République Algérienne Démocratique et Populaire

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Université Saad Dahleb Blida1

Faculté des Sciences

Département d'Informatique

Mémoire présenté par :

MECHTI Aicha

MEDJADJI Lamia

En vue d'obtenir le diplôme de Master

Domaine : Mathématique et Informatique

Filière : Informatique

Spécialité : Génie des Systèmes Informatiques

Sujet :Développement d'un système pour la supervision d'une cellule robotisée fortement automatisée

Soutenue le : 23/6/2015

Devant le jury :

M.Farah	présidente
M.Zahra	rapporteur
M.Hammouda	examinateurs

Dédicaces

A mes chers parents en signe de reconnaissance de l'immense bien que vous avez fait pour moi concernant mon éducation, vos conseils et vos bénédictions n'ont jamais fait défaut. Recevez à travers ce travail, toute ma gratitude et mes profonds sentiments. Que Dieu le tout puissant soit à vos côtés et vous accorde une meilleure santé et une longue vie.

A mes adorables petits neveux.

A mes frères Sidi, Mohammed, Messaoud.

A mes soeurs chéries Fadila, Farida, Hakima, Hafida et Khadidja qui m'ont beaucoup aidé et soutenu chacune à sa façon. Je vous adore.

A celle qui m'a soutenu et encouragé durant ces cinq années et durant ce projet, mon meilleur amie et mon binôme Lamia. Reçois à travers ce travail « notre travail » tout mon respect, ma gratitude et ma profonde reconnaissance.

Je ne saurai terminer sans citer mes amis : Feriel, Farah, Djalil.

Aicha

Dédicaces

A mes chers parents Mohammed et Wahiba en signe de reconnaissance de l'immense bien que vous avez fait pour moi concernant mon éducation, vos conseils et vos bénédictions n'ont jamais fait défaut. Recevez à travers ce travail, toute ma gratitude et mes profonds sentiments. Que Dieu le tout puissant soit à vos côtés et vous accorde une meilleure santé et une longue vie.

A mes adorables petits neveux Amira et Mohammed que j'aime très fort.

A mes grands frère Fethi, Ghanou et Sidali.Que j'ai une immense reconnaissance pour eux et pour les sacrifices qui m'ont offert. Je leurs souhaite tout le bonheur du monde.

A mes belles soeurs qui m'ont soutenue chacune à sa façon.

A ma grand mère , ma tante et mon oncle, que dieu vous garde pour moi. je vous adore.

A mes cousines Safia, Samia et Meriem.Que je pris dieu de leurs exaucer tout leurs vœux.

A celle qui m'a soutenu et encouragé durant ces cinq années et durant ce projet ,mon binôme Aicha. Reçois à travers ce travail « notre travail » tout mon respect, ma gratitude et ma profonde reconnaissance.

Je ne saurai terminer sans citer mes chers (es) amis (es) : Meriem, Rofaida, Asma, Baya,Oumaima, Djamila, Feriel, Farah, Ilhem, Amina, Kahina, Lamine, Dido, Moussa et Amine.

Lamia

Remerciements

Tout d'abord nous tenons à remercier Dieu le tout puissant pour la volonté et le courage qu'il nous a donné afin de pouvoir finir ce travail.

Nous tenons également, à remercier Monsieur M.Gaham responsable du projet pour avoir assuré notre encadrement au sein du centre de développement des technologies avancées (CDTA) Baba Hassen. Nous les remercions pour leurs orientation, la confiance et la patience qui ont été d'un apport considérable sans le lequel ce travail n'aurait pas abouti.

Nous témoignons nos sincères gratitudes à Madame Cherfa notre promotrice de l'université blida 1 qui a largement participé à notre encadrement et qui nous apporté une aide précieuse, à la fois par ses conseils sur le plan scientifique et sa disponibilité.

Nous tenons à exprimer nos sincères remerciements à nos professeurs qui nous ont aidé et soutenu dans la poursuite de nos études.

Nous remercions vivement Mesdames et Messieurs les membres de jury d'avoir accepté d'évaluer ce travail.

Nous remercions nos parents, nos frères et sœurs pour leurs soutient moral, leurs encouragement et leurs patience durant les étapes difficiles de nos études.

Enfin, Ces remerciement ne seraient pas complets sans une pensée sincère à mes amis (es) proches et à tous ceux qui nous ont soutenus de près ou de loin, tout au long de cette année, et qui se reconnaitront.

Medjadji et Mechti

Résumé

Pour répondre à des exigences sans cesse croissantes, les entreprises industrielles ont mis au point des méthodologies afin d'aboutir à leurs buts. La supervision industrielle, est un des outils exploités. C'est la surveillance de l'état de fonctionnement d'un procédé pour l'amener et le maintenir à son point de fonctionnement optimal. Ceci a augmenté le besoin d'un outil de visualisation des processus industriels, dans un contexte économique de productivité et de flexibilité.

Les nouvelles tendances dans le domaine industriel tels que les technologies d'identification (RFID et code barre) et les systèmes SCADA font état d'une futur intégration de la gestion de production le contrôle-commande, d'un regroupement des données de l'atelier avec celles des bureaux, de manière directe et saine. Ainsi, l'innovation des technologies web et la possibilité de les intégrer dans les systèmes de supervision industrielle sont des solutions inépuisables dont il faut tirer parti.

Note travail consiste à développer un système pour superviser une cellule robotisée fortement automatisée en exploitant de nouvelles technologies tel que Node.Js, de plus ce système est déployable sur le web et indépendant du système d'exploitation.

Mots clés : supervision industrielle, IHM, visualisation du processus industriel, RFID, code barre, SCADA, Node.Js

Abstract

To respond unceasingly increasing requirements, the company's industrialists developed methodologies in order to lead to their goals. Supervision, is one of the tools is exploited, from where the monitoring of the state operation of a process to bring it and maintain it at its point of optimal operation and the need for a tool for visualization of the industrial processes, in an economic context of productivity and flexibility, the supervision profited from a projection exceptional technology.

The new trends in the industrial field such as technologies of identification (RFID and code bar) and systems SCADA mention future an integration of the production control, of a regrouping of the data of the workshop with those of the offices, in a direct and healthy way. Thus, the innovation of technologies Web and the possibility of integrated in the industrial systems of supervision are inexhaustible solutions from which it is necessary to benefit.

Our work consists in developing a system to supervise a cell robotized strongly automated by exploiting new technologies such as Node.Js.

Key words :industrial supervisory, visualisation of industrial process, RFID, bar-code , SCADA, Node.Js.

ملخص

للاستجابة على تزايد الاحتياجات، اتخذ اصحاب المصانع منهجيّة من اجل الوصول إلى هدفهم. المراقبة الصناعية، واحدة من الوسائل التي تستعمل لمتابعة حسن سير عملية الانتاج وابقائه على نظام رتيب ودائم،و لهذاظهرت الحاجة لوسائل بيانية حديثة لتصوير العمليات الصناعية، في سياق اقتصاديّ من حيث الإنتاجيةوالمرونة،ان تقدم التكنولوجيات في الميدان الصناعي مثل وسائل التعرف على المنتوجات مثل تكنولوجيا التحقق (RFID وcodebarre) ونظام SCADA هو تنويه لمستقبل فيه دمج من الإنتاج و التحكم،يسمح بحيازة المعطيات من الورشة الى المكتب، مباشرة وبطريقة سليمة.وايضا، الإبداع من تكنولوجيا الويب وإمكانية ضمه ضمن مراقبة النظامالصناعيّ حل لا يجب اهماله.

يتألّف عملنا في تطوير نظام يقوم بمراقبة خلية صناعية باستعمال تكنولوجيا مثل Js.Node.

كلمات المفاتيح: مراقبة صناعيّة، تصوير العملية الصناعيّ, RFID ،, code barre ,SCADA ,Js.Node.

Liste des Abréviations

API : Automates Programmables Industriels

DEC: Digital Equipment Corporation

DNP3 : Distributed Network Protocol

FDI : Fault Détection and Isolation

GM : General Motors

GPAO : Gestion de la Production Assistée par Ordinateur

HF : Hautes Fréquences

HP: Hewlett-Packard Company

HTML: Hypertext Markup Language

IBM: International Business Machines Corporation

ID : Identifiant

IHM: Interface Homme-Machine

IP: Internet Protocol

ISO : International Organization for Standardization

LF: Low Frequency

MBAP : Modbus Application Protocole

MMS : Manufacturing Message Specification

MTU: Master Terminal Units

NTIC : Nouvelles Technologie de l'Information et de la Communication

OCE : Objet Commandable Elémentaire

OPC : OLE for Process Control (Object Linking and Embedding)

OS : Operating System

PAC: Contrôleurs *d'*automatisme programmables

PC : Partie Commande

Pcvue : Package logiciel pour automatismes industriels

PLC : Programmable LogicController

PO : Partie Opérative

RAM: Random Access Memory

RFID : Radio Frequency Identification

ROM: Read Only Memory

RLI : Réseaux Locaux Industriels

RTU: Remote Terminal Units

RS: RecommendedStandard

SAP : Système Automatisé de Production

SCADA: Supevisory Control And Data Acquisition

TCP: Transmission Control Protocol

UDP: User Datagram Protocol

UHF : Ultra Hautes Fréquences

WIMP: Windows, Icones, Menus and Pointing device

WYSIWYG : What You See Is What You Get

Table des figures

Sommaire

Introduction générale

Contexte général

L'industrie devient de plus en plus complexe, et les demandes en termes de sureté, robustesse, gain de productivité et de qualité ne cessent de croitre. Ce développement s'accompagne d'une évolution du processus d'automatisation.

En effet, entre les années 1950 et 1970, grâce aux progrès de l'électronique et de l'informatique, une première grande révolution technologique mondiale s'ébauche : celle de l'automatisation de la production industrielle. Cette technologie a apporté de profonds bouleversements dans la manière de concevoir et d'organiser le contrôle d'un processus.

Dans le passé, les systèmes automatisés de production ont aidé à assister l'opérateur dans des tâches de conduite automatique du processus pour améliorer la qualité des produits finis, la sécurité et le rendement des unités industrielles. L'une des évolutions essentielles dans la conception des systèmes automatisés, qu'ils soient unitaires (moteurs, machine industrielles,...) ou complexes (systèmes de production, de communication,...) concerne la prise en compte, dès les premières phases, des préoccupations relatives à leur sureté de fonctionnement.

La sécurité industrielle est une partie intégrante et indissociable de la commande, telle qu'on ne peut pas envisager une installation industrielle sans une couche qui assure la sécurité, en effet, la sécurité non seulement évite les explosions qui pourraient provoquer des dégâts matériels et humains mais aussi assure la continuité de la production, ainsi elle assure la survie de l'installation. Les techniques qui assurent la sécurité dans une installation industrielle sont : le diagnostic, la surveillance, la maintenance et la supervision, la supervision intervient plus souvent dans les milieux à haut risque tels que les installations nucléaires, chimiques etc.

La supervision est une parmi les solutions pour répondre à un des principaux objectifs dans le domaine industriels qui sont l'optimisation du temps de fonctionnement des processus de production. En effet, si le système de supervision permet aux opérateurs des salles de contrôle de déterminer précisément la cause d'une panne du processus, ceux-ci sont capables de résoudre le problème rapidement, le temps de redémarrage s'en trouve alors réduit, et le temps d'opération optimisé. Cependant, la supervision de processus complexes implique l'acquisition, la gestion, et la visualisation d'une grande quantité d'informations. De cela, la communication homme-machine est aussi un volet de la supervision ou on tient compte des problèmes ergonomiques. Le développement de logiciels de supervision efficaces est donc une question critique. De plus la supervision doit tenir en compte les technologies celle liées à l'identification des produits au sein des systèmes mais

aussi celles liées au monde du web puisque l'internet a envahi tous les secteurs y compris le secteur de l'industrie.

Il existe un environnement ou interviennent plusieurs outils de supervision, tels que, des logiciels conçus spécialement pour créer des interfaces graphique avec lesquelles l'opérateur pourra intervenir à modifier des paramètres de l'installation, ces logiciels offrent aussi des outils pour établir une communication à internet dans le but de superviser et de contrôler une installation située à des milliers de kilomètres du poste de pilotage (sur web) c'est l'environnement SCADA.

Problématique

L'émergence d'un nombre de technologies clés pour l'automatisation des cellules robotisées de production. Parmi ces technologies, l'identification des produits tels que les technologies RFID et code à barre. Également l'introduction des technologies web au sein des industries pour pouvoir superviser le déroulement d'une cellule automatisée.

La problématique est de pouvoir développer un système de supervision (IHM)qui tient en compte un protocole de communication sur le web et de réaliser une architecture Client/ Serveur indépendante du système d'exploitation et déployable aussi sur le web.

Objectifs du projet

Le travail de recherche réalisé durant ces 6 mois et mené au sein du centre du développement des technologies avancées a eu pour but de développer un système de supervision industrielle pour une cellule robotisée fortement automatisée, et montrer la faisabilité technique de l'introduction des deux technologies RFID et la technologie web.

Plan du mémoire

L'organisation du présent mémoire suit la démarche adoptée lors de la réalisation de notre travail :

Nous rappelons dans le premier chapitre quelques généralités sur les systèmes de production automatisés. Ensuite nous dressons un état de l'art en matière de la supervision et l'environnement SCADA, nous définissons les briques de base des systèmes de production et de la supervision dans le domaine industriel.

Pour le deuxième chapitre, en premier lieu nous expliquerons l'évolution des logiciels SCADA et le passage de ces derniers au monde du web, puis nous illustrons un exemple de logiciel web SCADA (Mango). En un autre lieu, nous donnerons les technologies fondamentales que nous avons adoptées pour la mise en œuvre de notre système qui sont Node.JS, JavaScript et HTML5.

Le troisème chapitre a pour vocation de présenter la conception de notre système et cela en décrivant le langage et la méthodologie de conception utilisées.

Le quatrième chapitre présente la phase d'implémentation de notre système de supervision.

Conclusion et perspectives

Enfin, la dernière partie de ce mémoire est réservée à la conclusion, qui est un bilan de notre travail et des perspectives des travaux futurs qu'il convient d'entreprendre.

Chapitre 1

La supervision dans les systèmes de production

1.1 Introduction

Plus récemment la robotique industrielle a beaucoup évolué laissant entrevoir la réalisation de taches de plus en plus complexes et repousser encore plus loin les limites de l'intervention humaine dans le domaine de l'industrie, de l'exploration spatiale, sous-marine, et dans les zones hostiles.

Aujourd'hui, la concurrence s'est nettement intensifiée sur le marché économique international, en effet les entreprises sont dans l'obligation de chercher sans cesse à satisfaire les besoins des utilisateurs, les pôles de cette compétitivité sont centrés sur l'amélioration continue de la qualité de leurs produits et de la vente de ces derniers au prix le plus bas.

Répondant aux besoins de la qualité, de quantité de la production et de la concurrence, les industriels ont tendance à améliorer et à élargir leurs installations. Notamment la possibilité de diagnostiquer un dysfonctionnement et de fournir des conseils d'action en termes de commandes à appliquer, sachant qu'un simple défaut de fonctionnement du système n'est pas du tout toléré, pour arriver à prévoir le moindre défaut ou panne qui risquerai de provoquer un arrêt du système, une bonne maitrise des techniques de surveillance et de supervision est requise.

Le but de ce chapitre est de donner quelques généralités sur les systèmes de production automatisés en décrivant les briques de base de ce dernier, puis nous présenterons quelques aspects fondamentaux de la supervision dans un environnement SCADA.

1.2 Systèmes de production et l'automatisme dans l'industrie

1.2.1 Définition des systèmes de production

Un système de production est un processus de transformation de ressources en produits ou en services. Il est composé de l'ensemble des éléments (matériels et immatériels) qui contribuent à la production. Il comprend à l'entrée, des facteurs de production (matériels et équipements, main d'œuvre) et, dans certains cas, des matières premières à transformer. Au cœur d'un système de production se trouve un processus de transformation industrielle, ou une succession d'activités dont le résultat est immatériel et que l'on appelle prestation de services. A la sortie d'un système de production, on a les biens et les services produits. Les systèmes de production se distinguent les uns des autres par la combinaison des facteurs, le déroulement du processus et la nature des produits à réaliser [18].

1.2.2 Evolution des systèmes de production

L'évolution des systèmes de production est étroitement liée au développement des technologies informatiques. La robotisation de la production a permis de remplacer l'homme pour les tâches les plus répétitives, les plus pénibles et les plus dangereuses, que ce soient pour les produits ou les services. La robotisation permet d'obtenir une plus grande précision (ex : micro-informatique) et une plus grande constance du niveau de qualité. Elle a augmenté le niveau de qualification de certains emplois (pilotage des robots, ...). L'informatisation de la production s'est poursuivie avec le transfert d'informations entre les robots pour organiser, gérer et suivre la production en cours. La GPAO (gestion de la production assistée par ordinateur) permet de rationaliser et d'optimiser les ressources et l'ensemble des tâches nécessaires à la production. De la conception du produit à sa réalisation, les informations nécessaires sont gérées par un ordinateur.

Les ateliers flexibles ont été conçus pour permettre très facilement de fabriquer des pièces différentes avec un outillage commun à plusieurs produits. Ils requièrent un personnel polyvalent et autonome. Le travail est plus enrichissant, car les salariés peuvent effectuer des opérations de conception, de dépannage, de maintenance et de contrôle de la qualité [24].

1.2.3 Automatisation des systèmes de production

L'automatisme est un sous-ensemble d'une machine, destinée à remplacer l'action de l'être humain dans des taches en générales simples et répétitives, réclamant précision et rigueur. On est passé d'un système dit manuel, à un système mécanisé, puis au système automatisé [29].

Depuis que les automatismes sont réalisés sur la base d'unités de traitement (automates programmables), les architectures ont fortement évolué et sont passées par différents stades pour arriver aux architectures actuelles, basées sur l'adoption des grands standards de communication et sur l'arrivée des NTIC (Nouvelles Technologie de l'information et de la communication). Il existe deux types d'automatismes, centralisés et décentralisés.

Les automatismes centralisés

Jusque dans les années 80, les automatismes, s'appuyant sur des automates programmables industriels (API), traitaient essentiellement des fonctions séquentielles. Par la suite, les API ont été amenés à gérer de nombreuses fonctions complémentaires comme des fonctions métier, des fonctions de diagnostic système et application, etc.

Les automatismes centralisés illustré dans la figure 1 géraient tout un ensemble de fonctions qui n'avaient pas forcement d'interactions entre elles. Ces automatismes centralisés amenaient des nombreuses contraintes

- Mise en service et maintenance lourdes et difficiles à effectuer du fait de la quantité d'E/S gérées ;
- Arrêt de l'ensemble des fonctions gérées par l'API en cas de défaut système de cet API ou d'arrêt pour la maintenance du moindre élément de l'outil de production[23].

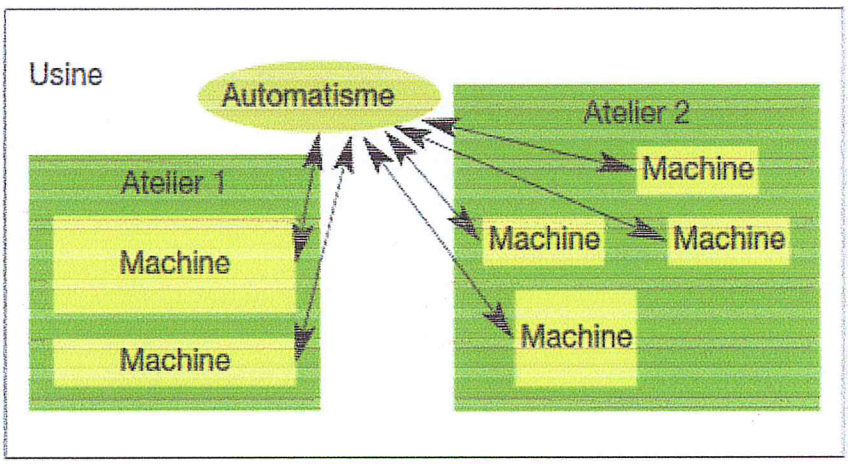


FIGURE 1 – *Architecture d'un automatisme centralisé [23]* .

Les automatismes décentralisés

Du fait des contraintes imposées par les systèmes centralisés, les utilisateurs se sont orientés vers une segmentation de l'architecture. Celle-ci a été faite en découpant l'automatisme en entités fonctionnelles. Elle permet de simplifier les automatismes en réduisant le nombre d'E/S gérées et présente donc l'avantage de faciliter la mise en service et la maintenance. Cette segmentation a généré le besoin de communication entre les entités fonctionnelles.

Les constructeurs d'API ont donc créé des offres de réseaux locaux industriels (RLI) afin d'assurer une communication efficace entre les différents API[23].La figure 2 explique les automatismes décentralisés.

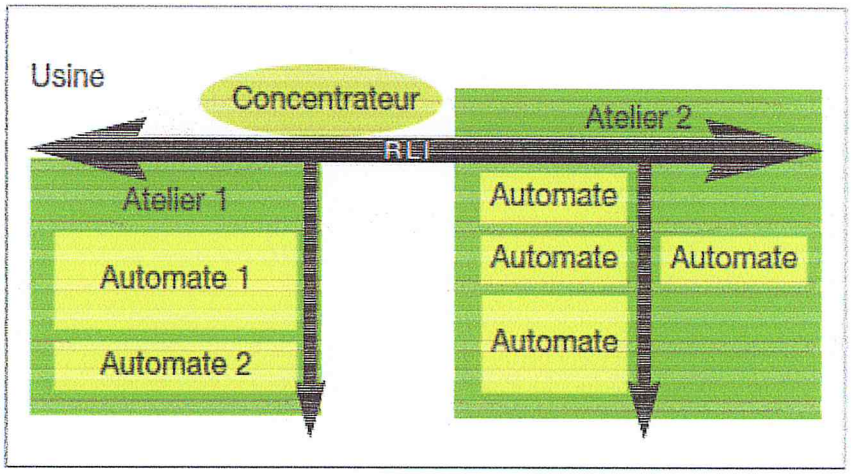


FIGURE 2 – *Architecture d'un automatisme décentralisé [23]* .

Rôle de l'informatique dans les automatismes

Le besoin d'assurer une communication entre les mondes de l'informatique et de l'automatisme est devenu indispensable du fait de la nécessité d'augmenter la productivité (fiabilité, pérennité...) des usines de fabrication. Les réseaux locaux industriels d'automatismes propriétaires ont rapidement bénéficié des grands standards développés sur les architectures informatiques. La communication entre ces deux mondes a d'abord été assurée par des liaisons séric (RS), puis par des produits issus de partenariats entre les constructeurs d'automates programmables et les grands de l'informatique tels que IBM, HP, DEC. Ces derniers ont proposé, dans leur catalogue, une offre de coupleurs RLI (Réseaux Locaux Industriels) comme Ethway ou Uni-Telway. Avec la standardisation progressive d'Ethernet dans les deux mondes et une tentative de standardisation d'un protocole MMS (Manufacturing Message Specification) comportant des services communs aux deux mondes, la frontière entre ces deux mondes devait tomber ! Quelques applications basées sur "cet espéranto automatisme/informatique" ont été réalisées, mais très vite, ce standard MMS n'a plus été utilisé de par sa complexité [23] .

1.2.4 Description d'un SAP (système de production automatisé)

Un système automatisé de production est un ensemble d'éléments en interaction, organisés dans un but précis : agir sur une matière d'œuvre afin de lui donner une valeur ajoutée d'une façon reproductible et rentable.

Tous les systèmes automatisés possèdent une structure générale composée de 3 parties fondamentales comme il est illustré dans la figure 3 :

- **Une partie opérative (PO) :** que l'on appelle également partie puissance, c'est la partie visible du système (corps) qui permet de transformer la matière d'œuvre entrante, elle est composée d'éléments mécaniques, d'actionneurs (vérins, moteurs), de préactionneurs (distributeurs et contacteurs) et des éléments de détection (capteurs, détecteurs)[37] ;
- **Une Partie Commande (PC) :** coordonnant la succession des actions sur la partie opérative avec la finalité d'obtenir cette valeur ajoutée. Cette partie de commande élabore les ordres transmis aux actionneurs à partir des informations fournies par la machine au moyen d'interrupteurs de position, thermostats et autres dispositifs appelés capteurs. La partie commande reçoit également des informations transmises par un opérateur en fonctionnement normal, ou un dépanneur en cas de réglage ou de mauvais fonctionnement de la partie commande ou de la partie opérative[30].

  Entre la partie commande et l'homme se trouve la partie dialogue qui permet à ce dernier de transmettre des informations au moyen de dispositifs adaptés (boutons

poussoirs, commutateurs... etc).

- **Un pupitre :** permet d'intervenir sur le système (marche, arrêt, arrêt d'urgence...) et de visualiser son état (voyants)[37].

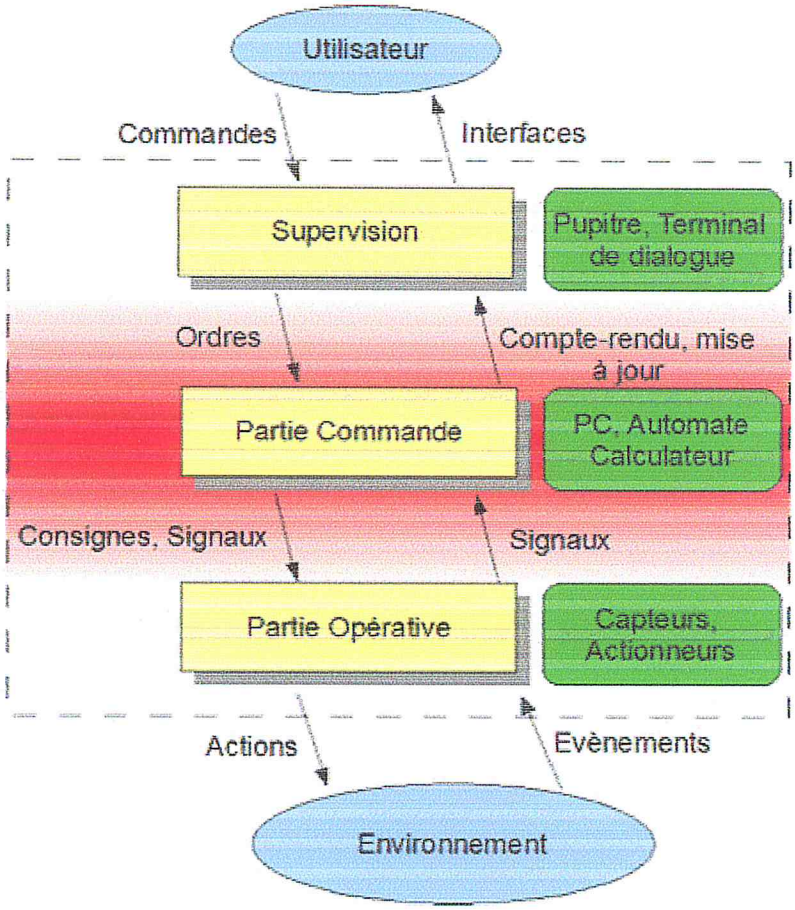


FIGURE 3 – *Vue simplifier d'un système automatisé [11] .*

1.3 Automate programmable industriel (API) l'élément essentiel de l'architecture

Un automate programmable est un dispositif électronique programmable structuré autour d'une unité de calcul et composé de modules tels que des entrées sorties analogiques et numériques et des modules de communication (bus de terrain ModBus et LonWorks)[8].

Les automates programmables industriels sont apparus dans les années soixante, à la demande de l'industrie automobile américaine (GM General Motors), qui réclamait plus d'adaptabilité de leurs systèmes de commandes. En 1968, sont apparus les premiers dispositifs de commande logique aisément modifiable : Les PLC (Programmable LogicController) par Allen Bradley, Modicom et Digital Equipement. Le premier dispositif français était le PB6 de Merlin Gerin en 1973 [7].

1.3.1 Structure d'un API

Elle ressemble à celle d'un micro-ordinateur, constitué d'une unité centrale (unité de traitement), des coupleurs, des modules d'entrées (interface d'E) et des modules de sortie (interfaces de S)

- **Le processeur :** Son rôle consiste d'une part à organiser les différentes relations entre la zone mémoire et les interfaces d'entrée/sortie et d'autres part à gérer les instructions du programme.
- **La mémoire :** Elle est conçue pour recevoir, gérer, stocker des informations issues des différents secteurs du système que sont le terminal de programmation (PC ou console) et le processeur, qui lui gère et exécute le programme. Elle reçoit également des informations en provenance des capteurs ;
- **Les interfaces :** il existe des interfaces d'entrées et celles de sorties :
  - L'interface d'Entrées comporte des adresses d'entrée,une pour chaque capteurrelié ;
  - L'interface de Sorties comporte des adresses de sortie, une pour chaque actionneur.

