Introduction au génie logiciel

2ème année/Génie Informatique
2021-2022/Semestre 1

❑**Mahamat Atteïb Ibrahim DOUTOUM**
Docteur d'Université, Maître assistant CAMES

Organisation/Modalités

- **Cours**
  - Présentation des concepts
  - **TD/Exposé**
- **Evaluation**
  - Contrôle continu (CC) obligatoire.
    - Participation
    - Sur table
    - Seuls documents autorisés au contrôle :
      - Transparents du cours
      - Fiches résumé
  - Examen : probablement le ,,,,

Organisation/Modalités

- **Méthodologie**
  - Qu'il faut travailler dès le début
  - Qu'il faut être assidu
  - Être concentré pendant les cours.
- Qu'il faut réviser de temps à autres

Organisation/Modalités

- **Volume : ~ 45h**
- Pré requis :
  - Prédisposition d'analyse inéluctable et logique.
  - Cours
    - met l'accent sur le principe de conception de base des systèmes informatiques reposant sur une compréhension technique approfondie de la reproduction des applications informatiques
    - application de l'ingénieure/dévp. des logiciels.

Mots clés

❑Partie I

- ❑Généralités
- ❑Lois d‘évolution du logiciel
- ❑Principes du GL
- ❑processus de développement logiciel
- ❑Cycle de vie du logiciel + Qualité du logiciel

❑Partie II

- ❑Modélisation orientée objet
- ❑Modélisation de processus de développement logiciel

Objectifs

- **Le cours de GL se veut pour objectif primordial**
  - **initier les étudiants, à la conception des app infor**
    - **façon méthodique**
    - **et rééditable;**

en les incitant à rechercher et établir les fonctionnalités d'une application, et à les modéliser sous forme de cas d'utilisation et de scénarios ainsi que rechercher les classes et les acteurs nécessaires à la conception de l'application.

Objectifs

- Vous familiariser avec les futures structures de votre vie professionnelle
- Vous donner les outils/adapter à la situation/vous serez acteurs.
- Développer vos capacités d'analyse de problèmes de conception logiciel.

Partie I

Généralités

- Les **projets** logiciels sont célèbres

- Les **projets** logiciels sont célèbres
  - dépassements de budget,
  - cahiers des charges non respectés.

- Les **projets** logiciels sont célèbres
  - dépassements de budget,
  - cahiers des charges non respectés.
  - De façon générale, le développement de logiciel est une activité complexe, qui est loin de se réduire à la prog.

- Les **projets** logiciels sont célèbres
  - dépassements de budget,
  - cahiers des charges non respectés.
  - De façon générale, le développement de logiciel est une activité complexe, qui est loin de se réduire à la prog.
  - Nécessite d'adopter 1 méthode de développement, qui permet d'assister 1 ou plusieurs étapes du cycle de vie/ logiciel.

Génie logiciel

Motivation 1ère du Génie Logiciel est la réduction des coûts de développement du logiciel.

Génie logiciel

Motivation 1ère du Génie Logiciel est la réduction des coûts de développement du logiciel.

GL: moyens (techniques et méthodologiques) permettant la construction de systèmes informatiques répondant à des critères de qualité préalablement définis.

Génie logiciel

Motivation 1ère du Génie Logiciel est la réduction des coûts de développement du logiciel.

GL: moyens (techniques et méthodologiques) permettant la construction de systèmes informatiques répondant à des critères de qualité préalablement définis.

GL : Méthodologie de construction en équipe d'un logiciel complexe et à multiples versions

Génie logiciel

Motivation 1ère du Génie Logiciel est la réduction des coûts de développement du logiciel.

GL: moyens (techniques et méthodologiques) permettant la construction de systèmes informatiques répondant à des critères de qualité préalablement définis.

GL : Méthodologie de construction en équipe d'un logiciel complexe et à multiples versions

Programmation & Génie Logiciel

- **Programmation : activité personnelle**

Génie logiciel

Motivation 1ère du Génie Logiciel est la réduction des coûts de développement du logiciel.

GL: moyens (techniques et méthodologiques) permettant la construction de systèmes informatiques répondant à des critères de qualité préalablement définis.

GL : Méthodologie de construction en équipe d'un logiciel complexe et à multiples versions

Programmation & Génie Logiciel

- **Programmation : activité personnelle**
- **Génie Logiciel : activité d'équipe**

Génie logiciel

Motivation 1ère du Génie Logiciel est la réduction des coûts de développement du logiciel.

GL: moyens (techniques et méthodologiques) permettant la construction de systèmes informatiques répondant à des critères de qualité préalablement définis.

GL : Méthodologie de construction en équipe d'un logiciel complexe et à multiples versions

Programmation & Génie Logiciel

- **Programmation : activité personnelle**
- **Génie Logiciel : activité d'équipe**
  - ✓ **Côté finance**

Génie logiciel

Motivation 1ère du Génie Logiciel est la réduction des coûts de développement du logiciel.

GL: moyens (techniques et méthodologiques) permettant la construction de systèmes informatiques répondant à des critères de qualité préalablement définis.

GL : Méthodologie de construction en équipe d'un logiciel complexe et à multiples versions

Programmation & Génie Logiciel

- **Programmation : activité personnelle**
- **Génie Logiciel : activité d'équipe**
  - ✓ **Côté finance**
  - ✓ **Coût du logiciel supérieur à celui du matériel**

Génie logiciel

Motivation 1ère du Génie Logiciel est la réduction des coûts de développement du logiciel.

GL: moyens (techniques et méthodologiques) permettant la construction de systèmes informatiques répondant à des critères de qualité préalablement définis.

GL : Méthodologie de construction en équipe d'un logiciel complexe et à multiples versions

Programmation & Génie Logiciel

- **Programmation : activité personnelle**
- **Génie Logiciel : activité d'équipe**
  - ✓ **Côté finance**
  - ✓ **Coût du logiciel supérieur à celui du matériel**
  - ✓ **Coût de maintenance supérieur au coût de conception**

Génie logiciel

Naissance du Génie Logiciel…

Génie logiciel

Naissance du Génie Logiciel…

Les pères du Génie Logiciel se prénomment Friedrich Bauer et Louis Bolliet.

Génie logiciel

Naissance du Génie Logiciel…

Les pères du Génie Logiciel se prénomment Friedrich Bauer et Louis Bolliet.

Cette spécialité a en effet vu le jour entre le 7 et le 11 octobre 1968,

- **à Garmisch-Partenkirchen,**
- **sous le parrainage de l'OTAN.**

Génie logiciel

Naissance du Génie Logiciel…

Les pères du Génie Logiciel se prénomment Friedrich Bauer et Louis Bolliet. Cette spécialité a en effet vu le jour entre le 7 et le 11 octobre 1968, à Garmisch-Partenkirchen, sous le parrainage de l'OTAN.

Elle avait pour objectif de répondre à 2 constations :

Génie logiciel

Naissance du Génie Logiciel…

Les pères du Génie Logiciel se prénomment Friedrich Bauer et Louis Bolliet.
Cette spécialité a en effet vu le jour entre le 7 et le 11 octobre 1968, à Garmisch-Partenkirchen, sous le parrainage de l'OTAN.
Elle avait pour objectif de répondre à 2 constations :

- **d'une part, le logiciel n'était pas fiable,**

Génie logiciel

Naissance du Génie Logiciel…

Les pères du Génie Logiciel se prénomment Friedrich Bauer et Louis Bolliet.
Cette spécialité a en effet vu le jour entre le 7 et le 11 octobre 1968, à Garmisch-Partenkirchen, sous le parrainage de l'OTAN.
Elle avait pour objectif de répondre à 2 constations :

- **d'une part, le logiciel n'était pas fiable,**
- **d'autre part, il était difficile à réaliser dans les délais et budgets prévus et en satisfaisant le cahier des charges**

Génie logiciel

Naissance du Génie Logiciel…

- **En 1993,**

Génie logiciel

Naissance du Génie Logiciel…

- **En 1993,**
  - **Branche informatique (Computer Society) de l'IEEE (Institute of Electrical and Electronics Engineers)**

Génie logiciel

Naissance du Génie Logiciel…

- **En 1993,**
  - **Branche informatique (Computer Society) de l'IEEE (Institute of Electrical and Electronics Engineers)**
  - **Organisation professionnelle américaine des informaticiens**

Génie logiciel

Naissance du Génie Logiciel…

- **En 1993,**
  - **Branche informatique (Computer Society) de l'IEEE (Institute of Electrical and Electronics Engineers)**
  - **Organisation professionnelle américaine des informaticiens**

  ont établi un comité conjoint dont la tâche était de définir :

