Mesure de l'information
Regard d'un ingénieur

Jacques Printz, Professeur au CNAM, Chaire de génie logiciel

Réunion AFSCET 19 Novembre 2007

1ère partie : Pourquoi « mesurer » l'information ?

L'information pour un ingénieur chargé de réaliser un système informatisé (1/2)

Dans le monde réel :

- L'information est un bruit qui porte du sens
  - Syntaxe de la représentation – Structure du signal
- L'information permet d'agir avec plus d'efficacité
  - Sémantique – Concept et mode opératoire – Événements associés
- L'information est élaborée par et pour des êtres humains
  - Pragmatique et contingence des actes humains : les erreurs humaines et la tolérance aux erreurs
    - Si l'on n'a pas été correctement éduqué, l'écriture n 'est que du bruit, mais il faut **distinguer lire et comprendre** ; on peut lire sans comprendre !

L'information pour un ingénieur chargé de réaliser un système informatisé (2/2)

Dans la machinerie informatique :

L'information est discrétisée :

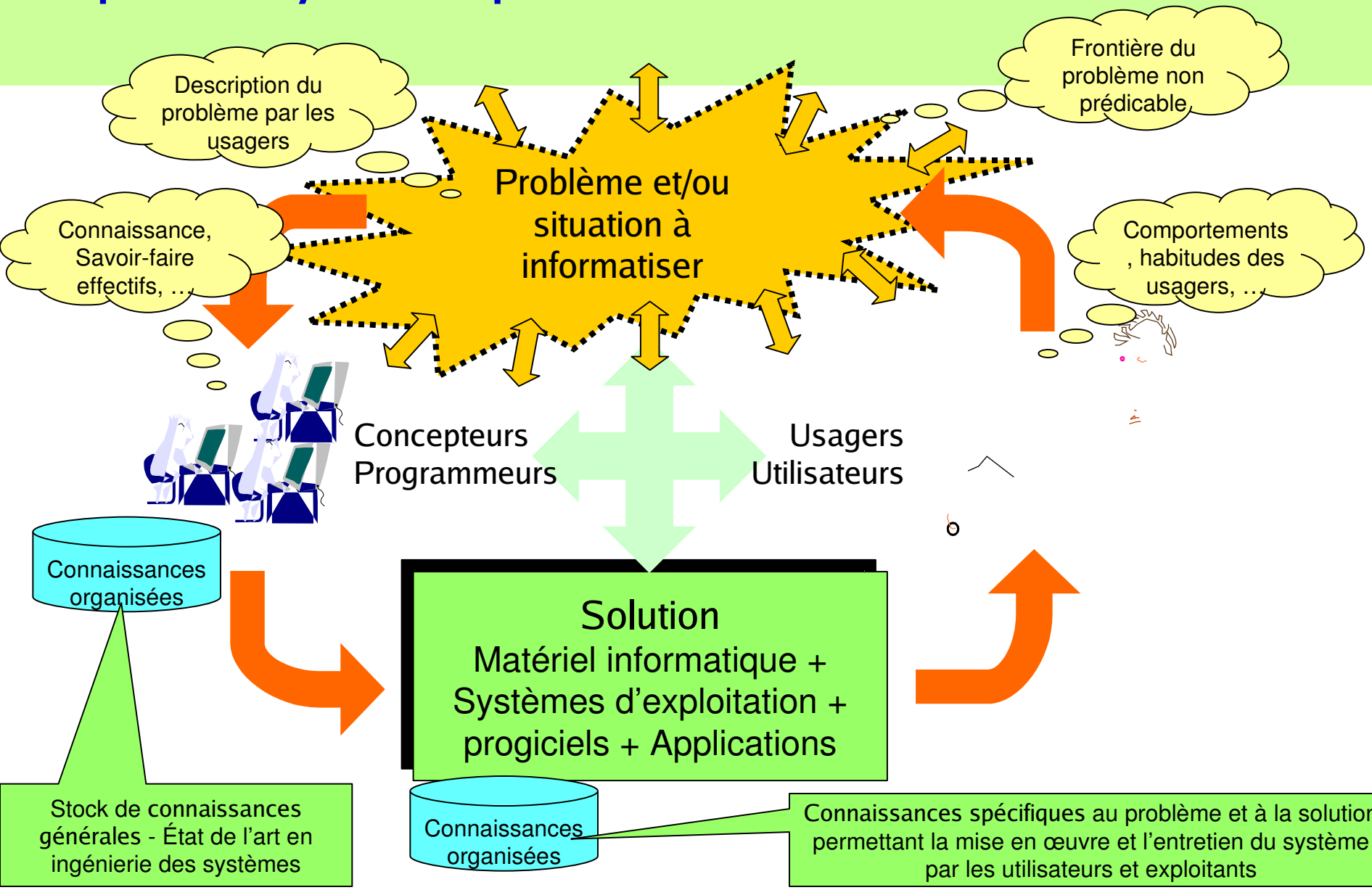
- Données (états) – Procédures (transformations effectuées par des programmes ; contraintes) – Événements (temporalité[s] et relations avec l'environnement)

Pour s'exécuter le programme a besoin de ressources – Règles de partage

- Calcul – Entrées/Sorties disques et réseaux – Services – …

La machine et son système d'exploitation apporte son propre bruit de fond, en plus des erreurs commises par les programmeurs

Aspects systémiques de la création d'information



Les questions posées à l'ingénieur : quelle ingénierie de l'information ?

Est-ce faisable ? Où sont les limites ?

Combien ça va coûter ?

Quand aurai-je une solution avec une garantie de qualité ?

Ai-je l'équipe compétente pour réaliser le projet ? Sinon, nature de la courbe d'expérience et dynamique de maturité ?

Comment faire changer les habitudes des usagers (en milliers pour les « grands » systèmes) et des utilisateurs ? Effort à fournir pour la conduite de ce changement ?

Toutes ces questions impliquent une certaine forme de mesure

- Mesure de la complexité du système à réaliser, en fonction des besoins exprimés
- Relation entre la « taille » du système, le coût de réalisation, le nombre de personnes de telles ou telles qualifications à intégrer dans l'équipe de réalisation, la durée probable de la réalisation, etc.
- Mesure de la difficulté à changer les comportements ; interactions nécessaires, acquisition de compétence

Chacune de ces mesures intègre à différents degrés les aspects syntaxe, sémantique, pragmatique de l'information

Étalons de mesure pour l'information : un problème ouvert

Données stockées dans la machine

Taille des fichiers et des bases de données – Mesure textuelle en nombre de caractères ; mais ce qui est important, c'est la classification associée (entités, attributs, relations) – Adressage, navigation

Programmes codifiant les procédures informatisées

L'étalon est une machine abstraite (cf. Turing – von Neumann)

Événements gérés, i.e. ceux pour lesquels la machine à une réponse appropriée

Distinction entre le nombre de classes d'événements et le nombre d'occurrences des événements individuels « hic et nunc » : un flux

Taille d'un problème

Nombre d'opérations nécessaires à l'obtention d'un résultat exploitable par des êtres humains et/ou d'autres machines

Perception de la mesure induite par le support de l'information

Information matérialisé

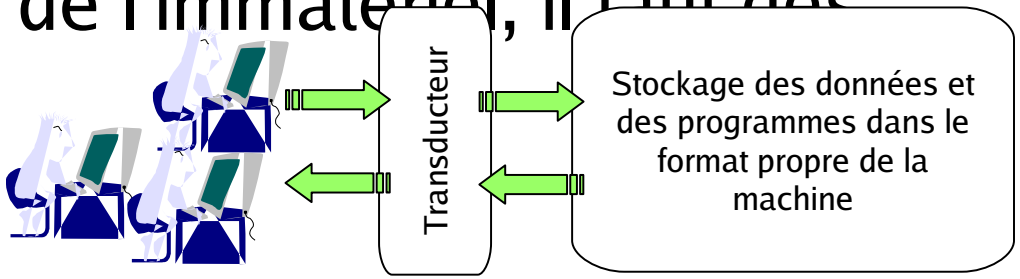
Nos organes des sens peuvent apprécier ou « mesurer » la quantité d'information via la taille du support physique – Illusions trompeuses liées à la sensation

