REPUPLIQUE ALGERIENNE DEMOCRATIQUE ET POLPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
ECOLE NATIONALE SUPERIEURE POLYTECHNIQUE

Département d'Automatique

Projet de fin d'études
En vue de l'obtention du diplôme d'ingénieur d'état en Automatique
Thème:

Contribution à la commande et à la Supervision Des Stations De Pompage d'un oléoduc

Etudié par :
Mr. MENASRI Riad
Mr. ABBAS Mohand Said

Proposé et dirigé par :
Mr. A. BERKOUK
Mr.M.S.MAARADJI

Juin 2010

Ecole Nationale Polytechnique, 10 Av. Hassan Badi, El Harrach, Alger, Algérie

REMERCIEMENTS

Nous tenons à exprimer nos vifs remerciements à notre promoteur Pr.BERKOUK de l'Ecole Nationale Polytechnique pour nous avoir encadrés durant notre projet de fin d'études et nous conseillé tout le long de notre travail.

Nous remercions également notre co-promoteur Mr MAARAJI Samir, de SHNEIDER ELECTRIC, pour son encadrement, ces conseils, et sa confiance au sein de l'entreprise.

Nous remercions chaleureusement les membres du jury pour l'honneur qu'ils nous ont fait en acceptent d'évaluer notre projet.

Nos sincères remerciements aux ingénieurs de Schneider qui nous ont conseillé et éclairé sur notre travail tout le long de notre projet. Nous souhaitons aussi remercier tous les enseignants de l'Ecole Nationale Polytechnique d'Alger, et en particulier, Nos professeurs d'Automatique qui nous ont encadrés auparavant et tous nos enseignants pour les connaissances qu'ils nous ont transmis, leur disponibilité et leurs efforts.

Nous remercions tout le personnel de l'école, de Schneider et tous les élèves de génie électrique.

Que tous ceux qui ont contribué de prés ou de loin à la réalisation de ce modeste travail trouvent ici l'expression de notre sincère gratitude.

Dédicaces

Je dédie ce modeste travail à mes chers parents.
A mon frère DJAFFAR et à ma sœur LIKHA.
A ma grand-mère YEMMA BAYA, qui m'a toujours soutenu.
A toute ma famille et particulièrement pour DDA LHADI et NNA MOUNIA et leur famille et à mon grand cousin DDA LYAZID et à FARIDA.
A KHALI A/NACER et à NABILA.
A mes amis AHMED, BOUALEM (bou3bou3), MOOOOH, MASSI, AMAR, RYM et LILA qui ont toujours été là pour moi.

ABBAS MOHAND SAID

Je dédie ce modeste travail à mes très chers parents

A mon frère Lounes, à mes deux sœurs et leurs maris, à une personne qui m'est chère à mon cœur.

A tous mes amis et à touts ceux je connais à l'Ecole Nationale Polytechnique et surtout à Nehal Massinissa.

Riad

RESUME

Le travail présent dans ce mémoire se base essentiellement sur l'utilisation des automates programmables SHNEIDER. Notre travail est la programmation de la gestion de deux stations de pompages pétrole à l'aide de logiciel Unity Pro par l'utilisation des DFB et la supervision à l'aide de logiciel Vijeo Citect. Sa gestion est assurée par un automate Quantum.

Mots clés : automates programmables SHNEIDER, la gestion d'une station de pompage, logiciel Unity Pro, DFB, logiciel Vijeo Citect, automate Quantum.

ABSTRACT

The work presented in this memory is based primarily on the use of the programmable automats SCHNEIDER. Our job is programming for the management of two oil pumping station with Unity Pro software through the use of DFB and supervision with Vijeo software Citect. Its management is provided by a PLC quantum.

Key word: Programmable logic controller SCHNEIDER, oil pumping station, Unity Pro software, DFB, Vijeo software Citect, Quantum.

ملخص:

العمل المنجز في هذه المذكرة يتمحور اساسا حول استخدام برمجة شنيدر لادارة محطة لضخ البترول و ذلك باستخدام برنامج "يونيتي برو". تصميم شاشة اشراف من خلالها نستطيع مراقبة جميع اجهزة المحطة باستخدام برنامج فيجيوسايتيكت.

تسييرها يتم عن طريق الالي المبرمج كانطوم.

الكلمات المفتاحية :

مسير صناعي مبرمج شنيدر, محطة ضخ البترول, برنامج يونيتي برو, برنامج فيجيوسايتيكت.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

La compétitivité des entreprises impose un recours à la fois fréquent et intensif à des technologies de production avancées. La productique et la complexité des opérations à exécuter, conduisent à la mise en œuvre des dispositifs et systèmes pour l'automatisation des ateliers de fabrication ou de production.

L'automate programmable industriel A.P.I est aujourd'hui le constituant le plus répandu pour réaliser des automatismes. On le trouve pratiquement dans tous les secteurs de l'industrie car il répond à des besoins d'adaptation et de flexibilité pour un grand nombre d'opérations. Cette émergence est due en grande partie, à la puissance de son environnement de développement et aux larges possibilités d'interconnexions.

Le domaine des hydrocarbures parmi d'autre, est témoin de cette révolution, et la majorité des filiales de la société SONATRACH sont équipées d'armoires d'automates et de pupitre de supervision, on les trouve dans les stations de pompage, stations de traitement….

Dans le but d'amélioration des applications (programmes) de gestion des stations de pompage Schneider Electric nous a proposé la réalisation des DFB (blocs fonctions dérivées) permettent de structurer et d'optimiser ces applications. Dans ce cadre : des DFB ainsi que des programmes vont être mises en place englobant l'essentiel des systèmes que nous puissions trouver dans une station de pompage réelle.

Parallèlement à la programmation, la création des vues de supervision joue aussi un rôle important dans la gestion des stations de pompage. Pour cela : on crée des objets des différents équipements qui se trouve au sein des stations tels que les vannes, les pompes, les bacs,…et on rassemblant ces objets, on construit des synoptiques qui décrivent le fonctionnement de la station. Ces derniers nous permettent de visualiser l'état des différents organes (en marche, à l'arrêt, présence de défaut,…) et aussi d'envoyer de commandes (ouverture, fermeture, démarrage,…).

Plan de travail

On a divisé le travail en cinq chapitres :

Le premier chapitre donne un aperçu général sur les automates programmables industriels (différents constituants, architecture…)

Le deuxième chapitre présente le logiciel de programmation Unity Pro : une présentation du logiciel (versions, fonctions, langages de programmations….) suivie d'une description de la méthode de configuration et on termine par un exemple de mise en œuvre d'une application.

Le troisième chapitre présente le logiciel de supervision Vijeo Citect : une présentation (fonctions, éléments d'un projet….) suivie de la méthode de création d'un projet avec toutes les configurations nécessaire et on termine par un exemple d'application.

Le quatrième chapitre décrit le fonctionnement de l'oléoduc OZ1 : un aperçu sur les différentes méthodes du transport des hydrocarbures suivis de la description de l'oléoduc OZ1 (différentes stations de pompages, mode d'exploitation…) et enfin une description générale du fonctionnement des stations de pompages.

Le cinquième chapitre est consacré à l'application à savoir la création des programmes de gestion de la station ainsi que les objets qui représente les différents organes et aussi les synoptiques.

On termine par une conclusion générale.

Chapitre I

Généralités sur les A.P.I

Chapitre I : Généralités sur les automates programmables industriels

I.1. Introduction [5]

Les progrès des automatismes industriels notamment dans le domaine des automates ont permis aux industriels d'augmenter leur productivité et de réduire les coûts. La généralisation de l'électronique et la souplesse des logiciels autorisent une meilleure exploitation et offrent de nouveaux outils de maintenance.

Les exigences des clients ont aussi considérablement évolué, et la concurrence avec, c'est dans ce sens là que SCHNEIDER ELECTRIC innove dans les automatismes et proposes des gammes très variées et très complète.

I.2. Organisation d'un système automatisé

Un système de production a pour but d'apporter une valeur ajoutée à de la matière d'œuvre dans un contexte donné. Quand ce système est automatisé, on peut généralement le décomposé deux parties :

- Une partie opérative dont les actionneurs agissent sur le processus automatisé.
- Une partie commande qui coordonne les différentes actions de la partie opérative et qui communique avec le ou les opérateurs.

C'est dans la partie commande que l'on retrouvera les Automates Programmables Industriels.

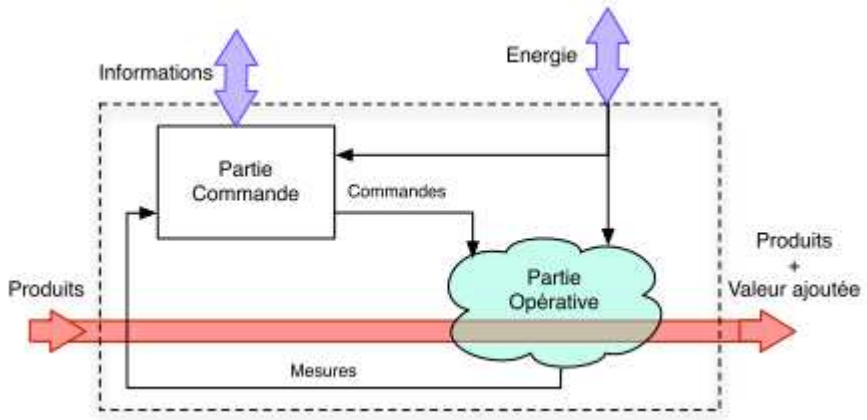


Figure I.1 : Strucute d'un système de production

I.3. Définition d'un automate programmable industriel

Un Automate Programmable Industriel (API) est une machine électronique programmable par un personnel non informaticien et destiné à piloter en ambiance industrielle

et en temps réel des procédés ou parties opératives. Un automate programmable est adaptable à un maximum d'application, d'un point de vue traitement, composants et langage. Selon l'application a développé, on trouve des automates compacts pour les petite applications et modulaire pour des applications assez grande et complexe.

I.4. Architecture générale d'un API [4]

En général un automate programmable se constitue essentiellement d'une unité centrale, un module d'entrées/sorties, un module d'alimentation, un module de stockage et de liaisons et des auxiliaires.

Cette architecture est représentée dans la figure suivante :

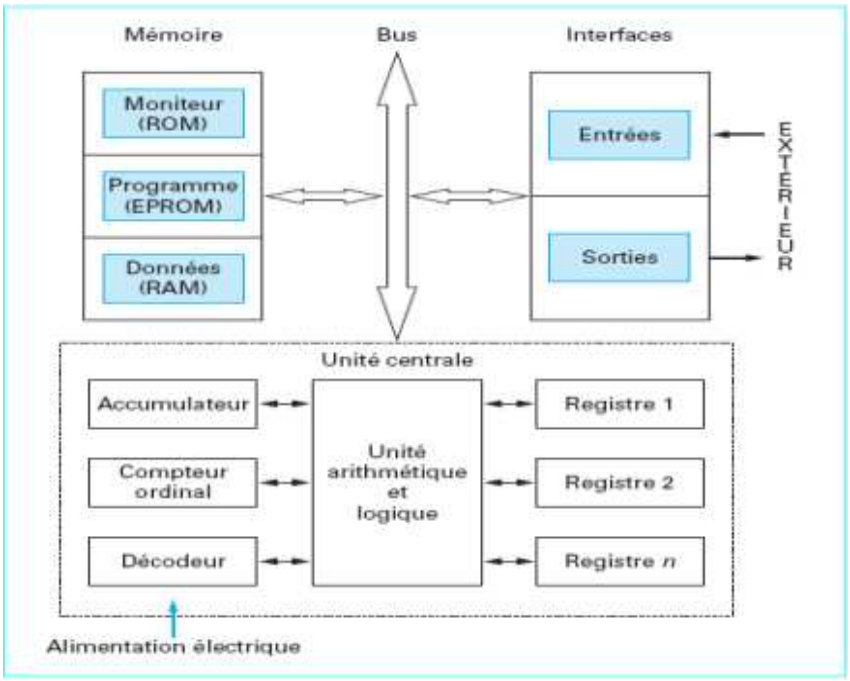


Figure I.2 : Architecture d'un API

I.4.1. Module d'alimentation

Ce module permet l'alimentation en tension continue nécessaire au bon fonctionnement de l'automate programmable ainsi que le circuit de charge. Il convertit la tension du réseau (AC 220) en tension de service (DC 24V, 15V ou 5V). Ce module doit posséder de bonnes performances face aux microcoupures du réseau, ainsi qu'un transformateur d'isolement pour lutter contre les perturbations du même réseau.

I.4.2. Unité centrale

L'unité centrale (CPU) est l'élément le plus important dans l'automate programmable, elle peut être considérée comme le cerveau du système. Elle est constituée de deux composants principaux:

- Le Processeur ;
- Les mémoires.

I.4.2.1. Le processeur

La principale fonction du processeur est de commander et gouverner les différentes activités du système. Il effectue cette tache en interprétant et en exécutant un ensemble de programmes système. Ces derniers forment un groupe de programmes superviseurs stockés de façon permanente dans le processeur. Grâce à ces programmes superviseurs le processeur peut ainsi exécuter toutes ses taches de contrôle, ainsi que divers fonctions domestiques.

Ces programmes appelés aussi « le pouvoir exécutif » assurent la communication entre l'API et l'utilisateur par le biais de dispositifs de programmation. Ils supportent aussi d'autres périphériques de communication tels que la surveillance des appareils de terrain, la lecture des données de diagnostic, l'alimentation, les modules d'entrées/sortie, les mémoires, et la communication avec les interfaces opérateurs.

I.4.2.2. Les mémoires

Tout système bâti autour d'un processeur possède un ou plusieurs types de mémoires. La mémoire système dans un API est composée de deux majeures parties :

- La mémoire exécutive : assure le stockage des programmes superviseurs.
- La mémoire d'application : est une zone de stockage dédiée aux programmes d'instructions utilisateur.

▪Les exigences de stockage et de récupération pour les programmes superviseurs et les programmes d'application ne sont pas les mêmes, par conséquent ils ne sont pas toujours stockés dans le même type mémoire. Ainsi on aura l'organisation suivante :

- ROM ou PROM : Ce sont des mémoires mortes dont l'utilisateur ne peut que lire le contenu (ROM) et éventuellement les programmer à l'aide d'outils spéciaux (PROM). On y retrouve dans notre cas les programmes superviseurs.

- EPROM : C'est une mémoire reprogrammable qui permet de stocker les programmes mis au point et utilisables.
- RAM : C'est une mémoire vive (volatile) secourue en général par une batterie, elle stocke les données système lors du fonctionnement.

I.4.3. Module d'entrées sorties (E/S)

Les modules d'E/S assurent le rôle d'interface de la partie commande, ils se situent entre la CPU et le processus, pour ce faire ils doivent :

- Regrouper les variables de même nature pour diminuer la complexité et le coût ;
- Assurer le dialogue avec la CPU ;
- Traduire les signaux industriels en information API et inversement.

▪Plusieurs types de modules sont disponibles sur les marchés comme :

- Modules d'E/S tout ou rien (TOR) : Ces modules traitent une information qui ne peut prendre que deux états (vrai ou faux, 0 ou 1), ils constituent l'interface entre l'API et les différents capteurs et pré-actionneurs présents.
- Modules d'E/S analogique : Dans ce cas, le signal traité est analogique et prend des valeurs comprises dans une plage bien déterminée. Ces modules sont munis de convertisseur analogique/numérique pour les entrées et respectivement de convertisseur numérique/analogique.
- Module spécialisés : l'information traitée est contenue dans des mots codés sous frome binaire ou bien hexadécimale. C'est le type d'information délivrée par un ordinateur ou un module intelligent.

I.4.4. Modules de communication

Les modules de communication comprennent les consoles et les boitiers tests :

- Les consoles : Les consoles permettent la programmation, le paramétrage et les relevés d'informations, ils peuvent également afficher le résultat de l'autotest comprenant l'état des modules d'entrées et de sorties, l'état de la mémoire, de la batterie, etc. Ils sont équipés (pour la plupart) d'un écran à cristaux liquides.
- Les boitiers tests : Les boîtiers de tests quand a eux sont destinés aux personnels d'entretien ; ils permettent de visualiser le programme ou les valeurs des paramètres

tes que l'affichage de la ligne de programme à contrôler, la visualisation de l'état des entrées et des sorties…).

I.4.5. Auxiliaires

Il s'agit principalement :

- D'un ventilateur : qui est en général indispensable dans les châssis comportant de nombreux modules ou dans le cas où la température ambiante est susceptible de devenir assez élevée (plus de 40 °C) ;
- Du support mécanique : Il peut s'agir d'un rack (structure métallique accueillant des cartes avec généralement un raccordement arrière), l'automate se présentant alors sous forme d'un ensemble de cartes, d'une armoire, d'une grille et des fixations correspondantes ;
- D'indicateurs d'état concernant la présence de tension, l'exécution du programme (mode RUN), la charge de la batterie, le bon fonctionnement des coupleurs…

I.5. Langages de programmation

La norme IEC 1131-3 (Commission Électrotechnique Internationale) définit cinq langages qui peuvent être utilisés pour la programmation des automates programmables industriels. Ces langages peuvent être divisés en deux catégories :

- Langages graphiques :
  - SFC « Sequential Funiculite Chart » ou GRAFCET ;
  - LD « Ladder Diagram » ou schéma à relais ;
  - FBD « Function Block Diagram » ou schéma par bloc.
- Langages textuels :
  - ST « structured text » ou texte structuré ;
  - IL « Instruction List » ou liste d'instructions.

Remarque : La gamme d'automate Schneider est présentée en [**annexe : A.1**].

I.6. Conclusion

Cette section nous a permis de faire une étude générale de la technologie des automates programmables industriels et de présenter les différents langages de programmation normalisés. Pour des détails sur la gamme des automates Schneider Electric et les différents modules la section A de l'annexe y revient.

Chapitre II

Logiciel Unity Pro

Chapitre II : Logiciel de programmation Unity Pro

II.1. Présentation du logiciel Unity Pro [1]

Unity Pro est un logiciel de programmation qui prend en charge les plates-formes matérielles : Modicon M340, Premium, Atrium et Quantum.

▪Il propose les langages suivants pour la création des programmes utilisateurs :

- Langage en blocs fonctionnels FBD ;
- Langage à contacts LD ;
- Liste d'instructions IL ;
- Littéral structuré ST ;
- Diagramme fonctionnel en séquence SFC.

▪Tous ces langages peuvent être utilisés ensemble dans le même projet et ils sont conformes à la norme CEI 61131-3.

▪Unity pro est proposé en 5 versions appelées prologiciel, qui sont :

- Unity Pro S;
- Unity Pro M;
- Unity Pro L;
- Unity Pro XL;
- Unity Pro XLS;
- Unity Developers Edition (UDE).

▪Un programme Unity Pro peut se composer :

- D'une tâche maître (MAST) ;
- D'une tâche rapide (FAST) ;
- D'une à quatre tâches auxiliaires (non disponibles pour Modicon M340) ;
- De sections auxquelles est affectée l'une des tâches définies ;
- De sections dédiées au traitement des événements temporisés (Timerx) ;
- De sections de traitement d'événements issus de modules d'entrées/sorties(EVTx) ;
- De sections de sous-programme (SR).

II.2. Interface utilisateur

L'interface utilisateur se compose de plusieurs fenêtres et barres d'outils pouvant être positionnées librement comme elle est représentée ci dessous :

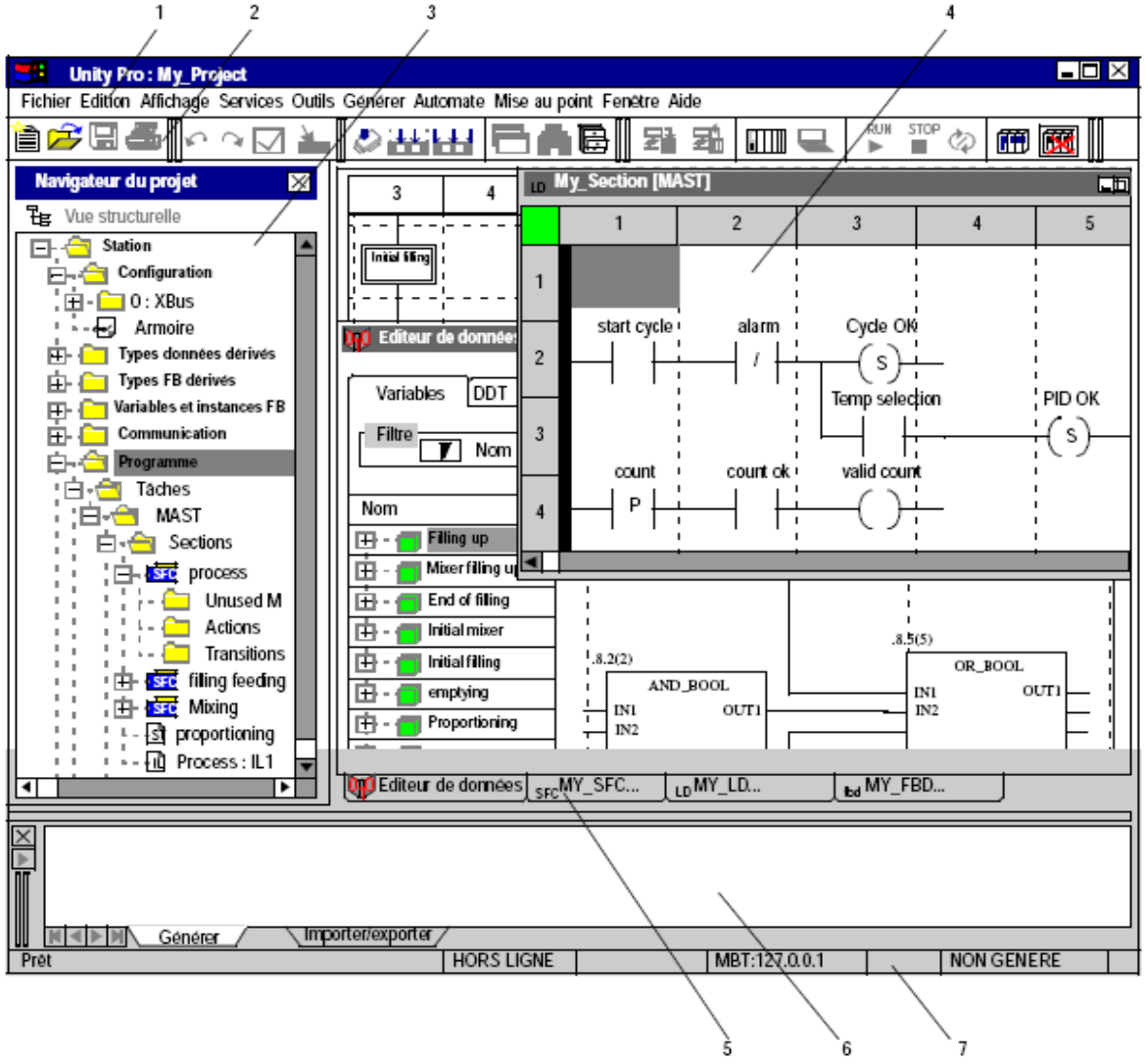


Figure II.1 : Interface utilisateur

1	Barre de menus
2	Barre d'outils
3	Navigateur du projet
4	Fenêtre de l'éditeur (éditeurs de langages, éditeur de données, etc.)
5	Onglets d'accès direct aux fenêtres de l'éditeur
6	Fenêtre d'information (donne des informations sur les erreurs survenues, le suivi des signaux, les fonctions d'importation, etc.)
7	Ligne d'état

Tableau II.1 : Différents éléments de l'interface utilisateur

II.3. Le navigateur du projet

La figure qui suit montre le navigateur de projet sous la vue structurelle :

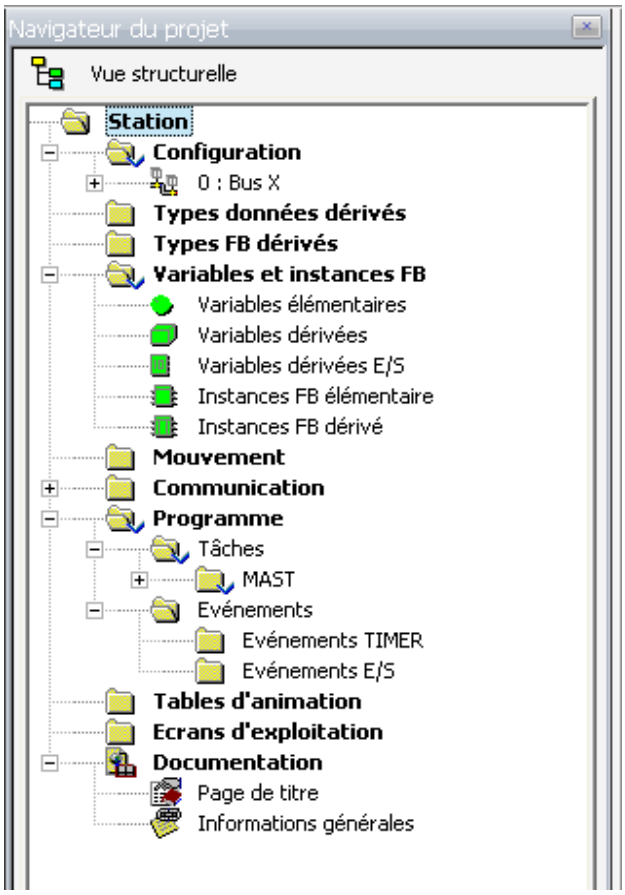


Figure II.2 : Navigateur de projet

II.3.1. Présentation du navigateur de projet

Le navigateur projet permet d'afficher le contenu d'un projet Unity Pro et de se déplacer dans ses différents éléments: configuration, données, programme, etc. On peut afficher le projet sous deux formes :

- La vue structurelle ;
- La vue fonctionnelle : qui permet d'afficher l'arborescence du projet, découpée en modules fonctionnels. Ce découpage ne prend pas en compte l'ordre d'exécution du programme par l'automate.

II.3.2. Différents répertoires du navigateur de projet

Le répertoire Station donne accès aux sous répertoires suivants, accessibles par le menu contextuel:

- Dossier configuration : donne accès à la configuration matérielle et au paramétrage des modules (bus, rack, module.).
- Dossier types données dérivés : donne accès aux types de DDT.
- Dossier types FB dérivés : donne accès aux types de DFB.
- Dossier variables et instances FB : permet d'accéder aux variables et aux instances de bloc fonction.
- Dossier mouvement : permet d'accéder à la déclaration et la configuration des variateurs.

- Dossier communication : permet d'accéder à la configuration des réseaux.
- Dossier programme : permet d'accéder aux différents programmes du projet.
- Tables d'animation : permet d'accéder aux tables d'animation.
- Ecrans d'exploitation : permet d'accéder aux écrans d'exploitation.
- Documentation : permet d'accéder à la documentation.

II.4. Gestion de bibliothèque de types [3]

La bibliothèque d'Unity Pro comprend tous les objets disponibles pouvant servir au développement d'un projet d'automatisation. Ces derniers peuvent comprendre les variables ou les fonctions suivantes :

- **EF** (fonctions élémentaires) ;
- **EFB** (blocs fonction élémentaire) ;
- **DFB** (blocs fonction utilisateur) ;
- **DDT** (variables).

▪Pour y accéder il faut aller dans le menu Outil, on sélectionne l'option Système de gestion de bibliothèque de types.

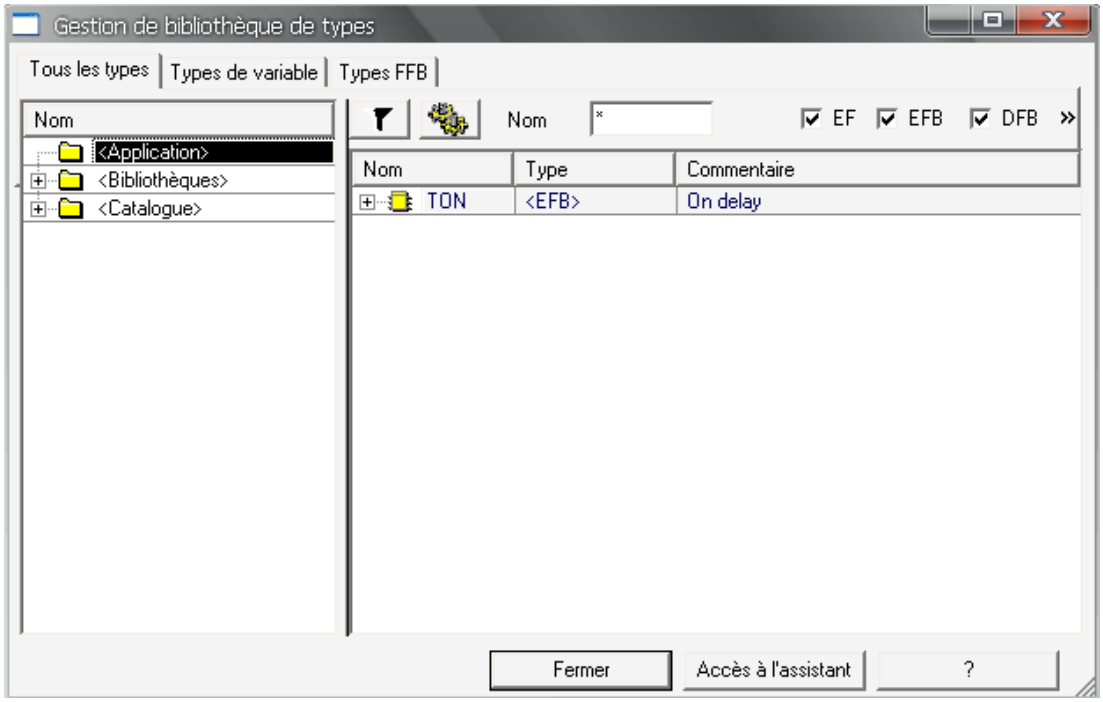


Figure II.3 : Gestion de bibliothèque de types

II.4.1. Fonction élémentaire (EF)

Une fonction élémentaire est représentée graphiquement sous forme de cadre avec des entrées et une sortie comme le montre la figure suivante :

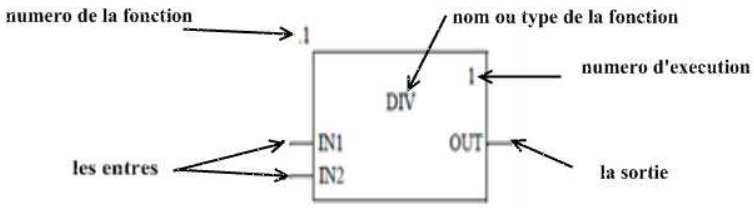


Figure II.4 : Fonction élémentaire

▪Comme le montre la figure, on a aussi un champ pour nom de la fonction (ou son type), le numéro d'exécution de la fonction et son numéro.

▪Les fonctions élémentaires (EF) ne disposent pas d'état interne et possèdent une seule sortie. Si les valeurs des entrées sont similaires, la valeur de la sortie est identique pour toutes les exécutions de la fonction. Par exemple, l'addition de deux valeurs donne le même résultat à chaque exécution de la fonction.

▪Une fonction élémentaire est représentée dans les langages graphiques (FBD et LD) sous la forme d'un rectangle avec des entrées et une sortie. Les entrées sont toujours représentées à gauche du rectangle et les sorties à droite. Le nom de la fonction (c'est-à-dire le type de fonction) est indiqué au centre du rectangle. Pour certaines fonctions élémentaires, il est possible d'augmenter le nombre d'entrées.

II.4.2. Bloc fonction élémentaire (EFB)

Les blocs fonction élémentaires (EFB) ont des états internes. Pour des valeurs égales aux entrées, la valeur à la sortie peut être différente pour toutes les exécutions de la fonction. Par exemple, pour un compteur, la valeur à la sortie augmente.
Les blocs fonction peuvent avoir plusieurs sorties.

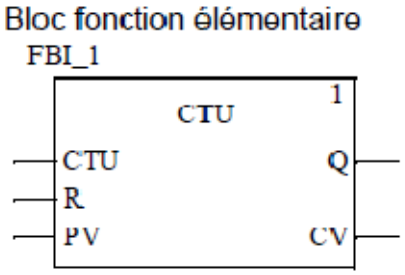


Figure II.5 : Bloc fonction élémentaire

II.4.3. Blocs fonctions dérivés

Les blocs fonction dérivés (DFB) ont les mêmes caractéristiques que les blocs fonction élémentaires. Ils sont cependant créés par l'utilisateur dans les langages FBD, LD, IL et/ou ST, l'unique différence par rapport aux blocs fonction élémentaires est que le bloc fonction dérivé est représenté graphiquement sous forme de cadre avec deux lignes verticales.

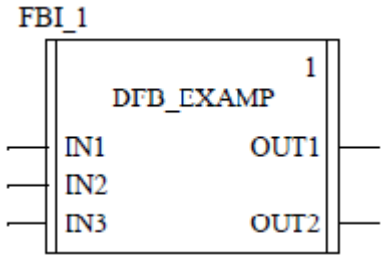


Figure II.6 : Bloc fonction dérivé

II.5. Editeur de données

L'éditeur de données est accessible à partir de la vue structurelle du projet. Il offre des services d'édition permettant :

- ✓ De créer des types de données ;
- ✓ D'archiver ou d'utiliser des types de données blocs fonction contenus dans une bibliothèque ;
- ✓ D'instancier des types de données ;
- ✓ De présenter de façon hiérarchique les structures de données ;
- ✓ De rechercher\ trier \filtrer les données.

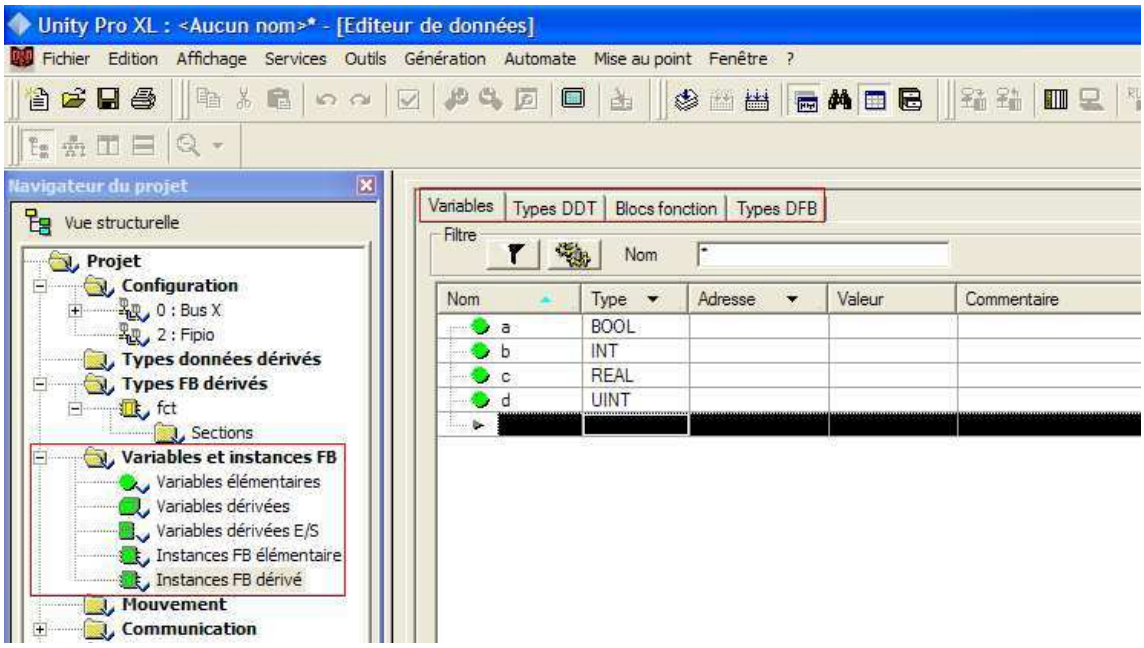


Figure II.7 : Editeur de données

On remarque quatre anglets :

- ➢ Onglet Variables : permet la gestion des instances de variables qui appartiennent aux familles EDT/ DDT/ IODDT ;
  - ▪EDT : Affichage des types de données élémentaires ;
  - ▪DDT : Affichage des types de données dérivés ;
  - ▪IODDT : Affichage des types de données dérivés (DDT) concernant les

entrées/sorties et ce type de données (structures) ont été prédéfinis par le constructeur et contiennent les objets langage d'E/S qui appartiennent à la voie d'un module métier (ou au module lui-même).

- Onglet Types DDT : permet la gestion des types de données dérivés (structures ou tableaux) ;
- Onglet Blocs fonction : permet la gestion des instances de variables du type EFB ou DFB qui appartiennent à la famille du bloc fonction.
- Onglet Types DFB : permet la gestion des types de données DFB (blocs fonction dérivés), ce dernier est un bloc fonction utilisateur personnalisé qui prend en compte la nature spécifique du projet et on peut le stocker dans la bibliothèque définie par l'utilisateur.

II.6. Communication

II.6.1. Présentation des éditeurs de communication

Les éditeurs de communication nous permettent de configurer et de gérer les différentes entités de communication au niveau du projet. Ils sont accessibles via le navigateur de projet en cliquant sur l'onglet Communication.

II.6.2. Configuration réseau

Sous Unity Pro, la mise en œuvre d'un réseau s'effectue à partir du navigateur d'application et à partir de l'éditeur de configuration matérielle. La méthode nécessite les quatre étapes suivantes :

- Création d'un réseau logique ;
- Configuration du réseau logique ;
- Déclaration du module de communication ou de la carte PCMCIA (pour Premium) ;
- Association de la carte ou du module au réseau logique.

II.7. Programmation [2]

La première opération pour créer un programme d'application consiste à définir les tâches. Par défaut seule la tâche maître est proposée. Il est possible de créer les tâches : rapide FAST et auxiliaires AUX0 à 3.

Un programme s'exécute en trois étapes, à savoir, lecture de données puis traitement du programme et en fin écriture des sorties. Il peut être ***cyclique*** ou ***périodique***, dans le premier cas il entame le prochain cycle juste après la mise à jour des sortie, alors que pour le deuxième il ne l'entame qu'après un temps fixe (période) même si la mise à jour des sorties a été faite.

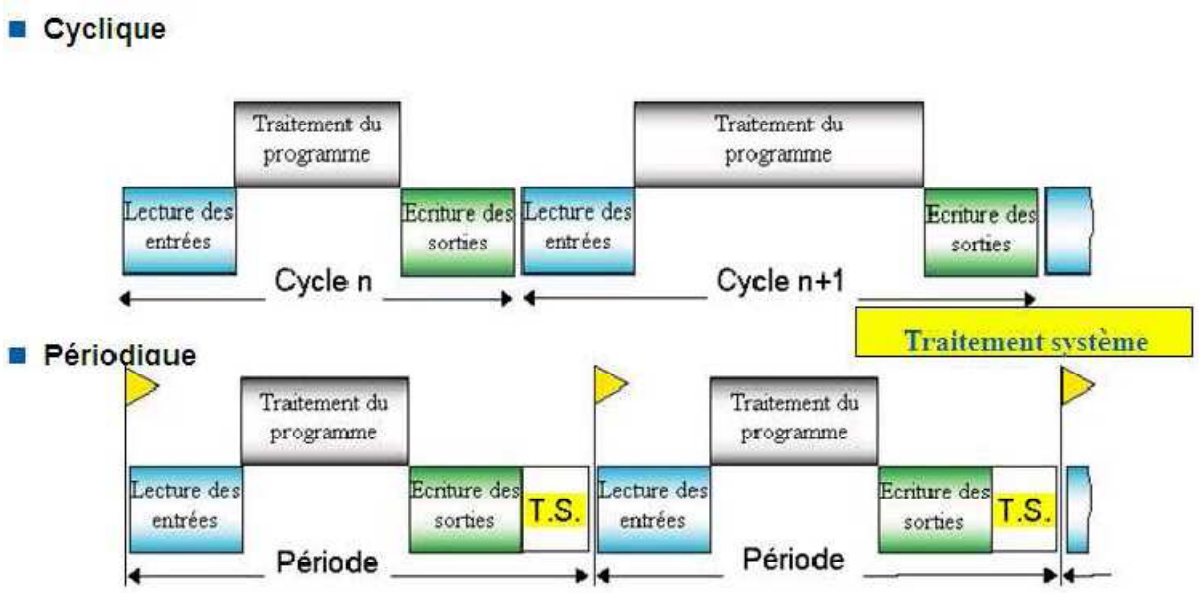


Figure II.8 : Etapes d'exécution d'un programme (cyclique et périodique)

▪Un programme peut être :

- Mono-tâche : Ne comprenant que la tâche principale (MAST) ;
- Multitâches : Comprenant les tâches MAST + tâches rapides (FAST) + tâches événementielles (EVT) + tâches auxiliaires (AUX).

▪Ces tâches diffèrent de part leur priorité d'exécution, la figure suivante montre l'ordre de priorité comme le montre la figure suivante :

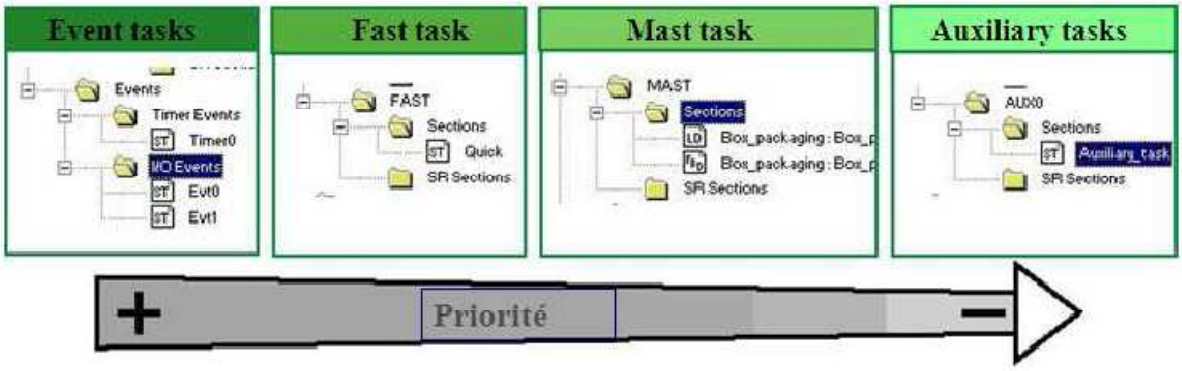


Figure II.9 : Ordre de priorité des différentes tâches

II.7.1. Différentes tâches

II.7.1.1. La tâche MAST

La tâche « maître » représente le programme principal. Elle est obligatoire quelle que soit la structure adoptée mono-tâche ou multitâche donc elle est crée par défaut.

Le programme de la tâche maître est constitué de plusieurs modules de programmes appelés sections et sous-programmes, qui sont liées à une tâche. Une même section ou sous programme ne peut pas appartenir simultanément à plusieurs tâches.

Les appels aux sous-programmes s'effectuent dans les sections ou depuis un autre sous programme (8 niveaux d'imbrications maximum) et enfin l'exécution de cette tâche peut être choisie (en configuration) : cyclique ou périodique.

II.7.1.2.Tâche FAST

C'est une tâche plus prioritaire que la tâche maître MAST et périodique avec une période configurable de 1 à 255 ms. Comme pour la tâche MAST, le programme associé se compose de sections et de sous-programmes. Le contrôle de la tâche ainsi que la visualisation des temps d'exécution se fait avec des bits systèmes associés.

▪Exemple : le bit système %S11 est positionné à 1 et l'application est déclarée en défaut bloquant pour l'automate en cas de débordement (chien de garde).

II.7.1.3. Tâches auxiliaires AUX

Une tâche auxiliaire est utilisée pour les tâches de traitement plus lentes. On peut programmer jusqu'à 4 tâches auxiliaires maximum (AUX0 à AUX3) sur Premium TSX P57 5•• et Quantum 140 CPU 6••••

Elle est composée de sections et des sous-programmes qui peuvent être programmés en LD, FBD, IL ou en ST, son exécution est périodique (de 10 ms à 2,55 s).

II.7.1.4. Tâches événementielles

Elle est utilisée pour réduire le temps de réponse du programme aux événements des modules d'entrée/sortie et aux temporisateurs d'événement, elle contient une seule section qui peut être programmée en LD, FBD, IL ou en ST.

II.7.2. Création d'une section

On peut utiliser l'importe quel langage pour la création des différentes taches, et dans chaque taches, on peut crée plusieurs sections. La seule restriction concerne le SFC, ce dernier peut être utilisé uniquement pour les taches MAST

II.7.3. Création d'un programme en utilisant le langage FBD

Un programme FBD possède les propriétés suivantes :

- Une section FBD est placée sur une grille ;
- Une unité de grille comprend 10 points de trame. Une unité de grille est l'espace le plus petit possible entre deux objets d'une section FBD ;
- Une section FBD peut être configurée en nombre de cellules (points de trame horizontaux et points de trame verticaux) ;
- Le langage FBD n'est pas basé sur les cellules. Les objets sont toutefois ajustés sur les unités de grille ;

- L'ordre d'exécution est défini par la position du FFB dans la section (exécution de gauche à droite et de haut en bas). Si les FFB sont ensuite liés avec des liaisons graphiques à un réseau, l'ordre d'exécution est défini par le flux des signaux, voir également le sous-chapitre Ordre d'exécution des FFB dans le manuel de référence. L'ordre d'exécution peut être influencé de plusieurs manières ;
- Une vérification de la syntaxe et de la sémantique a lieu directement après la saisie des instructions d'affectation. Le résultat de la vérification est indiqué par différentes couleurs de texte et d'objet ;
- Les sections comportant des erreurs de syntaxe ou de sémantique peuvent également être enregistrées.

▪Les objets du langage de programmation FBD (langage des blocs fonction) offrent des aides permettant de structurer une section en un ensemble de : EF et EFB (fonctions élémentaires et blocs fonction élémentaires), DFB (blocs fonction dérivés) et procédures. Ces objets, regroupés sous l'abréviation générique FFB, peuvent être liés les uns aux autres par : des liaisons ou des paramètres réels et la logique du programme peut être commentée avec des objets texte.

Remarque : FFB est le terme générique pour :

-Les fonctions élémentaires (EF) ;
-Les blocs fonction élémentaires (EFB) ;
-Les blocs fonction dérivés (DFB) ;
-Les Procédure.

▪L'activation de l'assistant de saisie FFB peut se faire des différentes manières suivantes :

- Exécution de la commande Edition → Assistant de saisie FFB... (Aucun objet ne doit être sélectionné) ;
- Utilisation de la commande de menu du menu contextuel (il se peut qu'aucun objet ne soit sélectionné). Ou on appuie sur les touches CTRL+I (aucun objet ne doit être sélectionné) ;

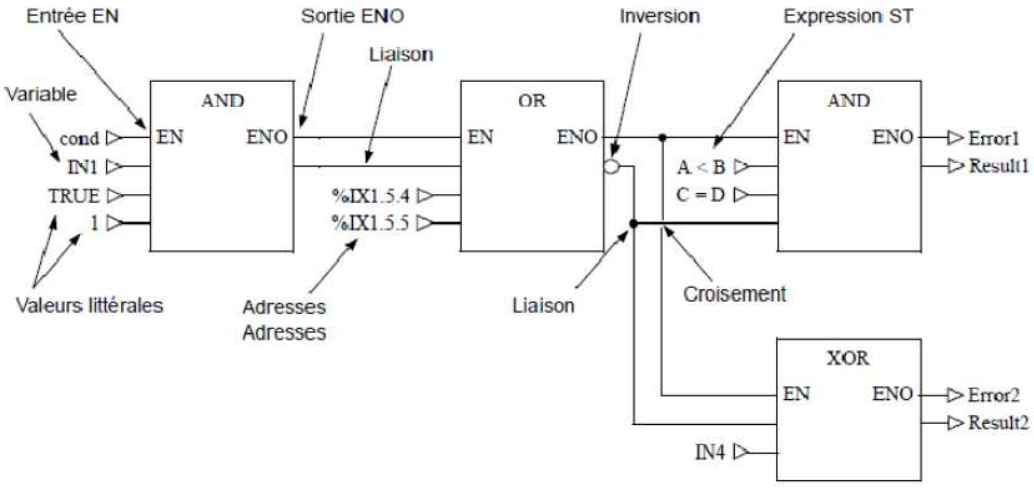


Figure II.10 : Représentation d'une section FBD

II.7.4. Création d'un programme en utilisant le langage LD

Le langage à contacts ou bien le Ladder (LD) est un langage graphique, il permet la transcription des schémas à relais, il est excellent pour le combinatoire et les problématiques d'inter verrouillage.

La structure d'un programme LD correspond à une voie de courant pour des montages à relais, sur le côté gauche de l'éditeur LD se trouve la barre d'alimentation gauche. Cette barre d'alimentation gauche correspond à la phase (conducteur L) d'une voie de courant. De même que sur une voie de courant, pour la programmation LD ne sont "traités" que les objets LD qui sont branchés sur l'alimentation en courant, c'est-à-dire qui sont reliés à la barre gauche d'alimentation. La barre d'alimentation droite correspond au conducteur neutre.

Tous les contacts et entrées FFB doivent être reliés directement ou indirectement à la barre d'alimentation gauche et toutes les bobines et sorties FFB doivent être reliées directement ou indirectement à la barre d'alimentation droite afin de créer un flux d'énergie.

▪Propriétés d'un programme LD :

- Une section LD a une largeur de 11-64 colonnes et 17-2 000 lignes, le nombre de colonnes est défini dans la boîte de dialogue Outils → Options du projet dans l'onglet Editeurs dans la zone de texte Nombre de colonnes ;
- Les programmes LD sont orientés sur les cellules, c'est-à-dire qu'un seul objet peut être placé dans chaque cellule ;
- L'ordre d'exécution des différents objets dans un programme LD est déterminé par le flux de données à l'intérieur de la section. Les réseaux branchés sur la barre d'alimentation gauche sont traités du haut vers le bas.

▪Les objets du langage de programmation LD offrent des aides permettant de structurer une section en un ensemble de :

- ✓ Contacts ;
- ✓ Bobines ;
- ✓ EF et EFB (fonctions élémentaires et blocs fonction élémentaires) ;
- ✓ DFB (blocs fonction dérivés) ;
- ✓ Procédures ;
- ✓ Blocs opération ;
- ✓ Blocs comparaison ;
- ✓ Sauts au sein de la section ;
- ✓ Appels du sous-programme.

▪Ces objets peuvent être liés les uns aux autres par :

- ➢ Des liaisons ;
- ➢ Des paramètres réels (FFB uniquement).

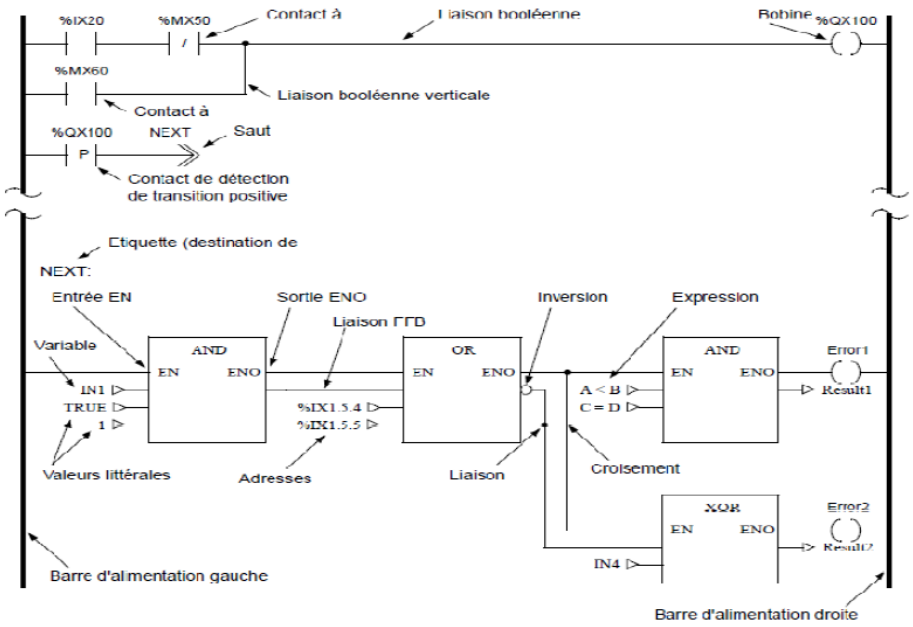


Figure II.11 : Représentation d'une section en langage contact

❖ **Connexion d'une FFB**

Les FFB ne sont traités que lorsqu'ils sont connectés directement ou indirectement à la barre d'alimentation gauche. Si l'FFB doit être exécuté sous conditions, vous pouvez effectuer une liaison préliminaire de l'entrée EN à l'aide de contacts ou d'autres FFB.

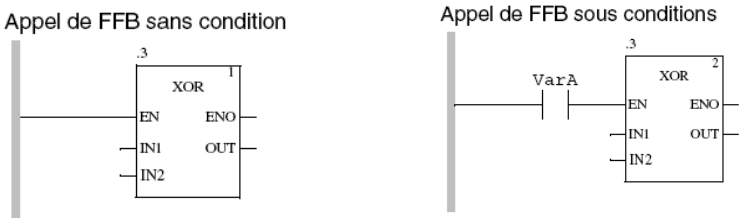


Figure II.12 : Connexion d'une FFB

▪Si la valeur de l'entrée EN est "0" lorsque le FFB est appelé, les algorithmes définis par le FFB ne sont pas exécutés et la sortie ENO est réglée sur "0".

▪Si la valeur de l'entrée EN est "1" lorsque le FFB est appelé, les algorithmes définis par le FFB sont exécutés. Une fois les algorithmes exécutés, la valeur de la sortie ENO est réglée sur "1". En cas d'erreur lors de l'exécution de ces algorithmes, la sortie ENO est réglée sur "0".

II.7.5. Création d'un programme en utilisant le langage SFC

Le langage séquentiel SFC est un langage à Diagramme fonctionnel en séquence. Un diagramme fonctionnel en séquence conforme à CEI se compose dans Unity Pro de sections SFC (niveau supérieur), de sections de transition et de sections d'action.

Les sections SFC ne sont autorisées que dans la tâche maître du projet. Dans les autres tâches ou les DFB, les sections SFC ne peuvent pas être utilisées. Chaque section SFC comprend un ou plusieurs réseaux SFC (séquences).

Chaque étape compte zéro ou plusieurs actions et chaque transition comprend une condition de transition.

La dernière transition de la séquence est toujours liée à la dernière étape de la séquence (via une liaison graphique ou un symbole de saut). Les chaînes d'étapes se déroulent donc de façon cyclique.

▪Caractéristiques d'un programme SFC :

- La section SFC comporte toujours une grille de fond, Pour des raisons de performance, il est recommandé de créer moins de 100 sections SFC dans un projet (les macro-sections ne sont pas comptabilisées) ;
- Une section SFC contient au maximum 200 lignes et 32 colonnes ;
- Théoriquement les objets SFC peuvent être placés dans toutes les cellules qui ne sont pas occupées ;
- Les étapes, transitions et sauts ont respectivement besoin d'une cellule ;
- Pour éviter de devoir diviser les chaînes d'étapes, il est possible de représenter verticalement 99 étapes enchaînées avec leurs transitions en plus d'un saut terminal avec sa transition.

▪Les objets du langage de programmation SFC (diagramme fonctionnel en séquence) offrent des aides permettant de structurer une section en un ensemble de :

- ✓ Etape ;
- ✓ Macro-étape (sous macro-étape intégrée) ;
- ✓ Transition (conditions de transition) ;
- ✓ Saut ;
- ✓ Chaîne en OU ;
- ✓ Chaîne en ET ;

▪Ces objets peuvent être liés les uns aux autres par des liaisons

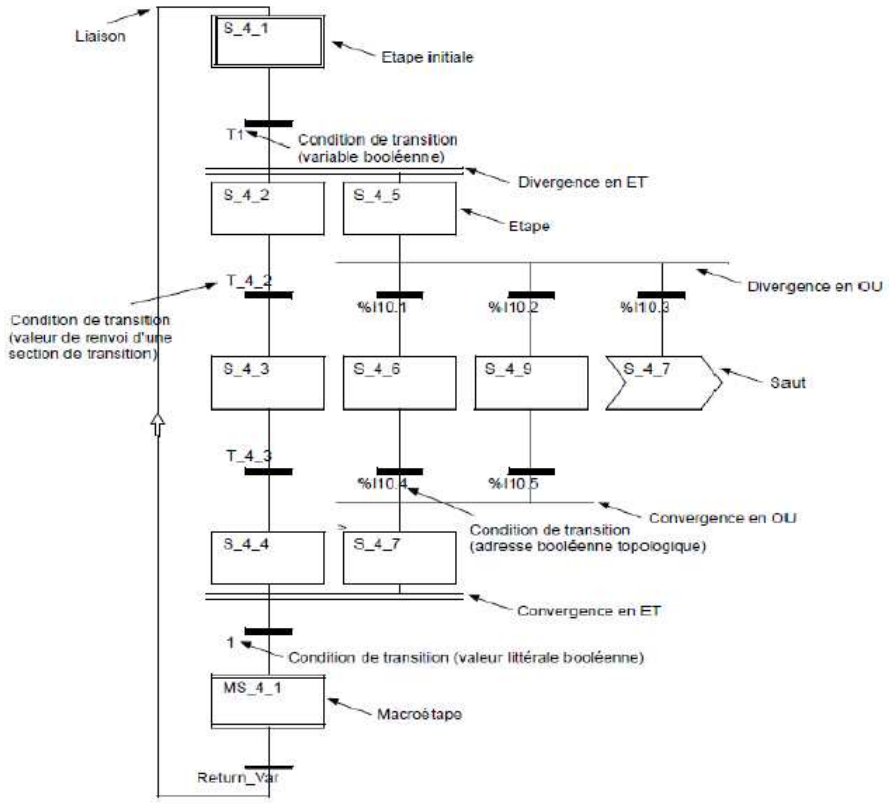


Figure II.13 : Représentation d'une section SFC

II.7.6. Création d'un programme en utilisant le langage IL

Un programme IL (liste d'instructions) comprend une suite d'instructions qui sont traitées l'une après l'autre par l'automate. Les listes d'instructions nous permettent par exemple d'appeler des blocs fonctions, des fonctions et des procédures de façon conditionnelle ou inconditionnelle, d'exécuter des affectations et d'exécuter des sauts conditionnels ou inconditionnels au sein de la section.

▪Propriétés d'un programme IL :

- Les instructions sont composées des éléments suivants : un opérateur, le cas échéant avec modificateur, si nécessaire un opérande et le cas échéant un commentaire ;
- Chaque instruction peut également comprendre une étiquette (Label) ;
- Chaque instruction commence dans une nouvelle ligne ;
- Une ligne est limitée à 300 caractères ;
- Il est possible d'utiliser des sauts de ligne dans les instructions (instructions d'affectation à plusieurs lignes) ;
- Des étiquettes, symboles et commentaires peuvent être librement placés dans la section. (Les commentaires peuvent être saisis à tout endroit où les espaces sont autorisés) ;
- Une vérification de la syntaxe et de la sémantique a lieu directement après la saisie des instructions d'affectation. Le résultat de la vérification est indiqué par différentes couleurs de texte ;

- Les sections comportant des erreurs de syntaxe ou de sémantique peuvent également être enregistrées ;
- Chaque instruction commence dans une nouvelle ligne et est composée de :
  - Commentaire (optionnel) : Informations supplémentaires sur la ligne.
  - Opérateur : définit la nature de l'instruction. Les différents opérateurs sont :
    - · Assignement (*LD, ST, S, R*);
    - · Logique (*AND, OR, XOR, NOT*);
    - · Arithmétique (*ADD, SUB, MUL, DIV, MOD*) ;
    - · Comparaison (*GT, GE, EQ, NE, LE, LT*) ;
    - · Structure (*JMP, RET*) ;
    - · Appel de fonction (*CAL function_name*) ;

  - Modificateur : influence l'exécution de l'opérateur, les différents modificateurs sont :
    - · N : invertit la valeur de l'opérande bit par bit ;
    - · C : l'instruction associée n'est exécuté que si le résultat est vrai ;
    - · CN : l'instruction associée n'est exécuté que si le résultat est faux ;
    - · ( ) :utilisées pour pouvoir combiner les différentes instructions.

  - Opérande : l'objet sur lequel agit l'opérateur, il peut être une adresse directe, une valeur, une variable, un DDT, un élément d'une DDT, une sortie d'une EFB ou bien un appel d'une EFB/DFB.

  - Etiquette (optionnel) : localise une séquence dans le programme.

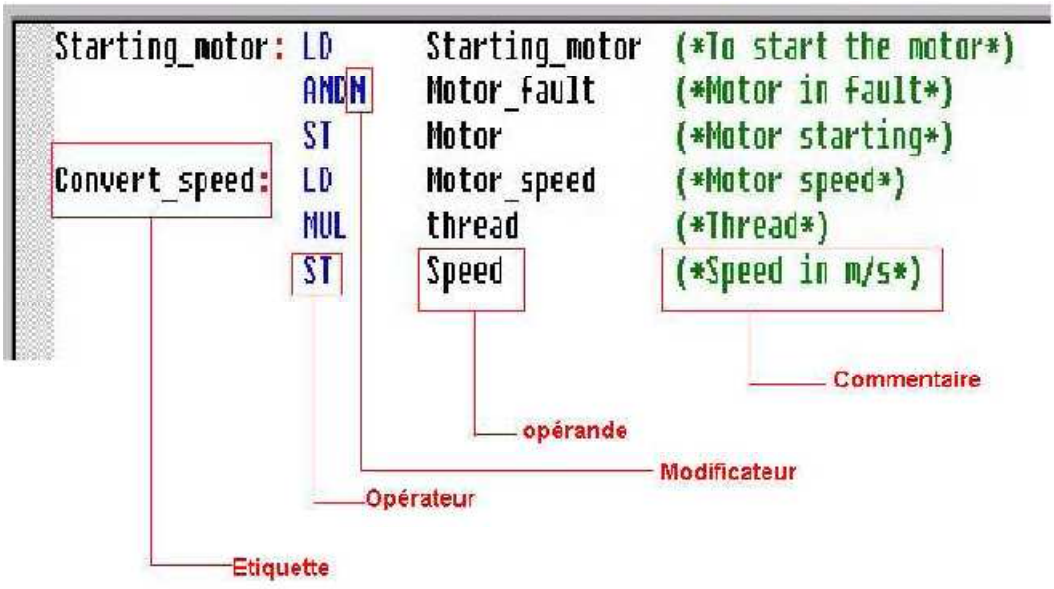


Figure II.14 : Représentation d'une section IL.

II.7.7. Création d'un programme en utilisant le langage ST

Un programme ST (littéral structuré) comprend une suite d'instructions qui sont traitées l'une après l'autre par l'automate. Les listes d'instructions vous permettent par exemple d'appeler des blocs fonctions, des fonctions et des procédures de façon conditionnelle ou inconditionnelle, d'exécuter des affectations, d'exécuter des instructions de façon conditionnelle, de répéter des instructions et des sauts au sein de la section de façon conditionnelle ou inconditionnelle.

▪Propriétés d'un programme ST :

- Les instructions sont composées des éléments suivants : un opérateur, un opérande, le cas échéant une expression et le cas échéant un commentaire ;
- Chaque instruction peut également comprendre une étiquette (Label) ;
- Les instructions doivent être terminées par des points-virgules ;
- Une ligne peut contenir plusieurs instructions (séparées par des points-virgules) ;
- Un seul point-virgule représente une instruction vide ;
- Une ligne est limitée à 300 caractères ;
- Il est possible d'utiliser des sauts de ligne dans les instructions (instructions d'affectation à plusieurs lignes) ;
- Des étiquettes, symboles et commentaires peuvent être librement placés dans la section. (Les commentaires peuvent être saisis à tout endroit où les espaces sont autorisés) ;
- Une vérification de la syntaxe et de la sémantique a lieu directement après la saisie des instructions d'affectation. Le résultat de la vérification est indiqué par différentes couleurs de texte ;
- Les sections comportant des erreurs de syntaxe ou de sémantique peuvent également être enregistrées.

```
VarA := VarB*VarB-4*VarC*VarD ;
IF VarA < 0.0 THEN VarE := 0 ;
ELSIF VarA = 0.0 THEN
VarE := 1 ;
X1 := -X2/(2.0*VarC) ;
ELSE
VarE := 2;
X1 := (-B+SQRT(D)) / (2.0*A) ;
X2 := (-B-SQRT(D)) / (2.0*A) ;
END_IF;
```

Figure II.15 : Représentation d'une section ST

▪Le littéral structuré utilise aussi des instructions :

- Le contrôle conditionnel IF…THEN…END_IF ;
- Le contrôle conditionnel REPEAT…END_REPEAT ;

- le contrôle conditionnel WHILE…END_WHILE ;
- le contrôle conditionnel FOR…END_FOR.

II.8. Configuration des racks sur le bus local

Lors de la création d'un projet, un rack par défaut est sélectionné. Son adresse est la suivante :

- **0** pour un automate de la famille Premium/Atrium ou Modicon M340 ;
- **1** pour un automate de la famille Quantum.

Ce rack contient le type de processeur sélectionné lors de la création du projet. Il est possible de remplacer ce processeur par un processeur compatible. Pour Premium et Quantum, le nombre de racks gérés diffère selon le processeur configuré. Pour Modicon M340, la station ne comporte qu'un seul rack.

- **Organisation d'une station Quantum sur bus local**

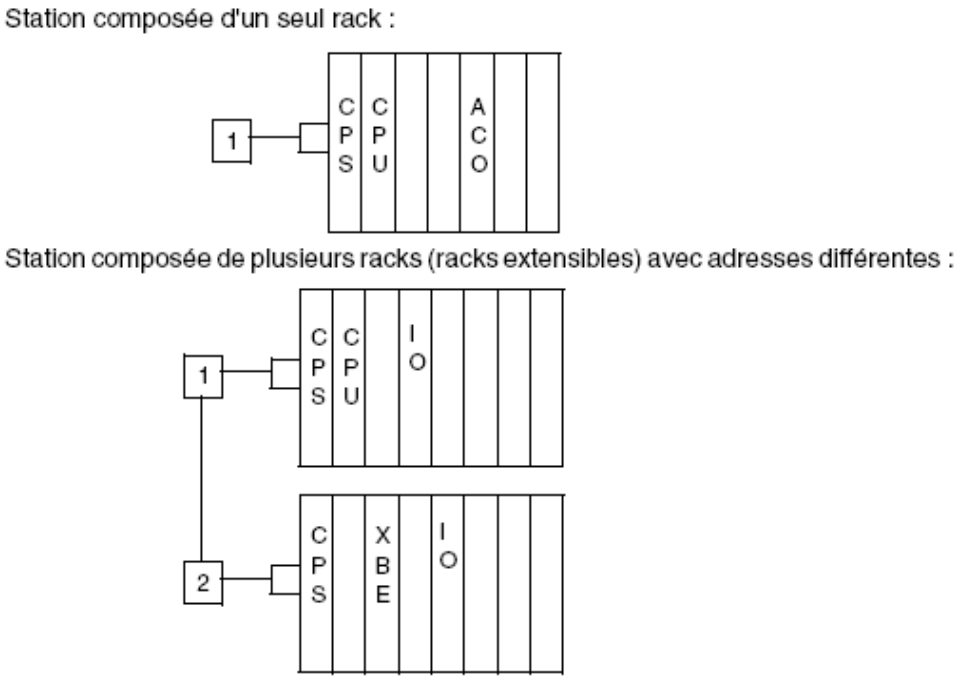
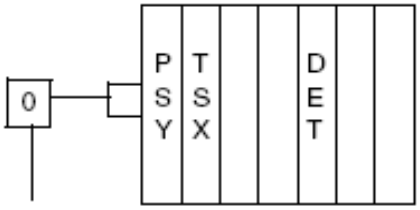


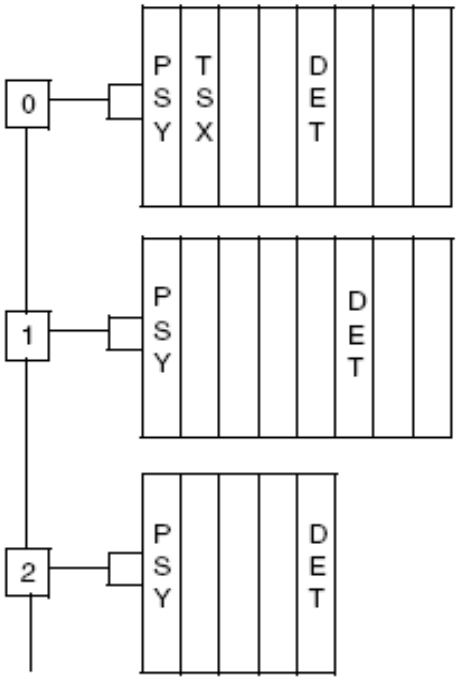
Figure II.16 : Organisation d'une station Quantum sur le bus local

- **Organisation d'une station Modicon M340 sur bus automate puis Organisation d'une station Premium/Atrium sur bus X**

Station composée d'un seul rack (rack standard) :



Station composée de plusieurs racks (racks extensibles) avec adresses différentes :



Station composée de plusieurs racks (racks extensibles) avec la même adresse :

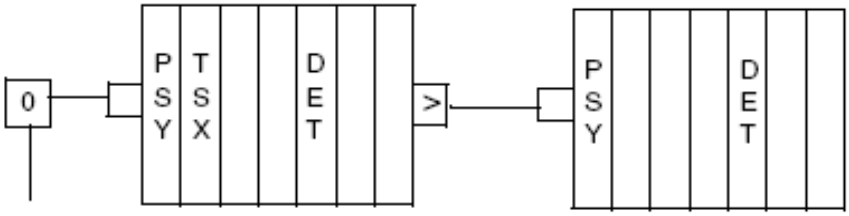


Figure II.17 : Organisation des stations Modicon M340, Premium et Atrium

II.9. Configuration des modules d'alimentation [3]

Lorsque on crée une application, deux zones sont générées selon qu'on sélectionne une station Premium/Atrium, une station Quantum ou une station Modicon M340 :

- ✓ dans une station Modicon M340 ou Premium, un module d'alimentation est configuré par défaut ;
- ✓ Dans une station Quantum, aucun module d'alimentation n'est configuré par défaut.

II.9.1. Règles pour une station Modicon M340

Le module d'alimentation doit occuper la position la plus à gauche du rack. Cette position ne dispose pas d'adresse.

II.9.2. Règles pour une station Premium/ Atrium

Le module d'alimentation doit occuper la position la plus à gauche du rack. Cette position ne dispose pas d'adresse.

Un module d'alimentation double format occupe aussi la position d'adresse 0 (occupée habituellement par le module processeur), dans ce cas le module processeur doit être configuré à la position d'adresse 1.

Remarque : Il y a un seul module d'alimentation par rack.

II.9.3. Règles pour une station Quantum

Le module d'alimentation peut occuper n'importe quelle position du rack. Il dispose d'une adresse. Les modules d'alimentation sont simple format et dans un rack, on peut configurer plusieurs modules d'alimentation.

II.10. Simulateur d'automate

Le simulateur permet de simuler l'UC d'un automate de la famille Premium ou Quantum. On peut alors tester notre projet dans cet automate simulé en utilisant des points d'arrêt (breakpoints), le pas à pas (stepping) et la fonction Atteindre.

▪Le simulateur est installé automatiquement avec Unity Pro, sa boite de dialogue donne les indications suivantes :

- ✓ Type d'automate simulé ;
- ✓ Etat actuel de l'automate simulé ;
- ✓ Nom du projet chargé ;
- ✓ Adresse IP et nom DNS du PC hôte du simulateur ;
- ✓ Adresses IP et noms DNS de tous les PC clients connectés ;
- ✓ Boîte de dialogue de simulation des événements E/S ;
- ✓ Bouton de réinitialisation pour simuler un démarrage à froid ;
- ✓ Bouton redémarrage pour simuler un démarrage à chaud ;
- ✓ Menu de raccourcis (bouton droit de la souris) pour contrôler le simulateur.

▪Le symbole de simulateur affiché dans la barre des tâches permet d'accéder aux fonctions suivantes :

- ➢ Affichage de l'état actuel de l'automate simulé ;
- ➢ Info-bulle qui affiche l'adresse IP du PC hôte du simulateur et le nom du projet chargé ;
- ➢ Menu de raccourcis (bouton droit de la souris) pour contrôler le simulateur.

▪Le processus de chargement d'un projet dans le simulateur est identique au principe de chargement d'un projet sur un automate réel.

▪La procédure standard consiste à installer le simulateur sur le même PC que celui accueillant le logiciel Unity Pro (hôte local). Lorsque la commande de menu Automate →Connexion est

sélectionnée, une connexion avec le PC hôte local est créée automatiquement. Il est également possible de charger un simulateur sur un autre PC via une connexion TCP/IP. Dans ce cas, on doit indiquer l'adresse TCP/IP du PC cible lors de la procédure de chargement d'un projet avant la création d'une connexion, ensuite, on sélectionne la commande de menu Unity Pro Automate →Définir l'adresse, puis dans la zone de texte Adresse dans la zone Simulateur, on saisie l'adresse TCP/IP du PC cible. On passe ensuite à la création de la connexion.

▪Si on souhaite charger le projet sur le simulateur du PC hôte local (Unity Pro et simulateur sur le même PC), on ne doit pas exécuter cette étape, car l'adresse du PC hôte local est saisie automatiquement.

❖ **Les limites du simulateur de l'automate sont**

Le simulateur de l'automate simule un projet complet avec toutes ses tâches utilisateur. Le comportement de l'exécution de la simulation n'étant cependant pas comparable à celui d'un automate réel, il ne permet en aucun cas de tirer des conclusions quant au comportement d'un automate réel. (Comportement multitâche et informations de temps).

Le simulateur de l'automate ne prend aucune forme d'E/S en charge. Bien que la simulation contienne tous les composants projet pour les E/S, ils ne sont pas traités par le simulateur de l'automate. Vous ne pouvez accéder aux entrées et sorties qu'à partir du projet ou via les fonctions en ligne du logiciel Unity Pro (lire, écrire, forcer, animer, etc.).

Le simulateur d'automate ne permet pas de déclencher des événements d'E/S en réglant/forçant les bits %I.

❖ **Limites de communication**

Le simulateur d'automate prend uniquement en charge les communications TCP/IP (port Schneider 502). Dans tous les autres cas, le système signale une erreur Modbus. Les protocoles Modbus, Modbus Plus et UNITE ne sont pas reconnus.

Le simulateur de l'automate ne peut pas communiquer avec d'autres ordinateurs ou simulateurs d'automate, que ce soit dans un réseau local ou à distance. Il n'a aucun délai d'attente de communication.

Les réseaux de communication tels qu'Uni-Telway, ETHWAY, FIWAY, Modbus, Modbus Plus, etc. ne sont pas compatibles avec le simulateur d'automate.

II.11. Présentation des tables d'animation

On y accède depuis le navigateur de projet, elle permet de :

- modifier les variables internes grâce à l'onglet *« modification »*, soit en en inscrivant la valeur dans le champ correspondant soit en utilisant la mise à 1/0 (pour les booléens) ;

- Forcer l'état d'une variable affectée à un module d'entrées grâce à l'onglet *« Forcer»*, et ça en inscrivant la valeur ou en utilisant aussi la mise à 1 ou mise à 0.

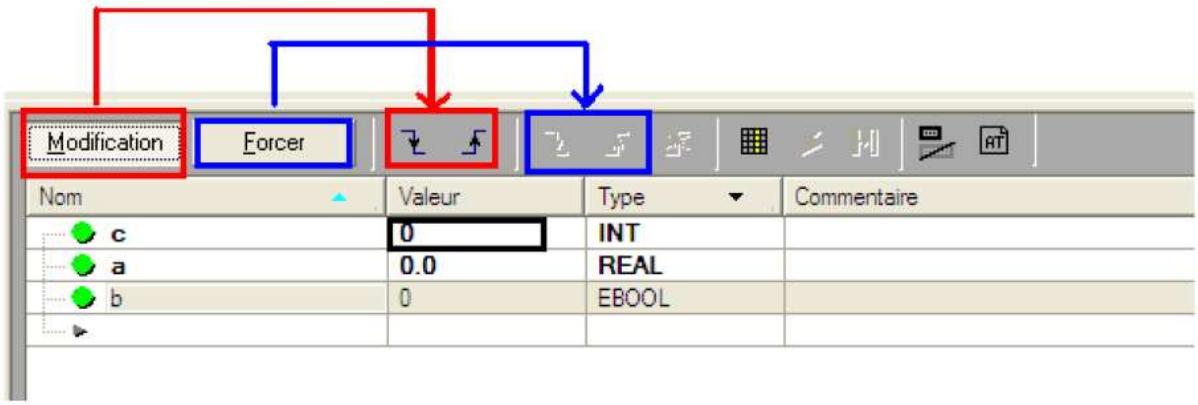


Figure II.18 : Table d'animation

▪Une table d'animation est découpée en 3 zones qui sont :

- Zone Mode ;
- Zone Commande ;
- Zone Affichage ;

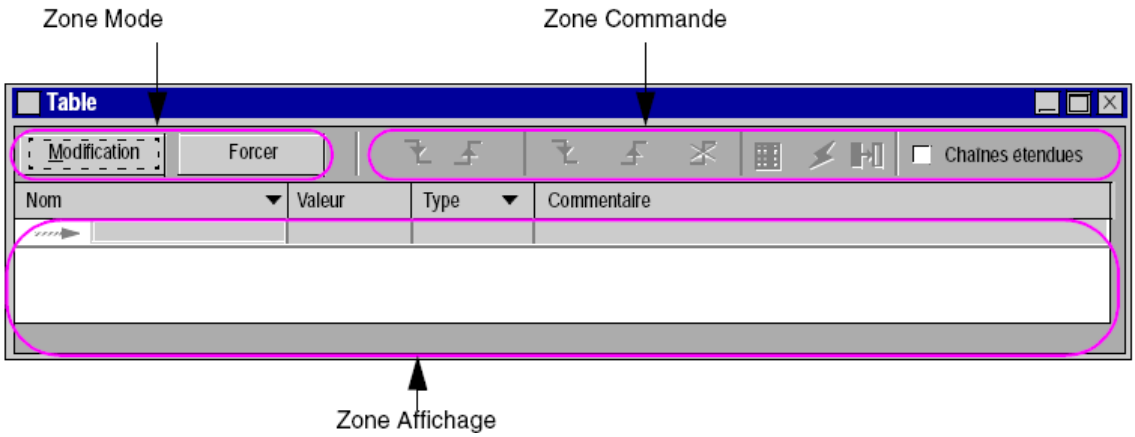


Figure II.19 : Zones de la table d'animation

II.12. Ecran d'exploitation

Les écrans d'exploitation intégrés sont destinés à faciliter l'exploitation d'un procédé automatisé. Ils utilisent dans le logiciel Unity Pro :

- Le navigateur projet : qui permet de naviguer dans les écrans et lancer les différents outils (l'éditeur graphique, l'éditeur de variables, l'éditeur de messages, ...) ;
- L'éditeur graphique : qui permet de créer ou modifier les écrans. En mode connecté, il permet également de visualiser les écrans animés et de conduire le procédé,
- La bibliothèque d'objets : qui présente des objets constructeur et permet de les insérer dans les écrans. Elle permet aussi de créer ses propres objets et de les insérer dans une famille de la bibliothèque.

▪Pour un projet donné, on peut créer des écrans d'exploitation, en utilisant l'éditeur graphique. Ces écrans sont réalisés au moyen de textes et d'objets graphiques qu'on peut dessiner (lignes, rectangles, courbes,...) ou récupérer dans la bibliothèque des objets graphiques. Ils sont constitués de parties statiques (fond de l'écran, titre,...) et de parties dynamiques ou animées qui permettent de refléter l'état du procédé.

▪Pour animer les objets dynamiques, on doit leurs affecter une variable dont la valeur déterminera l'affichage, pour conduire le procédé on peut également insérer dans les écrans des objets de pilotage (boutons, zones de saisie,...).

▪Les écrans peuvent être liés entre eux afin de répondre aux exigences spécifiques de l'automatisme.

▪Pour accéder aux écrans d'exploitations, il faut Visualiser le projet selon la vue structurelle (Affichage_Vue Structurelle), puis déployer le dossier écran d'exploitation. Si on veut ouvrir un nouvel écran il faut choisir « Nouvel écran » dans le menu contextuel du dossier écran d'exploitation.

▪Les objets qui peuvent être créés dans un écran graphique sont de 4 types :

- ✓ Les objets standards : ligne, rectangle, ellipse, courbe, polyligne, texte. Ils peuvent être statiques ou dynamiques (ayant une variable associée modifiant leur affichage) ;
- ✓ Les images : fichiers bitmap avec l'extension BMP ou JPG ;
- ✓ Les objets de pilotage (ou de commande) : bouton, case à cocher, champ de saisie, compteur, curseur, objet d'échange explicite, bouton de navigation écran…Ils sont activés par une action de la souris (ou du clavier). En fonction de l'attribut qui a été fixé, ces objets agissent sur leurs variables associées ;
- ✓ Les objets composés : ensemble d'objets des 3 types précédents, créé par l'utilisateur ou en provenance de la bibliothèque d'objets.

▪La bibliothèque d'objets présente les objets constructeurs et permet de les insérer dans les écrans d'exploitation. Les objets sont classés dans des familles. La bibliothèque permet aussi de créer ses propres objets en les insérant dans une famille de la bibliothèque, cette dernière s'ouvre à partir de la commande Outils_Bibliothèque des écrans d'exploitation.

II.13. Exemples d'applications

II.13.1. Simulation du fonctionnement d'une perceuse

▪Cahier de charge : Le système consiste en une perceuse qui doit effectuer un cycle selon la pièce posé comme la montre la figure :

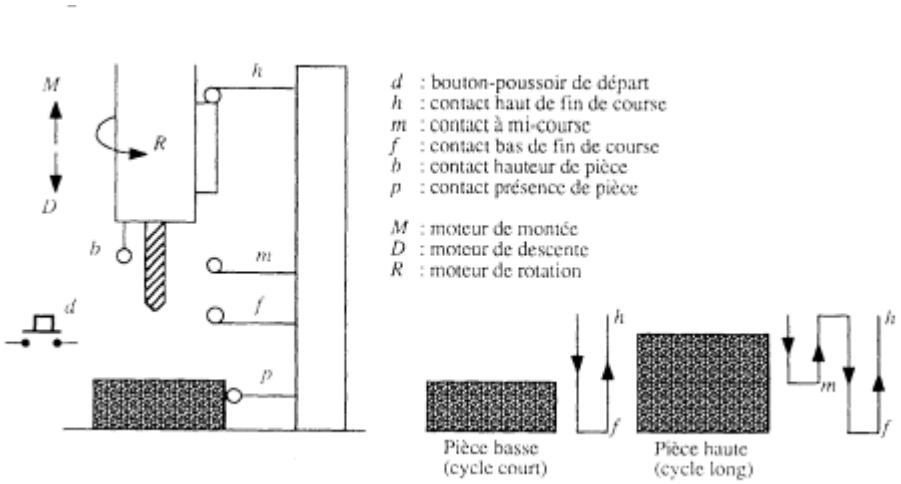


Figure II.20 : Exemple perceuse

- Le système comporte :

▪Trois contacts de fin de course : h, m, f.

▪Un bouton poussoir : « d » qui permet de démarrer le cycle.

▪Un contact « b » pour indiquer la hauteur.

▪Un contact « p » qui indique la présence d'une pièce.

▪Un moteur de rotation : R, moteur de descente : D et un moteur de monté : M.

