

Gestion de projets informatiques

Prof Badr Eddine El Mohajir
SMI S6

Faculté des Sciences de Tétouan
Département d'Informatique
UAE

Chapitre 1 Introduction et généralités

Un projet, c'est quoi?

**Un Projet consiste à vouloir réaliser
une IDEE ayant un caractère NOUVEAU**

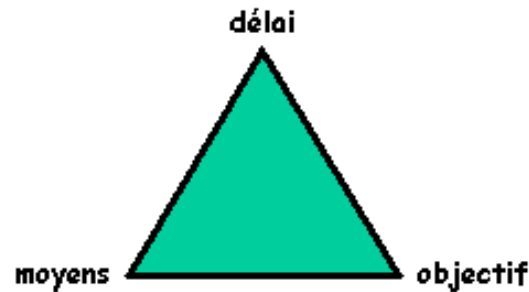
- ♦ Cette réalisation est **UNIQUE**
- ♦ Elle est **EPHEMERE**
- ♦ Il faut un **CERTAIN TEMPS** pour la réaliser

Composantes d'un projet

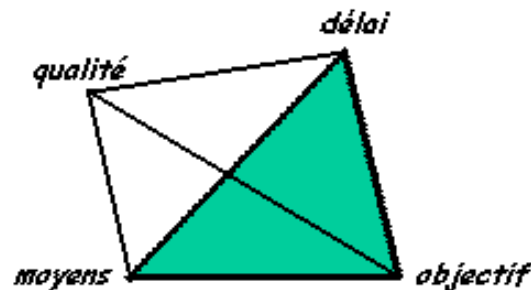
- ♦ Comment traduire une « **IDEE** » en « **PROJET** »:
 - ♦ Transformer l'IDÉE en OBJECTIFS:
 - TECHNIQUE Ce qu'on veut **FAIRE**
 - DE DELAI En combien de **TEMPS**?
 - DE COUT Avec quel **BUDGET**?
- ♦ Définir les MOYENS nécessaires
- ♦ Prévoir l'ORGANISATION et la GESTION du projet

Définition

- ♦ Notre propos est le développement de projet de **systèmes d'information**
- ♦ On définira un projet comme un triptyque **objectif** – **moyens** – **délai**. Il s'agit en effet d'atteindre un objectif défini avec des moyens adaptés et dans un délai donné

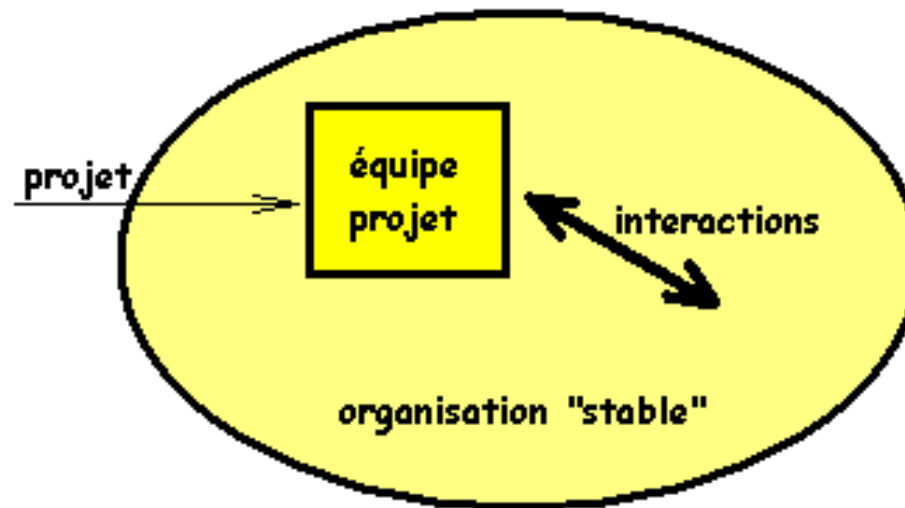


- ♦ On peut adjoindre un autre sommet correspondant au concept **qualité**



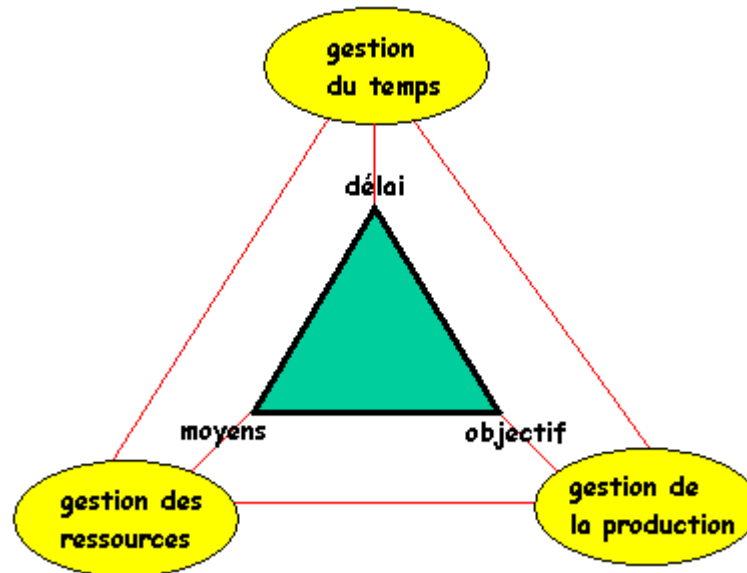
Types de gestion

- ♦ La **réalisation** d'un projet se fait par rapport à une **structure**; entreprise, administration ou autre. Ces organisations sont **stables** et possèdent leur propre mode et règles de fonctionnement. Un projet vient **mettre en épreuve l'organisation** et tester son efficacité



Types de gestion

- ♦ Aux trois ingrédients d'un projet correspond **trois types de gestion**:
 - Gestion de la **productivité**: organisation, direction de travaux et contrôle (suivi)
 - Gestion des **ressources humains et matériels**: choix de l'équipe, affectation de personnel, coordination, sélection matériel, gestion du budget et contrôle des coûts
 - Gestion du **temps** pour la maîtrise des délais
- ♦ Ces trois types de gestion sont interdépendants



Phases d'un projet

- ♦ Le développement d'un projet peut se résumer en deux grandes phases:

- **Phase de réflexion et de prévision:**

Activités d'analyse et d'organisation consistant au découpage du projet en unités (taches) et repérage des contraintes d'enchaînement de ces taches, puis ordonnancement (planification de l'exécution des taches (calendrier et affectation de ressources))

- **Phase de pilotage** pour le suivi de l'avancement, prise de décisions et de la production

Maîtrise d'ouvrage et Maîtrise d'oeuvre

- ♦ Un projet vise à satisfaire un ensemble de **besoins** exprimés par des futurs utilisateurs. Mais il est conduit à terme par des **spécialistes** en développement de SI



Maîtrise d'ouvrage

♦ Maître d'ouvrage:

- est **porteur des besoins**, commanditaire d'un résultat correspondant à des objectifs, à son calendrier, à son budget
- **n'a à priori ni les compétences, ni les moyens techniques** pour aboutir aux résultats. Il fait appel à un maître d'œuvre et **se fait représenter par un responsable du projet**, ou par une direction de projets (équipe de responsables) si le projet est important
- **peut faire appel à un prestataire** (maître d'ouvrage délégué) pour l'aider à définir ses besoins

Maîtrise d'oeuvre

♦ Maître d'oeuvre:

- **stipendié par le maître d'ouvrage** pour réaliser le produit convoité selon des conditions (délais, qualité, coûts, ...) définies par un contrat
- **choisi les moyens de travail** techniques et humaines. Il désigne aussi le chef de projet pour conduire le projet à terme
- **peut avoir recours à des prestataires** spécialisés (sous-traitance) quand il ne possède pas des compétences quelquefois pointues

Spécificité d'un projet de SI

- ♦ **Un projet de SI** possède au moins deux caractéristiques différentes par rapport aux projets industriels ou de travaux publics:
 - **l'objectif n'est pas toujours complètement défini au début** du projet. La part d'**aléas** est grande. Les moyens ne sont pas prédéfinies de manière exacte. On est souvent conduit à des **réajustements**, le plus souvent de l'objectif compte tenu des moyens et des délais constatés. Tout ceci est nettement contestable dans un projet de construction de route ou d'un véhicule automobile
 - **le projet de SI se déroule dans une organisation dont la « stabilité » peut être mis en cause par le projet**. L'interaction projet organisation est un facteur non négligeable qu'il n'est pas possible d'omettre (un SI peut changer la structure d'une organisation!)

Découpage d'un projet

- ♦ La résolution d'un problème complexe passe par son **découpage** en plusieurs sous problèmes dont la résolution sera vraisemblablement plus aisé. De même, un projet est généralement découpé en “morceaux”
- ♦ Un projet peut être défini par un ensemble d'**unités autonomes appelées parties**. Une partie se définit par les caractéristiques suivantes:
 - chaque partie conduit à un **résultat défini**
 - chaque partie nécessite une quantité de **ressources définie**
 - les contraintes d'**enchaînement entre parties sont identifiées** (certaines parties peuvent s'élaborer séquentiellement, d'autres parallèlement)
 - une partie peut éventuellement être **découpée elle-même en sous parties**

Découpage temporel et structurel

- ♦ Un projet peut être découpé de manière temporelle (succession d'étapes et de phases) ou structurelle (modularisation)

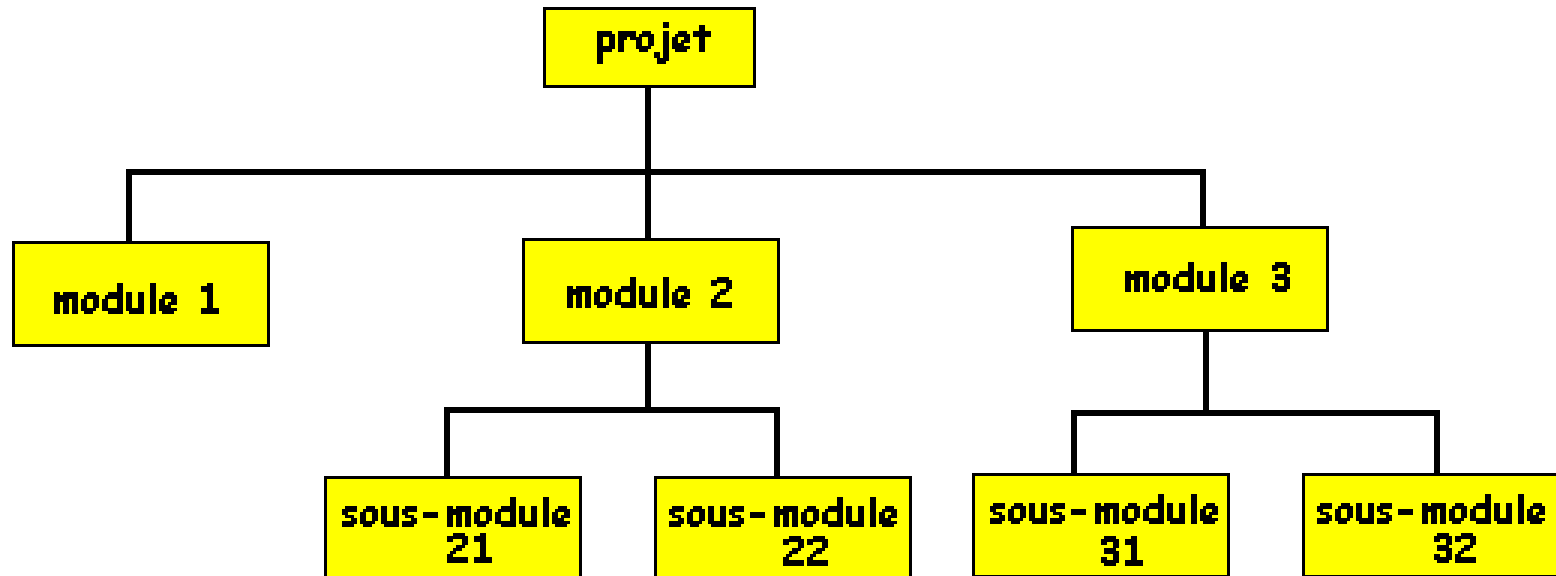
- **Découpage temporel:** un projet est découpé en **étapes**; une étape est découpée en **phases**; une phase est découpée en **tâches**.

Chaque étape, phase ou tâche comporte une date de début et une date de fin et produit un résultat défini

- **Découpage structurel:** un projet est découpé en **modules**, un module peut être, à son tour, découpé en sous modules

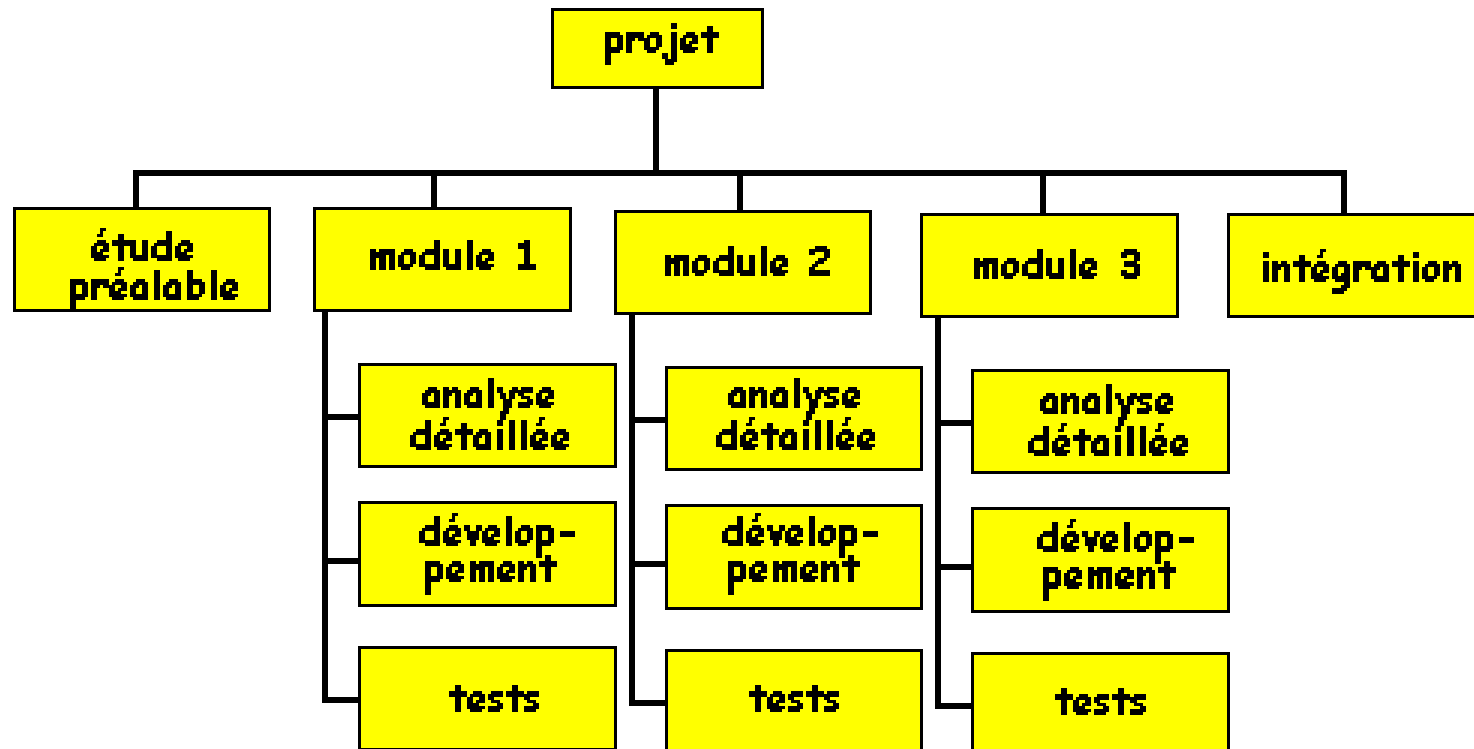
Normalisation de découpage: *PBS*

- ♦ Le découpage peut être purement structurel: *Product Breakdown Structure (PBS)*



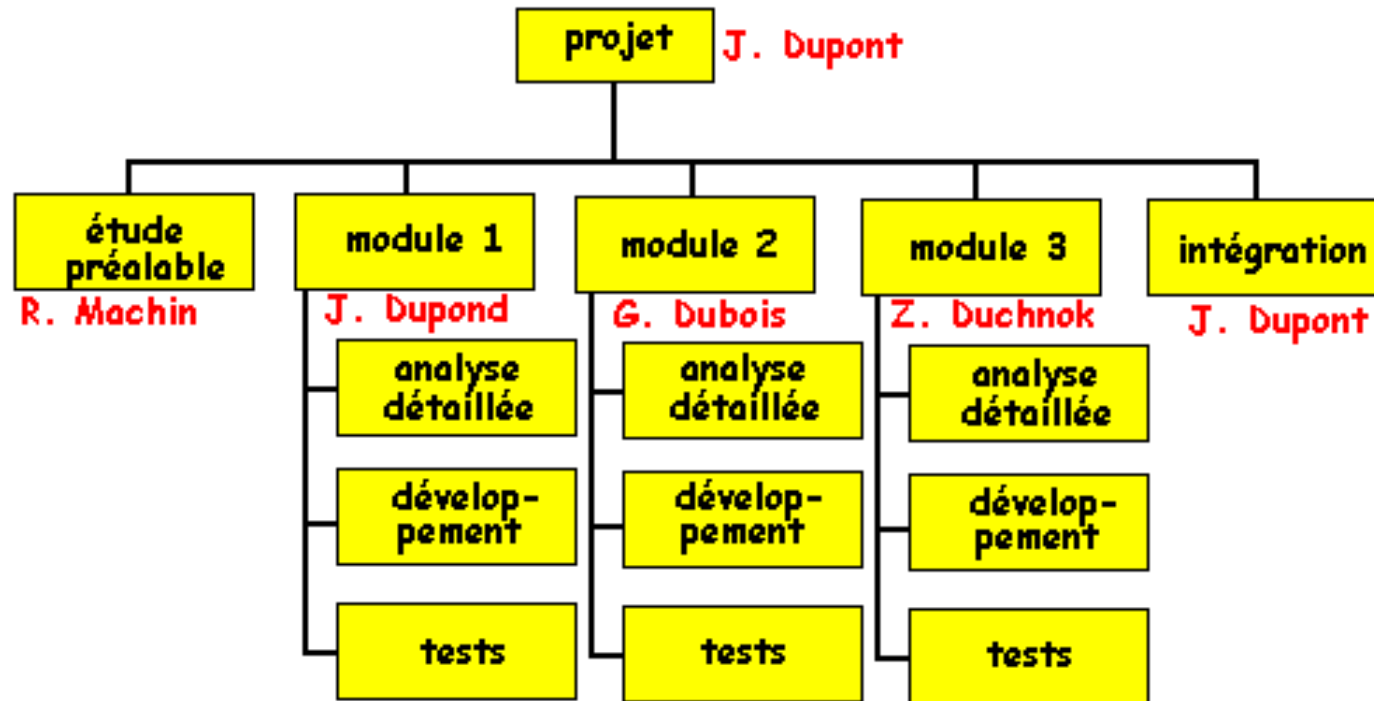
Normalisation de découpage: *WBS*

- ♦ Le découpage peut être à la fois structurel et temporel: *Work Breakdown Structure (WBS)*



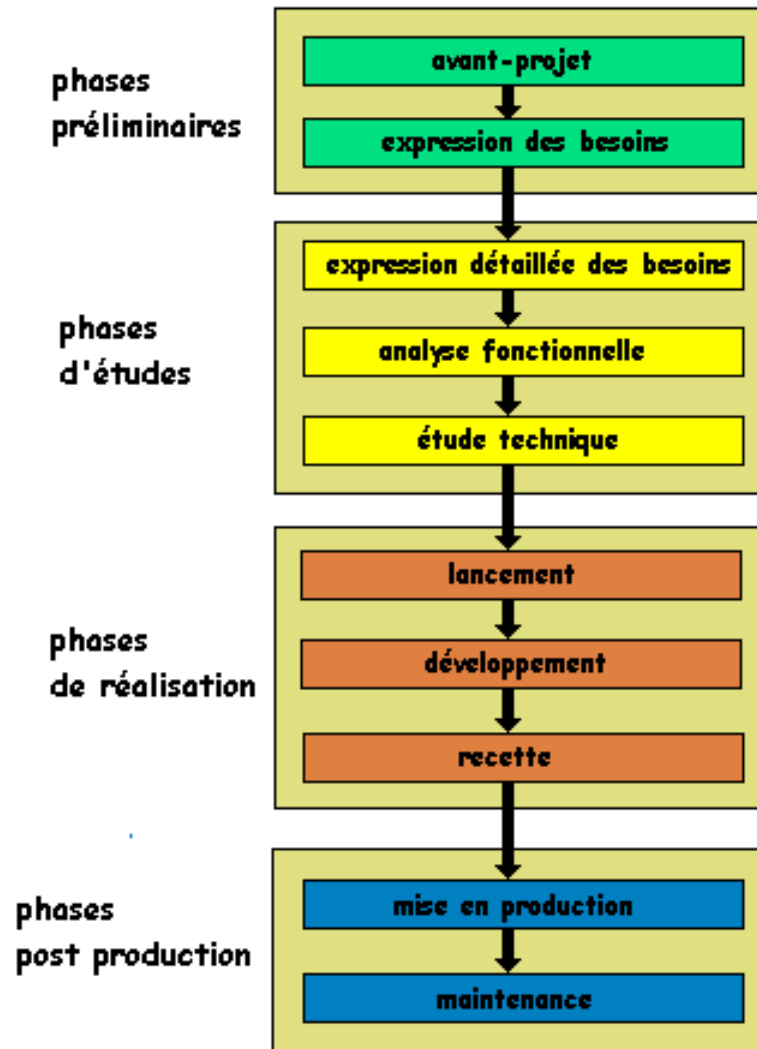
Normalisation de découpage: *OBS*

- ♦ On parle de découpage organisationnel quand on indique les responsables d'unités:
Organization Breakdown Structure (OBS)



