

# **Méthodologie de programmation – Norme IEC61499**

## **Projet IEC61499 – Universal Automation**

*Formation Informatique et Systèmes  
Intelligents Embarqués*  
**Année 2025 – 2026**

## PRI 5A

### Membres de l'équipe :

Damien LORIGEON – Chef de projet/Dev IEC61131 & IEC61499

### Client :

Jean Paul CHEMLA – Professeur Polytech  
Arthur OUSSOUNKIRI ELIEZER GAMBO – Doctorant Université de Reims  
Bernard RIERA – Professeur Université de Reims  
Stéphane LECASSE – Professeur Université de Reims

## Objectifs

Ce document détaille la méthodologie complète de développement d'une application de contrôle conforme à la norme IEC 61499 sous EcoStruxure Automation Expert (EAE), appliquée à une usine virtuelle de tri de pièces simulée dans Factory I/O. Visant à comparer les approches de programmation IEC 61131-3 et IEC 61499, ce guide a pour vocation de servir de référence pédagogique pérenne en fournissant une démarche structurée qui explicite la transition du GRAFCET vers une architecture événementielle, tout en documentant les bonnes pratiques de conception nécessaires à la reproduction intégrale du programme.

## Référence

### 1. Internes

Référence :	Titre	Lien
PRI Polytech Tours	Projet IEC61499 – Universal Automation	Dépôt GIT : <a href="https://github.com/EIDLOR/PRI-IEC61499-UniversalAutomation.git">https://github.com/EIDLOR/PRI-IEC61499-UniversalAutomation.git</a> Équipe Teams : <a href="#">Général   Projet - Universal Automation - IEC 61499   Microsoft Teams</a>

### 2. Externes

Référence :	Titre	Lien
UniversalAutomation.org	Ressources IEC 61499 / EAE	<a href="https://universalautomation.org">https://universalautomation.org</a>

## Définition

- **IEC 61131 (ECE)** : norme historique de programmation des automates (POU, Grafcet, ST, etc.).
- **IEC 61499 (EAE)** : norme orientée événements, blocs fonctionnels distribués.
- **Factory IO** : outil de simulation 3D de systèmes industriels.
- **EAE** : EcoStruxure Automation Expert.
- **ECE** : EcoStruxure Control Expert.

# Méthodologie de programmation - Norme IEC61499 pour le projet

« Projet IEC61499 – Universal Automation »

Descriptions		
Projet :	Projet IEC61499 – Universal Automation	
Clients	Jean Paul CHEMLA	jean-paul.chemla@univ-tours.fr
Auteurs	Damien LORIGEON	damien.lorigeon@univ-tours.fr
Date d'émission :	27/01/2026	

Validation			
Nom	Date	Valide (O/N)	Commentaires
LORIGEON	26/01/2026	O	
CHEMLA			
RIERA			
GAMBO			

Suivis des versions		
Version	Date	Description de la modifications
1	26/01/2026	Première version

## Table des matières

Table des matières .....	5
1. Introduction et Objectifs .....	7
1.1 Contexte du projet .....	7
1.2 Objectifs du document .....	7
1.3 Portée de l'application .....	7
2. Prérequis et Environnement .....	8
2.1 Logiciels requis .....	8
2.2 Configuration réseau .....	8
3. Méthodologie Générale : Du GRAFCET à IEC 61499 .....	9
3.1 Rôle du GRAFCET G7 dans la conception .....	9
3.2 Principes de traduction GRAFCET → IEC 61499 .....	10
3.3 Décomposition modulaire .....	10
4. Architecture du Programme EAE .....	11
4.1 Structure hiérarchique .....	11
4.2 Types de blocs développés .....	12
5. Conception des Function Blocks Basiques .....	13
5.1 FB_EntryLogic - Logique Convoyeur d'Entrée .....	13
5.2 FB_VisionSensor - Capteur de Vision .....	15
5.3 FB_TriLogic - Logique de Tri .....	16
5.4 FB_SorterLogic - Logique Trieur .....	18
5.5 Interface Adaptateur ICouleur .....	18
6. Conception des Composites (EQUIP) .....	19
6.1 Principe des blocs composites .....	19
6.2 EQUIP_EntryConveyor - Équipement Convoyeur d'Entrée .....	19
6.3 EQUIP_VisionSensor - Équipement Capteur Vision .....	19
6.4 EQUIP_TRIManager - Équipement Gestionnaire de Tri .....	20
6.5 EQUIP_Sorter - Équipement Trieur .....	20
6.6 EQUIP_ExitConveyor - Équipement Convoyeur de Sortie .....	21
7. Configuration Système et Communication .....	22
7.1 Mapping E/S Modbus .....	22
7.2 Blocs SYMLINK .....	22
8. Chaînes d'Événements .....	23
8.1 Chaîne d'initialisation (INIT) .....	23
8.2 Chaîne cyclique (REQ/CNF) .....	23
8.3 Événements process .....	24
9. Guide de Reproduction du Programme .....	24

9.1 �tapes de cr�ation dans EAE .....	24
9.2 Test et validation .....	25
Annexes : Emplacements des Sch�mas .....	25

# 1. Introduction et Objectifs

## 1.1 Contexte du projet

Ce document présente la méthodologie complète de développement d'une application conforme à la norme IEC 61499 sous EcoStruxure Automation Expert (EAE). L'application réalise le contrôle d'une usine virtuelle de tri de pièces par couleur simulée dans Factory I/O. Cette méthodologie a été développée dans le cadre du projet PRI IEC61499 - Universal Automation à Polytech Tours, avec pour objectif de comparer les approches IEC 61131-3 et IEC 61499 pour la programmation d'automatismes.

