Sécurité des Systèmes Industriels

Architectures des Systèmes Industriels

Maxime Puys

22 mars 2025

CLERMONT AUVERGNE INP Isima

Architectures typiques des systèmes industriels

Exemple

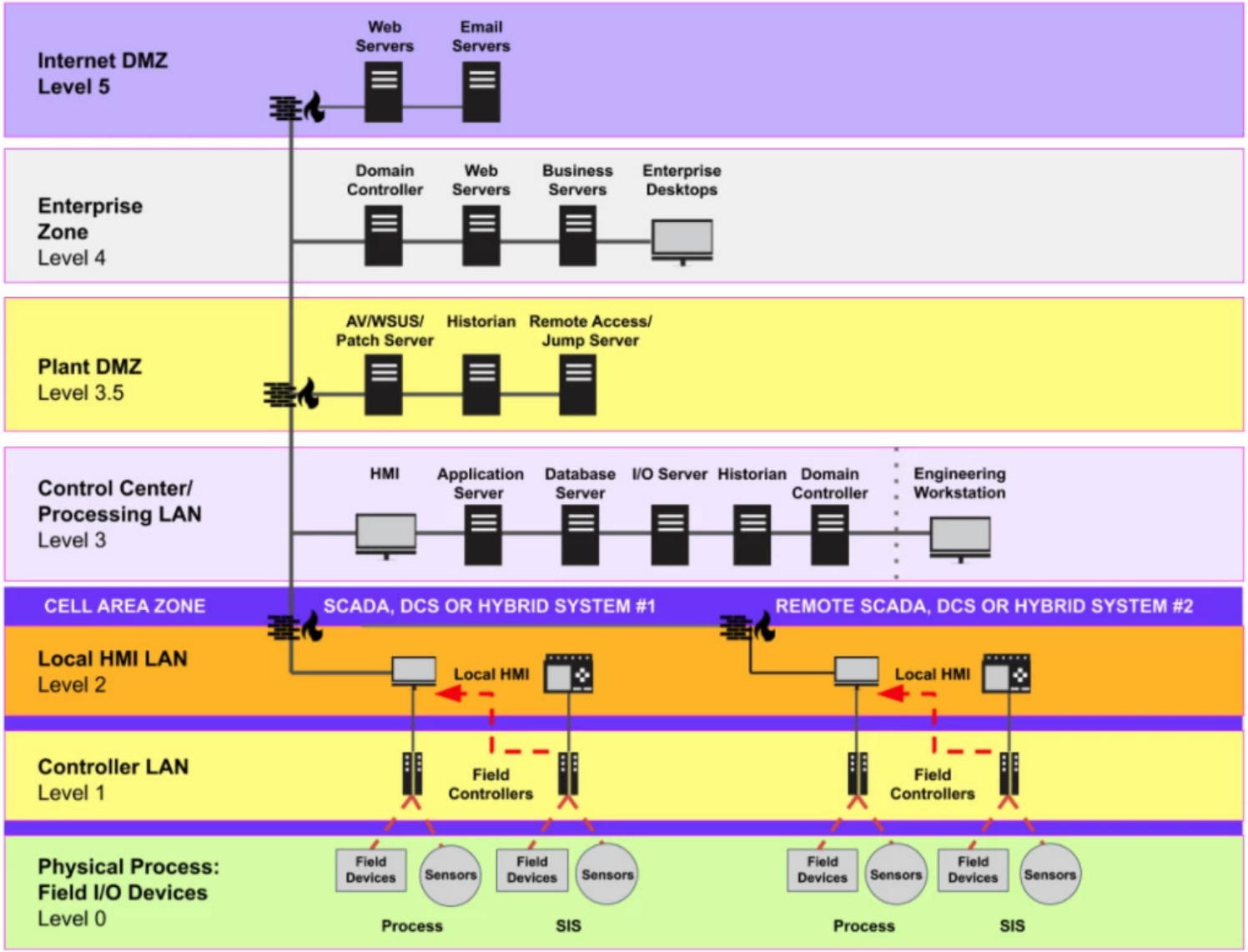
Figure – https://www.banelec.com/fr/how-do-industrial-control-systems-work/

Modèle de Purdue - Vue d'ensemble

- Hiérarchie à 5 niveaux :
  1. Niveau 0 : Dispositifs physiques (capteurs, actionneurs).
  2. Niveau 1 : Contrôleurs locaux (PLC, RTU).
  3. Niveau 2 : Contrôle des processus (SCADA, HMI).
  4. Niveau 3 : Gestion de la production (MES).
  5. Niveau 4 : Gestion de l'entreprise (ERP).
- Objectif : Structurer les flux de données et les responsabilités.

$\Rightarrow$ Dessin

Schéma hiérarchique du modèle Purdue



Modèle de Purdue - Avantages et limites

- **Avantages** :
  - Organisation claire des niveaux de responsabilité.
  - Compatibilité avec des approches sécurisées.
  - Approche éprouvée et largement utilisée.
- **Inconvénients** :
  - Pas adapté aux architectures distribuées modernes.
  - Défis pour l'intégration Industrie 4.0.

Industrie 4.0 - Principes fondamentaux

- **Contenu :**
  - Interconnexion des dispositifs via l'IoT.
  - Intégration de l'intelligence artificielle et du big data.
  - Pilotage temps réel des processus.
  - Communication horizontale et verticale (machine à machine et niveau stratégique).

Figure – https://www.forbes.com/sites/bernardmarr/2018/09/02/what-is-industry-4-0-heres-a-super-easy-explanation-for-anyone/

Industrie 4.0 - Avantages et risques

- **Avantages :**
  - Optimisation des performances.
  - Amélioration de la flexibilité et personnalisation.
- **Risques :**
  - Cybersécurité.
  - Complexité de l'implémentation (synchronisation).

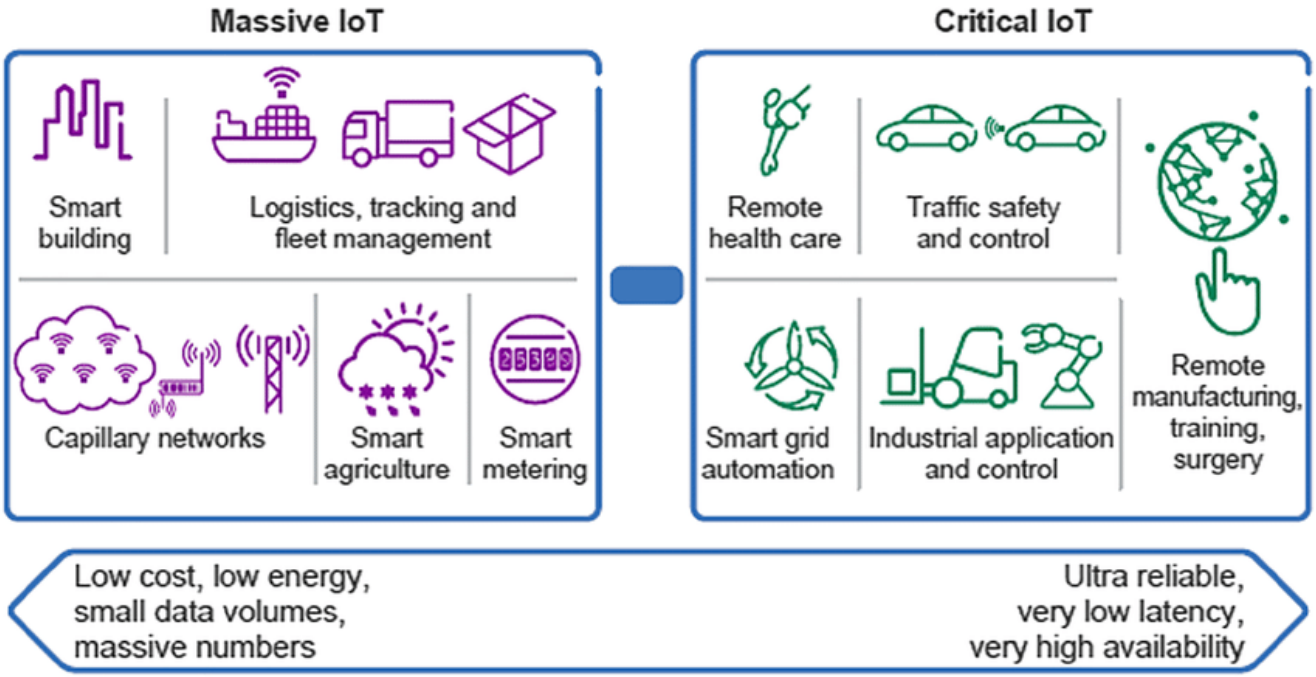


Figure – [Algahtani2019]

Autres topologies - Architectures distribuées

- **Contenu :**
  - Exemples : Redondance des PLC, architectures DCS.
  - Avantages : Haute disponibilité et résilience.

