Mètriques de programari

GESI-EPSEVG-UPC
Jordi Esteve 2009

Mètriques de programari

1. Objectius de les mètriques de programari
2. Mètriques. Definicions. Unitats de mesura
3. Mètriques basades en la grandària
4. Mètriques basades en les funcionalitats: Punts de Funció
5. Extensió 3D dels Punts de Funció

Objectiu de les mètriques de programari

- PVP = COST + BENEFICIS
- Estimar el cost d'un projecte abans de portar-lo a terme
- Cost: Recursos + Diners + Temps
- Idea: saber quant ens costarà (recursos, temps i diners) per poder-li dir al client una data d'entrega i un preu

Formalment, les mètriques de SW serveixen per a:

- Indicar la qualitat del producte
- Avaluar la productivitat de les persones que desenvolupen el producte
- Avaluar els beneficis (en termes de productivitat i de qualitat) derivats de l'ús de nous mètodes i eines d'enginyeria de SW
- Establir una *baseline* per a l'estimació temporal
- Ajudar a justificar l'ús de noves eines o de formació addicional

Mètrica de programari. Definicions

Una **mètrica de programari** és una mesura quantitativa del grau en què un sistema, un producte o un procés poseeix un determinat atribut.

Les diverses mètriques intenten avaluar tres grans magnituds generals:

- La **qualitat** i la **grandària** (sovint del producte)
- La **productivitat** (freqüentment del procés de construcció del producte)

Indicadors. Unitats de mesura

Exemples d'unitats de mesura i indicadors que mesuren:

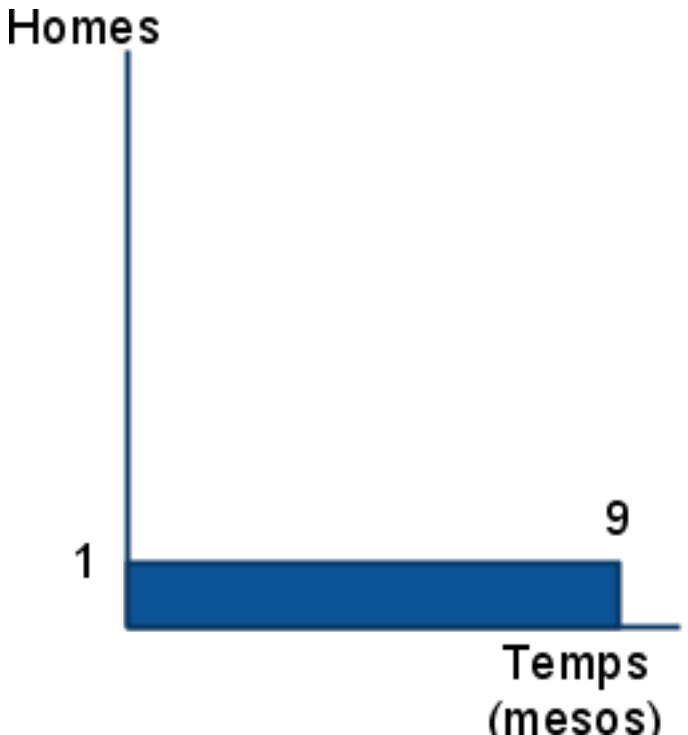
- Línies de codi per persona i dia (indicador de productivitat)
- Hores per implementar un punt de funció (indicador de productivitat)
- Nombre d'errors per cada mil línies de codi (indicador de qualitat)
- Euros per cada miler de línies de codi (indicador de cost)
- Pàgines de documentació per cada mil línies de codi (indicador de qualitat de la documentació)

