



Global Knowledge.



Support de cours

Table des matières

CHAPITRE I INTRODUCTION AU SYSTEME D'INFORMATION.....	7
1. LE SYSTEME D'INFORMATION	8
2. L'INFORMATIQUE	10
3. LA DIRECTION DES SYSTEMES D'INFORMATION.....	11
CHAPITRE II L'INFRASTRUCTURE	13
1. DEFINITION	14
2. LE DATA CENTER.....	14
2.1. Présentation	14
2.2. Equipements physiques du data center.....	15
2.3. Hébergement et Tier.....	16
3. LE SYSTÈME D'EXPLOITATION	17
3.1. Présentation	17
3.2. Les différents types de système d'exploitation	18
4. RESEAU	20
4.1. Les différents types de réseau	20
4.2. Les protocoles réseau	22
4.3. Les équipements réseaux	25
5. SYSTEME ET RESEAU	27
6. LA VIRTUALISATION	28
6.1. Objectifs	28
6.2. Notions prises en charge.....	28
6.3. Les enjeux de la virtualisation	30

CHAPITRE III LES ARCHITECTURES DISTRIBUEES	31
1. EVOLUTION DES ARCHITECTURES	32
1.1. Avant les années 90	32
1.2. Après les années 90	33
1.3. Après les années 2000	33
2. L'INTEROPERABILITE ET PORTABILITE	34
2.1. Définition et enjeux	34
2.2. Normes et standards	35
3. LES ARCHITECTURES DISTRIBUEES	36
4. LE MODELE DU GARTNER GROUP	37
4.1. Client – Serveur de première génération :	38
4.2. Client – Serveur de deuxième génération	41
4.3. Les évolutions du client - Serveur	44
5. L'ARCHITECTURE WEB	46
5.1. Présentation d'Internet	46
5.2. Composants architecturaux du Web	47
5.3. Autour du Web	50
6. ARCHITECTURE N-TIERS	53
6.1. Présentation	53
6.2. Les enjeux	54
7. ARCHITECTURES ORIENTEES SERVICES (SOA)	55
7.1. Présentation	55
7.2. Les enjeux	56
8. LE CLOUD COMPUTING	57
8.1. Présentation	57
8.2. Les enjeux	58
CHAPITRE IV LES COMPOSANTS D'UNE APPLICATION	59

1.	LE MODELE EN COUCHE	60
1.1.	Présentation	60
1.2.	Les principaux modèles.....	61
2.	LA COUCHE DONNEES.....	64
2.1.	Concepts de base	64
2.2.	Les systèmes de gestion de fichiers.....	65
2.3.	Les systèmes de gestion de bases de données (SGBD) et le langage SQL....	66
2.4.	Des bases relationnelles vers la Business Intelligence	69
2.5.	De la multitude des données vers le MDM	69
2.6.	Le Big Data.....	70
3.	LA COUCHE PRESENTATION.....	71
3.1.	Les interfaces mode caractère	71
3.2.	Les interfaces mode graphique	72
3.3.	Les interfaces Web.....	73
3.4.	Les interfaces riches.....	74
3.5.	Les interfaces mobiles.....	75
3.6.	Le portail.....	76
4.	LA COUCHE TRAITEMENT	77
4.1.	Les langages de programmation	77
4.2.	Les générations de langage	78
4.3.	Les langages d'échanges de données	80
4.4.	Les langages du Web.....	82
4.5.	Les outils de développement.....	87
5.	LES PLATES-FORMES JAVA EE ET .NET	88
5.1.	Présentation	88
5.2.	Les outils	90
6.	LES FRAMEWORKS	91
	 CHAPITRE V LE PROJET INFORMATIQUE.....	 93

1. DEFINITIONS ET OBJECTIF D'UN PROJET	95
2. MAITRISE D'OUVRAGE / MAITRISE D'ŒUVRE	96
2.1. Définitions.....	96
2.2. Les différents types de responsabilités dans le projet	97
2.3. Les démarches.....	98
2.4. Les différentes phases d'un projet.....	99
2.5. Détail des phases.....	99
3. CYCLES DE DEVELOPPEMENT	104
3.1. Cycle linéaire	104
3.2. Cycles itératifs	106
4. LES MÉTHODES DE CONCEPTION.....	110
4.1. Présentation	110
4.2. Les différentes générations de méthodes.....	110
4.3. Les méthodes systémiques - Merise	111
4.4. Les méthodes orientées objet - UML.....	116
5. LE PROCESSUS UNIFIE	120
6. DE L'UP VERS LE RUP ET LES METHODES AGILES.....	122
7. QUELQUES CHIFFRES CONCERNANT LES PROJETS.....	123
8. REFERENTIELS DE QUALITES	124
8.1. Présentation	124
8.2. Périmètres des dispositifs.....	125
8.3. ITIL - Information Technology Infrastructure Library.....	126
8.4. CMMI - Capability Maturity Model Integration.....	129
CHAPITRE VI POUR CONCLURE	131
1. UN PEU D'HUMOUR.....	132
2. POUR EN SAVOIR PLUS	133

2775cbe5ce
Global Knowledge

CHAPITRE I

INTRODUCTION AU SYSTEME D'INFORMATION

Objectif : situer le cadre de la formation et définir le SI Système d'Information, l'informatique, la DSI Direction des SI.

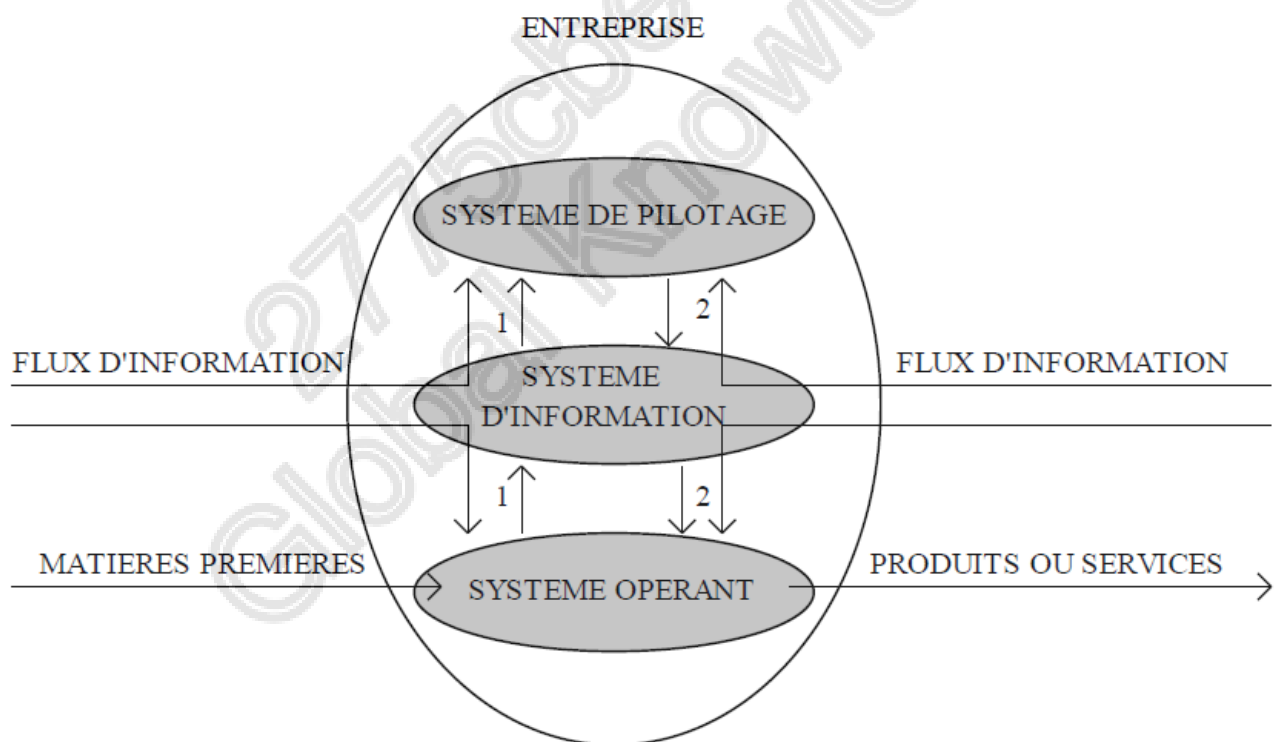
1. LE SYSTEME D'INFORMATION

L'entreprise peut être représentée par un ensemble de trois systèmes :

- un système opérant (ou logistique),
- un système de pilotage (ou de direction, de gestion, de décision),
- un système d'information.

Le système d'information peut être lui-même représenté comme un ensemble comprenant :

- une base d'information,
- un processus d'information composé de deux moniteurs, l'un qui gère le stockage des données, l'autre qui gère la circulation de l'information.



LEGENDE :

1 : RESULTATS 2 : REGLES

Le système d'information est le véhicule de la communication dans l'entreprise. Il possède un langage dont les mots sont les données. Il peut être décrit au moyen de cinq composantes :

- une **composante humaine informelle**, qui est celle de l'univers du discours, c'est-à-dire celle des communications informelles entre individus,
- une **composante humaine formelle** qui est celle des communications formelles imposées par l'organisation et les règles qui en découlent,
- une **composante informatique formelle** qui est la traduction informatisée de la partie informatisable du système d'information,
- une **composante informatique informelle** qui est celle de l'informatique individuelle ou plutôt indépendante et qui n'observe pas de règles précises au sein de l'organisation,
- une **composante externe** à l'entreprise qui est celle constituée par l'environnement et qui peut être formelle et informelle.

Le système informatisé n'est donc qu'une composante du système d'information. Socrate (SNCF), les distributeurs automatiques de billets (DAB) et autres guichets automatisés bancaires (GAB), les applications internet grand public ne sont donc que des applications informatisées d'un système d'information d'entreprise beaucoup plus large.

2. L'INFORMATIQUE

« Science de l'information ».

« Ensemble des techniques de la **collecte**, du **tri**, de la **mise en mémoire**, de la **transmission** et de **l'utilisation** des informations traitées automatiquement à l'aide de programmes mis en œuvre sur des ordinateurs »

L'informatique nécessite

- des hommes,
- des machines (hardware),
- des logiciels (software),
- des méthodes.

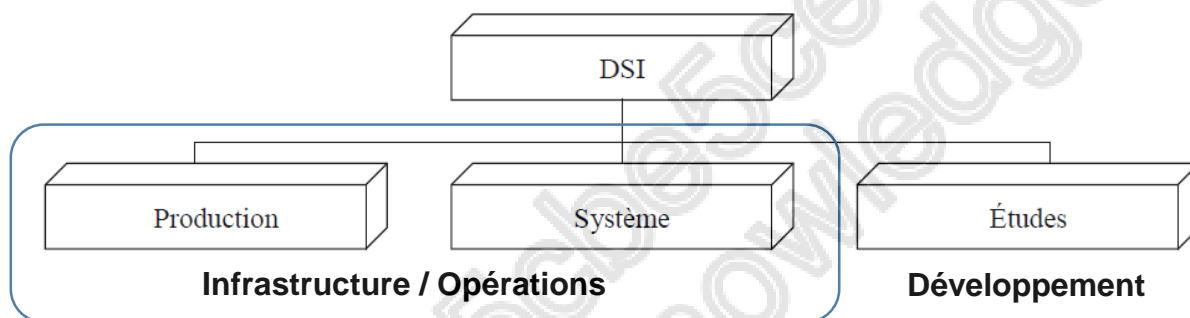
L'informatique intervient dans les domaines :

- de la gestion (comptable, commerciale, de production...),
- des télécommunications et réseaux,
- industriel (robotique...),
- scientifique (simulation...).

3. LA DIRECTION DES SYSTEMES D'INFORMATION

La Direction des Systèmes d'Information (DSI), remplit 3 grands rôles, correspondant, dans la majorité des entreprises, à 3 services différents :

- ⇒ la **Production** (ou **Exploitation**), en charge du bon fonctionnement quotidien du système d'information, tant d'un point de vue matériel qu'applicatif,
- ⇒ le **Système**, dont le rôle est de mettre en œuvre les évolutions nécessaires afférentes au matériel et aux logiciels systèmes, mais aussi de résoudre les pannes et les dysfonctionnements pouvant survenir, en relation avec les fournisseurs concernés,
- ⇒ les **Études** (ou Développement), en charge du développement des nouveaux applicatifs et de la maintenance des applicatifs existants.



- ⇒ Production et Système, peuvent former un seul service, que l'on nomme Infrastructure, ou les « opérations » (abrégées en ops) mais il existe toujours une **distinction entre la gestion du matériel**, du réseau et du fonctionnement des machines **et la mise en œuvre d'applications** pour les utilisateurs du SI. Que l'on explique par des objectifs différents de ces services. Le but des opérations est de garantir un système stable et fiable : la qualité est primordiale au détriment des coûts et du temps. A l'inverse côté développement, l'objectif est d'apporter des changements à moindre coûts et rapidement souvent au détriment de la qualité.
- ⇒ Fin 2009, le mouvement **devops** est né, visant à l'alignement de l'ensemble des équipes du SI, composé par l'association des "dev" ou "Dev engineers", « développeurs » chargés de faire évoluer le SI et les "ops" ou "Ops engineers", « opérations » chargés d'exploiter les applications existantes. L'idée étant : travaillons ensemble pour produire de la valeur pour l'entreprise.

CHAPITRE II

L'INFRASTRUCTURE

Objectif : Présenter les concepts des data centers, des systèmes d'exploitation, des réseaux et de la virtualisation pour comprendre le fonctionnement de l'infrastructure.

1. DEFINITION

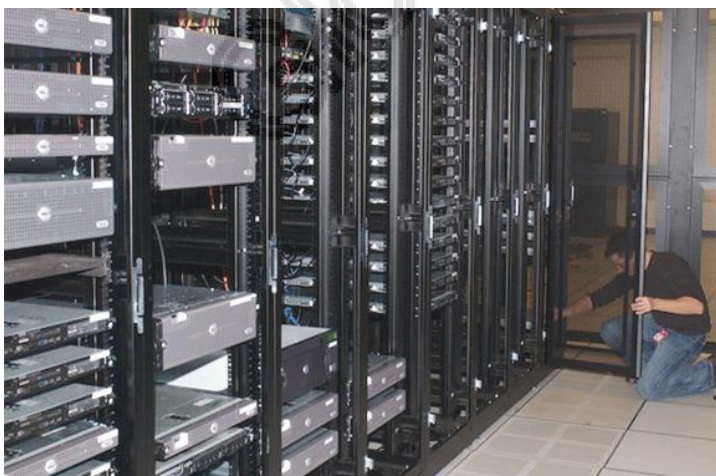


L'infrastructure désigne l'ensemble des éléments de type matériel et les logiciels composant le système informatique d'une entreprise ou d'une organisation.

2. LE DATA CENTER

2.1. Présentation

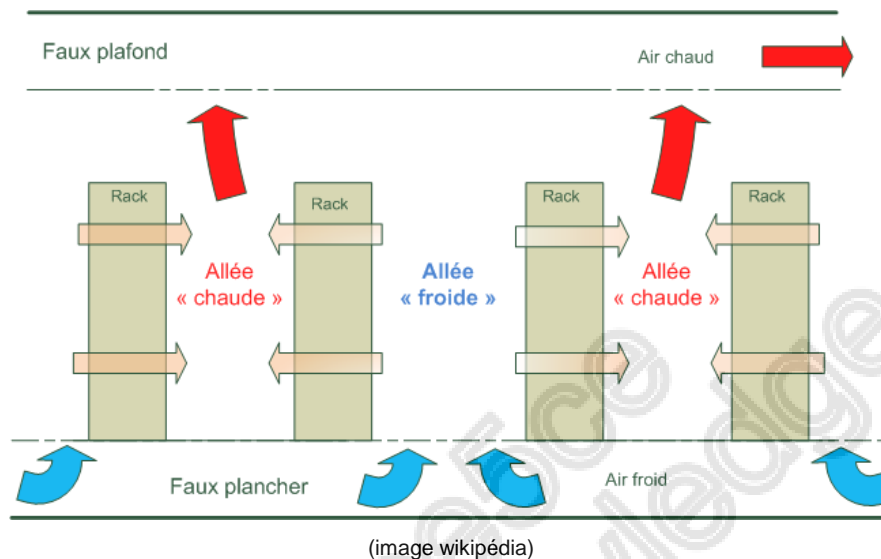
Le data center ou centre de traitements des données, est un local (un site) où sont regroupés les serveurs et tous les équipements réseaux, stockage, communication, électricité, etc. du système d'information soit d'une entreprise soit de plusieurs entreprises.



Les serveurs sont construits en rack, une **baie** en français, on parle aussi de U.

C'est-à-dire une « armoire » (aux dimensions standards) avec des rails pour glisser les composants **Hardware** constituant les serveurs (UC, disques, réseau) que l'on surnomme des boîtes à pizza.

Ces machines dégageant de la chaleur, il est nécessaire de refroidir la pièce. Dans le data center, les baies sont installées face à face afin de créer des allées chaudes et des allées froides.



2.2. Equipements physiques du data center

Etant donné qu'un data center conserve les données cruciales des entreprises, il faut avoir des niveaux de sécurité pour garantir la disponibilité, l'intégrité et la confidentialité de ces données.

Un data center est donc composé physiquement de :

- Climatisation
- Contrôle de la poussière (filtration de l'air)
- Unité de distribution de l'énergie
- Bloc d'alimentation d'urgence, et une unité de secours
- Câbles de paires torsadées de cuivre en Ethernet pour liaisons inter
- Fibres optiques pour liaisons inter-sites ou inter
- Conduites pour câbles au-dessus et au-dessous du plancher
- Système d'alerte d'incendie
- Extinction automatique des incendies (par microgouttelettes ou gaz inerte)
- Surveillance par caméras en circuit fermé
- Contrôle des accès, ainsi que sécurité physique du bâtiment
- Surveillance 24h/24 et 7j/7 des serveurs
- Service de sécurité continuellement présent

2.3. Hébergement et Tier

Quand une entreprise souhaite externaliser la gestion des serveurs, elle recherche un hébergeur.

Le choix de l'hébergeur, dépend bien sûr de la capacité des serveurs, (puissance, espace de stockage alloué, bande passante) mais aussi de tout le système de sécurité mis en œuvre. A savoir, pour une entreprise, l'indisponibilité du site ou la perte de données peuvent se révéler très problématique. **La sécurité n'a pas de prix, mais elle a un coût.**

L'organisme « Uptime Institute » (consortium d'entreprises) a défini une certification internationale et classé en 4 catégories, que l'on nomme « Tier ,» les centres de données :

Tier I - **basique** : Infrastructure non redondante, une seule alimentation électrique, la climatisation n'est pas redondante. (Dispose au moins d'un groupe électrogène et d'un onduleur). Une grande partie des maintenances et des pannes génèrent l'arrêt du site.

Tier II - **Redondance partielle** : Les éléments de production de froid et d'électricité sont redondants, mais la distribution d'électricité et de froid n'est pas redondante. Certaines maintenances et pannes génèrent un arrêt du site, notamment sur les circuits de distribution.

Tier III – **Maintenabilité** : Tous les composants sont maintenables sans arrêt de l'informatique. Les maintenances n'arrêtent pas le site, les erreurs humaines ou certaines pannes peuvent entraîner l'arrêt de l'informatique.

Tier IV : **Tolérance aux pannes**. Aucune panne n'arrête l'informatique (réponse automatique). Absence de SPOF (Single Point of Failure). Le Tier IV est tolérant aux maintenances, pannes (uniques), et incident même graves (incendie par exemple).

Pour information : en France jusqu'en 2014, il n'y avait aucun data center certifié. Fin 2015, quatre centres sont certifiés dont un en Tier IV (Crédit Agricole – Greenfield).

3. LE SYSTÈME D'EXPLOITATION

3.1. Présentation

« Programme, pouvant être livré par le constructeur de l'ordinateur, permettant à un utilisateur d'accéder aux ressources de la machine. »

Le système d'exploitation assure :

- la gestion des périphériques,
- la gestion de la mémoire,
- la gestion des tâches (exécutions de programmes),
- la gestion des anomalies.

3.2. Les différents types de système d'exploitation

3.2.1. Systèmes propriétaires, ouverts, libres

- Un système d'exploitation **propriétaire** (legacy system) est un système dépendant entièrement de l'architecture matérielle de la machine pour laquelle il a été conçu.

- Un système **ouvert** (open system) est un système d'exploitation conçu pour pouvoir dialoguer avec d'autres systèmes, même hétérogènes. Le premier système ouvert fut le système **UNIX**, implémenté sous des noms commerciaux différents chez chaque constructeur.

- Un système **libre**, est un système qui n'a pas été développé dans un but commercial ainsi chacun est libre de l'utiliser et de le modifier. C'est le cas de **Linux**, qui initialement a été conçu (dans les années 90) pour fonctionner sur plate-forme PC (en utilisant une version d'Unix) et est aujourd'hui présent sur de nombreuses plateformes.

Aujourd'hui, la plupart des constructeurs de système d'exploitation dit propriétaire (sauf Apple) ont ouvert leurs systèmes.

Principaux systèmes d'exploitation :

- **IBM** : z/OS (anciennement MVS) sur grands systèmes z/Series, System i , (OS/400) sur systèmes Power Systems (iSeries-AS400), AIX sur System p (machines RS/6000) est un système UNIX.
- **BULL** : GCOS 8 sur grands systèmes DPS 8 et DPS 9000, GCOS 7 sur minisystèmes DPS 7 et DPS 7000, IX sur minisystèmes Escalla (UNIX) (N'existe presque plus en 2017).
- **HP** : MPE sur mini systèmes HP 3000, HP-UX sur minisystèmes HP 9000 (UNIX).
- **Sun Microsystems** racheté par **Oracle** en 2009 Ex Sun-Solaris (UNIX), se nomme Oracle Solaris.
- **MICROSOFT** : Commercialise les systèmes d'exploitation Windows XP, Windows Vista, Windows 7 (seven), Windows 8, pour finir Windows 2010 et Windows Server 2003, Windows Server 2008, Windows Server 2012, Windows Server 2016 qui arrive courant 2016.
- **APPLE** : Commercialise les systèmes Mac OS X (Leopard, Snow Leopard, Lion, Mountain Lion).
- **Linux** : L'assemblage de Linux pour fournir un système clé en main est appelé une distribution, il existe différentes distributions : RedHat, SuSe, Debian, Ubuntu, Knoppix, Mandriva, Slackware, Gentoo...

3.2.2. Systèmes embarqués

Les systèmes embarqués sont des systèmes d'exploitation prévus pour fonctionner sur des **machines de petites tailles** : des appareils électroniques autonomes (sondes spatiales, robot, ordinateur de bord de véhicule, GPS, etc.), possédant une **autonomie réduite**.

Ainsi, une caractéristique essentielle des systèmes embarqués est leur gestion avancée de l'énergie et leur capacité à fonctionner avec des ressources limitées.

Les systèmes d'exploitation sont le plus souvent propriétaires. Cependant des partenariats existent entre les différents constructeurs.

Avec l'évolution des **appareils mobiles** (smartphones, tablettes et autres « phablettes »), des systèmes d'exploitation ont émergés :

- **Symbian** est un OS né d'une association entre Motorola, Nokia, Ericsson et Panasonic (anciennement Matsushita).
- **iOS** (Apple) sur iPhone et iPad
- **Android** (Google) sur téléphones et tablettes Samsung (gamme Galaxy), HTC, Sony (gamme Xperia)...
- BBOS (Blackberry), ex Palm
- **Windows Mobile** (ex Windows Phone) (Microsoft) sur téléphones Nokia (gamme Lumia), HTC... Microsoft commercialise sa propre tablette sous le nom de **Surface**.

Le marché des smartphones et des tablettes est aujourd'hui (2013) en très forte évolution. De nouveaux produits arrivent tous les 6 mois, les nouvelles versions des systèmes d'exploitation suivent quasiment le même rythme. Le Gartner group, cabinet américain, en 2016 estime que le marché des mobiles stagnera en 2018.

Côté parts de marché, Apple et Samsung se partagent plus des trois quarts du gâteau.

Au premier semestre 2013, Blackberry et Microsoft tentent une conquête du marché. Le premier avec la version 10 de son système (après l'échec de sa tablette), le second avec Windows 8 tablant sur la création d'un écosystème où PC, tablette et téléphone sont interconnectés et partagent la même interface utilisateur. En 2016, Blackberry n'a pas repris de part de marché, voire a même sombré. Pari plus ou moins réussi pour Microsoft, malgré son manque de part de marché, il a su créer son écosystème, l'OS Windows 10 Mobile sorti en mars 2016 reste gratuit comme la version 8, il intègre une version d'Office gratuite, optimisée pour le tactile.



4. RESEAU

4.1. Les différents types de réseau

La connexion de deux ordinateurs nécessite la présence d'un réseau. Il peut s'agir :

- d'un **réseau local** (LAN, *Local Area Network*), lorsqu'il est localisé au sein d'un bâtiment,
- d'un **réseau étendu** (WAN, *Wide Area Network*), lorsqu'il relie des ordinateurs situés dans des bâtiments distincts.
- de **VPN** , (Virtual Private Network), réseau d'entreprise privé, utilisant Internet comme support de transmission, en permettant d'envoyer des données sécurisées.

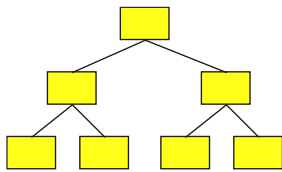
Jusqu'au 1^{er} janvier 1998, les réseaux étendus passaient uniquement par le biais de France Télécom – Orange, aujourd'hui sur le territoire national d'autres opérateurs sont apparus.

La répartition des machines **dans un réseau local** se fait selon 5 configurations possibles, on parle de **topologie** de réseaux

Il existe 2 modes de propagation classant les topologies :

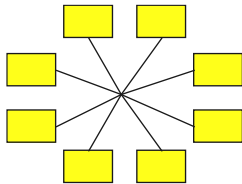
- Le mode diffusion, il n'utilise qu'un seul support de transmission. Le message est envoyé à tous, chaque destinataire l'analyse pour savoir si le message lui est adressé. (Cas des topologies bus et anneau).
- Le mode point à point, le support de transmission est relié à deux unités, qui pour communiquer passent par un intermédiaire nommé le **nœud**. (Cas des topologies en étoile ou maillée).

Topologie hiérarchique ou en arbre:



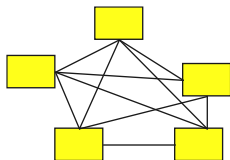
La machine de haut niveau est connectée à des machines inférieures, nommées **branches**. Les machines inférieures pouvant être aussi connectées à d'autres niveaux inférieurs.

Topologie en étoile :



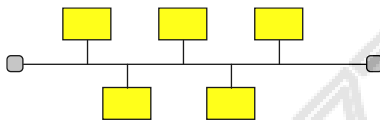
Les machines sont connectées à un matériel central, nommé concentrateur ou **hub** en anglais. Il s'agit d'un boîtier, permettant de raccorder les câbles réseaux des machines, pour assurer la communication.

Topologie maillée :



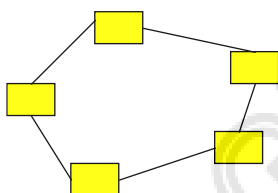
Il s'agit de plusieurs liaisons point à point, cas d'**Internet**. Chaque machine est reliée à toutes les autres. L'inconvénient est le nombre de liaisons nécessaire pour relier les machines entre elles.

Topologie en bus (topologie des réseaux Ethernet) :



Cas la plus simple à mettre en œuvre. Chaque machine est reliée à la même ligne de transmission, que l'on nomme un bus. L'inconvénient est que si une connexion est défectueuse, l'ensemble du réseau est affecté.

Anneau à jeton (topologie des réseaux IBM Token Ring) :

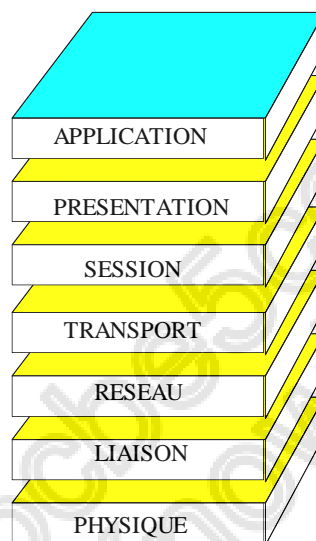


Les machines sont reliées à un **répartiteur** nommé MAU (Multistation Access Unit), qui permet d'allouer un temps de parole à chaque machine, elles communiquent donc à tour de rôle.

4.2. Les protocoles réseau

4.2.1. Le modèle de référence OSI

La communication entre ordinateurs hétérogènes n'est possible qu'à condition que chaque protagoniste adopte un langage commun. La traduction vers ce langage a été modélisée par l'ISO (International Standard Organization) via le modèle OSI (Open System Interconnections) à 7 couches :

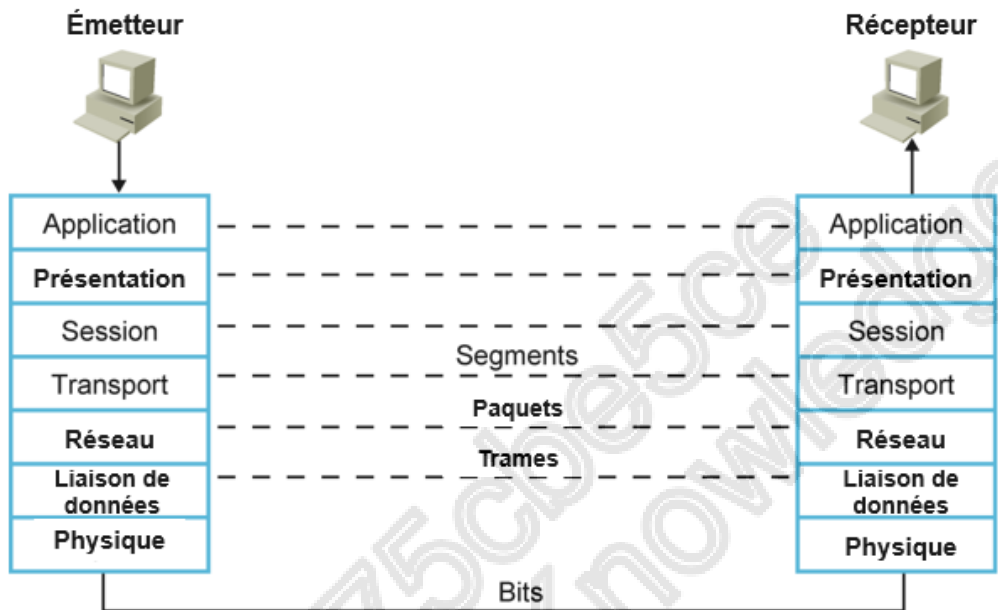


- La couche **physique** code l'information en influx électriques, lumineux ou en ondes. Normes V24, RS232, 10BaseT...
- La couche **liaison** contrôle l'information reçue. Normes BSC, HDLC...
- La couche **réseau** assure le routage (choix du chemin). Norme X25, IP...
- Les couches **transport** et **session** prennent en charge l'interconnexion de réseaux hétérogènes. Norme TCP pour une partie de la couche transport.
- La couche **présentation** met en forme les données.
- La couche **application** offre les services réseau (transfert de fichiers, connexion sur une machine distante, messagerie...).

Un protocole réseau est un ensemble de règles régissant la transmission de données sur le réseau.

4.2.2. Communication entre 2 ordinateurs

Pour permettre l'envoi d'information, chaque couche du modèle OSI, au niveau de l'émetteur doit communiquer avec sa couche homologue sur le récepteur. On nomme ce processus la communication peer-to-peer.



4.2.3. Le modèle TCP/IP

La suite TCP/IP est l'association de 2 protocoles Transmission Control Protocol et Internet Protocol. La suite est divisée en couche, exécutant des fonctions spécifiques pour communiquer.

La suite a été développée presque en même temps que le modèle OSI, par des organismes différents. Elle contient 4 couches, représentant les mêmes fonctionnalités que le modèle OSI. Afin d'effectuer un parallèle entre les 2 modèles la couche « accès réseau » de la pile TCP/IP peut être représentée schématiquement par 2 couches, d'où une représentation à 5 couches de cette pile. Le modèle OSI divise la couche applicative de la pile TCP/IP en 3 couches.

La couche d'accès réseau de la pile TCP/IP, couvre les couches « physique » (1) et « liaison de données » (2) de l'OSI.

