Rapport sur l'épreuve d'admissibilité
« Automatique et Informatique Industrielle »

Épreuve de 6 heures.

Cette année, l'épreuve d'automatique et d'informatique industrielle explorait quelques champs spécifiques à l'Informatique Industrielle, à savoir les réseaux, la modélisation UML ainsi que la programmation orientée objet avec pour langage support le C++. Cette évolution était inévitable, tant il est fréquent de rencontrer de jeunes agrégés de Génie Electrique affectés dans des sections de STS IRIS ou de DUT de GEII.

L'épreuve comportait donc une partie à forte connotation Informatique Industrielle (réseau + UML et C++ associé) et une partie d'asservissement linéaire ne comportant pas de difficulté particulière si ce n'est sa longueur. Compte tenu des nouveautés, le barème de notation de l'épreuve écrite a été conçu de façon à limiter leur incidence sur la note finale.

1ère partie : étude du réseau informatique

Cette partie a été abordée par environ deux tiers des candidats.

Si les concepts « grand public » relatifs aux réseaux (adresses et masque) ont été généralement bien traités, ceux plus spécifiques (notions de client et serveur, mode connecté ou non et analogies humaines correspondantes, choix des numéros de ports) ont franchement laissé à désirer.

2nde partie : modélisation UML

La modélisation UML avait volontairement été limitée à deux diagrammes :

- Cas d'Utilisation : bornage du système, outil de communication avec les intervenants non informaticiens d'un projet informatique…
- Classes : mise en évidence des héritages et génération automatique du squelette du code.

Quelques candidats ont très bien traité cette partie, n'hésitant pas à commenter le sujet, mais la très grande majorité est passée largement à côté.

Le diagramme des cas d'utilisation est quasi inconnu, et la modélisation graphique du système est censée expliquer de façon simple un système complexe ; il a trop souvent été constaté des envolées graphiques plus proches de l'art moderne que de l'UML.

Le diagramme de classes semble mieux connu, sans doute parce que la notion de classe l'est elle aussi, mais il convient de faire de grosses réserves : si les notions C++ « classiques » d'utilité d'une classe, d'instanciation et de classes filles sont connues par 13% des candidats, celles plus « fines », comme le polymorphisme ne le sont que par 4%. Les notions UML associées à l'héritage, spécialisation et généralisation, ne sont elles connues que par 2% des candidats.

Cette méconnaissance de la programmation orientée objet est regrettable quand on sait que la majorité des logiciels créés actuellement en entreprise utilisent des langages orientés objet dans le but avoué d'une réutilisation maximale du code. Il est donc indispensable que les agrégés de Génie Electrique à venir maîtrisent les outils du Génie Logiciel, d'une part pour générer automatiquement les squelettes des applications, d'autre part pour éviter les codes anarchiques et/ou redondants.

Il existe de très nombreux ouvrages ainsi que d'excellents sites qui permettront aux futurs candidats de mieux appréhender l'UML ainsi que la programmation orientée objet.

3ème partie : Asservissements Linéaires

Les asservissements linéaires étaient eux très classiques ; les cinq parties étaient :

- Modélisation ; cette partie a été correctement traitée dans l'ensemble.
- Régulation de vitesse dans le domaine linéaire ; idem, si ce n'est le placement des correcteurs (on ne demandait pourtant que des PI), le tracé des fonctions de transfert en boucle fermée.
- Limitation de couple, toujours dans le domaine linéaire ; si la partie relative à la machine à courant continu est correcte, les questions relatives à la modélisation ainsi qu'au placement des correcteurs PI ont posé plus de problèmes et n'ont été traitées que par environ 11% des candidats.
- Réalisation d'un estimateur de couple utilisant les équations d'état ; environ un tiers des candidats s'est intéressé à cette partie. Les notions fondamentales (unicité de la matrice d'état, gouvernabilité et observabilité) laissent encore beaucoup à désirer ; moins de 10 % des candidats y ont répondu correctement.
- Régulation numérique de vitesse ; cette dernière partie a très peu été traitée alors qu'elle posait bon nombre de questions de cours (définition de z, fonction de transfert d'un échantillonneur bloqueur d'ordre 0, condition de stabilité d'un système bouclé échantillonné …). Il est dommage de perdre des points sur de telles questions.

Remarques générales

Certains candidats n'ont traité que les parties réseau et UML et n'ont pas répondu à la partie automatique, ce qui traduit une méconnaissance grave des principes mêmes de l'automatique. Cet état de fait découle sans doute de la formation des candidats, mais le jury attire leur attention sur l'omniprésence de l'automatique aussi bien dans les systèmes électroniques qu'électrotechniques. Une bonne connaissance de la programmation et des réseaux ne saurait dispenser des connaissances de base dans les matières fondamentales du Génie Electrique.

Étude du réseau.

II.A.1 : Classe B ; adresse réseau : 172.20.0.0 ; masque : 255.255.0.0

II.A.2 : Ordinateur de charge : client ; variateur de charge : serveur.
Ordinateur de mesures : serveur ; système d'acquisition : client.

II.A.3 : C'est le client qui prend l'initiative de la communication.

II.A.4 : Port variateur < 1024 : correspond à un service normalisé (ici ModbusTCP)
Les autres ports sont > 1024 : on fait ce que l'on veut.
Un port = un service.

II.A.5 : Pour les parties charge et mesure : couches 7, 4, 3, 2 et 1 dans le cas présent.

II.A.6 : Il faut modifier le masque de sous réseau sur l'ordinateur de charge qui devient : 255.255.255.128 (128 ou + pour le dernier chiffre).

II.B.1 : Un brin Ethernet : 100 m maxi => il faut un répéteur (hub) ou un commutateur (switch) au milieu. Aucune modification d'adresse ni de masque.

II.B.2 : Il faut deux ponts pour établir une liaison inter pont qui supporte la distance.

II.B.3 : Il faut un routeur entre l'ordinateur de mesures et Internet, car 172.20.X.Y est une adresse privée => l'ordinateur de mesures conserve ses réglages réseau ; par contre le routeur doit avoir une adresse IP valide.

II.C.1 : Nouveau port : 1817.

II.C.2 : cf. DR1 (réponses en rouge dans le document pdf).

II.C.3 : cf. DR1 (réponses en rouge dans le document pdf).

II.C.4 : Les trames Ethernet finissent déjà par un CRC ; le CRC Modbus serait inutilement redondant.

II.C.5 : TCP : mode connecté. La capture de la trame illustre le « handshaking », c à d le dialogue incessant entre l'ordinateur de charge et le variateur.

II.C.6 : Mode connecté : analogie = communication téléphonique
Mode non connecté : analogie = lettre à la poste.
Dans le premier cas, on est sûr de l'arrivée du message envoyé ; dans le second, non.

II.C.7 : en mode connecté, on utilise de la bande passante pour le handshaking => on risque d'avoir des délais lors de l'envoi de trame à cause du CSMA/CD.

En mode non connecté, on n'est pas sûr que le message soit bien arrivé. Il faudra installer une procédure d'écho ; l'émetteur, voyant revenir son message, saura qu'il est bien arrivé. Trafic limité au strict besoin.

III.A.1 : Borner le système ; permettre aux intervenants non informaticiens d'un projet de comprendre ce qui va être programmé, et ainsi pouvoir exprimer leurs besoins.

III.A.2 : Un cas d'utilisation est un service que l'on attend du système.

III.A.3 : cf. DR2 (réponses en rouge dans le document pdf).

III.A.4 : cf. DR2 (réponses en rouge dans le document pdf).

III.B.1.1 : Une classe regroupe des données (attributs) et les fonctions (méthodes) qui les manipulent (notion d'encapsulation).

III.B.1.2 : L'instanciation.

III.B.1.3 : le fichier .h définit les types et visibilités des attributs et méthodes, ainsi que les libraires à inclure.
Le fichier .cpp explicite le contenu des méthodes.

III.B.1.4 : C'est une directive de compilation ; elle permet d'inclure les types définis dans modbus.h ; elle est utilisée par le pré processeur.

III.B.1.5 : Ces trois lignes forment une directive de compilation conditionnelle, afin d'éviter de multiples définitions d'un même attribut ou de multiples inclusions d'une même librairie, par exemple.

III.B.1.6 : dans IHM.h, il faut rajouter `#include "Modbus.h"`.
dans IHM.cpp, il faut rajouter :
`Modbus ModbusObjet;` pour la création de l'objet ;
`ModbusObjet.adresseEsclave = 0x01;` pour l'attribution d'une valeur ;
`ModbusObjet.completerTrame();` pour l'utilisation d'une méthode ;

III.B.1.7 : Cette méthode est le constructeur de la classe ; elle sert à initialiser les attributs d'un objet et est automatiquement appelée lors de la création de l'objet.

III.B.2.1 : Ce sont des classes filles.

III.B.2.2 : Spécialisation.

III.B.2.3 : Généralisation.

III.B.2.4 : La vision modulaire permet de découper un projet, et ainsi d'avoir une meilleure maintenance du code ainsi que la possibilité de réemploi du code déjà écrit pour d'autres classes.

III.B.2.5 : Parce que ces deux classes ont des méthodes communes, ce qui va obliger à écrire 2 fois la même chose.

III.B.2.6 : La méthode qui fixe la valeur du port (attribut privé).

III.B.3.1 : Dans ModbusTCP.h, la ligne `class ModbusTCP : public ModbusBase`

III.B.3.2 : Afin de limiter la visibilité des attributs.

III.B.3.3 : adresseEsclave public : affectation libre
Exemple : ObjetModbusBase.adresseEsclave = 0x01 ;
adresseRegistre protégé : affectation seulement par la classe mère ou les classes filles.
Exemple : ObjetModbusTCP.adresseRegistre = 0x1234 ;
port privé : affectation par une méthode à rajouter dans ModbusTCP.

III.B.3.4 : Algorithme :

```
Entrées:  menuVariateur, parametreVariateur Entiers de 16 bits
Sorties:  adresseRegistre Entier de 16 bits

Début calculerAdresseRegistre
     adresseRegistre ← menuVariateur * 100 + adresseRegistre – 1
Fin calculerAdresseRegistre
```

III.B.3.5 : Parce que la trame ModbusTCP contient des caractères nuls placés avant la fin de la trame => l'émission cesserait trop tôt.

III.B.4.1 : Non, car `ModbusBase` contient une méthode non virtuelle.

III.B.4.2.a : Déclaration d'un pointeur sur la classe `ModbusBase`.

III.B.4.2.b : Création d'un objet de type `ModbusSerie` ou `ModbusTCP` et affection de son adresse au pointeur MB.

III.B.4.2.c : Lors du new pour les objets de type `ModbusSerie` ou `ModbusTCP` et lors de la compilation du programme pour le pointeur `MB`.

III.B.4.3 : Les objets instanciés seraient détruits lorsqu'on quitte la fonction qui les a créés.

III.B.4.4 : Non. Seul un pointeur de la classe mère peut accéder à un objet enfant.

III.B.4.5 : Polymorphisme.

III.B.5.1 : cf. DR3 (réponses en rouge dans le document pdf).

III.B.5.2 : cf. DR3 (réponses en rouge dans le document pdf).

Modélisation du scooter

IV.A.1 : cf. DR4 corrigé (DAB en rouge dans le document pdf).

IV.A.2 : cf. DR4 corrigé (courbes exactes en orange dans le document pdf).

IV.A.3 : La pente est > à -20 dB/décade (algébriquement) => un premier ordre descendrait trop, et de plus, la phase varie énormément => il faut au numérateur un 1er ordre pour rajouter du gain avec un pôle à partie réelle >0 pour diminuer la phase. On n'a plus à faire à un système régulier.

Les DAB entourent assez bien le relevé pratique.

IV.A.4 : Tangente nulle à t = 0 => ordre du dénominateur ≥ 2.

On retrouve à peu près le gain statique : 3,8 * 20 / (1,3 * 15) = 3,9.

IV.A.5 : Travailler en petits signaux autour du point de repos.

IV.B.1 : $H_{BF}(p) = \dfrac{Vs(p)}{Vsc(p)} = \dfrac{H1(p)}{1+H1(p)}$, et $Vsc(p) = H1(p).\varepsilon(p)$ => $He(p) = \dfrac{\varepsilon(p)}{Vsc(p)} = \dfrac{1}{1+H1(p)}$.

AN : $He(p) = \dfrac{(1+\tau_1.p).(1+\tau_3.p)}{K.(1-\tau_2.P)+(1+\tau_1.p).(1+\tau_3.p)}$

IV.B.2.a Application aux erreurs :

$$\varepsilon_p = \lim_{t\to\infty} \varepsilon(t) = \lim_{p\to 0} p.He(p).\frac{1}{p} = \lim_{p\to 0} \frac{(1+\tau_1.p).(1+\tau_3.p)}{K.(1-\tau_2.P)+(1+\tau_1.p).(1+\tau_3.p)} = \frac{1}{K+1}$$

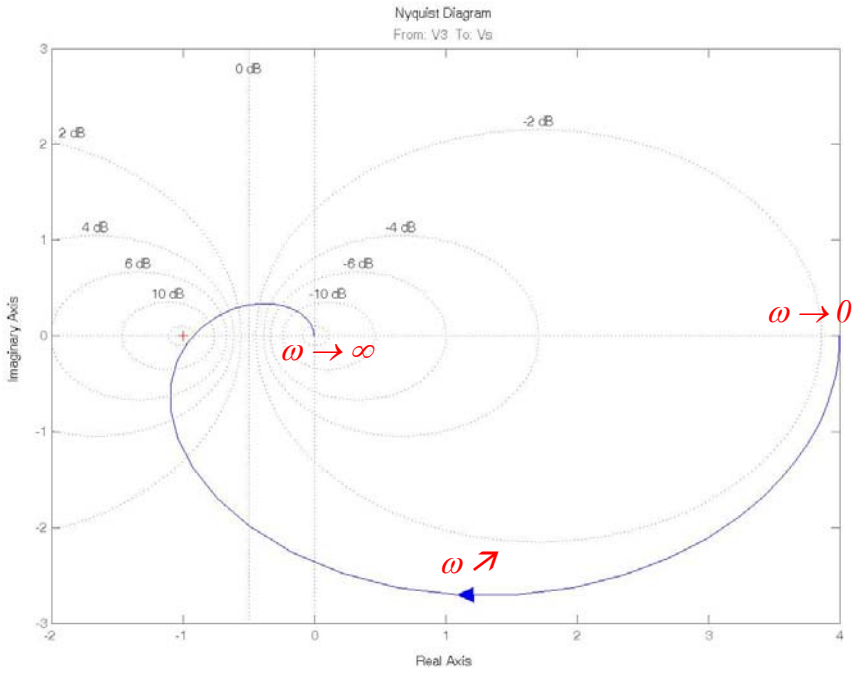
$$\varepsilon_v = \lim_{t\to\infty} \varepsilon(t) = \lim_{p\to 0} p.He(p).\frac{1}{p^2} = \lim_{p\to 0} \frac{(1+\tau_1.p).(1+\tau_3.p)}{K.(1-\tau_2.P)+(1+\tau_1.p).(1+\tau_3.p)}.\frac{1}{p} = +\infty$$

$$\varepsilon_a = \lim_{t\to\infty} \varepsilon(t) = \lim_{p\to 0} p.He(p).\frac{1}{p^3} = \lim_{p\to 0} \frac{(1+\tau_1.p).(1+\tau_3.p)}{K.(1-\tau_2.P)+(1+\tau_1.p).(1+\tau_3.p)}.\frac{1}{p^2} = +\infty$$

IV.B.2.b Le système est de classe 0 => $\varepsilon_p = \dfrac{1}{K+1}, \varepsilon_v = +\infty, \varepsilon_a = +\infty.$

IV.B.3.a Système à déphasage non minimal => il faut étudier le lieu de Nyquist complet.

