

# Génie Logiciel

Nistor GROZAVU

[nistor@lipn.univ-paris13.fr](mailto:nistor@lipn.univ-paris13.fr)

# Introduction

mai 2013

# GL : principes et problématique

**La production d'un logiciel tout comme la production de n'importe quel autre produit/bien nécessite la mise en œuvre de méthodes et outils dépassant largement le cadre de la seule programmation, regroupés dans "Génie Logiciel".**

- conférence de l'OTAN à Garmisch en 1969 : première utilisation du terme
- 1973 : première conférence sur la GL
- 1975 : première Revue IEEE sur la GL (IEEE Transactions on Software Engineering)

# GL : principes et problématique

**La programmation est une discipline récente purement intellectuelle, certains la qualifiaient même d'art [Knuth, The art of software programming] ; pour ces raisons, on y trouve des attitudes et des habitudes qui paraîtraient étranges dans d'autres disciplines plus mures.**

Not invented here (NIH) is the philosophy of social, corporate, or institutional cultures that avoid using or buying already existing products, research, standards, or knowledge because of their external origins and costs.

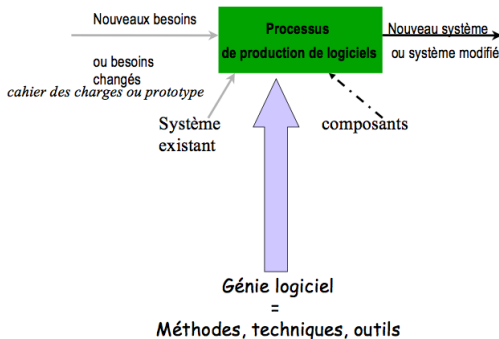
# GL : principes et problématique

## **Génie logiciel ; Génie chimique ;**

D'un point de vue économique, tout comme il convient de distinguer Chimie et Génie Chimique, il faut différencier programmation et génie logiciel. En chimie si deux réactions différentes conduisent au même produit on n'a aucune raison de préférer l'une à l'autre. Alors que le génie chimique différenciera parmi ces deux réactions celle qui met en jeu les produits de base les moins chers par exemple.

# GL : principes et problématique

Anne-Marie Hugues © 19/12/02



# GL : principes et problématique

## Les bugs...

- vers Vénus( années 60)
- vers Mars (99)
- Mariner 1, 1962
- La panne A TT (1990)
- L'échec d'Ariane 5 (1996)
- 28.09.1998, Dublin - Déconnection de 140 feux
- logiciels de surveillance médicale défaillants
- année 2000

## Bug informatique le plus grave

Entre 1985 et 1987, plusieurs patients sont décédés ou ont été gravement atteints dans leur santé suite à un bug de dosage d'un appareil médical de radiothérapie. L'appareil, le Therac-25, permet d'envoyer des faisceaux d'électrons, de rayons X ou de lumière (pour éclairer le patient). La puissance d'émission du faisceau d'électrons est 100 fois plus élevée que celle des rayons X. Un logiciel ajuste la puissance de l'émetteur, et un dispositif mécanique / optique permet de produire soit des rayons X, soit des électrons, soit de la lumière. Suite à un crash informatique durant le traitement, l'appareil envoyait au patient une dose de rayons X 100 fois supérieure à celle prévue. L'incident se reproduisit plusieurs fois dans divers hôpitaux des États-Unis et passa inaperçu jusqu'au jour où des patients, gravement atteints dans leur santé déposent plainte contre leur hôpital.