Génie logiciel

Naissance du Génie Logiciel…

- **En 1993,**
  - **Branche informatique (Computer Society) de l'IEEE (Institute of Electrical and Electronics Engineers)**
  - **Organisation professionnelle américaine des informaticiens ont établi un comité conjoint dont la tâche était de définir :**
    - **ensemble de critères**
    - **et de normes appropriés pour la pratique professionnelle du Génie Logiciel (SWEBOK).**

Génie logiciel (Définition)

Le génie logiciel est un domaine des sciences de l'ingénieur dont l'objet d'étude est :

Génie logiciel (Définition)

Le génie logiciel est un domaine des sciences de l'ingénieur dont l'objet d'étude est :

- **la conception,**

Génie logiciel (Définition)

Le génie logiciel est un domaine des sciences de l'ingénieur dont l'objet d'étude est :

- **la conception,**
- **la fabrication,**

Génie logiciel (Définition)

Le génie logiciel est un domaine des sciences de l'ingénieur dont l'objet d'étude est :

- **la conception,**
- **la fabrication,**
- **la maintenance des systèmes informatiques complexes.**

Génie logiciel (Définition)

D’après Classical and Object-Oriented Software Engineering with UML and Java de Schach

Génie logiciel (Définition)

D’après Classical and Object-Oriented Software Engineering with UML and Java de Schach

- **Le Génie Logiciel est une discipline qui a pour but :**

Génie logiciel (Définition)

D'après Classical and Object-Oriented Software Engineering with UML and Java de Schach

- **Le Génie Logiciel est une discipline qui a pour but :**
  - **la fabrication du logiciel sans faute,**

Génie logiciel (Définition)

D'après Classical and Object-Oriented Software Engineering with UML and Java de Schach

- **Le Génie Logiciel est une discipline qui a pour but :**
  - **la fabrication du logiciel sans faute,**
  - **livré dans le délai**
  - **le budget prévus à l'avance,**
  - **et qui satisfait aux besoins du client**

Génie logiciel (Définition)

D'après Classical and Object-Oriented Software Engineering with UML and Java de Schach

- **Le Génie Logiciel est une discipline qui a pour but :**
  - **la fabrication du logiciel sans faute,**
  - **livré dans le délai**
  - **le budget prévus à l'avance**

Génie logiciel (Définition)

D'après Classical and Object-Oriented Software Engineering with UML and Java de Schach

- **Le Génie Logiciel est une discipline qui a pour but :**
  - **la fabrication du logiciel sans faute,**
  - **livré dans le délai**
  - **le budget prévus à l'avance,**
  - **et qui satisfait aux besoins du client**

Génie logiciel (Définition)

- **Système : un ensemble d'éléments interagissant entre eux :**

Génie logiciel (Définition)

- **Système : un ensemble d'éléments interagissant entre eux :**
  - **suivant un certains nombres de principes**

Génie logiciel (Définition)

- **Système : un ensemble d'éléments interagissant entre eux :**
  - **suivant un certains nombres de principes**
  - **et de règles**

Génie logiciel (Définition)

- **Système : un ensemble d'éléments interagissant entre eux**
  - **suivant un certains nombres de principes**
  - **et de règles**

dans le but de réaliser un objectif.

Génie logiciel (Définition)

- **Système : un ensemble d'éléments interagissant entre eux**
  - **suivant un certains nombres de principes**
  - **et de règles**

dans le but de réaliser un objectif.

- **Logiciel est un ensemble d'entités nécessaires au fonctionnement d'un processus de traitement automatique de l'information.**

Génie logiciel (Définition)

D’après MC. Gaudel

Le Génie Logiciel :

- **art de spécifier,**
- **concevoir,**
- **réaliser**
- **faire évoluer, avec des moyens**
- **et dans des délais raisonnables :**

Génie logiciel (Définition)

D'après MC. Gaudel

Le Génie Logiciel :

- ❑ **art de spécifier,**
- ❑ **concevoir,**
- ❑ **réaliser**
- ❑ **faire évoluer, avec des moyens**
- ❑ **et dans des délais raisonnables :**
  - ✓ **programmes,**
  - ✓ **documentations**
  - ✓ **et des procédures de qualité**

Génie logiciel (Définition)

D'après MC. Gaudel

Le Génie Logiciel :

- ❑ **art de spécifier,**
- ❑ **concevoir,**
- ❑ **réaliser**
- ❑ **faire évoluer, avec des moyens**
- ❑ **et dans des délais raisonnables :**
  - ✓ **programmes,**
  - ✓ **documentations**
  - ✓ **et des procédures de qualité**

en vue d'utiliser un système informatique pour résoudre certains problèmes

Génie logiciel (Définition)

Ou encore… :

- **Le Génie Logiciel**
  - **art**
  - **science de concevoir et de construire, avec économie et élégance :**

Génie logiciel (Définition)

Ou encore… :

- **Le Génie Logiciel**
  - **art**
  - **science de concevoir et de construire, avec économie et élégance :**
    - ✓ **des applications,**
    - ✓ **des objets,**
    - ✓ **des frameworks**
    - ✓ **et d'autres systèmes informatiques,**

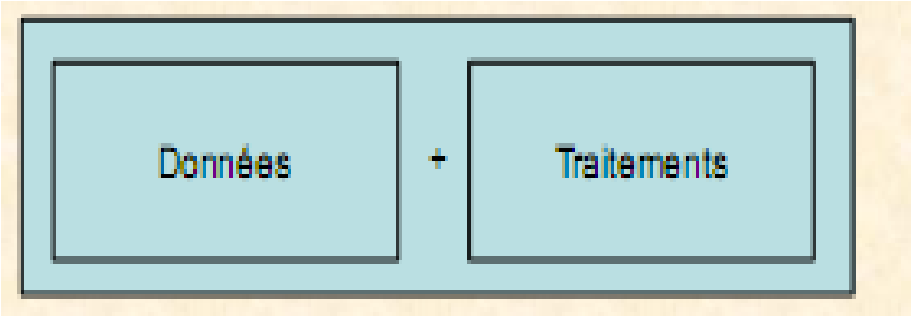
Génie logiciel (Définition)

Ou encore… :

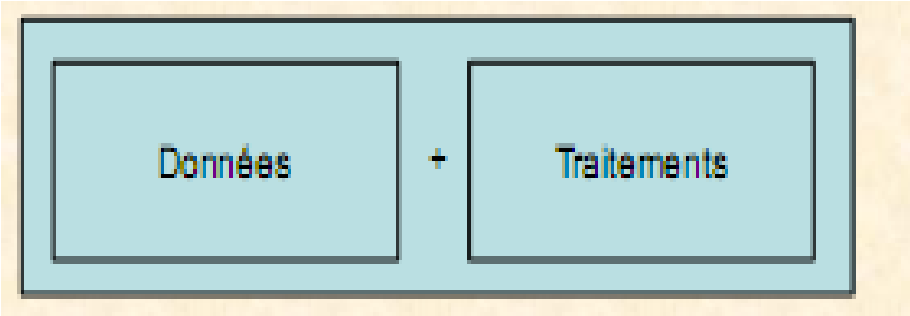
- **Le Génie Logiciel**
  - **art**
  - **science de concevoir et de construire, avec économie et élégance :**
    - ✓ **des applications,**
    - ✓ **des objets,**
    - ✓ **des frameworks**
    - ✓ **et d'autres systèmes informatiques,**

qui soient corrects, robustes, extensibles, réutilisables, sûrs, efficaces, faciles à maintenir et à utiliser

