IEC 61131-3 Tutoriel

Harmoniser la manière de voir le contrôle

Le futur est là

Eelco van der Wal
Managing Director PLCopen
Trad.: J.M. Boissard

Fiction?

Imaginez …

*** Vous travaillez dans le domaine du contrôle industriel**

*** Avec 4 marques de manufacturiers**

*** Chacun utilisant un dialecte différent pour chacun de ses langages**

- **Luttant pour harmoniser vos programmes entre vos programmeurs, vos ingénieurs électrique et votre personnel d'entretien dans l'usine**
- *** et découvrant que vos concurrents font mieux que vous**

Pourquoi? Qu'est-ce qui cloche ?

C’est la jungle !!!!!!

Tous ces problèmes peuvent être résolus en grande partie par un standardisation

... Et un tel standard existe

IEC 61131-3

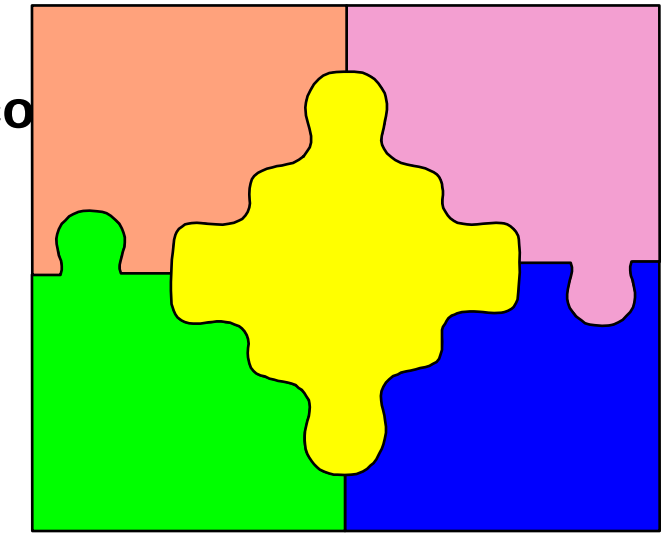
“La meilleure chose qui pouvait arriver dans le contrôle industriel”

“The best thing that happened to industrial control”

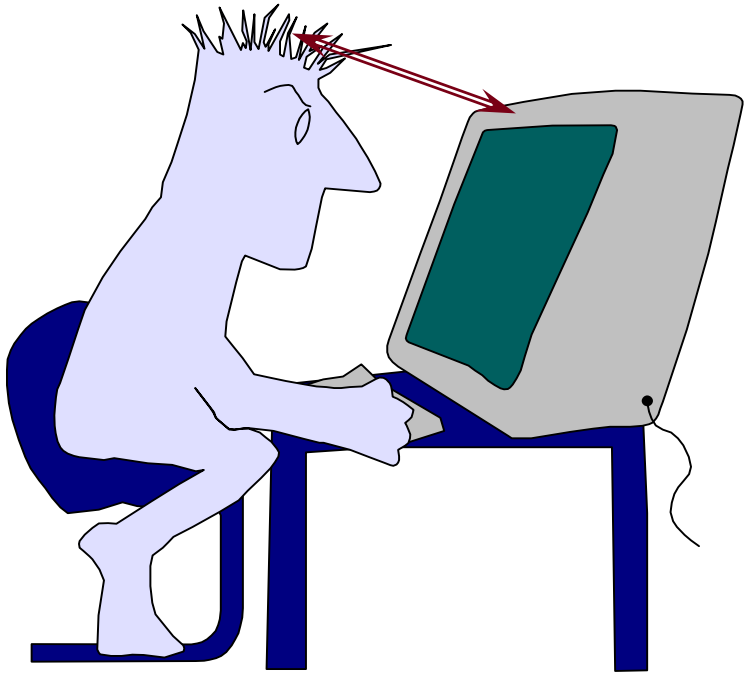
Sugar Lantic on Automation Maillist

Les 7 parties du standard IEC 61131

- 1 Présentation générale, définitions
- 2 Quincaillerie (Hardware)
- 3 Langages de programmation
- 4 Guides d'utilisation
- 5 Spécifications du service de messagerie (co
- 7 Logique floue (*Fuzzy Logic*)
- 8 Guides d'implémentation



IEC 61131-3 Langages de programmation
/
Programmation du contrôle industriel



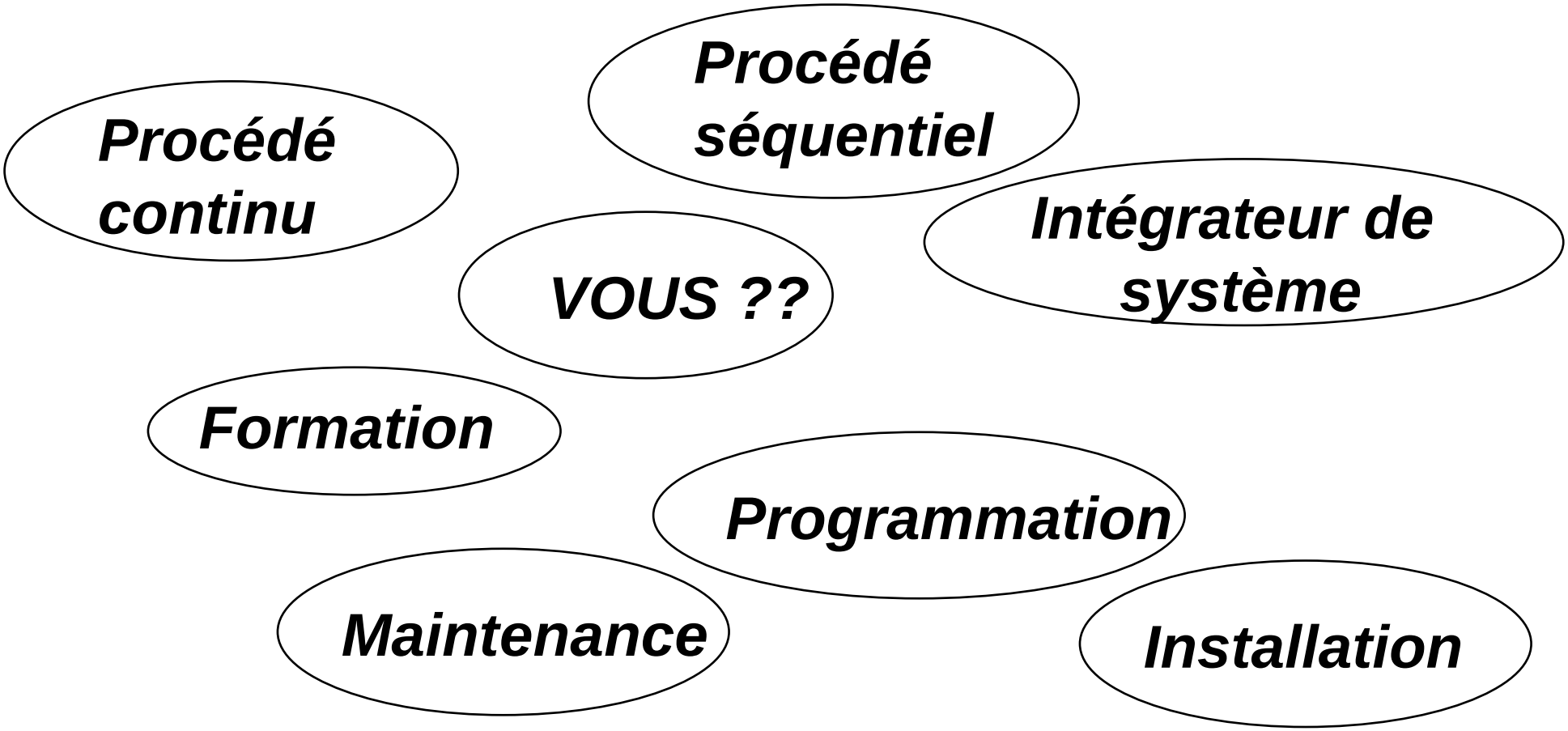
L’interface entre le programmeur et le système de contrôle

IEC 61131-3 Langages de programmation
/
Programmation du contrôle industriel

...avec le support pour les gens
...avec chacun leur approche

Quels sont les avantages d'un tel standard ?

Utilisateur? Quels utilisateurs?



Utilisateur? Quels utilisateurs?

- **Lignes de production automobile**
- **Usine d'épuration**
- **Industrie Agroalimentaire et conditionnement**
- **Fabrication de câble**
- **Stérilisation de procédés pharmaceutique ou de fabrication de semi-conducteurs**
- **Manège de parc d'attraction**
- **Usine de traitement de déchets radioactifs**

Cette grande diversité exige beaucoup de différentes compétences, de différentes approches

Quels sont les avantages d'un tel standard ?

- Réduire le gaspillage de ressources humaines (dans la formation, le déverminage, la maintenance et la documentation)

Quels sont les avantages d'un tel standard ?

- **Réduire le gaspillage de ressources humaines (dans la formation, le déverminage, la maintenance et la documentation)**

Se concentrer sur la solution du problème via la conception de programmes modulaires ré-utilisables (Réduction de l'investissement pour l'application et de la dépendance aux fournisseurs)

Quels sont les avantages d'un tel standard ?

- **Réduire le gaspillage de ressources humaines (dans la formation, le déverminage, la maintenance et la documentation)**
- **Se concentrer sur la solution du problème via la conception de programmes modulaires ré-utilisables (Réduction de l'investissement pour l'application et de la dépendance aux fournisseurs)**

Réduire les erreurs d'interprétation et de compréhension

Quels sont les avantages d'un tel standard ?

- **Réduire le gaspillage de ressources humaines (dans la formation, le déverminage, la maintenance et la documentation)**
- **Se concentrer sur la solution du problème via la conception de programmes modulaires ré-utilisables (Réduction de l'investissement pour l'application et de la dépendance aux fournisseurs)**
- **Réduire les erreurs d'interprétation et de compréhension**

Réutiliser des techniques de programmation dans différents environnements (Contrôle industriel général)

Quels sont les avantages d'un tel standard ?

