NORMALISATION DES LANGAGES DE PROGRAMMATION

des

Automates Programmables Industriels

CEI 61131-3

Jean-Jacques DUMÉRY

Plan de la présentation

- Présentation générale de la norme
- Les objectifs de la norme CEI 61131
- Les notions de base et les concepts importants
- les éléments communs aux différents langages
- Les langages de programmation

  Quelques exemples d ’utilisation
- Conclusions

Les différentes parties de la norme

Elles s ’appliquent aux automates programmables et aux périphériques associés tels que :

- Les outils de programmation et de mise au point
- Les équipements de test
- les interfaces homme - machine

Les cinq parties de la CEI 61131

- 1ère partie : Informations générales
- 2ème partie : Spécifications et essais des équipements
- 3ème partie : Langages de programmation
- 4ème partie : Guide pour l 'utilisateur
- 5ème partie : Communications

Les premières références internationales

CEI 61131-1 et -2 : octobre 1992

CEI 61131-3 : mars 1993

CEI/TR3 61131-4 : mars 1995

CEI 62231-5 : août 1999

version préliminaire avant publication

Voir http://www.iec.ch

Les premières références européennes

NF EN 61131-1 : septembre 1994

NF EN 61131-2 : octobre 1996

NF EN 61131-3 : novembre 1993

Elles comportent en plus des CEI une annexe normative (correspondances normes européennes et internationales)

Voir http://www.afnor.fr

Les objectifs de la norme
(trois premières parties)

- Donner les définitions et identifier les principales caractéristiques permettant de sélectionner et d 'utiliser les A.P.
- Spécifier les prescriptions électriques, mécaniques et fonctionnelles ainsi que les méthodes de test et les procédures à suivre pour vérifier la conformité avec ces prescriptions
- Spécifier la syntaxe, la sémantique et la représentation des langages de programmation devant être utilisés pour les A.P.

