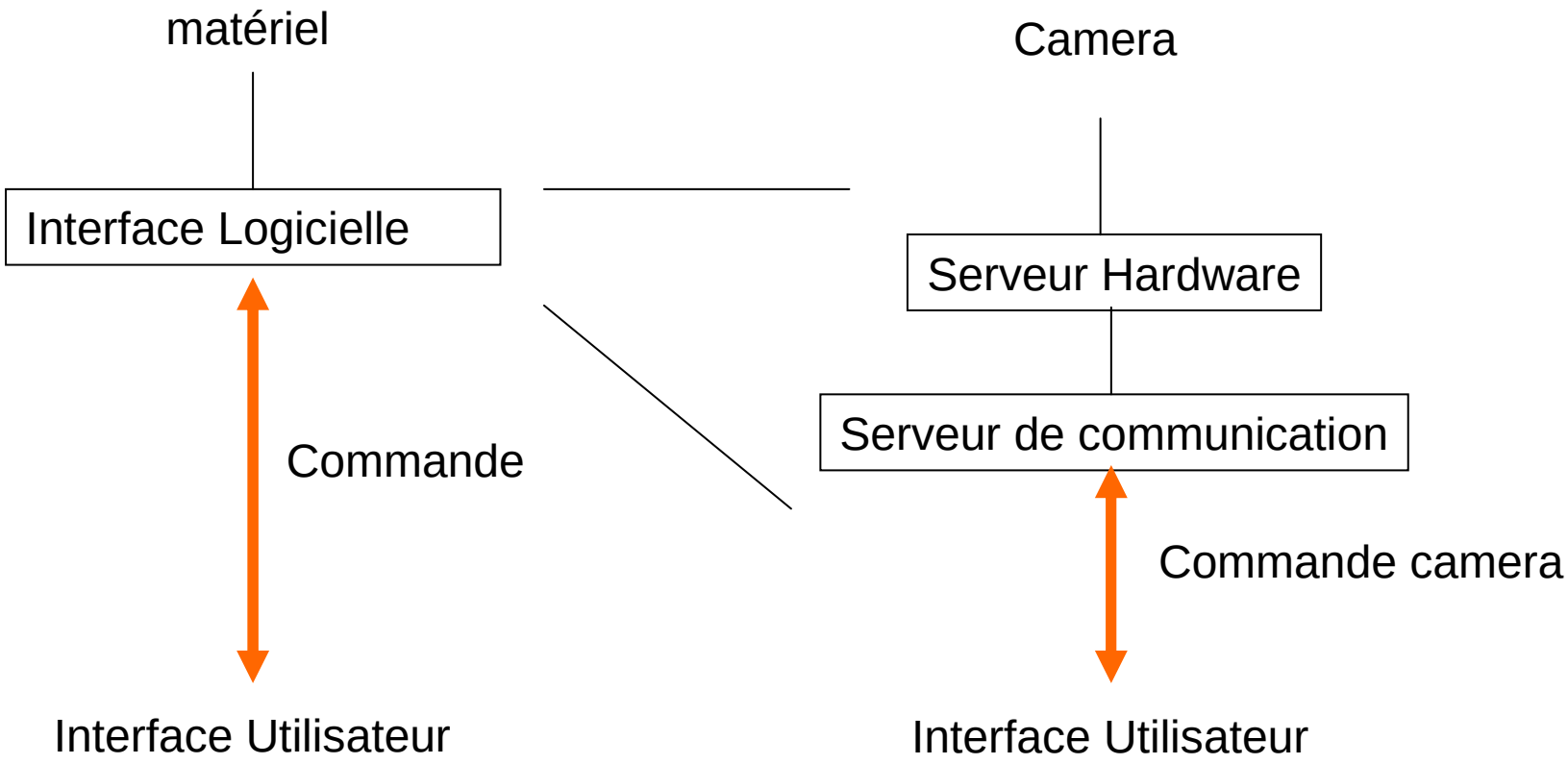


Les objectifs du « logiciel »

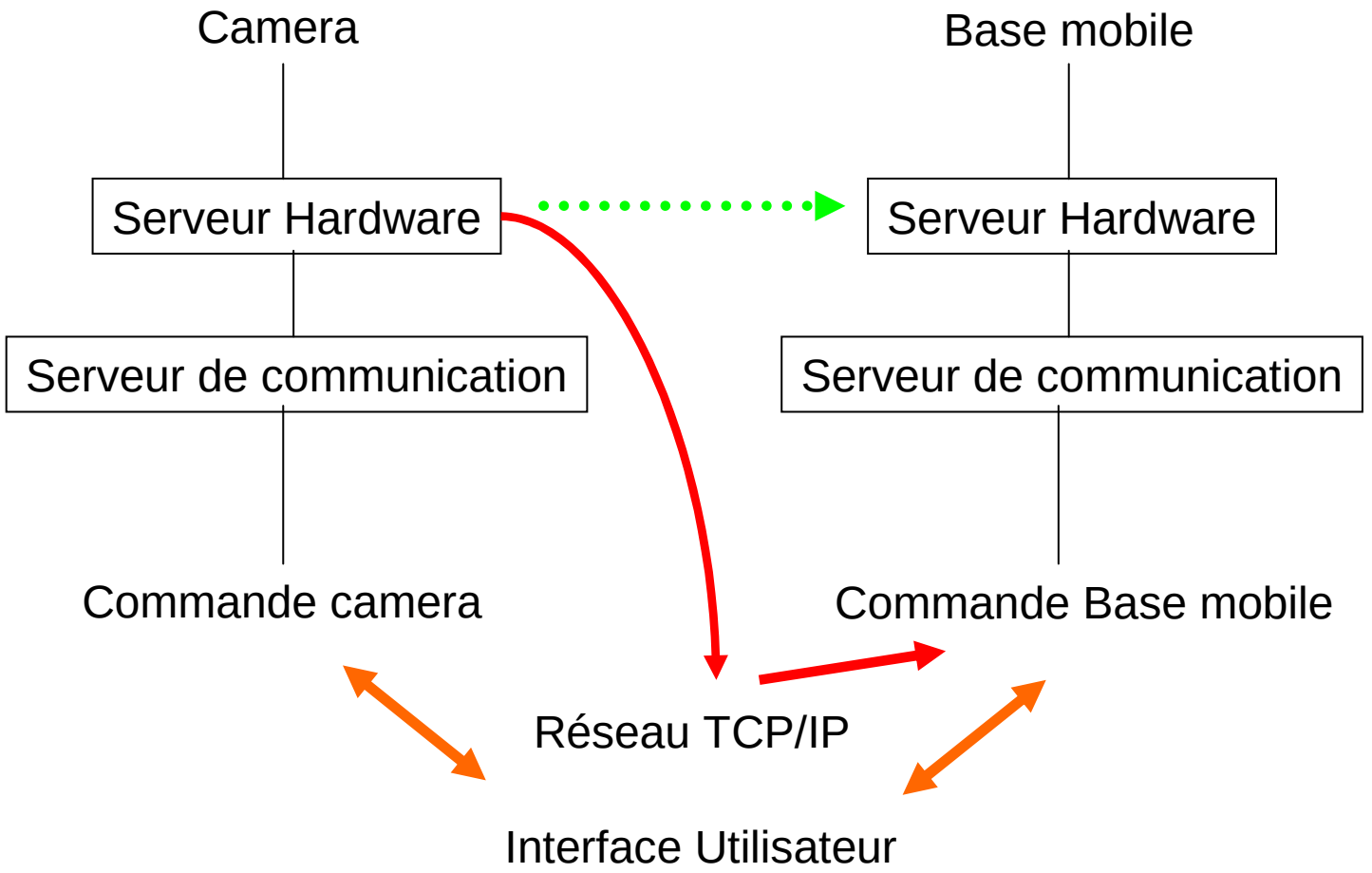


- Gestion du hardware
- Communication
- Echanges inter-modules
- Maintenance
- Evolutivité
- Robustesse

