ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

01
Chapitre

Introduction au génie logiciel

1I2AC1 : Génie logiciel et Conception orientée objet

Régis Clouard, ENSICAEN - GREYC

« Si les ouvriers construisaient les bâtiments
comme les développeurs écrivent leurs programmes,
le premier pic-vert venu aurait détruit toute civilisation. »
Gerald Weinberg

Organisation de l'enseignement

- Volume horaire
  - CM : 11 h
  - TD : 14 h
  - Pas de TP, mais
    - Lien avec le cours « Java et Programmation objet »
    - Katas et Coding dojos
- Plateforme pédagogique (course n°60)
  - Ressources du cours
    - Polycopié par chapitre
    - Diapositives par CM
    - Katas et Coding Dojos

Organisation de l'enseignement

- Examen
  - Le cahier de TD tient lieu d'annales d'examen
  - Il portera sur tout le polycopié (*les diapositives n'en présentent qu'une partie*)
  - Document autorisé : une feuille A4 recto/verso manuscrite
- Discipline
  - Pas d'appel
  - Respect de l'enseignant !

Question du jour

- Qu'est ce que le génie logiciel ?
  - « *Le génie logiciel est une science de génie industriel qui étudie les méthodes de travail et les bonnes pratiques des ingénieurs qui développent des logiciels.* » Wikipedia

Plan du chapitre

1

Pourquoi un cours sur le génie logiciel ?

Pourquoi un cours sur le génie logiciel ?

- La spécialité informatique de l'ENSICAEN forme des **ingénieurs développeurs logiciels**
  - *alias* **Architectes logiciels**
  - Ce sont des professionnels qui
    - conçoivent
    - déploient les systèmes logiciels
    - maintiennent
    - administrent
- Le génie logiciel est un enseignement de base pour les métiers liés au logiciel

Pourquoi un cours sur le génie logiciel ?

- Pourquoi ne pas se contenter d'un cours de programmation pour développer des logiciels ?

Question

- Soit le programme d'addition de deux nombres entiers :

```c
#include <stdio.h>
void main() {
  int nb1, nb2;
  scanf("%d", &nb1);
  scanf("%d", &nb2);
  printf("Résultat = %d\n", nb1 + nb2);
}
```

- Que faut-il ajouter pour en faire un logiciel ?

Confusion programmation / développement

- Analogie avec le génie civil

Confusion programmation / développement

- Programmation (Maçonnerie) : accessible à tous

Confusion programmation / développement

- Programmation (Maçonnerie) : accessible à tous

Confusion programmation / développement

- Programmation (Maçonnerie) : accessible à tous

Confusion programmation / développement

- Développement (Architecture) : réservé aux professionnels

Confusion programmation / développement

- Développement (Architecture) : réservé aux professionnels

Confusion programme / logiciel

- Les profanes créent des **programmes**
- Les développeurs créent des **logiciels**

- Quelle est la différence entre programme et logiciel ?

Différences programme / logiciel

1. Utilisateur
   - Programme : averti et bienveillant
   - Logiciel : « client »
2. Portabilité
   - Programme : un OS
   - Logiciel : tous les OS
3. Complexité
   - Programme : résident sur 1 seul nœud
   - Logiciel : réparti sur le réseau
4. Taille des sources

Différences programme / logiciel : La taille des sources

- Unité de mesure de la taille d'un logiciel :
  - **LOC** : lines of code
    - 1 MLOC : 10^6 LOC → 40 romans épais

Différences programme / logiciel : La taille des sources

- **Programme**
  - < KLOC
- **Logiciel**
  - Commandes de vol A380 : **1 MLOC**
  - OS Android : **15 MLOC**
  - Linux kernel 5.8 (2020): **53 MLOC**
  - Facebook : **62 MLOC**

Conséquence de la taille : Développement en équipe

- La taille des logiciels oblige à un travail en équipe
  - Unité de mesure :
    - **année-homme** (man-year)
    - ou mois-homme
    - ou heure-homme …
  - Par exemple :
    - Algorithme de recherche de Google est estimé à 1 000 années-hommes

Conséquence de la taille : Coût de développement

- Combien de temps pour développer ce bout de code ?
- Combien facturer ce bout de code ?

```
static void sort( int[] array ) {
   for (int i = 0; i < array.length - 1; i++) {
      for (int j = 0; j < array.length – 1 - i; j++) {
         if (array[j] > array[j + 1]) {
             int temp = array[j];
             array[j] = array[j+1];
             array[j+1] = temp;
         }
      }
   }
}
```

Conséquence de la taille : Coût de développement

- Ordre de grandeur :
  - 1 année-homme en France ≈ 1650 h
  - 1h ingénieur ≈ 50 €
  - Productivité ≈ 2 à 5 LOC/h (4 à 9 KLOC /an)
- Donc, le code coûte :
  - Temps : 1 heure-homme
  - Prix : 50 €

```
static void sort( int[] array ) {
   for (int i = 0; i < array.length - 1; i++) {
      for (int j = 0; j < array.length – 1 - i; j++) {
         if (array[j] > array[j + 1]) {
             int temp = array[j];
             array[j] = array[j+1];
             array[j+1] = temp;
         }
      }
   }
}
```