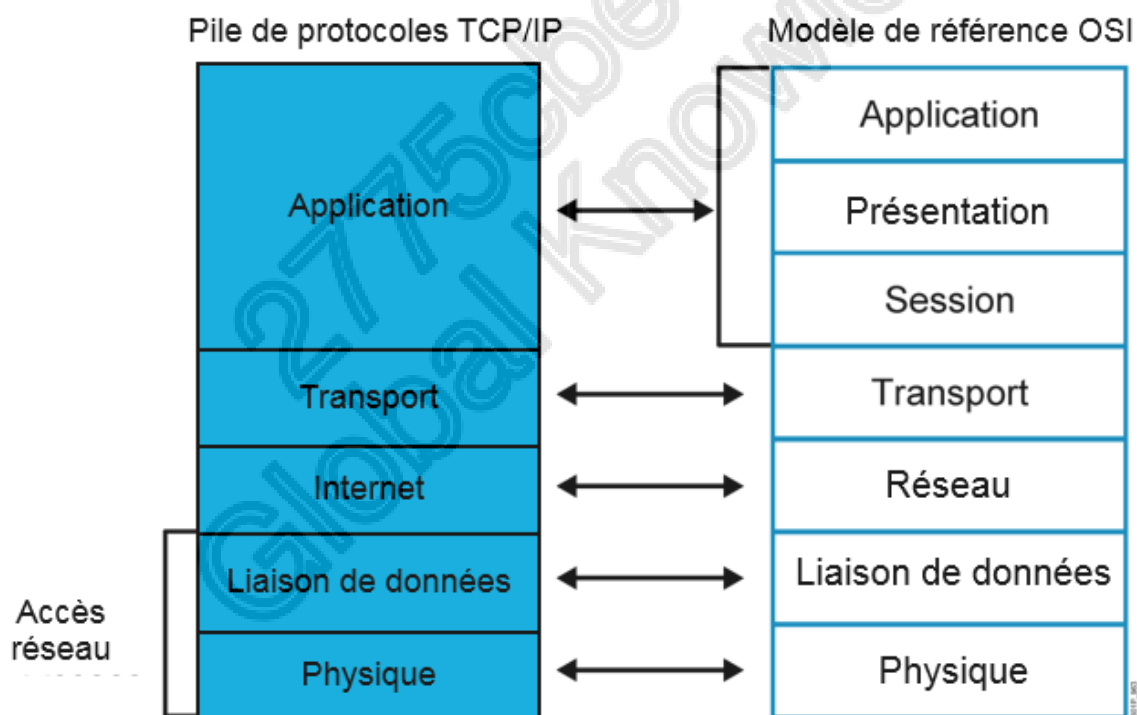
La couche Internet assure l'acheminement des données de l'émetteur au récepteur en définissant le paquet et le schéma d'adressage.

La couche transport fournit les services de communications aux processus des applications exécutés sur les hôtes du réseau.

La couche application fournit les applications telles que le transfert de fichiers, le dépannage du réseau, les activités sur Internet.

Le protocole TCP/IP est le plus utilisé aujourd'hui.

Pile TCP/IP et modèle OSI



4.3. Les équipements réseaux

- Un **hub**, ou un concentrateur, est appareil permettant de brancher plusieurs appareils informatiques à un port d'un PC. (Equivaut à une multiprise).
- Un **switch**, ou un commutateur, effectue des opérations sur la couche 2 = couche liaison de l'OSI. C'est un boîtier qui permet de relier plusieurs câbles au réseau, il dispose de plusieurs ports Ethernet. Le switch utilise les adresses MAC pour adresser plus rapidement les machines et éviter que chaque destinataire vérifie si la communication lui est adressée. Au niveau matériel le switch ressemble au hub.

Un switch



Un hub



- Un **router**, effectue des opérations sur la **couche 3** = couche réseau de l'OSI. C'est un élément intermédiaire du réseau qui permet de transmettre à un autre nœud les paquets. Le router utilise les adresses IP pour diriger les données.

Un router wifi



Des routers Cisco



- Un **firewall**, permet de faire respecter la politique de sécurité du réseau. Gère les connexions entrantes et sortantes de l'entreprise vers Internet. Cela peut-être un matériel informatique ou un logiciel.



- Un **NAS** (Network Attached Storage), est un équipement, relié au réseau dont la principale fonction est le stockage de données. Cela permet de centraliser les sauvegardes de données venant de serveurs hétérogènes. Les applications les plus souvent stockées sur des serveurs NAS sont les messageries d'entreprise et les serveurs web.



- Un **SAN** (Storage Area Network) est un réseau de stockage. Son but est de mutualiser les ressources de stockage. (Mutualiser : c'est-à-dire utiliser un même équipement pour des applications différentes afin de partager les coûts). Le NAS, peut être le point d'entrée de ce réseau, ou y être intégré.

5. SYSTEME ET RESEAU

Un **nœud** (node en anglais) est l'extrémité d'une connexion. Le nœud peut-être :

- Un hub , ou un concentrateur,
- Un switch, ou un commutateur
- un routeur,
- un serveur.

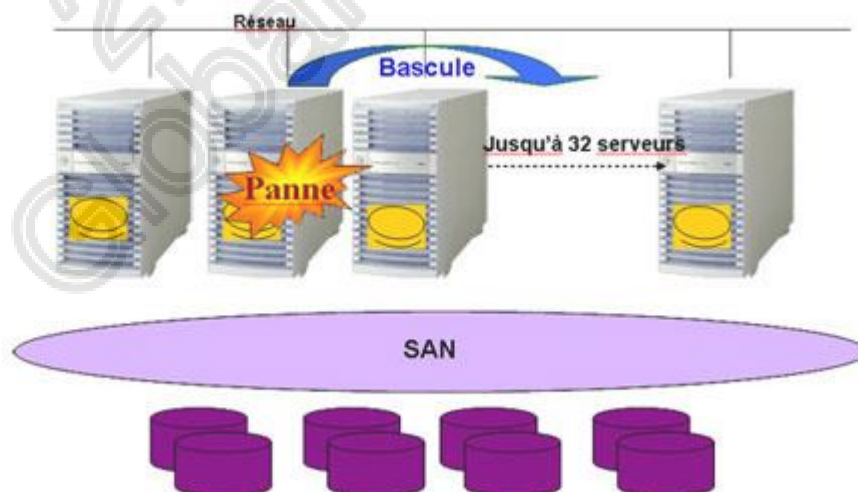
Un **cluster** est une **grappe** de serveurs (ou ferme de calcul) constitué de 2 serveurs au minimum que l'on appelle dans ce cas des nœuds.

Le cluster partage une « baie de disques » commune aux différents serveurs du cluster, cette baie peut permettre :

- d'assurer la disponibilité des services,
- de répartir la charge de calcul,
- de répartir la charge du réseau.

Ainsi, le cluster permet une gestion centralisée des ressources des serveurs (processeurs, mémoire, disque dur, bande passante sur le réseau) et permet de multiplier les capacités des machines. De plus, en cas de panne d'un serveur, le logiciel de *clustering* est capable de transférer l'information du serveur défaillant vers d'autres serveurs de la grappe. Cette mise en œuvre de disques partagés se retrouve dans le cadre d'un réseau SAN.

Le cluster est un ensemble de machines homogènes et localisées, organisées en grappe.



6. LA VIRTUALISATION

6.1. Objectifs

La virtualisation s'applique principalement à deux grandes familles : le stockage d'une part (disques et bandes magnétiques), les serveurs d'autre part.

Le but de la **virtualisation de serveurs** est de faire fonctionner sur un même ordinateur plusieurs systèmes d'exploitation, comme s'ils fonctionnaient sur des ordinateurs différents.

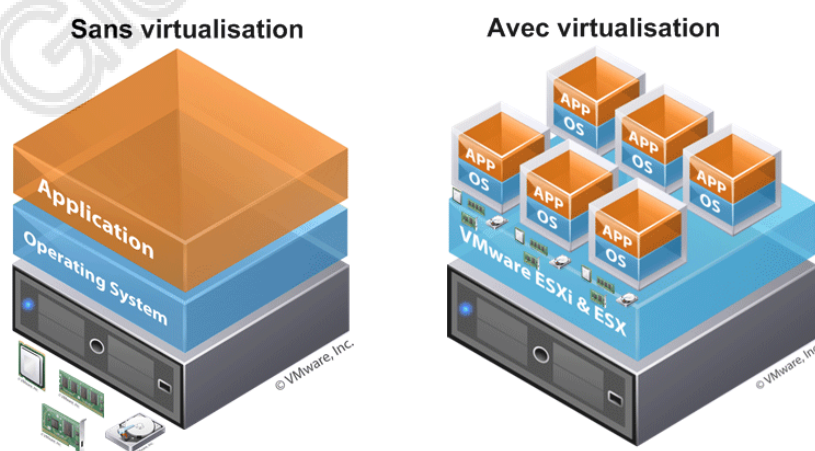
Dans l'entreprise, il existe souvent plusieurs machines qui ne sont pas utilisées à 100% de leur capacité. L'idée est de regrouper plusieurs serveurs sur une même machine afin d'utiliser le maximum de la capacité.

La virtualisation simplifie les tâches d'administration, de gestion, de réplication, de copie d'informations.

6.2. Notions prises en charge

La virtualisation prend en charge les notions suivantes :

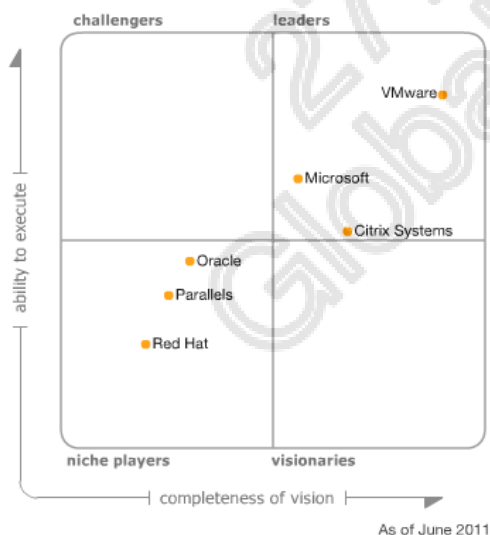
- Abstraction matérielle/logicielle.
- Système d'exploitation hôte.
- Système d'exploitation (ou application) virtualisé ou invité.
- Partitionnement, isolation et partage des ressources physiques et logicielles.
- Images manipulables : arrêt/démarrage – gel – clonage – sauvegarde/restauration – migration d'une machine vers une autre.
- Réseau virtuel : le réseau est purement logiciel, interne à la machine hôte entre les invités.



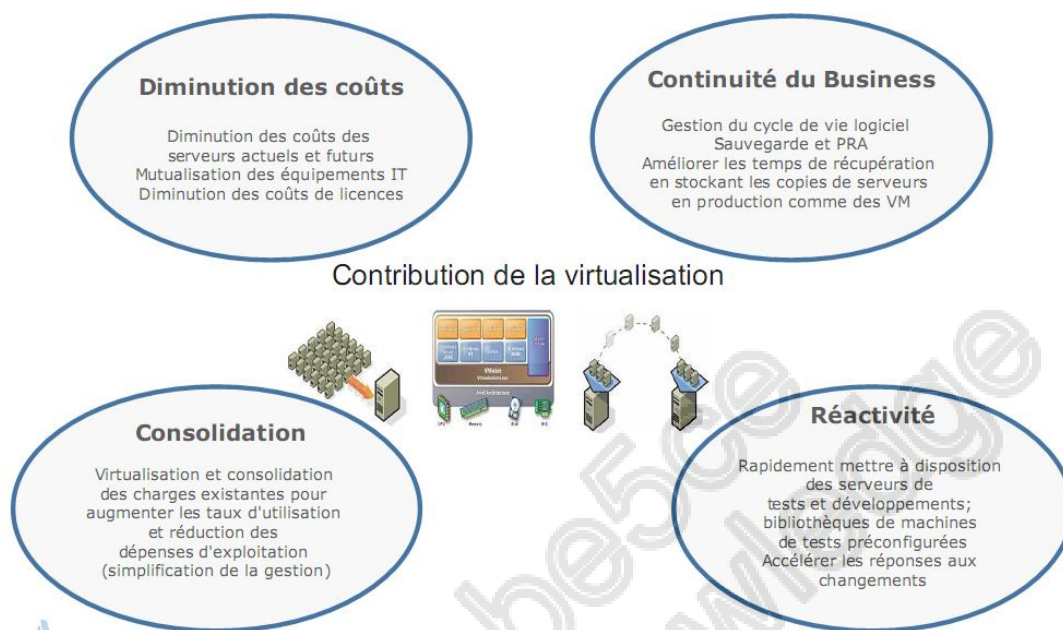
Offres

Deux types d'offres de virtualisation de serveurs sont présents qui ne cessent d'évoluer :

- Le 1er type d'offre englobe les offres de **Microsoft**, **EMC (VMWare)** ainsi qu'un projet Open Source de l'Université de Cambridge dénommé **Xen**.
- Le second type d'offre est proposé par **Swsoft** qui propose une solution permettant un partage d'une même instance d'un système d'exploitation, sans toutefois être capable de dédier des processeurs spécifiques à chaque machine virtuelle ni de pouvoir gérer différentes versions des OS, à l'inverse des solutions de Microsoft et de VMware.
- **MICROSOFT** : Deux solutions
 - Microsoft Virtual Server 2008 – Standard
 - Microsoft Virtual Server 2008 – Enterprise
- **EMC (VMWare)** : Deux solutions
 - ESX
 - CGX
- **Xen**
- **Swsoft** : Deux solutions
 - Virtuozzo Linux
 - Virtuozzo windows.
- Le Gartner Group présente les principaux acteurs de la virtualisation :



6.3. Les enjeux de la virtualisation



CHAPITRE III

LES ARCHITECTURES DISTRIBUEES

Objectif : décrire les différents types d'architecture et lister les enjeux de chaque architecture.

1. EVOLUTION DES ARCHITECTURES

1.1. Avant les années 90

L'informatique centralisée

Les premières architectures sont des architectures centralisées. Des terminaux passifs sont reliés à l'ordinateur central via le réseau.

L'ensemble de la solution est une solution « propriétaire », matériel et logiciel ayant été livrés par le même fournisseur.

L'informatique décentralisée

Les progrès de l'informatique aidant, de plus en plus de fonctions de l'entreprise, et donc de plus en plus de sites sont informatisés. Des besoins d'échanges de fichiers, de connexion à distance,... nécessitent l'interconnexion des différents ordinateurs de l'entreprise. Les relations entre les différentes machines sont des relations de type « maître-esclave ».

La micro-informatique isolée

Au début des années 1980, la micro-informatique fait son apparition dans les entreprises.

Loin d'avoir la puissance des machines actuelles, le micro-ordinateur est vu comme un outil bureautique uniquement, au même titre que la machine à écrire.

Informatique du système d'information et micro-informatique sont deux mondes qui s'ignorent.

La micro-informatique intégrée

Dorénavant, l'utilisateur sollicite de plus en plus son micro-ordinateur, dont la puissance, conformément à la loi de Moore, croît linéairement.

De nouveaux besoins, tels que le partage d'imprimante, l'émulation du host,... nécessitent l'interconnexion des micro-ordinateurs et donc l'apparition des réseaux locaux.

L'utilisateur a accès, depuis le même poste, à deux types d'applications :

- bureautiques en local,
- centrales en émulation.

1.2. Après les années 90

L'informatique distribuée

Aujourd'hui, le système d'information d'une entreprise est bien souvent constitué de machines, de logiciels et de réseaux provenant de fournisseurs différents.

Ces différents éléments sont interconnectés et sont de plus en plus utilisés pour exécuter des applications temps réel.

Cette architecture peut supporter des applications respectant :

- un modèle classique, toutes les ressources que l'application utilise étant localisées sur la même machine,
- un modèle client/serveur, les ressources utilisées pouvant être réparties sur n'importe quelle machine du réseau.

On nomme, client le poste qui se connecte au serveur.

1.3. Après les années 2000

L'informatique mobile

Avec le développement des smartphones, des tablettes et autres appareils, il devient possible de se connecter en mobilité.

Cette architecture peut supporter des applications :

- Natives : l'application est développée pour le système d'exploitation du smartphone concerné.
- En ligne : il s'agit alors d'application web indépendante du type de smartphone.

2. L'INTEROPERABILITE ET PORTABILITE

2.1. Définition et enjeux

L'interopérabilité entre les composants d'un système d'information distribué peut se définir comme la capacité de ces composants à échanger des services et des données. Elle repose sur des conventions adoptées entre les parties communicantes et régissant, entre autres, les protocoles d'échange de messages, la désignation des procédures à activer ou les codes d'erreur retournés.

Elle présente une dimension statique (comme, par exemple, la compatibilité des types de données) et une dimension dynamique (comme la compatibilité des procédures).

L'interopérabilité repose tout d'abord sur l'inter connectivité des composants, qui est elle-même régie par le respect de normes et de standards informatiques de communication.

L'interopérabilité impose l'indépendance vis-à-vis:

- Des plates-formes
- Des couches réseau
- Des systèmes d'exploitation
- Du langage de programmation
- Des formats de données
- Et c'est l'un des objectifs du client / serveur

2.2. Normes et standards

Ont émergé au cours de l'évolution du client serveur, quatre standards fonctionnels d'interopérabilité :

- **CORBA** (Common Object Request Broker Architecture) dû à l'OMG (Object Management Group), un groupe de travail rassemblant de nombreux constructeurs et éditeurs de logiciel dans le but de fournir une base de travail indépendante de la plateforme.
- **DCOM** (Distributed Component Object Model) concurrent direct de CORBA que Microsoft a cherché à imposer depuis les plateformes de type Windows.
- **Java RMI** (proposé par Sun à l'origine, Oracle aujourd'hui) système d'objets distribués axé sur l'utilisation du langage Java. Son rapprochement avec le système CORBA est indéniable.
- **Web Services** avec les standards coordonnés par le W3C tels que **SOAP**, **WSDL** et **UDDI** (détaillés plus loin).

Les évolutions matérielles et technologiques des 15 premières années du XXIème siècle ont permis l'émergence du **cloud computing**, l'informatique dans les nuages. Nous reviendrons sur ce concept, après avoir décrit les architectures.

3. LES ARCHITECTURES DISTRIBUEES

Une architecture est dite distribuée si elle est constituée de plusieurs entités (composants et clients) séparées en termes d'accès réseau.

« **Est conforme au client-serveur une application qui fait appel à des services distants au travers d'échanges de *messages* (requêtes) plutôt que par un échange de fichiers.** ».

Il existe plusieurs types de réseaux par lesquels les communications entre les composants et les clients s'achèment :

- LAN (Local Area Network) ou l'intranet, réseau interne fermé au monde extérieur.
- WAN (Wide Area Network) ou extranet, réseau interne offrant la possibilité à des entités extérieures autorisées de se connecter.
- Internet, réseau ouvert.

Cet échange de messages transite à travers le réseau reliant les deux machines. Il met en œuvre des mécanismes relativement complexes qui sont, en général, pris en charge par un **middleware**.

Le Gartner Group définit le middleware comme une interface de communication universelle entre processus. Il représente véritablement la clef de voûte de toute application client-serveur.

L'objectif principal du middleware est d'unifier, pour les applications, l'accès et la manipulation de l'ensemble des services disponibles sur le réseau, afin de rendre l'utilisation de ces derniers presque transparente.

Exemples de middleware :

SQL*Net : Interface propriétaire permettant de faire dialoguer une application cliente avec une base de données Oracle. Ce dialogue peut aussi bien être le passage de requêtes SQL que l'appel de procédures stockées.

ODBC : (Open DataBase Connectivity) Interface standardisée isolant le client du serveur de données. Elle se compose d'un gestionnaire de driver standardisé, d'une API s'interfaçant avec l'application cliente (sous Ms-Windows) et d'un driver correspondant au SGBD utilisé.

DCE : (Distributed Computing Environment) Mécanisme de RPC proposé par l'OSF. Permet l'appel à des procédures distantes depuis une application.

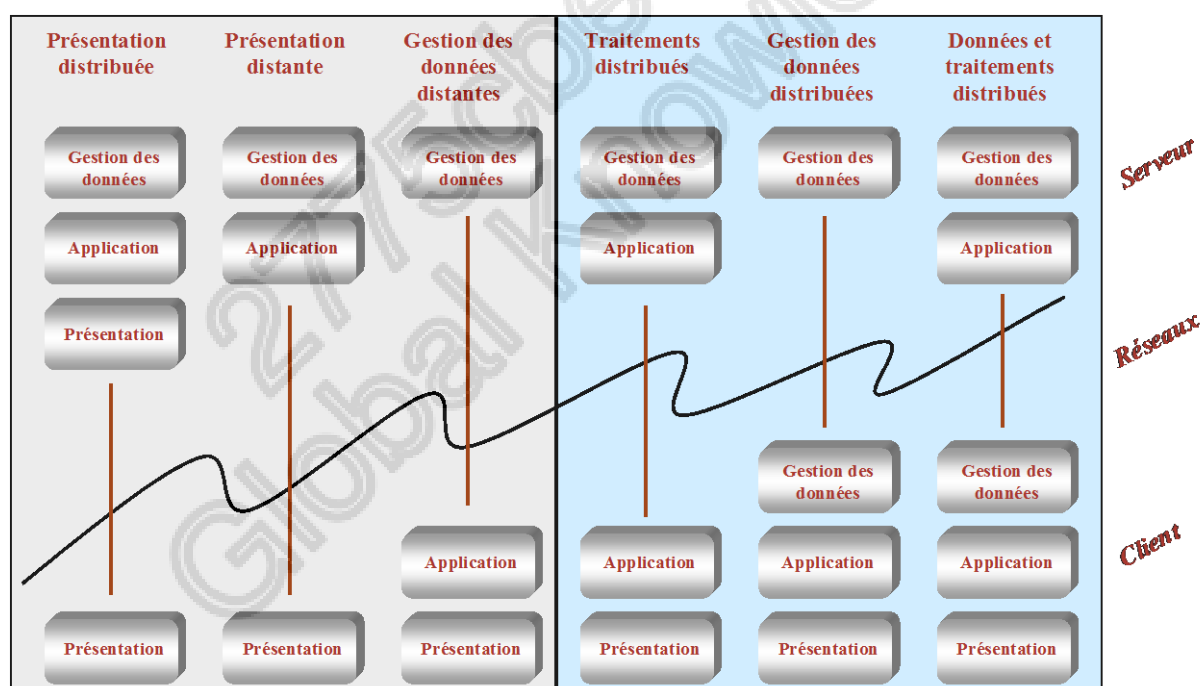
4. LE MODELE DU GARTNER GROUP

Tout programme applicatif contient trois types de traitements :

- les traitements liés à la **présentation**, c'est-à-dire à la gestion de l'interface utilisateur,
- les traitements liés aux **données** manipulées,
- les traitements métiers (application).

Le Gartner Group, cabinet de conseil américain, a modélisé les différents types de répartition des trois composants (présentation, données, application) sur deux machines.

Schéma du modèle du GARTNER GROUP



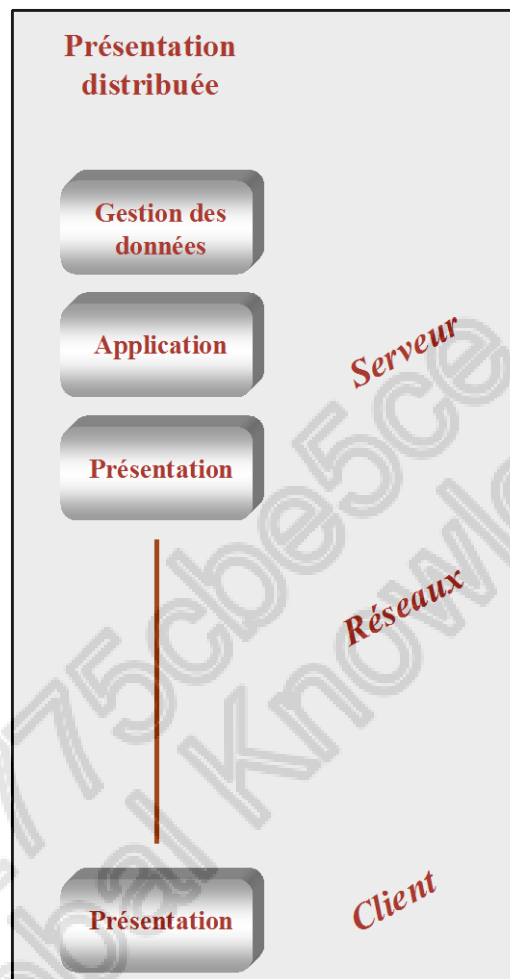
Ce graphique montre les différentes évolutions des architectures 2-tiers.

Nous y voyons les **principaux modèles de répartition des couches** entre le client et le serveur.

En général, on appelle client l'émetteur d'une requête et serveur le récepteur chargé de traiter la requête émise.

4.1. Client – Serveur de première génération :

4.1.1. Le rhabillage d'applications



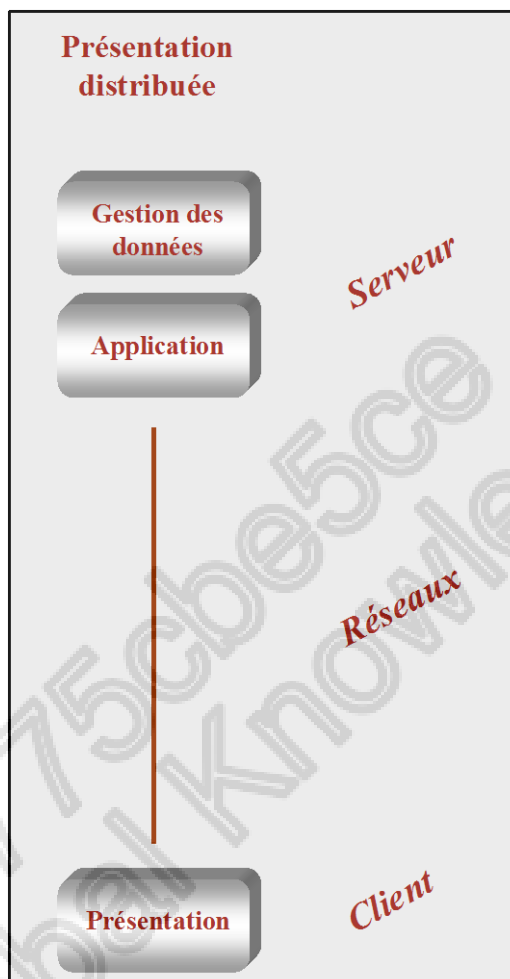
Le rhabillage d'applications (ou revamping) est une surcharge de la couche présentation.

Il consiste à intégrer le poste de travail passif du serveur dans un environnement micro.

Posant peu de difficultés techniques, il amène néanmoins un gain important à l'utilisateur final, en particulier en termes d'homogénéisation avec les autres applications. La présentation est distribuée, la responsabilité de la présentation est partagée entre le client et le serveur.

La classification client-serveur du revamping est souvent jugée abusive, du fait que l'intégralité des traitements originaux est conservée et que le poste client conserve une position d'esclave par rapport au serveur.

4.1.2. La présentation distante



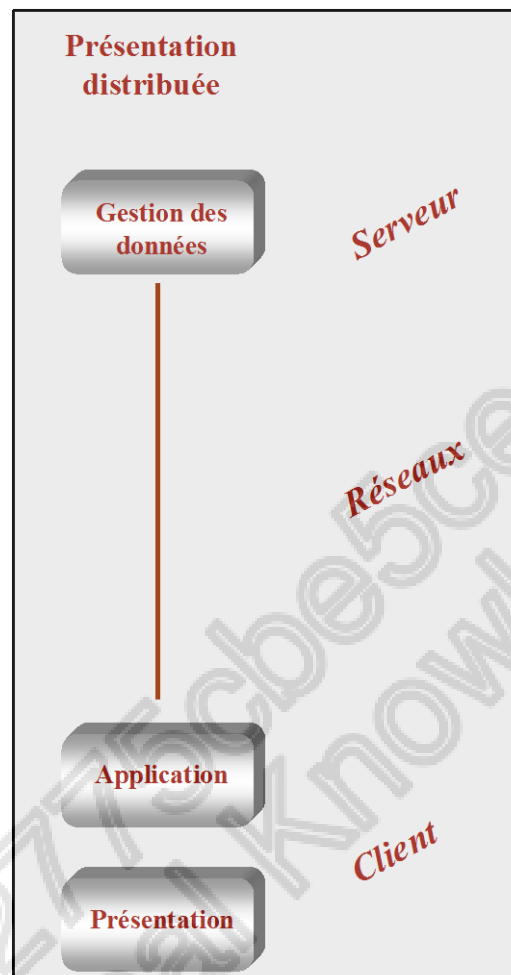
Dans ce mode de fonctionnement, tout l'affichage est réalisé sur le poste client. On parle de client léger et de client-serveur de présentation.

Cette configuration correspond à l'utilisation d'un terminal/X, utilisé dans le monde Unix.

S'il n'était que rarement retenu dans sa forme primitive, il connaît aujourd'hui un très fort regain d'intérêt avec l'exploitation des standards Internet.

La présentation est alors prise en charge par un navigateur en utilisant les standards du Web.

4.1.3. La gestion des données distantes



Le client réalise la présentation ainsi que tous les traitements de l'application. On parle dans ce cas, de client lourd / serveur léger et de client-serveur de données.

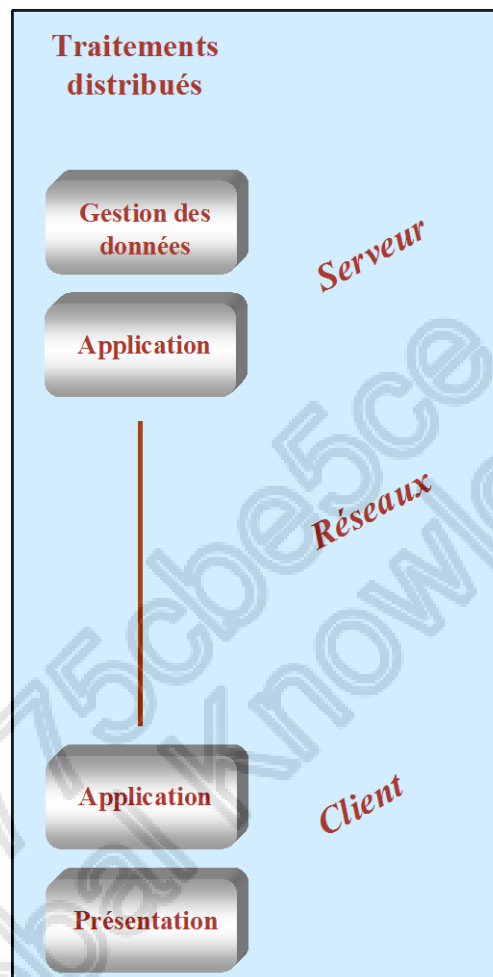
Du fait de l'absence de capacité de traitement sur le poste serveur, les ordres d'accès aux données ainsi que la totalité des informations traitées transitent par le réseau, ce qui génère de gros flux de données. Exploitable sur un réseau local, ce modèle est donc souvent inadapté à l'utilisation sur un réseau étendu.

Il présente aujourd'hui peu de difficultés techniques, la communication par le réseau étant souvent prise en charge par le SGBD/R.

Si ce principe offre l'avantage de pouvoir « rapidement » mettre en œuvre une application distribuée, il présente l'inconvénient (de taille) d'une administration très coûteuse.

4.2. Client – Serveur de deuxième génération

4.2.1. Les traitements distribués

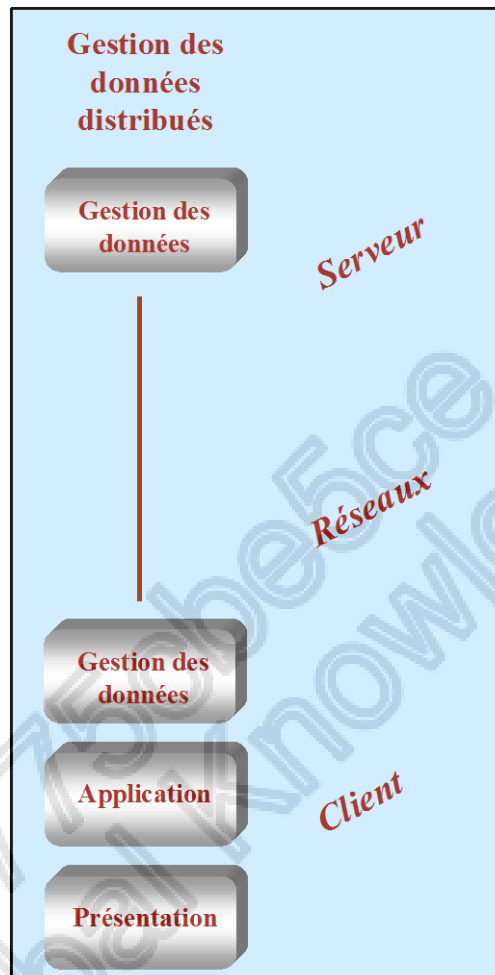


Le client prend en charge la présentation, les contrôles de premier niveau et les règles de gestion particulières de l'environnement du client. Le serveur prend en charge l'accès et la mise à jour des données, les traitements communs d'une application pour tous ses programmes clients et les traitements communs à toutes les applications.

