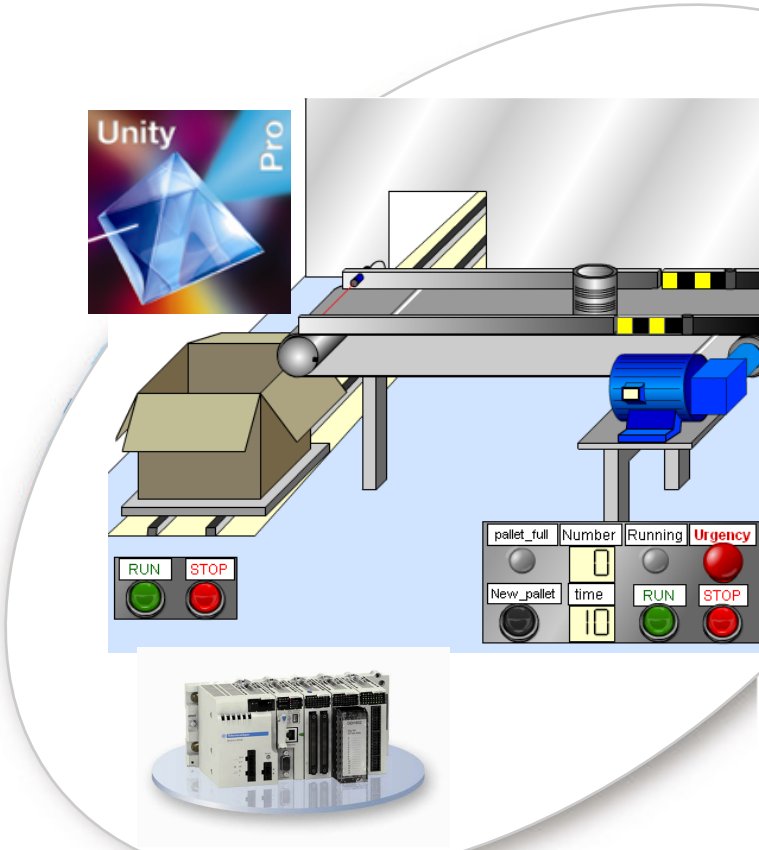
Didacticiel

Unity Pro & Modicon M340

Votre premier projet développé avec Unity Pro

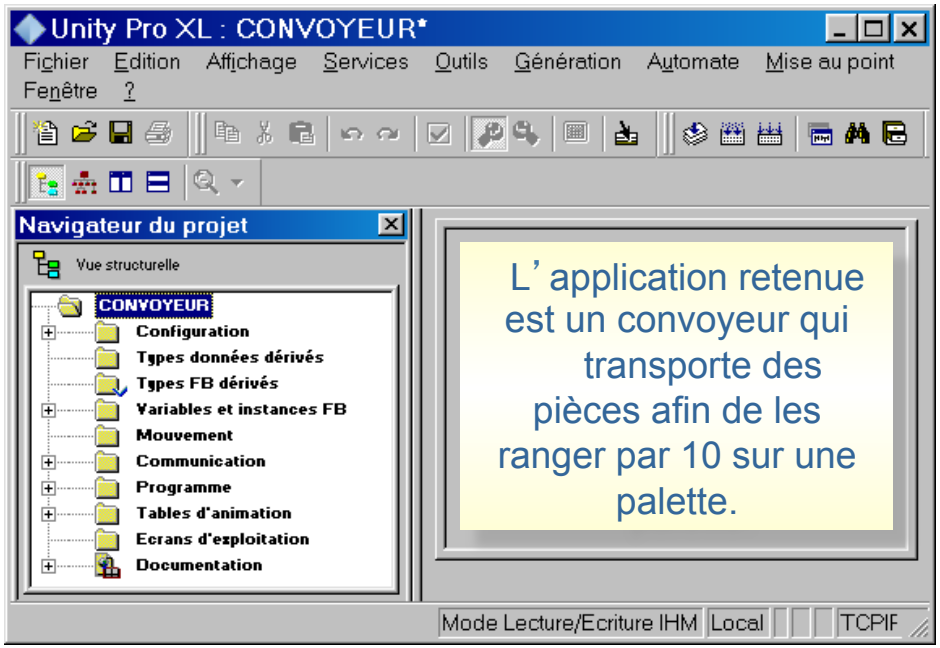


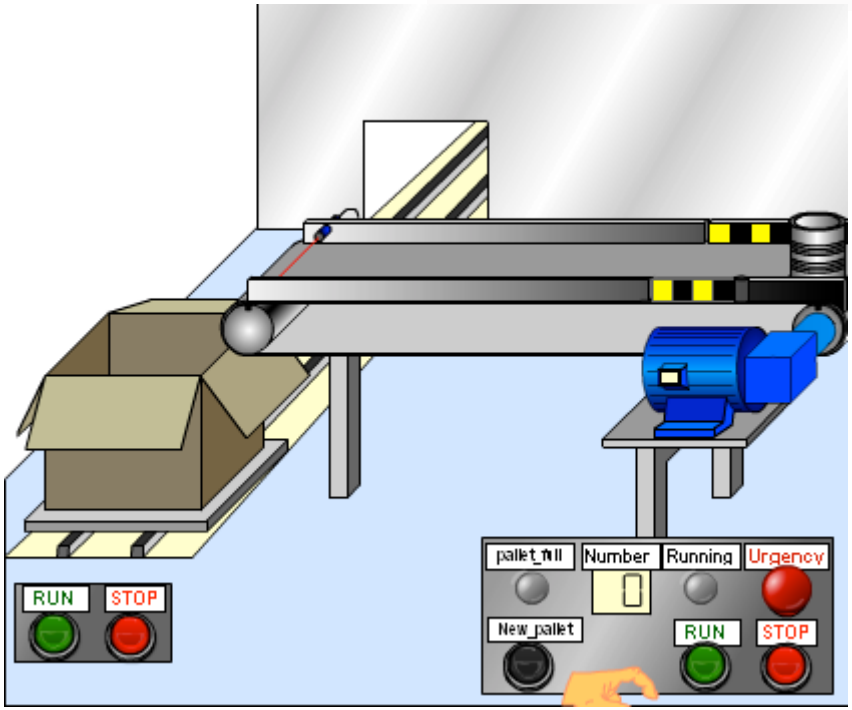
a brand of Schneider Electric

Telemecanique

Votre premier projet Unity Pro

Ce didacticiel a pour objectif de montrer la simplicité d'usage de Unity Pro par la mise en œuvre d'une application pilotée à partir de l'automate programmable Modicon M340.





Votre premier projet Unity Pro

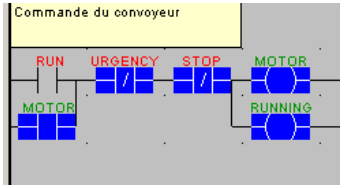
Objectifs

Programmer en LD | Programmer en ST | Modifier en ligne | Pour aller plus loin

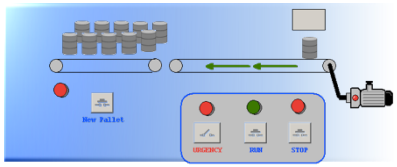
Partant du cahier des charges d'une application, nous allons vous montrer les principes de développement d'un projet avec Unity Pro

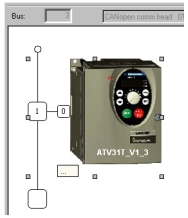
Le didacticiel comporte 4 parties :

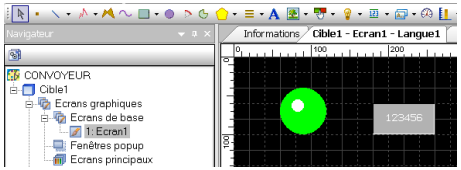
1. **Programmer en langage LD** (langage à contact) : cette partie présente la méthodologie de création d'un projet, les procédures de programmation en LD, ainsi que la mise au point du projet.
2. **Programmer en ST** (langage littéral structuré) : cette partie présente les procédures de programmation et de mise au point en ST ainsi que l'utilisation d'écrans d'exploitation pour visualiser et piloter le procédé.
3. **Modifier en ligne :** présente les procédures de modification en ligne du programme.
4. **Pour aller plus loin** : Cette dernière partie du didacticiel présente la simplicité d'intégration d'un automate Modicon M340 programmé avec Unity Pro dans une architecture d'automatisme : pilotage d'un variateur de vitesse ATV31, dialogue opérateur avec un terminal Magelis XBT GT.




```
(* Comptage pièces*)
If re (optical_sensor) then
        inc (number);
end_if;
if re (new_pallet) then
        number:=0;
end_if;
if number=10 then
        pallet_full:=true;
else
        pallet_full:=false;
end_if;
```







- Le projet sera mis au point sur le simulateur automate puis configuré pour s'exécuter sur l'automate réel.
- Des démonstrations enregistrées de modes opératoires vous seront également proposées.

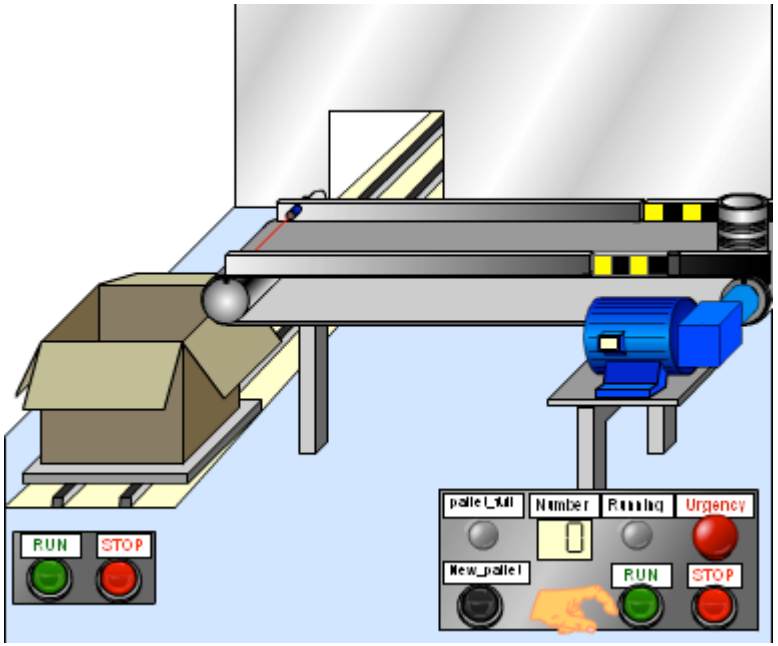
Votre premier projet Unity Pro

Approche méthodologique

L'installation à piloter est un convoyeur servant à transporter des pièces qui sont rangées par 10 puis évacuées sur une palette. Pour chaque fonction, un cahier des charges sera proposé ainsi qu'une méthodologie de développement.

L'application retenue comporte 3 fonctions :

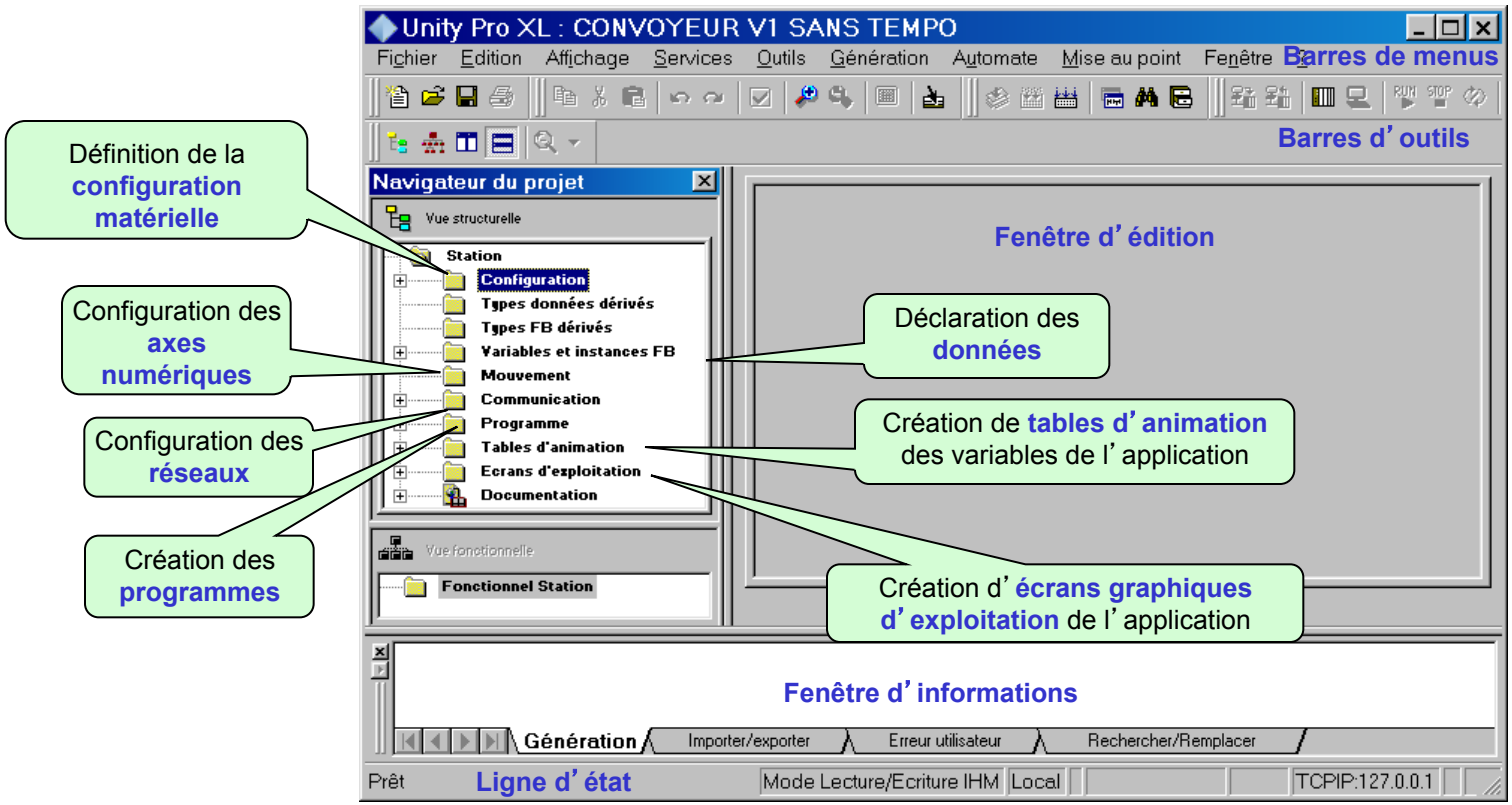
- La fonction **Convoyeur** qui permet le pilotage du convoyeur.
- La fonction **Palettiseur** qui permet le comptage des pièces et de traiter une nouvelle palette.
- La fonction **Gestion** qui permet d'arrêter le convoyeur au bout d'un certain temps, s'il n'y a plus de pièce sur le convoyeur.



Votre premier projet Unity Pro

Introduction à Unity Pro

Unity Pro permet de programmer les automates Modicon M340, Premium, Atrium et Quantum. Le navigateur de projet visualise l'organisation de l'application et donne l'accès aux éditeurs. Les fenêtres sont repositionnables sur l'écran et peuvent être affichées selon plusieurs modes (pleine page, réduit, flottant)



Programmation du convoyeur en langage LD

Cahier des charges du convoyeur

L'installation comprend un convoyeur et un pupitre de commande

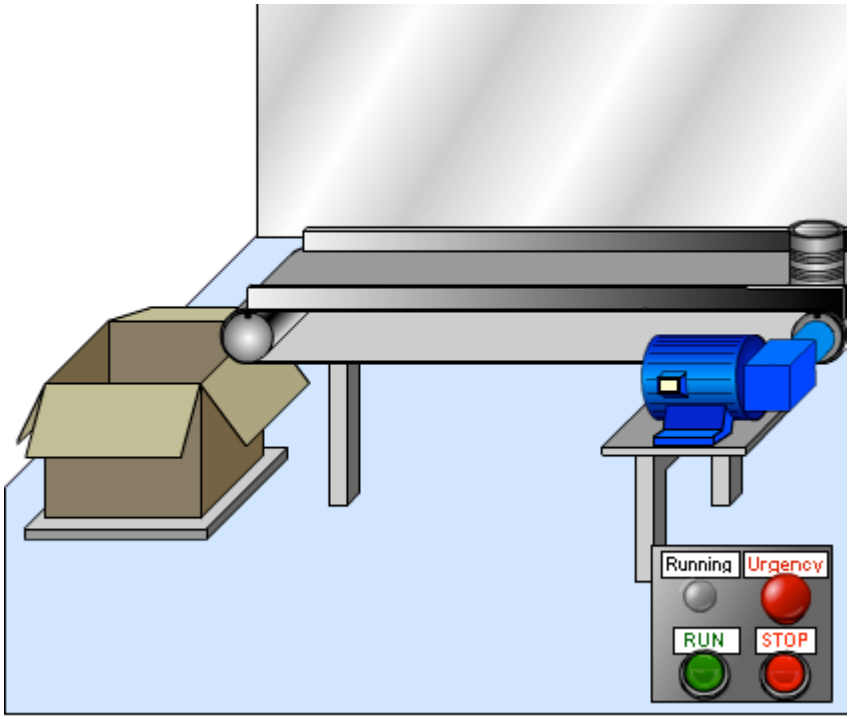
Les besoins en entrées :

- Une entrée **RUN** de mise en marche du convoyeur
- Une entrée **STOP** d'arrêt du convoyeur
- Une entrée **URGENCY** d'arrêt d'urgence

Les besoins en sorties :

- Une sortie commande moteur **MOTOR**
- Une sortie voyant **RUNNING**

Remarque :
Dans cette première phase basée sur l'utilisation du simulateur intégré à Unity Pro, nous définissons les informations d'entrées/sorties nécessaires sans affecter pour l'instant les adresses physiques.
L'affectation des entrées/sorties sera réalisée en phase 3.



Utilisez les boutons de la maquette pour comprendre le cahier des charges

Votre premier projet Unity Pro

Méthodologie de développement

La chronologie de développement que nous vous proposons est la suivante :

Prise en charge de l'application

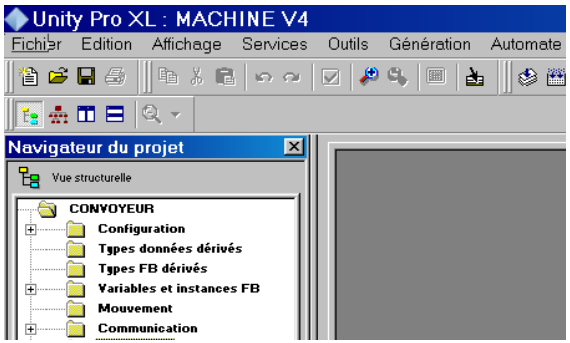
- Cahier des charges
- Analyse
- Lancement de Unity Pro

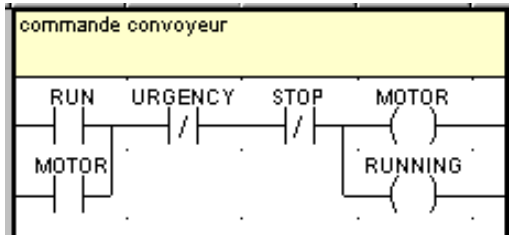
Réalisation du projet

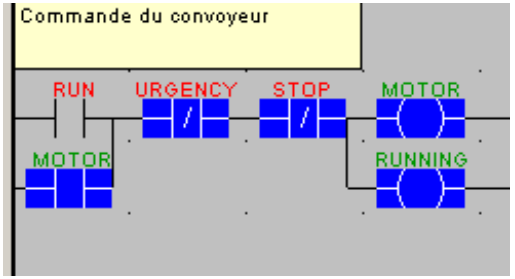
- Création du projet
- Déclaration des données
- Création du programme
- Analyse du programme et génération du code

Mise au point du projet

- Transfert du projet dans le simulateur
- Passage en Run Visualisation dynamique du programme et réglage de variables







Programmation du convoyeur en langage LD

Analyse du cahier des charges

Cette phase consiste à déterminer la logique de commande du convoyeur

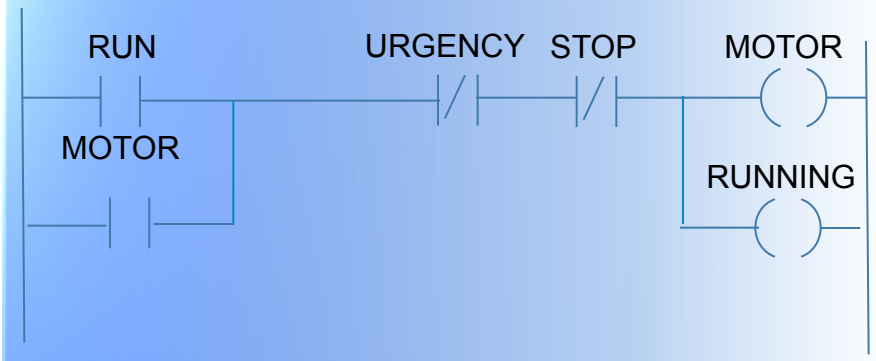
Structure du programme :

Le convoyeur démarre lorsque l'utilisateur appuie sur le bouton **RUN** et s'il n'y a pas **d'arrêt d'urgence**.

Le convoyeur s'arrête lorsque l'utilisateur appuie sur le bouton **STOP** ou sur **l'arrêt d'urgence**.

Le programme sera réalisé langage Ladder (LD).