  L'alimentation : Tous les automates actuels utilisent un bloc d'alimentation alimenté 240 Vcc et délivrant une tension de 24 Vcc ;
- **Coupleurs :**ce sont des cartes électroniques qui assurent la communication entre les périphériques (modules d'E/S ou autres) et l'unité centrale [34].

La figure ci-dessous montre la structure d'un API

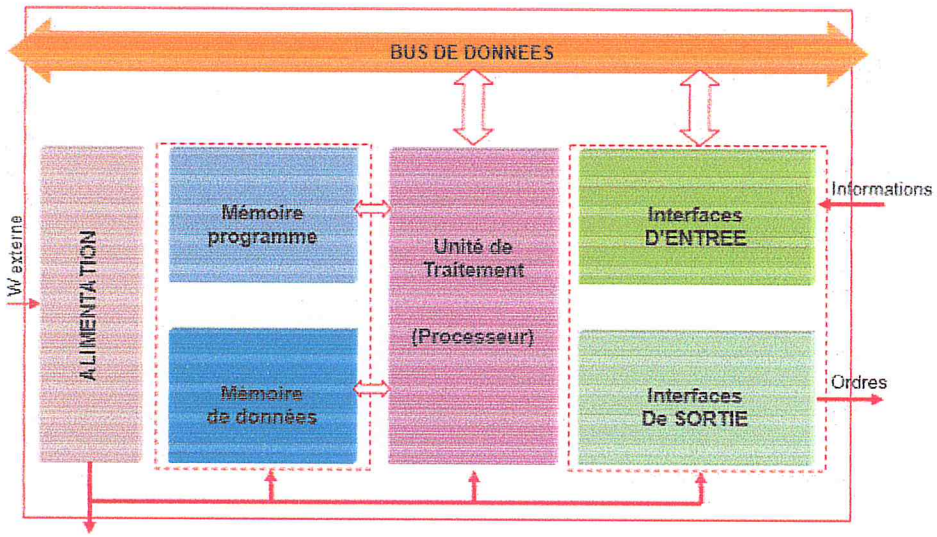


FIGURE 4 *Structure d'un API [27] .*

1.3.2 Comportement des Automates Programmables Industriels

Le moniteur d'exécution d'un API peut être composé de plusieurs sous-programmes appelés tâches. Une tâche est un ensemble d'opérations programmées pour s'exécuter successivement, puis s'arrêter jusqu'au prochain lancement. Dans un automate programmable industriel l'exécution d'une tâche est un cycle composé de trois phases (figure 5) :

- **L'acquisition des entrées :**les variables d'entrées sont accessibles en lecture seule. Pendant cette première phase, leurs valeurs sont lues et ensuite stockées dans la mémoire de l'API ;
- **Le traitement interne :** c'est une phase d'exécution du programme et de calcul des valeurs de sorties à partir des valeurs stockées en mémoire dans la phase précédente, les résultats des calculs sont ensuite à leur tour stockés en mémoire ;

L'affectation des sorties :les variables de sorties sont accessibles en écriture seule. Pendant cette phase, leurs valeurs sont mises à jour à partir des valeurs calculées dans la phase de traitement interne [38].

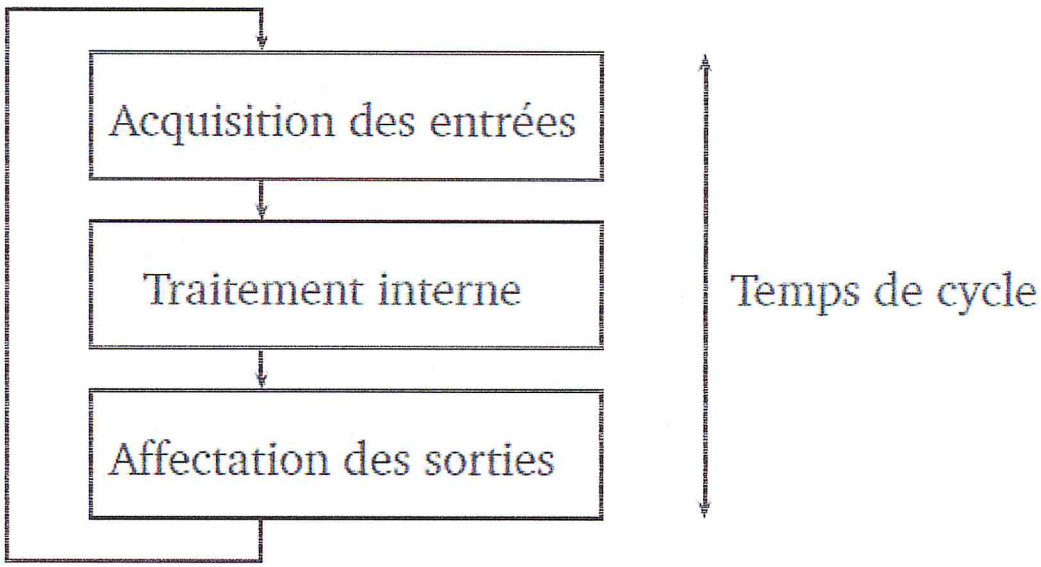


FIGURE 5 – *Cycle d'une tache [38].*

1.3.3 Intégration des robots dans les systèmes automatisés

Le concept de robot date de plusieurs siècles, mais le terme robot fut inventé par tchèque Karle Capek dans une pièce de théâtre écrite en 1920 .Ce terme est dérivé du verbe tchèque « robota » signifiant travail forcé ou corvée pour désigner des machines automatiques appelées à remplacer les hommes dans les taches les plus pénible.

Aujourd'hui, le mot robot est employé aussi bien pour désigner le plus simple des manipulateurs, et même des appareils ménagers, que la machine la plus sophistiquée. C'est pour cette raison que l'on a introduit la notion de robot industriel aux Etats-Unis vers 1960 [13].

ISO 8373 a défini le Robot manipulateur industriel comme une machine, un mécanisme constitué normalement d'une série de segments qui sont reliés par un joint assurant une rotation ou une translation relative entre segments, dont le but est de prendre et déplacer des objets (pièces ou outils) avec plusieurs degrés de liberté. Il peut être commandé par un opérateur, une unité de commande électronique ou un système logique (dispositif à cames relais câbles etc.)[35].

De nos jours, les robots industriels sont beaucoup utilisés dans les usines. En 2011, 5000 robots FANUC fabriqués chaque mois. Récemment, en 2014 environ 330 000 robots FANUC vendus à travers le monde. Ces robots peuvent être programmés de différentes façons et donc faire des tâches répétitives ou variées. Ils peuvent reproduire les opérations autrefois effectuées par l'homme [19].

1.3.4 Le protocole modbus

Modbus est un protocole industriel qui a été développé en 1979 par Modicon (Aujourd'hui intégré dans le groupe Schneider Electric) afin de permettre les communications entre des appareils d'automatisation. Initialement mis en place comme protocole d'application ayant pour but de transférer des données via une couche série, sa première implémentation basée sur le RS-232).

Modbus s'est étendu et inclus désormais des implémentations série, TCP/IP (1999) ainsi que le protocole UDP (User Datagram Protocol)

Définition

Modbus est un protocole de requête-réponse mis en œuvre en utilisant une relation maître-esclave. Dans une relation maître-esclave, la communication a toujours lieu par paires (un appareil doit initier une requête puis attendre une réponse), et l'appareil qui initie, le maître, est en charge d'initier toutes les interactions. En règle générale, le maître est une IHM (interface homme-machine) ou un superviseur de contrôle et d'acquisition de données (SCADA), et l'esclave est un capteur, un automate programmable industriel (PLC) ou un contrôleur d'automatisme programmable (PAC). Le contenu de ces requêtes et réponses ainsi que les couches réseau à travers lesquelles ces messages sont envoyés sont définis par les différentes couches du protocole [2].

Accès aux données dans Modbus et le modèle de données Modbus

En général, les données accessibles par Modbus sont stockées dans l'une des quatre banques de données ou gammes d'adresse : bobines, entrées discrètes, registres de maintien et registres d'entrée. Comme c'est le cas de la plupart des spécifications, les noms peuvent varier selon l'industrie ou l'application. Par exemple, les registres de maintien peuvent être appelés "registres de sortie", et les bobines peuvent être appelées "sorties numériques ou discrètes". Ces banques de données définissent le type et les droits d'accès des données contenues. Les appareils esclaves disposent d'un accès direct à ces données, qui sont hébergées localement sur ces appareils. Les données accessibles par Modbus sont généralement un sous-ensemble de la mémoire principale de l'appareil. Par contraste, les

maîtres Modbus doivent demander un accès à ces données par le biais de divers codes de fonction (figure 6)[2].

Function name	Function code
Read Discrete Inputs	2
Read Coils	1
Write Single Coil	5
Write Multiple Coils	15
Read Input Registers	4
Read Holding Registers	3
Write Single Holding Register	6
Write Multiple Holding Registers	16
Read/Write Multiple Registers	23
Mask Write Register	22
Read FIFO Queue	24
Read File Record	20
Write File Record	21
Read Exception Status	7
Diagnostic	8
Get Com Event Counter	11
Get Com Event Log	12
Report Slave ID	17
Read Device Identification	43
Encapsulated Interface Transport	43

FIGURE 6 – *Les codes fonction modbus [2]*

Formats standard : Les formats standards les plus connues sont le TCP, les terminaux distants (RTU).le RTU est utilisé à travers une ligne série, alors que le TCP est utilisé à travers des réseaux TCP/IP ou UDP/IP.

Le modbus RTU

Modbus RTU est un protocole point à point qui utilise une liaison série (RS 232, RS 485, RS 422 ...) comme support de transmission des données.

Lors de la communication, le modbus RTU utilise les silences dans la ligne de transmission pour indiquer le début et la fin du message. Le Silence est considéré comme un temps égal ou supérieur à 3,5 octets nécessaires pour transmettre. Pour chaque vitesse de transmission correspond un temps de silence spécifique. Une fois la transmission d'un message réalisé, ne peut pas commencer la transmission d'un autre message jusqu'à le que temps nécessaire de silence soit écoulé (3,5 fois le temps de transmission d'un octet) [17].

La trame modbus RTU est définie de la manière suivante (figure 7) :

START	Adresse	Fonction	Données	CRC	END
Silence	1 octet	1 octet	n octets	2 octets	Silence

N° esclave	Code fonction	1er paramètre		Autres paramètres	CRC16	
1 octet	1 octet	PF : 1 octet	Pf : 1 octet	N octets	PF : 1 octet	Pf : 1 octet

FIGURE 7 – *Format de trame RTU [28].*

Nunméro esclave : de 1 à 247 ;

1re paramètre : Adresse du bit ou du mot adressé ;

2ème paramètre : Quantité de mots adressés ou valeur du bit ou du mot écrit selon la fonction utilisée ;

Autres paramètres : Données écrites dans plusieurs mots consécutifs ;

CRC16 : Contrôle par redondance cyclique pour détecter les erreurs de transmission. La détection de fin de trame est réalisée sur un silence supérieur ou égal à 3 caractères [28].

Le modbus TCP/IP

Le protocole de communication Modbus TCP/IP utilise l'Ethernet comme norme de transmission. Il offre les mêmes fonctionnalités que le protocole Modbus RTU, il permet en plus des architectures multi-maître (figure 8).

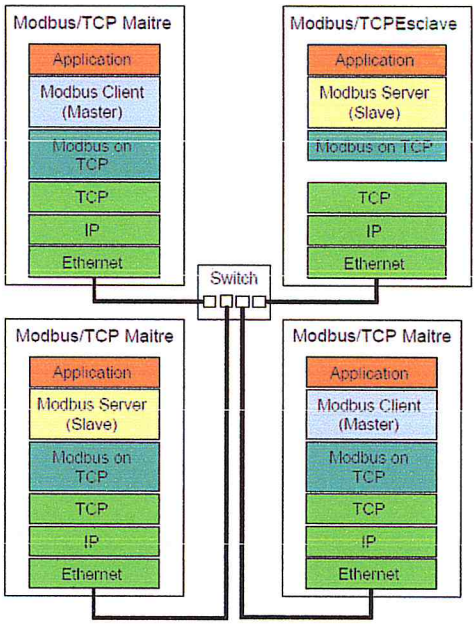


FIGURE 8 – *Communication maitre esclave avec modbus tcp/ip [50] .*

L'esclave Modbus peut servir différents Maîtres simultanément. Chaque communication Maître/Esclave utilise une connexion TCP séparée (Port 502). L'adresse IP est utilisée comme ID de composant. Le maître initialise les connexions et utilise le Polling. Les switchs et les routeurs standards peuvent être utilisés [50].

L'utilisation du protocole Modbus/TCP et du port Ethernet présente des avantages : vitesse élevée et possibilité d'atteindre des appareils répartis sur le réseau d'entreprise. Le protocole Modbus/TCP est standardisé : une trame Modbus est empaquetée dans un segment TCP ("tunnel") et transmise par Ethernet.

La trame modbus TCP/IP est définie de la manière suivante :

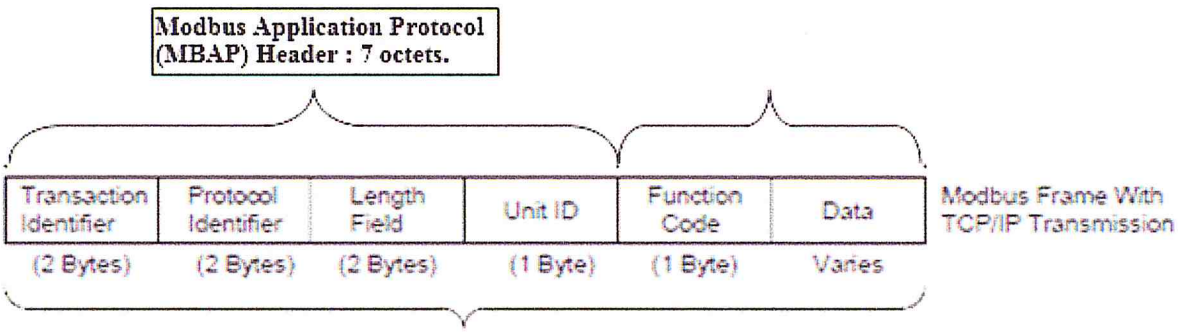


FIGURE 9 – *Trame modbus TCP/IP [21]*

L'entête MBAP (7 octets) contient les champs suivants :

Champ	*Octet*	*Description*	*Client (Master)*	*Server (Slave)*
Transaction Identifier	2	Numéro de transaction	Initialisé par le Client	Renvoyé par le Server
Protocol Identifier	2	0 = Protocole MODBUS	Initialisé par le Client	Renvoyé par le Server
Length	2	Numéro de Bytes qui suivent	Renvoyé par le Server	Renvoyé par le Server
Unit Identifier	1	Toujours à 255 ou Unit ID de l'afficheur	Renvoyé par le Server	Renvoyé par le Server

FIGURE 10 – *Entête MBAP du Protocole ModBus-TCP [21].*

1.4 La supervision dans les systèmes industriels

1.4.1 Généralité sur la supervision

Définition opératoire du concept

Etymologiquement : le mot "supervision" se décompose en deux mots latins "supra" et "videre", ce qui veut dire "regarder d'en haut". Il s'en dégage deux notions fondamentales : la notion de surveillance pour que les choses soient bien faites et la position même de celui qui est chargé de cette surveillance.

La supervision est avant tout le regard que porte un responsable (superviseur) sur son collaborateur (supervisé) pour s'assurer que les tâches qui ont été assignées à ce dernier sont exécutées conformément aux normes et instructions mise en place [16].

Supervision des systèmes informatiques

Les systèmes informatiques étant de plus en plus complexes, leur surveillance et la localisation des problèmes deviennent de plus en plus ardus pour l'administrateur réseaux et systèmes. La pression est d'autant plus forte que les entreprises ou organismes s'appuient sur le système d'information pour leur activité, demandant ainsi une très grande réactivité de la part de l'administrateur. Il lui faut donc automatiser la surveillance de son système.

La supervision désigne un ensemble de concepts recouvrant la surveillance du bon fonctionnement d'un système informatique (matériel, services, applicatifs) en production. On distingue trois types :

- **Supervision système.** La supervision système porte principalement sur les trois types principaux de ressources système : processeur, mémoire et stockage.
- **Supervision réseau.** La supervision réseau porte sur la surveillance de manière continue de la disponibilité des services en ligne - du fonctionnement, des débits, de la sécurité mais également du contrôle des flux.
- **Supervision des applications.** La supervision des applications (ou supervision applicative) permet de connaître la disponibilité des machines en terme de services rendus en testant les applications hébergées par les serveurs [36].

Supervision industrielle :

La supervision est une technique de pilotage et de suivi informatique de procédés industriels automatisés. Elle concerne l'acquisition de données (mesures, alarmes, retours d'état de fonctionnement) et de paramètres de commande des processus généralement confiés à des automates programmables.

Les systèmes de supervision industrielle offrent ainsi un ensemble de moyens utilisés pour gérer un procédé aussi bien en situation normale qu'anormale et répond aux préoccupations suivantes :

- **Technique** : pilotage de l'infrastructure et des machines, surveillance du réseau et mesure des performances ;
- **Applicative** : surveillance des applications et des processus métiers... [6] ; Contrat de service :surveillance respect des indicateurs ;
- **Métier** : surveillance des processus métier de l'entreprise [12] ;

Terminologies utilisées dans la supervision

- **Suivi** : Fonction maintenant en permanence un historique des traitements effectués par le système de commande/supervision et une trace des événements que perçoit le système [45] ;
- **Erreur** : Ecart entre une valeur mesurée ou estimée d'une variable et la vraie valeur spécifiée par un capteur étalon ou jugée (par un modèle) théoriquement correcte [10] ;
- **Erreur latente** : l'erreur est latente tant que la partie erronée du système n'est pas sollicitée. Elle devient effective au moment de la sollicitation de la partie erronée [15] ;
- **Dégradation** : une dégradation est l'état d'un composant présentant une perte de performance dans une ou plusieurs de ces fonctions pour lesquels est conçu [9] ;

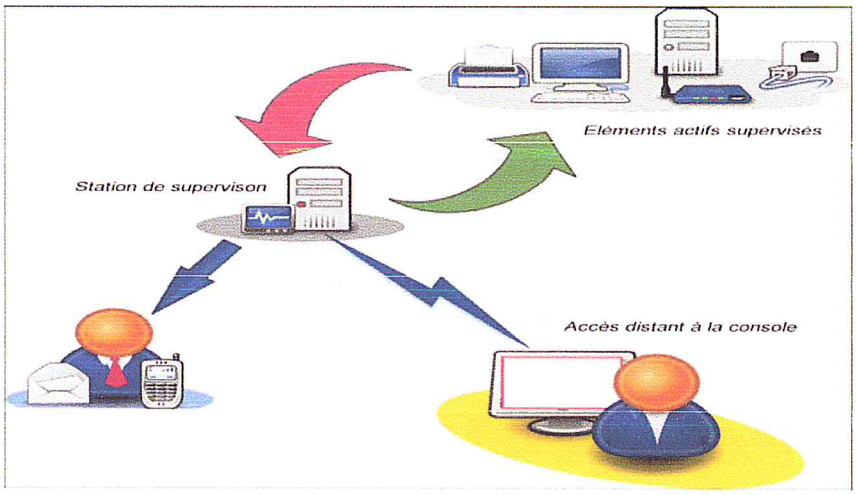


FIGURE 11 – *Schéma d'un système de supervision [39]*

- **Défaut ou faute** : Déviation d'une variable observée ou d'un paramètre calculé par rapport à sa valeur fixée dans les caractéristiques attendues du processus lui-même, des capteurs, des actionneurs ou de tout autre équipement ;
- **Défaillance** : Modification suffisante et permanente des caractéristiques physiques d'un composant pour qu'une fonction requise ne puisse plus être assurée dans les conditions fixées [10] ;
- **Dysfonctionnement** : exécution d'une fonction du système au cours de laquelle le service rendu n'est pas délivré ou est délivré de manière incomplète [15] ;
- **Modèle** : Un modèle est une formalisation mathématique d'un système physique qui permet de représenter les liens (ou relation de contraintes), existants entre des quantités (ou variables) données du système [14] ;
- **Résidus ou indicateur de faute** : Un résidu est un signal qui reflète la cohérence (ou la consistance) des données mesurées vis-à-vis d'un modèle comportemental du système [15] ;
- **Panne** : Interruption permanente de la capacité du système à réaliser sa fonction requise [10] ;
- **Contrainte** : Limitations imposées par la nature (lois physiques) ou l'opérateur [10] ;
- **Monitoring** : c'est une technique de surveillance qui permet de suivre l'avancement d'un programme pendant l'action, d'identifier les problèmes au fur à mesure qu'ils se posent, de choisir et de mettre en œuvre des actions correctrices afin d'assurer le bon déroulement des activités [16].

1.4.2 Objectifs et rôles de la supervision

La supervision a pour but de contrôler l'exécution d'une opération ou d'un travail effectué par d'autres sans rentrer dans les détails de cette exécution. La supervision recouvre l'aspect fonctionnement normal et anormal :

- en fonctionnement normal : son rôle est surtout de prendre en temps réel les dernières décisions correspondant aux degrés de liberté exigés par la flexibilité décisionnelle.
- en présence de défaillance : la supervision va prendre toutes les décisions nécessaires pour le retour vers un fonctionnement normal. Après avoir déterminé un nouveau fonctionnement, il peut s'agir de choisir une solution curative, d'effectuer des ré ordonnancements "locaux", de prendre en compte la stratégie de surveillance de l'entreprise, de déclencher des procédures d'urgence, etc [45].

De manière générale les objectifs de supervision sont :

- **En ce qui concerne le superviseur, il s'agit de :**
  - Se rendre compte de la manière dont les tâches sont exécutées ;
  - Avoir une meilleure connaissance des conditions de travail du supervisé ;
  - Identifier les difficultés et aider le personnel à les résoudre ;
  - Permettre de mieux apprécier le personnel et la qualité du travail qu'il exécute ;
  - S'assurer de l'exécution correcte des tâches ;
  - Assurer que les objectifs et activités poursuivis par le personnel correspondent aux besoins de la population cible ;
  - Motiver le personnel [16].
- **En ce qui concerne le supervisé, il s'agit de :**
  - Se sentir responsabilisé, valorisé, assisté et appartenant à une équipe ;
  - Avoir la possibilité de discuter de ses problèmes avec le superviseur ;
  - Etre formé pour accomplir correctement ses tâches [16].
- **Au niveau du service, la supervision vise à :**
  - Améliorer la qualité des services et des soins ;
  - Accroître l'efficience ;
  - Corriger les erreurs à temps ;
  - Permettre une meilleure organisation ;
  - Améliorer l'accessibilité des services ;
  - Créer un climat de bonne entente, de franche collaboration et de bonne communication [16].

1.4.3 L'Architecture de la supervision

La supervision est d'un niveau supérieur et qui superpose à la boucle de commande, elle assure les conditions d'opérations pour lesquelles les algorithmes d'estimation et de commande ont été conçus. Parmi les taches principales de la supervision se trouve la surveillance, l'aide à la décision, le diagnostic et la détection [9]. L'architecture d'un système de surveillance est illustrée sur la figure suivante :

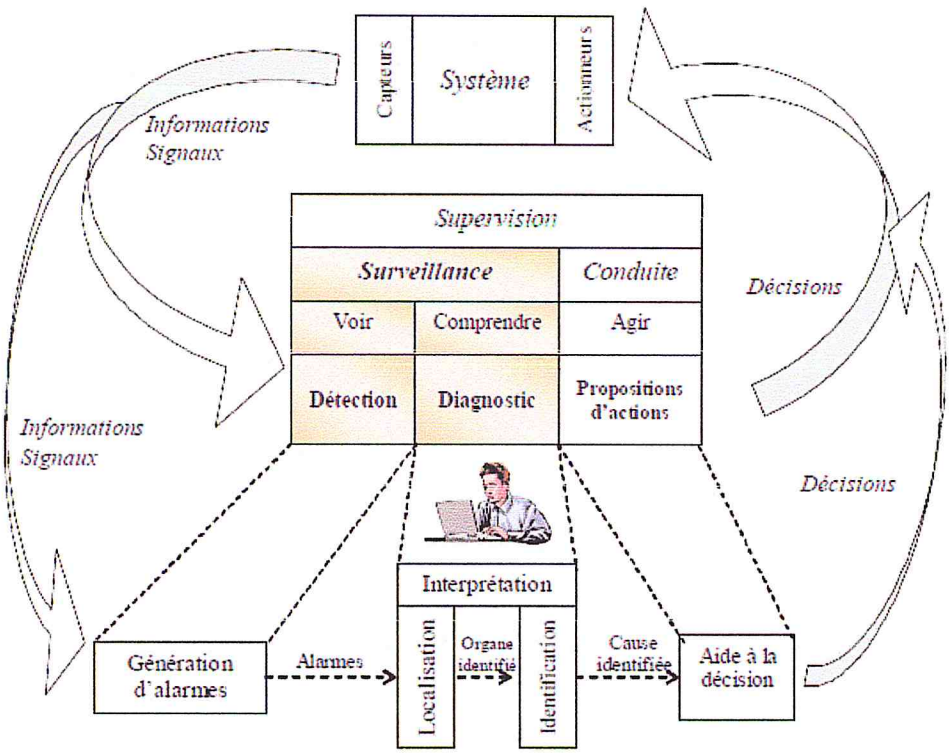


FIGURE 12 – *Architecture générale d'un système de Supervision en ligne [41].*

1.4.4 Les Techniques de la supervision

Pour concevoir un système de supervision on a besoin de maitriser les techniques suivantes :

1. **Acquisition de données** : l'acquisition de données est la première étape de la supervision, tel que, elle consiste à recueillir, à valider, et à assurer l'acheminement des informations sur l'état de système jusqu'au poste de pilotage, cette tâche est exécuter sans interruption et à chaque instant, ces opérations impliquent l'utilisation de capteurs permettant de mesurer les différentes variables du processus. Ces informations seront utilisées dans des relations de résidus pour accomplir l'étape de détection [9].
2. **Surveillance** : toute prise de décision est basée sur les informations acquises des ressources et de l'historique des activités exécutées. La surveillance est responsable de l'acquisition des signaux en provenance des ressources et de la commande. Ces informations sont utilisées pour la reconstitution de l'état réel du système commandé, et pour faire les inférences nécessaires afin de produire les données utilisées pour reconstituer l'état réel du système ou pour dresser des historiques de fonctionnement [45]. Il existe deux types de surveillance : la surveillance du système opérant et la surveillance de la commande, la figure suivante représente les types de surveillance.

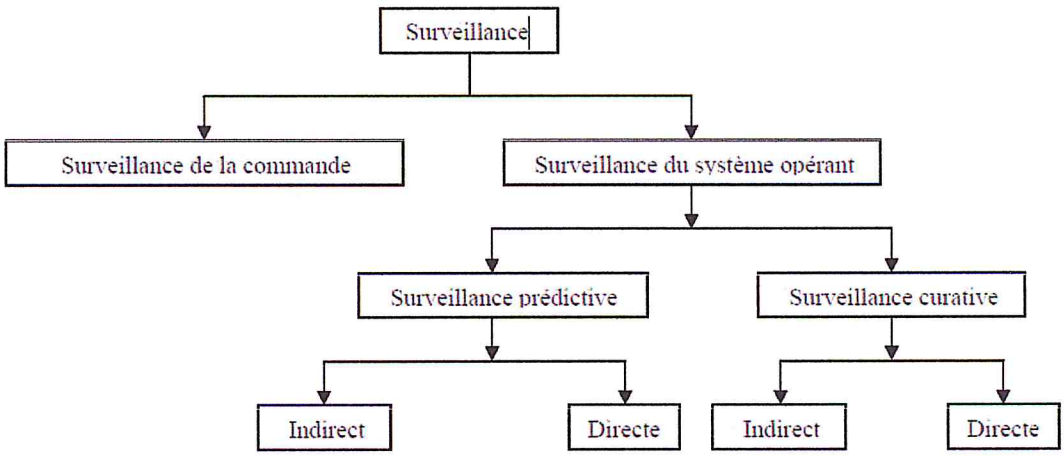


FIGURE 13 – *Les types de surveillance [9].*

- **Surveillance de la commande** :

  Basée sur la notion de filtre de commande, elle permet de vérifier que les ordres amis sont conformes à l'état de la partie opérative .le concept Objet Commandable Elémentaire (OCE) a été développé pour la conception de ces filtres de commande

- **Surveillance du système opérant** : elle a en charge la surveillance des défaillances du procédé qui,dans le cadre de la sureté de fonctionnement ,sont classées en deux catégories :
- Les défaillances cataleptiques ;
- Les défaillances progressives.
  - **Surveillance prédictive** : Comme pour la surveillance classique, la surveillance prédictive est un dispositif passif, informationnel, qui analyse l'état présent et passé du système et fournit des indicateurs sur les tendances d'évolution future du système [41] ;
  - **Surveillance curative** : elle comporte deux onctions la détection et la diagnostique.