Différences programme / logiciel

1. Utilisateur
2. Portabilité
3. Complexité
4. Taille des sources
5. Responsabilité des dysfonctionnements

Différences programme / logiciel : La responsabilité des dysfonctionnements

- **Programme**
  - L'utilisateur accepte les conséquences des dysfonctionnements lors de l'utilisation du programme
- **Logiciel**
  - Les développeurs sont tenus responsables par les utilisateurs des conséquences néfastes de l'utilisation du logiciel.
  - Les développeurs doivent proscrire toutes les conséquences néfastes de l'utilisation du logiciel :
    - Perte de données
    - Résultats erronés
    - Vol de données
    - Utilisation frauduleuse

Conséquence comique d'un dysfonctionnement

- Windows 95

Conséquence préoccupante d'un dysfonctionnement

- Bug sur les serveurs Microsoft Exchange. Les utilisateurs n'avaient plus accès à leur mail suite au changement d'année 2022 : le moteur d'analyse des malwares qui peuvent être contenus dans les mails est à l'origine de ce problème.
  - Codage des dates sur un entier signé (31 bits)
  - Or 2^31 = 2 147 483 648 ne peut pas coder les dates en 2022 qui ont une valeur minimale de 2 201 010 001

Conséquence tragique d'un dysfonctionnement

- Mort tragique d'une patiente de 72 ans à l'hôpital de Versailles en novembre 2011
  - Son allergie à un antibiotique, l'amoxicilline, était bien notée dans son dossier médical, mais le logiciel utilisé pour les prescriptions n'a pas intégré cette donnée.

Différences programme / logiciel

1. Utilisateur
2. Portabilité
3. Complexité
4. Taille des sources
5. Responsabilité des dysfonctionnements
6. Maintenance

Différences programme / logiciel : Maintenance

- Un logiciel ne s'use pas … mais il se détériore
  - Pas de logiciel sans maintenance
  - Sans maintenance un logiciel a une durée de vie estimée de 6 mois
  - La maintenance doit être préparée dès la conception ; pour le pont de Maillau, le remplacement des haubans est prévu dès la conception.

Conclusion préliminaire

- Un programme peut se réaliser de façon empirique
- Un logiciel ne peut pas se développer sans une méthode de gestion de projet et un haut niveau d'expertise reposant sur le génie logiciel
  - Projet étalé sur plusieurs mois
  - Projet nécessitant la collaboration de plusieurs personnes
  - Tâche complexe et compliquée
  - La programmation ne représente qu'une très petite partie du logiciel
- Développeur logiciel :
  - Un métier à haut niveau d'expertise
  - Un métier qui s'apprend
  - L'auto-formation donne des programmeurs

Plan du chapitre

1

Pourquoi un cours sur le génie logiciel ?

2

Création du génie logiciel

Génie logiciel (1968)

- Le terme anglais « *software engineering* » a été inventé par une pionnière du génie logiciel : **Margaret Hamilton**
  - Responsable de la partie logicielle embarquée du projet Apollo de la NASA
  - Le premier ingénieur en génie logiciel est une ingénieure

Margaret Hamilton

Génie logiciel (1968)

- Génie
  - Ensemble de pratiques **régulées** et basées sur des principes **scientifiques** et **économiques** (+ **RSE** et **DD**)

➡ Génie logiciel

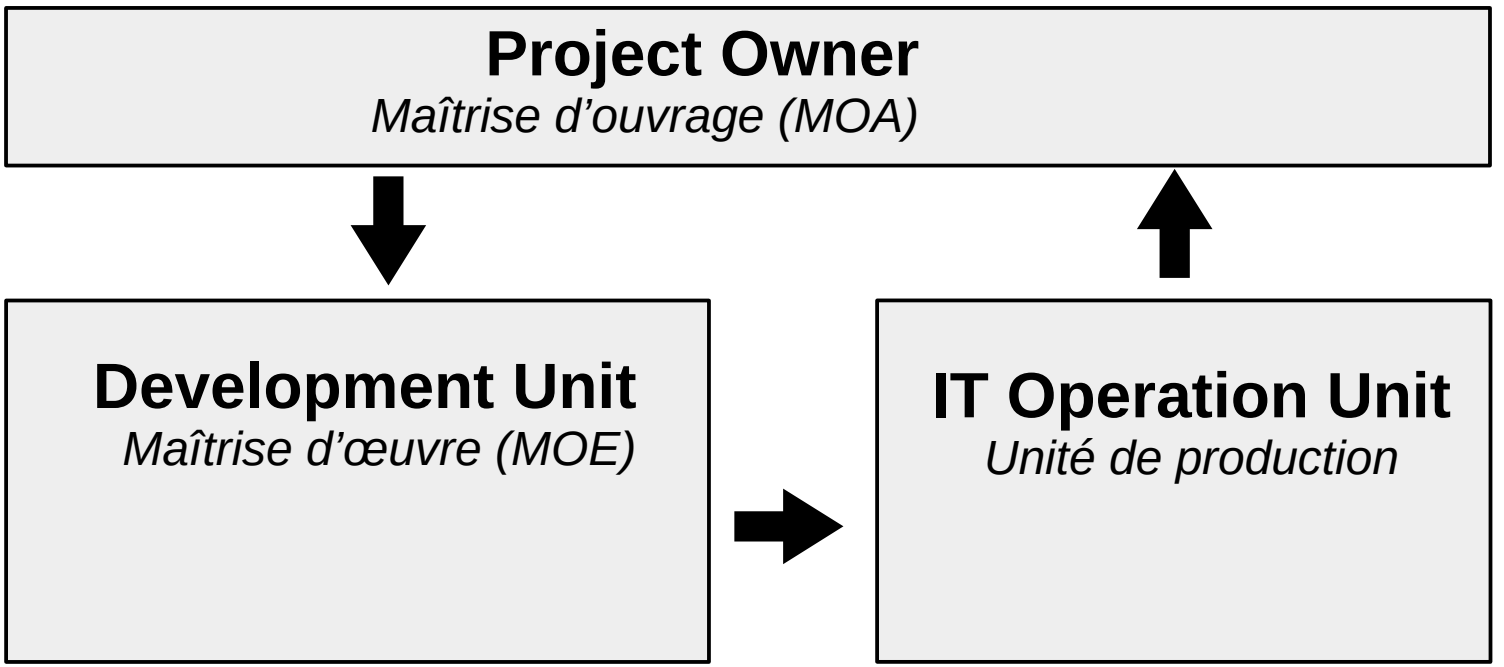
  - Le génie logiciel postule donc que l'on peut appliquer le génie au logiciel