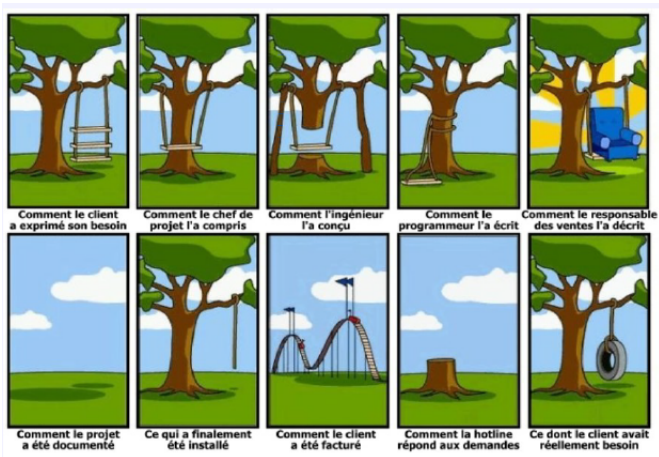


# Le bug de l'an 2000

Le bug de l'an 2000, aussi appelé bug du millénaire : un ou plusieurs bugs dans un logiciel qui manipule des dates provoquent des dysfonctionnements lorsque les dates sont postérieures au 31 décembre 1999. Une des causes est que les calculs sur les dates se font uniquement sur les deux derniers chiffres de l'année. Les problèmes potentiels posés par la date du 31 décembre 1999 ont été anticipés la première fois par Bob Berner en 1971<sup>18</sup>. Ils ont provoqué une importante mobilisation des entreprises de génie logiciel quelques années avant la date butoir et le coût total des travaux de contrôle et de maintenance préventive sont estimés à plus de 600 millions de dollars

# Pourquoi GL

## c Mireille Blay-Fornarino



# Organisation du processus

- Cycle de vie
- Responsabilités
- Rôles

# Cycle de vie d'un logiciel

**Le cycle de vie d'un logiciel (en anglais software lifecycle), désigne toutes les étapes du développement d'un logiciel, de sa conception à sa disparition.**

**C'est la description d'un processus couvrant les phases de :**

- Création d'un produit
- Distribution sur un marché
- Disparition

**Le but de ce découpage est de :**

- Maîtriser les risques
- Maîtriser au mieux les délais et les coûts
- Obtenir une qualité conforme aux exigences

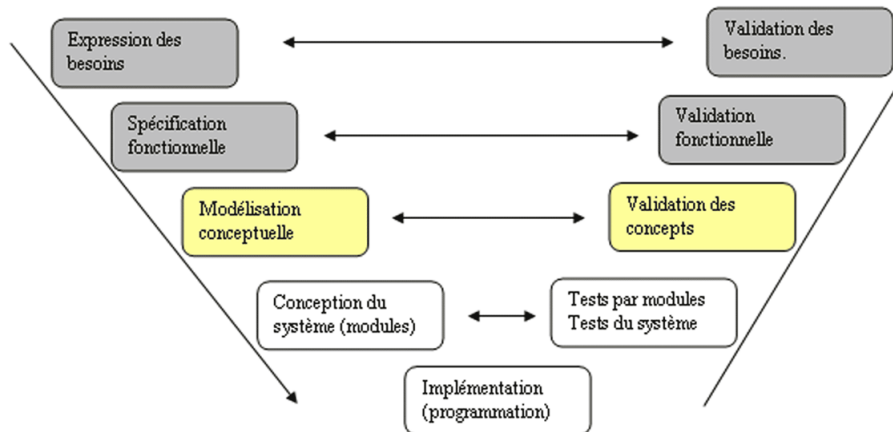
# Cycle de vie d'un logiciel

## les activités

- 1 Définition des objectifs
- 2 Analyse des besoins et faisabilité
- 3 Conception générale
- 4 Conception détaillée
- 5 Codage
- 6 Tests unitaires
- 7 Intégration
- 8 Qualification
- 9 Documentation
- 10 Mise en production
- 11 Maintenance

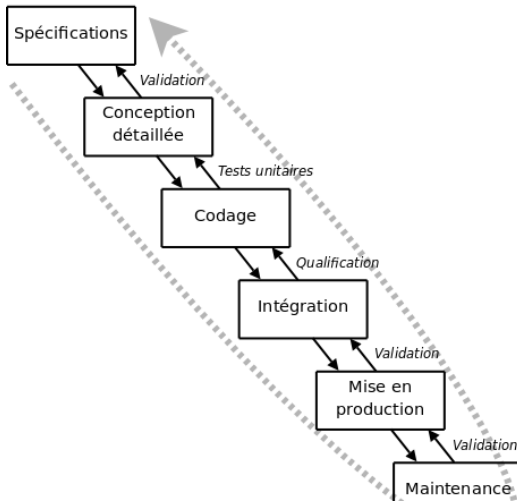
# Cycle de vie d'un logiciel

## Cycle de vie en V



# Cycle de vie d'un logiciel

## Cycle de vie en cascade



# Cycle de vie d'un logiciel

## L'approche Orienté Objet UML



# Références

- Anne-Marie Hugues © 19/12/02
- Laurent Audibert, Université Paris 13
- ...
- ...
- ...