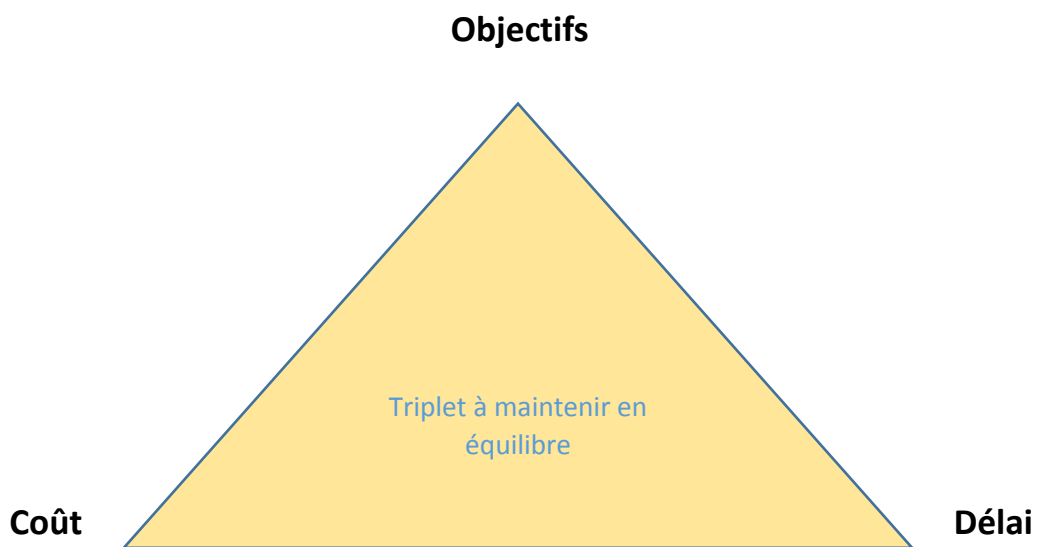


# Cours : Gestion De Projets

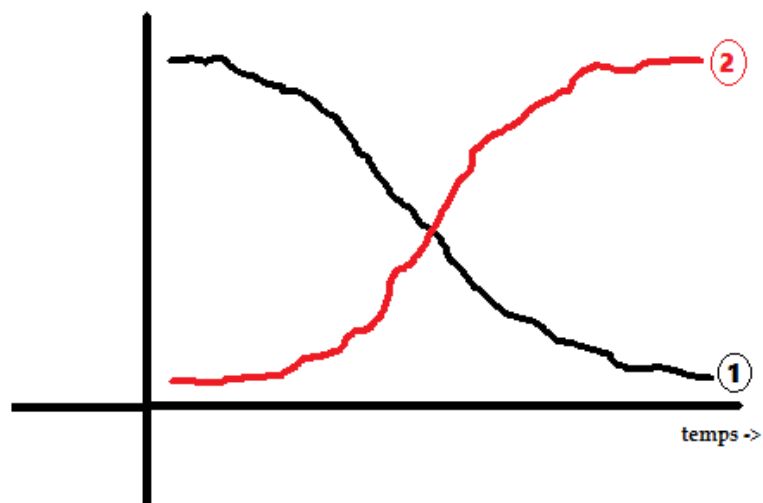
## Chapitre 1 : Notions Fondamentales

### ➤ Qu'est-ce qu'un projet ?

Un **projet** consiste en un ensemble de tâches ayant un but précis et qui devrait être réalisé en un temps limite.



