

Documentation Projet – Station de Pompage Automatisée

API (Machine Expert Basic) & IHM (Vijeo Designer Basic)

Hail Amine
Portfolio Automatisme / Supervision

décembre 2023

Table des matières

Résumé	3
1 Contexte et objectifs	4
1.1 Contexte	4
1.2 Objectifs fonctionnels (cahier des charges)	4
1.2.1 Commande de base	4
1.2.2 Signalisation d'état	4
1.2.3 Défaut et acquittement	4
1.2.4 Protections et sécurité	4
1.2.5 Multi-pompes et optimisation	5
1.2.6 Gestion du niveau analogique et du nombre de pompes	5
1.3 Outils logiciels	5
2 Architecture technique	6
2.1 Entrées / Sorties (E/S)	6
2.1.1 E/S TOR (Tout Ou Rien)	6
2.1.2 Entrée analogique	6
2.2 Hypothèses de fonctionnement	6
3 Analyse fonctionnelle et logique de commande (API)	7
3.1 Principes généraux	7
3.2 Gestion Marche / Arrêt (mémoire de commande)	7
3.2.1 Besoin	7
3.2.2 Principe	7
3.2.3 Condition d'autorisation	7
3.3 Gestion des défauts (surchauffe + AU) et Reset	8
3.3.1 Détection	8
3.3.2 Mémorisation et acquittement	8
3.3.3 Signalisation	8
3.4 Protection marche à sec (NTB)	8
3.5 Extension multi-pompes : alternance et optimisation	8
3.5.1 Alternance à chaque démarrage (2 pompes)	8
3.5.2 Temps de marche (compteur en secondes)	8
3.5.3 Alternance temporelle (pompe la moins utilisée)	9
3.5.4 Extension à 3 pompes	9
3.6 Gestion du nombre de pompes selon niveau analogique	9
3.6.1 Entrée NIV_RES	9
3.6.2 Règles de commande	9
3.6.3 Choix des pompes	9

4 Supervision IHM (Vijeo Designer Basic)	10
4.1 Création de la page synoptique	10
4.2 Déclaration de l'API (communication Modbus TCP/IP)	10
4.3 Importation des variables	10
4.4 Animation des objets	10
4.5 Simulation	10
4.6 Gestion des alarmes	10
4.7 Navigation entre pages	11
4.8 Droits utilisateurs (sécurité)	11
4.8.1 Création utilisateur	11
4.8.2 Restriction d'accès page alarmes	11
4.8.3 Bouton de login / logout	11
5 Validation et tests	12
5.1 Plan de tests (exemples)	12
6 Résultats et compétences	13
6.1 Livrables	13
6.2 Compétences mobilisées	13
Conclusion	14

Résumé

Ce projet consiste à automatiser une station de pompage pilotée par un automate programmable (API), puis à réaliser une interface homme-machine (IHM) pour superviser le fonctionnement. Le projet est construit par étapes : commande d'une pompe, ajout des états (voyants), gestion des défauts et sécurités (surchauffe, arrêt d'urgence, marche à sec), extension multi-pompes avec alternance et optimisation par temps de marche, prise en compte d'un niveau analogique, puis supervision (communication, animation, alarmes, navigation, droits utilisateurs).

Chapitre 1

Contexte et objectifs

1.1 Contexte

Une station de pompage doit assurer un fonctionnement fiable tout en protégeant les équipements :

- Démarrer/arrêter une pompe sur commande opérateur.
- Afficher clairement l'état (marche / arrêt / défaut).
- Empêcher les démarrages dangereux (défaut, marche à sec).
- Garantir la sécurité via arrêt d'urgence.
- Optimiser l'exploitation en répartissant l'usure sur plusieurs pompes.
- Superviser le système via une IHM (alarmes, navigation, accès sécurisé).

1.2 Objectifs fonctionnels (cahier des charges)

1.2.1 Commande de base

- La pompe démarre quand l'opérateur appuie sur **Marche**.
- La pompe s'arrête quand l'opérateur appuie sur **Arrêt**.

