heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Ecole d'Ingénieurs de l'état de Vaud
Eric Lefrançois
Février 2018

Génie Logiciel- La problématique

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Sommaire

Génie – Logiciel - La problématique

UNE PREMIERE SERIE DE DEVINETTES..

- Y a-t-il une différence entre :
  - Construire une cabane au fond du jardin;
  - Construire le bâtiment de l'HEIG-VD?
- Quel projet coûte le plus cher ?
- Quel projet est le plus long à réaliser ?
- Quel projet nécessite le plus de personnes pour sa réalisation ?
- Dans quel projet la phase de construction est-elle proportionnellement la plus longue (par rapport à la durée complète du projet)?

UNE DEUXIÈME SÉRIE...

Le bâtiment de l'HEIG-VD a été construit aux environs de 1975

À cette époque, il correspondait aux besoins de l'école

Les besoins d'aujourd'hui ont-ils été respectés ?

- Le bâtiment n'est pas modulaire
- Les normes de l'époque ne conviennent plus

⇨ La taille des bâtiments a une grande importance sur leur réalisation

Est-ce la même chose en informatique ?

LES PETITS LOGICIELS

- Le cahier des charges est…
  - Petit
  - Facile à comprendre
  - Parfois transmis oralement

- Le coût du logiciel est faible

- En général, le développeur connait bien le métier où s'applique le logiciel

- La conception du programme se fait souvent mentalement

Sans documentation !

LES PETITS LOGICIELS .. SUITE

- Choix des technologies (OS, langage, etc.) opéré par le développeur
- Développement réalisé par une personne
  et dure peu de temps : quelques semaines ou quelques mois

⇨ *Pas de planification à générer !*

- Le logiciel sera déployé une ou deux fois, par le développeur lui-même
- La maintenance du logiciel cesse de facto lorsque le développeur cesse de le supporter (départ de la société, etc.)

⇨ Cela entraîne souvent la mort du logiciel

LES PETITS LOGICIELS…

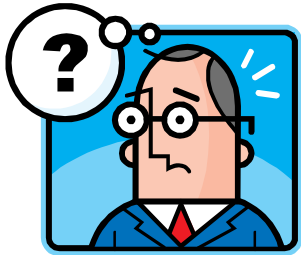
☺ Toutes les conditions pour le succès sont donc réunies !

ET LES GROS LOGICIELS ??

LES GROS LOGICIELS

- Cahier des charges est volumineux...
  - Difficile à comprendre
  - Mal écrit, le client n'a pas une idée très claire du résultat final
  - Incomplet
  - Écrit par des personnes non informaticiennes

- En général les développeurs ne connaissent pas le métier du client



- La conception du programme est un vrai casse-tête :
  - Incertitudes sur le cahier des charges
  - Taille du programme à réaliser

LES GROS LOGICIELS – SUITE..

- Le coût du logiciel est énorme ⇨ Gros enjeux 
- Le développement se fait par équipe, parfois assez grande
- Le développement peut durer des années !
- Une estimation préalable du temps de développement obligatoire
- Obligation de respecter les délais ⇨ Défi ! 
- Le logiciel doit être parfaitement documenté 
- Le logiciel n'est pas déployé par l'équipe de développement
  - ⇨ *Personnel spécialement formé chargé du déploiement*

LES GROS LOGICIELS – SUITE…

☹ *Toutes les conditions pour l'échec du développement sont réunies*

Posez-vous seulement la question

Dans quelle catégorie vous situez-vous lorsque vous développez un projet informatique :

- Dans le cadre scolaire ?
- Dans le cadre professionnel ?

Un petit exercice...

Combien coûte une ligne de code, sachant que :

- Un bon développeur coûte Frs 200'000.- par année à son entreprise tous frais compris
- Il code en moyenne 20 lignes par jour
- Il est absent en moyenne 1/5 du temps (vacances, armée, maladie, etc.)
- il y a environ 250 jours ouvrables par année

QUELQUES DÉFINITIONS CLÉS

- **Spécification (ou *Analyse*)**

  Décrire formellement ou semi-formellement les fonctions du système

  ⇨ *Q*ue doit faire le système ? *P*ourquoi ?

- **Conception**

  Imaginer, modéliser et décider comment implémenter le système

  ⇨ *C*omment réaliser le système ?