## 1.2 Objectifs du document

Ce guide méthodologique a pour objectifs :

- Fournir une démarche structurée pour développer des applications IEC 61499
- Expliquer comment traduire un GRAFCET en architecture événementielle
- Documenter les bonnes pratiques de conception sous EAE
- Permettre la reproduction complète du programme développé
- Servir de référence pour de futurs projets pédagogiques

## 1.3 Portée de l'application

L'usine de tri traite des pièces de trois couleurs (Bleu, Vert, Gris) et les dirige vers trois sorties distinctes via des aiguilleurs (Sorters). Les particularités fonctionnelles sont :

- Tri par couleur basé sur un capteur de vision (valeurs 1-9)
- Gestion des pièces consécutives de même couleur (pas de blocage)
- Blocage de l'entrée uniquement si la couleur change
- Gestion de l'arrêt d'urgence (AU), Stop et Reset

## 2. Pr requis et Environnement

### 2.1 Logiciels requis

Logiciel	Version	R�le
<b>EcoStruxure Automation Expert</b>	24.1	D�veloppement IEC 61499
<b>Factory I/O</b>	2.5.10	Simulation 3D de l'usine
<b>Soft dPAC Runtime</b>	Int�gr� EAE	Ex�cution du programme

### 2.2 Configuration r seau

La communication entre EAE et Factory I/O utilise le protocole Modbus TCP :

- Adresse IP : 10.172.6.93
- Port : 502
- Station ID : 1
- Timeout r ponse : 500ms
- Cycle bus : 200ms

**Important :** Dans cette configuration, Factory I/O agit en SERVEUR Modbus et EAE en CLIENT. C'est l'inverse de la configuration IEC 61131-3 avec Control Expert.



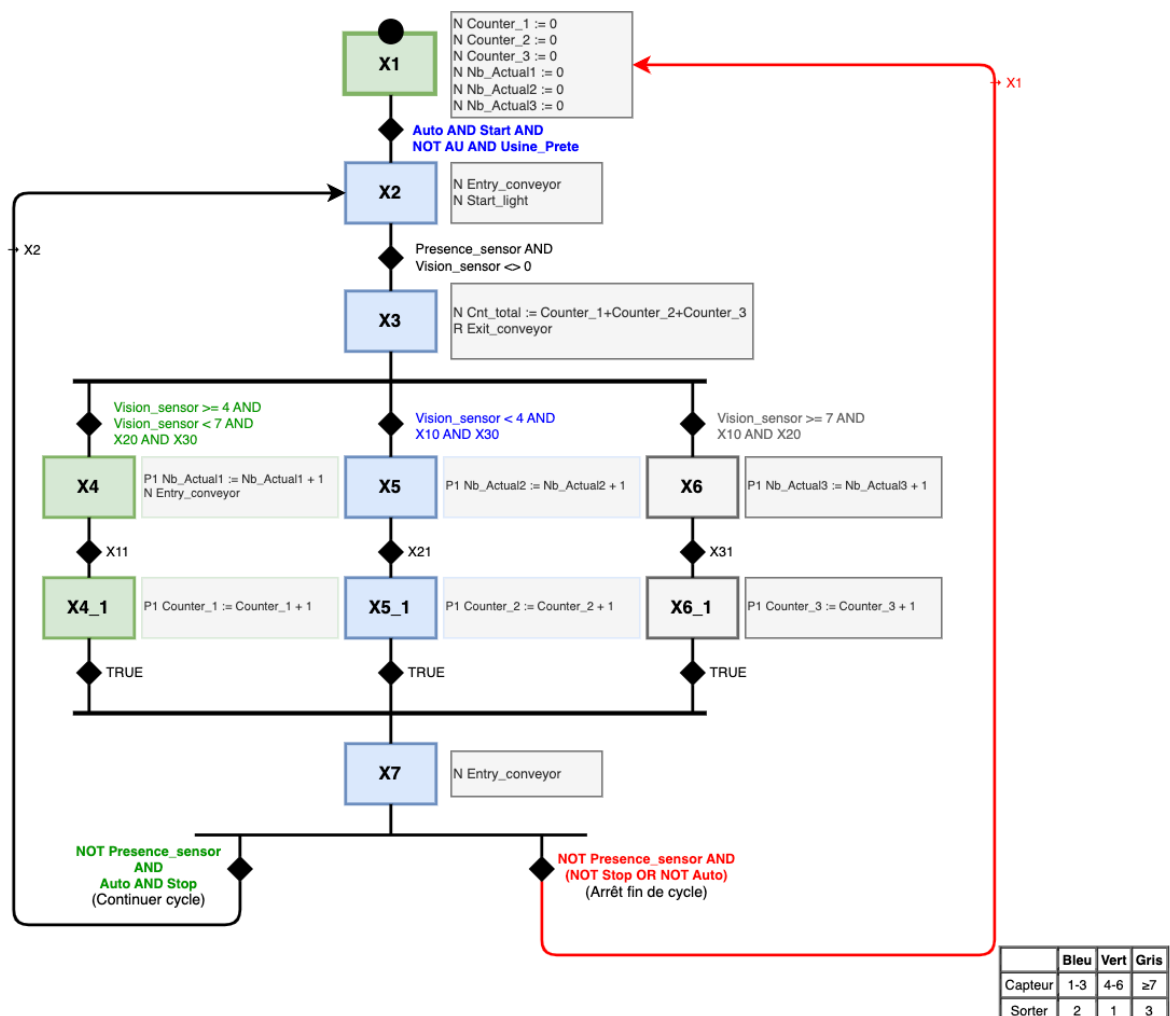
### 3. Méthodologie Générale : Du GRAFCET à IEC 61499

#### 3.1 Rôle du GRAFCET G7 dans la conception

Le GRAFCET niveau 2 (G7) a servi de modèle de référence pour les deux implémentations (IEC 61131-3 et IEC 61499). Bien que l'approche événementielle de IEC 61499 diffère fondamentalement de l'approche cyclique de IEC 61131-3, le GRAFCET reste pertinent comme outil de conception pour :

