

Introduction

Information générale

- *Cours:* **Génie logiciel**
IFT 2255 – Section A (TH) – 1275
- *Crédits:* 3
- *Préalables:* IFT 1025 ou IFT 1020
- *Lieu des cours:* Mardi 12:00 – 14:00
Vendredi 12:00 – 13:00
A distance
- ***Lieu des démos:* Vendredi 13:00 – 15:00**
A distance

Coordonnées

- *Professeur:* **Omar Benomar**
- *Courriel:* omar.benomar@umontreal.ca
- *Heures de bureau:* Mardi et Vendredi 12:00 – 13:00

- *Auxiliaire:* **Khalid Belharbi**
- *Courriel:* khalid.belharbi@umontreal.ca
- *Heures de bureau:* Vendredi 13:00 – 14:00

- *Auxiliaire:* **Song Yang**
- *Courriel:* song.yang.1@umontreal.ca
- *Heures de bureau:* Vendredi 13:00 – 14:00

Évaluation

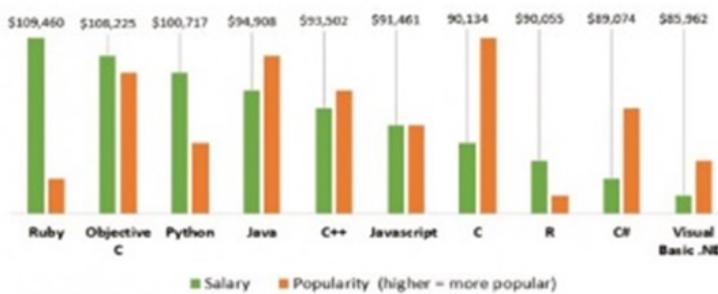
- **Devoirs maison 60%** (pondéré selon la difficulté)
 - 3 devoirs (~3 semaines chaque) en groupe
 - Le langage de programmation utilisé sera Java quand nécessaire
 - Tous les travaux seront évalués selon (par ordre d'importance):
 - L'exactitude
 - La quantité de travail
 - La structure et l'état complet
 - La présentation
 - Soumission: tous les devoirs sont dus à la date correspondante avant 23:55
 - Il y a 5% de pénalité par jour en retard, jusqu'à 2 jours au maximum, après quoi le devoir sera refusé
- **Tests 40%**
 - 4 tests comptant pour 10% chacun
- Il n'est pas possible de refaire ou faire plus de travail pour augmenter sa note

Notes

Catégorie	Appréciation	Note				
Excellent	Exceptionnel	A+	4.3	100	90	32%
	Parfait	A	4	89	85	
	Excellent	A-	3.7	84	80	
Très bon	Très bien	B+	3.3	79	77	47%
	Bien	B	3	76	73	
	Satisfaisant	B-	2.7	72	70	
Bon	Moyen	C+	2.3	69	67	18%
	Acceptable	C	2	66	63	
	Suffisant	C-	1.7	62	60	
Passable	Marginal	D+	1.3	59	57	3%
	Passable	D	1	56	53	
Échec	Faible	E	0.5	52	35	0%
	Nul	F	0	34	0	
	Absent	F*	0			

Pourquoi Java?

Programming Languages 10 Highest Salaries and Popularity



JAVA

One of the biggest reasons Java is so popular today is because it's a favorable option for client-server web applications. There are approximately 9 million Java developers - many of them can expect to earn nearly \$100,000.



This is a server-side interpreted compiled language, using a virtual machine. It is not JavaScript, and is not related to it.



Java was developed in 1995, and is one of the oldest programming languages on the web.



Java lets you:
Play online games



Upload photos



Take virtual tours



Use interactive maps

Java Jobs



Average Salary:
\$95,000

Job Count:
66,485



Top Cities:
New York
Washington D.C.
San Jose



Top Employers:
Amazon
IBM
eBay

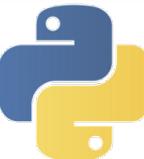
Tidbits of Wisdom



Users can disable Java on their machines.

Java is the basis of Android

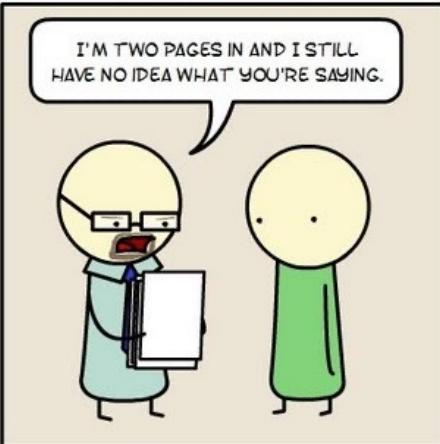
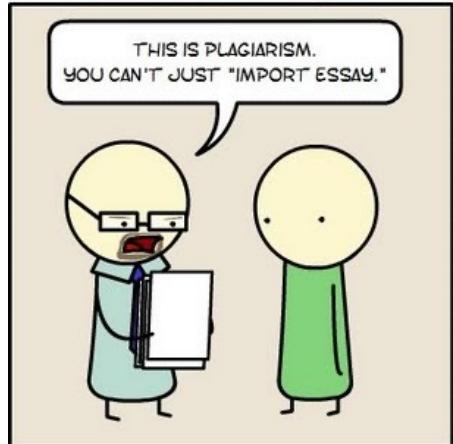
Slow to change, so it's easier to keep up with

 Java pour les programmeurs Python 

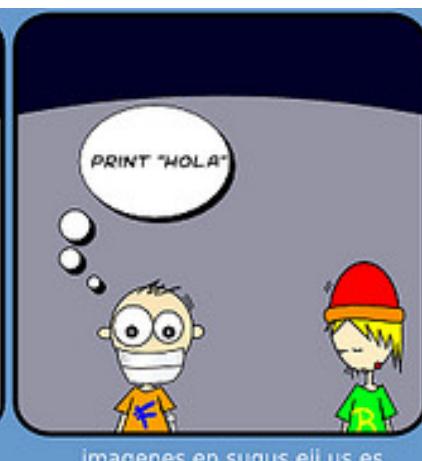
- <http://home.wlu.edu/~lambertk/pythontojava/>