- **Implémentation** ⇨ *C*odage

- **Test** ⇨ *V*érifier que le système fonctionne correctement

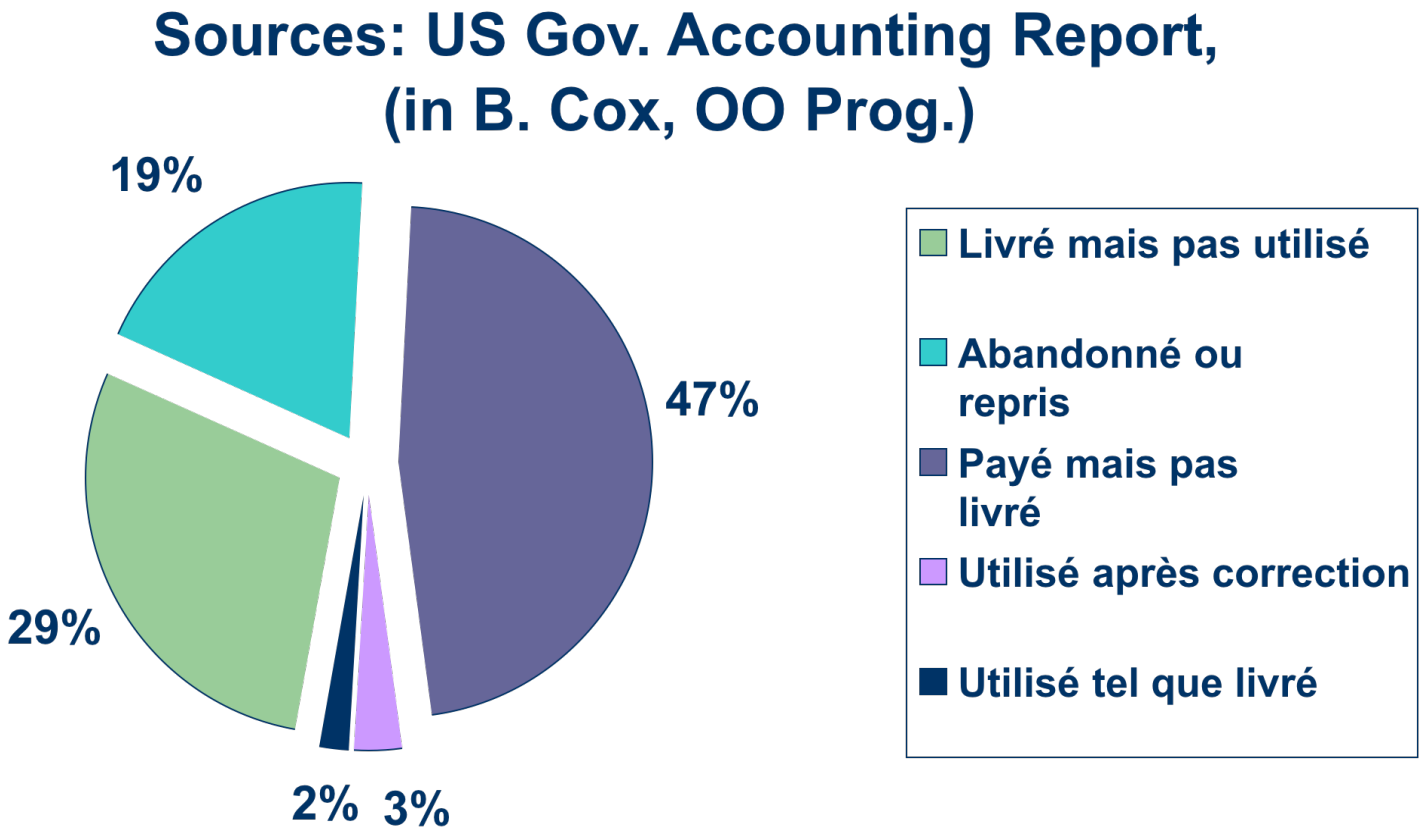
- **Maintenance** ⇨ *L*ogiciel est en service chez le client
  - Ajout de nouvelles fonctionnalités
  - Adaptation, modification de fonctionnalités existantes
  - Corrections des erreurs détectées par le client

LA CRISE DU LOGICIEL DES ANNEES 70

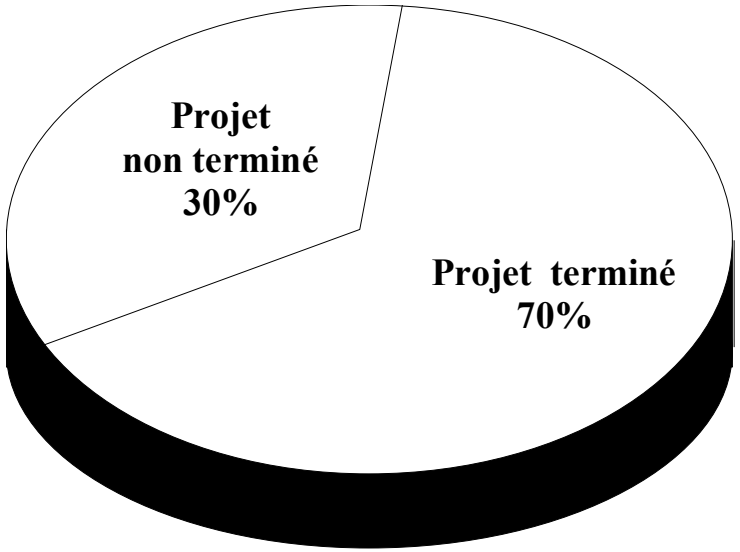
Avec l'accroissement de la complexité des logiciels (notamment de leur taille), le développement ne suit pas...

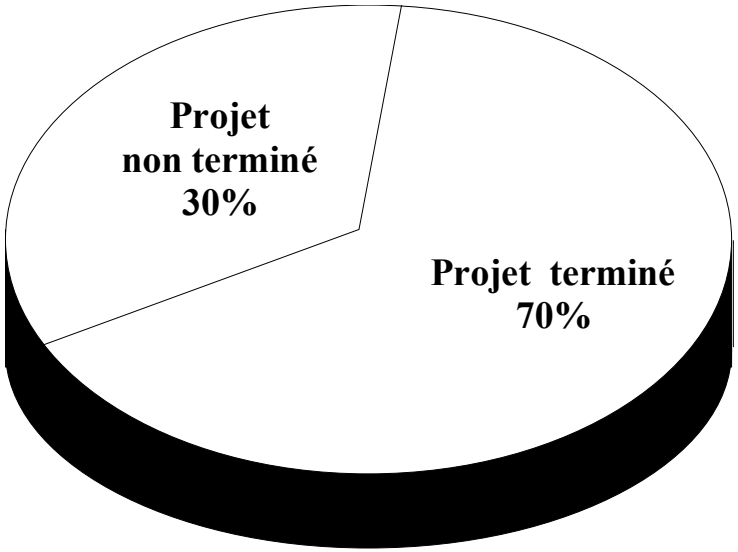
Il faudra 10 ans pour développer l'OS 360 des IBM 360 !!

Rapport du Congrès Américain en 1979 sur 487 projets



Un rapport américain de 1994 - Standish Group International





- Mais 53% des projets dépassent le budget prévu initialement de 200% !

- Environ $81 milliards dépensés pour des projets abandonnés aux USA en 1999

✍ Les projets analysés ont tous utilisé le **modèle en cascade** (***waterfall***)

PANIQUE À BORD DU BATEAU LOGICIEL

Problèmes rencontrés à tous les points de vue...

- Du point de vue utilisateur
- Du point de vue de l'équipe de développement
- Du point de vue gestion et planification

PANIQUE À BORD DU BATEAU LOGICIEL (2)..