Information dématérialisée

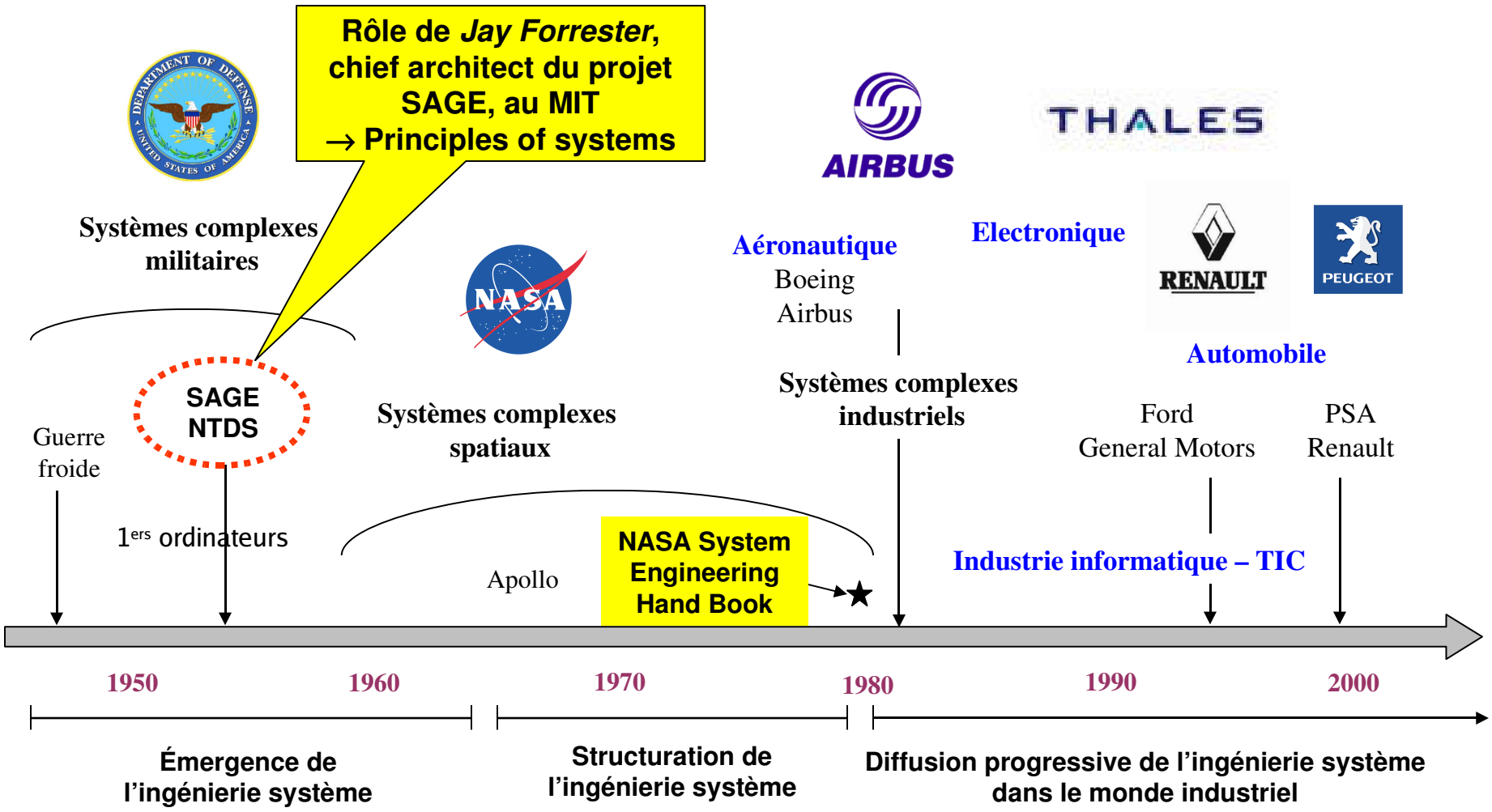
Une abstraction est immatérielle par nature – Un programme est immatériel

Il n'y a aucune relation entre la taille du support physique (papier, livres) et la quantité d'information nécessaire à son stockage (en bits utiles)

Pour capter/agir sur de l'immatériel, il faut des transducteurs



Un peu d'histoire systémique …



- SAGE = Semi-Automatic Ground Environment = 1er système de défense anti-aérienne américain
- NTDS = Navy Tactical Data System = = 1er système de défense naval américain

Trois pionniers de la mesure de l'information

Claude SHANNON

- Quantité d'information via un opérateur statistique
- Théorie du codage – Codes économiques et codes correcteurs d'erreurs

John von NEUMANN

- Invente l'ordinateur et son modèle abstrait : l'architecture de von Neumann
  - C'est une machine logique étalon, plus pertinente pour l'ingénieur que la machine de Turing
- Dimensionne la taille des calculs nécessaires à la résolution d'un problème en nombre d'instructions
- Théorie des automates et fiabilité de la machine à partir de composants non fiables
  - Comment « calculer » le volume de redondance utile
- Théorie des jeux (sous l'angle des décisions et des stratégies de décisions)
  - Cf. également Kahnemen & Tversky, Choices, values and frames, CUP

Grégory CHAITIN – A.KOLMOGOROV

- Définit la quantité d'information d'une procédure informatisée (un algorithme) comme la taille du programme le plus compact sur une machine logique abstraite qui sert d'étalon de mesure

La « taille » d'un « grand » programme

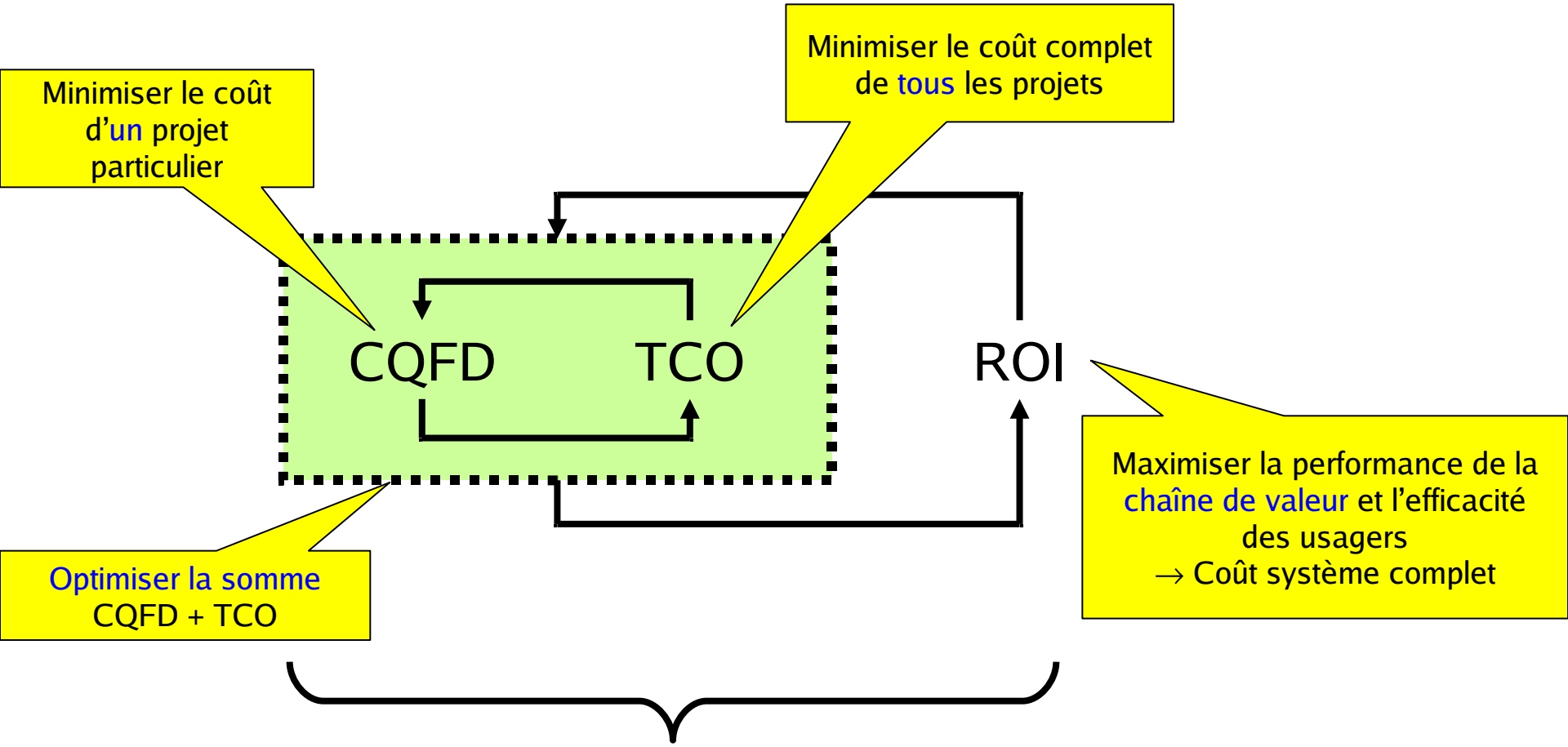
L'état de l'art consiste à compter les instructions écrites par les programmeurs