### ➤ Pourquoi la gestion de projet ?



Paradoxe de la gestion de projet

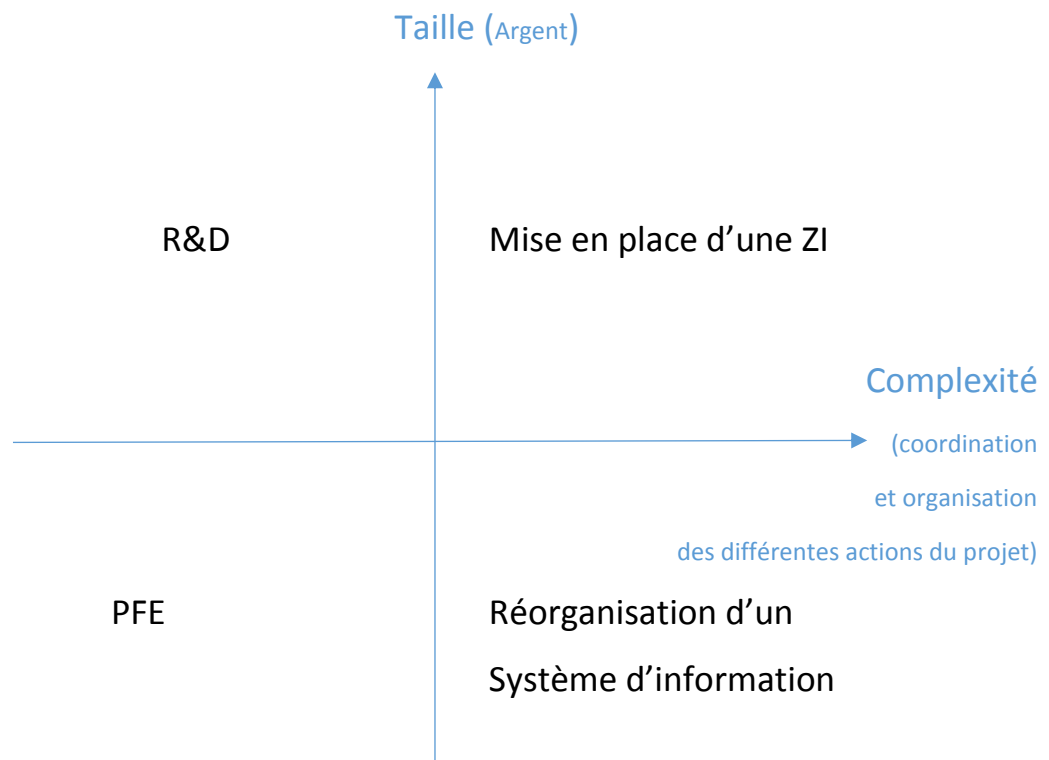
- 1- Capacité d'action sur le projet.
- 2- Connaissances sur le projet.

### **But de la GP :**

Essayer d'avoir un maximum de connaissances au début du projet pour avoir un maximum de possibilité d'action.

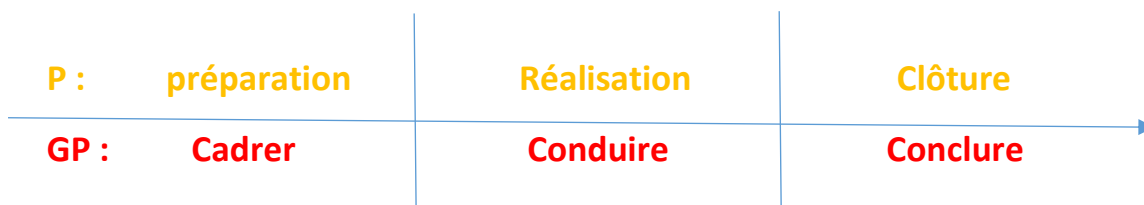
Les projets sont irréversibles c'est-à-dire qu'une décision prise au début ne peut plus être changée une fois réalisée.

### ➤ **Quel sont les types de projets ?**



### ➤ **Comment organiser la gestion d'un projet ?**

#### **Phases 3C : « Cadrer – Conduire – Conclure »**



Les phases **3C** reviennent dans tous les projets mais de manière différente selon la nature du projet

### 1- Phase cadrer :

- Etude préliminaire :
  - Étude du marché
  - Étude de faisabilité
- Montage :
  - Etude fonctionnelle
  - Découper les lots du travail
  - Elaboration de la matrice RACI
  - Planification du projet

### 2- Phase conduire :

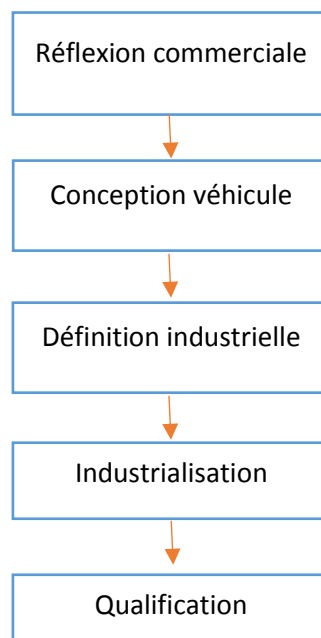
- Pilotage (vérifier que ce qui est en train de se réaliser est bien ce qui a été préparé)

### 3- Phase conclure :

- La recette du projet
- La mise en condition
- La formation des utilisateurs
- Post-Mortem

## Ingénierie Séquentielle :

Exemple : Ingénierie automobile



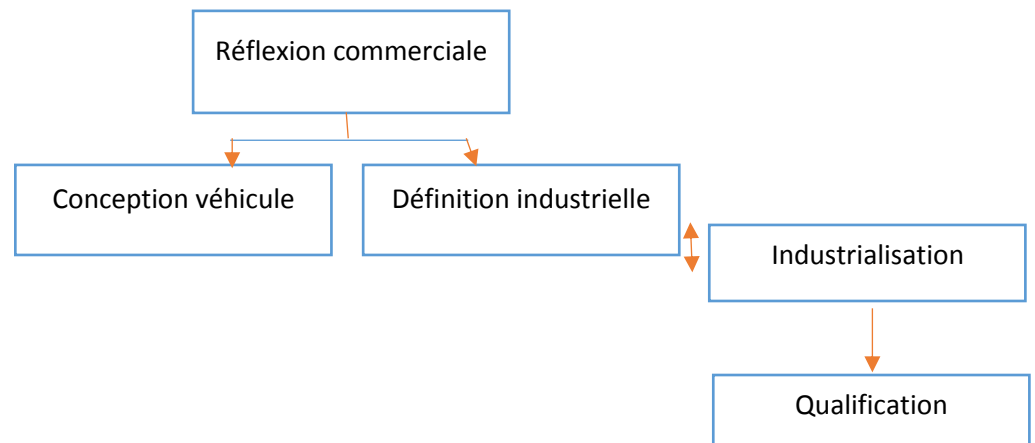
Supposons que la durée de chaque phase est de 60 jours

