REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE ABDELHAMID IBN BADIS DE MOSTAGANEM

Faculté des Sciences et Sciences de la Technologie

Département de Mécanique

MEMOIRE

Pour l'obtention du Diplôme de Magister

Spécialité : Génie Mécanique

Option : Modélisation, Simulation et Calculs Scientifiques Appliqués

---

Présenté par Monsieur

OUINAS Amine

CONCEPTION ET REALISATION D'UNE PLATEFORME WEB POUR UNE GESTION DE PRODUCTION DISTRIBUEE: CAS D'UN ATELIER DE FABRICATION DE PIECES MECANIQUES

Soutenue 1 Juin 2015 devant le **JURY :**

Président	: Mr N. Retiel	Professeur (U. Mostaganem)
Examinateur	: Mr A. Hebbar	Professeur (U. Mostaganem)
Encadreur	: Mme N.Taghezout	Maître de Conférences A (U.ORAN 1)
Co-encadreur	: Mr M. Sahnoun	Maître de Conférences B (U. Mostaganem)

Dédicaces

Au nom de Dieu le tout miséricordieux, le très miséricordieux.

Mon devoir est rendu grâce à Dieu qui m'a donnée la patience et le courage pour terminer ce modeste travail.

Je dédie ce travail à mes parents, à mes sœurs et frères et à toute ma famille, aussi qu'à tous ceux qui de prés ou de loin ont contribué à la réalisation de ce travail.

Remerciements

Louange à Allah le clément, le miséricordieux, pour son infinie bonté et sa miséricorde dont il nous a comblés, et nous a permis d'achever ce travail. Nous le prions pour qu'on puisse bénéficier des fruits de ce labeur.

J'exprime ma sincère gratitude et remerciements à mon encadreur Dr. **N.TAGHEZOUT,** MCA à Université d'Oran pour ses conseils précieux, son aide dans le cheminement de cette étude et pour la peine qu'elle s'est donnée tout au long de ce travail afin de faire de ce document ce qu'il représente.

Je remercie les membres de jury qui nous ont fait l'honneur d'avoir accepté d'examiner ce modeste travail.

Mes sentiments et profonde gratitude vont à mes précieux parents pour leur amour, affection et compréhension, à mes chers frères, surtout Djamel sans oublier mes très chères sœurs et à mes professeurs qui tout au long ces années d'études m'ont transmis leur savoir sans réserve. et tous mes amis pour leur soutien.

Résumé

Ce projet s'inscrit dans le cadre des travaux de recherche en informatique industrielle qui porte sur le développement d'outils et plateformes informatiques pour la gestion de production dynamique. Il s'agit de mettre en place une technologie WEB au service d'un programme de contrôle de gestion de production dans un atelier de fabrication de pièces mécaniques. La plateforme est développée en exploitant toutes les possibilités de communication et d'interaction entre les agents de production qui sont offerts par la plate-forme de la toute récente technologie JEE ; Java Platform Enterprise Edition d'Oracle. La plate-forme Java s'impose comme solution professionnelle de haut niveau dans l'entreprise. Notre platforme **IT technologies** offre des services aux clients qui désirent faire des commandes en ligne de produits visualisés sur la page d'acceuil ou des commandes selon des conceptions à faire valider par l'agent dessinateur. Notre plate-forme est conviviale et très simple à utiliser.

Mots-clés : Conception assistée par ordinateur, Agent Designer, gestion de production distribuée, Pièce mécanique, plateforme WEB.

Absract

This project is a part of the industrial informatics research that focuses on the tools development and software platforms for dynamic production management. This is to set up a Web technology to a production management control program in a workshop manufacture of mechanical parts. The platform is developed by exploiting all the possibilities of communication and interaction between the production agents that are offered by the platform of the recently JEE technology; Java Platform Enterprise Edition of Oracle. The Java platform itself is considered as professional solution for high level in the company.

Our technology platform **IT technologies** provides services to clients who wish to make online orders for products displayed on the home page or commands according to designs to be validated by the designer agent. Our platform is user friendly (convivial) and easy to use.

Key-words: Computer Aided Design, Designer agent, distributed production management, Mechanical parts, WEB platform.

Sommaire:

I. INTRODUCTION 1

II. CHAPITRE 1: GESTION DE PRODUCTION DISTRIBUEE 4

1.1 Introduction 4

1.2 Définitions de la gestion de production 5

1.3 Objectifs de la gestion de production 6

1.3.1 Respecter les délais 6

1.3.2 Minimiser les stocks intermédiaires 6

1.3.3 Optimiser l'utilisation des moyens 7

1.3.4 Autres objectifs liés à l'environnement de la production 7

1.4 Les différents types de systèmes de production 7

1.4.1 Production unitaire 7

1.4.2 Production en petite et moyenne série 7

1.4.3 Production en grande série 8

1.4.4 Production en continu 8

1.5 Caractéristiques des systèmes automatisés de production 9

1.5.1 Flexibilité 9

1.5.2 Réactivité 9

1.5.3 Pro-activité 9

1.5.4 Robustesse 10

1.5.5 La performance des systèmes de production 10

1.6 Les différents types d'ateliers manufacturiers 10

1.6.1 Machine unique 11

1.6.2 Machines parallèles 11

1.6.3 Ateliers à cheminement unique (Flow-shop) 12

1.6.4 Ateliers à cheminements multiples (Job-shop) 12

1.7 Les problèmes des systèmes de production 13

1.7.1 Problème d'ordonnancement 13

1.7.2 Problème d'optimisation 14

1.8 Ordonnancement 14

1.8.1 Notions de Base 14

1.8.1.1 Notion de tâches 15

1.8.1.2 Notion de ressources 15

1.8.2 Caractéristiques générales des ordonnancements 16

1.8.2.1 Ordonnancement admissible 16

1.8.2.2 Ordonnancement semi actif 16

1.8.2.3 Ordonnancement actif 17

1.8.2.4 Ordonnancement statique/dynamique 17

1.8.3 Méthodes d'ordonnancement 17

1.8.3.1 Simulation 18

1.8.3.2 Résolution de problèmes par agents 18

1.8.4 Paramètres liés à l'étude des problèmes d'ordonnancement 19

1.8.4.1 Les Contraintes 19

1.8.4.2 Qualité de la solution 19

1.9 Pilotage et ordonnancement 20

1.9.1 Notion de Pilotage 20

1.9.2 Pilotage et Ordonnancement 22

1.10 Gestion de Production Distribuée 22

1.11 Conclusion 23

2.1. Introduction.....24

2.2. Bases du Web 24

2.2.1 Architecture client serveur 24

2.2.2. Navigateur web 26

2.2.3. URL 27

2.2.4. HTML 28

2.3. Application web 29

2.4. Technologie de base 30

2.5. Terminologies du Web 31

2.5.1. Http 31

2.5.2. JavaScript 32

2.5.3. ActiveX 32

2.5.4. Un plug-in 32

2.5.5. Applet 32

2.5.6. Servlet 32

2.5.7. Cookie 33

2.5.8. JSP 33

2.5.9. PHP 33

2.5.10. CSS 34

2.5.11. CGI 35

2.5.12. ASP (Active Server Pages) 36

2.6. Les pages web dynamiques 37

2.7. Création 38

2.8. Sécurité 39

2.9. Pourquoi utiliser Java ? 39

2.9.1. Indépendance à l'égard des plates-formes 40

2.9.2. Efficacité 40

2.9.3. Accès aux API Java de l'entreprise 40

2.9.4. Réutilisation 40

2.9.5. Modularité40

2.10. Conclusion 41

III. CHAPITRE 3: ANALYSE ET CONCEPTION DU SYSTEME42

3.1. Introduction 42

3.2 Modélisation 42

3.2.1 Qu'est-ce qu'un modèle ? 42

3.2.1 Pourquoi modéliser ? 42

3.2.1 Qui doit modéliser ? 43

3.3 UML 43

3.3.1 Définition 43

3.3.2 Historique 44

3.3.3 Les diagrammes d'UML 45

3.3.3.1 Les diagrammes structurels 45

3.3.3.2 Les diagrammes de comportement 46

3.4 Schéma de spécification de notre approche 47

3.5. Le modèle proposé.....48

3.5.1 Description des agents.....48 3.5.1.1 Agent administrateur 48

3.5.1.2 Agent Data.....48 3.5.1.3 Agent production.....49 3.5.1.4 Agent commercial.....49 3.5.1.5 Agent designer.....49

3.5.2 Les sorties du modèle.....49 3.5.2.1 Le devis.....49

3.6 Les scénarios possibles dans la gestion de la commande de vente 52

3.6.1 Déroulement des scénarios.....52

3.6.2 les scénarios.....52

3.6.2.1 Le premier cas : commande en ligne.....52 3.6.2.2 Le deuxième cas : commande avec conception à valider .....53 3.6.2.3 Le troisième cas : commande urgente.....53

3.6.3. Le choix du mode de paiement 54

3.6.4. Le Modèle conceptuel des données.....54

3.7 Conclusion 54

IV. CHAPITRE 4: IMPLEMENTATION 55

4.1. Introduction.55

4.2 Les outils utilisés.....55

4.2.1 Netbeans 8.0.2.....55 4.2.2. Oracle.....56

4.2.2.1le serveur Oracle.....57

4.3 Le schéma générale de l'application.....57

4.3.1 La premier partie "Partie Client".....58
4.3.1.1 La procédure d'enregistrement de client dans la plate-forme.....58 4.3.1.2 Comment passer une commande ?.....65 4.3.1.3 Partie de traitement.....68 4.3.1.4 L'annulation de la commande 68

4.4 La deuxième partie du programme "partie administrateur".....68
4.4.1 Agent de production.....70 4.4.2 Agent Designer.....71

4.5 Conclusion.....77

V. CONCLUSION GENERALE 78

Référence bibliographique.....79

Référence webographique.....81

ANNEXE 82

Table des figures:

FIG. 1.1— Les flux informationnels et physiques [Site1]. 6
FIG. 1.2— Exemple de planification de production tirée de [Vacher, 2000]. 9
FIG. 1.3— Modèle d'une Machine unique. 11
FIG. 1.4— Modèle d'une Machine parallèle. 11
FIG. 1.5— Ateliers à cheminement unique (Flow-shop). 12
FIG. 1.6— Ateliers à cheminement multiple (Job-hop). 13
FIG. 1.7— Typologie par les ressources des problèmes d'ordonnancements 15
FIG. 1.8— Ordonnancement admissible [Billaut et Moukrim, 2005]. 16
FIG. 1.9— Ordonnancement semi actif [Billaut et Moukrim, 2005]. 17
FIG. 1.10— Ordonnancement actif [Billaut et Moukrim, 2005]. 17
FIG. 1.11— Modèle de l'environnement d'un système de pilotage. 20
FIG. 1.12— Modèle de référence ISO. 21
FIG. 1.13— Les Fonctions de Pilotage. 21
FIG. 2.1— Architecture client-serveur du WEB [Thierry et al, 2000]. 24
FIG. 2.2— Architecture Générale Client-Serveur du WEB [Thierry et al, 2000]. 25
FIG. 2.3— Principe de CGI 36
FIG. 3.1— Historique de UML 44
FIG. 3.2— Diagramme de séquence 47
FIG. 3.3—Un exemple de devis 51
FIG. 4.1—Logo de Netbeans 55
FIG. 4.2— Logo de ORACLE 12c 56
FIG. 4.3—Le schéma général d'application 57
FIG. 4.4—Représentation générale de l'application partie Client 58
FIG. 4.5— Page Index.jsp 59
FIG. 4.6—Page inscrire.jsp 59
FIG. 4.7— Formulaire d'inscription 60
FIG. 4.8—Page TraitementDeDonnees.jsp 62
FIG. 4.9—Captcha 63
FIG. 4.10— la réponse du système après la consultation de la base de données 64
FIG. 4.11— La réponse du système après l'envoi un E-mail de validation 64
FIG. 4.12—E-mail de validation reçu dans la boit mail du client 65
FIG. 4.13—Authentification du Client

.....65 FIG. 4.14— Formulaire Commande.....67 FIG. 4.15— Message du serveur au client .....68 FIG. 4.16—La liste des commandes .....68 FIG. 4.17—Formulaire d'accès au programme de contrôle.....69 FIG. 4.18—Table de contrôle.....69 FIG. 4.19— Tableau de la configuration de connexion à base de données.....70 FIG. 4.20—Tableau de contrôlé sur les produits.....70 FIG. 4.21—Table Commande.....71 FIG. 4.22— Exemple de consultation de fichier DAO.....71 FIG. 4.23—Agent designer écrit un message d'erreur au client.....72 FIG. 4.24—Message dans la boite E-mail client.....72 FIG. 4.25—Changement d'état de la commande.....73 FIG. 4.26—Génération de devis.....73 FIG. 4.27— Génération de devis en format Excel.....74 FIG. 4.28—Le devis en format Excel.....75 FIG. 4.29—Client peut faire le suivi des commandes.....76 FIG. 4.30—La liste des commandes de client.....76 FIG. 4.31—Devis format Excel.....77

Introduction

La conception d'un produit mécanique est une activité dynamique qui a pour objectif de répondre à des besoins spécifiés par un cahier de charges fonctionnel en termes de solutions technologiques et techniques. C'est une activité aux multiples facettes qui ne se borne pas uniquement à la définition géométrique d'un produit, mais qui interfère également avec les processus associés. Ainsi, la définition d'un produit mécanique nécessite l'intervention de différents métiers (Modélisation géométrique, Analyse, Fabrication et la Gestion de production ...) qui permettront d'aboutir à une description globale et cohérente de celui-ci. Les interactions entre métiers qui étaient jusqu'à présent encore très procédurales et séquentielles, c'est-à-dire, qui se bornaient à suivre une organisation et une planification de tâches relativement figées, tendent à devenir plus coopératives et dynamiques. Le but est d'assurer une meilleure interopérabilité entre ces différents métiers.

À mesure que l'envergure et la complexité des projets augmentent, chacune des étapes et des moyens mis en œuvre deviennent plus élaborés et nécessitent une certaine spécialisation. Ainsi le concepteur ne peut plus communiquer avec un technicien à l'aide d'un dessin fait rapidement à main levée au-delà du stade préliminaire. La réalisation des pièces est telle que beaucoup d'informations doivent être transmises de façon complète et non équivoque. Ceci a donné lieu au dessin industriel, c'est-à-dire la codification de la Communication graphique. De façon semblable, on ne peut plus analyser les contraintes dans une pièce mécanique ou la réponse d'un circuit par une simple formule tirée d'un manuel. On utilise plutôt un calcul numérique par une méthode discrète tels les éléments finis.

La conception assistée par ordinateur (CAO) comprend l'ensemble des logiciels et des techniques de modélisation géométrique permettant de concevoir, de tester virtuellement à l'aide d'un ordinateur des techniques de simulation numérique et de réaliser des produits manufacturés et les outils pour les fabriquer. Le pari des entreprises est que la CAO permettra une hausse semblable de la productivité des ingénieurs. Les progrès dans le domaine de l'électronique mettent, à la disposition de l'ingénieur, une puissance de calcul, de mémoire et du traitement énorme et ceci à bon marché. D'autre part, la création de

logiciels extrêmement évolués permet d'informatiser de nombreuses tâches quantitatives du processus de conception tout en libérant l'esprit pour les tâches de jugement et décisionnelles.

Le but de la Fabrication Assistée par Ordinateur ou FAO est d'écrire le fichier contenant le programme de pilotage d'une machine-outil à commande numérique. Ce fichier va décrire précisément les mouvements que doit exécuter la machine-outil pour réaliser la pièce demandée. Dans le cadre de la Fabrication Assistée par Ordinateur (FAO), les chercheurs ont appliqué la technologie des agents pour exécuter des tâches ; par exemple la production des pièces de contrôle, soit au niveau de l'atelier soit d'une manière distribuée. Nous pouvons citer par exemple, le travail de [Liu et Young, 2004] où une structure à plusieurs agents autonomes a été proposée pour assurer un contrôle intégré dans un atelier de fabrication. Les auteurs dans [Sikora et Shaw, 1998] ont présenté un mécanisme de coordination pour assurer un fonctionnement ordonné et une prise de décision partagée entre les agents. Sans oublier le travail entamé par [Nfaoui et al., 2006] où un système multi-agents (SMA) a été développé pour la conception de produits intégrés dans un réseau informatique orienté environnement CE.

Deux types d'architecture de coordination peuvent être distingués, à savoir les architectures centralisées et distribuées. Par rapport à l'approche distribuée, le système de contrôle centralisé semble être moins souple, plus fiable et plus facile à mettre en œuvre. Cette approche est avantageuse car elle évite de nombreuses interactions entre les agents, qui suppriment la nécessité pour chaque agent de prendre connaissance des autres agents.

La problématique abordée dans ce memoire peut être résumée par l'objectif suivant :

Concevoir une architecture d'une platforme Web permettant la gestion distribuée de la conception des differents modèles et designes. Cette architecture doit intégrer des mécanismes d'identification dynamique pour les clients et les différents agents distribués (Agent Administrateur, Agent Design et le client) et permettre ainsi une gestion de production en temps réel en faisant coopérer et collaborer les différents ateliers de fabrication.

Organisation du mémoire

Ce mémoire est organisé en quatre chapitres qui se présentent comme suit :

Le chapitre 1 : Ce chapitre présente tout d'abord les différentes étapes de la gestion de production pour ensuite aborder le problème d'ordonnancement et prendre une décision concernant une méthode de résolution à appliquer face à un problème d'ordonnancement ; il s'agit d'abord de préciser à quoi correspondent les notions de tâches, ressources,

contraintes et objectifs et ceci avant de terminer une description globale sur la gestion de production distribuée.

Le chapitre 2 : introduit les technologies Web en définissant les principes, les concepts de base, ainsi que les apports et les aspects techniques.

Le chapitre 3 : il faut d'abord organiser ses idées, les classer et les enregistrer, puis organiser la réalisation en définissant les modules et étapes de la réalisation. C'est cette démarche antérieure à l'écriture du code que l'on appelle *modélisation en UML* (*Unified Modeling Language*).Ce chapitre présente en particulier les détails de notre approche en partant de la définition du modèle proposé jusqu'à arriver au Modèle Conceptuel des données (MCD) en passant par les différentes étapes de modélisation de la solution.

Le chapitre 4 : A travers ce chapitre, nous décrivons l'implémentation de l'ensemble des concepts présentés auparavant dans ce mémoire, en exposant les principaux outils utilisés dans la conception de notre prototype, ainsi que les différentes interfaces du logiciel réalisé sur des scénarios d'execution.

La conclusion de ce mémoire nous permet de dresser le bilan des résultats obtenus et d'énumérer quelques perspectives futures.

1.1 Introduction

Depuis toujours, les entreprises ont eu besoin de gérer leurs productions pour imposer leur efficacité. Ainsi le rôle de la gestion de la production est aussi ancien que l'entreprise elle-même. On peut dater les premières réelles expériences en matière de gestion de la production au moment de la realisation des premières pyramides égyptiennes. Ces grands chantiers ont permis les premières réflexions dans le domaine des approvisionnements, des ressources humaines mais aussi de la standardisation des tâches.

D'un point de vue très général, on s'aperçoit vite que pour pouvoir fournir un produit ou un service à un client, l'entreprise doit être capable de mobiliser de nombreuses ressources (moyens de production, moyens de transport…), de nombreux intervenants internes ou externes à l'entreprise, des matières premières, des produits à acheter ou à fabriquer. Il faut mettre en œuvre un savoir-faire, dans un environnement souvent instable où l'on doit maitriser les évolutions des monnaies, les législations, des variations climatiques… et tout ceci avec des contraintes de temps, de qualité et financières.

Pour être capable de produire sereinement un produit ou un service, il faut donc un minimum d'organisation et de gestion. L'objectif de la gestion de production est de gérer cette complexité, et un bon moyen de gérer cette miltetude de variables consiste déjà à simplifier toute la complexité inutile.

La gestion de production est une source considérable de compétitivité. C'est ce qu'ont compris, sans doute avant d'autres, les meilleurs industriels de l'automobile tels que Toyota. Si fabriquer des produits de qualité est une condition nécessaire de compétitivité, ce n'est pas une condition suffisante. Il faut être capable de produire dans des délais très courts une grande variété de produits capables de satisfaire les clients.

Pour être compétitif, il faut fabriquer le juste nécessaire, ne pas ajouter d'opérations inutiles,qui alourdissent le coût de production se focaliser sur ce qui apporte de la valeur ajoutée pour le client.

Et pourtant, dans une entreprise, avec la complexité des flux de produits et des flux d'informations, il se crée chaque jour des opérations, des stockages, un allongement des délais qui nuit à la compétitivité sur lesquels la gestion de production aura une influence considérable [Taghezout , 2011].

1.2 Définitions de la gestion de production

Un problème d'ordonnancement consiste à affecter des tâches aux ressources et à décider de leur répartition dans le temps, de manière à optimiser un critère ou à trouver un compromis entre plusieurs critères [Esquirol et Lopez, 1999].

Une première définition consisterait à dire que c'est un ensemble de processus qui permet de mener à bien la fabrication de produits à partir d'un ensemble de données et de prévisions. En fait, la gestion de production regroupe un ensemble de problèmes liés à la production tels que la gestion des données, la planification et le contrôle (suivi) de la production, la gestion des stocks, la prévision, etc. L'ordonnancement, thème central de cette étude, est donc une fonction liée aux autres fonctions de la gestion de production et à tous les autres aspects inhérents au bon fonctionnement d'un atelier.

Le rôle de la gestion de production est, d'organiser et de piloter le fonctionnement des processus physiques mis en œuvre dans l'entreprise, afin d'assurer une meilleure utilisation des moyens humains, physiques et technologiques disponibles, et de satisfaire au mieux l'objectif global de production défini en terme de quantités à fabriquer avec une qualité demandée, et des délais à respecter [Trentesaux, 1996].

Cette définition présente la gestion de production comme une fonction qui doit tenir compte de nombreuses contraintes et objectifs diversifiés selon leurs natures, dans le but de piloter l'ensemble des moyens de production : contraintes décrites en termes d'utilisation efficace des moyens disponibles (ressources humaines ou matérielles) et objectifs définis en terme global de délai de qualité et de coût.

L'automatisation est le moyen le plus courant qui permet d'obtenir une gestion performante de la production.

Selon [Courtois, 1995] l'objectif principal de la gestion de production est de gérer les flux de matières et d'informations par rapport aux objectifs prioritaires définis par la direction générale de l'entreprise. Le schéma suivant montre l'ensemble des flux que gère, totalement ou partiellement, la gestion de production.

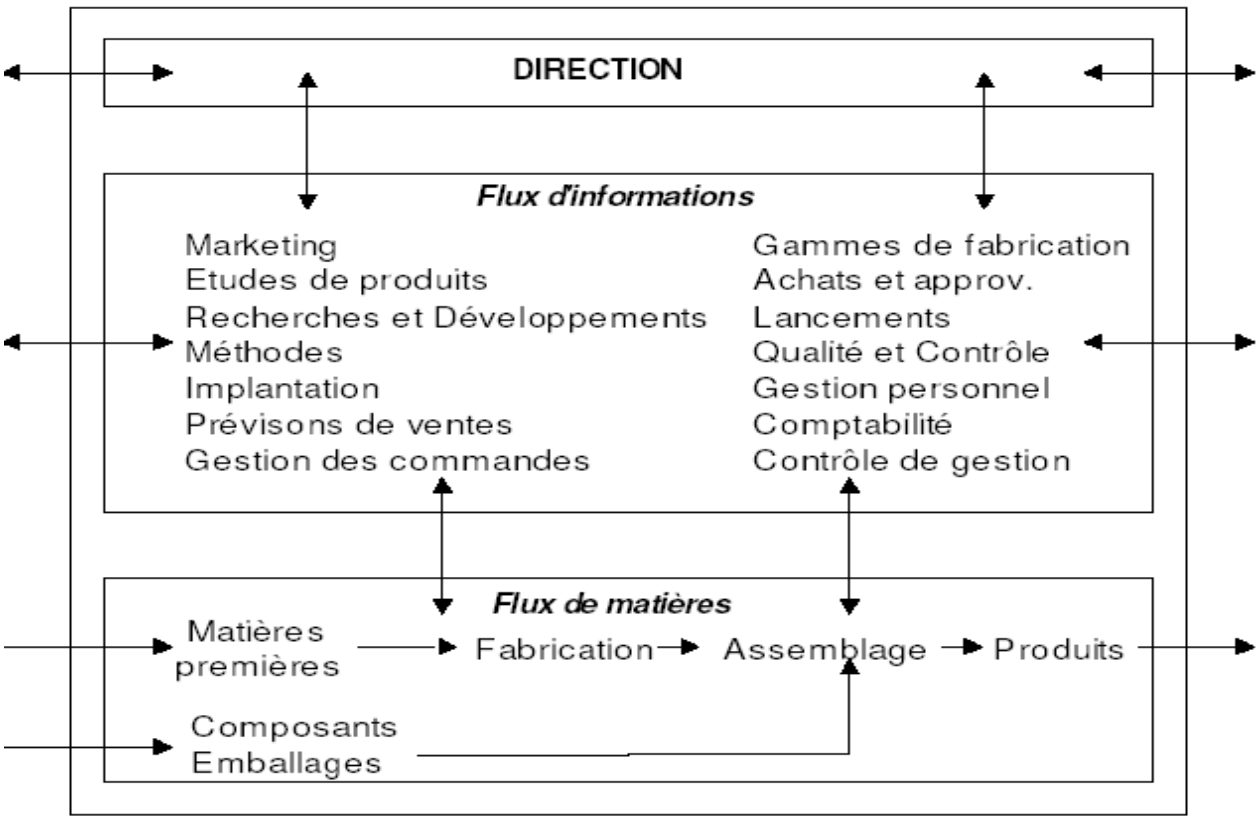


FIG. 1.1— Les flux informationnels et physiques [Site1].

1.3 Objectifs de la gestion de production

Les principaux objectifs d'une gestion de production efficace peuvent être considérés de la manière suivante:

1.3.1 Respecter les délais

Cet objectif est prioritaire vis-à-vis du client, étant donné que les commandes ont un caractère contractuel [Vacher, 2000] et leur non respect peut avoir des conséquences financières pénalisantes, aussi bien sur le plan économique que sur l'image de marque de la société ou de l'entreprisen de production.

1.3.2 Minimiser les stocks intermédiaires

A chaque niveau de fonctionnalité, les différents intervenants ne cernant pas de manière précise les flux et l'ensemble des paramètres entrant dans un processus complexe de fabrication, ne cherchent pas ou ne peuvent pas optimiser les stocks de matières et produits en cours de fabrication ou d'assemblage par exemple. Une tendance naturelle, mais couteuse, est d'accroître les quantités produites intermédiaires pour éviter toute rupture d'enchainement. Une défaillance au niveau de la gestion des stocks peut engendrer une rupture de stocks qui sera fâchense sur le plan économique pour l'entreprise.

1.3.3 Optimiser l'utilisation des moyens

Il est nécessaire et primordial d'optimiser en permanence la charge des moyens chers et détecter la présence de goulots d'étranglement limitant le débit de sortie des produits. Un goulot est un moyen de production dont la capacité est inférieure à la charge résultant d'une demande.

1.3.4 Autres objectifs liés à l'environnement de la production

En plus des objectifs cités précédemment, la gestion de production doit affecter un certain nombre d'objectifs de qualité, de quantité et de coûts. La fonction de production dépend aussi des autres grandes fonctions de l'entreprise que sont les ventes, l'approvisionnement et les stocks.

1.4 Les différents types de systèmes de production

Les entreprises de production sont réparties généralement en quatre catégories, même si dans certains cas un système peut prendre certaines caractéristiques que l'on peut retrouver dans plusieurs classes différentes.

1.4.1 Production unitaire

Ce type de production [Afnor, 1988] également désigné sous l'appellation de projet dans le cas d'une unité importante de construction est celui de la production artisanale et aussi celui de la réalisation de grands ouvrages. Leur point commun est la taille du produit qui oblige à tout organiser autour du chantier où l'unité doit être produite en une seule édition ou un petit nombre à la fois, avec souvent des modifications et des améliorations techniques d'une unité à l'autre.

1.4.2 Production en petite et moyenne série

C'est aussi la production par lots, elle est caractérisée par le fait que le produit est plus ou moins de taille modeste pour que ce produit se déplace de machine en machine dans un même atelier ou un groupe d'ateliers. Le procédé de fabrication d'un produit est, dans ce cas, exprimé par la gamme de fabrication ou routage. La gamme de production d'un article est l'énumération de la succession des actions et autres évènements nécessaires à la réalisation de l'article en question.
On distingue généralement les ateliers de production par lots en deux classes distinctes :

L'usinage et **le montage**. **Les ateliers d'usinage** partent d'une matière première qu'ils façonnent et transforment pour obtenir une pièce ou un produit fini. **Le montage**, à partir d'un certain nombre de pièces (fabriquées à l'usinage ou achetées à l'extérieur), regroupe ces pièces pour réaliser des produits finis complexes conformément aux indications des nomenclatures de décomposition et des gammes de montage.