- Source d'inspiration : le **génie civil**
  - Le génie civil a démontré toute son efficacité depuis des millénaires

Génie civil

- La gestion de projet selon le génie civil :
  - Découper le temps du projet en une suite d'étapes séquentielles
    - *Livrable : diagramme de Gantt / Pert*
  - Ne faire qu'une seule chose à la fois à chaque étape
    - *Métiers spécifiques*
  - Récolter les besoins avant de les réaliser
    - *Livrable : cahier des charges*
  - Bien réfléchir avant d'agir
    - *Livrable : documentation de conception et de réalisation*

Écosystème du développement logiciel

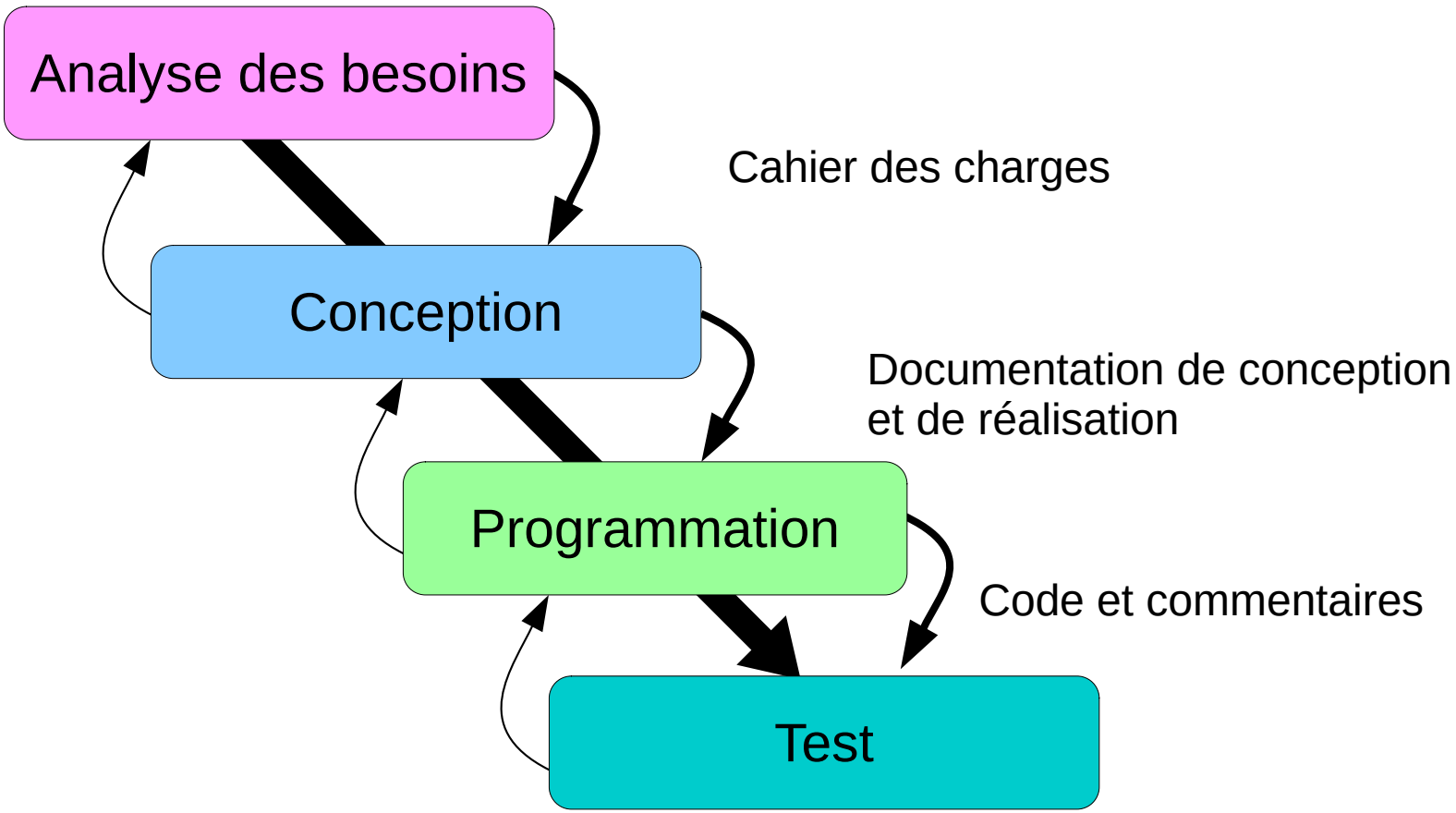


Génie logiciel

- Les éléments de base du génie civil appliqué au génie logiciel :
  - **Une méthode de gestion de projet**
    - Planifier une organisation du temps pour le travail en équipe
  - **Un paradigme de conception**
    - Fournir les briques de base de la conception et les mécanismes pour les assembler
  - **Un formalisme de modélisation**