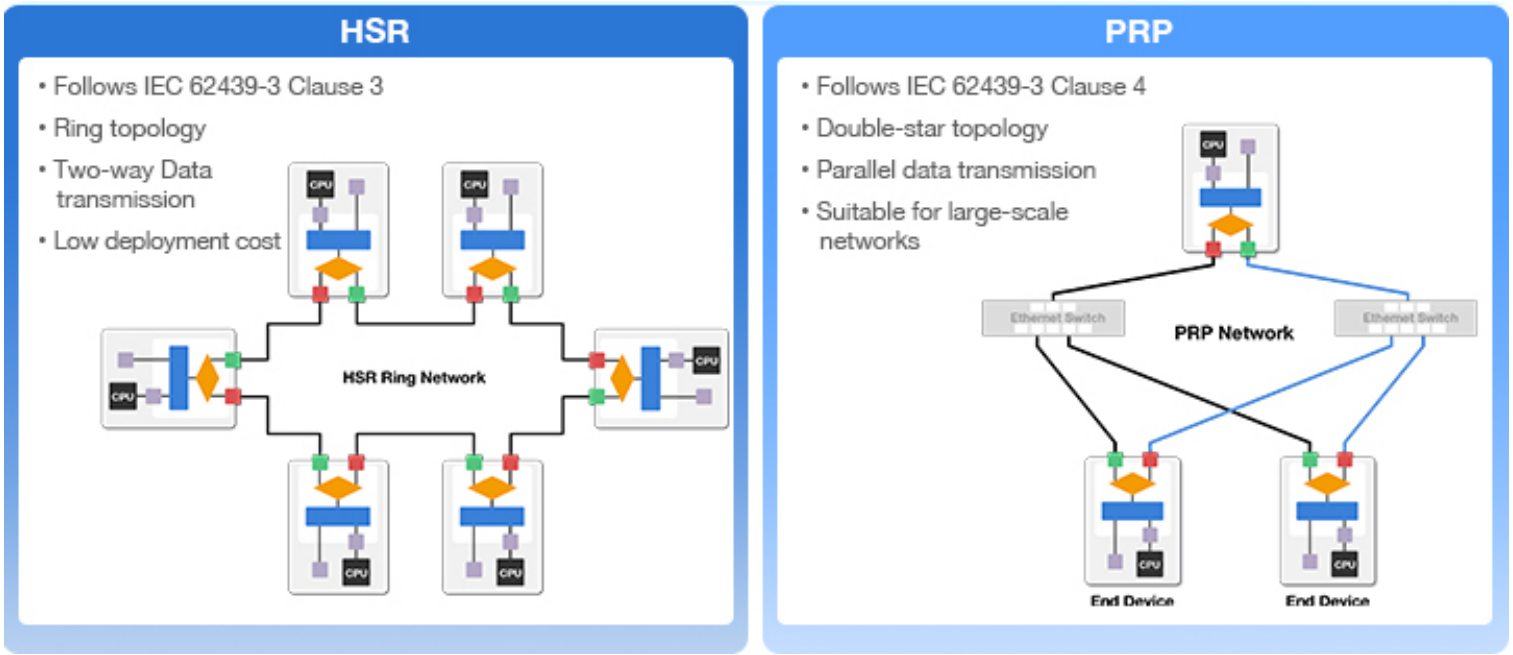


Figure – https://brodersen.com/hsr-prp-redundancy-protocols-now-native/

Autres topologies - Communications Intra Vehicule

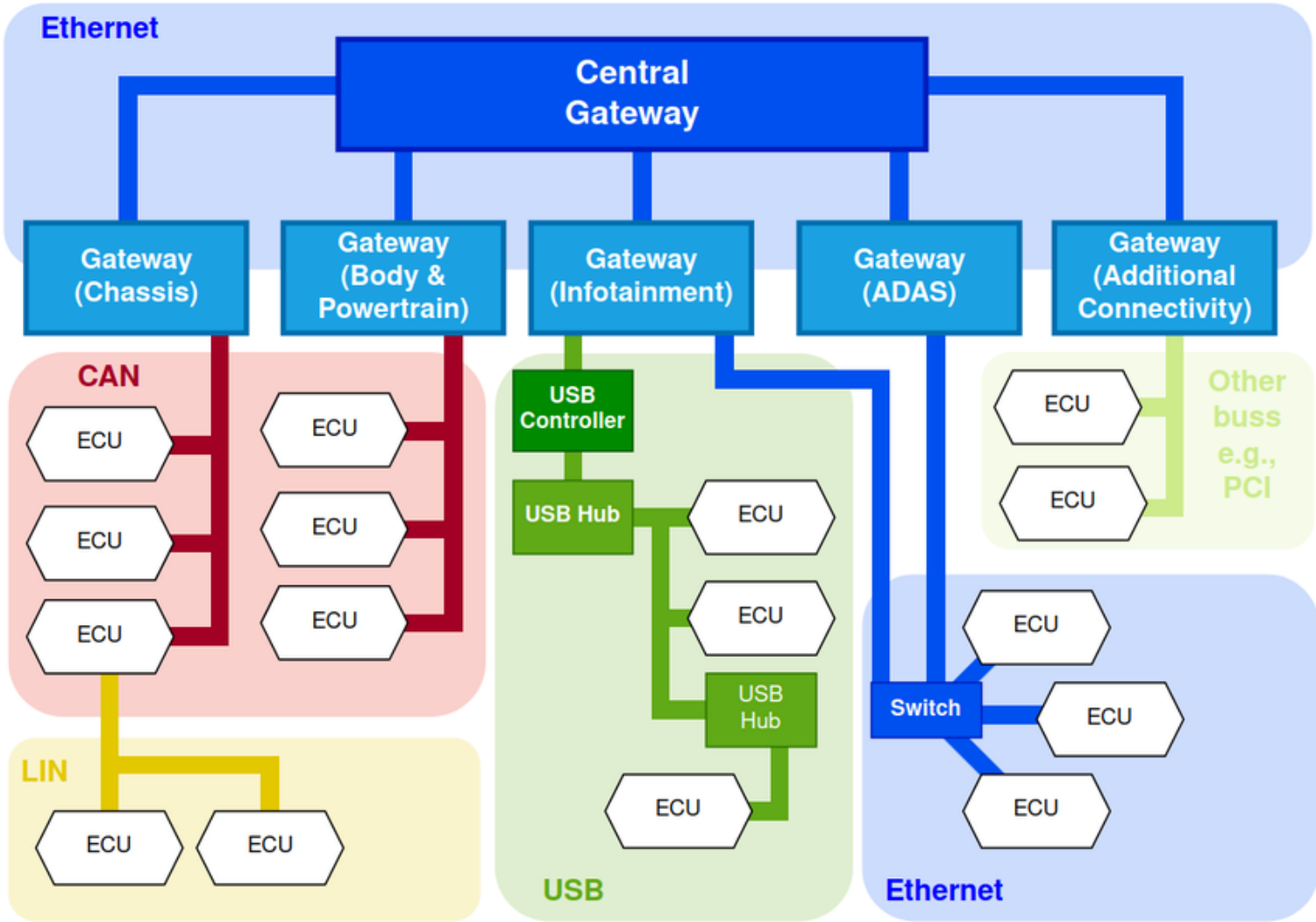


Figure – [Tiberti2023]

Autres topologies - VANETs

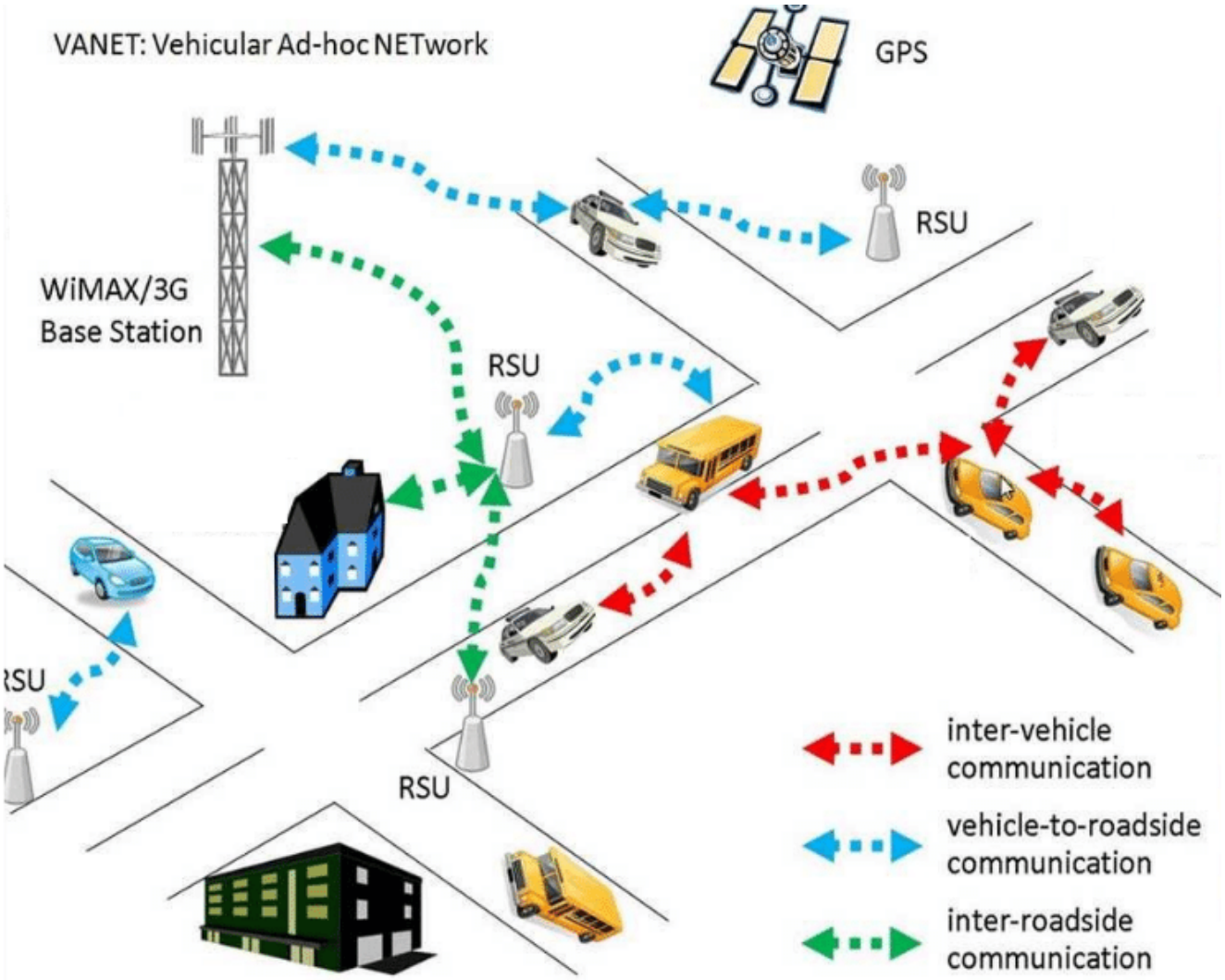


Figure – [Benadda2019]

Description des différents composants d'un système industriel

RTU (Remote Terminal Unit)

- **Fonction :** Collecte de données de capteurs distants, transmission aux centres de contrôle.
- **Communication :** Protocoles comme MODBUS, DNP3.

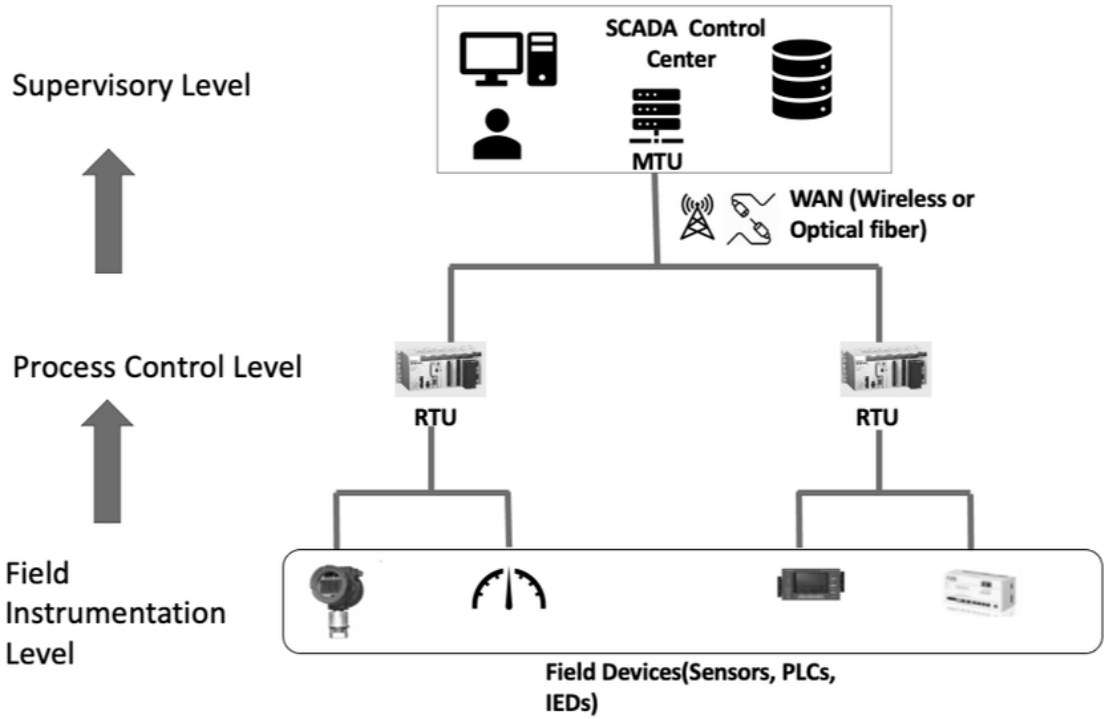


Figure – Ghosh2024

IHM (Interface Homme-Machine)

- **Rôle :** Interaction utilisateur pour la supervision des processus sur site.
- **Caractéristiques :** Interface graphique légère avec tendances, alarmes, boutons d'actions simples.