PYTHON

JAVA



Traduction:
"Il est évident que les femmes pensent en Java et les hommes en Python"

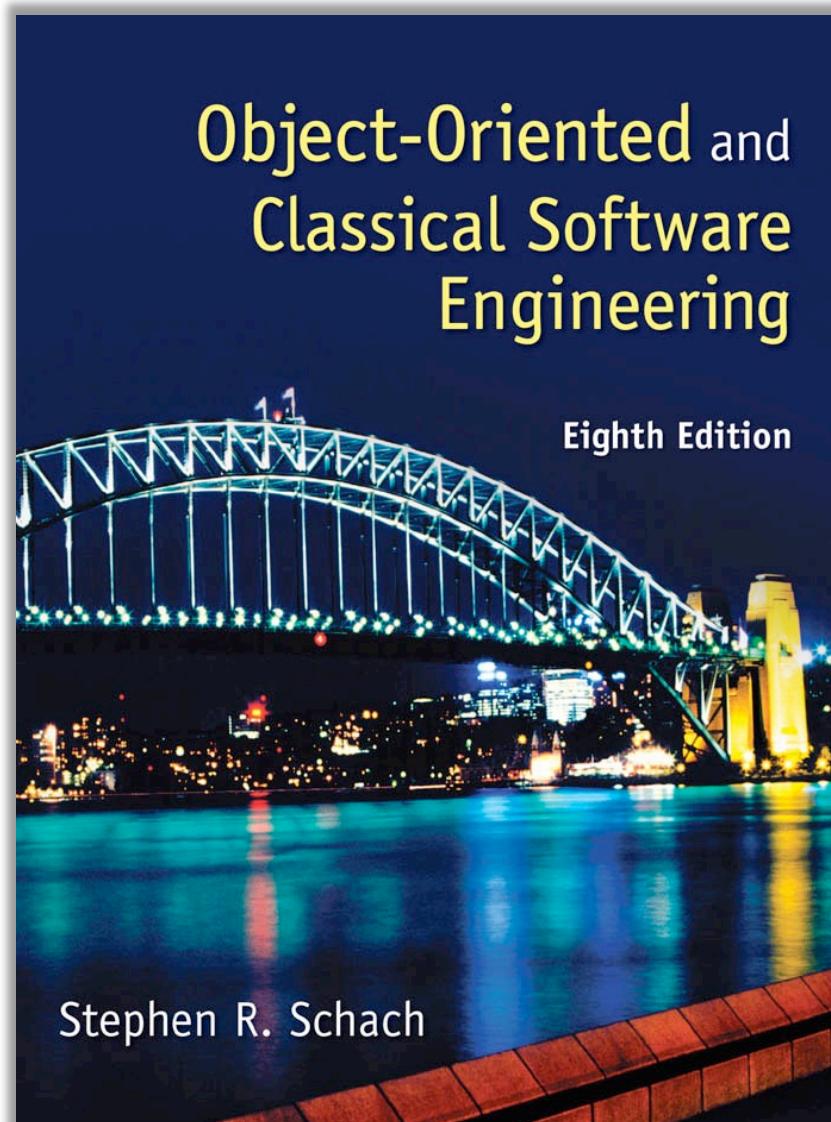


Les règles du jeu

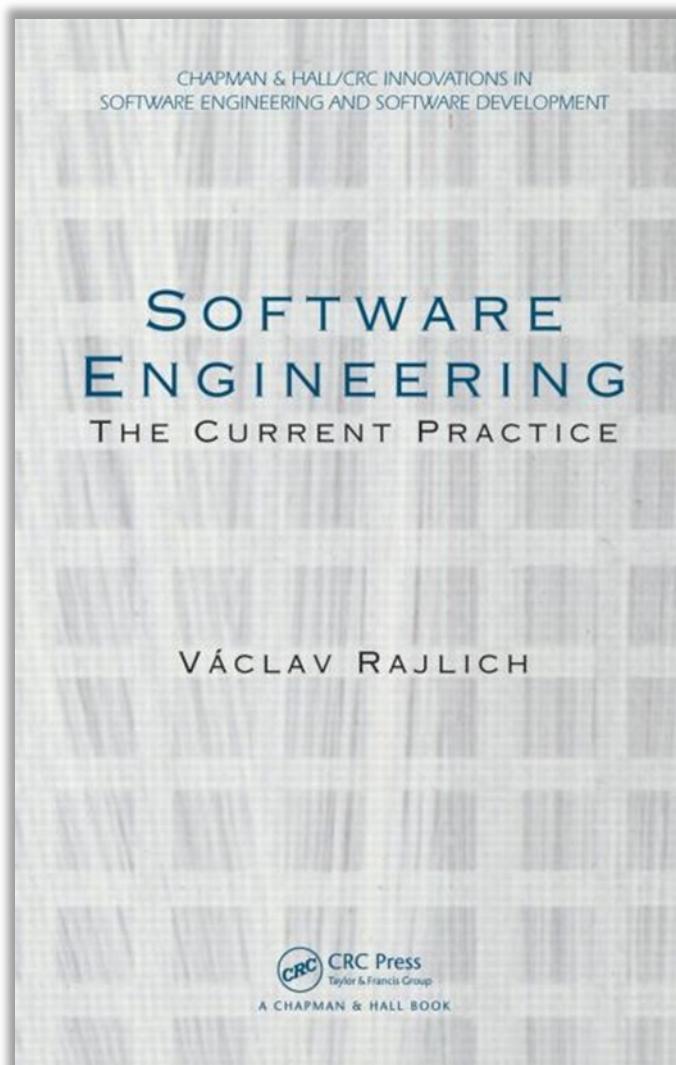
- Travail original
 - Vous êtes encouragé à vous entraider pour comprendre vos idées afin de solutionner les devoirs, mais chaque étudiant est tenu de soumettre son propre travail original. Remettre un travail qui n'est pas le vôtre, comme si ce l'était, est du plagiat. Toute réutilisation, collaboration ou inspiration doit être mentionnée *de manière explicite* dans vos devoirs rendus.
- Note sur l'intégrité, le plagiat ou la fraude
 - Tous les étudiants sont invités à consulter www.integrite.umontreal.ca et à prendre connaissance du Règlement disciplinaire sur le plagiat ou la fraude concernant les étudiants. Plagier peut entraîner un échec, la suspension ou le renvoi de l'Université.
- Présence en classe
 - Vous devez assister à tous les cours. De l'information supplémentaire sera disponible sur StudiUM, mais ne substitue en aucun cas votre participation en classe.
 - Si vous manquez un cours, vous êtes responsable de connaître tout le contenu et les annonces que vous auriez pu manquer
 - Vous serez évalué en majorité sur ce que je présente en classe
- Politique du cellulaire