- **Du point de vue utilisateur**
  - Les logiciels réalisés ne correspondent pas aux besoins des utilisateurs ;
  - Les changements de besoin du client sont difficiles à intégrer dans le développement ;

PANIQUE À BORD DU BATEAU LOGICIEL (3)…

- Du point de vue utilisateur

- **Du point de vue de l'équipe de développement**
  - La maintenance des logiciels est complexe et coûteuse
  - Trop d'erreurs
  - Performance du système souvent inacceptable
  - Logiciels rarement portables

PANIQUE À BORD DU BATEAU LOGICIEL (4)…

- Du point de vue utilisateur
- Du point de vue de l'équipe de développement
- **Du point de vue gestion et planification**
  - Le système est difficilement réutilisable pour de futurs développements
  - Coûts imprévisibles et généralement prohibitifs
  - Délais généralement dépassés

PANIQUE À BORD DU BATEAU LOGICIEL (5)…

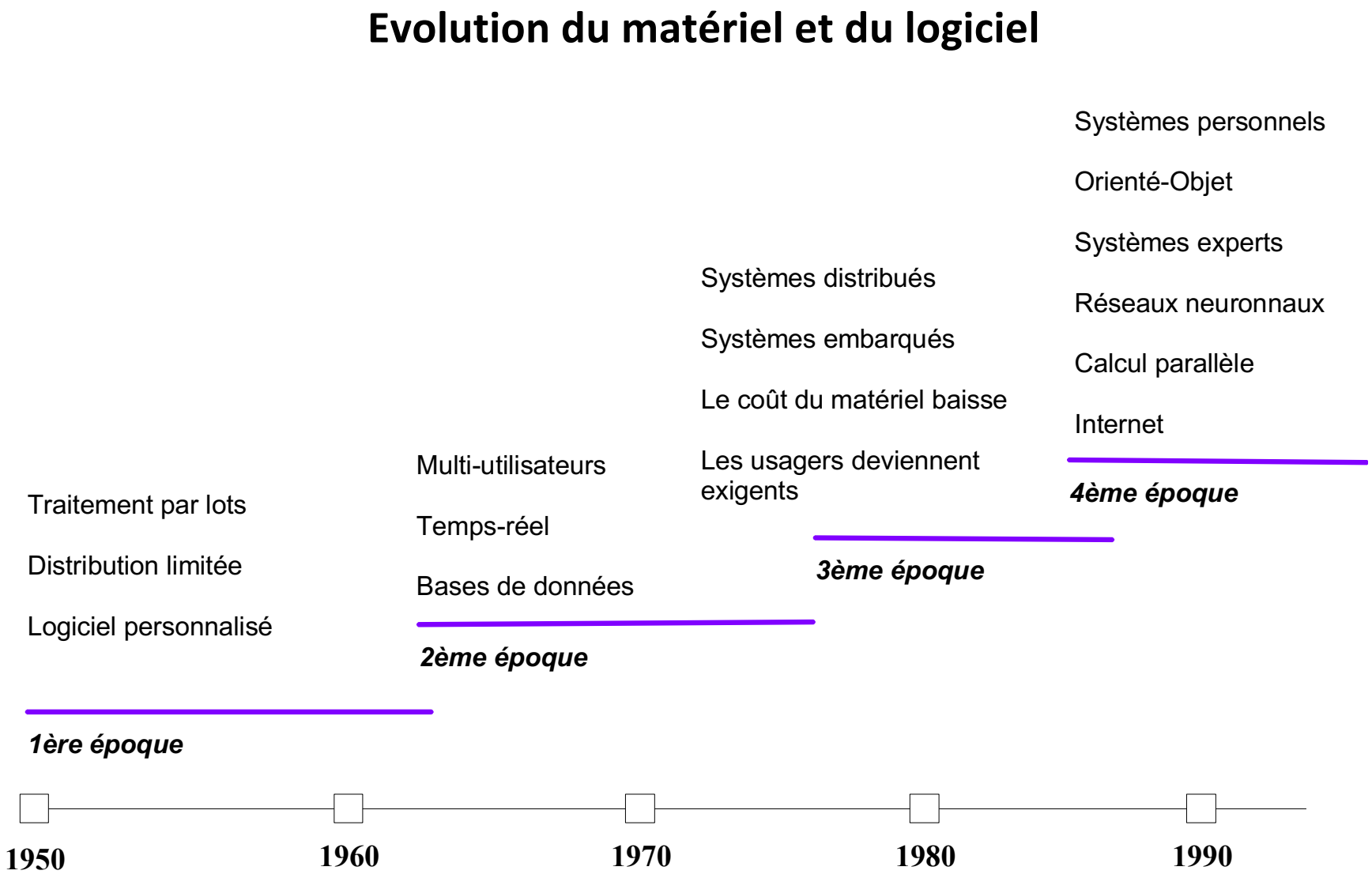
Et pourquoi donc ?

CAR LE LOGICIEL EST UNE TÂCHE COMPLEXE...