⇒ La durée totale est donc égale à 300 jours

Problème de l'ingénierie séquentielle :

- Time To Market
- Problème des retours arrière

### Ingénierie Simultanée : (parallèle)



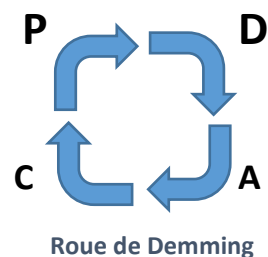
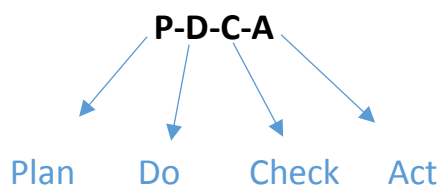
- 23% de temps gagné par rapport à l'IS

### ➤ Y a-t-il un vocabulaire important en gestion de projet ?

Dans une première approche on a deux grandes catégories d'acteurs qui sont :

- Maître d'ouvrage MOA : (Commanditaire – Bailleur de fond – Client) c'est celui qui paye, qui lance l'appel d'offre.
- Maître d'œuvre MOE : C'est celui qui réalise, il sous-traite éventuellement et coordonne entre les différents acteurs.

### ➤ Comment se déroulent les actions d'un projet ?



### **Toute action doit se faire en 4 étapes :**

On commence par fixer les objectifs et planifier (Plan), ensuite vient la deuxième étape où se déroule l'action (Do). En troisième étape on vérifie que les objectifs ont bien été réalisés dans le délai (Check), et en fin on agit : si les objectifs n'ont pas été atteints alors on lance un nouveau cycle, sinon on généralise que l'on a appris aux autres projets et on passe à la suite (Act).

**Conseil** : dans la gestion de projet il faut toujours anticiper et être flexible dans la première phase du projet.

## **Chapitre 2 : outils de la GP**

Il est important de savoir que la gestion de projet repose sur un découpage de travail à faire en précisant :

- **Ce qui doit être fait (taches).**
- **Par qui (ressources).**
- **Les résultats attendus (livrables).**
- **Comment vérifier l'avancement du travail (jalons).**

### ➤ **La planification : - taches**

C'est une action à mener pour obtenir un résultat, a chaque tache définie dans le projet il faut associer :

- **Un objectif précis est mesurable (livrable).**
- **Des ressources.**
- **Une charge exprimée en journée-homme.**
- **Une durée et une date de début.**

**Remarque :** les taches d'un projet seront reliées entre elle par des relations de précedence

### ⇒ **L'effet tunnel**

On parle d'un effet tunnel lorsqu'une longue période se déroule entre le début du projet et le moment ou on peut voir des résultats, le problème c'est le fait qu'on a pas de visibilité sur cette période.

On ne peut pas connaitre l'état d'avancement du projet

Et donc il faut passer par l'analyse des écarts

- **Principe des écarts :**

Dans la réalité on n'arrive jamais à prévoir avec exactitude ce qui va se passer, et on voit clairement la nécessité du cycle PDCA.

L'analyse des écarts se fait en posant des objectifs intermédiaires aux projets (jalons).

### ➤ **Jalon (MilesStone / PierreMiliaire)**

C'est généralement un livrable intermédiaire du projet, mais il peut aussi être un évènement clé du projet ou une date importante, chaque jalon permet de vérifier que les conditions nécessaires à la poursuite du projet sont réussies.

➤ **Notion de livrable**

Un livrable est tout résultat mesurable ou vérifiable qui résulte de l'achèvement d'une partie du projet ou du projet complet.

➤ **Ressource**

Une ressource est un élément (humain, matériel, financier, logiciel, matière première ... etc) nécessaire à la réalisation d'un projet ou d'une partie d'un projet

➤ **WBS (Work Breakdown structure / structure de découpage du projet)**

C'est la structure hiérarchique des tâches d'un projet, la conception de la WBS passe par :

- L'établissement d'une liste de résultats des travaux les plus importants du projet (livrables)
- La division des livrables ou sous livrables si nécessaire.
- Pour chaque livrable, lister les activités qui sont nécessaires à sa réalisation.
- La possibilité de diviser ces activités ou sous activités.