1.4.3 Production en grande série

Ce type de production est aussi parfois qualifié de production de masse, lorsque le nombre de produits à fabriquer augmente suffisamment il serait intéressant de constituer des installations dédiées. Une chaîne de production représente un ensemble de machines différentes dans l'ordre même dans lequel les opérations doivent être effectuées pour minimiser les temps de transfert et d'attente. Il faut que les machines aient le même rendement et le même débit et que les problèmes des aléas puissent être résolus rapidement avant d'entraîner l'arrêt complet de la chaîne. Toutes ces conditions

font que généralement il est nécessaire d'avoir une surcapacité en ressources humaines et matérielles impliquant des investissements importants.

1.4.4 Production en continu

La production en continu est en fait un cas particulier de la production en grande série. C'est une chaîne parfaitement équilibrée où il n'y a pas de stocks intermédiaires et toutes les machines qui constituent cette chaîne ont le même débit.

Cependant, traditionnellement, on réserve ce terme au cas où les matières et les produits se présentent sous forme liquide (ou du moins pouvant circuler dans des canalisations) ou sous forme gazeuse. Ce type de production concerne par exemple la production en raffinerie, chimie, sidérurgie ou en industrie du verre et du ciment. Il fait souvent intervenir des matières premières. En définitif ce procédé nécessite un approvisionnement continue en matières première.

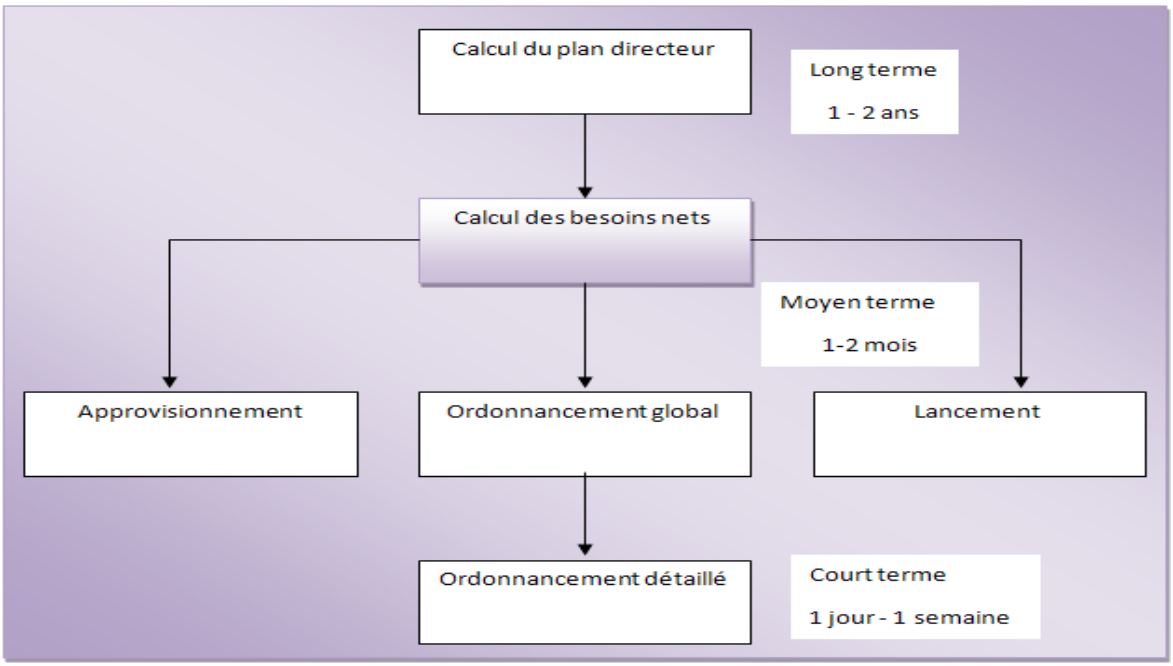


FIG. 1.2— Exemple de planification de production tirée de [Vacher, 2000].

1.5 Caractéristiques des systèmes automatisés de production

1.5.1 Flexibilité

De nos jours la flexibilité des systèmes de production est devenue primordiale à cause de la croissance continue de l'automatisation industrielle. Elle est considérée comme une possibilité pour un décideur de pouvoir à tout moment reconsidérer ses choix de manière à maintenir l'optimalité de sa décision [Dagoumau, 2000].

1.5.2 Réactivité

La réactivité d'un système de production est définie comme l'aptitude à répondre (réagir) dans un temps requis aux changements de son environnement interne ou externe. C'est à dire qu'il doit avoir une conduite qui lui permette de réagir et de s'adapter, en fonction des objectifs de production, aux fluctuations des besoins et aux aléas du système par rapport au régime (fonctionnement) permanent (stable) [Dauzère-Pérès et al., 2005].

Cette réactivité impose une vue dynamique des événements, pour cela trois fonctions s'avèrent nécessaires : Une fonction d'observation, Une fonction de surveillance et Une fonction de correction.

1.5.3 Pro-activité

La pro-activité d'un système de production se caractérise par ses capacités d'anticipation (prévoir et/ou provoquer) des changements d'état, d'apprentissage et d'enrichissement des connaissances, d'adaptation de ses règles de fonctionnement et par sa capacité de réorganisation reposant sur une architecture décentralisée et une délégation de responsabilité. Les fonctions d'observation, de surveillance et de correction sont nécessaires pour assurer la pro activité. Les processus de décision distribués offrent de réelles capacités en termes de réactivité et de pro activité.

1.5.4 Robustesse

La robustesse d'un système de production se définit par son aptitude à produire conformément aux résultats attendus. Cela suppose la garantie de l'obtention des performances souhaitées en présence d'incertitudes dans le système [Billaut, 2005].

1.5.5 La performance des systèmes de production

En raison de la diversité des sens qui peuvent lui être attribués dans ses utilisations courantes, il est bien difficile de caractériser la performance puisqu'elle peut être résultat, meilleur résultat, résultat idéal, ou encore action. La notion de performance est très rarement définie car considérée comme implicitement connue [Innal et Dutuit, 2006].

Les travaux du domaine du contrôle assimilent le plus souvent la performance à un indicateur de performance : temps d'exécution, quantité d'opérations, quantité de ressources utilisées...La plupart du temps, on considère qu'une proposition contribuant à l'amélioration de l'un de ces indicateurs, génère une réduction des coûts et implicitement une amélioration des performances. Certains chercheurs font l'effort de traduire un résultat en terme de coût, mais en utilisant malheureusement des pondérations simplistes issues du contrôle de gestion et admises, même si elles introduisent des distorsions dans l'image qu'elles donnent de la performance économique.

Depuis une vingtaine d'années, certains chercheurs en productique et en génie logiciel s'intéressent à la problématique de la performance industrielle et de l'instrumentation de son évaluation et définissent la performance globale d'un système comme étant l'obtention conjointe de la pertinence, de l'efficience, et de l'efficacité, appréciées en termes de coûts et de valeur, sur l'intégralité du cycle de vie du système [Tahon, 2003].

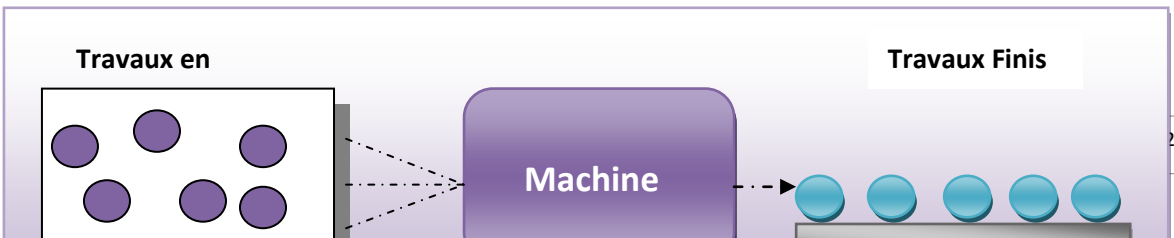
1.6 Les différents types d'ateliers manufacturiers

Une classification très répandue des ateliers, du point de vue ordonnancement, est basée sur les différentes configurations des machines. Les modèles les plus connus sont les suivants :

a. Machines uniques
b. Machines parallèles
c. Ateliers à cheminement unique (Flow-shop)
d. Ateliers à cheminement multiples (Job-shop)
e. Autres configurations (Open-shop et autres).

1.6.1 Machine unique

Dans ce cas, l'ensemble des tâches à réaliser est fait par une seule machine. Les taches sont alors effectuées lors d'une opération. L'une des situations intéressantes où on peut rencontrer ce genre de configuration est le cas où on est devant un Système de Production comprenant une machine goulot qui influence l'ensemble du processus. L'étude peut alors être restreinte à l'étude de cette machine comme cela est montré à la figure 1.3.



1.6.2 Machines parallèles

Dans ce cas, on dispose d'un ensemble de machines identiques pour réaliser les travaux, ces dernières se composent d'une seule opération et un travail exige une seule machine. L'ordonnancement s'effectue en deux phases ; La première phase consiste à effectuer les travaux sur machines et la deuxième phase consiste à établir la séquence de réalisation sur chaque machine (voir FIG 1.4).

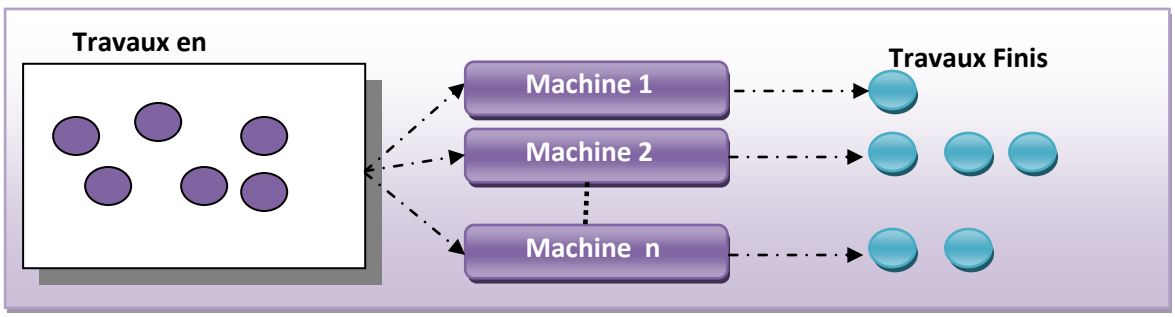


FIG. 1.4— Modèle d'une Machine parallèle.

1.6.3 Ateliers à cheminement unique (Flow-shop)

Un atelier à cheminement unique est un atelier où le processus d'élaboration de produits est dit « linéaire », c'est-à-dire lorsque les étapes de transformation sont identiques pour tous les produits fabriqués. Selon les types de produits élaborés, on distingue deux types de production : la production continue et la production discrète. La production continue est caractérisée par la fluidité de son processus et l'élimination du stockage, c'est le cas notamment dans les raffineries, les cimenteries, les papeteries …etc. La production discrète s'applique principalement aux produits de grande consommation fabriquée à la chaîne comme l'exemple de la fabrication automobile, la majorité du domaine textile, les machines outils…etc. Dans les deux cas, les machines peuvent être dédiées à une opération précise, et sont implantées en fonction de leurs séquences d'intervention dans la gamme de production. L'un des principaux objectifs dans le cas d'atelier à cheminement unique est de trouver une séquence de taches qui respecte un ensemble de contraintes et qui minimise le temps total de production.

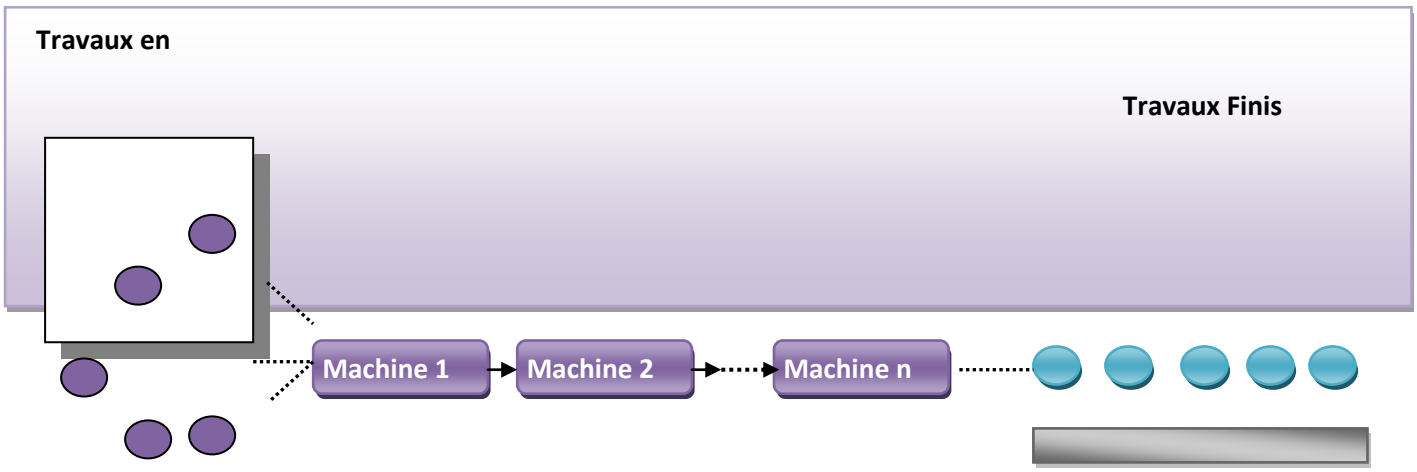


FIG. 1.5— Ateliers à cheminement unique (Flow-shop).

1.6.4 Ateliers à cheminements multiples (Job-shop)

Les ateliers à cheminement multiple (Job-shop) sont des unités manufacturières traitant une variété de produits individuels dont la production requiert divers types de machines dans des séquences variées. L'une des caractéristiques d'un atelier à cheminement multiple est la demande pour un produit particulier est généralement d'un volume petit ou moyen. Une autre caractéristique est la variabilité dans les opérations. Ainsi, il est nécessaire que le système soit de nature flexible. Dans un sens général, la flexibilité est la capacité d'un système à répondre aux variations dans l'environnement (voir figure 1.6).

L'objectif le plus considéré dans le cas d'un atelier à cheminements multiples est le même que celui considéré pour un atelier à cheminement unique, à savoir trouver une séquence de tâches sur les machines qui minimise le temps total de production [Esquirol et Lopez, 1999].

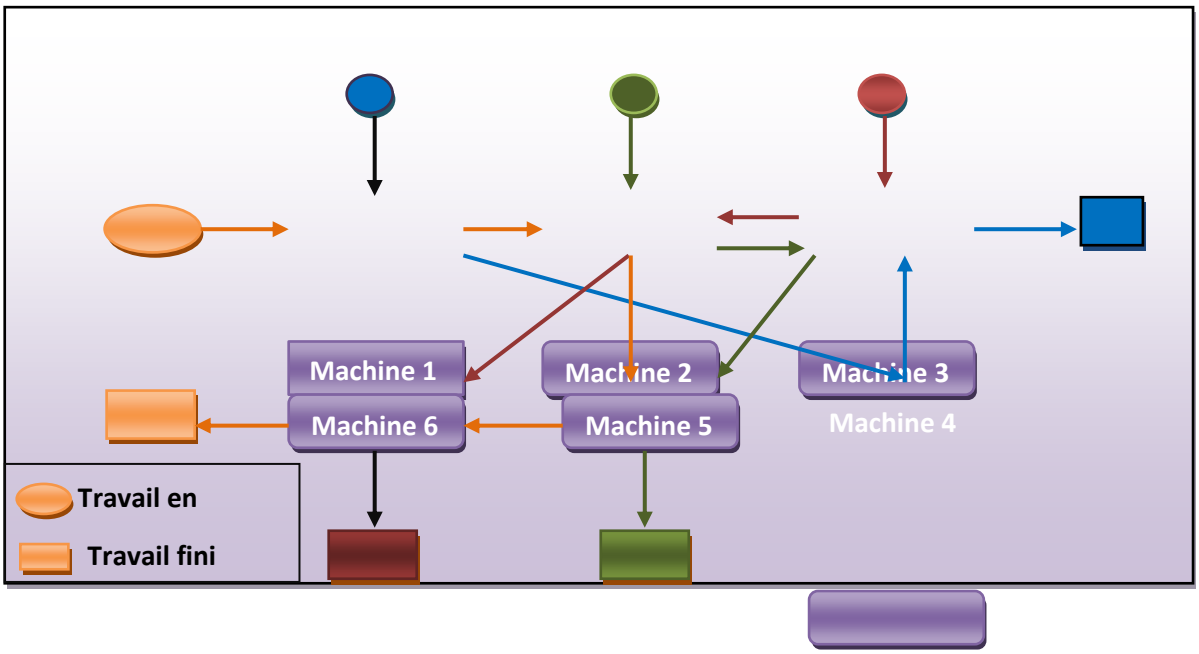


FIG. 1.6— Ateliers à cheminement multiple (Job-hop).

1.7 Les problèmes des systèmes de production

L'objectif des Systèmes de Production (SP) est de satisfaire le client et pour y arriver il est impératif de mieux gérer la production, ce qui implique un ordonnancement pouvant aboutir à des échéances acceptables. De ce fait la complexité des SP se résume en deux principaux problèmes [Vacher, 2000] : problème d'ordonnancement et problème d'optimisation.

1.7.1 Problème d'ordonnancement

L'ordonnancement est une fonction essentielle en système de production. C'est un problème difficile en raison du nombre de calculs à effectuer pour obtenir un ordonnancement qui optimise le critère retenu [Vacher, 2000]. Les problèmes d'ordonnancement consistent à organiser l'exécution de travaux d'un ensemble de ressources disponibles en quantité limitée. En pratique, on rencontre de tels problèmes

dans les systèmes de production manufacturière, ou des ordres de fabrication correspondants à des produits à réaliser doivent être séquencés sur les différentes machines de l'atelier. En particulier, le problème d'ordonnancement de projet à contraintes de ressources, consiste à déterminer les dates de début d'un ensemble de tâches soumises à des contraintes de succession ; chaque tâche nécessite une quantité donnée de ressources impliquées dans le projet pour être exécutée. Il y a aussi des problèmes cumulatifs dans la mesure où à chaque instant et pour chaque ressource la quantité utilisée ne peut dépasser la quantité disponible. Par ailleurs, les décideurs doivent faire face à des perturbations, comme par exemple l'arrivée d'une ou plusieurs tâche(s) imprévue(s) à ordonnancer remettant ainsi en cause la validité de l'ordonnancement.

1.7.2 Problème d'optimisation

Optimiser c'est tout d'abord mesurer avant d'agir sur les points critiques. Pour un système de production il faut mettre en place des indicateurs liés à la productivité en fonction des lots, aux arrêts de machines dont les résultats révèleront les véritables gisements de productivité. Dans la résolution d'un problème d'ordonnancement, on peut choisir entre deux grands types de stratégies, visant respectivement à l'optimalité des solutions, ou plus simplement à leur admissibilité. L'approche par optimisation suppose que les solutions candidates à un problème puissent être ordonnées de manière rationnelle selon un ou plusieurs critères d'évaluation numériques, construits sur la base d'indicateurs de performances [Lopez et Roubellat, 2001]. On cherchera donc à minimiser ou maximiser de tels critères. On note par exemple ceux :

✓ **Liés au temps**

Le temps total d'exécution ou le temps moyen d'achèvement d'un ensemble de tâches, différents retards ou avances par rapport aux dates limites fixées.

✓ **Liés aux ressources**

- La quantité totale ou pondérée de ressources nécessaires pour réaliser un ensemble de tâches.
- La charge de chaque ressource.
- Liés à l'énergie ou un débit.
- Liés aux coûts de lancement, de production, de transport, etc. ; mais aussi aux revenus, aux retours sur les investissements.

1.8 Ordonnancement

Le domaine de l'ordonnancement présente la particularité de se situer à la croisée de plusieurs disciplines scientifiques (mathématiques, économie, productique, automatique, informatique,….). L'ordonnancement est également d'une très forte relation entre

pratique et théorie. Ces caractéristiques se traduisent naturellement par une littérature riche et abondante. Nous présentons tout d'abord ce qu'est un problème d'ordonnancement en évoquant notamment les différents sous-problèmes qu'il recouvre.

1.8.1 Notions de Base

L'ordonnancement est une fonction essentielle en gestion de production, C'est un problème difficile en raison du nombre de calculs à effectuer pour obtenir un ordonnancement qui optimise le ou les critère(s) retenu(s). De plus, il existe de nombreux problèmes d'ordonnancement ou diverses méthodes exactes ou approches de résolution sont proposées pour résoudre une partie d'entre eux [Esquirol et Lopez, 1999]. Parmi les définitions du problème d'ordonnancement, on retrouve l'aspect commun de l'affectation des tâches dans le temps et dans l'espace au moindre cout et en un temps réduit.

Donc il ya problème d'ordonnancement :

- quand un ensemble de travaux est à réaliser,
- que cette réalisation est décomposable en tâches (ou opérations),
- que le problème consiste à définir la localisation temporelle des tâches et, ou la manière de leur affecter les moyens nécessaires.

1.8.1.1 Notion de tâches

Une **tâche** *i* est une entité élémentaire de travail localisée dans le temps par une **date de début** t_i ou de fin $c_{i,}$ dont la réalisation nécessite une **durée** $p_i = c_i - t_{i,et}$ qui consomme des moyens *k*.

1.8.1.2 Notion de ressources

Une **ressource** *k* est un moyen technique ou humain destiné à être utilisé pour la réalisation d'une tâche et disponible en quantité limitée, sa **capacité** A_k.

On distingue plusieurs types de ressources : une ressource est **renouvelable** si, après avoir été utilisée par une ou plusieurs tâches, elle est à nouveau disponible en même quantité. Dans le cas contraire, elle est **consommable.**

Par ailleurs, on distingue aussi dans le cas des ressources renouvelables, les ressources **disjonctives** (ou non partageables) qui ne peuvent exécuter qu'une tâche à la fois, et les ressources cumulatives (ou partageables) qui peuvent être utilisées par plusieurs tâches simultanément (équipe d'ouvriers, poste de travail). Une description est donnée à la FIG 1.7.

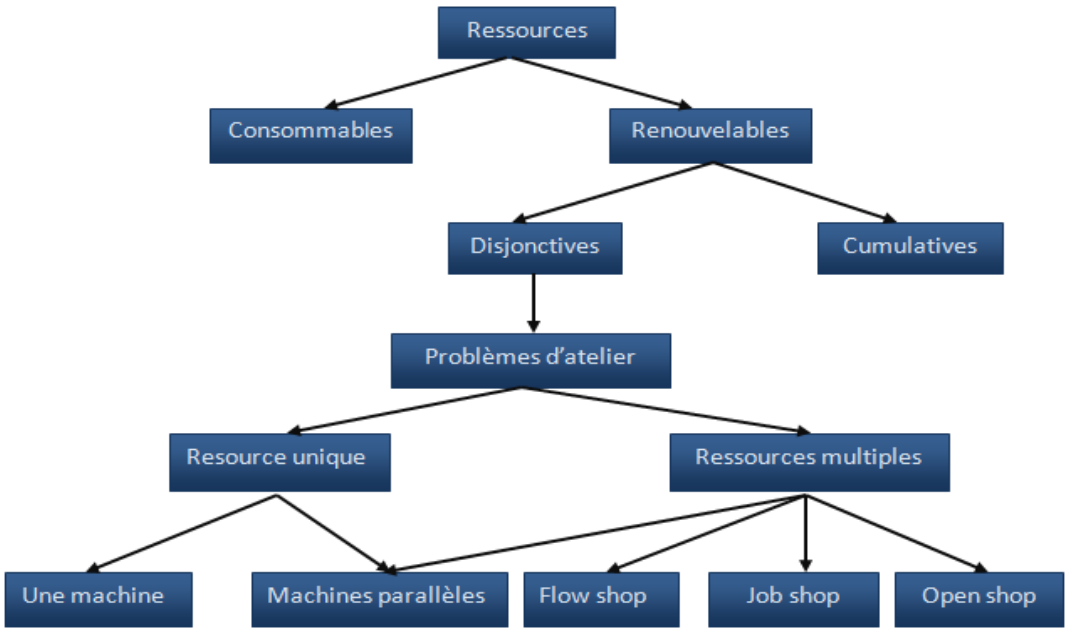


FIG. 1.7— Typologie par les ressources des problèmes d'ordonnancements**.**

1.8.2 Caractéristiques générales des ordonnancements

Des sous-ensembles d'ordonnancement particuliers sont ici présentés dont certains présentent des priorités de dominance vis à vis de tous critères réguliers [Esquirol, 1999].

1.8.2.1 Ordonnancement admissible

Un ordonnancement est dit admissible s'il respecte toutes les contraintes du problème

(Dates limitées, précédences, limitation des ressources…..)

Exemple. Considérons cinq tâches 1,2, 3, 4,5 telles que l'on doive respecter les contraintes de précédence 1 >2 <5 et 3 < 4, Un ordonnancement admissible est représenté à la FIG. 1.8.

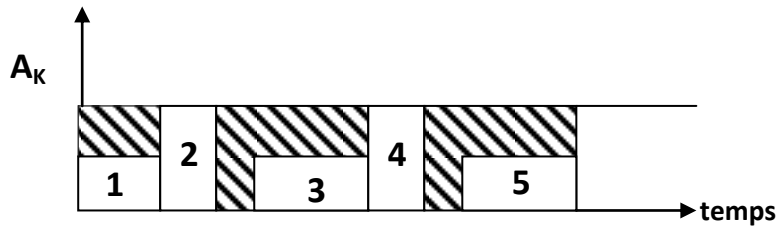


FIG. 1.8— Ordonnancement admissible [Billaut et Moukrim, 2005].

On parle de glissement à gauche local lorsqu'on avance le début d'une tache sans remettre en cause l'ordre relatif entre les tâches. Sur l'exemple ci-dessus bien qu'admissible la solution pourrait être améliorée du point de vue du temps total d'exécution, en faisant un glissement local de certaines tâches vers la gauche (3 et 4 d'une unité, puis 5 de trois unités).

On parle de glissement à gauche global lorsqu'on avance le début d'une tâche en modifiant l'ordre relatif entre au moins deux tâches. Sur l'exemple précédent, le repositionnement de la tâche 5 juste au dessus de la tâche 3 est admissible mais change la relation $4 < 5$ en $5<4$.

1.8.2.2 Ordonnancement semi actif

Dans un ordonnancement **semi actif,** aucun glissement à gauche local n'est possible : on ne peut avancer une tâche sans modifier la séquence sur la ressource. Sur l'ordonnancement admissible précédent, un glissement à gauche local permet par exemple d'obtenir l'ordonnancement semi actif de la FIG 1.9.

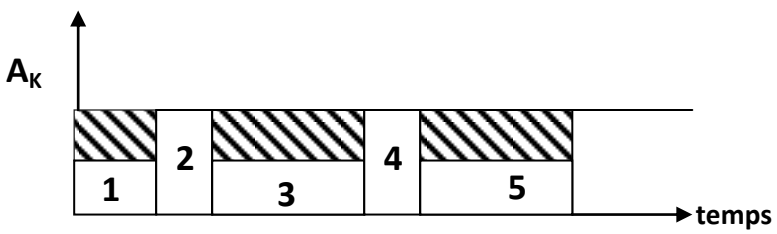


FIG. 1.9— Ordonnancement semi actif [Billaut et Moukrim, 2005].

1.8.2.3 Ordonnancement actif

Dans un ordonnancement **actif** aucun glissement à gauche local ou global n'est possible. Aucune tâche ne peut commencer plus tôt sans reporter le début d'une autre, comme cela est montré à la figure 1.10.

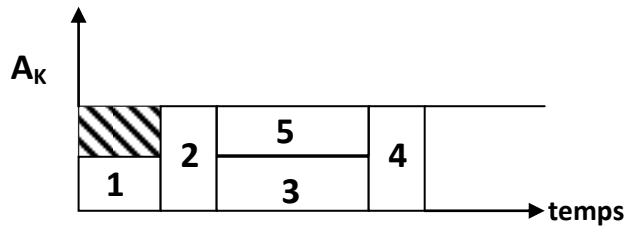


FIG. 1.10 — Ordonnancement actif [Billaut et Moukrim, 2005].

1.8.2.4 Ordonnancement statique/dynamique

Un problème d'ordonnancement est statique si l'ensemble des informations nécessaires à sa résolution est fixé à priori et n'est pas remis en cause durant la résolution (ensemble des tâches, des ressources, et leurs caractéristiques). La solution est alors un plan prévisionnel dont l'exécution nécessite un contrôle d'exécution. Celui-ci peut inclure une fonction de décision en temps réel si le système est doté d'une certaine souplesse.

