GESTION DE PROJETS

Gestion du temps
Estimation de l'effort
(développement de logiciels)

GP031
v102c
2020-02-10

Luc LAVOIE
Département d'informatique
Faculté des sciences

UNIVERSITÉ DE SHERBROOKE

Luc.Lavoie@USherbrooke.ca
http://info.usherbrooke.ca/llavoie

Plan

- Méthode analogique
- Méthodes paramétriques
  - « ad hoc »
  - Halstead
  - COCOMO
  - FP
- Méthodes synthétiques
  - Méthode ascendante (le fleuve Saint-Laurent)
  - Méthode descendante (le delta du Mékong)
  - Méthode mixte (les basses terres d'Europe)
- Synthèse

Méthode analogique

- Mettre en relation des noeuds de la SDP du projet à estimer avec ceux de la SDP d'un projet analogue, bien documenté.
- Ajuster les quantités en fonction des facteurs différentiels.
- Recourir à des sources généralement bien informées pour compléter (sic) :
  - votre petit doigt (*thumbs rule*),
  - un vétéran,
  - des experts (Delphi).

Méthodes paramétriques

- Toute méthode fondée
  - sur les expériences antérieures
- Permettant d'estimer l'effort sur la base
  - d'une fonction mathématique
  - dont les paramètres
    - macroscopiques
    - sont simples à évaluer
    - au moment de la définition de l'activité (ou de l'artéfact)

Méthode paramétrique « ad hoc »

- Applicable
  - aux artéfacts stéréotypés
- Fondée
  - sur les expériences antérieures
- Construite (le plus souvent) par
  - la combinaison linéaire
  - de variables
  - dont les coefficients
    - empiriques
    - sont issus des données historiques

HALSTEAD (1/2)

n1 = Nombre d'opérateurs distincts d'un composant

n2 = Nombre d'opérandes distincts d'un composant

N1 = Nombre total d'occurrences des opérateurs au sein du composant

N2 = Nombre total d'occurrences des opérandes au sein du composant

l = n1 + n2
Vocabulaire du programme

L = N1 + N2
Longueur observée du programme

Le = n1 (Log2 n1) + n2 (Log2 n2)
Longueur estimée du programme

V = L Log2 (n1 + n2)
Volume du programme

D = (n1/2) (N2/n2)
Difficulté du programme

L1 = 1/D
Niveau du programme

T = V/L1 = V D
Travail (en del)

del : discriminations élémentaires

HALSTEAD (2/2)

- Erreurs
  - B = V/K
- Effort (seconde-personne)
  - E = T/S

- où
  - S est le nombre de Stroud (typiquement 18 del/seconde)
  - K est le coefficient de fiabilité (typiquement 3000 del/erreur)

Méthode paramétrique COCOMO
Plan

- Modèle de calcul
  - empirique
  - prévisionnel
  - macroscopique
- Hypothèses
  - Existence d'une mesure uniforme du travail
  - Ressources adéquates et opportunes
- Prolongements
  - Ventilation par phase
  - Évaluation prévisionnelle de la qualité
  - Évaluation prévisionnelle des risques
- Composants
  - Formules de normalisation
    - Travail
  - Formules d'estimation
    - Effort
    - Durée
  - Tables paramétriques
    - Mise à l'échelle (SF)
    - Modularisation (EM)
    - Conversion
      - (UFP ⇨ KSLOC)
  - Guides de calibration
    - Tables paramétriques
  - Outils de calcul
    - Guides
    - Chiffriers
    - Logiciels spécialisés

Méthode paramétrique COCOMO
Références

Références

COCOMO II – Model Definition Manual, version 2.1
Center for Software Engineering, University of Southern California,
86 pages, 2000

Produits

http://www.softstarsystems.com

MÉTHODE PARAMÉTRIQUE COCOMO I

- Formules simplifiées
  - Effort : $PM = A \times KSLOC^{E}$
  - Durée : $TDEV = C \times PM^{F}$

