

# GÉNIE LOGICIEL

(SOFTWARE ENGINEERING)

## 1ÈRE PARTIE – INTRODUCTION

*Faculté des Sciences et Techniques*

**<http://perso.univ-st-etienne.fr/jacquene/gl/>**

Francois.Jacquenet@univ-st-etienne.fr

# Plan de cette partie du cours

2/52

- Définition du génie logiciel
- Quelques grands bugs de l'histoire
- Les protagonistes du génie logiciel
- La qualité du logiciel
- La dimension éthique dans le génie logiciel
- Ce que l'on verra ensuite

# Génie logiciel (définition)

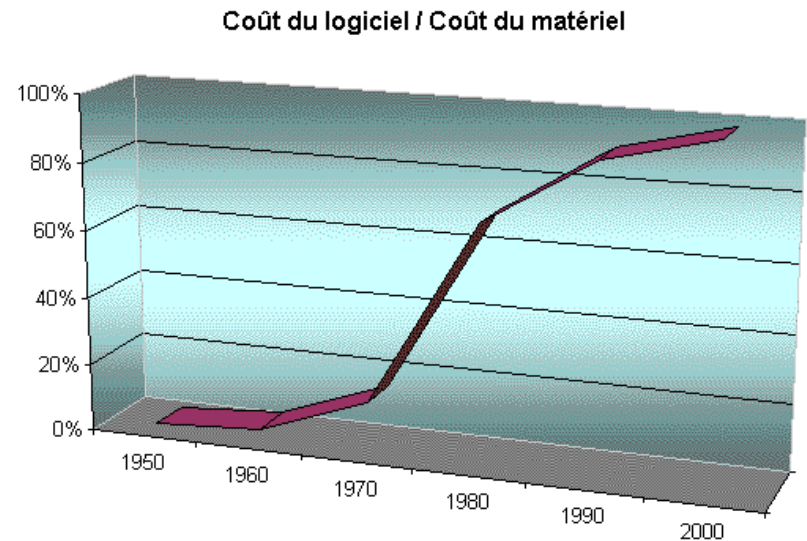
3/52

## □ Crise du logiciel

- Dès 1968
- Coût du logiciel > coût du matériel
- ONU réunit les spécialistes de la programmation
- Titre de la conférence : **Software Engineering**
- Constatations :
  - Projets en retard
  - Budgets dépassés
  - Informatisations manquées
  - Systèmes monstrueux
  - Graves erreurs de fonctionnement

## □ → Améliorer

- La qualité des logiciels
- Leur mode de production



# Génie logiciel (définition)

4/52

## □ De nos jours

- ▣ Les économies de toutes les nations développées dépendent du logiciel
- ▣ De plus en plus de systèmes sont contrôlés par des logiciels
- ▣ Les dépenses de logiciels représentent une part significative du PIB des nations développées
- ▣ Coût du logiciel > coût du matériel
- ▣ Les logiciels coûtent plus cher à maintenir qu'à développer. Systèmes à longue vie → la maintenance coûte plusieurs fois le coût du développement.

# Génie logiciel (définition)

5/52

## □ La crise du logiciel continue

### □ De 1965 à 1995 :

- le volume de chaque logiciel a été multiplié 100
- la productivité des développeurs a été multipliée par 3

### □ Développement de Microsoft Exchange Server (en 1995)

- 1 000 années x hommes pour 7 millions de lignes.
- Productivité de 30 lignes par homme par jour.

### □ Développement de Windows 2000

- 5 000 ingénieurs pendant 3 ans pour réécrire 70% des 16 millions de lignes de code de Windows NT.
- Productivité = 4,8 lignes par homme par jour.

### □ Pourtant un bon hacker, motivé, seul, avec les bons outils, sans se préoccuper d'un cahier des charges ou de relation client, peut écrire 3 000 lignes dans la même journée...

# Génie logiciel (définition)

6/52

## Définitions

- Génie logiciel  $\approx$  Programmation
  - ▣ Fait plus d'effet sur ses supérieurs
  - ▣ Fait plus d'effet au restaurant
  - ▣ Permet de demander une augmentation
- Evidemment faux mais :
  - ▣ Rappelle que la distance entre génie logiciel et programmation n'est pas aussi grande qu'on veut le croire
- Autre définition :
  - ▣ Tout ce qui concerne les circonstances spécifiques de la production industrielle de logiciels
    - Gestion de grands projets
    - Gestion de configurations
    - Mesures et modèles de coûts
    - ...
  - ▣ Insuffisant car exclue ce qui se rapporte à la programmation

# Génie logiciel (définition)

7/52

- Définition du terme « Logiciel » :
  - ▣ On regroupe sous le terme de **logiciel** les **différentes formes de programmes** qui permettent de faire fonctionner un ordinateur, et de l'utiliser pour résoudre des problèmes, **les données** qu'ils utilisent et **les documents** qui servent à concevoir ces programmes et ces données, à les mettre en œuvre, à les utiliser et à les modifier.
- Fait bien ressortir :
  - ▣ Programmes
  - ▣ Données
  - ▣ Documentation

# Génie logiciel (définition)

8/52

- Produits logiciels génériques
  - ▣ Systèmes autonomes commercialisés auprès de tout client souhaitant les acheter
  - ▣ Exemples : office, photoshop, jeux, ...
- Produits logiciels adaptés aux clients
  - ▣ Logiciels commandés par des clients spécifiques pour répondre à leurs propres besoins
  - ▣ Exemples : systèmes de contrôle embarqués, logiciel de contrôle de trafic aérien, ...



# Génie logiciel (définition)

9/52

- Différence au niveau de la spécification :
  - ▣ Produits génériques
    - La spécification de ce que doit faire le logiciel est maîtrisée par le développeur et les décisions concernant les changements du logiciel sont faites par le développeur
  - ▣ Produits adaptés aux clients
    - La spécification de ce que doit faire le logiciel est maîtrisée par le client de ce logiciel et c'est lui qui décide des changements qu'il souhaite y apporter.