- **Réduire le gaspillage de ressources humaines (dans la formation, le déverminage, la maintenance et la documentation)**
- **Se concentrer sur la solution du problème via la conception de programmes modulaires ré-utilisables (Réduction de l'investissement pour l'application et de la dépendance aux fournisseurs)**
- **Réduire les erreurs d'interprétation et de compréhension**
- **Réutiliser des techniques de programmation dans différents environnements (Contrôle industriel général)**

Combiner harmonieusement différents composants, et procédures de différents projets, locations, compagnies ou pays

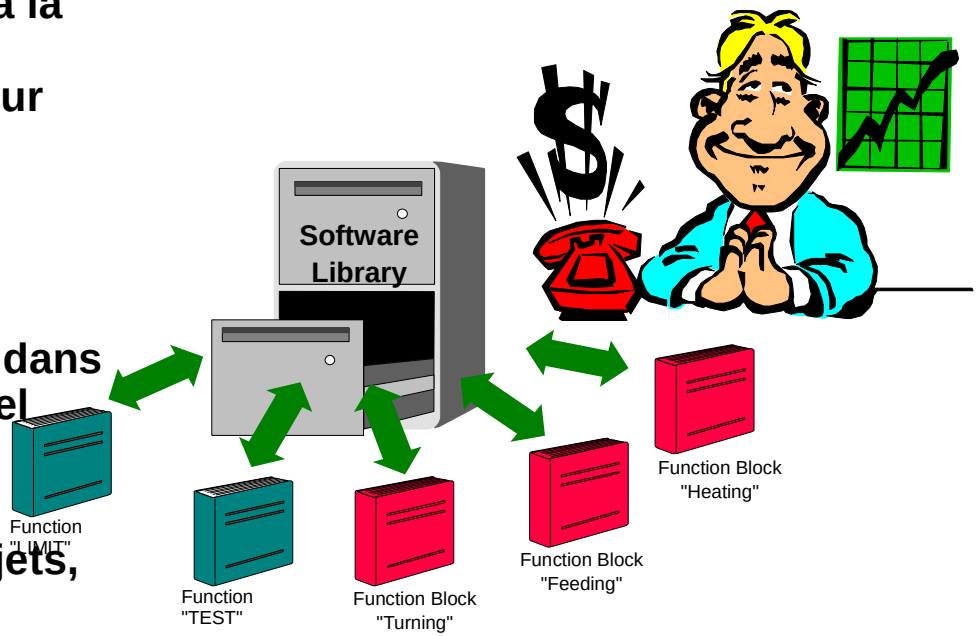
Quels sont les avantages d'un tel standard ?

- **Réduire le gaspillage de ressources humaines (dans la formation, le déverminage, la maintenance et la documentation)**
- **Se concentrer sur la solution du problème via la conception de programmes modulaires ré-utilisables (Réduction de l'investissement pour l'application et de la dépendance aux fournisseurs)**
- **Réduire les erreurs d'interprétation et de compréhension**
- **Réutiliser des techniques de programmation dans différents environnements (Contrôle industriel général)**
- **Combiner harmonieusement différents composants, et procédures de différents projets, locations, compagnies ou pays**

Accroitre l'interconnectivité des procédés (protection de l'investissement)

Quels sont les avantages d'un tel standard ?

- **Réduire le gaspillage de ressources humaines (dans la formation, le déverminage, la maintenance et la documentation)**
- **Se concentrer sur la solution du problème via la conception de programmes modulaires ré-utilisables (Réduction de l'investissement pour l'application et de la dépendance aux fournisseurs)**
- **Réduire les erreurs d'interprétation et de compréhension**
- **Réutiliser des techniques de programmation dans différents environnements (Contrôle industriel général)**
- **Combiner harmonieusement différents composants, et procédures de différents projets, locations, compagnies ou pays**
- **Accroître l'inter connectivité des procédés (protection de l'investissement)**



Les Avantages clés de IEC 61131-3

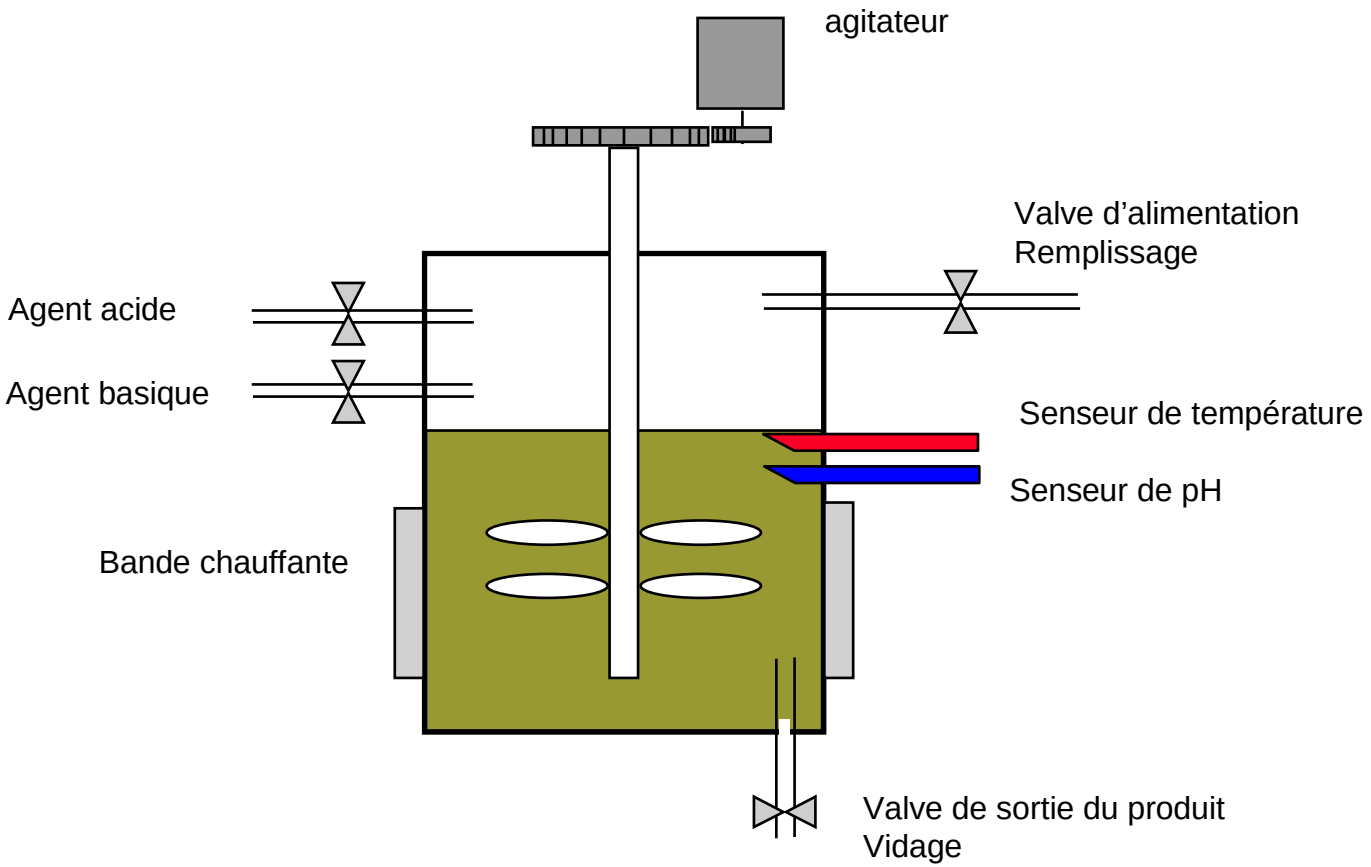
- **Programmes structurés** – *par l'utilisation de configurations, ressources et des unités d'organisation de programmes UOP (sections) Program Organization Units (POUs)*
- **Strucure solide des données** – *à travers l'utilisation de langages qui restreignent les opérations aux types de variables appropriées*
- **Contrôle de l'exécution** – *par le découpage en tâches*
- **Conduite de séquences complexes** – *par les grafcets (Sequential Function Charts, SFC)*
- **Encapsulation** – *par l'utilisation de structures et de données complexes des UOPs (POUs),*

Un exemple:

Système de contrôle de fermentation

Courtoisie de Omron Electronics

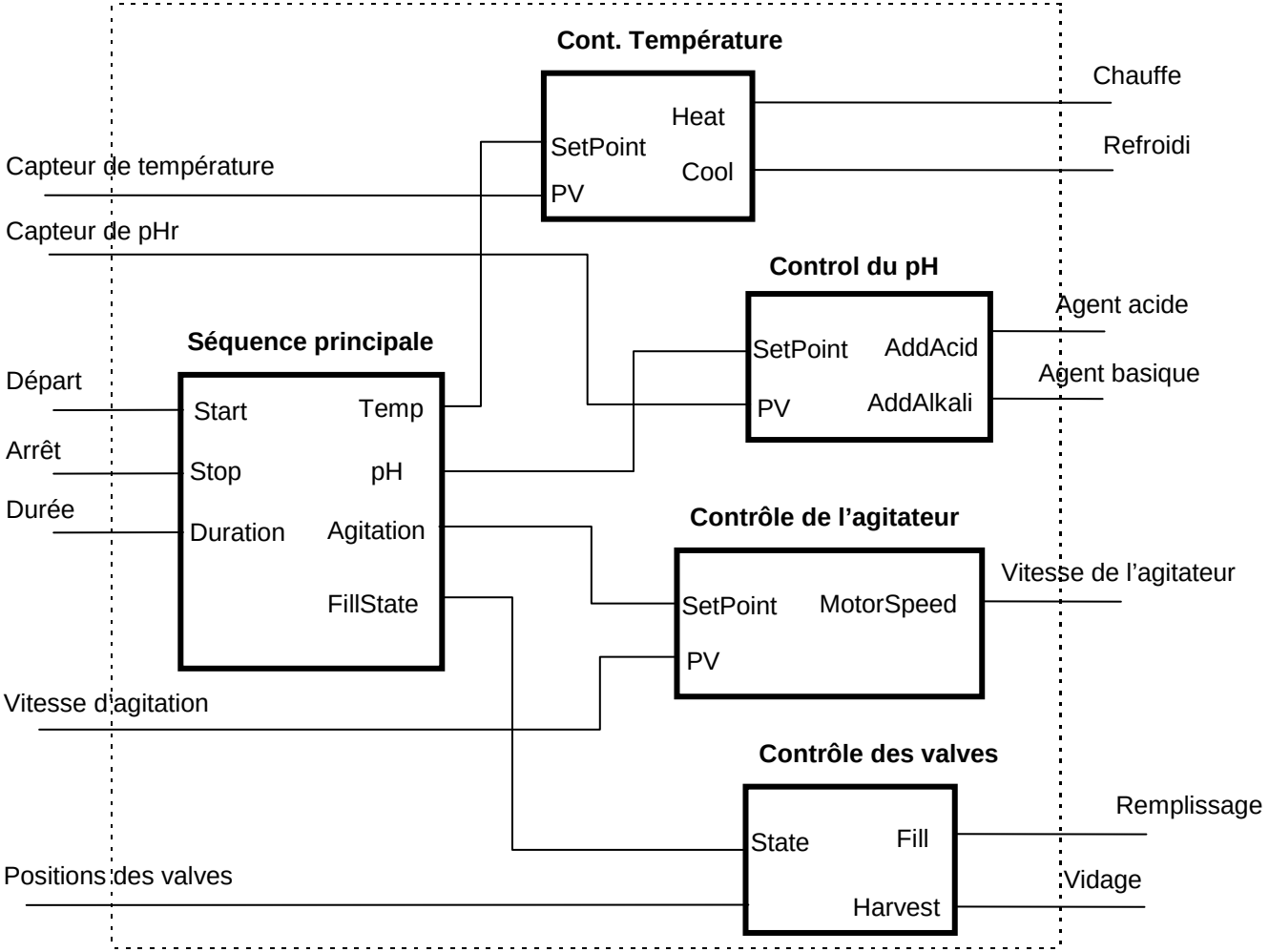
Procédé de fermentation



Décomposition du procédé de fermentation

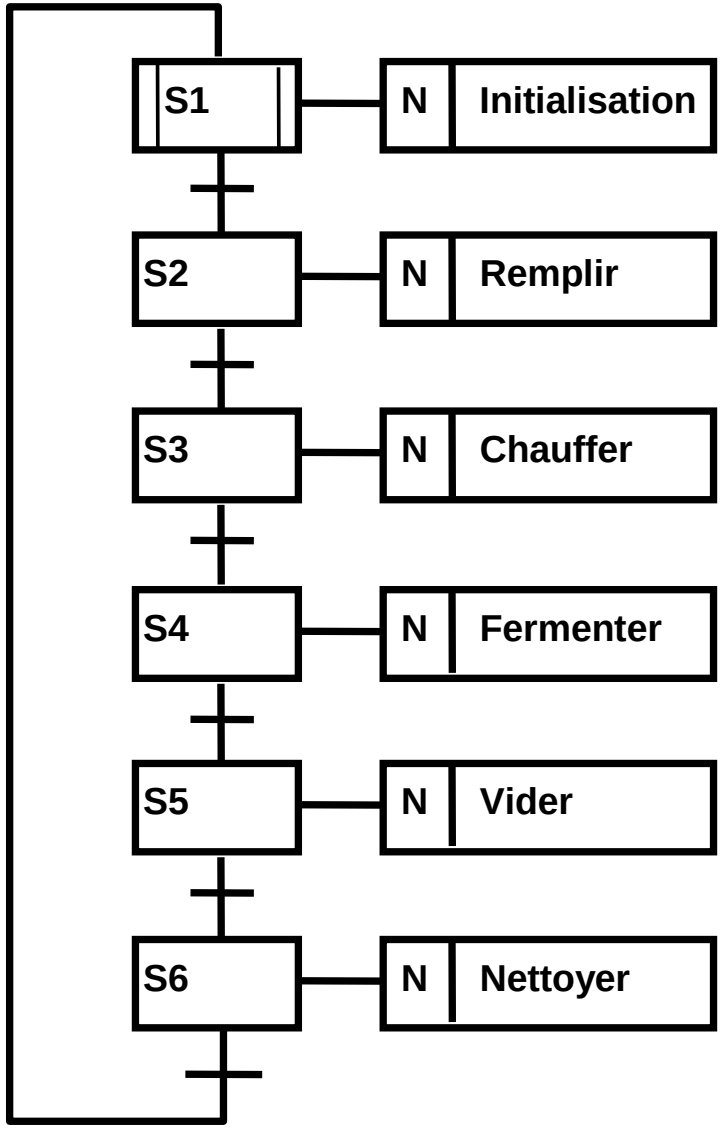
- **Séquence principale** *e.g. Principales séquences: - remplissage, chauffage, brassage, fermentation, tirage, nettoyage.*
- **Contrôle des valves** *e.g. Opération des valves de remplissage et de vidage*
- **Contrôle de Temperature** *pour contrôller la température du réservoir et moduler le chauffage*
- **Contrôl de l'agitateur** *pour activer le moteur de l'agitateur selon les consignes de la séquence principale*
- **Control du pH** *pour contrôller l'acidité du produit à fermenter et ajouter au besoin un agent acide ou basique*

Programme du contrôle de la fermentation



Séquence principale (SFC)

Montre les principales phases du procédé



IEC 61131-3

Survol...

Le Standard IEC 61131-3

Éléments communs
Common Elements

Langages de programmation
Programming Languages

Le Standard IEC 61131-3

Éléments communs
Common Elements

Langages de programmation
Programming Languages

Les Langages de Programmation IEC 61131-3

Liste d'instruction
Instruction List

```
LD      A
ANDN    B
ST      C
```

Texte structuré
Structured Text

```
C:= A   AND  NOT B
```

Diagramme bloc
Function Block Diagram

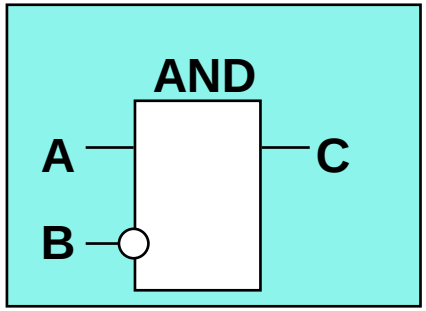


Diagramme en echelle
Ladder Diagram

```
A   B                C
-| |--|/|---------------( )
```

Les éléments communs comprennet :

- **Variables, Types de données et Déclarations**
- **Configuration, Ressources et Tâches**
- **Fonctions, Fonction Blocs et Programmes**
- **Grafcet (*Sequential Function Charts*)**

IEC 61131-3 : Éléments communs

Variables

- **Representation symbolique via les étiquettes (labels)**
- **Zone réservées pour le mapping des E/S**
- **Le code est indépendant du hardware**

Qu'est-ce que ceci?

01010101 10101010

IEC 61131-3 : Éléments communs

Types de données

comme:

BOOL BYTE

INTEGER : SINT, INT, DINT, LINT

USINT, UINT, UDINT, ULINT

REAL, LREAL

DATE

TIME_OF_DAY

DATE_AND_TIME

STRING

Déclaration de variable

- **Les variables sont déclarées par une étiquette textuelle:**
  - Un but: local (ou global)
  - Passage paramètres explicites par des variables (entrées ou sorties)
  - allocation de la mémoire
- **Lors de leur déclaration on peut y inclure les valeurs initiales**
- **Associées au Unité d'Organisation de Programme (Fonction, Bloc Fonction ou Programme)**

Déclaration de variable

Mot clé	Usage de la variable
VAR	Interne à l'unité d'organisation (POU)
VAR_INPUT	Origine externe, non modifiable dans l'unité
VAR_OUTPUT	Générée par l'unité aux entités externes
VAR_IN_OUT	Origine externe, mais peuvent être modifiée par l'unité
VAR_EXTERNAL	Fournie par configuration via VAR_GLOBAL
VAR_GLOBAL	Déclaration variable globale
VAR_ACCESS	Déclaration du chemin d'accès
RETAIN	Variables rétentives
CONSTANT	Constante (ne peu pas être modifiée)
AT	Assignement d'une location

Déclarations de variable : exemple

```
VAR
        CONDITION_RED : BOOL;
        IBOUNCE : WORD;
        MYDUB : DWORD;
        AWORD, BWORD, CWORD: INT;
        OKAY : STRING[10] := ‘OK’;
END_VAR
```

IEC 61131-3 : Éléments Communs

Types de données & Variables

Mais aussi:

Vos propres types de données…

Et types dérivés

Types de données dérivés

Dérivation directe depuis des types élémentaires e.g.:

```
TYPE R : REAL ; END_TYPE
```

Type de données énumérés, e.g.:

```
TYPE ANALOG_SIGNAL_TYPE : (SINGLE_ENDED, DIFFERENTIAL) ;
END_TYPE
```

Sous-groupe de type de données (Sub-range), e.g.:

```
TYPE ANALOG_DATA : INT (-4095..4095) ; END_TYPE
```

Types de données en tableau (Array), e.g.:

```
TYPE ANALOG_16_INPUT_DATA : ARRAY [1..16] OF ANALOG_DATA ;
END_TYPE
```

Vos propres types de données: types dérivés

```
TYPE
  ANALOG_CHANNEL_CONFIGURATION :
    STRUCT
       RANGE : ANALOG_SIGNAL_RANGE ;
       MIN_SCALE : ANALOG_DATA ;
       MAX_SCALE : ANALOG_DATA ;
    END_STRUCT ;

  ANALOG_16_INPUT_CONFIGURATION :
    STRUCT
       SIGNAL_TYPE : ANALOG_SIGNAL_TYPE ;
       FILTER_PARAMETER : SINT (0..99) ;
       CHANNEL : ARRAY [1..16] OF ANALOG_CHANNEL_CONFIGURATION ;
    END_STRUCT ;
END_TYPE
```

Vos propres types de données: types dérivés

```
TYPE
  ANALOG_CHANNEL_CONFIGURATION :
    STRUCT
      RANGE : ANALOG_SIGNA
      MIN_SCALE : ANALOG_DATA ;
      MAX_SCALE : ANALOG_DATA ;
    END_STRUCT ;

  ANALOG_16_INPUT_CONFIGURATION :
    STRUCT
      SIGNAL_TYPE : ANALOG_SIGNAL_TYPE ;
      FILTER_PARAMETER : SINT (0..99) ;
      CHANNEL : ARRAY [1..16] OF ANALOG_CHANNEL_CONFIGURATION ;
    END_STRUCT ;
END_TYPE
```