Les principales difficultés viennent de la localisation des traitements distants et de la communication réseau. Des solutions simples existent, comme celle de confier encore une fois cette tâche au SGBD/R. Mais ces types de solution n'apportent pas de réponse satisfaisante à toutes les configurations techniques, même des plus classiques. Modèle complexe à mettre en œuvre, mais permettant de prendre en compte des impératifs de performance et d'intégration.

La charge des traitements applicatifs est distribuée entre le client et le serveur afin de profiter des performances de certaines plateformes client.

4.2.2. Les données distribuées

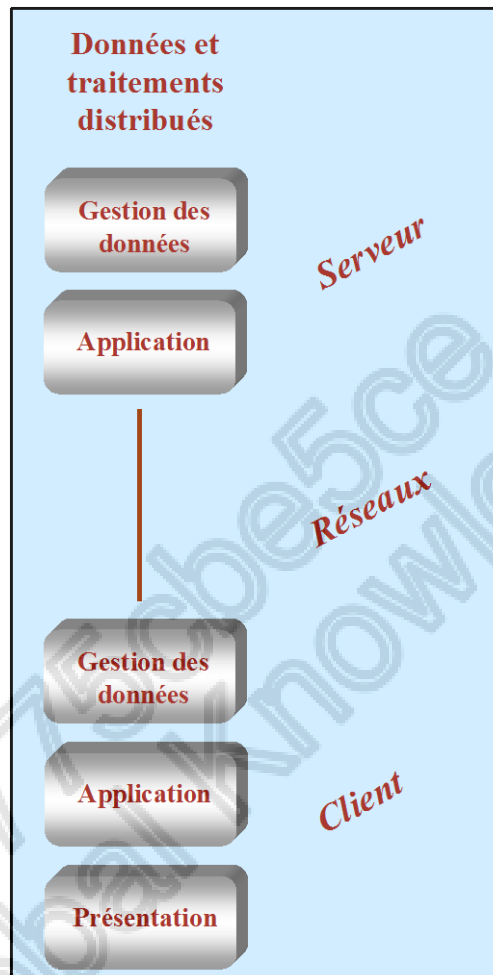


Le poste client assure la présentation, les traitements et la gestion des données privées. Le serveur assure la gestion des données communes à tous les clients.

Le principal problème posé par ce modèle est la synchronisation des données distribuées.

La synchronisation sera effectuée selon une procédure d'entérinement en deux étapes, dite de « *two phase commit* ».

4.2.3. Les traitements et les données distribués



Modèle le plus difficile à mettre en œuvre, on parle de répartition des données et des traitements, il faut pouvoir répartir et distribuer des éléments qui sont en fait des objets.

Modèle puissant qui permet l'utilisation de la notion de *composants réutilisables*.

La réutilisation peut être faite par l'usage de messages d'entreprise.

La réutilisation peut être effectuée par l'utilisation d'objets.

4.3. Les évolutions du client - Serveur

4.3.1. Le modèle client-serveur « classique »

Tout au long des années 1980, les évolutions du paysage informatique vont remettre en cause les conditions même d'organisation du travail et de production logicielle. Elles se caractérisent par :

- une augmentation des performances matérielles de la micro-informatique (vitesse des processeurs, capacité de stockage accrue) ;
- une fiabilité accrue du matériel ;
- une diminution significative des coûts ;
- une banalisation de l'utilisation de logiciels de bureautique qui ne sont plus seulement réservés à des informaticiens ;
- l'arrivée des interfaces en mode graphique.

Les raisons principales des dysfonctionnements de ce type sont :

- complexité de mise œuvre dans un environnement hétérogène,
- absence de standard dans la couche de middleware,
- performances moindres en regard d'une solution centralisée,
- coût excessif du déploiement et de la maintenance,
- syndrome du « client lourd ».

Plusieurs éditeurs se sont lancés dans l'aventure du client-serveur afin de proposer des produits uniques assurant la communication des traitements répartis : SQLWindows de Gupta corp., Powerbuilder de Sybase ou NS-DK de Nat Systems par exemple, mais le succès attendu ne fut pas au rendez-vous.

4.3.2. Le client-serveur universel sur Internet

Depuis la fin des années 1990, le raz de marée Internet bouleverse les architectures client-serveur et impose un modèle unique au moins en ce qui concerne les parties client et le middleware. Le client est redevenu « léger » : un simple navigateur est nécessaire (comme Mozilla ou Internet Explorer). Le middleware est standardisé : protocole unique de transport d'information (TCP-IP), protocoles simples et partagés d'échange de fichiers (HTTP, FTP...). Enfin, l'existence d'un langage universel simple, voire simpliste, (HTML) permet de définir la mise en forme des documents complexes.

Cette architecture simplifiée côté client peut s'enrichir de programmes téléchargés résidents sur la machine cliente (plug-in), ou de programmes écrits en java (applet) chargés en même temps que la page mais dont la durée de vie est liée à la durée de l'échange avec le serveur (session) ; des scripts inclus dans le code HTML peuvent aussi déporter certains traitements côté client.

4.3.3. L'avènement des services Web

Les grands acteurs du marché de l'informatique (Microsoft, IBM....) ont élaboré des solutions d'échanges de services applicatifs, ce sont les web services.

Les services informatiques peuvent proposer un service d'échange en ligne des fichiers basé sur des protocoles simples et transparents :

- le format standard HTTP,
- un format standard d'échange des documents, le langage XML,
- un format commun d'invocation des messages, le protocole SOAP.

Un service peut être accessible sur un serveur par le simple biais d'une adresse standard (URL), le client configuré pour communiquer (protocole SOAP) demande l'activation du service, le service retourne le résultat sous la forme d'un document au format standard XML.

On peut objecter que ce type de programmes pourrait être développé dans une architecture client-serveur classique mais c'est justement le mérite du service web que d'avoir normalisé des standards en utilisant des protocoles partagés par la communauté informatique.

5. L'ARCHITECTURE WEB

5.1. Présentation d'Internet

Internet est issu du réseau ARPANET, réseau militaire américain créé en 1969.

Il s'est progressivement ouvert à la communauté scientifique et aux universités américaines, à la France en 1986 (INRIA) et enfin aux entreprises et aux particuliers depuis 1989.

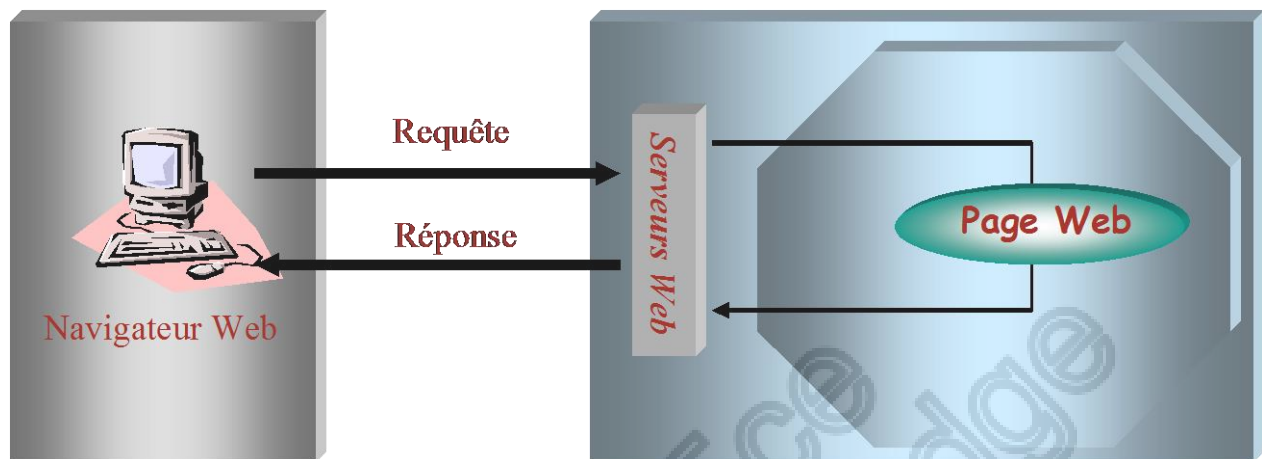
La technologie qui a permis l'engouement actuel est la création du protocole HTTP (Hyper Text Transfert Protocol), qui permet d'accéder aux différentes pages d'informations par le biais de liens hypertextes. L'ensemble des pages HTTP disponibles constitue le **World Wide Web**.

Accessible 24 heures sur 24, Internet est aujourd'hui **un réseau mondial** de communication d'informations, reliant des centaines de milliers d'ordinateurs.

Le réseau Internet offre trois fonctionnalités principales :

- la **consultation** de pages d'informations ou la consultation et la mise à jour de bases de données, *via* le protocole applicatif HTTP,
- les échanges de **courriers** électroniques (le *mail*), *via* le protocole SMTP,
- le **téléchargement** de fichiers, en utilisant le protocole FTP.

5.2. Composants architecturaux du Web



5.2.1. Le navigateur Web

Un navigateur Web repose sur la notion d'interface universelle :

- Il envoie des requêtes à un serveur Web quel qu'il soit.
- Il réceptionne et gère les réponses retournées par le serveur.
- Il interprète et affiche les réponses retournées.

Le format « universel de présentation » est HTML (Hyper Text Markup Language).

La page **HTML** peut aussi contenir du code qui sera interprété par le navigateur, le code est souvent écrit en JavaScript. Nous reviendrons sur les différents langages du Web.

Le navigateur peut aussi exécuter des applications dans l'environnement de la page Web (ActiveX ou Applets).

Plusieurs navigateurs sont disponibles sur le marché et intègrent l'accès aux pages Web, le transfert de fichier, le courrier électronique :

Microsoft Internet Explorer, Firefox, Opéra, Safari, Google Chrome...

5.2.2. Le serveur Web

Le serveur Web est un programme qui s'exécute sur un serveur. Il attend des requêtes pour les interpréter.

Suivant la nature de la requête, le serveur Web recherche une page ou exécute un programme.

Quelque soit le traitement, le serveur Web retournera toujours une page (même si c'est une erreur).

Le protocole de communication utilisé entre le client et le serveur est HTTP.

Le serveur est dit **serveur HTTP** lorsqu'il ne gère que des documents HTML.

On parle aussi de **Web statique**, même si la page contient en plus du HTML du code JavaScript.

Plusieurs serveurs HTTP sont disponibles sur le marché les plus connus étant :

Microsoft IIS (Internet Information Server) et Apache (Apache Foundation).

A partir du moment où l'on exécute une application sur le serveur pour générer la réponse faite à l'utilisateur, on parle de **Web dynamique**. Dans ce cas le serveur HTTP n'est pas suffisant, il fait appel à d'autres programmes que l'on nomme des serveurs d'applications.

Applications Web

Chaque serveur est identifié par une adresse **IP**, (exemple : 217.19.60.142), chaque page Web est identifiée par une **URL** (Unified Resource Locator).

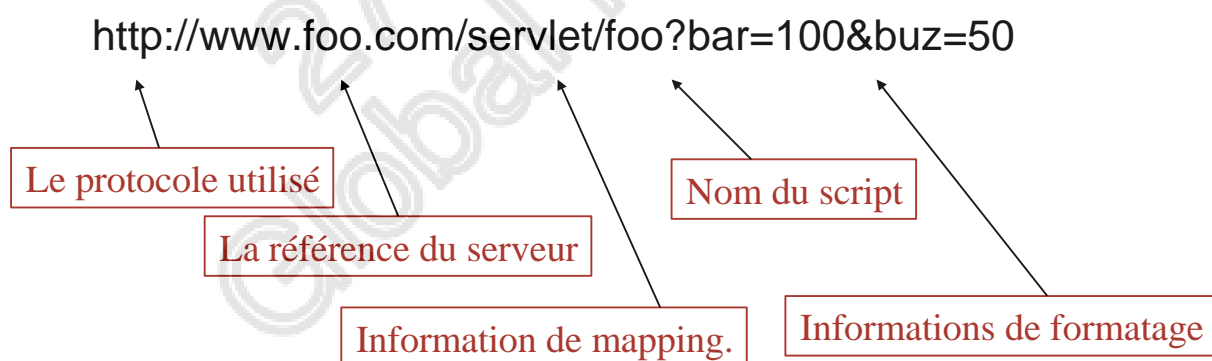
exemple : <http://www.globalknowledge.fr/search/?SearchTerm=projet+informatique>

Pour comprendre l'interprétation des requêtes par le serveur, il faut décrire préalablement son contenu.

Une requête HTTP est constituée des parties suivantes :

- Le protocole : http:
- Le nom du serveur : www.foo.com
- Une URL référençant la page ou le script à activer : /servlet/foo
- Des données dites de formatage : liste d'attribut=valeur
- Des informations d'en-tête : caractéristiques du navigateur, longueur et type de la requête, etc.

Structure des requêtes http :



Le serveur localise la ressource correspondante à la requête, l'interprète et renvoie au navigateur une page Web. Le navigateur interprète la page renvoyée par le serveur.

5.3. Autour du Web

5.3.1. Intranet-Extranet

L'**intranet** correspond à un ensemble de services réseau (site web, messagerie...) supportés par l'infrastructure Internet, mais dont l'accès est restreint à un groupe d'utilisateurs particulier, par exemple l'ensemble des salariés d'une entreprise.

L'**extranet** définit une restriction moins stricte que l'intranet. C'est par exemple le cas d'un site web réservé à une entreprise et à ses clients, qui ont accès au site en saisissant le mot de passe qui leur a été communiqué.

5.3.2. La sécurité

La mise en œuvre de restrictions d'accès, ainsi que la mise en œuvre plus globalement de la sécurité du réseau Internet, passe par un certain nombre d'outils et de techniques dont les plus fréquentes sont :

- le **firewall**, ou pare feu, composant matériel ou logiciel paramétrable qui permet de filtrer les accès à un réseau interne en fonction de l'adresse de la machine émetteur de la requête, de l'adresse de la machine destinataire, du type de service demandé...

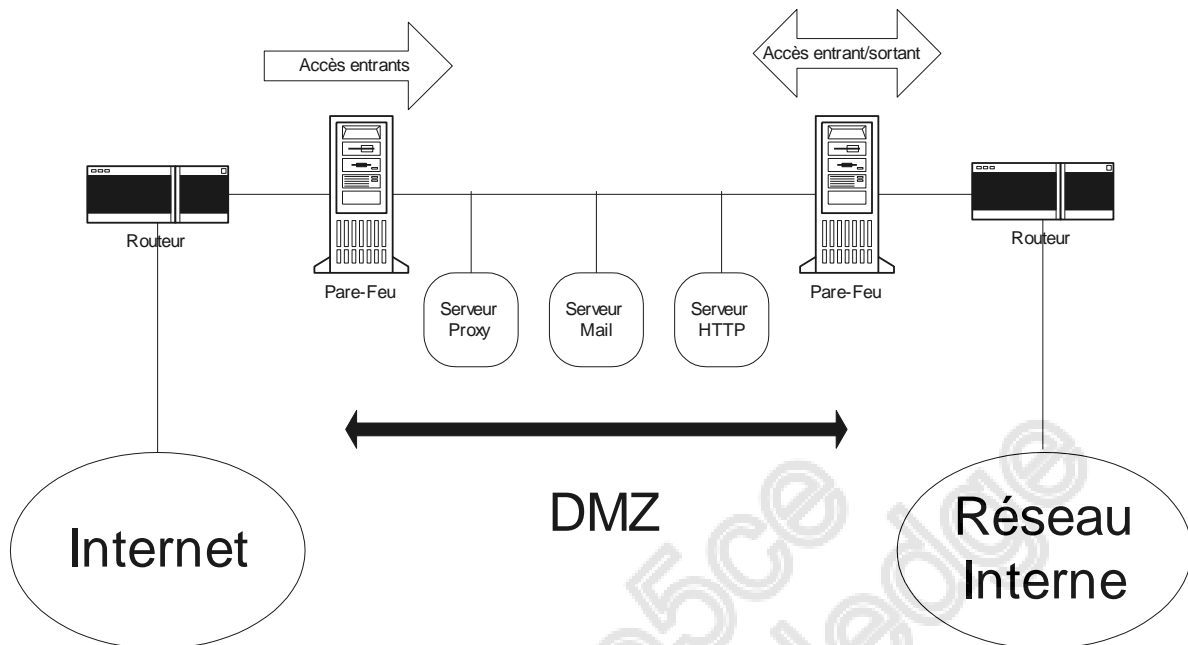
Exemple : interdire à un PC externe à l'entreprise l'accès à l'intranet.

- le **proxy-HTTP** qui, au-delà de ses fonctions de mutualisation de cache, permet d'interdire aux utilisateurs internes l'accès à certains sites et à certaines ressources Internet,

Exemple : interdire aux employés d'accéder à des sites à caractère non professionnel, ou susceptible de contenir des virus.

- la **DMZ**, la zone démilitarisée est une sous-réseau. Cette DMZ est isolée du réseau local par un pare-feu qui interdit toute connexion depuis la DMZ vers le réseau local. Elle est ensuite isolée du réseau Internet par un autre pare-feu qui ne laisse passer que certaines connexions d'un réseau vers l'autre.
- les techniques de **cryptage** qui garantissent la confidentialité, l'intégrité et/ou l'authenticité d'un message.

Exemple : masquer un code carte bleu saisi sur un site d'achat en ligne.



Un **cookie** est un fichier envoyé par le serveur Web. Il est stocké sur le poste utilisateur, et contient des informations permettant au serveur Web d'identifier ce même utilisateur à sa prochaine demande, et de connaître ses préférences. (Tant et si peu que l'utilisateur les ait renseignées). Le navigateur peut interdire le stockage de cookies. Cependant, le cookie n'est pas un programme enregistrant ou supprimant des informations du système de l'utilisateur. Il ne contient que ce que l'utilisateur a bien voulu donner.

Un **logiciel malveillant** (malware en anglais) est un logiciel écrit dans le but de nuire à l'informatique. Dans cette catégorie de logiciels, on trouve, les virus, les vers, les chevaux de Troie, les hoax (canulars), les spywares (logiciels espions)...

Le pare feu a aussi pour but de lutter contre ces logiciels malveillants.

Les antivirus sont des logiciels conçus pour neutraliser les logiciels malveillants ; les éditeurs les plus connus sont McAfee, Norton (Symantec), Microsoft (ForeFront).

5.3.3. Service Web

Un service Web (Web Service en anglais) est un programme permettant la communication et l'échange de données via le monde du Web entre systèmes hétérogènes.

Les services Web reposent sur un ensemble de protocoles et de standards gérés par le W3C, (Organisme international gérant la toile).

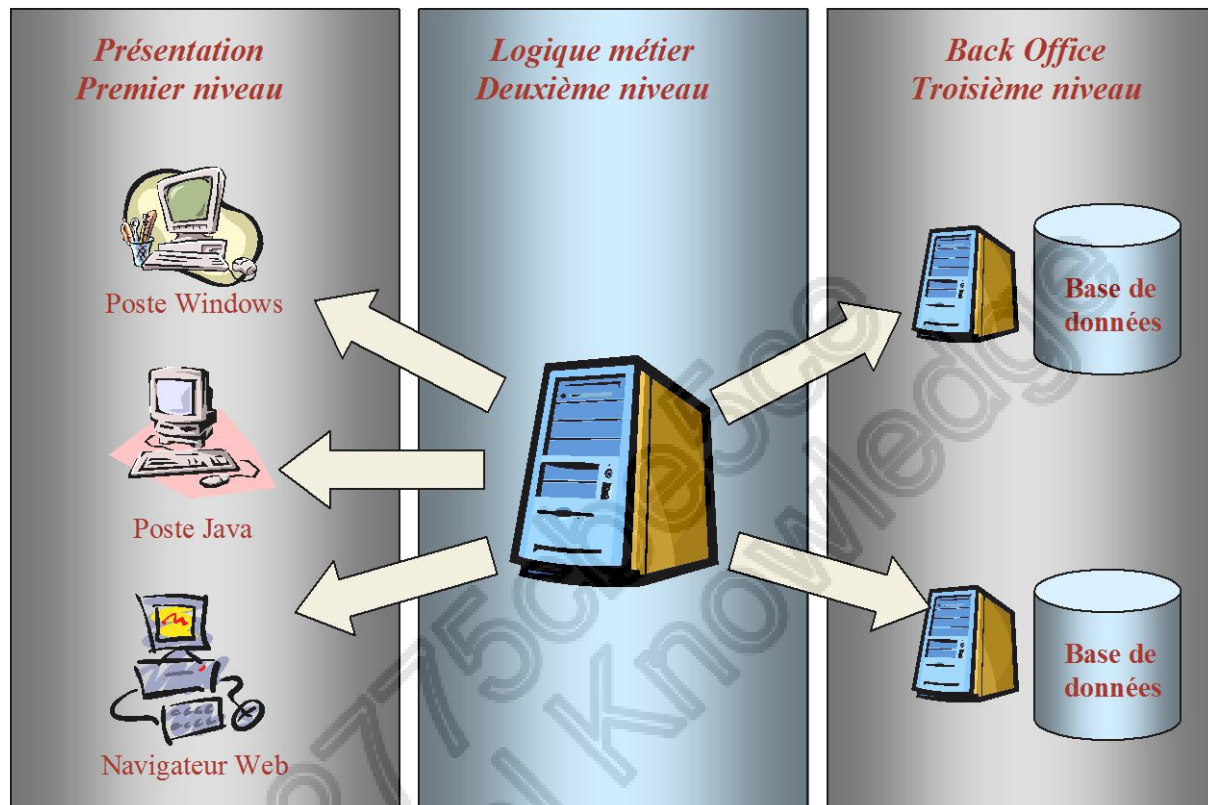
Le protocole de communication est **SOAP** (Simple Object Access Protocol).

La description du service (son nom, le type et le nombre des données échangées, sa localisation c'est-à-dire son URL, son protocole de communication) est prise en compte par un langage standard **WSDL** (Web Service Description Language).

Le service Web peut être enregistré dans un annuaire, permettant à tous les internautes de l'utiliser à distance, cet annuaire est **UDDI** (Universal Description Discovery and Integration).

6. ARCHITECTURE N-TIERS

6.1. Présentation



Les clients accèdent au **serveur d'applications**. Ce serveur fournit des services aux clients à partir de données récupérées sur un serveur de base de données. Cette architecture se nomme **architecture 3 tiers**. (3 niveaux de communication)

La simplicité avant tout :

- La réduction du nombre de protocoles utilisés
- La simplification des modèles de répartition
- Le modèle 3-tiers séparant la présentation, la logique métier et la persistance des données.

Le serveur d'applications étant capable de communiquer avec d'autres serveurs, il n'y a donc plus de limite d'échange et l'on arrive à des **architectures n-tiers**.

6.2. Les enjeux

Aujourd'hui, les applications doivent combiner :

- L'existant sous la forme de base de données, d'applications traditionnelles; on parle du système d'information de l'entreprise (EIS)
- De nouvelles fonctions de traitement pour offrir des services de qualité à un très large éventail d'utilisateurs

Les nouveaux services doivent répondre à plusieurs critères de **qualité**.

Les services doivent être :

- **disponibles** pour pouvoir être intégrés dans d'importants environnements de gestion
- **sécurisés** pour sauvegarder l'intégrité des données de l'entreprise
- **fiables et évolutifs** pour faire face aux évolutions technologiques

Les services sont organisés dans des architectures distribuées multi-niveaux (multi-tiers) :

- Les clients en frontal pour la présentation de l'information
- Les données et les applications traditionnelles en back end (EIS Enterprise Information System)
- De plus en plus, au niveau intermédiaire (middle-tiers), entre le client et l'EIS, un ensemble de modules de traitements qui réalisent la plus grande partie du travail

7. ARCHITECTURES ORIENTEES SERVICES (SOA)

7.1. Présentation

Une architecture orientée services (notée **SOA** pour Services Oriented Architecture) est une architecture logicielle **s'appuyant sur un ensemble de services** simples.

L'objectif d'une architecture orientée services est donc de **décomposer** une fonctionnalité en un ensemble de fonctions basiques, appelées services, fournies par des composants et de décrire finement le schéma d'interaction entre ces services.

L'idée sous-jacente **est de cesser de construire** la vie de l'entreprise **autour d'applications** pour faire en sorte de construire une architecture logicielle globale décomposée en services **correspondant aux processus métiers** de l'entreprise.

Service

Brique fonctionnelle accessible via une prise banalisée ; la prise étant matérialisée par un contrat.

Un service est donc une façade qui se positionne devant le composant pour le standardiser. Un service est une fonction qui reçoit des messages et les reconstitue après un traitement

Le composant sous-jacent est isolé du monde extérieur et peut ainsi être modifié à volonté tant que son contrat reste identique

Processus

Ensemble de plusieurs activités reliées les unes aux autres pour atteindre un objectif, généralement dans un contexte organisationnel.

Consommateur de service/processus

Entité logique utilisant un service au travers de son contrat.

SOA

Principe de construction architectural du système d'informations à partir de services/processus et de consommateurs de services/processus couplés de manière lâche.

**Les services/processus sont orientés métier :
ils exposent des fonctions de haut niveau**

7.2. Les enjeux

Les applications d'aujourd'hui ne sont plus monolithiques et doivent s'intégrer harmonieusement dans le système d'information de l'entreprise. Cela implique l'interaction avec l'existant (systèmes, plates-formes et applications), et une ouverture vers une réutilisation future des nouveaux modules fonctionnels ou techniques.

Les axes majeurs de la SOA sont :

- La **réutilisation et la composition**, permettant le partage de modules entre applications et les échanges inter-applicatifs ;
- La **pérennité**, qui implique notamment le support des technologies existantes et à venir ;
- L'**évolutivité**, car toute application est vivante, a une certaine durée de vie, peut se voir greffer de nouveaux modules et doit pouvoir répondre aux nouveaux besoins fonctionnels ;
- L'**ouverture et l'interopérabilité**, pour partager des modules applicatifs entre plates-formes et environnements ;
- La **distribution**, pour pouvoir utiliser ces modules à distance et les centraliser au sein de l'entreprise par exemple ;
- La **performance**, avec en priorité l'accent mis sur la montée en charge.

La notion de Service repose sur quatre fondamentaux :

- Support d'interfaces basées sur les standards du marché
- Séparation de l'interface et de l'implémentation
- Neutralité technologique
- Découverte dynamique du producteur par le consommateur

8. LE CLOUD COMPUTING

8.1. Présentation

Le cloud computing ou l'informatique dans les nuages se définit comme :

- Une fourniture de services et d'applications informatique en ligne accessibles partout, à tout moment et depuis n'importe quel terminal (smartphone, téléphone portable, ordinateur ou tablette).

Le cloud permet :

- De partager chez un fournisseur de Cloud, une infrastructure, une solution application ou une plateforme à toute personne qui en fait la demande à partir d'un site internet (portail).

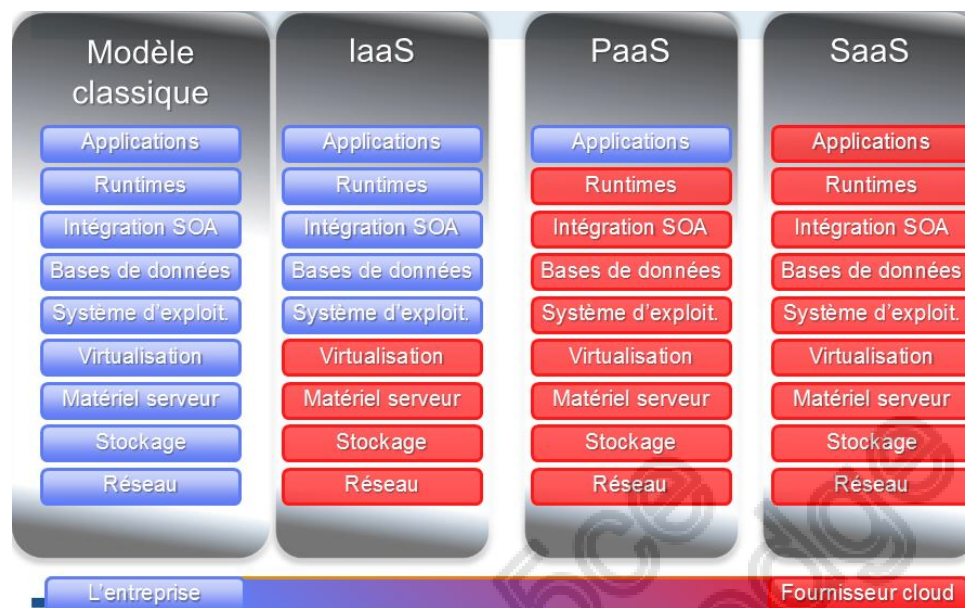
Les caractéristiques du cloud sont :

- Un libre service à la demande
- Une élasticité rapide
- Un accès réseau, des clients variés
- Une mise en commun des ressources
- Un service mesuré et une facturation à l'usage

Les modèles informatiques du cloud sont :

- IaaS (Infrastructure as a Service), juste l'infrastructure (les machines)
- PaaS (Platform as a Service), l'infrastructure + OS + outils
- SaaS (Software as a Service), les logiciels prêt à l'emploi

Dans ces différents modèles, entre l'entreprise et le fournisseur qui fournit quoi :



8.2. Les enjeux

Tant en termes de technologies que d'usages, le cloud computing répond aujourd'hui aux enjeux **d'agilité**, de **flexibilité**, de **productivité** et de **compétitivité** des organisations.

Les entreprises ont aujourd'hui pleinement conscience des bénéfices qu'elles peuvent tirer du cloud. La question qu'elles se posent désormais n'est plus ni pourquoi ni quand l'adopter, mais bien comment.

Alors que les **entreprises** migrent un nombre croissant de processus et de fonctions dans le cloud, elles se retrouvent souvent confrontées à un niveau de complexité élevé, pas toujours anticipé. Elles doivent particulièrement se préparer en termes **de gestion et de sécurité** des données, de **gouvernance**, **d'intégration** des systèmes, **d'impact financier**, ou encore de coordination entre les différents fournisseurs de cloud.

Du côté des **fournisseurs** de solutions, d'autres types d'enjeux existent : élaboration de business plan, financement, évaluation, contrôle interne, etc. Le développement soutenu du marché leur impose de s'adapter rapidement, soulevant également des interrogations relatives à la gestion de leur croissance et à leurs investissements. Les principaux fournisseurs et leurs produits sont Amazon Web Services, Microsoft Azure, IBM Soft Layer et Google.

Si le cloud est apporteur de belles opportunités pour les entreprises, les utilisateurs comme les fournisseurs doivent **être conscients des risques et des défis** qui y sont associés, afin de pouvoir y faire face.

CHAPITRE IV

LES COMPOSANTS D'UNE APPLICATION

Objectif : Décrire les composants d'une application, les évolutions de chacun durant ces dernières décennies. Comprendre les environnements de développement, d'exécution les outils nécessaires à la création d'une application.

1. LE MODELE EN COUCHE

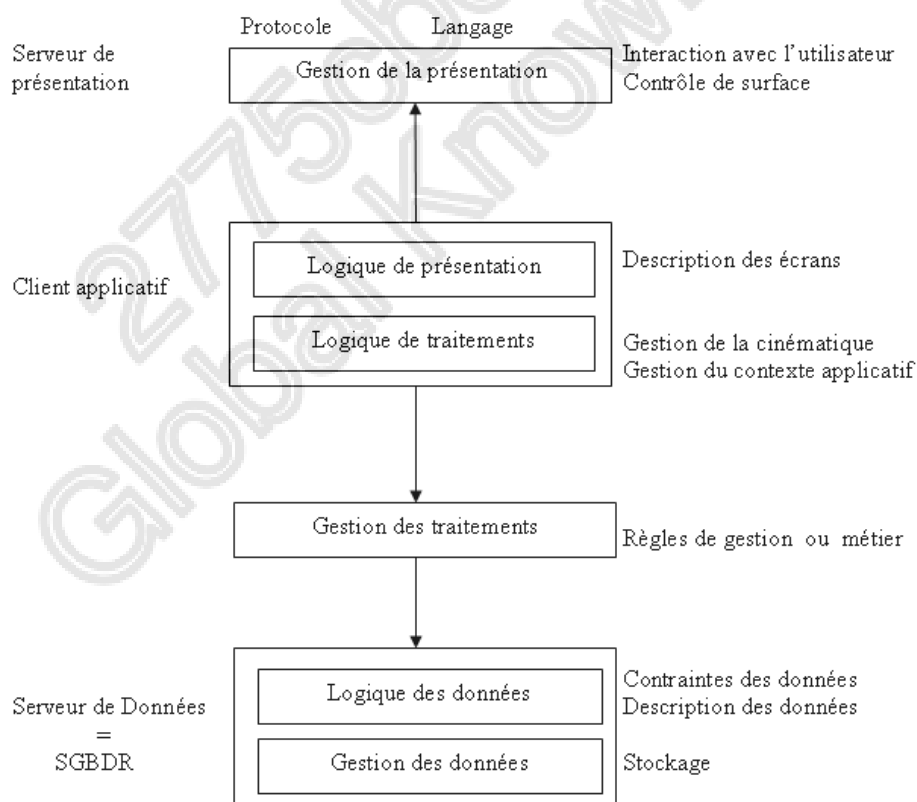
1.1. Présentation

Avec notre chapitre des architectures distribuées, nous avons noté qu'en Client/Serveur, une application était souvent présentée en 3 couches : Présentation – Traitement – Donnée. Nous avons pu voir aussi, que ce découpage correspondait à des architectures dites 3 tiers. Nous allons zoomer sur certains points pour mieux comprendre les mises en œuvre d'aujourd'hui.