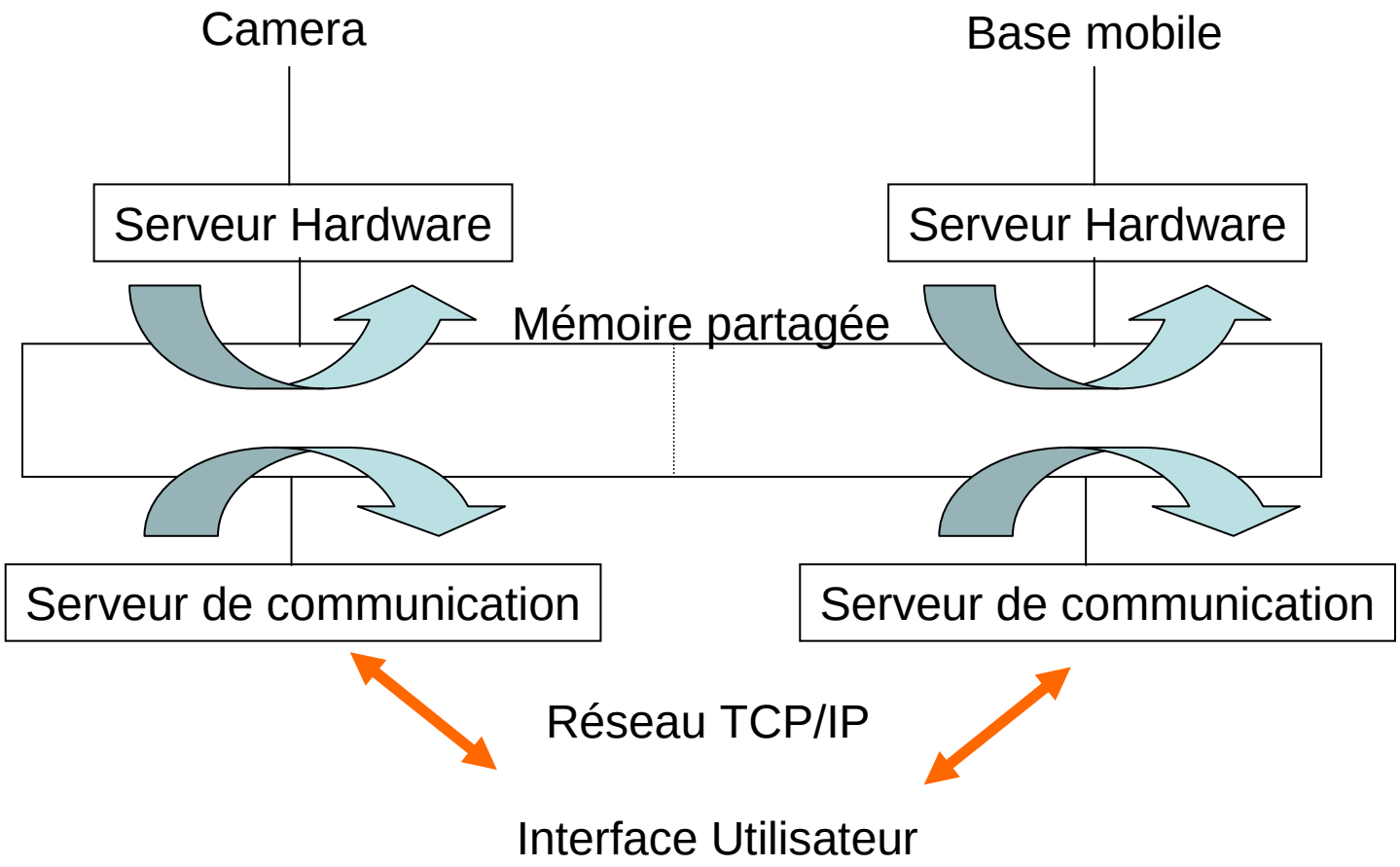
Séparation Hardware/Comm



Echanges intra modules



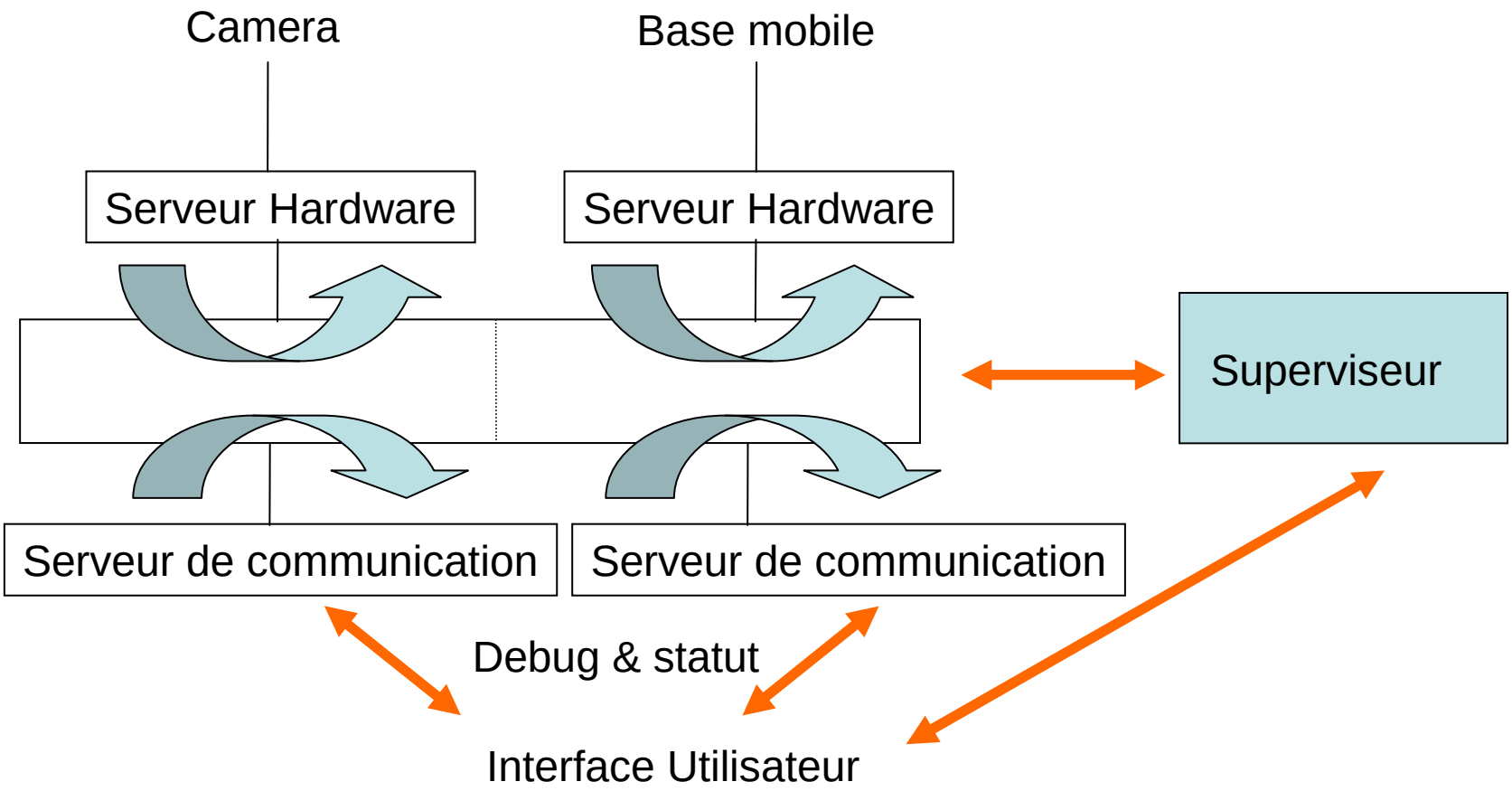
Echanges inter-modules



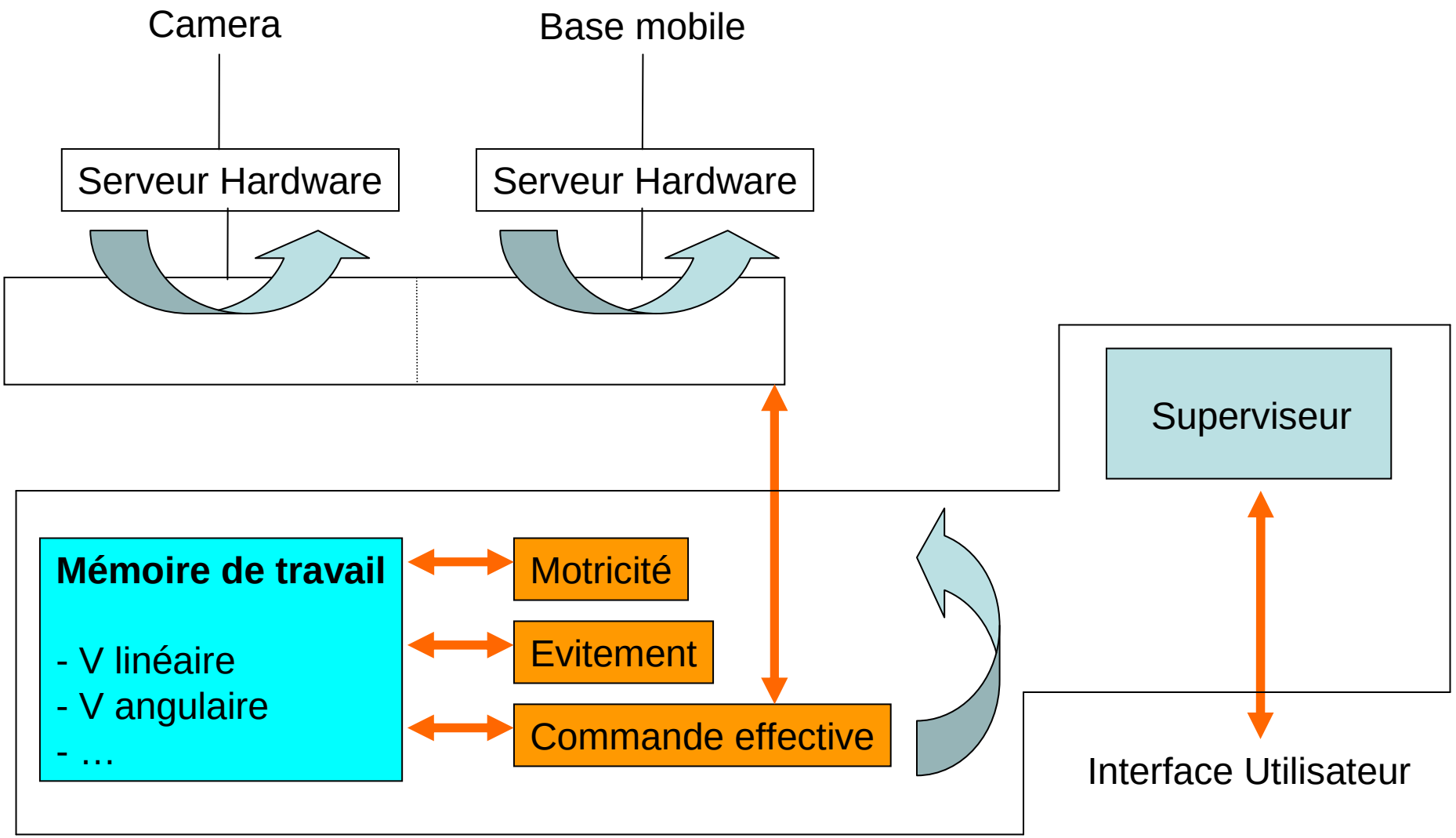
- Gérer les accès concurrentiels
- Optimiser les accès

Supervision

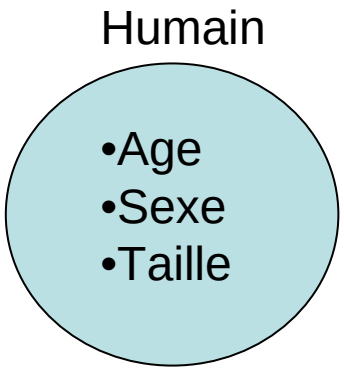