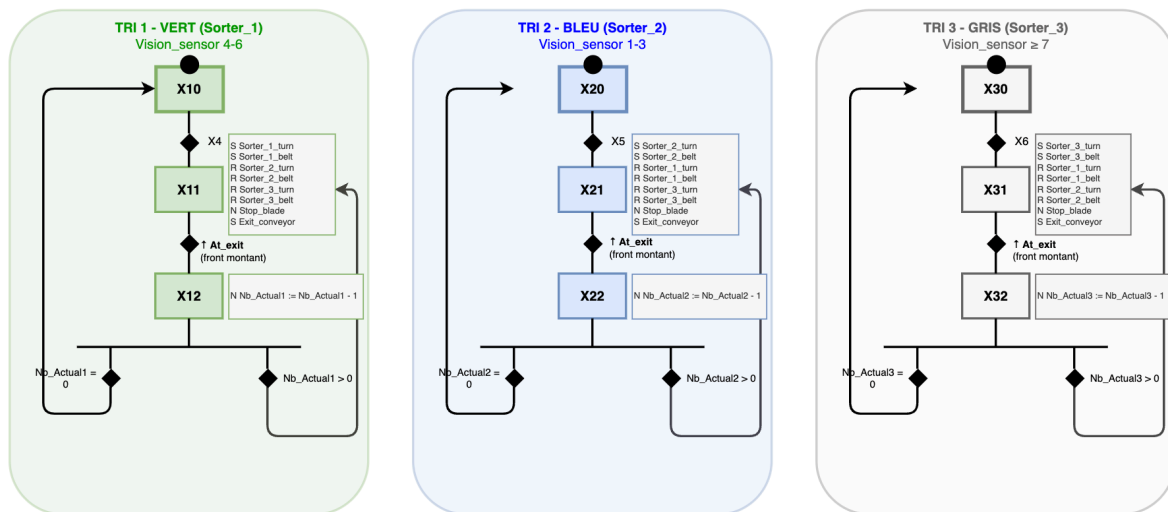
- Identifier les états du système (étapes)
- Définir les conditions de transition entre états
- Spécifier les actions associées à chaque état
- Visualiser la logique séquentielle globale

GRAFCET Niveau 2 : G7 Principal - UsineDeTri (ECE)



[SCHÉMA : G7 Principal - fichier G7\_NIVEAU2\_PRI\_PROJET-IEC61499\_V0\_2.drawio, onglet 'G7 Principal']

**GRAFCET Niveau 2 : Sous-GRAFCET de Tri (G7 Tri)**  
 3 grafjets parallèles pour le tri des pièces



[SCHÉMA : G7 Tri - fichier G7\_NIVEAU2\_PRI\_PROJET-IEC61499\_V0\_2.drawio, onglet 'G7 Tri']

## 3.2 Principes de traduction GRAFCET → IEC 61499

La traduction du GRAFCET en IEC 61499 suit ces principes :

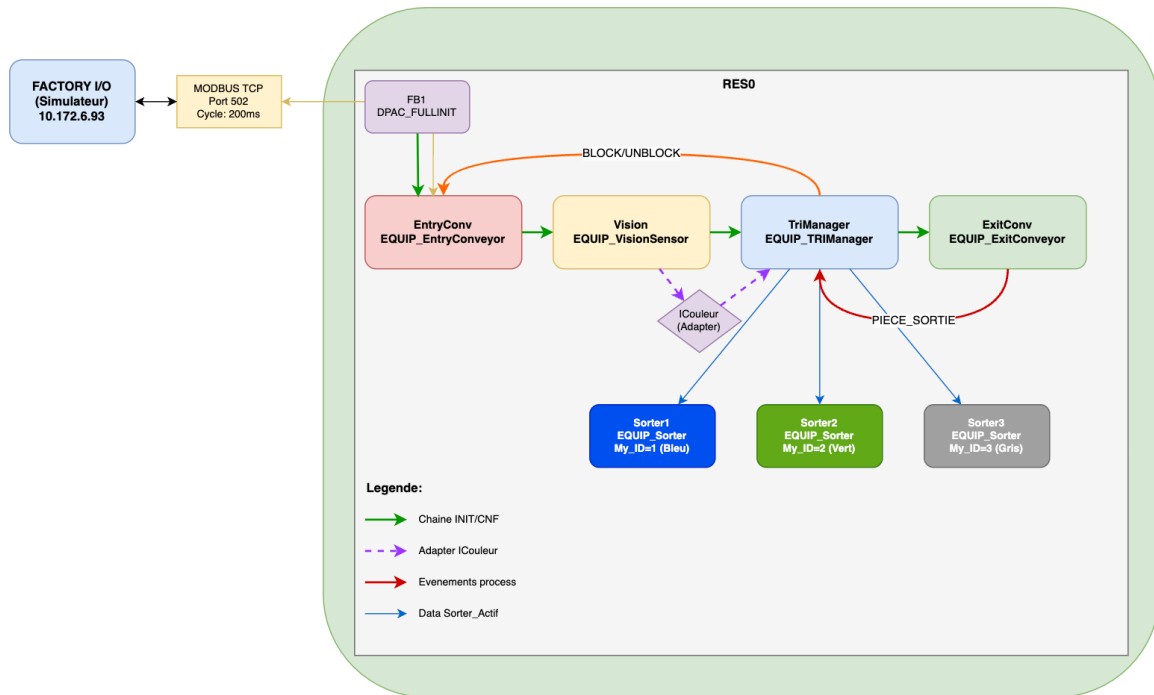
Concept GRAFCET	Équivalent IEC 61499	Implémentation EAE
Étape	État dans le diagramme ECC	État du FB avec algorithme associé
Transition	Événement + Condition	Transition ECC avec garde
Action	Algorithme	Code ST dans l'algorithme
Sous-GRAFCET parallèle	Function Block séparé	EQUIP_Sorter (instancié 3x)
Synchronisation	Connexions événementielles	BLOCK/UNBLOCK, PIECE_SORTIE

## 3.3 Décomposition modulaire

Le système a été décomposé en modules fonctionnels correspondant aux équipements physiques :

- Module Entrée (EntryConveyor)** : Gère le convoyeur d'entrée, la lame de blocage, les boutons IHM
- Module Détection (VisionSensor)** : Lit le capteur de vision et détermine la couleur
- Module Tri (TRIManager)** : Gère la logique de tri et le blocage/déblocage
- Module Sortie (ExitConveyor)** : Gère le convoyeur de sortie et détecte les pièces sorties
- Modules Aiguilleurs (Sorter 1/2/3)** : Contrôlent chaque aiguilleur individuellement

## ARCHITECTURE GENERALE - USINE DE TRI IEC 61499



[SCHÉMA : Architecture Générale – fichier SCHEMAS\_IEC61499\_PRI\_PROJET-IEC61499\_V0\_1.drawio, onglet '1-Architecture Generale']