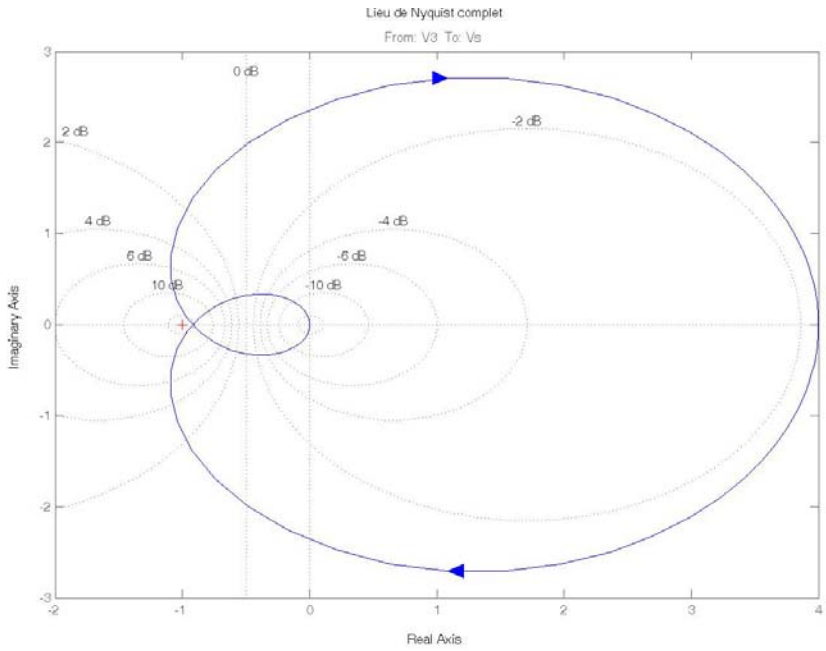
À partir de Bode, on peut tracer ce Nyquist (pulsations >0 uniquement).



IV.B.3.b Non, système non régulier (déphasage non minimal).

IV.B.3.c Il faut appliquer le critère complet de Nyquist, c a d compléter le lieu des pulsations >0 par celui des pulsations <0 et regarder le comportement autour de -1 avec la relation $N_{-1}=P_{BO>0} - P_{BF>0}$.

Le lieu de Nyquist complet est le suivant :



Application : Pas de pôle à Re > 0 en BO et le lieu de Nyquist n'entoure pas le point -1 => le système sera stable en BF.

IV.B.4 : $H_{BONC}(p) = \dfrac{K.(1-\tau_2.P)}{(1+\tau_1.p).(1+\tau_3.p)}$; Kc tel que le lieu de Nyquist passe par le point -1, d'où la double condition : Phase = -180° et Module = 1

On trouve $\omega_c = \sqrt{\dfrac{\tau_1+\tau_2+\tau_3}{\tau_1.\tau_2.\tau_3}} = 34{,}29 rad.s^{-1}$, soit F_C = 5,46 Hz.

Le calcul littéral de Kc (un peu long) donne Kc = 1,085 en remplaçant par les valeurs numériques.

IV.B.5 : Il faut un correcteur Proportionnel et Intégral (PI) ; $C1(p) = k_i.\dfrac{1+\dfrac{p}{\omega_i}}{\dfrac{p}{\omega_i}} = k_i.\dfrac{\omega_i+p}{p}$

IV.B.6 : On choisit ω_i de telle façon que la phase de $H_{BO}(\omega_i)$ vaille - 90° ; on trouve ω_i = 8,61 rad/s.
À cette pulsation, le gain de H_{BO} est de +7,51 dB, gain auquel se rajouteront les 3 dB du PI :
Il faut donc $k_i = 10^{\frac{-10,51}{20}} = 0{,}298$.

IV.B.7 : cf. DR5 (DAB en rouge dans le document pdf).

$$H_{BOC}(p) = 0{,}298.\frac{8{,}61 + p}{p}.\frac{K.(1-\tau_2.P)}{(1+\tau_1.p).(1+\tau_3.p)}.$$

IV.B.8 : chaîne directe = $H_{BOC}(p)$; retour unitaire => $H_{BFC}(p) = \frac{H_{BOC}(p)}{1+H_{BOC}(p)}$

1er cas : $H_{BOC}(p) >> 1 \Rightarrow H_{BFC}(p) \# 1$; 2nd cas : $H_{BOC}(p) << 1 \Rightarrow H_{BFC}(p) \# H_{BOC}(p)$.
cf. DR5 (DAB de la BOC en rouge, DAB de la BFC en mauve pointillé dans le document pdf).

IV.B.9 : En envoyant un échelon de couple sur le variateur de charge.

IV.B.10 : Non, car on va avoir une réponse oscillatoire ; il faudrait avoir un coefficient d'amortissement en BF ≥ 1.

IV.B.11 : Non, D1 est trop fort, de même pour la résonance ; de plus, undershot pour la réponse impulsionnelle. Tout cela est dû au zéro à partie réelle positive.

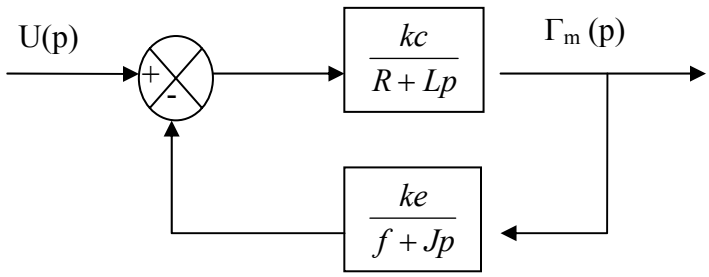
IV.C.1 : $u(t) = R.i(t) + L\frac{di(t)}{dt} + k_e.\Omega(t)$; $J\frac{d\Omega(t)}{dt} = k_c.i(t) - f.\Omega(t) + \Gamma_r(t)$.

IV.C.2 : Fonctions causales : $U(p) = (R + L.p).I(p) + k_e\Omega(p) \Leftrightarrow I(p) = \frac{1}{R+L.p}.[U(p) - k_e\Omega(p)]$

$$Jp\Omega(p) = k_c.I(p) - f.\Omega(p) + \Gamma_r(p) \Leftrightarrow \Omega(p) = \frac{1}{f+Jp}[k_c.I(p) + \Gamma_r(p)]$$

cf. DR6.

IV.C.3 : Schéma fonctionnel attendu :



Fonction de transfert correspondante : $Hc(p) = \frac{\Gamma_m(p)}{U(p)} = \frac{kc.(f + Jp)}{ke.kc + (f + Jp).(R + Lp)}$

IV.C.4 : K correspond au rapport cyclique du hacheur du scooter.

IV.C.5 : Le scooter ne roule que dans un seul sens ! !

IV.C.6 : Le correcteur sera un PI. Vu que la phase ne passe pas en dessous de -90°, on peut placer ce correcteur où l'on veut, par exemple à ωi = 2.π.10 (Fi = 10 Hz), d'où $Cc(p) = \frac{20.\pi + p}{p}$.

IV.C.7 : La réponse à cette question n'était pas immédiate, car il n'y avait pas de simulation de la boucle ouverte de la vitesse incorporant la boucle fermée de limitation de couple.

IV.D.1.a : $\begin{pmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\end{pmatrix} = \begin{pmatrix}\frac{-f}{J} & \frac{kc}{J}\\ \frac{-ke}{L} & \frac{-R}{L}\end{pmatrix}.\begin{pmatrix}x_1(t)\\ x_2(t)\end{pmatrix} + \begin{pmatrix}\frac{-1}{J} & 0\\ 0 & \frac{1}{L}\end{pmatrix}.\begin{pmatrix}\Gamma_r(t)\\ u(t)\end{pmatrix}$;

IV.D.1.b Tant que le vecteur d'état n'est pas défini, il en existe une infinité de matrices d'état (autant que de vecteurs différents possibles). Dès que le vecteur d'état est défini, il y a unicité de [A].

IV.D.2 Equation de sortie : $\Gamma_m(t) = \begin{pmatrix}0 & k_c\end{pmatrix}.\begin{pmatrix}x_1(t)\\ x_2(t)\end{pmatrix}$.

IV.D.3 Cf. DR6

IV.D.4.a Cf. DR6, réponses en rouge dans le document pdf.

IV.D.4.b Cf. DR6, réponses en vert dans le document pdf.

IV.D.5 Chaque boucle dispose de sa propre constante de temps, et peut donc aisément la modifier en modifiant le paramètre qui n'est pas utilisé dans les deux retours (R et f), même s'il on triche un peu.

IV.D.6.a Un système sera gouvernable si on peut amener le vecteur d'état d'une position $X(t_1)$ à une autre $X(t_2)$ par une commande ad hoc.

IV.D.6.b Il faut que la matrice de commandabilité $\mathcal{C}$ ait un rang de 2. $\mathcal{C} = \begin{pmatrix} B & A.B & A^2B & \dots \end{pmatrix}$.

IV.D.6.c Calcul de $\mathcal{C}$: B = $\begin{pmatrix} \frac{-1}{J} & 0 \\ 0 & \frac{1}{L} \end{pmatrix}$, et B est déjà de rang 2 => système gouvernable.

IV.D.7.a Un système sera observable si l'observation du vecteur d'état $X(t_1)$ pendant un temps fini permet de déduire l'état initial $X(t_0)$.

IV.D.7.b Il faut que la matrice d'observabilité $\mathcal{O}$ ait un rang de 2 ; $\mathcal{O} = \begin{pmatrix} C \\ CA \\ CA^2 \\ \dots \end{pmatrix}$

IV.D.7.c Calcul de $\mathcal{O}$: $C = \begin{pmatrix} 0 & kc \end{pmatrix}$; $CA = \begin{pmatrix} \frac{-ke.ke}{L} & \frac{-R.kc}{L} \end{pmatrix}$; si on arrête le calcul ici,

on obtient $\mathcal{O} = \begin{pmatrix} 0 & kc \\ \frac{-ke.kc}{L} & \frac{-R.kc}{L} \end{pmatrix}$ qui est de rang 2 => système gouvernable.

IV.D.8 Dans une base où la matrice d'état $[A]$ est diagonale, un système sera gouvernable (respectivement observable) si la matrice B (respectivement C) ne contient pas de 0.

IV.E.1 le filtre passe bas (anti repliement) avant le CAN.

IV.E.2 Cf. DR7 (réponses en rouge dans le document pdf).

IV.E.3 étude de la réponse impulsionnelle.

IV.E.4 un échantillonneur sur Vs(p).

IV.E.5 $z=e^{Te.p}$

IV.E.6. $H_1(p) = \frac{\tau_1 + \tau_2}{\tau_1 - \tau_3}.\frac{1}{1+\tau_1.p} - \frac{\tau_3 + \tau_2}{\tau_1 - \tau_3}.\frac{1}{1+\tau_3.p} = \frac{1{,}51}{1+\tau_1.p} - \frac{0{,}51}{1+\tau_3.p}$

IV.E.7 Idem, mais en rajoutant la FT de $B_0(p)$. $H_{BO}(p) = \frac{K.(1-\tau_2.P)}{(1+\tau_1.p).(1+\tau_3.p)}.\frac{1-e^{-Te.p}}{p} = T(p).(1-e^{-Te.p})$

Il faut donc décomposer $T(p) = \frac{K.(1-\tau_2.P)}{p.(1+\tau_1.p).(1+\tau_3.p)} = K.\left(\frac{1}{p} - \frac{0{,}254}{(1+\tau_1.p)} + \frac{0{,}014}{(1+\tau_3.p)}\right)$.

$T(p) = K.\left(\frac{1}{p} - \frac{0{,}254.\frac{1}{\tau_1}}{(p+\frac{1}{\tau_1})} + \frac{0{,}014.\frac{1}{\tau_3}}{(p+\frac{1}{\tau_3})}\right) = K.\left(\frac{1}{p} - \frac{1{,}512}{(p+5{,}95)} + \frac{0{,}519}{(p+37{,}04)}\right)$

D'où $T(z) = K.\left(\frac{z}{z-1} + \frac{1{,}512\ z}{z-\alpha} - \frac{0{,}519\ z}{z-\beta}\right)$, avec $\alpha = e^{-5{,}95.Te}$ et $\beta = e^{-37{,}04.Te}$.

Il suffit ensuite de rajouter le retard, d'où $H_{BO}(z) = K.\left(\frac{z}{z-1}+\frac{1{,}512\ z}{z-\alpha}-\frac{0{,}519\ z}{z-\beta}\right).\frac{z-1}{z}$.