    Fournir un langage formel pour parler du code de manière du code abstraite et non ambiguë

1/ Une méthode : Le cycle en cascade



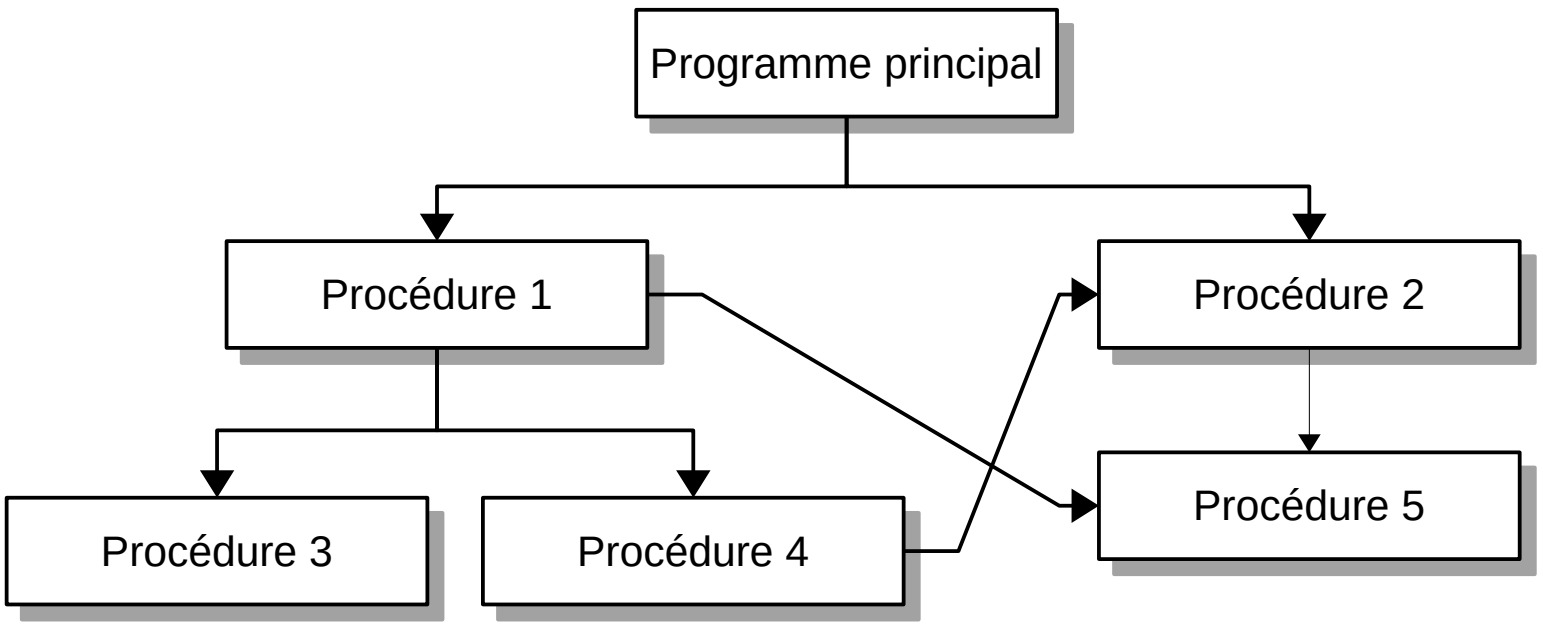
Ingénierie logicielle : les métiers

- Il en résulte deux métiers dans la MOE :
  1. **Analyste (ingénieur)**
     - Analyse des besoins
     - Spécification fonctionnelle
     - Conception générale et détaillée

     } Rédaction du cahier des charges

     } Rédaction de la documentation de conception et de réalisation
  2. **Programmeur (technicien)**
     - Écriture du code
     - Écriture des tests
     - Mise en production

2/ Un paradigme : La conception procédurale



3/ Un formalisme : L'algorithmique

INSERTION-SORT(A)	*cost*	*times*
1 **for** $j = 2$ **to** $A.length$	c_1	n
2 $key = A[j]$	c_2	$n-1$
3 // Insert $A[j]$ into the sorted sequence $A[1..j-1]$.	0	$n-1$
4 $i = j-1$	c_4	$n-1$
5 **while** $i > 0$ and $A[i] > key$	c_5	$\sum_{j=2}^{n} t_j$
6 $A[i+1] = A[i]$	c_6	$\sum_{j=2}^{n} (t_j - 1)$
7 $i = i-1$	c_7	$\sum_{j=2}^{n} (t_j - 1)$
8 $A[i+1] = key$	c_8	$n-1$

Ada Lovelace
vers 1843

- Le premier programmeur est une programmeuse : Ada Lovelace
  - Algorithme de calcul des nombres de la suite de Bernoulli $\sum_{k=0}^{n-1} k^m$