## 4. Architecture du Programme EAE

### 4.1 Structure hiérarchique

L'application EAE est organisée selon la hiérarchie IEC 61499 :

**System** (niveau supérieur)

**Device : SE.DPAC** (Soft dPAC)

**Resource : RES0** (EMB\_RES\_ECO)

**Application : APP1**

FB1 (DPAC\_FULLINIT)  
EntryConv (EQUIP\_EntryConveyor)  
Vision (EQUIP\_VisionSensor)  
TriManager (EQUIP\_TRIManager)  
ExitConv (EQUIP\_ExitConveyor)  
Sorter1 (EQUIP\_Sorter)  
Sorter2 (EQUIP\_Sorter)  
Sorter3 (EQUIP\_Sorter)

## 4.2 Types de blocs d velopp s

Le programme utilise trois cat gories de blocs :

**Function Blocks Basiques :**

- FB\_EntryLogic : Logique du convoyeur d'entr e
- FB\_VisionSensor : Traitement du capteur de vision
- FB\_TriLogic : Logique de tri et gestion du blocage
- FB\_SorterLogic : Contr le individuel d'un sorter

**Function Blocks Composites :**

- EQUIP\_EntryConveyor, EQUIP\_VisionSensor, EQUIP\_TRIManager, EQUIP\_Sorter, EQUIP\_ExitConveyor

**Adapter :**

- ICouleur : Interface de communication entre VisionSensor et TRIManager

## 5. Conception des Function Blocks Basiques

### 5.1 FB\_EntryLogic - Logique Convoyeur d'Entr e

**Description :** G re le convoyeur d'entr e des pi ces, la d tection de pr sence, la lame de blocage, les voyants IHM et l'arr t d'urgence.

####  v nements d'entr e :

- **INIT** : Initialisation du bloc
- **REQ** : Requite d'ex cution cyclique
- **BLOCK** : Commande de blocage de l'entr e (lame lev e)
- **UNBLOCK** : Commande de d blocage (lame baiss e)

####  v nements de sortie :

- **INITO** : Confirmation d'initialisation
- **CNF** : Confirmation d'ex cution
- **PIECE\_PRESENTE** : Nouvelle pi ce d tect e
- **AU\_OUT** : Signal d'arr t d'urgence propag 
- **RESET\_OUT** : Signal de reset propag 

#### Variables d'entr e :

- **QI (BOOL)** : Qualificateur d' v nement
- **Auto (BOOL)** : Mode automatique
- **Start (BOOL)** : Bouton d marrage
- **Stop (BOOL)** : Bouton arr t (NF)
- **AU (BOOL)** : Arr t d'urgence
- **Reset (BOOL)** : Bouton reset
- **Presence\_sensor (BOOL)** : Capteur de pr sence pi ce

#### Variables de sortie :

- **QO (BOOL)** : Qualificateur de sortie
- **Entry\_conveyor (BOOL)** : Commande convoyeur entr e
- **Stop\_blade (BOOL)** : Commande lame de blocage
- **Start\_light (BOOL)** : Voyant d marrage
- **Reset\_light (BOOL)** : Voyant reset
- **Stop\_light (BOOL)** : Voyant arr t

####  tats du diagramme ECC :

- **START → INIT** : Initialisation (ACT\_INIT)
- **RUNNINGG** : Convoyeur en marche (ACT\_RUN)
- **DETECT** : Pi ce d tect e →  met PIECE\_PRESENTE
- **BLOCKED** : Entr e bloqu e (ACT\_BLOCKED)
- **UNBLOCKING** : D blocage en cours (ACT\_UNBLOCK)
- **AU\_STATE** : Arr t d'urgence actif (ACT\_AU)
- **RESET\_STATE** : Reset en cours (ACT\_RESET)
- **STOPPEDD** : Usine arr t e (ACT\_STOPPED)



## 5.2 FB\_VisionSensor - Capteur de Vision

**Description :** Gère le capteur de vision pour la détection et l'identification des couleurs des pièces. Convertit les valeurs brutes du capteur en codes couleur et détecte les changements de couleur entre pièces successives.

### Événements d'entrée :

- **INIT** : Initialisation du bloc
- **REQ** : Requête d'exécution cyclique
- **DETECT** : Pièce présente, lecture de la couleur

### Événements de sortie :

- **INITO** : Confirmation d'initialisation
- **CNF** : Confirmation d'exécution
- **NEW\_COLOR** : Nouvelle couleur détectée (avec indication de changement)

### Variables d'entrée :

- **QI (BOOL)** : Qualificateur d'événement
- **Vision\_value (INT)** : Valeur brute du capteur de vision (0-9)

### Variables de sortie :

- **QO (BOOL)** : Qualificateur de sortie
- **Couleur (INT)** : Code couleur (0=Aucune, 1=Bleu, 2=Vert, 3=Gris)
- **Changement (BOOL)** : TRUE si couleur différente de la précédente

### Variables internes :

- **Couleur\_Precedente (INT)** : Couleur de la pièce précédente

### États du diagramme ECC :

- **START → INIT** : Initialisation (ACT\_INIT)
- **IDLEE** : Attente d'une pièce
- **READ** : Lecture de la valeur capteur (ACT\_READ)
- **DETECTED** : Pièce détectée, comparaison couleur (ACT\_DETECT)

### Algorithmes :

- **ACT\_INIT** : Initialise Couleur=0, Changement=FALSE
- **ACT\_READ** : Convertit Vision\_value en couleur (1-3→Bleu, 4-6→Vert, 7-9→Gris)
- **ACT\_DETECT** : Compare avec couleur précédente, met à jour Changement

### 5.3 FB\_TriLogic - Logique de Tri

**Description :** Gère la logique centrale de tri des pièces par couleur. Contrôle l'activation des trieurs, le blocage de l'entrée lors des changements de couleur, et maintient les compteurs de pièces triées.