Commande du Convoyeur



Programmation du convoyeur en langage LD

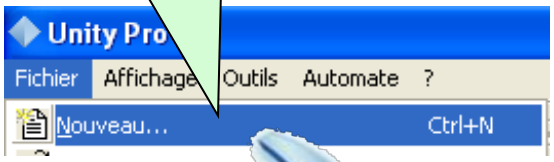
Création du projet (1/3)

A l'aide du logiciel Unity Pro nous allons maintenant :

- Créer un nouveau projet intitulé « Machine »
- Définir la base de l'automate

Sélectionner le menu **Fichier/Nouveau.**

1



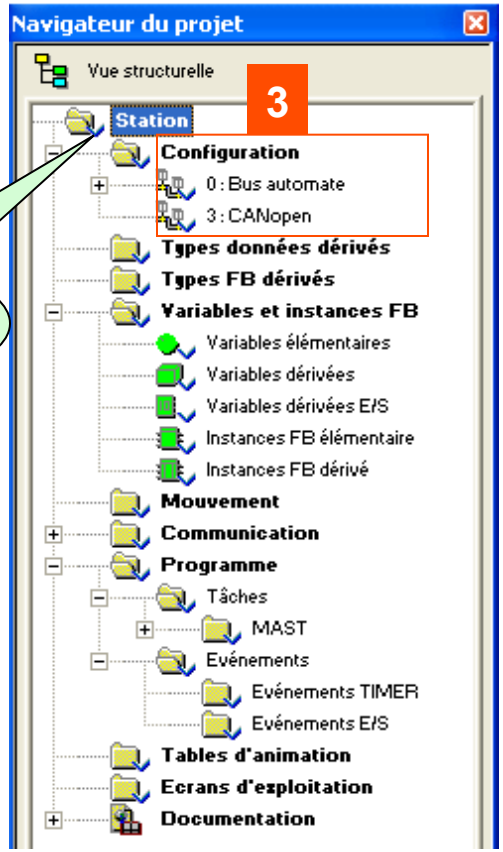
Automate	Ver... min.	Description
Modicon M340		
BMX P34 1000	01.00	CPU 340-10 Modbus
BMX P34 2010	01.00	CPU 340-20 Modbus CANopen
BMX P34 2020	01.00	CPU 340-20 Modbus Ethernet
BMX P34 2030	01.00	CPU 340-20 Ethernet CANopen

2

Sélectionner **la base automate : BMX P34 2030** et valider par OK.
Le navigateur présente la structure d'une application.

La **structure du projet** est créée.

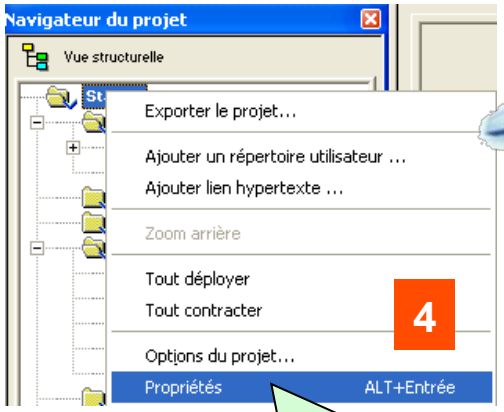
3

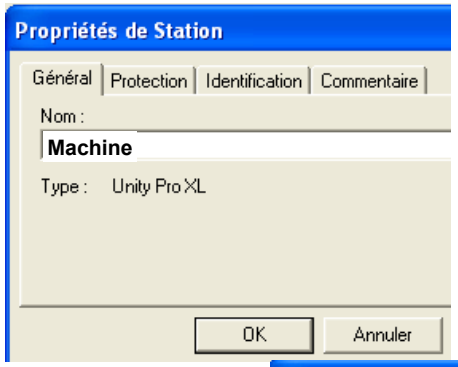


Programmation du convoyeur en langage LD

Création du projet (2/3)

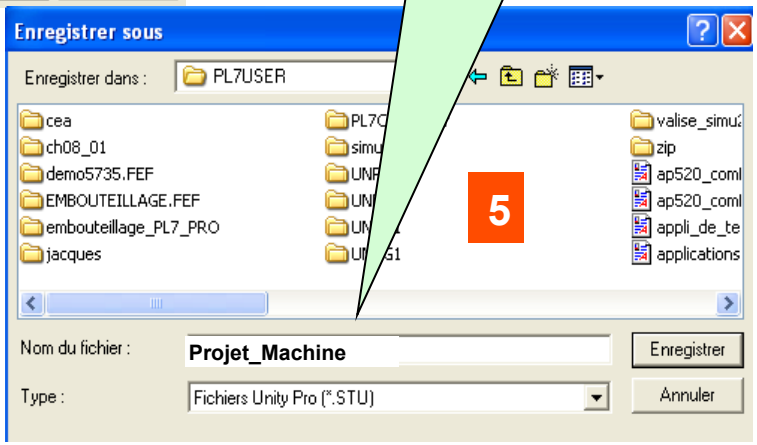
Nommage et enregistrement du projet :





Effectuer un clic droit sur Station et sélectionner le menu **Propriétés**, afin d'indiquer le **nom** et saisir le **commentaire** du projet.

Sauvegarder le projet à l'aide du menu **Fichier/ Enregistrer** (**Fichier *.STU**).



Programmation du convoyeur en langage LD

Création du projet (3/3)

Des démonstrations de différentes procédures vous sont proposées dans ce didacticiel.
Lancez ces démonstrations, puis effectuez vous-même la procédure présentée avec Unity Pro.

■ Démonstration

Cliquer sur l'icône ci-contre pour lancer la démonstration sur la création de l'application

Signification des commandes utilisables pendant la démonstration

Lecture Pause Rembobiner

A la fin de la démonstration, END est affiché puis , le film se repositionne en début.
Cliquer sur le bouton Pause si vous souhaitez rester sur la dernière image.
Fermer la démonstration avant de passer à la page suivante.

■ A vous de jouer…

Lancez le logiciel Unity Pro et réalisez les opérations de création du projet.

Programmation du convoyeur en langage LD

Déclaration des données (1/2)

La déclaration des données peut se faire :
- soit à partir de l'éditeur de données,
- soit au fil de l'eau, lors de la saisie du programme.

Nous allons déclarer les données d'entrées relatives au programme Convoyeur dans l'**éditeur de données**.
Les données de sortie seront déclarées au fil de l'eau lors de la création du programme.

Entrées

Nom	Type	Commentaire
RUN	EBOOL	Départ convoyeur
STOP	EBOOL	Arrêt convoyeur
URGENCY	EBOOL	Arrêt d'urgence

Sorties

Nom	Type	Commentaire
MOTOR	EBOOL	Commande Moteur convoyeur
RUNNING	EBOOL	Voyant Marche/Arrêt du moteur

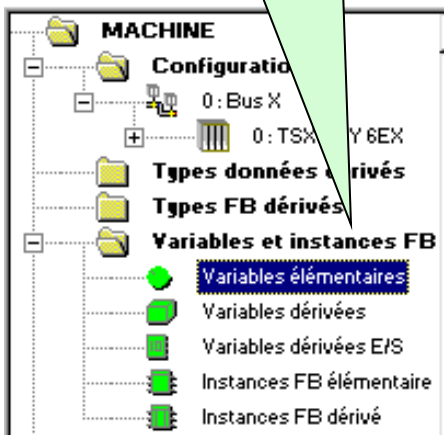
Remarque : Nous typerons les variables d'E/S en EBOOL de façon à pouvoir les associer ensuite aux voies des modules de l'automate.

Programmation du convoyeur en langage LD

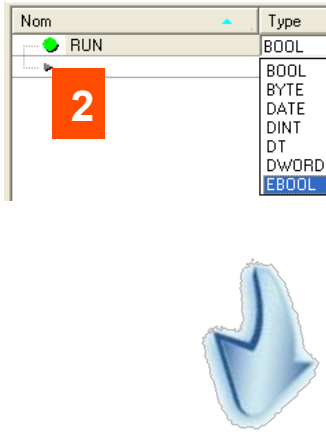
Déclaration des données (2/2)

Sur l'atelier Unity Pro, vous pouvez utiliser des variables en déclarant uniquement le nom et le type mais sans déclarer d'adresse : ces variables sont non localisées, c'est le système qui attribue de manière interne ces adresses.

Effectuer un double clic sur **Variables élémentaires** pour accéder à l'éditeur de données.



1



2

Indiquer :
Le **nom de la variable**
Le **type de la variable** : EBOOL
Le **commentaire** de la variable.

3

Déclarer toutes les variables suivantes

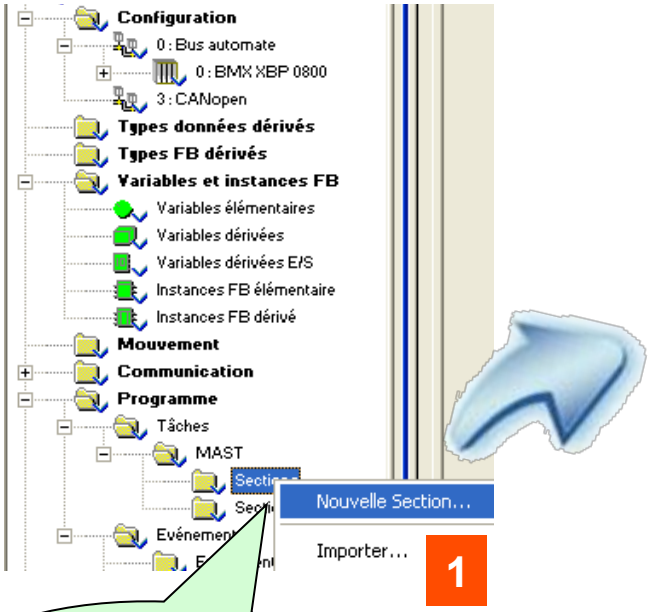
Nom	Type	Adresse	Valeur	Commentaire
RUN	EBOOL			Départ convoyeur
STOP	EBOOL			Arrêt convoyeur
URGENCY	EBOOL			Arrêt d'urgence

Remarque : les autres données seront déclarées au fil de l'eau lors de la création du programme LD

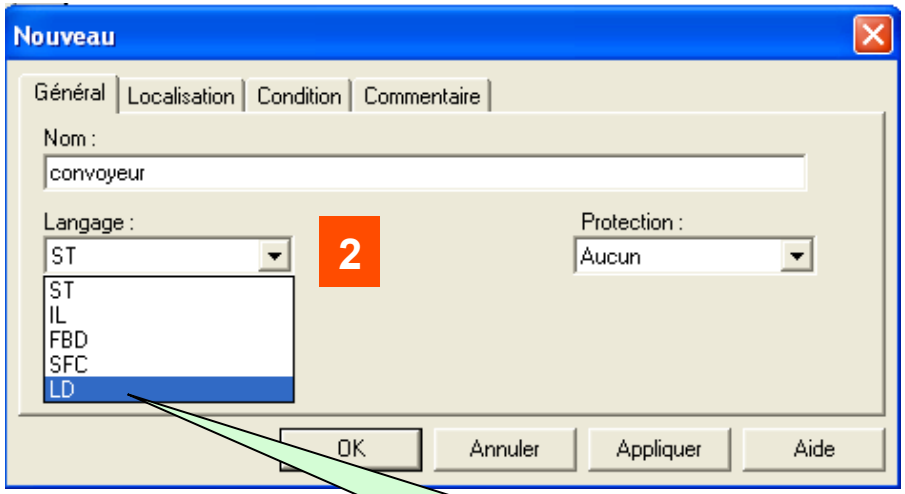
Programmation du convoyeur en langage LD

Création de la section convoyeur (1/3)

Une projet Unity Pro peut comporter plusieurs tâches (tâche maître créée par défaut et qui représente la tâche principale, des tâches événementielles, …). Les tâches sont composées de sections et sous-programmes. L'ordre des sections détermine l'ordre de scrutation du programme.



Effectuer un clic droit sur **Section** et sélectionner le menu **Nouvelle section.**

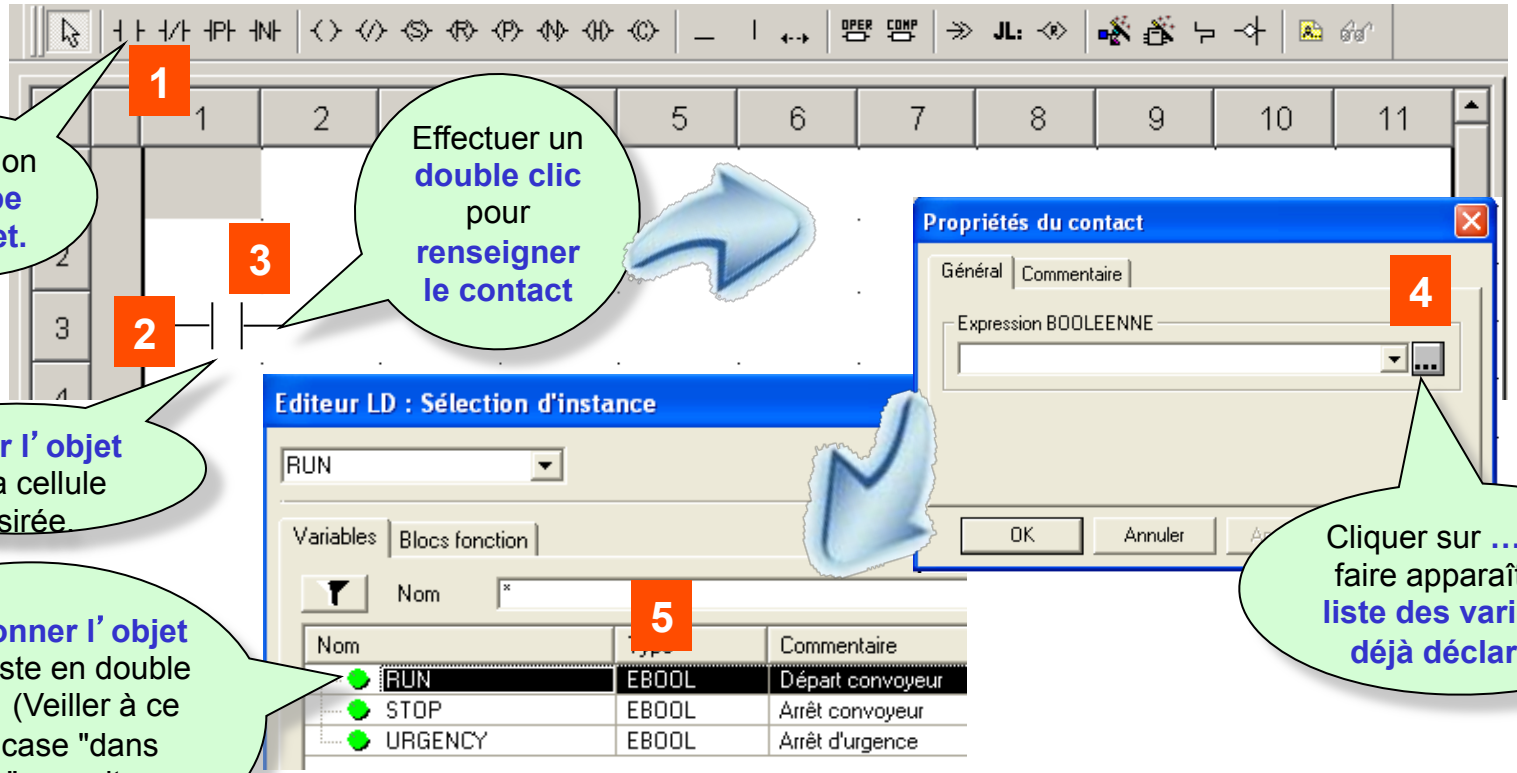


Saisir le **Nom de la section** et sélectionner le **langage LD** puis valider par OK.

Programmation du convoyeur en langage LD

Création de la section convoyeur (2/3)

Construction du réseau de contacts en utilisant les variables déjà déclarées

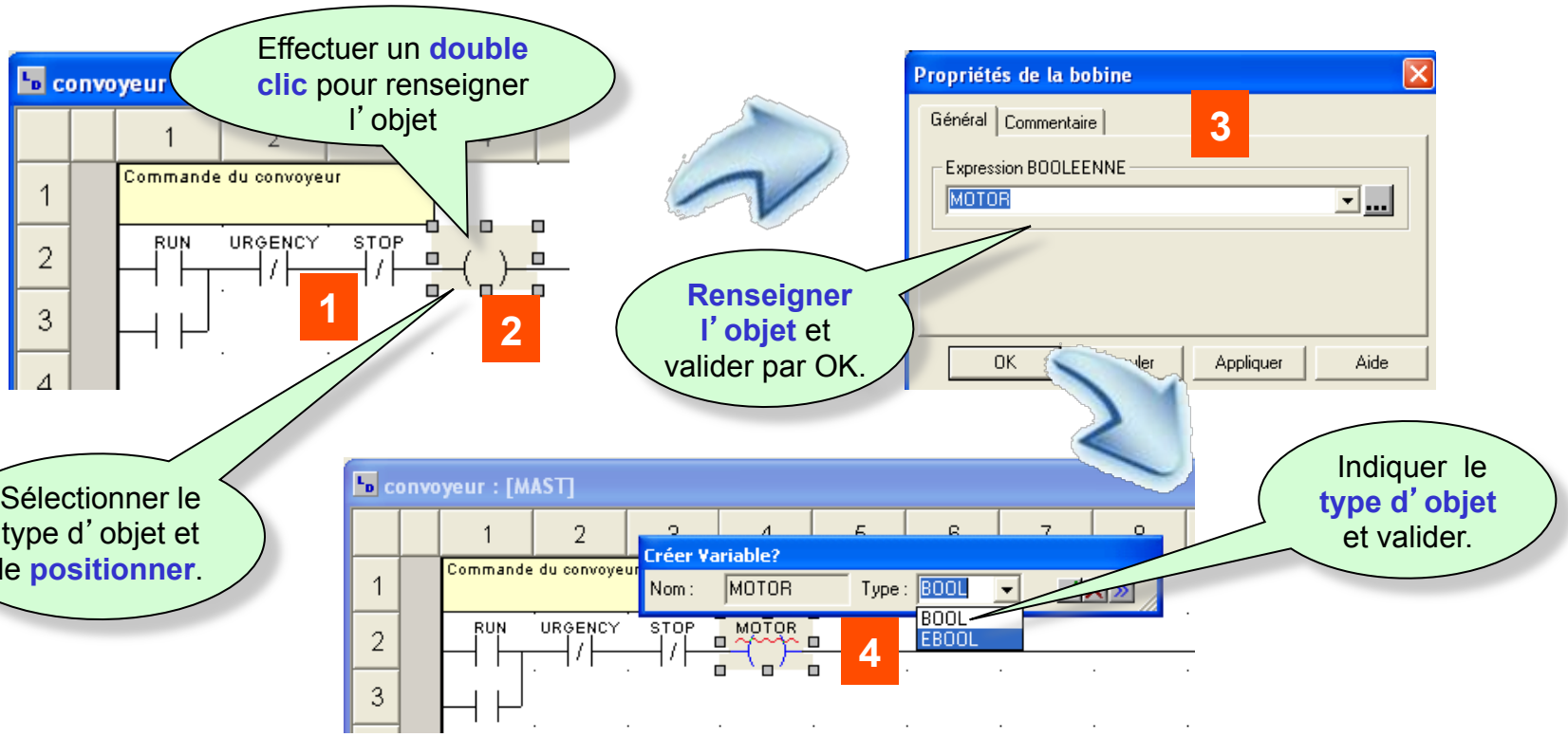


Remarque : L'écran de saisie est divisé en cellules recevant les différents objets. Un survol avec la souris donne la signification de l'objet.

Programmation du convoyeur en langage LD

Création de la section convoyeur (3/3)

Création du réseau de contacts avec déclaration des variables au fil de l'eau



Remarque : Le type d'objet proposé est toujours en cohérence avec l'objet sélectionné.

Programmation du convoyeur en langage LD

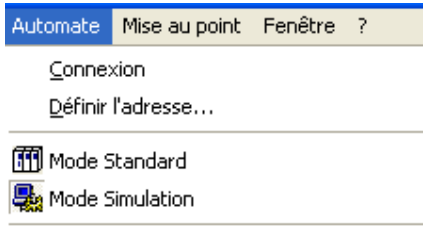
Analyse et première génération du projet (1/3)

La saisie d'un programme étant terminée, nous allons effectuer l'**Analyse du projet** (signaler les erreurs et avertissement dans le projet) puis la **Regénération du projet** (indispensable la première fois).

Exécution du programme sur le simulateur

Le projet peut être exécuté sur :
- L'automate et il faut dans ce cas définir la configuration.
- Un simulateur de l'automate et dans ce cas la définition de la configuration n'est pas nécessaire.

Lors de l'analyse et de la génération de code, Unity Pro tient compte de la cible automate ou simulateur



Remarque

Le simulateur d'automate permet de simuler un projet dans son ensemble avec toutes les tâches utilisateur associées.

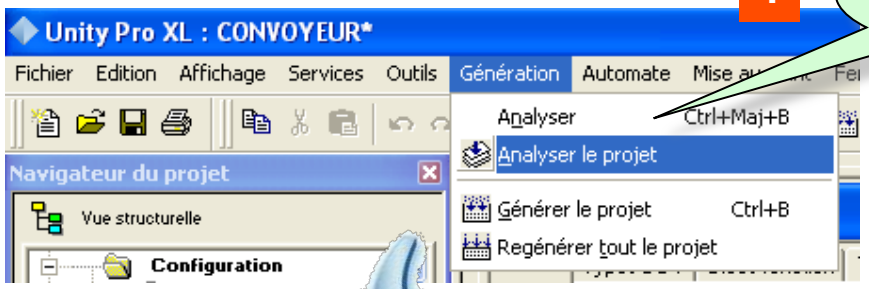
Programmation du convoyeur en langage LD

Analyse et première génération du projet (2/3)

L'analyse permet de vérifier l'absence d'erreurs de syntaxe dans votre application.
Affichage des erreurs et des avertissements avec navigation vers les éditeurs à l'origine de chacune des erreurs.

