

Pourquoi et comment le monde devient numérique

Gérard Berry

Chaire d'innovation technologique Liliane Bettencourt

Collège de France

17 janvier 2008

De grands bouleversements

Communication : Internet, tél. portable, tél. gratuit

Audio-visuel : MP3, photo/vidéo numérique, RDS, TNT, TVHD

Commerce : en ligne

Cartographie : cartes & photos interactives

Transports: GPS, pilotage, sécurité

Industrie: gestion, outillage, CAO, travail à distance

Sciences: modélisation et expérimentation numérique

Médecine: imagerie numérique, chirurgie robotique

Grande industrie qui irrigue toutes les autres

Les quatre piliers du numérique

1. La **numérisation** de l'information
2. La prodigieuse **machine à information**
3. La **science et la technologie** de sa conception et de son usage
4. Un espace d'**innovation** (presque) sans frein

Les grands jalons

30-45 : pionniers (von Neumann, Turing, Eckert,...)

55-75 : centres de calcul, scientifique, gestion

75-80 : mini-ordinateurs, algorithmique, programmation
télécommunications

80-95 : PC, informatique embarquée

explosion de l'industrie et des applications

95-05 : Internet, téléphone portable

la naissance des grands réseaux

05-15 : audiovisuel, informatique nomade, sans fil
sciences, médecine

15- : informatique ubiquitaire
objets dans le réseau

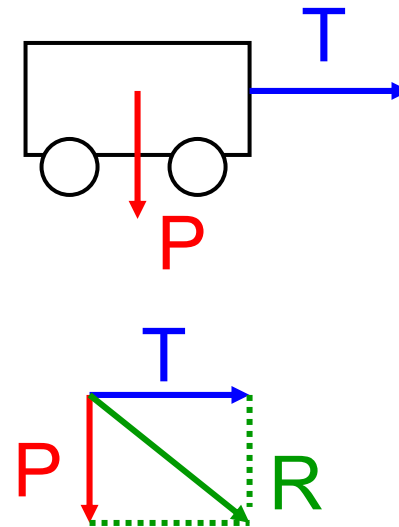
Alerte aux pucerons!



- infestation massive par pucerons enfouis partout
- qui peuvent communiquer entre eux
- mais de plus en plus d'applications **critiques**

Autrefois : *dépendance information / support*

Quel beau
texte!
Quelles belles
équations!



source Ressort Mécanique

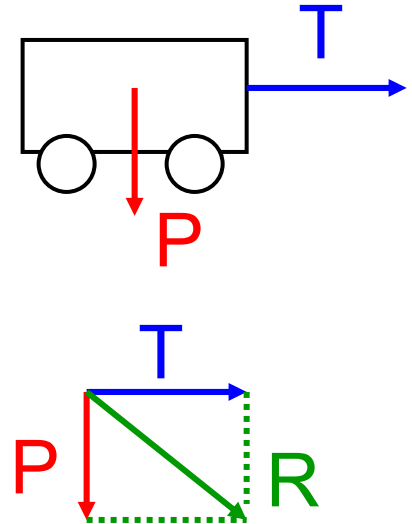
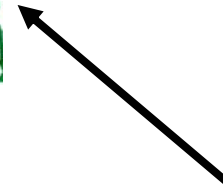
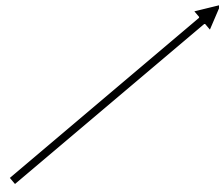
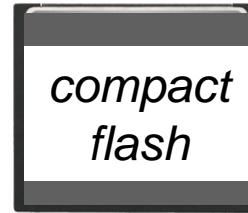
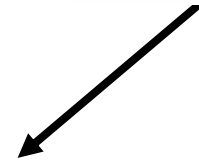
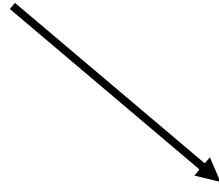


Maintenant : indépendance et convergence

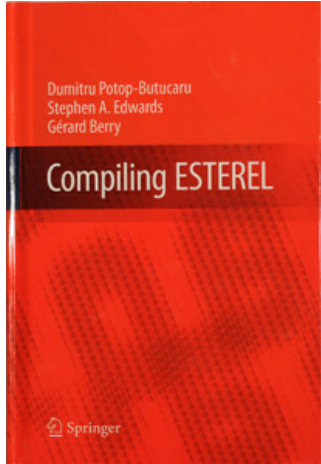
Quel beau
texte!
Quelles belles
équations!



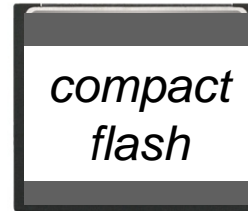
01100110111101
10010011101100



... sans abandon exagéré



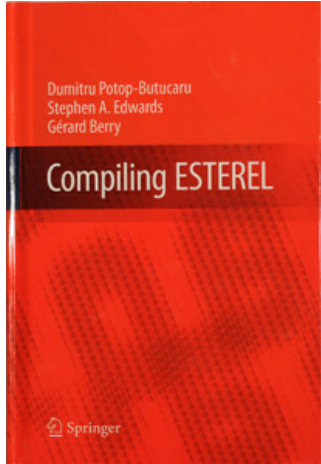
01100110111101
10010011101100



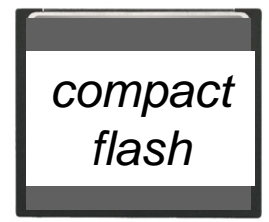
source Ressort Mécanique



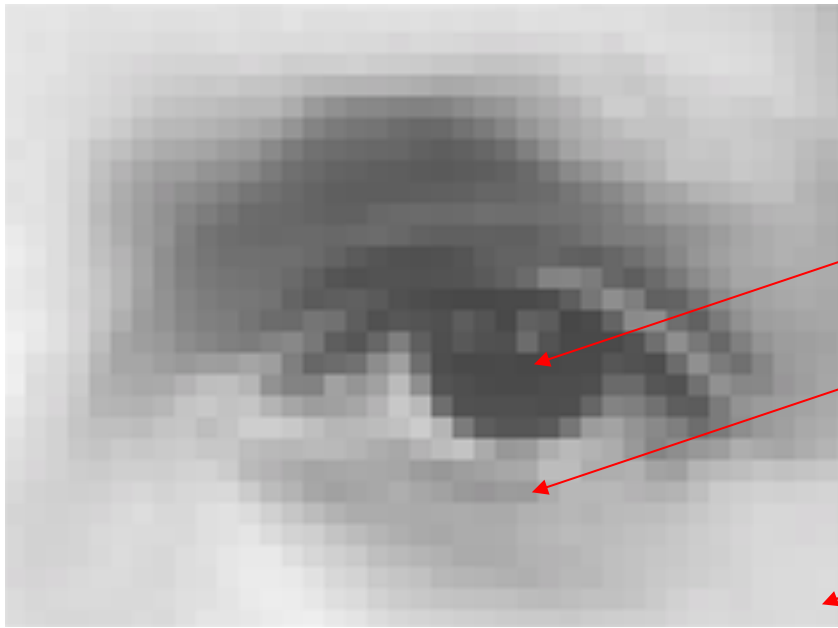
... sans abandon exagéré



01100110111101
10010011101100



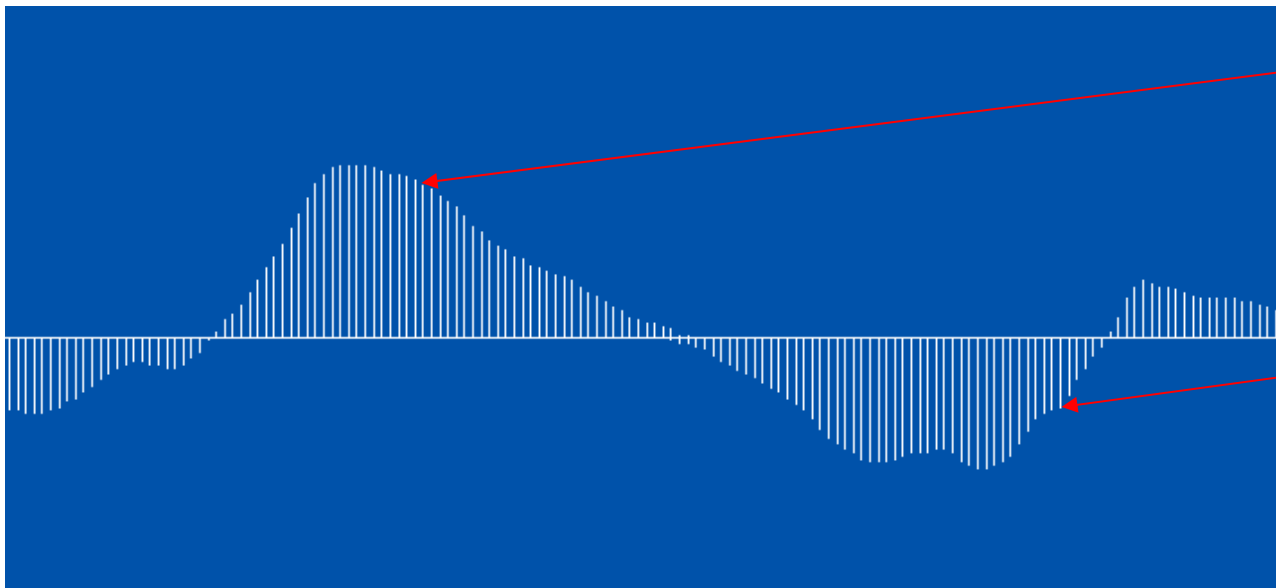
Source Berger Lahr



57

133

192



14427

-7322

Transporter et transformer l'information

- Algorithmes **génériques**
 - copier à l'infini sans aucune modification
 - stocker, diffuser, comprimer, crypter, etc.
- Algorithmes **spécifiques**
 - textes: recherche, correction, traduction (?)
 - sons, images: compression, amélioration
 - forces: mesures, contrôle
 - ...