#### Événements d'entrée :

- **INIT** : Initialisation du bloc
- **NEW\_COLOR** : Nouvelle couleur détectée par le capteur vision
- **REQ** : Requête d'exécution cyclique
- **PIECE\_SORTIE** : Une pièce a quitté le système
- **RESET** : Réinitialisation des compteurs
- **AU** : Arrêt d'urgence

#### Événements de sortie :

- **INITO** : Confirmation d'initialisation
- **CNF** : Confirmation d'exécution
- **BLOCK** : Commande de blocage de l'entrée (changement de couleur)
- **UNBLOCK** : Commande de déblocage de l'entrée

#### Variables d'entrée :

- **QI (BOOL)** : Qualificateur d'événement
- **Couleur (INT)** : Code couleur (1=Bleu, 2=Vert, 3=Gris)
- **Changement (BOOL)** : TRUE si couleur différente
- **At\_exit (BOOL)** : Capteur de sortie

#### Variables de sortie :

- **QO (BOOL)** : Qualificateur de sortie
- **Sorter\_Actif (INT)** : Trieur actif (1, 2 ou 3)
- **Exit\_conveyor (BOOL)** : Convoyeur de sortie actif
- **Counter\_1 (INT)** : Compteur pièces Bleues
- **Counter\_2 (INT)** : Compteur pièces Vertes
- **Counter\_3 (INT)** : Compteur pièces Grises

#### Variables internes :

- **Couleur\_EnCours (INT)** : Couleur de la pièce en cours de tri
- **Couleur\_Suivante (INT)** : Couleur de la prochaine pièce (si différente)
- **Is\_Blocked (BOOL)** : Indicateur d'entrée bloquée

#### États du diagramme ECC :

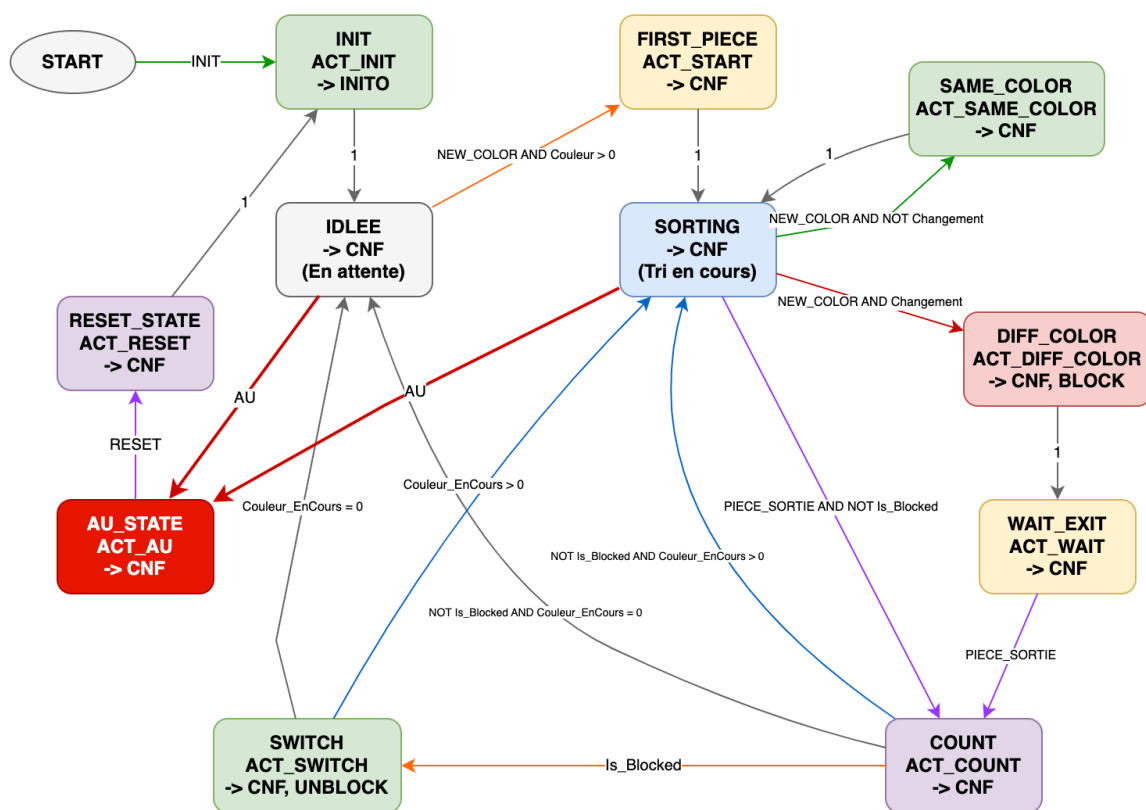
- **START → INIT** : Initialisation (ACT\_INIT)
- **IDLEE** : Attente, aucune pièce
- **FIRST\_PIECE** : Première pièce, activation trieur (ACT\_START)
- **SORTING** : Tri en cours
- **SAME\_COLOR** : Même couleur, pas de blocage (ACT\_SAME\_COLOR)
- **DIFF\_COLOR** : Couleur différente → émet BLOCK (ACT\_DIFF\_COLOR)
- **WAIT\_EXIT** : Attente sortie pièce (ACT\_WAIT)
- **COUNT** : Comptage pièce sortie (ACT\_COUNT)
- **SWITCH** : Changement trieur → émet UNBLOCK (ACT\_SWITCH)
- **AU\_STATE** : Arrêt d'urgence (ACT\_AU)
- **RESET\_STATE** : Réinitialisation (ACT\_RESET)



### Algorithmes :

- **ACT\_INIT** : Initialise compteurs=0, Sorter\_Actif=0
- **ACT\_START** : Active le trieur pour la première pièce
- **ACT\_SAME\_COLOR** : Continue sans bloquer
- **ACT\_DIFF\_COLOR** : Bloque l'entrée, mémorise nouvelle couleur
- **ACT\_WAIT** : Attente sortie pièce
- **ACT\_COUNT** : Incrémente le compteur approprié
- **ACT\_SWITCH** : Change de trieur, débloquent l'entrée
- **ACT\_AU** : Arrêt d'urgence
- **ACT\_RESET** : Réinitialise compteurs et états

### DIAGRAMME ECC - FB\_TriLogic



[SCHÉMA : Diagramme ECC FB\_TriLogic - fichier SCHEMAS\_IEC61499\_PRI\_PROJET-IEC61499\_V0\_1.drawio, onglet '4-ECC FB\_TriLogic']

## 5.4 FB\_SorterLogic - Logique Trieur

**Description :** Gère un trieur individuel (rotation et tapis). S'active lorsque son identifiant correspond au trieur actif commandé par FB\_TriLogic.