D'une manière générale, la génération d'un plan et le contrôle de son exécution posent un problème d'ordonnancement dynamique [Esquirol et Lopez, 1999].

1.8.3 Méthodes d'ordonnancement

Parmi les méthodes utilisées en ordonnancement, on peut distinguer les méthodes anciennes comme la méthode PERT, la méthode des potentiels, la simulation ainsi que les méthodes liées à l'intelligence artificielle ou l'emploi d'heuristiques. Cependant, les méthodes PERT ont des potentiels qui ne sont applicables que pour l'ordonnancement d'un ouvrage unitaire ou de production unitaire, ainsi le passage à la petite série leur est fatal.

Elles doivent passer le relais à des méthodes exactes issues des techniques de l'optimisation combinatoire (programmation dynamique, procédure par séparation et évaluation etc.) à des méthodes approchées (heuristiques) ou à des algorithmes génétiques.

Ces méthodes peuvent être classées en diverses familles ; nous donnons brièvement la description de quelques familles.

1.8.3.1 Simulation

L'idée de base de la simulation est de simuler le comportement d'un atelier en faisant avancer le temps et en chargeant les machines libres par un choix sur les opérations disponibles. L'approche de la simulation intègre donc la notion de règles de priorité lors du chargement des postes de travail. Un grand nombre de ces règles ont été citées dans la littérature. Elles ne sont pas toutes efficaces et il n'existe pas vraiment de règle donnant de bons résultats pour tous les types de problèmes et pour un critère donné.

En effet, l'approche de simulation fait partie des méthodes dites à construction progressive d'un ordonnancement. Ce type de méthode permet de déterminer rapidement un ordonnancement respectant les contraintes techniques. Il se caractérise par une grande puissance de modélisation. La qualité de l'ordonnancement dépend des règles de construction de celui-ci, exprimées sous la forme de priorités données aux différents travaux dans les files d'attente.

Règles de priorité : Un très grand nombre de règles ou combinaisons de règles ont été définies et testées. On distingue plusieurs critères de classification :

- Règle locale ou globale : une règle est locale si elle me prend en compte que les informations locales à la file d'attente.
- Statique ou dynamique : une règle est dite statique si la valeur de la priorité reste invariante pendant le séjour du travail dans la file d'attente. Une règle

dynamique peut conserver l'ordre relatif entre deux travaux (SLACK) mais la valeur de la priorité doit être mise à jour dès qu'un nouveau travail arrive dans la file d'attente.

- Informations prises en compte par la règle : on distingue les règles qui prennent en compte les temps opératoires, les dates de livraison, combinent les deux.
- Complexité : combinaison pondérée de règles, utilisation de paramètres à régler.

1.8.3.2 Résolution de problèmes par agents

Nous pouvons considérer que le problème d'ordonnancement est un problème bien posé puisque le traitement fait apparaître des caractères de répartition et de distribution plus ou moins collectives des actions au niveau des machines (ressources).

Pour cette raison, nous pouvons le traiter par une approche par agents. La notion d'organisation d'agents fait référence à la façon dont les agents réalisent en commun leurs actions pour contribuer à l'amélioration d'un objectif global. Les agents ont donc des tâches précisées à priori par une gamme de tâches et ils règlent par eux-mêmes la coordination de ces tâches.

1.8.4 Paramètres liés à l'étude des problèmes d'ordonnancement

1.8.4.1 Les Contraintes

Sur ces problèmes, il existe des contraintes qui peuvent être obligatoires ou sujettes à des relaxations:

- Des contraintes de précédence entre les opérations qui sont en général absolues et fortes,
- Des contraintes physiques qui reproduisent la capacité des diverses ressources de l'atelier,
- Des contraintes de disponibilité des ressources, qui concernent à la fois les ressources consommables (utilisables une seule fois) et les ressources récupérables, indiquant les ressources pouvant être utilisées durant une période de temps donné (comme les machines et la main-d'œuvre).

Parmi les contraintes relaxables (ou de préférences) figurent des contraintes de respect des livraisons, d'encombrement minimal, de lissage de charge, etc**.**

1.8.4.2 Qualité de la solution

Ainsi, on parle d'ordonnancement admissible ou réalisable pour un ordonnancement vérifiant ces contraintes absolues, c'est-à-dire la réalité de l'atelier et de la vie courante.

Cependant, si un ordonnancement est admissible, il faut encore pouvoir juger de sa qualité et s'il respecte les préférences que l'on s'est assignées. On ne peut parler de bon ordonnancement que si on l'évalue par rapport à un critère donné. Donc, il faut donner des objectifs à atteindre, au mieux, car il est toujours difficile d'optimiser une fonction comprenant plusieurs critères. On peut toujours citer parmi les objectifs les plus importants le respect des dates de livraison, la qualité des produits, et la limitation des en-cours de fabrication. On peut aussi envisager de respecter un niveau de stock, maximiser la productivité, répartir au mieux la charge sur les différentes machines, éviter les goulots d'étranglement (sources de gêne pour l'atelier et des problèmes en cas de panne). Il existe d'autres objectifs moins rigides comme la répartition au mieux du travail, les changements compliqués d'outils [Vacher, 2000].

Le coût d'une planification est affecté de manière plus ou moins forte par tous ces objectifs et l'on ne peut les inclure tous dans notre fonction de coût sous peine de rendre difficile, voire impossible, l'obtention d'une solution. Il faut aussi s'efforcer de respecter au mieux les décisions passées, et éviter de trop remettre en cause les ordonnancements. Le difficile compromis entre les différentes préférences impose donc une échelle des objectifs à satisfaire en distinguant ceux qui ont la priorité et ceux que l'on ne peut satisfaire qu'à moitié.

1.9 Pilotage et ordonnancement

1.9.1 Notion de Pilotage

A l'heure actuelle, il n'existe pas de définition unique ni stabilisée du terme pilotage. Une définition a été proposée dans les travaux de Trentesaux en 2000, dans laquelle il définit le pilotage comme suit :

Le pilotage consiste à décider dynamiquement des consignes pertinentes à donner à un système soumis à perturbation pour atteindre un objectif donné décrit en termes de maîtrise de performances [Trentesaux, 2002]. Le schéma proposé à la figure 1.11 place le pilotage dans son environnement. Ici, le concept de pilotage concerne la définition et l'organisation des interrelations entre deux sous-systèmes : un sous-système physique (appelé système opérant) et sous-système de décision (appelé système de pilotage), conduisant à la mise en œuvre d'une boucle de rétroaction.

Les informations en entrée du système de pilotage sont les objectifs (résultats souhaités du système opérant), les autres données et contraintes externes (issues de l'environnement) ainsi que les observations effectuées sur le système opérant (états internes et mesure des effets). Les informations en sortie sont les signaux de commande, les résultats du système opérant ainsi que le sous-ensemble des indicateurs de performance requis par l'environnement.

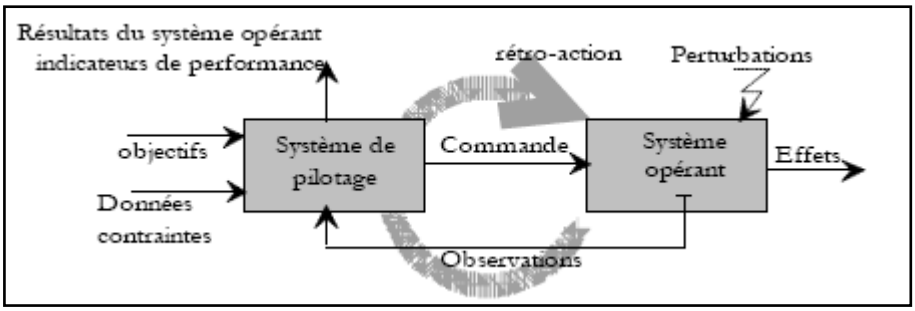


FIG. 1.11— Modèle de l'environnement d'un système de pilotage.

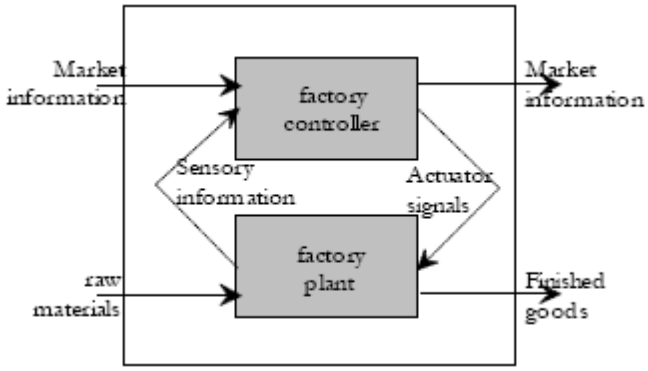


FIG. 1.12 — Modèle de référence ISO.

D'une manière générale, piloter un système de production fait appel à deux fonctions distinctes :

- Une fonction de gestion *à priori* (ou planification) dont le rôle est d'assurer la programmation d'un ensemble d'actions ou de décisions.
- Une fonction de gestion en temps réel (ou conduite) dont le rôle est de prendre les décisions qui relèvent de l'immédiat, c'est-à-dire qui sont motivées par les événements liés à l'état courant du système de production.

La fonction pilotage temps réel peut être décomposée en deux sous fonctions : **conduite** et **commande** (voir FIG 1.13).

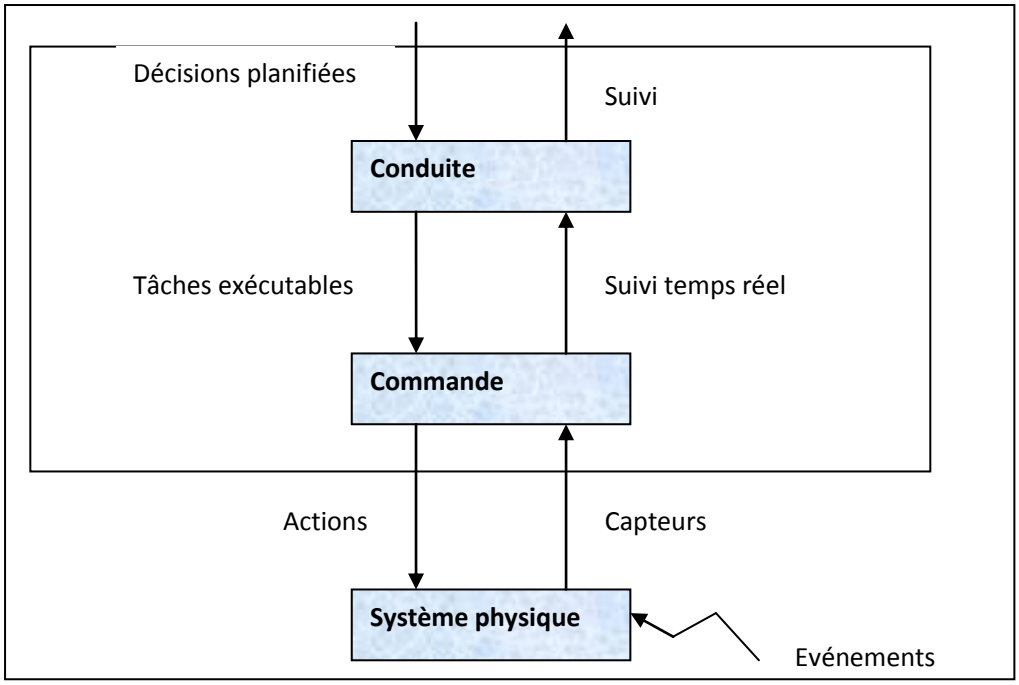


FIG. 1.13— Les Fonctions de Pilotage.

1.9.2 Pilotage et Ordonnancement

Parunak [Parunak,1991] définit l'ordonnancement prévisionnel comme étant le : « sous ensemble du produit cartésien Quoi*Où*Quand pour un sous ensemble de taches à réaliser ». On peut différencier les types de pilotage selon qu'un ordonnancement prévisionnel des taches est réalisé ou non :

- **Pilotage sans ordonnancement prévisionnel** : Les allocations se font en temps réel et de manière dynamique au fur et à mesure de l'évolution du système de production.
- **Pilotage à ordonnancement prévisionnel partiel** : Seule une partie des tâches à réaliser sont planifiées à priori. Les autres sont allouées dynamiquement.
- **Pilotage à ordonnancement prévisionnel total *:*** Toutes les tâches à réaliser sont allouées en amont du système de pilotage.

1.10 Gestion de Production Distribuée

De manière à demeurer compétitives, les entreprises externalisent les activités qui ne sont pas au cœur de leur métier, ou celles pour lesquelles elles ne possèdent pas d'avantages concurrentiels. Ceci les conduit en particulier à délocaliser la plupart de leurs activités à faible valeur ajoutée. Cette externalisation des activités se manifeste sous la forme de production distribuée, permettant à l'entreprise de se concentrer sur son propre savoir-faire et de partager les autres activités avec d'autres entreprises. Par conséquent, plusieurs réseaux d'entreprises regroupant les entreprises partenaires sont

créés avec des durées de collaboration plus ou moins courtes. Ce regroupement d'entreprises en réseau nécessite une collaboration et une coopération entre les différents partenaires pour gérer la réalisation des projets multi-sites.

Afin de gérer cette production distribuée, les entreprises se basent le plus souvent sur des approches traditionnelles qui exploitent des modèles de production plus ou moins centralisés. Des approches alternatives basées sur les concepts d'agents ont été proposées pour prendre en compte le contexte distribué de la production. L'avènement d'Internet et des nouvelles technologies de l'information et de la communication permet aujourd'hui aux entreprises de créer des places de marchés électroniques pour améliorer la gestion de la production distibuée et ainsi leur compétitivité.

Pour mieux gérer la réalisation des projets au sein de ces différentes organisations, les entreprises utilisent des systèmes et des applications de production basés sur des approches plus ou moins centralisées.

1.11 Conclusion

Quel que soit son secteur d'activité (mécanique, plastique, alimentaire, bois...), l'entreprise a besoin d'une gestion de production résolument moderne et efficace qui se traduit par la mise en œuvre de nouveaux principes de gestion de production, par l'implication, la formation des acteurs de l'entreprise et par la mise en œuvre de technologies.

La résolution d'un problème d'ordonnancement commence par la modélisation du système de fabrication et de son fonctionnement. Il faut donc définir un ensemble de paramètres représentant le système étudié, les liens existant entre ces paramètres et les critères qui permettront d'évaluer les solutions obtenues. Il est clair qu'il existe diverses manières de modéliser chaque problème d'ordonnancement, alors on est amené à choisir un mode de description. Parmi ces modes, on peut citer les équations mathématiques, les graphes potentiels-tâches, les réseaux de Pétri, etc.

Dans le prochain chapitre, nous allons présenter les technologies web.

2.1. Introduction

Aujourd'hui, nous sommes entrés dans une nouvelle ère où la plupart des utilisateurs sont connectés à un réseau (local ou global). L'échelle des ressources à représenter a fortement

grandi l'influence du Web permet aux utilisateurs d'exiger plus de simplicité. Le volume d'informations accessibles est aujourd'hui quasi illimité, il devient indispensable d'en faciliter l'accès. Cependant, il est nécessaire de remettre en cause nos réflexes vis-à-vis de l'ergonomie des interfaces graphiques.
Dans ce chapitre, nous allons présenter les bases du WEB, les technologies WEB et les motivations à choisir le langage JAVA pour le développement de notre application.

2.2. Bases du Web

Le Web fait appel à un certain nombre de concepts de base. Nous verrons les trois plus importants dans cette première partie.

2.2.1 Architecture client serveur

Le World Wide Web s'appuie sur la notion d'architecture client serveur. Un serveur est une machine en général assez puissante qui fournit un ou plusieurs services (accès à des sources de données, applications…). Pour fournir ces services elle fait tourner en permanence des programmes que l'on appelle aussi des serveurs, en l'occurrence ce sont des serveurs Web ou serveurs HTTP. De l'autre coté les utilisateurs font tourner sur leur machine (machine cliente) un programme client qui, comme son nom l'indique va être demandeur de services, en l'occurrence ce client est un navigateur Web qui va demander des pages Web à un serveur Web. Le dialogue entre le client et le serveur se compose donc de requêtes émises par le client et de réponses données par le serveur comme il est montré dans la FIG. 2.1.

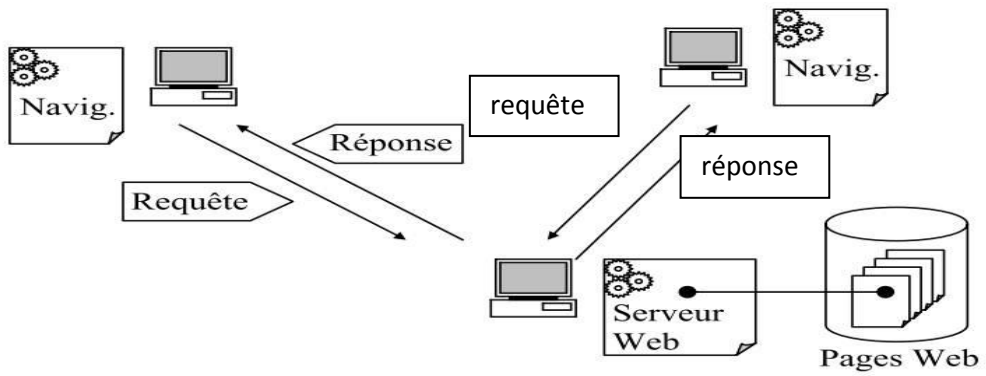


FIG.2.1–Architecture client-serveur du WEB [Thierry et al, 2000].

Un serveur est une machine qui est capable de 'servir' d'autres machines en fonction de leur requête, ces dernières sont appelées 'clients'. Pour cela elle doit toujours être connectée au réseau et exécuter le démon (daemon) correspondant au service rendu. On appelle démon un programme qui tourne en tâche de fond sur une machine et le cas échéant répond à des requêtes qui lui sont adressées ou déclenche des actions en réponse à des événements ou un planning. Sur le Web les documents s'échangent selon le protocole HTTP (HyperText Transfer Protocol) et le démon qui se charge de répondre aux requêtes des autres machines se nomme HTTPD (HyperText Transfer Protocol Daemon). De la même façon un serveur offrant des fichiers via FTP est une machine sur laquelle tourne un

serveur FTP encore appelé démon FTPD (File Transfer Protocol Daemon), de même pour l'e-mail, etc...

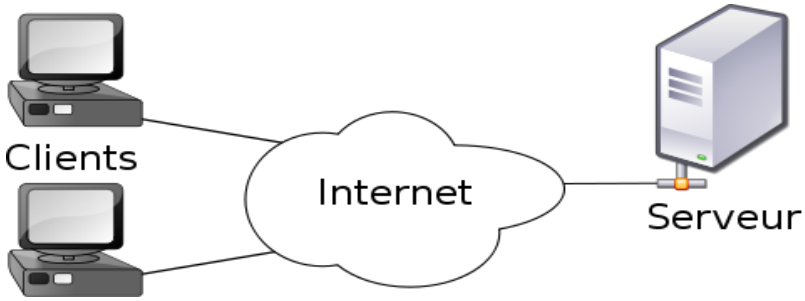


FIG.2.2–Architecture Générale Client-Serveur du WEB [Thierry et al, 2000].

Il y a deux cas : soit l'utilisateur cherche à visualiser une page disponible sur sa machine auquel cas le navigateur obtient le fichier par simple lecture directe sur un disque de la machine sur laquelle il s'exécute, soit l'utilisateur souhaite accéder à une page disponible sur une machine distante auquel cas le navigateur doit se connecter au serveur publiant cette page à travers le réseau. On se rappelle qu'internet est l'infrastructure internet (interconnected networks = réseaux interconnectés) d'un réseau informatique mondial. Ce réseau mondial se compose de réseaux d'ordinateurs locaux interconnectés et dont les échanges suivent les protocoles TCP, UDP et IP (Transmission Control Protocol and Internet Protocol) chaque ordinateur connecté étant adressé par un numéro IP ou un nom symbolique / nom de domaine. L'architecture client-serveur du Web repose sur ces bases en ce sens que le programme client (navigateur) se connecte au programme serveur (serveur Web) grâce aux protocoles TCP/IP et ainsi met en place une connexion bidirectionnelle fiable qu'il va utiliser pour obtenir les informations sollicitées (document, image et autres fichiers).

2.2.2. Navigateur web

Le navigateur web est le logiciel qui affiche les résultats envoyés par le serveur, reçoit les manipulations de l'utilisateur et les transmet au serveur.

Le client est couramment appelé un navigateur (ou encore browser, fureteur ou butineur). Les navigateurs les plus connus étant Netscape, Internet Explorer, Google Chrome, Mozilla Firefox, Opera. Les plus courant acceptent des extensions (Plug-In) permettant d'étendre leurs capacités (lecture des vidéo, réception du son ou des films en flot continu, …). Ils connaissent aussi fréquemment des langages évolués (Java, Javascript, Css…) permettant d'élargir le champ des possibilités de l'utilisation des pages Web. Les navigateurs sont des logiciels soigneusement étudiés pour faciliter et assister la navigation sur le Web. Ils proposent en standard des fonctionnalités d'historique pour revenir sur ses

pas, une gestion des signets pour pouvoir garder ses pages préférées bien organisées et facilement accessibles, ainsi que plusieurs autres fonctions utilitaires et outils d'assistance.

Une applet est un morceau de logiciel incorporé dans une page web, et exécuté par le navigateur web de l'ordinateur client. Lorsque l'utilisateur actionne un widget placé dans une page web, l'applet peut alors modifier la présentation de la page (technique appelée DHTML), afficher des messages ou envoyer des requêtes au serveur d'application.

JavaScript, initialement appelé *LiveScript* est un langage de programmation pour les applets, il a été développé par Netscape. Les applets écrites dans ce langage sont exécutées par un interprète inclus dans le navigateur web. Ainsi de nombreux navigateurs web ont un interprète Javascript.

Les premiers navigateurs web équipés d'un interprète Javascript présentaient des différences de dialecte qui font qu'une applet écrite pour un interprète en particulier n'est pas toujours comprise par un autre interprète. Pour pallier ce problème, Ecma International publie en 1999 la norme industrielle ECMA-262 ECMAScript qui spécifie la syntaxe que doivent comprendre les interprètes Javascript.

Le Document Object Model (abrégé *DOM*) est un ensemble d'objets normalisé qui représente la page affichée, ainsi que le navigateur web. Le DOM est l'interface de la programmation utilisée dans les applets pour effectuer des modifications sur la page.

La méthode Ajax consiste à utiliser de manière conjointe des technologies telles que JavaScript, CSS, XML, le DOM et le XMLHttpRequest dans le but de réaliser des applications Web qui offrent une maniabilité et un confort d'utilisation supérieur à ce qui se faisait jusqu'alors - les Rich Internet Application (abr. RIA).

Java est un langage de programmation développé par Sun Microsystems, qui peut être utilisé pour les applets. Les applets écrites dans ce langage sont préalablement compilées, et exécutées par un logiciel branché au navigateur web, le *plug-in* Java (traduction littérale : qui se branche dessus).

ActiveX est une technologie développée par Microsoft ou des composants logiciels peuvent être inclus en tant qu'applet dans des pages web. Cette technologie nécessite un système d'exploitation Windows sur l'ordinateur client.

Flash est une technologie développée par Adobe. Un logiciel branché au navigateur, le *plug-in Flash* permet d'afficher des animations, des images vectorielles, des vidéos, et exécuter des applets. Il comporte un interprète pour des applets en langage de programmation ActionScript, un langage similaire à Javascript. Cette technologie est d'usage courant pour les jeux vidéo en ligne [Site4].

2.2.3. URL

Pour accéder à une page Web il faut d'abord pouvoir décrire où elle se trouve. Pour repérer un document, un fichier, une source de données …on a développé la notation URL (Universal/Uniform Resource Locator). Un URL peut désigner un serveur ftp, un fichier

sur votre disque, un serveur gopher, une image, une adresse courrier, un serveur de News, un serveur telnet et bien sûr une page Web publiée par un serveur http, c'est-à-dire un serveur de Web. En particulier, dans ce dernier cas l'URL contient le nom du protocole d'accès au fichier (HTTP, SHTTP), le nom du serveur (adresse IP ou nom symbolique), le chemin d'accès au fichier et bien sûr le nom du fichier :

Les noms d'URL utilisent les lettres de l'alphabet en général en minuscule, les chiffres sont autorisés, certains caractères / . : # ont une signification particulière et sont donc réservés, enfin certains caractères sont dit non sûrs dans la mesure où ils sont interprétés ou interprétables différemment : les blancs, les étoiles, etc.

Les trois caractères **/ . :** sont des séparateurs simples, le **?** Est un séparateur introduisant une requête qui en général demande au serveur d'exécuter un programme (CGI,PHP,JSP…) pour générer la réponse. Exemple typique est celui des moteurs de recherche où ce type d'URL est utilisé pour envoyer des mots clefs à un programme qui génère la page des réponses. Exemple: si vous lancez une requête sur le mot clef 'chat' sous Google.fr l'URL est :

https://www.google.fr/#q=chat

Cela signifie que votre requête appelle un programme de recherche dans l'annuaire de Google.fr avec le paramètre 'chat'[site 5].

2.2.4. HTML

Le langage HTML (HyperText Markup Language) est utilisé sur le système de partage de l'information mondial WWW (World Wide Web) depuis 1990. Ce langage se compose d'un ensemble d'annotations, appelées étiquettes ou balises, qui permettent de créer et formater un document hypertexte. Un fichier HTML est un fichier texte ce qui a l'avantage de le rendre facilement lisible sur n'importe quelle plate-forme/ordinateur. Les balises du HTML sont insérées dans le texte du document et guident son affichage. Le navigateur interprète les commandes HTML contenues dans le document et en déduit le format d'affichage du document.

HTML est le langage standard d'édition de pages hypertexte pour le Web. Il existe plusieurs versions les plus communes et les plus supportées sont les versions 2.0 et 3.2. La toute dernière étant la version 5. Une page Web peut être crée directement avec un simple éditeur de texte en tapant des commandes HTML ou en utilisant un éditeur de page Web qui très souvent nous permet de créer notre document de façon très conviviale et générera pour nous le code HTML correspondant sans que nous ayons à connaître ce langage.

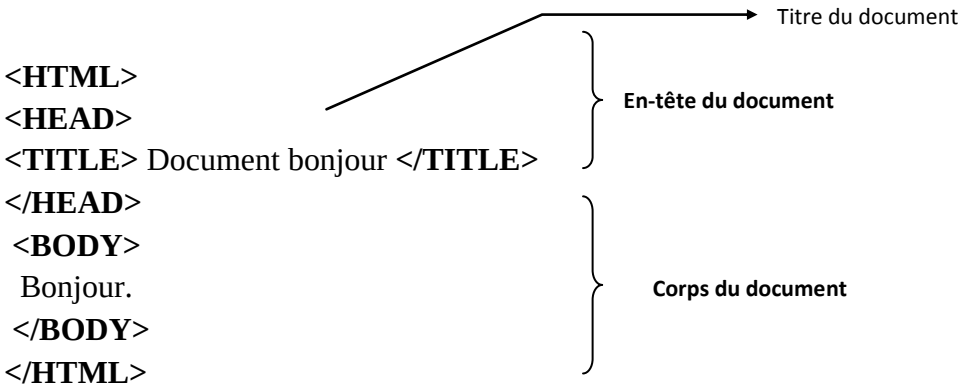
Le HTML n'est pas un langage de programmation, c'est un langage d'édition de documents. Une balise est un mot clé, une commande du langage insérée dans le corps du document pour introduire un effet particulier (début de mise en gras, fin de mise en gras, début de tableau…). Une balise commence toujours par un signe "<" et se termine toujours par un signe ">". La plupart des balises doivent être ouvertes et fermées pour délimiter

leur zone d'influence. La balise fermante contient la même commande que la balise ouvrante, mais précédée d'un caractère /

Par exemple pour mettre un texte en gras la commande est **B** (comme "Bold" en anglais qui veut dire "en gras") la balise ouvrante est **<B>** et la balise fermante est **</B>**. Donc si dans la phrase "Je suis étudiant à l'Université." je veux mettre le mot 'étudiant' en gras comme ceci "Je suis **étudiant** à l'Université.", le code HTML correspondant sera :

Je suis <B>étudiant</B> à l'Université.

Les balises peuvent utiliser des attributs, pour paramétrer leur effet. Comme dans tous les langages il y a un minimum d'informations à donner dans le fichier HTML. Le fichier HTML minimum ressemble à ceci :




```
<HTML>
<HEAD>
<TITLE> Document bonjour </TITLE>
</HEAD>
 <BODY>
  Bonjour.
 </BODY>
</HTML>
```

Les balises <HTML> et </HTML> stipulent que ce fichier texte est formaté selon le langage HTML et délimitent le contenu à interpréter.

Les balises <HEAD> et </HEAD> viennent du mot HEADER (Entête) et délimitent l'en-tête du document contenant son titre et des informations sur son contenu.