Livre à utiliser tout au long du cours



Livre optionnel



StudiUM

Informations, matériels et communications

Université de Montréal | StudiUM

IFT2255-A-E21

- Participants
- Badges
- Compétences
- Notes
- Téléchargement
- Introduction
- Semaine du 4 mai:
Introduction, Vie du Logiciel et Modèles de développement
- Semaine du 11 mai: Processus de développement, Logiciel en équipe et Versionnement de Logiciel
- Semaine du 18 mai: Exigences, Cas d'utilisation
- Semaine du 25 mai: Analyse
- Semaine du 1 juin: Conception, Modélisation
- Semaine du 8 juin:
Modélisation, Architecture
- Semaine du 15 juin: Paradigme Orienté Objet
- Semaine du 22 juin: Patron de conception, Implémentation
- Semaine du 29 juin:
Implémentation, Vérification et Validation

Accueil / Mes cours / IFT2255-A-E21

IFT2255-A-E21 - Génie logiciel

Syllabus de cours

Annonces

Discussions Devoirs

Discussions Cours

Lien Zoom pour le cours

Lien Zoom pour les séances de démos

Code d'honneur aux étudiant(e)s de l'UdeM

Je suis inscrit(e) au cours en ligne sur StudiUM et dans mon utilisation des outils de formation en ligne de l'Université de Montréal, je m'engage à respecter le code d'honneur suivant.

Je m'engage à :

- ne posséder qu'un seul compte d'utilisateur et à ne pas laisser d'autres personnes l'utiliser;
- respecter les conditions d'utilisation énoncées sur StudiUM qui traitent des droits et obligations des personnes inscrites sur le site;
- faire les tests et examens moi-même, sans l'aide d'autres personnes, sauf si le travail en équipe est expressément mentionné;
- ne remettre que mes propres travaux, qui ne contiennent aucun matériel plagié, en tout ou en partie, et à citer correctement mes sources;
- donner accès à l'équipe enseignante à mes données et travaux pour les besoins du cours;
- ne pas participer à des activités malhonnêtes visant à améliorer mes résultats ou à nuire aux résultats d'autres personnes participant à la formation.

»

Mes COURS

IFT2255-A-E21 - Génie logiciel

IFT2255-A-A20 - Génie logiciel

IFT2255-A-E20 - Génie logiciel

IFT3913-A-H15 - Qualité du logiciel et métriques

IFT1945-A-E13 - Internet et création de pages Web

Aide StudiUM

Formation - Non à la violence à caractère sexuel (Membres du personnel)

Tous les cours ...

Accès rapide

Visibilité du cours

Ce cours est présentement ouvert aux étudiants.

Fermer le cours

Calendrier provisoire

Cours	Demos	Commentaires
Introduction, Vie du Logiciel et Modèles de développement	Pas de démos	ch.1, ch.2 et Raj. ch.1, ch.2-3
Processus de développement, Logiciel en équipe et Versionnement de Logiciel	Équipes + outils nécessaires	ch.3, ch.4, ch.5.9-5.11 et Raj. ch3.3
Exigences, Cas d'utilisation	Contrôle de revisions + Besoins	ch.11-11.4, ch.11-11.4 (Devoir 1)
Analyse	Diagrammes de cas d'utilisation et d'activités	ch.17.4-17.5, ch.13
Conception, Modélisation	Diagrammes de classes Remise DM1 (DM2)	ch.14, ch.17
Modélisation, Architecture	Diagrammes de séquence	ch.7-7.3, ch.8-8.5.2, ch.8.11-8.13 (Devoir 2)
Paradigme Orienté Objet	Architecture	ch.7.4-7.9
Patron de conception, Implémentation	Patrons + Support DM2	ch.8.6-8.8, ch.15
Vérification et Validation	Correction DM2 + Implémentation	ch.6, ch.15.9-15.23 (Devoir 3)
Tests unitaires, Déploiement	JUnit	ch.16
Maintenance	Maintenance +	Raj. ch.5-7, ch.8-11

Démonstrations

- Très utiles!
- Révision du cours par des exercices théoriques
- Préparation pour vos devoirs par des exercices pratiques
 - Matériel complémentaire essentiel pour les devoirs
- Révision avant chaque examen
- Correction de chaque devoir

- Nous supposons que vous assistez à toutes les séances
 - Clarifications sur les devoirs
 - Savoir utiliser les outils

Ce à quoi vous attendre de ce cours

- Comprendre le **processus de développement** d'un logiciel
- Cueillir et documenter les **besoins** et les **spécifications** d'un produit logiciel
- Concevoir un système orienté objet en **UML**
- Implémenter un logiciel en programmation **orientée objet**
- Être capable de **tester** un logiciel
- Pouvoir incorporer des **changements** à un logiciel après sa **livraison**