**Événements d'entrée :**

- **INIT** : Initialisation du bloc (avec My\_ID)
- **REQ** : Requête d'exécution cyclique
- **AU** : Arrêt d'urgence

**Événements de sortie :**

- **INITO** : Confirmation d'initialisation
- **CNF** : Confirmation d'exécution

**Variables d'entrée :**

- **QI (BOOL)** : Qualificateur d'événement
- **Sorter\_Actif (INT)** : Identifiant du trieur actif (1, 2 ou 3)
- **My\_ID (INT)** : Identifiant propre du trieur (1, 2 ou 3)

**Variables de sortie :**

- **QO (BOOL)** : Qualificateur de sortie
- **Turn (BOOL)** : Commande rotation trieur
- **Belt (BOOL)** : Commande tapis convoyeur

**États du diagramme ECC :**

- **START → INIT** : Initialisation (ACT\_INIT)
- **IDLEE** : Attente/repos
- **UPDATE** : Mise à jour état trieur (ACT\_UPDATE)
- **AU\_STATE** : Arrêt d'urgence (ACT\_AU)

**Algorithmes :**

- **ACT\_INIT** : Initialise Turn=FALSE, Belt=FALSE
- **ACT\_UPDATE** : Active si Sorter\_Actif = My\_ID (Turn=TRUE, Belt=TRUE)
- **ACT\_AU** : Arrêt d'urgence (Turn=FALSE, Belt=FALSE)

## 5.5 Interface Adaptateur ICouleur

**Description :** Interface adaptateur pour la communication des données couleur entre le capteur de vision (EQUIP\_VisionSensor) et le gestionnaire de tri (EQUIP\_TRIManager).

**Événement :**

- **NEW\_COLOR** : Nouvelle couleur détectée

**Variables :**

- **Couleur (INT)** : Code couleur (0=Aucune, 1=Bleu, 2=Vert, 3=Gris)
- **Changement (BOOL)** : TRUE si couleur différente de la précédente

## 6. Conception des Composites (EQUIP)

### 6.1 Principe des blocs composites

Les blocs composites (EQUIP) encapsulent un FB basique avec les blocs de communication Modbus (SYMLINK). Ils exposent une interface simplifiée tout en masquant la complexité des E/S.

### 6.2 EQUIP\_EntryConveyor - Équipement Convoyeur d'Entrée

**Description :** Bloc composite encapsulant FB\_EntryLogic avec les blocs de lecture/écriture I/O pour le convoyeur d'entrée.

#### Événements d'entrée :

- **INIT** : Initialisation
- **REQ** : Requête cyclique
- **BLOCK** : Blocage convoyeur
- **UNBLOCK** : Déblocage convoyeur

#### Événements de sortie :

- **INITO** : Confirmation d'initialisation
- **CNF** : Confirmation d'exécution
- **PIECE\_PRESENT** : Pièce détectée
- **AU\_OUT** : Signal arrêt d'urgence
- **RESET\_OUT** : Signal reset

#### Réseau interne :

- **Read (SYMLINKMULTIVARDST)** : Lecture des variables Modbus d'entrée (Auto, Start, Stop, AU, Reset, Presence\_sensor)
- **Logic (FB\_EntryLogic)** : Logique métier du convoyeur d'entrée
- **Write (SYMLINKMULTIVARSRC)** : Écriture des variables Modbus de sortie (Entry\_conveyor, Stop\_blade, Start\_light, Reset\_light, Stop\_light)

### 6.3 EQUIP\_VisionSensor - Équipement Capteur Vision

**Description :** Bloc composite encapsulant FB\_VisionSensor avec interface adaptateur ICouleur pour la sortie couleur.

#### Événements d'entrée :

- **INIT** : Initialisation
- **REQ** : Requête cyclique
- **PIECE\_PRESENT** : Pièce présente

#### Événements de sortie :

- **INITO** : Confirmation d'initialisation
- **CNF** : Confirmation d'exécution

#### Adaptateur :

- **CouleurOut (Socket ICouleur)** : Interface sortie couleur vers TRIManager

#### Réseau interne :

- **ReadVision (SYMLINKMULTIVARDST)** : Lecture du registre Modbus Vision\_Sensor
- **Logic (FB\_VisionSensor)** : Conversion vision → couleur
- **Connexion adaptateur CouleurOut** : Transmission de Couleur et Changement via l'adapter

### 6.4 EQUIP\_TRIManager - Équipement Gestionnaire de Tri

**Description** : Bloc composite principal orchestrant la logique de tri avec entrée couleur via adaptateur.

#### Événements d'entrée :

- **INIT** : Initialisation
- **REQ** : Requête cyclique
- **PIECE\_SORTIE** : Pièce sortie du système
- **RESET** : Réinitialisation compteurs
- **AU** : Arrêt d'urgence

#### Événements de sortie :

- **INITO** : Confirmation d'initialisation
- **CNF** : Confirmation d'exécution
- **BLOCK** : Blocage entrée
- **UNBLOCK** : Déblocage entrée

#### Variables de sortie :

- **Sorter\_Actif (INT)** : Trieur actif (1, 2 ou 3)
- **Exit\_conveyor (BOOL)** : Convoyeur sortie actif
- **Counter\_1, Counter\_2, Counter\_3 (INT)** : Compteurs par couleur

#### Adaptateur :

- **CouleurIn (Plug ICouleur)** : Interface entrée couleur depuis VisionSensor

#### Réseau interne :

- **FB1 (FB\_TriLogic)** : Logique de tri complète
- **Réception couleur via CouleurIn** : Connexion adapter Plug pour recevoir NEW\_COLOR, Couleur et Changement

### 6.5 EQUIP\_Sorter - Équipement Trieur

**Description** : Bloc composite pour le contrôle d'un trieur avec chemins de variables configurables. Instancié 3 fois avec des paramètres différents.