IV.E.8 On utilisera la fonction de transfert en boucle fermée en recherchant les pôles du dénominateur.

IV.E.9 Un système sera stable si sa fonction de transfert H(p) n'a que des pôles à partie réelle négative.

IV.E.10 $p = j.\omega \Rightarrow z = e^{Te.p} = e^{Te.j\omega} = 1*\left[\cos(\omega.Te) + j.\sin(\omega.Te)\right]$ L'axe imaginaire se transforme en le cercle de centre O et de rayon unité.

IV.E.11 si on prend p=-1, alors de z correspondant est < à 1 ($e^{-Te} < 1$) ; si on prend p=1, alors de z correspondant est > à 1 ($e^{Te} > 1$). D'où : le demi-plan des parties réelles négatives devient l'intérieur du cercle unité. De même, le demi-plan de droite devient l'extérieur du cercle unité. D'où la règle : Un système échantillonné sera stable si sa fonction de transfert en z n'a que des pôles de module < 1.

IV.E.12 C1(z) = 1 $\Rightarrow H_{BF}(z) = \frac{H_{BO}(z)}{1+H_{BO}(z)}$; il faut donc chercher les zéros de 1 + H_{BO}(z).

$$H_{BO}(z) = K.\left(\frac{z}{z-1}+\frac{1{,}512\ z}{z-\alpha}-\frac{0{,}519\ z}{z-\beta}\right).\frac{z-1}{z}$$
$$= K.\left(\frac{(z-\alpha).(z-\beta) + 1{,}512\ (z-1).(z-\beta) - 0{,}519\ (z-\alpha).(z-1)}{(z-\alpha).(z-\beta)}\right)$$

d'où $1 + H_{BO}(z) = \frac{(K+1).(z-\alpha).(z-\beta) + 1{,}512\ K\ (z-1).(z-\beta) - 0{,}519\ K\ (z-\alpha).(z-1)}{(z-\alpha).(z-\beta)}$

$$= z^2(K+1+1{,}512K-0{,}519K) + z.[(K+1)(-\alpha-\beta) + 1{,}512K(-1-\beta) - 0{,}519K(-1-\alpha)] + [(K+1)(\alpha.\beta) + 1{,}512K.\beta - 0{,}519K.\alpha]$$

AN : $K = 3{,}8$; $\alpha = e^{-5{,}95.Te} = 0{,}971$; $\beta = e^{-37{,}04.Te} = 0{,}831$;

d'où : $1 + H_{BO}(z) = \frac{8{,}57\ z^2 - 15{,}28\ z + 6{,}733}{(z-\alpha).(z-\beta)}$

le discriminant $\Delta = 2{,}671 = 1{,}57^2$, d'où les deux zéros $z_1 = 0{,}983$ *et* $z_2 = 0{,}8$

IV.E.13 les modules des pôles de H_{BF}(z) sont < 1 $\Rightarrow$ système stable.

IV.E.14 $H_{BF}(z) = \frac{Vs(z)}{Vsc(z)} = \frac{C_1(z).H_{BO}(z).\varepsilon(z)}{Vsc(z)} = \frac{C_1(z).H_{BO}(z)}{1+C_1(z).H_{BO}(z)}$;

D'où $H_e(z) = \frac{\varepsilon(z)}{V_{sc}(z)} = \frac{1}{1+C_1(z).H_{BO}(z)}$

IV.E.15 on extrait C_1(z) du résultat précédent : $C_1(z) = \frac{1}{H_{BO}(z)}.\frac{Vsc(z) - \varepsilon(z)}{\varepsilon(z)}$.

IV.E.16 on étudiera la réponse à un échelon $\Rightarrow V_{sc}(z) = \frac{z}{z-1}$ avec un seul échantillon d'erreur non nul $\Rightarrow$

$\varepsilon(z) = \varepsilon_0$ et $H_{BO}(z) = K.\left(\frac{z}{z-1}+\frac{1{,}512\ z}{z-\alpha}-\frac{0{,}519\ z}{z-\beta}\right).\frac{z-1}{z}$

$$= K.\left(\frac{(z-\alpha).(z-\beta) + 1{,}512\ (z-1).(z-\beta) - 0{,}519\ (z-\alpha).(z-1)}{(z-\alpha).(z-\beta)}\right)$$

d'où : $C_1(z) = \dfrac{(z-\alpha).(z-\beta)}{K.\left[(z-\alpha).(z-\beta) + 1{,}512\ (z-1).(z-\beta) - 0{,}519\ (z-\alpha).(z-1)\right]} . \dfrac{\dfrac{z}{z-1} - \varepsilon_0}{\varepsilon_0}$

$= \dfrac{(z-\alpha).(z-\beta)}{K.\left[(z-\alpha).(z-\beta) + 1{,}512\ (z-1).(z-\beta) - 0{,}519\ (z-\alpha).(z-1)\right]} . \dfrac{z - \varepsilon_0.(z-1)}{\varepsilon_0.(z-1)}$

$= \dfrac{(z-\alpha).(z-\beta)}{K.\left[(z-\alpha).(z-\beta) + 1{,}512\ (z-1).(z-\beta) - 0{,}519\ (z-\alpha).(z-1)\right]} . \dfrac{z.(1-\varepsilon_0) + \varepsilon_0}{\varepsilon_0.(z-1)}$

Comme il n'y a pas de problème de causalité (numérateur et dénominateur du 3ème degré), ε_0 peut prendre la valeur que l'on veut ; pour simplifier et alléger le numérateur, on choisira $\varepsilon_0 = 1$.

D'où un correcteur possible :

$$C_1(z) = \frac{(z-\alpha).(z-\beta)}{K.\left[(z-\alpha).(z-\beta) + 1{,}512\ (z-1).(z-\beta) - 0{,}519\ (z-\alpha).(z-1)\right]} . \frac{1}{(z-1)}$$

En reprenant les résultats intermédiaires de la question **IV.E.12** ainsi que les valeurs numériques associées, on obtient :

$$C_1(z) = \frac{z^2 - 1{,}802\ z + 0{,}807}{(7{,}57\ z^2 - 15{,}28\ z + 6{,}733).(z-1)} = \frac{z^2 - 1{,}802\ z + 0{,}807}{7{,}57\ z^3 - 22{,}85\ z^2 + 22{,}013\ z - 6{,}733}$$

IV.E.17 La contrainte d'annulation des échantillons de la sortie est sans doute brutale ; cela risque d'engendrer une réponse oscillatoire lors de la réponse à 1 échelon.

DR 1

Question II.C.2

Chaque case correspond à un octet.

En-tête MBAP							FC	Données …							
00	01	00	00	00	06	01	06	02	79	00	01				

Remarque : en ce qui concerne la partie des données, la taille du tableau n'est qu'indicative ; le cas échéant, le candidat pourra bien évidemment le rallonger selon ses besoins.

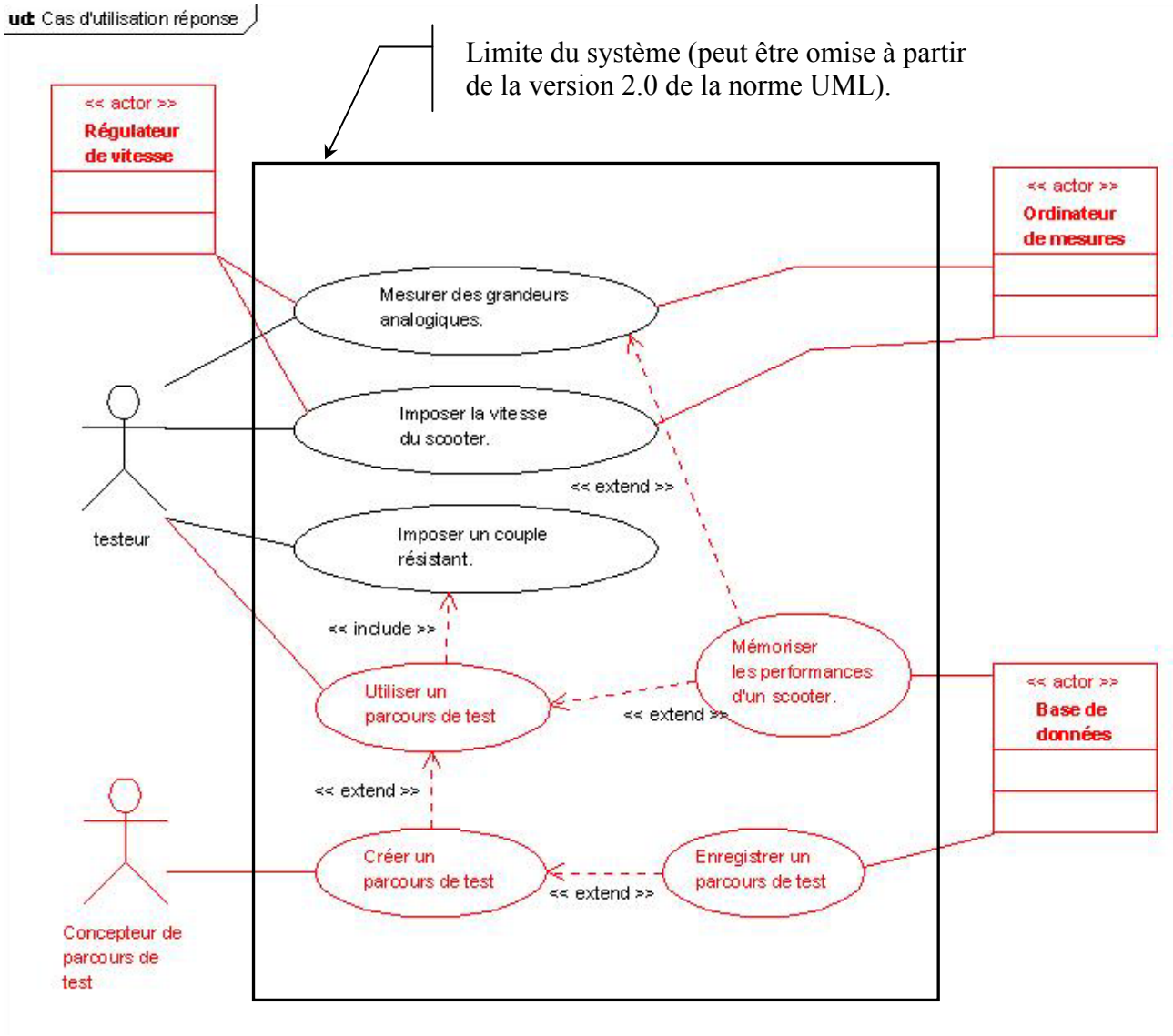
Question II.C.3

Adresse de l'esclave : 01

Menu et paramètre concernés : 0x279 = 633 => menu 6 paramètre 34

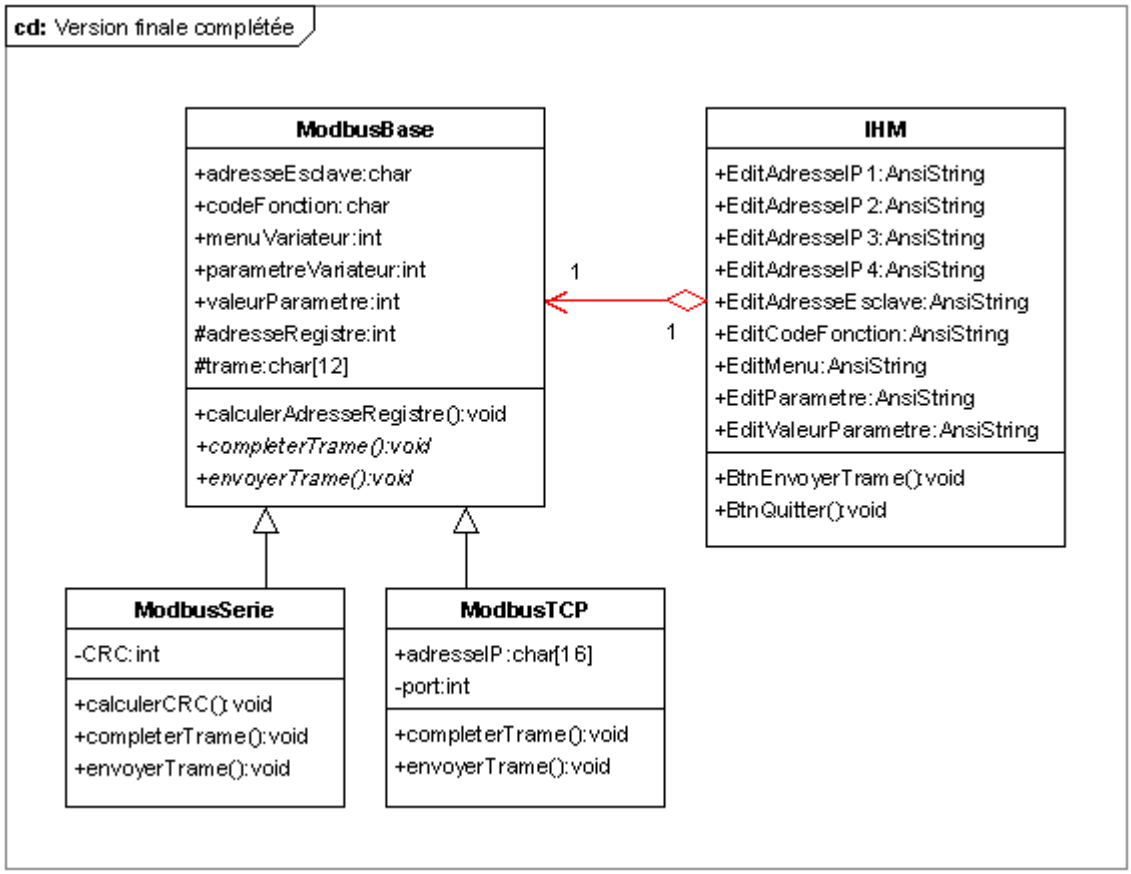
Opération réalisée : 06 = écriture d'un seul paramètre 16 bits dans le variateur.