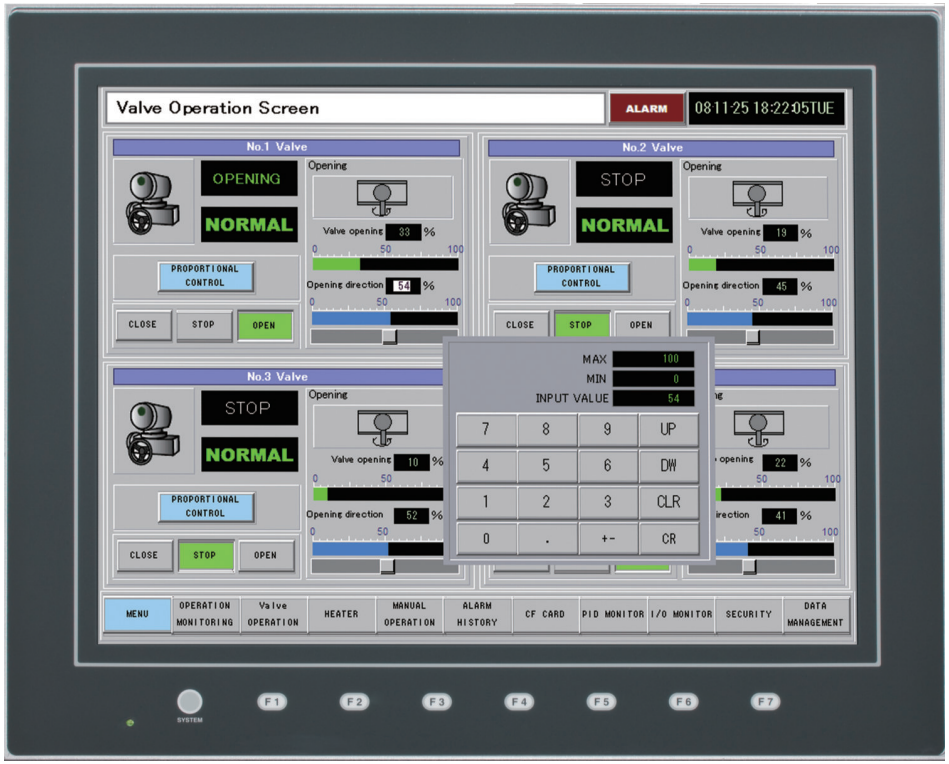


Figure – https://www.industrialcontrol.com/hmi

Automates programmables (PLC)

- **Fonction :** Commande temps réel des processus.
- Exécution de programmes définis par l'utilisateur.

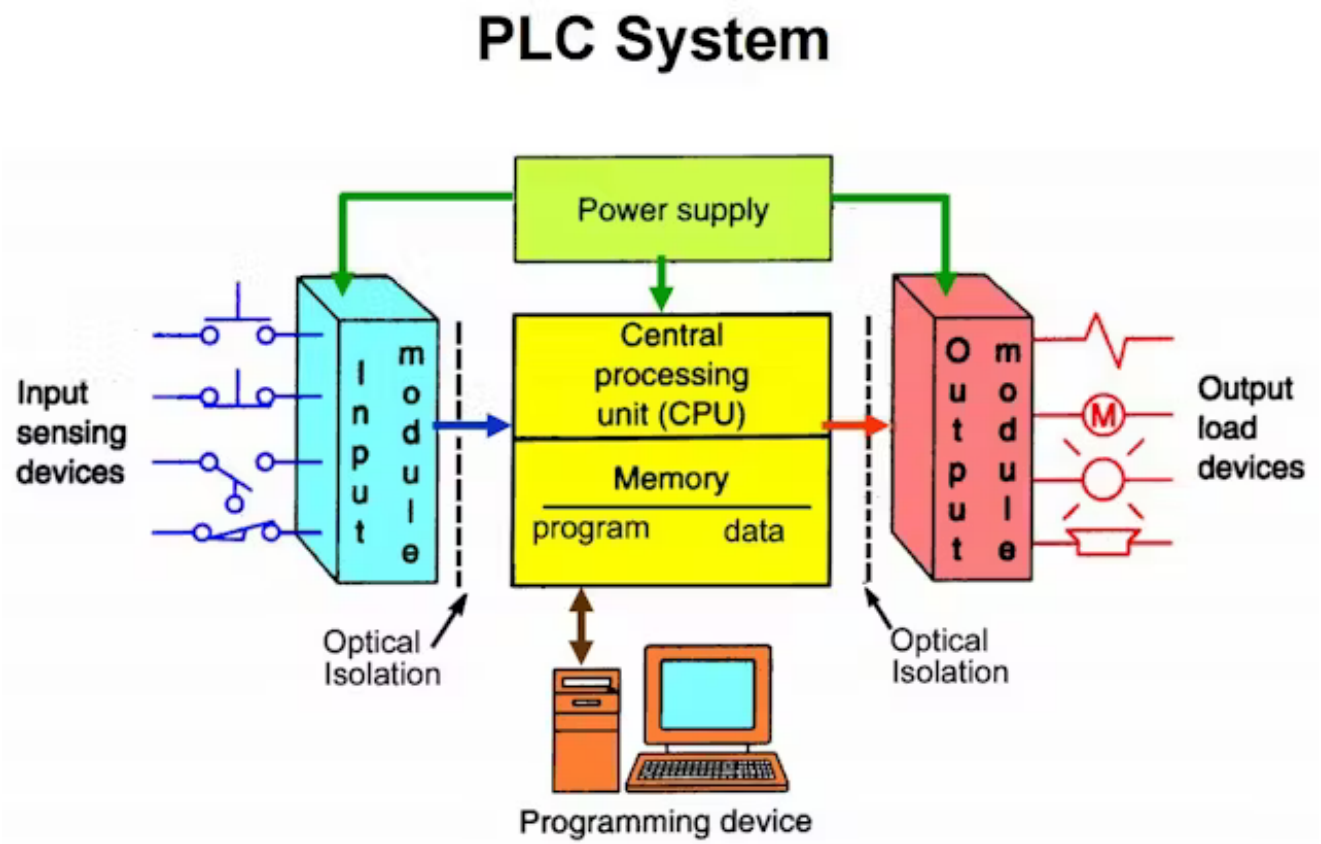


Figure – https://www.machinedesign.com/learning-resources/

SCADA (Systèmes de Contrôle et Acquisition de Données)

- **Fonction :** Supervision, contrôle et acquisition de données à distance.

Figure – https://www.machinedesign.com/learning-resources/

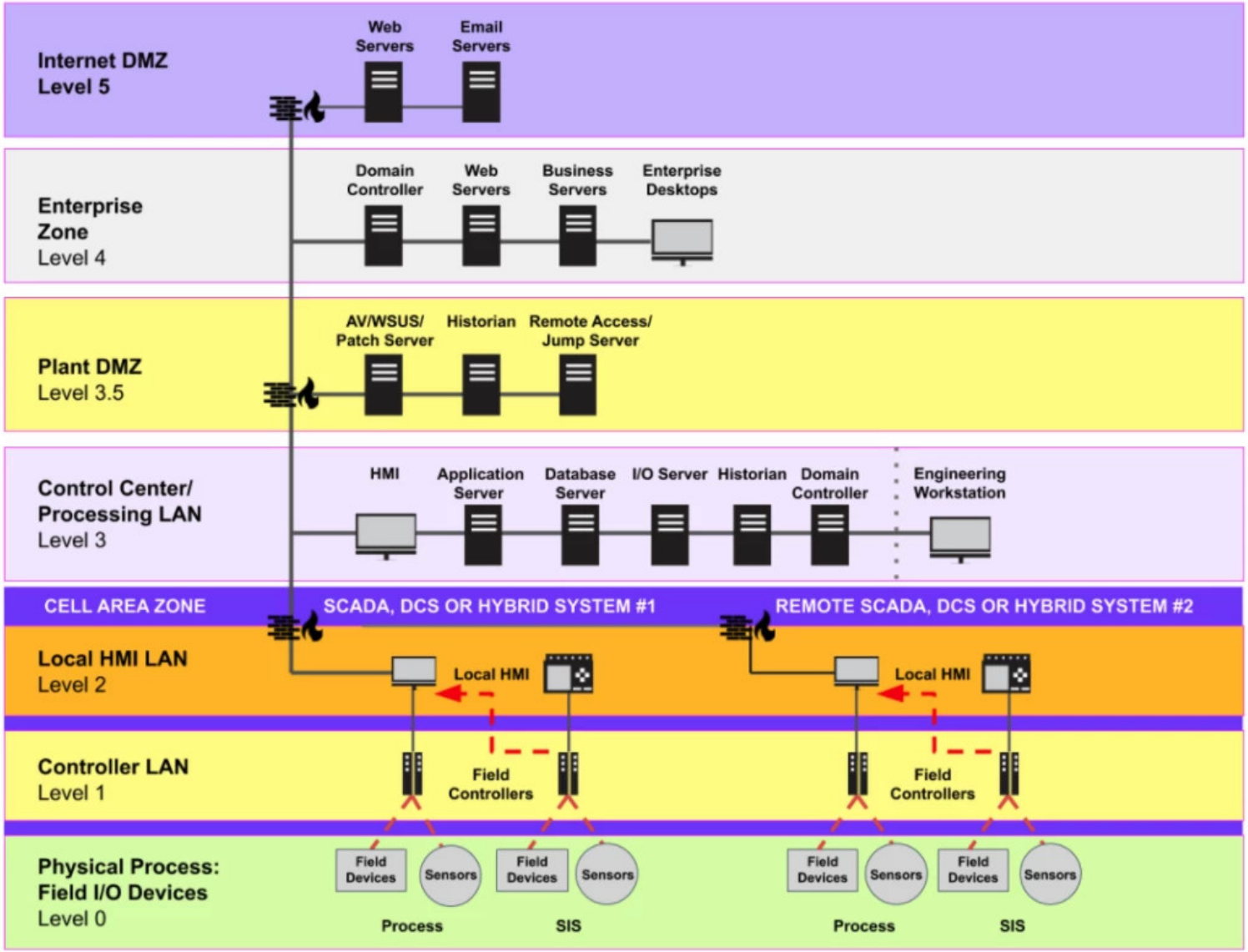
MES (Manufacturing Execution System)

- **Fonction :** Gestion des opérations de production.
- **Rôle :** Liaison entre SCADA et ERP pour optimiser la production.

ERP (Enterprise Resource Planning)

- **Fonction :** Gestion centralisée des ressources de l'entreprise (RH, finances, logistique).
- **Rôle :** Interface avec MES pour aligner production et stratégie d'entreprise.

Schéma hiérarchique du modèle Purdue



Programmation d'automates en langage Grafcet

Introduction au GRAFCET

- **Définition** : Graphique Fonctionnel de Commande par Étapes et Transitions.
- **Origine** : Créé en 1977 par un groupe d'universitaires et d'industriels de l'AFCET.
- **Objectifs** :
  - Simple et accepté par tous.
  - Intelligible pour concepteurs et exploitants.
  - Favorise une réalisation matérielle et logicielle efficace.

Historique du GRAFCET

- En 1975, l'AFCET a étudié les outils de modélisation existants :
  - Organigrammes.
  - Réseaux de Pétri.
  - Graphes d'état.
- En 1977, le GRAFCET a été officiellement défini comme un outil combinant ces approches.
- Publication initiale dans "Automatique et Informatique Industrielle" (1977).