Étapes dans le cadre d'un découpage temporel

- ♦ Les étapes usuelles que l'on retrouve (pas toujours sous le même nom) dans différentes “méthodes” de développement sont:



Phases préliminaires

♦ Avant projet:

- **Maître d'ouvrage** décrit les **grandes lignes** du projet
- S'assurer de la **pertinence** du projet, qu'il répond à des besoins réels, **rentabilité**
- Communiquer sur le projet auprès des **utilisateurs concernés**
- Susciter des réactions permettant de **conforter le projet**

♦ Expression des besoins:

- **Maître d'ouvrage** exprime les besoins en terme de **fonctionnalités**
- **Validation des spécifications** par la direction du projet et les utilisateurs concernés du produit envisagé

Phases d'études

♦ Expression détaillée des besoins:

- **Maître d'ouvrage** dresse une **liste détaillé et exhaustive des fonctionnalités** requises
- **Associe le maître d'oeuvre** pour compléter utilement un certain nombre de “blancs” ou de “silences”

♦ Analyse fonctionnelle:

- Conception sur le plan fonctionnelle du produit en se basant sur l'expression détaillée des besoins, déduire **l'architecture du produit à réaliser**
- Élaboration d'un **cahier de charge fonctionnel** qui doit être validé et approuvé par le **maître d'oeuvre**

♦ Étude technique:

- **Investir les moyes** et les ressources nécessaires au développement du produit
- **Développer une maquette** (prototype) pour vérifier que le produit à développer convient bien au commanditaire
- Élaboration d'un **cahier des clauses techniques particulières** que le **maîtrise d'ouvrage** doit valider

Phases de réalisation

♦ Lancement du projet (*kick-off*):

- Élaboration et présentation d'un **plan précis pour la concrétisation du projet**
- **Accord** entre maîtrise d'ouvrage et maîtrise d'œuvre sur le plan
- **Communication** sur le projet auprès des participants et des utilisateurs futurs

♦ Développement:

- **Phase dense pour la maîtrise d'oeuvre** pour suivre strictement le plan, en parant aux aléas inévitables (maladie, congés des développeurs, pannes de ressources ...etc.). Ces aléas sont tenu en compte lors de l'analyse du risque

♦ Recette:

- La recette se prépare dès le début du projet et un référentiel "recette" peut faire l'objet d'un accord entre la maîtrise d'oeuvre et la maîtrise d'ouvrage
- Livraison du produit par la maîtrise d'œuvre à la maîtrise d'ouvrage
- Réalisation des tests de conformité, de fonctionnement et de qualité

Phases post production

♦ Mise en oeuvre:

- **Mise en production**: installation chez le commanditaire, configuration ...etc
- **Formation** des utilisateurs finaux

♦ Maintenance:

- Mesures à prendre pour que le système continue à fonctionner normalement et puisse **évoluer** et **répondre** à de nouveaux besoins
- Cette phase fait l'objet d'un **contrat de prestation** complémentaire avec la maîtrise d'oeuvre

Au cours du projet

- ♦ La maîtrise d'oeuvre doit **informer de manière permanente** la maîtrise d'ouvrage de l'état d'avancement des travaux
- ♦ Assurer la **mise en oeuvre et la conservation de toute la documentation du projet**: documents fonctionnels et documents techniques afin de capitaliser les connaissances et aussi en cas d'intervention ultérieure

Chapitre 2

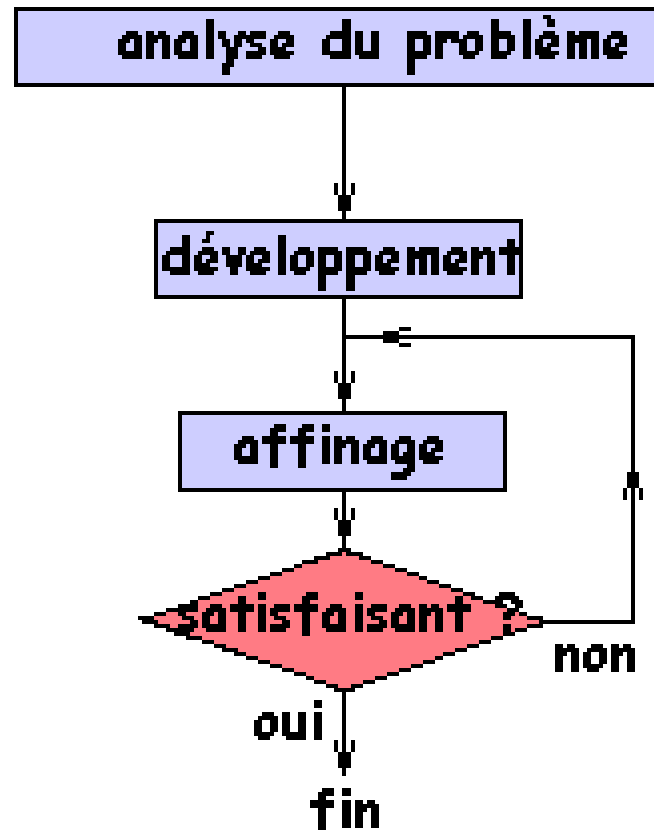
Cycles de développement et estimation des charges

Méthodes de développement

- ♦ **Méthodes** les plus utilisés dans les cycles de développements sont:
 - **Méthodes systémiques** dont **Merise** est le représentant le plus connu dans le monde francophone
 - **Méthodes orientées objet** qui sont plus des formalismes de description que des méthodes. **UML** est actuellement le représentant le plus connu

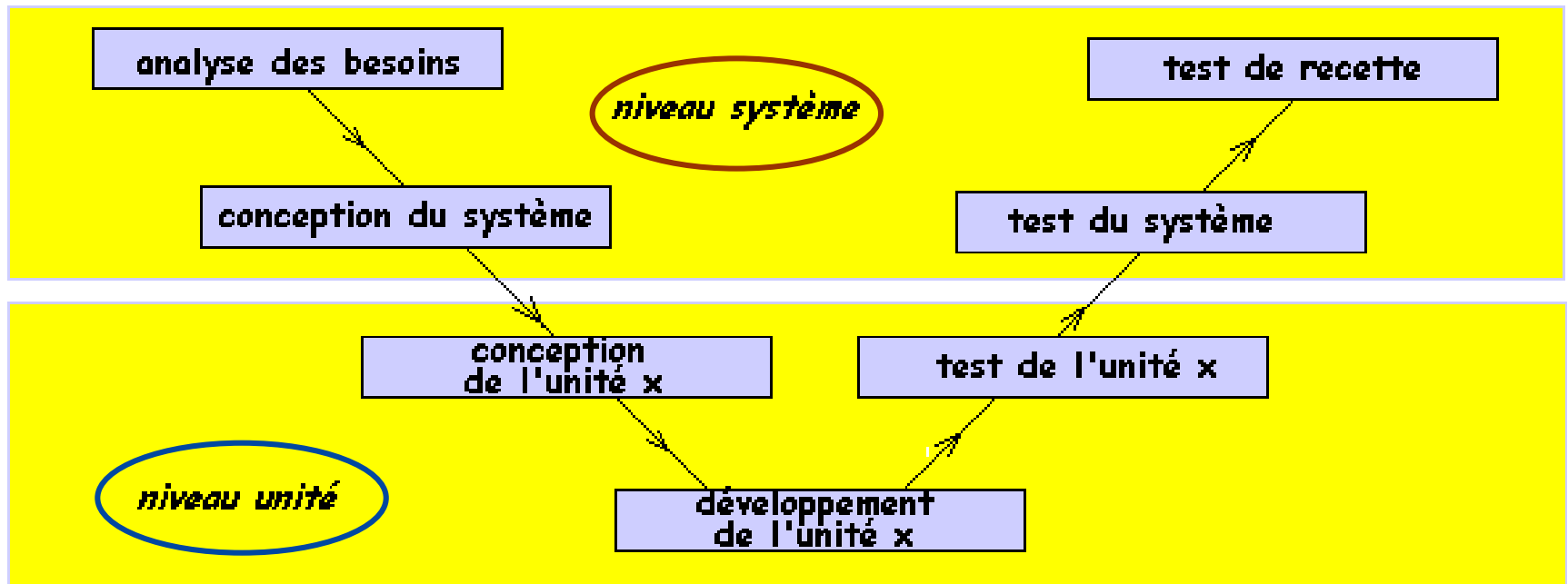
Cycles de développement

- ♦ **Cycle code-and-fix:** modèle du programmeur classique et "pressé".
La phase finale, en fait, peut être longue.



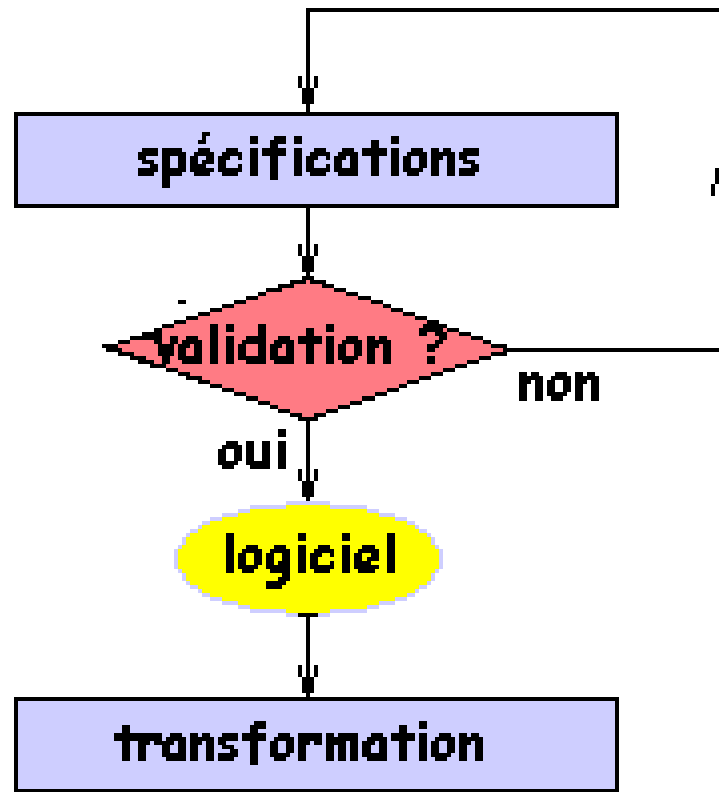
Cycles de développement

- ♦ **Cycle en V:** mélange de découpage structurel et temporel



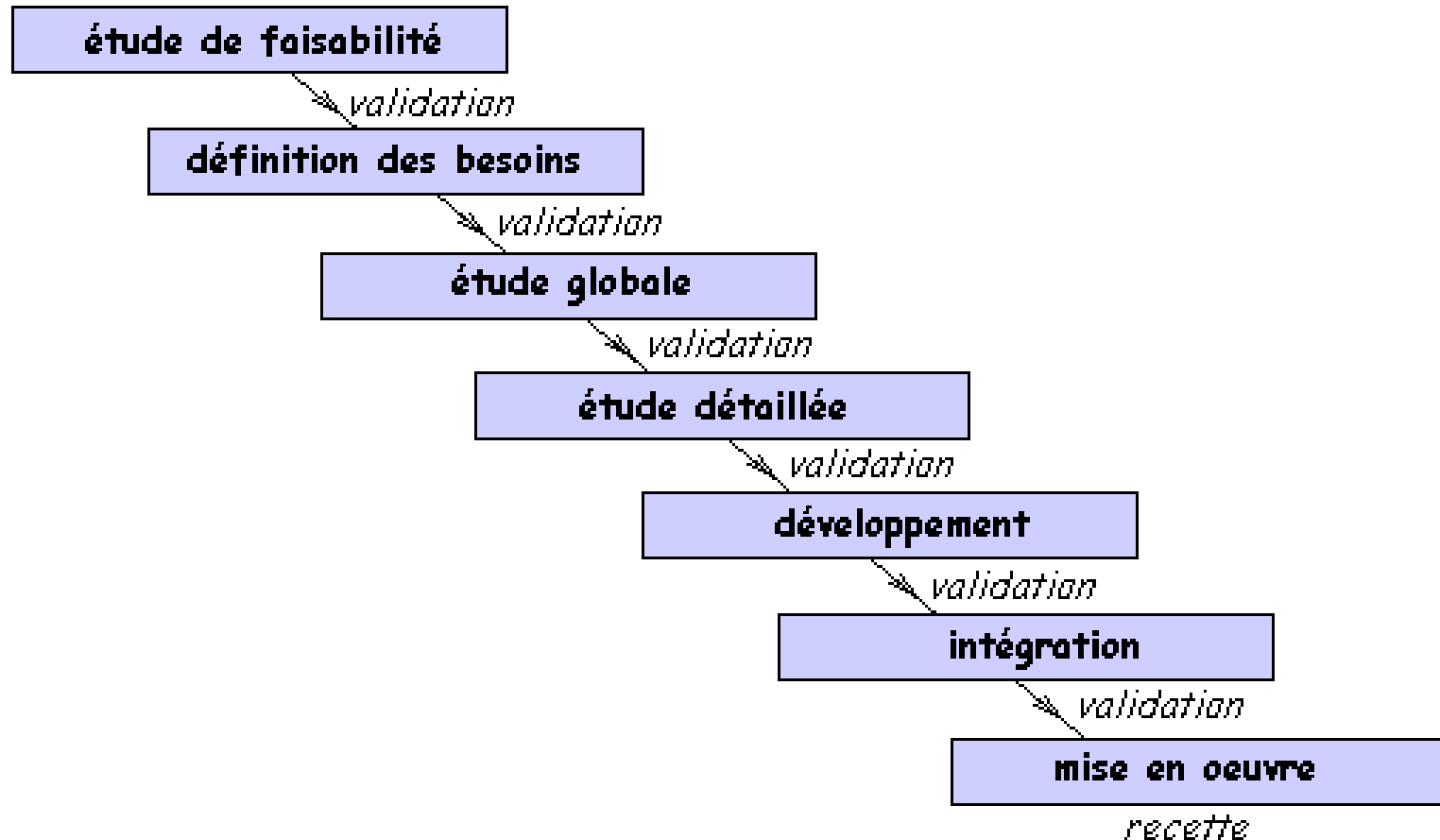
Cycles de développement

- ♦ **Cycle de la transformation:** utilisé dans quelques processus industriels bien maîtrisés; un logiciel se charge de la "programmation"



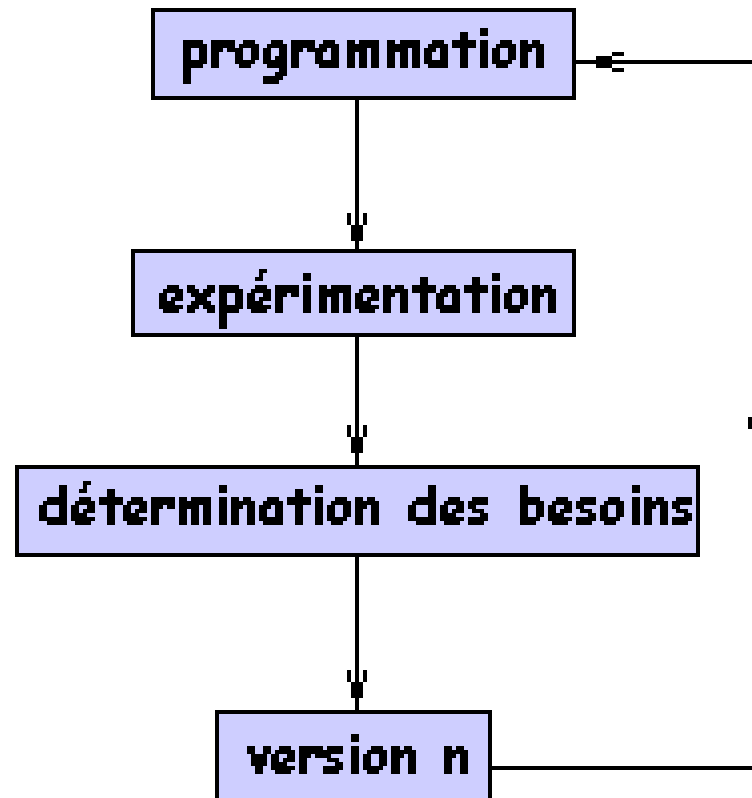
Cycles de développement

- ♦ **Cycle en cascade:** succession d'étapes; la passage d'une étape à la suivante nécessite une validation



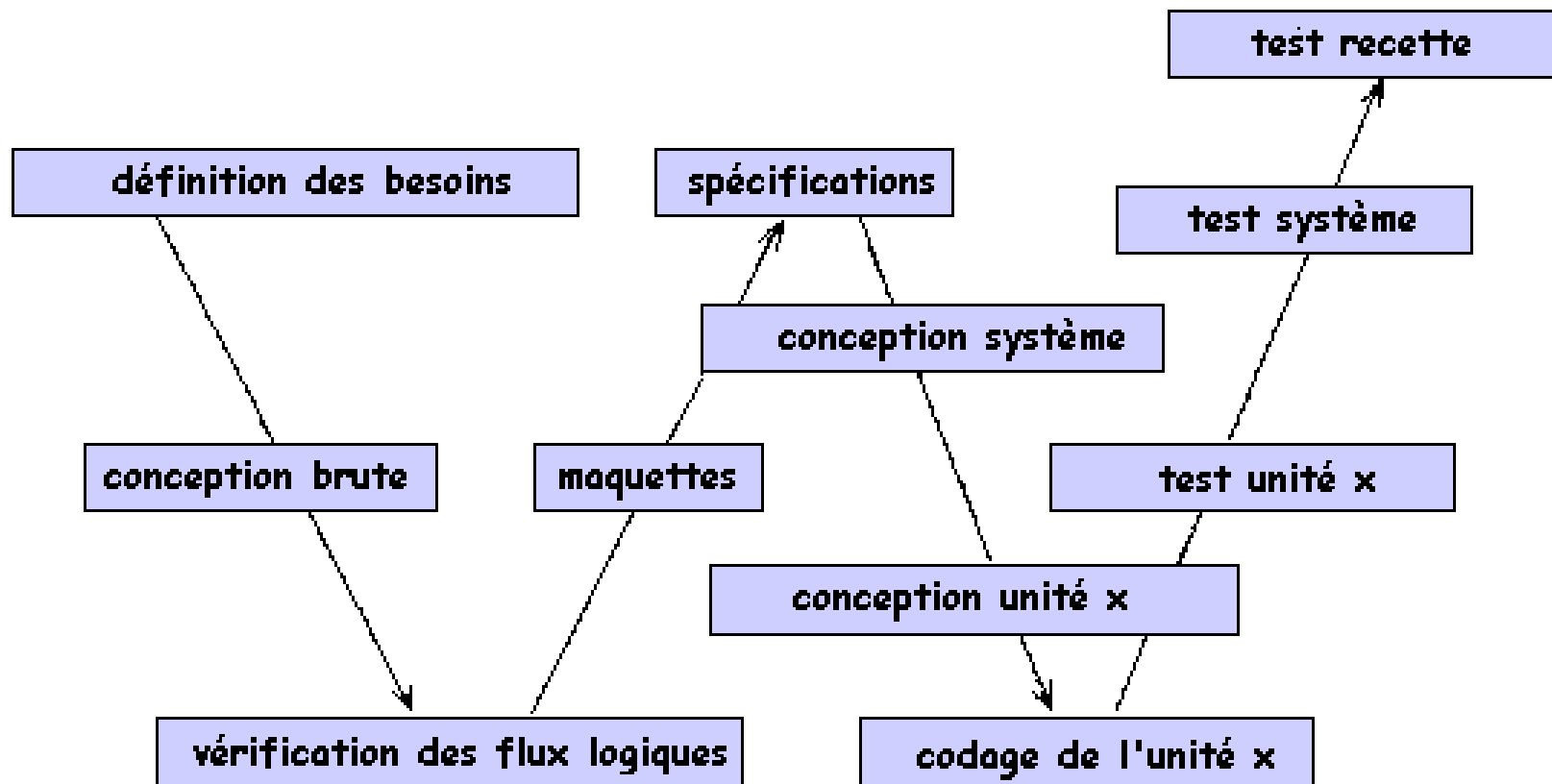
Cycles de développement

- ♦ **Cycle de développement évolutif:** non conseillé; c'est la méthode du bidouilleur: on programme et on regarde si ça marche !