Type	A	E	C	F
Autonome	2,40	1,05	2,50	0,38
Couplé	3,00	1,12	2,50	0,35
Embarqué	3,60	1,20	2,50	0,32

MÉTHODE PARAMÉTRIQUE COCOMO II

- Cheminement
  - 1 - Points de fonction (FP)
  - 2 - Travail (KSLOC)
  - 3 - Effort (PM)
  - 4 - Durée (TDEV)

- Note
  - On peut utiliser d'autres modèles d'estimation du travail brut que celui des points de fonctions

Méthode paramétrique COCOMO II

- Points de fonction
  - Déterminer
    - Exigences
    - Architecture
    - Langages
  - En déduire le travail (brut)
- Travail
  - Estimer
    - Réutilisation
    - Évolution (REVL)
    - Automatisation
  - Calculer travail (net)
  - En déduire l'effort
- Effort
  - Mettre à l'échelle (SF)
  - Modulariser (EM)
    - Pré-développement
    - Post-développement
- Durée
  - Mettre à l'échelle
  - Contextualiser

COCOMO II – SITUATION « INITIALE »

- Situation initiale
  - Développement 100%
  - Échéancier non contraint (*NS : nominal schedule*)
    - A = 2,94 B = 0,91
    - C = 3,67 D = 0,28

$$PM_{NS} = A \times Size^{E} \times \prod_{i=1}^{n} EM_i$$

$$\text{where } E = B + 0.01 \times \sum_{j=1}^{5} SF_j$$

- EM (*effort multiplier*)
  - pour moduler l'effort
- SF (*scale factor*)
  - pour moduler la durée

$$TDEV_{NS} = C \times (PM_{NS})^{F}$$

$$\text{where } F = D + 0.2 \times 0.01 \times \sum_{j=1}^{5} SF_j$$

$$= D + 0.2 \times (E - B)$$

Réf.: [a], extraits des pages 9-15

COCOMO II – SF (FACTEURS D'ÉCHELLE)

- PREC - *Precedentedness scale factor*.
  Facteur d'échelle prenant en compte l'expérience relativement à des projets similaires (tant celle de l'équipe que celle de l'organisation).
- FLEX - *Development Flexibility scale factor*.
  Facteur d'échelle prenant en compte la flexibilité des exigences (du système), soit la capacité de faire varier celles-ci dans la mesure où les objectifs du projet sont atteints.
- TEAM - *Team Cohesion scale factor*.
  Facteur de cohésion de l'équipe prenant principalement en compte la qualité et la facilité de communication entre les membres de l'équipe.
- RESL - *Architecture and Risk Resolution scale factor*.
  Facteur d'échelle prenant en compte la capacité (de l'équipe) de traiter la complexité de l'architecture du produit et les autres risques techniques de projet.
- PMAT - *Process Maturity scale factor*.
  Facteur d'échelle associé au niveau de maturité du processus de développement logiciel (de l'organisation) selon la grille du CMM (CMMI).

COCOMO II – SF (FACTEURS D'ÉCHELLE) - BIS

- PREC - *Precedentedness scale factor*.
  Facteur d'échelle prenant en compte l'expérience relativement à des projets similaires (tant celle de l'équipe que celle de l'organisation).
- FLEX - *Development Flexibility scale factor*.
  Facteur d'échelle prenant en compte la flexibilité des exigences (du système), soit la capacité de faire varier celles-ci dans la mesure où les objectifs du projet sont atteints.
- TEAM - *Team Cohesion scale factor*.
  Facteur de cohésion de l'équipe prenant principalement en compte la qualité et la facilité de communication entre les membres de l'équipe.
- RESL - *Architecture and Risk Resolution scale factor*.
  Facteur d'échelle prenant en compte la capacité (de l'équipe) de traiter la complexité de l'architecture du produit et les autres risques techniques de projet.
- PMAT - *Process Maturity scale factor*.
  Facteur d'échelle associé au niveau de maturité du processus de développement logiciel (de l'organisation) selon la grille du CMM (CMMI).