1

Lancer l'**analyse du projet.**



2

Affichage du **résultat de l'analyse** du projet.

Un **avertissement** signale un élément pouvant poser problème mais n'empêche pas le transfert dans le simulateur ou l'automate
Une **erreur** bloque tout transfert

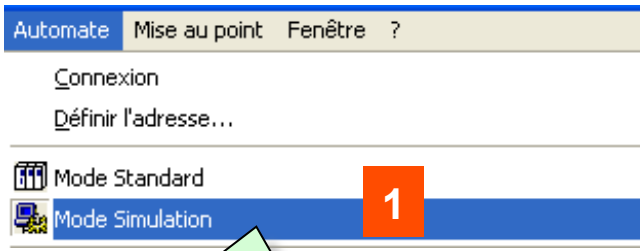
3

Effectuer un double clic sur l'**élément signalé en bleu ou en rouge**, Unity Pro se positionne automatiquement sur l'élément en cause.

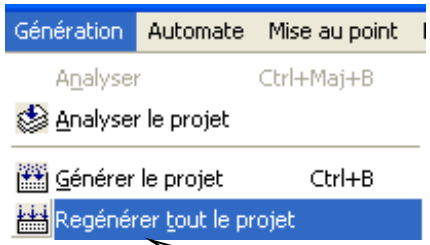
Programmation du convoyeur en langage LD

Analyse et première génération du projet (3/3)

Le premier transfert nécessite une regénération complète du projet.
Par la suite, il sera nécessaire uniquement de générer l'application pour prendre en compte les modifications.



Sélectionner la **cible d'exécution** du programme par le menu **Automate / Mode simulation.**



Sélectionner le menu **Génération / Regénérer le projet.**

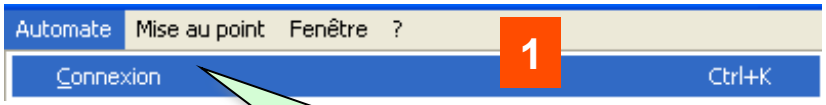
Affichage des **avertissements ou erreurs** éventuelles

Remarque : Les avertissements sont dus au fait que le bus CANopen n'est pas configuré

Programmation du convoyeur en langage LD

Mise au point du projet (1/6)

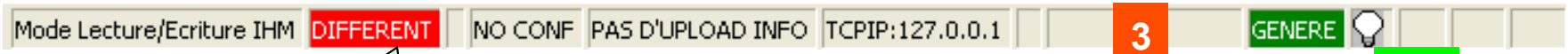
La regénération étant correcte, nous allons mettre au point l'application à l'aide du simulateur automate en nous connectant dans un premier temps avec celui-ci.



Sélectionner le menu **Automate / Connexion**
Le bandeau inférieur affiche l'état du simulateur.

2

Le simulateur se lance



Dans le bandeau, il est indiqué que **le projet ouvert dans Unity Pro et celui dans le simulateur sont différents.**

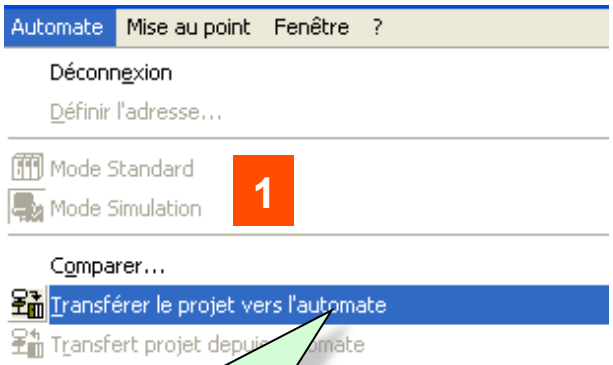
?

Remarque : Le **?** dans la barre des tâches signale que le simulateur est lancé **sans projet valide.**

Programmation du convoyeur en langage LD

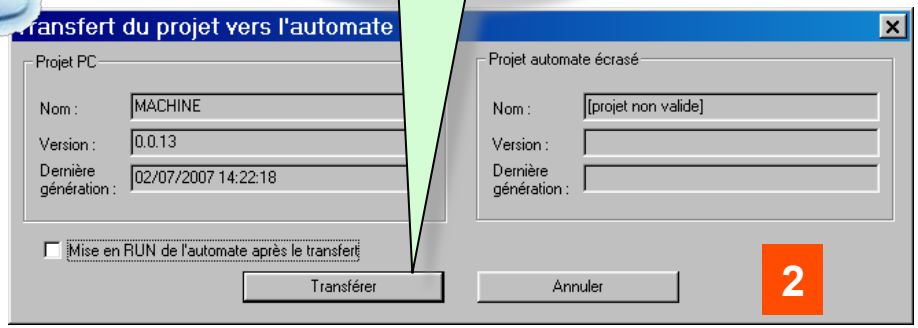
Mise au point du projet (2/7)

La connexion étant réalisée, nous pouvons transférer le projet dans le simulateur.

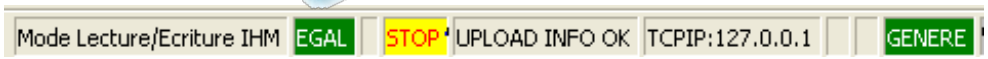


Sélectionner le menu **Automate / Transférer le projet vers l'automate.**

Cliquer sur **Transférer**.



3

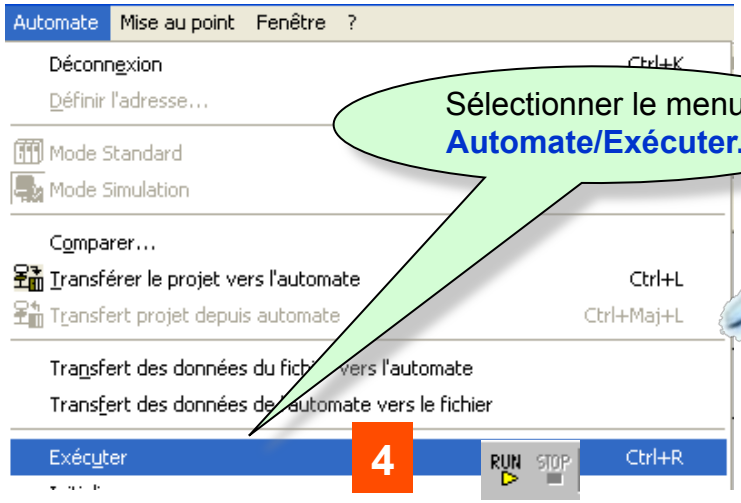


Le bandeau indique que **les programmes sont identiques** mais que l'automate est en **STOP**.

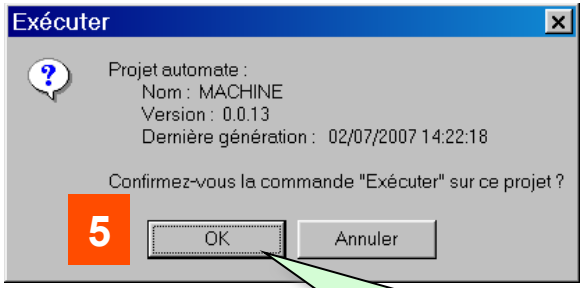
Programmation du convoyeur en langage LD

Mise au point du projet (3/7)

La connexion étant réalisée, nous pouvons exécuter le programme convoyeur dans le simulateur.

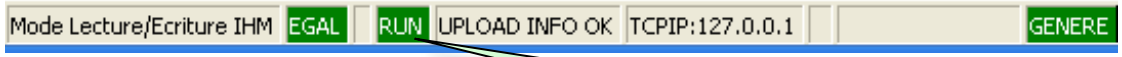


Sélectionner le menu **Automate/Exécuter.**



Cliquer sur Ok pour passer l'automate en RUN.

6

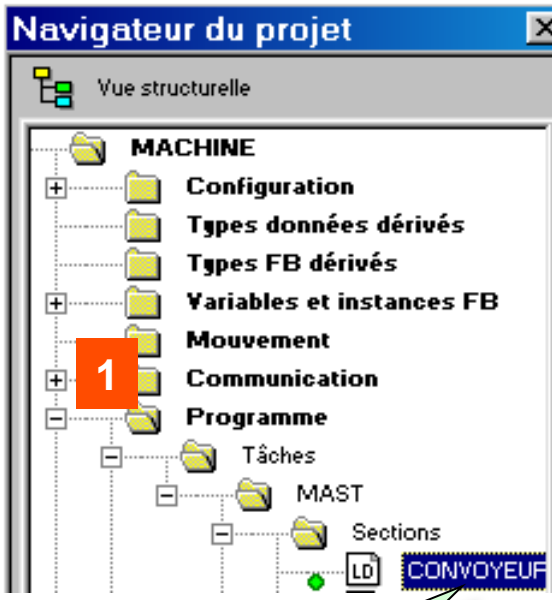


Le bandeau nous indique que l'automate est en **RUN**.

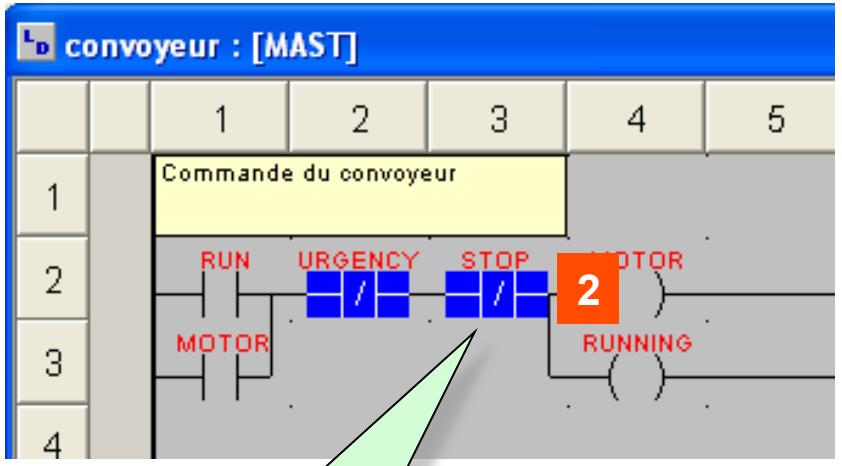
Programmation du convoyeur en langage LD

Mise au point du projet (4/7)

Nous allons pouvoir visualiser le programme en dynamique et modifier les variables pour simuler le fonctionnement du convoyeur.



Effectuer un double clic sur la **section Convoyeur**



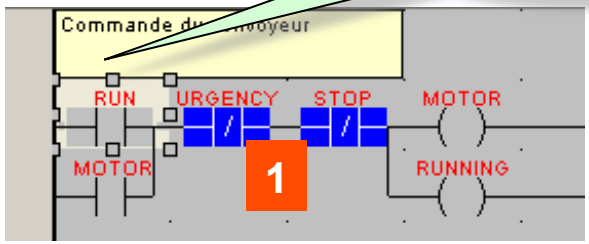
La section Convoyeur apparaît **en dynamique**. **Les contacts passants sont en vidéo inverse** (sur fond bleu).

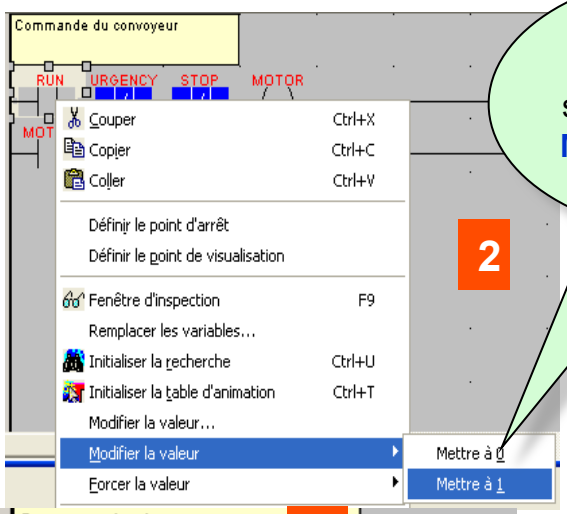
Programmation du convoyeur en langage LD

Mise au point du projet (5/7)

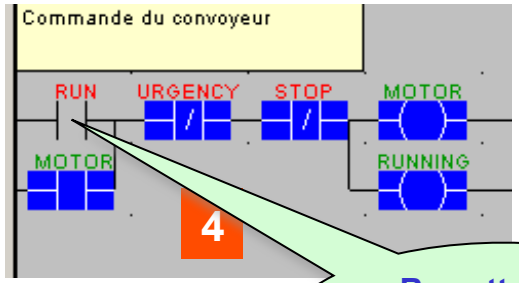
Modification des variables depuis l'écran de visualisation du schéma en ladder.

Sélectionner la **variable RUN**.

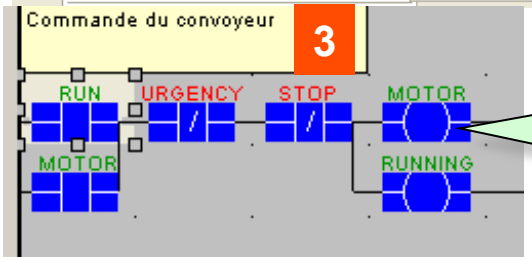




Effectuer un **clic droit** puis sélectionner le menu **Modifier la valeur** et Mettre la valeur à **1**.



Remettre à 0 la commande RUN
(En réitérant 1 et 2)

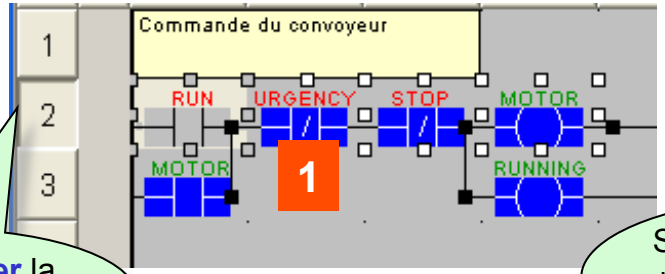


Le moteur a démarré et le voyant de marche est allumé.

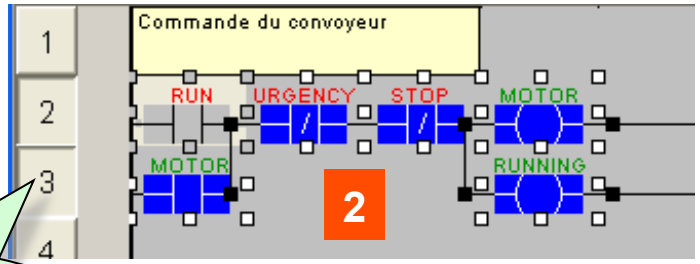
Programmation du convoyeur en langage LD

Mise au point du projet (6/7)

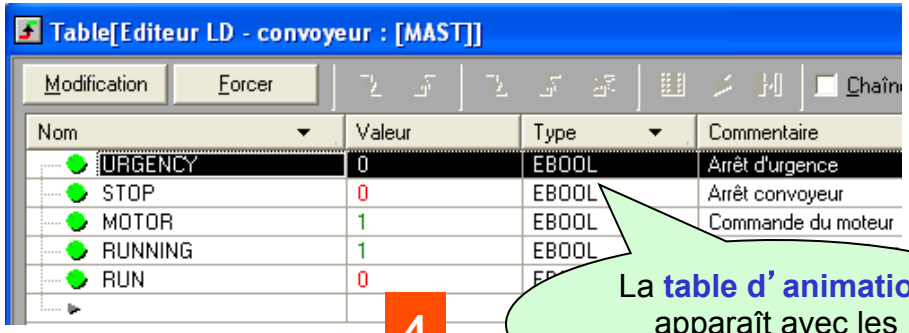
Initialisation d'une table d'animation pour visualiser l'état des variables de la section convoyeur.



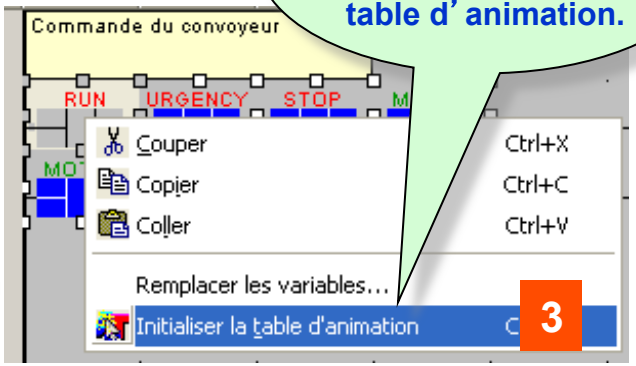
Sélectionner la **case 2**.



Sélectionner la case 3 par sélection multiple (**Shift**).



La **table d'animation** apparaît avec les variables sélectionnées.

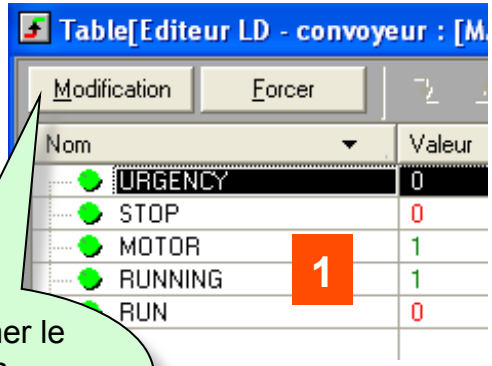


Effectuer un **clic droit** puis sélectionner le menu **Initialiser la table d'animation.**

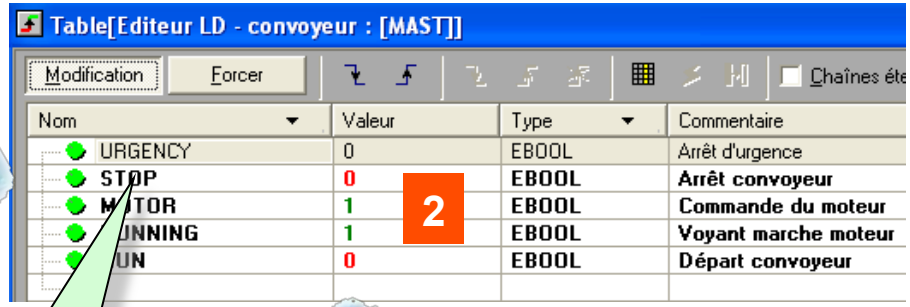
Programmation du convoyeur en langage LD

Mise au point du projet (7/7)

Modification des variables depuis la table d'animation.

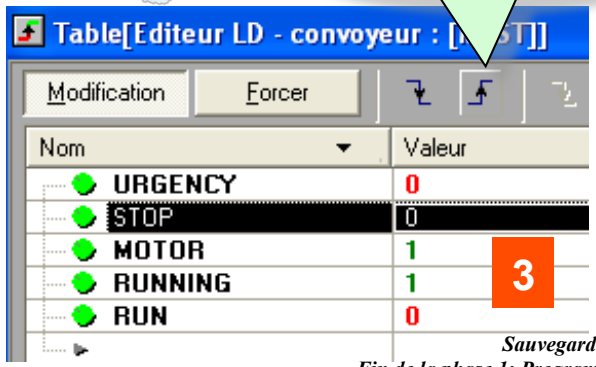


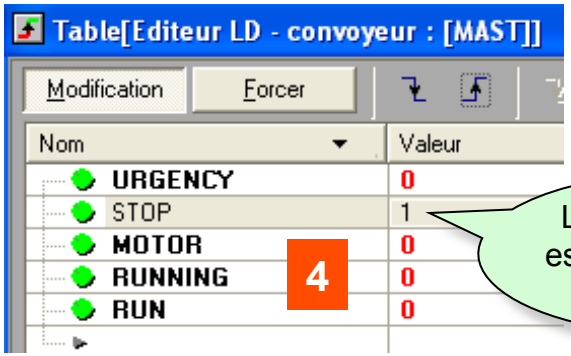
Sélectionner le bouton **Modification**.



Sélectionner la variable STOP.

Cliquer sur l'**icône de mise à 1**.





La variable STOP est à 1 et **le moteur s'arrête.**

Sauvegarder le Projet.
Fin de la phase 1: Programmer en LD.

Schneider Electric

Programmation du palettiseur en langage ST

Cahier des charges du palettiseur (1/2)

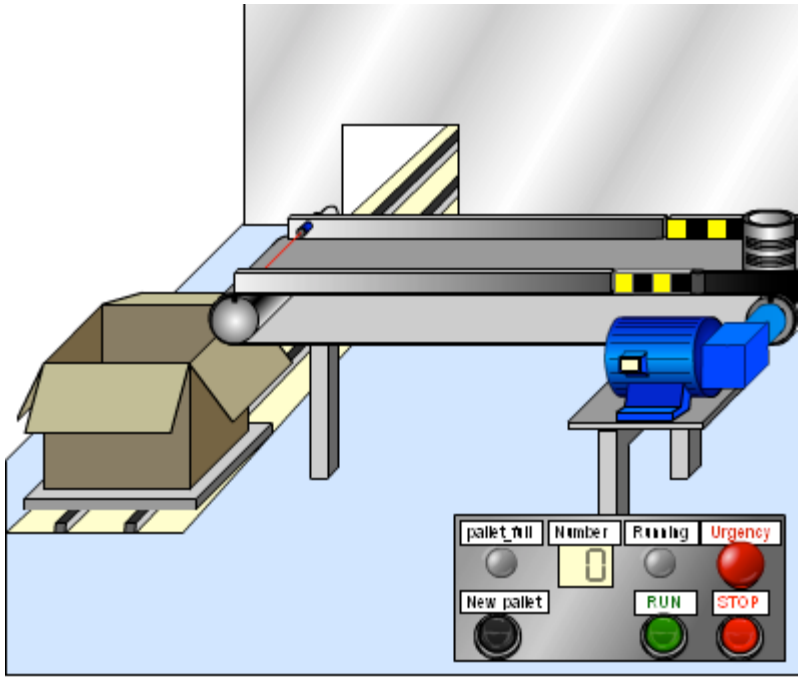
Nous allons maintenant réaliser une extension du projet en rajoutant le palettiseur à l'installation précédente. Le convoyeur amène des pièces qui sont ensuite rangées par 10 sur une palette.