Dans une application informatique, nous pouvons identifier **six couches logiques**, chacune ayant des rôles et des objectifs différents, l'application étant **monolithique ou découpée** pour sa diffusion sur une architecture distribuée.

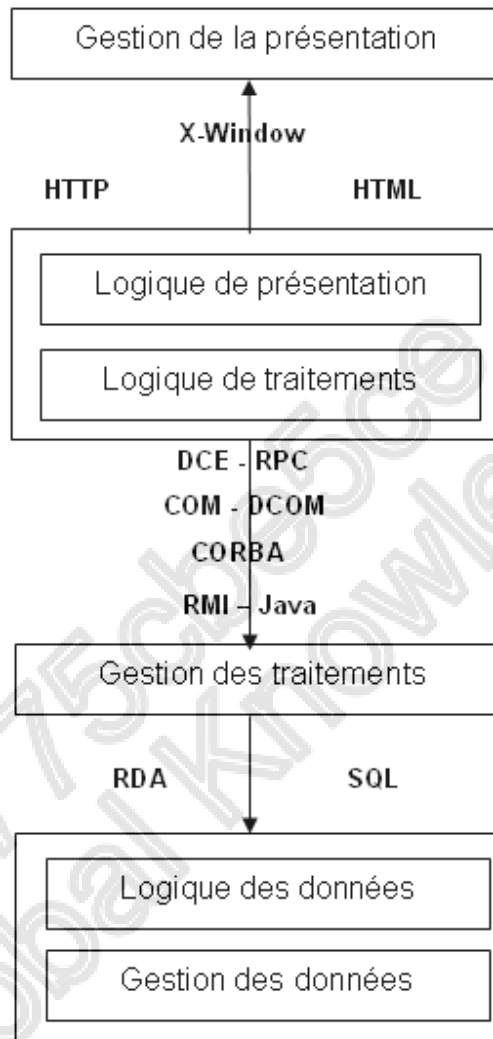
Ce modèle en couche est la base de la compréhension des nouveaux paradigmes de distributions.

A chaque couche, correspondra un protocole de communication et un langage d'échange entre les couches.

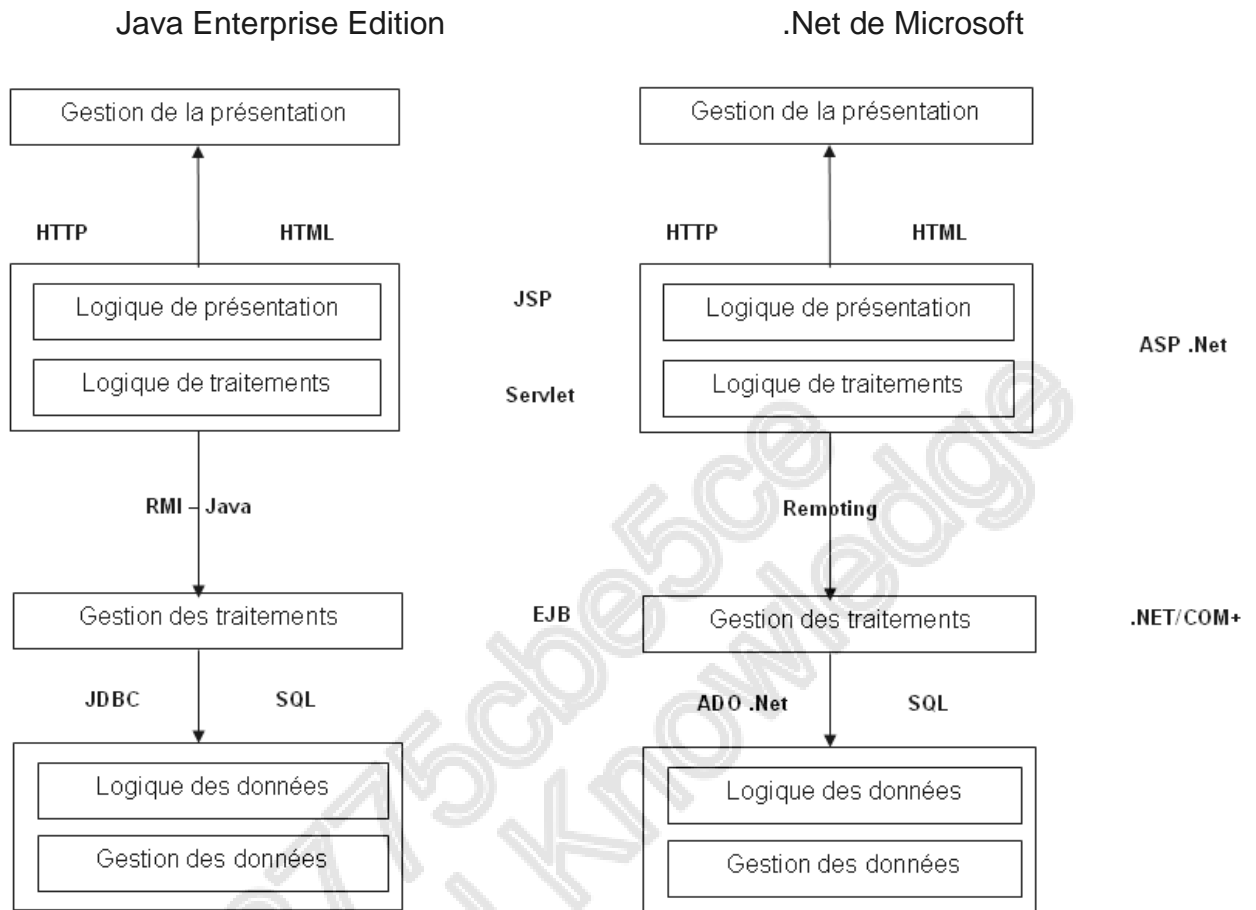


1.2. Les principaux modèles

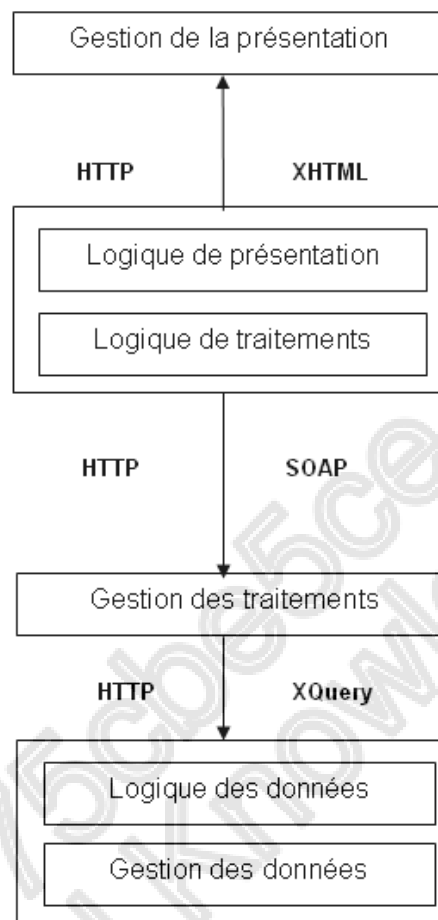
1.2.1. Modèle en couche Client / Serveur



1.2.2. Modèle en couche des plates-formes Java EE et .Net



1.2.3. Modèle en couche XML



2. LA COUCHE DONNEES

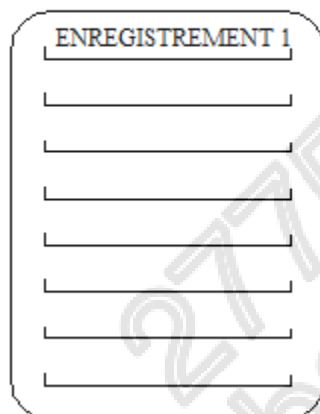
2.1. Concepts de base

Outre le support physique utilisé pour le stockage des données, celles-ci sont logiquement regroupées dans des **fichiers** ou des **bases de données**.

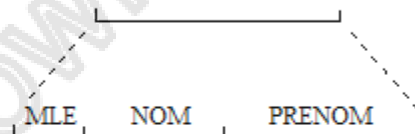
Un fichier ou une base de données correspond à un ensemble fonctionnellement cohérent d'informations (fichier des employés ou base des produits par exemple).

Au sein d'un fichier par exemple, chaque occurrence d'une entité fait l'objet d'un **enregistrement**, lui-même découpé en **rubriques**.

FICHER DES EMPLOYES



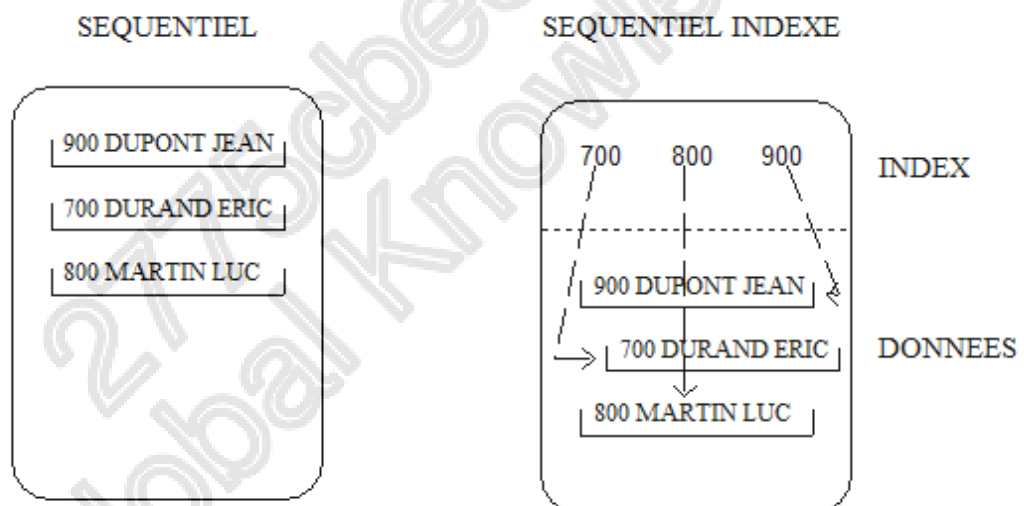
ENREGISTREMENT 1



2.2. Les systèmes de gestion de fichiers

Jusqu'à la fin des années 60, le seul ensemble de données connu est le fichier. L'accès aux informations par le développeur y est peu dissocié de l'ordonnancement de celles-ci sur le support de stockage. Deux organisations de fichier coexistent principalement :

- les **fichiers séquentiels**, dans lesquels les enregistrements sont physiquement implantés dans l'ordre chronologique de stockage. L'accès aux informations s'effectue séquentiellement. Cette organisation est à rapprocher de celle des cassettes audio,
- les **fichiers séquentiels indexés**, dans lesquels les enregistrements sont physiquement implantés dans un ordre quelconque, mais sont accessibles soit séquentiellement soit directement par l'intermédiaire d'une rubrique définie comme la **clé d'accès**, grâce à un index, véritable table des matières du fichier. Cette organisation est assimilable à celle d'un disque compact audio.



2.3. Les systèmes de gestion de bases de données (SGBD) et le langage SQL

À partir des années 60, les besoins de stockage d'informations plus complexes entraînent l'apparition des **bases de données**. Le stockage logique de l'information y est totalement indépendant de l'implantation physique des données. Au-delà de la conservation stricte des données, les bases de données permettent la traduction de liens entre enregistrements, de vues logiques ensemblistes...

Le logiciel permettant de gérer les données stockées dans une base de données est appelé un **SGBD** (Système de Gestion de Base de Données).

Les bases de données ont connues quatre évolutions majeures :

- hiérarchiques (IBM - 1960),
- réseaux (CODASYL - 1970),
- relationnelles (IBM - 1980).
- Big Data, début des années 2000, même si ce terme apparait pour la première fois (en 1997)

Bien que le big data en soit à ses débuts, il bouleversera ces prochaines années votre travail. Il faut savoir que **les bases de données relationnelles** ont été massivement implantées pendant près de 30 ans, grâce en particulier aux avantages suivants :

- le langage **SQL** (Structured Query Language) est le langage standard d'accès aux données, quel que soit l'éditeur de la base de données,
- le modèle relationnel est basé sur la théorie des ensembles, plus simple que la théorie des graphes sur laquelle sont basés modèles réseaux et hiérarchiques,
- les bases de données relationnelles permettent de faire évoluer le schéma des données sans modifier les programmes existants,
- elles sont entièrement intégrées aux architectures réseaux et proposent de nombreuses fonctionnalités liées au développement et à l'exploitation d'applications client-serveur.

Exemple d'accès aux données d'un SGBD/R (SGBD Relationnel)

- Soit les tables : EMPLOYE (MLE, NOM, PRENOM, SEXE)
AFFECTATION (MLE, CODPRO, DATDEB)
DEPT (CODEPT, DESIGN, DG)

MLE [DECIMAL(10 , 0)]	NOM [CHAR(20)]	PRENOM [CHAR(20)]	SEXE [CHAR(1)]
25	LORENT	CATHERINE	F
53	BLASQUEZ ...	NICOLAS	M
54	HERNANDEZ ...	ANTOINE	M
55	MARCIER	JACQUES	M
56	GALLET	BRUNO	M
100	DUCHE	SYLVIE	F
103	SOUPINACIO ...	JESUS	M
218	FONTAINE ...	JEAN-PIERRE	M
273	CHICHE	GLADYS	F
286	ROUVRAIS ...	PHILIPPE	M
303	SALAMI	JUSTIN	M
304	ALPHANDERY ...	ALPHONSE	M
333	PONCHEL	VINCENT	M
633	BALLARD	GASTON	M
652	BIBER	ALBERT	M
672	DUVAL	CHRISTINE	F

MLE [DECIMAL(10 , 0)]	CODPRO [DECIMAL(10 , 0)]	DATDEB [DATE]
25	6	01/04/96
53	120	02/02/97
54	120	02/02/97
55	120	02/02/97
56	6	02/02/97
100	12	01/04/96

CODEPT [DECIMAL(10 , 0)]	DESIGN [CHAR(20)]	DG [CHAR(20)]
2	CONSEIL	DUPONT
3	INDUSTRIE	PEREZ
6	ETHNOS	ROQUE ...
12	FORMATION	MARTIN
120	COMPTABILITE	MIRAN

- Requêtes possibles en SQL :

- Liste de tous les employés de l'entreprise :

```
SELECT MLE, NOM, PRENOM  
FROM EMPLOYE
```

- Liste des employées de l'entreprise :

```
SELECT MLE, NOM, PRENOM  
FROM EMPLOYE  
WHERE SEXE = 'F'
```

- Liste des employés du département de code 12 :

```
SELECT EMPLOYE.MLE, NOM, PRENOM  
FROM EMPLOYE  
JOIN AFFECTATION ON EMPLOYE.MLE = AFFECTATION.MLE  
WHERE CODPRO = 12
```

- Liste des employés du département dont le responsable est "MARTIN"

```
SELECT MLE, NOM, PRENOM  
FROM EMPLOYE  
WHERE MLE IN  
  (SELECT MLE  
   FROM AFFECTATION  
   WHERE CODPRO IN  
     (SELECT CODEPT  
      FROM DEPT  
      WHERE DG = 'MARTIN'))
```

Principales bases de données relationnelles :

ORACLE, société Oracle,

La famille **DB2** ou **UDB** (Universal DataBase) d'IBM,

SQL SERVER, **ACCESS** chez MICROSOFT,

MySQL, base de données open source rachetée par Sun Microsystems en 2008, puis par Oracle. Son concepteur n'ayant pas voulu suivre Oracle, a reproduit sa base sous le nom de : **MariaDB**.

PostgreSQL, base de données open source, dérivée de la base Ingres. **Ingres**, détenue jusqu'en 2005 par Computer Associates, est passée libre en 2004

2.4. Des bases relationnelles vers la Business Intelligence

Les bases de données relationnelles ont été conçues pour enregistrer rapidement des données de production, en évitant la redondance d'information afin de mettre à jour rapidement les données : telles que les commandes d'un client, la gestion de stock, les mouvements comptables, l'enregistrement des données de paye... Ces bases contiennent donc d'importantes données qui peuvent être analysées à des fins statistiques, marketing, voire de prises de décisions.

Le **traitement analytique en ligne** (en anglais *online analytical proccession*, OLAP) est un type d'application informatique orienté vers l'analyse sur-le-champ d'informations selon plusieurs axes, dans le but d'obtenir des rapports de synthèse tels que ceux utilisés en analyse financière. Les applications de type *OLAP* sont couramment utilisées en informatique décisionnelle (nommée aussi la Business Intelligence), dans le but d'aider la direction à avoir une vue transversale de l'activité d'une entreprise.

Ce type d'application s'oppose au traitement de transactions en ligne, on a donc copié les données de production du monde SQL dans des bases de données de type OLAP, ou de type SGBD.

2.5. De la multitude des données vers le MDM

En général, une entreprise dispose de plusieurs bases de données rangées chacune au sein d'un système d'information ou derrière une application métier particulière (gestion comptable, ventes, gestion des ressources humaines, serveur de suivi de production, etc.). C'est notamment le cas pour des structures ayant opté pour une approche best-of-breed à l'inverse d'une politique technologique articulée autour d'un progiciel de gestion intégrée ou ERP en anglais.

Best-of-breed = littéralement, le meilleur de sa catégorie. Se dit d'une solution logicielle prétendant offrir des fonctions avancées sur un segment de marché bien délimité. Avec ce type de solution, les informations du client, par exemple, peuvent se retrouver dupliquées, dans la base de la gestion commerciale, dans celle de la comptabilité, celle de la BI... On ne sait plus qui est maître de la donnée, qui la met à jour, comment les autres sont informés d'une mise à jour.

Il devient donc nécessaire de gérer la qualité et la cohérence des données contenues dans les bases et systèmes de l'entreprise, telle est la vocation, de la méthode applicative nommée MDM : Master Data Management.

2.6. Le Big Data

Big data est le terme utilisé pour désigner des données volumineuses difficiles à manipuler par les outils traditionnels, tels que les bases de données relationnelles. Ce nouveau concept nécessite de revoir :

- le stockage des données,
- la gestion de recherche,
- le partage de l'information entre de multiples utilisateurs,
- sans oublier la restitution
- et l'analyse des données.

Certains bouleversements sont donc à prévoir dans les prochaines années.

On parle alors de **NoSQL** qui est un Système de Gestion de Base de Données (SGBD), qui ne se base pas sur l'architecture traditionnelle du monde relationnel (avec le langage SQL).

Le Big data, prend de l'ampleur dans les années 2010, avec les géants d'internet, tels que Amazon, Google, Facebook, il nécessite donc de nouveaux outils. :

Cassandra est un SGBD NoSQL, initié par Facebook, projet de la fondation Apache qui permet de gérer des quantités massives de données **réparties sur plusieurs serveurs**, en assurant une **disponibilité maximale** de ces données.

CouchDB, MongoDB sont des SGBD NoSQL, orientés documents, conçus pour le Web afin d'être répartis sur plusieurs serveurs.

De nouveaux modèles de représentation voient le jour et permettent de **garantir les performances** sur les volumétries en jeu. Ces technologies, dites de Business Analytics & Optimization (BAO) permettent de gérer des bases massivement **parallèles**. Des patrons d'architecture "Big Data Architecture Framework (BDAF)" sont proposés par les acteurs de ce marché comme **MapReduce** développé par Google et utilisé dans le framework **Hadoop**. Avec ce système les **requêtes sont séparées et distribuées** à des nœuds parallélisés, puis exécutées en parallèles (map). Les résultats sont ensuite rassemblés et récupérés (reduce).

Etant donné que l'on parle de plusieurs serveurs pour répartir et stocker les données, avec le big data, on entre dans le monde du **cloud computing**. L'accès se faisant via le réseau.

3. LA COUCHE PRESENTATION

3.1. Les interfaces mode caractère

« Le terme d'interface utilisateur regroupe tous les composants logiciels qui se rapportent à la présentation des informations et des traitements sur l'écran de l'utilisateur final ».

Les **interfaces mode caractère** ne proposent qu'une présentation sommaire d'où le dessin est absent. Seuls les caractères accessibles au clavier peuvent y figurer. L'évolution de la technologie aidant, elles ont récemment intégré la gestion de la couleur et des menus déroulants.

```
Mon Jan 23 15:06:19 EST 1995          Copyright A.CASSAR  30/06/94

-----
! GESTION DES ENVIRONNEMENTS !
-----

1. PREPARATION ENVIRONNEMENT C
2. PREPARATION ENVIRONNEMENT UNIX
3. PREPARATION ENVIRONNEMENT SQL
D. DESTRUCTION ENVIRONNEMENTS STAGIAIRES
L. LISTE REPERTOIRES STAGIAIRES
M. LANCEMENT APPLICATION UNIX
O. RECHARGEMENT BASE ORACLE
P. LISTE ARBORESCENTE DES PROCESSUS
R. LISTE ARBORESCENTE DES REPERTOIRES
S. SURVEILLANCE CONNEXION UTILISATEUR
T. ARRET DU VEILLEUR
X. FIN

VOTRE CHOIX :
```


3.2. Les interfaces mode graphique

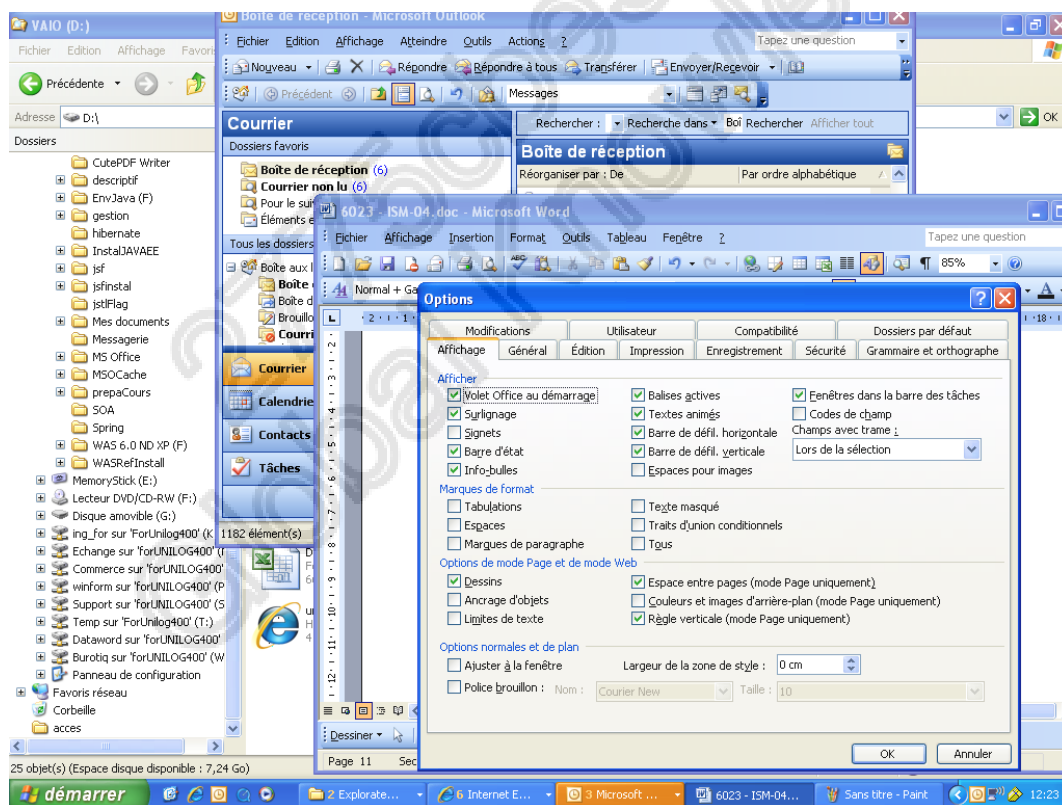
Les **interfaces mode graphique** permettent l'intégration de dessins et d'images sur l'écran de l'utilisateur. Elles diffèrent surtout des précédentes par la symbolisation des traitements à effectuer.

La présentation est gérée par des objets spécifiques :

- fenêtres,
- icônes,
- ascenseurs,
- .../...

Elle permet d'accéder à des fonctionnalités conviviales :

- couper ou copier / coller,
- ouverture de plusieurs tâches en parallèle.

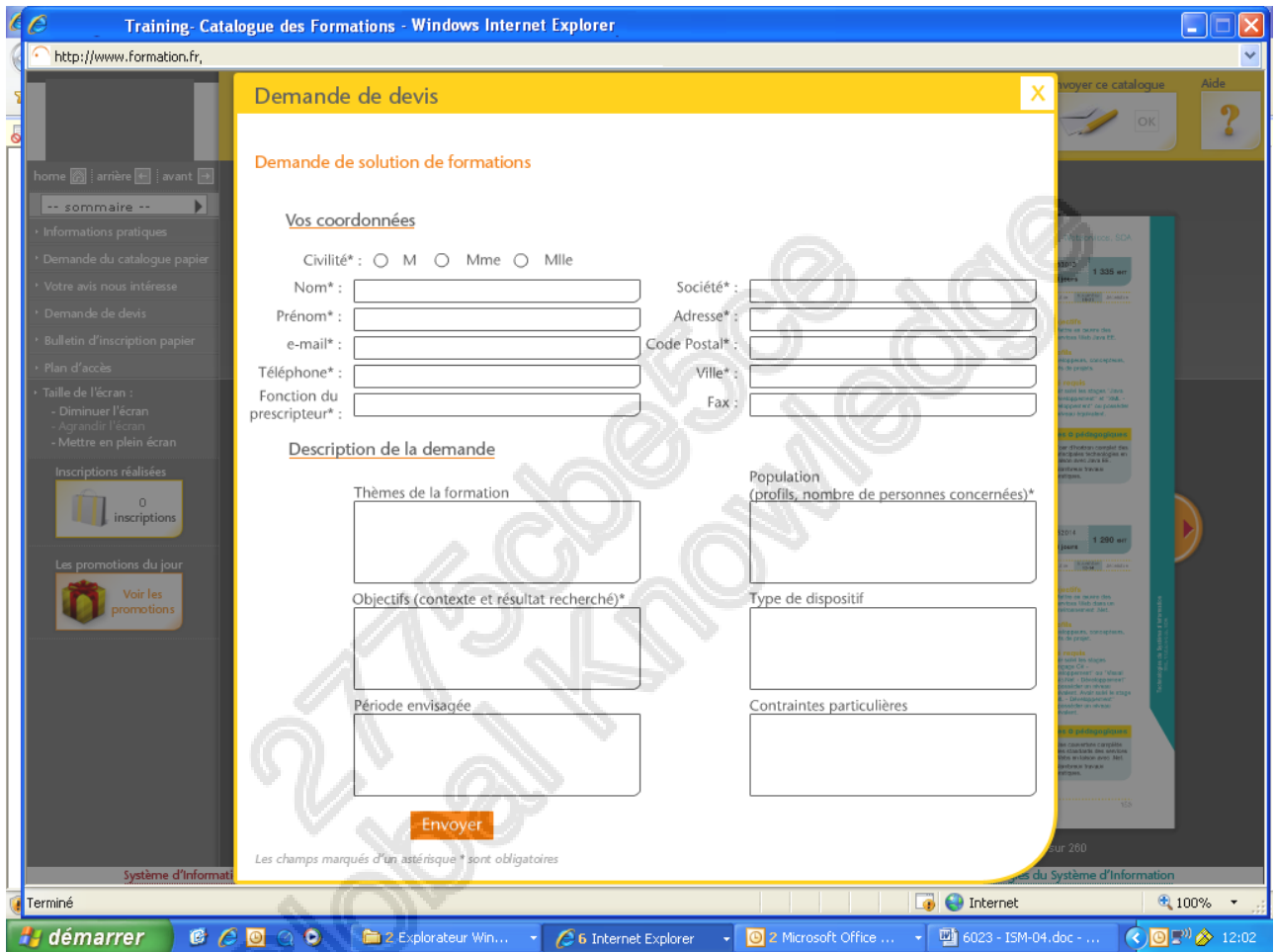


Lorsque l'application permettant cet affichage s'exécute sur le poste de l'utilisateur, on parle de **client lourd**, d'application de bureau : **Desktop**. Ce type d'application est autonome reste en local (contrairement aux applications web !).

3.3. Les interfaces Web

Les interfaces Web sont des interfaces graphiques.

Lorsqu'il faut faire saisir des informations à l'utilisateur, on retrouve des présentations simples.



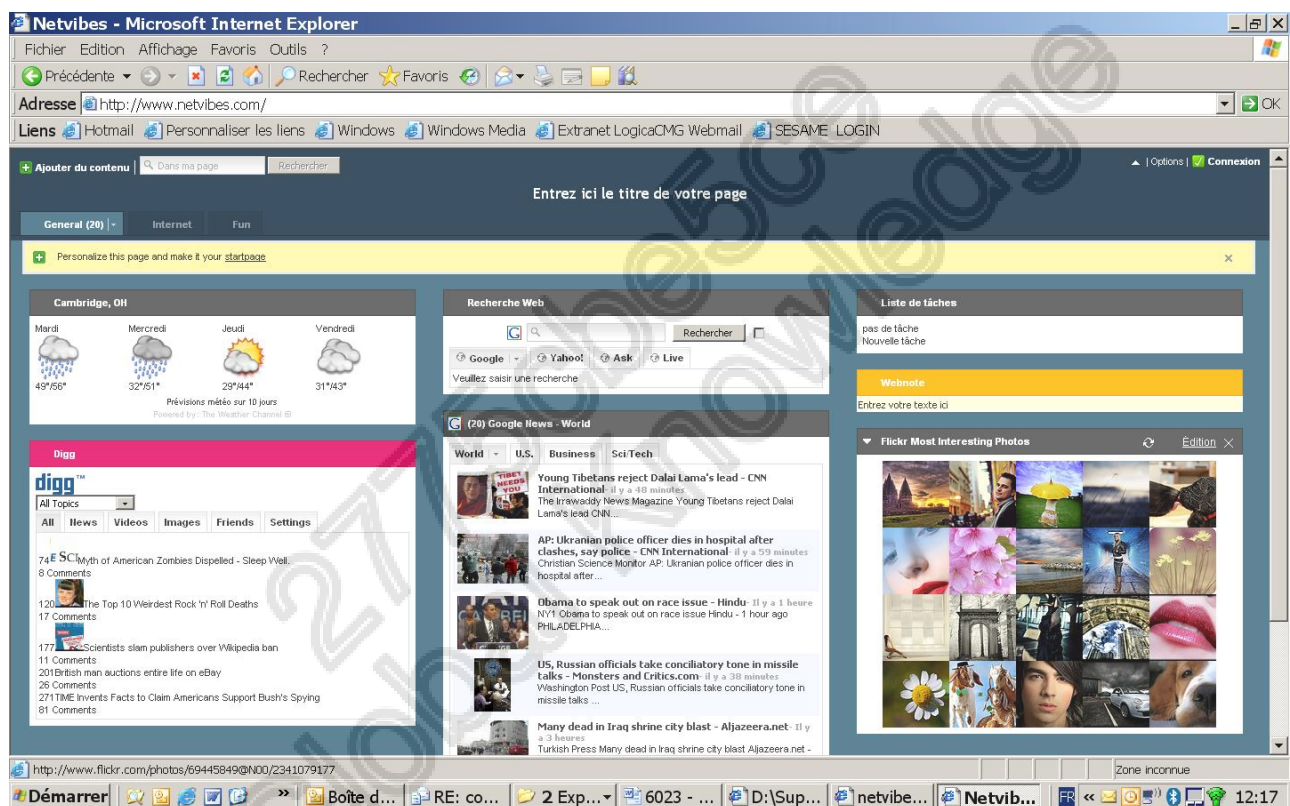
Lorsque l'application permettant cet affichage s'exécute sur le serveur et que le poste de l'utilisateur n'affiche que le résultat, on parle de **client léger**.