- Le cycle est le suivant :

Le cycle commence lorsque l'on appuie sur le bouton poussoir d, s'il y a une pièce présente. Les pièces à percer peuvent être de deux types: pièce basse ou pièce haute. Lorsque la pièce est basse le cycle est le suivant. Dès le début du cycle, on a mise en route du moteur de descente et du moteur de rotation de la broche portant le foret. Quand le contact f est atteint, la broche remonte jusqu'au contact h et la rotation s'arrête à ce moment-là. Lorsque la pièce est haute (ce qui est repéré par le fait que le contact b se produit avant le contact à mi-course m), la broche remonte jusqu'au contact h quand le contact m est atteint, puis redescend jusqu'au contact f avant de remonter dans les mêmes conditions que dans le cycle court correspondant à une pièce basse. Avant de recommencer un nouveau cycle, il faut que la pièce déjà percée ait été retirée et remplacée.

- Réalisation matérielle :

La réalisation matérielle du réseau de Pétri à base des bascules RS est la suivante :

X_0: $\begin{cases} S_0=X_4.\overline{P} \\ R_0=X_0.d.p \end{cases}$ X_1: $\begin{cases} S_1=X_0.d.p \\ R_1=X_1.m.\ \overline{b}+X_1.m.b \end{cases}$

X_2: $S_2 = X_1.m.\bar{b} + X_5.h$; $R_2 = X_2.f$

X_3: $S_3 = X_2.f$; $R_3 = X_3.h$

X_4: $S_4 = X_3.h$; $R_4 = X_4.\bar{P}$

X_5: $S_5 = X_1.m.b$; $R_5 = X_5.h$

▪Les équations des sorties sont:

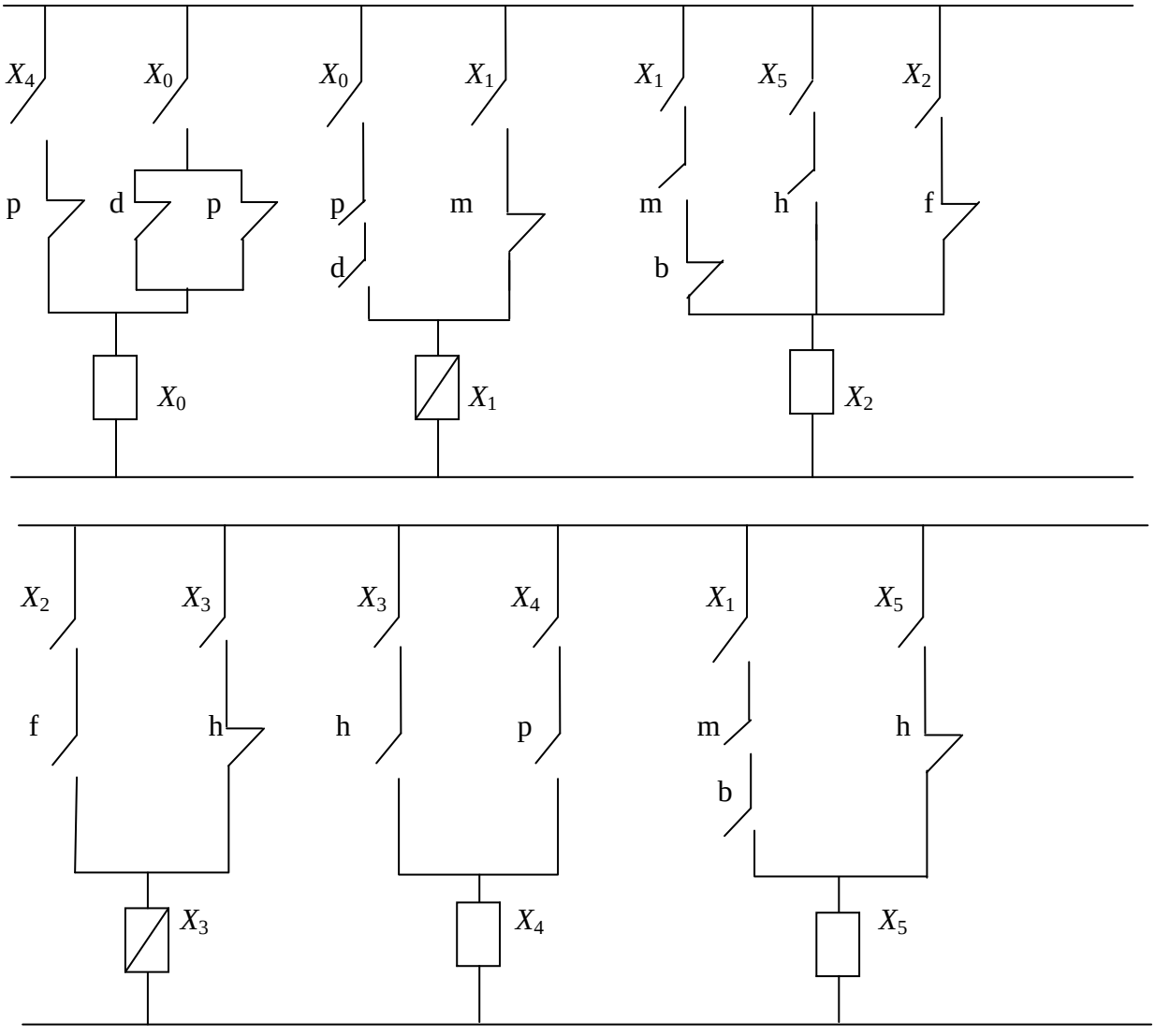
$D = X_1 + X_2$ (moteur de descente)

$M = X_3 + X_5$ (moteur de monté)

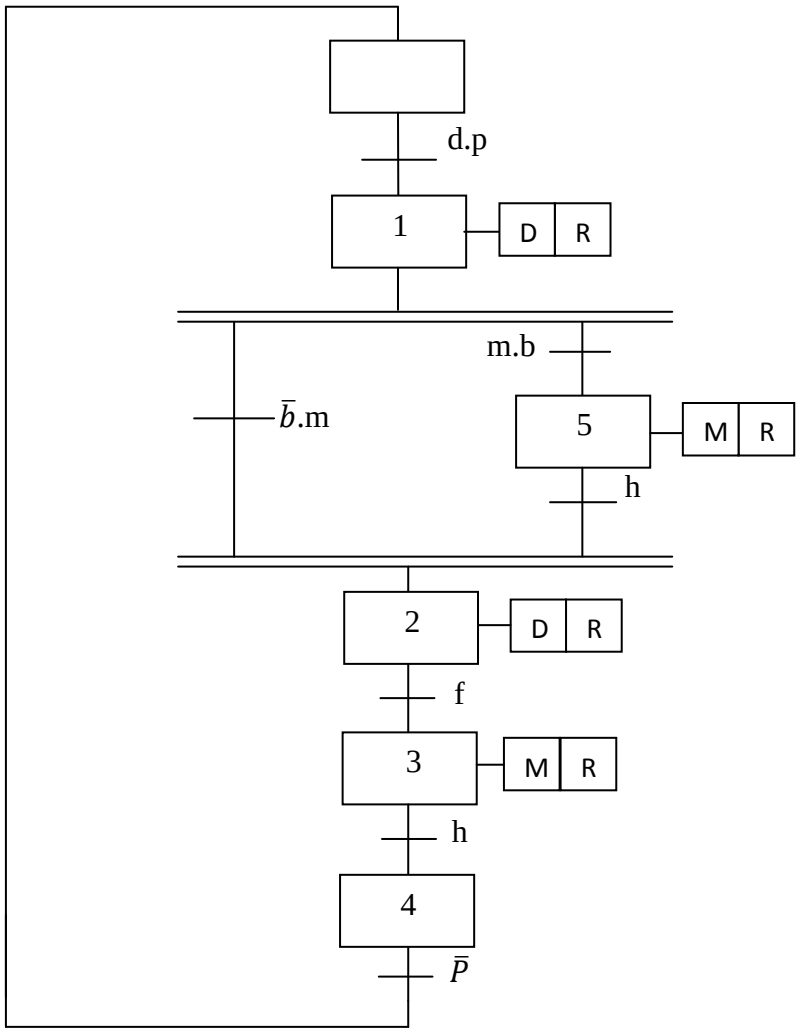
$R = X_1 + X_2 + X_3 + X_5$ (moteur de rotation)

- Réalisation avec les relais électriques:

La réalisation du réseau de Pétri avec les relais électriques est la suivante :



- Programme Grafcet:



Remarque : Le programme en Ladder de cet exemple est présenté en [**Annexe : D.1**].

- Table d'animation : la figure suivante présente la table d'animation de cet exemple :

Modification | Forcer | Chaînes étendues

Nom	Valeur	Type	Commentaire
b	0	BOOL	Contact hauteur de pièce
d	1	BOOL	Bouton poussoir de depart
f	0	BOOL	Contact bas de fin de course
h	0	BOOL	Contact haut de fin de course
Ini	0	BOOL	Initialisation
m	0	BOOL	Contact à mi course
p	0	BOOL	Contact présence de pèce
D1	0	BOOL	Moteur de descente
M1	0	BOOL	Moteur de monté
R	0	BOOL	Moteur de rotation

Figure II.21 : Simulation avec une table d'animation

II.13.2. Simulation d'un dispositif de production d'un melange

▪Cahier de charge : le système consiste en trois bacs, chacun d'eux posséde deux niveau(niveau bas L et niveau haut H) et un bac colecteur qui possède trois niveau N_1,N_2 et N_3. Le système comporte aussi un mélangeur et un moteur pour l'entrénner.

- Cycle de fonctionement :

En apuyant sur un bouton m et si le niveau du collecteur est entre N_1 et N_2 et si le niveau de chaque bac est au dessus du niveau bas, on demmare le process. On ouvre la vanne EV1(bac 1) jusqu'à L_1 ensuite on ouvre la vanne EV2 (bac 2) jusqu'à L_2. On demmare le moteur M à petite vitesse pendant 20 mn, après on arrette M et on ouvre la vanne EV3 (bac 3) jusqu'à L_2. On lance ne temporisation de 5 mn puis on demmare le moteur M à grande vitesse pendant 20 mn et on arrete le système.

- Les variables :

▪Trois vanne EV1, EV2, EV3 et leurs voyants VEV1, VEV2 et VEV3.

▪Moteur M : -petite vitesse Pv et le voyant VPv.

-grande vitesse Gv et le voyant VGv.

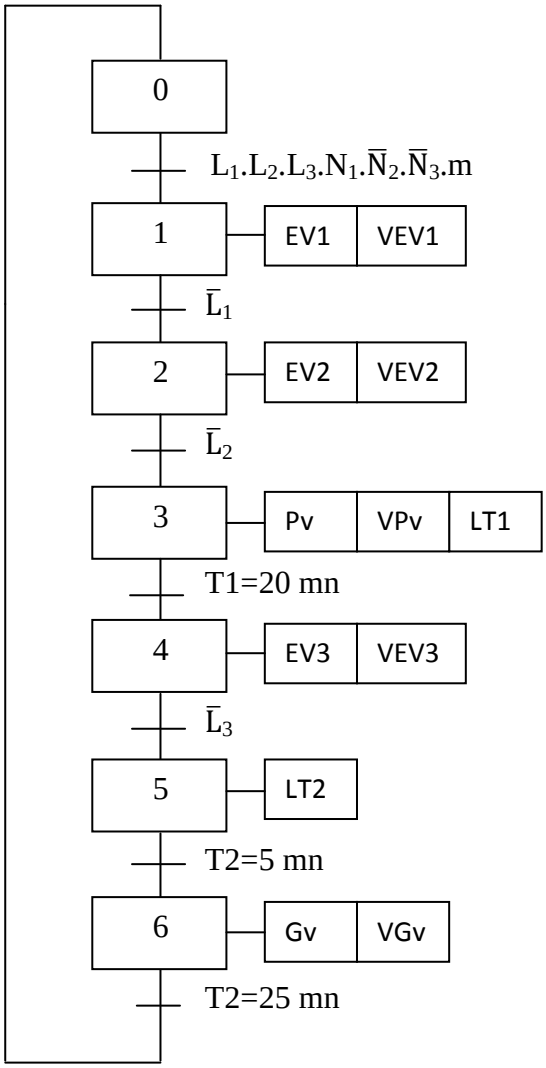
▪Niveau bas de chaque bacs (L_1, L_2, L_3).

▪Niveaux du bac collecteur N_1, N_2 et N_3.

▪Bouton m pour demmarer le process.

▪On a ajouter un voyant H qui indique que le niveau du collecteur est superieur à N_3.

- Programme en grafcet :



- Programmation sous Unity Pro :

▪Cette fois, on a utilisé un DFB pour le processus de remplissage des trois bacs et du collecteur et on a choisi le niveau 30 pour le déclenchement de L_1, L_2, L_3 et N_1, un autre DFB pour la vidange des bacs. On a choisi cette méthode parce que c'est la même condition de remplissage alors il suffit de crée un DFB et on lui fera appel (éviter la répétition des programmes) ensuite on a crée deux programmes : un pour le programme cyclique et l'autre pour l'appel des DFB.

▪Pour l'animation, on utilise seulement un écran d'exploitation, donc on utilise les objets proposés dans la barre d'outils ainsi que les objets qui se trouve dans la bibliothèque des objets des écrans d'exploitation, on peut accéder à cette dernière par la barre d'état « Outils ».

▪Le détail des programmes est présenté en [**Annexe : D.2**].

- L'écran d'exploitation du processus est :

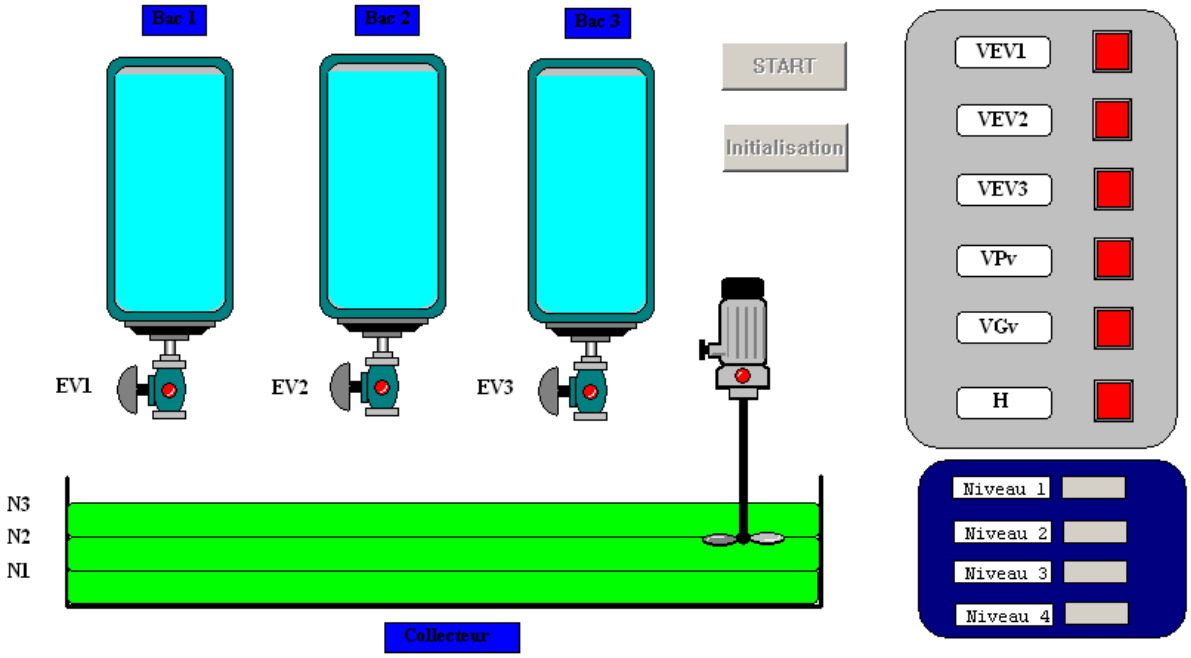


Figure II.22 : Simulation avec un écran d'exploitation

▪Pour la simulation, on introduit les niveaux : 1, 2, 3, 4 qui correspondent respectivement aux niveaux des bacs 1, 2 ,3 et du niveau du collecteur.

▪On utilise le bouton initialisation pour initialiser le programme cyclique et le bouton START pour démarrer le processus.

▪Pour que les valeurs entrées soient prises en comptes, il faut activer l'icône « valider l'écriture des variables » dans la barre d'outils.

II.14. Conclusion

Intégrant la totalité des langages et les outils nécessaires pour une programmation complète et souple, la gamme des logiciels Unity Pro offre une facilité d'utilisation et de compréhension.

Chapitre III

Logiciel Vijeo Citect

Chapitre III: Logiciel Vijeo Citect

III.1. Introduction [7]

Vijeo Citect est un système d'acquisition et de contrôle des données permettant de gérer et de surveiller des processus de fabrication, de production primaire, de distribution et de gestion d'installations, Il dispose d'outils de configuration puissants et simples à utiliser qui permettent de développer et de déployer rapidement des solutions pour des applications diverses.

Vijeo Citect se démarque par ses fonctionnalités uniques comme la redondance native, l'évolutivité et la fléxibilité, il a été conçu, dès l'origine, pour prendre en charge tous les besoins, quelle que soit la taille de l'application, de la plus petite application à la plus complexe, dans un système unique ; tout en garantissant performance et fiabilité.

En exploitant les avantages des solutions Microsoft, Vijeo Citect contribue à réduire les coûts d'acquisition, de déploiement et de gestion des systèmes de contrôle industriel à grande échelle.

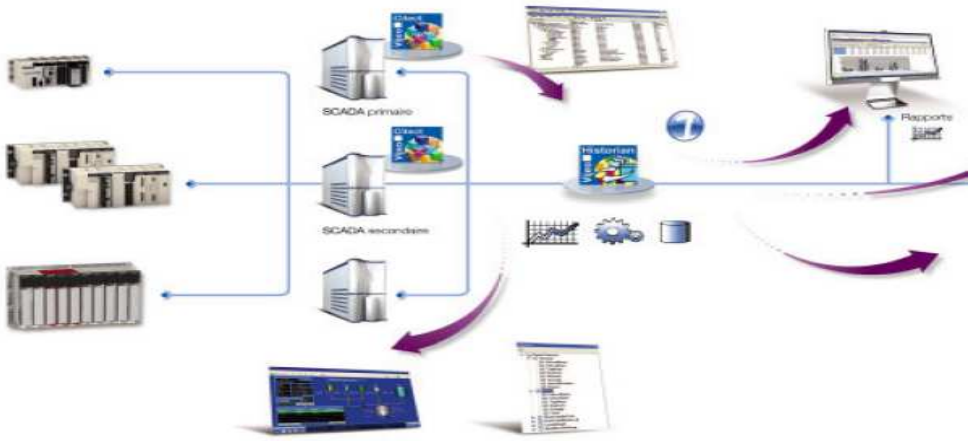


Figure III.1: Différente taches de Vijeo Citect

III.2. Utilisation de Vijeo Citect [8]

III.2.1. Besoins matériels

Pour exécuter Vijeo Citect version 6.1, le hardware minimum que doit avoir l'ordinateur est :

- Windows 2000-Pentium, processeur 500MHz avec 128 MB de RAM ;
- Windows XP ET Windows XP SP2-Pentium, processeur 500MHz avec 128 MB de RAM;
- Windows Server 2003-Pentium, 500MHz avec 256 MB de RAM.

▪La configuration préférable (recommandée) pour l'exécution de la même version de Vijeo Citect est :

- Windows 2000-Pentium, processeur 1 GHz avec 512 MB de RAM ;
- Windows XP ET Windows XP SP2-Pentium, processeur 1GHz avec 512 MB de RAM;
- Windows Server 2003-Pentium 1GHz avec 512 MB de RAM.

▪Pour le software nécessaire:

- Windows XP ET Windows XP SP2.
- Windows 2000.
- Windows Server 2003.

III.2.2. Projet

Un projet est une représentation numérique d'une installation de production, déployée en parallèle avec l'infrastructure pour permettre le suivi et le contrôle en temps réel du système. Donc Les graphiques, commandes, données de configuration et programmes associés à une installation Vijeo Citect sont configurés et implémentés dans des projets.

III.2.3. Eléments d'un projet

Les éléments incorporés dans un projet Vijeo Citect se répartissent logiquement entre les catégories suivantes :

- Éléments graphiques;
- Tags;
- Alarmes;
- Éléments système;
- Éléments de communication;
- Cicode / CitectVBA.

▪ Ces catégories sont représentées dans l'Explorateur Citect par le jeu de dossiers associé à chaque projet lors de sa création:

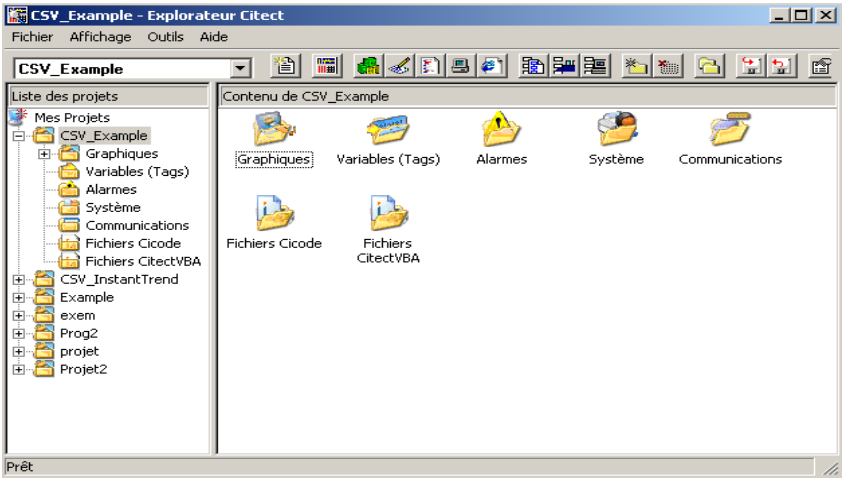


Figure III.2 : Eléments d'un projet Citect

▪Les éléments définis pour un projet Vijeo Citect sont ajoutés au fur et à mesure dans le dossier correspondant. La sélection d'un objet dans un de ces dossiers lance l'élément sélectionné dans l'outil nécessaire pour modifier ses propriétés.

III.2.3.1. Eléments Graphiques

Les éléments graphiques d'un projet Vijeo Citect sont utilisés pour créer les écrans affichés sur les clients de visualisation. Ils se composent de :

• Pages :	Base des écrans ;
• Modèles :	Ensemble de pages utilisées pour standardiser les écrans ;
• Symboles :	Objets graphiques stockés dans une bibliothèque en vue de leur réutilisation ;
• Génies :	Objets groupant plusieurs éléments graphiques et fonctionnels pour faciliter leur duplication ;
• Super Génies :	Génies pouvant transmettre des données spécifiques à un périphérique en phase d'exécution.

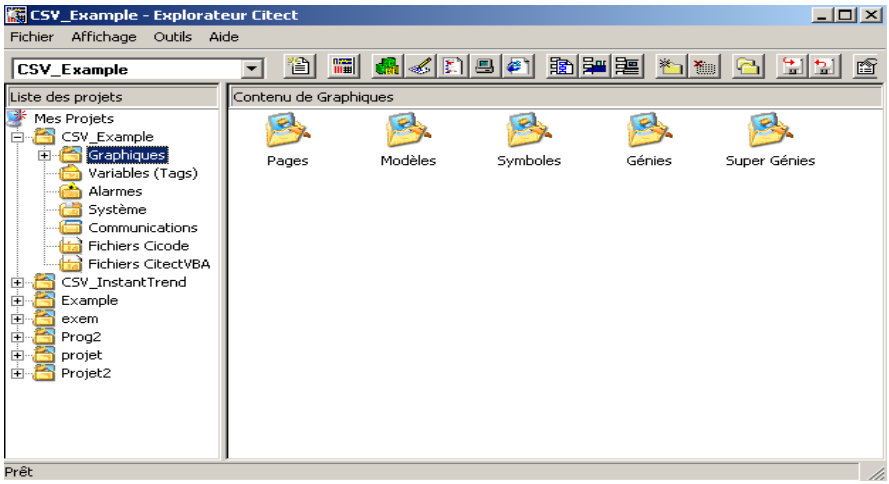


Figure III.3 : Eléments graphiques

▪Lorsque on crée des pages de projet dans l'éditeur graphique, les éléments inclus sont ajoutés dans le sous-répertoire correspondant du dossier Graphiques du projet courant.

▪Il faut noter que Vijeo Citect dispose d'une bibliothèque assez riche pour mieux représenter les diverses applications qui se trouve dans l'industrie.

III.2.3.2. Tags

Les tags permettent d'identifier les points de l'infrastructure qu'on vœux surveillez et contrôlez avec Vijeo Citect. Le nom qu'on donne à un tag devient une étiquette pour une adresse de registre, de sorte qu'il puisse être intuitivement appliqué aux pages graphiques et en cas d'alarme.

▪Il faut noter que vijeo Citect offre une manière d'affectation de noms aux variables mais cela reste facultatif [**Annexe : C.1**].

▪La déclaration des différents types de variable est illustré en [**Annexe : C.1**].

▪Trois types de tags sont inclus dans le dossier Tags de l'Explorateur Citect :

➢ Tags de variables :	Associés à des adresses de registre ;
➢ Tags de tendance :	Associés à des tendances de données ;
➢ Tags SPC :	Associés aux principes de la maîtrise statistique des processus.

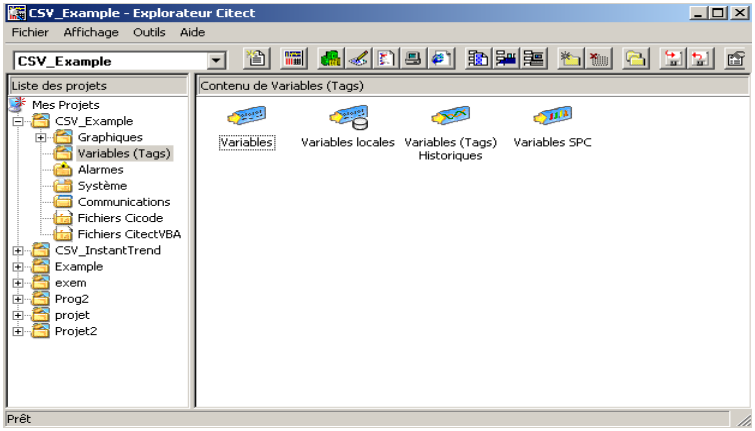


Figure III.4 : Tags

III.2.3.2.1. Variable locale

Les variables locales permettent de stocker des données en mémoire lorsqu'on lance le système superviseur. Elles sont crées au démarrage du système et ne gardent pas leur valeur lorsqu'on l'arrête. Les variables locales doivent être d'un type pris en charge par Vijeo Citect, y compris les tableaux à 2 dimensions et tous les types Vijeo Citect standard à l'exception des chaînes.

Les variables locales sont utiles lorsque chaque processus doit avoir une copie des données. Chaque processus a sa propre copie de chaque variable locale configurée dans le projet et les valeurs de la variable locale sont uniquement disponibles dans le processus qui les a écrites.

III.2.3.2.2. Variable de tendance

Le système de tendance permet de mieux comprendre les performances de l'installation et des équipements. Il peut fournir une analyse visuelle dynamique (graphiques de tendances et statistiques), des données de production et un suivi régulier de l'état des équipements, à des fins d'évaluation du rendement ou de maintenance préventive.

Les tags de tendances permettent de spécifier les données qu'on souhaite recueillir des périphériques d'E/S. Ces données peuvent être enregistrées à intervalles réguliers (tendance périodique) ou uniquement lorsqu'un événement se produit (tendance sur événement). Les tendances sur événement sont utilisées pour établir des tendances concernant des données non temporelles, par exemple, un produit sortant d'une ligne d'assemblage. Les données de tendances sont généralement sauvegardées sur disque pour une analyse ultérieure ou affichées dans un graphique de tendance.

Vijeo Citect peut collecter et enregistrer un volume de données quelconque. La capacité du disque dur est la seule limite en la matière.

III.2.3.2.3. Variable SPC

Les variables SPC désignent des données devant être recueillies à des fins statistiques. Une fois définies, les données peuvent être analysées dynamiquement en phase d'exécution (pour générer des diagrammes SPC et des alarmes). Les variables SPC sont semblables aux variables de tendances.

Lors de la définition d'une variable SPC, on doit impérativement entrer la limite supérieure spécifiée (USL) et la limite inférieure spécifiée (LSL) si on compte analyser la capacité d'un processus. Ces valeurs devraient être fidèles au cahier des charges du client et la valeur cible devrait être à mi-chemin entre les deux. Si on ne renseigne pas ces champs, l'analyse de la capabilité sera dépourvue de sens.

III.2.3.3. Les Alarmes

Les alarmes permettent d'identifier des conditions dans un système Vijeo Citect nécessitant une intervention. Vijeo Citect prend en charge sept types d'alarmes :

- Numérique ;
- Analogiques ;
- Horodatées ;
- Avancées ;
- Multi numériques ;
- Numériques horodatées ;
- Analogiques horodatées.

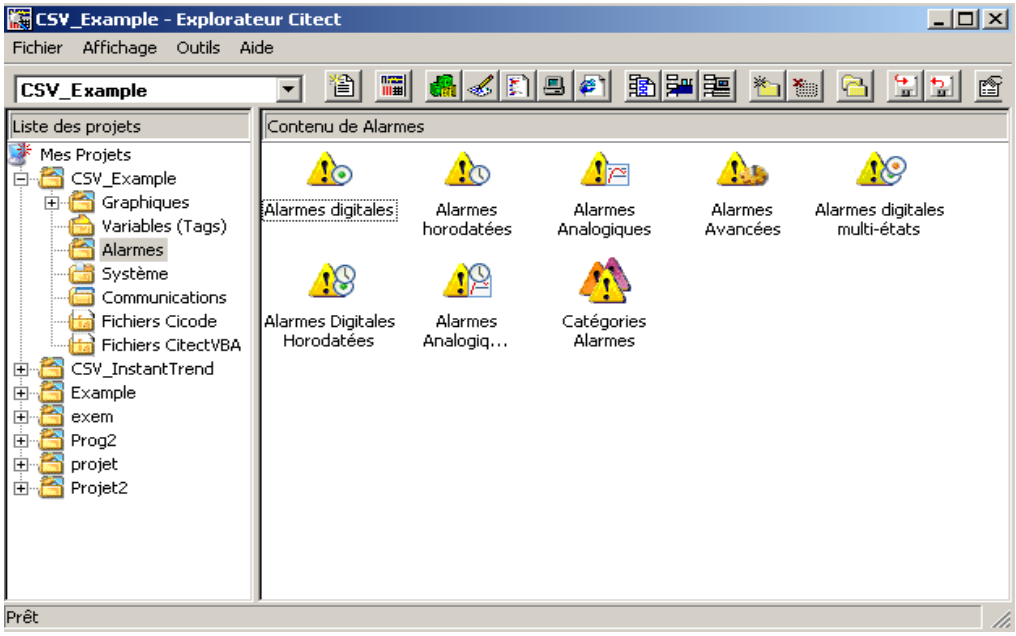


Figure III.5 : Différentes alarmes

III.2.3.3.1. Alarmes Numériques

On peut activer des alarmes numériques en fonction de l'état d'une ou de deux variables numériques. L'alarme devient active lorsque l'état de la condition de déclenchement reste vrai pendant l'intervalle de temps défini par ce paramètre.

III.2.3.3.2. Alarmes Analogiques

Les alarmes analogiques sont déclenchées lorsqu'une variable analogique dépasse une ou plusieurs valeurs spécifique(s). Chaque alarme peut être une combinaison des types suivants :

- alarme haute et très haute - lorsque la valeur est atypiquement haute ;
- alarme basse et très basse - lorsque la valeur est atypiquement basse ;
- alarme de déviation - lorsque la valeur s'écarte d'un point prédéfini ;
- alarme de rythme de changement - lorsqu'un important changement de valeur intervient dans un intervalle de temps donné.

III.2.3.3.3. Alarmes Horodatées

Les alarmes horodatées sont semblables aux alarmes numériques si ce n'est qu'elles utilisent un compteur pour dater les conditions de déclenchement et non pas uniquement l'heure à laquelle la variable a été interrogée. Elles peuvent uniquement être associées à une seule alarme numérique.

III.2.3.3.4. Alarmes Avancées

Une alarme avancée devient active lorsque le résultat d'une expression Cicode change. Vijeo Citect interroge l'expression à la fréquence définie par le paramètre Citect.ini [Alarm] ScanTime et recherche un changement d'état. Un avis de changement d'état est alors envoyé.

III.2.3.3.5. Alarmes Multi-Numériques

Les alarmes multi-numériques utilisent la sortie de trois variables numériques (par exemple : tags A, B, et C) pour définir huit états. Les états représentent toutes les combinaisons possibles de valeurs vrai/faux que les variables peuvent avoir.

Les valeurs de tag dans chaque état sont représentées dans l'ordre Tag C, Tag B, Tag A. Une valeur vraie est représentée par la lettre du tag et 0 (zéro) représente l'état faux.

Lors de la configuration des propriétés des alarmes multi-numériques, vous pouvez définir les états qui déclenchent une alarme ainsi que les fonctions Cicode à appeler lorsque les alarmes deviennent actives ou inactives.

III.2.3.3.6. Alarmes Numériques Horodatées

Les alarmes analogiques et numériques horodatées diffèrent des autres alarmes en ce qu'elles n'interrogent pas des variables pour déterminer des conditions d'alarme. Le serveur d'alarme est informé de tout changement de valeur d'une variable spécifiée grâce à la fonction Cicode [Alarm] NotifyVarChange.

Il utilise ces informations pour actualiser les alarmes qui suivent la variable. Ce processus permet d'associer une date et une heure précises à une condition d'alarme.

III.2.3.3.7. Alarmes Analogiques Horodatées

▪Mêmes fonctions que les alarmes numériques horodatées.

- **Catégories d'alarmes**

Dans la figure **III.5**, on trouve une icône nommé catégories d'alarmes. Cette dernière permet de configurer des options pour les différentes alarmes décrites précédemment.

Chaque alarme du système peut être affectée à une catégorie, et chaque catégorie peut être traitée comme un groupe. Pour chaque catégorie, on peut définir les caractéristiques de l'affichage (police, type de page), de l'enregistrement (imprimante ou fichier), et l'action à entreprendre lors du déclenchement d'une alarme (par exemple, activer une alarme sonore).

On peut également personnaliser l'ordre dans lequel les alarmes seront affichées sur la page de l'historique des alarmes à l'aide des paramètres SummarySort et SummarySortMode. (Cet ordre prime sur l'ordre de priorité des catégories d'alarmes).

Il est possible de définir jusqu'à 16 376 catégories d'alarmes. Si on ne spécifie pas de catégorie pour une alarme, elle aura les mêmes attributs que la Catégorie 0. Si on ne spécifie pas cette dernière, Vijeo Citect utilisera des valeurs par défaut.

Remarque : La configuration des différentes alarmes décrites précédemment est illustrée en [**Annexe : C.3**].

III.2.3.4. Eléments systèmes

Les éléments système d'un projet Vijeo Citect permettent de personnaliser, de gérer et de suivre le système superviseur. Ils comprennent:

- Touches clavier ;
- Commandes clavier ;
- Rapports ;
- Evénements ;
- Accumulateurs ;
- Périphériques ;
- Utilisateurs ;
- Groupes ;
- Macros ;
- Polices ;
- Paramètres ;
- Projets inclus.

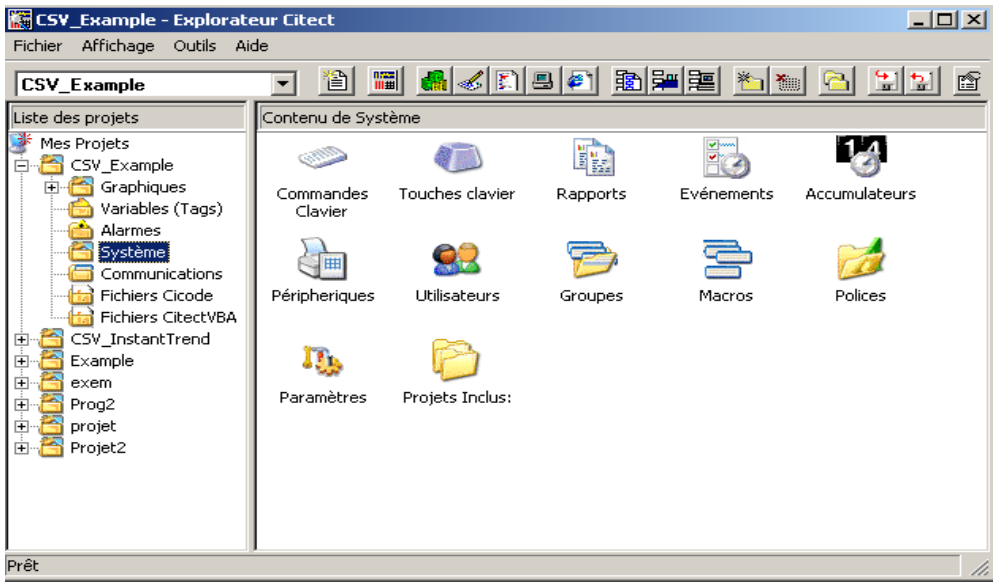


Figure III.6 : Eléments systèmes

▪La description ainsi que la configuration des ces différents éléments est illustré en [**Annexe :C.4**] .

III.2.3.5. Eléments de communication

Les éléments de communication d'un projet Vijeo Citect correspondent à la représentation configurée du matériel de communication du système. Ils comprennent :

- Ports ;
- Cartes ;
- Adresses réseaux ;
- Modems ;
- Périphériques d'E/S ;
- Remappage E/S ;
- Clusters ;
- Serveur d'E/S ;
- Serveurs d'alarmes ;
- Serveurs tendances ;
- Serveurs de rapport ;
- Assistant Express d'installation d'E/S.

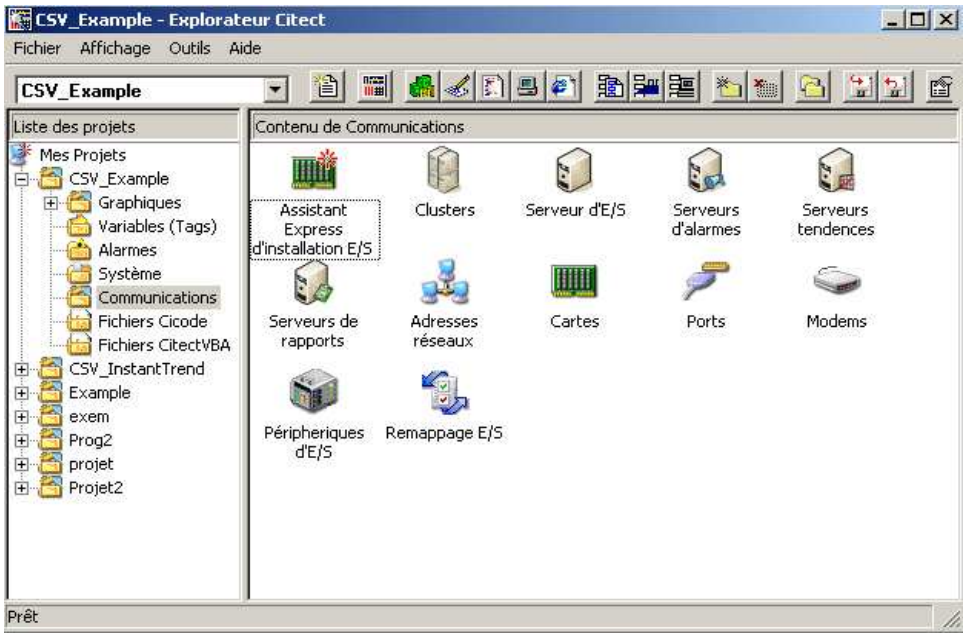


Figure III.7 : Eléments de communication

III.2.3.5.1. Ports

C'est la connexion physique entre la carte et le périphérique d'E/S, les propriétés d'un port dépendent du type de la carte installée dans le serveur d'E/S et du périphérique d'E/S connecté à ce port.

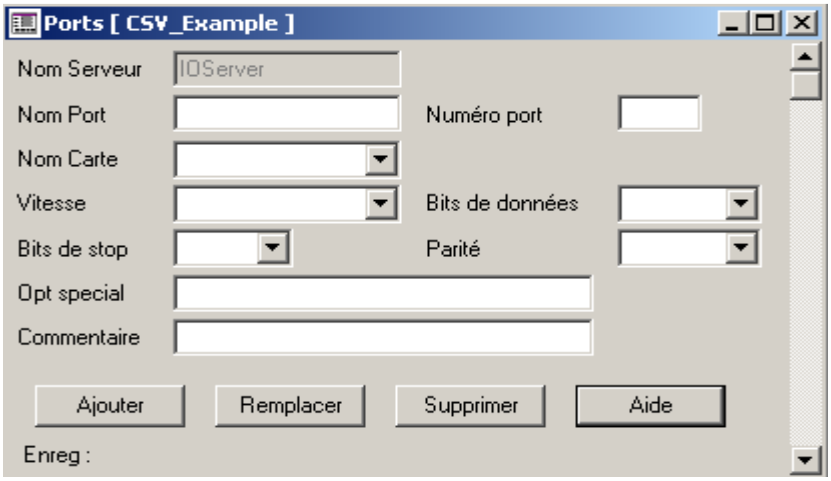


Figure III.8 : Ecran ports

▪Le champ Nom Port permet de donner un nom au port connecté aux périphériques d'E/S, chaque port doit avoir un nom propre à lui (maximum 16 caractères), le nom par défaut est : Board1_Port.

▪Le champ Numéro port permet de donner un numéro au port auquel le périphérique d'E/S est connecté, par exemple si on utilise le port COM de l'ordinateur on cherche le numéro dans la configuration de ce dernier.

▪Le champ Nom Carte permet de donner un nom à la carte installée.

▪Les Champs Vitesse, Bits de données, Bits de stop et Parité caractérisent la transmission de données (caractéristiques du port).

▪Le champ Option spéciale comme son l'indique caractérise toutes les options spéciales prises en charge par le port.

III.2.3.5.2. Cartes

C'est un composant permettant divers types de communication avec des périphériques d'E/S.

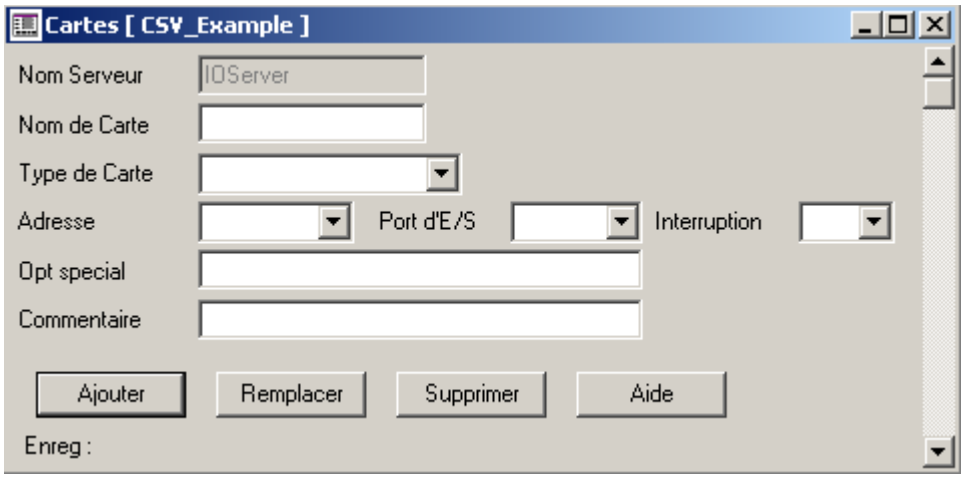


Figure III.9 : Ecran cartes

▪Pour la configuration, on doit associer un nom à la carte installée et on précise son type (TCPIP, PROFIBD,…), par exemple pour le port COM, son type est COMx.

▪On doit aussi spécifier l'adresse de début de la carte, si on n'utilise pas le port COM ou une carte série, l'adresse doit être 0.

▪ On introduit l'adresse du port d'E/S de la carte et on spécifie le numéro d'interruption utilisé par la carte (n'est pas nécessaire pour le port COM) et éventuellement on ajoute les options spéciales de la carte.

III.2.3.5.3. Adresses réseaux

Cette option permet de définir les réseaux existant et leur affecter leurs adresses correspondantes.

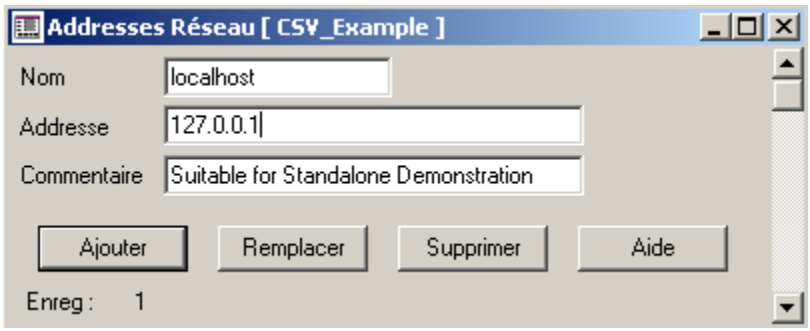


Figure III.10: Ecran adresses réseaux

III.2.3.5.4. Modems

L'écran Modems permet de configurer les composants utilisés pour connecter Vijeo Citect à un périphérique d'E/S distant accessible par numérotation.

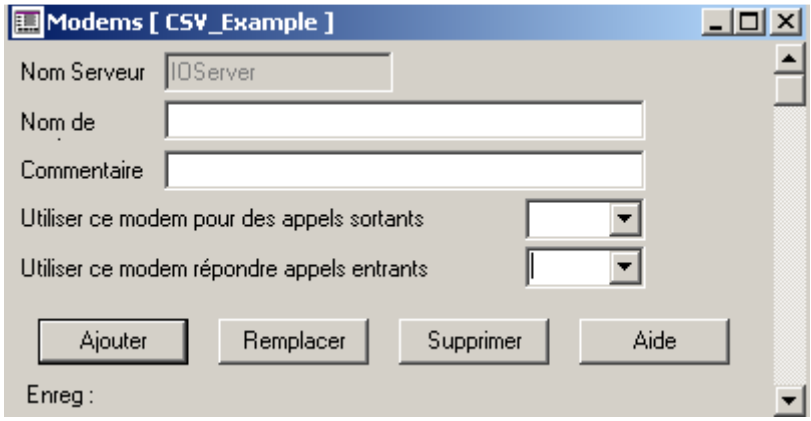


Figure III.11 : Ecran modems

▪On choisit un nom pour le modem (maximum 64 caractères), ce nom on le trouve dans la configuration de l'ordinateur, on doit spécifier son rôle soit pour lancer des appels aux périphériques, soit pour recevoir des appels de ces dernies.

III.2.3.5.5. Périphériques d'E/S

L'écran Périphériques d'E/S permet de configurer les appareils communiquant avec des équipements de contrôle ou de surveillance d'une installation industrielle.

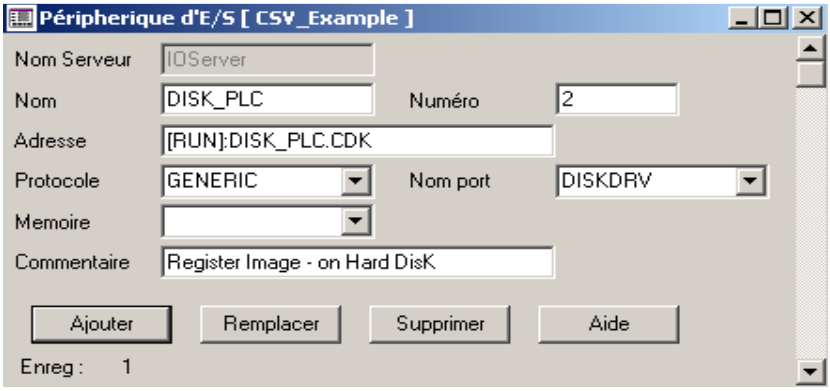


Figure III.12 : Ecran périphériques d'E/S

▪Pour la configuration, on donne un nom au périphérique et un numéro (entre 0 et 16383) qui doivent être unique pour chaque périphérique.

▪On introduit l'adresse du périphérique qui est déterminé par son type, le protocole utilisé pour communiquer avec lui.

▪Dans le champ Nom port, on introduit le nom du le port ou de la carte auquel le périphérique d'E/S est connecté, et enfin le champ Mémoire qui est utilisé pour spécifier si le périphérique est utilisé en mode mémoire ou non (option utiliser pour tester le périphérique).

III.2.3.5.6. Remappage d'E/S

C'est l'opération qui permet l'optimisation de la communication lors de la lecture de certaines valeurs numériques.

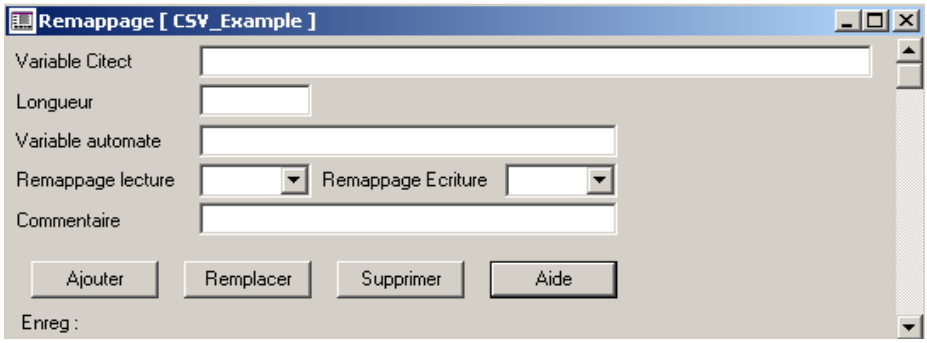


Figure III.13: Ecran remappage d'E/S

▪Dans cet écran, on spécifie la variable Citect (définie dans la base de donnée), sa longueur, la variable physique correspondante et le type de remppage (copiage) soit en lecture ou en écriture.

III.2.3.5.7. Clusters

Un cluster est un groupe logique de serveurs connectés à plusieurs machines physiques, donc c'est un groupe de serveurs d'alarmes, de tendances, de rapports et d'E/S. En général, il contient également des clients de visualisation Vijeo Citect locaux. Si l'installation comprend plusieurs sections ou systèmes, on peut utiliser plusieurs clusters à raison d'un par section.

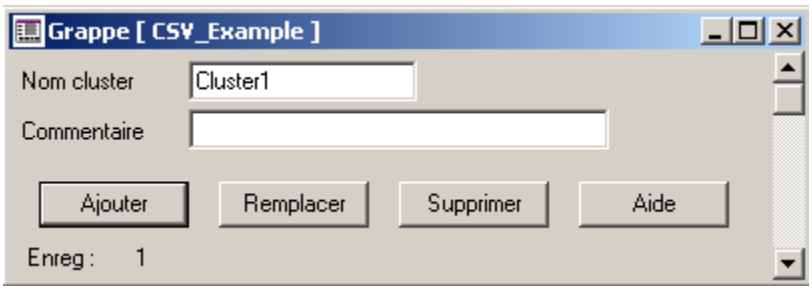


Figure III.14 : Ecran cluster (grappe)

▪Pour le définir, il suffit d'inscrire un nom et cliquer sur le bouton ajouter.

III.2.3.5.8. Serveurs d'E/S :

Ce sont des serveurs de communication spécialisés échangeant des données entre les périphériques d'E/S et les clients de visualisation Vijeo Citect.

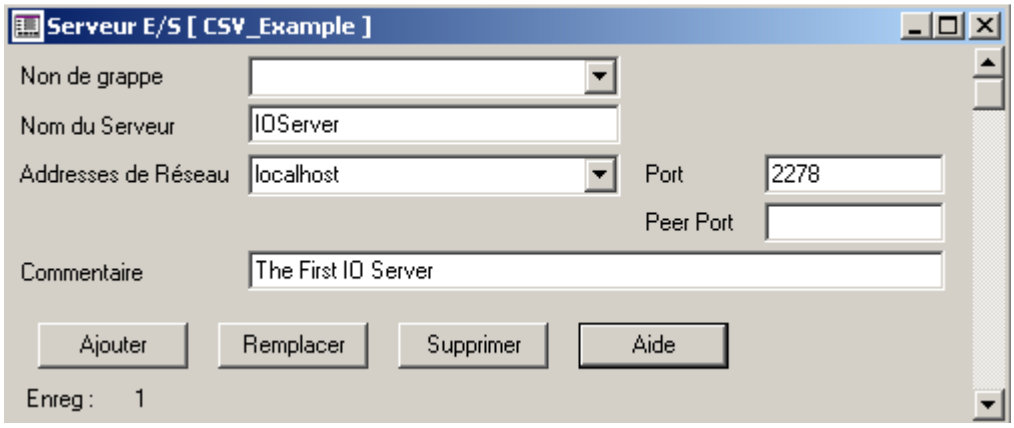


Figure III.15 : Ecran serveur E/S

▪Pour le définir, on lui donne un nom et on indique le cluster auquel il appartient. On spécifie aussi l'adresse réseau de la machine.

▪Le champ port permet d'insérer le numéro du port qui sera interroger par le serveur mais on peut le laisser vide et un numéro par défaut lui sera attribuer.

▪Le champ pree port est utilisé pour les communications entre les serveurs d'E/S pour fournir des informations actualisées sur les périphériques d'E/S.

III.2.3.5.9. Serveurs d'alarmes

Ce sont des serveurs surveillant les alarmes et les affichant sur les clients de visualisation appropriés.

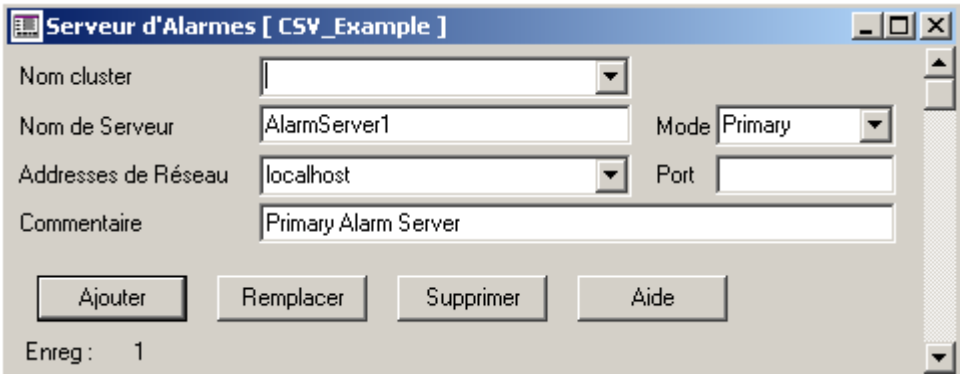


Figure III.16 : Ecran serveur d'alarmes

▪Pour le définir, on doit lui attribuer un nom et on indique le cluster auquel il appartient.

▪Le champ adresses de réseau permet d'insérer l'adresse de la machine ou son nom.

▪Le champ port permet d'insérer le numéro du port que le serveur interroge, on peut le laisser vide si on a un seul serveur et un numéro par défaut lui sera attribuer.

▪Le champ mode permet de spécifier le mode d'utilisation du serveur : primaire ou auxiliaire. Si ce champ est laissé vide, la valeur par défaut est « primaire ». Les serveurs primaires et auxiliaires doivent fonctionner sur des ordinateurs différents et seuls un serveur primaire et un serveur auxiliaire peuvent être définis par cluster.

III.2.3.5.10. Serveurs de tendances

Ce sont des serveurs contrôlant l'accumulation et l'enregistrement d'informations de tendances.

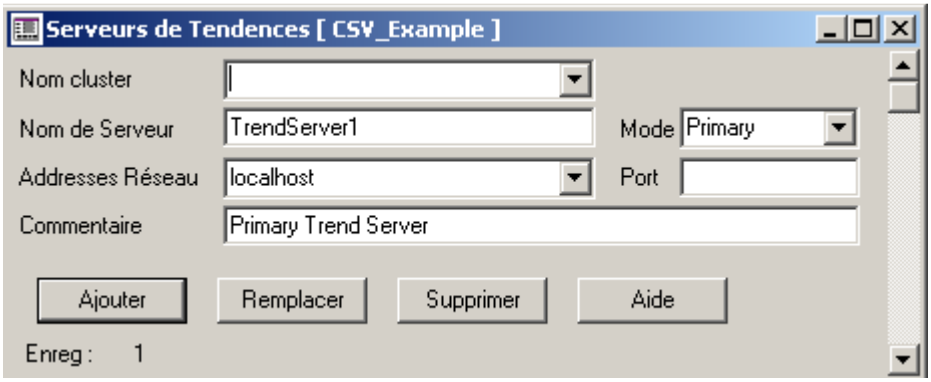


Figure III.17 : Ecran serveurs de tendances

▪Pour définir un serveur de tendance, on procède de la même manière que les serveurs d'alarmes.

III.2.3.5.11. Serveur des rapports

Ce sont des serveurs contrôlant le traitement des rapports.

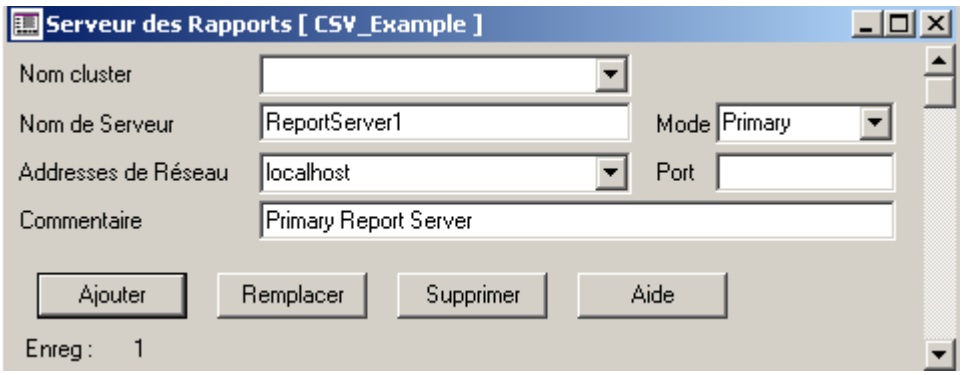


Figure III.18 : Ecran serveur des rapports

▪ Pour définir un serveur des rapports, on procède de la même manière que les serveurs d'alarmes.

III.2.3.5.12. Assistant express d'installation d'E/S

L'utilisation de cet assistant nous permet de configurer la communication avec les périphériques d'E/S et nous évite les étapes de configuration décrites précédemment concernant les éléments de communication. L'illustration de son utilisation sera décrite dans la partie suivante.

III.2.3.6. Fichiers Cicode/ Citect VBA

Vijeo Citect offre deux langages de programmation permettant de contrôler et de manipuler les éléments de Vijeo Citect qui sont :

- Le Cicode ;
- Le CitectVBA.

III.2.3.6.1. Fichiers Cicode

Le Cicode est un langage de programmation structuré conçu pour être utilisé avec Vijeo Citect pour la surveillance et le contrôle des équipements. Il s'agit d'un langage structuré similaire à Visual basic ou au langage C.

▪ Quelques propriétés de ce type de fichier sont présentées en [**Annexe : B.2**]

III.2.3.6.2. Fichiers Citect VBA

Le Citect VBA est un langage de script Basic de type Visual Basic pour Applications (VBA), dans la syntaxe est celle utilisée avec le langage Visual Basic.

III.3. Création d'un projet Vijeo Citect

Dans ce qui suit, on va essayer d'illustré les différentes étapes à suivre pour crée et exécuter un projet Citect, au lancement de l'explorateur Citect, il y'a trois fenêtres (éditeurs) qui apparaissent à savoir :

- Explorateur Citect ;
- Editeur de projet ;
- Editeur graphiques.

▪Il faut noter qu'on peut accéder à l'importe quel éditeur à partir d'un éditeur différent. Pour un premier temps, on va travailler en utilisant l'explorateur Citect et on illustre le fonctionnement des autres éditeurs à chaque fois que l'occasion se présente, l'explorateur est le suivant :

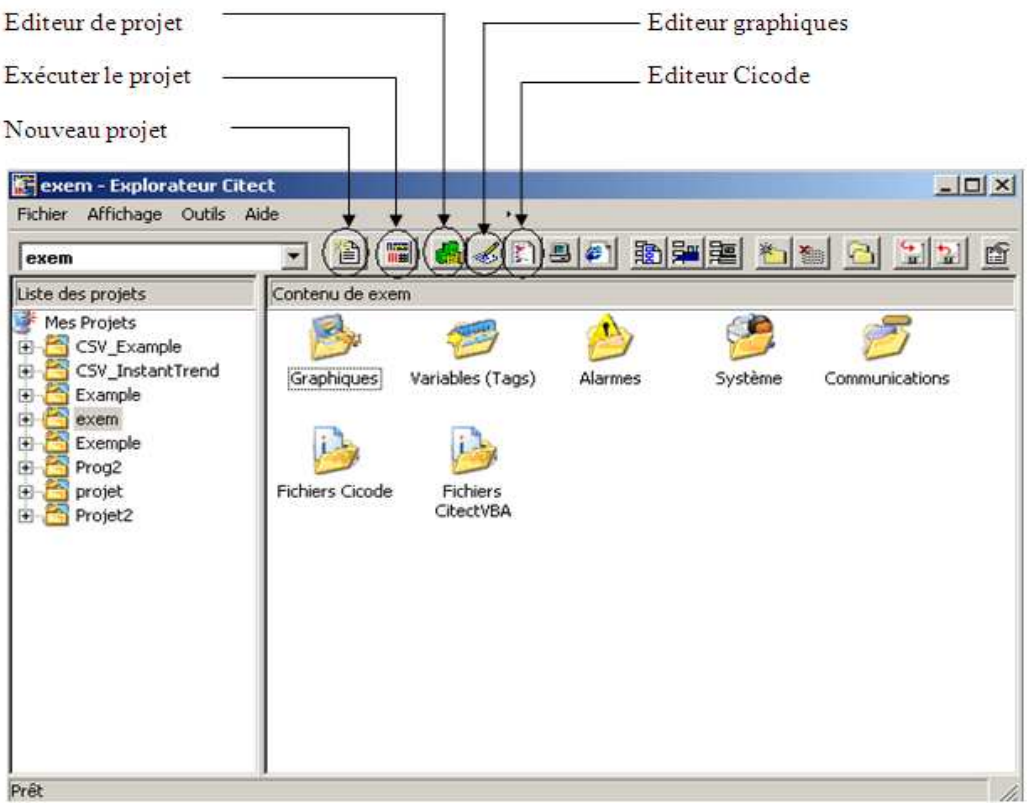


Figure III.19 : Explorateur Citect

▪Dans cette fenêtre : à gauche, on trouve la listes des projets crées et à droite, on trouve les dossiers associés à chaque projet (mêmes dossiers).

III.3.1. Etapes de création d'un projet

III.3.1.1. Création d'un nouveau projet

Pour crée un nouveau projet on clique sur le bouton nouveau et on aura la fenêtre suivante :

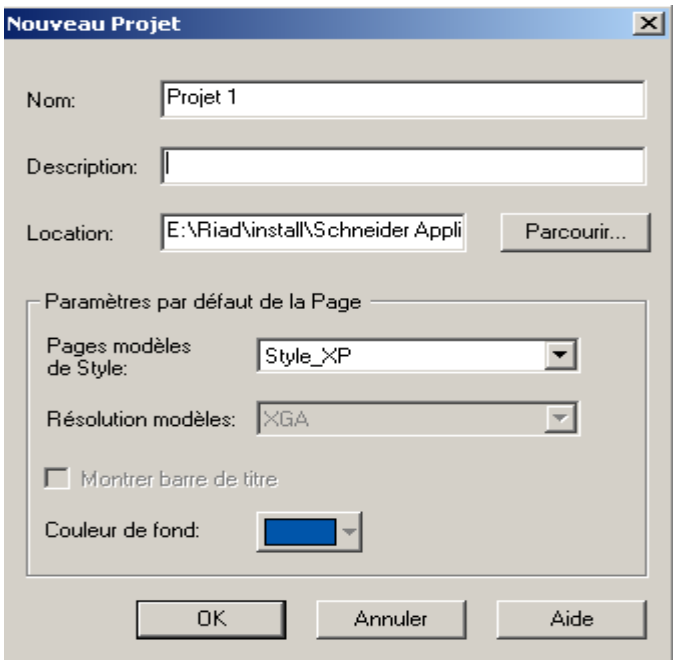


Figure III.20 : Ecran de création d'un nouveau projet

▪Dans cet écran, on choisi un nom du projet et on peut ajouter une description du projet.

▪On peut choisir l'endroit de sauvegarde du projet.

▪il y'a des champs pour les paramètres des pages, on utilise les paramètres par défaut.

▪Au moment ou on clic sur le bouton OK, le projet est inclus automatiquement dans la liste des projets dans l'explorateur Citect.

III.3.1.2. Création du Cluster

Après avoir crée le projet, on doit impérativement créer le cluster de travail, comme on l'a montré précédemment, on choisi le nom Cluster1.

III.3.1.3. Utilisation de l'assistant de communication

On l'utilise pour configurer la communication avec les périphériques d'E/S, il faut noter que cet assistant permet aussi de créer un serveur d'E/S et un périphérique d'E/S (besoin minimum pour échanger des données), les autres serveurs on les crée selon l'application. Qu'on lance l'assistant, la figure est la suivante :

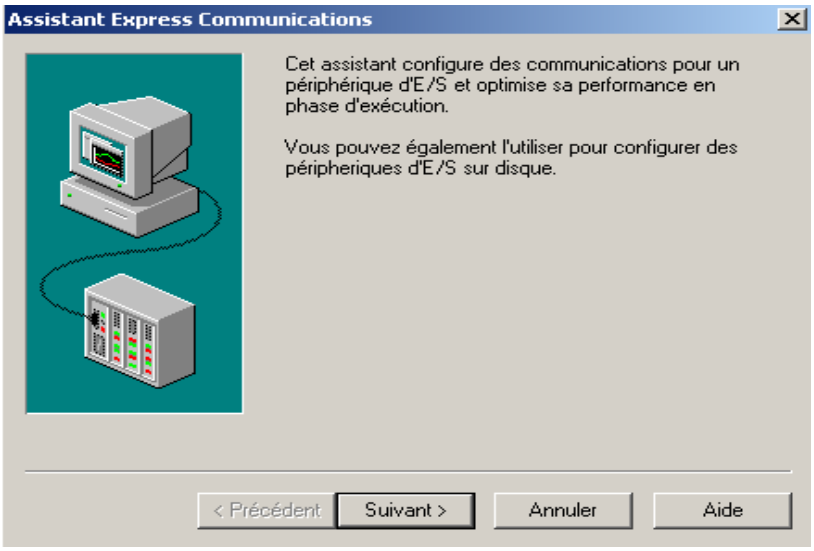


Figure III.21 : Ecran Assistant Express Communications_1

▪Il est clairement indiqué sur cette fenêtre le rôle de cet assistant, en cliquant sur le bouton suivant, on aura la figure :

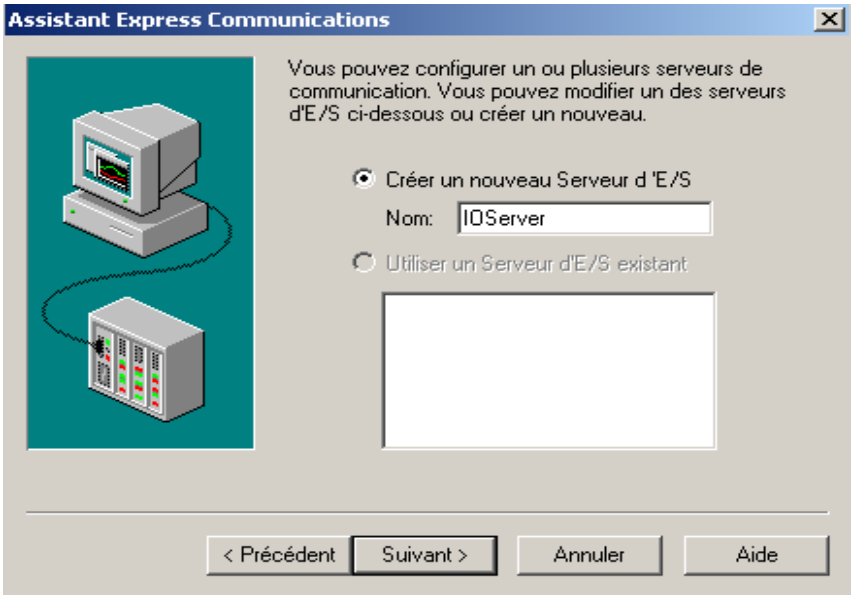


Figure III.22 : Ecran Assistant Express Communications_2

▪Cette fenêtre nous permet de créer un serveur d'E/S ou utiliser un serveur déjà crée au paravent, le nom choisi par défaut est IOServer mais on peut le changer, on clique sur le bouton suivant et on aura la figure :

Figure III.23 : Ecran Assistant Express Communications_3

▪Cette fenêtre permet créer le périphérique d'E/S, le nom par défaut est IODev mais on peut le changer, on continue en cliquant sur le bouton suivant et on aura la figure suivante :

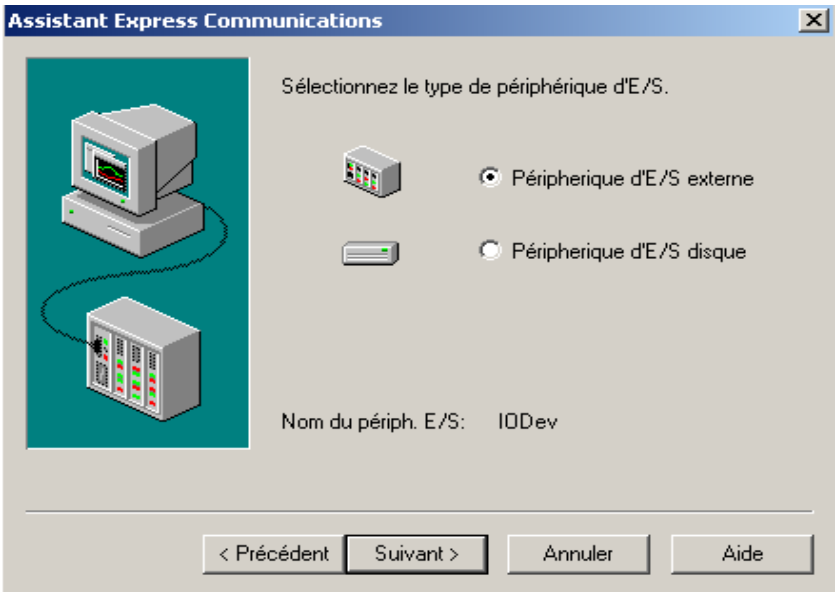


Figure III.24 : Ecran Assistant Express Communications_4

▪Cette fenêtre nous permet de choisir comme elle indique le type de périphérique d'E/S à savoir externe (automate), ou bien disque (mémoire de l'ordinateur).

▪L'utilisation des périphériques d'E/S disque est utile lorsque l'état de l'installation doit pouvoir être rétabli après un arrêt ou une panne du système. On peut configurer le système Vijeo Citect de façon à ce que les variables critiques définissant ces conditions internes soient continuellement enregistrées dans un périphérique d'E/S sur disque. Lorsqu'on redémarre le système après un arrêt ou une panne, Vijeo Citect peut rétablir immédiatement ces conditions.

▪On clique sur le bouton suivant et on aura la figure :

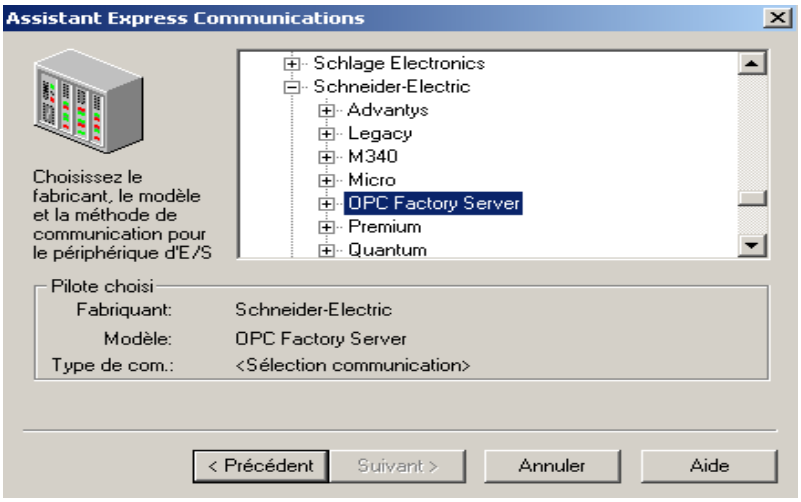


Figure III.25 : Ecran Assistant Express Communications_5

▪Cette fenêtre nous permet de choisir le type d'automate, selon le fabricant (Schneider Electric, Siemens, ABB,…) et aussi le protocole de communication entre Vijeo Citect et l'automate.

▪On a choisi pour la suite, la plate forme Quantum de Schneider et le protocole Modbus/TCP(Ethernet) et on clique sur le bouton suivant, on aura la figure:

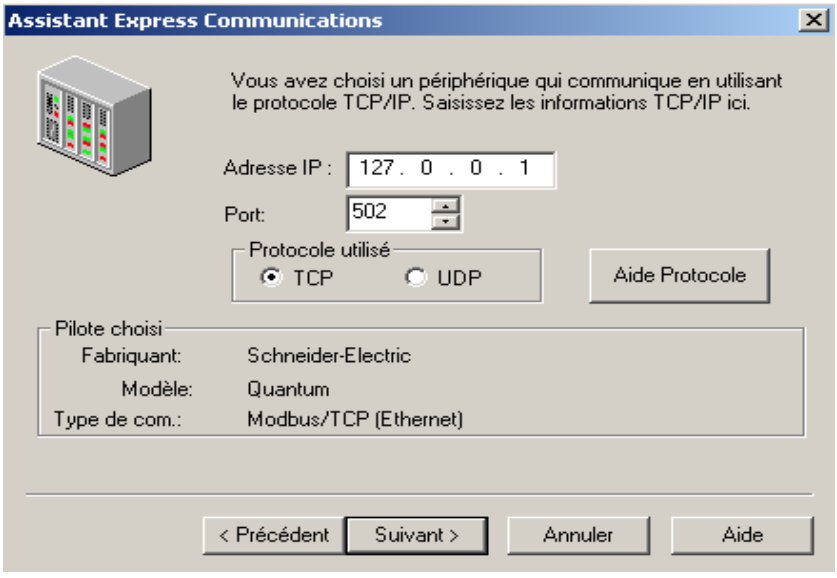


Figure III.26 : Ecran Assistant Express Communications_6

▪Comme le montre la fenêtre, on doit introduire l'adresse IP de l'automate, le nom du port de communication et le protocole utilisée.

▪Pour le protocole Modbus/TCP(Ethernet) : le nom du port est 502.

▪Pour nous, on choisi l'adresse IP : 127.0.0.1 qui est l'adresse du simulateur de Unity Pro qu'on utilisera par la suite, on continue en cliquant sur le bouton suivant et on aura :

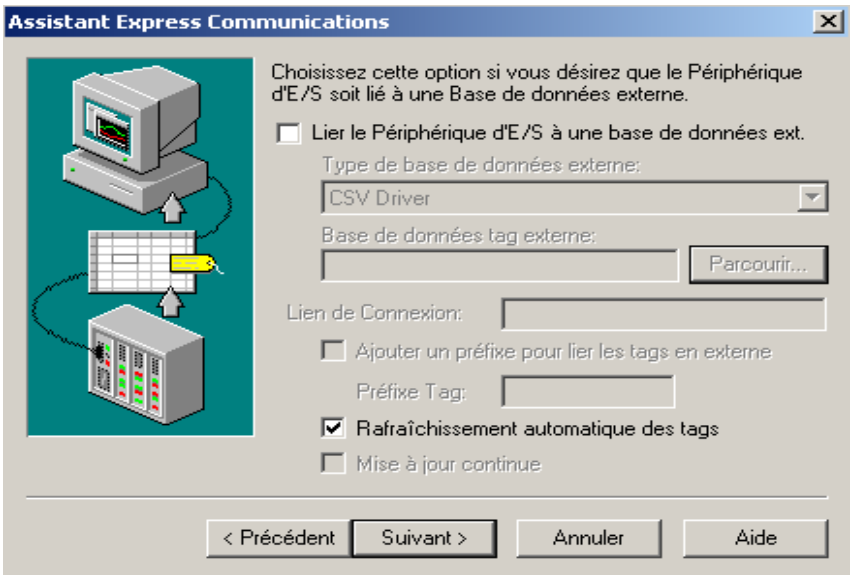


Figure III.27 : Ecran Assistant Express Communications_7

▪Cette fenêtre nous offre une option à savoir si l'automate est lié à une base de données par exemple SQLServer, alors on relié l'ordinateur ou se trouve le Citect à cette base.

▪Dans notre cas, on ne va pas utiliser des bases de données, alors on laisse la case vide ; il y'a une autre case qui indique le rafraichissement des variables est effectué automatiquement, on l'a laisse coché et clique sur suivant et on aura :

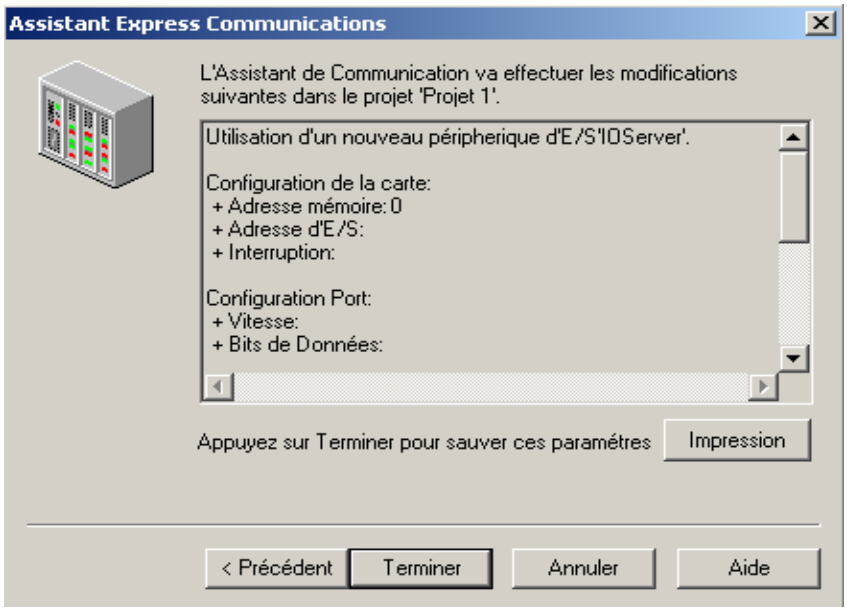


Figure III.28 : Ecran Assistant Express Communications_8

▪Cette fenêtre est une sorte de récapulatif de toutes les configurations faites précédemment, on clique sur le bouton Terminer pour la sauvegarde de ces paramètres et on peut revenir en arrière si on veut encore modifier.

III.3.1.4. Vérification de la configuration

On peut visualiser la configuration faite précédemment par l'assistant en ouvrant chaque écran (port, carte,…) et on verra les changements apportés mais avant ça on doit associer le serveur d'E/S au cluster de travail, chose qui n'est pas faite par l'assistant, pour ce faire : on ouvre le dossier communication et on clique sur le serveur d'E/S la figure suivante apparait :

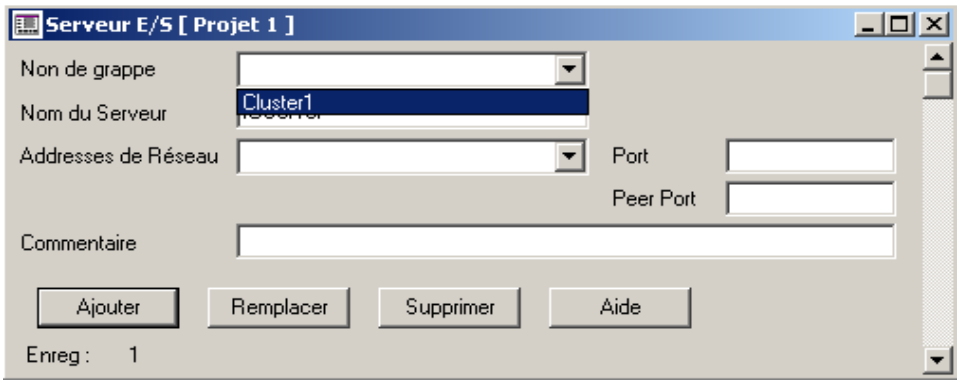


Figure III.29 : Association du serveur d'E/S au cluster

▪On choisi Cluster1, le seul qu'on a crée précédemment et on clique sur le bouton Remplacer pour enregistré les changements.

▪Pour la visualisation des changements apportés par l'assistant, les écrans sont les suivants :

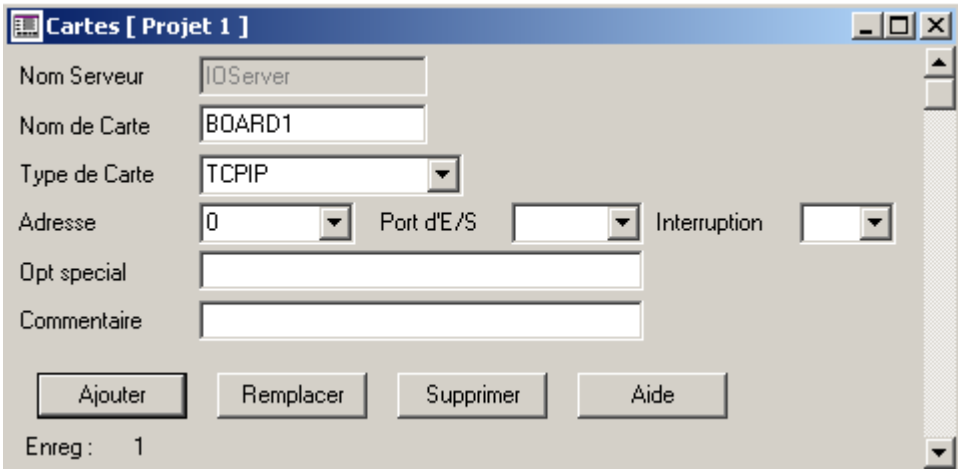


Figure III.30 : Ecran cartes

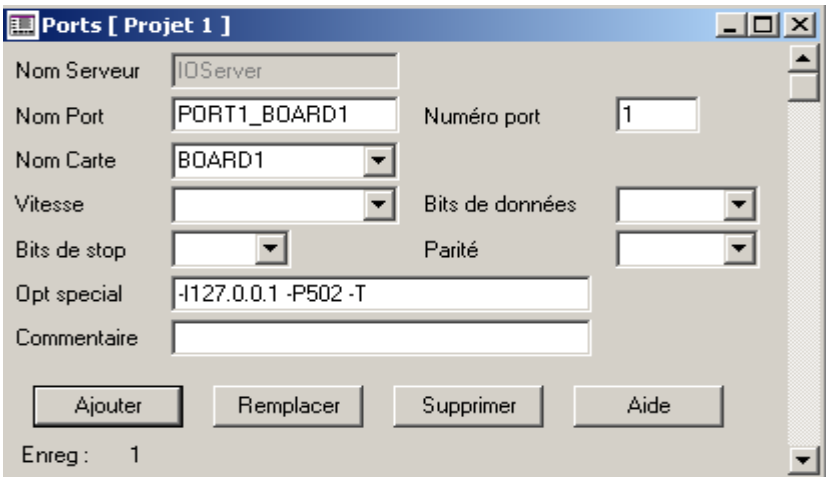


Figure III.31 : Ecran ports

▪Comme on le constate, les changements sont effectués automatiquement pour chaque équipement et Vijeo Citect s'en charge de remplir certains champs qu'on peut bien sur modifier par la suite (configuration minimale).

Remarques : l'assistant utiliser précédemment permet la configuration d'un seul périphériques d'E/S à la fois, s'il y'a plusieurs périphériques, on refait les étapes décrites précédemment pour chacun d'eux

▪A ce stade, on a crée le projet, il ne manque la création des pages graphiques et la déclaration des différentes variables.