Quelques définitions

- **Pour une configuration d ’A.P. :**

- (procédure d ’) arrêt de sécurité,

- reprise à froid, à chaud et immédiate …

- **pour les langages :**

- un délimiteur,

- un double mot, un mot long,

- une instance,

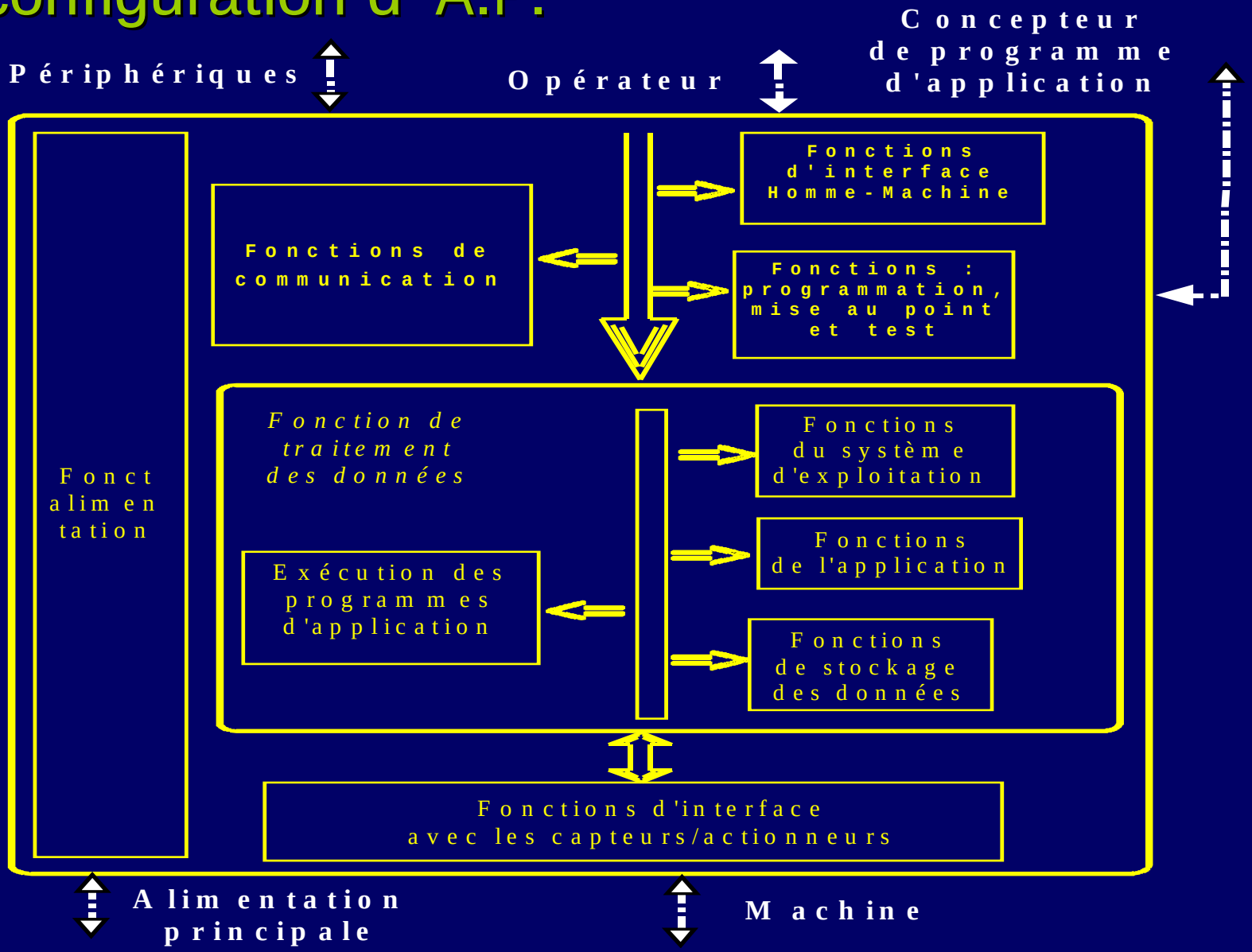
- une variable globale,

- un libellé,

- une donnée non volatiles,

- un champ d ’application ...

Structure fonctionnelle de base d ’une configuration d ’A.P.



Partie 3 : langages de programmation

Notions de base

- MODULES LOGICIELS

(Program organization units)

- le PROGRAMME (Program)
- le BLOC FONCTIONNEL (Function Block)
- la FONCTION (Function)

- LES LANGAGES DE PROGRAMMATION

(dans lesquels les modules peuvent être écrits)

La fonction

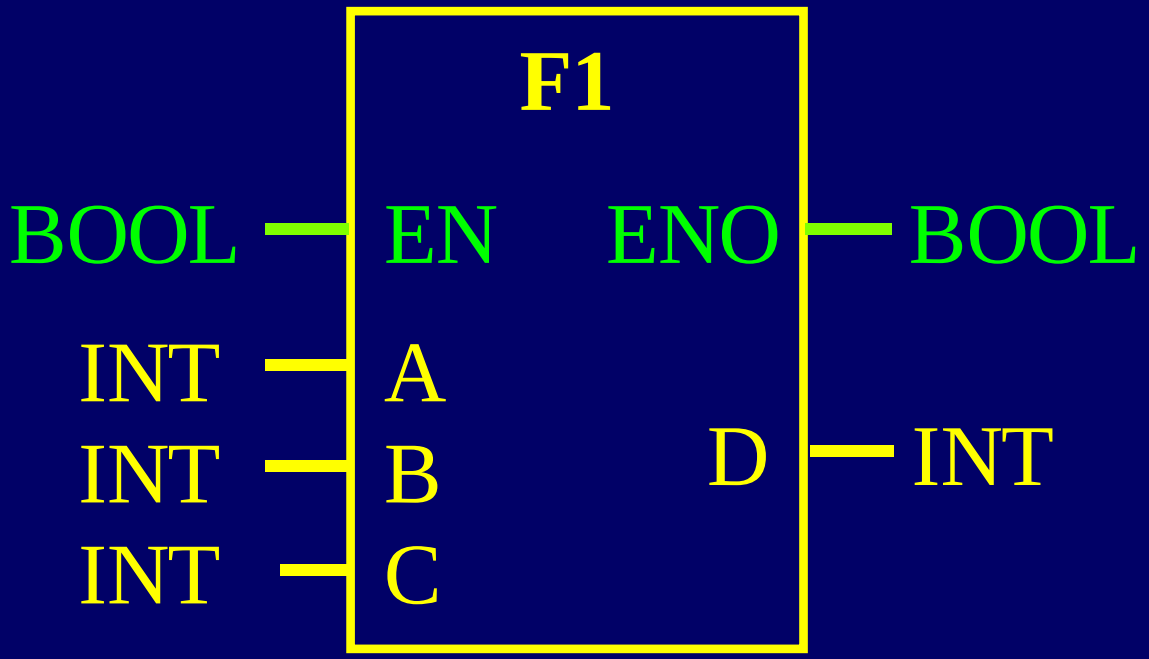
- Module logiciel ayant :
  - plusieurs variables d ’entrée possibles,
  - une seule variable de sortie,
  - pas de mémoire interne,
  - parfois une entrée EN (validation) et une sortie ENO (pas d ’erreur).