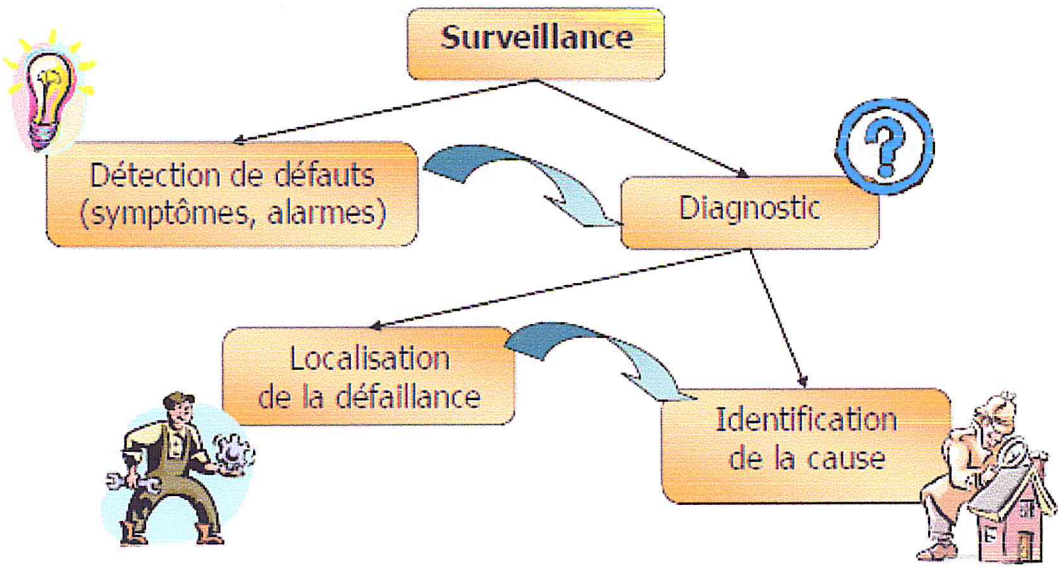


FIGURE 14 – *Composantes de la surveillance industrielle [41].*

3. **La détection** : La procédure de détection vise à déterminer l'apparition et l'instant d'occurrence d'une défaillance. Cette étape ne nécessite qu'un modèle de bon fonctionnement du système. Une panne sera détectable si au moins un résidu permet de la détecter. Lorsque le modèle permet de représenter exactement le système (aucune erreur de modélisation, connaissance de la nature des signaux inconnus agissant sur

le système, ...), les résidus générés seront strictement égaux à zéro en fonctionnement normal et différents de zéro en présence de pannes. La détection d'une défaillance se résumera alors à déclencher une alarme lorsqu'au moins un résidu différera de zéro [14].

4. **Le Diagnostic** : établit un lien de cause à effet entre un symptôme observé et la défaillance qui est survenue, ses causes et ses conséquences. On distingue classiquement trois étapes :
   - **La Localisation** : elle détermine le sous-système fonctionnel à l'origine de l'anomalie et progressivement affine cette détermination pour désigner l'organe ou dispositif élémentaire défectueux [15]. La procédure de localisation nécessite d'utiliser un ensemble (ou vecteur) de résidus. Pour permettre la localisation, le vecteur de résidus doit avoir un certain nombre de propriétés permettant de caractériser de manière unique chaque faute [14].
   - **L'identification** : détermine les causes qui ont engendré la défaillance constatée.
   - **Le Pronostic** : Le pronostic s'intéresse la propagation des défaillances. Il indique les conséquences inéluctable et mesure celles qui peuvent être anticipes de façon à ne pas solliciter les sous-systèmes de la ressource défaillante .le pronostic es une étape sans laquelle la prose décision n'est pas faisable [42].
5. **L'aide à la décision** : permet la gestion de la conduite et de la maintenance des procédés industriels nécessaires pour atteindre les degrés de sureté de fonctionnement souhaités alors les systèmes d'aide à la décision pour la supervision sont conçus pour maintenir un système avec un degré de sureté de fonctionnement élevé. Afin de maintenir ces indices aux niveaux souhaitables , ces systèmes doivent d'une part , aider les opérateurs à prendre des décisions rapidement face aux situations anormales , comme la prise de décision du démarrage ou d'arrêt du système ou des sous-systèmes ou la reconfiguration du système .D'autre part, ces systèmes doivent aider l'opérateur à prendre les décisions sur les politiques de maintenance des différents équipements [5] ;
6. **La maintenance** : la maintenance des systèmes industriels est devenue un point essentiel lors de leur conception et de leur exploitation, tant pour des questions de sécurité et de sûreté de fonctionnement, que pour des questions de rentabilité. Par exemple, un arrêt de production pour maintenance sur les chaînes de montage de chez Peugeot peut coûter jusqu'à un million de francs par jour. Une maintenance mal adaptée à un système peut également conduire à une situation critique, dangereuse aussi bien pour les personnes que pour le matériel ou l'environnement [3].

Il existe deux types de maintenance généralement utilisés dans les entreprises : la maintenance corrective et la maintenance préventive [49] la figure suivante illustre les différents types de maintenance :

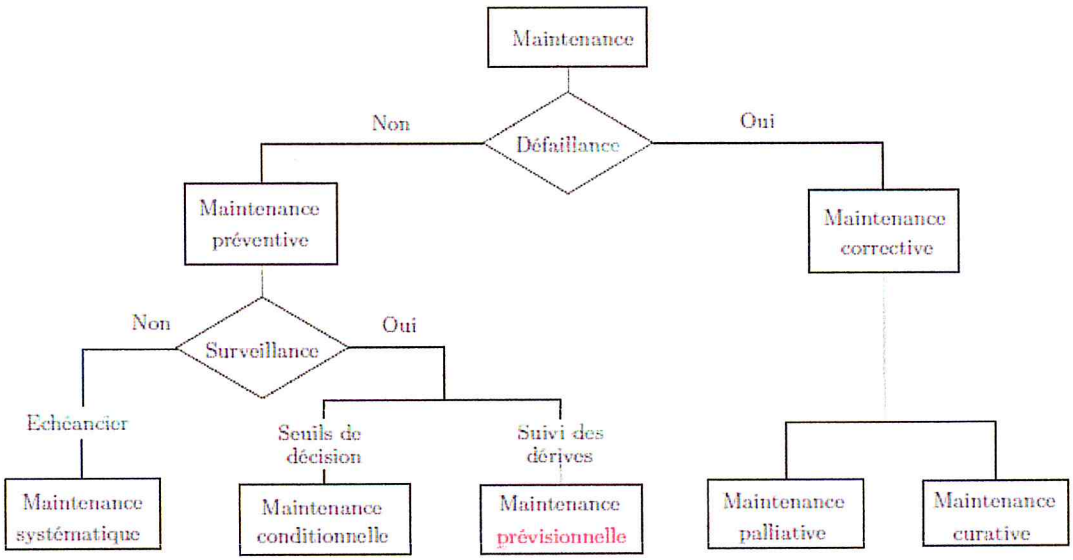


FIGURE 15 – *Types de maintenance [49] .*

- **La maintenance corrective** : La maintenance corrective regroupe l'ensemble des activités réalisées après la détection et la localisation d'une défaillance sur un procédé. Elle peut être palliative ou curative [49] ;
  - **La maintenance palliative** regroupe les actions permettant à un procédé d'accomplir provisoirement tout ou partie d'une fonction requise. Elle est appelée couramment le dépannage. Ces actions de maintenance sont principalement constituées d'actions à caractère provisoire qui devront être suivies d'actions curatives[49] ;
  - **La maintenance curative** a pour but de rechercher les causes initiales d'une défaillance et de réparer le ou les composant(s) défaillant(s), et ainsi d'éviter toute nouvelle occurrence de défaillance. Le résultat des activités réalisées doit présenter un caractère permanent [49] ;
- **La maintenance préventive** : La maintenance préventive a pour but d'améliorer la fiabilité et la disponibilité des procédés. Ainsi, elle permet de diminuer le coût de maintenance, notamment en limitant les arrêts de fonctionnement subis. La maintenance préventive comprend trois niveaux : la maintenance systématique, la maintenance conditionnelle et la maintenance prévisionnelle ou prédictive [49].

1.4.5 Analyse de méthodes de supervision

Les méthodologies de surveillance sont généralement divisées en deux groupes : méthodologies de surveillance avec modèle et sans modèle, la figure suivante présente les déférentes méthodes de surveillance

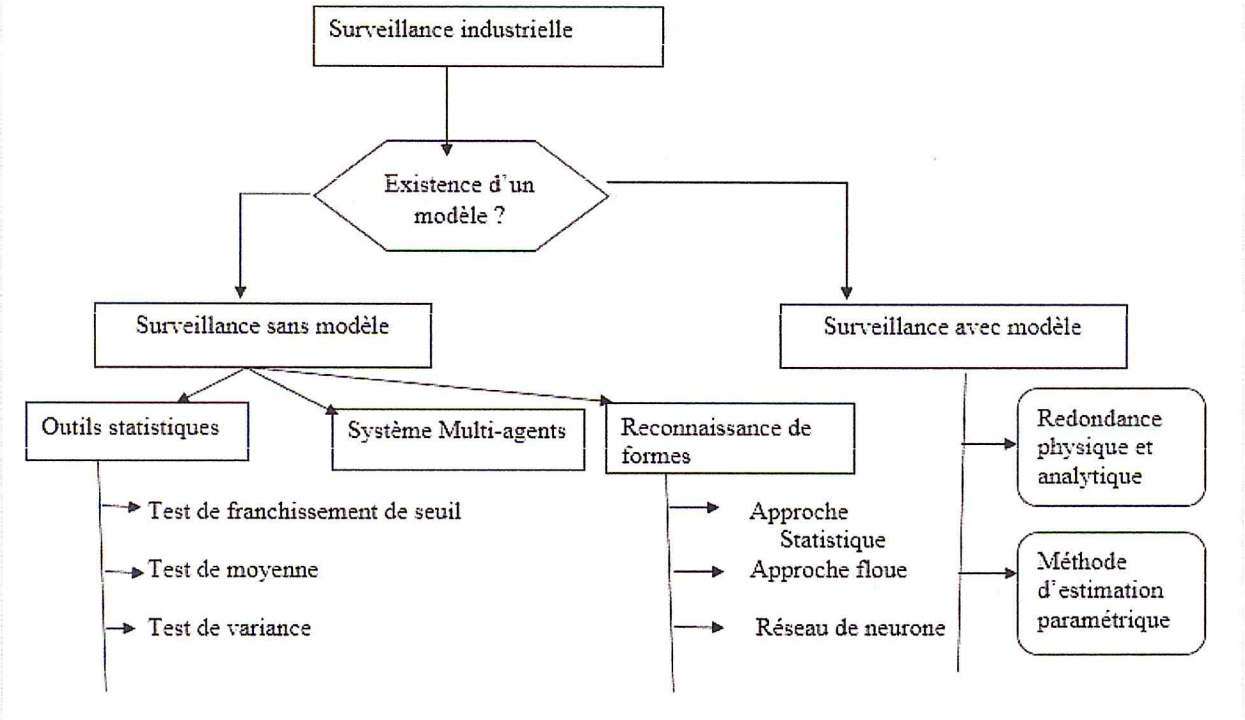


FIGURE 16 – *Méthodologie de surveillance industrielle [14].*

- **Surveillance à base de modèle :**

Les méthodes de surveillance à base de modèles ("Model-based FDI : Fault Détection and Isolation") ont été développées dès le début des années 70. Ces méthodes de surveillance utilisant un modèle reposent sur la génération et l'étude d'un signal particulier appelé "indicateur de défaut" ou "résidu". Les modèles utilisés peuvent être de nature et de complexité différente. Ils peuvent être : à temps continu ou à temps discret, qualitatifs, structurels ou analytiques, linéaires ou non linéaires, représentant le bon fonctionnement ou tenant compte des défaillances [14].

Ils peuvent être séparés en deux techniques : techniques d'estimation paramétrique et techniques de redondance physiques et analytiques[42].

- **Surveillance sans modèle :**

  Nombreuses sont les applications industrielles dont le modèle est difficile, voire impossible à obtenir suite à une complexité accrue ou à de nombreuses reconfigurations intervenants durant le processus de production. Pour ce type d'applications industrielles, les seules méthodes de surveillance opérationnelles sont celles sans modèle [41]. Les méthodes sans modèle ne disposent pas de modèle opératoire. Autrement dit, on ne dispose pas de modèle décrivant le comportement normal et les comportements défaillants du système. Les méthodes utilisées font alors appel à des procédures d'apprentissage et de reconnaissance de formes ou à l'intelligence artificielle.

1.5 Supervision dans un environnement SCADA

Le système SCADA fonctionne par l'acquisition des données provenant de l'installation, ces derniers sont affichés sur une interface graphique sous un langage humain, ces opérations sont exécutées en temps réel, ainsi les systèmes SCADA donnent ou opérateurs le maximum d'information pour une meilleure décision, ils permettent un très haut niveau de sécurité, pour le personnel et pour l'installation et permettent aussi la réduction des couts des opérations, les avantages qu'offre le SCADA sont obtenus avec la combinaison des outils softs et hard [9].

SCADA est un acronyme qui signifie le contrôle et la supervision par acquisition de données (en anglais : Supevisory Control And Data Acquisition). L'environnement SCADA collecte des données de diverses appareils d'une quelconque installation, puis transmit ces données à un ordinateur central, que ce soit proche ou éloigné, qui alors contrôle et supervise l'installation, ce dernier est subordonné par d'autres postes d'opérateurs [26].

1.5.1 Architecture du SCADA

SCADA entoure un transfert de données entre le Serveur (MTU,Master Terminal Units) et une ou plusieurs unités terminaux distantes (Remote Terminal Units RTU's) , et entre le Serveur et les terminaux des opérateurs.

La figure ci-dessous représente un schéma sur l'architecture d'un réseau SCADA qui utilise des routeurs pour joindre le poste de pilotage par billet de l'Internet [9].

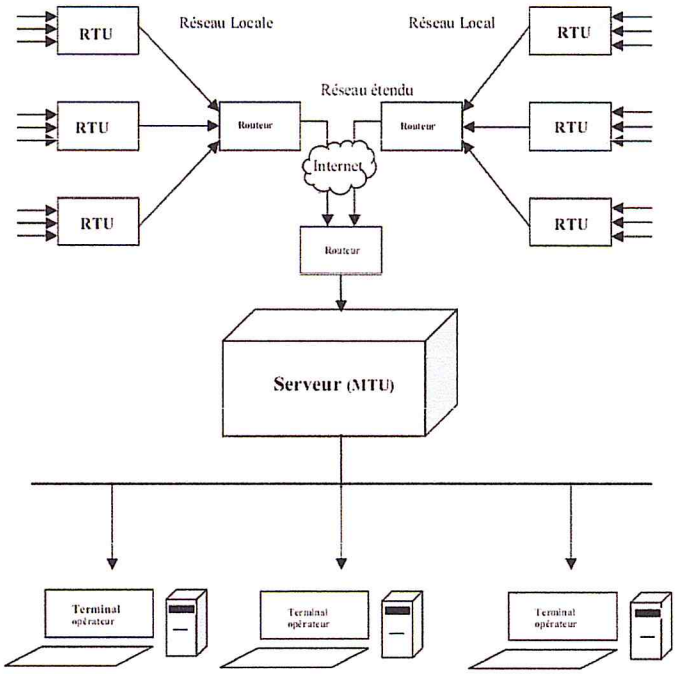


FIGURE 17 – *Architecture de la supervision dans environnement SCADA [9].*

Les **RTU** (Remote Terminal Units) : ou L'unité terminale distante Réalise un processus spécifique préconfiguré (analyse de la qualité de l'eau, calcul de flux ... etc).il a presque les mêmes fonctionnalités qu'un PLC (Programmable Logic Controller). La déférence primordiale est que RTU comme son nom l'indique permet est utlilisé pour la télémétrie distante (couvrant une longue distance) car utilisant des modes de communication sans fil pour l'acquisition et la transmission de donnée. Au plus, RTU est moins flexible, généralement plus compact et moins couteux en énergie [44].

MTU (Master Terminal Units) : ou unités de terminale maitre, dans les systèmes SCADA et un dispositif qui fournit les commandes à l'unité du terminale à distance (RTUs) qui sont située aux endroits à distance du contrôle [40].

1.5.2 Les avantages de SCADA

Il est fréquent que les systèmes SCADA disposent maintenant de capacité de contrôle en plus de leurs fonctions de supervision, ce qui les rend tout à fait de piloter totalement une chaine de production ou même un processus industriel complet. Ces systèmes permettent

le contrôle et l'assurance que toutes les performances désirées sont atteintes et la visualisation des performances désirées du système à chaque instant, et s'il y aurait une perte de performance, une alarme se déclenchera d'une manière automatique pour prévenir l'opérateur. Ils permettent de visualiser même la position où se situent la faute et l'élément défectueux, ce qui facilite la tâche du diagnostic et de l'intervention de l'opérateur.

Un système SCADA est capable de donner plusieurs informations sur le système de production ainsi aider l'opérateur à prendre la bonne décision, et ne pas se tromper dans son intervention. Au plus, il permet de Diminuer les tâches du personnel en les regroupant dans une salle de commande. En effet, avec la présence des interfaces graphiques, l'utilisateur peut suivre l'état de l'installation à chaque instant, ainsi il n'aura pas besoin de faire des visites de contrôle [26].

1.5.3 Fonctionnalités temps réels

La notion temps réel est devenue très importantes et indispensable dans la procédure de surveillance et de supervision en générale, elle permet de faire le rafraichissement des signaux à chaque instant, ce qui permet de suivre l'évolution de l'état du système d'une façon continue[9].

1.5.4 Les interfaces Homme-Machine dans SCADA (IHM)

L'interaction Homme-Machine représente l'ensemble des mécanismes d'échange d'information entre un humain et une machine pour accomplir une tâche ou atteindre un but particulier pour l'humain. Elle est caractérisée par le triplet (figure 25) : opérateur humain (ou utilisateur), machine et environnement de l'interaction (interface)

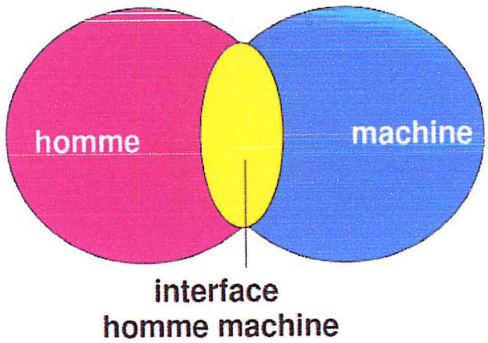


FIGURE 18 – *Interaction Homme Machine [46].*

Depuis quelques années, les interfaces Homme-Machine (IHM) s'imposent comme un facteur déterminant dans le succès des applications de supervision. Cet intérêt accordé aux interfaces Homme-Machine est à l'origine de différents travaux qui se sont intéressés à la définition, la caractérisation, la conception et l'implémentation des interfaces Homme-Machine. Ils sont un outil très important pour le bon déroulement du système industriel de même pour la réception d'informations de son environnement SCADA [46].

Définition d'une Interface Homme-Machine :

L'interface Homme-machine est l'ensemble des dispositifs matériel et logiciel permettant à un utilisateur d'interagir avec un système interactif. Il est présenté sous forme d'une interface graphique qui sert à afficher de manière claire et concise une vue d'ensemble du système d'information et l'état des services surveillés, de générer des rapports et de visualiser l'historique [47].

La figure 26 présente un exemple d'un IHM :

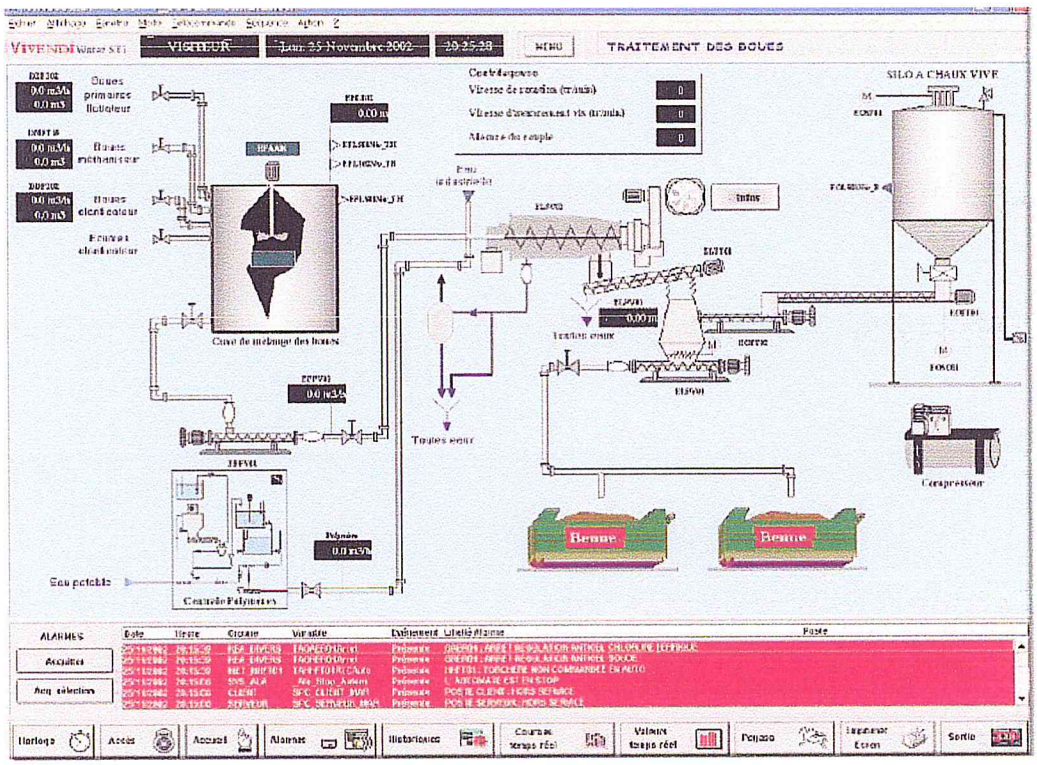


FIGURE 19 – *Exemple d'une IHM [47].*

Evolution des interfaces Homme machine

Les premières interfaces apparues, dès la fin des années 60, sont les interfaces en ligne de commandes basées sur une interaction textuelle entre l'utilisateur et la machine. Elles seront révolues lors de l'avènement en début des années 80 des interfaces WIMP (Windows, Icones, Menus and Pointing device) pour exprimer qu'elle utilise des métaphores de type : fenêtres, icônes, menus et dispositif de pointage. Les interfaces WIMP sont basées sur le principe WYSIWYG (What You See Is What You Get) pour ce que vous voyez est ce que vous obtenez. Ces interfaces offrent une interaction plus intuitive en réduisant les efforts d'apprentissage et ont grandement contribué à l'utilisation des machines par le grand public. Enfin, les interfaces post-WIMP dont le principe est de développer à l'extrême l'aspect intuitif de l'interaction afin de se rapprocher au plus de la communication entre humains, ont introduit des dispositifs d'interaction non conventionnels (manette de jeu, gant à retour d'effort, interface de supervision...) et utilisent de manière séquentielle et simultanée plusieurs modalités d'interaction conventionnelles (texte, manipulation directe) et non conventionnelles (parole, geste) [46].

1.5.5 Les logiciels de supervision SCADA

La présentation des IHM de supervision sous une forme graphique animée permet de faciliter la tâche à l'utilisateur pendant le processus de supervision. Plusieurs logiciels ade conception d'application dénommés génériquement SCADA ont été conçus pour faciliter la conception des systèmes de supervision.

Les logiciels SCADA (Supervisory Control And Data Acquisition) sont des logiciel permettant de faire l'acquisition de données en provenance d'un SAP dans le but est de le superviser ou réaliser une télésurveillance. Ces outils « temps-réel » permettent de visualiser les états physiques ou fonctionnels des équipements et de prendre en charge les fonctions avancées d'un procédé.

Il existe une multitude de logiciel SCADA, leurs nombre augmentent avec l'évolution des technologies et la demande du marché, nous allons citer quelques logiciels parmi eux :

InTouch : Depuis 25 ans, InTouch est le logiciel de supervision de référence grâce, notamment, à sa légendaire simplicité d'utilisation, sa fiabilité, son évolutivité, ses performances et ses nombreuses fonctionnalités. Avec une approche résolument différente des applications de supervision traditionnelle, InTouch offre des fonctionnalités graphiques avancées permettant aux développeurs d'être encore plus performants dans le développement et la maintenance des applications. Avec InTouch les opérateurs ont accès à toutes les données de l'atelier ou du bâtiment [25] ;

Pcvue (Package logiciel pour automatismes industriels) : PcVue est un logiciel modulaire exécutant plusieurs processus. Chaque processus est responsable d'une fonction ;

Panorama E2 : est une plateforme logicielle propriétaire commercialisé par la société CODRA pour bâtir rapidement des applications de supervision (SCADA) temps réel et temps différé, pour toutes les industries et dans tout le domaine. Au coeur de Panorama E2, un ensemble de services logiciels propose des fonctionnalités évoluées pour le traitement de l'information (acquisition de données, traitements des alarmes et évènements, calculs, archivage, animations ...) tout en gérant de manière automatisée ou manuelle les infrastructures (architectures réparties, redondantes, batteries d'équipements de communications, annuaires d'entreprise, diffusion et mise à jour des applications ...) [4] ;

WinCC : SIMATIC WinCC de siemens est un système de supervision de processus modulable qui offre des fonctions performantes de surveillance d'automatismes. WinCC offre des fonctionnalités SCADA complètes sous Windows pour tous les secteurs – depuis la configuration monoposte jusqu'aux configurations multipostes distribuées avec serveurs redondants et solutions multisites avec clients Web. WinCC se distingue en particulier par son ouverture totale. Ses fonctions sont facilement combinables avec des programmes standards et utilisateur pour donner naissance à des solutions HMI répondant exactement aux besoins de la pratique. Grâce aux interfaces rendues au domaine public, les intégrateurs peuvent développer leurs propres applications en basant les extensions système sur WinCC ;

Eclipse SCADA : Est un logiciel open SCADA, c'est une manière de relier les différents dispositifs industriels à un système de communication commun aussi bien que visualiser les données aux superviseurs.

La figure ci-dessous donne une comparaison entre les différents logiciels :

Logiciel	Langage	Remarques
Eclipse SCADA (openSCADA)	Java	- Un bon toolset - Manque des normes basées sur les IHM - Ne fonctionne pas sur les navigateurs web - Open Source
PV Browser	C++	- Une bonne boite d'outils - Mais basé sur un protocole coté client seulement - Conçu seulement pour le navigateur Firefox - Possède des widget pour l'animation SVG - Non Open Source
Mango	Java	- Non Open Source - Payant - Utilise AJAX - Et non temps réel
ScadaBr	Java	- Open source - Non temps réel

FIGURE 20 – *Comparaison entre les logiciels SCADA .*

1.6 Conclusion

Au cours de ce chapitre nous avons présenté brièvement des généralités sur l'automatisme et les systèmes de production automatisés en décrivant les équipements nécessaires dans la production automatisée.

Nous avons donné à quoi consiste la supervision et les outils nécessaires pour sa mise en œuvre, et l'importance de la supervision au sein des installations industriels. Nous nous sommes basée sur les systèmes SCADA puisque c'est l'objectif de notre travail qui est un outil qui permet de réaliser une supervision à distance qui est très utile pour les industries à haut risque.

Chapitre 2

SCADA et technologies WEB

2.1 Introduction

De nos jours SCADA est le noyau de beaucoup de systèmes industriels modernes, avec la nécessité d'être sur la même longueur d'onde de la complexité des composants de ces systèmes et vus les opportunités et les avantages compétitifs qu'ils offrent aux entreprise dans le domaine industriel.

Dans le chapitre précèdent, nous avons vu que l'environnement SCADA permet une supervision en temps réels, un bon control de sur le système offrant des interfaces graphiques (IHM) sur lesquelles est présentée le système avec la possibilité d'intervention en modifiant les valeurs d'un paramètre, et surtout il nous offre un suivi de tous les évènements ayant lieu au sein du système.

Dans ce chapitre nous présenterons l'évolution des logiciels SCADA puis nous abordons les logiciels SCADA existant sur le web, et on donnera un exemple illustratif. Ensuite nous procèderons à un nouveau logiciel pour le développement coté serveur et sa relation avec le HTML 5.