Les besoins en entrées :

- Une entrée pour le capteur optique permettant le comptage des pièces (**Optical_sensor**).
- Une entrée **New_palett** pour évacuer la palette, remettre le compteur (**Number**) à 0 et appeler une nouvelle palette.

Les besoins en sorties

- Une sortie pour le voyant **Palett_full**
- Une sortie **Blocker** pour piloter le bloqueur des pièces qui arrivent.



Utilisez les boutons de la maquette pour comprendre le cahier des charges : RUN, New Pallet

Programmation du palettiseur en langage ST

Cahier des charges du palettiseur (2/2)

La chronologie de développement que nous vous proposons est la suivante :

Prise en charge de l'application

Cahier des charges

Analyse

Réalisation du projet

Déclaration des données

Création de la section en langage ST

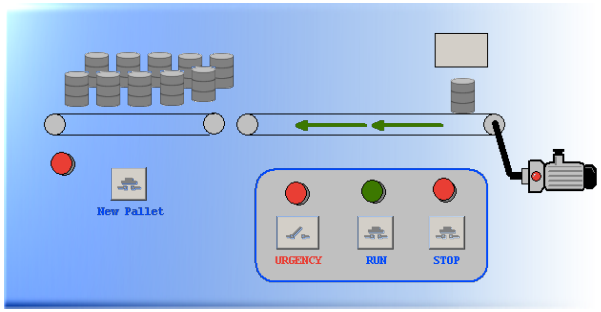
Mise au point

Ecrans d'exploitation

Pilotage à l'aide d'un écran d'exploitation

```
(* simulation du passage de piè
optical_sensor:= MOTOR and not

If re (optical_sensor) then
        inc (number);
end_if;
if re (new_pallet) then
        number:=0;
```



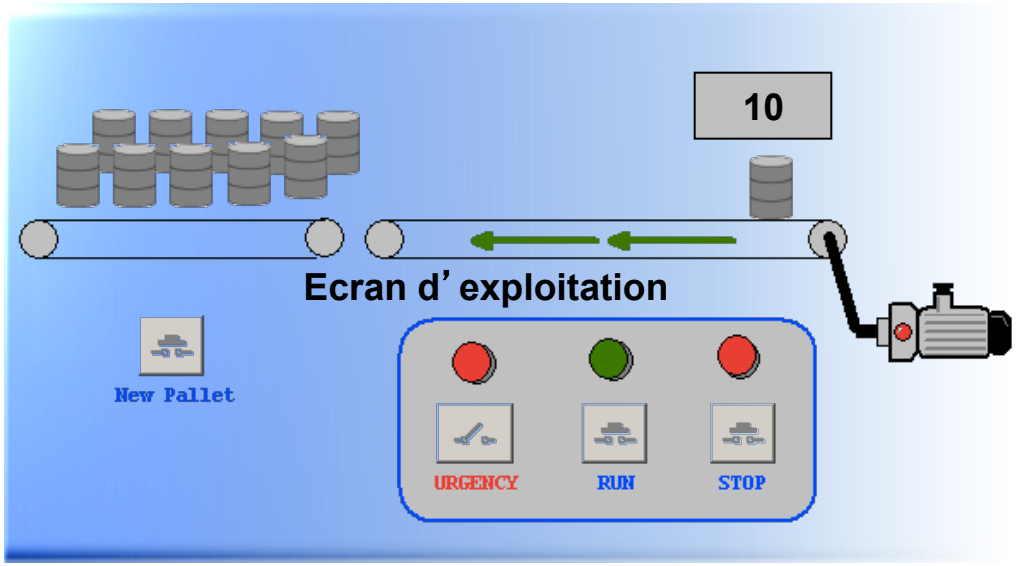
Programmation du palettiseur en langage ST

Analyse du cahier des charges

Structure du programme et visualisation du procédé dans Unity Pro avec un écran d'exploitation .

Structure de la section Palettisation

- Si le convoyeur est en marche, les boites sont déposées sur le convoyeur.
- Les boites sont comptées. Lorsque l'on atteint 10 boites, la palette est pleine et le bloqueur est activé.
- Sur demande opérateur, la palette est évacuée pour en traiter une nouvelle.
- La section sera programmée en langage Littéral Structuré (**ST**).
- Un écran d'exploitation permet la commande et la visualisation de l'état du procédé.



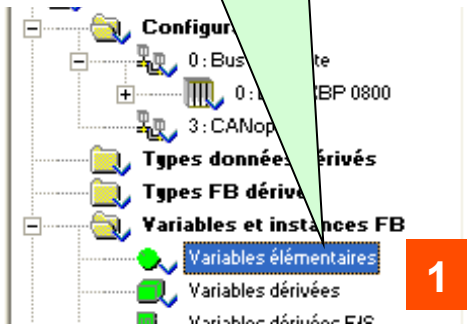
Remarque :
Pour simuler l'approvisionnement des boites, chaque seconde par exemple, nous utiliserons le **bit système %S6**

Programmation du palettiseur en langage ST

Déclaration des données

Nous allons ajouter les nouvelles variables à la liste des variables déjà existantes.
Se déconnecter du simulateur (Menu **AUTOMATE** / **Déconnexion**) pour revenir en mode local.

Effectuer un double clic sur **Variables élémentaires** pour accéder à l'éditeur de données.



Nom	Type	Adresse	Valeur	Commentaire	Utilisé
MOTOR	EBOOL			Commande du moteur	3
RUN	EBOOL			Départ convoyeur	2
RUNNING	EBOOL			Voyant marche mote...	2
STOP	EBOOL			Arrêt convoyeur	2
URGENCY	EBOOL			Arrêt d'urgence	2

2

L'écran apparaît avec les **variables déjà déclarées**.

Symboliser toutes les données sans laisser d'espace dans le nom

Nom	Type	...	Valeur	Commentaire
BLOCKER	EBOOL			Bloqueur de pièces
NUMBER	INT			Nombre de pièces dans la palette
MOTOR	EBOOL			Commande du moteur
NEW_PALETT	EBOOL			Appel nouvelle palette
OPTICAL_SENSOR	EBOOL			Détecteur de passage des pièces
PALETT_FULL	EBOOL			Palette pleine
RUN	EBOOL			Départ convoyeur
RUNNING	EBOOL			Voyant marche moteur
STOP	EBOOL			Arrêt convoyeur
URGENCY	EBOOL			Arrêt d'urgence

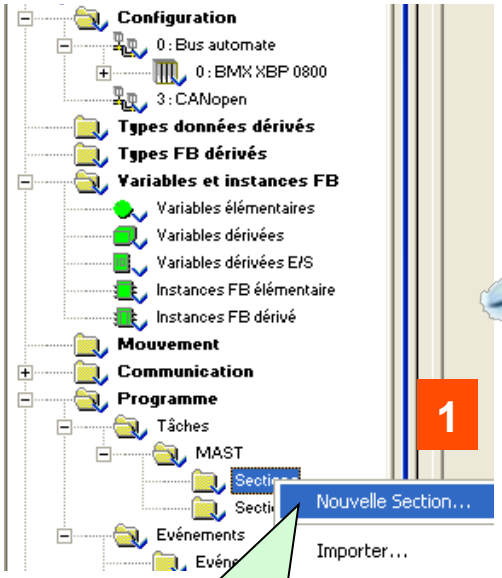
3

Aide à la saisie du projet

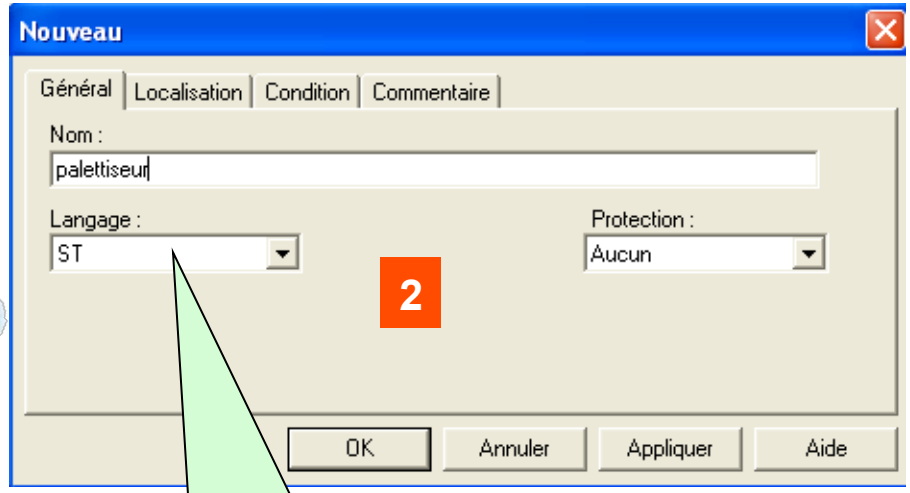
Programmation du palettiseur en langage ST

Création de la section palettiseur (1/4)

Création de la section palettiseur en langage structuré (ST).



Effectuer un clic droit sur **Section** et sélectionner le menu **Nouvelle Section.**

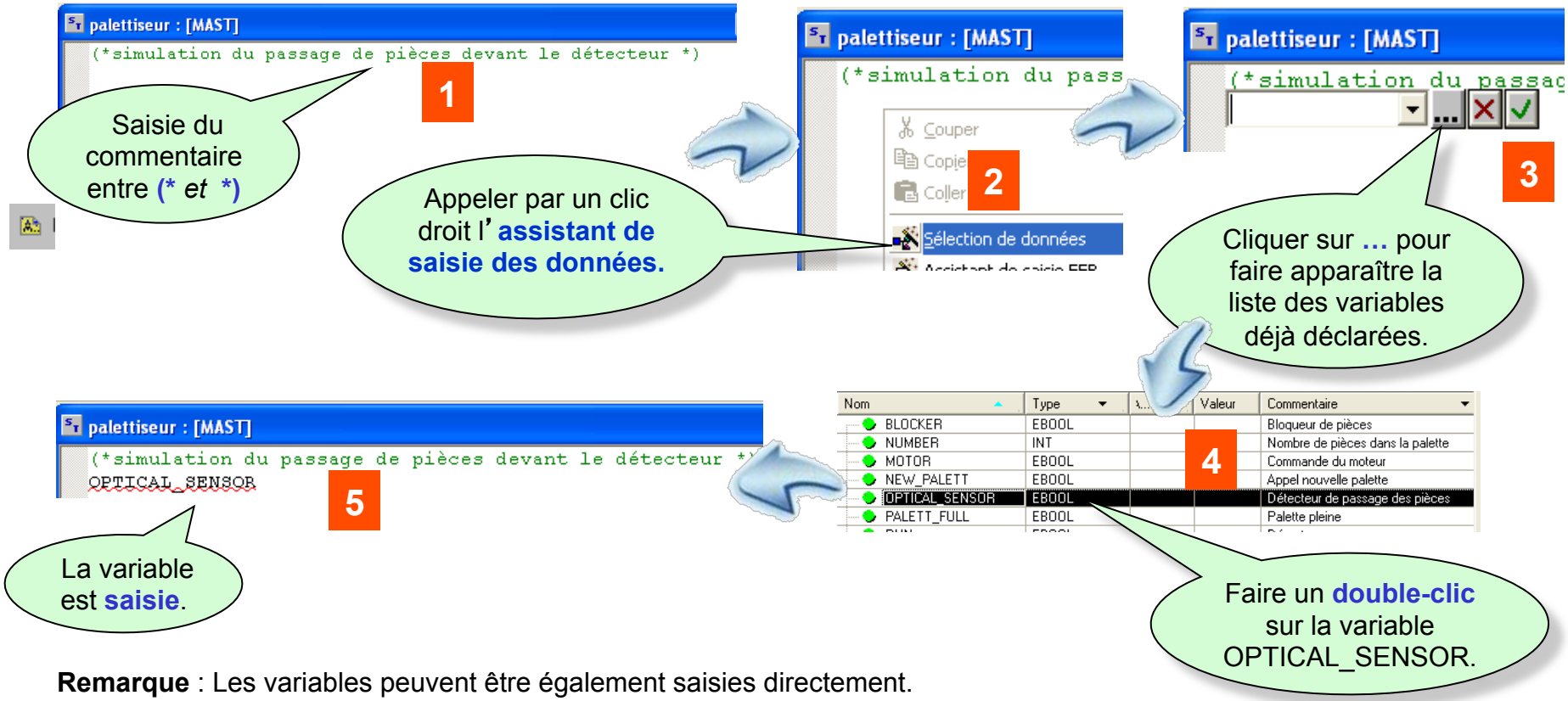


Saisir le **Nom de la section** (Palettiseur) et sélectionner le **langage** à utiliser, ici **ST** puis valider par OK.

Programmation du palettiseur en langage ST

Création de la section palettiseur (2/4)

Création du programme en littéral structuré : logique de simulation du détecteur optique en utilisant des variables déjà déclarées



Remarque : Les variables peuvent être également saisies directement.

Programmation du palettiseur en langage ST

Création de la section palettiseur (3/4)

Saisir le programme complet tel que ci-dessous.
Le contrôle du nombre de pièces se fait à l'aide de structure de contrôle : « If then else »
Un assistant de saisie est proposé.

Cliquer sur l'icône associé au IF
Un masque de saisie est affiché

2

Renseigner le masque

1

Cliquer sur l'emplacement ou vous souhaitez **insérer** la structure de contrôle

```
(* Simulation présence pièce *)
Optical_sensor := Motor And not Palett_Full And
(* Comptage pièces *)
IF RE (Optical_sensor) THEN
INC (number);
END_IF;
if re (New_palett) then
        number:=0;
end_if;
if number=10 then
        Palett_Full:=true;
else
        Palett_full:=false;
end_if;

(*blocage de l'arrivée des pièces si palette pleine*)
blocker:=Palett_full;
```

```
IF  THEN

  ELSIF  THEN

    ELSE

END_IF;
```

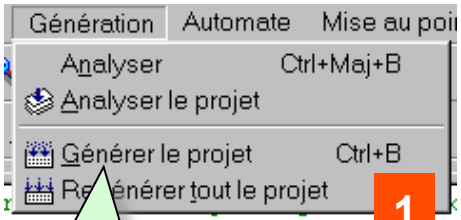
3

Aide à la saisie du projet

Programmation du palettiseur en langage ST

Création de la section palettiseur (4/4)

Analyse du projet, génération et transfert dans le simulateur pour la mise au point.



Sélectionner le menu **Analyse** pour contrôler les modifications, apporter les corrections si nécessaire, puis effectuer une génération partielle du code à l'aide du menu **Générer le projet.**

Passer en **Connexion** et **Transférer le projet** dans le simulateur par la même méthode que celle utilisée dans la phase 1 et passer en **RUN**

Remarque : la commande Générer le Projet/ Regénérer tout le projet enchaîne l'analyse, si celle-ci n'a pas été réalisée au préalable

Programmation du palettiseur en langage ST

Mise au point de la section palettiseur (1/2)

Des couleurs sont utilisées pour afficher l'état des variables en visualisation dynamique. Les fenêtres d'inspection permettent de compléter la visualisation dans l'éditeur.

Couleurs des variables en fonction de leur état

Type booléen :
Verte si la variable est TRUE,
Rouge si la variable est FALSE

Fond **Jaune** pour les autres types.

```
(* simulation du passage de pièces devant le détecteur*)
optical_sensor:= MOTOR and not pallet_full and  %S6;
(* Comptage pièces*)
If re (optical_sensor) then
        inc (number);
end_if;
if re (new_pallet) then
        number:=0;
end_if;
if number=10 then
        pallet_full:=true;
else
        pallet_full:=false;
end_if;

(*blocage de l'arrivée des pièces si palette pleine*)
blocker:=pallet_full;
```

Number
6

pallet_full : TRUE

Les **fenêtres inspection** permettent de visualiser l'état des variables.
Pour cela cliquer sur la variable puis sur l'**icône lunette**.
Les fenêtres inspection restent affichées malgré le scrolling

Pour visualiser la valeur d'une variable numérique dans une **info bulle** il suffit de pointer la variable avec la souris.

Programmation du palettiseur en langage ST

Mise au point de la section palettiseur (2/2)

Initialisation d'une table d'animation pour visualiser l'état des variables de la section Palettiseur.
Modification des variables pour vérifier le fonctionnement de la machine.

1

Sélectionner le programme de **la Section** palettiseur.

```
(* simulation du passage de pièces devant le détecteur*)
optical_sensor:= MOTOR and not pal

If re (optical_sensor) then
        inc (number);
end_if;
if re (new_pallet) then
        number:=0;
end_if;
if number=10 then
        pallet_full:=true;
else
        pallet_full:=false;
end_if;

(*blocage de l'arrivée des pièces
blocker:=pallet_full;
```

number 0

Couper Ctrl+X
Copier Ctrl+C
Coller Ctrl+V
Sélection de données Ctrl+D
Assistant de saisie EFB... Ctrl+I
Créer variable... Maj+Entrée
Commentaire F8
Sélection de sous-programme...
Atteindre... Ctrl+G
Initialiser la recherche Ctrl+U
Initialiser la table d'animation Ctrl+T
Aide ... F1
Propri... données Ctrl+Entrée
Caractéristiques Alt+Entrée

2

Effectuer un clic droit et sélectionner le menu **Initialiser la table d' animation** ou **CTL + T.**

Modification | Forcer | Chaîne

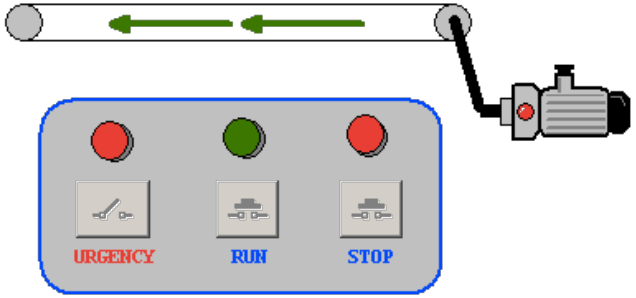
Nom	Valeur	Type	Commentaire
Optical_sensor	0	EBOOL	Détecteur de pièces
Motor	0	EBOOL	Commande moteur
Palett_Full	0	EBOOL	Palette pleine
%S6	1	BOOL	
number	0	INT	Nombre de pièces
New_palett	0	EBOOL	Nouvelle palette
blocker	0	EBOOL	Bloqueur de pièces

Programmation du palettiseur en langage ST

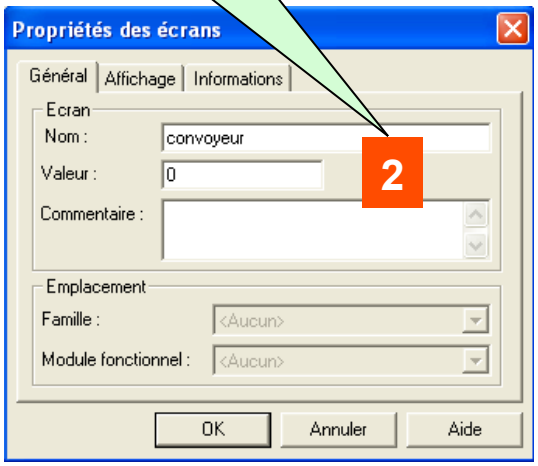
Les écrans d'exploitation (1/4)

Unity Pro propose des écrans d'exploitation destinés à faciliter l'exploitation d'un procédé automatisé.
Ces écran peuvent être construit en mode local ou connecté
Nous allons créer un écran associé à la machine en mode connecté

Sélectionner dans le navigateur **Ecrans d'exploitation** le menu **Nouvel écran.**



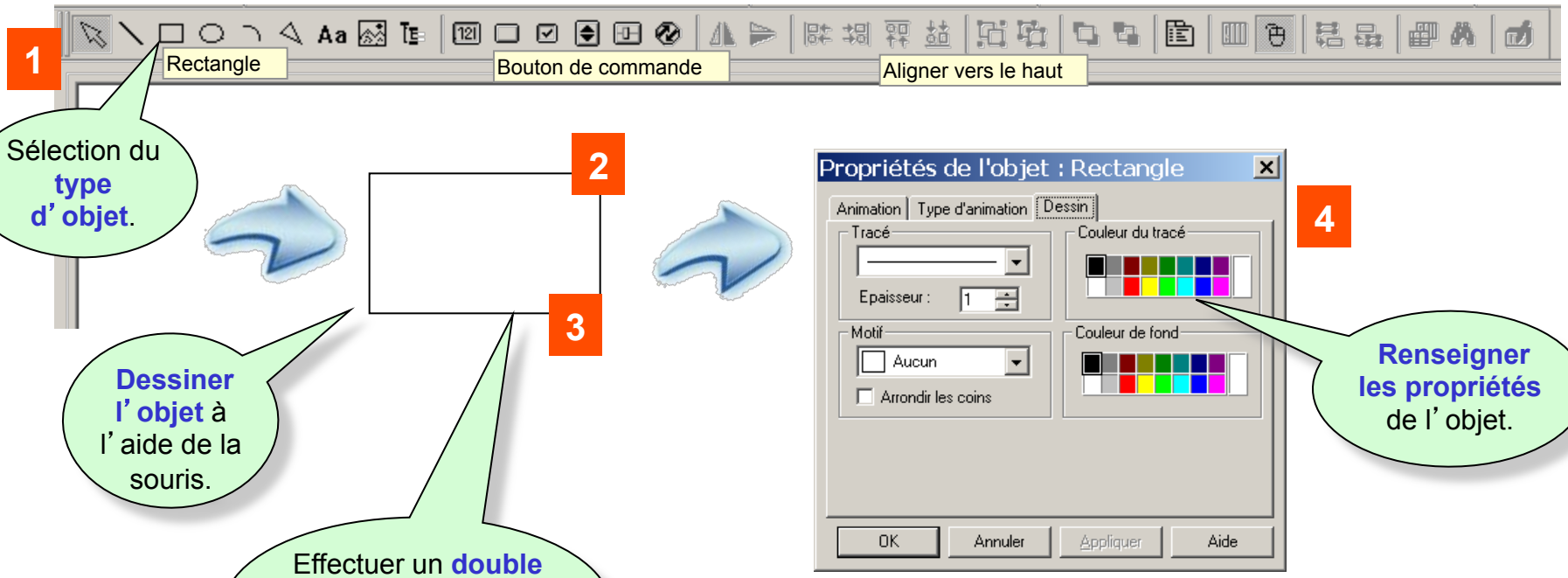
Saisir le nom de l'écran et valider par **OK**.