Ce que j'attends de vous

- Le niveau d'un étudiant de 2^e année en informatique
- Vous comprenez la base de la programmation informatique
 - Vous pouvez écrire un logiciel qui résout un problème
 - Vous savez comment déboguer et vérifier votre code
- Vous pouvez trouver l'information par vous-même et l'appliquer pour résoudre votre problème
 - Savoir lire du matériel technique, en extraire l'essentiel et l'appliquer
- En général vous devez investir 2 à 3 heures hors cours pour chaque heure de cours assistée
- Pour réussir, vous devez
 - Lire le matériel couvert: en cours et dans le livre
 - Se préparer pour chaque cours, assister aux cours et aux démonstrations
 - Participer activement aux réunions de votre équipe
- Discuter avec moi, les auxiliaires et entre vous sur des sujets de ce cours

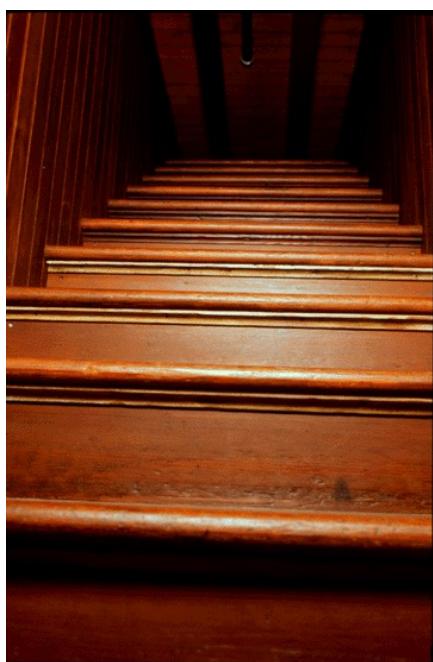
Le génie logiciel

Mauvaise ingénierie donne lieu à...



Maison mystère de Winchester
<http://www.winchestermysteryhouse.com/>

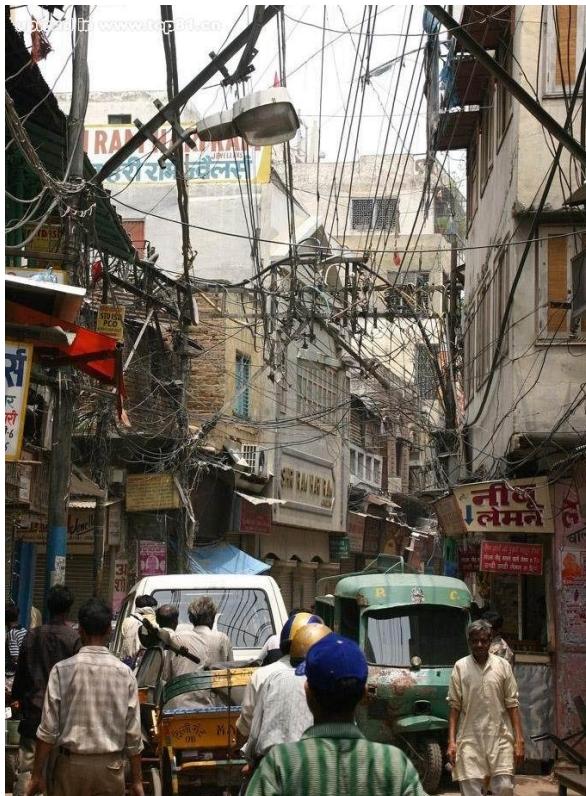
Le résultat de construire
continuellement sans avoir pris la
peine de concevoir adéquatement.



Résultat:

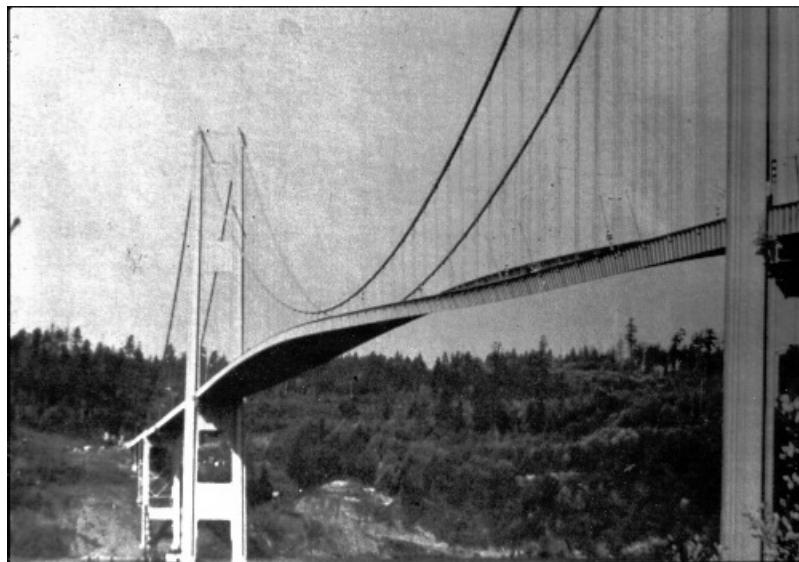
- Escaliers qui montent au plafond
- Fenêtres au milieu des chambres
- Portes qui s'ouvrent sur un mur

Impossible à maintenir!



- Comment maintenir si quelque chose ne fonctionne plus?
- Comment ajouter de nouvelles connexions ou fonctions?

