Étude d'un Système Numérique et d'Information
Option A (IR) - Session 2016
Correction tv

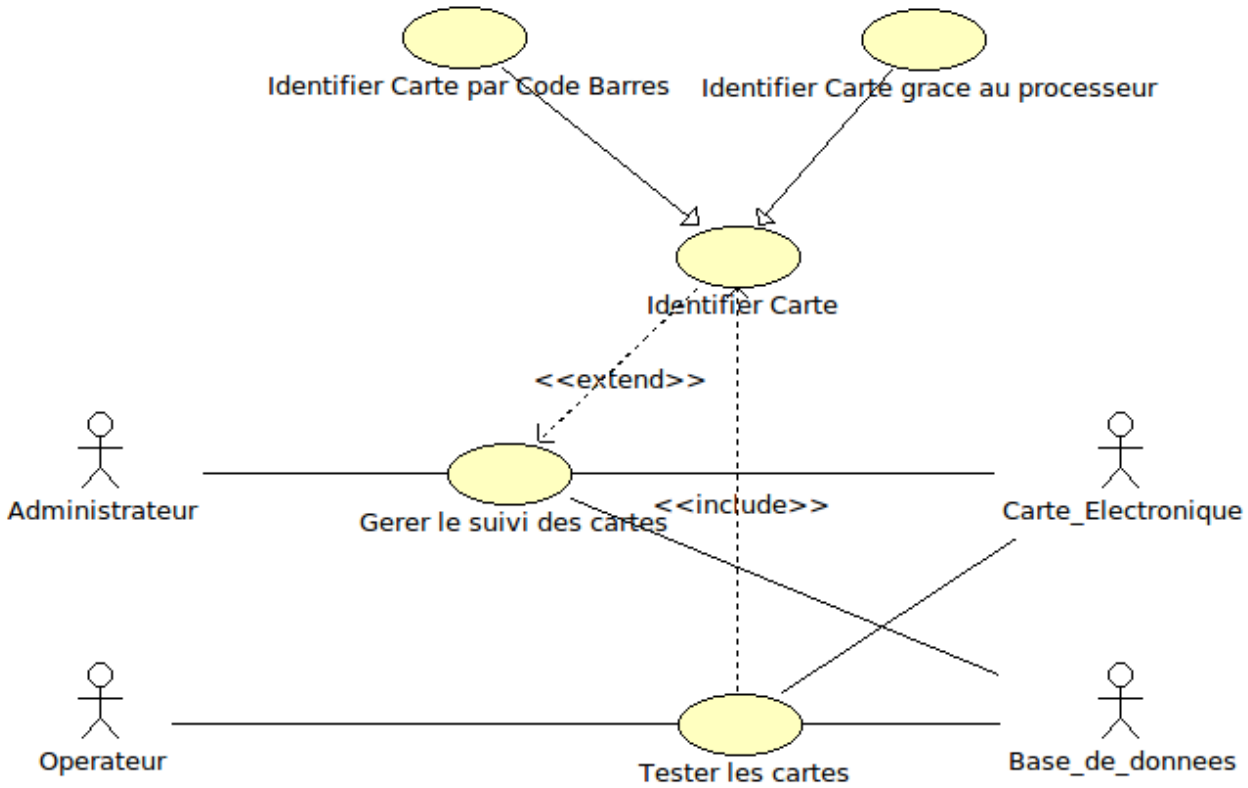
Test et Suivi de Cartes Électroniques

Partie A. Analyse du contexte

Q1.