#### Événements d'entrée :

- **INIT** : Initialisation (avec TurnPath, BeltPath, My\_ID)
- **REQ** : Requête cyclique
- **AU** : Arrêt d'urgence

#### Événements de sortie :

- **INITO** : Confirmation d'initialisation
- **CNF** : Confirmation d'exécution

#### Variables d'entrée :

- **TurnPath (STRING)** : Chemin variable Modbus pour rotation

- **BeltPath (STRING)** : Chemin variable Modbus pour tapis
- **My\_ID (INT)** : Identifiant trieur (1, 2 ou 3)
- **Sorter\_Actif (INT)** : Trieur actif (venant de TRIManager)

#### Réseau interne :

- **logic (FB\_SorterLogic)** : Logique de contrôle du trieur
- **Write (SYMLINKMULTIVARSRC)** : Écriture Turn et Belt vers Modbus via chemins variables

#### Paramètres d'instanciation :

- **Sorter1** : My\_ID=1, TurnPath='Sorter\_1\_turn', BeltPath='Sorter\_1\_belt'
- **Sorter2** : My\_ID=2, TurnPath='Sorter\_2\_turn', BeltPath='Sorter\_2\_belt'
- **Sorter3** : My\_ID=3, TurnPath='Sorter\_3\_turn', BeltPath='Sorter\_3\_belt'

## 6.6 EQUIP\_ExitConveyor - Équipement Convoyeur de Sortie

**Description** : Bloc composite pour le contrôle du convoyeur de sortie avec détection de front descendant pour signaler la sortie des pièces.

#### Événements d'entrée :

- **INIT** : Initialisation
- **REQ** : Requête cyclique

#### Événements de sortie :

- **INITO** : Confirmation d'initialisation
- **CNF** : Confirmation d'exécution
- **PIECE\_SORTIE** : Pièce sortie (front descendant At\_exit)

#### Variables d'entrée :

- **Exit\_conveyor (BOOL)** : Commande convoyeur sortie (depuis TRIManager)

#### Variables de sortie :

- **At\_exit\_Value (BOOL)** : Valeur capteur sortie

#### Réseau interne :

- **ReadAtExit (SYMLINKMULTIVARDST)** : Lecture du coil At\_exit depuis Modbus
- **WriteAtExit (SYMLINKMULTIVARSRC)** : Écriture de Exit\_conveyor vers Modbus
- **FallingTrig (E\_F\_TRIG)** : Détection front descendant → émet PIECE\_SORTIE

## 7. Configuration Système et Communication

### 7.1 Mapping E/S Modbus

La communication avec Factory I/O utilise les blocs SYMLINK pour le mapping des variables :

#### ENTRÉES (Lecture depuis Factory I/O) :

- MODBUSRCOIL8 : 8 coils à l'adresse 0 (capteurs TOR)
- MODBUSRREG1 : 1 registre à l'adresse 0 (capteur vision)

#### SORTIES (Écriture vers Factory I/O) :

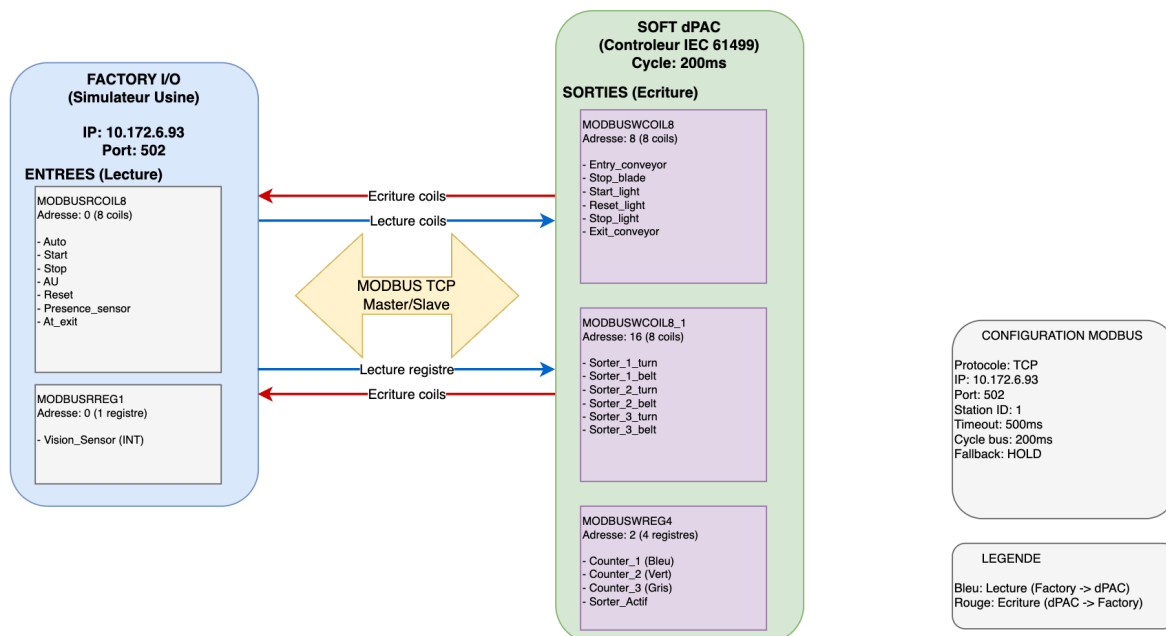
- MODBUSWCOIL8 : 8 coils à l'adresse 8 (actionneurs convoyeurs)
- MODBUSWCOIL8\_1 : 8 coils à l'adresse 16 (actionneurs sorters)
- MODBUSWREG4 : 4 registres à l'adresse 2 (compteurs HMI)

### 7.2 Blocs SYMLINK

Les blocs SYMLINKMULTIVARDST (lecture) et SYMLINKMULTIVARSRC (écriture) permettent de mapper les variables du programme vers les adresses Modbus. Ils sont configurés avec :

- Le chemin de la variable (ex: 'Auto', 'Entry\_conveyor')
- L'événement de déclenchement (REQ/CNF)

### COMMUNICATION MODBUS TCP



[SCHÉMA : Communication Modbus TCP - fichier SCHEMAS\_IEC61499\_PRI\_PROJET-IEC61499\_V0\_1.drawio, onglet '5-Communication Modbus']

## 8. Chaînes d'Événements

### 8.1 Chaîne d'initialisation (INIT)

Au démarrage (COLD ou WARM), les blocs s'initialisent en cascade :