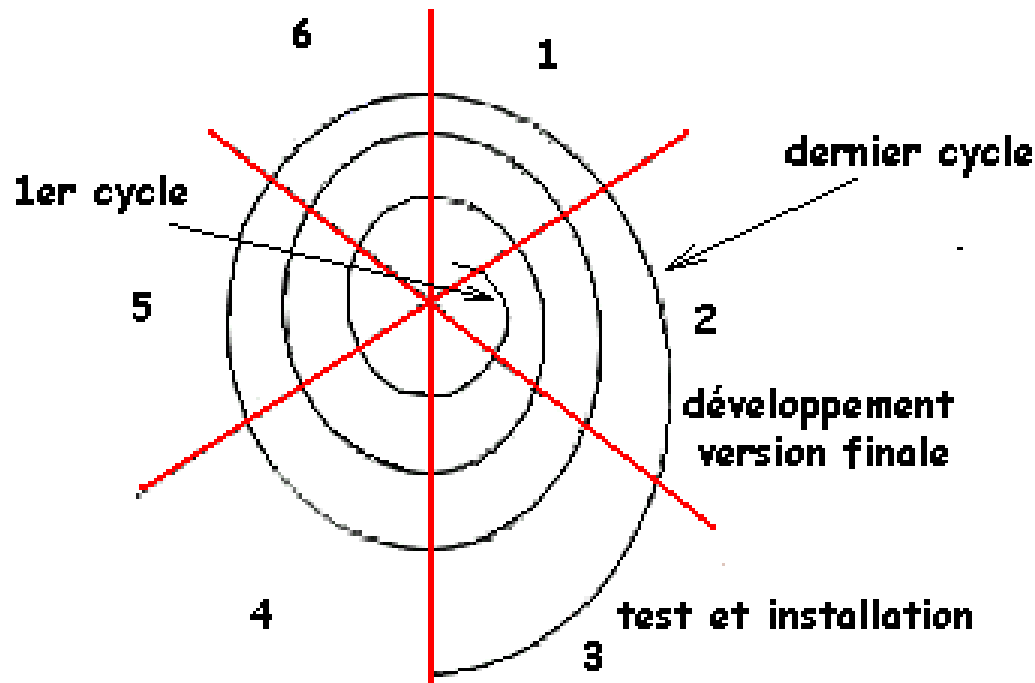
Cycles de développement

- ♦ **Cycle en W:** c'est le modèle en V auquel on rajoute une phase de maquettage; utilisation dans les développements délicats (en multimédia par exemple)



Cycles de développement

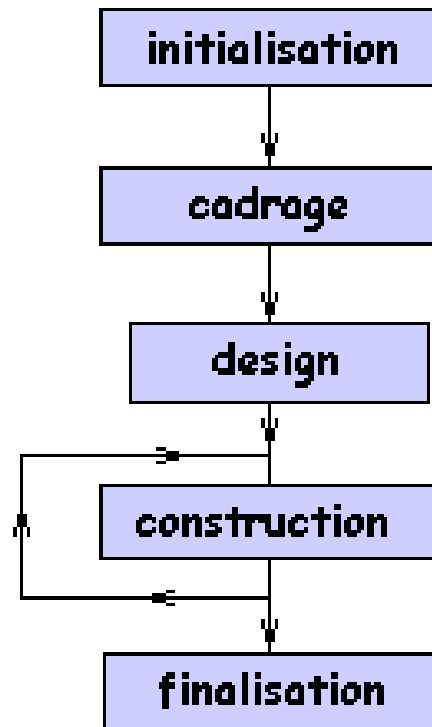
- ♦ **Cycle en spirale:** succession de cycles comprenant 6 phases : analyse du risque (1), développement d'un prototype (2), tests du prototype (3), détermination des besoins (4), validation des besoins (5), planification du prochain cycle (6). Utilisé dans des cas de faible visibilité



Cycles de développement

♦ **Cycle RAD:** RAD : Rapid Application Development



La phase de construction permet d'obtenir des prototypes successifs



Cycles de développement

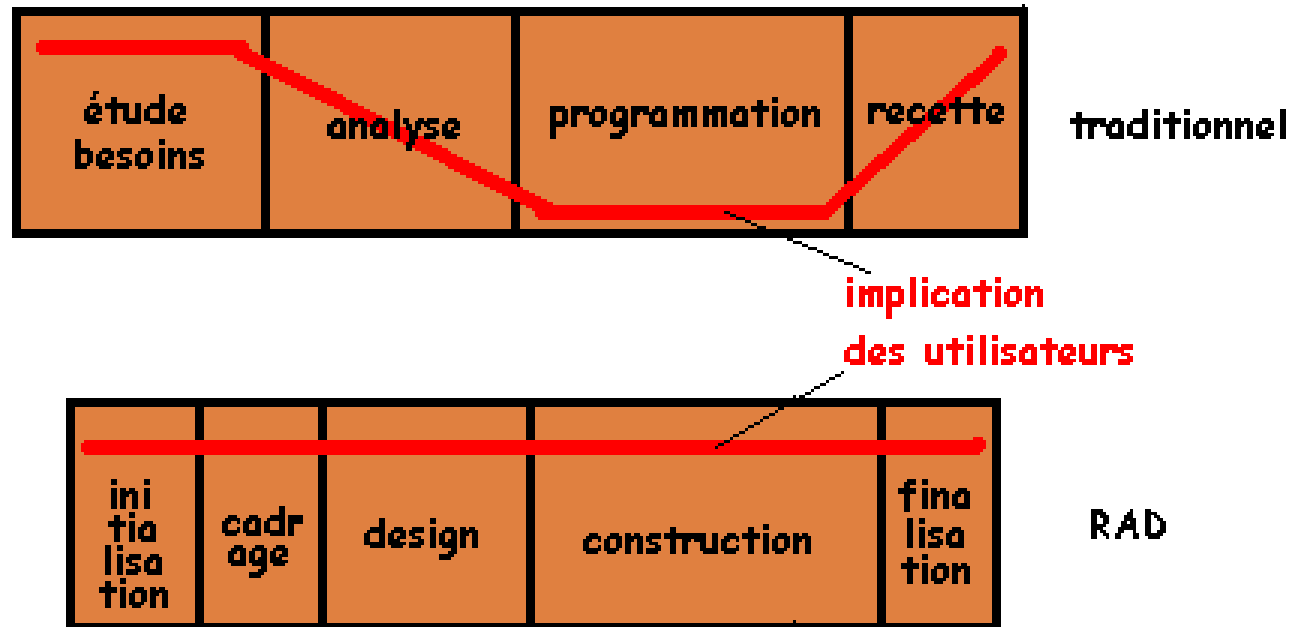
- ♦ **RAD** : Les objectifs du cycle de développement sont les suivants :
 - réduire la charge et le délai, et donc, par voie de conséquence, le coût du projet
 - garantir l'adéquation entre le produit réalisé et les besoins du commanditaire
 - impliquer largement la maîtrise d'ouvrage pendant le développement

5 phases du cycle RAD

Phases	Nature	% de charge du projet	Modèle
initialisation	<ul style="list-style-type: none"> définition du périmètre du projet structuration du projet en thèmes sélection des acteurs 	6%	 <p>cycle en cascade</p>
cadrage	<ul style="list-style-type: none"> expression des besoins par les utilisateurs 	9%	
design	<ul style="list-style-type: none"> conception/modélisation organisationnelle validation par les utilisateurs 	23%	
construction	<ul style="list-style-type: none"> construction itérative de processus validation des prototypes par les utilisateurs 	50%	 <p>cycle en spirale</p>
finalisation	<ul style="list-style-type: none"> recette globale installation du système 	12%	

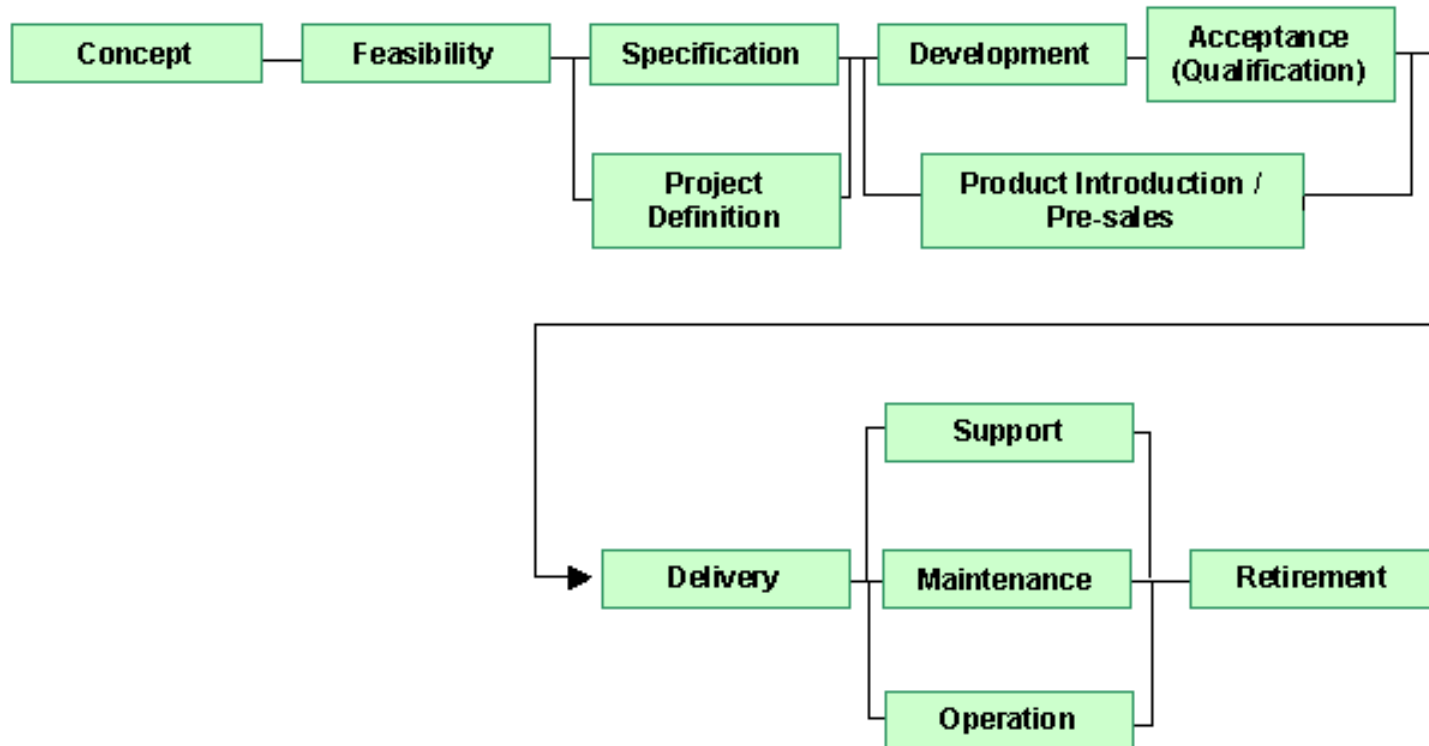
Cycle de développement RAD

- ♦ Les utilisateurs (ici véritables représentants de la maîtrise d'ouvrage !) sont relativement présents dans les différentes phases du développement. Ceci est une grande différence avec d'autres cycles de développement :



Exemple de cycle de développement

♦ Société de Télécommunication: **ENCOMPASS**



Estimation des charges de développement d'un SI

- ♦ L'estimation du **coût** et de la **durée** permet une visibilité croissante du projet même au niveau de la tâche. Cette estimation se fait en trois étapes :
 - lors d'une **réponse à un appel d'offres** où il s'agit de fournir au plus vite une réponse adaptée au marché
 - lors de la **planification du projet** où il s'agit d'établir le plan projet et le plan qualité qui serviront de cadre contractuel au projet,
 - lors du **déroulement du projet** afin d'affiner les prévisions et de les mettre à jour.
- ♦ De manière générale, le **processus d'estimation** passe par 3 étapes:
 - estimer la **taille du produit**
 - estimer l'**effort** (en hommexmois)
 - estimer la **durée** (en mois ou semaines calendaires)

Définition des charges de développement d'un SI

- ♦ La **charge** ou effort est la quantité de travail nécessaire mesurée en **joursxhommes** (ou **moisxhommes** ou **annéesxhommes**). Un mois correspond à 20 jours si les week-ends ne sont pas des périodes de travail.
- ♦ Connaissant la charge et le coût unitaire du moisxhomme, on peut avoir une estimation du **coût** en **ressources humaines** d'un **projet informatique** (coût d'un programmeur est estimé entre 200 et 400 DH par jour).

Poids du projet	Charge	Coût prévisionnel total
Petit projet	< 12 mh	< 96.000 DH
Projet moyen	entre 12 et 36 mh	entre 96.000 DH et 288.000 DH
Grand projet	> 36 mh	> 288.000 DH

Besoins en estimation au niveau global du projet

♦ Au niveau de **l'étape**

- Ordre de grandeur : moisxhommes
- Ajuster le découpage
- Sous-traiter
- Prévoir des délais pour planifier l'ordonnancement des étapes

♦ Au niveau de **la phase**

- Faire une planification précise
- Annoncer un calendrier de remise des différents résultats intermédiaires
- Prévoir et effectuer un suivi, pour surveiller les écarts
- Prévoir l'affectation des ressources

♦ Au niveau de **la tâche**

- Affectation des ressources individuelles
- Planification au niveau le plus fin

Méthodes de calcul de la charge d'un projet

♦ Il existe un certain nombre de méthodes pour calculer la charge d'un projet.
Elles sont cinq:

- méthode **Delphi**
- méthode de la **répartition proportionnelle**
- méthode **d'évaluation analytique**
- méthodes **Cocomo**
- méthode **des points fonctionnels**

Méthode Delphi

- ♦ Élaborée en 1948 par la Rand Corporation. Elle est fondée sur le jugement des experts qui, selon leur propre expérience, proposent confidentiellement une charge indicative. On procède alors en trois étapes:

1. **Formulation du problème**: définition précise du domaine d'investigation
2. **Choix des experts**: il est choisi pour sa capacité à envisager l'avenir, il doit être indépendant et son avis doit être récolté de manière anonyme: pas de propositions en groupe (pas de leader!)
3. **Déroulement et exploitation des résultats**: Des questionnaires successifs sont envoyés afin de réduire la dispersion des opinions. L'expert doit évaluer son propre niveau de compétence et peut à chaque fois réviser son jugement si son estimation est fortement déviante de celle du groupe, à moins s'il peut donner une justification. La réponse définitive est donnée au quatrième tour. La moyenne des estimations données donne une idée de la charge globale du projet.

Méthode de répartition proportionnelle

- ♦ Adaptée aux projets découpés en étapes, phases et tâches classiques: les étapes sont l'étude préalable, l'étude détaillée, l'étude technique, la réalisation, et la mise en oeuvre

étapes	ratios
étude préalable	10% du projet
étude détaillée	20-30% du projet
étude technique	5-15% de la charge de réalisation
réalisation	2 fois la charge d'étude détaillée
mise en oeuvre	30-40% de la charge de réalisation

Hors mis en oeuvre

Fonction de la nouveauté

Proportionnelle à la complexité
des programmes et nbr sites

- ♦ Ces ratios sont issus de l'**expérience**. Ce sont des **recommandations**
- ♦ L'évaluation de la charge de l'étape de réalisation peut se faire au moyen d'une autre méthode
- ♦ En cours de déroulement du projet, le temps consommé sur les étapes en amont redéfinit celui des étapes à venir: **estimation dynamique**

Méthode de répartition proportionnelle

- ♦ Seule l'étude préalable fait l'objet d'une quantification analytique claire: elle est divisée en 3 phases : observation, conception, appréciation.

étude préalable	
phases	ratios
observation	30-40%
conception	50-60%
appréciation	10%

- ♦ La méthode s'applique aussi à l'estimation des charges complémentaires annexes

charges complémentaires	
tâches	ratios
encadrement du projet (étape de réalisation)	20% de la charge de réalisation
encadrement du projet (autres étapes)	10% de la charge de l'étape
recette	20% de la charge de réalisation
documentation utilisateur	5% de la charge de réalisation

Méthode d'évaluation analytique

- ♦ Adaptée particulièrement aux développements informatiques couramment appelés "programmes". Elle s'appuie sur la typologie des programmes à développer
- ♦ Affecte un poids par type de programme et niveau de difficulté du projet (**unité: joursxhommes**). La charge obtenue est celle de **réalisation**; égale à $\sum (p_i * t_i)$; où **p** est le poids et **t** nombre de programmes du type **i**

type de programme		difficulté		
		facile	moyen	difficile
programmes transactionnels	menu	0,25	0,5	1
	consultation	1,5	2,5	5
	mise à jour	2	3	5
	édition temps réel	1	2	3
programmes batch	extraction	0,5	1	2
	mise à jour	2	3	5
	édition temps différé	1,5	3	5

- ♦ Il est convenu de rajouter aux charges ci-dessus des charges complémentaires :
 - Tests d'enchaînement : **10%** de la charge de réalisation
 - Encadrement : **20%** de la charge de réalisation

*CO*nstructive *CO*st *MO*del (Cocomo)

- ♦ Le modèle Cocomo (Boehm, 1981) repose sur l'hypothèse suivante :
 - Un programmeur **évalue mieux la taille** du logiciel à développer que la quantité de travail nécessaire
- ♦ COCOMO est divisé en trois modèles, qui affinent l'estimation en prenant en compte de plus en plus de paramètres:
 - **modèle de base**: prend en compte le nombre d'instructions que l'application doit contenir et la complexité de cette dernière
 - **modèle intermédiaire**: applique au modèle de base des coefficients prenant en compte les facteurs du coût (compétence de l'équipe, complexité de l'environnement ...etc.)
 - **modèle avancé**: adapté aux gros projets, reprend le modèle intermédiaire et affine les facteurs du coût en fonction de chaque phase du cycle du développement

Complexité des applications

- ♦ **Organique** (simple): **petit** projet (*Lignes Of Code* (LOC) < 50 000) menés avec de petites équipes, n'ayant pas de cas particuliers et de contraintes. Ils sont parfaitement déterministe
- ♦ **Médian** (moyen): projet **intermédiaire** (50 000 < LOC < 300 000) menés avec des équipes mixtes et le nombre de cas particuliers et de tests doivent être plus important que pour les applications de type S
- ♦ **Imbriqué** (complexe): **grand** projet (LOC > 300 000) devant obéir à des ensembles de contraintes et données complexes. Elles ne sont pas déterministes

Development Mode	Project Characteristics			
	Size	Innovation	Deadline/constraints	Dev. Environment
Organic	Small	Little	Not tight	Stable
Semi-detached	Medium	Medium	Medium	Medium
Embedded	Large	Greater	Tight	Complex hardware/ customer interfaces

Table 1: development modes

Modèle Cocomo de base

- ♦ S'applique qu'à l'étape de réalisation et suppose l'existence d'une corrélation entre la taille (en instructions source) d'un programme et la charge consommée:

$$\text{charge (en moisxhommes)} = a \{ \text{KLOC} \}^b$$

$$\text{Durée normale (en mois)} = c \{ \text{charge} \}^d$$

- où a, b, c et d sont des coefficients dépendant de la catégorie du projet, qui à son tour dépend de la taille du logiciel. KLOC représente le nombre, en milliers, de lignes de code ($LOC = \text{Lines Of Code}$); en fait il s'agit du nombre d'instructions source. Taille moyenne de l'équipe correspond à : $\text{charge} / \text{durée normale}$

Projet	a	b	c	d
Simple	2.4	1.05	2.5	0.38
Moyen	3.0	1.12	2.5	0.35
Complexe	3.6	1.20	2.5	0.32

Modèle intermédiaire Cocomo 81

- ♦ Le **modèle Cocomo81** est plus élaboré et prend en compte des facteurs d'ajustement intégrant les conditions de développement (compétence de l'équipe, complexité de l'environnement technique, etc.). La charge se calcule par:

$$\text{charge (en moisxhommes)} = a (EAF) \{KLOC\}^b$$

- où **EAF** (*Effort Adjustment Factor*), qui vaut 1 dans le modèle de base, est calculé à partir de 15 critères regroupés en 4 catégories:

1. **Attribut du produit**
2. **Attribut de l'environnement matériel et logiciel**
3. **Attribut du personnel**
4. **Attribut du projet**

♦ **Attribut du produit:**

- **Fiabilité requise du logiciel (FIAB):** notée sur une échelle allant de très faible, où une défaillance ne pose pas de problème particulier, à très élevée, où une défaillance met en péril la vie humaine, en passant par moyenne, où une défaillance est la cause de pertes recouvrables.
- **Taille de la base de données (DONN):** notée de faible, où la taille de la base de données (en octets) est moins de dix fois le nombre de lignes sources livrées, à très élevée, où la taille de la base de données est plus de mille fois plus grande que le programme, en passant par moyenne, où la taille de la base de données est entre dix et cent fois la taille du système.
- **Complexité du produit (CPLX):** notée sur une échelle allant de très faible à très élevée. Les produits de complexité faible utilisent des opérations d'E/S (entrées/sorties) simples, des structures de données simples et du code «linéaire». Les produits de complexité moyenne utilisent des opérations d'E/S évoluées, multi-fichiers, des procédures de bibliothèque et des communications entre modules. Une complexité élevée peut signifier: traitement parallèle, gestion de données complexes, etc.

♦ **Attribut de l'environnement matériel et logiciel:**

- **Contraintes de temps d'exécution (TEMP):** noté de moyen à très élevé. Moyen signifie que 50% de la puissance disponible sera utilisée, très élevé signifie que 95% de la puissance disponible sera utilisée.
- **Contraintes d'espace mémoire (ESPA):** noté de la même manière que TEMP.
- **Volatilité de la machine virtuelle (VIRT):** la machine virtuelle est la combinaison de matériel et de logiciel sur laquelle le produit logiciel est développé. Ce facteur sera noté faible si cette machine n'est modifiée qu'occasionnellement (une fois par an), moyen si elle est modifiée tous les six mois et très élevé si elle est modifiée toutes les deux semaines.
- **Contraintes du système de développement (CSYS):** notée de très faible pour le développement à l'aide d'un système interactif, à très élevée pour un système non interactif et peu disponible.