Qui Fait Quoi ?

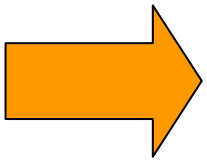


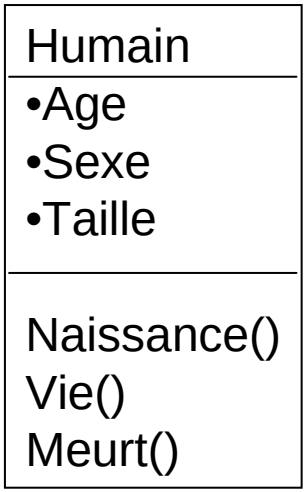
Modules supervision



Programmation objet (C++)







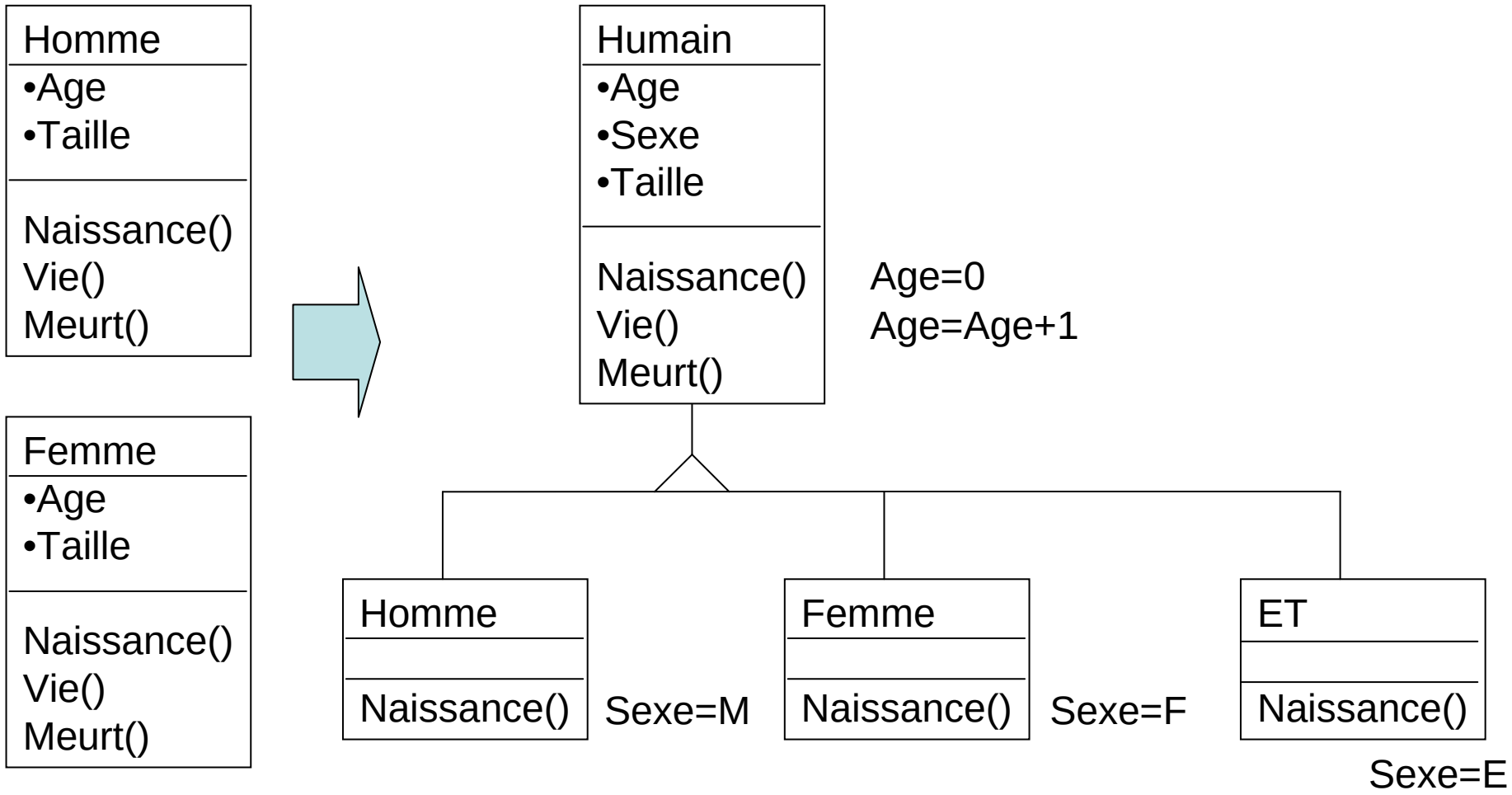
Naissance()
Vie()
Meurt()

- Programmation modulaire
- Accesseurs
- Protection des données
- Lisibilité accrue
- Boite Noire