DR 2 - Questions III.A.3 et III.A.4



On pourra aussi admettre que le cas d'utilisation « Mémoriser les performances d'un scooter » soit lié à l'ordinateur de mesures plutôt qu'au cas d'utilisation « Mesurer des grandeurs analogiques ».

DR 3 - Question III.B.5.1



Type de liaison : ***composition***

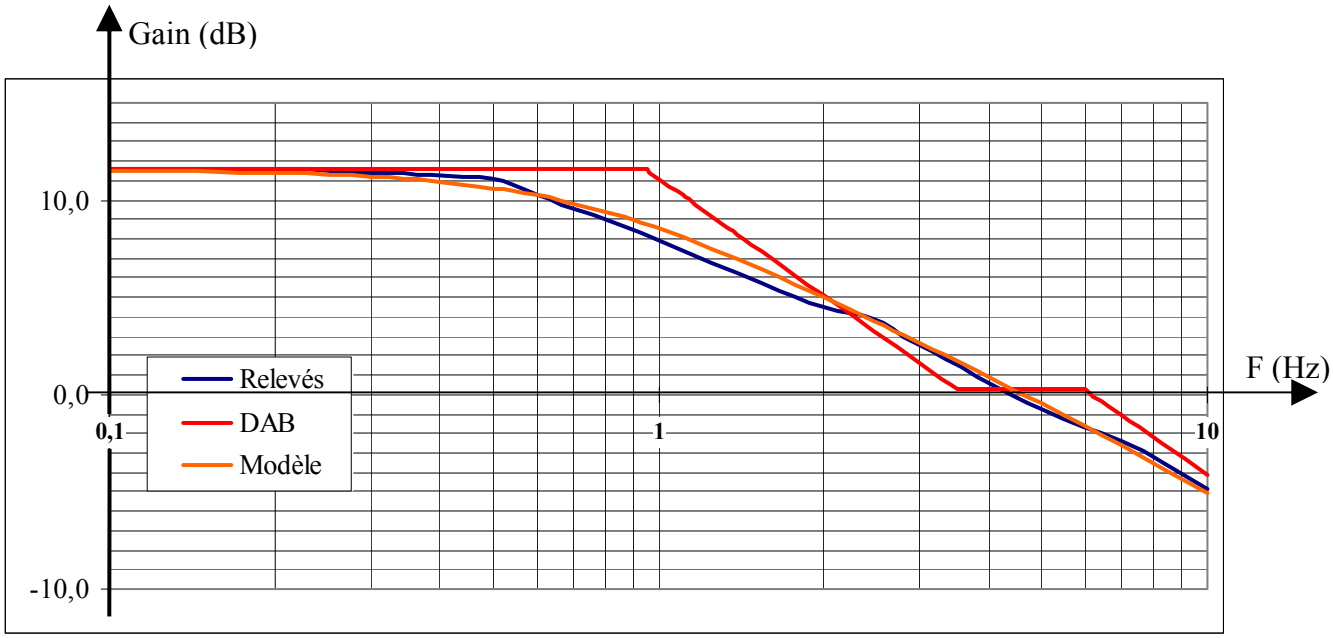
Justification : ***Composition : une instance de la classe ModbusBase n'a aucune raison d'exister si celle de la classe IHM qui la créée a disparu.***

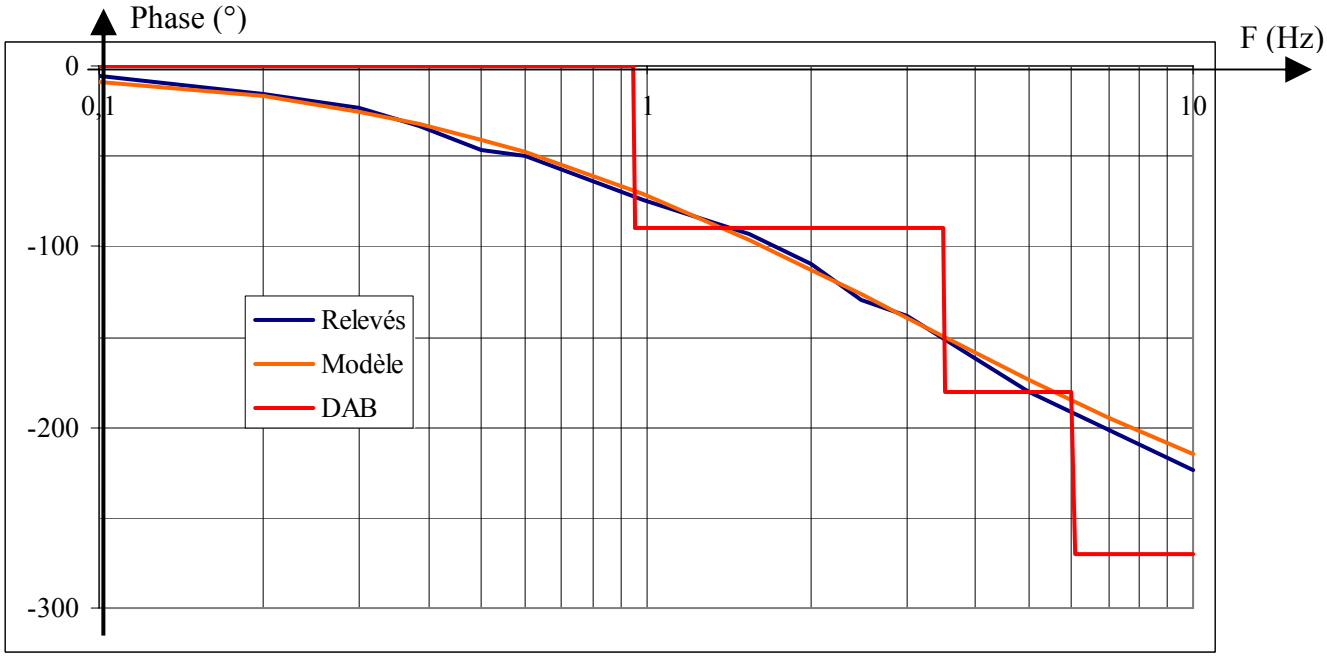
Navigabilité : toute instance de la classe IHM fera forcément référence à une instance de la classe ModbusBase.

1 côté IHM : Une instance de la classe ModbusBase ne peut avoir qu'une seule IHM

1 côté ModbusBase : Une instance de la classe IHM ne peut piloter qu'une seule instance ModbusBase

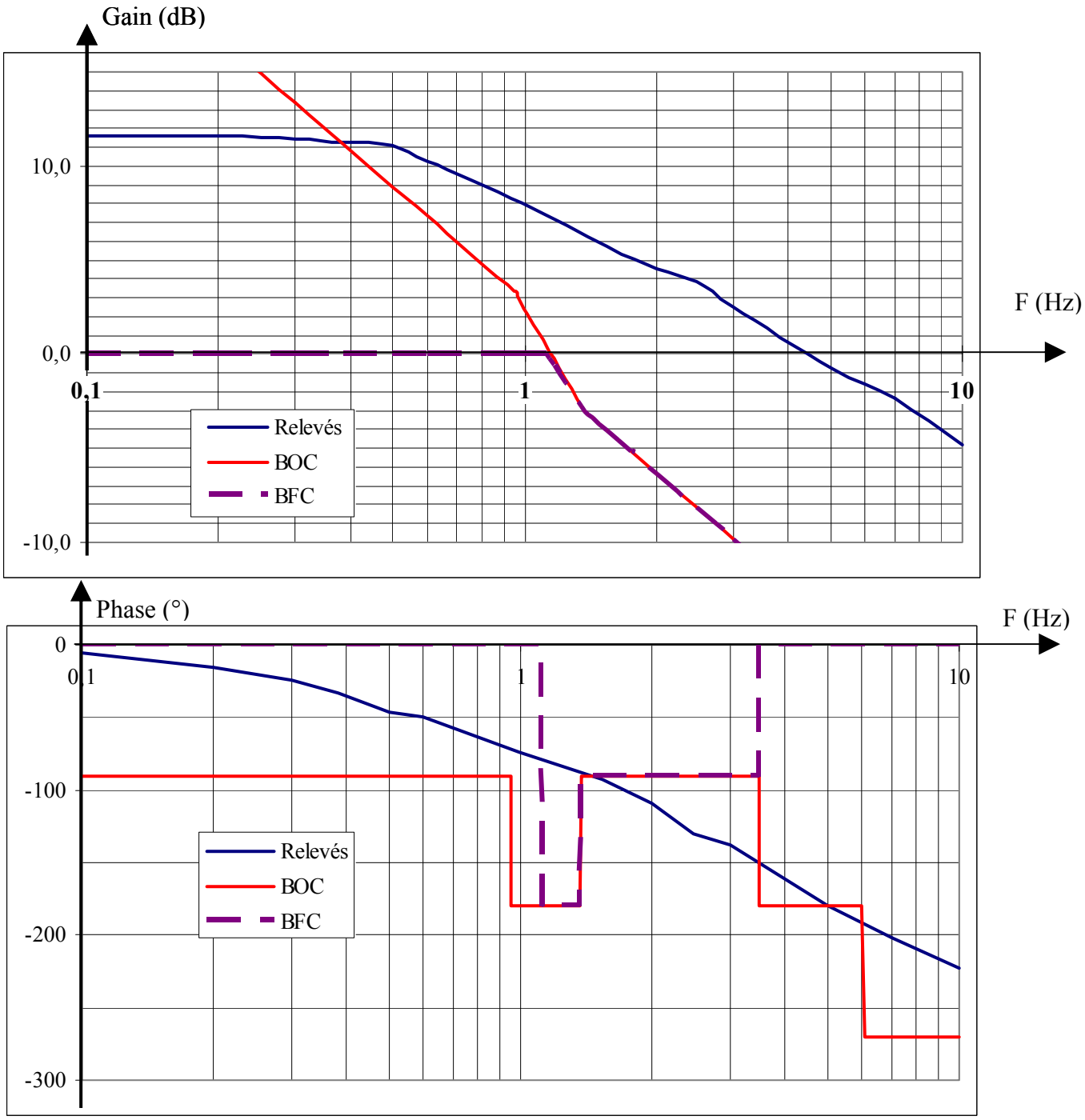
DR 4 - Questions IV.A.1 et IV.A.2





Au delà de 10 Hz, les mesures ne sont plus significatives (trop de bruit).

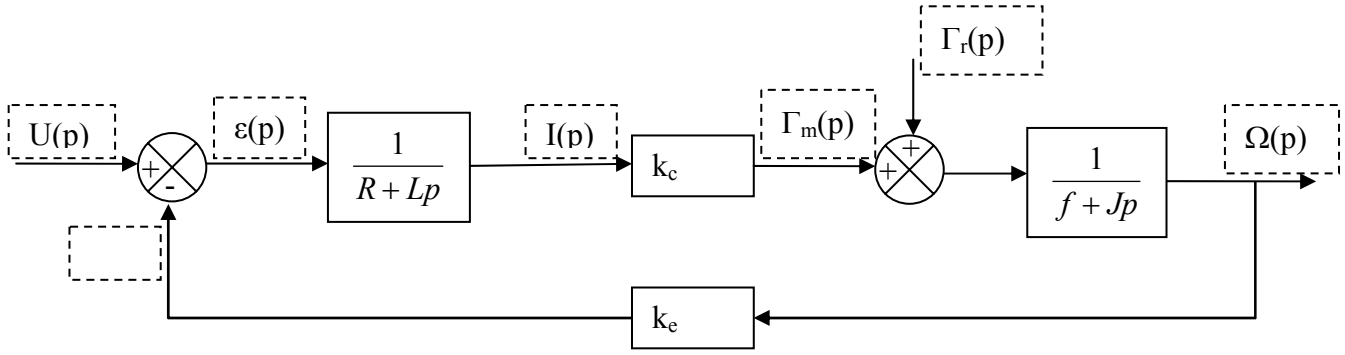
DR 5 - Questions IV.B.6 et IV.B.7



Au delà de 10 Hz, les mesures ne sont plus significatives (trop de bruit).

DR 6

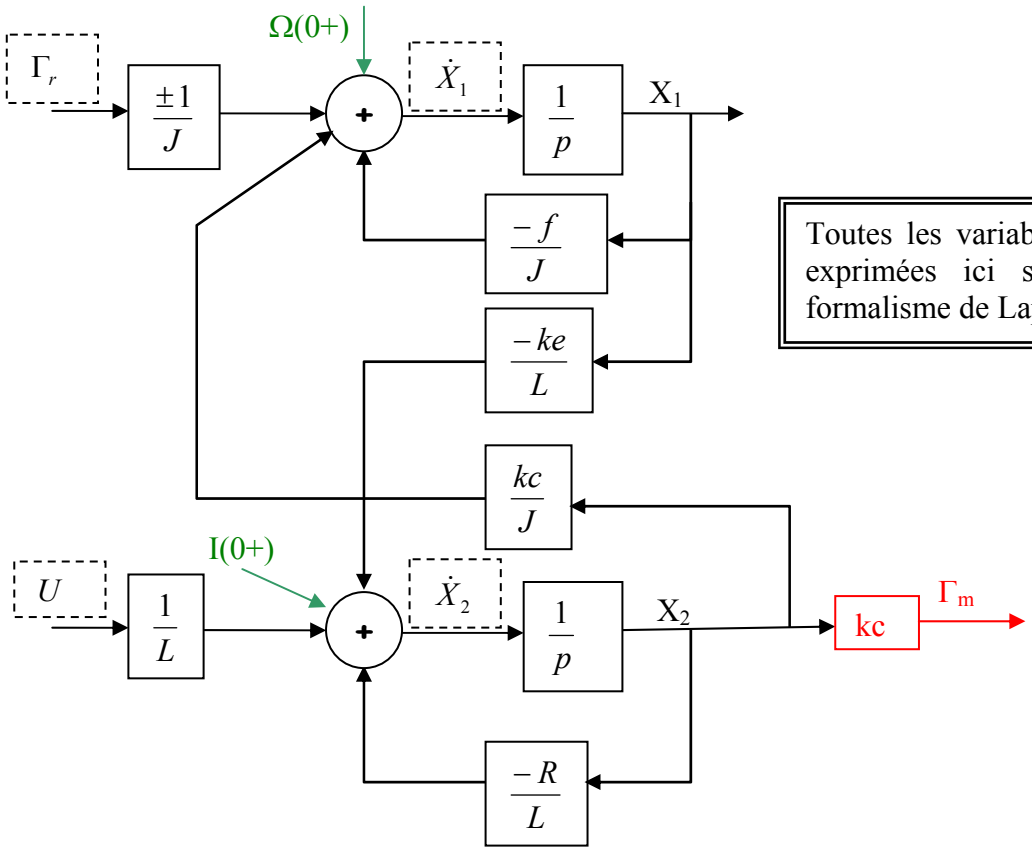
Question IV.C.2



Les cases pointillées seront remplies par le nom d'un signal, les autres par une fonction de transfert ou un gain.