2.2 Scada et les technologies web

En premier temps le développement des logiciels SCADA a été propre à chaque entreprise (par exemple siemens) ou dédié aux équipements de cette entreprise, en revanche ces logiciels ne sont pas accessibles par les navigateurs web récent (Firefox, google chrome), des fois ils ne disposaient même pas d'une interface (IHM). Néanmoins, relier les systèmes SCADA à Internet peut également fournir beaucoup d'avantages en termes de contrôle. L'apparition du web et la standardisation des protocoles de communication a permis aux responsables, à partir de leurs postes et via une interface sur un navigateur, d'accéder aux informations collectées.

Le développement web offre la possibilité de mettre en oeuvre l'architecture Client/serveur qui est nécessaire pour la configuration monoposte jusqu'à la configuration multiposte distribuée avec serveur redondants et avec des clients web.

Afin de pouvoir faire face aux exigences croissantes, le logiciel doit pouvoir être extensible à tout moment, sans pour autant impliquer des failles technologiques, de même employer des normes adoptées et portabilité en temps réel à travers le web.

Et pour arriver à concevoir un logiciel plus sophistiqué de SCADA il doit être modulaire « plugin », pas trop dédié à un type spécifique de composants et finalement libre « OpenSource ».

2.3 Un exemple scada/ihm basée sur le web (mango)

2.3.1 Description du logiciel Mango M2M

Mango est un élément important d'un système complet de M2M (machine to machine). Il fournit une base de données pour stocker les données rassemblées, une interface homme-machine pour fournir des graphiques, des données diagnostiques, et g l'information de gestion. Mango peut présenter l'information aux personnels exploitants dans une interface graphique facile à utiliser, sous forme de diagramme imitateur. Par exemple, une photo d'un thermomètre peut montrer à l'opérateur la température de la salle où une des sondes est localisée [31].

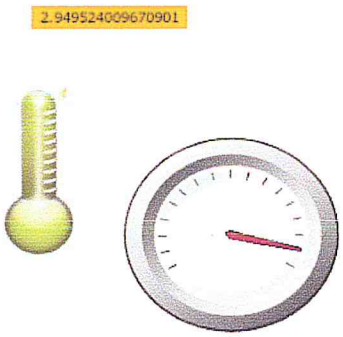


FIGURE 21 – *Représentation graphique des sondes [31].*

En outre, il permet à l'administrateur d'installer des alarmes sur des événements spécifiques, de noter ces événements, et pour réagir à ces événements d'une mode automatisée. Le système est conçu pour des multiples utilisateurs : il a un système de gestion sophistiqué d'utilisateur.

D'un point de vue développeur

Les concepts principaux du système doit être présenté, parce qu'ils seront visé à plusieurs occasions.

Les concepts principaux du système doit être présenté, parce qu'ils seront visé à plusieurs occasions. Le rassemblement des données commence par installer un point d'émission de données « DataSource ». Différents points d'émission de données sont disponibles, basés sur leur protocole sous-jacent pour recueillir des données[31]. Exemples pour les protocoles sont Modbus IP, Modbus serial.

Mango peut également recevoir des données par le HTTP : il peut télécharger une image, et il peut analyser le HTML, ainsi avec l'aide d'une expression régulière bien formée, il peut facilement extraire des données de valeur (par exemple données de la température) de toute page Web (site Web d'une station météorologique)[31].

FIGURE 22 – *Un exemple de DataSource avec DataPoint [31].*

Après l'installation d'un point d'émission de données, on doit définir de soi-disant points de repères « DataPoin ». Un point d'émission de donnée peut avoir beaucoup de différents points de repères. Un dispositif physique de mesure (un point d'émission de données) peut, par exemple, fournir des données au sujet de la température, de l'humidité et de l'éclat

d'une salle. Les derniers trois objets sont les points de repères. Ce n'est pas les points de repère. Il est crucial de ne pas confondre des points de repères avec leurs valeurs réelles : un point de repères est simplement un concept (i.e. humidité), alors que sa valeur sera 10 %. Les points de repères ont ainsi des valeurs de point de repères. Les points de repères ne changent pas, seulement leurs valeurs le font [31].

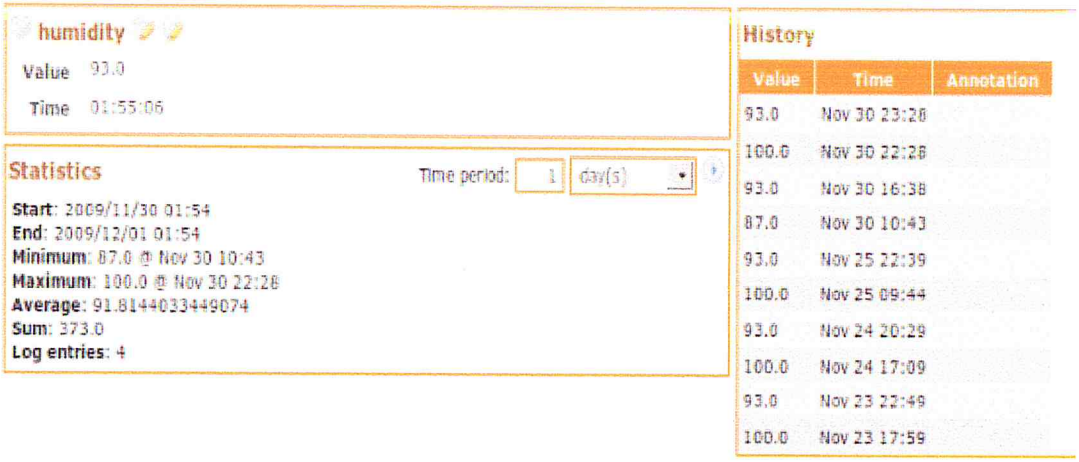


FIGURE 23 – *Les détails et les valeurs d'un Data Point [31] .*

On peut examiner et suivre ces points de repères (et leurs valeurs) à l'aide de la (watch lists) ou (graphical views). Les listes de montre sont un ensemble tabulaire de points qu'on souhaite regarder. Les valeurs de point et les temps de valeur se mettent à jour automatiquement [31].

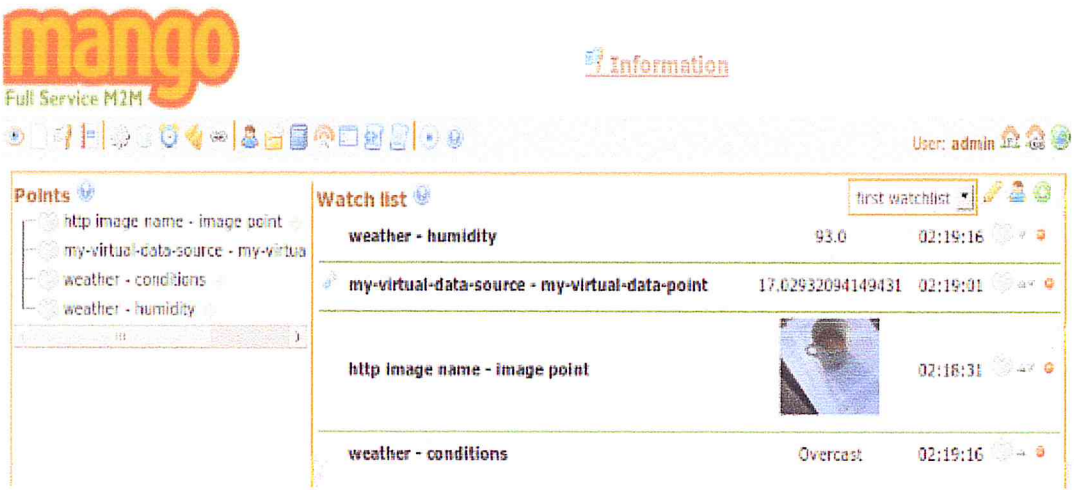


FIGURE 24 – *Watch lists[31].*

2.3.2 Les composants logiciels utilisés

- Spring's Web MVC (Model View Controller) framework, version 2.5.3, http ://www.springsource.(

Spirng est un framework open source, Il a été créé pour aborder la difficulté du développement d'applications d'entreprise. Avec une description plus technique on a pu la décrire comme injection. Spring se compose de plusieurs modules bien définis

- Direct Web Remoting, version 2.0.1, http ://directwebremoting.org/dwr/ DWR,

qui est l'abréviation le Web Remoting Direct, est un framework basé sur Java d'Ajax cela vous laisse à accéder d'une façon virtuelle à n'importe quel objet de Java de côté serveur par Javascript.

- Dojo JavaScript toolkit, version 0.4.2, http ://www.dojotoolkit.org, Dojo est un

framework ouvert de Javascript qui simplifie la programmation d'Ajax.C'est une collection des composants de Javascript pour construire des applications javascript .

Mango n'est plus gratuit et il n'est plus open source.

En premier temps nous avons choisi mango pour développer notre système de supervision mais nous avons découvert que le logiciel que nous travaillons avec c'était une version de démonstration seulement c'est pour ça nous avons opté pour un autre logiciel web.

2.4 Node.js

Il y'a quelques années, les développeurs web ont écrit des pages HTML avec du javascript pour créer des pages web avec peu d'interactivité, puis ils ont voulu créer des pages web complexes, ils ont adopté des langages comme PHP, Ruby, Java pour écrire du code serveur.

Si Node.js a su percer dans le domaine des applications web, c'est en partie parce que celuici a introduit un changement de paradigme par rapport aux stacks classiques, révolution à l'origine de gains de performances notables.

Node.js est une plateforme de développement Javascript. Ce n'est pas un serveur, ce n'est pas un framework, c'est juste le langage Javascript avec des bibliothèques permettant de réaliser des actions comme écrire sur la sortie standard, ouvrir/fermer des connections réseau ou encore créer un fichier.

La philosophie de node.js est de créer une boucle d'événements au lieu de se dupliquer (créer des threads), pour gérer plusieurs tâches en parallèle. De cette manière, paraît-il, un serveur web node.js peut gérer des millions de connexions concourantes. De cette manière surtout, nul besoin de gérer des accès simultanés à des structures de données : il n'y a pas d'accès simultané.

Il est souvent confondu avec un serveur car c'est son origine.

Node.js a été créé par Ryan Dahl dans le but de pouvoir créer des applications temps réel où le serveur est capable de pousser de l'information au client [1] .

2.4.1 Javascript coté seveur

Node.js nous permet d'utiliser le langage JavaScript sur le serveur. Il nous permet donc de faire du JavaScript en dehors du navigateur [32].

Node.js bénéficie de la puissance de JavaScript pour proposer une toute nouvelle façon de développer des sites web dynamiques [32].

JavaScript avait toujours été utilisé du côté du client, c'est-à-dire du côté du visiteur qui navigue sur notre site. Le navigateur web du visiteur (Firefox, Chrome, IE...) exécute le code JavaScript et effectue des actions sur la page web [32].

L'arrivée de node.js a offert un environnement côté serveur qui nous permet aussi d'utiliser le langage JavaScript pour générer des pages web. En gros, il vient en remplacement de langages serveur comme PHP, Java EE... etc.

Nous allons présenter dans la figure suivante l'architecture client/serveur de node.js .

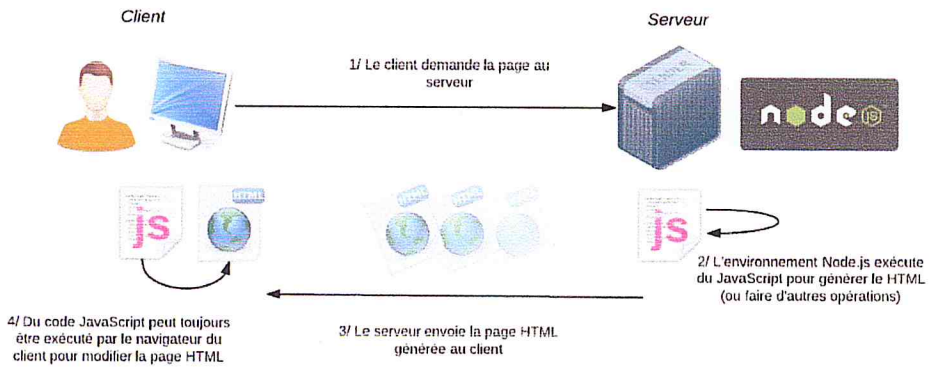


FIGURE 25 – *Le schéma client/serveur de Node js [1].*

NodeJs permet de faire des requêtes asynchrones, ce qui permet une gestion des entrées/sorties de manière non bloquante, très pratique pour les applications qui ont besoin de temps réel Avec Node.js, vous pouvez créer des applications rapides [1] comme :

- Un serveur de Chat
- Un système d'upload très rapide
- et de façon générale n'importe quelle application qui doit répondre à de nombreuses requêtes rapidement et efficacement, en temps réel.

2.4.2 Principe de fonctionnement

Node.js est monothread, contrairement à Apache. Cela veut dire qu'il n'y a qu'un seul processus, qu'une seule version du programme qui peut tourner à la fois en mémoire.

En effet, il ne peut faire qu'une chose à la fois et ne tourne donc que sur un noyau de processeur. Mais il fait ça de façon ultra efficace, et malgré ça il est quand même beaucoup plus rapide .Cela est dû à la nature "orientée évènements" de Node.js. Les applications utilisant Node ne restent jamais les bras croisés sans rien faire. Dès qu'il y a une action un peu longue, le programme redonne la main à Node.js qui va effectuer d'autres actions en attendant qu'un évènement survienne pour dire que l'opération est terminée [1]. La figure ci-dessous va nous illustré le fonctionnement monothread de node.js

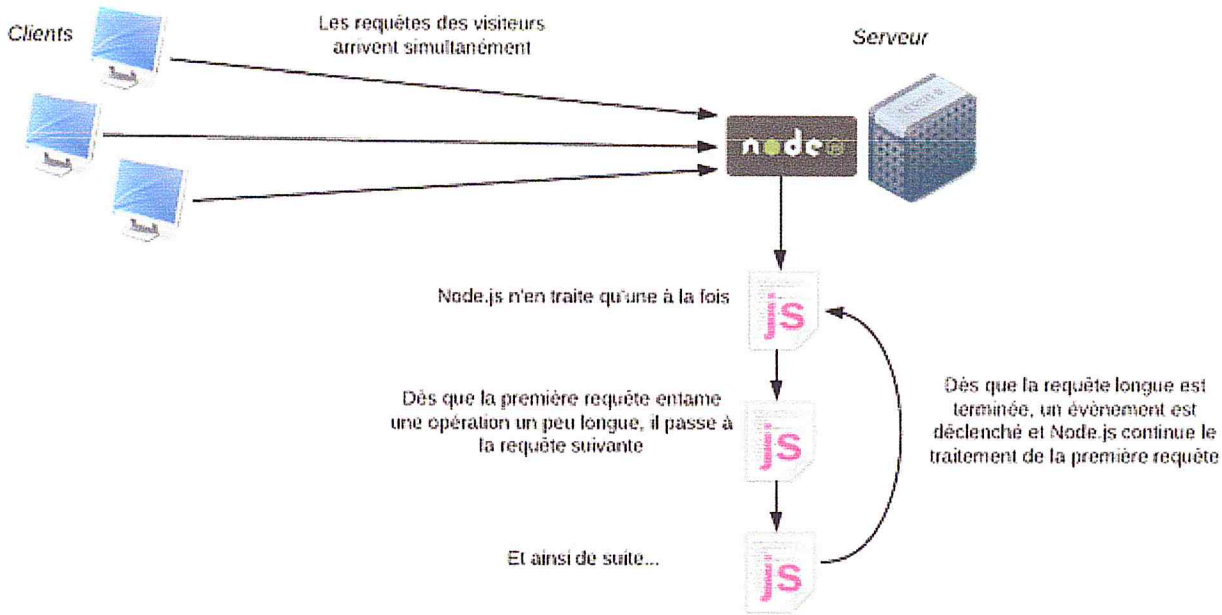


FIGURE 26 – *Architecture monothread de node.js [1].*

2.4.3 Les évènements

Node.js est un environnement de développement JavaScript basé sur les évènements. Node.js a pu mettre en place une architecture de code non bloquante [32].

Les évènements sont à la base de Node.js. C'est ce qui fait que Node.js est puissant mais aussi un peu plus difficile à appréhender, puisque ça nous impose de coder avec beaucoup de fonctions de callback.

La figure ci-dessous va nous expliquer la notion de la programmation non bloquante :

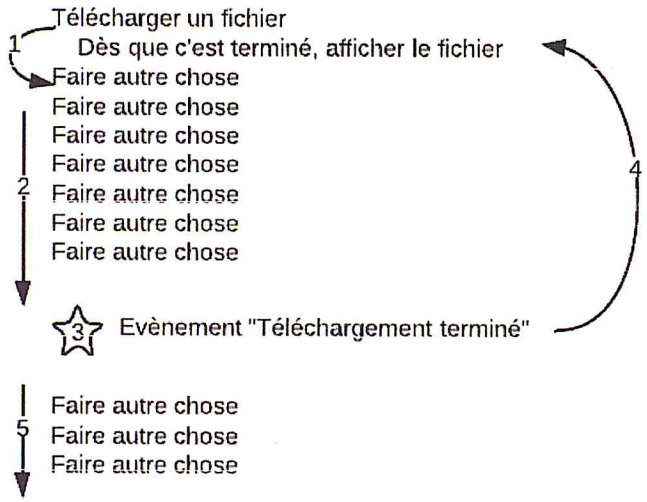


FIGURE 27 – *Le modèle non bloquant en programmation [1].*

2.4.4 Npm(Node Package Manager)

Npm est le gestionnaire de paquets pour node.js est un peu l'équivalent d'apt -get sous Linux pour installer des programmes. Une simple commande et le module est téléchargé et installé [1].

En plus, NPM gère les dépendances. Cela signifie, que si, un module a besoin d'un autre module pour fonctionner, NPM ira le téléchargé automatiquement NPM est très actif, il y'a plusieurs dizaines de milliers de modules disponibles.

NPM possède un site web : http ://npmjs.org

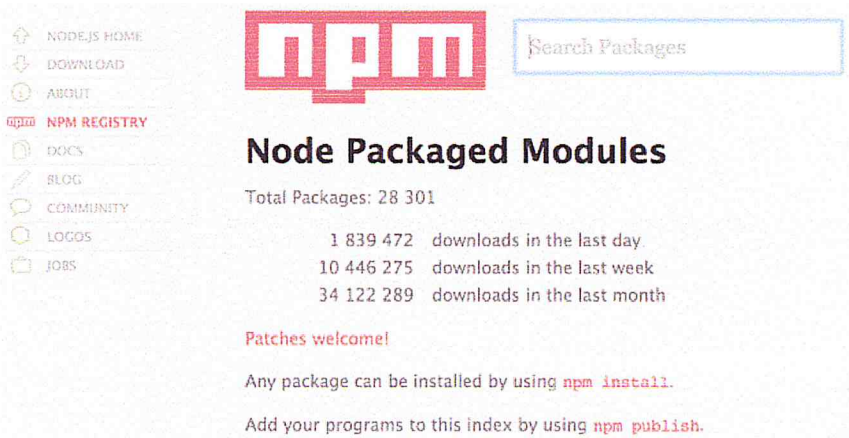


FIGURE 28 – *Le site web de NPM [1].*

Le package JSON Est un format d'échange de données, facile à assimiler pour les humains et aussi à interprété pour les ordinateurs, dérivé du langage de script de JavaScript pour représenter des structures de données simples et d'objets.

2.4.5 Les modules Node.js

Le noyau de Node.js est tout petit. Pourtant il est très riche grâce à son extensibilité .ces extensions de Node.js sont appelés **modules** .

Il existe des milliers de modules qui offrent des fonctionnalités variées : de la gestion des fichiers uploadés à la connexion aux bases de données MySQL ou à Redis, en passant par des frameworks, des systèmes de templates et la gestion de la communication temps réel avec le visiteur, de nouveaux modules apparaissent chaque jour. Les modules sont gérer par NPM, et ils évoluent de version en version.

- **Créer des modules**

  Il faut mettre votre fichier test.js dans un sous-dossier appelé **nodemodules**. C'est une convention de Node.js

  En résumé

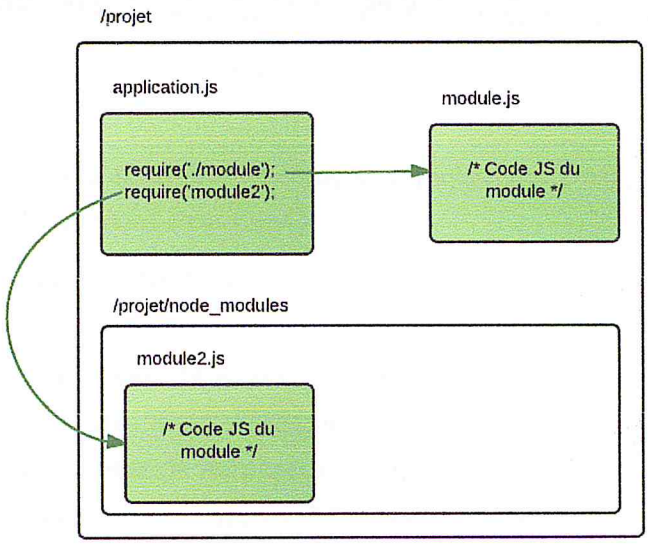


FIGURE 29 – *Node.js sait où chercher les modules [1].*

Et puis faire :

```
C:\Users\GHANOU\nodejs>npm publish
```

FIGURE 33 – *La publication du module.*

2.4.6 Quelques modules

Socket.IO

Socket.IO est une bibliothèque et se base sur plusieurs techniques différentes qui permettent la communication en temps réel.

Express.js

Express.js s'agit d'un microframework pour Node.js. Il fournit des outils de base pour aller plus vite dans la création d'applications Node.js [1].

2.5 Html5 et node.js

Avec node.js javascript est utilisé du côté du serveur mais aussi du côté du client pour manipuler les pages html.

Le récent standard Html5, couplé à javascript permet de réaliser de véritables applications dotées de fonctionnalités similaires à celles développées à l'aide d'un langage traditionnel [43]

Aujourd'hui,Le HTML5 est le langage le plus utilisé non seulement pour la création des sites web mais aussi pour la création des jeux performants et des stores sur des plateformes différentes. Pour cela le HTML5 peut être employé pour la réalisation d'une HMI dans un système de contrôle et d'acquisition de données grâce à ces deux nouveautés SVG et Canvas [43]

2.5.1 SVG (scalable Vector Graphics)

Permet comme son nom l'indique d'afficher des objets graphiques vectoriels. Les objets sont décrits dans un langage XML qui vont enrichir le DOM comme les autres éléments HTML. Pour générer le SVG, il existe des outils bien pratiques :

- InkScape : un logiciel,client lourd ,libre et gratuit ;
- SVG Editor : un logiciel en ligne qui marche donc directement dans votre navigateur.

2.5.2 Canvas

Il faut voir **canvas** comme une image PNG dynamique. Vous disposez d'une surface de dessin (une bitmap) dans laquelle vous allez dessiner à l'aide de primitive accessible via JavaScript. Ces primitives sont grosso-modo les mêmes qu'avec SVG : rectangles, lignes, remplissage de formes, courbes de Bézier, etc.

2.6 Conclusion

Dans ce chapitre nous avons parlé de l'évolution des logiciels SCADA existant dans le monde industriels, et comment ces logiciels ont envahi le web.

Nous sommes habitués à voir le JavaScript fonctionner coté client (plus précisément sur le navigateur) mais avec l'apparition de Node.js le JavaScript fonctionne aussi sur le côté serveur.

Nous avons présenté un aperçu sur les bases de Node.js (les modules, NPM, et le package JSON) et la puissance du HTML5 pour la création des sites web et la possibilité de réaliser des IHM avec les nouvelles améliorations de HTML5.

Chapitre 3

Conception du système

3.1 Introduction

Pour confronter la concurrence industrielle, les ingénieurs sont invités à se concentrer de plus en plus sur les côtés de recherches du marché et du développement des affaires de leurs activités. Ils doivent donc relever le défi d'être experts en matière d'affaires et programmeurs de logiciel avancé. En parallèle, les produits deviennent toujours dépendants du logiciel et exigent ainsi des qualifications de haut niveau dans la programmation.

En réponse à de tels défis les développeurs des applications avant d'arriver au produit final, adoptent une méthode pour concevoir leurs applications, pour établir les modèles d'application qui peuvent être émergents, et qui répondent aux exigences des clients.

Dans ce chapitre nous présentons la conception de notre système. En premier lieu nous donnons un aperçu du langage UML, puis nous procédons à la méthodologie ACCORD-UML. En deuxième lieu nous donnons l'architecture général de notre système suivie d'une analyse préliminaire qui consiste à donner le dictionnaire et décrire les cas d'utilisation, ensuite une analyse détaillée qui contient les digrammes de séquences avec la description de chacun et le diagramme d'état/transition.

3.2 Langage UML

Dans le secteur du développement logiciel, suite à l'adoption massive de l'approche objet pour la réalisation des applications, et face au besoin pour les développeurs d'avoir à leur disposition de nouvelles méthodologies d'aide à la conception objet, une multitude de méthodes a été lancée dont une cinquantaine entre 1990 et 1995, chacune essayant de s'imposer sur le vaste marché du développement logiciel. En 1995 cependant, répondant à un appel à projet lancé par l'OMG (Object Management Group) pour l'obtention d'une

méthodologie standard de modélisation, trois spécialistes recrutés par la société Rational Software et ayant chacun proposé une méthode particulière, décident de spécifier un nouveau langage, comprenant l'impossibilité de s'orienter vers une méthode unique, mais désireux d'obtenir un langage standardisé, ils créent UML (Unified Modeling Language) [11].

Les trois approches qui ont influencé sa spécification sont l'OMT, la méthode Booch et la méthode OOSE. Le cycle de vie du langage UML est représenté dans la figure 40 :

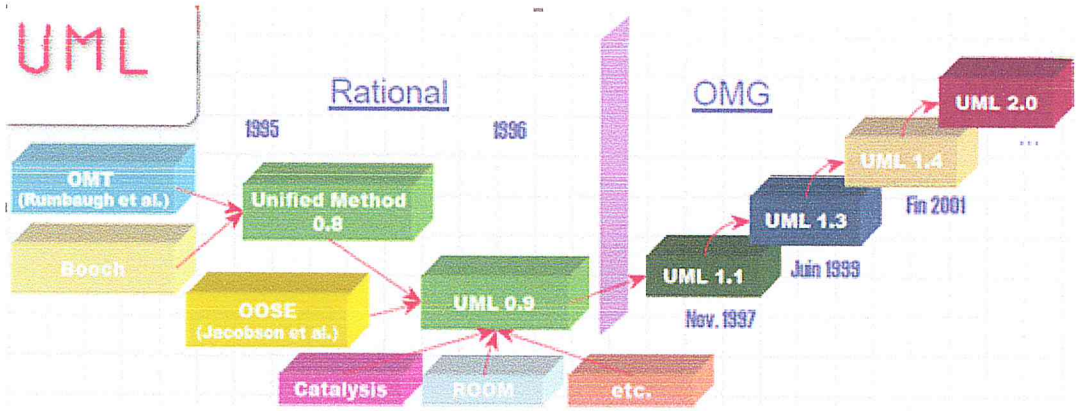


FIGURE 34 – *Ligne de vie du langage UML [33].*

Le langage UML est la pierre angulaire de l'approche MDA (Model Drivent Architecture) de l'OMG, il permet la description graphique de tous les aspects d'un système. UML est un langage standard conçu pour l'écriture des plans d'élaboration de logiciel, il peut être utilisé pour visualiser, spécifier, construire et documenter les artéfacts d'un système à fort composantes logiciel [11], mais UML n'est en définitive qu'un langage sans guides d'utilisation pour l'ordre et le choix des diagrammes dans le cycle de développement. Beaucoup de développeurs utilisent UML de façon intuitive ou se limitent à certains diagrammes proches de la logique liée à la technologie cible qu'ils veulent modéliser. [33].

Plusieurs projets de recherche sont menés à proposer des méthodes pour la mise en œuvre d'un processus de conception ou d'analyse de système basées sur le langage UML, Ces méthodologies sont fortement liées au domaine de l'informatique car elles sont issues pour certaines directement de travaux dans le contexte d'un langage de programmation donné. Néanmoins, les concepts abordés peuvent aisément s'appliquer à la définition de systèmes beaucoup plus larges, ne se limitant pas à une application informatique [11]. On peut citer des méthodologies qui ont essayé de s'imposer dans les processus de conception, notamment le RUP (Rational Unified Process) inspiré du Processus Unifié décrit par les créateurs d'UML et mis en place et commercialisé par Rational, la méthode COMET/UML orientée pour les systèmes concurrents et distribués.