- Tout le monde a une intuition de la « taille d'un texte »

Pour un système d'exploitation ≈ 5 millions d'instructions source en langage de type C/C++

- Soit ≈ 500 livres de 200 pages pour le texte brut
- Plus du double si l'on compte les différentes annexes nécessaires à la gestion et aux évolutions de ce programme
- Investissement cumulé sur 15 ans ≈ 5.000 hommes×an
- Construction progressive par tranche de 500 à 800 mille instructions source
- Taux d'erreurs ≈ 1 à 2 erreurs par millier d'instructions

Les régulations fondamentales



Optimiser l'ensemble, et non pas chacune des parties

2ème Partie :
Complexités textuelles, architecture et modèle de coût

Essai de définition d'une « mesure naturelle » du contenu informationnel d'un système du point de vue de l'ingénieur

Phénoménologie de la complexité

Deux métaphores pour mieux comprendre :

- L'horloge astronomique

  Le dispositif d'échappement (une **dynamique** qui donne un étalon de temps) + énergie potentielle (Poids et/ou ressorts) permet, par **transduction**, la mesure du temps et la prédiction du mouvement des astres

  Nombreuses pièces en **mouvement**, i.e. une **dynamique**, mais il y a une limite due au frottement

  Le **plan de montage-démontage** apparaît comme un élément clé de la complexité de l'horloge (cf. le « blind watchmaker » de R.Dawkins, ou la notion de chaîne de montage dans l'industrie automobile)

- Le puzzle

  L'intérêt est dans la **découpe** + recherche de sous-ensembles associés à des **formes signifiantes** (cf. local vs global)

  La complexité est de nature **topologique** et dans les **relations** entre les pièces (cf. les différents couplages possibles)

- … mais aussi, l'ordinateur lui-même est une excellente métaphore, ou la cathédrale …

Notion de CCI

La notion intuitive [informelle] de complexité dans les projets agrège différents aspects [i.e. une intrication] :

- Complexité intrinsèque
  - Du système (**sémantique**, sémiologie, … via une machine abstraite)
  - De l'**organisation** du projet de réalisation – **Double découpe** : produit et projet
- Complexité Complication
  - **Notations** utilisées pour programmer (syntaxe, symboles graphiques, …)
  - **Technologies** sous-jacentes, machines réelles
  - Procédures de **décisions/choix** rationnels (ou non !) de l'organisation
- Complexité Incertitude – Risque résultant de la découpe
  - Performance et fiabilité des **équipements**
  - Performance et fiabilité des **acteurs** (maturité des acteurs et de l'organisation ; connaissances requises)
  - **Risques** et aléas

Peut-on les séparer et isoler un noyau dur de complexité intrinsèque ?

Trois textes indispensables, présents dans tous les projets

Le référentiel projet

C'est un méta-texte, explicite ou implicite, qui fixe les règles que les programmeurs devront impérativement respecter pour éviter l'anarchie et le chaos des initiatives individuelles

En particulier les **interfaces**, qui font l'objet de négociations (**contrats**) entre les acteurs au sein d'une équipe et entre les équipes

La programmation

Description statique de ce que l'ordinateur est censé faire (i.e. ce qui va réellement s'exécuter dans l'ordinateur)

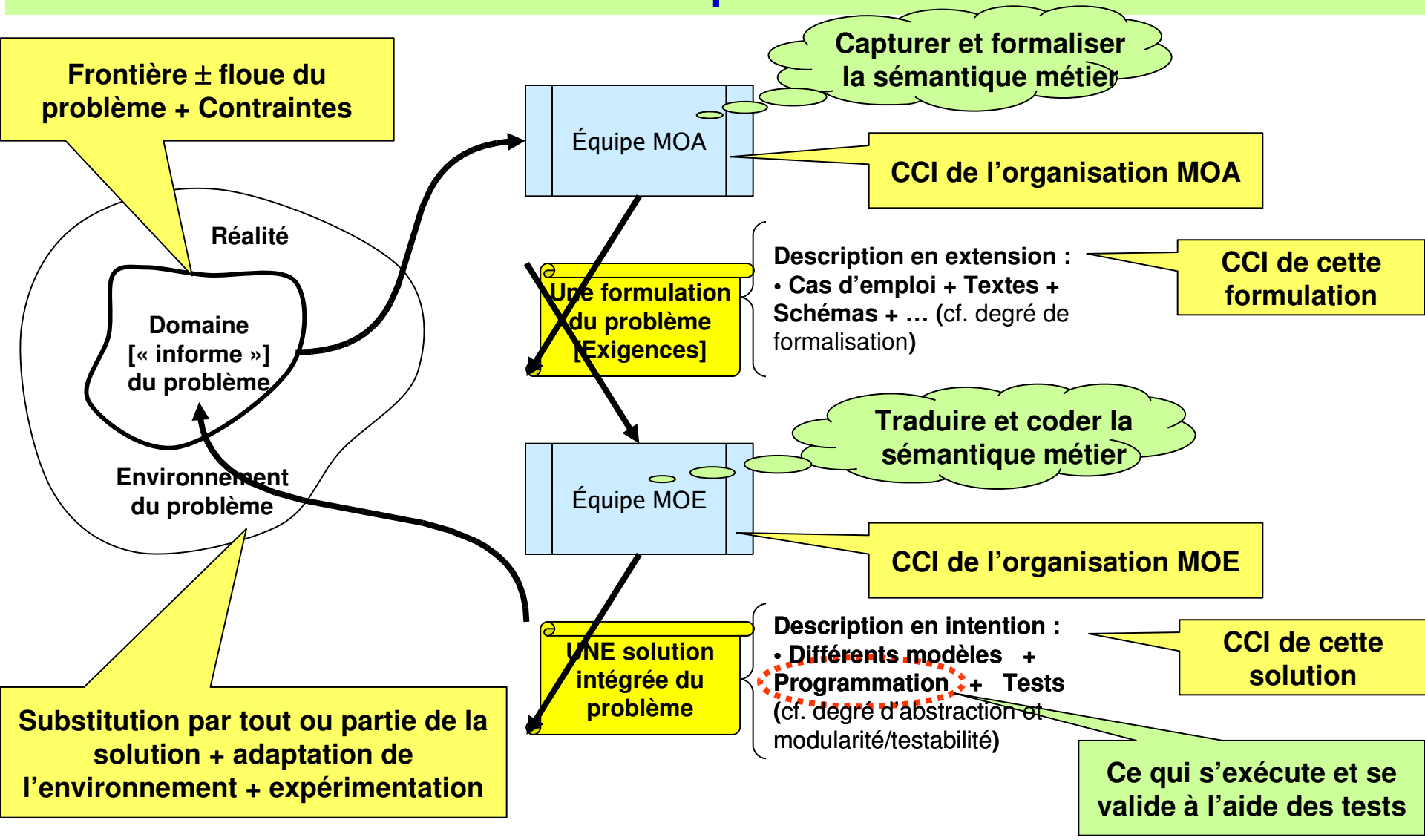
Taille du programme en nombre d'instructions + Nomenclatures X-ref

Les tests

Description dynamique de toutes et rien que les combinaisons autorisées qui permettent de vérifier et de valider ce qui a été programmé, compte tenu des exigences système (FURPSE + PESTEL)

Historiques, traces et lignes de vie du système

Contenu informationnel du système – Sémantique brute



Le paradoxe de la sémantique

Il est impossible d'exprimer la sémantique dans l'abstrait, sans faire appel à une notation particulière (signes, langages, …)

La connaissance de la notation ne garantit pas que l'on va correctement capter le sens du problème à résoudre

Comprendre un problème revient à définir une machine de référence associée à ce problème et à programmer le problème dans le langage de cette machine

Peut-on raisonnablement dire que la taille de ce programme, ou de ses tests, ou des deux, dénote la complexité du problème ?