Programmation du palettiseur en langage ST

Les écrans d'exploitation (2/4)

Création du contenu de l'écran Convoyeur.
L'écran de saisie propose un ensemble d'objets graphiques auxquels peuvent être associées des variables d'animation. Le principe de saisie est le suivant :



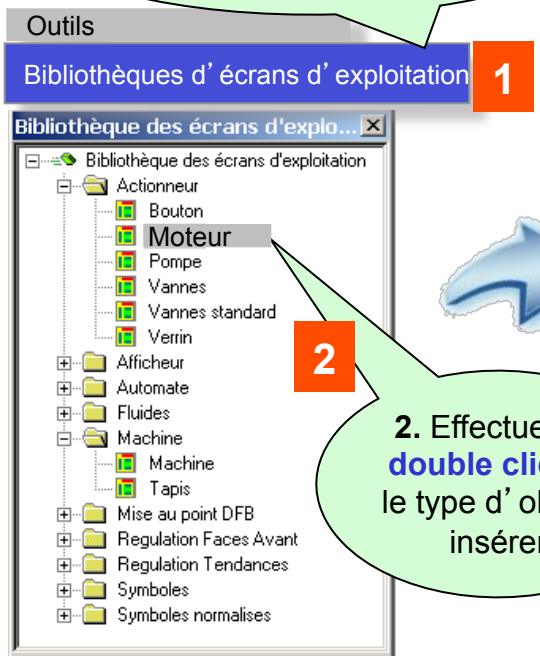
Onglet Dessin : Modification de la couleur de l'objet
Onglet Type d'animation : Choix du type d'animation et condition d'affichage
Onglet Animation : Choix de la variable d'animation et condition d'affichage

Programmation du palettiseur en langage ST

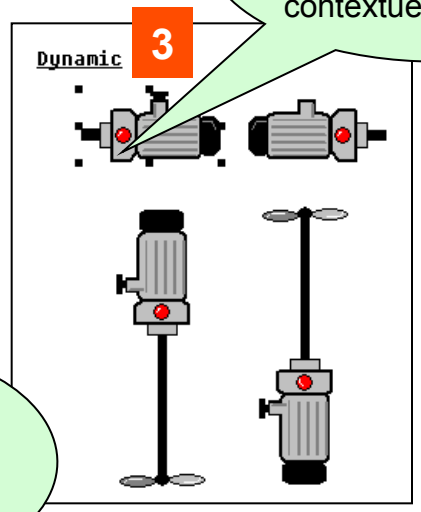
Les écrans d'exploitation (3/4)

Création du contenu de l'écran Convoyeur : Objets issus de la bibliothèque.
Unity propose une bibliothèque d'objets prédéfinis : actionneurs, afficheurs, automates, machines.
Le principe de saisie est le suivant :

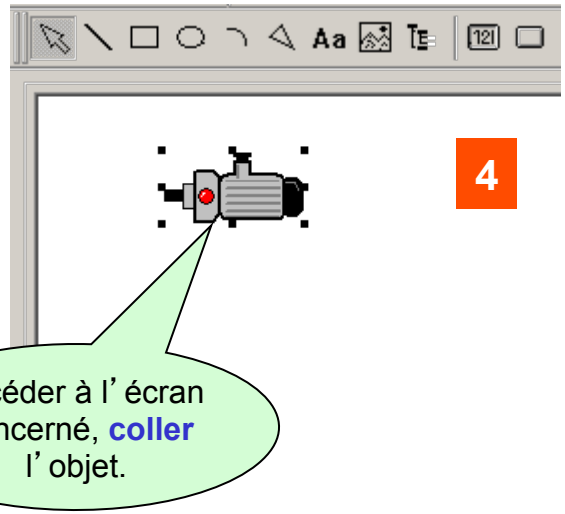
Sélectionner le menu **Outils / Bibliothèques des écrans d'exploitation.**



2. Effectuer un **double clic** sur le type d'objet à insérer



Sélectionner l'objet à insérer et le **copier** à l'aide du menu contextuel (**clic droit**)

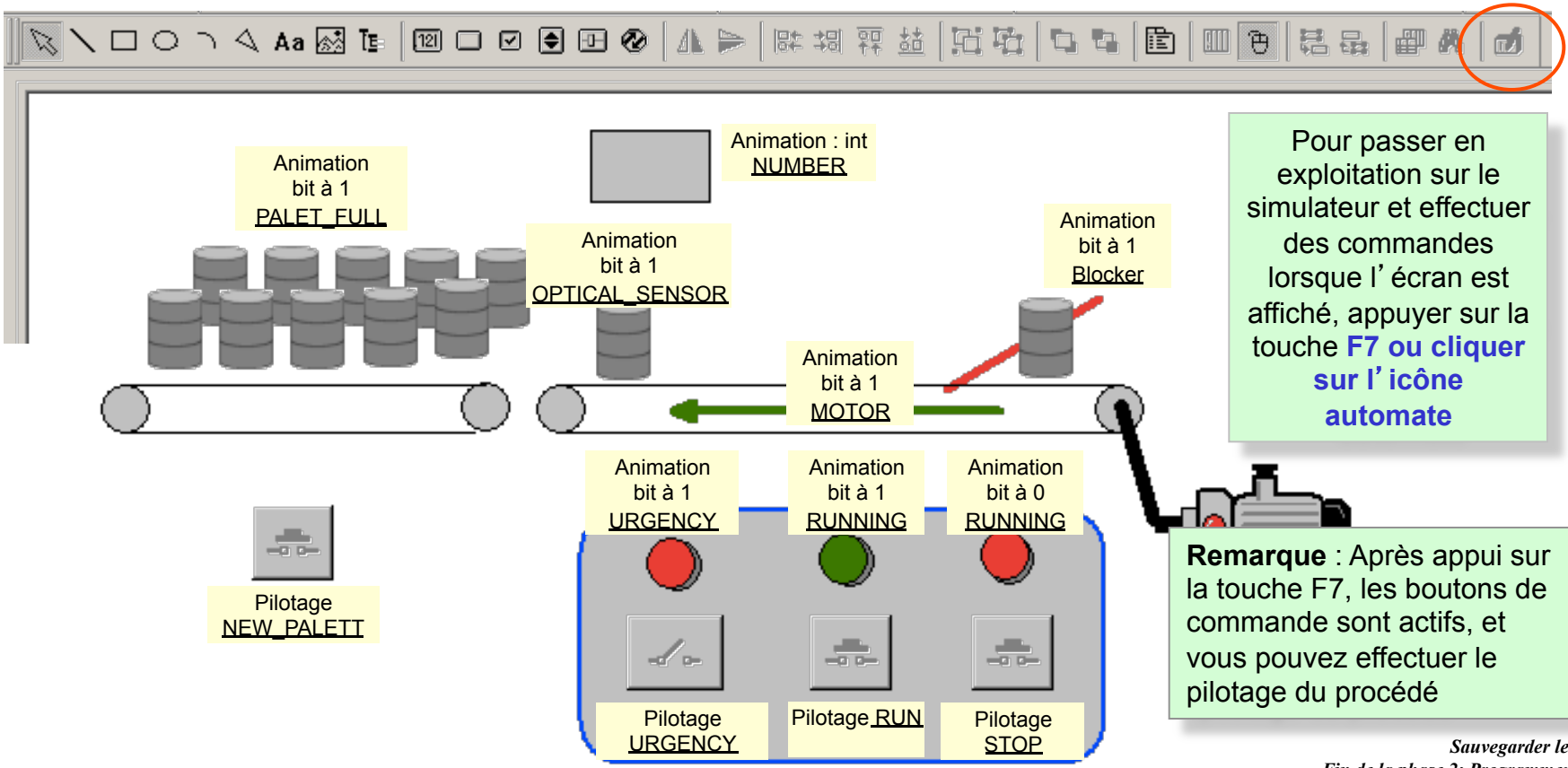


Accéder à l'écran concerné, **coller** l'objet.

Programmation du palettiseur en langage ST

Les écrans d'exploitation (4/4)

Vous allez créer le contenu de l'écran convoyeur, les textes en jaune indiquent les variables à associer aux objets (par l'onglet "animation" ou par l'onglet "pilotage" pour les propriétés de l'objet)



Pour passer en exploitation sur le simulateur et effectuer des commandes lorsque l'écran est affiché, appuyer sur la touche **F7 ou cliquer sur l'icône automate**

Remarque : Après appui sur la touche F7, les boutons de commande sont actifs, et vous pouvez effectuer le pilotage du procédé

Sauvegarder le projet.
Fin de la phase 2: Programmer en ST.

Extension du projet en mode connecté

Cahier des charges de la phase 3 (1/2)

Extension des fonctionnalités de la machine par modification du projet en ligne.
Notre application convoyeur s'arrête s'il n'y a pas de pièce présente pendant 10 s
Une commande de pilotage du convoyeur est installée à l'autre extrémité du convoyeur.

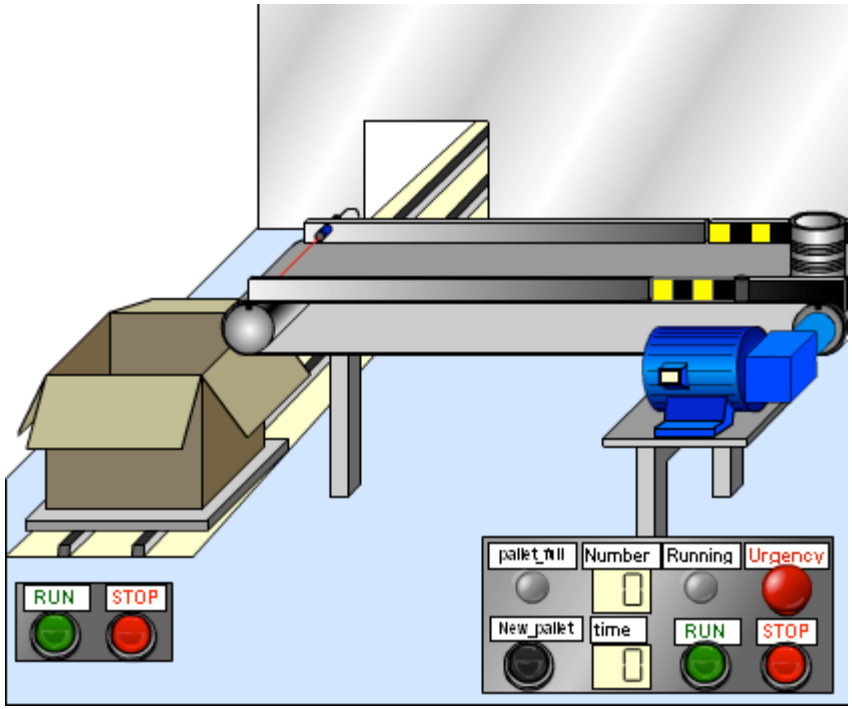
Les besoins en entrées :

- Le convoyeur peut être arrêté ou démarré depuis l'autre extrémité du convoyeur (entrées **RUN1** et **STOP1**)
- Une entrée **Absence_piece_entrée** permet la simulation de l'absence d'une pièce en entrée du convoyeur.

Les besoins en variables internes :

Afin d'économiser l'énergie, l'ajout d'une temporisation arrêtera le moteur automatiquement lorsqu'il y a absence de pièce pendant plus de 10 secondes.

- La variable fin de temporisation (**Tempo_moteur**) va arrêter le moteur.
- La variable(**Temps**) contient le temps courant de la temporisation.



Utilisez les boutons de la maquette pour comprendre le cahier des charges

Extension de l'application en mode connecté

Cahier des charges de la phase (2/2)

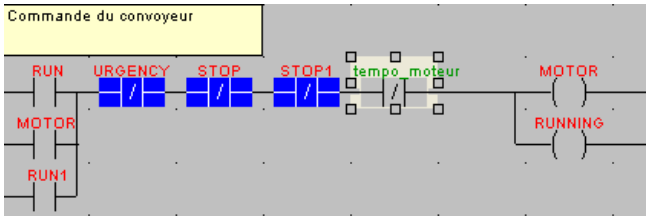
La chronologie de développement que nous vous proposons est la suivante :

Modification en ligne du projet

Cahier des charges

Modification du programme LD en ligne

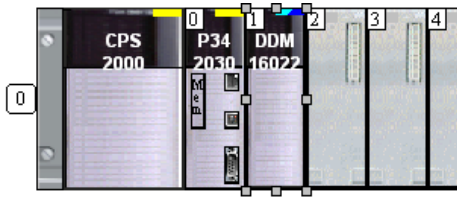
Mise au point



Utilisation d'un automate réel

Configuration des entrées/sorties de l'automate

Affectation des données sur les voies d'entrées/sorties



Nom	Type	Adresse
BLOCKER	EBOOL	%Q0.1.19
RUN	EBOOL	%I0.1.0

Extension de l'application en mode connecté

Modification en ligne de la section Convoyeur

Les modifications à effectuer en ligne sont les suivantes :

Modification à effectuer sur la structure du programme :

1.Création de la nouvelle commande de pilotage à l'extrémité du convoyeur : insertion des contacts RUN1 et STOP1

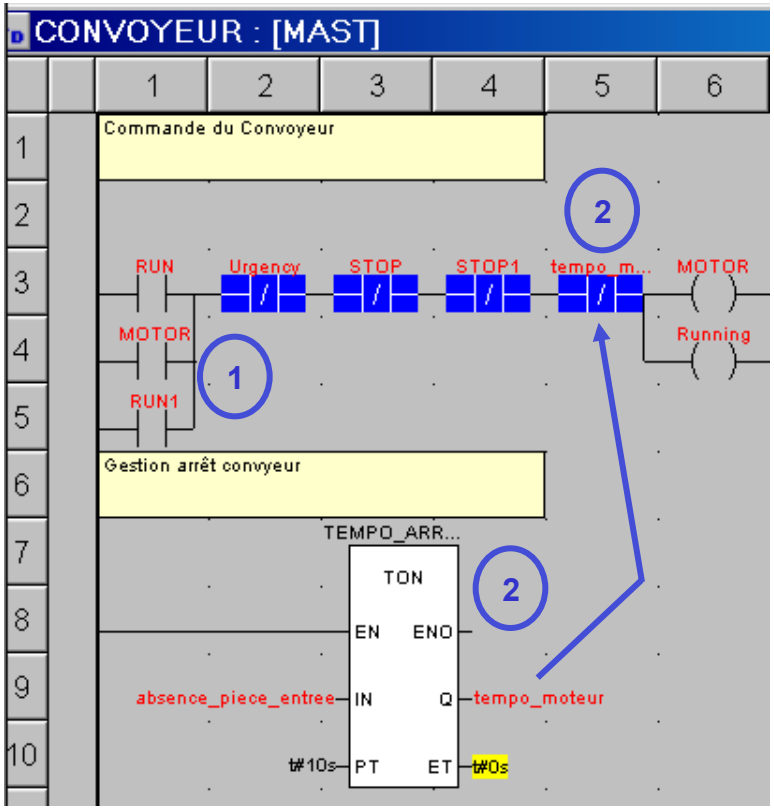
2. Insertion de la temporisation d'arrêt du convoyeur sur absence de pièce pendant 10 s

Les nouvelles variables d'entrées

Nom	Type	Commentaire
RUN1	EBOOL	Départ convoyeur 1
STOP1	EBOOL	Arrêt convoyeur1
Absence-piece-entree.	BOOL	Simulation absence pièces

Les nouvelles variables de sorties

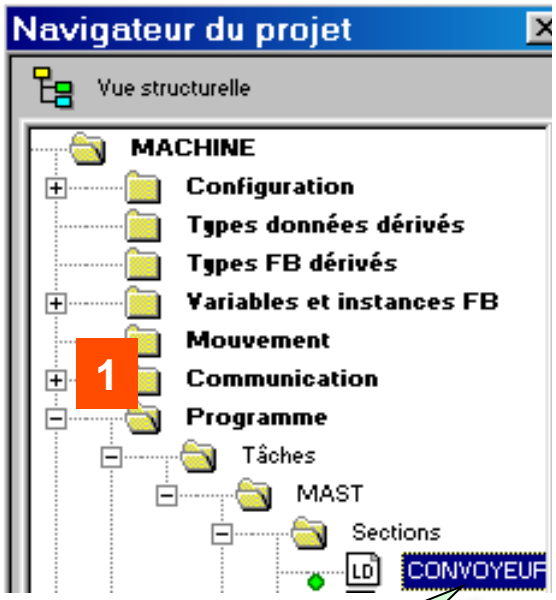
Nom	Type	Commentaire
Tempo_Motor	EBOOL	Fin tempo 10s



Extension de l'application en mode connecté

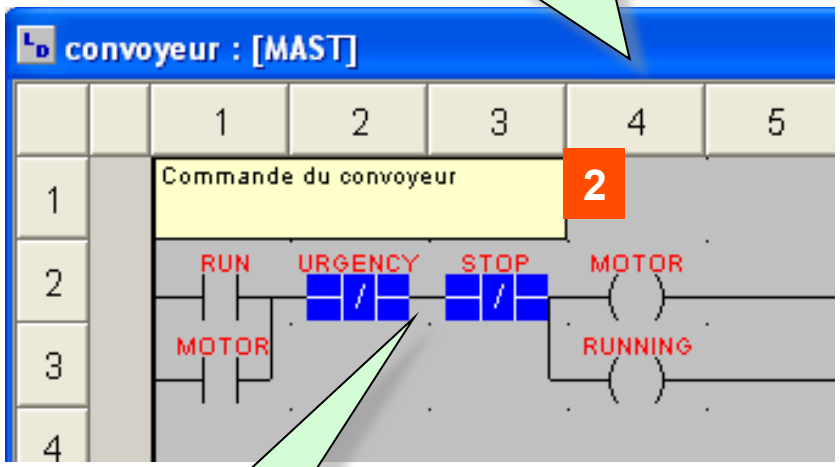
Modification en ligne de la section Convoyeur

Affichage de la section « convoyeur » en visualisation dynamique



Effectuer un **double clic** sur la section Convoyeur

La **section** convoyeur apparaît en **dynamique**.

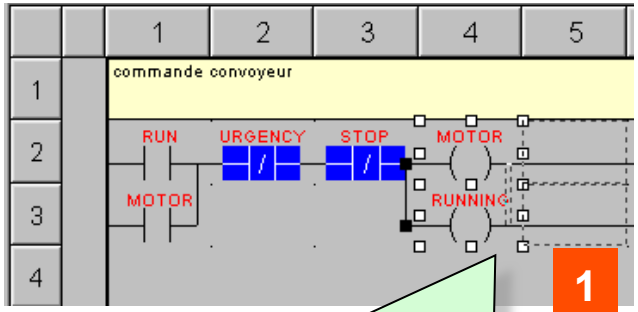


Les contacts passants sont en vidéo inverse (sur fond bleu)

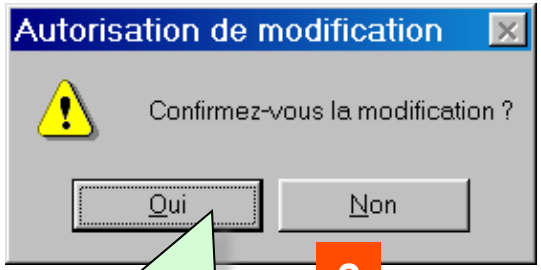
Extension de l'application en mode connecté

Modification en ligne de la section Convoyeur

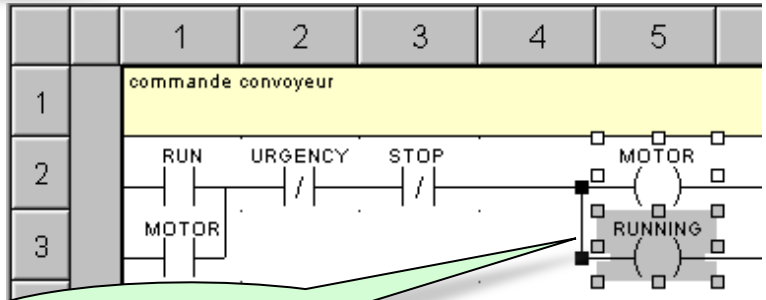
Insertion des contacts RUN1 et STOP1 :
Un déplacement des bobines va être nécessaire pour insérer un contact dans le réseau.



Sélectionner les 2 contacts et la barre verticale avec **Shift/Clic droit**. Maintenir le Clic droit et déplacer la sélection.