3.4. Les interfaces riches

Les interfaces Web, ne sont pas très riches en présentation, mais intéressantes, car il n'y a presque rien à installer sur le poste de l'utilisateur. Les interfaces graphiques sont très conviviales pour les utilisateurs. On a pris le meilleur de ces deux architectures et créé le **client riche**.

L'interface utilisateur est une interface graphique riche (glisser-déposer, onglets, menus déroulants, ...) identique à un client lourd.

A savoir qu'il existe des interfaces riches en client lourd.



3.5. Les interfaces mobiles

Les tablettes et les smartphones ont eux aussi apporté des nouveautés dans la présentation aux utilisateurs. De nouveaux affichages ont vu le jour, souvent « collé » au système d'exploitation sur lequel l'application est présentée.

A l'ère de l'**Internet Mobile**, quelles que soient vos préférences en termes de marque et de modèle, en matière de smartphones, on peut considérer qu'il y a un avant et un après Iphone. Dès sa sortie, le smartphone d'Apple a fait un véritable carton... et a boosté la pratique de l'internet mobile ! Les autres fabricants lui ont tous emboîté le pas avec plus ou moins de succès !

Les Google Phones se sont distingués comme de véritables bijoux multimédias. Ces mobiles, équipés du système d'exploitation de Google « **Android** » se sont imposés comme de véritables smartphones tournés vers l'expérience Internet avec des browsers performants qui permettent une navigation plus rapide et plus fluide.

L'offre de Microsoft, permet aujourd'hui d'avoir le **rendu visuel commun** aux Tablettes (surface MS &), aux téléphones, aux PC.



La tendance est déjà bien amorcée, depuis le lancement du premier Iphone en 2007, on peut imaginer que d'ici à 5 ans, la quasi-totalité de l'offre mobile sera équipée d'**écrans tactiles**.

Plus rapide et plus intuitif, le tactile remporte les suffrages des utilisateurs ! Par ailleurs, les systèmes d'exploitation des mobiles, de plus en plus performants et de plus en plus poussés pourraient permettre aux smartphones de devenir de véritables mini-portables intégrant notamment des clients mails, la gestion de documents bureautiques ou l'accès à des applications en cloud-computing. Si certains smartphones intègrent tout ou partie de ces fonctionnalités, on peut imaginer que ces usages vont peu à peu se généraliser.

3.6. Le portail

On appelle "**portail d'entreprise**" une plate-forme donnant accès à des données de l'entreprise ainsi qu'à des ressources du système d'information regroupées au sein d'une **interface unique**. Il s'agit d'une interface Web.

Le *portail d'entreprise* est ainsi la porte d'entrée vers les données du système d'information de l'entreprise pour l'ensemble du personnel et éventuellement les partenaires.

L'enjeu du portail est de chercher à centrer l'utilisateur au sein du système d'information.

Principaux fournisseurs de portail d'entreprise :

- **WebSphere Portal** , d'IBM,
- **WebLogic Portal** , d'Oracle,
- **JBoss Portal**, produits Open Source,
- **Office SharePoint Server** de Microsoft, MOSS.



4. LA COUCHE TRAITEMENT

4.1. Les langages de programmation

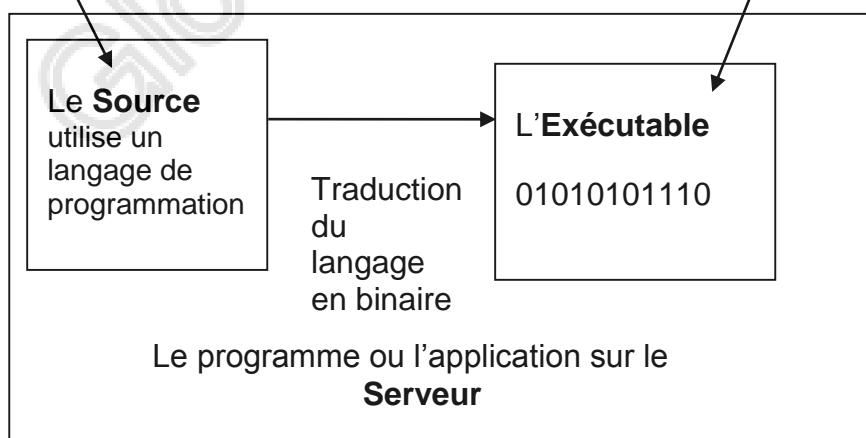
L'exécution d'un traitement par un ordinateur nécessite de décrire ce traitement à l'aide d'un langage approprié. Tout programme existe sous deux états :

- le **source** décrit le traitement à l'aide du langage de programmation choisi, explicite pour l'informaticien mais non pour la machine,
- l'exécutable (module chargeable ou load-module) résulte de la traduction du source en binaire, seul format exploitable par la machine. Le **binaire** est constitué d'une succession de " 0 " et de " 1 " (chiffres binaires, binary digits ou bits) qui traduisent l'absence ou la présence d'influx électriques interprétables par le processeur. L'exécutable est obtenu via un mécanisme nommé **compilation**.



L'Informaticien

L'Utilisateur



4.2. Les générations de langage

Le langage **assembleur** permet d'attribuer un code mnémonique à chaque série de bits correspondant à une fonctionnalité du processeur. Il nécessite une phase de traduction (l'assemblage) pour être exploitable par le processeur. L'assembleur reste lié au processeur et ne permet donc pas d'écrire des programmes portables au niveau du source sur des machines hétérogènes.

Les **langages de commandes**, tel Shell ou DOS utilisés principalement par les systèmes d'exploitation et les **langages de Script**, JavaScript, Ruby, VBScript étant les plus connus, ne sont pas compilés, mais interprétés par le système sur lequel ils s'exécutent. Ils n'ont donc pas les fonctionnalités nécessaires à la conception d'un logiciel.

Les **langages évolués** datent de la fin des années 50. Ils offrent au programmeur une sémantique plus lisible et surtout sont indépendants du processeur. Ils autorisent donc l'écriture de programmes portables au niveau du source. La phase de traduction (compilation) est beaucoup plus longue que l'assemblage. Les langages évolués sont ciblés vers le développement d'applicatifs spécifiques :

Cobol, le plus répandu, est un langage orienté gestion,

Pascal, est utilisé quasi-uniquement dans le monde universitaire,

PL/1, est un langage d'IBM, utilisé sur les grands systèmes à des fins de gestion.

RPG, (**GAP** en français) est positionné par IBM comme PL/1 mais pour le System i,

Fortran, est un « vieux » langage scientifique, encore utilisé parfois en informatique industrielle,

C, a été conçu à l'origine pour des besoins systèmes (développement du système d'exploitation Unix) ; il est utilisé aujourd'hui principalement pour le développement d'applications client/serveur surtout sur des machines Unix,

C++, est la version objet du langage C et est couramment utilisé pour des développements sur micro-ordinateur,

Smalltalk est un langage purement objet utilisé dans le monde scientifique,

Ada, est un langage militaire,

Simula est destiné à des applications de simulation (balistique par exemple),

Basic est un langage historiquement destiné aux débutants, que l'on utilise à des fins professionnelles dans sa version Visual Basic pour des développements sur micro-ordinateur,

Java est un langage destiné aux développements en architecture distribuée et est particulièrement présent dans le monde Internet,

C#, de Microsoft, est destiné à concurrencer Java sur les développements client-serveur et Internet.

Objective C, d'Apple est destiné au développement d'applications pour iOS. En juin 2015, Apple annonce son remplaçant **Swift**

ABAP, est un langage propriétaire, faisant partie des logiciels SAP, sa version 4 est objet. Le 4 faisant aussi référence aux langages de 4^{ème} génération.

Certains de ces langages (Cobol, C, C++, Java) font l'objet d'une normalisation *via* l'ANSI (*American National Standard Institute*), dépendant directement de l'ISO.

Les **langages objet**, il s'agit d'une façon de développer qui impose des règles et une philosophie aux développeurs. Les concepts sont les mêmes pour tous les langages objets (C++, Java, C#, Objective C et Swift, Ruby...).

Les **langages de quatrième génération** (L4G), ne constituent pas, à proprement parler, des langages de programmation. Ce sont des environnements de développement, basés sur une syntaxe de langage évolué ou sur une syntaxe spécifique, et qui offrent des possibilités de génération de code source, en particulier la génération du programme pilotant l'interface graphique. Avec l'ère d'internet et de l'utilisation massive des langages objet, il y a de moins en moins de développement avec ces plateformes.

Visual Basic, VB.Net, Visual C++, Visual J++ de Microsoft, basés respectivement sur les langages Basic, C++ et Java. Ces langages sont intégrés dans **Visual Studio**.

PowerBuilder de Powersoft / Sybase, basé sur un langage spécifique,

DreamWeaver d'Adobe,

VisualAge C++, VisualAge Smalltalk, VisualAge Cobol, VisualAge Java, d'IBM, basés sur les syntaxes des langages évolués correspondants,

Delphi de Borland.

Forms d'Oracle.

Les **langages des Bases de données relationnelles**, SQL n'étant pas un langage procédural, on a dû créer des langages permettant de combiner des déclarations de variables, des alternatives, boucles, ... et des **requêtes SQL**.

PL/SQL d'Oracle,

Transact-SQL de Microsoft,

SQL PL d'IBM.

4.3. Les langages d'échanges de données

4.3.1. L'EDI

L'EDI (Echange de Données Informatisées – Electronic Data Interchange) est un système de messagerie informatique permettant aux entreprises de s'échanger des données de type commande, facturation.

Première mise en œuvre de la dématérialisation des documents comptables d'achat, de vente et de règlement.

L'EDI est empreint de nombreuses normes :

- sur le plan de la forme,
- du contenu des messages,
- des déclarations administratives,

à opérer avant toute coopération entre deux entreprises.

Des traducteurs permettent aux progiciels de gestion de l'entreprise de comprendre et de générer les messages EDI.

Techniquement, des messages de toutes sortes (commerciaux, logistiques, financiers...) transitent via un réseau entre partenaires. Ils sont rédigés dans une **syntaxe spécifique**. Un logiciel permet ensuite à chaque partie de traduire ces messages et d'intégrer directement les données à leur applicatifs évitant ainsi toute ressaisie et tout risque d'erreur.

Le langage **EANCOM** est conseillé par GS1 France (anciennement Gencod EAN France). C'est le seul qui permette aux entreprises du commerce et à leurs partenaires de communiquer partout dans le monde. Il s'appuie sur une structure et une liste de codes établie pour le langage des Nations Unies : **EDIFACT**, mais également sur les codes EAN qui doivent être parfaitement en place dans les entreprises.

Dans le monde de la Logistique, EANCOM est souvent associé au code à barre.



Dans le monde bancaire, EDIFACT (Échange de données informatisées pour l'administration, le commerce et le transport en anglais, *Electronic Data Interchange for Administration, Commerce and Transport*), est associé au protocole de communication **ETEBAC** (Échanges Télématiques Banque Client), par exemple pour les virements bancaires.

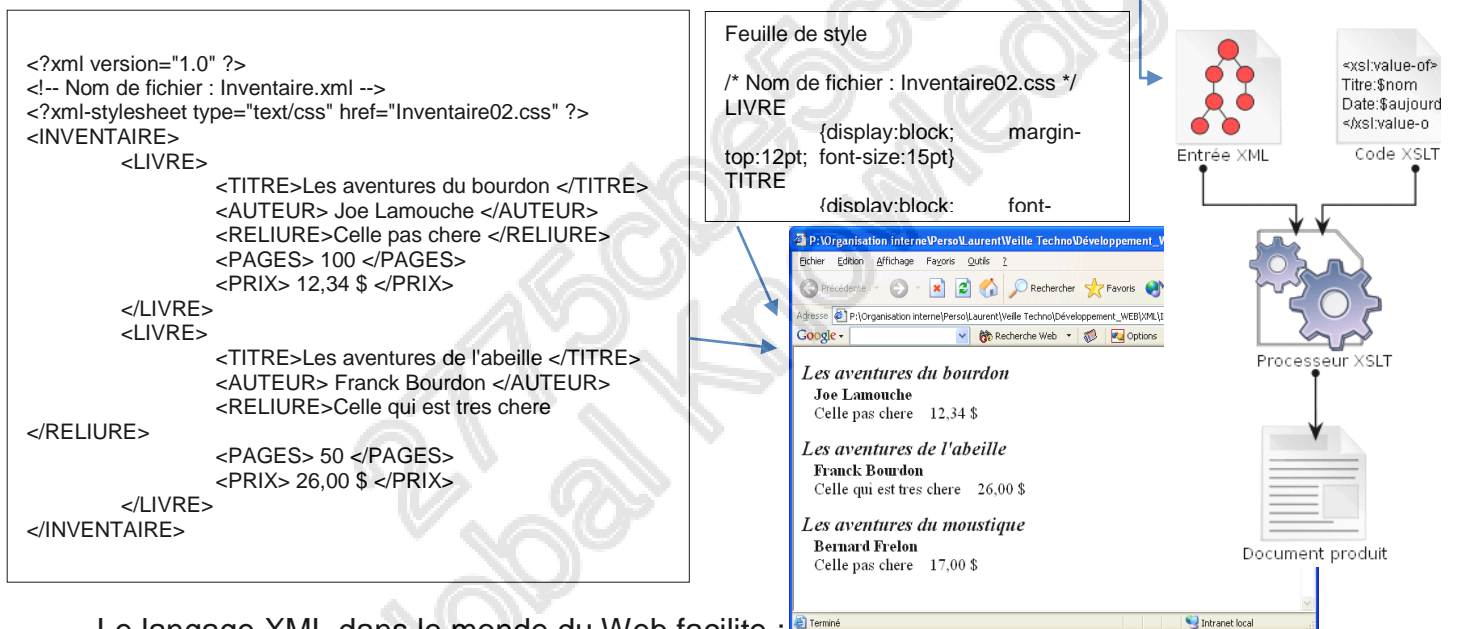
4.3.2. XML

Le standard XML eXtensible Markup Language est une recommandation de l'organisation W3C (World Wide Web Consortium) depuis février 1998.

XML a été élaboré pour publier des documents sur le Web. Il tend de plus en plus à servir de **format d'échange** entre **applications** et plus généralement entre **systèmes d'information**, sur des systèmes hétérogènes. XML permet de définir sa propre grammaire.

Afin de manipuler les données au format XML correctement et de la même manière, entre partenaires, on associe à XML une définition du document une **DTD** (Document Type Definition), ou un **schéma XML**.

XML gère les données et non leur représentation. La mise en page est prise en charge par **XSL** (eXtensible Stylesheet Language). On peut aussi se servir d'XML, pour générer différents documents comme des fichiers textes ou PDF



Le langage XML dans le monde du Web facilite :

- La recherche d'information
- L'intégration des données de sources variées
- La manipulation des données sur le poste client
- La création de vues multiples de données
- Les mises à jour incrémentales
- La publication multi-langues
- La navigation avancée
- La mise en œuvre des Web Services

XQuery est un langage de requête informatique permettant :

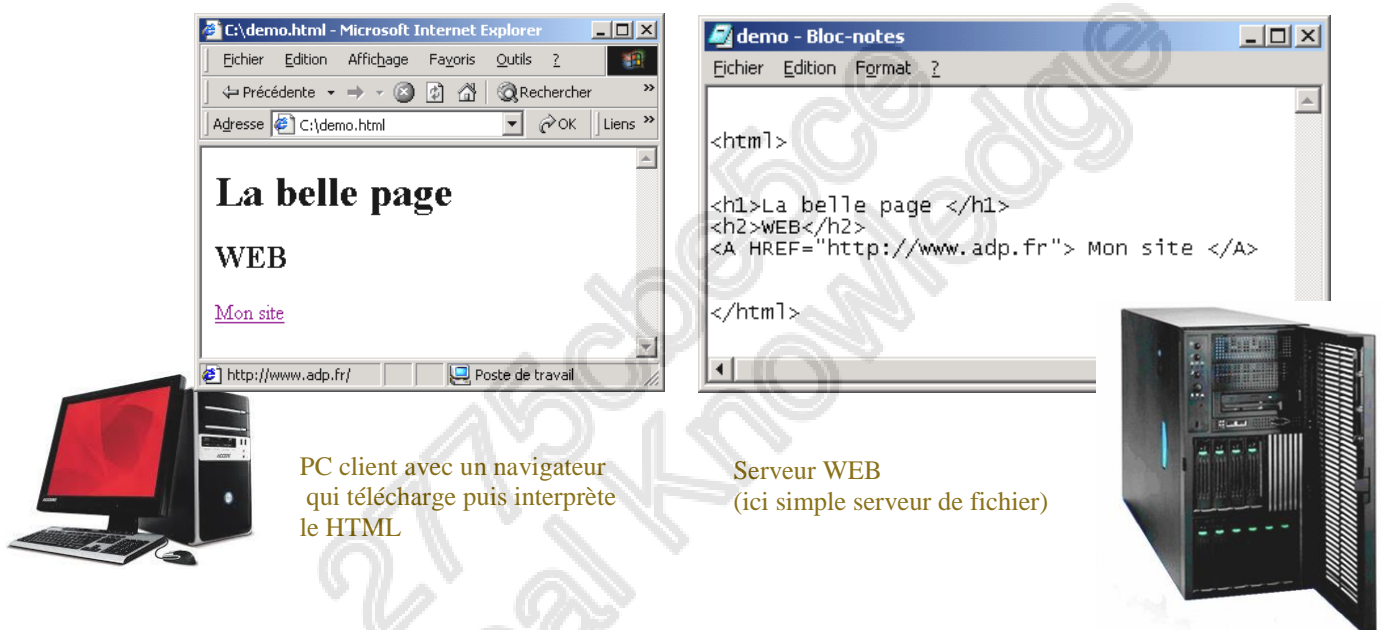
- D'extraire des informations d'un document XML.
- D'effectuer des calculs complexes à partir des informations extraites et de reconstruire de nouveaux documents ou fragments XML.

XQuery est aux documents XML ce que SQL est aux bases de données relationnelles.

4.4. Les langages du Web

Le premier langage de présentation de page Web a été **HTML** (Hyper Text Markup Language). Le langage **XHTML**, pour Extensible HTML est une reformalisation de HTML. Le HTML est la base de tout développement Web, la dernière révision du langage est la 5, d'où l'utilisation de HTML 5. Le langage HTML s'associe en parallèle à **CSS** (Cascading Style Sheets).

HTML permet de coder le **contenu** (titre, paragraphe, menu...) pendant que le **CSS** le **met en forme** (couleur, choix des polices de caractère, disposition des éléments ...).



Pour ajouter des contrôles de saisies, des calculs, des manipulations de dates, des mises en forme dynamiques de pages, on utilise d'autres technologies : le **Dynamic HTML** (DHTML), JavaScript.

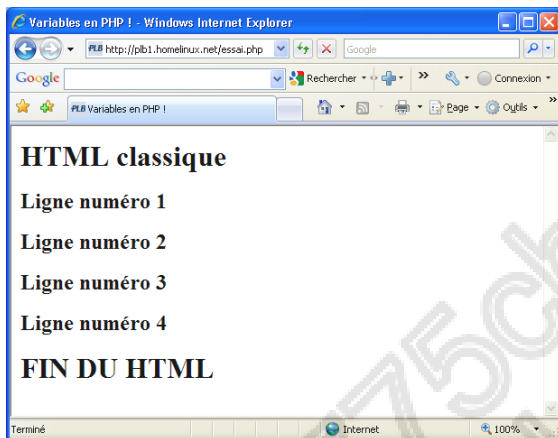
JavaScript est un langage interprété par le navigateur, ce langage dit langage de **script** (car il n'est pas compilé) ne pourra ni lire, ni écrire de données sur le poste local pour des raisons de sécurité.

La page Web est stockée sur le serveur pour être envoyée aux utilisateurs, on parle d'un site Web statique (Origine du Web).

Dans un site **Web dynamique**, la page est construite via un **programme** stocké sur le serveur. Il existe différentes technologies pour écrire ce programme :

- La technologie Java avec **JSP** (Java Serveur Page), .
- La technologie Microsoft avec **ASP.Net** .
- La technologie **PHP** (Hypertext Preprocessor).

Au début des sites Web dynamiques : les langages PERL et C ont été les plus utilisés, mais chaque serveur pouvait exécuter n'importe quel langage qu'il supportait. Ces programmes se nommaient des CGI, (common gateway interface). Chez Microsoft existait la technologie ASP.



```
<html>
<head>
  <title>Variables en PHP !</title>
</head>
<body>
<H1> HTML classique
<h2>
<?php
    for($i = 1; $i < 5; $i++)
        echo ' <p>Ligne numéro '.$i.'</p>'. "\n";
?>
</h2>
FIN DU HTML
</h1>
</body>
</html>
```



L'une des technologies émergentes de JavaScript est connue sous le nom d'**AJAX**.

AJAX est un acronyme anglais signifiant **Asynchronous Javascript And XML** (Javascript et XML asynchrones). Il s'agit d'envoyer et recevoir des requêtes HTTP, pour communiquer avec des **scripts situés sur un serveur**.

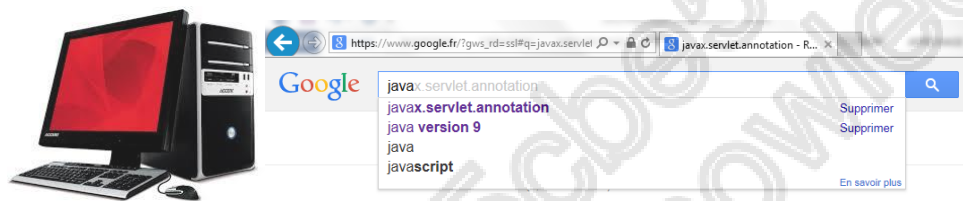
Cela permet d'échanger des informations sous différents formats (XML, HTML ou texte), mais son principal attrait est sa nature « asynchrone » : tout cela peut se faire sans recharger la page et sans bloquer l'utilisateur.

C'est ce qui permet de mettre à jour certaines parties d'une page sur la base d'événements déclenchés par l'utilisateur (messagerie Gmail ou Yahoo, complétude à la saisie avec la barre Google, ...).

Les deux fonctionnalités combinées sont les possibilités de :

- faire des requêtes vers le serveur sans avoir à recharger la page ;
- analyser et travailler avec des documents XML.

AJAX est une des technologies phares du mouvement **Web 2.0** qui définit les interfaces riches permettant à l'internaute une plus grande interactivité avec la page Web.



jQuery est une bibliothèque libre JavaScript qui porte sur l'interaction entre JavaScript (avec AJAX) et HTML. Son but est de simplifier des commandes communes de Javascript. La première version date de janvier 2006. Cet ensemble contient des fonctionnalités comme le parcours et la modification du document, la gestion des événements, des effets et animations, la manipulation de feuilles de style, des utilitaires (version du navigateur...), des plugins.

De nombreux **langages voient le jour encore aujourd'hui**, ils sont créés pour répondre à certaines fonctionnalités. Les derniers en date : Dart, Go, Scala, Groovy . Dart et Go ont été créés par Google, respectivement en 2011 et 2009, Dart se positionne pour remplacer JavaScript.

Google aimerait voir utiliser Dart couramment dans la programmation web, car JavaScript convient très bien quand on doit mettre de l'interactivité simple dans des pages web, mais quand les applications atteignent des milliers de lignes de code, ses faiblesses apparaissent. Voilà pourquoi Google a créé Dart. Quant à Go, c'est un langage d'usage général qui convient à peu près pour tout, du développement d'application à la programmation de systèmes. En ce sens, cela ressemble davantage à du C ou du C++ qu'à du Java ou du C#. Mais, comme ces derniers, Go inclut des fonctions considérées comme modernes (« garbage collector » pour le nettoyage de la mémoire).

Chaque année, un classement des langages les plus populaires est édité par divers cabinets et instituts, qui obtiennent les informations en fonction des demandes sur les moteurs de recherche du Web, en fonction de sondages qu'ils effectuent ... Bref diverses collectes qui peuvent nous laisser des interrogations. Voici quelques classements pour l'année 2014.

Classement du TIOPE, (Source <http://www.developpez.com/actu/65954/Classement-TIOBE->)

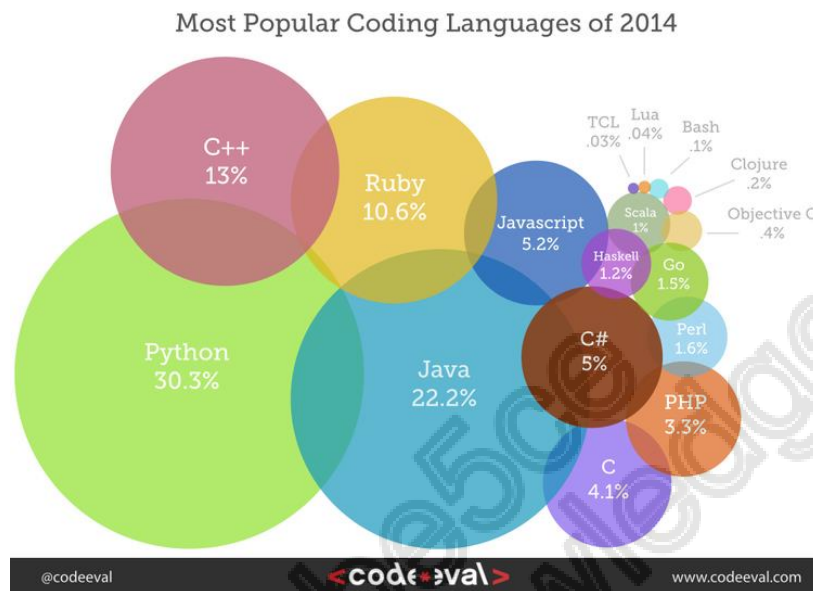
Jan 2014	Jan 2013	Change	Programming Language	Ratings	Change
1	1		C	17.871%	+0.02%
2	2		Java	16.499%	-0.92%
3	3		Objective-C	11.098%	+0.82%
4	4		C++	7.548%	-1.59%
5	5		C#	5.855%	-0.34%
6	6		PHP	4.627%	-0.92%
7	7		(Visual) Basic	2.989%	-1.76%
8	8		Python	2.400%	-1.77%
9	10	↑	JavaScript	1.569%	-0.41%
10	22	↑↑	Transact-SQL	1.559%	+0.98%

Classement pour Pyll (issu de la même source developpez.com)

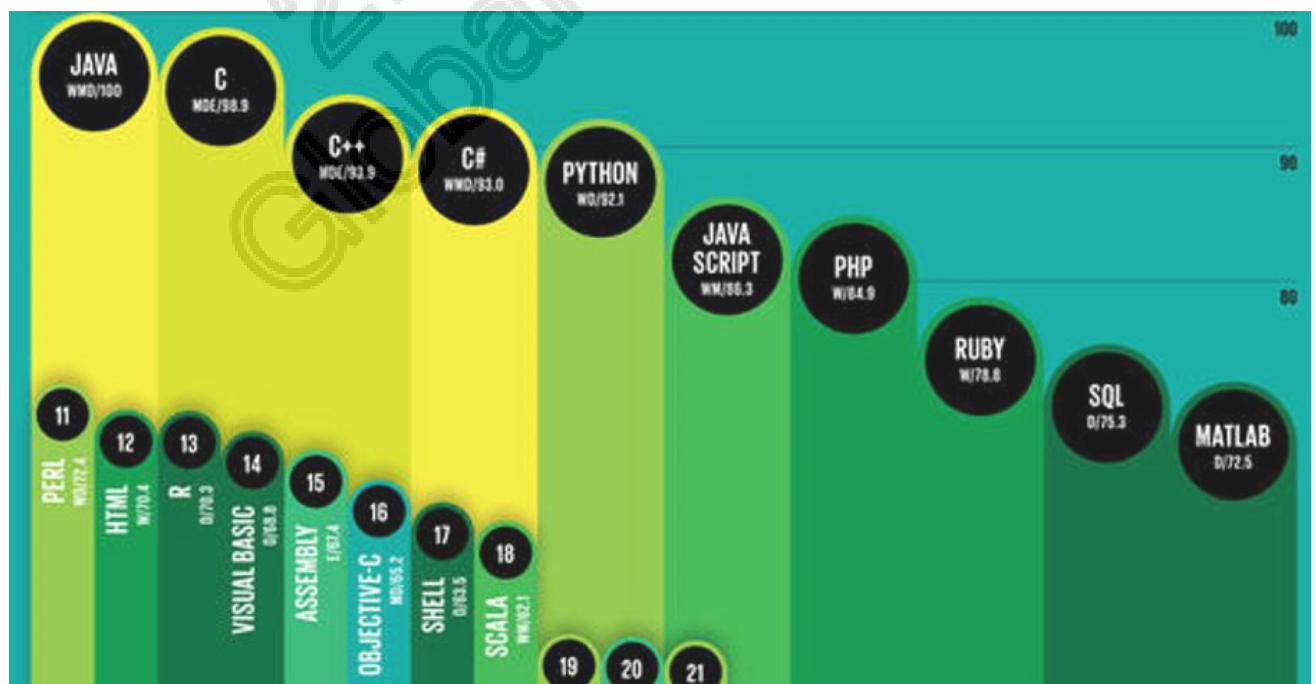
Position Jan 2014	Position Jan 2013	Delta in position	Programming language	Share in Jan 2014	Twelve month trends
1	1		Java	26.2 %	-0.6 %
2	2		PHP	13.2 %	-1.6 %
3	6	↑↑↑	Python	10.2 %	+1.3 %
4	3	↓	C#	9.6 %	-0.4 %
5	4	↓	C++	8.9 %	-0.0 %
6	5	↓	C	8.1 %	-0.2 %
7	7		Javascript	7.6 %	+0.3 %
8	8		Objective-C	6.6 %	+0.8 %
9	9		Ruby + Rails	5.1 %	+0.5 %
10	10		Visual Basic	2.8 %	-0.2 %
© 2014 Pierre Carbonnelle					

Résultats de l'entreprise CodeEval, qui a traité plus de 100 000 tests et défis de programmation par plus de 2000 employés.

(Source <http://www.carrementplus.net/2014/02/05/les-langages-de-programmation-les-plus-populaires-en-2014/>)



IEEE Spectrum a publié un classement des langages de programmation les plus populaires. Cette information, produite par Nick Diakopoulos, est le fruit d'un travail combinant une dizaine de sources (IEEE Xplore, Google, and GitHub...) et une douzaine de métriques pour pondérer le tout. (Source <http://www.open-source-guide.com/Actualites/Les-langages-de-programmation-les-plus-populaires>)



4.5. Les outils de développement

Il existe deux méthodes pour développer des applications. La première consiste à utiliser un simple éditeur de texte, un compilateur et l'environnement d'exécution. La deuxième, de plus en plus utilisée pour une **meilleure productivité** et un confort de travail, est la manipulation de « studios de développement ».

Les « studios de développement » peuvent être de type IDE, Integrated Development Environment ou de type RAD Rapid Application Development. Ce sont des logiciels multifonctions réunissant dans la même interface tous les outils dont le développeur a besoin. Les outils les plus manipulés sont :

- L'éditeur de source
- L'explorateur de code
- Le compilateur,
- Le débogueur
- Le générateur d'interfaces graphiques
- Le gestionnaire de projet

L'IDE est un logiciel tourné vers la productivité au niveau du code source.

Il propose un éditeur de texte complété d'un gestionnaire de projet, ainsi que des menus offrant la possibilité de compiler rapidement.

Le programmeur peut, sans quitter son environnement, exécuter toutes les étapes de la création d'une application.

L'IDE intègre souvent des fonctionnalités supplémentaires telles que les explorateurs de codes ou les assistants à la programmation.

RAD est destiné à faciliter l'interface graphique. Même si la réalisation d'une IHM, Interface Homme Machine devient plus facile il est nécessaire d'avoir le code source. En général, les RAD offrent tous un module IDE minimal.

Conclusion : la frontière entre les deux outils est limite. Tous les RAD intègrent des fonctions IDE et les IDE les plus poussés proposent des fonctions de RAD. Lorsqu'un outil offre le meilleur des deux mondes, on parle alors de VDE, Visual Development Environment. Cependant, le terme le plus utilisé, quel que soit l'outil, reste IDE.

5. LES PLATES-FORMES JAVA EE ET .NET

5.1. Présentation

L'évolution actuelle tend à considérer que **le langage** de programmation, même sous sa forme la plus aboutie du L4G, **n'apporte qu'une réponse partielle aux besoins des équipes de développement** qui doivent systématiquement s'équiper de produits complémentaires pour répertorier les ressources utilisées (bases de données, annuaires), établir des connexions applicatives entre plusieurs ordinateurs (middleware)...