TYPE ANALOG_DATA : INT (-4095..4095) ; END_TYPE

Vos propres types de données: types dérivés

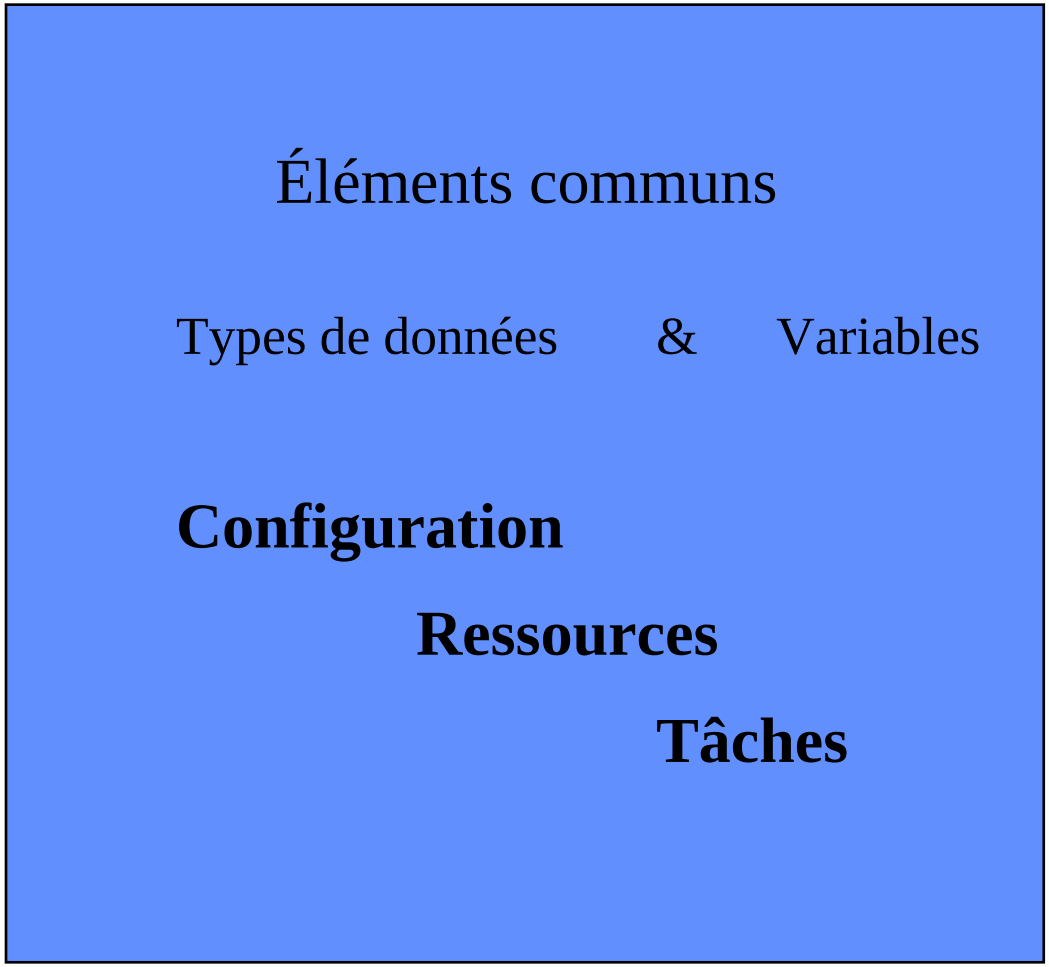
```
TYPE
  ANALOG_CHANNEL_CONFIGURATION :
    STRUCT
      RANGE : ANALOG_SIGNAL_RANGE ;
      MIN_SCALE : ANALOG_DATA ;
      MAX_SCALE : ANALOG_DATA ;
    END_STRUCT ;

  AI
    STRUCT
      SIGNAL_TYPE : ANALOG_SIGNAL_TYPE ;
      FILTER_PARAMETER : SINT (0..99) ;
      CHANNEL : ARRAY [1..16] OF ANALOG_CHANNEL_CONFIGURATION ;
    END_STRUCT ;
END_TYPE
```

```
TYPE ANALOG_SIGNAL_TYPE : (SINGLE_ENDED, DIFFERENTIAL) ; END_TYPE
```

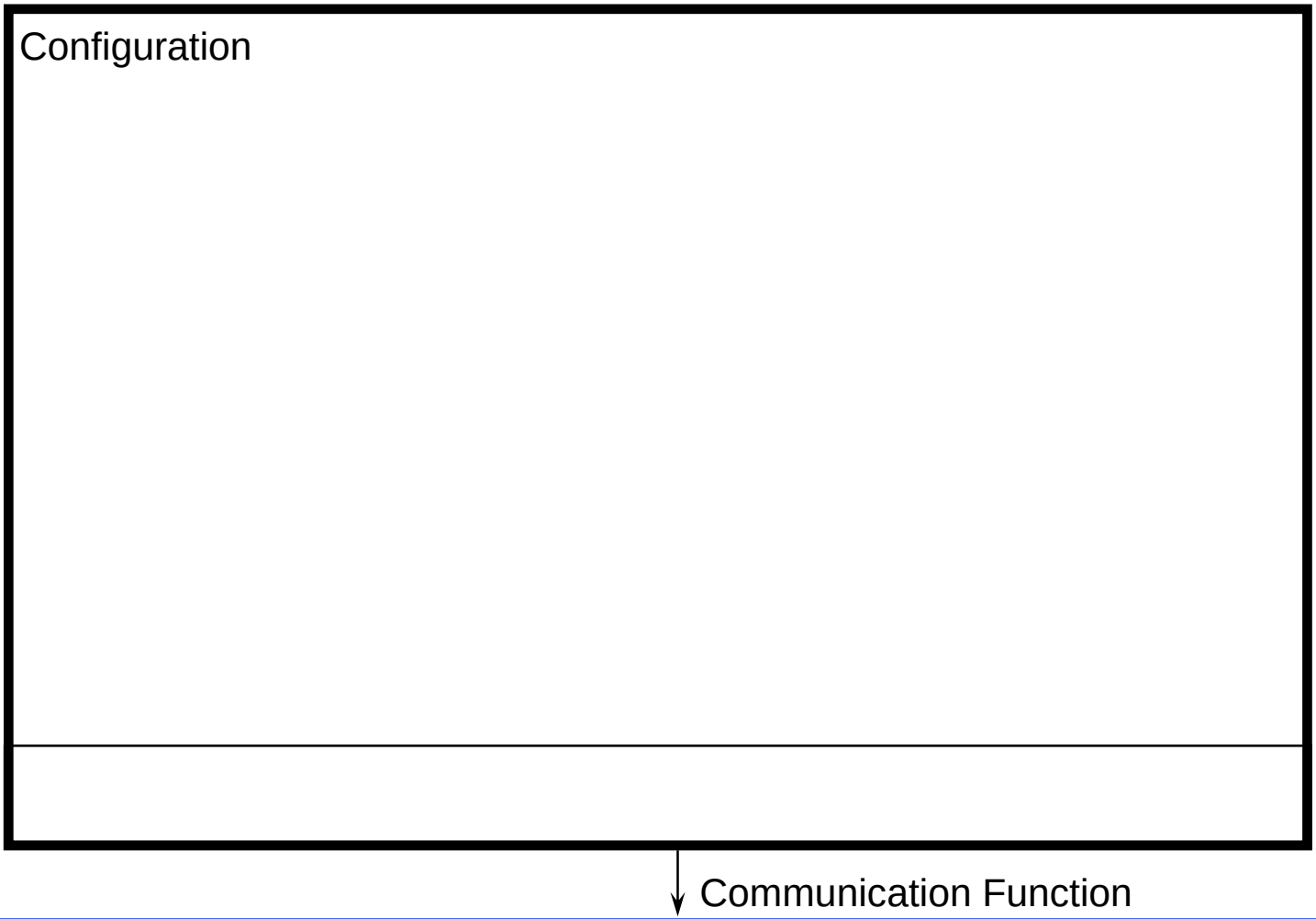
Variables directes : %

Préfixe	Signification	Type par défaut
I	Entrée	
Q	Sortie	
M	Bit mémoire	
X	Bit de mot	BOOL
None	Bit de mot	BOOL
B	Byte (8 bits)	BYTE
W	Word (16 bits)	WORD
D	Double word (32 bits)	DWORD
L	Long (quad) word (64 bits)	LWORD