Modèle intermédiaire Cocomo 81

- ♦ **Attribut du personnel**: Ils sont tous notés de **très faible**, qui signifie peu ou **pas d'expérience**, à **très élevé**, qui signifie plus de **trois ans d'expérience**, en passant par **moyen**, qui signifie **au moins un an d'expérience**.

- Aptitude à l'analyse (APTA)
- Expérience dans le domaine d'application (EXPA)
- Expérience en machine virtuelle (EXPV)
- Aptitude à la programmation (AFTP)
- Expérience en langage de programmation (EXPL)

Modèle intermédiaire Cocomo 81

- ♦ **Attribut du projet:** Les attributs du projet sont liés à l'utilisation d'outils, au contrôle de l'avancement du projet et à l'utilisation de **méthodes de programmation modernes** telles que la conception fonctionnelle descendante, les revues de conception et de codage, la programmation structurée, etc.
- **Méthodes de programmation modernes (PMOD):** noté de très faible, en l'absence de méthode, à très élevé lorsque la méthode est évoluée et l'équipe expérimentée dans son utilisation
- **Outils logiciels (OLOG):** la disponibilité d'outils logiciels peut avoir un impact significatif sur l'effort nécessaire au développement d'un système. Cette disponibilité est notée de très faible lorsque seuls des outils très primitifs, tels que des assembleurs, sont utilisés, à très élevée lorsque des outils couvrant l'intégralité du cycle de vie sont disponibles
- **Durée du développement requise (DREQ):** cet attribut déterminera l'écart entre la durée de développement et la durée obtenue avec le modèle COCOMO de base. Une valeur très faible signifie qu'on veut écourter la durée, alors qu'une valeur très élevée signifie qu'on veut rallonger la durée. Des valeurs élevées ou des valeurs faibles impliquent toutes deux un effort de développement supplémentaire

Détermination de la charge

♦ **EAF** est le produit de toutes ces valeurs

♦ Par ailleurs, seule la valeur de a change (b, c et d restent inchangés):

Mode	a
Simple	3.2
Moyen	3.0
Complexe	2.8

Facteur de coût	Description	Taux					
		Très faible	Faible	Nominal	Élevé	Très élevé	Extra élevé
Product							
FIAB	Fiabilité requise du système	0.75	0.88	1.00	1.15	1.40	-
DONN	Taille de la base de donnée	-	0.94	1.00	1.08	1.16	-
CPLX	Complexité du produit	0.70	0.85	1.00	1.15	1.30	1.65
Computer							
TEMP	Contrainte temps d'exécution	-	-	1.00	1.11	1.30	1.66
ESPA	Contrainte espace mémoire	-	-	1.00	1.06	1.21	1.56
VIRT	Volatilité machine virtuelle	-	0.87	1.00	1.15	1.30	-
CSYS	Contrainte système de devpt	-	0.87	1.00	1.07	1.15	-
Personnel							
APTA	Aptitude de l'analyste	1.46	1.19	1.00	0.86	0.71	-
EXPA	Expérience en application	1.29	1.13	1.00	0.91	0.82	-
APTP	Aptitude en programmation	1.42	1.17	1.00	0.86	0.70	-
EXPV	Expérience en machine virtuel	1.21	1.10	1.00	0.90	-	-
EXPL	Expérience en langage de prog	1.14	1.07	1.00	0.95	-	-
Project							
PMOD	Méthode program moderne	1.24	1.10	1.00	0.91	0.82	-
OLOG	Outils logiciels	1.24	1.10	1.00	0.91	0.83	-
DREQ	Durée développement requise	1.23	1.08	1.00	1.04	1.10	-

Modèle Cocomo avancé

- ♦ Les facteurs de pondération diffèrent suivant la phase de développement. 4 phases de développement sont proposées:
 - **étude globale** (*RPD : requirements planning and product design*)
 - **étude détaillée** (*DD : detailed design*)
 - **programmation** (*CUT : code and unit test*)
 - **intégration** (*IT : integration and test*)
- ♦ La charge se calcule donc phase par phase à partir de la formule du Cococmo81 intermédiaire avec une pondération différente pour chaque phase. Par exemple pour APTA:

Cost Driver	Rating	RPD	DD	CUT	IT
Aptitude à l'Analyse (APTA)	Très Faible	1.80	1.35	1.35	1.50
	Faible	0.85	0.85	0.85	1.20
	Nominal	1.00	1.00	1.00	1.00
	Élevé	0.75	0.90	0.90	0.85
	Très élevé	0.55	0.75	0.75	0.70

Modèle Application Composition de Cocomo II

- ♦ Utilisé dans le **prototypage**. Au lieu de qualifier la taille du logiciel en SLOC (lignes de code source), on utilise ici les "**points objets**".
- ♦ Les objets considérés sont les **écrans**, les **rapports** et les **programmes** en langage de 3ème génération.
- ♦ Chaque objet est caractérisé comme **simple**, **moyen**, **complexe**. Les programmes en langage de troisième génération sont caractérisés comme complexes.
- ♦ Pour les écrans et les rapports, la complexité dépend du nombre de tables de données source et du nombre de vues

Modèle Application Composition de Cocomo II

ECRAN	nombre de tables de données sources		
nombre de vues	<4	<8	>=8
<3	simple	simple	moyen
3 à 7	simple	moyen	complexe
8 et +	moyen	complexe	complexe

Rapport	nombre de tables de données sources		
nombre de vues	<4	<8	>=8
0 à 1	simple	simple	moyen
2 à 3	simple	moyen	complexe
4 et +	moyen	complexe	complexe

- ♦ Après avoir dénombré les objets par type et complexité, un facteur (poids de complexité) est affecté à chaque objet:

type/complexité	simple	moyen	complexe
écran	1	2	3
rapport	2	5	8
composant 3GL	-	-	10

Modèle Application Composition de Cocomo II

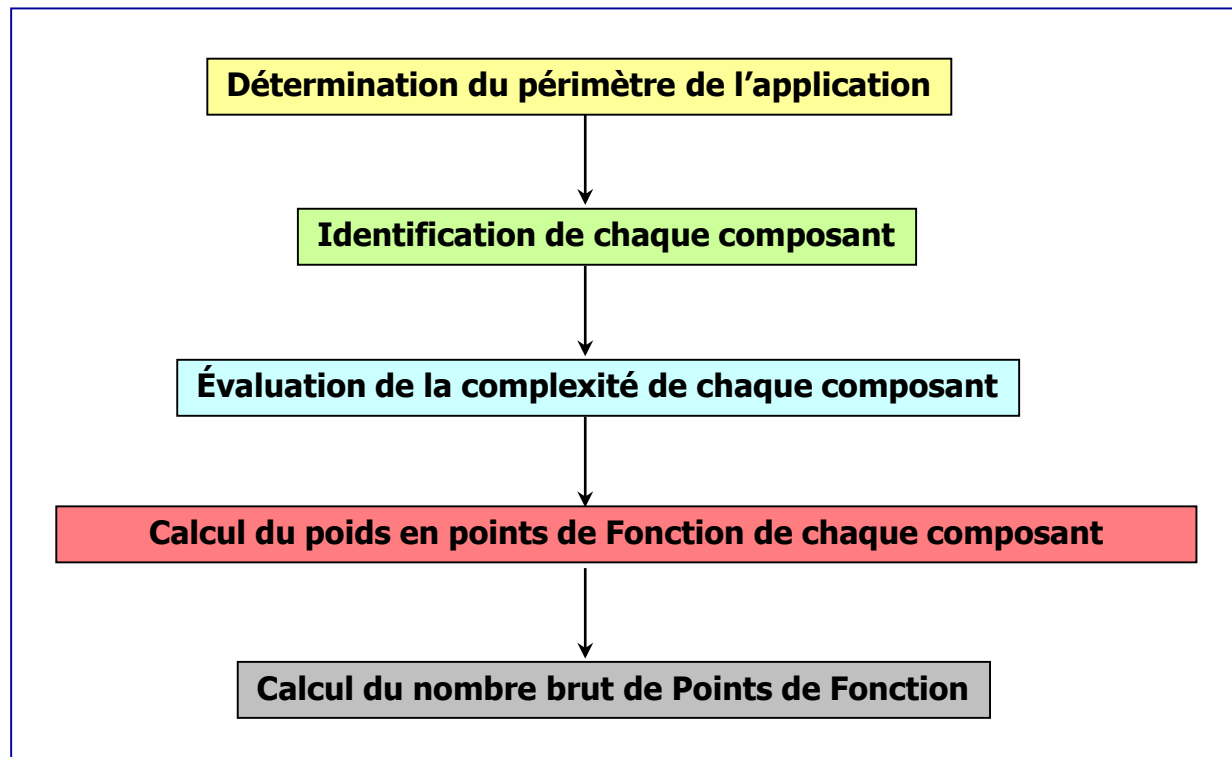
- ♦ De plus, le fait de réutiliser des composants déjà existants est pris en considération avec un coefficient donné par un pourcentage : **R** (pourcentage d'objets réutilisés). Par ailleurs, la productivité est également prise en compte selon le tableau suivant.

expérience de développement	très basse	basse	moyenne	forte	très forte
performance du produit	très basse	basse	moyenne	forte	très forte
push	4	7	13	25	50

- ♦ La formule magique donnant la charge est alors la suivante:
charge (en moisxhommes) = (**nombre de points objet**)*(100-**R**)/(100***push**)
- ♦ Pour un projet où toutes les composantes sont réutilisées (**R = 100**), la charge de développement est nul

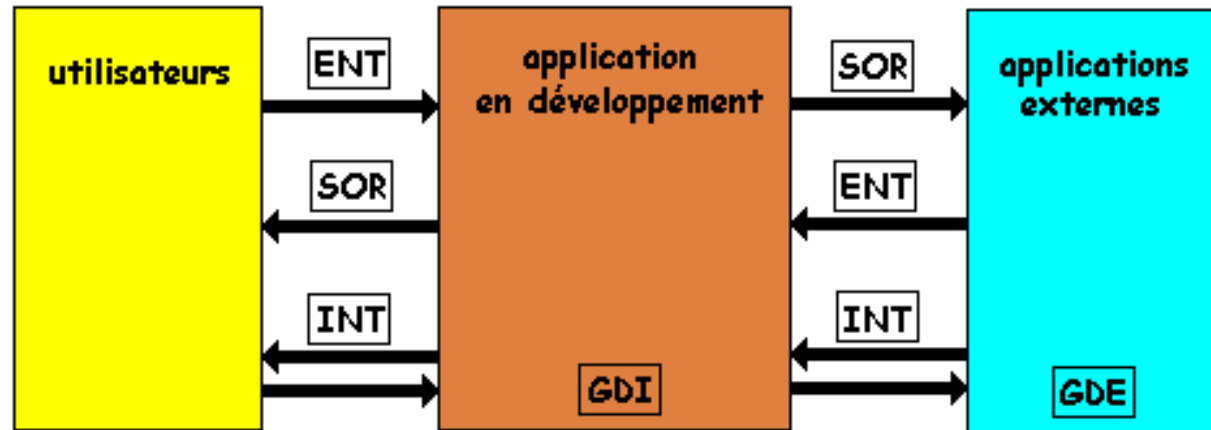
Méthode des points fonctionnels

- ◆ Définie Chez IBM par Alain Albrecht (1979) et s'appuie sur une **description fonctionnelle du système à développer**, en terme d'unités d'oeuvre, mais aussi en terme de complexité. A chaque **couple (unité, complexité)** on **affecte** un poids appelé **point de fonction**
- ◆ Étapes du comptage en nombre brut de points de fonction:



Méthode des points fonctionnels

◆ Composants fonctionnels :



- Groupe logique de données internes (**GDI**), pouvant être constitué de sous-ensembles logiques de données (**SLD**) eux-mêmes composées de données élémentaires (**DE**)
- Groupe logique de données externes (**GDE**)
- Entrée de traitement (**ENT**), les informations saisies se réfèrent à des GDI ou des **GDR** (groupe de données référencés), composés de DE
- Sortie de traitement (**SORT**), GDR composés de DE
- Interrogation (**INT**), combinaison E/S qui ne fait pas de mise à jour de GDI

Méthode des points fonctionnels

- ♦ La complexité des composants fonctionnels est déterminée comme suit:

Complexité des GDI - GDE			
	1 à 19 DE	20 à 50 DE	51 DE ou plus
1 SLD	faible	faible	moyenne
2 à 5 SLD	faible	moyenne	élevée
6 SLD ou plus	moyenne	élevée	élevée

Complexité des SOR - INT			
	1 à 5 DE	6 à 19 DE	20 DE ou plus
0 ou 1 GDR	faible	faible	moyenne
2 à 3 GDR	faible	moyenne	élevée
4 GDR ou plus	moyenne	élevée	élevée

Complexité des ENT			
	1 à 5 DE	6 à 19 DE	20 DE ou plus
0 ou 1 GDR	faible	faible	moyenne
2 à 3 GDR	faible	moyenne	élevée
4 GDR ou plus	moyenne	élevée	élevée

Méthode des points fonctionnels

- ♦ On affecte ensuite des **points de fonction** à chaque **unité** d'oeuvre suivant sa complexité :

complexité	faible	moyenne	élevée
points de fonction des GDI	7	10	15
points de fonction des GDE	5	7	10
points de fonction des ENT	3	4	6
points de fonction des SOR	4	5	7
points de fonction des INT	3	4	6

- ♦ Connaissant le nombre d'unités de chaque type et leur complexité, on obtient le nombre de points pour chacune d'elles . On en fait ensuite **la somme pour obtenir le nombre brut de points de fonction NBPF.**

Méthode des points fonctionnels

- ♦ On ajuste ensuite **NBPF** par une opération de correction basée sur 14 caractéristiques générales du système. Pour chaque caractéristique, on évalue le degré d'influence **DI** compris entre **0** et **5** (0 : aucune influence, 5 : très grande influence)
- ♦ On obtient le nombre net de points de fonctions **NNPF** suivant la formule:
$$\text{NNPF} = [\underbrace{0,65 + (\sum \text{DI})/100}_{\text{Facteur d'ajustement ou facteur technique de complexité}}] * \text{NBPF}$$

Degré d'influence	
communication de données	système distribué
performance	intensité d'utilisation de la configuration matérielle
taux de transaction	saisie interactive
convivialité	mise à jour en temps réel des GDI
complexité des traitements	réutilisation du code de l'application
facilité d'installation	facilité d'exploitation
portabilité de l'application	facilité d'adaptation

Méthode des points fonctionnels

- ♦ Albrecht émet l'hypothèse qu'il existe une relation entre le nombre de points de fonction et la charge de développement. On peut passer du **NNPF** à la **charge** en multipliant le premier par un ratio dépendant du type de projet. Le tableau ci-dessous donne un ordre de grandeur du coefficient de transformation:

taille du projet	en fin d'étude préalable	en fin d'étude détaillée
petit	2 joursxhommes	2 joursxhommes
moyen	3 joursxhommes	2 joursxhommes
grand	4 joursxhommes	2 joursxhommes

- ♦ La méthode de **PF** permet aussi de donner une estimation sur le nombre d'instructions source (**LOC**: Line Of Code) utile pour COCOMO via la formule :

$$\mathbf{LOC} \text{ (langage procédural)} = 118,7 * \mathbf{NNPF} - 6490$$

Chapitre 3

Planification d'un projet

Éléments de planification

- ♦ La fonction du chef de projet et de fournir un plan pour:
 - l'exécution des activités, l'utilisation des ressources matérielles, personnel
 - s'assurer que la réalisation du projet est conforme aux objectifs fixés
- ♦ Le planning est la prévision dans le temps de l'exécution d'un projet. Ceci revient à définir, parmi différentes possibilités, **l'ordonnancement d'un ensemble d'opérations** sur le plan des **délais** et sur le plan de **l'utilisation des ressources**.
- ♦ Connaissant le "poids" du projet en terme de charges et de durée, on détermine la succession des tâches permettant d'aboutir à un calendrier des opérations **optimal**
- ♦ Dans le développement d'un projet, le plan permet:
 - d'effectuer régulièrement des **contrôles de suivi**
 - d'apporter éventuellement des **modifications** au calendrier

Décomposition structurée des activités (WBS)

- ♦ La planification commence par un **recensement des tâches à réaliser** dans le projet. La décomposition structurée des activités (**WBS Work Breakdown Structure**) permet de recenser l'ensemble des activités d'un projet et de les décomposer
- ♦ L'organisation des activités dans le temps revient d'abord à évaluer les dépendances entre elles tout en estimant l'effort nécessaire pour chaque activité (durée maximum et minimum)
- ♦ La **WBS** doit être **complète** et **non ambiguë** car elle conditionne le budget. Elle doit définir des activités dont le **résultat est mesurable**, ces activités feront l'objet d'affectation de ressources
- ♦ Cette décomposition en tâches élémentaires permet au chef de projet d'établir **le graphe PERT** et **le diagramme de Gantt** pour le suivi budgétaire du projet

Énumération des activités (*WBS*)

- ♦ Critères d'arrêt de décomposition et de complétude microscopique:
 - **Entrées et sorties** (livrables) définies
 - **Évènements** d'entrée et de sortie définis
 - **Processus** défini
 - **Ressources** nécessaires définies
 - **Coût** (effort), durée, facilement calculables
 - **État d'avancement** mesurable
 - **Déroulement** de l'activité indépendant des autres
 - **Ampleur** raisonnable de l'activité

Program Evaluation and Review Technique (*PERT*)

- ♦ ***PERT*** est un modèle de gestion de projets inventé en **1958** par le *US Navy* du département de défense pour un projet de lancement de missile
- ♦ *PERT* utilise une représentation en graphe pour déterminer:
 - le **temps nécessaire pour compléter chaque tâche** en tenant compte des contraintes d'enchaînement (ordonnancement)
 - le **temps minimale nécessaire pour la terminaison** du projet complet

Graphe des tâches et événement

- ♦ Une **tâche** est représentée par un rectangle et les transitions par des flèches



- ♦ Un **événement** (fin d'une tâche par exemple) est représenté par un ovale et les tâches par des flèches.

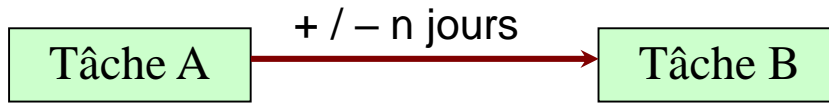


Types de contraintes

- ♦ La réalisation des différentes tâches doit être projeté en respectant les **contraintes logiques d'enchaînement** et les **contraintes de disponibilités de ressources**. On cite 4 catégories:
 - **Contrainte d'antériorité**: respect de règles de succession entre les diverses tâches
 - **Contrainte de localisation temporelle**: exprime que le début et la fin d'une tâche sont effectués à l'intérieur d'une période de temps imposée
 - **Contrainte disjonctive**: exprime que plusieurs opérations ne peuvent être simultanées (utilisation de mêmes ressources humaines et matériels dont le nombre ne saurait être augmenté)
 - **Contrainte cumulative**: exprime qu'à un moment donné, les moyens nécessaires ne peuvent dépasser un certain plafond

Liens entre tâches

♦ Lien **fin** → **début**:



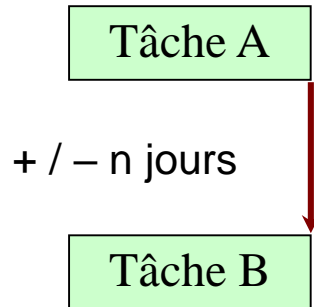
la tâche suivante (B) ne peut commencer que si la tâche précédente (A) est finie,

(-) veut dire qu'on peut commencer à l'avance

(+) veut dire qu'on commence en retard

Liens entre tâches

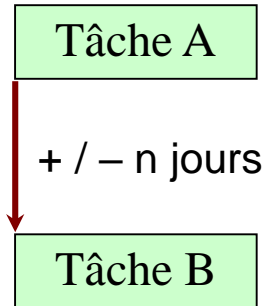
- ♦ Lien **fin** → **fin**:



la tâche suivante (B) se termine quand la tâche précédente (A) se termine
(parallélisation partielle)

Liens entre tâches

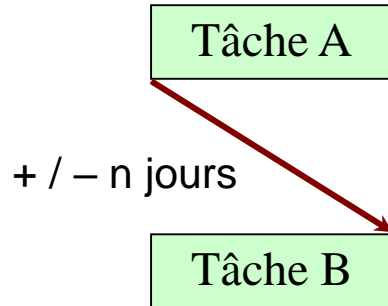
- ♦ Lien **début** → **début**:



le début de la tâche précédente (A) déclenche le début de la tâche suivante (B)
(parallélisation partielle)