Evolution des approches

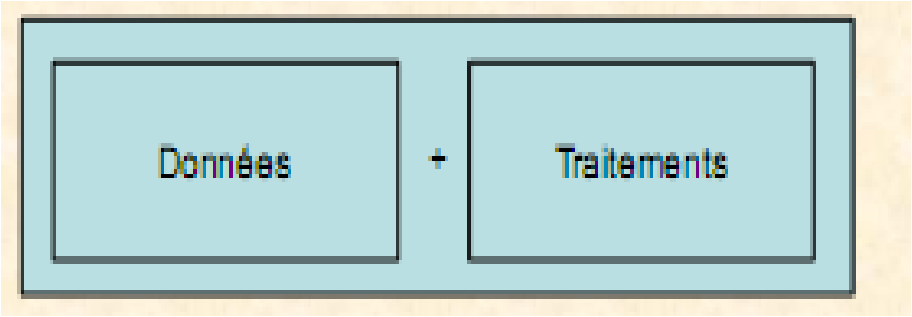


Evolution des approches



Logiciels classiques

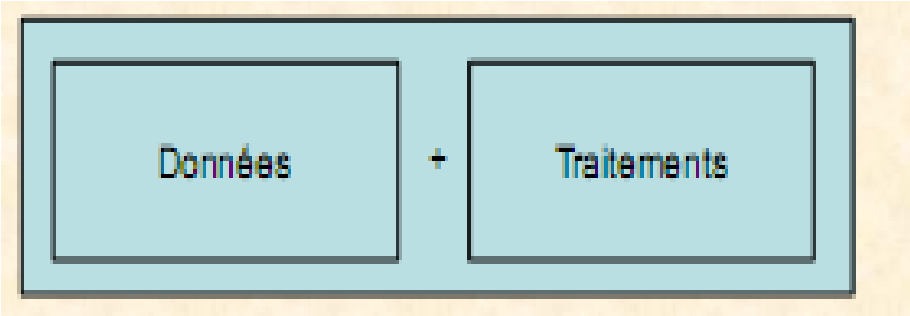
Evolution des approches



Logiciels classiques

Propriétés
Statiques

Evolution des approches

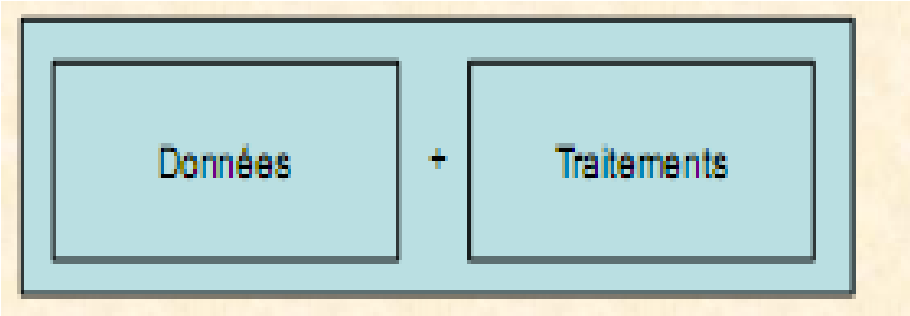


Logiciels classiques

Propriétés Statiques

Propriétés dynamiques

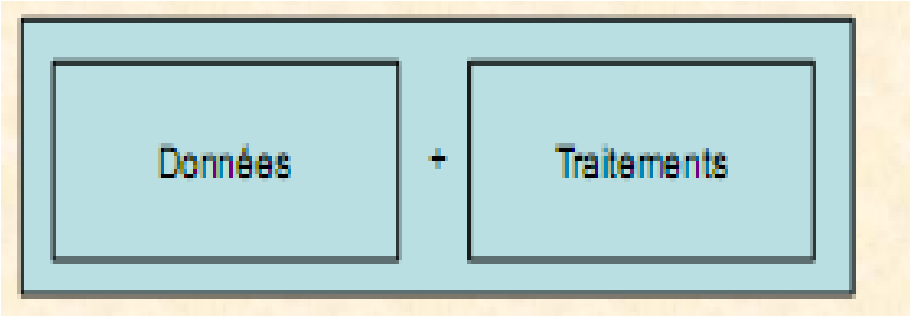
Evolution des approches



Système informatique

Les traitements prennent de plus en plus d'importance

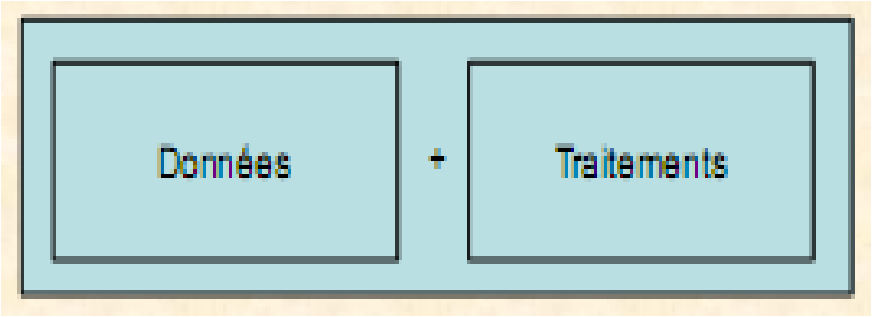
Evolution des approches



Systèmes d'Information (BD)

Les données sont au centre du système et persistantes

Evolution des approches



Systèmes d'Information d'Aide à la Décision

- **Les données sont au centre du système :orientation (entrepôt)**
  - Les données sont persistantes
  - Les systèmes procèdent de plus en plus à des post-traitements : fouille de données et aide à la décision

Bêtisier

Bêtisier

- **Défaillances : les spectaculaires**

Bêtisier

- **Défaillances : les spectaculaires**

La sonde Mariner qui devait effectuer un passage à 5 000 km de Vénus s'est perdue à 5 000 000 km de ladite planète à cause d'une erreur de programme Fortran :

Bêtisier

- **Défaillances : les spectaculaires**

La sonde Mariner qui devait effectuer un passage à 5 000 km de Vénus s'est perdue à 5 000 000 km de ladite planète à cause d'une erreur de programme Fortran : le point avait été remplacé par une virgule

Bêtisier

- **Défaillances : les navrantes**

Bêtisier

- **Défaillances : les navrantes**
  - **Le Therac-25, un appareil d'irradiation thérapeutique a provoqué la mort de 2 personnes, et l'irradiation de 4 autres,**

Bêtisier

- **Défaillances : les navrantes**
  - **Le Therac-25, un appareil d'irradiation thérapeutique a provoqué la mort de 2 personnes, et l'irradiation de 4 autres,**
    →cause : erreur logicielle.

Bêtisier

❑Défaillances : les navrantes

❖**Un navire de guerre anglais a été coulé par un Exocet français, au cours de la guerre des Malouines.**

Bêtisier

❑Défaillances : les navrantes

❖**Un navire de guerre anglais a été coulé par un Exocet français, au cours de la guerre des Malouines.**

Bilan de la catastrophe : plusieurs centaines de morts.

Bêtisier

❑Défaillances : les navrantes

❖**Un navire de guerre anglais a été coulé par un Exocet français, au cours de la guerre des Malouines.**

Bilan de la catastrophe : plusieurs centaines de morts.

Le vaisseau anglais n'avait, en effet, pas activé ses défenses, l'Exocet n'étant pas répertorié comme missile ennemi.

Bêtisier

- **Défaillances : les navrantes**
  - **En 1983, toute la vallée du Colorado a été inondée.**

Bêtisier

- **Défaillances : les navrantes**
  - **En 1983, toute la vallée du Colorado a été inondée.**

La cause ?

Bêtisier

- **Défaillances : les navrantes**
  - **En 1983, toute la vallée du Colorado a été inondée.**

La cause ?

Une mauvaise modélisation dans le logiciel du temps d'ouverture du barrage

Bêtisier

- **Défaillances : les risibles**

Bêtisier

- **Défaillances : les risibles (ridicules)**

Bêtisier

- **Défaillances : les risibles**
  - **Une personne s'est retrouvée saisie pour une dette impayée de 0,01F et cela bien qu'elle se soit acquittée de l'intégralité de sa dette.**

Bêtisier

- **Défaillances : les risibles**
  - **Une personne s'est retrouvée saisie pour une dette impayée de 0,01F et cela bien qu'elle se soit acquittée de l'intégralité de sa dette.**

- **Le coupable ?**

Bêtisier

- **Défaillances : les risibles**
  - **Une personne s'est retrouvée saisie pour une dette impayée de 0,01F et cela bien qu'elle se soit acquittée de l'intégralité de sa dette.**
    - **Le coupable ?**
      - **un arrondi mal maîtrisé dans les calculs.**

Bêtisier

- **Défaillances : les risibles**
  - **Une centenaire (106 ans) s’est retrouvée convoquée à l'école.**

Bêtisier

- **Défaillances : les risibles**
  - **Une centenaire (106 ans) s'est retrouvée convoquée à l'école.**

La cause ?

Bêtisier

- **Défaillances : les risibles**
  - **Une centenaire (106 ans) s'est retrouvée convoquée à l'école.**

La cause ?