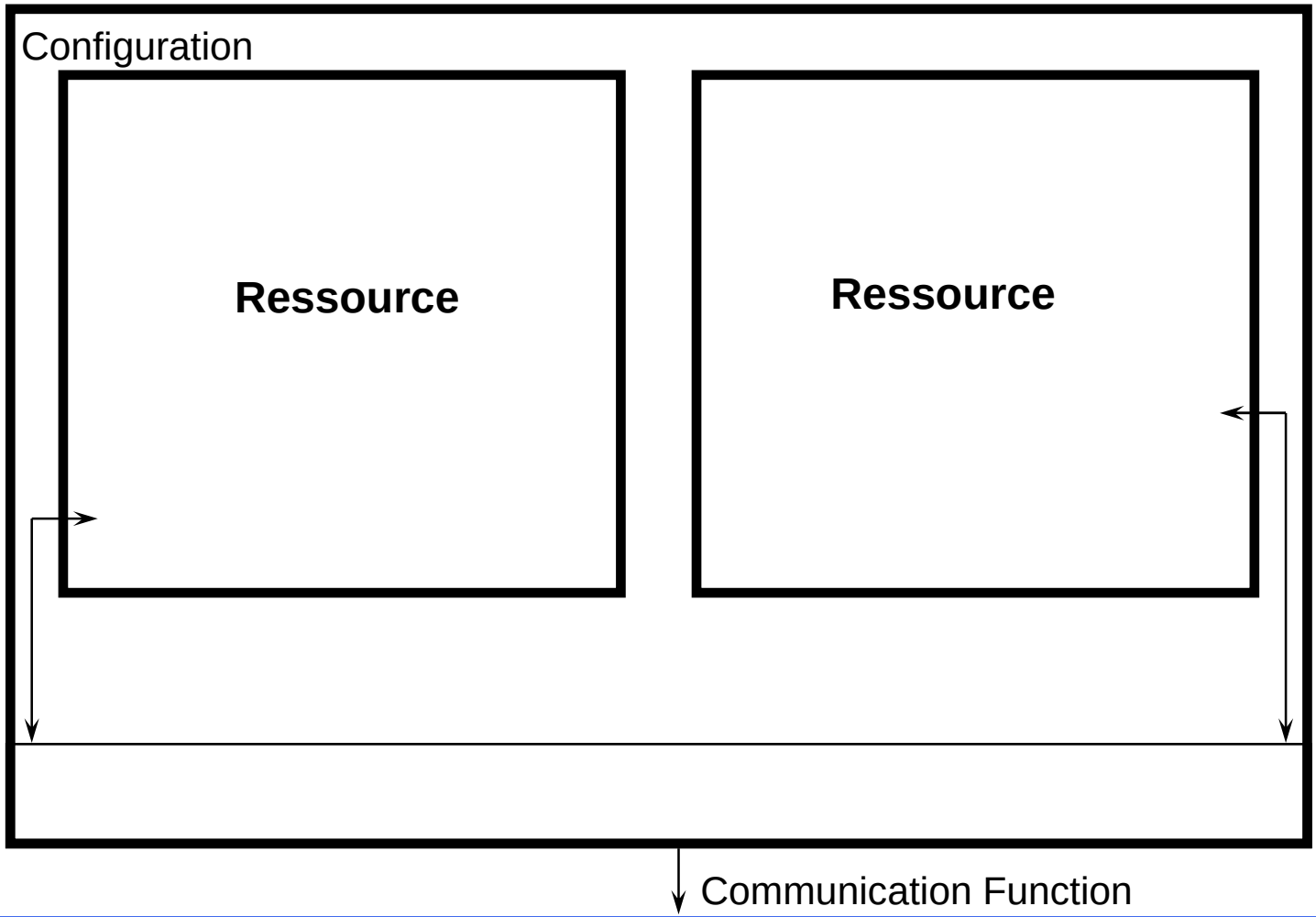
IEC 61131-3 : Éléments Communs



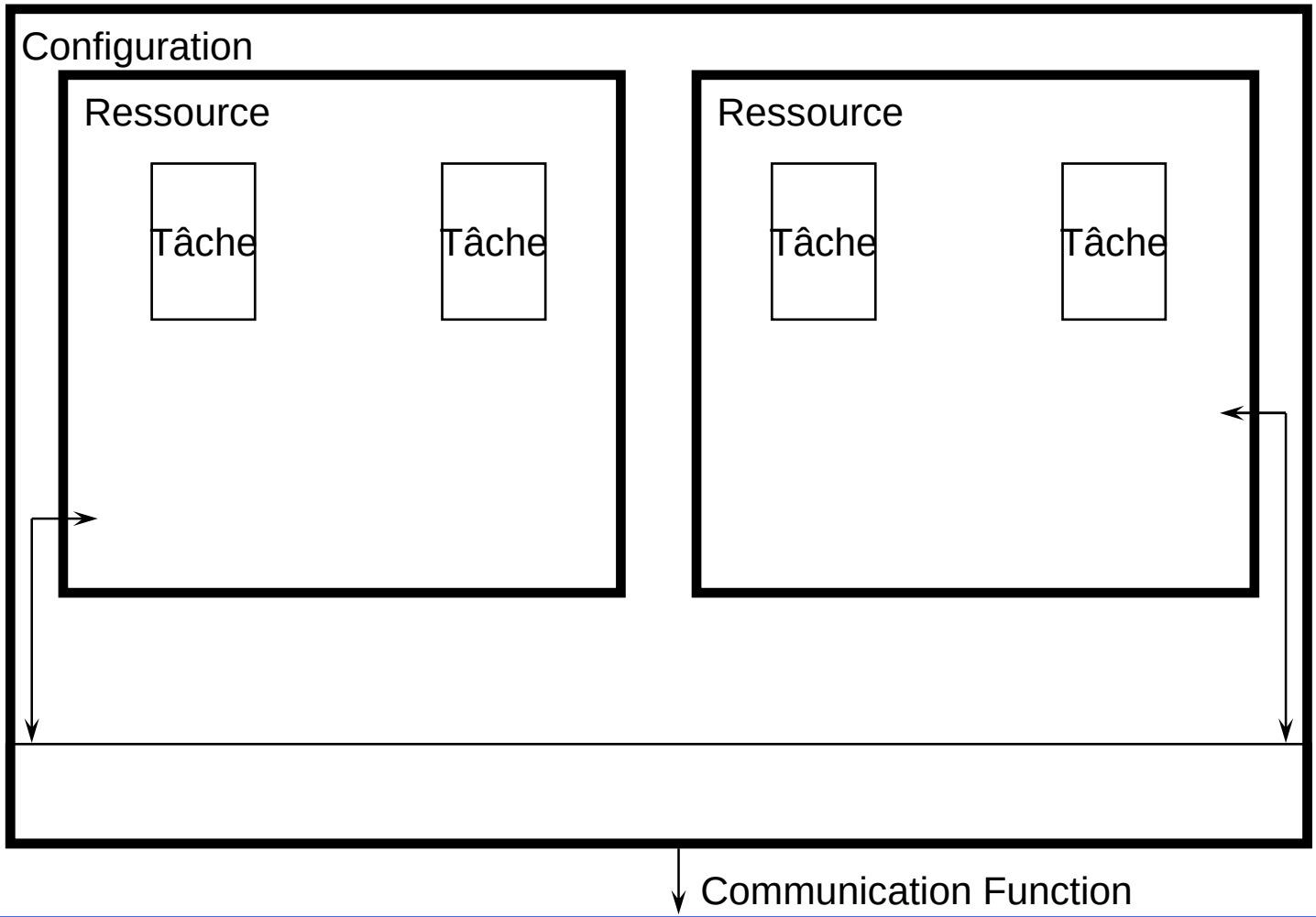
IEC 61131-3 Modèle de programme



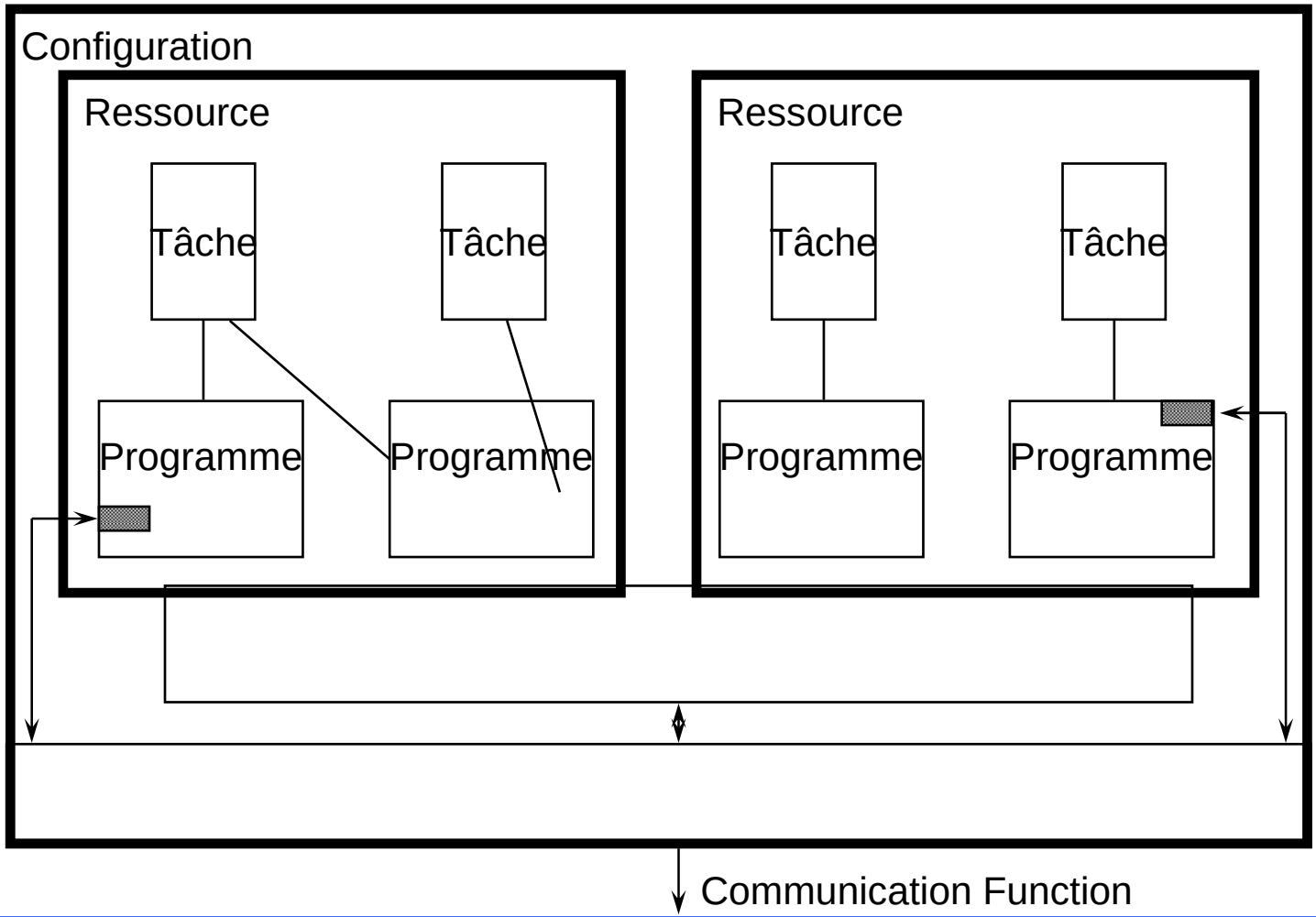
IEC 61131-3 Modèle de programme



IEC 61131-3 Modèle de programme

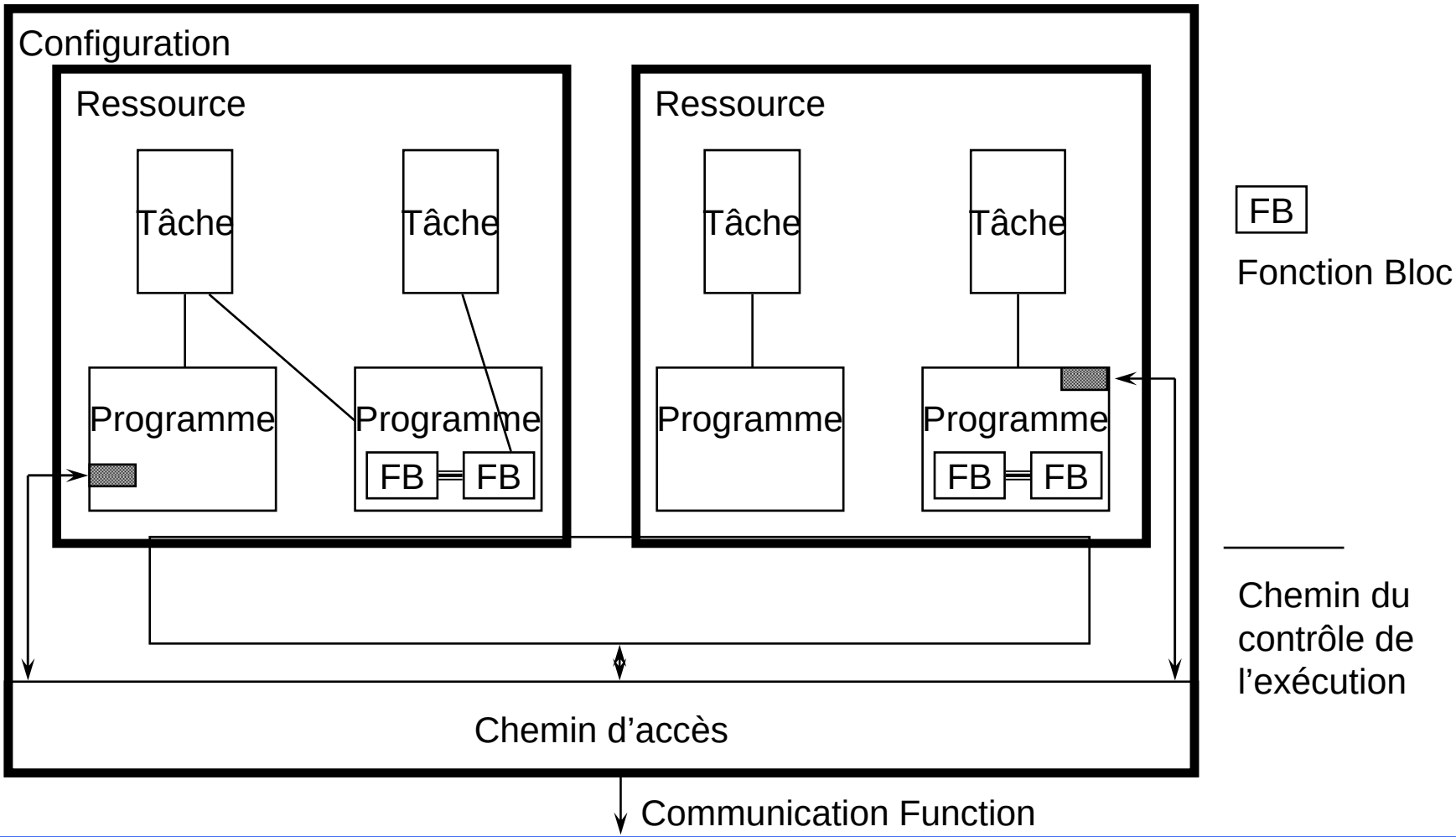


IEC 61131-3 Modèle de programme

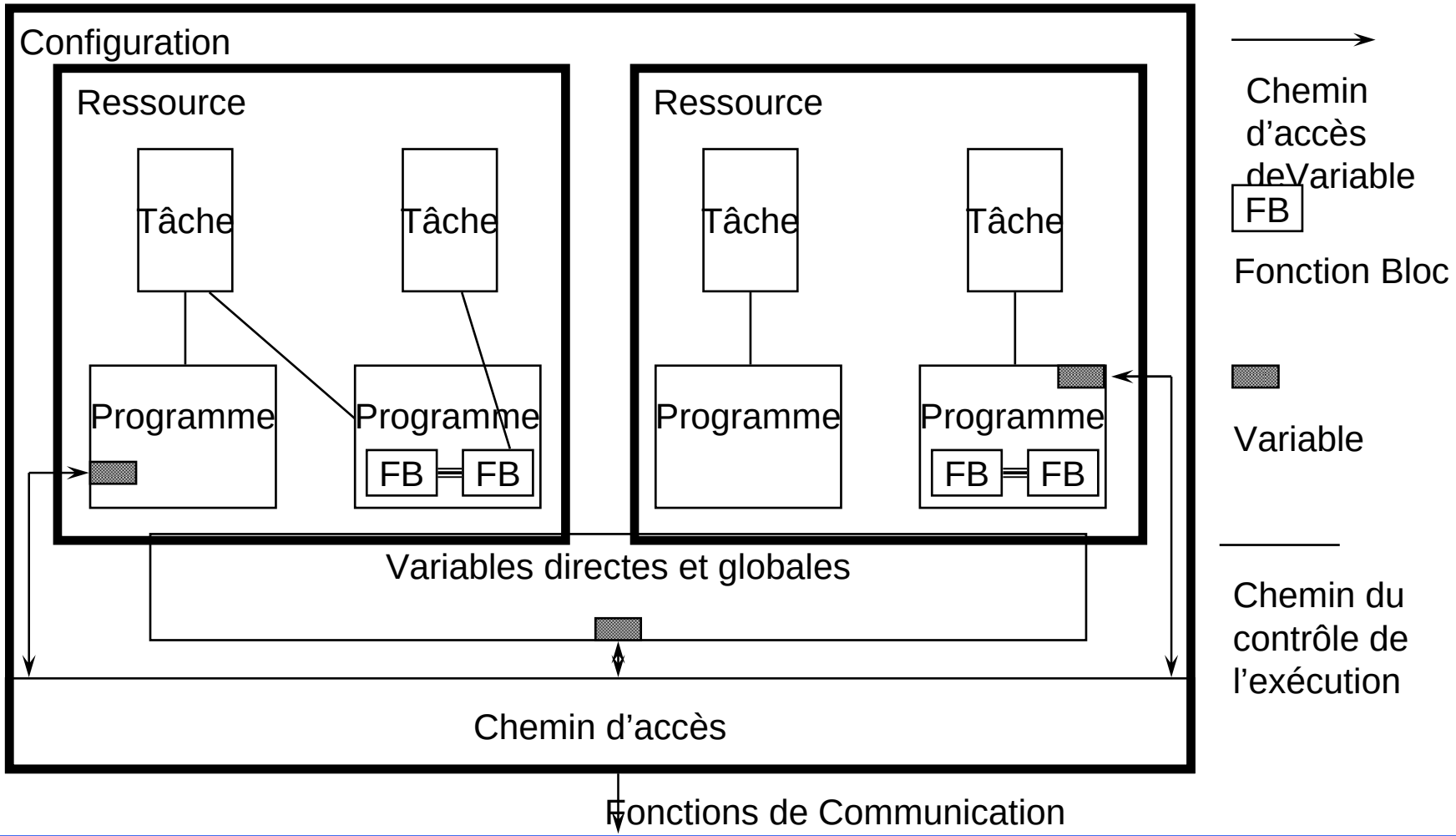


Chemin du contrôle de l'exécution

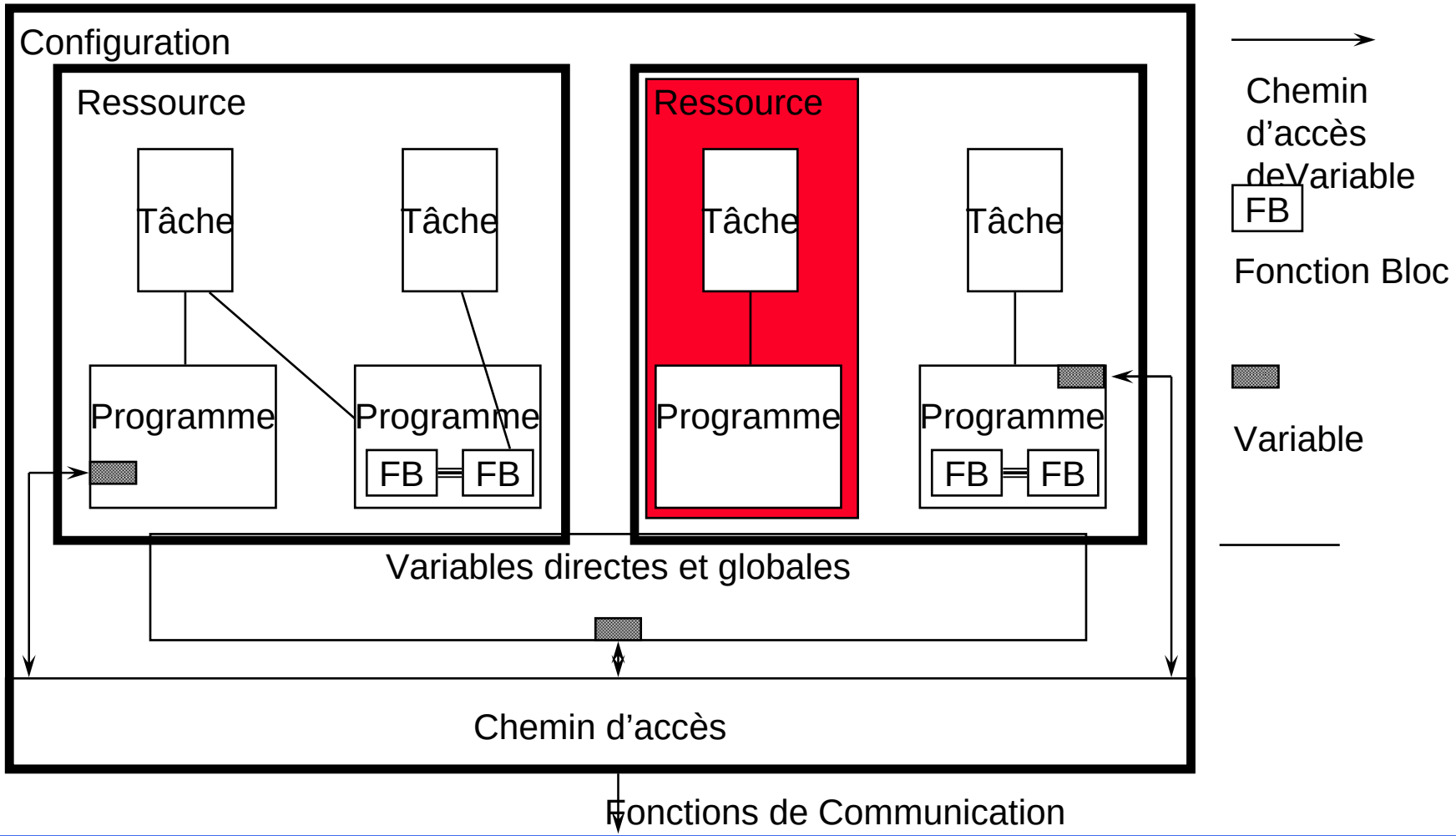
IEC 61131-3 Modèle de programme



IEC 61131-3 Modèle de programme



IEC 61131-3 vs PLC conventionnel



Configuration des éléments

- **Configuration**
- **Ressources**
- **Tâches**
- **Variables globales**
- **Chemins d’accès**

Configuration, Ressources et Chemins d'accès (-Déclaration)

CONFIGURATION … END_CONFIGURATION

VAR_GLOBAL … END_VAR (within CONFIGURATION)

RESOURCE … ON … END_RESOURCE

VAR_GLOBAL … END_VAR (within RESOURCE)

PERIODIC TASK

NON-PERIODIC TASK

IEC 61131-3 : Éléments Communs

ÉLÉMENTS COMMUNS

Types de données & Variables

Configuration, Ressources, Tâches

Unités d'organisation de programmes (POU)

* **Fonctions**
* **Fonction Blocs**
* **Programmes**

Fonctions .....

*** Fonctions Standards**

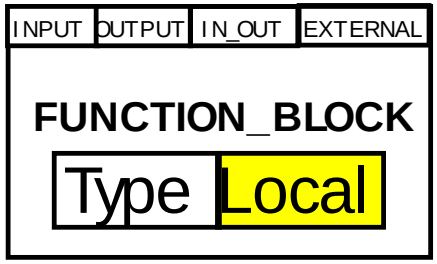
ADD, SQRT, SIN, COS, GT, MIN, MAX, AND, OR, etc.