▪S'il y'a une erreur de configuration, Vijeo Citect la signalera au moment de la compilation du projet, une opération qu'on montrera après.

▪On peut remarquer aussi que l'utilisation de cet assistant implique directement que le poste est au moins un serveur. Donc, pour les postes clients seulement, on ne pourra pas l'utiliser et pour configurer la communication, on doit passer par les éléments de communications.

III.3.1.5 : Editeur graphiques

Cet éditeur permet la création des graphiques représentant le processus, on peut accéder à cet éditeur de plusieurs manières comme on l'a déjà mentionné, la figure qui suit le représente :

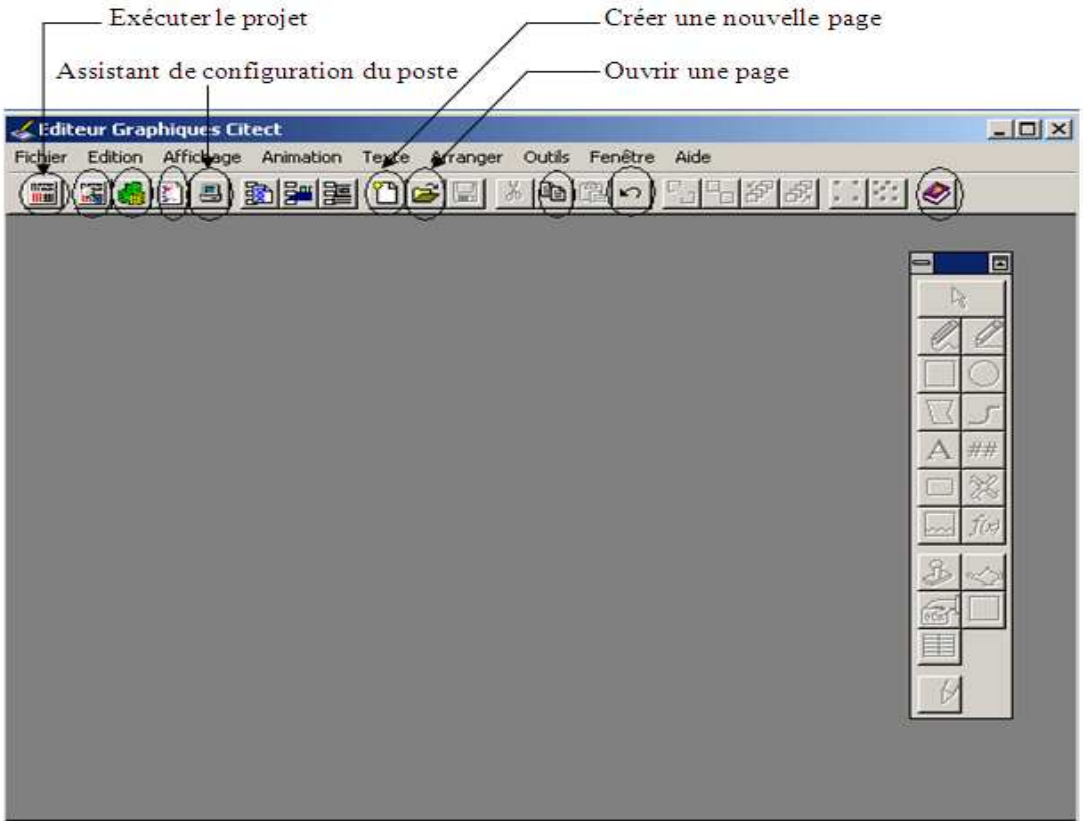


Figure III.32 : Editeur graphique

▪Pour créer une nouvelle page, on clique sur nouveau et on aura la figure :

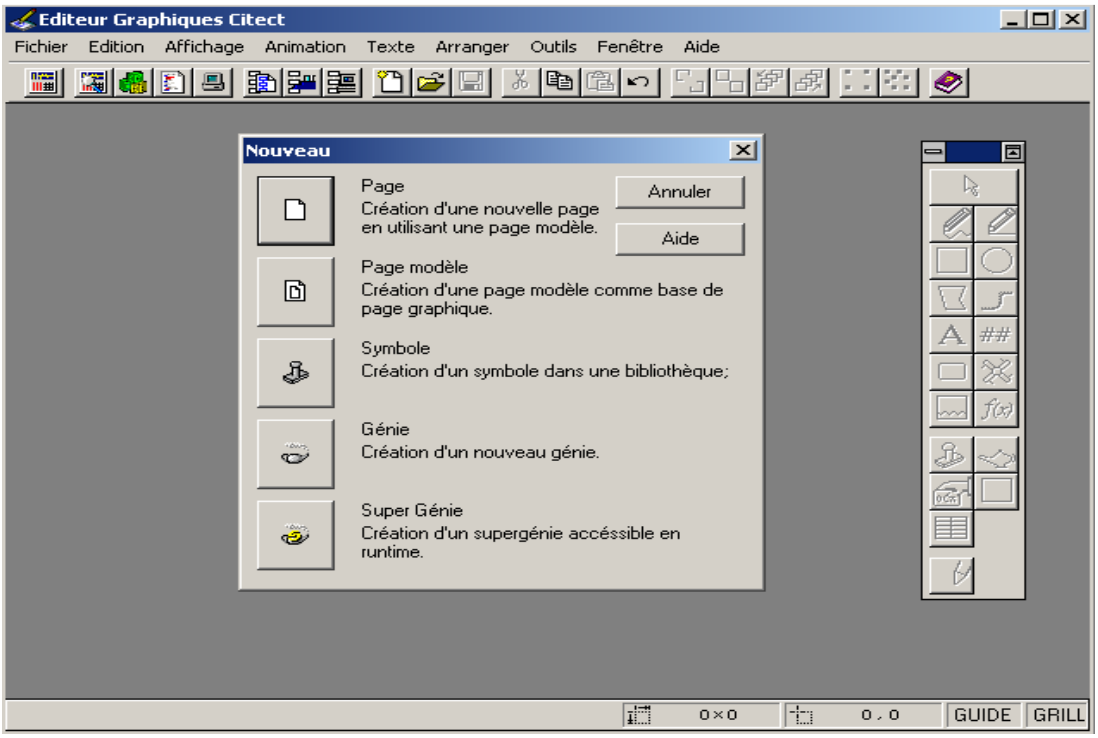


Figure III.33 : Créer une nouvelle page

▪Comme la fenêtre le montre, on peut créer des pages, des pages modèles, des symboles, des génies et des super génies ;

▪On commence par créer une page simple et on illustre les autres par la suite ;

▪Un clic sur le bouton page donne la figure suivante :

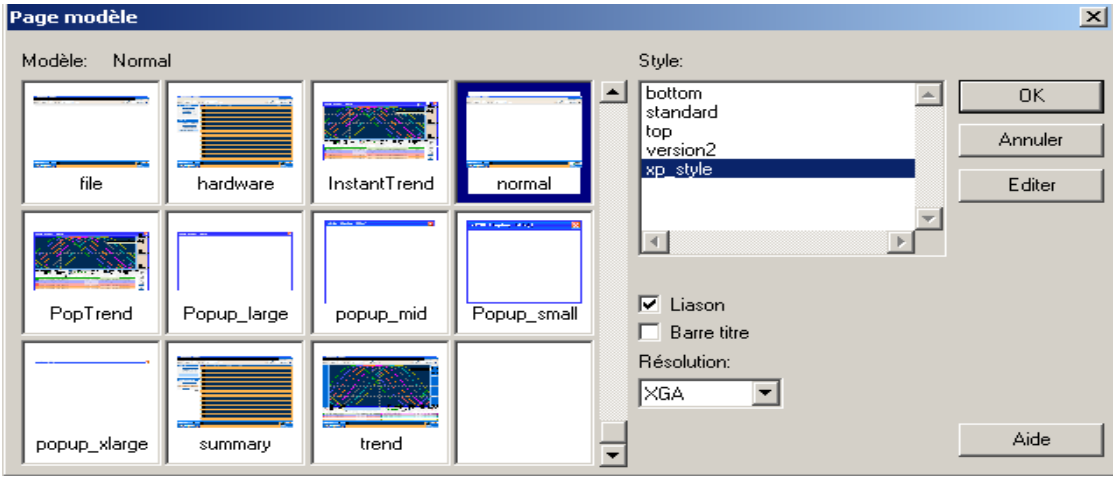


Figure III.34 : Paramètres de la page

▪Cette fenêtre nous permet de choisir les paramètres de la page (modèle, style,…) et ça dépend de l'application à développer, ce qui concerne notre cas, on choisi les paramètres affichés sur la figure à savoir : un modèle normal, style xp_style et la résolution XGA et on clique sur OK pour créer la page qui est la suivante :

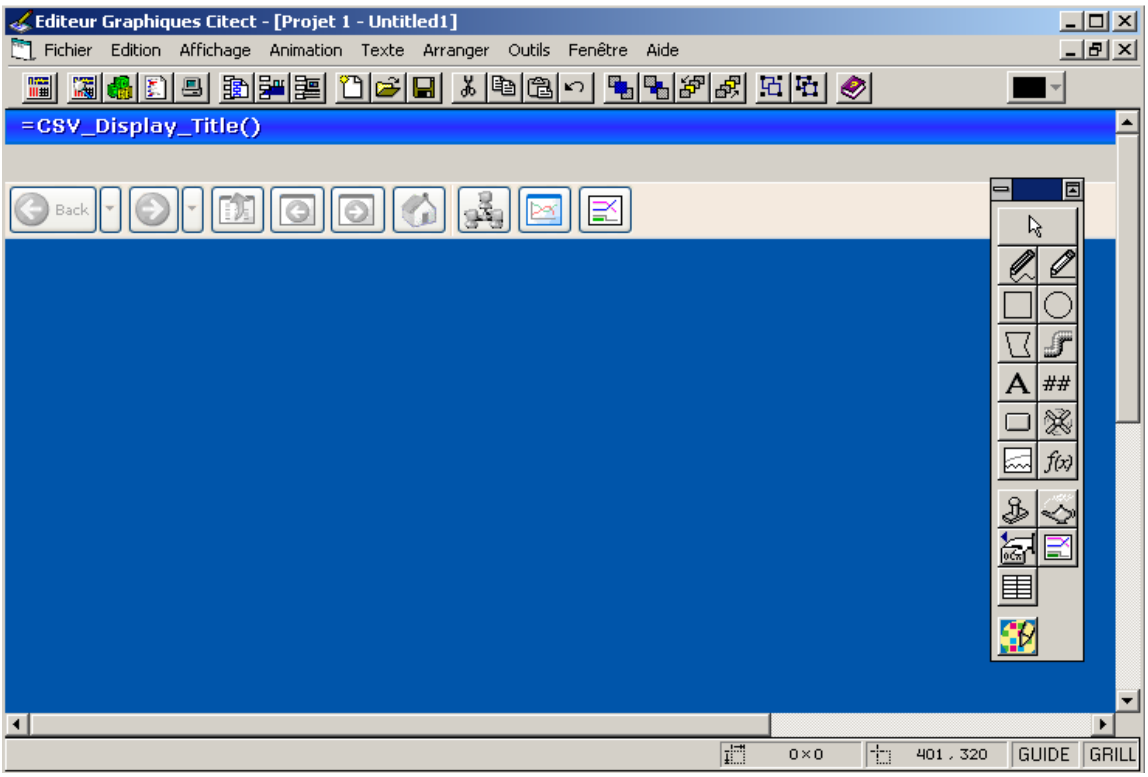


Figure III.35 : Page graphique

▪Cette figure montre la page créer, on trouve la barre d'outil et la barre d'état qu'on utilisera pour insérer les objets du graphique.

▪L'avantage d'utiliser ce type de page est que les pages d'alarmes et de tendance sont déjà crées et on a accès a eux par des icones sur la page, il suffit de faire les configurations nécessaire et de visualiser.

▪le tableau suivant illustre les détails des symboles de la barre d'état qu'on utilisera par la suite :

Symbole	Utilisation
	Curseur de sélection
	Utiliser pour dessiner des traits droits ou libres
	Utiliser pour dessiner des formes
	Utiliser pour insérer un texte
	Utiliser pour créer des boutons
	Utiliser pour insérer des objets de la librairie
	Utiliser pour les insérer les objets configurables
	Utiliser pour insérer des génies
	Utiliser pour l'affichage numérique

Tableau III.1 : Objets de la barre d'état

▪Après avoir crée les graphiques, on sauvegarde la page en lui spécifiant un nom.

▪Il faut noter qu'on peut créer autant de page qu'on veut selon l'application à développer.

III.3.1.6. Variables

Après avoir terminé le graphique, on déclare les variables et on les affecte aux différents objets configurables des pages crées et on utilise aussi les commande proposées par Vijeo Citect et enfin on compile le projet en utilisant l'éditeur de projet.

III.3.2. Assistant de Configuration du poste

On peut laisser cette étape à la fin, elle consiste à la configuration de l'ordinateur ou le projet crée sera exécuté, soit en client ou serveur….

Pour accéder à cet assistant, une petite icône qui représente un ordinateur et on l'a trouve dans les trois éditeurs comme le montre la figure II.44, l'utilisation de cet assistant est illustré en [**Annexe : B.5**].

III.4. Exemple d'application

On reprend le cahier de charge de l'exemple **II.13.2** mais au lieu d'utiliser l'écran d'exploitation pour la simulation comme on l'a déjà fait, on va utiliser Vijeo Citect pour la visualisation.

▪Pour la configuration de la communication, on utilise celle décrite dans la partie précédente.

▪Pour la saisie des niveaux d'entrés, on utilise une sorte de curseur que offre Vijeo Citect et on utilise l'éditeur graphique pour développer le graphique du processus, on aura la représentation suivante :

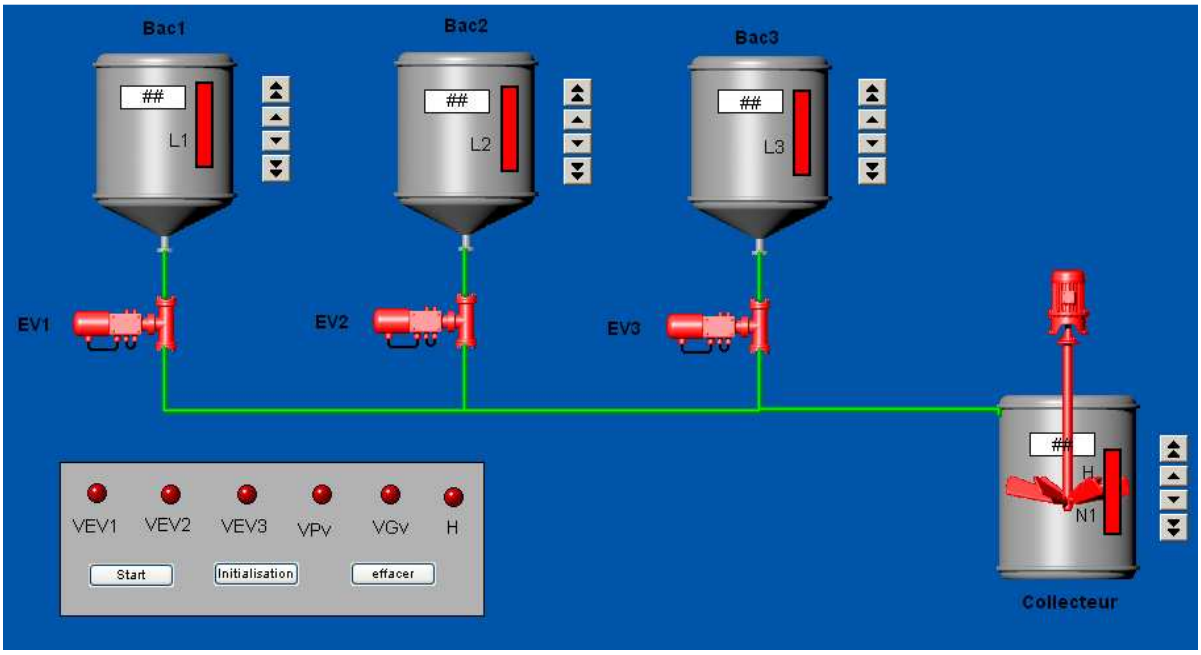


Figure III.36 : Vue du processus sous Vijeo Citect

▪l'étape importante pour la configuration après la communication, c'est de faire correspondre les variables crées en Unity Pro et celle de Vijeo Citect et ça en leurs affectant les mêmes adresses comme le montre les deux figures suivantes :

Niveau4	INT	%MW4
Niveau3	INT	%MW3
Niveau2	INT	%MW2
Niveau1	INT	%MW1
Dec	EBOOL	%M20
H	EBOOL	%M19
VGv	EBOOL	%M18
Gv	EBOOL	%M17
VPv	EBOOL	%M16
Pv	EBOOL	%M15
VEV3	EBOOL	%M14
EV3	EBOOL	%M13
VEV2	EBOOL	%M12
EV2	EBOOL	%M11
VEV1	EBOOL	%M10
EV1	EBOOL	%M9
N3	EBOOL	%M8
N2	EBOOL	%M7
N1	EBOOL	%M6
L3	EBOOL	%M5
L2	EBOOL	%M4
L1	EBOOL	%M3
Ini	EBOOL	%M2
Start	EBOOL	%M1

Figure III.37 : Déclaration des variables sous Unity Pro

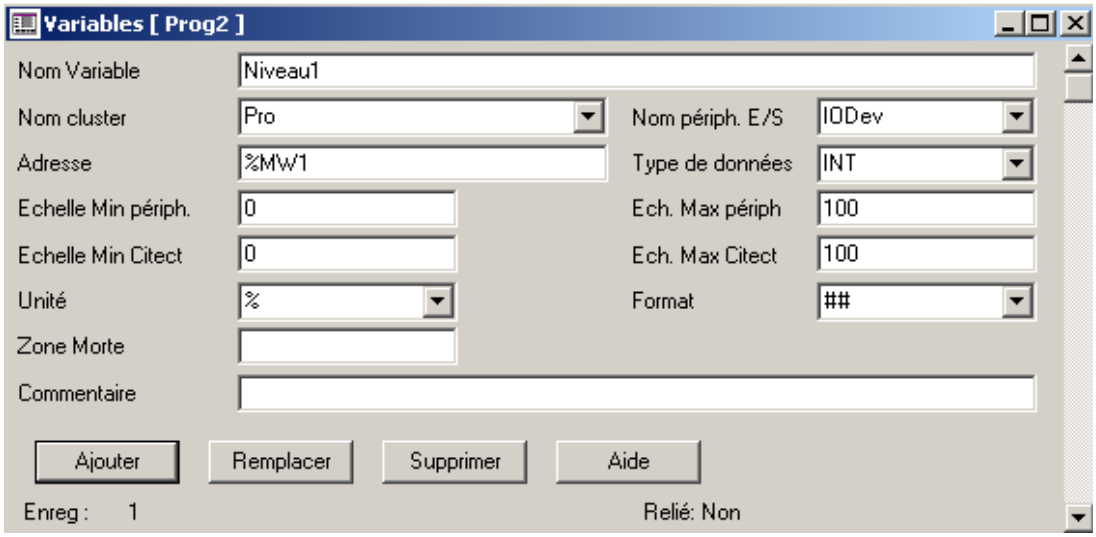


Figure III.38 : exemple de déclaration des variables sous Vijeo Citect

▪Pour la simulation, on lance d'abord le simulateur d'Unity Pro ensuite on execute le projet sous Vijeo Citect, on aura la figure suivante :

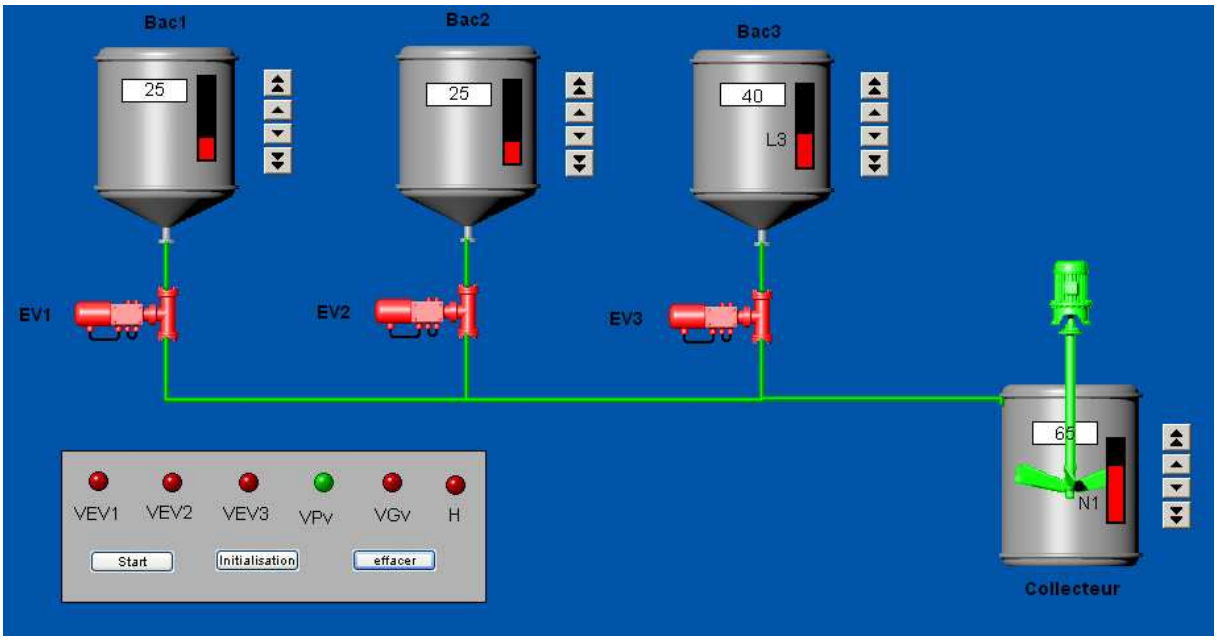


Figure III.39 : Vue de la troisième étape de l'exécution

III.5. Conclusion

Schneider Electric propose un grand choix en terme de pupitre de supervision et de contrôle. La description précédente du logiciel Vijeo Citect été de base, beaucoup d'autres options sont aussi offertes par ce dernier.

Chapitre IV

Transport des hydrocarbures

Chapitre IV : Transport des hydrocarbures

IV.1. Généralités sur le transport des hydrocarbures [9]

IV.1.1 : mode de transport

Les moyens de transport des hydrocarbures existants actuellement sont :

- Transport maritime : par navires pétroliers ou par caboteurs. Pour le gaz liquéfié, on utilise des méthaniers ;
- Transport ferroviaire : en utilisant des wagons citernes ;
- Transport routier : par camions citernes ;
- Transport par canalisation : pipelines.

▪Les moyens de transport utilisés en Algérie sont, dans l'ordre d'importance : Pipelines ; Wagons-citernes ; Camions-citernes.

▪On essayera dans ce qui suit de décrire le transport par canalisation et plus précisément, les oléoducs.

IV.1.2. Caractéristiques des pipelines

Un pipeline est caractérisé par :

- Une grande longueur ;
- Une pression de service élevée (jusqu'à 100 bars) ;
- Un gros diamètre ;
- Des épaisseurs variant de 6.35 mm à 30 mm et même plus.

IV.1.3. Exploitation des pipelines

Les taches principales d'un service d'exploitation d'un pipeline consistent à :

- Organiser le trafic d'un ou plusieurs produits ;
- Déterminer les régimes économiques du pompage ou de compression ;
- Maintenir le régime de fonctionnement des installations ;
- Surveillance, entretien et réparation de la ligne, et des équipements des stations et de protection contre la corrosion ;
- Un centre de dispatching dirige les manœuvres journalières ;
- On doit prévoir le renforcement des consignes de sécurité au passage des zones à forte densité de population aussi bien lors de la construction de la conduite (renforcement de l'épaisseur de la conduite, lestage, gaine,…)

IV.1.4. Utilisation des pipelines

En plus du pétrole et du gaz, on utilise les conduites pour transporter les produits suivant :

- Plusieurs produits par une même conduite, appelé batching ou pompage successif ;

- Produits visqueux et paraffine ayant la température de congélation au dessous de zéro. Pour cela, on utilise souvent le réchauffage pour vaincre les résistances hydrauliques à cause de la viscosité importante ;
- Du gaz liquéfié, avec une température de transport jusqu'à moins 160°C (courtes distances) ;
- De l'eau ;
- Des minéraux mélangés avec l'eau ;
- Eventuellement du matériel contenu dans des containers poussés par l'air sous pression fourni par des compresseurs.

IV.2. Oléoducs

Les oléoducs sont les moyens de transport du pétrole (liquide) et les gazoducs pour le gaz, les oléoducs sont constitués de :

- Les stations de pompage de départ et intermédiaire ;
- La ligne (pipeline) ;
- Les terminaux de départ et d'arrivée (T.D et T.A) ;
- Des points de livraison et de réception du produit ;
- Des postes de sectionnement et des postes de coupures ;
- Des installations de postes cathodiques ;
- Des installations de télécommande et de télécommunication.

IV.2.1. Exploitation des Oléoducs

IV.2.1.1. Méthodes d'augmentation du débit

Le débit de la conduite peut être augmenté par les méthodes suivantes :

- Pose de looping (élargissement) ;
- Doublement des S.P ;
- Augmentation du nombre de pompes ;
- Variation de la vitesse du moteur ;
- Méthode combinée (doublement des S.P, pose de looping).

IV.2.1.2. Oléoducs avec livraison et réception

Les livraisons et réceptions du brut sur le parcours de la conduite peuvent être discontinues ou continues suivant l'importance du client ou des clients pour la livraison et suivant le gisement ou les gisements pour la réception. Le calcul technologique des conduites avec livraison ou réception peut se faire par partie (par tronçon) limité par des points de réception ou de livraison. Si les livraisons (réceptions) ne sont pas importantes, on ne tient pas compte dans les calculs. Cependant dans le cas ou celles-ci se feraient périodiquement il faut en tenir compte pour la régulation des régimes de fonctionnement des machine (S.P).

IV.2.2. Description de l'OZ1

L'oléoduc OZ1existant reliant Haoud El Hamra (HEH) à Arzew d'une longueur totale de 801 km, a pour vocation l'évacuation du pétrole brut des champs de production situés en amont du terminal départ HEH.

IV.2.2.1. Injections intermédiaires

Suite a la découverte de nouveau gisements sur l'itinéraire de la conduite, celle-ci (OZ1) assure l'évacuation de la production de ces gisements. Il y a donc quatre points d'injection de même capacité.

Ces injections sont raccordées également à l'oléoduc OZ2, mais seulement en cas d'arrêt, de ligne OZ1.

Sur chaque ligne installées les nouveaux systèmes suivants :

▪Filtre en ligne avec by-pass et instrumentation locale, avec transmission via SCADA, de la pression différentielle.

▪Système de comptage du débit avec transmission, via SCADA, pour le contrôle de la portée dans les stations de pompages.

IV.2.2.2. Caractéristiques de la ligne OZ1

Les caractéristiques de la ligne OZ1 sont :

- Diamètre extérieur : 28'' (711.2 mm) ;
- Longueur : 801 km environ ;
- Acier : API 5Lx 52 ;
- Epaisseur télescopique : 6.35 à 11.13 mm.

▪Le profil de la ligne est caractérisé par un point haut (appelé point de contrôle Nador) au delà duquel les écoulements sont gravitaires.

▪Il faut noter que les deux oléoducs (OZ1 et OZ2) fonctionnent simultanément, mais on peut considérer que l'OZ2 est le principale et l'OZ1 est le secondaire, utilisé dans certain tronçon en cas d'urgence, comme le montre le tableau suivant :

Station	PK	Location	Fonctionne-Ment	Débit max (m^3/h)	Débit normal (m^3/h)
SP1	000.000	Hassi-Messaoud	OZ1+OZ2	5257(OZ2) 2783(OZ1)	4329(OZ2) 821(OZ1)
SP2	125.777	Ouargla	OZ2normal OZ1urgence	5257(OZ2) 2783(OZ1)	4329(OZ2) 0000(OZ1)
SP3	219.716	El Atteuf	OZ1+OZ2	5257(OZ2) 2783(OZ1)	4329(OZ2) 1581(OZ1)
SP4	306.732	Hassi R'mel	OZ2normal OZ1urgence	5257(OZ2) 2783(OZ1)	4329(OZ2) 0000(OZ1)

SP5	418.528	Tadjmout	OZ1+OZ2	5257(OZ2) 2783(OZ1)	4329(OZ2) 1701(OZ1)
SP6	538.176	Faidja	OZ2normal OZ1urgence	5257(OZ2) 2783(OZ1)	4329(OZ2) 0000(OZ1)
TA	821. 333	Arzew	OZ1+OZ2		

Tableau IV.1 : Fonctionnement générale de l'OZ1 et OZ2

IV.2.2.3. Les différentes stations de pompages

IV.2.2.3.1. Station SP1

Situé à HEH, au sein des parcs de stockage d'hydrocarbures existants, elle est composée des collecteurs, de la station basse pression (booster), de la station principale SP1 de Hassi Messaoud dans la wilaya de Ouargla, aux quels elle relié avec les quatre collecteurs.

La station booster est relié à la station principale par deux collecteurs de 1km environ.

La station SP1 est équipée de :

- Groupe de pompage permettant de faire fonctionner simultanément OZ1 et OZ2 ;
- Du séparateur API ;
- Du réseau de purge et détente ;
- Du réseau des égouttures ;
- Des shelters, des routes.

IV.2.2.3.2. Station SP2

Situé au PK 125.777 commune de Ouargla, elle sera équipé en phase deux de 4 turbopompes permettent de pomper, sur OZ2, un débit de 5257m^3/h (34 MTA).

IV.2.2.3.3. Station SP3

Situé au PK 219.716, commune El Atteuf, wilaya de Ghardaia, elle est conçue pour fonctionner sur l'OZ1 et OZ2 simultanément, et équipée de six turbopompes permettant de porter la capacité de transport à 11 MTA sur l'OZ1 (2 groupes) et 34 MTA sur l'OZ2 (4 groupes).

IV.2.2.3.4. Station SP4

Situé au PK 306.732, commune de Hassi R'mel, wilaya de Laghouat au sein du groupe de station SP4 existantes, elle est équipée de quatre groupes de pompages fonctionnant normalement sur l'OZ2 et permettant de porter le tonnage transporté à 34 MTA (5257 m^3/h).

IV.2.2.3.5. Station SP5

Conçue pour fonctionner simultanement sur l'OZ1 et l'OZ2, elle est équipée de six turbopompes permettant des tonnages identiques à SP3.

IV.2.2.3.6. Station SP6

Situé au PK 238.176, commune de Faidja, wilaya de Tiaret, elle est équipée de quatre turbopompes, qui, en fonctionnement normal, permettant de transporter 34 MTA sur l'OZ2.

IV.2.2.4. Description des terminaux

On a deux terminaux :

- Le terminal de départ (TD), situé à HEH ;
- Le terminal d'arrivé (TA), situé à Arzew.

IV.2.2.4.1. Terminal HEH

Ce terminal est compartimenté par zone en fonction des produits stockés, l'ensemble du parc de stockage est d'une capacité totale de 978400 m^3.

En plus des réservoirs, chaque zone possède son propre manifold lui permettant d'alimenter directement les pompes boosters au travers d'un collecteur qui lui est affecté.

Dans ce terminal, on trouve le parc de stockage OZ1, ce dernier possède huit réservoirs (4 de 35000 et 4 de 50000 m^3). Chaque réservoirs est connecté sur manifold existant et peut alimenter le collecteur 40''OZ1 par l'intermédiaire des vannes motorisées de 24''. Il faut noter aussi que toute les vannes du manifold OZ1 sont commandées localement ou à distances depuis la salle de contrôle SP1.

IV.2.2.4.2. Terminal d'Arzew

D'une manière générale, on distingue deux types d'installation sur le terminal TA, les installations existantes ou rénovées et les installations d'équipements neufs.

Le TA comprend :

- Une zone de stockage ou sont installés les réservoirs ;
- Une section faisant la continuité de la ligne principale.

A partir de la section ligne, le TA est raccordé au parc de stockage par un nouveau collecteur de 34'', placé en parallèle avec le collecteur existant de 28''. De l'amont vers l'aval ce collecteur comprend :

▪Une batterie de soupape haute pression ;

▪Une batterie de soupape basse pression ;

▪Un échantillonneur automatique ;

▪Un système complet de comptage à ultrason.

IV.2.2.5. Processus d'exploitation de l'OZ1

Le processus d'exploitation comporte trois phases :

- Phase 01 : Réception du produit ;
- Phase 02 : Stockage ;
- Phase 03 : Expédition.

IV.2.2.5.1. Phase 01(réception de produit)

Le pétrole brut provenant des champs du sud Algérien, transite par le centre de dispatching (CDHL) qui comptabilise et canalise le produit selon les besoins des terminaux, les arrivées sont :

- Le 30'' Ohanet ;
- Le 20'' Hassi-Messaoud ;
- Le 16'' El-Gassi ;
- Le 14'' R/El Baghel ;
- Le 26'' Mesbah.

IV.2.2.5.2. Phase 02(stockage)

La phase de stockage consiste en :

▪Préparation d'un bac de réception ;

▪Remplissage du bac à la hauteur voulue, le produit arrivé de CDHL, est canalisé vers un autre bac ;

▪Dès la fin du remplissage, le produit passe à la phase de décantation d'une durée de six heurs ;

▪Des échantillons sont prélevés dès la fin de la réception et avant expédition d'un bac pour déterminer, après analyse au niveau du laboratoire, les paramètres :

- Viscosité du produit ;
- Tension de vapeur ;
- Salinité,…

IV.2.2.5.3. Phase 03(expédition)

Le produit stocké est expédié six heures après la réception vers le terminal arrivé d'Arzew, en passant par un processus bien défini. Le produit arrive par gravité dans un collecteur principal d'aspiration des pompes boosters, ce derniers est aspiré puis refoulé entre 4 et 6 bars, afin d'éviter la cavitation par basse pression à 3 bars(seuil minimale admissible), et de ce fait, assurer une pression et un débit suffisant aux pompes principales, en tenant compte des contraintes et paramètres d'exploitation. Le produit expédié est comprimé par les stations de lignes jusqu'à son arrivée au terminal d'Arzew.

Phases	Capacité (MTA)	Nombre de station
I	10	3
II	18.7	6
III	21.6	6

Tableau IV.2 : Différente phases d'exploitation d'OZ1

▪Au cours des premières années d'exploitation, des gisements ont été découverts sur l'itinéraire, pour cela, la configuration de la ligne a été modifiée par la réalisation de quatre points d'injections répartis comme suit :

Injection	PK	Altitude(m)	Débit moyen (m^3/h)
Haoud Berkaoui	72.5	210	230
Guellala	75.1	205	280
Oued Noumer	205	440	250
Hassi R'mel	295.5	460	120

Tableau IV.3 : Différents points d'injection dans l'oléoduc OZ1

IV.3. Fonctionnement générale des stations de pompages

Dans cette partie, on va essayer de décrire le fonctionnement général des stations de pompages, on peut diviser cette dernière en plusieurs sections à savoir :

- Entrée station ;
- Filtration ;
- Stockage ;
- Pomperie boosting ;
- Pomperie principale ;
- Expédition ;
- Réseau de drain ;
- Réseau de purge.

▪La figure suivante montre ces différentes sections et les interactions entre elles :

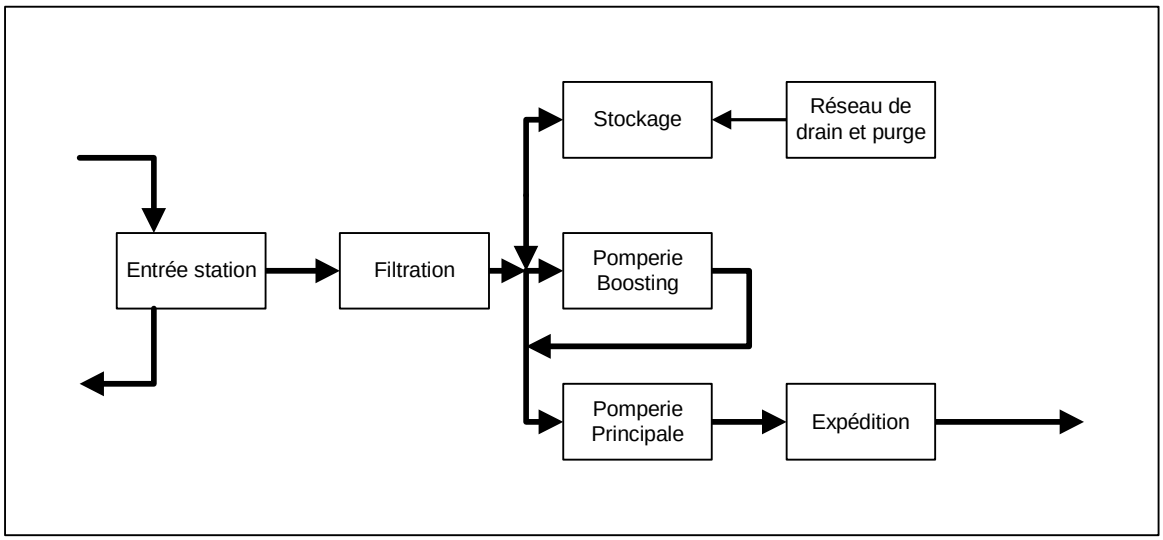


Figure IV.1 : Différentes sections

▪De plus, la station est protégé par un réseau anti incendie.

IV.3.1. Entrée station

Dans cette section, on trouve essentiellement deux vannes : une vanne régulatrice qui assure un débit constant pour la station et une vanne motorisé qui est soit ouverte ou fermée.

▪ La vanne motorisé est en position ouverte dans le cas ou l'alimentation de la pomperie principale se fait directement par le pipe de l'entrée ainsi dans le cas ou effectue le remplissage des bacs de stockage.

▪La vanne motorisé est position fermée dans le cas d'arrêt totale de la station ainsi que dans le cas ou l'alimentation de la pomperie principale se fait par les bacs de stockage via la pompeire boosting.

IV.3.2. Filtration

Cette section est installée à l'entrée de la station pour l'élimination des particules solides ayant une taille supérieure à 200µ. Elle est composée généralement de deux unités de filtration, une en marche et l'autre en réserve d'une capacité de traitement de 1400 m^3/h chacune.

▪En plus du filtre, chaque unité de filtration est composé de :

- Une vanne motorisée d'entrée ;
- Un transmetteur de pression différentielle ;
- Une vanne motorisée de sortie.

▪Cette section opère en deux modes : un mode normal et un mode secours, le passage du mode normal au mode secours fait passer les quatre vannes des unités de filtration en mode

automatique et le passage du mode secours au mode normal passe les quatre vannes des unités de filtration en mode manuel.

▪En mode secours, le basculement du filtre en service sur le filtre en réserve se fait sur la détection de l'alarme haute pression différentielle. Dans ce cas, les vannes du filtre en réserve s'ouvrent simultanément et dès confirmation de la position ouverte de ces deux vannes motorisée, les vannes du filtre en service se ferment.

IV.3.3. Stockage

La section de stockage se compose au minimum de deux bacs de stockage de brut, mais on peut avoir plus, d'une capacité de 12000 m^3 chacun. Ces bacs de stockage servent de tampons de réserve à la station de pompage.

▪Chaque bac de stockage se compose de :

- Une vanne motorisée d'entrée sur le réseau de drain et purge ;
- Une vanne motorisée d'entrée/sortie sur le réseau d'arrivée du brut ;
- Un transmetteur de niveau ;
- Un switch de détection de niveau haut ;
- Un switch de détection de niveau bas.

▪Le remplissage de ces bacs de stockage est réalisé soit depuis le réseau d'arrivée du brut (réseau principal) soit depuis le réseau de drain et purge (réseau secondaire) et ces bacs ne peuvent être vidés que par le réseau principal.

IV.3.4. Pomperie boosting

La pomperie boosting se compose de trois pompes de gavage. Ces pompes sont utilisées dans le cas où l'alimentation de la station depuis le pipe à l'entrée est rendue indisponible ou insuffisante et que la pomperie principale est alimentée par les bacs de stockage.

▪Le fonctionnement des pompes de gavage est prévu en mode 2 sur 3, ce qui signifie deux pompes en service et une pompe en réserve.

▪Chaque ligne de gavage se compose de :

- Une vanne motorisée d'aspiration ;
- Un switch de débit aspiration très bas ;
- Une pompe de gavage ;
- Un switch de pression haute refoulement ;
- Une mesure de débit de refoulement ;
- Une ligne de recirculation ;
- Une vanne motorisée de refoulement.

▪Les paramètres de sortie des pompes de gavage sont indiqués dans le tableau suivant :

Nombre de pompe en service	Débit	Pression
1	290 m3/h	13 bar
2	580 m3/h	13 bar

Tableau IV.4 : Paramètres de sortie des pompes boosting

IV.3.5. Pomperie principale

La pomperie principale se compose de quatre pompes d'expédition alimentées soit par la déviation du pipe d'entrée, soit par la pomperie boosting.

▪Le fonctionnement des pompes d'expédition est prévu en mode 2 sur 4 ou en mode 3 sur 4. Cela signifie que deux pompes sont en service avec deux pompes en réserve pour le mode 2 sur 4 et que trois pompes sont en service avec une pompe en réserve pour le mode 3 sur 4, et ça en fonction du débit.

▪Chaque ligne d'expédition se compose de :

- Une vanne motorisée d'aspiration ;
- Un switch de débit aspiration très bas ;
- Un switch de pression aspiration basse ;
- Une pompe d'expédition avec variateur de vitesse ;
- Une mesure de débit de refoulement ;
- Une ligne de recirculation ;
- Une vanne motorisée de refoulement ;
- Un switch de pression très haute refoulement.

▪Les paramètres de sortie des pompes principales sont :

Nombre de pompe en service	Régime de fonctionnement (Débit global)	Débit nominal (pour une pompe)	Pression de refoulement	Vitesse de rotation (pour une pompe)
1	217 m3/h	217 m3/h	59 bar	2865 tr/mn
2	430 m3/h	215 m3/h	59 bar	2698 tr/mn
3	580 m3/h	193 m3/h	63 bar	2762 tr/mn

Tableau IV.5 : Paramètres de sortie des pompes principales

▪Comme le montre le tableau, on a trois régimes de fonctionnement.

IV.3.6. Expédition

Dans cette section, on trouve deux fonctions :

- Le banc de comptage ;
- La gare racleur : utilisée pour le nettoyage du pipe.

▪Pour la gare racleur, elle est composé de :

- Une vanne motorisée entrée gare racleur ;
- Une vanne motorisée sortie gare racleur ;
- Une vanne motorisée by pass gare racleur ;

- Un contact de présence racleur dans la gare ;
- Un contact de détection de passage racleur.

IV.3.7. Réseau de drain

Le réseau de drainage permet de collecter les drains du site de part un réseau secondaire. Ce pétrole est stocké dans une cuve contenant :

- Un transmetteur de niveau ;
- Une pompe verticale.

IV.3.8. Réseau de purge

Le réseau de purge permet de collecter les purges du site de part un réseau secondaire. Ce pétrole est stocké dans deux cuves contenant chacune un transmetteur de niveau.

IV.4. Notre objectif

Notre objectif est de réaliser les programmes de gestion de certains équipements présents au sein de la station, créer des objets représentants ces équipements en utilisant Vijeo Citect et faire la simulation. Pour cela, on a créé des programmes sous Unity Pro en utilisant les DFB, Pour la gestion de certaines sections décrites précédemment.

Chapitre V

Application

V. Application

V.1. DFB vanne motorisé

V.1.1. Cahier de charge

La vanne motorisé reçoit les signaux d'ouverture et de fermeture et peut envoyer les deux signaux indiquant son état (complètement ouverte ou complètement fermée). Le DFB reçoit 8 signaux d'entrés et envoi 7 signaux de sorties.

▪Les 8 entrées :

- Ouvrir : Commande ouvrir la vanne (signal envoyé par l'utilisateur ou par un autre programme) ;
- Fermer : Commande fermer la vanne (signal envoyé par l'utilisateur ou par un autre programme) ;
- FCO : Fin de course ouverture de la vanne (signal générer par la vanne) ;
- FCF : Fin de course fermeture de la vanne (signal générer par la vanne) ;
- Déf : Défaut au niveau du moteur de la vanne ;
- Acquit : Acquittement d'un défaut ;
- Auto : Mode automatique de la vanne (vanne peut être piloté à partir de la salle de commande) ;
- Local : Mode local de la vanne (vanne piloté sur site).

▪Les 7 sorties :

- CMD_O : Commande d'ouverture de la vanne (signal envoyé vers la vanne) ;
- CMD_F : Commande de fermeture de la vanne (signal envoyé vers la vanne) ;
- Disc : Défaut de discordance (ouverture ou fermeture);
- Inco : Défaut d'incohérence ;
- Etat_O : signal d'état d'ouverture de la vanne ;
- Etat_F : Signal d'état de fermeture de la vanne ;
- Aver : Signal d'avertissement (signal clignotant en cas de défaut de discordance ou d'incohérence).

▪Le DFB Vanne doit assurer :

- La génération des signaux d'ouverture et de fermeture de la vanne en cas d'absence de défaut ;
- La génération du signal de discordance, soit en ouverture ou en fermeture :
  -En ouverture : la discordance est générée lorsque on envoie la commande ouvrir la vanne et qu'on ne reçoit pas le signal « FCO » indiquant que la vanne est complètement ouverte et cela après un certain temps (la vanne prend un temps pour s'ouvrir due à son inertie). Pour des raisons de sécurité ce temps serai une variable locale et non pas d'entrée pour le bloc.
  -De même pour la discordance en fermeture.
- La génération du signal incohérence lorsque on reçoit les deux signaux FCF et FCO en même temps pendant une duré réglée par le programmeur (variable locale) et cela de peur que la présence de ces deux signaux n'est qu'un parasite.

- L'acquittement des défauts : discordance et incohérence, l'acquittement de ces défauts n'est appliqué qu'à l'absence du défaut et après un appuie sur le bouton Acquit.

V.1.2. Programme du DFB gestion de la vanne

Vanne		<DFB>		
<entrées>				
Ouvrir	1	EBOOL		Commande d'ouverture de la vanne
Fermer	2	EBOOL		Commande de fermeure de la vanne
FCO	3	EBOOL		Fin de course ouverture de la vanne
FCF	4	EBOOL		Fin de course fermeture de la vanne
Def	5	EBOOL		Defaut
Acquit	6	EBOOL		Acquitement de defaut
Auto	7	EBOOL		Mode automatique de la vanne
Local	8	EBOOL		Mode local de la vanne
<sorties>				
CMD...	1	EBOOL		Signal de commande d'ouverture
CMD...	2	EBOOL		Signal de commande de fermeture
Disc	3	EBOOL		Defaut discordance
Inco	4	EBOOL		Defaut incoherance
Etat_O	5	EBOOL		Etat ouverte de la vanne
Etat_F	6	EBOOL		Etat fermer de la vanne
Avert	7	EBOOL		Avertissement de defaut
<entrées...				
<public>				
Temp1		TIME	20S	
Temp2		TIME	20S	

Figure V.1: Les E/S du DFB Vanne

▪Le programme du DFB Vanne est en [**Annexe : D.3.1**].

V.2. DFB défaut groupe électropompe (GEP)

Pour des raisons de sécurité, le GEP ne doit démarrer que si tous les systèmes fonctionnent correctement, donc plusieurs facteurs de défaut doivent être vérifiés avant le démarrage.

V.2.1. Cahier de charge DFB défaut

Le but de ce DFB est la signalisation de 3 défauts principaux :

- Défaut de température des bobinages du moteur de la pompe, ce dernier est équipé de trois capteurs de température ;
- Défaut de température des paliers (deux capteurs de température) ;
- Défaut de vibration du GEP.

▪ Ce DFB présente 16 signaux en entrée et 4 en sorties.

▪Les entrées :

- TempB1, 2 et 3 : Températures des bobines de la pompe ;
- Tempe_B_Seu : Température seuil des bobines de la pompe ;
- TempP1 et 2 : Températures des paliers de la pompe ;
- Temp_P_seuil : Température seuil des paliers de la pompe ;

- Vibra : vibration mécanique de la pompe ;
- Vibra_Seu : vibration seuil de la pompe ;
- Def_Van : défaut de la vanne ;
- Disj_Van : disjoncteur vanne ;
- Disj_GEP : disjoncteur GEP ;
- Def_Iso : défaut isolement ;
- Def_Dem : défaut démarreur progressive ;
- GEP_Sou_Tension : GEP sous-tension (présence d'alimentation) ;
- Acquit : pour l'acquittement des défauts.

▪Les 4 sorties :

- Def_T_B : Défaut de température des bobinages ;
- Def_T_P : Défaut de température des paliers ;
- Def_V : Pour dire que la vibration a dépassé le seuil ;
- Def : Qui résume tout les défauts.

▪Le DFB Defaut doit envoyer 4 signaux signalant l'existence d'un défaut et le préciser.

V.2.2. Programme DFB Defaut

Defaut		<DFB>		
<entrées>				
Tem...	1	INT		Temperature de la bobine 1
Tem...	2	INT		Temperature de la bobine 2
Tem...	3	INT		Temperature de la bobine 3
Tem...	4	INT		Temperature du seuil des bobines
Tem...	5	INT		Temperature du palier 1
Tem...	6	INT		Temperature du palier 2
Tem...	7	INT		Temperature de seuil des paliers
Vibra	8	INT		Vibration mecanique de la pompe
Vibra...	9	INT		Seuil de vibration de la pompe
Def_...	10	EBOOL		Defaut vanne
Disj_...	11	EBOOL		Disjoncteur vanne
Disj_...	12	EBOOL		Disjoncteur GEP
Def_...	13	EBOOL		Defaut isolement
Def_...	14	EBOOL		Defaut demareur progrissive
GEP...	15	EBOOL		GEP sous tensio
Acquit	16	EBOOL		Acquitement de defauts
<sorties>				
Def_...	1	EBOOL		Defaut temperature bobinage
Def_...	2	EBOOL		Defaut temperature palier
Def_V	3	EBOOL		Defaut niveau de vibration
Def	4	EBOOL		Defaut genenrale

Figure V.2 : Les E/S du DFB Defaut

▪Le programme du DFB Défaut est en [**Annexe : D.3.2**].

V.3. DFB Pompe (groupe électropompe)

Le groupe électropompe est constitué de la pompe et son moteur, le GEP envoie deux signaux, un pour indiquer que son démarrage est terminé et un autre pour indiquer la fin de son arrêt. Le DFB GEP assurera un bon fonctionnement de la pompe (un démarrage et un arrêt sécurisé) et permet aussi de calculer le temps de marche du groupe électropompe.

V.3.1. Cahier de charge GEP

Le DFB Pompe est constitué de 13 entrées et de 13 sorties.

▪Les entrées :

- Permi : Autorisation de démarrage du GEP ;
- Marche : Commande marche du GEP (signal envoyé par l'utilisateur ou un autre programme) ;
- Arret : Commande Arrêt du GEP (signal envoyé par l'utilisateur ou un autre programme) ;
- Fin_Dim : Signal de fin de démarrage (envoyé par le GEP) ;
- Fin_Arr : Pour dire que le GEP est complètement arrêté (envoyé par le GEP) ;
- V_Amo : Etat de la vanne amont (ouverte ou fermée) ;
- V_Ava : Etat de la vanne aval (ouverte ou fermé) ;
- V_Reg : Etat de la vanne régulatrice (ouverte ou fermé) ;
- Man : Mode manuel de la pompe ;
- Auto : Mode automatique de la pompe ;
- Débit : Mesure du débit à la sortie de la pompe (au refoulement) ;
- Débit_N : Débit nominal du fonctionnement de la pompe.

▪Les sorties :

- CMD_M : Commande de démarrage du GEP (signal envoyé au moteur de la pompe) ;
- CMD_A : Commande d'arrêt du GEP (signal envoyé au moteur de la pompe) ;
- O_V_Av : Signal d'ouverture de la vanne aval ;
- O_V_Am : Signal d'ouverture de la vanne amont ;
- O_V_Re : Signal d'ouverture de la vanne régulatrice ;
- F_V_Av : Signal de fermeture de la vanne aval ;
- F_V_Am : Signal de fermeture de la vanne amont ;
- F_V_Re : Signal de fermeture de la vanne régulatrice ;
- E_M : Etat marche de la pompe ;
- E_A : Etat arrêt de la pompe ;
- Ph_D : Phase de démarrage de la pompe ;
- Ph_A : Phase arrêt de la pompe ;
- Def : Signl de défaut ;
- Avert : Signal d'avertissement ;
- T_Sec : Composante en seconde du temps de fonctionnement ;
- T_Min : Composante en minute du temps de fonctionnement ;
- T_Heur : Composante en heure du temps de fonctionnement.

V.3.2. Programme DFB Pompe

Pompe		<DFB>		
<entrées>				
Permi	1	EBOOL		Autorisation de demmarage
Marc...	2	EBOOL		Commande marche
Arret	3	EBOOL		Commmande arret
Fin_...	4	EBOOL		Fin de demmarage
Fin_Arr	5	EBOOL		Arret total
V_A...	6	EBOOL		Etat de la vanne amont
V_Ava	7	EBOOL		Etat de la vanne aval
V_Reg	8	EBOOL		Etat de la vanne regulatrice
S_def	9	EBOOL		Signal defaut
Man	10	EBOOL		Mode manuel
Auto	11	EBOOL		Mode automatique
Debit	12	INT		Debit a la sortie de la pompe
Debi...	13	INT		Debit nominal de fonctionnement de la pompe
<sorties>				
CMD...	1	EBOOL		Commande marche
CMD...	2	EBOOL		Commande arret
O_V...	3	EBOOL		Ouverture de la vanne aval
F_V_R	4	EBOOL		Fermeture de la vanne regulatrice
Def	5	EBOOL		Signal de defaut
E_M	6	EBOOL		La pompe est en etat de marche
E_A	7	EBOOL		La pompe est a larret

Figure V.3 : Les E/S du DFB Pompe

▪Le programme DFB GEP est en [**Annexe : D.3.3**].

V.4. DFB Régime

Le DFB Régime permet de sélectionner le régime de fonctionnement ainsi de préciser le débit nominal de fonctionnement des pompes, ce DFB possède une seul entrée et quatre sorties.

▪L'entrée :

- Debit : C'est le débit de fonctionnement de la station (débit voulu à la sortie de la station).

▪Les sorties :

- Regi1 : Sortie qui correspond au régime 1 ;
- Regi2 : Sortie qui correspond au régime 2 ;
- Regi3 : Sortie qui correspond au régime 3 ;
- Debit_N : Débit nominal de fonctionnement des pompes correspond à chaque régime.

V.4.1. Programme DFB Regime

Regime		<DFB>		
<entrées>				
Debit	1	INT		Debit de fonctionnement
<sorties>				
Regi1	1	EBOOL		Régime 1 de fonctionnement
Regi2	2	EBOOL		Régime 2 de fonctionnement
Regi3	3	EBOOL		Régime 3 de fonctionnement
Debi...	4	INT		Debit nominal de fonctionnement
<entrées...				
<public>				
<privé>				
<sections>				

Figure V.4 : Les E/S du DFB Regime

▪Le programme du DFB Regime est en [**Annexe : D.3.4**].
▪Le programme écrit précédemment été d'une manière générale, ce qui signifie qu'on n'a pas précisé les débits correspondant à chaque régime ainsi que les débits nominaux qui vont avec.

V.5. DFB Autorisation

C'est le DFB qui donne les autorisations de démarrage aux lignes d'expéditions, permettant ainsi de contrôler les pompes. Ce DFB possède 7 entrées et 6 sorties.
▪Les entrées :

- Regi1 : Indique que c'est le premier régime (signal qui vient du DFB Regime) ;
- Regi2 : Indique que c'est le deuxième régime (signal qui vient du DFB Regime) ;
- Regi3 : Indique que c'est le troisième régime (signal qui vient du DFB Regime) ;
- Activ1 : Activation de la première ligne (action effectuer par l'opérateur) ;
- Activ2 : Activation de la deuxième ligne (action effectuer par l'opérateur) ;
- Activ3 : Activation de la troisième ligne (action effectuer par l'opérateur) ;
- Activ4 : Activation de la quatrième ligne (action effectuer par l'opérateur) ;

▪Les sorties :

- Auto1 : Autorisation de démarrage de la première ligne (signal envoyé au DFB pompe) ;
- Auto2 : Autorisation de démarrage de la deuxième ligne (signal envoyé au DFB pompe) ;
- Auto3 : Autorisation de démarrage de la troisième ligne (signal envoyé au DFB pompe) ;
- Auto4 : Autorisation de démarrage de la quatrième ligne (signal envoyé au DFB pompe) ;
- Erreur : Signal d'erreur ;
- Averti1 : Signal d'avertissement.

▪Ce DFB gère d'une certaine manière la redondance entre les lignes d'expéditions, les deux signaux : Erreur et Averti1 sont générés si on active un nombre de ligne supérieur au nombre

tolérer par le régime de fonctionnement, par exemple, pour le premier régime, on a besoin d'une seul ligne alors si on active deux lignes, un message erreur va apparaitre sur l'écran de visualisation.

V.5.1. Programme DFB Autorisation

Autorisation		<DFB>		
<entrées>				
Regi1	1	EBOOL		Régime 1 de fonctionnement
Regi2	2	EBOOL		Régime 2 de fonctionnement
Regi3	3	EBOOL		Régime 3 de fonctionnement
Activ1	4	EBOOL		Activation de la premiere ligne
Activ2	5	EBOOL		Activation de laa deuxieme ligne
Activ3	6	EBOOL		Activation de la troisieme ligne
Activ4	7	EBOOL		Activation de la quatrieme ligne
<sorties>				
Auto1	1	EBOOL		Autorisation d'activation de la ligne 1
Auto2	2	EBOOL		Autorisation d'activation de la ligne 2
Auto3	3	EBOOL		Autorisation d'activation de la ligne 3
Auto4	4	EBOOL		Autorisation d'activation de la ligne 4
Erreur	5	EBOOL		Message d'erreur
Averti1	6	EBOOL		Avertissement
<entrées...				
<public>				
<privé>				
<sections>				

Figure V.5 : Les E/S du DFB Autorisation

▪Le programme du DFB Autorisation est en [**Annexe : D.3.5**].

V.6. DFB Validation

Ce DFB permet de faire le choix définitif des lignes de pompes qui vont être mis en service, donc l'autorisation finale de démarrage des pompes n'est acceptée que si on valide notre choix et ça sans avoir le signal d'erreur qui est généré par le DFB Autorisation.

▪Les entrées :

- Valider : Permet de valider le choix ;
- Permi1 : Permission pour la première ligne (signal envoyé par le DFB Autorisation) ;
- Permi2 : Permission pour la deuxième ligne (signal envoyé par le DFB Autorisation) ;
- Permi3 : Permission pour la troisième ligne (signal envoyé par le DFB Autorisation) ;
- Permi4 : Permission pour la quatrième ligne (signal envoyé par le DFB Autorisation) ;
- Erreur : Signal d'erreur envoyé par le DFB Autorisation.

▪Les sorties :

- Auto1 : Autorisation final de démarrage de la première ligne (signal envoyé au DFB pompe) ;
- Auto2 : Autorisation final de démarrage de la deuxième ligne (signal envoyé au DFB pompe) ;

- Auto3 : Autorisation final de démarrage de la troisième ligne (signal envoyé au DFB pompe) ;
- Auto4 : Autorisation final de démarrage de la quatrième ligne (signal envoyé au DFB pompe) ;

V.6.1. Programme DFB Validation

Validation		<DFB>		
<entrées>				
Valider	1	EBOOL		Valider le choix
Permi1	2	EBOOL		Permission pour la premiere ligne
Permi2	3	EBOOL		Permission pour la deuxieme ligne
Permi3	4	EBOOL		Permission pour la troisieme ligne
Permi4	5	EBOOL		Permission pour la quatrieme ligne
Erreur	6	EBOOL		Erreur de choix
<sorties>				
Auto1	1	EBOOL		Autorisation final pour la premiere ligne
Auto2	2	EBOOL		Autorisation finale pour la deuxieme ligne
Auto3	3	EBOOL		Autorisation finale pour la troisieme ligne
Auto4	4	EBOOL		Autorisation final pour la quatrieme ligne
<entrées...				
<public>				
<privé>				
<sections>				
S1		<ST>		

Figure V.6 : Les E/S du DFB Validation

▪Le programme du DFB Validation est en [**Annexe : D.3.6**].

V.7. Programmes principaux

V.7.1. Programme de simulation de la vanne

Après la déclaration des variables, on aura le programme suivant (on utilise le langage FBD) :

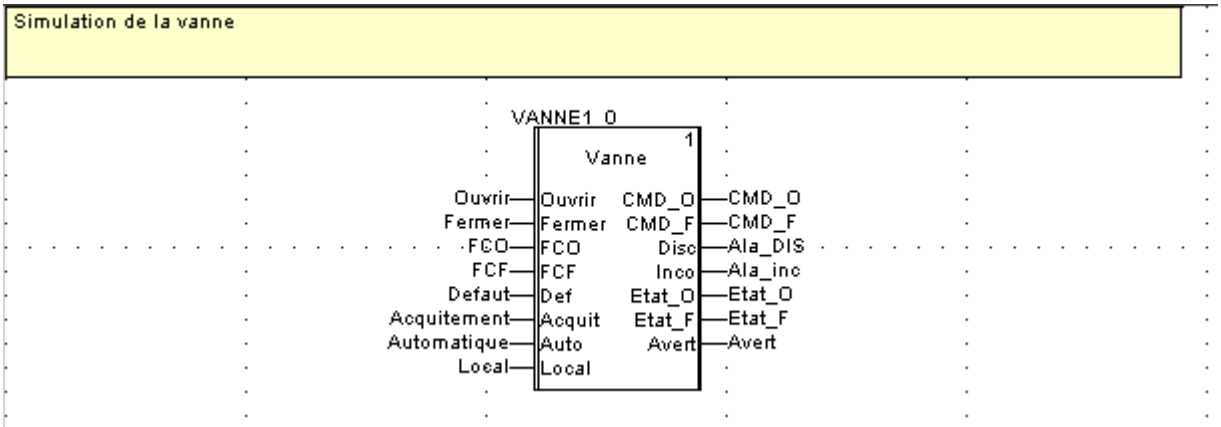
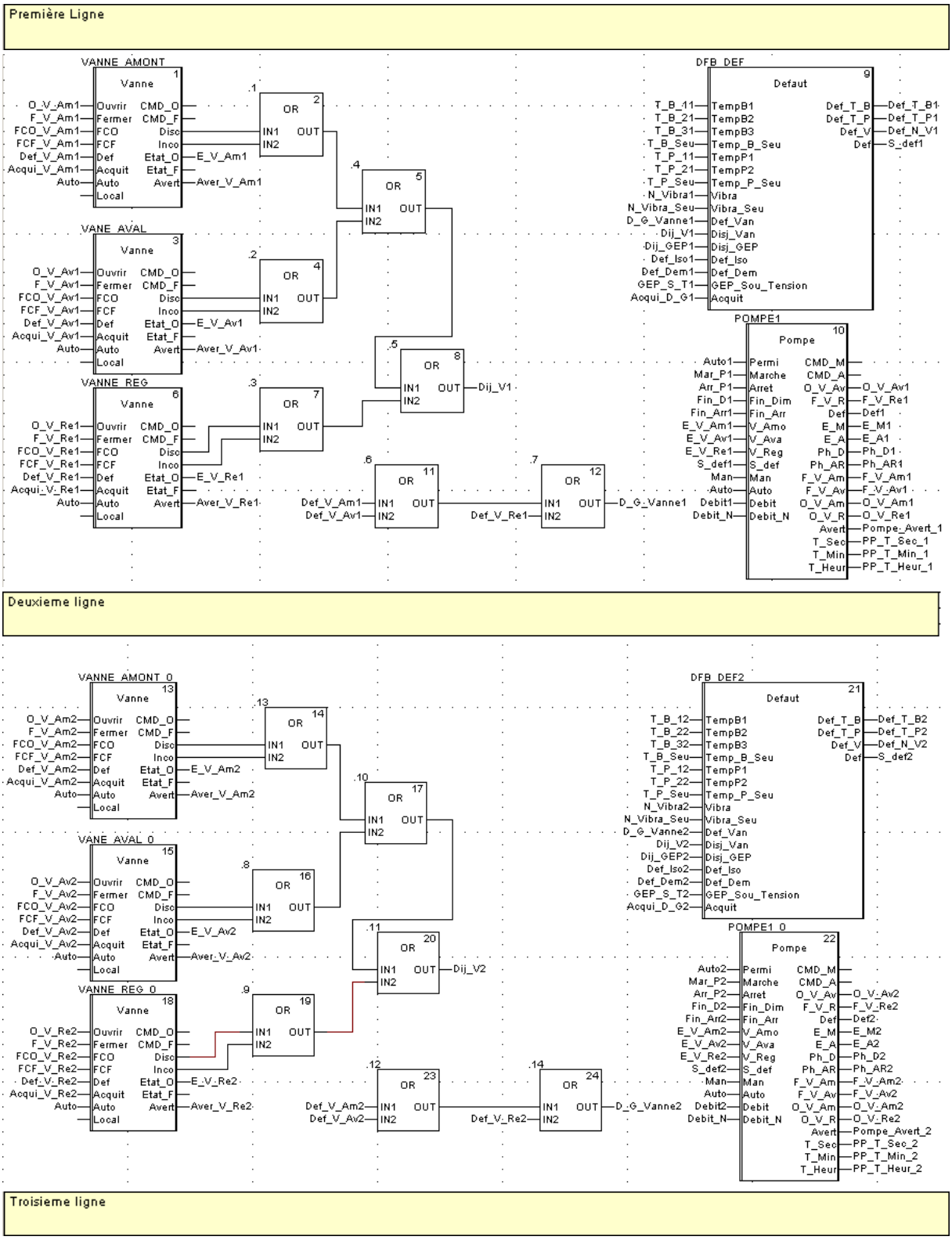
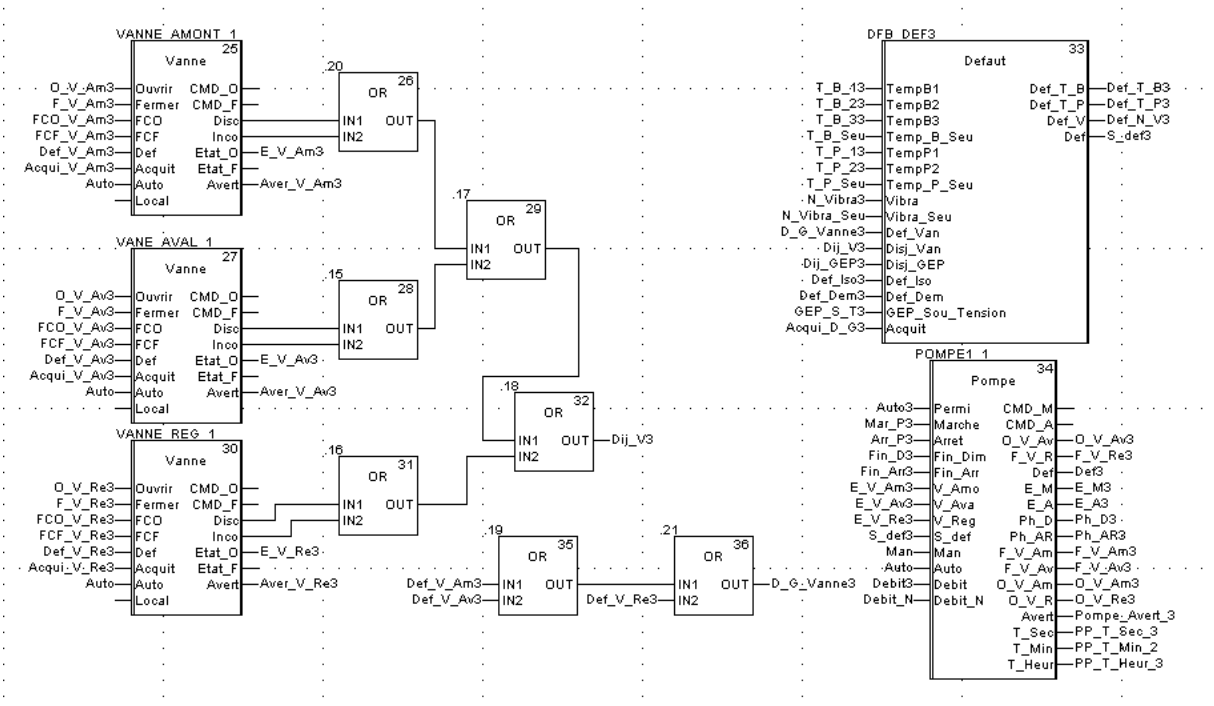


Figure V.7 : Programme simulation vanne

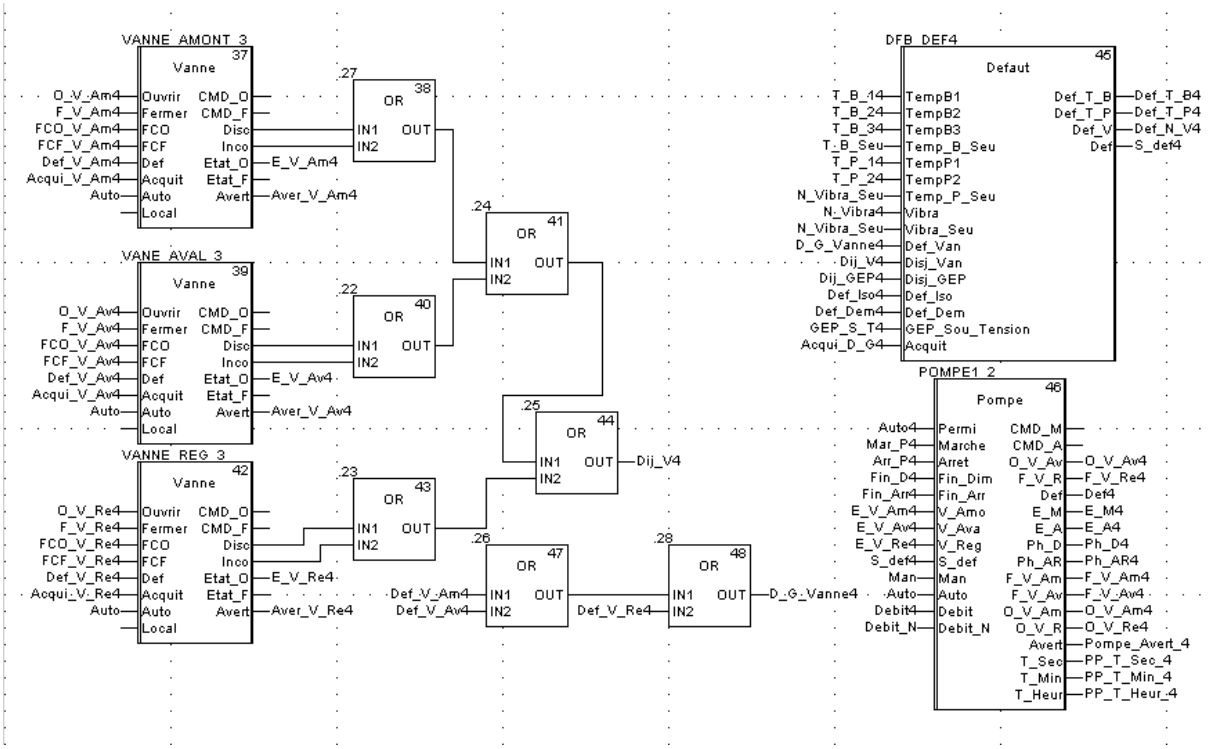
V.7.2. Programme de gestion des lignes d'expéditions

Après la déclaration de toutes les variables nécessaire, on aura le programme suivant (écrit en langage FBD) :





Quatrieme ligne



Distribution des autorisation de demmarage

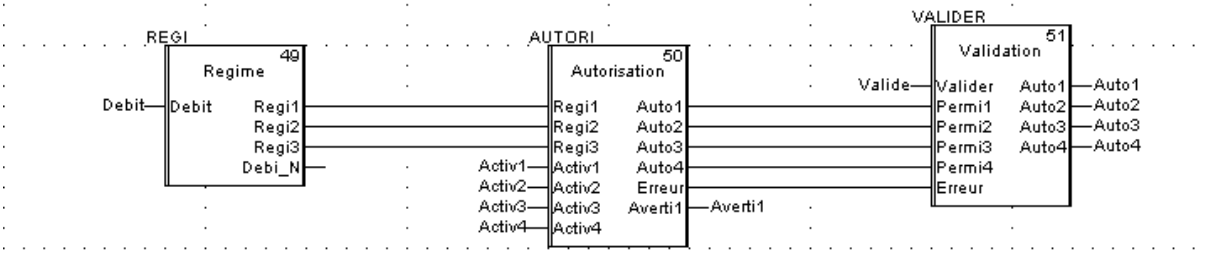
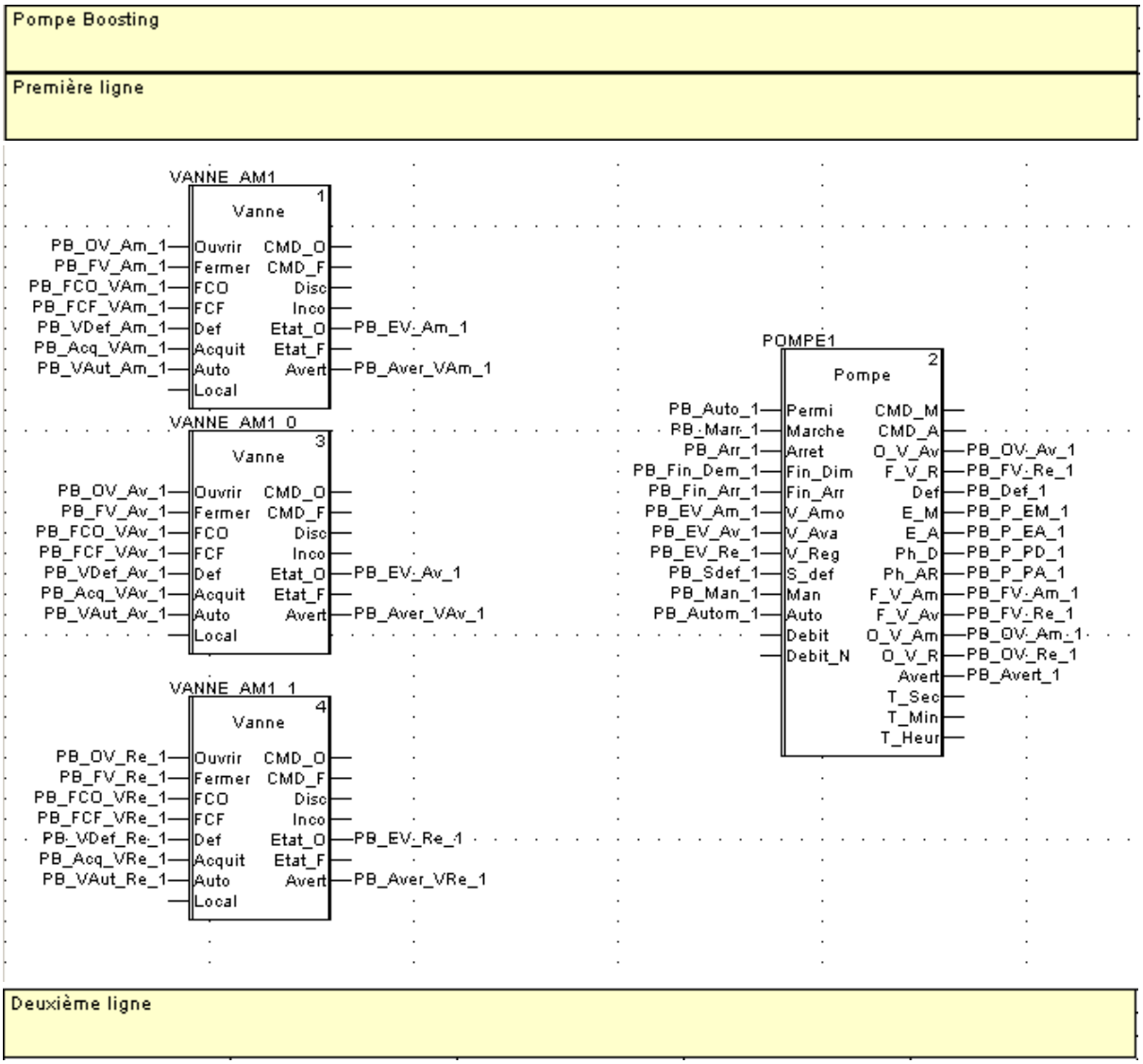
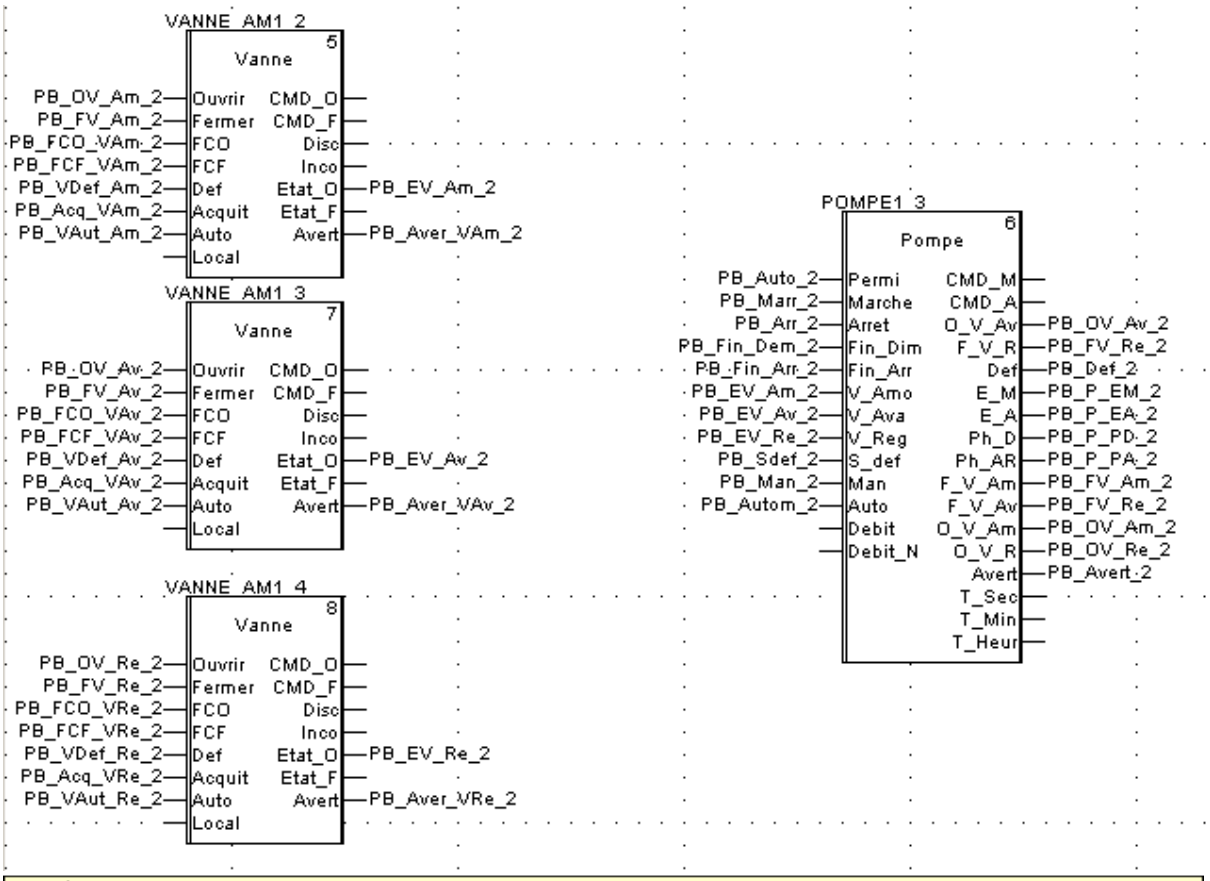


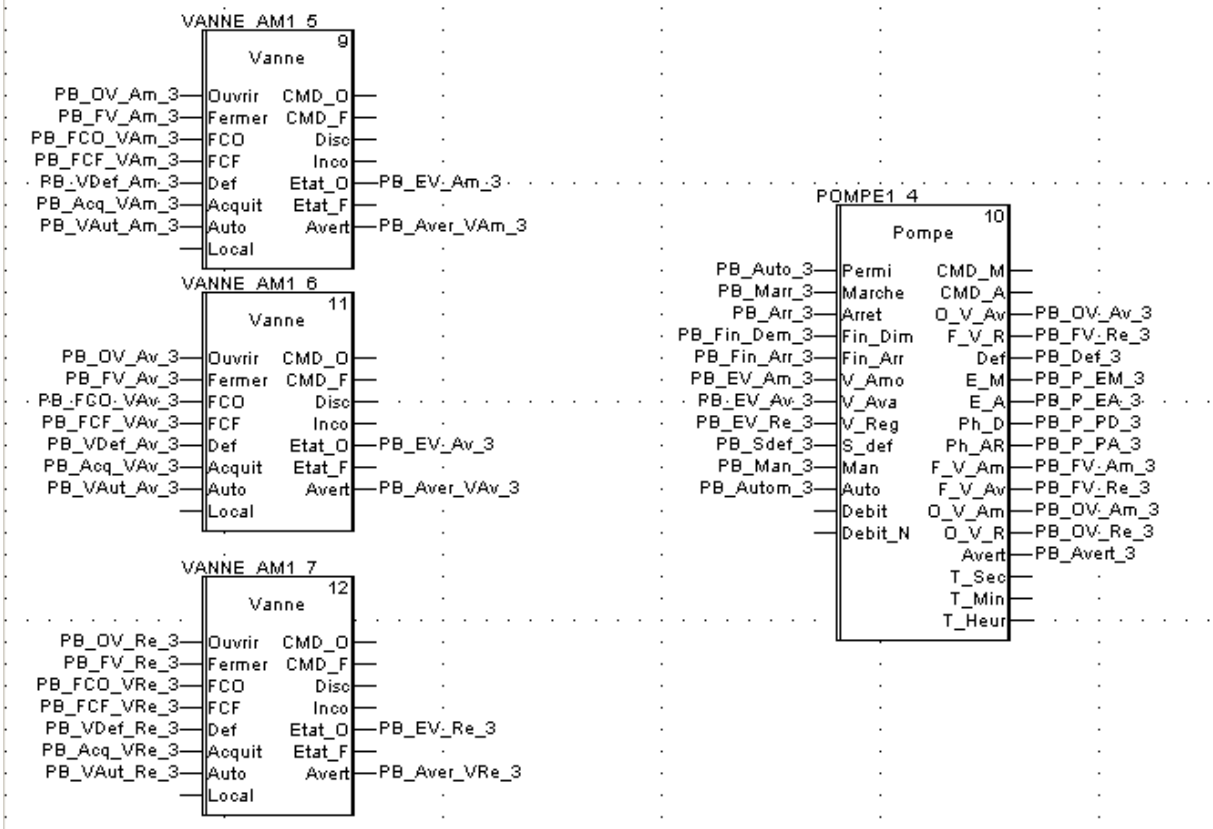
Figure V.8 : Programme lignes d'expédition

V.7.3. Programme de gestion des pompes boosting





Troisième ligne



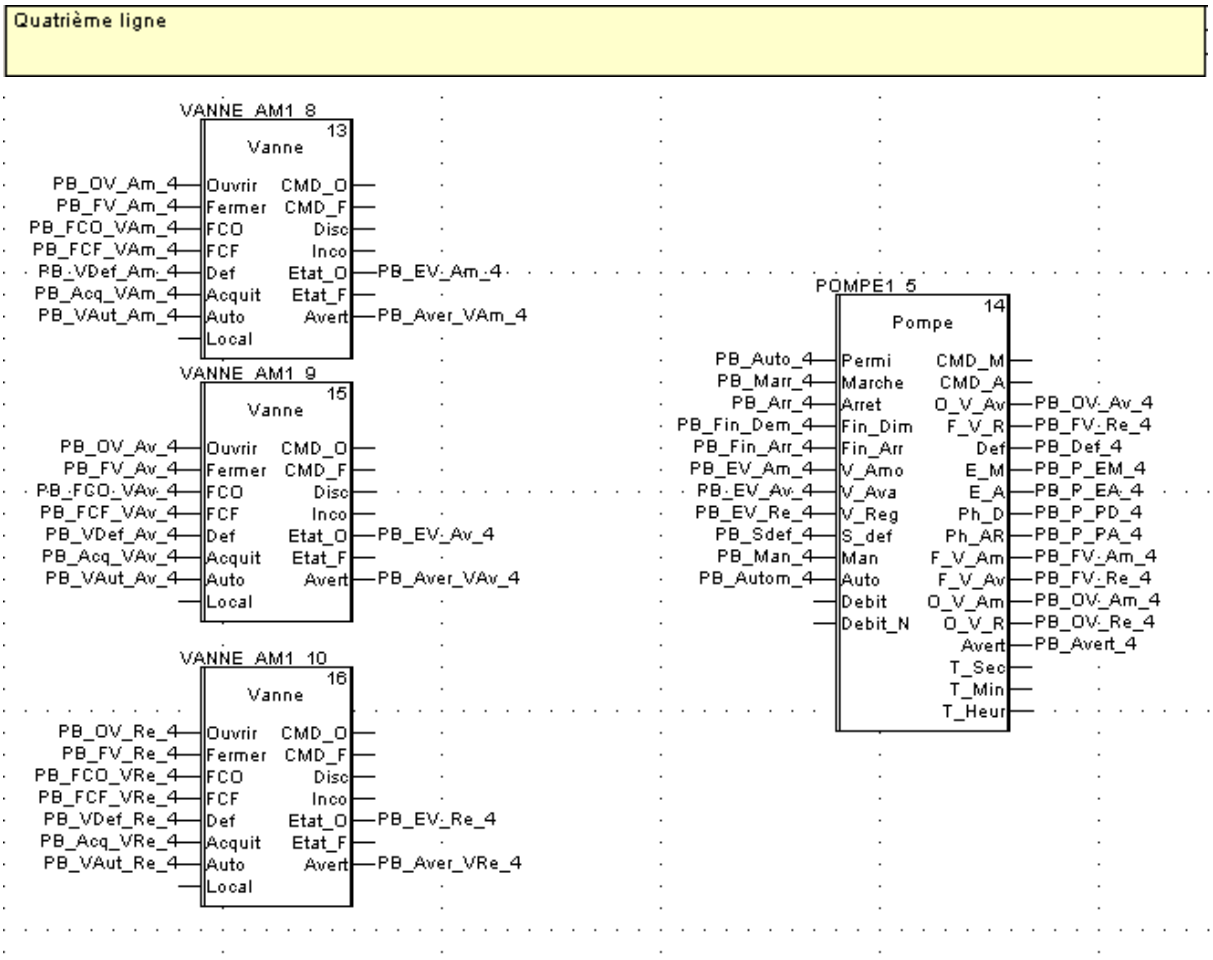


Figure V.9 : Programme gestion des pompes boosting

V.7.4. Programme de gestion de la section stockage

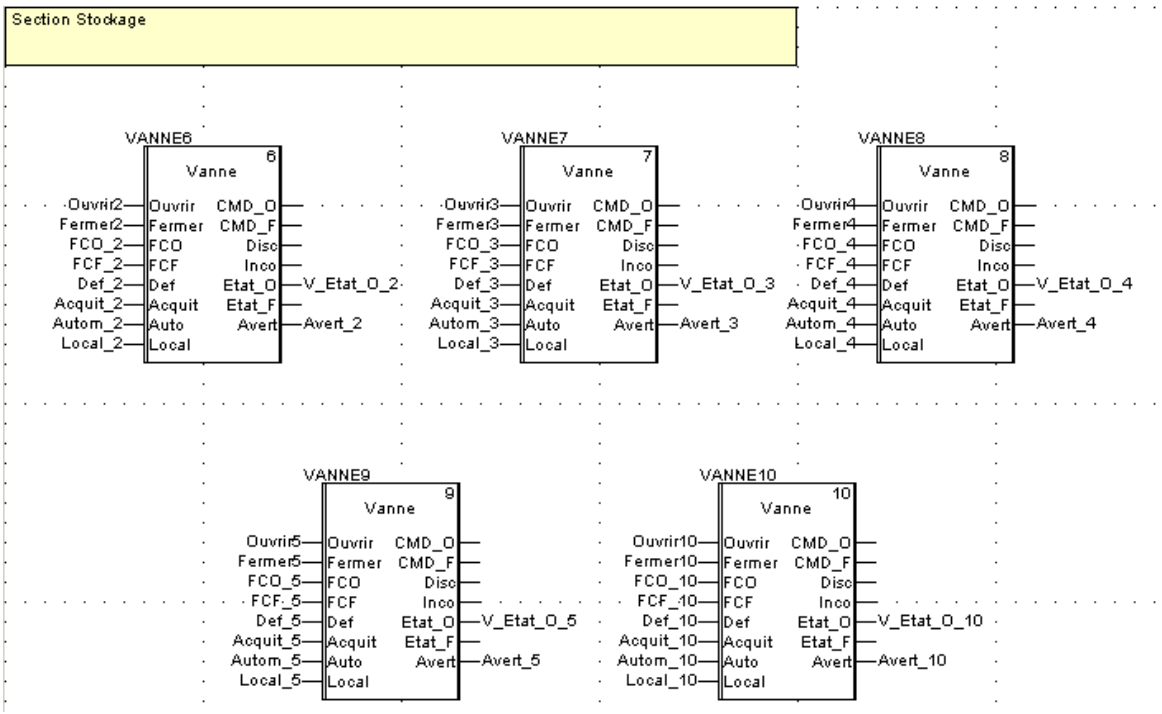


Figure V.10 : Programme de gestion de la section stockage

V.7.5. Programme de gestion de la section entrée station

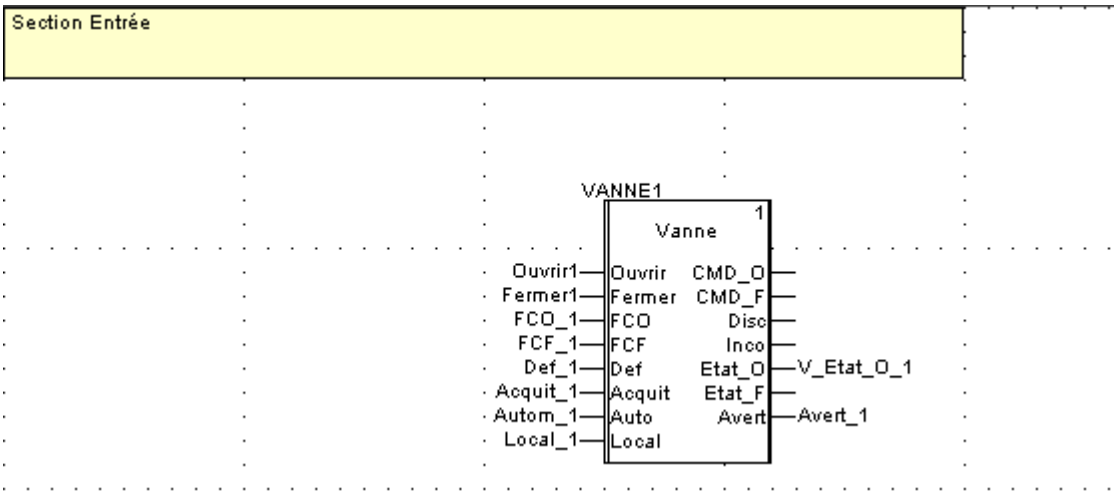


Figure V.11 : Programme de gestion entrée station

V.7.6. Programme de gestion de la section filtration

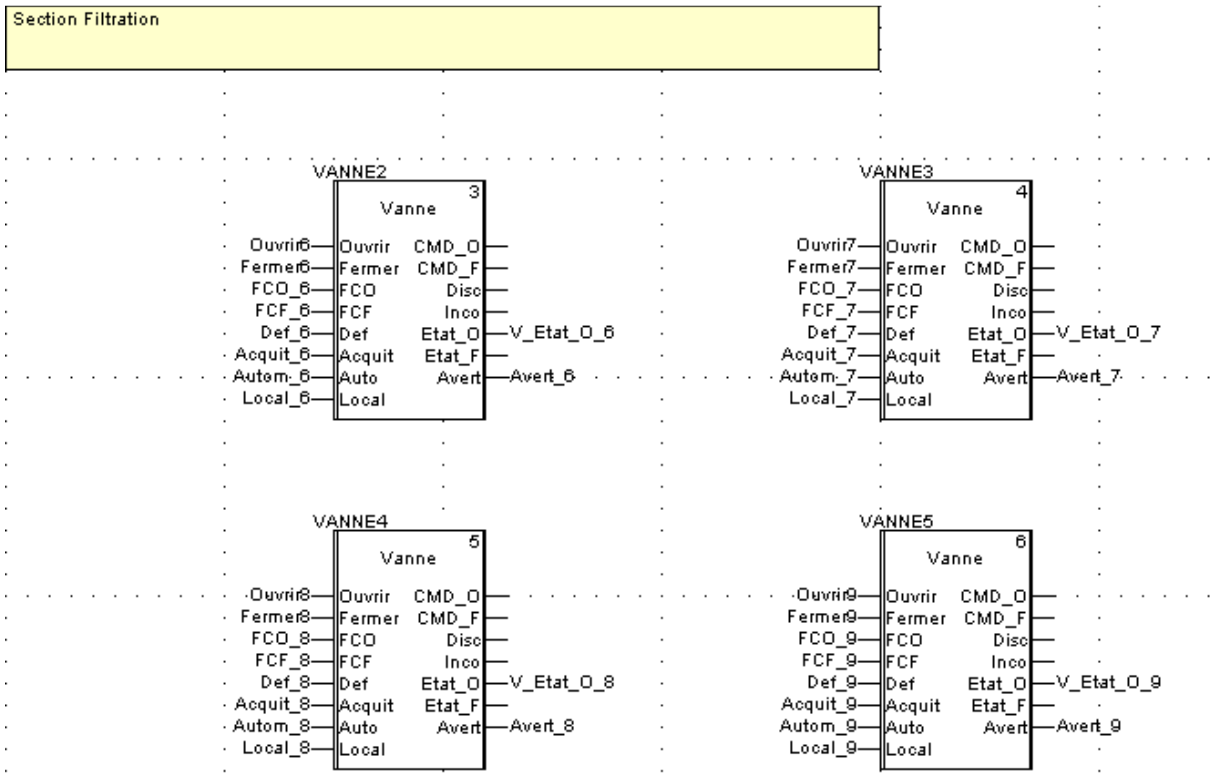


Figure V.12 : Programme de gestion section filtration

V.8. Création des objets

Pour cette partie, on a créé des objets représentant la vanne motorisé et le groupe électropompe. Pour ce faire, on a utilisée les génies et les super génies en respectant la représentation normalisé des objets.