- **Codage de l'année de naissance sur deux caractères.**

Bêtisier

- **Défaillances : les coûteuses**

Bêtisier

- **Défaillances : les coûteuses**
- **Bug du Pentium en 1994. Coût : 500 millions de dollars**
- **Bug de l'an 2000. Coût de la mise à niveau des logiciels à la France : qqs milliards de francs.**
- **Le 22 décembre 2001, en pleine période des achats de Noël, les 750 000 terminaux de paiement ne répondent plus chez les commerçants, entraînant de longues files d'attente de clients excédés dont beaucoup finissent par abandonner leurs chariots. La cause : la saturation des serveurs de la société Atos en charge des autorisations de paiement dépassant 600F. Les autorisations de débit qui prennent habituellement quelques dizaines de secondes, nécessitent ce jour-là quasiment la demi-heure.**

Bêtisier

- **Défaillances : les coûteuses**
- **Bug du Pentium en 1994. Coût : 500 millions de dollars**
- **Bug de l'an 2000. Coût de la mise à niveau des logiciels à la France : qqs milliards de francs.**
- **Le 22 décembre 2001, en pleine période des achats de Noël, les 750 000 terminaux de paiement ne répondent plus chez les commerçants, entraînant de longues files d'attente de clients excédés dont beaucoup finissent par abandonner leurs chariots. La cause : la saturation des serveurs de la société Atos en charge des autorisations de paiement dépassant 600F. Les autorisations de débit qui prennent habituellement quelques dizaines de secondes, nécessitent ce jour-là quasiment la demi-heure. Le coût du préjudice pour le seul groupe Leclerc : 2 millions d'Euros.**

Bêtisier

- **Dépassement des délais et des coûts :**
- **En mars 1993, la bourse de Londres a renoncé au projet informatique Taurus qui devait assurer complètement le suivi de l'exécution des transactions. Le système avait coûté directement 60 millions de livres et les opérateurs sur le marché avait dépensé 400 millions de livres pour y adapter leurs propres logiciels.**

Bêtisier

- **Dépassement des délais et des coûts :**
- **En mars 1993, la bourse de Londres a renoncé au projet informatique Taurus qui devait assurer complètement le suivi de l'exécution des transactions. Le système avait coûté directement 60 millions de livres et les opérateurs sur le marché avait dépensé 400 millions de livres pour y adapter leurs propres logiciels.**
- **Toujours en 1993, en Californie, le DMV a renoncé au projet informatique démarré 6 ans plus tôt qui devait intégrer les systèmes d'information destinés à gérer les immatriculations et les permis. Les coûts étaient de 6,5 fois ceux prévus initialement et la date de livraison probable avait été reportée à 1998.**

Quelques Etudes

- **Respect des engagements :**

Quelques Etudes

- **Respect des engagements :**
- **30% des projets informatiques sont annulés avant la mise en production (Aberdeen).**

Quelques Etudes

- **Respect des engagements :**
- **30% des projets informatiques sont annulés avant la mise en production (Aberdeen).**
- **En 1995, aux USA, on estime à 91 billions de dollars, la somme dépensée pour des projets arrêtés avant la fin.**

Quelques Etudes

- **Respect des engagements :**
- **30% des projets informatiques sont annulés avant la mise en production (Aberdeen).**
- **En 1995, aux USA, on estime à 91 billions de dollars, la somme dépensée pour des projets arrêtés avant la fin.**
- **90% des projets informatiques sortent en retard (Aberdeen).**

Quelques Etudes

❑Respect des engagements :

❑30% des projets informatiques sont annulés avant la mise en production (Aberdeen).

❑En 1995, aux USA, on estime à 91 billions de dollars, la somme dépensée pour des projets arrêtés avant la fin.

❑90% des projets informatiques sortent en retard (Aberdeen).

❑50% des projets informatiques ne répondent pas au cahier des charges Business (Gartner).

Quelques Etudes

❑Respect des engagements :

❑30% des projets informatiques sont annulés avant la mise en production (Aberdeen).

❑En 1995, aux USA, on estime à 91 billions de dollars, la somme dépensée pour des projets arrêtés avant la fin.

❑90% des projets informatiques sortent en retard (Aberdeen).

❑50% des projets informatiques ne répondent pas au cahier des charges Business (Gartner).

❑50% des projets informatiques dépassent le budget prévu (Gartner).

Quelques Etudes

- **Respect des engagements :**
- **30% des projets informatiques sont annulés avant la mise en production (Aberdeen).**
- **En 1995, aux USA, on estime à 91 billions de dollars, la somme dépensée pour des projets arrêtés avant la fin.**
- **90% des projets informatiques sortent en retard (Aberdeen).**
- **50% des projets informatiques ne répondent pas au cahier des charges Business (Gartner).**
- **50% des projets informatiques dépassent le budget prévu (Gartner). Ce surplus de coût se répartit de la manière suivante :**

Quelques Etudes

❑Respect des engagements :

❑30% des projets informatiques sont annulés avant la mise en production (Aberdeen).

❑En 1995, aux USA, on estime à 91 billions de dollars, la somme dépensée pour des projets arrêtés avant la fin.

❑90% des projets informatiques sortent en retard (Aberdeen).

❑50% des projets informatiques ne répondent pas au cahier des charges Business (Gartner).

❑50% des projets informatiques dépassent le budget prévu (Gartner). Ce surplus de coût se répartit de la manière suivante :

- **❖maintenance corrective : 20 %**
- **❖maintenance adaptative : 25 %**
- **❖maintenance évolutive : 55 %**

Quelques Etudes

- **Coût des modifications :**

en fonction du moment où elles interviennent dans le cycle de vie du projet

Quelques Etudes

Logiciels	Coût
payés, jamais livrés	3.2 M $
livrés, jamais utilisés	2.0 M $
abandonnés ou recommencés	1.3 M $
utilisés après modification	0.2 M $
utilisés en l'état	0.1 M $

Quelques Etudes

Logiciels	Coût
payés, jamais livrés	3.2 M $
livrés, jamais utilisés	2.0 M $
abandonnés ou recommencés	1.3 M $
utilisés après modification	0.2 M $
utilisés en l'état	0.1 M $

Beaucoup de logiciels ne sont pas livrés, pas utilisés ou abandonnés.
Plus précisément , le coût se répartit de la façon suivante

Quelques Etudes

Logiciels	Coût
payés, jamais livrés	3.2 M $
livrés, jamais utilisés	2.0 M $
abandonnés ou recommencés	1.3 M $
utilisés après modification	0.2 M $
utilisés en l'état	0.1 M $

D'après cette étude, seul 5% du coût, correspondant à des logiciels utilisés soit en l'état, soit après modification, est donc « acceptable ».

Quelques Etudes

Etude du gouvernement américain - 1979,

La crise du logiciel

Logiciels	Coût
payés, jamais livrés	3.2 M $
livrés, jamais utilisés	2.0 M $
abandonnés ou recommencés	1.3 M $
utilisés après modification	0.2 M $
utilisés en l'état	0.1 M $

D'après cette étude, seul 5% du coût, correspondant à des logiciels utilisés soit en l'état, soit après modification, est donc « acceptable ».

Quelques Etudes

Autre étude (1995-USA)

Logiciels	Coût
abandonnés ou jamais utilisés	31%
coûts largement dépassés	53%
réalisés suivant les plans	16%

La « crise du logiciel » est donc loin d'être terminée.

Quelques Etudes

Autre étude (1995-USA)

Logiciels	Coût
abandonnés ou jamais utilisés	31%
coûts largement dépassés	53%
réalisés suivant les plans	16%

La « crise du logiciel » est donc loin d'être terminée.

(prise de conscience que le coût du logiciel dépassait le coût du matériel)

Quelques Etudes

❑Coûts de l'informatique :

- **❑Les dépenses en informatique sont extrêmement importantes. A titre indicatif, elles représentent aux USA, en pourcentage du PNB :**
  - **❖1980 : 5 %**
  - **❖1985 : 8 %**
  - **❖1990 : 12,5 %**
- **❑Boehm estimait qu'en 1985, le coût des développements logiciels dans le monde s'élevait à 140 billions de dollars avec une progression de 12 % par an.**

Quelques Etudes

❑Productivité :

❑Une étude réalisée par Seckman en 1968 sur la productivité, basée sur le travail de 12 programmeurs chevronnés, met en évidence les écarts suivants :

- **❖heures de codage : de 1 à 25**
- **❖heures de mise au point : de 1 à 26**
- **❖taille du programme : de 1 à 5**
- **❖temps d'exécution : de 1 à 13**

❑Après élimination des extrêmes :

- **❖indice de production : de 1 à 5**

Les Constats

- **Années 70 voient naître une crise de l'industrie du logiciel dont les principales raisons sont :**
  - le non-respect (augmentation) des coûts
  - le non-respect (augmentation) des délais

Les Constats

- **Années 70 voient naître une crise de l'industrie du logiciel dont les principales raisons sont :**
  - le non-respect (augmentation) des coûts
  - le non-respect (augmentation) des délais
  - l'inadéquation avec les besoins des utilisateurs
  - le non-respect des spécifications

Les Constats

- **Années 70 voient naître une crise de l'industrie du logiciel dont les principales raisons sont :**
  - le non-respect (augmentation) des coûts
  - le non-respect (augmentation) des délais
  - l'inadéquation avec les besoins des utilisateurs
  - le non-respect des spécifications
  - la non-fiabilité
  - les difficultés d'évolution