**Exemple :**

Aménagement d'un bureau au niveau d'un sous-sol d'une maison.

**Solution :**

- **1ere étape**

- a- Plan d'aménagement (d'architecture)
- b- La listes des équipements achetés (acquis)
- c- Le permis de construction.
- d- La liste des équipements pour la réalisation des travaux.

- **2eme étape**

- a- On peut la découper en sous livrables
- b- On peut la découper en sous livrables
- c- On ne peut pas la découper en sous livrables.

d- On peut la découper en sous livrables.

- **3eme étape**

a- Trouver un architecte pour la réalisation.

b- Choisir un fournisseur - Procédure d'achat

c- Lancer la procédure administrative.

d- Faire un contrat avec une entreprise pour les travaux - Obtenir les équipements - Obtenir un prêt bancaire - Réaliser les travaux

- **4eme étape**

a- Faire une annonce – Comparer les offres techniques – Comparer les offres financières – Filtrer les CV.

b- Lancer les commandes – Vérifier la conformité des commandes – Payer les commandes.

c- Les papiers nécessaires (la paperasse) – autorisation de la mairie.

d- Choisir les meilleures offres – Appel d'offre.

➤ **OBS (Organizational Breakdown Structure / diagramme des responsabilités)**

Pour mener à bien le projet, il est important de s'assurer que chaque tâche est attribuée à la bonne personne, cela se traduit par la définition des rôles de chaque acteur de la façon la plus appropriée, en fonction de ses compétences et de ses responsabilités.

OBS est un schéma qui représente les responsabilités de chaque membre pour chaque tâche du projet, OBS peut être monté à partir d'une matrice RACI

La matrice RACI est une matrice des responsabilités qui indique les rôles des intervenants au sein d'un projet, cette matrice présente les activités en ligne et les rôles en colonnes et dans chaque cellule elle indique la responsabilité du rôle pour l'activité en utilisant les lettres R-A-C-I

- A : pour une tâche donnée il faut qu'une personne rende des comptes et assume la responsabilité en cas de problème (**A**ccountable). Il n'y a qu'un seul A dans une matrice RACI pour une tâche.
- C : certaines personnes peuvent être consultées, c'est-à-dire qu'elles ont le droit de donner un avis sur la tâche en question (**C**onsulted)
- I : d'autres personnes doivent être simplement informées c'est-à-dire qu'elles doivent savoir si quelque chose va se passer (**I**nformed).



- R : Quelqu'un doit réaliser la tâche en question et donc être responsable de son exécution (**R**esponsible). Il doit y avoir au moins un R par activité, il peut y avoir plusieurs R et peut arriver que le R et le A soient une même personne.

#### Remarque

Il est conseillé d'utiliser des libellés génériques de fonction pour qualifier les rôles plutôt que des noms de personnes

### Exemple

	Product Mannager	Project Mannager	Architecter	Devlopper	QA	Training
Market Analyses	R	I				
Sponsory Project	R	I				
General Request	A/R	C	I			
Detail Request	A	C	R	I		
Test Plan	A	C	C		R	
Build Program	A	I	C	R		I
Check Test	A	I	I	I	R	
Migrat Data	A	I	I	R		
Training User	A					R
Go No Go	R	A	I	I	C	I
Move to production	A	I	I	R	I	I

- **L'analyse de la matrice RACI**

Permet de vérifier si les responsabilités sont bien équilibrées tout au long du projet pour empêcher le sous ou sur engagement, l'analyse permet aussi de s'assurer que les tâches sont bien prises en charge tout au long du projet

- L'analyse Vertical (Rôle)

⇒ Trop de R : la ressource peut-elle réaliser autant de travail ?

- ⇒ Pas de case libre : la ressource a-t-elle vraiment besoin d'être impliquée dans autant de tâche ?
- ⇒ Pas de R ni de A : ce rôle peut-il être exclus du projet ?
- ⇒ Trop de A : Faut-il répartir le pouvoir de décision entre différents intervenants (rôle) ? « bottleWeek »

#### Remarque

Cette analyse montre les exigences imposées à un rôle, ça permet de voir si une personne convient à un rôle.

- L'analyse Horizontale :
  - ⇒ Pas de R : Qui doit faire le travail ?
  - ⇒ Trop de R : c'est le signe d'une tâche mal évaluée (trop complexe – trop grande)
  - ⇒ Pas de A : il faut designer un responsable.
  - ⇒ Trop de A : too many fingers in the cake (blague raconté au cours : 9alek wa7ed l'usine algerienne raho lihom la tele aya li yes9souh wach tekhdem i9olelhom ana chef d'équipe 7ta le79o lwa7ed miskin kan yekhdem 9aloulou wnta wach le poste ta3ek 9alelhom ana howa l'équipe hhhhh li mayed7akch -1). Ça crée une confusion sur celui qui donne le dernier mot
  - ⇒ Toutes les cases sont pleines : y a-t-il réellement un bénéfice à impliquer tous ces rôles.
  - ⇒ Trop de C : Risque de ralentir le travail
  - ⇒ Trop de I : Risque de ralentir le travail

Très peu de R sur toute la matrice : le travail risque d'être ralenti (toujours par rapport aux tâches mais on analyse toute la matrice et non pas par ligne)

#### ➤ L'ordonnancement

C'est l'élaboration d'un plan d'action permettant de déterminer le séquençage ou au contraire les parallélismes possibles entre les tâches d'un projet.