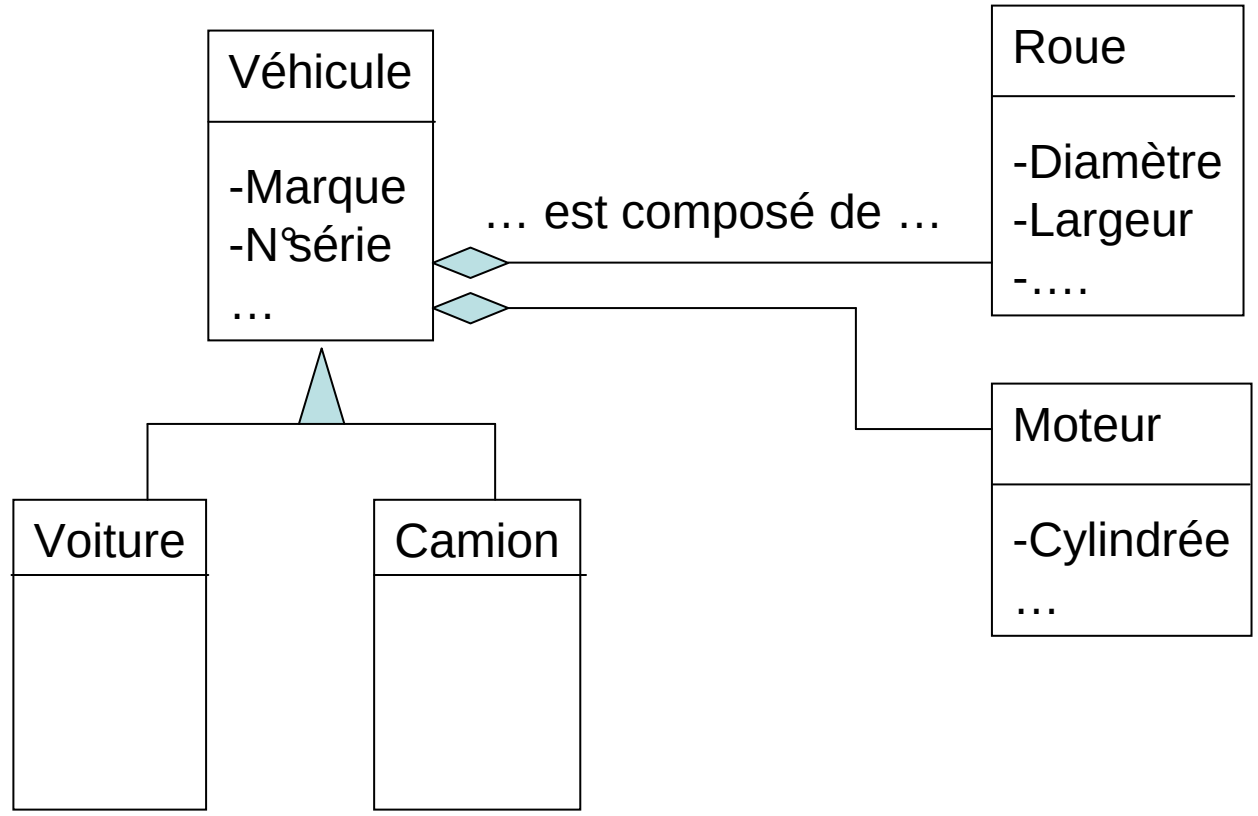
```
{
  Humain H
  Naissance(H)
  Vie(H)
  H.Sexe=F
  Meurt(H)
}
```

```
{
  Humain H
  H.Naissance()
  H.Vie()
  H.Sexe=F //Interdit
  H.Meurt()
}
```

Héritage

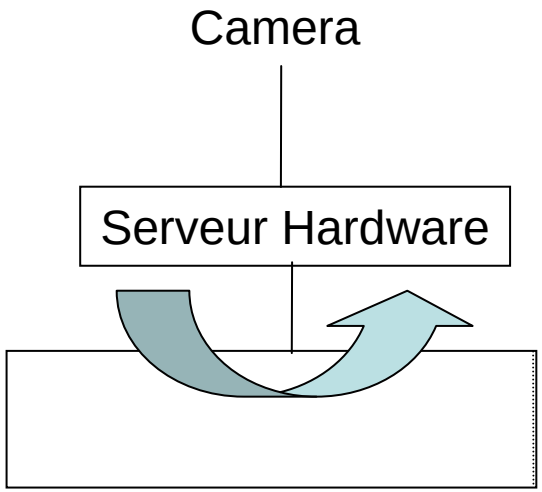

Homme
•Age
•Taille
Naissance()
Vie()
Meurt()
Femme
•Age
•Taille
Naissance()
Vie()
Meurt()
Humain
•Age
•Sexe
•Taille
Naissance()
Vie()
Meurt()
Age=0
Age=Age+1
Homme
Naissance()
Sexe=M
Femme
Naissance()
Sexe=F
ET
Naissance()
Sexe=E

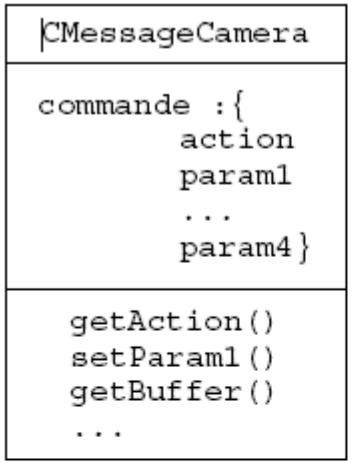
Héritage et composition

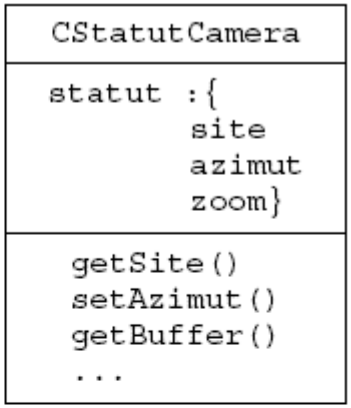


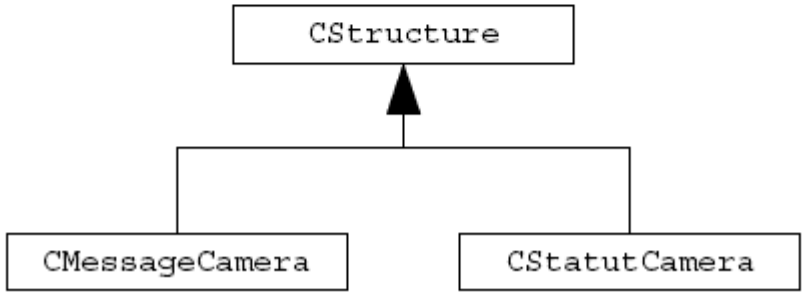
…. est une sorte de ……

Structures **Message** et **Statut** (Ex : Caméra)