# Génie logiciel (définition)

10/52

- Applications indépendantes
  - ▣ Applications fonctionnant sur un ordinateur local et qui possède toute les fonctionnalités et ne nécessite donc pas de se connecter à d'autres systèmes
- Applications interactives à base de transactions
  - ▣ Applications qui s'exécutent sur une machine distante accédée par l'utilisateur depuis sa machine (sites Web, e-commerce, ...)
- Systèmes de contrôle embarqués
  - ▣ Logiciels de contrôlent de dispositifs matériels (contrôle missiles, fusées, ...)

# Génie logiciel (définition)

11/52

- Applications batch
  - ▣ Principalement des applications de gestion destinées à traiter de gros volumes de données pour produire de gros volumes de résultats en sortie (gestion des comptes bancaires)
- Logiciels de jeux
- Systèmes de modélisation et de simulation
  - ▣ Systèmes développés par les scientifiques pour modéliser des processus physiques (météo, écoulement de fluide sur les ailes d'avions, ...)
- Systèmes de collectes de données
  - ▣ Systèmes qui collectent des données à l'aide de capteurs et les transmettent à d'autres systèmes pour traitement

# Génie logiciel (définition)

12/52

- Définition du terme « Génie Logiciel » :
  - ▣ On appelle génie logiciel l'application de méthodes scientifiques au développement de **théories, méthodes, techniques, langages** et **outils** favorisant la production de logiciels de **qualité**.
- Eléments essentiels
  - ▣ Présence d'une base scientifique
  - ▣ 5 directions de développement fondamentales, de la plus abstraite (théorie) à la plus pratique (outils)
  - ▣ Notion de **qualité**
- Reste toutefois très générale
- Limite le génie logiciel aux activités de deuxième niveau
  - ▣ Développer un outil de construction de prog : GL
  - ▣ Développer un simple logiciel d'application : pas GL

# Génie logiciel (définition)

13/52

- Définition par l'IEEE du terme « Génie Logiciel » :
  - ▣ L'application d'approches **systematiques, rigoureuses, quantifiables** pour le développement, la mise en œuvre et la maintenance d'un logiciel, c'est à dire **l'application de l'ingénierie** au domaine du logiciel.
- Culture générale :
  - ▣ ACM : <http://www.acm.org>
  - ▣ IEEE : <http://www.ieee.org>

# Importance du Génie logiciel

14/52

- Les vies de plus en plus d'individus et de sociétés reposent sur des systèmes logiciels évolués
  - ▣ ➔ produire des logiciels fiables
  - ▣ ➔ produire des logiciels en qui l'on puisse avoir confiance
  - ▣ ➔ produire des logiciels rapidement
  - ▣ ➔ produire des logiciels avec des coûts maîtrisés

# Importance du Génie logiciel

15/52

- ☐ Approche GL vs Approche amateur ?
- ☐ Plus facile de travailler dans son coin qu'en groupe ?
- ☐ Plus facile de bidouiller ?
- ☐ A long terme et sur des projets d'envergure :  
**NON**

# Quelques questions fréquentes à propos du génie logiciel

16/52

Question	Réponse
Qu'est-ce qu'un logiciel ?	Programmes, documentation et données. Des produits logiciels peuvent être développés pour un client particulier ou pour un marché plus global.
Quelles sont les qualités d'un bon logiciel ?	Un bon logiciel doit posséder les fonctionnalités et performances attendues du client. Il doit aussi être maintenable, facilement utilisable, ... (voir section qualité du logiciel).
Qu'est-ce que le génie logiciel ?	Le génie logiciel est une discipline d'ingénierie concernant tous les aspects de la production du logiciel.
Quelles sont les activités fondamentales du génie logiciel ?	Spécification, développement, validation et évolution.



# Quelques questions fréquentes à propos du génie logiciel

17/52

Question	Réponse
Quels sont les challenges auxquels doit faire face le génie logiciel ?	Diversité grandissante, demande permanente pour des temps et coûts de production réduits tout en maintenant des niveaux de fiabilité admissible.
Quels sont les principaux coûts du génie logiciel ?	60% des coûts pour le développement 40% des coûts pour les tests. Pour les logiciels clients les coûts d'évolution dépassent souvent les coûts de développement.
Quelles sont les meilleures méthodes et techniques du génie logiciel ?	Bien que tous les projets logiciels doivent être gérés et développés avec professionnalisme, les techniques sont adaptées aux types de systèmes. Par exemple, pour les jeux : prototypes, pour les systèmes critiques embarqués : méthodes formelles..
Quel a été l'apport du Web dans le génie logiciel ?	Disponibilité de services Web et possibilité de développer des système hautement distribués à base de services. Retombées importantes en terme de langages de programmation et de réutilisation du logiciel.

# Quelques grands bugs de l'histoire

18/52



Yaahooo !!! J'ai réussi à faire planter  
le système de la tour de contrôle !!!

# Quelques grands bugs de l'histoire

19/52

Panne du site <http://www.france.fr> le 14 juillet 2010 dû soit disant à un trop grand nombre de connexions (25 000)...

Panne du site Web <http://www.voyages-sncf.com> le 20/11/2008 pendant plusieurs heures suite à une « mise en ligne d'une nouvelle version du site »...

Perte de Mars Climate Orbiter, le 23/09/1999 après 9 mois de voyage.

- ▣ Coût : 120 M\$
- ▣ Cause : confusion entre pieds et mètres.