#### • Diagramme de Gantt

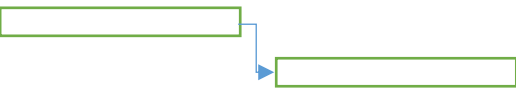

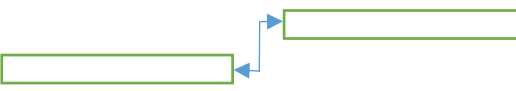

Ce diagramme est une représentation graphique permettant de situer les tâches du projet dans le temps.

En ligne, on liste les tâches et en colonne les jours, les semaines ou bien les mois ; les tâches sont représentées par des barres dont la longueur est proportionnelle

a la durée estimée, les tâches peuvent se succéder ou se réaliser en parallèle entièrement ou partiellement.

Pour chaque tâche, on peut renseigner les ressources humaines et matériels nécessaires pour son accomplissement.

Dans un diagramme de GANTT il y a 4 types de liaisons :

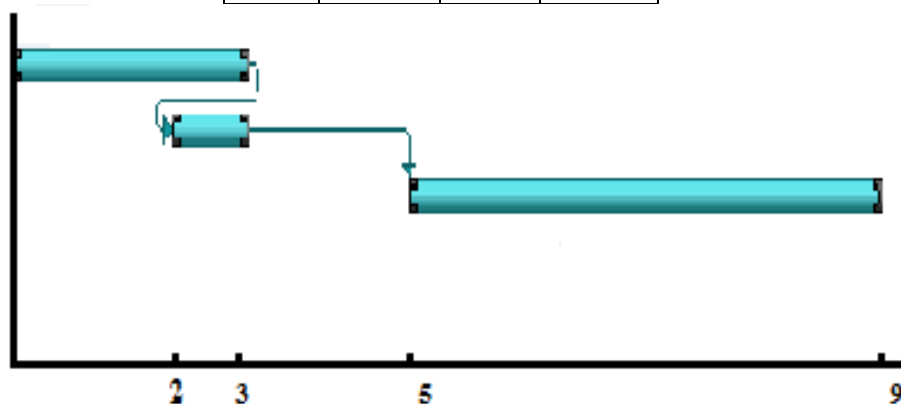
- FD : 
- FF : 
- DF : 
- DD : 

#### Remarque

Ces quatre relations peuvent être modifiées également par une variable de départ positive ou négative

Exemple :

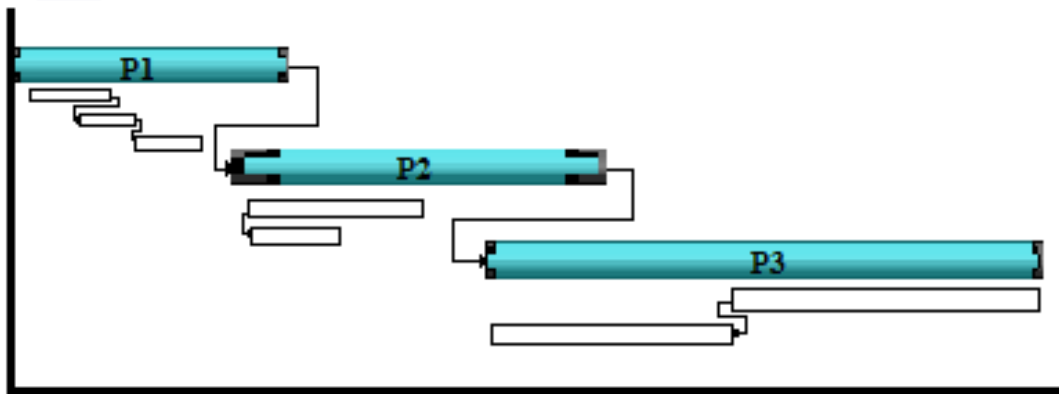
	Durée	Pred	Ref
T1	3jr	-	-
T2	1jr	T1	FD-1
T3	4jr	T2	FD+2



- Macro-Tache (Tache enveloppe-phases)

Pour clarifier un projet, on privilège le découpage du projet en autant de phases qu'il le faut

Les conditions d'antériorité s'appliquent alors a l'intérieur de ces phases comme étant des sous projets différents, se sont des phases qu'on synchronisera en suite