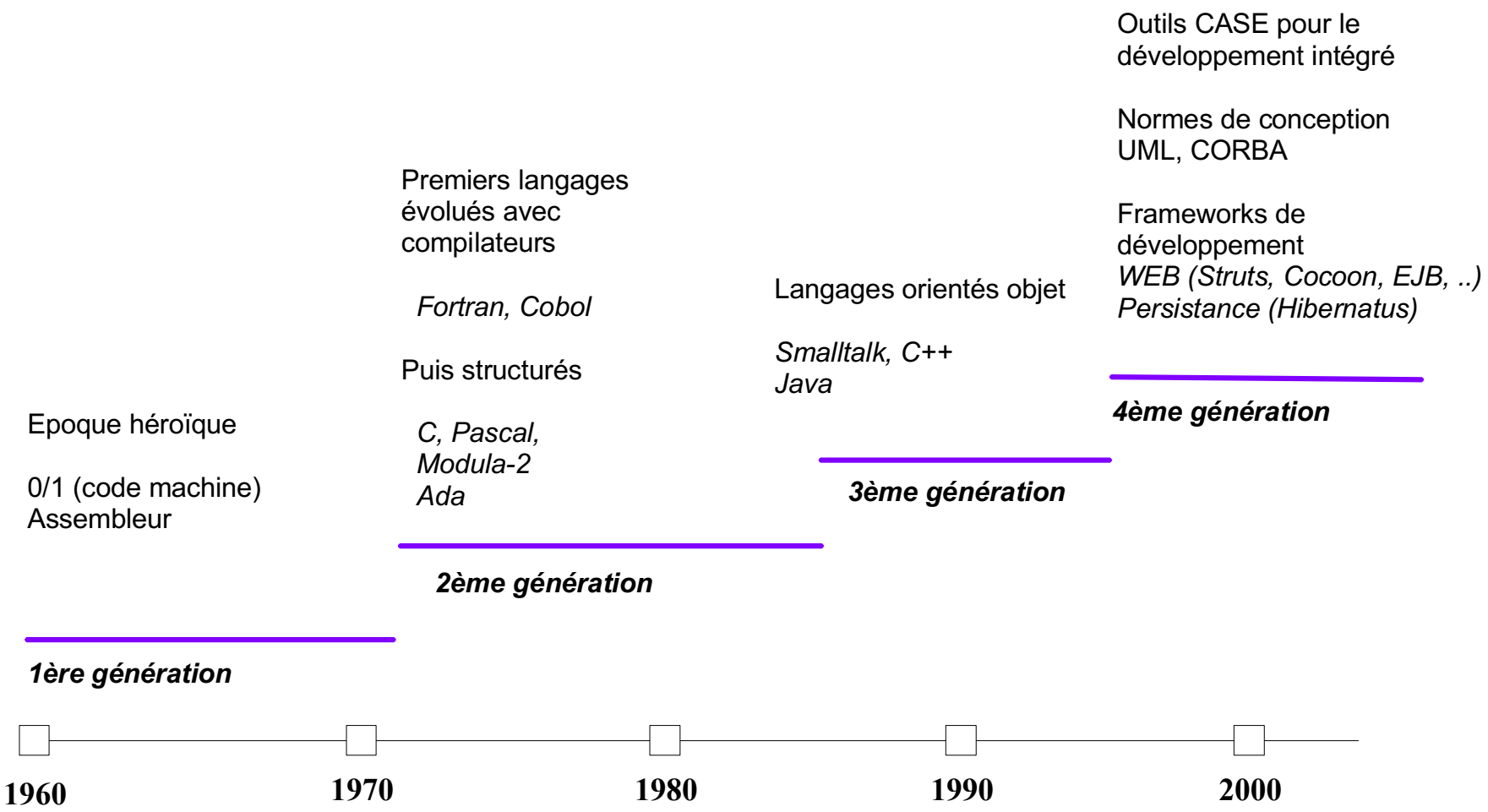
Reconnaissons tout d'abord que ça n'est pas une tâche facile...

- De par sa **taille**
- Complexité du système à informatiser
  ⇨ **Modélisation du « monde réel »**
- Complexité de **communication** entre les futurs utilisateurs
- Complexité de **communication** entre utilisateurs et les développeurs
  ⇨ vocabulaire technique et non-technique

CAR L'ÉVOLUTION EST TROP RAPIDE...

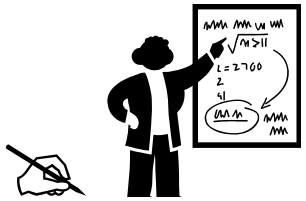


Evolution des langages et des outils



CAR AU DELÀ DES MYTHES, LA RÉALITÉ...