1.2.2 Signalisation d'état

- Voyant **vert** allumé si la pompe est en marche.
- Voyant **rouge** allumé si la pompe est à l'arrêt.

1.2.3 Défaut et acquittement

- Voyant **orange** allumé en cas de défaut (ex : surchauffe).
- Le défaut reste mémorisé et ne disparaît qu'après appui sur **RESET**.
- Tant que le défaut est actif : **interdiction de démarrage**.

1.2.4 Protections et sécurité

- Protection contre la marche à sec via une **poire de niveau NTB**.
- Présence d'un **arrêt d'urgence (AU)**, traité comme un défaut.

1.2.5 Multi-pompes et optimisation

- Ajout d'une 2^e pompe : alternance à chaque démarrage.
- Si la pompe sélectionnée est en défaut : bascule vers une autre pompe.
- Calcul du **temps de marche** de chaque pompe (en secondes), à l'aide d'une variable système.
- Alternance **temporelle** : démarrer la pompe ayant le temps de fonctionnement le plus faible.
- Extension à 3 pompes selon la même logique.

1.2.6 Gestion du niveau analogique et du nombre de pompes

- Ajout d'une sonde analogique : niveau mesuré entre **0 et 500 cm**.
- Règles de fonctionnement :
 - Au-dessus de **350 cm** : **2 pompes** en marche.
 - Au-dessus de **200 cm** : **1 pompe** en marche.
 - En dessous de **100 cm** : **arrêt des pompes**.
 - Choix des pompes basé sur le **temps de marche le plus faible**.

1.3 Outils logiciels

- Programmation API : **Machine Expert Basic**.
- Supervision : **Vijeo Designer Basic**.
- Communication IHM-API : **Modbus TCP/IP**.

Chapitre 2

Architecture technique

2.1 Entrées / Sorties (E/S)

2.1.1 E/S TOR (Tout Ou Rien)

Signal	Adresse	Rôle
BP_MARCHE	%I0.0	Commande de démarrage
BP_ARRET	%I0.1	Commande d'arrêt
SURCHAUFFE_PMP	%I0.2	Détection défaut surchauffe
RESET	%I0.3	Acquittement défaut (réarmement)
POIRE_NTB	%I0.4	Niveau bas (anti-marche à sec)
ARR_URG	%I0.5	Arrêt d'urgence (défaut sécurité)
MAR_POMPE_1	%Q0.0	Sortie commande pompe 1
V_MARCHE	%Q0.1	Voyant vert : pompe en marche
V_ARRET	%Q0.2	Voyant rouge : pompe arrêt
V_DEFAULT	%Q0.3	Voyant orange : défaut

2.1.2 Entrée analogique

Signal	Adresse	Rôle
NIV_RES	%IW1.0	Niveau analogique du réservoir (0 à 500 cm)

2.2 Hypothèses de fonctionnement

- Une pompe ne peut démarrer que si les sécurités sont satisfaites (pas de défaut, niveau bas non atteint).
- L'arrêt d'urgence a priorité sur toute commande (mise en sécurité immédiate).
- Les sorties voyants reflètent l'état réel de l'installation (marche/arrêt/défaut).

Chapitre 3

Analyse fonctionnelle et logique de commande (API)

3.1 Principes généraux

La logique API s'articule autour de trois blocs :

1. **Gestion des commandes opérateur** (Marche / Arrêt).
2. **Gestion des défauts et sécurités** (surchauffe, AU, marche à sec).
3. **Gestion des pompes** (sélection, alternance, temps de marche, nombre de pompes en service).

3.2 Gestion Marche / Arrêt (mémoire de commande)

3.2.1 Besoin

La pompe doit rester en marche après un appui sur **Marche** et s'arrêter sur appui **Arrêt**.

3.2.2 Principe

On utilise une **mémoire de marche** (bascule) :

- SET par BP_MARCHE (si autorisé).
- RESET par BP_ARRET (ou défaut / AU / niveau bas).