L'information textuelle dans l'ordinateur

Dans un ordinateur, tout est texte (information linéaire, dans une mémoire linéaire)

Mais le cerveau humain est à l'aise en 2D et en 3D

Tous les textes exécutables sont fabriqués, directement ou indirectement, par des programmeurs (au sens large)

Avec une variété de langages ± grande (c'est un codage de l'information)

Une mesure textuelle est intuitive, du point de vue cognitif, pour les acteurs humains, ce qui est le plus simple à faire partager

Mais qui agrège des éléments hétérogènes

Les mesures textuelles les plus connues

La taille du code, en milliers de LS, dans les modèles de la famille COCOMO

Comptage des instructions écrites par le programmeur

Corrélation statistique forte entre les instructions (loi des grands nombres) et l'effort pour les programmer (équation d'effort)

Les mesures fonctionnelles en Points de Fonctions

Fondées sur les données (entités-attributs) manipulées par l'application (CRUD) – Très utilisées dans l'ingénierie des SI (BD)

Intérêt : Indépendantes des langages de programmation

Les mesures « information science » de M.Halstead

Inspirées de la théorie de l'information (C.Shannon, quantité d'information)

Mal fondées, difficiles à interprétées, sont restées sans suite

Peut-on, et comment, « mesurer » la complexité

Constat : Il n'y a pas de mesure absolue de la complexité

C'est une notion relative qui dépend du niveau d'organisation que l'on observe, et de l'acteur qui observe

Le meilleur exemple est l'ordinateur lui-même, depuis le transistor jusqu'à …

Quel sens donner à l'expression : « plus complexe » ?

Plus grande taille du code, en nombre d'instructions

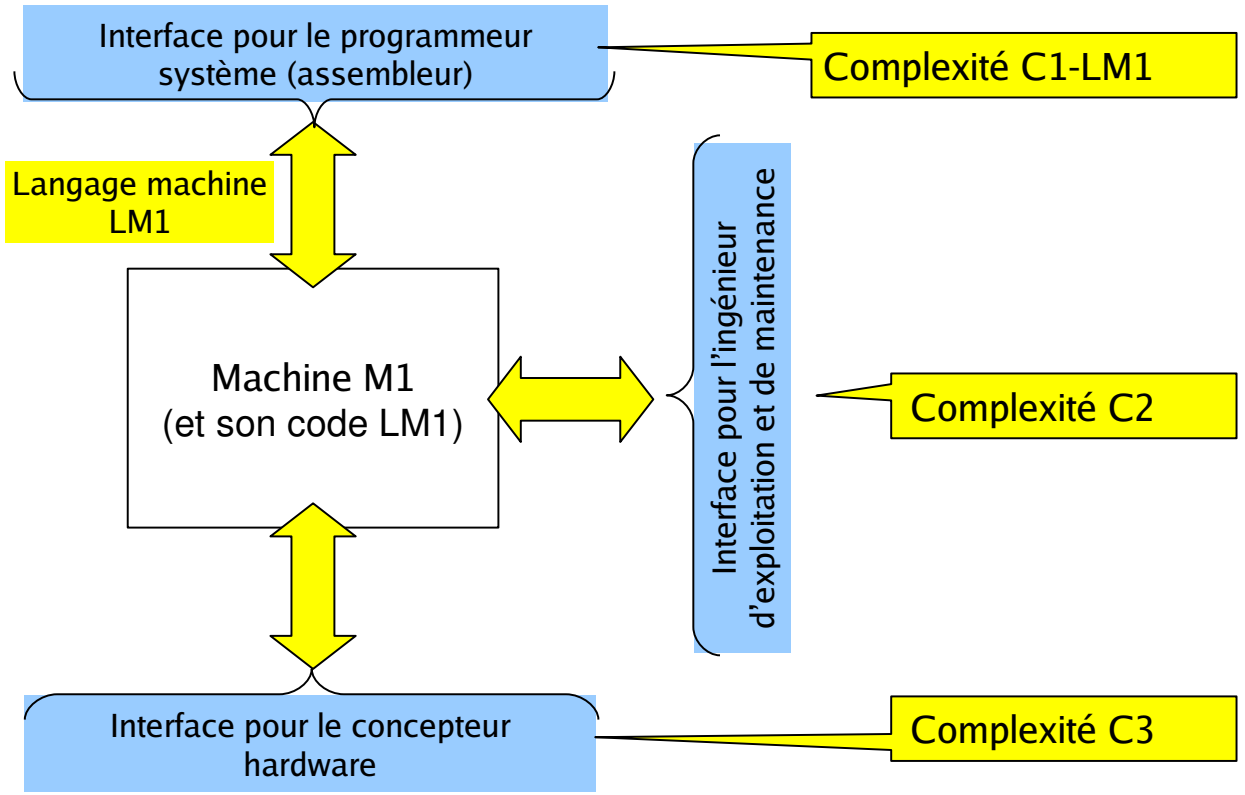
Plus grande taille des tests, mesurée via un niveau de couverture

Plus coûteux, plus long à réaliser, à intégrer, à modifier, …

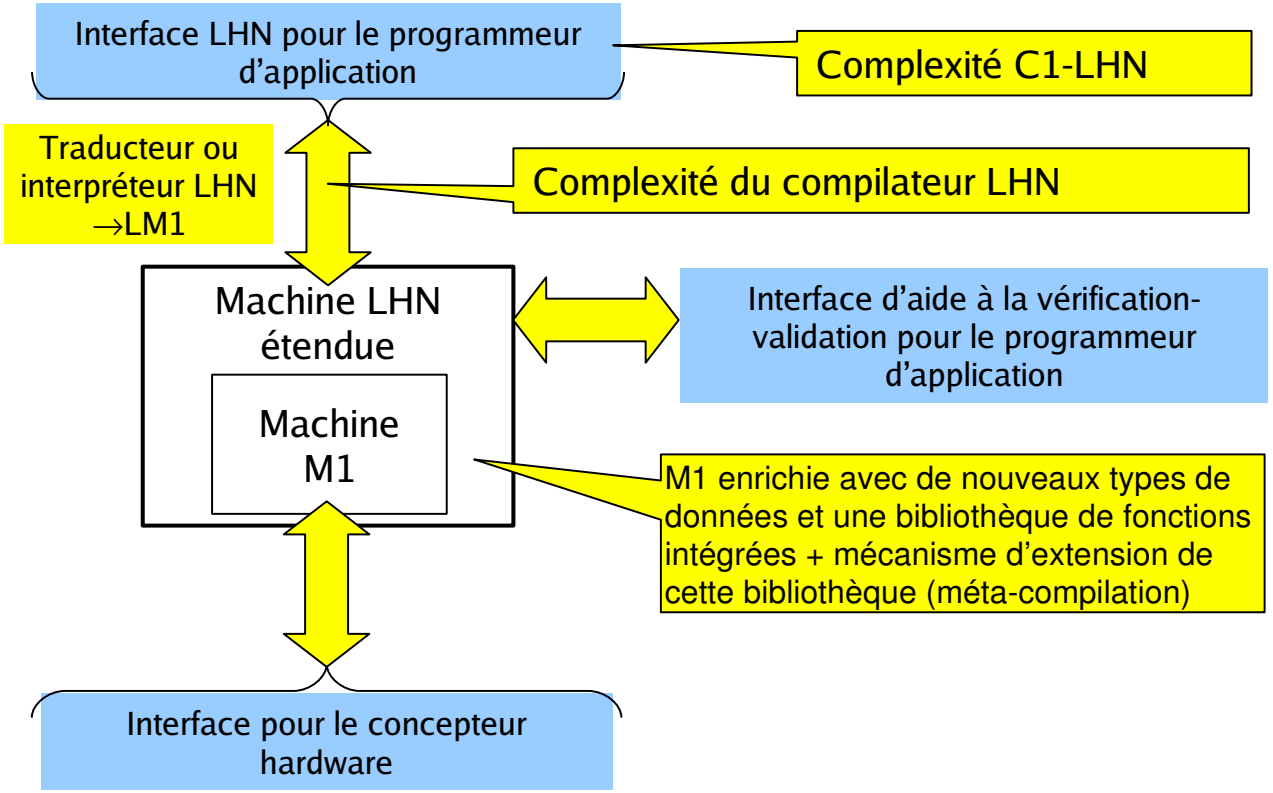
Densité des différentes matrices de couplages utilisées en intégration