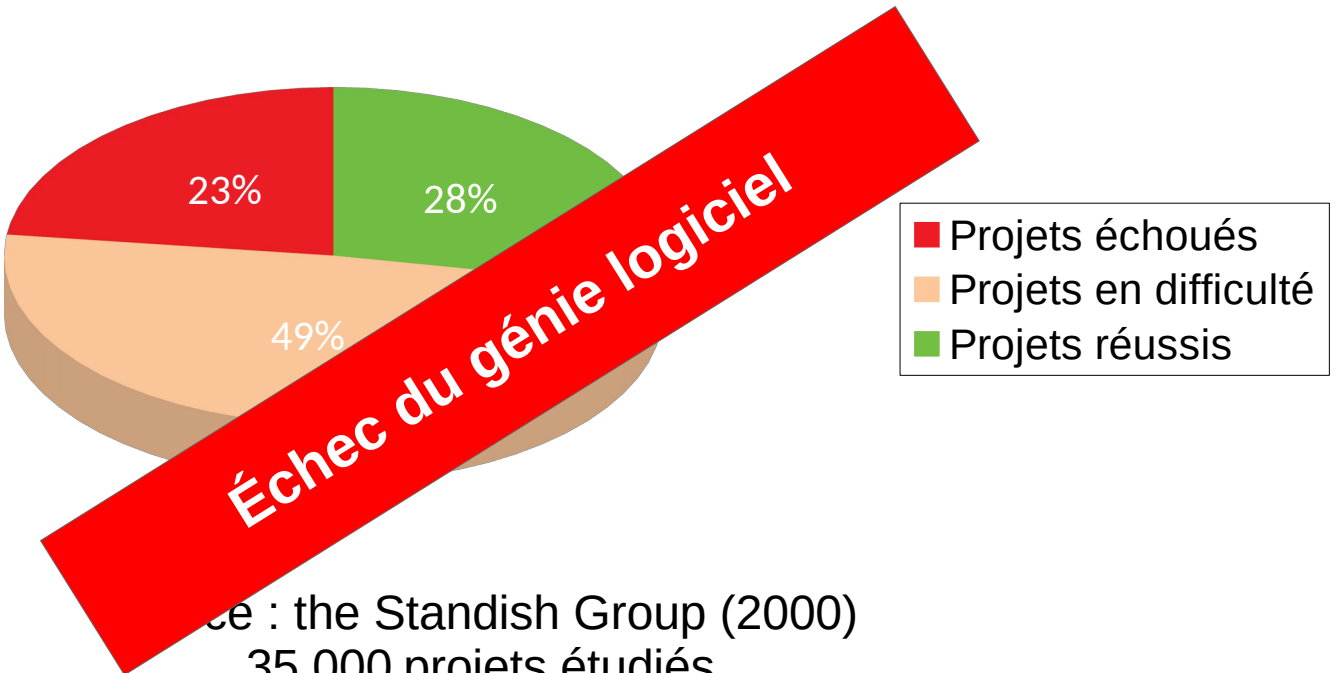
Plan du chapitre

1
Pourquoi un cours sur le génie logiciel ?

2
Création du génie logiciel

3
Bilan

Bilan en 2001

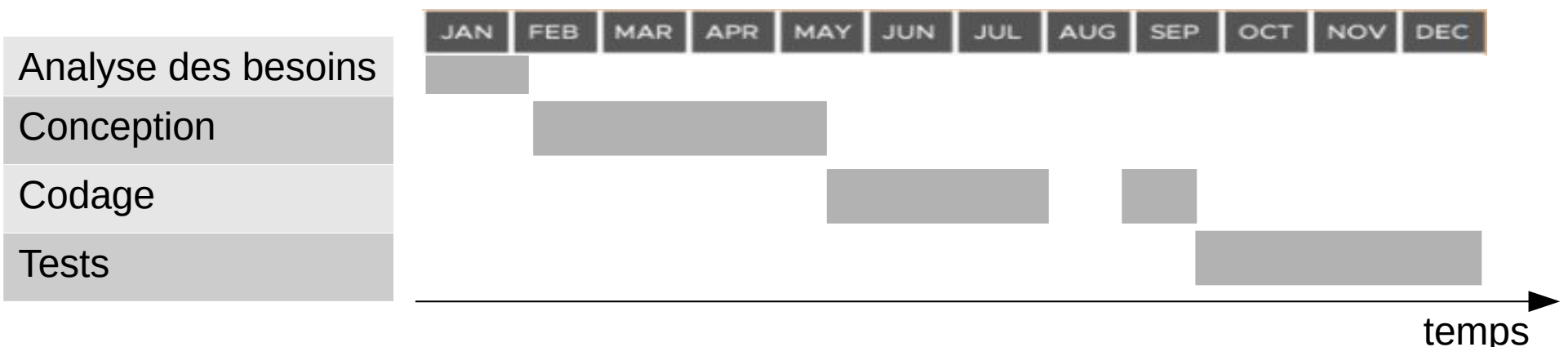


ce : the Standish Group (2000)
35.000 projets étudiés

Causes de l’échec

Cause d'échec n°1 :
Suivre un plan coûte que coûte

- Le cycle en cascade définit des étapes séquentielles précises
  - Engagement sur plusieurs mois
    - cf. diagramme prévisionnel de Gantt



  - Conséquences
    - La moindre modification du planning met le projet en danger

« Bon, les gars, il y a eu un accident au quai.
Une petite auto blanche est tombée à l'eau.
Qu'est ce qu'on fait ? »

« Pas de problème, on va appeler une grue
pour remonter l'automobile. »

« Ça va très bien, la voiture est toute petite,
dans une demi-heure tout est terminé."