COCOMO II – SF (LA MATRICE)

Table 10. Scale Factor Values, SF_j, for COCOMO II Models

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC	thoroughly unprecedented	largely unprecedented	somewhat unprecedented	generally familiar	largely familiar	thoroughly familiar
SF_j:	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	rigorous	occasional relaxation	some relaxation	general conformity	some conformity	general goals
SF_j:	5.07	4.05	3.04	2.03	1.01	0.00
RESL	little (20%)	some (40%)	often (60%)	generally (75%)	mostly (90%)	full (100%)
SF_j:	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions
SF_j:	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	The estimated Equivalent Process Maturity Level (EPML) or					
	SW-CMM Level 1 Lower	SW-CMM Level 1 Upper	SW-CMM Level 2	SW-CMM Level 3	SW-CMM Level 4	SW-CMM Level 5
SF_j:	7.80	6.24	4.68	3.12	1.56	0.00

Réf.: [a], extraits des pages 18

COCOMO II – EM (MULTIPLICATEURS D'EFFORT)

- Quatre catégories de facteurs :
  - produit
  - plate-forme
  - personnel
  - projet

COCOMO II – EM : FACTEURS LIÉS AU PRODUIT

- RELY
  niveau de fiabilité recherché
  - (semble inclure la validité)
- DATA
  envergure des données rémanentes
  - (à partir du ratio D/P ≡ Ko/KSLOC)
- CPLX
  mesure intégratrice de la complexité du logiciel selon :
  - algorithmique,
  - calcul,
  - entrées-sorties,
  - gestion des données,
  - IPM
- RUSE
  développement en vue d'une réutilisation
- DOCU
  envergure, qualité et complexité de la documentation

COCOMO II – EM : FACTEURS LIÉS À LA PLATE-FORME

- TIME
  exigence en temps relativement à la
  - capacité de la plate-forme visée
- STOR
  exigence en espace relativement à la
  - capacité de la plate-forme visée
- PVOL
  volatilité de la plate-forme relativement aux
  - délais entre les mises à jour

COCOMO II – EM : FACTEURS LIÉS AU PERSONNEL

- ACAP
  évaluation des analystes
  (en percentile)
- PCAP
  évaluation des concepteurs (programmeurs)
  (en percentile)
- PCON
  évaluation de la stabilité du personnel
  (relativement au pourcentage de mobilité annuelle)
- APEX
  évaluation de l'expérience relative au domaine d'application
- PLEX
  évaluation de l'expérience relative à la plate-forme
- LTEX
  évaluation de l'expérience relative à l'environnement de développement

COCOMO II – EM : FACTEURS LIÉS AU PROJET

- TOOL
  qualité, couverture et ergonomie de l'environnement de développement
- SITE Collaboration
  infrastructure de soutien au travail collaboratif
- SITE Communication
  infrastructure de communication
- SCED
  facteur de compression (ou d'expansion) de l'échéancier (en pourcentage)

COCOMO II – EM : SYNTHÈSE (1/3)

Cost Drivers	Very Low	Low	Nominal	High	Very High	Extra High
RELY	slight inconvenience	low, easily recoverable losses	moderate, easily recoverable losses	high financial loss	risk to human life	
DATA		Testing DB bytes / Pgm SLOC < 10	10 ≤ D/P < 100	100 ≤ D/P < 1000	D/P > 1000	
CPLX	*** voir tableau complémentaire ***					
RUSE		none	across project	across program	across product line	across multiple product lines
DOCU	Many life-cycle needs uncovered	Some life-cycle needs uncovered.	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very excessive for life-cycle needs	
TIME			≤ 50% use of available execution time	70%	85%	95%
STOR			≤ 50% use of available storage	70%	85%	95%

COCOMO II – EM : SYNTHÈSE (2/3)