Confirmer la modification en cliquant sur **Oui.**

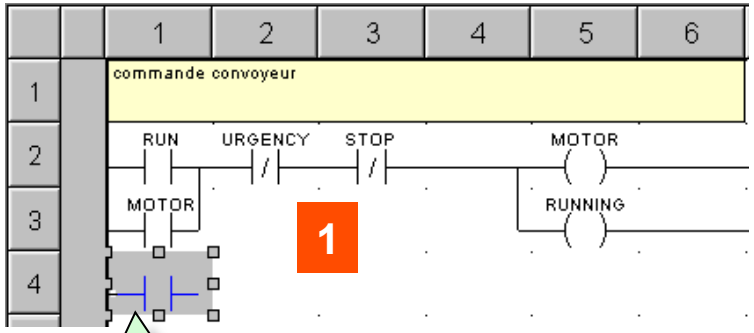


Les contacts ont été déplacés.

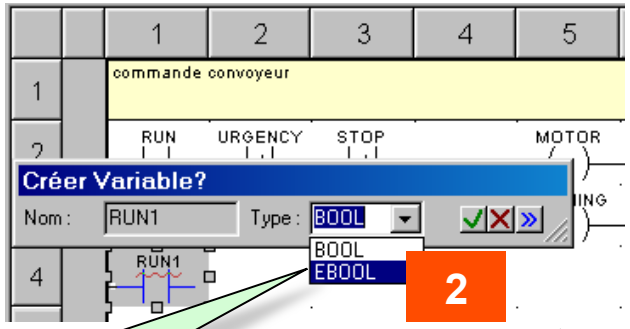
Extension de l'application en mode connecté

Modification en ligne de la section Convoyeur

Insertion des contacts supplémentaires avec déclaration de variables au fil de l'eau.

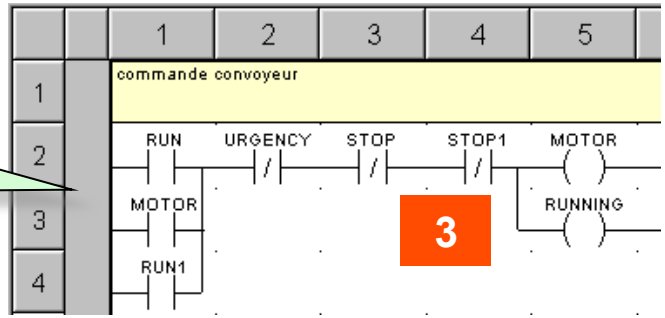


Insertion du contact et double clic pour renseigner l'objet.



Renseigner l'objet et le type d'objet puis valider.

Compléter le schéma avec les contact RUN1 et STOP1

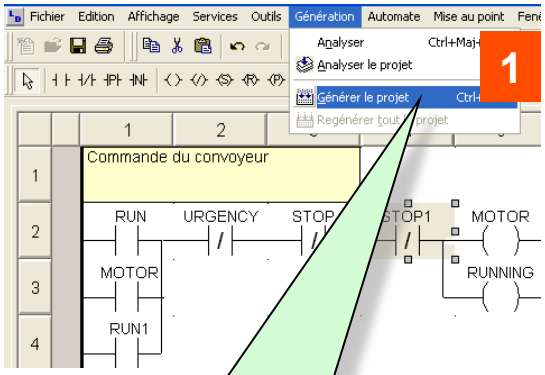


Remarque : Les procédures sont identiques à celles réalisées en phase 1

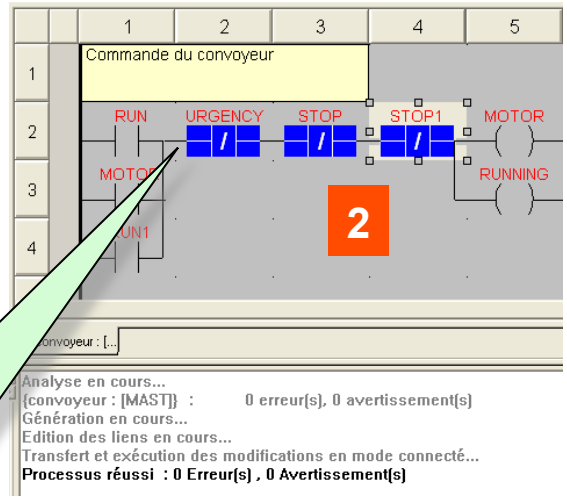
Extension de l'application en mode connecté

Modification en ligne de la section Convoyeur

Prise en compte des modifications : analyse des modifications et génération partielle du code



Sélectionner le menu **Analyse** pour vérifier que le projet ne comporte pas d'erreur, puis effectuer une génération partielle du code à l'aide du menu **Générer le projet.**



Le **projet** modifié **devient actif** dans l'automate (la modification a été effectuée automate en RUN).

Remarques :

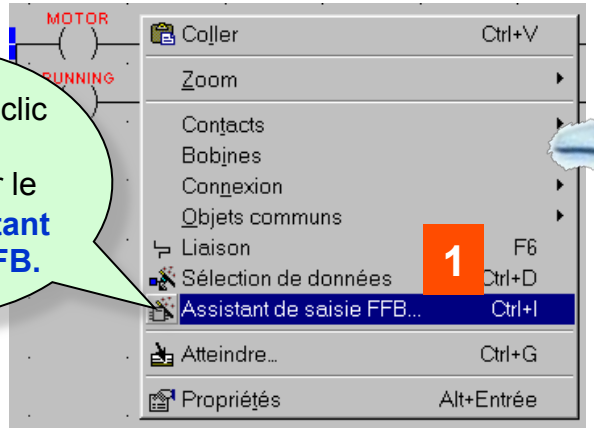
La mise au point s'effectue de la même façon que lors de la réalisation de la section convoyeur, via une table d'animation

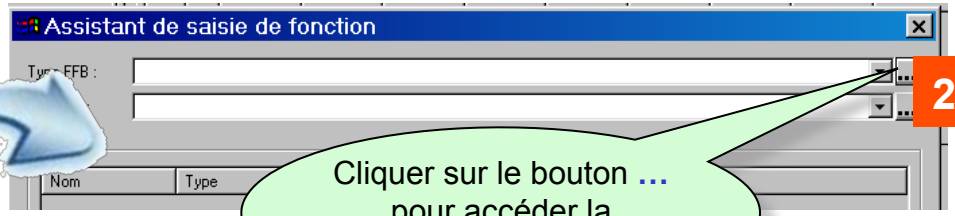
Extension de l'application en mode connecté

Modification en ligne de la section Convoyeur

Sélection d'un temporisateur pour la surveillance d'arrivée pièces.

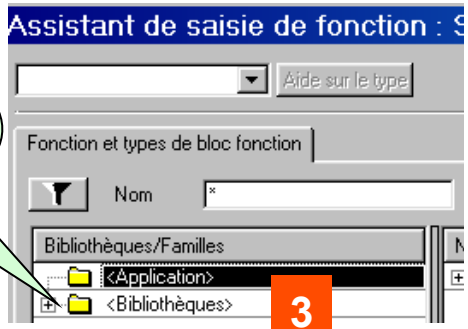
Effectuer un clic droit puis sélectionner le menu **Assistant de saisie FFB.**

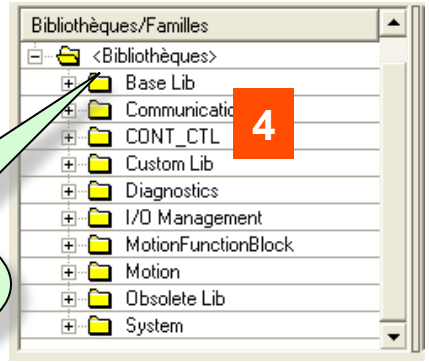




Cliquer sur le bouton **…** pour accéder la **bibliothèque des EFB**

Cliquer sur **Bibliothèques**





Sélectionner la famille **Base Lib**

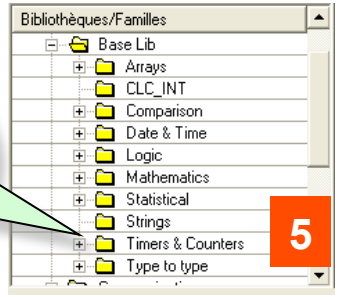
Remarque : Le répertoire **Application** contient les blocs fonctions déjà utilisés dans le projet

Extension de l'application en mode connecté

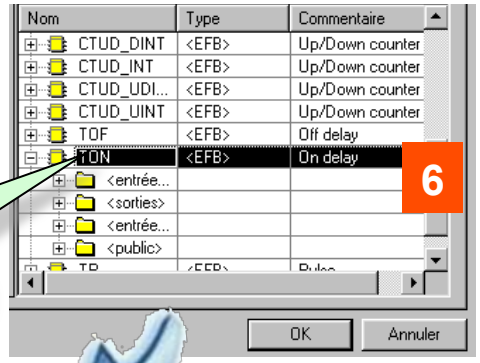
Modification en ligne de la section Convoyeur

Ajout de la temporisation d'arrêt convoyeur, renseignement du bloc fonction TON à l'aide de l'assistant de saisie de fonction

Sélectionner la famille **Timers & Counters**

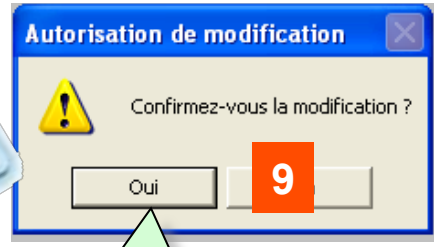


Sélectionner la temporisation **TON**



Saisir le nom de l'instance : **Stop_motor.**

Saisir le nom des variables:
IN : **absence_piece_entree**
PT : **T#10s**
Q : **Tempo_moteur**
ET : **temps**



Confirmer la modification par **Oui.**

Extension de l'application en mode connecté

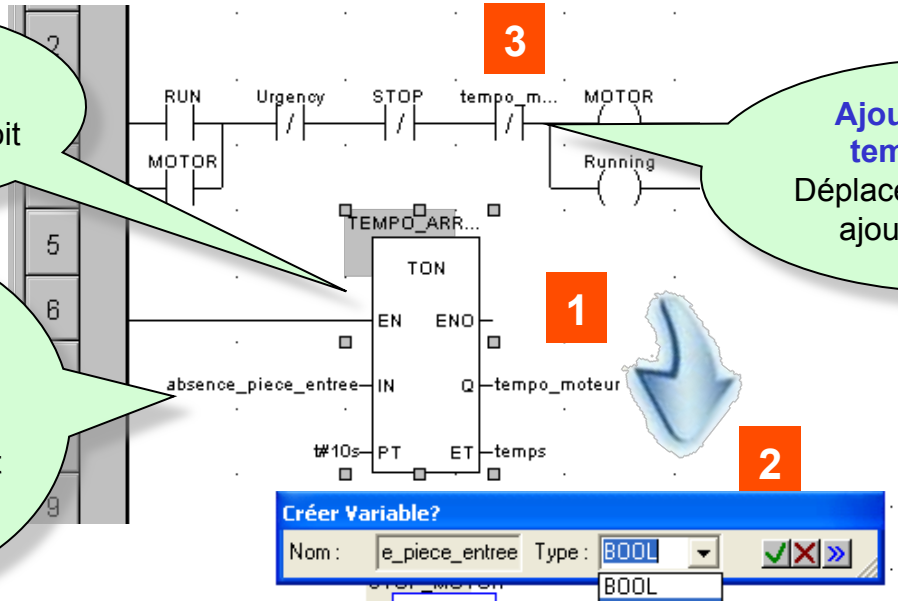
Modification en ligne de la section Convoyeur

Insertion du bloc fonction et renseignement par création des nouvelles variables au fil de l'eau.
Insertion du contact fin de temporisation dans la commande du moteur.
Prise en compte des modifications : analyse des modifications et génération partielle du code.

Déposer la fonction sur l'écran à l'endroit voulu.

Créer les variables dans la base de données. Pour cela effectuer un double clic sur la variable et la déclarer comme dans la phase 1

Ajout du contact tempo_moteur
Déplacer les bobines et ajouter le contact



Rappel de la liste des variables du projet

Nom	Type
Absence_pi...	BOOL
BLOCKER	EBOOL
MOTOR	EBOOL
New_Palett	EBOOL
NUMBER	INT
Optical_sensor	EBOOL
Palett_Full	EBOOL
RUN	EBOOL
RUN1	EBOOL
RUNNING	EBOOL
STOP	EBOOL
STOP1	EBOOL
tempo_moteur	BOOL
temps	TIME
URGENCY	EBOOL

Remarque 1 : Après avoir effectué les modifications, sélectionner le menu **Analyse** pour vérifier que le projet ne comporte pas d'erreur, puis effectuer une génération partielle du code à l'aide du menu **Générer le projet.**

Remarque 2 : Modifier la première ligne de code de la section Palettiseur écrite en langage ST

```
(* simulation du passage de pièces devant le détecteur*)
optical_sensor:= MOTOR and not pallet_full and not absence_piece_entree and %S6;
```

Extension de l'application en mode connecté

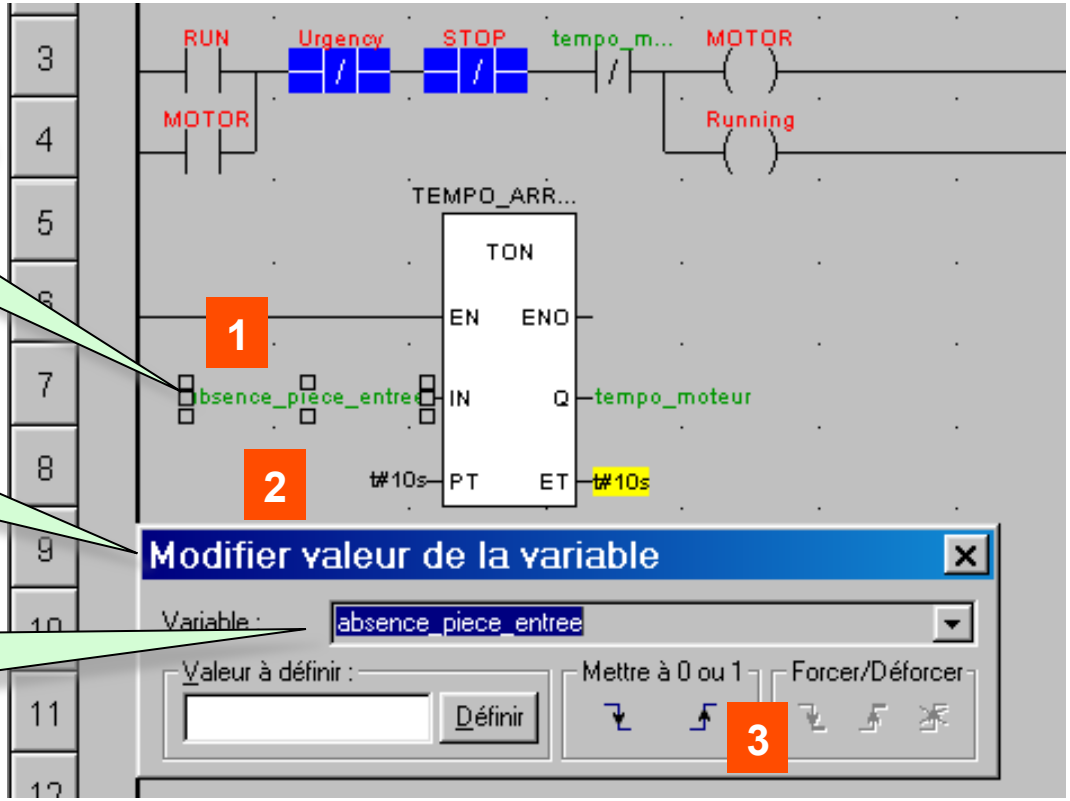
Mise au point

Vérification de l'exécution du temporisateur TON

Sélectionner la variable **absence_pièce_entree.**

Effectuer un clic droit et sélectionner le menu **Modifier valeur**

Mettre à 1 **absence_pièce_entrée**, la temporisation s'écoule et le moteur s'arrête



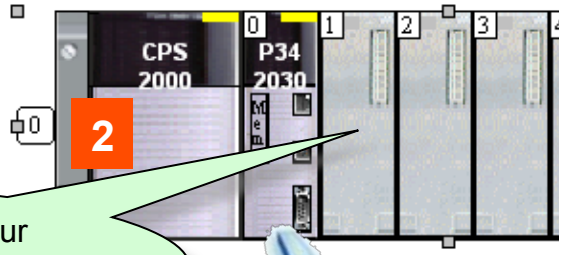
Utilisation de l'automate Modicon M340

Configuration de l'automate (1/2)

Utilisation de l'automate réel : création de la configuration physique.
Nous allons travailler avec un automate réel : se déconnecter du simulateur pour revenir en mode local et fermer éventuellement le simulateur (clic droit sur l'icône vert en bas de l'écran)

1

Effectuer un double clic sur **Bus automate** pour accéder à la **configuration**, le rack est affiché nous allons le configurer.



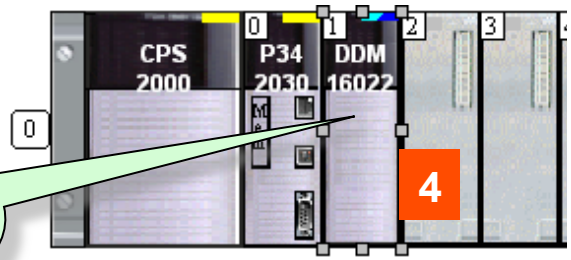
2

Effectuer un **double clic** sur l'emplacement 1 qui recevra le module d'entrées/sorties.

Référence	Description
Station d'E/S locale Modicon M340	
Analogique	
Communication	
Comptage	
TOR	
BMX DAI 1604	Dig 16I 100 to 120 Vac
BMX DDI 1602	Dig 16I 24 Vdc Sink
BMX DDI 3202K	Dig 32I 24 Vdc Sink
BMX DDI 6402K	Dig 64I 24 Vdc Sink
BMX DDM 16022	Dig 8I 24 Vdc 8Q Source Tr
BMX DDM 16025	8I 24 Vdc 8Q Relays

3

Sélectionner dans les références TOR, le **module 8 entrées/ 8 sorties** DDM16022 et valider par OK, le module est **inséré** dans le rack.



4

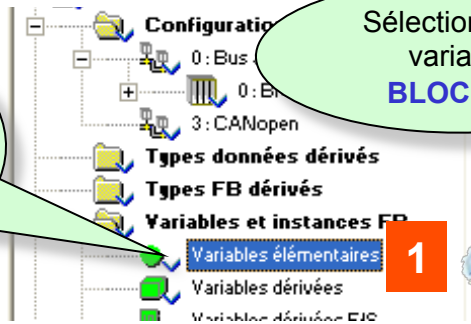
Le **module** est **configuré**.

Utilisation de l'automate Modicon M340

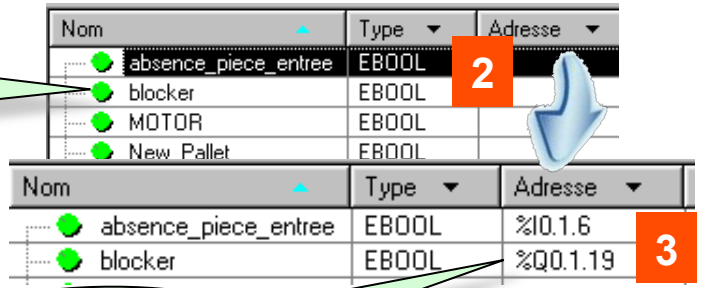
Déclaration des données d'entrées/sorties (1/2)

La configuration de l'automate étant définie, nous allons déclarer les adresses des entrées et des sorties.

Effectuer un double clic sur **Variables élémentaires** pour accéder à l'éditeur de données.



Sélectionner la variable **BLOCKER**



Affecter une adresse réelle à la variable.

Nom	Type	Commentaire	Adresse
RUN	EBOOL	Départ convoyeur	%I0.1.0
RUN1	EBOOL	Départ convoyeur 1	%I0.1.1
STOP	EBOOL	Arrêt convoyeur	%I0.1.2
STOP1	EBOOL	Arrêt convoyeur1	%I0.1.3
URGENCY	EBOOL	Arrêt d'urgence	%I0.1.4
New_Palett	EBOOL	Nouvelle palette	%I0.1.5

Nom	Type	Commentaire	Adresse
MOTOR	EBOOL	Commande Moteur convoyeur	%Q0.1.16
RUNNING	EBOOL	Voyant Marche.Arrêt du moteur	%Q0.1.17
Palett_Full	EBOOL	Palette pleine	%Q0.1.18
Blocker	EBOOL	Bloqueur pièces	%Q0.1.19

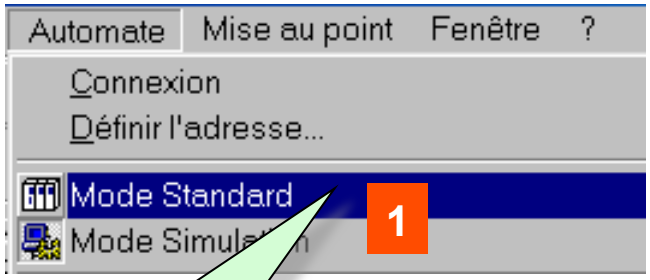
Remarques

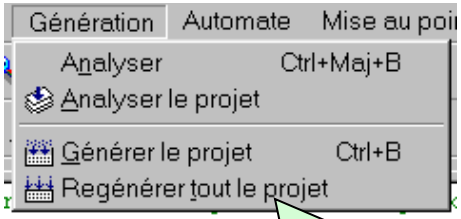
Effectuer la même manipulation pour toutes les variables d'entrées/sorties :
Adressage : I (entrée), Q (Sortie). rack.emplacement.voie.