Définition du GRAFCET

- **Objectif** : Modéliser les évolutions séquentielles d'un automatisme.
- **Caractéristiques principales** :
  - Correspondance entre entrées (opérateurs, capteurs) et sorties (commandes).
  - Langage graphique exploitable par les API.
  - Différence entre "GRAFCET" (modèle) et "SFC" (langage de programmation).

Éléments fondamentaux du GRAFCET

1. **Étapes** :
   - États actifs ou inactifs du système.
   - Une étape initiale obligatoire.
   - Représentation par un carré ou carré double pour l'étape initiale.

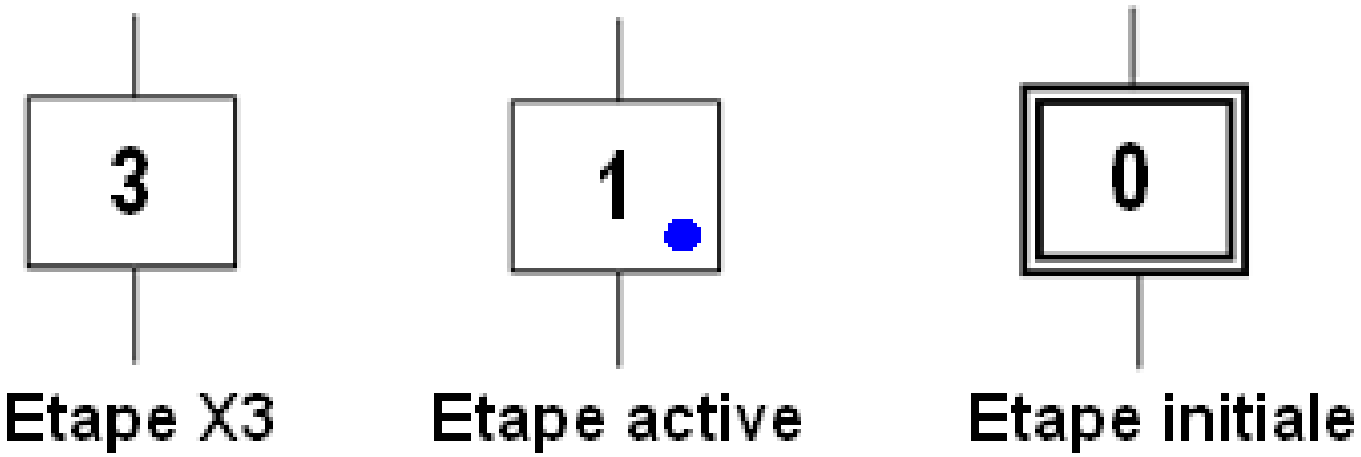


Figure – Représentation d'une étape

2 **Actions associées** :

- Ordres vers la partie opérative ou d'autres grafcets.

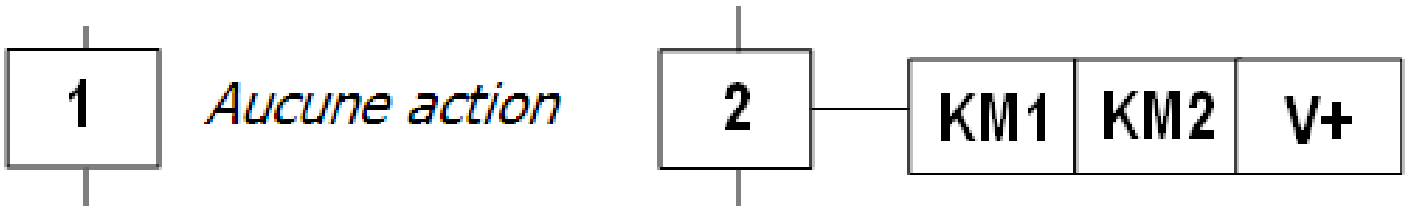


Figure – Actions associées

3 **Transitions** :

- Connectent les étapes et définissent les conditions d'évolution.

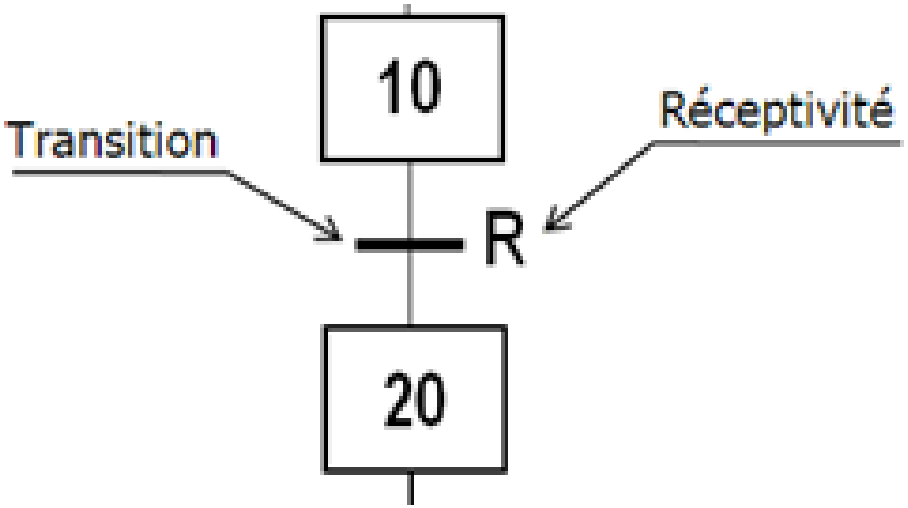


Figure – Représentation d'une transition

4 **Liaisons orientées** :

- Traits verticaux reliant étapes et transitions.

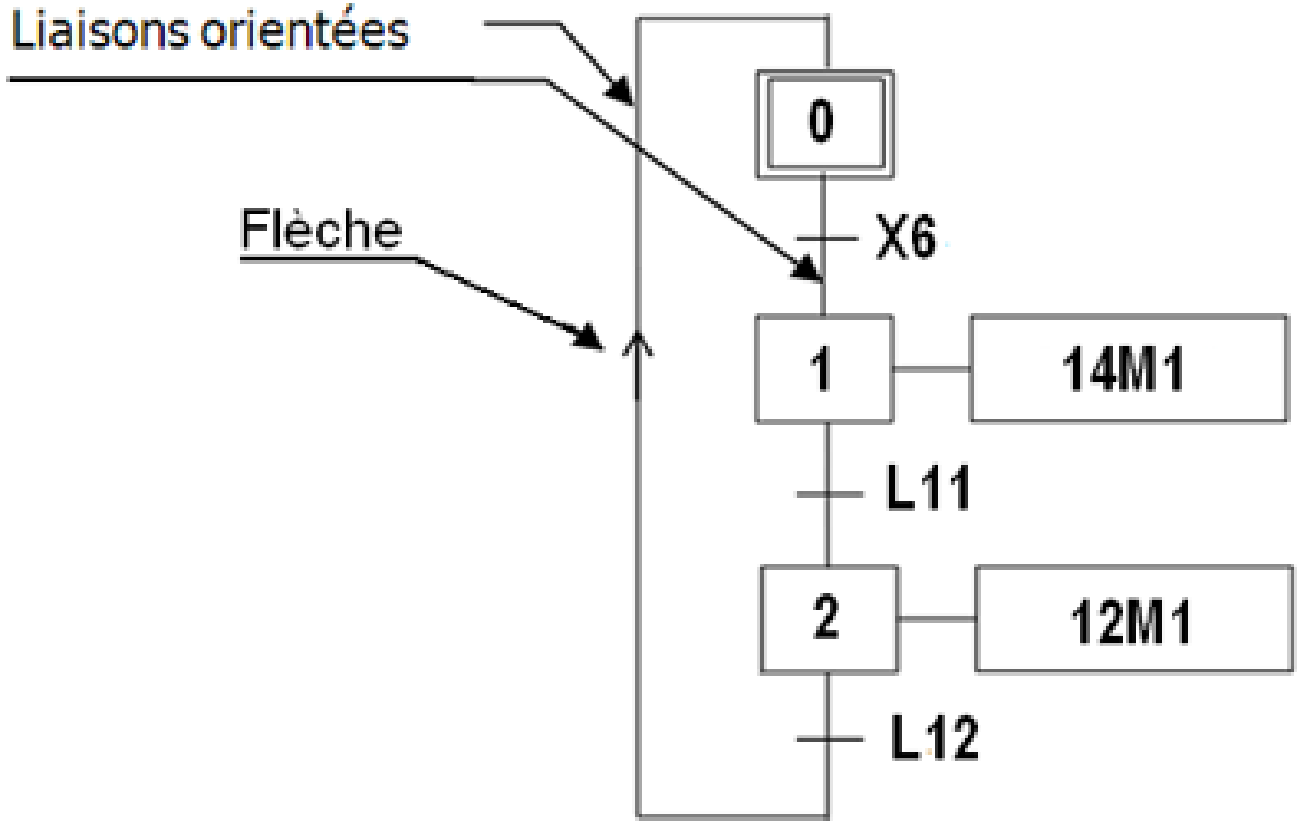


Figure – Liaisons orientées

Types d'actions

1. **Actions continues** :
   - Ordre maintenu tant que l'étape est active.

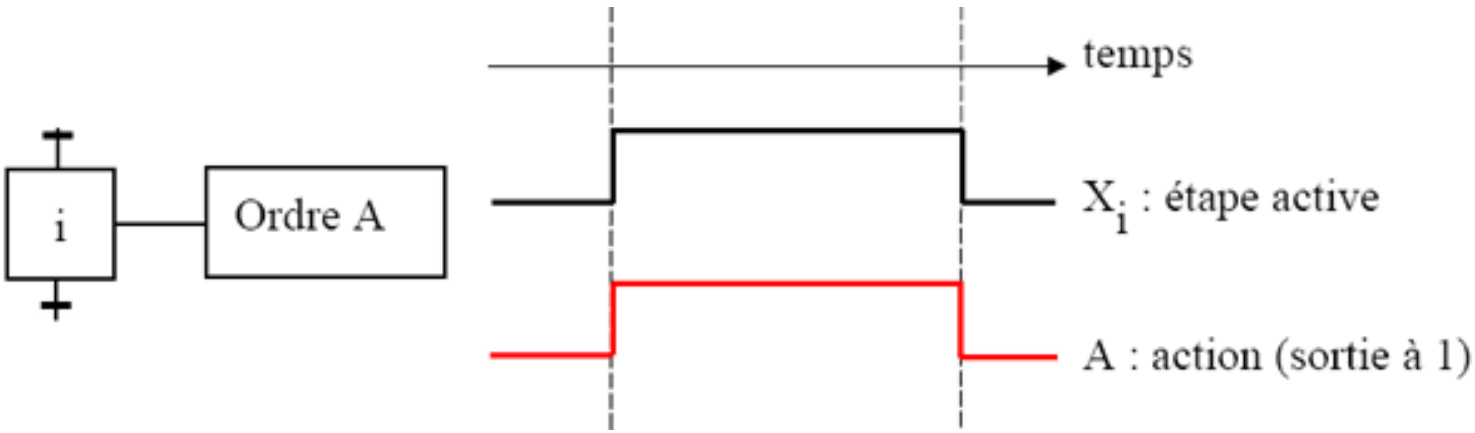


Figure – Actions continues

Types d'actions

2 Actions conditionnelles :

- Ordres exécutés sous certaines conditions :
  - → **Type C : condition simple.**
  - → Type D : condition avec délai.
  - → Type L : durée limitée.

