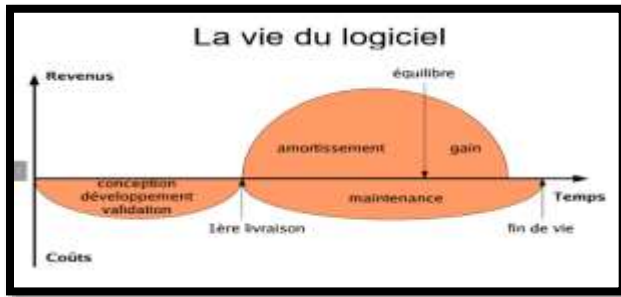


## Chapitre 1 : Qu'est-ce que c'est le génie logiciel et sa place dans les sujets d'informatique

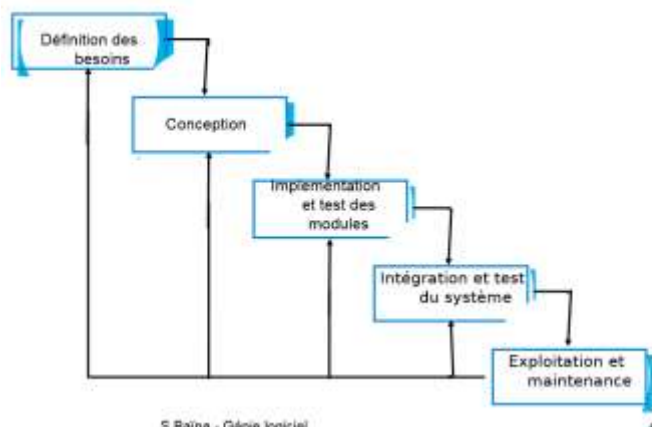
### A- Définitions :

<b>Génie logiciel</b>	Art de bien faire de bons programmes
<b>Logiciel</b>	<ul style="list-style-type: none"> <li>Programmes et la documentation associée – cahier de charges, modèles, manuels</li> </ul>
<b>Types Logiciel</b>	<ul style="list-style-type: none"> <li>Générique – Individuel – Hérité</li> </ul>
<b>projet logiciel Succès</b>	livré à temps, à l'intérieur des budgets et des spécifications originales
<b>projet logiciel mitigé</b>	livré et opérationnel mais avec moins de fonctions que prévu en dépassant le budget et/ou les échéanciers
<b>projet logiciel Echec</b>	abandonné en cours de route, ou résultat livré mais jamais utilisé
<b>Processus du logiciel</b>	Un ensemble d'activités dont l'objectif est le développement et l'évolution du logiciel.
<b>activités</b>	1. Spécification – ce que le logiciel doit faire et les contraintes posées au développement  2– Développement - production logiciel  3– Validation – vérification si le logiciel est celui qui est attendu du client. 4– Évolution – modification du logiciel en accordance avec les besoins.
<b>modèle</b>	une présentation simplifiée d'un point de vue différent
<b>Points de vue</b>	: – Flux d'activités – Flux des données – Rôles/activités
<b>Modèles génériques –</b>	Cascade (Waterfall) – Itérative – Composants
	 <p>Le diagramme intitulé "La vie du logiciel" illustre le cycle de vie d'un logiciel sur un graphique à deux axes : l'axe vertical représente les "Revenus" (positif) et les "Coûts" (négatif), et l'axe horizontal représente le "Temps". La courbe des coûts est négative et se termine à l'origine. La courbe des revenus est positive, commence à l'origine, passe par un point d'équilibre, atteint un pic de "gain", puis décroît vers la "fin de vie". Les phases du cycle sont indiquées : "conception", "développement", "validation" (sous les coûts), et "1ère livraison", "amortissement", "gain", "maintenance" (sous les revenus).</p>