Plus grand graphe, plus grand nombre cyclomatique, …

Les complexités vues de l'architecture de la machine

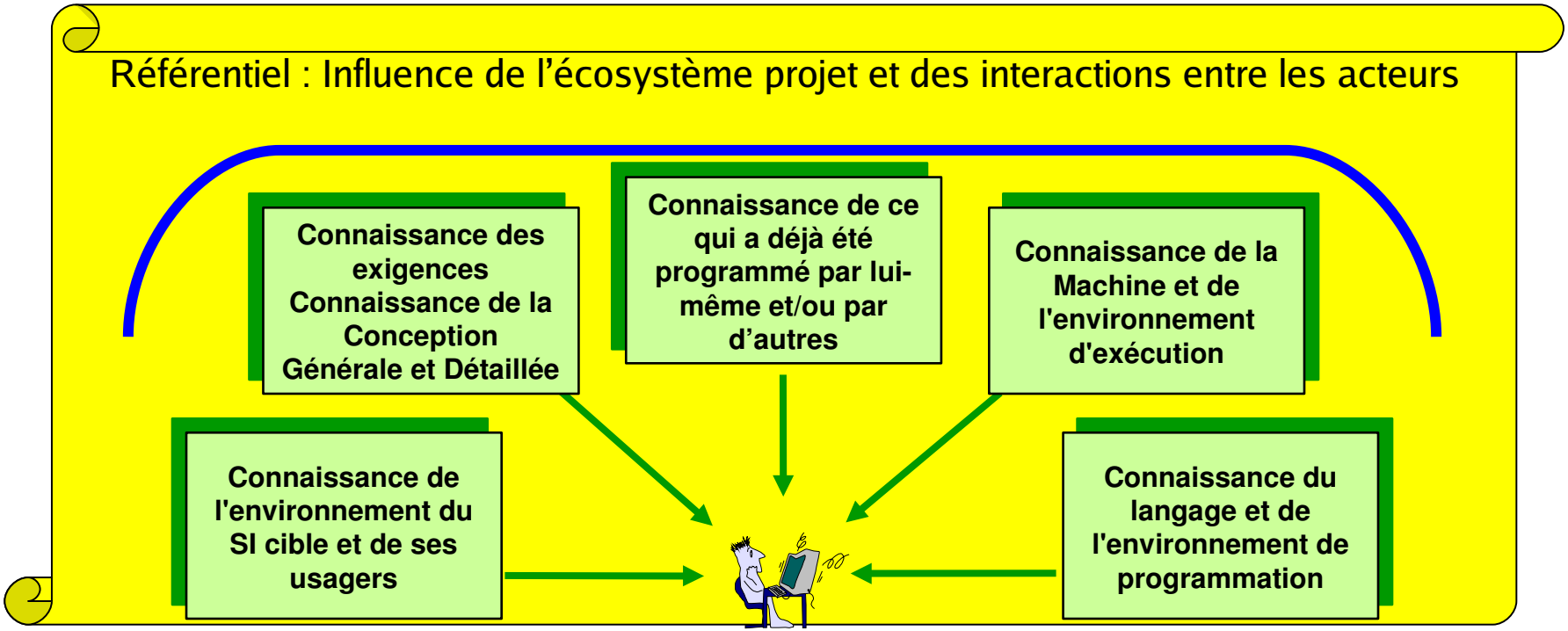


Machine étendue LHN

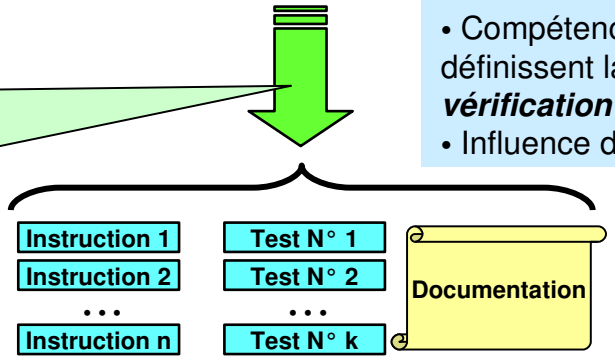


Écosystème du programmeur – Acte de programmation (1/2)

Référentiel : Influence de l'écosystème projet et des interactions entre les acteurs



Programme = Code + Tests + Documentation
Test Driven Development et testabilité des systèmes