# Quelques grands bugs de l'histoire

20/52

- Explosion lors du premier lancement de la fusée Ariane V le 4 juin 1996
  - ▣ logiciel de plate forme inertielle repris tel quel d'Ariane IV sans nouvelle validation.
  - ▣ Ariane V ayant des moteurs plus puissants s'incline plus rapidement que Ariane IV, pour récupérer l'accélération due à la rotation de la Terre.
  - ▣ Les capteurs ont bien détecté cette inclinaison d'Ariane V, mais le logiciel l'a jugée non conforme au plan de tir (d'Ariane IV), et a provoqué l'ordre d'auto destruction.
  - ▣ En fait tout se passait bien...
  - ▣ Coût du programme d'étude d'Ariane V : 38 milliards de Francs, pour 39 secondes de vol après 10 années de travail....

# Quelques grands bugs de l'histoire

21/52

- Destruction de la sonde Mariner 1 le 27/07/1962 pour cause d'une erreur d'un caractère dans un programme Fortran (coût – 80 millions de dollars)
- Au passage de l'équateur, un F16 se retrouve sur le dos : changement de signe de la latitude mal compris...
- Le 22/12/2001 750 000 terminaux de paiement ne répondent plus → longues files d'attente
  - Saturation des serveurs de la société Atos chargés des autorisations de paiements dépassant 600F. Habituellement quelques dizaines de secondes, ce jour là : 30mn.
  - Des clients abandonnent leurs chariots pleins. Le groupe Leclerc a chiffré son préjudice à 2 millions d'euros.

# Quelques grands bugs de l'histoire

22/52

## □ Bug de l'an 2000



# Le développement peut être un échec

23/52

- ❑ Malgré de nombreux efforts, les résultats sont souvent négatifs
  - ❑ Dépassement de budgets
  - ❑ Dépassement de temps ou en inadéquation avec les besoins du marché
  - ❑ Le logiciel ne satisfait pas les besoins définis du client
  - ❑ Moins bonne qualité que prévue
  - ❑ Les performances ne correspondent pas aux attentes
  - ❑ Le logiciel est trop difficile à utiliser

# Le développement peut être un échec

24/52

- Raisons expliquant ces conséquences négatives
  - ▣ Il n'y a pas que les erreurs de programmation !!!
  - ▣ Projet non réaliste
  - ▣ Management de projet de mauvaise qualité
  - ▣ Mauvaise estimation des ressources
  - ▣ Mauvaise définition des besoins du système
  - ▣ Mauvais reporting du status du projet
  - ▣ Risques non gérés
  - ▣ Mauvaise communication entre les clients, développeurs, utilisateurs
  - ▣ Inaptitude à gérer la complexité du projet
  - ▣ Mauvaise méthodologie de conception
  - ▣ Mauvais outils de développement
  - ▣ Mauvaise méthodologie de test
  - ▣ Mauvaise couverture des tests
  - ▣ Processus de développement inapproprié



# Les protagonistes du génie logiciel (les quatre P)

25/52

La production de logiciels est une tâche extrêmement complexe qui posent de nombreux challenges.

- Coordination de nombreuses **personnes** (organisées en équipes)
- Objectif premier = construire un **produit** (qui répond aux besoins du/des client(s)).
- Effort de développement doit être organisé en **projet** (avec un planning rigoureux pour assurer le succès).
- Pour développer le produit avec succès, les activités des personnes doivent être organisées suivant un **processus** ordonné bien défini.

# Les protagonistes du génie logiciel

26/52

## Personnes

- Business management
- Equipe de management du Projet
- Equipe de développement
- Clients
- Utilisateurs finaux

# Les protagonistes du génie logiciel

27/52

## Produit

- Bien plus que le code source
- Documentation produite durant la définition et le développement
- Code source et objet
- Document des tests (plans, cas, résultats)
- Documentation utilisateurs
- Diverses mesures de productivité

# Les protagonistes du génie logiciel

28/52

## Projet

- Les diverses activités d'un projet
  - ▣ Planification
  - ▣ Analyse des besoins
  - ▣ Conception
  - ▣ Implémentation
  - ▣ Tests
  - ▣ Maintenance
- Paradigmes utilisés (objets, ...)

# Les protagonistes du génie logiciel

29/52

## Processus

- Façon dont vont se succéder les diverses activités d'un projet
- *(on en reparlera dans la partie du cours concernant le processus de développement d'un logiciel)*

# La qualité du logiciel

30/52

## □ Distinction entre

- **Facteurs de qualité** : caractéristiques externes de la qualité (perçues par les utilisateurs du produit – exemple : fiabilité)
- **Critères de qualité** : caractéristiques internes de la qualité (peuvent être analysées par les informaticiens – exemple : modularité)
- Pour l'utilisateur, seuls les facteurs comptent
- Mais le respect de critères au cours du développement conditionne la production d'un logiciel conforme aux facteurs de qualité.

# La qualité du logiciel

31/52

## □ 10 facteurs de qualité

### □ Validité

- Aptitude d'un produit logiciel à remplir exactement ses fonctions, définies par le cahier des charges et la spécification

### □ Fiabilité

- Aptitude d'un produit logiciel à fonctionner dans des conditions éventuellement anormales

### □ Extensibilité

- Facilité avec laquelle un logiciel se prête à une modification ou à une extension des fonctions qui lui sont demandées

### □ Réutilisabilité

- Aptitude d'un logiciel à être réutilisé, en tout ou partie, dans de nouvelles applications

### □ Compatibilité

- Facilité avec laquelle un logiciel peut être combiné avec d'autres

# La qualité du logiciel

32/52

## □ 10 facteurs de qualité (suite)

### ▣ Efficacité

- Utilisation optimale des ressources matérielles

### ▣ Portabilité

- Facilité avec laquelle un produit peut être transféré dans différents environnements matériels et logiciels