Questions IV.D.3, IV.D.4.a et IV.D.4.b

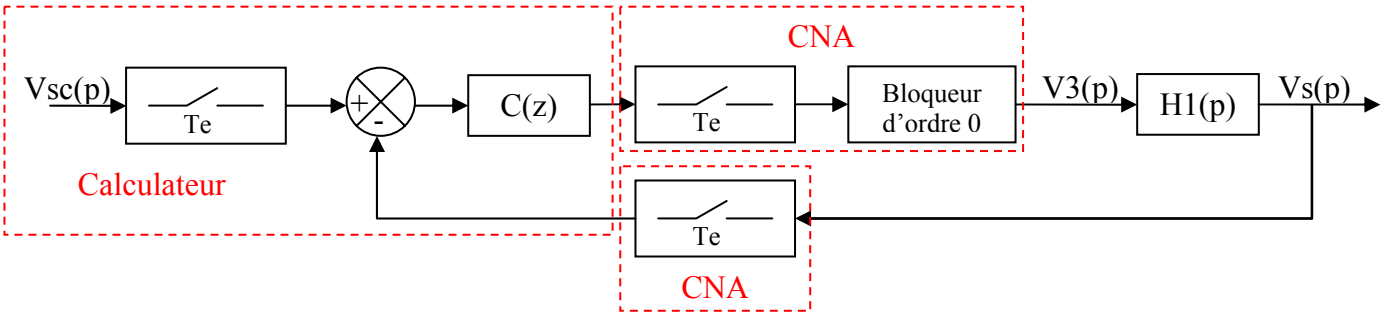
En noir en rouge en vert



Toutes les variables sont exprimées ici selon le formalisme de Laplace.

Les cases pointillées seront remplies par le nom d'un signal, les autres par une fonction de transfert ou un gain. Le couple résistant étant par essence même aléatoire, le candidat ne se formalisera pas sur son signe.

DR 7 – Question IV.E.2



L'échantillonnage de Vsc est fictif

Annexe 1 – Le protocole Modbus.

I Présentation succincte.

Le protocole MODBUS (MODicon BUS) apparaît en 1978 comme protocole de communication entre les automates de la société Modicon et la console de programmation, et fut ensuite utilisé par plusieurs constructeurs d'automates.

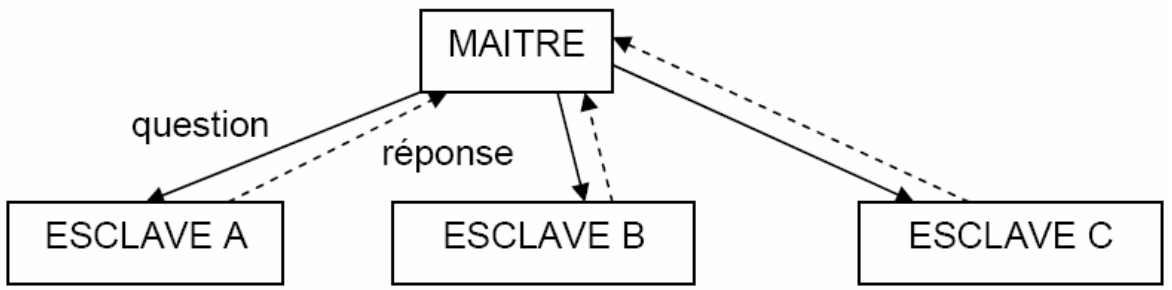
Initialement, les échanges se faisaient sur une liaison série RS232 (1 seul esclave) ou RS485. Actuellement, on utilise une liaison Ethernet pour véhiculer ce protocole. Le service de messagerie MODBUS TCP doit fournir une socket d'écoute sur le port 502.

Ce protocole n'est donc pas dépendant du support physique.

II Le réseau MODBUS.

II.A Architecture du réseau.

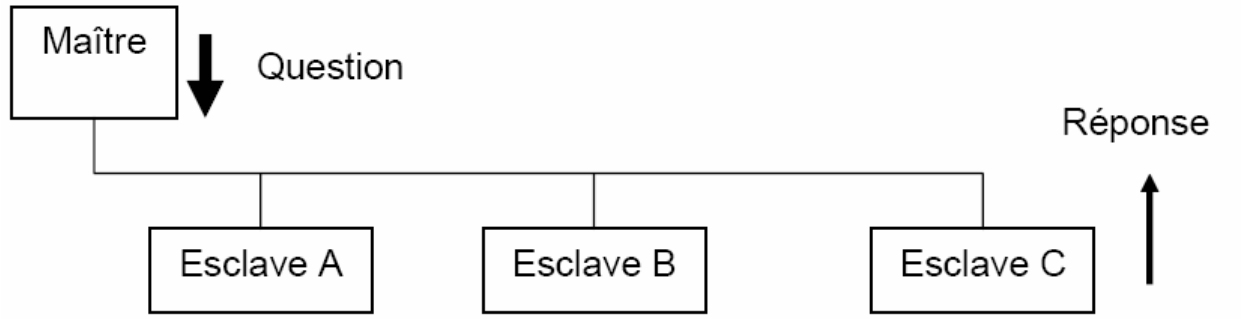
Le protocole MODBUS TCP est un protocole maître esclave, où seul le maître gère les échanges et en a l'initiative. Le maître envoie une demande et attend la réponse de l'esclave concerné. Le dialogue maître esclave peut être schématisé sous forme de liaisons point à point successives.



Deux esclaves ne peuvent dialoguer ensemble. L'esclave doit avoir un temps de réponse variant entre 0,2 et 1 seconde.

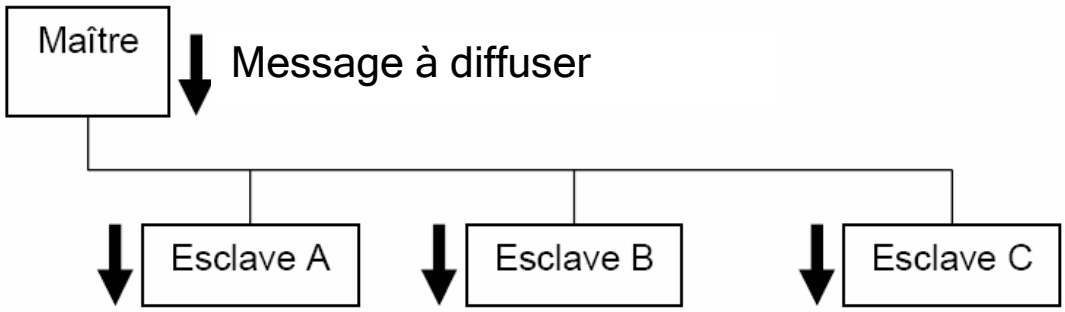
II.B Echange maître esclave.

Le maître interroge un esclave de numéro unique sur le réseau et attend de la part de cet esclave une réponse.



II.C Echange maître vers tous les esclaves

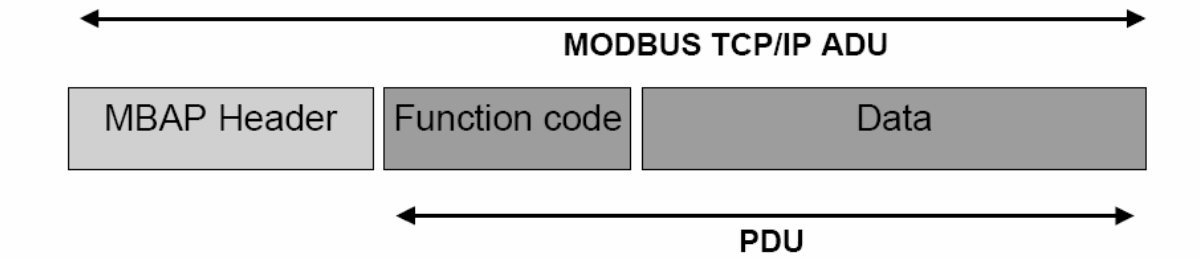
Le maître diffuse un message à l'adresse 0 et atteint ainsi tous les esclaves présents sur le réseau ; ceux-ci exécutent l'ordre du message sans émettre de réponse.



III Trame MODBUS TCP.

III.A Structure de la trame.

Une trame contient les trois champs suivants :



1er champ : l'en-tête MBAP (ModBus Application Protocole). Ce champ fait 7 octets et contient :

- Le numéro de la transaction, codé sur 2 octets. Ce nombre sert à numéroter les trames et peut servir au diagnostic en cas de défaillance ; il peut aussi être laissé à 0.
- L'identificateur de protocole, codé sur 2 octets, inutile ici. Il sera laissé à 0.
- La longueur du PDU + 1 (l'adresse de l'esclave), codé sur 2 octets.
- L'adresse de l'esclave, codée sur 1 octet.

2nd champ : le code de la fonction que l'on veut réaliser. On se limitera ici aux accès aux registres du variateur. Quelques valeurs de fonction :

- **0x03** : lecture d'une série de registres de 16 bits contigus. Le champ Data contiendra l'adresse du premier registre à lire (2 octets) ainsi que le nombre de registres à lire (2 octets) (maxi = 125). On spécifiera ce nombre à 1 pour ne lire qu'un seul registre.
- **0x10** : écriture d'une série de registres de 16 bits contigus. Le champ Data contiendra l'adresse du premier registre à écrire (2 octets), le nombre de mots de 16 bits à écrire (maxi = 100) (2 octets), le nombre d'octets à écrire (soit le double du précédent) (1 octet), et enfin les mots de 16 bits à écrire dans les registres de l'esclave.
- **0x06** : écriture dans un seul registre de 16 bits de l'esclave. Le champ Data contiendra l'adresse du registre (2 octets) suivi du mot de 16 bits à écrire.

3ème champ : les données qui complètent le code de la fonction envoyée en accord avec ce qui précède.

Le protocole MODBUS utilise la représentation « big endian » pour exprimer les nombres codés sur plus d'un octet ; l'octet de plus fort poids du nombre à envoyé sera donc transmis en premier.

III.B Application au variateur de charge.

Le court extrait de la documentation du variateur illustre de quelle façon les adresses des registres sont calculées.

I4.4 - Affectation des paramètres

Les variateurs UNIDRIVE SP sont paramétrés en utilisant une notation **menu.paramètre**.
Les index "menu" et "paramètre" peuvent prendre les valeurs 0 à 99. Le menu.paramètre est affecté à un registre MODBUS RTU **menu x 100 + paramètre**.

Pour affecter correctement les paramètres, l'esclave incrémente (+1) l'adresse du registre reçu.
Exemple : X = menu ; Y = paramètre

Paramètre variateur	Adresse registre (niveau protocole)
X.Y	(X x 100) + (Y – 1)
Exemples :	
1.02	101
1.00	99
0.01	0
70.00	6999

IV Trame MODBUS série.

L'extrait suivant de la documentation du variateur illustre le protocole MODBUS lorsqu'il est véhiculé par une liaison série.

Hormis le CRC et la disparition d'une grande partie de l'en-tête, tout ce qui est vrai pour le protocole MODBUS série est vrai pour le protocole MODBUS TCP.

Les trames seront les mêmes, que la liaison série soit une RS232 ou une RS485.

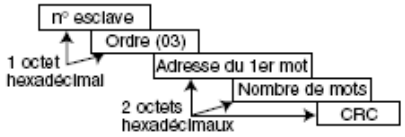
Contrairement au protocole MODBUS TCP, seul ce CRC est représenté en « little endian », ce n'est pas une erreur.

I4.6.1 - Code fonction 3 : lecture

Lecture d'une zone contiguë de registres. L'esclave impose une limite haute sur le nombre de registres qui peuvent être lus. Si la limite est dépassée, l'esclave produira une exception code 2.

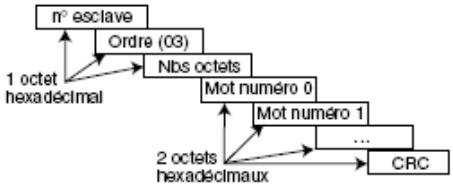
Trame envoyée par le Maître :

Octets	Description
0	Adresse de l'esclave (1 à 247)
1	Code fonction 0x03
2	Poids fort de l'adresse du premier mot
3	Poids faible de l'adresse du premier mot
4	Poids fort du nombre de mots à lire
5	Poids faible du nombre de mots à lire
6	Poids faible du CRC
7	Poids fort du CRC



Trame renvoyée par l'esclave :

Octets	Description
0	Adresse de l'esclave
1	Code fonction 0x03
2	Nombre d'octets à lire
3	Poids fort du mot 0
4	Poids faible du mot 0
5	Poids fort du mot 1
6	Poids faible du mot 1
...	...
n	Poids faible du CRC
n + 1	Poids fort du CRC



Annexe 2 –

modèle OSI.

OSI = Open Systems Interconnection. Ce modèle a été mis en place par l'ISO (International Standard Organisation) afin de mettre en place des normes de communications entre les ordinateurs d'un réseau, c'est-à-dire les règles qui gèrent les communications entre des ordinateurs.