PVOL		major change every 12 mo.; minor change every 1 mo.	major: 6 mo.; minor: 2 wk.	major: 2 mo.; minor: 1 wk.	major: 2 wk.; minor: 2 days	
ACAP	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
PCAP	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
PCON	48% / year	24% / year	12% / year	6% / year	3% / year	
APEX	≤ 2 months	6 months	1 year	3 years	6 years	
PLEX	≤ 2 months	6 months	1 year	3 years	6 year	
LTEX	≤ 2 months	6 months	1 year	3 years	6 year	
TOOL	edit, code, debug	simple, frontend, backend CASE, little integration	basic lifecycle tools, moderately integrated	strong, mature lifecycle tools, moderately integrated	strong, mature, proactive lifecycle tools, well integrated with processes, methods, reuse	

COCOMO II – EM : SYNTHÈSE (3/3)

Cost Drivers	Very Low	Low	Nominal	High	Very High	Extra High
SITE: Collo-cation	International	Multi-city **and** multi-company	Multi-city **or** multi-company	Same city or metro area	Same building or complex	Fully collocated
SITE: Com-muni-cation	Some phone, mail	Individual phone, FAX	Narrow-band email	Wide-band electronic communica-tion.	Wide-band elect. comm, occasional video conf.	Interactive multimedia
SCED	75% of nominal	85% of nominal	100% of nominal	130% of nominal	160% of nominal	

ESTIMATION DE COMPLEXITÉ (1/2)

...

C'est complexe !

	Control Operations	Computational Operations	Device-dependent Operations	Data Management Operations	User Interface Management Operations
Very Low	Straight-line code with a few non-nested structured programming operators: DOs, CASEs, IF-THEN-ELSEs. Simple module composition via procedure calls or simple scripts.	Evaluation of simple expressions: e.g., A=B+C*(D-E)	Simple read, write statements with simple formats.	Simple arrays in main memory. Simple COTS-DB queries, updates.	Simple input forms, report generators.
Low	Straightforward nesting of structured programming operators. Mostly simple predicates	Evaluation of moderate-level expressions: e.g., D=SQRT(B**2-4.*A*C)	No cognizance needed of particular processor or I/O device characteristics. I/O done at GET/PUT level.	Single file subsetting with no data structure changes, no edits, no intermediate files. Moderately complex COTS-DB queries, updates.	Use of simple graphic user interface (GUI) builders.
Nominal	Mostly simple nesting. Some intermodule control. Decision tables. Simple callbacks or message passing, including middleware-supported distributed processing	Use of standard math and statistical routines. Basic matrix/vector operations.	I/O processing includes device selection, status checking and error processing.	Multi-file input and single file output. Simple structural changes, simple edits. Complex COTS-DB queries, updates.	Simple use of widget set.

ESTIMATION DE COMPLEXITÉ (2/2)

« The complexity rating is the subjective weighted average of the selected area ratings. »

Cocomo II Model Reference Manual

p. 26

	Control Operations	Computational Operations	Device-dependent Operations	Data Management Operations	User Interface Management Operations
High	Highly nested structured programming operators with many compound predicates. Queue and stack control. Homogeneous, distributed processing. Single processor soft real-time control.	Basic numerical analysis: multivariate interpolation, ordinary differential equations. Basic truncation, round-off concerns.	Operations at physical I/O level (physical storage address translations; seeks, reads, etc.). Optimized I/O overlap.	Simple triggers activated by data stream contents. Complex data restructuring.	Widget set development and extension. Simple voice I/O, multimedia.
Very High	Reentrant and recursive coding. Fixed-priority interrupt handling. Task synchronization, complex callbacks, heterogeneous distributed processing. Single-processor hard real-time control.	Difficult but structured numerical analysis: near-singular matrix equations, partial differential equations. Simple parallelization.	Routines for interrupt diagnosis, servicing, masking. Communication line handling. Performance-intensive embedded systems.	Distributed database coordination. Complex triggers. Search optimization.	Moderately complex 2D/3D, dynamic graphics, multimedia.
Extra High	Multiple resource scheduling with dynamically changing priorities. Microcode-level control. Distributed hard real-time control.	Difficult and unstructured numerical analysis: highly accurate analysis of noisy, stochastic data. Complex parallelization.	Device timing-dependent coding, micro-programmed operations. Performance-critical embedded systems.	Highly coupled, dynamic relational and object structures. Natural language data management.	Complex multimedia, virtual reality, natural language interface.