Les Constats

- S'il n'est pas simple d'y remédier, les causes de ces dysfonctionnements sont cependant faciles à identifier :
  - **une communication difficile :**
    - **jargon utilisateur versus jargon informatique**
    - **évolution de la manière de voir un problème**
    - **réflexion utilisateur pas assez mature**

Les Constats

- S'il n'est pas simple d'y remédier, les causes de ces dysfonctionnements sont cependant faciles à identifier :
  - **une communication difficile :**
    - **jargon utilisateur versus jargon informatique**
    - **évolution de la manière de voir un problème**
    - **réflexion utilisateur pas assez mature**
    - **marché, législation et besoin en perpétuelle évolution**
    - **addition successive d'imprécision dans la communication entre les individus d'une équipe (la complexité ++ avec le nombre d'intervenants)**

Les Constats

- **S'il n'est pas simple d'y remédier, les causes de ces dysfonctionnements sont cependant faciles à identifier :**
  - **une conception et un développement difficiles :**
    - **complexité croissante des logiciels**
      (la productivité -- lorsque la complexité du logiciel ++)
    - **trop d'interrelations entre composants logiciels**

Les Constats

- **S'il n'est pas simple d'y remédier, les causes de ces dysfonctionnements sont cependant faciles à identifier :**
  - **une conception et un développement difficiles :**
    - **complexité croissante des logiciels**
      (la productivité -- lorsque la complexité du logiciel ++)
    - **trop d'interrelations entre composants logiciels**
    - **trop de modifications (évolutions)**
    - **tests trop souvent insuffisants**

Lois d‘évolution du logiciel

Lois d‘évolution du logiciel

Meir Lehman a donné des lois sur l'évolution des logiciels

Lois de Lehman

Lois de Lehman

A propos d'évolutions… en 1985, Lehman a émis un certain nombre d'hypothèses concernant la dynamique d'évolution des logiciels :

Lois de Lehman

- **Modifications perpétuelles (1ère) : Un logiciel utilisé dans un environnement réel doit nécessairement évoluer sinon il devient de moins en moins utile dans cet environnement**

Lois de Lehman

- **Modifications perpétuelles (1ère) : Un logiciel utilisé dans un environnement réel doit nécessairement évoluer sinon il devient de moins en moins utile dans cet environnement**
- **Complexité croissante (2ème) : Au fur et à mesure qu'un logiciel évolue, sa structure a tendance à se complexifier.**

Lois de Lehman

- **Évolution des gros logiciels (3ème) : L'évolution d'un logiciel est un processus auto-régulé. Les attributs du système comme sa taille, le temps entre 2 versions ou le nombre de bogues signalés ne varient quasiment pas d'une version à l'autre**

Lois de Lehman

- **Évolution des gros logiciels (3ème) : L'évolution d'un logiciel est un processus auto-régulé. Les attributs du système comme sa taille, le temps entre 2 versions ou le nombre de bogues signalés ne varient quasiment pas d'une version à l'autre**
- **Stabilité organisationnelle (4ème) : Sur la durée de vie d'un logiciel, le taux de modification est à peu près constant, quel que soit l'effort qu'on lui consacre**

Lois de Lehman

- **Évolution des gros logiciels (3ème) : L'évolution d'un logiciel est un processus auto-régulé. Les attributs du système comme sa taille, le temps entre 2 versions ou le nombre de bogues signalés ne varient quasiment pas d'une version à l'autre**
- **Stabilité organisationnelle (4ème) : Sur la durée de vie d'un logiciel, le taux de modification est à peu près constant, quel que soit l'effort qu'on lui consacre**
- **Conservation de la familiarité (5ème) : Sur la durée de vie d'un logiciel, les modifications incrémentales de chaque nouvelle version sont à peu près constantes**

Principes du GL

1. La rigueur
2. La décomposition des problèmes en sous-problèmes
3. La modularité
4. L'abstraction
5. L'anticipation des évolutions
6. La généricité
7. La construction incrémentale

- ❑Grands principes se retrouvent dans toutes ces méthodes
- ❑Liste proposée par Ghezzi

Principes du GL

- **P1 : Rigueur**
  - Les principales sources de défaillances d'un logiciel sont d'origine humaine.
  - À tout moment, il faut se questionner sur la validité de son action.
  - Des outils de vérification accompagnant le développement peuvent aider à réduire les erreurs.
    - ✓Cette famille d'outils s'appelle CASE (Computer Aided Software Engineering).

Exemple
Générateurs de code, assistants de preuves, profileurs, …

Principes du GL

- **P2 : Décomposition des problèmes en sous-problèmes**

  Il s'agit de :

  - **Traiter qu'un seul problème à la fois.**
  - **Simplifier les pb (tempr) pour aborder leur complexité progressivement.**

Exemple (Comment créer dynamiquement une page internet pour visualiser et modifier le contenu d'une base donnée sans la corrompre ?)

Décomposition en trois composants :

- Modèle : gérer le stockage des données.
- Vue : formatter les données.
- Contrôleur : autoriser que les modifications correctes

Principes du GL

- **P3 : Modularité**
  - C'est une instance cruciale du principe de décomposition des problèmes.
  - Il s'agit de partitionner le logiciel en modules qui :
    - ✓ ont une cohérence interne (invariants) ;
    - ✓ possèdent une interface ne divulguant sur le contenu du module que ce qui est strictement nécessaire aux modules clients.
  - L'évolution de l'interface est indépendante de celle de l'implémentation du module.
  - Les choix d'implémentation sont indépendants de l'utilisation du module.
  - Ce mécanisme s'appelle le camouflage de l'information (information hiding).

Principes du GL

- **P4 : Abstraction**
  - Exhiber des concepts généraux regroupant un certain nombre de cas particuliers et de raisonner sur ces concepts généraux plutôt que sur chacun des cas particuliers.
  - Le fait de fixer la bonne granularité de détails permet :
    - ✓raisonner plus efficacement ;
    - ✓factoriser le travail en instanciant le raisonnement général sur chaque cas particulier.

Exemple (Support dans les langages de programmation)

- classes abstraites dans les langages à objets, le generics de Java,. . .

Principes du GL

- **P5 : Anticipation des évolutions**
  - **Un logiciel a un cycle de vie plus complexe que l'habituel commande**
    → spécification → production → livraison
  - **La maintenance est la gestion des évolutions du logiciel.**
  - **Il est primordial de prévoir les évolutions possibles d'un logiciel pour que la maintenance soit la plus efficace possible.**
  - **Pour cela, il faut s'assurer que les modifications à effectuer soient le plus locales possibles.**
  - **Ces modifications ne devraient pas être intrusives car les modifications du produit existant remettent en cause ses précédentes validations.**
  - **Concevoir un système suffisamment riche pour que l'on puisse le modifier incrémentalement est l'idéal.**

Principes du GL

- **P6 : Généricité**
  - **Un logiciel réutilisable a beaucoup plus de valeur qu'un composant dédié.**
  - **Un composant est générique lorsqu'il est adaptable.**

Principes du GL

- **P7 : Construction incrémentale**
  - **Un développement logiciel a plus de chances d'aboutir si il suit**
  - **une cheminement incrémental (baby-steps).**

Exemple
Laquelle de ses deux méthodes de prog est la plus efficace ?

1. Écrire l'ensemble du code source d'un programme et compiler.
2. Écrire le code source d'une fonction ou module, le compiler, et passer à la suivante.

Pourquoi ?

Cycle de vie et qualité du logiciel

Cycle de vie du logiciel

Différents modèles ont été proposés

Le cycle en V est le cycle qui a été normalisé
Il est largement utilisé, notamment en informatique industrielle et télécoms.

Cycle de vie du logiciel

Étapes (non exhaustive...)

- Définition des besoins
  - Dans le langage du client
- Spécifications
  - Traduction des besoins dans un langage plus informatique
  - Ce que doit faire le logiciel (et non comment il le fait)
- Conception
  - Traduction des spécifications en termes de concepts logiciels
- Codage
  - Traduction de la conception en code
- Tests
  - unitaires (Test de chaque module individuellement)
  - d'intégration (Test de la composition de plusieurs modules)

Cycle de vie du logiciel

Étapes (non exhaustive...)

- Validation - Vérification
  - Avons-nous construit le bon logiciel ? ->Validé
  - Avons-nous bien construit le logiciel ? ->Vérifié
- Livraison / Diffusion
- Support
  - formation
- Maintenance
- Evolution
  - nouvelles versions, ...