Exemples de fonctions

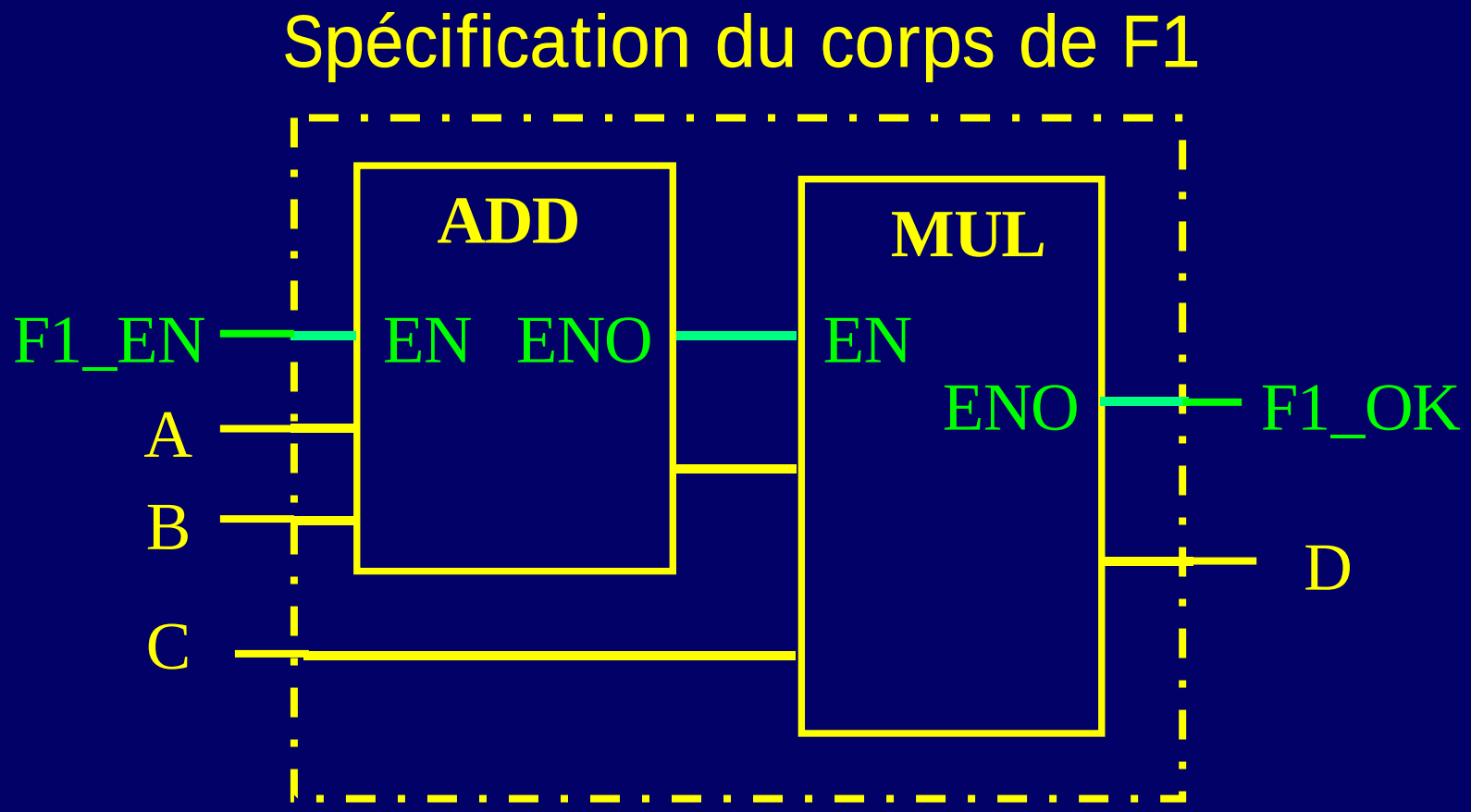
- fonctions de conversion de type,
- fonctions arithmétiques,
- fonctions sur chaînes de bits,
- fonctions sur chaînes de caractères,
- fonctions de sélection et comparaison,
- ...