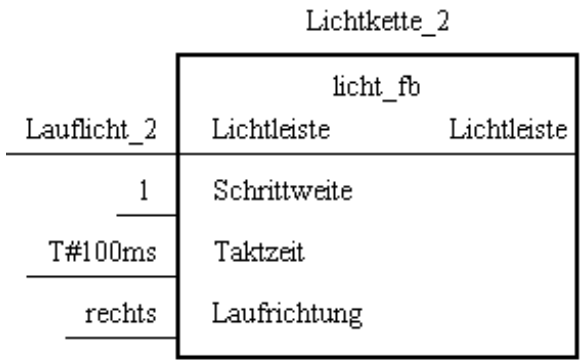
*** Fonctions définies:**

```
FUNCTION SIMPLE_FUN : REAL
        VAR_INPUT
                A, B     : REAL;
                C                : REAL := 1.0;
        END_VAR
SIMPLE_FUN := A*B/C;
END FUNCTION
```

…. & Blocs Fonction

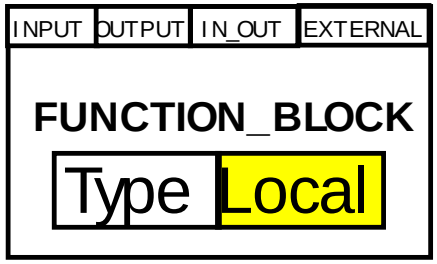
◆Blocs Fonction Standard

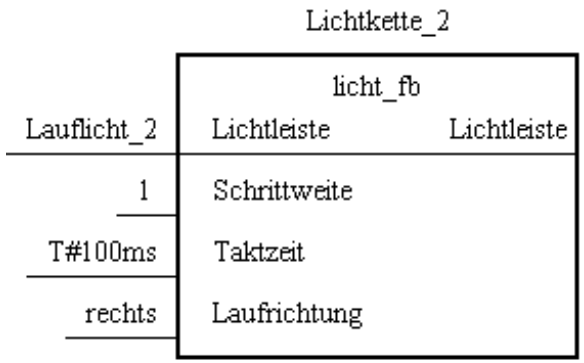




…. & Blocs fonction

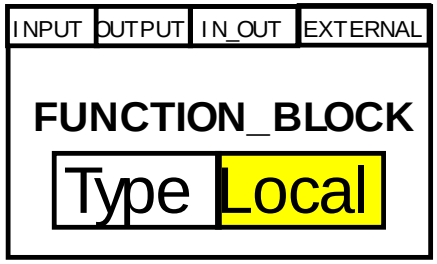
- **Blocs fonction standards**
- **Blocs fonction additionnels**

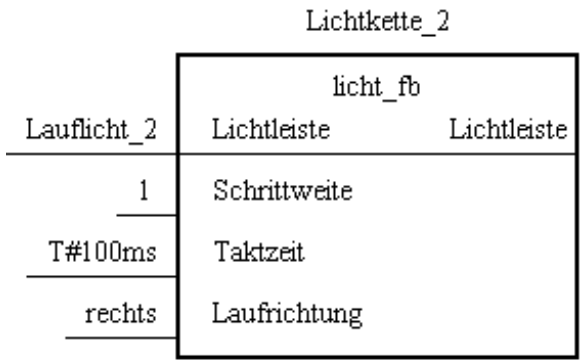




…. & Blocs fonctions

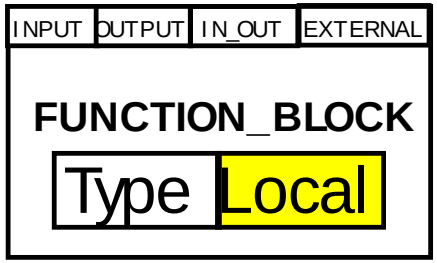
- **Blocs fonction**
- **Blocs fonction additionels**
- **Blocs de fonction définis**

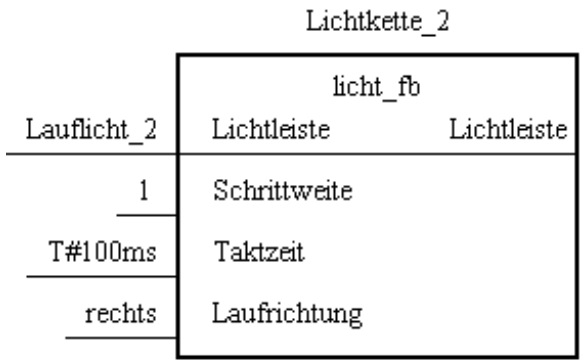




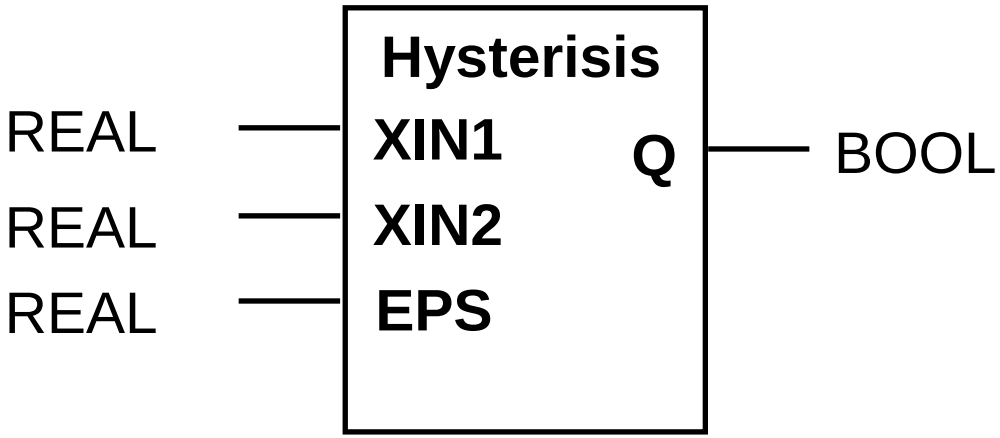
…. & Blocs fonctions

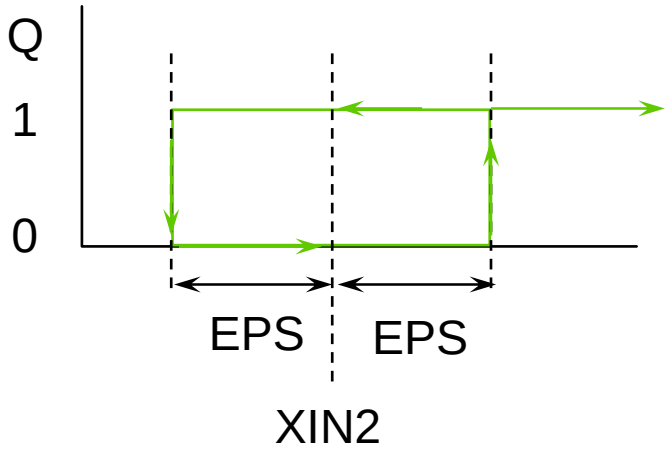
- **Blocs fonction**
- **Blocs fonction additionnels**
- **Blocs de fonction définis**
- **Les blocs de fonctions sont réutilisables à volonté…**