V.8.1. Vanne motorisé

Le génie de la vanne motorisé est représenté dans la figure suivante :

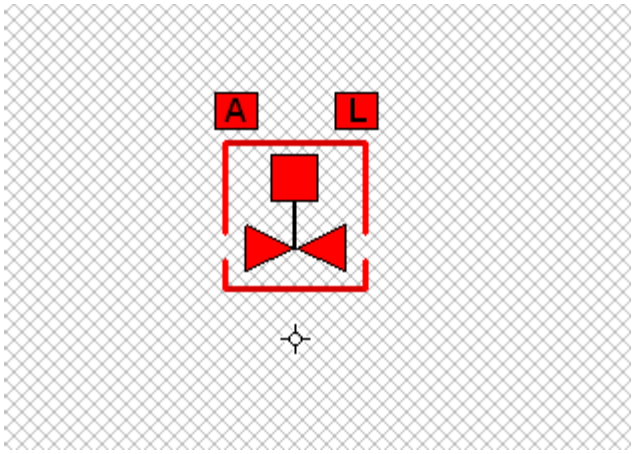


Figure V.13 : Génie vanne motorisé

▪La vanne est de couleur rouge quand elle est fermée et en vert quand elle est ouverte.

▪Si la vanne est en mode automatique, le carré ou se trouve la lettre A devient de couleur verte et si elle est en mode local, le carré ou se trouve la lettre L devient de couleur verte.

▪Le rectangle qui entoure la vanne est utilisé pour clignoter en cas de présence de défaut.
▪A l'exécution, un clic sur la vanne permet de faire apparaitre le super génie de cette dernière, qui représenté si après :

Figure V.14 : Super génie vanne_1

▪Comme la figure le montre, ce super génie nous permet de savoir l'état de la vanne (ouverte ou fermé), mode de travail de la vanne (local ou automatique), des boutons pour envoyer les commandes d'ouverture ou de fermeture, un bouton pour acceder à la page des alarme, un bouton pour l'acquittement des défauts et enfin un bouton quitter qui permet de fermer la fenetre.
▪Ce super génie est utilisé dans le cas ou la vanne est piloté par l'opérteur. Si les commandes d'ouverture ou de fermeture sont envoyées par un autre programme alors on a pas besoin des bouton Open et Close, alors on a crée un autre super génie qui est le suivant :

Figure V.15 : Super génie vanne_2

▪Dans notre cas, on a besoin aussi d'une vanne progressive, on a pri le meme programme de gestion mais la representation est différente comme le montre la figure suivante :

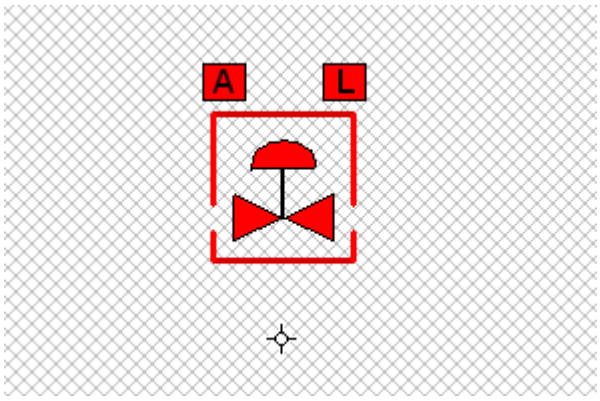


Figure V.16 : Génie de la vanne progressive

▪Les supers génies de cette vanne est les meme que les précedents.

V.8.2. GEP avec variateur de vitesse

Le génie du groupe électropompe est représenté dans la figure suivante :

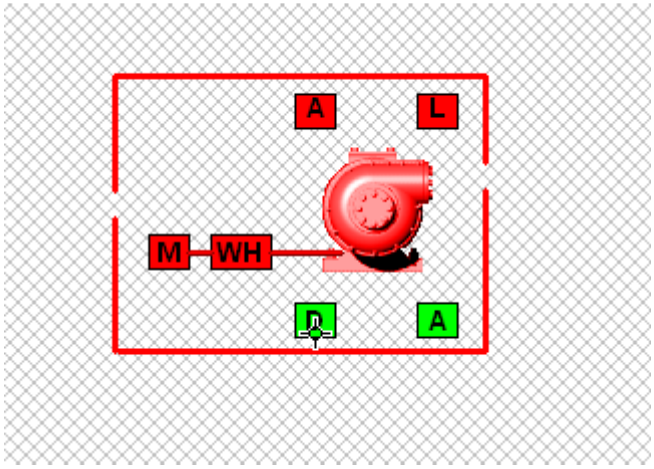


Figure V.17 : Génie GEP_1

▪Comme pour la vanne, la couleur verte de la pompe indique que celle ci est en état de marche et la couleur rouge indique qu'elle est à l'arret.
▪Les carrés au dessus de la pompe ont les meme propriètés que ceux de la vanne (mode local ou automatique).
▪Les carrés au dessous de la pompe indiquent si la pompe en phase de demmarage ou en phase d'arret.
▪Les carrés à gauche indiquent le moteur ainsi que le variateur de vitesse.
▪A l'execution, un clic sur le dessin ouvre la fenetre suivante (super géne pompe) :

Figure V.18 : Super génie pompe

▪Comme celui de la vanne cu super génie permet de voir l'état de la pompe (en marche ou à l'arret), mode de travail de la pompe (local ou automatique).

▪Un bouton pour envoyé la commande marche et un autre pour l'arret.

V.8.3. GEP sans variateur de vitesse

Le génie du groupe electropompe sans variateur de vitesse(utilisé pour les pompes boosting) est le suivant :

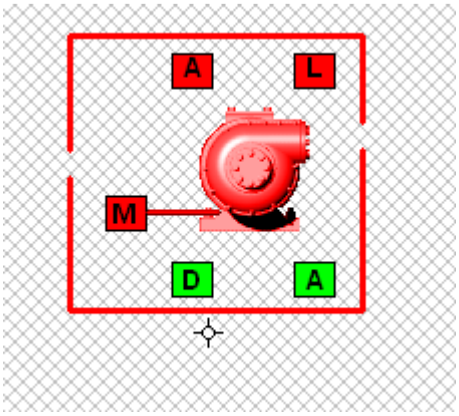


Figure V.19 : Génie GEP_2

▪On a les memes proprietés que le génie precedent et on utilise le meme super génie.

V.8.4. Pages principales

On a à choisir entre deux stations et on les a nommées station pompage1 et station pompage2 (comme le montre le sinoptique suivante) , qui communiquent avec le terminale ARZEW par un reseau Ethernet / TCP/IP/modbus qui sera dailleurs detaillé dans les prochaines sections.

Figure V.20 : choix station

V.8.4.1. Pages principales pour la station 1

Pour les pages de cette station de pompage, on a utilisé un seul modèle et on a representé le mieux possible les differentes sections existantes comme le montres les figures qui suivent :

▪La figure qui suit représente le synoptique des lignes d'expedition :

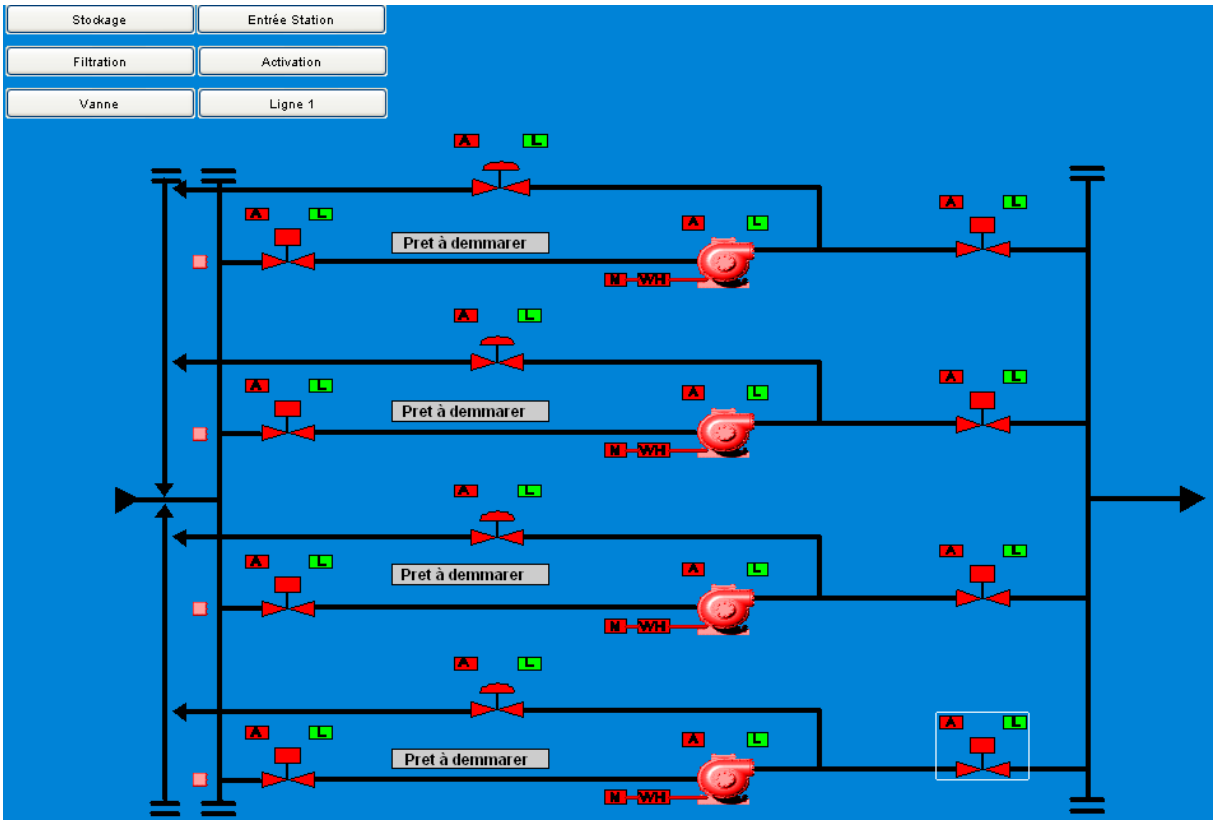


Figure V.21 : Pomperie principale

▪Comme le montre la figure, toutes les pompes ainsi que les vannes sont en mode local, aucune ligne n'est activé et ça est reperé par la couleur rouge des carrés en face des ligne, aucun défaut n'est signaler et ça est reperé par l'expression « Prêt à demmarer ».

▪La figure suivante représente un zoom de la première ligne avec les differents fin de course pour la simulation :

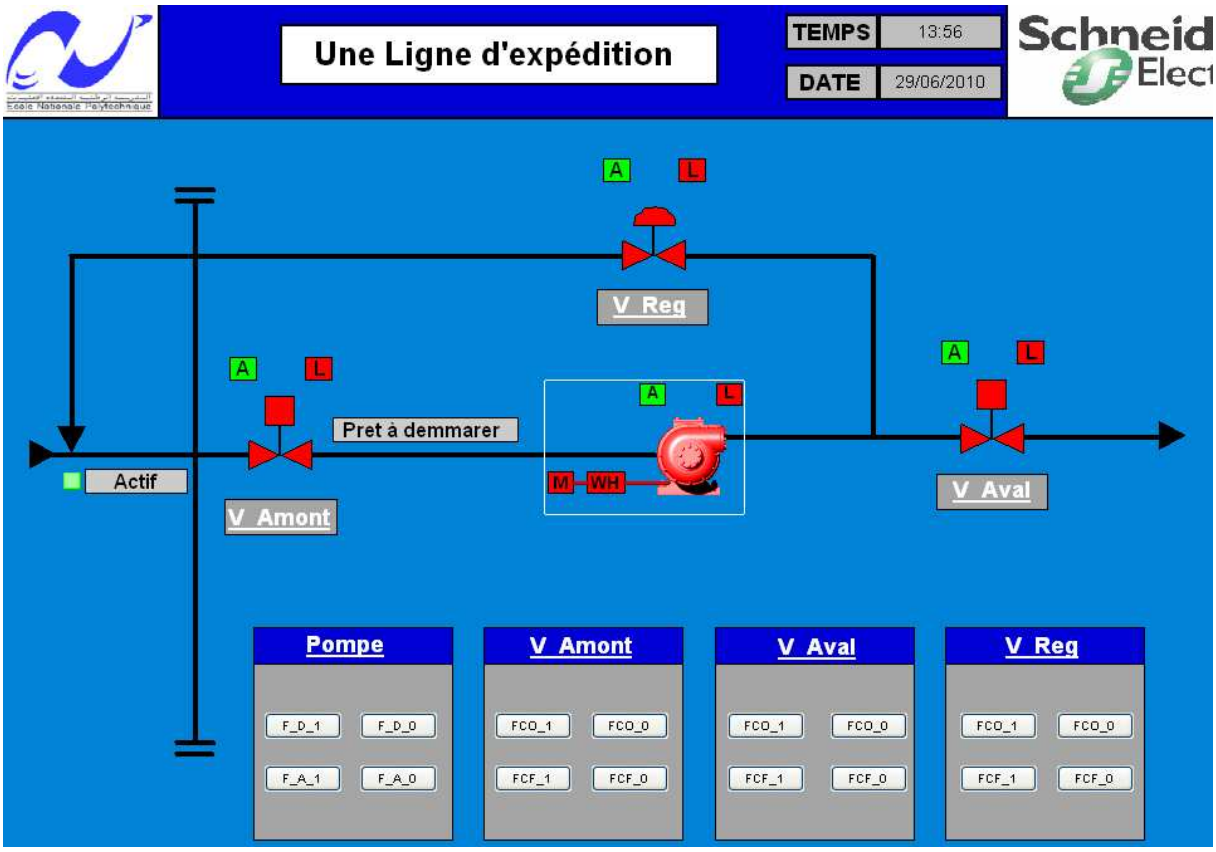


Figure V.22 : Représentation d'une ligne d'expedition

▪On a aussi crée une page pour la simulation de la vanne, avec les fin de coures, qui est la suivante :

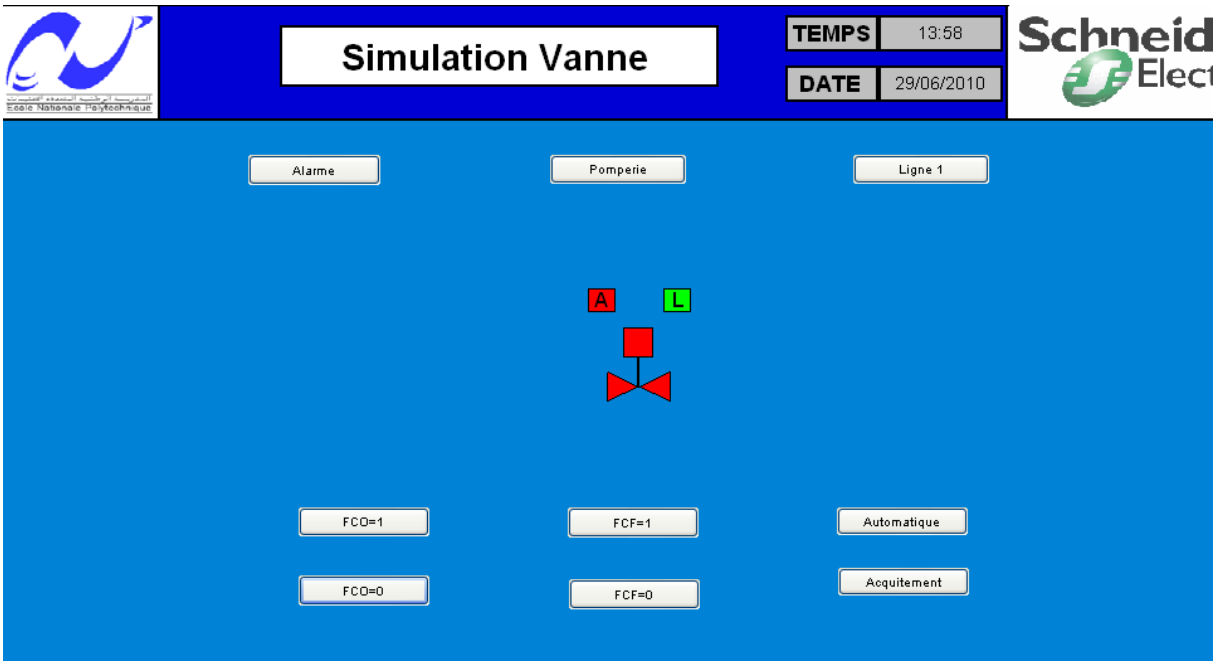


Figure V.23 : Page vanne

▪Une autre page pour le choix du régime de fonctionnement ainsi que l'activation des lignes d'expédition selon le temps de marche et le régime :

Figure V.24 : Page Activation

▪Il faut noter qu'on a ajouter des boutons permettant ainsi d'aller d'une page à l'autre.

▪Une page représentant la pomperie boosting :

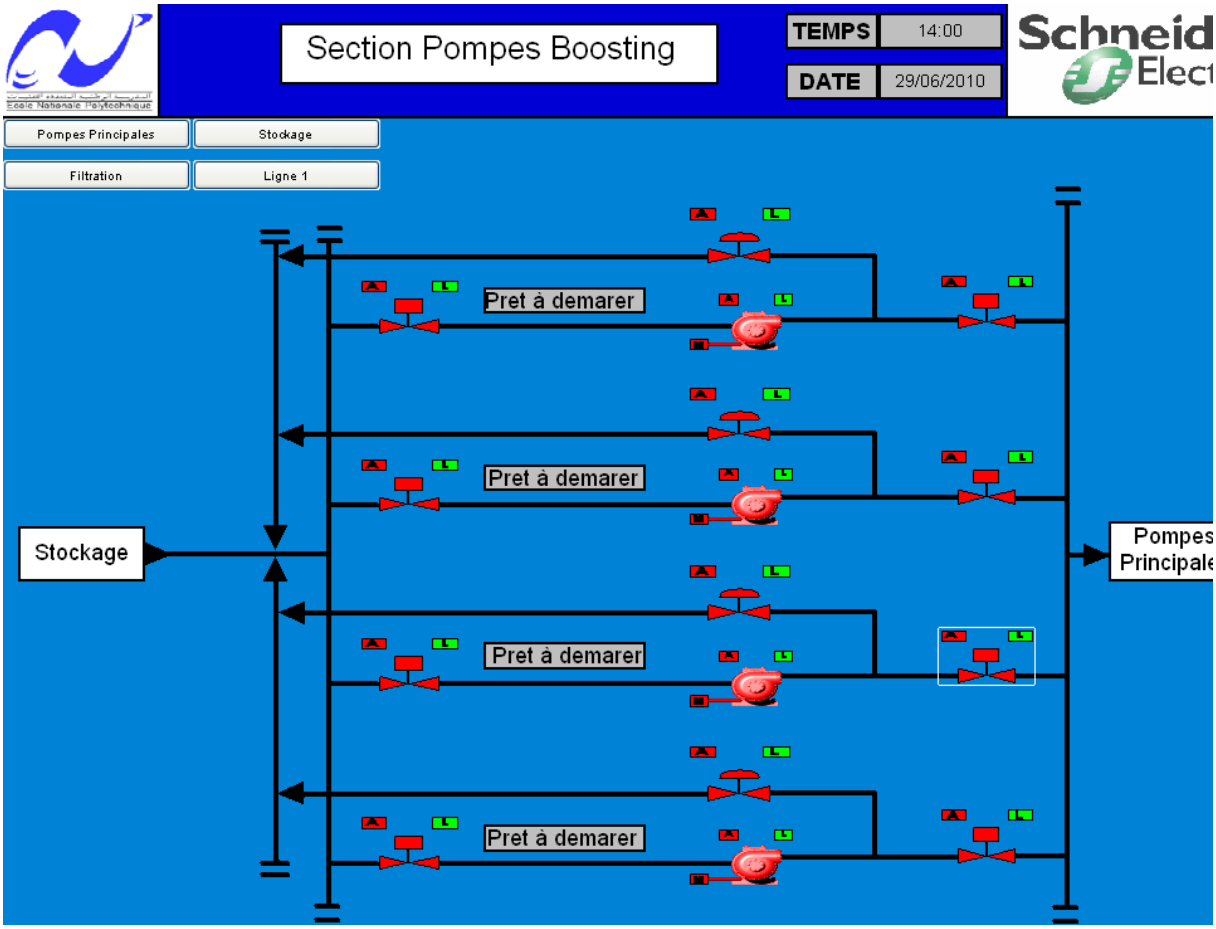


Figure V.25 : Pomperie boosting

▪Une page pour la représentation de l'entrée de la station qui est la suivante :

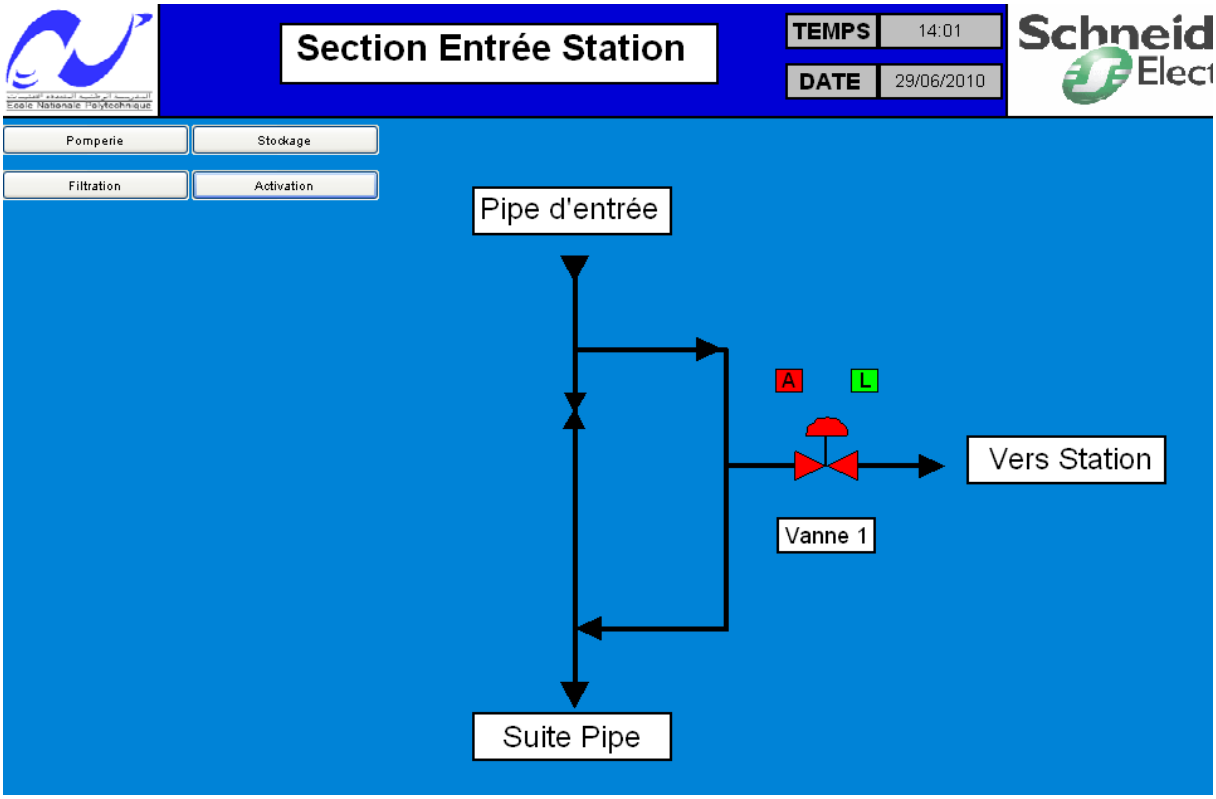


Figure V.26 : Page entrée station

▪Une page pour la section stockage qui est la suivante :

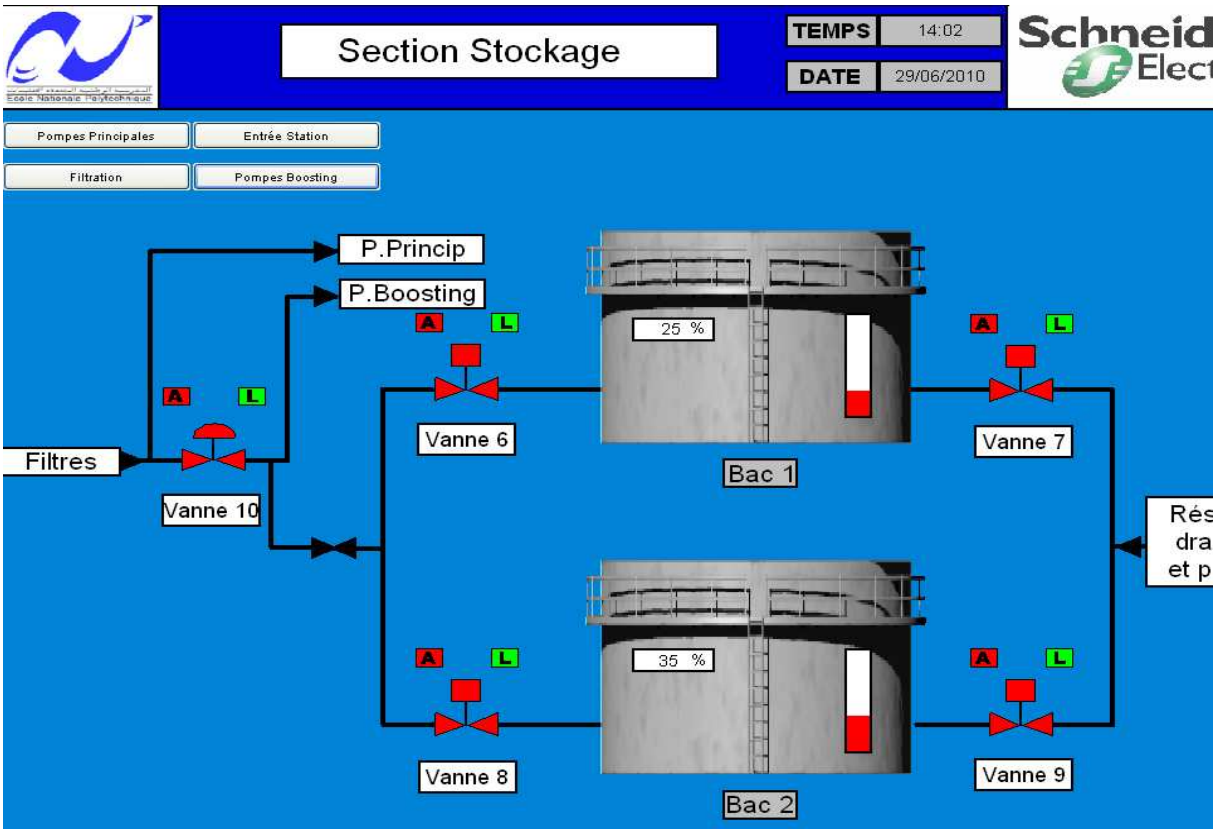


Figure V.27 : Page section stockage

▪Une page pour la section filtration qui est la suivante :

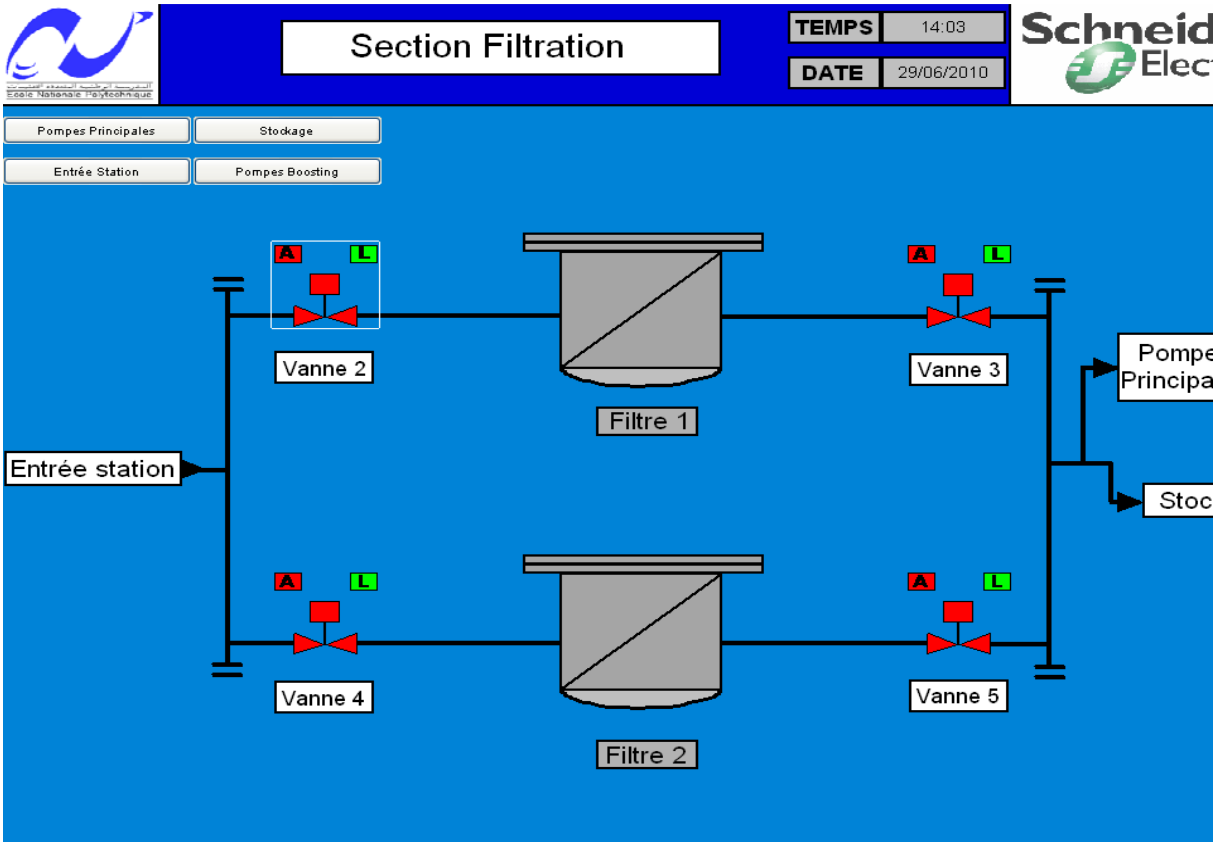


Figure V.28 : Page section filtration

V.8.4.2. Pages principales pour la station2

Les figures qui suivent représentent les synoptiques de toute la station de pompage2:

V.8.4.2.1. Page vue générale

Dans cette synoptique on a un aperçu sur toute la station . de cette vu on peut voir les différentes sections de la station et distinguer les pompes allumées de celles ouvertes, et méme chose à propos des vannes. on peut agir sur la vanne d'entrée et celle allant vers les pompes principales. On peut aussi aperçevoir les defauts des differentes sections .

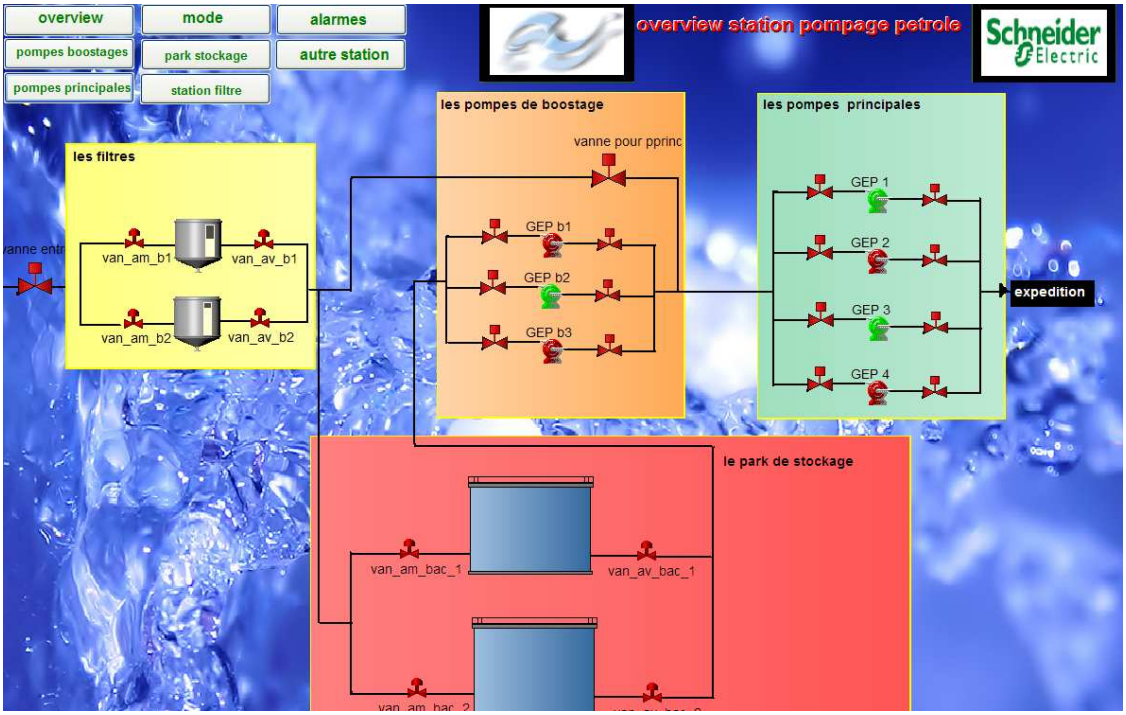


Figure V.29 : Page vue générale

V.8.4.2.2. Page choix du mode

Dans cette section on peut choisir les modes de fonctionnement de la station .

Figure V.30 : Page choix du mode.

V.8.4.2.3. Page des pompes boostages

Dans cette section on peut choisir les pompes de boostage deja autorisé à allumer et celles à etteindre. On peut aussi voir le temps de marche de chaque pompe. Et voir exactement quel genre de défaut qui atteind chaque pompe ou que vanne puis l'acquitter si c'est possible.

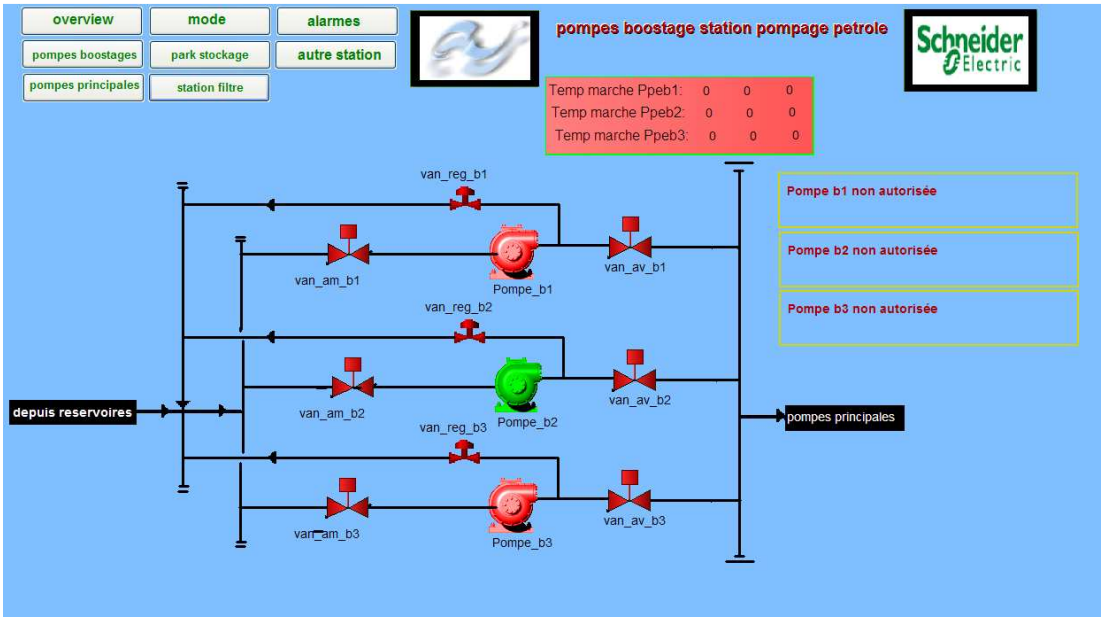


Figure V.31 : Page des pompes boostages.

V.8.4.2.4. Page des pompes principales

Dans cette section on peut choisir les pompes principales déjà autorisé à allumer et celles à etteindre. On peut aussi voir le temps de marche de chaque pompe. Et voir

exactement quel genre de defaut qui atteind chaque pompe ou que vanne puis l'acquitter si c'est possible.

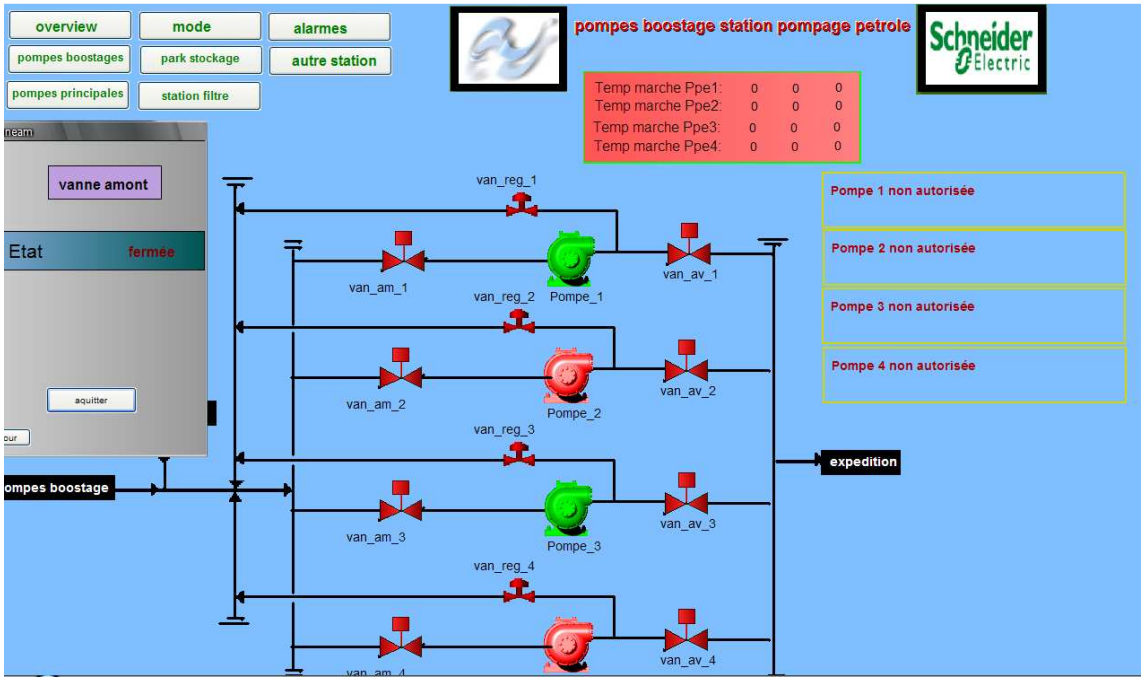


Figure V.32 : Page des pompes principales.

V.8.4.2.5. Page du parc de stockage

Dans cette section on peut aperçevoir le niveau des deux bacs de stockages et l'état des quatre vannes .

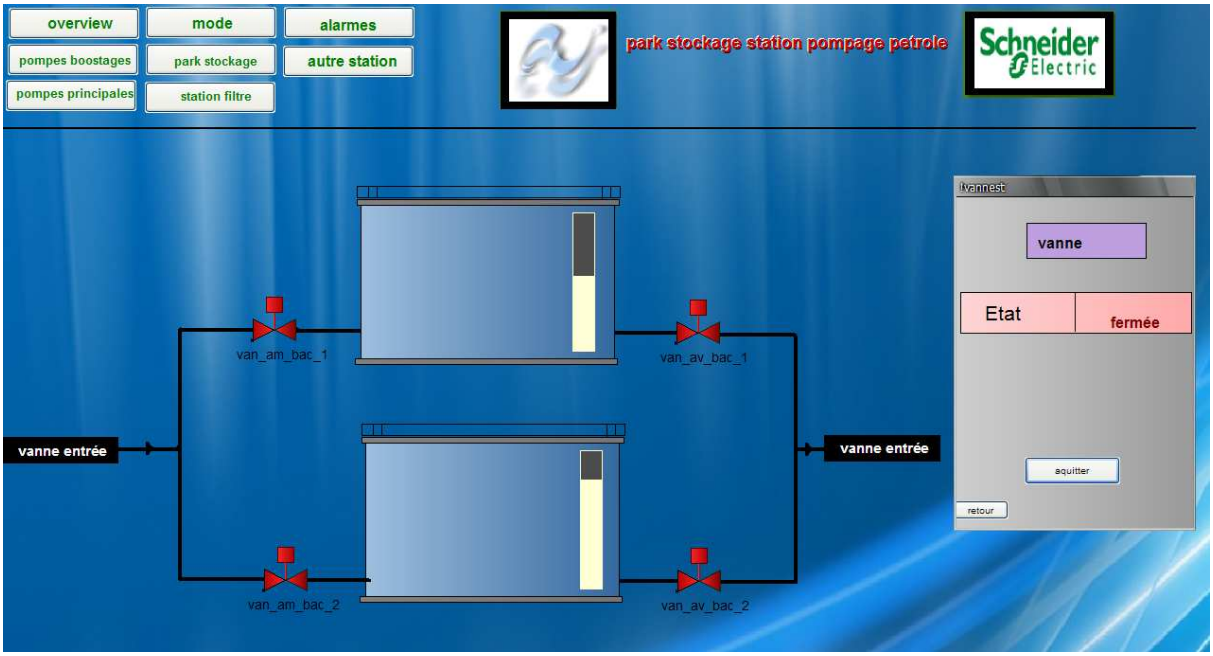


Figure V.33 : Page du park de stockage.

V.8.4.2.6. Page de la section filtrage

Dans cette section on peut aperçevoir le niveau des deux bacs de filtrage et l'etat des quatre vannes aussi.

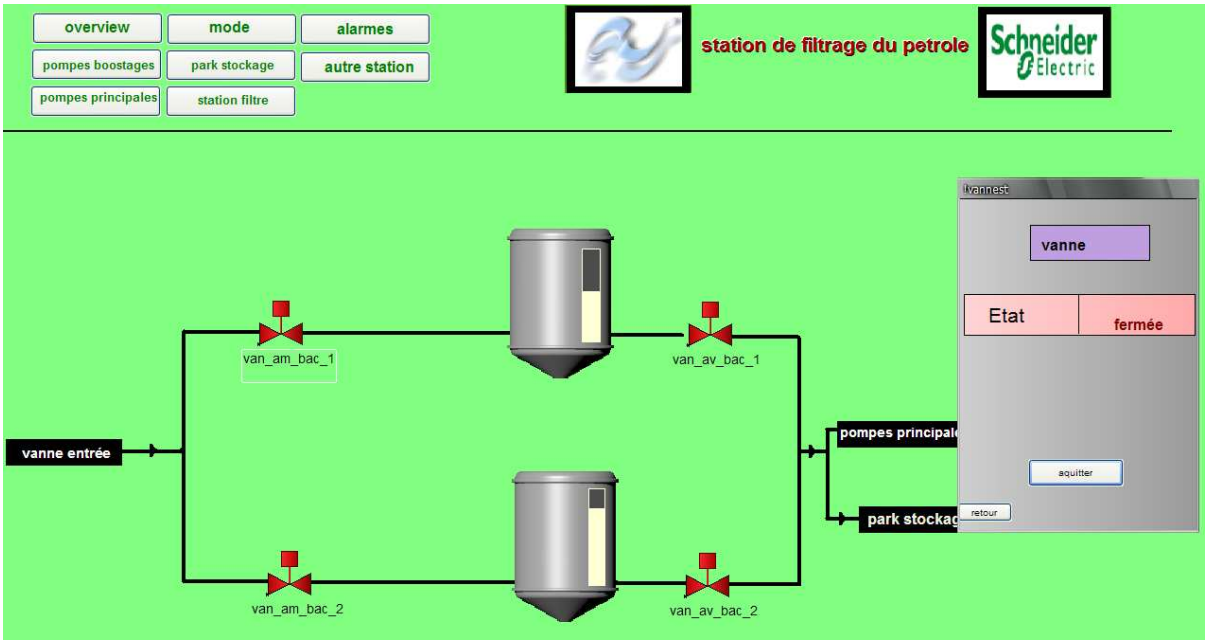


Figure V.34 : Page de la section filtrage.

V.9. Simulation

Pour la simulation, on utilise les paramètres de configuration, concernant la communnication, décrits dans le chapitre III.

V.9.1. Simulation Vanne

Concernant la simaltion de la vanne, soit on utilise la page vanne(Figure IV.25) ou bien l'écran d'exploitation suivant, qu'on a crée avec Unity pro :

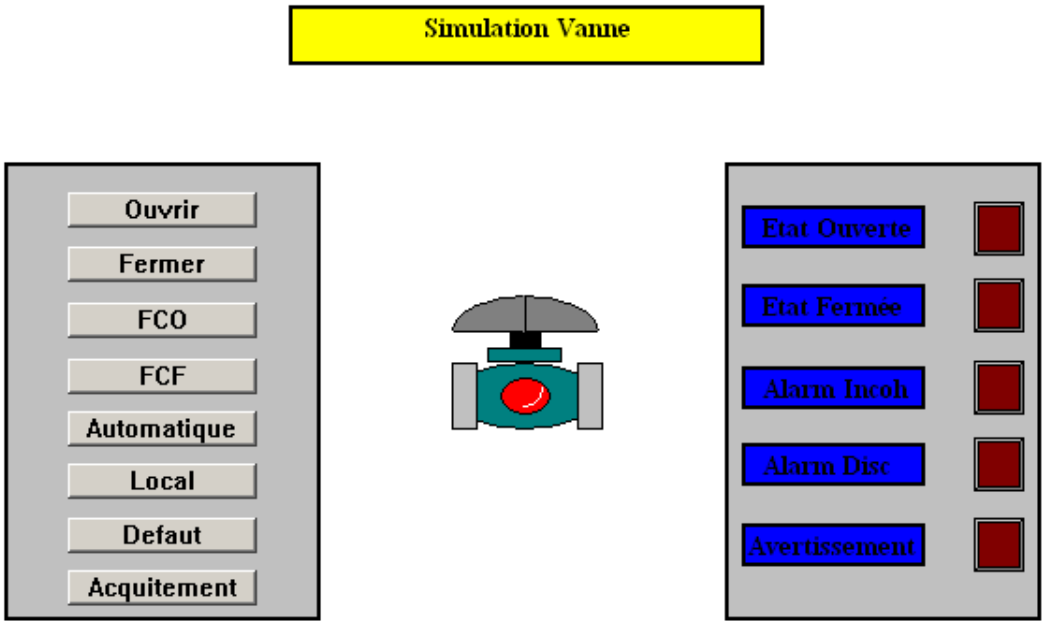


Figure V.35 : Ecran d'exploitation vanne

▪Dans notre cas, on ne va pas utiliser l'ecarn d'exploitation mais on utilise la page crée dans Vijeo Citect.

▪Avant d'envoyer les commandes, on passe en mode automatique en cliquant sur le bouton automatique et on ouvre le super génie de la vanne et on aurala figure suivante :

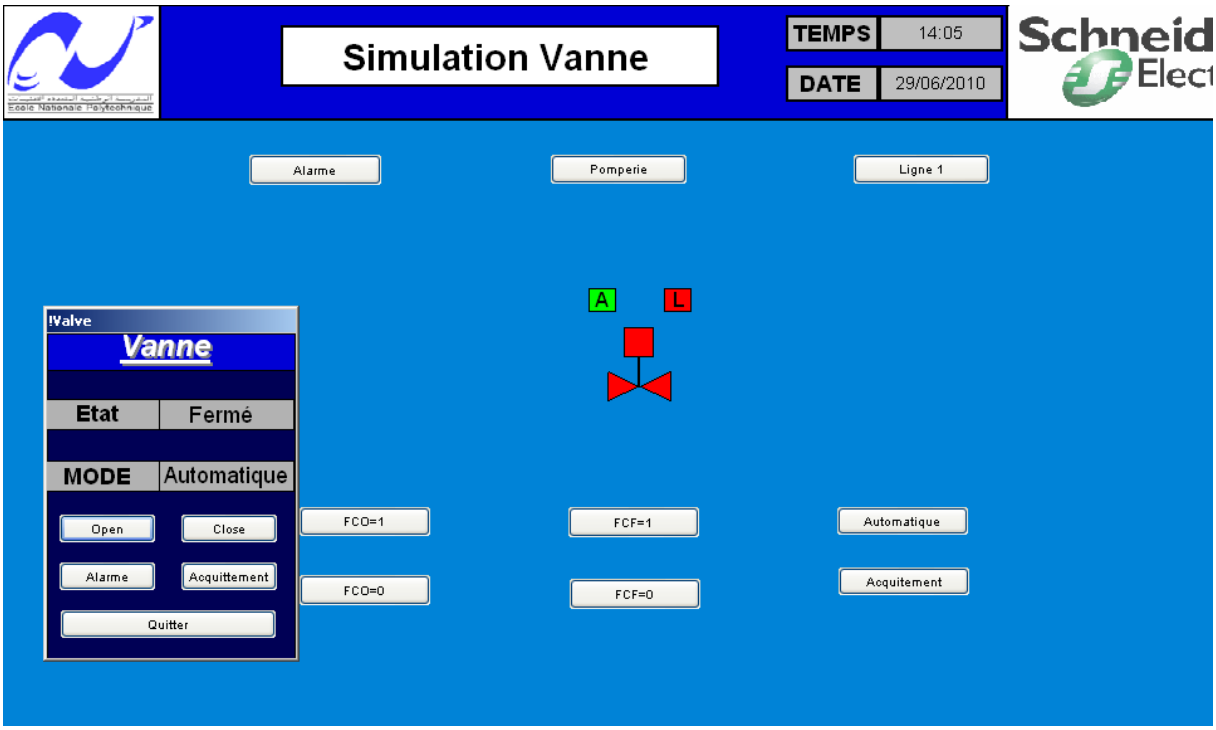


Figure V.36 : Simulation vanne_1

▪Comme la figure le montre, on bien au mode automatique et la vanne est fermé et ça indiqué par les coouleurs correspondante ainsi que par le super génie.

▪On envoie la commande ouvrir en cliquant sur le bouton Open et on met le fin de course d'ouverture FCO à 1 pour ne pas avoit de defaut et on aura la figure suivante :

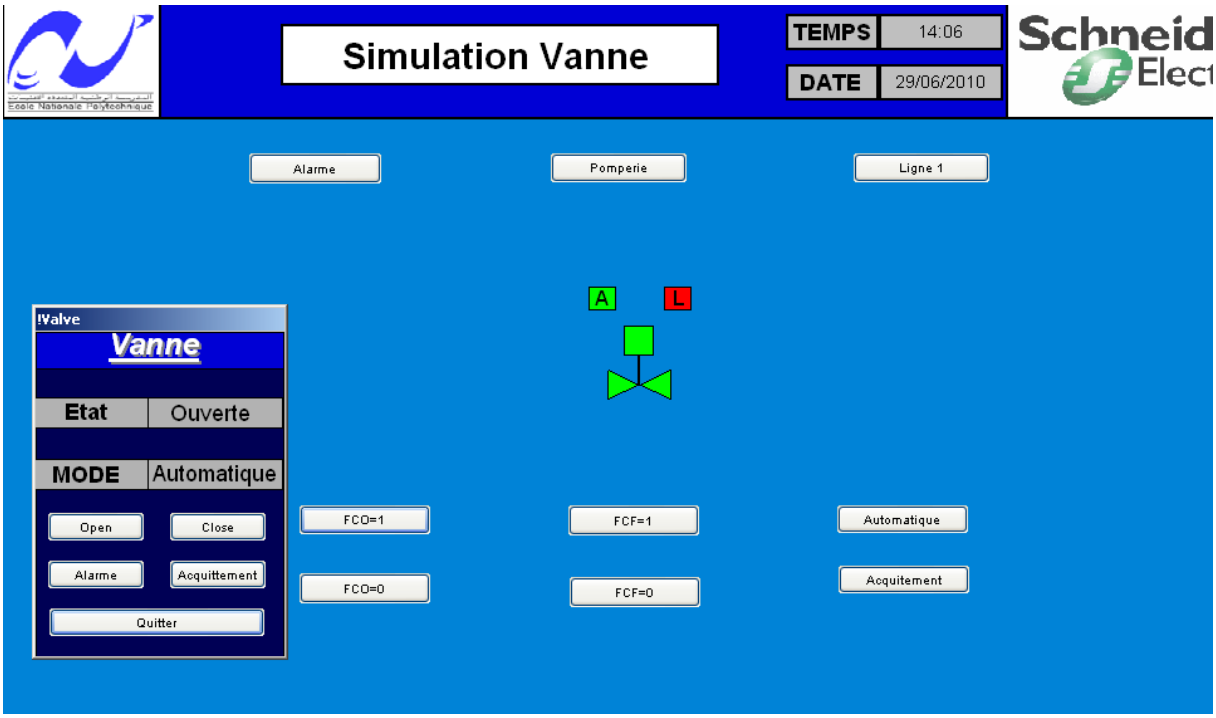


Figure V.37 : Simulation vanne_2

▪Comme la figure le montre, la vanne est ouverte.

▪On va généré un defaut et on aura la figure suivante :

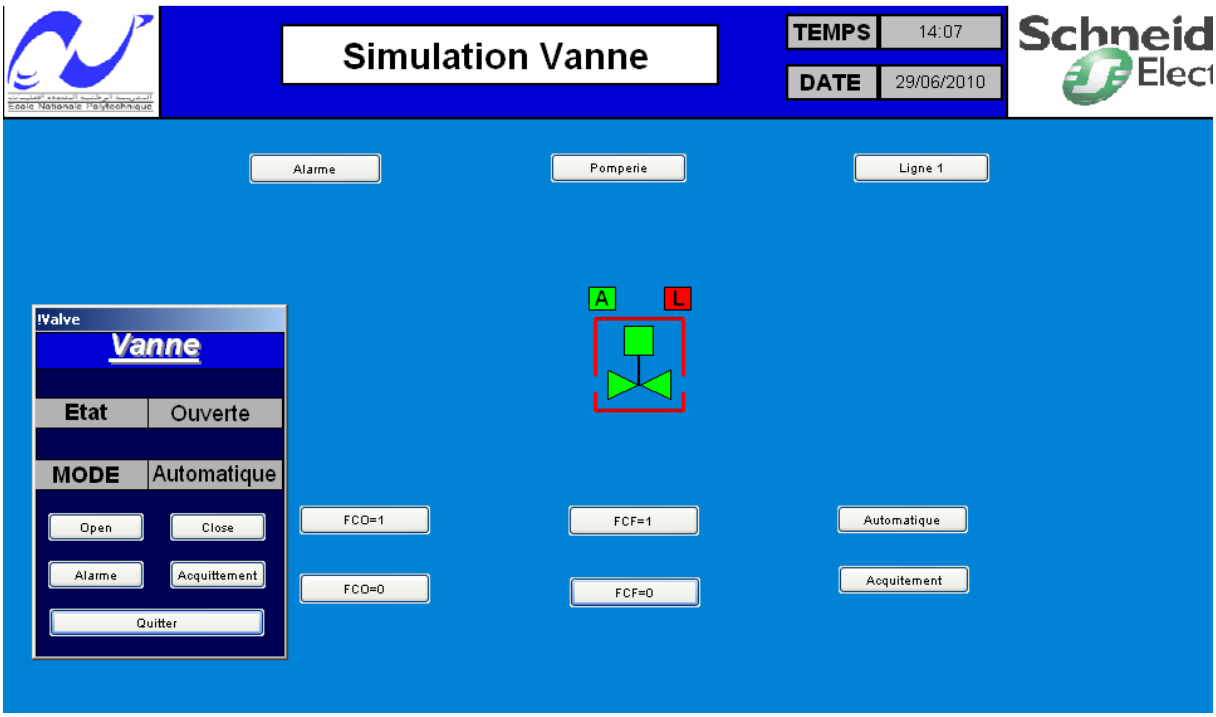


Figure V.38 : Simulation vanne_3

▪On voit bien le rectangle qui apparait pour indiquer la présence d'un defaut.il faut noter qu'au mode d'execution, ce rectangle clignante, chose faite expré pour mieux attirer l'attention de l'opérateur.

▪Comme ce defaut on l'a declarer comme une alarme, en cliquant sur le bouton alarme, on aura la figure suivante :

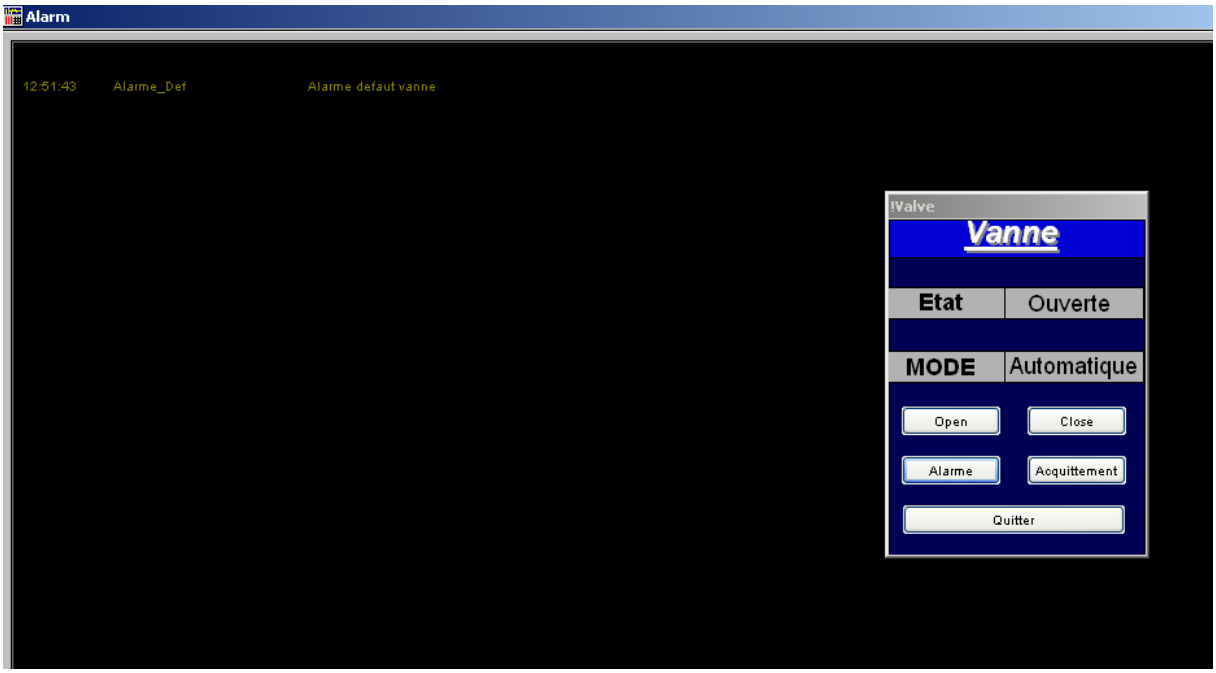


Figure V.39 : Simulation vanne_4

▪Comme la figure le montre, ce defaut est bien enregistré dans la page d'alarme.

V.9.2. Simulation d'une ligne d'expédition

La figure suivante représente une étape de simulation concernant la ligne d'expédition :

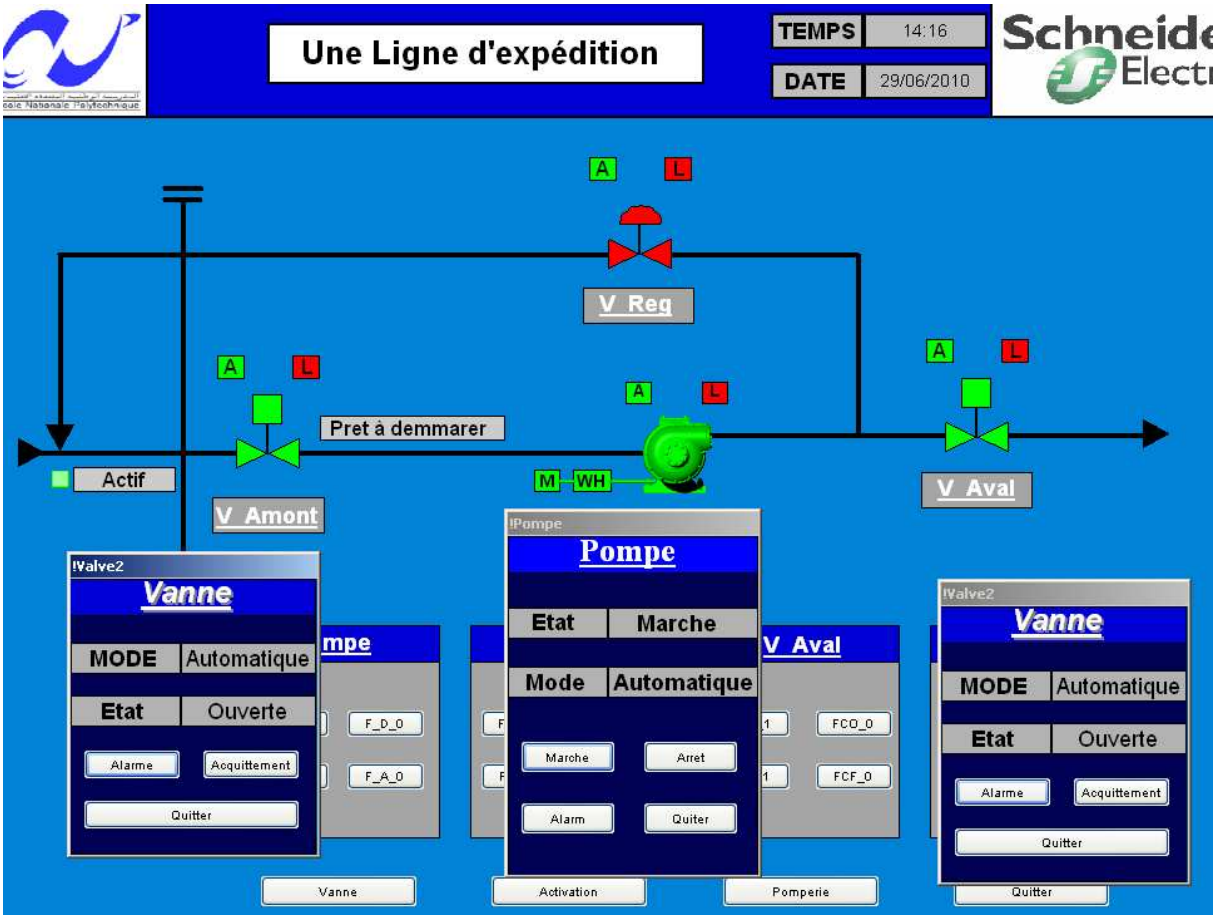


Figure V.40 : Simulation d'une ligne d'expédition

V.10. Architecture proposée

Pour un projet de supervision, définir une architecture réseau est primordial car il faut établir la communication entre les stations et le terminal d'arrivée, ce dernier aura accès à toutes les informations disponible sur les stations et aussi peut envoyer des commandes. Dans ce qui suit, on donnera un aperçu de l'architecture réseau au niveau des stations ainsi que l'architecture générale.

V.10.1. Architecture au niveau des stations

La figure qui suit montre l'architecture au niveau d'une des stations (même principe pour les autres) :

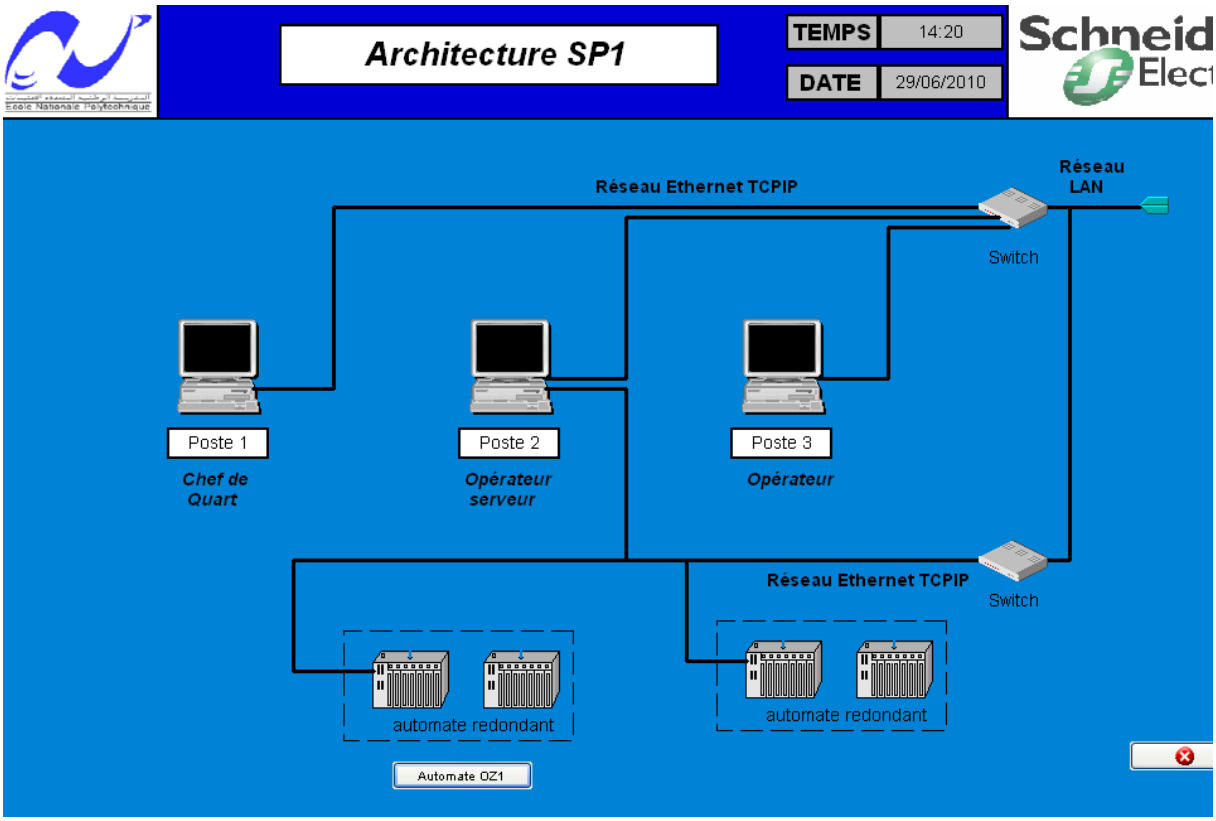


Figure V.41 : Architecture au niveau des stations

▪Comme le montre la figure, on a un réseau local (LAN) et le protocole utilisé est l'Ethernet TCPIP, qui est le plus utilisé dans l'industrie à cause de sa performance.

▪On remarque aussi qu'il y'a des postes serveurs qui ont accès aux données des automates et des opérateurs qui visualisent les différents synoptiques de la station, il faut noter que l'utilisation de Vijeo Citect comme logiciel de supervision permet de configurer les postes en clients ou en serveur et aussi limite les accès aux données en utilisant des mots de passe.

▪Le nombre d'automate diffère d'une station à une autre et ça selon les données à traiter.

▪La figure qui suit montre l'architecture du terminal d'arrivée :

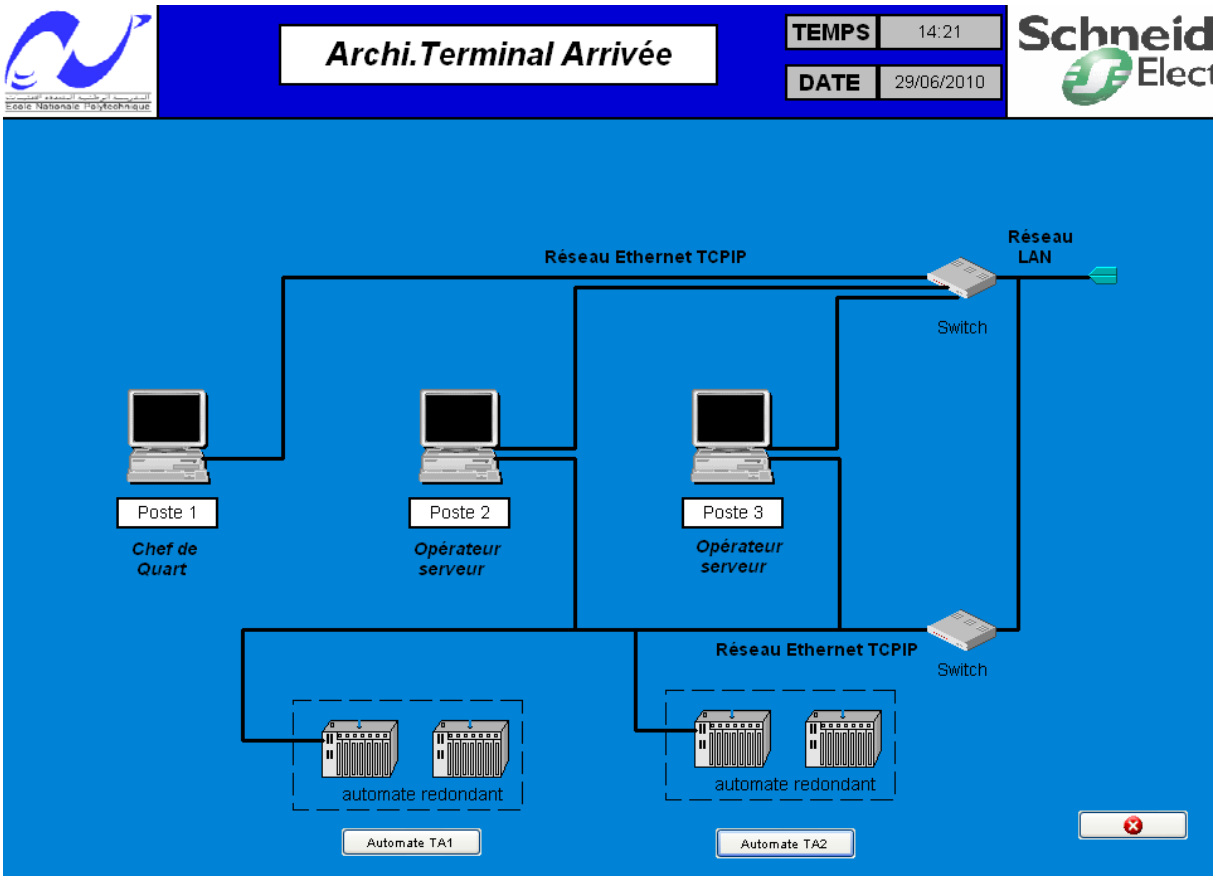


Figure V.42 : Architecture du terminal d'arrivée

V.10.2. Architecture générale

La figure qui suit donne une idée sur l'architecture générale entre les différentes stations et le terminal d'arrivée (on a pris uniquement deux stations) :

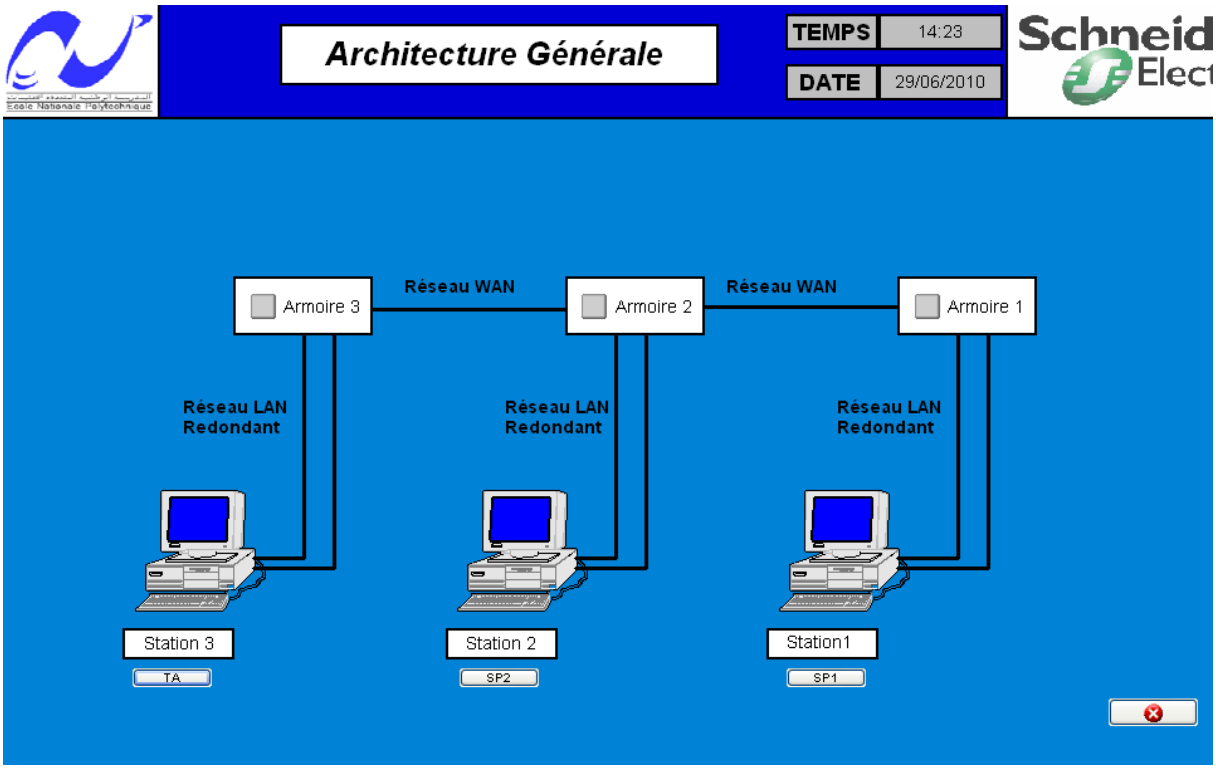


Figure V.43 : Architecture générale

▪Comme le montre la figure, les différentes stations sont reliées entre elles par réseau étendu (WAN), les protocoles utilisés sont : SDH, OTN et DWDM et le câblage se fait essentiellement par la fibre à optique.