De même la communauté de chercheurs travaillant sur les MAS (Multi-Agent Systems) a proposé des évolutions possibles du langage, réunies dans l'extension de langage AUML (Agent Unified Modeling Language) pour pouvoir prendre en compte les spécificités des architectures à base d'agents. On retrouve donc tout naturellement des propositions de méthode et d'outils basés sur UML et adressant le domaine particulier du développement d'applications temps-réel : la proposition de RT-UML (Real Time UML) introduisant le concept de capsule et de port qui a donné ensuite lieu à une succession de travaux de thèse et d'articles sur des extensions UML pour les systèmes temps réel et les contraintes très particulières de ces domaines, Le projet DESS(acronyme) propose une intégration des modèles et d'outils utilisés pour le temps réel et leur adaptation à un processus de développement UML afin d'enrichir la sémantique des modèles d'UML Le projet Protes quant à lui vise un objectif de standardisation auprès de l'OMG d'un profil UML 2.0 pour les systèmes temps réel embarqués. Cela implique des travaux au niveau du méta-modèle d'UML afin d'y intégrer les éléments nécessaires à la prise en compte de certaines caractéristiques spécifiques au temps réel. Le projet AIT WOODES qui Mise au point La méthodologie ACCORD-UML qu'on va suivre son processus de développement dans le cadre de notre projet .cette méthode est plus orientée vers l'aspect méthodologique avec développement d'outil industriel [11].

UML compte maintenant parmi les langages de modélisation les plus répandus, enseignés et outillés pour le génie logiciel. Bien que langage de modélisation généraliste, UML a la capacité d'être adapté aux besoins d'un domaine particulier d'application au travers de la définition de stéréotypes, valeurs étiquetées et contraintes réunis dans un profil UML. C'est ainsi que UML s'est également répandu dans des domaines où initialement il n'aurait pas pu trouver sa place.

3.3 La méthodologie ACCORD-UML

3.3.1 Définition et principe

Accord UML est une méthodologie complètement basée sur UML et visant le développement des systèmes par des non-experts du domaine temps réel [48]. L'objectif général de la méthodologie est de masquer autant que possible les aspects d'implantation autour d'une approche dirigée par les modèles (patrons de conception, raffinement automatique de modèle, génération de code, validation par construction de modèles...), afin de permettre aux développeurs de se concentrer sur les aspects métier du système (fonctionnalités, contraintes de performance...)[20]. Elle repose ainsi sur un ensemble minimal d'artéfacts suffisamment abstraits pour être employés par des non-spécialistes de temps réel tout en permettant une expression des aspects qualitatifs (concurrence, comportement, commu-

nication, ...) et quantitatifs (contraintes temps réel) des applications à modélisées [48].

En plus des artéfacts de base, Accord|UML définit un ensemble de règles de modélisation guidant l'utilisateur durant tout le cycle de développement de l'application, ainsi qu'un ensemble de règles de transformation permettant le passage d'une phase de modélisation à une autre en assurant la continuité du développement [48].

La validation dans la méthode ACCORD concerne trois aspects : la vérification, la simulation et le test. L'aspect vérification utilise les méthodes formelles par le biais de l'outil AGATA en générant un modèle exécutable pour vérifier que le système décrit par le modèle de conception satisfait bien les exigences définies dans les diagrammes d'exigences. Ces dernières sont dérivées pour être spécifiées formellement soit par des formules de logique soit par des observateurs. Les exigences vérifiées incluent les propriétés de sureté, vivacité, blocages temporels et comportements nonconformes au cahier des charges reposant sur la description d'un temps logique [48].

3.3.2 Processus de développement

La figure suivante décrit les principales étapes d'une modélisation selon la méthode Accord ainsi que les relations et actions à mettre en œuvre pour passer d'un modèle à l'autre :

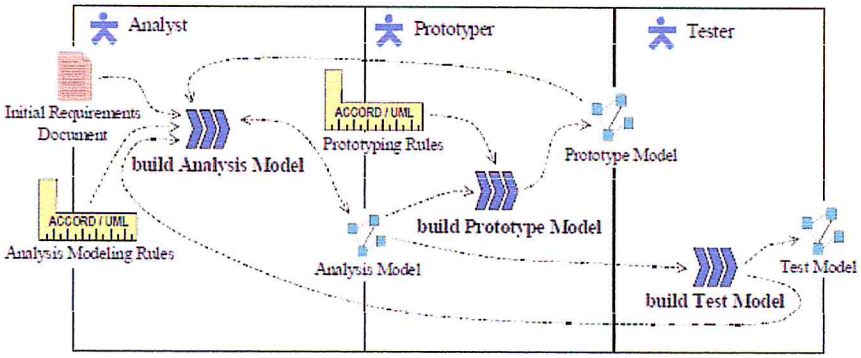


FIGURE 35 – *Processus de développement ACCORD [48].*

La phase d'analyse : elle décrit ce qu'on attend de system. Elle est coupée en deux sous-modèles :

- Modélisation d'analyse préliminaire PAM (Preliminary Analysis Model)) avec une première analyse des attentes vis-à-vis du système comprenant une identification des cas d'utilisation et une première description comportementale à l'aide de scénario [11].

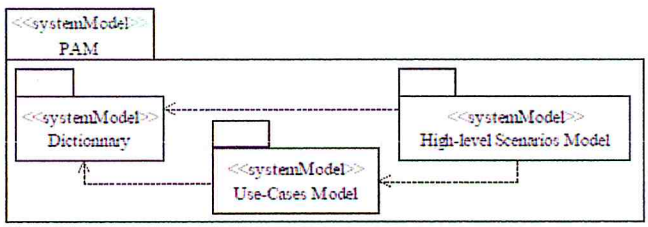


FIGURE 36 – *Analyse préliminaire PAM [22].*

- Modélisation d'analyse détaillée DAM (Detailed Analysis model) avec une première caractérisation structurelle du système et l'identification de classes et de composants, une caractérisation des interactions entre ces objets puis de leur comportement interne [11].

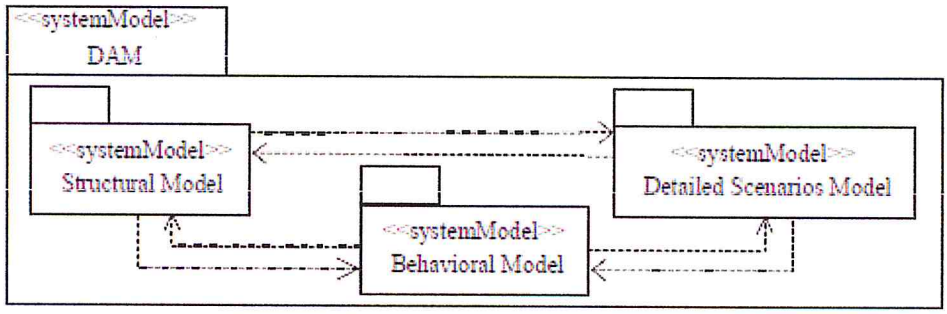


FIGURE 37 – *Analyse détaillé DAM [22] .*

La phase de prototypage (PrM) : le but de la phase de prototypage est d'obtenir une maquette courante complète de l'application à partir de sa DAM en précisant le comportement des instances du système et les contraintes de temps réels qui lui sont liées [33].

La figure ci-dessous contient les différents diagrammes UML utilisés dans chaque étape :

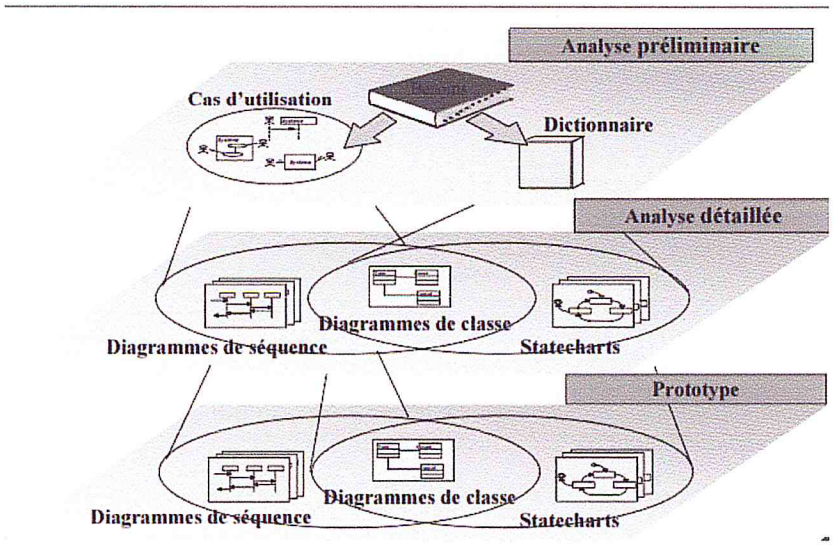


FIGURE 38 – *les diagrammes utilisés dans chaque phase [22].*

3.4 L'architecture générale

l'architecture de notre application est une architecture client/serveur . d'une part le serveur node js se connecte avec l'automate (PLC) via le protocole de communication modbus TCP/IP. L'automate à son tour gère les équipement de la cellule flexible (RFID, Capteurs, Convoyeur, Robot) . D'autre part, le serveur node js communique avec un ou plusieurs clients (HMI) par l'intermidiare de Socket.io le schéma suivant décrit cette architecture :

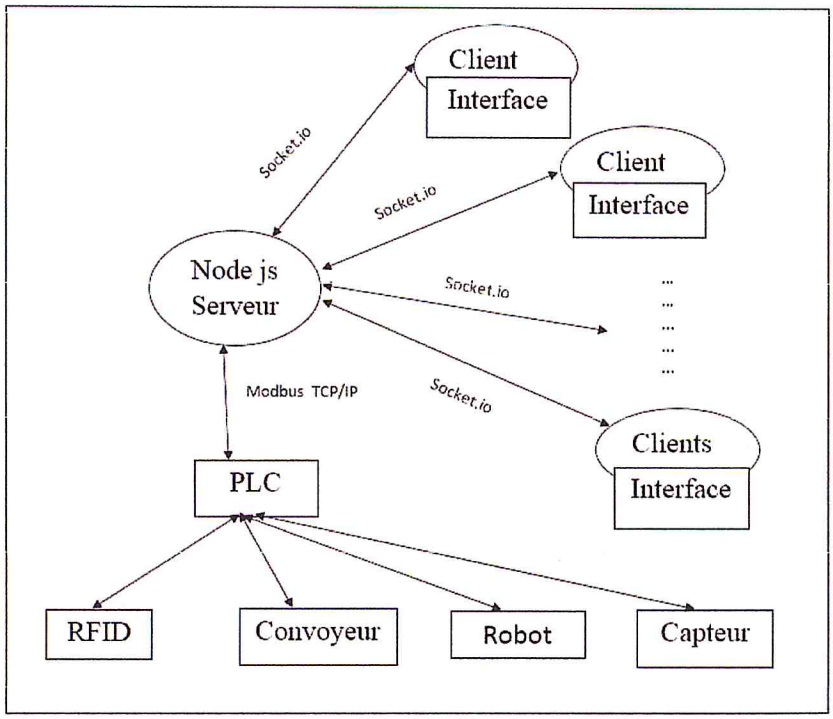


FIGURE 39 – *Architecture client/serveur.*

3.5 Modélisation du système

3.5.1 Analyse préliminaire

Construire le dictionnaire :

Noms : Acteurs	Qualificatifs : attributs ou relations	Verbes : Opération
Utilisateur	Relation avec le robot	-Vérifier le bon fonctionnement du robot -Modifier la tâche du robot
	Avec l'automate	Contrôler le système : -Détection des pannes el des erreurs et les corrige -Acquisition des données (affichage des données dans l'IHM) -Gestion du démarrage et d'arrêt de la cellule

FIGURE 40 – *Le dictionnaire des cas d'utilisattion*

Décrire les cas d'utilisation

L'utilisateur du système peut suivi et piloter la cellule robotisée. Le suivi consiste vérifier le bon fonctionnement de chaque équipement de la cellule et le bon déroulement du procédés par l'intermédiaire de l'HMI ,il permit aussi de visualiser l'échange d'états du robot. Lorsque le fonctionnement devient anormal, le rôle d'utilisateur est de détecter les erreurs et les pannes et les corrige. Il gère aussi le démarage et l'arrét de la cellule et change la tache de robot pour réponder au besoins.

La figure suivante représente le diagramme de cas d'utilisation de notre système.

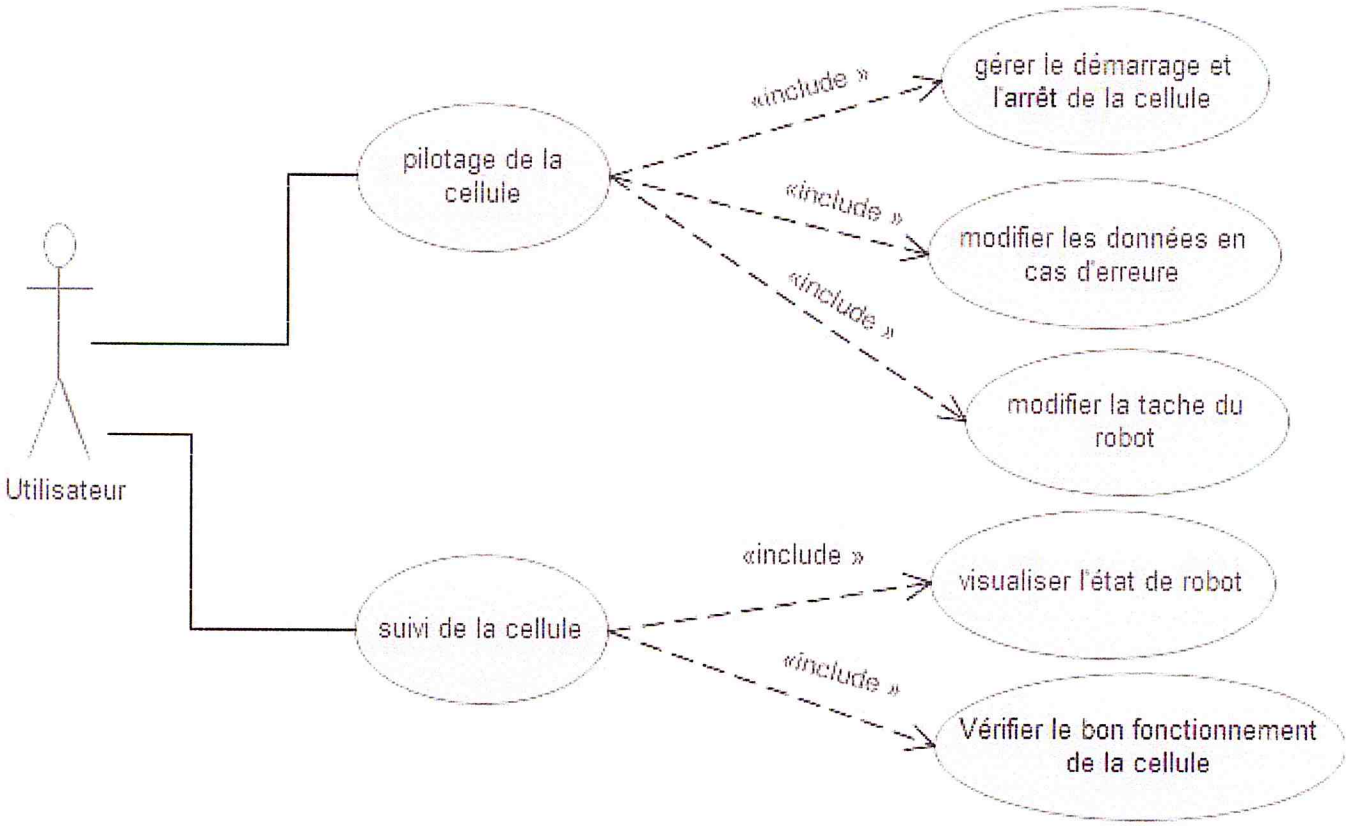


FIGURE 41 – *Diagramme de cas d'utilisation.*

3.5.2 Analyse détaillée

Les diagrammes de séquence

1. **Gérer le démarrage et l'arrêt de la cellule :**

   Pour que la cellule soit en marche l'utilisateur doit appuyer sur le bouton allumer dans l'HMI, l'HMI est connectée avec l'automate via le modbus TCP/IP, l'automate à son tour met la cellule en marche. Un message est livré depuis l'HMI pour l'utilisateur disant que la cellule est en marche.

   Pour que la cellule s'arrête l'utilisateur doit appuyer sur le bouton arrêt dans l'HMI, L'HMI est connectée avec l'automate via le modbus TCP/IP, l'automate à son tour met la cellule en arrêt, un message est livré à l'utilisateur disant que la cellule est en arrêt.

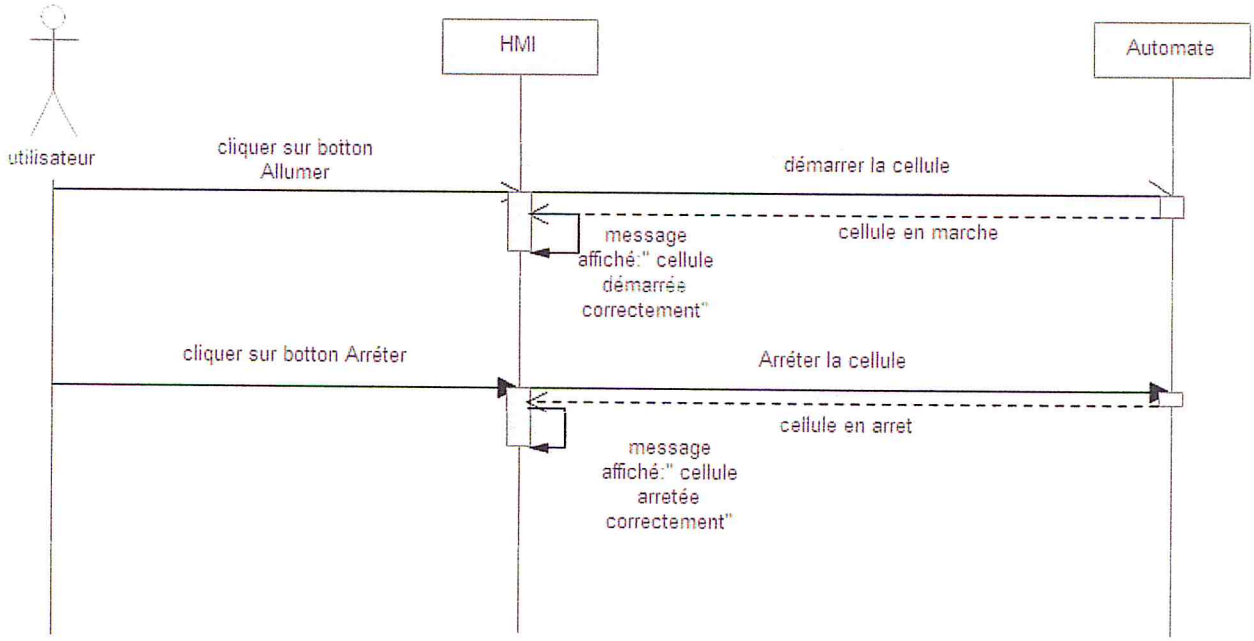


FIGURE 42 – *Diagramme de séquence pour l'arrêt et démarrage de la cellule.*

2. Modifier les données en cas d'erreur

Lors d'un fonctionnement anormal un signal d'erreur se génère en temps réel, l'automate l'envoie à l'HMI, un message d'erreur est attribué à l'utilisateur. L'utilisateur à son rôle essaye de détecté l'erreur qui s'est produite, si c'est une erreur qui a une relation avec les paramétrages l'utilisateur modifies les paramètres par les 2 méthodes (writeMultipleCoils ou writeMultipleRegisters) dans l'automate pour que le fonctionnement soit normal sinon si l'erreur détectée est une erreur matérielle un arrêt d'urgence est imposé.

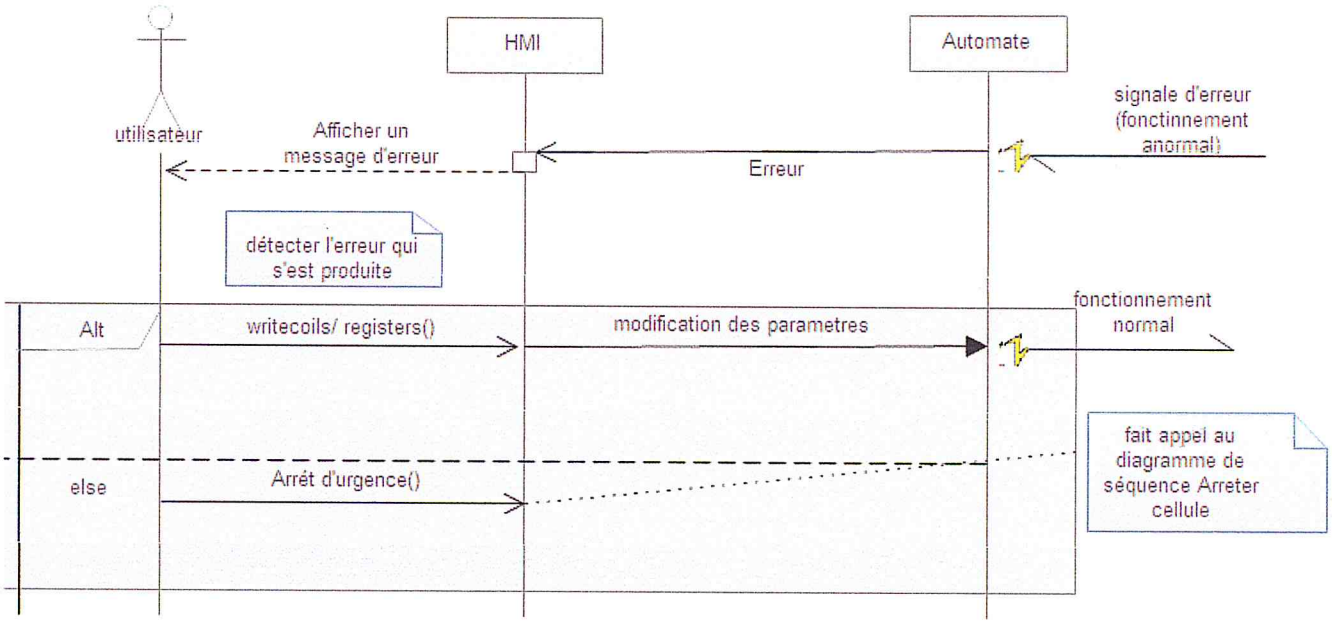


FIGURE 43 – *Diagramme de séquence pour la modification des données.*

3. Modifier la tâche du robot

Quand l'utilisateur veut modifier la tâche du robot, il l'a modifié via l'HMI pour que cette dernière envoie le changement voulu au robot, le robot à son tour change de comportement, la tache voulue est affiché sur l'HMI et le changement est effectué avec succès ;

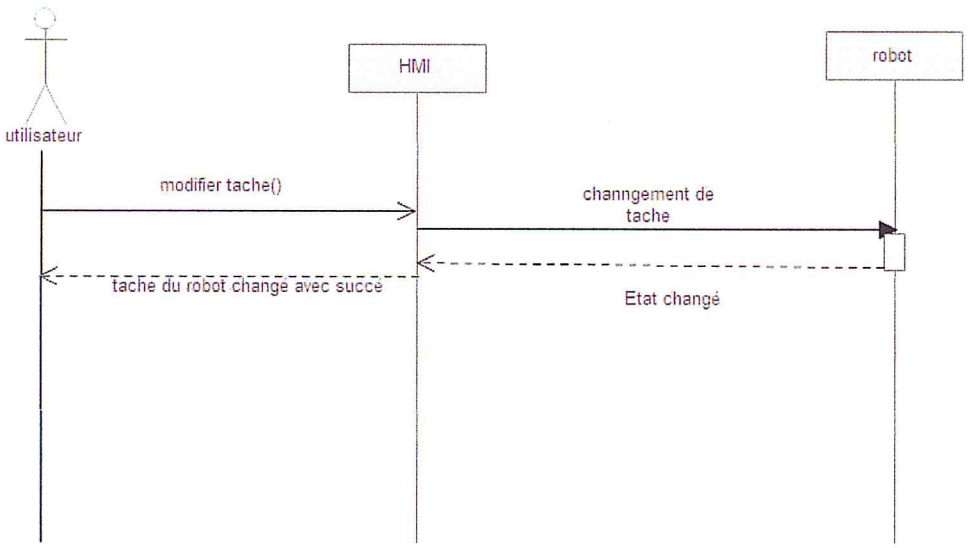


FIGURE 44 – *Diagramme de séquence pour la modification de la tache du robot.*

4. **Visualiser l'état de robot**

L'utilisateur peut vérifier et visualiser le changement d'états du robot, en plus savoir toutes les informations qui proviennent du robot avec un simple accès depuis l'HMI (toutes les informations possibles sont affichées sur l'HMI) ;

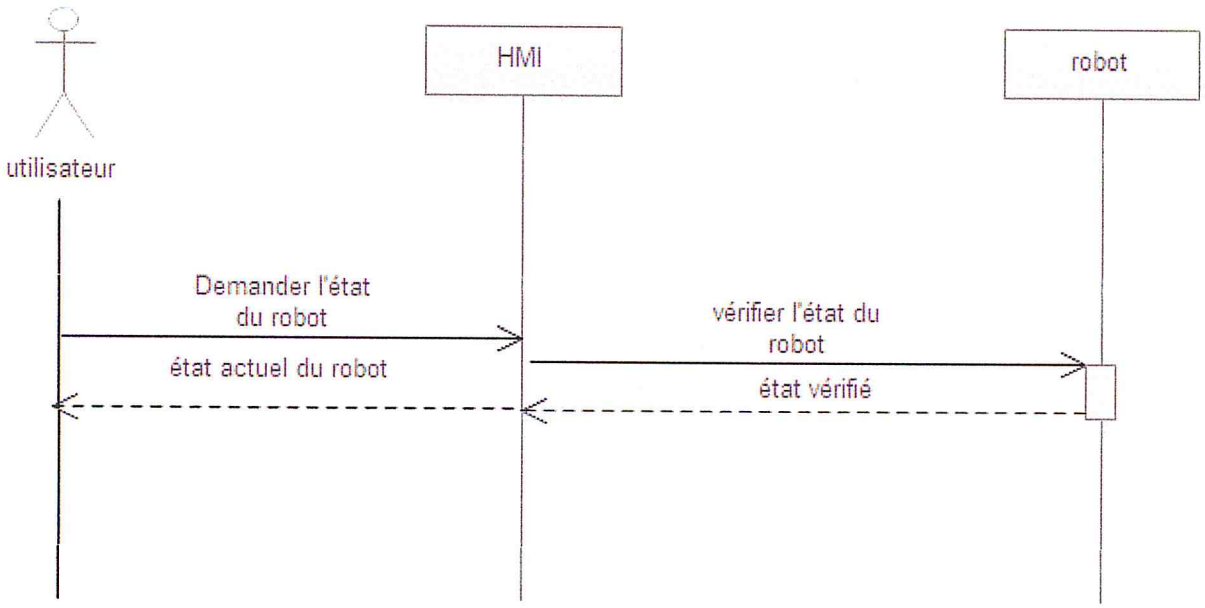


FIGURE 45 – *Visualiser l'état du robot.*

5. **Vérifier le bon fonctionnement de la cellule**

Lorsque l'utilisateur veut vérifier le bon fonctionnement de la cellule, il peut lire les informations qui résident dans l'automate à travers l'HMI, l'HMI récupère les paramètres depuis l'automate qui envoie à son tour des entrées/sorties pour que l'HMI les affichent. Les paramètres affichés doivent être vérifié par un expert pour qu'il puisse détecter des anomalies s'ils existent, en cas d'erreur il procède à modifier les paramètres qui ont causé l'erreur et enfin le bon fonctionnement est vérifié.