Les balises <BODY> et </BODY> délimitent le corps du document contenant le texte, son formatage les objets et les liens qu'il inclut.

On peut taper ce petit exemple dans un éditeur de texte, le sauver (par habitude, les fichiers HTML ont pour extension .html ou .htm exemple : bonjour.html ou bonjour.htm) et l'ouvrir grâce à l'option 'ouvrir un fichier' du menu fichier de notre navigateur.

Comme nous le disions au début, pour que l'édition de grands documents ne soit pas fastidieuse, on peut utiliser un éditeur HTML cependant il faut savoir que toutes les subtilités du langage ne sont pas forcément disponibles au travers d'un tel logiciel et que le

code généré n'est pas toujours de bonne qualité. Enfin insistons sur le fait que le HTML n'est pas un langage de programmation en lui même et que de plus il n'y a pas de compilation car tout se fait sur le principe de l'interprétation du document tel qu'il est décrit dans le fichier texte. L'aspect programmation n'apparaît qu'au travers de l'utilisation du Java, du JavaScript, des CGI, PHP,… et ces domaines sont à eux seuls très complexes et les seront abordés de façon abrégée [Site6].

2.3. Application Web

En informatique, une application web (aussi appelée web app, de l'anglais) est une application manipulable grâce à un navigateur web. De la même manière que les sites web, une application web est généralement placée sur un serveur et se manipule en actionnant des *widgets* à l'aide d'un navigateur web, via un réseau informatique (Internet, intranet, réseau local, etc.). Les messageries web, les systèmes de gestion de contenu, les wikis et les blogs sont des applications web.

Les moteurs de recherche, les logiciels de commerce électronique, les jeux en ligne, les logiciels de forum peuvent être sous forme d'application web. Des appareils réseau tels que les routeurs sont parfois équipés d'une application web dans leur micro-logiciel. Les applications web font partie de l'évolution des usages et de la technologie du Web appelée Web 2.0.

2.4. Technologie de base

Dans la technologie client-serveur, utilisée pour le World Wide Web, le navigateur Web envoie au serveur des requêtes relatives à des pages Web. Le serveur répond aux demandes en envoyant les pages au navigateur Web. Le navigateur affiche alors les pages à l'utilisateur.

Les applications Web utilisent cette technique pour mettre en œuvre leur interface graphique. Celle-ci est composée de pages créées de toutes pièces par le logiciel lors de chaque requête. Chaque hyperlien contenu dans la page provoque l'envoi d'une nouvelle requête, qui donnera en résultat une nouvelle page. À la différence d'un site web *statique* où les pages sont des fichiers préalablement enregistrés.

Les pages Web contiennent divers *widgets* tels des boutons poussoirs, des icônes et des zones de texte, permettant la manipulation de l'application. Chaque manipulation d'un bouton poussoir provoque l'envoi d'une nouvelle requête.

Contrairement à d'autres logiciels, une application Web mise en place sur un serveur est immédiatement utilisable par le consommateur sans procédure d'achat et d'installation sur son propre ordinateur, du moment que l'ordinateur du consommateur est équipé d'un navigateur Web et d'une connexion réseau. Ceci évite des interventions des administrateurs système, interventions qui sont souvent plus coûteuses que le logiciel lui-même. L'application Web est souvent mise à disposition du consommateur par l'éditeur du logiciel sur ses propres serveurs - technique appelée *Software as a Service*.

L'usage du navigateur Web comme partie client - un logiciel qui est disponible sur de nombreux systèmes d'exploitation - assure la portabilité d'une application Web.

Dans la technologie la plus courante, l'application web s'oriente autour d'un serveur web sur lequel est branché le logiciel applicatif, le tout parfois accompagné d'un serveur de base de données. L'ensemble est appelé serveur d'applications.

Le code source du logiciel applicatif est placé directement dans des pages web. Ces pages sont stockées par le serveur. Lorsque le client demande une page, le serveur web va rechercher la page, puis exécute les instructions qu'elle contient. Ces instructions peuvent faire appel au serveur de base de données. Le serveur web transmet la page avec le résultat de l'exécution au client.

Les applications web font souvent usage du mécanisme des cookies : en réponse à une requête, le serveur envoie une information de repérage au client (le *cookie*). Puis le client lui renvoie cette information lors de la prochaine requête. Le mécanisme est utilisé pour identifier le client et suivre les manipulations.

Les pages web peuvent en outre contenir des applets. Ce sont des morceaux de code source qui sera exécuté par le navigateur web après transmission de la page - contrairement à la majorité du code source qui est exécuté par le serveur web *avant* la transmission. ActiveX, Java et Adobe Flash sont des technologies utilisées pour les applets [Site7].

2.5. Terminologies du Web

En 1991, le National Center for Super computing Applications publie la norme industrielle CGI, qui spécifie par quel biais un logiciel de serveur web peut être branché à un logiciel applicatif. Cette norme est initialement destinée à permettre la réalisation de moteurs de recherche.

La transmission des informations entre le client et le serveur se fait selon le protocole HTTP, protocole également utilisé pour les sites web. Ce qui permet d'utiliser le même logiciel client - un navigateur web [Site7].

2.5.1. Http

Le protocole de base du World Wide Web est HTTP (HyperText Transfer Protocol) qui peut être utilisé pour n'importe quelle application client-serveur impliquant de l'hypertexte. Ce protocole est capable d'assurer le transfert de texte, hypertexte, fichiers audio, images ou tout autre type d'information pouvant se mettre sous la forme d'un fichier.

Le scénario de dialogue classique entre un navigateur et un serveur Web est le suivant. Le navigateur Web client établit une connexion TCP avec le serveur Web qui contient la page qui l'intéresse. Une fois la connexion établie, le client émet une requête HTTP contenant

une commande, une URL, et parfois d'autres informations. Lorsque le serveur Web reçoit la requête il essaie d'exécuter la commande qu'elle contient. Il retourne ensuite comme réponse le résultat obtenu qui peut être des données, un message d'erreur, et d'autres informations. Une fois que le client a reçu sa réponse la connexion est fermée et détruite [Site7].

2.5.2. JavaScript

JavaScript est un langage léger, mais relativement complexe et puissant, qui apporte des fonctions dynamiques à HTML dans les navigateurs. Malgré son nom, il est très différent du langage Java et non vraiment orienté objet (pas d'héritage…) [Anne, 2002].

2.5.3. ActiveX

Ce sont des composants logiciels inclus dans une page HTML et téléchargés par le navigateur en même temps que celle-ci. Ils sont dérivés du modèle de composant COM (Component Object Model) popularisé avec Visual Basic. Ce modèle permet d'ajouter des fonctionnalités à Windows en enregistrant des composants compilés dans la base des registres. Ces composants deviennent alors disponibles et exploitables par toutes les applications Windows.

2.5.4. Un plug-in

Est un programme téléchargé et installé automatiquement par le navigateur lorsque son usage est requis pour la première fois. Le principe est relativement simple : Lorsque le navigateur est confronté à un type de données inconnu, il fait appel au plug-in concerné pour le traiter. Les plug-in les plus utilisés à l'heure actuelle sont ceux qui permettent de profiter de fonctionnalités multimédia ou de mises en page non supportées par HTML. Ils sont généralement gratuits et téléchargeables depuis le site Web de leurs auteurs [Anne, 2002].

2.5.5. Applet

Une applet est un programme Java dont le code est téléchargé sur le poste client depuis le serveur web et s'exécute ensuite dans le navigateur en utilisant un interpréteur Java (la machine virtuelle, ou JVM, qui est intégrée dans la plupart des navigateurs). Le modèle de sécurité des applets est très strict mais évolutif depuis Java 2 (utilisation des certificats) [Anne, 2002].

2.5.6. Servlet

Le programme Java s'exécute dynamiquement sur le serveur Web et permet l'extension des fonctions de ce dernier, typiquement : accès à des bases de données, transactions d'e-commerce, etc. Un servlet peut être chargé automatiquement lors du

démarrage du serveur Web ou lors de la première requête du client. Une fois chargés, les servlets restent actifs dans l'attente d'autres requêtes du client [Marty et Larry ,2000].

2.5.7. Cookie

C'est un petit ensemble d'informations qu'un serveur peut demander à un client de garder, pour lui demander de les restituer par la suite. Une application web peut par exemple utiliser un cookie transitoire pour que le serveur suive la trace d'un navigateur client particulier, tout au long de son parcours parmi les pages du site web. Quant aux cookies permanents, ils servent souvent de tickets d'entrée virtuels évitant au client de ressaisir ses informations d'accès personnelles.

2.5.8. JSP

C'est une extension de la technologie Java Servlet de Sun qui permet de programmer simplement l'affichage de contenus dynamiques sur le Web. JSP consiste en une page HTML incluant du code Java qui s'exécutera soit sur le serveur Web, soit sur le serveur d'application. Le langage HTML décrit la manière dont s'affiche la page, le code Java servant à effectuer un traitement, par exemple récupérer les informations nécessaires pour effectuer une requête dans une base de données [Duane et Mark, 2000].

2.5.9. PHP

PHP est un langage de script utilisé le plus souvent côté serveur : dans cette architecture, le serveur interprète le code PHP des pages web demandées et génère du code (HTML, XHTML, CSS par exemple) et des données (JPEG, GIF, PNG par exemple) pouvant être interprétées et rendues par un navigateur. PHP peut également générer d'autres formats comme le PDF.

Il a été conçu pour permettre la création d'applications dynamiques, le plus souvent développées pour le Web. PHP est le plus souvent couplé à un serveur Apache bien qu'il puisse être installé sur la plupart des serveurs HTTP tel que IIS. Ce couplage permet de récupérer des informations issues d'une base de données, d'un système de fichiers(contenu de fichiers et de l'arborescence) ou plus simplement des données envoyées par le navigateur afin d'être interprétées ou stockées pour une utilisation ultérieure.

C'est un langage peu typé et souple et donc facile à apprendre par un débutant mais, de ce fait, des failles de sécurité peuvent rapidement apparaître dans les applications. Pragmatique, PHP ne s'encombre pas de théorie et a tendance à choisir le chemin le plus direct. Néanmoins, le nom des fonctions (ainsi que le passage des arguments) ne respecte pas toujours une logique uniforme, ce qui peut être préjudiciable à l'apprentissage.

Son utilisation commence avec le traitement des formulaires puis par l'accès aux bases de données. L'accès aux bases de données est aisé une fois l'installation des modules correspondants effectuée sur le serveur. La force la plus évidente de ce langage est qu'il a permis au fil du temps la résolution aisée de problèmes autrefois compliqués et est devenu par conséquent un composant incontournable des offres d'hébergements.

Il est multi plate-forme : autant sur Linux qu'avec Windows il permet aisément de reconduire le même code sur un environnement à peu près semblable (prendre en compte les règles d'arborescence de répertoires qui peuvent changer).

Libre, gratuit, simple d'utilisation et d'installation, ce langage nécessite comme tout langage de programmation une bonne compréhension des principales fonctions usuelles ainsi qu'une connaissance aiguë des problèmes de sécurité liés à ce langage.

La version 5.3 a introduit de nombreuses fonctionnalités : les espaces de noms — un élément fondamental de l'élaboration d'extensions, de bibliothèques et de frame works structurés, les fonctions anonymes, les fermetures, etc.

Aujourd'hui près de 80% des sites internet utilisent le langage PHP sous ses différentes versions. Plusieurs développeurs PHP responsables de ces sites utilisent en majorité la version 5.4 (38%) dans leurs missions quotidiennes [Frédéric et al, 2001].

2.5.10. CSS

Le concept de **feuilles de style** est apparu en 1996 avec la publication par le W3C d'une nouvelle recommandation intitulée « *Cascading Style Sheets* » (*feuilles de style en cascade*), notée **CSS**.

Le principe des feuilles de style consiste à regrouper dans un même document des caractéristiques de mise en forme associées à des groupes d'éléments. Il suffit de définir par un nom un ensemble de définitions et de caractéristiques de mise en forme, et de l'appeler pour l'appliquer à un texte. Il est ainsi possible de créer un groupe de titres en police Arial, de couleur verte et en italique.

Les feuilles de style ont été mises au point afin de compenser les manques du langage HTML en ce qui concerne la mise en page et la présentation. En effet, le HTML offre un certain nombre de balises permettant de mettre en page et de définir le style d'un texte, toutefois chaque élément possède son propre style, indépendamment des éléments qui l'entourent. Grâce aux feuilles de style, lorsque la charte graphique d'un site composé de plusieurs centaines de pages web doit être changée, il suffit de modifier la définition des feuilles de style en un seul endroit pour changer l'apparence du site tout entier !

Elles sont appellées « *feuilles de style en* ***cascade*** » (en anglais « **Cascading Style Sheets** ») car il est possible d'en définir plusieurs et que les styles peuvent être hérités en cascade.

Les feuilles de style permettent notamment :

- d'obtenir une présentation homogène sur tout un site en faisant appel sur toutes les pages à une même définition de style ;
- de permettre le changement de l'aspect d'un site complet entier par la seule modification de quelques lignes ;
- une plus grande lisibilité du HTML, car les styles sont définis à part ;

- des chargements de page plus rapides, pour les mêmes raisons que précédemment ;
- un positionnement plus rigoureux des éléments [site 2].

2.5.11. CGI

Un script **CGI** (*Common Gateway Interface, traduction interface de passerelle commune*) est un programme exécuté par le serveur web (on dit généralement « côté serveur »), permettant d'envoyer au navigateur de l'internaute un code HTML créé automatiquement par le serveur (basé par exemple sur une autre application, telle qu'un système de gestion de base de données, d'où le nom de passerelle).

Un des principaux intérêts de l'utilisation de CGI est la possibilité de fournir des pages dynamiques, c'est-à-dire des pages personnalisées selon un choix ou une saisie de l'utilisateur. L'application la plus fréquente de cette technique repose sur l'utilisation de formulaires HTML permettant à l'utilisateur de choisir ou de saisir des données, puis de cliquer sur un bouton de soumission du formulaire, envoyant alors les données du formulaire en paramètre du script CGI. **[Site 3]**

Les serveurs Web qui mettent en œuvre CGI agissent comme une passerelle entre la requête de l'utilisateur et les données dont elle a besoin. Pour cela, ils commencent par créer un processus dans lequel le programme sera exécuté (voir figure suivante). Ensuite, ils chargent les environnements d'exécution nécessaires, ainsi que le programme. Enfin, ils transmettent une requête et appellent le programme. Une fois le programme terminé, le serveur Web lit la réponse dans le *stdout*.

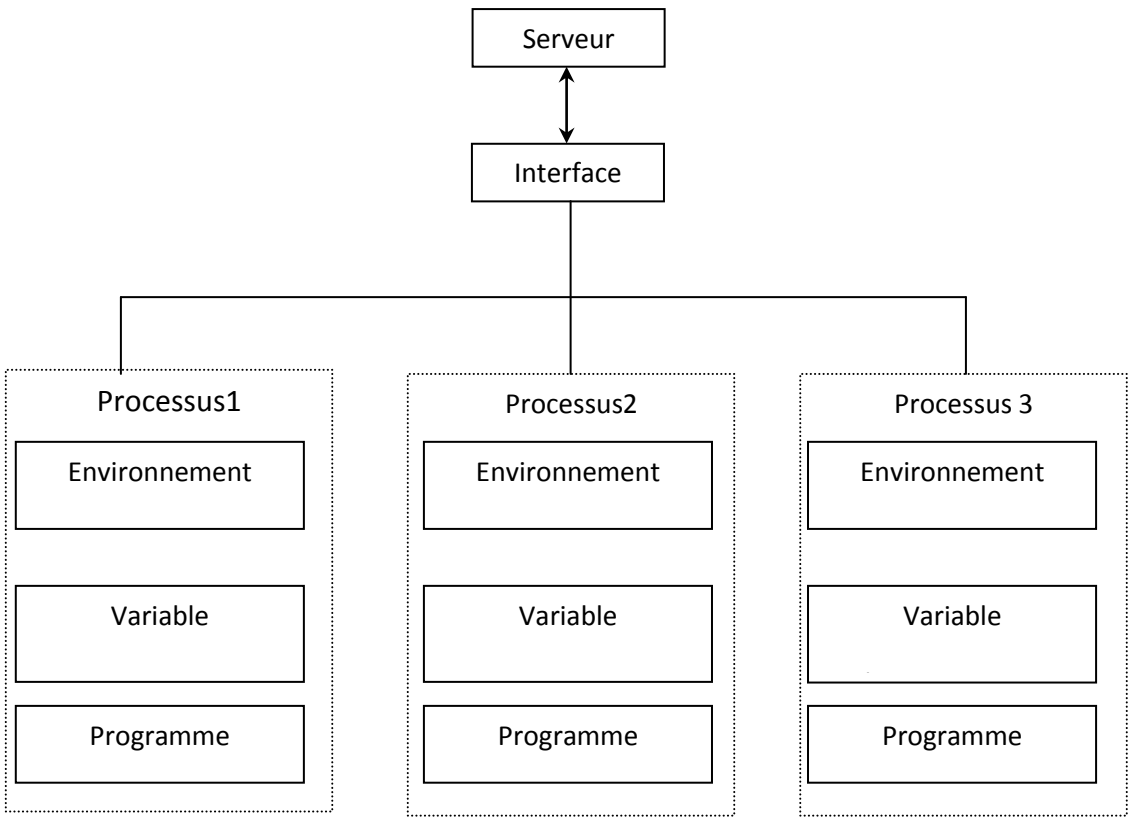


FIG.2.3– Principe de CGI.

L'inconvénient majeur de CGI réside dans son manque d'évolutivité. En effet, chaque fois que le serveur Web reçoit une requête, un processus est créé. Chaque processus possède un jeu de variables d'environnement, une instance de l'environnement d'exécution requis, une copie du programme, et un espace mémoire alloué au programme. Il est aisé de deviner ce qui se produit sur le serveur lorsqu'un grand nombre de requêtes sont reçues au même moment. Dans ce cas, le serveur est très sollicité, et peut même aller jusqu'à « se planter ».

C'est là qu'interviennent les techniques tels que *FastCGI* et *mod_perl* d'Apache. Elles permettent en effet de résoudre des problèmes de performance : *FastCGI,* en partageant une instance de chaque programme CGI, et *mod_perl*, en interprétant et en exécutant les scripts Perl au sein du serveur Web Apache.

2.5.12. ASP (Active Server Pages)

Microsoft vient de lancer une technique Web, appelée ASP *(Active Server Pages),* qui réunit dans un seul fichier le code HTML, les scripts et des composants côté serveur. Lorsque le serveur reçoit une requête relative à un fichier ASP, il recherche d'abord la page compilée, puis l'exécute. Si la page n'a pas encore été compilée, le serveur s'en charge, puis l'exécute. On obtient alors une page Web finie, qui est renvoyée au navigateur.

Une page ASP peut être écrite en HTML, JScript, ou VBScript. Grâce aux scripts, elle peut accéder aux composants côté serveur. Ces composants peuvent être écrits dans un langage quelconque, à condition qu'il possède une interface COM (spécification de composant Microsoft). L'avantage majeur des fichiers ASP est que tout est exécuté sur le serveur. Ainsi, les pages ne sont pas dépendantes d'un navigateur particulier. Elles sont uniquement limitées par les capacités du serveur.

Par contre, la technique ASP(Active Server Pages) présente un inconvénient majeur : elle ne peut être utilisée qu'avec un serveur Web Microsoft (IIS, ou le serveur Web personnel, PWS) s'exécutant sur un système d'exploitation Microsoft *(Win9x,* WinNT). Il est possible de transférer cette technologie vers d'autres plates-formes et serveurs Web. Toutefois, la prise en charge COM étant limitée, ASP a une utilité limitée. Un autre problème important est lié à la combinaison de script et de code HTML, c'est-à-dire aux deux groupes d'informations, qui peut vite devenir une contrainte [Marty et Larry ,2000].

2.6. Les pages web dynamiques

On appelle « **page web statique** » une page web constituée d'un fichier texte contenant un code HTML et éventuellement des images et des liens vers d'autres documents. Un site constitué de pages web statiques se verra ainsi qualifié de « **site web statique** ».

Un site web statique est suffisant lorsqu'il ne contient que quelques dizaines de pages mais son exploitation et sa mise à jour peuvent vite atteindre les limites suivantes :

- une maintenance difficile due à l'obligation de modifier manuellement chacune des pages (notamment dans le cas où toutes les pages possèdent un même menu)
- l'impossibilité de renvoyer une page personnalisée selon le visiteur
- l'impossibilité de créer une page dynamiquement selon les entrées d'une base de données
- etc.

C'est pourquoi des solutions permettant d'automatiser la génération de pages web du côté du serveur ont été mises au point. Il existe de nombreuses solutions permettant de mettre en œuvre un langage de script sur le serveur web, parmi lesquelles les plus répandues sont les suivantes :

- La première historiquement, appelée CGI (*Common Gateway Interface*) consistait à interpréter des programmes (généralement écrits en perl ou en langage C), puis de leur faire renvoyer un contenu compatible avec le protocole HTTP.

- Le langage ASP (*Active Server Pages*) de Microsoft a permis de simplifier l'écriture de tels scripts en manipulant des objets en VBScript.

- Le langage PHP (*Hypertext preprocessor*) emploie son propre langage (dérivé du C++ et de Perl) et permet de nombreuses fonctionnalités (équivalentes à celles de la technologie ASP).

- Le langage JSP (*Java Server Pages*) est la plus récente parmi ces technologies. Elle permet d'utiliser toute la puissance de Java pour créer des pages web dynamiques[Duane et Mark, 2000].

2.7. Création

Les applications web sont souvent créées par des équipes composées à la fois de développeurs et de designers. Le développement nécessite la connaissance des différents langages utilisés dans les technologies du Web : HTML pour la présentation des pages, CSS pour la charte graphique, JavaScript, Java ou ActionScript pour les automatismes exécutés par le client, ainsi qu'un langage tel que Java, PHP, C# ou VBScript pour les automatismes exécutés par le serveur.

Les applications web sont faites d'un ensemble de composants logiciels et de pages "porteuses" ; les composants sont regroupés dans des *bibliothèques logicielles* (*servlets*).

Un logiciel *serveur web* prévu à cet effet (serveur d'applications web) exécute un composant donné lors de la réception de chaque requête. ASP. NET, Websphere, JBoss ou Apache Tomcat sont des logiciels serveurs d'application web.

Une application web est typiquement utilisée simultanément par plusieurs usagers ; elle est équipée de mécanismes de contrôle d'accès logique, ceux-ci sont basés sur les mécanismes de contrôle d'accès propre au serveur d'application web et au système d'exploitation. Ils utilisent parfois des mécanismes existants tels que l'authentification unique(Single sign-on).

Pour les travaux de construction, les ingénieurs utilisent des environnements de développement intégré qui aident à la fois à la programmation informatique et la conception de site Web tels que Visual Studio, Eclipse et NetBeans [Richard ,2000].

2.8. Sécurité

La sécurité d'une application web est étroitement liée à l'environnement qui l'héberge. Des failles telles le cross-site scripting peuvent mettre l'application en péril si le développeur n'a pas pris de précaution pour sécuriser son code. Différentes façons d'attaquer une application peuvent être utilisées, par exemple l'injection SQL ou JavaScript.

Exemple

L'attaque par tautologie consiste à rendre toujours vraie une condition suivant un WHERE. La requête SQL ainsi construite retournera donc toujours au moins un résultat lors de son exécution. Cette attaque est couramment utilisée pour contourner une authentification ou pour récupérer les données d'une table. L'attaquant doit modifier la structure syntaxique de la condition et introduire une tautologie qui permettra de modifier le comportement de la requête

Considérons une requête d'authentification nécessitant un mot de passe et un pseudonyme, si un résultat est retourné alors on estime que l'utilisateur est authentifié.

```
SELECT id FROM Users WHERE login='(Pseudonyme)' AND password='(Mot de passe)'
```

Si aucune vérification n'est effectuée alors l'attaquant peut contourner le système d'authentification en insérant ' OR 1=1 -- dans login.

```
SELECT id FROM Users WHERE login='' OR 1=1 --' and password=''
```

Ici la tautologie 1=1 est insérée de manière à rendre inutile toutes les autres conditions de la requête. Pour que la tautologie ne soit pas considérée comme une simple chaîne et puisse donc être évaluée par le moteur SQL il suffit d'utiliser le symbole ' délimitant une

chaine de caractère. Le symbole -- de commentaire SQL permet de s'assurer que le reste de la requête sera ignoré lors de l'évaluation [David et al, 2002].

2.9. Pourquoi utiliser Java ?

Devant la multitude de produits de développement côté serveur, la question du pourquoi Java est si exceptionnelle et se pose obligatoirement. La réponse est simple : les techniques Java côté serveur offrent l'indépendance à l'égard des plates-formes, l'efficacité, l'accès à d'autres API de l'entreprise, la réutilisation et la modularité. Chacune de ces caractéristiques est détaillée dans ce qui suit :

2.9.1. Indépendance à l'égard des plates-formes

Les *servlets* Java satisfont au même modèle d'indépendance que le langage Java. Le code de la *servlet* est compilé en byte codes, qui sont à leur tour, interprétés sur le serveur Web, par une machine virtuelle Java (JVM) spécifique à la plate-forme. Ces *Servlet* étant composées de byte codes indépendants des plates-formes, on peut les déplacer à loisir entre les plates-formes, à condition qu'elles prennent en charge Java.

2.9.2. Efficacité

Comparées à **CGI** *(Common Gateway Interface),* les *servlets* Java offrent une méthode de gestion des requêtes utilisateur beaucoup plus efficace. Parce qu'un programme CGI doit créer un processus distinct pour chaque requête utilisateur. Lorsqu'une *servlet* reçoit une requête utilisateur, elle se contente de générer un nouveau thread au sein du même processus et traite la requête. Cette méthode permet à des centaines, voire des milliers d'utilisateurs d'accéder à la même *servlet* simultanément, sans compromettre les performances du serveur.

2.9.3. Accès aux API Java de l'entreprise

Les *servlets* étant une extension de la plate-forme Java, elles peuvent accéder à toutes les API Java, En effet, une *servlet* Java peut envoyer et recevoir des messages électroniques, appeler des méthodes d'objets distants à l'aide de RMI ou CORBA, obtenir des informations relatives aux répertoires via JNDI, exploiter les EJB *[Enterprise JavaBeun), ou* tout autre élément de la plate-forme Java[Marty et Larry ,2000].

2.9.4. Réutilisation

La réutilisation de code est le Saint Graal de tout programmeur. La création de composants d'une application est l'une des méthodes de réutilisation de code, l'utilisation d'objets *pour* encapsuler des fonctionnalités partagées, en est une autre. Java utilise ces deux méthodes. Il s'agit d'un langage 100 % orienté objet. C'est pour cela qu'il intègre la réutilisation de code.

2.9.5. Modularité

Lors du développement d'une application complète côté serveur, les programmes développés peuvent rapidement gagner en volume et en complexité. Il vaut toujours mieux diviser une application en différents modules, chacun étant chargé d'une tâche bien définie. Ainsi, l'application est beaucoup plus facile à gérer et à comprendre. Les *servlets* Java, les JSP et JavaBeans permettent de séparer l'application en différents modules, de la décomposer en niveaux et en tâches.

Jusqu'à présent, aucune technique globale ne permettait de développer des applications Web côté serveur. Les API Java Enterprise offrent une technologie simple, efficace, indépendante des plates-formes, orientée objet, et extensible, qui sert à intégrer des données dynamiques au sein du World Wide Web.

2.10. Conclusion

Dans ce chapitre, les points suivants ont été traités **:**

- Une application Web se compose d'un navigateur Web qui recueille les données de l'utilisateur, les envoie vers le serveur Web pour traitement, et qui reçoit les résultats en fin de processus.

- Une requête HTTP est généralement une requête GET ou POST. GET est utilisée pour extraire des informations. Elle envoie les données en même temps que l'URL de la requête. POST est utilisée lorsque la requête modifie les données contenues sur le serveur. Toutes les données de la page sont assemblées et envoyées immédiatement en réponse à la requête.

- Un programme serveur est chargé de recevoir la requête, de la traiter et d'envoyer la réponse au serveur Web (qui, à son tour, l'envoie au navigateur). Les techniques les plus utilisées côté serveur sont les suivantes : CGI, JSP, PHP, ASP,ASP.NET et les *servlets* Java.

- Les *servlels* Java sont indépendantes des plates-formes, performantes et 100 % orientées objet. Elles permettent aux développeurs de faire bénéficier le serveur Web des avantages de la plate-forme Java.

- Les JSP sont moins code Java que les Servlets puisqu'elles contiennent du code HTML, elles font appel à JavaBeans pour pouvoir accéder à une base de données ORACLE MYSQL...etc.

Nous présenterons dans le chapitre suivant les détails de notre approche.