Des conséquences désastreuses



<http://www.nwrain.net/~newtsuit/recoveries/narrows/narrows.htm>

Même dans le logiciel

- Simulateur de F16 (1986)

- L'avion faisait volte-face à chaque fois qu'il traversait l'équateur



- Ariane 5 (1996)

- Dépassement de capacité dans la conversion d'un entier de 64 bits à 16 bits qui a activé le module d'autodestruction

- Panne de courant en Amérique du nord (2003)

- Problème de concurrence a paralysé le système d'alarme durant plus d'une heure



- Toyota Prius (2015)

- Paramètres du logiciel surchauffent le moteur qui éteint le système hybride durant la conduite



Le logiciel

- Le logiciel est omniprésent



- Synonymes: programme, application
- Les personnes qui développent le logiciel
 - Ingénieurs logiciel, développeurs logiciel, programmeurs
 - Possèdent des talents et utilisent des outils qui permettent de développer et faire évoluer les logiciels
 - Talents: créativité, logique, mathématique, méthodologie, résolveur de problèmes
 - Outils: d'autres logiciels (ex: IDE)

Caractéristiques du logiciel

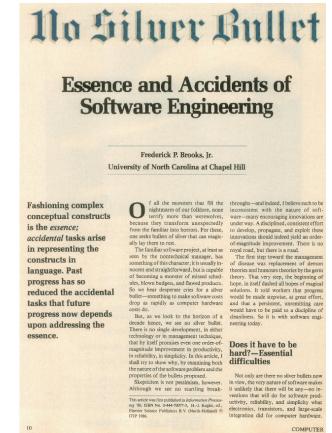
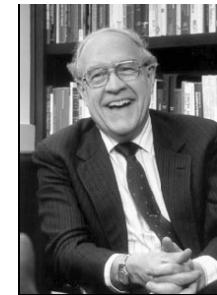
- Logiciel est ubiquitaire dans tous les domaines de métier
- Logiciel s'étend à tous les aspects de la vie humaine
 - Simple ou très complexe
 - Pour usage interne ou ciblé pour le grand public
- But
 - Dédié à une tâche spécifique, ex: gestion de paie
 - Application qui prend en charge toutes les fonctions d'une organisation
- Lieu
 - Exécuté dans un emplacement
 - Distribué sur plusieurs machines à travers le monde
- Exécution
 - Utilisation unique
 - En lot (*batch*), sans interaction
 - En temps réel, avec interaction humaine

Difficultés auxquelles font face les programmeurs

Fred Brooks (1987)

- **Complexité accidentelle:** plus facile à résoudre

- Due aux technologies utilisées
- Imprévisus de l'environnement
- Problèmes transitoires

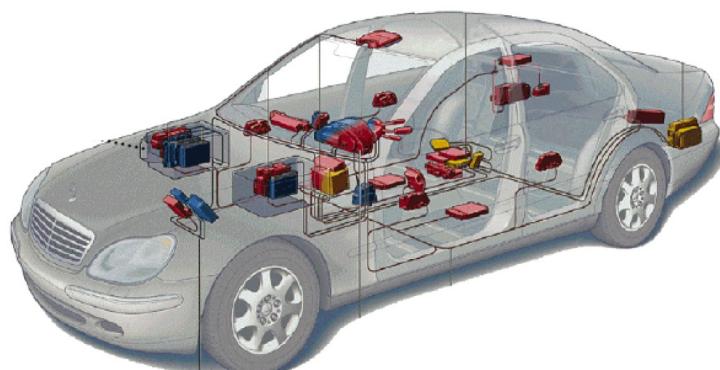
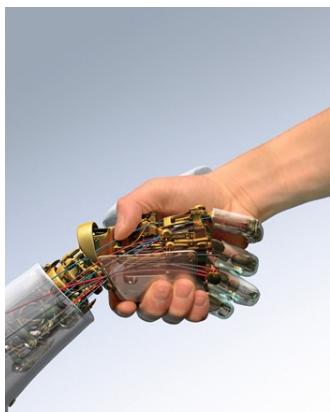


- **Complexité essentielle:** difficultés inhérentes difficiles à résoudre
 - Complexité, invisibilité, versatilité, conformité, discontinuité