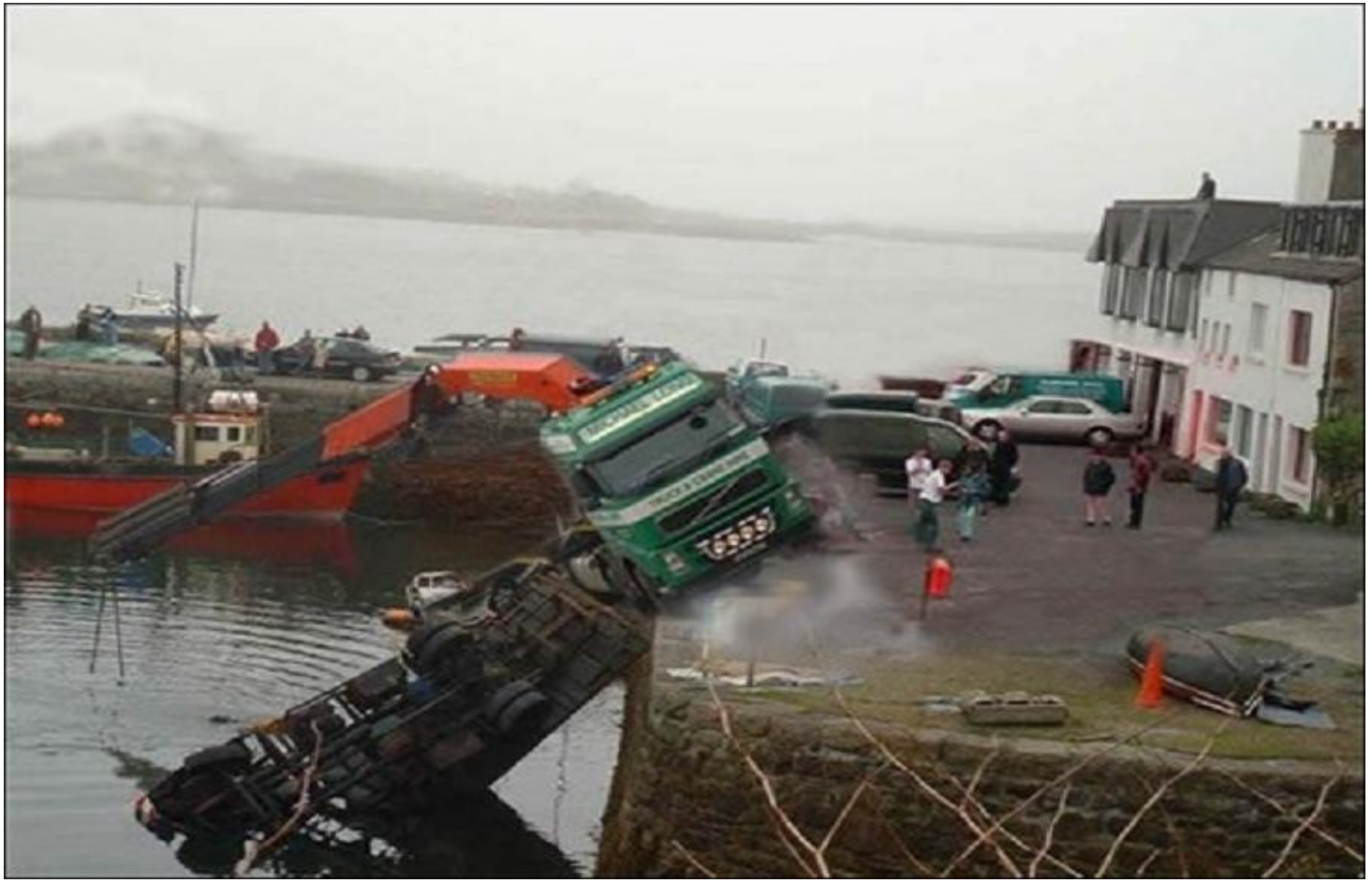
« ARGH!!!! La grue est tombée aussi à l'eau avec la petite voiture ! »

« Ça va vraiment mal.
Et là, qu'est ce qu'on fait ? »

« Pas de problème, on a appelé une autre grue, une GROSSE celle-là. »

« Bon, la petite automobile est déjà sortie,
on aurait du appeler cette grue la première fois... »

« Il ne reste qu'à sortir la petite grue maintenant, dans une demi-heure tout devrait être fini. »



« ARGHHHHHH !!! ENCORE !!!!!!!!! »

« Pas de problème, on a appelé une autre grue, plus grosse encore.. »

Cause d'échec n°2 :
Estimer la durée d'un projet

- Déterminer le nombre d'années-hommes est extrêmement difficile
  - Trop d'aléas et d'incertitudes
  - Pas de règles et pas de mesures systématiques

Cause d'échec n°2 : Estimer la durée d'un projet

- Mythe de l'**année-homme**, cf. «The Mythical Man-Month», Fred Brooks, 1995
  - Non-linéarité de la charge de travail (source *Borland Software Corporation*) :

Taille équipe	Productivité par personne (KLOC/année)	Productivité de l'équipe (KLOC/année)	Gain
1	15.0	15.0	
2	11.9	23.8	**1.6**
10	7.0	69.6	**4.6**
25	5.1	128.2	**8.5**

  - Certaines tâches ne sont pas parallélisables :
    - « Neuf femmes ne font pas un enfant en un mois »