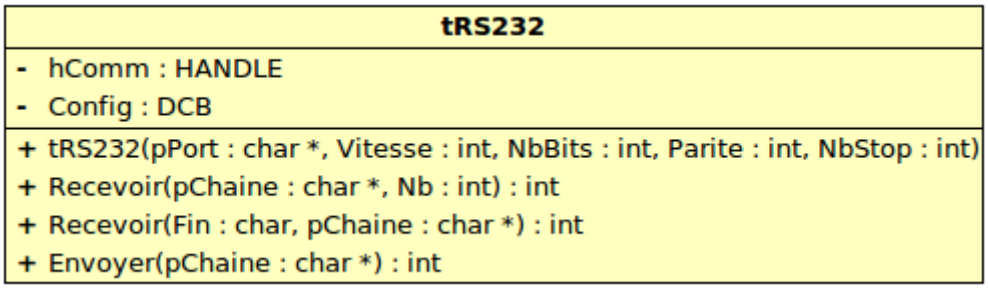
Critères	2of5	CODABAR	**Code 39**	**Code 128**	EAN-13	DataMatrix
Codage des lettres	non	non	**oui**	**oui**	non	oui
Codage des chiffres	oui	oui	**oui**	**oui**	oui	oui
Vérification	option	non	**option**	**oui**	oui	oui
Type 1D	oui	oui	**oui**	**oui**	oui	non 2D
Code barres retenu	non	non	**oui**	**oui**	non	oui (au delà des besoins)

Q2.



Partie B. Conception

Q3.



Q4.

```
class tLecteurCB
{
    private:
        char *chaineLue;
        tRS232 *liaisonCom;

    public:
        tLecteurCB(int numPortCom);
        void LireCB(char *CBLu);
};
```

Q5.

```
void tLecteurCB::LireCB(char *CBLu)
{
    liaisonCom->Recevoir('\r', CBLu);
}
```

Q6.

C'est **légal** car c'est une **surcharge**. On différencie les méthodes par leur signature, c'est-à-dire par **le type et/ou le nombre de ses arguments (ou paramètres)**.

Q7.

```
int Recevoir(char Fin, char *pChaine, int Nb);
```

Q8.

```
int tRS232::Recevoir(char Fin, char *pChaine, int Nb)
{
    char carRecu;
    int nbRecu = 0;
    do
    {
        carRecu = LireChar();
        pChaine[nbRecu++] = carRecu;
    }
    while(carRecu != Fin && nbRecu < Nb);
}
```

Partie C. Le bus de communication

Q9.

Le dialogue est toujours à l'initiative du maître : il envoie une demande et l'esclave répond. Les esclaves ne peuvent pas communiquer entre eux.

Q10.

RS485 : bus (multi-point jusqu'à 32 équipements) bidirectionnel (half-duplex)

Q11.

Le **BCC** (*Block Check Character*) permettra de **détecter les erreurs de transmission**.

Q12.

Trame requête = 20 caractères et Trame réponse = 20 caractères soit un total de **40 caractères**

1 caractère = 1 bit START + 7 bits de données + 1 bit de parité + 1 bits de STOP = **10 bits**

Débit = 500 kbits/s

Durée totale de transmission = ((20 + 20) x 10) / 500000 = 0,0008 s soit **0,8 ms**

Q13.

Trame émise avec l'adresse **"Y260"** :

ENQ	Adresse TEDI				Commande		ETX	BCC
0x05	'Y'	'2'	'6'	'0'	'D'	'T'	0x03	**0x11**
5	89	50	54	48	68	84	3	= 401%128 = **17**

Le BCC vaut **17 soit 0x11**.

Calcul du BCC :

```
#include <stdio.h>

unsigned char calculerBCC(char *trame, int nb)
{
    int i;
    unsigned char bcc = 0; // sur 8 bits

    for (i = 0; i < nb; i++)
    {
        bcc += trame[i]; // la somme fera 145 (401-256)
    }

    return (bcc%128);
}
```

```
int main()
{
    char trame [] = { 0x05, 'Y', '3', '7', '0', 'D', 'T', 0x03 };
    unsigned char bcc;

    bcc = calculerBCC(trame, 8);
    printf("BCC = 0x%02X\n", bcc); // Affiche 0x13

    return 0;
}
```

Partie D. La base de données

Q14.