Al Khuwārizmī

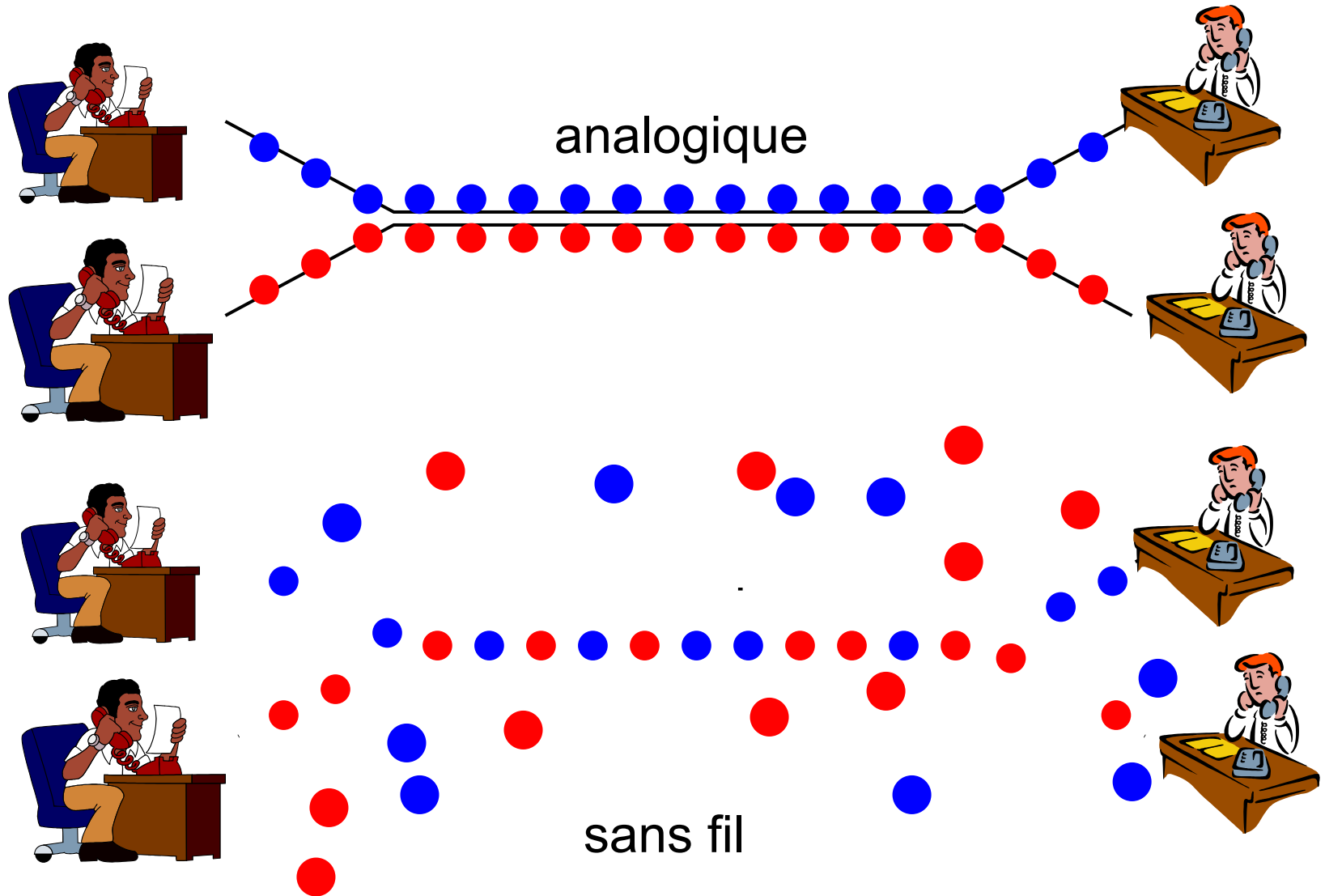
~ 783 - 850

algorithme
algèbre



Source Jean Vuillemin

La téléphonie numérique



La photographie numérique



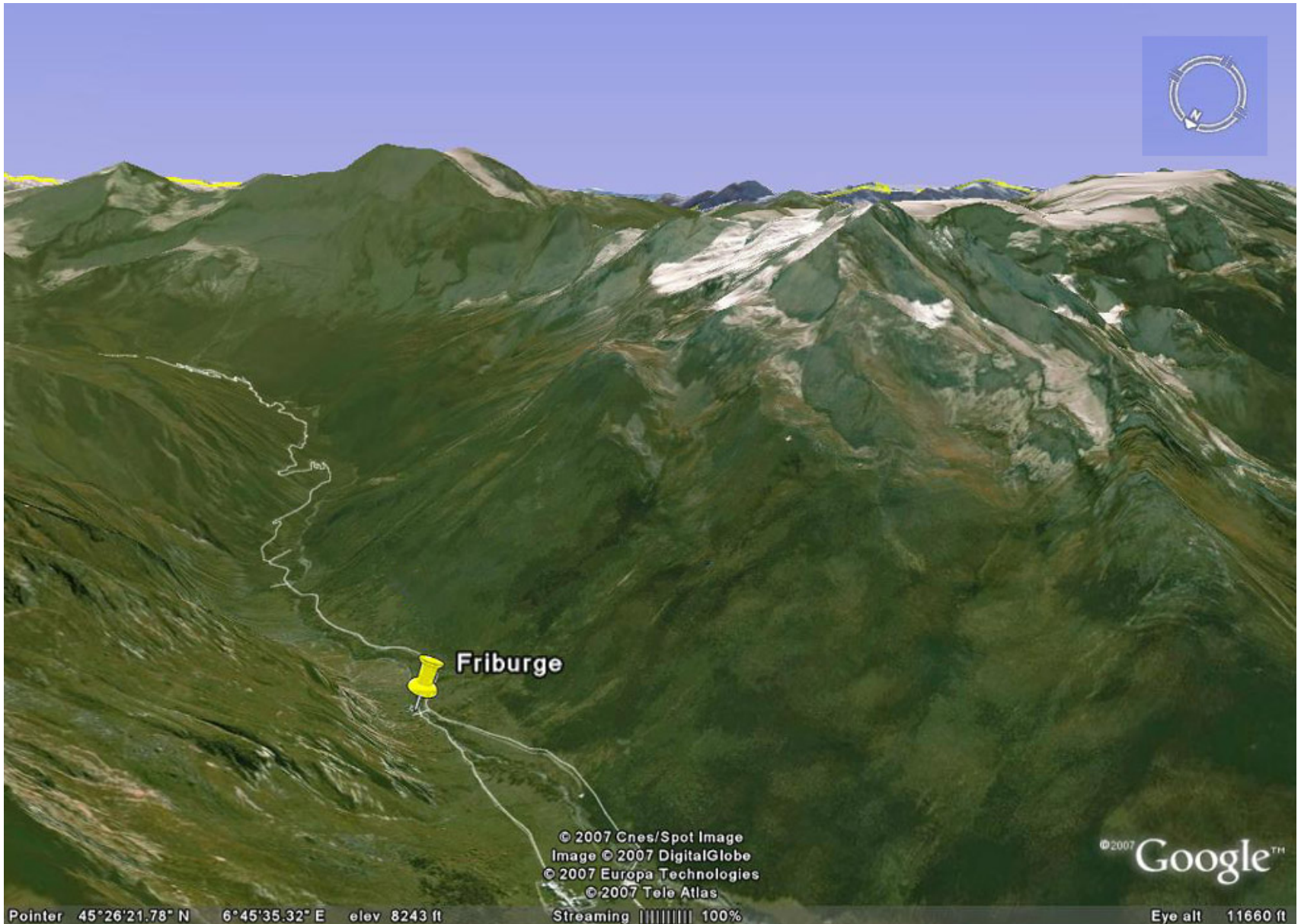
Source DxO Labs

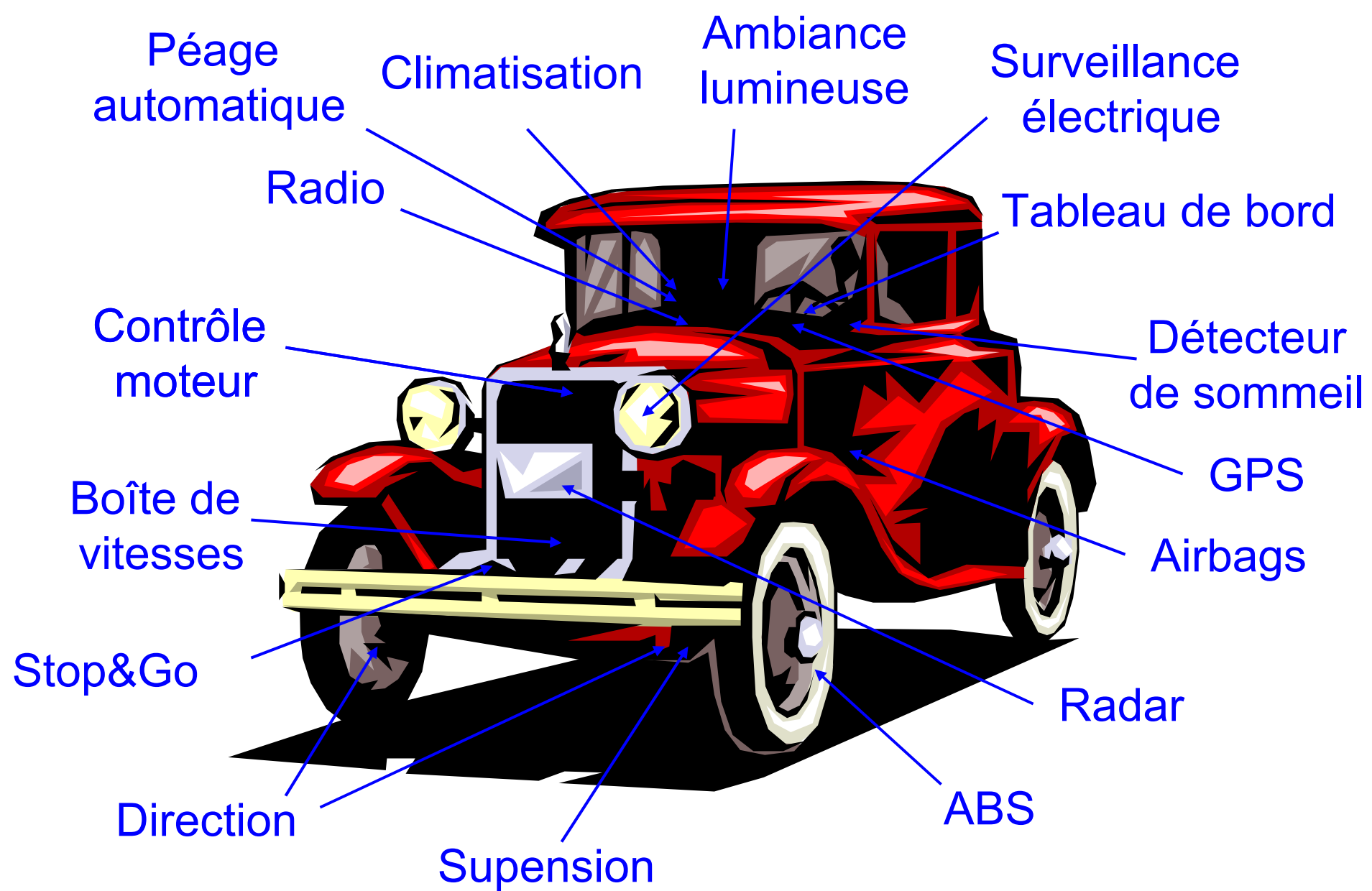
...une révolution, pas une substitution!



Source DxO Labs

La cartographie numérique





Coordination avec la route et les autres voitures