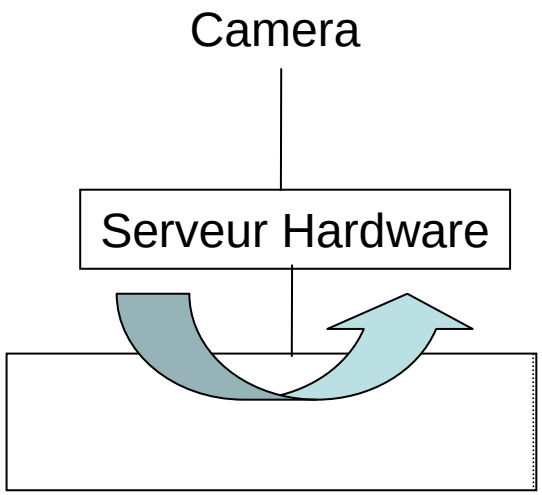


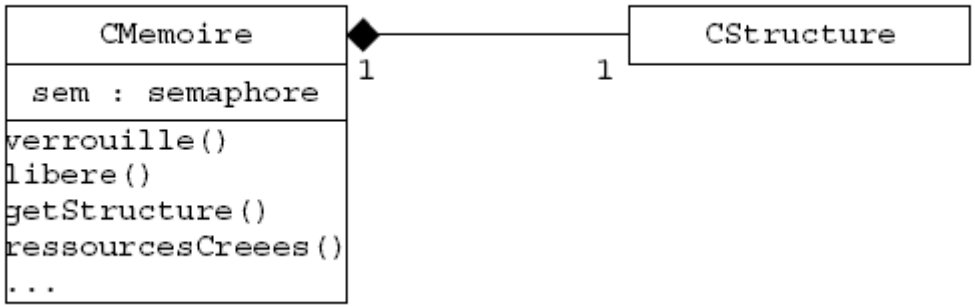


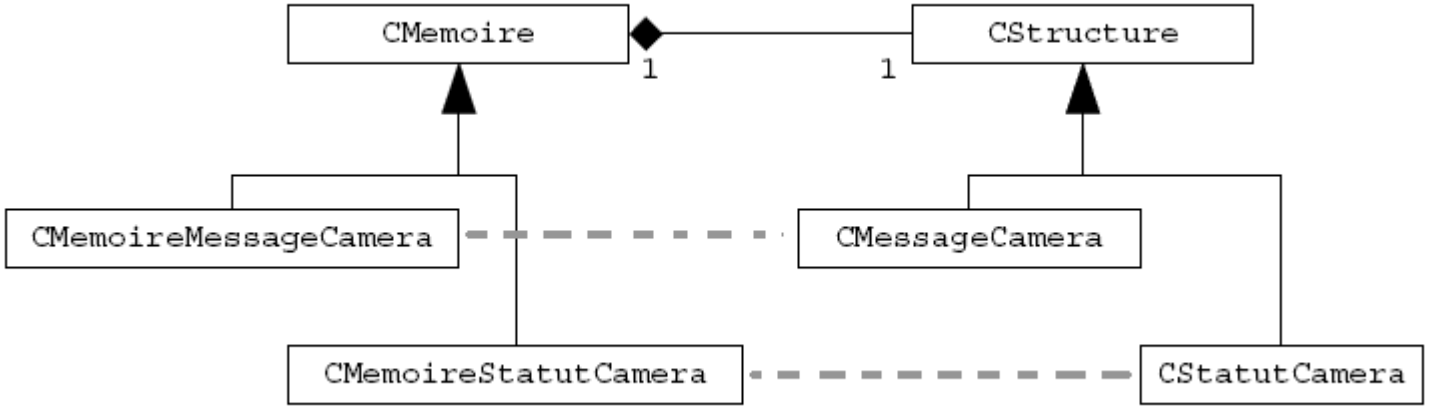




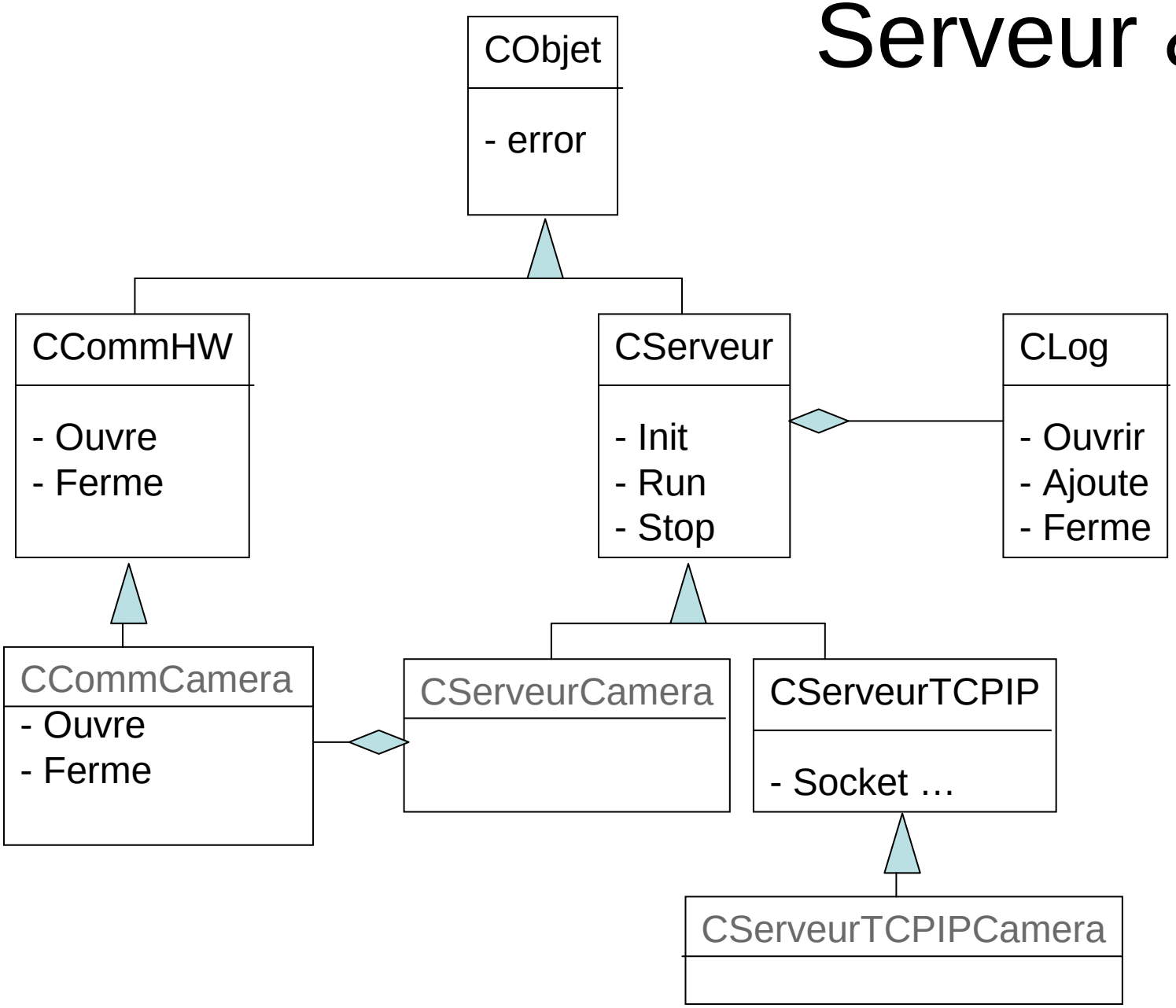
Mémoire partagée



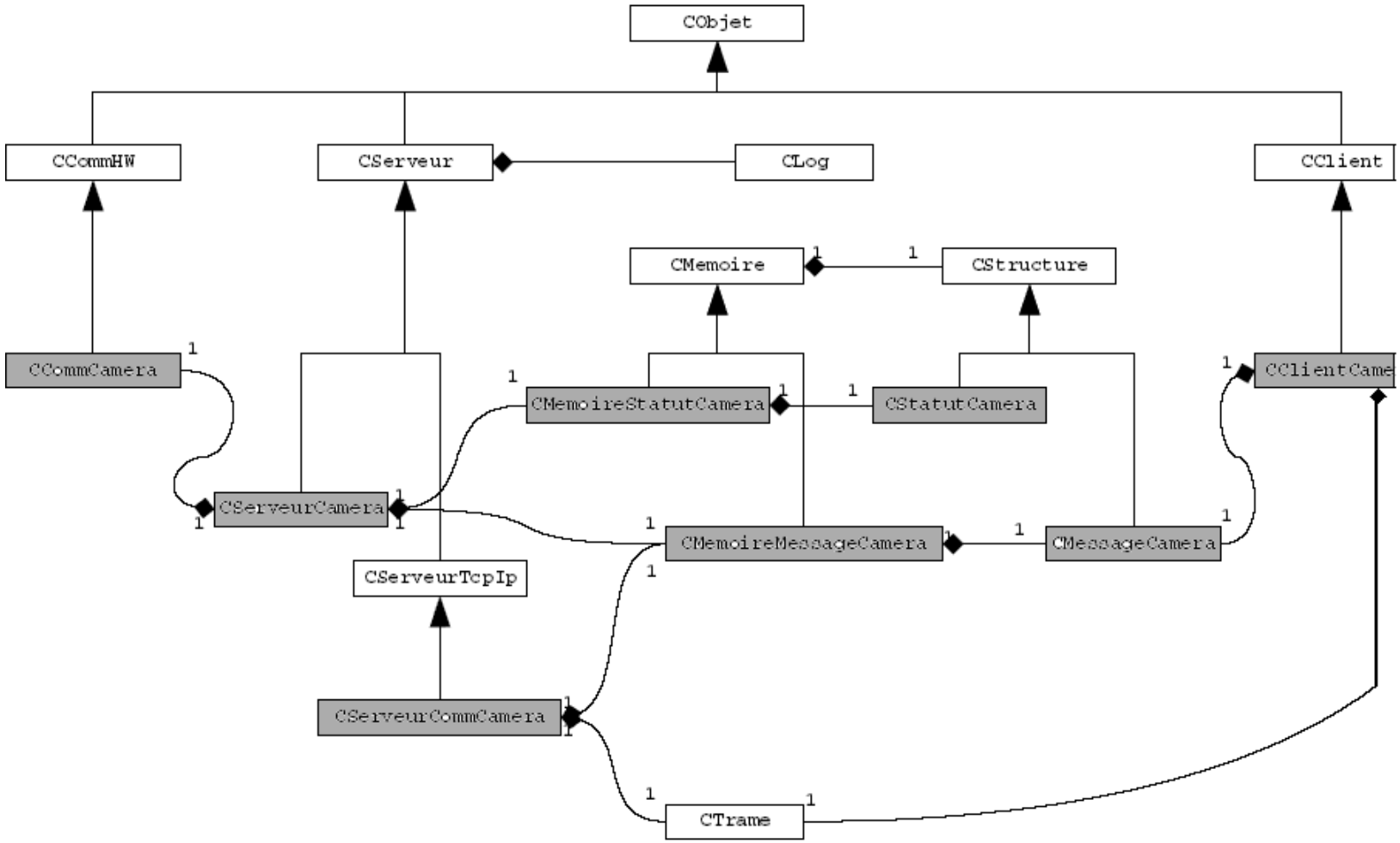




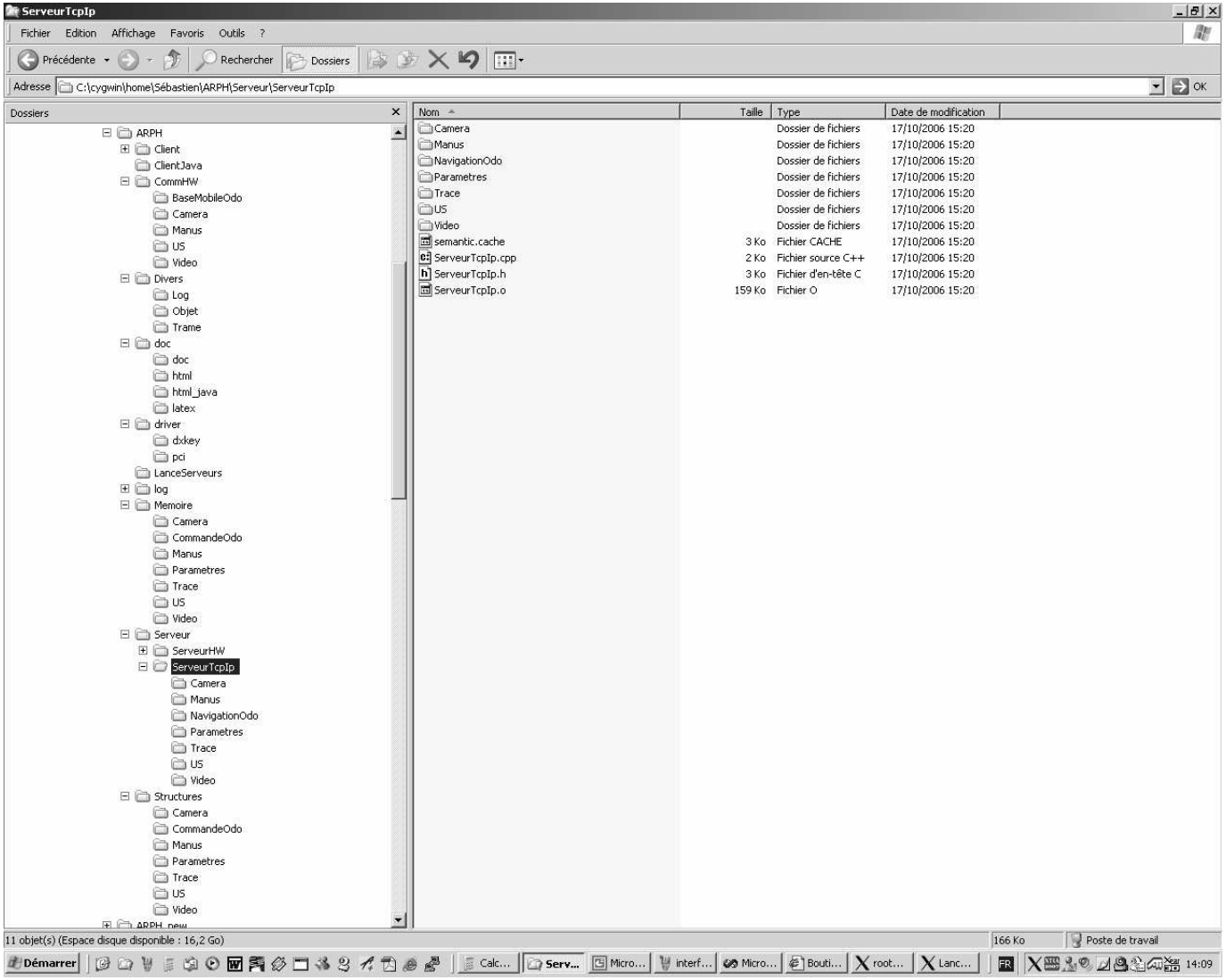