Ce modèle est découpé en 7 couches :

n° de la couche	Nom de la couche.	Rôle (s) de la couche
7	Application	Comme son nom l'indique. Il s'agit du niveau le plus proche des utilisateurs.
6	Présentation	La couche présentation définit le format des données manipulées par le niveau applicatif (leur représentation, éventuellement leur compression et/ou leur chiffrement), indépendamment du système.
5	Session	La couche session définit l'ouverture et la destruction des sessions de communication entre les machines du réseau.
4	Transport	Transporter des messages, éventuellement en les fractionnant en paquets ; gérer les éventuelles erreurs de transmission.
3	Réseau	Transporter des paquets. La couche réseau permet de gérer l'adressage et le routage des données, c'est-à-dire leur acheminement via le réseau.
2	Liaison	Transporter des trames, avec correction d'erreur. La couche liaison définit l'interface avec la carte réseau et le partage du média de transmission.
1	Physique	Transporter des bits, sans correction d'erreur. La couche physique définit la façon dont les données sont physiquement converties en signaux numériques sur le média de communication (impulsions électriques, modulation de la lumière, etc…).

Les couches 1 à 4 sont appelées **couches basses**, les trois autres **couches hautes**. Cette dichotomie est justifiée par le fait que, vu de la couche 5, on est incapable de dire de quelle façon les données sont acheminées sur le réseau. On n'a aucune information sur la taille des paquets, si des erreurs ont été corrigées ou non …

Matériels réseau possibles

Hub ou répéteur: ce dispositif diffuse à tous les éléments qui y sont connectés l'intégralité des trames reçues.

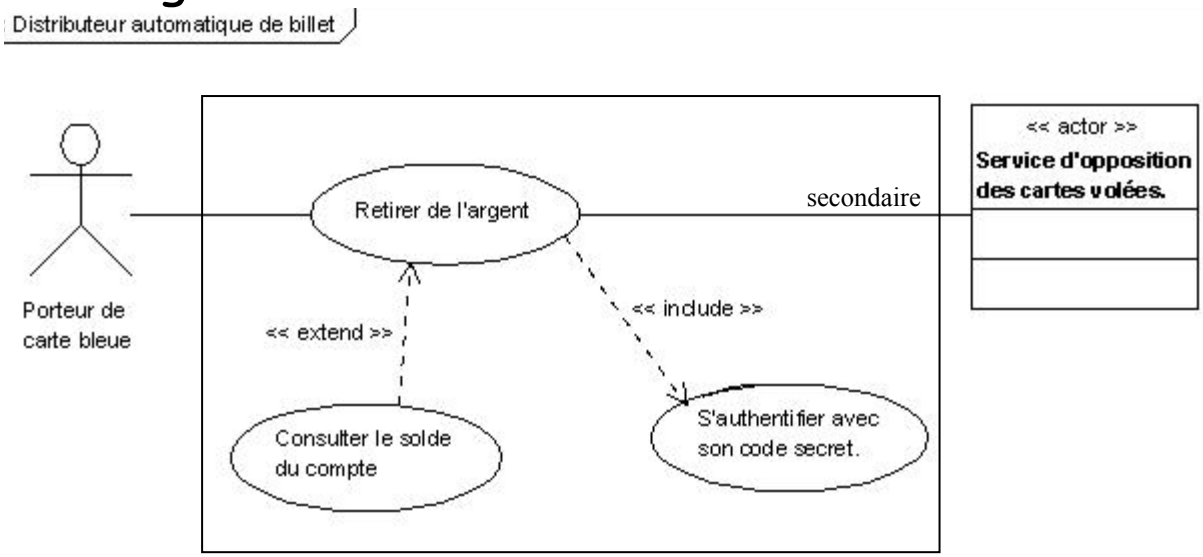
Switch ou commutateur : idem au répéteur, mais il ne renvoie les trames reçues qu'à la seule machine destinataire qu'il identifie par l'adresse de sa carte réseau ; cette adresse est codée sur 6 octets.

Pont : dispositif qui fournit une interface entre le réseau Ethernet et un autre réseau (fibre optique, par exemple). Agit au niveau de la couche 2.

Routeur : dispositif qui assure le routage des données en transit. Agit au niveau de la couche 3

Annexe 3 – diagrammes UML

Diagramme des cas d'utilisation.



Le diagramme illustre de façon très simplifiée le fonctionnement d'un distributeur automatique de billets de banque.

On trouve deux acteurs :

- Un acteur primaire : le porteur de carte bleue. Il agit sur le système.
- Un acteur secondaire : le service d'opposition des cartes volées. Il est sollicité par le système.

Le cas d'utilisation *retirer de l'argent* se voit adjoindre deux cas avec deux dépendances différentes :

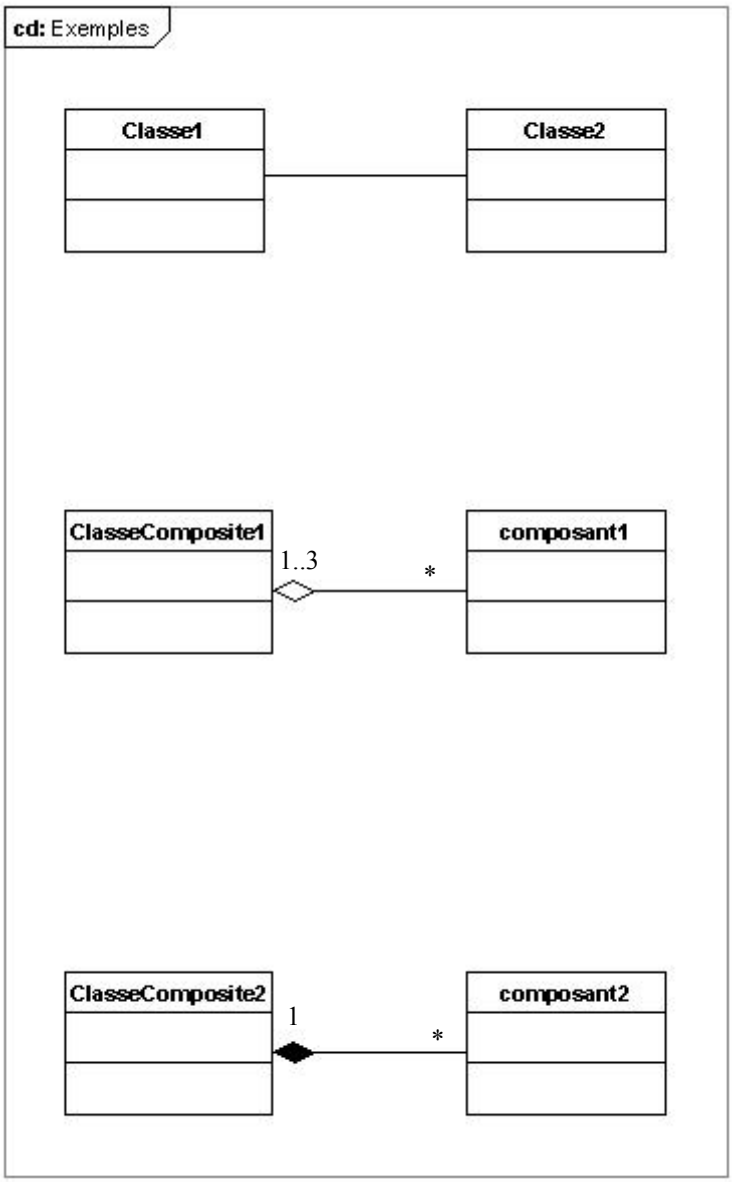
- Dépendance **include** : le cas d'utilisation *retirer de l'argent* a toujours besoin du cas d'utilisation *s'authentifier avec son code secret*. Par contre, le cas d'utilisation *s'authentifier avec son code secret* ne peut pas être associé à l'acteur principal, car on ne va pas vers un distributeur de billets pour le plaisir de s'authentifier, histoire de passer le temps…
- Dépendance **extend** : le cas d'utilisation *retirer de l'argent* peut avoir besoin du cas d'utilisation *consulter le solde du compte*. La consultation du solde est une possibilité offerte, elle n'est en aucun cas obligatoire.

Cette explication n'est qu'une façon de voir, on aurait aussi pu considérer que l'acteur principal pourrait désirer consulter le solde de son compte uniquement, ce qui induirait une association entre ce cas et l'acteur, et une nouvelle dépendance d'inclusion entre les cas *consulter le solde du compte* et *s'authentifier avec son code secret*, ce dernier étant le cas inclus.

Le raisonnement ci-dessus montre bien que la démarche UML est **itérative** ; il est normal de faire évoluer ses schémas au fur et à meure du développement du projet (c'est plutôt le contraire qui ne le serait pas). Le candidat sera sans doute amené à faire de même au fur et à mesure de son appropriation du projet.

Il est aussi important de se persuader que la modélisation UML n'exprime pas forcément la réalité, mais une vision subjective de celle-ci qui suffit au problème posé.

Diagramme de classes.



Le premier exemple illustre une **association**.
But : mettre en relation des objets appartenant à 2 classes différentes.
Caractéristiques d'une association :
Son **nom**, qui exprime l'idée du lien.
Les **rôles** joués par les objets de chaque classe.
Les **cardinalités** (nb d'objets de chaque classe) ; sera détaillé plus bas.
L'**orientation** ou **navigabilité**, qui précise les modalités de parcours du lien par une flèche. Pas de flèche ⇔ bidirectionnel.

Le second exemple illustre une **agrégation**.
La classe composite 1 est constituée d'objets issus de la classe composant 1 qui peuvent être partagés avec d'autres classes.
Exemple : une équipe (composite) et constituée de joueurs (composants) ici en nombre illimité. Par contre un joueur peut appartenir à plusieurs équipes (club + 2 sélections).
La disparition de l'équipe n'implique pas celle des joueurs.

Le dernier exemple illustre une **composition**.
La classe composite 2 est constituée d'objets issus de la classe composant 2 qui ne peuvent être partagés.
Exemple : un clavier (composite) et constituée de touches (composants) ici en nombre illimité. Par contre une touche ne peut appartenir qu'à un seul clavier.
La disparition du clavier implique celle des touches.

Dans une association non bidirectionnelle, le sens de la flèche indique quelle classe fait référence à l'autre. Une flèche partant de classe_1 et allant vers classe_2 signifierait que tous les objets qui seront créés à partir de classe_1 feront forcément référence à un objet créé à partir de classe_2. En l'absence de flèche, l'association est bidirectionnelle.

L'agrégation est une forme particulière d'association non symétrique qui exprime l'idée qu'un objet « fait partie » d'un autre objet. Sa sémantique détermine généralement un lien fort entre les classes concernées. L'une des classes est composée d'objets issus de l'autre classe et joue donc un rôle de conteneur. Dans la théorie de ensembles, l'agrégation correspond à la notion d'appartenance.

Remarque : on aurait aussi pu mettre `0..3` au lieu de `1..3` du côté de la classe composite 1, car un joueur peut ne pas avoir d'équipe.

Dans la théorie de ensembles, la composition correspond à une inclusion. Dans le cas de la composition, du fait du caractère non partageable, la cardinalité maximum relative au composite est toujours 1. Si, de plus, la cardinalité minimum est 1, alors la partie ne peut exister en l'absence du tout et tous les composants seront automatiquement détruits lors de la destruction du composite.

Dans une composition, l'existence du composant n'a de sens que si le composite existe.

Annexe 4 - formalisme algorithmique

La première qualité d'un algorithme est d'être lisible et compréhensible par n'importe quel technicien. A ce titre, il est souvent très utile de le commenter.

Conventions d'écriture:

```
{}                  {ceci indique un commentaire}
" "                 {contenu d'un tableau: "1 2 6 7"}
' '                 {caractère: 'a'}
```

Opérateurs:

```
=                   {comparaison d'égalité dont le résultat est Booléen}
NON=                {comparaison de différence}
<-                  {flèche qui symbolise une affectation, un chargement}
>                   {supérieur}
<                   {inférieur}
^                   {puissance 100000=10^5}
ainsi que +,-,/,*, ET, OU, NON
                    {le * est le multiplier informatique}
les fonctions particulières seront notées en toutes lettres:
RacineCarrée(VARIABLE)
Sinus(VARIABLE)
Concaténation("bonjour", "monsieur")
                    {résultat "bonjourmonsieur"}
PartieEntière(VARIABLE){supprime les décimales de VARIABLE}
```

Types:

```
Entier NB bits      {NB est un entier}
Réel compris entre -NB1 et +NB2 avec NB3 décimales
Caractère           {8 bits, notation: 'c' pour le caractère c}
Booléen             {1 bit, pouvant prendre les valeurs VRAI ou FAUX, plus claires que}
                    {1 ou 0 que l'on utilisera toutefois pour les tableaux de booléens}
Tableau de NB TYPE  {Tableau de 35 Réels compris entre -3^3 et +10^6 avec 3 décimales}
                    {le tableau contient les variables nom_tableau[0] à nom_tableau[NB-1]}
                    {affectation:  nom_tableau<-"bonjour" pour un tableau de caractères}
                    {              nom_tableau<-"0 1 0 1 1" pour un tableau de booléens}
                    {              nom_tableau<-"12 13 15 78 2" pour un tableau d'entiers}
                    {remplir la 5ème case avec le nombre 45 s'écrit: nom_tableau[4]<-45}
```

Architecture:

```
Déclaration du type de chaque entrée
Déclaration du type de chaque sortie
Déclaration du type de chaque variable interne
Début   nom_algo
        ...
Fin     nom_algo
```

Structures itératives: {il est recommandé de décaler les structures vers la droite}
{lorsqu'elles sont imbriquées dans d'autres}

Tant que (condition) Faire	Répéter	Pour nom_var allant de NB1 à NB2 Faire
...	...	...
FinTantque	Jusqu'à (condition)	FinPour
	FinRépéter	

{la structure Répéter est équivalente au "Faire...Tantque()"}

Structure conditionnelle:

Si (condition) Alors	Au Cas ou VARIABLE vaut
...	VALEUR1 Faire
Sinon	...
...	VALEUR2 Faire
FinSi	...
	VALEUR3 Faire
	...
	Autrement Faire
	...
	FinCas

{le Sinon ou l'Autrement peuvent être absent des structures}

Accès à une adresse, pointeurs:

Pour accéder à une adresse 16 bits, on déclarera le tableau PlanMemoire de 65536 entiers de 8 bits. Charger dans A la donnée présente à l'adresse X s'écrira: A<-PlanMemoire[X]. Cette écriture permet de gérer les accès à la mémoire du micro, les pointeurs, les vecteurs de redirection et les E/S du micro.