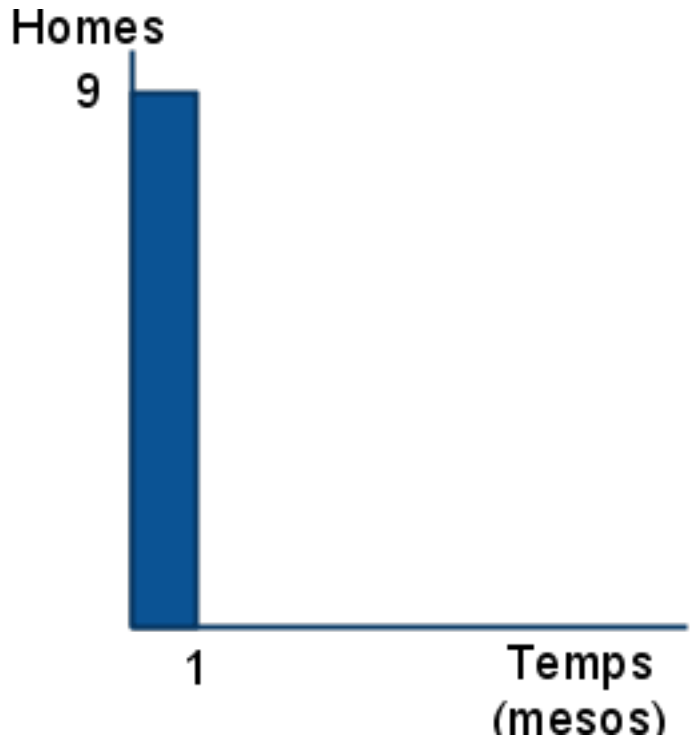
Mesures de l'esforç. Mite home-mes

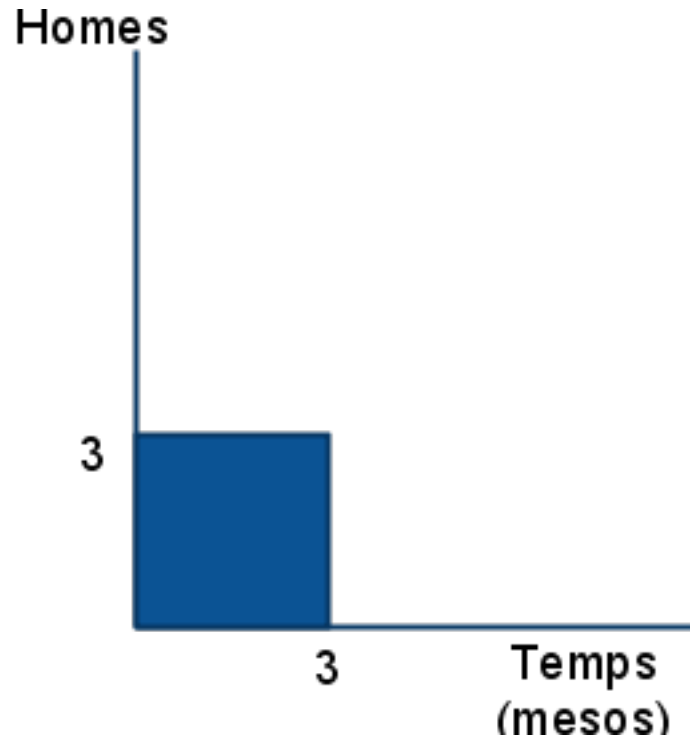
Les mesures habituals de l'esforç es redueixen sempre a la feina que fa un professional mitjà en una determinada unitat de temps (persona-dia, persona-mes, persona-any)

persona-mes= home-mes= man-month= MM= staff-month

Mite home-mes: Els mesos i els homes no són intercanviables.







Mètriques basades en la grandària

Compten la qualitat de línies de codi que caldran per desenvolupar una aplicació.

- Unitats de mesura: LOC, KLOC, ...
- Mesuren la productivitat en KLOC/persona-mes i la qualitat en errors/KLOC.
- Problemes:
  - Conversió entre llenguatges (un mateix mètode té diferents LOC en diferents llenguatges i en diferents paradigmes de programació).
  - Poc útil per al programari actuals amb milions de línies de codi i per al paradigma de la orientació a objectes.
  - Hi ha línies de codi (o parts de programes) que tenen un cost més elevat que altres (complexitat...)