LES MYTHES DU COTE USAGER

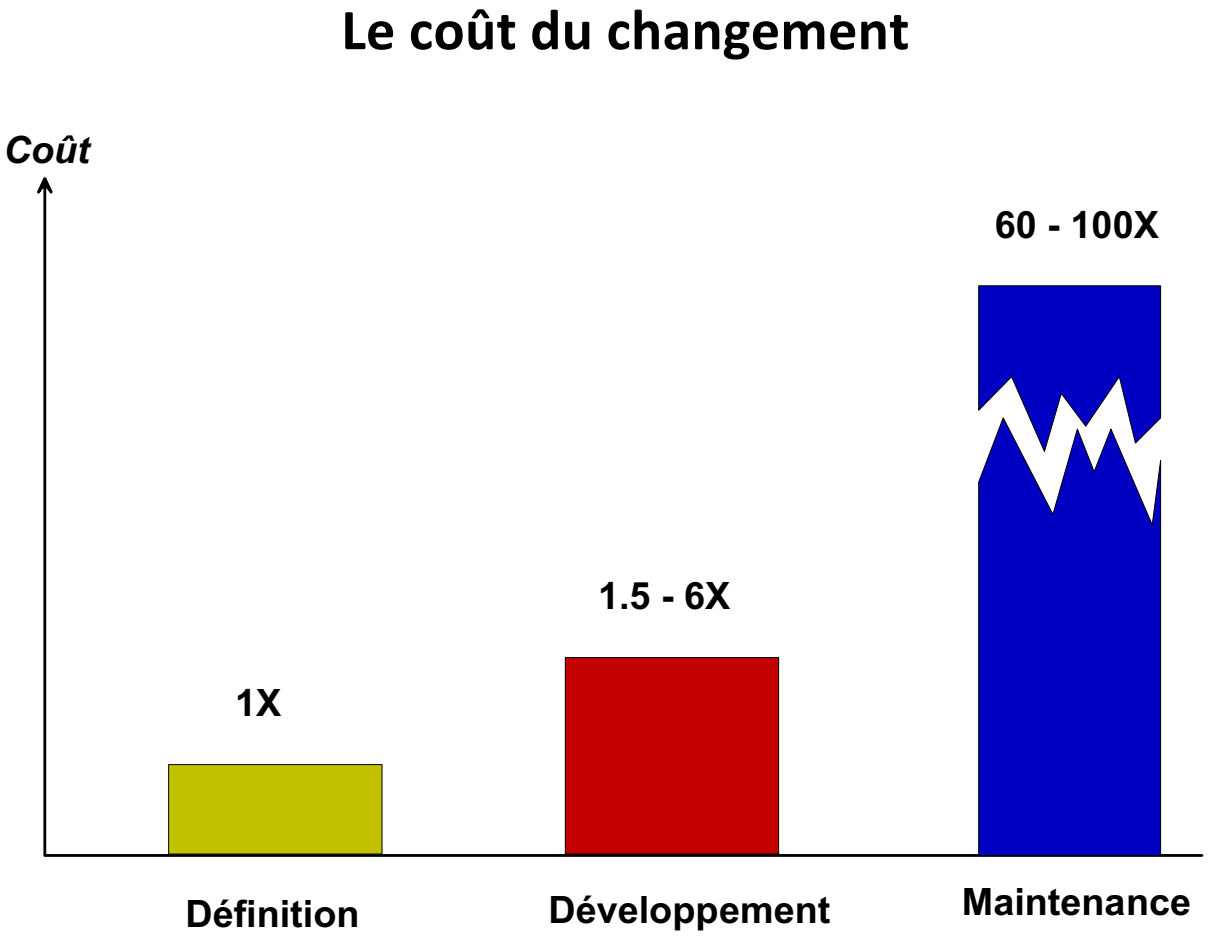


Les besoins du projet changent, mais on incorporera les modifications facilement parce que le logiciel est flexible !

☹***La réalité :*** Les coûts pour un changement de spécification augmentent de manière dramatique si on arrive dans les dernières phases du développement.

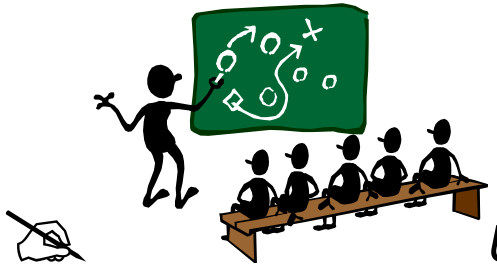
CAR AU DELÀ DES MYTHES, LA RÉALITÉ…

LES MYTHES DU COTE USAGER (2)



CAR AU-DELÀ DES MYTHES, LA RÉALITÉ…

LES MYTHES DU COTE DEVELOPPEUR

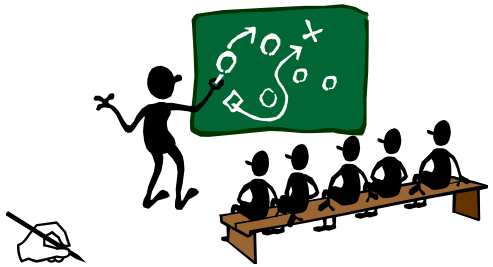


Une fois que le programme est écrit et qu'il fonctionne, le travail du développeur est terminé !

☹***La réalité :*** 50% à 70% de l'effort consacré à un programme se produit après la livraison à l'usager.

CAR AU DELÀ DES MYTHES, LA RÉALITÉ…

LES MYTHES DU CÔTÉ DÉVELOPPEUR - SUITE



Le succès d'un projet tient essentiellement de la livraison d'un programme fonctionnel !

☹***La réalité:*** Une configuration logicielle inclut toute la documentation, des données d'entrée pour les tests, etc..

CAR AU DELÀ DES MYTHES, LA RÉALITÉ…

LES MYTHES DU CÔTÉ GESTIONNAIRE

L'entreprise possède des normes, le logiciel développé devrait être satisfaisant !

☹ ***La réalité :*** Les standards sont-ils utilisés, appropriés ?

☹ ***La réalité :*** Plus que des outils … Il faut aussi une bonne pratique.

CAR AU-DELÀ DES MYTHES, LA RÉALITÉ…

LES MYTHES DU CÔTÉ GESTIONNAIRE - SUITE

Si le projet prend du retard, il suffira d'ajouter quelques programmeurs.

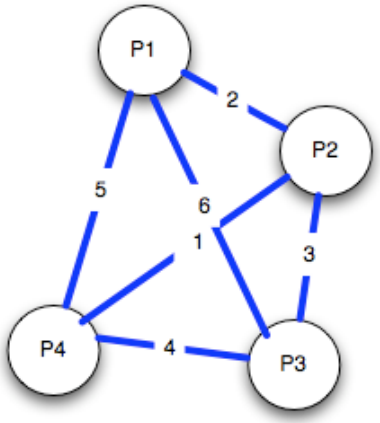
☹ ***La réalité :*** Le développement du logiciel n'est pas une activité mécanique. Ajouter des programmeurs peut empirer la situation

CAR AU DELÀ DES MYTHES, LA RÉALITÉ…

LES MYTHES DU CÔTÉ GESTIONNAIRE (4)

Loi de Brooks

Considérons un projet avec 4 programmeurs P1..P4

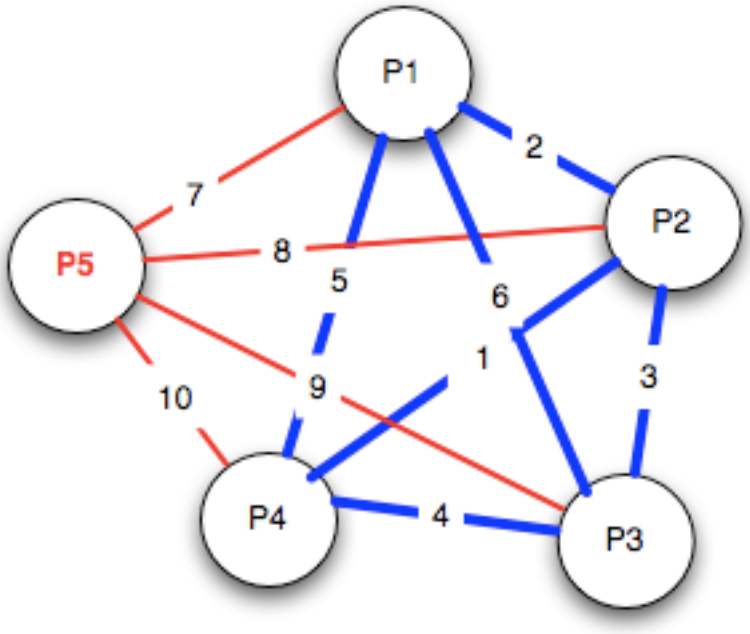


⇨ « If a project is late..”