<b>Méthodes de génie logiciel</b>	
<b>Composants des méthodes</b>	<ul style="list-style-type: none"> <li>– Modèles : graphiques (objets, flux des données, machine d'états et c.)</li> <li>– Règles : contraintes</li> <li>– Recommandations : bonne pratique</li> <li>– Direction et gestion : la séquence des activités</li> </ul>
<b>CASE</b>	(pour <i>computer-aided software engineering</i> , en français <i>génie logiciel assisté par ordinateur</i> ) désigne les environnements de <b>développement graphiques</b> qui facilitent la création rapide de <b>logiciels</b>
<b>Upper CASE</b>	Support les activités de conception et de définition des besoins
<b>Lower CASE</b>	Support les activités tardives – programmer, déboguer, tester
<b>Les propriétés du bon logiciel</b>	<ul style="list-style-type: none"> <li>• Avoir la fonctionnalité désirée.</li> <li>• Facilement maintenable</li> <li>• Sûr – on doit avoir confiance en lui</li> <li>• Efficace – de ne pas gaspiller les ressources du système</li> <li>• Accepté, compris par les usagers</li> </ul>
<b>Les défis devant le GL</b>	<ul style="list-style-type: none"> <li>• Hétérogénéité des plateformes</li> <li>• Délivrance (respecter les termes et la qualité à la fois)</li> <li>• Confiance des usagers</li> <li>• Responsabilités professionnelles et éthiques</li> </ul>
<b>Fiabilité</b>	
<b>Causes de non fiabilité</b>	<ul style="list-style-type: none"> <li>• Panne du matériel</li> <li>• Échec du logiciel</li> <li>• Erreur opérationnelle – le plus souvent</li> </ul>
<b>Origines de non fiabilité 'quelle étape ?'</b>	Spécification ***** Conception ***** Codage ** Autres ***
<b>Composants de la fiabilité (Dependability)</b>	<ul style="list-style-type: none"> <li>– Disponibilité</li> <li>– Fiabilité (Reliability)– l'habilité de assurer les services comme ils sont spécifiées</li> <li>– Sécurité               <ul style="list-style-type: none"> <li>• (Safety)De fonctionner sans échec catastrophique</li> <li>• (Security)De se protéger des attaques externes</li> </ul> </li> <li>– Habilité de restauration après un échec</li> <li>– Habilité d'être maintenu – à quel degré il s'adapter vers de nouvelles exigences</li> <li>– Habilité de survivre – de quel degré il continue de fonctionner sous attaque</li> <li>– Tolérance d'erreurs – à quel degré il tolère les erreur de l'utilisateur.</li> </ul>

## Chapitre 2 : Le processus du logiciel : Quel est le cycle de vie d'un logiciel et ses modèles?

<b>Activités</b>	<p>1. Spécification – ce que le logiciel doit faire et les contraintes posées au développement</p> <p>2– Développement - production logiciel</p> <p>3– Validation – vérification si le logiciel est celui qui est attendu du client.</p> <p>4– Évolution – modification du logiciel en accordance avec les besoins.</p>
<b>Modèles génériques –</b>	<p>Cascade (Waterfall) – Itérative – en spirale – Extreme Programming (XP) – Basé à l'assemblage de composants</p>

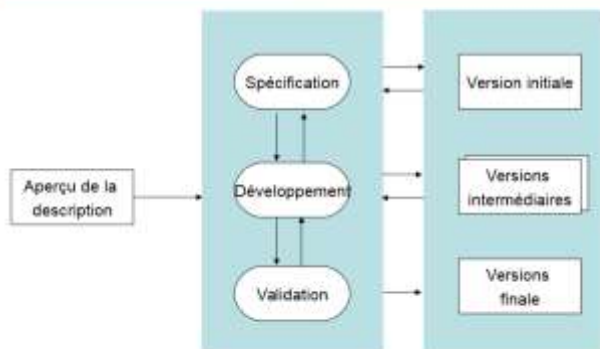


- **Problèmes**
  - Il est difficile de séparer les étapes
  - On peut l'utiliser quand les besoins sont bien définis et ils sont stables.
- **Avantages**
  - Bien documenté à chaque phase
- **Désavantages**
  - Rigide (on ne peut pas de répondre au besoins nouveaux ou modifiés des clients)