3.1. Introduction

L'informatisation est le phénomène le plus important de notre époque. Elle s'immisce maintenant dans la plus part des objets de la vie courante et ce, que ce soit dans l'objet proprement dit, ou bien dans le processus de conception ou de fabrication de cet objet. Actuellement, l'informatique est au cœur de toutes les entreprises, et un outil indispensable pour leur développement. Nous présentant dans ce qui suit notre démarche par la conception d'une plate-forme Web.

3.2 Modélisation

3.2.1 Qu'est-ce qu'un modèle ?

Un modèle est une représentation abstraite et simplifiée, d'une entité (phénomène, processus, système, etc.) du monde réel en vue de le décrire, de l'expliquer ou de le prévoir. Le modèle est synonyme de théorie, mais avec une connotation pratique : un modèle, c'est une théorie orientée vers l'action qu'elle doit servir.

Un modèle permet de réduire la complexité d'un phénomène en éliminant les détails qui n'influencent pas son comportement de manière significative. Il reflète ce que le concepteur croit important pour la compréhension et la prédiction du phénomène modélisé. Les limites du phénomène modélisé dépendent des objectifs du modèle en question.

3.2.2 Pourquoi modéliser ?

Modéliser un système avant sa réalisation permet de mieux comprendre le fonctionnement du système. C'est également un bon moyen de maîtriser sa complexité et d'assurer sa cohérence. Un modèle est un langage commun, précis, qui est connu par tous les membres de l'équipe et il est donc, à ce titre, un vecteur privilégié pour communiquer. Cette communication est essentielle pour aboutir à une compréhension commune aux différentes parties prenantes et précise d'un problème donné.

Dans le domaine de l'ingénierie du logiciel, le modèle permet de mieux répartir les tâches et d'automatiser certaines d'entre elles. C'est également un facteur de réduction des coûts et des délais. Par exemple, les plateformes de modélisation savent maintenant exploiter les modèles pour faire de la génération de code (au moins au niveau du squelette) voire des aller et des retours entre le code et le modèle sans perte d'information. Le modèle est enfin indispensable pour assurer un bon niveau de qualité et une maintenance efficace. En effet, une fois mise en production, l'application va devoir être maintenue, probablement par une

autre équipe et, qui n'est, pas nécessairement de la même société que celle ayant créée l'application.

Le choix du modèle a donc une influence capitale sur les solutions obtenues. Les systèmes non-triviaux sont mieux modélisés par un ensemble de modèles indépendants. Selon les modèles employés, la démarche de modélisation n'est pas la même.

3.2.3 Qui doit modéliser ?

La modélisation est souvent faite par la maîtrise d'œuvre informatique (MOE[1]). C'est malencontreux, car les priorités de la MOE résident dans le fonctionnement de la plate-forme informatique et non dans les processus de l'entreprise.

Il est préférable que la modélisation soit réalisée par la maîtrise d'ouvrage (MOA[2]) de sorte que le métier soit maître de ses propres concepts. La MOE doit intervenir dans le modèle lorsque, après avoir défini les concepts du métier, on doit introduire les contraintes propres à la plate-forme informatique.

Il est vrai que certains métiers, dont les priorités sont opérationnelles, ne disposent pas toujours de la capacité d'abstraction et de la rigueur conceptuelle nécessaires à la formalisation. La professionnalisation de la MOA a pour but de les doter de ces compétences. Cette professionnalisation réside essentiellement dans l'aptitude à modéliser le système d'information du métier : le maître mot est modélisation. Lorsque le modèle du système d'information est de bonne qualité, sobre, clair, stable, la maîtrise d'œuvre peut travailler dans de bonnes conditions. Lorsque cette professionnalisation a lieu, elle modifie les rapports avec l'informatique et déplace la frontière des responsabilités, ce qui contrarie parfois les informaticiens dans un premier temps, avant qu'ils n'en voient apparaître les résultats de leur travail.

3.3 UML

3.3.1 Définition

Unified Modeling Language ou langage de modélisation unifié est un langage de modélisation graphique à base de pictogrammes. Il est apparu dans le monde du génie logiciel, dans le cadre de la « conception orientée objet ». UML est couramment utilisé dans les projets logiciels.

UML est donc un métalangage car il fournit les éléments permettant de construire le modèle qui, lui, sera le langage du projet, il est impossible de donner une représentation graphique complète d'un logiciel, ou de tout autre système complexe. Mais il est possible de donner sur un tel système des vues partielles, analogues chacune à une photographie d'une statue, et dont la conjonction donnera une idée utilisable en pratique sans risque d'erreur grave.

[1]MOE : Le terme maîtrise d'œuvre (souvent abrégé MOE ou MŒ) désigne une personne ou entité chargée de la conception puis de la conduite opérationnelle de travaux généralement pour le compte d'autrui.
[2]MOA : maîtrise d'ouvrage représente le client du projet, celui sera normalement le propriétaire de l'ouvrage.

3.3.2 Historique

UML est né de la fusion des trois méthodes qui ont le plus influencé la modélisation objet au milieu des années 90 : OMT, Boochet OOSE. UML a été inventé par des experts de grande renommée. En quelques années, UML est devenu un langage incontournable. L'approche objet est pourtant loin d'être une idée récente. Les premiers compilateurs C++ datent du début des années80 et de nombreux langages orientés objets "académiques" ont étayé les concepts objets (Eiffel, Objective C, Loops...). L'unification et la normalisation des méthodes objet dominantes (OMT, Booch et OOSE) ne datent que de1995. UML est le fruit de cette fusion. UML, ainsi que les méthodes dont il est issu, s'accordent sur un point : une analyse objet passe par une modélisation objet.

Fin 1997, UML est devenu une norme OMG (Object Management Group).

UML comble une lacune importante des technologies objet. Il permet d'exprimer et d'élaborer des modèles objet, indépendamment de tout langage de programmation. Mais UML offre bien plus encore, c'est un langage formel, défini par un méta modèle. Le méta modèle d'UML décrit de manière très précise tous les éléments de modélisation ainsi que la sémantique de ces éléments. En d'autres termes : UML normalise les concepts objet.

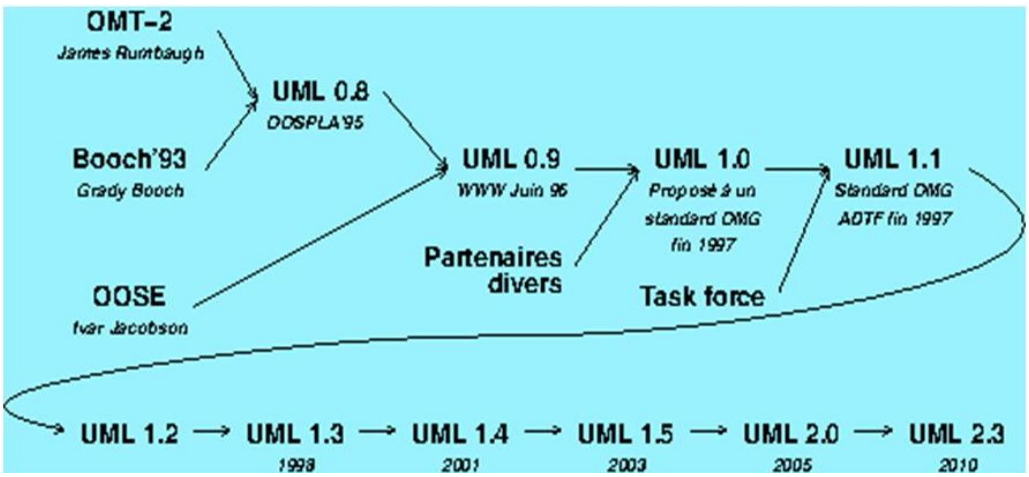


FIG. 3.1— Historique de UML

3.3.3 Les diagrammes d'UML

3.3.3.1 Les diagrammes structurels

a- Diagramme de classes :

Le diagramme de classes est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces des systèmes ainsi que les différentes relations entre celles-ci. Ce diagramme fait partie du modèle statique d'UML car il fait abstraction des aspects temporels et dynamiques.

b- Diagramme d'objets :

Le diagramme d'objets permet d'éclairer un diagramme de classes en l'illustrant par des exemples. Il est utilisé pour vérifier l'adéquation d'un diagramme de classes à différents cas possibles.

c- Diagramme de déploiement :

Un diagramme de déploiement décrit la disposition physique des ressources matérielles qui composent le système et montre la répartition des composants sur ces matériels. Chaque ressource étant matérialisée par un nœud, le diagramme de déploiement précise comment les composants sont répartis sur les nœuds et quelles sont les connexions entre les composants ou les nœuds.

d- Diagramme de packages :

Les diagrammes de paquetage peuvent utiliser des paquetages pour illustrer les différentes couches de l'architecture en couches d'un système logiciel. Les dépendances entre paquetages peuvent être parées d'étiquettes ou de stéréotypes pour indiquer les mécanismes de communication entre les couches.

e- Diagramme de composants :

Le diagramme de composants décrit l'organisation du système du point de vue des éléments logiciels comme les modules (paquetages, fichiers sources, bibliothèques, exécutables), des données (fichiers, bases de données) ou encore d'éléments de configuration (paramètres, scripts, fichiers de commandes). Ce diagramme permet de mettre en évidence les dépendances entre les composants.

f- Diagramme de structure composite :

Le diagramme de structure composite expose la structure interne d'une classe ainsi que les collaborations que cette dernière rend possible. Les éléments de ce diagramme sont les parties, les ports par le biais desquels les parties interagissent entre elles, avec différentes instances de la classe ou encore avec le monde extérieur, et enfin les connecteurs reliant les parties et les ports. Une structure composite est un ensemble d'éléments interconnectés collaborant dans un but commun lors de l'exécution d'une tâche. Chaque élément se voit attribuer un rôle dans la collaboration.

3.3.3.2 Les diagrammes de comportement

a- Diagramme de cas d'utilisation :

Le diagramme de cas d'utilisation décrit la succession des opérations réalisées par un acteur (personne qui assure l'exécution d'une activité). C'est le diagramme principal du modèle UML, celui où s'assure la relation entre l'utilisateur et les objets que le système met en œuvre.

b- Diagramme d'activité :

Le diagramme d'activité n'est autre que la transcription dans UML de la représentation du processus telle qu'elle a été élaboré lors du travail qui a préparé la modélisation : il montre l'enchaînement des activités qui concourent au processus.

c- Diagramme d’état :

Le diagramme d'état représente la façon dont évoluent durant le processus les objets appartenant à une même classe. La modélisation du cycle de vie est essentielle pour représenter et mettre en forme la dynamique du système.

d- Diagramme de communication :

Contrairement à un diagramme de séquence, un diagramme de communication rend compte de l’organisation spatiale des participants à l’interaction, il est souvent utilisé pour illustrer un cas d’utilisation ou pour décrire une opération. Le diagramme de communication aide à valider les associations du diagramme de classe en les utilisant comme support de transmission de messages.

e- Diagramme de temps :

Les diagrammes de temps sont utilisés pour explorer le comportement des objets d'un système à travers une période de temps.

3.4 Schéma de spécification de notre approche

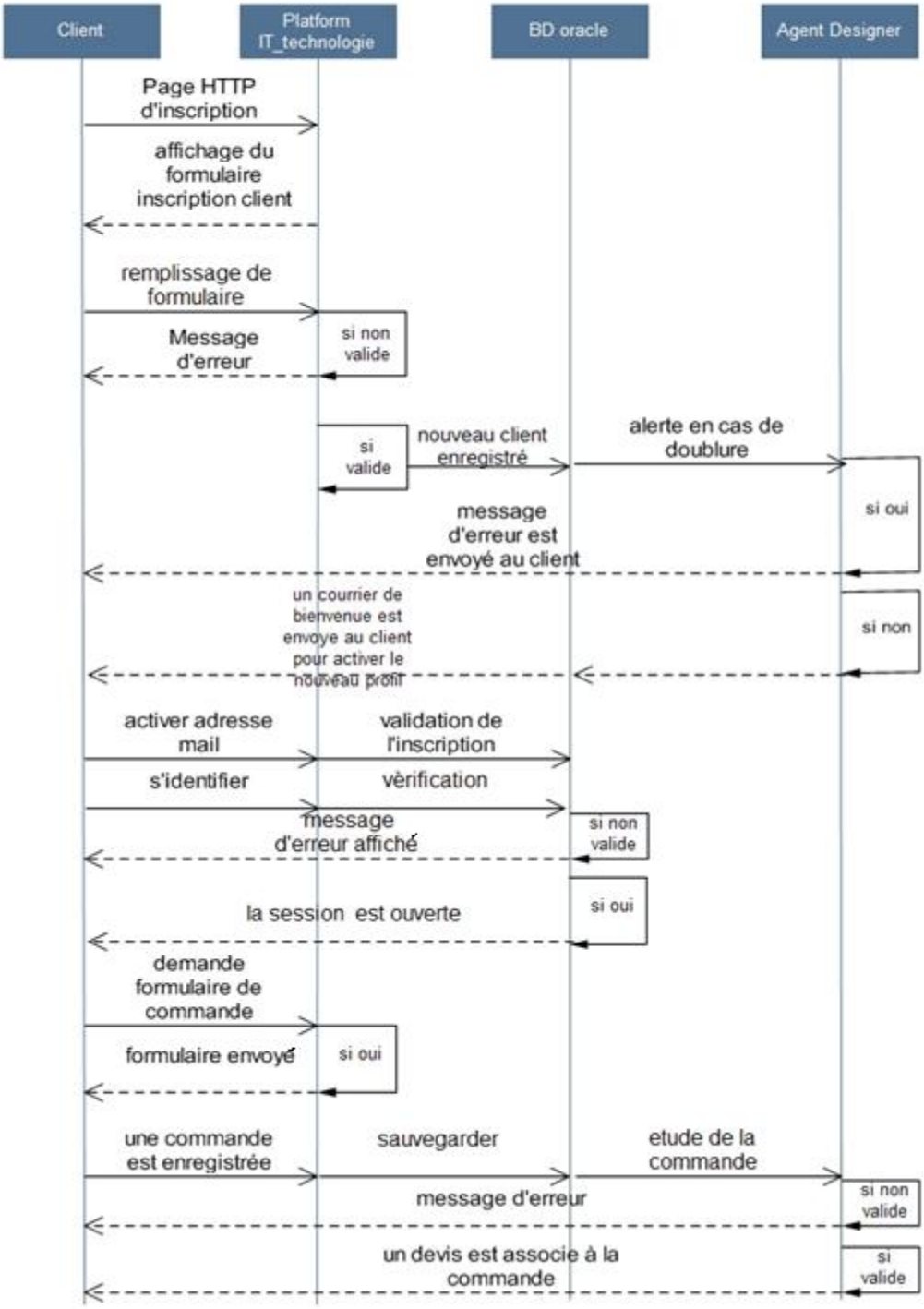


FIG. 3.2— : Diagramme de séquence

3.5. Le modèle proposé

3.5.1 Description des agents

Notre approche s'articule autour de 5 agents distincts dont les taches se complètent comme suit :

- L'Agent administrateur
- L'Agent Data.
- L'Agent de production.
- L'Agent commercial.
- L'Agent designer.

Ces agents ont pour rôle de réaliser et faire le suivi des tâches suivantes:

3.5.1.1 L'Agent administrateur

C'est le premier agent qui est responsable du bon fonctionnement du système et la sécurité de la base des données. Il assume la responsabilité de la gestion et la manipulation du système puisque c'est l'unique agent qui a un accès physique et direct au serveur. Les tâches de l'administrateur sont les suivantes:

- ✓ Procède à l'installation, la configuration et assure le bon fonctionnement du système et de la plateforme.
- ✓ Installe, protège et Sécurise le serveur,
- ✓ Supervise la base de données **"ITtechn "** et se charge du suivi du fonctionnement ORACLE.
- ✓ Enregistre une copie, et clone la base des données,
- ✓ Procède à la réparation du système en cas de panne,
- ✓ Et se charge de la gestion de la table designer (ajouter, supprimer, modifier ....etc.),

3.5.1.2 L'Agent Data

Cet agent est chargé de contrôler des tables à savoir rechercher, masquer, détecter et consulter les données en double (cas de répétition), a l'aide de logiciel développé qui fait le balayage dans les tables de la base des données, la table client, la table commande, la table stocks ...etc., une fois qu'il détecte une ressemblance des données, le programme envoie une alerte à l'agent data qui a son tour entreprend l'action nécessaire qui génère une annulation de l'alerte ou la confirmation à l'agent administrateur. Les données en double doivent être supprimées ou traitées chaque fois que cela est possible afin de réduire les coûts et améliorer la précision des données.

3.5.1.3 L'Agent production

Cet agent permet de réaliser la gestion de production. Rappelons que la gestion de production est l'ensemble des activités qui participent à la conception, la planification des ressources (matérielles, financières, ou humaines), leur ordonnancement, l'enregistrement et la traçabilité des activités de production et le contrôle des activités de production de l'entreprise.

3.5.1.4 L’Agent commercial

L’agent commercial assure la Gestion des stocks, Quand on parle de gestion des stocks, on ne parle pas uniquement de produits finis. Il s’agit aussi du stock de matières premières ou de produits semi-finis. La gestion des stocks consiste à planifier et à mettre en œuvre une méthode pour maximiser la rentabilité, et éviter toute rupture de stock durant le processus de production.

Une bonne gestion de stock consiste à avoir la quantité nécessaire au bon moment. Si le stock n’est pas assez important on parle de rupture de stock, ce qui est mauvais pour la production qui risque d’être interrompue. Un excédent de stock, par contre, coûte cher sans oublier qu’il y a risque de dépréciation du stock.
Une bonne gestion des stocks consiste donc à **trouver cet équilibre qui permettra de maximiser le profit en minimisant les coûts.** Les prévisions et la planification sont des outils efficaces au service de la gestion des stocks.

3.5.1.5 L’Agent designer

Cet agent permet d’assurer la gestion de commande, les exigences auxquelles doivent faire face les entreprises deviennent de plus en plus complexes et des quantités d’informations toujours plus nombreuses doivent être traitées dans un temps de plus en plus court. Les exigences quant à la qualité des informations de gestion, devant pratiquement être disponibles à tout instant, augmentent dans le même temps. Seul un système de logiciel intégré permet de répondre à ces exigences.

3.5.2 Les sorties du modèle

Nous citons parmi les sorties du modèle :

3.5.2.1 Le devis

Un client peut demander un devis au fournisseur préalablement à la réalisation des pièces, des travaux, ou des prestations de services. Un devis n'est pas seulement une estimation, lorsqu'il est signé par les deux parties, il prend la valeur de contrat et constitue un engagement ferme aussi bien pour le fournisseur "Entreprise" que pour le client. Le devis est un document contractuel régi par le droit civil. Il fait donc office d'élément de référence auquel le tribunal se reporte en cas de litige entre les deux parties. Le devis doit être édité en trois exemplaires : un pour l'Entreprise et deux pour le client qui renvoie un exemplaire signé au fournisseur pour signifier son accord.

L'établissement d'un devis constitue souvent la première étape d'un engagement commercial. Sur ce document doit figurer un certain nombre d'informations, pour l'entreprise, des particularités à respecter.

Toutes les informations légales devant figurer sur un **devis d'Entreprise** sont paramétrables et apparaissent automatiquement aux endroits prédéfinis. Trois parties distinctes sont à renseigner pour créer un devis:

- Informations client, entreprise.
- Détail du devis.
- Informations 'bas de page'.

Informations client, entreprise: le client doit être enregistré dans la base des données, on choisit pour cela dans la liste des contacts, la personne (le client) pour laquelle on édite le devis. Les informations concernant cette personne (le nom ou la raison sociale, adresse, téléphone .etc.) et pour le coté entreprise on doit mentionner obligatoirement:

- La raison sociale de l'entreprise,
- la date à laquelle le devis a été rédigé,
- le numéro d'identification unique de l'entreprise,
- le lieu de son siège social (adresse),
- si l'entreprise est une société, la nature de celle-ci.

Détail du devis: on enregistre les informations qu'on souhaite faire apparaitre sur celui-ci (quantités, désignations, prix unitaires, montant global).

- le décompte détaillé, en quantité et en prix, de chaque prestation, matière et produit nécessaire à l'opération prévue: dénomination, prix unitaire et désignation de l'unité à laquelle il s'applique (taux horaire de la main d'œuvre, mètre linéaire ou mètre carré, etc. ..), et la quantité prévue,
- Les frais de déplacement, livraison,
- la somme à payer HT (Hors Taxes) ou TTC (Toutes Taxes Comprises), en précisant les différents taux de TVA (Taxe sur la Valeur Ajoutée),

Informations 'bas de page': on spécifie des montants supplémentaires (exemple: frais de dossier), les conditions de règlement, la validité du devis, Mode de paiement, etc.

- l'indication manuscrite, datée et signée du client : "Devis reçu avant la fabrication des pièces, lu et accepté",
- la mention "lu et accepté", datée et signée par l'entreprise.

Un exemple de devis,

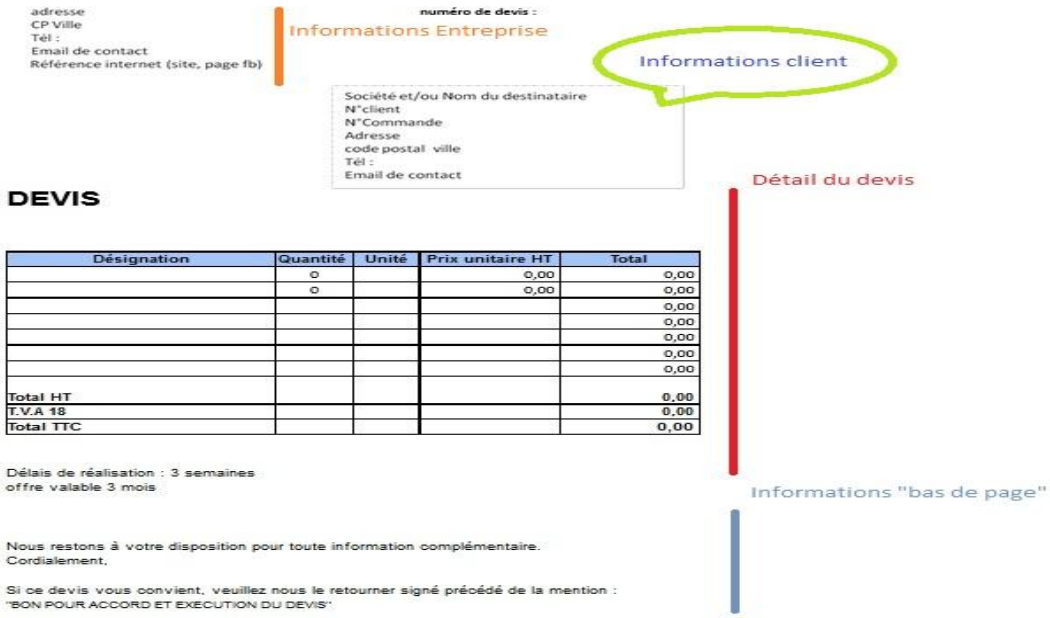


FIG. 3.3— Un exemple de devis

La Gestion des commandes joue pour cela un rôle central. Elle contrôle donc la plupart des processus, qui affectent également des applications telles que la Gestion des projets, la Gestion de la production et la gestion des stocks.

Il y a deux cas de commande :

- le premier cas : si le client commande des produits existants dans le stock : c'est le cas simple de la vente directe de produit.

- le deuxième cas : si le client commande un produit qui ne figure pas dans le catalogue de l'entreprise, le client envoi alors son fichier DAO (Dessin Assisté par Ordinateur).

3.6. Les scénarios possibles dans la gestion de la commande de vente

Ces scénarios décrivent comment traiter des commandes clients pour des ventes de produits catalogués et de nouveaux produits au fur et à mesure de clients DAO. La gestion des commandes client inclut l'avant-vente, la commande client réelle et les actions consécutives qui en résultent, telles que la livraison, les réclamations/retours et la facturation. Notons qu'il existe plusieurs scénarios pour gérer des commandes client à partir de scénarios d'un point de vente.

3.6.1 Déroulement des scénarios

Le déroulement comprend les étapes suivantes :

1. Création de commandes clients standards, de commandes d'urgence, de commandes à livraison directe, commandes avec fichier DAO.
2. Traitement et ré-ordonnancement des commandes (balayage de table).
3. Etude de la commande et envoi du devis au clients.
4. Etablissement des factures client.
5. Gestion des paiements par l'agent commercial.

1- La création de la commande se fait par le client directement au site web en remplissant le formulaire dans la page 'commande.jsp', et aussi la page produite où il peut sélectionner directement des articles de catalogue.

2-Le balayage de la table commande aide l'agent data à détecter les répétitions de commande pour chaque client, et à améliorer la précision des données.

3.6.2 les scénarios

3.6.2.1 Le premier cas : commande en ligne

- Commandes à livraison directe: le client demande des articles à partir du catalogue, un devis sera envoyé dans un court délai "directement" avec les prix prédéfinis dans la base des données "Table produit" incluant les frais supplémentaires (transport , délai....). Dans ce cas, l'agent commercial peut mettre une commande en état d'attente, ou la refuser en cas de rupture de stocks.

3.6.2.2 Le deuxième cas : commande avec conception à valider

- Commande avec un fichier de DAO:

Dans la partie traitements des commandes par l'agent designer on trouve plusieurs scénarios.

1 - Un Rejet définitif avec une conception correcte, la commande est refusée par l'agent commercial pour l'unique raison si l'entreprise n'a pas les moyens pour fabriquer la pièce demandée Soit elle ne dispose pas de la matière première, dans ce cas précis, le client sera dirigé vers d'autres entreprises s'il le désire.

2- L'agent designer fait une étude sur la pièce dessinée dans le fichier DAO; s'il détecte une anomalie dans le design, la commande sera bloquée en informant le client par un message détaillé sur l'erreur de la conception, si le client envoie de nouveau la conception corrigée l'agent designer revient pour contrôler de nouveau la conception, il peut contrôler jusqu'a quatre fois. La quatrième correction sera le dernier contrôle et

s'il trouve de nouveau des erreurs dans le schéma, la commande sera classée en avisant le client par un mail.

3- Si la conception est correcte mais l'entreprise ne peut accomplir le travail dans le délai demandé par le client. Faute de personnel ou une surcharge dans les usines et les ateliers; dans ce cas, le client recevra plusieurs propositions de l'agent commercial par exemple celle qui comporte des devis sur sa pièce avec plusieurs délais "c'est à dire que chaque devis établit un délai avec un taux de remise".

4- Si le client formule une commande avec un fichier de conception correct avec un délai possible à accomplir de la part de l'entreprise, alors l'agent designer passera la commande, un devis sera établi par l'agent commercial et envoyé à l'adresse mail du clients.

Si la conception est bonne un devis complet sera envoyé à la boite Email du client**.** La **Validation de la commande** Une commande n'est définitive que lorsqu' elle a été confirmée par un message électronique d'acceptation du devis et des présentes conditions, rédigée par le client ou toute personne dûment mandatée à cet effet.

3.6.2.3 Le troisième cas : commande urgente

La commande en urgence à des priorités sur les autres commandes, l'agent designer la traite en premier avec un coût supplémentaire.

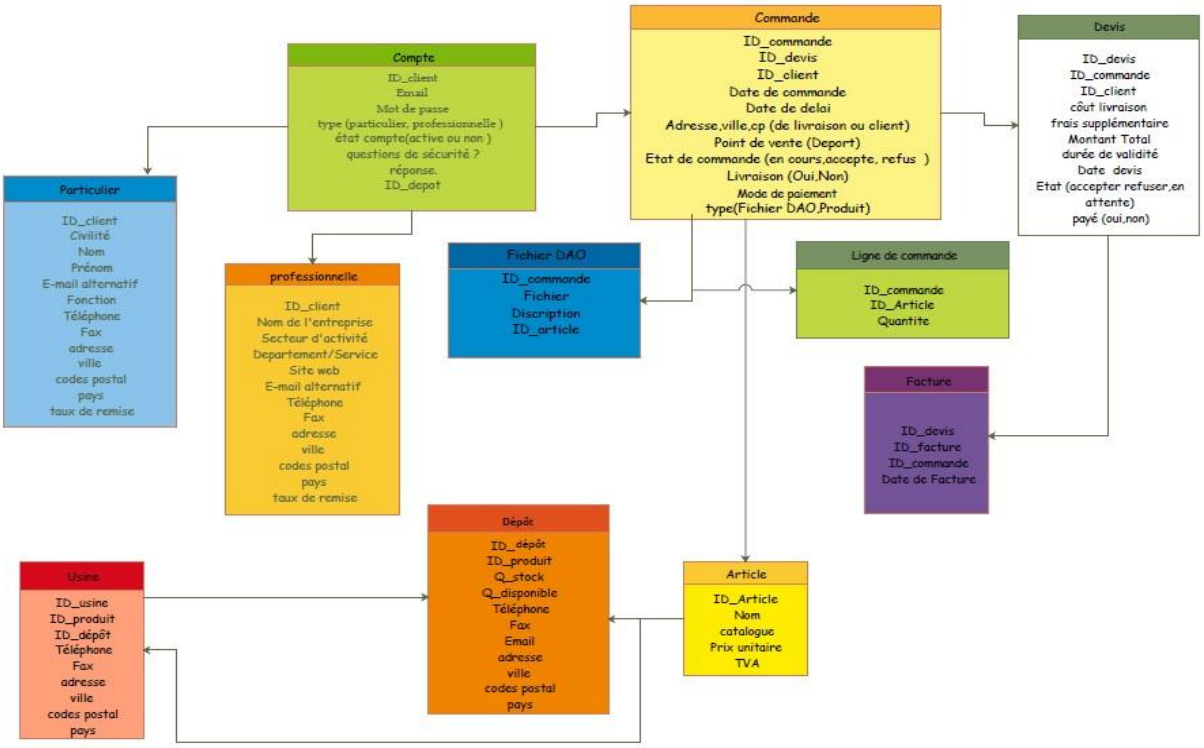
Le devis n'est pas en soi un contrat, mais un engagement unilatéral qui ne devient un contrat que lorsque le devis a été accepté par le client au quel il est remis; A moins que le devis n'indique la limite de temps pendant lequel l'entreprise s'engage à maintenir son offre, celle-ci est censée être faite pour un temps raisonnable dont la longueur est fonction des usages de la profession à laquelle appartient l'entreprise. En particulier, on considère que les commandes urgentes sont celles ou les devis de livraison ne dépassent pas les 24 h.