Serveur & co


CObjet
- error
CCommHW
- Ouvre
- Ferme
CServeur
- Init
- Run
- Stop
CLog
- Ouvrir
- Ajoute
- Ferme
CCommCamera
- Ouvre
- Ferme
CServeurCamera
CServeurTCPIP
- Socket …
CServeurTCPIPCamera

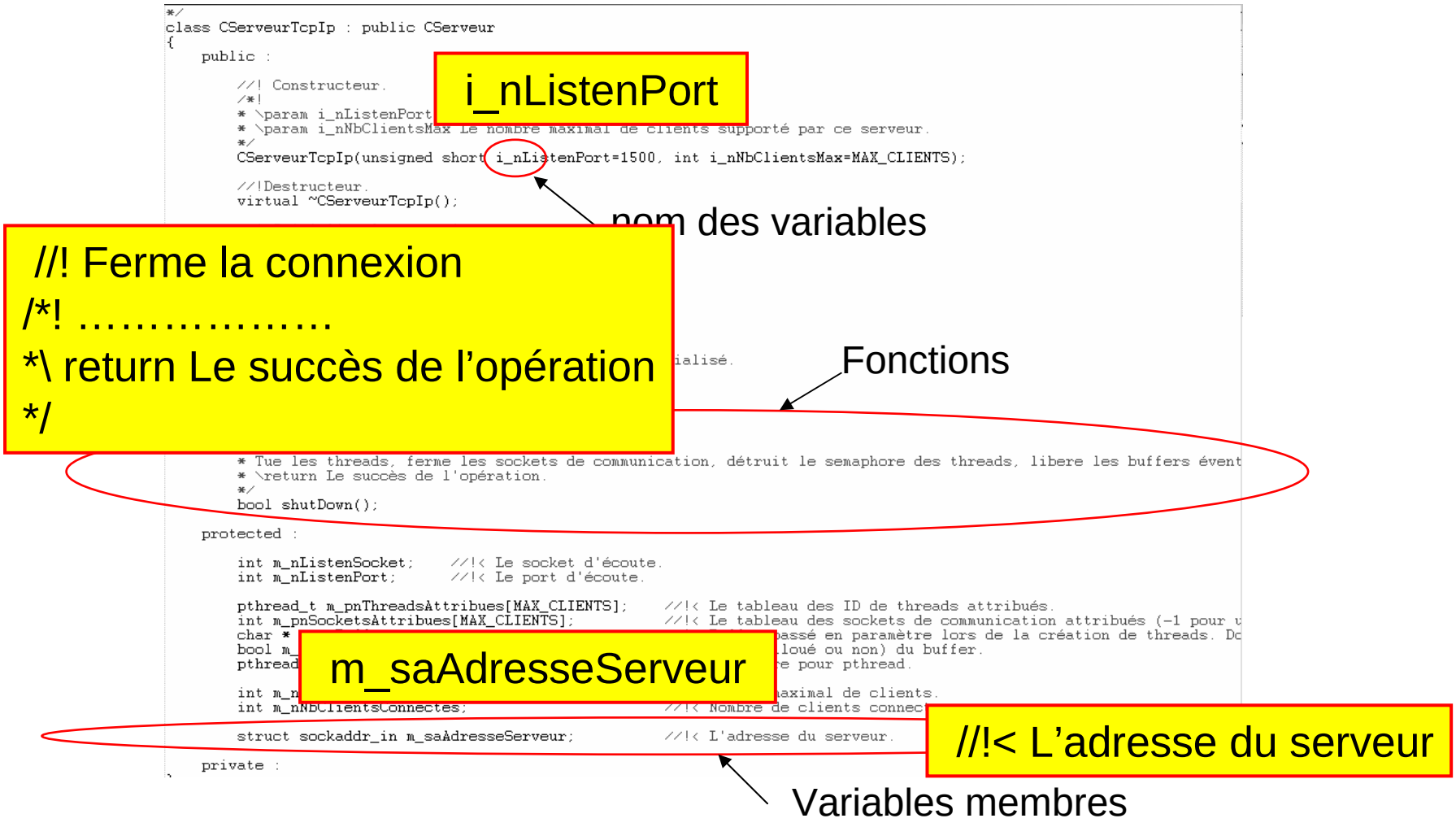
Globalement ….