Exemples:

ALGORITHME DE CONVERSION DECIMAL BINAIRE DANS UNE CHAINE DE CARACTERE

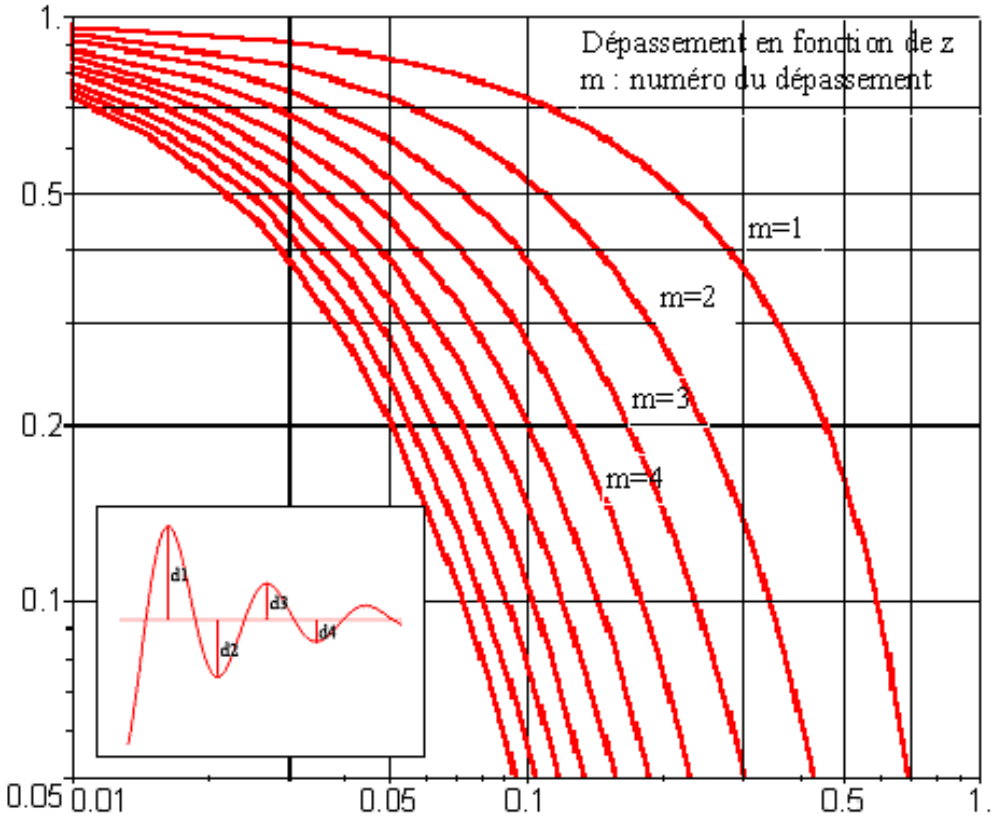
```
Entrées:       NB_DECIMAL Entier de 32 bits
Sorties:       MOT_BINAIRE Tableau de 32 Caractères
Variables internes:   QUOTIENT Réel compris entre 0 et +4,3^9 avec 1 décimale
                      INDICE Entier de 5 bits
Début Conversion_Décimal_Binaire_dans_chaîne_de_caractère
       QUOTIENT<-NB_DECIMAL
       INDICE<-0
       Répéter
              QUOTIENT<-PartieEntière(QUOTIENT)/2
              Si (PartieEntière(QUOTIENT)=QUOTIENT) Alors
                                                  {il n'y a pas de décimale dans QUOTIENT : }
                                                  {le reste de la division par deux est zéro}
                     MOT_BINAIRE[31-INDICE]<-'0'  {on remplit le tableau en commençant par}
                                                  {la dernière case (le bit de poids faible)}
              Sinon
                     MOT_BINAIRE[31-INDICE]<-'1'  {le reste de la division par deux est 1}
              FinSi
              INDICE<-INDICE+1                    {permet un déplacement dans le tableau}
       Jusqu'à (PartieEntière(QUOTIENT)=0)        {fin de la conversion : plus de partie entière}
       FinRépéter
       Tant que (INDICE NON= 32)
              MOT_BINAIRE[31-INDICE]<-'0'         {on remplit le début du tableau avec des zéros}
              INDICE<-INDICE+1
       FinTantque
Fin Conversion_Décimal_Binaire_dans_chaîne_de_caractère
```

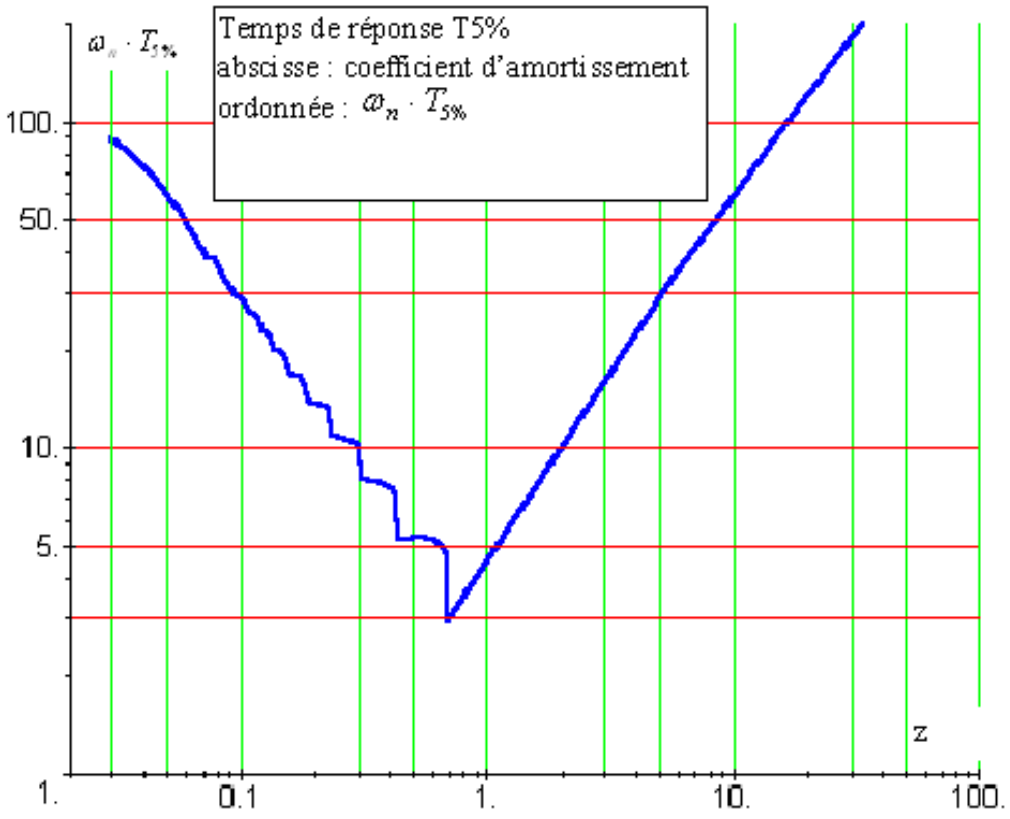
ALGORITHME AFFICHAGE DE LA VALEUR BINAIRE D'UN NOMBRE DECIMAL SAISI AU CLAVIER

```
Entrées:       DECIMAL Entier de 32 bits
Sorties:       BINAIRE Tableau de 32 Caractères
Variables internes:   REPONSE Caractère
Début Saisie_nombre_décimal_et_affichage_de_sa_valeur_binaire
       Répéter
              Afficher("saisir au clavier le nombre à convertir")
              DECIMAL<-LectureClavier
              BINAIRE<-Conversion_Décimal_Binaire_dans_chaîne_de_caractère(DECIMAL)
              Afficher("sa valeur binaire est:")
              Afficher(BINAIRE)
              Afficher("continer? O/N")
              REPONSE<-LectureClavier
       Jusqu'à (REPONSE='N' OU REPONSE='n')
       FinRépéter
Fin Saisie_nombre_décimal_et_affichage_de_sa_valeur_binaire
```

Ces exemples expliquent le fonctionnement des programmes désirés sans aucune référence à une quelconque implémentation, tant au niveau matériel (pas d'emploi de nom de registre de processeur…) que logiciel (on ne sait pas encore en quoi ce programme sera écrit).

Annexe 5 – Abaques du 2nd ordre.





Annexe 6- transformées de Laplace et en z.

f(t)	**F(p)**	**F(z)**
$\delta(t)$	1	1
$\delta(t-nTe)$	$e^{-n.Te.p}$	z^{-n}
u(t)	$\frac{1}{p}$	$\frac{z}{z-1}$
t.u(t)	$\frac{1}{p^2}$	$\frac{Te.z}{(z-1)^2}$
$\frac{t^2}{2}.u(t)$	$\frac{1}{p^3}$	$\frac{Te^2.z.(z+1)}{2.(z-1)^3}$
$e^{-a.t}.u(t)$	$\frac{1}{p+a}$	$\frac{z}{z-e^{-a.Te}}$

DR 1

Question II.C.2

Chaque case correspond à un octet.

En-tête MBAP							FC	Données …							

Remarque : La taille du tableau n'est qu'indicative, en ce qui concerne la partie des données ; le cas échéant, le candidat pourra bien évidemment le rallonger selon ses besoins.