V.11. Régulation de débit

Dans ce qui suit, on essayera de donner un aperçu sur la technique de régulation de débit ainsi que les différents constituant de la boucle de régulation et ça en tenant compte que l'algorithme de calcul est un bloc dans l'automate.

La régulation de débit peut se faire selon deux manières : la première repose sur la mesure de débit directement et la deuxième sur la mesure de pression, dans notre cas, on s'intéressera qu'à la première méthode.

V.11.1. Constitution d'une boucle de régulation [10]

La régulation regroupe l'ensemble des techniques utilisées visant à contrôler une grandeur physique. Exemples de grandeur physique : Pression, température, débit, niveau,…

▪Une boucle de régulation doit comporter au minimum les éléments suivants :

- Un capteur de mesure ;
- Un transmetteur souvent intégré au capteur ;
- Un régulateur ;
- Un actionneur.

▪Elle est souvent compléter par :

- Un enregistreur ;
- Des convertisseurs ;
- Des sécurités ;

▪Le schéma de principe d'une boucle de régulation est représenté dans la figure suivante :

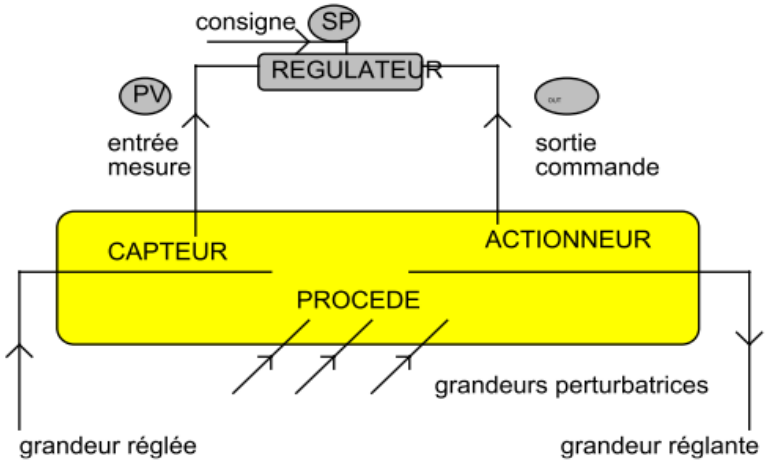


Figure V.44 : Schéma d'une boucle de régulation

▪La grandeur réglée, c'est la grandeur physique que l'on désire contrôler. Elle donne son nom à la régulation, par exemple : une régulation de température.

▪La consigne : C'est la valeur que doit prendre la grandeur réglée.

▪La grandeur réglante est la grandeur physique qui a été choisie pour contrôler la grandeur réglée. Elle n'est généralement pas de même nature que la grandeur réglée.

▪Les grandeurs perturbatrices sont les grandeurs physiques qui influencent la grandeur réglée. Elles ne sont généralement pas de même nature que la grandeur réglée.

V.11.2. Nature des signaux utilisés

Les signaux reçus et transmis par le régulateur doivent être normalisés afin de permettre l'interchangeabilité du matériel, ils peuvent être de nature différente :

- Electriques ;
- Pneumatiques ;
- Numériques ;
- Plus rarement hydraulique.

▪Ces signaux sont normalisés comme illustré si dessous :

NATURE DU SIGNAL ANALOGIQUE	SYMBOLE	VALEURS NORMALISEES LES PLUS COURANTES
non définis	———	
électriques	·····	4<i<20 mA
pneumatiques	// // //	200<P<1000mb
numériques	– – – – – – – – –	codées sur 8, 16, 32, ... bits

En électrique (intensité) :

0% échelle ----> 4 mA
100% échelle ----> 20mA

En pneumatique (pression) :

0% échelle ----> 200 mb (3 psi)
100% échelle ----> 1000mb (15 psi)

▪En fonction de la nature des capteurs, des actionneurs et des régulateurs (analogique ou numérique), des convertisseurs sont indispensables à différents points de la boucle pour normaliser les signaux.

V.11.3. Capteur de mesure

Un capteur est l'élément d'un appareil de mesure servant à la prise d'informations relatives à la grandeur à mesurer.

Le capteur est l'élément capital et le premier maillon d'une chaîne de mesure, Il a pour rôle de saisir et de transformer la grandeur physique à mesurer (ou mesurande) et le contenu de son information en une autre grandeur physique accessible aux sens humains ou aux maillons suivants de la chaîne d'acquisition.

Ce sont donc des organes sensibles, transformant la grandeur à mesurer en un signal électrique, pneumatique, hydraulique ou numérique, normalisé, représentatif de l'information originelle.

Cette transformation nécessite généralement un apport d'énergie extérieure au système. En règle générale, l'élément sensible du capteur est lié à un traducteur ou transducteur permettant la transformation du déplacement ou de la déformation de cet élément sensible en un signal ou une indication de mesure.

▪La figure suivante montre le schéma principe d'un capteur :

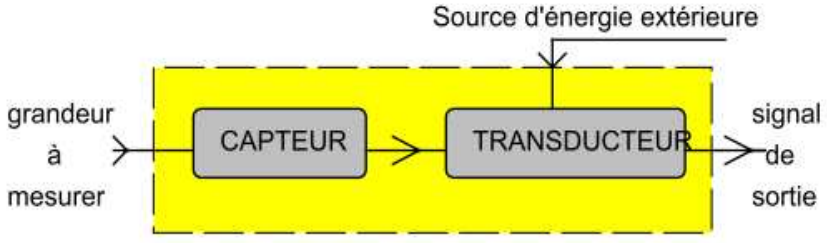


Figure V.45: Schéma principe d'un capteur

Par simplification, l'ensemble est appelé CAPTEUR.

▪Concernant la mesure de débit, il existe plusieurs techniques, on trouve essentiellement :

- Des compteurs volumétriques ;
- Des turbines ;
- Des débitmètres électromagnétiques, à effet Vortex, à ultrasons, massiques, à cible ;
- Des tubes de Pitot et de Prantl ;
- Des organes déprimogénes : diaphragme, venturi, tuyère.

V.11.4. Organes de commande (Actionneurs)

Ils sont appelées aussi les servomoteurs, leur fonction est de convertir la commande en une énergie mécanique.

Les technologies rencontrées pour les actionneurs sont : pneumatique, hydraulique ou électrique et les commandes arrivant sur les actionneurs sont le plus souvent:

- Electrique (4-20 mA, 0-20 mA) ;
- Pneumatique (0,2-1 bar, 3-15 PSI / 0,4-2 bar, 6-30 PSI) ;
- Tout ou rien ou proportionnelle.

▪Dans notre cas, l'actionneur utilisé est la vanne automatique de régulation, cette dernière est un organe comportant un orifice de dimension variable ainsi elle permet le réglage du débit du fluide.

▪Il faut noter que le débit est proportionnel à la racine carrée de la perte de charge entre l'amont et l'aval de la vanne.

V.11.4.1. Caractéristiques de débit

On appelle caractéristique d'une vanne régulatrice la relation qui lie la levée du clapet au débit qui traverse cette vanne à perte de charge constante. On rencontre trois types de caractéristiques liées à la géométrie des clapets qui sont :

- Caractéristique linéaire ;
- Caractéristique exponentielle ou égale pourcentage ;
- Caractéristique ouverture rapide.

Il faut noter que chaque vanne est caractérisée par son coefficient de débit C_v (donnée du constructeur).

V.11.4.2. Différents types de vannes

Les vannes automatiques peuvent se décomposer en deux grandes familles, suivant le type de déplacement utilisé, qui est soit longitudinale ou bien circulaire.

▪Pour les vannes a déplacement longitudinale, on trouve :

- Vanne à simple siège ;

- Vanne à double siège.

▪Pour ceux a déplacement circulaire, on trouve :

- Type CAMFLEX ;
- Type MAXFLOW.

▪Le choix de la technologie de la vanne a utilisé prend compte plusieurs critères : nature du fluide, raccordement de la vanne,….

V.11.5. Le régulateur

Le régulateur est considéré comme le cerveau de la boucle de régulation, ce dernier reçoit deux informations :

- Le signal de mesure provenant du capteur ;
- La consigne qui peut être local ou externe.

En fonction de l'écart entre ces deux valeurs ainsi que de l'algorithme de calcul pour lequel il a été configuré, il délivre un signal de sortie (commande) dirigé vers l'actionneur afin d'annuler cet écart et de ramener la mesure vers la valeur de consigne.

▪Dans notre cas, on a réalisé un régulateur PI (utilisé plus souvent pour ce type de régulation), donc on a besoin d'un bloc qui permet d'effectuer l'intégration numérique.

V.11.5.1. Intégration numérique

Pour notre cas, on a utilisé la méthode de Range Kutta 45 qu'on peut être expliqué comme suit : Considérons le problème suivant :

Pour $t \in [t_0,t_f]$ et le système qui s'écrit selon l'équation différentielle : $y'=f(t,y)$ avec $y(t_0)=y0$

▪Pour un pas de discrétisation h, l'algorithme du calcul numérique est donné par les équations :

$y_{n+1}=y_n+h/6*(K_1+2K_2+2K_3+K_4)$, avec :

$K_1=f(t_n,y_n)$

$K_2=f(t_n+h/2,y_n+K_1/2)$

$K_3=f(t_n+h/2,y_n+K_2/2)$

$K_4=f(t_n+h,y_n+K_3)$

▪Pour un système multi variable, par exemple 2 variables, le problème devient :

$\dot{y}=f_1(t,y,w)$

$\dot{w}=f_2(t,y,w)$

Pour un pas de discrétisation h, l'algorithme du calcul numérique est donné par les équations :

$y_{n+1}=y_n+h/6(k_1+2K_2+2k_3+K_4)$

$w_{n+1}=w_n+h/6(V_1+2V_2+2V_3+V_4)$

Ou :

$K_1=f_1(t_n,y_n)$ $V_1=f_2(t_n,y_n)$

$K_2=f_1(t_n+h/2,y_n+h*K_1/2,w_n+h*V_1/2)$ $V_2=f_2(t_n+h/2,y_n+h*K_1/2,w_n+h*V_1/2)$

$K_3=f_1(t_n+h/2,y_n+h*K_2/2,w_n+h*V_2/2)$ $V_3=f_2(t_n+h/2,y_n+h*K_2/2,w_n+h*V_2/2)$

$K_4=f_1(t_n+h,y_n+h*K_3,w_n+h*V_3)$ $V_4=f_2(t_n+h,y_n+h*K_3,w_n+h*V_3)$

▪Donc, on a crée un DFB qui permet la discrétisation des systèmes continues d'ordre 1, 2 et 3 qui s'écrivent de la manière suivante :

$\dot{y}+a_1*y+a_2=a_3*u$

$\ddot{y}+a_1*\dot{y}+a_2*y+a_3=a_4*u$

$\dddot{y}+a_1*\ddot{y}+a_2*\dot{y}+a_3*y+a_4=a_5*u$

▪Le programme du DFB intégration est en [**Annexe : D.3.7**].

▪Il faut noter qu'on a utilisé ce programme d'un coté pour discrétiser le système et aussi pour le calcul de l'action intégrale du régulateur.

V.11.5.2. Régulateur PI

L'algorithme de calcul qu'on a utilisé est celui d'un régulateur PI, qui permet la transmission instantané du signal d'erreur à l'aide de l'action proportionnelle et l'intégration du même signal ce qui permet d'annuler l'erreur en régime permanent.

▪Le programme du DFB régulateur est en [**Annexe : D.3.8**].

V.11.6. Bloc de mise à l'échelle

Puisque les automates reçoivent que des signaux normalisés alors il nous faut un bloc qui permet d'à partir du signal reçu reconstruire la vrai mesure du débit pour permettre la comparaison avec la cosigne, ce bloc va dépendre du transmetteur ainsi que du convertisseur utilisés, par exemple, si le transmetteur utilisé nous donne un signal de fréquence qui correspond au débit mesuré alors il nous faut un convertisseur de fréquence/intensité et selon la caractéristique de ce dernier (souvent linéaire), on programme ce bloc.

V.11.7. Exemple d'application

On vérifie les blocs programmés précédemment avec un exemple. On considère un système stable de premier ordre et on visualise les courbes en avec Vijeo Citect qui sont représenté dans la figure suivante :

Figure V.46 : Teste du régulateur programmé

▪On voit bien qu'on a une poursuite parfaite de la consigne (ceci dépend bien sur de la valeur du gain proportionnelle ainsi que de la valeur de la constante d'intégrale Ti).

V.12. Conclusion

Comme conclusion, on a crée les différents DFB nécessaires pour la gestion des différentes sections qui se trouvent au sein des stations de pompage et ça en respectant les cahiers de charge réels.

Conclusion générale

Afin de mieux aborder les stations de pompages, nos premières étapes consistaient en :

- La présentation de la gamme d'automates télémécanique, Où nous avons pris connaissance de leur architecture, leur fonctionnement, leur environnement avec les différents modules et fonctions ;
- Le logiciel UNITY PRO pour la commande et l'automatisation du système, et pour mieux maitriser le logiciel on a programmé quelques exemples là où on a illustré la création de quelques DFB et la programmation par les cinq langages d'UNITY PRO puis l'utilisation du simulateur d'automate ;
- Le logiciel VIJEO CITECT de supervision. Notamment la création de génies et de supers génies et enfin mettre en communication les interfaces de VIJEO CITECT avec le simulateur d'UNITY PRO ;
- On a aussi approfondie nos connaissances sur les réseaux notamment les plus utilisés comme Ethernet / TCP/IP/modbus, le modbus plus, le profibus, A-si qui est un réseau de terrain.

Et après avoir eu des idées concrètes sur l'oléoduc OZ1 parmi lesquelles sa longueur, son diamètre, les six stations de pompages intermédiaires, le terminal de départ et le terminal d'arrivée puis aux différents phases d'exploitation de OZ1 qui consistent en la réception du produit puis son stockage et enfin l'expédition du pétrole. On s'est intéressé plus tard au principe de fonctionnement d'une station de pompage intermédiaire et à ces différentes sections parmi lesquelles : station entrée, filtration, parc de stockage, pomperie boostage, pomperie principale, l'expédition. Et ce pour cerner le process et mieux l'assimiler et plus tard définir nos objectifs et un organigramme de travail.

Comme notre objectif était dés le départ de définir une interface de supervision pour les opérateurs opérant à partir du terminal ARZEW, et pouvant agir à distance sur la station à distance donc on devait définir un programme sur UNITY PRO permettant de répondre aux exigences de l'opérateur. Donc on a définie des DFB pour des vannes et des pompes et reprendre les programmes existant localement sur la station mais avec des automates d'autres sociétés donc avec d'autres logiciels. Enfin on a définie une interface de supervision conviviale et simple à assimiler pour l'opérateur, là où il pourra avoir à chaque instant un

aperçu sur la station et êtres au courant des différents défauts de la station, des alarmes aussi. Et il pourra aussi agir sur les différents groupes électropompes de la station et choisir les différents régimes de fonctionnement de la station de pompage.

Notre contribution donc s'est portée sur la réalisation des DFB permettent de structurer et d'optimiser le programme de gestion de la station de pompage ces DFB sont présenté d'une façon simple et compréhensible .Les résultats et tests obtenus répondent bien aux cahiers des charges. Mais aussi de proposer une interface de supervision complète traitant tout les états possibles de la station.

Perspectives

A la lumière des résultats obtenus pour ce projet de fin d'étude, de nombreuses perspectives s'ouvrent :

- ✓ Commande et supervision des 4 stations de pompages restantes et celle de terminal de départ pour une meilleure maitrise du process au niveau du terminal arrivée ;
- ✓ établir un algorithme de détection de fuite au niveau du pipeline ;
- ✓ Réécrire les DFB avec le langage de programmation C afin de réduire leurs volumes et accélérer leurs exécution, il s'agit de créer des blocs EFB et les intégrer carrément dans les nouvelles versions de l'Unity Pro.

Bibliographie

[1] Unity Pro 3.1 Modes de marche. Cd rom de documentation technique Telemecanique Unity Pro V4.0

[2] Unity Pro Langages de programmation et structure Manuel de référence. CD ROM de documentation technique Telemecanique Unity Pro V4.0

[3] Unity Pro 3.1 Gestion des E/S Bibliothèque de blocs. CD ROM de documentation technique Telemecanique Unity Pro V4.0

[4] Dali Ali et Fihakhir Amine Mahdi **«Gestion Et Supervision D'une Station De Pompage à Base D'automate SCHNEIDER à L'aide Des DFB»**, PFE à l'Ecole Nationale Polytechnique

[5] Feknous Baya et Hami Elias **«Développement de blocs fonction sous UNITY PRO. Application aux régulateurs P.I.D. auto-ajustables»**, PFE à l'Ecole Nationale Polytechnique

[6] Mohamed amine Almani et Rabah Boukra **« ETUDES ET APPLICATIONS A LA MAQUETTE PEDAGOGIQUE « CONVOYEUR AS-I » DE SCHNEIDER ELECTRIC»**, PFE à l'Ecole Nationale Polytechnique

[7] Document internet **« www.citect.com »**

[8] Document Schneider Electric **« Vijeo Citect Configuration Training Manuel »**

[9] Documents **SONATRACH**

[10] Document internet **«www.educnet.education.fr/rnchimie/gen_chim/regulation/rhode/reg_ana.pdf »**

REPUPLIQUE ALGERIENNE DEMOCRATIQUE ET POLPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
ECOLE NATIONALE SUPERIEURE POLYTECHNIQUE

Département d'Automatique

Annexe

Contribution à la commande et à la supervision des stations de pompage d'un oléoduc

Etudié par :
Mr. MENASRI Riad
Mr. ABBAS Mohand Said

Proposé et dirigé par :
Mr. A. BERKOUK
Mr.M.S.MAARADJI

Juin 2010

Ecole Nationale Polytechnique, 10 Av. Hassan Badi, El Harrach, Alger, Algérie

Annexe A

Gamme Automate Schneider

A.1. Gammes des automates Schneider Electric

Les automates mis sur le marché sont :

- Modules programmables Zelio Logic ;
- Contrôleurs programmables Twido ;
- Automate Modicon TSX Micro ;
- Automates Modicon M340 ;
- Modicon Premium ;
- Modicon Quantum.

A.1.1. Modules programmables Zelio Logic

Les modules logiques Zelio Logic sont destinés à la réalisation de petits équipements d'automatismes. Ils sont utilisés dans les secteurs d'activité de l'industrie et du tertiaire :

- Pour l'industrie :
  - automatismes de petites machines de finition, de confection, d'assemblage ou d'emballage ;
  - automatismes décentralisés sur les annexes de grosses et moyennes machines dans les domaines du textile, du plastique, de la transformation de matériaux ;
  - automatismes pour machines agricoles (irrigation, pompage, serre...).
- Pour le tertiaire/bâtiment :
  - automatismes de barrières, de volets roulants, de contrôle d'accès ;
  - automatismes d'éclairage ;
  - automatismes de compresseurs et de climatisation.

Figure A.1 : Module Zelio Logic

▪Il existe deux catégories de ces modules :

- Modules logiques compactes : les modules logiques compacts répondent aux besoins d'automatismes simples.
  Les entrées/sorties sont au nombre de :
  - 12 ou 20 E/S, alimentées en 24V AC ou 12V DC.
  - 10, 12 ou 20 E/S, alimentées en 100…240V AC ou 24V DC.
- Modules logiques modulaires et extensions : les entrées/sorties pour les modules logiques modulaires sont au nombre de :
  - 26 E/S, alimentées en 12V DC.

- 10 ou 26 E/S, alimentées en 24V AC, 100…240V AC ou 24V DC.

▪Pour plus de performance et de flexibilité, les modules Zelio Logic modulaires peuvent recevoir des extensions afin d'obtenir un maximum de 40 E/S :

- ✓ Extensions de communication réseau Modbus ou Ethernet, alimentées en 24V DC par le module Zelio Logic de même tension ;
- ✓ Extension d'entrées/sorties analogiques avec 4 E/S, alimentées en 24V DC par le module Zelio Logic de même tension ;
- ✓ Extensions d'entrées/sorties TOR avec 6, 10, ou 14 E/S, alimentées par le module Zelio Logic de même.

A.1.2. Contrôleurs programmables Twido

A.1.2.1. Bases compactes

La gamme des contrôleurs programmables compacts Twido offre une solution "tout en- un" dans un encombrement réduit de : 80 à 157 x 90 x 70 mm. Huit contrôleurs compacts sont disponibles, différents par leur capacité de traitement et leur nombre d'entrées 24V DC, de sorties à relais et à transistor (10, 16, 24 et 40 entrées/sorties).

Figure A.2: TWD LDA 10 DRF

Base compacte	Entrées 24 V DC	Sorties relais	Réglage Analogiques	Ports série	Expansion d'entrées/sorties	Module afficheur	Cartouche optionnelle
TWD LC•A 10DRF	6	4	1 point de 0…1023	1×RS 485	NON	OUI	1 emplacement horodateur ou mémoire
TWD LC•A 16DRF	9	7	1 point de 0…1023	1×RS 485 en option 1×RS 232C /485	NON	OUI	1 emplacement horodateur ou mémoire
TWD LC•A 24DRF	14	10	1 point de 0…1023 1 point de 0…511	1×RS 485 en option 1×RS 232C /485	OUI, 4 MAXI	OUI	1 emplacement horodateur ou mémoire
TWD LC•A 40DRF	24	14+2 sorties à transistor source	1 point de 0…1023 1 point de 0…511	1×RS 485 en option 1×RS 232C /485	OUI, 4 MAXI	OUI	1 emplacement horodateur ou mémoire

Tableau A.1 : Caractéristiques des différents modules Twido compact

A.1.2.2. Bases Modulaires

L'offre des contrôleurs programmables modulaires propose cinq bases, différentes par leur capacité de traitement et leurs nombre et type d'entrées/sorties (20 ou 40 E/S à raccordement par borniers à vis ou connecteur type HE 10, à sorties relais ou à transistor sink/source). Elles peuvent recevoir en expansion tous les modules d'entrées/sorties (18 modules TOR et analogiques). Toutes les bases modulaires utilisent une alimentation 24V DC. Ces bases modulaires offrent une modularité s'adaptant aux besoins de l'application à partir de base pouvant recevoir jusqu'à 4 ou 7 modules d'expansion d'entrées/sorties TOR ou analogiques (selon modèle).

Base modulaire	Entrée 24v DC	Sorties	Type de raccordement	Ports série	Expansion d'entrées /sorties	Extension module interface	Cartouche optionnelle
TWD LMDA 20DTK	12	8 à transistor source	Connecteur type HE 10	1×RS 485 en option 1×RS 232C /485	4 modules	1 module : afficheur ou liaison série	2 emplacement horodateur ou mémoire
TWD LMDA 20DUK	12	8 à transistor sink	Connecteur type HE 10	1×RS 485 en option 1×RS 232C /485	4 modules	1 module : afficheur ou liaison série	2 emplacement horodateur ou mémoire
TWD LMDA 20DRT	24	16 à transistor source	Connecteur type HE 10	1×RS 485 en option 1×RS 232C /485	7 modules	1 module : afficheur ou liaison série	2 emplacement horodateur ou mémoire
TWD LMDA 40DTK	24	16 à transistor source	Connecteur type HE 10	1×RS 485 en option 1×RS 232C /485	7 modules	1 module : afficheur ou liaison série	2 emplacement horodateur ou mémoire

Tableau A.2: Caractéristiques des différentes bases modulaires Twido

A.1.3. Automate Modicon TSX Micro

A.1.3.1. Automates TSX 37 05 (et TSX 37 08)

L'automate TSX 37 05 (TSX 37 08) comprend un bac intégrant une alimentation à 100/240 V, un processeur incluant une mémoire RAM de 11 K mots (programme, données et constantes), 1 mémoire de sauvegarde Flash EPROM, un module d'entrées/sorties "Tout ou Rien" TSX DMZ 28DR (16 entrées et 12 sorties à relais) et un emplacement disponible. L'emplacement disponible peut recevoir :

- 1 module d'entrées/sorties TOR au format standard de tout type ;
- 2 modules demi format de type entrées/sorties TOR, sécurité, entrées/sorties analogiques et comptage.

Figure A.3: Automate TSX 35 05

A.1.3.2. Automates TSX 37 10

Les automates TSX 37 10 compacts et modulaires se différencient par leur tension d'alimentation et le type de module d'entrées/sorties “Tout ou Rien” implanté de base dans le premier emplacement.

Chaque configuration TSX 37 10 comprend un bac intégrant une alimentation (24V DC ou 100/240V AC), un processeur incluant une mémoire RAM de 14 K mots (programme, données et constantes), une mémoire de sauvegarde Flash EPROM, un horodateur, un module d'entrées/sorties “Tout ou Rien” (28 ou 64 entrées/sorties) et un emplacement disponible. Un mini bac d'extension TSX RKZ 02 permet d'augmenter le nombre d'emplacements de 2 (4 positions).

Chaque emplacement disponible peut recevoir :

- Un module d’entrées/sorties TOR au format standard de tout type ;
- Deux modules demi-formats de type entrées/sorties TOR, sécurité, entrées/sorties analogiques et comptage.

De plus, les automates TSX 37 10 peuvent se connecter au réseau Ethernet TCP/IP ou à un Modem via le coupleur autonome externe TSX ETZ 410/510.

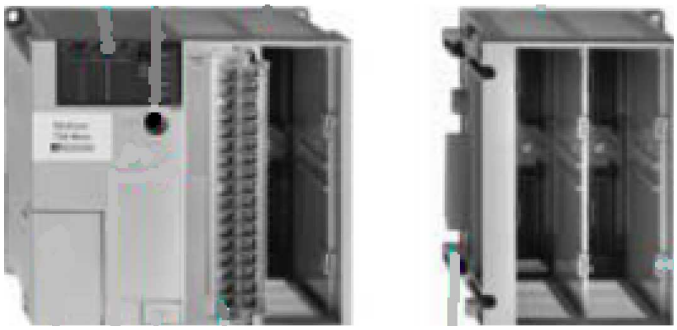


Figure A.4: Automate TSX 37 10

A.1.3.3. Automates TSX 37 21/22

Les automates TSX 37 21/22 modulaires se différencient entre eux par leur tension d'alimentation et/ou la possibilité d'effectuer sur la base, du comptage rapide et des fonctions analogiques.

Chaque automate comprend : un bac à 3 emplacements libres intégrant une alimentation (24V DC ou 100/240V AC), un processeur incluant une mémoire RAM de 20 K mots (programme, données et constantes), une mémoire de sauvegarde Flash EPROM, un horodateur, 2 emplacements pour carte PCMCIA (1 carte communication et une carte

extension mémoire de 128 K mots maximum). Un mini bac d'extension TSX RKZ 02 permet d'augmenter le nombre d'emplacements de 2 (4 positions).

Chaque emplacement disponible peut recevoir :

- 1 module d'entrées/sorties TOR au format standard ;
- 2 modules demi-formats de type entrées/sorties TOR, sécurité, entrées/sorties analogiques et comptage.

De plus, les automates TSX 37 21/22 peuvent se connecter au réseau Ethernet TCP/IP ou à un Modem via le coupleur autonome externe TSX ETZ 410/510.

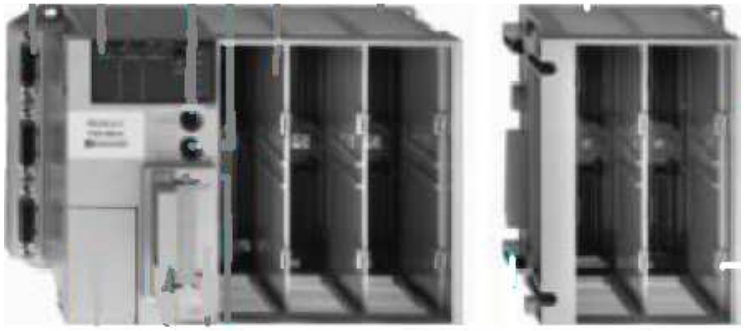


Figure A.5: Automate TSX 37 22

A.1.4. Automates Modicon M340

A.1.4.1. Modules processeurs

Les processeurs Standard et Performance de la plateforme d'automatisme Modicon M340 gèrent l'ensemble d'une station mono rack automate dont 11 emplacements maximum peuvent être équipés de :

- ✓ Modules d'entrées/sorties "Tout ou Rien" ;
- ✓ Modules d'entrées/sorties analogiques ;
- ✓ Modules métiers (comptage, communication Ethernet TCP/IP).

Figure A.6: Modicon M340

▪Quatre processeurs sont proposés, ils se différencient par leurs capacités mémoire, vitesses de traitement, nombre d'E/S et nombre et type de ports.

▪De plus, selon le modèle, ils proposent au maximum et d'une manière non cumulative :

- De 512 à 1024 entrées/sorties “Tout ou Rien” ;
- De 128 à 256 entrées/sorties analogiques ;
- De 20 à 36 voies métiers comptage ;
- De 0 à 2 réseaux Ethernet TCP/IP (avec ou sans port intégré et un module réseau).

▪Selon les modèles, les processeurs Modicon M340 intègrent :

- Un port Ethernet TCP/IP 10BASE-T/100BASE-TX ;
- Un bus machines et installations CANopen ;
- Une liaison série Modbus ;
- Une prise TER de type USB (pour connexion d’un terminal de programmation).

▪Chaque processeur est fourni avec une carte mémoire permettant :

- La sauvegarde de l’application (programme, symboles et constantes).

A.1.5. Modicon Premium

Les processeurs des plates-formes d'automatisme Premium TSX P57 ●●3M/3AM et 23M/23AM gèrent l'ensemble d'une station automate constituée de modules d'entrées/sorties “Tout ou Rien”, modules de sécurité Preventa, de modules d'entrées/sorties analogiques et de modules métiers qui peuvent être répartis sur un ou plusieurs racks connectés sur le bus X ou peuvent être distribués sur bus de terrain.

A.1.5.1. Les Processeurs TSX P57

Les processeurs proposés sont segmentés par des capacités différentes au niveau de la mémoire, des entrées/sorties “In rack”, des communications ainsi que par leurs vitesses de traitement. Selon le modèle :

- De 4 à 16 racks ;
- De 512 à 2040 entrées/sorties “Tout ou Rien” ;
- De 24 à 256 entrées/sorties analogiques ;
- De 8 à 64 voies métiers. Chaque module métier (comptage, commande de mouvement, liaison série ou pesage) compte pour 1 ou plusieurs voies métiers ;
- De 1 à 4 réseaux (Ethernet TCP/IP, Fipway, Ethway, Modbus Plus), de 2 à 8 bus capteurs/actionneurs AS-Interface, de 1 à 2 bus de terrain (CANopen, INTERBUS, Profibus DP), 0 ou 1 bus de terrain Fipio, des liaisons séries (Modbus, Uni-Telway) ;
- De 10 à 20 voies de régulation.

Type de processeurs			TSX P57 103M	TSX P57 153M	TSX P57 203M	TSX P57 2623M	TSX P57 253M	TSX P57 2823M
Configuration maximale	Nb de racks	4/6/8 Emplacements	4		16			
		12 emplacements	2		8			
	Nb d'emplacements maximal pour modules		32		128			
Fonctions	Nb maximal	E/S TOR	512		1024			
		E/S analogiques	24		80			
		Voies de régulation	-		10 (jusqu'à 30 boucles simples)			
		Voies métiers	8		24			
	Connexions intégrées	Ethernet TCP/IP	-			1	-	1
		Fipio gestionnaire	-	1 (63 agents)	-		1 (127 agents)	
		Liaison série	1 liaison avec 2 connecteurs (TER et AUX) 19.2 Kbit/s					
	Nb max de connexions	Réseaux	1		1	1, aucun si Ethernet intégré utilisé	1	1, aucun si Ethernet intégré utilisé
		Bus AS-i	2		4			
		Bus CANopen	1	-	1			
		Bus InterBus ou Profibus DP	-		1, aucun si CANopen utilisé			
Mémoires	Capacité maximale	Sans carte PCMCIA(Kmots)	32, programme et données		48, programme et données		64, programme et données	
		Avec carte PCMCIA(Kmots)	64, programme 32, données		160,programme 48, données		160, programme 64, données	
		Stockage de données (Kmots)	128		2688			
	Taille max des zones objets	Bits internes localisés (%Mi) (bits)	4096		8132			
		Données internes Localisées (Kmots)	30.5 pour mots internes %M•i 32 pour mots constants %K•i					

Tableau A.3: Caractéristiques spécifiques à chaque processeur TSX premium

A.1.6. Modicon Quantum

A.1.6.1. Présentation

Les unités centrales de la plate-forme d'automatisme Modicon Quantum sont basées sur des processeurs haute performance 486 et Pentium, et sont compatibles avec le logiciel Unity Pro.

De nombreuses fonctionnalités sont incluses de base dans les processeurs Quantum :

- Temps de cycle réduit avec acquisition rapide des entrées/sorties ;
- Traitement d'interruption sur événement de temps ou en provenance d'entrées ;
- Traitement possible en tâche rapide comme en tâche maître ;
- Extension des capacités mémoire par cartes mémoire PCMCIA ;
- Multiples ports de communication intégrés au processeur ;
- Diagnostic et maintenance aisés grâce au bloc de visualisation LCD en face avant des processeurs haut de gamme.

▪Les processeurs proposés se différencient par leurs capacités mémoire, leurs vitesses de traitement et leurs possibilités de communication.

Figure A.7: Automate Modicon Quantum

A.1.6.2. Mémoire sauvegardée protégée

Les processeurs supportent de base leur programme application en mémoire RAM interne sauvegardée par pile. Cette pile est logée en face avant du processeur et peut-être remplacée, processeur en fonctionnement.

Pour protéger le programme application en cas de mauvaise manipulation, les processeurs sont équipés en face avant d'un commutateur à clé destiné à protéger la mémoire. Ce commutateur peut être également utilisé pour autoriser la commande Run/Stop d'exécution du processeur. Le processeur 140 CPU 311 10 ne dispose que d'un commutateur pour la protection de la mémoire. Un bit de protection mémoire, à positionner en mode configuration, est également disponible pour le verrouillage de toute modification de programme (via PC de programmation ou par téléchargement de programme).

A.1.6.3. Ports de communication

Tous les processeurs s'intègrent dans les architectures de réseaux Modbus et Modbus Plus. Des commutateurs rotatifs en face arrière des modules permettent de définir l'adresse du (des) port(s) Modbus Plus. Chaque équipement réseau Modbus Plus doit avoir une adresse unique dans la plage 1...64. Les réglages des ports Modbus comprennent : la vitesse, la parité, le nombre de bits de données, le nombre de bits de stop, le protocole et l'adresse de l'Esclave. Par défaut, ces réglages sont 9600 bit/s, parité paire, 8 bits de données, 1 bit de stop, mode RTU et adresse 1.

Un commutateur en face avant des processeurs permet de paramétrer le port Modbus comme support de communication modem (2400 bit/s, parité paire, 7 bits de données, 1 bit de stop, mode ASCII et adresse 1).
Les processeurs 140 CPU 434 12 A et 140 CPU 534 14B disposent de 2 ports séries Modbus:

- Port 1 Modbus, paramétrable comme modem ;

- Port 2 Modbus, gestion de flux RIS/CTS (ne supporte pas la liaison modem).

A.2. Catalogues des modules d'alimentations

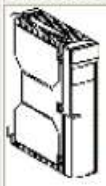
Références	TSX PSY 2600	TSX PSY 5500	TSX PSY 8500
			
Caractéristiques d'entrées			
Tensions nominales	100...240 VCA	100...120 VCA / 200...240 VCA	100...120 VCA / 200...240 VCA
Valeurs limites	85...264 VCA	85...140 VCA / 190...264 VCA	85...140 VCA / 190...264 VCA
Fréquence limite	47...63 Hz	47...63 Hz	47...63 Hz
Puissance apparente	50 VA	150 VA	150 VA
Courant nominal d'entrée	0,5 A à 100 V 0,3 A à 240 V	1,7 A à 100 V 0,5 A à 240 V	1,7 A à 100 V 0,5 A à 240 V
Caractéristiques de sorties			
Puissance totale	26 W	50 W	80 W
Tensions de sortie	5 V, 24 VR (1) 24 VC (2)	5 V, 24 VR (1) 24 VC (2)	5 V, 24 VC (2)
Courant nominal 5 V	5 A	7 A	15 A
Courant nominal 24 VR	0,6 A	0,8 A	non fourni
Courant nominal 24 VC	0,5 A	0,8 A	1,6 A

Tableau A.4 : Catalogue des alimentations pour réseaux à courant alternatif

Références	TSX PSY 1610	TSX PSY 3610	TSX PSY 5520
			
Caractéristiques d'entrées			
Tensions nominales	24 VCC non isolée	24 VCC non isolée	24...48 VCC isolée
Valeurs limites	19,2...30 VCC	19,2...30 VCC	19,2...60 VCC
Durée micro-coupures secteur acceptée	inférieure ou égale à 1 ms	inférieure ou égale à 1 ms	inférieure ou égale à 1 ms
Courant nominal d'entrée	≤ 1,5 A	≤ 2,7 A	≤ 3 A/24 V 1,5 A/48 V
Caractéristiques de sorties			
Puissance totale	26 W	50 W	80 W
Tensions de sortie	5 V, 24 VR (1)	5 V, 24 VR (1)	5 V, 24 VR (1)
Courant nominal 5 V	5 A	7 A	7 A
Courant nominal 24	0,6 A	0,8 A	0,8 A

Tableau A.5 : Catalogue des alimentations pour réseaux à courant continu

A.3. Catalogues des modules d'entrées/sortie analogiques

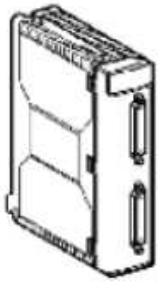
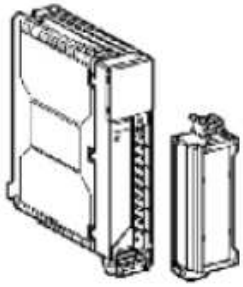
Type de module	Entrées				
					
Nombre de voies	16	8	4	16	4
Plage	+/- 10 V 0 à 10 V 0 à 5 V 1 à 5 V 0 à 20 mA 4 à 20 mA			+/- 80 mV Thermocouple	+/- 10 V 0 à 10 V +/- 5 V 0 à 5 V 1 à 5 V 0 à 20 mA 4 à 20 mA -13 à +63 mV 0 à 400 ohms 0 à 3 850 ohms Thermosonde Thermocouple

Tableau A.6 : Catalogue des modules d'entrées analogiques

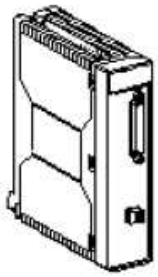
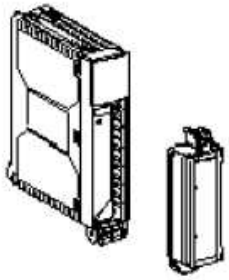
Type de module	Sorties analogiques	
		
Nombre de voies	8	4
Plage	+/- 10 V 0 à 20 mA 4 à 20 mA	
Resolution	14 bits en tension 13 bits en courant	11 bits signe +

Tableau A.7 : catalogue des modules de sorties analogiques

A.4. Catalogues des modules d'entrées/sorties TOR

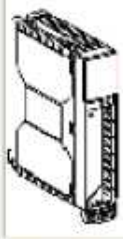
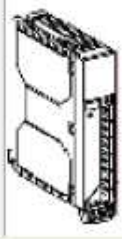
Type de module	Entrées avec raccordement par bornier à vis					
Illustration	Module d'entées TOR 		Module d'entrées TOR 			
Nombre de voies	8 entrées	16 entrées				
Gamme	24 VCC	48 VCC	24 VCA 24 VCC	48 VCA	100..120 VCA	200..240 VCA
Isolement	Entrées isolées					

Tableau A.8 : Catalogue des modules d'entrées TOR (à bornier a vis)

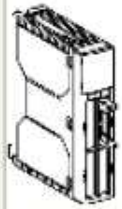
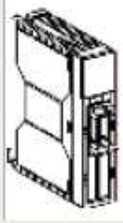
Type de module	Entrées avec raccordement par connecteur **HE10**			
Illustration	Module d'entrées TOR 	Mod. E TOR 	Mod. E TOR 	Mod. E TOR 
Nombre de voies	16 entrées rapides	32 entrées		64 entrées
Gamme	24 VCC		48 VCC	24 VCC
Isolement	Entrées isolées			

Tableau A.9 : Catalogue des modules d'entrées TOR (connecteur HE10)

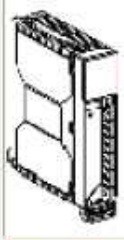
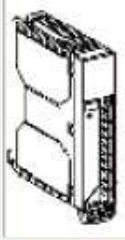
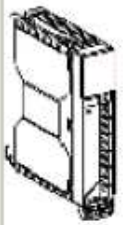
Type de module	Sorties à transistors avec raccordement par bornier à vis				
Illustration	Module de sorties TOR 	Module de sorties TOR 	Module de sorties TOR 	Module de sorties TOR 	Module de sorties TOR 
Nombre de voies	8 sorties			16 sorties	
Gamme	24 VCC		48 VCC	24 VCC	48 VCC
Isolement	Sorties isolées				
Courant	0,5 A	2 A	1 A	0,5 A	0,25 A

Tableau A.10 : Catalogue des modules de sortie à transistors TOR (bornier à vis)

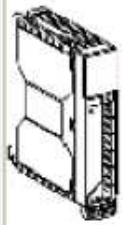
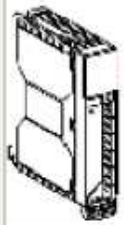
Type de module	Sorties à relais avec raccordement par bornier à vis			
Illustration	Module TOR 	Module de sorties TOR 		Module TOR 
Nombre de voies	8 sorties			16 sorties
Gamme	12..24 VCC ou 24..240 VCA	24..130 VCC	24..48 VCC ou 24..240 VCA	12..24 VCC ou 24..240 VCA
Isolement	Sorties isolées entre contact et terre			
Courant	3 A	5 A		3 A

Tableau A.11 : Catalogue des modules de sortie à relais TOR (bornier à vis)

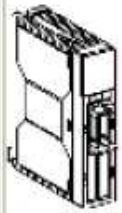
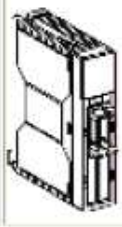
	Type de module	Sorties statiques avec raccordement par connecteur HE 10	
	Illustration	Module mixte E/S TOR 	Module mixte E/S TOR 
	Nombre de voies	16 entrées rapides 12 sorties	16 entrées rapides 16 sorties réflexes

Tableau A.12 : Catalogue d'entées sorties mixtes

Annexe B

Logiciel Unity Pro

B.1. Gestion globale d'un projet

Dans cette section, on va présenter les différents outils qui permettent : la création, sauvegarde, analyse, génération, sauvegarde d'un projet. Puis le transfert du projet dans l'automate, gestion de la mémoire utilisateur.

B.1.1.création d'un projet

Pour créer un nouveau projet, on effectue les actions suivantes :

- On sélectionne : Nouveau dans le menu Fichier, et on a l'écran Nouveau projet qui s'affiche comme le montre la figure suivante.

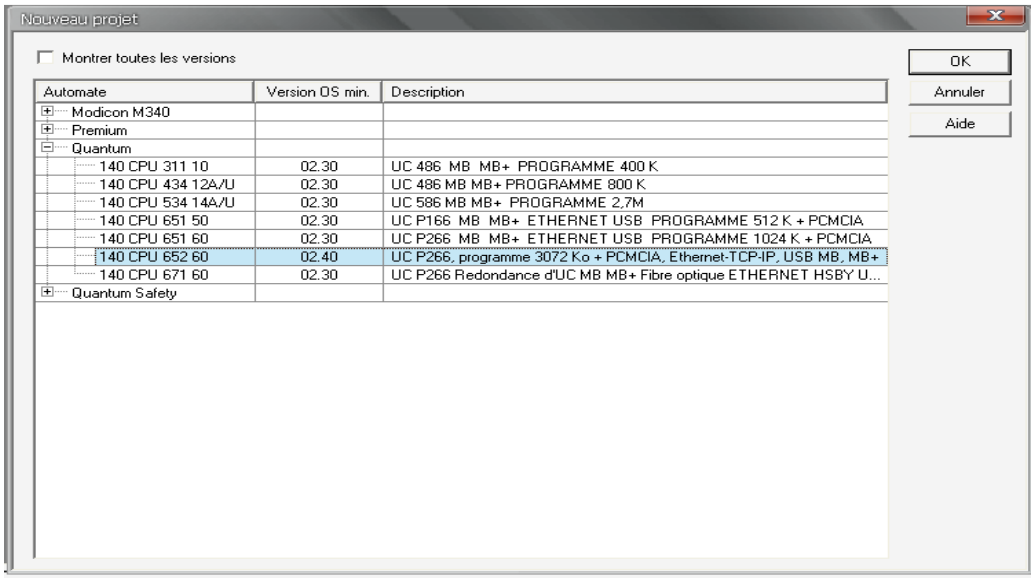


Figure B.1 : Ecran nouveau projet

▪ On choisi la station de travail et le processeur selon l'application a développée, et on clique sur OK pour valider le choix.

B.1.2. Convertir une application PL7

Pour convertir une application PL7 ou Concept vers une application sou Unity Pro, on procède comme suit :

- Dans le menu fichier, on active la commande ouvrir ;
- On sélectionne le fichier relatif à l'application :

-*.FEF ou *.DFB, pour lancer la procédure de conversion d'une application PL7.
-*.ASC, pour lancer la procédure de conversion d'une application Concept.

- On valide par le bouton ouvrir.

B.1.3. Archivage sauvegarde d'un projet

Pour archiver un projet au format STA, on procède comme suit :

- Dans le menu fichier, on utilise la commande archiver ;
- Si nécessaire, on peut choisir le répertoire dans lequel le projet sera stocké ;
- On saisie le nom du fichier ;

- On valide en cliquant sur enregistrer.

❖ Les caractéristiques du fichier STA sont les suivantes :

- Le fichier STA est très compressé (environ 50 fois plus que le fichier STU). Il permet de transférer des projets sur des réseaux (réseau local ou Internet, par exemple) ;
- le fichier STA peut être utilisé pour transférer des projets entre versions du logiciel Unity Pro (par exemple entre la version 1.1 et la version 1.2, pour la version 1.0 seul le fichier XEF permet de passer à la version 1.1) ;
- le fichier STA contient la totalité du projet :

-le binaire de l'automate,
-les informations d'Upload : commentaires et tables d'animations,
-les écrans d'exploitation.

B.1.4. Analyser un projet

Pour analyser la syntaxe de votre projet, on procède comme suit :

- On active la commande **Analyser le projet** du menu **Génération** ;
- Les éventuelles erreurs détectées sont affichées dans la fenêtre d'informations en bas de l'écran.

B.1.5. Analyser et générer un projet

Pour analyser et générer simultanément un projet (génération de liens entres les modules d'entrée/sortie et les objets déclarés dans le projet, etc.), on procéde comme suit :

- On active la commande **Regénérer tout le projet** du menu **Génération** ;
- Les éventuelles erreurs détectées sont affichées dans la fenêtre d'informations en bas de l'écran.

B.1.6. Sauvegarder un nouveau projet

Pour sauvegarder un nouveau projet, on effectue les actions suivantes :

- On active la commande **Enregistrer** ou **Enregistrer sous** du menu **Fichier** ;
- Si nécessaire, on choisi le répertoire dans lequel sera stocké le projet (disque et chemin) ;
- On introduit le nom du fichier et on enregistre.

B.2. Accès à un automate via un réseau

On procède comme suit pour accéder à un nouvel automate via le réseau :

- On sélectionne la commande **Automate → Définir l'adresse** afin d'afficher l'écran correspondant ;
- Dans le champ **Adresse**, on introduit l'adresse du nouvel automate ;
- Dans la liste déroulante **Support**, on sélectionne le type de communication qu'on souhaite utiliser ;
- On peut modifier :

  -les paramètres de communication à l'aide des boutons **Paramètres de communication** des champs **Automate** et **Simulateur**,

-les bandes passantes de connexion à l'aide du bouton **Bande passante**.

- On peut effectuer un test de connexion en cliquant sur le bouton **Test connexion** ;
- Un clic sur **OK** pour valider les paramètres.

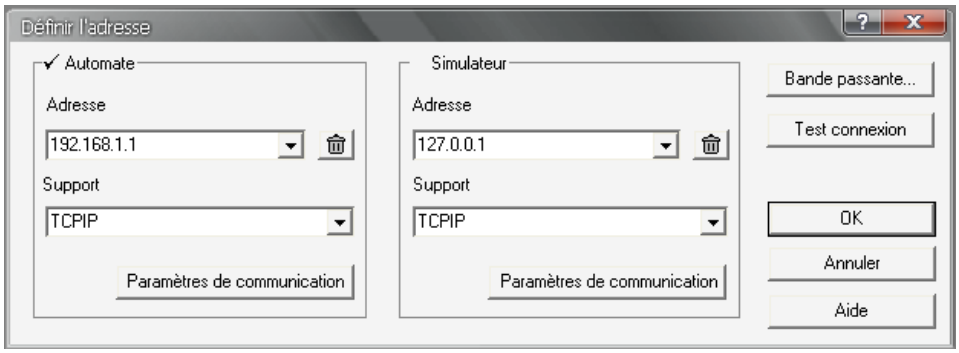


Figure B.2 : Ecran Définir l'adresse

B.2.1. Bande passante

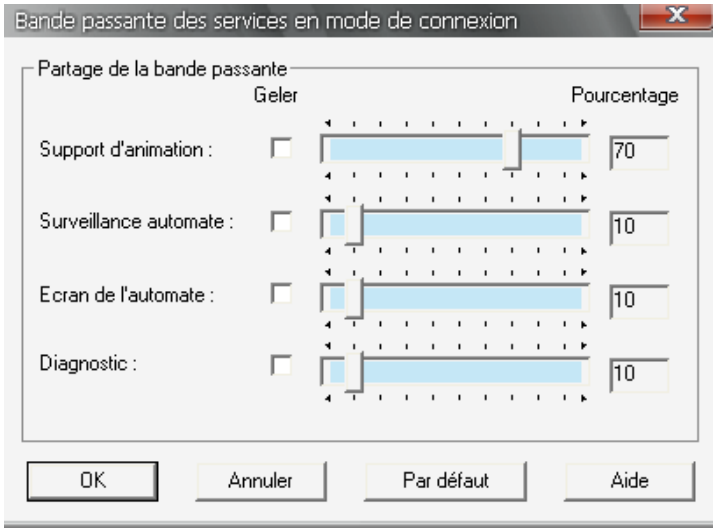


Figure B.3 : Ecran bande passante

▪Cet écran permet de définir le pourcentage de bande passante alloué à chacune des 4 catégories de fonctionnalités du mode connecté :

- Support d'animation : tables d'animation, écrans d'exploitation, animation des éditeurs langage, écrans métier. Plus le pourcentage est élevé, plus la fréquence de scrutation des données dans l'automate sera élevée ;
- Surveillance automate : surveillance du mode de marche global de l'automate (Run ou Stop), ou de ses tâches pour la fonction de "Mise au point de programme". Plus le pourcentage sera élevé, plus la fréquence de scrutation de l'état de l'automate et de ses tâches sera élevée ;
- Ecran automate (écran de mise au point UC) : plus le pourcentage est élevé, plus la fréquence de scrutation des informations sur l'automate et l'application est élevée ;
- Diagnostic (Viewer de diagnostic). Plus le pourcentage est élevé, plus la fréquence d'acquisition des alarmes dans l'automate sera élevée.

▪Cet écran permet ainsi d'optimiser les performances du mode connecté, en ajustant ces paramètres en fonction :

- des caractéristiques du projet chargé dans l'automate : alarmes de diagnostic nombreuses et fréquentes, nombreux écrans d'animation ouverts simultanément ;

-et de l'usage qu'il est fait du mode connecté : les fonctionnalités les plus fréquemment utilisées.

▪Une fois la communication établie et configurée on peut se mettre en mode connecté et là, on peut effectuer les taches suivantes :

- **Comparaison de projets :** La comparaison de projets nous permet de visualiser de manière synthétique, les différences éventuelles entre le projet embarqué dans l'automate et celui présente dans le terminal ;
- **Transfert du projet vers l'automate :** transfère le projet du terminal dans l'automate ;
- **Transfert du projet depuis l'automate** : transfère le projet de l'automate dans le terminal.

▪Pour se faire, on procède comme suit :

- On active la commande **Automate** puis **Comparer**, ce qui affiche l'écran suivant

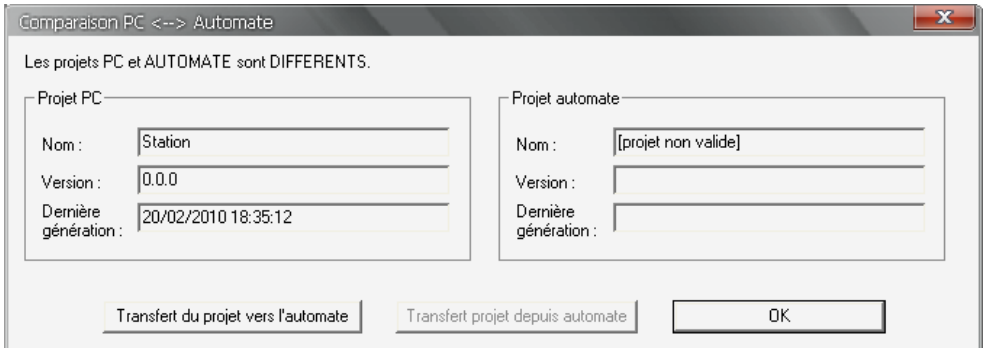


Figure B.4 : Ecran comparaison de projet

- On valide par **OK** ou autres.

B.2.2. Envoi d'une commande à l'automate

Les commandes Run/Stop et Initialiser permettent de commander depuis le terminal, le projet embarqué dans un automate cible :

- Run/Stop lance ou arrête l'exécution du projet ;
- Initialiser initialise le projet.

▪Ces commandes existent dans la commande **automate**

B.2.3. Bilan mémoire

La fonction de bilan mémoire permet de visualiser :

- la répartition physique de la mémoire de l'automate (mémoire interne et carte mémoire) ;
- l'occupation mémoire d'un projet (données, programme, configuration, système).

▪Cette fonction permet également de réorganiser la mémoire lorsque cela est possible.

▪On procède comme suit pour accéder aux détails du bilan mémoire de l'automate :

- On sélectionne **Automate → Utilisation de la mémoire**, afin d'afficher l'écran correspondant. Le bilan mémoire d'un projet n'est accessible que si on a généré au préalable son exécutable ;
- On peut réorganiser la mémoire pour en améliorer les performances et cela en activant la commande **Optimiser**.

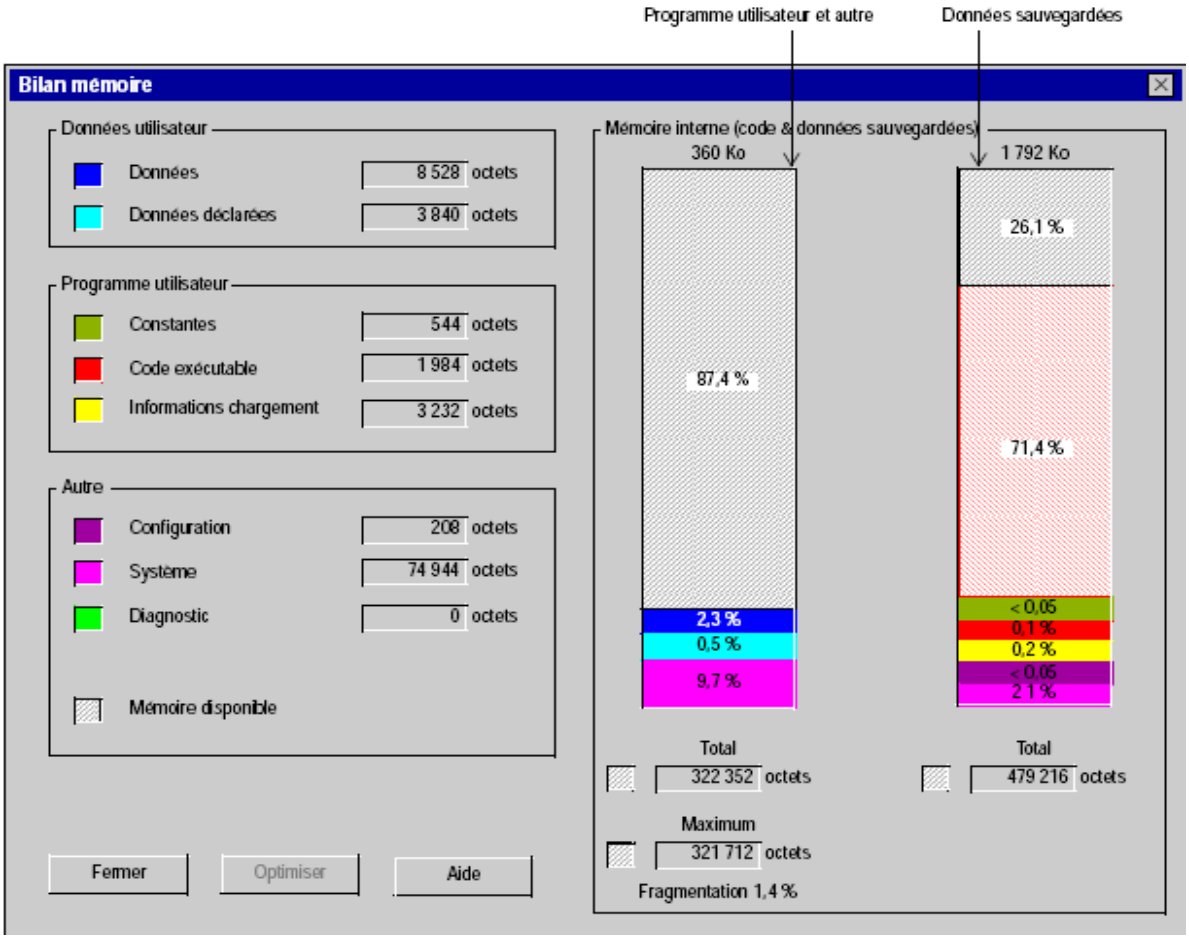


Figure B.5 : Ecran Bilan mémoire d'un Modicon M340

B.3. Protection d'un projet

La fonction protection du projet est accessible depuis l'écran **Propriétés** du projet en mode local. Cette fonction permet de protéger les sections programme.

La marche à suivre pour activer la protection des sections et la création du mot de passe est la suivante:

- On sélectionne dans le navigateur projet le répertoire **Station** ;
- On sélectionne dans le menu contextuel la commande **Propriétés** ;
- On sélectionne l'onglet **Protection** et la fenêtre suivante apparait :

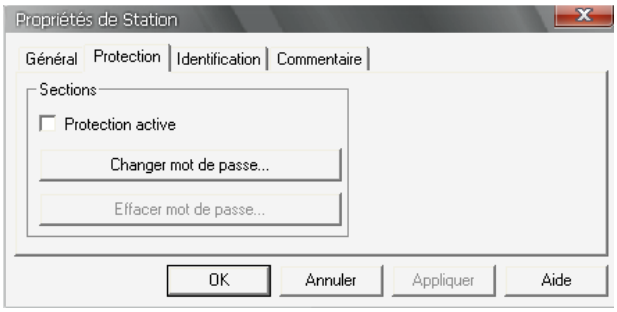


Figure B.6 : Ecran propriétés station

- On active la protection en cochant la case **Protection active**, la figure suivante apparait :

Figure B.7 : Ecran mot de passe

- On introduit un mot de passe et on valide.

B.4. Création des différents blocs

B.4.1. Création d'un type de structure

Pour la création d'un type de structure, on procède comme suit :

- Sur l'onglet **Types DDT** de l'éditeur de données, double-clique sur la cellule **Nom** vide (signalée par une flèche) et On saisi le nom du type de structure (par ex. IDENTITY), le type **<Struct>** est choisi par défaut ;
- On déploie la structure nouvellement créée en cliquant sur **+** ;
- Un double-clique sur le champ **Nom** correspondant, puis on introduit le nom du premier élément de la structure (par ex., Nom, Age) et son type.
- On clique deux fois sur la cellule suivante (signalée par une flèche) pour entrer le nom de l'élément suivant, etc.
- On se positionne sur le nom du type de la structure (IDENTITY), puis on sélectionne **Analyser type** dans le menu contextuel.

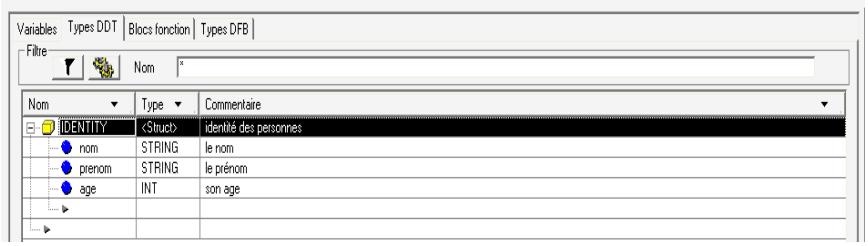


Figure B.8 : Exemple d'une structure

B.4.2. Création d'un type de tableau

- Sur l'onglet **Types DDT** de l'**éditeur de données**, on double-clique sur la cellule **Nom** (signalée par une flèche) et on saisit le nom du type de tableau (par ex. SERIAL) ;
- On double-clique dans la cellule **Type**, on sélectionne **<Tableau>** et on valide avec la touche **Entrée**.

▪Résultat : La fenêtre suivante s'affiche :

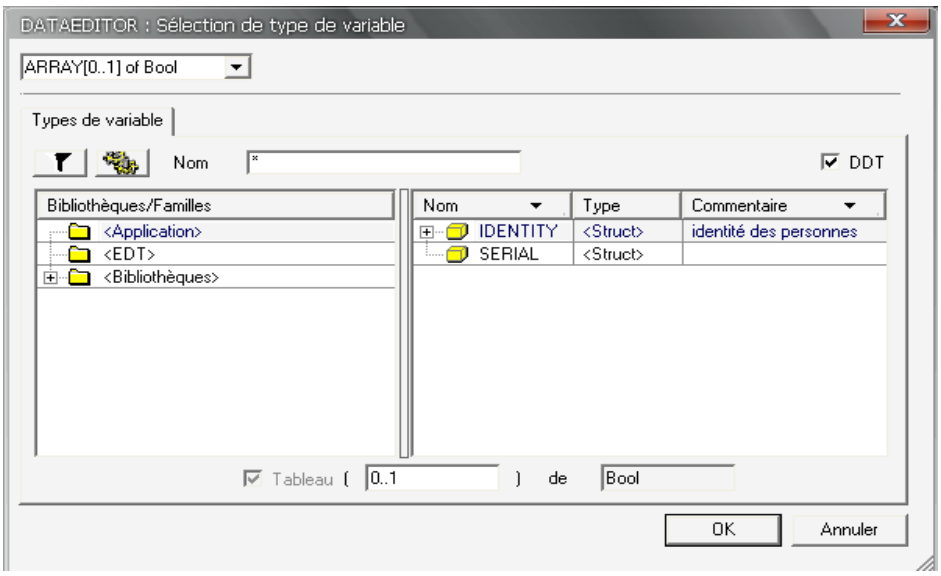


Figure B.9 : Tableau de sélection de type variable.

- La case **ARRAY** est cochée automatiquement :
  -On remplit le champ correspondant avec la dimension du tableau (par ex. 1..6).
  -Dans la zone **Bibliothèques/Familles**, on choisit les types à afficher :
  -le répertoire **<EDT>** pour un type élémentaire
  -le répertoire **<Application>** pour un type dérivé présent dans l'application.
  -le répertoire **<Bibliothèques>** pour un type dérivé archivé dans une bibliothèque.

- On se positionne sur le nom du type de tableau, puis on sélectionne **Analyser Type** dans le menu contextuel.

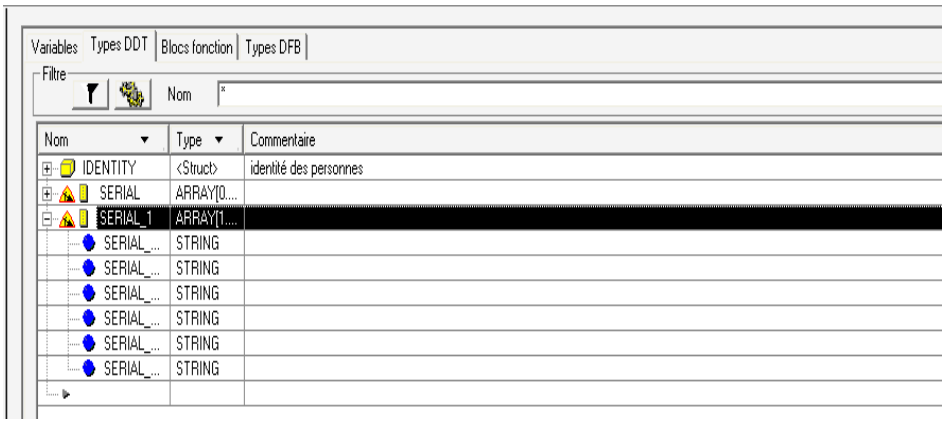


Figure B.10 : Création d'un type DDT

B.4.3. Création des instances de variables de type EDT

Pour les instances, on a deux cas et ça selon que les variables soient localisés ou pas.

B.4.3.1. La variable est non localisée

Pour les variables non localisé, on procède comme suit :

- On sélectionne l'onglet **Variables** dans l'éditeur de données ;
- On clique deux fois sur la cellule vide **Nom** (signalée par une flèche) et on saisit le nom de l'instance ;
- On double-clique sur la cellule **Type** correspondante et on choisit le type de l'instance ;
- On saisit éventuellement un commentaire.

B.4.3.2. La variable est localisée

Dans ce cas, la procédure est la suivante :

- On sélectionne l'onglet **Variables** dans l'éditeur de données ;
- On clique deux fois sur la cellule vide **Nom** (signalée par une flèche) et on saisit le nom de l'instance ;
- On double-clique sur la cellule **Type** correspondante et on choisit le type de l'instance ;
- On double-clique sur la cellule **Adresse** correspondante et on saisit une adresse mémoire automate ;
- On saisit éventuellement un commentaire.

B.4.5. Création d'un type d'instance de variable IODDT

Pour la création de ce type d'instance, on procède comme suit :

- On sélectionne l'onglet **Variables** dans l'éditeur de données ;
- On clique deux fois sur la cellule vide **Nom** (signalée par une flèche) et on saisit le nom de l'instance ;
- On clique deux fois sur le champ **Type** correspondant et on clique sur ;
- Dans la zone **Bibliothèque/Famille**, on affiche l'index du **<Catalogue>**. La fenêtre suivante s'affiche :

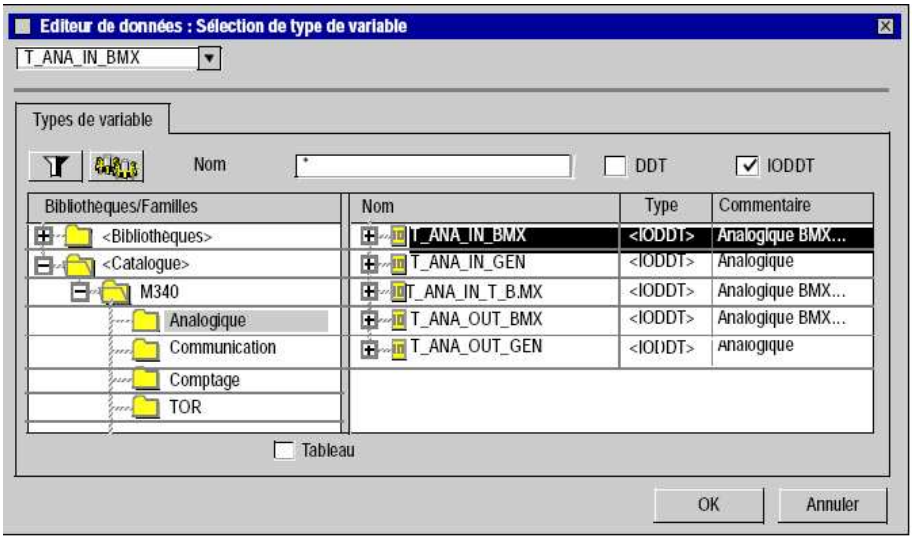


Figure B.11 : Type IODDT

- On sélectionne le type de contrôleur, la fonction et le type IODDT correspondant. Confirmez avec **OK**.

B.4.6. Création DFB

On procède comme suit pour créer la structure vide d'un type DFB :

- On ouvre l'onglet registre **types DFB** dans l'éditeur de données ;
  Résultat : La liste des DFB s'affiche ;
- On sélectionne la première cellule **Nom** vide (signalée par une flèche) et on saisit le nom du paramètre ou de la variable puis validez par **Entrée** ;
  Résultat : La structure vide du type DFB est créée. Le nouveau DFB est ajouté à la liste des DFB existants. Il apparaît également dans l'arborescence **Types FB dérivés**.

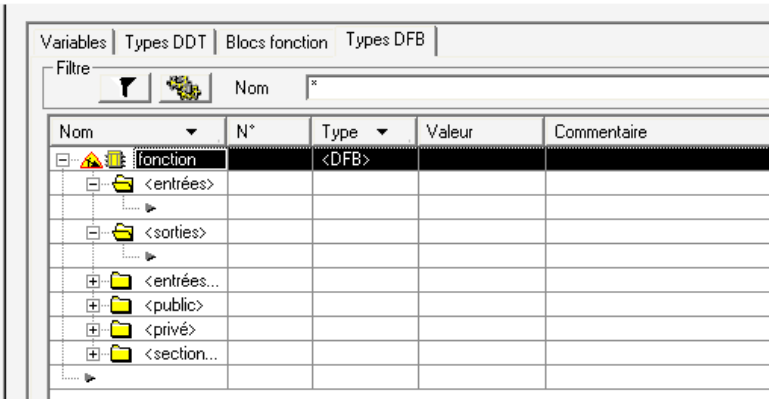


Figure B.12 : Types FB dérivés

- On clique sur le signe + du répertoire à développer : **entrée**, **sortie**, **entrée/sortie**, **public**, **privé ;**
- On sélectionne la première cellule **Nom** vide (signalée par une flèche) et on saisit le nom du paramètre ou de la variable, puis on confirme avec **Entrée** ;

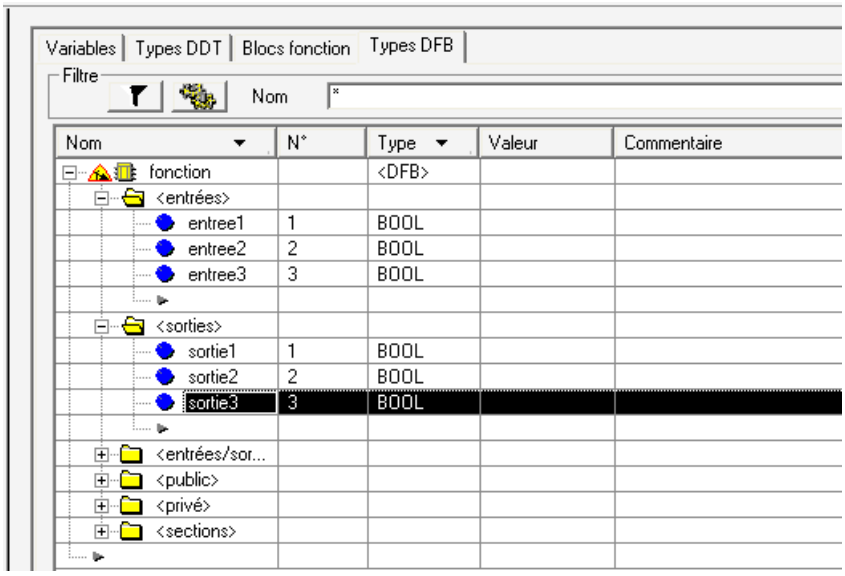


Figure B.13 : Types DFB

Remarque : on peut modifier le type des variables en double-cliquant sur le type.

▪Pour la programmation des DFB, dans l'angle **types FB dérivés** on déploie la fonction qu'on veut programmer et dans le menu contextuel on sélectionne **nouvelle section** et la page suivante s'ouvre :

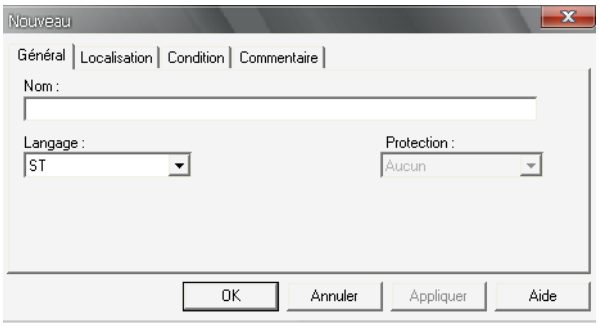


Figure B.14 : Programmation des DFB

▪Puis on choisit le langage de programmation souhaité dans le menu déroulant du champ Langage.

▪Pour l'archivage de la fonction on active la commande **Copier dans la bibliothèque.**

B.5. Configuration réseau

Sous Unity Pro, la mise en œuvre d'un réseau s'effectue à partir du navigateur d'application et à partir de l'éditeur de configuration matérielle.

▪La méthode nécessite les quatre étapes suivantes :

- Création d'un réseau logique ;
- Configuration du réseau logique ;
- Déclaration du module ou de la carte PCMCIA (pour Premium) ;
- Association de la carte ou du module au réseau logique.

B.5.1. Création d'un réseau logique

La première étape de la mise en œuvre d'un réseau de communication consiste à créer un réseau logique, pour se faire, on procède comme suit :

- à partir du navigateur de projet, on développe le répertoire Communication ;
- On clique avec le bouton droit sur le sous-répertoire Réseaux et on choisit l'option **Nouveau réseau**, la figure suivante apparait :

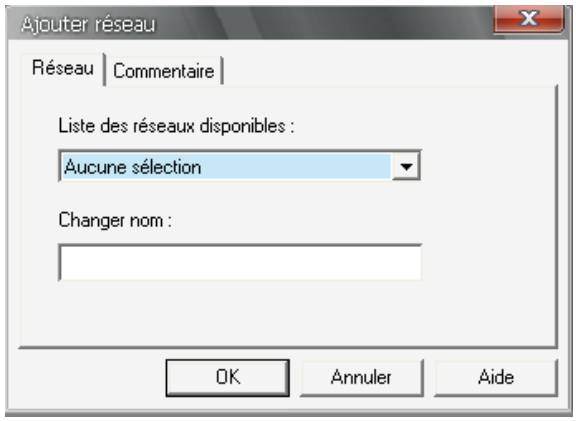


Figure B.15 : Ajouter réseau

- On sélectionne le réseau à créer dans la liste des réseaux disponibles et on lui donne un nom significatif ;
  - **Résultat :** Exemple d'un réseau Ethernet :
- Puis on clique sur OK, un nouveau réseau logique est créé.

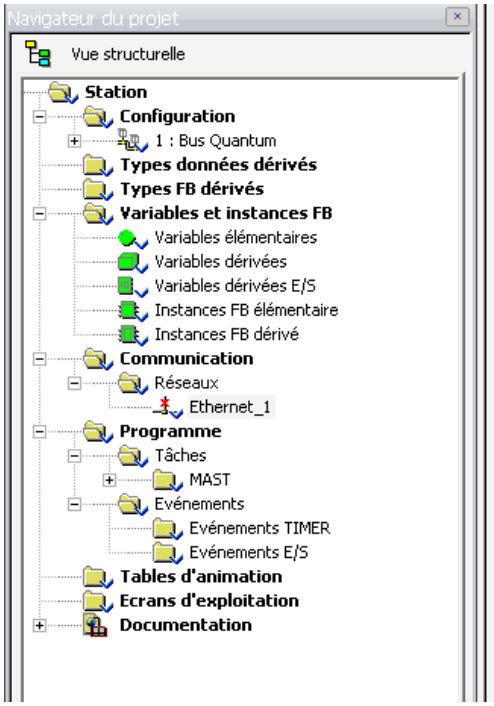


Figure B.16 : navigateur du projet

NB : Comme on le constater, une petite icône indique que le réseau logique n'est associé à aucun matériel de l'automate. D'autre part, le petit signe en "v" bleu indique que le projet nécessite une régénération pour pouvoir être utilisé dans l'automate.

B.5.2. Configuration d'un réseau logique

Pou la configuration du réseau logique crée précédemment, on procède comme suit :

- Dans le navigateur de projet, on développe l'arborescence située sous le sous-onglet **Réseaux** de l'onglet **Communication** afin de visualiser l'ensemble des réseaux du projet ;
- On clique deux fois sur le réseau à configurer pour obtenir la fenêtre de configuration du réseau.

Remarques : Les fenêtres diffèrent selon la famille de réseaux choisie. Toutefois, pour tous les réseaux, c'est à partir de cette fenêtre qu'on pourrait configurer les utilitaires Global Data, IO Scanning, Peer Cop, les mots communs, etc.

▪Pour les réseaux Ethernet, une étape intermédiaire est nécessaire. Il s'agit de choisir la famille du module qui sera utilisé dans la configuration matérielle.

▪Pour ce qui est d'Ethernet la fenêtre suivante apparait

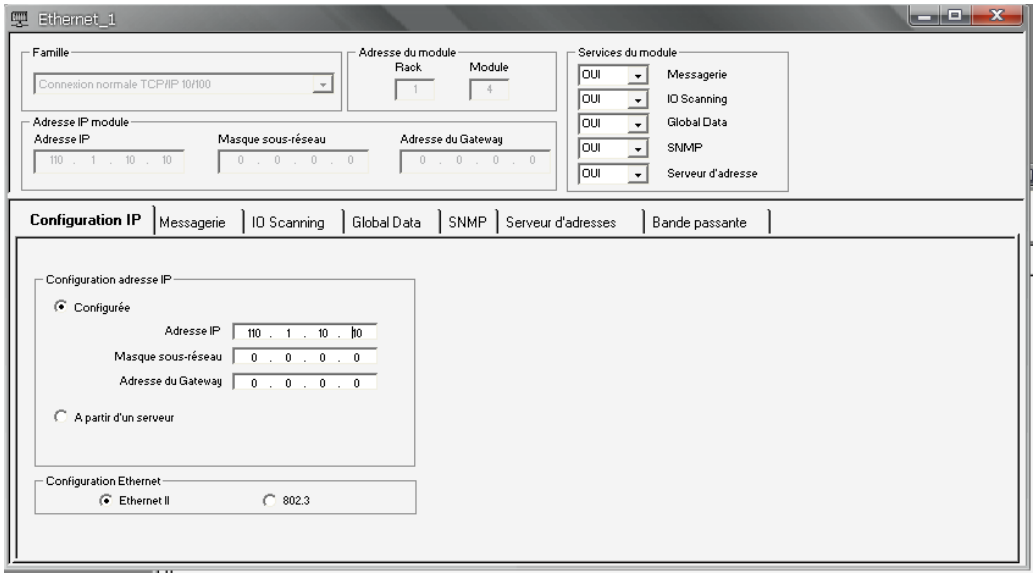


Figure B.17 : Configuration d'un réseau logique**.**

▪L'onglet **Configuration IP** permet de configurer les paramètres d'adresse IP du module Ethernet. Les paramètres de l'adresse IP deviennent actifs dans les cas suivants :

- une fois le matériel connecté.
- après le téléchargement de la configuration vers l'automate dans le module Ethernet.

Contenu de la fenêtre :

- ✓ **Configurée** : Active **l'adresse IP**, le **masque de sous-réseau** et **l'adresse de passerelle**. Les données sont activées après le chargement de la configuration dans l'automate ;

- ✓ **Configuration Ethernet** : permet de sélectionner le protocole par défaut en tant qu'Ethernet ou 802.3.

▪La **messagerie** Ethernet permet à l'utilisateur d'envoyer et de recevoir des messages Ethernet.
▪Le trafic de données est géré par la procédure client/serveur.

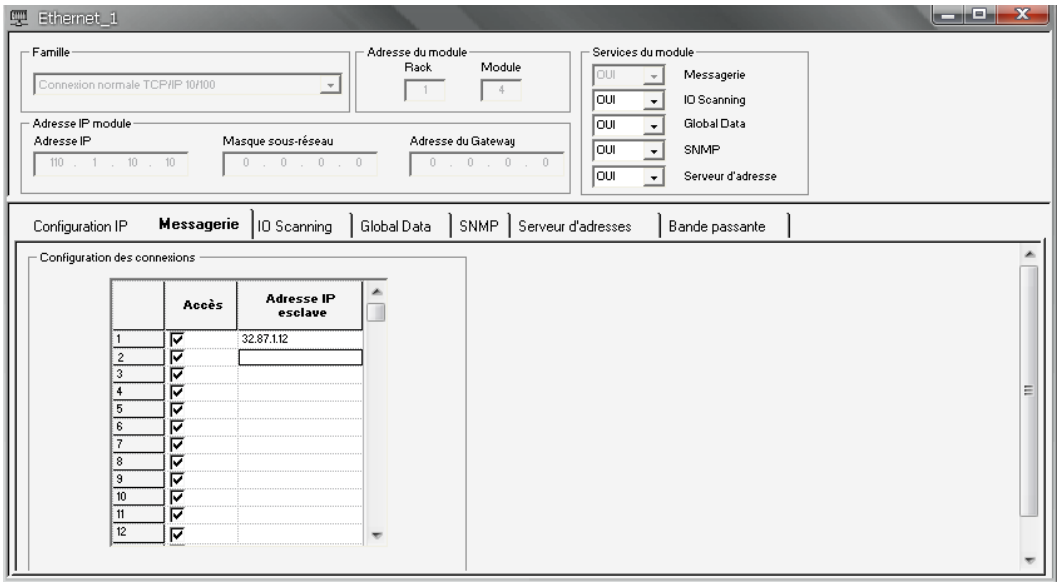


Figure B.18 : Messagerie Ethernet

▪**Description des paramètres :**

-Configuration des connexions : Active le transfert de données général ;
-Accès : Active le transfert de données entre des nœuds spécifiques ;
-Adresse IP esclave : Définit le nœud pour la procédure de messagerie Ethernet.

▪**L'I/O Scanner** est une fonction du **module NOE** qui permet la lecture et/ou l'écriture répétées sur des équipements d'entrée/de sortie, la liste de scrutation des E/S est une table de configuration identifiant les cibles avec lesquelles les communications à répétition sont autorisées. La liste contient suffisamment d'informations pour permettre à chaque cible de créer le message **MODBUS** adressé à l'équipement distant spécifié et de déterminer l'emplacement sur l'automate local sur lequel les données d'entrée et de sortie seront affectées à la fin de l'analyse. Lorsque l'automate fonctionne, le module NOE transmet les données vers et depuis les registres et les bobines de l'automate, tel qu'indiqué par la liste de scrutation des E/S.