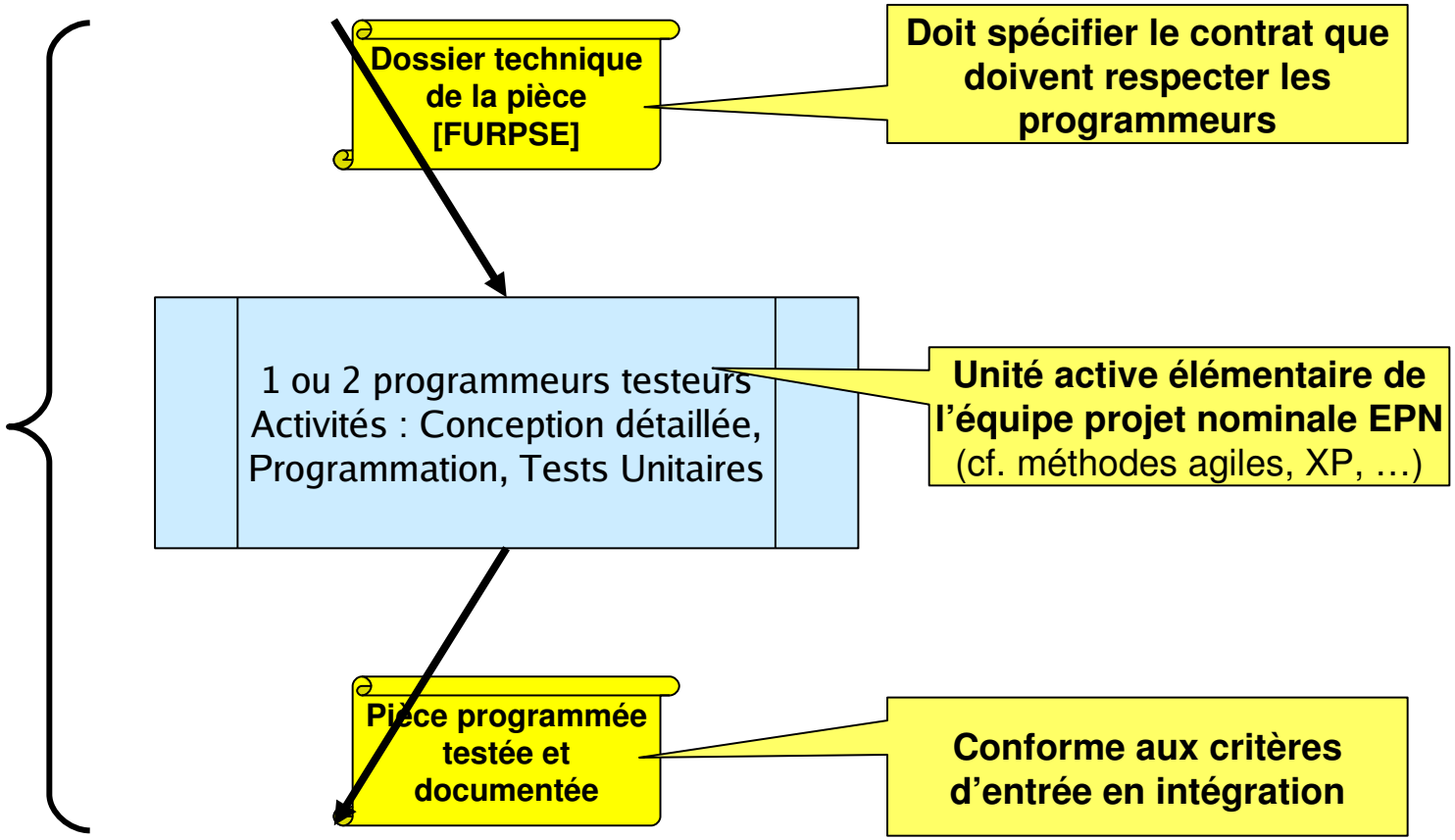


- Compétence et capacité du programmeur définissent la ***vitesse de fabrication et de vérification*** des instructions produites
- Influence de la motivation des acteurs

Une « pièce »

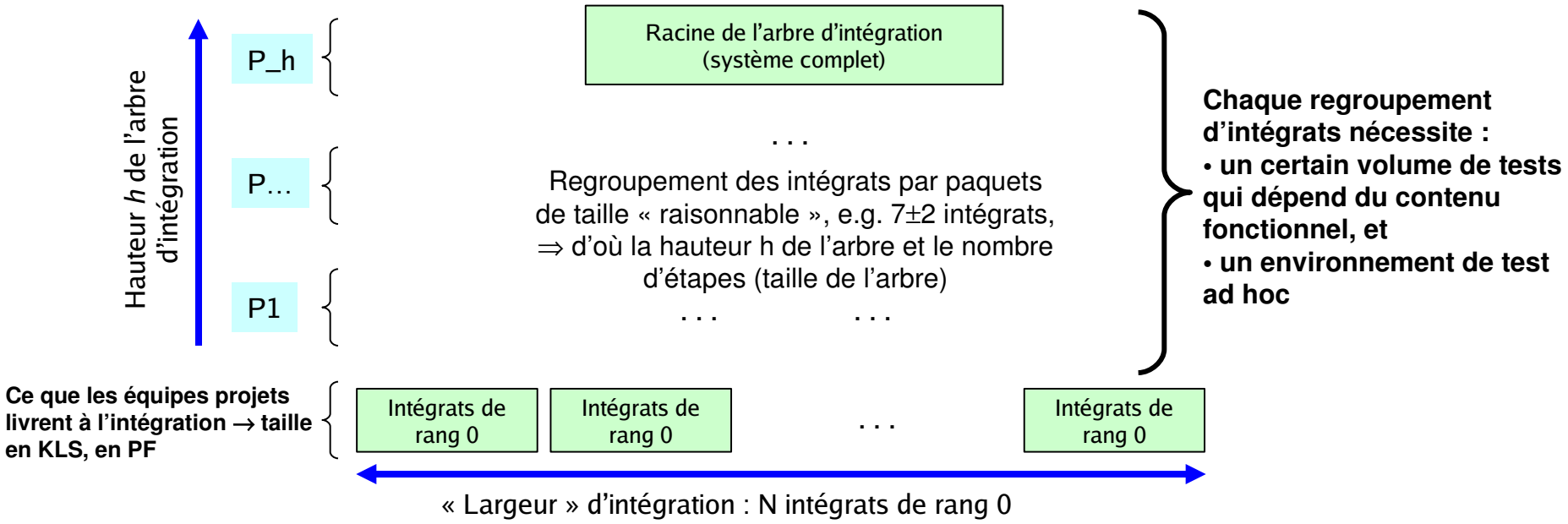
Écosystème du programmeur – Acte de programmation (2/2)

« Pas » ou « grain » de construction élémentaire + CQFD de la pièce



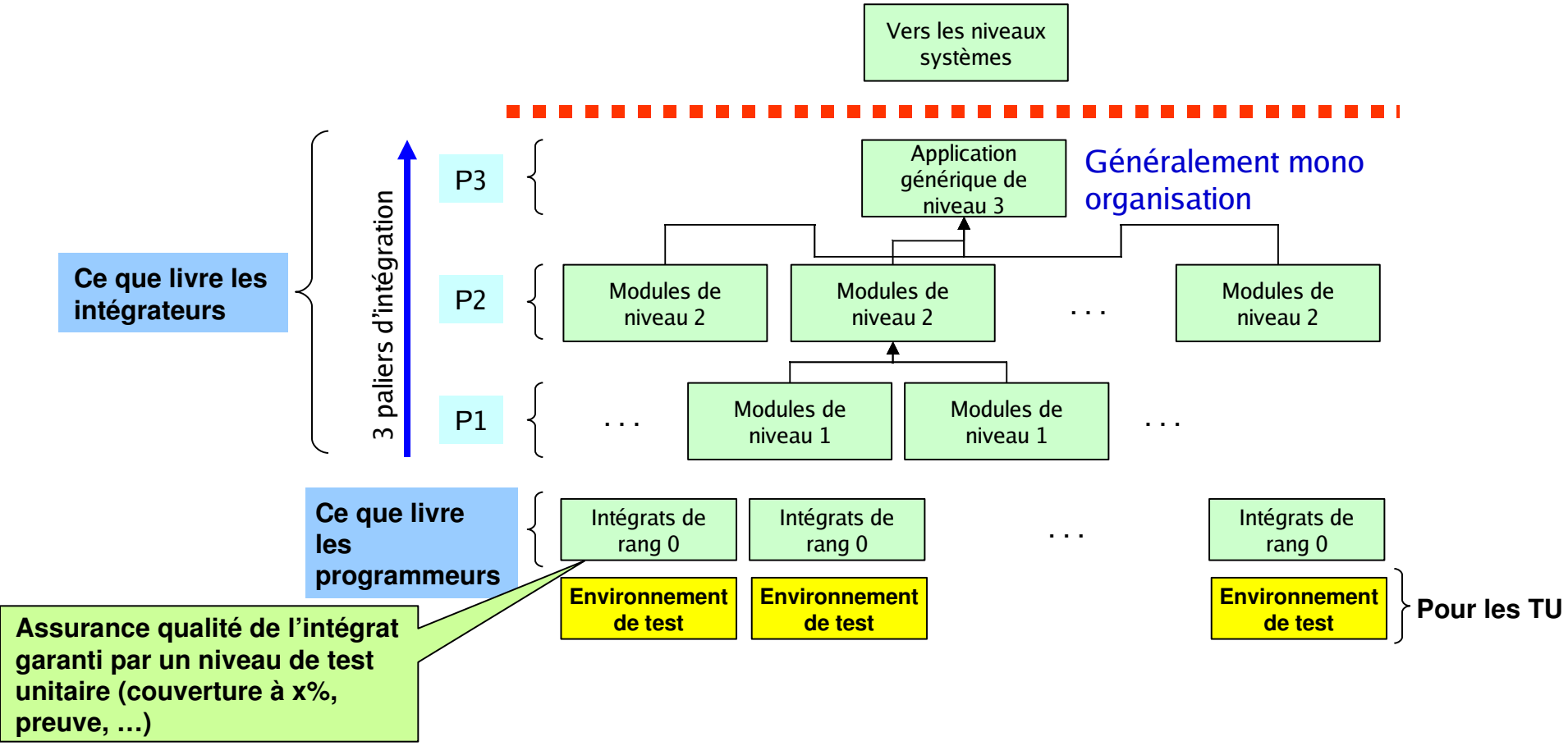
Taille et effort indicatifs : 1-2 KLS, 2-3 HM (contrainte psycho-cognitive)