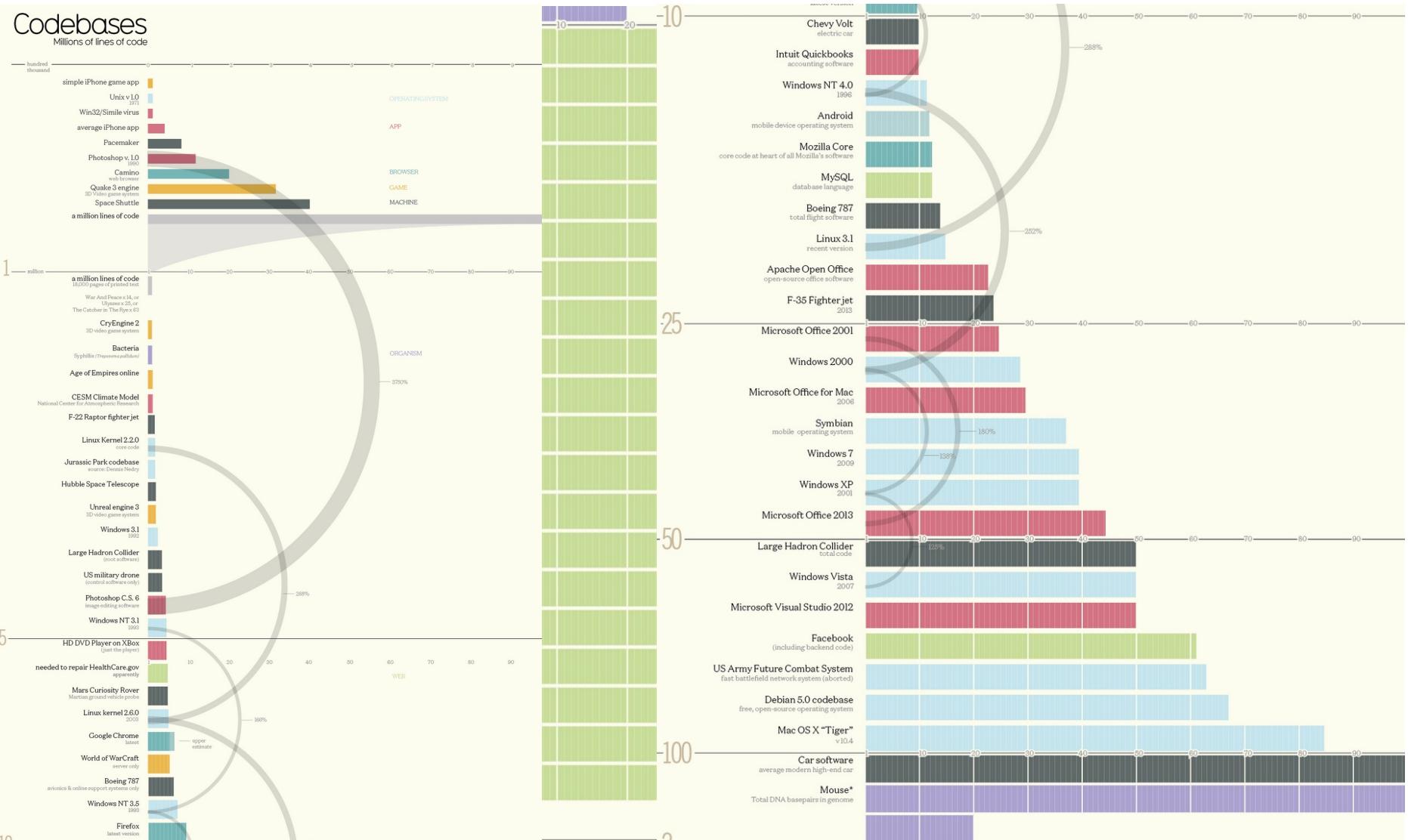


Complexité

- Les programmes sont parmi les systèmes les plus complexes jamais créés
- Notre mémoire à court terme peut accommoder ± 7 choses (Miller, 1956)

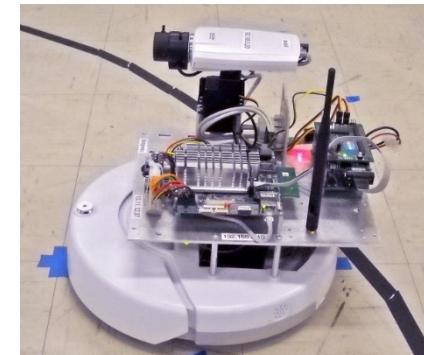


Programmes en 10^6 lignes de code



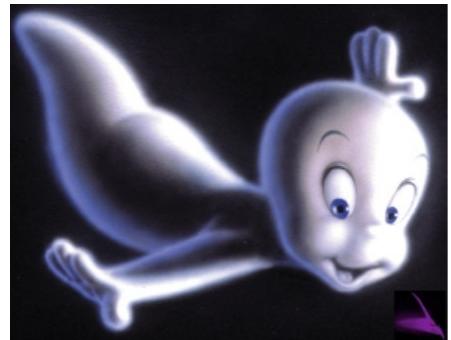
Conformité

- Gros logiciels composés de matériel hardware, d'usagers, d'interactions avec d'autres logiciels, etc.
 - *Cyber-Physical Systems* mélangent des composants physiques et logiciel, ex: robots
- Logiciel intègre toutes ces parties dans un seul système
 - Préserve **l'unité du domaine**
 - Communique avec les **composants hétérogènes** du domaine
- Logiciel incarne le domaine
 - Propriétés du domaine s'infiltrent dans le logiciel
 - Logiciel doit se conformer au domaine
 - Besoin de connaissance du domaine



Invisibilité

- Pouvez-vous toucher une classe?
- Logiciel est invisible
- Diffère des lois de la physique et des mathématiques (continues)
- Pas moyen de représenter un produit au complet
 - Vues complètes (ex: code) sont incompréhensibles
 - Vues partielles/incomplètes (ex: modèles) et peut-être illusoires
- Nos sens ne peuvent pas être facilement utilisés pour comprendre
 - Visualisation et sonification du logiciel très laborieux