COCOMO II – CORRIGER LA SITUATION « INITIALE »

- Réutilisation
  - ajuster le travail ($KSLOC_{EQ} \leftarrow KSLOC_{NA}$)
- Automatisation
  - scinder le calcul de l'effort en deux
    - travail automatisé (PM_{auto})
    - travail non automatisé (PM_{NS})
  - $PM = PM_{NS} + PM_{auto}$
- Volatilité des exigences
- Compression de l'échéancier

COCOMO II – RÉUTILISATION

$$\text{Equivalent KSLOC} = \text{Adapted KSLOC} \times \left(1 - \frac{\text{AT}}{100}\right) \times \text{AAM}$$

$$\text{where AAM} = \begin{cases} \dfrac{[\text{AA} + \text{AAF}(1 + (0.02 \times \text{SU} \times \text{UNFM}))]}{100}, & \text{for AAF} \leq 50 \\ \dfrac{[\text{AA} + \text{AAF} + (\text{SU} \times \text{UNFM})]}{100}, & \text{for AAF} > 50 \end{cases}$$

$$\text{AAF} = (0.4 \times \text{DM}) + (0.3 \times \text{CM}) + (0.3 \times \text{IM})$$

AA : Assessment and assimilation increment
AT : Automated translation (portion automatisable du travail)
SU : Understanding increment
UNFM : Programmer unfamiliarity
DM : Design modified proportion
CM : Code modified proportion
IM : Integration required proportion

COCOMO II – RÉUTILISATION (SU)

Table 5. Rating Scale for Software Understanding Increment SU

	Very Low	Low	Nominal	High	Very High
Structure	Very low cohesion, high coupling, spaghetti code.	Moderately low cohesion, high coupling.	Reasonably well-structured; some weak areas.	High cohesion, low coupling.	Strong modularity, information hiding in data / control structures.
Application Clarity	No match between program and application world-views.	Some correlation between program and application.	Moderate correlation between program and application.	Good correlation between program and application.	Clear match between program and application world-views.
Self-Descriptive-ness	Obscure code; documentation missing, obscure or obsolete.	Some code commentary and headers; some useful documentation.	Moderate level of code commentary, headers, documentation.	Good code commentary and headers; useful documentation; some weak areas.	Self-descriptive code; documentation up-to-date, well-organized, with design rationale.
SU Increment to ESLOC	50	40	30	20	10

COCOMO II – RÉUTILISATION (AA ET UNFM)

Table 6. Rating Scale for Assessment and Assimilation Increment (AA)

AA Increment	Level of AA Effort
0	None
2	Basic module search and documentation
4	Some module Test and Evaluation (T&E), documentation
6	Considerable module T&E, documentation
8	Extensive module T&E, documentation

Table 7. Rating Scale for Programmer Unfamiliarity (UNFM)

UNFM Increment	Level of Unfamiliarity
0.0	Completely familiar
0.2	Mostly familiar
0.4	Somewhat familiar
0.6	Considerably familiar
0.8	Mostly unfamiliar
1.0	Completely unfamiliar

COCOMO II – RÉUTILISATION (DM, CM, IM)

Table 8. Adapted Software Parameter Constraints and Guidelines

Code Category	Reuse Parameters DM	CM	IM	AA	SU	UNFM
New all original software			not applicable			
Adapted changes to preexisting software	0% - 100% normally > 0%	0'% - 100% usually > DM and must be > 0%	0% - 100+% IM usually moderate and can be > 100%	0% – 8%	0% - 50%	0 - 1
Reused unchanged existing software	0%	0%	0% - 100% rarely 0%, but could be very small	0% – 8%	not applicable	
COTS off-the-shelf software (often requires new glue code as a wrapper around the COTS)	0%	0%	0% - 100%	0% – 8%	not applicable	