Largeur et hauteur de l'arbre d'intégration



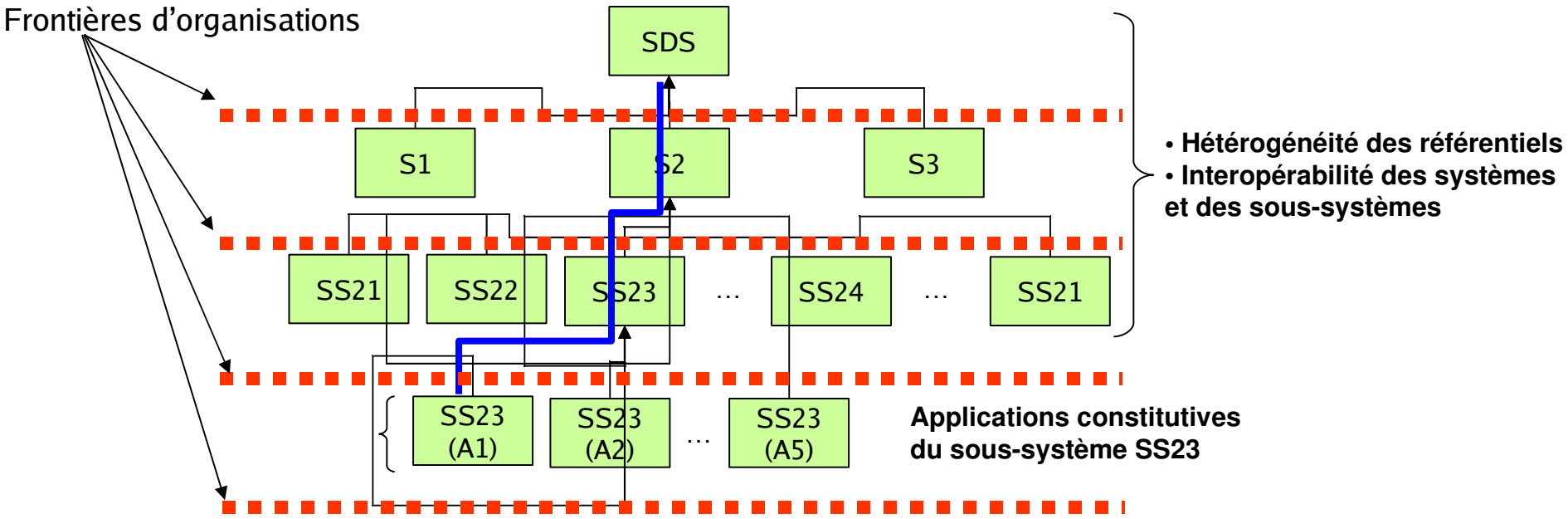
NB : la hauteur h croît comme le Log à base 7±2 du nombre d'intégrats

Arbres et paliers d'intégration – niveau application



Chaque intégrat nécessite un environnement de test ad hoc
Importance de la standardisation pour faciliter les opérations de rejeu des tests (non régression)
Pb potentiels avec les COTS et le logiciel libre (en particulier incertitudes des ENF)

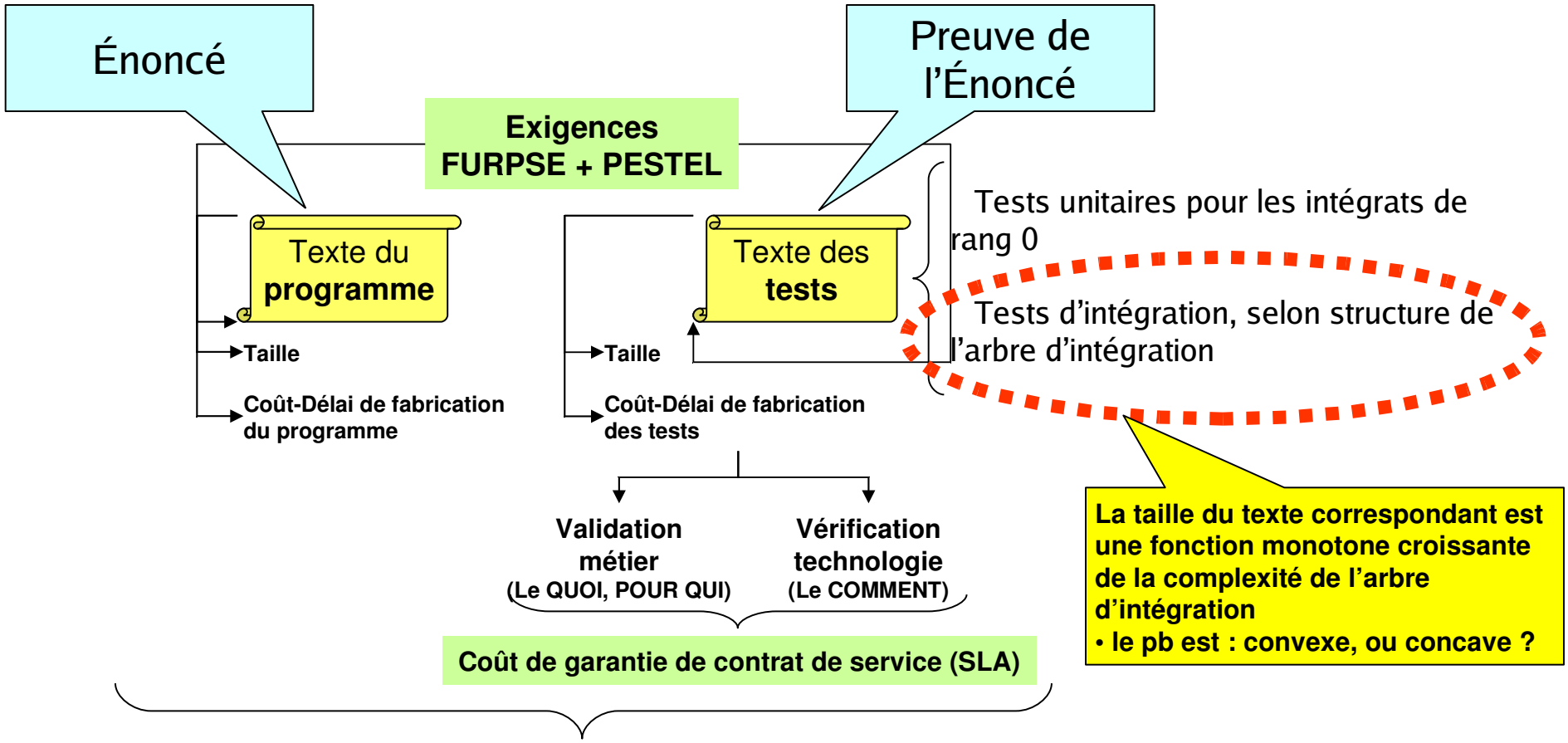
Arbres et paliers d'intégration – niveau système et système de systèmes



Les interfaces qui matérialisent les frontières d'organisations sont les plus coûteuses à définir (cf. les adaptateurs pour l'utilisation des EAI + interopérabilité)

Les frontières d'organisations peuvent/doivent être matérialisées par des hiérarchies de bus logiciels

Mesure combinée texte du programme + Texte des tests



Équation de l'effort COCOMO

Peut-on justifier la forme de l'équation ?

$$Effort = \underbrace{k} \times \left(KLS \right) ^{\overbrace{1+\alpha}}$$

Terme linéaire

Terme NON linéaire

Facteurs d'influences

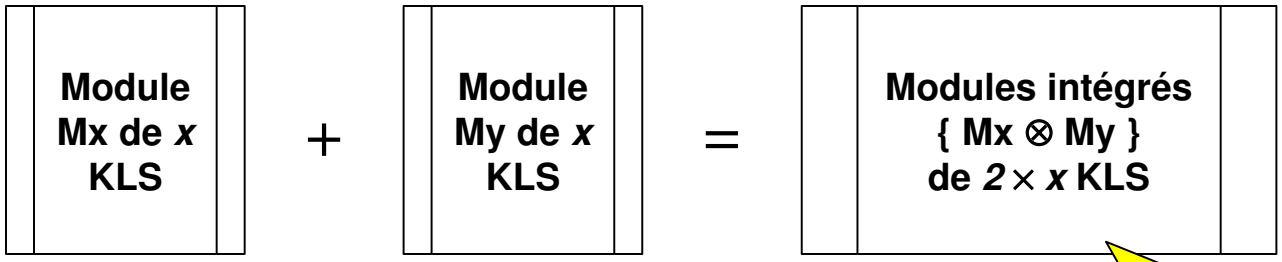
Dépend de la **puissance expressive** et du « grain » sémantique du langage de programmation
+ ***Expérience des acteurs***

Facteurs de coût

Dépend de la **complexité** de l'application et de la maturité du processus de développement
α est le ***facteur d'intégration***

Facteurs d'échelle

Coût de l'intégration de n modules



$$Eff_x(n) = n \times Eff_x(1) \times \underbrace{CUI(n)}_{\text{Coût Unitaire des Interaction entre les } n \text{ modules}}$$

La taille du code est invariante mais la quantité d'information a nécessairement augmentée
• Le pb est : forme de cette croissance ?