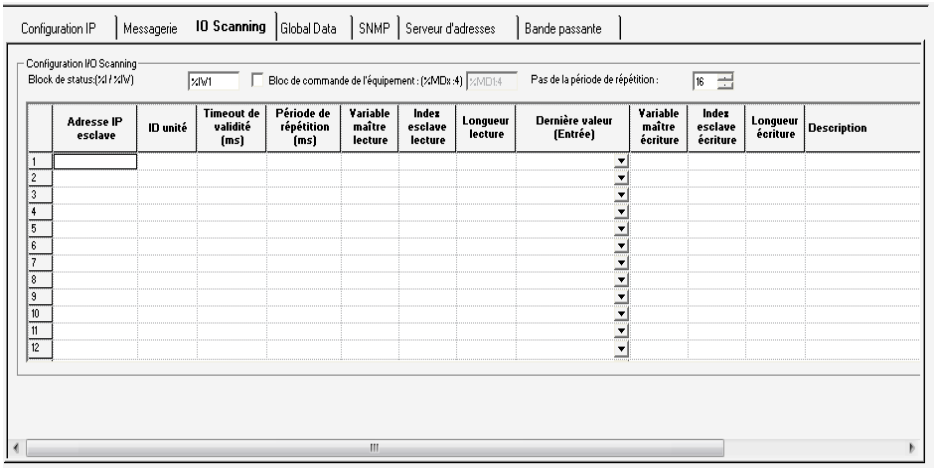


Figure B.19: I/O Scanner

▪Les différents paramètres :

-Adresse IP esclave : Adresse IP de l'équipement Ethernet scruté ;
-ID unité : ID spécifique de l'équipement sur le réseau Modbus/Modbus Plus.
L'équipement se connecte à Ethernet via un routeur ;
-Timeout de validité (ms) : Durée du timeout de 1 à 65535 ms ;
-Variable maître (lecture) : Adresse cible dans l'automate pour les opérations de lecture ;
-Index esclave (lecture) : Adresse source du module d'E/S pour les opérations de lecture ;
-Longueur (lecture) : Nombre de mots à lire ;
-Dernière valeur (entrée) : Etat des entrées en cas d'erreur ;
-Variable maître (écriture) : Adresse source de l'automate pour les opérations d'écriture. L'écriture est effectuée via des mots ;
-Variable esclave (écriture) : Adresse cible de l'esclave pour les opérations d'écriture ;
-Longueur (écriture) : Nombre de mots à écrire.

B.5.3. Déclaration du module ou de la carte PCMCIA (pour Premium)

Cette partie consiste en la déclaration des modules ou de la carte PCMCIA dans la configuration matérielle, et on procède comme suit :

- A partir du navigateur projet on développe le paramètre **configuration** et on double-clique sur **bus** ;

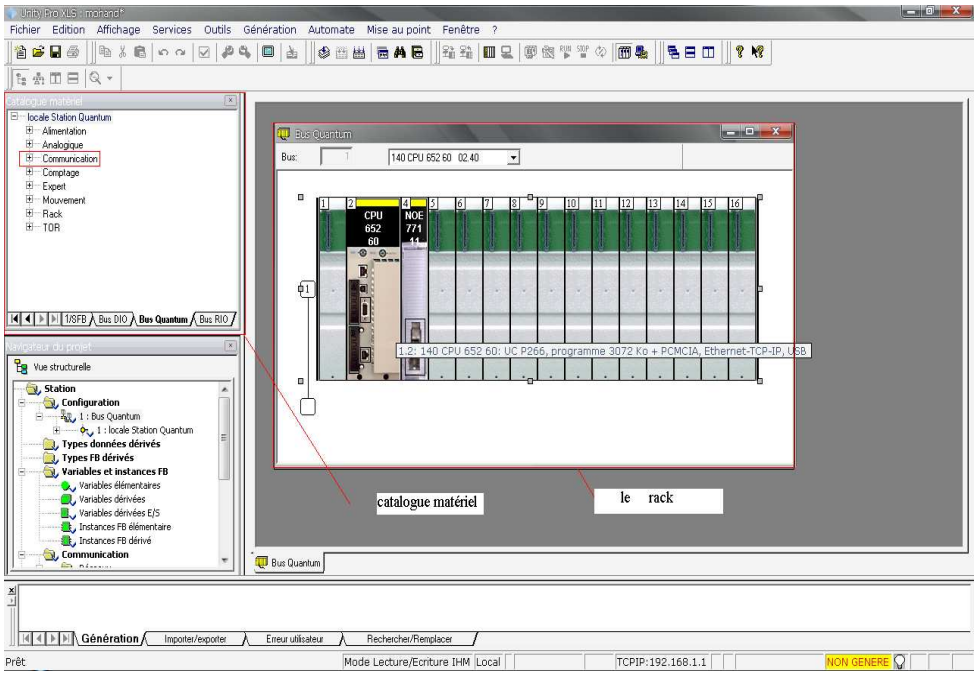


Figure B.20: Déclaration des modules ou de la carte PCMCIA

- Dans l'angle **communication** qui est dans la fenêtre **catalogue matériel** on choisi un module de communication qui nous arrange puis on le place dans les emplacements disponibles sur le rack.

B.5.4. Association d'un réseau logique via un matériel réseau

L'étape finale de mise en œuvre d'un réseau de communication consiste à associer un réseau logique à un module réseau, une carte Modbus Plus ou une carte Fipway. Bien que les écrans diffèrent, la procédure est la même pour chaque équipement réseau qui est :

- On ouvre l'éditeur de configuration matérielle ;
- On clique avec le bouton droit sur l'équipement (module Ethernet, carte PCMCIA Fipway ou carte PCMCIA Modbus Plus) à associer à un réseau logique ;
- On sélectionne la voie et la fonction ;
- Dans le champ **Ligne réseau**, on sélectionne le réseau à associer à la carte ;
- On confirme notre choix, puis on ferme la fenêtre.

▪**Résultat :** Le réseau logique est associé à l'équipement. L'icône associée à ce réseau logique change et indique l'existence d'une liaison avec un automate. En outre, les numéros de rack, de module et de voie sont actualisés dans l'écran de configuration du réseau logique.

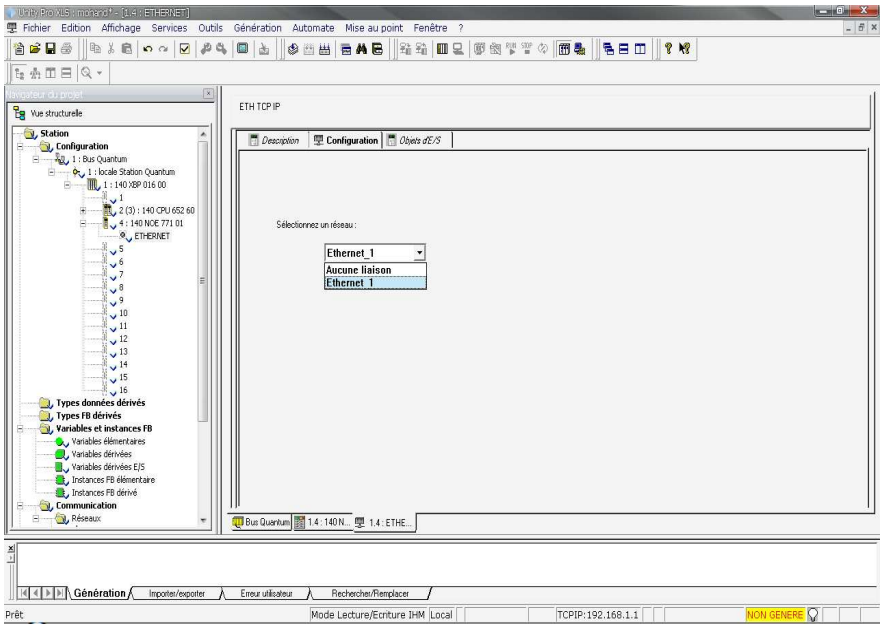


Figure B.21: Associer le réseau logique au module réseau

B.7. Création et configuration d'une tâche

On exécute les actions suivantes :

- Dans le navigateur projet on double clique sur le répertoire programme. Le répertoire MAST apparait sous le répertoire **Tâches**.
- On effectue un clic droit sur le répertoire Tâches, le menu contextuel permet alors d'accéder à la commande **Nouvelle tâche...**.
- On clique sur la commande **Nouvelle tâche...**, la boîte de dialogue suivante apparait :

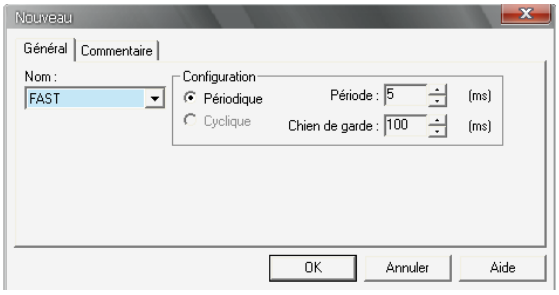


Figure B.22: Création d'une nouvelle tache

- On choisit la tâche :
  -FAST : tâche rapide
  -AUX 0, AUX1, AUX2 ou AUX3 : tâches auxiliaires (dans le cas des processeurs intégrant les tâches auxiliaires)
- On choisit le type de scrutation: Périodique ou cyclique (uniquement pour la tâche maître) ;
- On définit la période de la tâche ;
- On fixe la valeur du **Chien de garde**, la valeur du Chien de garde doit être supérieure à la valeur de la période.

B.8. Création d'une section FBD, LD, IL ou ST

Pour créer une section :

- On effectue un clic gauche sur le répertoire **Sections** de la tâche désirée ;
- On sélectionne dans le menu contextuel la commande **Nouvelle section**.

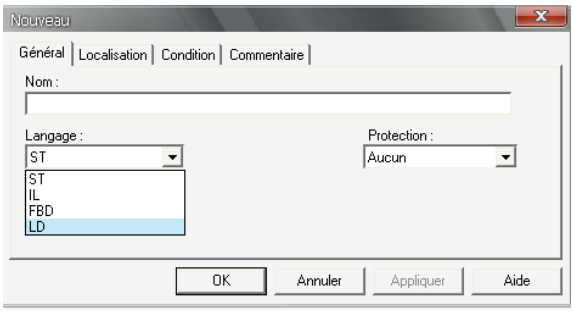


Figure B.23: sélection d'un langage FBD, LD, IL ou ST

- On saisit le nom de la section. Le nom de la section doit être unique dans tout le projet et conforme aux conventions de désignation générales ;
- On sélectionne le langage de programmation de la section.

▪Pour affecté des paramètres réels avec l'assistant de saisie de fonction, on effectue les opérations suivantes :

- On sélectionne le FFB ;
- On ouvre l'assistant de saisie de fonction via la commande de menu **Edition** → **Assistant de saisie FFB...**.

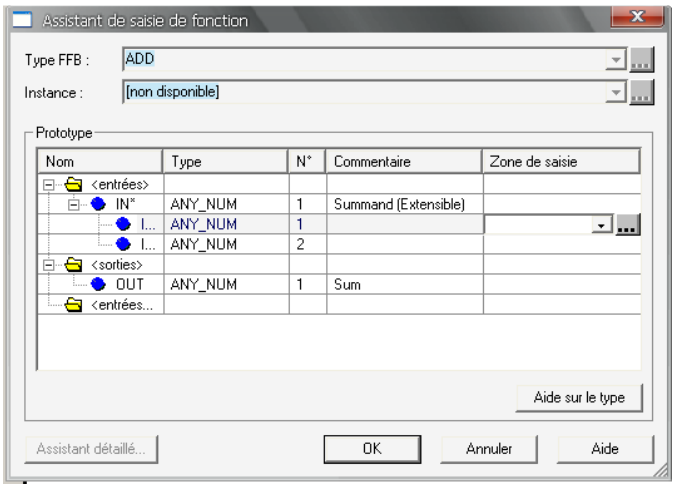


Figure B.24: Assistant de saisie de fonction

- On clique deux fois sur la cellule **Zone de saisie** du premier paramètre formel et on indique le paramètre réel à utiliser ;
- On confirme les entrées à l'aide du bouton de commande **OK** ;

❖ **Les EN/ENO**

Pour tous les FFB une entrée EN et une sortie ENO peuvent être projetées,

▪Pour les masquer ou démasquer, il suffit d'activer dans l'onglet **Propriétés FFB** la case à cocher **Afficher EN/ENO** pour démasquer les EN/ENO, ou on désactive cette case à cocher pour masquer les EN/ENO.

B.9. Configuration des modules d'entrée/sortie pour une station Premium Atrium, Quantum ou Modicon M340

B.9.1. Modicon, Premium ou Atrium

On procède comme suit :

- on sélectionne avec la souris le module à configurer ;
- on Sélectionne par le menu contextuel la commande **Ouvrir le module**.

▪Exemple d'écran pour un module TOR :

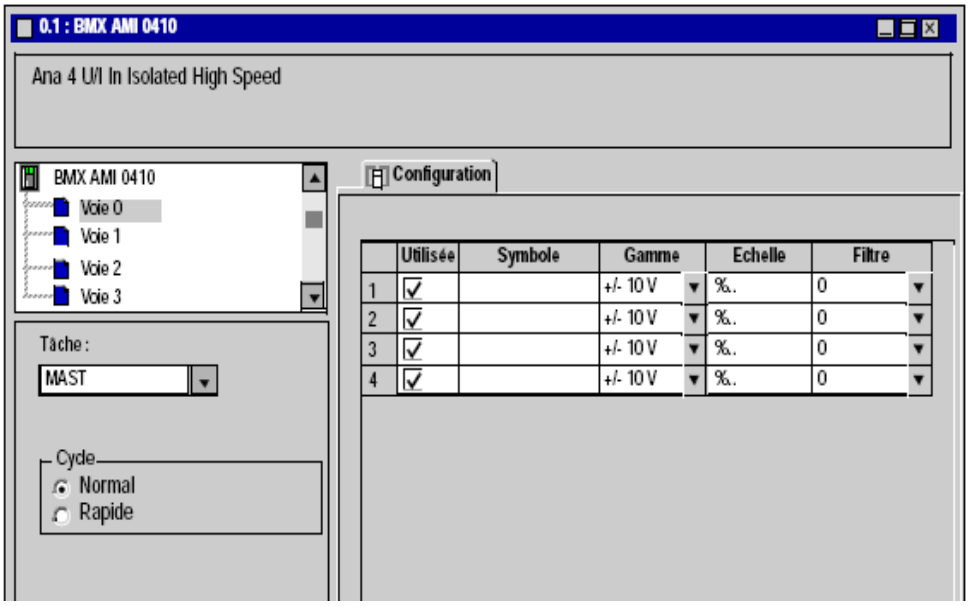


Figure B.25: Ecran pour un module TOR Modicon, Premium ou Atrium

B.9.2. Quantum

On procède comme suit :

- on sélectionne avec la souris le module à configurer ;
- On sélectionne par le menu contextuel la commande **Ouvrir le module**.

▪Exemple d'écran pour un module TOR

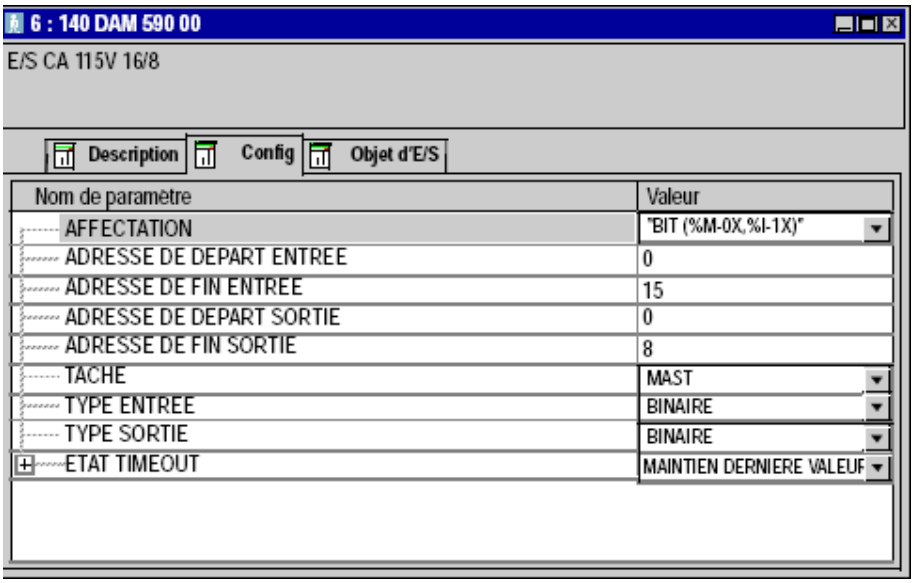


Figure B.26: Ecran pour un module TOR Quantum

B.10. Présentation de la fonction métier de communication

La fonction métier de communication permet d'échanger des données entre différents équipements connectés à un bus ou à un réseau.

- Cette fonction est disponible pour :
  - les processeurs disposant d'une liaison Ethernet, Modbus ou CANopen ;
  - des modules de communication spécifiques montés sur le rack ;
  - le port du terminal d'un processeur ;
  - les cartes PCMCIA d'un processeur ou d'un module monté sur le rack.
- Les différents types de communication sont :
  - Réseau TCP/IP ou Ethway Ethernet ;
  - Réseau Fipway ;
  - Réseau Modbus Plus ;
  - Bus Fipio (gestionnaire et agent) ;
  - Bus Uni-Telway;
  - Bus Modbus/JBus;
  - Liaison série en mode caractère ;
  - Bus de terrain CANopen ;
  - Bus de terrain Interbus ;
  - Bus de terrain Profibus ;
  - Port rapide du terminal de norme USB.

B.10.1. Les réseaux préconisés par Schneider Electric

Les réseaux les plus utilisés par Schneider Electric sont :

- Ethernet Modbus TCP/IP : La généralisation d'Ethernet dans les entreprises et sur Internet en a fait un standard de communication incontournable. Son usage généralisé a permis de réduire les coûts de connexion, d'augmenter les performances, la fiabilité et les fonctionnalités. Sa rapidité ne limite pas les applications, son architecture permet facilement les évolutions. Les produits et les logiciels demeurent compatibles, ce qui

assure la pérennité des systèmes. Le protocole “Modbus”, standard de fait dans l’industrie, fournit une couche application simple et peu onéreuse à mettre en œuvre.

- Can Open : Can Open est la version industrielle du bus CAN. Créé pour l’automobile, ce réseau à prouvé sa souplesse et sa fiabilité depuis plus de 10 ans dans de multiples applications telles que les équipements médicaux, les trains, les ascenseurs, ainsi que dans de multiples machines et installations. La forte diffusion de ce réseau a confirmé Schneider Electric dans son choix.
- As-Interface : Les machines modernes ont une grande quantité d’actionneurs et de capteurs et souvent des contraintes de sécurité. AS-Interface est le réseau niveau capteur conforme aux exigences des automatismes industriels. Il présente l’avantage d’offrir une connectique rapide et un seul câble pour véhiculer les informations et l’alimentation.

B.10.1.1. Ethernet TCP/IP

L’Ethernet est basé sur le principe d’accès au media régit par un mécanisme de détection de collision. Chaque station est identifiée par une clé globalement unique, appelée adresse MAC, afin de s’assurer que tous les postes sur un réseau Ethernet aient des adresses distinctes. Cette technologie connue sous le nom de Carrier Sense Multiple Access with Collision Detection (Détection de porteuse avec accès multiples et détection de collision) ou CSMA/CD garantit qu’une seule station à la fois transmet un message sur le media.
Les évolutions successives d’Ethernet ont donné naissance au standard IEEE 802.3.

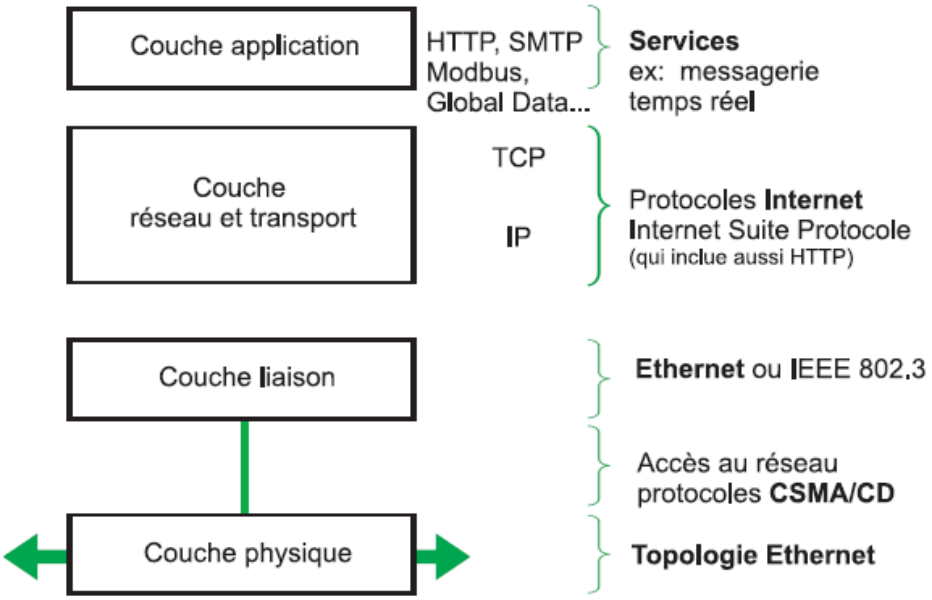


Figure B.27 : Les différentes couches

B.10.1.1.1. La couche physique

La couche physique décrit les caractéristiques physiques de la communication, comme les conventions à propos de la nature du médium utilisé pour les communications (les câbles, les liens par fibre optique ou par radio), et tous les détails associés comme les connecteurs, les types de codage ou de modulation, le niveau des signaux, les longueurs d’ondes, la synchronisation et les distances maximales.

B.10.1.1.2. La couche de liaison de données

La couche de liaison de données spécifie le contrôle d'accès au media et comment les paquets sont transportés sur la couche physique, et en particulier le tramage (les séquences de bits particulières qui marquent le début et la fin des paquets). Les en-têtes des trames Ethernet.

B.10.1.1.3. La couche réseau et transport

- **Adressage IP**

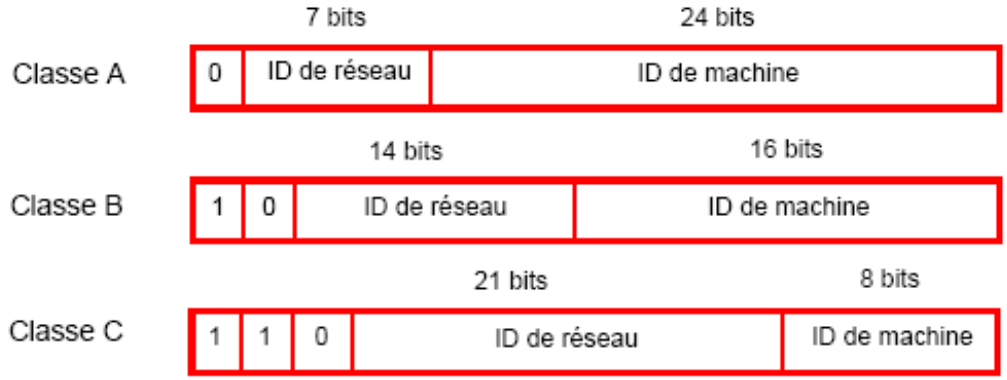
▪La couche de réseau résout le problème de l'acheminement de paquets à travers un seul réseau et vers un réseau destinataire. Ceci implique généralement le routage des paquets à travers un réseau de réseaux, connu sous le nom d'Internet.

▪Chaque station est identifiée par une adresse IP qui lui est propre.

▪L'unicité des adresses est gérée de la manière suivante :

- Lorsque l'environnement réseau est de type ouvert, l'unicité de l'adresse est garantie par l'attribution d'un identificateur réseau par l'autorité appropriée dans le pays où le réseau est situé ;
- Si le type d'environnement est fermé, l'unicité de l'adresse est gérée par le gestionnaire du réseau de l'entreprise.

▪Une adresse IP est définie par 32 bits. Elle est constituée de 4 nombres, un pour chaque octet de l'adresse, en fonction de la taille du réseau, trois classes d'adresses peuvent être utilisées :



Espaces réservés pour les différentes classes d'adresses IP :

Classe	Plage
A	**0**.0.0.0 à **127**.255.255.255
B	**128**.0.0.0 à **191**.255.255.255
C	**192**.0.0.0 à **223**.255.255.255

Figure B.28 : Les trois classes d'adresses

▪Une adresse IP est composée de deux identificateurs, un qui identifie le réseau et l'autre qui identifie la machine connectée. En réalité, l'identificateur de la machine peut également contenir un identificateur de sous-réseau.

▪Dans un environnement ouvert, ayant reçu un identificateur de l'autorité appropriée, l'administrateur du système local a la possibilité de gérer plusieurs réseaux. Cela signifie que des réseaux locaux peuvent être installés sans affecter le monde extérieur, qui ne voit qu'un seul réseau désigné par son identificateur.

▪Le masque de sous-réseau permet de visualiser le nombre de bits attribués respectivement à l'identificateur du réseau et à l'identificateur du sous-réseau (bits à 1), puis à l'identificateur de la machine (bits à 0).

▪Exemple : 140.186.90.3

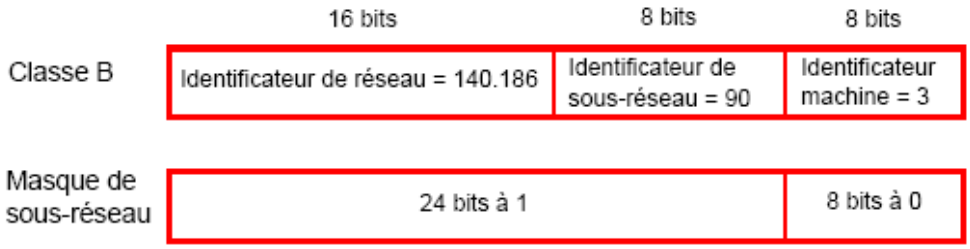


Figure B.29 : Exemple de classe réseau

- **le protocole TCP**

TCP est un protocole de transport, orienté connexion. Il fournit un flux d'octets fiable assurant l'arrivée des données sans altération et dans l'ordre, avec retransmission en cas de perte, et élimination des données dupliquées.

B.10.1.1.4. La couche application :

C'est dans la couche application que se situent la plupart des fonctions applications du réseau. Elles incluent HTTP (World Wilde Web), FTP (transfert de fichiers), SMTP (messagerie), SSH (connexion à distance sécurisée), DNS (recherche de correspondance entre noms et adresses IP) et beaucoup d'autres.

B.10.1.2. Modbus Plus

Modbus Plus est un réseau local conçu pour les applications de contrôle industriel. Le réseau permet à des automates programmables, des ordinateurs hôtes et d'autres appareils, de communiquer entre eux entre les différentes zones de production d'une usine. Il supporte jusqu'à 64 appareils abonnés adressables,

Il est possible de raccorder directement jusqu'à 32 appareils sur le câble de réseau, sur une distance de 450 mètres. Les répéteurs servent à augmenter la longueur de câble et le nombre d'abonnés possibles.

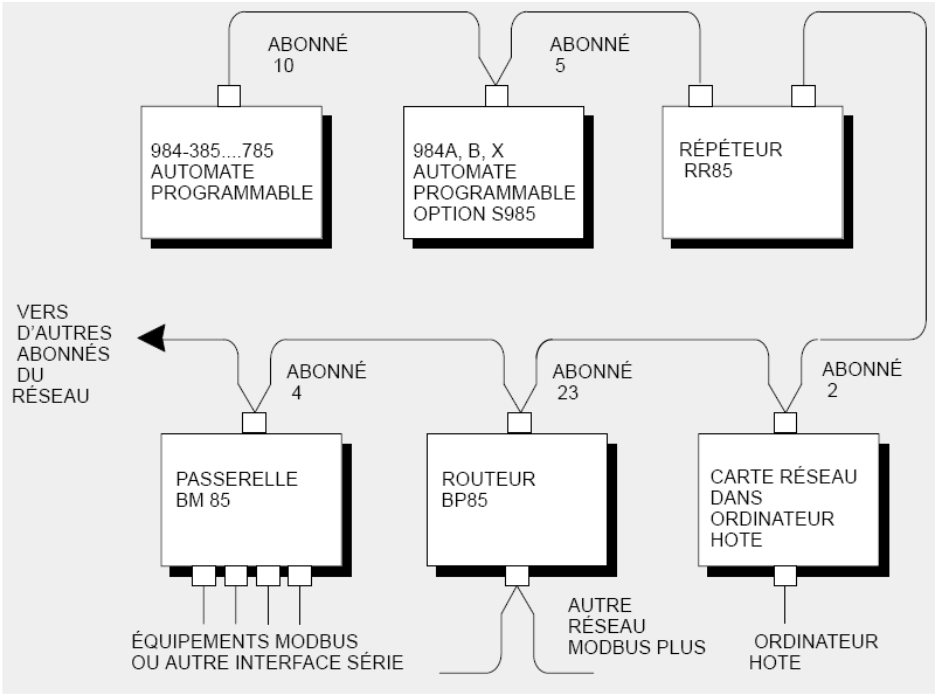


Figure B.30 : Exemple de structure Modbus plus

▪Le système d'adressage Modbus Plus est basé sur le chemin d'accès à suivre pour atteindre l'équipement destinataire. Ce chemin est déterminé par les routeurs Modbus Plus, également appelés ponts Plus. Ainsi, lorsqu'un équipement doit communiquer avec un autre, il est nécessaire de déterminer le chemin pris par les données à communiquer.

▪Un segment de réseau Modbus Plus peut contenir jusqu'à 64 équipements adressables. Chaque équipement possède une adresse unique comprise entre 1 et 64.
Plusieurs segments peuvent être liés par des ponts Plus.

▪Le chemin de routage est déterminé par les 5 octets qui indiquent successivement les adresses des équipements à parcourir avant d'arriver à destination.

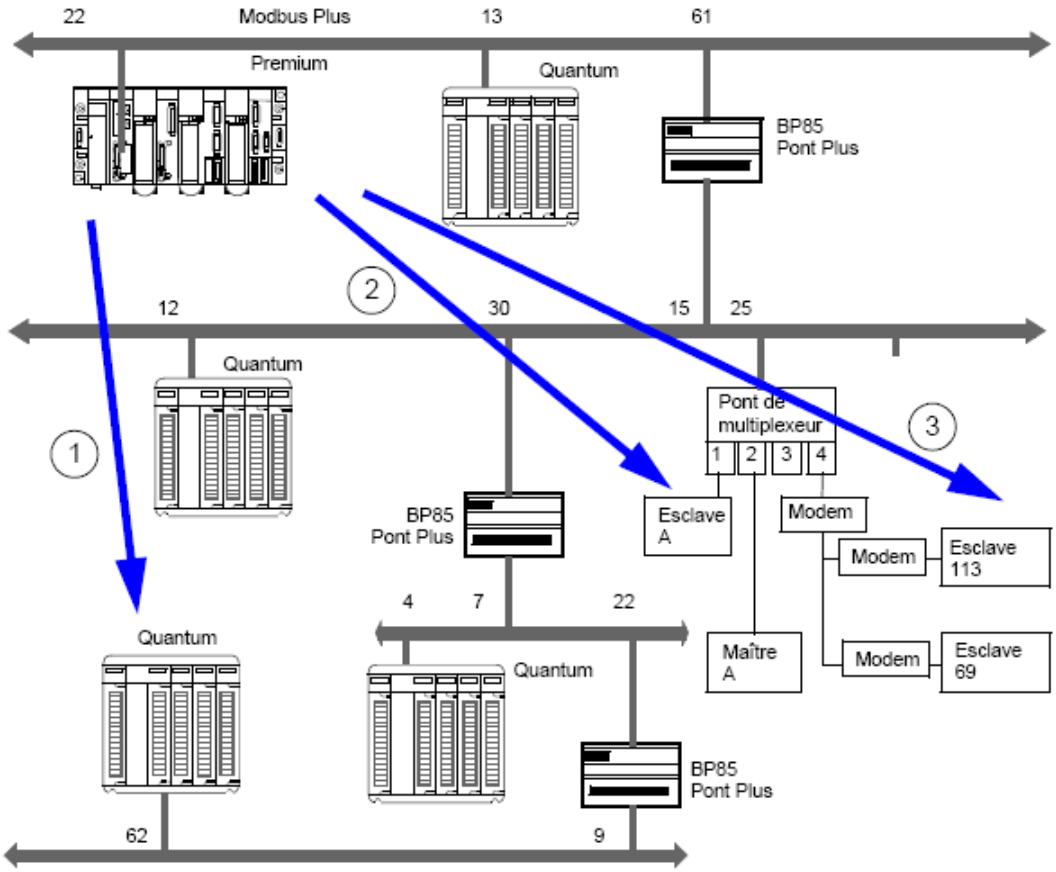


Figure B.31 : Structure d'un réseau Modbus Plus multi-segment

(1) : Le chemin de routage pour accéder à la station Quantum est : **61, 30, 22, 62, 0**.
(2) : Le chemin de routage pour accéder à l'esclave A est : **61, 25, 1, 0, 0**.
(3) : Le chemin de routage pour accéder à l'esclave 113 est : **61, 25, 4, 113, 0**.

B.10.1.3. Le protocole Modbus sur TCP/IP

Le protocole Modbus sur TCP/IP est défini comme suit

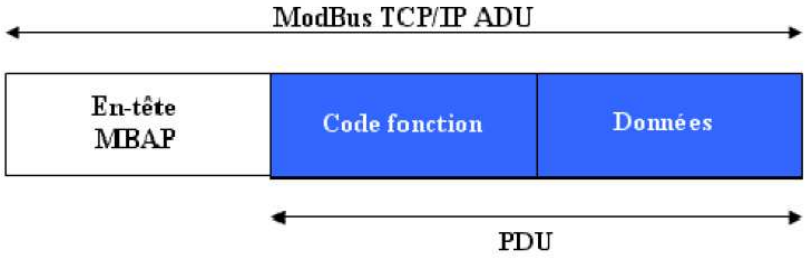


Figure B.32 : Trame modbus

-ADU : Application Data Unit ;
-PDU : Protocol Data Unit ;
▪Toutes les requêtes et réponses de Modbus sont conçues de telle manière que le destinataire puisse vérifier qu'un message est fini. L'information de longueur est diffusée dans l'en-tête de MBAP composé de 7 octets pour permettre au destinataire d'identifier les frontières des messages même si le message a été coupé en plusieurs paquets pour la transmission. Donc,

un même coupleur peut communiquer avec un équipement distant en mode client (par exemple un automate Quantum) et un autre équipement distant en mode serveur (par exemple un PC superviseur).

B.10.1.4. CAN open

B.10.1.4.1. Présentation

CAN est un bus série basé sur un modèle publication-souscription. L'expéditeur (éditeur) émet le message accompagné d'un identifiant. Les destinataires (souscrivants) filtrent les messages du bus suivant leurs critères d'envoi. Si ce message leur est destiné, ils vont alors le lire et le traiter. Le destinataire devient alors un expéditeur.

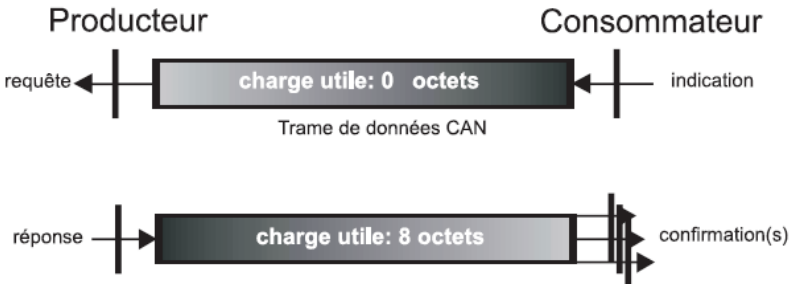


Figure B.33: Les modes « push » et « pull » du modèle publication-souscription

B.10.1.4.2. La trame CAN

La figure qui suit représente une trame CAN

Figure B.34: Trame CAN

-SOT : Start Of Trame
-Les bits de contrôle sont : IDE, DLC
-FCS : séquence de vérification de la trame
-ACK : bit récessif d'acquittement
-EOF : Le bit fin de trame
-IFS : Intermission Frame Space

▪Le bus CAN est souvent décrit comme un CSMA/CA (accès multiple avec écoute de porteuse et évitement de collision).

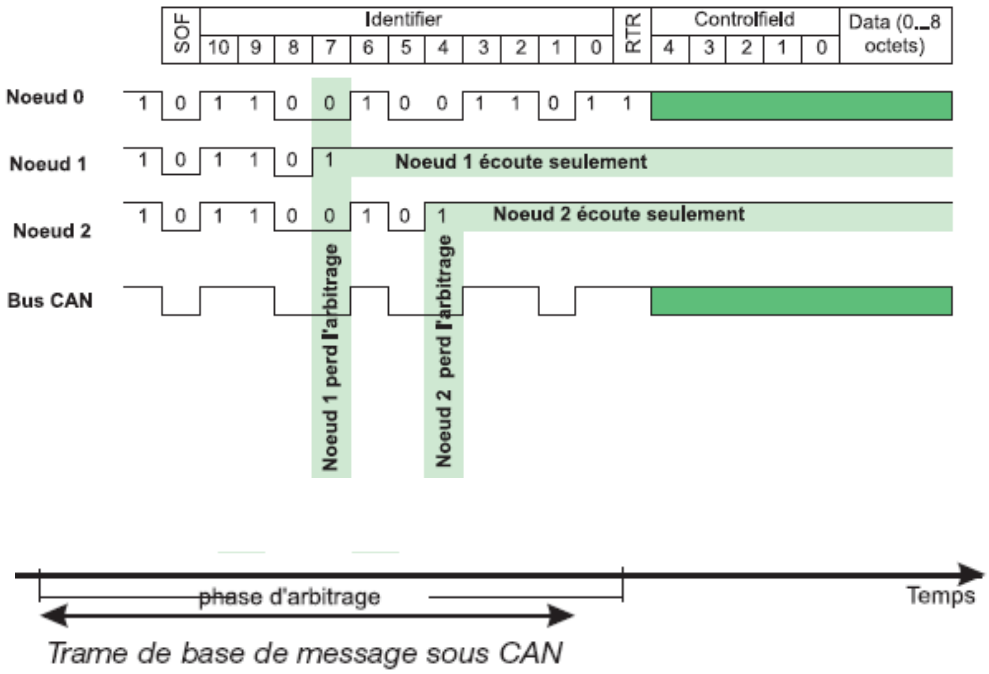


Figure B.35: Une trame de base

B.10.1.5. Bus AS-Interface (AS-I)

B.10.1.5.1. Présentation

AS-Interface est le réseau niveau capteurs / actionneurs qui correspond aux besoins des systèmes d'automatismes industriels. Le réseau AS-Interface véhicule la puissance et les données nécessaires sur un seul câble composé de deux fils.

▪AS-Interface est caractérisé par un câble jaune et une forme particulière, qui permet d'éviter les inversions entre les deux fils.

▪AS-Interface est exclusivement un bus de terrain de type maître/esclave qui comprend un "maître" (PC, API, Contrôleur …) chargé d'assurer la gestion des états des capteurs/actionneurs et de les transmettre à l'automatisme.

Figure B.36 : Constituants de l'AS-Interface

B.10.1.5.2. Les avantages d'AS-Interface

Les avantages de l'ASI sont :

- La simplicité du système de câblage est due à :
  - L'usage d'un câble unique pour raccorder tous les actionneurs et capteurs dans un système d'automatisme ;
  - La gestion des communications intégrée aux produits.
- Les coûts peuvent être réduits jusqu'à 40 % par :
  - Réduction des temps de conception, installation, mise en service et évolutions ;
  - Espace gagné dans les enveloppes, grâce à des produits plus compacts et à l'élimination des enveloppes intermédiaires depuis que la majorité des fonctions peuvent être délocalisées sur la machine ;
  - Elimination des canalisations des câbles de contrôle.

- AS-Interface permet d'améliorer la fiabilité, la disponibilité opérationnelle et la sécurité :
  - Les erreurs de câblage ne sont plus possibles ;
  - Pas de risque de mauvaise connexion ;
  - Haute immunité aux interférences électromagnétiques (EMC) ;
  - Les fonctions de sécurité de la machine peuvent être totalement intégrées dans AS-Interface.

B.10.1.5.3. Les composants d'AS-Interface

Les composants d'ASI interface sont :

- Les interfaces pour produits génériques : Ils permettent à n'importe quel produit standard (capteur, actionneur, démarreur, etc.) d'être connecté sur un réseau AS-Interface ;
- Les interfaces dédiées : (modules de communication...) ils permettent la communication avec le câble AS-Interface ;
- Les composants dédiés : intègrent une interface et peuvent donc être connectés directement sur le câble AS-Interface ;
- Le maître : C'est le composant central du système, sa fonction est de gérer les échanges de données avec les interfaces et composants (appelés aussi esclaves) répartis dans l'installation. Il peut recevoir :
  -31 interfaces ou composants en version V1 (temps cycle 5ms) ;
  -62 interfaces ou composants en version V2 (temps de cycle 10ms) ;

  Le maître est : soit intégré dans un contrôleur programmable, sous la forme d'une extension par exemple, soit connecté au bus de terrain, il s'agit alors d'une passerelle ;
- L'alimentation AS-Interface : C'est une source très basse tension de 29.5 à 31.6V ;
- Le câble plat : Le câble jaune, connecté à l'alimentation de puissance, assure les deux fonctions suivantes :
  - Transmettre les données entre le maître et les esclaves ;
  - Alimenter les capteurs et actionneurs ;

- Le terminal d'adressage : Puisque les composants sont connectés en parallèle sur le bus AS-Interface, il est nécessaire d'assigner une adresse différente à chacun. Cette fonction est assurée par un terminal qui se raccorde individuellement aux différents composants.

B.10.1.5.4. Principe de fonctionnement du réseau AS-i

B.10.1.5.4.1. Connectique ou prise vampire

Le connecteur comporte deux aiguilles qui effectuent la liaison électrique par percement de l'isolant du câble. Les deux parties du connecteur sont ensuite visées l'une sur l'autre pour assurer la qualité de la connexion.

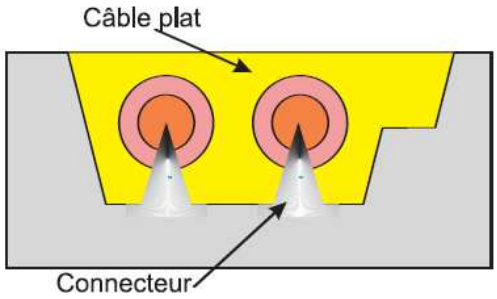


Figure B.37: Principe du raccordement As-i

B.10.1.5.4.2. Modulation des signaux

Le principe utilise la modulation du courant basée sur un encodage Manchester, deux inductances situées dans l'alimentation convertissent ce signal en tension sinusoïdale.

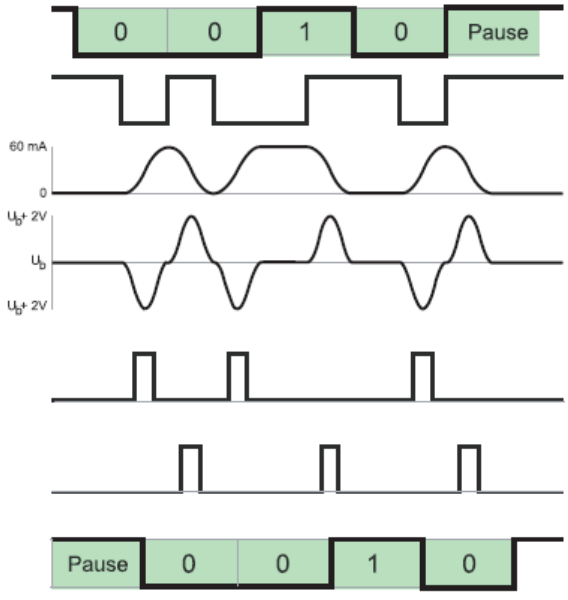


Figure B.38: Forme des signaux courant et tension

B.10.1.5.4.3. Principe du protocole

Le principe de communication est basé sur un protocole à un seul maître. Le maître système interroge les esclaves à tour de rôle, ceux-ci répondent en envoyant les données requises. Quand tous les esclaves ont été interrogés, le cycle se répète et continue indéfiniment.

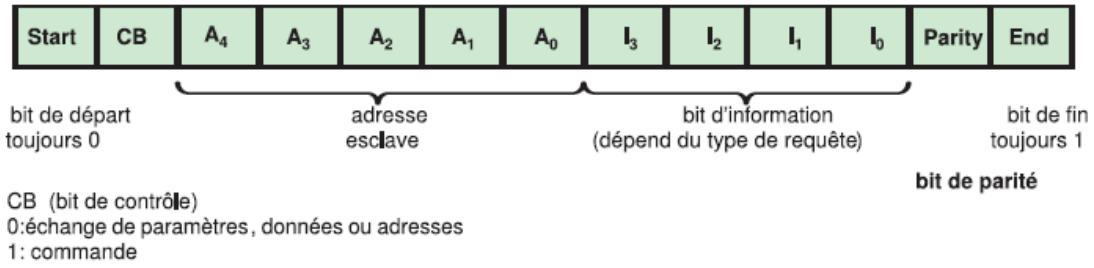


Figure B.39 : Trames maître

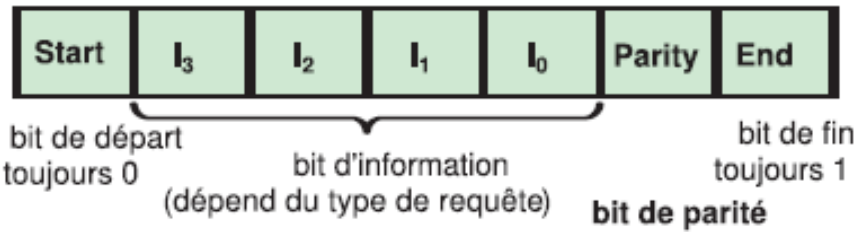


Figure B.40: Trame esclave

B.10.1.5.4.4. Longueur du réseau

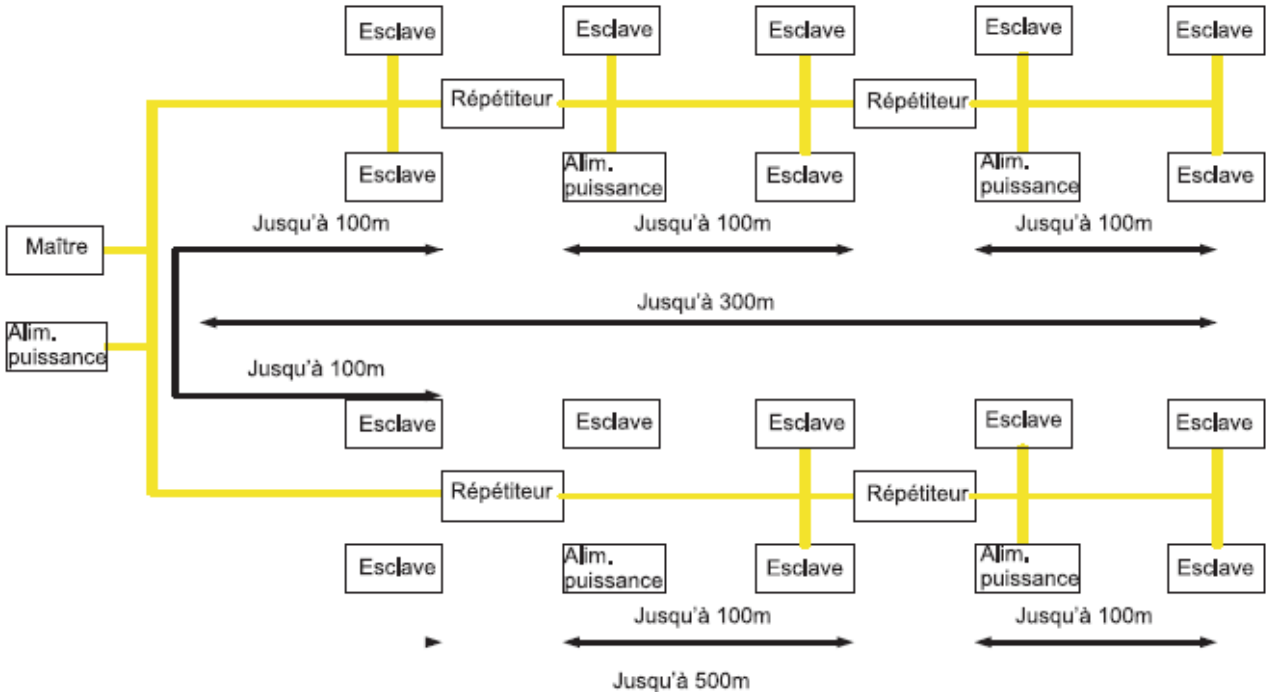


Figure B.41 : Longueurs limites du réseau As-i

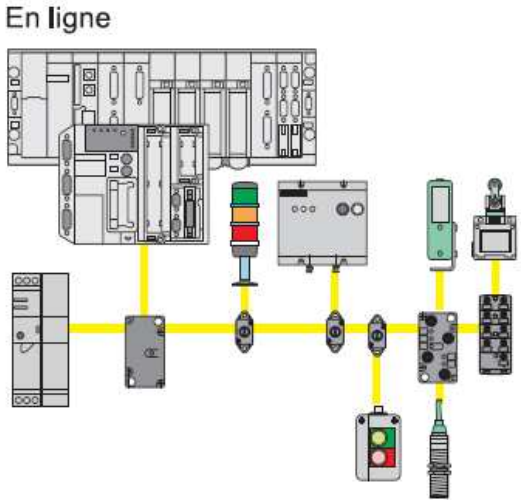


Figure B.42 : AS-I sur un réseau en BUS

B.10.2. Les services disponibles par communication

Les services disponibles peuvent être classés en trois catégories :

- Services de messagerie explicite : Messagerie Modbus, Messagerie UNI-TE, Télégrammes. Ces services permettent à un équipement (Client) d'envoyer un message à un autre équipement (Serveur) et d'obtenir une réponse et ce, sans avoir à programmer quoique ce soit dans l'équipement serveur. Ils sont utilisés pour accéder aux données de temps en temps ;

- Services d'accès implicite à une base de données : Global data, Mots communs, Tables partagées. Ces services permettent de partager des données réactualisées automatiquement et régulièrement. Ils sont utilisés pour synchroniser des applications ou pour obtenir, de manière transparente, des images en temps réel d'un système situé sur plusieurs automates distants ;

- Services de gestion implicite des entrées/sorties : I/O Scanning, Peer cop. Ces services permettent de gérer des E/S distantes sur un réseau, de manière transparente et automatique. Ils sont utilisés pour surveiller un ensemble de systèmes distribués dans un réseau.

Fonction	Fipway	Fipio	Uni-Telway	Mode caractères	Modbus/ Jbus	Modbus Plus	Ethway	TCP/IP	CANopen	USB
Services de messagerie										
Fonctions de communication	Les fonctions de communication utilisables dépendent étroitement du type de communication sur lesquelles elles s'appliquent (voir *Disponibilités des fonctions selon les protocoles, p. 36*).									
Services d'accès implicite à une base de données										
Global Data	-	-	-	-	-	-	-	X	-	-
Mots communs	X	-	-	-	-	-	X	-	-	-
Tables partagées	X	-	-	-	-	-	X	-	-	-
Echanges de données périodiques	-	X	-	-	-	-	-	-	-	-
Services de gestion implicite des E/S										
Scrutation d'E/S	-	-	-	-	-	-	-	X	-	-
Peer Cop	-	-	-	-	-	X	-	-	-	-
Autre	-	X	-	-	-	X	-	-	X	-
Légende :										
X	Oui									
-	Non									

Tableaux B.1 : Les différents services disponibles en fonction des types de communication

B.10.2.1. Service Global Data

L'objectif du service Global Data, pris en charge par les modules Ethernet, est de fournir un échange de données automatique pour la coordination d'applications d'automate. Les données sont partagées selon une méthode de publication/ souscription entre les différents équipements.

- Fonctionnement : Les modules de communication sont regroupés dans un groupe de distribution. Chaque module de communication publie une variable d'application locale vers les autres modules de communication du groupe de distribution. Chaque module de communication peut également souscrire aux variables d'applications publiées par tous les autres modules appartenant au même groupe de distribution.

▪Le service Global Data doit être configuré afin de déterminer l'emplacement et le nombre de variables d'applications de chaque module de communication. Une fois les modules configurés, les échanges entre les modules de communication appartenant au même groupe sont réalisés automatiquement dans l'automate en mode RUN.

Illustration :

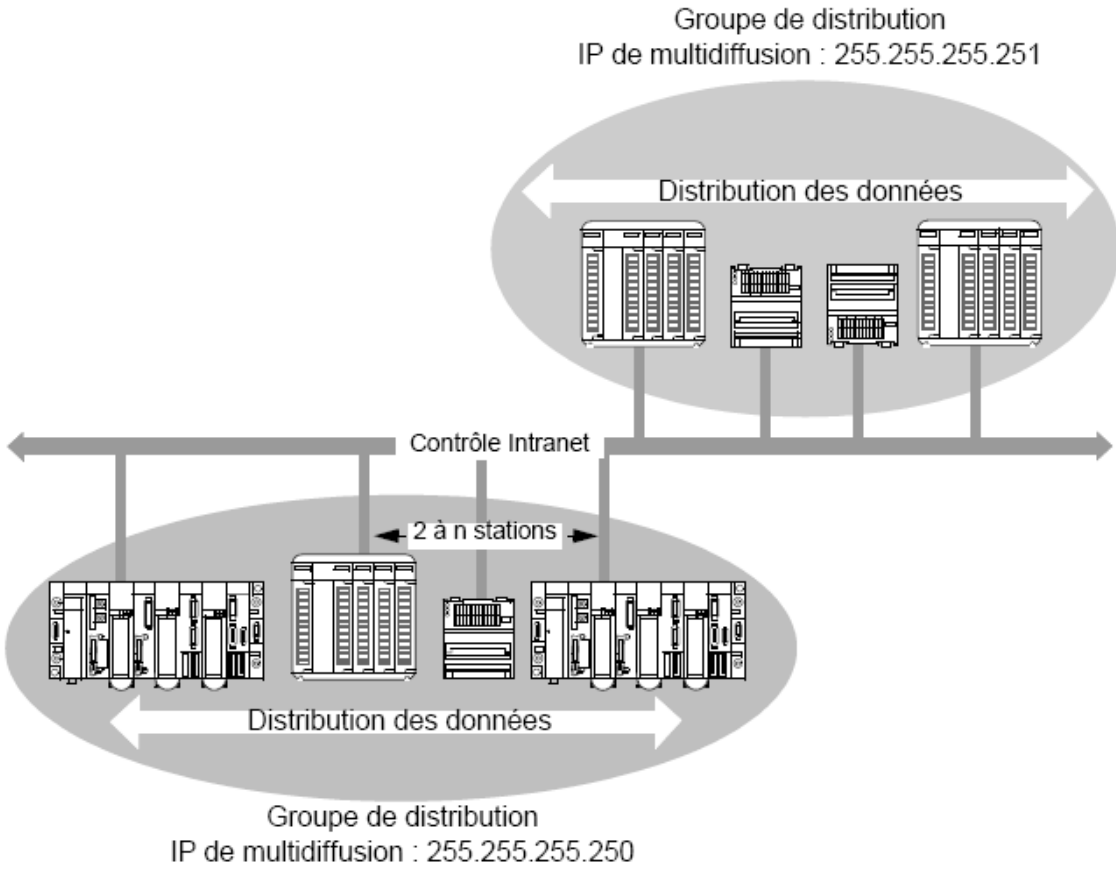


Figure B.43 : Le service Global Data

▪Un groupe de distribution est un groupe de modules de communication identifiés par la même adresse IP de multidiffusion. Des échanges en 'multidiffusion' sont utilisés pour distribuer les données globales. Plusieurs groupes de distribution indépendants peuvent coexister sur un même sous-réseau avec leur propre adresse de multidiffusion.

▪Un protocole Publication/Souscription sur UDP/IP est utilisé pour la distribution des données.

▪Et il n'y a pas de limites théoriques au nombre de stations pouvant appartenir à un groupe de distribution. La principale limitation est le nombre de variables échangées dans un groupe (64 variables).

B.10.2.2. Service IO Scanning

Le scrutateur d'ES permet, de manière périodique, de lire ou d'écrire des entrées/ sorties distantes sur le réseau Ethernet sans programmation spécifique ;

▪Ce service comporte les éléments essentiels suivants :

- Une zone de lecture regroupant toutes les valeurs des entrées distantes ;
- Une zone d'écriture regroupant toutes les valeurs des sorties distantes ;

- Des périodes de scrutation indépendantes du cycle automate et dédiées à vérifier chaque équipement distant.

- Fonctionnement : Ce service fonctionne avec tous les équipements supportant la communication Modbus sur le profil TCP/IP en mode serveur. Le mécanisme d'échange qui est transparent, est effectué par des requêtes :
  - requêtes de lecture ;
  - requêtes d'écriture ;
  - requêtes de lecture et d'écriture.

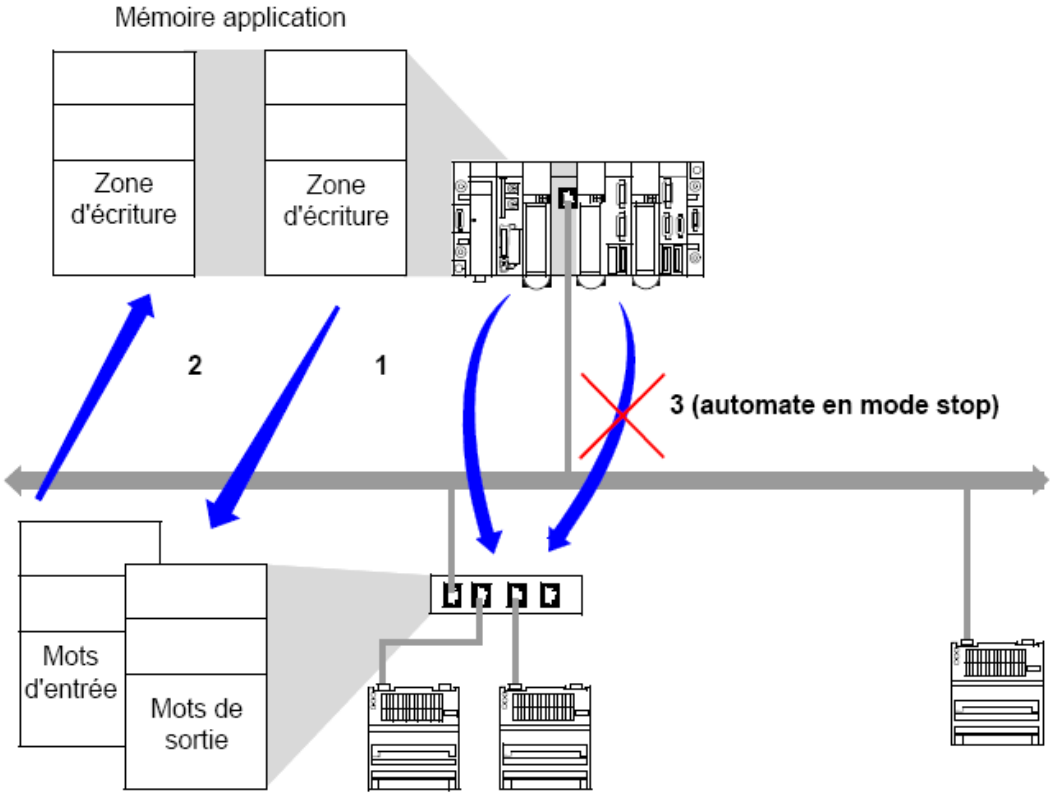


Figure B.44 : Le fonctionnement de la scrutation des entrées/sorties distantes.

1. Dès que l'automate passe en mode Run, le module ouvre une connexion par équipement scruté ;
2. Ensuite, le module effectue une lecture périodique des mots d'entrée et une écriture périodique des mots de sortie de chaque équipement ;
3. Lorsque l'automate passe en mode Stop, les connexions à chaque équipement sont fermées.

B.10.3. Technologies des réseaux

B.10.3.1. Topologie des réseaux

Un réseau industriel est constitué d'automates programmables, des interfaces hommes/machines, d'ordinateurs, des équipements d'entrées/sorties, reliés entre eux grâce à des lignes de communication, telles que des câbles électriques, des fibres optiques, des liaisons radio et des éléments d'interface, tels que des cartes réseaux, des gateways. L'arrangement physique du réseau est appelé **topologie physique** ou architecture du réseau.

Lorsque l'on considère la circulation des informations, on utilise la terminologie de topologie logique. On distingue généralement les topologies suivantes :

- en bus,
- en étoile,
- en arbre,
- en anneau,
- maillées.

B.10.3.1.1. Topologie en bus

Cette organisation est une des plus simples. Tous les éléments sont reliés à une même ligne de transmission par l'intermédiaire de câbles. Le mot bus désigne la ligne physique. Cette topologie est facile à mettre en œuvre, la défaillance d'un nœud ou d'un élément ne perturbe pas le fonctionnement des autres organes. Les réseaux du niveau machine et capteurs, appelés d'ailleurs bus de terrain, utilisent cette méthode. La typologie bus se met en œuvre soit par chaînage des équipements les uns avec les autres, soit par connexion via un boîtier de raccordement (TAP) au câble principal.

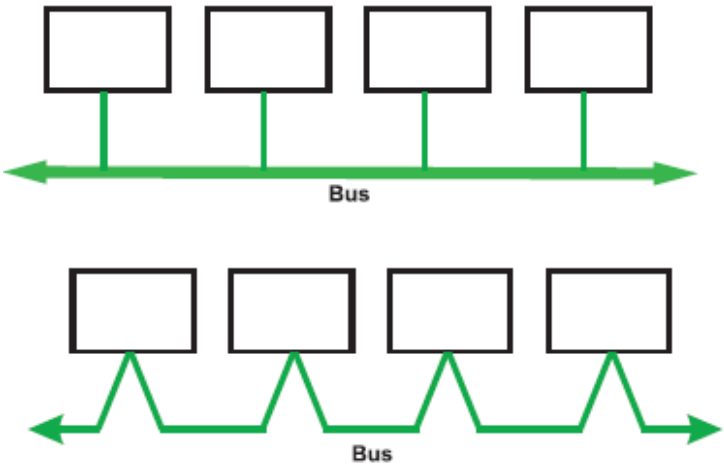


Figure B.44 : Topologie en bus

B.10.3.1.2. Topologie en étoile

Cette typologie est la plus courante au niveau de l'entreprise et de l'atelier. Elle est celle du réseau Ethernet. Elle présente l'avantage d'être très souple en matière de gestion et de dépannage. Les stations finales sont reliées ensemble à travers un équipement intermédiaire (répéteur, commutateur). La défaillance d'un nœud ne perturbe pas le fonctionnement global du réseau, en revanche, l'équipement intermédiaire qui relie tous les nœuds constitue un point unique de défaillance.

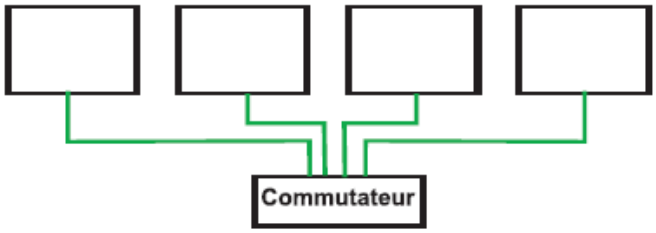


Figure B.45 : Topologie des réseaux en étoile

B.10.3.1.3. La typologie en anneaux

Reprend la topologie physique de l'étoile en apportant une meilleure disponibilité du réseau.

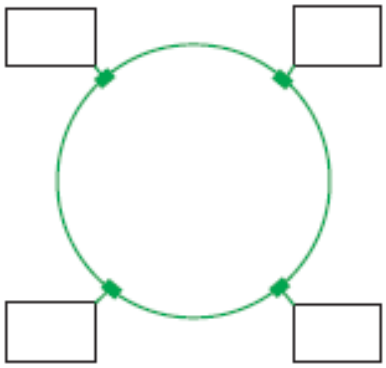


Figure B.46 : Topologie des réseaux en anneaux

B.10.3.1.4. La topologie en réseau maillé

Cette topologie est assez peu utilisée dans l'industrie et présente l'inconvénient d'un nombre important de liaisons.

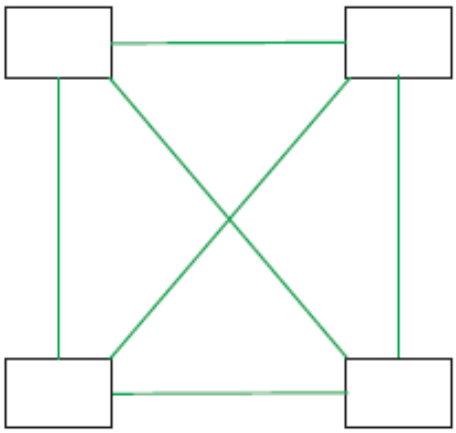


Figure B.47 : La topologie en réseau maillé

B.10.3.2. L'architecture globale d'un réseau

Sous l'effet conjugué des contraintes des utilisateurs, des technologies et des standards, les architectures actuelles se structurent en quatre niveaux distincts et interconnectés par des réseaux.

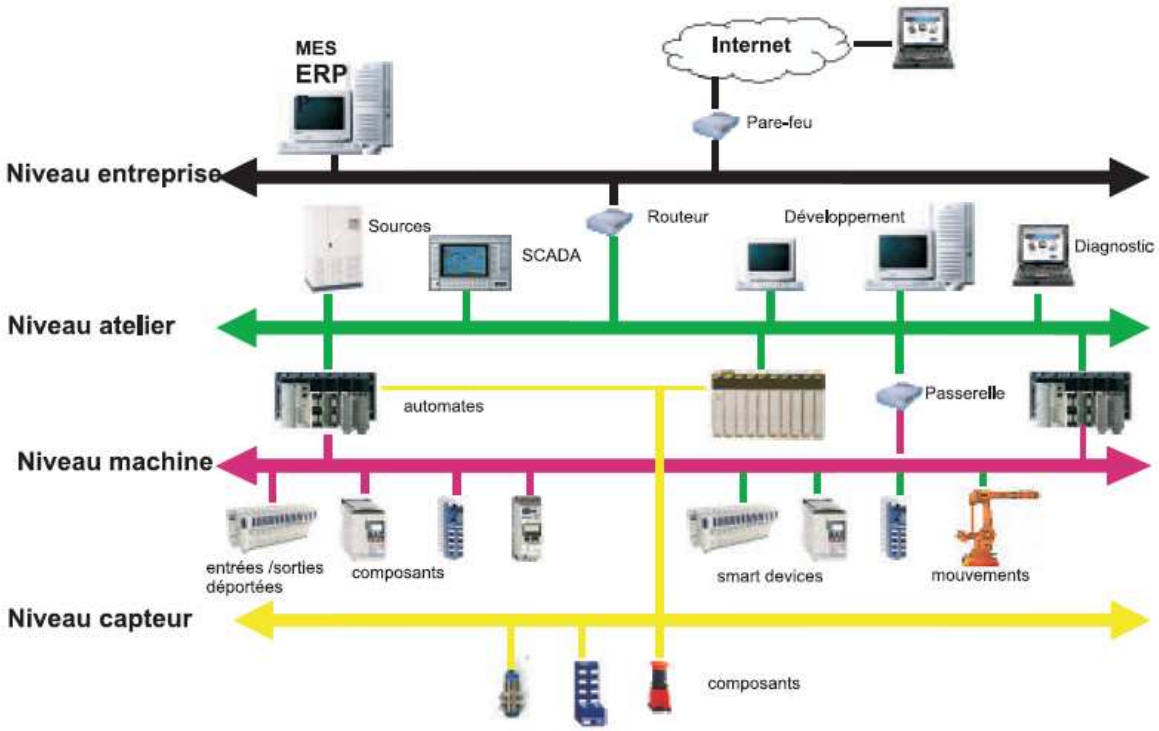


Figure B.48 : Exemple de niveaux d'architecture

Niveau	Besoin	Volume d'informations à transmettre	Temps de réponse	Distance	Topologie réseau	Nombre d'adresses	Médium
Entreprise	Echange de données. Sécurité informatique. Standards entre progiciels.	Fichiers Mbits	1mn	Monde	Bus, étoile	Non limitée	Electrique, optique, radio
Atelier	Synchronisation des API' d'un même îlot d'automatisme échanges d'information en mode client/serveur avec les outils de conduite (HMI, supervision). Performances Temps réel.	Données Kbits	50 à 500 ms	2 à 40 Km	Bus, étoile	10 à 100	Electrique, optique, radio
Machine	Architecture distribuée. Intégration fonctionnelle et transparence des échanges. Topologie et coût de connexion.	Données Kbits	5 à 100 ms (cycle de l'API)	10 m à 1 Km	Bus, étoile	10 à 100	Electrique, optique, radio
Capteur	Simplification du câblage distribution des alimentations des capteurs et actionneurs. Optimiser les coûts de câblage.	Données bits	T	1 à 100 m	Sans contrainte	10 à 50	Electrique radio

Tableaux B.2 : Les besoins et les contraintes de communication

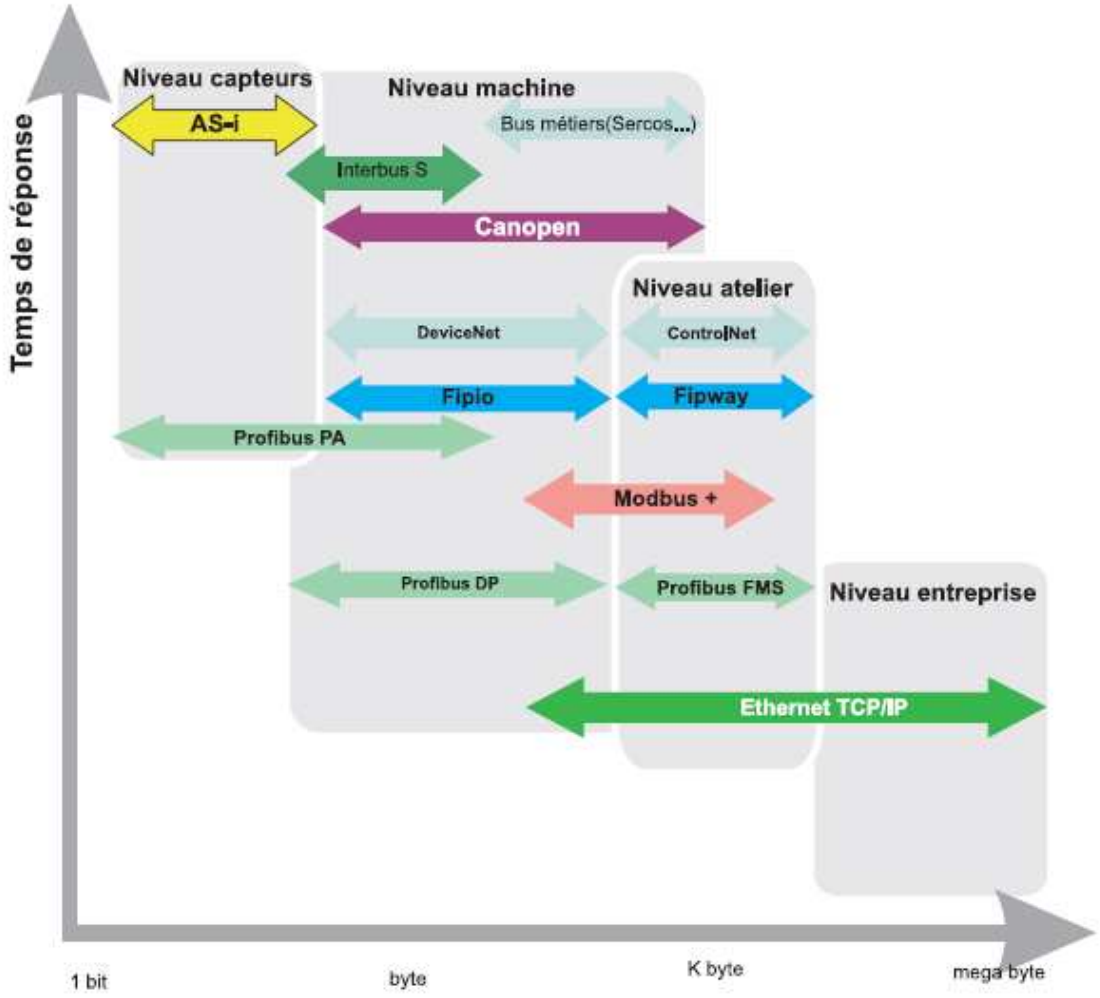


Figure B.49 : Principaux réseaux industriels

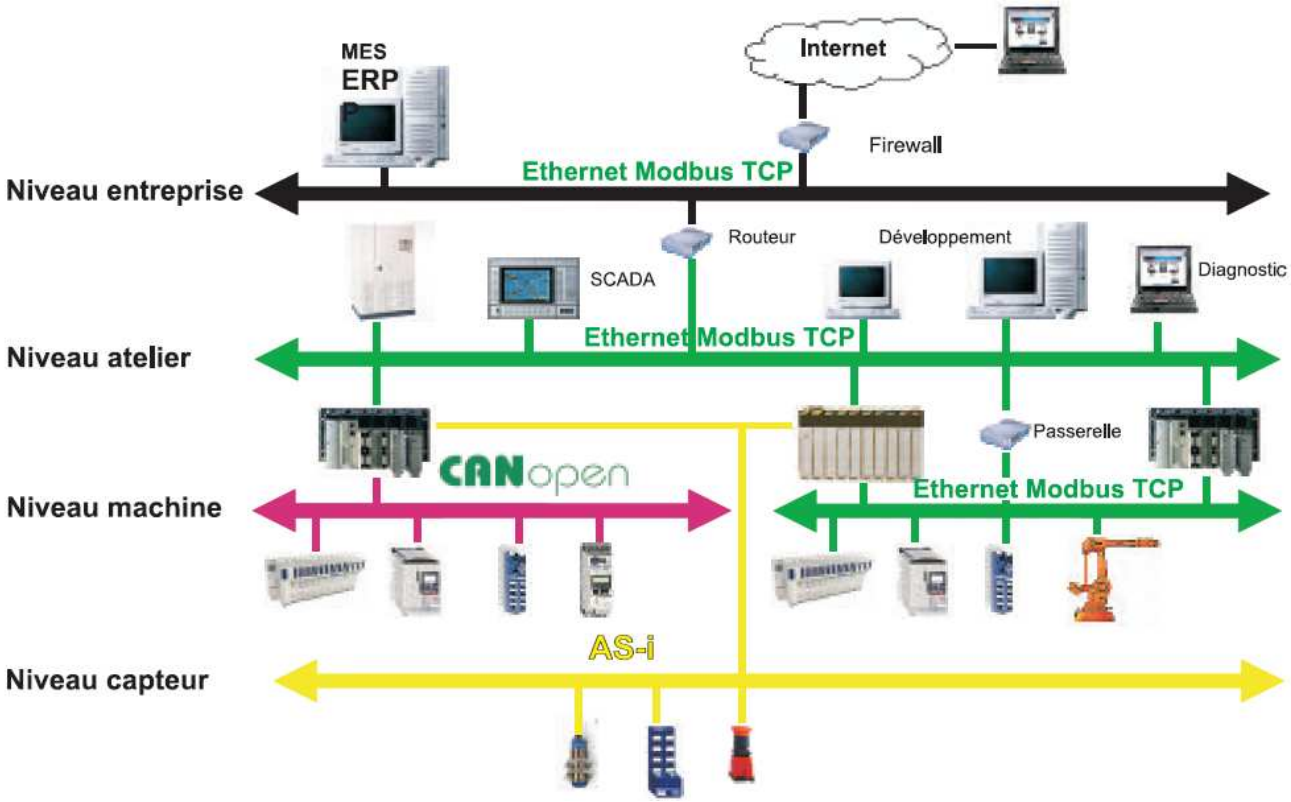


Figure B.50 : Les niveaux de communication retenus par Schneider Electric

Annexe C

Logiciel Vijeo Citect

C.1. Variables

C.1.1. Affectation des noms aux variables

Vijeo Citect met certaines restriction pour l'affectation des noms aux différentes variable, mais en utilisant cette méthode de nomination, le projet crée sera plus facile à manipuler (pour les fichiers Cicodes) et à modifier quand on désire.

Chaque nom de variable peut contenir jusqu'à 32 caractères, pour être conforme avec la méthode, on divise le nom en 4 sections dont chacune d'elles nous donne une information sur la variable. La convention de nomination proposée par Vijeo Citect est de la forme:

Area_ Type_ Occurrence_ Attribut

▪La section Area permet en cas de duplication, de distinguer l'appartenance de chaque variable, par exemple si on a trois procédés ayant les mêmes fonctions, on déclare les variables du premier procédé et on les copie aux autres, pour différencier l'appartenance de chaque variable, on change seulement l'indice comme le montre le tableau :

Partie	Noms de variables
Procédé 1	**P1**_TIC_101_PV
Procédé 2	**P2**_TIC_101_PV
Procédé 3	**P3**_TIC_101_PV

Tableau C.1 : Section Area

▪La section Type comme son nom l'indique permet d'identifier le type du paramètre (équipement d'un processus, paramètre de contrôle,…) comme le montre le tableau suivant :

Nom de la variable	Sens
P1_**TIC**_101_PV	Indicateur de contrôle de la température
P1_**PUMP**_101_PV	Pompe
P1_**VALVE**_101_PV	Valve

Tableau C.2 : Section Type

▪La section Occurrence permet de définir le nombre de l'élément, si un élément est utilisé plusieurs fois, il suffit de changer le numéro dans cette section comme le montre le tableau si dessous :

Nom de la variable	Sens
P1_PUMP_**101**_PV	Pompe 101
P1_PUMP_**102**_PV	Pompe 102

Tableau C.3 : Section Occurrence

▪La section Attribut permet d'identifier la particularité du paramètre qui est associé à l'équipement comme le montre le tableau suivant :

Nom de la variable	Sens

P1_TIC_101_**PV**	Processus variable
P1_TIC_101_**V**	Valeur (déclencher/arrêter)
P1_TIC_101_**P**	Gain proportionnel
P1_TIC_101_**CMD**	Signal de commande d'une pompe
P1_TIC_101_**M**	Mode Auto/Manuel

Tableau C.4 : Section Attribut

C.1.2. Déclaration des variables

L'écran de déclaration des variables est le suivant :

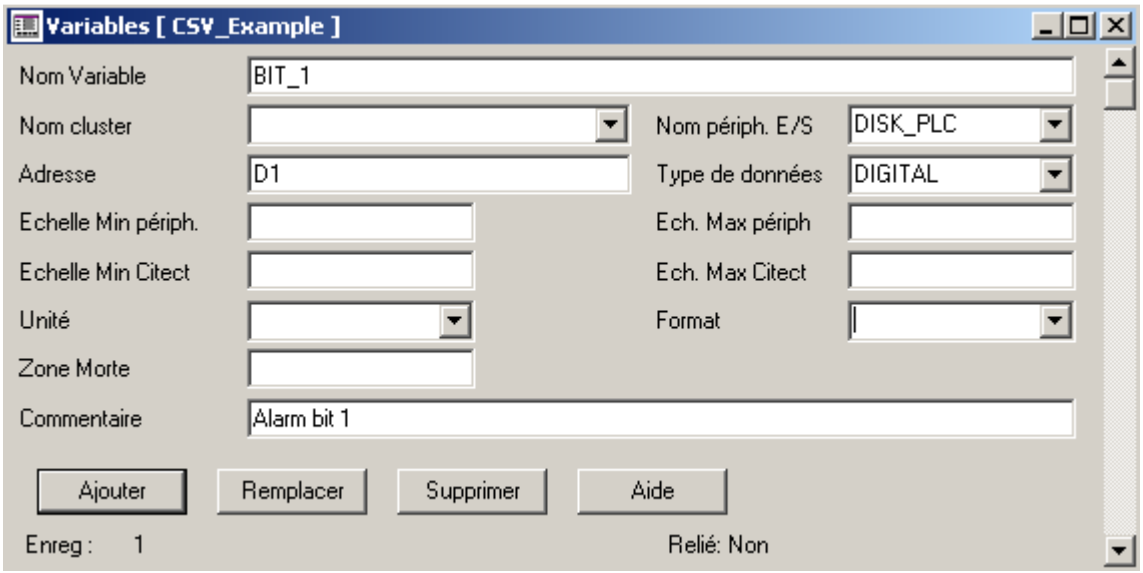


Figure C.1 : Ecran de déclaration des variables

▪Pour définir une variable, on doit spécifie son nom, le cluster de travail, nom du périphérique d'E/S, l'adresse de la variable et son type.

▪On spécifie l'échelle Min et Max du périphérique et l'échelle correspondante en Vijeo Citect (pour la représentation graphique).

▪On spécifie l'unité de la variable (seconde, volt, ampère, …) et le format de représentation dans Vijeo Citect.

▪Le champ zone morte permet d'insérer le pourcentage de la valeur, en unités de travail, des tags de variables devant être modifié pour qu'une mise à jour soit envoyée au système. La valeur par défaut est 1 %.

▪On utilise le bouton ajouter pour un nouvel enregistrement et le bouton remplacer pour le remplacement d'un enregistrement existant.

▪On peut aussi ajouter un commentaire.

C.1.3. Déclaration des variables locales

L'écran de déclaration des variables locales est :

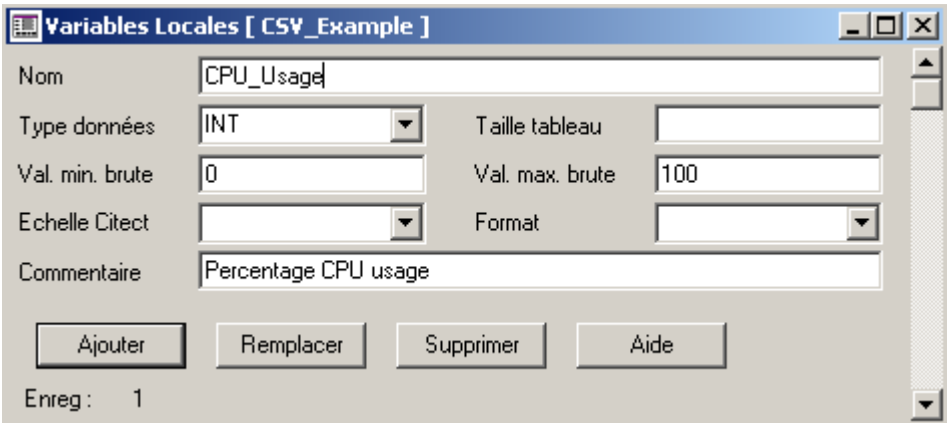


Figure C.2 : Ecran Variables locales

▪Pour définir une variable locale, on insère son nom, son type et sa valeur Min et Max.

▪si la variable est un tableau, on utilise le champ taille tableau pour insérer sa taille.

▪Le champ Echelle Citect permet d'insérer l'unité de la variable et on introduit aussi le format d'affichage.