La machine à information

1. Le matériel

circuits, disques, réseaux, capteurs

2. Le logiciel

infrastructure ou applicatif

3. L'interface homme-machine

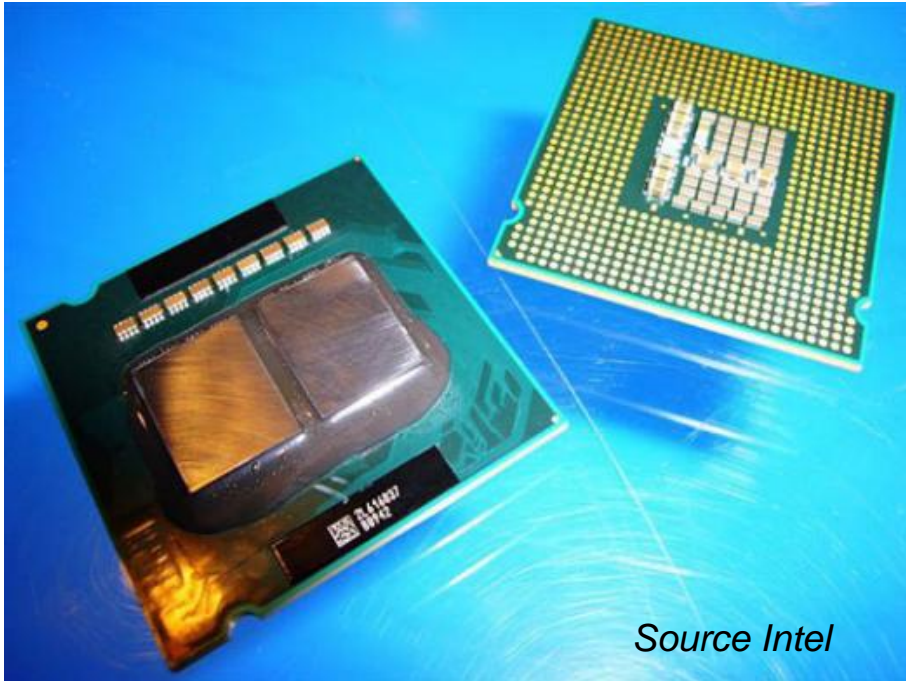
souvent un point de faiblesse

4. Les bugs

une malencontreuse spécificité !

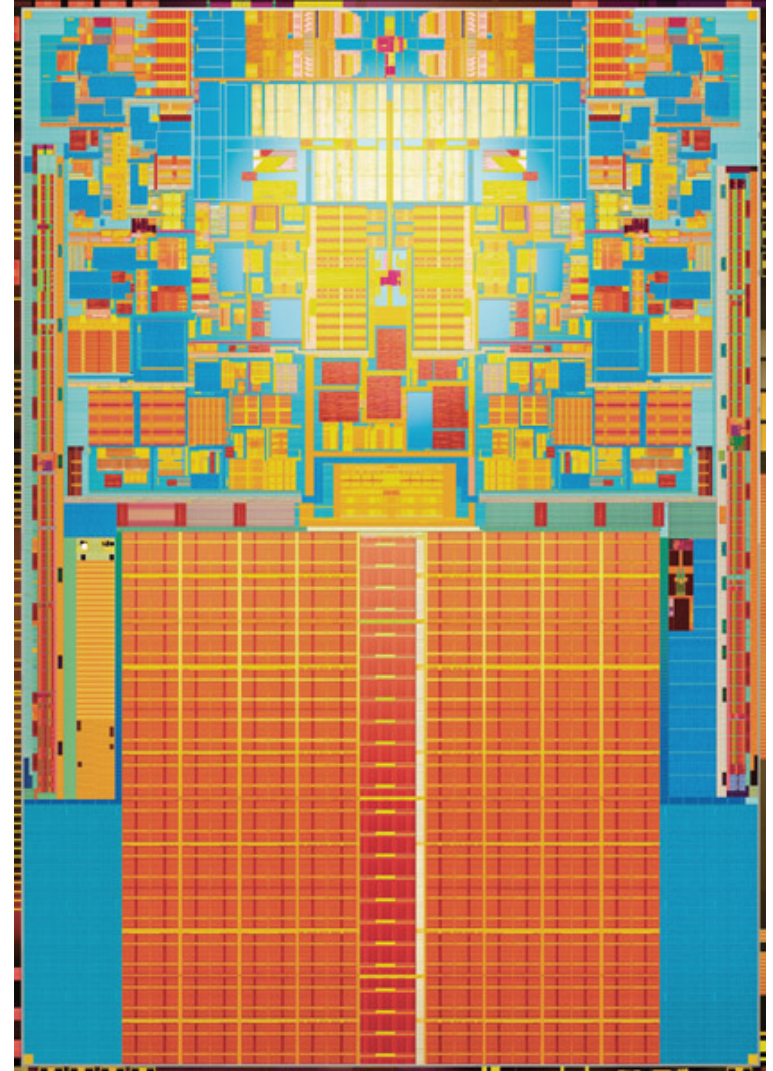
Pour réaliser une logique fondamentalement invisible