S. Raina - Génie logiciel

5

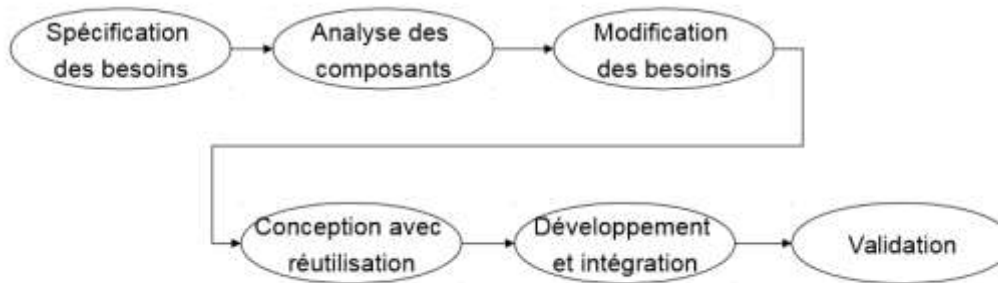
### Processus évolutif



### Processus évolutif

- **Problèmes**
  - Manque de visibilité
  - Mauvaise structure
  - Exige des qualités spéciales des programmeurs
- **Application**
  - Systèmes de petite et moyenne taille
  - Parties de grands systèmes
  - Systèmes de courte vie.

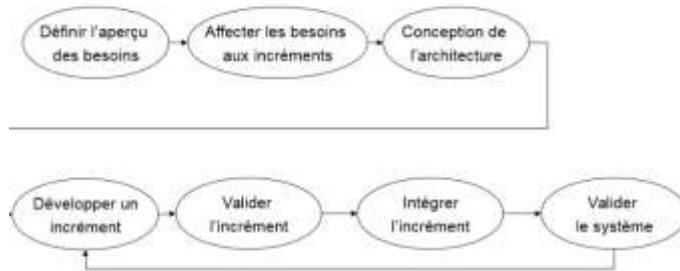
# Développement par composants



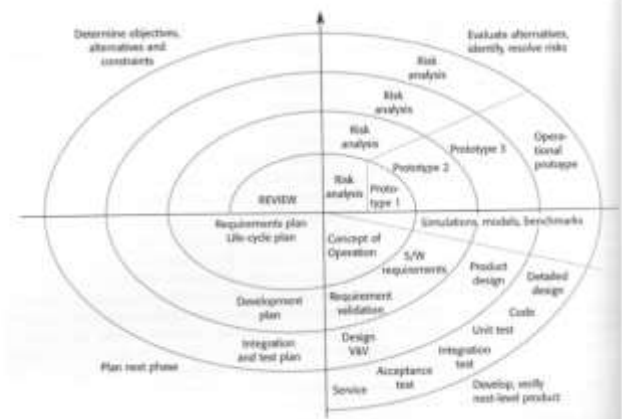
Processus itérative: • Approches

– Livrer par incréments – Développement spirale

## Livrer par incréments



## Développement spirale



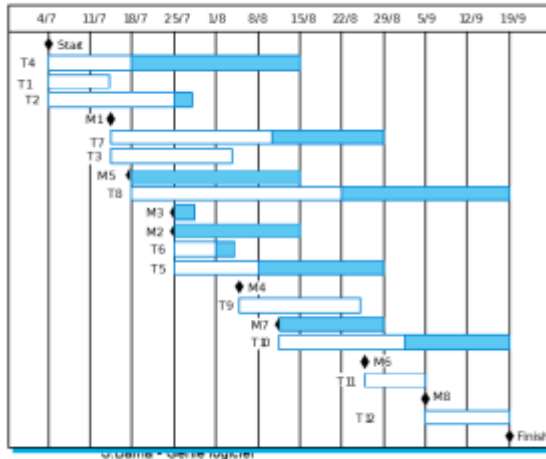
## Développement spirale

### •travail par Secteurs

- Préciser les objectifs
  - définir des objectifs principaux
  - Identifier des contraintes sur le processus et le produit
  - Identifier des risques principaux
  - Planifier des stratégies
- Définir et minimiser le risque
- Développement et validation
  - Choisir un modèle de développement approprié.
- Planifier l'itération suivante