Exemple de déclaration de fonction

Spécification externe de F1



Exemple de déclaration de fonction



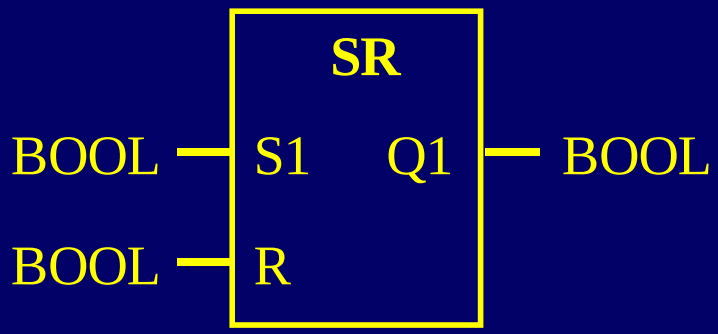
Le bloc fonctionnel

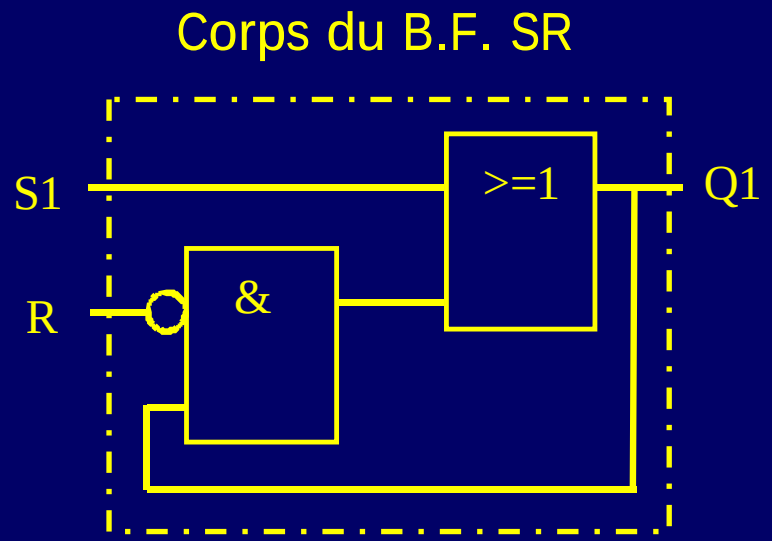
- Module logiciel ayant :
  - plusieurs variables de sortie possibles,
  - une mémoire interne.

Exemples de blocs fonctionnels

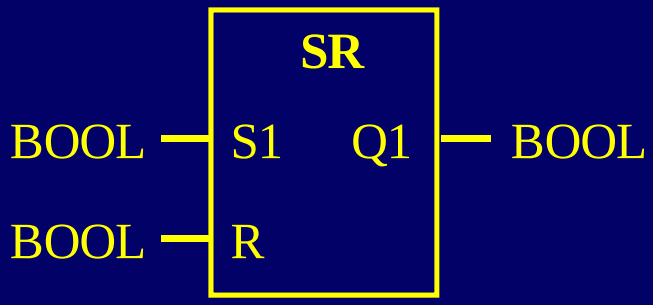
- mémoires,
- détection de fronts,
- compteurs, temporisations,
- blocs de communication,
- ...

Exemple de bloc fonctionnel standard

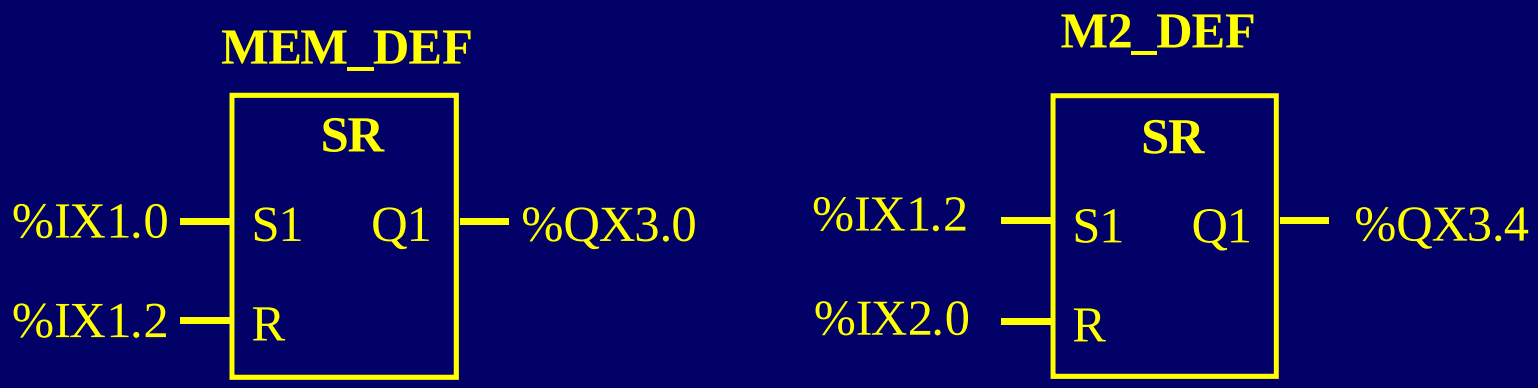




Bloc fonctionnel instancié



Il est possible de créer plusieurs instances d 'un même B.F.
(dans un programme ou un autre B.F.)