Cycle de vie du logiciel

Modèles – Méthodes/Définitions

- **Cycle de vie**
  - **Le cycle de vie désigne toutes les étapes du développement d'un logiciel, de sa conception à sa disparition.**

Cycle de vie du logiciel

Modèles – Méthodes/Définitions

- **Cycle de vie**
  - **Le cycle de vie désigne toutes les étapes du développement d'un logiciel, de sa conception à sa disparition.**
- **Modèles**
  - **Les modèles décrivent les liens, les relations entre les différentes étapes du cycle de vie du logiciel.**

Cycle de vie du logiciel

Modèles – Méthodes/Définitions

- **Cycle de vie**
  - **Le cycle de vie désigne toutes les étapes du développement d'un logiciel, de sa conception à sa disparition.**
- **Modèles**
  - **Les modèles décrivent les liens, les relations entre les différentes étapes du cycle de vie du logiciel.**
- **Méthodes**
  - **Les méthodes permettent de mettre en oeuvre un développement logiciel selon un modèle en organisant les différentes étapes du cycle de vie du logiciel.**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

1. **Analyse & définition des besoin**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

1. **Analyse & définition des besoin**
2. **Conception**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

1. **Analyse & définition des besoin**
2. **Conception**
3. **Mise en œuvre**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

1. **Analyse & définition des besoin**
2. **Conception**
3. **Mise en œuvre**
4. **Validation**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

1. **Analyse & définition des besoin**
2. **Conception**
3. **Mise en œuvre**
4. **Validation**
5. **Maintenance**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Analyse & définition des besoin

- **Consiste à déterminer les attentes des futurs utilisateurs (CC).**
- **Décrie à la fois le système & l'envir dans lequel le système sera exécuté.**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Analyse & définition des besoin

- **Consiste à déterminer les attentes des futurs utilisateurs (CC).**
- **Décrie à la fois le système & l'envir dans lequel le système sera exécuté.**
- **Cette étape comprend également une étude de faisabilité de la part des experts**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Analyse & définition des besoin

- **Consiste à déterminer les attentes des futurs utilisateurs (CC).**
- **Décrie à la fois le système & l'envir dans lequel le système sera exécuté.**
- **Cette étape comprend également une étude de faisabilité de la part des experts**

C'est sur la base de cette étape que le contrat est signé

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Conception

- **Spécification système**
- **Conception architecturale**
- **Conception détaillée**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Conception

Spécification système :

- **La spécification du système est une description des fonctionnalités.**
- **Il s'agit de décrire ce que le système doit faire, sans préciser comment ces fonctionnalités seront implémentées.**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Conception

Conception architecturale

La conception de l'architecture consiste à décrire la structure générale du système

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Conception

Conception détaillée

La conception détaillée du système consiste en une description des composants, des algorithmes et des structures de données.

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Mise en œuvre

Consiste à programmer le logiciel, en suivant les choix effectués lors de l'analyse et la conception.

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Validation

Consiste à s'assurer que le programme est de qualité.
Il existe plusieurs techniques de validation :

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Validation

Consiste à s'assurer que le programme est de qualité.
Il existe plusieurs techniques de validation :

- **tests**
- **analyse statique**
- **preuve formelle…**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Validation

Consiste à s'assurer que le programme est de qualité.
Il existe plusieurs techniques de validation :

- **Tests**
  - **Unitaire**
  - **Intégration**
  - **Système**
  - **acceptation**
- **analyse statique**
- **preuve formelle…**

Étapes du cycle de vie du logiciel

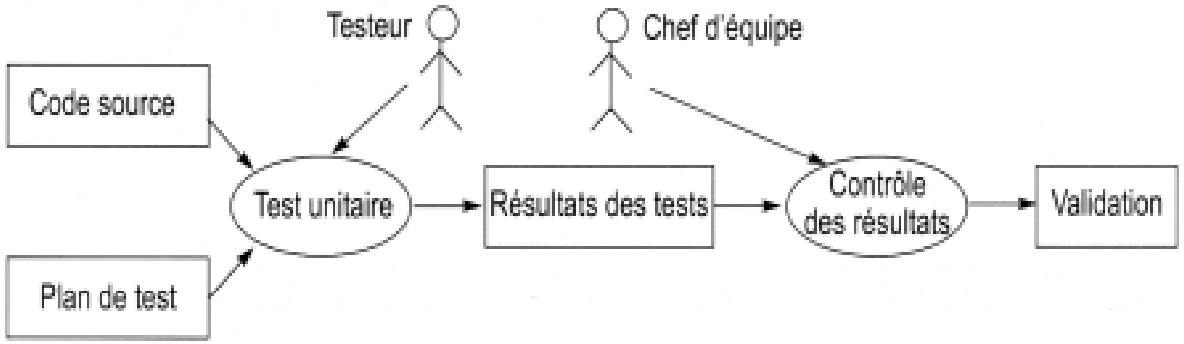
Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Validation

Consiste à s'assurer que le programme est de qualité.
Il existe plusieurs techniques de validation :

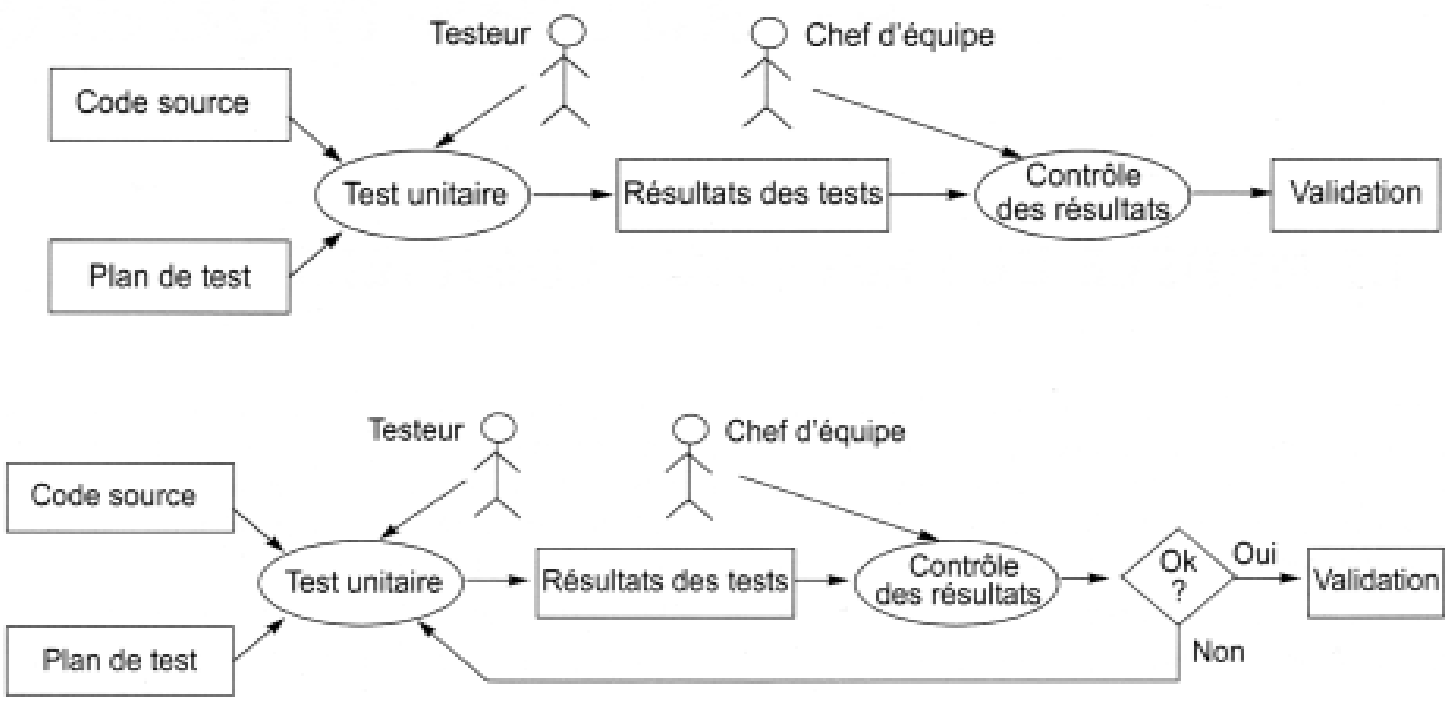
- **Tests (constituent la principale méthode de validation)**
  - **Unitaire**
  - **Intégration**
  - **Système**
  - **acceptation**
- **analyse statique**
- **preuve formelle…**

Étapes du cycle de vie du logiciel



- **Tests (constituent la principale méthode de validation)**
  - **Unitaire (exemple)**

Étapes du cycle de vie du logiciel



- **Tests (constituent la principale méthode de validation)**
  - **Unitaire**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Maintenance

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Maintenance

- **maintenance corrective**
- **maintenance évolutive**
- **maintenance adaptative**

Étapes du cycle de vie du logiciel

Le développement de logiciel impose d'effectuer un certain nombre d'étapes.