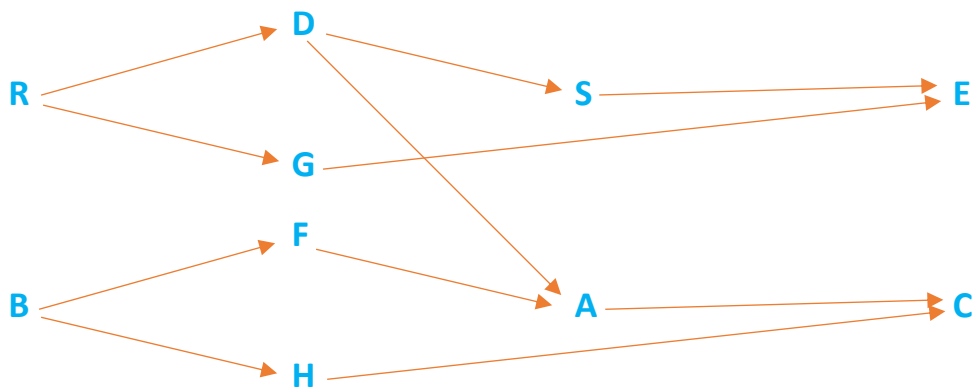


- **PERT (Program Evaluation and Review Technic)**

C'est une technique qui consiste à mettre les taches en ordre sous forme d'un réseau grâce à leurs dépendances

	Tache antérieur	Durée en journée
F	B	3
C	AHB	4
G	R	3
E	GSR	3
R	-	3
S	DG	2
D	R	2
H	B	4
A	DFR	5
B	-	4

Graphe de niveau :



- ⇒ Date de début au plutôt : c'est la date avant laquelle la réalisation de la tâche ne peut être démarrée (la tâche antérieure n'est pas achevée).
- ⇒ Date de début au plus tard : la date à laquelle on peut entamer la réalisation de la tâche sans courir le risque de retarder l'ensemble du projet.

#### Remarque

- 1- Si la tâche de début au plutôt coïncide avec la date de début au plus tard il y a aucune marge de manœuvre possible sur cette tâche, on dit alors que c'est une tâche critique.
- 2- La durée de la tâche est le temps qui s'écoule entre son début et sa fin.

### 1. Calcul des dates au plutôt :

Partant de la tâche de début, il s'agit de calculer de la gauche vers la droite (calcul aller) les dates à partir de zéro. Ce calcul donne un délai de réalisation de projet.

- 1.1. Date de début au plutôt : DTO = plus grande des dates fin au plutôt (FTO) des tâches qui la précède.
- 1.2. Date fin au plutôt : FTO = DTO + durée de la tâche

### 2. Calcul des dates au plus tard :

Il s'agit donc d'effectuer sur le réseau le calcul dit retour (de droite vers la gauche), on démarre de l'hypothèse que la date de fin du projet est égale à la plus grande date de fin au plutôt de ses prédécesseurs, cette date de fin de projet nous permet de lancer le calcul inverse ; nous déterminerons pour chaque tâche ses DTA et FTA sachant que :

- 2.1. Date Fin au plus tard FTA = la plus petite des DTA des taches suivantes
- 2.2. Date début au plus tard DTA = FTA – durée de la tache

## DEFINITIONS :

- **Chemin critique**

C'est la chaine de taches partant du début et aboutissant à la fin du projet tel que toutes les taches soient critiques, ce chemin correspond à la séquence de taches qui détermine la durée totale du projet, toutes modification sur la durée d'une tache critique est intégralement répercuté sur la durée du projet

- **Marge**

C'est la possibilité pour une tache d'être retarder sans impacter le projet, les taches critiques ont une **marge nulle**