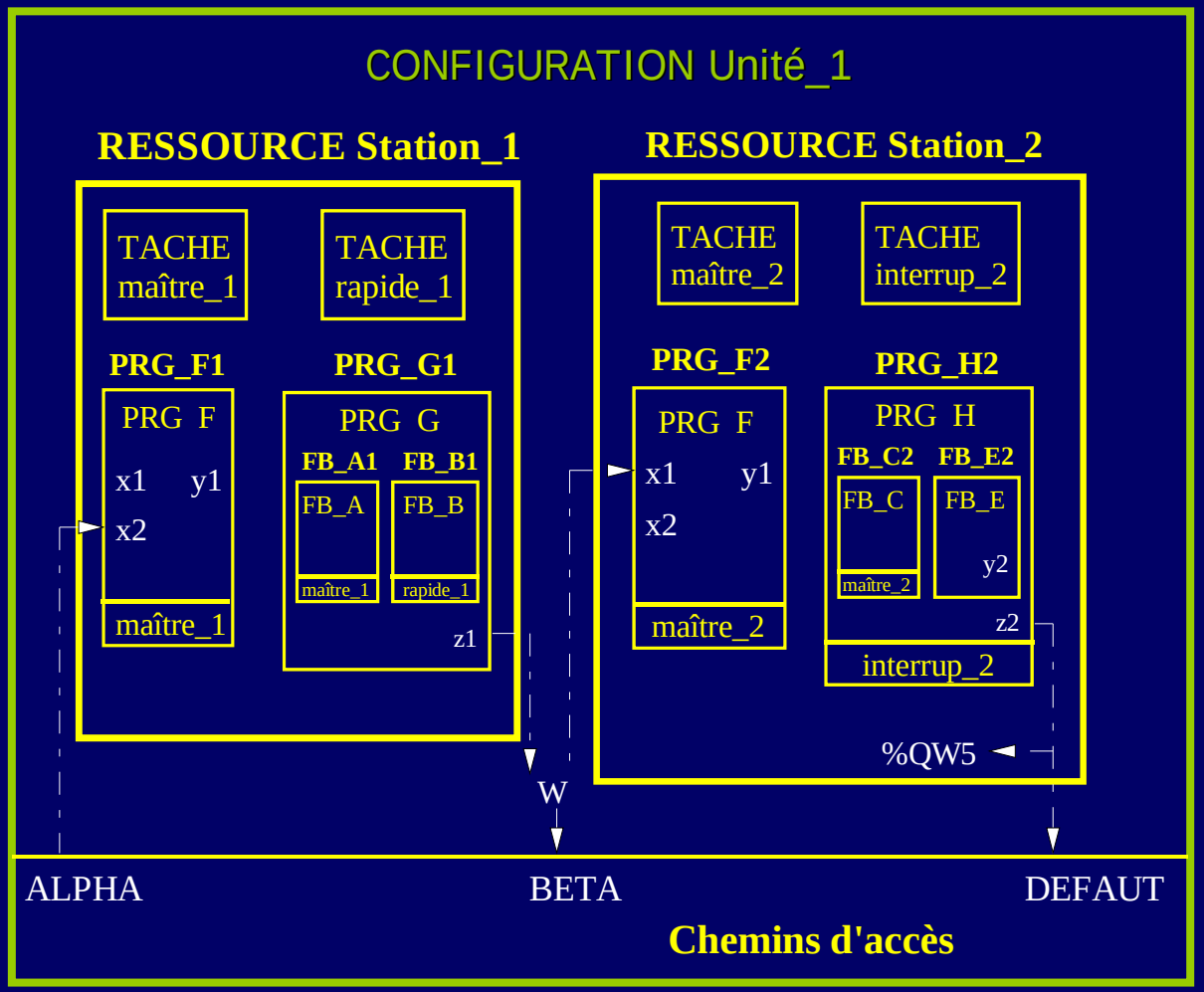
Le programme

- Module logiciel construit à l ’aide de :
  - fonctions,
  - et blocs fonctionnels.

Les programmes ne peuvent être instanciés que dans des RESSOURCES

Des VARIABLES GLOBALES pourront être déclarées

La configuration logicielle



Les éléments communs aux différents langages

- Les identificateurs ARRET_TECHN, RETOUR_OK
- Les mots clés FUNCTION, END_FUNCTION_BLOCK
- Les commentaires (*production normale*)
- Les libellés : +234, 16#E0, ‘ ARRET ’, TIME#2.7s
  numériques, de chaînes de caractères, de datation et de temps,
- Les types de données
- Les variables

Les types de données, exemples

BOOL	Booléen	1 bit
BYTE	Chaîne de bits de longueur 8	8 bits
WORD	Mot	16 bits
DWORD	Mot double	32 bits
LWORD	Mot long	64 bits
INT	Entier	16 bits
UINT	Entier non signé	16 bits
UDINT	Entier double non signé	32 bits

Les variables à un seul élément

PREFIXE	SIGNIFICATION
I	Emplacement d'entrée
Q	Emplacement de sortie
M	Emplacement de mémoire
X	Taille d'un seul bit
Aucun	Taille d'un seul bit
B	Taille d'un octet (8 bits)
W	Taille d'un mot (16 bits)
D	Taille d'un double mot (32 bits)
L	Taille d'un mot long (64 bits)