Liens entre tâches

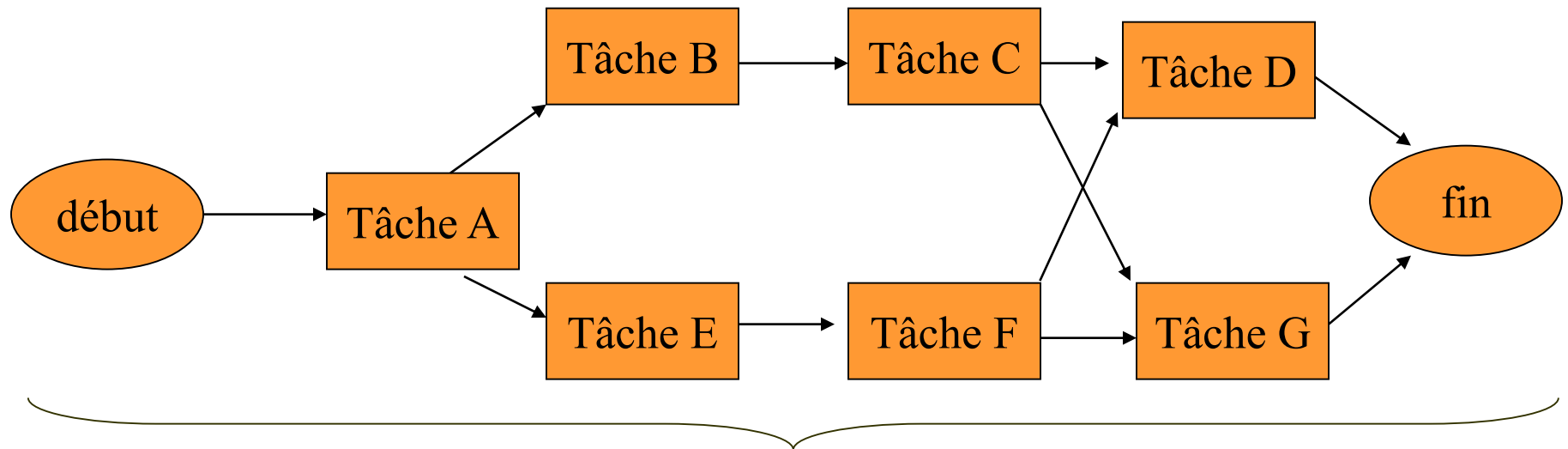
- ♦ Lien **début** → **fin**:



le début de la tâche précédente (A) marque la fin de la tâche suivante (B)

PERT: représentation Réseau

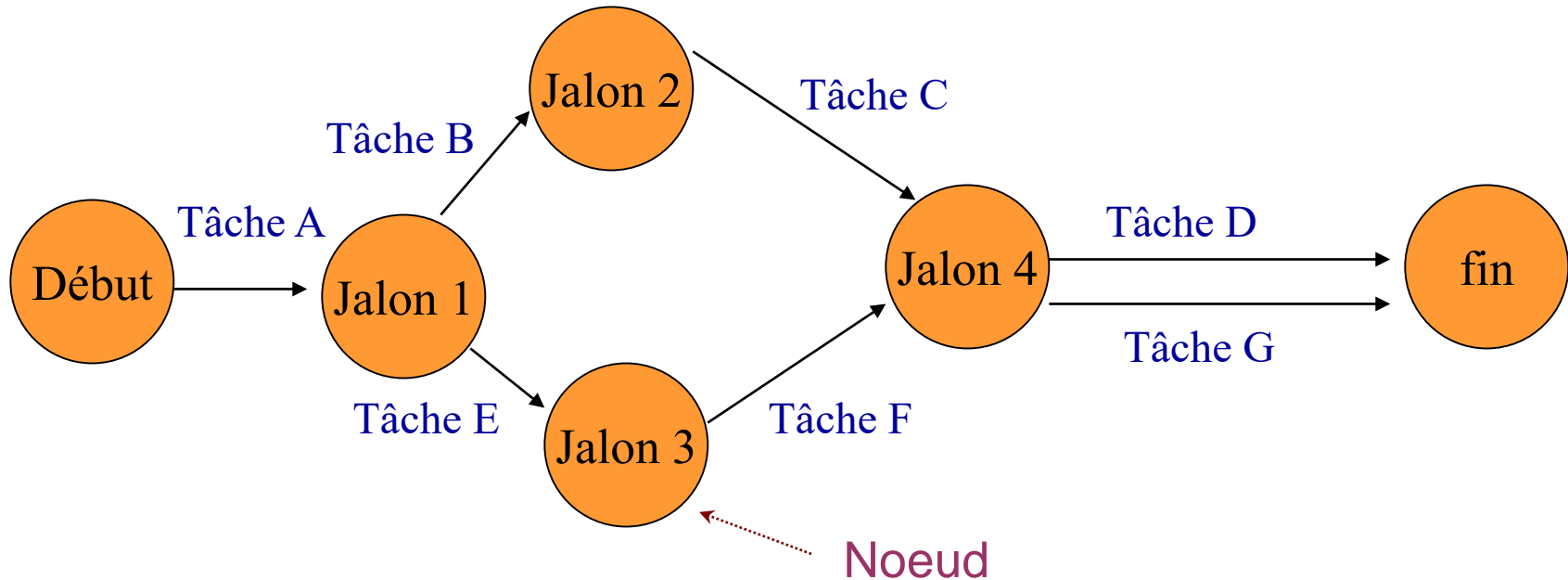
- ♦ Exemple de **représentation Réseau** de graphes de tâches avec parallélisme fort. Les flèches représentent les liens



Représentation Réseau

PERT: représentation normale

- ♦ Exemple de graphes des potentiels événements



- ♦ Jalon:

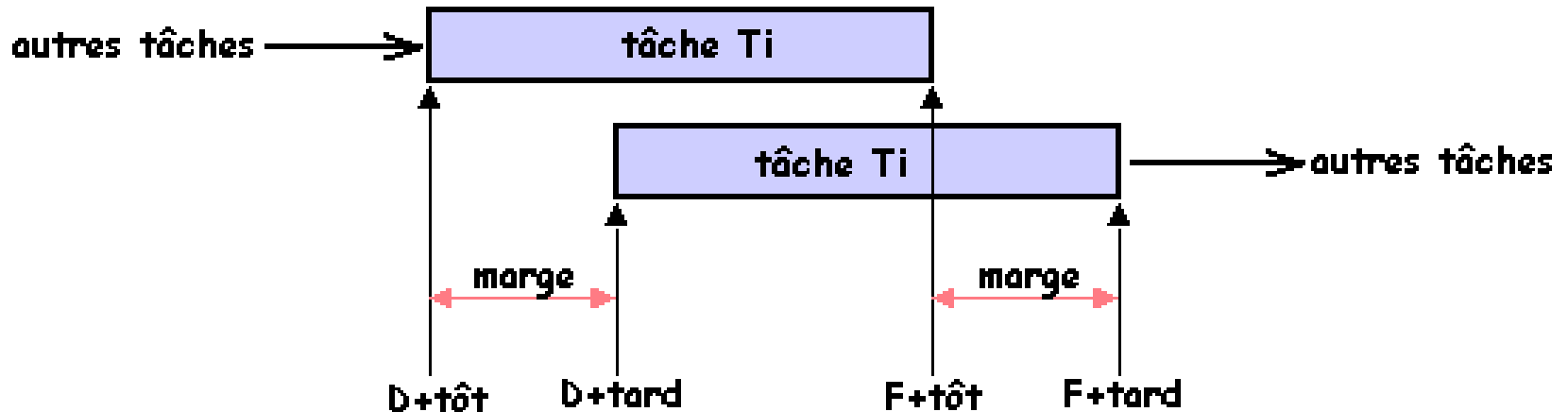
- Point de référence marquant un événement important dans l'avancement du projet
- Utilisé pour contrôler l'avancement du projet
- Un jalon correspond à une tâche, en général de durée nulle

Paramètres clés du réseau PERT

- ♦ Parmi tous les chemins d'un graphe PERT, il en existe un appelé **chemin critique**. Il relie les tâches "critiques" qui sont les tâches dont le retard provoquera un retard sur la durée prévue du projet entier
- ♦ Supposons un projet qui commence au temps D et finit au temps F (date limite). Pour chaque tâche, on peut calculer les paramètres suivants:
 - **dates au plus tôt** : début (D_tôt) et fin (F_tôt)
 - **dates au plus tard** : début (D_tard) et fin (F_tard)
 - **marge** (*total Slack*): $(D_{\text{tard}}) - (D_{\text{tôt}}) = (F_{\text{tard}}) - (F_{\text{tôt}})$

Dates pour tâche au sein du projet

- ♦ Supposons une tâche T_i se trouvant au sein du projet et de durée estimée à d_i :



- dates de début au plus tôt et de fin au plus tôt:

$$\text{début } (D_{tôt}(T_i)) = \sup \{ F_{tôt}(\text{prédécesseurs}(T_i)) \}$$

$$\text{fin } (F_{tôt}(T_i)) = D_{tôt}(T_i) + d_i$$

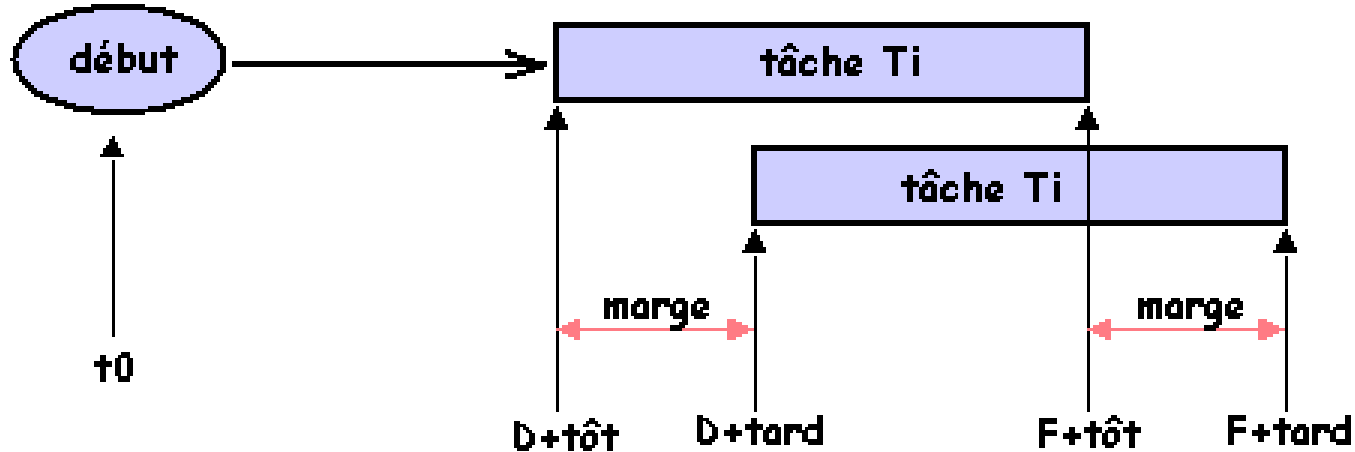
- dates de fin au plus tard et de début au plus tard:

$$\text{fin } (F_{tard}(T_i)) = \inf \{ D_{tard}(\text{successeurs}(T_i)) \}$$

$$\text{début } (D_{tard}(T_i)) = F_{tard}(T_i) - d_i$$

Dates pour tâche en début du projet

- ♦ Supposons que la tâche T_i , la durée estimé est d_i , se trouve en début de projet:



- date de début au plus tôt :

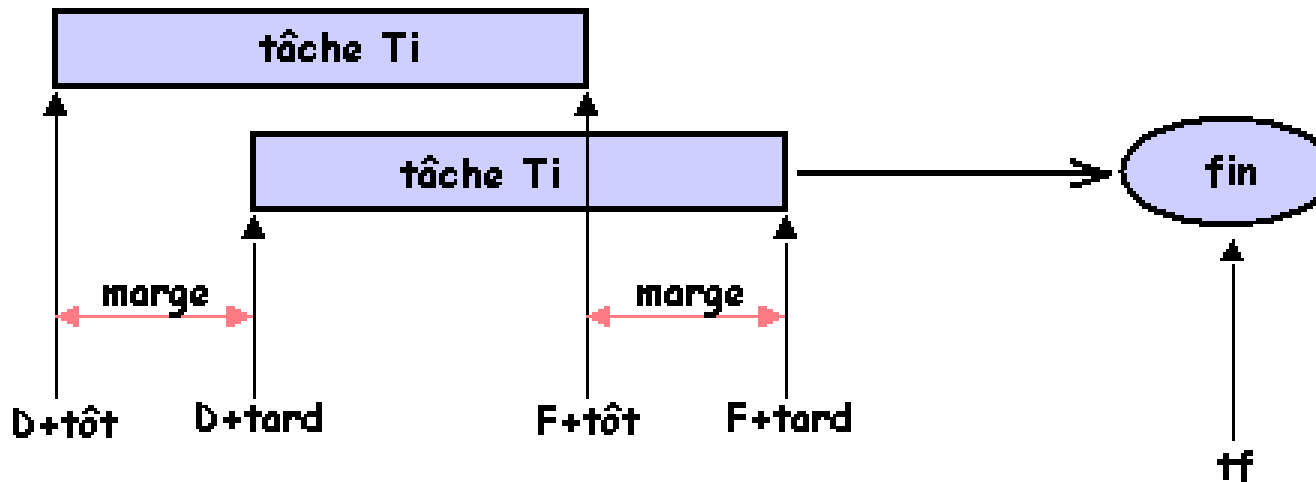
$$\text{début } (D_tôt(T_i)) = t_0 \text{ (date de début de projet)}$$

- date de fin au plus tôt:

$$\text{fin } (F_tôt(T_i)) = t_0 + d_i$$

Dates pour tâche en fin du projet

- ♦ Supposons que la tâche T_i , la durée estimé est d_i , se trouve en fin de projet:



- date de fin au plus tard :

$$\textbf{fin} (F_{\text{tard}}(T_i)) = tf$$

- date de début au plus tard:

$$\textbf{début} (D_{\text{tard}}(T_i)) = tf - d_i$$

Paramètres clés du réseau PERT

- ♦ Avec les trois séries de règles précédentes, il est possible de déterminer le chemin critique. On procède de la manière suivante:
 - En commençant par les **tâches de début**, on détermine les dates **au plus tôt**
 - puis en commençant par les **tâches de fin**, on détermine les tâches **au plus tard**
 - On calcule ensuite, pour chaque tâche, la marge totale

Le chemin critique est le chemin correspondant aux marges nulles, ou dans le cas échéant aux marges les plus petites

- ♦ On parle aussi de la marge libre (**free Slack**) qui n'est rien d'autre que le retard maximum que peut encourir une activité sans compromettre le moment de démarrage au plus tôt des activités conséquentes

Exemple d'application

- ♦ Soit un projet constitué des tâches suivantes (durée en jours):

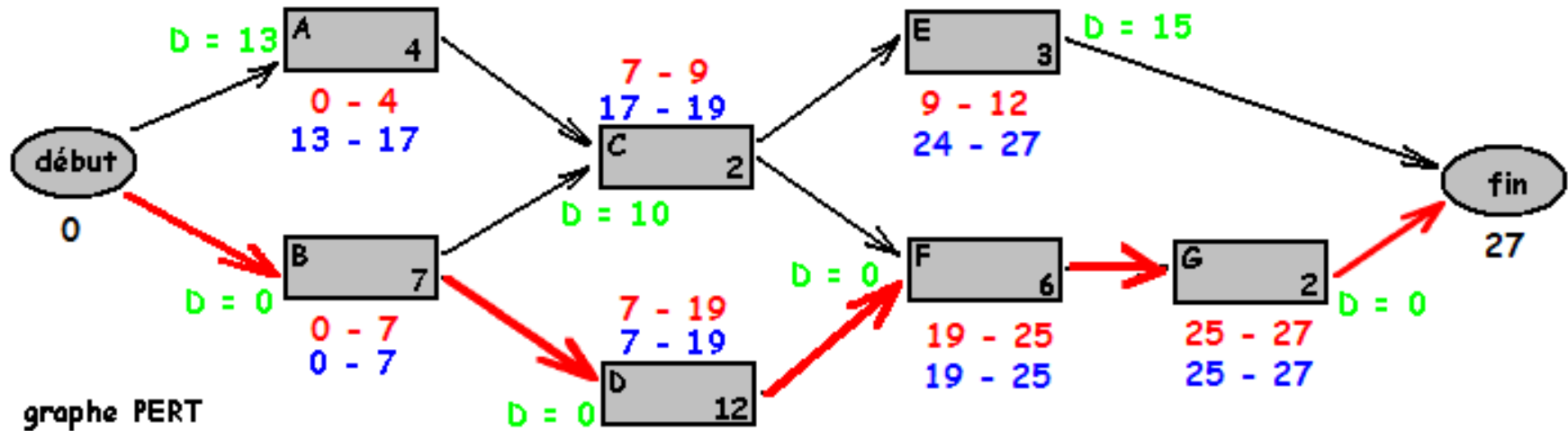
tâche	durée	prédécesseurs	successeurs
A	4		C
B	7		C, D
C	2	A, B	E, F
D	12	B	F
E	3	C	
F	6	C, D	G
G	2	F	

Donnez le graphe PERT et déterminez le chemin critique?

Exemple d'application

♦ Graphe PERT et chemin critique:

- **Dates au plus tôt**
- **Dates au plus tard**
- **Marge**



→ Le chemin critique est l'épine dorsale de l'évolution du projet. Les modifications portent ensuite sur les tâches qui ne sont pas sur le chemin critique

Diagramme de Gantt

- ♦ Le graphe PERT a été établi sans tenir compte des **contraintes des ressources** et de leurs disponibilités (ressources infinies, maximum de parallélisme, délai d'exécution minimum)
- ♦ A partir du graphe de PERT, on peut dresser le **diagramme de Gantt** qui permet d'établir un calendrier de travail en considérant les ressources et leurs disponibilités. A chaque opération on **affectera une ressource**
- ♦ Supposons que l'on dispose de **deux ressources** (personnes) interchangeables. Une possibilité de programmation serait:

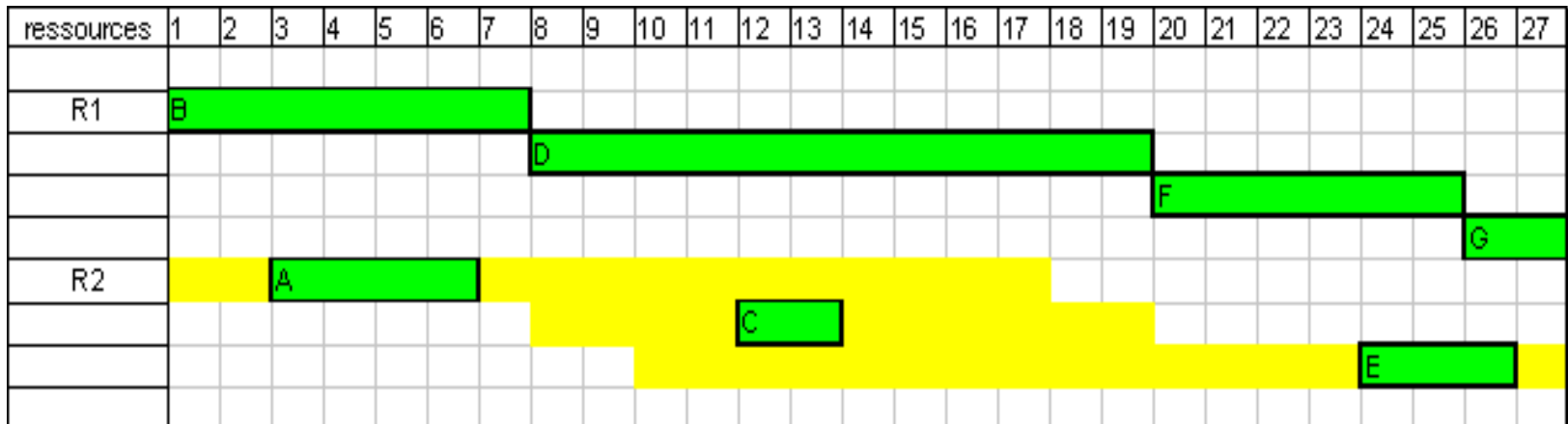
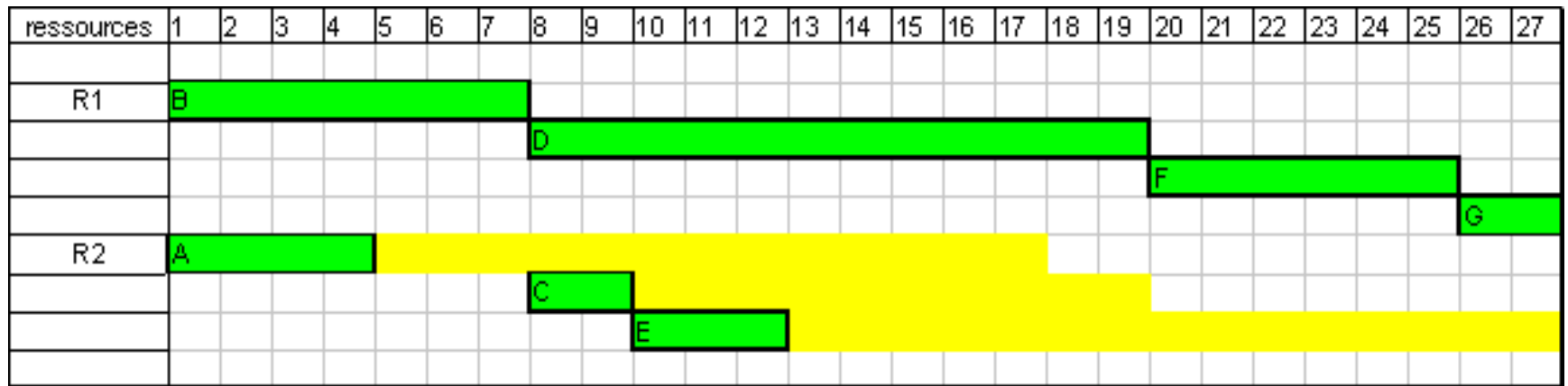


Diagramme de Gantt

- On peut aussi planifier **au plutôt**:



- On peut aussi planifier **au plus tard**:

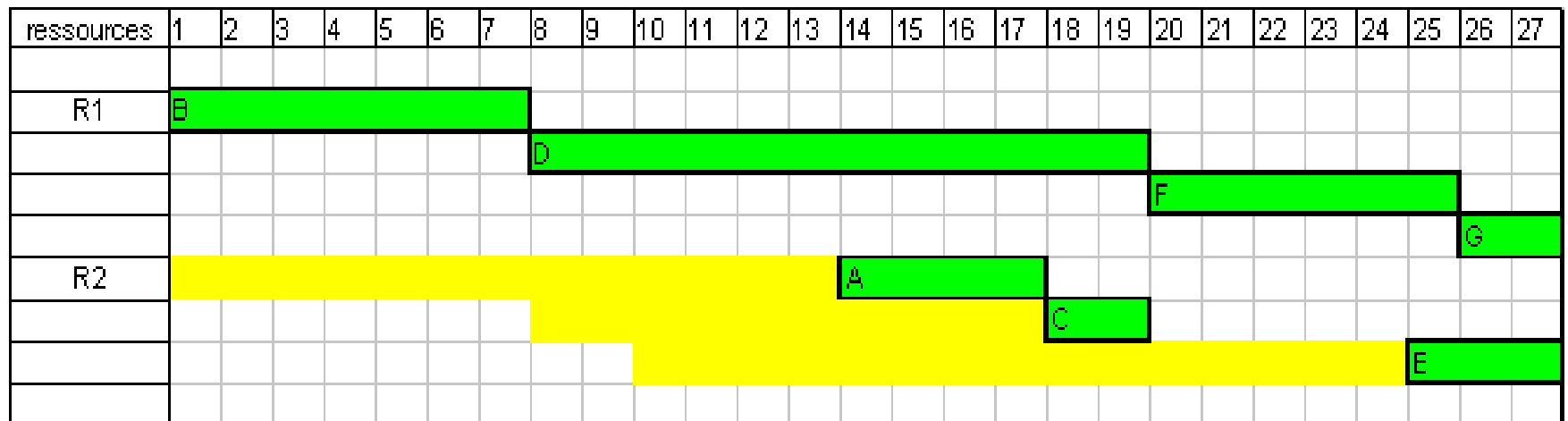
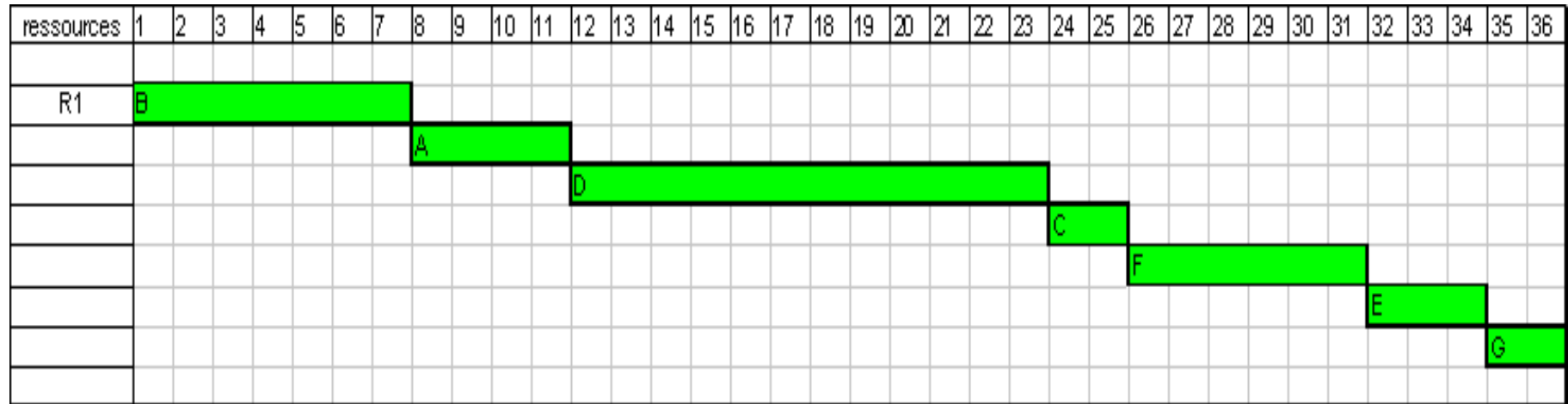


Diagramme de Gantt

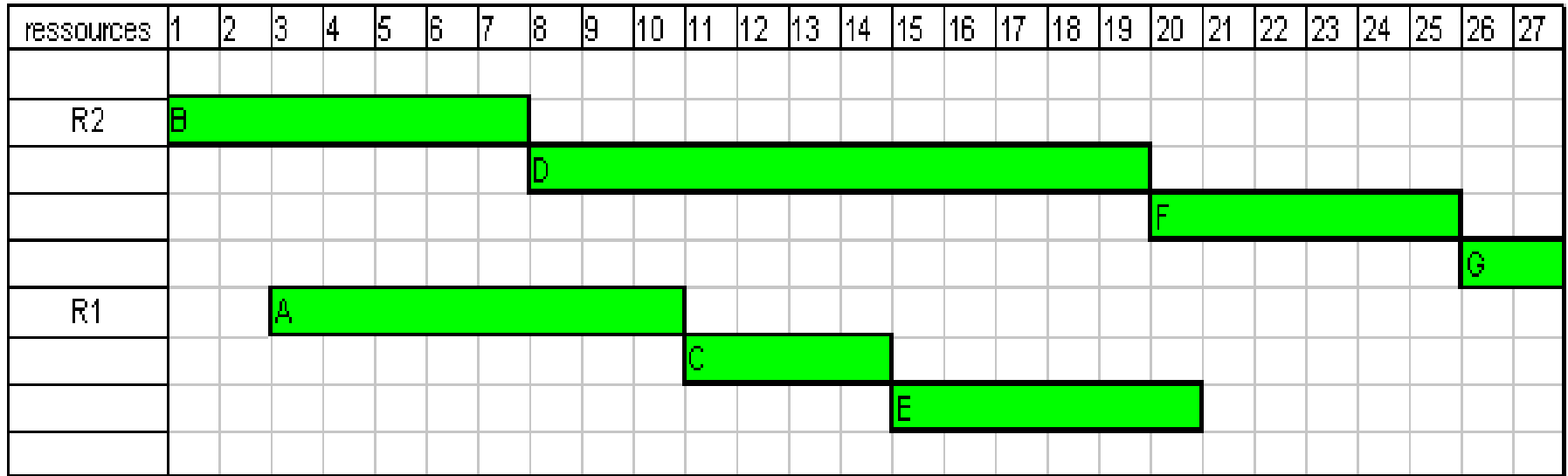
- ♦ Si l'on ne dispose que d'une seule ressource, bien entendu, la durée du projet sera rallongée; le diagramme de Gantt devient donc:



- ♦ Deux opérations peuvent être appliquées sur le diagramme de Gantt:
le nivellement et **le lissage**.

Diagramme de Gantt

- ♦ **Le nivellement** consiste à maintenir les ressources en dessous d'une certaine limite, en général une réduction allonge la durée du projet. le passage de deux ressources à une seule en est un exemple.
- ♦ **Le lissage** consiste en un ajustement de répartition de la charge de travail de chaque ressource pour éviter les surcharges ou les sous charges. Si, par exemple, la ressource R1 travaille à mi-temps et la ressource R2 à temps complet, on pourra avoir, avec l'exemple étudié:



PERT probabiliste

- ♦ Il existe une version du PERT qui prend en compte les aléas sur les dates et les durées de réalisation des tâches. Parmi ses objectifs:
 - Calculer la **probabilité qu'un projet se termine dans un délai déterminé**
 - Estimer le **risque** et l'incertitude de chaque tâche, utile pour des projets à forte incertitude
- ♦ La méthode PERT probabiliste prend en considération l'aléatoire en gestion de projets. Le fait que la durée de chaque tâche comprend une variable aléatoire implique que la durée totale de projet comprend aussi une variable aléatoire
- conditions d'application:
 - nombre de tâches suffisamment élevé (min. 4 sur le chemin)
 - ordre de grandeur semblable pour toutes les tâches
 - indépendance entre les durées des tâches

PERT probabiliste

1ère étape :

Concerne la recherche de la **loi de probabilité de la durée de chaque tâche T_i** . Dans la pratique on adopte une loi universelle: la loi Bêta basée sur trois paramètres :

- la durée **optimiste** de la tâche T_i : $\text{topt}(T_i)$
- la durée **pessimiste** de la tâche T_i : $\text{tpes}(T_i)$
- la durée **vraisemblable** de la tâche T_i : $\text{tvra}(T_i)$

• On définit quelquefois le risque par la quantité:

$$R(T_i) = [\text{tpes}(T_i) - \text{topt}(T_i)] / \text{tpes}(T_i).$$

Risque faible:

$$R < 0.25:$$

Risque moyen:

$$0.25 < R < 0.5;$$

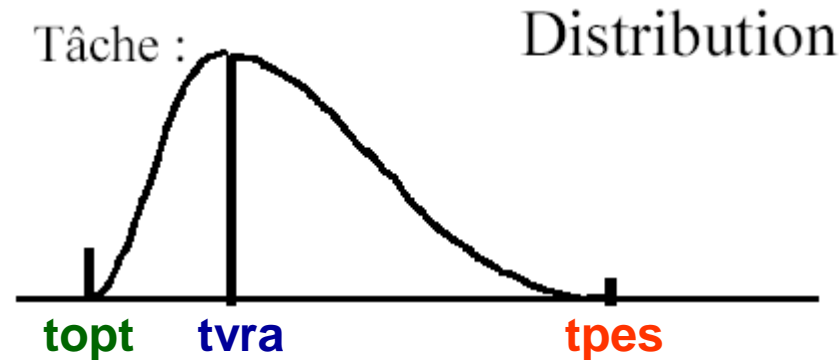
Risque fort:

$$R > 0.5$$

PERT probabiliste

2ème étape :

A partir des paramètres précédents, on calcule de nouveaux paramètres (pour la loi Bêta) :



- la durée probable de la tâche T_i :

$$\mathbf{tprob(T_i)} = [\mathbf{topt(T_i)} + 4\mathbf{tvra(T_i)} + \mathbf{tpes(T_i)}] / 6$$

- l'écart-type

$$\mathbf{e(T_i)} = [\mathbf{tpes(T_i)} - \mathbf{topt(T_i)}] / 6$$

- la variance

$$\mathbf{v(T_i)} = \mathbf{e(T_i)}^2$$

plus la variance est grande moins l'estimation sera précise

3ème étape :

Pour chaque chemin, on peut alors calculer:

- la durée estimée pour toutes les tâches T_i du chemin:

$$D_{est} = \sum_i t_{prob}(T_i)$$

- la variance estimée pour toutes les tâches T_i du chemin:

$$V_{est} = \sum_i e(T_i)^2$$

- l'écart-type estimé :

$$E_{est} = V_{est}^{1/2}$$

PERT probabiliste

- ♦ On suppose usuellement que la durée des chemins obéit à la loi normale (de Gauss) de paramètres **Dest** et **Eest**. En utilisant une table de Gauss on peut alors en déduire soit une durée à une probabilité fixée, soit une probabilité d'achèvement du projet dans un délai donné
- ♦ On obtient la durée du chemin pour laquelle la probabilité d'avoir terminé le projet sera **p**:

$$D(p) = \text{Dest} + \text{Eest} \times G(p)$$

où G est la fonction associée à la loi normale centrée réduite (extrait):

p	$G(p)$	p	$G(p)$
99,9	3,00	89,1	1,23
99	2,31	85,1	1,04
98	2,06	70,2	0,53
97	1,88	50	0
95	1,65	42,1	-0,2
92,1	1,41	34,5	-0,4
90	1,28	27,4	-0,6

Table de Gauss

p	G(p)	p	G(p)	p	G(p)
99.9	3	90	1.28	42.1	-0.2
99	2.31	89.1	1.23	34.5	-0.4
98	2.06	85.1	1.04	27.4	-0.6
97	1.88	80.2	0.85	21.2	-0.8
96	1.75	75.2	0.68	15.9	-1
95	1.65	70.2	0.53	11.5	-1.2
94.1	1.56	65.2	0.39	8.1	-1.4
93.1	1.48	60.3	0.26	5.5	-1.6
92.1	1.41	55.2	0.13	3.6	-1.8
91	1.34	50	0	2.3	-2

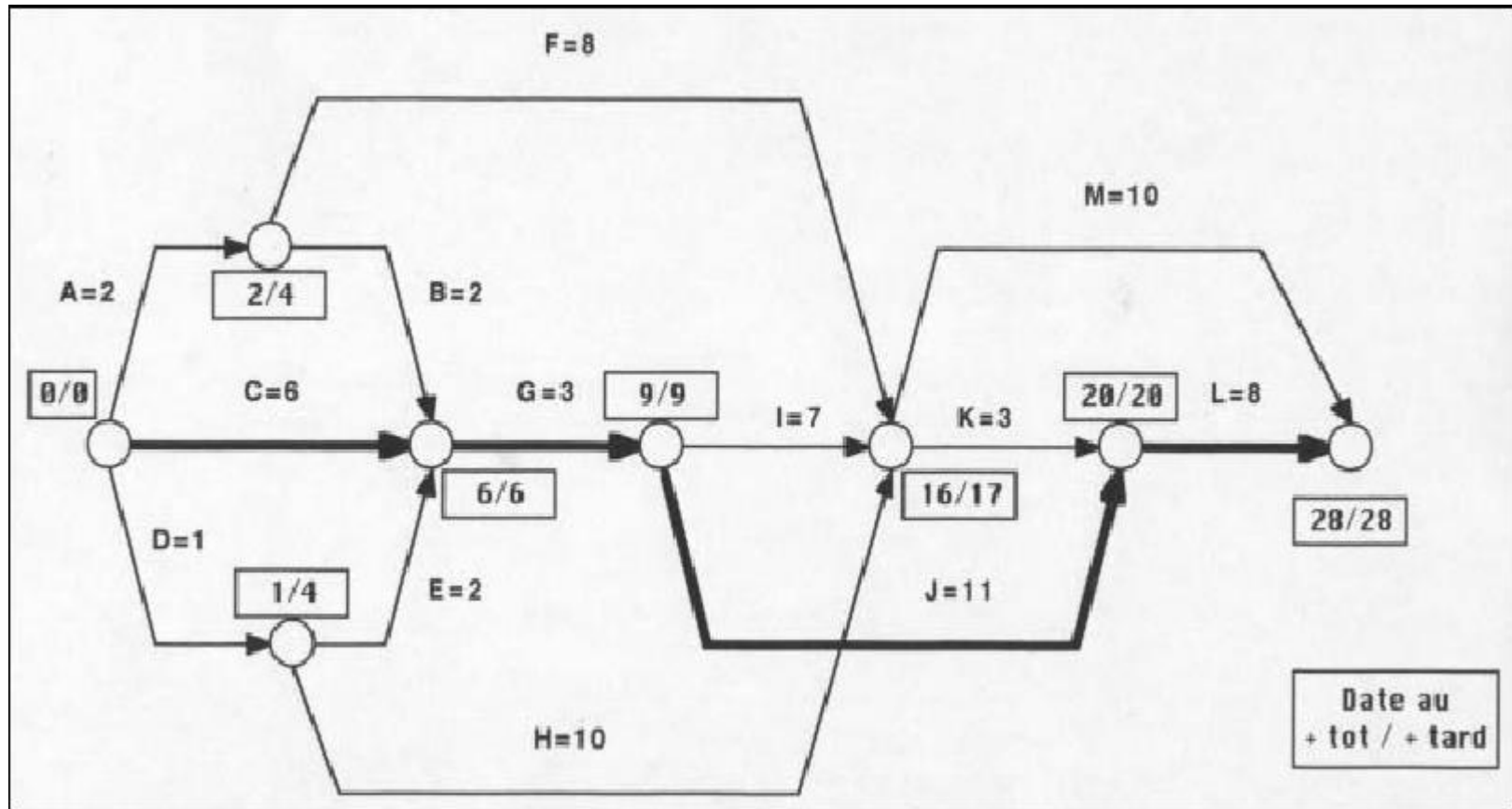
Exemple d'application

♦ On désire planifier un projet comprenant 13 tâches repérées 1 à 13 avec la logique suivante:

TACHE	DUREE	ANTECEDANTS	SUIVANTS
A	2	DEBUT	B,F
B	2	A	G
C	6	DEBUT	G
D	1	DEBUT	E,H
E	2	D	G
F	8	A	K,M
G	3	B,C,E	I,J
H	10	D	K,M
I	7	G	K,M
J	11	G	L
K	3	F,H,I	L
L	8	J,K	FIN
M	10	F,H,I	FIN

- Tracer le planning PERT normal
- Calculer les dates au plutôt et les dates au plus tard
- Tracer le chemin critique
- Déterminer pour chaque tâche la valeur des marges: totale et libre