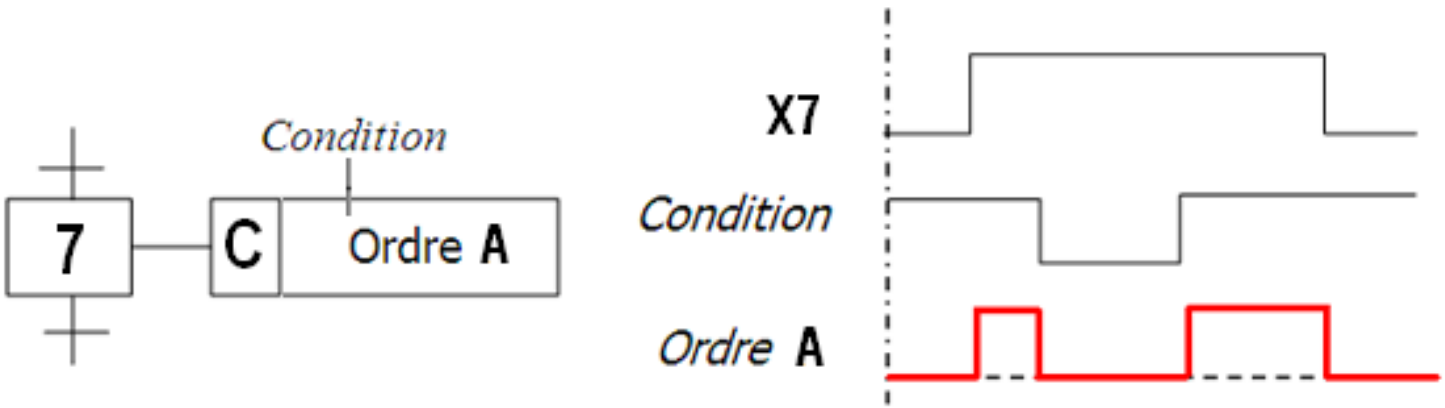


Figure – Actions conditionnelles

Types d'actions

2 Actions conditionnelles :

- Ordres exécutés sous certaines conditions :
  - → Type C : condition simple.
  - → **Type D : condition avec délai.**
  - → Type L : durée limitée.

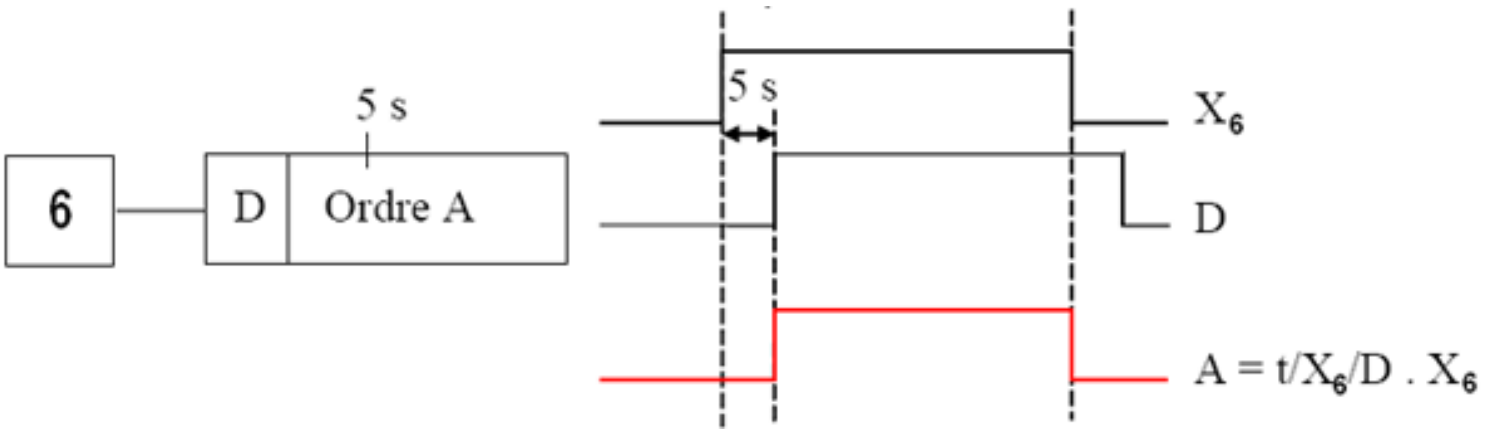


Figure – Actions conditionnelles

Types d'actions

2 Actions conditionnelles :

- Ordres exécutés sous certaines conditions :
  - → Type C : condition simple.
  - → Type D : condition avec délai.
  - → **Type L : durée limitée.**

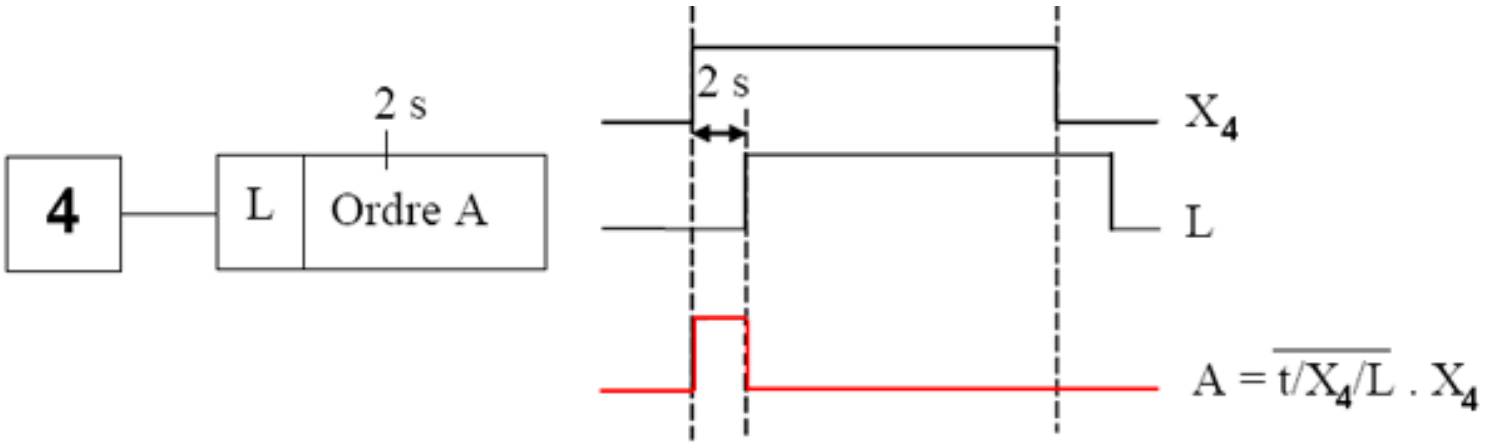


Figure – Actions conditionnelles

Types d'actions

3 **Actions maintenue sur plusieurs étapes** :

- Ordres maintenus sur plusieurs étapes grâce à une fonction mémoire.

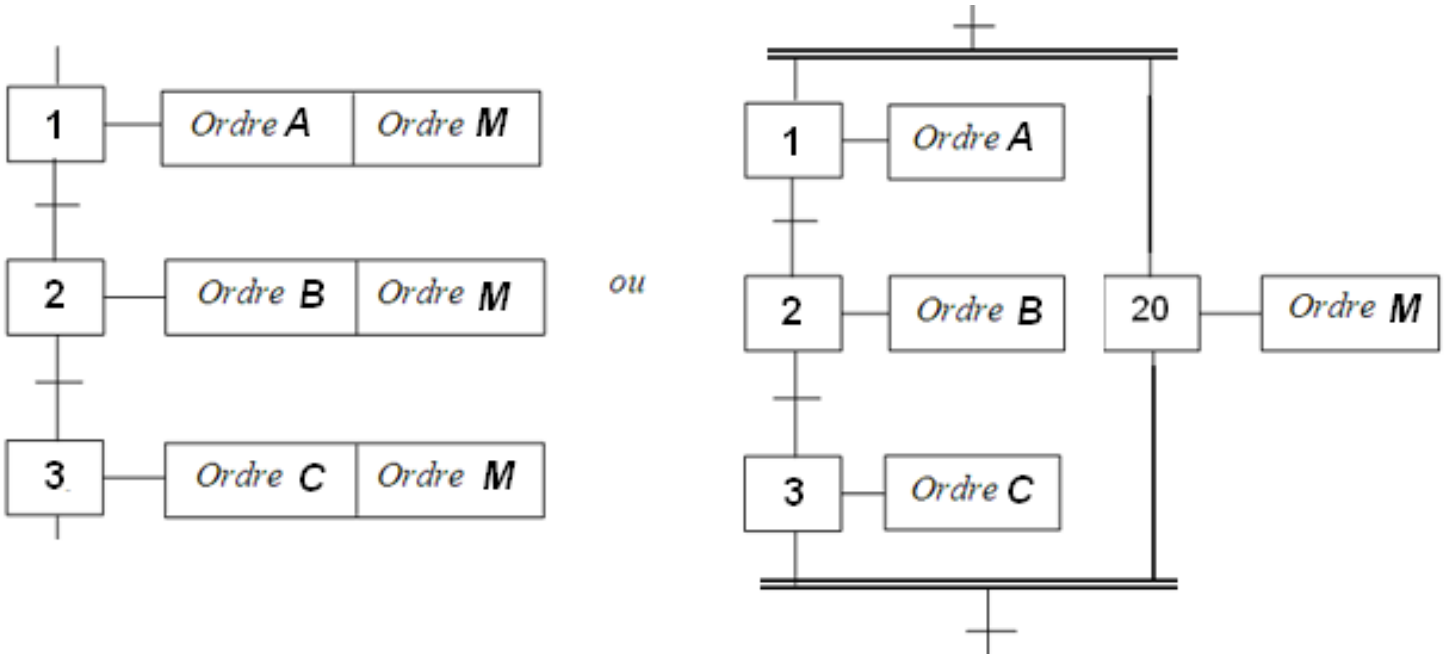


Figure – Action mémorisée

Types d'actions

4 **Actions mémorisées** :

- ▷ Ordres maintenus sur plusieurs étapes grâce à une fonction mémoire.
  - → Type S : SET
  - → Type R : RESET