Cause d’échec n°2 :
Estimer la durée d’un projet

- Rattrapage du retard
  - Contrairement à l’intuition, ajouter des personnes à une équipe d'ingénieurs ne permet pas de rattraper le retard, au contraire

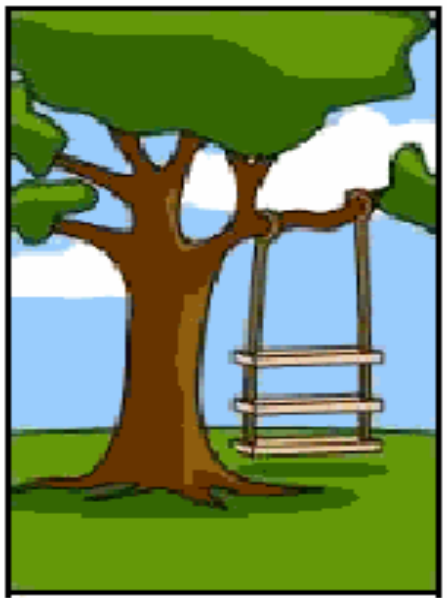
Estimer la durée d'un projet :
Une règle empirique

1. *Estimer honnêtement le temps de développement*
2. *Multiplier par 3*
3. *Passer à l'unité supérieure*

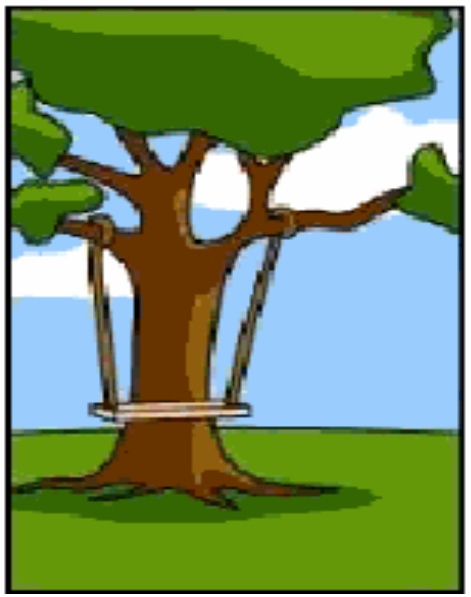
Cause d’échec n°3 :
Définir les besoins au début

- Les clients ne savent pas identifier exactement ce qu’ils veulent
- Les clients ne savent pas exprimer clairement les besoins identifiés
- Les clients changent leurs besoins au cours du projet

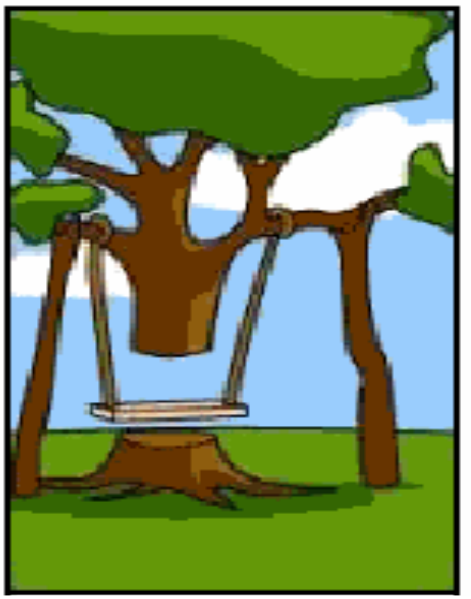
http://projectcartoon.com/cartoon/2



Ce que le client explique

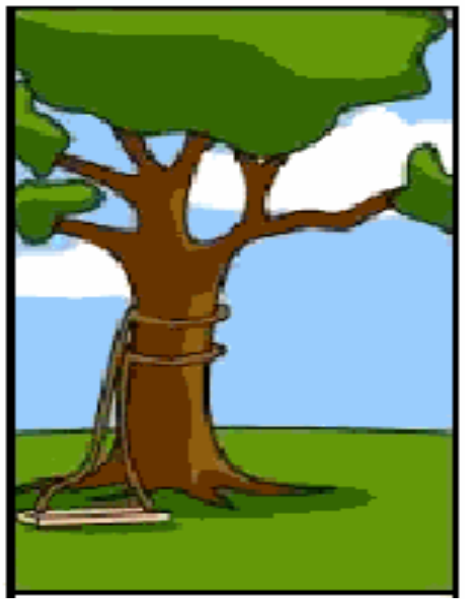


Ce que le chef de projet comprend



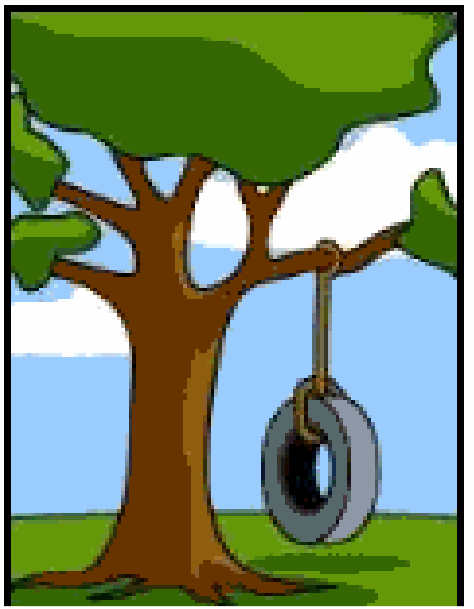
Ce que les analystes modélisent

http://projectcartoon.com/cartoon/2



Ce que les développeurs programment

Ce que les commerciaux vendent



Ce que le client a réellement besoin

Cause d'échec n°4 :
Réaliser les tests à la fin

- Les tests sont généralement la variable d'ajustement du temps
  - Les tests sont faits selon le temps restant (ou pas)
- Pourtant, il est impossible de garantir des logiciels sans défaut
  - Estimation : 1 à 10 bugs / KLOC
    - Windows 10 (80 MLOC) → 80 000 bugs !