### ▣ Vérifiabilité

- Facilité de préparation des procédures de recette et de validation

### ▣ Intégrité

- Aptitude d'un logiciel à protéger son code et ses données contre des accès non autorisés

### ▣ Facilité d'emploi

- Facilité d'apprentissage, d'utilisation, de préparation des données, de correction des erreurs d'utilisation, d'interprétation des résultats



# La qualité du logiciel

33/52

- ➔ on va chercher à maximiser ces facteurs de qualité
- Mais ce n'est pas aussi simple ➔ Expérience de Weinberg
  
- 5 équipes
- Un même projet
- Des objectifs différents pour chaque équipe
  - ▣ Optimiser l'effort pour réaliser le logiciel
  - ▣ Optimiser le nombre d'instruction
  - ▣ Optimiser l'espace mémoire
  - ▣ Optimiser la clarté du programme
  - ▣ Produire des sorties les plus clairs possibles

# La qualité du logiciel

34/52

## □ Résultats de l'expérience de Weinberg

Objectif : optimiser	Classement pour la tâche				
	effort	Nombre instructions	Espace mémoire	Clarté programme	Clarté sorties
effort	<b>1</b>	4	4	5	3
Nombre instructions	2-3	<b>1</b>	2	3	5
Espace mémoire	5	2	<b>1</b>	4	4
Clarté du programme	4	3	3	<b>2</b>	2
Clarté des sorties	2-3	5	5	1	<b>1</b>

# La qualité du logiciel

35/52

## Conclusions de l'expérience de Weinberg

- Si l'on fixe des objectifs clairs à des programmeurs, ils sont très motivés pour les atteindre
- Mais, des objectifs différents peuvent entrer en conflit

# La dimension éthique dans le génie logiciel

36/52

- Le génie logiciel induit des responsabilités plus larges que simplement appliquer des compétences techniques
- Les ingénieurs logiciels doivent se comporter d'une manière honnête et éthique s'ils veulent être respectés professionnellement
- Un comportement éthique est bien plus que de respecter la loi, il implique de suivre un ensemble de principes moralement corrects

# Responsabilité professionnelle

37/52

## □ Confidentialité

- ▣ Les ingénieurs doivent respecter la confidentialité de leurs employeurs ou de leurs clients, même si des clauses de confidentialité n'ont pas été signées

## □ Compétence

- ▣ Un ingénieur ne doit pas se faire plus beau qu'il n'est. Il ne doit pas accepter une tâche qu'il sait ne pas être capable de réaliser correctement

# Responsabilité professionnelle

38/52

- Respect des droits de propriété intellectuelle
  - ▣ Les ingénieurs devraient être sensibilisés aux lois locales concernant les droits d'auteurs, les brevets, etc. Ils devraient être attentifs à faire en sorte que cela soit respecté pour les clients et l'employeur
  
- Ne pas faire mauvais usage de l'informatique
  - ▣ Les ingénieurs ne devraient pas faire usage de leur compétences techniques pour faire un mauvais usage des ordinateurs des autres gens (cela va de jouer sur l'ordi de la société à disséminer des virus...)

# Code de pratiques éthiques ACM/IEEE

39/52

- Les associations d'informaticiens (ACM et IEEE) ont coopéré pour mettre au point un code de pratiques éthiques
- Les membres de ces associations signent ce code de pratiques lorsqu'ils adhèrent
- Contient 8 principes relatifs aux comportements des informaticiens

# Préambule aux principes éthiques

40/52

- *Computers have a central and growing role in commerce, industry, government, medicine, education, entertainment and society at large. Software engineers are those who contribute by direct participation or by teaching, to the analysis, specification, design, development, certification, maintenance and testing of software systems.*
- *Because of their roles in developing software systems, **software engineers have significant opportunities to do good or cause harm, to enable others to do good or cause harm, or to influence others to do good or cause harm.** To ensure, as much as possible, that their efforts will be used for good, software engineers must commit themselves to making software engineering a beneficial and respected profession.*



# Les principes éthiques (ACM/IEEE)

41/52

## Software Engineering Code of Ethics and Professional Practice

ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices

### PREAMBLE

The short version of the code summarizes aspirations at a high level of the abstraction; the clauses that are included in the full version give examples and details of how these aspirations change the way we act as software engineering professionals. Without the aspirations, the details can become legalistic and tedious; without the details, the aspirations can become high sounding but empty; together, the aspirations and the details form a cohesive code.

Software engineers shall commit themselves to making the analysis, specification, design, development, testing and maintenance of software a beneficial and respected profession. In accordance with their commitment to the health, safety and welfare of the public, software engineers shall adhere to the following **Eight Principles**:

# Les principes éthiques (ACM/IEEE)

42/52

1. PUBLIC - Software engineers shall act consistently with the public interest.
2. CLIENT AND EMPLOYER - Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.
3. PRODUCT - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.
4. JUDGMENT - Software engineers shall maintain integrity and independence in their professional judgment.
5. MANAGEMENT - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.
6. PROFESSION - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
7. COLLEAGUES - Software engineers shall be fair to and supportive of their colleagues.
8. SELF - Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

# Parfois on se trouve face à un dilemme

43/52

- Désaccord avec la politique managériale de l'entreprise
- L'employeur agit de façon non éthique et produit un logiciel critique sans réaliser un test complet du système
- L'entreprise se lance dans le développement de systèmes d'armement
- ...