COCOMO II – AUTOMATISATION

$$PM_{Auto} = \frac{\text{Adapted SLOC} \times \left({}^{AT}/_{100} \right)}{ATPROD}$$

AT : portion automatisable du travail
ATPROD : travail automatisé produit par mois-personne
AUTO : automatisation

Réf.: [a], extraits des pages 9-15

COCOMO II – ET LE RESTE

- Volatilité des exigences

$$\text{Size} = \left(1 + \frac{\text{REVL}}{100}\right) \times \text{Size}_{\text{D}}$$

- Compression d'échéancier
  - voir le manuel du modèle

FP – L'IDÉE

- Objectif
  - déterminer la *taille* d'une application sur une base fonctionnelle
- Corollaires
  - l'estimation est réalisable sur la base des seules exigences fonctionnelles
  - l'estimation peut être affinée par la suite sur la base des
    - exigences non fonctionnelles
    - architecture
    - conception globale

FP – LES MÉTHODES

- version 0 – Boehm
- version 1 – IFPUG
- version 2 – COSMIC
- version 3 – ISO
- version 3.01 – COSMIC-FP

FP0 – LA MÉTHODE

- Identifier les fonctions
- Déterminer les points de fonction
- Pondérer les points de fonction
- Caractériser les points de fonction
- Pour chaque fonction *f*, calculer
  - $AFP[f] = UFP[f] \times (0{,}65 + 0{,}01 * TGC[f])$

  - *AFP : adjusted function point*
  - *UFP : unadjusted function point*
  - *TGC : total global complexity*

FP0 – DÉTERMINATION (UFP[F])

Table 1. User Function Types

Function Point	Description
External Input (EI)	Count each unique user data or user control input type that enters the external boundary of the software system being measured.
External Output (EO)	Count each unique user data or control output type that leaves the external boundary of the software system being measured.
Internal Logical File (ILF)	Count each major logical group of user data or control information in the software system as a logical internal file type. Include each logical file (e.g., each logical group of data) that is generated, used, or maintained by the software system.
External Interface Files (EIF)	Files passed or shared between software systems should be counted as external interface file types within each system.
External Inquiry (EQ)	Count each unique input-output combination, where input causes and generates an immediate output, as an external inquiry type.

Paramètre	simple	moyen	complexe
Intrant (EI)	3	4	6
Extrant (EO)	4	5	7
Requête (EQ)	3	4	6
Structure interne (ILF)	7	10	15
Structure externe (EIF)	5	7	10

FP0 –CATÉGORISATION (UFP[F])

Table 2. FP Counting Weights

For Internal Logical Files and External Interface Files

	Data Elements		
Record Elements	1 - 19	20 - 50	51+
1	Low	Low	Avg.
2 - 5	Low	Avg.	High
6+	Avg.	High	High

For External Output and External Inquiry

	Data Elements		
File Types	1 - 5	6 - 19	20+
0 or 1	Low	Low	Avg.
2 - 3	Low	Avg.	High
4+	Avg.	High	High

For External Input

	Data Elements		
File Types	1 - 4	5 - 15	16+
0 or 1	Low	Low	Avg.
2 - 3	Low	Avg.	High
3+	Avg.	High	High

FP0 – CARACTÉRISATION (TGC)

1. sauvegarde/restauration
2. communication
3. processus distribué
4. criticité de performance
5. environnement existant et achalandé
6. saisie en ligne
7. multiplicité des contextes de saisie
8. mise à jour en ligne
9. complexité générale des paramètres (!)
10. complexité générale du traitement
11. développement en vue de la réutilisation
12. migration et mise en exploitation
13. déploiements multiples
14. prise en charge ergonomique