START.COLD/WARM → FB1.INIT → FB1.INITO → EntryConv.INIT → EntryConv.INITO → Vision.INIT → Vision.INITO → TriManager.INIT → TriManager.INITO → ExitConv.INIT → ExitConv.INITO → Sorter1.INIT → Sorter1.INITO → Sorter2.INIT → Sorter2.INITO → Sorter3.INIT → Sorter3.INITO

### 8.2 Chaîne cyclique (REQ/CNF)

L'exécution cyclique propage les événements REQ/CNF :

EntryConv.CNF → Vision.REQ → Vision.CNF → TriManager.REQ → TriManager.CNF → ExitConv.REQ → ExitConv.CNF → Sorter1.REQ → Sorter1.CNF → Sorter2.REQ → Sorter2.CNF → Sorter3.REQ → Sorter3.CNF

#### FLUX D'ÉVÉNEMENTS - CHAÎNE D'EXECUTION

##### 1. CHAÎNE D'INITIALISATION (INIT)



##### 2. CHAÎNE CYCLIQUE (REQ/CNF)



##### 3. DETECTION PIECE



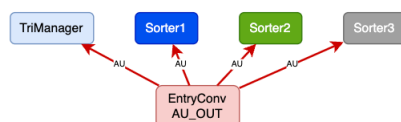
##### 4. SORTIE PIECE



##### 5. BLOCAGE/DEBLOCAGE



##### 6. ARRÊT D'URGENCE



[SCHEMA : Flux d'Événements - fichier SCHEMAS\_IEC61499\_PRI\_PROJET-IEC61499\_V0\_1.drawio, onglet '2-Flux Evenements']

## 8.3 Événements process

### Détection pièce :

EntryConv.PIECE\_PRESENTES → Vision.PIECE\_PRESENTES → Vision.NEW\_COLOR (via adapter) → TriManager.NEW\_COLOR

### Sortie pièce :

ExitConv.PIECE\_SORTIE (front descendant At\_exit) → TriManager.PIECE\_SORTIE

### Blocage/Débloccage :

TriManager.BLOCK → EntryConv.BLOCK / TriManager.UNBLOCK → EntryConv.UNBLOCK

### Arrêt d'urgence :

EntryConv.AU\_OUT → TriManager.AU, Sorter1.AU, Sorter2.AU, Sorter3.AU

## 9. Guide de Reproduction du Programme

### 9.1 Étapes de création dans EAE

#### Étape 1 - Créer le projet

- File → New Project
- Nommer le projet
- Sélectionner le type de contrôleur : SE.DPAC

#### Étape 2 - Créer les FB basiques

- Clic droit sur 'Function Blocks' → Add → Basic Function Block
- Créer FB\_EntryLogic avec les événements/variables décrits section 5.1
- Dessiner le diagramme ECC selon le schéma fourni
- Créer les algorithmes en ST pour chaque état
- Répéter pour FB\_VisionSensor, FB\_TriLogic, FB\_SorterLogic

#### Étape 3 - Créer l'adapter ICouleur

- Clic droit sur 'Adapters' → Add → Adapter Type
- Définir l'événement NEW\_COLOR
- Ajouter les variables Couleur (INT) et Changement (BOOL)

#### Étape 4 - Créer les composites EQUIP

- Clic droit sur 'Function Blocks' → Add → Composite Function Block
- Pour chaque EQUIP, ajouter les FB internes et les blocs SYMLINK
- Connecter les événements et données internes
- Définir les événements et variables exposés

#### Étape 5 - Configurer la ressource RES0

- Ouvrir la ressource EMB\_RES\_ECO (RES0)
- Ajouter FB1 (DPAC\_FULLINIT)
- Instancier tous les EQUIP
- Connecter les chaînes d'événements INIT/CNF
- Connecter les événements process et les données

#### Étape 6 - Configurer la communication Modbus

- Configurer le device SE.DPAC avec l'adresse IP
- Configurer le client Modbus TCP (IP: 10.172.6.93, Port: 502)
- Définir les zones de lecture/écriture Modbus



## 9.2 Test et validation

- Compiler le projet (Build)
- Déployer sur le Soft dPAC
- Lancer Factory I/O avec la scène de tri
- Vérifier la connexion Modbus
- Tester le cycle complet (Start, tri de pièces, Stop, AU)

Se référer aux fichiers (.xlsx) de tests unitaires et d'intégrations :

[FICHE\\_TESTS\\_INTEGRATIONS\\_EAE\\_IEC61499\\_PROJET-IEC61499\\_V2.xlsx](#)

[FICHE\\_TESTS\\_UNITAIRES\\_EAE\\_IEC61499\\_PROJET-IEC61499\\_V2.xlsx](#)

## Annexes : Emplacements des Schémas

Les schémas référencés dans ce document sont disponibles dans les fichiers draw.io suivants :

**Fichier : SCHEMAS\_IEC61499\_PRI\_PROJET-IEC61499\_V0\_1.drawio**

Onglet	Contenu	Section du document
1-Architecture Generale	Vue globale des modules et connexions	Section 3.3, 4.1
2-Flux Evenements	Chaînes INIT, REQ/CNF, process	Section 8
3-ECC FB_EntryLogic	Diagramme ECC complet	Section 5.1
4-ECC FB_TriLogic	Diagramme ECC complet	Section 5.3
4-Communication Modbus	Flux Modbus	Section 7.2

**Fichier : G7\_NIVEAU2\_PRI\_PROJET-IEC61499\_V0\_2.drawio**

Onglet	Contenu	Section du document
G7 Principal	GRAFCET principal avec gestion AU/Stop/Reset	Section 3.1
G7 Tri	Sous-GRAFCET de tri (3 parallèles)	Section 3.1

**Fichiers de tests :**

- FICHE\_TESTS\_UNITAIRES\_EAE\_IEC61499\_PROJET-IEC61499\_V2.xlsx
- FICHE\_TESTS\_INTEGRATIONS\_EAE\_IEC61499\_PROJET-IEC61499\_V2.xlsx

**Note :** Les fichiers .drawio peuvent être ouverts avec draw.io (diagrams.net). Pour les inclure dans ce document, les exporter en PNG, PDF ou SVG.