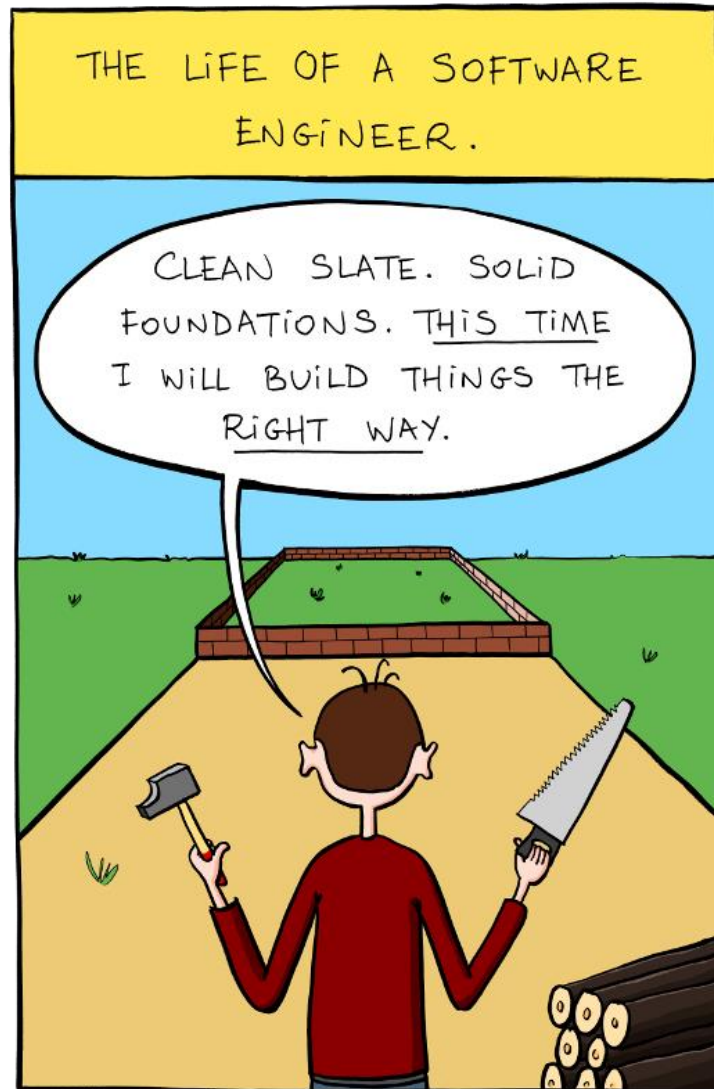
# Ce que l'on va voir par la suite

44/52

- Processus de développement du logiciel
- Ingénierie des besoins
- UML
- Estimation du coût d'un logiciel
- Test du logiciel
- Gestion de projets
  
- Un projet !!!