Les circuits



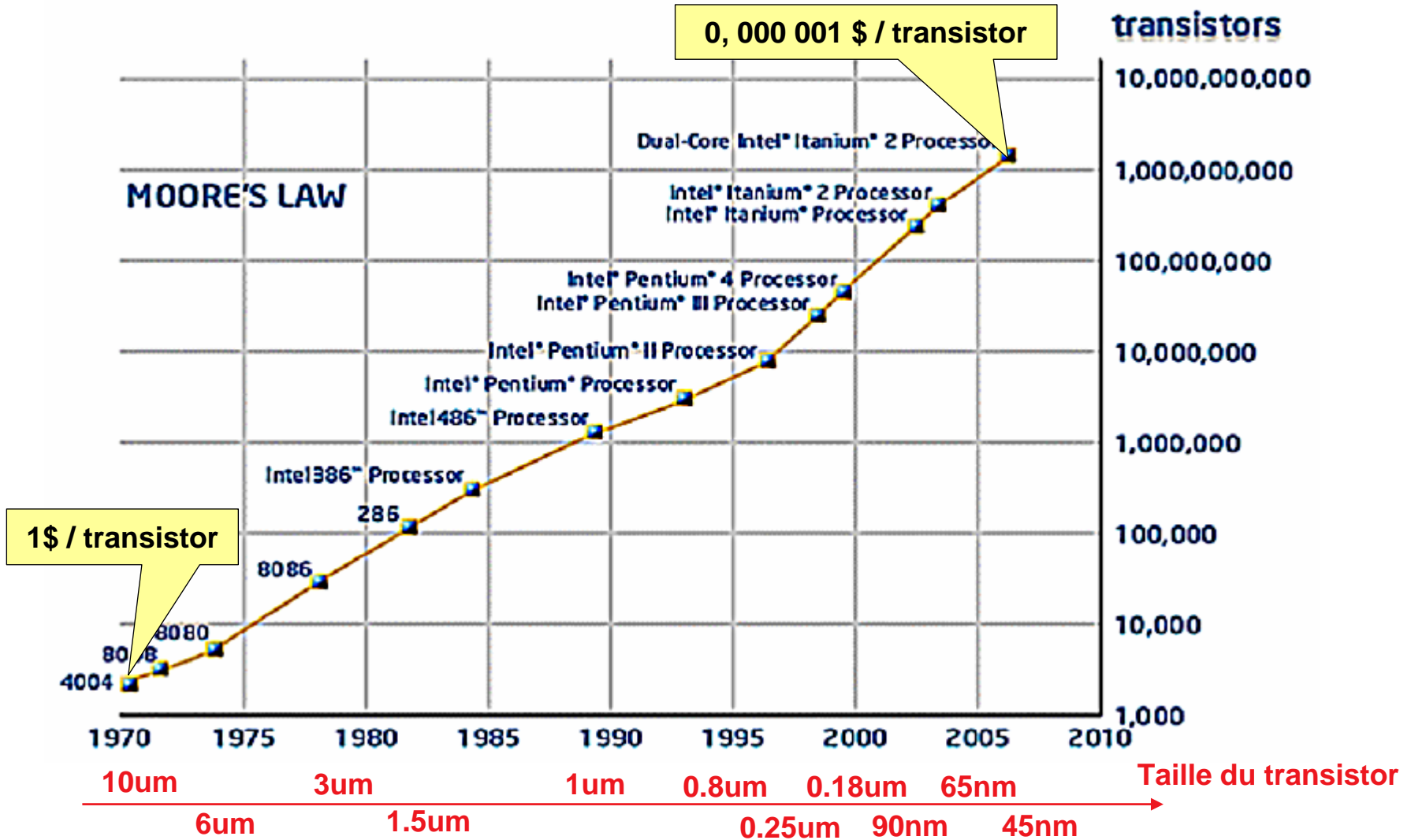
SoC = System on Chip

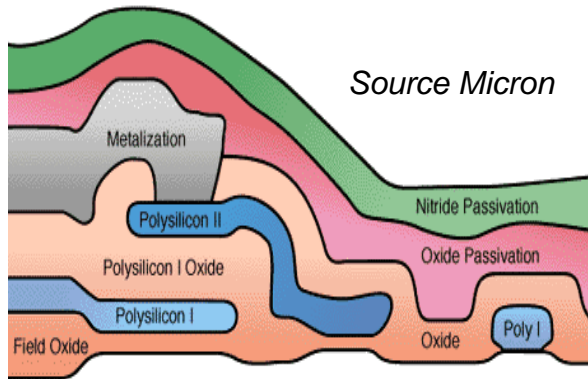
microprocesseurs de PC
téléphones, DVD, TV, GPS



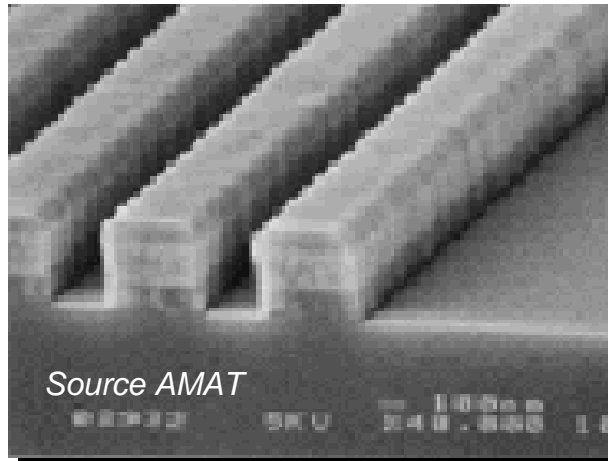
Source Intel

La loi de Moore

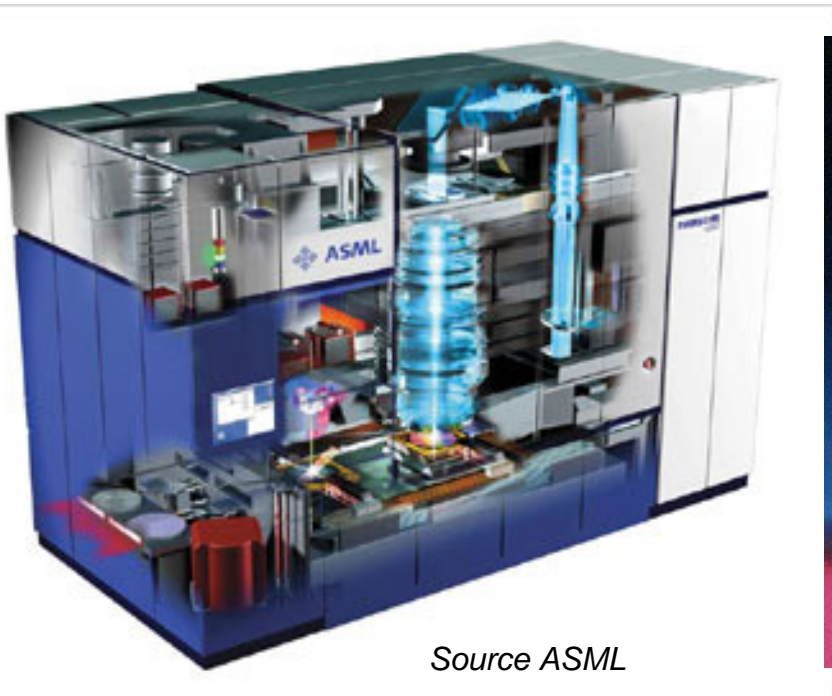
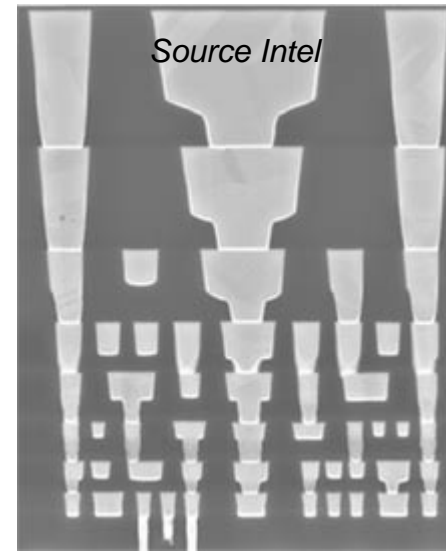