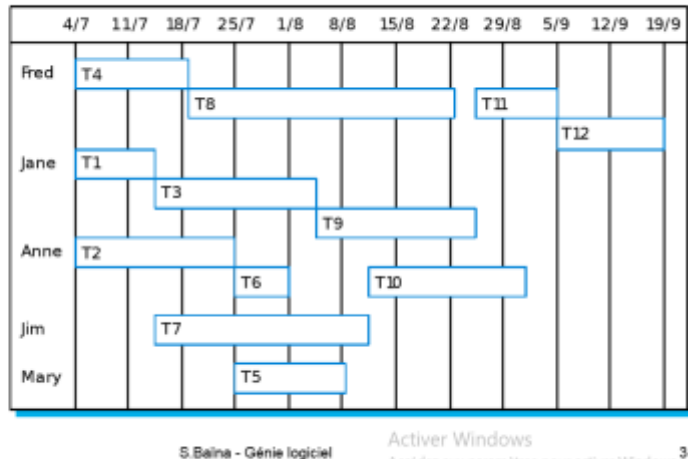
<b>Spécification des besoins</b>	<h3 style="text-align: center;">La Spécification des besoins</h3> <pre> graph TD     EF(Etude de faisabilité) --&gt; RF[Rapport de faisabilité]     EF --&gt; TBA(Trouver les besoins et analyse)     TBA --&gt; MS[Modèles du système]     TBA --&gt; SB[Spécification des besoins]     SB --&gt; VBS[Validation des besoins]     SB --&gt; BUS[Besoins d'utilisateur et du système]     VBS --&gt; DB[Document des besoins Cahier de charge]     MS --&gt; DB     BUS --&gt; DB   </pre>
<b>Conception et implémentation</b>	<p>Processus permettant de convertir la spécification en un exécutable logiciel.</p> <p>1 Conception – la structure du logiciel qui réalise la spécification</p> <p>2 Implémentation – Traduit la structure en un programme exécutable.</p>
<div style="display: flex; justify-content: space-between;"> <div style="width: 48%;"> <h3 style="text-align: center;">Conception</h3> <p style="text-align: center;">Activités de la conception</p> <p style="text-align: center;">Produits du projet</p> </div> <div style="width: 48%;"> <h3 style="text-align: center;">Modèles graphiques</h3> <ul style="list-style-type: none"> <li>• Modèle objet</li> <li>• Modèle des séquences</li> <li>• Modèle de transition d'états</li> <li>• Modèle structural</li> <li>• Modèle des flux de données</li> <li>• Modèle des activités</li> </ul> </div> </div>	
<div style="display: flex; justify-content: space-between;"> <div style="width: 48%;"> <h3 style="text-align: center;">Programmer et déboguer</h3> <ul style="list-style-type: none"> <li>• Programmer, un acte individuel.</li> <li>• Déboguer</li> </ul> <pre> graph LR     LE(Localiser l'erreur) --&gt; CC(Conception la correction)     CC --&gt; C(Correction)     C --&gt; TP(Test programme)   </pre> </div> <div style="width: 48%;"> <h3 style="text-align: center;">Validation</h3> <ul style="list-style-type: none"> <li>• Test des composants</li> <li>• Test du système</li> <li>• Test acceptant</li> </ul> <pre> graph LR     TC(Test composants) --&gt; TS(Test système)     TS --&gt; TA(Test acceptant)     TA --&gt; TS     TS --&gt; TC   </pre> </div> </div>	
<b>Gestion du projet</b>	<p>. Contraintes : – Le temps – Le budget – Le personnel</p>
<b>Activités de la gestion</b>	<ul style="list-style-type: none"> <li>• Écrire une proposition</li> <li>• Planification et emploi de temps</li> <li>• Calculer les coûts</li> <li>• Suivi et révisions</li> <li>• Sélection et évaluation du personnel.</li> <li>• Écrire des présentations et rapports</li> </ul>

## Diagramme des activités



32

## Diagramme de Gant



S.Baine - Génie logiciel

Active Windows  
Analyser une ressource pour un logiciel Microsoft

3

## Gestion des risques

- Qu'est ce que c'est risque?  
La probabilité que quelque circonstances défavorables vont arriver
- Types de risque
  - De projet – affecte l'emploi de temps ou les ressources
  - De produit – affecte la qualité et le comportement du logiciel
  - D'organisation – affecte le l'organisation

## Gestion des risques

- Identification de risque
- Analyse de risque
  - Estimer la probabilité et conséquences;
- Planifier le risque
  - Plans d'éviter et minimiser les effet du risque;
- Suivi du risque