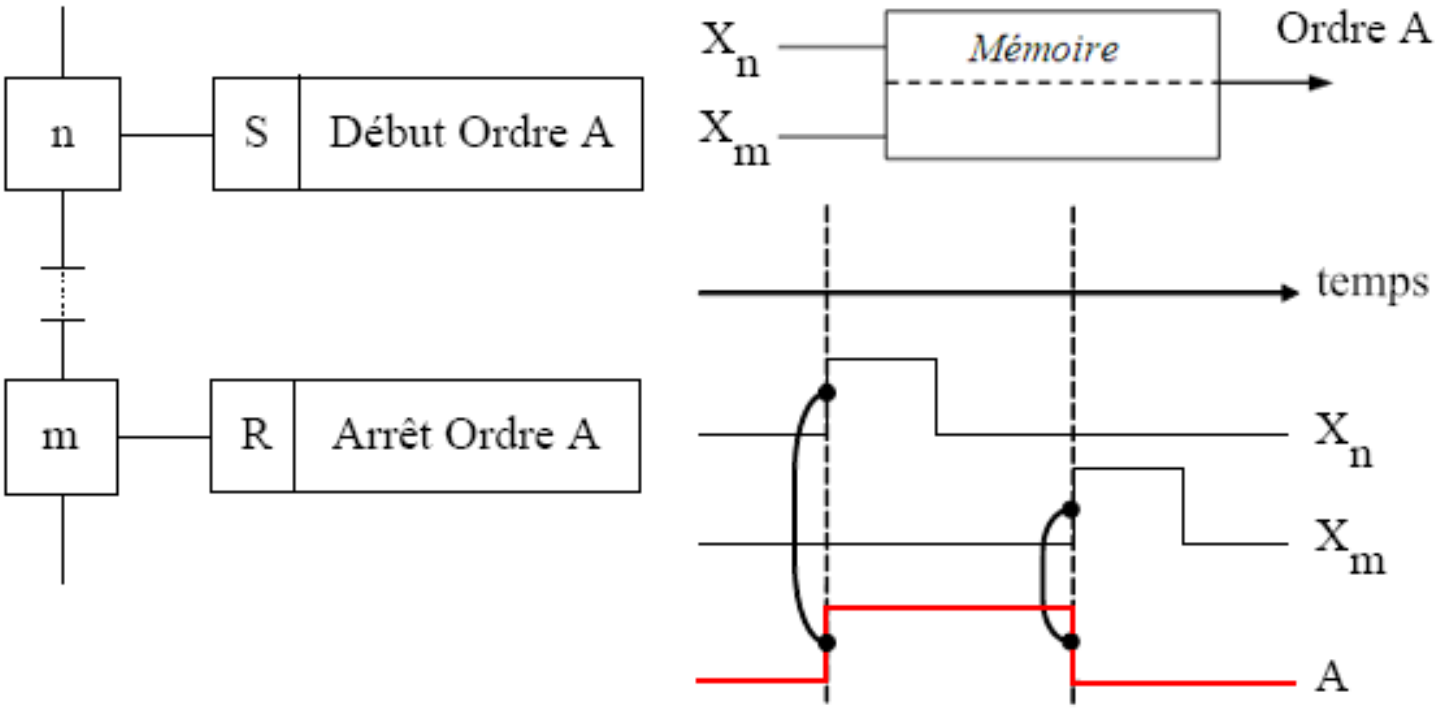


Figure – Action mémorisée

Règles d'évolution du GRAFCET

1. **Condition initiale** :
   - ▷ À l'initialisation, seules les étapes initiales sont actives.

Règles d'évolution du GRAFCET

2 **Franchissement d'une transition** :

- ▷ Transition validée si :
  - → Étapes amont actives.
  - → Réceptivité vraie.

Règles d'évolution du GRAFCET

3 **Évolution des étapes actives** :

- ▷ Désactivation des étapes amont et activation des étapes aval.

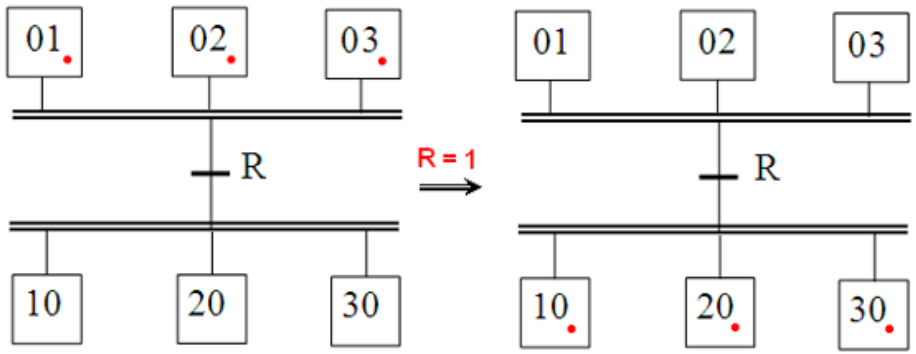


Figure – Évolution des étapes actives

Règles d'évolution du GRAFCET

4. **Franchissement simultané** :
   - ▷ Toutes les transitions franchissables sont simultanément franchies.

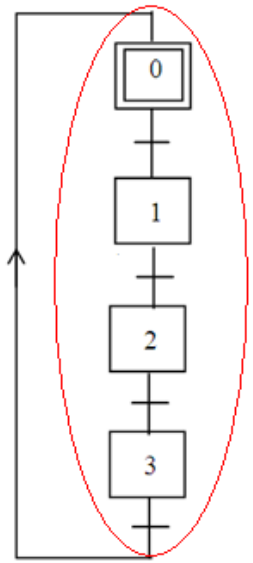
Règles d'évolution du GRAFCET

5 **Conflit d'activation** :

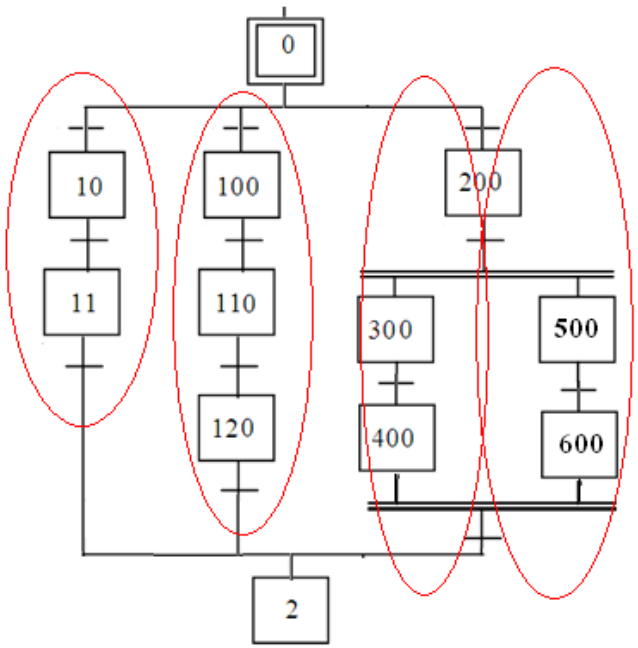
- ▷ Une étape reste active si simultanément désactivée et réactivée.

Notion de séquences

- **Définition** :
  - Les séquenses sont des suites d'étapes à exécuter l'une après l'autre. Autrement dit chaque étape ne possède qu'une seule transition AVAL et une seule transition AMONT
- **Rappel des règles de base** :
  - Une étape est activée uniquement si la transition précédente est validée.
  - Les transitions sont franchies si les réceptivités associées sont vraies.



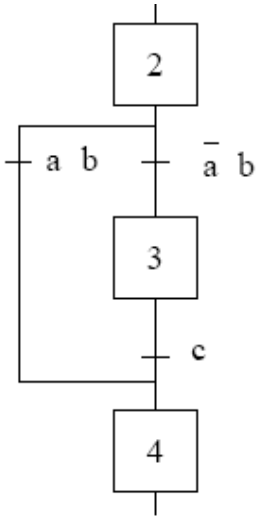
Grafcet à séquence unique.



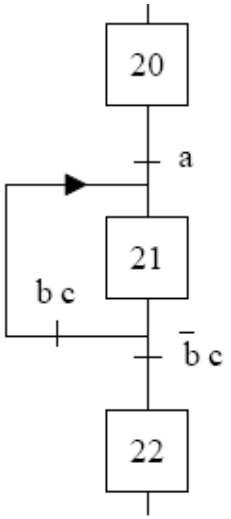
Grafcet à plusieurs séquences.

Saut d'étapes et reprise de séquence

- **Saut d'étapes** :
  - Permet de passer directement d'une étape à une autre sans respecter l'ordre séquentiel habituel.
  - Utilisé pour optimiser les séquences ou répondre à des états exceptionnels.
- **Reprise de séquence** :
  - Retour à une étape précédente pour réinitialiser ou redémarrer une séquence.



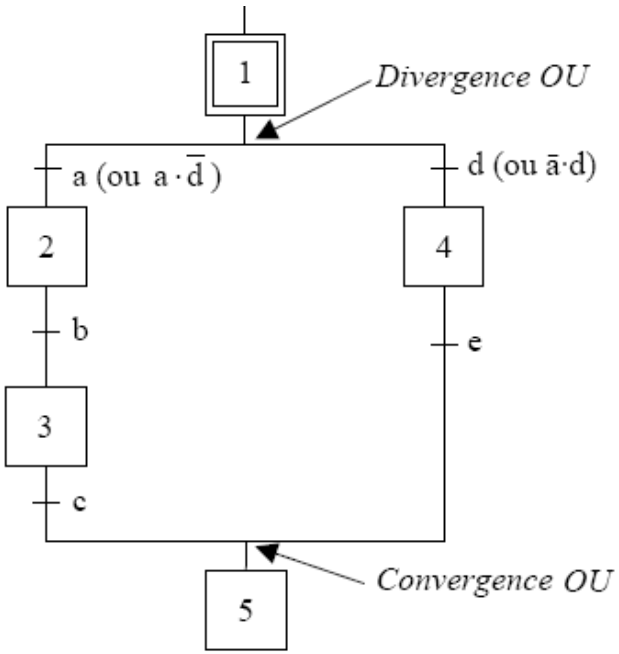
Saut d'étape



Reprise de séquence

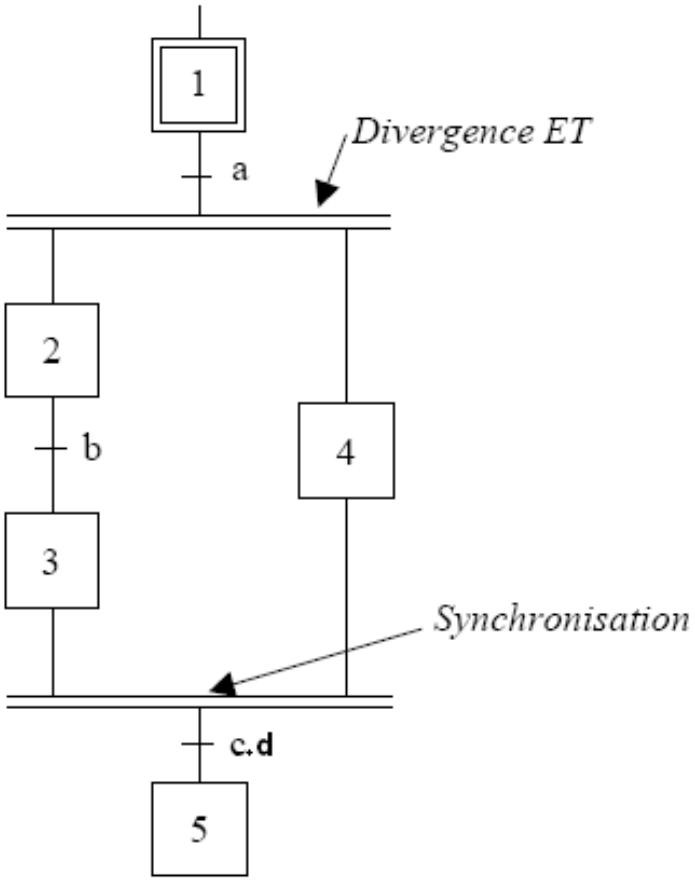
Aiguillage entre deux ou plusieurs séquences (Divergence en OU)

- **Définition** :
  - Une divergence en **OU** (exclusif) permet de choisir entre plusieurs chemins selon des conditions spécifiques.
- **Fonctionnement** :
  - Une transition unique mène à plusieurs branches possibles.
  - La réceptivité de chaque branche détermine le chemin suivi.



Parallélisme entre deux ou plusieurs séquences

- **Définition** :
  - Une divergence en **ET** permet d'exécuter deux chemins en parallele.
  - La synchronisation permet d'attendre la fin de plusieurs activités se déroulant en parallèle, pour continuer par une seule.