Maintenance

- ❑ **maintenance corrective**
- ❑ **maintenance évolutive**
- ❑ **maintenance adaptative**

Modifications du logiciel après sa livraison

Fouiller /TPE

Émergence des méthodes agiles :

- RAD (Rapid Application Development)
- UP (Unified Process), RUP (Rational Unified Process),
- 2TUP (Two Tracks Unified Process)
- ASD (Adaptative Software Development)
- DSDM (Dynamic Software Development Method)

Fouiller /TPE

Émergence des méthodes agiles :

- RAD (Rapid Application Development)
- UP (Unified Process), RUP (Rational Unified Process),
- 2TUP (Two Tracks Unified Process)
- ASD (Adaptative Software Development)
- DSDM (Dynamic Software Development Method)

Approches par la structure et par les besoins

Modélisation du processus de développ.

Modélisation du processus de dévelopg.

Un processus de dévelopg. logiciel est un ensemble (structuré) d'activités que conduisent à la production d'un logiciel

Modélisation du processus de dévelopg.

Un processus de dévelopg. logiciel est un ensemble (structuré) d'activités que conduisent à la production d'un logiciel

- **Il n'existe pas de processus idéal.**

Modélisation du processus de développ.

Un processus de développ. logiciel est un ensemble (structuré) d'activités que conduisent à la production d'un logiciel

- **Il n'existe pas de processus idéal.**
- **La plupart des entreprises adapte les processus existants à leurs besoins**

Modélisation du processus de dévelopg.

Un processus de dévelopg. logiciel est un ensemble (structuré) d'activités que conduisent à la production d'un logiciel

- **Il n'existe pas de processus idéal.**
- **La plupart des entreprises adapte les processus existants à leurs besoins.**
- **Ces besoins varient en fonction**
  - **du domaine,**
  - **des contraintes**

Modélisation du processus de dévelopg.

Un processus de dévelopg. logiciel est un ensemble (structuré) d'activités que conduisent à la production d'un logiciel

- **Il n'existe pas de processus idéal.**
- **La plupart des entreprises adapte les processus existants à leurs besoins.**
- **Ces besoins varient en fonction**
  - **du domaine,**
  - **des contraintes**
  - **de qualité,**
  - **des personnes impliquées.**

Modélisation du processus de dévelopment.

Un processus de développement permet de décrire l'enchaînement des différentes étapes du développement.

Objectif : processus qui permet de contrôler le développement afin que le logiciel :

Modélisation du processus de développ.

Un processus de développement permet de décrire l'enchaînement des différentes étapes du développement.
Objectif : processus qui permet de contrôler le développement afin que le logiciel :

- **soit livré dans les délais ;**
- **respecte le budget ;**
- **soit de qualité.**

Modélisation du processus de développ.

Un processus de développement permet de décrire l'enchaînement des différentes étapes du développement.
L'objectif est de proposer un processus qui permet de contrôler le développ. afin que le logiciel :

- ❑soit livré dans les délais ;
- ❑respecte le budget ;
- ❑soit de qualité.

D'où les modèles du cycle de vie (MCV) du logiciel

Modélisation du processus de développ.

Un processus de développement permet de décrire l'enchaînement des différentes étapes du développement.
L'objectif est de proposer un processus qui permet de contrôler le développ. afin que le logiciel :

- ❑ **soit livré dans les délais ;**
- ❑ **respecte le budget ;**
- ❑ **soit de qualité.**

Les MCV du logiciel sont des « plans de travail » qui permettent de planifier le développement.

Modélisation du processus de développ.

Un processus de développement permet de décrire l'enchaînement des différentes étapes du développement.
L'objectif est de proposer un processus qui permet de contrôler le développ. afin que le logiciel :

- ❑soit livré dans les délais ;
- ❑respecte le budget ;
- ❑soit de qualité.

Les MCV du logiciel sont des « plans de travail » qui permettent de planifier le développement.
Plus le logiciel à développer est complexe (taille, algorithmes) et critique, plus il est important de bien contrôler le processus de développement et plus les documents qui accompagnent le logiciel doivent être précis et détaillés

Activités du développement logiciel

Activités des processus de DL se regroupnt en 5 grdes catégories :

1. **Spécification du logiciel définit ses fonctionnalités/contraintes.**
2. **La conception . . .**
3. **Implémentation : chargée de réaliser le logiciel, en conformité avec sa spécification.**
4. **Validation s'assure effectivement du respect de la spécification par le logiciel produit.**
5. **Évolution adapte le logiciel aux besoins futurs de ses clients.**

Schéma général d'un processus de dévp

Il est très rare d'appliquer un processus comme une unique séquence des 5 activités/précédentes.

- **En encontre du principe d'incrémentalité.**
- **Un logiciel complet est le fruit de plusieurs itérations.**
- **Chaque itération contient les 5 activités :**
  - ✓ **Spécification,**
  - ✓ **Conception,**
  - ✓ **Implémentation,**
  - ✓ **Validation**
  - ✓ **Évolution.**

Schéma général d'un processus de dévp

Il est très rare d'appliquer un processus comme une unique séquence des 5 activités/précédentes.

- ❑ **En encontre du principe d'incrémentalité.**
- ❑ **Un logiciel complet est le fruit de plusieurs itérations.**
- ❑ **Chaque itération contient les 5 activités :**
  - ✓ **Spécification,**
  - ✓ **Conception,**
  - ✓ **Implémentation,**
  - ✓ **Validation**
  - ✓ **Évolution.**

Il existe différents modèles de processus qui organisent de façon différentes ces activités

Modèle du cycle de vie

Modèle du cycle de vie

Modèle du cycle de vie en cascade
Modèle du cycle de vie en V
Modèle du cycle de vie incrémental
Modèle du cycle de vie en spirale

Modèle du cycle de vie

Modèle du cycle de vie en cascade

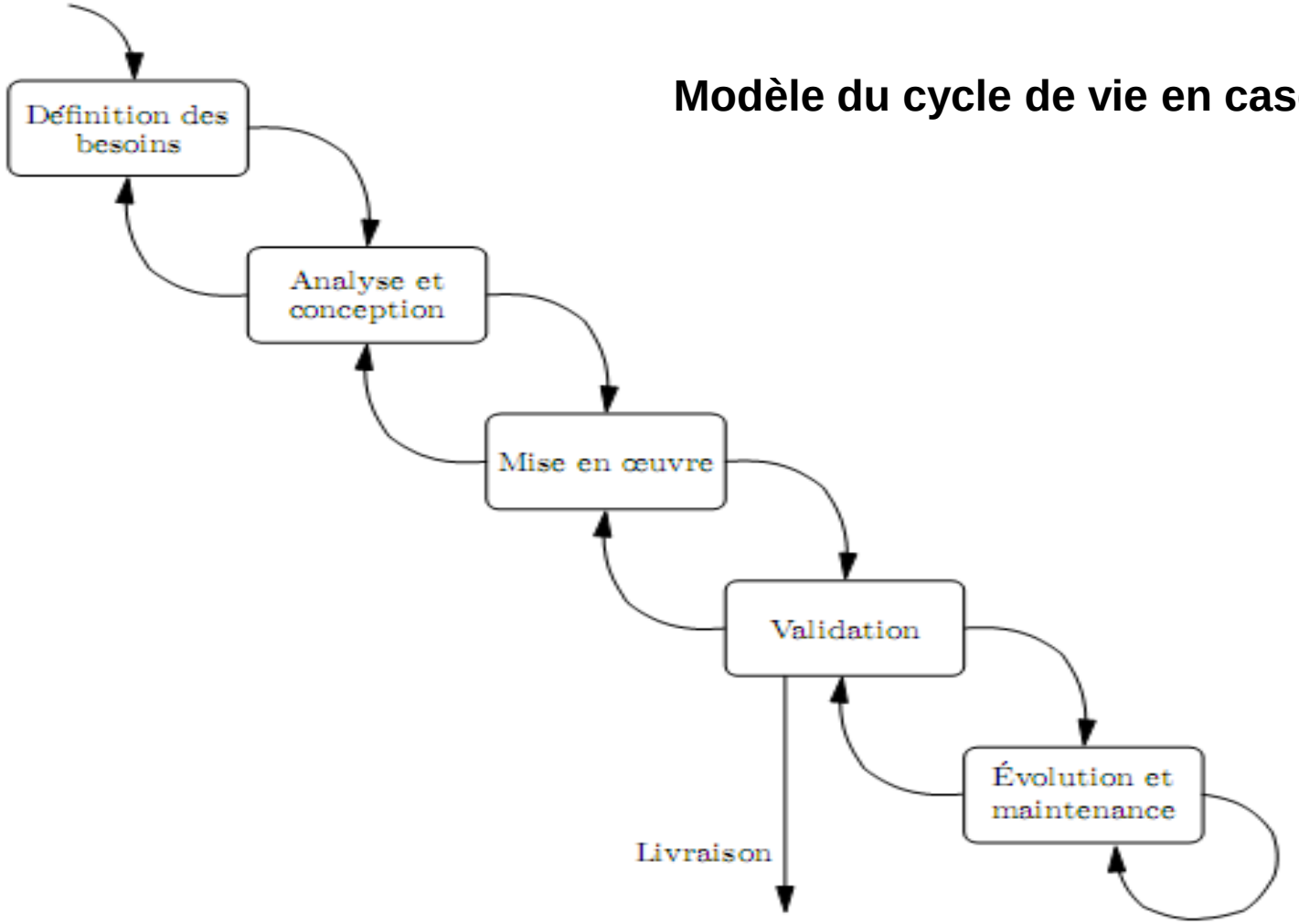
Projet de petite taille & domaine d'étude maitrisé