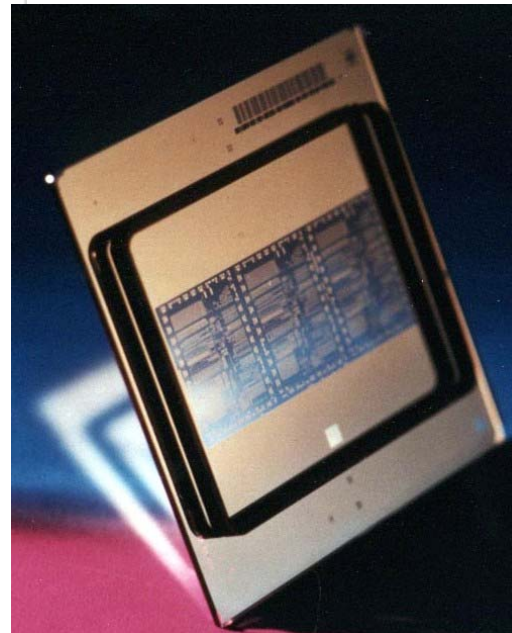
Source Micron



Source AMAT



Source ASML

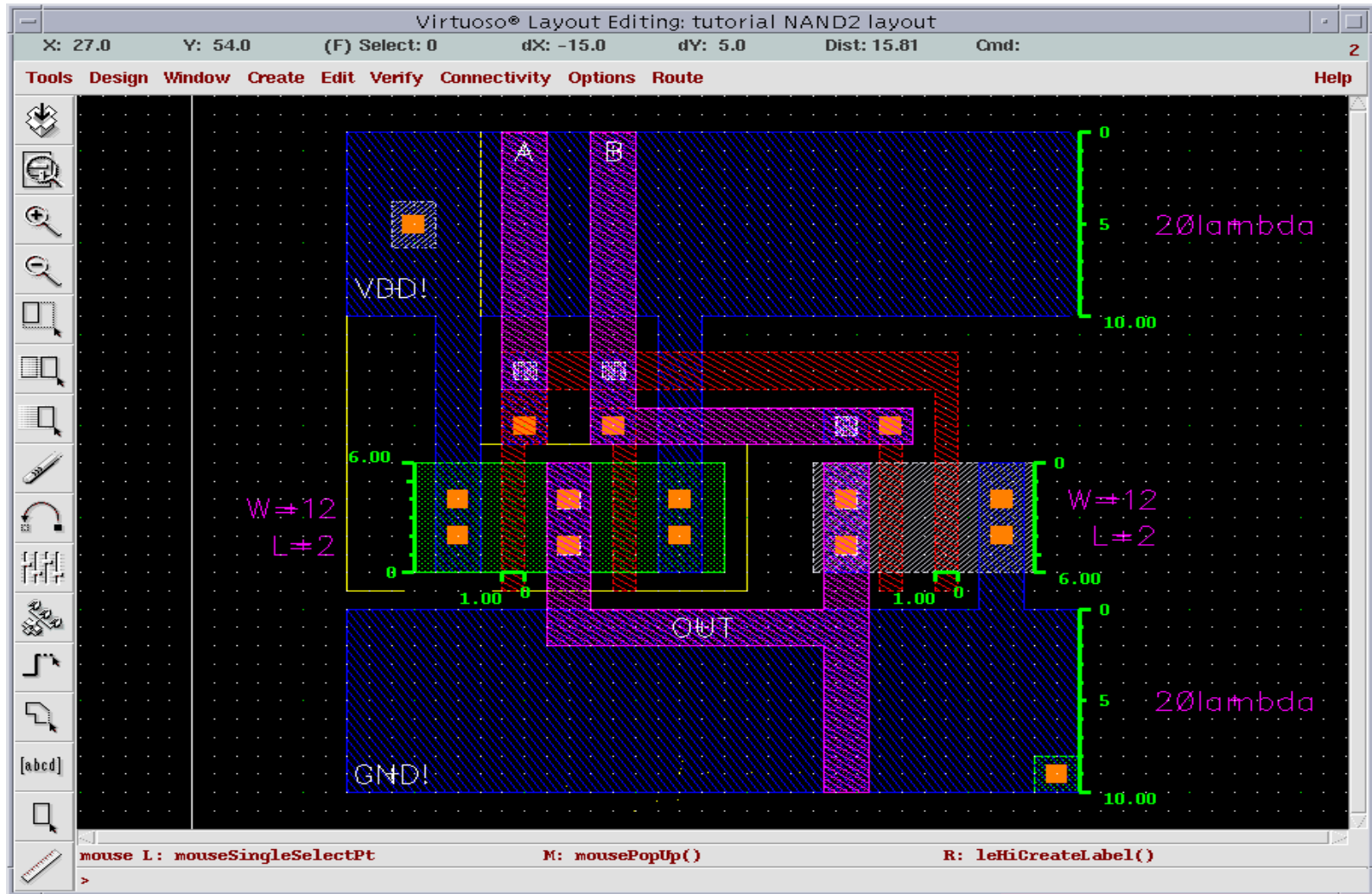


Source Photronics : réticule

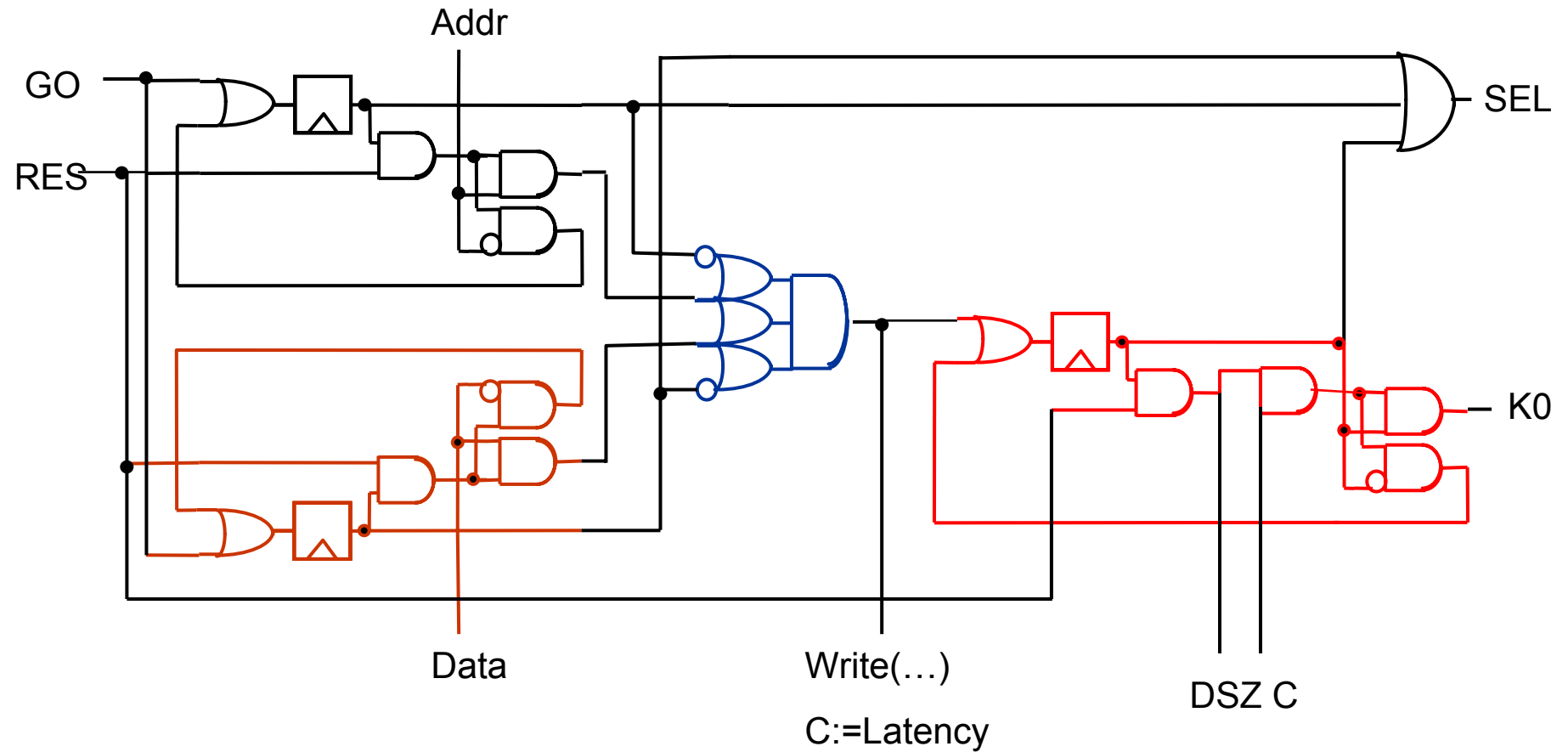


Source ASML

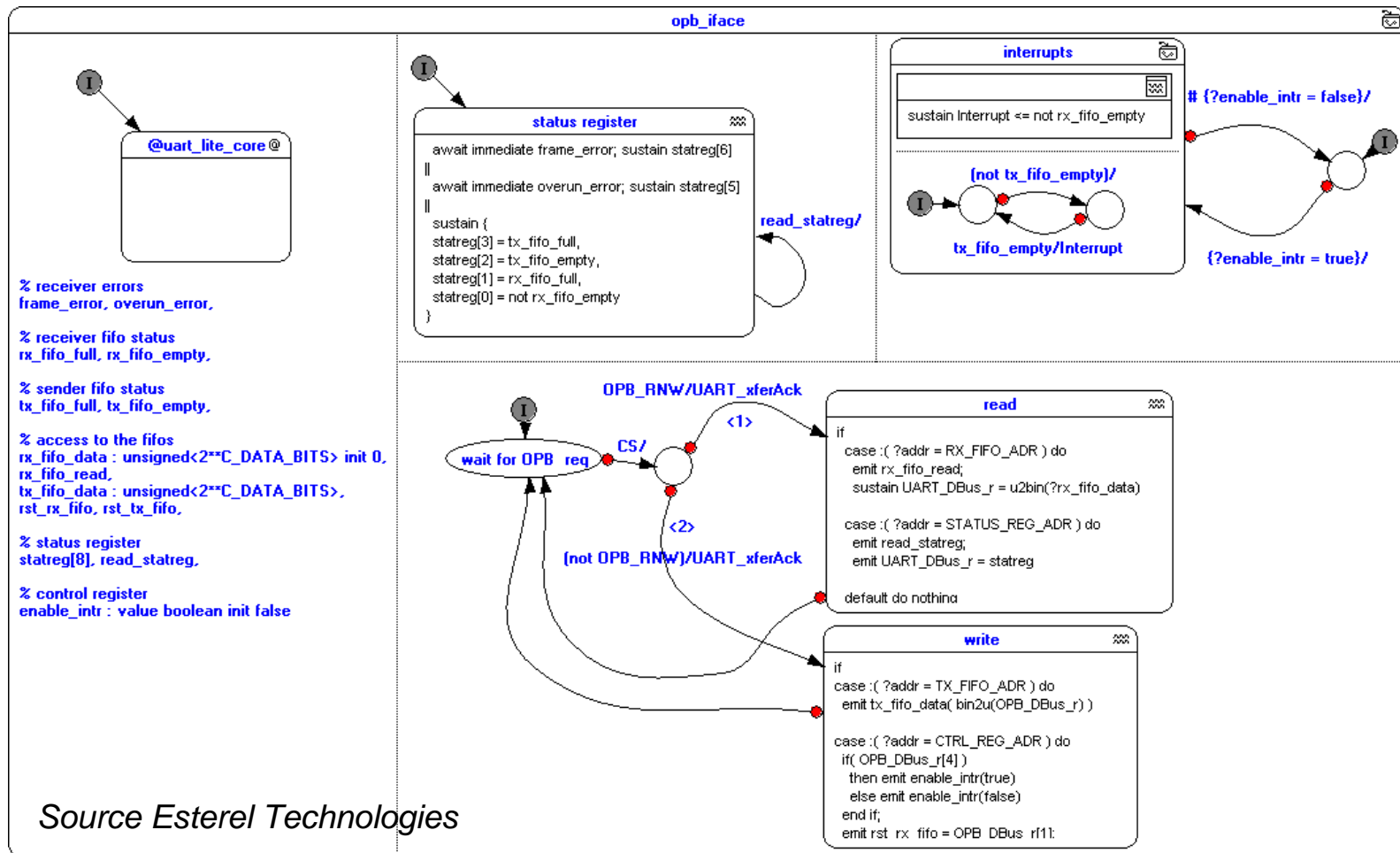
Une porte de base (NAND)