Représentation des variables à un seul élément

La représentation directe d'une variable à un seul élément est assurée par l ’enchaînement :

- du signe "**%**",
- d'un préfixe d'emplacement,
- d'un préfixe de taille,
- et d'un ou plusieurs entiers non signés
- séparés par le symbole "**.**"

Exemples : %I2.0, %Q3.2, %MD25

Les langages de programmation

- Les langages littéraux :
  - IL liste d ’instructions,
  - ST langage littéral structuré.
- Les langages graphiques :
  - LD langage à contacts,
  - FBD langage à blocs fonctionnels.
- Le langage SFC

Le langage IL

Etiquette	Opérateur	Opérande	Commentaire
MV1 :	LD	%IX1	(*Etiquette non oblig.*)
	AND N	%MX5	
	ST	%QX2	(*Marche ventilateur*)

Des fonctions et des blocs fonctionnels peuvent être lancés en I L

Le langage ST

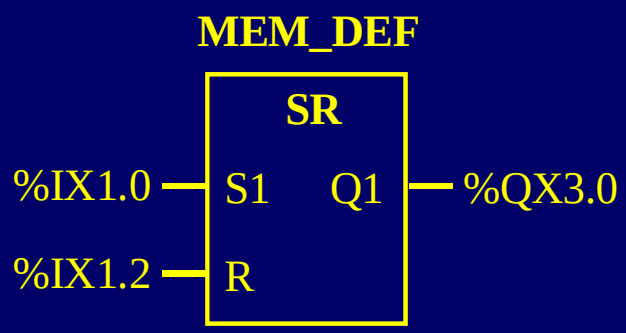
Le langage littéral structuré ST utilise :

- des expressions (E<F) AND NOT C
- et des énoncés
  - les énoncés d ’affectation, C := C + 1 ;
  - les énoncés de sélection, IF ... THEN ... ELSE ..., CASE
  - les énoncés d ’itération, FOR ... TO ..., WHILE ... REPEAT ...
  - les énoncés de commande.

de fonctions et B.F.

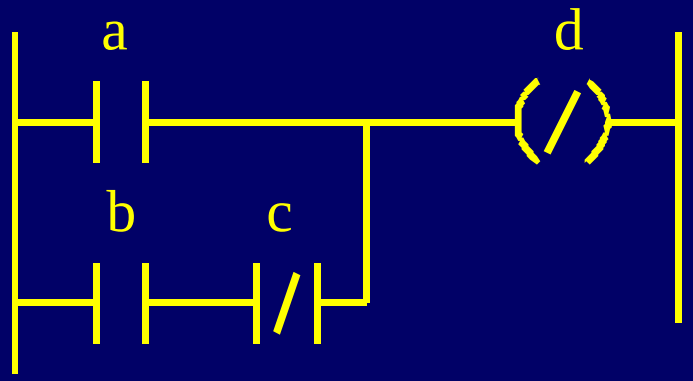
Le langage ST

Exemple d ’un énoncé de commande

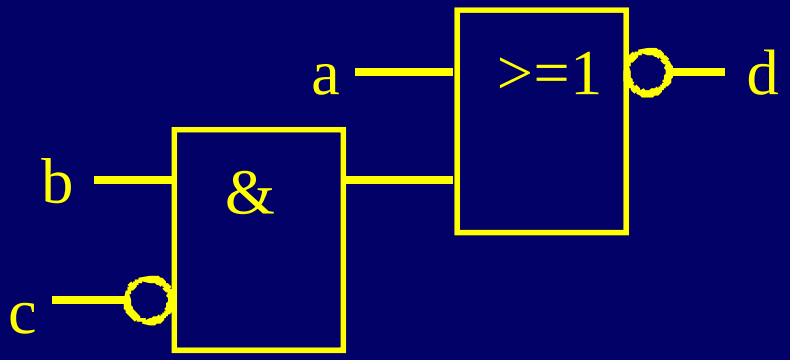



```
(*déclaration*)
VAR MEM_DEF : SR; END VAR
(*exécution*)
MEM_DEF (S1 := %IX1.0,  R := %IX1.2);
(*affectation*)
%QX3.0 := MEM_DEF.Q1;
```