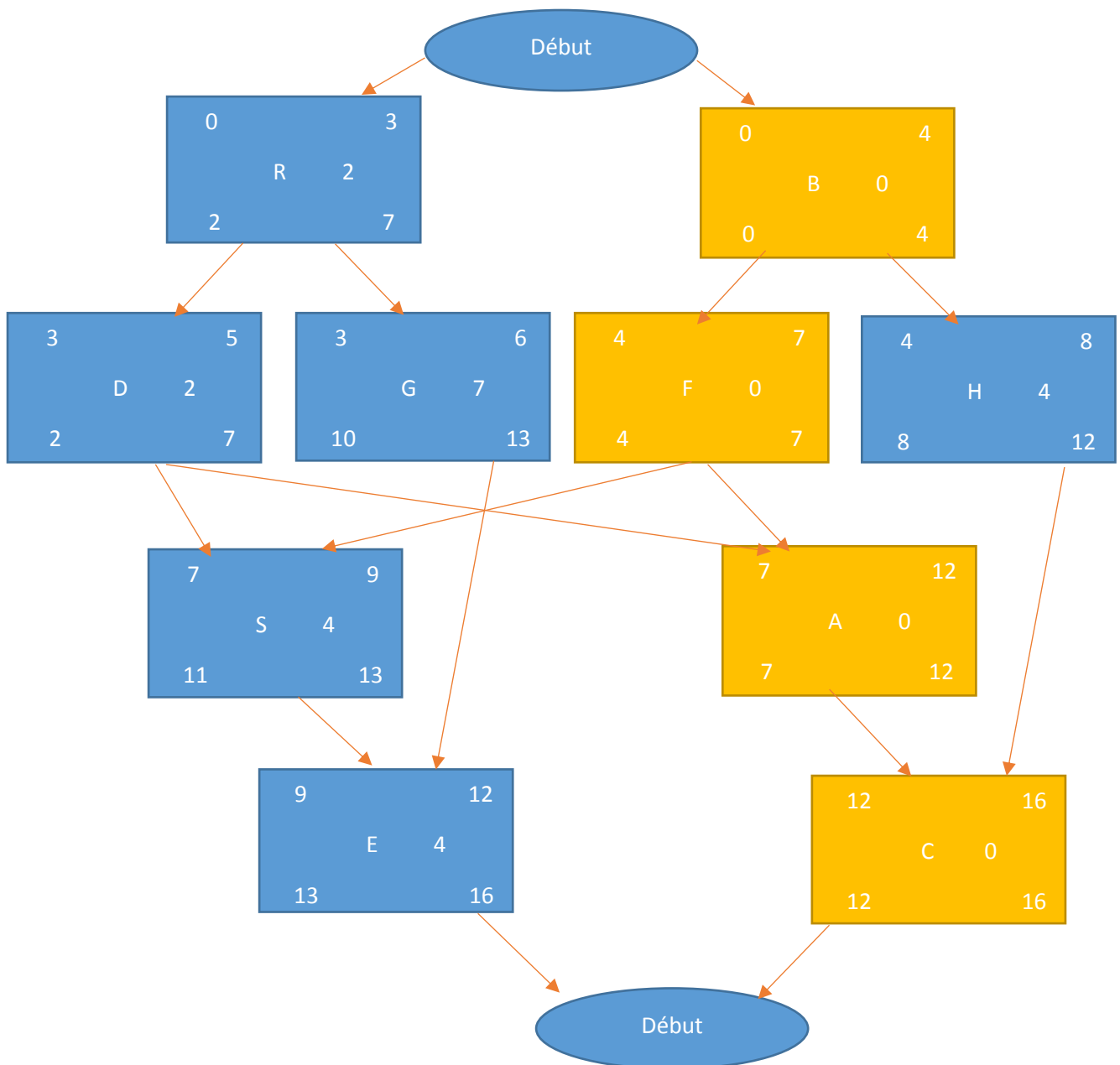
- **Marge libre**

C'est la différence entre la plus petite DTO des taches immédiatement suivantes et la FTO de la tache considérée. Elle correspond à la plage de temps dans laquelle on peut déplacer librement la tache sans modifier aucune DTO des taches immédiatement suivantes, elle est toujours inferieur ou égale à la marge totale.

- **Marge totale**

C'est la différence entre FTA et FTO (ou bien entre DTA et DTO) d'une même tache, elle est définie aussi comme la différence entre DTA de la tache suivante la plus contraignante et FTO de la tache considérée, c'est le temps maximum dans lequel on peut déplacer la tache sans modifier la date fin du projet.

Exemple précédent :



## **Chapitre 3 : L'estimation des coûts d'un projet**

### ⇒ **Techniques d'estimation**

1. Jugement d'expert
2. Estimation par analogie
3. Estimation du prix gagnant
4. Estimation descendante
5. Estimation ascendante
6. **Modélisation algorithmique des coûts**

### ⇒ **Définition de la modélisation algorithmique des coûts**

On développe un modèle basé sur un historique des coûts, ou on utilise une relation entre les coûts et une métrique du logiciel (taille).

#### **Remarque**

C'est l'approche la plus scientifique bien qu'elle ne soit pas la plus précise, pour un même projet différentes modélisations donnent différents résultats, c'est dû au fait que les métriques sont étroitement liées à l'organisme qui les utilise.

### ⇒ **Point Fonctions**

L'estimation de la taille du code (nombre de ligne) est difficile par nature, il y a plusieurs facteurs qui vont influencer cette taille (matériel, langage choisi, style d'écriture .... Etc.), une alternative consiste à ne pas utiliser la taille du code mais plutôt d'utiliser les points fonctions qui se rapporte aux fonctionnalités du logiciel. En analysant un historique de différents projets on déduit que le nombre de ligne de code par point de fonction pour un langage donnée qu'on va appeler LM, ainsi la taille du code sera égale à  $LM \times \text{nombre de points fonctions}$ .

L'avantage est qu'on peut estimer le nombre de points fonctions à partir de la spécification des besoins, ainsi on peut prédire la taille du code assez tôt dans le projet.