CAR AU-DELÀ DES MYTHES, LA RÉALITÉ…

LES MYTHES DU CÔTÉ GESTIONNAIRE (5)

Loi de Brooks



N programmeurs

⇨ N(N-1)/2 liens de communications

"Adding more programmers to a late project makes it later"

POINTS MARQUANTS DE L'HISTOIRE DU GÉNIE LOGICIEL

ANNÉES 1960 ⇨ BALBUTIEMENTS

- Mise en évidence du problème
- Crise du logiciel
- Nato Meetings (Conférences Otan) :
  Rome ⇨ Guerre au « Goto »
  Garmish-Partenkirchen ⇨ « Software Engineering » (68)
- Réponse du Dod à la crise ⇨ langage Ada
  (Département de la Défense américaine)

POINTS MARQUANTS DE L'HISTOIRE DU GÉNIE LOGICIEL (2)

ANNÉES 1960 ⇨ BALBUTIEMENTS

ANNÉES 1970 ⇨ RECHERCHE D'UN MODÈLE (OUTILS, GESTION)

Début 70 : « Structured Programming » , inspiré par Pascal

Milieu 70 : Notion de cycle de vie ⇨ Cycle en cascade

Fin 70 : Méthodologies de spécification ⇨ « Structured Analysis »

Méthodologies de conception ⇨ « Structured Design »

POINTS MARQUANTS DE L'HISTOIRE DU GÉNIE LOGICIEL (3)

ANNÉES 1960 ⇨ BALBUTIEMENTS

ANNÉES 1970 ⇨ RECHERCHE D'UN MODÈLE (OUTILS, GESTION)

ANNÉES 1980 ⇨ RASSEMBLEMENT, COHÉSION

- Méthodes de développement globales : OMT, Booch, Fusion… puis UP
- Amélioration des outils : POO notamment
- On constate des gains de productivité

ANNEES 2000 ⇨ METHODES AGILES

DÉFINITION DU GÉNIE LOGICIEL

Définition 1 : Le génie logiciel

Sommerville Ian « Le génie logiciel », Addison-Wesley France, 1992

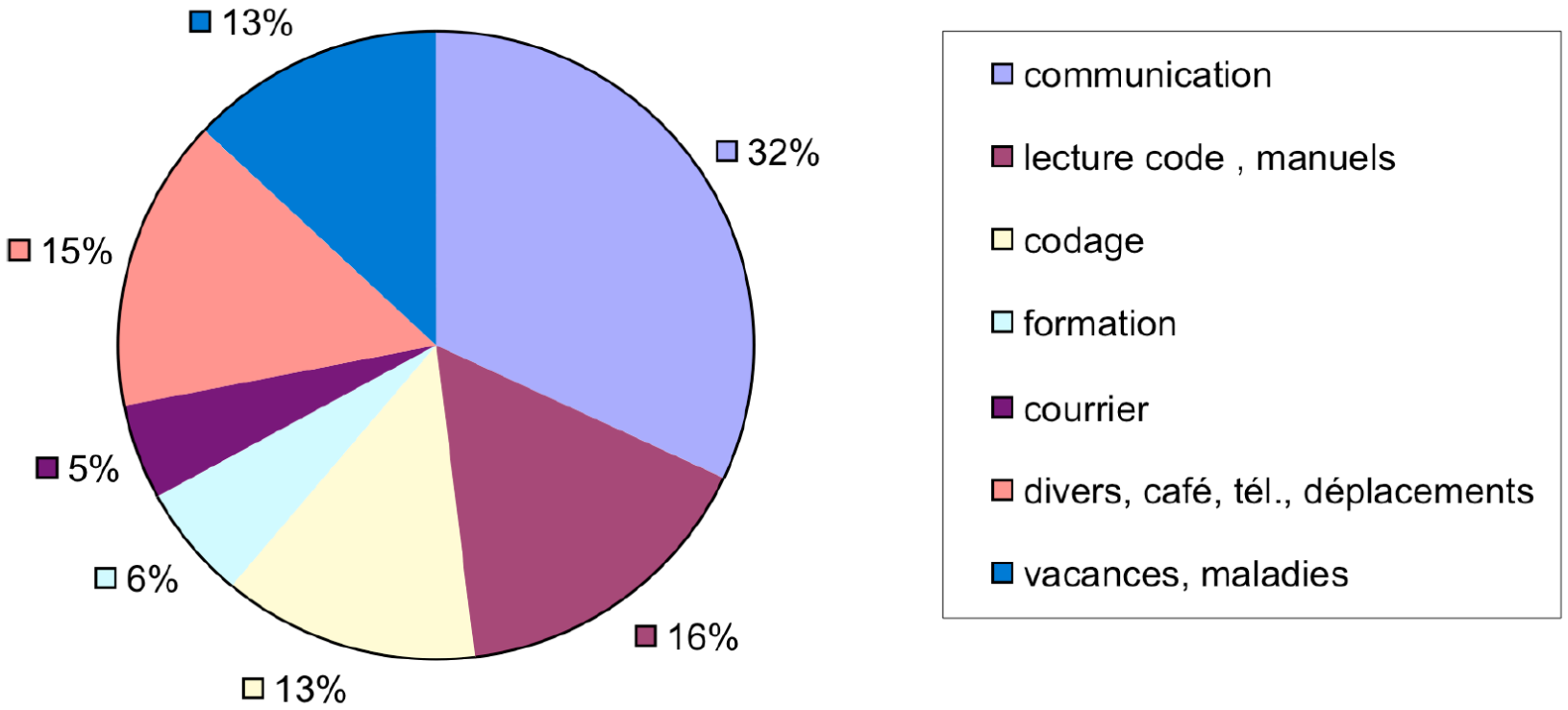
⇨ Discipline d'ingénierie concernée par le problème pratique du développement de **grands systèmes** logiciels

Définition 2 : Le génie logiciel

Discipline pour spécifier, construire, distribuer et maintenir des logiciels, en assurant :