Utilisation de l'automate Modicon M340

Déclaration des données d'entrées/sorties (2/2)

Si vous disposer d'un automate Modicon M340, vous pouvez à présent tester votre programme. Effectuer l'analyse des modifications et génération du code comme dans les phases précédentes, puis transférer l'application dans l'automate et enfin passer en RUN.

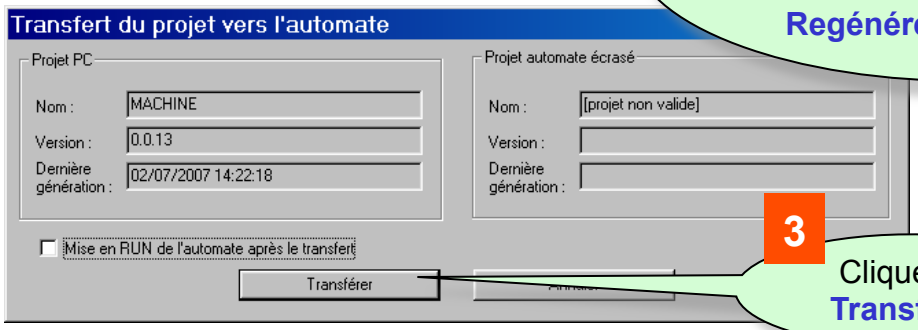




2

Afin de prendre en compte la modification, effectuer une Regénération complète à l'aide du menu **Regénérer tout le projet**

Sélectionner la **cible d'exécution** du programme par le menu **Automate / Mode Standard**



3

Cliquer sur **Transférer.**

Remarque

Il est possible de revenir en mode simulation après avoir configuré un automate réel. Pour cela, procéder comme dans la phase 1 : Programmer en LD.

Sauvegarder le projet.
Fin de la phase 3 : Modifier en ligne

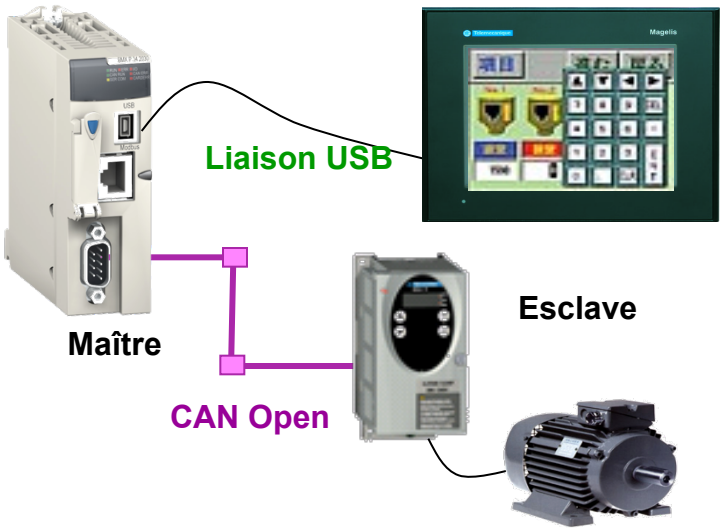
Pour aller plus loin

Intégration dans une architecture d'automatisme

Unity Pro permet de manière très simple de configurer des architectures d'automatisme comprenant un automate Modicon M340, un variateur de vitesse et un terminal opérateur XBT GT

Pilotage d'un variateur de vitesse ATV31 sur bus CAN Open par automate Modicon M340

Affichage de données automate sur un terminal opérateur Magelis XBT GT



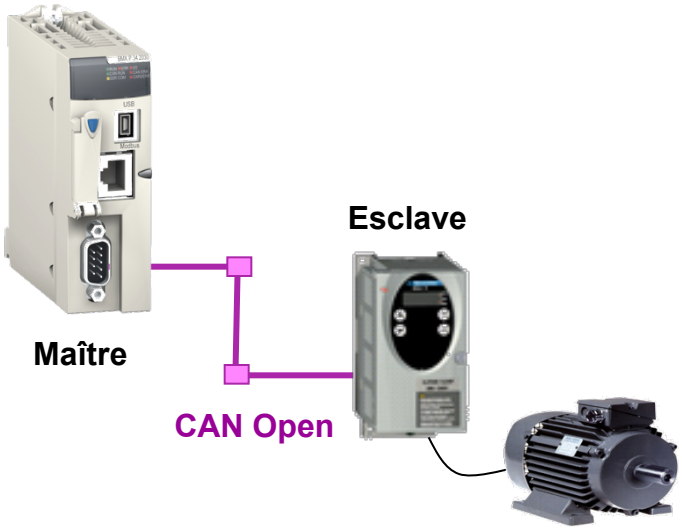
Pour aller plus loin

Pilotage d'un ATV 31 par Modicon M340 (1/6)

Dans notre installation, un variateur ATV31 pilote le moteur.
La communication entre l'automate et l'ATV se fait par le bus CAN Open

Principe de mise en oeuvre

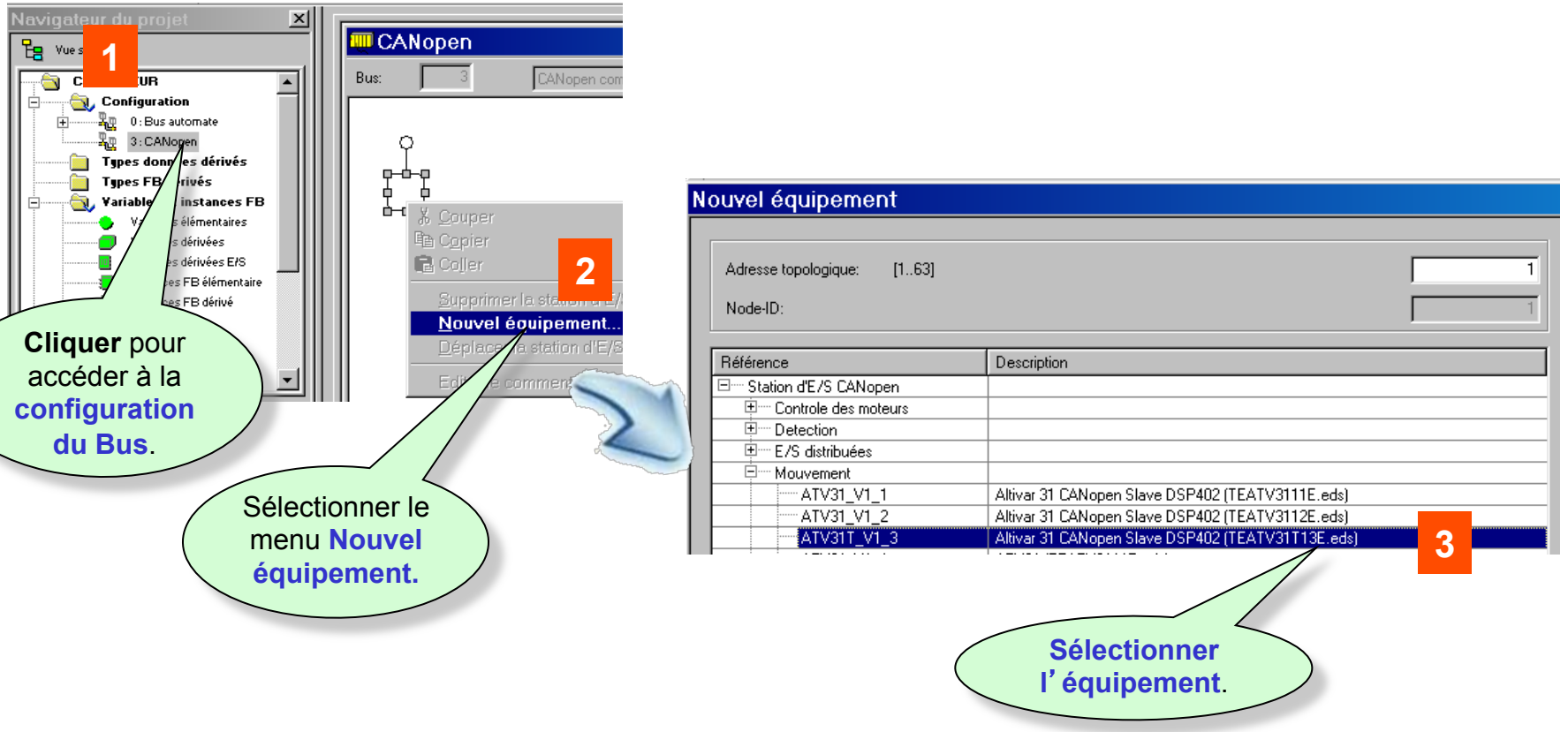
1. Déclarer les équipements sur le bus CANopen dans Unity Pro
   Sélection de l'équipement : ATV31
2. Choix d'un profil d'échange des données entre automate Modicon M340 et ATV31
3. Modification éventuelle du profil des échanges (ajout ou suppression de données échangées)
4. Pré symbolisation des variables
5. Utilisation des données du variateur dans l'application automate



Pour aller plus loin

Pilotage d'un ATV 31 par Modicon M340(2/6)

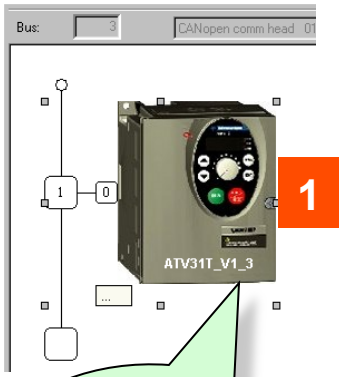
Déclaration de l'équipement ATV sur le bus CAN Open à partir du catalogue.



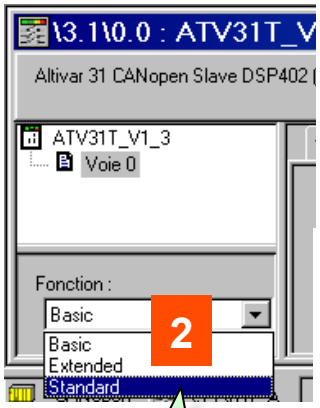
Pour aller plus loin

Pilotage d'un ATV 31 par Modicon M340 (3/6)

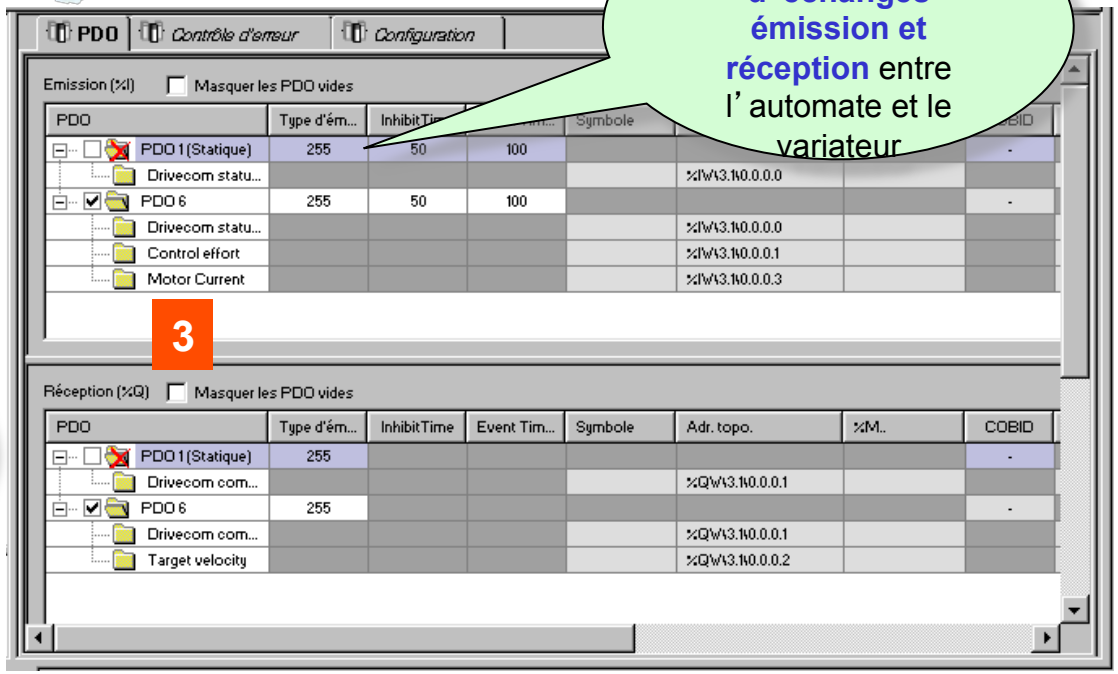
Chaque profil contient la liste prédéfinie de variables échangées entre l'automate et le variateur



Double cliquer pour accéder à la **configuration de l'équipement**.



Sélectionner par exemple le **profil Standard**.



Affichage des **tables d'échanges émission et réception** entre l'automate et le variateur

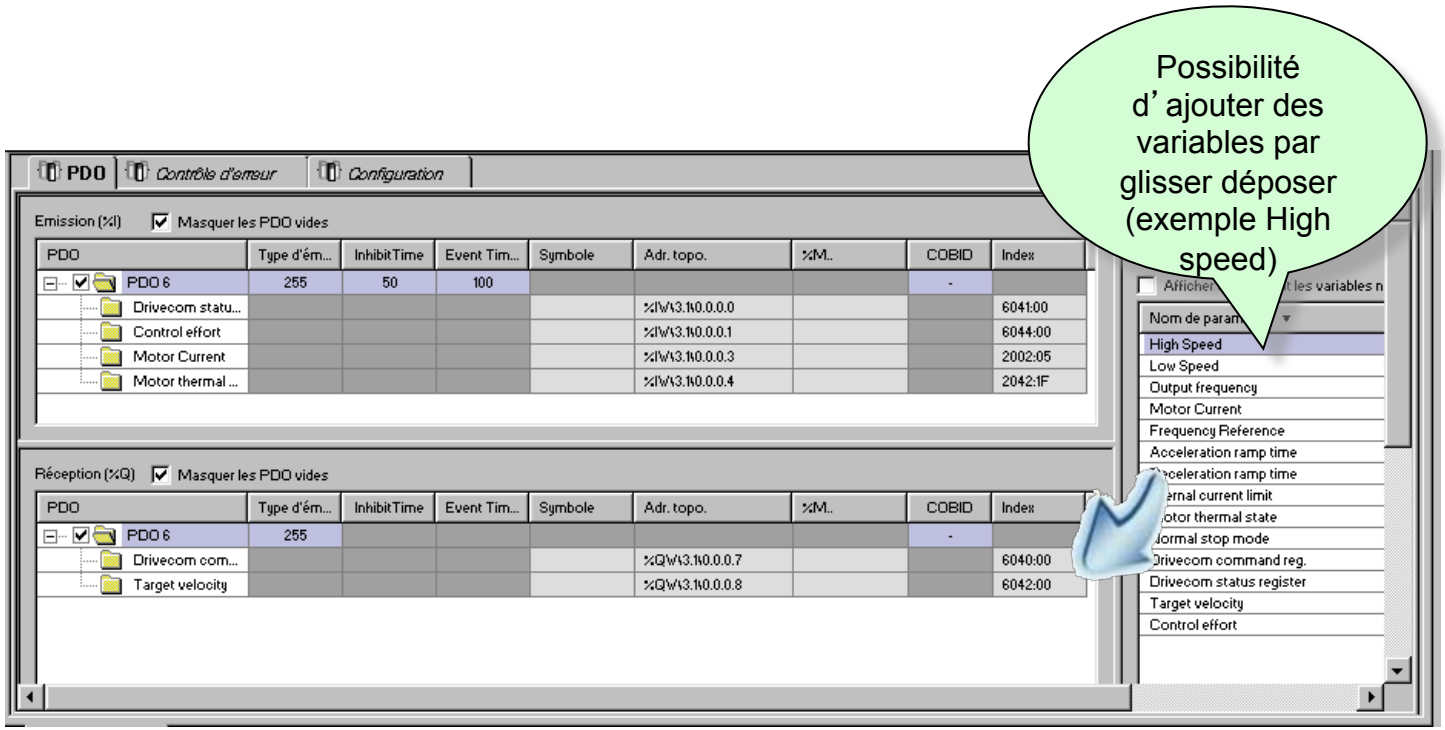
Pour aller plus loin

Pilotage d'un ATV 31 par Modicon M340 (4/6)

Modification éventuelle du profil des échanges : ajout ou suppression de données échangées

Variables prédéfinies par le profil

Variables prédéfinies par le profil



Pour aller plus loin

Pilotage d'un ATV 31 par Modicon M340 (5/6)

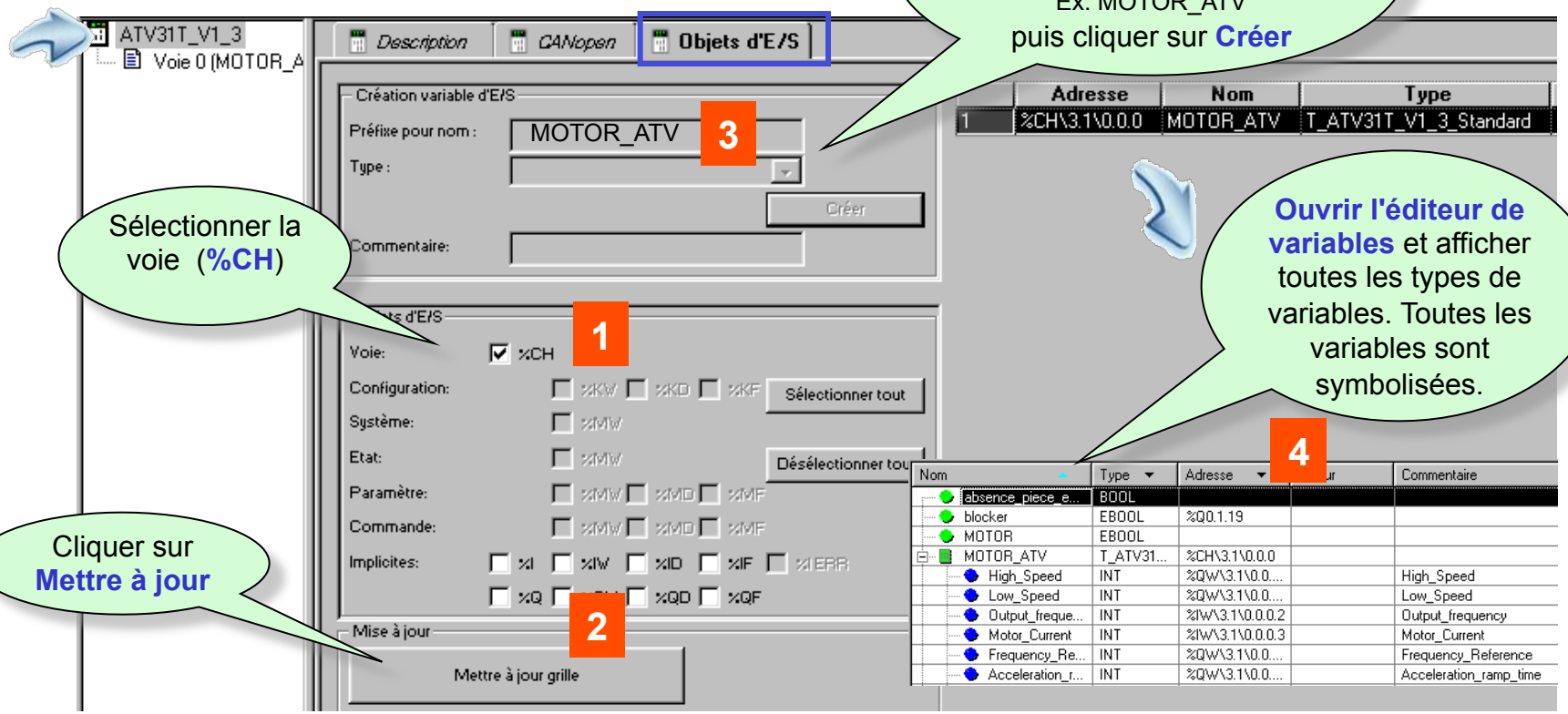
Pré symbolisation des variables

Saisir un préfixe pour le nom de l'ensemble des variables.
Ex: MOTOR_ATV
puis cliquer sur **Créer**

Sélectionner la voie (**%CH**)

Cliquer sur **Mettre à jour**

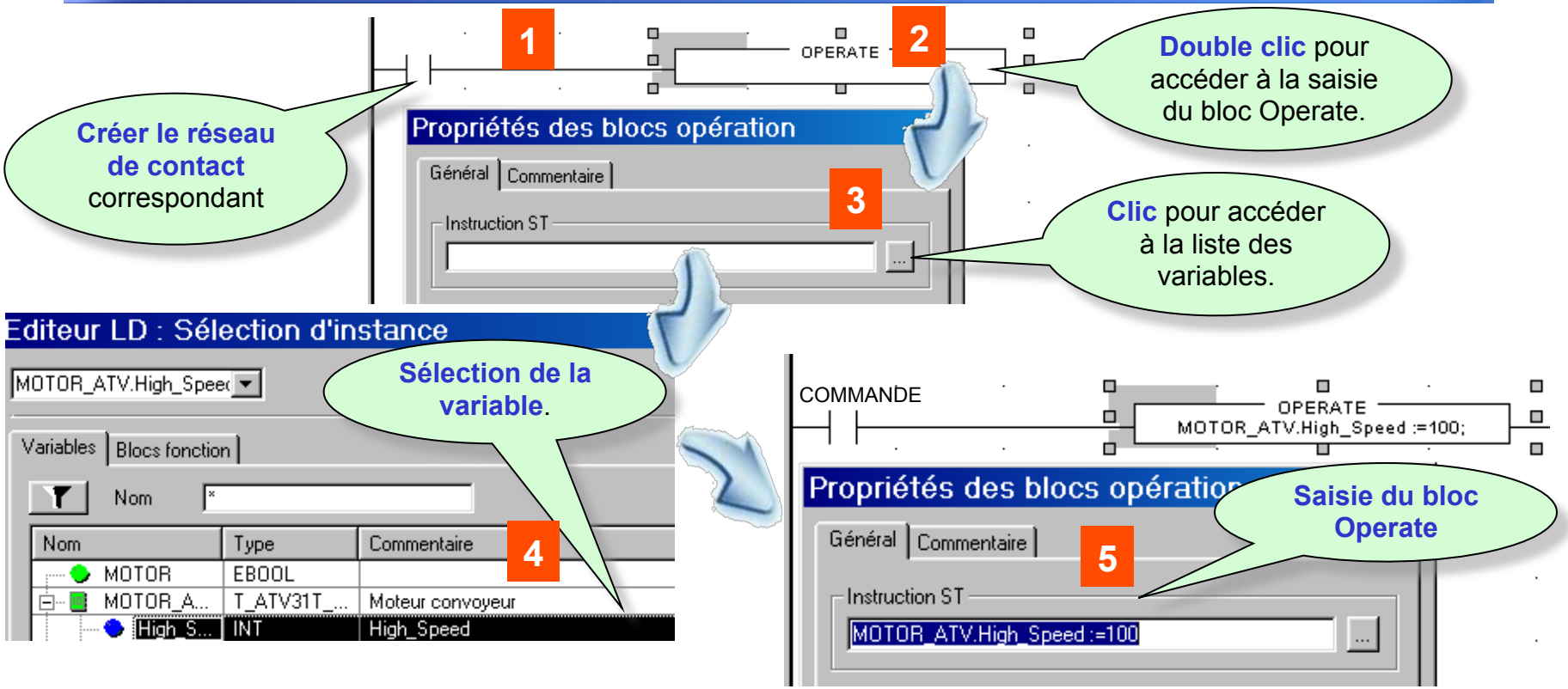
Ouvrir l'éditeur de variables et afficher toutes les types de variables. Toutes les variables sont symbolisées.