... à raison de 5 points par critère

COCOMO II – (FP -> TRAVAIL) : SLOC / UFP

Table 4. UFP to SLOC Conversion Ratios

Language	Default SLOC / UFP	Language	Default SLOC / UFP
Access	38	Jovial	107
Ada 83	71	Lisp	64
Ada 95	49	Machine Code	640
AI Shell	49	Modula 2	80
APL	32	Pascal	91
Assembly - Basic	320	PERL	27
Assembly - Macro	213	PowerBuilder	16
Basic - ANSI	64	Prolog	64
Basic - Compiled	91	Query – Default	13
Basic - Visual	32	Report Generator	80
C	128	Second Generation Language	107
C++	55	Simulation – Default	46
Cobol (ANSI 85)	91	Spreadsheet	6
Database – Default	40	Third Generation Language	80
Fifth Generation Language	4	Unix Shell Scripts	107
First Generation Language	320	USR_1	1
Forth	64	USR_2	1
Fortran 77	107	USR_3	1
Fortran 95	71	USR_4	1
Fourth Generation Language	20	USR_5	1
High Level Language	64	Visual Basic 5.0	29
HTML 3.0	15	Visual C++	34
Java	53		

FP0 – LA MÉTHODE DÉTAILLÉE

- Identifier les fonctions
- Déterminer les points de fonction
- Pondérer les points de fonction
- Caractériser les points de fonction
- Pour chaque fonction *f*, calculer
  - AFP[*f*] = UFP[*f*] x (0,65 + 0,01 * TGC**[*f*]**)

  - *AFP : adjusted function point*
  - *UFP : unadjusted function point*
  - *TGC : total global complexity*

FP - RÉFÉRENCES

- Références
  - Functional measurement meta-standard
    - ISO 12143-1
  - IFPUG
    - www.ifpug.org/
  - COSMIC 3.0 et ISO
    - ISO 19761
  - COSMIC 3.01
    - http://www.cosmicon.com/

Méthode ascendante

- La somme des quantités associées aux descendants est associée à l'ancêtre.
- La subdivision de tâches n'étant généralement pas à coût nul, une estimation du coût induit doit être ajoutée :
  - le coût est principalement déterminé par les tâches de synchronisation des participants et d'assemblage des artéfacts ;
  - en général, ce coût croît en fonction du carré du nombre des parties.

Méthode descendante

- Répartir itérativement la quantité associée à un noeud entre ses descendants.
- La subdivision de tâches n'étant généralement pas à coût nul, une estimation du coût induit doit être ajoutée :
  - le coût est principalement déterminé par les tâches de synchronisation des participants et d'assemblage des artéfacts ;
  - en général, ce coût croît en fonction du carré du nombre des parties.

Méthode mixte

- Utiliser une méthode d'estimation pour quantifier les composants d'une architecture.
- Utiliser la méthode ascendante pour estimer le coût du système (la racine).
- Utiliser la méthode descendante pour estimer le coût des sous-composants (les feuilles).

Synthèse - Principes

- Une estimation est une projection fondée sur des expériences passées prenant en compte les différences entre les cas passés et le cas ciblé.
- Toute estimation repose sur un ensemble d'hypothèses et de contraintes.
- La ré-estimation est impérative :
  - lors du suivi de projet ;
  - lors de modifications aux hypothèses, aux contraintes, aux risques ou aux exigences.

Synthèse -Constats

- La fabrication d'un échéancier requiert au moins un degré de liberté.
- La compression multi-paramétrique et il est difficile d'identifier tous les paramètres.
- La compression est non linéaire.
- La compression a une limite.

Synthèse - Conclusion

- Pas d'estimation
  - sans marge d'erreur,
  - sans données historiques,
  - sans ré-estimation.
- Complémentarité des méthodes
  - A: Paramétrique, historique – ascendante
  - B: FP, Cocomo – descendante
  - C: Analogique, PERT – ascendante
  - D: Analogique, Monte-Carlo – mixte
  - A+B, A+D
  - C+B, C+D
  - B+D