Les langages graphiques LD et FBD

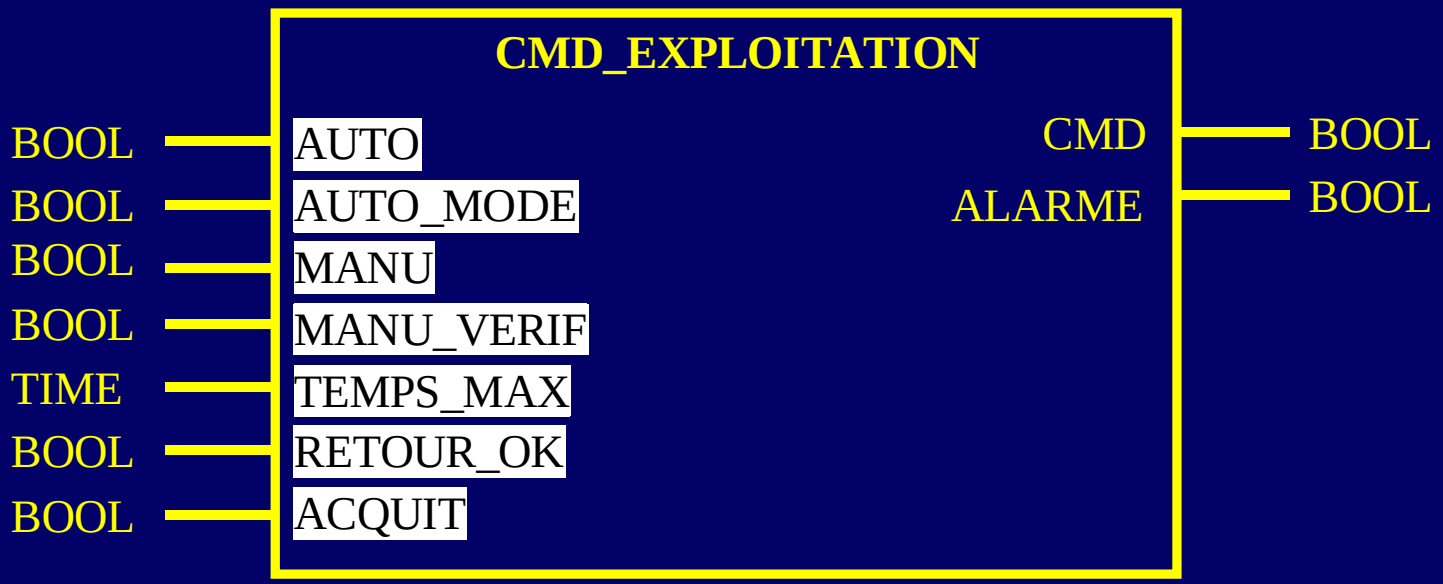


Langage LD



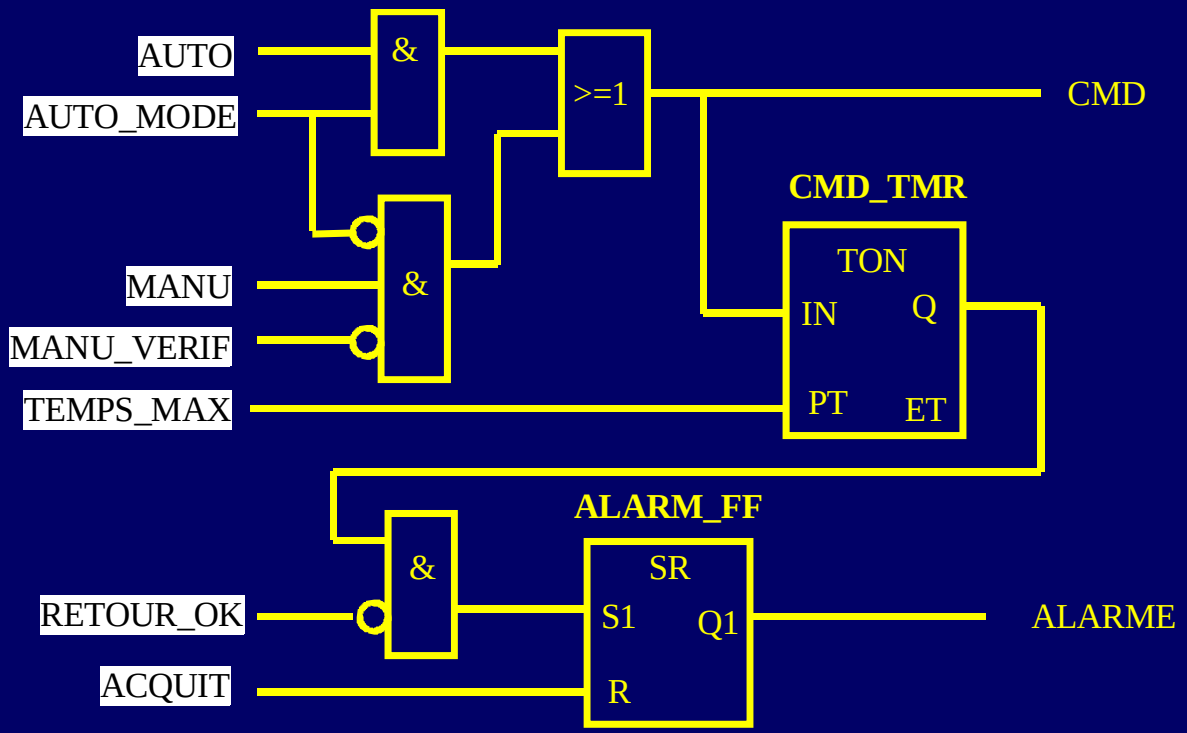
Langage FBD

Langage FBD, exemple



Description externe

Langage FBD, exemple



Description interne

Le langage SFC

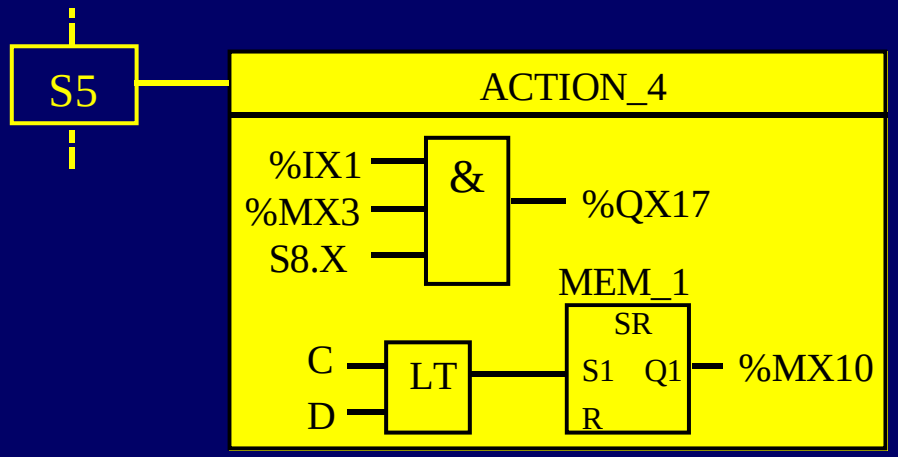
Il est destiné à être utilisé pour la structuration de l ’organisation interne d ’un module logiciel

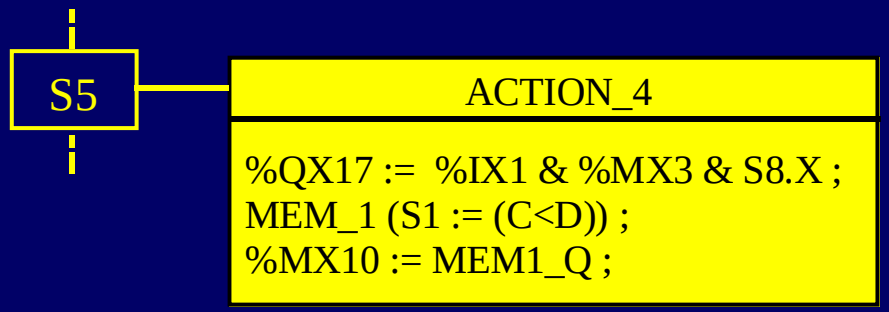
dans le but d ’assurer :

des fonctions de commande séquentielle

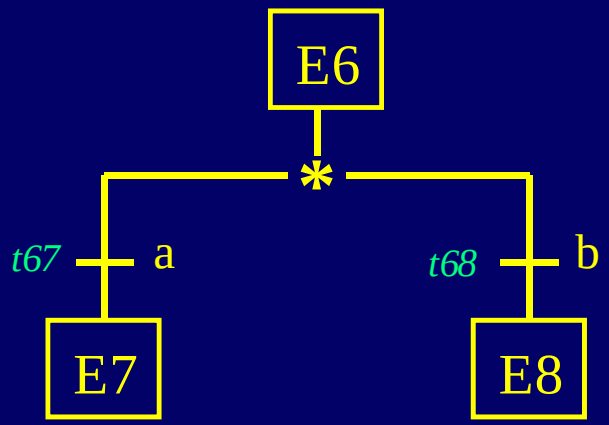
Le langage SFC, remarques

Tous les langages peuvent être utilisés dans les blocs d 'actions





Le parallélisme interprété est exclu



CONCLUSIONS

La norme CEI 61131-3 répond à une attente des utilisateurs concernant les langages de programmation des API :

- harmonisation des vocabulaires utilisés,
- notions et concepts de base s'appuyant sur une norme,
- syntaxe et sémantique des langages les plus indépendants possibles d'un constructeur d'API donné,
- facilité de mise en œuvre de principes tels que structuration et modularité des programmes,
- **Possibilité de définir ses propres blocs fonctionnels « utilisateur »**

Nécessité d 'une spécification structurée en amont de la phase de codage