CUI a 2 origines :

Interactions entre les acteurs

Complexité résultant de l'organisation et des méthodes de travail

Interactions entre les modules intégrats de rang 0

Complexité résultant de l'architecture choisie

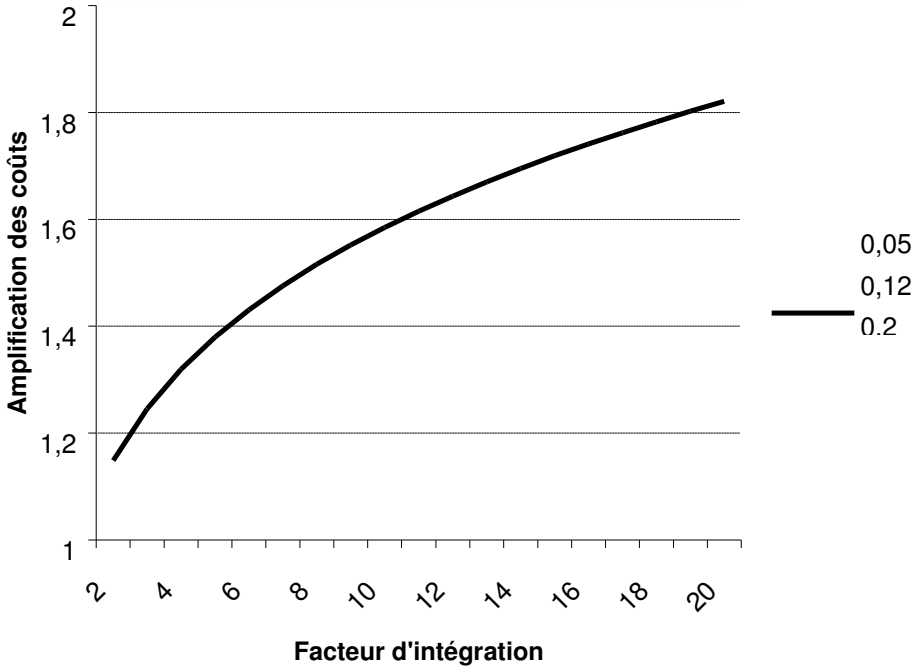
Si aucune interaction *CUI*=1

Architecture modulaire – Impact du coefficient d'intégration α

Tout ajout de code génère un coût d'intégration qui ne dépend que du terme complexité

$$\xi = \frac{EFF(n \times x)}{n \times EFF(x)} = n^{\alpha}$$

Ne dépend que du nombre de modules intégrés



Courbes établies à partir des équations COCOMO

La complexité vécue par les intégrateurs

Taille des programmes et des tests

En nombre de lignes source et/ou de points de fonctions

Nombre de modules élémentaires à intégrer

Intégrat de rang 0

Intégrats agrégés ayant fait eux-mêmes l'objet d'une intégration préalable

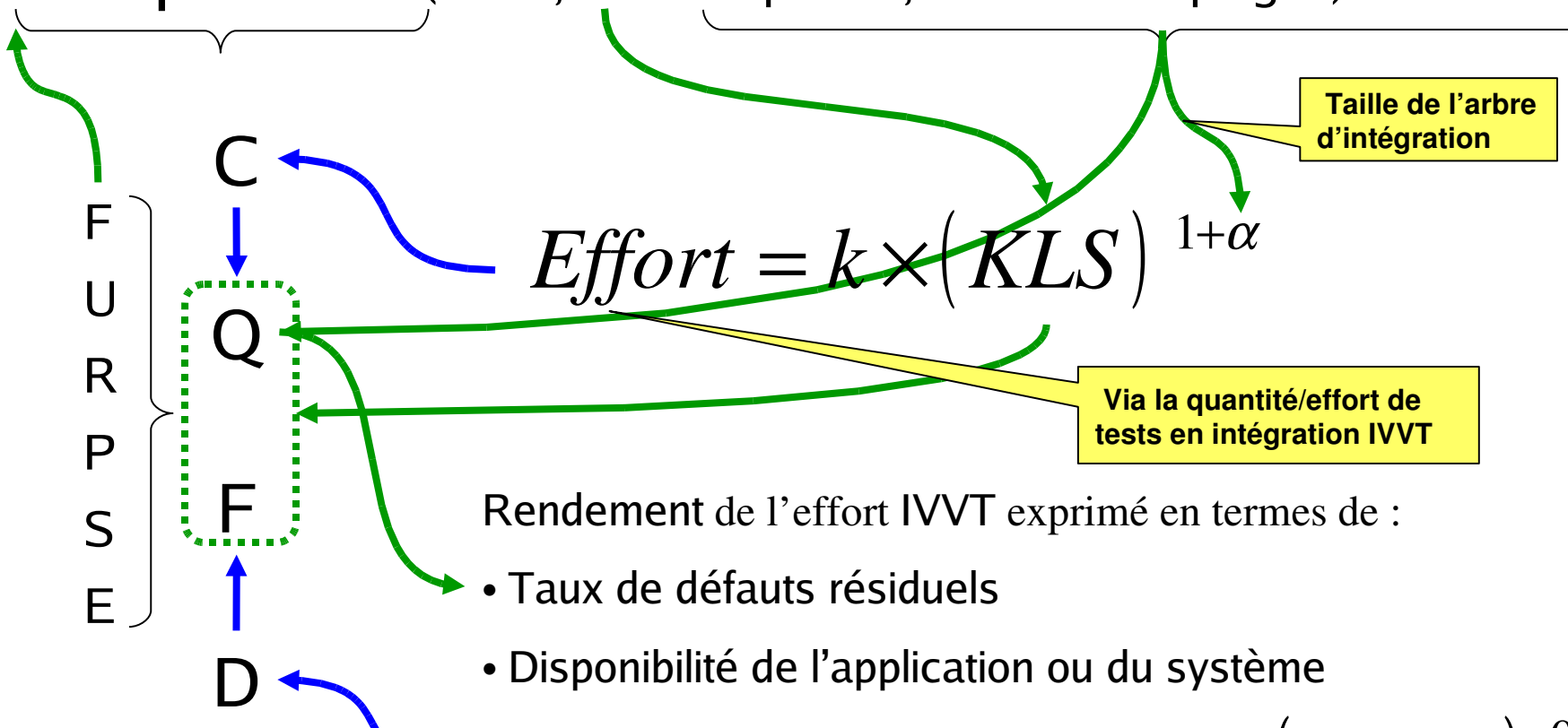
Combinatoire des relations entre ces modules : couplages – Contrats d'interfaces

Dépendances entre les intégrats (enchaînements, données et événements)

Couplages explicites avec contrats – Couplages implicites (canaux cachés n'ayant pas fait l'objet de contrat)

Relation entre la complexité, les équations COCOMO, CQFD et FURPSE

Complexité → (Taille, Nbre de pièces, Nbre de couplages)



Pour aller plus loin …