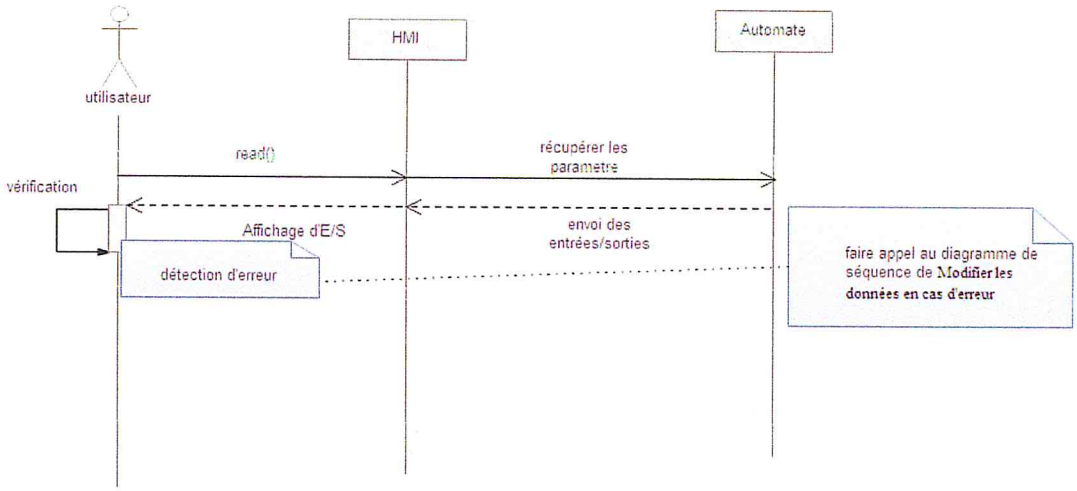


FIGURE 46 – *Vérifier le bon fonctionnement.*

Le diagramme d'état de transition

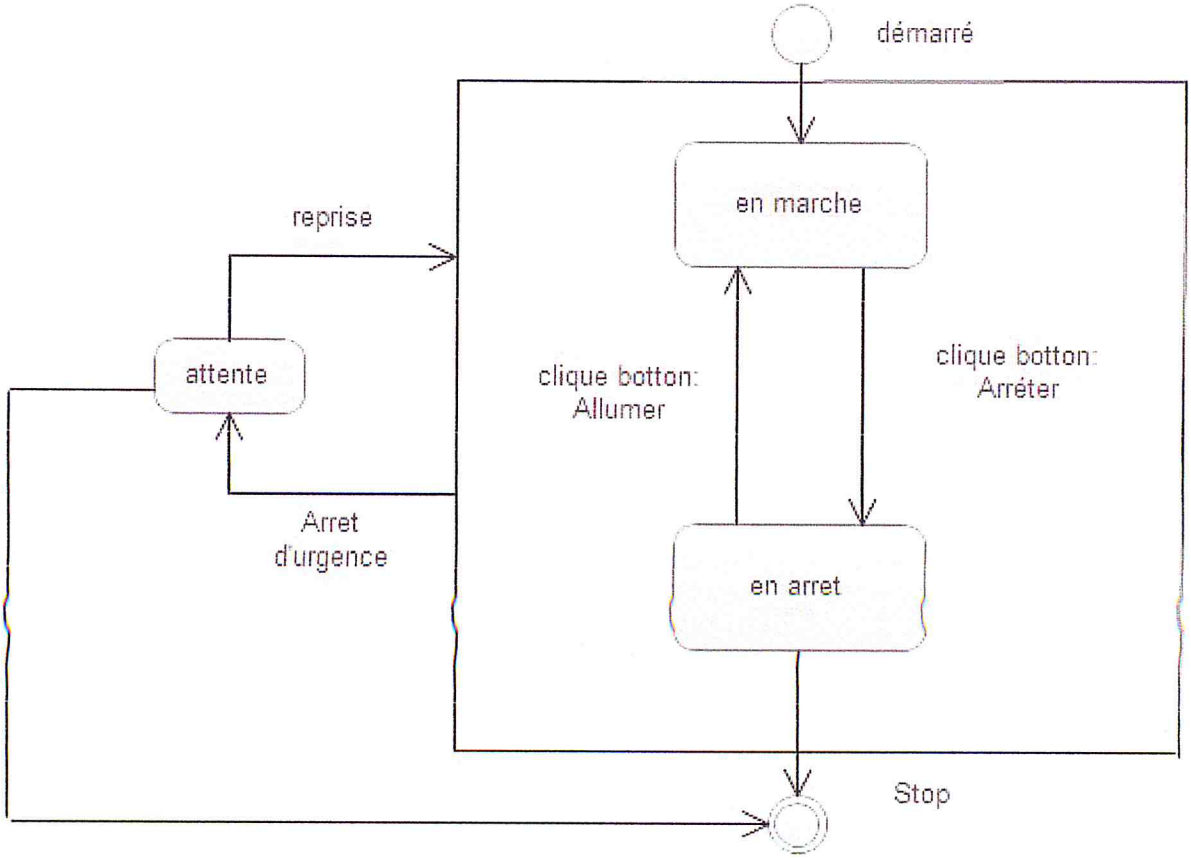


FIGURE 47 – *Diagramme d'état de transitions.*

3.6 Conlusion

Comme stipulé dans ce chapitre le but pour les développeurs est de relever le défi coté commerciale et coté logiciel pour satisfaire les besoins mais aussi pour gagner l'argent et être à la hauteur des attentes.

Nous avons choisi UML comme langage pour concevoir notre système puisque c'est le langage qui convient pour exprimer ce que nous attend de notre système, en choisissant Accord Uml comme méthodologie vu qu'elle nous a permis de mettre en valeur la notion temps réel également avec le processus de développement qui détermine la structure brique à brique de l'application (analyse préliminaire et analyse détaillé).

Chapitre 4

Implémentation et réalisation du système

4.1 Introduction

Dans le chapitre précèdent nous avons présenté la méthode AccordUML qui est utilisée pour la conception des systèmes ou la notion temps réel est fortement recommandée ; à savoir les deux briques de bases de cette méthodologie qui consistent à l'analyse préliminaire et l'analyse détaillée. Et comme notre système est un système temps réel nous avons employé cette méthode pour mieux décrire les fonctionnalités de notre système soit en ce qui concerne la vue interne ou externe du système, l'architecture générale du système explique d'une façon explicite les éléments nécessaires pour le fonctionnement du système et les diagrammes aussi.

Dans ce chapitre nous présenterons les démarches que nous avons adopté pour l'implémentation de notre système, sans oublié de passer par la description de notre cellule flexible et aussi les logiciels que nous avons utilisé pour aboutir aux résultats voulu,puis nous procéderons à la présentation de l'application, nous commencerons avec une simulation des fonctions programmées, ensuite nous concluons avec l'interface graphique.

4.2 Description de la cellule flexible

La cellule flexible de production de l'équipe SRP (Systèmes robotisés de production) du centre de développement des technologies avancées (CDTA) est un modèle typique d'une cellule industrielle vouée à des manœuvres d'assemblage et de manutention . Elle est constituée : d'un Robot 6 axes (GT6A), d'un convoyeur, d'un Automate programmable Modicon M340, des capteurs, d'un système RFID. La figure 48 présente la cellule robotisée :

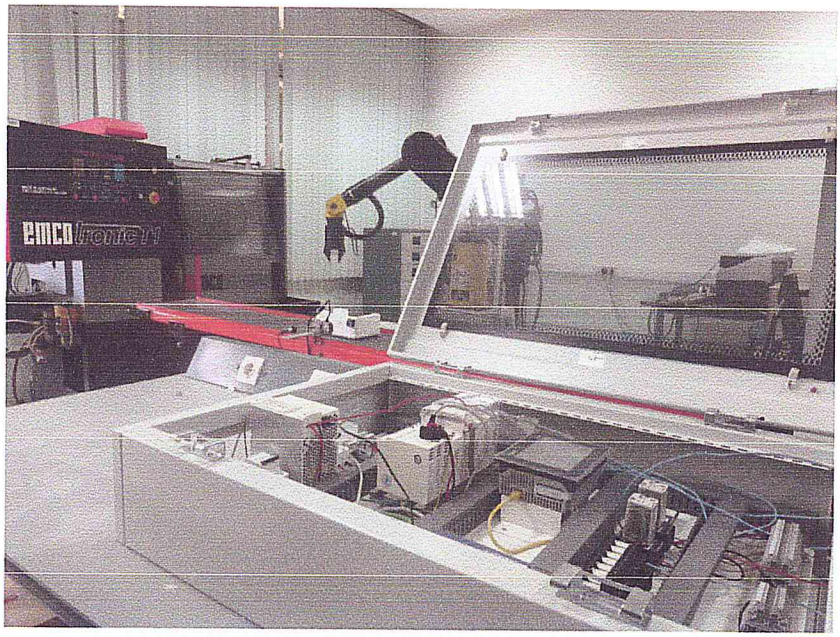


FIGURE 48 – *La cellule flexible de production.*

4.2.1 Automate programmable Modicon M340

La cellule flexible est équipée d'un API Modicon M340 comme illustré dans la figure 49, il pilote, et contrôle, les différentes composantes de la cellule :

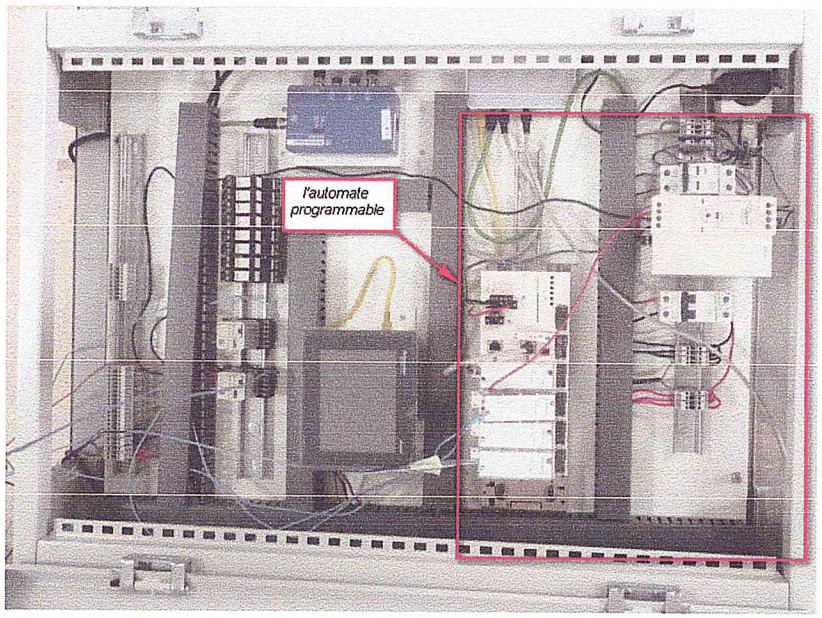


FIGURE 49 – *L'automate programmable Modicon M340.*

L'architecture de l'automate programmable Modicon M340

- **L'unité centrale CPU** :

Modicon propose quatre processeurs, qui se différent par leur capacités mémoire, vitesse de traitement, nombre d'E/S et nombre et type de ports de communication.

Le processeur de la cellule fléxible est le modèle BMX P34 2010, qui proposent de :

- 1024 entrées/sorties (Tout ou Rien) ;
- 256 entrées/sorties analogiques ;
- 36 voies métiers comptage.

Le processeur Modicon M340 BMX P34 2010 intègre :

- Un port Ethernet TCP/IP 10BASET/100BASETX ;
- Un bus machines & installations CANopen et une liaison série Modbus.

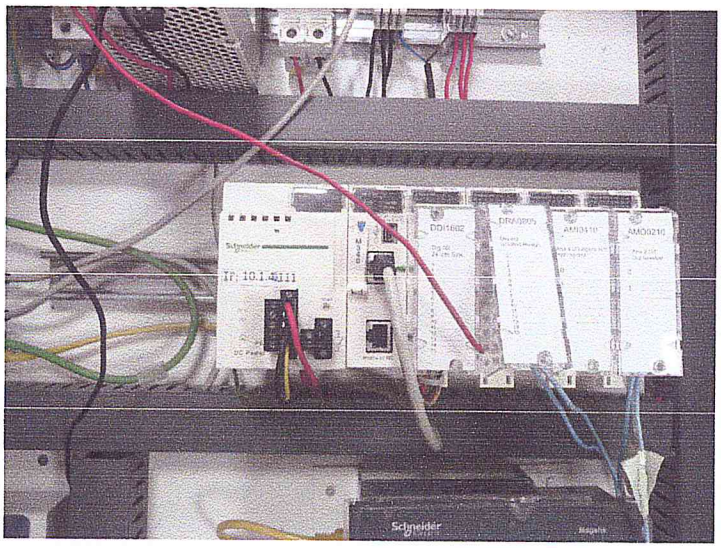


FIGURE 50 – *UC et les modules d'E/S Modicon M340.*

- **Le module d'alimentation :** Comme la cellule contient le processeur BMX P34 2010, La carte d'alimentation désigner sera le modèle BMX CPS 2010.

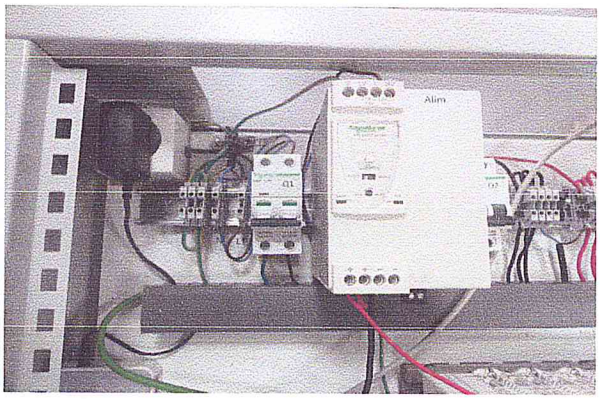


FIGURE 51 – *Alimentation modèle BMX CPS 2010.*

- **Modules d'entrée/sorties analogique :** Les modules d'entrées/sorties analogiques comprend un module d'entrées analogiques 4 voies rapides analogiques 16 bits, tension ou courant BMX AMI 0410 et un module de sorties analogiques 2 voies tension/courant BMX AMO 0210. Ils sont équipés d'un connecteur pour bornier débrochable 20 contacts ;
- **Modules d'entrée/sorties Tout ou Rien (DRA 0805 et DDI 1602) :** Les modules d'entrées/sorties "Tout ou Rien" de l'offre Modicon M340 sont des modules standards occupant un seul emplacement.

Les entrées reçoivent les signaux en provenance des capteurs et réalisent plusieurs fonctions (acquisition, adaptation, filtrage, protection contre les signaux parasites...).

Les sorties réalisent les fonctions de mémorisation des ordres donnés par le processeur, pour permettre la commande des préactionneurs au travers de circuits de découplage et d'amplification.
La figure ci-dessous montre les modules d'entrée/ sorties utilisées dans la cellule flexible de production.

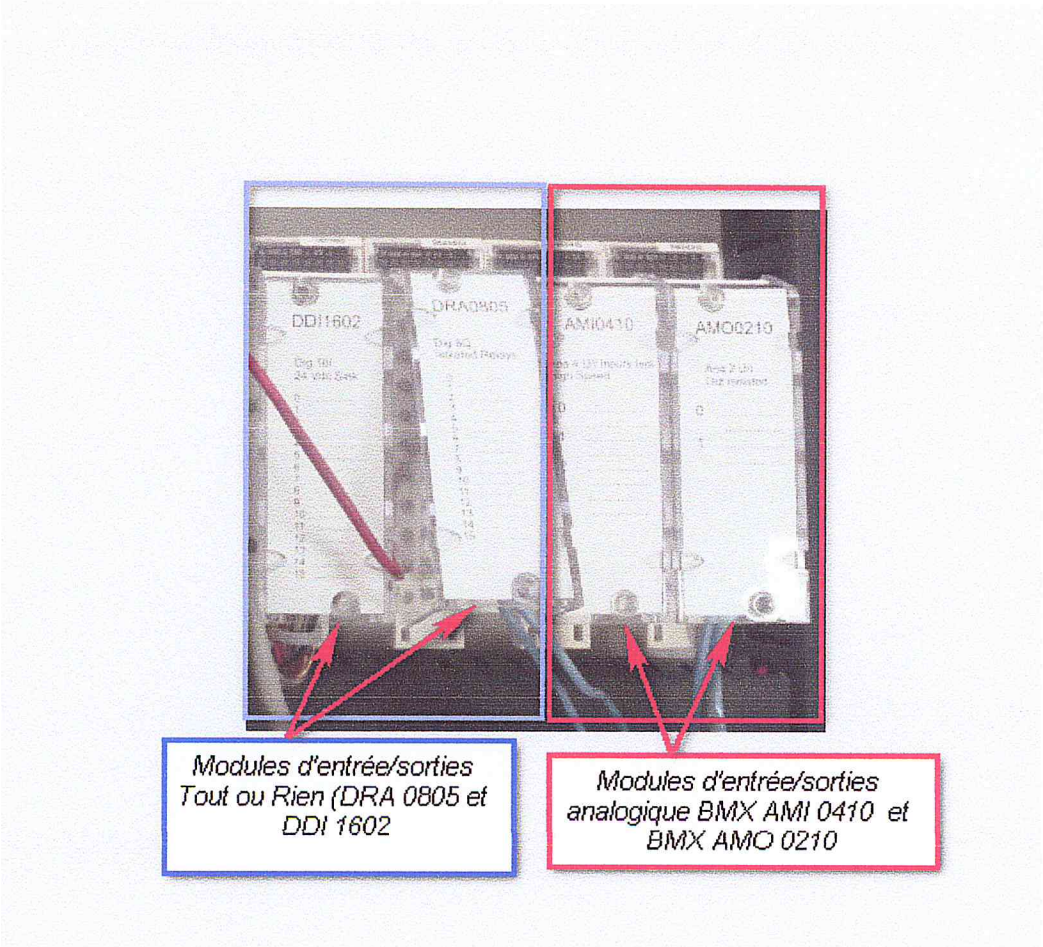


FIGURE 52 – *Modules d'E/S.*

4.2.2 Les capteurs de la cellule flexible

La cellule flexible est dotée de deux capteurs qui ont été placés sur le convoyeur afin de détecter la présence des objets, le premier est un capteur infrarouge à base d'émetteur et de récepteur alors que le deuxième est un détecteur photo électrique constitué aussi d'émetteur et de récepteur.

- **Le détecteur photos électrique :** Le détecteur photo électrique de chez Télémécanique se compose d'un émetteur de lumière (diode électroluminescente) affidé à un récepteur sensible à la quantité de lumière reçue (phototransistor), placés l'un en face de l'autre. Le phénomène de détection est effectué lorsque la pièce mécanique transperce le faisceau lumineux émis par le détecteur. Cette opération altère passablement la quantité de lumière reçu par le récepteur ce qui entraine un changement d'état de la sortie.

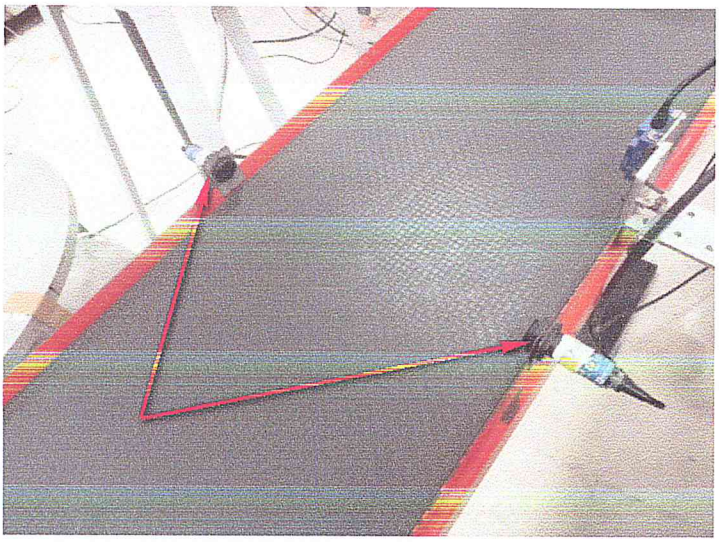


FIGURE 53 – *Détecteur photos électrique.*

4.2.3 Le convoyeur

Un convoyeur est un mécanisme ou machine qui permet le transport d'une charge isolée (cartons, bacs, sacs, ...) ou de produit en vrac (terre, poudre, aliments...) d'un point A à un point B.

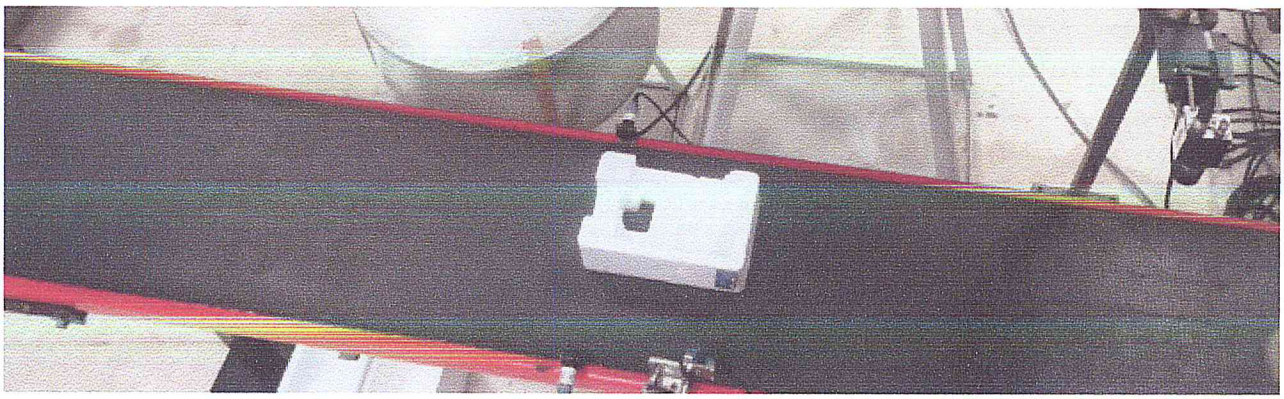


FIGURE 54 – *Convoyeur.*

4.2.4 Le bras manipulateur GT 6A

Le bras manipulateur est commandé par un ordinateur (partie commande) doté d'une carte de mouvement, cette dernière contrôle et pilote les axes et assure l'asservissement

L'étage de puissance (la partie puissance) va quand a elle fournir la puissance nécessaire pour faire fonctionner l'ensemble du montage, elle est en haute tension (220V) Le bras robotisé GT 6A comprend 6 degré de liberté, 3 axes destinés au positionnement et 3 axes à l'orientation, qui offrent la possibilité de déplacer et d'aiguiller les pièces mécaniques à partir du convoyeur vers la table tournante ou inversement.

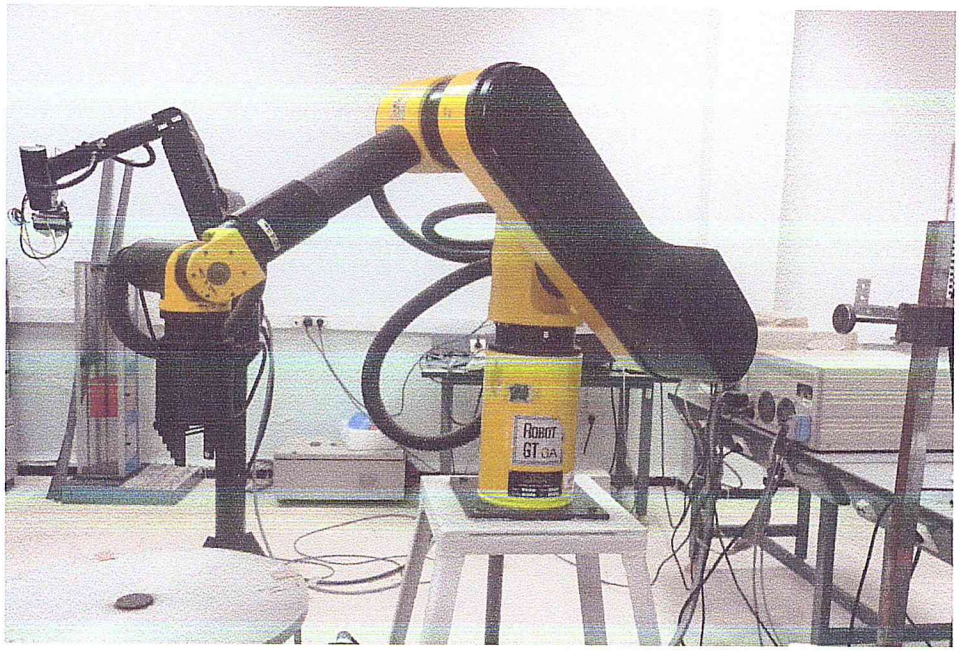


FIGURE 55 – *Le bras GT 6A.*

4.2.5 Le Système RFID

la cellule flexible constituée d'un système d'identification par radio fréquence Ositrack RFID (OsiSense XG) de Télémécanique (Schneider Electric). Ce système intègre le protocole Modbus TCP/IP. Il est composé des éléments suivants :

- **Une station compacte** (lecteur RFID Schneider XGCS4901201 CTR1058) : il s'agit d'une station compacte format plat 40 permet de lire et écrire des étiquettes RFID, il est caractérisé par :
  - Fréquence RFID : 13.56 MHz ;
  - Vitesse de transmission : 9600 bauds...115200 bauds (détection automatique) ;
  - Portée nominale : 10...70 mm ;

- tension d'alimentation : 24 V cc. conformément à Très Basse Tension de Protection ;
- Raccordement électrique : 5 broche(s)connecteur mâle déporté M12.

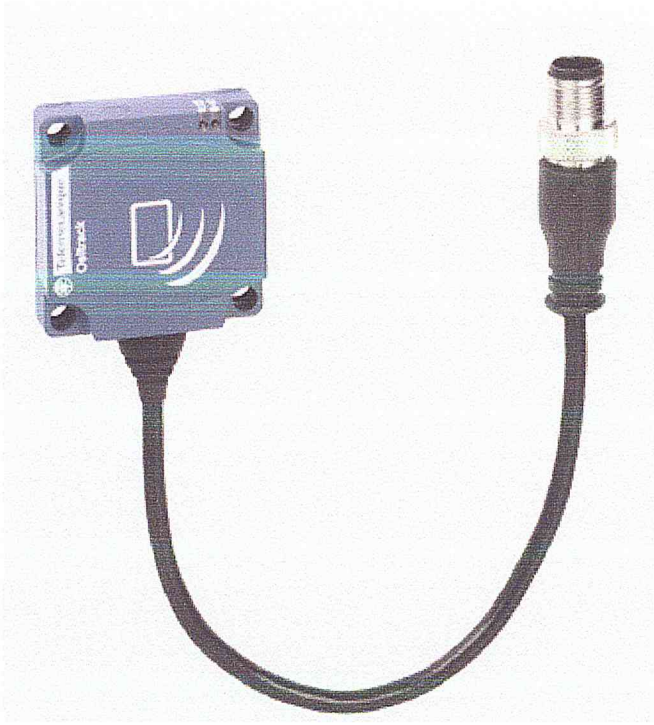


FIGURE 56 – *Lecteur RFID.*

- **Des étiquettes électroniques** :contiennent des informations sur le produit.

FIGURE 57 – *Etiquette RFID.*

La figure ci dessous présente la station compacte et l'étiquette de la cellule flexible :

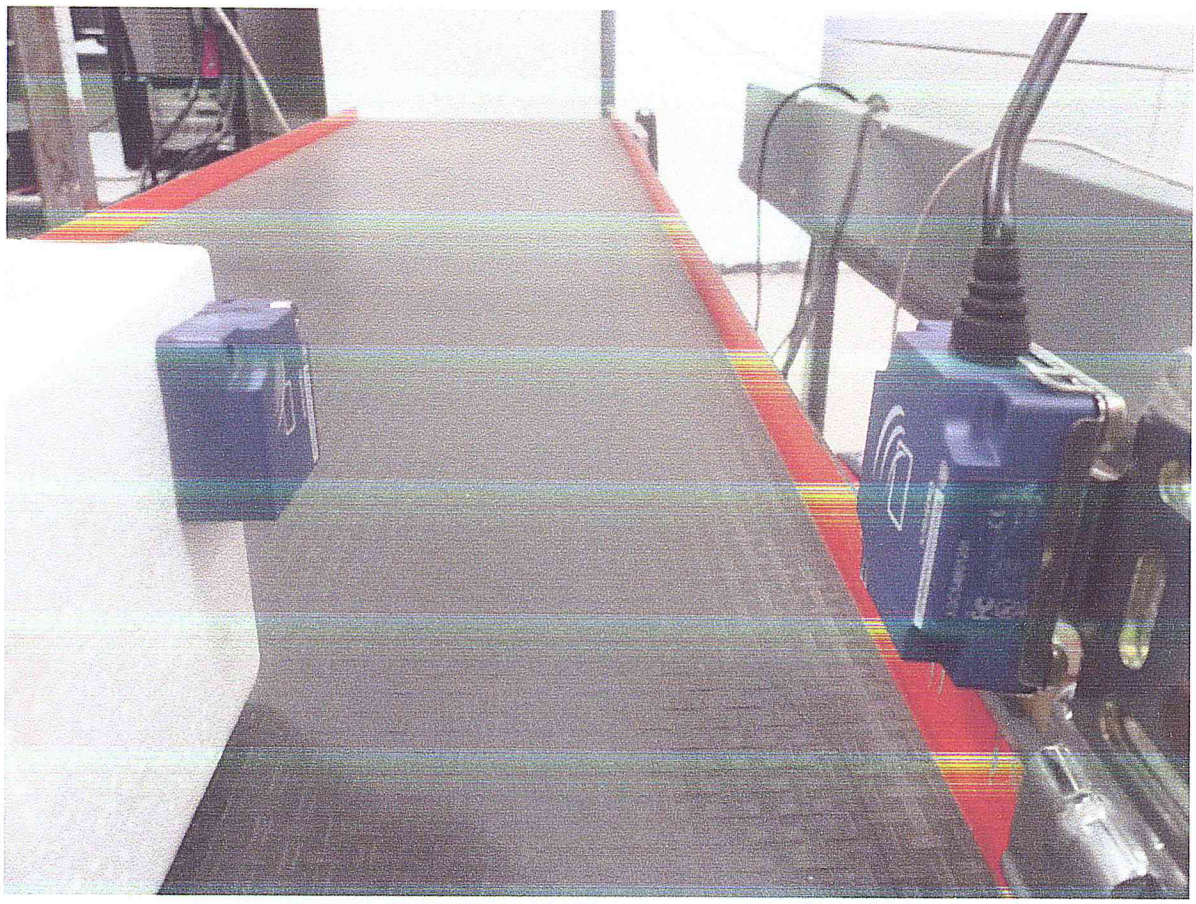


FIGURE 58 – *Lecteur et étiquette RFID de la cellule flexible de production.*

4.2.6 Une boîte de raccordement réseau (XGS Z33ETH)

Permet de raccorder la station compacte au réseau Ethernet (protocole Modbus TCP/IP). Elle permet aussi à l'automate ou PC d'accéder aux fonctions des stations (lecture/écriture d'étiquettes, commande et controle, surveillance...) La boîte XGS Z33ETH est munie de connecteurs M12. Elle est utilisée pour raccorder l'alimentation, le réseau Ethernet et 1 à 3 stations. Elle se présente sous la forme d'un boitier métallique étanche comprenant :

- Mise sous tension et voyants de signalisation du réseau Ethernet ;
- Une embase Ethernet type M12 codage D ;
- Une embase alimentation type M12 mâle, 4 contacts ;
- 3 embases type M12 femelle codage A, pour raccorder 1 à 3 stations XGC S.