Modèle du cycle de vie

Modèle du cycle de vie en cascade

Modèle du cycle de vie

Modèle du cycle de vie en cascade



Modèle du cycle de vie

Modèle du cycle de vie en cascade

Projet de petite taille & domaine d'étude maitrisé

Modèle incontrôlable

Lorsque qu'un test remet en cause la conception détaillée ou architecturale ou, pire, les spécifications ou la définition des besoins, le modèle devient incontrôlable

Modèle du cycle de vie

Modèle du cycle de vie en V

Domaine d’étude maitrisé

Modèle du cycle de vie

Modèle du cycle de vie incrémental

Modèle du cycle de vie

Modèle du cycle de vie incrémental

Modèle itératif, qui consiste à sélectionner successivement plusieurs incréments. Un incrément du logiciel est un sous-ensemble du logiciel complet, qui consiste en un petit nombre de fonctionnalités.

Modèle du cycle de vie

Modèle du cycle de vie incrémental

Modèle itératif, qui consiste à sélectionner successivement plusieurs incréments. Un incrément du logiciel est un sous-ensemble du logiciel complet, qui consiste en un petit nombre de fonctionnalités.

Pour chaque incrément, on réalise :

- **l'analyse,**
- **la conception,**
- **l'implémentation**
- **et la validation,**

puis on livre la version du logiciel.

On recommence ainsi jusqu'`a obtenir un logiciel complet.

Modèle du cycle de vie

Modèle du cycle de vie en spirale

Modèle du cycle de vie

Modèle du cycle de vie en spirale

- **Mixte**

Modèle du cycle de vie

Modèle du cycle de vie en spirale

- **Mixte**
- **Planification de la version se fait selon une analyse de risques.**

Modèle du cycle de vie

Modèle du cycle de vie en spirale

- **Mixte**
- **Planification de la version se fait selon une analyse de risques.**
- **Idée est de s'attaquer aux risques les plus importants assez tôt.**

Qualité du logiciel

Qualité du logiciel

En plus du respect de sa spécification, la qualité d'un logiciel dépend des 4 critères :

Qualité du logiciel

En plus du respect de sa spécification, la qualité d'un logiciel dépend des 4 critères :

- ❑ **Maintenabilité**
- ❑ **Robustesse**
- ❑ **Efficacité**
- ❑ **Utilisabilité**

Qualité du logiciel

En plus du respect de sa spécification, la qualité d'un logiciel dépend des 4 critères :

- **Maintenabilité : Peut-on faire évoluer le logiciel ?**
- **Robustesse :**
- **Efficacité :**
- **Utilisabilité :**

Qualité du logiciel

En plus du respect de sa spécification, la qualité d'un logiciel dépend des 4 critères :

- **Maintenabilité : Peut-on faire évoluer le logiciel ?**
- **Robustesse : Le logiciel est-il sujet à des dysfonctionmts ?**
- **Efficacité :**
- **Utilisabilité :**

Qualité du logiciel

En plus du respect de sa spécification, la qualité d'un logiciel dépend des 4 critères :

- **Maintenabilité : Peut-on faire évoluer le logiciel ?**
- **Robustesse : Le logiciel est-il sujet à des dysfonctionmts ?**
- **Efficacité : Le logiciel fait-il bon usage de ses ressources ?**
- **Utilisabilité : Est-il facile à utiliser ?**

Qualité du logiciel

En plus du respect de sa spécification, la qualité d'un logiciel dépend des 4 critères :

- **Maintenabilité : Peut-on faire évoluer le logiciel ?**
- **Robustesse : Le logiciel est-il sujet à des dysfonctionmts ?**
- **Efficacité : Le logiciel fait-il bon usage de ses ressources ?**
- **Utilisabilité : Est-il facile à utiliser ?**

Qualité du logiciel

Historiquement, il y a eu une prise de conscience dans les années 70, appelée la crise du logiciel, dû à un tournant décisif :

Qualité du logiciel

Historiquement, il y a eu une prise de conscience dans les années 70, appelée la crise du logiciel, dû à un tournant décisif :
c'est à cette époque que le coût de construction du logiciel est devenu plus important que celui de la construction du matériel.

Qualité du logiciel

Historiquement, il y a eu une prise de conscience dans les années 70, appelée la crise du logiciel, dû à un tournant décisif :
c'est à cette époque que le coût de construction du logiciel est devenu plus important que celui de la construction du matériel.

Pour répondre à cette crise, on a essayé d'appliquer les méthodes connues de l'ingénieur au domaine du logiciel, pour établir des méthodes fiables sur lesquelles construire une industrie du logiciel.

Qualité du logiciel

Il s’agit de se donner un cadre rigoureux pour :

- **Guider le développement du logiciel, de sa conception à sa livraison.**
- **Contrôler les coûts, évaluer les risques et respecter les délais.**
- **Établir des critères d’évaluation de la qualité d’un logiciel.**

Les spécificités du logiciel

Les spécificités du logiciel

Construction d’un logiciel diffère de celle d’un pont

Les spécificités du logiciel

Construction d'un logiciel diffère de celle d'un pont

- **Modification infime peut avoir des conséquences critique;**

Les spécificités du logiciel

Construction d'un logiciel diffère de celle d'un pont

- **Modification infime peut avoir des conséquences critique;**
- **Progrès tech très rapides peuvent rendre un logiciel caduque ;**

Les spécificités du logiciel

Construction d'un logiciel diffère de celle d'un pont

- **Modification infime peut avoir des conséquences critique;**
- **Progrès tech très rapides peuvent rendre un logiciel caduque ;**
- **Difficile de raisonner sur des programmes ;**

Les spécificités du logiciel

Construction d'un logiciel diffère de celle d'un pont

- **Modification infime peut avoir des conséquences critique;**
- **Progrès tech très rapides peuvent rendre un logiciel caduque ;**
- **Difficile de raisonner sur des programmes ;**
- **Domaines des entrées des logiciels sont trop grands pour test exhaustif ;**

Les spécificités du logiciel

Construction d'un logiciel diffère de celle d'un pont

- **Modification infime peut avoir des conséquences critique;**
- **Progrès tech très rapides peuvent rendre un logiciel caduque ;**
- **Difficile de raisonner sur des programmes ;**
- **Domaines des entrées des logiciels sont trop grands pour test exhaustif ;**
- **Défaillances des prog sont en général dues à des erreurs humaines ;**

Les spécificités du logiciel

Construction d'un logiciel diffère de celle d'un pont

- **Modification infime peut avoir des conséquences critique;**
- **Progrès tech très rapides peuvent rendre un logiciel caduque ;**
- **Difficile de raisonner sur des programmes ;**
- **Domaines des entrées des logiciels sont trop grands pour test exhaustif ;**
- **Défaillances des prog sont en général dues à des erreurs humaines ;**
- **Chaque logiciel a son organisation et sa logique propre ;**

Les spécificités du logiciel

Construction d'un logiciel diffère de celle d'un pont

1. **Modification infime peut avoir des conséquences critique;**
2. **Progrès tech très rapides peuvent rendre un logiciel caduque ;**
3. **Difficile de raisonner sur des programmes ;**
4. **Domaines des entrées des logiciels sont trop grands pour test exhaustif ;**
5. **Défaillances des prog sont en général dues à des erreurs humaines ;**
6. **Chaque logiciel a son organisation et sa logique propre ;**