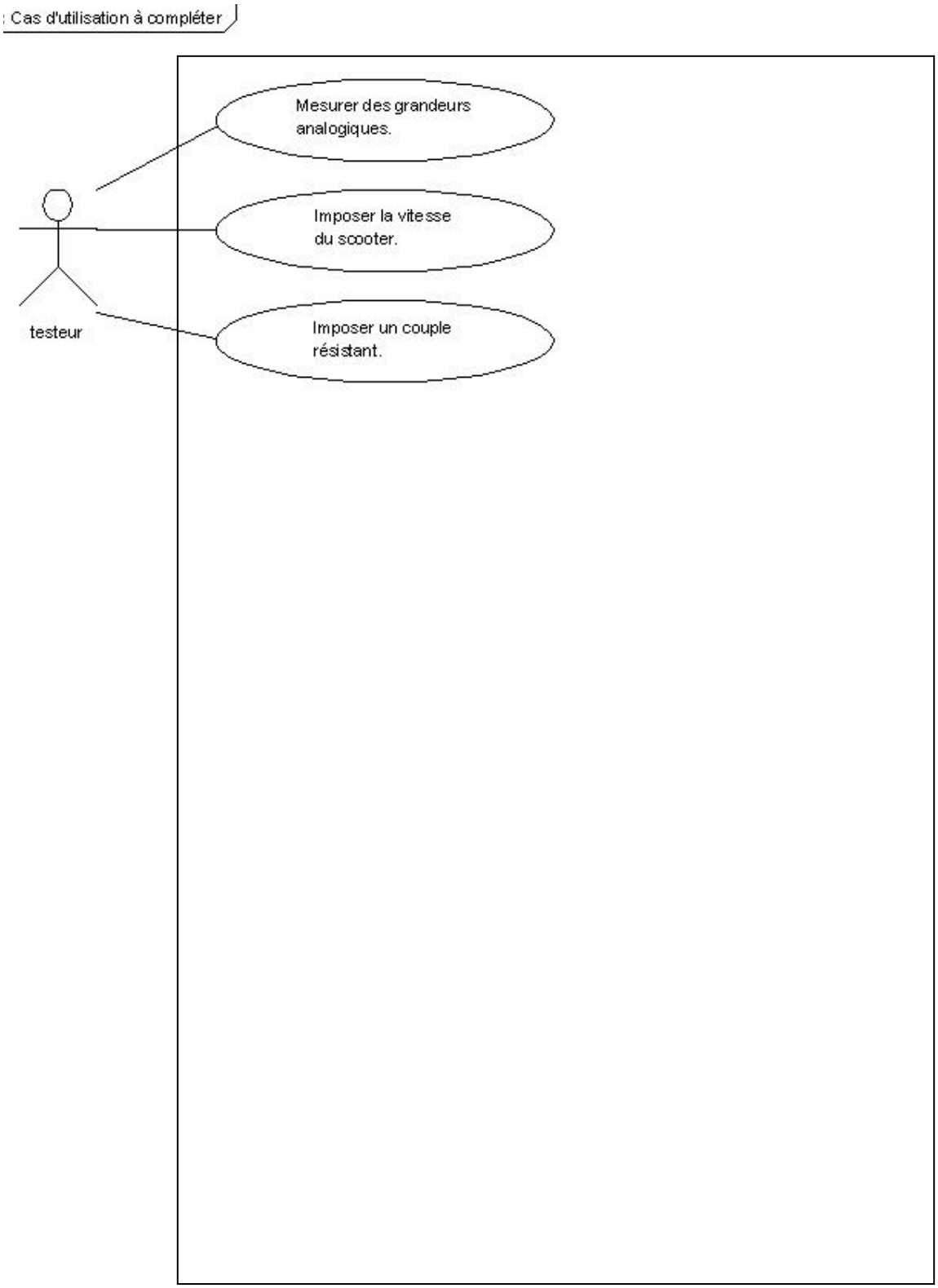
Question II.C.3

Adresse de l'esclave : ----------

Menu et paramètre concernés : ----------

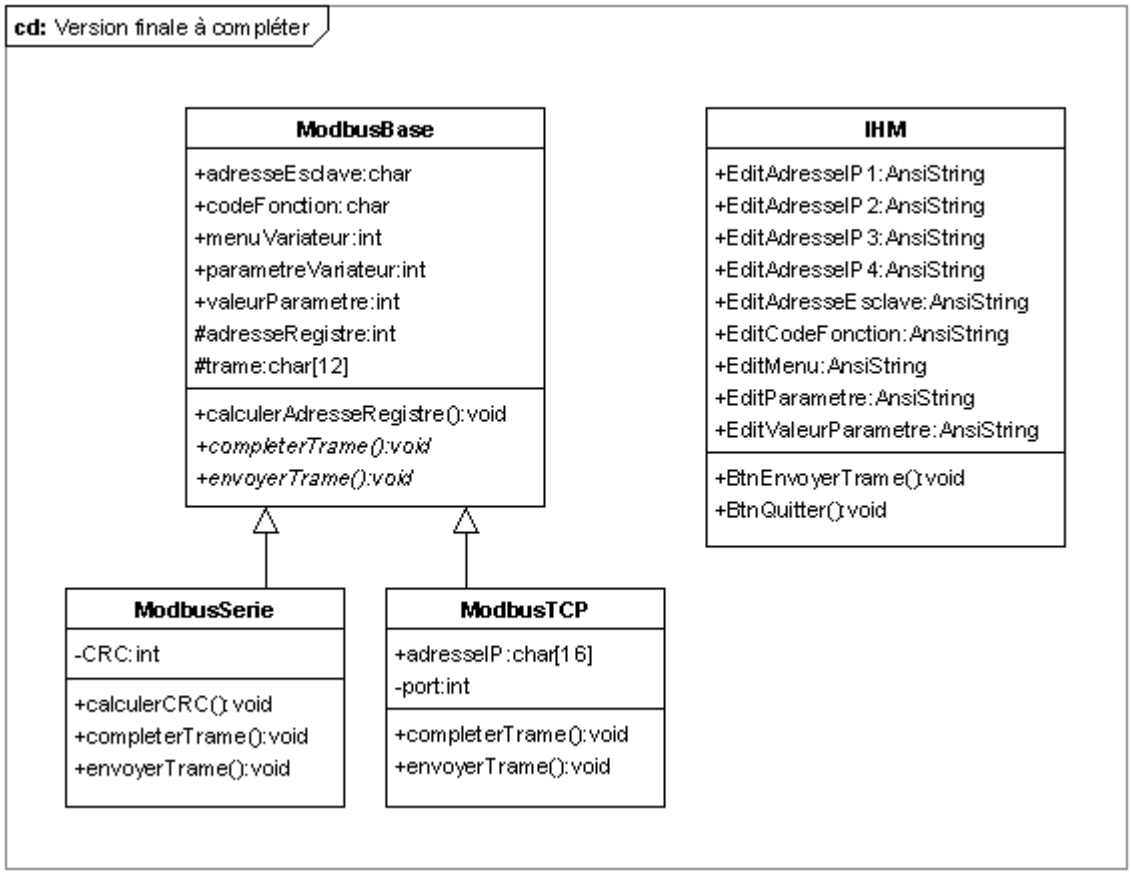
Opération réalisée : ----------

DR 2 - Questions III.A.3 et III.A.4



Rem : *en cas de manque de place pour écrire le contenu d'un cas d'utilisation, le candidat pourra le repérer puis expliciter clairement ce contenu sur sa copie.*

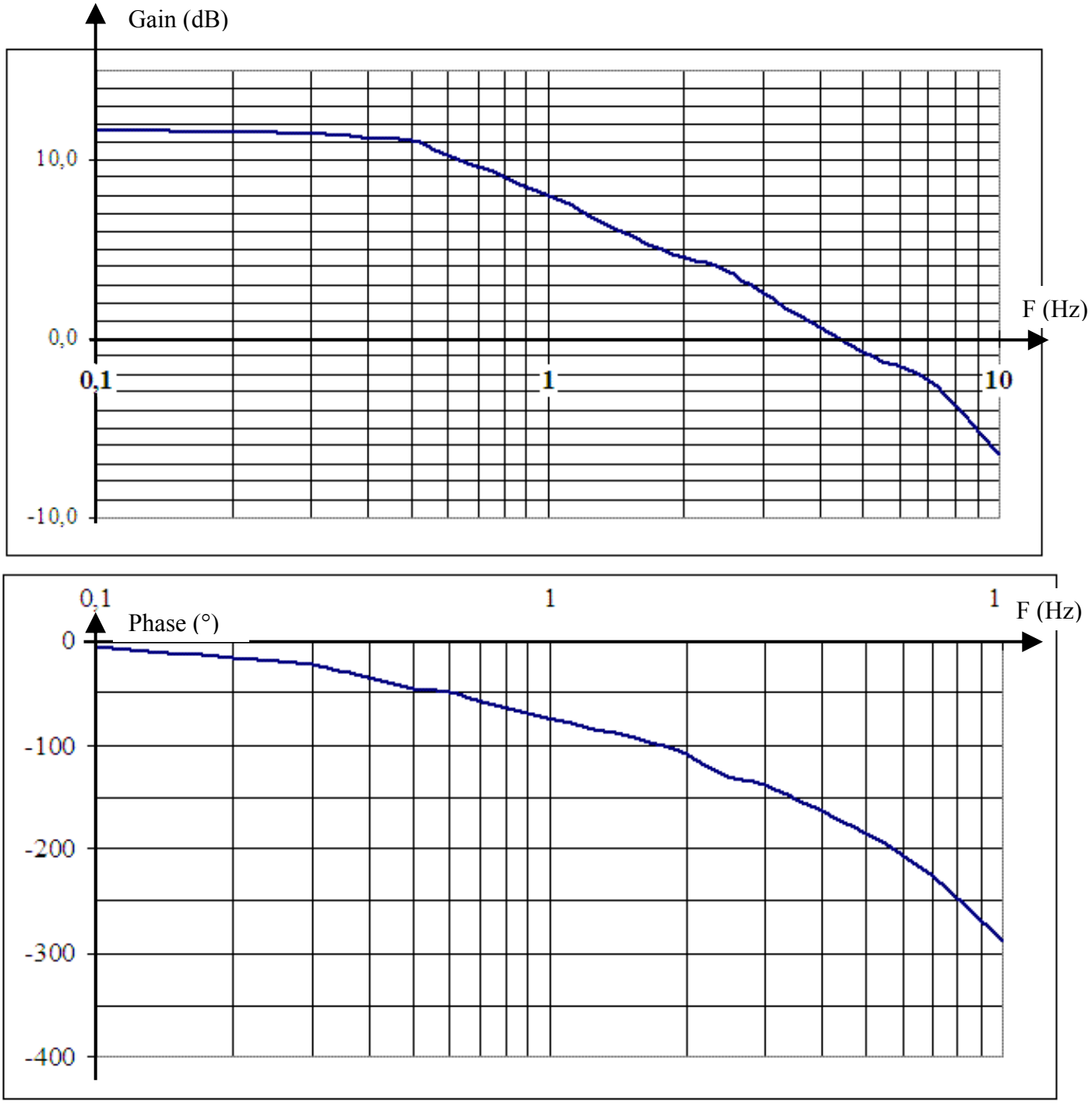
DR 3 - Question III.B.5.1



Type de liaison : ----------

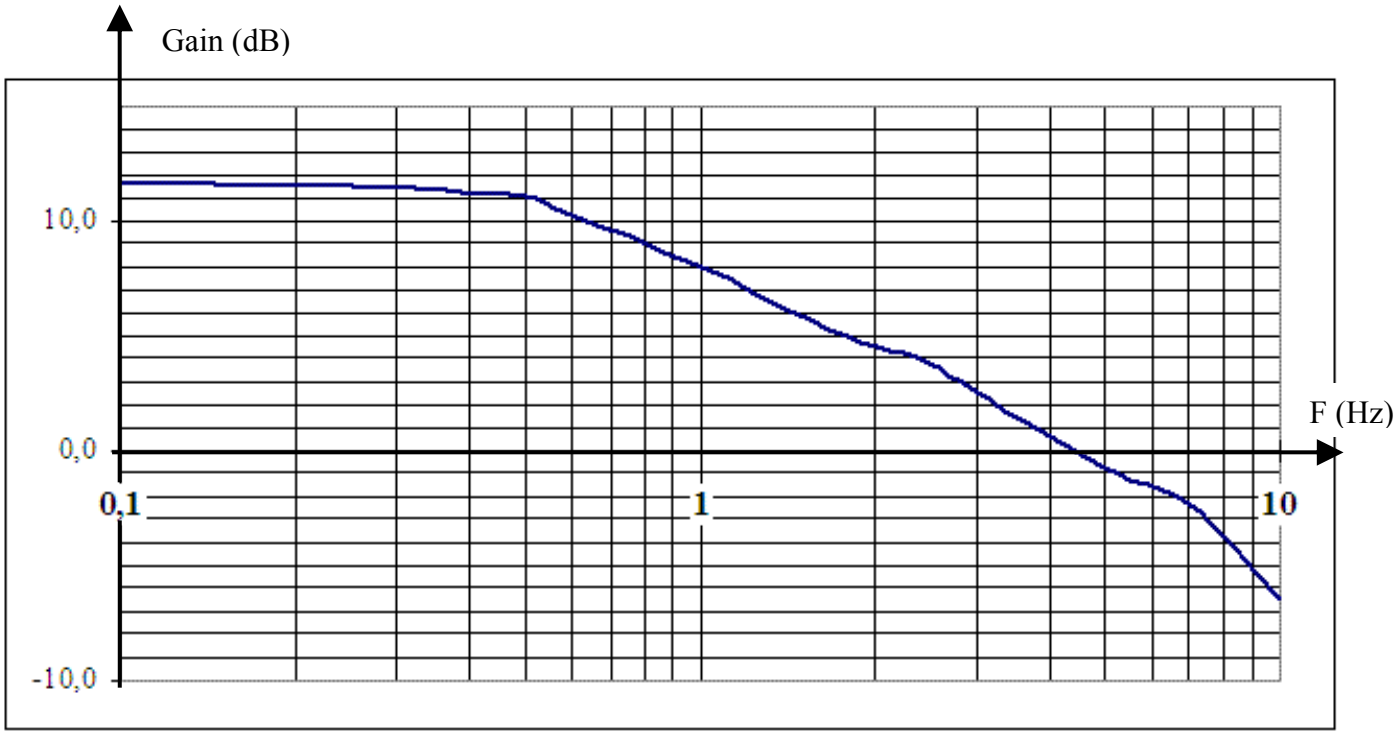
Justification : ----------

DR 4 - Questions IV.A.1 et IV.A.2



Au delà de 10 Hz, les mesures ne sont plus significatives (trop de bruit).

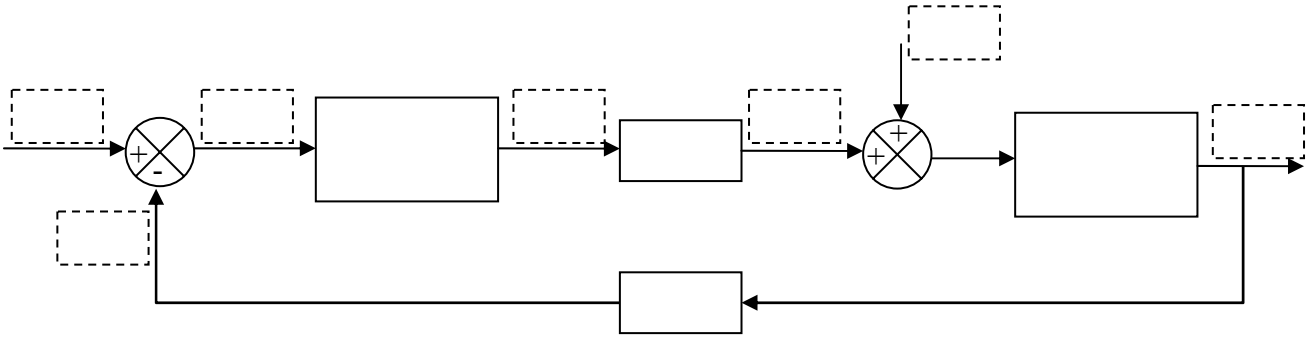
DR 5 - Questions IV.B.6 et IV.B.7



Au delà de 10 Hz, les mesures ne sont plus significatives (trop de bruit).

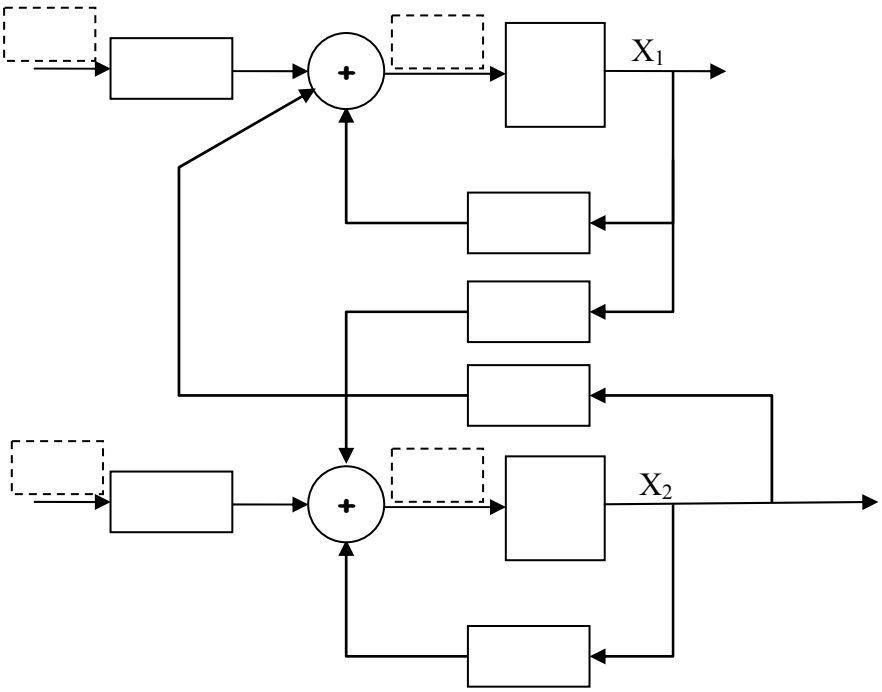
DR 6

Question IV.C.2



Les cases pointillées seront remplies par le nom d'un signal, les autres par une fonction de transfert ou un gain.

Questions IV.D.3, IV.D.4.a et IV.D.4.b



Les cases pointillées seront remplies par le nom d'un signal, les autres par une fonction de transfert ou un gain. Le couple résistant étant par essence même aléatoire, le candidat ne se formalisera pas sur son signe.

DR 7 – Question IV.E.2