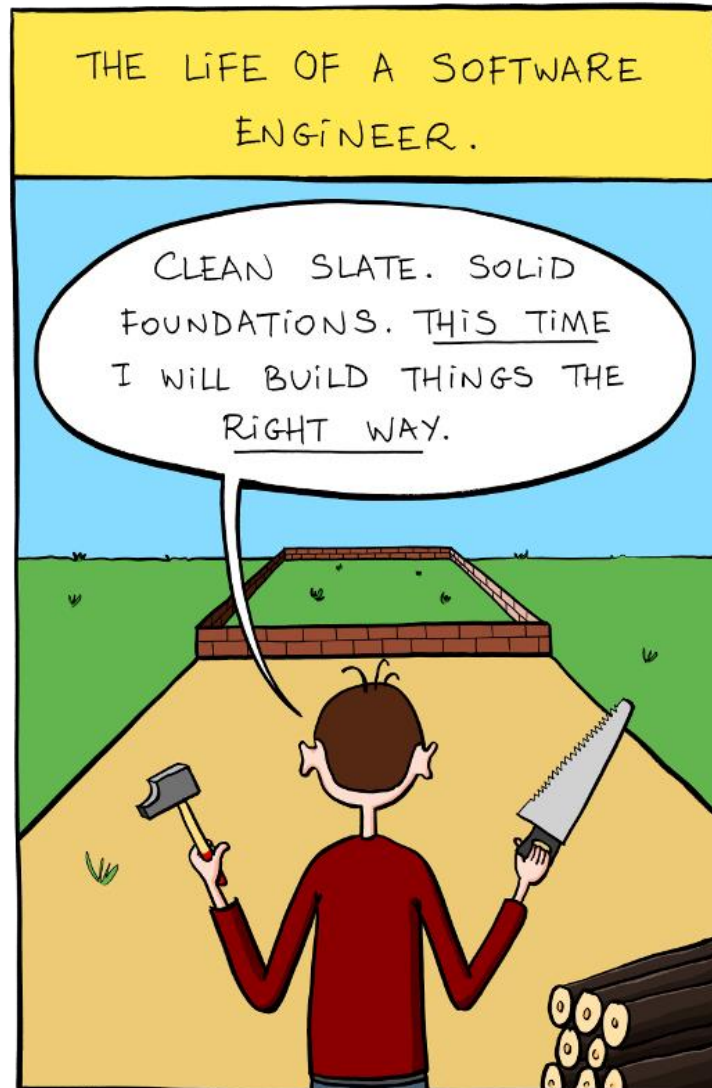
# Ce que l'on veut éviter

45/52



# Ce que l'on veut éviter

46/52



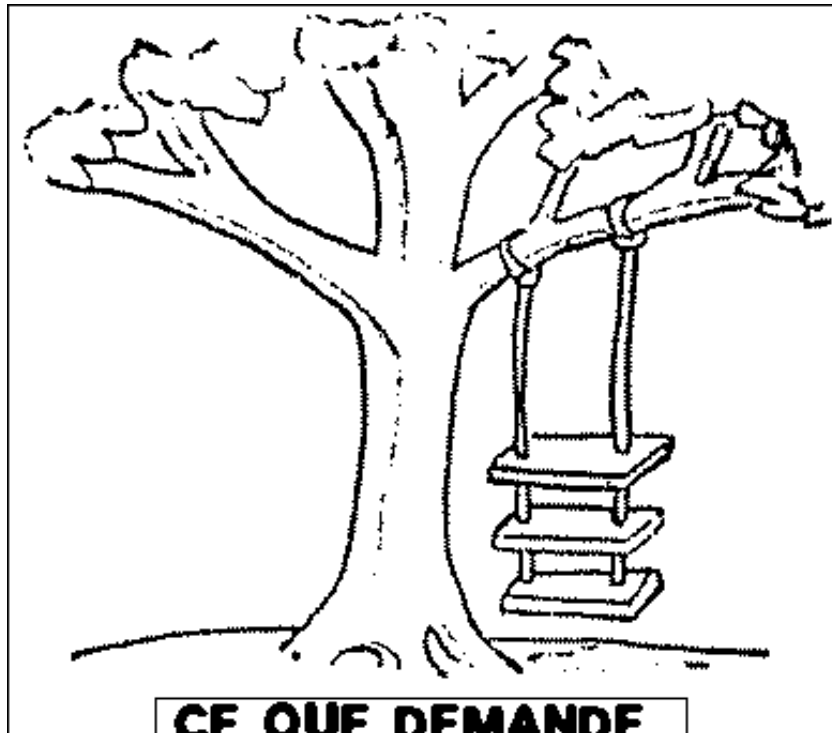
# En prévision du 1<sup>er</sup> TD

47/52

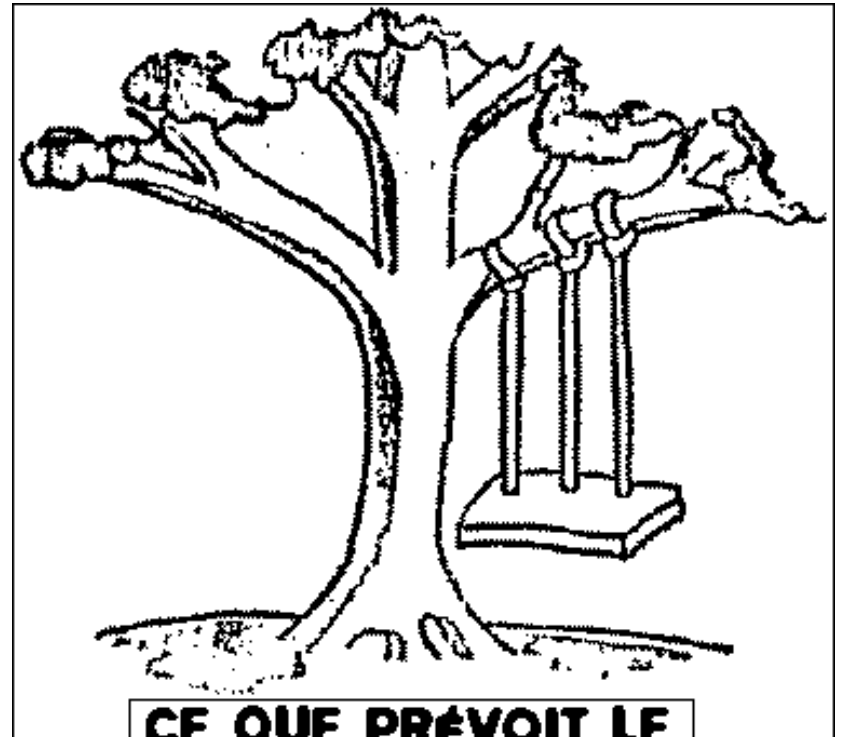


# En prévision du 1<sup>er</sup> TD

48/52



**CE QUE DEMANDE  
LE CLIENT**

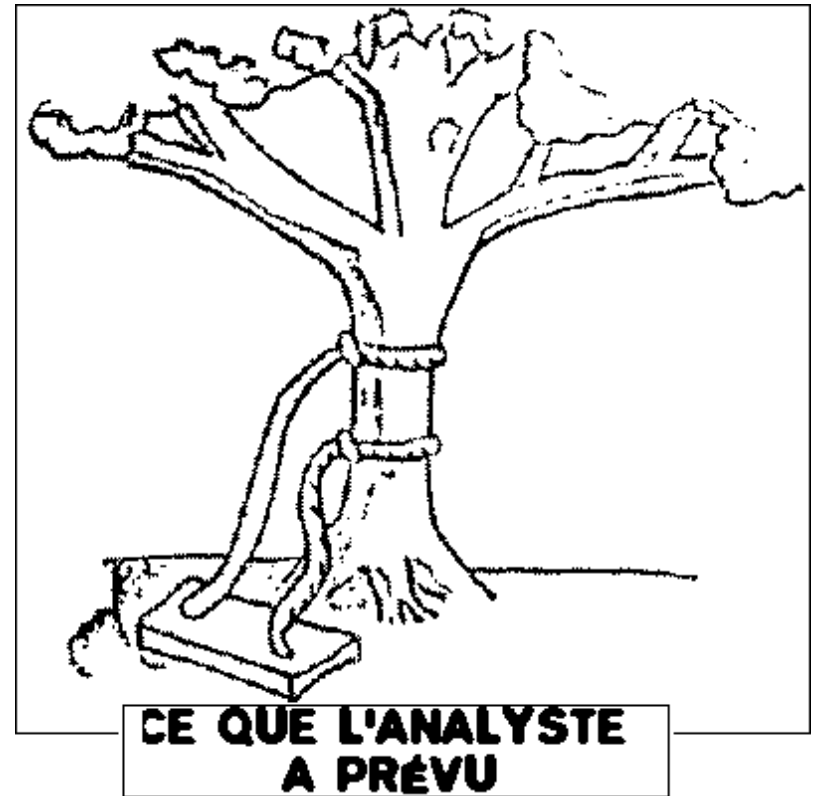


**CE QUE PRÉVOIT LE  
CONTRAT**