FIGURE 59 – *La boîte Ethernet Ositrack XGS Z33ETH.*

4.3 Présentation des logiciels utilisés

Voici une présentation des différents logiciels utilisés au cours de ce travail :

4.3.1 Node js

java script coté serveur.

Le répertoire node _ modules avant l'instalation des modules

la figure ci dessous représente le répertoire qui contient les modules de Node

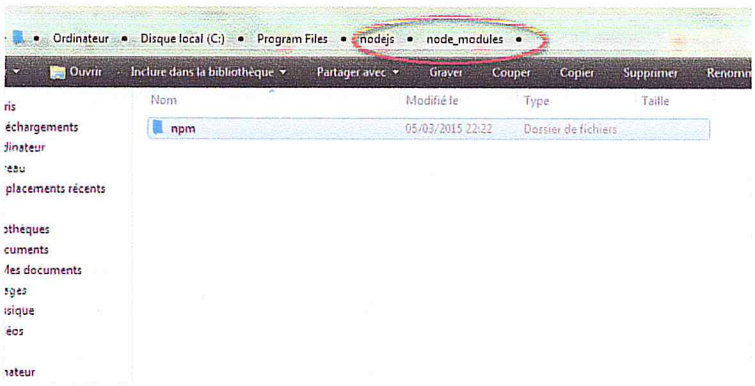


FIGURE 60 – *Le répertoire Node _ modules.*

Instalation des modules nécessaire avec npm

1. Express

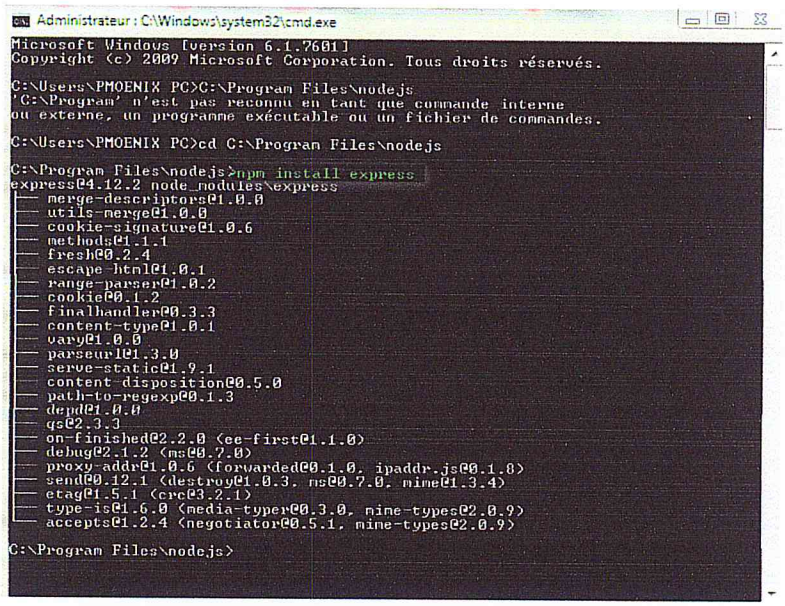


FIGURE 61 – *Installation du module Express.*

2. Socket.io

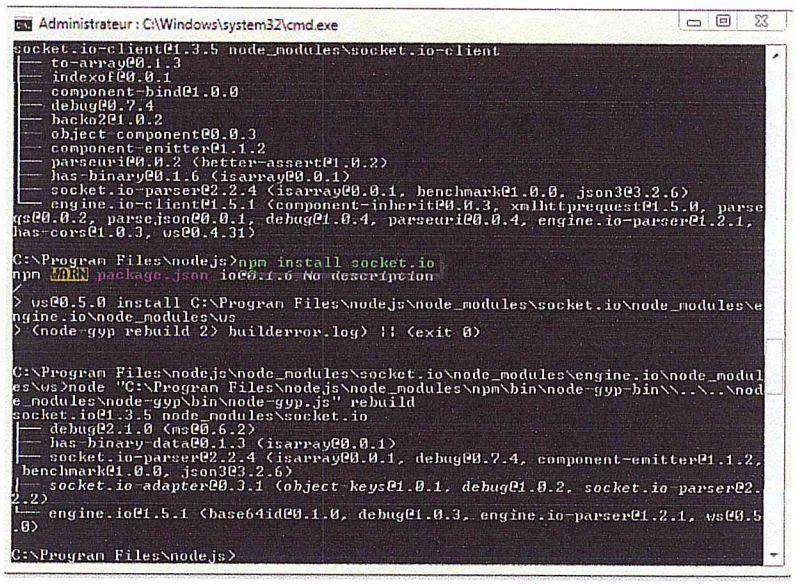


FIGURE 62 – *Installation du module Socket.Io.*

3. **Serialport**

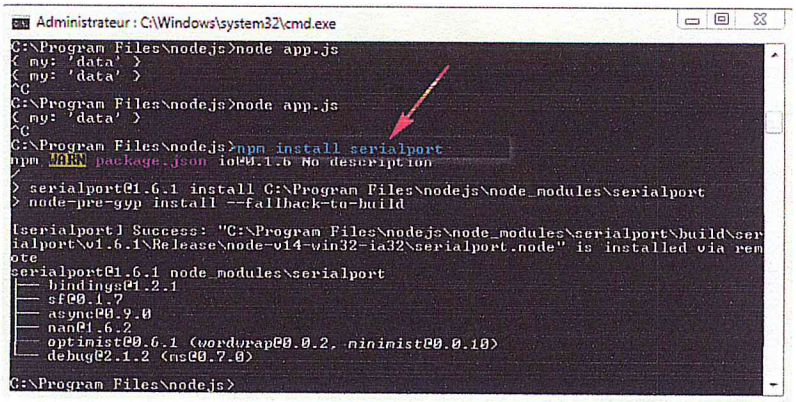


FIGURE 63 – *Installation du module Serialport.*

Le répertoire node_ modules après l'instalation des modules

la figure ci dessous représente le changement du répertoire node_ modules

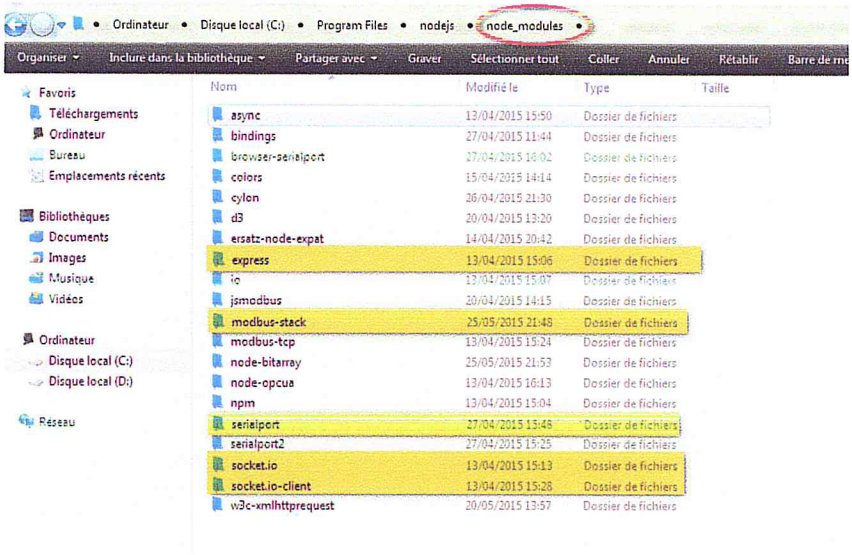
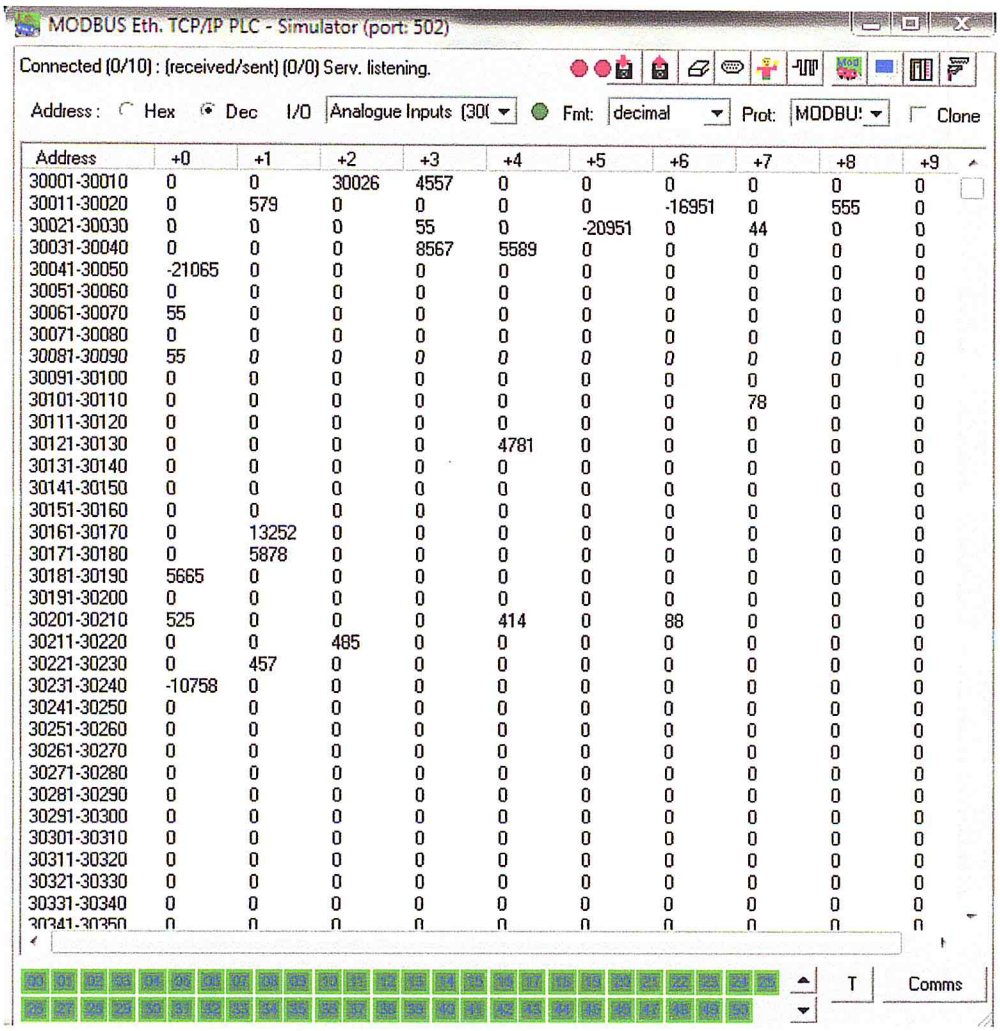


FIGURE 64 – *Changement du répertoire node _ modules aprés l'installation des modules .*

4.3.2 Le Simulateur ModRSsim

ModRSsim est une application permettant la simulation d'un serveur Modbus aussi appelé Slave supportant de nombreux protocoles tels que TCP/IP, RS 232, DF1. Son interface intuitive permet la visualisation en temps réel de l'état des Coils, valeurs booléennes variant de 0 à 1 en Modbus, ou des Registers, valeurs numériques variant de 0 à plus de 32000, elle permet également la modification des données par un simple double clique sur la valeur correspondante. Ce logiciel est une solution pour le test d'applications clients Modbus, aussi appelés Master, ou simplement pour apprendre les protocoles Modbus.



Address	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9
30001-30010	0	0	30026	4557	0	0	0	0	0	0
30011-30020	0	579	0	0	0	0	-16951	0	555	0
30021-30030	0	0	0	55	0	-20951	0	44	0	0
30031-30040	0	0	0	8567	5589	0	0	0	0	0
30041-30050	-21065	0	0	0	0	0	0	0	0	0
30051-30060	0	0	0	0	0	0	0	0	0	0
30061-30070	55	0	0	0	0	0	0	0	0	0
30071-30080	0	0	0	0	0	0	0	0	0	0
30081-30090	55	0	0	0	0	0	0	0	0	0
30091-30100	0	0	0	0	0	0	0	0	0	0
30101-30110	0	0	0	0	0	0	0	78	0	0
30111-30120	0	0	0	0	0	0	0	0	0	0
30121-30130	0	0	0	0	4781	0	0	0	0	0
30131-30140	0	0	0	0	0	0	0	0	0	0
30141-30150	0	0	0	0	0	0	0	0	0	0
30151-30160	0	0	0	0	0	0	0	0	0	0
30161-30170	0	13252	0	0	0	0	0	0	0	0
30171-30180	0	5878	0	0	0	0	0	0	0	0
30181-30190	5665	0	0	0	0	0	0	0	0	0
30191-30200	0	0	0	0	0	0	0	0	0	0
30201-30210	525	0	0	0	414	0	88	0	0	0
30211-30220	0	0	485	0	0	0	0	0	0	0
30221-30230	0	457	0	0	0	0	0	0	0	0
30231-30240	-10758	0	0	0	0	0	0	0	0	0
30241-30250	0	0	0	0	0	0	0	0	0	0
30251-30260	0	0	0	0	0	0	0	0	0	0
30261-30270	0	0	0	0	0	0	0	0	0	0
30271-30280	0	0	0	0	0	0	0	0	0	0
30281-30290	0	0	0	0	0	0	0	0	0	0
30291-30300	0	0	0	0	0	0	0	0	0	0
30301-30310	0	0	0	0	0	0	0	0	0	0
30311-30320	0	0	0	0	0	0	0	0	0	0
30321-30330	0	0	0	0	0	0	0	0	0	0
30331-30340	0	0	0	0	0	0	0	0	0	0
30341-30350	0	0	0	0	0	0	0	0	0	0

FIGURE 65 – *Modbus simulator .*

4.4 Présentation de l'application

4.4.1 Mode simulation

La fonction READ COILS

Son code fonction est **1**

- **Etat du simulateur**

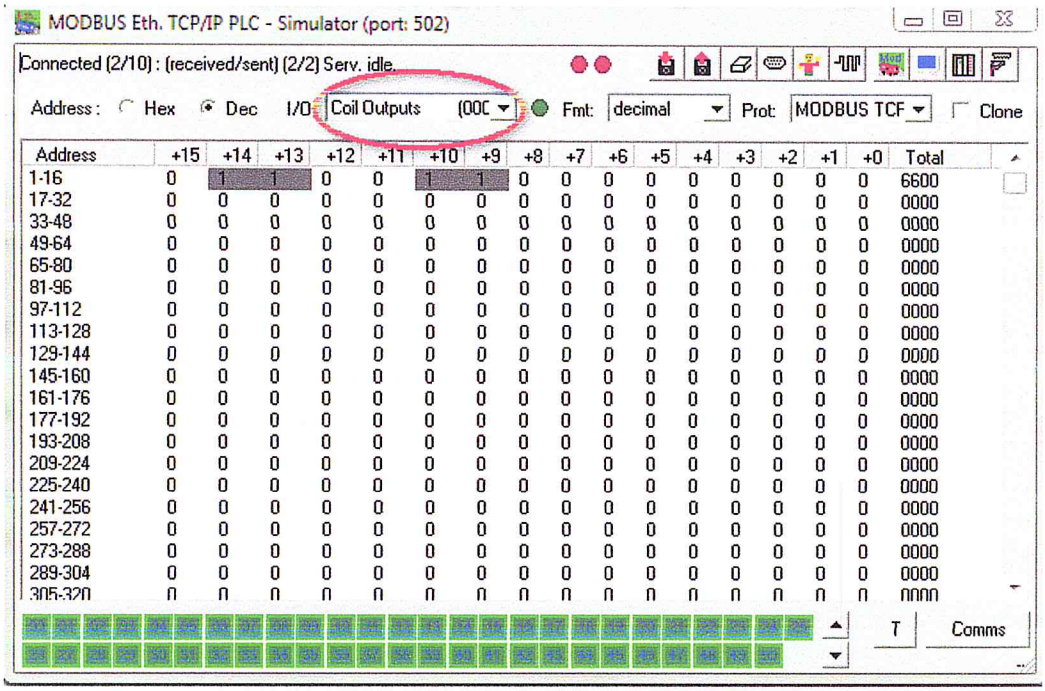


FIGURE 66 – *Simuler Read Coils.*

- **L'affichage en mode console** lire 9 adresses à partir de l'adresse 1 :

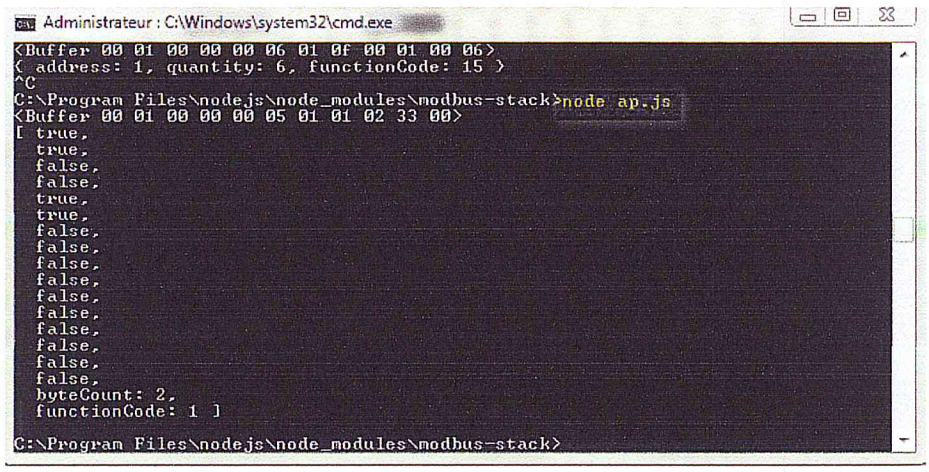


FIGURE 67 – *Read Coils en mode console.*

READ DISCRETE INPUTS

Son code fonction est **2**

- **L'état de simulateur et l'affichage mode console** : lire 8 adresses à partir de 2

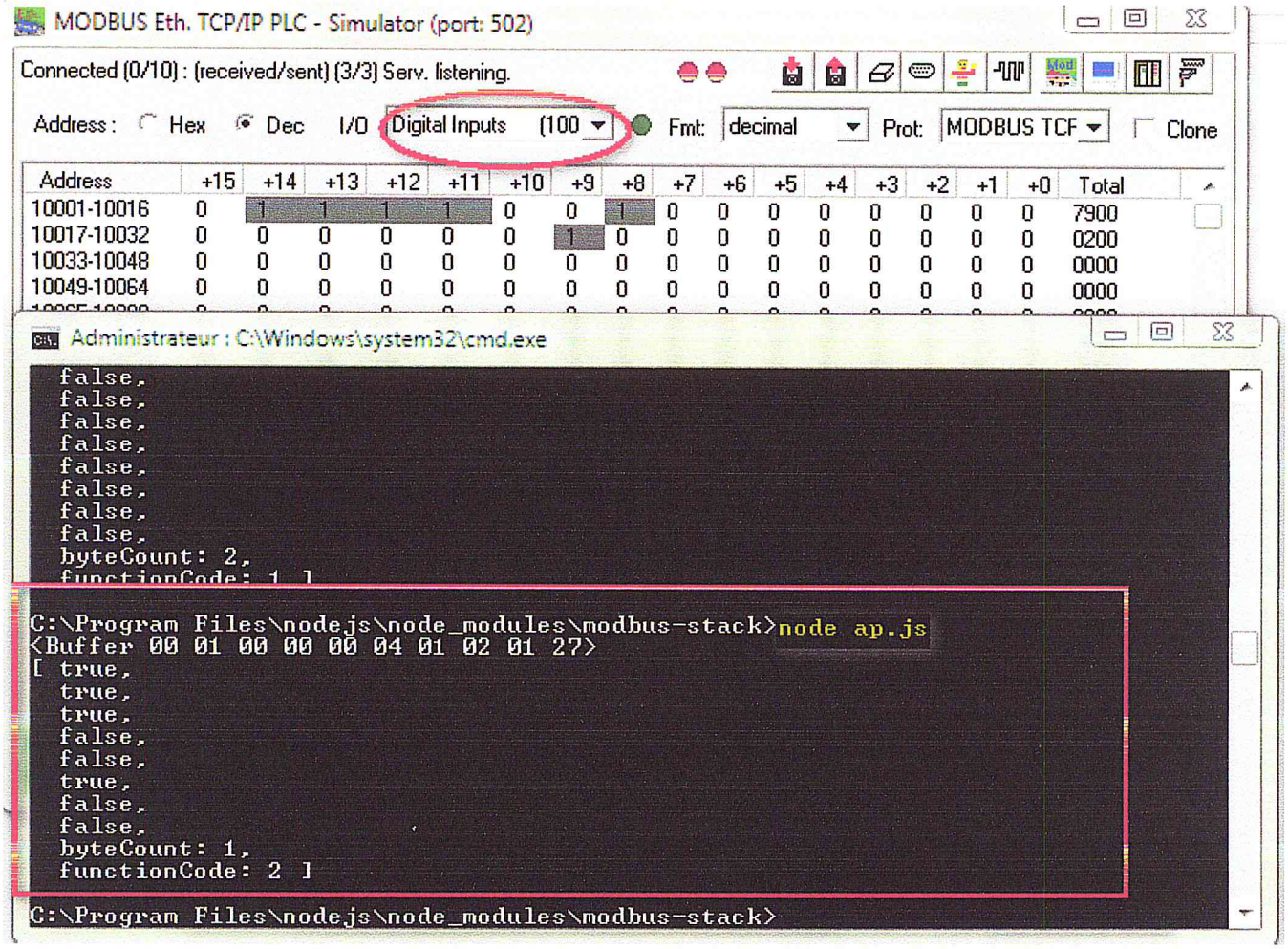


FIGURE 68 – *Simuler READ DISCRETE INPUTS.*

READ HOLDING REGISTERS

Son code fonction est **3**

Lire 21 adresses à partir de 9 :

- **Etat du simulateur et de la console**

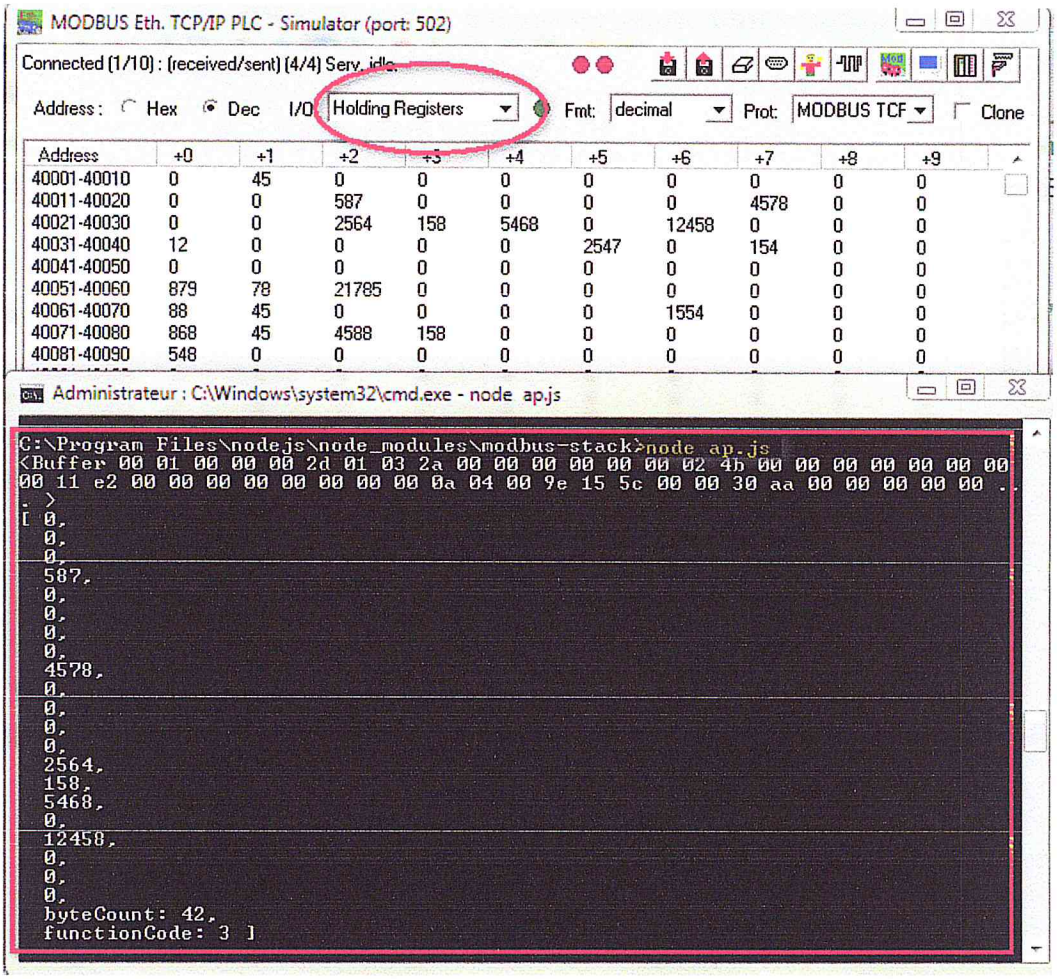


FIGURE 69 – *Simuler READ HOLDING REGISTERS.*

READ INPUT REGISTERS

La fonction READ INPUT REGISTERS : le code fonction= 4

- **Etat du simulateur**

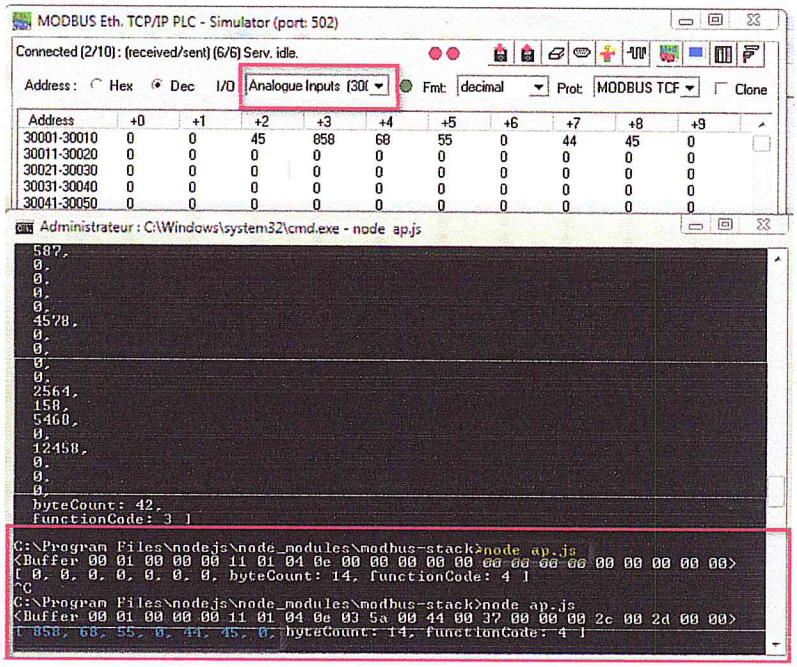


FIGURE 70 – *Simuler READ INPUT REGISTERS.*

Pour le reste de fonctions nous avant Forcé tous les registres à prendre les valeurs 0 :