Les éditeurs apportent une réponse à ce besoin plus global **en « packageant » un ensemble de produits** dont ils garantissent la compatibilité. C'est le cas des offres :

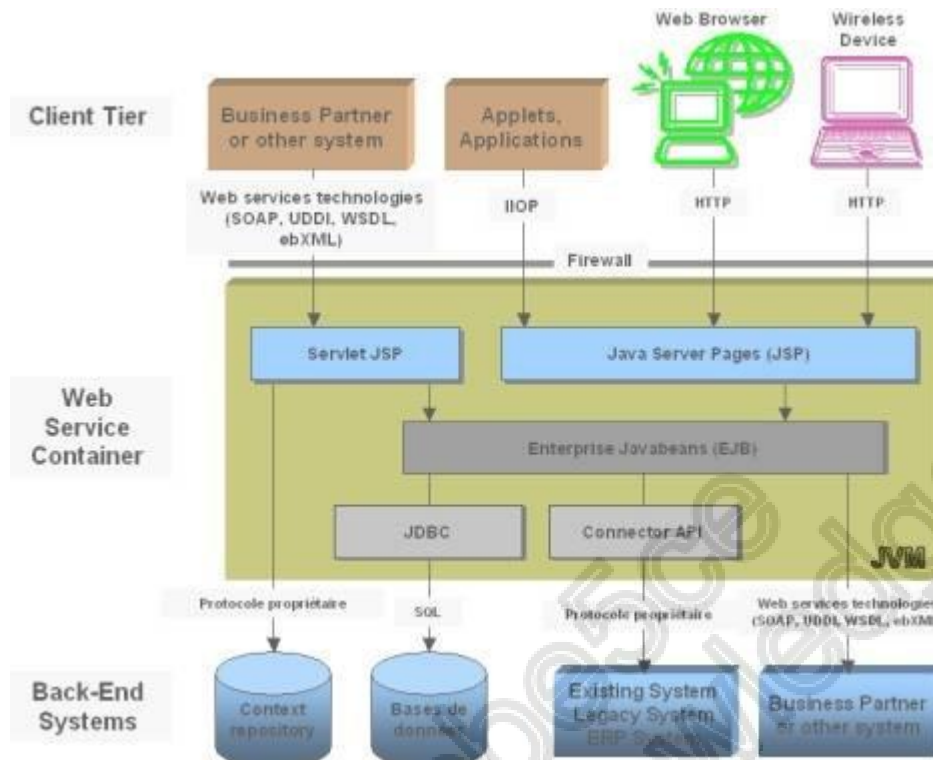
Java EE (anciennement **J2EE**, Java Enterprise Edition) constituée sous l'égide d'Oracle depuis son rachat de Sun ,

.Net, de Microsoft.

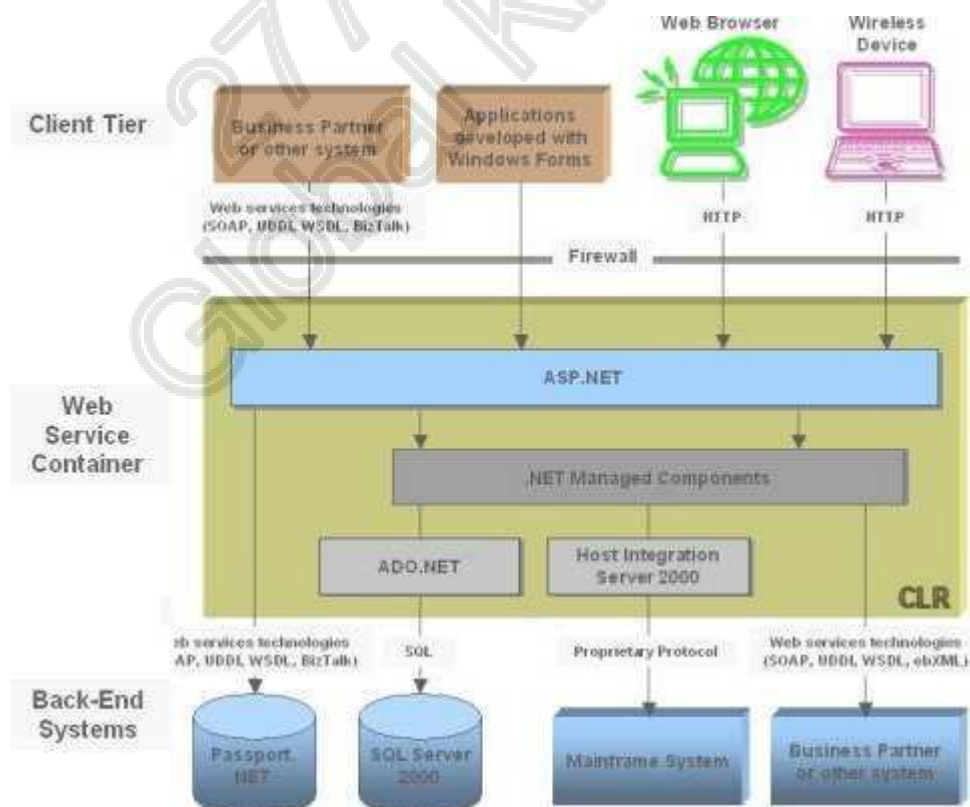
L'utilisation de ces plates-formes nécessite :

- pour la conception/réalisation : des outils de développement, les fameux studios ou IDE.
- pour les tests : des serveurs d'applications, afin d'exécuter l'application développée.

La plate-forme Java EE



La plate-forme .Net



5.2. Les outils

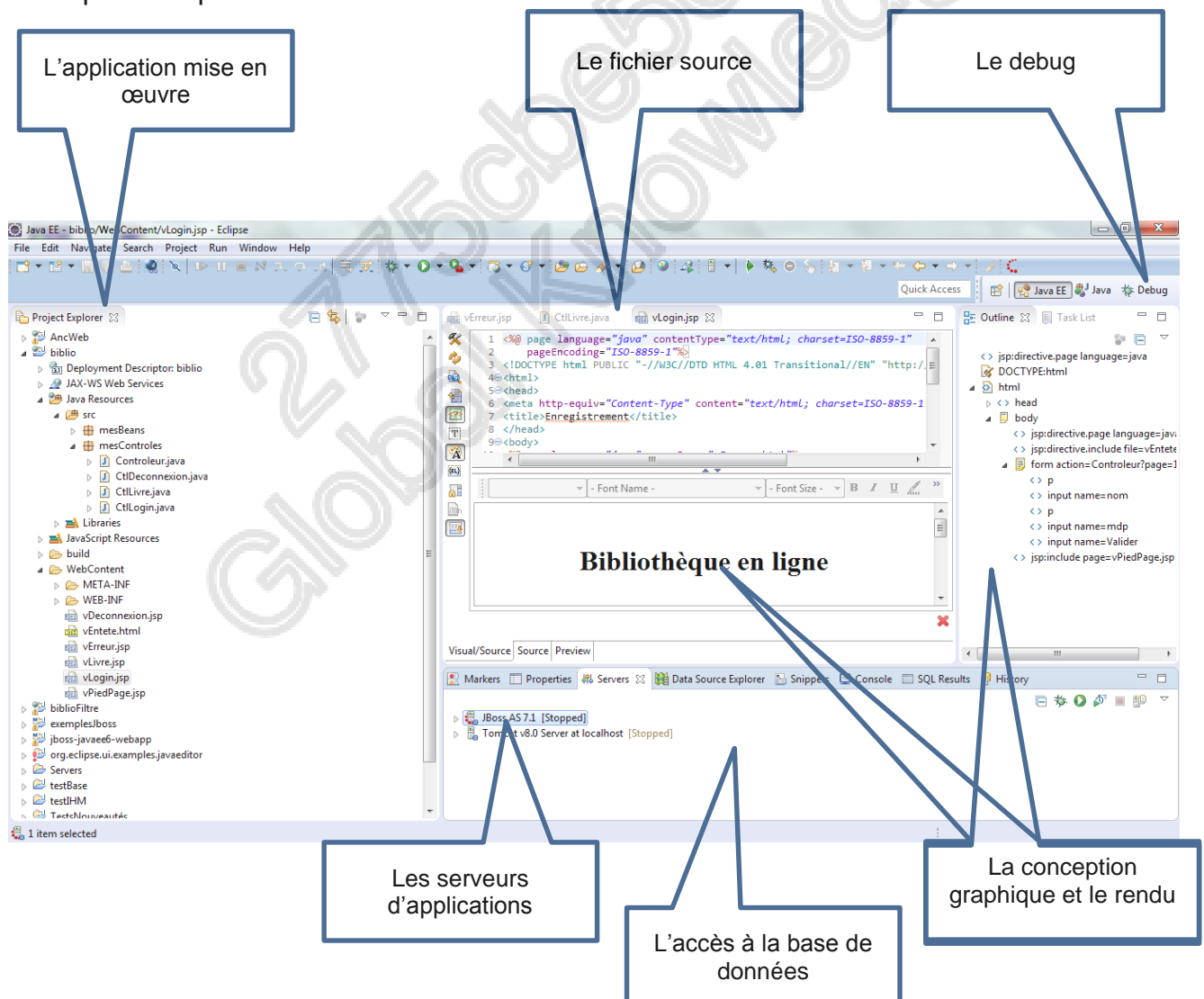
Microsoft propose **VisualStudio** pour .Net.

Sun/Oracle a laissé l'ensemble des éditeurs proposer des outils autour de Java EE. Quelques-uns ont vu le jour tel que JBuilder de Borland, Sun One Studio de Sun, Workshop de BEA system, Rational Development d'IBM (anciennement WebSphere Studio).

Aujourd'hui il existe un produit Open Source, plébiscité par l'ensemble des développeurs Java, c'est **Eclipse**. Cet outil a été créé par IBM puis mis à la disposition du monde Open Source. On pourra aussi utiliser l'IDE que Sun a mis en open source en 2000 : NetBeans.

Ces IDE, supportent plusieurs langages de programmation. Quoi qu'il en soit, quel que soit le produit, le développeur retrouve les mêmes fonctionnalités, pour mettre en œuvre des applications, et s'adapte à l'outil qu'il utilise.

Exemple d'Eclipse :



6. LES FRAMEWORKS

Le terme framework désigne un « cadre de travail » pour le développeur

- Il s'agit en général d'une bibliothèque logicielle qui simplifie le développement en proposant des briques de plus haut-niveau (gain de temps)
- L'utilisation du framework doit également permettre d'écrire un programme de qualité (lisibilité, performance, ...).
- Un framework est souvent accompagné d'un environnement logiciel qui permet de bien l'utiliser

Il y aura des frameworks spécifiques à Java et d'autres à .Net, d'ailleurs on parle du framework .net.

Pour vous donner un exemple de ce que peut être un framework, nous avons présenté les 3 couches principales d'une application (d'un logiciel) :

- La présentation à l'utilisateur
- Les traitements à mettre en œuvre (pour présenter les données)
- Les données d'entreprise

En tant que développeur, il y aura toujours à effectuer du code pour ces 3 couches mais aussi à mettre en œuvre le code nécessaire pour faire communiquer ces 3 couches. Soit on programme tout, soit on se fait aider d'un framework qui aura déjà codé la communication. Le type de framework qui peut aider à mettre en œuvre ces 3 briques, est défini comme un framework MVC : Model View Controller :

- le Modèle étant les données d'entreprise,
- la Vue, les présentations,
- le Contrôleur, les traitements.

Quand on dispose d'une solution pour résoudre un problème informatique, on parle de **Design Pattern**, de modèle de conception, c'est aussi un ensemble de bonne pratique.

En développement logiciel, vous serez souvent confronter, aux mêmes problèmes, d'autres auront peut-être capitalisé, et auront réfléchi à une solution avant vous. De nombreux pattern existent, ensuite il faudra choisir le framework implémentant cette solution. Dans le monde Java, les frameworks MVC les plus utilisés sont Struts, JSF (Java Server Faces), Spring MVC.

Pour définir l'accès aux données d'entreprise, il faudra utiliser le design pattern DAO (Data Access Object) et choisir un framework. Hibernate a été le plus utilisé dans la technologie Java.

CHAPITRE V

LE PROJET INFORMATIQUE

Objectif : Définir le projet informatique, les acteurs, les démarches et méthodes de conception, ainsi que leurs évolutions. Positionner les référentiels de qualité

1. DEFINITIONS ET OBJECTIF D'UN PROJET

Un **projet** est constitué par l'ensemble des actions à entreprendre pour atteindre un objectif donné.

Il est caractérisé par :

- un budget,
- un calendrier,
- une organisation (ressources humaines, logicielles, techniques),
- des produits (logiciel exécutable, manuels d'utilisation et d'exploitation, documentation).

Le projet est terminé lorsque les objectifs sont atteints.

Pour une bonne maîtrise du projet et son bon déroulement, il est recommandé d'avoir une méthodologie et une méthode de projet.

La méthodologie est l'ensemble des actions à entreprendre pour parvenir à un but. Elle répond aux questions

QUE FAUT-IL FAIRE, QUI LE FAIT et QUAND ?

La **méthode** est un ensemble de démarches raisonnées, suivies pour parvenir à un but. Elle répond à la question :

COMMENT FAUT-IL LE FAIRE ?

Une méthode de projet recouvre trois aspects :

- la démarche,
- l'approche,
- la conduite.

Ainsi, **pour faciliter la conduite de projet**, on dispose de **démarches de projet**,

L'approche, détient des techniques de représentation, de **modélisation**. La conduite est aussi appelée pilotage de projet.

2. MAITRISE D'OUVRAGE / MAITRISE D'ŒUVRE

2.1. Définitions

Le **Maître d'Ouvrage** est une personne, physique ou morale qui **a un besoin « métier »**. Il finance et commande un ouvrage qui doit répondre à :

- Des spécifications.
- À des exigences de qualité, coûts et délais.

L'ouvrage est un ensemble cohérent de fournitures, livrables et de services associés. C'est le **résultat d'un projet**.

Le Maître d'Ouvrage délégué

- Veille à l'acceptation des modifications
- Mesure les conséquences

Le **Maître d'œuvre** est une personne, physique ou morale, **responsable** de l'ensemble des **travaux de réalisation** d'un ouvrage (conception, construction et mise en œuvre). Il assure la mise à disposition des moyens, il livre le projet.

L'œuvre est un ensemble des **travaux nécessaires à la réalisation** d'un ouvrage. Le travail de production d'une chose.

Le Maître d'œuvre délégué

- Délivre un produit défini
- Met à disposition des ressources

Pour la mise en œuvre, il existe 3 types de délégation ou **sous-traitance** :

- de **spécialité** : l'entreprise, ne disposant pas du savoir-faire nécessaire pour fabriquer le produit ou le réaliser, donne l'ordre à une entreprise de le faire,
- de **capacité** : l'entreprise est dans l'incapacité de répondre, à un moment donné, à produire des commandes supplémentaires,
- de **marché** : une entreprise confie à une autre entreprise un marché conclu avec un maître d'ouvrage. Cela met donc en relation le maître d'ouvrage, l'entreprise donneur d'ordres, et le sous-traitant.

2.2. Les différents types de responsabilités dans le projet

Le projet informatique définit 2 types de responsabilité qui se succèdent au cours du projet :

- la **maîtrise d'ouvrage** a en charge la définition **fonctionnelle** de la future application.
- la **maîtrise d'œuvre** a en charge la conception **technique** et la réalisation de la future application.
- enfin, la **maîtrise d'ouvrage** est garante de la bonne mise en production de l'application, et donc assume la responsabilité des tests.

Le projet est donc marqué par 2 changements de responsabilité, qui nécessitent contractualisation :

- le **cahier des charges** définit les attendus de la maîtrise d'ouvrage vis-à-vis de la maîtrise d'œuvre,
- la **recette** formalise la procédure de réception et d'acceptation (ou de non acceptation) contractuelle du produit fourni par la maîtrise d'œuvre.

2.3. Les démarches

Les démarches de projet, découpent en cinq phases principales, tout processus de développement logiciel :

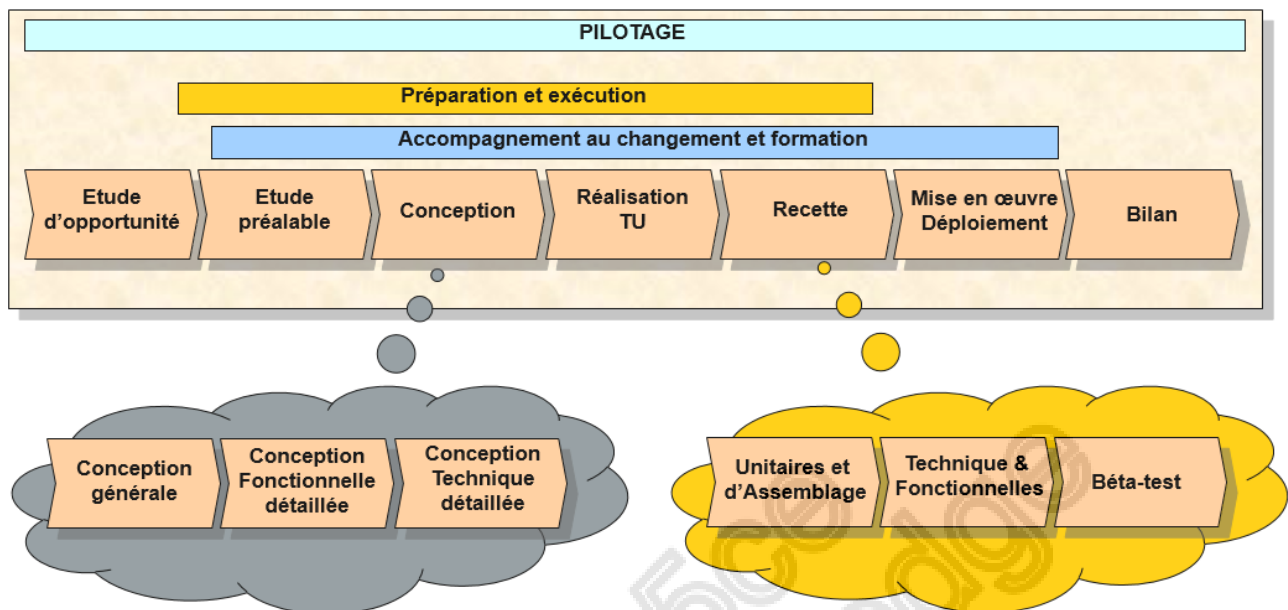
- l'**analyse** du problème,
- la **conception** de la solution,
- la **réalisation** du système,
- l'**installation** du système,
- l'**évaluation** des résultats,
- phases auxquelles il faut ajouter celle de **démarrage**, initialisation indispensable du processus.

Les démarches décrivent de plus les activités transverses menées tout au long du projet :

- l'**estimation** des charges et des délais phase par phase, et la ré-estimation en cours de projet le cas échéant,
- l'**évaluation** des risques inhérents au projet,
- la **planification** des différentes phases et des activités qui les constituent,
- l'**affectation** des ressources à chaque phase et à chaque activité,
- le **suivi** de l'avancement du projet,
- la communication de projet, en particulier vers les futurs utilisateurs et la maîtrise d'ouvrage.

	MERISE	AXIAL	M.C.P.	SDM/S	UP
Conception	Schéma directeur	Schéma directeur			
	Etude préalable	Etude initiale	Etude d'opportunité	Définition des Besoins du Système	Définition des besoins
			Etude du système d'information	Choix d'Architecture du Système	
	Etude détaillée	Conception générale	Elaboration du cahier des charges	Spécification Externes du Système	Analyse
Réalisation	Etude technique	Conception détaillée	Etude du système informatique	Spécifications Internes du Système	Conception
	Production	Réalisation	Programmation	Programmation Tests	Implémentation
			Mise en oeuvre	Conversion Installation	Tests
	Exploitation			Bilan	

2.4. Les différentes phases d'un projet



2.5. Détail des phases

2.5.1. Démarrage du projet

Le projet est initialisé par une demande ayant pour origine les utilisateurs, le plan informatique, le schéma directeur ou le département informatique. En début de projet, il est nécessaire de mener une brève étude qui permet :

- d'identifier la portée du projet,
- d'apprécier le besoin et de vérifier que les avantages attendus sont réalisables,
- d'identifier les relations entre le domaine concerné, les autres systèmes et le plan informatique,
- d'identifier des solutions possibles,
- de proposer des orientations pour l'ensemble du développement, préparer le plan de travail, constituer une équipe...

Cette étape de démarrage permet de mieux cerner les attentes du demandeur, de mieux estimer les ressources nécessaires pour entreprendre le développement et de ce fait offre les informations nécessaires à une prise de décision :

- faire,
- faire partiellement,
- différer le développement,
- abandonner.

2.5.2. Analyse du problème

Cette fonction recouvre une **étude** complète du système **existant** et une **analyse** critique permettant d'en détecter les dysfonctionnements pour ne pas les reproduire. Ceci se fait en étroite relation avec l'utilisateur.

Cette étape contient également le recueil des besoins du nouveau système et les exigences des utilisateurs dont on fera un classement.

A partir de l'étude du système existant et des nouveaux besoins, quelques solutions (les scénarios) seront présentées dans un dossier de choix soumis aux décideurs nommés en début de projet. On donnera une présentation fonctionnelle et technique de chaque solution ainsi que son coût de développement et une estimation du coût futur d'exploitation.

A l'issue de cette étape, un seul scénario sera retenu pour mener la suite du projet.

2.5.3. La conception

Conception de la solution : suite à l'analyse du problème, plusieurs solutions ont été présentées et une seule a été retenue. Il s'agit maintenant d'en faire une description complète (étendue à tout le domaine d'étude) et détaillée.

La conception se fait à deux niveaux. D'une part sur un plan fonctionnel, il sera fait une description complète du nouveau système qui sera soumise à l'utilisateur pour validation, d'autre part sur un plan technique, cette partie correspondant à l'analyse organique.

La **conception fonctionnelle**, ou **conception externe**, s'attache à décrire les fonctionnalités du futur système, c'est-à-dire la partie qui en sera visible par les utilisateurs :

- les fonctionnalités couvertes, *i.e.* les services rendus par l'application,
- les maquettes des écrans,
- les dessins des états.

La **conception organique**, ou **conception interne**, s'attache à décrire la structuration interne des différents composants de l'application, et en particulier :

- le découpage des fonctionnalités en programmes et sous-programmes,
- la structuration des fichiers et des bases, ainsi que leur organisation.

La conception fonctionnelle et la conception organique se mènent le plus souvent en s'appuyant sur une méthode de conception, telles que **Merise** ou **UML**.

2.5.4. La réalisation – les tests

Réalisation du système : cette partie recouvre la construction du produit final respectant les fonctionnalités désirées par l'utilisateur et l'architecture technique mise en place.

La phase de réalisation recouvre les activités de :

- codage des programmes applicatifs,
- production de la documentation de programme, dans le but de faciliter les maintenances ultérieures,
- production des directives d'exploitation du système (politique de sauvegarde et de restauration, par exemple),
- production de guides utilisateurs et d'aides en ligne le cas échéant, voire des dispositifs de formation.

La phase de réalisation du système, acceptée au sens large, recouvre aussi les tests de l'application. Plusieurs activités de test sont nécessaires pour garantir l'adéquation du système aux besoins exprimés :

- des **tests unitaires**, au cours desquels les différents programmes sont testés indépendamment les uns des autres, tests menés à partir de jeux d'essais, c'est-à-dire de cas d'utilisation conçus spécifiquement dans un but de validation,
- des **tests d'intégration**, tests de fonctionnalités qui enchaînent les différents programmes d'une même unité fonctionnelle, puis les unités fonctionnelles entre elles, menées, eux aussi, sur la base de jeux d'essai,
- des **tests systèmes**, dont l'objectif est de vérifier la capacité du système à supporter la future application,
- des **tests opérationnels**, qui sont menés à partir de données et de contextes de production.

Le terme de **recette** désigne l'ensemble des tests que pratiquent les futurs utilisateurs d'un système lors de la livraison de l'application par les informaticiens. Ce terme implique un cadre contractuel.

2.5.5. L'installation

Installation du système : lorsque l'utilisateur déclare que le logiciel est suffisamment testé et répond à ses attentes, il sera installé sur la machine réelle et l'on pourra procéder à la formation des utilisateurs finals.

L'installation du système peut nécessiter la conception et le développement de programmes spécifiques, en particulier des programmes permettant de récupérer (ou de convertir) les données existantes le cas échéant.

2.5.6. L'évaluation

Évaluation des résultats : en fin de projet, on procédera à un bilan du projet, de son déroulement, de l'atteinte des objectifs aux coûts et délais prévus, mais également un bilan du logiciel : répond-il aux attentes des utilisateurs ? Ceci afin d'envisager d'éventuelles évolutions.

Cette évaluation a aussi comme but de capitaliser l'expérience apportée par ce projet, **capitalisation** d'autant plus importante que l'évolution rapide des techniques, des outils et des méthodes informatiques rend caduques les expériences d'il y a 5 ou 10 ans...

ETAPES D'UN PROJET INFORMATIQUE

DEMARRAGE	ANALYSE DU PROBLEME	CONCEPTION DE LA SOLUTION	REALISATION DU SYSTEME	INSTALLATION	EVALUATION DES RESULTATS
<ul style="list-style-type: none"> Délimiter les frontières de l'étude Recenser les utilisateurs Nommer les décideurs Choisir un cycle de développement Déterminer les points de validation Estimer et planifier le travail 	<ul style="list-style-type: none"> Analyser le système existant Recueillir les nouveaux besoins Proposer des solutions 	<p><u>A partir de la solution retenue :</u></p> <ul style="list-style-type: none"> Etudier en détail les fonctions du nouveau système Définir les critères d'acceptation Décrire la base de données Décrire le nouveau système en tenant compte de l'environnement technique Organiser et planifier la programmation, les tests et l'installation 	<ul style="list-style-type: none"> Ecrire les programmes Effectuer les tests 	<ul style="list-style-type: none"> Mettre l'application en exploitation et tester 	<ul style="list-style-type: none"> Faire un bilan du projet

POINTS DE DECISION / VALIDATION

DEMARRAGE	ANALYSE DU PROBLEME	CONCEPTION DE LA SOLUTION	REALISATION DU SYSTEME	INSTALLATION	EVALUATION DES RESULTATS
<u>POINT 1</u>	<u>POINT 2</u>	<u>POINT 3</u>	<u>POINT 4</u>		<u>POINT FINAL</u>
<ul style="list-style-type: none"> Décision de faire, annuler ou différer le projet 	<ul style="list-style-type: none"> Choix de la solution retenue 	<ul style="list-style-type: none"> A l'issue de l'analyse fonct : Validation du dossier fonctionnel Approbation des critères d'acceptation 	<ul style="list-style-type: none"> Décision de passer l'application en exploitation Validation du manuel utilisateur Validation du dossier d'exploitation 		

INTERVENANTS SUR LE PROJET

DEMARRAGE	ANALYSE DU PROBLEME	CONCEPTION DE LA SOLUTION	REALISATION DU SYSTEME	INSTALLATION	EVALUATION DES RESULTATS
<ul style="list-style-type: none"> Chef de projet Utilisateur responsable 	<ul style="list-style-type: none"> Chef de projet Analystes Utilisateurs 	<ul style="list-style-type: none"> Chef de projet Administrateur de BdD Analystes Analystes organiques Utilisateurs 	<ul style="list-style-type: none"> Chef de projet Analystes organiques Analystes programmeurs Utilisateurs (tests) 	<ul style="list-style-type: none"> Chef de projet Analystes organiques Responsable d'exploitation 	<ul style="list-style-type: none"> Chef de projet Utilisateur responsable

3. CYCLES DE DEVELOPPEMENT

Le choix du cycle de développement du projet consiste à répartir les tâches du projet entre des tranches de temps, ou phases, séparées par des jalons. Les démarches nécessaires pour conduire le projet peuvent aujourd'hui se classer en deux grandes familles :

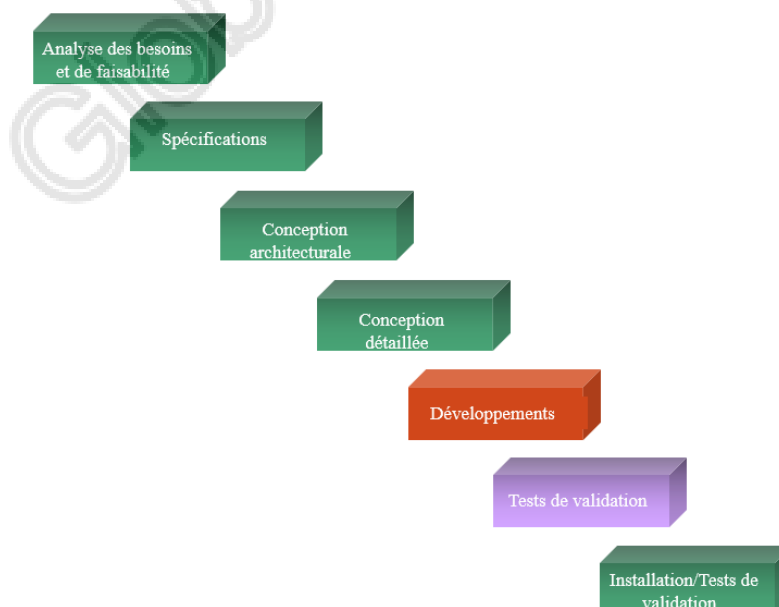
- les démarches linéaires, les plus traditionnelles,
- les démarches itératives, basées sur des techniques de maquettage et/ou de prototypage.

On admet à présent que le meilleur des cycles de développement est celui qui est le mieux adapté aux caractéristiques majeures du projet : sa taille, son couplage interne, le degré d'implication des utilisateurs, le caractère novateur du logiciel à construire, le caractère novateur des techniques de développement envisagé, les risques correspondants.

3.1. Cycle linéaire

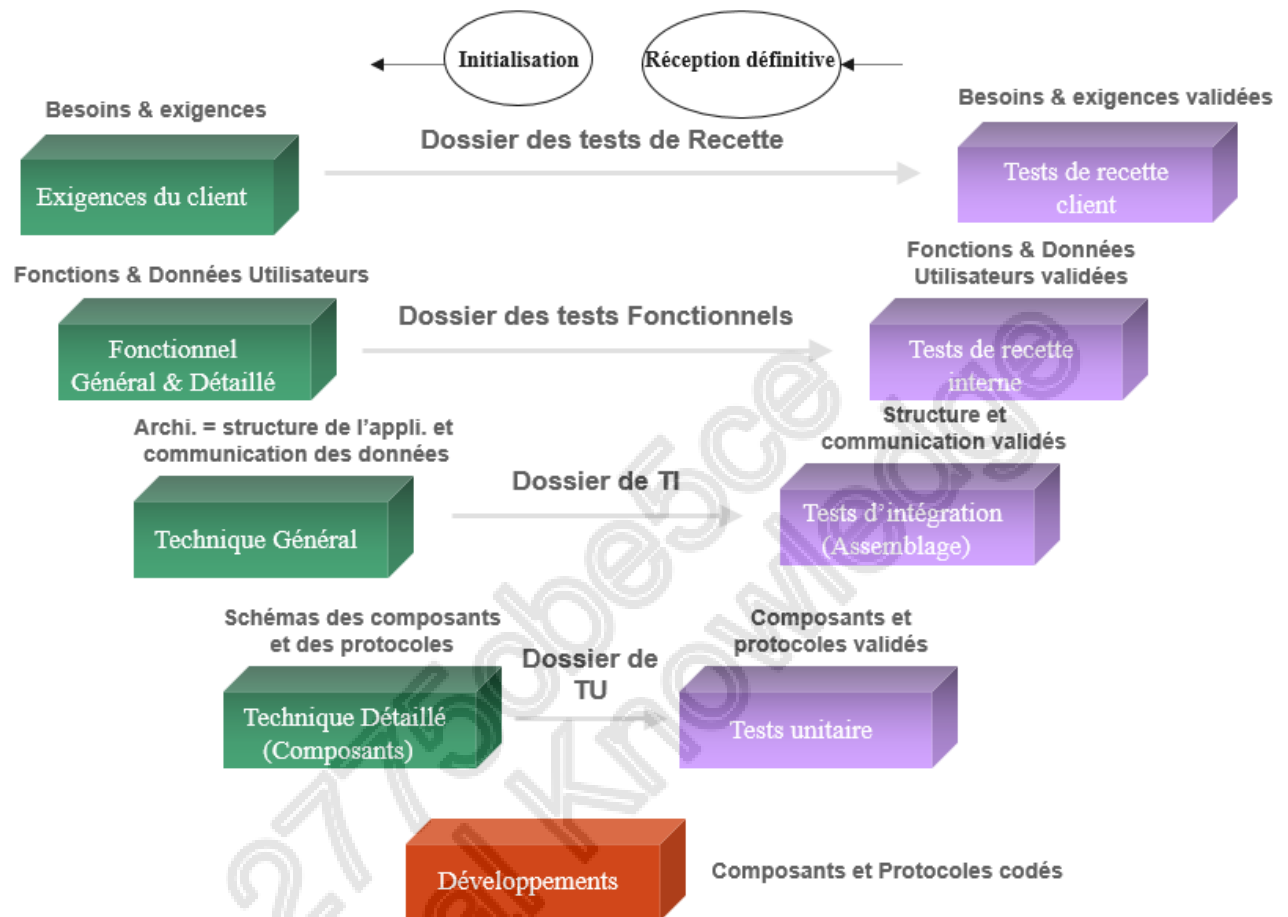
3.1.1. Cycle en cascade

Les vertus de ce cycle résident dans le respect des niveaux d'abstraction (du conceptuel au physique) ainsi que dans la maîtrise des processus décisionnels. A chaque fin de phase, l'équipe de développement présente un résultat tangible et prédéfini, à partir duquel les décideurs peuvent se déterminer pour la poursuite du projet. Du strict point de vue de l'organisation, paraît simplifier les choses. Mais cela se paye par une **rigidité** jugée souvent excessive et une grande **difficulté d'adaptation**. Pour un projet important, les phases d'étude peuvent mobiliser de gros moyens, sans autre retombée que des dossiers.