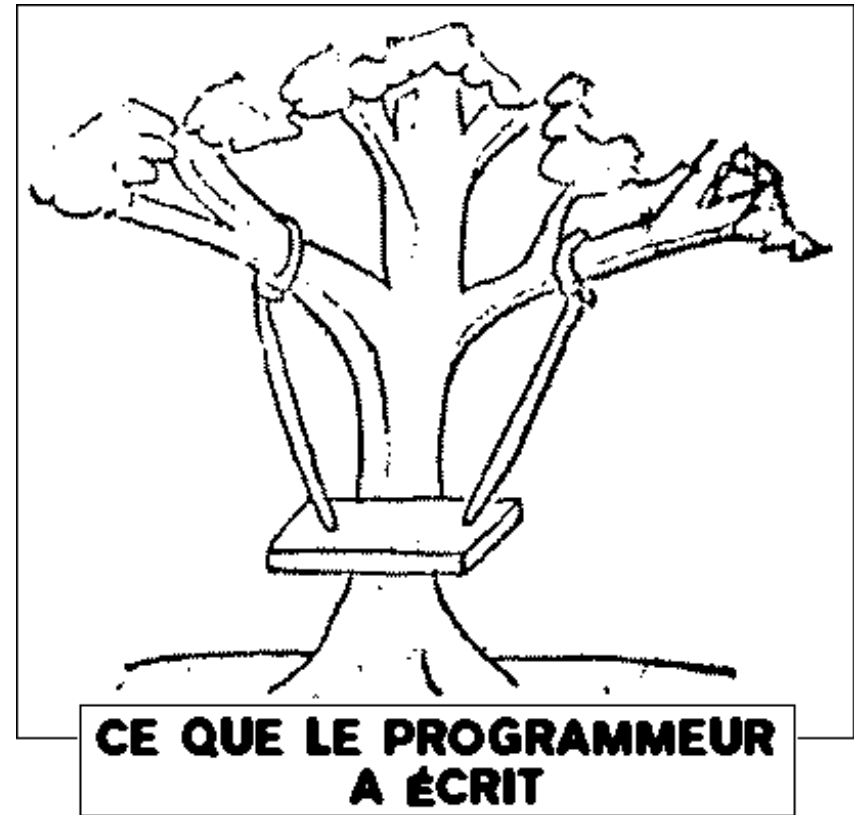
# En prévision du 1<sup>er</sup> TD

49/52



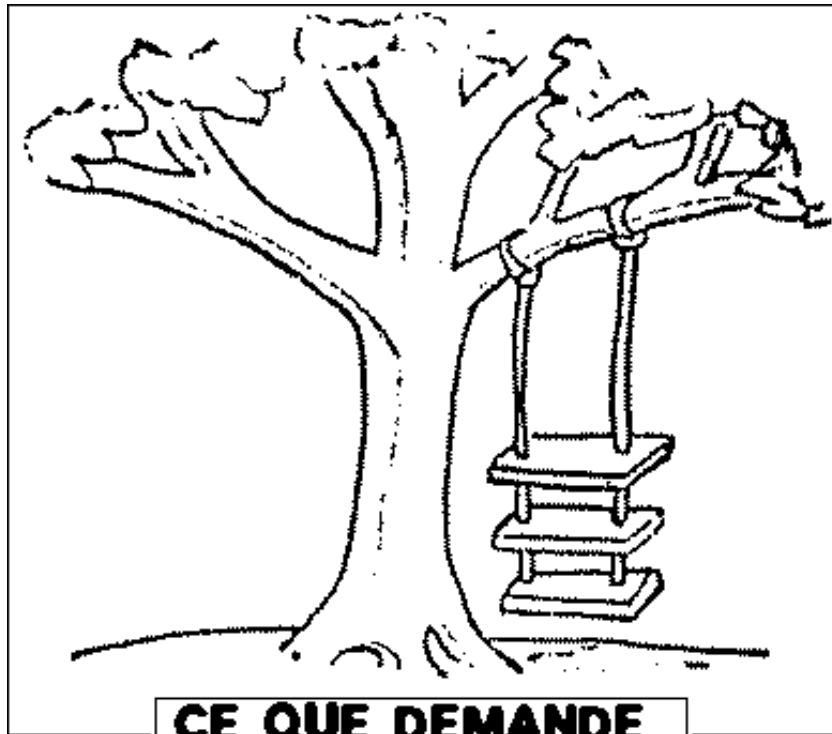
# En prévision du 1<sup>er</sup> TD

50/52

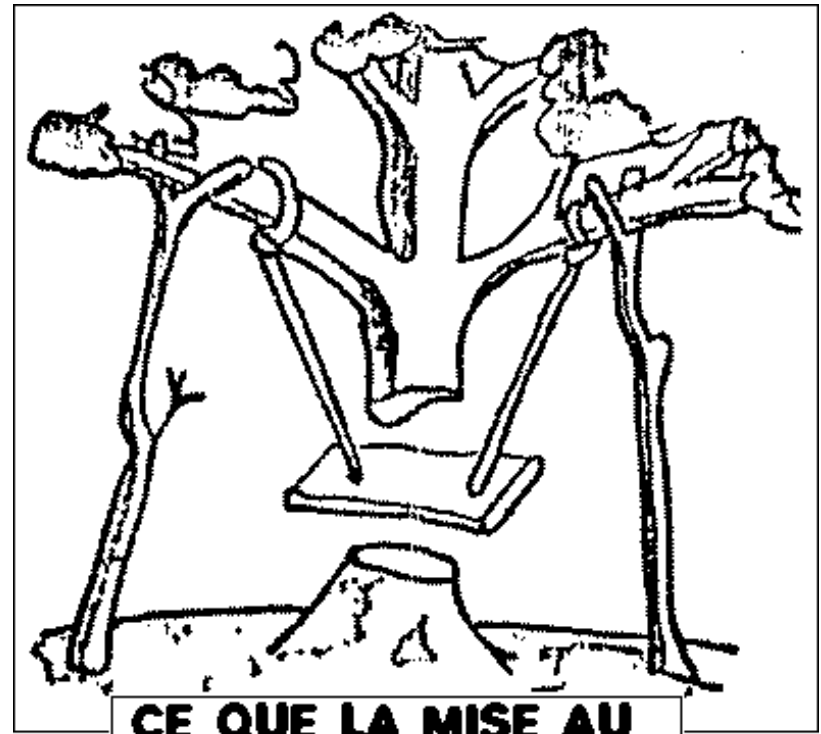


# En prévision du 1<sup>er</sup> TD

51/52



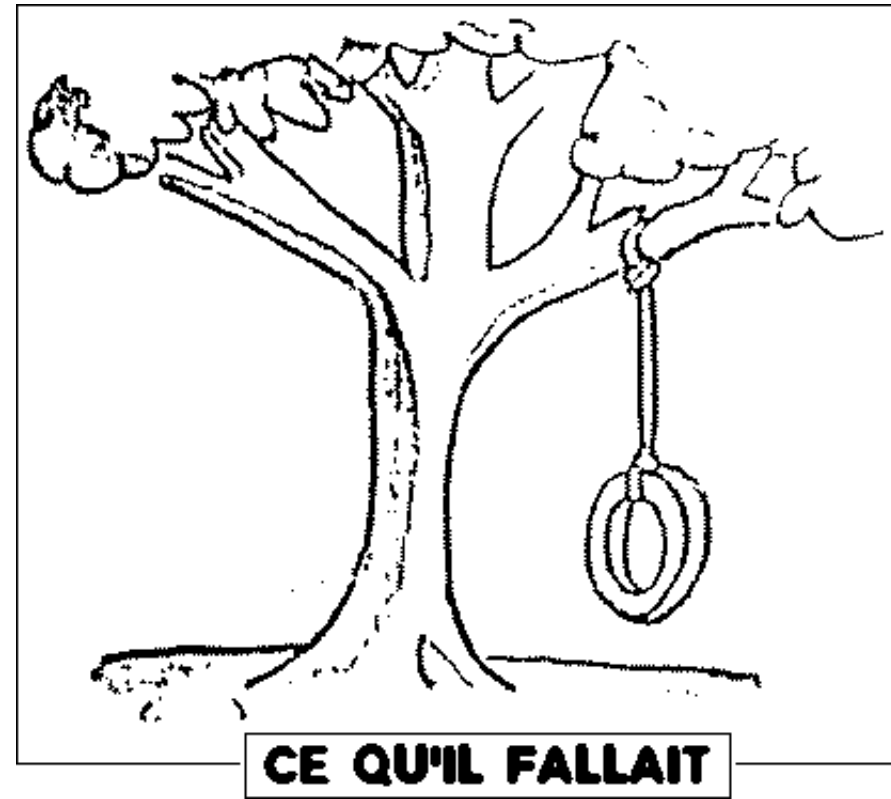
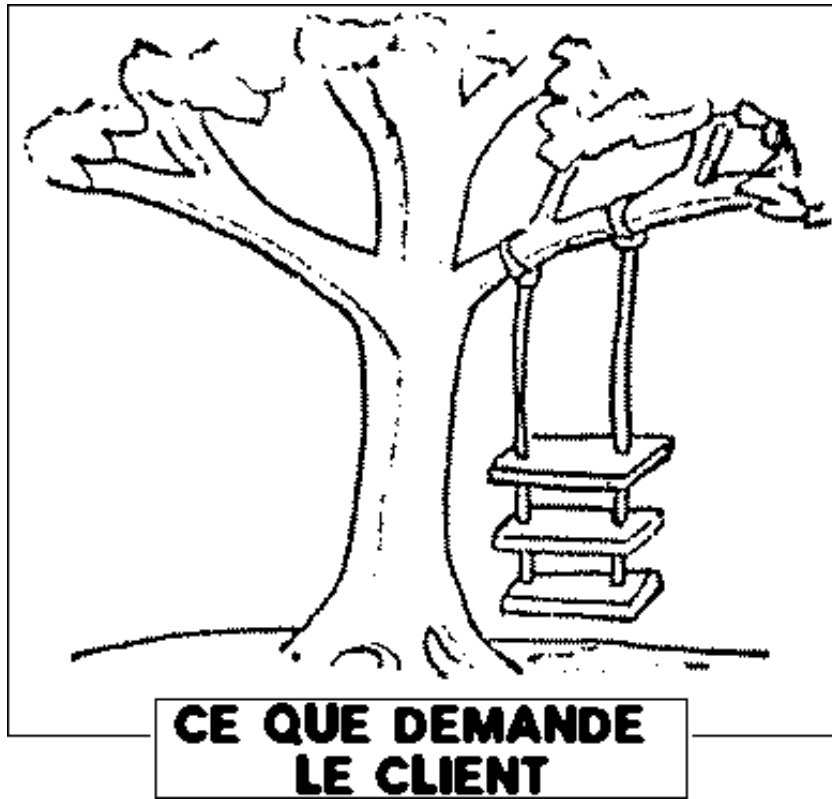
**CE QUE DEMANDE  
LE CLIENT**



**CE QUE LA MISE AU  
POINT A FAIT**

# En prévision du 1<sup>er</sup> TD

52/52



# En prévision du 1<sup>er</sup> TD

53/52

- Travail en groupes de 3 ou 4
- Client
  - Directeur
  - Directeur financier
- SSII
  - Directeur informatique
  - Chef de projets
- Client commande un projet
- SSII essaie de comprendre
- Rédaction de documents

FIN DE LA 1<sup>ère</sup> PARTIE