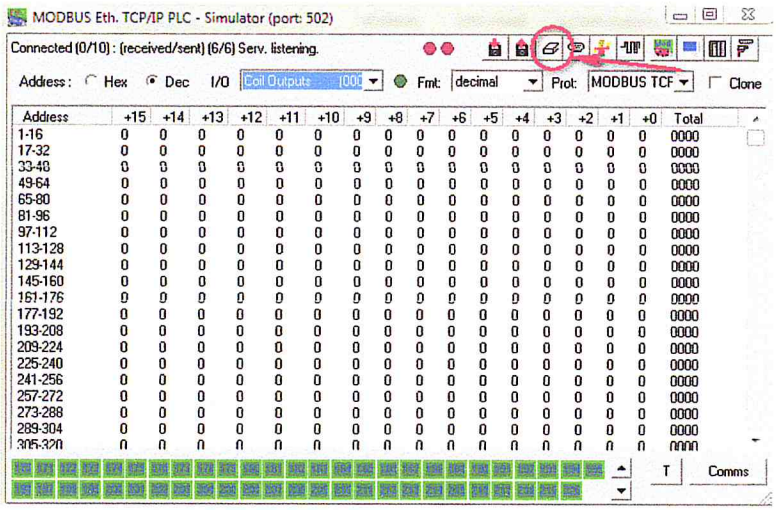


FIGURE 71 – *READ INPUT REGISTERS.*

La fonction WRITE SINGLE COIL : le code fonction= 5

Ecrire sur l'adresse 8 :

- **Etat du simulateur**

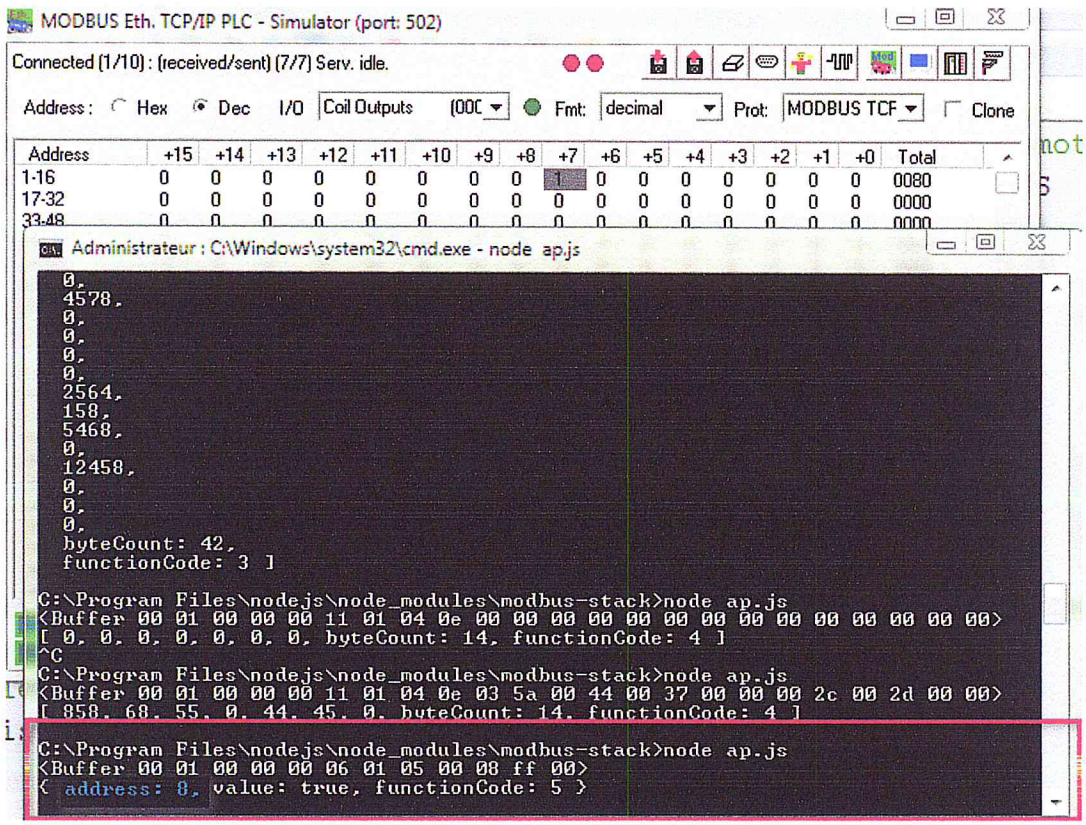


FIGURE 72 – *Simuler WRITE SINGLE COIL.*

La fonction WRITE SINGLE REGISTER : le code fonction= 6

Ecrire sur l'adresse 12 la valeur 865 :

- **Etat du simulateur**

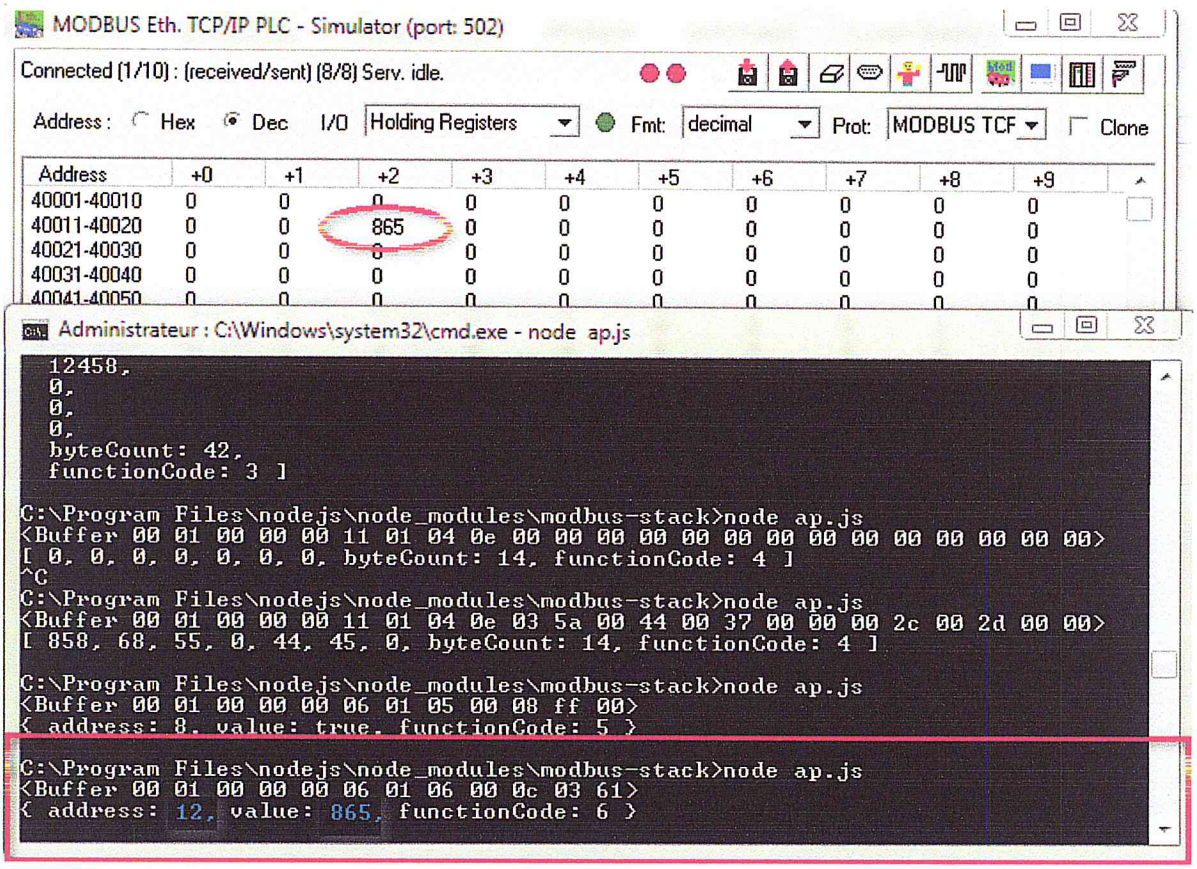


FIGURE 73 – *Simuler WRITE SINGLE REGISTER.*

La fonction WRITE MULTIPLE COILS : le code fonction= 15

Changer les valeurs à partir de 2 de prendre [true, true, false ; true, true, true, true] :

- **Etat du simulateur et la console**

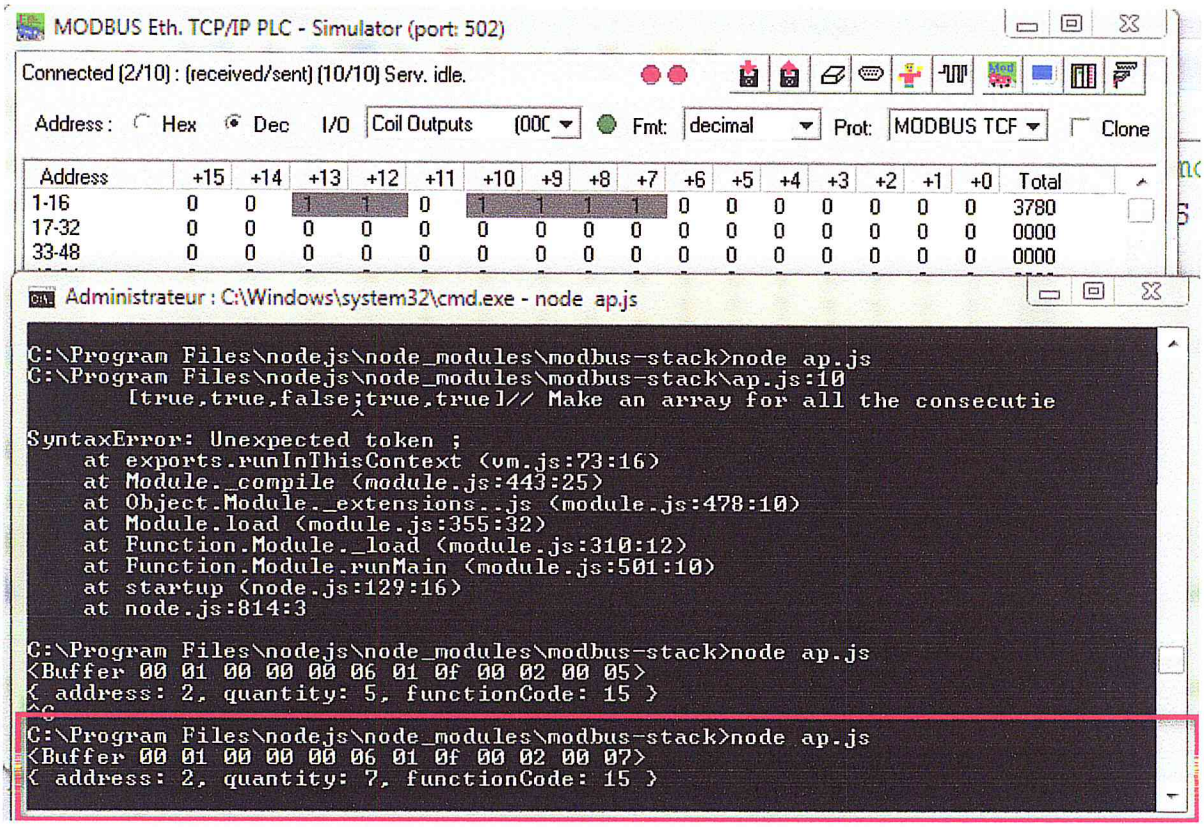


FIGURE 74 – *Simuler WRITE MULTIPLE COILS.*

La fonction WRITE MULTIPLE REGISTERS : le code fonction = 16

Changer les valeurs à partir de 6 de prendre [145,47,789,145,458,974,4,0,0,0,1,56] :

- **Etat du simulateur et la console**

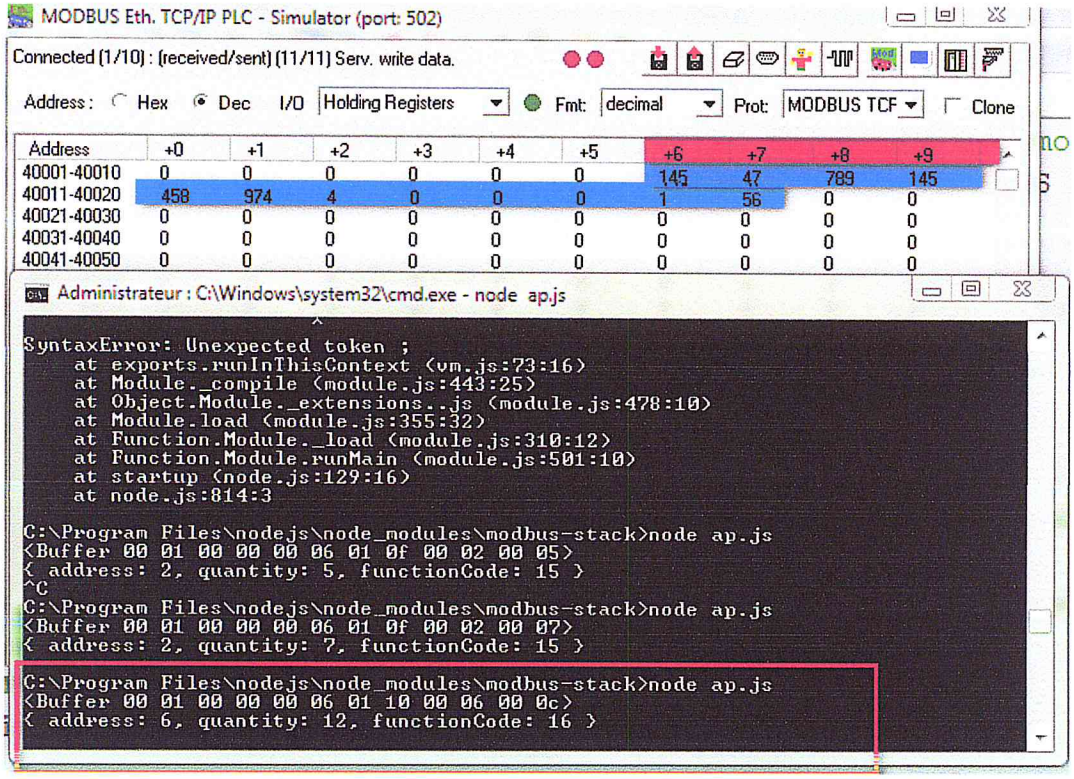


FIGURE 75 – *Simuler WRITE MULTIPLE REGISTERS.*

Remarque Importante : Cette simulation est effectuée non seulement sur localhost mais aussi entre deux pc un pc comme serveur et l'autre comme client.

4.4.2 Présentation de l'interface

Page d'accueil

Pour accéder à l'interface, il faut entrer un pseudo :

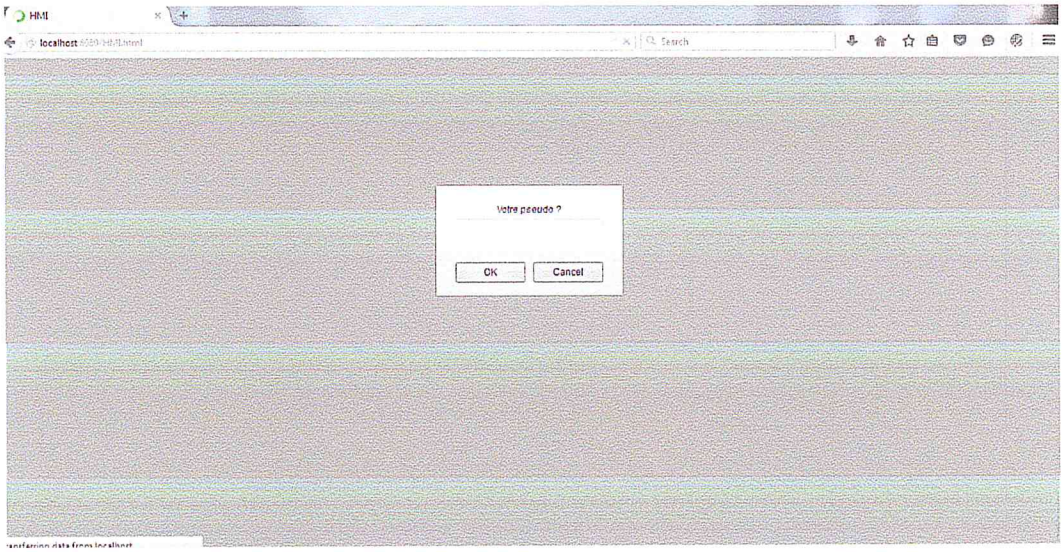


FIGURE 76 – *Accès à l'interface.*

Et puis la page d'accueil s'affiche :

FIGURE 77 – *Page d'accueil de l'interface.*

Les informations du robot

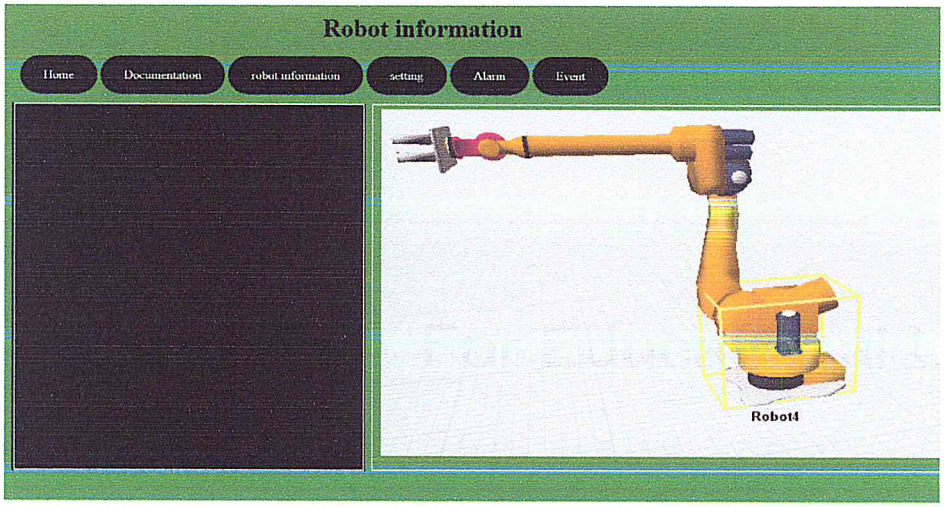


FIGURE 78 – *Les informations du robot.*

la page de documentation

Cet onglet représente une description de la cellule :

FIGURE 79 – *page de documentation.*

4.5 Conclusion

Dans ce chapitre après avoir décrit la cellule flexibles et les équipements qu'elle contient un par un, en se basant sur le rôle de chaque instrument.

Nous avons présenté notre application en mode simulation entre le serveur qui est le simulateur et le client qui lui-même lis et écris sur le simulateur de l'automate. Nous avons mentionné que cette opération est même réalisable entre deux pc. Ensuite nous avons donné une vue sur notre interface graphique.

Conclusion générale et perspectives

Tout d'abord, nous tenons à souligner l'expérience très enrichissante acquise au cours du stage au sein du Centre de Développement des Technologies Avancées, pour l'élaboration de ce projet de fin d'étude. Le travail qui nous a été demandé consiste à réaliser un système pour la supervision d'une cellule robotisée. La cellule comprend deux automates, un système de convoyage, un système de lecture RFID et un robot industriel.

Nous avons réalisé un système de supervision qui consiste à se connecter à l'automate via le Modbus TCP/IP pour lire des entrées sorties qui sont en relation avec le produit capté avec le lecteur RFID, nous avons non seulement arrivé à lire des entrées sorties depuis l'automate mais aussi à les modifier. Il suffit qu'un ou plusieurs client ont une adresse dans le méme sous réseau que le serveur et la connexion est établie. Cela peut effectuer sur le localhost ou sur deux pc différents. Néanmoins, durant la réalisation de ce projet plusieurs difficultés ont été rencontrées. Sans être exhaustif, nous citons :

- le choix le logiciel SCADA convenable pour le développement de notre système, au début nous avons choisi Eclipse SCADA mais sa configuration a été complexe (1 mois de recherche) et aussi il ne possède pas d'outils graphiques cela ne permet pas de visualiser le procédé. Puis, nous avons choisi Mango qui au début nous a semblé parfait surtout que c'est un logiciel basé sur le web mais nous avons découvert que c'était juste une version de démonstration (non opensource et non gratuit).
- Comme Node.js est une technologie jeune, il existe peu de documentation claire et pas de communauté française.
- Le protocole de communication qu'on devrait utiliser au début qui est OPC UA ne convenait pas à Node.Js puisque il est un peu vieux alors nous avons été obligé de trouver un autre protocole qui fonctionne avec Node.JS.
- Le téléchargement de la page HTML (l'interface graphique) lors du lancement du serveur nous a posé problème.

Au cours de ce projet nous avons acquis beaucoup de connaissances que ce soit sur le plan hard ou sur le plan soft. En premier lieu nous avons eu de la chance d'interagir directement avec des équipements industriels, de comprendre leur rôle dans notre cellule et même de les manipuler par leurs logiciels de base, nous avons découvert que l'automate est l'élément principal de la cellule.

En deuxième lieu nous avons appris le langage JavaScript qui est le langage de programmation pour Node.js et le HTML 5. Sans oublié la manipulation du logiciel de simulation et aussi les principes de base pour Modbus TCP/IP.

Avec l'apprentissage de Node.js nous avons découvert que Node.js permet de faire beaucoup de chose dans développement web. Node.js représente un véritable plus pour nous

dans ce projet et nous ouvre d'autres perspectives d'avenir en raison de rapidité, multiplateformes (mobile, desktop...) et mode asynchrone qu'il offre pour le développeur surtout dans le domaine de supervision industrielle. Comme perspectives le travail peut être amélioré. D'un côté, La possibilité de tester la communication pour d'autres types d'automates et de protocoles. De l'autre coté, il est également parfait de développer une IHM qui permet l'échange des messages et donne une vue dynamique aux utilisateurs.

Bibliographie

[1] *Des applications ultra rapide avec Node.Js*. novembre 2013. www.openclassrooms.

[2] *Etude approfondie du protocole Modbus*. www.ni.com, aout 2014.

[3] Maintenance des grands systémes, 2014.

[4] *Panorama E2*, codra edition, mars 2015.

[5] Carlos Daniel GARCIA BELTRAN. *OutilsS pour l'aide á la supervision de procédés dans une architecture multiagents*. PhD thesis, INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE, septembre 2004.

[6] Philippe Auroux. La supervision industrielle. décembre 2012.

[7] Slim Bensaoud. *Automatismes*, chapter Les automates programmables industriels. octobre 2004.

[8] Salomé Benslama. Mise en place d'un sytéme domotique et d'un scada pour la gestion des fluxénergétiques de la plateforme minotoring et habitat intelligent de predis. Technical report, Laboratoir G-SCOP,Grenoble INP, 2010.

[9] Ikhlef Boualem. *Contribution á l'étude de la Supervision Industrielle Automatique dans un environement SCADA*. PhD thesis, Université M'hamed Bougara de Boumerdes, 2009.

[10] Belkacem Ould Bouamama. Conception intégrée de systémes de supervision. Master's thesis, Université Lille 1, 2010.

[11] Fabien Chiron. *Contribution á la flexibilité et á la rapidité de conception des systèmes automatisés avec l'utilisation d'UML*. PhD thesis, Université Blaise Pascal de Clermont Ferrand, 2008.

[12] Grégory Cladera. *Supervision en mode SaaS*. RG supervision, juin 2012.

[13] R Clavel. *Robots industriels*, chapter 1. 2009.

[14] Vincent Cocquempot. *Contribution á la surveillance des systémes industriels complexes*. PhD thesis, UNIVERSITE DES SCIENCES ET TECHNOLOGIES DE LILLE, decembre 2007.

[15] M. Combacau. Systémes de production surs de fonctionnement, mars 2000.

[16] Ministére de la santé. Directives nationales de supervision des acteurs du systéme de santé. République du Bénin, mai 2010.

[17] DITEL, 08915 Badalona (Barcelona) Espagne. *Communication avec les afficheurs DITEL série DINOS-DMG en protocole MODBUS*, juin 2013.

[18] Pascal Duval. Gestion de production ou logistique industrielle, juillet 2010.

[19] Fanuc. *Réparer son vaisseau.* Fanuc Robotics, juin 2014.

[20] Benjamin FONTAN. *MÉTHODOLOGIE DE CONCEPTION DE SYSTéMES TEMPS REEL ET DISTRIBUéS EN CONTEXTE UML/SysML.* PhD thesis, UNIVERSITE TOULOUSE III PAUL SABATIER, janvier 2008.

[21] Jérome fourrier. Le protocole modbus tcp/ip. Lycée des Métiers Jacques Prévert, 2010.

[22] Sébastien Gerard. Using the model paradigm for real times systems developement accord/uml. *CEA*, 2009.

[23] Merlin Gerin. *Les bus et les réseaux de terrain en automatisme industriel.* SchneiderElectrics, novembre 2002.

[24] Stéphane Goze. Mangement des organisations, juillet 2006.

[25] Andre Grujovski. Intouch, janvier 2009.

[26] REZIGUE HAMZA. *Modélisation, Surveillance d'une centrale à Beton par l'outil BOND GRAPH et validation en ligne par un systeme SCADA.* PhD thesis, Université Sétif (Algérie), 2014.

[27] Philippe Hoaro. les automates programmable industriels. mai 2007.

[28] Jérome. Laison serie modbus, février 2013. Lycée Gustave Eiffel.

[29] C jossin. Autom buts de l'automatisme, septembre 2009.

[30] Bacem Jrad. Systèmes automatisés.

[31] Béla Juhasz. *DEVELOPING M2M APPLICATIONS WITH MANGO.* PhD thesis, JAMK UNIVERSITY OF APPLIED SCIENCES, december 2009.

[32] Manuel Kiessling. *The Node Beginer Book.* mars 2012. www.developpez.com.

[33] Hermann Kopetz. *Design methods and applications for disributed Embedded systems.* Kluwer Academic Publishers, 2004.

[34] L.Bergougnoux. Api les automates progrmmables. Polytech Marseille Département de Mécanique Energétique.

[35] Hadrien Lleida. La robotique, avril 2007.

[36] Hakima Manseri. Mise en oeuvre d'un systéme de supervision au sein du service si2 de lirmm. Technical report, Université de Montpellier 2, septembre 2009.

[37] M.Caussade. Cours d'automatisme. avril 2008.

[38] Houda Bel Mokadem. *Vérification des propriétés temporisées des Automates Programmables Industriels*. PhD thesis, Ecole Normale Supèrieure de Cachan, 2006.

[39] Benkhelouf Ikram Nouar Nour Elhouda. *Administartion et configuration d'un outil de supervision ZABBIX sous linux*. PhD thesis, Université Abou Bakr Belkaid Tlemcen, juin 2014.

[40] John Querry. Master terminal units mtu in scada, juin 2011.

[41] Daniel RACOCEANU. *Contribution à la Surveillance des Systémes de Production et Utilisant les Techniques de l'Intelligence Artificielle*. PhD thesis, Université de Franche-Comté, 2006.

[42] MAHADOUI RAFIK. *Diagnostic industriel par Neuro-Flou-Application á un systéme de production*. PhD thesis, Université de Batna, mars 2008.

[43] Henrik Rydstedt. Html5 as hmi in a command and control system, march 2014. University UPPSALA.

[44] Schneider Electric. *SCADA Systems*, march 2012.

[45] Marcos DA SILVEIRA. *SUR LA DISTRIBUTION AVEC REDONDANCE PARTIELLE DE MODELES A EVENEMENTS DISCRETS POUR LA SUPERVISION DE PROCEDES INDUSTRIELS*. PhD thesis, L'Université Paul Sabatier, octobre 2003.

[46] Ahmed Skaf. *ETUDE D'UN SYSTEME DE SUPERVISION ET DE COMMANDE D'UN PROCEDE COMPLEXE COMME UN ELEMENT DE BASE D'UNE ORGANISATION DISTRIBUEE COMPRENANT DES MACHINES ET DES HOMMES*. PhD thesis, UNIVERSITE JOSEPH FOURIER, decembre 2001.

[47] Outhman Souli. Mise en place d'un systéme de supervision open source. Technical report, Université Virtuelle de TUNIS, 2011.

[48] Yan Tanguy. Transformation accord/uml vers signal. Technical report, CEA/SACLAY, février 2002.

[49] Moussa Amadou TRAORE. *Supervision adaptative et pronostic de défaillance pour la maintenance prévisionnelle de systémes évolutifs complexes*. PhD thesis, Université Lille 1, décembre 2010.

[50] YNAUDE. Introdduction ethernet industriel, juin 2011.