Niveau logique 0/1

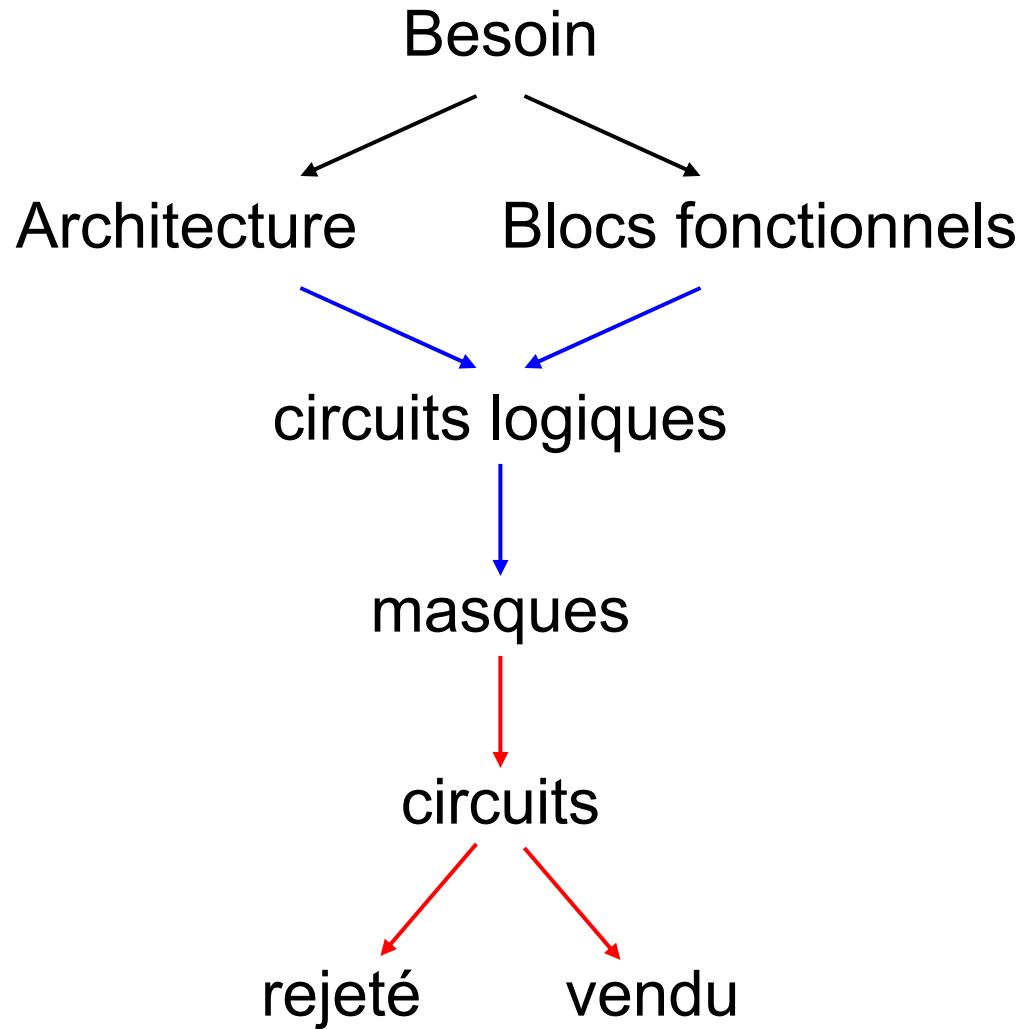


Haut niveau : Esterel v7



Texte + graphique, définition mathématique

La chaîne des niveaux d'abstraction



\$\$



\$\$\$\$



\$\$\$\$\$\$\$\$\$\$\$\$

Objet léger, mais industrie lourde !

L'avenir des circuits

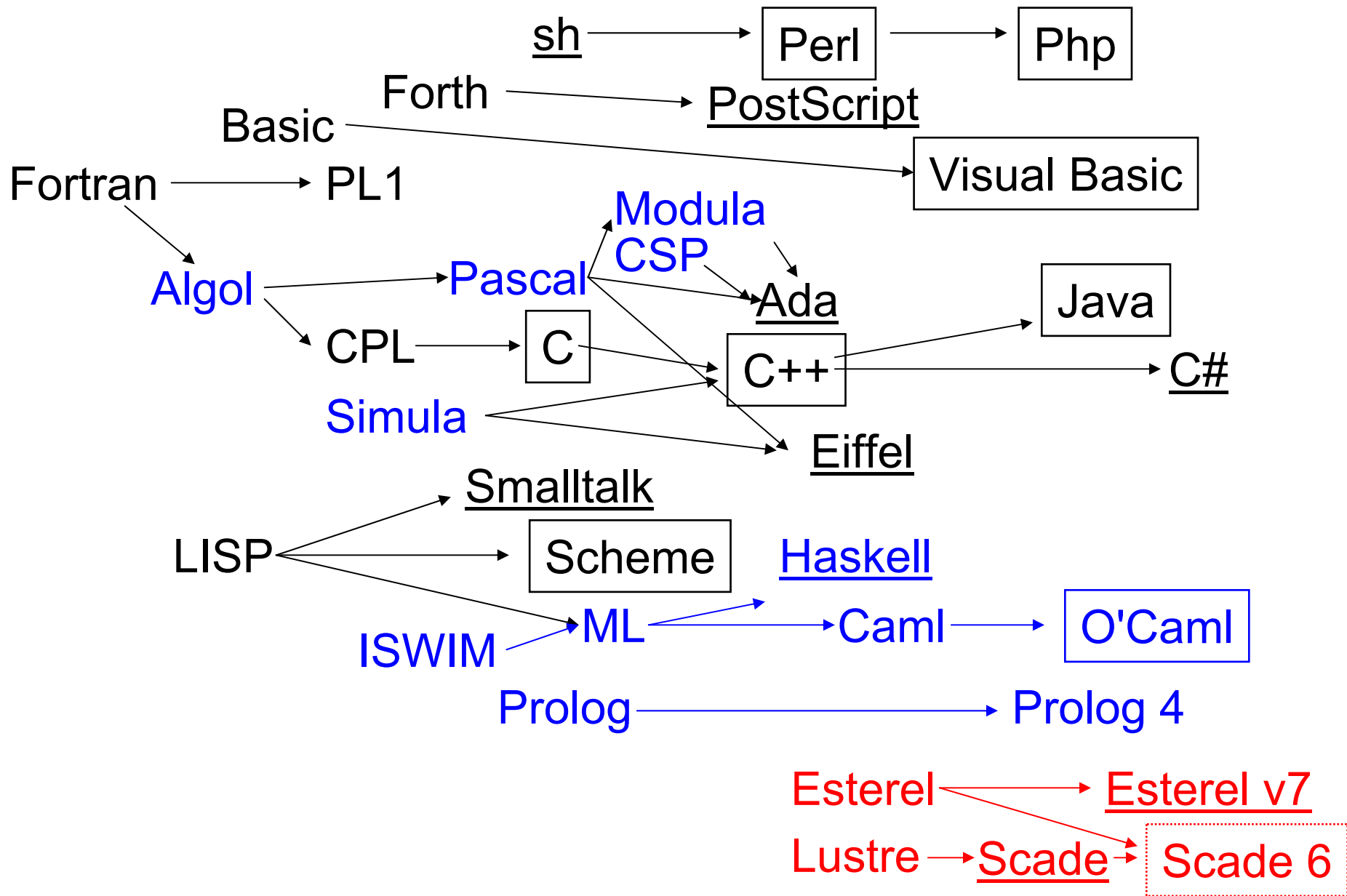
- La réduction de taille peut continuer
 - le problème actuel est la **puissance dissipée**
- Mais c'est l'économie qui pourrait flancher !
 - progression concomitante du **coût des usines** (> 2Md\$)
 - et du coût de **conception des puces** (X*1000 hommes-an)
 - conçues à partir de **composants hétérogènes**
mais faites en une fois et **pas réparables**
 - et rentables seulement en **très grand volume**

Aussi complexe qu'un avion

Le logiciel

Un circuit ne fait que des choses très simples
Il en fait des milliards par seconde, et sans erreur
Le logiciel : spécifier quoi faire, dans tous les détails

- Très long texte dans un langage très technique
- Circuits peu variés, logiciels très variés
- Circuits très rigides, logiciels très souples
- Mais de plus en plus gros (millions de lignes)
- **Et pas de loi de Moore....**



```

const bool SimpleSignal::CheckEmitterOk (const NetInfo& sourceNetInfo) const {
    const Module& module = GetModule();
    if (! IsClock()) {
        return true;
    }
    // signal is a clock, no problem if no emitter so far
    OrGate& sigcurOrGate = sigcur_or_gate_set.GetGate(IncarnIndex());
    if (sigcurOrGate.GetPFirstFaninGateCell() == NULL) {
        return true;
    } else {
        // two emissions, reject
        // fetch current emitter statement
        if (!sourceNetInfo.IsStatementNetInfo()) {
            StnInternalError("bad netinfo for statement gate");
        }
        const StatementNetInfo& statementNetInfo =
            static_cast<const StatementNetInfo&>(sourceNetInfo);
        const Statement& statement = statementNetInfo.GetStatement();
        // fetch previous emitter statement
        const Gate& prevSourceGate =
            sigcurOrGate.GetPFirstFaninGateCell()->GetConnection().GetSourceGate();
        const NetInfo& prevSourceNetInfo = prevSourceGate.GetNetInfo();
        if (!prevSourceNetInfo.IsStatementNetInfo()) {
            StnInternalError("bad netinfo for statement gate");
        }
        const StatementNetInfo& prevStatementNetInfo =
            static_cast<const StatementNetInfo&>(prevSourceNetInfo);
        const Statement& prevStatement = prevStatementNetInfo.GetStatement();
        const char* errMsg = "clock defined twice";
        StnErrorWithTwoPragmaLists(errMess,
                                    prevStatement.GetPragmaList(),
                                    statement.GetPragmaList(),
                                    module);

        GenerationError = true;
        return false;
    }
}