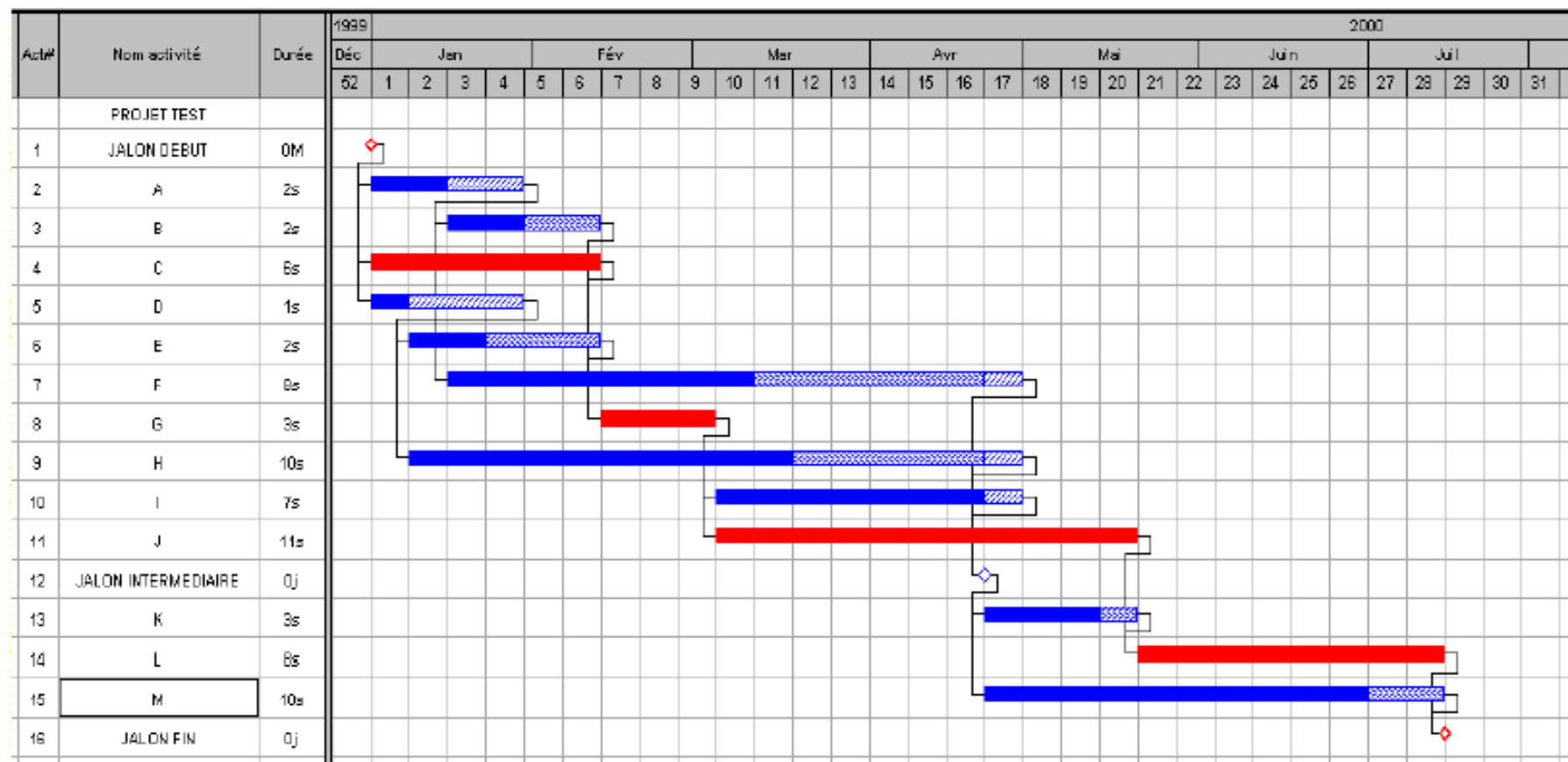
PERT: Tracé normal et calcul des dates



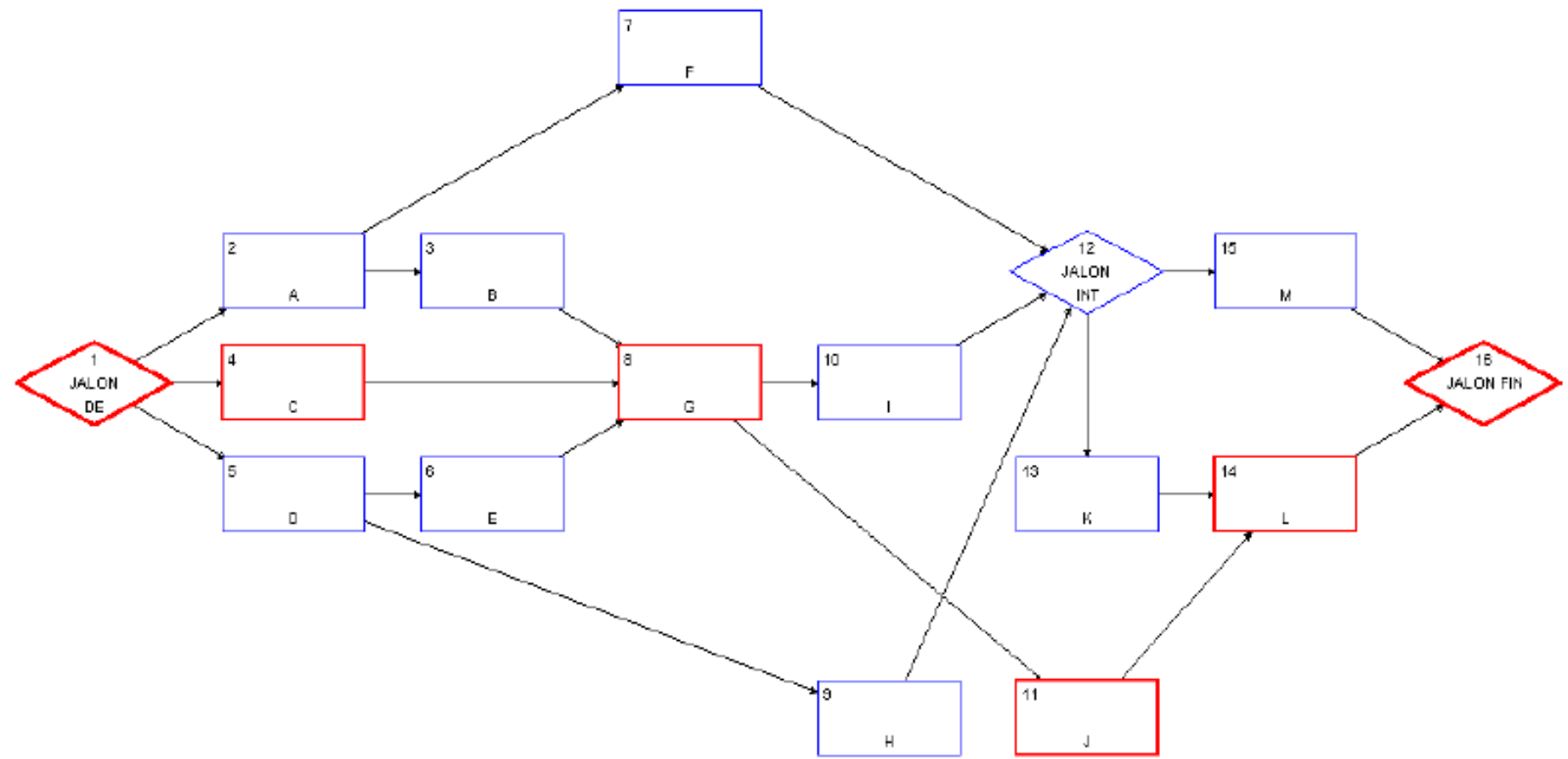
PERT: Calcul des marges

TACHE	DEBUT		DUREE	FIN		MARGE		CRITICITE	
	au plus tôt	au plus tard		au plus tôt	au plus tard	Totale	Libre	Critique	Subcritique
A	0	0	2	2	4	2	0		
B	2	4	2	6	6	2	2		
C	0	0	6	6	6	0	0	x	
D	0	0	1	1	4	3	0		
E	1	4	2	6	6	3	3		
F	2	4	8	16	17	7	6		
G	6	6	3	9	9	0	0	x	
H	1	4	10	16	17	6	5		
I	9	9	7	16	17	1	0		x
J	9	9	11	20	20	0	0	x	
K	16	17	3	20	20	1	1		x
L	20	20	8	28	28	0	0	x	
M	16	17	10	28	28	2	2		x

Représentation Gantt



Représentation Réseau



Représentation « MILESTONES »

- ♦ Cette représentation synthétique permet de visualiser l'ensemble des dates importantes du projet:

	2000											
	J	F	M	A	M	J	J	A	S	O	N	D
Début du Projet	▼											
Fin des Etudes		▼										
Début de l' intégration				▼								
Livraison client							▼					

Durées

- ♦ Il faut distinguer entre la durée estimée et la durée réalisée
- ♦ La durée critique est la durée minimale du projet (sans tenir compte des ressources)
- ♦ Durée du projet est le temps entre le début et la fin du projet (en tenant compte des ressources à disposition)

$$\text{Durée critique} \leq \text{durée du projet}$$

- ♦ Charge (aussi appelée Durée effort)
= $\sum (\text{Durée tâche } i * \text{taux de mobilisation tâche } i)$
- ♦ Taux de mobilisation est le nombre d'unités de ressources mobilisées en même temps: Nombre d'unités (1, 2, 3 personnes) ou part de temps (20%, 100%)

$$\text{Charge} = \text{Durée} * \text{Taux de mobilisation}$$

A retenir

- ♦ PERT reste le meilleur outil de réflexion préalable pour mener un projet
- ♦ PERT se dessine de façon très simple et permet le travail en équipe
- ♦ Grâce aux dates calculés, chemin critique, probabilité de terminaison, PERT permet d'avoir une idée globale de la faisabilité du projet et de modifier les logiques au fil de la réflexion
- ♦ Quand on a une bonne idée du planning, on peut donc utiliser un logiciel de gestion de projet qui permettra la représentation sous forme de PERT réseau et Gantt

Chapitre 4

Pilotage d'un projet

Concept de Pilotage

- ♦ Le pilotage consiste à modifier le train du processus projet de façon à maintenir la possibilité d'aboutir aux résultats désirées
- ♦ Le pilotage est un:
 - Ensemble des processus permettant de maîtriser et de guider l'évolution d'un système
 - Ses deux concepts principaux sont le contrôle et la régulation

Gestion des ressources humaines

- ♦ Un projet se réalise avec une équipe. Le management des ressources humaines (**organisation**, **communication**, **motivation**) est un facteur clé pour la réussite ou d'échec d'un projet
- ♦ L'un des rôles du chef de projet est de manager l'équipe ce qui est certainement le rôle de plus difficile (**chef d'orchestre**)
- ♦ Le choix et la composition de l'équipe chargée du développement du projet sont des aspects cruciaux (**leadership**, **expertise**)

Équipe du projet

- ♦ L'équipe doit comprendre des personnes compétentes sur le plan de la maîtrise des technologies et sur le plan de l'expérience: **base solide**
- ♦ L'effectif de l'équipe doit être raisonnable (fonction de la taille du projet) afin de couvrir les délais prévus et pour un management efficace de la part du chef de projet
- ♦ La **stabilité de l'équipe projet** est aussi un facteur très important. Rien n'est plus déstabilisant qu'un turn over du personnel (en général au moment critique !). Il faut donc s'assurer que l'équipe ne bougera pas (c'est malheureusement généralement pas le cas !) jusqu'à la fin du projet et donc limiter les risques: **Récompenser, motiver et valoriser**

Organisation du projet

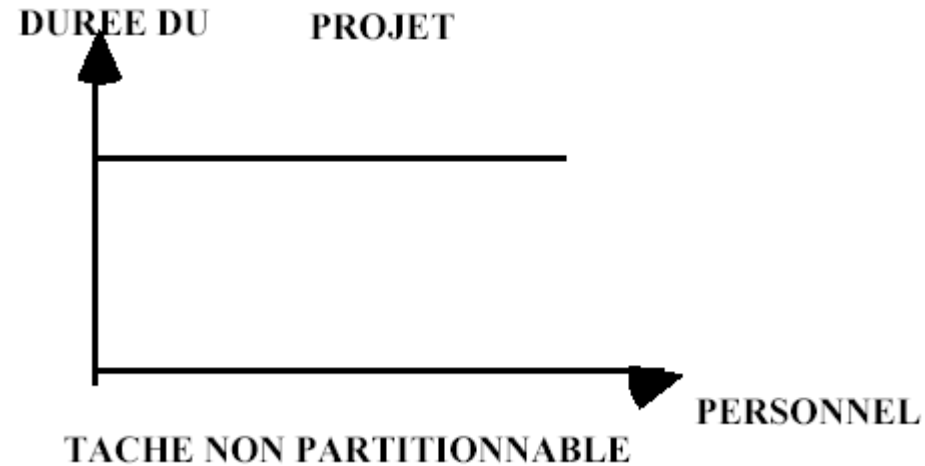
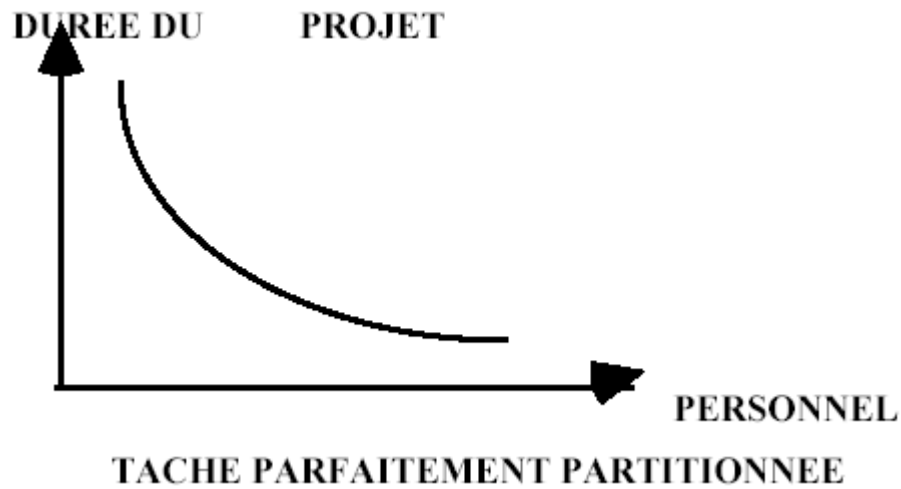
- ♦ Préalablement à son lancement, le chef de projet se doit de "penser" son organisation. Celle-ci doit porter sur plusieurs aspects, notamment:
 - **la répartition du travail**
 - **le système d'information du projet**
 - **les structures de consultation ou de décision**
 - **la définition des "livrables"**
 - **le plan d'assurance qualité**

Répartition du travail

- ♦ Le **découpage** d'un projet en phases, étapes, modules, implique une répartition des membres de l'équipe projet entre ces différents composantes
- ♦ Cette répartition doit bien évidemment prendre en compte les **compétences** et **adéquation** de chacun des **membres** de l'équipe projet et éviter certains points de **blocage** (non souhait, confier une tâche critique à une seule personne!)
- ♦ L'inconvénient évident de la division du travail est le risque de **manque de cohérence**. Une coordination doit donc accompagner la répartition et la réalisation des tâches:
 - Cette coordination peut se manifester de **relationnelle** (concepteurs-développeurs, chef projet – membres de l'équipe)
 - La coordination peut aussi exister sous une forme **codifiée** (résultats, processus: adoption de méthodes et modèles communs ...etc.)

Éléments de réflexion pour le partage de travail

- ♦ Les diagrammes suivants donnent une idée de la durée du projet en fonction du nombre de personnes affectées dans 3 cas différentes:



Profils des membres d'une équipe

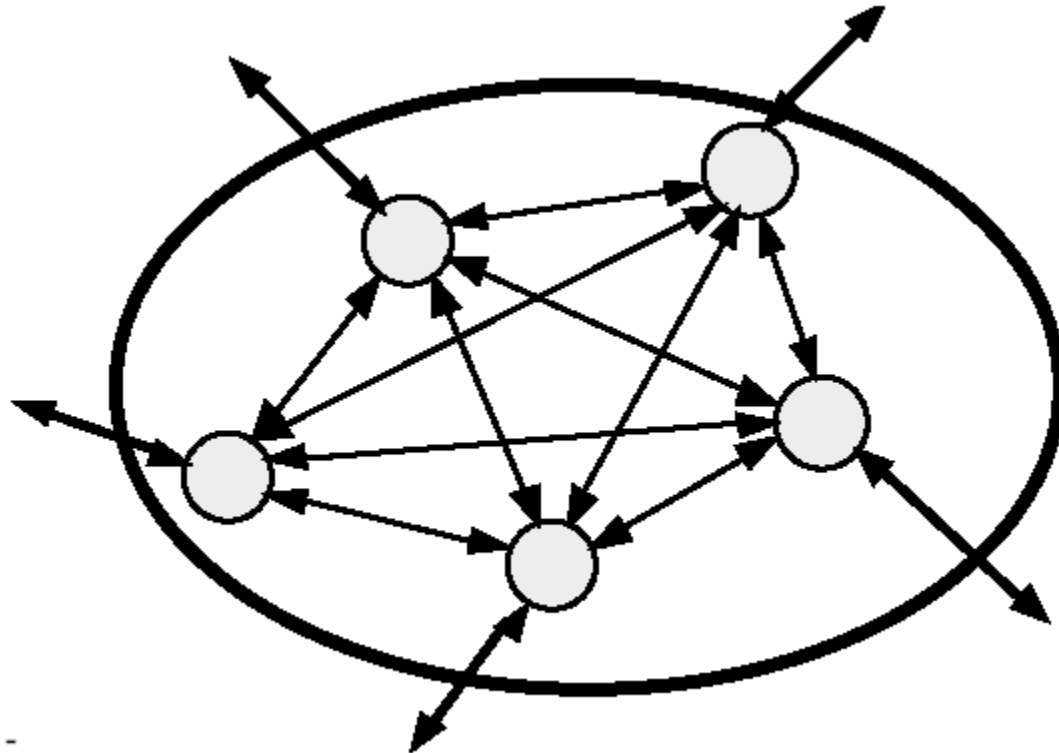
- ♦ Ces profils doivent être choisis en fonction de l'intervention dans les différentes phases du cycle de vie. Nous répertorions les qualités requises suivant les phases:
 - **Définition et analyse des besoins**
Savoir anticiper; Savoir analyser les stratégies
 - **Spécifications fonctionnelles**
Clarté, précision, cohérence, rigueur
 - **Conception**
Cette phase demande des communications intenses; Capacités à formaliser et à abstraire
 - **Programmation et tests unitaires**
Discipline de programmation (style), rigueur, communication, sens du groupe
 - **Intégration**
C'est une phase où la compétence requise est bien souvent celle d'un ingénieur système.

Types d'organisation

♦ Les extrêmes:

PETIT GROUPE DE TRAVAIL SANS AUTORITE DEFINIE : EGOLESS PROGRAMMING

Le travail s'effectue par consensus; le travail de chacun devient le travail de tous. L'équipe s'enrichit par le travail quotidien en commun, les revues et inspections de code. On élimine l'attachement à *son* programme, ses idées.



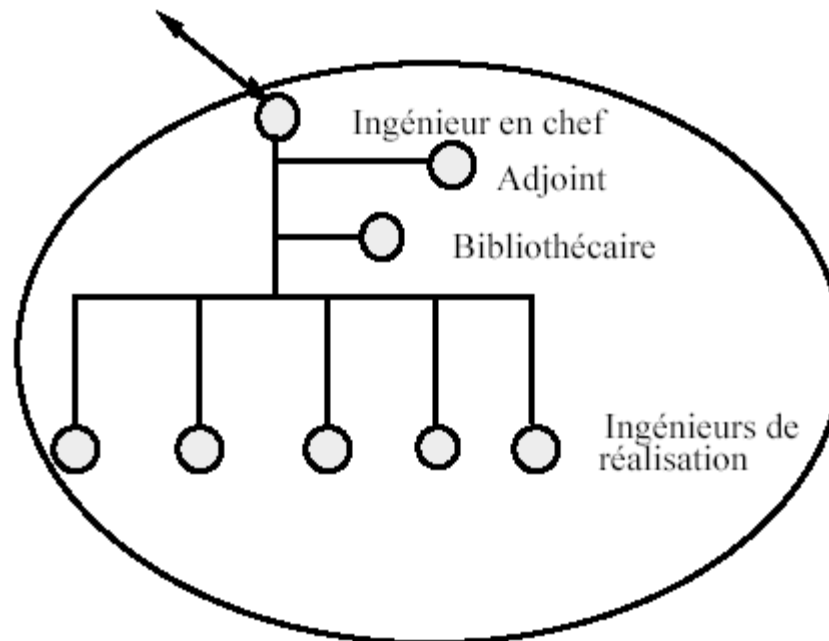
Types d'organisation

♦ L'ORGANISATION "CHIEF PROGRAMMER TEAM"

Il s'agit d'une structuration de l'egoless programming, faisant suite à un projet réussi publié par IBM en 1970.

Un ingénieur en chef ou chef de projet ou senior programmeur dirige l'équipe composée de 2 à 5 personnes.

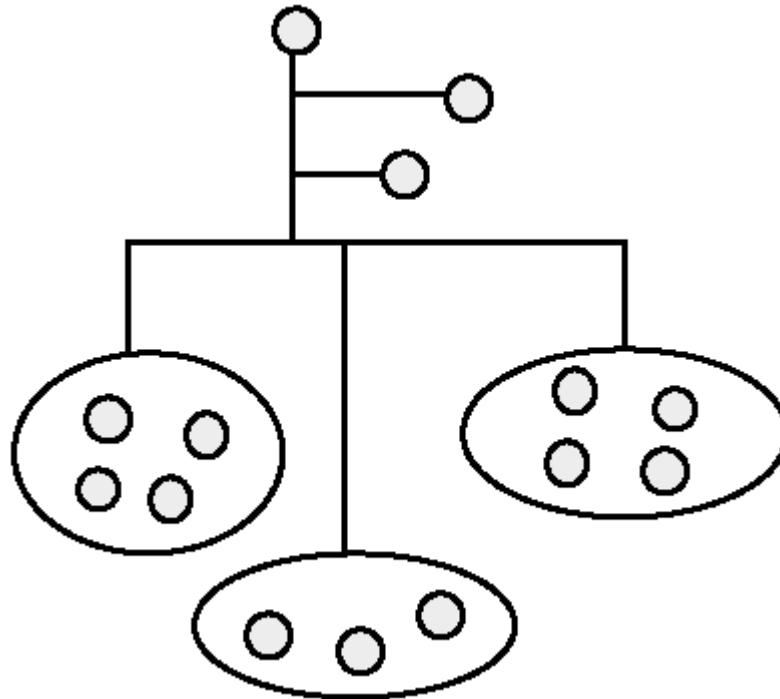
Il coordonne, planifie et vérifie le travail effectué. L'adjoint seconde l'ingénieur en chef et peut à tout instant le remplacer.



Types d'organisation

- ♦ Les structures intermédiaires:

CHEF DE PROJET POUR PLUSIEURS EQUIPES



Taille équipe et facteurs humains

- ♦ Une équipe projet est constitué généralement de 2 à 10 personnes (**nombre magique 7 +/- 2**)
- ♦ Outre l'organisation du travail d'équipe il faut veiller:
 - Aux motivations individuelles et à la motivation collective de l'équipe.
 - Aux relations entre les membres et avec l'extérieur.
 - A la dynamique du chef de projet.
 - Aux écueils à éviter :

Sur-spécialisation;	Trop de niveaux;
Pas assez de niveaux;	Déresponsabilisation
 - A la formation et en particulier au temps d'apprentissage du domaine.
- ♦ Les grandes variations liées aux facteurs humains peuvent être *réduites* par
 - Méthodologies de développement.
 - Standards, normes, ...
 - "Confort" : matériel, logiciel

Mais en aucun cas ces variations ne peuvent être annulées

Système d'information du projet

- ♦ Le système d'information du projet est un **référentiel** pour tous les membres de l'équipe projet sur les éléments caractéristiques du projet (systèmes de documentation et d'échange d'information, Requête de changement...etc.)
- ♦ Dans la plupart des cas, le SI projet sera constitué:
 - d'une **partie "actualités"** précisant l'objectif, le découpage, le planning, les acteurs, (avec mise à jour permanente)
 - d'une **partie "dictionnaires"** répertoriant tous les objets constituant du projet de manière à ce que le vocabulaire soit commun à tous les membres de l'équipe projet (nom du projet, nom des phases et procédures, responsables d'approbation...etc.)

Structure de consultation et de décision

- ♦ En dehors du **comité directeur du projet**, l'instance importante est le **comité de pilotage** qui rassemble des représentants de la maîtrise d'ouvrage et de la maîtrise d'oeuvre.
- ♦ Le comité de pilotage aura pour rôle d'effectuer un suivi du projet sous tous ses aspects. Il devra se réunir périodiquement et prendre des **décisions d'ordre opérationnel**, souvent suggérées par le chef de projet (grâce au tableau de bord).

Structure de consultation et de décision

- ♦ **Le chef de projet** est, quant à lui, l'élément pivot d'un projet. Son rôle est très important (et pas facile !). Il se décline en les tâches suivantes:
 - fixer les objectifs, quelquefois la stratégie, définir les horaires, les étapes, les moyens, les délais et les coûts
 - coordonner les actions et les travaux, suivi du plan projet
 - maîtriser, c'est-à-dire **modifier** les horaires, les étapes et les ressources **si l'objectif évolue** (et il évoluera !) et modifier les étapes suivantes en conséquence
 - optimiser la répartition des ressources en vue d'arriver à une solution optimale au moindre coût

Autres acteurs principaux d'un projet

- ♦ **Le responsable qualité:** Il est responsable de la mise en oeuvre du manuel qualité sur un projet donné. Il rédige et fait appliquer le plan qualité du projet.
- ♦ **Le responsable des ressources matérielles:** Il assure la disponibilité du matériel conformément à la planification.
- ♦ **Le responsable de l'intégration:** Il est responsable de la mise en oeuvre du plan d'intégration
- ♦ **Le responsable de la qualification:** Il est responsable de la mise en oeuvre du plan de qualification

Autres acteurs principaux d'un projet

- ♦ **Le responsable des performances:** Il est responsable des tests de performances conformément au cahier des charges et au plan qualité. Il prend les mesures nécessaires pour atteindre les objectifs de performance définis dans le cahier des charges (simulation, prototypes)
- ♦ **Le responsable de la maintenance:** C'est le responsable de l'équipe qui prendra en charge la maintenance du produit (en général différente de l'équipe de développement).
- ♦ **Le responsable de la documentation:** Il est responsable de la documentation du projet. Ceci ne veut pas dire qu'il rédige toute la documentation associée au projet, mais plutôt qu'il s'assure de sa rédaction. Il définit les normes de présentation des documents en accord avec le manuel qualité de l'entreprise. Il s'assure de la mise à jour de la documentation

La documentation

- ♦ Le **système documentaire** est une composante du système d'information du projet. Il constitue une référence commune à tous les acteurs du projet et, par conséquent, concourt à la coordination générale.
- ♦ Quelques règles générales:
 - Concevoir, après réflexion et discussions, l'architecture du système documentaire, en particulier le **système de classement des documents** ce qui implique une stratégie d'indexation (recherche facile)
 - Proposer, puis imposer une **structure type de chaque sorte de document**. En particulier, pour chaque document, doivent figurer son titre, type, date de création ou de mise à jour, son ou ses auteurs, version, statut...
 - Faire **signer chaque document** par ses auteurs; c'est une opération de responsabilisation des collaborateurs
 - **Bonne rédaction**, lisible et compréhensible. Utiliser un vocabulaire commun pour éviter les ambiguïtés (un glossaire serait le bienvenu).