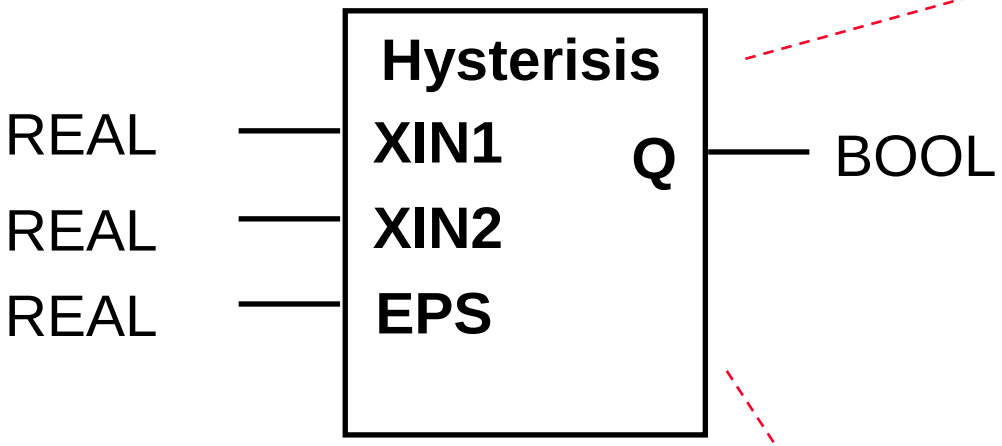


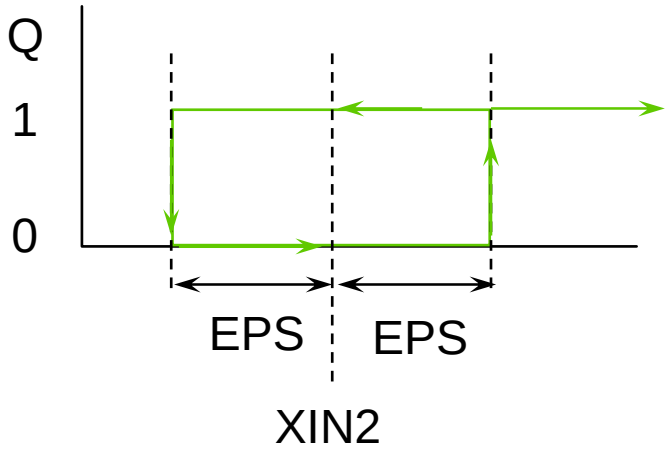
Exemple de bloc fonction





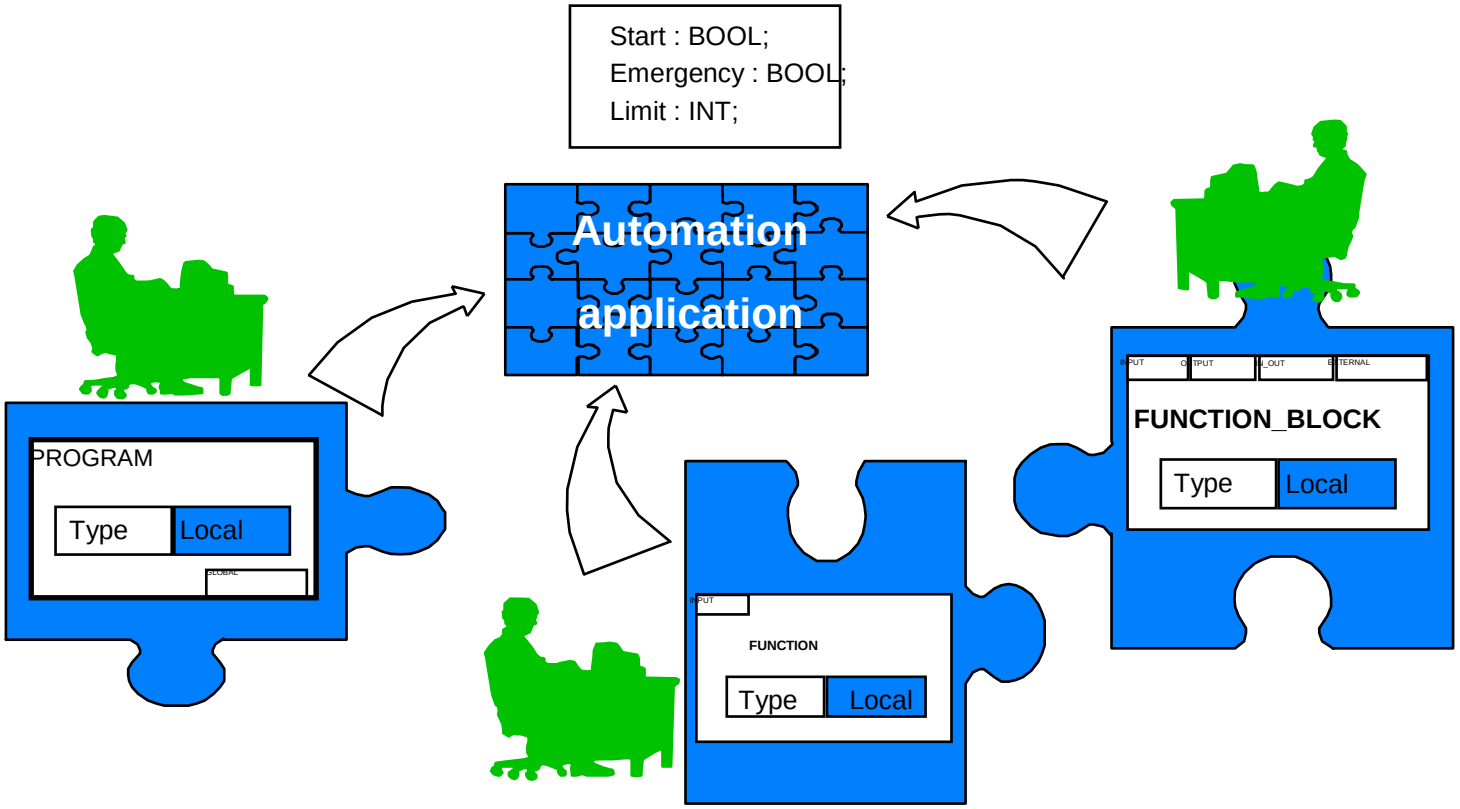
Exemple de bloc fonction



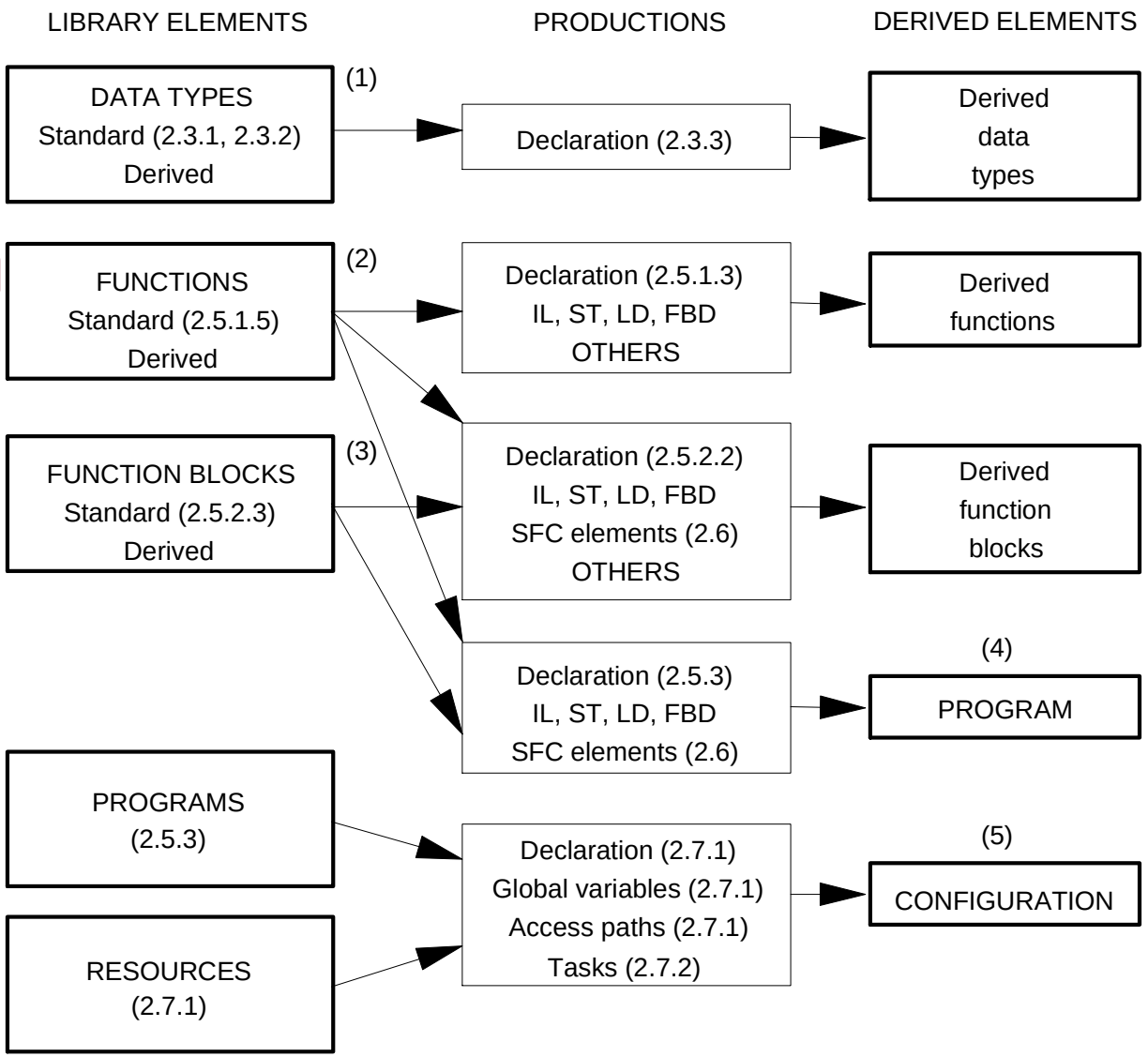



```
FUNCTION_BLOCK HYSTERISIS
 VAR_INPUT
   XIN1, XIN2 : REAL;
   EPS : REAL;  (* Hysterisis band *)
 END_VAR
 VAR_OUTPUT
   Q : BOOL := 0
 END_VAR
 IF Q THEN
   IF XIN1 < (XIN2-EPS) THEN
     Q := 0 (* XIN1 decreasing *)
   END_IF;
 ELSIF XIN1 > (XIN2 + EPS ) THEN
   Q := 1; (* XIN1 increasing *)
 END_IF;
END_FUNCTION_BLOCK
```

Programmes : conception hiérarchisée



Modèle de Programmation



IEC 61131-3 : Éléments Communs

ÉLÉMENTS COMMUNS

a.o.

Types de données Variables

Unités d'organisation de programmes

* Fonctions
* Blocs Fonction
* Programmes

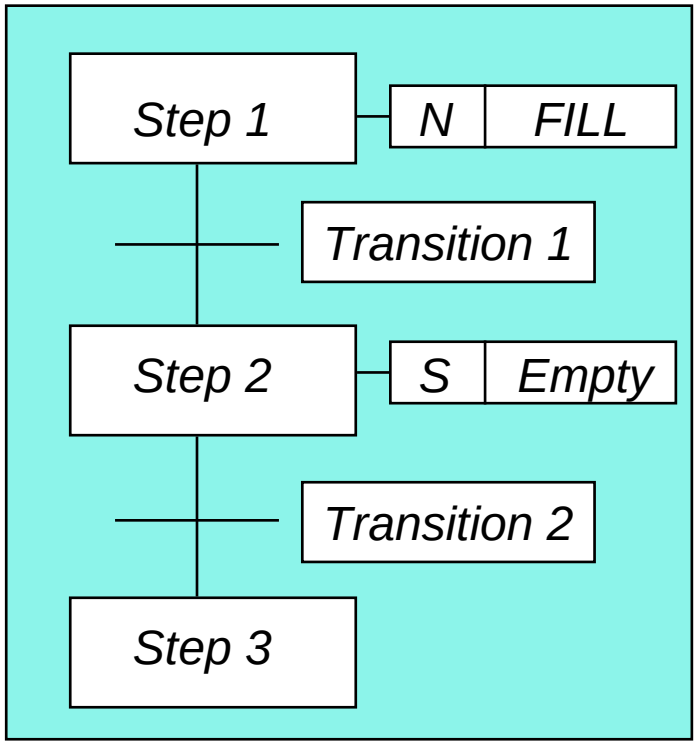
Configuration, Ressources, Tâches

Grafcet (SFC)

* Étapes
* Transitions
* Blocs d'action

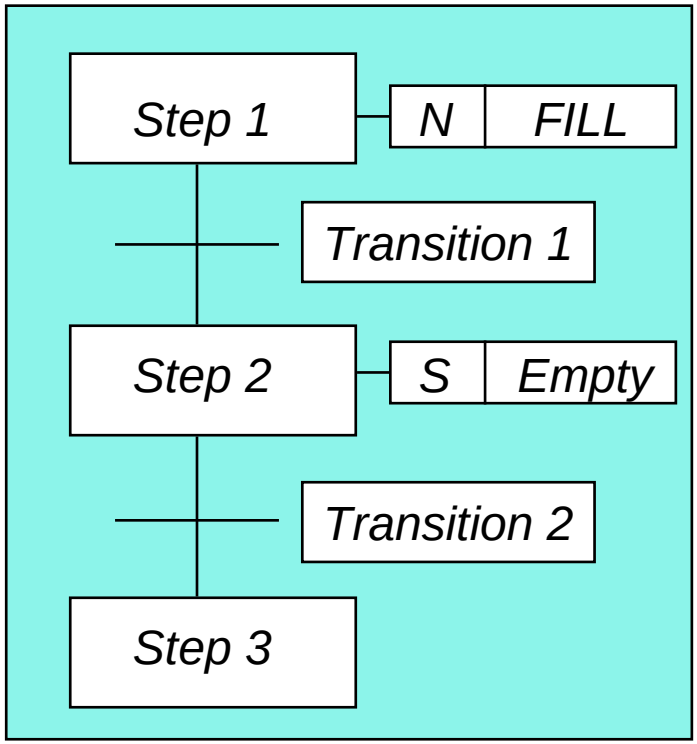
Grafcet *Sequential Function Chart, SFC*

- **Technique graphique puissante pour DÉCRIRE l'évolution séquentielle d'un programme de contrôle**
- **Utile pour décomposer un problème de contrôle**
- **Montre clairement le cheminement et aussi très efficace pour un diagnostique rapide**



Grafcet *Sequential Function Chart, SFC*

- **Technique graphique puissante pour DÉCRIRE l'évolution séquentielle d'un programme de contrôle**
- **Utile pour décomposer un problème de contrôle**
- **Montre clairement le cheminement et aussi très efficace pour un diagnostique rapide**
- **Les éléments de base sont les ÉTAPES avec les BLOCS D'ACTION et les TRANSITIONS avec leur RÉCÉPTIVITÉS**
- **Choix de séquences et séquences parallèles**



Le Standard IEC 61131-3

Éléments Communs

Langages de programmation

Descendant

Ascendant

Les environnements de programmation au standard IEC 1131-3

La plupart offrent:

- **Écrans graphiques de programmation**
- **Plusieurs fenêtres simultanées**
- **souris**
- **menus déroulant**
- **Aide contextuelle**
- **Vérification du code durant la conception**