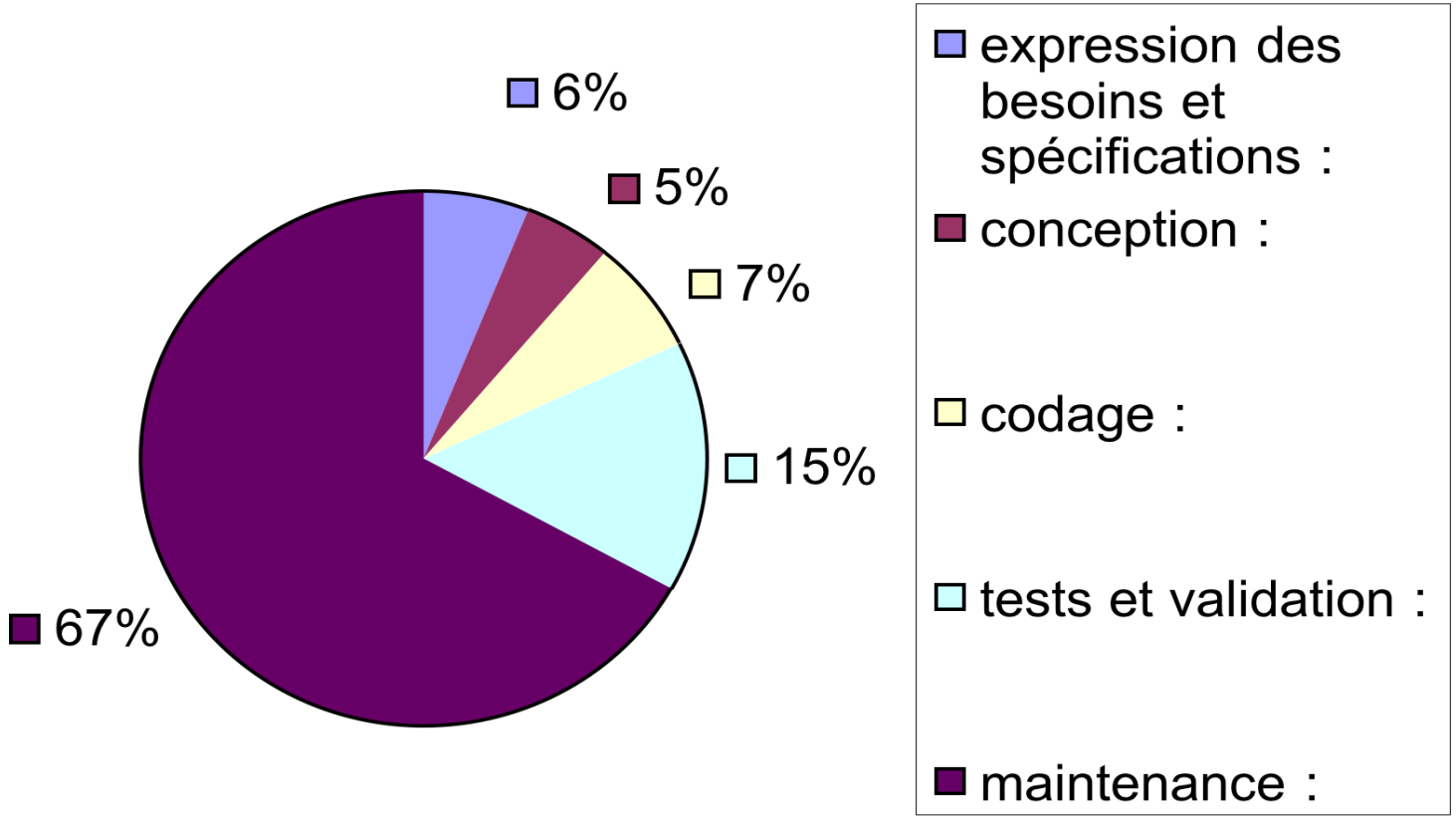
- La **qualité** (fiables, adaptés, conviviaux, évolutifs)
- Des **coûts contrôlés**
- Des **délais garantis**

PARLONS PRODUCTIVITÉ..



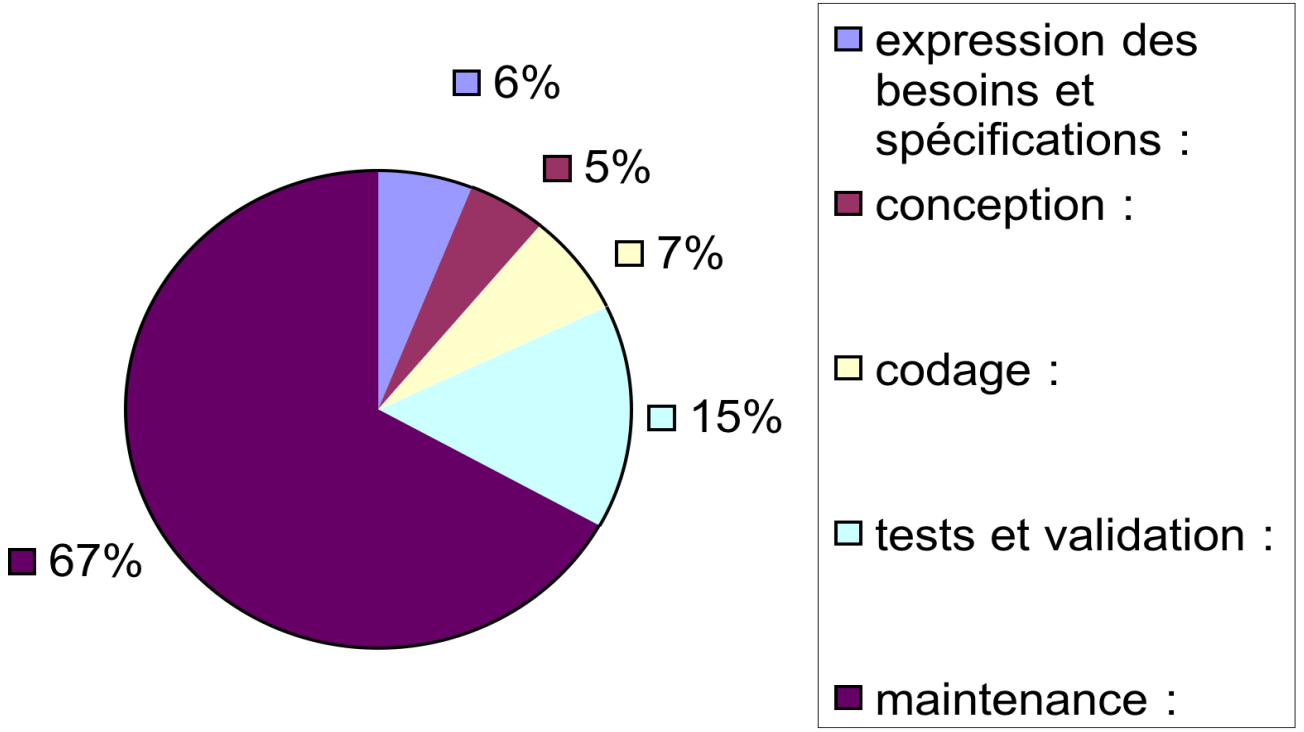
⇨ *Une instruction coûte en moyenne 40$ à 200$ (USA 1980)*