C.1.4. Déclaration des variables de tendance

L'écran de déclaration de ce type de variable est :

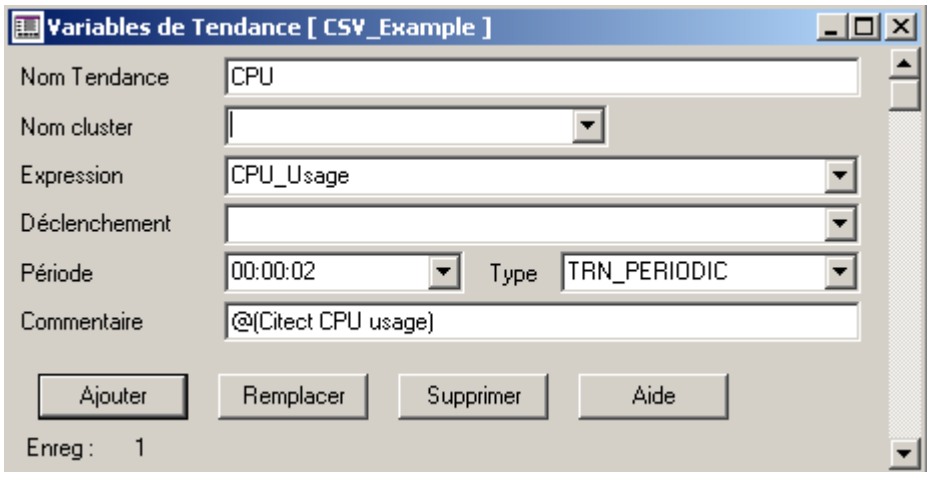


Figure C.3 : Ecran Variables de tendance

▪Pour définir ce type de variable, en plus des champs présenté précédemment, on introduit la valeur enregistrée du tag de tendance dans le champ expression.

▪On insère la variable ou l'expression Cicode provoquant le déclenchement dans le champ déclenchement.

▪Le champ Période permet d'insérer la période d'échantillonnage des données.

▪Le champ Type permet d'introduire le type de la tendance (TRN_PERIODIC correspond à une tendance échantillonnée continuellement à une fréquence spécifiée).

C.1.5. Déclaration des variables SPC

L'écran de déclaration des variables SPC est :

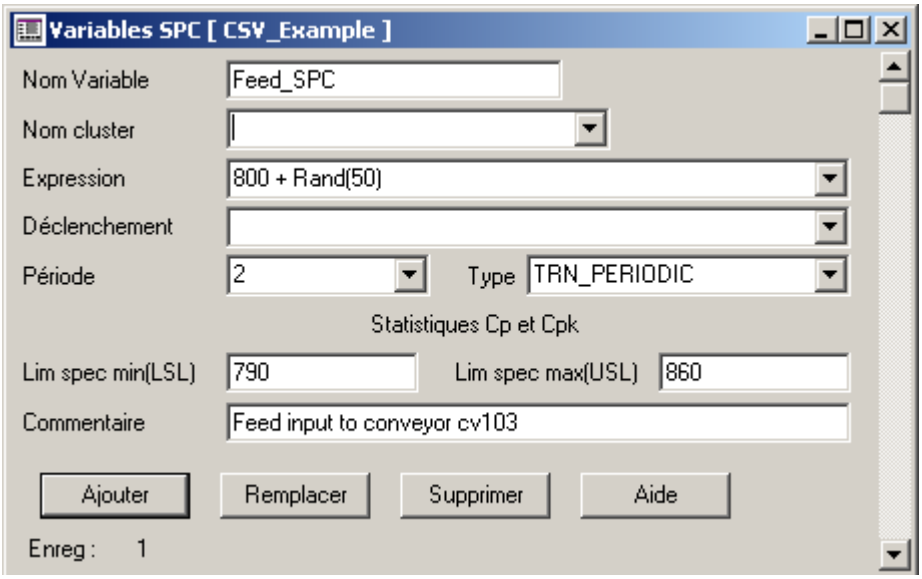


Figure C.4 : Ecran Variables SPC

▪En plus des champs présentés pour les variables tendances, on les champs Lim spec min et Lim spec max qui permettent d'insérer respectivement les limites inferieur et supérieur, ces valeur permettent de déterminer la capabilité du processus.

Remarque : il faut noter que Vijeo Citect offre la possibilité de déclarer les variables en utilisant des Microsoft Excel et ça en créant des fichiers (.dbf).

C.1.6. Types de données supportées par Vijeo Citect

Le tableau suivant présente les types de données supportées par Vijeo Citect :

Types de données	Variable	Taille	Valeurs autorisées
BCD	Nombre décimal codé en binaire	2 octets	0 à 9999
Byte	Octet	1 octet	0 à 255
Digital	Numérique	1bit ou 1 octet	0 ou 1
INT	Entier	2 octets	-32 768 à 32 767
UINT	Entier non signé	2 octets	0 à 65 535
LONG	Entier long	4 octets	-2 147 483 648

			à 2 147 483 647
LONGBCD	nombre décimal long codé en binaire	4 octets	0 à 99 999 999
REAL	Nombre à virgule flottante	4 octets	-3,4E38 à 3,4E38
STRING	chaine	256 octets	ASCII (terminé par un caractère nul)

Tableau C.5: Type de données Citect

C.2. Fichiers Cicodes

Vijeo Citect propose une manière de déclaration standard des variables suite à l'utilisation des fichiers Cicodes et ça en tenant compte de plusieurs considération, la syntaxe de déclaration est :

[<Scope>] <Type> <Nom> [=Valeur initial] < commentaire>

- Scope : présente l'étendue de la variable (module, locale ou globale) ;
- Type : type de la variable (INT, REAL,…) ;
- Nom : nom de la variable ;
- Valeur initial : valeur initial de la variable ;
- Commentaire : on peut ajouter un commentaire.

Exemple :

```
Module          int     miRecipeMax=100;

                int     iRecipeMax;


            string      sRecipeDefault  ="Tasty";
```

❖ **La nomenclature des variables**

▪Les noms des variables devraient avoir une petite lettre en préfixe dans les cas comme suit :

Type	Préfixe	Utiliser pour
INT (32 bits)	i	Boucle de comptage, index
INT (32 bits) et objet (32 bits)	h	Handle

INT (32 bits)	b	Booléen
REAL (32 bits)	r	Réel
STRING (255 bytes)	s	String

Tableau C.6 : Nomenclature des variables

▪Les noms des variables consistent typiquement jusqu'à trois mots. Chaque mot dans un nom variable devrait commencer avec une lettre capitale, par exemple :

iTrendType, rPeriod, sFileName.

▪Les noms des variables modules devraient être préfixés avec un "m", par exemple : miTrendType, mrPeriod, msFileName.

▪Les noms des variables globaux devraient être préfixés avec un "g", par exemple: giTrendType, grPeriod, gsFileName.

▪Les noms des variables locaux ne devraient pas être préfixés (quand on déclare une variable sans spécifier l'étendue, elle est considérée comme une variable locale par défaut), par exemple : iTrendType, rPeriod, sFileName.

▪Dans Cicode, il n'y a aucun équivalent de « # » définit du langage C, ou un type qui forcera des variables à être des constantes (variables lecture seule). Cependant, la convention de la nomenclature des variables rend facilement des constantes identifiables donc les développeurs traiteront ces variables comme variables lecture seule.

▪Les constantes doivent avoir le préfixe "c", elles doivent être déclarées et initialisées au commencement du fichier Cicode, exemple :

```
   INT    ciTrendTypePeriodic = 1;
   INT    ciTrendTypeEvent = 2;
STRING csPageName = "Mimic";
```

C.3. Configuration des alarmes

C.3.1. Configuration des alarmes digitales

Pour la configuration on utilise l'écran suivant quand trouve dans le dossier alarmes :

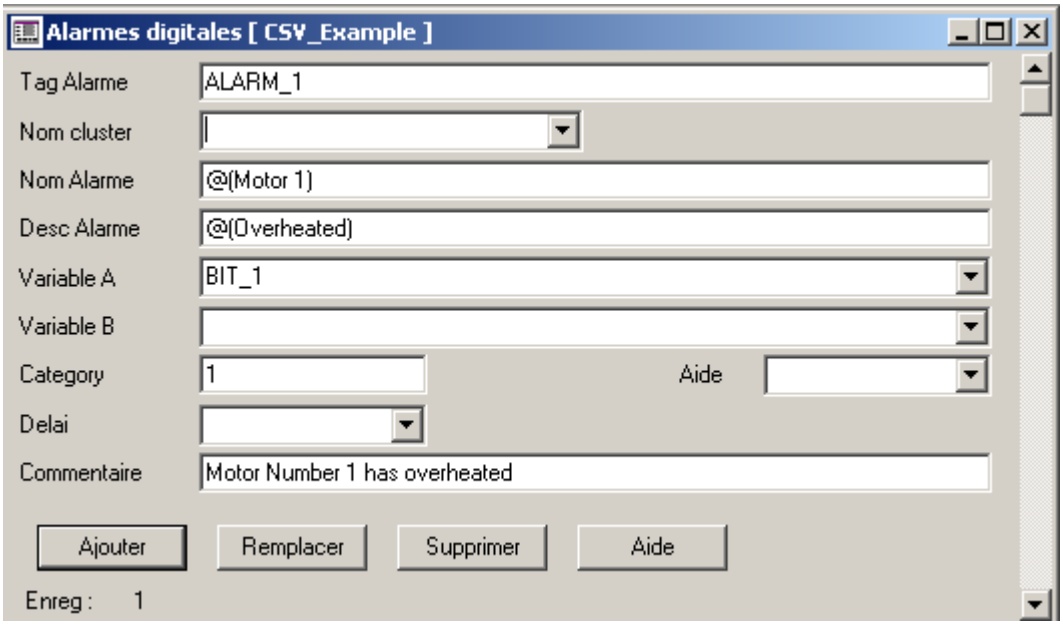


Figure C.5 : Ecran Alarmes digitales

▪On introduit un nom à l'alarme dans le champ tag Alarme (maximum 79 caractères) et on spécifie le nom du cluster.

▪Le champ Nom Alarme nous permet d'insérer le nom du périphérique physique associé à l'alarme mais ça reste est facultatif.

▪Le champ Desc Alarme permet d'insérer la description de l'alarme (facultatif).

▪Les champs Variable A et Variable B nous permet d'insérer les variables qui déclenchent l'alarme, on peut utiliser une ou deux variables.

▪Le champ catégorie permet d'insérer le numéro ou l'étiquette de l'alarme (facultative). Si aucun numéro n'est insérer alors l'alarme est par défaut de catégorie 0.

▪Le champ Delai nous permet d'insérer le délai dont la variable doit resté active pour que l'alarme se déclenche.

▪Le champ Aide permet d'insérer le nom de la page graphique qui s'affiche lorsque la fonction AlarmHelp est appelée (maximum 64 caractères).

C.3.2. Configuration des alarmes analogiques

L'écran de configuration des alarmes analogiques est :

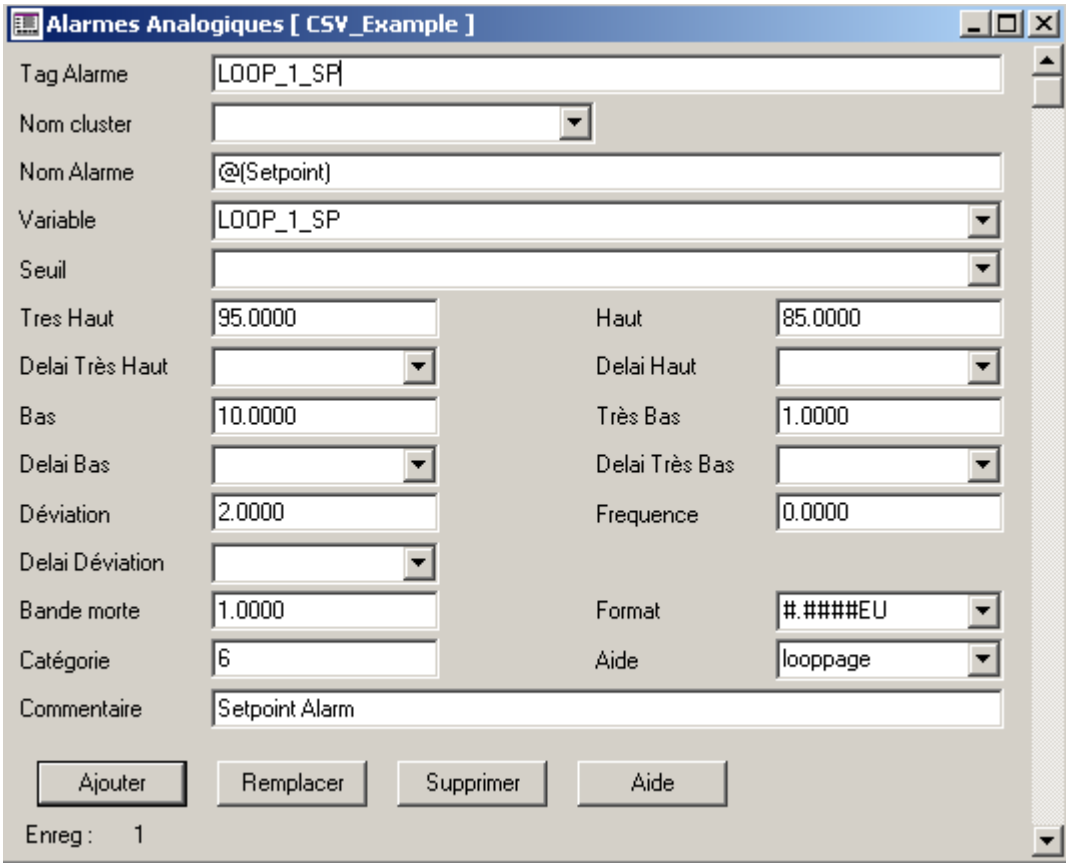


Figure C.6 : Ecran Alarmes analogiques

▪On remarque qu'il y'a des champs commun pour tout type d'alarmes et ils se configurent de la même façon (Tag Alarme, Nom cluster,…) et on peut configurer des alarmes hautes et des alarme très hautes (ou basse ou très basse).

▪Le champ seuil nous permet d'insérer la variable analogique de base qui provoque le déclenchement (facultatif), utilisé pour les alarmes de déviation.la valeur par défaut est 0.

▪Les champs haut, très haut, bas et très bas permettent d'insérer les valeurs qui provoquent le déclenchement, donc la variable analogique est en dessus pour les alarmes hautes et très hautes ou en dessous pour les alarmes basses et très basses.

▪Les champ Delai pour les différentes alarmes nous permet d'insérer la duré d'activation de l'alarme.

▪Le champ Déviation permet d'insérer la valeur utilisée comme condition de déclenchement d'une alarme de déviation (10 caractères maximum). Une alarme de déviation est activée lorsque la valeur du tag reste hors de la plage de déviation (déterminée par le point de consigne) pendant l'intervalle de temps défini par le délai de déviation.

▪Le champ fréquence est facultatif, utiliser selon la façon dont la variable change.

▪Le champ Format décrit le format de présentation de la valeur de la variable lors de l'affichage sur une page graphique.

▪Le champ Bande morte précise la valeur à laquelle doit revenir le tag de variable pour que l'alarme de déviation devienne inactive.

C.3.3. Alarmes horodatées

L'écran de configuration des alarmes horodatées est :

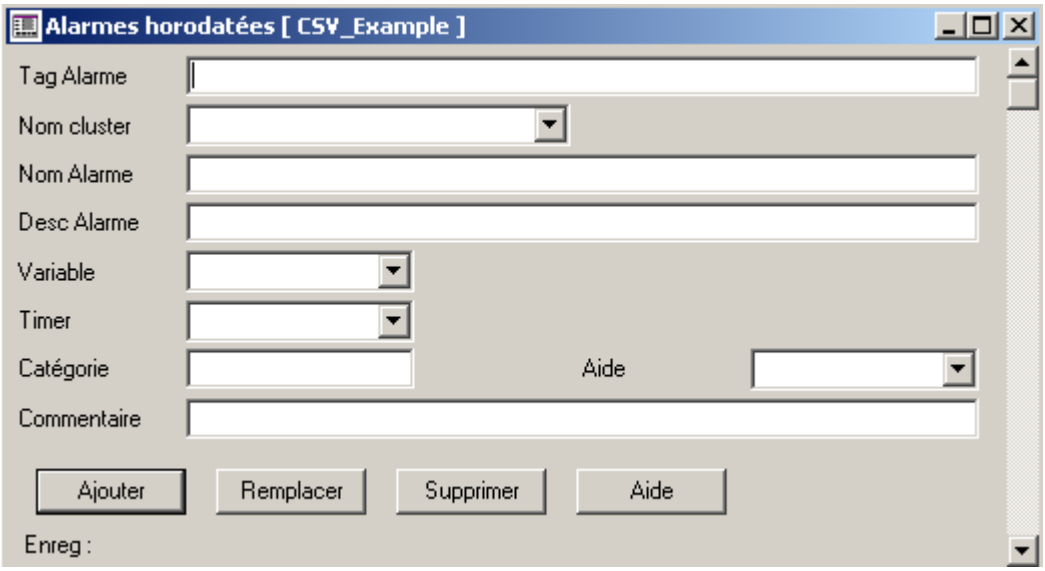


Figure C.7 : Ecran Alarmes horodatées

▪Tout les champs sont décrits précédemment sauf le champ Timer qui permet d'insérer une ou expression Cicode représentant le compteur (ou l'horloge à la milliseconde près) configuré dans le périphérique d'E/S.

C.3.4. Alarmes avancées

L'écran de configuration des alarmes avancées est:

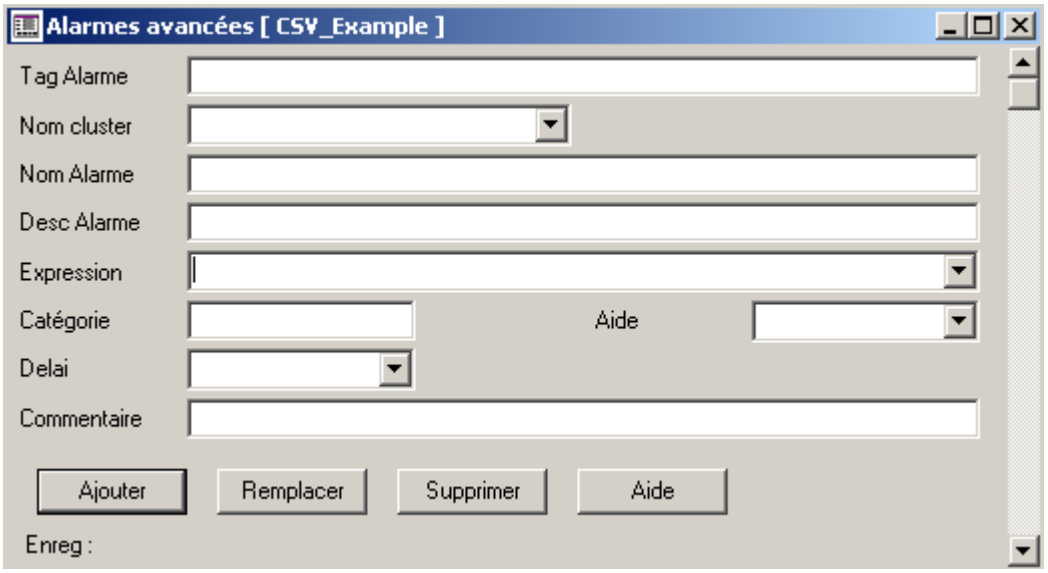


Figure C.8 : Ecran Alarmes avancées

▪Pour le champ expression permet d'insérer l'expression Cicode déclenchant l'alarme (254 caractères maximum). L'alarme est déclenchée lorsque cette expression retourne la valeur VRAI.

C.3.5. Alarmes Multi-Numériques

L'écran de configuration de ce type d'alarme est :

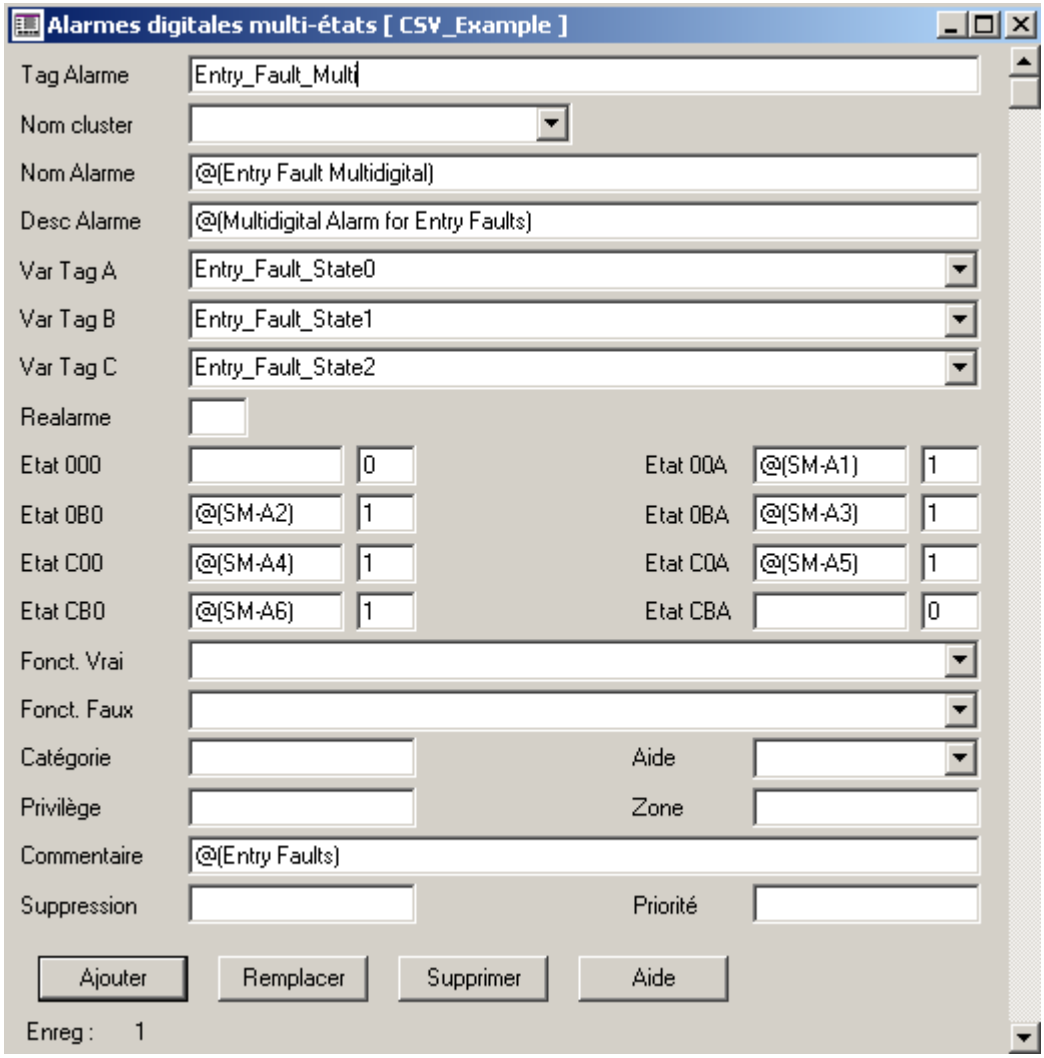


Figure C.9 : Ecran Alarmes digitales multi-états

▪Globalement, cette alarme est déclenchée selon la configuration attribuée à l'état des trois variables.

C.3.6. Alarmes numériques horodatées

L'écran de configuration de ce type d'alarme est le suivant :

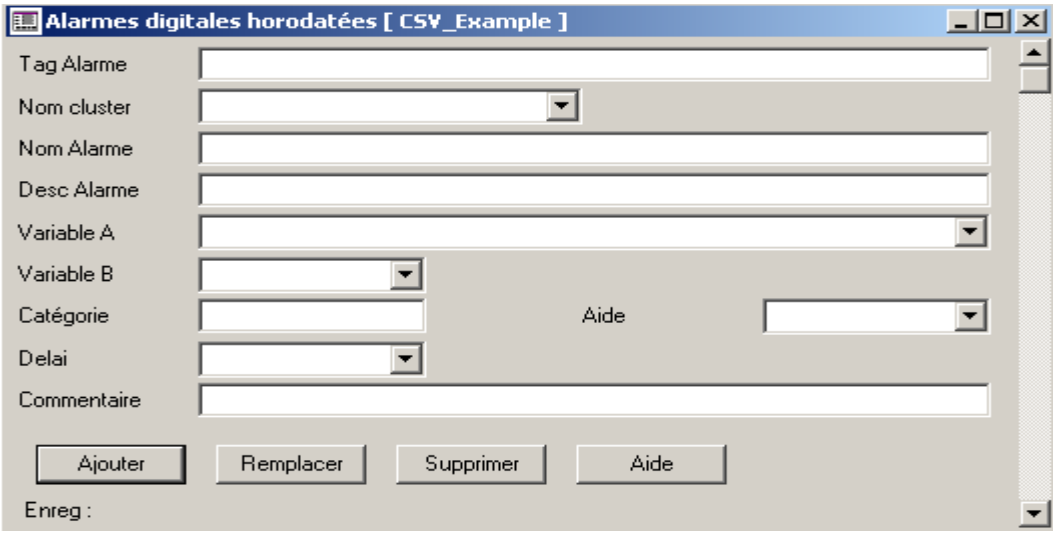


Figure C.10 : Ecran Alarmes numériques horodatées

▪Pour ce type d'alarme, la configuration est la même que l'alarme numérique mais le fonctionnement n'est pas le même.

C.3.7. Alarmes analogiques horodatées

L'écran de configuration de ce type d'alarme est :

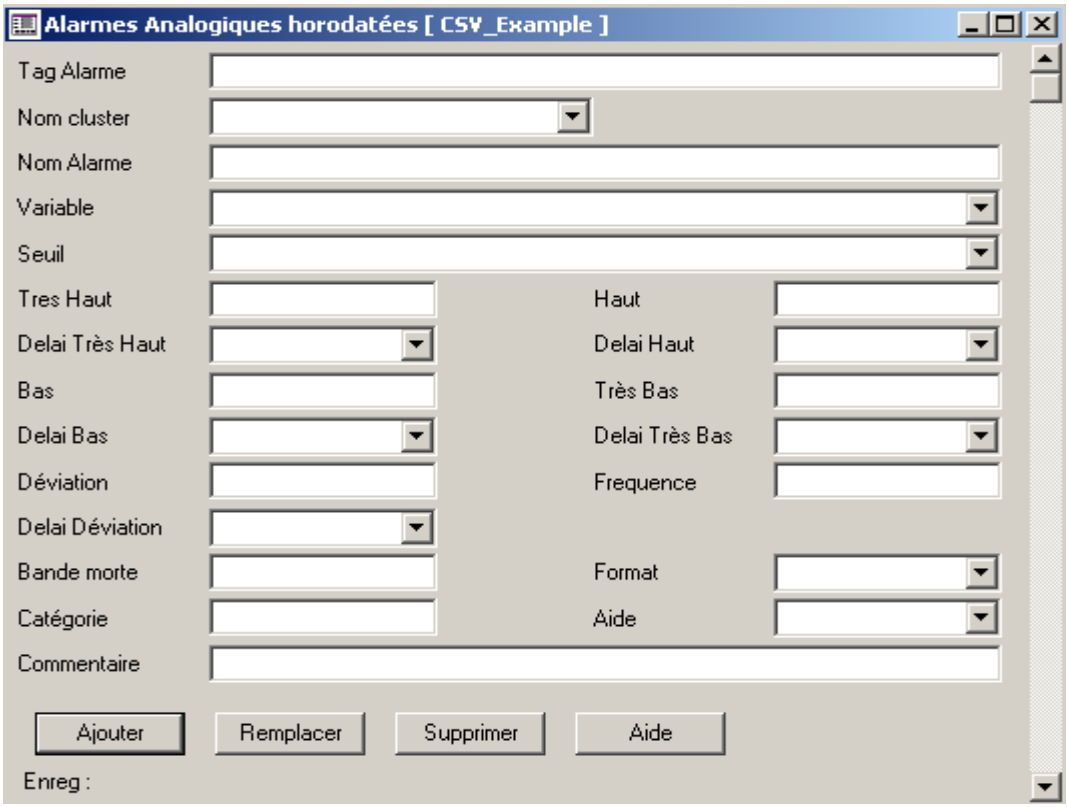


Figure C.11 : Ecran Alarmes horodatées

▪La configuration est la même que celle des alarmes analogiques.

C.3.8. Catégories d'alarmes

L'écran de configuration de l'option catégorie d'alarme est le suivant :

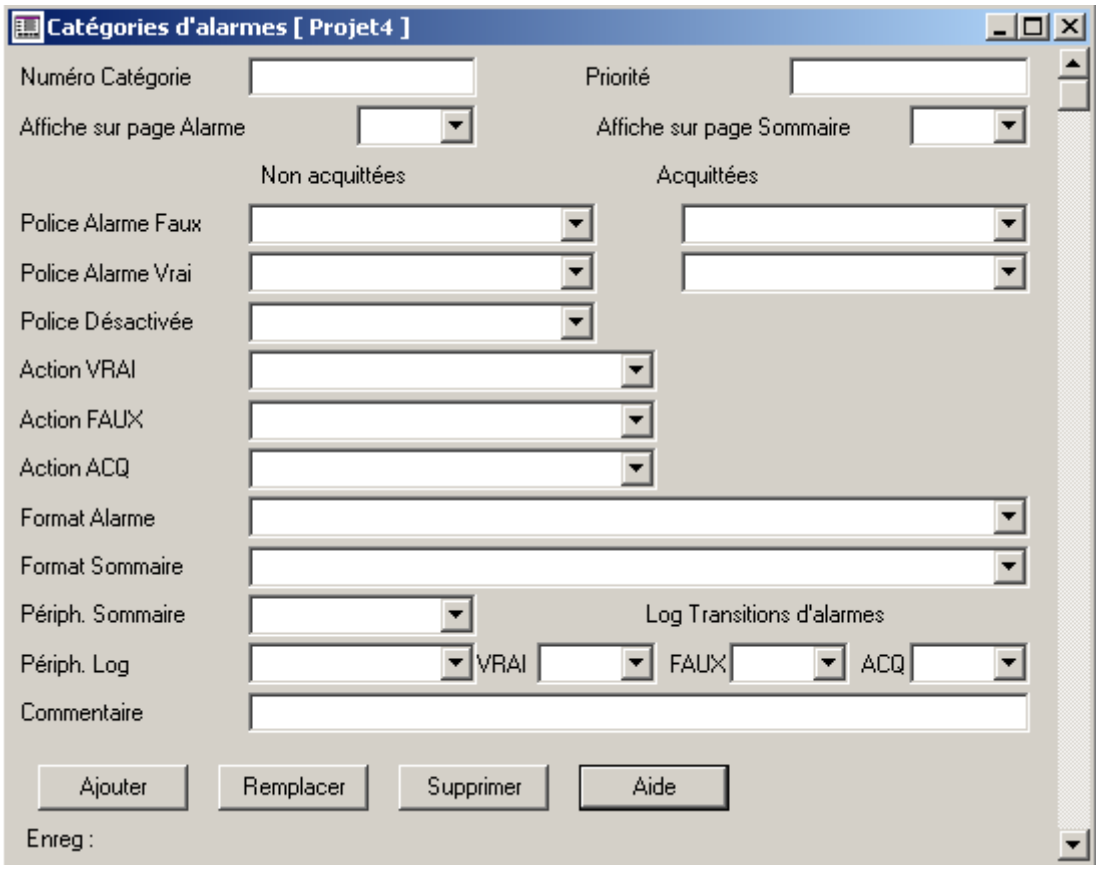


Figure C.12 : Catégories d'alarmes

▪Le champ Numéro catégorie permet de donner un numéro à la catégorie compris entre 0 et 16375.

▪Le champ priorité permet de spécifier la priorité de la catégorie entre 0 et 255, et toutes les alarmes affectées à cette catégorie auront cette priorité, cette dernière, concerne l'ordre d'affichage,…

▪Les deux champs Afficher sur une page d'alarme ou sur page sommaire permette de choisir la page d'affichage et selon ce choix qu'on crie les pages après.

▪Les champs de choix de polices permet de choisir la police d'affichage selon l'état de l'alarme.

▪Les champs Actions permettent de définir les fonctions à exécuter selon l'état de l'alarme.

▪Les champs formats permettent de spécifier le format d'affichage de l'alarme.

▪Les champs périphériques permettent d'insérer les périphériques d'enregistrement des alarmes.

▪Les champs transitions s'ils sont actives permettent d'enregistrer les détails des alarmes selon leur état de transition.

▪Il faut noter que certains champs sont facultatifs.

C.4. Eléments systèmes

C.4.1. Touches clavier

Cette option permet d'affecter des noms (maximum 16 caractéres) au differentes touches du clavier, et on utilise ces derniers pour définir des commandes clavier.

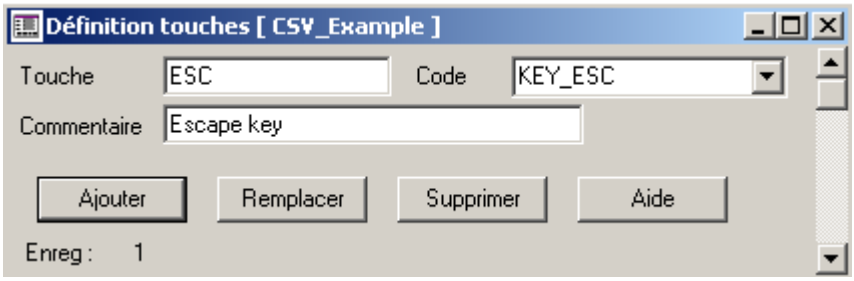


Figure C.12: Ecran Touches Clavier

▪Le code affecté au nom de la touche(maximum16 caractères), est le lien entre le nom et la touche. Il peut être défini à l'aide d'une valeur hexadécimale, ou on peut utiliser l'étiquette Vijeo Citect standard déjà associée à cette touche qui est de la forme (KEY_ Nom de la touche).

C.4.2. Commandes claviers

Cette option permet d'affecter aux touches clavier des instructions à éxecuter.

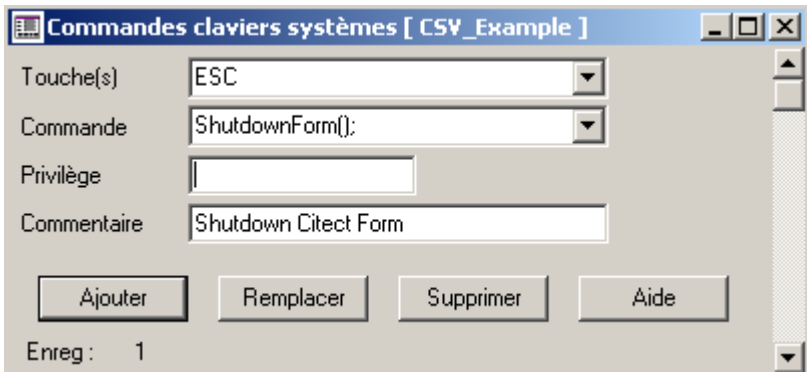


Figure C.13 : Ecran Commandes Claviers

▪Le champ Touche(s) permet de saisir le nom de la touche ou la combinaison de plusieurs touches(maximum 32 caractères).

▪Le champ Commande pemet de choisir une commade parmi la liste proposées par Vijeo Citect qui sera exécuter en appuiyant sur la touche.

▪Le champs privilège permet de definir le privilège d'accès requis d'un opérateur pour émettre la commande.

▪Quelques commandes seront illustrées dans l'annexe#.

C.4.3. Rapports

Cette option permet une présentation personalisée de quelques données du système.

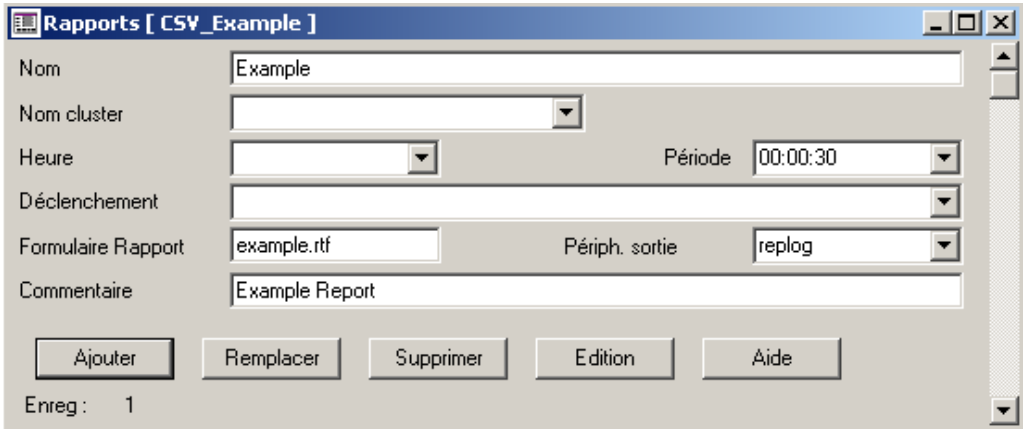


Figure C.14 : Ecran Rapports

▪Dans cet écran, on doit affeceter un nom au rapport(maximum 64 caractères), on spécifie le nom du cluster(qui sera défini après) aux quel il appartient, l'heure de début du rapport, la période ou la durée du rapport, la variable ou l'expression Cicode qui va déclencher le rapport, le format du rapport (par défaut .RTF), et enfin le périphirique aux quel est envoyé le rapport.

C.4.4. Evénements

Cette option permet de configurer des commandes à éxecuter suite à des déclenchement provoqués par des expression Cicode ou par des variables.

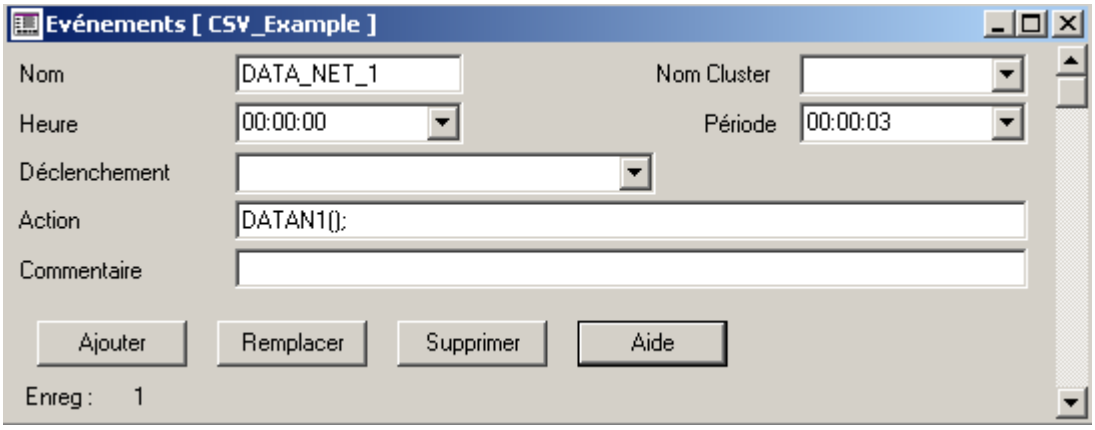


Figure C.15 : Ecran Evénements

▪On a présque les meme configurations que les écran rapports et on en plus le champ action

qui permet de définir la commande à éxecuter suite à l'évenement.

C.4.5. Accumulateurs

Cette option permet de configurer des variables qui seront associées en continu à des données du système superviseur.

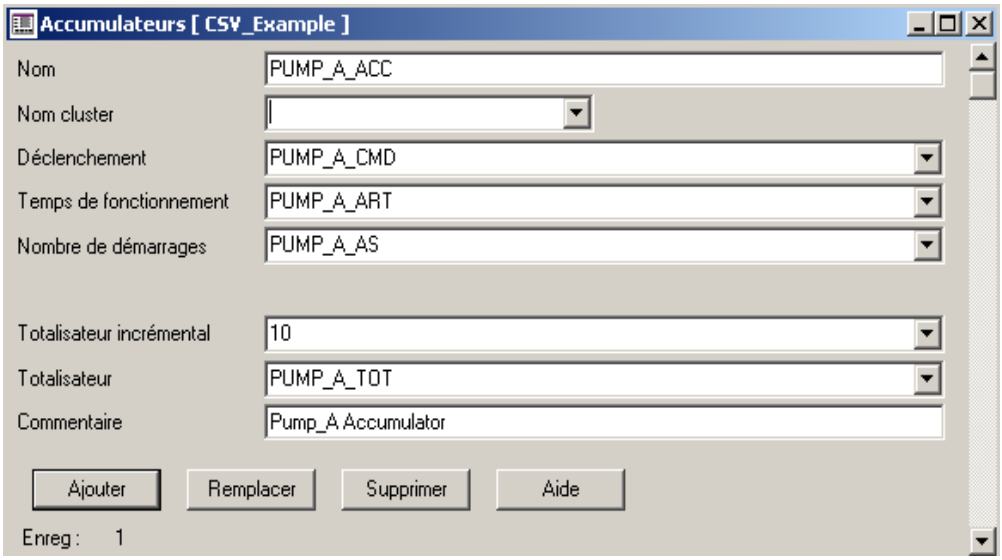


Figure C.16 : Ecran Accumulateurs

▪Pour la configuration, on affecte un nom pour l'accumulateur et on spécifie le cluster de travail.

▪On specifie la variable déclenchant l'accumulateur, la variable ou est affecter le temps de fonctionnement(en seconde) et aussi la variable qui caracterise le nombre de démarrage de l'accumulateur.

▪Le champ totaliseur incrémental caracterise l'expression à ajouter à la variable affecter dans le champ totliseur à chaque déclenchement et on peut ajouter un commentaire.

C.4.6. Péripheriques

Cette option permet de configurer des éléments pouvant transférer des données de haut niveau à d'autres éléments tels que des fichiers RTF, des fichiers ASCII et des imprimantes.

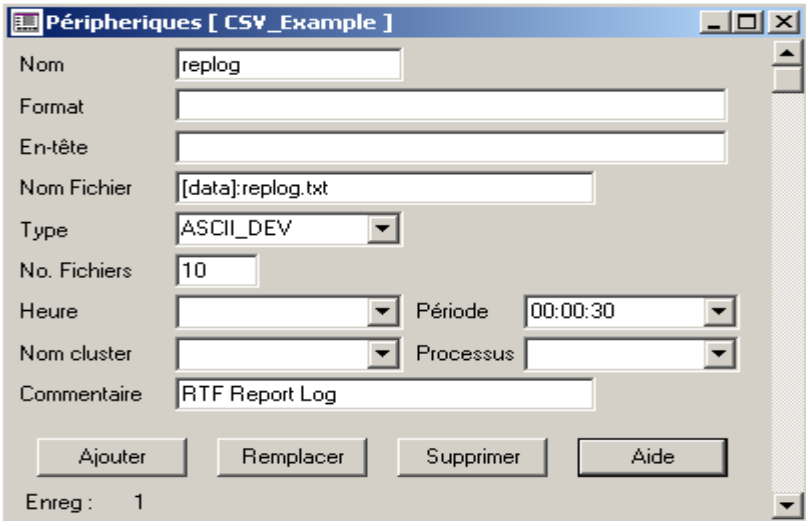


Figure C.17 : Ecran Péripheriques

▪On procède pour la configuration de la même manière que précedement, à léxception qu'on doit spécifier le format de la donnée , l'en-tête et le processus qui differe d'un péripheriques à un autre..

C.4.7. Utilisateurs

Cette opion est utilisée pour specifier les profils d'utilisateurs pour limiter et autoriser l'accès au système superviseur.

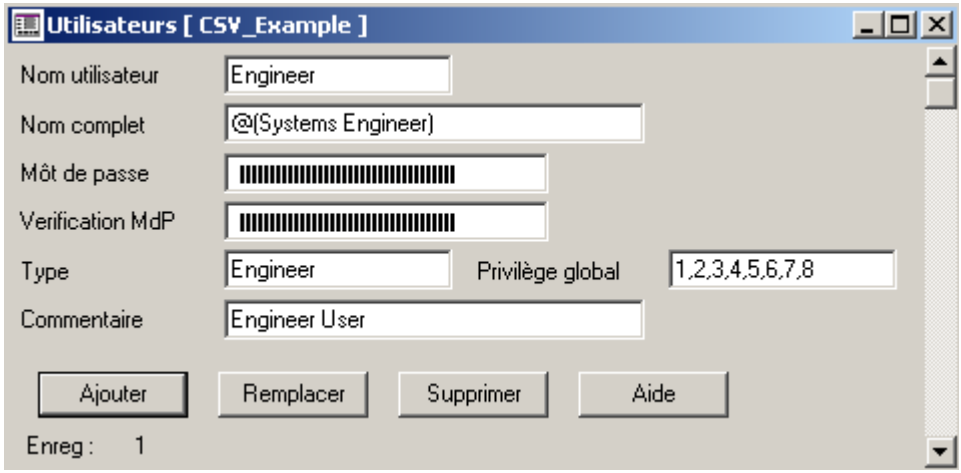


Figure C.18 : Ecran Utilisateurs

▪Pour configurer cet écran, on introduit un nom d'utilisateur(maximum 16 caractères),et un mot de passe.

▪Le champ nom complet est utiliser comme cmmentaire pour l'affichage des journaux d'alarmes et de commandes,la case type spécifie le type de l'utilisateur et on introduit aussi les privilège qui lui sera accordés.

C.4.8. Groupes

Cette option permet de spécifier des groupes de zones utilisés pour simplifier la gestion des profils d'utilisateurs.

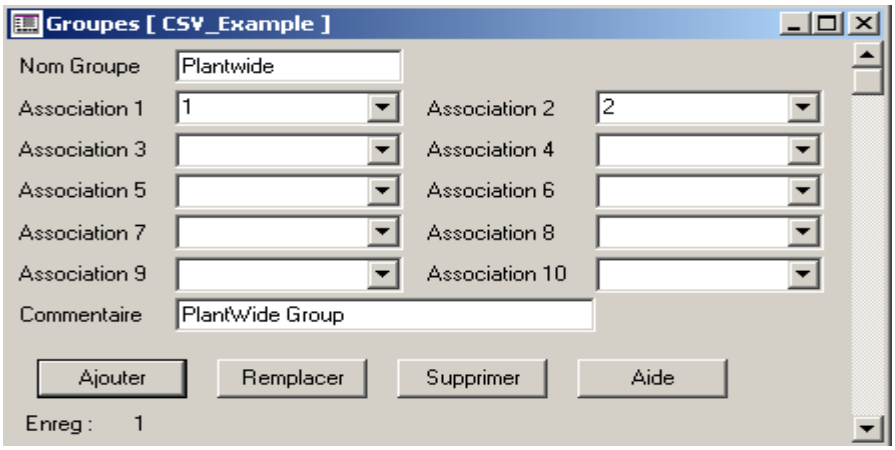


Figure C.19 : Ecran Groupes

▪A chaque groupe, on doit associer un nom de groupe(maximum 16 caractères), une fois définie, il peut etre utilisé comme une entité.

▪Les associations de 1 à 10 sont des listes d'entités (maximum 16 caractères). Une association peut etre un numéro, u nom ou un autre groupe.

C.4.9. Macros

Cette option est utilisée pour définir des substitutions couramment utilisées dans l'ensemble du système pour remplacer des commandes et des expressions.

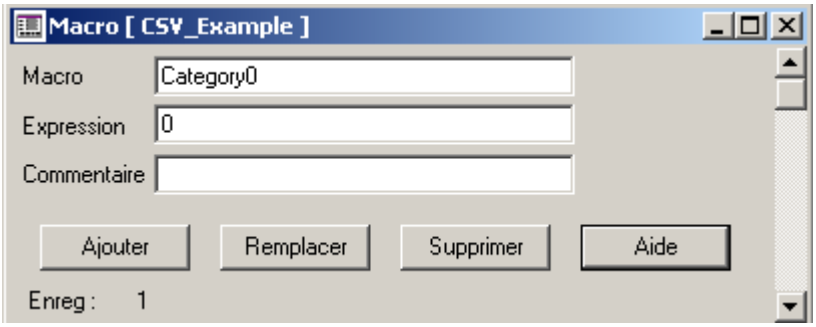


Figure C.20 : Ecran Macros

▪Le champ Macro permet de saisir le nom du macro(maximum 64 caractères) qui sera remplacer par l'expression saisie dans le champ expression (254 caractères au maximum).

C.4.10. Polices

Cette option permet de choisir la police utilisée pour afficher les alarmes et les objets.

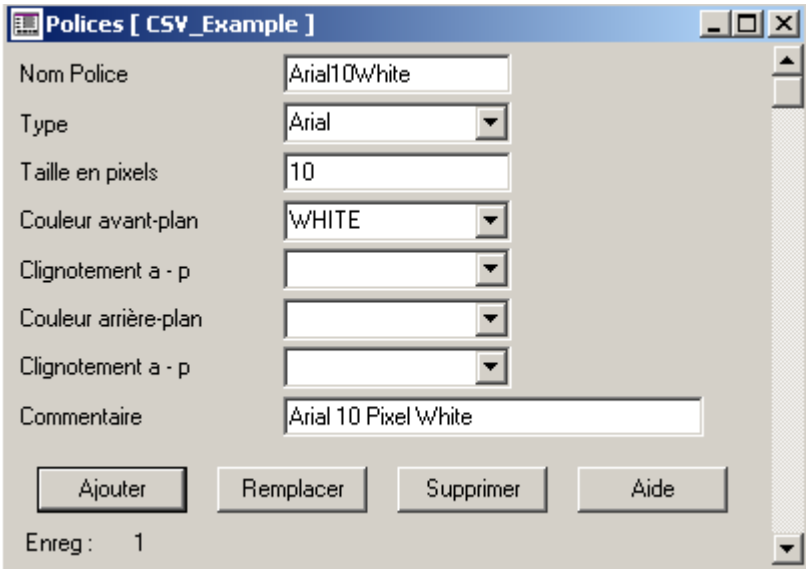


Figure C.21 : Ecran Polices

▪Cet écran permet de choisir le type de police, on peut lui affecter un nom(maximum 16 caractères), la taille en pixels et aussi les couleurs avant –plan et arrière plant.

C.4.11. Paramètres

Cette option permet de spécifier les paramètres opérationnels intégrés pour effectuer un réglage précis du système superviseur.

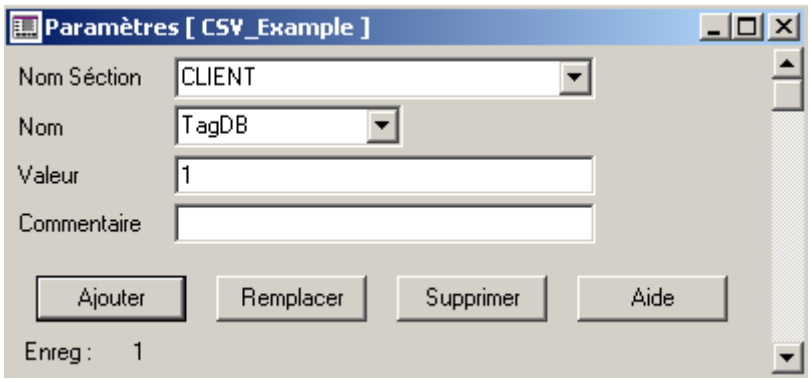


Figure C.22 : Ecran Paramètres

▪On doit spécifier le nom de la section ou se trouve le paramètre(16 caractères), le nom du paramètre dont on veut changer la valeur(maximum 16 caractères) et aussi la valeur qu'on veut attribuer au paramètre(maximum 254 caractères).

C.4.12. Projets Inclus

Cette option permet d'ajouter au projet en cours des projets prédéfinis avec enregistrements de base de données automatiquement inclus dans les projets utilisateurs.

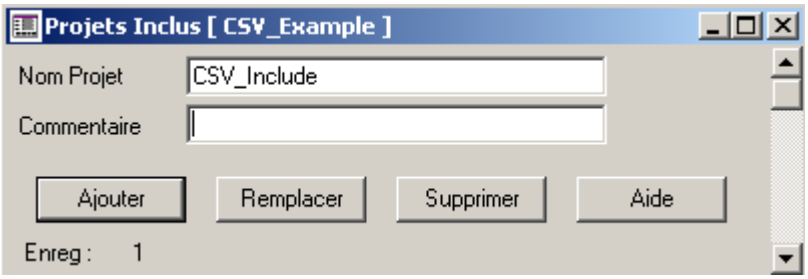


Figure C.23 : Ecran Pojets Inclus

▪Il suffit d'inscrire le nom du projet et de cliquer sur le bouton ajouter.

C.5. Utilisation de l'assistant de configuration du poste

Au lancement de l'assistant, la figure suivante apparait :

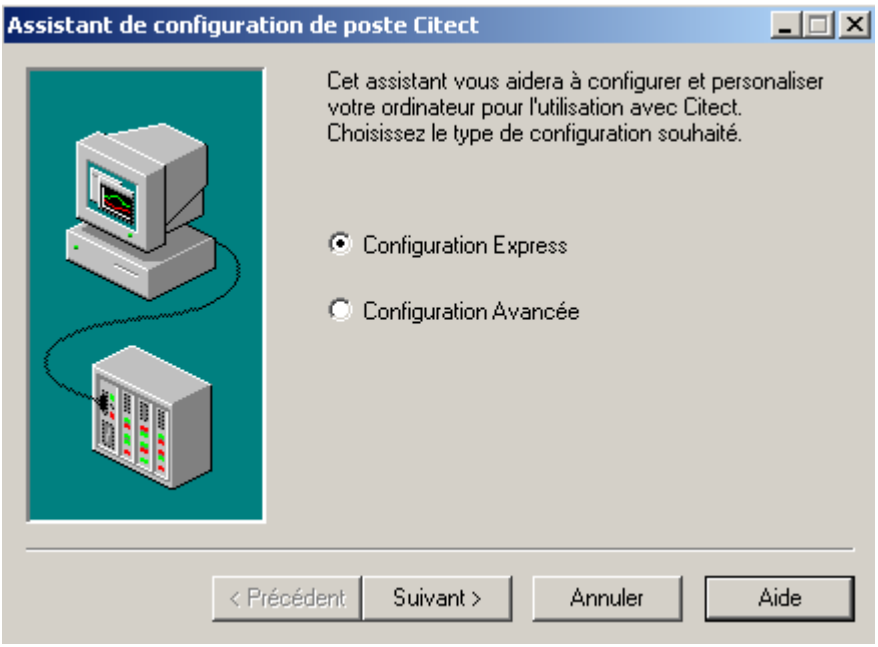


Figure C.24 : Assistant de configuration du poste_1

▪Comme le montre la figure, on a un choix entre la configuration express ou la configuration avancée, en faite, la configuration avancée est plus large (configuration des éléments de sécurité,…) et que la première est inclus dans la seconde ; donc on choisi la configuration avancée, en cliquant sur le bouton suivant, la figure qui suit va apparaitre :

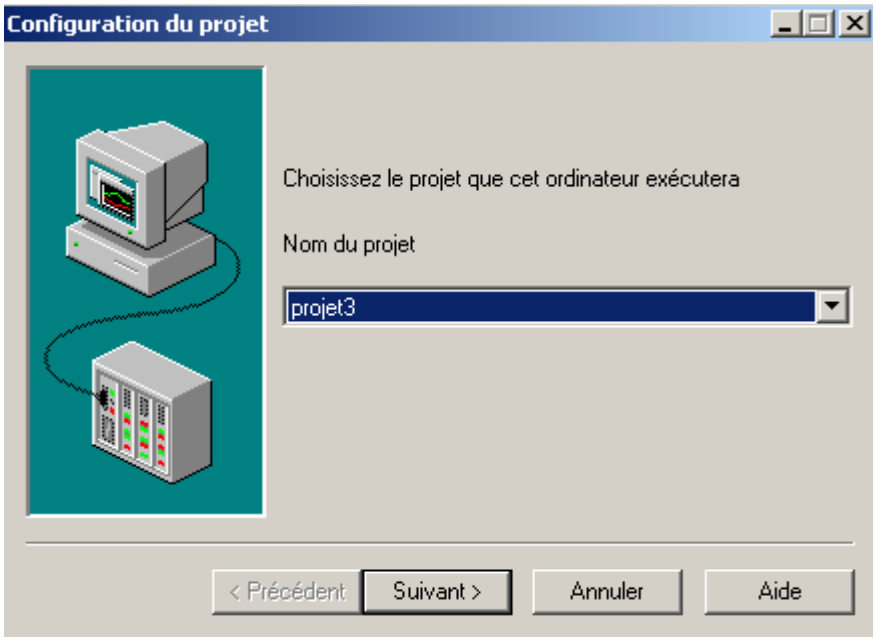


Figure C.25 : Assistant de configuration du poste_2

▪Cet écran nous permet de choisir le projet qu'on va executer sur l'ordinateur, on trouve la liste de tous les projets crées mais seulement ceux qui ont été compilé, donc pour choisir un projet précis, il faut le compié et eventuellement corriger toute les erreurs s'il y'on a, en continue en cliquant sur le bouton suivant et la figure suivante apparait :

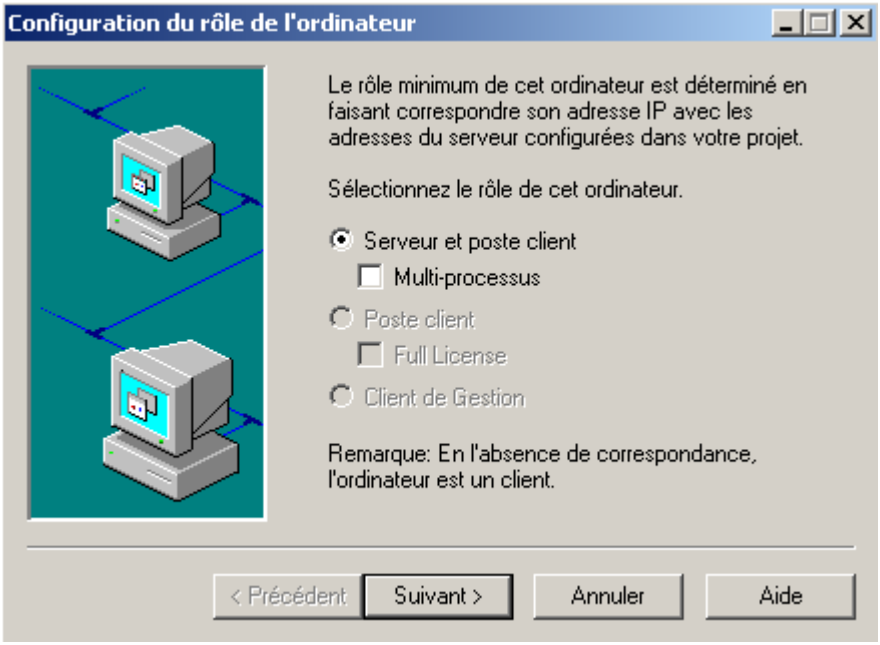


Figure C.26 :Assistant de configuration du poste_3

▪Cette fois, on configure le role de l'ordinateur ou le projet sera executer, on a trois options :

- Serveur et poste client : l'ordinateur sera un Serveur d'E/S et un Client de visualisation Vijeo Citect autonome ou en réseau. Cette option est désactivée si aucun

composant de serveur n'est affecté à l'ordinateur. La sélection de cette option coche la case Multiprocessus, cette dernière est

Sélectionnée pour séparer le client et les composants de serveur en processus individuels. Cette option peut être utilisée pour distribuer les composants sur plusieurs UC. Si la case Multiprocessus n'est pas sélectionnée, Vijeo Citect exécute le client et tous les composants de serveurs dans un processus unique.

- Poste client : L'ordinateur est uniquement utilisé comme client de visualisation. Cette option est désactivée si un composant de serveur a été affecté à l'ordinateur ;
- Client de gestion : L'ordinateur est uniquement utilisé comme client de gestion. Cette option en lecture seule est désactivée si un composant de serveur a été affecté à l'ordinateur.

▪Donc, on fait notre choix, selon le besoin. Dans notre cas,on choisi la première option,on continue et la figure est la suivante :

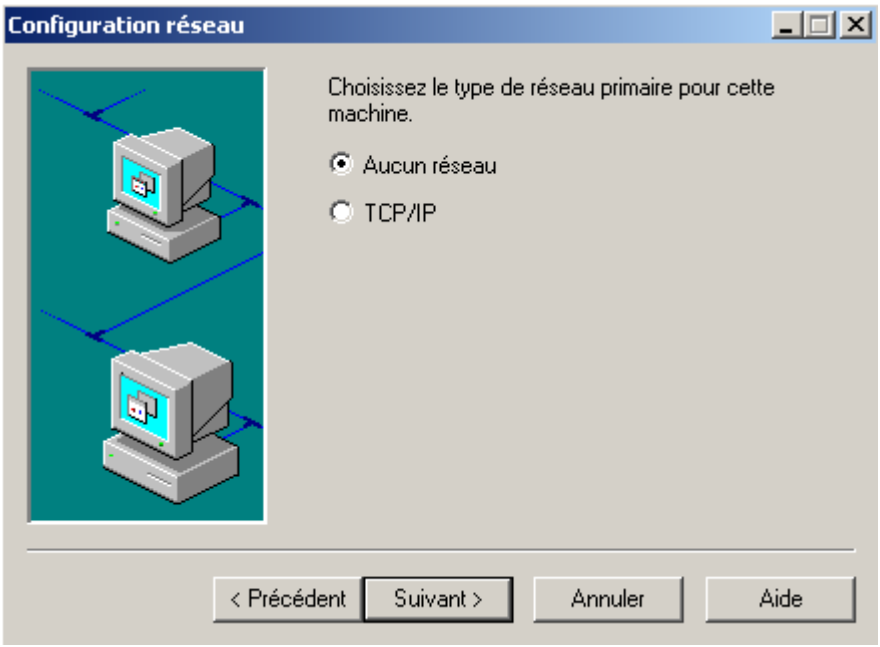


Figure C.27 : Assistant de configuration du poste_4

▪Cet écran nous permet de configurer le réseau, on a deux choix :

- Aucun réseau : ordinateur autonome ;
- TCP/IP : ordinateur appartient à un réseau de communication.

▪Dans le cas ou on choisi la première option, l'ordinateur est forcement un serveur et un client de visualisation en meme temps.

▪On a choisi la première option, on continue et la figure sera :

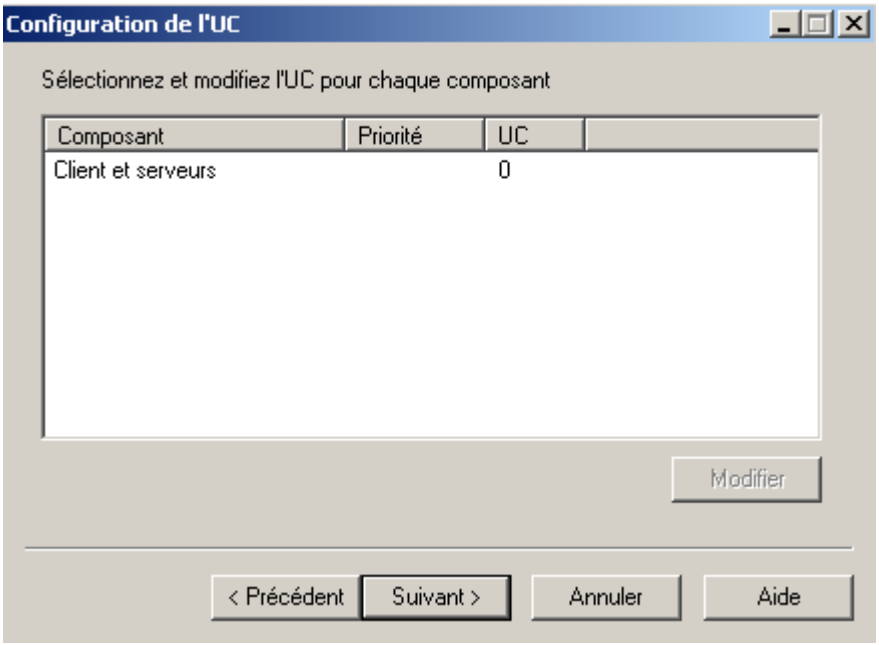


Figure C.28 : Assistant de configuration du poste_5

▪Cet écran nous permet d'affecter un client et des composants de serveurs à des processeurs spécifiques dans une machine multiprocesseurs.

▪Cette page contient le nom complet de chaque composant, y compris le cluster auquel il appartient, la priorité et l'affectation de l'UC. Si l'option Multiprocessus n'a pas été sélectionnée sur la page Configuration du rôle de l'ordinateur, celle-ci contiendra une seule entrée, à savoir Client ou Client et serveurs. Si l'option Multiprocessus a été sélectionnée, on peut affecter des UC spécifiques au client, au serveur d'E/S, au serveur d'alarmes, au serveur de tendances et aux serveurs de rapports.

▪On continue et on aura la figure :

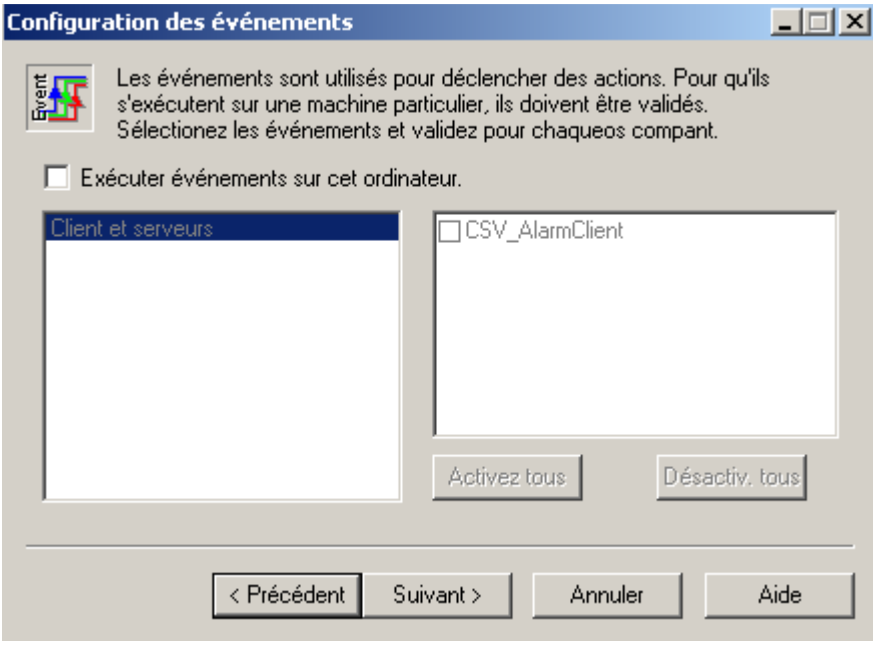


Figure C.29 : Assistant de configuration du poste_6

▪Cet écran permet d'inserer les événements permettent de déclencher des actions, telles qu'une commande ou un jeu de commandes. Ils permettent par exemple d'indiquer l'achèvement d'un processus à un opérateur ou de déclencher l'exécution d'une série d'instructions lorsqu'un processus franchit une étape spécifique, on insére ces événements et on coche la case exécuter événements sur cet ordinateur.

▪On va pas utiliser cette option, on continue et on aura la figure :

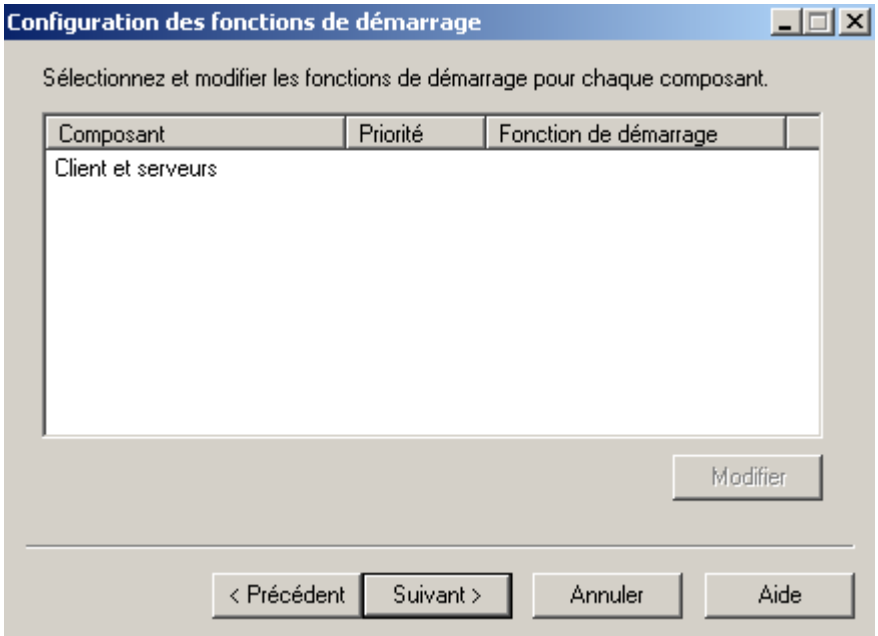


Figure C.30 : Assistant de configuration du poste_7

▪Cet écran nous permet de définir le Cicode de démarrage exécuté par chaque processus Vijeo Citect.

▪Un clic sur le bouton suivant, on aura :

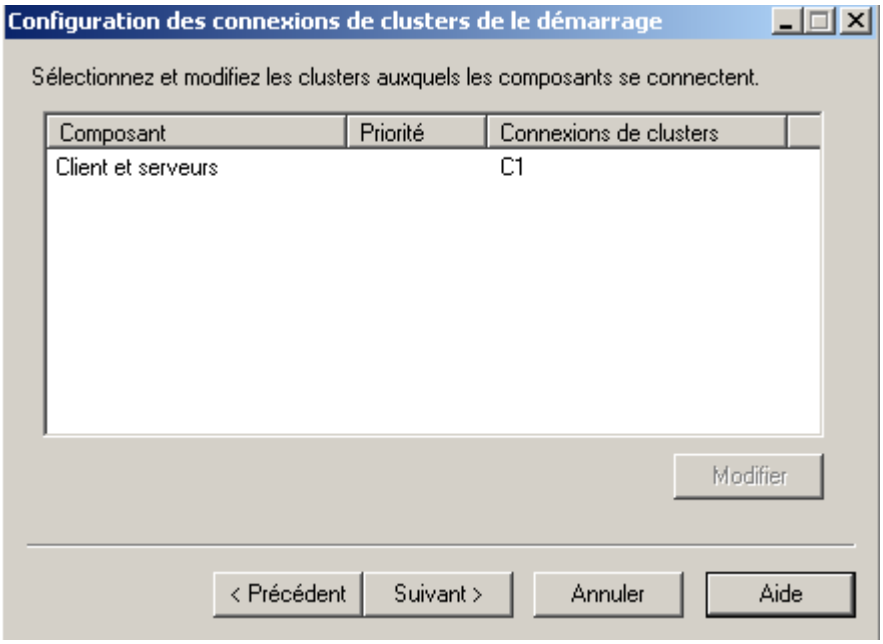


Figure C.31 : Assistant de configuration du poste_8

▪Cet écran nous permet de spécifier les clusters auxquels chaque composant se connecte au démarrage. Ceci permet de contrôler les flux de données qu'un composant peut voir dans le système.

▪On continue, on aura la figure :

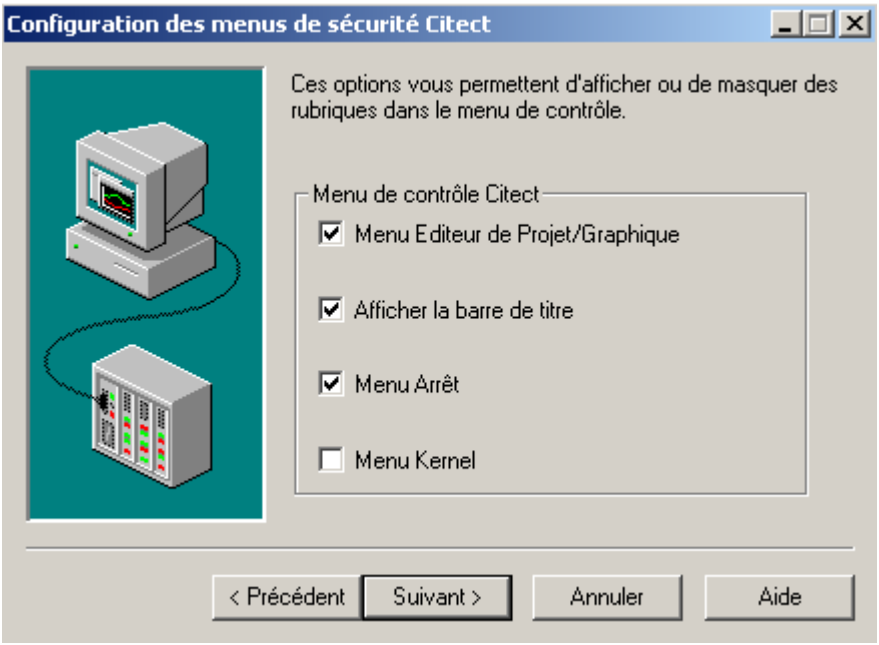


Figure C.32 : Assistant de configuration du poste_9

▪Cet écran permet de configurer l'accés des opérateurs au système.

▪On continue, on aura :

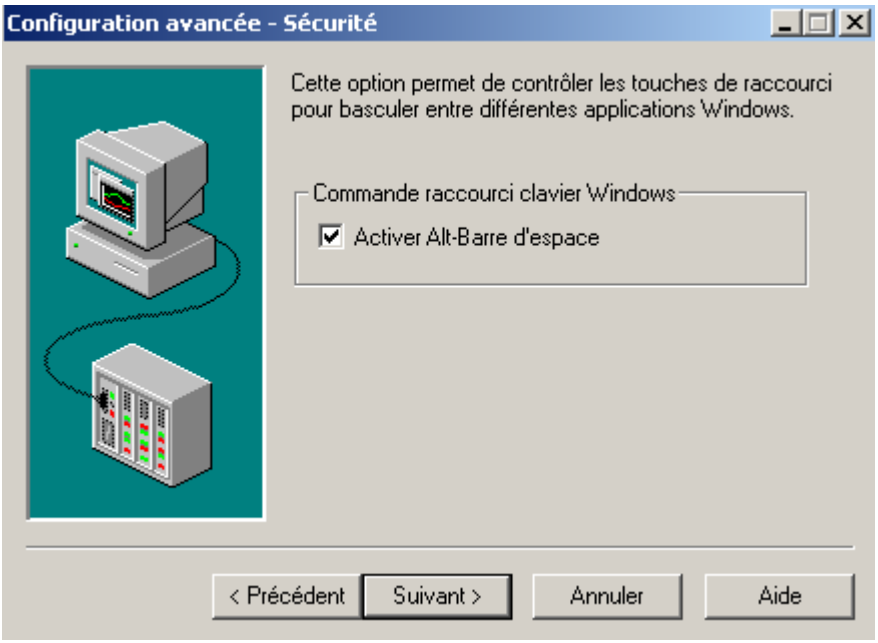


Figure C.33 : Assistant de configuration du poste_10

▪ Windows possède un jeu de commandes de commutation de tâches pouvant être pris en charge par Vijeo Citect en phase d'exécution. Cette option permet d'activer ou de désactiver la commande Windows Alt-Espace en face d'exécution. Alt-Espace permet d'accéder au menu système Windows même si la barre de titre a été désactivée.

▪On continue, on aura :

Figure C.34 : Assistant de configuration du poste_11

▪ Certaines fonctionnalités Windows standard peuvent nuire à l'opération sécurisée du système Vijeo Citect. On utilise cet écran (divers) pour désactiver ces fonctionnalités.

▪On continue, on aura :

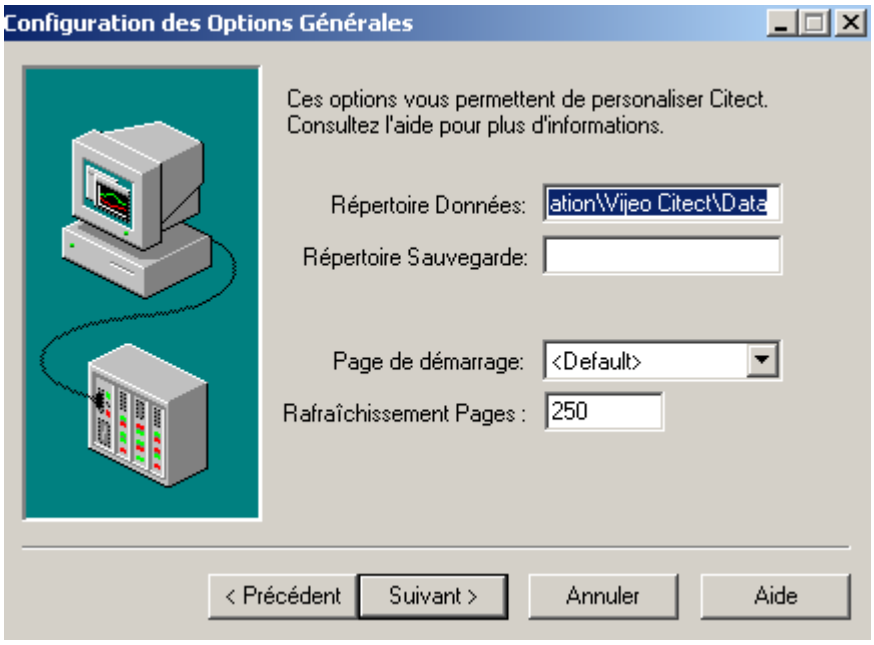


Figure C.35 : Assistant de configuration du poste_12

▪Cet écran nous permet, de configurer certaines options générales :

- Répertoire de données : répertoire où se trouvent les fichiers de données Vijeo Citect.

Les fichiers de données Vijeo Citect sont générés en phase d'exécution : fichiers de tendances, etc ;

- Répertoire de sauvegarde : répertoire de sauvegarde utilisé si une base de données est inaccessible en phase d'exécution (erreur disque ou perte de fichiers) ;
- Page de démarrage : Nom de la Page graphique à afficher au démarrage de Vijeo Citect ;
- Rafraichissement des pages : délai (en millisecondes) entre la mise à jour d'une page graphique et le début du cycle de communications suivant. Il s'agit de la fréquence de mise à jour des pages graphiques. Lorsqu'une page est mise à jour, toutes les données pertinentes (tags, etc. figurant sur la page graphique) sont interrogées pour déterminer si les conditions ont changé sur le terrain. Ce paramètre est ignoré si une fréquence d'analyse a été spécifiée dans les propriétés de page . Une valeur de 250 (valeur par défaut) indique que Vijeo Citect essaie de mettre la page à jour toutes les 250 ms. Toutefois, si Vijeo Citect ne peut lire toutes les données d'un périphérique d'E/S dans le temps imparti (250 ms), il les traitera à une fréquence inférieure. Par exemple, si la lecture des données d'un périphérique d'E/S prend 800 ms, Vijeo Citect les traitera toutes les 800 ms.

▪On continue et on aura la fenetre :

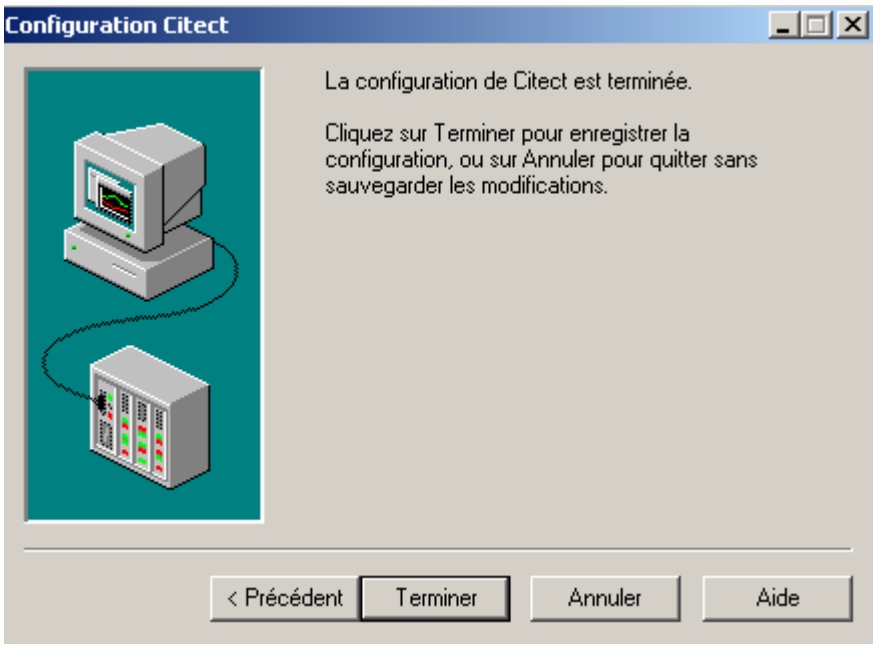


Figure C.36 : Assistant de configuration du poste_13

▪Cet écran permet de valider les configurations faites précedement et ça en cliquant sur le bouton terminer et si ainsi que la configuration du poste est finie.

C.6. Création des genies et des super genies

C.6.1. Création des génies

Les génies sont des objets configurables crés pour faciliter la duplication, par exemple, si on a plusieurs pompes et on doit visualiser leurs états, il suffit de créer un génie et on lui fait appel et lui faire correspondre les variables de chaque pompes.

▪Pour la création d'un super génie, on utilise l'éditeur graphique. Après l'avoir lancé et on cliquant sur nouveau, on choisie la case Génie (Figure **III.33**) et on dessine l'obj qu'on veut.

▪Pour l'affectation des variable, on assossié au génie des étiquettes et au moment de l'appel du génie, on fait correspondre chaque étiquette à une variable.

▪Les étiquettes sont de la formes : % nom_étiquette%, on peut créer autant d'étiquette qu'on veut.

▪Au moment de la sauvegarde du génie, on peut l'ajouter à la bibliothéque génie déjà éxistantes et on peut l'utiliser autant de fois qu'on aura besoin.

❖ **Exemple de création d'un génie**

▪Supposons qu'on a à créer une pompe et au moment de l'affichage, on veut visuliser l'état marche ou arret.

▪On prend deux pompes de la biblithèque des objet avec deux couleur différente comme le montre la figure suivante :

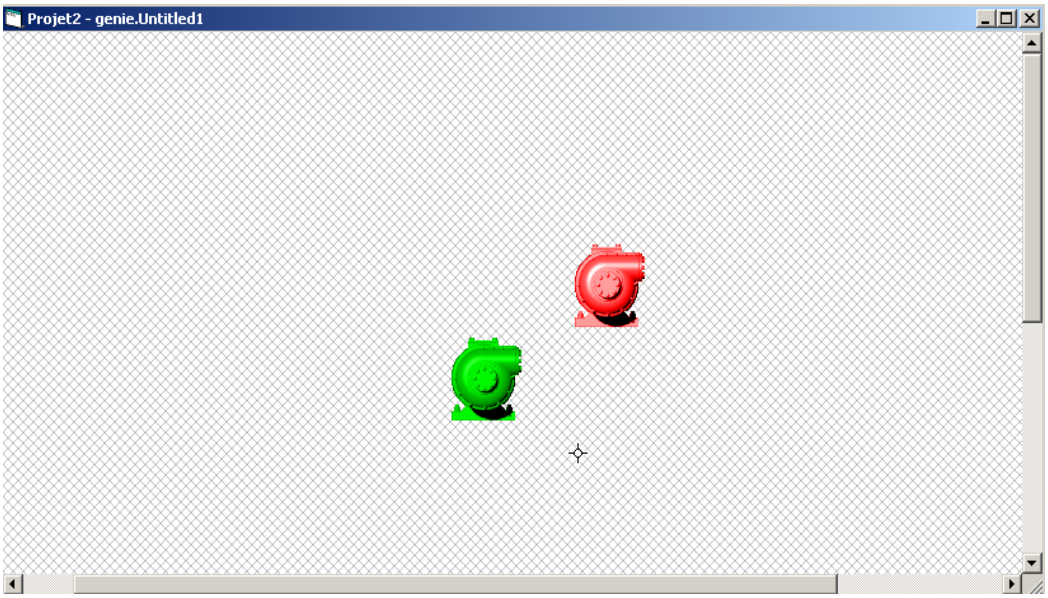


Figure C.37 : Choix des objets

▪La coouleur rouge correspond à l'état arret et la couleur verte à l'état marche.