### ⇒ **COCOMO**

C'est une méthode de modélisation des coûts, ce modèle existe en 3 formes

- Forme simplifier (COCOMO de base)
- Forme intermédiaire



- Forme détaillée

## 1. COCOMO de base

Il utilise deux éléments :

- La taille estimée du logiciel
- Le type du logiciel qu'on développe
  - Type organique : des projets avec des équipes relativement petites travaillant dans un environnement familier sur des applications bien comprises. En conséquence, le surcout dû à la communication est faible. Les membres de l'équipe savent ce qu'ils font et ils s'acquittent rapidement de leurs tâches.
  - Type semi détaché : il s'agit d'un mode intermédiaire entre le mode organique et le mode intégré. Dans ce mode, l'équipe est composée à la fois de personnes expérimentés et débutants. Les membres ont une expérience limitée de ce genre de systèmes et ne sont pas familiers de certains aspects du système qu'ils développent.
  - Type intégré : les projets en mode intégré concernent le développement de logiciels étroitement liés à des systèmes matériels et logiciels complexe. Un tel logiciel doit fonctionner sous des contraintes particulièrement fortes, il est pratiquement impossible de modifier les besoins pour contourner un problème logiciel et les coûts de validations sont élevés. A cause de la complexité et de la nature très varié de ces projets. Il est rare que les membres de l'équipe aient une grande expérience de l'application qu'ils développent.

Exemple :

Soit un projet en mode organique dont la taille est égale à 32000ISL

$E=91\text{HM}$

$TDEV=14\text{mois}$

$N=7\text{personne}$

$P=352\text{ ISL/HM}$

## 2. COCOMO intermédiaire

Il y a plusieurs facteurs en plus de la taille et le type du projet qui vont influencer l'effort nécessaire pour mener à bien un projet, ces facteurs seront pris en compte dans le COCOMO intermédiaire

Le COCOMO intermédiaire prend comme point de départ :

- Les calculs du COCOMO simplifié
- Application d'une série de multiplicateurs pour prendre en compte des facteurs comme la fiabilité, la taille de la BD, les contraintes d'exécution et de stockage, les caractéristiques du personnel... Etc.

Généralement les caractéristiques sont divisées en 4 classes

- **Caractéristiques de produit**
  - Fiabilité :
    - Une fiabilité très faible signifie qu'une panne aura peu de conséquences.
    - Une fiabilité normale signifie qu'une panne provoque des pertes récupérables.
    - Une fiabilité très haute signifie que la conséquence d'une panne est vitale.
  - Taille de la BD :
    - Faible valeur correspond à une taille de la BD en Octet inférieur à 10 fois le nombre d'ISL.
    - Entre 10 et 100 fois la taille la valeur est dite normale
    - Supérieur à 100 fois la taille du système correspond une très grande valeur.
  - Complexité du produit :
    - Un code peu complexe utilise des E/S simples et des structures de données simples.
    - Un code de complexité normale effectue des E/S multifonction, et utilise des bibliothèques et des modèles qui communiquent entre eux
    - Un code très complexe peut être réentrant ou récursif, utilise des gestionnaires de fichiers complexes,

des traitement parallèles et des gestions de donnée complexe

- **Caractéristiques de l'ordinateur** : décrivent les contraintes liées à la plateforme matériel imposé au logiciel, ils influencent le projet car il faut adapter le logiciel aux mutations du matériel.
  - Contrainte du temps d'exécution :
    - Valeur normale lorsqu'on utilise 50% du temps d'exécution disponible.
    - Valeur élevée lorsqu'on utilise 95%
  - Contrainte du stockage (espace mémoire) :
    - Valeur normale = 50% de l'espace disponible
    - Valeur élevée = 95% de l'espace disponible
  - Temps de réponse (système de développement)
    - Très faible = développement interactif du système
    - Très élevé = lorsque le temps de réponse peut attendre des heures.
- **Caractéristiques du personnel**
  - Capacité d'analyse
  - Expérience dans le domaine de l'application.
  - Aptitude dans la programmation.
  - Expérience de la machine virtuelle.
  - Expérience dans le langage de programmation.

Très élevé = plus de 3ans d'expérience.

Très faibles = Peu ou pas d'expérience.

Normal = au moins une année d'expérience.

- **Caractéristiques du projet**
  - Utilisation d'outils logiciel.
  - Plan de développement (durée requise)
  - Utilisation de méthodes modernes

### 3. COCOMO complet

L'inconvénient des modèle simple et intermédiaire est qu'ils considèrent le produit logiciel comme un cout au quel ils appliquent des multiplicateurs, mais en réalité les gros logiciels sont constitués de sous système qui ne sont jamais homogènes (certains appartiennent au mode organique d'autre au mode intégré, certains doivent être très fiable d'autres pas du tout ...Etc.)

Le modèle COCOMO complet tient compte de ces diversités, il estime séparément le cout de chaque sous système puis il effectue la somme.

GOOD LUCK

Karima ♥