3.6.3. Le choix du mode de paiement

Le client doit choisir un type de virement après avoir accepté le devis,

- Carte de crédit
- Chèques personnels ou chèques de banque
- Virement bancaire

3.6.4. Le Modèle conceptuel des données



3.7 Conclusion

Dans ce chapitre nous avons modélisé notre application avec le langage UML, cette modélisation nous a permis de bien comprendre le fonctionnement de notre application et nous a aidé à bien préparer la phase de l'implémentation.

Dans le chapitre suivant nous allons décrire la phase d'implémentation de notre application.

4.1 Introduction

Ce chapitre est consacré essentiellement au développement d'une application basée sur le concept de l'architecture JEE. L'application porte sur l'étude de cas de la conception et vente de pièces mécaniques par un site web développe dans la plat-forme Java. Dans cette implémentation, nous allons respecter le diagramme de séquence qui a été décrit dans le chapitre précédant.

Au début, nous allons définir brièvement les outils utilisés, argumenter leur utilisation, puis décrire et donner un aperçu du fonctionnement de notre plate-forme.

Ensuite, dans le but de rendre explicite notre étude, nous allons afficher les interfaces de chaque cas d'utilisation, puis mentionner les codes appropriés.

4.2 Les outils utilisés

4.2.1 Netbeans 8.0.2

Cet IDE a été créé à l'initiative de Sun Microsystems. Il présente toutes les caractéristiques indispensables à un environnement de qualité, que ce soit pour développer en Java, Ruby, C/C++ ou même PHP.

NetBeans est sous licence OpenSource, il permet de développer et déployer rapidement et gratuitement des applications graphiques Swing, des Applets, des JSP/Servlets, des architectures J2EE, dans un environnement fortement personnalisable.

FIG. 4.1 — Logo de Netbeans.

L'IDE NetBeans repose sur un noyau robuste, la plateforme NetBeans, qu'on peut également utiliser pour développer nos propres applications Java, et un système de plugins performant, qui permet d'avoir un IDE modulable.

A coté de la version complète de l'IDE NetBeans, il existe différentes déclinaisons qui se concentrent sur une plateforme ou un langage précis (Java ME, Java : SE + ME + EE, Ruby, C/C++, PHP).

NetBeans contient, en plus du support pour CVS et SubVersion, un support pour ClearCase, mais aussi pour Mercurial.

Il permet également de déployer des applications Web, non seulement vers Tomcat et Glassfish qui sont livrées avec le "Pack Web", mais aussi vers JBoss, WebSphere 6.1, WebLogic 9.

NetBeans détient un support de développement d'applications Web avec des améliorations pour l'édition des JSP, la gestion serveur et le support des dernières versions de Tomcat.

Enfin cet IDE possède un débogueur de grande qualité ainsi qu'une interface graphique améliorée.

NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X et Open VMS[Site 8].

4.2.2. Oracle

Oracle est un SGBD édité par la société (Oracle Corporation), leader mondial des bases de données. La société Oracle Corporation a été créée en 1977 par Larry Ellison, Bob Miner, et Ed Oates. Elle s'appelle Relational Software Incorporated (RSI).

FIG. 4.2 — Logo de ORACLE 12c.

Larry Ellison a identifié une opportunité que d'autres entreprises ont manquée : il a pris connaissance de la description d'un prototype opérationnel d'une base de données relationnelle, et découvert qu'aucune entreprise n'avait pris l'initiative de commercialiser cette technologie. Larry Ellison et ses co-fondateurs Bob Miner et Ed Oates ont réalisé que le modèle de la base de données relationnelle représentait un potentiel énorme, mais ils n'imaginaient pas qu'ils transformeraient pour toujours l'informatique professionnelle.

Ce produit est caractérisé par la diversité d'outils de manipulation de données (SQL*plus, SQL plus Worcksheet, iSQL*plus) et de programmation (pro c-c++, pro cobol, pro perl) en basant sur les langages **SQL** et **PL/SQL**.

De plus il procure des assistants graphiques pour aider à la gestion et d'autres pour diagnostiquer les données et d'autres plusieurs outils qui peuvent être optionnels selon l'édition d'oracle.

Il existe trois éditions d'oracle DB utilisées selon les besoins et l'environnement où réside la base de données qui sont : Edition entreprise, édition standard, édition personnel.

4.2.2.1Le serveur Oracle

IL s'agit du **système** installé sur une machine qui va permettre la gestion de toutes les bases de données disponibles sur la machine. Le serveur oracle est basé sur une Architecture MultiServeurs. Le serveur est responsable du traitement de toutes les activités de la base de données tel que l'exécution des instructions SQL, gestion des ressources et utilisateur, et gestion du stockage, Pour consulter les données, l'utilisateur doit tout d'abord se connecter à un Serveur Oracle. Il existe trois types de connexions grâce auxquelles un utilisateur peut accéder à un Serveur Oracle :

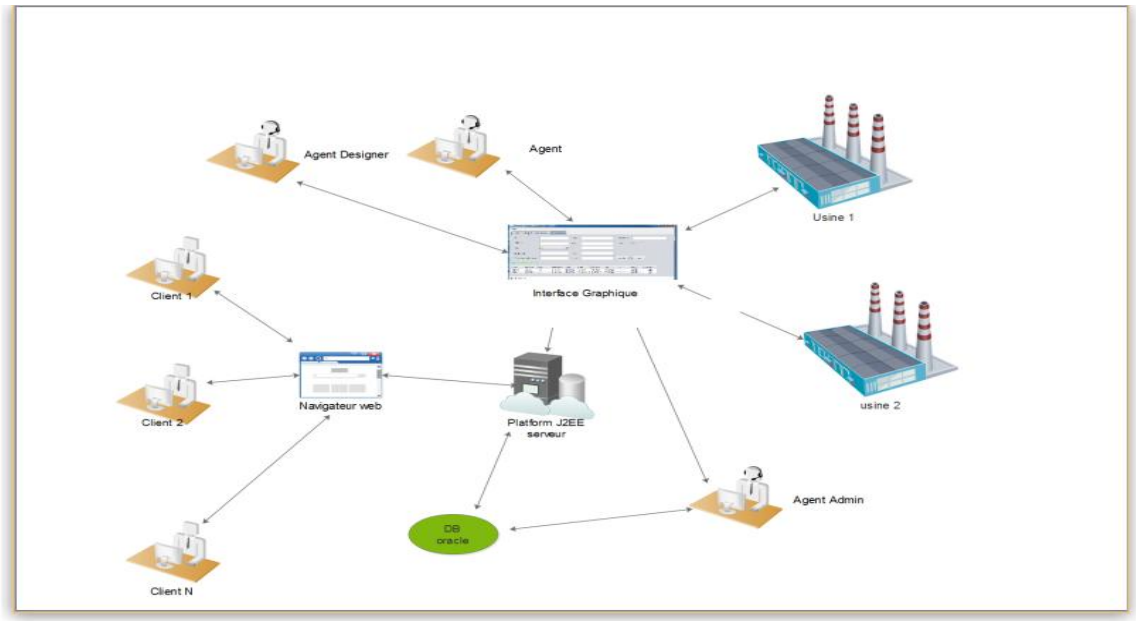
1. Connexion locale.
2. Connexion Deux Tiers.
3. Connexion Multi Tiers.

4.3 Le schéma général de l'application

Notre application se compose de deux parties principales comme le montre la FIG 4.3.

1- La première partie est une application de type J2EE destineé aux clients.

2- La deuxieme partie contient une application de contrôle destinée à l'usage des agents et



à l'administrateur.

4.3.1 La premier partie "Partie Client"

Dans cette partie de l'application, il est nécessaire de distinguer entre la logique de présentation des données aux clients, la logique applicative qui consiste à traiter les données et préparer les résultats et enfin le choix d'un modéle de données. le schéma montré à la figure FIG. 4.4, permettant de distinguer ces différents niveaux, a été retenu.

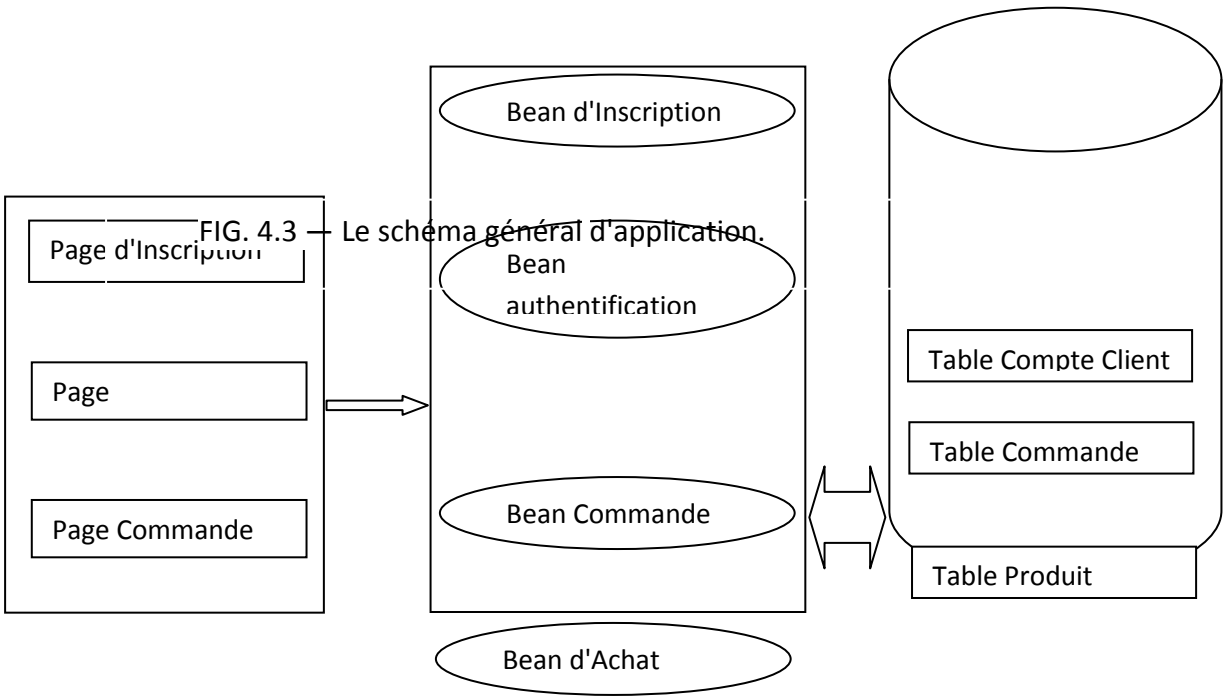


FIG. 4.3 — Le schéma général d'application.

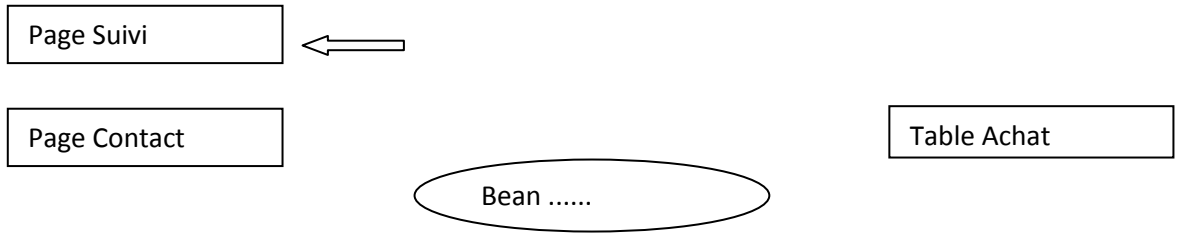


FIG. 4.4 —Représentation générale de l'application partie Client.

Cette partie est construite essentiellement par des JSP pour afficher des pages web et des servlets pour la logique d'application et des beans pour la connection avec le serveur de bases de données.

4.3.1.1 La procédure d'enregistrement de client dans la plate-forme

La page principale de l'interface Web qui s'ouvre dans notre site cad la page **index. jsp** est montrée dans la figure FIG 4.5.

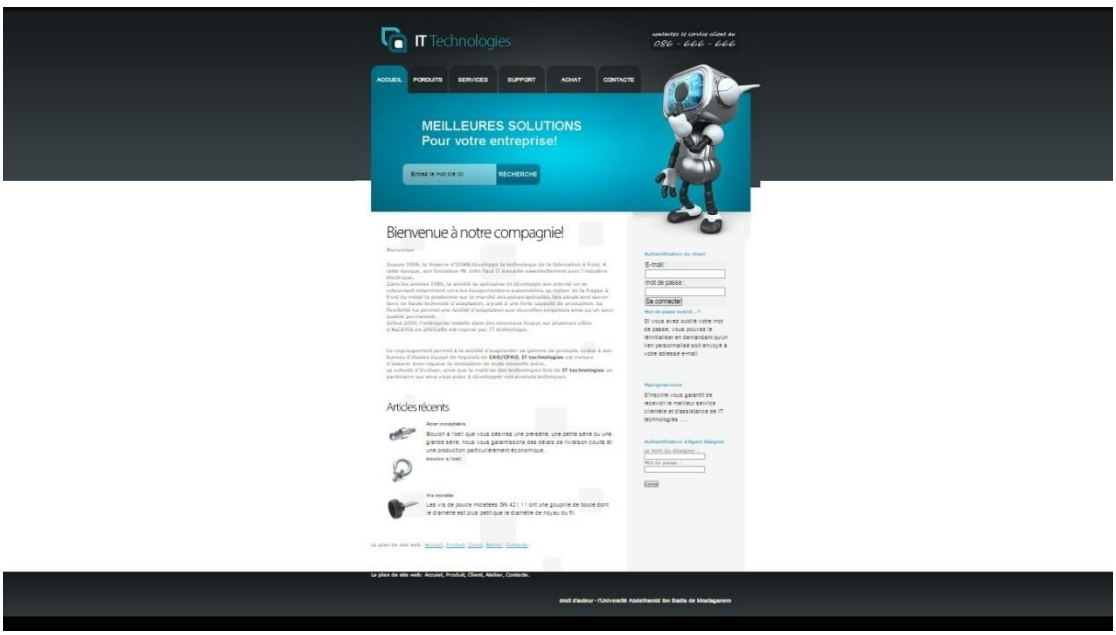


FIG. 4.5— Page Index.jsp.

Pour que le client puisse effectuer des achats et passer des commandes en ligne il doit être enregistré et identifié dans la base des données.

Et, pour que le client puisse s'enregistrer dans notre plate-forme, nous mettons à sa disposition une page **inscrire.jsp**. où se trouve le formulaire d'inscription. Comme cela est visualisé dans la figure 4.6.

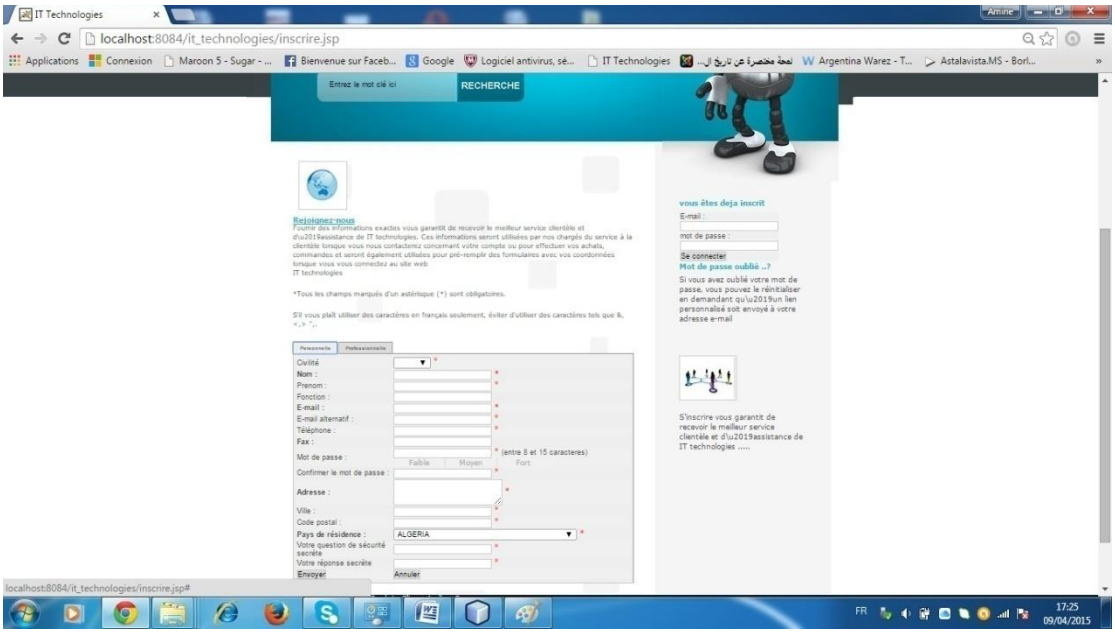


FIG. 4.6— Page inscrire.jsp.

Cette page contient deux formulaires d'inscription, la premiere est destinée aux clients personnels et la deuxième pour les clients professionnels c'est-à-dire des entreprises.

Après le remplissage du formulaire, il existe une série de contrôles sur les champs de saisie. Nous pouvons citer parmi elles :

- Contrôle de HTML .
- Contrôle en JavaScripte.
- Contrôle dans le serveur en "Java".

Premièrement et par défaut, nous utilisons un contrôle HTML dans les balises de formulaire, on cite le type de chaque enregistrement, par exemple on prend

Partie Code HTML

```
<form onsubmit="return Verifchamps()" method= "post" action="TraitementDeDonnees.jsp"
name="f1">

 <tr>

 <td> E-mail :</td>

<td><input type="email" maxlength="30" onkeyup="javascript:couleur(this);"

                              required="required"  id="email" name="email" />

<font color="red"> *</font>

</td>

</tr>
```

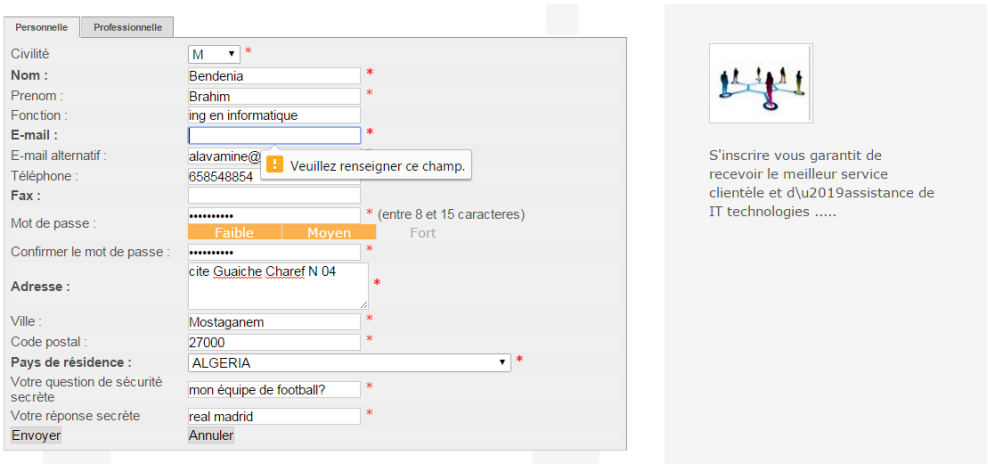


FIG. 4.7— Formulaire d'inscription.

dans la balise <input > on définit le type de champs avec l'option type="email" qui fait un contrôle systématique sur la saisie, l'option required="required" nous oblige à remplir le champ.

Un autre type de contrôle sur les formulaires, qu'on peut voir au début du formulaire on appelle une procedure de contrôle de type JavaScripte onsubmit="return Verifchamps()" ce contrôle est plus efficace que le HTML.

Partie code javaScripte

```
function IsMail(email)
{  // Cette fonction  vérifier la bonne conformité d'une adresse email.
   // Comme : user@domain.com ou user.perso@domain.com
   var i;
   var message="Merci de v\351rifier votre adresse e-mail"

   // Recherche de @
   i = email.indexOf("@");
   if (i == -1) {alert(message); return false;    }

   // Séparation du nom de l'utilisateur et du nom de domaine.
   var username = email.substring(0, i);
   var domain = email.substring(i + 1, email.length)

   // Recherche des espaces au début du nom de l'utilisateur.
   i = 0;
   while((username.substring(i, i + 1)==" ")&&(i < username.length))
  { i++; }
   // Les enlève s'il en trouve.
   if (i > 0) {
      username = username.substring(i, username.length);
   }
   // Recherche d'espaces à la fin du nom de domaine.
   i = domain.length - 1;
   while ((domain.substring(i, i + 1) == " ") && (i >= 0)) {
      i--;
   }
   // Les enlève s'il en trouve.
```

```
if (i < (domain.length - 1)) {
    domain = domain.substring(0, i + 1);
}

// Vérifie que le nom de l'utilisateur et du domaine ne soit pas vide.
if ((username == "") || (domain == "")) {
    alert(message);return false;
}
```

Le contrôle de L' HTML et de javascripte se font au niveau de la machine client. Ce qui le rend accessible et manipulable par le client, avec la connaissance de langage HTML, le client peut désactiver ce contrôle,et on'est obligé de le refaire au niveau du serveur.

Au niveau du serveur la page jsp "TraitementDeDonnées.jsp contrôle les données envoyées par le client. on lui demande la confirmation s'il n'y a pas d'erreur.

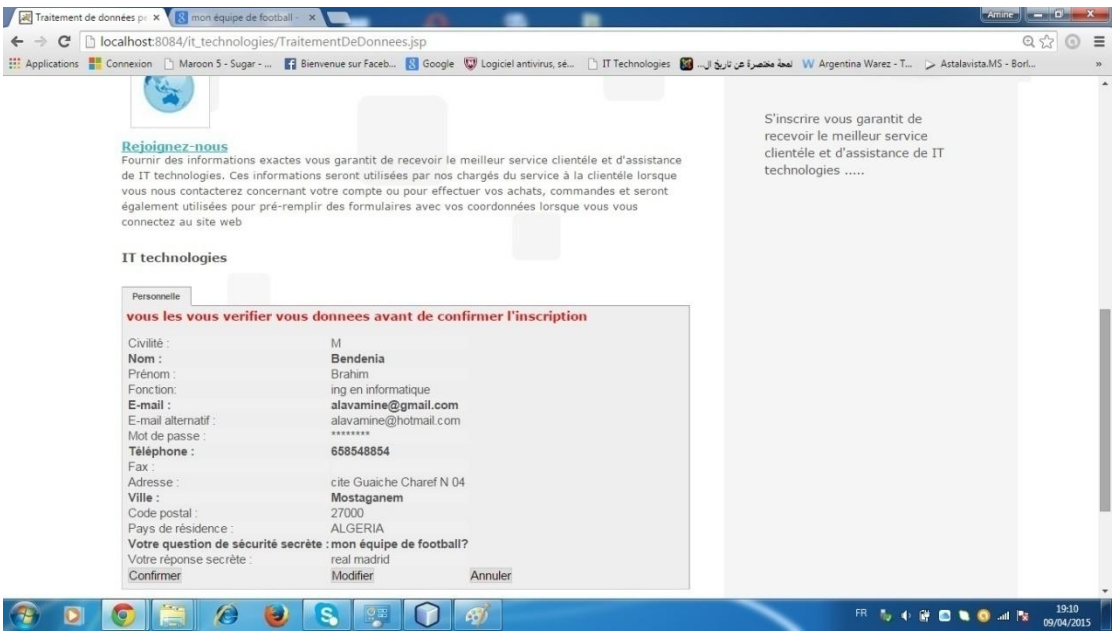


FIG. 4.8 —Page TraitementDeDonnées.jsp.

Le client a la possibilité de faire le choix entre continuer l'inscription, annuler ou bien modifier ces coordonnées .

Après la confirmation de ces coordonnées, le système de sécurité envoi un CAPTCHA au client comme nous le montre la figure FIG 4.9.

A quoi sert un captcha ?

Le *“ captcha ”* est un moyen de validation des formulaires d'inscription présents sur les sites Web. Son but est de différencier un véritable utilisateur d'un programme informatique du reste. Aujourd'hui, la quasi-totalité des sites qui nécessitent une inscription ou la création d'un compte utilisent ce système de validation.

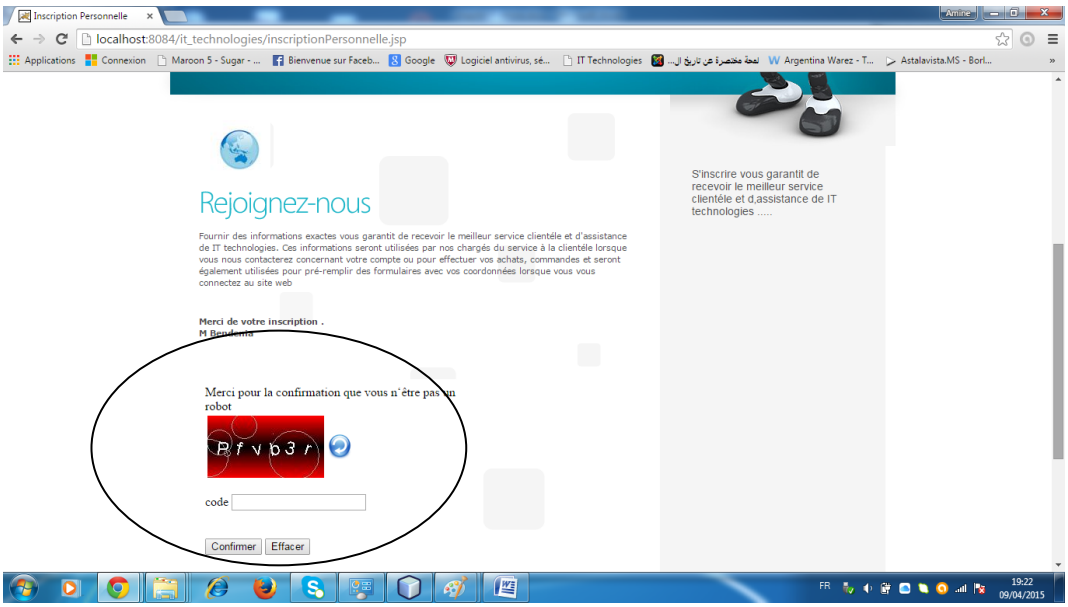


FIG. 4.9 — Captcha.

Après l'envoi du code captcha le systéme s'aperçoit que le client n'est pas une machine "un programme" donc il va consulter la base des données pour l'ajouter dans la table client.

Dans notre cas, nous affichons la réponse suivante .

désole cet email existe déja dans la base des données.

Qui veut dire que ce client est déjà enregistré dans notre plate-forme"IT technologie ", si le client a oublié son mot de passe, il pourra le récupérer en demandant à travers le formulaire saisissant son adresse email et la réponse à sa question de sécurité enregistrée dans le formulaire de l'inscription.

Si la réponse est correcte , le mot de passe sera envoyé à son adresse email.

Bien que les JSP contiennent du code java, elles ne pourront pas acceder directement à la base de données, et donc elles doivent faire appel à un composant faisant partie de la plate forme JEE, c'est le java bean.