3.1.2. Cycle linéaire en V

Une variante du cycle en cascade est le cycle en V, qui met en relation chaque phase du développement avec une des phases de test.



Ces 2 cycles, définis dans des démarches linéaires, sont très découpés et rendent les retours arrière très difficiles. On peut rencontrer des difficultés d'intégrer des nouveaux besoins en conception.

3.2. Cycles itératifs

3.2.1. Présentation

Les démarches itératives apparaissent progressivement pour pallier les déficiences des démarches linéaires.

Les deux principales méthodes sont celle définies par la norme AFNOR Z67-111 (modèle en spires successives), ou la **spirale de Boehm**.

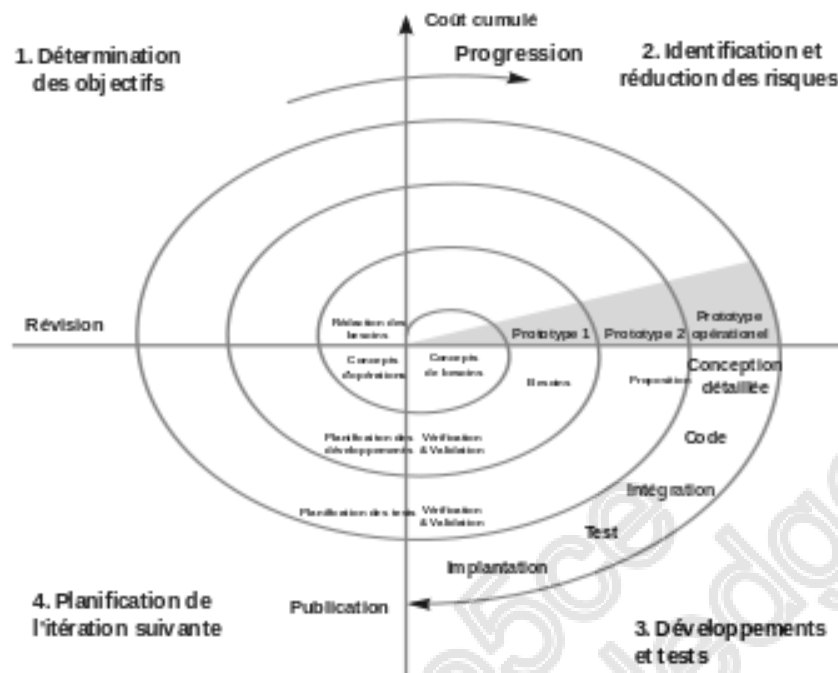
Le cycle de vie itératif présente trois caractéristiques principales :

- il affiche le caractère incrémental et non linéaire du développement,
- il met l'accent sur l'interactivité entre émetteur du besoin et équipe de développement, la spirale ramenant face à face les interlocuteurs à chaque passage de spire,
- il instaure des ruptures entre les différents états de la réalisation logicielle.

Ces cycles de vie sont adaptés au **maquettage** et au **prototypage**. Pour une fonctionnalité donnée, chaque tour de spire correspond à la livraison d'un des produits suivants :

- la maquette de plate-forme, validée par le développeur sur des scénarii prédéfinis, dans un environnement d'exécution simplifié et sur un nombre de fonctions réduites,
- la maquette expérimentale, validée par l'utilisateur sur des données opérationnelles, dans un environnement d'exécution simplifié et sur un nombre de fonctions réduites,
- le prototype de plate-forme, validé par le développeur sur des scénarii prédéfinis, dans l'environnement d'exécution cible et sur l'exhaustivité des fonctions implémentées,
- le prototype expérimental, validé par l'utilisateur sur des données opérationnelles, dans l'environnement d'exécution cible et sur l'exhaustivité des fonctions implémentées,
- le produit fini.

3.2.2. Cycle en spirale



Chaque tour de spire correspond à l'ensemble des phases à entreprendre pour développer un état intermédiaire ou l'état fini du logiciel :

- analyse du besoin, en particulier détermination du niveau de finition du logiciel à obtenir,
- conception du besoin,
- réalisation du besoin,
- test du besoin,
- intégration et mise en exploitation, lorsque le produit obtenu est un produit fini,
- validation du besoin,
- bilan du besoin, de manière à entreprendre des actions correctives qui seront prises en compte pour le développement du besoin suivant,
- corrections éventuelles,
- planification du développement du besoin suivant.

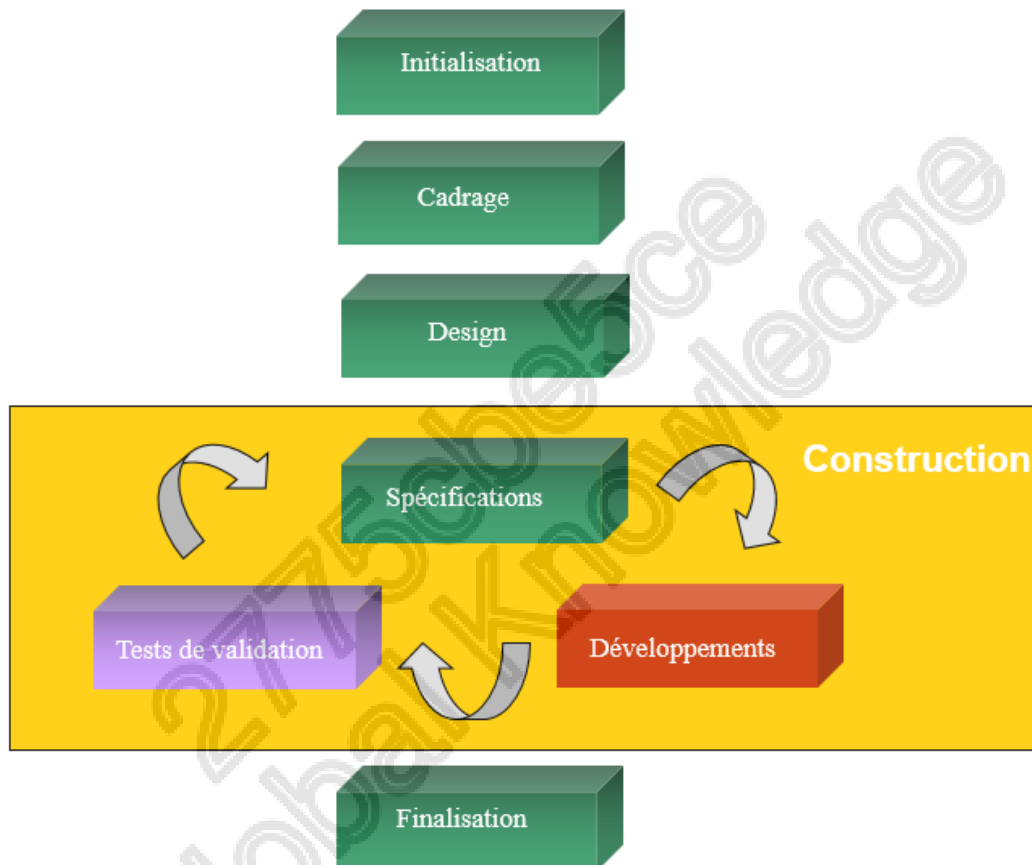
Les projets conduits selon une méthode de conduite de projet itérative peuvent être menés selon les principes **JAD/RAD** (*Joint Application Design / Rapid Application Design*).

Cette démarche permet l'intégration de nouveaux besoins, possible à tout moment, nécessitant des résultats rapides. Cependant, il faudra éviter la spirale infinie.

3.2.3. Cycle semi itératif

Le cycle semi-itératif pallie le risque de spirale infinie en intégrant une phase de définition de besoin linéaire sur l'ensemble du besoin.

Le cycle semi-itératif est découpé en trois étapes. Les deux premières étant classiques et consistent en l'expression des besoins et la conception de la solution. C'est l'étape de construction du produit qui est itérative.

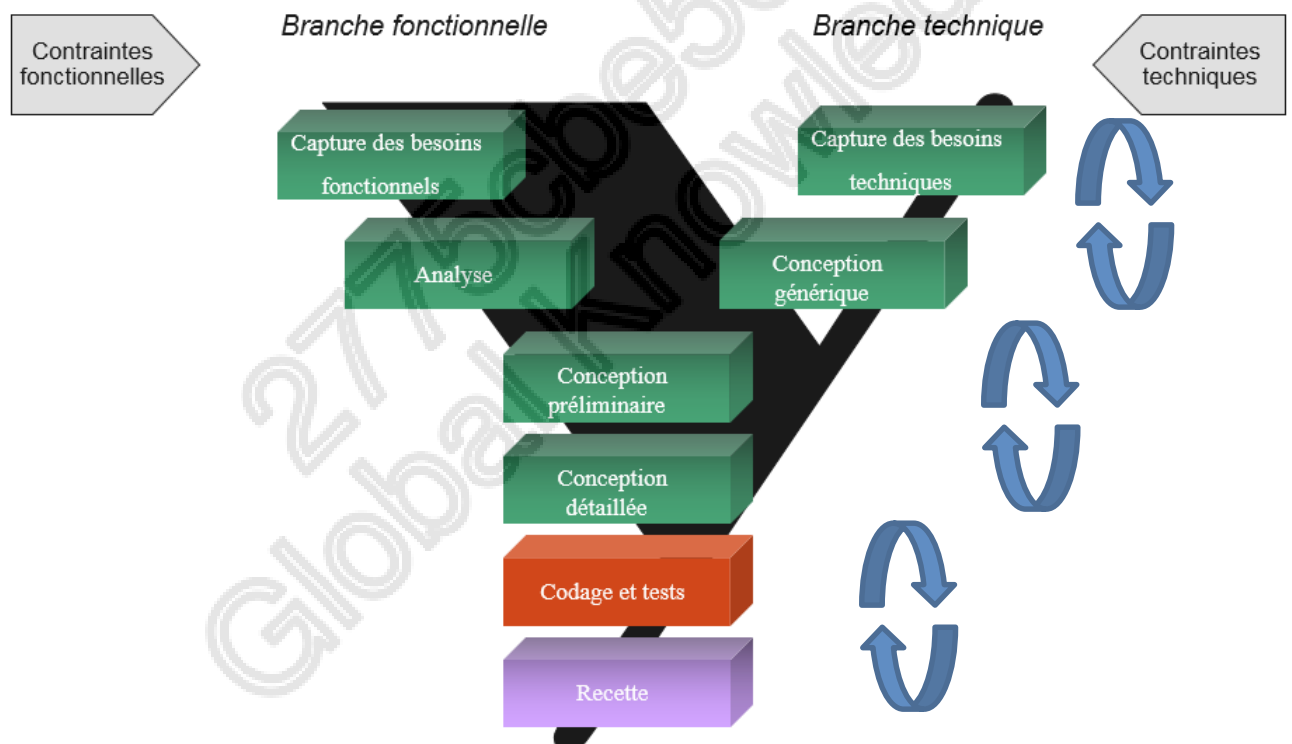


3.2.4. Cycle en Y

Un produit logiciel dépend :

- Des besoins des utilisateurs,
- Des contraintes techniques :
 - L'architecture matérielle
 - Le(s) système(s) d'exploitation
 - Le(s) système(s) de Gestion de Bases de Données
 - Le(s) réseau(x)
 - L'existant

L'itération est une répétition d'une séquence, dans le but de reprendre la séquence sur des données différentes, à chaque répétition. Dans le cycle en Y l'itération se fera sur **l'architecture, l'analyse, et sur la conception, réalisation.**



La branche de gauche capitalise la connaissance du métier de l'entreprise.

La branche de droite capitalise un savoir-faire technique.

On voit que les plans de tests sont parallélisés avec les phases de conception. Ils devront donc être planifiés très en amont de ce qui est fait dans le cadre d'un cycle en V classique.

4. LES MÉTHODES DE CONCEPTION

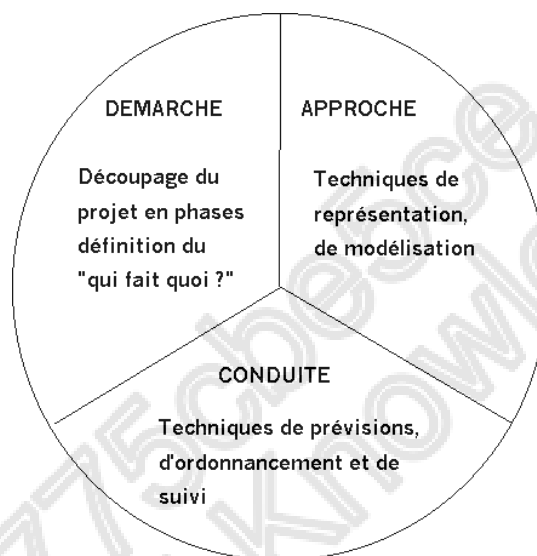
4.1. Présentation

Rappels : Une méthode de projet permet de répondre à la question :

COMMENT FAUT-IL FAIRE LE PROJET ?

Elle recouvre trois aspects la démarche, l'approche, la conduite.

C'est au travers de l'approche que l'analyse du projet est réalisée.



4.2. Les différentes générations de méthodes

La première génération de méthodes de conception, les **méthodes hiérarchiques**, datent des années 60 et décrivent principalement le processus de conception organique. CORIG ou WARNIER sont les représentants les plus connus de cette famille

Les méthodes de conception dites « **d'analyse et de représentation systémique** » apparaissent à la fin des années 70 pour tenter de pallier les nombreuses déficiences des méthodes hiérarchiques. En France, la méthode Merise est utilisée de manière quasi unique au cours des années 1980.

L'évolution de la complexité des logiciels de gestion, ainsi que les nouvelles possibilités offertes par les outils de développement, ont donné naissance dans les années 90 aux **méthodes orientées objet** telle que la méthode UML.

4.3. Les méthodes systémiques - Merise

MERISE, issu des travaux de Tardieu, Rochfeld et Colletti, est le représentant le plus répandu en France de cette génération de méthodes. D'autres méthodes existent, telles que **AXIAL**, **SDM/S** ou **SADT**.

La plus importante caractéristique de **MERISE** est de traduire une **vue globale de l'entreprise** de façon à lier la mise en place d'un système informatisé de gestion à une refonte de l'organisation. Le fondement théorique sur lequel repose cette vision de l'entreprise en termes de systèmes est le système d'information (Voir Chapitre I).

Autre spécificité fondamentale, l'**approche par les données et les traitements** ; traités à égalité lors de la conception du système, MERISE induit toutefois un raisonnement entraînant une **séparation des données et des traitements** qui sont simplement validés de façon croisée pour assurer la cohérence de l'ensemble.

Enfin, parce que la maintenance d'une application informatique met clairement en évidence plusieurs types de problèmes, depuis les modifications qu'entraîne un changement de matériel jusqu'à la refonte complète de l'application qu'exige la mise en place d'une réglementation totalement nouvelle, il a paru essentiel de dégager des niveaux d'abstraction correspondant à ces préoccupations différentes. Ces niveaux d'abstraction sont détaillés dans les pages suivantes.

MERISE distingue pour les données et les traitements les niveaux **conceptuel**, **organisationnel ou logique** et **physique** :

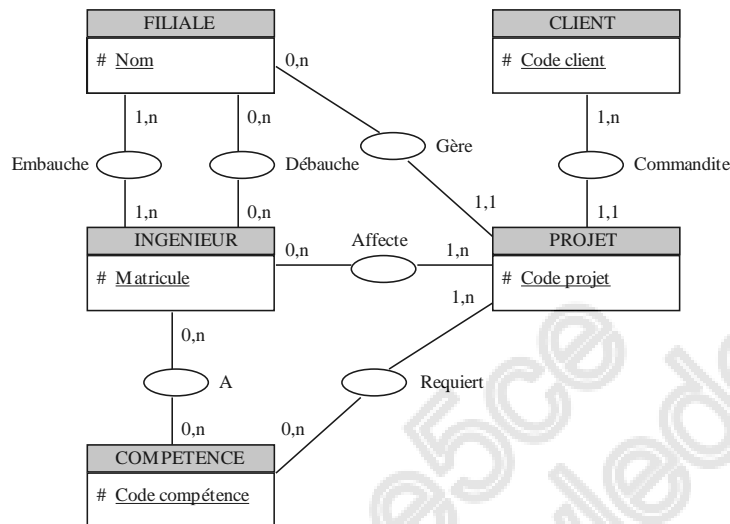
NIVEAU D'ABSTRACTION	DONNEES	TRAITEMENTS	CHOIX PRIS EN COMPTE
CONCEPTUEL	Modèle conceptuel des données (MCD)	Modèle conceptuel des traitements (MCT)	Choix de gestion QUOI
ORGANISATIONNEL	Modèle logique des données (MLD)	Modèle organisationnel des traitements (MOT)	Choix d'organisation QUI, OU, QUAND
PHYSIQUE	Modèle physique des données (MPD)	Modèle physique des traitements (MPT)	Choix techniques COMMENT

Une modélisation MERISE suivra donc 7 étapes :

1. construction du modèle conceptuel des données,
2. construction du modèle conceptuel des traitements,
3. construction du modèle organisationnel des traitements,
4. validation des modèles construits,
5. construction du modèle logique des données,
6. construction du modèle physique des données,
7. construction du modèle physique des traitements.

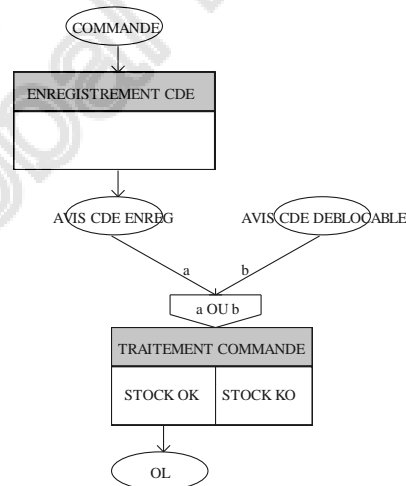
Le **modèle conceptuel des données** est une représentation schématique des données et des rapports instaurés entre elles. Il est l'aspect statique de la représentation du système opérant dans son environnement.

Exemple : MCD simplifié d'une SSII :



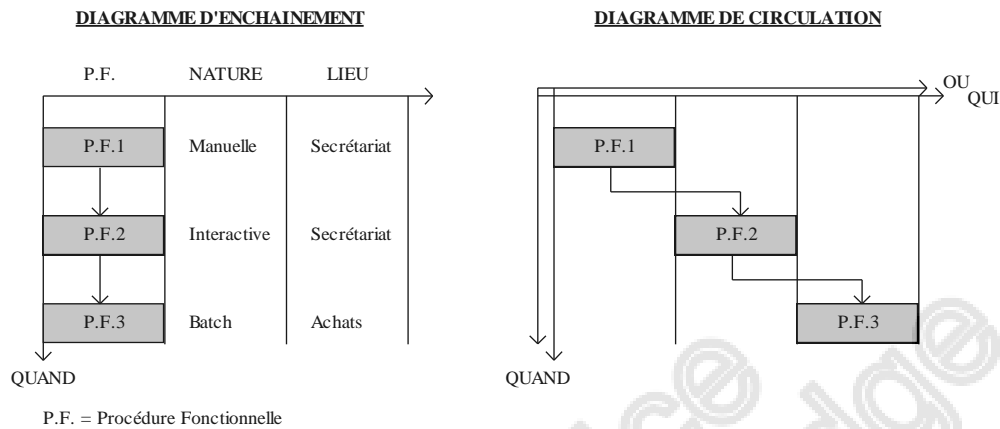
Le **modèle conceptuel des traitements** est une représentation schématique des traitements, indépendante de la façon dont ils seront organisés.

Exemple : MCT traitement d'une commande :



Le **modèle organisationnel des traitements** est une représentation schématique des traitements qui traduit les choix d'organisation de l'entreprise.

Exemple : MOT



Une fois ces trois modèles construits (MCD, MCT et MOT), MERISE impose une phase de **validation** dont le but est de vérifier la cohérence parfaite des visions statiques et dynamiques du système d'information.

Cette étape de validation est fondamentale, elle est la garantie de la cohérence données-traitements. Elle constitue la clé de voûte de l'approche MERISE.

Le **modèle logique des données** est une représentation du modèle conceptuel des données en fonction des possibilités techniques actuelles des matériels et logiciels informatiques.

Le **modèle physique des données** a pour but de traduire le modèle logique des données en un langage compréhensible par la machine et spécifique au SGBD choisi.

Ce langage de définition des données (LDD ou DDL pour *Data Definition Language*) incorpore les paramètres physiques spécifiques au SGBD, choisis de façon à optimiser les accès physiques aux informations.

Le **modèle physique des traitements** a pour but de répondre à la question COMMENT, c'est-à-dire d'aboutir à une solution informatique satisfaisant les objectifs vus dans le MOT et tenant compte des contraintes techniques.

Ce niveau correspond à ce que l'on appelle couramment l'analyse organique et décrit les tâches suivantes :

- réalisation des dessins d'états ou d'écrans définitifs,
- découpage des procédures fonctionnelles en unités de traitement puis en programmes,
- définition des fichiers intermédiaires et des zones de communication entre programmes,
- définition des règles de sécurité, de confidentialité, d'exploitation et de reprise,
- description détaillée du contenu de chacun des programmes.

4.4. Les méthodes orientées objet - UML

L'évolution des méthodes systémiques vers les méthodes objets répond à 3 buts principaux :

- mieux appréhender la conception et donc la réalisation des **logiciels complexes**, de plus en plus fréquents en informatique de gestion, en utilisant conjointement l'abstraction et la modularisation en phase de conception,
- forcer la **réutilisation** des composants logiciels, *via* des contraintes logicielles aussi bien que par une organisation de projet adaptée, cette réutilisation pouvant aller jusqu'à l'acquisition, auprès d'éditeurs spécialisés, de briques logicielles toutes faites, appelées **composants logiciels réutilisables**,
- répondre aux besoins de stockage et d'exploitation de ressources multimédias, telles que des fonds documentaires ou des bordereaux manuscrits.

Pour atteindre ces objectifs, l'approche objet propose, en s'appuyant sur les méthodes plus traditionnelles :

- de redéfinir la notion d'entité, rebaptisée **classe**, de manière à modéliser, conjointement aux attributs (i.e. les propriétés Merise), le comportement de l'entité, c'est-à-dire ces aspects dynamiques et réactifs, appelés méthodes,
- de mener la modélisation **des données et des traitements**, non plus parallèlement, mais simultanément et de façon indissociable ; cette modélisation est représentée sur le modèle objet,
- de créer de **nouveaux types de liens** entre classes, tel que le lien dit d'héritage, dans la mesure où les possibilités offertes par les langages de programmation objet, permettent de traduire simplement ces liens et concourent à la simplicité du source produit,
- d'encourager l'utilisation de **modèles complémentaires**, tels que des modèles dynamiques ou des modèles fonctionnels.

Cette importante évolution induit un changement de langage de programmation pour adopter des langages objets, voire dans certains cas un changement de système de gestion de bases de données.

Principales méthodes de conception objet :

- **UML**, de Rational Software, issue de la fusion des méthodes OMT et Booch, définit la notation utilisée, le processus étant décrit par le **processus unifié**,
- **OMT**, de James Rumbaugh,
- **OD**, de Grady Booch,

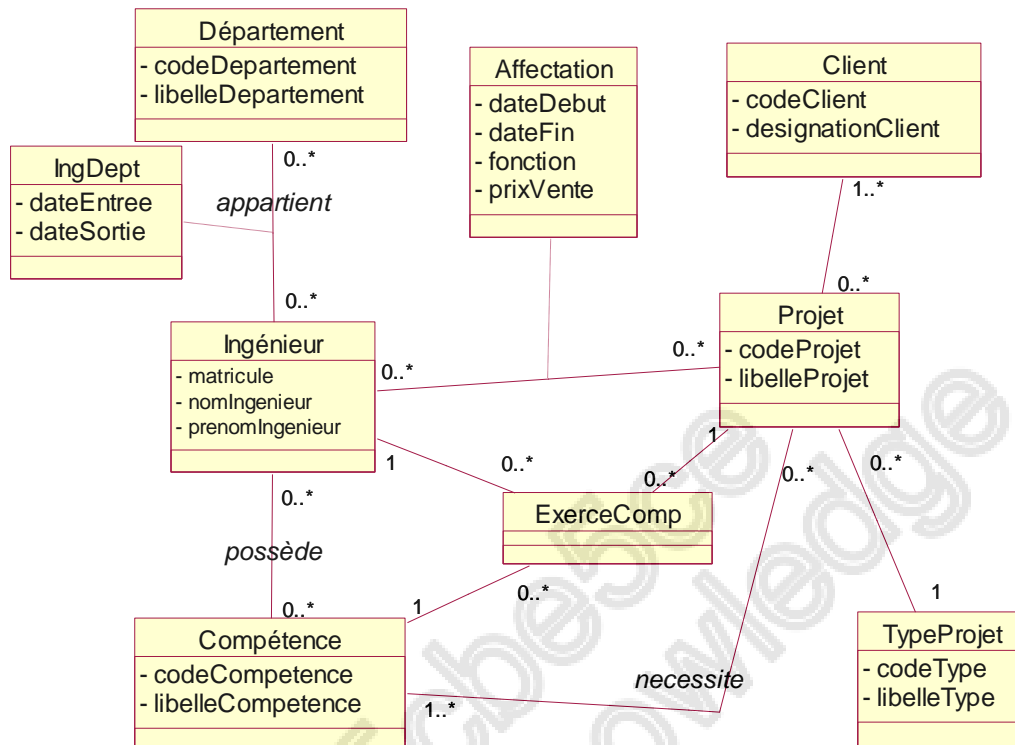
UML (Unified Modeling Language) met à disposition un certain nombre de diagrammes permettant de modéliser le système. Les principaux diagrammes d'UML sont présentés ci-dessous.

Contrairement à Merise, UML **utilise les mêmes diagrammes tout au long de l'analyse**. Ils sont enrichis par les éléments spécifiques de chaque phase.

Diagramme		But	Concepts centraux
Le diagramme des cas d'utilisation (<i>use-case diagram</i>)		Montrer les interactions des acteurs externes avec le système. Recenser les fonctionnalités.	Cas d'utilisation, acteurs
Les diagrammes d'interaction	Le diagramme de séquence (<i>sequence diagram</i>)	Illustrer la dynamique du système en montrant les interactions entre plusieurs objets.	Objets, interactions (messages), flots de contrôle
	Le diagramme de collaboration (<i>collaboration diagram</i>)	Même but que le diagramme de séquence.	Objets, interactions (messages)
Le diagramme de classes (<i>class diagram</i>)		Montrer la structure du système en termes de classes.	Classes, relations (association, héritage)
Le diagramme des objets (<i>object diagram</i>)		Illustrer une portion du diagramme des classes pour clarifier un point particulier.	Objets, relations (liens)
Le diagramme d'états-transitions (<i>state diagram</i>)		Exprimer le cycle de vie des instances de la classe.	Etats, transitions
Le diagramme d'activité (<i>activity diagram</i>)		Décrire l'activité interne d'un objet. (Technique d'algorithme).	Activités, flots de contrôle (décisions, conditions)
Les diagrammes d'Architecture	Le diagramme des composants (<i>component diagram</i>)	Spécifier l'architecture logicielle (représenter les composants et leurs relations).	Composants logiciels, systèmes, sous-systèmes

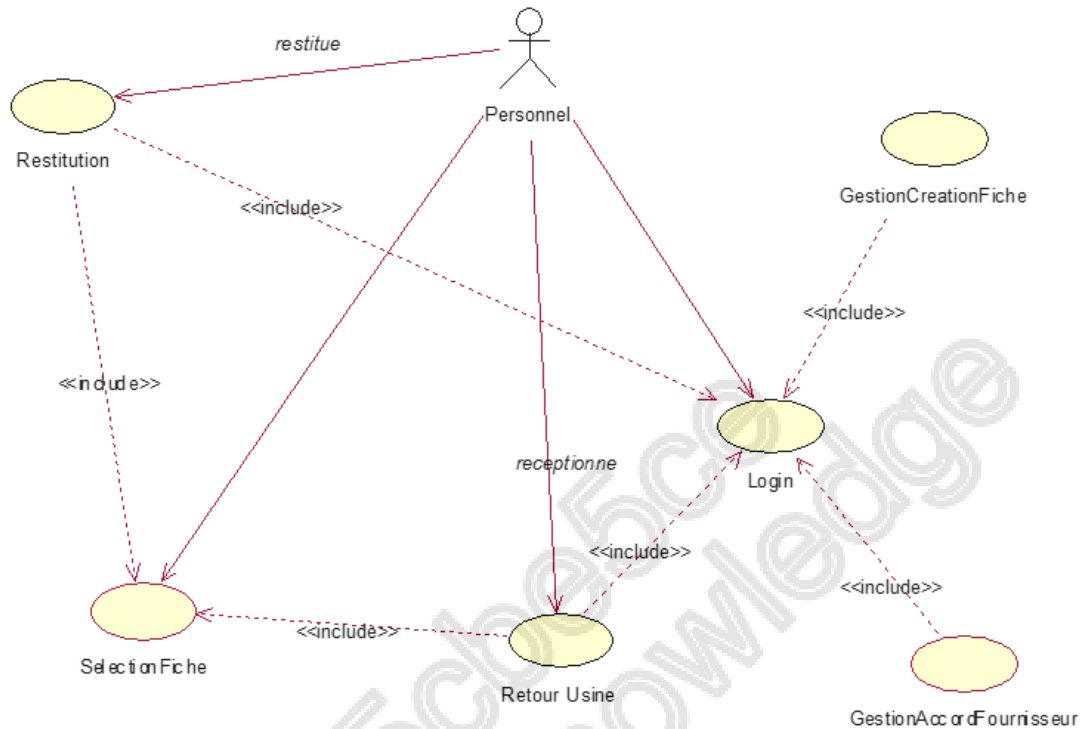
Exemple de diagramme de classes d'une ESN

Montrer la structure du système et les liens entre les classes.



Exemple cas d'utilisation

Montrer les interactions des acteurs avec le système.
Recenser les fonctionnalités.



Exemple d'un diagramme d'architecture

Spécifier l'architecture logicielle, en représentant les composants et leurs relations.

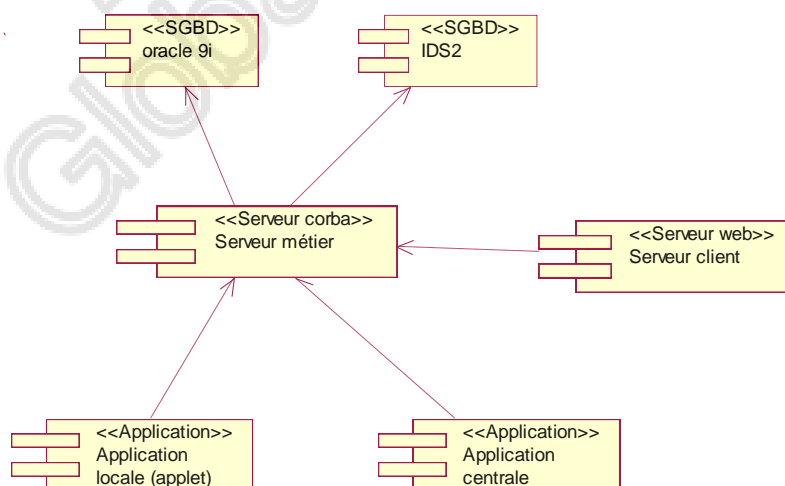


Diagramme de composant

5. LE PROCESSUS UNIFIE

Le processus unifié PU (UP en anglais pour Unified Process), est un **processus de développement** logiciel construit sur **UML**, à base de composants.