REPARTITION DES COUTS DE DEVELOPPEMENT



Pour des projets importants (> 5 personnes travaillant pendant > 12 mois)

REPARTITION DES COUTS DE DEVELOPPEMENT (2)



Un logiciel coûte plus cher à maintenir qu'à développer !!

LES QUALITÉS REQUISES D'UN LOGICIEL

Définition 3 : Le génie logiciel

Donner les moyens nécessaires à l'équipe de développement pour que le système réalisé :

- Respecte les coûts et les délais de développement ;
- Soit un **produit de qualité**.

Vous avez dit « Qualité » ?

Aborder le concept «**Qualité**» selon 3 points de vue..

- **Qualité Externe** (point de vue de l'utilisateur)
- **Qualité Interne** (point de vue du développement)
- **Qualité du Processus de Développement**

Qualité Externe (point de vue de l'utilisateur)

CORRECTION : Un logiciel « correct» satisfait à sa **spécification**

FIABILITE : Mesure le **degré de confiance** dans le fonctionnement...

Marche aujourd'hui, marchera-t'il demain ?

ROBUSTESSE : Aptitude à fonctionner dans des conditions an-ormales...

PERFORMANCE : Evaluation par mesure, analyse et simulation

ERGONOMIE : Facilité d'utilisation

Qualité Interne (point de vue de l'équipe de développement)

- **Maintenabilité (~60% du coût)**
  - maintenance **corrective**: pour éliminer les erreurs
  - maintenance **adaptative**: changement dans l'environnement
  - maintenance **perfective**: changement pour améliorer ses qualités
- **Réparabilité** ⇨ Peut facilement être corrigé
- **Evolutivité** ⇨ Peut s'adapter à des changements de spécification
- **Réutilisation** ⇨ Développer une étagère de composants
- **Portabilité** ⇨ Produit est indépendant du genre d'environnement
- **Interopérabilité** ⇨ **Capacité à interagir avec d'autres systèmes**
  - Protocole SOAP pour les services WEB
  - Sérialisation XML/Json
  - Couche internet

- **Qualité Externe** (point de vue de l'utilisateur)
- **Qualité Interne** (point de vue du développement)

- **Qualité du Processus de Développement**

  - **Productivité et réutilisation**
  - **Respect des délais**
  - **Echange d'informations**

    ⇨ Bonne documentation, accessible à tous

AUTOPSIE DE QUELQUES CATASTROPHES..

1962: La perte de Mariner I

Pour quelques lignes de FORTRAN..

```
DO 5 K = 1.3
     T(K) = W0
     Z = 1.0/(X**2)*B1**2+3.0977E-4*B0**2
     D(K) = 3.076E-2*2.0*(1.0/X*B0*B1+3.0977E-4*
               *(B0**2-X*B0*B1))/Z
     E(K) = H**2*93.2943*W0/SIN(W0)*Z
     H = D(K)-E(K)
5 CONTINUE
```

La première ligne est fausse, elle aurait dû s'écrire

```
DO 5 K = 1,3
```

Boucle avec K variant de 1 à 3

Au lieu de cela, le compilateur FORTRAN a compris

```
DO 5 K = 1.3
```

Boucle parcourue 1 fois avec K valant 1.3

⇨ ~~Fortran~~

⇨ Revoir les procédures de test

MARS CLIMATE ORBITER ($125M)

- Le 23 septembre 1999 à 11h27, la NASA perd tout contact avec la sonde spatiale 'Mars Climate Orbiter'
- La sonde s'est désintégrée dans l'atmosphère de Mars en essayant de se mettre en orbite à une hauteur de 53 km au lieu des 193 km qui étaient planifiés

L'ORIGINE DE LA PANNE..

- Lockheed Martin Astronautics a utilisé les mesures anglo-saxonnes ;
- Jet Propulsion Laboratory a utilisé le système métrique ;
- Personne n'a converti les données échangées (1 livre = 4.48 Newton).

ERREUR DANS LES PROCESSUS DE VALIDATION DE LA NASA

Rapport de la NASA (2000) 

- L'erreur de navigation n'a pas été détectée par la simulation informatique de la propulsion
- L'équipe opérationnelle de navigation n'était pas assez informée de la manière dont la sonde était orientée dans l'espace
- Certains canaux d'information entre groupes d'ingénieurs étaient trop informels
- Le personnel n'était pas assez entraîné à remplir les formulaires formels d'anomalies

AUTRES ÉCHECS RETENTISSANTS

- En 1972, en France, 72 ballons contenant des instruments de mesure détruits à cause d'un défaut dans le logiciel
- En 1981, le premier lancement de la navette spatiale retardé de deux jours à cause d'un problème logiciel.

La navette a d'ailleurs été lancée sans que l'on ait localisé exactement le problème (mais les symptômes étaient bien délimités)

- 4 juin 1996 : explosion de la fusée européenne Ariane 5
  - Coût : un demi-milliard de dollars (non assuré !)
  -  Faute logicielle d'un composant dont le fonctionnement n'était pas indispensable durant le vol