Etant donné les beans traitant les accès à la base de données avec des requêtes de données, en utilisant la classe SQL qui se trouve dans la bibliothèque de JDK *java.sql.**

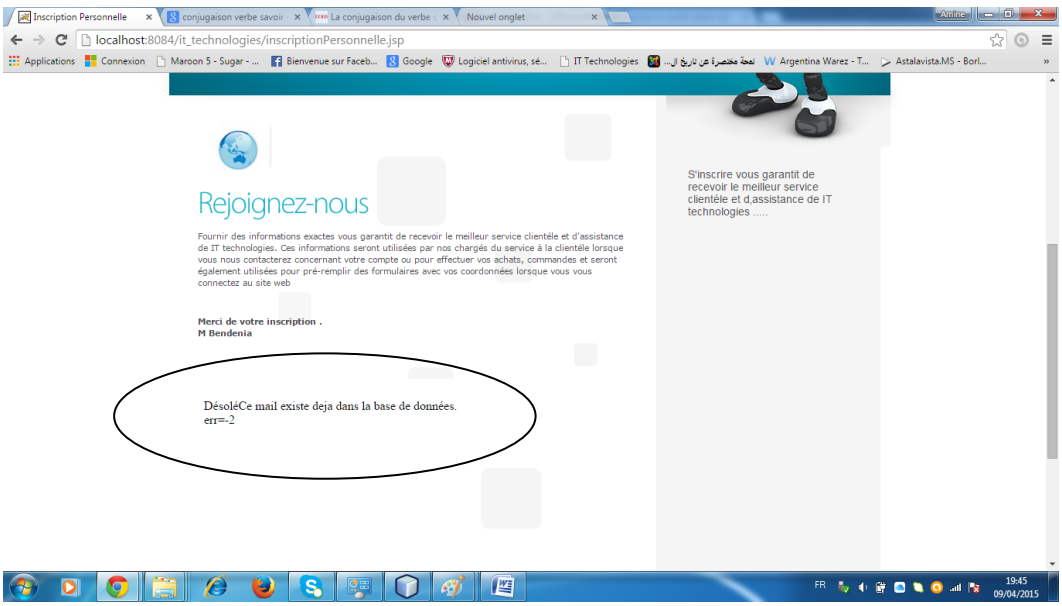


FIG. 4.10 — la réponse du système après la consultation de la base de données.

Par contre si le client s'inscrit pour la première fois, il recevra ce message :

"Pour finaliser votre inscription à IT technologie." Un E-mail de confirmation vient de vous être envoyé. Votre inscription sera termineé après la validation de votre e-mail. Merci"

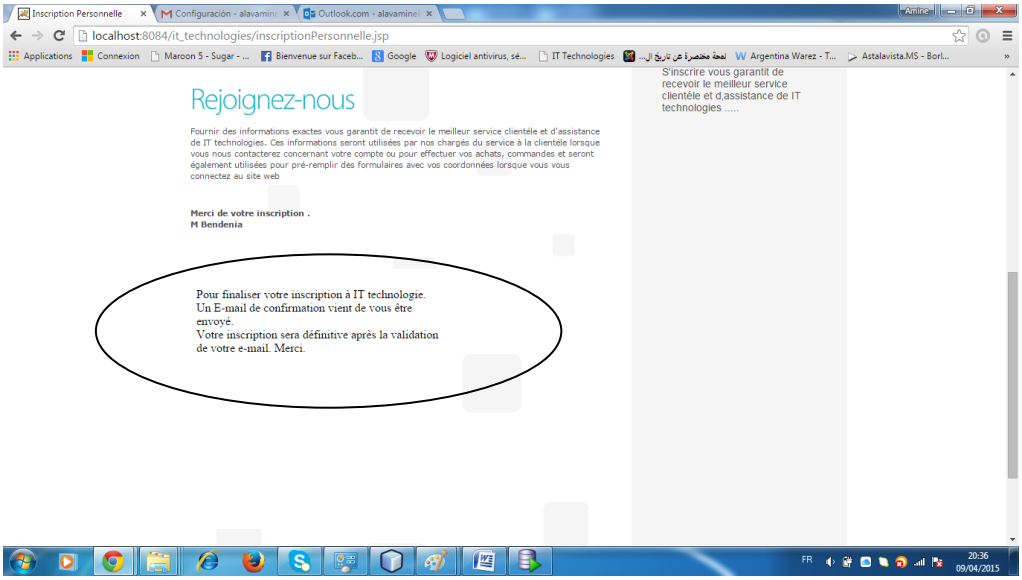


FIG. 4.11— La réponse du système après l'envoi un E-mail de

En plus le client reçoit un E-mail avec le lien d'activation de son compte dans la base de données comme le montre la figure FIG. 4.12

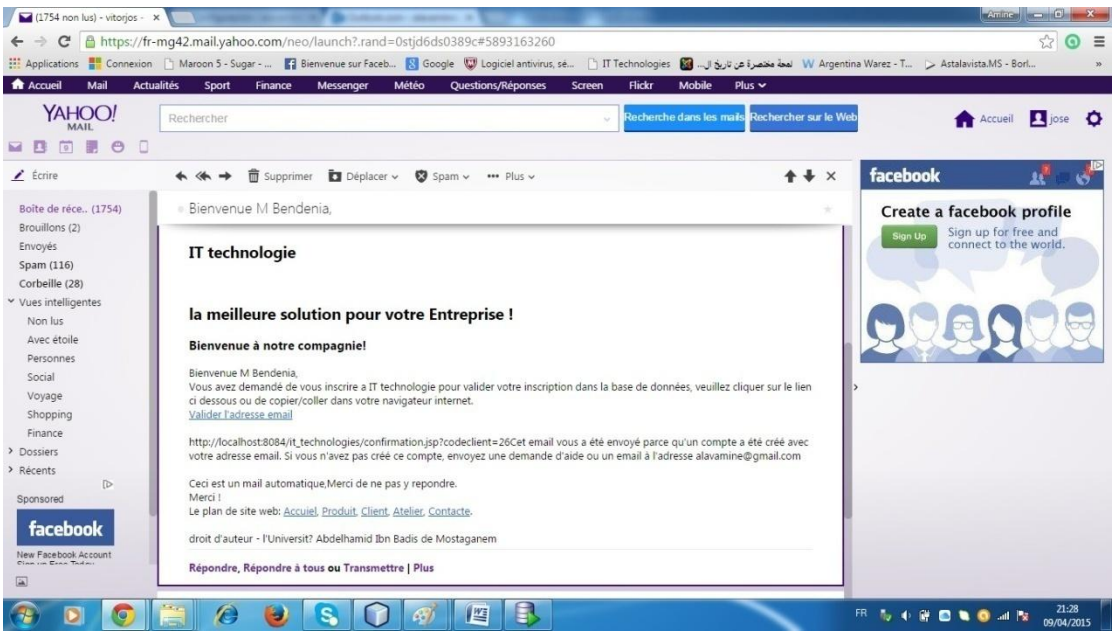


FIG. 4.12 — E-mail de validation reçu dans la boite mail du client.

Aprés la validation d'email, le client peut être connecté à la plate-forme IT- technologie pour effectuer des opérations d'achat ou de commande.

4.3.1.2 Comment passer une commande ?

1 - le client s'identifie avec (email, mot de passe) comme cela est montré à la FIG 4.13

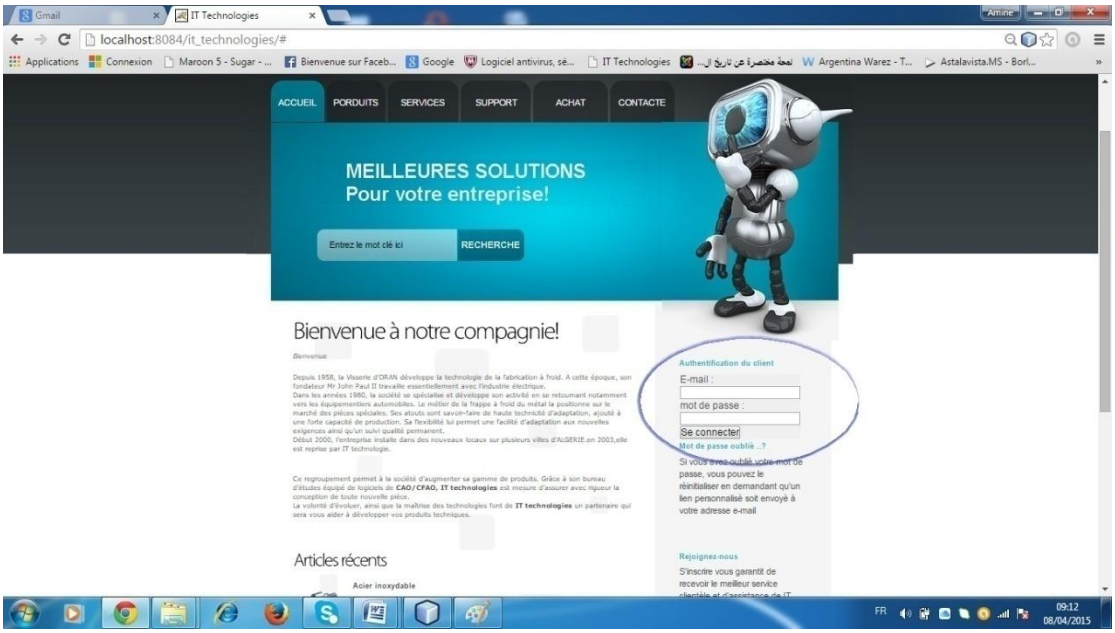


FIG. 4.13 Authentification du Client.

Chaque client enregistre dans la base de données sa propre session, après l'authentification, le système ouvre sa session. alors le client peut effectuer des achats ou des commandes de fabrication de pièces.

2 -Dans la figure FIG 4.14, le client envoie sa demande par le formulaire de la page commande.jsp.

Dans ce formulaire, le client enregistre sa demande de la fabrication de la pièce, on peut trés bien voir que le client peut envoyer son fichier de la conception au serveur qui sera enregistré dans la base de données IT_thec sous Oracle 12c. aprés une serie de contrôle

sur les champs de saisie du formulaire. Le client peut télécharger le fichier de la conception de la pièce. à condition que la taille du fichier soit entre 100 ko et 10 Mo. avec les extensions suivantes *.pdf *.doc * . docx *.xls *.xlsx *.jpg. On peut y ajouter d'autres extensions.

Une fois que la commande est envoyée au serveur, elle sera enregistrée dans la base de données dans la table COMMANDE.Aussi elle sera automatiquement enregistrée avec l'état de traitement" en cours". avec un numéro de Clé.

Le client peut faire la consultation, le suivi ou l'annulation de la commande a n'importe quel moment.

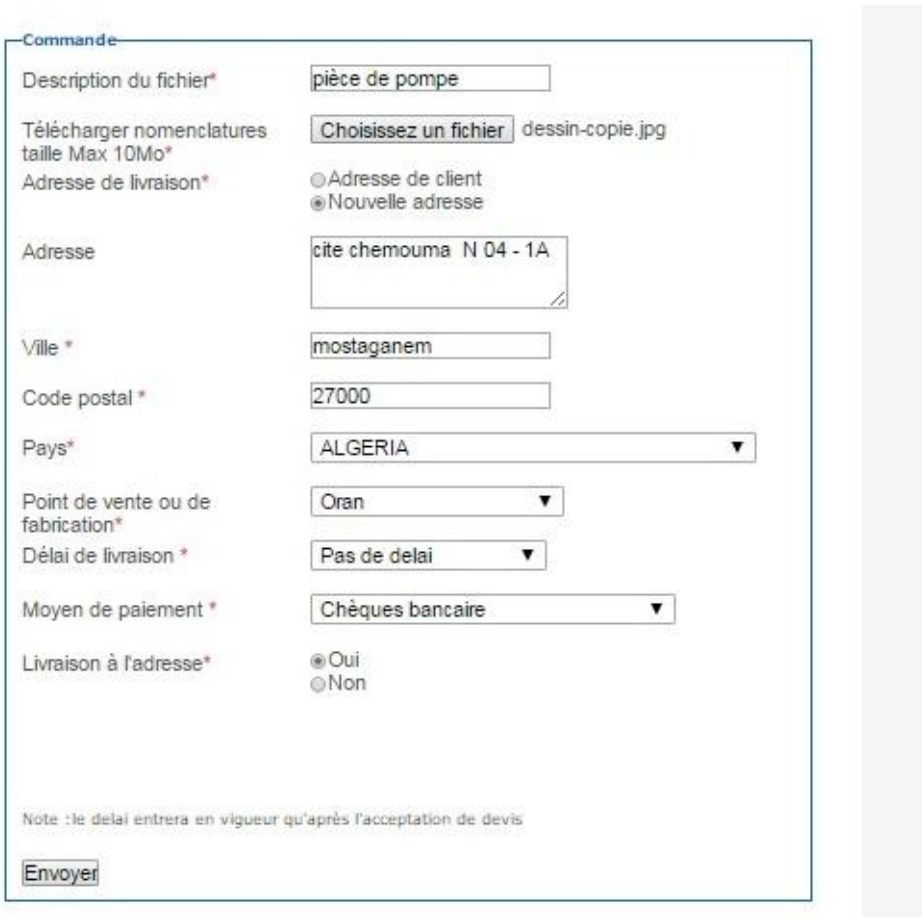


FIG. 4.14—Formulaire Commande.

Une fois que le client click sur le bouton Envoyer, la plate-forme applique une série de contrôle sur les champs envoyé, si tout est bon le client reçois ce message d'enregistrement de sa commande. FIG 4.15

FIG. 4.15—Message du serveur au client.

4.3.1.3 Partie de traitement

Un agent designer fait le suivi de la commande.Il procède à l'étude du fichier de la conception. En cas d'erreur ou d'anomalie dans la conception, L'agent designer envoie un message de refus de la commande à travers son email. Il actualise l'état de la commande dans la table Commande « État erreur ».

FIG. 4.16 — La liste des commandes.

Au cas où la conception de la pièce est correcte et bonne, le designer établit un devis global étudiant le coût (La conception, le transport, ... Etc.). Une fois que le devis est établi, il sera sauvegardé dans la table Devis ; à partir de ce moment,le client peut effectuer le paiement de son devis.

Le client paye sa facture (État de paiement changera de "non" à "oui»). Donc on rentre dans une période de délai si elle est limitée par le client. La pièce où la marchandise sera livrée et la date d'expédition sera datée.

4.3.1.4 L'annulation de la commande

Après l'étude de la commande, un devis sera livré au client, si le client accepte la commande, il doit payer par le moyen qui lui convient. (Carte de crédit, versement bancaire... Etc.). Il peut annuler la commande qui figure sur la liste des commandes.

4.4 La deuxième partie du programme "partie administrateur"

Cette partie de notre programme est destinée à l'usage de l'administrateur et des autres

agents (Designer, Data, Production et Commercial). L'accès a cette application se fait par l'email de l'utilisateur et le mot de passe comme le montre la figure suivante.

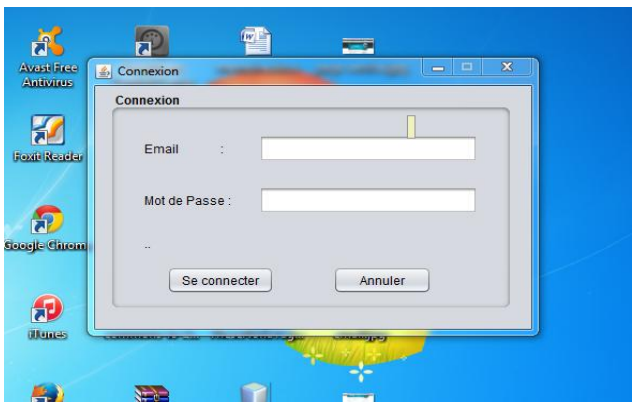
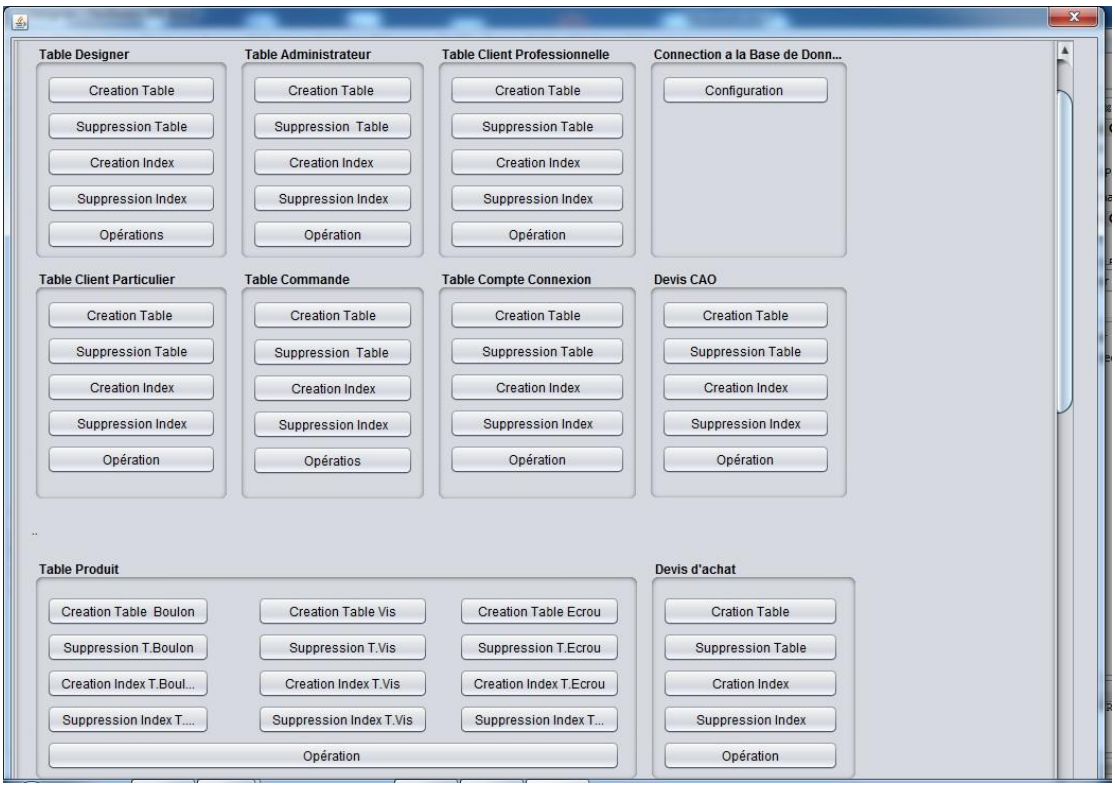


FIG. 4.17 —Formulaire d'acces au programme de contrôle.

Si les coordonnées sont correctes l'administrateur ou l'agent peut se connecter à la table de



contrôle. (voir la figure 4.18).

Dans ce tableau, l'administrateur a le contrôle de toutes les tables de la base de données.

Seul l'administrateur a le pouvoir et le privilège de faire des opérations directes sur la base de données. comme la suppression, ajout de table dans IT-thech, ainsi que les séquences de chaque table "les conteurs" car Oracle gére les séquences à part. Il peut aussi faire configurer la connexion à la base de données à travers la table de configuration.

FIG. 4.18— Table de contrôle.

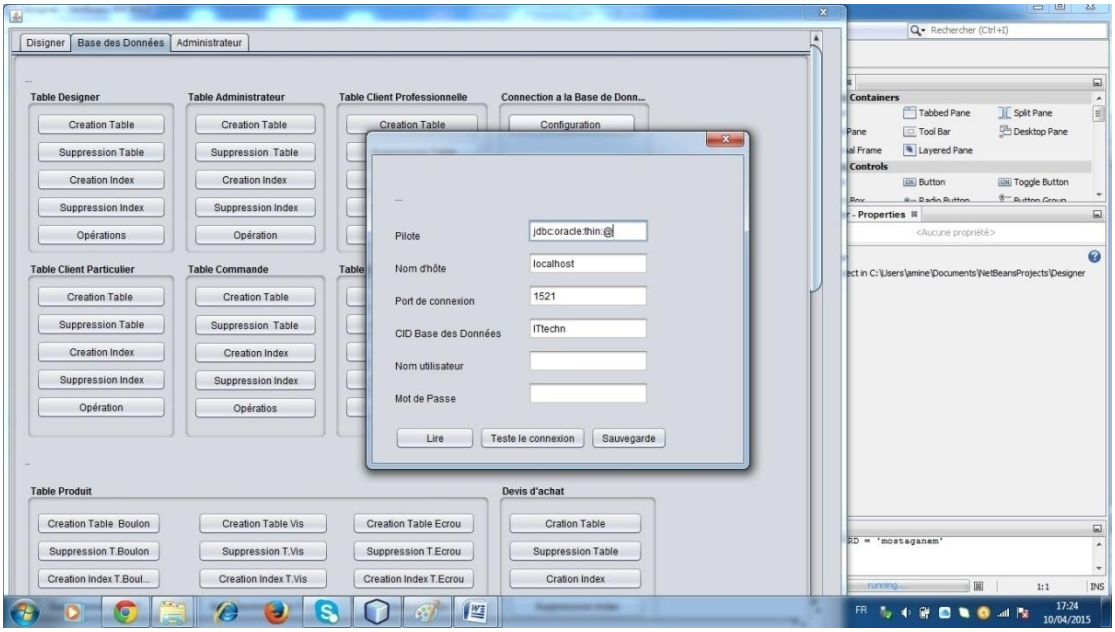


FIG. 4.19 —Tableau de la configuration de connexion à base de données.

4.4.1 Agent de production

L'agent de production peut contrôler la table produit et effectuer les opérations d'ajout, de suppression et de recherche sur les articles, de chaque type de produit. Comme nous le montre la figure suivante.

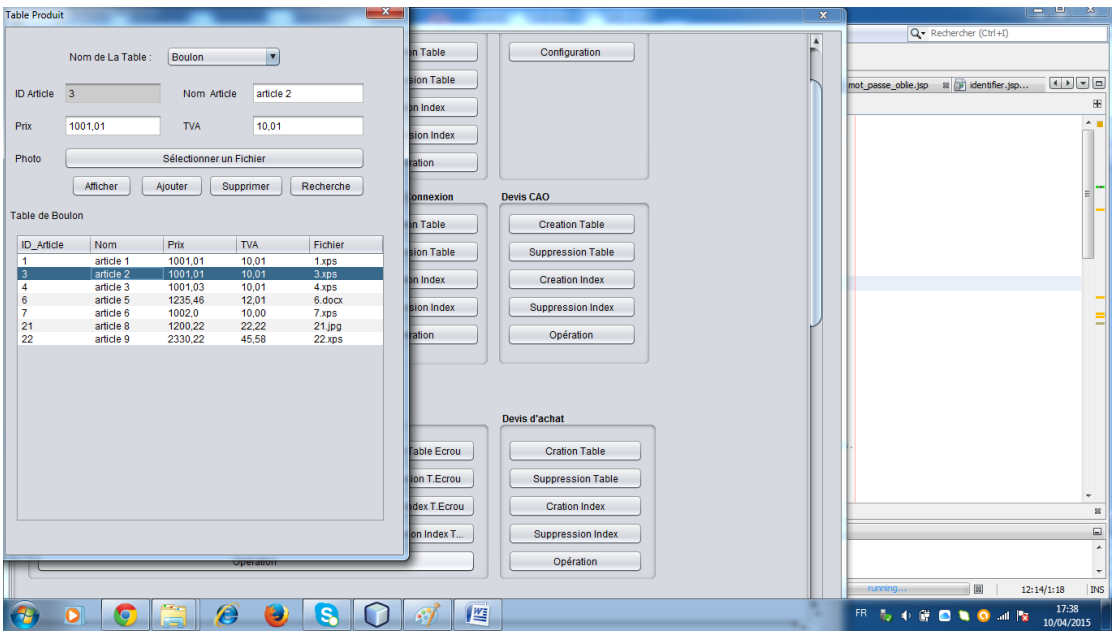


FIG. 4.20— Tableau de contrôle sur les produits.

4.4.2 Agent Designer

Cet agent joue le rôle de contrôle de la conception des fichiers DAO, il peut accéder à la liste des commandes envoyées par les différents clients, qui sont enregistrés dans la table commande pour consulter le plan de la conception DAO de chaque commande. Comme nous le montre les figures suivantes.

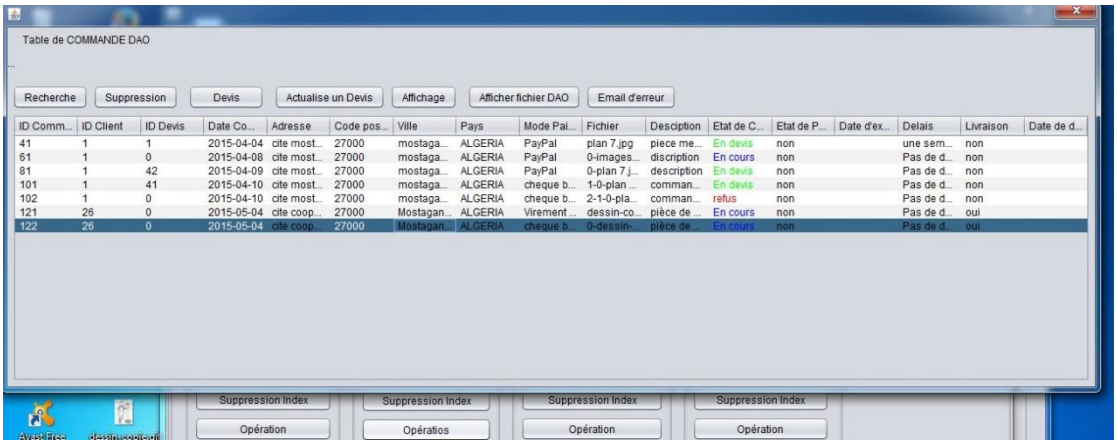


FIG. 4.21 —Table Commande.

On prend par exemple la dernière commande enregistrée par le client numéro 26, l'agent designer peut faire la consultation du fichier DAO en cliquant sur le bouton "Afficher fichier DAO", on obtient la figure suivante.

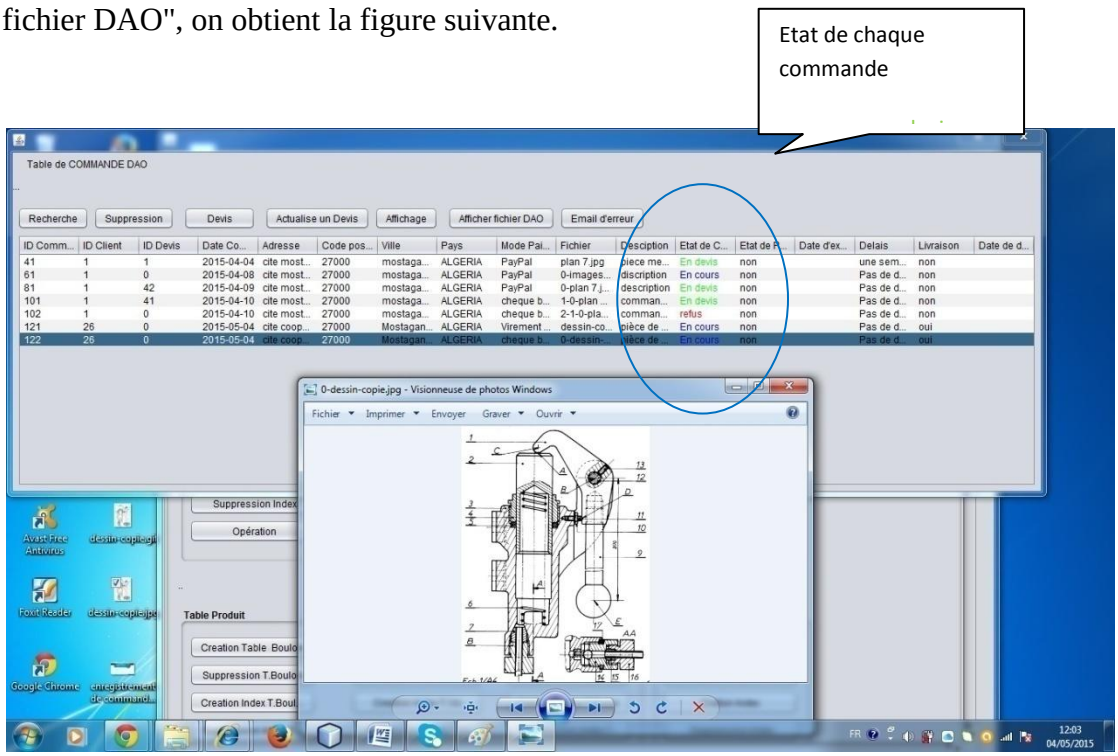


FIG. 4.22—Exemple de consultation de fichier DAO.

Donc l'agent designer consulte le fichier DAO s'il détecte des erreurs il envoie un massage d'erreur à la boite E-mail du client comme nous montre la figure suivante.