Pour aller plus loin

Pilotage d'un ATV 31 par Modicon M340 (6/6)

Utilisation des variables échangées dans l'application automate
exemple : modification d'une consigne de vitesse sur commande



Pour aller plus loin

Affichage de données sur terminal XBTG (1/6)

Les données définies dans Unity Pro sont réutilisables dans Vijeo Designer sans double saisie.

Principe de mise en oeuvre

Localisation des variables à afficher sur le terminal sur des adresses (dans Unity Pro)
Validation et sauvegarde des modifications

↓

Création du projet dans Vijeo Designer

↓

Définition de la connexion terminal/automate

↓

Création des liens avec l'application Unity Pro et sélection des variables

↓

Utilisation des variables dans l'écran de dialogue

Pour aller plus loin

Affichage de données sur terminal XBTG (2/6)

Localisation des variables à afficher sur le terminal sur des adresses (dans Unity Pro).
Validation et sauvegarde des modifications.

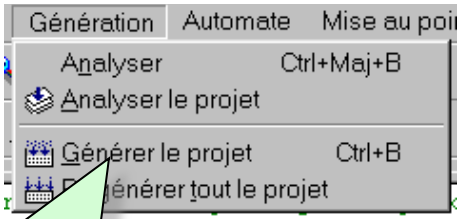
Accéder à l'**éditeur de variable**

1

Nom	Type	Adresse
number	INT	%MW1
absenc...	BOOL	%M1

2

Affecter une adresse aux variables qui seront utilisées dans Vijeo Designer



3

Afin de prendre en compte la modification, effectuer une génération à l'aide du menu **Générer le projet** puis **Enregistrer le projet**

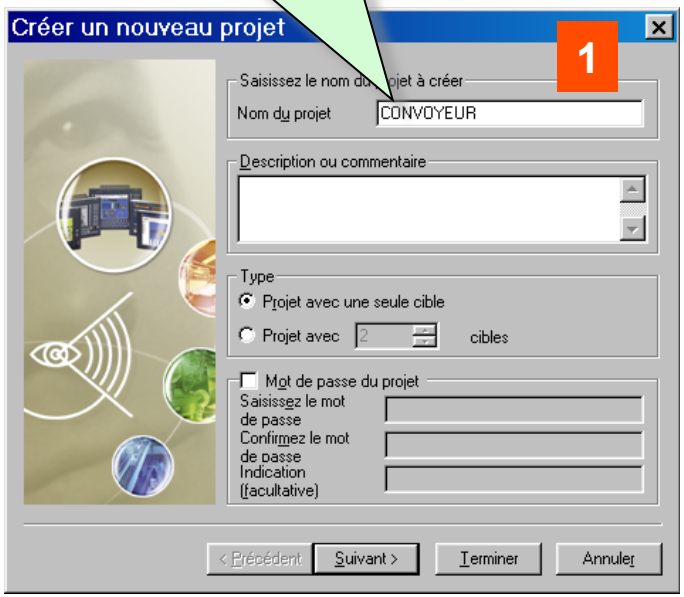
Remarque : Il est possible de trier les variables grâce à une personnalisation des colonnes

Pour aller plus loin

Affichage de données sur terminal XBTG (3/6)

Création du projet dans Vijeo Designer

Sélectionner le menu **Fichier/ Nouveau projet** et le nommer CONVOYEUR

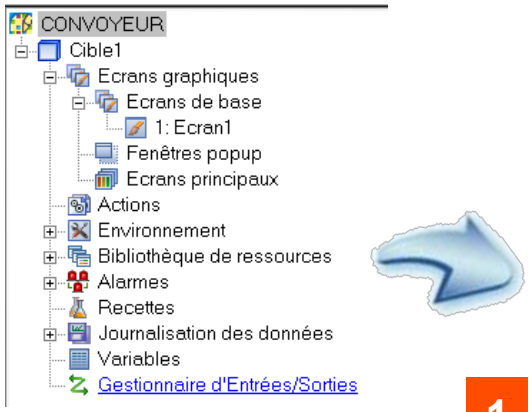


Nommer le terminal
Sélectionner le modèle
exemple 7,5 pouces

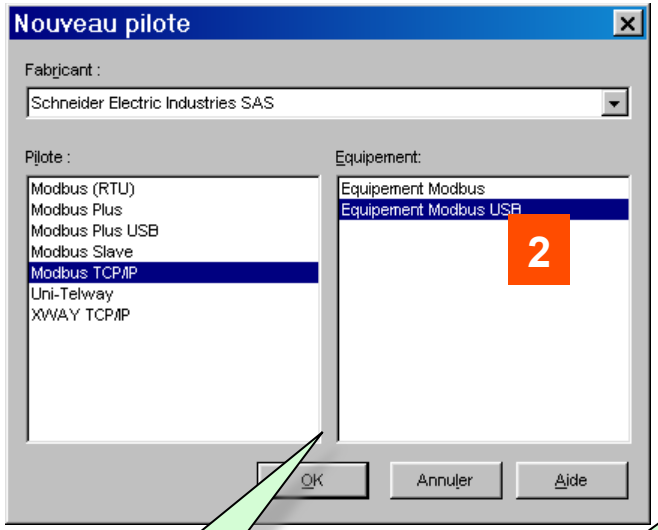
Pour aller plus loin

Affichage de données sur terminal XBTG (4/6)

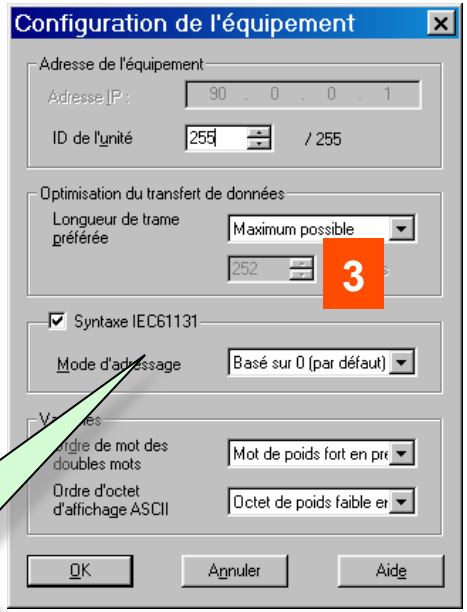
Définition de la connexion Automate/Terminal.
Utilisation du port USB de l'automate Modicon M340 : Modbus TCP/IP



Sélectionner le menu **Nouveau pilote** par un clic droit



Sélectionner le pilote

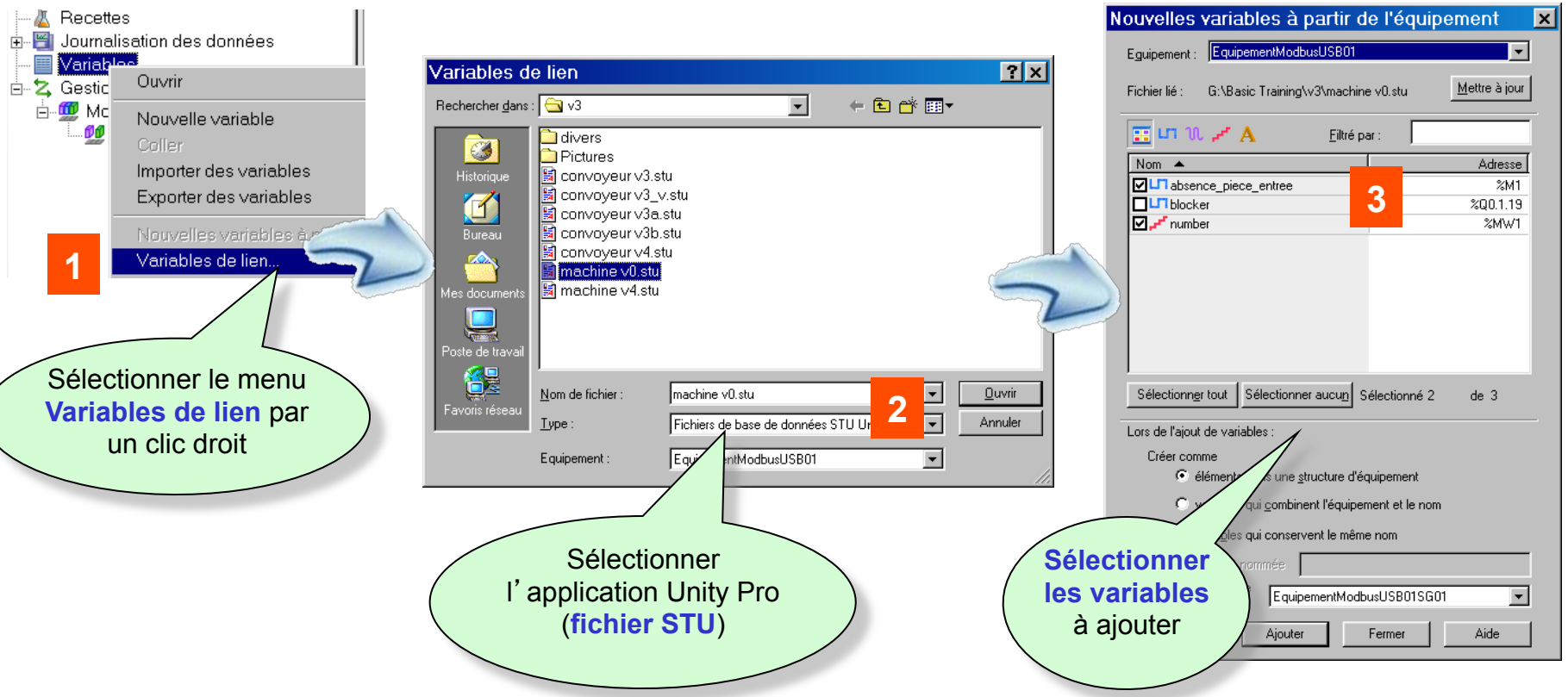


Configurer l'équipement : adresse de l'unité et syntaxe IEC

Pour aller plus loin

Affichage de données sur terminal XBTG (5/6)

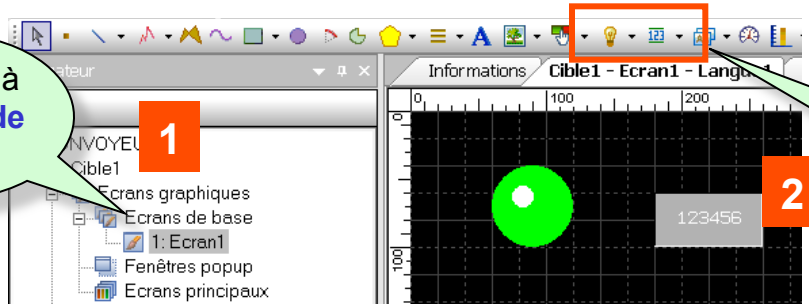
Création des liens avec l'application Unity Pro et sélection des variables :
l'application Vijeo Designer est associée à l'application automate



Pour aller plus loin

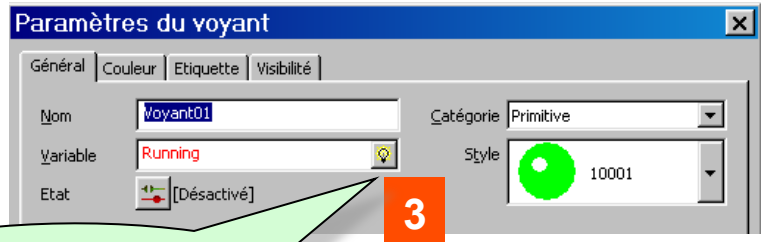
Affichage de données sur terminal XBTG (6/6)

Utilisation des variables dans l'écran de dialogue : affichage absence pièce et compteur de pièces sur le terminal opérateur

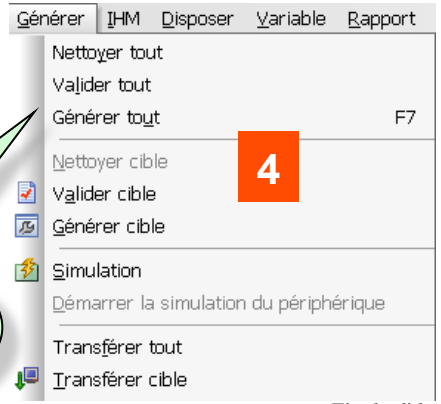


Accéder à l'**écran de saisie**

Dessiner les objets à l'aide des icônes



Affecter une variable à chaque objet en la **sélectionnant dans la liste**



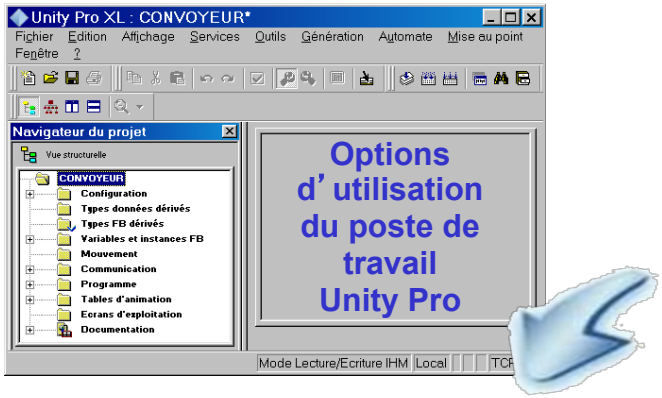
Générer le projet et le **transférer dans le terminal**

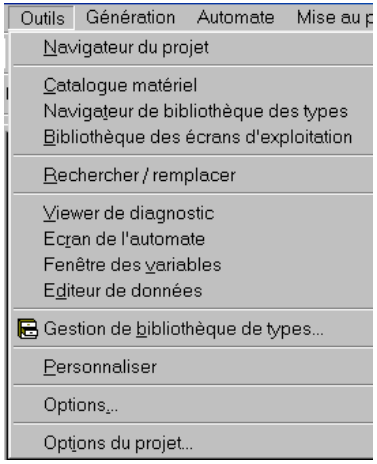
Fin du didacticiel

Votre premier projet Unity Pro

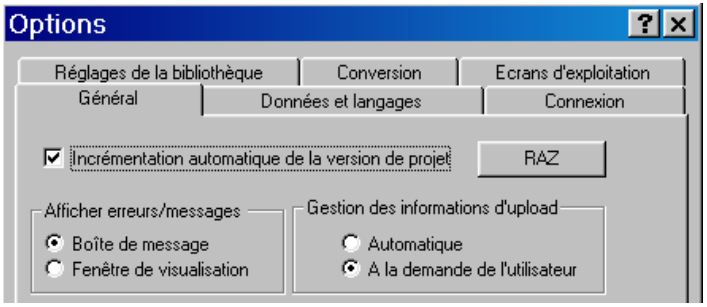
Personnalisation de Unity Pro (1/3)

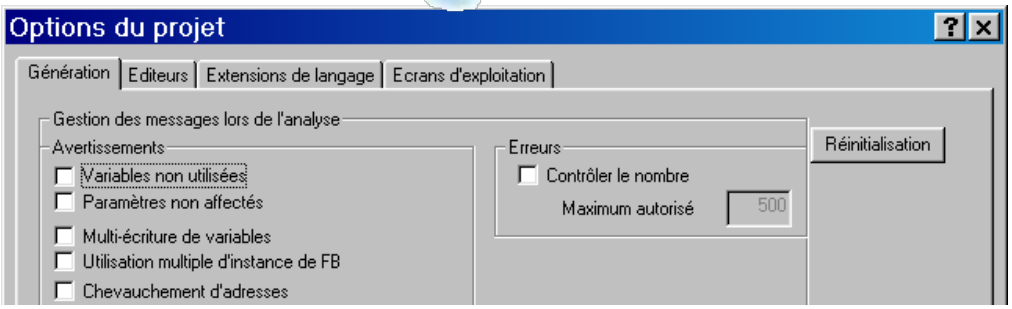
Unity permet de définir différentes options de travail :
- les options relatives à l'utilisation de Unity Pro (menu Outils/Options)
- les options relatives au projet (menu Outils/Options du projet)





Options embarquées dans le projet





Votre premier projet Unity Pro

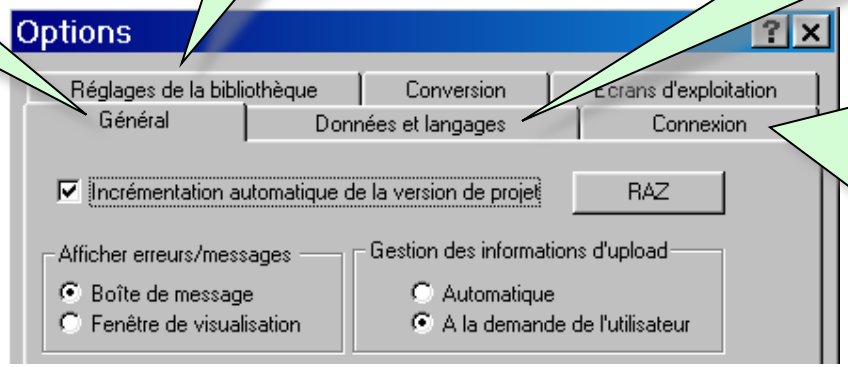
Personnalisation de Unity Pro (2/3)

Unity Pro permet de définir différentes options du poste de travail.
Voici quelques exemples.

- Incrémentation automatique des versions projet.
- Modification des différents répertoires.

- Chemin de la bibliothèque des blocs fonctions prédéfinis

- Affectation automatique d'une variable à tout objet graphique
- Paramétrage éditeur Ladder



- Mode programmation : Le PC est connecté, par défaut, en mode programmation.

- Mode surveillance : Le PC est connecté en mode surveillance (modification de variable uniquement).

Votre premier projet Unity Pro

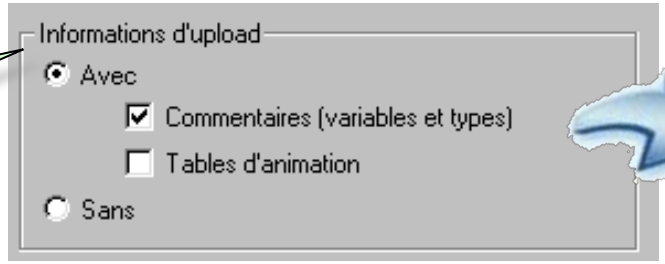
Personnalisation de Unity (3/3)

Unity Pro permet de définir différentes options qui seront embarquées dans le projet.
Elles ont donc conservées sur n'importe quel poste de travail Unity Pro.
Voici 3 exemples.

Bobines alignées à droite
(Onglet Général)

Lorsque cette case est cochée, les bobines sont automatiquement placées sur la barre d'alimentation droite.

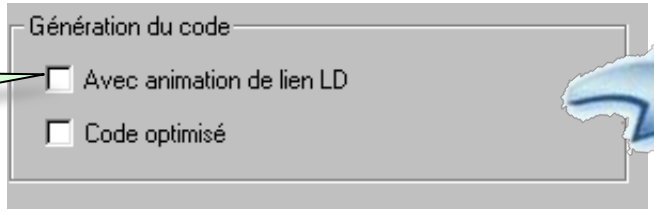
Information d'upload
(Onglet Général)



Les informations d'upload sont constituées du code, des symboles et commentaires des variables, des tables d'animation, ...
Sauvegardées lors d'un enregistrement du projet, elles peuvent être ou ne pas être incluses avec la mémoire automate.

Ces informations permettent de travailler avec un poste ne contenant pas le projet.

Animation des liens
(Onglet Général)



Permet d'inclure ou pas dans le code généré les informations qui assurent l'animation des liens en langage à contacts.