Cause d'échec n°5
Raisonner au niveau procédure

- Le paradigme procédurale est inadapté à la conception de programmes de très grande taille
  - Programmer au niveau procédural est une activité complexe qui revient à construire une voiture au niveau moléculaire ; le niveau atomique correspondrait à l'assembleur

Constat préliminaire

- La logiciel se prête mal aux méthodes du génie civil
  - Suivre un plan élaborer au début du projet
  - Estimer la durée des étapes d'un projet
  - Définir les besoins au début
  - Réaliser les tests à la fin
- Le logiciel n'est pas le matériel : il est mou
  - On peut construire une tour de 200 m en commençant par un bungalow
  - On peut construire un bungalow en commençant par la salle de bain
  - On peut une salle de bain en commençant par le carrelage
- Il doit donc être possible de repenser le génie logiciel en s'émancipant du génie civil

Plan du chapitre

1
Pourquoi un cours sur le génie logiciel ?

2
Première génération du génie logiciel

3
Bilan : échec

4
Le génie logiciel aujourd'hui

Révolution du génie logiciel (2001)

- Nouvelle définition du génie logiciel : **Agilité**

Le génie logiciel est l'art et l'ensemble des moyens techniques, industriels et humains qu'il faut réunir pour construire, distribuer et maintenir des logiciels de qualité

- Ajoute une dimension **artisanale** au développement (cf ***software craftsmanship***) en prônant qu'il ne suffit pas qu'un logiciel soit fonctionnel, mais il faut qu'il soit **bien conçu**
- Le développeur travaille comme un artisan, guidé par son talent, son expérience et ses connaissances théoriques puisées dans le **partage** et la **formation**
- Le développeur s'appuie sur l'expérience des développeurs expérimentés. Il existe aujourd'hui près de 60 ans d'expérience en génie logiciel dont il est nécessaire de s'inspirer

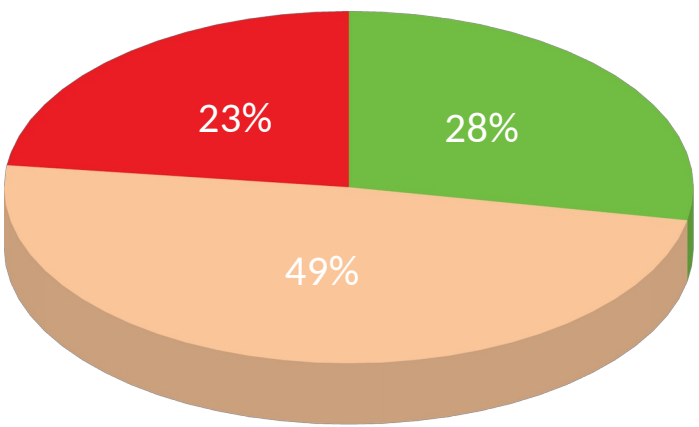
Le génie logiciel aujourd'hui

- Nouvelle méthode de gestion de projet
  - **Cycle itératif et incrémental**
- Nouveau paradigme de conception logicielle
  - **Conception orientée objet**
- Nouveau formalisme de modélisation
  - **UML**
- Ne reste qu'un seul métier : développeur logiciel

- Ces points sont développés dans la suite du cours

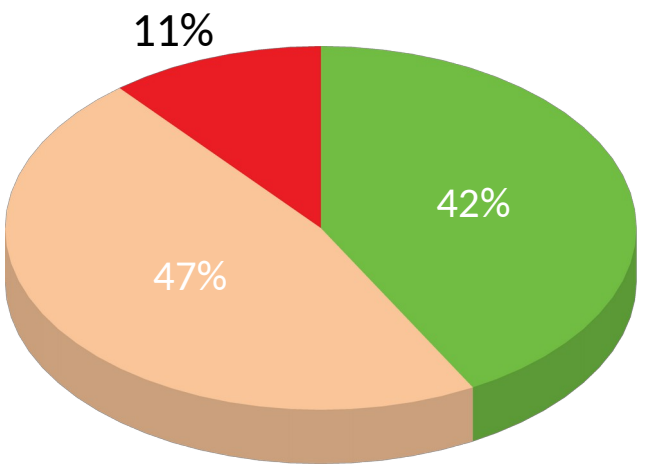
Effet sur la réussite des projets

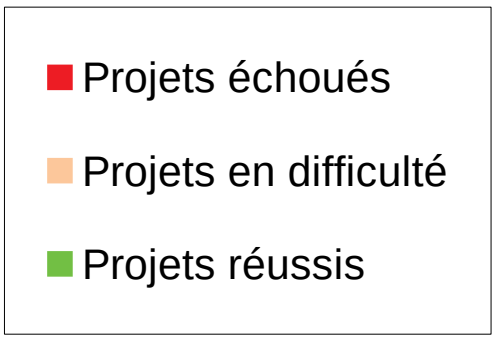
Rappel 2000



(35.000 projets étudiés)

2020





(10.000 projets étudiés)

Source : the Standish Group