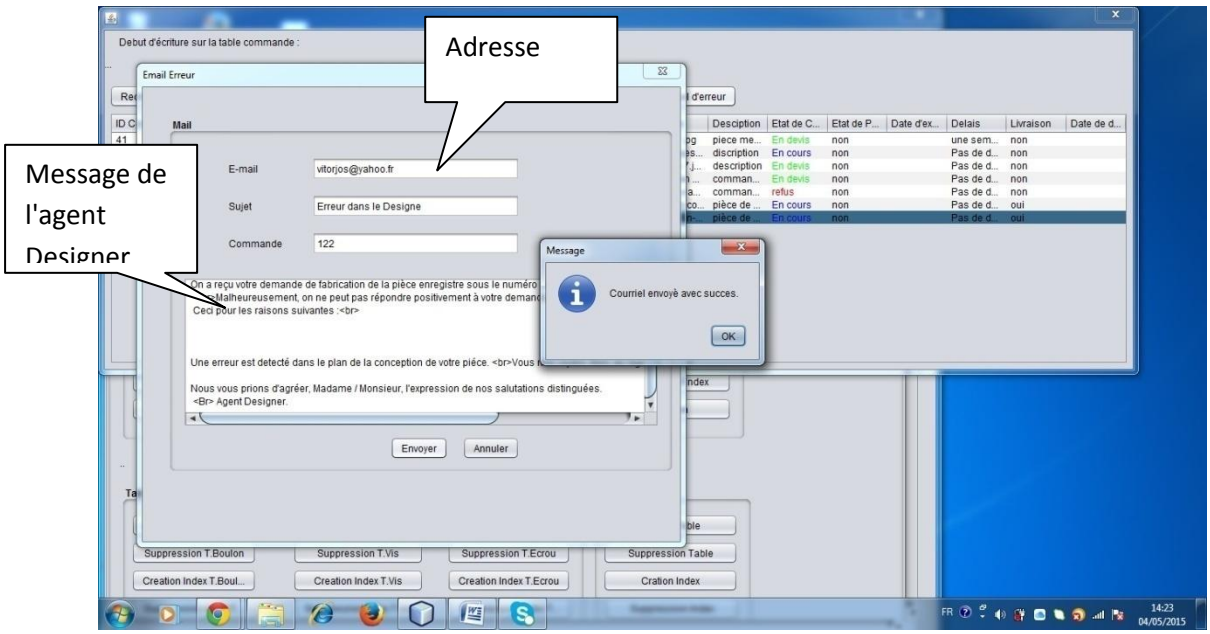


FIG. 4.23 —Agent designer écrit un message d'erreur au client.

Quand le client consulte sa boite E-mail il trouvera le message d'erreur comme la montre la figure FIG. 4.24

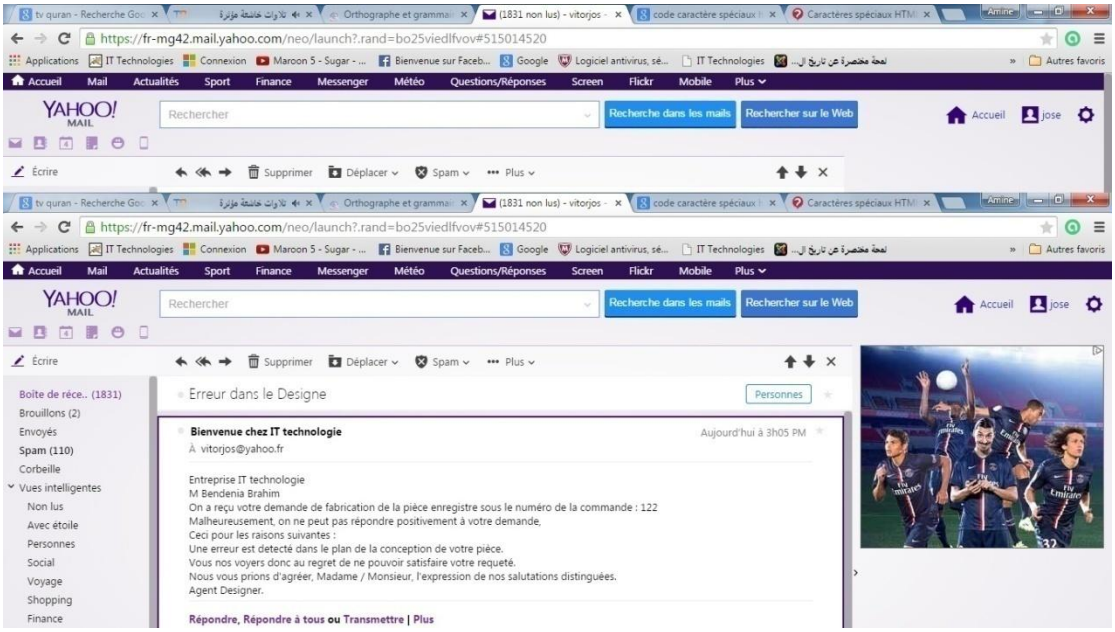


FIG. 4.24 —Message dans la boite E-mail client.

Le message :

Entreprise IT technologie
M Bendenia Brahim
On a reçu votre demande de fabrication de la pièce enregistrée sous le numéro de la commande : 122
Malheureusement, on ne peut pas répondre positivement à votre demande,
Ceci pour les raisons suivantes :
Une erreur est detectée dans le plan de la conception de votre pièce.
Nous avons le regret de ne pouvoir satisfaire votre requête.
Nous vous prions d'agréer, Madame / Monsieur, l'expression de nos salutations distinguées.
Agent Designer.

Dans la suivante figure, l'état de la commande changera d'état "En cours" à l'état "refus". Et le compteur de refus de la commande sera augmente par un +1. Agent designer peut consulter le fichier DAO de la commande jusqu'a quatre fois. À partir de la quatrième fois, la commande sera supprimée définitivement.

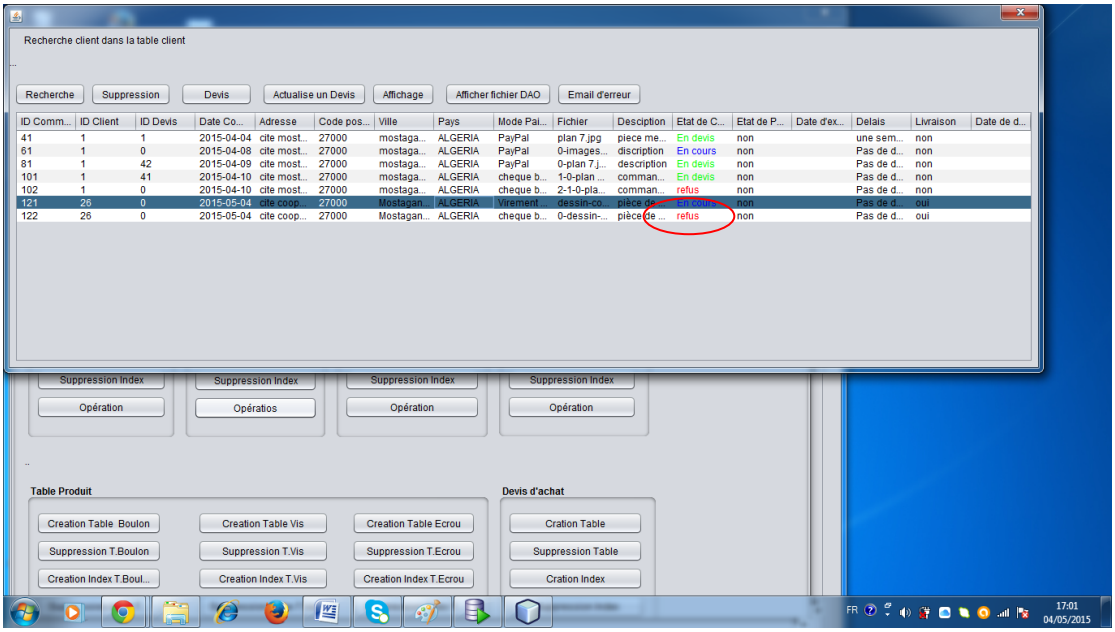


FIG. 4.25 —Changement d'état de la commande.

Par contre s'il n'y pas erreur, et que tout est bon, l'agent crée et génère un devis sous le format Excel à partir du bouton Devis.

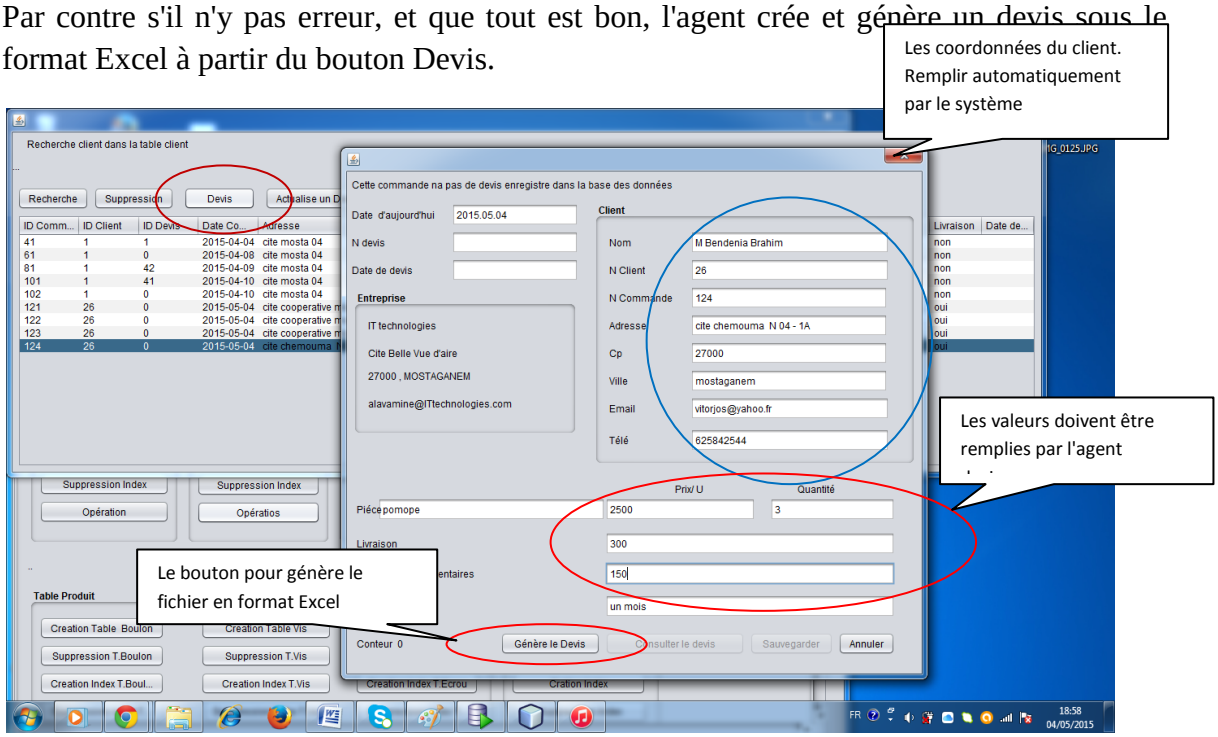


FIG. 4.26 — Génération de devis.

L'agent Designer peut génèrer un fichier temporaire de devis en format Excel, cliquant sur le bouton Génère le devis. À partir de ce moment-là, l'agent peut consulter et ou sauvegarder le devis dans la base de données. Comme nous montre la suivante figure.

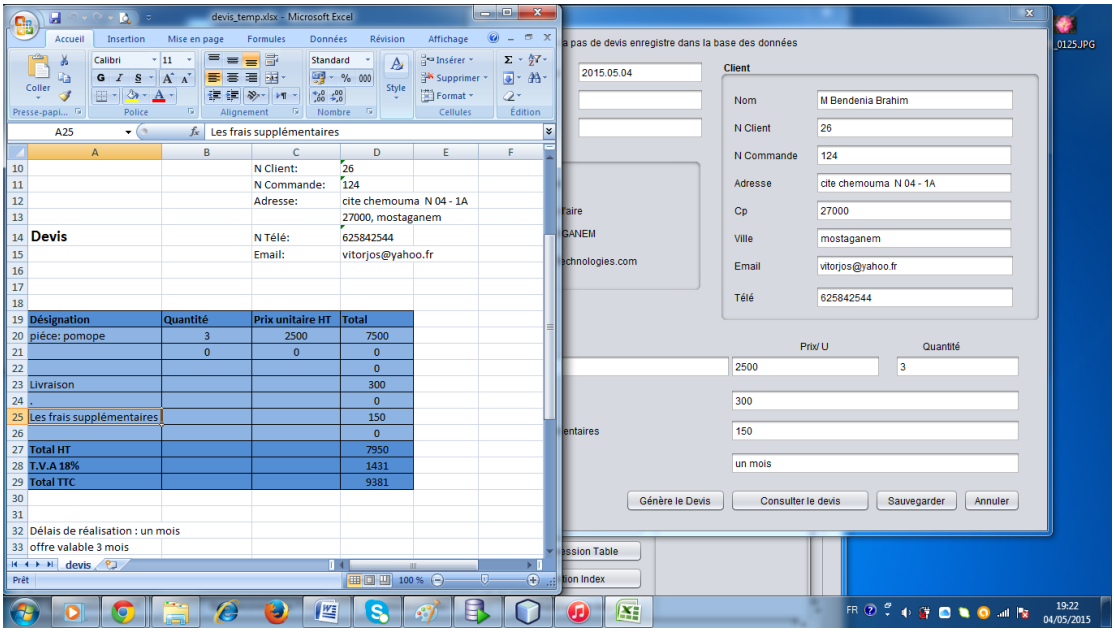


FIG. 4.27 — Génération de devis en format Excel.

Si le devis sera enregistré,le système génère un numéro de devis et l'état de la commande sera "En devis". On obtient ce fichier-là.

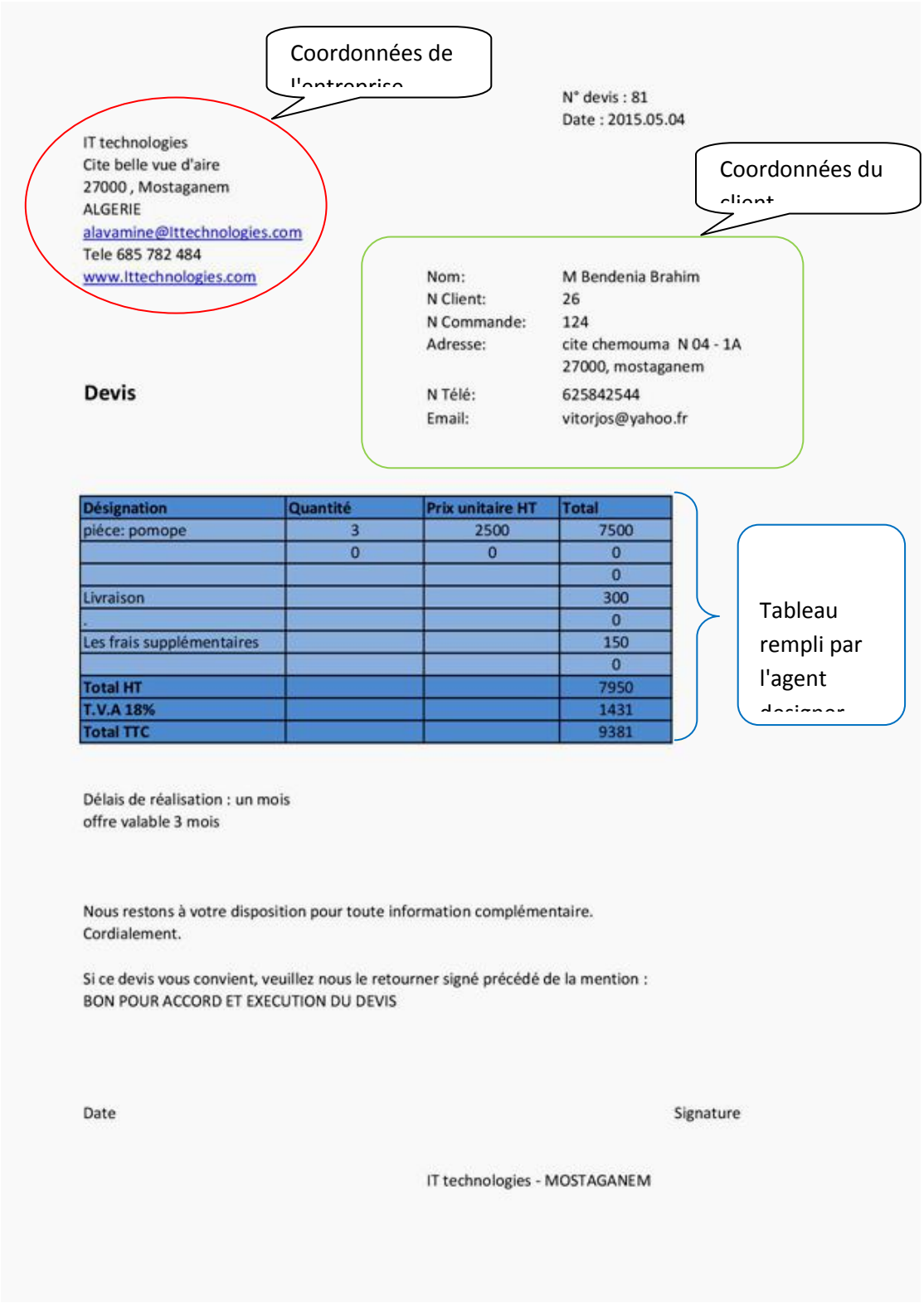


FIG. 4.28 — Le devis en format Excel.

Le client peut consulter le devis de ses commandes dans sa session "***Devis et dèlais de livraison en ligne***"

FIG. 4.29 — Client peut faire le suivi des commandes.

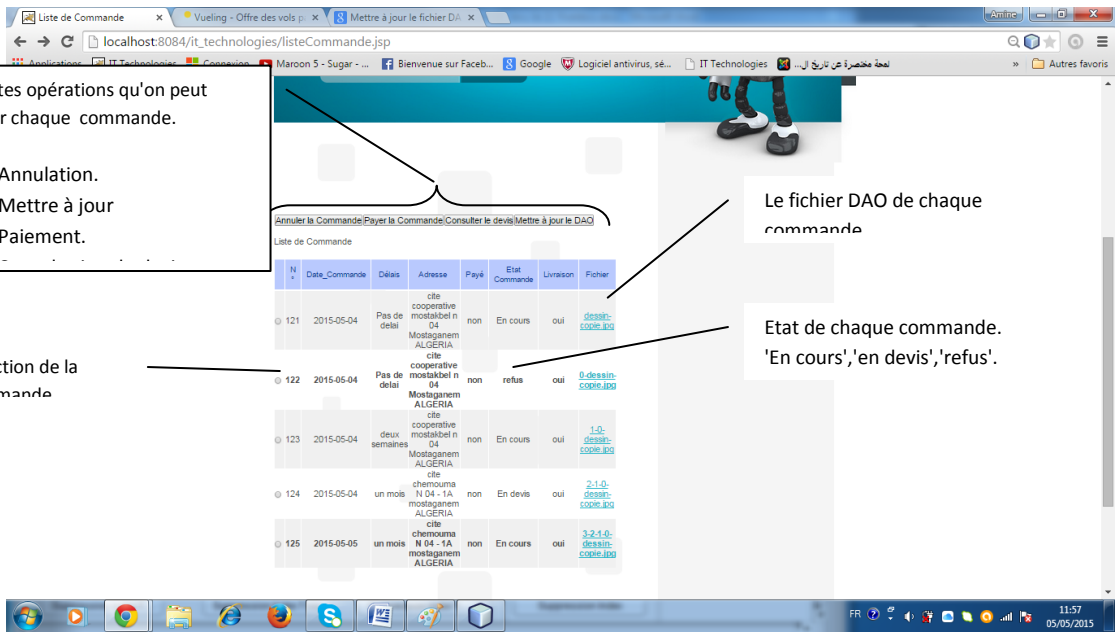


FIG. 4.30 — La liste des commandes de client.

Si le client sélectionne une commande en état "En devis". Il peut consulter le devis qui est en format d'Excel, cliquant le bouton "Consulter le devis" ce que nous montre la figure suivante.

FIG. 4.31 — Devis format Excel.

4.5 Conclusion

Notre travail qui se termine par la conception d'une plate-forme réalisée sous le langage Java a base de la technologie JEE qu'on nomme "IT-technologie". "IT-technologies" permet aux clients de s'enregistrer dans notre base de données et d'effectuer des commandes de conception et de fabrication de pièces mécaniques ou des achats de pièces à partir du magasin en temps réel. Ces commandes seront enregistrées et traitées par différents agents de l'entreprise, qui vont les soumettre à différents contrôles. Chaque commande se terminera par une réponse qui peut être soit un devis soit un refus. Le devis donnera lieu à une facture si le client acceptera ce devis, sinon il sera annulé. Notre plate-forme doit être hébergée dans un serveur applicatif. Ce projet nous a permis d'améliorer et d'utiliser plusieurs bonnes pratiques, ainsi qu'il nous a permis de nous perfectionner en améliorant nos connaissances en programmation et en conception.

Nous avons essayé de réaliser ce projet pour le but de faciliter la gestion de l'entreprise, d'améliorer le suivi des commandes et la vente de pièces. On a applique le maximum possible de règles de base qui' ont permis d'avoir une application performante. Nous avons par ailleurs appliqué l'UML pour concevoir une grande partie de notre travail. On a utilisé Netbeans et Oracle pour implémenter notre plate-forme.

Grâce à l'architecture utilisée (client/serveur) et du fait que Java est un langage adaptable dans plusieurs domaines, notre application peut avoir des extensions ou des modifications dans le futur.

Conclusion générale:

Dans ce mémoire, nous avons développé un modèle qui vise à répondre dans un cadre général aux besoins réels d'une conception assistée par ordinateur (CAO) pour la fabrication des produits en temps réel. Pour répondre aux besoins d'une gestion de production distribuée en temps réel en partant de la conception assistée par ordinateur dans un environnement multi-utilisateurs jusqu'à la validation des prototypes des concepteurs ' designers', une nouvelle approche est proposée dans ce contexte.

En effet, l'outil réalisé permet de combiner entre la technologie WEB et un certain type d'agents tels que l'agent designer et l'agent production et ceci dans l'unique objectif de satisfaire les commandes des clients qu'elles soient en ligne ou en différé par la proposition de leurs propre conception de leurs pièces.

L'incorporation d'agent dans un système en général lui offre l'opportunité de participer directement à l'implémentation des décisions.

Les différents agents ont plusieurs fonctionnalités comme par exemple, l'agent designer, l'agent d'évaluation, l'agent des ressources, l'agent de planification et l'agent coordinateur. Tous ces agents interagissent de manière cohérente pour réaliser la production et l'acheminement des pièces.

L'architecture centralisée de notre système a nécessité la mise en place d'un agent chargé de la coordination, ce dernier peut contrôler toutes les interactions et prendre des décisions efficaces et rapides pouvant rendre le système plus flexible. En outre, toute perturbation ou anomalie d'un agent n'affecte pas les fonctions des autres agents.

Comme perspectives de recherche, de nouvelles fonctions peuvent être ajoutées à notre système, nous pouvons citer :

- Proposition d'une extension permettant une architecture totalement hybride pour pallier aux inconvénients de la gestion centralisée.

- Intégration des agents mobiles dans notre système.

- Traitement des redondances de certains traitements.

Références bibliographiques:

[Afnor, 1988] Afnor. "Concepts fondamentaux de la gestion de production" . Technical Report X50. 310, Agence française de normalisation, 1988.

[Anne, 2002] Anne Tasso Eyrolles. "Le livre de java premier langage", 2002.

[Billaut , 2005] J-C. Billaut et A. Moukrim. " Introduction à la flexibilité et à la robustesse en ordonnancement dans Flexibilité et robustesse en ordonnancement (Billaut) ". Sciences publications Lavoisier Hermes. Paris, 2005.

[Billaut et Moukrim, 2005] J-C. Billaut et A. Moukrim. " Introduction à la flexibilité et à la robustesse en ordonnancement dans Flexibilité et robustesse en ordonnancement (Billaut) ". Sciences publications Lavoisier Hermes. Paris, 2005.

[Courtois ,1995] A.Courtois, M. Pillet, et C. Martin. "Gestion de Production". Editions d'organisation, 2ème édition, 1995.

[Dagoumau, 2000] N. Dagoumau. "Contribution à la gestion des modes des systèmes automatisés de production". Thèse de doctorat, Université des sciences et technologies de Lille, 2000.

[Dauzère-Pérès et al., 2005] S.Dauzère-Pérès, P. Castagliola et C.Lahlou. "Niveau de service en ordonnancement dans Flexibilité et robustesse en ordonnancement (Billaut) ".Sciences publications Lavoisier Hermes. Paris, 2005.

[Duane et Mark, 2000] Duane K. Fields, Mark A. Kolb." JSP - JavaServer Pages", 2000.

[David et al, 2002] David Jordan, Craig Russell, Dr. Rick Cattell "JAVA DATA OBJECTS",2002.

[Esquirol et Lopez, 1999] P.Esquirol et P.Lopez. "L'ordonnancement". Ed.ECONOMICA, Collection GESTION. Série : Production et techniques quantitatives appliquées à la gestion, 1999.

[Frédéric et al, 2001] Frédéric Berqué , Serge rezefond , Ludovic Sorriaux, **"Java-XML et Oracle**", 2001.

[Innal et Dutuit, 2006] F. Innal, Y. Dutuit : "Evaluation de la performance d'un système de production et des contributions individuelles de ses unités constituves". *6 Conférence Francophone de MOdélisation et SIMulation - MOSIM'06 - du 3 au 5 avril 2006 – Rabat – Maroc « Modélisation, Optimisation et Simulation des Systèmes : Défis et Opportunités », 2006.*

[Liu et Young, 2004] S.Liu et R.I.M. Young, "Utilizing information and knowledge models to support global manufacturing co-ordination decisions", Int. J. of Computer Integrated Manufacturing, Vol. 17, pp.479–492, 2004.

[Lopez et al. , 1996] P. Lopez, L. Haudot, P. Esquirol et M.Sicard. "Conception d'un Système Coopératif en ordonnancement de production. Une approche pluridisciplinaire". Dans procedding 5eme Congrès International de Génie Industriel (GI'5), Grenoble, 2-4 avril 1996.

[Lopez et Roubellat, 2001] P.Lopez et F. Roubellat. "Ordonnancement de la production". Ed. Hermes Science, 2001.

[Marty et Larry ,2000] Marty Hall , Larry Brown " Core Servlets and Javaserver Pages: Core Technologies", 2000.

[Nfaoui et al., 2006] H.Nfaoui, Y.Ozrout SCM, AUML : "un modèle d'agents pour la simulation proactive et l'aide à la décision dans le Supply Chain", Conférence MajecSTIC 2006 MAnifestation des JEunes Chercheurs en Sciences et Technologies de l´Information et de la Communication., Lorient France, 22-24 Novembre 2006.

[Parunak, 1991] Cité dans [Trentesaux, 2000].

[Richard ,2000] Richard Monson-Haefel,"Enterprise JavaBeans 1.1", 2000.

[Sikora et Shaw, 1998] R. Sikora, et M.J. Shaw, "A multi-agent framework for the coordination and integration of information systems", Management Science, Vol. 44, pp.65–78, 1998.

[Taghezout , 2011] N.Taghezout. Conception et Développement d'un système multi-agent d'Aide à la Décision pour la gestion de production dynamique. thèse de doctorat soutenu à L'université Paul Sabatier de Toulouse e, juillet 2011.

[Tahon, 2003] C.Tahon. "Evaluation des performances des systèmes de production". Editions Lavoisier, 2003.

[Thierry et al, 2000] Thierry Brethes, Francois Hisquin, Pierre Pezziardi, "Serveurs d'applications",2000.

[Trentesaux, 1996] D. Trentesaux. "Conception d'un système de pilotage distribué, supervisé et multicritère pour les systèmes automatisés de production". Thèse de Doctorat en Sciences, Institut National Polytechnique de Grenoble, 1996.

[Trentesaux, 2002] D. Trentesaux. "Pilotage hétérarchique des systèmes de production".

Thèse d'habilitation. Université de Valenciennes et du Hainaut-Cambrésis (UVHC),

2002.

[Vacher, 2000] J. Vacher. "Un système adaptatif par agents avec utilisation des

algorithmes génétiques multi-objectifs : Application à l'ordonnancement d'atelier de type job-shop N *M". Thèse de Doctorat. Université du HAVRE, 2000.

Référence web graphique:

[Site1][http::://perso.enstimac.fr/fontanil/THESE/5_partie1_p13_43.pdf].

[Site2][http://www.commentcamarche.net/contents/230-css-feuilles-de-style].

[Site3][http://www.commentcamarche.net/contents/144-cgi-introduction-a-la-programmation des-cgi].
[Site4][http://www.coursdinfo.fr].
[**Site5]**[http://www.ifadem.org].
[**Site6]**[http://www.eyrolles.com].
[**Site7]**[https://aresu.dsi.cnrs.fr/IMG/pdf/failles_de_securite_v1-3.pdf.].
[Site8][http://fr.wikipedia.org/wiki/NetBeans].