Unitats de mesura de línies de codi

- LOC: línies de codi (Lines of Code)
- KLOC: milers de línies de codi (Kilo LOC)
- DSI i KDSI: instruccions de codi font realment lliurades (Delivered Source Instructions)
- NCSS i KNCSS: línies de codi font sense tenir en compte els comentaris (Non Comment Source Statements)
- NSLOC: noves línies de codi font (New Source LOC)

Ràtios de productivitat:

- 10 LOC per dia i persona ocupada en el projecte (Barry W. Boehm a inicis dels anys vuitanta).
- 350 NCSS per persona i mes (Robert O. Grady a inicis dels anys noranta en projectes de l'empresa Hewlett Packard). Serien gairebé 16 LOC per persona-dia.

Comparativa LOC - llenguatges

Esforç i línies de codi per a desenvolupar la mateixa funcionalitat en diversos llenguatges

Llenguatges	Grandària (LOC)	Esforç (persona-mes)	LOC per persona-any
Assemblador	10.000	40	3.000
Macroassemblador	6.666	28	2.856
C	5.000	22	2.727
COBOL	3.333	26	2.500
Pascal	2.500	13	2.307
Ada	2.222	12	2.222
BASIC	2.000	11	2.181

La productivitat aparent

En el cas concret del COBOL

Taxa	LOC persona-any
Codificació	25.000
Disseny, codificació i proves d'un sol mòdul	12.000
Disseny, codificació i proves d'un programa complet	6.000
Activitat d'un programador durant tot un any (interrupcions, altres feines)	4.000
L'anterior + estudi d'oportunitat, anàlisi de requeriments, activitats de control, gestió de la qualitat i documentació	2.500
L'anterior + suport per un centenar d'usuaris i la seva formació	750
Esforç total en programari d'una organització	250

Mètriques basades en les funcionalitats: Punts Funció (I)

- Vinculada directament amb el QUÈ ha de fer el sistema i no en COM ho ha de fer

- Es basa la mesura en el càlcul del cost de:
  - Funcions de dades:
    - ILF: Internal Logical Files
    - ELF: External Logical Files
  - Funcions de transaccions
    - EI: External Inputs
    - EO: External Outputs
    - EQ: External Querys

Mètriques basades en les funcionalitats: Punts Funció (II)

- ILF: Internal Logical File
  - Grup de dades del món real reconeguts per a l'usuari i que ha d'estar representats al sistema.
  - Les hem de mantenir (actualitzar) dins de la frontera de la nostra aplicació.
  - Exemple: classes d'objectes, taules de la b.d.
-
- ELF: External Logical File
  - Grup de dades o informació de control reconeguda per l'usuari.
  - Es mantenen for a de la frontera de l'aplicació.
  - Exemple: cotització canvi €/$

Mètriques basades en les funcionalitats: Punts Funció (III)

- EQ: External Inputs
  - Qualsevol "cosa" que creuï la frontera del sistema de for a cap a dins.
  - Exemples: dades d'un formulari, fitxers de text d'entrada...
- EO: External Outputs
  - Qualsevol "cosa" que creuï la frontera dels sistema de dins cap a for a i que requereixi un procés important (càlculs matemàtics...)
  - Exemples: generació de factures, nòmines...
- EQ: External Queries
  - Peticions al sistema que no requereixin operacions complexes.
  - Exemples: llistats d'usuaris, informes...

Mètriques basades en les funcionalitats: Punts Funció (IV)

- Comptatge de Pfs:

				Factor de pes				
Paràmetre		Qtat		Simple	Mitjà	Complex		
EI	X		X	3	4	6	=	
EO	X		X	4	5	7	=	
EQ	X		X	3	4	6	=	
ILF	X		X	7	10	15	=	
ELF	X		X	5	7	10	=	

COMPTE TOTAL =

Mètriques basades en les funcionalitats: Punts Funció (V)

- Però un programari no és només requeriments funcionals, n'hi ha de no-funcionals molt costosos.
- S'utilitzen llavors els punts de característica [ART85]
- Per detectar-los i comptabilitzar-los, s'ha de contestar a les següents 14 preguntes amb un valora entre 0 (sense influència) i 5 (essencial). El valor final estarà entre 0 i 70.