La documentation

- ♦ Les familles de documents liés à **une phase du cycle de vie**:
 - **le cahier des charges détaillé**: présentation détaillée du projet, description détaillée des fonctionnalités du produit à réaliser, description des schémas d'utilisation du produit, liste des données concernées par le produit
 - **le cahier des spécifications techniques**: architecture logique du produit, spécifications logicielles, description technique des différents composants, moyens matériels et logiciels mis en oeuvre,...
 - **le plan d'intégration**
 - **la procédure de validation**: test unitaires, régression

La documentation

♦ Documents **transversaux** à la vie du projet:

- le glossaire, la liste des abréviations
- le plan du projet
- le plan qualité

♦ Documents **remis au client**:

- le manuel utilisateur
- le manuel d'installation
- le manuel maintenance

Le suivi du projet

- ♦ Un projet étant organisé et planifié, il faut maintenant en **effectuer le suivi** afin de s'assurer de sa réalisation et de prendre des décisions au bon moment
- ♦ Divers **facteurs nécessitent souvent un réajustement**: besoins mal identifiés à l'origine, non validation d'étapes, sous-estimation des charges, changement de l'environnement, **aléas divers**
- ♦ D'une manière générale, le suivi d'un projet s'effectue par tenue et examen d'un **tableau de bord** qui peut comporter divers composants, notamment:
 - un **suivi individuel** des ressources humaines affectées au projet
 - un **suivi global** du projet
- ♦ Enfin de projet, un récapitulatif global permet de **capitaliser** des connaissances sur le projet, **points de succès** et **points de défaillances** (elles pourront servir de guide pour l'estimation et réalisation de projets futurs).

Suivi individuel

- ♦ Le **suivi individuel** récapitule les tâches affectées à chaque personne. Pour chaque tâche, les paramètres suivants sont consignés:
 - la charge initiale nécessairement estimée
 - la charge affectée: personnalisation de la charge initiale (dépend de la personne, en particulier de son expérience et de ses performances)
 - la charge actualisée (réajustement continu au cours du temps)
 - l'activité périodique : **temps T passé sur la tâche** pendant la période (consommation de ressource humaine) et **"reste à faire" R** (valeur de la différence : (charge affectée ou actualisée - temps passé) ou valeur estimée de cette différence).

Suivi individuel

période : mois de juin 2001, semaine 1	tâches	charge initiale	charge affectée	charge actualisée	temps passé T	reste à faire R
Ali Taher	programmation du module traitement congé annuel	24	24	23	3 2	20
Aya Kalal	tests de l'interface de saisie activité syndicale	4	4	4	4 1	0
Said Nil	programmation du module sorties imprimées maladie	12	12	11	2 3	9

Suivi individuel

- ♦ Il peut être intéressant d'avoir des indicateurs se rapportant à l'**avancement** du projet, à la **productivité** et à la **performance** des personnels. On définit alors les paramètres suivants :

- **avancement en fin de période n:** $A(n) = R(n-1) - R(n)$

- **productivité de la période n:** $Prod(n) = A(n) * 100 / T(n)$;
ce paramètre mesure la vitesse d'exécution ou vitesse d'avancement.

- **performance :**
 $Perf = (Charge\ affectée * 100) / (temps\ total\ passé + reste\ à\ faire)$;
ce paramètre mesure le degré d'atteinte des objectifs.

- ♦ Ces paramètres peuvent apparaître dans deux documents : le récapitulatif individuel mensuel et le bilan individuel mensuel.

Récapitulatif individuel mensuel

moi de juin 2001	tâches	charges affectées	semaine 1			semaine 2			semaine 3			semaine 4			total mois		
			T	R	A	T	R	A	T	R	A	T	R	A	T	R	A
Ali Taher	prog. module traitement	23	3	20	3	4	14	6	3	11	3	2	9	2	12	9	14
	prog. module Web	16							2	14	2	3	11	3	5	11	5
	tests module interface	4				1	3	1							1	3	1
total		43													18	23	20

T: temps passée / R: reste à faire / A: avancement

Bilan individuel mensuel

mois de juin 2001	tâches	charges affectées	T(2)	R(2)	T(3)	R(3)	A(3)	Prod(3)	Récapitulatif depuis le début du projet	
Ali Taher									temps total passé	Perf
	prog. module Compta.	23	14	9	7	1	8	114,29	21	104,54
	prog. module Web	16	5	11	10	1	10	100,00	15	100,00
	tests module saisie	4	1	3	3	0	3	100,00	4	100,00
	Total	43	20	23	20	2	21	105,00	40	102,38

Suivi global

- ♦ Le suivi du projet reprend la comptabilisation des travaux effectués par les différents membres de l'équipe. A périodes régulières, on peut ainsi évaluer:

- **l'évolution de la charge restante = $T(n) - A(n)$**

Donne la tendance du passé récent entre le mois $n-1$ et le mois n

- **l'évolution globale de la charge** définie par:

evol = temps total passé + reste à faire - charge initiale estimée

et son équivalent en pourcentage:

%evol = evol * 100 / charge initiale estimée

- **l'avancement :**

%avan = (charge initiale estimée - reste à faire)*100/charge initiale estimée

- ♦ D'autres paramètres économiques peuvent aussi être estimés:
 - **CBTP** : Coût Budgété du Travail Prévu,
budget initial basé sur l'estimation des charges et des ressources
 - **CRTE** : Coût Réel du Travail effectué,
Coût réel à la date t des travaux réalisés
 - **CBTE** : Coût Budgété du Travail Effectué.
Coût des travaux réalisés valorisés au coût standard utilisé pour le CBTP

Tableau d'avancement du projet

Tâches	mois n-1		mois n			Évolution charge restante	Charge initiale	Temps total passé	Évolution globale	%avancement
	T	R	T	R	A					
A										
B										
C										

Capitalisation

- ♦ Il est évidemment intéressant, en fin de projet, de procéder à une capitalisation du savoir-faire. On dresse alors **une fiche de bilan du projet** dont la physionomie ressemble à la figure suivante:

tâches	charge initiale estimée	charge constatée	nombre de jours d'écart	%écart par rapport à la charge estimée
A	25	50	25	100 %
B	20	10	-10	-100 %
C	30	30	0	0 %

Chapitre 5

Qualité et gestion des risques

Système d'information de qualité

- ♦ **Orienté Business**: répond parfaitement aux besoins des clients et utilisateurs
- ♦ **Opérationnel**: stable et convivial de point de vu administration
- ♦ **Intègre**: non corrompu
- ♦ **Performant**: temps de réponse raisonnable
- ♦ **Extensible**: permet une augmentation du volume des données
- ♦ **Maintenable**: code lisible et architecture flexible

Définition du risque

- ♦ **Dictionnaire**: Danger éventuel plus ou moins prévisible
- ♦ **Assurance**: Éventualité d'un événement ne dépendant pas exclusivement de la volonté des parties et pouvant causer la perte d'un objet ou tout autre dommage
- ♦ **ISO13335**: Conséquences potentielles d'une menace exploitant une vulnérabilité d'un bien ou d'un groupe de bien
- ♦ Possibilité qu'un projet ne s'exécute pas conformément aux prévisions: Date, Coût, Spécifications. Ces écarts / prévisions étant considérés comme inacceptables
- ♦ Ne pas confondre 'risque' et 'problème' dans le contexte d'un projet

Statistiques projets

♦ Succès	16,2%
♦ Retard ou budget dépassé	52,7%
♦ Abandon	31,1%

Source : *Standish Group, étude de 8.300 projets*

Pourquoi la gestion des risques

- ♦ Un projet ayant par nature un caractère **novateur** présente évidemment des risques, plus aléas projets S.I
- ♦ Taille des projets de plus en plus grande → structure complexe
- ♦ Complexité des détails et de l'intégration (multidisciplinaire)
- ♦ Projet « time to market »

Tout ceci complexifié par la diminution des budgets : « Design to cost »

Quelques communalités

- ♦ Un projet mal planifié prendra trois fois plus de temps à réaliser que prévu, alors qu'un projet bien planifié ne prendra que deux fois plus de temps
- ♦ S'il est quasiment impossible que quelque chose marche mal, ça marchera mal quand même (Loi de Murphy)
- ♦ Les groupes de projet détestent les comptes-rendus hebdomadaires (ou mensuels) d'avancement du projet, car ceux-ci mettent trop en lumière l'absence de progrès
- ♦ Tous les projets progressent rapidement jusqu'à atteindre 90 % de leur achèvement, puis ils restent inachevés à 90 % pour toujours

Cause d'échecs des projets informatiques

- ♦ Partage des rôles entre les intervenants imprécis, fonctions non identifiées, taux de participation trop floue
- ♦ Absence d'études de faisabilité / d'orientation
- ♦ Spécifications vagues, incomplètes, changeantes
- ♦ Mauvaise estimation des charges
- ♦ Interlocuteurs concernés non impliqués. Manque de suivi régulier
- ♦ Manque d'expérience du chef de projet

Plan gestion des risques

- ♦ Trois attitudes de base possibles face à un risque
 - Élimination du risque
 - Atténuation du risque
 - Acceptation du risque

- ♦ Quelques techniques utilisées pour constituer un plan de gestion des risques
 - Mitiger le risque (plan d'atténuation ou d'élimination)
 - Plans alternatifs
 - Assurance (Navettes spatiales, assurances vies)
 - Concept de 'réserve' (budget normal + budget risque)

Évaluer les risques

- ♦ On ne doit pas attacher la même importance à tous les risques potentiels dans un projet
- ♦ L'évaluation des risques aide à prioriser l'attention à accorder aux différents risques auxquels est confronté un projet
- ♦ Deux concepts importants
 - La probabilité que l'évènement survienne
 - Le dommage potentiel si l'évènement survient
- ♦ Exprimer les types de risques les plus craints sous la forme de scénarios de faute. Pour chaque scénario, estimer le coût

Défaillances dans les projets informatiques

- ♦ Le déroulement d'un projet peut être perturbé par différents aléas :
 - **Aléas fonctionnels**: évolution de la réglementation externe, changement imprévu des règles de gestion interne, etc.
 - **Aléas techniques** : défaillance des équipements, non disponibilité des outils ou du matériel informatique.
 - **Aléas organisationnels**: non disponibilité d'informaticiens ou de quelques membres de l'équipe, locaux non disponibles pour le développement du projet.

Erreurs humaines

- ♦ Mauvaise perception
- ♦ Non respect des procédures
- ♦ Erreurs de communication
- ♦ Erreurs décisionnelles

Organisations défaillantes

- ♦ Moyens insuffisants
- ♦ Instances de pilotage ou de gestion ou de résolution de problèmes inexistantes ou inadaptées
- ♦ Circuits administratifs lents
- ♦ Absence d'audit ou de suivi

Risques hors projet

- ♦ Accident
- ♦ Malveillance (perte de matériel ...etc)
- ♦ Mouvements sociaux (grève)
- ♦ Catastrophe naturelle
- ♦ Risque stratégique (produit obsolète)
- ♦ Risque juridique (utilisation de logiciels sans licence)

Démarche risque

- ♦ Démarche qui regroupe l'ensemble des méthodes mises en œuvre pour identifier, estimer et réduire les risques du projet
- ♦ La criticité se calcule de la manière suivante :
 - Pr : Probabilité ou fréquence d'apparition ou occurrences
 - G : Gravité ou sévérité des conséquences éventuelles
 - Nd: Probabilité de non-détection

$$C = G * Pr$$

- ♦ Indice de criticité :

$$RPN = G * Pr * Nd$$

RPN : Risk Priority Number

Permet de savoir ce qui le moins et/ou le plus critique

La « priorisation » du risque

Probabilité	5 Élevée					
	4					
	3					
	2					
	1 Faible					
		1 Mineur	2	3	4	5 Majeur
	Impacts des dommages					

Paramètres risques dans un projet SI

- ♦ Taille du projet
- ♦ Difficulté technique
- ♦ Degré d'intégration
- ♦ Configuration organisationnelle
- ♦ Changement
- ♦ Instabilité de l'équipe projet

Projet à faible risque

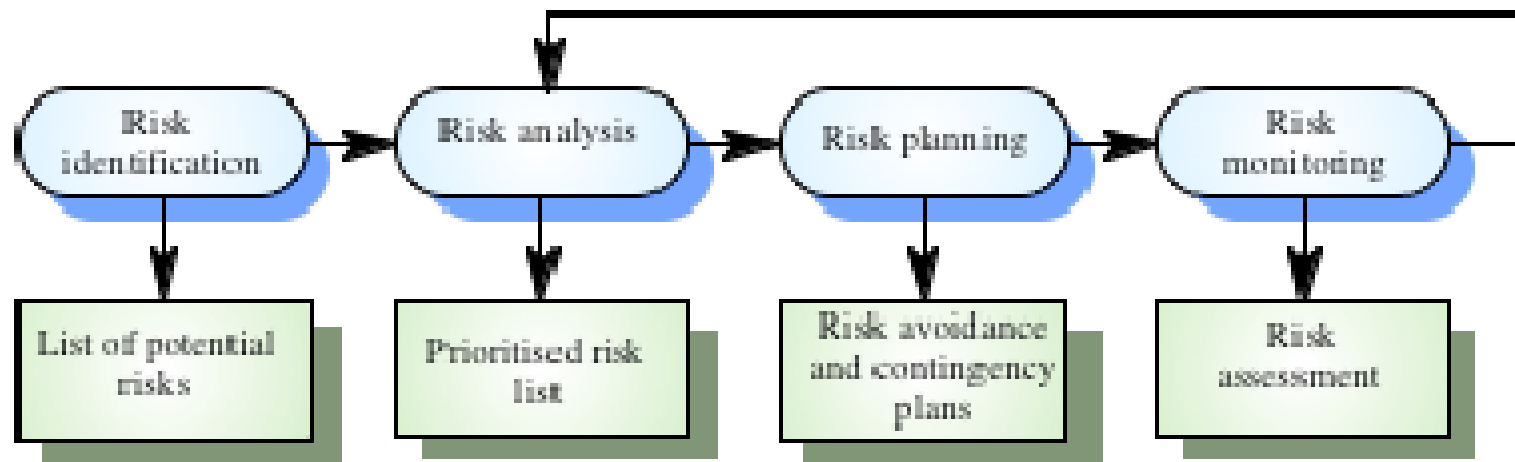
paramètres	degré du risque					
	0	1	2	3	4	5
taille du projet		●				
difficulté technique	●					
degré d'intégration		●				
configuration organisationnelle	●					
changement		●				
instabilité de l'équipe projet	●					

Projet à grand risque

paramètres	degré du risque					
	0	1	2	3	4	5
taille du projet					●	
difficulté technique						●
degré d'intégration						●
configuration organisationnelle					●	
changement				●		
instabilité de l'équipe projet						●

Processus gestion du risque: les activités

Évolution du risque durant le projet



Processus gestion du risque

- ♦ Percevoir et identifier le risque
- ♦ Mesurer la gravité de chaque risque (dégâts ...)
- ♦ Qualifier sa potentialité
- ♦ Définir le capteur
- ♦ Calculer la criticité
- ♦ Hiérarchiser les risques du projet (fonctionnel, organisationnel ...etc.)
- ♦ Développer le [plan détaillé de gestion des risques](#)

Les moyens

♦ Gestion de projet

- Pilotage plus serré
- Évaluation et planification plus détaillée
- Suivi plus précis de l'avancement (profond et fréquent)
- Communication plus fréquentes

♦ Production

- Produit intermédiaires (Document, prototypes,..)

♦ Qualification des produits

- Approfondir les activités de qualification

Les moyens

♦ Contrôle qualité des processus

- Contrôle du respect des normes en interne
- Audit par une tierce partie

♦ Gestion des ressources humaines

- Contrôle des expertises des membres de l'équipe
- suivi, formation, coaching

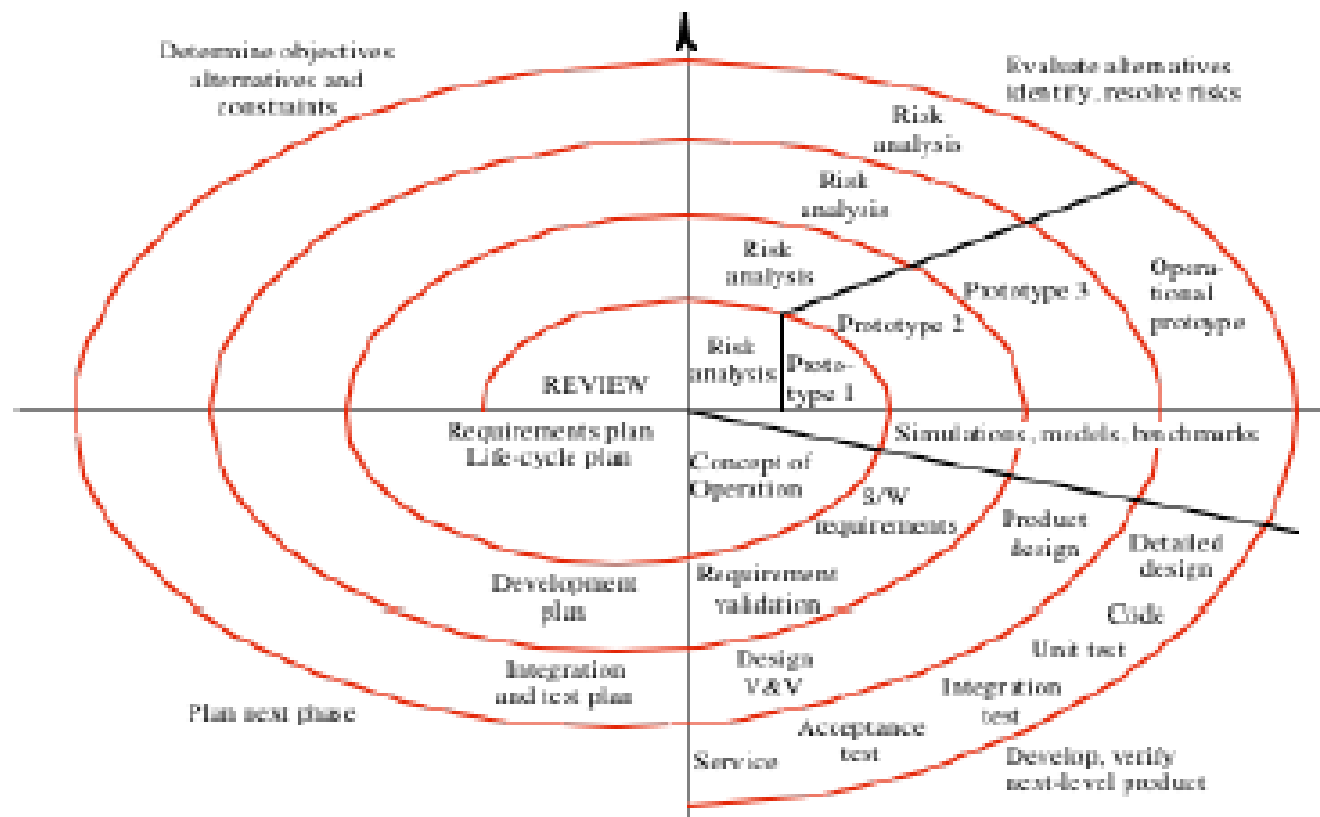
♦ Logistique du projet

- Améliorer la logistique:
matériels, logiciels, dispositifs de communication, etc..

Processus gestion des risques

♦ Le modèle spiral

- Dans ce modèle, le risque est un facteur qui est pris explicitement en considération
- Analysez le risque au début de chaque nouvelle phase



Exemple des risques dans un projet logiciel

Risque	Type de risque	Description
Staff turnover	Projet	Experienced staff will leave the project before it is finished.
Management change	Projet	There will be a change of organisational management with different priorities.
Hardware unavailability	Projet	Hardware which is essential for the project will not be delivered on schedule.
Requirements change	Projet et produit	There will be a larger number of changes to the requirements than anticipated.
Specification delays	Projet et produit	Specifications of essential interfaces are not available on schedule
Size underestimate	Projet et produit	The size of the system has been underestimated.
CASE tool under-performance	Produit	CASE tools which support the project do not perform as anticipated
Technology change	Entreprise	The underlying technology on which the system is built is superseded by new technology.
Product competition	Entreprise	A competitive product is marketed before the system is completed.