3.2.3 Condition d'autorisation

La mise en marche est autorisée si :

- Aucun défaut n'est actif (y compris arrêt d'urgence).
- La poire NTB indique que la marche à sec n'est pas présente (niveau bas non atteint).

3.3 Gestion des défauts (surchauffe + AU) et Reset

3.3.1 Détection

Le défaut est activé si :

- SURCHAUFFE_PMP = 1 ou
- ARR_URG = 1

3.3.2 Mémorisation et acquittement

- Le défaut est **mémorisé** (latched).
- Il ne disparaît qu'à l'appui sur **RESET** (et si la cause du défaut a disparu).
- Tant que défaut actif : commande pompe forcée à 0 + démarrage interdit.

3.3.3 Signalisation

- V_DEFAUT = 1 si défaut actif.

3.4 Protection marche à sec (NTB)

- Si POIRE_NTB signale niveau bas : arrêt immédiat des pompes.
- Démarrage interdit tant que la condition NTB n'est pas redevenue correcte.

3.5 Extension multi-pompes : alternance et optimisation

3.5.1 Alternance à chaque démarrage (2 pompes)

Besoin

À chaque nouvelle demande de marche, alterner automatiquement entre pompe 1 et pompe 2.

Principe

- Une variable interne **SEL** (sélecteur) bascule à chaque démarrage validé.
- La pompe choisie est commandée si elle n'est pas en défaut.
- Si la pompe choisie est en défaut : bascule vers l'autre pompe.

3.5.2 Temps de marche (compteur en secondes)

Besoin

Mesurer la durée cumulée de fonctionnement de chaque pompe.

Principe

- Utiliser une impulsion système (ex : %S6) comme base de temps (1 s).
- Quand la pompe est active, incrémenter son compteur **T_P1**, **T_P2**, etc.

3.5.3 Alternance temporelle (pompe la moins utilisée)

Besoin

Choisir la pompe ayant le **temps de marche le plus faible** pour équilibrer l'usure.

Principe

- À chaque démarrage :
 - comparer **T_P1** et **T_P2**,
 - sélectionner la plus petite valeur,
 - vérifier la disponibilité (pas de défaut), sinon sélectionner l'autre.

3.5.4 Extension à 3 pompes

- Ajouter une 3^e pompe et son compteur **T_P3**.
- À chaque démarrage, sélectionner la pompe disponible ayant le plus petit temps ($\min(T_P1, T_P2, T_P3)$).
- Si la pompe sélectionnée est en défaut : choisir la suivante la plus faible parmi les disponibles.

3.6 Gestion du nombre de pompes selon niveau analogique

3.6.1 Entrée NIV_RES

Le niveau est lu sur **NIV_RES (%IW1.0)** et correspond à une plage de **0 à 500 cm**.

3.6.2 Règles de commande

- Si **NIV_RES > 350 cm** : **2 pompes** demandées.
- Si **NIV_RES > 200 cm** : **1 pompe** demandée.
- Si **NIV_RES < 100 cm** : **0 pompe** (arrêt).

3.6.3 Choix des pompes

- Pour 1 pompe : choisir la pompe disponible ayant le plus faible temps.
- Pour 2 pompes : choisir les **deux** pompes disponibles avec les temps les plus faibles.
- En cas de défaut sur une pompe : l'exclure automatiquement de la sélection.

Chapitre 4

Supervision IHM (Vijeo Designer Basic)

4.1 Création de la page synoptique

- Réaliser le **dessin de la station** (réservoir, pompes, voyants, niveau, boutons).

4.2 Déclaration de l'API (communication Modbus TCP/IP)

1. Dans **IO Manager** : clic droit → **New Driver**.
2. Choisir **Modbus TCP/IP – Modbus Equipment**.
3. Configurer l'équipement (adresse IP, paramètres Modbus).

4.3 Importation des variables

- Importer le projet / tags depuis **Machine Expert Basic** vers **Vijeo Designer Basic**.