▪Pour l'affecatation des étiquettes, double clic sur la pompe de couleur rouge et dans l'onglet visibilité, on insère l'étiquette comme le montre la figure suivante :

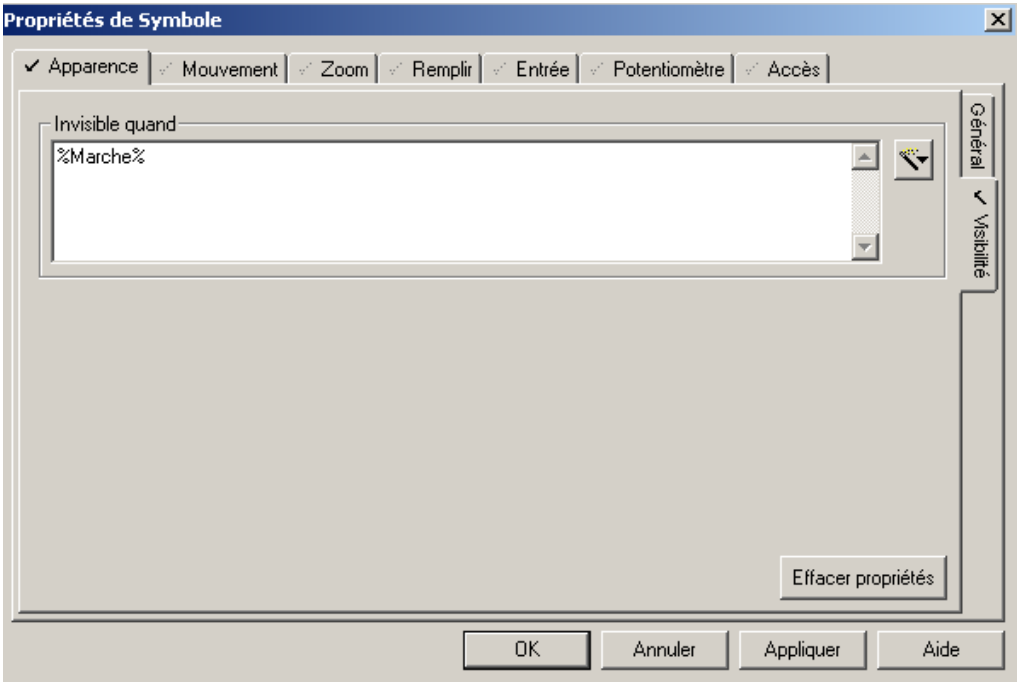


Figure C.38 : Afectation d'étiquette

▪Comme la condition de visiblité est inversé, alors quand l'étiquette Marche soit vrai , la ompe de couleur rouge ne va pas apparaitre. Pour la pompe de couleur verte, on fait l'inverse donc son étiquette sera NOT(%Marche%) dans le méme onglet (visibilité).

▪On superpose les deux pompes et on sauvegarde en donnant un nom au génie comme le montre la figure suivante : on a choisi le nom Pompe

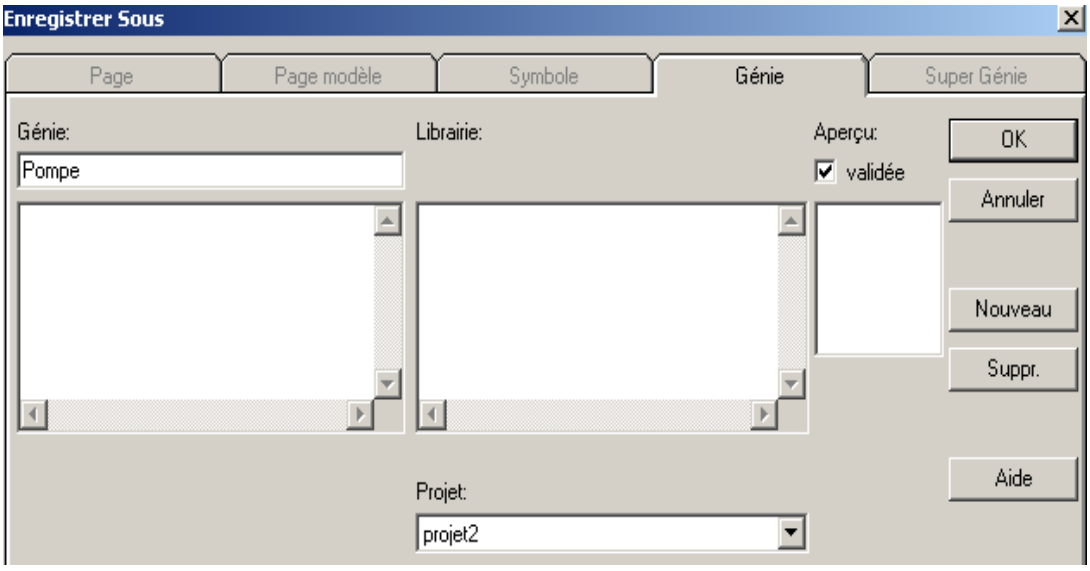


Figure C.39 : Sauvegarde du génie

▪Il faut noter qu'on peut le sauvegarder dans l'importe quel projet qui se trouve dans la liste, en cliquant sur le bouton OK, on aura la figure :

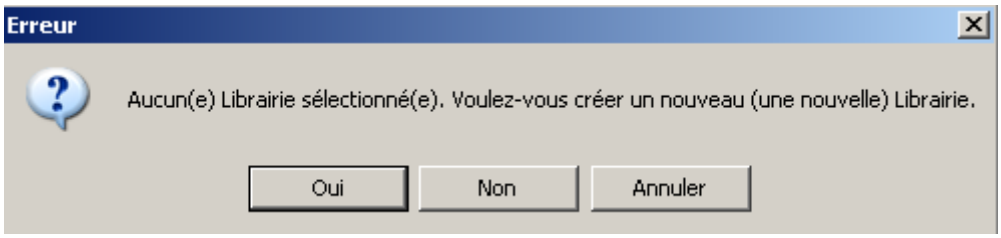


Figure C.40 : Création d'une nouvelle librairie_1

▪Comme le montre la figure, on peut créer une nouvelle librairie contenant ce génie, on clique sur oui, on aura :

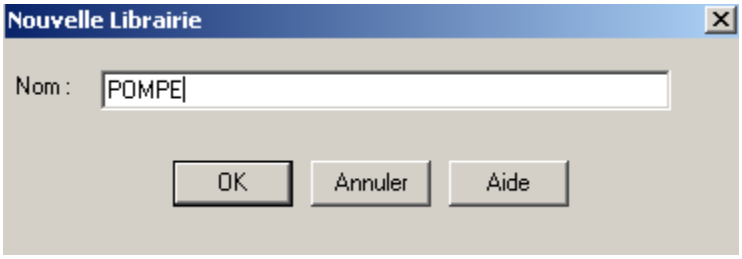


Figure C.41 : Création d'une nouvelle librairie_2

▪On a choisi le nom POMPE pour le nom de la librairie et on sauvegarde.

▪Dans la page principale, on utilise l'icône génie (Tableau **III.1**) pour insérer un nouveau génie, on insère celui qu'on a créer (on cherche le nom POMPE) et au moment de l'insértion on aura la figure suivant :

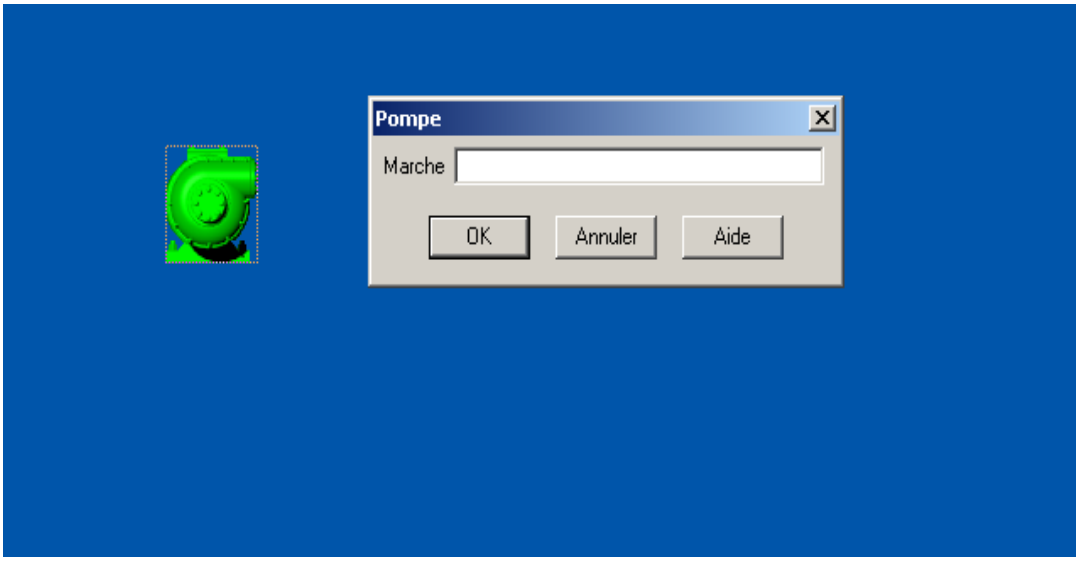


Figure C.42 : Insértion du génie

▪Comme le montre la figure, on doit assossiè à l'étiquette Marche une variable de telle sorte que cette dernière quand elle est vrai(reçoit 1) la pompe verte apparait et quan elle est fausse(reçoit 0), la pompe rouge apparait.

C.6.2. Création d'un super génie

Les supers génies comme on a l'a mentioné dans le rapport sont des génies pouvant transmettre des données spécifiques à un périphérique en phase d'exécution, par exemple pour le génie crée précédemment, on peut créer un bouton marche pour envoyer une commande de faire marcher la pompe, chose qu'on ne peut pas faire avec les génies.

▪Pour créer un super génie, soit on utilise la case super génie (Figure **III.33**) ou bien, on crée une page normale et au moment de la sauvegarde, le nom de la page doit être précédé par le symbole « ! » et Vijeo Citect le considéra automatiquement comme super génie.

▪En ne peut pas faire appel directement au super génie comme les génies mais cela se fait en utilisant les génies et en cliquant sur l'objet crée, donc les génies et les super génies sont utilisés ensemble.

▪Contrairement aux génies, le nombre d'étiquette que peut supporter un super génie ne doit pas dépasser huit. Ces étiquettes sont de la forme : ? type numéro ?, le type indique le type de la variable qui va être associé à l'étiquette (Int, digital,…), le numéro indique le numéro de l'étiquette(de 1 jusqu'à 8).

❖ **Exemple de création d'un super génie**

▪Pour le génie créer précédemment, on va créer un bouton d'envoi du signal marche et un autre pour envoyer le signal arrêter.

▪On va créer une page normal avec les deux boutons comme le montre la figure suivante :

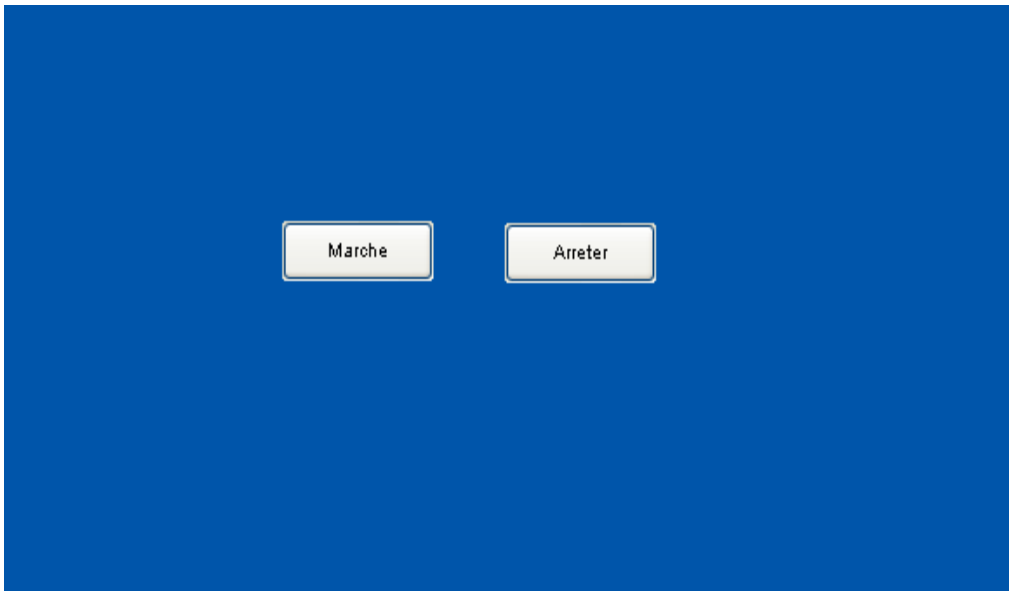


Figure C.43 : Créer un super génie_1

▪Pour que le clic sur l'un de ces deux boutons permet d'envoyer un signal, dans les propriétés des boutons, on utilise l'onglet Entrée.

▪Comme ces des signaux tout ou rien, le type est digital, on affecte le numéro 1 pour le bouton marche et le numéro 2 pour le bouton arrêter comme le montre les figures suivantes :

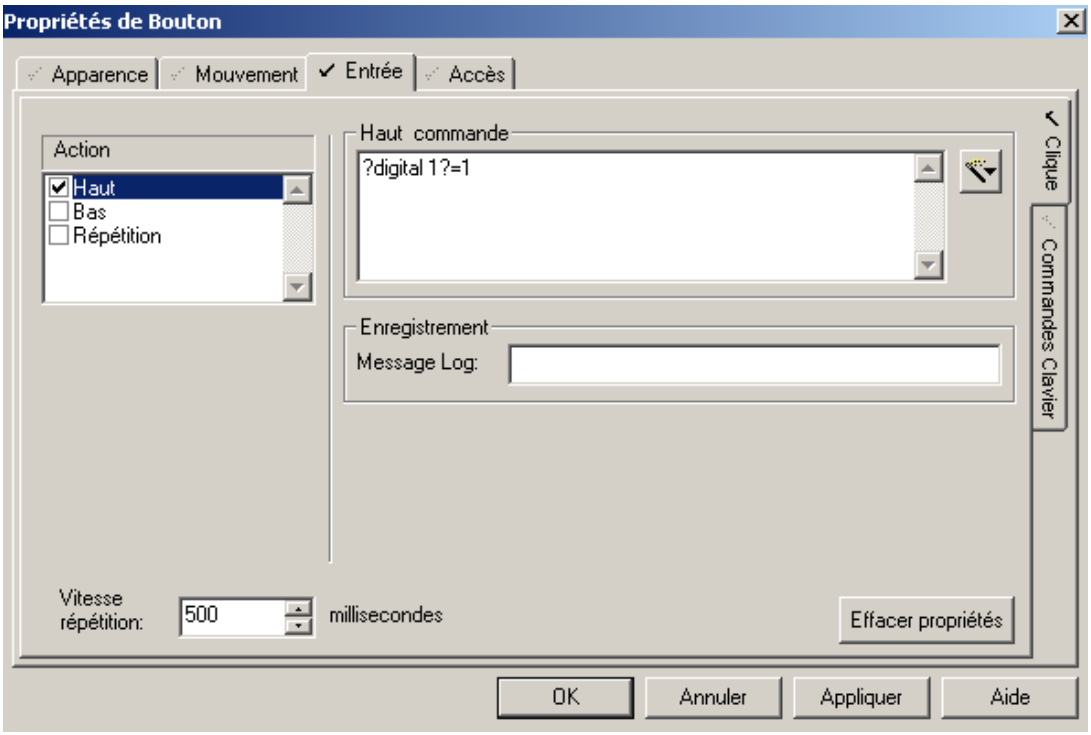


Figure C.44 : Propriété du bouton marche

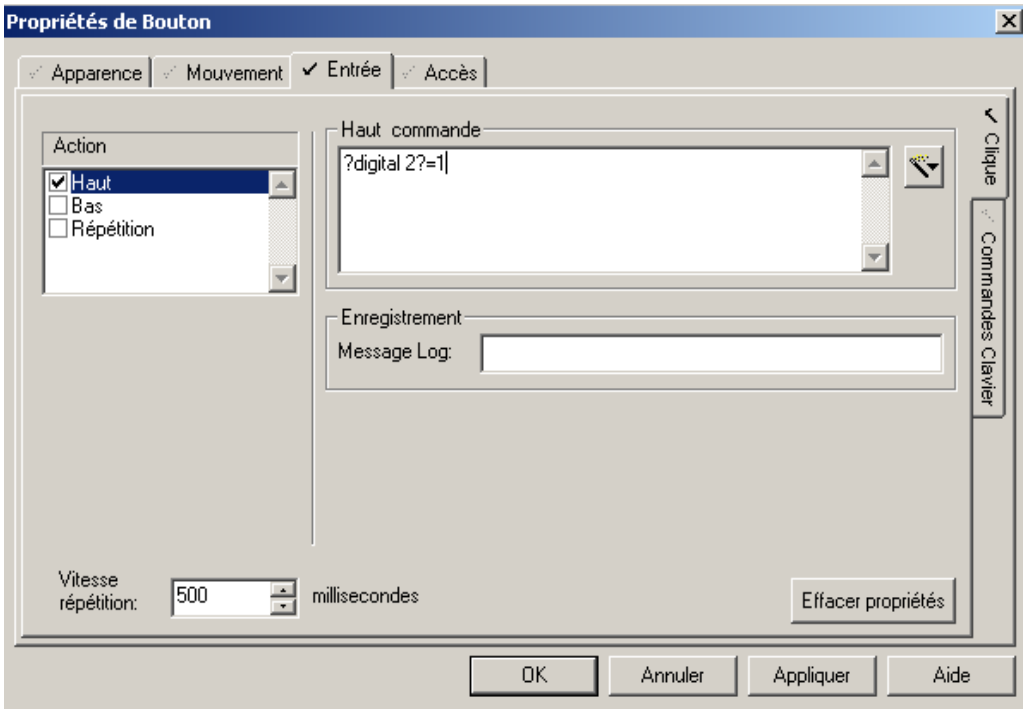


Figure C.45 : Propriété du bouton arreter

▪On sauvegarde la page sous le nom !Pompe.

▪Pour faire appel au super génie a partir du génie créer précédement, on utilise la commande AssPopUp, on ouvre le génie et on regroupe les deux pompes(action de regrouper se trouve dans la barre d'état de l'éditeur). Donc, on a crée un groupe qui se ciompose des deux pompes.

▪Pour faire apple, on utilise l'oglet Entrée, l'appel sera comme suit :

AssPopUp(''Nom du super génie '', ''%Etiq1%'',''%Etiq2%, … ,''%Etiq 8%'').

▪Dans notre cas, comme il n'ya que deux étiquettes, la commande est la suivante :

AssPopUp('' !Pompe'',''%CMD_M%'',''%CMD_A%'')

▪On peut choisir l'importe quel nom pour les étiquettes, ces noms sert uniquement pour l'afectation des variables.

▪il faut écrire les étiquettes en ordre, dans notre cas : CMD_M corespond au bouton marche (le bouton marche est repéré par le numéro 1) et CMD_A correspond au bouton arreter (repéré par le numéro 2).

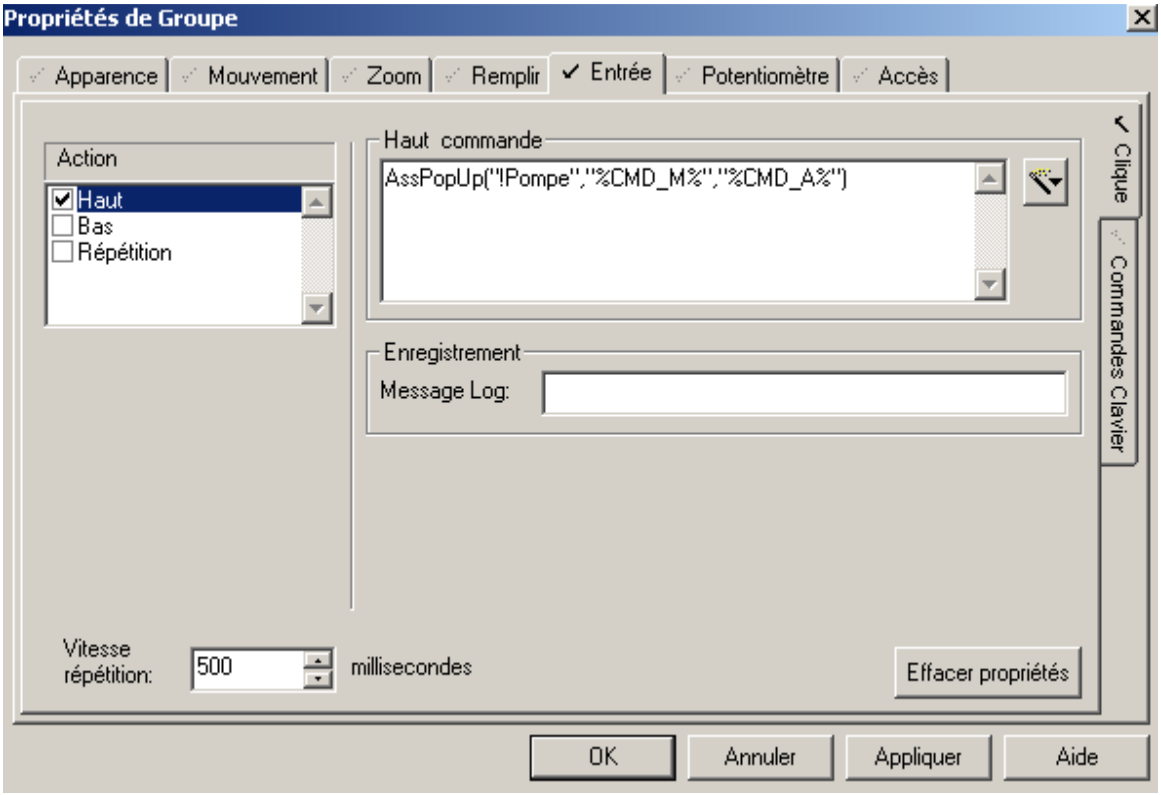


Figure C.46 : Appel du super génie

▪On sauvegarde et à chaque fois qu'on effectue des changement, on met à jour les pages(option qu'on trouve dans la barre d'état de l'éditeur graphique menu Outils).

▪Dans la page principales, quaud on insère le génie pompe qu'on créer, on aura la figure suivante :

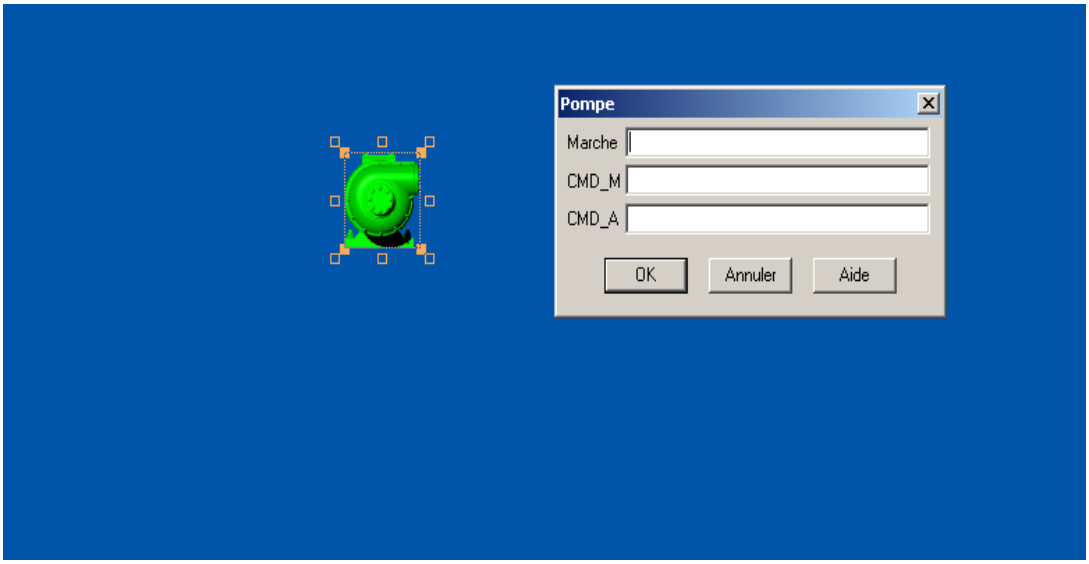


Figure C.47 : Insértion du génie

▪Comme la figure le montre, on doit insérer les vaiables du génie et ceux du super génie. A l'execution, quand on clique sur l'objet pompe la fentre du super génie qu'on créer apparait.

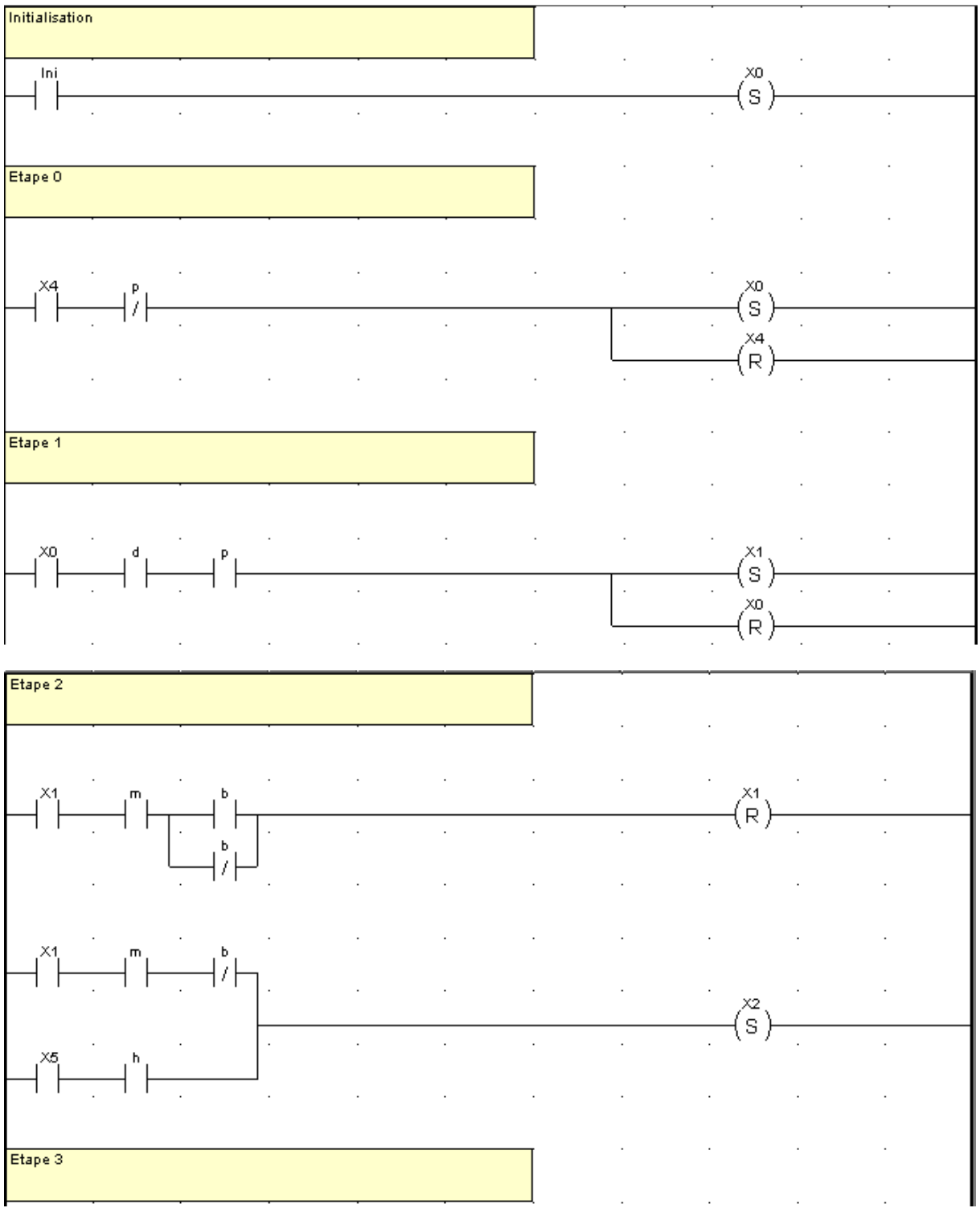
▪Il faut noter que les fenétres des super génies sont géneralement de taille petite et ça se fait en jouant sur les proprités de la page.

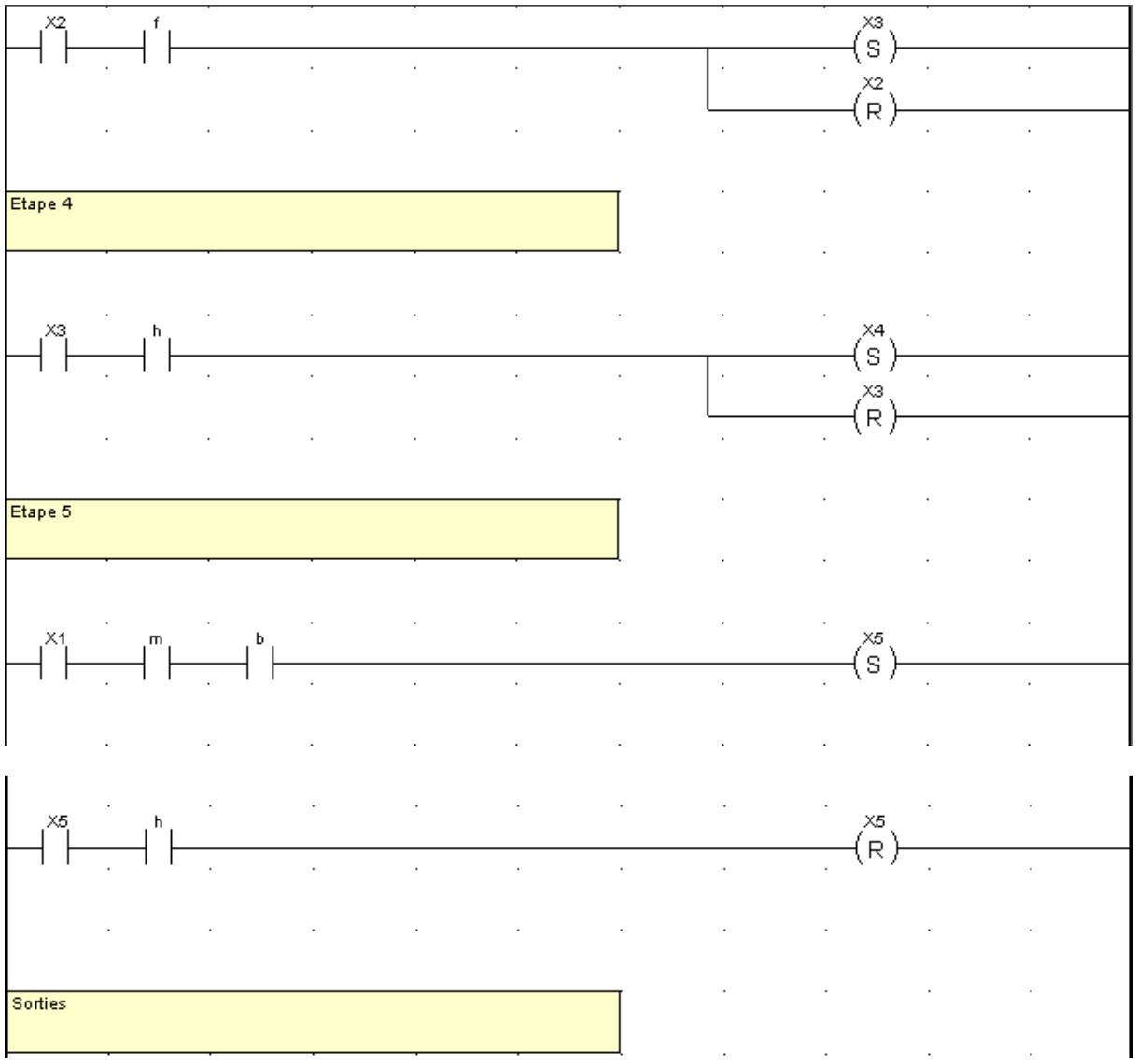
Annexe D

Programme

D.1. Programme de l'exemple de fonctionnement d'une perceuse

- Le programme en langage Ladder est le suivant :




X2
f
X3
S
X2
R
Etape 4
X3
h
X4
S
X3
R
Etape 5
X1
m
b
X5
S

X5
h
X5
R
Sorties

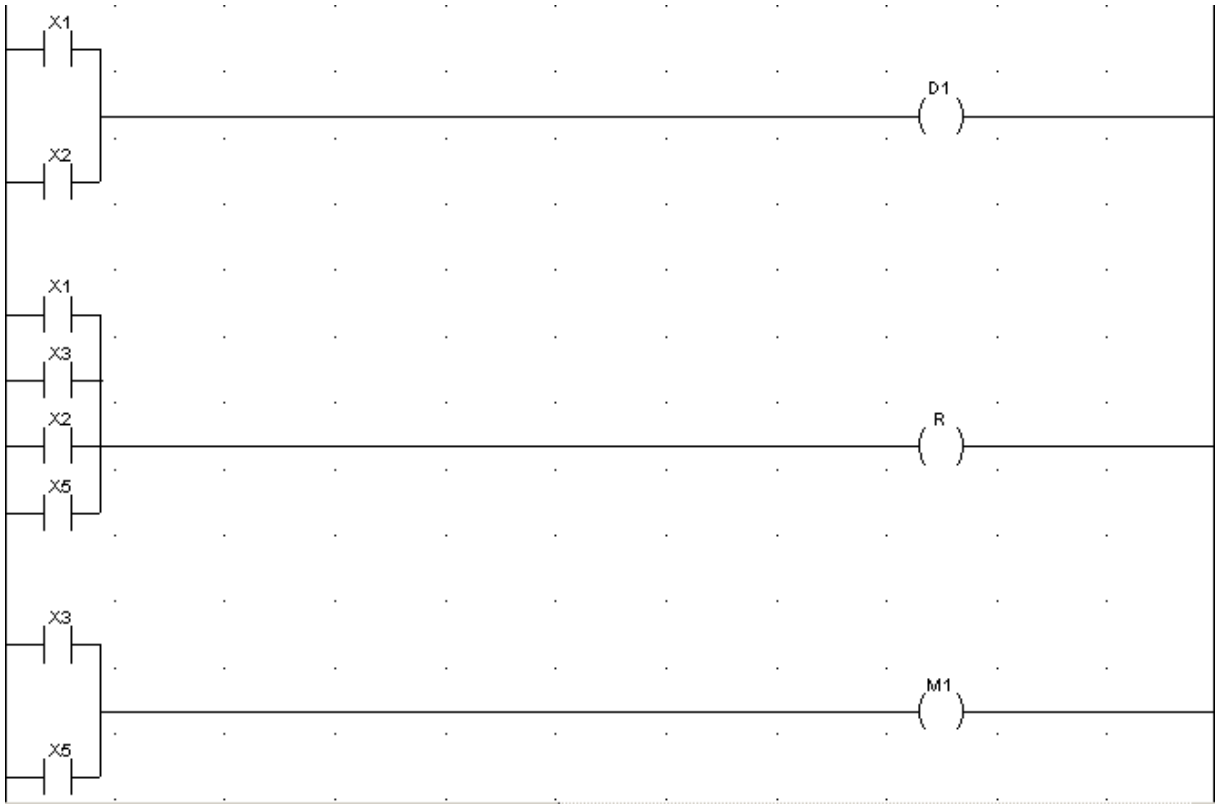
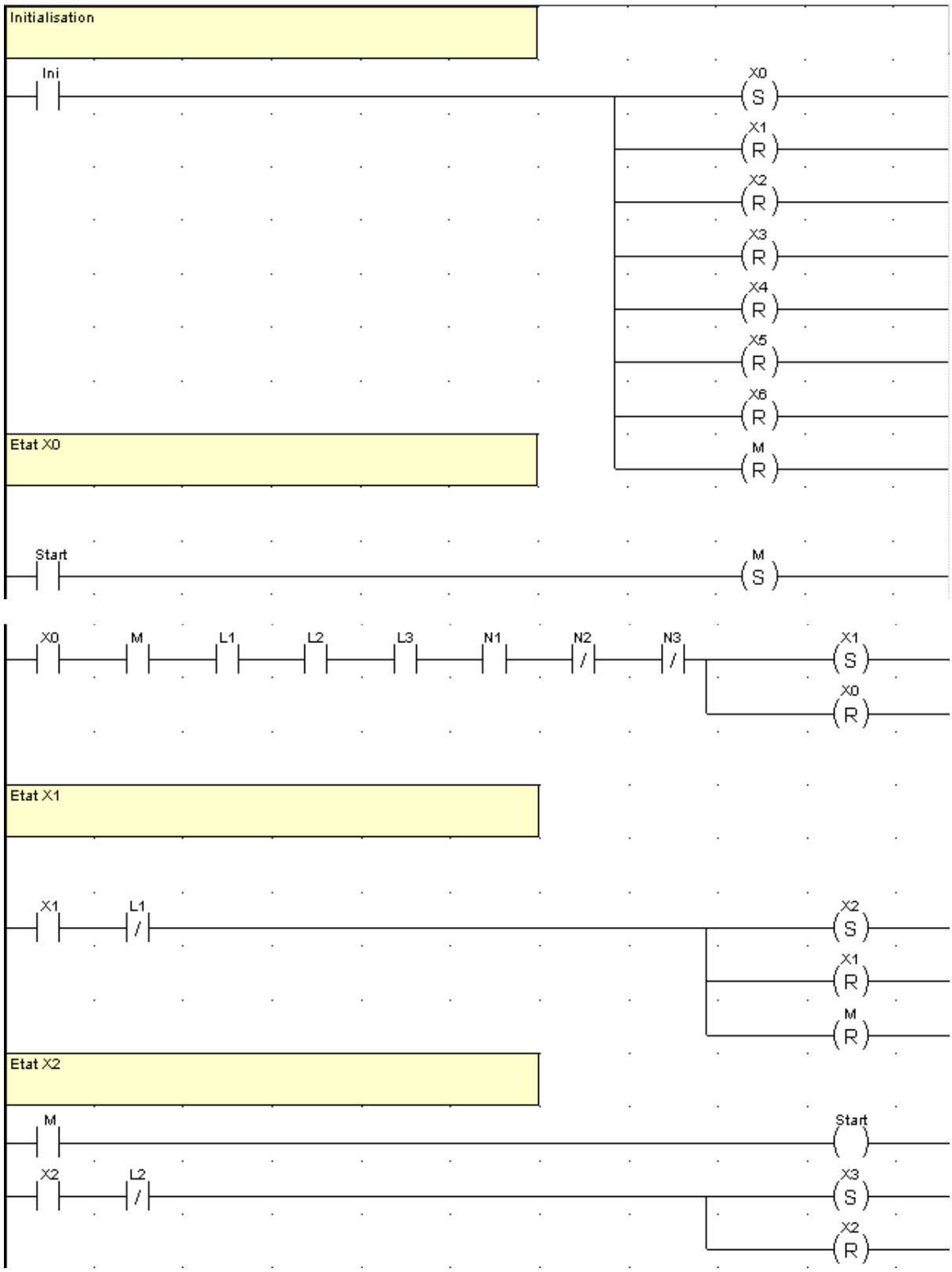


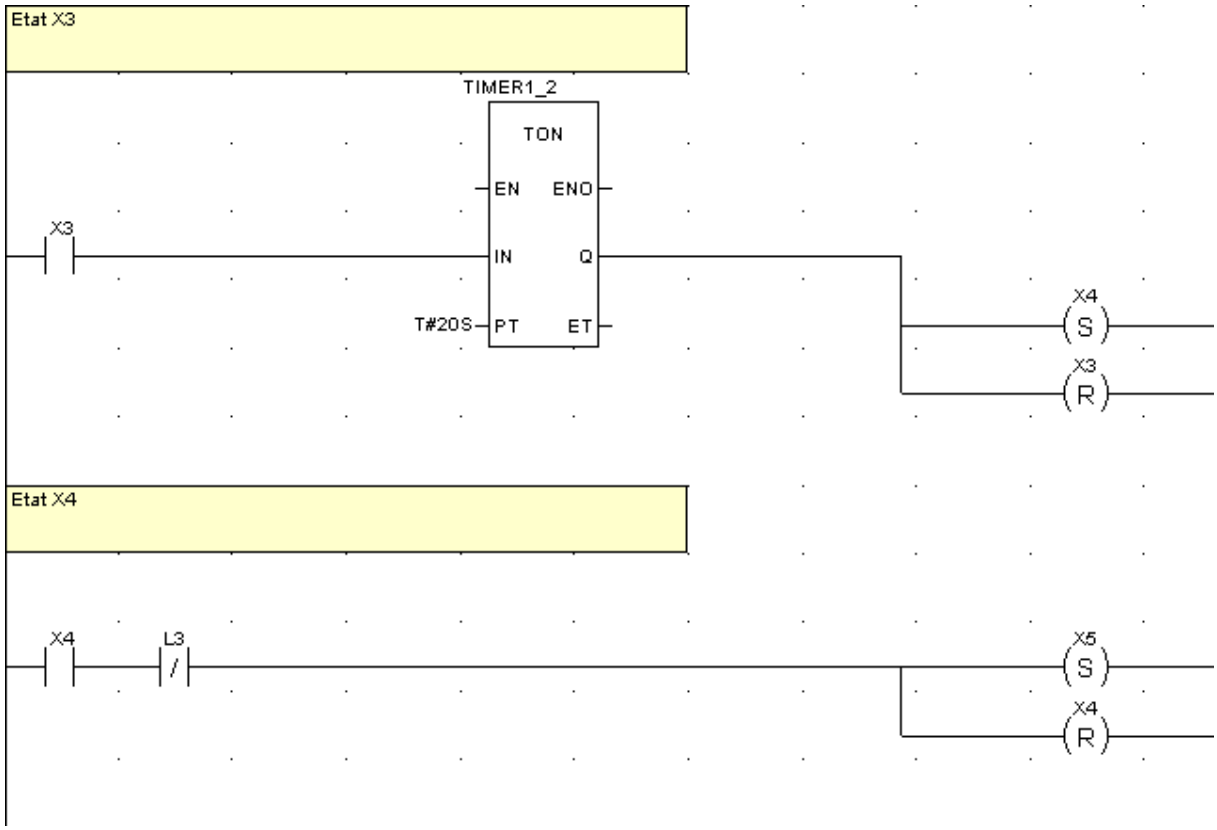
Figure D.1 : Programme perceuse

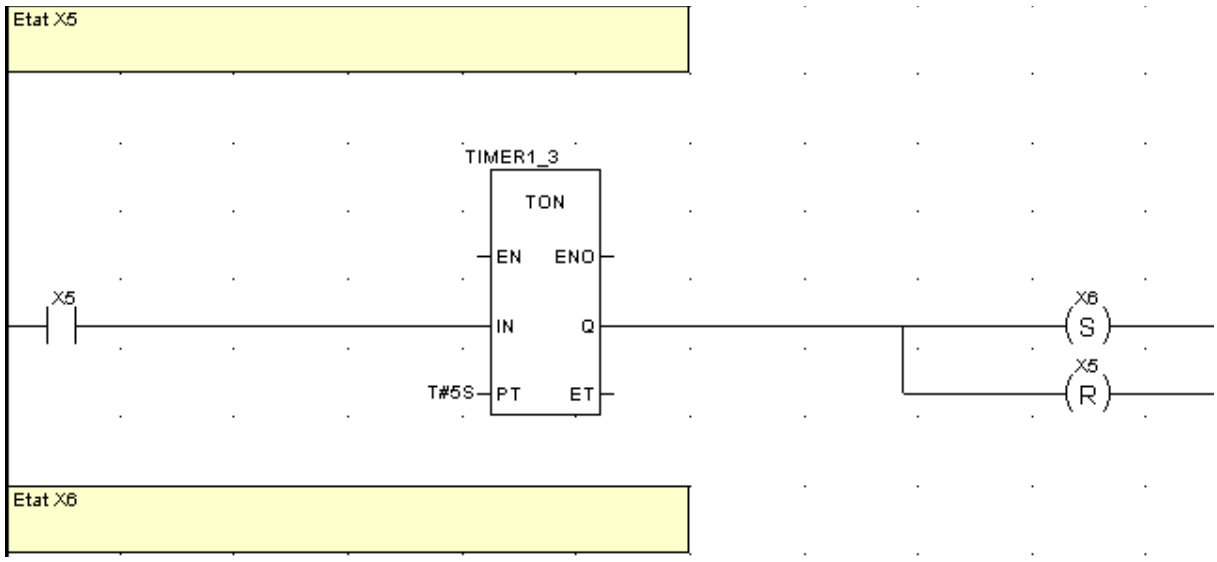
D.2. Programme du processus de production du mélange

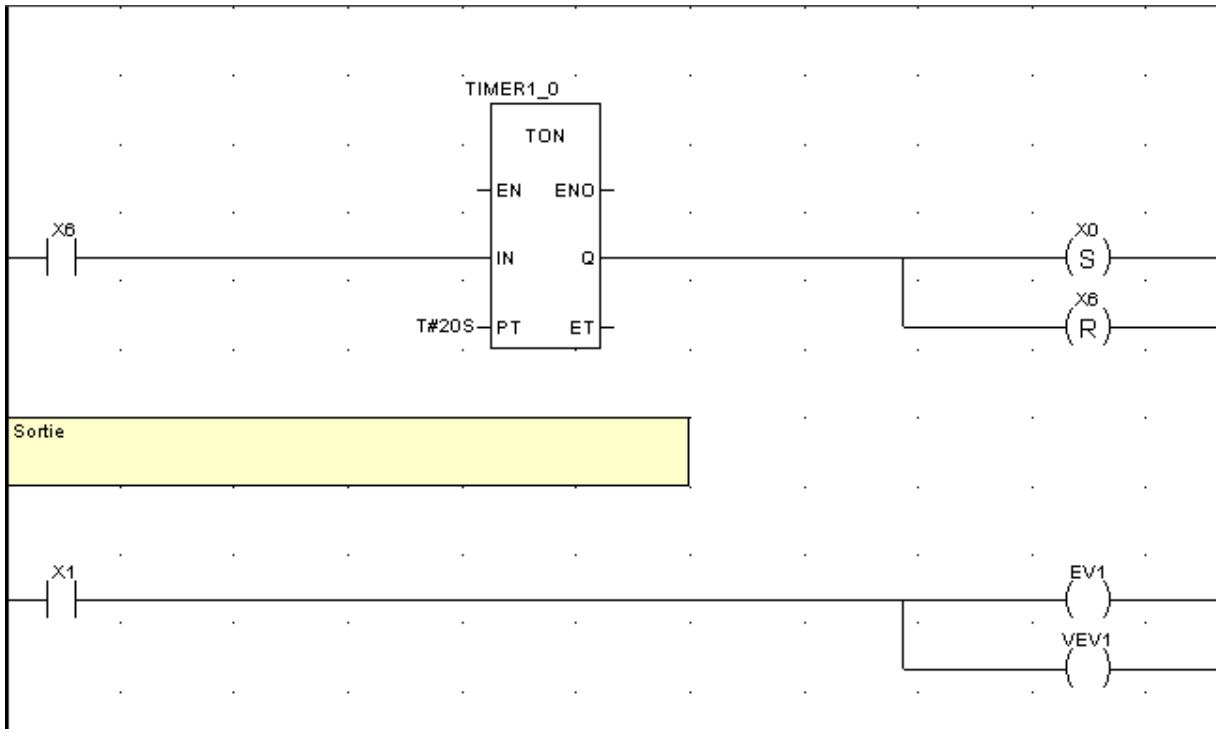
D.2.1. Programme principale 1

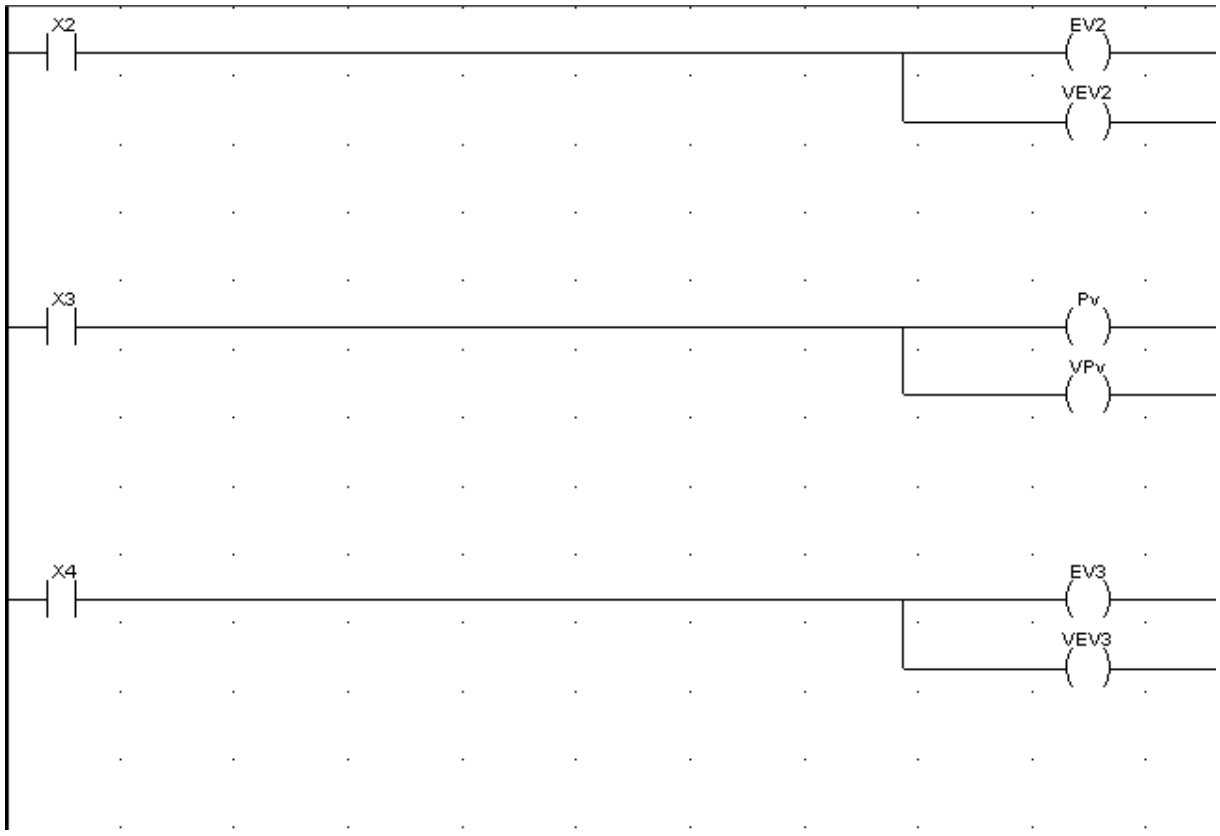
- Le programme en langage Ladder du programme cyclique est le suivant :


Initialisation
Ini
X0
S
X1
R
X2
R
X3
R
X4
R
X5
R
X6
R
M
R
Etat X0
Start
M
S
X0
M
L1
L2
L3
N1
N2
N3
X1
S
X0
R
Etat X1
X1
L1
X2
S
X1
R
M
R
Etat X2
M
Start
X2
L2
X3
S
X2
R


Etat X3
TIMER1_2
TON
EN
ENO
X3
IN
Q
T#20S
PT
ET
X4
S
X3
R
Etat X4
X4
L3
/
X5
S
X4
R


Etat X5
TIMER1_3
TON
EN
ENO
X5
IN
Q
T#5S
PT
ET
X6
S
X5
R
Etat X6


TIMER1_0
TON
EN
ENO
X6
IN
Q
T#20S
PT
ET
X0
S
X6
R
Sortie
X1
EV1
VEV1


X2
EV2
VEV2
X3
Pv
VPv
X4
EV3
VEV3

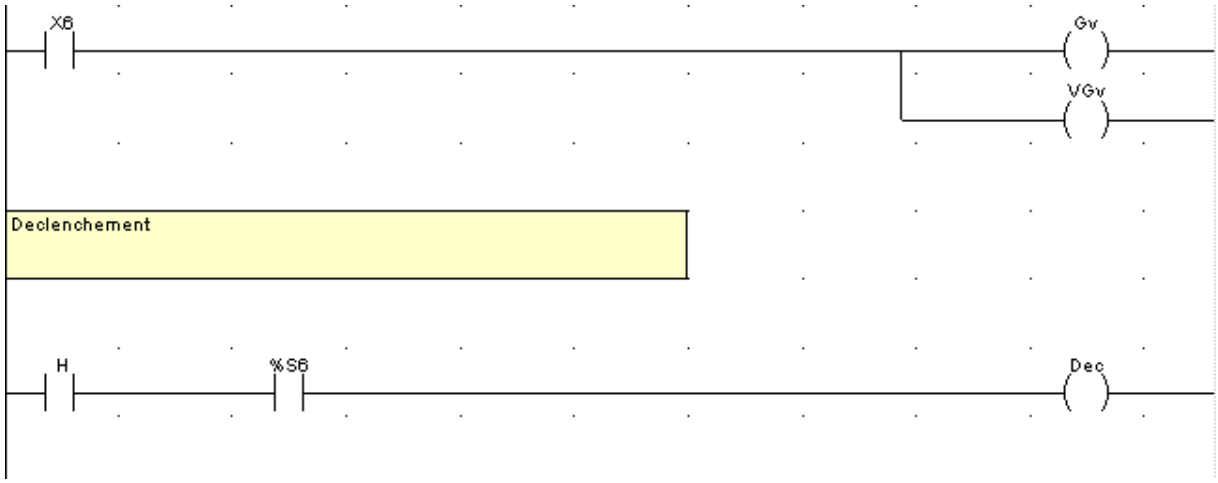
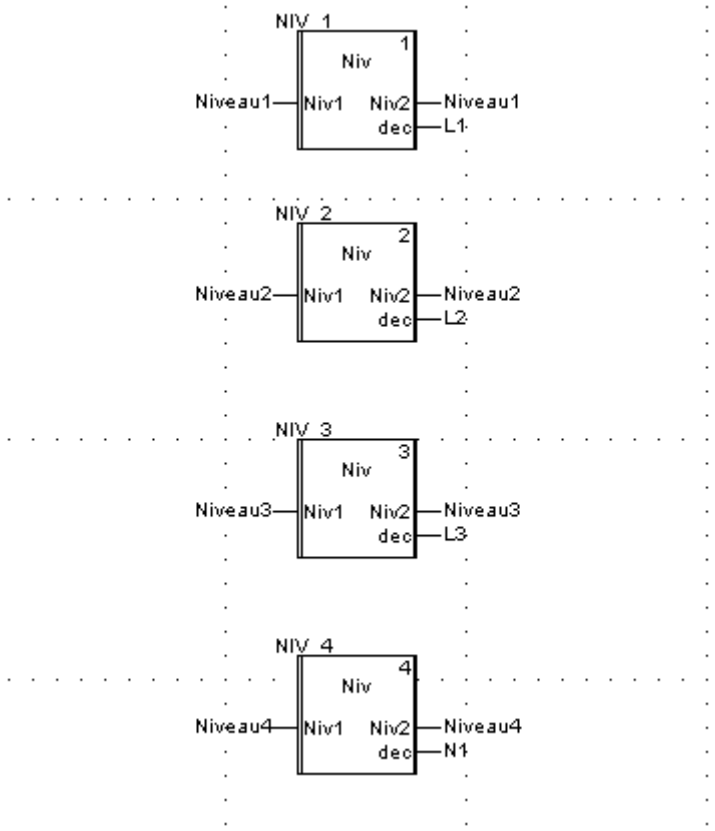


Figure D.2 : Programme principale 1

D.2.2. Programme principale 2

- Le programme est en langage FBD est le suivant :



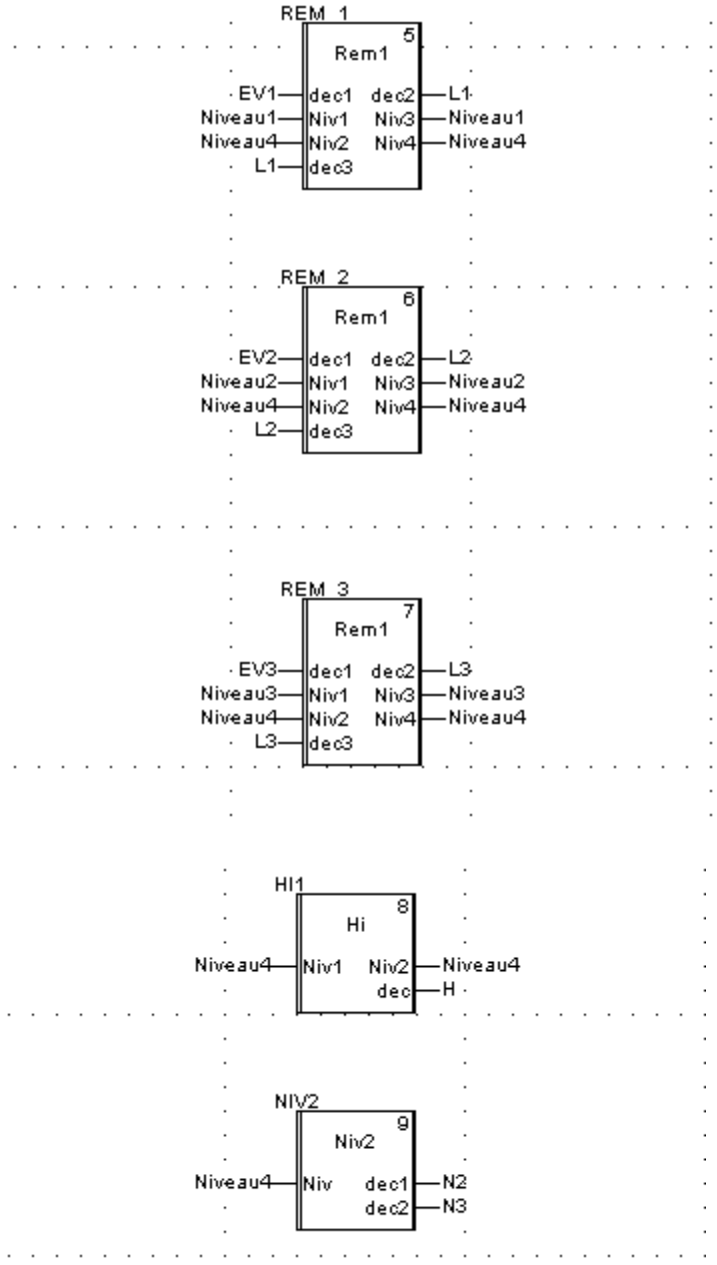


Figure D.3 : Programme principale 2

D.2.3. Programmes DFBs

- Le programme en ST du déclenchement du niveau bas:

```
If Niv1> 25 then
   dec:=1;
   m:=Niv1;
   else
   dec:=0;
   m:=Niv1;
   end_if;
   Niv2:=m;
```

Figure D.4 : DFB déclenchement niveau bas (Niv)

- Le programme en ST du remplissage et du la vidange :

```
if dec1=1 then
        m:=Niv2+Niv1-25;
        Niv3:=25;
        dec2:=0;

    else
        m:=Niv2;
        dec2:=dec3;
        Niv3:=Niv1;
    end_if;
        Niv4:=m;
```

Figure D.5 : DFB remplissage et vidange (Rem1)

- Le programme en ST du déclenchement du clignotant :

```
if Niv1> 90 then
    m:=Niv1;
    dec:=1;
    else
    m:=Niv1;
    dec:=0;
    end_if;
    Niv2:=m;
```

Figure D.6 : DFB clignotant (HI)

D.3. Programmes de l'application

D.3.1. Programme DFB Vanne

```
(***************** Programme de gestion de la vanne******************)


       (*********Ouverture de la vanne*************)
if Ouvrir AND Auto AND NOT Def AND NOT Local then
  set(CMD_O);
  reset(CMD_F);
end_if;

    Tempo1(IN:=CMD_O,PT:=Temp1,Q=>dis1);

        (********* Fermeture de la vanne***********)

If Fermer AND Auto AND NOT Def AND NOT Local then
  set(CMD_F);
  reset(CMD_O);
end_if;

Tempo2(IN:=CMD_F,PT:=Temp1,Q=>dis2);


         (************* Discordance en ouverture ************)
```

```
if dis1 then
   if FCO then
      Disc:=0;
      Etat O:=1;
      Etat F:=0;
      CMD O:=0;
    else
      Disc:=1;
      Etat O:=0;
      Etat F:=1;
      CMD O:=0;
      end_if;
      end_if;


         (************ Discordance en fermeture ****************)
If dis2 then
   if FCF then
     Disc:=0;
     Etat F:=1;
     Etat O:=0;
     CMD F:=0;
   else
     Disc:=1;
     Etat F:=0;
     Etat O:=1;
     CMD F:=0;
     end_If;
     end_if;


         (*************** Incoherance****************)

Tempo3(IN:=(FCO and FCF),PT:=Temp2,Q=>dis3);

IF dis3 THEN
     Inco:=1;
     else
     Inco:=0;
end_if;

          (************** Acquitement de defaut************)

if Acquit then
  Disc:=0;
  Inco:=0;
end_if;
```

▪La section 2 concerne le signal d'avertissement, elle est écrite en langage Ladder (LD) :

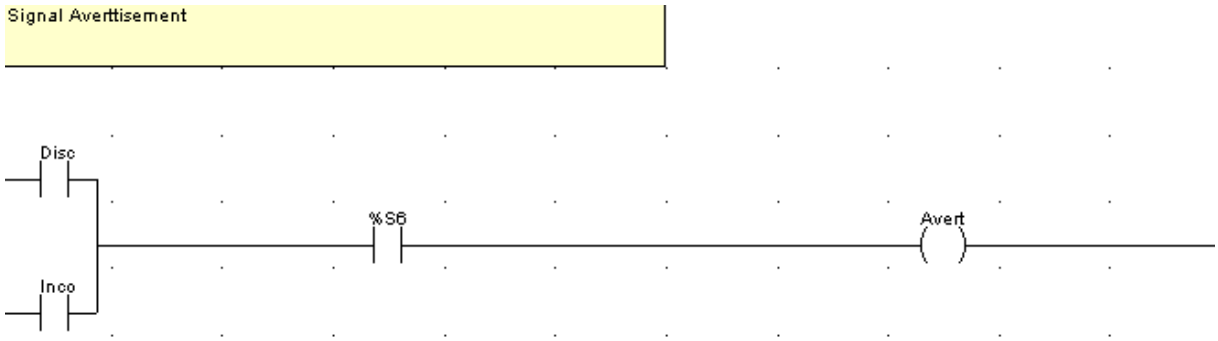


Figure D.7 : Programme DFB Vanne

D.3.2. Programme DFB Défaut

```
(************* DFB Defaut************)

(************ Defaut temperature bobinage ******************)

If (TempB1>Temp_B_Seu) OR (TempB2>Temp_B_Seu) OR (TempB3>Temp_B_Seu) the
   set(Def_T_B);
end_if;


(************* Defaut temperature palier *********************)

If (TempP1>Temp_P_Seu)OR(TempP2>Temp_P_Seu) then
   set(Def_T_P);
end_if;


(****************Defaut de vibration ********************)

IF (Vibra>Vibra_Seu) then
    set(Def_V);
end_if;


 (***************** Defaut analogique generale*************)

If (Def_T_B) OR (Def_T_P) OR (Def_V) then
    set(Def_A);
end_if;
```

```
(******************Defaut logique generale ******************)

if (Def_Van)OR(Disj_Van)OR(Disj_GEP)OR(Def_Iso)OR(Def_Dem)OR(GEp_Sou_Tension)
   set(Def_L);
end_IF;

(******************* Defaut generale*******************)

IF (Def_A) OR (Def_L) then
    set (Def);
end_if;

(********************** Acquitement des defauts ************************)

IF Acquit then
   reset(Def_T_B);
   reset(Def_T_P);
   reset(Def_V);
   reset(Def_A);
   reset(Def_L);
   reset(Def);
   end_if;
```

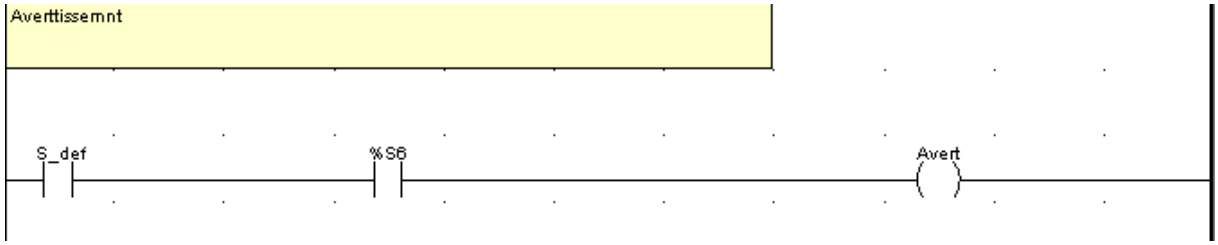
Figure D.8 : Programme DFB Defaut

D.3.3. Programme DFB GEP

```
       (*********** Préparation des vannes ************)

IF Permi THEN
   set(O_V_Am);      (* Ouverture de la vanne amont *)
   set(O_V_R);       (*Ouverture de la vanne regulatrice  *)
   set(F_V_Av);    (* Fermeture de la vanne en aval *)
END_IF;


       (************* Demmarage de GEP ***************)

IF Marche AND (NOT Arret) AND V_Amo AND V_Reg AND (NOT V_Ava) AND (NOT S_def)
   set(CMD_M);       (* envoi du signal de marche *)
   set(Ph_D);        (*etat de demmarage  *)
   set(E_M);         (*etat de marche  *)
   reset(CMD_A);     (*arreter le signal de commande d'arret  *)
   reset(O_V_Am);
   reset(O_V_R);
   reset(F_V_Av);
   set(Dec);
END_IF;


       (***************fin de l'etat de demmarage de la pompe**********)

IF Fin_Dim THEN
    reset(CMD_M);     (* arreter le signal de commande de marche *)
    reset(E_A);       (* arreter l'etat d'arret  *)
    reset(Ph_D);      (* arreter la phase de demmarage *)
END_IF;
```

```
      (*****fermeture de la vanne regulatrice et ouverture de la vanne aval***)

IF Debit>Debit_N AND V_Reg AND NOT V_Ava THEN
    set(O_V_Av);      (* Ouverture de la vanne aval *)
    set(F_V_R);     (* fermeture de la vanne regulatrice *)
END_IF;

IF V_Ava AND (NOT V_Reg) THEN
  reset(O_V_Av);(* arreter le signal de la vanne en aval *)
  reset(F_V_R);(* Arreter le signal si la vanne est completement fermer *)
END_IF;


      (********** fermeture de la vanne amont avant l'arret du GEP *******)

IF Arret AND (NOT Marche) AND V_Amo AND V_Ava AND Auto AND (NOT Man) THEN
   set(CMD_A);        (* envoi du signal de commande d'areter *)
   set(F_V_Am);       (* Fermeture de la vanne amont *)
   reset(CMD_M);      (* arreter le signal de commande de marche *)
   set(Ph_AR);        (* activer la phase arret *)
   reset(E_M);        (* disactiver letat de marche *)
   set(E_A);          (* activer l'etat arret *)
END_IF;

       (*************Arreter le signal si la vanne est fermer**********)
IF NOT V_Amo THEN
   reset(F_V_Am);
END_IF;


      (************fermeture des vannes a l'arret complet de la pompe******)

IF Fin_Arr THEN
    reset(Dec);
    reset(CMD_A);     (* arreter le signal de commande d'arret *)
    reset(Ph_AR);     (* arreter la phase d'arret *)
    set(F_V_Av);    (* fermeture de la vanne aval *)
       IF NOT V_Ava THEN
          reset (F_V_Av);(*Arreter le siganl si la vanne est completement fer
       END_IF;
END_IF;


      (*****************sortie*************)
Def:=S_def;
```

▪Pour le signal d'avertissement, la section est écrite en langage LD :



▪Pour le compteur de marche, la section est écrite en langage LD :

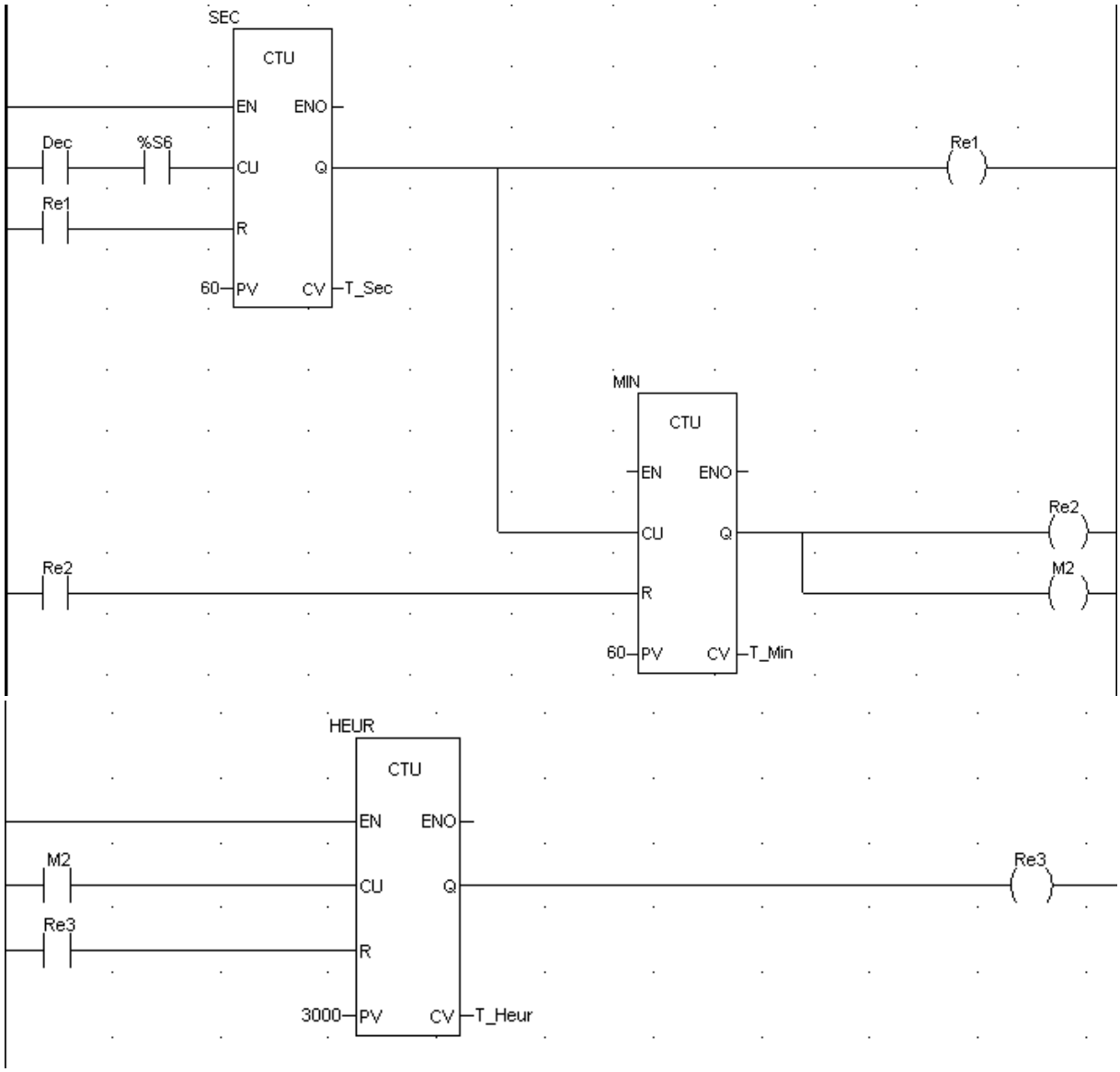


Figure D.9 : Programme DFB GEP

D.3.4. Programme DFB Regime

```
    (****************Cas du premeir régime***************)

IF Debit=N1 THEN
   set(Regi1);
   reset(Regi2);
   reset(Regi3);
   Debi_N:=D1_N;
END_IF;

    (*****************Cas DU deuxieme régime***********)

IF Debit=N2 THEN
   reset(Regi1);
   set(Regi2);
   reset(Regi3);
   Debi_N:=D2_N;
END_IF;

    (*****************Cas du troisième régime*****************)

IF Debit=N3 THEN
   reset(Regi1);
   reset(Regi2);
   set(Regi3);
   Debi_N:=D3_N;
END_IF;
```

Figure D.9 : Programme DFB Regime

D.3.5. Programme DFB Autorisation

```
            (*****************Cas du premier regime****************)

IF Regi1 THEN
   IF Activ1 AND NOT Activ2 AND NOT Activ3 AND NOT Activ4 THEN
      set(Auto1);
      reset(Auto2);
      reset(Auto3);
      reset(Auto4);
      reset(Erreur);
   END_IF;
   IF Activ2 AND NOT Activ1 AND NOT Activ3 AND NOT Activ4 THEN
      reset(Auto1);
      set(Auto2);
      reset(Auto3);
      reset(Auto4);
      reset(Erreur);
   END_IF;
   IF Activ3 AND NOT Activ1 AND NOT Activ2 AND NOT Activ4 THEN
      reset(Auto1);
      reset(Auto2);
      set(Auto3);
      reset(Auto4);
      reset(Erreur);
   END_IF;
```

```
    IF Activ4 AND NOT Activ1 AND NOT Activ2 AND NOT Activ3 THEN
        reset(Auto1);
        reset(Auto2);
        reset(Auto3);
        set(Auto4);
        reset(Erreur);
    END_IF;
    IF Activ1 THEN
        IF Activ2 OR Activ3 OR Activ4 THEN
            set(Erreur);
        END_IF;
    END_IF;
    IF Activ2 THEN
        IF Activ1 OR Activ3 OR Activ4 THEN
            set(Erreur);
        END_IF;
    END_IF;
    IF Activ3 THEN
        IF Activ1 OR Activ2 OR Activ4 THEN
            set(Erreur);
        END_IF;
    END_IF;
    IF Activ4 THEN
        IF Activ1 OR Activ2 OR Activ3 THEN
            set(Erreur);
        END_IF;
    END_IF;
END_IF;

          (*****************Cas DU deuxieme regime*****************)

IF Regi2 THEN
    IF Activ1 AND Activ2 AND NOT Activ3 AND NOT Activ4 THEN
        set(Auto1);
        set(Auto2);
        reset(Auto3);
        reset(Auto4);
        reset(Erreur);
    END_IF;
    IF Activ1 AND NOT Activ2 AND ACtiv3 AND NOT Activ4 THEN
        set(Auto1);
        reset(Auto2);
        set(Auto3);
        reset(Auto4);
        reset(Erreur);
    END_IF;
    IF Activ1 AND NOT Activ2 AND NOT Activ3 AND Activ4 THEN
        set(Auto1);
        reset(Auto2);
        reset(Auto3);
        set(Auto4);
        reset(Erreur);
    END_IF;
```

```
    IF NOT Activ1 AND Activ2 AND Activ3 AND NOT Activ4 THEN
        reset(Auto1);
        set(Auto2);
        set(Auto3);
        reset(Auto4);
        reset(Erreur);
    END_IF;
    IF NOT Activ1 AND Activ2 AND NOT Activ3 AND Activ4 THEN
        reset(Auto1);
        set(Auto2);
        reset(Auto3);
        set(Auto4);
        reset(Erreur);
    END_IF;
    IF NOT Activ1 AND NOT Activ2 AND Activ3 AND Activ4 THEN
        reset(Auto1);
        reset(Auto2);
        set(Auto3);
        set(Auto4);
        reset(Erreur);
    END_IF;
    IF Activ1 AND Activ2 THEN
        IF Activ3 OR Activ4 THEN
            set(Erreur);
        END_IF;
    END_IF;

    IF Activ1 AND Activ3 THEN
        IF Activ2 OR Activ4 THEN
            set(Erreur);
        END_IF;
    END_IF;
    IF Activ1 AND Activ4 THEN
        IF Activ2 OR Activ3 THEN
            set(Erreur);
        END_IF;
    END_IF;
    IF Activ2 AND Activ3 THEN
        IF Activ1 OR Activ4 THEN
            set(Erreur);
        END_IF;
    END_IF;
    IF Activ2 AND Activ4 THEN
        IF Activ1 OR Activ3 THEN
            set(Erreur);
        END_IF;
    END_IF;
    IF Activ3 AND Activ4 THEN
        IF Activ1 OR Activ2 THEN
            set(Erreur);
        END_IF;
    END_IF;
END_IF;
```

```
        (*****************Cas du troisieme regime*************)


IF Regi3 THEN
   IF Activ1 AND Activ2 AND Activ3 AND NOT Activ4 THEN
      set(Auto1);
      set(Auto2);
      set(Auto3);
      reset(Auto4);
   END_IF;
   IF Activ1 AND Activ2 AND NOT Activ3 AND Activ4 THEN
      set(Auto1);
      set(Auto2);
      reset(Auto3);
      set(Auto4);
   END_IF;
   IF Activ1 AND NOT Activ2 AND Activ3 AND Activ4 THEN
      set(Auto1);
      reset(Auto2);
      set(Auto3);
      set(Auto4);
   END_IF;
   IF NOT Activ1 AND Activ2 AND Activ3 AND Activ4 THEN
      reset(Auto1);
      set(Auto2);
      set(Auto3);
      set(Auto4);
   END_IF;
```

```
    IF Activ1 AND Activ2 AND Activ3 THEN
        IF Activ4 THEN
            set(Erreur);
        ELSE
            reset(Erreur);
        END_IF;
    END_IF;
    IF Activ1 AND Activ2 AND Activ4 THEN
        IF Activ3 THEN
            set(Erreur);
        ELSE
            reset(Erreur);
        END_IF;
    END_IF;
    IF Activ1 AND Activ3 AND Activ4 THEN
        IF Activ2 THEN
            set(Erreur);
        ELSE
            reset(Erreur);
        END_IF;
    END_IF;
    IF Activ2 AND Activ3 AND Activ4 THEN
        IF Activ1 THEN
            set(Erreur);
        ELSE
            reset(Erreur);
        END_IF;

    END_IF;
END_IF;
```

▪La deuxième section est écrite en LD, qui concerne le signal d'avertissement :

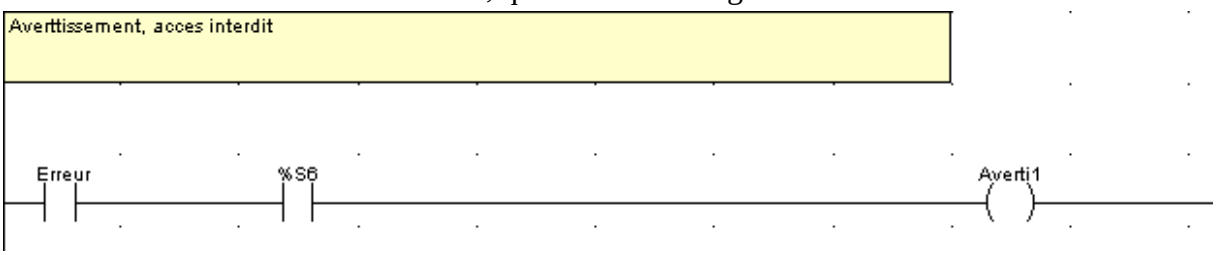


Figure D.10 : Programme DFB Autorisation

D.3.6. Programme DFB Validation

```
      (***********Autorisation finale pour la premiere ligne**********)

IF Permi1 AND (NOT Erreur) THEN
      IF Valider THEN
         set(Auto1);
      ELSE
      reset(Auto1);
   END_IF;
END_IF;

      (***********Autorisation finale pour la deuxieme ligne**********)

IF Permi2 AND (NOT Erreur) THEN
   IF Valider THEN
      set(Auto2);
   ELSE
      reset(Auto2);
   END_IF;
END_IF;

       (***********Autorisation finale pour la troisieme ligne**********)


IF Permi3 AND (NOT Erreur) THEN
   IF Valider THEN
      set(Auto3);
   ELSE
      reset(Auto3);
   END_IF;
END_IF;

       (***********Autorisation finale pour la quatrieme ligne**********)
IF Permi4 AND (NOT Erreur) THEN
   IF Valider THEN
      set(Auto4);
   ELSE
      reset(Auto4);
   END_IF;
END_IF;
```

Figure D.11 : Programme DFB Validation

D.3.7. Programme DFB intégration

```
        (**************Initialisation***********)
IF %S21 THEN
   Y1:=Y0;
   Y2:=dY0;
   Y3:=d2Y0;
END_IF;

         (*****************Equation du premier ordre***********)

IF n=1 THEN

              (******************Calcul des Ki************)

       K1:=-a1*Y1-a2+a3*u;
       K2:=-a1*(Y1+h*K1/2.0)-a2+a3*u;
       K3:=-a1*(Y1+h*K2/2.0)-a2+a3*u;
       K4:=-a1*(Y1+h*K3)-a2+a3*u;

              (********************Mise a jour************)

       Y1:=Y1+(h/6.0)*(K1+2.0*K2+2.0*K3+K4);

              (*************Sortie**************)

       Y:=Y1;

         (*****************Equation du deuxième ordre***********)

ELSE IF n=2 THEN

               (**************Calcul des Ki et des Vi**********)

       K1:=Y2;
       V1:=-a1*Y2-a2*Y1-a3+a4*u;

       K2:=Y2+h*K1/2.0;
       V2:=-a1*(Y2+h*K1/2.0)-a2*(Y1+h*V1/2.0)-a3+a4*u;

       K3:=Y2+h*K2/2.0;
       V3:=-a1*(Y2+h*K2/2.0)-a2*(Y1+h*V2/2.0)-a3+a4*u;

       K4:=Y2+h*K3;
       V4:=-a1*(Y2+h*K3)-a2*(Y1+h*V3)-a3+a4*u;

              (**************Mise a jour**************)

       Y1:=Y1+(h/6.0)*(K1+2.0*K2+2.0*K3+K4);
       Y2:=Y2+(h/6.0)*(V1+2.0*V2+2.0*V3+V4);

              (***************Sortie******************)
       Y:=Y1;

           (*****************Equation du troisième ordre***********)
```

```
ELSE IF n=3 THEN

          (**************Calcul des Ki, Vi et Qi**********)

      K1:=Y2;
      V1:=Y3;
      Q1:=-a1*Y3-a2*Y2-a3*Y1-a4+a5*u;

      K2:=Y2+h*K1/2.0;
      V2:=Y3+h*V1/2.0;
      Q2:=-a1*(Y3+h*V1/2.0)-a2*(Y2+h*K1/2.0)-a3*(Y1+h*Q1/2.0)-a4+a5*u;

      K3:=Y2+h*K2/2.0;
      V3:=Y3+h*V2/2.0;
      Q3:=-a1*(Y3+h*V2/2.0)-a2*(Y2+h*K2/2.0)-a3*(Y1+h*Q2/2.0)-a4+a5*u;

      K4:=Y2+h*K3;
      V4:=Y3+h*V3;
      Q4:=-a1*(Y3+h*V3)-a2*(Y2+h*K3)-a3*(Y1+h*Q3)-a4+a5*u;

            (**************Mise a jour**************)

      Y1:=Y1+(h/6.0)*(K1+2.0*K2+2.0*K3+K4);
      Y2:=Y2+(h/6.0)*(V1+2.0*V2+2.0*V3+V4);
      Y3:=Y3+(h/6.0)*(Q1+2.0*Q2+2.0*Q3+Q4);

           (***************Sortie******************)

      Y:=Y1;

END_IF;
END_IF;
END_IF;
```

Figure D.12 : Programme DFB intégration

D.3.8. Programme régulateur PI

```
        (*************L'erreur************)

E:=C-M;

        (************Integration de l'erreur***********)

INTEG1(u:=E,a1:=0.0,a2:=0.0,a3:=1.0,a4:=0.0,h:=h,Y0:=0.0,dY0:=0.0,n:=1,Y=>M1);

        (************Action proportionnelle et integrale*********)
Yp:=Kp*E;
Yi:=(Kp/Ti)*M1;

         (*****************Signal de sortie**************)
Y:=Yp+Yi;
```

Figure D.13 : Programme DFB régulateur PI