L'objectif est de **maitriser la complexité des projets** informatiques, en diminuant les risques.

UML est souvent qualifié de langage de modélisation et permet de « penser objet » au moment de la conception, de la modélisation, pour permettre un développement objet plus aisé. Le cycle de vie du logiciel n'est pas pris en charge par UML, pour la raison suivante :

Comment prendre en compte la diversité des projets, des problématiques, des équipes et des cultures d'entreprise dans une seule et unique méthode ?

C'est à cette question que répond le PU.

Le processus unifié est une méthodologie (Qui-Quoi-Quand) et une méthode (Comment) puisqu'il détermine précisément :

- **Qui** participe au projet
- **Quoi**, que produit le projet (quels livrables)
- **Quand** le livrable est-il réalisé
- **Comment** le livrable est réalisé.

Le processus unifié **privilégie l'interaction et les individus**.

L'accent est mis sur les exigences de l'utilisateur, ce dernier travaillant en étroite collaboration avec les développeurs.

Ce processus utilise **le cycle de développement en Y**.

Le processus unifié est :

- Piloté par les cas d'utilisation,
- Centré sur l'architecture,
- Itératif et incrémental.

Piloté par les cas d'utilisation :

Le but d'un système d'information est de répondre aux besoins de l'utilisateur. Le processus de développement est donc accès sur l'utilisateur. Les cas d'utilisation d'UML permettent d'illustrer ce besoin.

Un cas d'utilisation est une fonctionnalité du système, un objectif utilisateur.

Un scénario est une fiche descriptive d'un cas d'utilisation.

Un acteur est le rôle joué par quelqu'un (ou quelque chose) qui interagit avec le système.

Centré sur l'architecture :

Un produit logiciel dépend du besoin utilisateur mais aussi de contraintes techniques :

- L'architecture matérielle
- Le(s) système(s) d'exploitation
- Le(s) système(s) de Gestion de Bases de Données
- Le(s) réseau(x)
- L'existant
- ...

Itératif et incrémental :

Une itération prend en compte un certain nombre de cas d'utilisation et traite en priorité les risques majeurs.

L'itération est une répétition d'une séquence, dans le but de reprendre la séquence sur des données différentes, à chaque répétition. Dans le cycle en Y l'itération se fera sur l'architecture, l'analyse, et sur la conception, réalisation.

6. DE L'UP VERS LE RUP ET LES METHODES AGILES

Le RUP (Rational Unified Process) est l'une des plus célèbres **implémentations du processus unifié**, livrée clés en main, permettant de donner un cadre au développement logiciel, répondant aux exigences fondamentales préconisées par les créateurs d'UML.

PU et ses diverses implémentations : RUP, XUP (Extreme UP), AUP (Agile UP), EUP (Enterprise UP), 2TUP (Two Tracks UP), EssUP (Essential UP) permettent de prendre en compte les différents projets.

C'est pour préserver une nécessaire adaptabilité au contexte d'entreprise que **PU est** avant tout **générique**.

Les méthodes dites agiles décrivent les processus de développement d'application basés sur la modélisation, la conception et la documentation. Ce sont des **bonnes pratiques** de modélisation que l'on applique à des méthodes déjà existantes. Cependant l'assemblage de PU avec les bonnes pratiques n'est pas évident.

Souvent les démarches les plus utilisées concentrent les décisions les plus importantes en début de projet. Cependant les équipes de développement, travaillant dans des environnements complexes et changeants savent qu'il est difficile de rester fidèles aux décisions initiales.

L'idée des méthodes agiles est :

Accepter le changement, ne pas être rigide lors des développements, rendre le logiciel malléable, ne pas chercher à le concevoir dès le début mais à le faire évoluer au fur et à mesure.

Le logiciel doit fonctionner, même au détriment de la documentation.

L'accent est mis sur la **communication**. Les méthodes agiles ont ainsi des avantages côté conduite de projet :

- Côté **utilisateur**, car il est toujours mis au courant dans un laps de temps court.
- Côté **équipe de développement**, qui est considérée comme plus importante que les moyens matériels.

Principales méthodes agiles :

- Extreme Programming **XP**,
- Processus Urbanisant les Méthodes Agiles PUMA,
- Adaptive Software Development ASD,
- Crystal clear,
- Dynamic Systems Development Method DSDM,
- **Scrum**.

7. QUELQUES CHIFFRES CONCERNANT LES PROJETS

En 2009, le cabinet Standish Group publie les chiffres suivants:

Environ 30% des projets informatique n'aboutissent pas

Plus de 30% des projets informatique dépassent le délai initial

Environ 50% des projets informatique dépassent leur budget initial

Et ils coûtent alors en moyenne 189% de ce budget initial !

Moins de 40% seulement réussissent dans les délais et les budgets

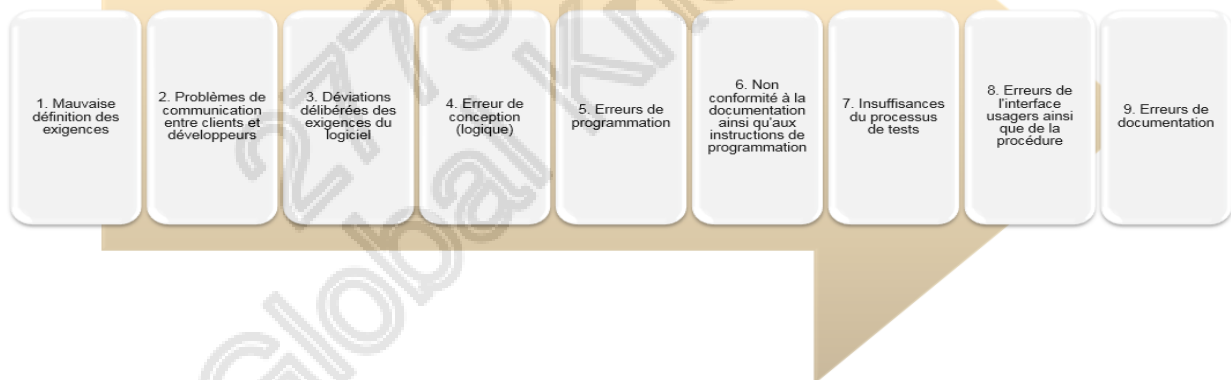
Mais ils couvrent alors seulement 42% des fonctionnalités initialement prévues !

Les causes de ces échecs sont souvent dues à 2 facteurs :

Le volet humain négligé,

La gouvernance mal adaptée.

Les sources d'erreur des logiciels



Nécessité de mettre en place l'assurance qualité pour garantir :

- Une meilleure évaluation de la performance des SI,
- Une gestion des ressources des SI plus efficace,
- Une gestion des risques plus pertinente,
- Une amélioration de la valeur des services de l'entreprise par le biais de ses SI,
- Une meilleure adéquation des SI à la stratégie de l'entreprise.

8. REFERENTIELS DE QUALITES

8.1. Présentation

Ces référentiels ont été mis en œuvre pour garantir la reproductibilité d'un processus projet et l'adapter en fonction des évolutions.

Face à une externalisation, voire délocalisation (nearshore, offshore) de plus en plus fréquente des développements informatiques, les référentiels qualités sont devenus une nécessité. Certains sortent du lot :

- CMMI Evaluation d'entreprise sur les processus informatiques,
- ITIL Evaluation de personne sur l'activité Informatique,
- PMP Evaluation de personne sur le management de projet,
- Lean Six Sigma Evaluation de personne multi-activités,
- Prince2 Evaluation de personne sur le management de projet

CMMI est un référentiel d'évaluation de la capacité à gérer et terminer un projet correctement.

ITIL est une bibliothèque de bonnes pratiques dans le domaine de la gestion des services informatiques. ITIL indique ce qu'il faut faire pour gérer un département informatique mais pas comment faut le faire.

PMP (Project Management Professional) est une certification en gestion de projet reconnue internationalement, délivrée aux chefs de projets par le PMI (Project Management Institute).

Lean Six Sigma permet à une personne physique d'obtenir une certification pour ses compétences en matière d'amélioration de processus.

Prince2 (Project IN Controlled Environment), permet à une personne physique d'obtenir une certification en matière de management de projet.

8.2. Périmètres des dispositifs

Entité évaluée	Dispositif	Activités concernées	Secteur d'activité
Entreprise ou entité	EFQM ISO 9001 Sarbanes-Oxley SAS 70	Toutes	Tous domaines
Processus	ISO 15504		
Service	Certification de services		
Personne	Six Sigma		
Entreprise ou entité	ISO 27001	Sécurité des systèmes d'information	
Processus	SSE-CMM		
Personne	CobIT-CISA CobIT-CISM CISSP		
Produit	ISO 15408 Critères communs ITSEC		
Personne	AFITEP-CDP AFITEP-CGP PMP PRINCE2	Management de projet	
Entreprise ou entité	ISO 20000 TickIT eSCM	Informatique	
Processus	CMMI SW-CMM		
Produit	ISO 25051		
Personne	ITIL PCIE SWEBOK		
Entreprise ou entité	HAS	Toutes	Santé
	TL 9000 Trillium		Télécommunications

8.3. ITIL - Information Technology Infrastructure Library

Objectifs d'ITIL

L'objectif d'ITIL est de doter les directions des systèmes informatiques (DSI) **d'outils** et de **documents** leur permettant **d'améliorer la qualité** de leurs prestations.

Quand on dit améliorer les prestations cela veut dire améliorer la satisfaction de leurs utilisateurs cités en tant que clients, tout en répondant au mieux aux objectifs stratégiques de l'entreprise.

On doit considérer le service informatique comme un ensemble de processus étroitement liés, au service des clients.

Origine d'ITIL

ITIL (*Information Technology Infrastructure Library*) est un cadre de référence, proposé par l'OGC (Office of Government Commerce) du Royaume-Uni rassemblant, dans un ensemble de guides, **les meilleures pratiques** en matière de management des services informatiques.

La bibliothèque ITIL a été initiée dès le début des années 80 par le gouvernement britannique afin d'améliorer le service rendu par leurs directions informatiques. ITIL s'est implanté aux USA, via des entreprises comme Accenture, Deloitte, Ernst & Young, Hewlett-Packard,

Couverture d'ITIL

ITIL couvre plusieurs domaines, permettant d'aborder l'ensemble des problématiques géré par les DSI. Le cœur de la démarche ITIL est couvert par les deux premiers domaines.

- Service Support
- Service Delivery
- Infrastructure Management
- Applications Management
- Service Management
- Business Perspective
- Business Requirements
- Technology

◆ **Service Support** le soutien des Services

Comment s'assurer que le "client" a accès aux services informatiques appropriés.

- ⇒ **Gestion des configurations**, gérer l'infrastructure technologique en faisant un état des lieux de l'existant afin de mieux le gérer et le faire évoluer.
- ⇒ **Gestion des incidents**, mieux détecter les incidents, améliorer le délai de résolution des incidents selon leur criticité sur le fonctionnement de l'entreprise.
- ⇒ **Gestion des problèmes**, mieux gérer les problèmes récurrents et mettre en œuvre des solutions de prévention afin de réduire leur occurrence, voire les supprimer.
- ⇒ **Gestion des changements**, mettre en œuvre des démarches de conduite du changement afin d'anticiper les effets de bord.
- ⇒ **Gestion des mises en œuvre**, s'assurer de l'adéquation du service avec les besoins métiers.
- ⇒ **Gestion de la disponibilité**, assurer un niveau de disponibilité suffisant à un coût raisonnable.

◆ **Service Delivery** la fourniture des Services

Les services devant être fournis pour répondre aux besoins de l'entreprise de manière adéquate.

- ⇒ **Gestion des niveaux de service**, maintenir un certain niveau de qualité de service grâce à des contrats de service renégociés périodiquement.
- ⇒ **Gestion des capacités**, vérifier l'adéquation des capacités et performances avec les exigences actuelles et à venir.
- ⇒ **Gestion de la continuité des services IT**, définir et mettre en œuvre des délais contractuels pour la reprise après incident.
- ⇒ **Gestion financière des services IT**, gérer la rentabilité des moyens mis en œuvre pour fournir le service.

Bénéfices d'ITIL

Lorsque l'on utilise ITIL, on constate le plus généralement :

- Satisfaction des utilisateurs (personnel et clients),
- Clarification des rôles,
- Amélioration de la communication entre les services,
- Mise sous contrôle des processus avec des indicateurs pertinents et mesurables, permettant d'identifier les leviers pour réaliser des économies,
- Meilleure compétitivité,
- Sécurité accrue (disponibilité, fiabilité, intégrité),
- Capitalisation des données de l'entreprise ,
- Optimisation de l'utilisation des ressources.

Certification ITIL

La certification aux bonnes pratiques ITIL se fait **pour des individus**. Ainsi, une entreprise ne peut pas être certifiée ITIL, même si ses processus de production informatique suivent l'intégralité des pratiques ITIL.

Le personnel peut être certifié ITIL, cela valide sa connaissance et sa compréhension du référentiel.

Il existe 3 niveaux de certifications :

- **Foundation Certificate** : Certification de premier niveau accordée après un test sous forme de questions à choix multiples, qui suit normalement une formation de 2 à 3 jours par un formateur accrédité. Ce certificat valide une connaissance générique des fondamentaux d'ITIL.
- **Practitioners Certificates** : Certifications accordées pour une discipline spécifique après une formation de 2 à 3 jours chez un formateur accrédité et un test sous forme de questions à choix multiples fondé sur un cas concret. Le Foundation Certificate est un prérequis.
- **Managers Certificate** : Certification accordée après deux tests de 3 heures qui suit une formation de 10 jours par un formateur accrédité. Le Foundation Certificate est un prérequis.

8.4. CMMI - Capability Maturity Model Integration

Objectifs de CMMI

L'objectif de CMMI est d'encourager les entreprises qui développent des logiciels, à mettre leurs processus sous contrôle, à les améliorer de façon continue et d'évaluer leur niveau de maturité sur une échelle de cinq niveaux.

Origine de CMMI

Le CMMI, pour Capability Maturity Model Integration (Modèle intégré du niveau de maturité), est une extension de la spécification CMM, créée pour le ministère de la Défense américain en 1989 afin de déterminer si un projet interne ou tiers serait terminé dans les temps, selon le budget et les spécifications.

En 2001, le SEI (Software Engineering Institute) propose l'appellation CMMI. CMMI étend CMM en y élargissant le périmètre pour pallier les manques de CMM.

La version actuelle du modèle CMMI a été réactualisée en 2006, elle n'intègre pas la gestion des ressources humaines.

Couverture de CMMI

CMMI propose un référentiel des meilleures pratiques en matière de développement logiciel. Bonnes pratiques liées :

- à la gestion,
- au développement,
- à la maintenance d'applications
- à la maintenance de systèmes.

Ces bonnes pratiques sont regroupées en 24 processus, eux-mêmes regroupés en 4 types (Process Management, Project Management, Engineering et Support) et 5 niveaux de maturité.

◆ Niveaux de maturité

- **Initial** : Les facteurs de réussite des projets ne sont pas identifiés, la réussite ne peut donc être répétée.

Le niveau le plus bas montre que l'organisation n'est pas prête, et le projet pas stable. Ce dernier dépend d'une poignée de personnes, qui ne font pas appel à des processus éprouvés. Il se peut cependant que le projet aboutisse, mais en dépassant certainement le budget et le temps alloués. Le projet ne construit pas sur les succès passés.

- **Reproductible** : Les projets sont pilotés individuellement et leurs succès sont *répétables*.

Le projet construit sur ce qui a été appris précédemment, en faisant appel à une certaine discipline et à une gestion de projet basique. De fait, le projet est géré selon les plans, avec étapes-clefs et vérification des coûts et des fonctionnalités.

- **Défini** : Les processus de pilotage des projets sont mis en place au niveau de l'organisation par l'intermédiaire de normes, procédures, outils et méthodes.

Ce n'est plus le projet qui dispose d'une bonne discipline, mais l'ensemble de l'organisation, de manière cohérente. Tous les projets s'en trouvent améliorés.

- **Géré** : La réussite des projets est quantifiée. Les causes d'écart peuvent être analysées.

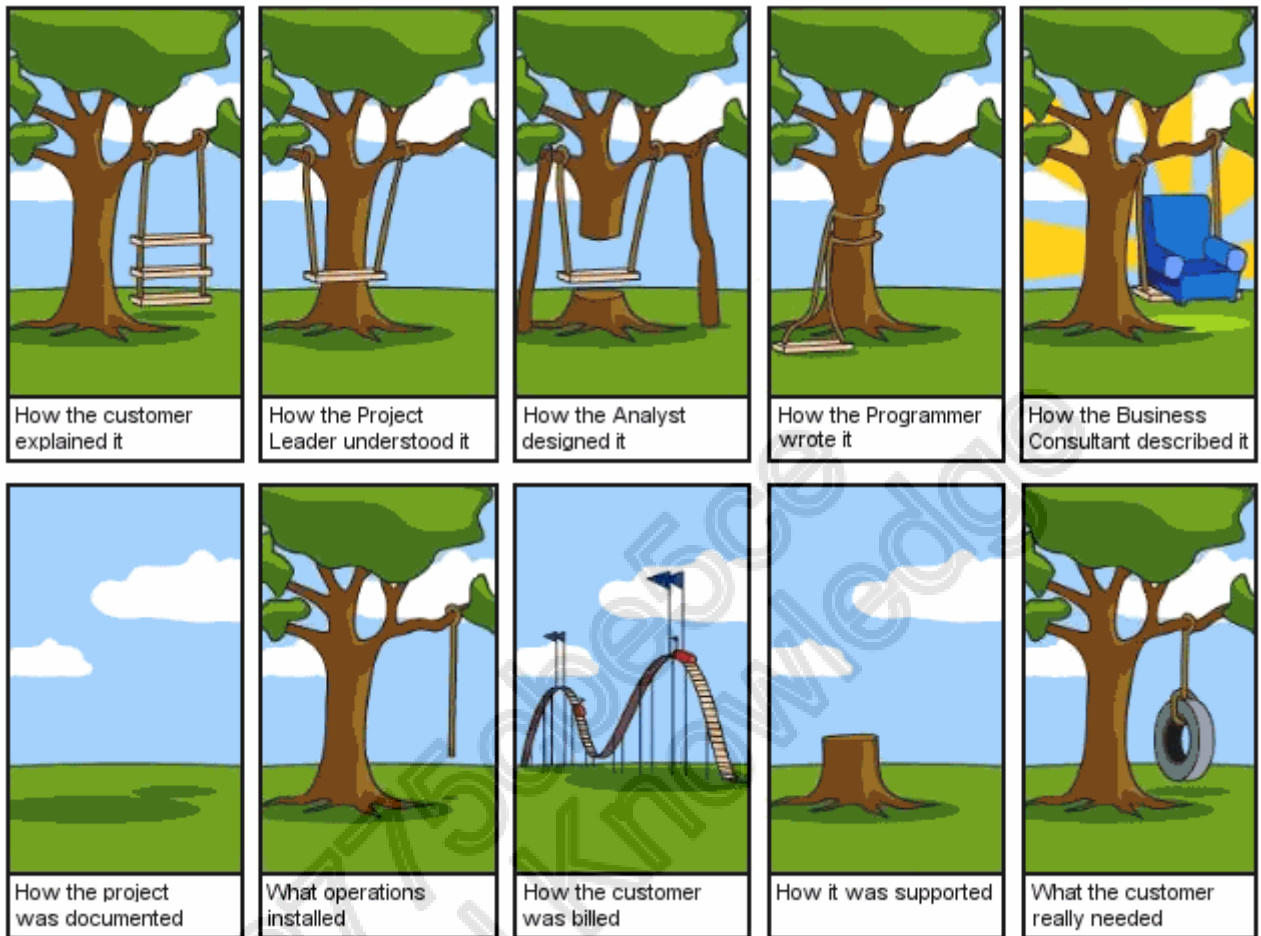
Les efforts de mesure et de gestion autorisent un contrôle sans effort du développement, avec capacité d'ajuster et adapter des projets précis sans troubler les autres. Les performances des processus sont prévisibles en quantité comme en qualité.

- **Optimisé** : La démarche d'optimisation est continue.

Les processus sont constamment améliorés de manière incrémentale et innovante. Les objectifs sont revus en permanence pour rester proches des besoins du marché. Les évolutions sont anticipées et gérées de bout en bout.

CHAPITRE VI POUR CONCLURE

1. UN PEU D'HUMOUR



2. POUR EN SAVOIR PLUS

Wikipédia et son portail pour l'informatique permet d'obtenir des informations tant de bases que détaillées.

<https://fr.wikipedia.org/>

Le journal du net, contient des définitions et articles intéressants mais aussi des statistiques : www.journaldunet.com

<http://www.journaldunet.com/web-tech/chiffres-internet>

Developpez.com et openClassRoom (anciennement le site du zéro) sont des sites spécialisés dans le monde du développement, plusieurs tutoriaux sont proposés. Des forums d'entraide, des FAQs sont aussi présentes

<http://java.developpez.com/>

<https://openclassrooms.com/>

Vous pouvez consulter la page officielle chez Oracle pour le monde Java:

<http://www.oracle.com/technetwork/java/index.html>

Celle de Microsoft pour .Net

IBM offre aussi de nombreuses informations avec son centre de connaissances :

<http://www-01.ibm.com/support/knowledgecenter/>

Et n'oubliez pas Google qui reste notre ami.

<https://www.google.fr/>

Index

.		ascenseur	72
.		ASP	83
.		ASP.Net	83
.		assemblage	78
.Net	62, 88	assembleur	78
.		Axial	111
1		B	
10BaseT	22	baie	14
3		baie de disques	27
3-tiers	53	base de données	64, 66
A		Basic	78
ABAP	79	batch	114
abstraction	104, 111, 112	Big data	70
Access	68	Big Data	66
ActiveX	47	bilan	102, 107
Ada	78	binaire	77
adresse IP	49	binary digit	77
affectation	98	bit	77
AFNOR	106	Blackberry	19
AFNOR Z67-111	106	Boehm	106
Agile UP	122	Borland	79
AIX	18	branches	21
AJAX	84	BSC	22
analyse	98, 103, 107	Bull	18
anneau à jeton	21	bus	21
annuaire	88	Business Analytics	70
ANSI	79	Business Intelligence	69
antivirus	51	C	
Apache	48	C	78
appareils mobiles	19	C#	78
Apple	18, 75	C++	78
Applets	47	cahier des charges	97
application de bureau	72	capitalisation	102
arbre	21	cas d'utilisation	121
architecture 3 tiers	53	Cassandra	70
architecture centralisée	32	centre de données	14
architecture décentralisée	32	CGI	83
architectures n-tiers	53	CGX	29
Arpanet	46	chevaux de Troie	51
AS400	18	chiffre binaire	77
		Cisco	25
		classe	116

clé d'accès	65	DAO	91
client léger	73	Dart	84
client lourd	72	data center	14
client riche	74	Data Definition Language	115
Client/serveur	33	DB2 UDB	68
Client/Serveur	60	DCOM	61
clonage	28	DDL	115
cloud computing	35, 57, 70	de modèle de conception	91
cluster	27	Debian	18
CMMI	124	délocalisation	124
Cobol	78	Delphi	79
CODASYL	66	démarche de projet	95
codes EAN	80	démarche itérative	104, 106
Colletti	111	démarche linéaire	104
communication de projet	98	démarrage du projet	98, 103
communication peer-to-peer	23	Design Pattern	91
commutateur	25	desktop	72
compilation	77, 78	devops	11
composant logiciel réutilisable	116	Direction des Systèmes d'Information ..	11
composants Hardware	14	disponibilité	15
Computer Associates	68	distribution	18
concentrateur	21	DMZ	50
conception	98, 100, 103, 107	donnée	9, 64, 65, 66
conception externe	100	DOS	78
conception fonctionnelle	100	DPS 7	18
conception interne	100	DPS 7000	18
conception organique	100	DPS 8	18
conceptuel	112	DPS 9000	18
confidentialité	15	DreamWeaver	79
cookie	51	DSI	11
Corba	61	DTD	81
Corig	110	Dynamic HTML	82
CouchDB	70		
couche application	22	E	
couche liaison	22, 25	EANCOM	80
couche physique	22	Eclipse	90
couche présentation	22	EDI	80
couche réseau	22	EDIFACT	80
couche session	22	EMC	29
couche transport	22	enregistrement	64, 65, 66
Crédit Agricole - Greenfield	16	ERP	69
cryptage	50	Escalla	18
CSS	82	estimation	98
cycle de développement en Y	120	ESX	29
cycle en V	105	ETEBAC	80
		Ethernet	21, 25
D		étoile	21
d'Adobe	79	évaluation	98, 102, 103
d'Oracle	76	exécutable	77

externalisation	124
extranet	36, 50

F

Facebook	70
fenêtre	72
ferme	27
fichier	64, 65
fichier séquentiel	65
fichier séquentiel indexé	65
Firefox	47
firewall	26, 50
Forefront	51
Forms	79
Fortran	78
framework	91
France Télécom	20
FTP	46

G

GAP	78
GCOS 7	18
GCOS 8	18
Gencod	80
Gentoo	18
Go	84
Google	84
Google Chrome	47
grand système	18
Groovy	84

H

Hadoop	70
hardware	10
HDLC	22
hébergeur	16
Hewlett Packard	18
Hibernate	91
hiérarchique	21, 66
hoax	51
HP 9000	18
HP-UX	18
HTML	48, 82
HTTP	46, 48
hub	21, 25, 27

I

IaaS	57
IBM	18, 66, 68, 76, 79
IBM i	18
icône	72
IDE	87
IHM	87
index	65
informatique dans les nuages	57
infrastructure	14
installation	98, 102, 103
intégration	107
interactif	114
interface mode caractère	71
interface mode graphique	72
interface universelle	47
interface utilisateur	71
interface Web	73, 74, 76
interfaces graphiques	73, 74
interfaces riches	84
International Standard Organization	22
Internet	21, 36, 46
Internet Mobile	75
intranet	36, 50
Intranet	39
iOS	19
IP	22, 23
iSeries	18
ISO	22, 79
isolation	28
ITIL	124

J

J2EE	88
JAD/RAD	107
Java	78
Java EE	62, 88
Java Enterprise Edition	88
Java RMI	61
JavaScript	48, 82, 84
JBoss	76
jQuery	84
JSF	91
JSP	83

K

Knoppix 18

L

l'intégrité 15
 L4G 79
 la zone démilitarisée 50
 LAN 20, 36
 langage de programmation 77
 langage de quatrième génération 79
 langage de script 82
 langage évolué 78
 langage objet 79
 LDD 115
 Lean Six Sigma 124
 legacy system 18
 lien d'héritage 116
 Linux 18
 livrables 120
 load-module 77
 Local Area Network 20
 logiciel malveillant 51
 loi de Moore 32

M

mail 46
 maillé 21
 maîtrise d'œuvre 97
 maîtrise d'ouvrage 97
 malware 51
 Mandriva 18
 MapReduce 70
 maquettage 104, 106
 maquette 106
 maquette de plate-forme 106
 maquette expérimentale 106
 MAU 21
 McAfee 51
 MCD 112, 113, 114
 MCT 112, 113, 114
 MDM 69
 mémoire 17
 menu déroulant 71
 Merise 100, 110, 111
 méthode agile 122

méthode
 d'analyse et de représentation
 systémique 110
 méthode hiérarchique 110
 méthode orientée objet 110
 méthodologie de conduite de projet 95
 Microsoft 18, 68, 75, 76, 79, 83, 88
 Microsoft IIS 48
 Microsoft Internet Explorer 47
 Microsoft Surface 19
 Microsoft Virtual Server 29
 middleware 88
 mise en exploitation 107
 MLD 112, 115
 mode diffusion 20
 mode point à point 20
 modèle 3-tiers 53
 modèle conceptuel des données 112, 113
 modèle conceptuel des traitements 112, 113
 Modèle du Gartner Group 37
 modèle dynamique 116
 modèle en couche 61, 62, 63
 modèle fonctionnel 116
 modèle logique des données 112, 115
 modèle organisationnel des traitements 112, 114
 modèle OSI 22, 24
 modèle physique des données 112, 115
 modèle physique des traitements 112, 115
 modélisation 112
 module chargeable 77
 MongoDB 70
 MOSS 76
 MOT 112, 114, 115
 MPD 112, 115
 MPE 18
 MPT 112, 115
 MVC 91
 MVS 18

N

NAS 26
 navigateur 47
 nearshore 124
 NetBeans 90
 nœud 27
 Norton 51

NoSQL 70

O

Office 19
 offshore 124
 OLAP 69
 OMT 116
 OOD 116
 Open Source 76, 90
 open system 18
 Open System Interconnection 22
 Opéra 47
 Oracle 18, 68, 79
 Orange 20
 organisationnel 112
 OS/390 18
 OS/400 18
 OSI 22, 25
 outils de développement 88

P

PaaS 57
 Palm 19
 paquets 23
 Partitionnement 28
 Pascal 78
 périphérique 17
 PERL 83
 phablettes 19
 PHP 83
 physique 112
 Pile TCP/IP 24
 PL/1 78
 PL/SQL 79
 planification 98
 plate-forme de développement 88
 plates-formes 62
 PMI 124
 PMP 124
 point à point 21
 portail d'entreprise 76
 Power Systems 18
 PowerBuilder 79
 Powersoft 79
 Prince2 124
 procédure fonctionnelle 114
 processeur 78

processus
 unifié 116, 120
 projet 95
 protocole 22
 prototypage 104, 106
 prototype 106
 prototype de plate-forme 106
 prototype expérimental 106
 proxy-HTTP 50

R

rack 14
 RAD 87
 Rational 116
 réalisation 98, 101, 103, 107
 recette 97, 101
 RedHat 18
 réestimation 98
 référentiels qualités 124
 relationnel 66
 répartiteur 21
 réplication 28
 requête 49
 réseau 20, 22, 66
 réseau étendu 20
 réseau local 20
 réseau SAN 27
 Réseau virtuel 28
 réseaux 36
 réutilisation 116
 revamping 38
 rhabillage d'applications 38
 Rochfeld 111
 routage 22
 router 25
 RPG 78
 RS/6000 18
 RS232 22
 rubrique 64, 65
 Ruby 78
 RUP 122

S

S/390 18
 SaaS 57
 SADT 111
 Safari 47

SAN	26
Scala	84
Scrum	122
SDM/S	111
séparation des données et des traitements	111
serveur d'applications	53
serveur HTTP	48
serveur Web	48
serveurs d'applications	48, 88
service Études	11
service Exploitation	11
service Production	11
service Système	11
service Web	52
Services Oriented Architecture	55
SGBD	66
SharePoint	76
Slackware	18
Smalltalk	78
smartphones	19
SMTP	46
SOA	55
SOAP	52, 63
software	10
source	77, 78
spirale de Boehm	106
Spring MVC	91
SQL	66
SQL PL	79
SQL Server	68
Struts	91
studios de développement	87
suivi	98
Sun	88
Sun Microsystems	68
Sun-Solaris	18
SuSe	18
Swift	78
switch	25, 27
Sybase	79
Symantec	51
Symbian	19
System i	18
System p	18
Système d'exploitation hôte	28
Système d'exploitation invité	28
Système de Gestion de Base de Données	66
système de pilotage	8
système d'exploitation	17
système d'information	8
système libre	18
système opérant	8, 113
système ouvert	18
système propriétaire	18
Système propriétaire	32
T	
tablettes	19
tâche	17
Tardieu	111
TCP	22, 23
téléchargement de fichier	46
test	107
test d'intégration	101
test opérationnel	101
test système	101
test unitaire	101
Tier	16
Token Ring	21
topologie de réseau	20
trame	23
Transact-SQL	79
Two phase commit	42
U	
UDDI	52
UML	100, 110, 116, 120
une grappe de serveurs	27
Unified Process	120
Unix	18, 78
Uptime Institute	16
URL	49
V	
V24	22
validation	107, 112, 114
VB.Net	79
VDE	87
vers	51
virtualisation	28
Virtuoizzo	29
virus	51
Visual Basic	79

Visual C++	79
Visual J++	79
VisualAge C++	79
VisualAge Cobol	79
VisualAge Java	79
VisualAge Smalltalk	79
VisualStudio	90
VMWare	29
VPN	20

W

W3C	52
WAN	20, 36
Warnier	110
Web	49, 52
Web 2.0	84
Web dynamique	48, 83
Web Service	52
Web statique	48, 82
WebLogic	76
WebSphere	76
Wide Area Network	20
Windows 10 Mobile	19
Windows 2003, 2008, 2010...	18
Windows 7, 8	18
Windows Mobile	19
Windows Phone	19
Windows Server	18
Windows vista	18
Windows XP	18
World Wide Web	46
WWW	46

X

X25	22
Xen	29
XHTML	82
XML	63, 81
XQuery	63, 81
XSL	81
XUP	122

Z

z/OS	18
z/Series	18