Programmation d'automates en langage Ladder

Concept et historique du langage Ladder

- **Définition** : Également appelé langage à contacts ou schéma à relais, un langage graphique standardisé par la norme CEI 61131-3.
- **Origine** : Inspiré des schémas électriques, conçu pour programmer des automates industriels.
- **Caractéristiques** : Basé sur des contacts (ouverts ou fermés) et des bobines pour représenter des états logiques.

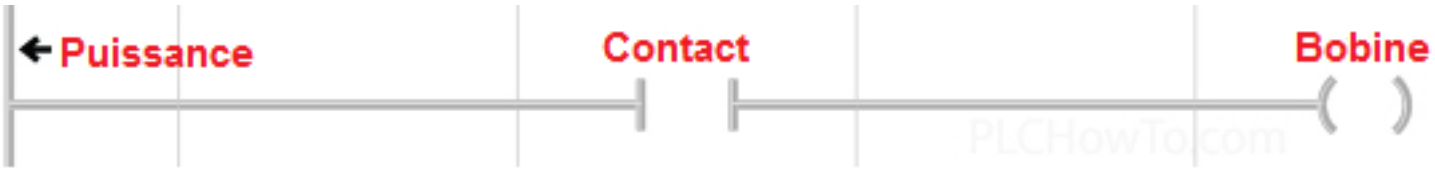
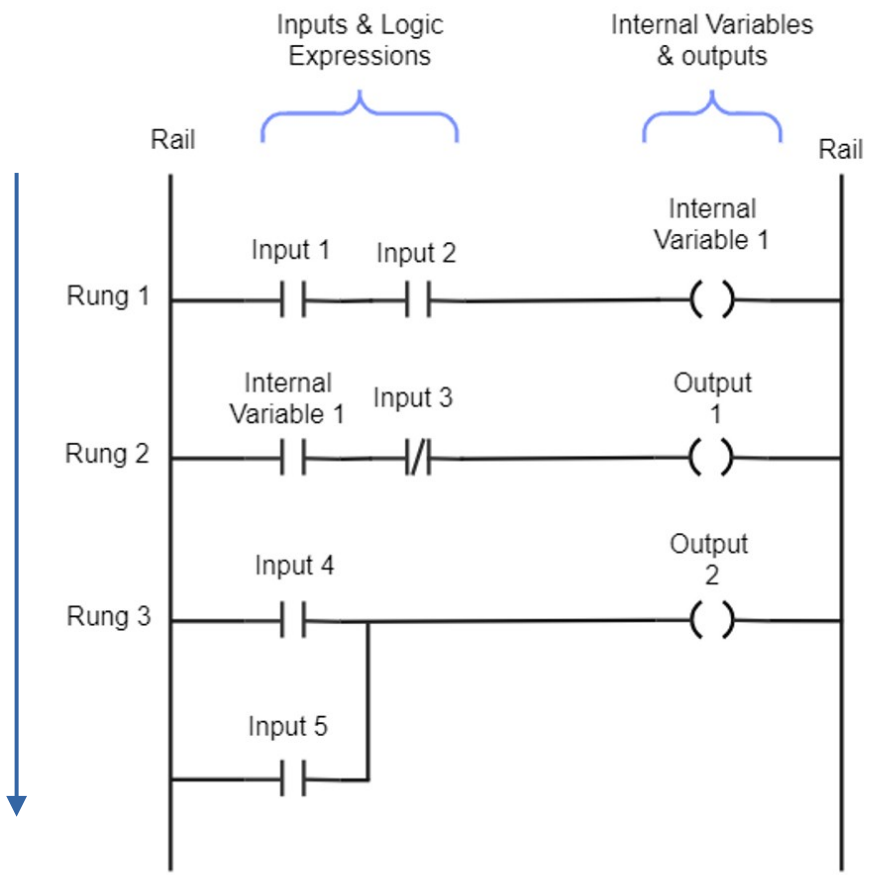


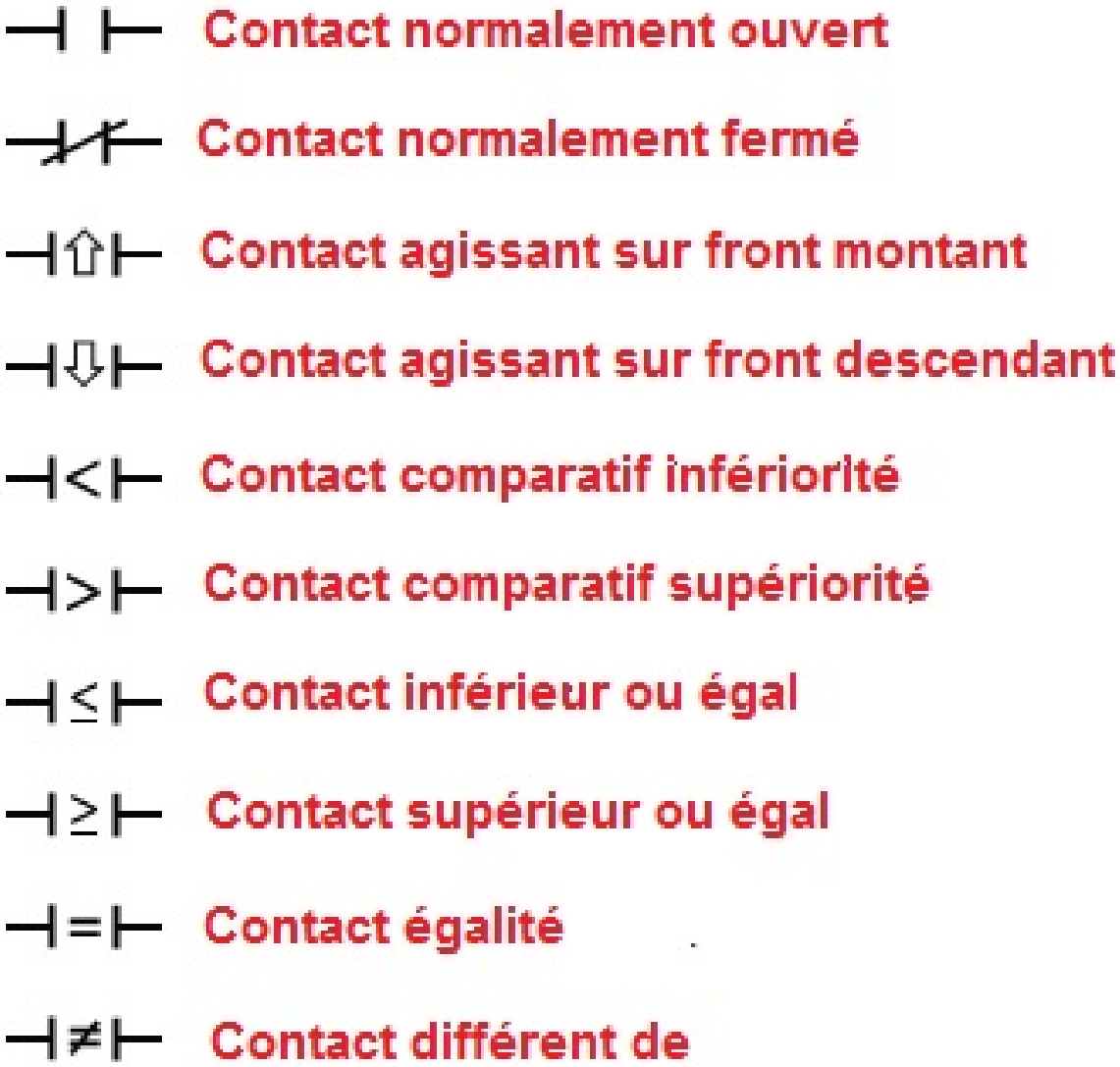
Figure – Concept du langage Ladder

Ordre d'exécution

- Les *rung* sont évalués de gauche à droit et de haut en bas.
- Les sorties sont affectées après évaluation du dernier rung.
- On peut avoir des sorties du rung qui annulent les sorties précédentes !



Les différents types de contacts



Les temporisateurs

- **IN** : Signal d'entrée
- **PT** : Durée de la temporisation
- **Q** : Signal de sortie
- **ET** : Temps écoulé (principalement pour du monitoring)

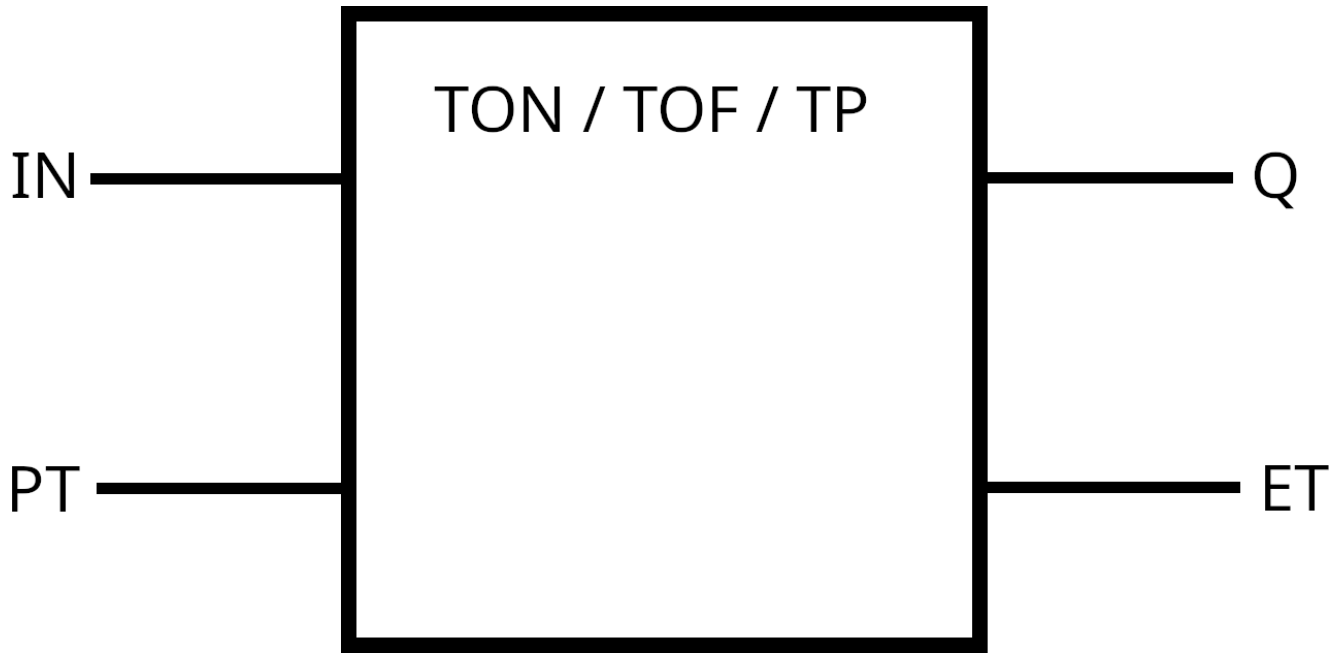
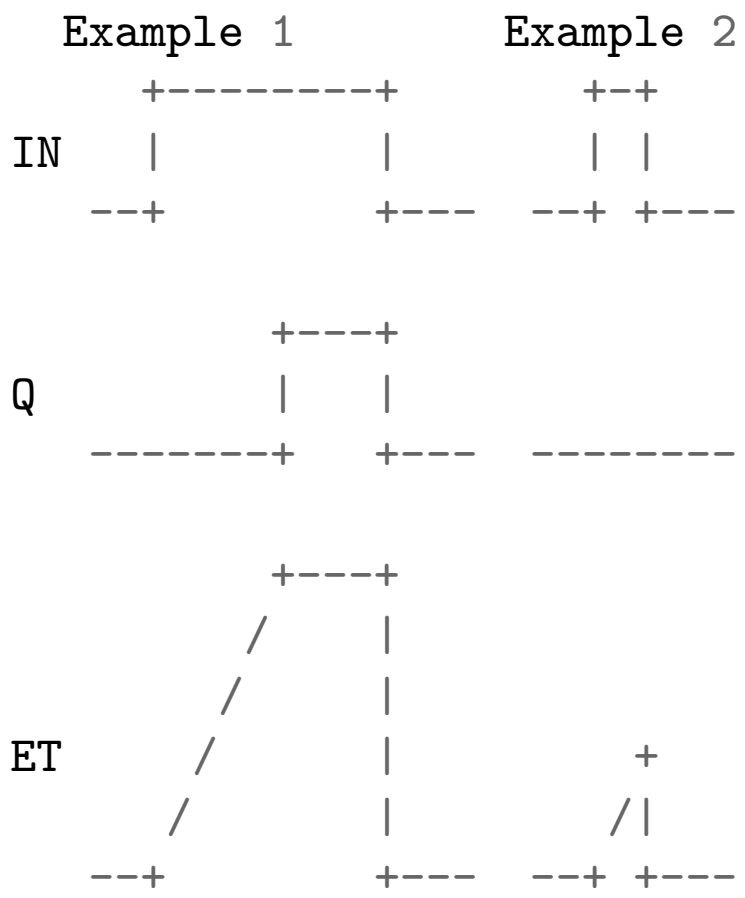