```



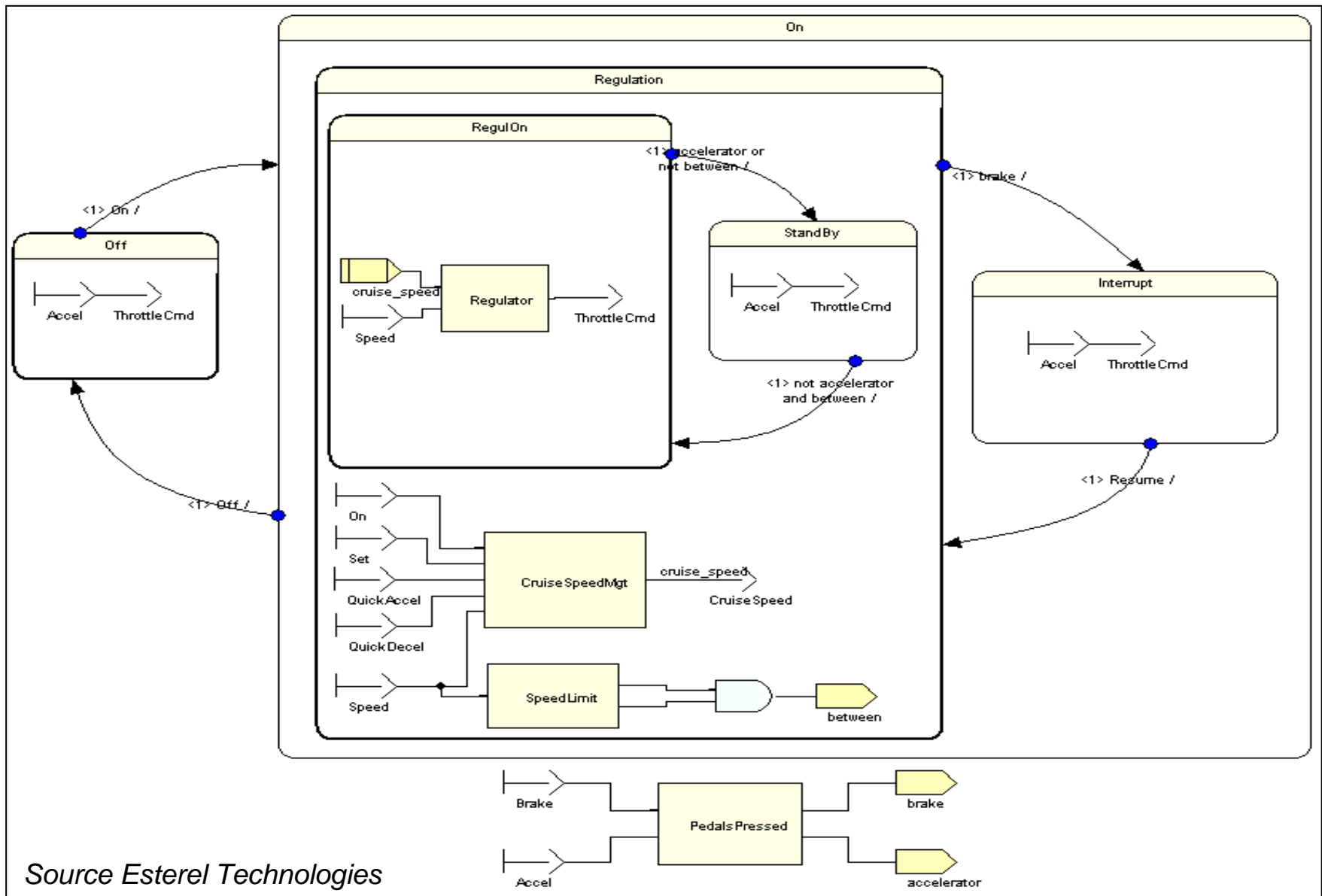
```

const bool SimpleSignal::CheckEmitterOk (const NetInfo& sourceNetInfo) const {
    const Module& module = GetModule();
    if (! IsClock()) {
        return true;
    }
    // signal is a clock, no problem if no emitter so far
    OrGate& sigcurOrGate = sigcur_or_gate_set.GetGate(IncarnIndex());
    if (sigcurOrGate.GetPFirstFaninGateCell() == NULL) {
        return true;
    } else {
        // two emissions, reject
        // fetch current emitter statement
        if (!sourceNetInfo.IsStatementNetInfo()) {
            StnInternalError("bad netinfo for statement gate");
        }
        const StatementNetInfo& statementNetInfo =
            static_cast<const StatementNetInfo&>(sourceNetInfo);
        const Statement& statement = statementNetInfo.GetStatement();
        // fetch previous emitter statement
        const Gate& prevSourceGate =
            sigcurOrGate.GetPFirstFaninGateCell()->GetConnection().GetSourceGate();
        const NetInfo& prevSourceNetInfo = prevSourceGate.GetNetInfo();
        if (!prevSourceNetInfo.IsStatementNetInfo()) {
            StnInternalError("bad netinfo for statement gate");
        }
        const StatementNetInfo& prevStatementNetInfo =
            static_cast<const StatementNetInfo&>(prevSourceNetInfo);
        const Statement& prevStatement = prevStatementNetInfo.GetStatement();
        const char* errMsg = "clock defined twice";
        StnErrorWithTwoPragmaLists(errMess,
                                    prevStatement.GetPragmaList(),
                                    statement.GetPragmaList(),
                                    module);

        GenerationError = true;
        return false;
    }
}

```

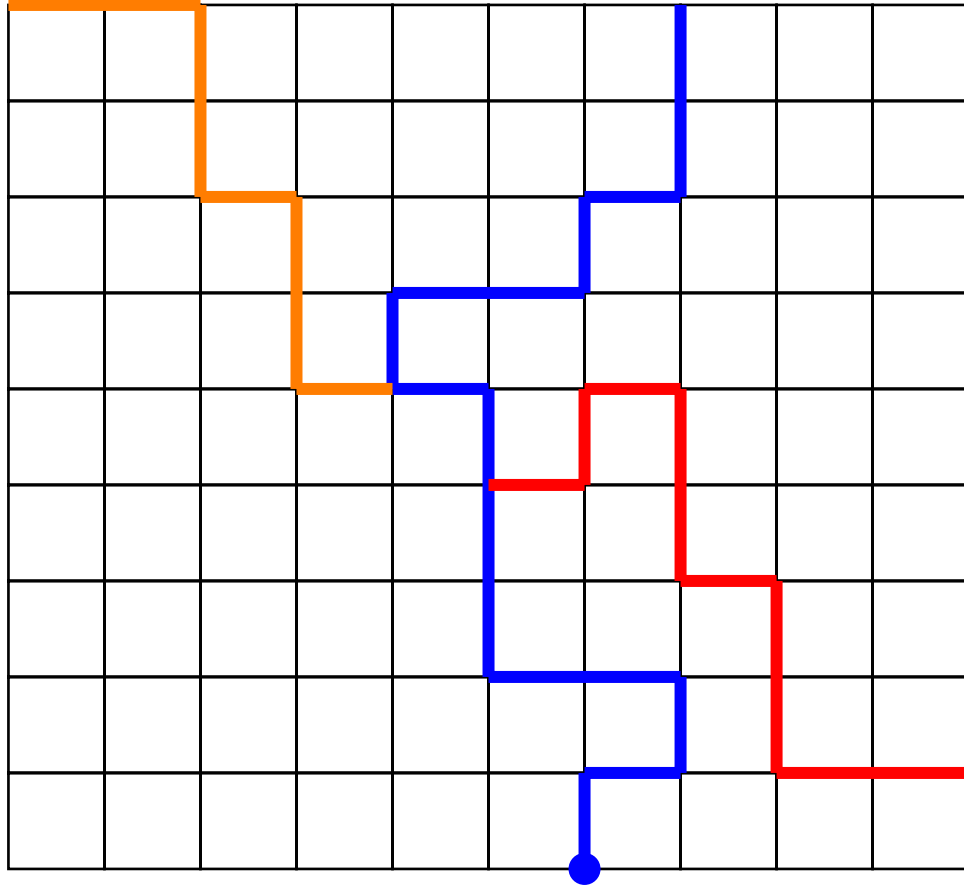
Scade 6 : avions, trains, voitures, etc.