Mètriques basades en les funcionalitats: Punts Funció (VI)

- Requereix el sistema copies de seguretat i de recuperació fiables?
- Es requereix comunicació de dades?
- Existeixen funcions de procés distribuït?
- És crític el rendiment?
- Serà executat en un entorn operatiu existent i fortament usat?
- Requereix el sistema l'entrada de dades interactiva?
- Requereix l'entrada de dades interactiva que les transaccions d'entrada es desenvolupin sobre múltiples pantalles o diverses operacions?
- S'actualitzen els *master files* de forma interactiva?

Mètriques basades enles funcionalitats: Punts Funció (VI)

- Són complexes les entrades, les sortides, els arxius o les peticions (EI, EO, EQ, ELF, ILF)?
- És complex el processament intern?
- S'ha dissenyat el codi per a ser reutilitzable?
- Estan incloses al disseny la conversió i la instal·lació?
- S'ha dissenyat el sistema per a suportat múltiples instal·lacions en diferents organitzacions?
- S'ha dissenyat l'aplicació per tal de facilitar els canvis i per a ser fàcilment emprada per l'usuari final?

Mètriques basades en les funcionalitats: Punts Funció (VII)

- Finalment, el càlcul de Punts de Funció es fa amb la següent fòrmula:

$$PF = COMPTE\ TOTAL + [0.65 + 0.01 * SUM(F_i)]$$

on:

  - COMPTE TOTAL és el resultat de la taula.
  - SUM (F_i) és la suma dels valors de la resposta a les preguntes dels punts de característica.

- Compte!!! Massa matemàtica vol donar-li sentit científic, però no és més exacte necessàriament.

Extensió 3D dels Punts Funció (I)

- Quan són els EIs, EOs, EQs, ILFs i ELFs... simples? Mitjans? Complexos?
- Dificultat per determinar la complexitat del sistema.
- Idea: determinar-la a partir de l'experiència de cada grup de desenvolupament i en funció dels costos de resolució de problemes similars anteriors.
- Guia per neòfits: Extensió 3D dels PF.

Extensió 3D dels Punts Funció (II)

- Per a ILF i ELF:
  - Data Element Type
    - Comptar un DET per cada camp que composi el Fitxer i que sigui reconeixible per l'usuari.
    - Comptar un DET per cada Foreign Key cap un altre Fitxer.
    - Un únic camp lògic, emmagatzemant en diversos camps físics (per tècniques d'implementació, compta com un únic DET.
    - Els camps no reconeguts per l'usuari (inclosos per tècniques d'implementació) no compten com a DETs.
    - Exemple: Atributs de les classes d'objectes.

6 Extensió 3D dels Punts Funció (III)

- Record Element Type:
  - Comptar un RET per cada subgrup de dades elementals reconeixibles per l'usuari i per cada subgrup dels següents que apareguin.
  - Subgrup Opcional: aquells que l'usuari pot decidir usar o no. Ex: Camps opcionals d'un formulari.
  - Subgrup Obligatori: aquell que ha d'usar necessariament.
  - S'intenta comptar les formes de "donar d'alta" un objecte.