SELECT codebarre FROM cartes WHERE dateajoutbase='2010-04-10'

Q15.

SELECT nom FROM cartes, emplacements, localisationcartes WHERE localisationcartes.idemplacement=emplacements.id AND cartes.id=localisationcartes.idnumcarte AND cartes.codebarre='HC125AA2-0004-0818-000691'

OU

SELECT nom FROM emplacements INNER JOIN localisationcartes ON localisationcartes.idemplacement=emplacements.id INNER JOIN cartes ON cartes.id=localisationcartes.idnumcarte WHERE cartes.codebarre='HC125AA2-0004-0818-000691'

Q16.

Découper le champ **codebarre** en au moins 3 champs :

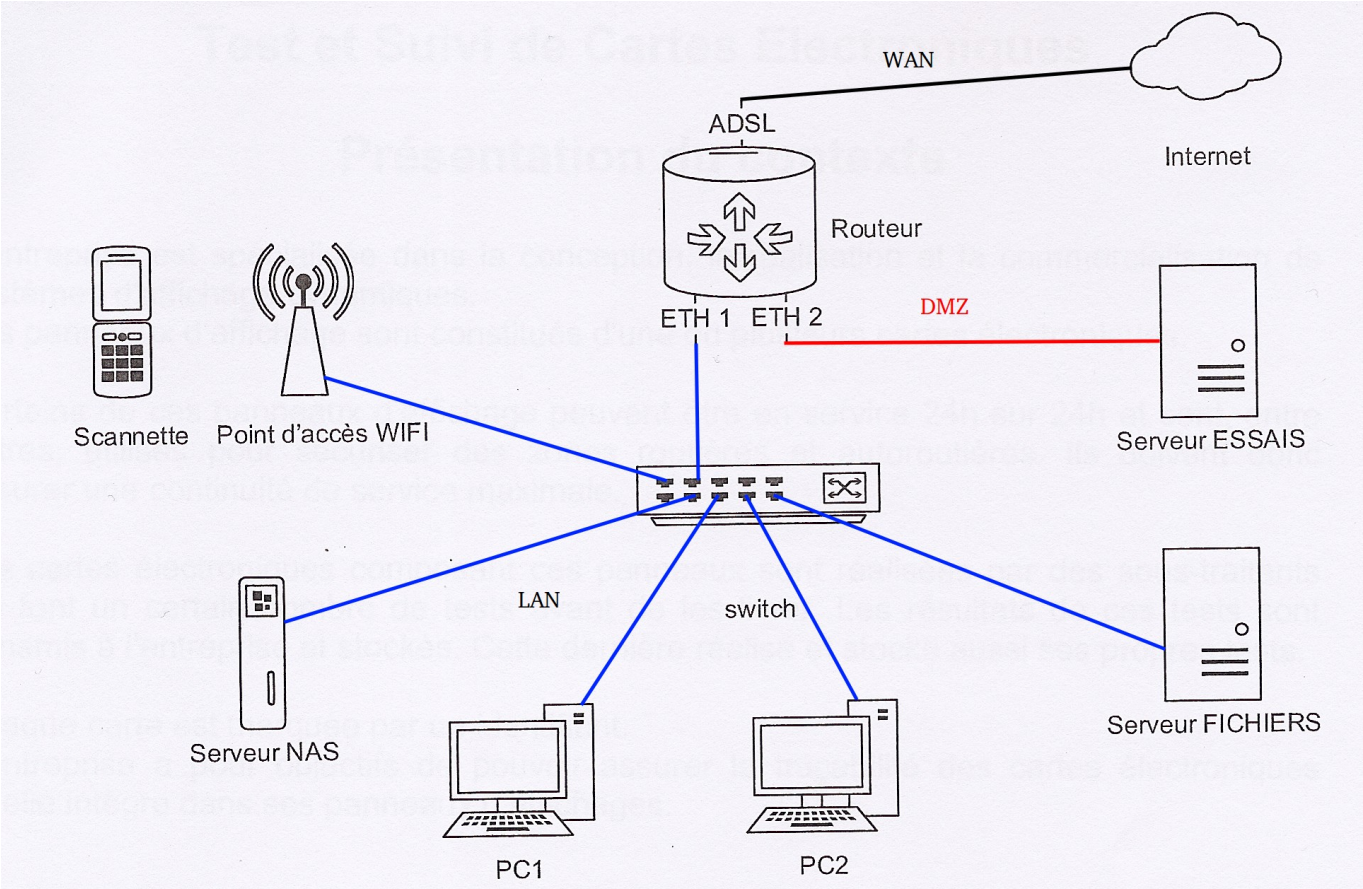
- code : HC125AA2-0004
- **numéro de lot** : 0818
- **numéro de carte** : 000691

```
CREATE TABLE `cartes` (
  `id` bigint(20) NOT NULL,
  `code` varchar(13) NOT NULL,
  `numerolot` varchar(4) NOT NULL,
  `numerocarte` varchar(6) NOT NULL,
  `commentaire` varchar(150) DEFAULT NULL,
  `datetest` date DEFAULT NULL,
  `dateajoutbase` date DEFAULT NULL,
  PRIMARY KEY (`id`)
);
```