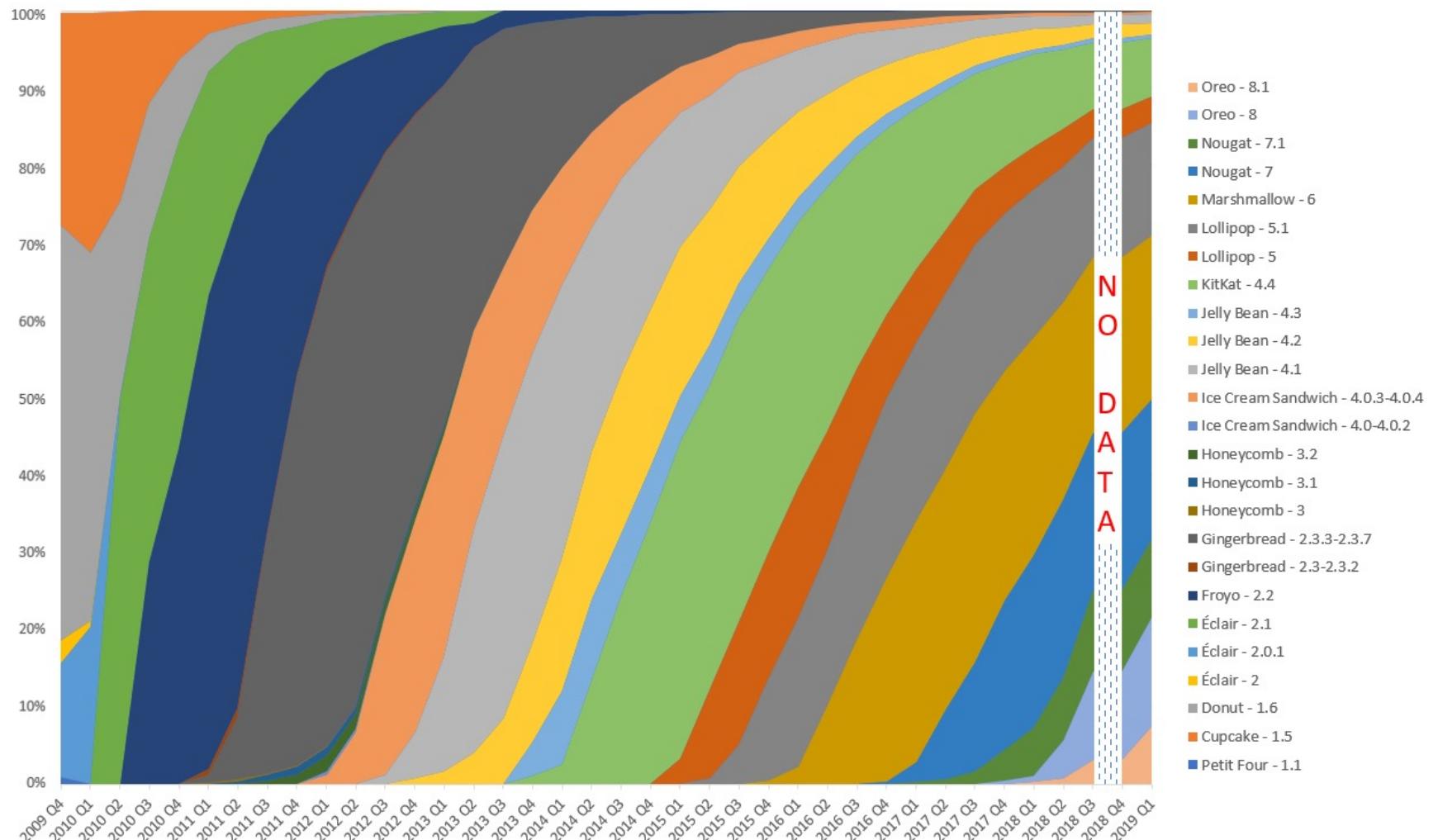


Versatilité

- Logiciels évoluent et changent constamment
 - Changement dans les besoins
- Ce qu'on savait hier peut être obsolète aujourd'hui
 - Nouvelle/meilleure techno émerge continuellement (loi de Moore)
- Le logiciel est facile à modifier
- Mais la défi est de le modifier **correctement**

Évolution en continu de logiciels

Distribution des versions Android sur les appareils



Discontinuité

- L'humain comprend facilement les systèmes linéaires ou semi-linéaires



- Les logiciels sont discontinus: petits changements en entrée résultent en un énorme changement en sortie



Origines du génie logiciel

- Jusqu'au milieu des années 1960, logiciels construits à l'improviste
- Programmes codés par des experts de divers métiers
 - Ingénieurs matériels, mathématiciens
 - Le temps d'utilisation d'un ordinateur coûtait plus cher (600\$/h) que le salaire de ses opérateurs (2\$/h)
- Programmes devenaient de plus en plus complexes avec de nouvelles techno, des besoins qui évoluent et la diversité des programmeurs
- Il n'y avait pas de méthodologie pour construire un logiciel et le **changer**
 - Coder puis corriger (*code-and-fix programming*)

Naissance du génie logiciel

- En 1968, la conférence de l'OTAN se réunit pour discuter d'un nouveau domaine: le génie logiciel (GL)
 - Rendre GL une discipline à part entière
 - Suivant les mêmes méthodologies que les disciplines de génie traditionnelles
- Dijkstra, Naur, Bauer, Perlis, Gries, McIlroy, Randell
- Ils proposent des recommandations sur comment développer du logiciel