6 Extensió 3D dels Punts Funció (IV)

- Un cop identificats DETs i RETs de ELFs i ILFs, emprar la següent taula:

	1 a 19 DETs	**20 a 50 DETs**	**51 o més**
1 RET	Baixa	Baixa	Mitjana
2 a 5 RETs	Baixa	Mitjana	Alta
6 ó + RETs	Mitjana	Alta	Alta

Extensió 3D dels Punts Funció (V)

- Per a EI
  - Data Element Type:
    - Comptar un DET per cada camp que creua la frontera de l'aplicació (camps d'un formulari).
    - No comptar els camps recuperats o derivats pel sistema i emmagatzemats en un ILF si aquest no creua la frontera de l'aplicació.
    - Comptar un DET per cadascun dels següents:
  - Definició de l'inici de l'execució del procés que processa l'entrada (Acceptar, Enter, Submit). Si hi ha més d'una forma, comptar-ne només 1.
  - Missatges d'error que puguis generar-se durant el procés d'execució (un DET per a tots ells).
  - File Referenced Type:
    - Comptar +1 per cada Fitxer Lògic (Intern o Extern) llegit/actualitzat per l'entrada.

Extensió 3D dels Punts Funció (VI)

- Un cop identificats DETs i FRTs de les EIs, emprar la següent taula:

	1 a 4 DETs	5 a 15 DETs	16 o més
1 FRT	Baixa	Baixa	Mitjana
2FRTs	Baixa	Mitjana	Alta
3 ó + FRTs	Mitjana	Alta	Alta

Extensió 3D dels Punts Funció (VII)

- Per a EO i EQ:
  - Data Element Type:
    - Comptar +1 per cada camp que creua la frontera de l'aplicació.
    - Comptar +1 per cada camp reconegut per l'usuari, no repetit, que entra a l'aplicació i que es requereix per especificar quan, què o com han de ser recuperades les dades.
    - Si una dada entra i surt de l'aplicació, comptar-a només una vegada.
    - Comptar +1 pel conjunt dels missatges d'error que puguin sorgir.
    - No comptar les dades recuperades ni derivades del sistema si no han creuat la frontera de l'aplicació.
  - File Referenced Type:
    - Comptar +1 per cada Fitxer Lògic (Intern o Extern) llegit/actualitzat per l'entrada.

Extensió 3D dels Punts Funció (VIII)

- Un cop identificats DETs i FRTs de les EO i/o EQ, emprar la següent taula:

	1 a 5 DETs	**6 a 19 DETs**	**20 o més**
1FRT	Baixa	Baixa	Mitjana
2 a 3 FRTs	Baixa	Mitjana	Alta
4 ó + FRTs	Mitjana	Alta	Alta

Productivitat en hores per punt de funció

Activitat	Descripció	Hores/FP	Ponderació (%)	Total hores/FP
01	Requeriments	0,75	5	3,75
04	Pla del projecte	0,26	2	0,52
05	Disseny inicial	0,75	8	6,00
06	Disseny detallat	0,88	10	8,80
08	Codificació	2,64	40	105,60
15	Documentació d'usuari	1,89	10	18,90
16	Prova unitària	0,88	10	8,80
17	Prova funcional	0,88	4	3,52
18	Prova d'integració	0,75	4	3,00
19	Prova del sistema	0,66	4	2,64
25	Gestió del projecte	1,32	3	3,96

Mitjana total hores/FP = 1,65 (valor optimista, realitat 2 o 3 h./FP)

Relació punts funció i línes de codi

Mesurant milers de projectes informàtics s'ha calculat:

Llenguatge	**LOC/FP**
Assemblador	320
Macroassemblador	213
C	150
COBOL	106
Fortran	106
Pascal	91
RPG	80
PL/I	80
Ada	71
BASIC	64

Exercicis

1. És cert que l'estimació inicial de costos d'un projecte informàtic amb mètriques basades en les funcionalitats (punts de funció d'Albrecht, per exemple) és més objectiva que l'estimació efectuada amb mètriques basades en la grandària del projecte (KLOC, per exemple)?
2. Quan un projecte informàtic es retarda amb relació a la seva planificació inicial, es pot dir que la causa és sempre un defecte en l'estimació de costos?
3. Què s'inclou quan es parla de mesurar la grandària d'un projecte informàtic en línies de codi?
4. Per un mateix projecte, el nombre de línies de codi escrites en COBOL és més gran o més petit que les escrites en un llenguatge RAD (Rapid Application Developement)?
5. Com es mesuren els punts de funció d'Albrecht?