4.4 Animation des objets

- Associer chaque objet IHM (voyant, bouton, valeur niveau) à la variable correspondante (état pompe, défaut, etc.).

4.5 Simulation

- Lancer la simulation via **Build → Simulate**.
- Vérifier l'affichage, les commandes, et le retour d'état.

4.6 Gestion des alarmes

1. Ajouter un **groupe d'alarmes**.
2. Ajouter les variables concernées (défauts, AU, niveau bas, etc.).

3. Créer une **page d'alarmes**.
4. Insérer un **tableau d'alarmes** pour afficher la liste.

4.7 Navigation entre pages

- Ajouter deux boutons pour passer entre :
 - la page synoptique,
 - la page alarmes.

4.8 Droits utilisateurs (sécurité)

4.8.1 Crédation utilisateur

- Déclarer l'utilisateur **opérateur1** avec un mot de passe.
- **opérateur1** est le seul autorisé à accéder à la page d'alarmes.

4.8.2 Restriction d'accès page alarmes

- Définir le niveau de sécurité de la page d'alarmes (ex : **SecurityGroup01**) dans **Security level**.

4.8.3 Bouton de login / logout

- Ajouter le bouton de login depuis la librairie.
- Ajouter une action de type **Switch** :
 - action **Logout**,
 - puis action **Login**.

Chapitre 5

Validation et tests

5.1 Plan de tests (exemples)

Test	Procédure	Résultat attendu
Marche / Arrêt Voyants	Appuyer Marche puis Arrêt Observer V_MARCHE / V_ARRET	La pompe démarre puis s'arrête Vert si marche, rouge si arrêt
Défaut surchauffe	Forcer SURCHAUFFE_PMP=1	V_DEFAUT=1, pompe stoppée, démarrage interdit
Reset défaut	Appuyer RESET après disparition cause	V_DEFAUT=0, démarrage possible
Marche à sec	Activer POIRE_NTB (niveau bas)	Arrêt pompes, démarrage interdit
Arrêt d'urgence	Activer ARR_URG	Arrêt immédiat, défaut actif
Alternance 2 pompes	Démarrer/arrêter plusieurs fois	Sélection alternée, bascule si défaut
Temps de marche	Laisser tourner pompe	T_P* augmente en secondes
Niveau analogique	Faire varier NIV_RES	0/1/2 pompes selon seuils (100/200/350 cm)
IHM / alarmes	Provoquer un défaut	Apparition dans la page alarmes
Droits opérateur	Accéder alarmes sans login	Accès refusé ; OK après login opérateur1

Chapitre 6

Résultats et compétences

6.1 Livrables

- Programme API (Machine Expert Basic) : logique pompes, sécurités, alternance, compteurs temps.
- Projet IHM (Vijeo Designer Basic) : synoptique, animations, alarmes, navigation, gestion utilisateurs.
- Dossier de tests : scénarios et validations.

6.2 Compétences mobilisées

- Analyse fonctionnelle (commande, états, défauts, sécurité).
- Programmation d'automate (mémorisation, priorités, temporisation/compteurs).
- Gestion d'E/S TOR et analogiques (lecture niveau, seuils).
- Stratégie d'exploitation multi-pompes (alternance, équilibrage par temps de marche).
- Mise en place supervision : communication Modbus TCP/IP, import tags, animation, alarmes.
- Sécurisation de l'IHM : utilisateurs, niveaux d'accès, login/logout.

Conclusion

Le projet met en oeuvre une automatisation complète d'une station de pompage, depuis la commande locale jusqu'à la supervision. Les fonctions de sécurité (défaut, arrêt d'urgence, marche à sec) assurent la protection des équipements et des personnes. L'extension multi-pompes, combinée au suivi des temps de marche et à la mesure de niveau analogique, améliore la disponibilité et l'équilibrage d'usure. Enfin, l'IHM apporte une supervision structurée (alarmes, navigation) et une sécurisation des accès via comptes utilisateurs.