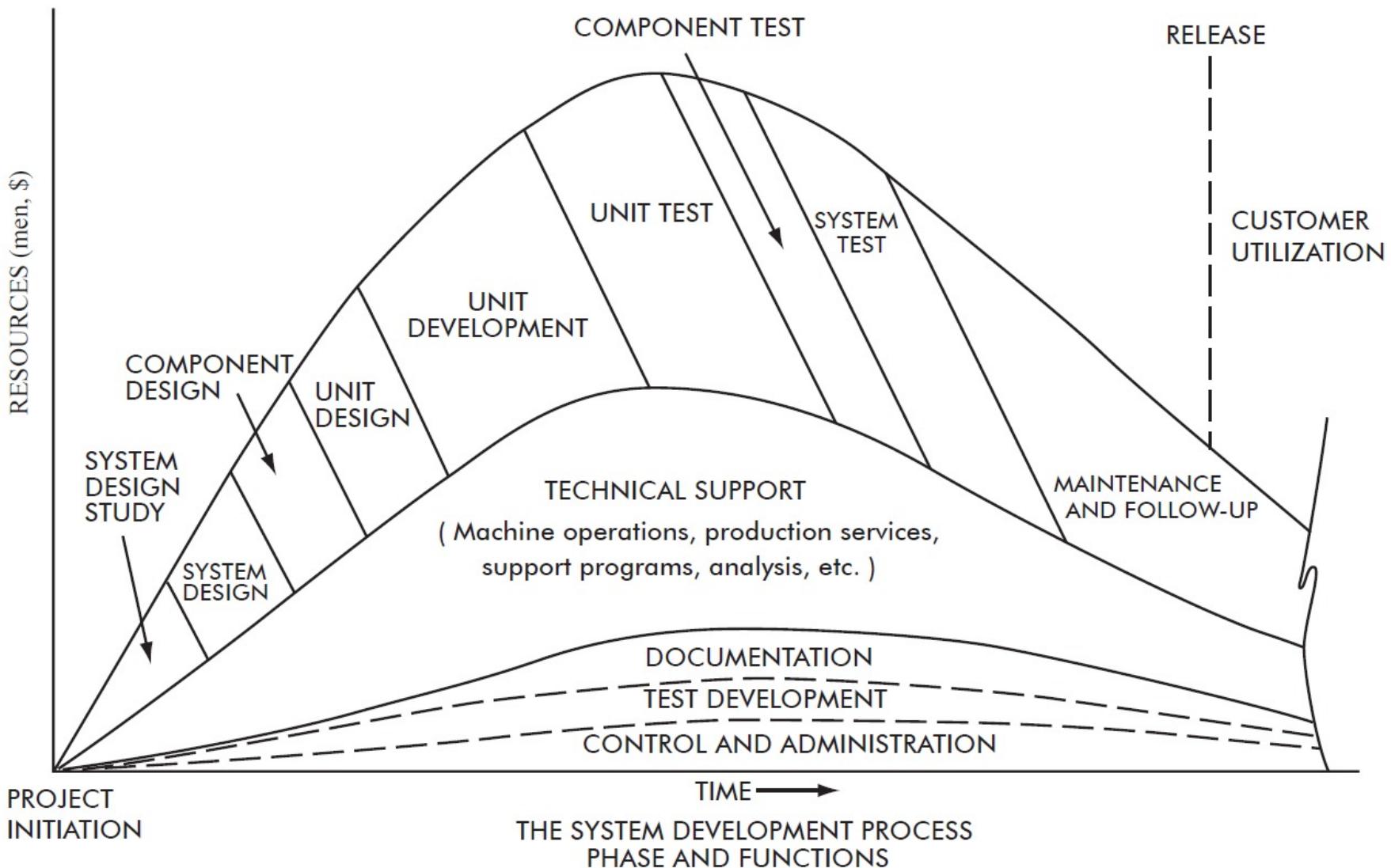


Définition originale du génie logiciel (*Bauer*)

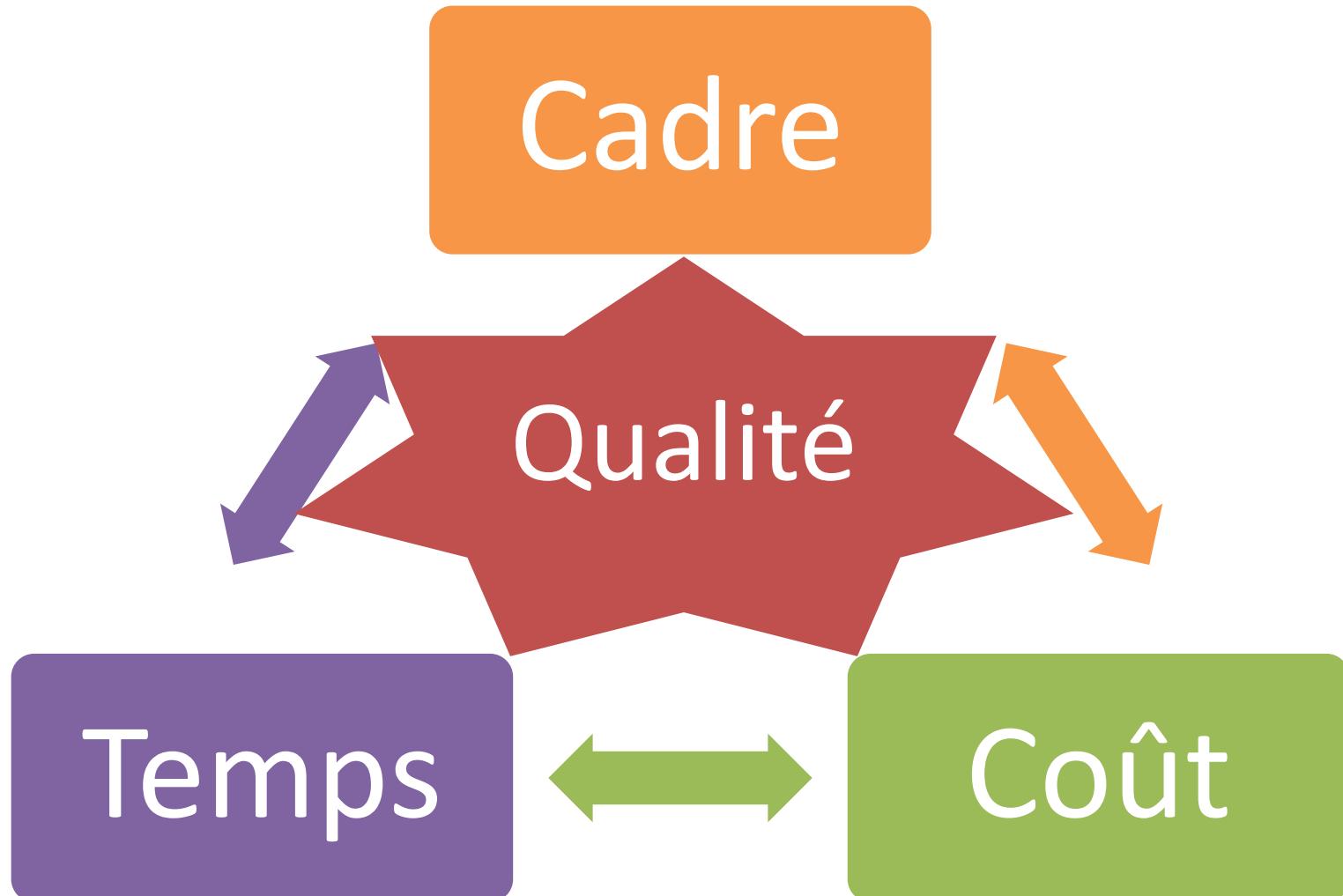
“[Software engineering is] the establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines”

« [Le génie logiciel est] l'établissement et l'utilisation de principes du génie afin d'obtenir un logiciel économique, fiable et qui fonctionne de manière efficace sur de vraies machines. »

Problème des projets logiciels (Nash)



Les forces d'impact du logiciel



Défis de l'ingénieur logiciel

- Logiciel doit correspondre aux **besoins** du client
 - Tout en fournissant une solution **efficace** au problème
- Logiciel doit être **rentable**
 - Coûts et temps de développement
 - Exigences croissantes
- La solution doit être de **haute qualité**
 - Réduire efforts de maintenance



Génie logiciel

Les 5 grandes activités d'un ingénieur logiciel

- **Décrire:** besoins, spécification de conception, documentation
- **Implémenter:** conception, programmation
- **Évaluer:** test, vérification, validation, révision
- **Gérer:** planification, échelonnage, communication
- **Faire fonctionner:** déploiement, installation, maintenance