Le bug

- Comportement anormal du système
- Pas une défaillance de la machine, mais **défaillance du concepteur**, i.e., mauvais ordre
- Coût annuel global: **100 milliards de dollars** (DoC)

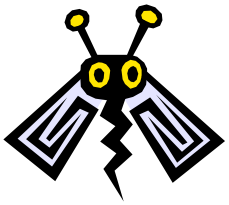
**A petits bugs
grandes conséquences!**



TDGGTDTTGDDTGDGTG
TDGGTDTDGDDTGDGTG
TDGGTDTTGTDTGDGTG

Question pour PDG

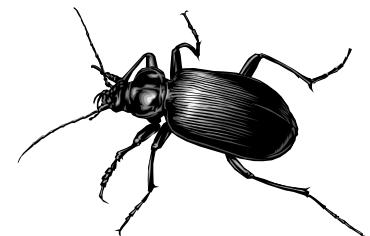
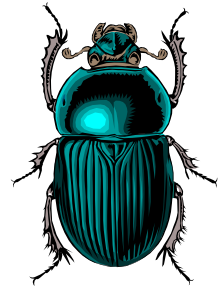
Sauriez vous faire marcher votre entreprise si tous les employés faisaient exactement ce qu'ils sont censés faire, à l'heure dite?



Travaux pratiques



- Plantages des ordinateurs, distributeurs bancaires, systèmes de réservation, sites Web, etc.
- **Micro-bugs** navigateurs Web => **macro-attaques** pirates
- Blocages de téléphones ou d'appareils photo
- Grattages de tête des conducteurs et garagistes
- Crash du téléphone interurbain américain
une ligne mal placée sur 1 million
- Explosion d'Ariane 501, pertes de satellites
Ariane : débordement arithmétique dans un calcul inutile
- Bug subtil dans la division flottante du Pentium
coût : 470 millions de dollars pour Intel
- ...



A la chasse aux bugs

- Utiliser des techniques rigoureuses de **génie logiciel**
 - documentation, revues de code, tests intensifs
 - certification externe (avionique)
- Rendre **visuel** ce qui est invisible
 - environnements de débogage
 - prototypes virtuels animés
- Utiliser des **méthodes formelles**
 - algorithmes validés mathématiquement
 - langages de programmation plus abstraits
 - compilateurs certifiés ou vérifiés formellement

Détecter les bugs *avant* l'exécution

La science informatique

- Théorie de l'information : **coder et transporter efficacement**
- Algorithmique : **faire vite et bien**
- Théorie de la programmation : **écrire vite et juste**
- Reliées : mathématiques discrètes,
automatique & signal, analyse numérique

Bases anciennes: **logique**, **calcul**, et leurs relations

Euclide, Archimède, Brahmagupta, Khuwārizmī, Fibonacci
Hilbert, Church, Gödel, Turing, Von Neumann,...

L'algorithmique

texture éclairage
fusion mise 3D
contours segmentation
corrections optiques
images

trajectoires
déformations
tomographie
surfaces volumes
géométrie

gestion de trafic
diffusion
protocoles
codage
réseaux

éléments finis
matrices prog. linéaire
premiers cryptage
4 opérations
nombres

grammaires
automates
croisement
tri, recherche
classement
mots, textes

emploi du temps
circulation
routage
optimisation

Les critères algorithmiques

- La **machine**: séquentielle, parallèle, distribuée?
- La **correction**
- Le **temps de calcul**
- La **taille mémoire**
- La consommation d'**énergie**

Des milliers de compromis pour des milliers de problèmes
Choisir ? Analyse mathématique sophistiquée !

Structures algorithmiques

objets

bits, entiers, flottants
graphes, matrices
mots, images

structures de données

listes, arbres, tables, etc.
tableaux, chaînages
diagrammes de Voronoi

structures de contrôle

séquence, boucle
récursion
parallélisme synchrone
parallélisme aynchrone

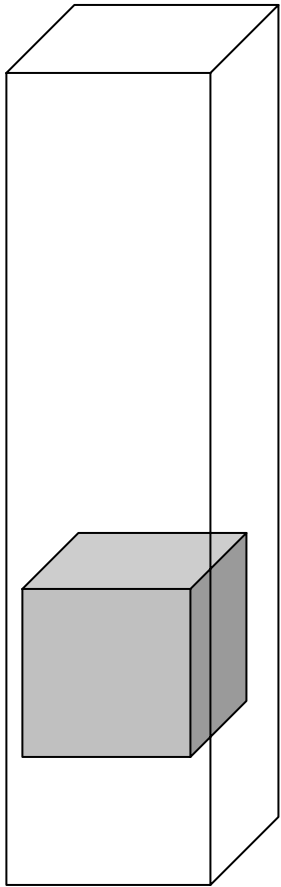
Principes : diviser pour régner, exploiter l'aléa, etc.

La théorie de la programmation

- Comment mieux programmer ?
 - langages plus abstraits, sémantique formelle
 - langage défensifs (assertions, contrats)
- Comment trouver les bugs avant l'exécution?
 - analyse statique des programmes
 - vérification formelle des propriétés
- Exemple: pas de bug arithmétique sur l 'A380
 - spécifications SCADE des commandes de vol
 - interprétation abstraite

Utiliser des programmes pour calculer sur les programmes
... sans se mordre la queue !

Exemple : l'ascenseur



- **Sûreté** : l'ascenseur ne voyage jamais la porte ouverte



- **Vivacité** : l'ascenseur atteint tous les étages et y ouvre ses portes
(plus dur)

L'informatique et les autres sciences

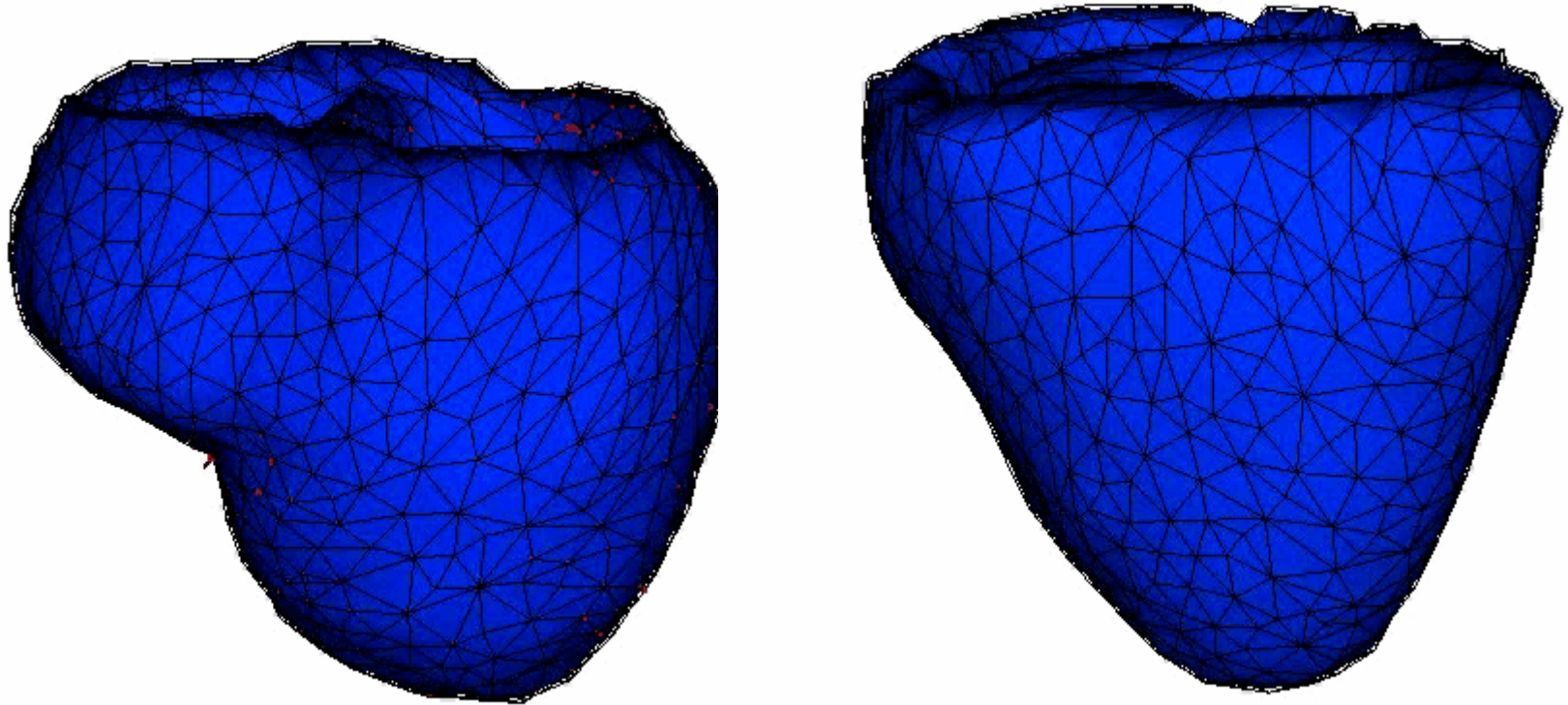
Les mathématiques ont une réussite insolente en sciences

Eugene Wigner

Même chose pour l'informatique, pour les mêmes raisons !

- Publication scientifique : **journaux numériques**
- Calcul formel, mathématiques formelles
- Modélisation / simulation numérique
- Physique : **instruments numériques ou virtuels**
- Médecine : **imagerie numérique**
chirurgie robotisée
prothèses intelligentes
- Biochimie : **géométrie des interactions**
transferts d'information

Simulation d'un infarctus



Source INRIA

Grandes questions

- Saurons nous **contrôler les bugs**, ou **vivre avec** ?
- Et inventer des **principes de calcul** moins brutaux ?
- Pourrons nous être en **sécurité informatique** ?
- Allons nous **contrôler** l'informatique ubiquitaire ou bien tomber dans **Big Brother** ?
- Le sujet est-t-il pris **à son importance réelle** ?
Dans l'industrie, la recherche, l'enseignement ?

- Le sujet est-t-il pris **à son importance réelle** ?

Dans l'industrie, la recherche, l'enseignement ?