Figure – Schéma global d'un temporisateur

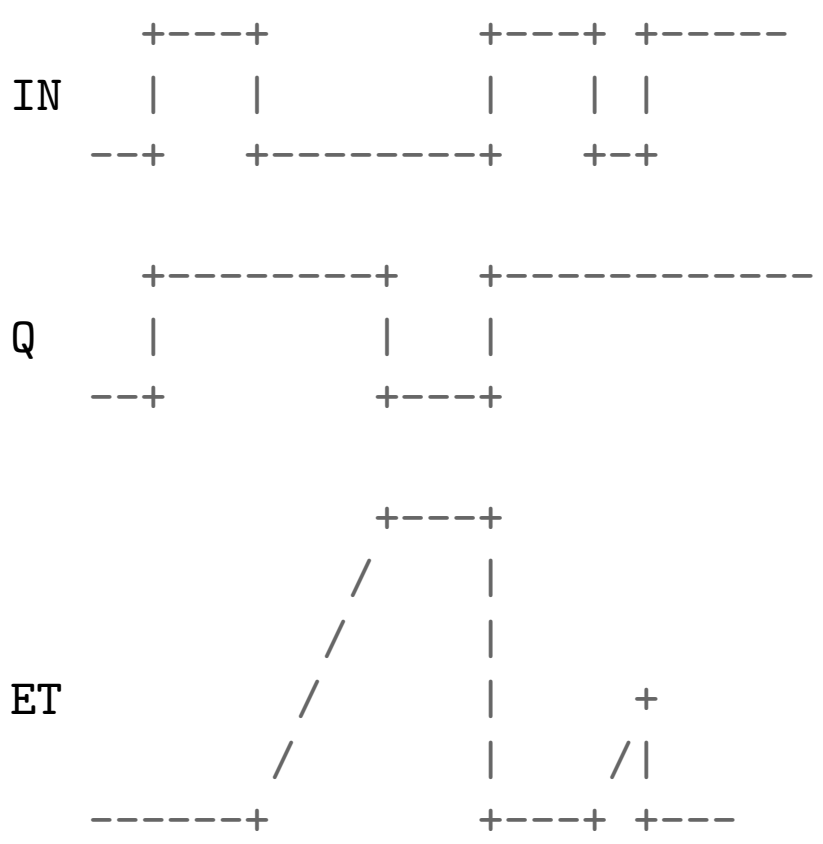
Temporisateur TON – Sortie retardée

- Si IN == FALSE :
  - Q := FALSE
  - ET := T#0s
- Si IN == TRUE :
  - ET va augmenter jusqu'à atteindre PT
  - Quand ET == PT :
    - → Q := TRUE



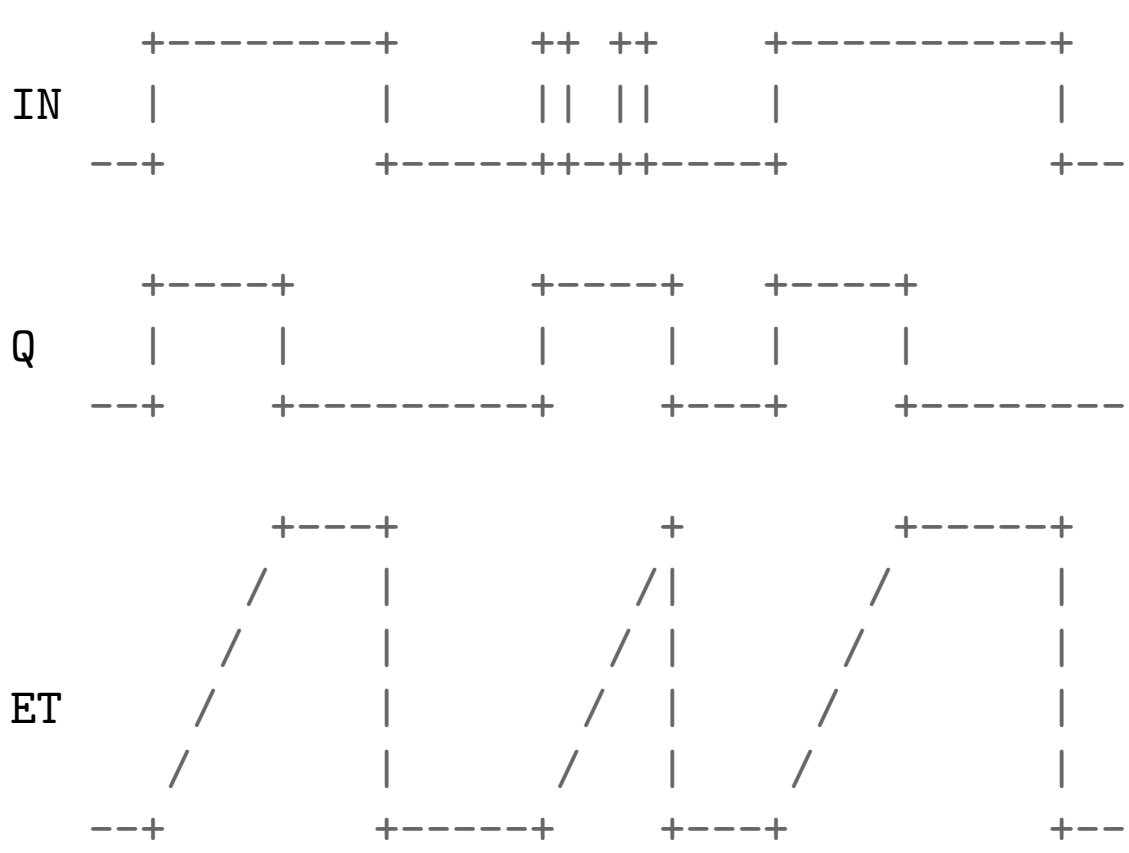
Temporisateur TOF – Sortie maintenue

- Si IN == TRUE :
  - Q := TRUE
  - ET := T#0s
- Si IN := FALSE :
  - ET va augmenter jusqu'à atteindre PT
  - Quand ET == PT :
    - → Q := FALSE



Temporisateur TP – Sortie à impulsion

- Si IN := TRUE :
  - Q := TRUE
  - ET := T#0s
- ET va augmenter jusqu'à atteindre PT (valeur de IN ignorée)
  - Quand ET == PT :
    - → Q := FALSE



Les différents types de bobines

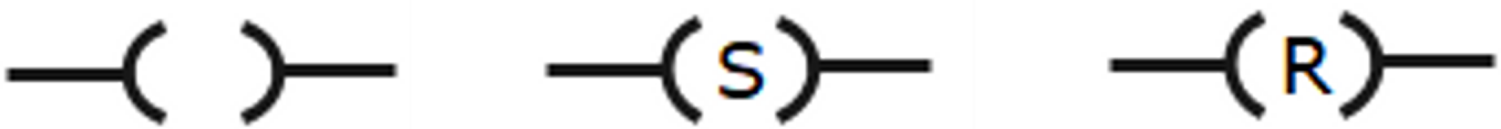


Figure – Exemples de bobines

- **Bobine vide** : Sortie immédiate
- **Bobine set** : Sortie maintenue
- **Bobine reset** : Sortie annulée

Opération logique ET (AND)

- **Principe** : Tous les contacts en série doivent être activés pour que la bobine s'active.
- **Schéma logique** :
  - Chaque contact représente une condition.
  - Le courant circule uniquement si toutes les conditions sont remplies (imaginez un circuit électrique).
- **Exemple pratique** : Activation d'une machine nécessitant plusieurs prérequis (sécurité, disponibilité de matériel).

Figure – Exemple de logique AND

Opération logique OU (OR)

- **Principe** : Un seul des contacts en parallèle doit être activé pour que la bobine s'active.
- **Schéma logique** :
  - Les contacts représentent des conditions alternatives.
  - Le courant circule dès qu'une des conditions est remplie (imaginez un circuit électrique).
- **Exemple pratique** : Éclairage contrôlé par plusieurs interrupteurs.

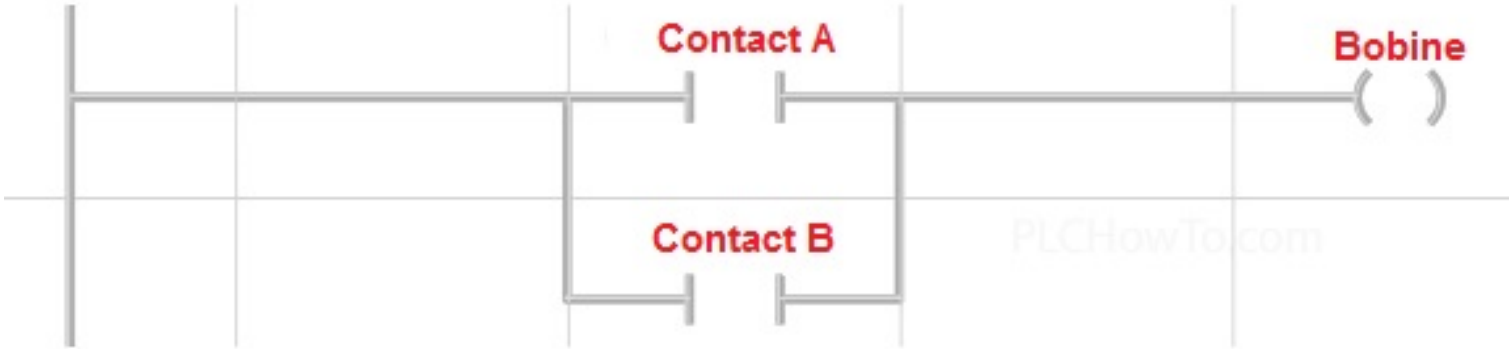


Figure – Exemple de logique OR

Références

Références utilisées dans le cours

- https://www.fernhillsoftware.com/help/iec-61131/common-elements
- https://www.technologuepro.com/cours-genie-electrique/cat-3-cours-automatisme-informatique-industrielle/