Remarque : l'utilisation d'un id en AUTO_INCREMENT comme clé primaire n'est pas judicieux. Il serait préférable d'utiliser le code barre comme id en clé primaire. Dans ce cas, il serait impossible d'insérer plusieurs fois le même code barre dans la table cartes.

Partie E. Réseau

Q17.



Q18.

Une **DMZ** (*De-Militarized Zone*), appelée aussi réseau de service ou zone démilitarisée, est une zone considérée comme moins protégée que le réseau local de l'entreprise pour pouvoir y placer des serveurs publiquement accessibles de l'extérieur.

Q19.

Réseau local : 192.168.1.0/24
DMZ : 192.168.2.0/24

	Adresse IP	Masque de sous réseau	Passerelle par défaut
Routeur interface ADSL	80.86.125.34	/23	80.86.124.1
Routeur interface ETH1	**192.168.1.1**	**/24**	X (*)
Routeur interface ETH2	**192.168.2.1**	**/24**	X (*)
Serveur ESSAIS	**192.168.2.2**	**/24**	**192.168.2.1**
Serveur FICHIERS	**192.168.1.2**	**/24**	**192.168.1.1**
Serveur NAS	**192.168.1.3**	**/24**	**192.168.1.1**
PC1	**192.168.1.4**	**/24**	**192.168.1.1**
PC2	**192.168.1.5**	**/24**	**192.168.1.1**
Point Acces Wifi	**192.168.1.6**	**/24**	**192.168.1.1**
Scannette Wifi	**192.168.1.7**	**/24**	**192.168.1.1**

Q20.

Nombre de sous-réseaux : 2 (adminstratif et production) → **1 bit** nécessaire pour la partie subnet
Nombre de bits de la partie host : (32 - 24) – 1 = **7 bits**
Masque de sous-réseau : **255.255.255.128** ou **192.168.1.0/25**

Q21.

Nombre maximal d'hôtes par sous-réseau : $2^7 - 2$ = **126 hôtes**/sous-réseau

Q22.

Réseau n°1 : 192.168.1.0/25 **Broadcast : 192.168.1.127**
Réseau n°2 : 192.168.1.128/25 **Broadcast : 192.168.1.255**

Q23.

Le **DNS** (*Domain Name System*) est un service permettant de traduire un nom de domaine en adresses IP notamment.

Le **DHCP** (*Dynamic Host Configuration Protocol*) est un protocole réseau qui permet de configurer automatiquement l'adressage IP d'un hôte en lui affectant au moins une adresse IP et un masque de sous-réseau.

Q24.

Elle indique l'**interface** et le **port d'écoute** du serveur.

Q25.

```
Listen 192.168.1.22:80
```