Organisation des fichiers

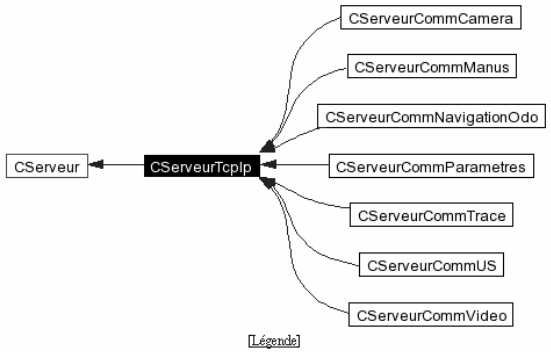


Normes de codage

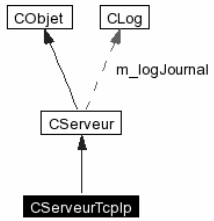


Documentation automatique (DOxygen)

Référence de la classe CServeurTcpIp



[Légende]



[Légende]

Membres publics

	CServeurTcpIp (unsigned short i_nListenPort=1500, int i_nNbClientsMax=**MAX_CLIENTS**) *Constructeur.*
virtual	**~CServeurTcpIp** () *Destructeur.*
bool	**init** () *Initialise la connexion.*
virtual bool	**run** ()=0 *Boucle principale.*
bool	**shutDown** () *Ferme la connexion.*

Attributs Protégés

int	**m_nListenSocket** *Le socket d'écoute.*
int	**m_nListenPort** *Le port d'écoute.*
pthread_t	**m_pnThreadsAttribues** [MAX_CLIENTS] *Le tableau des ID de threads attribués.*
int	**m_pnSocketsAttribues** [MAX_CLIENTS] *Le tableau des sockets de communication attribués (-1 pour une case vide).*
char *	**m_pcBuffer** *Buffer passé en paramètre lors de la création de threads. Doit être alloué par run() et désalloué dans shutDown().*
bool	**m_bBufferAlloue** *Etat (alloué ou non) du buffer.*
pthread_mutex_t	**m_mutMutex** *Semaphore pour pthread.*
int	**m_nNbClientsMax** *Nombre maximal de clients.*
int	**m_nNbClientsConnectes** *Nombre de clients connectes.*
sockaddr_in	**m_saAdresseServeur** *L'adresse du serveur.*

Documentation automatique

Documentation des méthodes

bool CServeurTcpIp::init( ) [virtual]

Initialise la connexion.

Paramètre l'écoute des clients.

Renvoie:
Le succès de l'opération.

Implémente **CServeur**.

Redéfinie dans **CServeurCommCamera**, **CServeurCommManus**, **CServeurCommNavigationOdo**, **CServeurCommParametres**, **CServeurCommTrace**, **CServeurCommUS**, et **CServeurCommVideo**.

Voici le graphique d'appel pour cette fonction:

virtual bool CServeurTcpIp::run( ) [pure virtual]

Boucle principale.

Ne fait rien. A surcharger dans un serveur spécialisé.

Renvoie:
true

Implémente **CServeur**.

Implémenté dans **CServeurCommCamera**, **CServeurCommManus**, **CServeurCommNavigationOdo**, **CServeurCommParametres**, **CServeurCommTrace**, **CServeurCommUS**, et **CServeurCommVideo**.

bool CServeurTcpIp::shutDown( ) [virtual]

Ferme la connexion.

Tue les threads, ferme les sockets de communication, détruit le semaphore des threads, libere les buffers éventuellement alloués et ferme le socket.

Renvoie:
Le succès de l'opération.

Implémente **CServeur**.

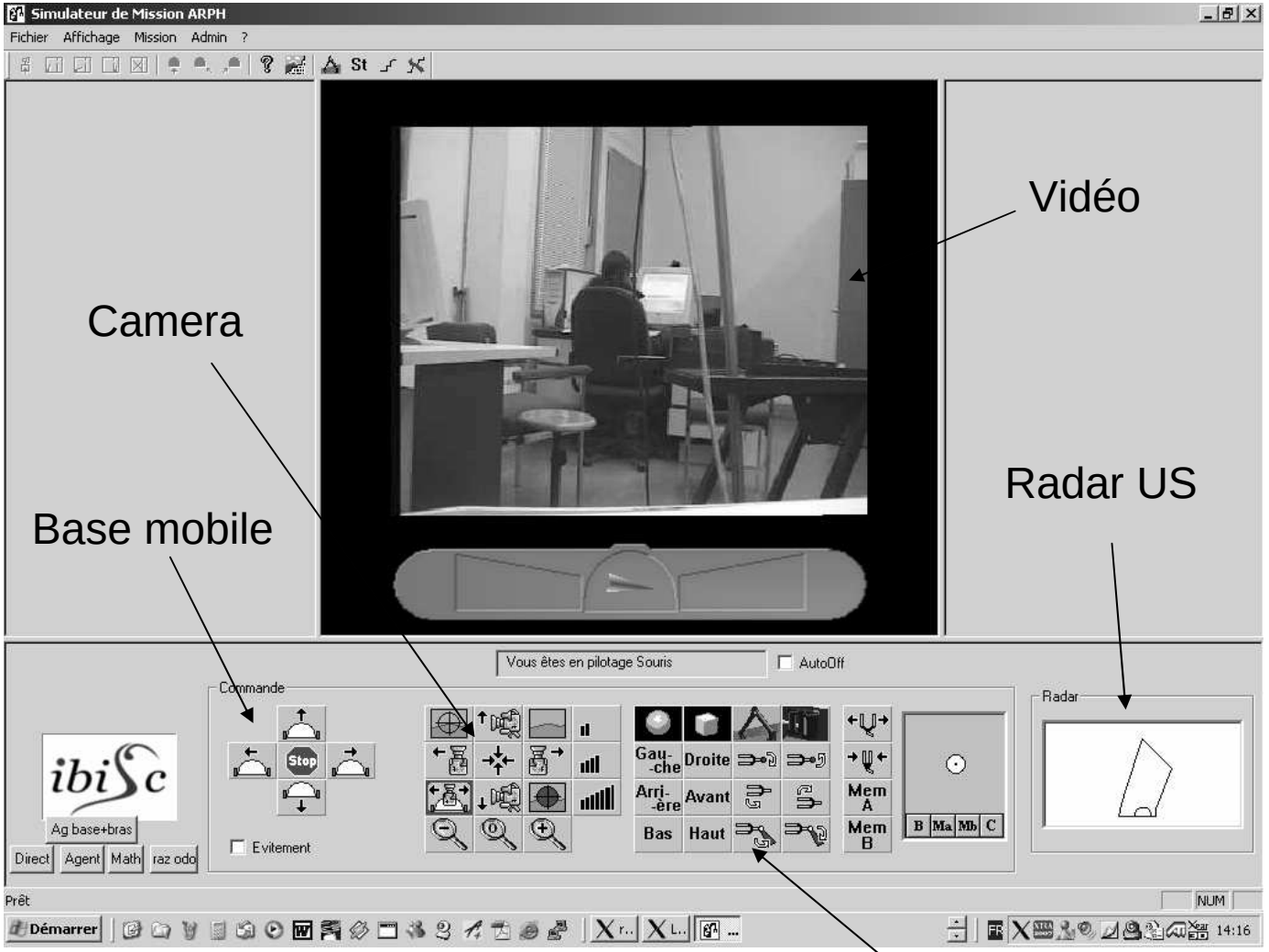
Voici le graphique d'appel pour cette fonction:

Lancement des serveurs

Serveurs Materiels

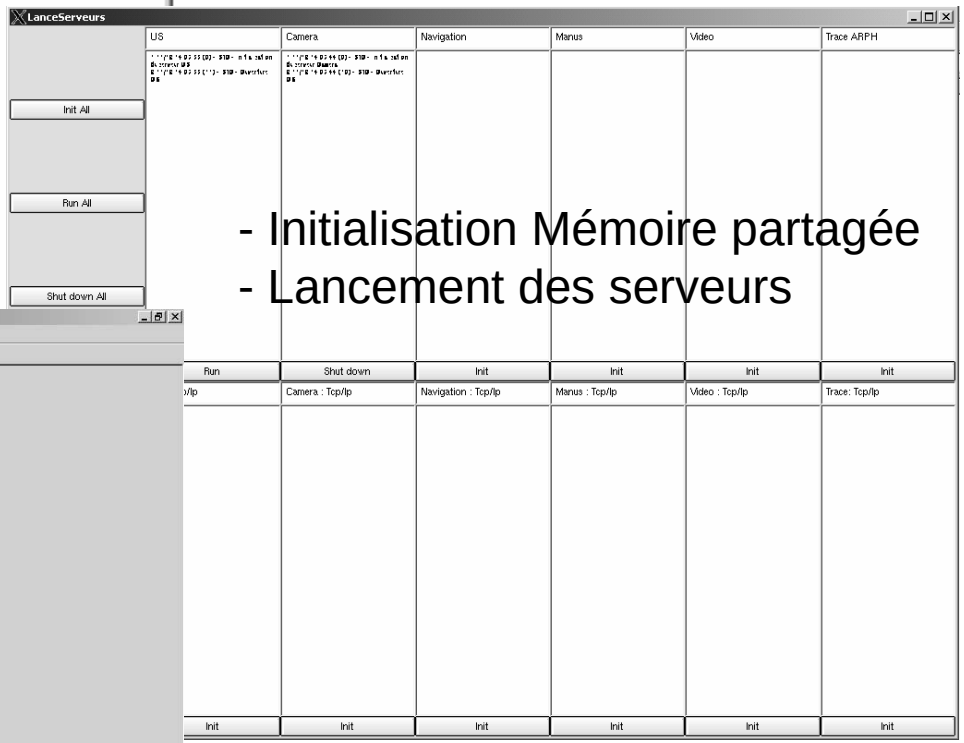
Serveurs Logiciels

Interface Utilisateur



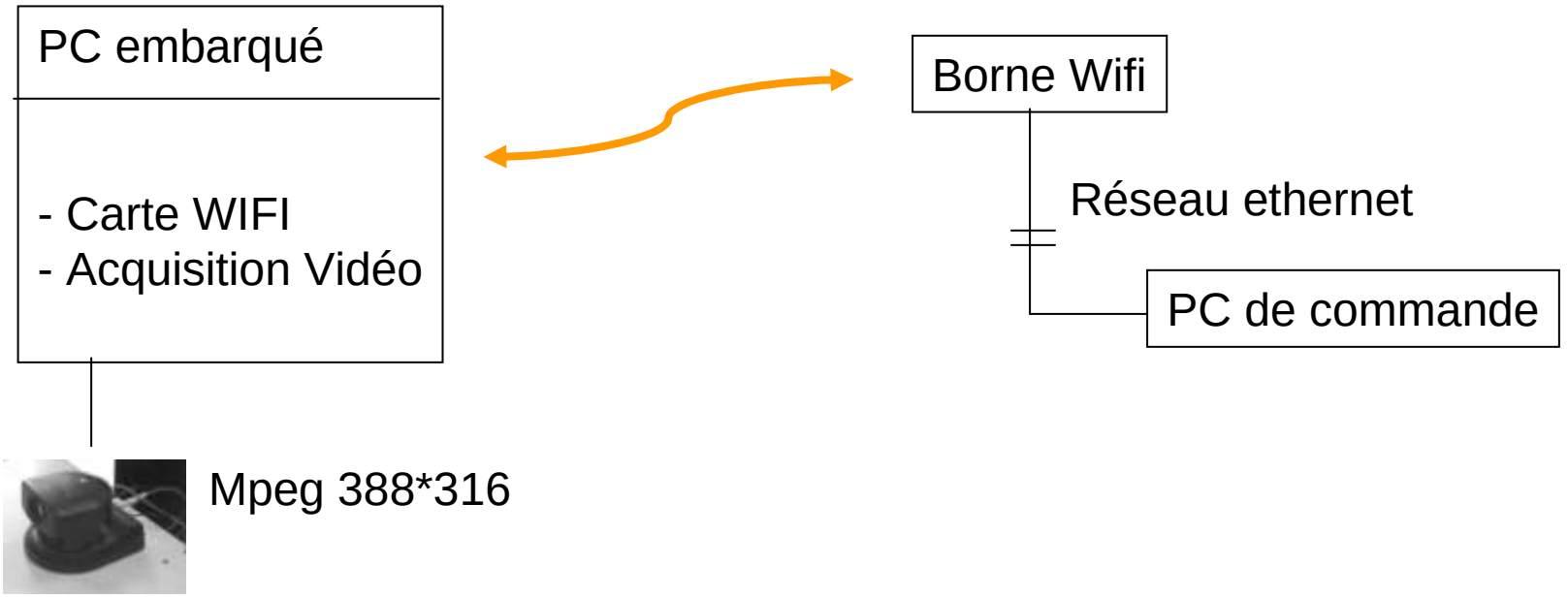
Lancement général

-Initialisation des drivers



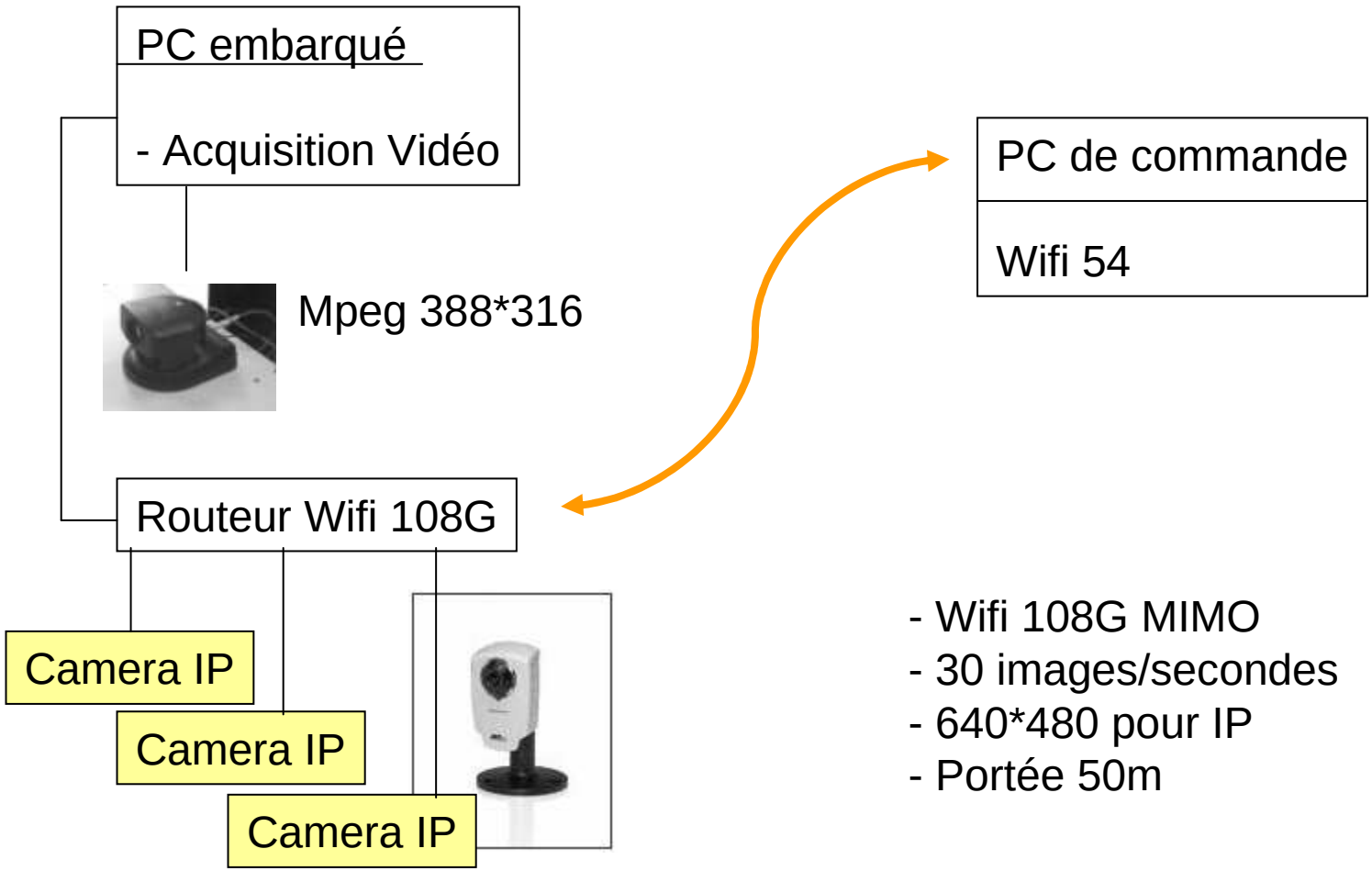
- Initialisation Mémoire partagée
- Lancement des serveurs

Architecture matérielle



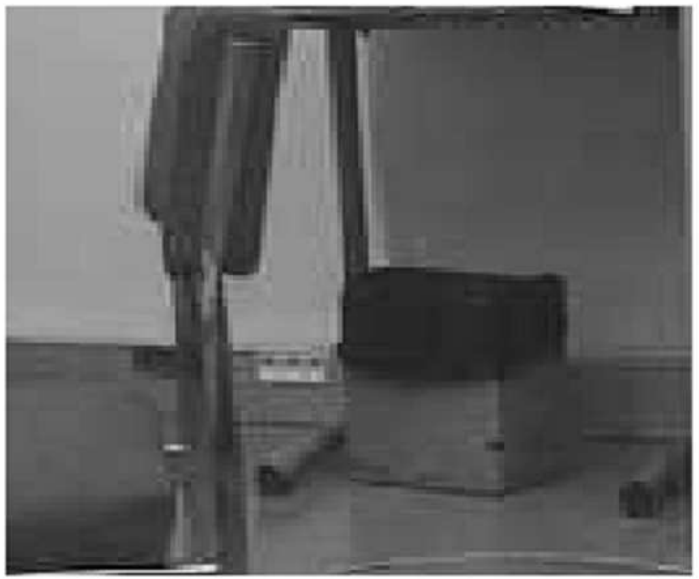
- Wifi 11Mb/s
- 10 images/secondes
- Portée 10m

Evolutions



- Wifi 108G MIMO
- 30 images/secondes
- 640*480 pour IP
- Portée 50m

Comparatifs Vidéo



Question?